**GoldenGate**

**GoldenGate Tutorial 1 – Concepts and Architecture**

GoldenGate enables us to extract and replicate data across a variety of topologies as shown the diagram below as well as the exchange and manipulation of data at the transactional level between a variety of database platforms like Oracle, DB2, SQL Server, Ingres, MySQL etc.

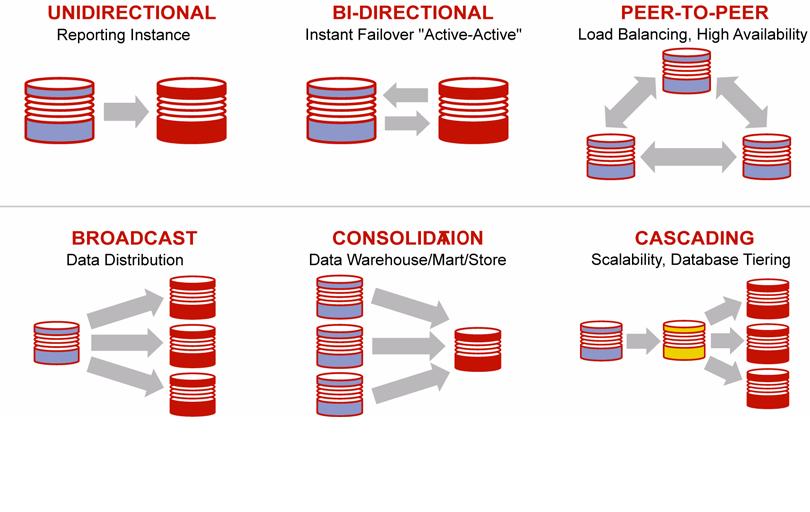
It can support a number of different business requirements like:

 Business Continuity and High Availablity

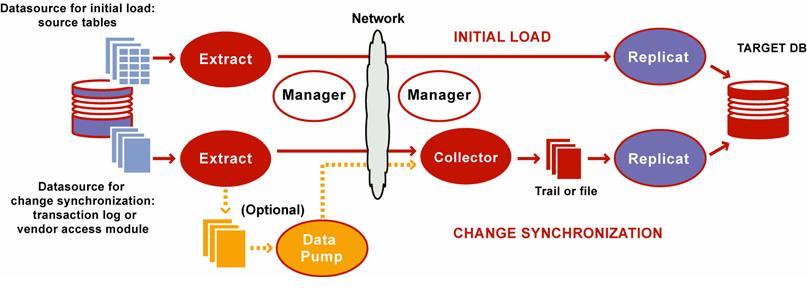
 Data migrations and upgrades

 Decision Support Systems and Data Warehousing

 Data integration and consolidation



Let us know look at the differents components and processes that make up a typical GoldenGate configuration on Oracle.

  
(*source: Oracle GoldenGate Administration Guide*)

**Manager**

The Manager process must be running on both the source as well as target systems before the Extract or Replicat process can be started and performs a number of functions including monitoring and starting other GoldenGate processes, managing the trail files and also reporting.

**Extract**

The Extract process runs on the source system and is the data caoture mechanism of GoldenGate. It can be configured both for initial loading of the source data as well as to synchronize the changed data on the source with the target. This can be configued to also propagate any DDL changes on those databases where DDL change support is available.

**Replicat**

The Replicat process runs on the target system and reads transactional data changes as well as DDL changes and replicates then to the target database. Like the Extract process, the Replicat process can also be configured for Initial Load as well as Change Synchronization.

**Collector**

The Collector is a background process which runs on the target system and is started automatically by the Manager (Dynamic Collector) or it can be configured to stsrt manually (Static Collector). It receives extracted data changes that are sent via TCP/IP and writes then to the trail files from where they are processed by the Replicat process.

**Trails**

Trails are series of files that GoldenGate temporarily stores on disks and these files are written to and read from by the Extract and Replicat processes as the case may be. Depending on the configuration chosen, these trail files can exist on the source as well as on the target systems. If it exists on the local system, it will be known an Extract Trail or as an Remote Trail if it exists on the target system.

**Data Pumps**

Data Pumps are secondary extract mechanisms which exist in the source configuration. This is optional component and if Data Pump is not used then Extract sends data via TCP/IP to the remote trail on the target. When Data Pump is configured, the the Primary Extract process will write to the Local Trail and then this trail is read by the Data Pump and data is sent over the network to Remote Trails on the target system.

In the absence of Data Pump, the data that the Extract process extracts resides in memory alone and there is no storage of this data anywhere on the source system. In case of network of target failures, there could be cases where the primary extract process can abort or abend. Data Pump can also be useful in those cases where we are doing complex filtering and transformation of data as well as when we are consolidating data from many sources to a central target.

**Data source**

When processing transactional data changes, the Extract process can obtain data directly from the database transaction logs (Oracle, DB2, SQL Server, MySQL etc) or from a GoldenGate Vendor Access Module (VAM) where the database vendor (for example Teradata) will provide the required components that will be used by Extract to extract the data changes.

**Groups**

To differentiate between the number of different Extract and Replicat groups which can potentially co-exist on a system, we can define processing groups. For instance, if we want to replicate different sets of data in parallel, we can create two Replicat groups.

A processing group consisits of a process which could be either a Extract or Replicat process, a corresponding parameter file, checkpoint file or checkpoint table (for Replicat) and other files which could be associated with the process.

# GoldenGate Tutorial 2 – Installation (Oracle 11g on Linux)

This example will illustrate the installation of Oracle GoldenGate on an RHEL 5 platform. We had in an earlier post discussed the architecture and various components of a GoldenGate environment.

Select the Product Pack “Oracle Fusion Middleware” and the platform Linux X86-64.

Then select “Oracle GoldenGate on Oracle Media Pack for Linux x86-64″ and since we are installing this for an Oracle 11g database, we download “Oracle GoldenGate V10.4.0.x for Oracle 11g 64bit on RedHat 5.0″

$ **unzip V18159-01.zip**  
Archive: V18159-01.zip  
inflating: ggs\_redhatAS50\_x64\_ora11g\_64bit\_v10.4.0.19\_002.tar

$**tar -xvof ggs\_redhatAS50\_x64\_ora11g\_64bit\_v10.4.0.19\_002.tar**

$ **export PATH=$PATH:/u01/oracle/ggs**

$ **export LD\_LIBRARY\_PATH=$ORACLE\_HOME/lib:/u01/oracle/ggs**

$ **ggsci**

Oracle GoldenGate Command Interpreter for Oracle  
Version 10.4.0.19 Build 002  
Linux, x64, 64bit (optimized), Oracle 11 on Sep 17 2009 23:51:28

Copyright (C) 1995, 2009, Oracle and/or its affiliates. All rights reserved.

GGSCI (redhat346.localdomain) 1>

GGSCI (redhat346.localdomain) 1> **CREATE SUBDIRS**

Creating subdirectories under current directory /u01/app/oracle/product/11.2.0/dbhome\_1

Parameter files /u01/oracle/ggs/dirprm: created  
Report files /u01/oracle/ggs/dirrpt: created  
Checkpoint files /u01/oracle/ggs/dirchk: created  
Process status files /u01/oracle/ggs/dirpcs: created  
SQL script files /u01/oracle/ggs/dirsql: created  
Database definitions files /u01/oracle/ggs/dirdef: created  
Extract data files /u01/oracle/ggs/dirdat: created  
Temporary files /u01/oracle/ggs/dirtmp: created  
Veridata files /u01/oracle/ggs/dirver: created  
Veridata Lock files /u01/oracle/ggs/dirver/lock: created  
Veridata Out-Of-Sync files /u01/oracle/ggs/dirver/oos: created  
Veridata Out-Of-Sync XML files /u01/oracle/ggs/dirver/oosxml: created  
Veridata Parameter files /u01/oracle/ggs/dirver/params: created  
Veridata Report files /u01/oracle/ggs/dirver/report: created  
Veridata Status files /u01/oracle/ggs/dirver/status: created  
Veridata Trace files /u01/oracle/ggs/dirver/trace: created  
Stdout files /u01/oracle/ggs/dirout: created

We then need to create a database user which will be used by the GoldenGate Manager, Extract and Replicat processes. We can create individual users for each process or configure just a common user – in our case we will create the one user GGS\_OWNER and grant it the required privileges.

SQL> **create tablespace ggs\_data**  
2 **datafile ‘/u02/oradata/gavin/ggs\_data01.dbf’ size 200m;**

SQL> **create user ggs\_owner identified by ggs\_owner**  
2 **default tablespace ggs\_data**  
3 **temporary tablespace temp**;

User created.

SQL> **grant connect,resource to ggs\_owner**;

Grant succeeded.

SQL> **grant select any dictionary, select any table to ggs\_owner**;

Grant succeeded.

SQL> **grant create table to ggs\_owner**;

Grant succeeded.

SQL> **grant flashback any table to ggs\_owner**;

Grant succeeded.

SQL> **grant execute on dbms\_flashback to ggs\_owner**;

Grant succeeded.

SQL> **grant execute on utl\_file to ggs\_owner;**

Grant succeeded.

We can then confirm that the GoldenGate user we have just created is able to connect to the Oracle database

$ **ggsci**

Oracle GoldenGate Command Interpreter for Oracle  
Version 10.4.0.19 Build 002  
AIX 5L, ppc, 64bit (optimized), Oracle 11 on Sep 17 2009 23:54:16

Copyright (C) 1995, 2009, Oracle and/or its affiliates. All rights reserved.

GGSCI (devu007) 1> **DBLOGIN USERID ggs\_owner, PASSWORD ggs\_owner**  
Successfully logged into database.

We also need to enable supplemental logging at the database level otherwise we will get this error when we try to start the Extract process –

2010-02-08 13:51:21 GGS ERROR 190 No minimum supplemental logging is enabled. This may cause extract process to handle key update incorrectly if key  
column is not in first row piece.

2010-02-08 13:51:21 GGS ERROR 190 PROCESS ABENDING.

SQL> **ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS**;

Database altered

# GoldenGate Tutorial 3 – Configuring the Manager process

The Oracle GoldenGate Manager performs a number of functions like starting the other GoldenGate processes, trail log file management and reporting.

The Manager process needs to be configured on both the source as well as target systems and configuration is carried out via a parameter file just as in the case of the other GoldenGate processes like Extract and Replicat.

After installation of the software, we launch the GoldenGate Software Command Interface (GGSCI) and issue the following command to edit the Manager parameter file

**EDIT PARAMS MGR**

The only mandatory parameter that we need to specify is the PORT which defines the port on the local system where the manager process is running. The default port is 7809 and we can either specify the default port or some other port provided the port is available and not restricted in any way.

Some other recommended optional parameters are AUTOSTART which which automatically start the Extract and Replicat processes when the Manager starts.

The USERID and PASSWORD parameter and required if you enable GoldenGate DDL support and this is the Oracle user account that we created for the Manager(and Extract/Replicat) as described in the earlier tutorial.

The Manager process can also clean up trail files from disk when GoldenGate has finished processing them via the PURGEOLDEXTRACTS parameter. Used with the USECHECKPOINTS clause, it will ensure that until all processes have fnished using the data contained in the trail files, they will not be deleted.

