ORACLE

Oracle Database 12*c* –
Pluggable Databases – Part I

ORACLE

Using the Player

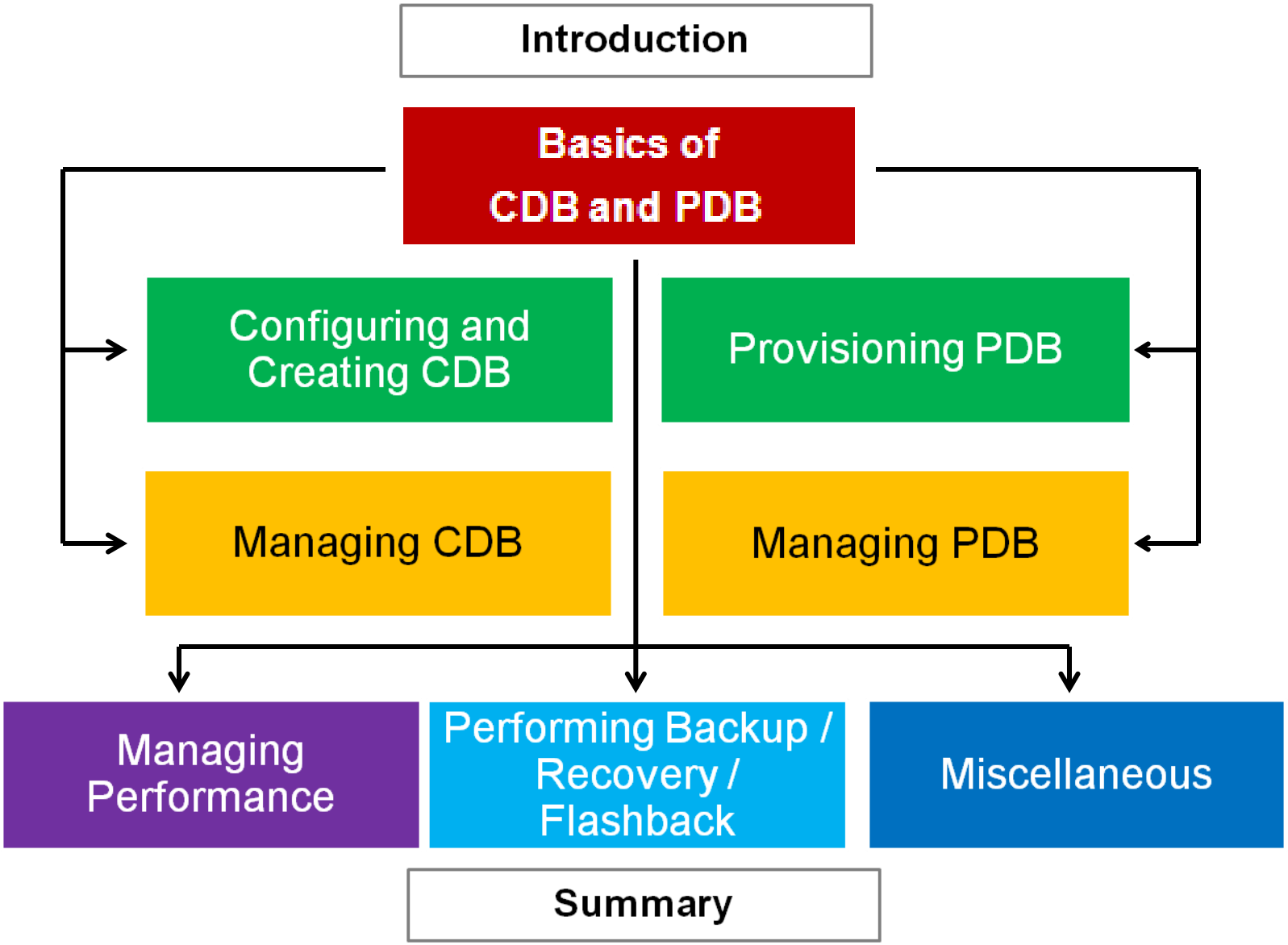
Print Version

Course Outline

Player Controls

Change View

Road Map



Road Map

PART – I

Basics of CDB and PDB

Configuring and Creating CDB

Provisioning PDB

PART – II

Managing CDB

Managing PDB

PART – III

Managing Performance

Performing Backup / Recovery / Flashback

Miscellaneous

About This Course

Search

What skills will I learn?

At the end of this course, you should be able to:

- Describe the challenges and benefits for a container database
- Explain the architecture of a container database
- Define the different types of containers in a container database
- Describe the content of the root container
- Describe the content of a pluggable database container
- Describe the differences between the root and a pluggable database
- Configure and create container databases
- Understand the methods for creating and plugging pluggable databases
- Drop pluggable databases

Who is the target audience?

What are the prerequisites?

PROPERTIES

Allow user to leave interaction:	After viewing all the steps
Show 'Next Slide' Button:	Show upon completion
Completion Button Label:	Next Slide

Properties...

Edit in Engage

Why Take This Course?

- What's in it for me?
- What are challenges I face on the job?
- How can this help me do my job better?

ORACLE

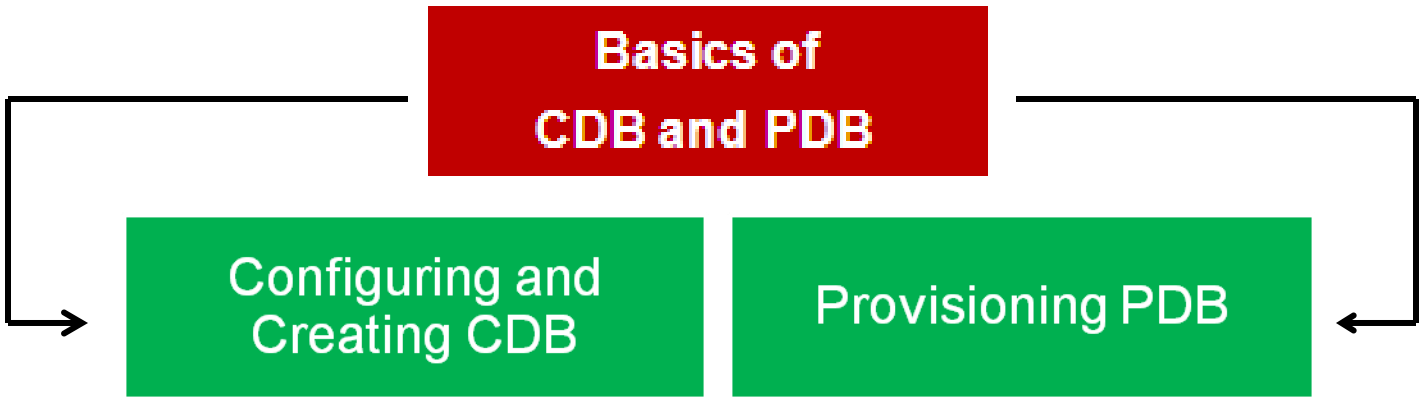
ORACLE THIRTY YEARS 30

Oracle Database 12*c* –
Basics of Container and Pluggable Databases

ORACLE

Road Map

PART – I



Challenges

Many Oracle customers have large numbers of “departmental” applications built on Oracle RDBMS.

- Do NOT use a significant percentage of the hardware on which they are deployed
- Have instance and storage overhead preventing large numbers of “departmental” databases from being placed on the same physical and storage server
- Are NOT sufficient complexity to require 100% of the attention of a full time administrator
- Do require significant time to patch or upgrade all applications

Benefits of 12*c* Pluggable Databases

- Operates **multiple monolithic databases in a centrally managed platform** to lower costs:
  - Less instance overhead
  - Less storage cost
- Reduces DBA resources costs and maintains security
  - No application changes
  - **Fast and easy provisioning**
  - **Time saving for patching and upgrade**
  - **Maintain separation of duties** between:
    - Different application administrators
    - Application administrators and DBA
    - Users within application
- **Provides isolation**

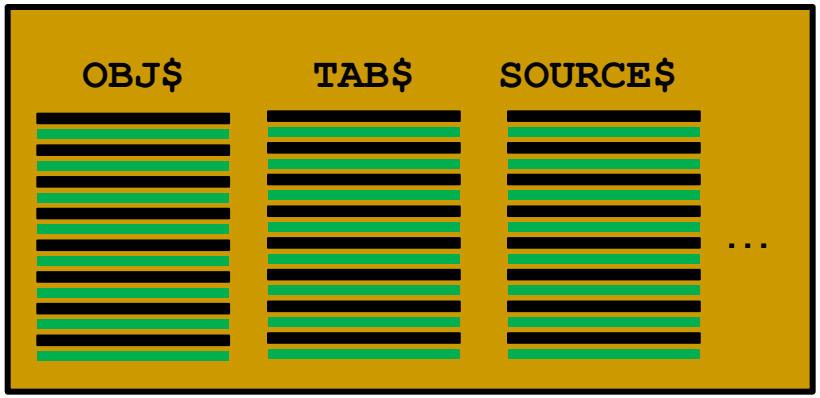
Benefits of 12*c* Pluggable Databases

- Ensures **full backwards-compatibility** with non-CDB
- Fully operates with RAC
- Is integrated with Enterprise and Resource Manager
- Allows central management and administration of multiple databases
  - Backups / Disaster recovery
  - Patching and Upgrades

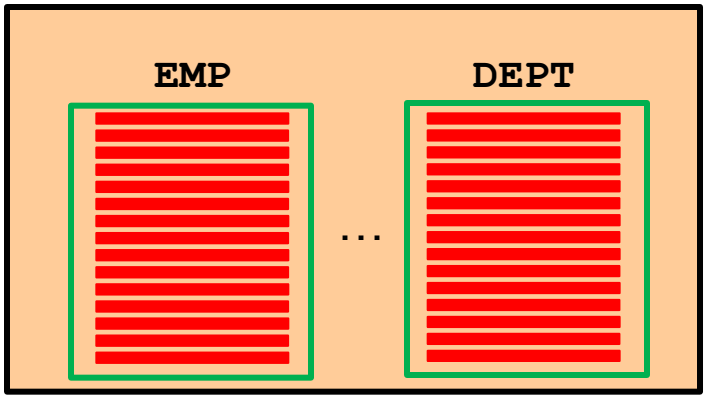
User Data is Added

In a non-CDB, user data is added:

- The metadata is mixed with the Oracle supplied data in the data dictionary.

