ORACLE

7

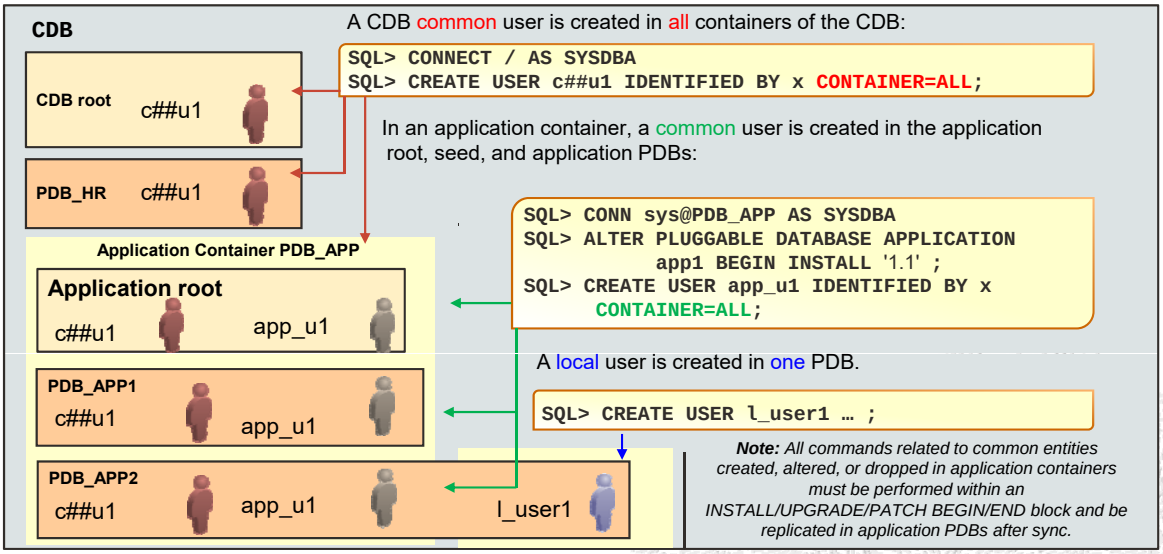
Security

Objectives

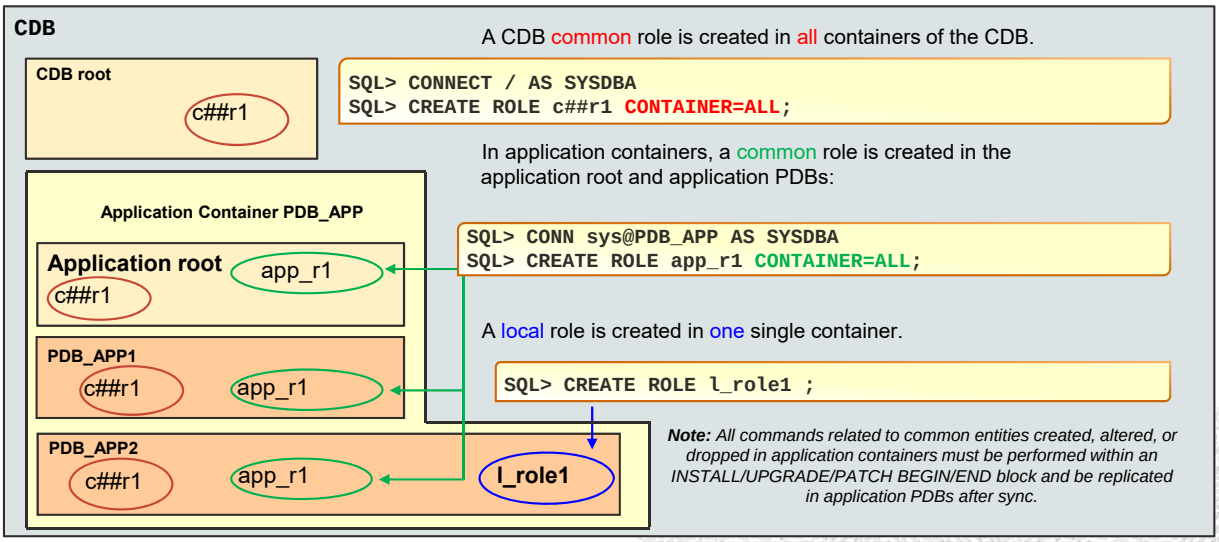
After completing this lesson, you should be able to:

