

2016

Kumar Gorrela, Nagesh

AstraZeneca GTC

1/10/2016



12C Administration

**Topics:**

**Section 1: Introduction to 12C Administration in AZ env.**

**Practical sessions:**

**Section 2: How to configure 12C CDB and PDB databases**

**Section 3: Managing 12C CDB and PDB databases**

**Section 4: Online Move datafile in Oracle Database 12c**

**Section 5: Performing Point-In-Time Recovery for a Pluggable Database**

**Section 6: RMAN Recover TABLE**

### **Oracle (RDBMS) Database Version 12c.**

Oracle database 12c Release 1 (12.1.0.2.0) is the latest version of database released by Oracle Corporation. Oracle 12c provides new features to the Oracle database engine like- Multitenant Architecture, Automatic data optimization, Data Guard Far sync, Flex ASM, Transaction Guard, Privilege analysis etc. Currently the Oracle 12c database is supported on the following OS in AstraZeneca. Configuration of Multitenant feature will be done on need basis.

* AIX
* Linux

|  |  |  |
| --- | --- | --- |
| Supported Platforms | | |
| Vendor | Operating System | Release/Version |
| IBM | AIX | IBM AIX 7.1 (64-bit) on IBM PowerPC; 6.1? refer to platform standard – Steve to check |
| Red Hat | Linux | RHEL 7.0 (64-bit) on x86-64 6.0 |
| Oracle | Linux | Oracle Linux 6.5 (64-bit) on x86-64 |
| Oracle Linux 7.0 (64-bit) on x86-64 |

### **Oracle Enterprise Manager**

Oracle Enterprise Manager (OEM) is a separate Oracle product that combines a graphical console, agents, common services, and tools to provide an integrated and comprehensive systems management platform for managing Oracle products.

OEM will be deployed as a separate instance of an Oracle Database supporting multiple Oracle application databases. It should have its own database and Oracle environment, on a separate network from the core Oracle infrastructure together with its own Oracle Data Guard database, so that it can be used by administrators globally.

**OEM 12c Grid Control** is the Enterprise version in that you can monitor different targets from different operating systems at the same time. These include Application Servers, Listeners, Operation Systems, and Non Oracle Database Systems using plugins (from 11gR2).

### **Oracle Real application Cluster (RAC)**

Oracle Real Application Cluster (RAC) is a well-known product among Oracle’s solutions to maintain high availability. Oracle RAC allows the work load to be shared among all the cluster nodes, with N-1 tolerance configuration in case of node failures, where N is the total number of nodes. The new 12c version incorporates two properties called “Flex ASM” and “Flex Cluster” that gives support to the demand requirements on Cloud Computing oriented environments. Prior to Oracle 12c, for a database instance to use ASM it is expected that the ASM instance must be up and running on all nodes before the database instance is brought up. Failure of ASM instance to come-up means that database instance using ASM at the storage level cannot be brought up. This literally means that the database instance is not accessible immaterial of the technologies put in use i.e. RAC, ASM and Shared Storage.

Architect team approved AIX as a standard platform for future RAC builds and exiting RAC databases will be migrated to above mentioned platforms.

### **Oracle Data guard**

Oracle Data Guard is the most effective and comprehensive data availability, data protection and disaster recovery solution for enterprise databases. Data Guard maintains these standby databases as synchronized copies of the production database. It can be a logical or a physical standby database. These standby databases can be located at remote disaster recovery sites thousands of miles away from the production data center(WAN), or they may be located in the same city, same campus, or even in the same building (LAN). If the production database becomes unavailable because of a planned or an unplanned outage, Data Guard can switch any standby database to the production role, thus minimizing the downtime associated with the outage, and preventing any data loss. It can address both High Availability and Disaster Recovery requirements, and is the ideal complement to Oracle Real Application Clusters (Oracle RAC). Data Guard has the requisite knowledge of the Oracle database to reliably protect a standby database from corruptions that attempt to propagate from a primary database. It enables all standby databases, both physical and logical, to be used for productive purposes while in standby role.

In 12c, Data Guard is set up at the Container level and not the individual Pluggable database level. Failover and switch back are simplifier in 12c compare with old version.

Also, we can use Active data guard database for reporting purpose and the corrupted block in production database recovered automatically from the standby database. Oracle Streams

### **Oracle Multitenant Architecture**

The multitenant architecture enables an Oracle database to function as a multitenant container database (CDB) that includes zero, one, or many customer-created pluggable databases (PDBs). A PDB is a portable collection of schemas, schema objects, and non-schema objects that appears to an Oracle Net client as a non-CDB.

* Containers in a CDB  
    
  A container is either a PDB or the root container (also called the root). The root is a collection of schemas, schema objects, and no schema objects to which all PDBs belong

Every CDB has the following containers:  
  
\*    exactly one root  
  
    The root stores Oracle-supplied metadata and common users. An example of metadata is the source code for Oracle-supplied PL/SQL. A common user is a database user known in every container. The root container is named CDB$ROOT.  
  
\*    Exactly one seed PDB  
  
    The seed PDB is a system-supplied template that the CDB can use to create new PDBs.   
    The seed PDB is named PDB$SEED. You cannot add or modify objects in PDB$SEED.  
  
 \*   Zero or more user-created PDBs  
  
   A PDB is a user-created entity that contains the data and code required for a specific set of features. For example, a PDB can support a specific application, such as a human resources or sales application. No PDBs exist at creation of the CDB. DBA add PDBs based on your business requirements.

### **Oracle SINGLEtenant Architecture**

New database builds and existing databases upgrade to 12C (12.1.0.2) will be with Singletenent configuration (one PDB per CDB) to enable all multitenant features without added licensing cost. This will be big help for planned consolidation and future readiness of cloud platform.

### **Oracle VM**

Oracle VM is the server virtualization offering from Oracle Corporation. Oracle VM Server for x86 incorporates the free and open-source Xen hypervisor technology, supports Windows, Linux, and Oracle Solaris guests and includes an integrated Web based management console.

Oracle VM is a platform that provides a fully equipped environment to better leverage the benefits of virtualization technology. Oracle VM enables you to deploy operating systems and application software within a supported virtualization environment. The components of Oracle VM are:

* Oracle VM Manager: Provides the user interface, which is a standard ADF (Application Development Framework) web application, to manage Oracle VM Servers, virtual machines, and resources. Use Oracle VM Manager to:
  + Create virtual machines from installation media or from a virtual machine template
  + Delete virtual machines
  + Power off virtual machines
  + Import virtual machines
  + Deploy and clone virtual machines
  + Perform live migration of virtual machines
  + Import and manage ISOs
  + Create and manage virtual machine templates
  + Create and manage shared virtual disks
* Oracle VM Server: A self-contained virtualization environment designed to provide a lightweight, secure, server-based platform to run virtual machines. Oracle VM Server is based upon an updated version of the underlying Xen hypervisor technology, and includes Oracle VM Agent.
* Oracle VM Agent: Installed with Oracle VM Server. It communicates with Oracle VM Manager to manage virtual machines.

### **Oracle Exadata Machine:**

Oracle Exadata Machine is an easy to deploy solution to host oracle databases which delivers very high performance. It works for OLTP as well as OLAP systems. It primarily composes of Real Application Clusters and Exadata storage and runs on the latest version of Oracle 11gR2.The intelligent storage brings in lot of services like database aware storage which

Can offload the database load on to server storage which is completely transparent to SQL and database applications. Apart from this it brings in various other features like Exadata Smart Flash Cache, Hybrid columnar compression etc.

