Cheatography

Oracle Multitenant Cheat Sheet

by Adam Cumming via cheatography.com/121025/cs/21994/

Essential Terminology

Multitenant	The Oracle Architecture that consists of a CDB, and one or more PDBs
Container Database (CDB)	A traditional database instance, but has the ability to support PDBs
Pluggable Database (PDB)	A collection of Tablespaces which supports it's own independant Role and User security, and can be easily moved between CDBs
Root Database	The instance administrative layer that sets above PDBs. Users and Roles here must be preceded by c##
Seed Database	A PDB that remains offline to be used as a template for creating new blank PDBs

Daily Use Commands (from SQL command line)

Connect to Contaner or PDB

```
CONN <user>/<pwd>@//<host>:<listener port>/<service> {as sysdba};
```

```
CONN <user>/<pwd>@//<tns_entry> {as sysdba};
```

Display Current Container or PDB

```
SHOW CON_NAME;
```

```
SELECT SYS_CONTEXT('USERENV','CON_NAME')
FROM DUAL;
```

```
SHOW CON_ID;
```

List Containers and PDBs on Instance

```
SELECT PDB_NAME, Status
FROM DBA_PDBS
ORDER BY PDB_Name;
```

```
SELECT Name, Open_Mode
FROM V$PDBS
ORDER BY Name;
```

```
SELECT Name,PDB
FROM V$SERVICES
ORDER BY Name;
```

Change Container or PDB

```
ALTER SESSION SET container= <name>;
```

```
ALTER SESSION SET container=cdb$root;
```

Cloning/Creating a PDB

First, set your source and target datafile paths...

```
ALTER SESSION SET PDB_FILE_NAME_CONVERT='</seed path/>','</target path/>';
```

Then run the create command from the target root container...

```
CREATE PLUGGABLE DATABASE <New PDB Name>
                          ADMIN USER <Username>
                                IDENTIFIED BY <Password>
                          FROM <Source PDB[@dblink]>
```

Finally, Open the newly created database...

```
ALTER PLUGGABLE DATABASE <target pdb> OPEN;
```

NOTE: Creating a PDB is just cloning from the seed db.

[@dblink] is optional and used when creating PDB from existing PDB on another instance.

If using dblink, the link user should be an administrative user on the source PDB

By **Adam Cumming**
cheatography.com/adam-cumming/

Published 1st May, 2020.
Last updated 30th April, 2020.

Managing a Multitenant Database

Startup and Shutdown

Startup and Shutdown of a multitenant database function the same as on a regular database, however, if connected to pluggable database, only the pluggable database shuts down. If connected to the root container database then the entire instance shuts down. Pluggable databases also have their own commands that can be run from the root container or other pluggable db.

```
ALTER PLUGGABLE DATABASE <name>OPEN READ WRITE{RESTRICTED}{FORCE};
ALTER PLUGGABLE DATABASE <name> OPEN READ ONLY {RESTRICTED}{FORCE};
ALTER PLUGGABLE DATABASE <name> OPEN UPGRADE {RESTRICTED};
ALTER PLUGGABLE DATABASE <name> CLOSE {IMMEDIATE};
```

To retain state as startup state of container...

```
ALTER PLUGGABLE DATABASE <name> SAVE STATE;
```

Roles and Users

Common Users and Roles must be created in the root container and prefixed by the characters `c##`

Local Users and Roles must be created in pdb

Granting Roles and Privileges

```
GRANT <privelege/role> TO <user> CONTAINER=<PDB name or ALL>;
```

If local only, grant from pdb and omit container argument.

Backup and Recovery

Backup

RMAN connection to root container...