- Manage common and local users, roles, privileges, and profiles in PDBs
- Manage common and local objects in application containers
- Enable common users to access data in PDBs
- Manage PDB lockdown profiles
- Audit users in CDB and PDBs
- Manage other types of policies in application containers
- Protect data with Database Vault policies in CDB and PDBs
- Encrypt data in PDBs
- Configure isolated PDB keystores
- Unplug and plug an encrypted PDB in a one-step operation
- Allow per-PDB wallets for certificates

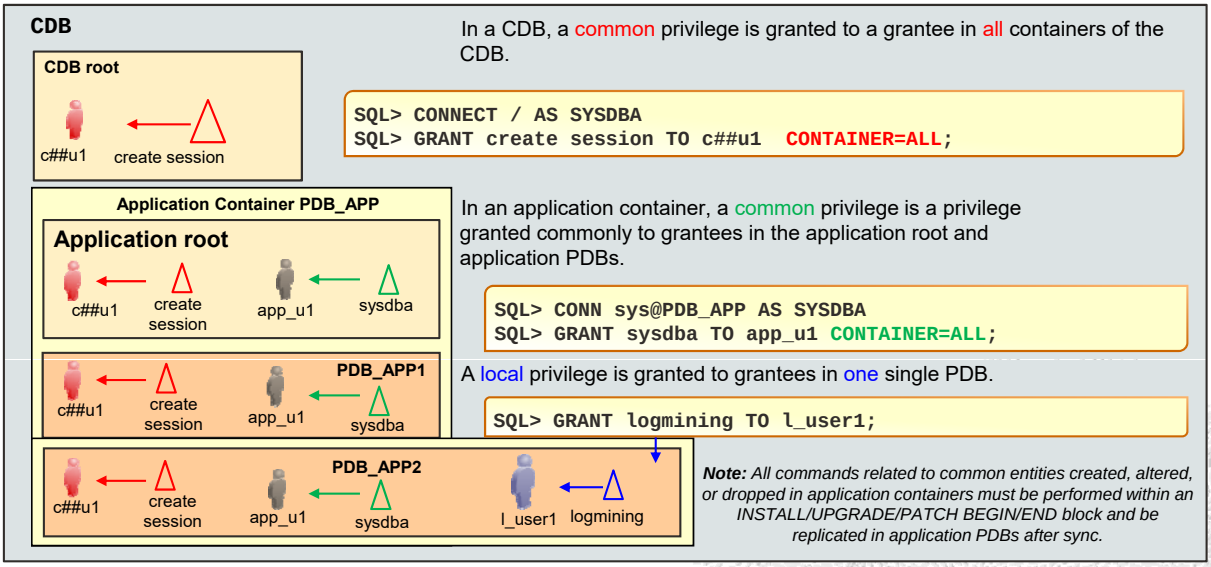
Creating Common Users in the CDB and PDBs



Creating Common Roles in the CDB and PDBs



Granting Privileges Commonly in the CDB and PDBs



In a CDB, a common privilege is granted to a grantee in all containers of the CDB.

```
SQL> CONNECT / AS SYSDBA
SQL> GRANT create session TO c##u1  CONTAINER=ALL;
```

In an application container, a common privilege is a privilege granted commonly to grantees in the application root and application PDBs.