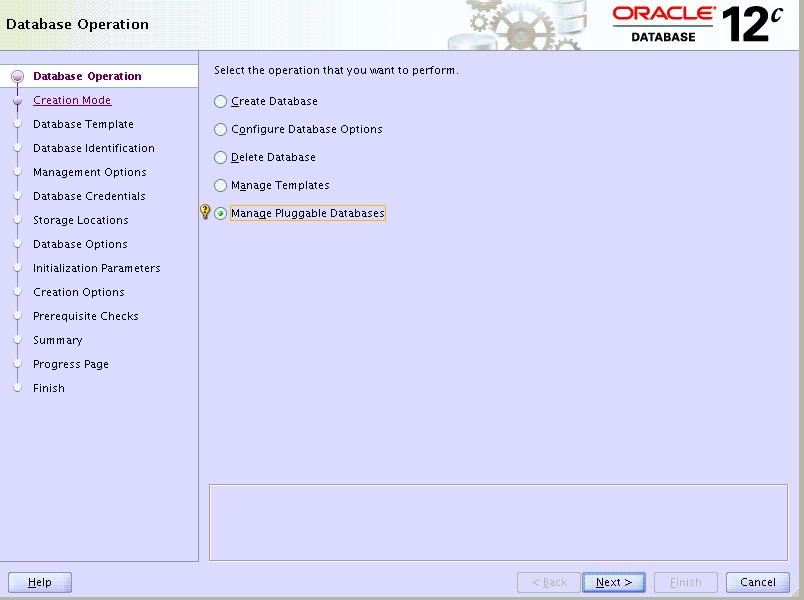
The Oracle Exadata machine mainly comprises of RAC clusters, Infiniband interconnect and Exadata storage. The Infiniband interconnect acts as an interconnect between the servers and storage. Each database server and exadata cell has dual port Quad Data Rate (QDR) Infiniband connectivity for high availability. It provides a very high level of data transmission which is much higher than the traditional server networks. This high bandwidth not only rules out any network issues but also acts as a high performance cluster interconnect for Oracle RAC.As the demand increases the capacity can be increased by increasing the storage and database servers.

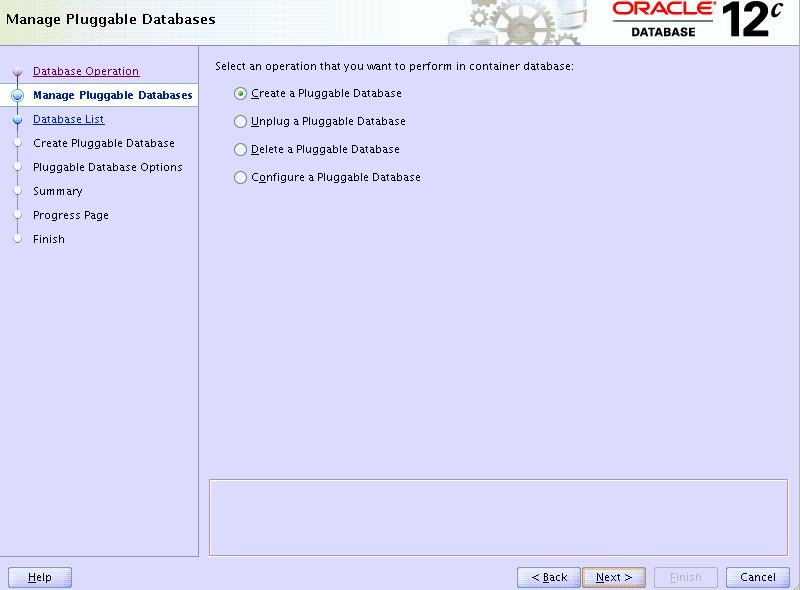
One of the main differences on the traditional storage and exadata storage is that with exadata it returns only the rows and columns that satisfy the database query are returned instead of the whole table being queried.

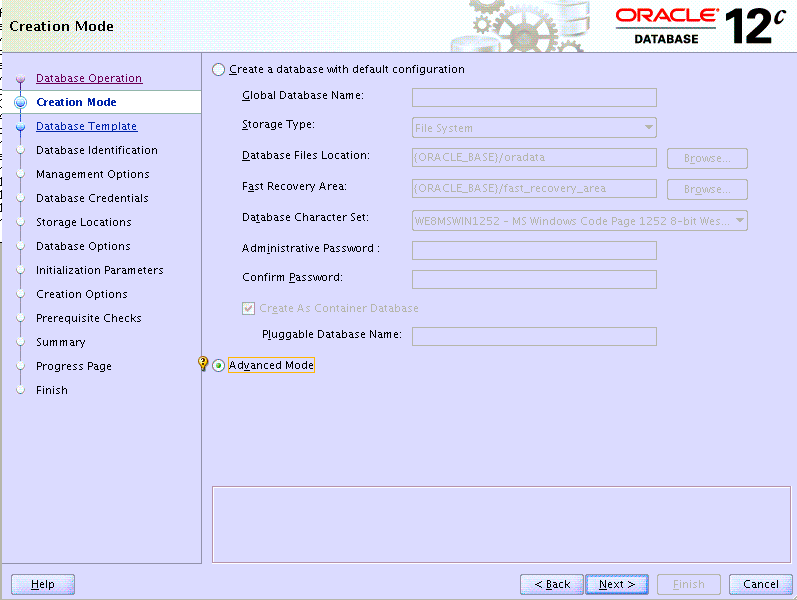
# **Section 2: How to configure 12C CTDB and PDB databases.**

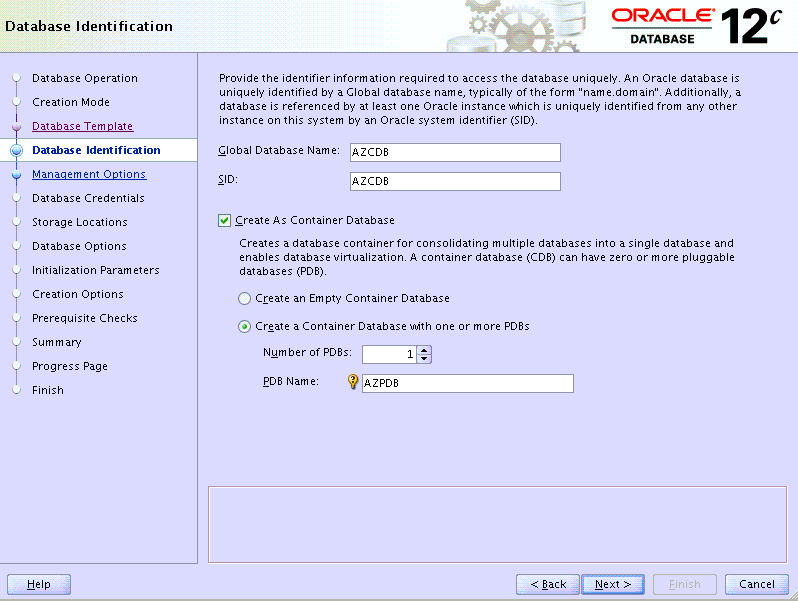
Steps:

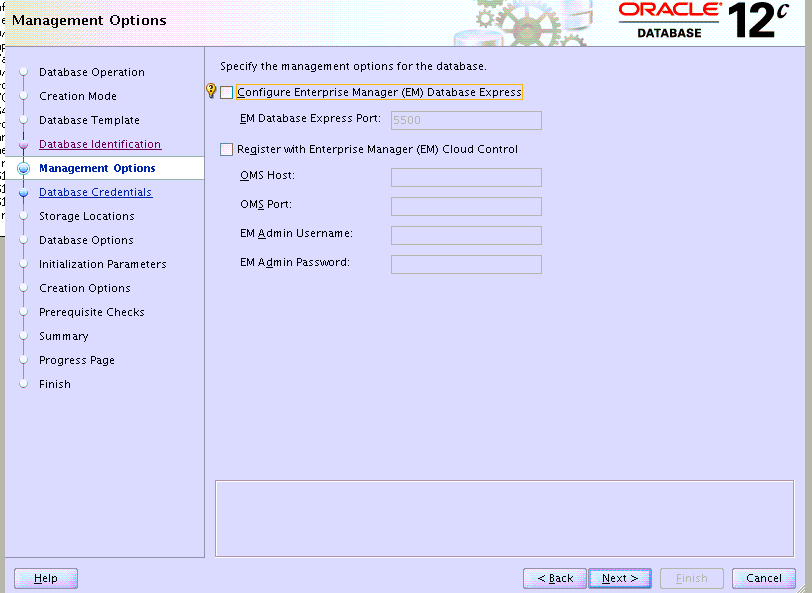
1. Download the 12C software from Metalink.
2. Follow the Read me steps and install the 12C software.
3. After install the software, run the DBCA to configure CDB and PDB databases.

