Practice 11: Managing Common and Local Users

Practice Overview

In this practice you will examine the principles that govern managing the common and local users in CDB and PDBs.

Specifically, you will learn:

- How to manage common users
- How to manage local users
- How to manage common and local roles
- Granting privileges as common or local
- Enabling common users to view information about specific PDBs

Practice Assumptions

- `CDB1` (in `srv1`) database is up and running.

Managing Common and local Users

A. Managing Common Users

In the following steps, you will create and examine the principles of creating and managing common users.

1. View all common users in `CDB1`.

```
sqlplus / as sysdba

set pagesize 80
col username format a30

SELECT DISTINCT USERNAME, ORACLE_MAINTAINED
FROM CDB_USERS WHERE COMMON='YES'
ORDER BY 1;
```

2. Create a common user and grant *commonly* the `CREATE SESSION` privilege to it.

```
CREATE USER C##USER1 IDENTIFIED BY oracle CONTAINER=ALL;

GRANT CREATE SESSION TO C##USER1 CONTAINER=ALL;
```

3. Connect to the CDB root as `C##USER1` user and then to `PDB2` using the same credentials.

```
connect C##USER1/oracle@cdb1
connect C##USER1/oracle@pdb2
```

4. Try creating a **local** user `LUSER1` in the root container.

   An error should be returned as you cannot create a local user in the root container.

```
connect / as sysdba

CREATE USER LUSER1 IDENTIFIED BY oracle CONTAINER=CURRENT;
```

5. Create another common user (`C##USER2`) and grant `CREATE SESSION` as a local privilege to the user.

   Granting a local privilege to a common user is not a recommended practice. It is added in this practice only for demonstration purpose.

```
CREATE USER C##USER2 IDENTIFIED BY oracle CONTAINER=ALL;

-- because CONTAINER was not set, its default value is CURRENT
GRANT CREATE SESSION TO C##USER2;
```

6. Try connecting to the root and a PDB using the new common user.

   The `CREATE SESSION` privilege was *locally* granted to the user. It takes effect only on the root.

```
CONNECT C##USER2/oracle@//srv1:1521/pdb2.localdomain

CONNECT C##USER2/oracle@//srv1:1521/CDB1.localdomain
```

7. Drop the common user `C##USER2`.

```
CONNECT / as sysdba
DROP USER C##USER2;
```

B. Managing Local Users

In the following steps, you will create examine the principles of creating and managing local users.

8. View all local users in CDB1.

```
col pdb_name format a10

SELECT U.USERNAME, P.PDB_NAME
FROM   CDB_USERS U, CDB_PDBS P
WHERE  U.CON_ID = P.CON_ID AND COMMON='NO'
ORDER BY 2,1;
```

9. Create a local user, LUSER2, in PDB2, and grant CREATE SESSION privilege to it.

```
ALTER SESSION SET CONTAINER=PDB2;

CREATE USER LUSER2 IDENTIFIED BY oracle;

GRANT CREATE SESSION TO LUSER2;
```

10. Try creating a common user in PDB2.

Common users can only be created in the root.

```
CREATE USER C##USER2 IDENTIFIED BY x CONTAINER=ALL;
```

11. As LUSER2, try connecting to PDB2, PDB3, and CDB

```
connect LUSER2/oracle@//srv1:1521/pdb2.localdomain
connect LUSER2/oracle@//srv1:1521/pdb3.localdomain
connect LUSER2/oracle@CDB1
```

C. Managing the Common and Local Roles

In the following steps, you will create examine the principles of creating and managing common and local roles and assigning them to the users.

