Exploring Oracle Database 12c Multitenant Best Practices for your Cloud

Ami Aharonovich

Oracle ACE & OCP

Ami@DBAces.com

About Me

- Oracle ACE

ORACLE ACE

- Oracle Certified Professional DBA (OCP)

ORACLE Certified Professional

- Founder and CEO, DBAces
- President, Israel Oracle User Group

www.iloug.org.il ilOUG Israel Oracle User Group ארגון משתמשי אורקל בישראל

- Oracle DBA consultant and instructor, dealing with Oracle database core technologies
- Frequent speaker at Oracle Open World annual event and various user group conferences around the globe

About Brillix-DBAces

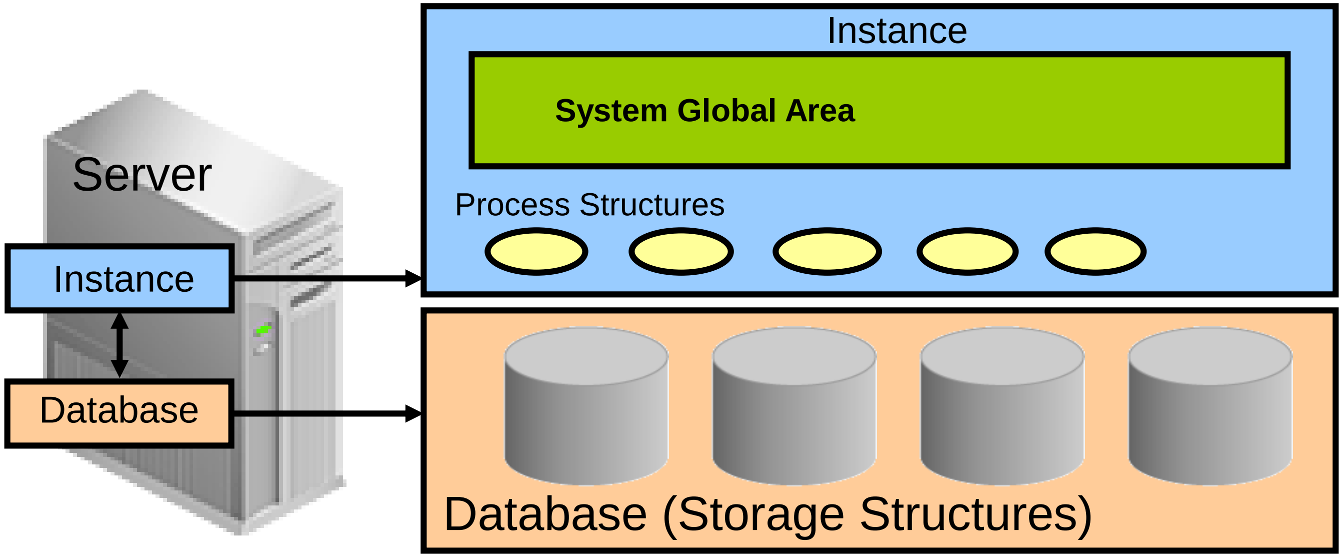
We are committed to provide the highest quality of services delivered by our dedicated team of industry’s top experts. We offer:

- Complete end-to-end solutions based on best-of-breed innovations in database, security and big data technologies
- On-site professional customized trainings led by our team of Oracle ACEs and Oracle Certified Professionals
- Comprehensive security solutions and services for leading database platforms and business applications, leveraging a world-class team of security experts

ORACLE Gold Partner ORACLE EDUCATION RESELLER Microsoft Partner

Oracle Database in 11*g* Release 2

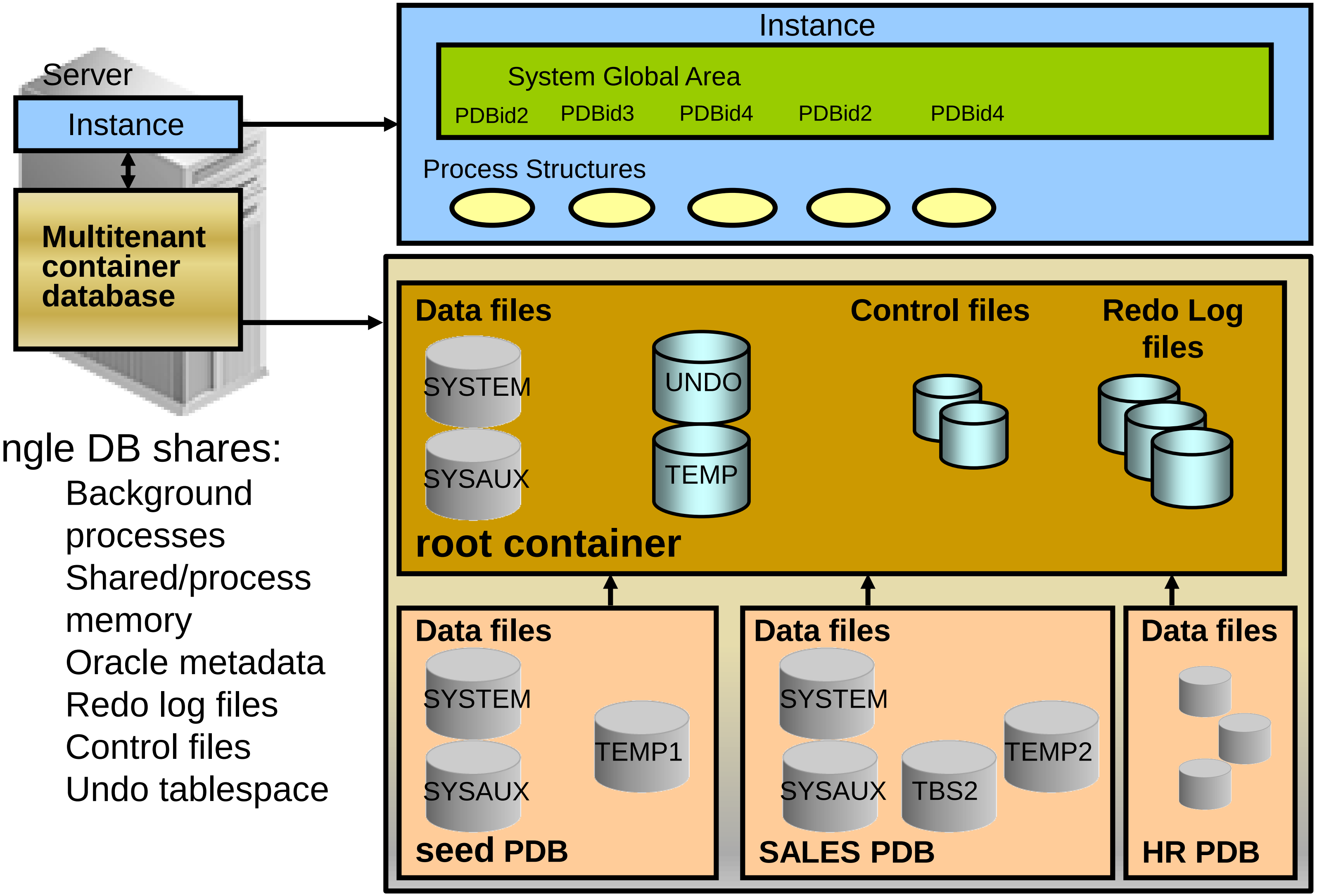
Pre-12*c*



Multiple non-CDBs share nothing:

- Too many background processes
- High shared/process memory
- Many copies of Oracle metadata

Multitenant Container Database Architecture



Single DB shares:

- Background processes
- Shared/process memory
- Oracle metadata
- Redo log files
- Control files
- Undo tablespace

New Multitenant Architecture: Benefits

- Operates **multiple databases in a centrally managed platform** to lower costs:
  - Less instance overhead
  - Less storage cost
- No application changes
- **Fast and easy provisioning**
- Ensures **full backwards-compatibility** with non-CDBs
- Fully operates with RAC and Data Guard
- Is supported by Enterprise Manager
- Allows **central management and administration of multiple databases**
  - Backups or disaster recovery
  - Patching and upgrades

Containers

Two types of containers in `V$CONTAINERS`:

- The root container:
  - The first container created at CDB creation
  - Mandatory
  - Oracle system-supplied common objects and metadata
  - Oracle system-supplied common users and roles
- Pluggable database containers (PDBs):
  - A container for an application:
    - Tablespaces (permanent and temporary)
    - Schemas / objects / privileges
    - Created / cloned / unplugged / plugged
  - Particular seed PDB:
    - `PDB$SEED` provides fast provisioning of a new PDB
  - Limit of 253 PDBs in a CDB including the seed
  - Limit of 1024 services in a CDB

Deployment – CDB Creation and Configuration

- Use DBCA
- Standardize your database options and character set
- Size the CDB as you would a large database
  - Configure Huge Pages is SGA > 30GB
  - Modify memlock limits accordingly
  - Use ASMM
  - Set processes to 100 * physical core
  - Set SGA_TARGET to 60% of physical memory
  - Automatic PGA memory management (20% of SGA)
  - Redo: minimum 4GB and size to switch max <= 10-20 mins, 3-4 redo log groups, archive

Deployment – PDB Creation and Configuration

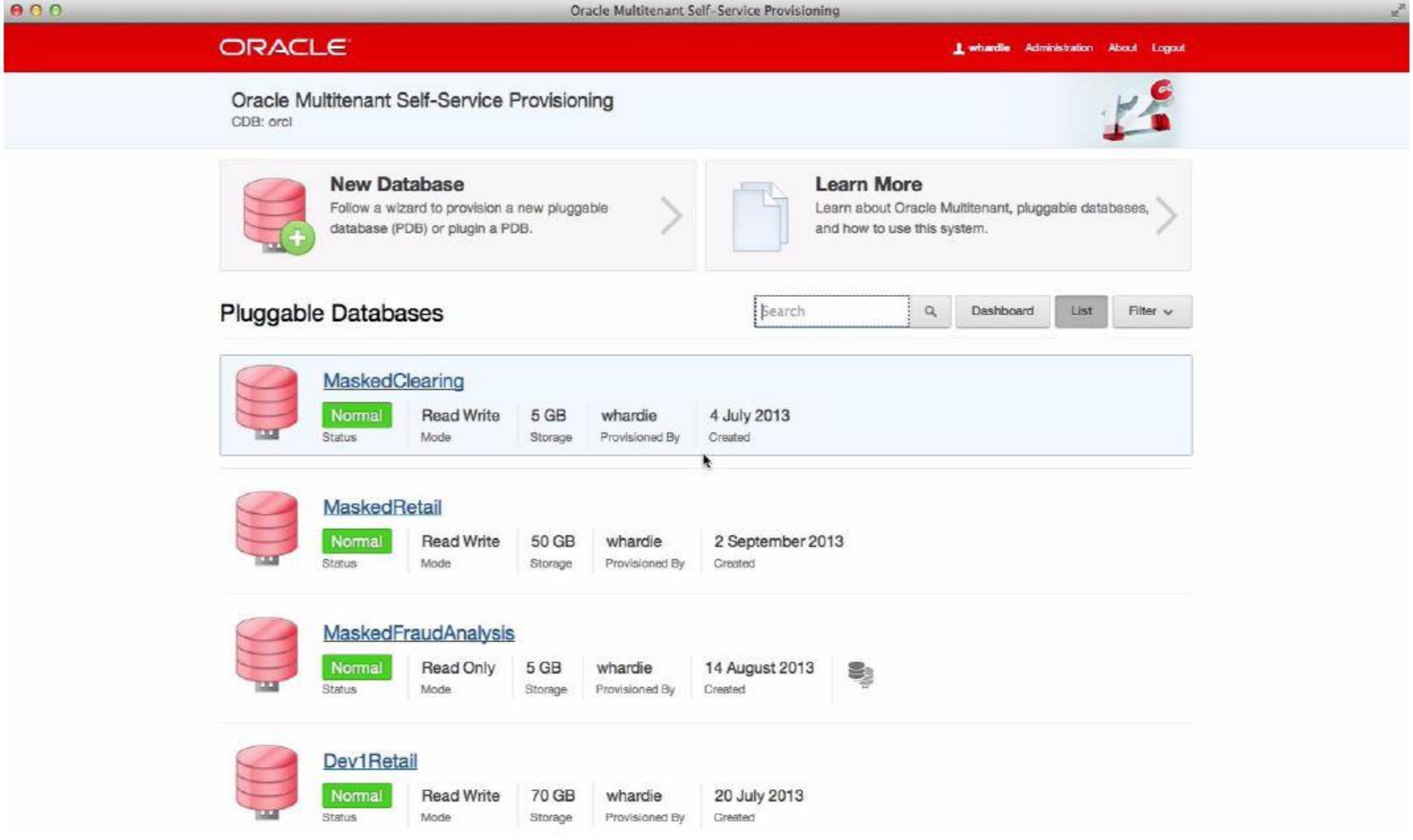
- Clone
- Configure clone quotas and storage limits
- Don’t modify PDB$SEED
- Create and customize your own SEED
- Use CREATE_FILE_DEST for PDB file destination (12.1.0.2)
- Check ISPDB_MODIFIABLE evaluate and adjust parameters that affects application performance (optimizer, cursors…)
- Check PDB parameter settings in your session

Oracle Pluggable Database Self-Service Provisioning Application

- Self-service provisioning of pluggable databases (PDBs)
- Easy and productive way for DBAs and developers to create, clone, plug and unplug PDBs
- Prerequisites:
  - Oracle Database 12c Release 1 (12.1.0.2.0 or above)
  - Oracle Application Express 4.2.5 or above
  - Oracle REST Data Services 2.0.6 or above

http://www.oracle.com/technetwork/database/multitenant/downloads/multitenant-pdbss-2016324.html

Oracle Pluggable Database Self-Service Provisioning Application



Oracle Multitenant Self-Service Provisioning

ORACLE

whardie Administration About Logout

Oracle Multitenant Self-Service Provisioning

CDB: orcl

New Database

Follow a wizard to provision a new pluggable database (PDB) or plugin a PDB.

Learn More

Learn about Oracle Multitenant, pluggable databases, and how to use this system.

