Move to Oracle Database 23ai

Everything you need to know about Oracle Multitenant - part 1

Photo by John Bakator on Unsplash

Safe harbor statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described for Oracle's products may change and remains at the sole discretion of Oracle Corporation.

The materials in this presentation pertain to Oracle Health, Oracle, Oracle Cerner, and Cerner Enviza which are all wholly owned subsidiaries of Oracle Corporation. Nothing in this presentation should be taken as indicating that any decisions regarding the integration of any EMEA Cerner and/or Enviza entities have been made where an integration has not already occurred.

Oracle

DBAs

run the world

ROY SWONGER
Vice President
Database Upgrade, Utilities & Patching

royfswonger

MIKE DIETRICH

Senior Director Product Management
Database Upgrade, Migrations & Patching

in mikedietrich

@mikedietrichde

https://mikedietrichde.com

DANIEL OVERBY HANSEN
Senior Principal Product Manager
Database Upgrade, Migrations & Patching

dohdatabase

@dohdatabase

https://dohdatabase.com

RODRIGO JORGE
Senior Principal Product Manager
Database Upgrade, Migrations & Patching

rodrigoaraujorge

@rodrigojorgedba

https://dbarj.com.br/en

ALEX ZABALLA
Distinguished Product Manager
Database Upgrade, Migrations & Patching

alexzaballa

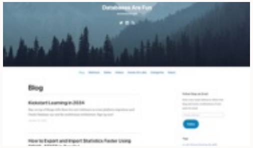
@alexzaballa

https://alexzaballa.com

Find Slides and Much More on Our Blogs

MikeDietrichDE.com

Mike.Dietrich@oracle.com



dohdatabase.com

Daniel.Overby.Hansen@oracle.com

DBArj.com.br

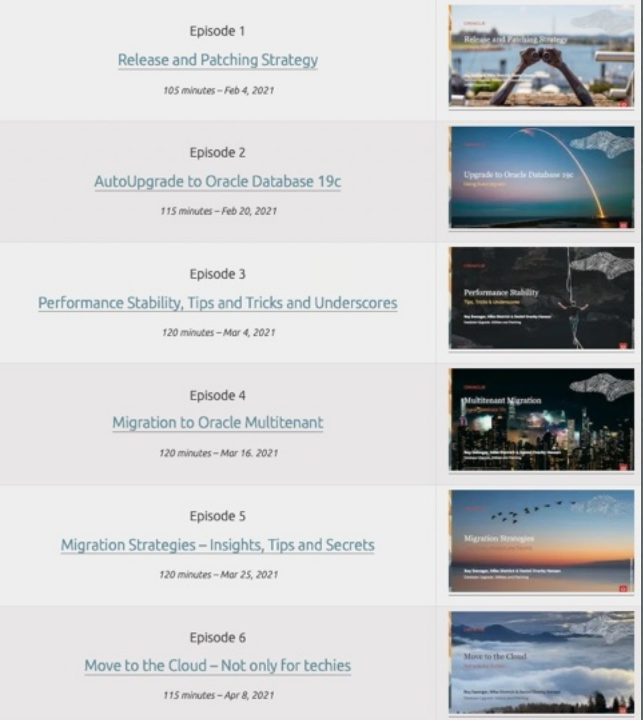
Rodrigo.R.Jorge@oracle.com

AlexZaballa.com

Alex.Zaballa@oracle.com

Get the Slides

https://dohdatabase.com/slides



Episode 1
Release and Patching Strategy
105 minutes – Feb 4, 2021

Episode 2
AutoUpgrade to Oracle Database 19c
115 minutes – Feb 20, 2021

Episode 3
Performance Stability, Tips and Tricks and Underscores
120 minutes – Mar 4, 2021

Episode 4
Migration to Oracle Multitenant
120 minutes – Mar 16. 2021

Episode 5
Migration Strategies – Insights, Tips and Secrets
120 minutes – Mar 25, 2021

Episode 6
Move to the Cloud – Not only for techies
115 minutes – Apr 8, 2021

Recorded Web Seminars

https://MikeDietrichDE.com/videos

More than 35 hours of technical content, on-demand, anytime, anywhere

Part 2

Move to Oracle Database 23ai
- Everything you need to know about Oracle Multitenant

