Practice

Creating Pluggable Databases (PDBs)

Practice Target

In this practice, you will create PDB databases from the seed using two methods.

In this practice you will perform the following:

- Create a PDB using SQL*Plus
- Create a PDB using the `dbca`
- Perform basic exploring to the created PDBs.

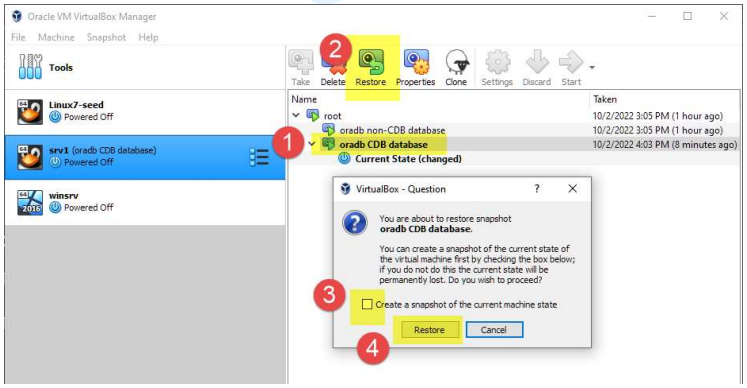
Practice Assumption

The practice assumes that `srv1` is restored from its **CDB** snapshot and is up and running.

To restore the vm from its CDB snapshot:

a. Shut `srv1` down (if it is already running).

b. In Oracle VirtualBox, select the "**oradb CDB database**" snapshot then click on **Restore** button. Unmark the checkbox "**Create snapshot of the current machine stat.**", then click on **Restore** button.



c. Start the vm.

In the assumption of nearly every future practice in this course, it will state which snapshot to be used in the practice: the non-CDB or the CDB. In either way, you just perform the same steps for the needed snapshot.

Creating a PDB from the Seed using SQL*Plus

In the following steps, you will create a PDB from the seed container using the SQL statement `CREATE PLUGGABLE DATABASE`.

1. Create a Putty session to `srv1` and login as `oracle`

2. Login to the CDB as sysdba in SQL*Plus

```
sqlplus / as sysdba
```

3. Verify that the OMF is configured.

When `DB_CREATE_FILE_DEST` is set to a non-Null value, we say the OMF is configured. This parameter takes a directory value. It defines where the datafiles (of the tablespaces) will be by default created.

```
show parameter DB_CREATE_FILE_DEST
```

4. Issue the following command to create a PDB from the seed container.

This statement assumes the OMF is configured, you do not want to specify a location for the datafiles different from the default location and you do not want to specify the default tablespace.

Note: In the code below, `TIMING` is a SQL*Plus variable. When it is set to `ON`, SQL*Plus displays the time period elapsed by every statement it runs.

```
set timing on

CREATE PLUGGABLE DATABASE PDB2
 ADMIN USER pdb2admin IDENTIFIED BY ABcd##1234
 STORAGE (MAXSIZE 2G);

set timing off
```

5. Check the open mode of the created pluggable database.

The newly created PDB is opened in `MOUNT` mode. Users cannot connect to it at this stage.

```
SELECT OPEN_MODE FROM V$PDBS WHERE NAME='PDB2';
```

6. Open `PDB2` in read/write mode.

After a PDB is created, its status is not open. We should manually open it.

```
ALTER PLUGGABLE DATABASE pdb2 OPEN;
```

7. Test the connection to `PDB2` using the EasyConnect method

You will learn about connection methods to the database later in the course.

```
conn system/ABcd##1234@//srv1:1521/pdb2.localdomain

disconnect
```

Creating a PDB from the Seed using dbca

In this section of the practice, you will create a PDB from the seed using the `dbca` utility.

8. Login to `srv1` and as `oracle` user in the VM window

9. Open a terminal window and execute the `dbca` utility

10. Response to the utility windows as shown in the highlighted areas in following screenshots:

Window	Response
Creation Mode	select **Manage Pluggable database**
Manage Pluggable Databases	Create a **Pluggable database**
Select Database	oradb Username: **sys** Password: **ABcd##1234**
Create Pluggable Database	select Create a new Pluggable database from PDB seed
PDB Identification	Pluggable database Name: pdb3 Administrator user name: pdb3admin Administrator password: ABcd##1234 Confirm administrator password: ABcd##1234
Pluggable Database Options	Mark the checkbox "Create default user tablespace"
Summary	Click on **Finish** button
	Click on **Close** when it is finished.

11. Check the open mode of the created pluggable database.

Observe the PDB created by the `dbca` is automatically opened by the `dbca`.

```
conn / as sysdba
SELECT OPEN_MODE FROM V$PDBS WHERE NAME='PDB3';
```

12. Test the connection to `PDB3` using the EasyConnect method

```
conn system/ABcd##1234@//srv1:1521/pdb3.localdomain

disconn
```

Obtaining Information about the Created PDBs

In this section of the practice, you will retrieve some basic information about the PDBs that you just created. You will learn more details about exploring the PDBs later in the course.

13. In the Putty session, connect to the root container as `SYS` and execute the following query.

```
conn / as sysdba

col name format a10
SELECT NAME, CON_ID, OPEN_MODE, GUID FROM V$PDBS ORDER BY 1;
```

Observe from the output the following:

- The seed PDB name is `PDB$SEED` and it is opened in READ ONLY mode.
- As you will observe in the next step, the `GUID` value is used to create a directory that hosts the PDB datafiles.

14. Execute the following query.

```
set  pagesize 15
col c format a2
col file_name format a65

SELECT SUBSTR(CON_ID,1,1) AS C, FILE_NAME
FROM CDB_DATA_FILES
ORDER BY 1;
```

Observe from the output the following:

- The datafiles of the PDBs were created under the OMF directory using the following format. This format makes it hard to know from the directory structure alone to which PDB specific subdirectory belongs to.

  `<OMF>/<CDB Name>/<PDB GUID>/datafile/<auto-generated file name>`

Extra Practice

If you wish to perform extra practice on this topic, try cloning `PDB3` using SQL statements.

Cleanup

15. Drop the pluggable databases `PDB2` and `PDB3`.

We must close the PDB before we can drop it.

```
conn / as sysdba
ALTER PLUGGABLE DATABASE PDB2 CLOSE;
ALTER PLUGGABLE DATABASE PDB3 CLOSE;
DROP PLUGGABLE DATABASE PDB2 INCLUDING DATAFILES;
DROP PLUGGABLE DATABASE PDB3 INCLUDING DATAFILES;
```

Summary

In this practice you learnt how to create a PDB from the seed container using SQL*Plus and the `dbca` utility.