Pluggable Databases

Search | Dashboard | List | Filter

MaskedClearing

Normal	Read Write	5 GB	whardie	4 July 2013
Status	Mode	Storage	Provisioned By	Created

MaskedRetail

Normal	Read Write	50 GB	whardie	2 September 2013
Status	Mode	Storage	Provisioned By	Created

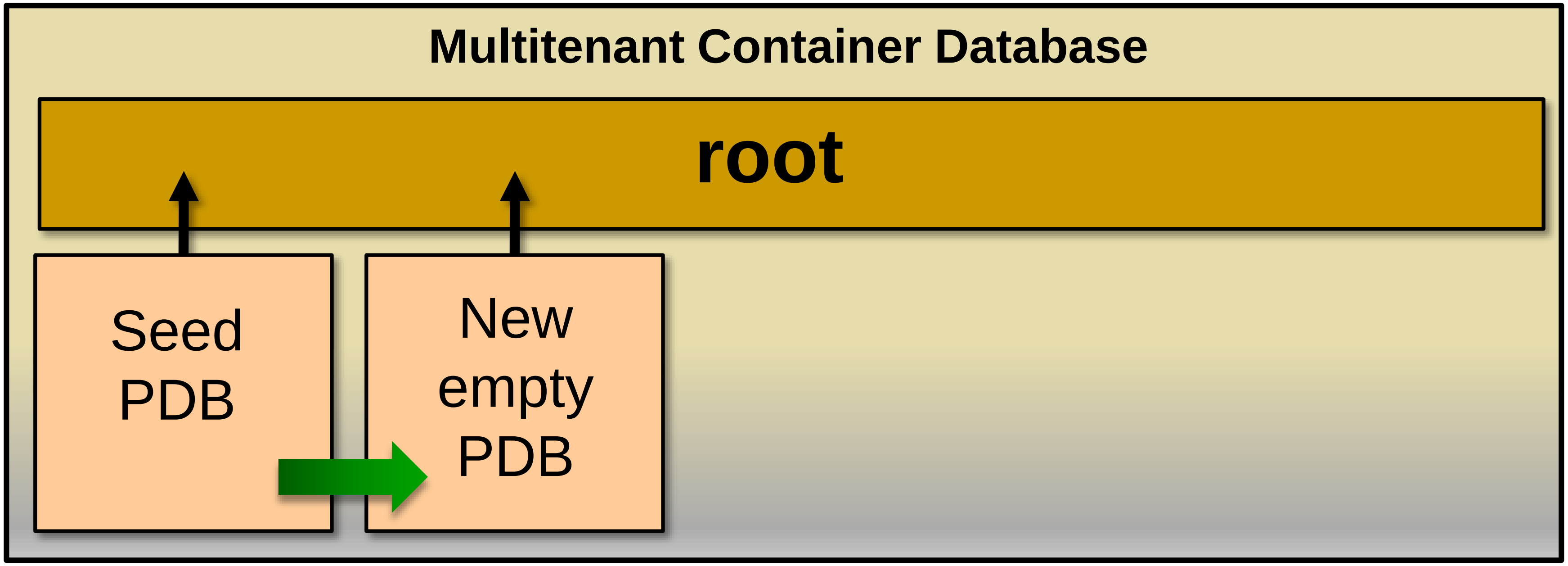
MaskedFraudAnalysis

Normal	Read Only	5 GB	whardie	14 August 2013
Status	Mode	Storage	Provisioned By	Created

Dev1Retail

Normal	Read Write	70 GB	whardie	20 July 2013
Status	Mode	Storage	Provisioned By	Created