12. List all predefined roles in the CDB (all common and local roles of the root and PDBs).

```
connect / as sysdba

col role format a30

SELECT ROLE, COMMON, CON_ID
FROM CDB_ROLES
ORDER BY ROLE, CON_ID;
```

13. List the roles defined in the root.

All the root roles are common. You cannot create local users in the root.

```
SELECT ROLE, COMMON FROM DBA_ROLES
ORDER BY ROLE;
```

14. Create a common role named `C##ROLE1`. This role will be referred to as “the common role” in the rest of the practice.

```
CREATE ROLE C##ROLE1 CONTAINER=ALL;
```

15. Try creating a local role, named as `LROLE,` in root.

The command should fail. Local roles cannot be created in the root.

```
CREATE ROLE LROLE CONTAINER=CURRENT;
```

16. List all predefined roles in `PDB2`.

```
connect system/oracle@PDB2

col role format a30
SELECT ROLE, COMMON, CON_ID FROM CDB_ROLES ORDER BY 1;
SELECT ROLE, COMMON FROM DBA_ROLES ORDER BY ROLE;
```

17. Create a local role in `PDB2`. This role will be referred to as “the local role” in the rest of the practice.

```
CREATE ROLE LROLE_PDB2 CONTAINER=CURRENT;
SELECT ROLE FROM DBA_ROLES WHERE COMMON ='NO' ORDER BY ROLE;
```

18. Grant the common role to the common user from the root.

Note that the **common** role is granted **locally** to the common user. The granted role is applicable only in the root.

```
connect / as sysdba

GRANT C##ROLE1 TO C##USER1;
```

```
-- the COMMON column in this select is not a property of the role. It is how
--  the privilege was granted
col grantee format A16
col granted_role format A16
SELECT GRANTEE, GRANTED_ROLE, COMMON, CON_ID
FROM CDB_ROLE_PRIVS WHERE GRANTEE='C##USER1';

CONNECT C##USER1/oracle

SELECT * FROM SESSION_ROLES;
```

19. Grant the common role to the common user from the root as common (to be applicable in all the containers)

```
CONNECT / as sysdba

-- specifying CONTINAER=ALL makes the grant common
GRANT C##ROLE1 TO C##USER1 CONTAINER=ALL;

-- you should see the common grants in addition to the local grant
SELECT GRANTEE, GRANTED_ROLE, COMMON, CON_ID
FROM CDB_ROLE_PRIVS WHERE GRANTEE='C##USER1';

-- connect as the common user to the root and check out the applicable role
CONNECT C##USER1/oracle
SELECT * FROM SESSION_ROLES;

-- connect as the common user to PDB2 and check out the applicable role
CONNECT C##USER1/oracle@PDB2
SELECT * FROM SESSION_ROLES;
```

20. Revoke the common role from the common user so that the role cannot be used in any container.

```
CONNECT / as sysdba
-- this revokes the common grant
REVOKE C##ROLE1 FROM C##USER1 CONTAINER=ALL;

-- the local grant is still applicable
SELECT GRANTEE, GRANTED_ROLE, COMMON, CON_ID
FROM CDB_ROLE_PRIVS WHERE GRANTEE='C##USER1';

-- revoke the local grant
REVOKE C##ROLE1 FROM C##USER1;

-- verify
SELECT GRANTEE, GRANTED_ROLE, COMMON, CON_ID
FROM CDB_ROLE_PRIVS WHERE GRANTEE='C##USER1';
```

```
-- verify again
CONNECT C##USER1/oracle
SELECT * FROM SESSION_ROLES;

CONNECT C##USER1/oracle@PDB2
SELECT * FROM SESSION_ROLES;
```

21. Grant the common role to a local user in `PDB2`.

```
CONNECT SYSTEM/oracle@PDB2

GRANT C##ROLE1 TO LUSER2;

SELECT GRANTEE, GRANTED_ROLE, COMMON, CON_ID
FROM CDB_ROLE_PRIVS WHERE GRANTEE='LUSER2';

-- Test the connection as the local user.
CONNECT LUSER2/oracle@PDB2
SELECT * FROM SESSION_ROLES;
```

22. Grant the local role to the local user in `PDB2`.

```
CONNECT SYSTEM/oracle@PDB2

GRANT LROLE_PDB2 TO LUSER2;

SELECT GRANTEE, GRANTED_ROLE, COMMON, CON_ID
FROM CDB_ROLE_PRIVS WHERE GRANTEE='LUSER2';

-- test the connection as the local user
CONNECT LUSER2/oracle@PDB2
SELECT * FROM SESSION_ROLES;
```

D. Granting the Privileges as Common or Local

In this practice, you will manage privileges granted as common or local in CDB and PDBs.