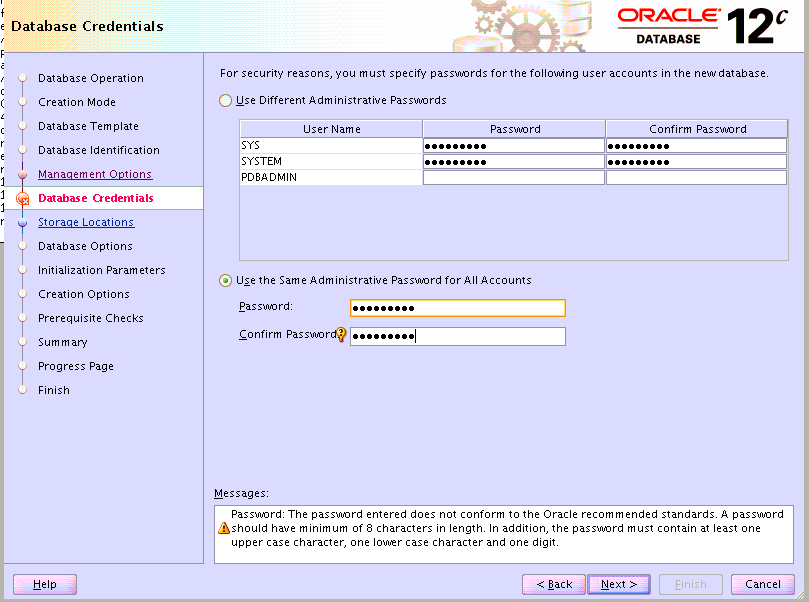


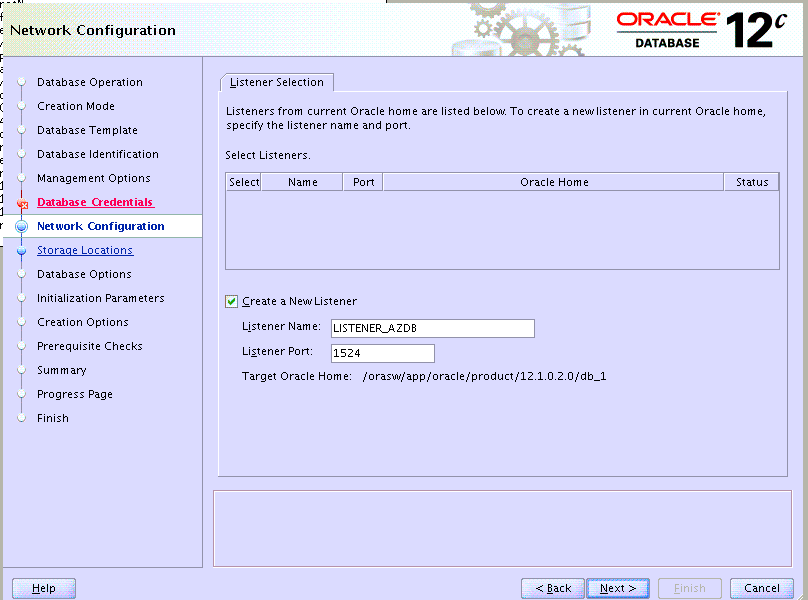


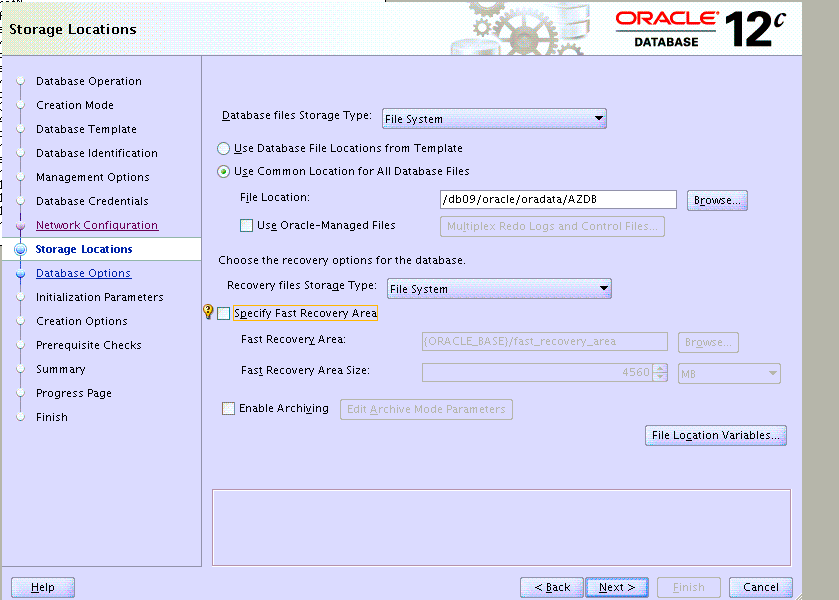


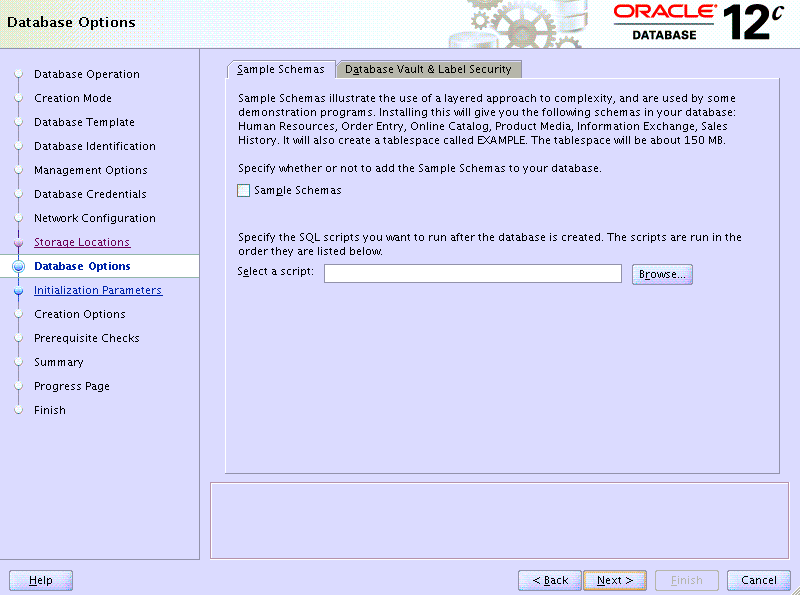


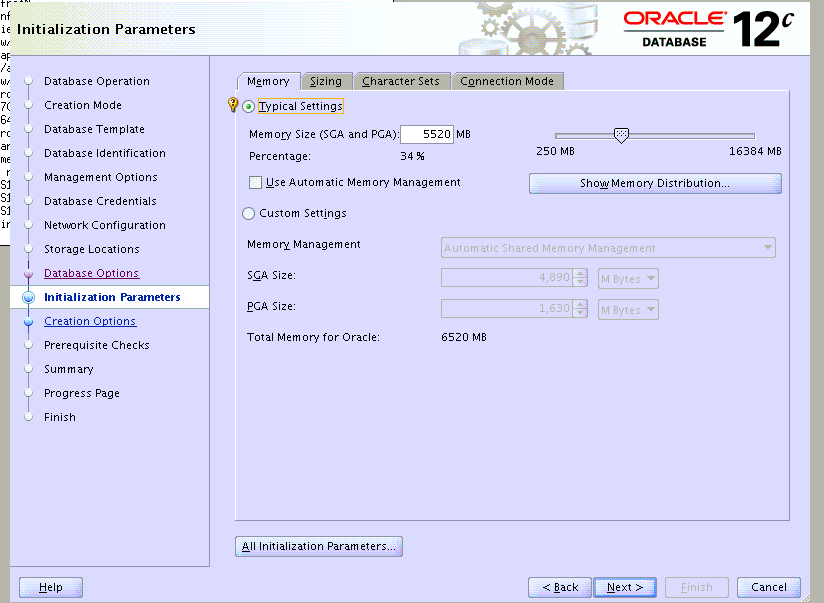


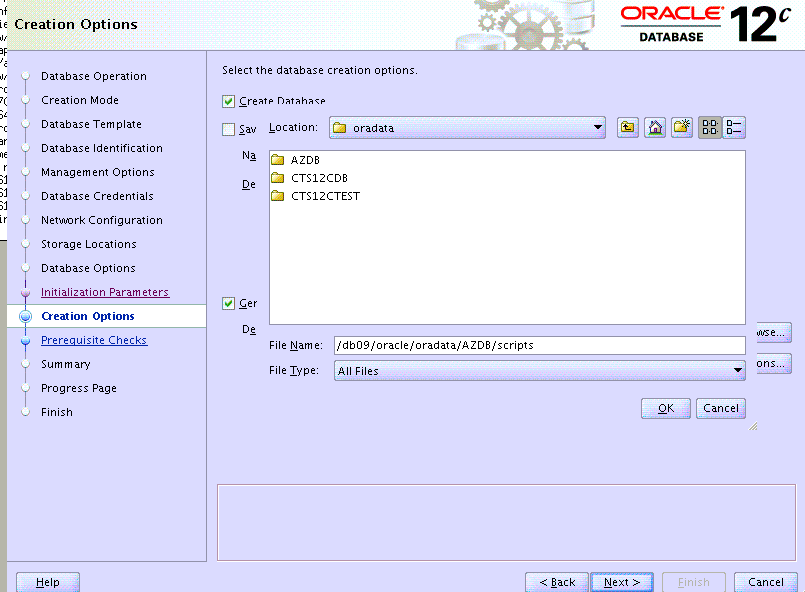


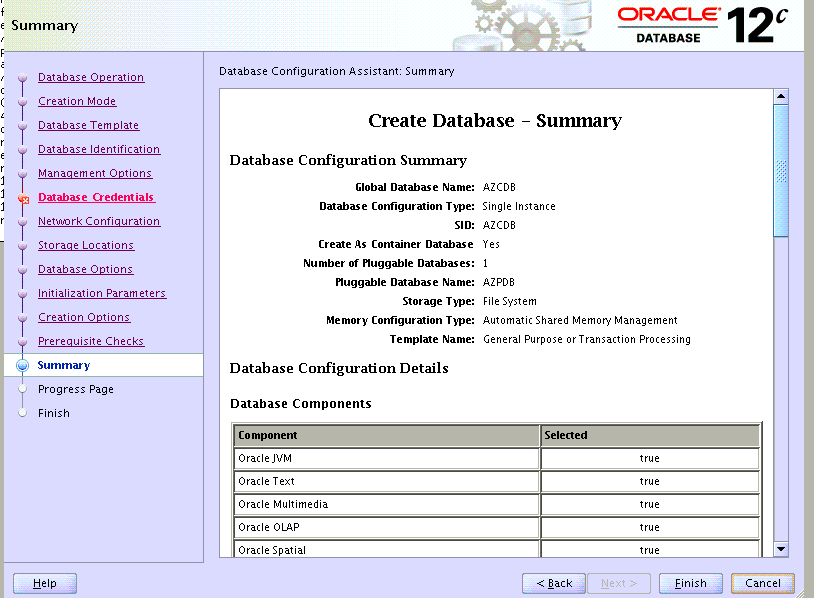


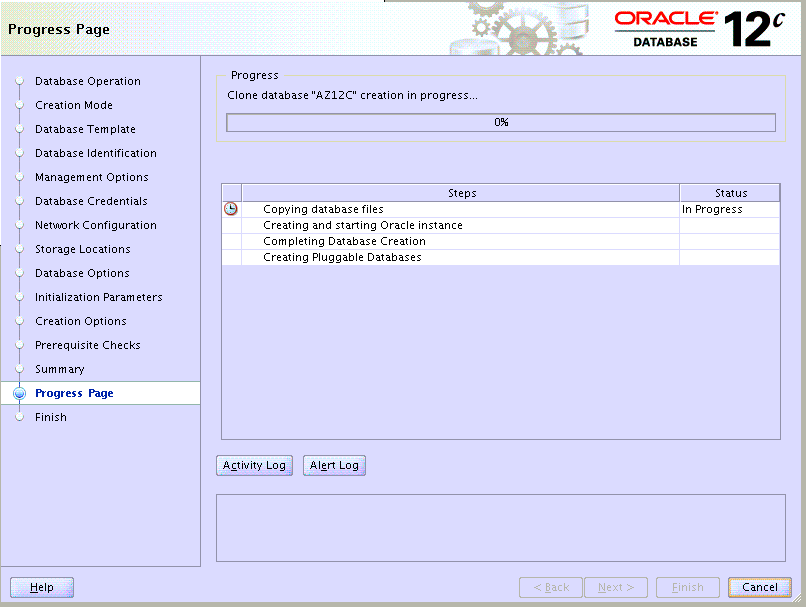


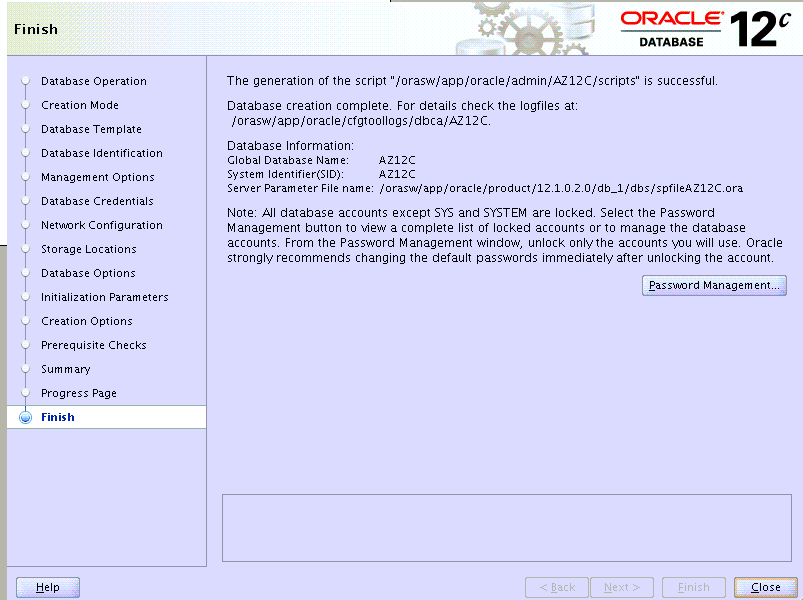








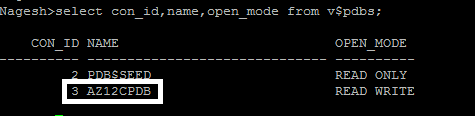




* Successfully configured the PDB and CDB

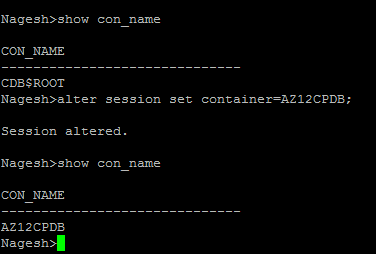
# **Section 3: Managing 12C CDB and PDB databases**

How to connect Pluggable database and shutting down the pluggable database?

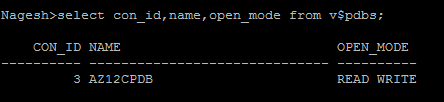


In our scenario our pluggable database name is AZ12CPDB.

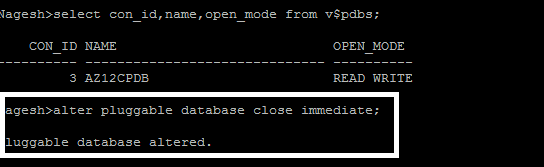
Connect to Pluggable database from container.



1. Check the status of pluggable database.

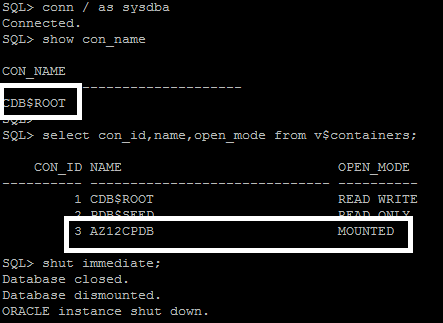


1. Shutting down the pluggable database.

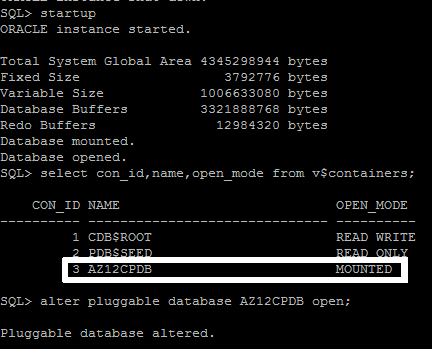


1. Shutdown container database. (Please make sure that, before shutting down container database, all pluggable databases should be shutdown (Mount status).

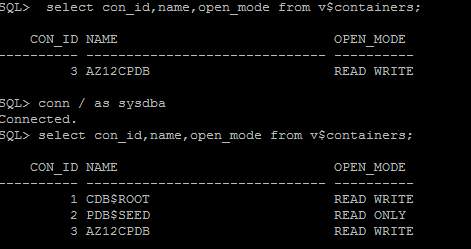
* Connect to Root DB and shutdown the container.



1. Starting Container and pluggable database:



Note: If you see the above, pluggable database is in mount status after restart the container database. We have to start the pluggable database manually in 12.1.0.1 version.



**Pluggable Database (PDB) Automatic Startup**

The 12.1.0.2 patchset has introduced the ability to preserve the startup state of PDBs, so you probably shouldn't be implementing a trigger in the manner discussed in this section.

Prior to 12.1.0.2, when the CDB is started, all PDBs remain in mounted mode. There is no default mechanism to automatically start them when the CDB is started. The way to achieve this is to use a system trigger on the CDB to start some or all of the PDBs.

CREATE OR REPLACE TRIGGER open\_pdbs

AFTER STARTUP ON DATABASE

BEGIN

EXECUTE IMMEDIATE 'ALTER PLUGGABLE DATABASE ALL OPEN';

END open\_pdbs;

/

You can customise the trigger if you don't want all of your PDBs to start.

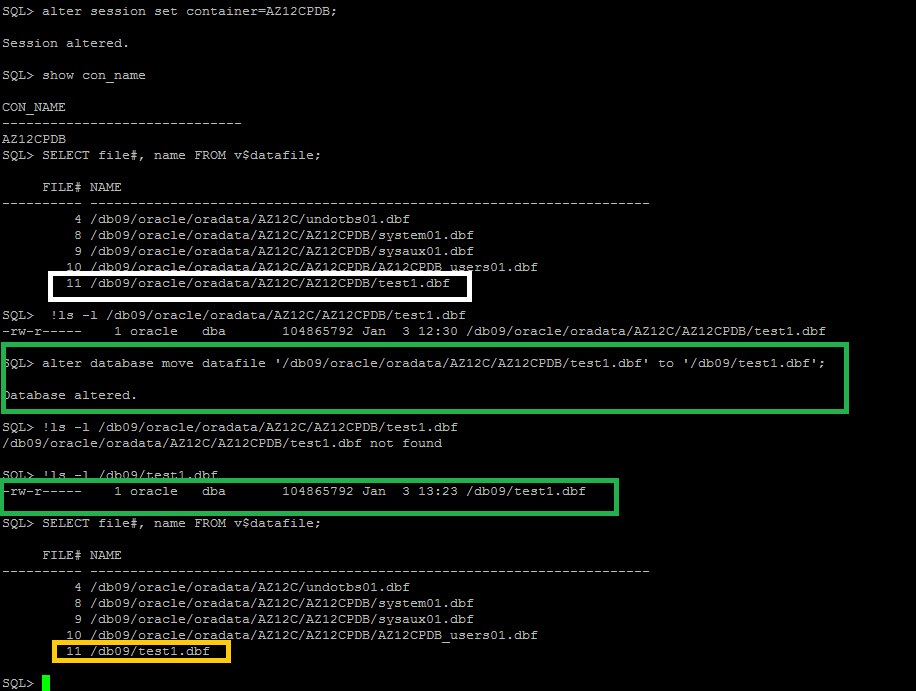
# **Section 4: Online Move Datafile in Oracle Database 12c**

Prior to Oracle 12c, moving datafiles has always been an offline task. There were certain techniques you could employ to minimize that downtime, but you couldn't remove it completely. Oracle 12c includes an enhancement to the ALTER DATABASE command to allow datafiles to be moved online.

