Recovery and Flashback

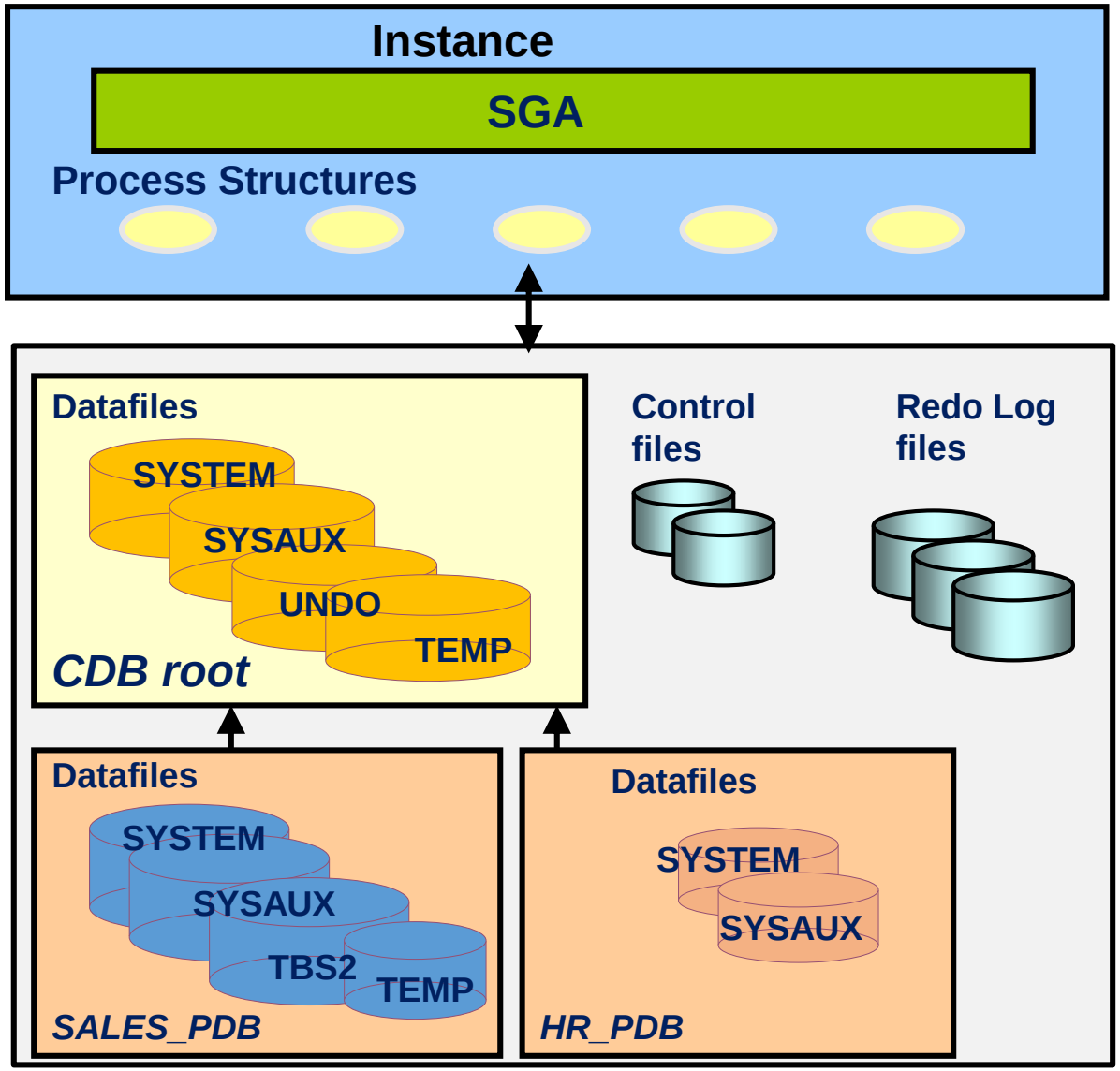
Objectives

- After completing this lesson, you should be able to:
  - Recover a PDB from essential file damage
  - Recover a PDB from nonessential file damage
  - Reuse preplugin backups after conversion of a non-CDB to a PDB
  - Reuse preplugin backups after plugging/relocating a PDB into another CDB
  - Perform CDB flashback
  - Perform PDB flashback
  - Use clean restore points to complete PDB flashback
  - Manage PDB snapshots
  - Switch over a refreshable cloned PDB

Goals

- Recover CDB or PDBs:
  - Instance failure: CDB level
  - Complete media recovery:
    - CDB or PDB tempfile
    - Controlfile / redo log file / CDB root essential datafile: CDB mounted
    - PDB datafile: PDB opened if nonessential datafile / PDB mounted if essential datafile
  - Incomplete media recovery: CDB mounted or PDB closed
  - Flashback database: CDB mounted or PDB closed

Instance Failure and Instance Recovery



PDB instance recovery is **impossible**.

After instance failure:

- Connect to the CDB root.
- Open the CDB root.
- Open all PDBs.

```
SQL> STARTUP
SQL> ALTER PLUGGABLE DATABASE ALL OPEN;
```

NOARCHIVELOG Mode

- If the database is in NOARCHIVELOG mode, and a datafile is lost, perform the following tasks:
  - Shut down the instance if it is not already down.
  - Restore the entire CDB including all datafiles and control files.
  - Start up the instance and open the CDB and all PDBs.
- Users must reenter all changes made since the last backup.

PDB Tempfile Recovery

- SQL statements that require temporary space to execute may fail if one of the tempfiles is missing.

```
SQL> CONNECT local_user@HR_PDB
SQL> select * from my_table order by 1,2,3,4,5,6,7,8,9,10,11,12,13;
select * from my_table order by 1,2,3,4,5,6,7,8,9,10,11,12,13
              *
ERROR at line 1:
ORA-01565: error in identifying file
'/u01/app/oracle/oradata/CDB1/HR_PDB/temp2_01.dbf'
ORA-27037: unable to obtain file status
Linux Error: 2: No such file or directory
```

- Automatic re-creation of temporary files at PDB opening
- Manual re-creation also possible

PDB SYSTEM or UNDO Tablespace Recovery

The CDB and all other PDBs can be left opened.

1. Connect to the PDB.
2. "Shutdown abort" the PDB if it is not automatically done.

```
$ sqlplus sys@sales_pdb as sysdba
SQL> SHUTDOWN ABORT
```

or

```
SQL> ALTER PLUGGABLE DATABASE CLOSE ABORT;
```

3. Restore and recover the PDB or the missing tablespace or the damaged datafile:

```
$ rman target sys@sales_pdb
RMAN> RESTORE DATABASE;
RMAN> RECOVER DATABASE;
RMAN> ALTER PLUGGABLE DATABASE sales_pdb OPEN;
```

PDB non-SYSTEM Tablespace Recovery

- Similar to non-CDBs: Perform the recovery within the PDB
  - Connect to the PDB.
  - Put the tablespace OFFLINE.
  - Other PDBs are not impacted.

```
SQL> CONNECT system@sales_pdb
SQL> ALTER TABLESPACE tbs2 OFFLINE IMMEDIATE;
RMAN> CONNECT TARGET /
RMAN> RESTORE TABLESPACE sales_pdb:tbs2;
RMAN> RECOVER TABLESPACE sales_pdb:tbs2;
SQL> ALTER TABLESPACE tbs2 ONLINE;
```

Note: You can also use the `REPAIR` command.

