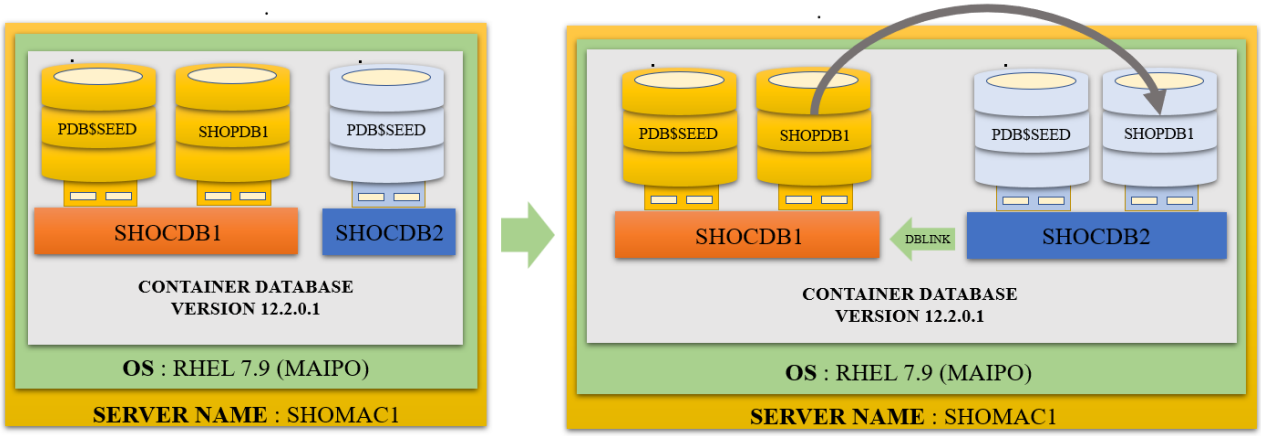
CLONE ORACLE 12c PLUGGABLE DATABASE IN ANOTHER CDB

Clone pluggable database PDB (SHOPDB1) from container database CDB (SHOCDB1) to another container database (SHOCDB2) as pluggable database PDB (SHOPDB1)



Let's understand our environment

We can see that we have 2 container databases i.e., SHOCDB1 & SHOCDB2 running over the same server i.e., SHOMAC1

```
[oracle@SHOMAC1 ~]$
[oracle@SHOMAC1 ~]$ ps -ef|grep -i pmon_ |grep -v grep
oracle   24268     1  0 15:45 ?        00:00:00 ora_pmon_shocdb2
oracle   26166     1  0 Aug31 ?        00:00:05 ora_pmon_shocdb1
[oracle@SHOMAC1 ~]$
```

When we login into the first container database i.e., SHOCDB1

We can see that it has below PDBS

The pluggable database: PDB$SEED, SHOPDB1

```
[oracle@SHOMAC1 ~]$ . oraenv
ORACLE_SID = [shocdb1] ? shocdb1
The Oracle base remains unchanged with value /u01/app/oracle
[oracle@SHOMAC1 ~]$
[oracle@SHOMAC1 ~]$ sqlplus "/ as sysdba"

SQL*Plus: Release 12.2.0.1.0 Production on Thu Sep 1 16:07:07 2022

Copyright (c) 1982, 2016, Oracle.  All rights reserved.


Connected to:
Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production

SYS@shocdb1 01-SEP-22>show pdbs

    CON_ID CON_NAME                       OPEN MODE  RESTRICTED
---------- ------------------------------ ---------- ----------
         2 PDB$SEED                       READ ONLY  NO
         3 SHOPDB1                        READ WRITE NO
SYS@shocdb1 01-SEP-22>
```