The following is an example of a manager parameter file

[oracle@redhat346 ggs]$ **./ggsci**

Oracle GoldenGate Command Interpreter for Oracle  
Version 10.4.0.19 Build 002  
Linux, x64, 64bit (optimized), Oracle 11 on Sep 17 2009 23:51:28

Copyright (C) 1995, 2009, Oracle and/or its affiliates. All rights reserved.

GGSCI 2> **EDIT PARAMS MGR**

**PORT** 7809  
**USERID** ggs\_owner, **PASSWORD** ggs\_owner  
**PURGEOLDEXTRACTS**/u01/oracle/ggs/dirdat/ex, **USECHECKPOINTS**

The manager can be stopped and started via the GSSCI commands **START MANAGER**and **STOP MANAGER**.

Information on the status of the Manager can be obtained via the **INFO MANAGER** command

GGSCI (devu007) 4> **info manager**

Manager is running (IP port devu007.7809).

# Oracle GoldenGate Tutorial 4 – performing initial data load

This example illustrates using the **GoldenGate direct load** method to extract records from an Oracle 11g database on Red Hat Linux platform and load the same into an Oracle 11g target database on an AIX platform.

The table PRODUCTS in the SH schema on the source has 72 rows and on the target database the same table is present only in structure without any data. We will be loading the 72 rows in this example from the source database to the target database using GoldenGate Direct Load method.

**On Source**

1) **Create the Initial data extract process ‘load1′**

GGSCI (redhat346.localdomain) 5> **ADD EXTRACT load1, SOURCEISTABLE**  
EXTRACT added.

Since this is a one time data extract task, the source of the data is not the transaction log files of the RDBMS (in this case the online and archive redo log files) but the table data itself, that is why the keyword SOURCEISTABLE is used.

2) **Create the parameter file for the extract group load1**

**EXTRACT**: name of the extract group  
**USERID/PASSWORD**: the database user which has been configured earlier for Extract ( this user is created in the source database)  
**RMTHOST**: This will be the IP address or hostname of the target system  
**MGRPORT**: the port where the Manager process is running  
**TABLE**: specify the table which is being extracted and replicated. This can be specified in a number of ways using wildcard characters to include or exclude tables as well as entire schemas.

GGSCI (redhat346.localdomain) 6> **EDIT PARAMS load1**

EXTRACT load1  
USERID ggs\_owner, PASSWORD ggs\_owner  
RMTHOST devu007, MGRPORT 7809  
RMTTASK replicat, GROUP load2  
TABLE sh.products;

**On Target**

3) **Create the initial data load task ‘load2′**

Since this is a one time data load task, we are using the keyword SPECIALRUN

GGSCI (devu007) 1> **ADD REPLICAT load2, SPECIALRUN**  
REPLICAT added.

4) **Create the parameter file for the Replicat group, load2**

REPLICAT: name of the Replicat group created for the initial data load  
USERID/PASSWORD: database credentials for the Replicat user (this user is created in the target database)  
ASSUMETARGETDEFS: this means that the source table structure exactly matches the target database table structure  
MAP: with GoldenGate we can have the target database structure entirely differ from that of the source in terms of table names as well as the column definitions of the tables. This parameter provides us the mapping of the source and target tables which is same in this case

GGSCI (devu007) 2> **EDIT PARAMS load2**

“/u01/oracle/software/goldengate/dirprm/rep4.prm” [New file]

REPLICAT load2  
USERID ggs\_owner, PASSWORD ggs\_owner  
ASSUMETARGETDEFS  
MAP sh.customers, TARGET sh.customers;

**On Source**

SQL> **select count(\*) from products**;

COUNT(\*)  
———-  
72

**On Target**

SQL> **select count(\*) from products;**

COUNT(\*)  
———-  
0

**On Source**

5) **Start the initial load data extract task on the source system**

We now start the initial data load task load 1 on the source. Since this is a one time task, we will initially see that the extract process is runningand after the data load is complete it will be stopped. We do not have to manually start the Replicat process on the target as that is done when the Extract task is started on the source system.

**On Source**

GGSCI (redhat346.localdomain) 16> **START EXTRACT load1**

Sending START request to MANAGER …  
EXTRACT LOAD1 starting

GGSCI (redhat346.localdomain) 28> **info extract load1**

EXTRACT LOAD1 Last Started 2010-02-11 11:33 Status RUNNING  
Checkpoint Lag Not Available  
Log Read Checkpoint Table SH.PRODUCTS  
2010-02-11 11:33:16 Record 72  
Task SOURCEISTABLE

GGSCI (redhat346.localdomain) 29> **info extract load1**

EXTRACT LOAD1 Last Started 2010-02-11 11:33 Status STOPPED  
Checkpoint Lag Not Available  
Log Read Checkpoint Table SH.PRODUCTS  
2010-02-11 11:33:16 **Record 72**  
Task SOURCEISTABLE

**On Target**

SQL> **select count(\*) from products;**

COUNT(\*)  
———-  
72

**Oracle GoldenGate Tutorial 5 – configuring online change synchronization**

In our earlier tutorial, we examined how to create a GoldenGate environment for initial data capture and load.

In this tutorial, we will see how by using GoldenGate change synchronization, changes that occur on the source (Oracle 11g on Linux) are applied near real time on the target (Oracle 11g on AIX). The table on the source is the EMP table in SCOTT schema which is being replicated to the EMP table in the target database SH schema.

These are the steps that we will take:

Create a GoldenGate Checkpoint table  
Create an Extract group  
Create a parameter file for the online Extract group  
Create a Trail  
Create a Replicat group  
Create a parameter file for the online Replicat group

**Create the GoldenGate Checkpoint table**

GoldenGate maintains its own Checkpoints which is a known position in the trail file from where the Replicat process will start processing after any kind of error or shutdown. This ensures data integrity and a record of these checkpoints is either maintained in files stored on disk or table in the database which is the preferred option.

We can also create a single Checkpoint table which can used by all Replicat groups from the single or many GoldenGate instances.

In one of the earlier tutorials we had created the GLOBALS file. We now need to edit that GLOBALS file and add an entry for **CHECKPOINTTABLE** which will include the checkpoint table name which will be available to all Replicat processes via the EDIT PARAMS command.

GGSCI (devu007) 2> **EDIT PARAMS ./GLOBALS**

GGSCHEMA GGS\_OWNER  
**CHECKPOINTTABLE GGS\_OWNER.CHKPTAB**

GGSCI (devu007) 4> **DBLOGIN USERID ggs\_owner, PASSWORD ggs\_owner**  
Successfully logged into database.

GGSCI (devu007) 6> **ADD CHECKPOINTTABLE GGS\_OWNER.CHKPTAB**

Successfully created checkpoint table GGS\_OWNER.CHKPTAB.

apex:/u01/oracle/software/goldengate> **sqlplus ggs\_owner/ggs\_owner**

SQL\*Plus: Release 11.1.0.6.0 - Production on Mon Feb 8 09:02:19 2010

Copyright (c) 1982, 2007, Oracle. All rights reserved.

Connected to:

Oracle Database 11g Enterprise Edition Release 11.1.0.6.0 - 64bit Production

With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> **desc chkptab**

Name Null? Type

----------------------------------------- -------- ----------------------------

GROUP\_NAME NOT NULL VARCHAR2(8)

GROUP\_KEY NOT NULL NUMBER(19)

SEQNO NUMBER(10)

RBA NOT NULL NUMBER(19)

AUDIT\_TS VARCHAR2(29)

CREATE\_TS NOT NULL DATE

LAST\_UPDATE\_TS NOT NULL DATE

CURRENT\_DIR NOT NULL VARCHAR2(255)

**Create the Online Extract Group**

GGSCI (redhat346.localdomain) 1> **ADD EXTRACT ext1, TRANLOG, BEGIN NOW**  
EXTRACT added.

**Create the Trail**

We now create a trail – note that this path pertains to the *GoldenGate software location on the target system* and this is where the trail files will be created **having a prefix ‘rt’**which will be used by the Replicat process also running on the target system

GGSCI (redhat346.localdomain) 2> **ADD RMTTRAIL /u01/oracle/software/goldengate/dirdat/rt, EXTRACT ext1**  
RMTTRAIL added.

**Create a parameter file for the online Extract group ext1**

GGSCI (redhat346.localdomain) 3> **EDIT PARAMS ext1**

EXTRACT ext1  
USERID ggs\_owner, PASSWORD ggs\_owner  
RMTHOST devu007, MGRPORT 7809  
RMTTRAIL /u01/oracle/software/goldengate/dirdat/rt  
TABLE scott.emp;

**ON TARGET SYSTEM**

**Create the online Replicat group**

GGSCI (devu007) 7> **ADD REPLICAT rep1, EXTTRAIL /u01/oracle/software/goldengate/dirdat/rt**  
REPLICAT added.

Note that the EXTTRAIL location which is on the target local system conforms to the RMTTRAIL parameter which we used when we created the parameter file for the extract process on the source system.

**Create a parameter file for the online Replicat group, rep1**

GGSCI (devu007) 8> **EDIT PARAMS rep1**

REPLICAT rep1  
ASSUMETARGETDEFS  
USERID ggs\_owner, PASSWORD ggs\_owner  
MAP scott.emp, TARGET sh.emp;

**ON SOURCE**

**Start the Extract process**

GGSCI (redhat346.localdomain) 16> **START EXTRACT ext1**

Sending START request to MANAGER …  
EXTRACT EXT1 starting

GGSCI (redhat346.localdomain) 17> **STATUS EXTRACT ext1**  
EXTRACT EXT1: RUNNING

GGSCI (redhat346.localdomain) 16> **INFO EXTRACT ext1**

EXTRACT EXT1 Last Started 2010-02-08 14:27 Status RUNNING  
Checkpoint Lag 00:00:00 (updated 00:00:09 ago)  
Log Read Checkpoint Oracle Redo Logs  
2010-02-08 14:27:48 Seqno 145, RBA 724480

**ON TARGET**

**Start the Replicat process**

GGSCI (devu007) 1> **START REPLICAT rep1**  
Sending START request to MANAGER …  
REPLICAT REP1 starting

GGSCI (devu007) 2> **INFO REPLICAT rep1**

REPLICAT REP1 Last Started 2010-02-08 14:55 Status RUNNING  
Checkpoint Lag 00:00:00 (updated 00:00:01 ago)  
Log Read Checkpoint File **/u01/oracle/software/goldengate/dirdat/rt000001**  
2010-02-08 14:27:57.600425 RBA 1045

Note: the trail file has a prefix of ‘rt’ (which we had defined earlier)

**LET US NOW TEST …**

**ON SOURCE**

SQL> conn scott/tiger  
Connected.

SQL> **UPDATE emp SET sal=9999 WHERE ename=’KING';**

1 row updated.

SQL> COMMIT;

Commit complete.

**ON TARGET**

SQL> **SELECT SAL FROM emp WHERE ename=’KING';**

SAL  
———-  
**9999**

# Oracle GoldenGate Tutorial 6 – configuring Data Pump process

The Data Pump (not to be confused with the Oracle Export Import Data Pump) is an optional secondary Extract group that is created on the source system. When Data Pump is not used, the Extract process writes to a remote trail that is located on the target system using TCP/IP. When Data Pump is configured, the Extract process writes to a local trail and from here Data Pump will read the trail and write the data over the network to the remote trail located on the target system.