PITR

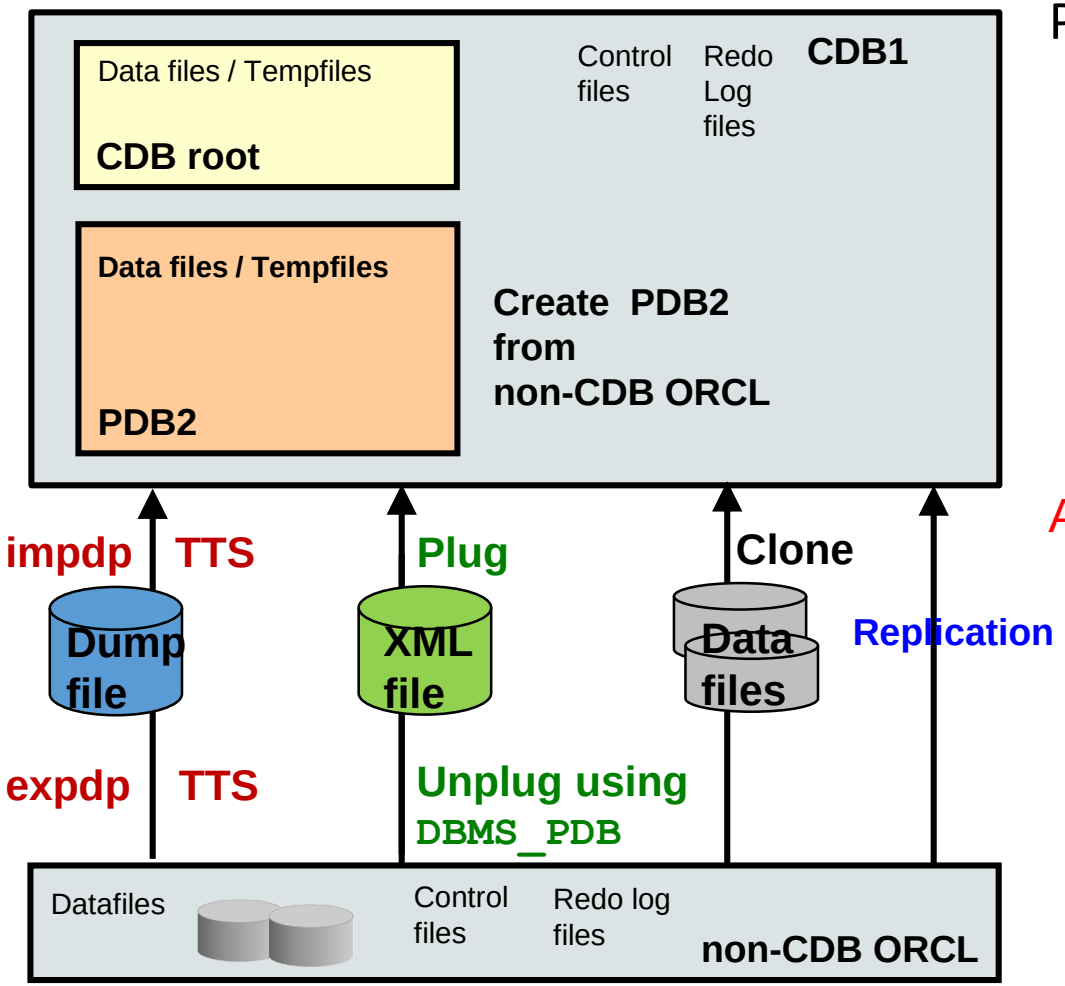
- PDB PITR

```
RMAN> ALTER PLUGGABLE DATABASE pdb1 CLOSE;
RMAN> RUN {
        SET UNTIL SCN = 1851648 ;
        RESTORE pluggable DATABASE pdb1;
        RECOVER pluggable DATABASE pdb1
         AUXILIARY DESTINATION='/u01/app/oracle/oradata';
        ALTER PLUGGABLE DATABASE pdb1 OPEN RESETLOGS;
           }
```

- PDB Tablespace PITR

```
RMAN> RECOVER TABLESPACE pdb1:test_tbs
                UNTIL SCN 832972
                AUXILIARY DESTINATION '/tmp/CDB1/reco';
RMAN> ALTER TABLESPACE pdb1:test_tbs ONLINE;
```

Migrating a Non-CDB to a CDB



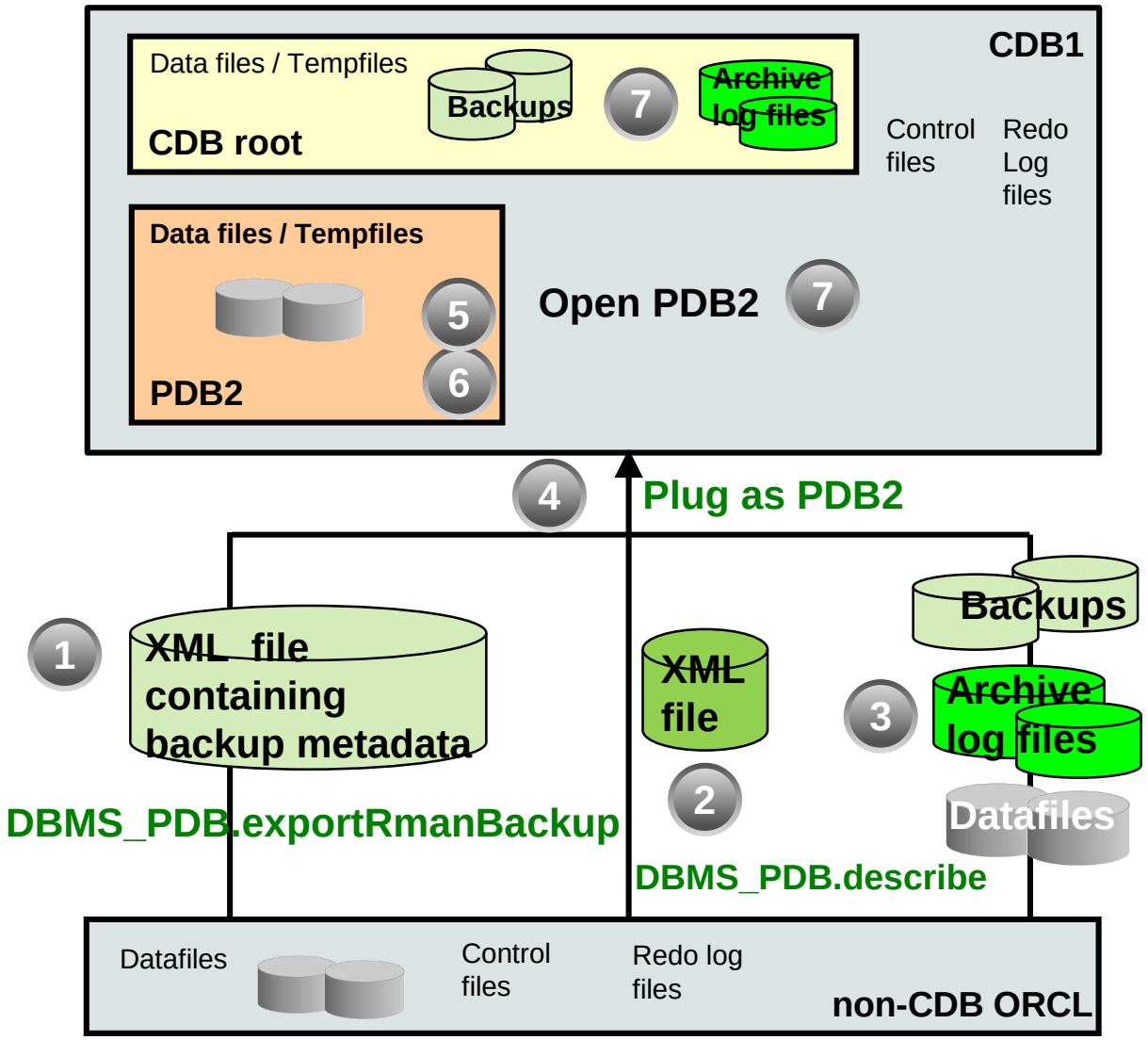
Possible methods:

- Data Pump (TTS or TDB or full export/import)
- Plugging (XML file definition with `DBMS_PDB`)
- Cloning
- Replication

After conversion:

- Is it possible to recover the PDB back in time before the non-CDB was converted?
- Are the non-CDB backups transported with the non-CDB?

Migrating a Non-CDB and Transporting Non-CDB Backups to a CDB

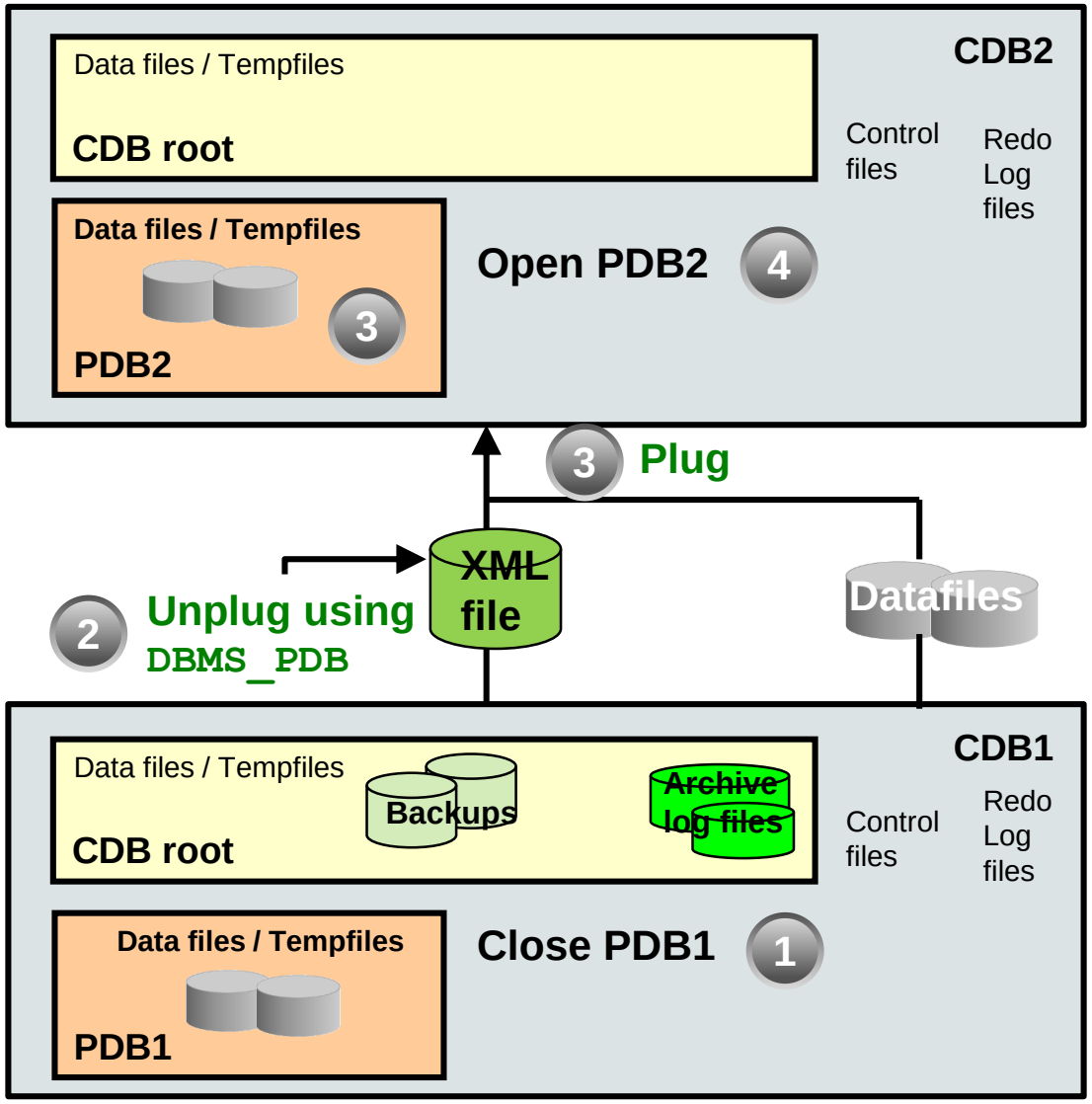


1. Export backups metadata with `DBMS_PDB.exportRmanBackup`.
2. Unplug the non-CDB by using `DBMS_PDB.describe`.
3. Archive the current redo log file.
4. Transfer datafiles including backups to the target CDB.
5. Plug using the XML file.
6. Execute the `noncdb_to_pdb.sql` script.
7. Open PDB. This automatically imports backups metadata into the CDB dictionary.

Restore/recover the PDB with preplugin backups:

1. Catalog the archived redo log file.
2. Restore PDB by using preplugin backups.
3. Recover PDB by using preplugin backups.