Now when we login into the second container database i.e., SHOCDB2

We can see that it has below PDB

The pluggable database: PDB$SEED

```
[oracle@SHOMAC1 ~]$ . oraenv
ORACLE_SID = [shocdb2] ? shocdb2
The Oracle base remains unchanged with value /u01/app/oracle
[oracle@SHOMAC1 ~]$
[oracle@SHOMAC1 ~]$ sqlplus "/ as sysdba"

SQL*Plus: Release 12.2.0.1.0 Production on Thu Sep 1 16:05:23 2022

Copyright (c) 1982, 2016, Oracle.  All rights reserved.


Connected to:
Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production

SYS@shocdb2 01-SEP-22>show pdbs

    CON_ID CON_NAME                       OPEN MODE  RESTRICTED
---------- ------------------------------ ---------- ----------
         2 PDB$SEED                       READ ONLY  NO
SYS@shocdb2 01-SEP-22>
```

The location of the datafiles

When we login into the first container database i.e., SHOCDB1

The location of the datafiles for the PDBs are shown as below:

The container database: SHOCDB1 → /u01/app/oracle/oradata/SHOCDB1/datafile/

The Pluggable database: SHOPDB1 → /u01/app/oracle/oradata/SHOCDB1/E79387E71AAD6720E0536538A8C001D5/datafile/

Note: here we can see that the datafile's naming convention is using OMF (oracle managed format) and SHOPDB1 PDB has a GUID as E79387E71AAD6720E0536538A8C001D5

```
SYS@shocdb1 31-AUG-22>col name for a120
SYS@shocdb1 31-AUG-22>select name from v$datafile;

NAME
--------------------------------------------------------------------------------------------------------------------------------
/u01/app/oracle/oradata/SHOCDB1/datafile/o1_mf_system_kjzyld81_.dbf
/u01/app/oracle/oradata/SHOCDB1/datafile/o1_mf_sysaux_kjzy2j5z_.dbf
/u01/app/oracle/oradata/SHOCDB1/datafile/o1_mf_undotbs1_kjzy3999_.dbf
/u01/app/oracle/oradata/SHOCDB1/datafile/o1_mf_system_kjzy5sh2_.dbf
/u01/app/oracle/oradata/SHOCDB1/datafile/o1_mf_sysaux_kjzy5scj_.dbf
/u01/app/oracle/oradata/SHOCDB1/datafile/o1_mf_users_kjzy3bdx_.dbf
/u01/app/oracle/oradata/SHOCDB1/datafile/o1_mf_undotbs1_kjzy5sh8_.dbf
/u01/app/oracle/oradata/SHOCDB1/E79387E71AAD6720E0536538A8C001D5/datafile/o1_mf_system_kjzyhq30_.dbf
/u01/app/oracle/oradata/SHOCDB1/E79387E71AAD6720E0536538A8C001D5/datafile/o1_mf_sysaux_kjzyhq38_.dbf
/u01/app/oracle/oradata/SHOCDB1/E79387E71AAD6720E0536538A8C001D5/datafile/o1_mf_undotbs1_kjzyhq39_.dbf
/u01/app/oracle/oradata/SHOCDB1/E79387E71AAD6720E0536538A8C001D5/datafile/o1_mf_users_kjzyjlcv_.dbf

11 rows selected.

SYS@shocdb1 31-AUG-22>
```

Now Let's check the CDB undo mode?

You can configure a CDB to use local undo in every container or to use shared undo (default) for the entire CDB.

A CDB runs either in local or shared undo mode. The undo mode applies to the entire CDB. Therefore, every container either uses shared undo or local undo.

To determine the current CDB undo mode, run the following query in the CDB root:

Col PROPERTY_NAME for a25

Col PROPERTY_VALUE for a19

SELECT PROPERTY_NAME, PROPERTY_VALUE

FROM DATABASE_PROPERTIES

WHERE PROPERTY_NAME = 'LOCAL_UNDO_ENABLED';

If the query returns TRUE for the PROPERTY_VALUE, then the CDB is in local undo mode. Otherwise, the CDB is in shared undo mode.

```
SYS@shocdb1 31-AUG-22>
SYS@shocdb1 31-AUG-22>Col PROPERTY_NAME for a25
SYS@shocdb1 31-AUG-22>Col PROPERTY_VALUE for a19
SYS@shocdb1 31-AUG-22>SELECT PROPERTY_NAME, PROPERTY_VALUE FROM   DATABASE_PROPERTIES
  2  WHERE  PROPERTY_NAME = 'LOCAL_UNDO_ENABLED';

PROPERTY_NAME             PROPERTY_VALUE
------------------------- -------------------
LOCAL_UNDO_ENABLED        TRUE

SYS@shocdb1 31-AUG-22>
```

Please note that, if the CDB is in shared undo mode, then the pluggable database PDB must be in open read-only, follow the below 2 steps before cloning.