Live on June 27, 14:00 CEST
Sign up

Architecture

Multitenant Evolution

2013

12c

- Oracle Database 12.1
- Multitenant is born
- Unplug-plug
- Cold Cloning

2016

12c

- Oracle Database 12.2
- Hot Clones
- Refreshable PDB
- Online PDB Relocation
- Application Container

2018

18c

- Snapshot Carousel
- Refreshable PDB Switchover
- CDB Fleet

2019

19c

- Foundation for ADB
- NetSuite 24k tenant PDBs

Starting with Oracle Database 21c, installation of non-CDB Oracle Database architecture is no longer supported

Upgrade Guide, 23ai

Once you upgrade beyond Oracle Database 19c, you must convert to the multitenant architecture

Oracle Database 19c is the last release to support the non-CDB architecture

Next Long Term Support release

Oracle Database 23ai

Upgrade possible only from:

- Oracle Database 19c
- Oracle Database 21c

Multitenant enables an Oracle database to function as a container database

Generally, you don't need to change your application to use a pluggable database

Single vs. Multitenant

Single Tenant

One PDB
No extra license

Multitenant

Multiple PDBs
Extra license if more than 3 PDBs

```
--Use up to 3 user-created PDBs
--without a license for Multitenant option.
--Applies to Oracle Database 19c and newer, including SE2

alter system set max_pdbs=3;
```

Multitenant

1

2

3

You always create a new CDB

- `CREATE DATABASE ... ENABLE PLUGGABLE DATABASE`
- Using DBCA

Or clone an existing CDB

Multitenant

1

2

3

When you create a new CDB, it contains:

- The root container
- The seed container

Multitenant

1

2

3

You can create PDBs:

- From the seed container
- By cloning other PDBs
- By converting a non-CDB

Be cautious making changes to *PDB$SEED*

- Your own customizations do not belong *PDB$SEED*

Containers

CDB$ROOT is always 1

PDB$SEED is always 2

User-created PDBs have ID 3 or above

```
SQL> select con_id, name
     from   v$containers;


   CON_ID         NAME
_________ ____________
        1     CDB$ROOT
        2     PDB$SEED
        3         PDB1
        4         PDB2
```

```
alter session set container=CDB$ROOT;
show con_id

CON_ID
------------------------------
1

alter session set container=PDB1;
select sys_context('USERENV', 'CON_ID') as con_id from dual;

CON_ID
------------------------------
3
```

You can switch between containers,
but a transaction belong to one PDB only

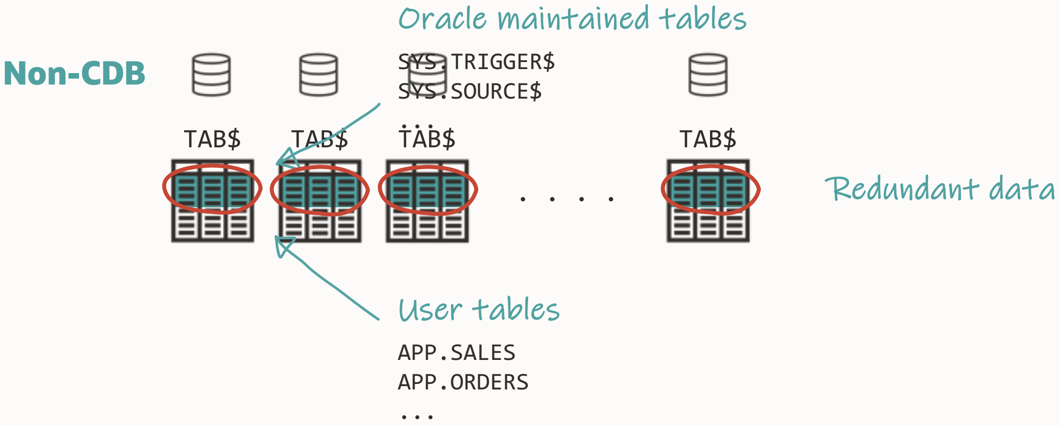
```
alter session set container=PDB1;
insert into table1 values (...);
```

```
alter session set container=PDB2;
insert into table2 values (...);
```

```
ORA-65023: active transaction exists in container PDB1
```