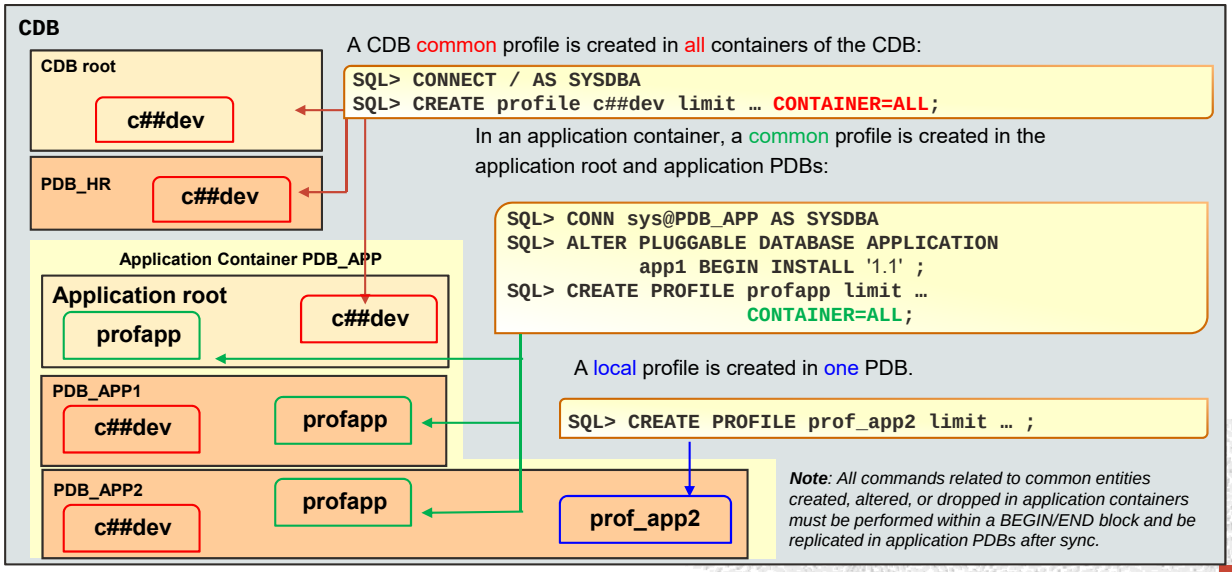
```
SQL> CONN sys@PDB_APP AS SYSDBA
SQL> GRANT sysdba TO app_u1 CONTAINER=ALL;
```

A local privilege is granted to grantees in one single PDB.

```
SQL> GRANT logmining TO l_user1;
```

Note: *All commands related to common entities created, altered, or dropped in application containers must be performed within an INSTALL/UPGRADE/PATCH BEGIN/END block and be replicated in application PDBs after sync.*

Creating Common Profiles in the CDB and PDBs



A CDB common profile is created in all containers of the CDB:

```
SQL> CONNECT / AS SYSDBA
SQL> CREATE profile c##dev limit … CONTAINER=ALL;
```

In an application container, a common profile is created in the application root and application PDBs:

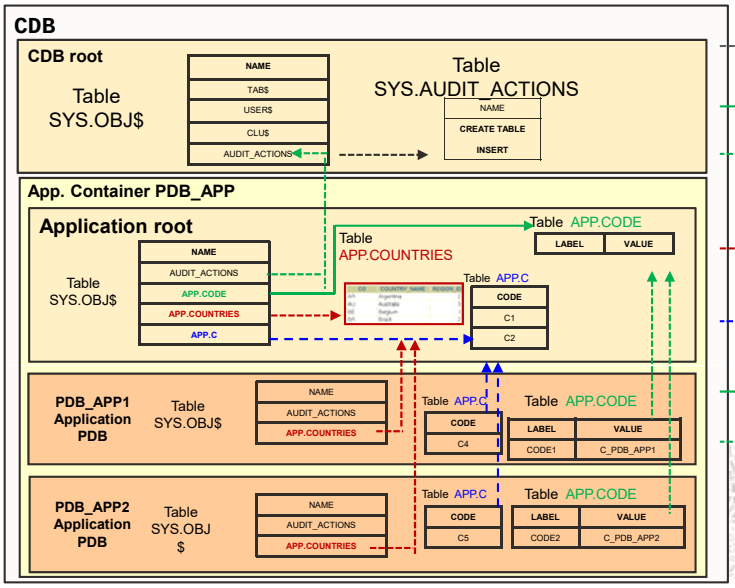
```
SQL> CONN sys@PDB_APP AS SYSDBA
SQL> ALTER PLUGGABLE DATABASE APPLICATION
           app1 BEGIN INSTALL '1.1' ;
SQL> CREATE PROFILE profapp limit …
                    CONTAINER=ALL;
```

A local profile is created in one PDB.

```
SQL> CREATE PROFILE prof_app2 limit … ;
```

Note*: All commands related to common entities created, altered, or dropped in application containers must be performed within a BEGIN/END block and be replicated in application PDBs after sync.*

