Practice 6: Creating PDB by Plugging in an Unplugged PDB

Practice Overview

In this practice you will perform the following:

- Create a PDB by plugging in an unplugged PDB using xml file.
- Create a PDB by plugging in an unplugged PDB using the archived pdb file.

Practice Assumptions

- You have the `srv1` and its CDB database up and running.

A. Create a new PDB by Plugging in an Unplugged PDB

In this section you will create a new PDB, named `PDB5`, by unplugging `PDB4` then plugging it in as `PDB5`. `PDB4` will be dropped.

1. In SQL*Plus, connect to the CDB root as `sysdba`.

```
sqlplus / as sysdba
```

2. Retrieve list of the `PDB4` datafiles. This step is just to understand where the datafiles are located.

```
set linesize 100
col name format a100
SELECT NAME
FROM V$DATAFILE WHERE CON_ID = (SELECT CON_ID FROM V$PDBS WHERE NAME='PDB4');
```

3. Retrieve the `GUID` of `PDB4` and take a note of it.

```
SELECT GUID FROM V$PDBS WHERE NAME='PDB4';
```

4. Close `PDB4`.

```
ALTER PLUGGABLE DATABASE pdb4 CLOSE;
```

5. Unplug `PDB4`. Save the XML file in `oracle` home directory.

```
ALTER PLUGGABLE DATABASE pdb4
UNPLUG INTO '/home/oracle/pdb4.xml';
```

6. Open the generated XML file.

   The XML file contains information about the physical structure of the PDB; for example, the tablespaces and the datafiles.

```
host vi /home/oracle/pdb4.xml
```

7. Verify that the unplugged PDB is still part of the CDB catalog.

   When you unplug a PDB, records will be saved in the data files to indicate that the PDB was properly and successfully unplugged.

```
SELECT PDB_NAME, STATUS FROM CDB_PDBS WHERE PDB_NAME IN ('PDB4');
```

8. Drop PDB4 and keep its data files.

```
DROP PLUGGABLE DATABASE pdb4 KEEP DATAFILES;
```

9. Verify that the unplugged PDB is **not** now part of the CDB catalog.

```
SELECT PDB_NAME, STATUS FROM CDB_PDBS WHERE PDB_NAME IN ('PDB4');
```

10. Check the compatibility of the unplugged PDB with the CDB.

In our scenario this step is actually not needed because you are going to plug the PDB into the same CDB. But if you were plugging the PDB into different CDB, you should check the compatibility of the unplugged PDB with the new host CDB.

If the output of the code below indicates that the PDB is not compatible, examine the `PDB_PLUG_IN_VIOLATIONS` view to see why it is not compatible.

```
set serveroutput on
DECLARE
   compatible BOOLEAN := FALSE;
BEGIN
   compatible := DBMS_PDB.CHECK_PLUG_COMPATIBILITY(
        PDB_DESCR_FILE => '/home/oracle/pdb4.xml');
   if compatible then
      DBMS_OUTPUT.PUT_LINE('It is compatible');
   else
      DBMS_OUTPUT.PUT_LINE('It is NOT compatible');
   end if;
END;
/
```

11. Create the new pdb using the `NOCOPY` method.

If you need to plug a PDB with a new `GUID` (different from the unplugged PDB), use the "`AS CLONE..MOVE`" method.

```
CREATE PLUGGABLE DATABASE pdb5 USING '/home/oracle/pdb4.xml'
NOCOPY TEMPFILE REUSE;
```

12. Verify the status and open mode of the plugged PDB.

```
col pdb_name format a5

SELECT PDB_NAME, STATUS FROM CDB_PDBS WHERE PDB_NAME='PDB5';
SELECT OPEN_MODE FROM V$PDBS WHERE NAME='PDB5';
```

13. Open the pdb in read/write mode.

```
ALTER PLUGGABLE DATABASE pdb5 OPEN;
```

14. Test connecting to the plugged-in PDB

```
connect sys/oracle@srv1:1521/PDB5.localdomain as SYSDBA
SHOW CON_NAME
```

15. Get the `GUID` of the plugged PDB. Compare it to the `GUID` of the unplugged PDB.

```
SELECT GUID FROM V$PDBS WHERE NAME='PDB5';
```

16. Remove the xml file. It is no longer needed.

```
host rm /home/oracle/pdb4.xml
```

B. Create a new PDB by Plugging in an Unplugged PDB using PDB Archive file

In this section you will create a new PDB, named `PDB6`, by unplugging `PDB5` using the PDB archive file. This option is made available in Oracle release 12.2.

17. Get the total size of the objects within `PDB5`.

```
ALTER SESSION SET CONTAINER=pdb5 ;

SELECT ROUND(SUM(BYTES)/1024/1024,2) SPACE_IN_MB
FROM DBA_SEGMENTS;
```

18. Close `PDB5`

```
ALTER SESSION SET CONTAINER=cdb$root;
ALTER PLUGGABLE DATABASE pdb5 CLOSE;
```

19. Unplug `PDB5` into a PDB archive file.

```
ALTER PLUGGABLE DATABASE pdb5 UNPLUG INTO '/home/oracle/pdb5.pdb';
```

20. Check out the size of the generated PDB file.

The size of the generated file should be a fraction of the total size of the `PDB5` objects. The generated file is a compressed file.

```
host ls -alh /home/oracle/pdb5.pdb
```

21. Try creating a new PDB (named `PDB6`) from the PDB archive file using `COPY` option.

You should receive the following error. This is because source PDB has not been dropped.

```
ORA-65122: Pluggable database GUID conflicts with the GUID of an existing
container.
```

```
CREATE PLUGGABLE DATABASE pdb6 USING '/home/oracle/pdb5.pdb' COPY;
```

22. Try creating the `PDB6` using `AS CLONE` option.

`AS CLONE` clause makes Oracle Database generate a new `GUID` for the new PDB.

```
CREATE PLUGGABLE DATABASE pdb6 AS CLONE USING '/home/oracle/pdb5.pdb' COPY;
```

23. Open the new PDB

```
ALTER PLUGGABLE DATABASE pdb6 OPEN;
```

24. Check out the status and open mode of the unplugged and the plugged PDBs.

```
col pdb_name format a5

SELECT PDB_NAME, STATUS FROM CDB_PDBS WHERE PDB_NAME IN ('PDB5','PDB6');
SELECT OPEN_MODE FROM V$PDBS WHERE NAME IN ('PDB5','PDB6');
```

25. To save disk space, let's drop `PDB5` and `PDB6`.

```
DROP PLUGGABLE DATABASE pdb5 INCLUDING DATAFILES;
host rm /home/oracle/pdb5.pdb

ALTER PLUGGABLE DATABASE pdb6 CLOSE;
DROP PLUGGABLE DATABASE pdb6 INCLUDING DATAFILES;
```

26. List the PDBs that are currently available. You should have `PDB1`, `PDB2`, and `PDB3`.

```
SELECT NAME FROM V$PDBS ORDER BY 1;
```

Summary

You can create PDBs by plugging in an unplugged PDB. You can export the definitions of the source PDB into xml file and copy it together with the datafiles to the target system.

Alternatively, you can include all the datafiles into a compressed archive file and copy the file to the target system.