Near Zero-Downtime Database Migration and Upgrade

CHAPTER 7

Overview

The need to continue processing transactions while performing major tasks on 24/7 production database server is becoming an obligation for Oracle installations running core business applications and mission critical systems. The objective of this chapter is to cover the techniques of using Oracle GoldenGate and Oracle Database technologies to perform an end-to-end database migration and upgrade with a flexible failback option; leaving no impact on business continuity to achieve near zero-downtime of database availability.

For mission critical systems, Oracle Database upgrade is strategic. The upgrade project plan must be developed in accordance to Oracle software policy life cycle. Database administrators must be aware of Oracle policy milestones and plan their upgrade project accordingly; this keeps their database releases current with Oracle supported platforms for applying fixes and avoiding the additional of premium support services fees. Nevertheless, the upgraded database takes advantage of Oracle software new release features and innovations, which transparently enhances the database functionality, performance and security.

The near zero-downtime migration and upgrade activities is divided into series of related phases; each of the phases composed of several tasks for completion. The list of phases and brief description follows.

1. Deploy Oracle GoldenGate
2. Online clone the production database
3. Migrate to automatic storage management
4. Upgrade to Oracle Database 12c
5. Applications Regressions testing
6. Synchronize the target database
7. Data verification

Though the merge of Phase 2 and 3 is possible, the chapter illustrates them as two separate phases. The order of Phase 3 and 4 is exchangeable, the chapter adapts the order listed above; performing migration first then upgrade of the database. However

Phases 5 is application specific and out of the scope of the book, but discussed briefly.

Deploy Oracle GoldenGate

Oracle GoldenGate deployed as unidirectional topology, data flows from the source system (production) to the target system. The primary extracts create local trail files, the secondary extracts (data pumps) ship the transactions to the target system, no transformations performed on either extracts. The manager process on the source system is configured to optimize the storage level by purging local trail files up on successfully shipped to the target system. The remote trail files are maintained on the target system until the database upgrade is completed and ready to synchronize the target database aiming for a sub-second latency in preparation of application transition phase.

Online Clone the Production Database

Cloning the database is the key for near zero-downtime database upgrade project. It serves two purposes:

1. As an initial-load technique from an active database
2. Eliminates the various types of downtimes

The database accumulative downtime is effected by two time factors: the timeframe of the database upgrade assistant (DBUA) to successfully upgrade the database, and the timeframe of applications functional and technical testing. By cloning the database, it avoids the DBUA shutting down the production database required during the database dictionary upgrade, at the same time it develops a highly reliable and compatible platform for applications testing. The clone is performed using Oracle Recovery Manager (RMAN) command-line with Linux Network File System (NFS) is configured to allow files sharing by RMAN between the source and target system.

Migrate to Automatic Storage Management

It seems Oracle Automatic Storage Management (ASM) is strategic direction for Oracle database storage. The migration from file system to ASM requires the provision of ASM disks by the storage administrator. For the purpose of the implementation, ASM disks are simulated by creating raw files type using the Linux operating system "dd" command. This is an optional phase but highly recommended to be considered and can be executed using either Oracle Recovery Manager (RMAN) or Oracle Enterprise Manager Database control. We will use RMAN to complete the migration from file system to automatic storage management (ASM) environment.

Upgrade to Oracle Database 12c

The upgrade to Oracle Database 12c is executed on the cloned database. Oracle Database 12c binaries are installed on a separate software location (home). Pre-requisites requirements are verified in preparation of launching the Database Upgrade Assistant (DBUA). DBUA has two upgrade techniques, Graphical User Interface (GUI) and command-line; we will be using the recommended approach, GUI. The database restarted to operate on archive log mode and database flashback enabled.

Applications Regressions Testing

There are two types of testing; technical and functional. For technical testing, the adaption of Oracle Real Applications Testing (RAT) Suite produces highly accurate results and quickly. The functional testing is more time consuming. This is because a complete transaction cycle is tested followed by reports generation. It's preferred to conduct both tests on the cloned database for producing highly realistic results. The cloned database flashback feature is enabled for rewinding the database to remove data entered during the functional testing phase.

Synchronize the Target Database

It's necessary to flashback the target database after the functional testing completes. The flashback rewinds the database to the system change number calculated by RMAN clone command. The next step is to apply the remote trail files by starting the Replicat processes from the system change number calculated by RMAN, this avoid the overhead of data collision, if enabled, and Replicat terminates with errors due to duplicate records. Depending on the size of the remote trail, the number of Replicat configured, and the current transaction volume, the timeframe to reach sub-second delay varies which is necessary to achieve before the application transition phase begins.

Data Verification

Now, the Replicats have applied the transactions to the target database; It's time to verify data integrity and accuracy. Configure and deploy Oracle GoldenGate Veridata for data sets confirmation. It's required to disconnect applications servers for a short moment of time to allow the last streams of transactions to be applied to the target database; and this where the near zero-downtime occurs. Execute Oracle GoldenGate Veridata jobs to confirm all data sets are in-sync status, then route the application servers to the new database.

Technical Architecture

Near zero-downtime migrations and upgrades from Oracle database 11g to Oracle database 12c is depicted in Figure 7-1. There are several major components to develop and execute for each phase. The Oracle technologies used to complete the migration and upgrade:

- Oracle GoldenGate 12c
- Oracle GoldenGate 11g Veridata
- Oracle Database Automatic Storage Management (ASM)
- Oracle Database Recovery Manager (RMAN)
- Oracle Database Configuration Assistant (DBCA)
- Oracle Database Upgrade Assistant (DBUA)
- Oracle Database Flashback Technology
- Oracle Database Real Application Testing (RAT)

Keep in mind, depending on the scope of the project; some of the above components may becomes optional or adapted using alternative technologies. But, since this chapter covers Oracle-to-Oracle database upgrade, only the above list of Oracle components are adapted.

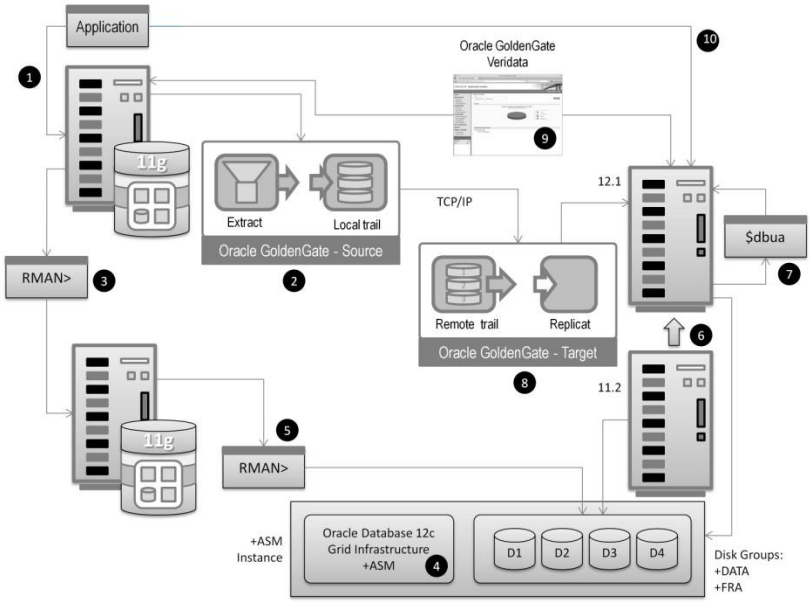


Figure 7-1: Near zero-downtime migration and upgrade architecture

Figure 7-1 illustrates highly-available applications running on the production system (Source – S1E2 database). Oracle GoldenGate unidirectional configuration deployed, the primary extracts capture the transactions logs, the data pumps send data to remote system (Target). Oracle Database 12c Automatic Storage Management (ASM) instance manages database storage (Disk Groups). Online cloning of the production database leaves this database intact, and avoids downtimes occurred during the database data dictionary upgrade. Oracle Database 12c software is installed on the target system, the upgrade of the database executed using DBUA. The Replicat applies the remote trail files starting from the System Change Number (SCN) identified by the Oracle Recovery Manager (RMAN) DUPLICATE command. Oracle GoldenGate Veridata ascertain the source and target database are identical before diverting the applications from the source to the target database.

Transactions Workload Simulator

The ultimate goal of this chapter is to implement near zero-downtime migration and upgrade using Oracle GoldenGate while users are concurrently processing business transactions. Hence the transactions simulator, TRNAS_LOAD will remain running on the production system (S1E2 instance) until the upgrade is completed, then interrupted and redirected to the target system (T1E2 instance). Oracle GoldenGate deployed to capture and ship the transactions to the target system in preparation to be applied after successfully upgrading the database from Oracle Database 11g to Oracle Database 12c.

Start by opening a terminal windows and execute the following anonymous PL/SQL block on the source system (production). *Do not close this terminal windows, it must remains open for the duration of the migration and upgrade tasks.*

```
SQL> CONN osm$repapi/oracle
Connected.
SQL> DESC trans_load
PROCEDURE trans_load
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 PLOAD_LEVEL                    NUMBER                  IN     DEFAULT
 PSTARTFROM                     NUMBER                  IN     DEFAULT

SQL> BEGIN
  2     FOR i IN 100001..1000000 LOOP
  3         TRANS_LOAD(1, i);
  4         DBMS_LOCK.SLEEP(2);
  5     END LOOP;
  6  END;
  7  /
```

Using the procedure DBMS_LOCK.SLEEP(2) allows an elapsed time of two seconds of delay for every transaction. The above transactions volume should lasts for the

duration of the upgrade. Our objective is to ultimately interrupt the load simulator TRANS_LOAD explicitly and move it to the migrated and upgraded environment.

Phase I: Deploy Oracle GoldenGate

Successfully deploying Oracle GoldenGate is a key component of near zero-downtime migration and upgrade. While the applications continue to deliver business transactions, Oracle GoldenGate captures the transactions logs and send data to the target system. For applications with complex database design such as enterprise ERP where it's challenging to maintain the order of foreign keys referential integrity constraints, it's required the target database foreign key constraints remain disabled during the synchronization process and successfully enabled before the applications redirected to the new database. The disable/enable foreign key constraints step is developed using database scripts generated from the database dictionary.

Figure 7-2 shows Oracle GoldenGate deployed as a unidirectional classical capture configuration. Another alternative is to employ Oracle GoldenGate integrated capture where Oracle GoldenGate is installable on a down-stream system avoiding the additional overhead to the production server as a result of transactions logs capture processing and local trail storage. Refer to chapter 12 for details regarding Oracle GoldenGate integrated capture implementation.

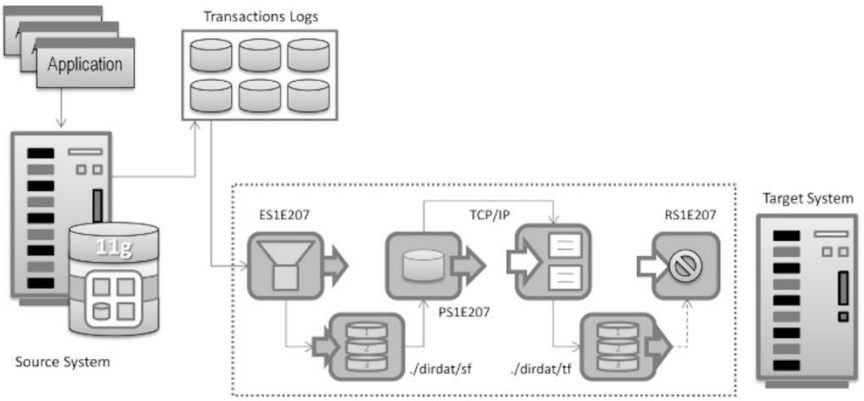


Figure 7-2: Unidirectional setup

The configuration composed of the one or more primary extracts, one or more secondary extracts (data pumps). The primary extracts reads the transactions logs for the configured list of database tables, and create the corresponding local trails. The data pump extracts, over TCP/IP ship the data to the remote system creating the corresponding remote trails. The Replicat processes remains stopped until the target

database (cloned) is upgraded from Oracle Database 11g to Oracle Database 12c. Though the use of data pumps is optional, but it's highly recommended for near zero-downtime implementation to eliminate production server overhead in event of network unavailability to the target system. Phase I consists of the following tasks.

1. Oracle GoldenGate software installation
2. Prepare of the source database
3. Enable table-level logging
4. Configure and test capture, route and delivery
5. Start primary Extract (ES1E207)
6. Start data pump Extract (PS1E207)

Oracle GoldenGate Software Installation

Successful installation of Oracle GoldenGate on the source (production) and target system must be completed to proceed further. Consider the platform architecture, the chapter uses Oracle GoldenGate 12c for Oracle Database 11g 64-bit, the same bundle of Oracle GoldenGate 12c is used for Oracle Database 12c 64-bit. Refer to chapter 2 to learn about installing Oracle GoldenGate 12c.

Prepare of the Source Database

Prior to configuring Oracle GoldenGate Extracts and Replicats, the following database related tasks must be completed.

- Enable database supplemental logging
- Create Oracle GoldenGate database user

Database Supplemental Logging

Because the database supplemental logging is not enabled by default when the database created, it's required to enable the minimum database supplemental logging manually. The database supplemental logging records the primary key information, for update and delete operations; Oracle GoldenGate locates the target row to update or delete on the target database using the primary key.

Query the data dictionary dynamic view V$DATABASE to find out the level of supplemental logging.

```
S1E2> SELECT supplemental_log_data_min "Minimum",
   2         supplemental_log_data_pk  "Primary key",
   3         supplemental_log_data_ui  "Unique Key",
   4         supplemental_log_data_fk  "Foregin Key",
```

```
   5         supplemental log data all "All"
   6  FROM   v$database;

Minimum  Pri Uni For All
-------- --- --- --- ---
NO       NO  NO  NO  NO

S1E2>
```

Refer to Chapter 1 for more details regarding supplemental logging.

Oracle GoldenGate Database User

The use of a dedicated database user for Oracle GoldenGate is highly recommended. Avoid using an existing active database user. This user requires database privileges listed on Table 7-1.

Database Privilege	Source Database	Target Database
CREATE SESSION, ALTER SESSION	X	X
RESOURCE	X	X
CONNECT	X	X
SELECT ANY DICTIONARY	X	X
FLASHBACK ANY TABLE or FLASHBACK ON<owner.table>	X	
SELECT ANY TABLE or SELECT ON <owner.table>	X	X
INSERT, UPDATE, DELETE ON <target.table>		X
CREATE TABLE		X
EXECUTE on DBMS_FLASHBACK package	X	

Table 7-1: Database Privileges

Refer to Chapter 2 for more details regarding creating Oracle GoldenGate database user.

Enable Table-Level Logging

Table-level is added from Oracle GoldenGate GGSCI command-line interface. The command ADD TRANDATA adds table-level logging using different techniques:

- Specify the schema and table name
- Specify the schema followed by Oracle GoldenGate wildcard character
- Execute ADD TRANDATA using an obey file

The ADD TRANDATA command requires login to the database from within GGSCI.

```
GGSCI (ggs-source) 1> DBLOGIN USERID ggs_admin, PASSWORD oracle
Successfully logged into database.

GGSCI (ggs-source) 2>
```

Refer to Chapter 2 for more details regarding adding table-level logging.

Configure and Test Capture, Route and Delivery

The source database has been successfully prepared for Oracle GoldenGate capture. The configuration uses a data pump process to send data to the target system, this enables the capture process to continue operating in the event the target system becomes unreachable for a considerable timeframe, it also offloads the primary extract from TCP/IP communications with the target's server collector. The primary process become dedicated for capturing and creating the local trail only. The data pump process continues resending data up on the restoration of the network.

The next three sections implement the configuration. The delivery process remains stopped until the database upgrade successfully completed and it's ready to apply transactions to the target database.

- Configure the primary Extract
- Configure the secondary Extract
- Configure the Replicat

Begin by analyzing the database schemas and their tables. This enables the capture of schema by schema, mapping Extract to schema and identifying unsupported data types. The following configuration uses the schema OSM$REPAPI.

```
SQL> SELECT table_name
  2  FROM   user_tables;

TABLE_NAME
------------------------------
RANDOM_DOMAIN
RANDOM_FIRST
RANDOM_MIDDLE
RANDOM_LAST
POLICY_TYPES
DISCOUNT_TYPES
CUSTOMERS
POLICIES
OSM_DOCUMENTS

9 rows selected.

SQL>
```

Configure the Primary Extract

The primary Extract reads the transactions logs (captures) and writes to the local trail. The Extract begins capture from the time it's successfully created, or altered to begin just before starting the database cloning task, and continues to capture while the database upgrade is in progress.

Prepare the primary Extract parameter file, it identifies the local trail and the list of target tables. For near zero-downtime implementation, the inclusion of all tables is likely required. However; to exclude specific tables, use the parameter TABLEEXECLUDE followed by the list of table to include. The TABLE parameter uses the Oracle GoldenGate wildcard pattern represented by OSM$REPAPI.* to include the remaining tables.

```
EXTRACT es1e207
SETENV (ORACLE_SID="S1E2")
SETENV (ORACLE_HOME="/u01/app/oracle/product/11.2.0/dbhome_1")
SETENV (NLS_LANG="AMERICAN_AMERICA.AL32UTF8")
USERID ggs_admin, PASSWORD oracle
EXTTRAIL ./dirdat/sf
TABLEEXCLUDE osm$repapi.OSM_DOCUMENTS;
TABLE osm$repapi.*;
```

Next, create the primary Extract ES1E207. Because it's a single-instance database, the parameter THREADS 1 is specified. The Extract starts transactions logs capture up on its successfully created, but it may be altered to begin capture just before the database cloning begins.

```
GGSCI (ggs-source) 4> ADD EXTRACT es1e207, TRANLOG, BEGIN now, THREADS 1
EXTRACT added.
```

Associate the primary Extract ES1E207 to its local trail. Specify the maximum size of the local trail file. The default size is 100MB.

```
GGSCI (ggs-source) 6> ADD EXTTRAIL ./dirdat/sf, EXTRACT es1e207, MEGABYTES
16
EXTTRAIL added.
```

Configure the Secondary Extract

It's highly recommended configuring secondary Extracts – data pumps. Prepare the parameter file. Because of no transformation is required, specify the parameter PASSTHRU mode. The target system is specified by the parameter RMTHOST, followed by the Oracle GoldenGate manager listening port. Use the same TABLEEXECLUDE and TABLE parameters to identify the list of tables.

```
EXTRACT ps1e207
USERID ggs_admin, PASSWORD oracle
PASSTHRU
RMTHOST 192.168.1.209, MGRPORT 7812
RMTTRAIL ./dirdat/tf
TABLEEXCLUDE osm$repapi.OSM_DOCUMENTS;
TABLE osm$repapi.*;
```

Create the data pump Extract. The data pump reads the local trail and send data to the remote system.

```
GGSCI (ggs-source) 5> ADD EXTRACT ps1e207, EXTTRAILSOURCE ./dirdat/sf
EXTRACT added.
```

Associate the data pump Extract with its remote trail. Specify the maximum size of the local trail file. The default size is 100MB. For reattempt of applying the transaction to the target database, do not configure purge of the remote trails files until the database upgrade is completed.

```
GGSCI (ggs-source) 6> ADD RMTTRAIL ./dirdat/tf, EXTRACT ps1e207, MEGABYTES
16
RMTTRAIL added.
```

Configure the Replicat

Prepare the Replicat parameter file. Because the Replicat requires a database table for checkpoint, an initial database is created for testing the Replicat. Because database cloning is treated as the technique for initial data load, do not include the HANDLECOLLISIONS parameter, instead the Replicat is altered to begin applying transactions using the System Change Number (SCN) of the database cloning, this saves unnecessary data collision overhead.

```
REPLICAT rs1e207
SETENV (ORACLE_SID="T1C2")
SETENV (ORACLE_HOME="/u01/app/oracle/product/11.2.0/dbhome_1")
SETENV (NLS_LANG="AMERICAN_AMERICA.AL32UTF8")
USERID ggs admin, PASSWORD oracle
ASSUMETARGETDEFS
DISCARDFILE ./dirrpt/rs1e207.dsc, PURGE
MAPEXCLUDE osm$repapi.OSM_DOCUMENTS;
MAP osm$repapi., TARGET osm$repapi.*;
```

```
GGSCI (ggs-target) 3> ADD REPLICAT rs1e207, EXTTRAIL ./dirdat/tf
REPLICAT added.
```

Next, start the primary and secondary Extracts. The Replicat remains stopped until database upgraded successfully completed.