Normal Backup will capture full instance

For just Root Container

```
BACKUP DATABASE ROOT
```

For Pluggable Databases

```
BACKUP PLUGGABLE DATABASE <pdb1,pdb2,pdb3>
```

RMAN connection to pluggable database will only work on that pdb

Restore

Connect to root container. Normal restore is full instance. For pdb...

```
RUN {ALTER PLUGGABLE DATABASE <pdb> CLOSE;
   SET UNTIL TIME "<timeset value>";
   RESTORE PLUGGABLE DATABASE <pdb>;
   RECOVER PLUGGABLE DATABASE <pdb>;
   ALTER PLUGGABLE DATABASE <pdb>OPEN; }
```

Moving PDB's (Unplugging/Plugging in PDB)

Exporting/Unplugging An Existing PDB

To unplug a database, use the following commands. It is recommended that the path used match the datafile storage location.

```
ALTER PLUGGABLE DATABASE <pdb_name> CLOSE;
ALTER PLUGGABLE DATABASE <pdb_name> UNPLUG INTO '</path/><name>.xml';
DROP PLUGGABLE DATABASE <pdb_name> KEEP DATAFILES;
```

Importing/Plugging in PDB into a CDB

Before importing/plugging in a PDB into a CDB a small procedure should be run to Validate the integrity and compatibility of the PDB.

```
SET SERVEROUTPUT ON
DECLARE
     l_result BOOLEAN;
BEGIN
     l_result := DBMS_PDB.CHECK_PLUG_COMPATIBILITY(
                         PDB_DESCR_FILE => '</path/><name>.xml',
                         PDB_NAME       => '<name>');
     IF l_result THEN
          DBMS_OUTPUT.PUT_LINE('Compatible, OK to Proceed');
     ELSE
          DBMS_OUTPUT.PUT_LINE('Incompatible, See PDB_PLUG_IN_VIOLATIONS for details');
     END IF;
END;
```

If the pdb is validated, then use the following commands to import/plug it in. Reference the xml file path specified during export, and the datafile path...

```
CREATE PLUGGABLE DATABASE <new_pdb_name> USING '</path/><name>.xml'
       FILE_NAME_CONVERT=('</source path/>','</dest path/>');
ALTER PLUGGABLE DATABASE <new_pdb_name> OPEN;
```

PROXY Database Functionality

A special type of PDB is a Proxy PDB. A Proxy PDB essentially is a PDB that is linked to another PDB so that if a PDB is being migrated to another environment and there is a desire to not modify all source code to new location references first, they can still use the old references on a Proxy and the actions will take place on the New DB.

To setup, first setup a dblink to the pluggable target

```
CREATE PLUGGABLE DATABASE <proxy pdb name> AS PROXY FROM <target pdb>@<dblink>;
```

NOTE: dblink may be dropped after proxy db is created

In a proxy DB the alter Database and Alter Pluggable Database commands apply to the proxy db. All other DDL applies to the target db.

By **Adam Cumming**
cheatography.com/adam-cumming/

Published 1st May, 2020.
Last updated 30th April, 2020.

Misc Other Multitenant Management Commands

Cloning from NonCDB to CDB

NonCDB must support multitenant and use dblink on NONCDB to connect

DBLink user must have CREATE SESSION and CREATE PLUGGABLE DATABASE privileges

```
CREATE PLUGGABLE DATABASE <new_pdb> FROM NON$CDB@<dblink>
       FILE_NAME_CONVERT=('</source datafile path/>','</target datafile path/>');
@ORACLE_HOME/rdbms/admin/noncdb_to_pdb.sql
ALTER PLUGGABLE DATABASE <target pdb> OPEN;
```

Moving a PDB

```
CREATE PLUGGABLE DATABASE <new pdb> FROM <old pdb>@<dblink> RELOCATE;
ALTER PLUGGABLE DATABASE <new pdb> OPEN;
```

Removing a PDB

```
ALTER PLUGGABLE DATABASE <name> CLOSE;
DROP PLUGGABLE DATABASE <name> INCLUDING DATAFILES;
```

Exporting/Unplugging a pdb to a single compressed file

```
ALTER PLUGGABLE DATABASE <pdb_name> UNPLUG INTO '</path/><filename>.pdb';
```

Importing/Plugging in a pdb from a single compressed file

```
CREATE PLUGGABLE DATABASE <new pdb name> USING '</path/><filename>.pdb';
```

Note that compressed pdb files for export and import are suffixed by .pdb and are a zip fle format.

By **Adam Cumming**
cheatography.com/adam-cumming/

Published 1st May, 2020.
Last updated 30th April, 2020.