STEP 1 - Close the pluggable database PDB.

alter pluggable database SHOPDB1 close immediate;

```
SYS@shocdb1 31-AUG-22>
SYS@shocdb1 31-AUG-22>alter pluggable database SHOPDB1 close immediate;

Pluggable database altered.

SYS@shocdb1 31-AUG-22>
```

STEP 2 - Open the pluggable database PDB to READ ONLY.

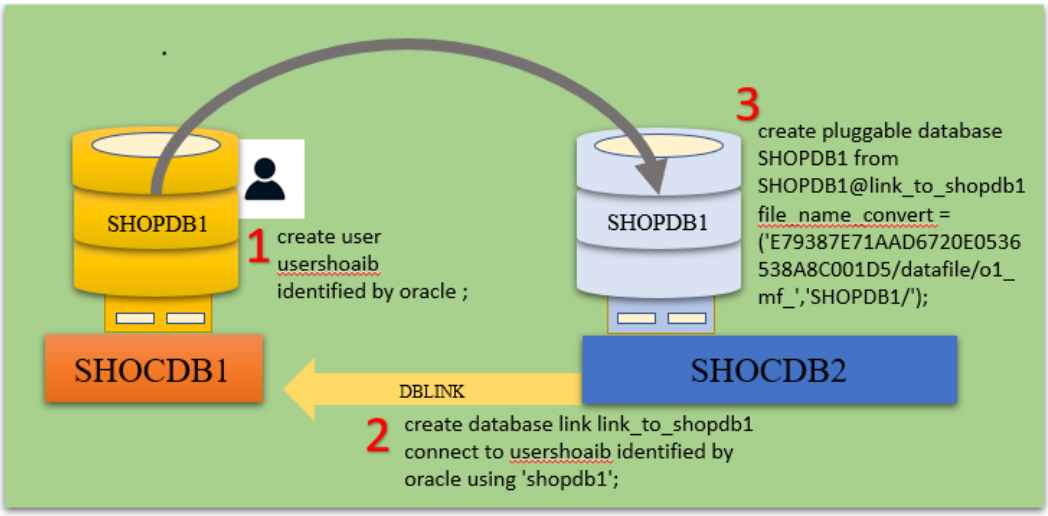
alter pluggable database SHOPDB1 open read only;

```
SYS@shocdb1 31-AUG-22>alter pluggable database SHOPDB1 open read only;

Pluggable database altered.

SYS@shocdb1 31-AUG-22>show pdbs

    CON_ID CON_NAME                       OPEN MODE  RESTRICTED
---------- ------------------------------ ---------- ----------
         2 PDB$SEED                       READ ONLY  NO
         3 SHOPDB1                        READ ONLY  NO
SYS@shocdb1 31-AUG-22>
SYS@shocdb1 31-AUG-22>
```

In our environment the container database CDB is local undo mode (it is recommended for the production environment too), so we will skip above step 1 and step 2.

Note: If container database CDB is with local undo mode then we can clone the PDB online (read/write).