Common Objects in Application Containers



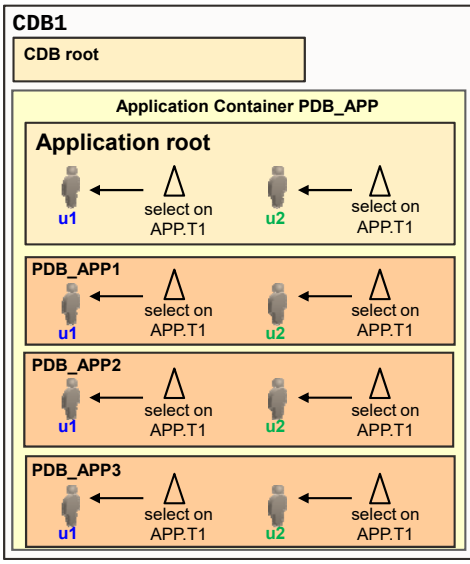
CDB level:
- A data-linked object and its data reside in the CDB root only and are shared by all PDBs.
- Metadata-linked objects store metadata about dictionary objects only in the CDB root.

Each PDB has a private data copy of an object pointing to a metadata link stored in the CDB root.

Application container level:
A data-linked object and its data reside in an application root only and are shared by all application PDBs.
An extended data-linked object combines data found in a table in an application PDB with data from a corresponding table in the application root.
A metadata-linked object stores the object definition in an application root only.
Each application PDB has a private data copy pointing to a metadata-linked object that is stored in an application root.

PDB level: A local object contains the definition and private data in the application PDB where it is created.

Enabling Common Users to Access Data in PDBs



1.Enable data access to application metadata-linked tables:

```
SQL> CONNECT sys@pdb_app AS SYSDBA
SQL> ALTER TABLE app.t1 ENABLE CONTAINER_DATA;
```

2.Enable common users to access data related to specific PDBs:

```
SQL> ALTER USER u1 SET CONTAINER_DATA =
   (PDB_APP, PDB_APP1, PDB_APP2, PDB_APP3)
     FOR app.t1 CONTAINER=CURRENT;
```

```
SQL> ALTER USER u2 SET
        CONTAINER_DATA=(PDB_APP, PDB_APP1)
        FOR app.t1 CONTAINER=CURRENT;
```

3.U1 views all rows in APP.T1:

C1	CON_ID
VAL1	3
VAL2	4
VAL3	5
VAL4	6

U2 views some rows:

C1	CON_ID
VAL1	3
VAL2	4

Finding Information About CONTAINER_DATA Attributes

Find information about the default (user-level) and object-specific CONTAINER_DATA attributes that are explicitly set to a value other than DEFAULT.

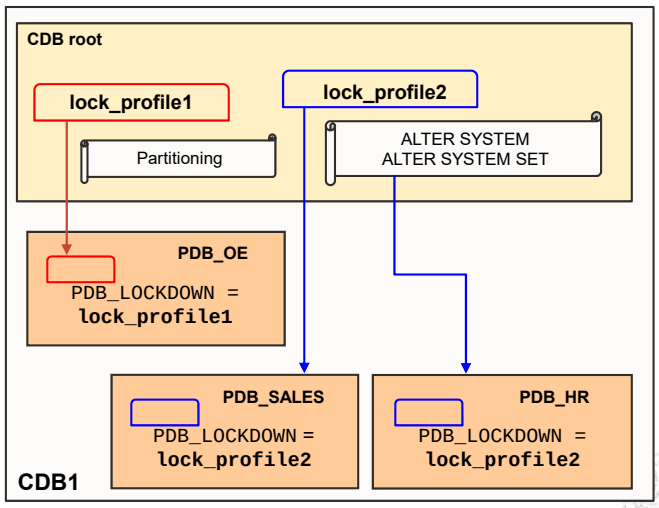
```
SQL> SELECT username, default_attr, object_name, all_containers, container_name,
           con_id
     FROM   cdb_container_data ORDER BY object_name;

USERNAME          DEFAULT OBJECT_NAME  ALL CONTAINER_  CON_ID
----------------- ------- -------------- --- ----------  ------
C##JIM            N       V$SESSION       N   PDB_HR           1
C##JIM            N       V$SESSION       N   CDB$ROOT         1
C##JIM            N       V$SESSION       N   PDB2_2           1
SYSTEM            Y                       Y                    1
DBSNMP            Y                       Y                    1
SYSBACKUP         Y                       Y                    1
SYS               Y                       Y                    1
```


Restricting Operations with PDB Lockdown Profiles

- A potential for elevation of privileges exists where identity is shared between PDBs.
- You can restrict operations, features, and options used by users connected to a given PDB by using three ALTER SYSTEM clauses.