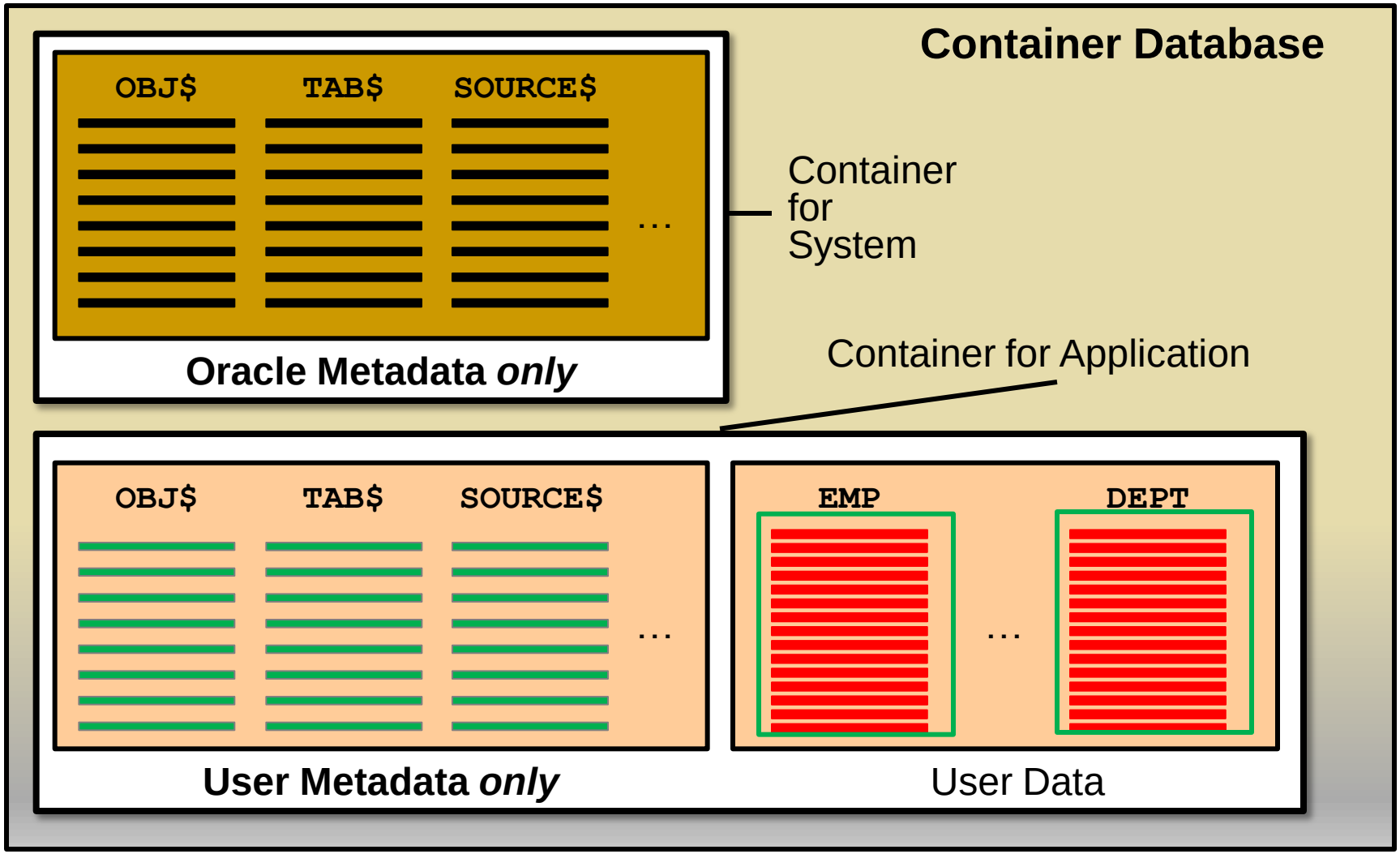


Oracle System data mixed with User metadata

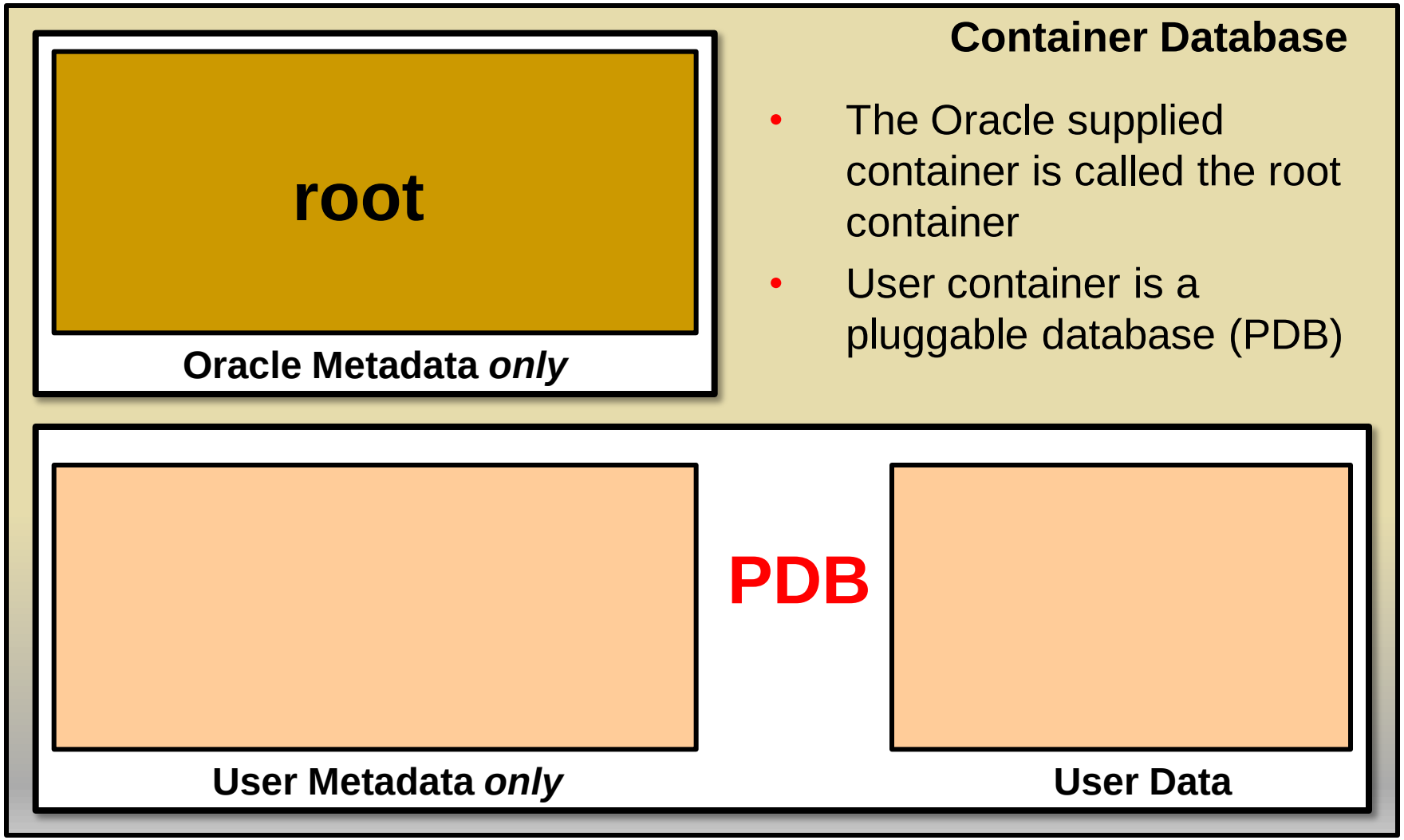


User Data

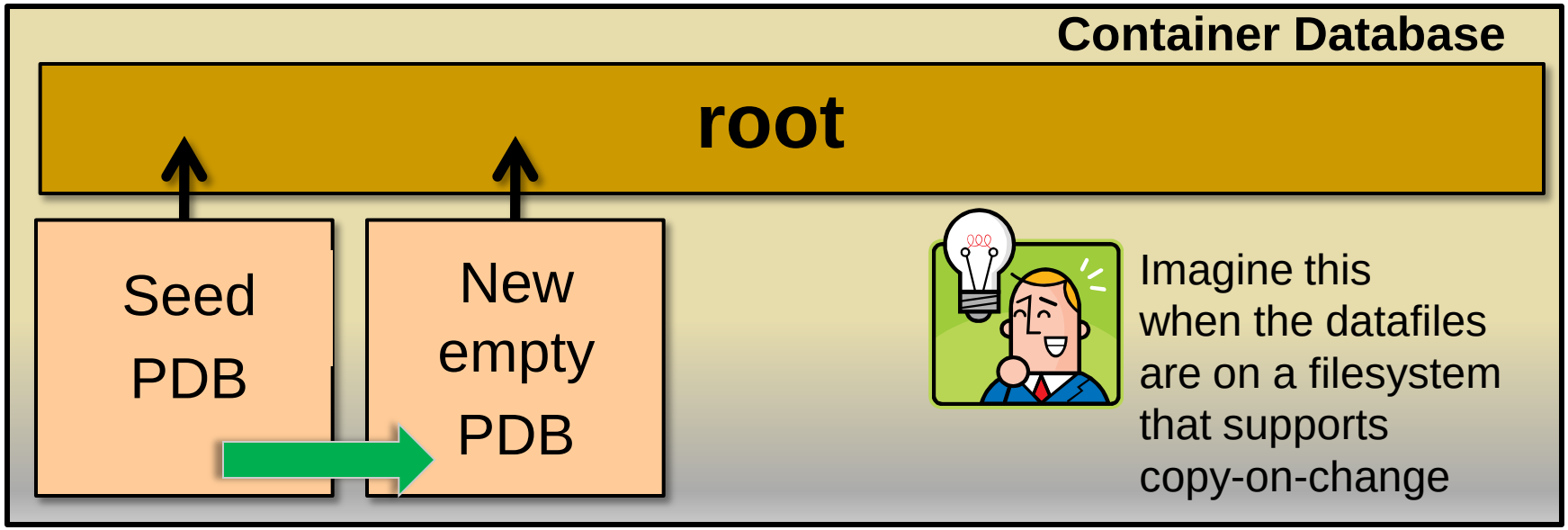
Separating System and User Data



Naming the Containers within Container Database