Relocating/Plugging a PDB into Another CDB

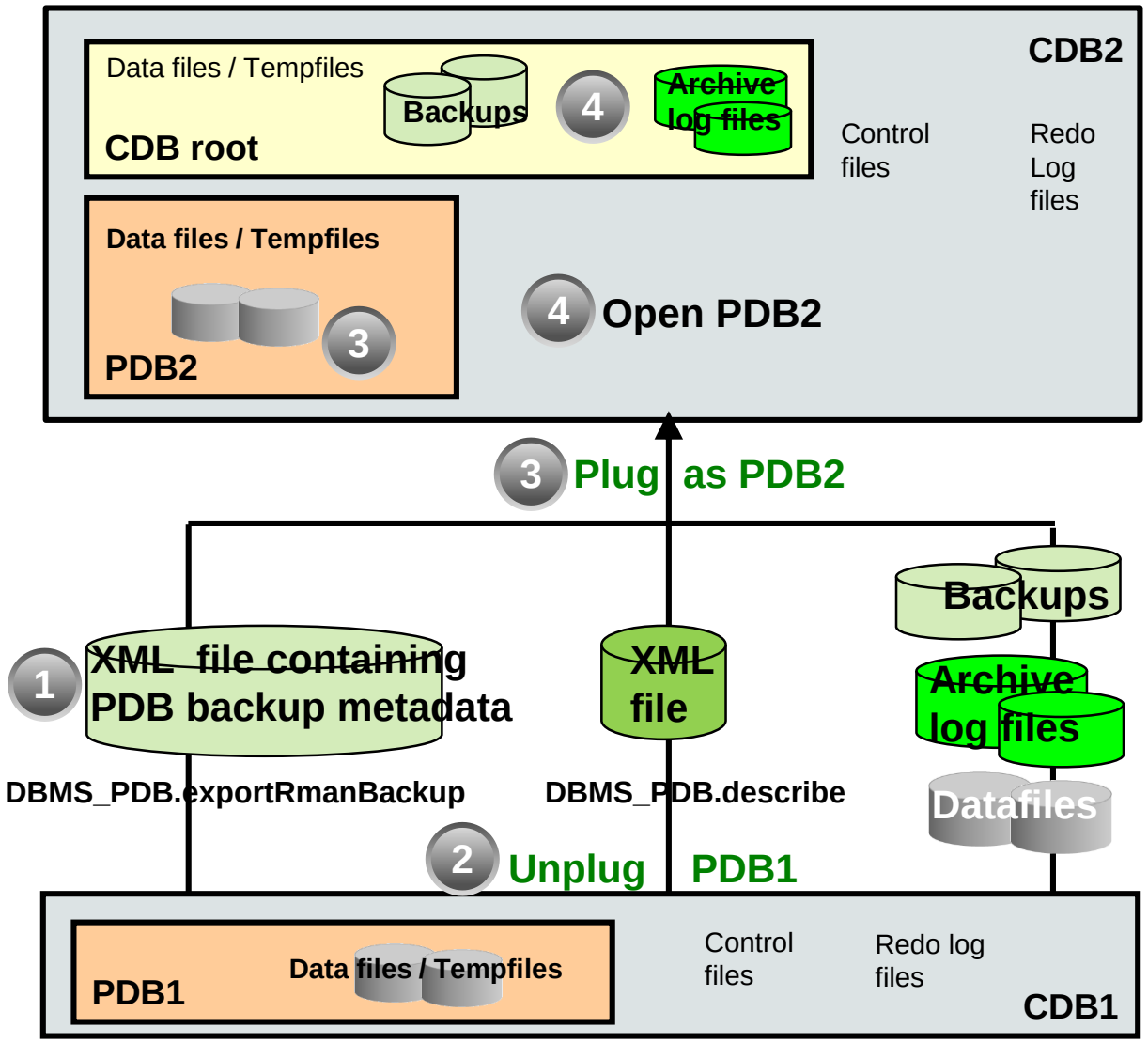


After relocating/plugging the PDB into another CDB:

- Is it possible to recover the PDB back in time before it was relocated/unplugged?
- Are the PDB backups transported with the relocated/unplugged PDB?

Plugging a PDB and Transporting PDB Backups to a CDB - 1

19c

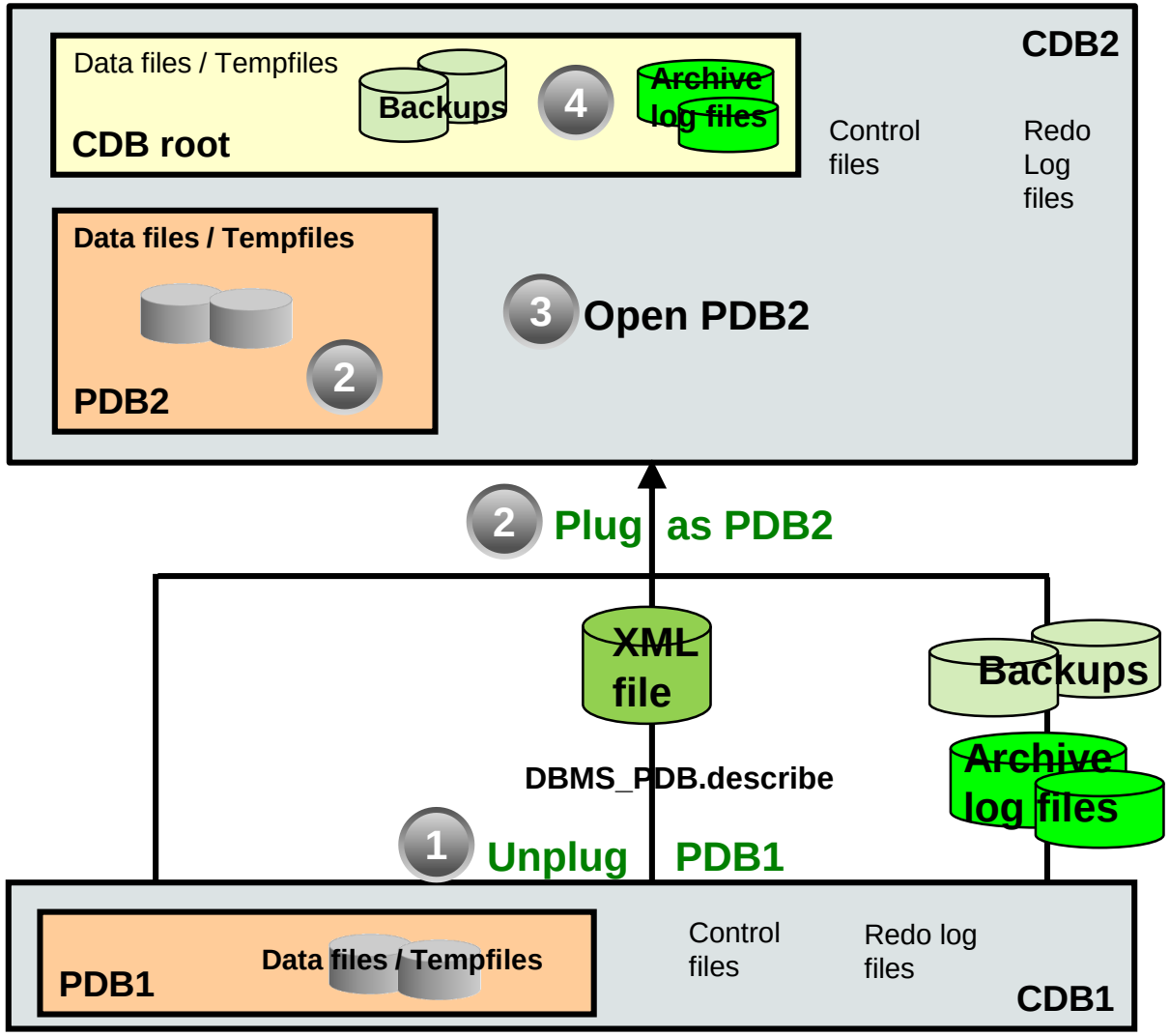


1. Export backups metadata by using `DBMS_PDB.exportRmanBackup`.
2. Unplug the PDB by using `DBMS_PDB.describe`.
3. Transfer the datafiles including backups to the target CDB.
4. Plug using the XML file.
5. Open PDB. This automatically imports backups metadata into the CDB dictionary.