23. Check how the system privilege `CREATE SESSION` was granted to `C##USER1` and `LUSER2` users.

There is nothing called common privileges or local privileges. Privileges are neither common nor local, but they can be granted as common or as local.

```
CONNECT SYSTEM/oracle

col grantee format a18
col privilege format a14

SELECT GRANTEE, PRIVILEGE, COMMON, CON_ID
FROM   CDB_SYS_PRIVS
WHERE  GRANTEE IN ('C##USER1', 'LUSER2');


CONNECT SYSTEM/oracle@PDB2

SELECT GRANTEE, PRIVILEGE, COMMON
FROM DBA_SYS_PRIVS
WHERE GRANTEE IN ('C##USER1', 'LUSER2');
```

24. Grant the system privileges `CREATE TABLE` and `UNLIMITED TABLESPACE` to common user `C##USER1` to be applicable in any container (a privilege *commonly* granted).

```
CONNECT system/oracle

GRANT CREATE TABLE, UNLIMITED TABLESPACE TO C##USER1 CONTAINER=ALL;

col grantee format a12
col privilege format a30

SELECT GRANTEE, PRIVILEGE, COMMON, CON_ID
FROM CDB_SYS_PRIVS
WHERE GRANTEE = 'C##USER1'
ORDER BY 1,2;
```

25. Grant the system privilege `CREATE SEQUENCE` to common user `C##USER1` to be applicable in root only (a privilege *locally* granted)

```
CONNECT system/oracle

GRANT CREATE SEQUENCE TO C##USER1 CONTAINER=CURRENT;

col grantee format a12

SELECT GRANTEE, PRIVILEGE, COMMON, CON_ID
FROM CDB_SYS_PRIVS
WHERE GRANTEE = 'C##USER1' AND PRIVILEGE ='CREATE SEQUENCE';
```

26. Grant the system privilege `CREATE SYNONYM` to common user `C##USER1` to be applicable in `PDB2` only (a privilege *locally* granted)

```
CONNECT system/oracle@PDB2
GRANT CREATE SYNONYM TO C##USER1 CONTAINER=CURRENT;

col grantee format a18
SELECT GRANTEE, PRIVILEGE, COMMON, CON_ID
FROM CDB_SYS_PRIVS
WHERE GRANTEE = 'C##USER1' AND PRIVILEGE ='CREATE SYNONYM';
```

27. Grant the system privilege `UNLIMITED TABLESPACE` to local user `LUSER2` to be applicable in `PDB2` only. (a privilege *locally* granted)

```
CONNECT SYSTEM/oracle@PDB2
GRANT UNLIMITED TABLESPACE TO LUSER2;

col grantee format a18
SELECT GRANTEE, PRIVILEGE, COMMON, CON_ID
FROM CDB_SYS_PRIVS
WHERE GRANTEE = 'LUSER2';
```

E. Enabling Common Users to View Information about specific PDBs

In this section of the practice, you will manage the CONTAINER_DATA attributes of common users to control what information common users can view about PDB objects in specific PDBs.