Start the Primary Extract, ES1E207

Now, start the primary Extract. Make the status is RUNNING and it's updating the checkpoint timestamp.

```
GGSCI (ggs-source) 31> START EXTRACT es1e207

Sending START request to MANAGER ...
EXTRACT ES1E207 starting

GGSCI (ggs-source) 32> INFO EXTRACT es1e207

EXTRACT    ES1E207   Last Started 2014-04-11 11:24   Status RUNNING
Checkpoint Lag       00:00:02 (updated 00:00:09 ago)
Process ID           32142
Log Read Checkpoint  Oracle Redo Logs
                     2014-04-11 11:25:11  Thread 1, Seqno 92, RBA 1688972
                     SCN 0.3623058 (3623058)
```

Start the Data Pump Extract, PS1E207

Then, start the data pump Extract. Make the status is RUNNING and it's updating the checkpoint timestamp.

```
GGSCI (ggs-source) 33> START EXTRACT ps1e207

Sending START request to MANAGER ...
EXTRACT PS1E207 starting

GGSCI (ggs-source) 34> INFO EXTRACT ps1e207

EXTRACT    PS1E207   Last Started 2014-04-11 11:25   Status RUNNING
Checkpoint Lag       00:00:00 (updated 00:01:31 ago)
Process ID           32216
Log Read Checkpoint  File ./dirdat/sf000000
                     2014-04-11 11:24:22.000000  RBA 491227
```

Verify the Remote Trail

Use the Oracle GoldenGate Logdump utility to ascertain the transactions are captured and sent to the target system.

```
$ ls -al
total 476
drwxr-x---  2 oracle oinstall   4096 Apr 11 17:35 .
drwxr-xr-x 28 oracle oinstall   4096 Apr  1 23:42 ..
-rw-r-----  1 oracle oinstall 474974 Apr 11 17:37 tf000000
$
```

The following displays transactions details using Oracle GoldenGate Logdump utility. Perform the same verification on several remote trail files.

Open the current remote trail file

Open the trail file. Logdump utility open one trail file at a time.

```
Logdump 3 >OPEN ./dirdat/tf000000
Current LogTrail is /u01/app/ggs/12.1.0/gghome_1/dirdat/tf000000
```

Display transactions summary

Display summary details, each trail file indicate the total number of records and each type of DML size.

```
Logdump 4 >COUNT
LogTrail /u01/app/ggs/12.1.0/gghome_1/dirdat/tf000000 has 2017 records
Total Data Bytes            244700
  Avg Bytes/Record             121
Insert                        2016
Others                           1
After Images                  2016

Average of 2017 Transactions
    Bytes/Trans .....         169
    Records/Trans ...           1
    Files/Trans .....           1

Logdump 5 >
```

Table-level transactions details

This is an important Logdump utility command. It shows the details for each table in terms of DML transactions details.

```
Logdump 6 >COUNT, DETAIL
LogTrail /u01/app/ggs/12.1.0/gghome_1/dirdat/tf000000 has 2017 records
Total Data Bytes            244700
  Avg Bytes/Record             121
Insert                        2016
Others                           1
After Images                  2016

Average of 2017 Transactions
    Bytes/Trans .....         169
    Records/Trans ...           1
    Files/Trans .....           1

*FileHeader*                                        Partition 0
Total Data Bytes              1456
  Avg Bytes/Record            1456
```

```
Others                                1

OSM$REPAPI.CUSTOMERS                                    Partition 4
Total Data Bytes                 97326
  Avg Bytes/Record                  96
Insert                            1008
After Images                      1008

OSM$REPAPI.POLICIES                                     Partition 4
Total Data Bytes                145918
  Avg Bytes/Record                 144
Insert                            1008
After Images                      1008
Logdump 7 >
```

Refer to Chapter 14 for more details regarding Oracle GoldenGate Logdump utility.

Phase II: Online Clone of the Source Database

Achieving near zero-downtime upgrade requires cloning the source database, which is an image copy of the database at a timestamp, referred to as the System Change Number (SCN). The primary purpose of cloning the source database is to avoid the database shutdown by the Database Upgrade Assistant (DBUA) during the database data dictionary upgrade. There are several techniques for cloning the database; However, this chapter employ Oracle Recovery Manager (RMAN) DUPLICATE command to clone the database S1E2 and create the target database T1C1. The database clone is an online operation, it does not incur database shutdown. Figure 7-3 presents clone database from source to target system.

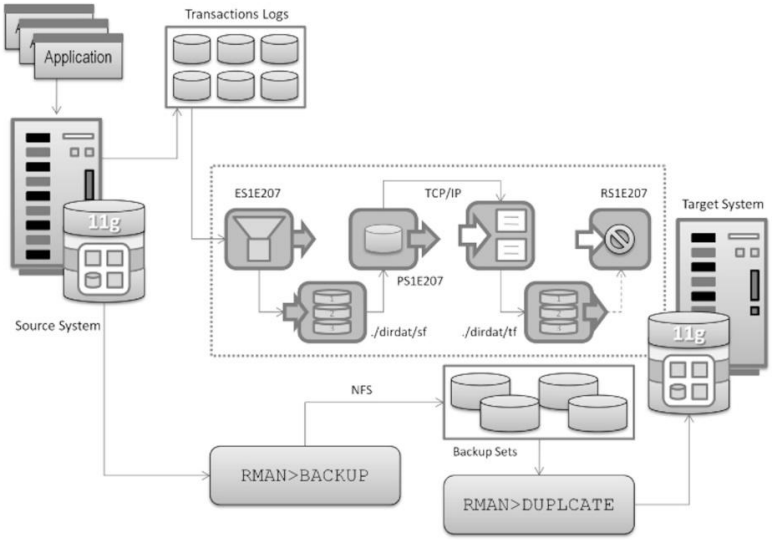


Figure 7-3: Database cloning to target system using RMAN

The cloned database uses a different Unique Database Identifier (DBID) from the source database. Cloning the database performed either from backup or from an active database instance. It's created on the same system or remotely. The clone database is one of the initial data load techniques, it includes the complete database contents. There are four techniques for cloning the database:

1. From an active database, connected to the source and target instances
2. From backup, connected to the source and target instances
3. From backup, connected to the target instance and recovery catalog
4. From backup, connected to the target instance

The chapter uses technique 2, From backup, connected to the source and target instance. Database and archive log files source database backup is completed using RMAN. Using Linux File System (NFS), the backup directory is shared on the remote system.

When database cloning is performed outside of this context, the source database/instance is referred to as the 'Target' and the target database/instance is referred to as the 'Auxiliary'.

Another option of cloning the database is using Oracle Enterprise Manager Database Control (OEM). This is a Graphical User Interface (GUI) and commonly used by less experienced Database Administrators (DBA). Figure 7-4 presents cloning the database using OEM.

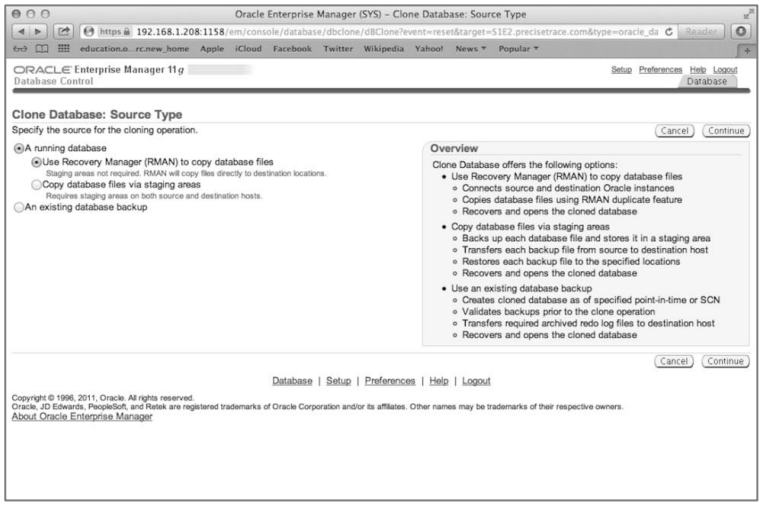


Figure 7-4: Database cloning using Oracle Enterprise Manager (OEM)

Because the cloned database created on a different system, configuring Linux Network File System server and client is illustrated first.

Install Oracle Database 11g

The database clone on a separate system requires the installation of Oracle Database 11g on the target system. The cloned database delivers three functions:

- Initial data load from an active source database
- Migration and upgrade platform
- Application testing platform

Install Oracle Database 11g software only, do not attempt to create the target database as it will be created as cloned database. Figure 7-5 shows Oracle Database 11g Universal Installer, Step 3 of 10 select the 'install database software only'.

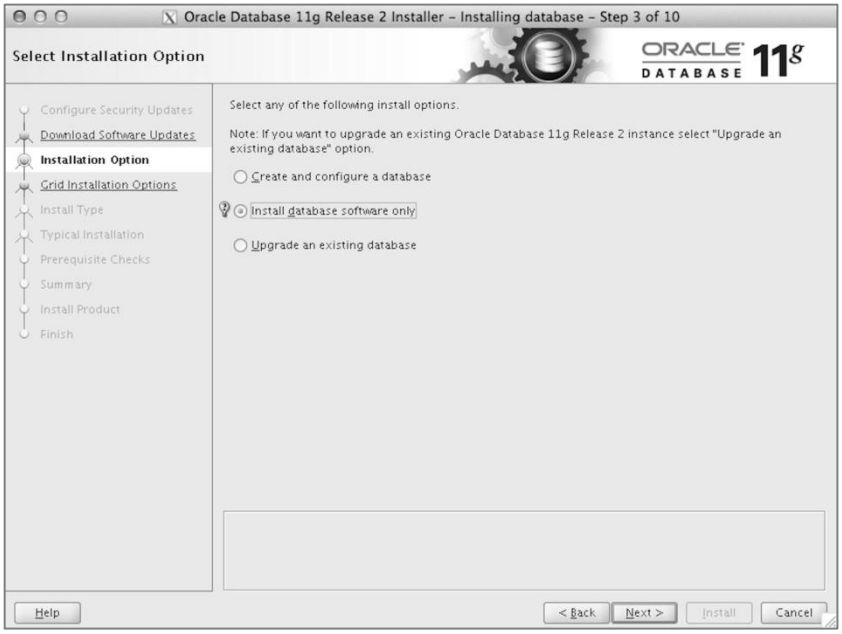


Figure 7-5: Oracle Database 11g Installation

Configure Linux Network File System

The purpose of configuring Network File System (NFS) is to allow the target system shares directories (mount points) on the source system as if it's stored on the local

system storage. This enable us to quickly access the backup files created by RMAN on the source system database when creating backup-based clone database.

Considering the source system IP address is 192.168.1.213/192.168.1.208 and the target system is 192.168.1.214/192.168.1.209, execute the steps as follows:

Source System (Server)

Login as root user and execute the steps below:

- Create the shared directory used for RMAN backup files destination.

```
# mkdir –p /home/oracle/backup
# cd /home/oracle
# chown –R oracle:oinstall backup
```

- Allows the target system to mount the directory by modifying the file /etc/exports. Add the below line indicating the client IP address and allowed mount options.

```
/home/oracle/backup    192.168.1.214 (rw, sync)
```

- Restart NFS services to activate the configuration.

```
# /sbin/service nfs restart
Shutting down NFS mountd:                                  [  OK  ]
Shutting down NFS daemon:                                  [  OK  ]
Shutting down NFS quotas:                                  [  OK  ]
Shutting down NFS services:                                [  OK  ]
Starting NFS services:                                     [  OK  ]
Starting NFS quotas:                                       [  OK  ]
Starting NFS daemon:                                       [  OK  ]
Starting NFS mountd:                                       [  OK  ]
# /sbin/service portmap restart
Stopping portmap:                                          [  OK  ]
Starting portmap:                                          [  OK  ]
#
```

- Verify NFS configuration settings.

```
# /usr/sbin/showmount -e 192.168.1.213
```

Target System (Client)

Login as root user and execute the steps below.

- Create the directory used for RMAN backup files destination

```
# mkdir -p /home/oracle/backup
# cd /home/oracle
# chown -R oracle:oinstall backup
```

- Mount the directory /home/oracle/backup on the source system. Specify the IP address of the server followed by the source and target shared directory.

```
# mount -t nfs 192.168.1.213:/home/oracle/backup /home/oracle/backup
```

- Run RMAN backup command to test NFS setup by specifying the backup location as the NFS mounted directory, /home/oracle/backup.

```
$ rman

Recovery Manager: Release 11.2.0.3.0 - Production on Sat Apr 12 09:48:44
2014

Copyright (c) 1982, 2011, Oracle and/or its affiliates.  All rights
reserved.

RMAN> CONNECT TARGET

connected to target database: S1E2 (DBID=4279993164)

RMAN> BACKUP TABLESPACE users
2> FORMAT '/home/oracle/backup/user_%U.dbf';

Starting backup at 12-APR-14
using channel ORA_DISK_1
channel ORA_DISK_1: starting full datafile backup set
channel ORA_DISK_1: specifying datafile(s) in backup set
input datafile file number=00004
name=/u01/app/oracle/oradata/S1E2/users01.dbf
channel ORA_DISK_1: starting piece 1 at 12-APR-14
channel ORA_DISK_1: finished piece 1 at 12-APR-14
piece handle=/home/oracle/backup/user_03p5hkkf_1_1.dbf
tag=TAG20140412T095111 comment=NONE
channel ORA_DISK_1: backup set complete, elapsed time: 00:00:12
Finished backup at 12-APR-14

RMAN>
```

- Verify RMAN backup is created on the target system. Change the permission on the backup directory to avoid permission denied by the operating system during restore operation.

```
$ cd /home/oracle/backup
$ ls -al
total 100404
drwxrwxrwx 2    500      501      4096 Apr 12 08:51 .
drwx------ 4 oracle oinstall      4096 Apr 12 00:56 ..
-rw-r----- 1    500      501 102694912 Apr 12 08:51
user_03p5hkkf_1_1.dbf
```

```
$ chown oracle:oinstall *
```

Creating a Backup-Based Clone Database

A backup-based clone database uses an online backup of the database files and archived log files. Because the target database (clone) is located on a separate system, override the default backup destination, Fast Recovery Area (FRA), using the FORMAT parameter to specify a shared directory configured using Network File System (NFS). The NFS shared directory ensures the backup is immediately available for RMAN DUPLICATE command to clone the database.

The following RMAN script performs an online backup of the database file and archived log file. It begins by allocating three channels to enable backup parallelism. The script begins backup of the database files, followed by the archived log files. Both backups are written to the destination '/home/oracle/backup' accessible by the target system as an NFS mounted directory. Up on successful completion of the backup, the channels allocated are released.

Connect using the TARGET option which designates the source database. Verify the database name and DBID returned.

```
RMAN> CONNECT TARGET

connected to target database: S1E2 (DBID=4279993164)

RMAN> RUN {
2> ALLOCATE CHANNEL c1 TYPE DISK;
3> ALLOCATE CHANNEL c2 TYPE DISK;
4> ALLOCATE CHANNEL c3 TYPE DISK;
5>    BACKUP DATABASE INCLUDE CURRENT CONTROLFILE
6>    FORMAT '/home/oracle/backup/db_bck_%U.dbf';
7>    SQL 'ALTER SYSTEM SWITCH LOGFILE';
8>    BACKUP ARCHIVELOG ALL
9>    FORMAT '/home/oracle/backup/db_bck_%U.arc';
10> RELEASE CHANNEL c1;
11> RELEASE CHANNEL c2;
12> RELEASE CHANNEL c3;
13> }
```

From the target system, verify the backup created and backup files owner, group and permission are correct. Otherwise manually execute the operating system shell commands 'chown' and 'chmod' for correcting the backup files settings.

```
# cd /home/oracle/backup
# chown oracle:oinstall *; chmod 777 *
```

Now, let's divide the database clone into 3 related phases:

1. Configure the target instance
2. Start the target instance in NOMOUNT mode
3. Perform clone database using RMAN

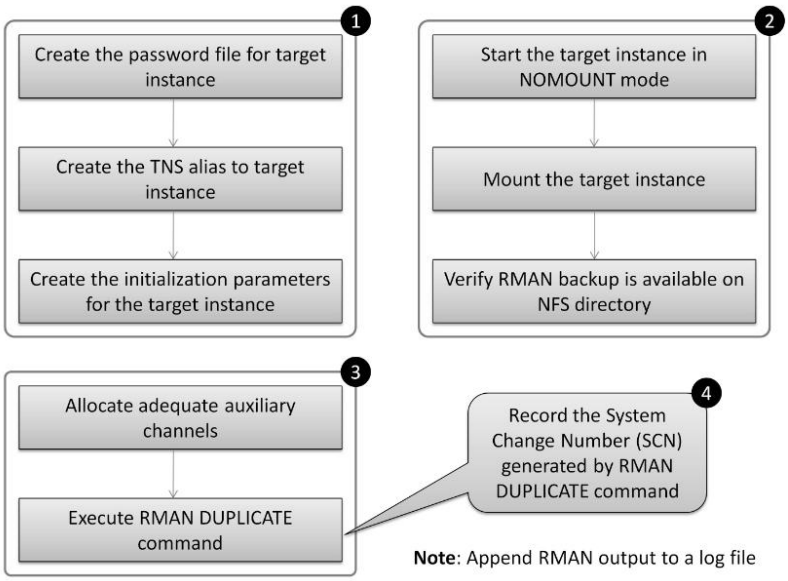


Figure 7-5: Clone Database from Backup

Configure the Target Instance

The configuration prepares the target instance for database cloning. Use the Oracle password utility ORAPWD to create the password file, the Oracle NETMGR or NETCA utility to create the TNS database connect string and create the parameter file, 'initS1E2.ora'.

- Create the password file on the default location $ORACLE_HOME/dbs

```
$ pwd
/u01/app/oracle/product/11.2.0/dbhome_1/dbs
$ orapwd file=orapwT1C1.ora password=oracle_4U force=y
$ ls
init.ora  orapwT1C1.ora
```

- Create the listener on the target system, then create the database connect strings from source to target and vice versa; Test the database connect strings using Oracle TNSPING utility.