STATEMENT	FEATURE	OPTION
ALTER SYSTEM FLUSH SHARED_POOL, CHECKPOINT, SWITCH LOGFILE, SET	NETWORK_ACCESS UTL_TCP, UTL_SMTP, UTL_HTTP, UTL_INADDR, XDB_PROTOCOLS, DBMS_DEBUG_JDWP	Partitioning
	COMMON_SCHEMA_ACCESS	Advanced Queuing
	OS_ACCESS UTL_FILE, JAVA_OS_ACCESS, EXTERNAL_PROCEDURES	Real Application Clusters
	XDB_PROTOCOLS	Oracle Data Guard
	JAVA, JAVA_RUNTIME	

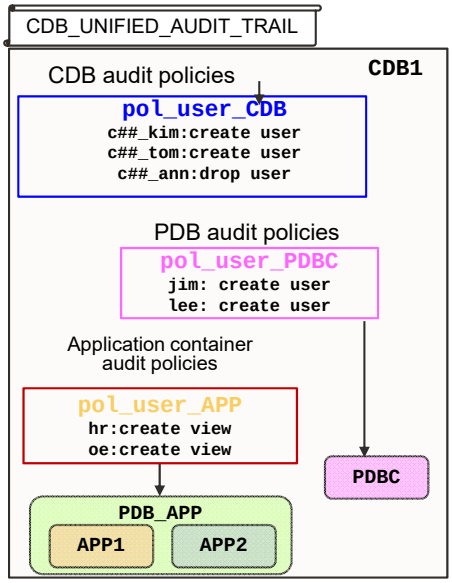
Restricting Operations in a PDB Lockdown Profile



CDB_LOCKDOWN_PROFILES

1. Create PDB lockdown profiles.
2. Define enabled and disabled:
   - Statement and clauses
   - Feature
   - Option
3. Set the PDB_LOCKDOWN parameter to a PDB lockdown profile for all PDBs.
4. Optionally set the PDB_LOCKDOWN parameter to another PDB lockdown profile for a PDB.

Auditing Actions in the CDB and PDBs



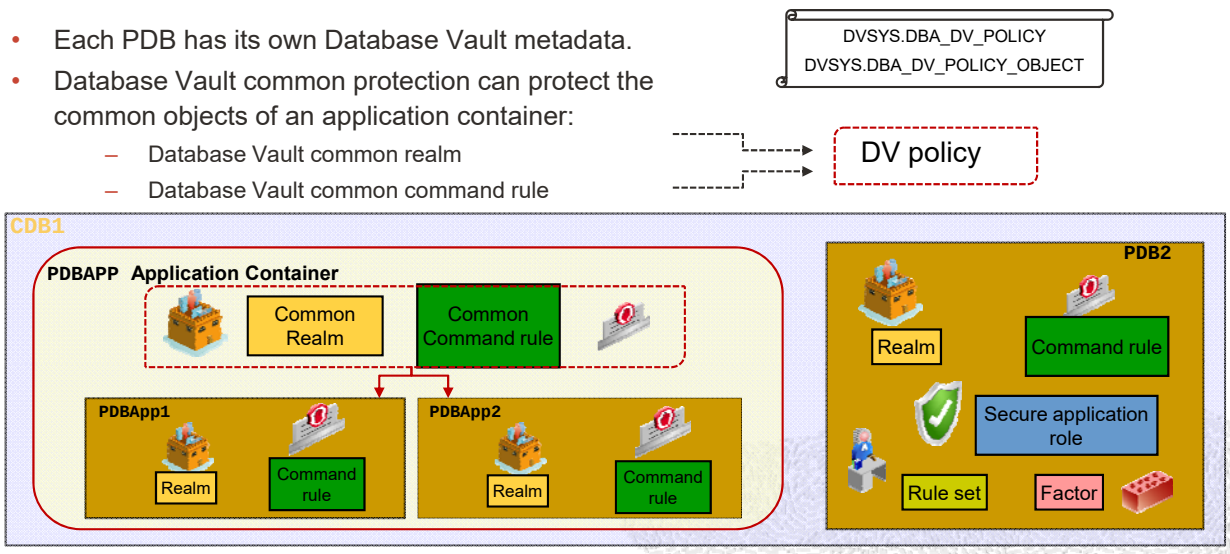
1. Connect to the CDB root or to an application root or to a regular PDB.
2. Create common or local unified audit policies:
   - For all PDBs (*connect to CDB root*)
   - For all application PDBs of an application container (*connect to the application root*)
   - For a regular PDB or a specific application PDB (*connect to the PDB*)
3. Enable/disable audit policies:
   - Define users or users being granted roles to be audited (*DBA role*)
   - Use AUDIT POLICY and NOAUDIT POLICY commands

