Practice 18: Managing Applications and Application Containers

Practice Overview

In this practice, you will.

- Create Application root and install application in the application root
- Create Application PDBs and sync them with the application root
- Study how Application PDBs operate

Practice Assumptions

- `CDB1` database (in `srv1`) is up and running.
- You successfully downloaded the `hr_app_v1.0.sql` script from the lecture downloadable resources.

About the Application Script used in this practice

You will use the `hr_app_v1.0.sql` script to create application components in an application root. Basically, the script creates sequences and tables that are very similar to the tables owned by the standard `HR` example schema. After creating the sequences and the tables, it load the data-linked tables and the extended data-linked tables with some sample rows.

However, in real life scenario, an application schema typically has much more than just tables and sequences. As you know, it normally has views, constraints, table relationships, PL/SQL program units …etc.

When I try to create a more complex schema that has all those object types, unfortunately later I get hit with more than one bug when I perform the basic tasks on the application. Some of those bugs have patches to resolve them and some are still under development (at the time of this writing).

If you want to test on a more complex schema, you can use a downloadable script file named as `hr_app_sample.sql` to create one schema with multiple object types in it.

Note: I recommend taking a snapshot of the appliance before implementing the practice.

A. Create Application root

In this section of the practice, you will create an application root (named HR_AC) and grant its administrator the privileges to *totally* manage it.

1. Login to the CDB root as SYSDBA and create an application root named HR_AC.

In this practice we assume the application root administrator will be responsible for managing the application root and the application PDBs that are associated with it.

```
sqlplus sys/oracle@cdb1 as sysdba

CREATE PLUGGABLE DATABASE hr_ac AS APPLICATION CONTAINER
ADMIN USER hr_acadm IDENTIFIED BY oracle;

ALTER PLUGGABLE DATABASE hr_ac OPEN;

col name format a10
SELECT CON_ID, NAME, OPEN_MODE
FROM V$PDBS WHERE APPLICATION_ROOT='YES';

-- save the state of HR_AC (otherwise, it will be closed when you reboot):
ALTER PLUGGABLE DATABASE hr_ac SAVE STATE;
```

2. View the tablespaces and the datafiles created for the application container HR_AC.

Obtain the CON_ID value from the query in the previous step.

Observe that the tablespaces SYSTEM, SYSAUX, and UNDOTBS1 are created.

```
SELECT FILE_NAME, TABLESPACE_NAME FROM CDB_DATA_FILES WHERE CON_ID=5;
```

3. Switch the current container to the application root and check which privilege is granted to HR_ACADM

Observe that the application root administrator is only granted the role PDB_DBA. This role has three privileges granted to it.

```
ALTER SESSION SET CONTAINER=HR_AC;

-- check the roles granted to the user:
col grantee format a10
col granted_role format a15
SELECT GRANTEE, GRANTED_ROLE, COMMON
FROM DBA_ROLE_PRIVS where GRANTEE ='HR_ACADM';

-- check the privileges granted to the role:
col role format a10
col privilege format a30
SELECT ROLE, PRIVILEGE, ADMIN_OPTION, COMMON, INHERITED
  FROM ROLE_SYS_PRIVS WHERE ROLE='PDB_DBA';
```

4. Configure the `tnsnames.ora` file to allow connecting to the application root through the listener.

```
host vi /u01/app/oracle/product/12.2.0.1/db_1/network/admin/tnsnames.ora

HR_AC =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = srv1)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = hr_ac.localdomain)
    )
  )
```

5. Test the configuration

```
conn sys/oracle@hr_ac as sysdba
```

B. Install an application in the application root

In this section of the practice, you will install an application (named HR_APP) in the HR_AC application root. This application is nearly the same as the standard HR example schema.

6. Login to the application root as SYSDBA.

```
conn sys/oracle@hr_ac as sysdba
```

7. Begin installing the HR_APP application.

When you install a new application in an application root, you always start with this statement.

```
ALTER PLUGGABLE DATABASE APPLICATION hr_app BEGIN INSTALL '1.0';
```

8. Create the application tablespace (HR_TBS) and create the application owner user (HR).

The application owner (HR) should have the privileges enough to control the application objects, like the application tables, indexes, sequences, PL/SQL program units… etc.

