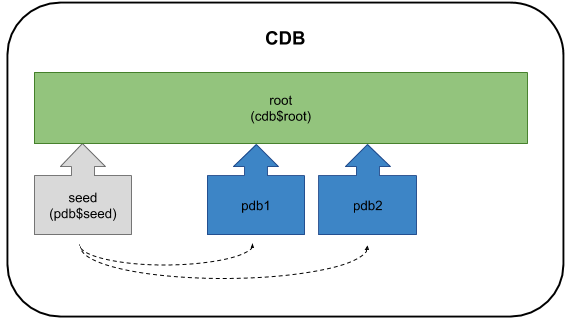
Manual (SQL\*Plus)

There are lots of variations on the CREATE PLUGGABLE DATABASE and ALTER PLUGGABLE DATABASE commands, so we will keep things simple here and only focus on those that mimic what is possible in the DBCA.

For all the operations listed here you must be connected to the CDB with the container set to root (the default). Typically you will be connected to a common user with SYSDBA or SYSOPER privilege. When creating a new pluggable database, the user must have the CREATE PLUGGABLE DATABASE system privilege.

Create a Pluggable Database (PDB) Manually

To create a new pluggable database from the seed database, all we have to do is tell Oracle where the file should be placed.



We can do this using one of three methods. If we are using Oracle Managed Files (OMF) we don't need to worry about the file placement. Oracle will handle it for us.

ALTER SYSTEM SET db\_create\_file\_dest = '/u02/oradata';

From 12.1.0.2 onward there is an inline variation of this using the CREATE\_FILE\_DEST clause. The path set in this clause will be used as the OMF location for the new PDB.

CREATE PLUGGABLE DATABASE pdb2 ADMIN USER pdb\_adm IDENTIFIED BY Password1

CREATE\_FILE\_DEST='/u01/app/oracle/oradata';

The second method uses the FILE\_NAME\_CONVERT clause in the CREATE PLUGGABLE DATABASE statement.

CONN / AS SYSDBA

CREATE PLUGGABLE DATABASE pdb2 ADMIN USER pdb\_adm IDENTIFIED BY Password1

FILE\_NAME\_CONVERT=('/u01/app/oracle/oradata/cdb1/pdbseed/','/u01/app/oracle/oradata/cdb1/pdb2/');

Alternatively, we can specify the PDB\_FILE\_NAME\_CONVERT initialization parameter before calling the command without using the FILE\_NAME\_CONVERT clause.

CONN / AS SYSDBA

ALTER SESSION SET PDB\_FILE\_NAME\_CONVERT='/u01/app/oracle/oradata/cdb1/pdbseed/','/u01/app/oracle/oradata/cdb1/pdb3/';

CREATE PLUGGABLE DATABASE pdb3 ADMIN USER pdb\_adm IDENTIFIED BY Password1;

Every time there is a need to convert file locations, either of these two methods will work. For the remainder of the article I will stick to using the FILE\_NAME\_CONVERT method to cut down on the variations I have to display.

We can see the PDBs are present by querying the DBA\_PDBS and V$PDBS views.

COLUMN pdb\_name FORMAT A20

SELECT pdb\_name, status

FROM dba\_pdbs

ORDER BY pdb\_name;

PDB\_NAME STATUS

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

PDB$SEED NORMAL

PDB1 NORMAL

PDB2 NEW

PDB3 NEW

SQL>

COLUMN name FORMAT A20

SELECT name, open\_mode

FROM v$pdbs

ORDER BY name;

NAME OPEN\_MODE

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

PDB$SEED READ ONLY

PDB1 MOUNTED

PDB2 MOUNTED

PDB3 MOUNTED

SQL>

You can also use the SHOW PDBS command from SQL\*Plus.

SQL> SHOW PDBS

CON\_ID CON\_NAME OPEN MODE RESTRICTED

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

2 PDB$SEED READ ONLY NO

3 PDB1 MOUNTED

4 PDB2 MOUNTED

5 PDB3 MOUNTED

SQL>

The PDBs are created with the status of 'NEW'. They must be opened in READ WRITE mode at least once for the integration of the PDB into the CDB to be complete.