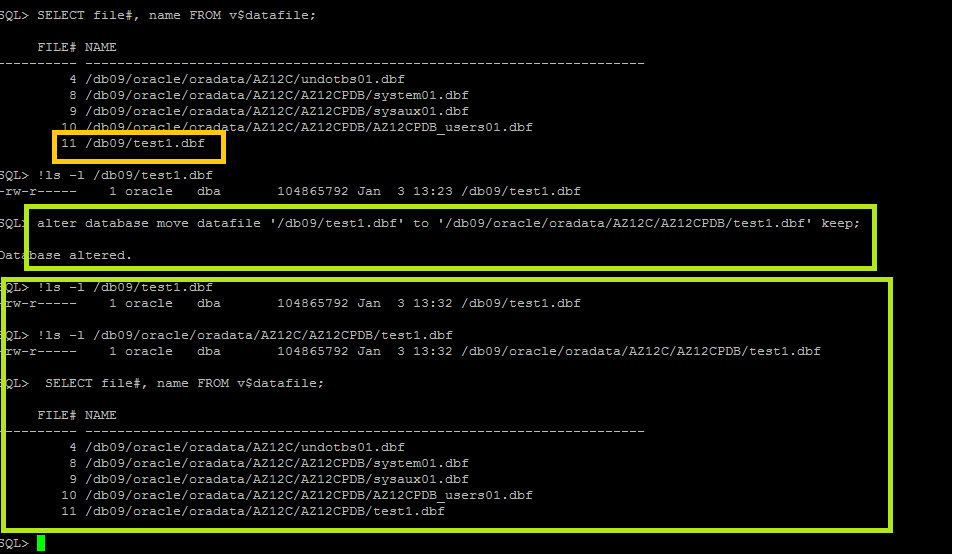
Note: The container database (CDB) cannot move files that belong to a pluggable database, so we have to connect to the pluggable databases while moving the datafiles.

Steps:

In our first example, we are moving the pluggable database datafile test1 from one location to another location.



Second example with **KEEP** option: we can move the datafile test1 to same old location and keeps the existing datafile in same location for safer side.



Note: Temp files we cannot move.

Keep option will ignored for OMF datafile move.

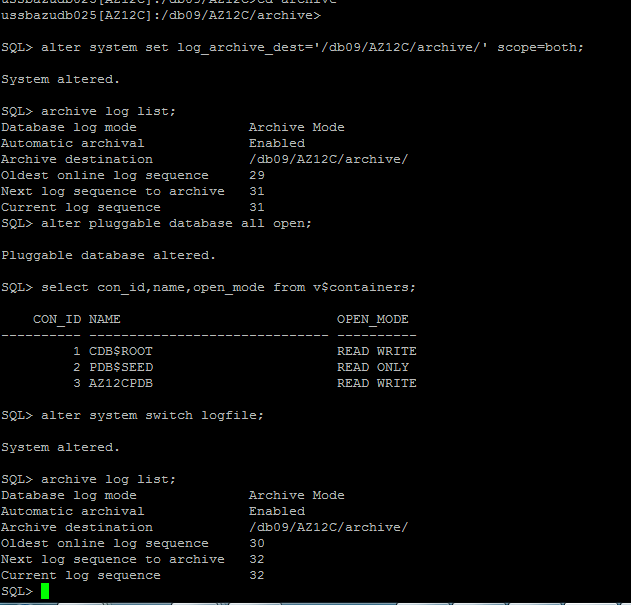
# **Section 5: Perofrming point-in-time recovery for a pluggable database**

DB name: AZ12C (Container DB)

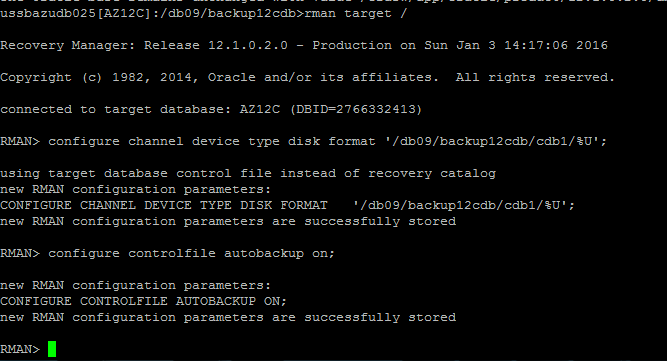
Pluggable DB: AZ12CPDB

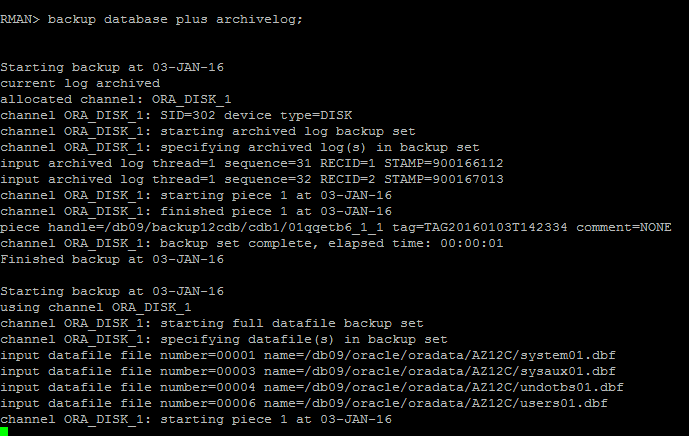
Backup of CDB:

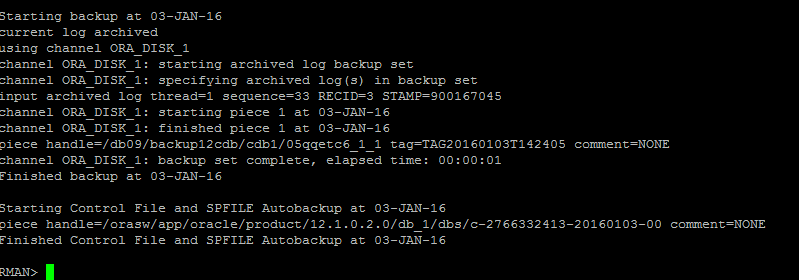
1. Create backup directory to take RMAN backup.
2. Navigate to /db09 or wherever you can find enough available space to store the backup files.
3. **cd /db09**
4. Create a subdirectory called backup12cdb under /db09.
5. **mkdir backup12cdb**
6. Create a subdirectory called cdb1 under /db09/backup12cdb
7. **mkdir cdb1**
8. **Keep the database in Archive log mode:**



**9)set the env and connect to the rman prompt,allocate the channel to disk,controlfile autobackup on and initiate the database backup plus archivelog.**





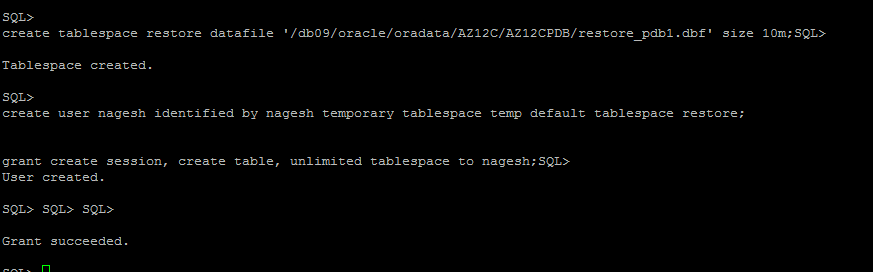


SQL>

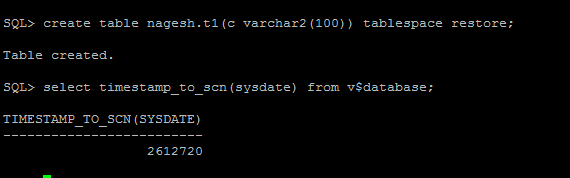
create tablespace restore datafile '/db09/oracle/oradata/AZ12C/AZ12CPDB/restore\_pdb1.dbf' size 10m;SQL>

Tablespace created.