  Source to target connect string testing.

```
$ tnsping T1C1

TNS Ping Utility for Linux: Version 11.2.0.3.0 - Production on 18-APR-
2014 03:08:06

Copyright (c) 1997, 2011, Oracle.  All rights reserved.

Used parameter files:

Used TNSNAMES adapter to resolve the alias
Attempting to contact (DESCRIPTION = (ADDRESS_LIST = (ADDRESS =
(PROTOCOL = TCP)(HOST = 192.168.1.209)(PORT = 1521))) (CONNECT_DATA =
(SERVICE_NAME = T1C1.precisetrace.com)))
OK (40 msec)
[oracle@ggs-source ~]$
```

Target to source connect string testing

```
$ tnsping S1E2

TNS Ping Utility for Linux: Version 11.2.0.3.0 - Production on 18-APR-
2014 10:09:57

Copyright (c) 1997, 2011, Oracle.  All rights reserved.

Used parameter files:

Used TNSNAMES adapter to resolve the alias
Attempting to contact (DESCRIPTION = (ADDRESS_LIST = (ADDRESS =
(PROTOCOL = TCP)(HOST = 192.168.1.208)(PORT = 1521))) (CONNECT_DATA =
(SERVICE_NAME = S1E2.precisetrace.com)))
OK (10 msec)
$
```

- Create the target parameter file from the source database instance, then modified it to suite the target database.

```
SQL> CONN / AS SYSDBA
Connected.

SQL> CREATE pfile FROM spfile;

File created.

SQL>

*.audit_file_dest='/u01/app/oracle/admin/T1C1/adump'
*.audit_trail='db'
*.compatible='11.2.0.0.0'
*.control_files='/u01/app/oracle/oradata/T1C1/control01.ctl','/u01/app/o
racle/fast_recovery_area/T1C1/control02.ctl'
*.db_block_size=8192
*.db_domain='precisetrace.com'
*.db_name='T1C1'
*.db_recovery_file_dest='/u01/app/oracle/fast_recovery_area'
```

```
*.db_recovery_file_dest_size=4322230272
*.diagnostic_dest='/u01/app/oracle'
*.dispatchers='(PROTOCOL=TCP) (SERVICE=T1C1XDB)'
*.log_archive_format='%t_%s_%r.dbf'
*.memory_target=2147483648
*.open_cursors=300
*.processes=150
*.remote_login_passwordfile='EXCLUSIVE'
*.undo_tablespace='UNDOTBS1'
*.db_file_name_convert='/u01/app/oracle/oradata/S1E2/',
                       '/u01/app/oracle/oradata/T1C1/'
*.log_file_name_convert='/u01/app/oracle/oradata/S1E2/',
                       '/u01/app/oracle/oradata/T1C1/'
```

Modify, save the file as initT1C1.ora and copy it over to the target system.

```
$ scp initT1C1.ora
oracle@192.168.1.209:/u01/app/oracle/product/11.2.0/dbhome_1/dbs
```

Create the directory structure as needed for the database instance T1C1.

```
$ cd $ORACLE_BASE/admin
$ mkdir -p T1C1/cdump T1C1/udump T1C1/adump
$ cd T1C1
$ ls
adump  cdump  udump
$ cd $ORACLE_BASE/oradata
$ mkdir T1C1
$ cd $ORACLE_BASE
$ cd fast_recovery_area/
$ mkdir T1C1
$
```

Startup the Target Instance

Start the target instance using the parameter file 'initT1C1.ora'. The instance is started in NOMOUNT mode. Then create the server parameter file (spfileT1C1.ora). Shutdown and startup the instance in MOUNT mode using the parameter file spfileT1C1.ora.

```
$ export ORACLE_HOME=/u01/app/oracle/product/11.2.0/dbhome_1
$ export ORACLE_SID=T1C1
$ sqlplus /nolog

SQL*Plus: Release 12.1.0.1.0 Production on Fri Apr 18 09:38:42 2014

Copyright (c) 1982, 2013, Oracle.  All rights reserved.

SQL> CONN / AS SYSDBA
Connected to an idle instance.
SQL> STARTUP NOMOUNT
pfile='/u01/app/oracle/product/11.2.0/dbhome_1/dbs/initT1C1.ora'
ORACLE instance started.
```

```
Total System Global Area   551165952 bytes
Fixed Size                   2230232 bytes
Variable Size              331352104 bytes
Database Buffers           213909504 bytes
Redo Buffers                 3674112 bytes
SQL>

SQL> CREATE spfile FROM PFILE;

File created.

SQL>
```

Perform Clone Database using RMAN

Now we are ready to perform the database clone. While logged in to the target system, connect to the target instance (T1C1) and remotely connect to the source instance (S1E2). The following is RMAN DUPLICATE command runs on verbose mode. To avoid permission denied error returned from the operating system while RMAN restores backup files, prior to staring the command ascertain the correct permission settings on target system NFS directory /home/oracle/backup.

```
$ export ORACLE_HOME=/u01/app/oracle/product/11.2.0/dbhome_1
$ cd $ORACLE_HOME/bin
$ ./rman

Recovery Manager: Release 11.2.0.3.0 - Production on Fri Apr 18 10:14:59
2014

Copyright (c) 1982, 2011, Oracle and/or its affiliates.  All rights
reserved.

RMAN> CONNECT AUXILIARY sys/oracle_4U

connected to auxiliary database: T1C1 (not mounted)

RMAN> CONNECT TARGET sys/oracle_4U@S1E2

connected to target database: S1E2 (DBID=4279993164)

RMAN>

RMAN> RUN {
2> DUPLICATE TARGET DATABASE TO T1C1
3> BACKUP LOCATION '/home/oracle/backup'
4> NOFILENAMECHECK;
5> }
```

The above RMAN DUPLICATE script overrides the default backup location, the Fast Recovery Area (FRA), by specifying non-default destination using the parameter BACKUP LOCATION. The parameter NOFILENAMECHECK prevents RMAN from checking whether source database data files with the same name as clone

database data files are in use. Specify this option when the source database and clone database data files and redo log files will have the same names. Because the clone database is on a different system that has the same disk configuration, file system structure, and file names as the source database system. Without the parameter NOFILENAMECHECK, the command will terminate with an error.

If you receive the error message, RMAN-06025: no backup of archived log for thread 1 with sequence 255 and starting SCN of 8375157 found to restore. Then use the NOREDO option as illustrated below.

```
RMAN> RUN {
2> DUPLICATE TARGET DATABASE TO T1C1
3> NOREDO
4> BACKUP LOCATION '/home/oracle/backup'
5> NOFILENAMECHECK;
6> }
```

Target Replicat Starting SCN

The above RMAN DUPLICATE command determined the System Change Number (SCN) for performing point in-time recovery and open the target database in RESETLOGS mode.

The following section of RMAN DUPLICATE command log shows the SCN 14274705, this will be the applying point by the Replicat for synchronizing the target database – T1C1.

```
contents of Memory Script:
{
   set until scn 14274705;
   set newname for datafile  1 to
 "/u01/app/oracle/oradata/T1C1/system01.dbf";
   set newname for datafile  2 to
 "/u01/app/oracle/oradata/T1C1/sysaux01.dbf";
   set newname for datafile  3 to
 "/u01/app/oracle/oradata/T1C1/undotbs01.dbf";
   set newname for datafile  4 to
 "/u01/app/oracle/oradata/T1C1/users01.dbf";
   restore
   clone database
   ;
}
```

Post Database Clone

Modify the file /etc/oratab to include the clone database reference. To avoid a re-clone database in the event of failure, start a consistent database backup using

RMAN. First, shutdown the database using IMMEDIATE option, then startup in MOUNT state.

```
RMAN> STARTUP MOUNT

connected to target database (not started)
Oracle instance started
database mounted

Total System Global Area     551165952 bytes

Fixed Size                     2230232 bytes
Variable Size                369100840 bytes
Database Buffers             176160768 bytes
Redo Buffers                   3674112 bytes

RMAN> BACKUP DATABASE;
```

Verify the Clone Database Structure

Because RMAN DUPLICATE command did not exclude any of the source database tablespaces, after the clone database operation successfully completed, the clone database (target) must have identical physical data files structures.

The following is the list of the clone database data files, control files, online redo log files, list of tables owned by OSM$REPAPI and the current row counts of the table POLICIES.

```
$ . oraenv
ORACLE_SID = [oracle] ? T1C1
The Oracle base remains unchanged with value /u01/app/oracle
$ sqlplus /nolog

SQL*Plus: Release 11.2.0.3.0 Production on Fri Apr 18 18:51:14 2014

Copyright (c) 1982, 2011, Oracle.  All rights reserved.

SQL>CONN / AS SYSDBA
Connected.
SQL> SELECT file_name FROM dba_data_files UNION
  2  SELECT file_name FROM dba_temp_files UNION
  3  SELECT name      FROM v$controlfile  UNION
  4  SELECT member    FROM v$logfile;

FILE_NAME
--------------------------------------------------------------------------------
/u01/app/oracle/fast_recovery_area/T1C1/control02.ctl
/u01/app/oracle/oradata/T1C1/control01.ctl
/u01/app/oracle/oradata/T1C1/redo01.log
/u01/app/oracle/oradata/T1C1/redo02.log
/u01/app/oracle/oradata/T1C1/redo03.log
/u01/app/oracle/oradata/T1C1/sysaux01.dbf
/u01/app/oracle/oradata/T1C1/system01.dbf
```

```
/u01/app/oracle/oradata/T1C1/temp01.dbf
/u01/app/oracle/oradata/T1C1/undotbs01.dbf
/u01/app/oracle/oradata/T1C1/users01.dbf

10 rows selected.

SQL> SELECT table_name
  2  FROM   dba_tables
  3  WHERE  owner='OSM$REPAPI';

TABLE_NAME
------------------------------
RANDOM_DOMAIN
RANDOM_FIRST
RANDOM_MIDDLE
RANDOM_LAST
POLICY_TYPES
DISCOUNT_TYPES
CUSTOMERS
POLICIES
OSM_DOCUMENTS

9 rows selected.

SQL> SELECT count(*) FROM osm$repapi.policies;

  COUNT(*)
----------
    921740

SQL>
```

Phase III: Migrate to Automatic Storage Management

The migration from file system to Automatic Storage Management (ASM) allows the upgraded database to take advantage of highly availability features delivered by ASM, necessary for supporting 24/7 systems. After the migration is successfully completed, the database continues to function as it was before the migration. However; ASM enables the Database Administrator (DBA) to manage the database storage more effectively by performing online operations that may require database shutdown on non-ASM environment. An example of an online operation delivered by ASM is adding and removing disks storage while the database is operational, avoiding user interruption by keeping the database open during the operation execution.

Before migration from file system to ASM, perform the following configurations:

- Create, configure and provision ASM disks
- Create and configure ASM disk groups

Figure 7-x depicts ASM architecture, and migrating database data file from file system to ASM using RMAN. The high-level component of ASM is the disk group, which logically groups ASM disks – the physical disk partitions provisioned for ASM. A single disk group may be referenced by one or more database.

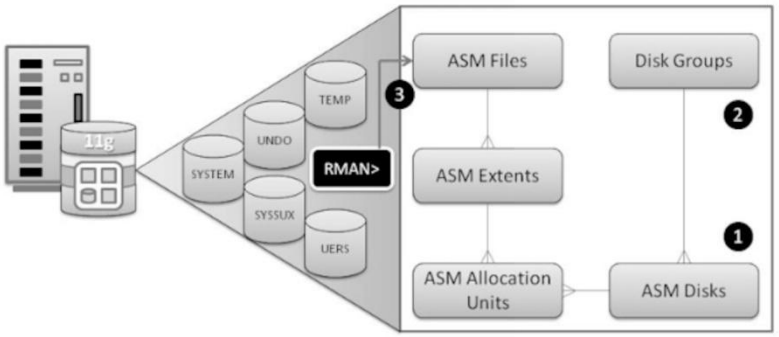


Figure 7-ASM: Automatic Storage Management Architecture

Oracle ASM disks, these are the storage devices associated with an Oracle ASM disk group. There are several forms of an ASM disk: An entire physical disk, combination of one or more disk partitions, a Logical Unit Number (LUN), a logical volume (LV), or network file system. Each ASM disk composed of ASM allocation units, the smallest contiguous amount of disk space that ASM allocates. When you create an ASM disk group, the ASM allocation unit of 1, 2, 4, 8, 16, 32, or 64 MB is specified as disk group attribute, the default is 1MB. ASM allocation units forms an ASM extent associated with an ASM file. An Oracle ASM extent is the raw storage, holds the contents of an Oracle ASM file. An Oracle ASM file consists of one or more extents tracked using an extent map table.

Installation of Oracle 12c Grid Infrastructure

Before the migration to ASM, it's required to install Oracle 12c Grid Infrastructure and configure disk groups. The installation requires using a dedicated operating system user, different from Oracle Database 12c. Figure 7-xx show the installation options.

The Oracle Universal Installer for Oracle 12c Grid Infrastructure has four installation options:

- Install and Configure Oracle Grid Infrastructure for a Cluster
- Install and Configure Oracle Grid Infrastructure for a Standalone Server
- Upgrade Oracle Grid Infrastructure or Oracle Automatic Storage Management

- Install Oracle Grid Infrastructure Software Only

Because the ASM disks are not provisioned yet, select the option 'Install Oracle Grid Infrastructure Software Only'. After the ASM disk are provisioned, then use Oracle Automatic Storage Management Configuration Assistant (ASMCA) to create, start the ASM instance +ASM and create and configure the disk groups +DATA and +FRA.

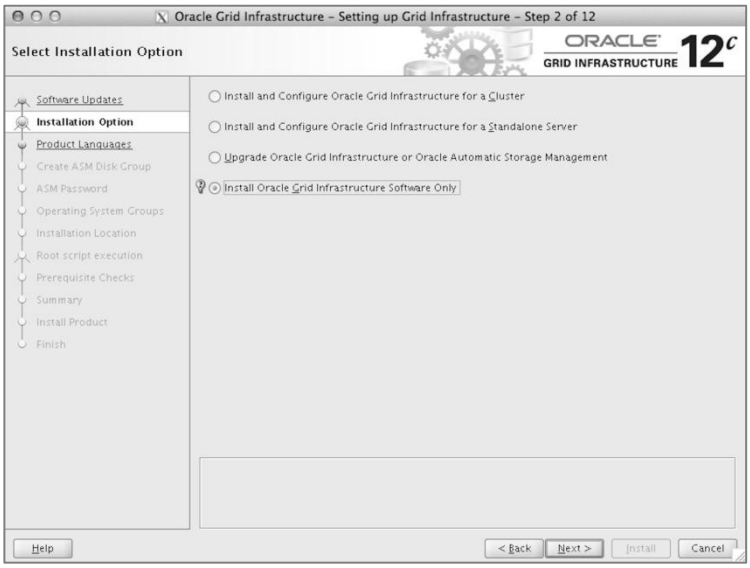


Figure 7-x: Oracle Grid Infrastructure 12c

For details regarding creating and provisioning ASM disks, refer to the section Provisioning ASM Disks.

Working with ASM Instance

An ASM instance is installed on Oracle Database 12c server. There must be one and only one ASM instance running on the system. The ASM instance must be started in order for the database instance to access data files. Connect as SYSASM to start the ASM instance, which mount the list of created and configured disk groups.

```
$ . oraenv
ORACLE_SID = [+ASM] ? +ASM
The Oracle base remains unchanged with value /u02/app/grid
$ sqlplus /nolog

SQL*Plus: Release 12.1.0.1.0 Production on Wed Apr 30 21:40:21 2014

Copyright (c) 1982, 2013, Oracle.  All rights reserved.
```

```
SQL> CONN / AS SYSASM
Connected to an idle instance.
SQL> STARTUP
ASM instance started

Total System Global Area   1135747072 bytes
Fixed Size                    2297344 bytes
Variable Size              1108283904 bytes
ASM Cache                    25165824 bytes
ASM diskgroups mounted
SQL>
```

The ASM instance uses fewer parameters than the database instance, the only mandatory parameter-value is INSTANCE_TYPE=ASM. View the ASM instance parameters related to list the auto-mounted disk groups and ASM disks storage devices.

```
SQL> SHOW PARAMETERS disk

NAME                                 TYPE           VALUE
-------------------------------   -----------    ----------------------
asm_diskgroups                       string         DATA, FRA
asm_diskstring                       string         /dev/x*
SQL>
```

This instance works in conjunction with the database instance to handle database files management. When the database instance submits the following CREATE TABLESPACE statement:

```
SQL> CREATE TABLESPACE edge
  2  DATAFILE '+DATA' SIZE 50M AUTOEXTEND on;

Tablespace created.

SQL> SELECT file_name
  2  FROM   dba_data_files
  3  WHERE  tablespace_name='EDGE';

FILE_NAME
-----------------------------------------------------------------
+DATA/t1c1/datafile/edge.267.846273245

SQL>
```

The tablespace data file is created as an ASM file as follows:-

1. The database requests creating the data file on the specified disk group, +DATA.
2. The ASM instance allocates the storage based on the Allocation Unit (AU) of the disk group. ASM spreads the allocation units evenly across all ASM disks within the disk group specified to balance load and avoid hot spots. It also mirrors allocation units for the disk group with redundancy normal or high attribute.

3. The database background process ASMB receives the extent map for the new file. The extent map is a table that maps data extents in a file to allocation units on disk. This allows the database instance access to the data.
4. The file is now open and the database process starts formatting the file.
5. The ASM instance confirms the file creation and mark the file as created.
6. The file is close and reopened by the database instance.

Database Disk Groups

There are two disk groups created; +DATA and +FRA. Creating disk groups requires ASM disks. The +DATA disk group hosts all types of database data files and +FRA disk group dedicated for the Fast Recovery Area (FRA) files only.

Figure 7-x is Automatic Storage Management Configuration Assistant (ASMCA), it manages disk groups. Use ASMCA to create/alter/drop disk groups, add disks to a disk group, drop disks from a disk group, mount or dismount disk groups.