Then you can restore/recover the PDB by using the transported backups:

1. Restore PDB by using preplugin backups.
2. Recover PDB by using preplugin backups.

Plugging a PDB and Transporting PDB Backups



1. Unplug the PDB with `DBMS_PDB.describe`.
2. Transfer the datafiles including backups to the target CDB.
3. Plug using the XML file.
4. Open PDB.
5. Catalog preplugin backups into CDB.

Then you can restore/recover the PDB using the transported backups:

1. Restore PDB by using preplugin backups.
2. Recover PDB by using preplugin backups.

Using PrePlugin Backups

Use the `PrePlugin` option to perform RMAN operations using preplugin backups.

- Restore a PDB from its preplugin backups cataloged in the target CDB.

```
RMAN> RESTORE PLUGGABLE DATABASE pdb_noncdb FROM PREPLUGIN;
```

- Recover a PDB from its preplugin backups until the datafile was plugged in.

```
RMAN> RECOVER PLUGGABLE DATABASE pdb_noncdb FROM PREPLUGIN;
```

- Check whether preplugin backups and archive log files are cataloged in the target CDB.

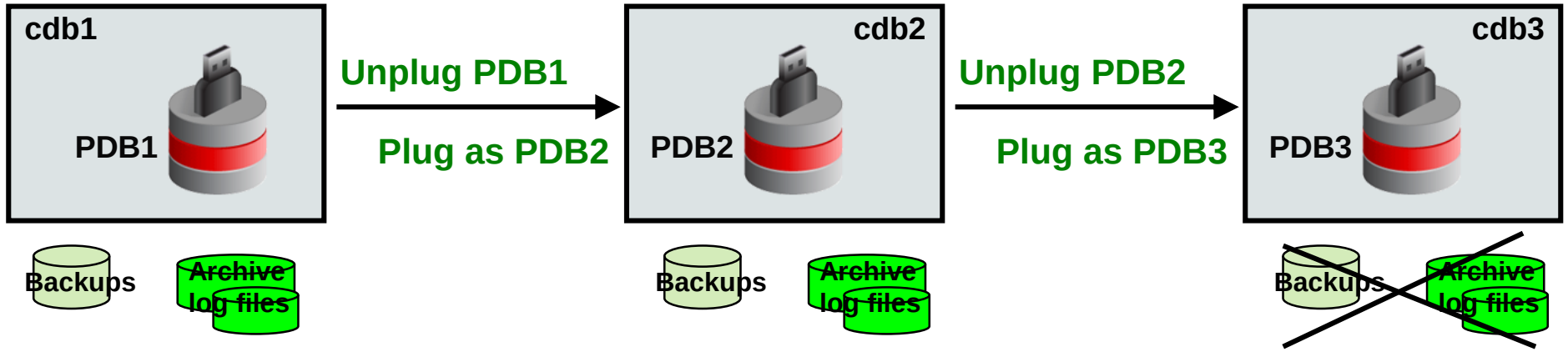
```
RMAN> SET PREPLUGIN CONTAINER pdb1;
RMAN> LIST PREPLUGIN BACKUP;
RMAN> LIST PREPLUGIN ARCHIVELOG ALL;
RMAN> LIST PREPLUGIN COPY;
```

- Verify that cataloged preplugin backups are available on disk.

```
RMAN> CROSSCHECK PREPLUGIN BACKUP;
RMAN> DELETE PREPLUGIN BACKUP;
```

To Be Aware Of

- The source and destination CDBs must have `COMPATIBLE` set to 18.1 or higher to create/restore/recover preplugin backups.
- In case of plugging in a non-CDB, the non-CDB must use `ARCHIVELOG` mode.
- The target CDB does not manage preplugin backups.
  - Use `CROSSCHECK` and `DELETE` commands to manage the preplugin backups.
- A `RESTORE` using preplugin backups can restore datafiles from one PDB only.
- Backups taken by the source `cdb1` are visible in target `cdb2` only.



Example

```
RMAN> SET PREPLUGIN CONTAINER pdb1;
RMAN> CATALOG PREPLUGIN ARCHIVELOG '/u03/app/…/o1_mf_1_8_dnqwm59v_.arc';
RMAN> RUN { RESTORE PLUGGABLE DATABASE pdb1 FROM PREPLUGIN;
            RECOVER PLUGGABLE DATABASE pdb1 FROM PREPLUGIN;
          }
RMAN> RECOVER PLUGGABLE DATABASE pdb1;
```

CDB and PDB Flashback

1. **Configure the FRA.**
2. **Set the retention target.**
3. **Enable Flashback Database.**

```
SQL> STARTUP MOUNT
SQL> ALTER DATABASE ATCHIVELOG;
SQL> ALTER DATABASE OPEN;
SQL> ALTER SYSTEM SET DB_FLASHBACK_RETENTION_TARGET=2880 SCOPE=BOTH;
SQL> ALTER DATABASE FLASHBACK ON;
SQL> ALTER DATABASE OPEN;
```

- No flashback of CDB root without flashing back the whole CDB
- PDB flashback similar to CDB flashback

```
RMAN> CONN sys@pdb1
RMAN> ALTER PLUGGABLE DATABASE CLOSE;
RMAN> FLASHBACK PLUGGABLE DATABASE pdb1 TO SCN 411010;
RMAN> ALTER PLUGGABLE DATABASE pdb1 OPEN RESETLOGS;
```

PDB Flashback and Clean Restore Point

- Clean PDB restore points can be created after a PDB is closed and ONLY in shared undo mode.
- The benefits of clean PDB restore points include:
  - Faster than other types of PDB flashback
    - No restore of any backup
    - No clone instance created
  - No need to take a new backup