The advantages of this can be seen as it protects against a network failure as in the absence of a storage device on the local system, the Extract process writes data into memory before the same is sent over the network. Any failures in the network could then cause the Extract process to abort (abend). Also if we are doing any complex data transformation or filtering, the same can be performed by the Data Pump. It will also be useful when we are consolidating data from several sources into one central target where data pump on each individual source system can write to one common trail file on the target.

**Create the Extract process**

GGSCI (devu007) 1> **ADD EXTRACT ext1, TRANLOG, BEGIN NOW**  
EXTRACT added.

**Create a local trail**

Using the ADD EXTRAIL command we will now create a local trail on the source system where the Extract process will write to and which is then read by the Data Pump process. We will link this local trail to the Primary Extract group we just created, ext1

GGSCI (devu007) 3> **ADD EXTTRAIL /u01/oracle/software/goldengate/dirdat/lt, EXTRACT ext1**  
EXTTRAIL added.

**Create the Data Pump group**

On the source system create the Data Pump group and using the EXTTRAILSOURCE keywork specify the location of the local trail which will be read by the Data Pump process

GGSCI (devu007) 4> **ADD EXTRACT dpump, EXTTRAILSOURCE /u01/oracle/software/goldengate/dirdat/lt**  
EXTRACT added.

**Create the parameter file for the Primary Extract group**

GGSCI (devu007) 5>**EDIT PARAMS ext1**

“/u01/oracle/software/goldengate/dirprm/ext1.prm” [New file]

EXTRACT ext1  
USERID ggs\_owner, PASSWORD ggs\_owner  
EXTTRAIL /u01/oracle/software/goldengate/dirdat/lt  
TABLE MONITOR.WORK\_PLAN;

**Specify the location of the remote trail on the target system**

Use the RMTTRAIL to specify the location of the remote trail and associate the same with the Data Pump group as it will be wriiten to over the network by the data pump process

GGSCI (devu007) 6> **ADD RMTTRAIL /u01/oracle/ggs/dirdat/rt, EXTRACT dpump**  
RMTTRAIL added.

**Create the parameter file for the Data Pump group**

Note- the parameter PASSTHRU signifies the mode being used for the Data Pump which means that the names of the source and target objects are identical and no column mapping or filtering is being performed here.

GGSCI (devu007) 2> **EDIT PARAMS dpump**

“/u01/oracle/software/goldengate/dirprm/dpump.prm” [New file]

EXTRACT dpump  
USERID ggs\_owner, PASSWORD ggs\_owner  
RMTHOST redhat346, MGRPORT 7809  
RMTTRAIL /u01/oracle/ggs/dirdat/rt  
PASSTHRU  
TABLE MONITOR.WORK\_PLAN;

**ON TARGET SYSTEM**

**Create the Replicat group**

The EXTTRAIL clause indicates the location of the remote trail and should be the same as the RMTTRAIL value that was used when creating the Data Pump process on the source system.

GGSCI (redhat346.localdomain) 2> **ADD REPLICAT rep1, EXTTRAIL /u01/oracle/ggs/dirdat/rt**  
REPLICAT added.

**Create the parameter file for the Replicat group**

GGSCI (redhat346.localdomain) 3> **EDIT PARAMS rep1**

REPLICAT rep1  
ASSUMETARGETDEFS  
USERID ggs\_owner, PASSWORD ggs\_owner  
MAP MONITOR.WORK\_PLAN, TARGET MONITOR.WORK\_PLAN;

**ON SOURCE**

**On the source system, now start the Extract and Data Pump processes**.

GGSCI (devu007) 3> **START EXTRACT ext1**

Sending START request to MANAGER …  
EXTRACT EXT1 starting

GGSCI (devu007) 4> **START EXTRACT dpump**

Sending START request to MANAGER …  
EXTRACT DPUMP starting

GGSCI (devu007) 5> **info extract ext1**

EXTRACT EXT1 Last Started 2010-02-18 11:23 Status RUNNING  
Checkpoint Lag 00:40:52 (updated 00:00:09 ago)  
Log Read Checkpoint Oracle Redo Logs  
2010-02-18 10:42:19 Seqno 761, RBA 15086096

GGSCI (devu007) 6> **INFO EXTRACT dpump**

EXTRACT DPUMP Last Started 2010-02-18 11:23 Status RUNNING  
Checkpoint Lag 00:00:00 (updated 00:00:02 ago)  
Log Read Checkpoint File /u01/oracle/software/goldengate/dirdat/lt000000  
2010-02-18 11:15:10.000000 RBA 5403

**Note**- the data pump process is reading from the Local Trail file – /u01/oracle/software/goldengate/dirdat/lt000000

**ON TARGET SYSTEM**

**Start the Replicat process**

GGSCI (redhat346.localdomain) 4> **START REPLICAT rep1**

Sending START request to MANAGER …  
REPLICAT REP1 starting

GGSCI (redhat346.localdomain) 5> **STATUS REPLICAT rep1**  
REPLICAT REP1: RUNNING

# Oracle GoldenGate Tutorial 7 – configuring DDL synchronization

In addition to providing replication support for all DML statements, we can also configure the GoldenGate environment to provide DDL support as well.

A number of prerequisite setup tasks need to be performed which we willl highlight here.

**Run the following scripts from the directory where the GoldenGate software was installed**.

The assumption here is that the database user GGS\_OWNER has already been created and granted the required roles and privileges as discussed in our earlier tutorial.

Note - run the scripts as SYSDBA

SQL> **@marker\_setup**

Marker setup script

You will be prompted for the name of a schema for the GoldenGate database objects.

NOTE: The schema must be created prior to running this script.

NOTE: Stop all DDL replication before starting this installation.

Enter GoldenGate schema name:**GGS\_OWNER**

Marker setup table script complete, running verification script...

Please enter the name of a schema for the GoldenGate database objects:

Setting schema name to GGS\_OWNER

MARKER TABLE

-------------------------------

OK

MARKER SEQUENCE

-------------------------------

OK

Script complete.

SQL> **alter session set recyclebin=OFF;**

Session altered.

SQL> **@ddl\_setup**

GoldenGate DDL Replication setup script

Verifying that current user has privileges to install DDL Replication...

You will be prompted for the name of a schema for the GoldenGate database objects.

NOTE: The schema must be created prior to running this script.

NOTE: On Oracle 10g and up, system recycle bin must be disabled.

NOTE: Stop all DDL replication before starting this installation.

Enter GoldenGate schema name:GGS\_OWNER

You will be prompted for the mode of installation.

To install or reinstall DDL replication, enter INITIALSETUP

To upgrade DDL replication, enter NORMAL

Enter mode of installation:INITIALSETUP

Working, please wait ...

Spooling to file ddl\_setup\_spool.txt

Using GGS\_OWNER as a GoldenGate schema name, INITIALSETUP as a mode of installation.

Working, please wait ...

RECYCLEBIN must be empty.

This installation will purge RECYCLEBIN for all users.

To proceed, enter yes. To stop installation, enter no.

Enter yes or no:yes

DDL replication setup script complete, running verification script...

Please enter the name of a schema for the GoldenGate database objects:

Setting schema name to GGS\_OWNER

DDLORA\_GETTABLESPACESIZE STATUS:

Line/pos Error

---------- -----------------------------------------------------------------

No errors No errors

CLEAR\_TRACE STATUS:

Line/pos Error

---------- -----------------------------------------------------------------

No errors No errors

CREATE\_TRACE STATUS:

Line/pos Error

---------- -----------------------------------------------------------------

No errors No errors

TRACE\_PUT\_LINE STATUS:

Line/pos Error

---------- -----------------------------------------------------------------

No errors No errors

INITIAL\_SETUP STATUS:

Line/pos Error

---------- -----------------------------------------------------------------

No errors No errors

DDLVERSIONSPECIFIC PACKAGE STATUS:

Line/pos Error

---------- -----------------------------------------------------------------

No errors No errors

DDLREPLICATION PACKAGE STATUS:

Line/pos Error

---------- -----------------------------------------------------------------

No errors No errors

DDLREPLICATION PACKAGE BODY STATUS:

Line/pos Error

---------- -----------------------------------------------------------------

No errors No errors

DDL HISTORY TABLE

-----------------------------------

OK

DDL HISTORY TABLE(1)

-----------------------------------

OK

DDL DUMP TABLES

-----------------------------------

OK

DDL DUMP COLUMNS

-----------------------------------

OK

DDL DUMP LOG GROUPS

-----------------------------------

OK

DDL DUMP PARTITIONS

-----------------------------------

OK

DDL DUMP PRIMARY KEYS

-----------------------------------

OK

DDL SEQUENCE

-----------------------------------

OK

GGS\_TEMP\_COLS

-----------------------------------

OK

GGS\_TEMP\_UK

-----------------------------------

OK

DDL TRIGGER CODE STATUS:

Line/pos Error

---------- -----------------------------------------------------------------

No errors No errors

DDL TRIGGER INSTALL STATUS

-----------------------------------

OK

DDL TRIGGER RUNNING STATUS

-----------------------------------

ENABLED

STAYMETADATA IN TRIGGER

-----------------------------------

OFF

DDL TRIGGER SQL TRACING

-----------------------------------

0

DDL TRIGGER TRACE LEVEL

-----------------------------------

0

LOCATION OF DDL TRACE FILE

--------------------------------------------------------------------------------

/u01/app/oracle/diag/rdbms/gavin/gavin/trace/ggs\_ddl\_trace.log

Analyzing installation status...

STATUS OF DDL REPLICATION

--------------------------------------------------------------------------------

SUCCESSFUL installation of DDL Replication software components

Script complete.

SQL>

SQL> **@role\_setup**

GGS Role setup script

This script will drop and recreate the role GGS\_GGSUSER\_ROLE

To use a different role name, quit this script and then edit the params.sql script to change

the gg\_role parameter to the preferred name. (Do not run the script.)

You will be prompted for the name of a schema for the GoldenGate database objects.

NOTE: The schema must be created prior to running this script.

NOTE: Stop all DDL replication before starting this installation.

Enter GoldenGate schema name:GGS\_OWNER

Wrote file role\_setup\_set.txt

PL/SQL procedure successfully completed.

Role setup script complete

Grant this role to each user assigned to the Extract, GGSCI, and Manager processes, by using the following SQL command:

GRANT GGS\_GGSUSER\_ROLE TO

where is the user assigned to the GoldenGate processes.

SQL> **grant ggs\_ggsuser\_role to ggs\_owner**;

Grant succeeded.

SQL> **@ddl\_enable**

Trigger altered.

SQL> **@ddl\_pin GGS\_OWNER**

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

**Turn Recyclebin OFF**

We need to set the parameter recyclebin to OFF via the ALTER SYSTEM SET RECYCLEBIN=OFF command in order to prevent this error which we will see if we try and configure DDL support and then start the Extract process.