* Create user and give the privileges.



* Create the table in the schema and note the current SCN.



SQL> select timestamp\_to\_scn(sysdate) from v$database;

TIMESTAMP\_TO\_SCN(SYSDATE)

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

2612720

Insert data in the table.

**begin  
 for i in 1.. 10000 loop  
    insert into nagesh.t1 values('howru');  
 end loop;  
 commit;  
end;  
/**

**SQL> select count(\*) from nagesh.t1;**

**COUNT(\*)**

**----------**

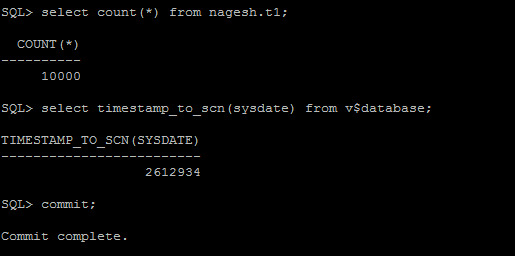
**10000**

**SQL> select timestamp\_to\_scn(sysdate) from v$database;**

**TIMESTAMP\_TO\_SCN(SYSDATE)**

**-------------------------**

**2612934**



**run {**

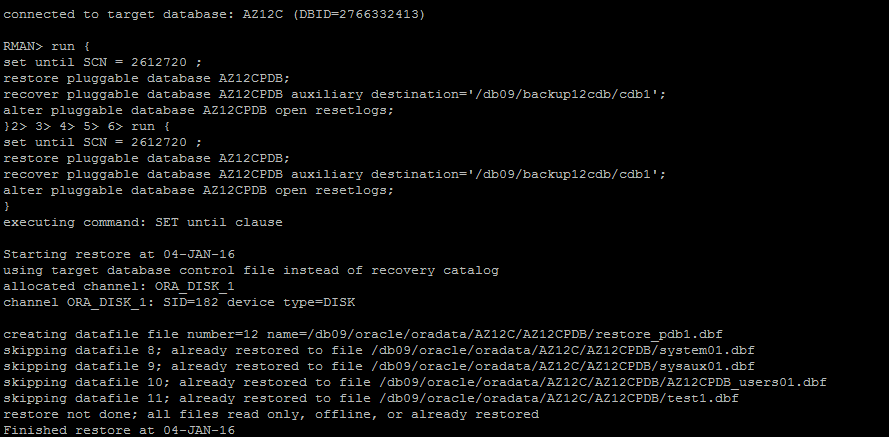
**set until SCN = 2612720 ;**

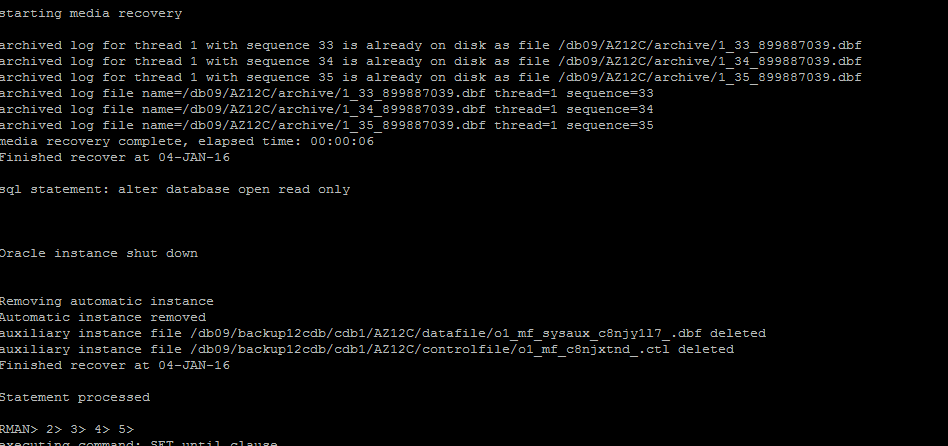
**restore pluggable database AZ12CPDB;**

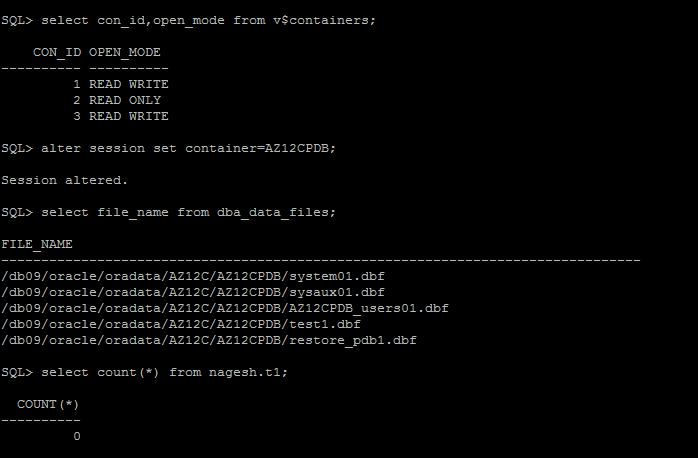
**recover pluggable database AZ12CPDB auxiliary destination='/db09/backup12cdb/cdb1';**

**alter pluggable database AZ12CPDB open resetlogs;**

**}**







Successfully restored the table to SCN **2612720 .**

# **Section 6: RMAN RECOVER TABLE**

One of the good new features in Oracle 12c is the ability to **restore a single table or a single partition of a partitioned table from an RMAN backup via the RECOVER TABLE**command. Prior to 12c restoring a table was a long drawn out and difficult affair.

So when may we use this fearure?

There is TSPITR ( Tablespace point-in-time recovery) but what if we only want to restore a single table or subset of tables and the tablespace has a large number of tables.

Tables have been logically corrupted or records wrongly purged and we cannot use the FLASHBACK TABLE feature because we do not have enough undo data available to go back to the required point in time.

FLASHBACK DATABASE has not been turned on and remember even if it was turned on, if we are going to use flashback database just to recover some tables, the entire database is being put to a point in time in the past which is not very desirable.

So what does the RMAN RECOVER TABLE command do behind the scenes?

It creates an auxiliary database or instance which is used to recover the tables to a specific point in time. This database will contain a few system related data files like SYSTEM, SYSAUX, UNDO and  data files belonging to the tablespace containing the tables we are looking to restore.