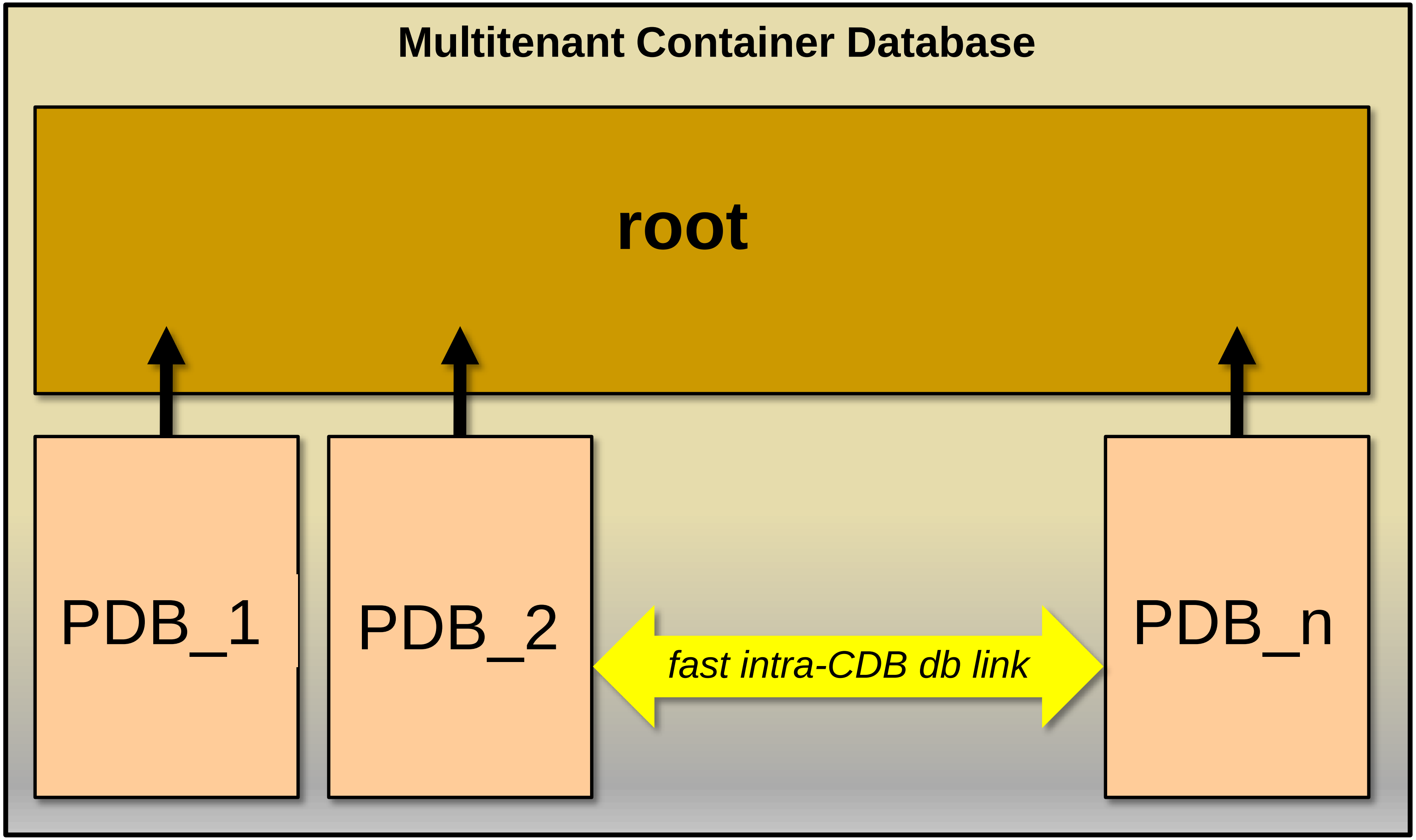
Provisioning a Pluggable Database



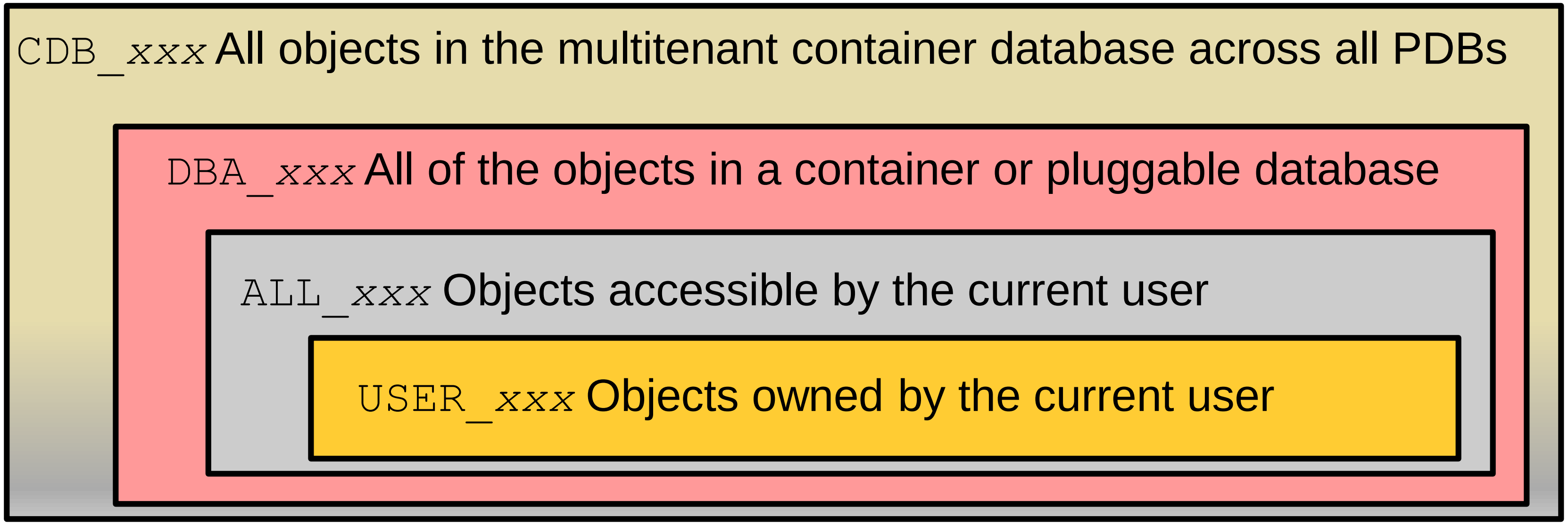
Different methods:

- Create new PDB from `PDB$SEED` pluggable database.
- Plug in a non-CDB.
- Clone a non-CDB into a CDB, or a PDB into the same or another CDB.
- Plug an unplugged PDB into a CDB.

Interacting Within Multitenant Container Database



Data Dictionary Views



`CDB_xxx` All objects in the multitenant container database across all PDBs

`DBA_xxx` All of the objects in a container or pluggable database

`ALL_xxx` Objects accessible by the current user

`USER_xxx` Objects owned by the current user

```
SQL> SELECT view_name FROM dba_views WHERE view_name like 'CDB%';
```

- `CDB_pdbs`: All PDBS within CDB
- `CDB_tablespaces`: All tablespaces within CDB
- `CDB_users`: All users within CDB (common and local)

DBA dictionary views providing information within PDB:

```
SQL> SELECT table_name FROM dict WHERE  table_name like 'DBA%';
```