ALTER PLUGGABLE DATABASE pdb2 OPEN READ WRITE;

ALTER PLUGGABLE DATABASE pdb3 OPEN READ WRITE;

SELECT pdb\_name, status

FROM dba\_pdbs

ORDER BY pdb\_name;

PDB\_NAME STATUS

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

PDB$SEED NORMAL

PDB1 NORMAL

PDB2 NORMAL

PDB3 NORMAL

SQL>

SELECT name, open\_mode

FROM v$pdbs

ORDER BY name;

NAME OPEN\_MODE

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

PDB$SEED READ ONLY

PDB1 MOUNTED

PDB2 READ WRITE

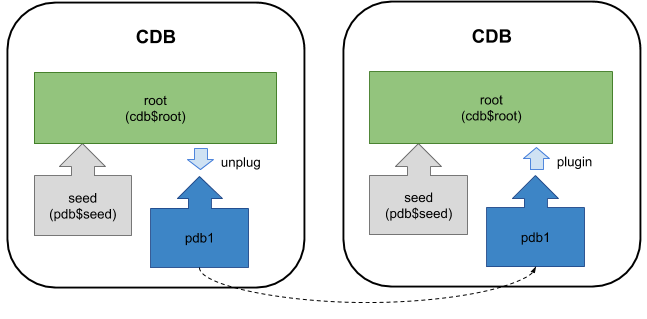
PDB3 READ WRITE

SQL>

 Depending on the syntax used, you may need to grant the PDB\_DBA role to the local admin users for the PDB.

Unplug a Pluggable Database (PDB) Manually

 There is an update to this functionality in Oracle 12.2



Before attempting to unplug a PDB, you must make sure it is closed. To unplug the database use the ALTER PLUGGABLE DATABASE command with the UNPLUG INTO clause to specify the location of the XML metadata file.

ALTER PLUGGABLE DATABASE pdb2 CLOSE;

ALTER PLUGGABLE DATABASE pdb2 UNPLUG INTO '/u01/app/oracle/oradata/cdb1/pdb2/pdb2.xml';

The pluggable database is still present, but you shouldn't open it until the metadata file and all the datafiles are copied somewhere safe.

SELECT name, open\_mode

FROM v$pdbs

ORDER BY name;

NAME OPEN\_MODE

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

PDB$SEED READ ONLY

PDB1 MOUNTED

PDB2 MOUNTED

PDB3 READ WRITE

SQL>

You can delete the PDB, choosing to keep the files on the file system.

DROP PLUGGABLE DATABASE pdb2 KEEP DATAFILES;

SELECT name, open\_mode

FROM v$pdbs

ORDER BY name;

NAME OPEN\_MODE

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

PDB$SEED READ ONLY

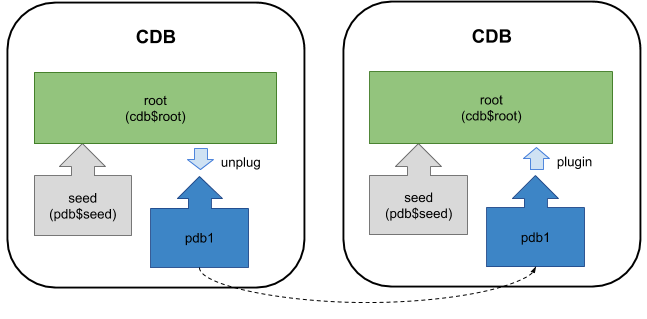
PDB1 MOUNTED

PDB3 READ WRITE

SQL>

Plugin a Pluggable Database (PDB) Manually

 There is an update to this functionality in Oracle 12.2



Plugging in a PDB into the CDB is similar to creating a new PDB. First check the PBD is compatible with the CDB by calling the DBMS\_PDB.CHECK\_PLUG\_COMPATIBILITY function, passing in the XML metadata file and the name of the PDB you want to create using it.

SET SERVEROUTPUT ON

DECLARE

l\_result BOOLEAN;

BEGIN

l\_result := DBMS\_PDB.check\_plug\_compatibility(

pdb\_descr\_file => '/u01/app/oracle/oradata/cdb1/pdb2/pdb2.xml',

pdb\_name => 'pdb2');

IF l\_result THEN

DBMS\_OUTPUT.PUT\_LINE('compatible');

ELSE

DBMS\_OUTPUT.PUT\_LINE('incompatible');

END IF;

END;

/

compatible

PL/SQL procedure successfully completed.