2010-02-19 11:13:30 GGS ERROR 2003 RECYCLEBIN must be turned off. For 10gr2 and up, set RECYCLEBIN in parameter file to OFF. For 10gr1, set \_RECYCLEBI  
N in parameter file to FALSE. Then restart database and extract.  
2010-02-19 11:13:30 GGS ERROR 190 PROCESS ABENDING.

**Enable additional logging at the table level**

Note- We had earlier enabled additional supplemental logging at the database level. Using the ADD TRANDATA command we now enable it at even the table level as this is required by GoldenGate for DDL support.

GGSCI (redhat346.localdomain) 5> **DBLOGIN USERID ggs\_owner, PASSWORD ggs\_owner**  
Successfully logged into database.

GGSCI (redhat346.localdomain) 6> **ADD TRANDATA scott.emp**

Logging of supplemental redo data enabled for table SCOTT.EMP.

**Edit the parameter file for the Extract process to enable DDL synchronization**

We had earlier created a parameter file for an Extract process ext1. We now edit that parameter file and add the entry  
**DDL INCLUDE MAPPED**

This means that DDL support is now enabled for all tables which have been mapped and in this case it will only apply to the SCOTT.EMP table as that is the only table which is being processed here. We can also use the INCLUDE ALL or EXCLUDE ALL or wildcard characters to specify which tables to enable the DDL support for.

GGSCI (redhat346.localdomain) 1> **EDIT PARAM EXT1**

EXTRACT ext1  
USERID ggs\_owner, PASSWORD ggs\_owner  
RMTHOST 10.53.100.100, MGRPORT 7809  
RMTTRAIL /u01/oracle/software/goldengate/dirdat/rt  
**DDL INCLUDE MAPPED**  
TABLE scott.emp;

**Test the same**

We will now alter the structure of the EMP table by adding a column and we can see that this new table structure is also reflected on the target system.

**On Source**

SQL> **ALTER TABLE EMP ADD NEW\_COL VARCHAR2(10);**  
Table altered.

**On Target**

SQL> desc emp  
Name Null? Type  
—————————————– ——– —————————-  
EMPNO NOT NULL NUMBER(4)  
ENAME VARCHAR2(10)  
JOB VARCHAR2(20)  
MGR NUMBER(4)  
HIREDATE DATE  
SAL NUMBER(7,2)  
COMM NUMBER(7,2)  
DEPTNO NUMBER(2)  
MYCOL VARCHAR2(10)  
**NEW\_COL VARCHAR2(10**)

# Oracle Goldengate Tutorial 8 – Filtering and Mapping data

Oracle GoldenGate not only provides us a replication solution that is Oracle version independent as well as platform independent, but we can also use it to do data transformation and data manipulation between the source and the target.

So we can use GoldenGate when the source and database database differ in table structure as well as an ETL tool in a Datawarehouse type environment.

We will discuss below two examples to demonstrate this feature – column mapping and filtering of data.

In example 1, we will filter the records that are extracted on the source and applied on the target – only rows where the JOB column value equals ‘MANAGER” in the MYEMP table will be considered for extraction.

In example 2, we will deal with a case where the table structure is different between the source database and the target database and see how column mapping is performed in such cases.

**Example 1**

**Initial load of all rows which match the filter from source to target**. The target database MYEMP table will only be populated with rows from the EMP table where filter criteria of JOB=’MANAGER’ is met.

**On Source**

GGSCI (redhat346.localdomain) 4> **add extract myload1, sourceistable**  
EXTRACT added.

GGSCI (redhat346.localdomain) 5> **edit params myload1**

EXTRACT myload1  
USERID ggs\_owner, PASSWORD ggs\_owner  
RMTHOST devu007, MGRPORT 7809  
RMTTASK replicat, GROUP myload1  
TABLE scott.myemp, FILTER (@STRFIND (job, “MANAGER”) > 0);

**On Target**

GGSCI (devu007) 2> **add replicat myload1, specialrun**  
REPLICAT added.

GGSCI (devu007) 3> **edit params myload1**

“/u01/oracle/software/goldengate/dirprm/myload1.prm” [New file]  
REPLICAT myload1  
USERID ggs\_owner, PASSWORD ggs\_owner  
ASSUMETARGETDEFS  
MAP scott.myemp, TARGET sh.myemp;

**On Source – start the initial load extract**

GGSCI (redhat346.localdomain) 6> **start extract myl**oad1

Sending START request to MANAGER …  
EXTRACT MYLOAD1 starting

**On SOURCE**

SQL> **select count(\*) from myemp;**

COUNT(\*)  
———-  
14

SQL> **select count(\*) from myemp where job=’MANAGER';**

COUNT(\*)  
———-  
9

**On TARGET**

SQL> **select count(\*) from myemp where job=’MANAGER';**

COUNT(\*)  
———-  
9

**Create an online change extract and replicat group using a Filter**

GGSCI (redhat346.localdomain) 10> **add extract myload2, tranlog, begin now**  
EXTRACT added.

GGSCI (redhat346.localdomain) 11> **add rmttrail /u01/oracle/software/goldengate/dirdat/bb, extract myload2**  
RMTTRAIL added.

GGSCI (redhat346.localdomain) 11> **edit params myload2**

EXTRACT myload2  
USERID ggs\_owner, PASSWORD ggs\_owner  
RMTHOST 10.53.200.225, MGRPORT 7809  
RMTTRAIL /u01/oracle/software/goldengate/dirdat/bb  
TABLE scott.myemp, **FILTER (@STRFIND (job, “MANAGER”) > 0**);

**On Target**

GGSCI (devu007) 2> add replicat myload2, exttrail /u01/oracle/software/goldengate/dirdat/bb  
REPLICAT added.

GGSCI (devu007) 3> edit params myload2

“/u01/oracle/software/goldengate/dirprm/myload2.prm” [New file]  
REPLICAT myload2  
ASSUMETARGETDEFS  
USERID ggs\_owner, PASSWORD ggs\_owner  
MAP scott.myemp, TARGET sh.myemp;

**On Source – start the online extract group**

GGSCI (redhat346.localdomain) 13> **start extract myload2**

Sending START request to MANAGER …  
EXTRACT MYLOAD2 starting

GGSCI (redhat346.localdomain) 14> **info extract myload2**

EXTRACT MYLOAD2 Last Started 2010-02-23 11:04 Status RUNNING  
Checkpoint Lag 00:27:39 (updated 00:00:08 ago)  
Log Read Checkpoint Oracle Redo Logs  
2010-02-23 10:36:51 Seqno 214, RBA 103988

**On Target**

GGSCI (devu007) 4> **start replicat**myload2

Sending START request to MANAGER …  
REPLICAT MYLOAD2 starting

GGSCI (devu007) 5> **info replicat myload2**

REPLICAT MYLOAD2 Last Started 2010-02-23 11:05 Status RUNNING  
Checkpoint Lag 00:00:00 (updated 00:00:08 ago)  
Log Read Checkpoint File /u01/oracle/software/goldengate/dirdat/bb000000  
First Record RBA 989

On Source we now insert two rows into the MYEMP table – one which has the JOB value of ‘MANAGER’ and the other row which has the job value of ‘SALESMAN’

**On SOURCE**

SQL> INSERT INTO MYEMP  
2 (empno,ename,job,sal)  
3 VALUES  
4 (1234,’GAVIN’,’**MANAGER**‘,10000);

1 row created.

SQL> commit;

Commit complete.

SQL> INSERT INTO MYEMP  
2 (empno,ename,job,sal)  
3 VALUES  
4 (1235,’BOB’,’**SALESMAN**‘,1000);

1 row created.

SQL> commit;

Commit complete.

SQL> **select count(\*) from myemp;**  
COUNT(\*)  
———-  
**16**

SQL> select count(\*) from myemp where job=’MANAGER';

COUNT(\*)  
———-  
**10**

**On Target**, we will see that even though two rows have been inserted into the source MYEMP table, on the target MYEMP table only one row is inserted because the filter has been applied which only includes the rows where the JOB value equals ‘MANAGER’.

SQL> **select count(\*) from myemp**;

COUNT(\*)  
———-  
10

**Example 2 – source and target table differ in column structure**

In the source MYEMP table we have a column named SAL whereas on the target, the same MYEMP table has the column defined as SALARY.

**Create a definitions file on the source using DEFGEN utility and then copy that definitions file to the target system**

GGSCI (redhat346.localdomain) > **EDIT PARAMS defgen**

**DEFSFILE /u01/oracle/ggs/dirsql/myemp.sql**  
USERID ggs\_owner, PASSWORD ggs\_owner  
TABLE scott.myemp;

[oracle@redhat346 ggs]$ **./defgen paramfile /u01/oracle/ggs/dirprm/defgen.prm**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
Oracle GoldenGate Table Definition Generator for Oracle  
Version 10.4.0.19 Build 002  
Linux, x64, 64bit (optimized), Oracle 11 on Sep 18 2009 00:09:13

Copyright (C) 1995, 2009, Oracle and/or its affiliates. All rights reserved.

Starting at 2010-02-23 11:22:17  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Operating System Version:  
Linux  
Version #1 SMP Wed Dec 17 11:41:38 EST 2008, Release 2.6.18-128.el5  
Node: redhat346.localdomain  
Machine: x86\_64  
soft limit hard limit  
Address Space Size : unlimited unlimited  
Heap Size : unlimited unlimited  
File Size : unlimited unlimited  
CPU Time : unlimited unlimited

Process id: 14175

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
\*\* Running with the following parameters \*\*  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
DEFSFILE /u01/oracle/ggs/dirsql/myemp.sql  
USERID ggs\_owner, PASSWORD \*\*\*\*\*\*\*\*\*  
TABLE scott.myemp;  
Retrieving definition for SCOTT.MYEMP

**Definitions generated for 1 tables in /u01/oracle/ggs/dirsql/myemp.sql**

If we were to try and run the replicat process on the target without copying the definitions file, we will see an error as shown below which pertains to the fact that the columns in the source and target database are different and GoldenGate is not able to resolve that.

2010-02-23 11:31:07 GGS WARNING 218 Aborted grouped transaction on ‘SH.MYEMP’, Database error 904 (ORA-00904: “SAL”: invalid identifier).