Managing Other Types of Security Policies in Application Containers

Policy Type	Compatible in Application Containers	Created in Install / Upgrade / Patch BEGIN-END block	Automatic synchronization in application PDBs
Unified Audit	Y	Y (explicit or implicit)	Y (explicit or implicit)
FGA	Y	Y	N
Application Context & VPD	Y	Y	N
TSDP	Y	N	n/a
OLS	N	n/a	n/a

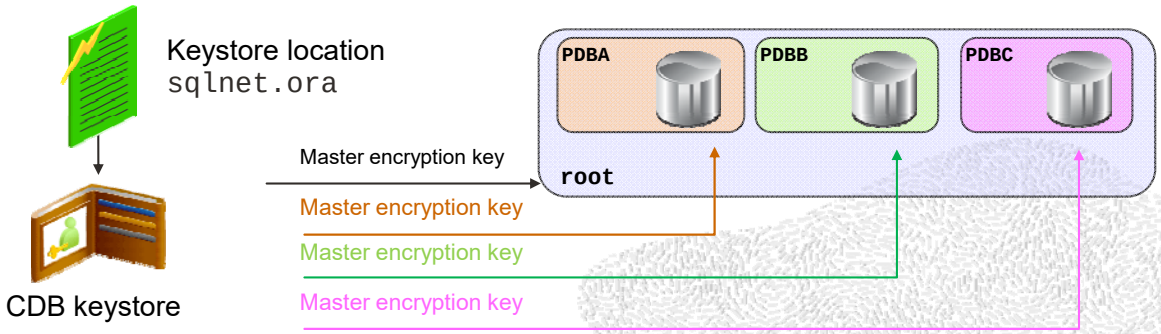
Securing Data with Oracle Database Vault

- Each PDB has its own Database Vault metadata.
- Database Vault common protection can protect the common objects of an application container:
  - Database Vault common realm
  - Database Vault common command rule



Managing Keystore in the CDB and PDBs

- There is one TDE master encryption key per PDB to encrypt PDB data.
- The TDE master encryption key must be transported from the source database keystore to the target database keystore when a PDB is moved from one host to another.



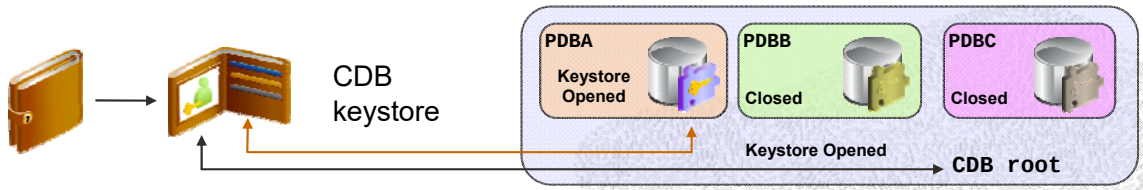
Creating and Opening a Keystore

- Create the unique keystore in the CDB root.

```
SQL> ADMINISTER KEY MANAGEMENT CREATE KEYSTORE
              'u01/app/oracle/product/19.1.0/dbhome_1/wallet'
              IDENTIFIED BY k_password;
```

- Open the keystore in the CDB root and then for a specific PDB.