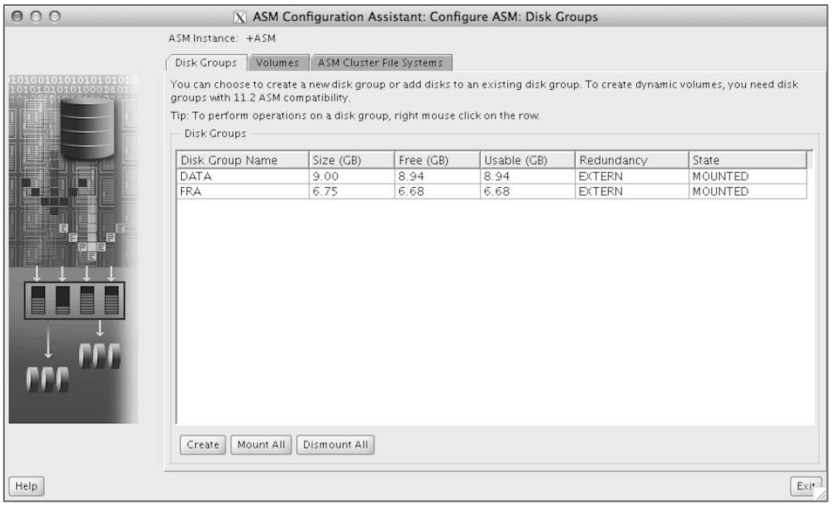


Figure 7-xx: Automatic Storage Management Configuration Assistant (ASMCA)

ASM instance has its own command-line – ASMCMD. ASMCMD adapts Linux-like file system management commands. The following illustrates the copy of the archived log file from ASM to file system using the copy command. The '+' character designates the ASM root directory, use the change directory 'cd' command to navigate across the ASM directory structure.

```
$ . oraenv
ORACLE_SID = [grid] ? +ASM
The Oracle base has been set to /u02/app/grid
```

```
$ asmcmd
ASMCMD> ls
DATA/
FRA/
ASMCMD> cd +FRA/T1C1/ARCHIVELOG/2014_05_02
ASMCMD> ls
thread_1_seq_24.257.846500305
ASMCMD> cp * /home/grid/arc
copying +FRA/T1C1/ARCHIVELOG/2014_05_02/thread_1_seq_24.257.846500305 ->
/home/grid/arc/thread_1_seq_24.257.846500305
ASMCMD>
```

Testing the creation of tablespace in +DATA disk group. The next CREATE TABLESPACE command creates the tablespace EDGE. The data file is auto extended with an initial size of 100MB.

```
$ . oraenv
ORACLE_SID = [+ASM] ? T1C2
The Oracle base has been changed from /u02/app/grid to /u01/app/oracle
$ sqlplus /nolog

SQL*Plus: Release 11.2.0.3.0 Production on Tue Apr 22 18:34:38 2014

Copyright (c) 1982, 2011, Oracle.  All rights reserved.

SQL> CONN / AS SYSDBA
Connected.
SQL> CREATE TABLESPACE edge
  2  DATAFILE '+DATA';

Tablespace created.

SQL> SELECT tablespace_name
  2  FROM   dba_tablespaces;

TABLESPACE_NAME
------------------------------
SYSTEM
SYSAUX
UNDOTBS1
TEMP
USERS
EDGE

6 rows selected.

SQL> DROP TABLESPACE edge;

Tablespace dropped.

SQL>
```

The above commands was done on a test database – T1C2, it's necessary to create a test database of type Automatic Storage Management (ASM). The test confirms Oracle Grid Infrastructure 12c is properly installed, configured and disk groups are

configured for the migration. View the +ASM instance alert file, investigate and resolve errors found before proceeding to Migration to Automatic Storage Management. Use DBCA to drop the test database, this will free disk groups space.

The next section to perform the migration to Automatic Storage Management (ASM).

Migration to Automatic Storage Management

The migration of clone database storage from file system to Automatic Storage Management (ASM) executed using either the Oracle Enterprise Manage (OEM) 'Migrate Database to ASM' wizard or the Oracle Recovery Manager (RMAN) DUPLICATE command. Very Large Database (VLDB) environments benefit using RMAN for migrating the database gradually. For example, processing the migration based on the disk storage availability and the ownership of tablespaces by applications, this technique allows faster and manageable restoration in case of failure. The RMAN DUPLICATE command supports database clone from an active database, this option minimizes the amount of trails files to synchronize the target database after successfully upgrading the database to Oracle Database 12c. Figure 7-xx presents the migration workflow using RMAN DUPLICATE command, which requires preparation of the disk groups +DATA and +FRA for the database files storage.

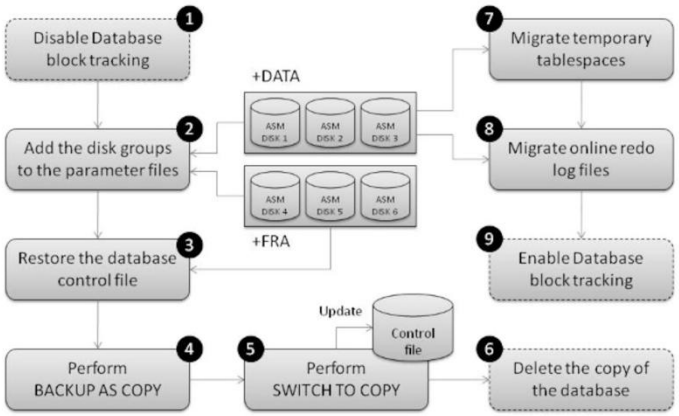


Figure 7-xx: Migration workflow using RMAN.

The following steps are executed on the clone database using SQL*Plus and Oracle Recovery Manager (RMAN).

1. Disable the database block change tracking process
2. Modify the parameter file by adding the ASM disk groups

3. Migrate the control file
4. Migrate the data files
5. Migrate the temporary tablespaces
6. Migrate the online redo log files groups
7. Enable the database block change tracking process

Disable Block Change Tracking

When the block change tracking is enabled, for each database block that is changed, the background process CTWR writes the physical address to an external files – outside of the database. The purpose is to support RMAN incremental backups. During the migration to ASM, if the database block change tracking process is enabled, then it must be disabled prior to starting the migration steps. To determine if the process is enable or disabled, query the database dynamic view V$BLOCK_CHANGE_TRACKING. If the status column returns 'ENABLED', then disable it, otherwise proceed to the next step.

```
SQL> SELECT status, filename FROM v$block_change_tracking;

STATUS     FILENAME
---------- ------------------------------------------------------------
ENABLED    /u01/app/oracle/…/T1C1/changetracking/o1_mf_9ozlbm2w_.chg

SQL> ALTER DATABASE DISABLE BLOCK CHANGE TRACKING;

Database altered.

SQL>
```

Define Disk Groups

The migration to Automatic Storage Management (ASM) requires the ASM disk groups are created and mounted. This migration uses two disk groups, +DATA and +FRA. The disk group +DATA stores the database data files, redo log files and control files. The disk group +FRA is the location of the database Fast Recovery Area (FRA), the default location for RMAN backup files, flashback log files and the mirror copy of the control file and online redo log files.

- Determine the name and status of +DATA and +FRA disk groups by querying the ASM dynamic view V$ASM_DISKGROUP.

```
SQL> SELECT name, state
  2  FROM   v$asm_diskgroup;

NAME                           STATE
------------------------------ -----------
FRA                            MOUNTED
```

```
DATA                            MOUNTED

SQL>
```

- Create the parameter file 'initT1C1.ora' from the server parameter file 'spfileT1C1.ora'. Use IMMEDIATE shutdown option to force checkpoint and closes the database files.

```
SQL> SHUTDOWN IMMEDIATE
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> CONN / AS SYSDBA
Connected to an idle instance.
SQL> CREATE pfile FROM spfile;

File created.

SQL>
```

- Add and/or modify the parameters file by adding the disk groups +DATA and +FRA as the locations of the control files, data files, and online redo log files. Table 7-2 lists the modified parameters.

Parameter	Description
CONTROL_FILES	The location of Oracle managed control files.
DB_CREATE_ONLINE_LOG_DEST_1	The location of the first member of the online redo log files, and control file.
DB_CREATE_FILE_DEST	The location of the data file
DB_RECOVERY_FILE_DEST	The location of the Fast Recovery Area (FRA).

Table 7-2: Automatic Storage Management files locations

```
*.audit_file_dest='/u01/app/oracle/admin/T1C1/adump'
*.audit_trail='db'
*.compatible='11.2.0.0.0'
*.control_files=(+DATA, +FRA)
*.db_create_online_log_dest_1=+DATA
*.db_create_file_dest=+DATA
*.db_recovery_file_dest=+FRA
*.db_recovery_file_dest_size=4322230272
*.db_block_size=8192
*.db_domain='precisetrace.com'
*.db_name='T1C1'
*.db_recovery_file_dest_size=4322230272
*.diagnostic_dest='/u01/app/oracle'
*.dispatchers='(PROTOCOL=TCP) (SERVICE=T1C1XDB)'
*.log_archive_format='%t_%s_%r.dbf'
*.memory_target=2147483648
*.open_cursors=300
*.processes=150
```

```
*.remote_login_passwordfile='EXCLUSIVE'
*.undo_tablespace='UNDOTBS1'
```

- Create the server parameter file spfileT1C1.ora from the modified parameter file initT1C1.ora. The two parameters files are currently stored on the default location - $ORACLE_HOME/dbs. After the migration to ASM is completed, use the Oracle supplied PL/SQL package DBMS_FILE_TRANSFER to move the server parameter file to ASM storage.

```
SQL> CREATE spfile FROM pfile;

File created.

SQL>
```

Control File Migration

The migration begins by restoring an image copy of the current control file to the location(s) specified by the parameter CONTROL_FILES. Start the Oracle Recovery Manager (RMAN), connect to the target database, start the instance in NOMOUNT mode and perform the RMAN RESTORE command of the control file.

- Start Oracle Recovery Manager (RMAN) and start the instance in NOMOUNT

```
$ rman

Recovery Manager: Release 11.2.0.3.0 - Production on Tue Apr 22 19:35:20
2014

Copyright (c) 1982, 2011, Oracle and/or its affiliates.  All rights
reserved.

RMAN> CONNECT TARGET

connected to target database (not started)

RMAN> STARTUP NOMOUNT

Oracle instance started

Total System Global Area     551165952 bytes

Fixed Size                     2230232 bytes
Variable Size                331352104 bytes
Database Buffers             213909504 bytes
Redo Buffers                   3674112 bytes

RMAN>
```

- Based on the current location of the control file identified from initT1C1.ora, use Oracle Recovery Manager (RMAN) RESTORE CONTROLFILE command to restore a copy of the control file to +DATA and +FRA.

```
RMAN> RESTORE CONTROLFILE FROM
2> '/u01/app/oracle/oradata/T1C1/control01.ctl';

Starting restore at 22-APR-14
using channel ORA_DISK_1

channel ORA_DISK_1: copied control file copy
output file name=+DATA/t1c1/controlfile/current.256.845581127
output file name=+FRA/t1c1/controlfile/current.257.845581127
Finished restore at 22-APR-14

RMAN>
```

Data File Migration

After the control is successfully restored, this next step is to migrates the database permanent and undo data files. One method to achieve this is by creating a backup copy of the database, which is a consistent backup of the database data files. The file location is explicitly identified by the FORMAT parameter as +DATA disk group. Using the RMAN COPY command creates an image copy of the data files to the location specified by FORMAT parameter.

- Optionally, enable Recovery Manager (RMAN) parallelism. This enhances the data files copy operations by allocating multiple channels. A parallelism of 6 allocates six RMAN database channels to create the database image copy.

```
RMAN> STARTUP MOUNT

database is already started
database mounted
released channel: ORA_DISK_1

RMAN> CONFIGURE DEVICE TYPE DISK PARALLELISM 6;

new RMAN configuration parameters:
CONFIGURE DEVICE TYPE DISK PARALLELISM 6 BACKUP TYPE TO BACKUPSET;
new RMAN configuration parameters are successfully stored
```

- Create an image copy of the database data files to +DATA disk group. The database is in MOUNT state to create a consistent image backup files. The COPY command excludes the file belonging to the database temporary tablespace, which will migrated separately.

```
RMAN> BACKUP AS COPY DATABASE FORMAT '+DATA';
```

```
Starting backup at 22-APR-14
allocated channel: ORA_DISK_1
channel ORA_DISK_1: SID=26 device type=DISK
allocated channel: ORA_DISK_2
channel ORA_DISK_2: SID=27 device type=DISK
allocated channel: ORA_DISK_3
channel ORA_DISK_3: SID=28 device type=DISK
allocated channel: ORA_DISK_4
channel ORA_DISK_4: SID=29 device type=DISK
allocated channel: ORA_DISK_5
channel ORA_DISK_5: SID=30 device type=DISK
allocated channel: ORA_DISK_6
channel ORA_DISK_6: SID=31 device type=DISK
channel ORA_DISK_1: starting datafile copy
input datafile file number=00002
name=/u01/app/oracle/oradata/T1C1/sysaux01.dbf
channel ORA_DISK_2: starting datafile copy
input datafile file number=00001
name=/u01/app/oracle/oradata/T1C1/system01.dbf
channel ORA_DISK_3: starting datafile copy
input datafile file number=00004
name=/u01/app/oracle/oradata/T1C1/users01.dbf
channel ORA_DISK_4: starting datafile copy
input datafile file number=00003
name=/u01/app/oracle/oradata/T1C1/undotbs01.dbf
output file name=+DATA/t1c1/datafile/users.259.845581373
tag=TAG20140422T194250 RECID=7 STAMP=845581412
channel ORA_DISK_3: datafile copy complete, elapsed time: 00:00:51
output file name=+DATA/t1c1/datafile/undotbs1.260.845581377
tag=TAG20140422T194250 RECID=8 STAMP=845581424
channel ORA_DISK_4: datafile copy complete, elapsed time: 00:00:57
output file name=+DATA/t1c1/datafile/system.258.845581371
tag=TAG20140422T194250 RECID=9 STAMP=845581560
channel ORA_DISK_2: datafile copy complete, elapsed time: 00:03:12
output file name=+DATA/t1c1/datafile/sysaux.257.845581371
tag=TAG20140422T194250 RECID=10 STAMP=845581564
channel ORA_DISK_1: datafile copy complete, elapsed time: 00:03:24
Finished backup at 22-APR-14

Starting Control File and SPFILE Autobackup at 22-APR-14
piece handle=+FRA/t1c1/autobackup/2014_04_22/s_845580320.258.845581577
comment=NONE
Finished Control File and SPFILE Autobackup at 22-APR-14

RMAN>
```

- Now, the control files and data files are moved to Automatic Storage Management (ASM), the next step is to update the control files with the new locations of the database data files, then open the database on normal mode. RMAN SWITCH command updates the file names for a database, tablespace, or data file to the latest image copies available for the specified files or database.

```
RMAN> SWITCH DATABASE TO COPY;

using target database control file instead of recovery catalog
```

```
datafile 1 switched to datafile copy
"+DATA/t1c1/datafile/system.258.845581371"
datafile 2 switched to datafile copy
"+DATA/t1c1/datafile/sysaux.257.845581371"
datafile 3 switched to datafile copy
"+DATA/t1c1/datafile/undotbs1.260.845581377"
datafile 4 switched to datafile copy
"+DATA/t1c1/datafile/users.259.845581373"

RMAN> ALTER DATABASE OPEN;

database opened

RMAN> exit

Recovery Manager complete.
$
```

The database migration to Automatic Storage Management (ASM) is partially completed and the database is open on read/write mode. The next steps manually migrate the remaining of the database – temporary files and online redo log files. Migration of temporary and online redo log files is an in-place operation, the database remains open while executing migration commands.

Temporary Files Migration

The Recovery Manager (RMAN) command BACKUP AS COPY DATABASE excludes the temporary tablespace files. It's performed by manually creating an interim temporary tablespace to allow the recreation of the existing temporary tablespace as Automatic Storage Management (ASM) type.

- Determine the name and size of the temporary files. Because the database is open, query the database data dictionary view DBA_TEMP_FILES.

```
SQL> CONN / AS SYSDBA
Connected.
SQL> SELECT file_name, bytes/1024/1204 MB
  2  FROM   dba_temp_files;

FILE_NAME                                                MB
-------------------------------------------------- -----------
/u01/app/oracle/oradata/T1C1/temp01.dbf                549.42

SQL>
```

- Create an interim temporary tablespace, then alter the default database-wide temporary tablespace. Query the data dictionary DATABASE_PROPERTIES view to ascertain the default database-wide temporary tablespace.

```
SQL> CREATE TEMPORARY TABLESPACE temp_1
  2  TEMPFILE '/u01/app/oracle/oradata/T1C1/temp02.dbf' SIZE 550M;
```

```
Tablespace created.

SQL> ALTER DATABASE DEFAULT TEMPORARY TABLESPACE temp_1;

Database altered.
```

- Drop and recreate the original temporary tablespace. The new temporary tablespace is created as Automatic Storage Management (ASM) file type. Make the temporary tablespace size matches the original size.

```
SQL> DROP TABLESPACE temp INCLUDING CONTENTS;

Tablespace dropped.

SQL> CREATE TEMPORARY TABLESPACE temp
  2  TEMPFILE '+DATA' SIZE 550M;

Tablespace created.

SQL> ALTER DATABASE DEFAULT TEMPORARY TABLESPACE temp;

Database altered.
```

- Drop the interim temporary tablespace and verify the temporary tablespace is created as Automatic Storage Management (ASM) file type.

```
SQL> DROP TABLESPACE temp_1 INCLUDING CONTENTS;

Tablespace dropped.

SQL> SELECT file_name
  2  FROM   dba_temp_file;

FILE_NAME
-------------------------------------------------------------------------
+DATA/t1c1/tempfile/temp.261.845582177

SQL>
```

Online Redo Log Files Migration

Because Oracle Recovery Manager (RMAN) does not backup online redo log files, online redo log files migration from file system to Automatic Storage Management (ASM) type is performed manually. As illustrated below, it's an in-place operation and does not require shutdown of the database.

- List of the current online redo groups members by a join query of V$LOGFILE and V$LOG.

```
SQL> SELECT l.group#, lf.member, l.bytes/1024/1024
```

```
  2  FROM   v$logfile lf, v$log l
  3  WHERE  lf.group#=l.group#
  4  ORDER BY 1;

    GROUP# MEMBER                                    L.BYTES/1024/1024
---------- ----------------------------------------  -----------------
         1 /u01/app/oracle/oradata/T1C1/redo01.log                  50
         2 /u01/app/oracle/oradata/T1C1/redo02.log                  50
         3 /u01/app/oracle/oradata/T1C1/redo03.log                  50

SQL>
```

- Create new online redo log groups, each group composed of two members. The first member is stored on +DATA disk group and the second member on +FRA disk group. According to the above query, the next three groups to be created are 4, 5 and 6, use the same returned size.

```
SQL> ALTER DATABASE ADD LOGFILE GROUP 4 ('+DATA', '+FRA') SIZE 50M;

Database altered.

SQL> ALTER DATABASE ADD LOGFILE GROUP 5 ('+DATA', '+FRA') SIZE 50M;

Database altered.

SQL> ALTER DATABASE ADD LOGFILE GROUP 6 ('+DATA', '+FRA') SIZE 50M;

Database altered.
```

- Drop the non-ASM online redo log groups. Start by dropping INACTIVE and UNUSED groups only. The query below shows group 1 status is CURRENT and cannot be dropped until several log switches occur, archived and it becomes INACTIVE.

```
SQL> SELECT group#, status FROM v$log;

    GROUP# STATUS
---------- ----------------
  1 CURRENT
  2 INACTIVE
  3 INACTIVE
  4 UNUSED
  5 UNUSED
  6 UNUSED

6 rows selected.

SQL> ALTER DATABASE DROP LOGFILE GROUP 2;

Database altered.

SQL> ALTER DATABASE DROP LOGFILE GROUP 3;

Database altered.
```

- To drop an active group, perform multiple log switches until the group status becomes INACTIVE.

```
SQL> SELECT group#, status FROM v$log;

    GROUP# STATUS
---------- ----------------
  1 CURRENT
  4 UNUSED
  5 UNUSED
  6 UNUSED

SQL> ALTER SYSTEM SWITCH LOGFILE;

System altered.

SQL> select GROUP#, STATUS from v$log;

    GROUP# STATUS
---------- ----------------
  1 INACTIVE
  4 CURRENT
  5 UNUSED
  6 UNUSED

SQL> ALTER DATABASE DROP LOGFILE GROUP 1;

Database altered.
```

- Verify the online redo log file migrated to Automatic Storage Management (ASM). As per the query output below, there are two member for each group stored on +DATA and +FRA disk groups. Multiplexing the online redo log groups avoids single point of failure.

```
SQL> SELECT member
  2  FROM   v$logfile;

MEMBER
--------------------------------------------------------------------------------
+DATA/t1c1/onlinelog/group_4.262.845583489
+FRA/t1c1/onlinelog/group_4.262.845583491
+DATA/t1c1/onlinelog/group_5.263.845583507
+FRA/t1c1/onlinelog/group_5.263.845583509
+DATA/t1c1/onlinelog/group_6.264.845583523
+FRA/t1c1/onlinelog/group_6.264.845583525

6 rows selected.

SQL>
```

This ends the migration of the database from file system to Automatic Storage Management (ASM). The ASM instance must remain operational to provide support for the database instance access of database files.