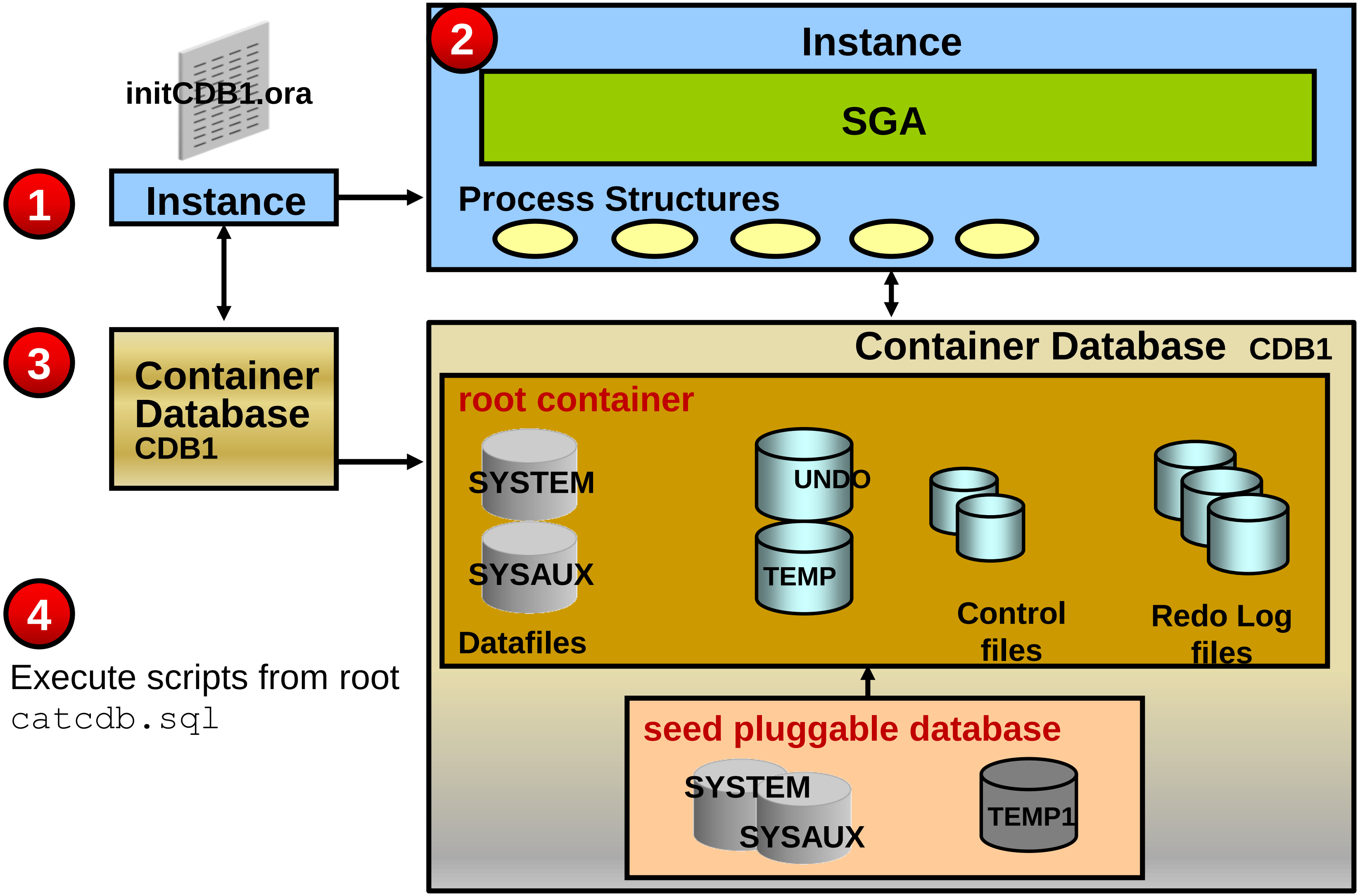
Impacts

- One character set for all PDBs (Unicode recommended)
- PDB initialization parameters but a single SPFILE
- No PDB qualified database object names
  - SELECT * FROM ~~HR:apps.tab1~~
  - Use DB Links: SELECT * FROM apps.tab1@HR
- Oracle Data Guard at CDB level
- Oracle Database Vault per PDB only
- One master key per PDB to encrypt PDB data
- Unified audit both at CDB and PDB levels
- Oracle Scheduler
- Oracle GoldenGate
- Oracle Streams
- Oracle XStream both at CDB and PDB levels

Tools

	SQL*Plus	OUI	DBCA	EM Cloud Control	EM Database Express	SQL Developer	DBUA
Create a new CDB or PDB	Yes	Yes	Yes	Yes (PDB only)	Yes (PDB only)	Yes (PDB only)	
Explore CDB instance, architecture and PDBs	Yes			Yes	Yes	Yes	
Upgrade a 12.1 CDB to 12.x CDB				Yes			Yes

Steps to Create a Container Database



Execute scripts from root

```
catcdb.sql
```

Creating a Container Database: Using SQL*Plus

1. Instance startup:
   a. Set ORACLE_SID=CDB1
   b. Set in initCDB1.ora:
      - Set CONTROL_FILES to CDB control file names.
      - Set DB_NAME to CDB name.
      - Set ENABLE_PLUGGABLE_DATABASE to TRUE.

```
SQL> CONNECT / AS SYSDBA
SQL> STARTUP NOMOUNT
```

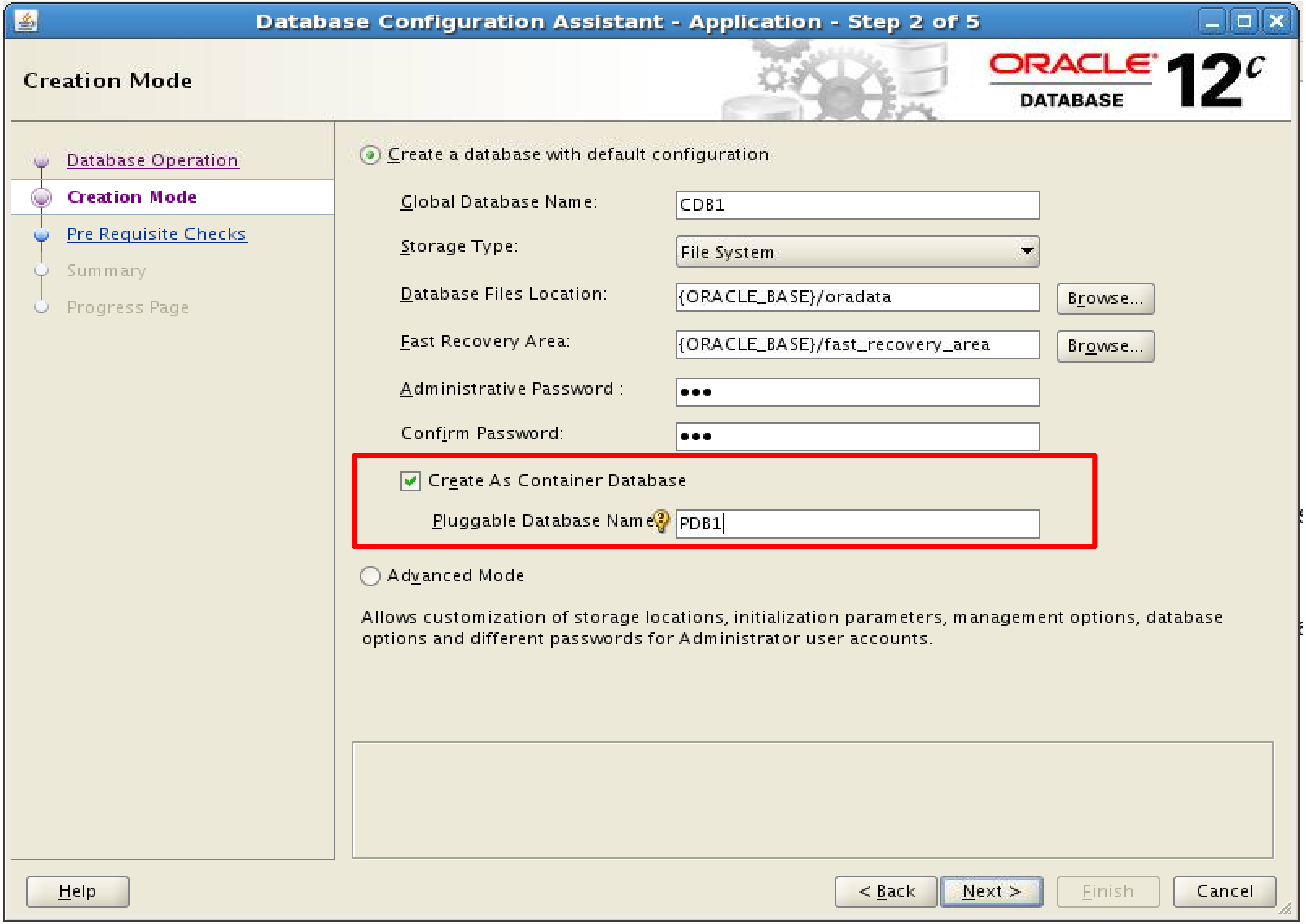
2. Create the database:

```
SQL> CREATE DATABASE CDB1 ENABLE PLUGGABLE DATABASE …
     SEED FILE_NAME_CONVERT ('/oracle/dbs','/oracle/seed');
```

   - CDB$ROOT container
   - PDB$SEED pluggable database

3. Run the catcdb.sql script.

Creating a Container Database: Using DBCA



● Create a database with default configuration

Global Database Name: CDB1

Storage Type: File System

Database Files Location: {ORACLE_BASE}/oradata Browse...

Fast Recovery Area: {ORACLE_BASE}/fast_recovery_area Browse...

Administrative Password : •••

Confirm Password: •••

☑ Create As Container Database

Pluggable Database Name: PDB1

○ Advanced Mode

Allows customization of storage locations, initialization parameters, management options, database options and different passwords for Administrator user accounts.