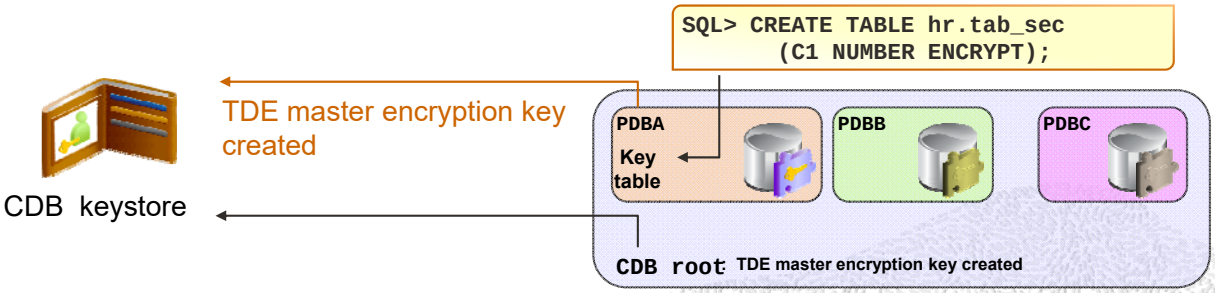
```
SQL> CONNECT john@PDBA AS SYSKM
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY k_password
               CONTAINER = CURRENT;
```



Setting TDE Master Encryption Keys

3. Set the TDE master encryption key for a PDB.

```
SQL> CONNECT john@PDBA AS SYSKM
SQL> ADMINISTER KEY MANAGEMENT SET KEY IDENTIFIED BY k_password WITH BACKUP
               CONTAINER = CURRENT;
```



You can now encrypt data in tablespaces and tables.

Migrating a PDB Between Keystore Types

To migrate a PDB from using wallet as the keystore to using Oracle Key Vault if the PDB is running in isolated mode:

1. Upload the TDE encryption keys from the isolated keystore to Oracle Key Vault by using a utility.
2. Set the `TDE_CONFIGURATION` parameter of the PDB to the appropriate value:

```
SQL> ALTER SYSTEM SET tde_configuration = 'KEYSTORE_CONFIGURATION=OKV';
```

Unplugging and Plugging a PDB with Encrypted Data

- Unplugging an encrypted PDB exports the master encryption key of the PDB.

```
SQL> ALTER PLUGGABLE DATABASE pdb1
          UNPLUG INTO '/tmp/pdb1.xml'
          ENCRYPT USING "tpwd1";
```

PDB wallet opened

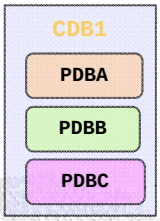
- Plugging the encrypted PDB imports the master encryption key of the PDB into the CDB keystore.

```
SQL> CREATE PLUGGABLE DATABASE pdb1
           USING '/tmp/pdb1.xml'
           KEYSTORE IDENTIFIED BY keystore_pwd1
           DECRYPT USING "tpwd1";
```

Target CDB wallet opened

Per-PDB Wallet for PDB Certificates

- There is only one `sqlnet.ora` file and one `WALLET_LOCATION` parameter per CDB.
- Each PDB has its own keystore to store the TLS credentials and identity to communicate with other PDBs.



Summary

In this lesson, you should have learned how to:

- Manage common and local users, roles, privileges, and profiles in PDBs
- Manage common and local objects in application containers
- Enable common users to access data in PDBs
- Manage PDB lockdown profiles
- Audit users in CDB and PDBs
- Manage other types of policies in application containers
- Protect data with Database Vault policies in CDB and PDBs
- Encrypt data in PDBs
- Configure isolated PDB keystores
- Unplug and plug an encrypted PDB in a one-step operation
- Allow per-PDB wallets for certificates

Practice 7: Overview

- 7-1: Managing common and local users, privileges, and roles
- 7-2: Managing common and local objects in application containers
- 7-3: Enabling common users to view information about PDB objects
- 7-4: Applying recorded statements in application PDBs
- 7-5: Managing PDB lockdown profiles
- 7-6: Auditing operations in PDBs
- 7-7: Managing PDB keystores
- 7-8: Unplugging and plugging encrypted PDBs