The application owner is common user in the application root.

```
CREATE TABLESPACE hr_tbs;

CREATE USER HR IDENTIFIED BY oracle DEFAULT TABLESPACE HR_TBS QUOTA UNLIMITED ON
HR_TBS CONTAINER = ALL;

GRANT CREATE SESSION, CREATE TABLE, CREATE SEQUENCE, CREATE VIEW, CREATE
PROCEDURE, CREATE TRIGGER TO HR;
```

9. Switch the current schema to the application user.

```
ALTER SESSION SET CURRENT_SCHEMA=hr;
```

10. Examine the code in the hr_app_v1.0.sql script. Notice the following:

- The SHARING option in the CREATE statements.
- The shared data in the end of the script file.

11. Create a script file in srv1 and copy the code from the hr_app_v1.0.sql script file into it. Do not exit from the SQL*Plus session.

If you open the script file using the Windows Notepad utility, make sure the "**Word Wrap**" is deselected before you copy the code in the file. This option can be accessed from the "**Format**" menu in the Notepad.

```
host mkdir /home/oracle/scripts
host vi /home/oracle/scripts/hr_app_v1.0.sql
```

12. Run the script file.

```
@/home/oracle/scripts/hr_app_v1.0.sql
```

13. End the application installation, if all the commands in the script successfully run.

```
ALTER PLUGGABLE DATABASE APPLICATION hr_app END INSTALL '1.0';
```

14. Verify that the application has been successfully created.

```
column app_name format a15
column app_version format a10
column app_status format a15

SELECT APP_NAME, APP_VERSION, APP_STATUS
FROM DBA_APPLICATIONS
WHERE APP_IMPLICIT='N';
```

C. Create application PDBs from the application root

In this section of the practice, you will create a couple of application PDBs from the application root container `HR_AC`.

15. Login to the application root as `SYSDBA`.

```
conn sys/oracle@hr_ac as sysdba
```

16. Create an application PDB named `HR_PDB1`.

The PDB is considered to be an application container because the current container is the application root.

```
CREATE PLUGGABLE DATABASE hr_pdb1 admin user hr_pdb1adm identified by oracle;
```

17. Open the application PDB for read/write operations and save its state.

```
ALTER PLUGGABLE DATABASE hr_pdb1 OPEN;
ALTER PLUGGABLE DATABASE hr_pdb1 SAVE STATE;
```

18. Check out the open status of the application PDB:

```
SELECT STATUS FROM DBA_PDBS WHERE PDB_NAME='HR_PDB1';
```

19. Switch the current PDB to the application PDB and sync the application with the application root.

```
ALTER SESSION SET CONTAINER=hr_pdb1;

ALTER PLUGGABLE DATABASE APPLICATION hr_app SYNC;
```

20. Connect to the application pdb as HR user and verify the tables and their data are there.

```
conn hr/oracle@//srv1:1521/hr_pdb1.localdomain
SELECT TNAME FROM TAB ;
SELECT * FROM REGIONS ;
SELECT SEQUENCE_NAME FROM USER_SEQUENCES ;
```

21. Repeat the steps in this section to create another application PDB named `HR_PDB2`

```
conn sys/oracle@hr_ac as sysdba
CREATE PLUGGABLE DATABASE hr_pdb2 admin user hr_pdb2adm identified by oracle;
ALTER PLUGGABLE DATABASE hr_pdb2 OPEN;
ALTER PLUGGABLE DATABASE hr_pdb2 SAVE STATE;
ALTER SESSION SET CONTAINER=hr_pdb2;
ALTER PLUGGABLE DATABASE APPLICATION hr_app SYNC;

conn hr/oracle@//srv1:1521/hr_pdb2.localdomain
SELECT TNAME FROM TAB ;
```

D. Test the application PDBs

In this section of the practice, you will perform some testing actions on the application PDB that you created. The target of this sectionis to have a practical understanding on how application users can use the common objects.

Using Data-linked objects

The REGIONS table is a data-linked table and you will use it to perform some tests on it.

22. Verify that you can query the REGIONS table from each application PDB.

```
conn hr/oracle@//srv1:1521/hr_pdb1.localdomain
SELECT REGION_ID, REGION_NAME FROM REGIONS;

conn hr/oracle@//srv1:1521/hr_pdb2.localdomain
SELECT REGION_ID, REGION_NAME FROM REGIONS;
```

23. Login to HR_PDB1 and try inserting a record in the REGIONS table.

Because REGIONS table is a data-linked table, application users cannot insert data into it. They can only query the table.

```
ORA-65097: DML into a data link table is outside an application action
```

```
conn hr/oracle@//srv1:1521/hr_pdb1.localdomain
INSERT INTO REGIONS (REGION_ID,REGION_NAME) VALUES (101,'TEST');
```

Using Metadata-linked objects

The DEPARTMENTS table is a metadata-linked table and you will use it to perform some tests on it.