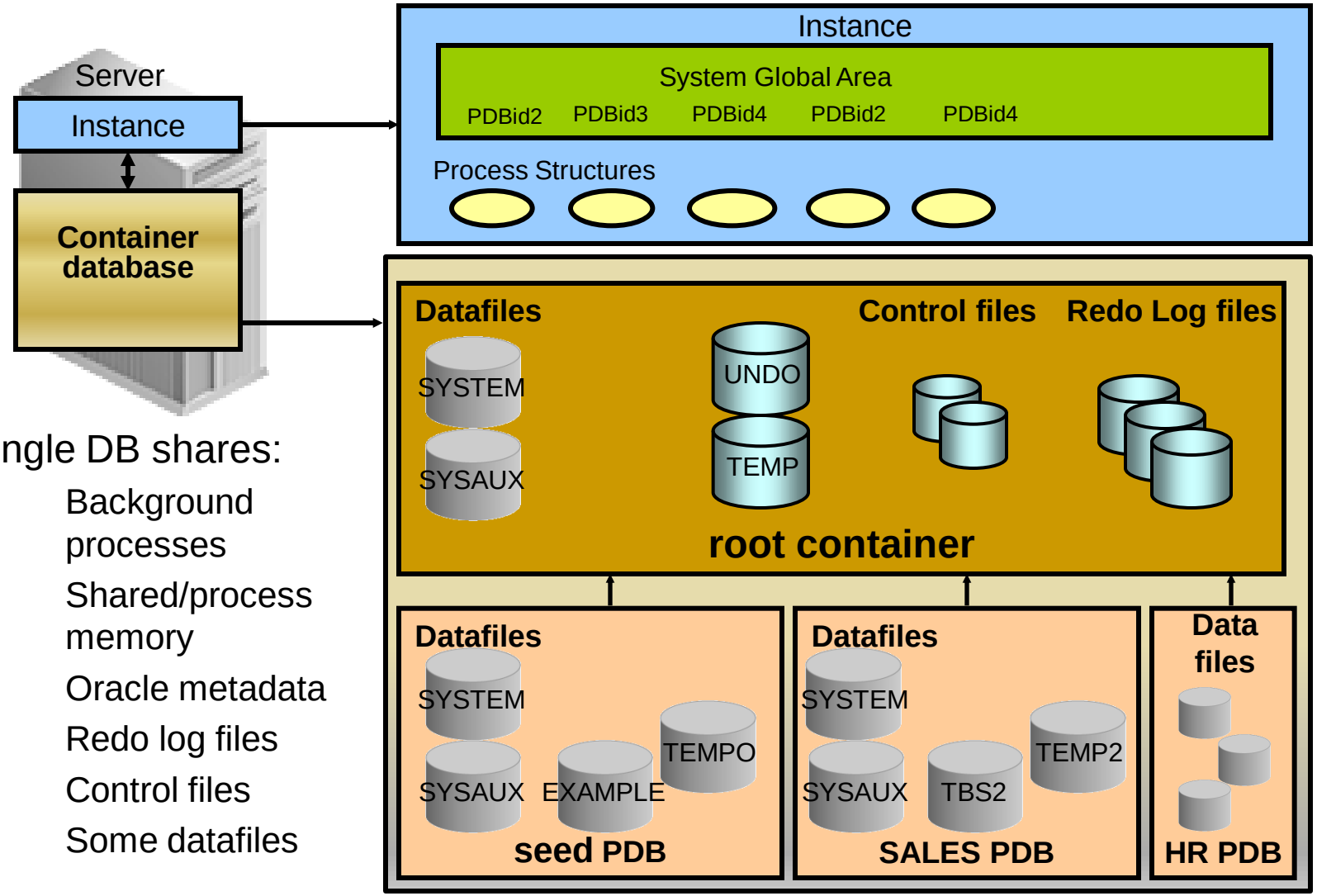
Provisioning Pluggable Databases



Four methods:

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

Container Database Architecture



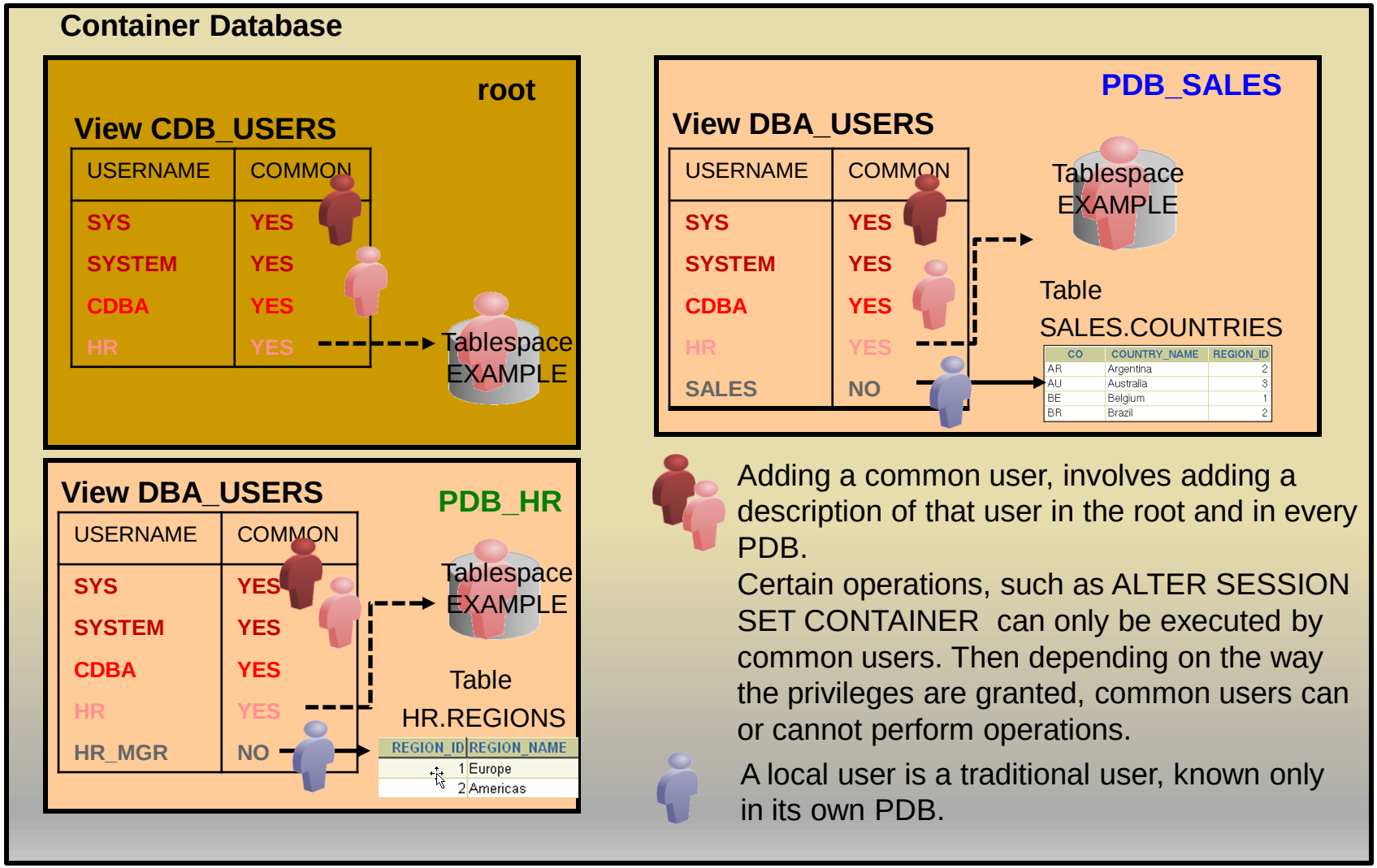
Single DB shares:

- Background processes
- Shared/process memory
- Oracle metadata
- Redo log files
- Control files
- Some datafiles

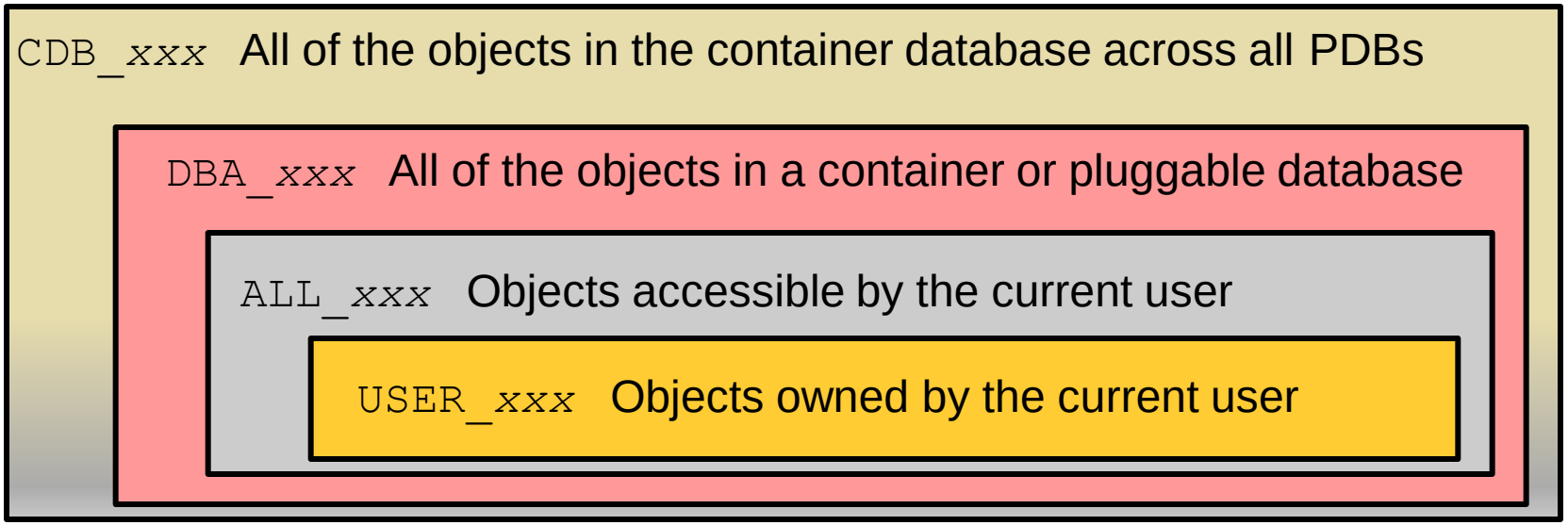
Terminology

- Common vs Local:
  - Users
  - Roles
  - Privileges
  - Objects
- CDB vs PDB level:
  - Global Resource Manager plan vs local RM plan
  - Unified audit at CDB or PDB level
  - Xstream at CDB or PDB level

Common and Local Users



Data Dictionary Views




```
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%';
```

Backup/Recovery/Flashback Database

Backup

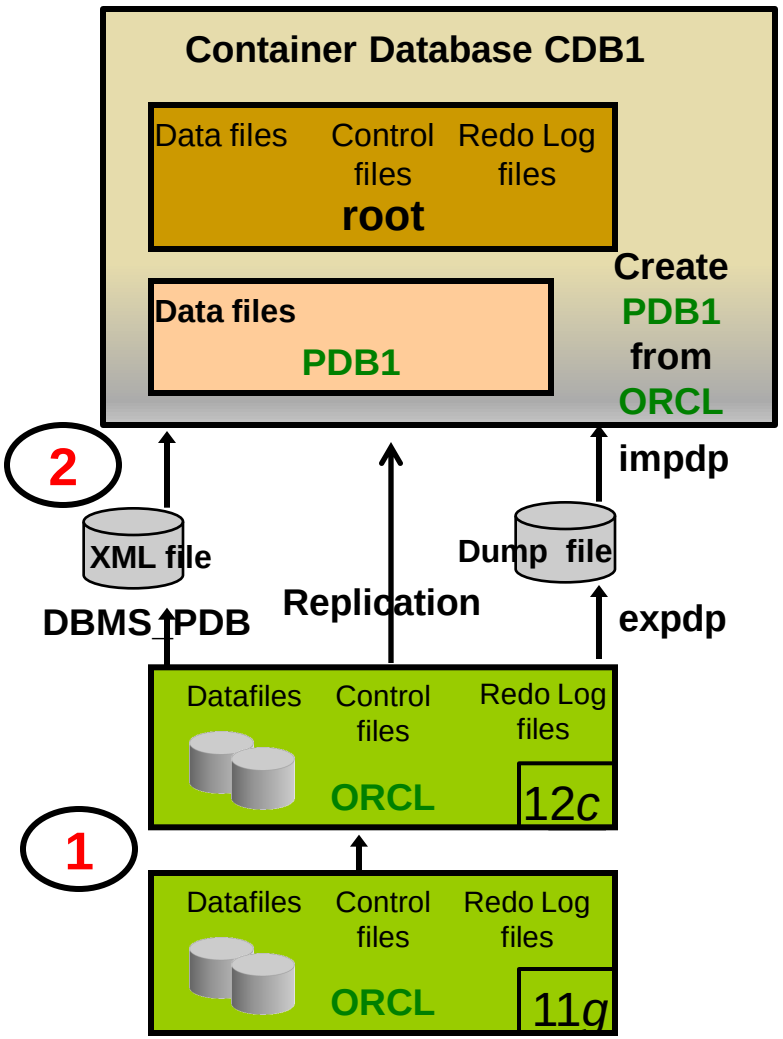
- Whole CDB backup: control files, root and PDBs datafiles
- Whole PDB backup: PDBs datafiles
- Hot backups: not available in Beta1
  - PDB level

Backup/Recovery/Flashback Database

Recovery

- Instance recovery: **CDB level only**
- Complete media recovery from file loss or damage
  - CDB level
  - PDB level: not available in Beta1
  - Tablespace level: same as for non-CDB
- Incomplete media recovery after file loss or corruption
  - CDB level: same as for non-CDB
  - PDB level not available in Beta1
  - TSPITR for root tablespaces ONLY except SYSTEM, UNDO, SYSAUX
- Flashback database
  - **CDB level only**
  - PDB level not available in Beta1
- Block recovery: no change

Migrating pre-12*c* Databases to 12*c* CDB



There are two methods:

Method 1

1. Upgrade a pre-12*c* database to 12*c*
2. Plug-in the non-CDB into a CDB using different methods

Or

Method 2

1. Pre-create a PDB in CDB
2. Use 11*g* expdp / 12*c* impdp

   or

2. Use replication between the non-CDB and the PDB

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
- Data Guard at CDB level
- Database Vault per PDB only
- Unified audit both at CDB and PDB level

Demo: Basics of CDB and PDBs Architecture

PROPERTIES

Allow user to leave interaction:	Anytime
Show ‘Next Slide’ Button:	Show always
Completion Button Label:	Next Slide

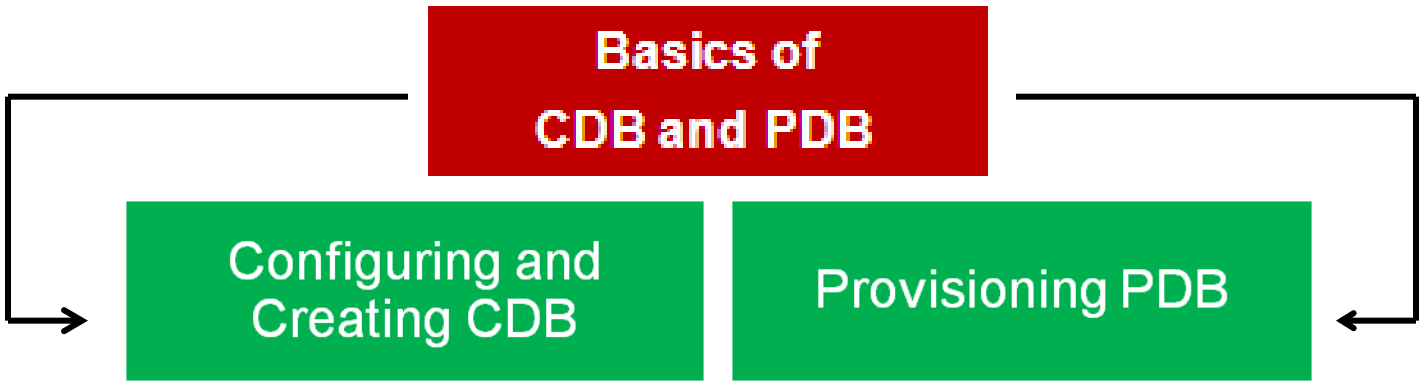
Properties...