2010-02-23 11:31:07 GGS WARNING 218 SQL error 904 mapping SCOTT.MYEMP to SH.MYEMP OCI Error ORA-00904: “SAL”: invalid identifier (status = 904), SQL *.*

*We then ftp the definitions file from the source to the target system – in this case to the dirsql directory located in the top level GoldenGate installed software directory*

*We now go and make a change to the original replicat parameter file and change the parameter ASSUMEDEFS to****SOURCEDEFS****which provides GoldenGate with the location of the definitions file.*

*The other parameter which is included is the****COLMAP****parameter which tells us how the column mapping has been performed. The ‘USEDEFAULTS’ keyword denotes that all the other columns in both tables are identical except for the columns SAL and SALARY which differ in both tables and now we are mapping the SAL columsn in source to the SALARY column on the target.*

*REPLICAT myload2****SOURCEDEFS /u01/oracle/software/goldengate/dirsql/myemp.s****ql  
USERID ggs\_owner, PASSWORD ggs\_owner  
MAP scott.myemp, TARGET sh.myemp,****COLMAP (usedefaults,  
salary = sal);***

*We now go and start the originall replicat process myload2 which had abended because of the column mismatch (which has now been corrected via the parameter change) and we see that the process now is running without any error.*

*now go and start the process which had failed after table modification*

*GGSCI (devu007) 2> info replicat myload2*

*REPLICAT MYLOAD2 Last Started 2010-02-23 11:05 Status ABENDED  
Checkpoint Lag 00:00:03 (updated 00:11:44 ago)  
Log Read Checkpoint File /u01/oracle/software/goldengate/dirdat/bb000000  
2010-02-23 11:31:03.999504 RBA 1225*

*GGSCI (devu007) 3>****start replicat myload2***

*Sending START request to MANAGER …  
REPLICAT MYLOAD2 starting*

*GGSCI (devu007) 4>****info replicat myload2***

*REPLICAT MYLOAD2 Last Started 2010-02-23 11:43 Status RUNNING  
Checkpoint Lag 00:00:00 (updated 00:00:03 ago)  
Log Read Checkpoint File /u01/oracle/software/goldengate/dirdat/bb000000  
2010-02-23 11:31:03.999504 RBA 1461*

# Oracle GoldenGate Tutorial 9 – Monitoring GoldenGate

The following tutorial will briefly discuss the different commands we can use to monitor the GoldenGate environment and get statistics and reports on various extract and replicat operations which are in progress.

More details can be obtained from Chapter 19 of the Oracle GoldenGate Windows and Unix Administration guide – Monitoring GoldenGate processing.

**Information on all GoldenGate processes running on a system**

GGSCI (devu007) 21> **info all**

Program Status Group Lag Time Since Chkpt

MANAGER RUNNING

EXTRACT RUNNING DPUMP 00:00:00 00:00:04

EXTRACT RUNNING EXT1 00:00:00 00:00:09

EXTRACT RUNNING EXT2 00:00:00 00:00:07

EXTRACT ABENDED GAVIN 00:00:00 73:29:25

EXTRACT STOPPED WORKPLAN 00:00:00 191:44:18

REPLICAT RUNNING MYLOAD2 00:00:00 00:00:09

REPLICAT RUNNING MYREP 00:00:00 00:00:08

**Find the run status of a particular process**

GGSCI (devu007) 23> **status manager**

Manager is running (IP port devu007.7809).

GGSCI (devu007) 24> **status extract ext1**

EXTRACT EXT1: RUNNING

**Detailed information of a particular process**

GGSCI (devu007) 6> **info extract ext1, detail**

EXTRACT EXT1 Last Started 2010-02-19 11:19 Status RUNNING

Checkpoint Lag 00:00:00 (updated 00:00:02 ago)

Log Read Checkpoint Oracle Redo Logs

2010-02-26 10:45:18 Seqno 786, RBA 44710400

Target Extract Trails:

Remote Trail Name Seqno RBA Max MB

/u01/oracle/software/goldengate/dirdat/lt 2 55644 10

Extract Source Begin End

/u02/oradata/apex/redo03.log 2010-02-19 11:13 2010-02-26 10:45

/u02/oradata/apex/redo02.log 2010-02-19 11:04 2010-02-19 11:13

/u02/oradata/apex/redo02.log 2010-02-18 10:42 2010-02-19 11:04

Not Available \* Initialized \* 2010-02-18 10:42

Current directory /u01/oracle/software/goldengate

Report file /u01/oracle/software/goldengate/dirrpt/EXT1.rpt

Parameter file /u01/oracle/software/goldengate/dirprm/ext1.prm

Checkpoint file /u01/oracle/software/goldengate/dirchk/EXT1.cpe

Process file /u01/oracle/software/goldengate/dirpcs/EXT1.pce

Stdout file /u01/oracle/software/goldengate/dirout/EXT1.out

Error log /u01/oracle/software/goldengate/ggserr.log

**Monitoring an Extract recovery**

GGSCI (devu007) 35> **send extract ext1 status**

Sending STATUS request to EXTRACT EXT1 ...

EXTRACT EXT1 (PID 1925238)

Current status: Recovery complete: At EOF

Sequence #: 786

RBA: 40549888

Timestamp: 2010-02-26 09:59:57.000000

Output trail #1

Current write position:

Sequence #: 2

RBA: 55644

Timestamp: 2010-02-26 09:59:54.337574

Extract Trail: /u01/oracle/software/goldengate/dirdat/lt

**Monitoring processing volume - Statistics of the operations processed**

GGSCI (devu007) 33> **stats extract ext1**

Sending STATS request to EXTRACT EXT1 ...

Start of Statistics at 2010-02-26 09:58:27.

DDL replication statistics (for all trails):

\*\*\* Total statistics since extract started \*\*\*

Operations 19.00

Mapped operations 2.00

Unmapped operations 9.00

Other operations 8.00

Excluded operations 17.00

Output to /u01/oracle/software/goldengate/dirdat/lt:

Extracting from GGS\_OWNER.GGS\_MARKER to GGS\_OWNER.GGS\_MARKER:

\*\*\* Total statistics since 2010-02-19 11:21:03 \*\*\*

No database operations have been performed.

\*\*\* Daily statistics since 2010-02-26 00:00:00 \*\*\*

No database operations have been performed.

\*\*\* Hourly statistics since 2010-02-26 09:00:00 \*\*\*

No database operations have been performed.

\*\*\* Latest statistics since 2010-02-19 11:21:03 \*\*\*

No database operations have been performed.

Extracting from MONITOR.WORK\_PLAN to MONITOR.WORK\_PLAN:

\*\*\* Total statistics since 2010-02-19 11:21:03 \*\*\*

Total inserts 4.00

Total updates 46.00

Total deletes 0.00

Total discards 0.00

Total operations 50.00

\*\*\* Daily statistics since 2010-02-26 00:00:00 \*\*\*

Total inserts 0.00

Total updates 16.00

Total deletes 0.00

Total discards 0.00

Total operations 16.00

\*\*\* Hourly statistics since 2010-02-26 09:00:00 \*\*\*

No database operations have been performed.

\*\*\* Latest statistics since 2010-02-19 11:21:03 \*\*\*

Total inserts 4.00

Total updates 46.00

Total deletes 0.00

Total discards 0.00

Total operations 50.00

End of Statistics.

**View processing rate - can use 'hr','min' or 'sec' as a para**meter

GGSCI (devu007) 37> **stats extract ext2 reportrate hr**

Sending STATS request to EXTRACT EXT2 ...

Start of Statistics at 2010-02-26 10:04:46.

Output to /u01/oracle/ggs/dirdat/cc:

Extracting from SH.CUSTOMERS to SH.CUSTOMERS:

\*\*\* Total statistics since 2010-02-26 09:29:48 \*\*\*

Total inserts/hour: 0.00

Total updates/hour: 95258.62

Total deletes/hour: 0.00

Total discards/hour: 0.00

Total operations/hour: 95258.62

\*\*\* Daily statistics since 2010-02-26 09:29:48 \*\*\*

Total inserts/hour: 0.00

Total updates/hour: 95258.62

Total deletes/hour: 0.00

Total discards/hour: 0.00

Total operations/hour: 95258.62

\*\*\* Hourly statistics since 2010-02-26 10:00:00 \*\*\*

No database operations have been performed.

\*\*\* Latest statistics since 2010-02-26 09:29:48 \*\*\*

Total inserts/hour: 0.00

Total updates/hour: 95258.62

Total deletes/hour: 0.00

Total discards/hour: 0.00

Total operations/hour: 95258.62

End of Statistics.

**View latency between the records processed by Goldengate and the timestamp in the data source**

GGSCI (devu007) 13> **send extract ext2, getlag**

Sending GETLAG request to EXTRACT EXT2 ...

Last record lag: 3 seconds.

At EOF, no more records to process.

GGSCI (devu007) 15> **lag extract ext\***

Sending GETLAG request to EXTRACT EXT1 ...

Last record lag: 1 seconds.

At EOF, no more records to process.

Sending GETLAG request to EXTRACT EXT2 ...

Last record lag: 1 seconds.

At EOF, no more records to process.

**Viewing the GoldenGate error log as well as history of commands executed and other events**

We can use the editor depending on operating system – vi on Unix for example to view the **ggserr.log** file which is located at the top level GoldenGate software installation directory.

We can also use the GGSCI command **VIEW GGSEVT** as well.

**View the process report**

Every Manager, Extract and Replicat process will generate a report file at the end of each run and this  
report can be viewed to diagnose any problems or errors as well as view the parameters used, the environment variables is use, memory consumption etc

For example:

GGSCI (devu007) 2> **view report ext1**

GGSCI (devu007) 2> **view report rep1**

GGSCI (devu007) 2> **view report mgr**

**Information on Child processes started by the Manager**

GGSCI (devu007) 8> **send manager childstatus**

Sending CHILDSTATUS request to MANAGER ...

Child Process Status - 6 Entries

ID Group Process Retry Retry Time Start Time

---- -------- ---------- ----- ------------------ -----------

0 EXT1 1925238 0 None 2010/02/19 11:07:54

1 DPUMP 2195496 0 None 2010/02/19 11:08:02

2 MSSQL1 422034 0 None 2010/02/22 13:54:59

4 MYREP 1302702 0 None 2010/02/23 09:08:34

6 MYLOAD2 1200242 0 None 2010/02/23 11:05:01

7 EXT2 2076844 0 None 2010/02/26 08:29:22

# Oracle GoldenGate Tutorial 10- performing a zero downtime cross platform migration and 11g database upgrade

This note briefly describes the steps required to perform a cross platform database migration (AIX to Red Hat Linux) and also a database upgrade from 10g to 11g Release 2 which is attained with zero downtime using a combination of RMAN, Cross Platform TTS and GoldenGate to achieve the same.

This is the environment that we will be referring to in this note:

10..2.0.4 Database on AIX – **DB10g**  
10.2.0.4 Duplicate database on AIX – **Clonedb**  
11.2 database on Linux – **DB11g**

**Steps**

1) **Create the GoldenGate Extract process on source AIX DB10g and start the same. This extract process will be capturing changes as they occur on the 10g AIX database in the remote trail files located on the Linux target system. Since the Replicat process is not running on the target at this time, the source database changes will accumulate in the extract trail files**.

GGSCI (devu026) 12> add extract myext, tranlog, begin now  
EXTRACT added.

GGSCI (devu026) 13> add rmttrail /u01/oracle/ggs/dirdat/my, extract myext  
RMTTRAIL added.