Verify New Database Storage

Verify the database storage has been migrated to Automatic Storage Management (ASM) files type. Use SQL*Plus or ASMCMD command-line to confirm the database files are successfully migrated to ASM.

```
SQL> SELECT file_name FROM dba_data_files UNION
  2  SELECT file_name FROM dba_temp_files UNION
  3  SELECT member    FROM v$logfile      UNION
  4  SELECT name      FROM v$controlfile;

FILE_NAME
--------------------------------------------------------------------------------
+DATA/t1c1/controlfile/current.256.845581127
+DATA/t1c1/datafile/sysaux.257.845581371
+DATA/t1c1/datafile/system.258.845581371
+DATA/t1c1/datafile/undotbs1.260.845581377
+DATA/t1c1/datafile/users.259.845581373
+DATA/t1c1/onlinelog/group_4.262.845583489
+DATA/t1c1/onlinelog/group_5.263.845583507
+DATA/t1c1/onlinelog/group_6.264.845583523
+DATA/t1c1/tempfile/temp.261.845582177
+FRA/t1c1/controlfile/current.257.845581127
+FRA/t1c1/onlinelog/group_4.262.845583491

FILE_NAME
--------------------------------------------------------------------------------
+FRA/t1c1/onlinelog/group_5.263.845583509
+FRA/t1c1/onlinelog/group_6.264.845583525

13 rows selected.

SQL>
```

DBMS_FILE_TRANSFER Package

Another approach to migrate and to copy files from file system to ASM is to use the Oracle supplied PL/SQL package DBMS_FILE_TRANSFER. The example below illustrates the use of the package to migrate a temporary tablespace from file system to Automatic Storage Management (ASM) using the following steps:

- Create and provide access to the source and target directories – SRCDIR and TGTDIR. The source data file is located on the directory /home/oracle. The target directory is +DATA disk group.

  ```
  SQL> CREATE DIRECTORY srcdir as '/home/oracle';

  Directory created.

  SQL> GRANT all ON DIRECTORY srcdir to system;
  ```

```
Grant succeeded.

SQL> CREATE DIRECTORY tgtdir as '+DATA';

Directory created.

SQL> GRANT all ON DIRECTORY tgtdir to system;

Grant succeeded.

SQL>
```

- Execute DBMS_FILE_TRANSFER.COPY_FILE procedure

```
SQL> BEGIN
  2     DBMS_FILE_TRANSFER.COPY_FILE(
  3          'SRCDIR', 'temp_1.dbf', 'TGTDIR', 'temp_1.f');
  4  END;
  5  /

PL/SQL procedure successfully completed.

SQL>
```

- Update the database data dictionary with the new file location. The tablespace must be either offline or the database is on MOUNT mode. Obtain the name of the copied file by querying V$ASM_FILES.

```
SQL> SHUTDOWN IMMEDIATE
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> STARTUP MOUNT
ORACLE instance started.

Total System Global Area       551165952 bytes
Fixed Size                       2230232 bytes
Variable Size                  352323624 bytes
Database Buffers               192937984 bytes
Redo Buffers                     3674112 bytes
Database mounted.
SQL> ALTER DATABASE RENAME FILE '/home/oracle/temp_1.dbf'
  2  TO '+DATA/temp_1.f';

Database altered.

SQL> ALTER DATABASE OPEN;

Database altered.

SQL>
```

- Verify the file new location of the migrated temporary file.

```
SQL> SELECT tablespace_name, file_name
```

```
  2  FROM   dba_temp_files;

TABLESPACE_NAME      FILE_NAME
-------------------- --------------------------------------------------
TEMP_2               +DATA/t1c1/tempfile/copy_file.267.846281147
TEMP_1               +DATA/temp_1.f
TEMP                 +DATA/t1c1/tempfile/temp.263.846267555

SQL>
```

The next two steps are optional, but highly recommended to account for the disk space used by unneeded database files.

Delete Database Files

After migrating the database to Automatic Storage Management (ASM), it's recommended to remove all previous backups and data file copies. Removing file must be executed using Oracle Recovery Manager (RMAN) and not manually. RMAN removes the files from the FRA, update the control file and data dictionary.

```
$ rman

Recovery Manager: Release 11.2.0.3.0 - Production on Tue Apr 22 20:30:41
2014

Copyright (c) 1982, 2011, Oracle and/or its affiliates.  All rights
reserved.

RMAN> CONNECT TARGET

connected to target database: T1C1 (DBID=3821873595)

RMAN> RUN {
2> DELETE COPY OF DATABASE;
3> }

using target database control file instead of recovery catalog
allocated channel: ORA_DISK_1
channel ORA_DISK_1: SID=42 device type=DISK
allocated channel: ORA_DISK_2
channel ORA_DISK_2: SID=47 device type=DISK
allocated channel: ORA_DISK_3
channel ORA_DISK_3: SID=45 device type=DISK
allocated channel: ORA_DISK_4
channel ORA_DISK_4: SID=43 device type=DISK
allocated channel: ORA_DISK_5
channel ORA_DISK_5: SID=44 device type=DISK
allocated channel: ORA_DISK_6
channel ORA_DISK_6: SID=49 device type=DISK
List of Datafile Copies
=======================

Key     File S Completion Time Ckp SCN    Ckp Time
------- ---- - --------------- ---------- ---------------
11      1    A 22-APR-14       5993451    22-APR-14
```

```
        Name: /u01/app/oracle/oradata/T1C1/system01.dbf

12      2    A 22-APR-14       5993451    22-APR-14
        Name: /u01/app/oracle/oradata/T1C1/sysaux01.dbf

13      3    A 22-APR-14       5993451    22-APR-14
        Name: /u01/app/oracle/oradata/T1C1/undotbs01.dbf

14      4    A 22-APR-14       5993451    22-APR-14
        Name: /u01/app/oracle/oradata/T1C1/users01.dbf

Do you really want to delete the above objects (enter YES or NO)? YES
deleted datafile copy
datafile copy file name=/u01/app/oracle/oradata/T1C1/system01.dbf RECID=11
STAMP=845581916
deleted datafile copy
datafile copy file name=/u01/app/oracle/oradata/T1C1/sysaux01.dbf RECID=12
STAMP=845581916
deleted datafile copy
datafile copy file name=/u01/app/oracle/oradata/T1C1/undotbs01.dbf RECID=13
STAMP=845581916
deleted datafile copy
datafile copy file name=/u01/app/oracle/oradata/T1C1/users01.dbf RECID=14
STAMP=845581916
Deleted 4 objects

RMAN> exit

Recovery Manager complete.
```

Removal of Unneeded Files

After all database data files, online redo log files, and control files are migrated to ASM, It's now time to remove the old files from file system. Be careful when executing this step, make sure your current directory is /u01/app/oracle/oradata/T1C1.

```
$ cd /u01/app/oracle/oradata/T1C1/
$ ls -al
total 167092
drwxr-xr-x 2 oracle oinstall      4096 Apr 22 20:31 .
drwxr-x--- 3 oracle oinstall      4096 Apr 18 10:04 ..
-rw-r----- 1 oracle oinstall  10076160 Apr 22 19:25 control01.ctl
-rw-r----- 1 oracle oinstall  52429312 Apr 22 20:20 redo01.log
-rw-r----- 1 oracle oinstall  52429312 Apr 22 20:12 redo02.log
-rw-r----- 1 oracle oinstall  52429312 Apr 22 20:12 redo03.log
-rw-r----- 1 oracle oinstall  49291264 Apr 22 19:21 temp01.dbf
-rw-r----- 1 oracle oinstall 209723392 Apr 22 19:55 temp02.dbf
$ rm *.log *.dbf *.ctl
$
```

Enable Block Change Tracking

This step performed for testing purpose only. The block change tracking process should remain disabled during the next phase – the database upgrade.

```
$ sqlplus /nolog

SQL*Plus: Release 11.2.0.3.0 Production on Tue Apr 22 20:37:31 2014

Copyright (c) 1982, 2011, Oracle.  All rights reserved.

SQL> CONN / AS SYSDBA
Connected.

SQL> ALTER DATABASE ENABLE BLOCK CHANGE TRACKING;

Database altered.

SQL> ALTER DATABASE DISABLE BLOCK CHANGE TRACKING;

Database altered.

SQL>
```

Whole Database Backup

The database has been migrated from file system to Automatic Storage Management (ASM), before starting the next Phase IV - Upgrade to Oracle database 12c, use Oracle Recovery Manager (RMAN) to perform a whole database backup. This backup is available in the event of failure during the database upgrade.

```
RMAN> CONFIGURE CONTROLFILE AUTOBACKUP on;

new RMAN configuration parameters:
CONFIGURE CONTROLFILE AUTOBACKUP ON;
new RMAN configuration parameters are successfully stored

RMAN> BACKUP DATABASE PLUS ARCHIVELOG TAG="Before_Upgrade";
```

Up on successful backup, the next step is to upgrade the clone database from Oracle Database 11g to Oracle Database 12c, as illustrated on Phase IV.

Phase IV: Upgrade to Oracle Database 12c

There are several methods to upgrade the clone database to Oracle Database 12c. Our aim is to achieve near zero-downtime database upgrade, keeping the source database up and running throughout the upgrade process. The database upgrade is performed using Oracle Database Upgrade Assistant (DBUA) or manually using SQL upgrade

commands. Oracle Corporation recommends using DBUA Graphical User Interface (GUI) as illustrated in this section. Figure 7-xx shows the clone database upgrade.

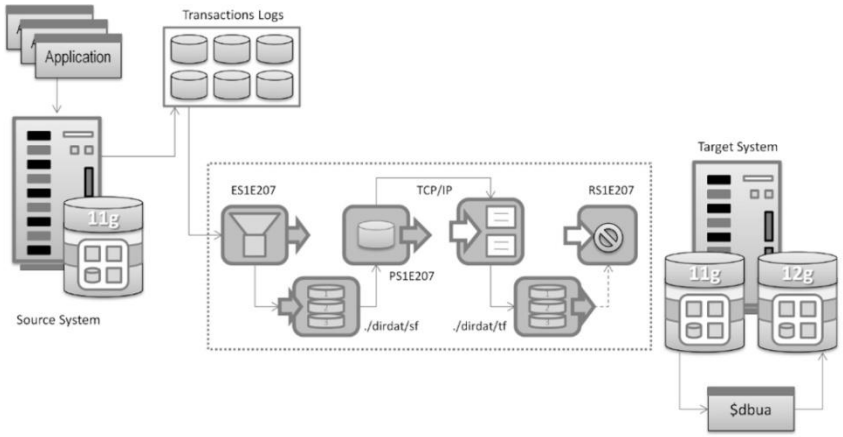


Figure 7-xx: Clone Database Upgrade using DBUA

Let's start by discussing the upgrade checklist, this ensures the database upgrade process is successfully executed without issues.

Preparation Checklist

- Direct upgrade path certification
- Delete the database recycle bin
- Check for INVALID and duplicate object in SYS and SYSTEM schemas
- Remove outdated and obsolete parameters
- Check for INVALID components
- Check for mandatory components
- Remove obsolete components
- Apply patch sets, Critical Patch Updates (CPUs) and patches
- Minimum disk space requirements

Direct Upgrade Certification Path

It's crucial to ascertain direct database upgrade is certified with Oracle Database 12c Release 1. For specific database releases, it may be required to perform an interim database upgrade to reach a certified source version for direct upgrade to Oracle Database 12c version 12.1.0.1.0. As of the time of writing the book, the current direct upgrade path is shown in table 7-1.

Source Database Release	Source Version	Direct Upgrade Support

Oracle Database 11g Release 2	11.2.0.2 and later	Yes
Oracle Database 11g Release 1	11.1.0.7	Yes
Oracle Database 10g Release 2	10.2.0.5	Yes

Table 7-1: Direct Upgrade Certification Path

Visit http://www.oracle-ggs.com for the latest database upgrade certification path to Oracle Database 12c.

Delete Database Recyclebin

The recycle bin must be purged before starting the Database upgrade. Use the command below to empty the database recycle bin for all users on the source database.

```
SQL> PURGE DBA_RECYCLEBIN;

DBA Recyclebin purged.

SQL>
```

Invalid Objects

Find out if there are invalid objects owned by SYS and SYSTEM. Find and fix invalid objects as illustrated below.

```
SQL> SELECT DISTINCT object_name, object_type, owner
  2  FROM   dba_objects
  3  WHERE  status='INVALID'
  4  ORDER BY owner, object_name, object_type;

no rows selected

SQL>
```

Do not start the database upgrade when there are invalid data dictionary objects. To recompile invalid objects, connect as SYSDBA and run the supplied script utlrp.sql as shown next.

```
SQL> @?/rdbms/admin/utlrp.sql
```

Outdated parameters

When Oracle releases a new database release, a parameter may become obsolete or deprecated. Obsolete parameters, underscore parameters and events must be removed prior to starting database upgrade.

Obsolete parameters on Oracle Database 12c Release 1

- _app_ctx_vers
- _log_io_size

For list of depreciated parameters, query the dynamic view V$PARAMETER as show next.

```
SQL> SELECT name
  2  FROM   v$parameter
  3  WHERE  isdeprecated ='TRUE'
  4  ORDER BY name;

NAME
--------------------------------------------------------------------------------
active_instance_count
background_dump_dest
buffer_pool_keep
buffer_pool_recycle
commit_write
cursor_space_for_time
fast_start_io_target
global_context_pool_size
instance_groups
lock_name_space
log_archive_local_first

NAME
--------------------------------------------------------------------------------
log_archive_start
max_enabled_roles
parallel_automatic_tuning
parallel_io_cap_enabled
parallel_server
parallel_server_instances
plsql_debug
plsql_v2_compatibility
remote_os_authent
resource_manager_cpu_allocation
sec_case_sensitive_logon

NAME
--------------------------------------------------------------------------------
serial_reuse
sql_trace
standby_archive_dest
user_dump_dest

26 rows selected.

SQL>
```

Invalid Components

Query DBA_REGISTRY to list the components status. The status for all components must be valid before attempting the upgrade. Also, the DBUA pre-upgrade script reports installed component status.

```
SQL> SELECT comp_name component, version, status
  2  FROM   dba_registry;

COMPONENT                                VERSION        STATUS
---------------------------------------- -------------  ------
OWB                                      11.2.0.3.0     VALID
Oracle Application Express               3.2.1.00.12    VALID
Oracle Enterprise Manager                11.2.0.3.0     VALID
OLAP Catalog                             11.2.0.3.0     VALID
Spatial                                  11.2.0.3.0     VALID
Oracle Multimedia                        11.2.0.3.0     VALID
Oracle XML Database                      11.2.0.3.0     VALID
Oracle Text                              11.2.0.3.0     VALID
Oracle Expression Filter                 11.2.0.3.0     VALID
Oracle Rules Manager                     11.2.0.3.0     VALID
Oracle Workspace Manager                 11.2.0.3.0     VALID
Oracle Database Catalog Views            11.2.0.3.0     VALID
Oracle Database Packages and Types       11.2.0.3.0     VALID
JServer JAVA Virtual Machine             11.2.0.3.0     VALID
Oracle XDK                               11.2.0.3.0     VALID
Oracle Database Java Packages            11.2.0.3.0     VALID
OLAP Analytic Workspace                  11.2.0.3.0     VALID
Oracle OLAP API                          11.2.0.3.0     VALID

18 rows selected.

SQL>
```

Mandatory Components

Oracle Database 12c Release 1 requires few mandatory components. XDB (Oracle XML database) is a mandatory component, it must valid and upgraded successfully.

Obsolete Components

Oracle Database 12c Release 1 has made the following components obsolete:

- Enterprise Manager (EM) Database Control Repository
- UltraSearch

When performing upgrade using DBUA, obsolete components are skipped. However; it's best to remove them from the clone database before starting the upgrade.

Database Patches

This is highly important step and must applied. The database patching either to bring you to a certified source database version, or to apply fixes necessary for successful upgrade. Apply all patch sets, Patch Set Updates (PSU) and patches to the clone database installation. Applying all types of patches, it's likely the database upgrade path is confirmed as per Oracle certification matrix.

Disk Space Requirements

The Database Upgrade Assistant (DBUA) report the disk space usage and requirements. Ascertain to meet the minimum recommended disk space for ASM disk groups, flashback logs and archived log files. The disk space requirements take in consideration the selected DBUA recovery option. For a clone database created from backup, it's not necessary to preserve any kind of DBUA option as the clone database may be recreated in case of database upgrade failure and re-attempt.

System Global Area Requirements

Allocate enough memory the shared pool and Java pool. This is done either using Automatic Shared Memory Management (ASMM) or manually. Failure to allocate the required memory causes the database upgrade terminates with errors.

Now, let's perform the actual upgrade phases: Pre-Upgrade, Upgrade and Post-Upgrade. Figure 7-xx presents the upgrade as a top-down approach highlighting the tasks for a successful database upgrade projects.

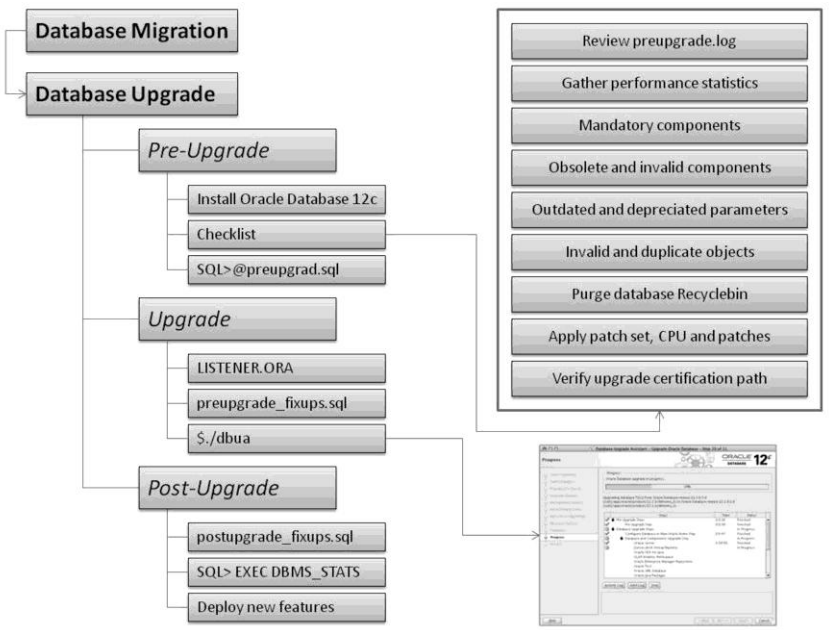


Figure 7-x: Database Upgrade Workflow

Database Pre-Upgrade

Before starting Oracle Database 12c Database Upgrade Assistant (DBUA), the pre-upgrade script (preupgrd.sql) must be executed. It quickly generates a pre-upgrade actions report. The pre-upgrade script is shipped with Oracle Database 12c and executed on the source database, it performs sequence of checks against the database, but do not perform any action.

