Practice 21 Managing CDB Fleet

Practice Overview

In this practice, you will configure a CDB Fleet consisting of `oradb` (lead) and `oradb2` (member).You will then obtain information about the member containers from the lead.

Assumption

This practice assumes that you have `srv1` and `srv2` and their databases up and running.

A. Configure the CDB Fleet

In this section of the practice, you will configure a CDB Fleet consisting of `oradb` (lead) and `oradb2` (member)

1. Open Putty and start two sessions as `oracle`, one to `srv1` and the other one to `srv2`
2. In `srv1` session, start SQL*Plus and login to its cdb `oradb`
3. In `srv2` session, start SQL*Plus and login to its cdb `oradb2`
4. In `srv1` session, verify that oradb is not a lead CDB in a CDB fleet.

```
col PROPERTY_VALUE for a20
SELECT PROPERTY_VALUE FROM DATABASE_PROPERTIES WHERE PROPERTY_NAME='LEAD_CDB';
```

5. Designate `oradb` in `srv1` to be the lead CDB in the feel configuration.

```
ALTER DATABASE SET LEAD_CDB = TRUE;
```

6. Run the following command to verify that `oradb` is a lead CDB in the fleet configuration.

```
SELECT PROPERTY_VALUE FROM DATABASE_PROPERTIES WHERE PROPERTY_NAME='LEAD_CDB';

host cat /u01/app/oracle/diag/rdbms/oradb/oradb/trace/alert_oradb.log | grep
Fleet
```


7. In `oradb`, create a common user.

   This user will be used by the member CDBs to create database links to the lead CDB.

```
CREATE USER c##fleet_user IDENTIFIED BY ABcd##1234;
GRANT CREATE SESSION TO c##fleet_user;
```

8. In `srv2` session, verify that `oradb2` is not a member of a fleet.

```
sqlplus / as sysdba

col PROPERTY_VALUE for a20
SELECT PROPERTY_VALUE FROM DATABASE_PROPERTIES WHERE
PROPERTY_NAME='LEAD_CDB_URI';

SELECT SYS_CONTEXT('USERENV', 'IS_MEMBER_CDB') MEMBER_CDB FROM DUAL;
```

9. Create a database link to the fleet lead

The database link is a **public** database link, which means all the users in the CDB root can access it. However, all the users in the CDB root are common users.

```
CREATE PUBLIC DATABASE LINK lead_cdb_link CONNECT TO c##fleet_user IDENTIFIED BY
ABcd##1234 USING 'ORADB';

# test the db link:
SELECT SYSDATE FROM DUAL@lead_cdb_link;
```

10. Set `oradb2` as a member of the fleet.

```
ALTER DATABASE SET LEAD_CDB_URI = 'dblink:LEAD_CDB_LINK';
```

11. Verify that `oradb2` is a CDB member in the CDB fleet.

```
SELECT PROPERTY_VALUE FROM DATABASE_PROPERTIES WHERE  PROPERTY_NAME =
'LEAD_CDB_URI';

SELECT SYS_CONTEXT('USERENV', 'IS_MEMBER_CDB') MEMBER_CDB FROM DUAL;

host cat /u01/app/oracle/diag/rdbms/oradb2/oradb2/trace/alert_oradb2.log | grep
Fleet
```


12. In `srv1` session, list all the PDBs.

You should observe that pdb21 is returned in the command output. Its status is `MOUNTED` (and not `OPEN`) because we cannot open `pdb21` from `oradb`.

```
show pdbs
```

13. In `srv1` session, list the pdbs from the view `DBA_PDBS`

Observe the following from querying the `DBA_PDBS`:

- The member CBB (`oradb2`) is returned by the view.
- The status of the member PDBs is "`STUB`". This is a way to distinguish fleet member PDBs from the lead PDBs.
- The proxy flag of the fleet member PDBs is set to `YES`. However, CDB fleet member PDBs do not inherit the full functionality of the proxy PDBs. You will examine this point soon in this practice.

```
col PDB_NAME for a15
col IS_PROXY_PDB for a15

SELECT PDB_ID, PDB_NAME, STATUS, IS_PROXY_PDB FROM DBA_PDBS;
```

14. In `srv1` session, issue the following query.

By referring to the column `MEMBER_CDB` in the `V$CONTAINERS`, we can know which fleet member container is a PDB or a CDB.

```
col NAME for a20
col MEMBER_CDB for a15
SELECT NAME, OPEN_MODE, MEMBER_CDB FROM V$CONTAINERS;
```

So far, `oradb2` is registered as a fleet member in `oradb`. In the following steps, you will test the CDB Fleet functionality.

Retrieving Cross-container Dictionary Information

In the following step, you will test retrieving cross-container information from the data dictionary views.

