Oracle GoldenGate - DDL Tracing

The following procedure summarizes Oracle GoldenGate DDL tracing

The page is based on Oracle GoldenGate version 11.2.1.0.1. For a basic configuration I used two Linux VMs (OEL5U6) running single instance Oracle 11.2.0.3 databases. I created both databases using DBCA.

This configuration uses following hosts and databases.

	Source	Target
Hostname	vm4	vm5
Database Name	NORTH	SOUTH

The configuration includes the following on both nodes:

- Creation of a GoldenGate schema owner called GG01.
- Specification of GGSCHEMA as GG01 in GoldenGate parameters
- Creation of GOLDENGATE tablespace which is default tablespace for GG01
- Installation of DDL support:
- omarker_setup.sql
- oddl_setup.sql
- orole_setup.sql
- oddl_enable.sql

The GoldenGate process names are:

	Source	Target
Extract	ex1	-
Data Pump	dp1	-
Replicat	-	rep1

Tracing the DDL Trigger

The DDL trigger is already instrumented with trace statements. It is also possible to enable SQL trace within the trigger.

DDL Trace File

By default trace is written to a file called ggs_ddl_trace.log which is in the directory specified by the USER_DUMP_DEST database parameter.

DDL Trace Level

The DDL trace level determines the amount of trace written by the DDL trigger. The default level is 0 which is writes a minimal amount of trace. Additional trace can be written by setting the trace level to 1 or 2. The documentation recommends that the trace level is only changed under supervision of Oracle support.

Trace messages are written to ggs_ddl_trace.log in the directory specified by the USER_DUMP_DEST database parameter.

To determine the current trace level, use the following query:

```
[oracle@vm4]$ sqlplus gg01/gg01

SQL> SELECT value FROM ggs_setup
WHERE property = 'DDL_TRACE_LEVEL';

VALUE
```

```
--------------------------------------------------------------------------------
0
```

To modify the trace level, use the ddl_tracelevel.sql script.

For example to set the trace level to 1 use:

```
[oracle@vm4]$ cd /home/oracle/goldengate

[oracle@vm4]$ sqlplus gg01/gg01

SQL> @ddl_tracelevel

Set DDL replication trace level script.

Please enter GoldenGate schema name (schema for GoldenGate database objects).
NOTE: GoldenGate schema must be created prior to running this script.

Enter GoldenGate schema name (read above first):GG01

Please enter trace level:1

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Script complete, running verification script...
```

The script calls ddl_status.sql with the GGSCHEMA name to verify the trace configuration. The output from this script has been omitted for clarity.

Check the current trace level using:

```
[oracle@vm4]$ sqlplus gg01/gg01

SQL> SELECT value FROM ggs_setup
WHERE property = 'DDL_TRACE_LEVEL';

VALUE
--------------------------------------------------------------------------------
1
```

To reset the trace level, run ddl_tracelevel.sql again specifying a value of 0.

DDL SQL Trace

SQL trace can also be enabled for the DDL trigger. There are probably more sophisticated ways to enable SQL trace in newer versions. The GoldenGate supplied trace relies on the SQL_TRACE parameter which enables event 10046 level 1. This trace does not include bind variables, wait events or plan statistics (Oracle 11.2 and above).

Trace messages are written to the server process trace file in the directory specified by the USER_DUMP_DEST parameter. For example NORTH_ora_6589.trc.

To check if DDL SQL tracing is currently enabled use:

```
[oracle@vm4]$ sqlplus gg01/gg01

SQL> SELECT value FROM ggs_setup
WHERE property = 'DDL_SQL_TRACING';

VALUE
--------------------------------------------------------------------------------
```

```
0
```

If the value is 0, then trace is disabled; if the value is 1 then trace is enabled.

```
[oracle@vm4]$ cd /home/oracle/goldengate

[oracle@vm4]$ sqlplus gg01/gg01

SQL> @ddl_trace_on

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.

Enter GoldenGate schema name:GG01

You will be prompted for your own tracing identifier (for files in UDUMP directory,
this session only)
Enter your own tracing identifier:DDL

Working, please wait ...
Spooling to file ddl_trace_on_spool.txt


PL/SQL procedure successfully completed.


PL/SQL procedure successfully completed.

1 row updated.

Commit complete.

Session altered.

Script complete, running verification script...
```

The script prompts for an identifier. This is used as the TRACEFILE_IDENTIFIER parameter for the current session. For example:

```
ALTER SESSION SET tracefile_identifier='&identname';
```

Note that the above value will only persist for the duration of the session.