Steps to Clone the Pluggable database (PDB)



Create a database link on SHOCDB2 pointing to the SHOCDB1.

create database link link_to_shopdb1 connect to usershoaib identified by oracle using 'shopdb1';

```
[oracle@SHOMAC1 ~]$  . oraenv
ORACLE_SID = [shocdb2] ? shocdb2
The Oracle base remains unchanged with value /u01/app/oracle
[oracle@SHOMAC1 ~]$ sqlplus "/ as sysdba"

SQL*Plus: Release 12.2.0.1.0 Production on Thu Sep 1 16:41:38 2022

Copyright (c) 1982, 2016, Oracle.  All rights reserved.


Connected to:
Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production

SYS@shocdb2 01-SEP-22>show pdbs

    CON_ID CON_NAME                       OPEN MODE  RESTRICTED
---------- ------------------------------ ---------- ----------
         2 PDB$SEED                       READ ONLY  NO
SYS@shocdb2 01-SEP-22>
SYS@shocdb2 01-SEP-22>create database link link_to_shopdb1 connect to usershoaib
  2  identified by oracle using 'shopdb1';

Database link created.

SYS@shocdb2 01-SEP-22>
```

Clone the target PDB from the remote database via a database link.

create pluggable database SHOPDB1 from SHOPDB1@link_to_shopdb1 file_name_convert=('E79387E71AAD6720E0536538A8C001D5/datafile/o1_mf_','SHOPDB1/');

```
[oracle@SHOMAC1 ~]$ . oraenv
ORACLE_SID = [shocdb2] ? shocdb2
The Oracle base remains unchanged with value /u01/app/oracle
[oracle@SHOMAC1 ~]$
[oracle@SHOMAC1 ~]$ sqlplus " / as sysdba"

SQL*Plus: Release 12.2.0.1.0 Production on Thu Sep 1 17:37:52 2022

Copyright (c) 1982, 2016, Oracle.  All rights reserved.


Connected to:
Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production

SYS@shocdb2 01-SEP-22>show pdbs

    CON_ID CON_NAME                       OPEN MODE  RESTRICTED
---------- ------------------------------ ---------- ----------
         2 PDB$SEED                       READ ONLY  NO
SYS@shocdb2 01-SEP-22>
SYS@shocdb2 01-SEP-22>create pluggable database SHOPDB1 from SHOPDB1@link_to_shopdb1
  2  file_name_convert=('E79387E71AAD6720E0536538A8C001D5/datafile/o1_mf_','SHOPDB1/');

Pluggable database created.

SYS@shocdb2 01-SEP-22>show pdbs

    CON_ID CON_NAME                       OPEN MODE  RESTRICTED
---------- ------------------------------ ---------- ----------
         2 PDB$SEED                       READ ONLY  NO
         3 SHOPDB1                        MOUNTED
SYS@shocdb2 01-SEP-22>
```

Parallel keyword can be used during PDB cloning, If we have to clone a big database, say > 10 TB , so we can add some degrees of parallelism.

create pluggable database SHOPDB1 from SHOPDB1@link_to_shopdb1 file_name_convert=('E79387E71AAD6720E0536538A8C001D5/datafile/o1_mf_','SHOPDB1/') parallel 8;

```
SYS@shocdb2 01-SEP-22>create pluggable database SHOPDB1 from SHOPDB1@link_to_shopdb1
  2  file_name_convert=('E79387E71AAD6720E0536538A8C001D5/datafile/o1_mf_','SHOPDB1/') parallel 8;

Pluggable database created.

SYS@shocdb2 01-SEP-22>
```

Open the SHOPDB1 pluggable database (PDB) to READ WRITE.

```
SYS@shocdb2 01-SEP-22>show pdbs

    CON_ID CON_NAME                       OPEN MODE  RESTRICTED
---------- ------------------------------ ---------- ----------
         2 PDB$SEED                       READ ONLY  NO
         4 SHOPDB1                        MOUNTED
SYS@shocdb2 01-SEP-22>
SYS@shocdb2 01-SEP-22>alter pluggable database SHOPDB1 open;

Pluggable database altered.

SYS@shocdb2 01-SEP-22>show pdbs

    CON_ID CON_NAME                       OPEN MODE  RESTRICTED
---------- ------------------------------ ---------- ----------
         2 PDB$SEED                       READ ONLY  NO
         4 SHOPDB1                        READ WRITE NO
SYS@shocdb2 01-SEP-22>
```

Now finally in the second container database i.e., SHOCDB2

We can see that it has below PDBs

The pluggable database: PDB$SEED, SHOPDB1