24. Insert two rows in the DEPARTMENTS table in each PDB. Make sure each PDB cannot see the data inserted by the other PDB.

This is because the DEPARTEMENTS table is a metadata-linked table. The structure is shared among the associated PDBs but not the data.

Note: if there had been a foreign key reference between DEPARTMENTS and LOCATIONS table, the following step in the practice will return error. This is bug and can be fixed by the patch number 21955394.

```
conn hr/oracle@//srv1:1521/hr_pdb1.localdomain

INSERT INTO DEPARTMENTS (DEPARTMENT_ID, DEPARTMENT_NAME, MANAGER_ID, LOCATION_ID)
VALUES  ('10', 'MANAGEMENT', NULL, 1000);

INSERT INTO HR.DEPARTMENTS (DEPARTMENT_ID, DEPARTMENT_NAME, MANAGER_ID,
LOCATION_ID)
VALUES  ('20', 'DEVLOPEMENT', NULL, 1000);

COMMIT;
```

```
conn hr/oracle@//srv1:1521/hr_pdb2.localdomain
INSERT INTO HR.DEPARTMENTS (DEPARTMENT_ID, DEPARTMENT_NAME, MANAGER_ID,
LOCATION_ID)
VALUES  ( '100', 'MANAGEMENT', NULL, 2000);

INSERT INTO HR.DEPARTMENTS (DEPARTMENT_ID, DEPARTMENT_NAME, MANAGER_ID,
LOCATION_ID)
VALUES  ( '200', 'DEVLOPEMENT', NULL, 2000);
COMMIT;

SELECT DEPARTMENT_ID, DEPARTMENT_NAME, MANAGER_ID, LOCATION_ID
FROM DEPARTMENTS
ORDER BY 1;
```

25. Try adding a column to the `DPARTMENTS` table.

The command fails and returns the following error. Metadata-link object definitions can be changed only from the application root. Even dropping the table is not allowed.

```
ORA-65274: operation not allowed from outside an application action
```

```
ALTER TABLE DEPARTMENTS ADD ( NOTES VARCHAR2(100)) ;
```

Using Extended Data-linked objects

The `LOCATIONS` table is an extended data-linked table and you will use it to perform some tests on it.

26. Login to `hr_pdb1` as HR account and insert a row in the `LOCATIONS` table in `HR_PDB1`.

```
conn hr/oracle@//srv1:1521/hr_pdb1.localdomain

SELECT COUNT(*) FROM LOCATIONS;
INSERT INTO LOCATIONS (LOCATION_ID, CITY, COUNTRY_ID)
VALUES (9000, 'Man U.', 'UK') ;
COMMIT;
SELECT COUNT(*) FROM LOCATIONS;
```

27. Insert the same row in the same table in `HR_PDB2`.

Each application PDB can insert its own data into the extended data-linked table.

```
conn hr/oracle@//srv1:1521/hr_pdb2.localdomain

SELECT COUNT(*) FROM LOCATIONS;
INSERT INTO LOCATIONS (LOCATION_ID, CITY, COUNTRY_ID)
VALUES (9000, 'Man U.', 'UK') ;
COMMIT;
SELECT COUNT(*) FROM LOCATIONS;
```

28. Try deleting the row of `LOCATION_ID` that equals to 3200.

That row was inserted by the application installation and, therefore, it cannot be deleted from the PDB. The `DELETE` statement does not return error though.

```
SELECT COUNT(*) FROM LOCATIONS WHERE LOCATION_ID=3200;
DELETE LOCATIONS WHERE LOCATION_ID=3200;
```

29. Try deleting the row of `LOCATION_ID` that equals to 9000.

This row was inserted into the table from the PDB and, therefore, can be deleted from the PDB.

```
SELECT COUNT(*) FROM LOCATIONS WHERE LOCATION_ID=9000;
DELETE LOCATIONS WHERE LOCATION_ID=9000;
COMMIT;
```

Testing the Sequences

You can perform the following further testing on the sample application PDB:

- Test how the sequence `EMPLOYEES_SEQ` increment behaves in an application PDB. You will notice the sequence increments independently in each PDB.
- Try changing any of the sequence properties (like its `MAXVALUE`). Oracle disallows the operation because it is a metadata-linked object.

Summary

- The application, application root, and application PDB are new approach introduced by Oracle 12c Release 2 to develop and manage the applications that are used by multiple clients or branches.
- With this new approach, you create an application root, install an application in it, and then create an application PDB for each client or branch. The application contains mainly common objects.
- In this practice, you implemented the entire cycle of creating an application PDB. You also gained a practical understanding of the concepts that control the behavior of the common objects.