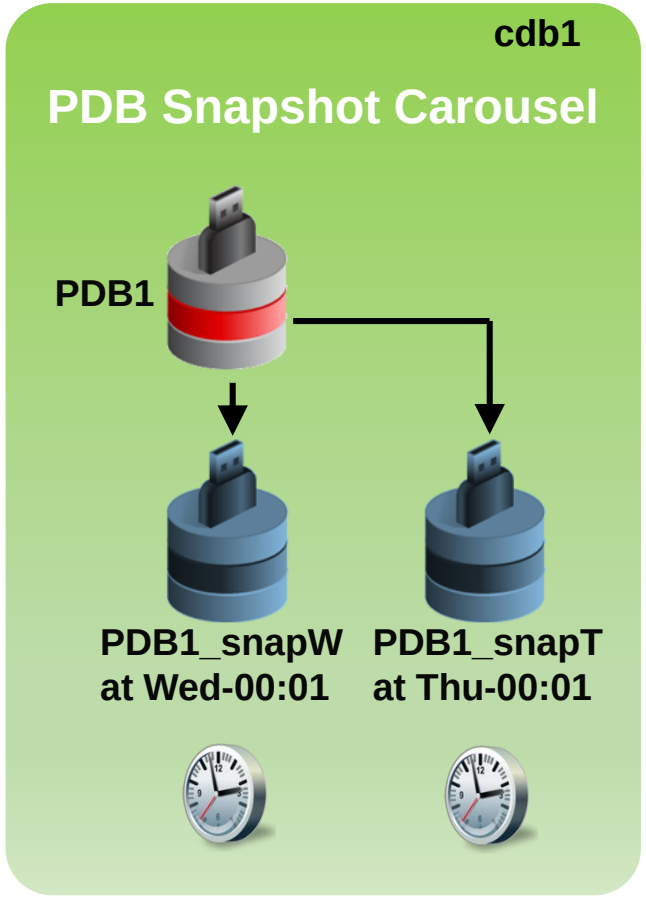
V$RESTORE_POINT

```
SQL> CONNECT / AS SYSDBA
SQL> ALTER PLUGGABLE DATABASE pdb1 CLOSE;
SQL> CREATE CLEAN RESTORE POINT start_step1 FOR PLUGGABLE DATABASE pdb1
                                            GUARANTEE FLASHBACK DATABASE;
SQL> ALTER PLUGGABLE DATABASE pdb1 OPEN;
SQL> @script_patch_step1
SQL> ALTER PLUGGABLE DATABASE pdb1 CLOSE;
```

```
$ rman target /
RMAN> FLASHBACK PLUGGABLE DATABASE pdb1 TO RESTORE POINT start_step1;
RMAN> ALTER PLUGGABLE DATABASE pdb1 OPEN RESETLOGS;
```

PDB Snapshot Carousel

- A PDB snapshot is a named copy of a PDB at a specific point in time.



- Recovery extended beyond flashback retention period
- Reporting on historical data kept in snapshots
- Storage-efficient snapshot clones taken on periodic basis
- Maximum of eight snapshots for CDB and each PDB

Example:

On Friday, need to recover back to Wednesday.

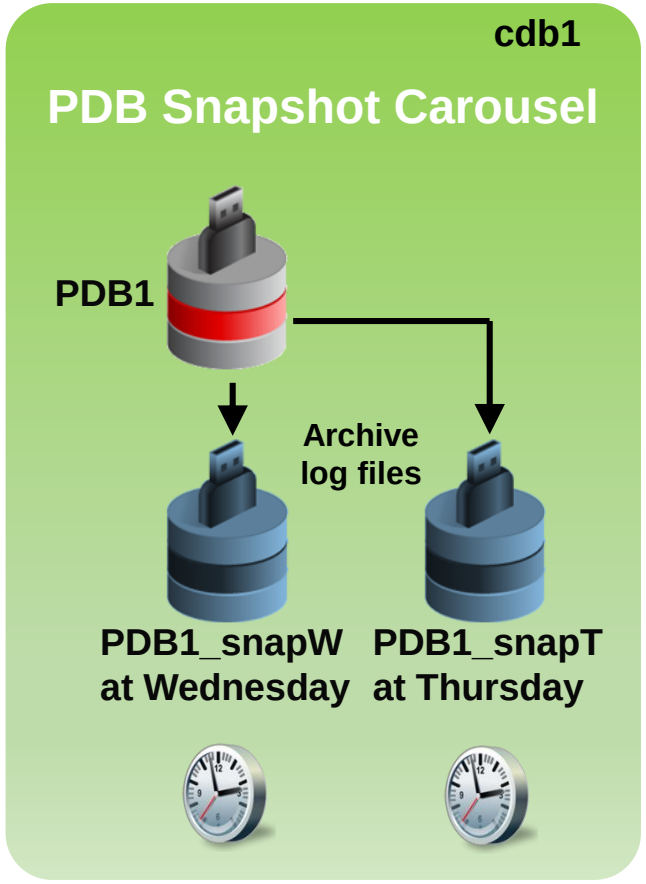
- Restore `PDB1_snapW`.

Creating PDB Snapshot

```
DATABASE_PROPERTIES
  PROPERTY_NAME = MAX_PDB_SNAPSHOTS
  PROPERTY_VALUE  = 8
```

```
DBA_PDB_SNAPSHOTS
DBA_PDBS
    SNAPSHOT_MODE
```

- To create PDB snapshots for a PDB:



1. Enable a PDB for PDB snapshots.

```
SQL> CREATE PLUGGABLE DATABASE pdb1 …
                SNAPSHOT MODE MANUAL;
```

```
SQL> ALTER PLUGGABLE DATABASE pdb1
                SNAPSHOT MODE EVERY 24 HOURS;
```

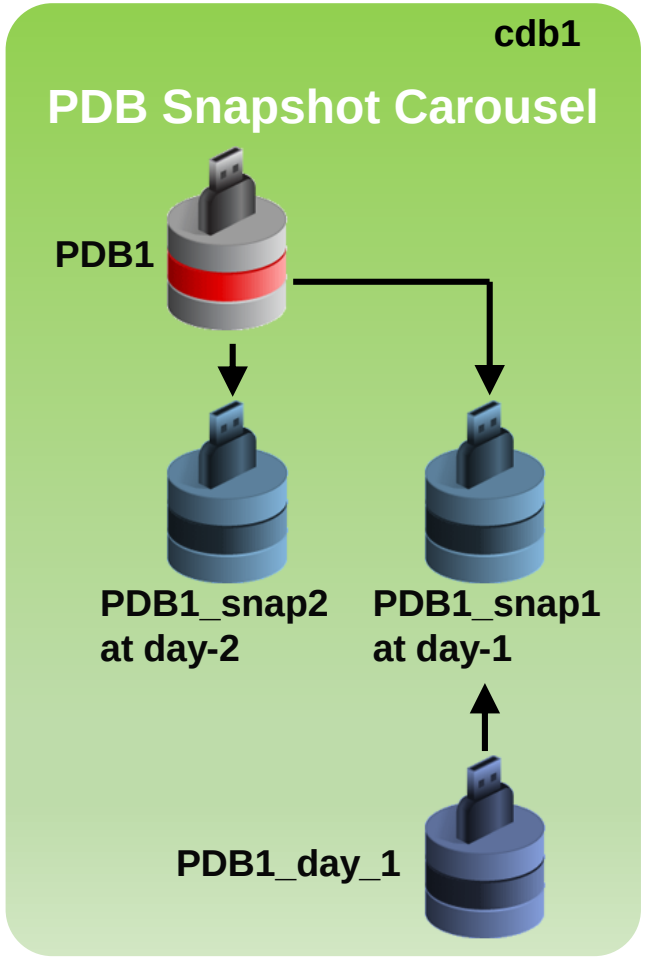
2. You can create multiple manual PDB snapshots of a PDB.

```
SQL> ALTER PLUGGABLE DATABASE pdb1
                SNAPSHOT pdb1_first_snap;
SQL> ALTER PLUGGABLE DATABASE pdb1
                SNAPSHOT pdb1_second_snap;
```