GGSCI (devu026) 14> edit params myext

“/u01/rapmd2/ggs/dirprm/myext.prm” 7 lines, 143 characters  
EXTRACT myext  
USERID ggs\_owner, PASSWORD ggs\_owner  
SETENV (ORACLE\_HOME = “/u01/oracle/product/10.2/rapmd2″)  
SETENV (ORACLE\_SID = “db10g”)  
RMTHOST 10.1.210.35, MGRPORT 7809  
RMTTRAIL /u01/oracle/ggs/dirdat/my  
DISCARDFILE discard.txt, APPEND  
TABLE sh.\*;  
TABLE hr.\*;  
TABLE pm.\*;  
TABLE oe.\*;  
TABLE ix.\*;

START THE EXTRACT PROCESS NOW

GGSCI (devu026) 16> START EXTRACT MYEXT

Sending START request to MANAGER …  
EXTRACT MYEXT starting

GGSCI (devu026) 17> INFO EXTRACT MYEXT

EXTRACT MYEXT Last Started 2010-03-04 08:42 Status RUNNING  
Checkpoint Lag 00:31:07 (updated 00:00:01 ago)  
Log Read Checkpoint Oracle Redo Logs  
2010-03-04 08:11:26 Seqno 8, RBA 2763280

2) **Using RMAN create a duplicate database in the source AIX environment (Clonedb) – this database will be used as the source for the export of database structure (no rows export) and tablespace meta data**

Follow this white paper to get all the steps involved.

**\*\*\*\*\*\*\*\*\*\*\*ON SOURCE – UPDATE 1\*\*\*\*\*\*\*\*\*\***

SQL> conn sh/sh  
Connected.  
SQL> update mycustomers set cust\_city=’Singapore';

55500 rows updated.

SQL> commit;

Commit complete.

3) **Create a skeleton database on the Linux platform in the 11g Release 2 environment – DB11g**

Note – we will then set up the GoldenGate user GGS\_OWNER in the database and grant it the required privileges as well as create the checkpoint table. Read one of the earlier tutorials which details the set up of the GGS\_OWNER user in the database.

4) **Take a full export of the database without any table data to get just the structure of the database – this is now taken from the clonedb duplicate database created in step 2**

db10g:/u01/oracle> **expdp dumpfile=full\_norows.dmp directory =dumpdir content=metadata\_only exclude=tables,index full=y**

Export: Release 10.2.0.4.0 – 64bit Production on Thursday, 04 March, 2010 9:02:44

Copyright (c) 2003, 2007, Oracle. All rights reserved.

Username: sys as sysdba  
Password:

Connected to: Oracle Database 10g Enterprise Edition Release 10.2.0.4.0 – 64bit Production  
With the Partitioning, Data Mining and Real Application Testing options  
FLASHBACK automatically enabled to preserve database integrity.  
Starting “SYS”.”SYS\_EXPORT\_FULL\_01″: sys/\*\*\*\*\*\*\*\* AS SYSDBA dumpfile=full\_norows.dmp directory =dumpdir content=metadata\_only exclude=tables,index full=y  
Processing object type DATABASE\_EXPORT/TABLESPACE  
Processing object type DATABASE\_EXPORT/PROFILE  
Processing object type DATABASE\_EXPORT/SYS\_USER/USER  
Processing object type DATABASE\_EXPORT/SCHEMA/USER  
Processing object type DATABASE\_EXPORT/ROLE  
Processing object type DATABASE\_EXPORT/GRANT/SYSTEM\_GRANT/PROC\_SYSTEM\_GRANT  
Processing object type DATABASE\_EXPORT/SCHEMA/GRANT/SYSTEM\_GRANT  
Processing object type DATABASE\_EXPORT/SCHEMA/ROLE\_GRANT  
…………………  
…………………….

5) **Import the dumpfile into the 11g database DB11g which has the database structure without the table data – this will create all the users, roles, synonyms etc**

We had to create a role and also create the directory before doing the full database import. Ignore he errors during the import as it will pertain to objects which already exist in the scratch database.

SQL> create role xdbwebservices;

Role created.

SQL> create directory dumpdir as ‘/u01/oracle';

Directory created.

[oracle@redhat346 ~]$ **impdp dumpfile=full\_norows.dmp directory=dumpdir full=y**

Import: Release 11.2.0.1.0 – Production on Mon Mar 8 13:09:16 2010

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

…………  
……….

6) **On the clonedb database, we now will export the tablespace meta data – make the required tablespaces read only. Note that the original source 10g database is in read write mode and is being accessed by the users with no downtime as yet.**

clonedb:/u01/rapmd2/ggs> **expdp dumpfile=tts\_meta.dmp directory =dumpdir transport\_tablespaces=EXAMPLE,TTS**

Export: Release 10.2.0.4.0 – 64bit Production on Monday, 08 March, 2010 13:01:38

Copyright (c) 2003, 2007, Oracle. All rights reserved.

Username: sys as sysdba  
Password:

Connected to: Oracle Database 10g Enterprise Edition Release 10.2.0.4.0 – 64bit Production  
With the Partitioning, Data Mining and Real Application Testing options  
Starting “SYS”.”SYS\_EXPORT\_TRANSPORTABLE\_01″: sys/\*\*\*\*\*\*\*\* AS SYSDBA dumpfile=tts\_meta.dmp directory =dumpdir transport\_tablespaces=EXAMPLE,TTS  
Processing object type TRANSPORTABLE\_EXPORT/PLUGTS\_BLK  
Processing object type TRANSPORTABLE\_EXPORT/TYPE/TYPE\_SPEC  
Processing object type TRANSPORTABLE\_EXPORT/TABLE  
Processing object type TRANSPORTABLE\_EXPORT/GRANT/OWNER\_GRANT/OBJECT\_GRANT  
Processing object type TRANSPORTABLE\_EXPORT/INDEX  
Processing object type TRANSPORTABLE\_EXPORT/CONSTRAINT/CONSTRAINT  
Processing object type TRANSPORTABLE\_EXPORT/INDEX\_STATISTICS  
Processing object type TRANSPORTABLE\_EXPORT/COMMENT  
Processing object type TRANSPORTABLE\_EXPORT/CONSTRAINT/REF\_CONSTRAINT  
Processing object type TRANSPORTABLE\_EXPORT/TRIGGER  
Processing object type TRANSPORTABLE\_EXPORT/INDEX/FUNCTIONAL\_AND\_BITMAP/INDEX  
Processing object type TRANSPORTABLE\_EXPORT/INDEX/STATISTICS/FUNCTIONAL\_AND\_BITMAP/INDEX\_STATISTICS  
Processing object type TRANSPORTABLE\_EXPORT/TABLE\_STATISTICS  
Processing object type TRANSPORTABLE\_EXPORT/DOMAIN\_INDEX/TABLE  
Processing object type TRANSPORTABLE\_EXPORT/DOMAIN\_INDEX/SECONDARY\_TABLE/INDEX  
Processing object type TRANSPORTABLE\_EXPORT/DOMAIN\_INDEX/INDEX  
Processing object type TRANSPORTABLE\_EXPORT/MATERIALIZED\_VIEW  
Processing object type TRANSPORTABLE\_EXPORT/POST\_INSTANCE/PROCACT\_INSTANCE  
Processing object type TRANSPORTABLE\_EXPORT/POST\_INSTANCE/PROCDEPOBJ  
Processing object type TRANSPORTABLE\_EXPORT/POST\_INSTANCE/PLUGTS\_BLK  
Master table “SYS”.”SYS\_EXPORT\_TRANSPORTABLE\_01″ successfully loaded/unloaded  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
Dump file set for SYS.SYS\_EXPORT\_TRANSPORTABLE\_01 is:  
/u01/oracle/tts\_meta.dmp  
Job “SYS”.”SYS\_EXPORT\_TRANSPORTABLE\_01″ successfully completed at 13:02:17

7) **Copy the datafiles from the read only tablespaces ( from clonedb) to the target Linux system and using RMAN convert the datafiles from the AIX platform to the Linux platform**

RMAN> **CONVERT DATAFILE ‘/u01/oracle/example01.dbf’**  
2> **FROM PLATFORM=’AIX-Based Systems (64-bit)’**  
3> **FORMAT ‘/u02/oradata/db11g/example01.dbf';**

Starting conversion at target at 08-MAR-10  
using target database control file instead of recovery catalog  
allocated channel: ORA\_DISK\_1  
channel ORA\_DISK\_1: SID=11 device type=DISK  
channel ORA\_DISK\_1: starting datafile conversion  
input file name=/u01/oracle/example01.dbf  
converted datafile=/u02/oradata/db11g/example01.dbf  
channel ORA\_DISK\_1: datafile conversion complete, elapsed time: 00:00:03  
Finished conversion at target at 08-MAR-10

RMAN> **CONVERT DATAFILE ‘/u01/oracle/tts01.dbf’**  
2> **FROM PLATFORM=’AIX-Based Systems (64-bit)’**  
3> **FORMAT ‘/u02/oradata/db11g/tts01.dbf';**

Starting conversion at target at 08-MAR-10  
using channel ORA\_DISK\_1  
channel ORA\_DISK\_1: starting datafile conversion  
input file name=/u01/oracle/tts01.dbf  
converted datafile=/u02/oradata/db11g/tts01.dbf  
channel ORA\_DISK\_1: datafile conversion complete, elapsed time: 00:00:01  
Finished conversion at target at 08-MAR-10

8) **Import the tablespace meta data into the 11g database and plug in the tablespaces -make the tablespaces read write**

[oracle@redhat346 ~]$ **impdp dumpfile=tts\_meta.dmp directory=dumpdir transport\_datafiles=”/u02/oradata/db11g/example01.dbf”,”/u02/oradata/db11g/tts01.dbf”**

Import: Release 11.2.0.1.0 – Production on Mon Mar 8 13:21:37 2010

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Username: sys as sysdba  
Password:

Connected to: Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 – 64bit Production  
With the Partitioning and Real Application Testing options  
Master table “SYS”.”SYS\_IMPORT\_TRANSPORTABLE\_01″ successfully loaded/unloaded  
Starting “SYS”.”SYS\_IMPORT\_TRANSPORTABLE\_01″: sys/\*\*\*\*\*\*\*\* AS SYSDBA dumpfile=tts\_meta.dmp directory=dumpdir transport\_datafiles=/u02/oradata/db11g/example01.dbf,/u02/oradata/db11g/tts01.dbf  
Processing object type TRANSPORTABLE\_EXPORT/PLUGTS\_BLK  
Processing object type TRANSPORTABLE\_EXPORT/TYPE/TYPE\_SPEC  
ORA-31684: Object type TYPE:”PM”.”ADHEADER\_TYP” already exists  
ORA-31684: Object type TYPE:”PM”.”TEXTDOC\_TYP” already exists  
ORA-31684: Object type TYPE:”IX”.”ORDER\_EVENT\_TYP” already exists  
ORA-31684: Object type TYPE:”OE”.”PHONE\_LIST\_TYP” already exists  
ORA-31684: Object type TYPE:”OE”.”CUST\_ADDRESS\_TYP” already exists  
ORA-31684: Object type TYPE:”PM”.”TEXTDOC\_TAB” already exists  
Processing object type TRANSPORTABLE\_EXPORT/TABLE  
Processing object type TRANSPORTABLE\_EXPORT/GRANT/OWNER\_GRANT/OBJECT\_GRANT  
Processing object type TRANSPORTABLE\_EXPORT/INDEX  
Processing object type TRANSPORTABLE\_EXPORT/CONSTRAINT/CONSTRAINT  
Processing object type TRANSPORTABLE\_EXPORT/INDEX\_STATISTICS  
Processing object type TRANSPORTABLE\_EXPORT/COMMENT  
Processing object type TRANSPORTABLE\_EXPORT/CONSTRAINT/REF\_CONSTRAINT  
Processing object type TRANSPORTABLE\_EXPORT/TRIGGER  
Processing object type TRANSPORTABLE\_EXPORT/INDEX/FUNCTIONAL\_AND\_BITMAP/INDEX  
Processing object type TRANSPORTABLE\_EXPORT/INDEX/STATISTICS/FUNCTIONAL\_AND\_BITMAP/INDEX\_STATISTICS  
Processing object type TRANSPORTABLE\_EXPORT/TABLE\_STATISTICS  
Processing object type TRANSPORTABLE\_EXPORT/DOMAIN\_INDEX/TABLE  
Processing object type TRANSPORTABLE\_EXPORT/DOMAIN\_INDEX/SECONDARY\_TABLE/INDEX  
…………………..  
……………………………..