After CDB Creation: To-Do List

After CDB creation, the CDBA has to:

- Set a separate default tablespace for the root and for each PDB
- Set a default temporary tablespace for each container
- Start the listener
- Plug non-CDBs
- Test startup/shutdown procedures
- Define default PDB state to automate PDBs opening 12.1.0.2
- Create backup and recovery procedures

Provisioning New Pluggable Databases

Different methods:

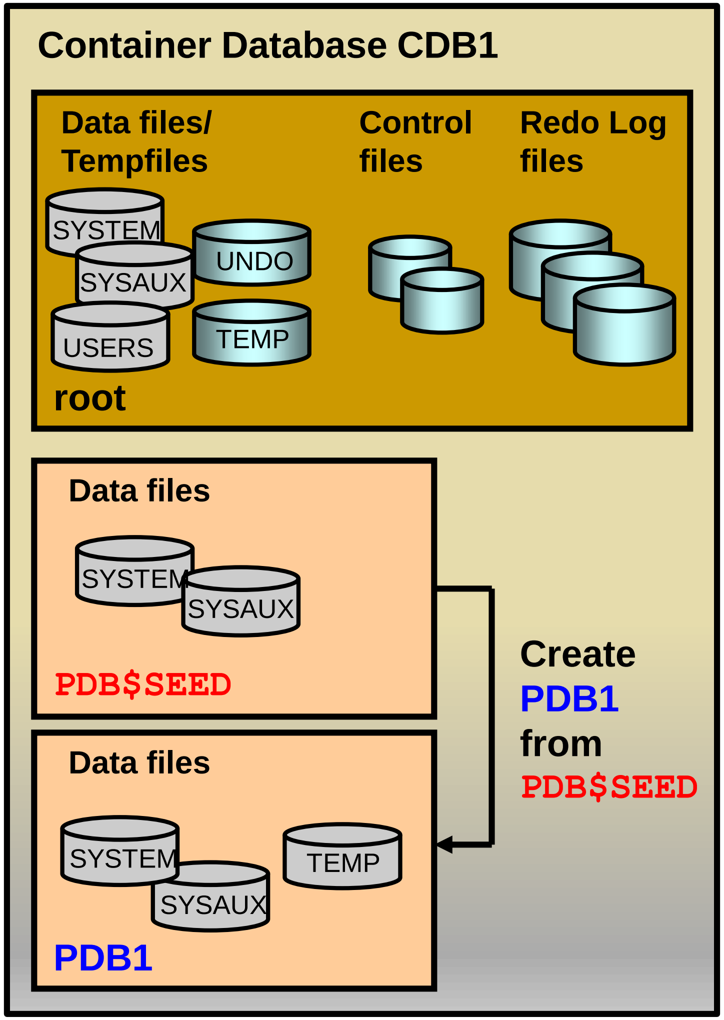
- Create a new PDB from the seed PDB.
- Plug or clone a non-CDB into a CDB.
- Clone:
  - A local PDB into the same CDB
  - A remote PDB into a CDB
- Plug an unplugged PDB into another CDB.

Tools

To provision new PDBs, you can use:

- SQL*Plus
- SQL Developer
- Enterprise Manager Cloud Control
- Enterprise Manager Database Express
- DBCA
  - Copy from seed
  - By unplugging/plugging method

Method 1: Create New PDB from PDB$SEED



- Copies the data files from PDB$SEED data files
- Creates SYSTEM and SYSAUX tablespaces
- Creates a full catalog including metadata pointing to Oracle-supplied objects
- Creates a temporary tablespace, TEMP
- Creates common users:
  - Superuser SYS
  - SYSTEM
- Creates a local user (PDBA) granted local PDB_DBA role
- Creates a new default service

Steps: With Location Clauses

Connect to the root as a common user with the `CREATE PLUGGABLE DATABASE` privilege:

- Use **FILE_NAME_CONVERT**:

```
SQL> CREATE PLUGGABLE DATABASE pdb1
     ADMIN USER admin1 IDENTIFIED BY p1 ROLES=(CONNECT)
     FILE_NAME_CONVERT = ('PDB$SEEDdir', 'PDB1dir');
```

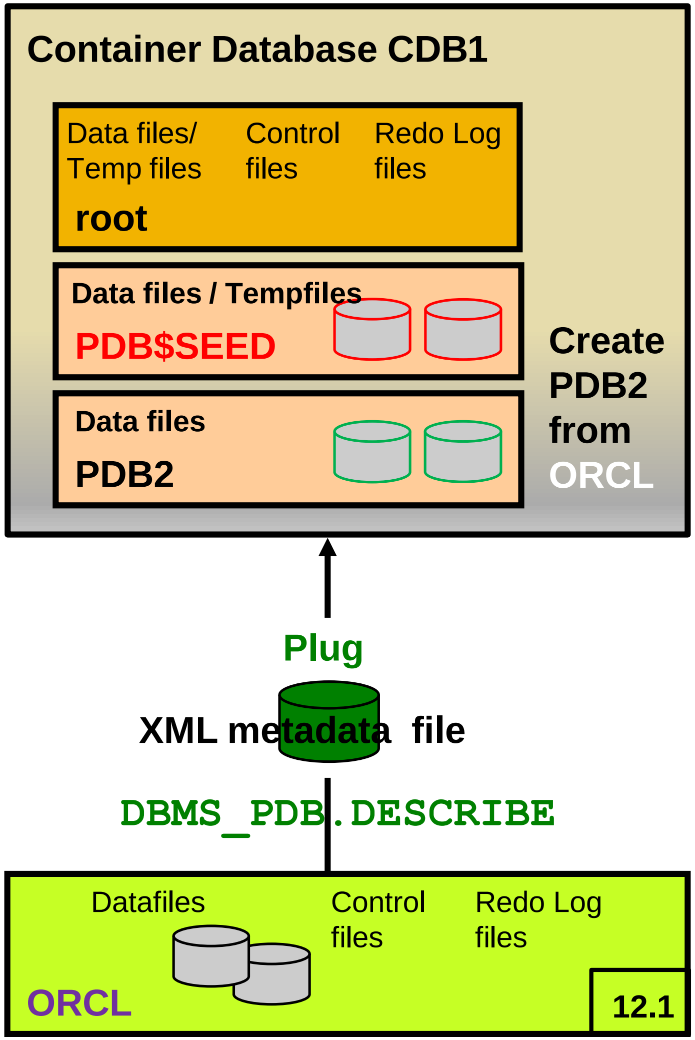
- Use **CREATE_FILE_DEST**: 12.1.0.2

```
SQL> CREATE PLUGGABLE DATABASE pdb2
     ADMIN USER admin2 IDENTIFIED BY p2 ROLES=(CONNECT)
     CREATE_FILE_DEST = 'PDB2dir';
```

- Use views to verify:

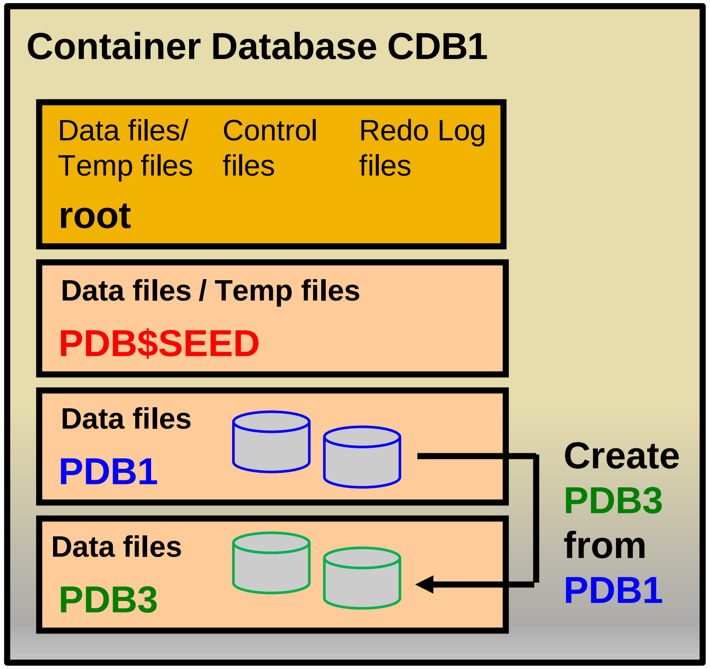
```
SQL> CONNECT / AS SYSDBA
SQL> SELECT * FROM cdb_pdbs;
SQL> SELECT * FROM cdb_tablespaces;
SQL> SELECT * FROM cdb_data_files;
SQL> CONNECT sys@pdb1 AS SYSDBA
SQL> CONNECT admin1@pdb1
```

Plug a Non-CDB in to CDB Using DBMS_PDB



1. Open ORCL in READ ONLY mode.
2.
```
SQL> EXEC DBMS_PDB.DESCRIBE
                    ('/tmp/ORCL.xml')
```
3. Connect to the target CDB1 CDB as a common user with CREATE PLUGGABLE DATABASE privilege.
4. Plug in the unplugged ORCL as PDB2.
```
SQL> CREATE PLUGGABLE DATABASE
     PDB2 USING '/tmp/ORCL.xml';
```
5. Run the noncdb_to_pdb.sql script.
```
SQL> CONNECT sys@PDB2 AS SYSDBA
SQL> @$ORACLE_HOME/rdbms/admin/noncdb_to_pdb
```
6. Open PDB2.
```
SQL> ALTER PLUGGABLE DATABASE
     PDB2 OPEN;
```

Method 3: Clone Local PDBs



PDB3 owns:

- SYSTEM, SYSAUX tablespaces
- Full catalog
- A temporary tablespace
- SYS, SYSTEM common users
- Same local administrator name
- New service name

1. Set the DB_CREATE_FILE_DEST or DB_FILE_NAME_CONVERT instance parameter or use the CREATE_FILE_DEST clause. 12.1.0.2
2. Connect to the root.
3. Quiesce PDB1:

```
SQL> ALTER PLUGGABLE DATABASE
     pdb1 CLOSE;
SQL> ALTER PLUGGABLE DATABASE
     pdb1 OPEN READ ONLY;
```

4. Clone PDB3 from PDB1:

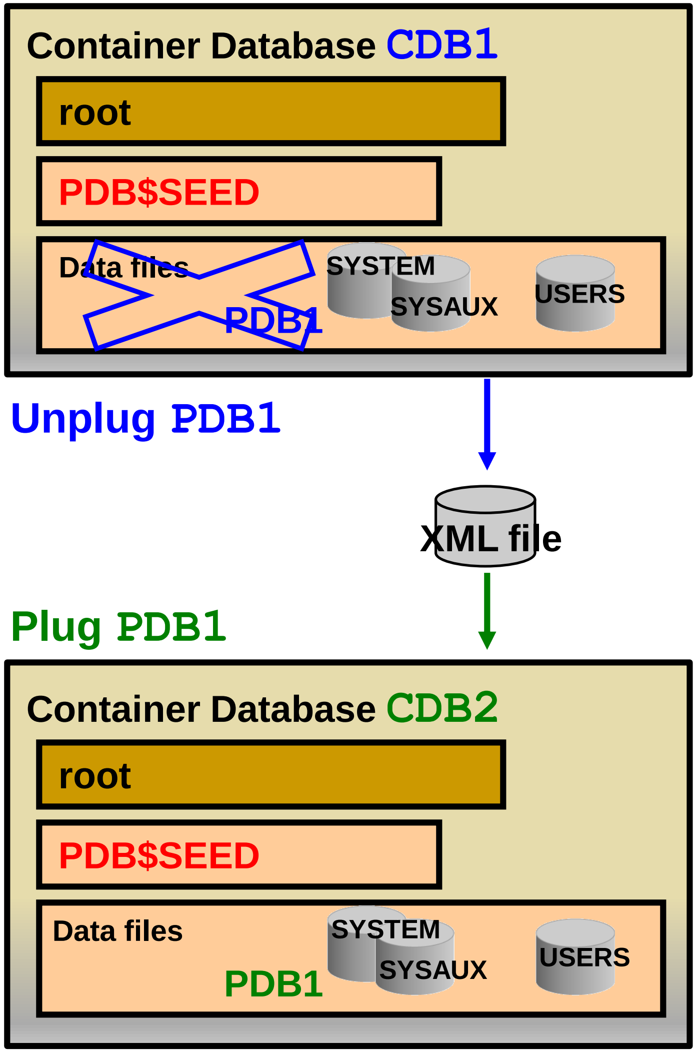
```
SQL> CREATE PLUGGABLE DATABASE
     pdb3 FROM pdb1;
```