The Pre-upgrade report includes the following sections:

- Renamed parameters
- Obsolete and deprecated parameters
- Component list and status
- Tablespace and their space usage
- Pre-Upgrade checks and required actions
- Pre-Upgrade recommendations

From Oracle Database 12c software installation directory on the clone database server, follow the steps below to execute the pre-upgrade script.

```
$ cd /u01/app/oracle/product/12.1.0/db_1/rdbms/admin
$ sqlplus /nolog

SQL*Plus: Release 11.2.0.3.0 Production on Wed Apr 23 18:49:26 2014

Copyright (c) 1982, 2011, Oracle.  All rights reserved.

SQL> CONN / AS SYSDBA
Connected.
SQL> @preupgrd.sql
Loading Pre-Upgrade Package...
Executing Pre-Upgrade Checks...
Pre-Upgrade Checks Complete.
      ************************************************************

Results of the checks are located at:
 /u01/app/oracle/cfgtoollogs/T1C1/preupgrade/preupgrade.log

Pre-Upgrade Fixup Script (run in source database environment):
 /u01/app/oracle/cfgtoollogs/T1C1/preupgrade/preupgrade_fixups.sql

Post-Upgrade Fixup Script (run shortly after upgrade):
```

```
 /u01/app/oracle/cfgtoollogs/T1C1/preupgrade/postupgrade_fixups.sql

      ************************************************************

      Fixup scripts must be reviewed prior to being executed.

      ************************************************************

      ************************************************************
               ====>> USER ACTION REQUIRED       <<====
      ************************************************************

 The following are *** ERROR LEVEL CONDITIONS *** that must be addressed
                prior to attempting your upgrade.
          Failure to do so will result in a failed upgrade.

        You MUST resolve the above errors prior to upgrade

      ************************************************************

SQL>
```

Because the clone database and Oracle Database 12c are on the same host, another listener LISTENER2 is configured and started to support the upgraded database. Export the environment variables TNS_ADMIN and LD_LIBRARY_PATH as shown below.

```
$export TNS_ADMIN=/u01/app/oracle/product/12.1.0/db_1/network/admin
$cd /u01/app/oracle/product/12.1.0/db_1/bin
$./lsnrctl start LISTENER2
$export LD_LIBRARY_PATH=/u01/app/oracle/product/12.1.0/db_1/lib
```

Database Upgrade

The Near Zero-Downtime Database Upgrade requires the use of Oracle GoldenGate as part of the database upgrade technique. There are two methods for upgrade an Oracle Database.

- Oracle Database Upgrade Assistant
- Manual Upgrade using SQL

Oracle Corporation recommends using the Oracle Database Assistant (DBUA). The completion of the Database pre-upgrade step checklist ascertains the successful completion of the database upgrade using Oracle Database Assistant (DBUA). The database upgrade timeframe is approximately 90-180 minutes regardless of the database size, and it does not add, modify or delete data.

Database Upgrade Assistant

The Oracle Database Upgrade Assistant (DBUA) is Oracle recommend method for upgrading the database. There are two interface options to execute DBUA:

- Graphical User Interface (GUI)
- Silent Interface

The Database Upgrade Assistant (DBUA) guides you through the upgrade steps, by finding and implementing recommendations to ensure the database is successfully completed. DBUA interface shows the progress of the database component and the completion status, at the same time writes to log files. DBUA alters and/or creates auxiliary files to support the upgraded database, provision database recoverability in the even to of failure and fully supports database running on Real Application Clusters. For highly integrated and automated upgrade, DBUA is available on silent mode from the command-line.

However; when the Database Administrator requires full control of the database upgrade runtime, the upgrade is done manually using SQL upgrade commands only. This method is more error prone when compared with the DBUA.

Starting Oracle Database Upgrade Assistant

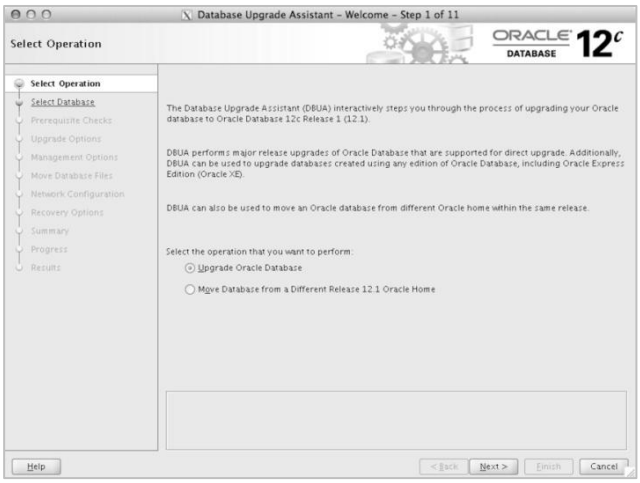
The Database Upgrade Assistant (DBUA) is started from Oracle Database 12c home by executing the following steps:

```
$ /u01/app/oracle/product/12.1.0/db_1/bin/dbua
```

Step 1 of 11: Select Operation

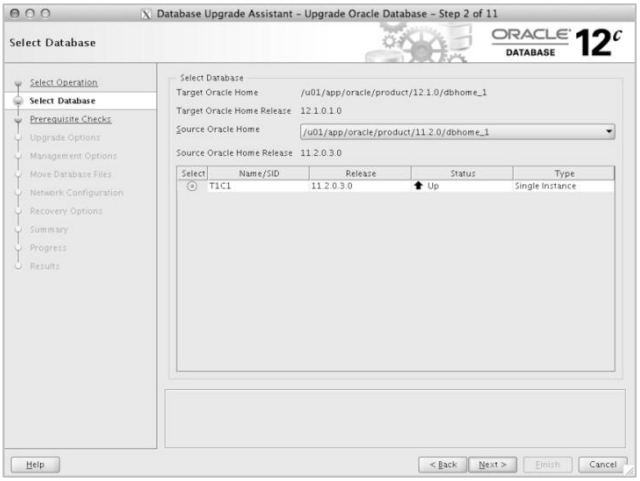
The Database Upgrade Assistant (DBUA) supports the following type of operations:

- Upgrade Oracle Database
- Merge Database from a Different Release 12.1 Oracle Home



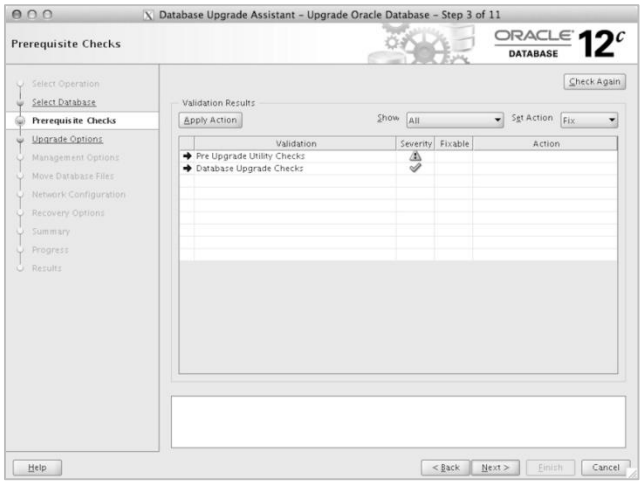
Step 2 of 11: Select Database

DBUA detects the source and target database software releases from /etc/oratab file. From the list of databases, select the database name for upgrade. The status of the database must be Up.



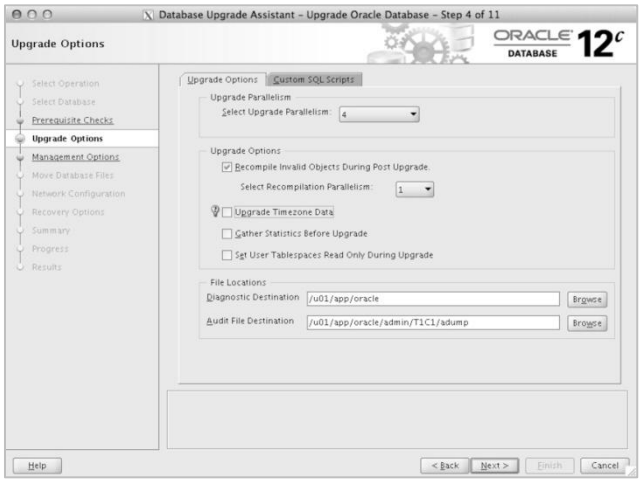
Step 3 of 11: Prerequisite Check

This step checks the prerequisite for the target database. Because of deprecated components, deprecated parameters some alerts will be generated. Examine the alerts generated which may require taking actions before proceeding further.



Step 4 of 11: Upgrade Options

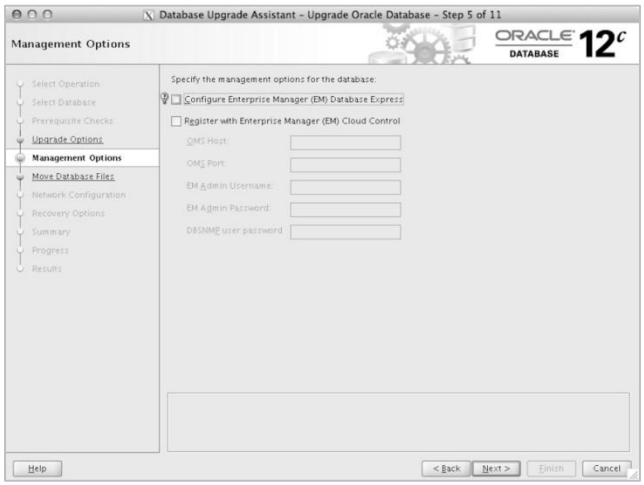
Select the database upgrade parallelism, which is set to 4 by default. Parallelism enhances the database upgrade performance and reduces the downtime incurred during the database data dictionary upgrade. From the Upgrade Options select 'Recompile Invalid Object During Post Upgrade', This eliminates the need of recompiling invalid object manually by running the supplied script 'utlrp.sql'.



Step 5 of 11: Management Options

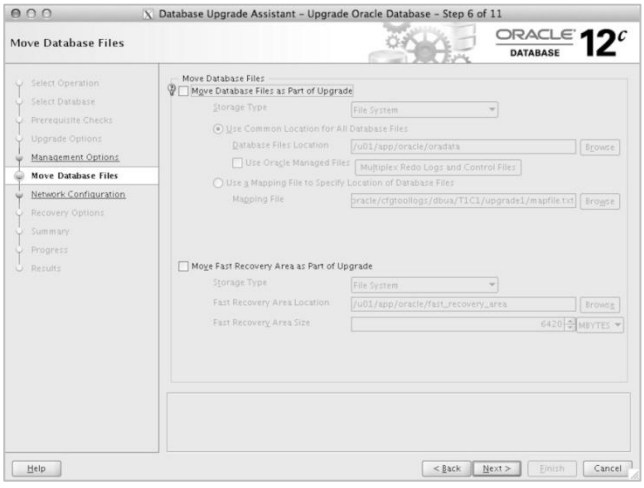
Oracle Enterprise Manager 11g Database control is deprecated by Oracle Database 12c. There are two management options are available:

- Enterprise Manager (EM) Database Express
- Enterprise Manager (EM) Cloud Control



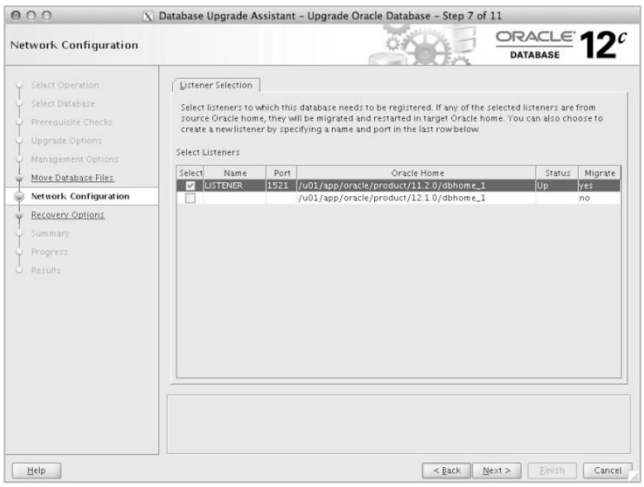
Step 6 of 11: Move Database Files

This enables you to move the database files and Fast Recovery Area (FRA) location. The storage types available are File System and Automatic Storage Management (ASM). For File System you may select the option Oracle Manage File (OMF) option.



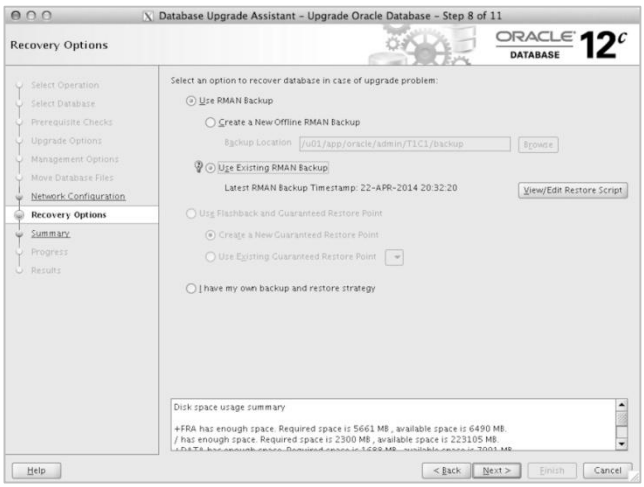
Step 7 of 11: Network Configuration

Select the listeners from the available list. The target database will be registered with the selected listeners. Or enter the listener name and the port number to be created by DBUA.



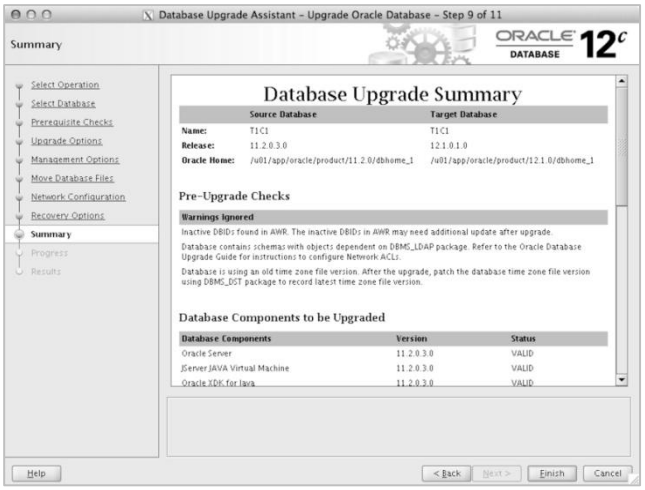
Step 8 of 11: Recovery Options

There are two recovery options: Use RMAN backup or Database Flashback. Because this is a clone database from an existing backup, select the option 'I have my own backup and restore strategy'. Consider the disk space requirements for +DATA, +FRA, flashback logs and archived log file. Disk space requirements must be met before proceeding further, if any.



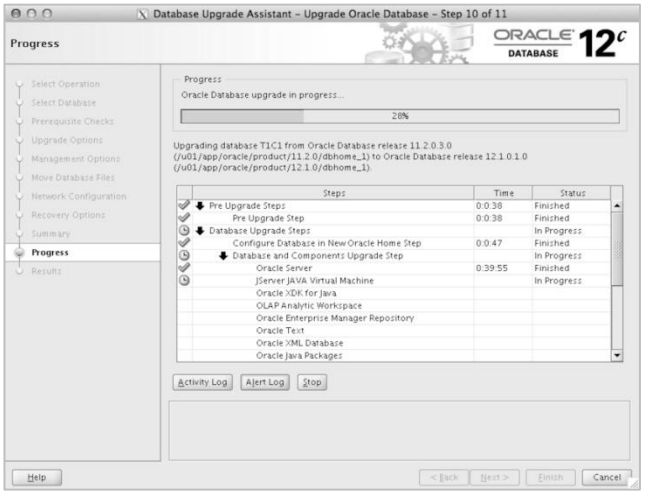
Step 9 of 11: Summary

Review the database upgrade summary sections. Verify the source and target database details – Database name, release and home. The Pre-Upgrade Checks section lists the issues found and the corresponding actions. The summary includes Database parameters old and changed values as recommended by the pre-upgrade script. Click 'Finish' to begin the database upgrade.



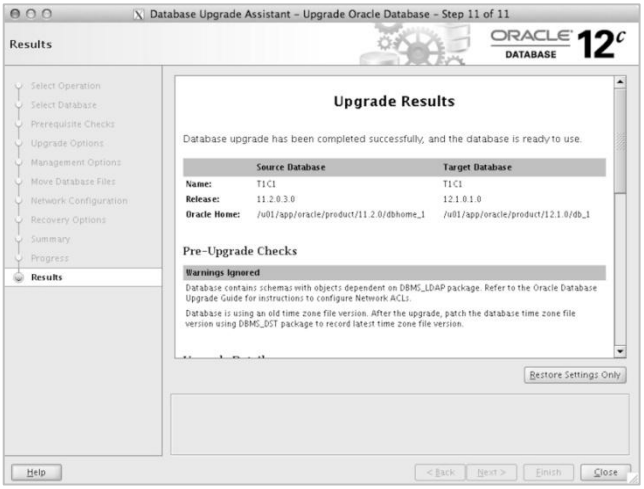
Step 10 of 11: Progress

Depending on the number of components to upgrade, the timeframe varies. An average of 90-120 minutes is common to complete the upgrade for all components.



Step 11 of 11: Upgrade Result

Review the upgrade results. For each step, DBUA generate a log file. Review the log files for recommendations as a post-upgrade step.



Verify Database Upgrade

Congratulation! The database upgrade successfully completed. The target database instance is started and the database is opened. Query the dynamic view V$VERSION to confirm the database core components versions.

```
$ . oraenv
ORACLE_SID = [T1C1] ? T1C1
The Oracle base remains unchanged with value /u01/app/oracle
$ sqlplus /nolog

SQL*Plus: Release 12.1.0.1.0 Production on Sun Apr 27 19:36:50 2014

Copyright (c) 1982, 2013, Oracle.  All rights reserved.

SQL> CONN / AS SYSDBA
Connected.
SQL> SELECT banner
  2  FROM   v$version;

BANNER
--------------------------------------------------------------------------------
Oracle Database 12c Enterprise Edition Release 12.1.0.1.0 - 64bit Production
PL/SQL Release 12.1.0.1.0 - Production
CORE 12.1.0.1.0     Production
TNS for Linux: Version 12.1.0.1.0 - Production
NLSRTL Version 12.1.0.1.0 - Production

SQL>
```

Silent Mode Upgrade

Another method of performing database upgrade is running DBUA on silent mode. The main reason of running DBUA on silent mode is the luck of X Windows manager, but it also provides more control over DBUA GUI runtime. The following script illustrates running DBUA on silent mode.

```
$ /u01/app/oracle/product/12.1.0/db_1/bin/dbua -silent \
> -sid T1C1 \
> -oracleHome /u01/app/oracle/product/11.2.0.3/dbhome_1 \
> -diagnosticDest /u01/app/oracle \
> -autoextendFiles \
> -recompile_invalid_objects true \
> -degree_of_parallelism 2 \
> -upgradeTimezone \
> -emConfiguration NONE \
> -keepHiddenParams \
> -gatheringStatistics \
> -createGRP true \
> -upgrade_parallelism 2
```

The timeframe of database upgrade using DBUA silent mode is the same as DBUA GUI mode. Performing the pre-upgrade checklist ensures database components upgraded successfully.

Manual Upgrade

The manual upgrade provides more control when performing the database upgrade. The steps for performing manual database upgrade using SQL command are highlighted below.