SQL> alter tablespace tts read write;

Tablespace altered.

SQL> alter tablespace example read write;

Tablespace altered.

**\*\*\*\*\*\*\*\*\*\*\*ON SOURCE – UPDATE 2\*\*\*\*\*\*\*\*\*\***

SQL> conn sh/sh  
Connected.  
SQL> update mycustomers set cust\_city=’Hong Kong';

55500 rows updated.

SQL> commit;

Commit complete.

Note:

As we make changes in the source database, the trail files on the target start getting populated. These are located in the destination we specified when creating the RMTTRAIL.

[oracle@redhat346 dirdat]$ pwd  
/u01/oracle/ggs/dirdat

[oracle@redhat346 dirdat]$ ls -lrt

-rw-rw-rw- 1 oracle oinstall 9999950 Mar 8 09:41 gs000000  
-rw-rw-rw- 1 oracle oinstall 9999641 Mar 8 09:41 gs000001  
-rw-rw-rw- 1 oracle oinstall 9999629 Mar 8 10:00 gs000003  
-rw-rw-rw- 1 oracle oinstall 9999724 Mar 8 10:00 gs000002  
-rw-rw-rw- 1 oracle oinstall 9999741 Mar 8 10:00 gs000004  
-rw-rw-rw- 1 oracle oinstall 2113226 Mar 8 10:00 gs000005  
-rw-rw-rw- 1 oracle oinstall 9999791 Mar 8 10:35 rm000000  
-rw-rw-rw- 1 oracle oinstall 9999721 Mar 8 10:35 rm000001  
-rw-rw-rw- 1 oracle oinstall 9999249 Mar 8 10:49 rm000003  
-rw-rw-rw- 1 oracle oinstall 9999309 Mar 8 10:49 rm000002  
-rw-rw-rw- 1 oracle oinstall 9999818 Mar 8 10:49 rm000004  
-rw-rw-rw- 1 oracle oinstall 9999430 Mar 8 10:49 rm000005  
-rw-rw-rw- 1 oracle oinstall 9999412 Mar 8 10:49 rm000006  
-rw-rw-rw- 1 oracle oinstall 9999588 Mar 8 10:54 rm000007  
-rw-rw-rw- 1 oracle oinstall 9999481 Mar 8 10:54 rm000009  
-rw-rw-rw- 1 oracle oinstall 9999399 Mar 8 10:54 rm000008  
-rw-rw-rw- 1 oracle oinstall 9999787 Mar 8 10:54 rm000010  
-rw-rw-rw- 1 oracle oinstall 9999770 Mar 8 10:57 rm000011  
-rw-rw-rw- 1 oracle oinstall 9999941 Mar 8 10:57 rm000012  
-rw-rw-rw- 1 oracle oinstall 9999913 Mar 8 10:57 rm000013  
-rw-rw-rw- 1 oracle oinstall 9999429 Mar 8 11:09 rm000014  
-rw-rw-rw- 1 oracle oinstall 9999812 Mar 8 11:09 rm000015  
-rw-rw-rw- 1 oracle oinstall 9999240 Mar 8 11:09 rm000016  
-rw-rw-rw- 1 oracle oinstall 9999454 Mar 8 11:09 rm000017  
-rw-rw-rw- 1 oracle oinstall 9999914 Mar 8 11:09 rm000018  
-rw-rw-rw- 1 oracle oinstall 9999820 Mar 8 11:16 rm000019  
-rw-rw-rw- 1 oracle oinstall 9999766 Mar 8 11:16 rm000020  
-rw-rw-rw- 1 oracle oinstall 9999706 Mar 8 12:56 rm000021  
-rw-rw-rw- 1 oracle oinstall 9999577 Mar 8 12:56 rm000022  
-rw-rw-rw- 1 oracle oinstall 9999841 Mar 8 12:56 rm000023  
-rw-rw-rw- 1 oracle oinstall 9999890 Mar 8 13:26 rm000024  
-rw-rw-rw- 1 oracle oinstall 9999604 Mar 8 13:26 rm000025  
-rw-rw-rw- 1 oracle oinstall 9999536 Mar 8 13:26 rm000026  
-rw-rw-rw- 1 oracle oinstall 918990 Mar 8 13:26 rm000027

9) **On the target Linux environment now we create and start the GoldenGate Replicat process/processes. They will now start reading from the Extract trail files created in Step 1 and will start applying them to the 11g database.**

GGSCI (redhat346.localdomain) 1> **add replicat myrep, extrail /u01/oracle/ggs/dirdat/rm**  
REPLICAT added.

GGSCI (redhat346.localdomain) 6> **edit params myrep**

REPLICAT myrep  
SETENV (ORACLE\_HOME = “/u01/app/oracle/product/11.2.0/dbhome\_1″)  
SETENV (ORACLE\_SID = “db11g”)  
ASSUMETARGETDEFS  
USERID ggs\_owner, PASSWORD ggs\_owner  
MAP sh.\*, TARGET sh.\*;  
MAP pm.\*, TARGET pm.\*;  
MAP oe.\*, TARGET oe.\*;  
MAP hr.\*, TARGET hr.\*;  
MAP ix.\*, TARGET ix.\*;

10) **Once all the changes in the trail files have been applied by the Replicat process and we confirm that both source and target are in sync (we can use another GoldenGate product called Veridata for this), we can now point the users and application to the 11g Linux database with no or minimal downtime which will depend on the infrastructure**.

We can see the Replicat process going through and reading all the trail files until it has completed processing all the files

GGSCI (redhat346.localdomain) 131> **info replicat myrep**

REPLICAT MYREP Last Started 2010-03-08 13:42 Status RUNNING  
Checkpoint Lag 03:07:37 (updated 00:00:17 ago)  
Log Read Checkpoint File **/u01/oracle/ggs/dirdat/rm000002**  
2010-03-08 10:35:27.001328 RBA 6056361  
…….  
………..

GGSCI (redhat346.localdomain) 156> **info replicat myrep**

REPLICAT MYREP Last Started 2010-03-08 13:42 Status RUNNING  
Checkpoint Lag 02:53:49 (updated 00:00:00 ago)  
Log Read Checkpoint File **/u01/oracle/ggs/dirdat/rm000007**  
2010-03-08 10:49:39.001103 RBA 2897635

………………  
……………..

GGSCI (redhat346.localdomain) 133> **info replicat myrep**

REPLICAT MYREP Last Started 2010-03-08 13:48 Status RUNNING  
Checkpoint Lag 00:00:00 (updated 00:00:07 ago)  
Log Read Checkpoint File **/u01/oracle/ggs/dirdat/rm000027**  
2010-03-08 13:26:43.000861 RBA 918990

GGSCI (redhat346.localdomain) 134> **lag replicat myrep**

Sending GETLAG request to REPLICAT MYREP …  
Last record lag: 1363 seconds.  
At EOF, no more records to process.

**TEST!**

**Now check and confirm from the database if second update statement (UPDATE 2) run on the source database has been applied on the target**

SQL> select distinct cust\_city from mycustomers;

CUST\_CITY  
——————————  
**Hong Kong**

**We can now point our clients to the upgraded 11g database!**

**GoldenGate - What is supported and what is not....**

Very often in these forums I get asked the question regarding data types which GoldenGate supports or Oracle software versions or questions like can it work with a database running in NOARCHIVELOG mode or does it support Partitioned and Index Organised tables and so forth.

I have included some snippets taken from the official documentation as well as some of my own experiences and hopefully it will help answer many of the those frequently asked questions.

**What Oracle Versions are supported?**

Oracle 9.2 and above, Oracle 10.1 and 10.2 and Oracle 11gR1 and 11gR2

**How many Extract or Replicat processes can we start?**

GoldenGate supports upto 300 Extract and Replicat processes per GoldenGate instance. But keep in mind that each process consumes around 50 MB of memory.

**What Oracle Data Types are supported?**

NUMBER  
BINARY FLOAT  
BINARY DOUBLE  
CHAR  
VARCHAR2  
LONG  
NCHAR  
NVARCHAR2  
RAW  
LONG RAW  
DATE  
TIMESTAMP  
CLOB  
NCLOB  
BLOB (SECUREFILE and BASICFILE are both supported.)  
XML data types  
User defined types

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

Oracle GoldenGate does not support REF types.

Oracle GoldenGate supports SDO\_GEOMETRY, SDO\_TOPO\_GEOMETRY, and SDO\_GEORASTER for Oracle 10g and later.

**Non-Supported Data Types**

ORDDICOM  
ANYDATA  
ANYDATASET  
ANYTYPE  
BFILE  
MLSLABEL  
TIMEZONE\_ABBR  
TIMEZONE\_REGION  
URITYPE  
UROWID

**DML Support**

Oracle GoldenGate supports the following DML operations made to regular tables, index organized tables, clustered tables, and materialized views.

INSERT  
UPDATE  
DELETE

Note:

Oracle GoldenGate **supports tables with virtual columns**, but does not capture change.

**Tables created as EXTERNAL are not supported**.

**Tables created with table compression or OLTP table compression are not supported.**

**Materialized views created WITH ROWID are not supported**.

**Support for Sequences**

Oracle GoldenGate supports the replication of sequence values by means of the SEQUENCE parameter  
The cache size and the increment interval of the source and target sequences must be identical.

**DDL Replication Support**

Oracle Goldengate supports DDL operations on the following database objects:

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

Note:

Oracle GoldenGate **supports the synchronization of TRUNCATE statements**as well as part of the DDL synchronization support.

**DDL On Oracle Reserved Schemas is not supported**.

These include:

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

Because of a known issue in Oracle 10g, the **Oracle recycle bin must be turned off to support Oracle GoldenGate DDL replication**.

**Source and Target database charactersets**

The character set of the target database must be a superset of the character set of the source database.

**Can I run the source database in NOARCHIVELOG mode?**

Oracle GoldenGate reads the online redo logs by default, but will read the archived logs if an online log is not available. If using in NOARCHIVELOG mode ensure that the redo log files are created with a larger size and more memebsrs are created to prevent then form being recycled before the Extract process has completed processing them.

**GoldenGate using Data Guard Standby as a source**

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

Oracle GoldenGate will connect to the secondary database to get metadata and other required data as needed. Supplemental logging at the table level and the database level must be enabled for the tables from the source database.