In this example it restores SYSTEM, SYSAUX, UNDO tablespaces for CDB and SYSTEM and SYSAUX and EXAMPLE tablespace for PDB

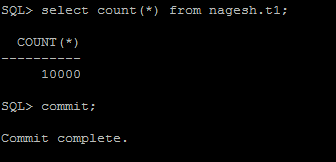
Then it creates a Data Pump export dump file which will contain the recovered table or partitions of tables.

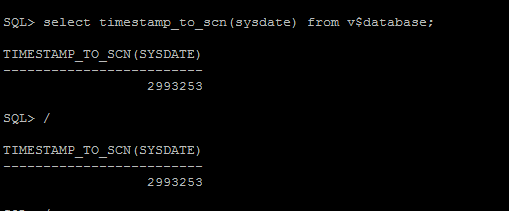
It will then import the data into the target database using Data Pump Import.

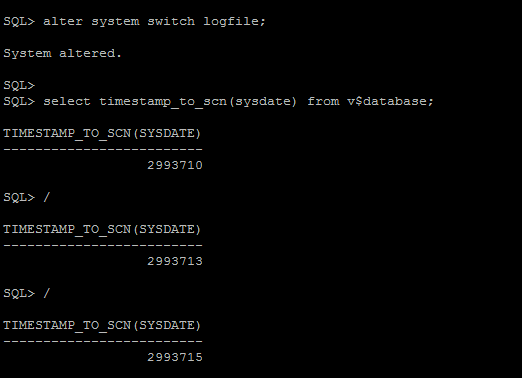
Finally it will remove the temporary auxiliary instance.

Let us see an example where we drop the NAGESH.T1 table and test the restore and recovery using the new 12c RMAN RECOVER TABLE feature

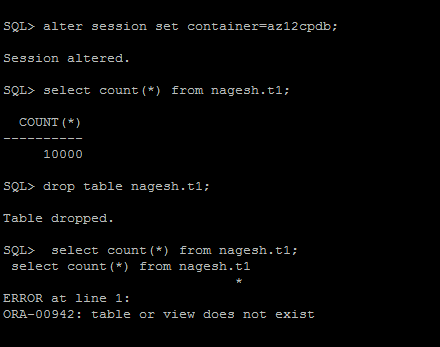
* Insert some data into the table and commit the records.







* **Drop the table and restore it to above SCN 2993715**



RMAN> run

2> {

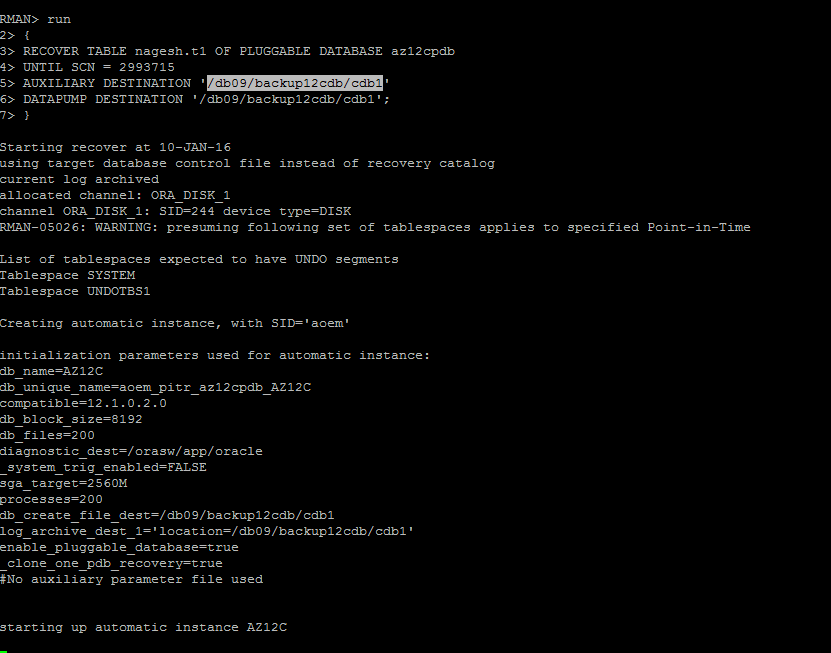
3> RECOVER TABLE nagesh.t1 OF PLUGGABLE DATABASE az12cpdb

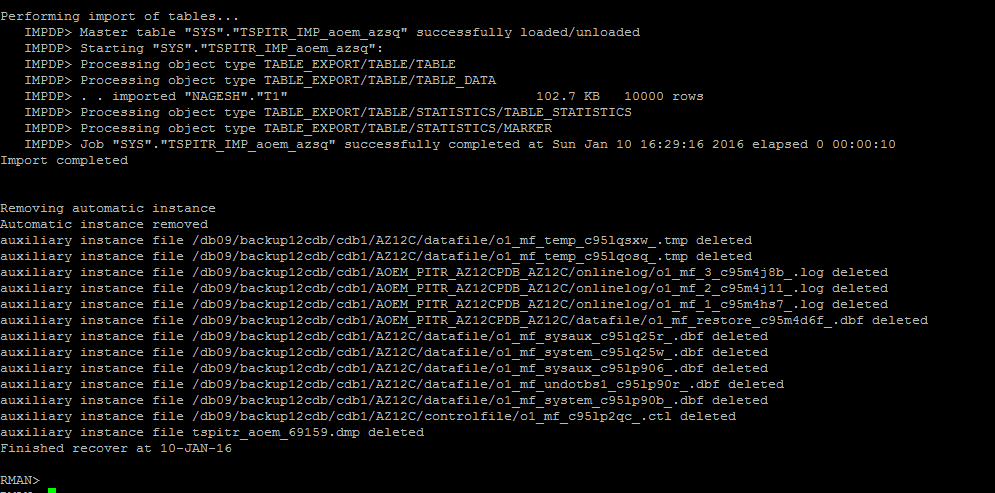
4> UNTIL SCN = 2993715

5> AUXILIARY DESTINATION '/db09/backup12cdb/cdb1'

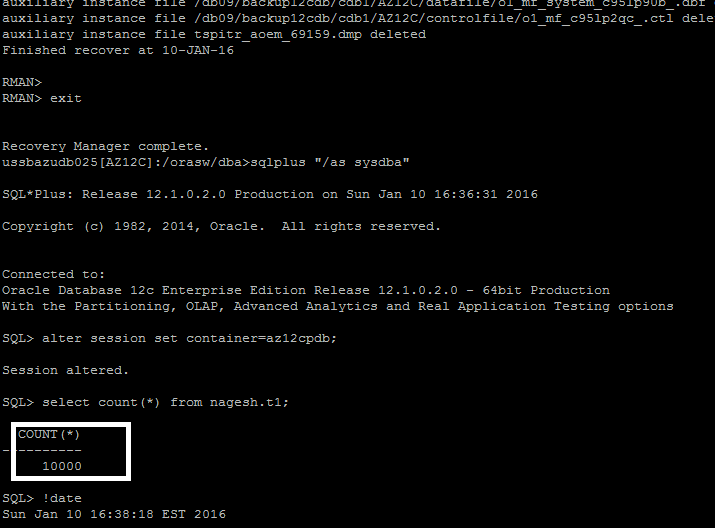
6> DATAPUMP DESTINATION '/db09/backup12cdb/cdb1';

7> }









Finally Table has been restored successfully. ------------------ END------------------------