Edit in Engage

Lesson Summary

In this first lesson, we discussed:

PART – I



- The benefits for a container database
- The architecture of a container database
- The root container and the pluggable database container
- The methods for provisioning new pluggable databases
- The backup and recovery capabilities and upgrade methods
- The impacts

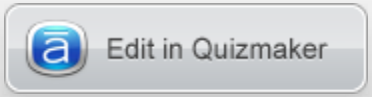
Lesson Review

Drag and drop the shapes below to match the choices with the appropriate assumption.

Query CDB_tables from the root.	Displays the list of tables you have been granted access to in the PDB.
Query CDB_tables from a PDB.	Displays all tables of the PDB.
Query ALL_tables from a PDB.	Displays all tables of the root and PDBs.
Query USER_tables from the root.	Displays all tables owned by the user connected in root.

PROPERTIES

On passing, 'Finish' button: **Goes to Next Slide**
On failing, 'Finish' button: **Goes to Next Slide**
Allow user to leave quiz: **At any time**
User may view slides after quiz: **At any time**
User may attempt quiz: **Unlimited times**



ORACLE

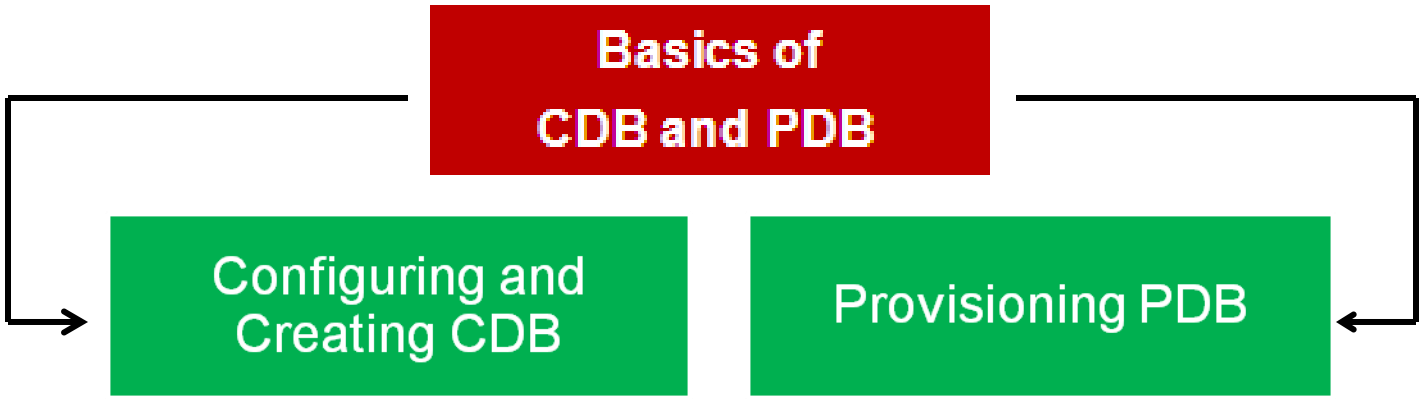
ORACLE THIRTY YEARS 30

Oracle Database 12*c* –
Configuring and Creating a Container Database

ORACLE

Road Map

PART – I

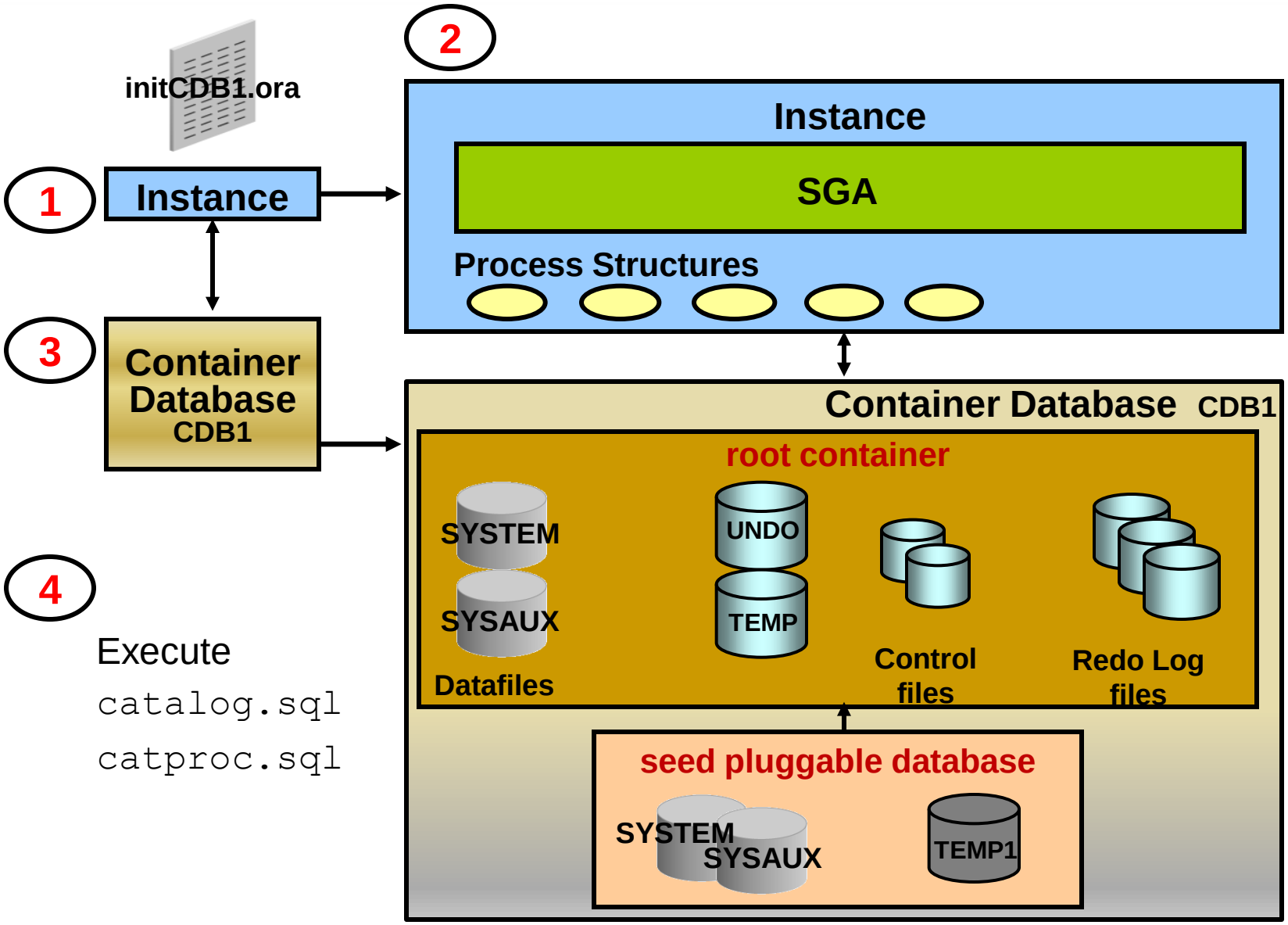


Goals

Create a container database:

- To consolidate many pre-12.1 non-CDBs into a single, larger database
- To prepare a container
  - For plugging any future new application
  - For testing applications
  - For diagnosing application performance
  - For unplugging/plugging operations
- To simplify and reduce time for patching and upgrade

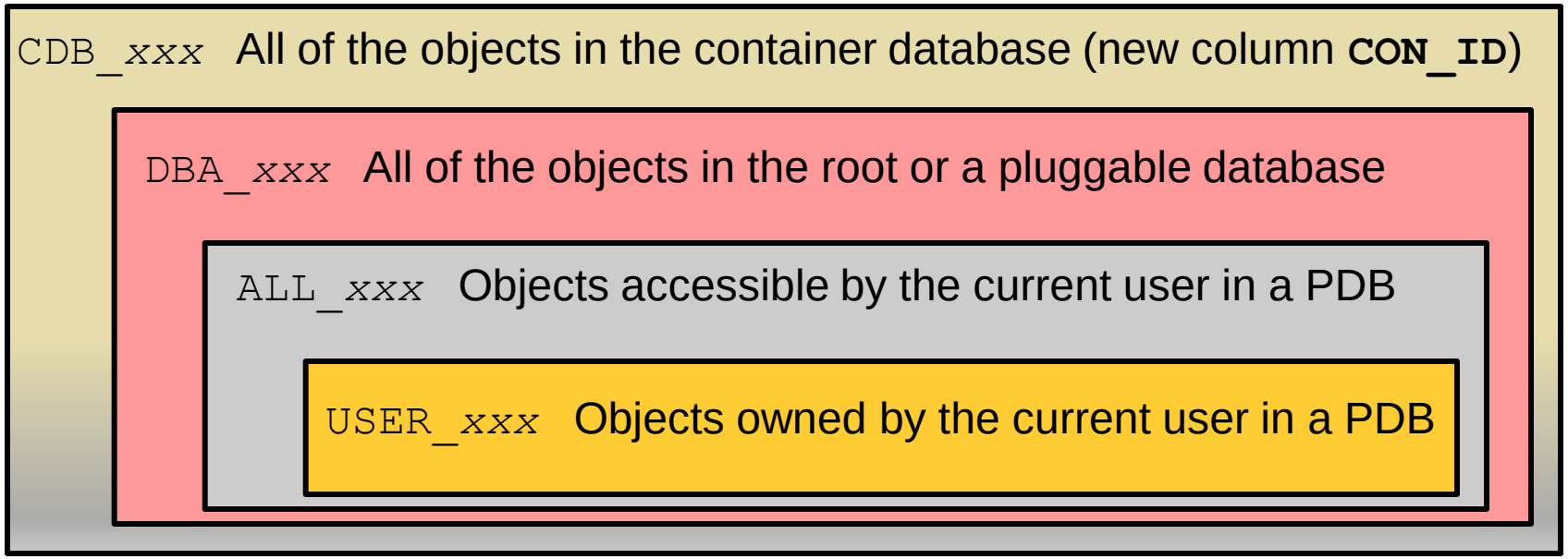
Steps to Create a Container Database



A CDB has new characteristics compared to non-CDBs:

- **Two containers**:
  - The **root** (CDB$ROOT)
  - The **seed** PDB (PDB$SEED)
- **Several services**: one per container
  - Name of root service = name of the CDB: cdb1
- **Common users** in root and seed: SYS, SYSTEM …
- **Common privileges** granted to common users
- **Common roles**
- Tablespaces and datafiles associated to each container:
  - **root**: SYSTEM contains system-supplied metadata and no user data, SYSAUX
  - **seed**: SYSTEM, SYSAUX

Data Dictionary Views – CDB_*xxx and V$xxx*



CDB dictionary views provide information across PDBs:

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

- CDB_pdbs : all PDBS within the CDB
- CDB_tablespaces : all tablespaces within the CDB
- CDB_data_files : all datafiles within the CDB
- CDB_users : all users within the CDB (common and local)

Demo: Configuring and Creating CDB

PROPERTIES

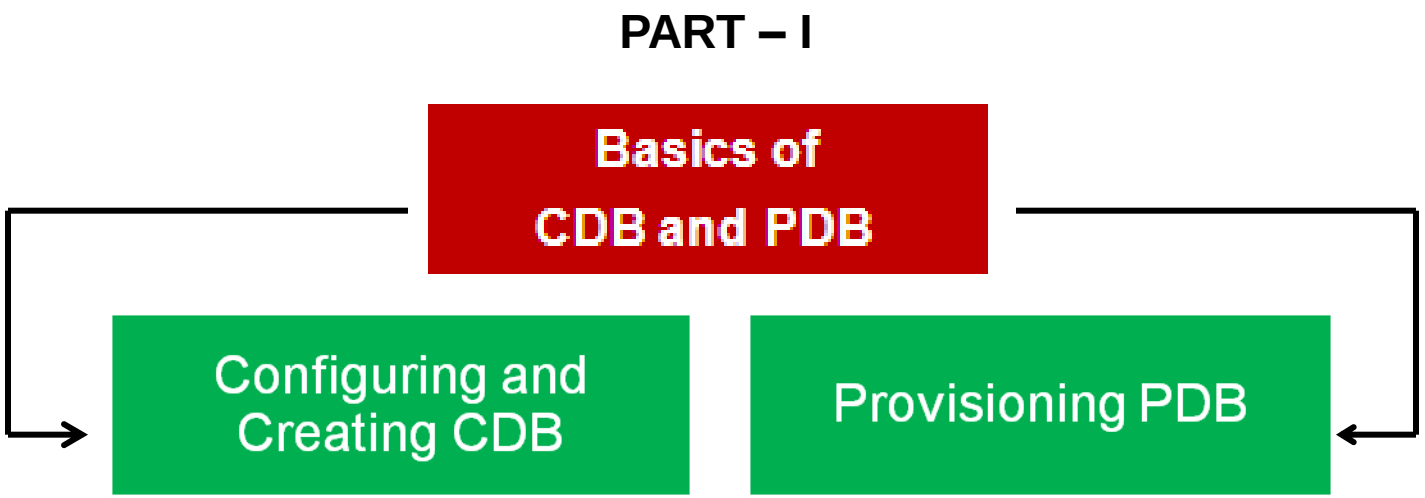
Allow user to leave interaction:	Anytime
Show ‘Next Slide’ Button:	Show always
Completion Button Label:	Next Slide

Properties...

Edit in Engage

Lesson Summary

In this second lesson, we discussed:



- Create a container database
- Explore CDB instance, files and PDB structures

Lesson Review

You manage the tablespaces in a CDB and its PDBs. Choose the correct choice.

If you change the default tablespace for the CDB, it applies to all PDBs.

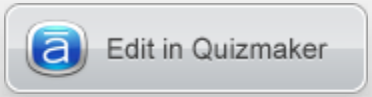
You can change the default tablespace of a PDB independently of its CDB default tablespace and all other PDBs.

The default temporary tablespace is necessarily the same for all containers.

If you create a common user with a default tablespace, the tablespace is not required in all PDBs, only in the root.

PROPERTIES

On passing, 'Finish' button: **Goes to Next Slide**
On failing, 'Finish' button: **Goes to Next Slide**
Allow user to leave quiz: **At any time**
User may view slides after quiz: **At any time**
User may attempt quiz: **Unlimited times**



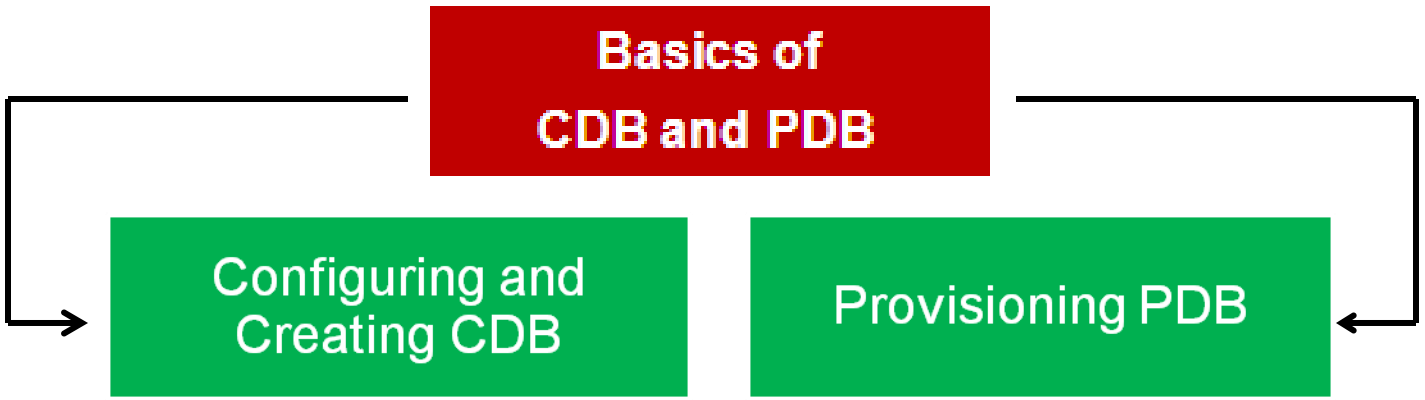
ORACLE

Oracle Database 12*c* –
Pluggable Databases Provisioning

ORACLE

Road Map

PART – I



Goals

Why provisioning pluggable databases in a CDB ?

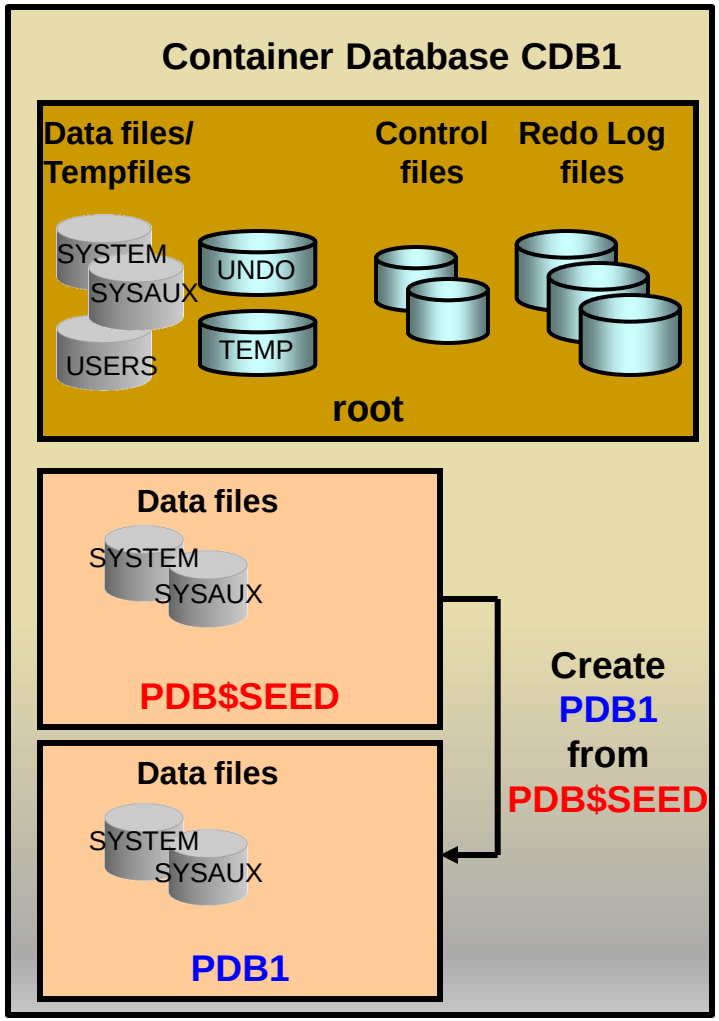
- To consolidate many non-CDBs into a single, larger database, the CDB
  - To lower costs
  - To better exploit hardware and DBA resources
- To avoid the creation of new databases and instances when:
  - Implementing new application
  - Testing new applications
  - Diagnosing new application performance
- To simplify and reduce time for patching and upgrade