- On the source database, run the pre-upgrade script tool shipped with Oracle Database 12c. The script file is preupgrd.sql.
- Create a whole database backup using RMAN, this serves as the database recovery solution in the event of failure.
- The installation of Oracle Database 12c is already completed. Prepare the new Oracle Home database auxiliary files; parameters, password file and network files.
- Shutdown the source database
- While still connected as SYSDBA, execute the SQL upgrade command, STARTUP UPGRADE.
- Restart the database instance
- Run the post-upgrade script utlu121s.sql
- Run the post-upgrade script catupst.sql, which performs the remaining database upgrade operations.

Database Post-Upgrade

The post-upgrade configures files related to Oracle Database 12c. When using Oracle Database Configuration Assistant (DBUA), the database post-upgrade changes are minimal. The following is the list of post-upgrade changes:

- Update the Operating System environment variables
- Update /etc/oratab, for manual upgrade
- Update client files
- Upgrade Recovery Manager catalog, if used
- Enable database vault, if used.
- Deploy Oracle Database 12c new features, gradually.
- Enable unified auditing

Enable Database Flashback

Using the Flashback Database, it quickly brings the database to an earlier point in time by undoing changes occurred since that time. The advantage of the database Flashback, it's fast operation compared to the traditional RMAN restore-recover operations to perform point-time database recovery.

The database Flashback step is to support rewinding the database after Phase V – Application Regression Testing is completed. The purpose is to reestablish the target database without the need to re-clone the database. Follow the steps below to configure and enable database Flashback.

Though there are several database Flashback options which are: Timestamp, System Change Number (SCN), restore point and Flashback sequence number, the use of SCN or restore point is preferred. Begin by obtaining the current database System Change Number (SCN). This will be used on subsequent step to Flashback the database.

```
SQL> SELECT current_scn
  2  FROM   v$database;

CURRENT_SCN
-----------
    9294723

SQL>
```

Enabling the database Flashback starts a background process RVWR to write the Flashback logs to the Fast Recovery Area (FRA). The amount of Flashback is set to 129600 minutes (90 days) to target, but not to guarantee, enough timeframe to

complete the applications functional testing. The Fast Recovery Area (FRA) size is proportional the retention target.

```
SQL> ALTER SYSTEM SET DB_FLASHBACK_RETENTION_TARGET=129600
  2  SCOPE=both;

System altered.

SQL> ALTER DATABASE FLASHBACK on;

Database altered.

SQL> SELECT flashback_on
  2  FROM   v$database;

FLASHBACK_ON
------------------
YES

SQL>
```

Settings the parameter DB_FLASHBACK_RETENTION_TARET is optional, but it specifies an upper limit, in minutes, on how far back the database can be flashed back, the default values is 1440 minutes (1 day). When setting the value for the parameter DB_FLASHBACK_RETENTION_TARET consider two important elements, the timeframe to complete the testing and the size of the Fast Recovery Area (FRA) to sustain the accumulative database flashback logs.

Keep monitoring the storage requirement and usage while the database Flashback is enabled. Use of Oracle Enterprise Manager or perform queries against the database dynamic views:

V$FLASHBACK_DATABASE_LOG
V$FLASH_RECOVERY_AREA_USAGE

Phase V: Applications Regression Testing

Before the transition to the new system, functional and technical testing must be thoroughly conducted. The purpose of various types of testing is to uncover issues that may be a raised due to changes and enhancements to the database management system internals. A very common area to address is the behavior of the application with the new database optimizer. Oracle Corporation has addressed this issue using Oracle Real Application Testing suite, which consists of two solutions:-

- Oracle Database replay
- SQL Performance Analyzer

Oracle Database Replay

Oracle Database Replay is the Database upgrade companion tool. The database replay enables users to perform real-time, real-world testing by capturing actual database workloads on the production system and replaying them on the clone database system. The test on the clone database can be performed with production characteristics including timing and transactions concurrency. The main feature of Database Replay, it provides analysis and reporting to highlight potential problems such as a report is taking longer time to complete on the new environment. The Oracle Database Replay has two interfaces; Oracle Enterprise Manager and Oracle PL/SQL supplied packages. Figure 7-xx shows Oracle Database Replay workflow, which composed of the three tasks:-

- Capture Workload, captures a workload from the production (source) environment. The workload timeframe is done during normal business hours to create real-world baseline.
- Pre-process Workload, this task prepare the capture files for replay on the target system. The workload capture files must be moved to the target system before the next task when the pre-process is done on the source system.
- Replay Workload, this task replay the pre-processed workload on the clone database.

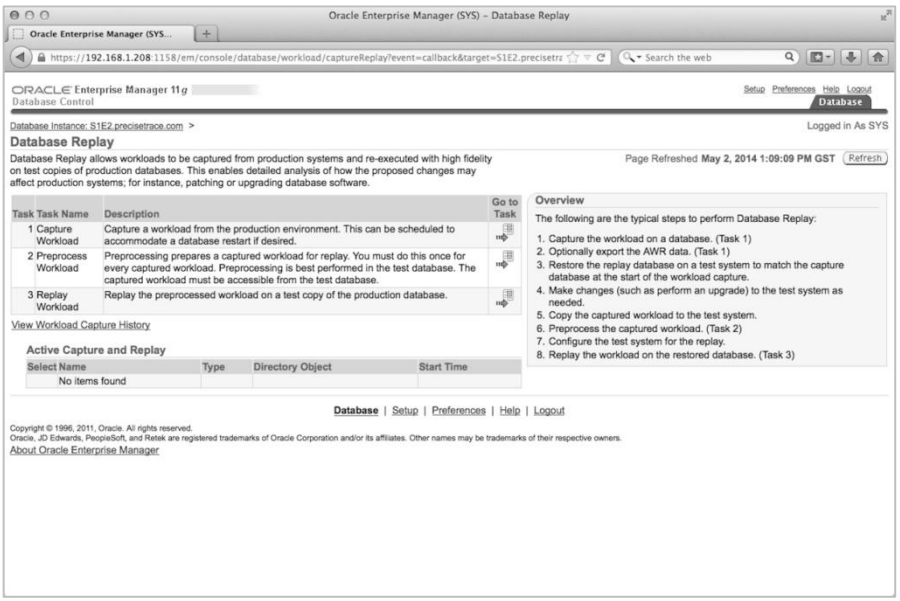


Figure 7-xx: Oracle Database Replay

SQL Performance Analyzer

The SQL Performance Analyzer (SPA) can be used to quickly and accurately predicts and prevents the potential performance problems for any database changes that alter the original sequence of steps of the execution plan. The change can be any of the following:

- Database upgrade
- Operating System upgrade
- Database Migration
- Deploying new application
- Database schema changes, adding indexes
- Statistic gathering
- Database parameter changes
- Hardware and Operating System changes

Figure 7-xx depicts Oracle SQL Performance Analyzer (SPA). The workflow run is either predetermined or guided workflow. Select the workflow that corresponds to the upgrade path, Upgrade from 11g to 12c.

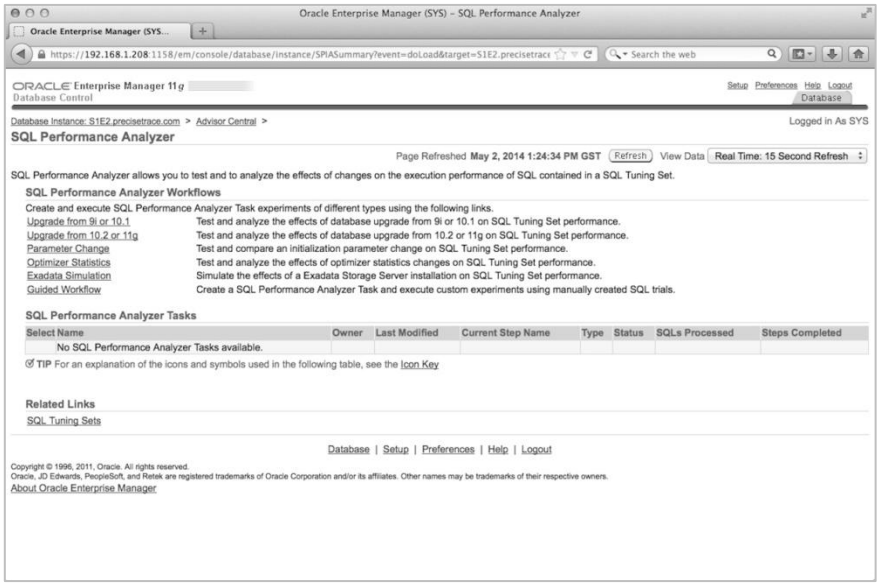


Fig 7-xx: Oracle SQL Performance Analyzer (SPA)

This is an overview of Oracle Database Real Application Testing Suite solutions. Use either Oracle Database Replay or SQL Performance Analyzer. Phase V – Applications

Regressions Testing is out of the scope of this book, as our objective is limited to implementation of using Oracle GoldenGate 12c for Near Zero-Downtime Migration and Upgrade of the Oracle Database only.

Perform Database Flashback

After completing the functional and technical testing, the database must be restored to the state just after the upgrade is completed. The following procedure illustrates performing database flashback.

```
SQL> SHUTDOWN IMMEDIATE
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> STARTUP MOUNT
ORACLE instance started.

Total System Global Area   551165952 bytes
Fixed Size                 2230232 bytes
Variable Size              373295144 bytes
Database Buffers           171966464 bytes
Redo Buffers               3674112 bytes
Database mounted.
SQL> FLASHBACK DATABASE TO SCN 9294723;
SQL> ALTER DATABASE OPEN RESETLOGS;

Database altered.

SQL>
```

Query the dynamic view V$SESSION_LONGOPS to monitor the progress of the database Flashback. Another method is to monitor the instance alert file.

```
$ cd /u01/app/oracle/diag/rdbms/t1c1/T1C1/trace
$ tail -f alert_T1C1.log
Flashback Restore Start
Flashback Restore Complete
Flashback Media Recovery Start
Serial Media Recovery started
Recovery of Online Redo Log: Thread 1 Group 6 Seq 3 Reading mem 0
  Mem# 0: +DATA/t1c1/onlinelog/group_6.266.846267799
  Mem# 1: +FRA/t1c1/onlinelog/group_6.260.846267803
Incomplete Recovery applied until change 9319751 time 05/03/2014 07:23:44
Flashback Media Recovery Complete
Completed: flashback database to scn 9319750
```

Disable Database Flashback

Now, disable database Flashback. The disable command terminates the Flashback background process RVWR and deletes existing Flashback logs stored at Fast Recovery Area (FRA).

```
SQL> ALTER DATABASE FLASHBACK off;

Database altered.

SQL> SELECT flashback_on FROM v$database;

FLASHBACK_ON
------------------
NO

SQL>
```

Phase VI: Apply Target Database Synchronization

Simultaneity, while the clone database is going through an upgrade to Oracle Database 12c, Oracle GoldenGate is running on the source system (S1E2) continuously capturing and sending transactions, trail files, to the target system (T1C1). After successfully completing the database upgrade, the target database synchronization starts from the System Change Number (SCN) identified by the RMAN DUPLICATE command, saving the Replicat processing time for reading, applying and handling conflicts of unnecessary transactions. It's recommended the target Oracle GoldenGate manager process is configured without the trail files purge option. Remote trail files purge will be configured after migration of users, when it certain the trail files are no longer needed. Figure 7-xx depicts the synchronization of the target database.

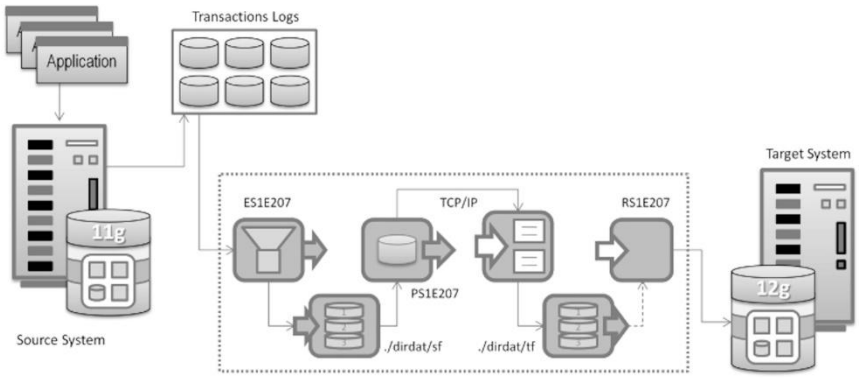


Figure 7-xx: Synchronization of the target database

The Replicat process applies transactions to the target database in the order of commit System Change Number (SCN), preserving data integrity of the target database while the applications continues to run on the source database. Implement parallel downstream Extract-Replicat processes carefully to avoid escalation of data

integrity issues. The list of tables must be only related to tables handled by the Extract-Replicat group.

Remote Trail Files

The remote trail is shipped from the source to target system using the data pump process. The remote trail files are accumulated, ultimately applied to synchronize the target database. The size of the remote trail is proportional to the source transactions volume and the timeframe to complete the database migration and upgrade. Verify and compare the trail files on the source and the it's availability on the target system.

```
$ cd /u01/app/ggs/12.1.0/gghome_1/dirdat
$ ls -ltr
total 395056
-rw-r----- 1 oracle oinstall   492190 Apr 11 11:24 tf000000
-rw-r----- 1 oracle oinstall 15999967 Apr 11 21:00 tf000001
-rw-r----- 1 oracle oinstall 15999941 Apr 12 06:37 tf000002
-rw-r----- 1 oracle oinstall 15999756 Apr 12 16:15 tf000003
-rw-r----- 1 oracle oinstall 15999820 Apr 14 21:09 tf000004
-rw-r----- 1 oracle oinstall 15999990 Apr 15 06:50 tf000005
-rw-r----- 1 oracle oinstall 15999742 Apr 15 16:32 tf000006
-rw-r----- 1 oracle oinstall 15999867 Apr 16 02:15 tf000007
-rw-r----- 1 oracle oinstall 15999954 Apr 16 11:59 tf000008
-rw-r----- 1 oracle oinstall 15999940 Apr 16 21:44 tf000009
-rw-r----- 1 oracle oinstall 15999999 Apr 17 07:30 tf000010
-rw-r----- 1 oracle oinstall 15999852 Apr 17 17:17 tf000011
-rw-r----- 1 oracle oinstall 15999867 Apr 18 03:05 tf000012
-rw-r----- 1 oracle oinstall 15999872 Apr 18 12:59 tf000013
-rw-r----- 1 oracle oinstall 15999870 Apr 18 22:51 tf000014
-rw-r----- 1 oracle oinstall 15999906 Apr 19 08:44 tf000015
-rw-r----- 1 oracle oinstall 15999879 Apr 19 18:36 tf000016
-rw-r----- 1 oracle oinstall 15999783 Apr 20 04:29 tf000017
-rw-r----- 1 oracle oinstall 15999984 Apr 20 14:24 tf000018
-rw-r----- 1 oracle oinstall 15999955 Apr 21 00:19 tf000019
-rw-r----- 1 oracle oinstall 15999811 Apr 21 10:16 tf000020
-rw-r----- 1 oracle oinstall 15999790 Apr 21 20:14 tf000021
-rw-r----- 1 oracle oinstall 15999876 Apr 22 06:13 tf000022
-rw-r----- 1 oracle oinstall 15999796 Apr 22 16:12 tf000023
-rw-r----- 1 oracle oinstall 15999887 Apr 23 02:13 tf000024
-rw-r----- 1 oracle oinstall 15999797 Apr 23 12:16 tf000025
-rw-r----- 1 oracle oinstall  3440714 Apr 23 14:25 tf000026
$
```

Checkpoint Database Table

Create the checkpoint table used by the Replicat process. The default checkpoint location is a database table as defined by the GLOBALS parameter files.

```
GGSCI (ggs-target) 9> DBLOGIN USERID ggs admin, PASSWORD oracle
Successfully logged into database.
```

```
GGSCI (ggs-target) 10> ADD CHECKPOINTTABLE

No checkpoint table specified. Using GLOBALS specification
(ggs_admin.chkpttab)...

Successfully created checkpoint table ggs_admin.chkpttab.

GGSCI (ggs-target) 11>
```

The Delivery Group

The delivery group is composed of the parameter file RS1E207.PRM, the Replicat process PS1E207, the remote trail ./dirdat/tf, and the checkpoint file ./dirchk/ES1E207.CPR. The delivery process uses the target database SQL native calls to apply transactions by reading the remote trail files.

Replicat Parameter File

Though the Replicat process is starting from the System Change Number (SCN) identified by RMAN DUPLICATE command, to prevent duplicate records error message, the HANDLECOLISSIONS parameter is enabled.

```
GGSCI (apps-srv1) 3> EDIT PARAMS rs1e207
```

```
REPLICAT rs1e207
SETENV (ORACLE SID="T1C1")
SETENV (ORACLE_HOME="/u01/app/oracle/product/12.1.0/db_1")
SETENV (NLS LANG="AMERICAN AMERICA.AL32UTF8")
USERID ggs_admin, PASSWORD oracle
ASSUMETARGETDEFS
HANDLECOLISSIONS
DISCARDFILE ./dirrpt/rs1e207.dsc, PURGE
MAPEXCLUDE osm$repapi.OSM_DOCUMENTS;
MAP osm$repapi.*, TARGET osm$repapi.*;
```

Define Replicat Process

The Replicat process reads from ./dirdat/tf remote trail and apply transactions to the target database – T1C1.

```
GGSCI (ggs-target) 12> ADD REPLICAT rs1e207, EXTTRAIL ./dirdat/tf
REPLICAT added.

GGSCI (ggs-target) 13>
```

Remote Trail Settings

Verify the details of the remote trial. The Replicat process applies the remote trail files to the target database.

```
$ ./ggsci

Oracle GoldenGate Command Interpreter for Oracle
Version 12.1.2.0.0 17185003 OGGCORE_12.1.2.0.0_PLATFORMS_130924.1316_FBO
Linux, x64, 64bit (optimized), Oracle 11g on Sep 25 2013 00:31:13
Operating system character set identified as UTF-8.

Copyright (C) 1995, 2013, Oracle and/or its affiliates. All rights reserved.

GGSCI (ggs-source) 2> info rmttrail ./dirdat/tf

       Extract Trail: ./dirdat/tf
             Extract: PS1E207
               Seqno: 1
                 RBA: 9045389
           File Size: 16M

GGSCI (ggs-source) 3>
```

The GGSCI INFO command shows the location, the Replicat name, the current sequence, the current RBA (Relative Byte Address) and the trail file size.

Start and Monitor Replicat Process

The Replicat is started after a specified System Change Number (SCN) using the option AFTERCSN of the START REPLICAT GGSCI command.

```
GGSCI (ggs-target) 16> INFO replicat RS1E207

REPLICAT   RS1E207   Initialized   2014-05-04 05:16   Status STOPPED
Checkpoint Lag       00:00:00 (updated 95:49:26 ago)
Log Read Checkpoint  File ./dirdat/tf000000
                     First Record  RBA 0


GGSCI (ggs-target) 17> START REPLICAT rs1e207 AFTERCSN 14274705

Sending START request to MANAGER ...
REPLICAT RS1E207 starting


GGSCI (ggs-target) 18> INFO ALL

Program     Status      Group       Lag at Chkpt  Time Since Chkpt

MANAGER     RUNNING
REPLICAT    RUNNING     RS1E207     08:57:15      00:00:00
```

```
GGSCI (ggs-target) 19>
```

Perform quick basic test using SQL*Plus, this ascertains the Replicat is applying transaction to the target database tables. Repeat to confirm rows count is changing.

```
SQL> CONN osm$repapi/oracle
Connected.
SQL> SELECT COUNT(*)
  2  FROM   customers;

  COUNT(*)
----------
   2553617

SQL>
```

At this point, the source and target database are synchronizing, the source transactions are applied in real-time to the target database achieving sub-seconds of latency. Now, it's the moment of users migration to the new system.