15. In `srv1`, try querying a few v$ or CDB views that retrieve information about the PDBs.

```
# retrieve list of all the datafiles in all the PDBs
col FILE_NAME for a85
SELECT CON_ID , FILE_NAME FROM CDB_DATA_FILES ORDER BY 1;

# retrieve number of objects in each seen PDB
SELECT CON_ID, COUNT(*) FROM CDB_OBJECTS GROUP BY CON_ID;
```


None of the queries above retrieves any data about the member PDB. I have been in contact with Oracle support on this issue for a while. I have not learnt from them about the root cause or the resolution. I must update this document once we reach to a conclusion.

Note: Proxy PDB is an alternative solution to this requirement.

Using CONTAINERS Claus to Query Cross-container Common User Data

In the following steps, you will create testing common-user data in the lead and member PDBs. Then, you will test querying the data from the lead CDB.

16. In `srv1` session, run the following commands to create a common user and a testing table in each container.

```
conn sys/oracle@oradb as sysdba

CREATE USER c##common_user IDENTIFIED BY ABcd##1234 QUOTA UNLIMITED ON users;
GRANT CREATE SESSION, CREATE TABLE, CREATE SYNONYM TO c##common_user
CONTAINER=ALL;

conn c##common_user/ABcd##1234@oradb
CREATE TABLE c##common_user.TEST (ID NUMBER(2), MESSAGE VARCHAR2(80));

conn c##common_user/ABcd##1234@pdb1
CREATE TABLE c##common_user.TEST (ID NUMBER(2), MESSAGE VARCHAR2(80));
INSERT INTO c##common_user.TEST VALUES ( 1, 'INSERTED IN PDB1');
INSERT INTO c##common_user.TEST VALUES ( 2, 'INSERTED IN PDB1');
COMMIT;
```

17. In `srv2` session, run the following commands to create a common user and a testing table in each container.

```
conn sys/oracle@oradb2 AS SYSDBA

CREATE USER c##common_user IDENTIFIED BY ABcd##1234 QUOTA UNLIMITED ON users;
GRANT CREATE SESSION, CREATE TABLE, CREATE VIEW, CREATE SYNONYM TO c##common_user
CONTAINER=ALL;

conn c##common_user/ABcd##1234@oradb2
CREATE TABLE c##common_user.TEST (ID NUMBER(2), MESSAGE VARCHAR2(80));

conn c##common_user/ABcd##1234@pdb21
CREATE TABLE c##common_user.TEST (ID NUMBER(2), MESSAGE VARCHAR2(80));
INSERT INTO c##common_user.TEST VALUES ( 3, 'INSERTED IN PDB21');
INSERT INTO c##common_user.TEST VALUES ( 4, 'INSERTED IN PDB21');
COMMIT;
```

18. In `srv1` session, test querying the testing table from all CDB Fleet containers.

```
conn c##common_user/ABcd##1234@oradb
col MESSAGE for a35
SELECT * FROM CONTAINERS(TEST) ORDER BY id;
```

The data should be successfully retrieved from all the CDB containers.

Note: from my experience, if the CDB Member has a PDB with a same name as a PDB in the Lead CDB, it doesn't appear in the Lead. As an additional practice, try it yourself .

Deactivating CDB Fleet

After examining the CDB Fleet feature, let's undo what we have configured.

19. In `srv2`, issue the following command to unregister the member CDB from `oradb`.

```
conn / as sysdba

ALTER DATABASE SET LEAD_CDB_URI = '';
```

20. In `srv1`, verify that `oradb2` is unregistered.

```
conn / as sysdba

col NAME for a20
col MEMBER_CDB for a15
SELECT NAME, OPEN_MODE, MEMBER_CDB FROM V$CONTAINERS;
```

21. In `srv1`, disable the CDB lead.

```
ALTER DATABASE SET LEAD_CDB = FALSE;

col PROPERTY_VALUE for a20
SELECT PROPERTY_VALUE FROM DATABASE_PROPERTIES WHERE PROPERTY_NAME='LEAD_CDB';
```


22. Verify the member containers are not seen anymore.

```
col NAME for a20
col MEMBER_CDB for a15
SELECT NAME, OPEN_MODE, MEMBER_CDB FROM V$CONTAINERS;
```

Note: To disable the CDB Fleet, always unregister all the members first, then disable the CDB lead, in that order. If there is any member still registered in the CDB Fleet, it stays registered even if the CDB lead is disabled.

23. In `srv2`, drop the database link and the testing common user.

```
DROP PUBLIC DATABASE LINK lead_cdb_link;

-- the following command drops the user from all the local PDBs
DROP USER c##common_user CASCADE;
```

24. In `srv1` session, drop the created common users.

```
conn / as sysdba
DROP USER c##fleet_user;
DROP USER c##common_user CASCADE;
```

Summary

CDB Fleet provides the following capabilities:

- Submit a cross-container query on v$ and CDB views in the lead CDB that will automatically execute in all PDBs across the CDB fleet. In our testing cases, this capability works only on the views `V$PDBS`, `DBA_PDBS`, `V$CONTAINERS`.
- Use the `CONTAINERS` clause to run queries across all PDBs in the CDB fleet using common schemas and common application objects in different application containers across the CDB fleet.
- To deactivate a CDB fleet, first unregister all the members and then disable the CDB lead.