The script calls ddl_status.sql with the GGSCHEMA name to verify the trace configuration. The output from this script has been omitted for clarity.

Check the current trace level using:

```
[oracle@vm4]$ sqlplus gg01/gg01

SQL> SELECT value FROM ggs_setup
WHERE property = 'DDL_SQL_TRACING';

VALUE
--------------------------------------------------------------------------------
1
```

To reset the trace level, run the ddl_trace_off.sql script.

Trace Status

The ddl_status.sql script reports the current status of the parameters and objects required for GoldenGate DDL tracing. The script is also invoked by the ddl_tracelevel.sql, ddl_trace_on.sql and ddl_trace_off.sql scripts.

This script can be executed standalone. For example:

```
[oracle@vm4]$ cd /home/oracle/goldengate

[oracle@vm4]$ sqlplus gg01/gg01

SQL> @ddl_status
Please enter the name of a schema for the GoldenGate database objects:
GG01
Setting schema name to GG01

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 IGNORE TABLE
-----------------------------------
OK

DDL IGNORE LOG TABLE
-----------------------------------
OK

DDLAUX  PACKAGE STATUS:

Line/pos   Error
---------- -------------------------------------------------------------------
No errors  No errors

DDLAUX PACKAGE BODY STATUS:
```

```
Line/pos   Error
---------- -----------------------------------------------------------------
No errors  No errors

SYS.DDLCTXINFO  PACKAGE STATUS:

Line/pos   Error
---------- -----------------------------------------------------------------
No errors  No errors

SYS.DDLCTXINFO  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
----------------------------------
1

DDL TRIGGER TRACE LEVEL
----------------------------------
1

LOCATION OF DDL TRACE FILE
--------------------------------------------------------------------------------
/u01/app/oracle/diag/rdbms/north/NORTH/trace/ggs_ddl_trace.log

Analyzing installation status...


STATUS OF DDL REPLICATION
--------------------------------------------------------------------------------
SUCCESSFUL installation of DDL Replication software components
SQL>
```

GGSCI Dump Commands

The contents of the metadata stored in GGS_DDL_HIST table can be dumped using the DUMPDDL SHOW command in GGSCI.

Consider the following statement:

```
ALTER TABLE t200 ADD c2 VARCHAR2(30);
```

This can be dumped using the GGSCI DUMPDDL command as follows:

```
[oracle@vm4 goldengate]$ ggsci

GGSCI (vm4.juliandyke.com) 1> DBLOGIN USERID us03 PASSWORD us03

GGSCI (vm4.juliandyke.com) 2> DUMPDDL SHOW

*** Dumping DDL Metadata for DDL sequence [1572]...
Time of capture                  = Before DDL
Time of DDL operation            = 2013-04-14 09:44:32
DDL operation (maybe partial)    = [ALTER TABLE t200 ADD c2 VARCHAR2(30) ]
Start SCN of DDL operation       = 2034347
DDL operation type               = ALTER
Object type                      = TABLE
DB Blocksize                     = 8192

Object owner                     = US03
Object name                      = T200
Object ID                        = 77210
Base object owner                = US03
Base object name                 = T200
Data object ID                   = 77210

Object valid                     = VALID
Clustered columns                =
Log group exists                 = 0
Subpartition                     = NO
Partition                        = NO
Total number of columns          = 1
```

```
Number of columns used           = 1

Column #1, name                  = C1
Column #1, ID                    = 1
Column #1, type                  = 2
Column #1, length                = 22
Column #1, is NOT NULL           = 1
Column #1, precision             =
Column #1, scale                 =
Column #1, charset ID            = 0
Column #1, charset form          = 0
Column #1, alternate column ID   = 1
Column #1, alternate name        = C1
Column #1, alternate type        = NUMBER
Column #1, alternate precision   =
Column #1, alternate char used   =
Column #1, alternate XML type    = 0

Finished displaying metadata information (sequence number [1572], DDL history
table [gg01.GGS_DDL_HIST]).
```

The table was originally created using:

```
CREATE TABLE t200 (c1 NUMBER)
```

In the above example the DUMPDDL command shows the columns in the table BEFORE the ALTER TABLE statement was executed.

Output is dependent on the number of columns in the table.

Note that the output is generated from GGS_DDL_HIST. CREATE statements and excluded objects are not stored in this table, so DUMPDDL SHOW will not produce any output in these cases.