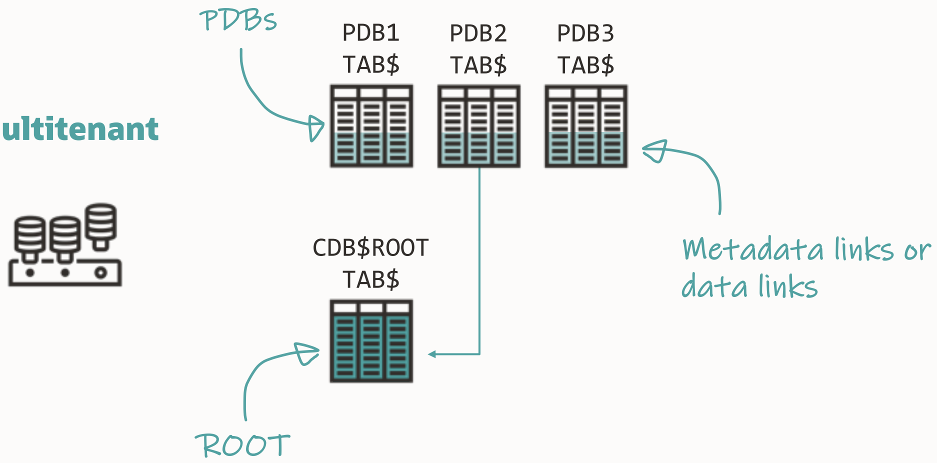
Multitenant implements data dictionary separation

Data Dictionary Architecture



Data Dictionary Architecture

Multitenant



Data Dictionary Architecture

Deduplication

By storing data just once, you can save space in the data dictionary.

Faster

Smaller dictionaries take less time to patch or upgrade.

Easier

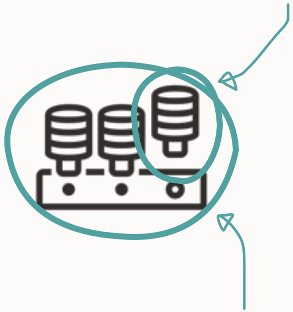
With much metadata stored in root, there is less work for a patch or upgrade.

CDB views describe the entire CDB including all PDBs

- Column **CON_ID** indicates the originating container

Data Dictionary Architecture

```
SELECT * FROM USER_OBJECTS
              ALL_OBJECTS
              DBA_OBJECTS
```



```
SELECT * FROM CDB_OBJECTS
```

Applies to any
data dictionary view

```
CDB_ALL_TABLES
CDB_ANALYTIC_VIEW_ATTR_CLASS
.
.
.
CDB_XTERNAL_TAB_SUBPARTITIONS
```

Originating container

```
alter session set container=CDB$ROOT;

select con_id, tablespace_name
from   cdb_tablespaces;
```

```
CON_ID TABLESPACE_NAME
------ ---------------
     1 SYSTEM
     1 SYSAUX
     1 UNDOTBS1
     1 TEMP
     1 USERS
     2 SYSTEM
     2 SYSAUX
     2 UNDOTBS1
     2 TEMP
     4 SYSTEM
     4 SYSAUX
     4 UNDOTBS1
     4 TEMP
```

PDB$SEED information, hidden by default

```
alter session set container=PDB1;

select tablespace_name from dba_tablespaces;
```

```
TABLESPACE_NAME
---------------
SYSTEM
SYSAUX
UNDOTBS1
TEMP
```

A PDB never sees information from other PDBs

```
alter session set container=CDB$ROOT;
select count(*) from cdb_objects;

   COUNT(*)
-----------
     114658

alter session set container=PDB1;
select count(*) from cdb_objects;

   COUNT(*)
-----------
      23980
```

You define common objects in root, and they are available in all PDBs

- Most options in application containers

```
alter session set container=CDB$ROOT;

create user C##OPS identified by oracle;
grant create session to C##OPS container=all;

create pluggable database pdb1 ... ;
alter pluggable database pdb1 open;

conn C##OPS/oracle@pdb1
Connected.
```

Common Objects

You can define common:

- Profiles
- Roles
- User
- Audit configuration
- Other objects available in application containers

Useful for:

- Maintenance and monitoring users
- Self-service functionality
- Separation of duties
- Enforcing security

You can change the common prefix with `COMMON_USER_PREFIX`

- We do not recommend changing it
- Use extreme care if you choose to do so

Create same set of common objects in all CDBs to avoid issues during clone/relocate

The database creates common directories

```
alter session set container=PDB1;

select directory_path, origin_con_id
from   cdb_directories
where  directory_name='DATA_PUMP_DIR';

DIRECTORY_PATH                                                   ORIGIN_CON_ID
/u01/app/oracle/admin/CDB2/dpdump/13D6BC6605416ECEE065000000000001             1

create or replace directory DATA_PUMP_DIR AS '/tmp';

ERROR at line 1:
ORA-65040: operation not allowed from within a pluggable database

drop directory DATA_PUMP_DIR;

ERROR at line 1:
ORA-65040: operation not allowed from within a pluggable database
```

PDB GUID

Common directory, created in root

Use your own directories
if you want to decide on the directory path

You make most configuration in the root container

```
alter session set container=PDB1;

alter database backup controlfile to trace;

ORA-65040: operation not allowed from within a pluggable
database
```

Non-CDB Compatible

- Some `ALTER DATABASE` and `ALTER SYSTEM` commands fail in a PDB

- Enable non-CDB compatibility by setting `NONCDB_COMPATIBLE=TRUE`
  - When you can't change the application
  - When you accept the reduced security

```
SQL> alter system set noncdb_compatible=true;
SQL> shutdown immediate
SQL> startup
SQL> alter system set noncdb_compatible=true;
SQL> shutdown immediate
SQL> startup

SQL> alter session set container=PDB1;
SQL> alter database backup controlfile to trace;

Database altered.
```

Fine-tune PDBs with instance parameters

- Parameters apply to PDBs as well
- Some parameters are PDB modifiable

```
SQL> select name from v$system_parameter where ispdb_modifiable='TRUE';

NAME
__________________________________
adg_account_info_tracking
allow_rowid_column_type
approx_for_aggregation
approx_for_count_distinct
approx_for_percentile
.
.
.
xml_handling_of_invalid_chars

246 rows selected.
```

Use ORAdiff to find PDB modifiable parameters

- Free tool
- https://oradiff.oracle.com

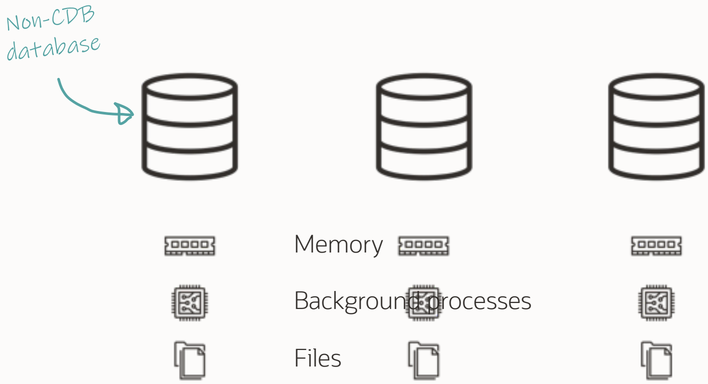
A cloned or moved PDB keeps the changed parameters

- Certain exceptions exist

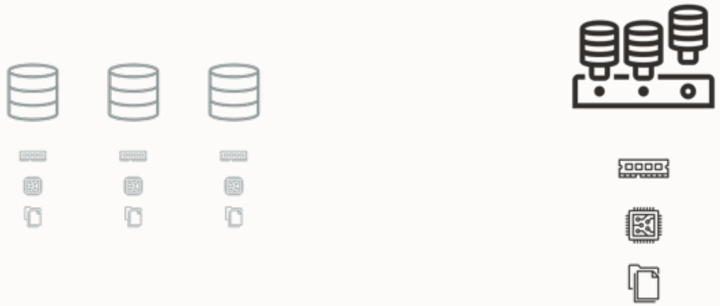
```
--Find specific parameters that has been defined in a specific PDB

select name, value from v$system_parameter where con_id=<id>;
```