Phase VII: Data Verification

Prior to diverting applications from source environment running Oracle Database 11g to the target environment running Oracle Database 12c, an important task must be performed and delivered with 100% accuracy - *Data Verification*. Data verification compares data sets on source and target database and it's best performed using Oracle GoldenGate Veridata. Oracle GoldenGate Veridata assures data consistency without the need to rely on in-house developed SQL scripts or applications to compare table-to-table; This becomes a complex task for very large database (VLDB) environment with large number of tables. In such environment, Oracle GoldenGate Veridata play the major rule by automating the creation of compare pairs groups using different methods such as wildcard patterns, which eliminates creating compare pairs explicitly.

Oracle GoldenGate Veridata

Oracle GoldenGate Veridata performs advanced data comparisons between compare pair groups of the source and target database. The use of Oracle GoldenGate Veridata requires software installation and configuration of:

- Oracle GoldenGate Server database repository
- Oracle GoldenGate Server
- Oracle GoldenGate Veridata agent (Java agent) on source and target systems

To perform comparison on data sets, Oracle GoldenGate Veridata requires the configuration of related components, use Oracle GoldenGate Veridata web-based administration tool to configure the following:

- Configure connections to source (S1E2) and target (T1C1) database
- Configure groups
- Configure compare pair groups
- Configure compare jobs

The process of performing comparisons of data sets between the source and target database is illustrated by presenting the following major steps.

- Compare pair groups job runtime
- Initial compare step
- Out-of-sync confirmation step

For details, refer to Chapter 6 regarding implementing Oracle GoldenGate Veridata.

Compare Pair Groups Job Runtime

After configuring connections, compare groups, compare pair groups and compare job, the job is executed for the compare groups as shown in Figure 7-xx. There two compare groups; OSM_APPS for transactional compare pairs and OSM_LOOKUP for lookup compare pairs.

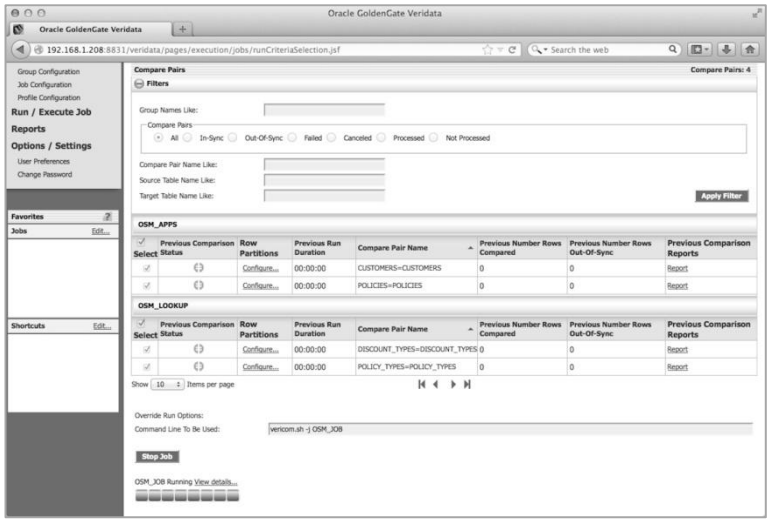


Figure 7-xx: Compare pair group job runtime progress

The compare groups and compare pairs configured are shown in Table 7-x.

Compare Group	Compare Pair
OSM_APPS	CUSTOMERS=CUSTOMERS POLICIES= POLICIES
OSM_LOOKUP	POLICY_TYPES= POLICY_TYPES DISCOUNT_TYPES= DISCOUNT_TYPES

Table 7-x: Compare Groups Configuration

Initial Compare Step

While Oracle GoldenGate is performing Change Data Capture (CDC) on the source database, sending and applying the transactions to the target database, data deviation continues to exist. To ascertain accurate compare result, the compare pair groups includes all tables; transactional and lookup tables. Figure 7-xx shows the initial compare with out-of-sync results, this occurs because of the delay on applying the transactions to the target database.

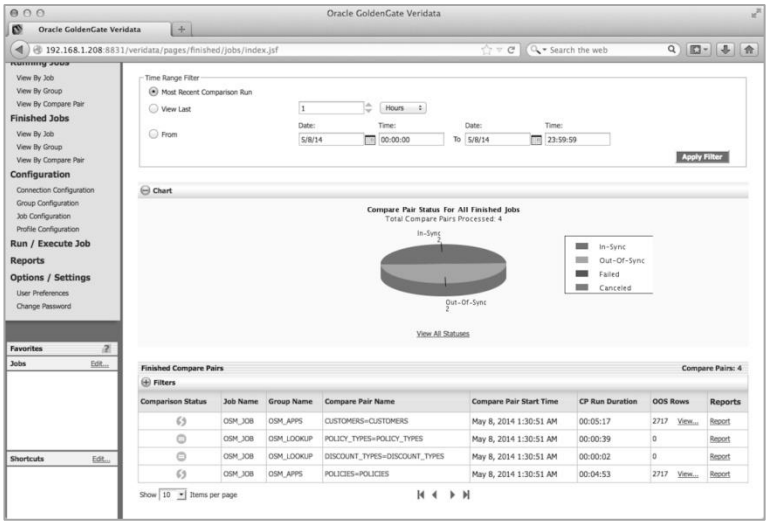


Figure 7-xx: Initial Compare Step – Out of Sync

The OOS rows column indicates the extent of data mismatch, click on View… link to display the Out-of-Sync summary report. Out-of-Sync is expected to occur and the extent depends on the current transactions volume of the source database.

Out-of-Sync Confirmation

For an enterprise running multiple application servers, the next steps is to disconnect the application servers gradually during the minimum database workload. The purpose of using this approach is to achieve sub-seconds latency between the source and target database. Use Oracle GoldenGate GGSCI LAG command to determine the current lag of the Replicat process, when it becomes within the expected upgrade downtime timeframe, it's time to disconnect the remaining application servers to allow data convergence. When the lag is becomes zero, re-run of the compare job to obtain the current Out-of-Sync result.

For our case, disconnecting the application server is terminating the PL/SQL database application simulator by issuing the SQL KILL SESSION command.

```
SQL> SELECT s.sid, s.serial#
  2  FROM   v$session s, v$sqlarea a
  3  WHERE  s.username = 'OSM$REPAPI' AND
  4         s.sql_id = a.sql_id       AND
  5         UPPER(a.sql_text) like '%TRANS_LOAD%';

       SID    SERIAL#
---------- ----------
     19       1559

SQL> ALTER SYSTEM KILL SESSION '19, 1559' IMMEDIATE;

System altered.

SQL>
```

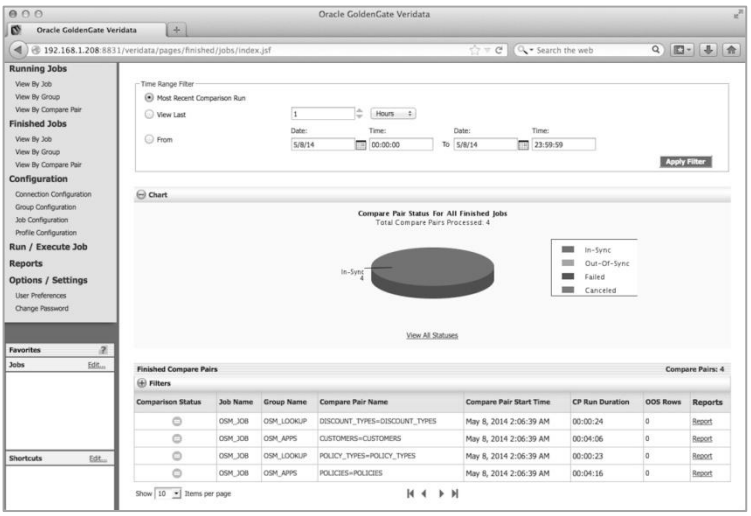


Fig 7-yy: Synchronized Source and Target Database

Figure 7-xx shows an in-sync compare pair groups following terminating the PL/SQL transactions simulator; All transactions logs from the source database have been shipped and completely applied the transactions to the target database.

Achieving Near Zero-Downtime Optimization

The major factors impacting the database upgrade downtime are listed below, they are all related and listed in the order of higher-lower impact.

- Transactions volume on the source database during out-of-sync confirmation

  High transactions volume effect the primary Extracts capture from the transactions logs. Consider configuring parallelism of Extracts, data pumps and Replicat processes. Also, consider disabling applications access during low system activities to minimize the transactions logs

- Extracts (primary and secondary) and Replicat lag

  Within the downstream, there are three types of lags: The primary Extract lag is the time difference of writing to the local trail and the transactions logs (online redo log file). The data pump Extract lag is the time difference between writing to the remote trail and reading from local trail. The Replicat lag is the time difference of applying transactions to the target database and reading from the remote trail. Fine tuning the lags reduces the downtime. Use the GGSCI LAG command to determine the current lag as shown next.

```
GGSCI (ggs-target) 5> LAG REPLICAT RS1E207

Sending GETLAG request to REPLICAT RS1E207 ...
No records yet processed.
At EOF, no more records to process.

GGSCI (ggs-target) 6>
```

- Application Servers redirection timeframe

  The redirection of Application Servers instances increases the downtime. An example is Oracle Application Express, the changes are made to DADS.CONF file to specify the new host name, listener port number and SID. The preparation of redirection changes ahead of time reduces the downtime as application servers are started immediately up on the source and target database are confirmed synchronized.

Removing Oracle GoldenGate Configurations

After the applications servers are diverted to the target system, perform the following tasks to remove Oracle GoldenGate instance configuration. However; do not de-install Oracle GoldenGate software and instance if there are more 24/7 databases to be upgraded. The following is the list of actions for removing the configurations.

- Stop, backup parameter files and remove Extracts configurations on the source system.
- Stop the source instance manager
- Delete the local trail files
- Stop, backup parameter files and remove Replicat configurations on the target system.
- Stop the target instance manager.
- Delete the remote trail files.
- Drop the checkpoint table

Refer to Chapter 2 for details regarding Oracle GoldenGate 12c de-installation and removing configurations.

This concludes using Oracle GoldenGate for Near Zero-Downtime Migration and Upgrade. The chapter followed a phased approach, which may varies by combining, adding, modifying or eliminating specific tasks and/or phases, but ultimately meeting the objective – Using Oracle GoldenGate for Near Zero-Downtime Migration and Upgrade.

Automatic Storage Management Disk Provisioning

Automatic Storage Management (ASM) employs provisioned disks for creating disk groups. The following technique simulates disk partitioning by creating raw files using the Linux operating system command 'dd'. The following sequence of steps executed for creating and setting the permissions. *The following setup is for training purpose, it should not be employed for production environment.*

Provisioning ASM Disks

One of the tool provided by Linux/UNIX operating system is the
Create the directory /u01/asmdisks, then run the 'dd' command. In between each 'dd' command, the Operating System SYNC command is executed to write buffered data to disk, releasing memory for the next command 'dd' command.

```
# mkdir -p /u01/asmdisks
# sync; echo 3 > /proc/sys/vm/drop_caches
```

```
# dd if=/dev/zero of=/u01/asmdisks/_file_disk_01 bs=1k count=2359296
2359296+0 records in
2359296+0 records out
2415919104 bytes (2.4 GB) copied, 16.2344 seconds, 149 MB/s
# sync; echo 3 > /proc/sys/vm/drop_caches
# dd if=/dev/zero of=/u01/asmdisks/_file_disk_02 bs=1k count=2359296
2359296+0 records in
2359296+0 records out
2415919104 bytes (2.4 GB) copied, 16.2344 seconds, 149 MB/s
# sync; echo 3 > /proc/sys/vm/drop_caches
# dd if=/dev/zero of=/u01/asmdisks/_file_disk_03 bs=1k count=2359296
2359296+0 records in
2359296+0 records out
2415919104 bytes (2.4 GB) copied, 30.4624 seconds, 79.3 MB/s
# sync; echo 3 > /proc/sys/vm/drop_caches
# dd if=/dev/zero of=/u01/asmdisks/_file_disk_04 bs=1k count=2359296
2359296+0 records in
2359296+0 records out
2415919104 bytes (2.4 GB) copied, 23.2605 seconds, 104 MB/s
# sync; echo 3 > /proc/sys/vm/drop_caches
# dd if=/dev/zero of=/u01/asmdisks/_file_disk_05 bs=1k count=2359296
2359296+0 records in
2359296+0 records out
2415919104 bytes (2.4 GB) copied, 19.2781 seconds, 125 MB/s
# sync; echo 3 > /proc/sys/vm/drop_caches
# dd if=/dev/zero of=/u01/asmdisks/_file_disk_06 bs=1k count=2359296
2359296+0 records in
2359296+0 records out
2415919104 bytes (2.4 GB) copied, 14.8911 seconds, 162 MB/s
# sync; echo 3 > /proc/sys/vm/drop_caches
# dd if=/dev/zero of=/u01/asmdisks/_file_disk_07 bs=1k count=2359296
2359296+0 records in
2359296+0 records out
2415919104 bytes (2.4 GB) copied, 16.2389 seconds, 149 MB/s
# sync; echo 3 > /proc/sys/vm/drop_caches
#
```

Verify the list of raw files created on the directory.

```
# cd /u01/asmdisks/
# ls -al
total 23616048
drwxr-xr-x 2 root   root           4096 Apr 19 11:03 .
drwxr-xr-x 4 oracle oinstall       4096 Apr 19 10:59 ..
-rw-r--r-- 1 root   root     2415919104 Apr 19 11:00 _file_disk_01
-rw-r--r-- 1 root   root     2415919104 Apr 19 11:00 _file_disk_02
-rw-r--r-- 1 root   root     2415919104 Apr 19 11:01 _file_disk_03
-rw-r--r-- 1 root   root     2415919104 Apr 19 11:01 _file_disk_04
-rw-r--r-- 1 root   root     2415919104 Apr 19 11:02 _file_disk_05
-rw-r--r-- 1 root   root     2415919104 Apr 19 11:02 _file_disk_06
-rw-r--r-- 1 root   root     2415919104 Apr 19 11:02 _file_disk_07
#
```

Link the raw files created …

```
# losetup /dev/loop1 /u01/asmdisks/_file_disk_01
```

```
# losetup /dev/loop2 /u01/asmdisks/_file_disk_02
# losetup /dev/loop3 /u01/asmdisks/_file_disk_03
# losetup /dev/loop4 /u01/asmdisks/_file_disk_04
# losetup /dev/loop5 /u01/asmdisks/_file_disk_05
# losetup /dev/loop6 /u01/asmdisks/_file_disk_06
# losetup /dev/loop7 /u01/asmdisks/_file_disk_07
```

Link 2

```
# ln -s /dev/loop1 /dev/xvdb
# ln -s /dev/loop2 /dev/xvdc
# ln -s /dev/loop3 /dev/xvdd
# ln -s /dev/loop4 /dev/xvde
# ln -s /dev/loop5 /dev/xvdf
# ln -s /dev/loop6 /dev/xvdg
# ln -s /dev/loop7 /dev/xvdh
#
# chown -R oracle:oinstall /u01/asmdisks
# chmod -R 666 /u01/asmdisks
# chown oracle:oinstall /dev/loop[1-7]
# chmod 666 /dev/loop[1-7]
#
```

Link 3

```
#!/bin/sh
#
# chkconfig: 345 13 89
# description: Allocates the flat files in /u01/asmdisks as loopback
devices.
#           Creates links to those loopback devices to simulate the naming
#           convention of Oracle VM Server virtaual block devices.
#
# Source function library.
. /etc/init.d/functions
PATH=/sbin:/bin:/usr/sbin:/usr/bin
RETVAL=0
start()
{
     losetup /dev/loop1 /u01/asmdisks/_file_disk_01
     losetup /dev/loop2 /u01/asmdisks/_file_disk_02
     losetup /dev/loop3 /u01/asmdisks/_file_disk_03
     losetup /dev/loop4 /u01/asmdisks/_file_disk_04
     losetup /dev/loop5 /u01/asmdisks/_file_disk_05
     losetup /dev/loop6 /u01/asmdisks/_file_disk_06
     losetup /dev/loop7 /u01/asmdisks/_file_disk_07
     ln -s /dev/loop1 /dev/xvdb
     ln -s /dev/loop2 /dev/xvdc
     ln -s /dev/loop3 /dev/xvdd
     ln -s /dev/loop4 /dev/xvde
     ln -s /dev/loop5 /dev/xvdf
     ln -s /dev/loop6 /dev/xvdg
     ln -s /dev/loop7 /dev/xvdh
     chown oracle:oinstall /dev/loop[1-7]
     chmod 666 /dev/loop[1-7]
     success
     echo
}
```

```
stop()
{
     losetup -d /dev/loop1
     losetup -d /dev/loop2
     losetup -d /dev/loop3
     losetup -d /dev/loop4
     losetup -d /dev/loop5
     losetup -d /dev/loop6
     losetup -d /dev/loop7
     rm -f /dev/xvd[b-t]
     success
}
case "$1" in
     start)
           start
           ;;
     stop)
           stop
           ;;
     *)
           echo $"Usage: $0 {start|stop}"
           exit 1
esac
exit $RETVAL
```

Link 5

```
chmod 755 /etc/init.d/losetup
chkconfig losetup --add
chkconfig losetup on
chkconfig losetup –list
```

Summary

Employing Oracle GoldenGate for near zero-downtime migration and upgrade demonstrates a key application area of the software. Since Oracle GoldenGate is a multitask software, for enterprises running 24/7 systems, the software may be used to achieve near zero-downtime migration and upgrade while it's already installed and running on the system performing query offloading. This is an Oracle GoldenGate Return on Investment (ROI) strategy when deployed using best practices.

This chapter has demonstrated the migration and upgrade from Oracle Database 11g to Oracle Database 12c, the multitenant database, providing an innovative approach for databases consolidation. The topics covered are:

- Deploy Oracle GoldenGate on source and target
- Online database clone
- Migration to automatic storage management
- Upgrade from Oracle Database 11g to Oracle Database 12c
- Target database synchronization and data verification

Though the chapter has demonstrated the upgrade from Oracle Database 11g to Oracle Database 12c, when using Oracle GoldenGate, it enables a jumping upgrade approach. For example, direct upgrade from Oracle Database 9i to Oracle Database 12c. The jumping upgrade uses an initial-load from an active database rather than database clone. Such approach provides higher flexibility when compared with the traditional upgrade methods such as using Oracle database export and import.

The chapter delivered upgrade from Oracle Database 11g to Oracle Database 12c using DBUA Graphical User Interface (GUI) as it's the recommended technique for database upgrade. Other database upgrade techniques are DBUA on silent mode and manual upgrade using SQL commands. The DBUA silent mode provides similar level of control as the DBUA GUI. However; manual upgrade is executed step-by-step which promotes more control throughout the upgrade process.

It's critical to verify the data before the application transition occurs. Using Oracle GoldenGate Veridata delivers the best result in terms of accuracy and performance. Because all tables within list of schemas are included, the compare pairs are best created using Oracle GoldenGate wildcard pattern for quickly verifying the data.

The next chapter covers Oracle GoldenGate heterogeneity. In particular, the source database is on Oracle Database 12c and the target database is on Oracle MySQL. This has positioned Oracle GoldenGate distinctively for data replication.