28. Display the default (user-level) and object-specific CONTAINER_DATA attributes for container data objects.

```
CONNECT / as sysdba

column username format a10
column default_attr format a7
column owner format a6
column object_name format a11
column all_containers format a3
column container_name format a10
column con_id format 999
set pages 100
set line 200

SELECT USERNAME, DEFAULT_ATTR, OWNER, OBJECT_NAME,
ALL_CONTAINERS, CONTAINER_NAME, CON_ID
FROM CDB_CONTAINER_DATA
WHERE username NOT IN ('GSMADMIN_INTERNAL', 'APPQOSSYS', 'DBSNMP')
ORDER BY USERNAME;
```

29. Create the common user C##USER2 and commonly grant the system privileges CREATE SESSION and SET CONTAINER to it.

```
CREATE USER C##USER2 IDENTIFIED BY oracle CONTAINER=ALL;

GRANT CREATE SESSION, SET CONTAINER TO C##USER2 CONTAINER=ALL;
```

30. Grant the object privileges SELECT on V_$SESSION view to C##USER2.

V_$SESSION is the dictionary view on the V$VESSION fixed view

```
GRANT SELECT ON sys.V_$SESSION TO C##USER2 CONTAINER=ALL;
```

31. Create a second session connected to PDB2 as user SYS.

```
sqlplus sys/oracle@pdb2 as sysdba
```

32. In the first session, you should see one row for PDB2.

```
SELECT USERNAME, CON_ID FROM V_$SESSION
WHERE USERNAME IS NOT NULL AND USERNAME <> 'DBSNMP'
  AND CON_ID = ( SELECT CON_ID FROM V$PDBS WHERE NAME='PDB2');
```

33. Still in the first session, you connect as the common user C##USER2.
The common user does not see any information in V_$SESSION related to PDB2.

```
CONNECT C##USER2/oracle
SELECT USERNAME, CON_ID FROM V$SESSION
 WHERE USERNAME IS NOT NULL AND USERNAME <> 'DBSNMP';
```

34. Enable the common user C##USER2 to see information in V_$SESSION related to PDB2.

```
CONNECT / AS SYSDBA
ALTER USER C##USER2 SET CONTAINER_DATA = (CDB$ROOT, PDB2)
FOR V_$SESSION CONTAINER=CURRENT;
```

35. Connect as the common user C##USER2 and view information in V$SESSION related to PDB2.

```
CONNECT C##USER2/oracle
SELECT USERNAME, CON_ID
FROM SYS.V_$SESSION
WHERE USERNAME IS NOT NULL AND USERNAME <> 'DBSNMP';
```

36. View the attribute set for the common user C##USER2 on object CONTAINER_DATA

```
CONNECT / AS SYSDBA

column username format a20
column default_attr format a7
column owner format a10
column object_name format a15
column all_containers format a3
column container_name format a10
column con_id format 999
set pages 100
set line 200

SELECT USERNAME, DEFAULT_ATTR, OWNER, OBJECT_NAME,
ALL_CONTAINERS, CONTAINER_NAME, CON_ID
FROM CDB_CONTAINER_DATA
WHERE username = 'C##USER2'
ORDER BY OBJECT_NAME;
```

37. Clean up steps

```
CONNECT / as sysdba

DROP USER C##USER1;
DROP USER C##USER2;
DROP ROLE C##ROLE1;
ALTER SESSION SET CONTAINER=PDB2;
DROP USER LUSER2;
DROP ROLE LROLE_PDB2;
```

Summary

In this practice you have tested and examined the following basic rules that control managing common and local users, common and local roles, and granting the privileges commonly and locally.

Following are the summary points those rules:

- Common users can only be created on the root.
- Local users can only be created in a PDB.
- Common roles can only be created on the root.
- Local roles can only be created in a PDB.
- Common roles can be granted to common or local users.
- Common users can be granted common or local roles.
- A common role can be grated as common or as local (default) to a common user.
- Privileges can be granted as common (to common users) or as local.
- You can control the dictionary views that a common user can view and the PDBs that the common user can access using the `CONTAINER_DATA` attribute.

Tips

- To avoid confusion, always use the `CONTAINER` clause when you issue the `GRANT` command.
- Use common users for management and maintenance tasks.
- Create common roles for only common users.
- Create local roles for only local users.