**How do we handle Triggers and ON-DELETE cascade constraints on the target?**

For Oracle 10.2.0.5 and later patches, and for Oracle 11.2.0.2 and later, you can use the Replicat parameter DBOPTIONS with the SUPPRESSTRIGGERS option to cause Replicat to disable the triggers during its session.

For Oracle 9.2.0.7 and later, you can use the Replicat parameter DBOPTIONS with the DEFERREFCONST option to delay checking and enforcement of integrity constraints until the Replicat transaction commits.

# Defining Multiple Replicat Processes to Increase GoldenGate Performance

Oracle states that GoldenGate can achieve near real-time data replication. However, out of the box, GoldenGate may not meet your performance requirements.

The GoldenGate documentation states "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".

But what if you have some large tables with a high data change rate within a source schema and you cannot logically separate them from the remaining tables due to referential constraints? GoldenGate does provide a solution to this problem by "splitting" the data and not the schema via the @RANGE function.

The Replicat process is typically the 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. Therefore, to leverage parallel operation, the more Replicats the better (dependant on the number of CPUs on the target system).

### The RANGE Function

The way the @RANGE function works is it computes a hash value of the columns specified in the input. If no columns are specified it uses the table’s primary key. GoldenGate adjusts the total number of ranges to optimise the even distribution across the number of ranges specified. This concept can be compared to Hash Partitioning in Oracle as a means of dividing data.

With any division of data during replication, the integrity is paramount and will have an effect on performance. Therefore, tables having a relationship with other tables in the source schema must be included in the configuration. If all your source schema tables are inter-related, it is possible to include all tables!

The example below is from Oracle GoldenGate Reference Guide version 10.4:

(Replicat group 1 parameter file)

MAP sales.acct, TARGET sales.acct, FILTER (@RANGE (1, 3, ID));

(Replicat group 2 parameter file)

MAP sales.acct, TARGET sales.acct, FILTER (@RANGE (2, 3, ID));

(Replicat group 3 parameter file)

MAP sales.acct, TARGET sales.acct, FILTER (@RANGE (3, 3, ID));

In my example in the next section, I include 3 tables in the source schema and walk through the complete configuration from start to finish.

### Adding Replicats with @RANGE Function

I currently have an existing Replicat process on my target machine (linuxserver1) named RTARGET1 that includes the following 3 tables:

ORDERS

ORDER\_ITEMS

PRODUCTS

My source database schema name is SRC and target schema TGT.

The following steps add a new Replicat named RTARGET2 with the relevant configuration and adjusts Replicat RTARGET1 parameters to suit.

N.B. Before conducting any changes, first stop the existing Replicat processes.

1. Check existing Replicat process is running.

GGSCI (linuxserver1) 61> info all

Program Status Group Lag Time Since Chkpt

MANAGER RUNNING

REPLICAT RUNNING RTARGET1 00:00:00 00:00:12

1. Stop existing Replicat process.

GGSCI (linuxserver1) 68> stop REPLICAT RTARGET1

Sending STOP request to REPLICAT RTARGET1 ...

Request processed.

1. Add the new Replicat process, using existing trail file.

GGSCI (linuxserver1) 69> add REPLICAT RTARGET2, exttrail ./dirdat/tb

REPLICAT added.

1. Now add the configuration by creating a new parameter file.

GGSCI (linuxserver1) 74> edit params RTARGET2

--

-- Example Replicator parameter file to apply changes

-- to target tables

--

--CHECKPARAMS

REPLICAT RTARGET2

SOURCEDEFS ./dirdef/mydefs.def

SETENV (ORACLE\_SID= TARGET)

USERID ggs\_admin, PASSWORD ggs\_admin

DISCARDFILE ./dirrpt/rtarget2.dsc, PURGE

ALLOWDUPTARGETMAP

CHECKPOINTSECS 30

GROUPTRANSOPS 2000

-- set LOBWRITESIZE to 64K ckunks

DBOPTIONS LOBWRITESIZE 65536

-- This starts the macro

MACRO #exception\_handler

BEGIN

, TARGET ggs\_admin.exceptions

, COLMAP ( rep\_name = "RTARGET2"

, table\_name = @GETENV ("GGHEADER", "TABLENAME")

, errno = @GETENV ("LASTERR", "DBERRNUM")

, dberrmsg = @GETENV ("LASTERR", "DBERRMSG")

, optype = @GETENV ("LASTERR", "OPTYPE")

, errtype = @GETENV ("LASTERR", "ERRTYPE")

, logrba = @GETENV ("GGHEADER", "LOGRBA")

, logposition = @GETENV ("GGHEADER", "LOGPOSITION")

, committimestamp = @GETENV ("GGHEADER", "COMMITTIMESTAMP"))

, INSERTALLRECORDS

, EXCEPTIONSONLY;

END;

-- This ends the macro

REPERROR (DEFAULT, EXCEPTION)

REPERROR (DEFAULT2, ABEND)

REPERROR (-1, EXCEPTION)

REPERROR (-1403, EXCEPTION)

MAP SRC.ORDERS, TARGET TGT.ORDERS, FILTER (@RANGE (1,2));

MAP SRC.ORDERS #exception\_handler()

MAP SRC.ORDER\_ITEMS, TARGET TGT.ORDER\_ITEMS, FILTER (@RANGE (1,2));

MAP SRC.ORDER\_ITEMS #exception\_handler()

MAP SRC.PRODUCTS, TARGET TGT.PRODUCTS, FILTER (@RANGE (1,2));

MAP SRC.PRODUCTS #exception\_handler()

1. Now edit the configuration of the existing Replicat process. Add the @RANGE function to the FILTER clause of the MAP statement.

GGSCI (linuxserver1) 75> edit params RTARGET2

--

-- Example Replicator parameter file to apply changes

-- to target tables

--

--CHECKPARAMS

REPLICAT RTARGET1

SOURCEDEFS ./dirdef/mydefs.def

SETENV (ORACLE\_SID=TARGET)

USERID ggs\_admin, PASSWORD ggs\_admin

DISCARDFILE ./dirrpt/rtarget1.dsc, PURGE

ALLOWDUPTARGETMAP

CHECKPOINTSECS 30

GROUPTRANSOPS 2000

-- set LOBWRITESIZE to 64K ckunks

DBOPTIONS LOBWRITESIZE 65536

-- This starts the macro

MACRO #exception\_handler

BEGIN

, TARGET ggs\_admin.exceptions

, COLMAP ( rep\_name = "RTARGET1"

, table\_name = @GETENV ("GGHEADER", "TABLENAME")

, errno = @GETENV ("LASTERR", "DBERRNUM")

, dberrmsg = @GETENV ("LASTERR", "DBERRMSG")

, optype = @GETENV ("LASTERR", "OPTYPE")

, errtype = @GETENV ("LASTERR", "ERRTYPE")

, logrba = @GETENV ("GGHEADER", "LOGRBA")

, logposition = @GETENV ("GGHEADER", "LOGPOSITION")

, committimestamp = @GETENV ("GGHEADER", "COMMITTIMESTAMP"))

, INSERTALLRECORDS

, EXCEPTIONSONLY;

END;

-- This ends the macro

REPERROR (DEFAULT, EXCEPTION)

REPERROR (DEFAULT2, ABEND)

REPERROR (-1, EXCEPTION)

REPERROR (-1403, EXCEPTION)

MAP SRC.ORDERS, TARGET TGT.ORDERS, **FILTER (@RANGE (2,2));**

MAP SRC.ORDERS #exception\_handler()

MAP SRC.ORDER\_ITEMS, TARGET TGT.ORDER\_ITEMS, **FILTER (@RANGE (2,2));**

MAP SRC.ORDER\_ITEMS #exception\_handler()

MAP SRC.PRODUCTS, TARGET TGT.PRODUCTS, **FILTER (@RANGE (2,2));**

MAP SRC.PRODUCTS #exception\_handler()

1. Check both Replicat processes exist.

GGSCI (linuxserver1) 76> info all

Program Status Group Lag Time Since Chkpt

MANAGER RUNNING

REPLICAT STOPPED RTARGET1 00:00:00 00:10:35

REPLICAT STOPPED RTARGET2 00:00:00 00:12:25

1. Before starting both Replicat processes, obtain the log Sequence Number (SEQNO) and Relative Byte Address (RBA) from the original trail file.

GGSCI (linuxserver1) 78> info REPLICAT RTARGET1, detail

REPLICAT RTARGET1 Last Started 2010-04-01 15:35 Status STOPPED

Checkpoint Lag 00:00:00 (updated 00:12:43 ago)

Log Read Checkpoint File ./dirdat/tb000**279** **<- SEQNO**

2010-04-08 12:27:00.001016 RBA **43750979** **<- RBA**

Extract Source Begin End

./dirdat/tb000279 2010-04-01 12:47 2010-04-08 12:27

./dirdat/tb000257 2010-04-01 04:30 2010-04-01 12:47

./dirdat/tb000255 2010-03-30 13:50 2010-04-01 04:30

./dirdat/tb000206 2010-03-30 13:50 First Record

./dirdat/tb000206 2010-03-30 04:30 2010-03-30 13:50

./dirdat/tb000184 2010-03-30 04:30 First Record

./dirdat/tb000184 2010-03-30 00:00 2010-03-30 04:30

./dirdat/tb000000 \* Initialized \* 2010-03-30 00:00

./dirdat/tb000000 \* Initialized \* First Record

1. Adjust the new Replicat process RTARGET2 to adopt these values, so that the process knows where to start from on startup.

GGSCI (linuxserver1) 79> alter replicat RTARGET2, extseqno 279

REPLICAT altered.

GGSCI (linuxserver1) 80> alter replicat RTARGET2, extrba 43750979

REPLICAT altered.

***Tip:*** *Failure to complete this step will result in either duplicate data or ORA-00001 against the target schema, because GoldenGate will attempt to replicate the data from the beginning of the trail file (*./dirdat/tb000000*) if it exists, else the process will stall.*

1. Start both Replicat processes. Note the use of wildcard (\*)

GGSCI (linuxserver1) 81> start replicat RTARGET\*

Sending START request to MANAGER ...

REPLICAT RTARGET1 starting

Sending START request to MANAGER ...

REPLICAT RTARGET2 starting

1. Check both Replicat processes are running.

GGSCI (linuxserver1) 82> info all

Program Status Group Lag Time Since Chkpt

MANAGER RUNNING

REPLICAT RUNNING RTARGET1 00:00:00 00:00:22

REPLICAT RUNNING RTARGET2 00:00:00 00:00:14

1. Check the detail of the new Replicat processes.

GGSCI (linuxserver1) 83> info REPLICAT RTARGET2, detail

REPLICAT RTARGET2 Last Started 2010-04-08 14:18 Status RUNNING

Checkpoint Lag 00:00:00 (updated 00:00:06 ago)

Log Read Checkpoint File ./dirdat/tb000279

First Record RBA 43750979

Extract Source Begin End

./dirdat/tb000279 \* Initialized \* First Record

./dirdat/tb000279 \* Initialized \* First Record

./dirdat/tb000279 \* Initialized \* 2010-04-08 12:26

./dirdat/tb000279 \* Initialized \* First Record

1. Allow users to connect to the source system and monitor the lag.

GGSCI (linuxserver1) 84> lag REPLICAT RTARGET\*