Provisioning New Pluggable Databases

Four methods:

- Create a new PDB from the seed PDB
- Plug a non-CDB in a CDB
- Clone a PDB from another PDB:
  - Into the same CDB
  - Into another CDB
- Plug an unplugged PDB into another CDB

Method 1 – Create New PDB from PDB$SEED



- Copies the datafiles from `PDB$SEED` datafiles
- Creates tablespaces `SYSTEM`, `SYSAUX`
- Creates a full catalog including metadata pointing to Oracle supplied objects
- Creates common users:
  - Superuser `SYS`
  - `SYSTEM`
- Optionally creates a local user granted local `PDB_DBA` role
- Creates a new default service

Steps to create a new PDB from the seed using **FILE_NAME_CONVERT** clause:

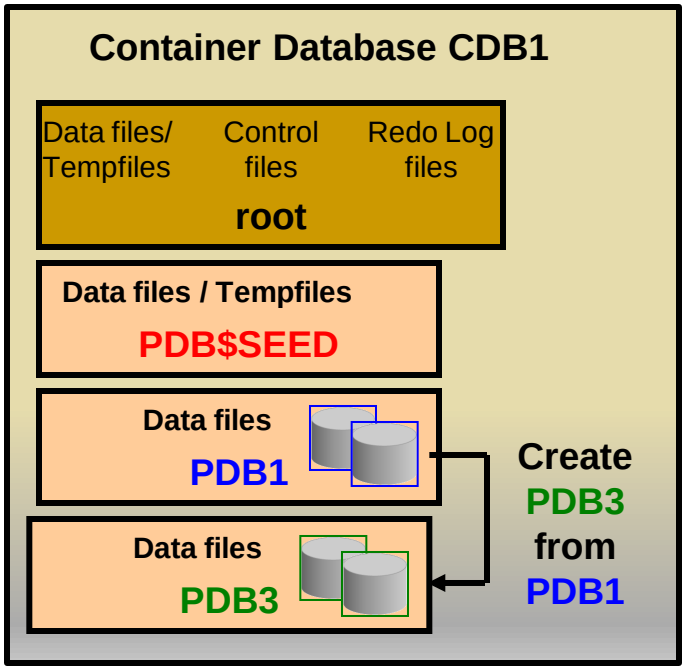
1. Connect to the root as a common user with `CREATE PLUGGABLE DATABASE` system privilege:

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

2. 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> ALTER PLUGGABLE DATABASE pdb1 OPEN;
SQL> CONNECT sys/password@pdb1 AS SYSDBA
SQL> CONNECT admin1/p1@pdb1
```

Method 2 – Clone PDBs within same CDB



PDB3 owns:

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

1. In init.ora, set `DB_CREATE_FILE_DEST='PDB3dir'` or `PDB_FILE_NAME_CONVERT='PDB1dir','PDB3dir'`
2. Connect to the root as a common user with `CREATE PLUGGABLE DATABASE` privilege
3. Quiesce **PDB1**: Close **PDB1** first, then

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

4. Clone **PDB3** from **PDB1:**

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

5. Open **PDB3**:

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

Migrating pre-12.1 Databases to 12.1 CDB

There are two methods:

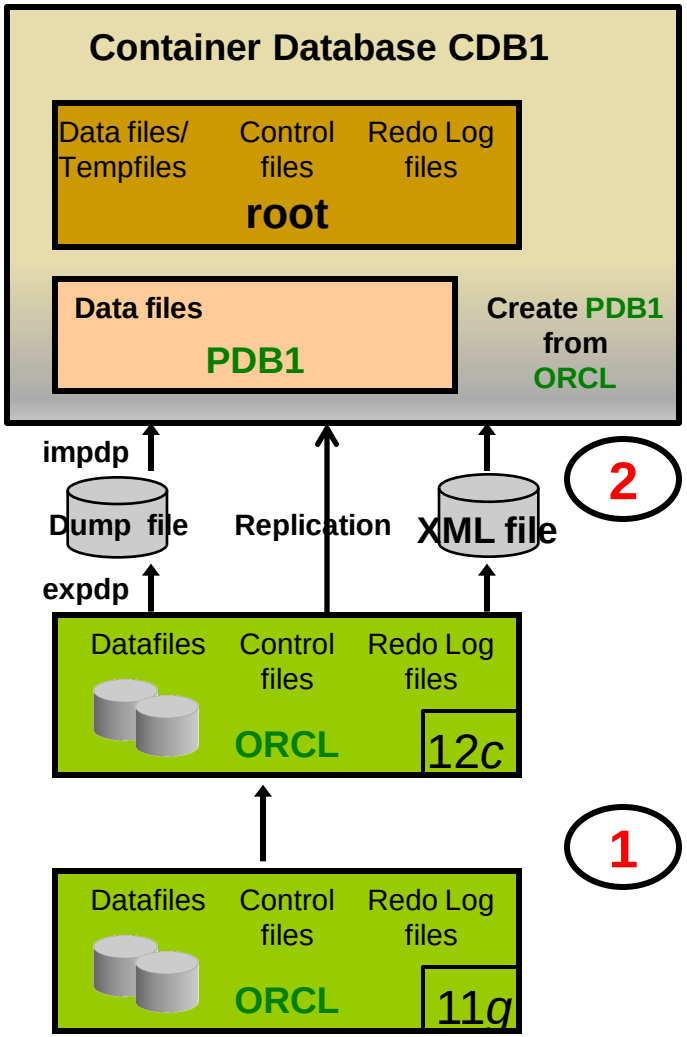
1. Upgrade an existing pre-12.1 database to 12*c*
2. Plug-in non-CDB into a CDB

Or

1. Pre-create a PDB in CDB
2. Use 11*g* expdp / 12*c* impdp

   or
2. Use replication between non-CDB and PDB

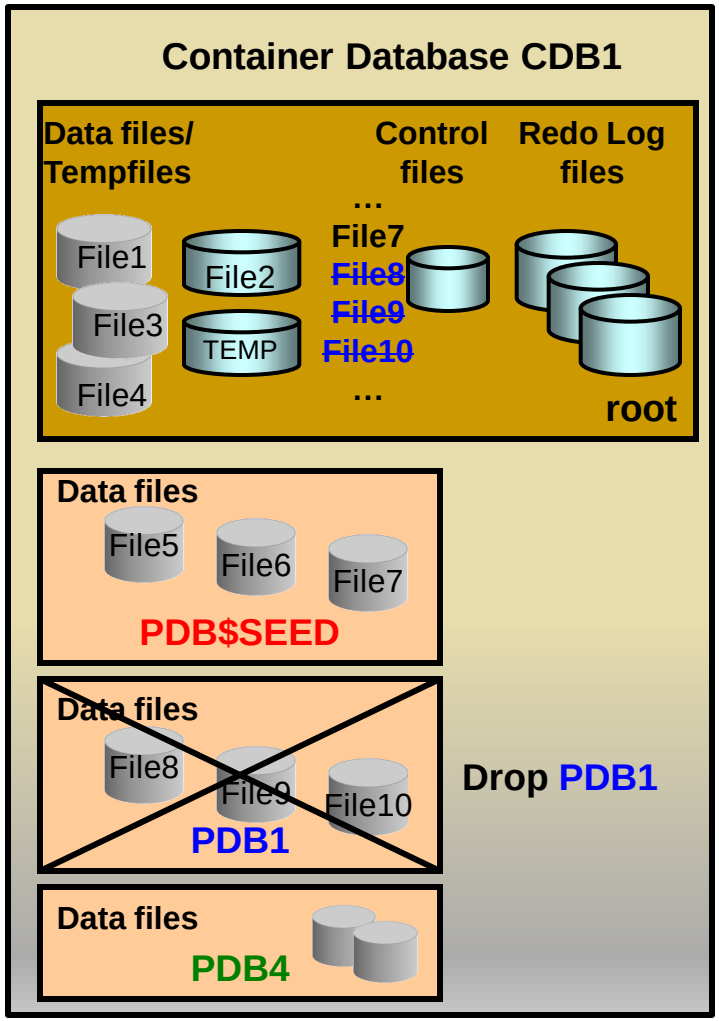
Migrating pre-12.1 Databases to 12.1 CDB



First method:

1. Upgrade an existing 11*g* database to 12*c*
2. Plug-in the non-CDB into a CDB using:
   - Export/import
     - TTS
     - TDB
     - Without transportable
   - Replication
   - `DBMS_PDB.DESCRIBE` to generate XML file

Dropping a PDB




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

- If `KEEP DATAFILES` (default):
  - Retain datafiles
  - Can be plugged in another or same CDB
- If `INCLUDING DATAFILES`:
  - Removes **PDB1** datafiles
- Updates controlfiles
- Cannot drop seed PDB

Demo: Creating PDB from Seed

PROPERTIES

Allow user to leave interaction:	Anytime
Show ‘Next Slide’ Button:	Show always
Completion Button Label:	Next Slide

Properties...

Edit in Engage

Demo: Cloning PDB from Another PDB

PROPERTIES

Allow user to leave interaction:	Anytime
Show ‘Next Slide’ Button:	Show always
Completion Button Label:	Next Slide

Properties...

Edit in Engage

Demo: Plugging a non-CDB into a CDB

PROPERTIES

Allow user to leave interaction:	Anytime
Show ‘Next Slide’ Button:	Show always
Completion Button Label:	Next Slide

Properties...

Edit in Engage

Demo: Plugging an Unplugged PDB into a CDB

PROPERTIES

Allow user to leave interaction:	Anytime
Show ‘Next Slide’ Button:	Show always
Completion Button Label:	Next Slide

Properties...

Edit in Engage

Demo: Dropping a PDB

PROPERTIES

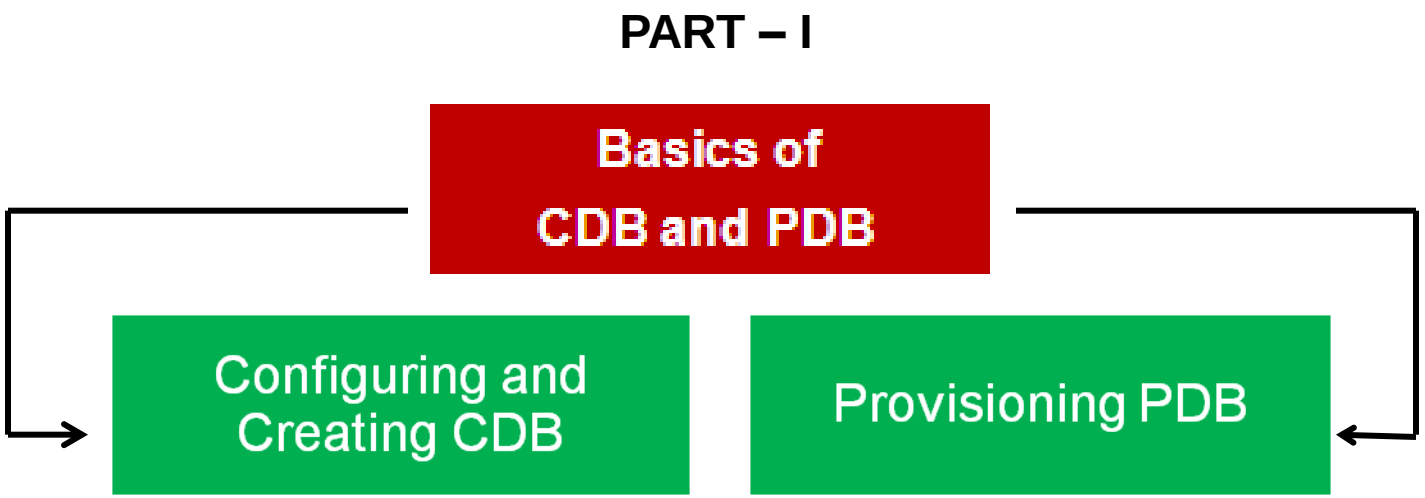
Allow user to leave interaction:	Anytime
Show ‘Next Slide’ Button:	Show always
Completion Button Label:	Next Slide

Properties...

Edit in Engage

Lesson Summary

In this third lesson, we discussed:



- The different methods to provision new pluggable databases
- The migration methods
- The impact of dropping a pluggable database

Lesson Review

Choose the right answer when cloning PDBs.

- You can clone PDBS across CDBs.
- You can only clone PDBS within the same CDB.

PROPERTIES

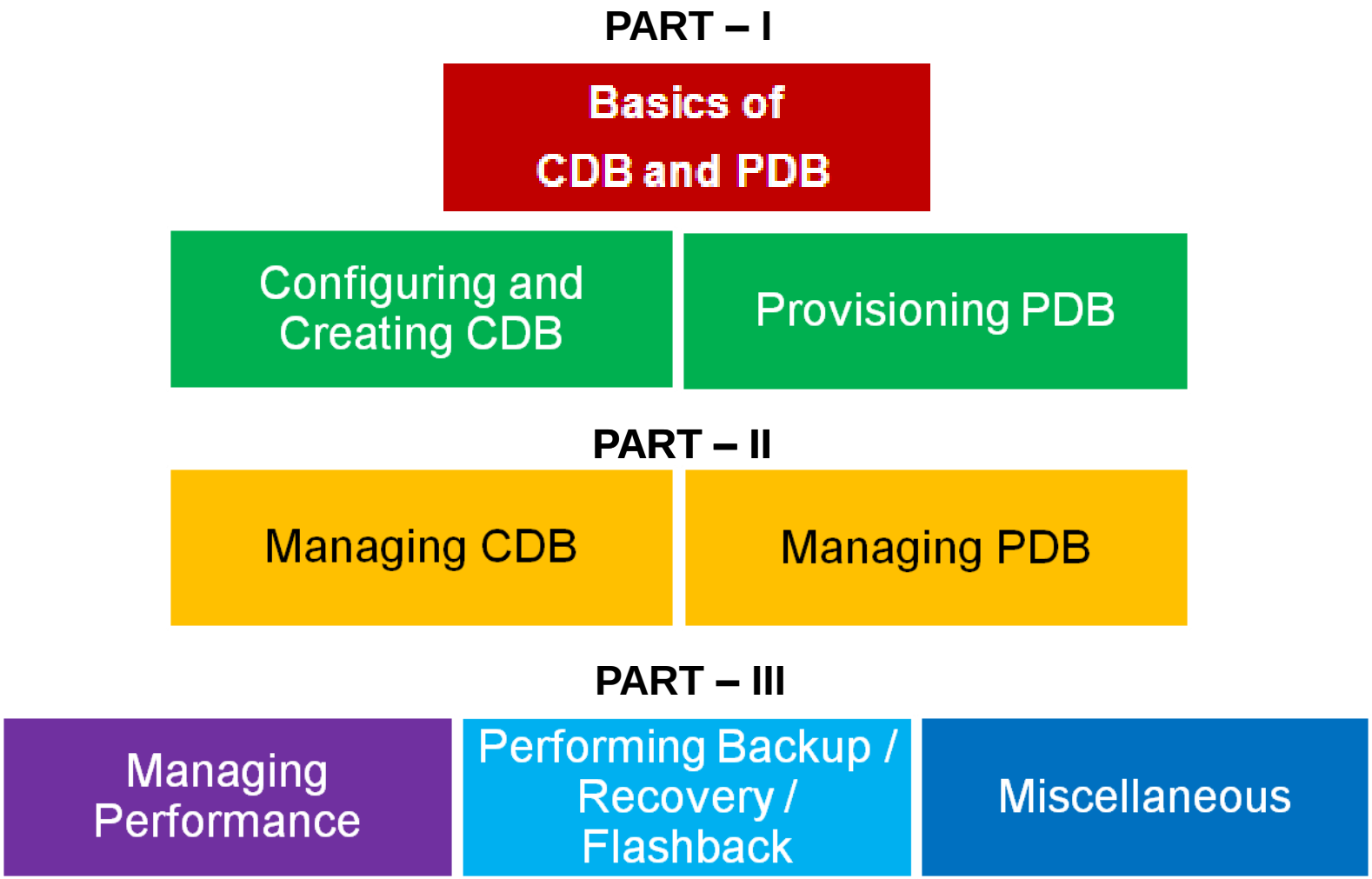
On passing, 'Finish' button: **Goes to Next Slide**
On failing, 'Finish' button: **Goes to Next Slide**
Allow user to leave quiz: **At any time**
User may view slides after quiz: **At any time**
User may attempt quiz: **Unlimited times**


Properties...


Edit in Quizmaker

Course Summary

In this first part of the course, we discussed:

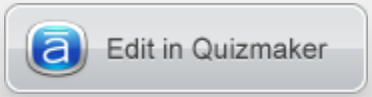


Course Review

Fill in the blank. A container database allows the consolidation of several databases and prevents redundant _____ .

PROPERTIES

On passing, 'Finish' button: **Goes to Next Slide**
On failing, 'Finish' button: **Goes to Next Slide**
Allow user to leave quiz: **At any time**
User may view slides after quiz: **At any time**
User may attempt quiz: **Unlimited times**



How Can I Learn More?

Self-Study Courses

Instructor-led Courses

Other Resources

More Information

There are a variety of channels from which you can learn more about Database 12*c*. Click on a tab on the left for more information about just a few of the possibilities.

We hope you found this sample self-study course informative and useful.

PROPERTIES

Allow user to leave interaction:	Anytime
Show 'Next Slide' Button:	Show always
Completion Button Label:	Next Slide

Properties...

Edit in Engage