Share resources between PDBs

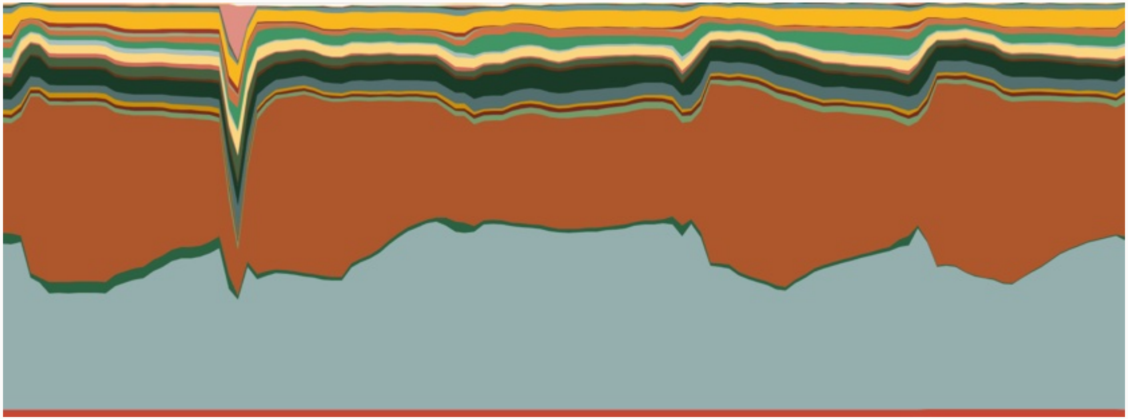
Resource Consolidation



Resource Consolidation



Resource Consolidation



Consolidation Strategies?

There is no "*best*" strategy

- Don't mix PDBs with different SLAs
- Total memory consumption minus 20%-30%
- Increase consolidation factor slowly
- Constrain PDBs with instance caging (CPU_COUNT)
- Be open to start with a completely new naming schema

Using a Swingbench benchmark,
a single-core machine could host nine non-CDBs
before reaching 75 % CPU utilization

By going multitenant the number of databases reached 123 PDBs

A US Health Care provided managed to

- Reduce the number of database instances by 7x
- Reduce the number of physical servers by 50 %

You can run multiple CDBs on the same host and out of the same Oracle home

Consolidation

Schema consolidation

Virtual Private Database

⇛

PDB consolidation

- Less complexity
- Better isolation
- Operational benefits
- Easier cloning

A global provider of financial services states

The multitenant architecture gives us complete client separation out of the box, without having to maintain a Virtual Private Database setup.

We went away from Virtual Private Database and consolidated our different clients in individual PDBs.

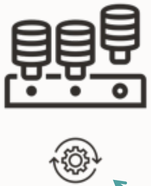
This reduced the complexity of our database implementation and made operations much easier.

The *many-as-one* principle eases maintenance operations

Many-as-one

Patch databases
one by one

Many-as-one



Many-as-one

Applies to:

- Upgrading
- Patching
- Configuring and performing backups backups
- Configuring Data Guard
- Configuring RAC
- Monitoring
- ... and many other operations