5. Open PDB3 in read-write mode.

```
SQL> ALTER PLUGGABLE DATABASE
     pdb3 OPEN;
```

6. Reopen PDB1.

Method 4: Plug Unplugged PDB in to CDB



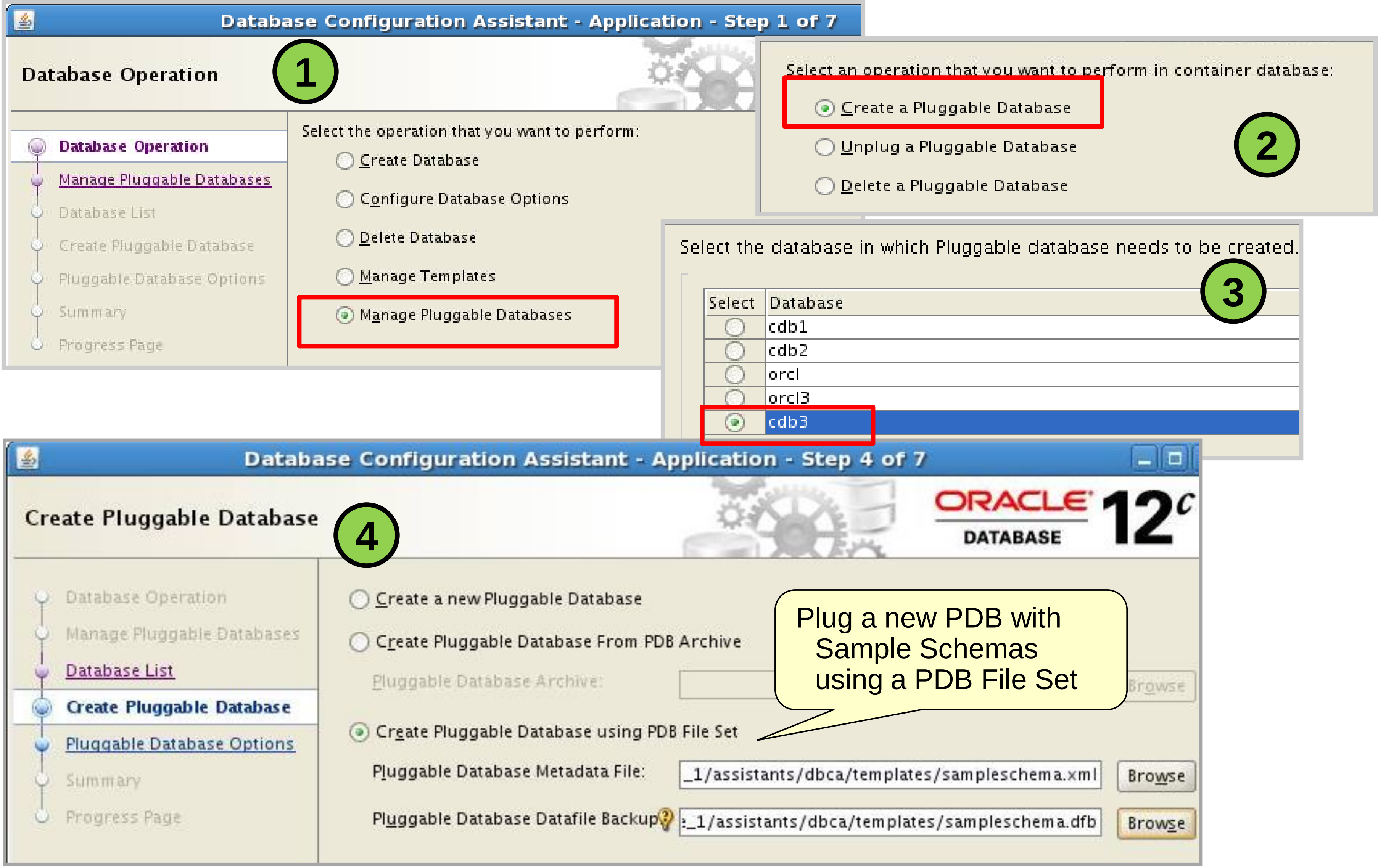
Unplug PDB1 from CDB1:

1. Connect to CDB1 as a common user.
2. Verify that PDB1 is closed.
3.
```
SQL> ALTER PLUGGABLE DATABASE
     pdb1 UNPLUG INTO
                'xmlfile1.xml';
```
4. Optionally, drop PDB1 from CDB1.

Plug PDB1 in to CDB2:

1. Connect to CDB2 as a common user.
2. Use DBMS_PDB package to check the compatibility of PDB1 with CDB2.
3.
```
SQL> CREATE PLUGGABLE DATABASE
       pdb1 USING 'xmlfile1.xml'
     NOCOPY;
```
4. Open PDB1 in read-write mode.

Plug Sample Schemas PDB: Using DBCA



1

Database Configuration Assistant - Application - Step 1 of 7

Database Operation

- Database Operation
- Manage Pluggable Databases
- Database List
- Create Pluggable Database
- Pluggable Database Options
- Summary
- Progress Page

Select the operation that you want to perform:

○ Create Database
○ Configure Database Options
○ Delete Database
○ Manage Templates
● Manage Pluggable Databases

2

Select an operation that you want to perform in container database:

● Create a Pluggable Database
○ Unplug a Pluggable Database
○ Delete a Pluggable Database

3

Select the database in which Pluggable database needs to be created.

Select	Database
○	cdb1
○	cdb2
○	orcl
○	orcl3
●	cdb3

4

Database Configuration Assistant - Application - Step 4 of 7

Create Pluggable Database

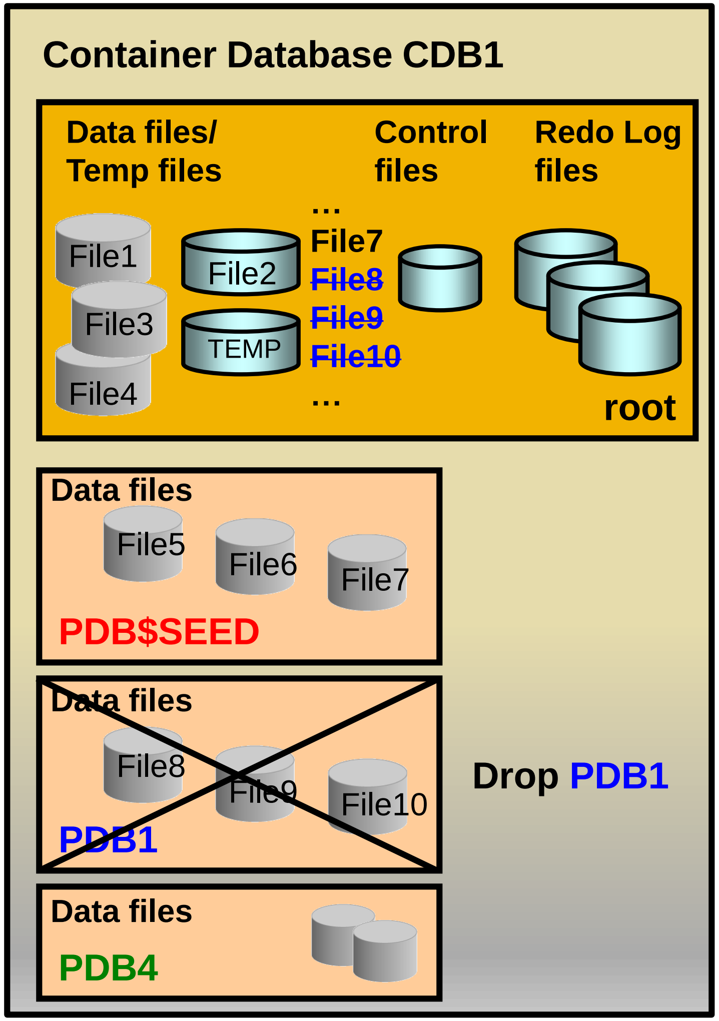
ORACLE DATABASE 12c

- Database Operation
- Manage Pluggable Databases
- Database List
- Create Pluggable Database
- Pluggable Database Options
- Summary
- Progress Page

○ Create a new Pluggable Database
○ Create Pluggable Database From PDB Archive
Pluggable Database Archive: Browse
● Create Pluggable Database using PDB File Set
Pluggable Database Metadata File: _1/assistants/dbca/templates/sampleschema.xml Browse
Pluggable Database Datafile Backup: _1/assistants/dbca/templates/sampleschema.dfb Browse

Plug a new PDB with Sample Schemas using a PDB File Set

Dropping a PDB



```
SQL> ALTER PLUGGABLE DATABASE
     pdb1 CLOSE;
SQL> DROP PLUGGABLE DATABASE
     pdb1 [INCLUDING DATAFILES];
```

- Updates control files
- If `INCLUDING DATAFILES`:
  - Removes **PDB1** datafiles
- If `KEEP DATAFILES` (default):
  - Retain data files
  - Can be plugged in another or the same CDB
- Requires `SYSDBA` privilege
- Cannot drop seed PDB

Exploring Oracle Database 12c Multitenant Best Practices for your Cloud

Ami Aharonovich

Oracle ACE & OCP

Ami@DBAces.com