3. Disable snapshot creation for a PDB.

```
SQL> ALTER PLUGGABLE DATABASE pdb1 SNAPSHOT MODE NONE;
```

Creating PDBs Using PDB Snapshots



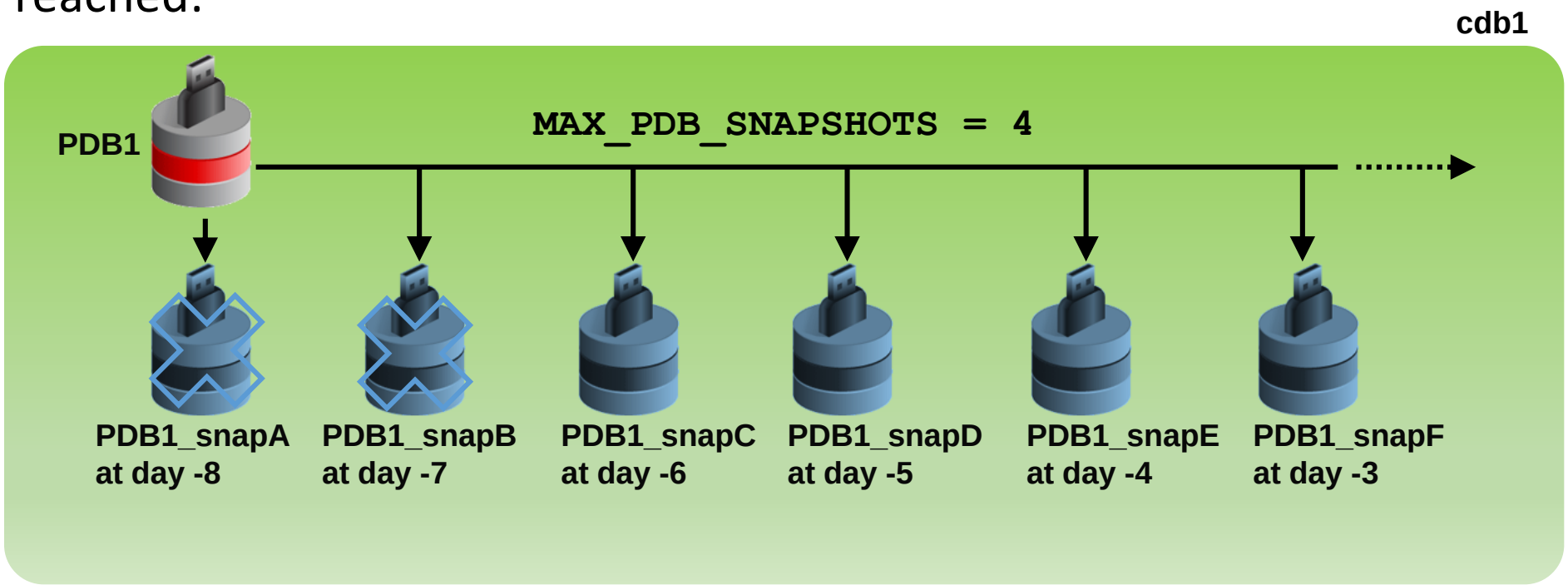
After a PDB snapshot is created, you can create a new PDB from it:

```
SQL> CREATE PLUGGABLE DATABASE pdb1_day_1 FROM pdb1
             USING SNAPSHOT <snapshot_name>;
```

```
SQL> CREATE PLUGGABLE DATABASE pdb1_day_2 FROM pdb1
             USING SNAPSHOT AT SCN <snapshot_SCN>;
```

Dropping PDB Snapshots

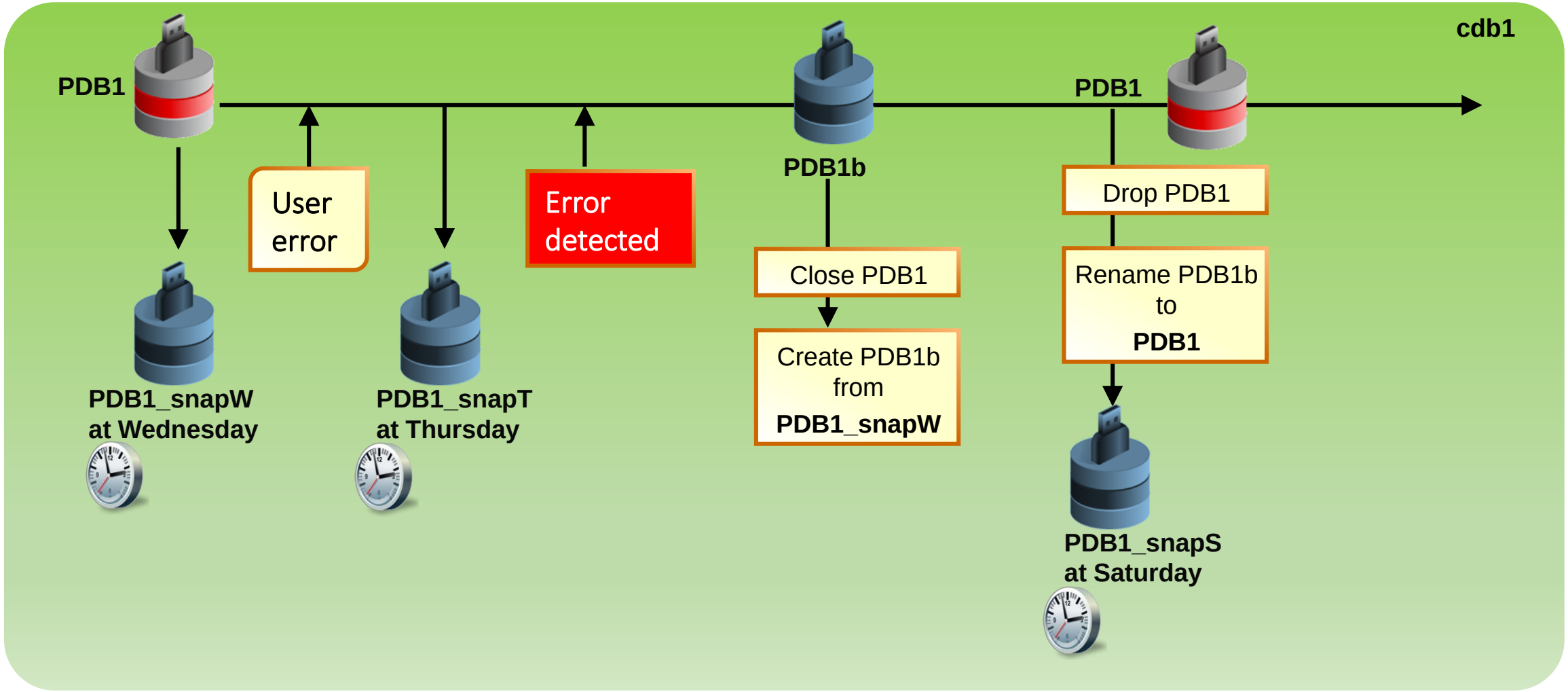
- Automatic PDB snapshot deletion when `MAX_PDB_SNAPSHOTS` limit is reached:



- Manual PDB snapshot deletion:

```
SQL> ALTER PLUGGABLE DATABASE pdb1 DROP SNAPSHOT pdb1_first_snap;
```

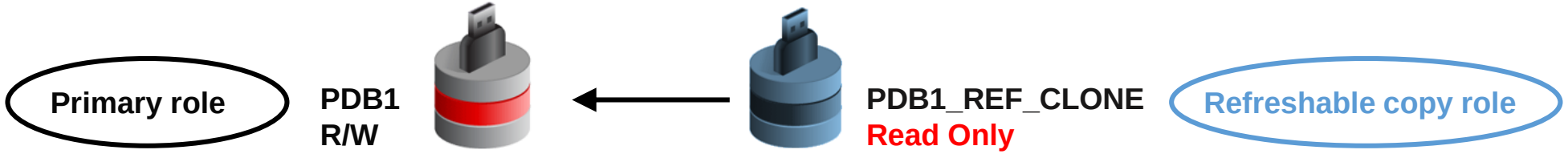
Flashbacking PDBs Using PDB Snapshots



Switching Over a Refreshable Cloned PDB

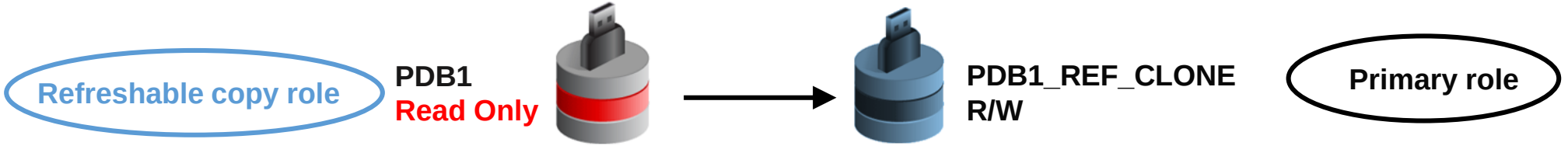
- Switchover at the PDB level:

1. A user creates a refreshable clone of a PDB.



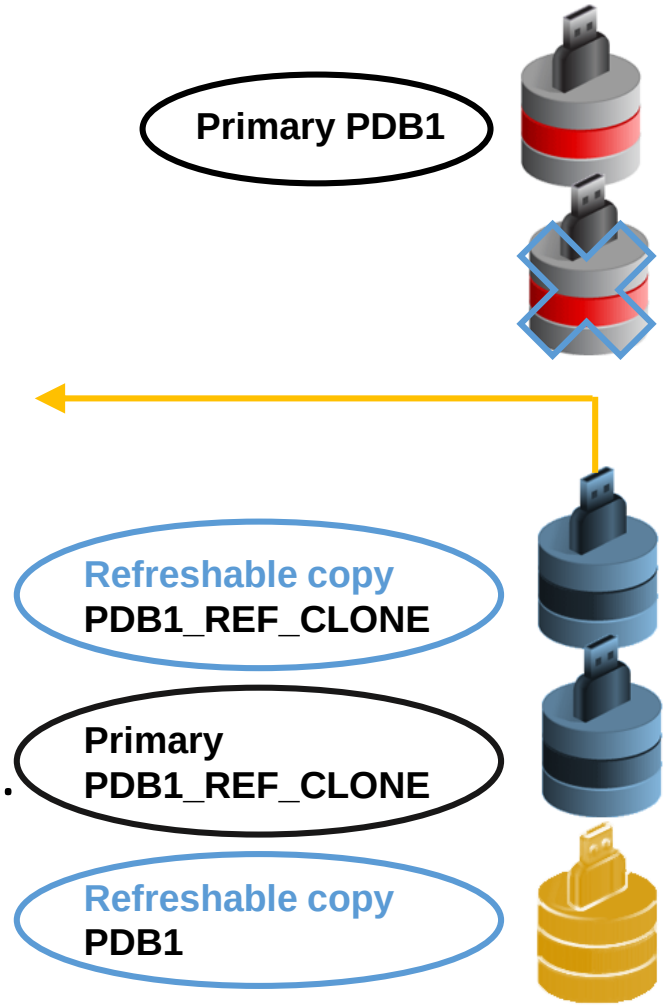
2. The roles can be reversed: the refreshable clone can be made the primary PDB.
   - The new primary PDB can be opened in read/write mode.
   - The primary PDB becomes the refreshable clone.

```
SQL> CONNECT sys@PDB1 AS SYSDBA
SQL> ALTER PLUGGABLE DATABASE REFRESH MODE EVERY 6 HOURS
            FROM pdb1_ref_clone@link_cdb_source_for_clone SWITCHOVER;
```



Unplanned Switchover

- When a PDB with an associated refreshable clone encounters an issue, complete an unplanned switchover:
  1. Close the primary PDB.
  2. Archive the current redo log file.
  3. Drop the primary PDB.
  4. Copy the archive redo log files to a new folder.
  5. Set the destination for the archive redo log files.
  6. Refresh the refreshable clone PDB.
  7. Disable the refresh mode of the refreshable clone PDB.
  8. Open the refreshed PDB that became the new primary PDB.
  9. Optionally, create a new refreshable clone.



Summary

- In this lesson, you should have learned how to:
  - Recover a PDB from essential file damage
  - Recover a PDB from nonessential file damage
  - Reuse preplugin backups after conversion of a non-CDB to a PDB
  - Reuse preplugin backups after plugging/relocating a PDB into another CDB
  - Perform CDB flashback
  - Perform PDB flashback
  - Use clean restore points to complete PDB flashback
  - Manage PDB snapshots
  - Switch over a refreshable cloned PDB

Practice 9: Overview

- 9-1: RMAN recovery from SYSTEM PDB datafile loss
- 9-2: RMAN recovery from nonessential PDB datafile loss
- 9-3: PDB PITR
- 9-4: Recovering a plugged non-CDB by using preplugin backups
- 9-5: Recovering a plugged PDB by using preplugin backups
- 9-6: Flashing back an application upgrade by using restore points
- 9-7: Managing and using PDB snapshots
- 9-8: Switching over refreshable cloned PDBs