SQL>

If the PDB is not compatible, violations are listed in the PDB\_PLUG\_IN\_VIOLATIONS view. If the PDB is compatible, create a new PDB using it as the source. If we were creating it with a new name we might do something like this.

CREATE PLUGGABLE DATABASE pdb5 USING '/u01/app/oracle/oradata/cdb1/pdb2/pdb2.xml'

FILE\_NAME\_CONVERT=('/u01/app/oracle/oradata/cdb1/pdb2/','/u01/app/oracle/oradata/cdb1/pdb5/');

Instead, we want to plug the database back into the same container, so we don't need to copy the files or recreate the temp file, so we can do the following.

CREATE PLUGGABLE DATABASE pdb2 USING '/u01/app/oracle/oradata/cdb1/pdb2/pdb2.xml'

NOCOPY

TEMPFILE REUSE;

ALTER PLUGGABLE DATABASE pdb2 OPEN READ WRITE;

SELECT name, open\_mode

FROM v$pdbs

ORDER BY name;

NAME OPEN\_MODE

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

PDB$SEED READ ONLY

PDB1 MOUNTED

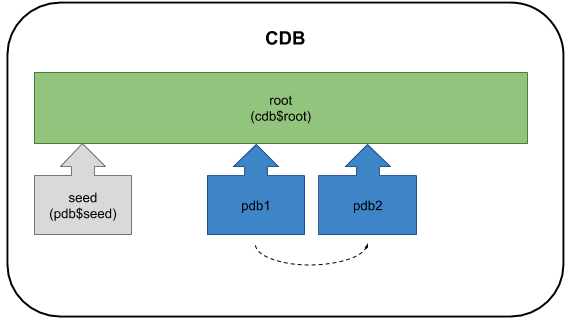
PDB2 READ WRITE

PDB3 READ WRITE

SQL>

Clone a Pluggable Database (PDB) Manually

Cloning an existing local PDB is similar to creating a new PDB from the seed PDB, except now we are using non-seed PDB as the source, which we have to identify using the FROM clause.



If you are using 12.1, or 12.2 without local undo mode, make sure the source PDB is open in READ ONLY mode.

-- Setting the source to read-only is not necessary for Oracle 12cR2.

ALTER PLUGGABLE DATABASE pdb3 CLOSE;

ALTER PLUGGABLE DATABASE pdb3 OPEN READ ONLY;

CREATE PLUGGABLE DATABASE pdb4 FROM pdb3

FILE\_NAME\_CONVERT=('/u01/app/oracle/oradata/cdb1/pdb3/','/u01/app/oracle/oradata/cdb1/pdb4/');

ALTER PLUGGABLE DATABASE pdb4 OPEN READ WRITE;

-- Switch the source PDB back to read/write if you made it read-only.

ALTER PLUGGABLE DATABASE pdb3 CLOSE;

ALTER PLUGGABLE DATABASE pdb3 OPEN READ WRITE;

The cloning syntax also allows for cloning from remote databases using a database link in the local CBD. There are a few restriction associated with this functionality.

This functionality does not work properly in the 12.1.0.1 release of the database, but it has been fixed in 12.1.0.2. You can see some articles specifically on this subject.

Clone a Pluggable Database (PDB) Manually (Metadata Only : NO DATA)

The 12.1.0.2 patchset introduced the ability to do a metadata-only clone. Adding the NO DATA clause when cloning a PDB signifies that only the metadata for the user-created objects should be cloned, not the data in the tables and indexes. You can read more about this feature in the following article.

Delete a Pluggable Database (PDB) Manually

When dropping a pluggable database, you must decide whether to keep or drop the associated datafiles. The PDBs must be closed before being dropped.

ALTER PLUGGABLE DATABASE pdb2 CLOSE;

DROP PLUGGABLE DATABASE pdb2 KEEP DATAFILES;

ALTER PLUGGABLE DATABASE pdb3 CLOSE;

DROP PLUGGABLE DATABASE pdb3 INCLUDING DATAFILES;

ALTER PLUGGABLE DATABASE pdb4 CLOSE;

DROP PLUGGABLE DATABASE pdb4 INCLUDING DATAFILES;

SELECT name, open\_mode

FROM v$pdbs

ORDER BY name;

NAME OPEN\_MODE

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

PDB$SEED READ ONLY

PDB1 MOUNTED

SQL>