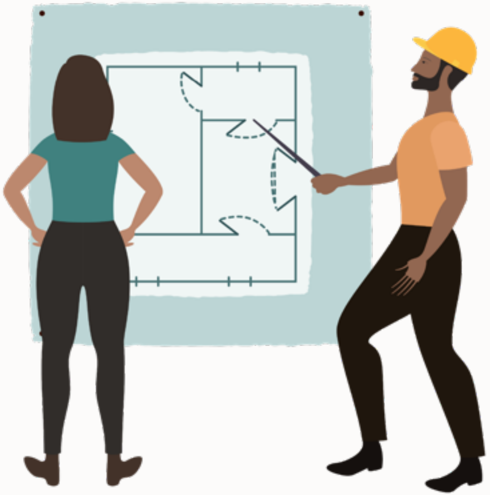
Benefits

The multitenant architecture enables

1 Self-contained PDBs

2 Common and easier operations

3 Resource sharing and consolidation



Create

Create Container Database

1 Character set

2 Components

3 `COMPATIBLE`

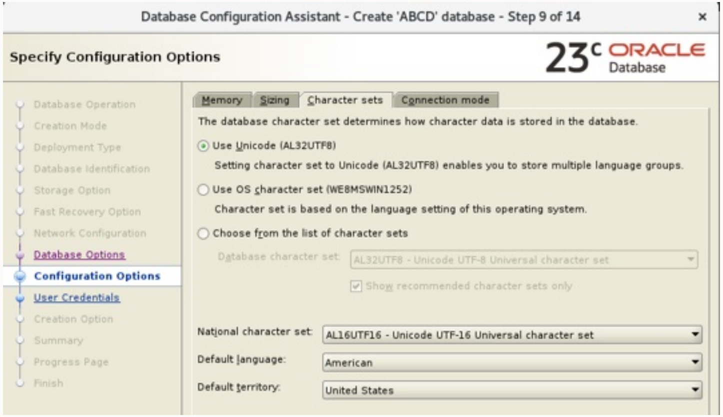
Create Container Database

1 Character set

- Always choose AL32UTF8
- Allows PDBs with any character set

2 Components

3 COMPATIBLE


Database Configuration Assistant - Create 'ABCD' database - Step 9 of 14

Specify Configuration Options

The database character set determines how character data is stored in the database.

(●) Use Unicode (AL32UTF8)

Setting character set to Unicode (AL32UTF8) enables you to store multiple language groups.

( ) Use OS character set (WE8MSWIN1252)

Character set is based on the language setting of this operating system.

( ) Choose from the list of character sets

Database character set: AL32UTF8 - Unicode UTF-8 Universal character set

[✓] Show recommended character sets only

National character set: AL16UTF16 - Unicode UTF-16 Universal character set

Default language: American

Default territory: United States

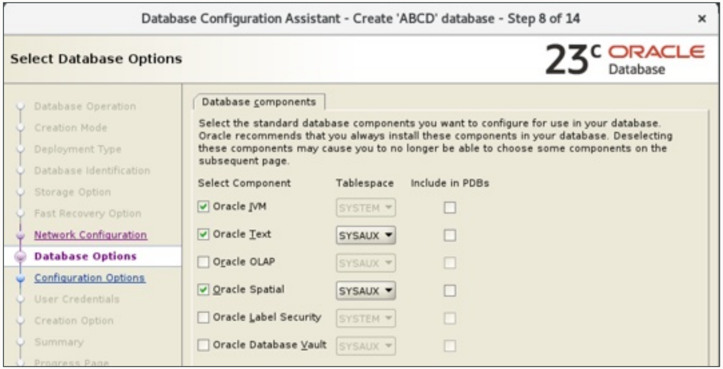
Create Container Database

1 Character set

2 Components

- Install as many as you need
- No more than that

3 COMPATIBLE



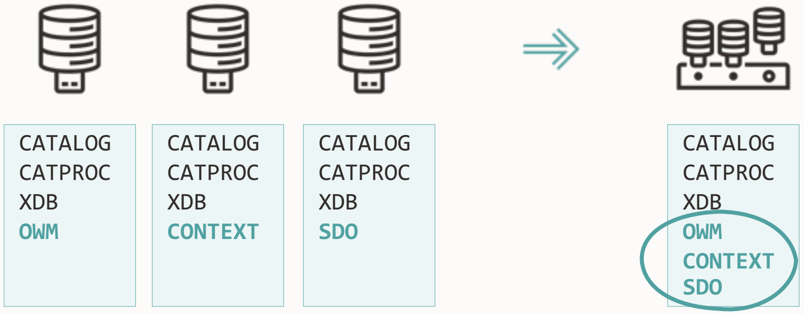
Database Configuration Assistant - Create 'ABCD' database - Step 8 of 14

Select Database Options

Select the standard database components you want to configure for use in your database. Oracle recommends that you always install these components in your database. Deselecting these components may cause you to no longer be able to choose some components on the subsequent page.

Select Component	Tablespace	Include in PDBs
✓ Oracle JVM	SYSTEM	☐
✓ Oracle Text	SYSAUX	☐
☐ Oracle OLAP	SYSAUX	☐
✓ Oracle Spatial	SYSAUX	☐
☐ Oracle Label Security	SYSTEM	☐
☐ Oracle Database Vault	SYSAUX	☐

Components


CATALOG
CATPROC
XDB
OWM
CATALOG
CATPROC
XDB
CONTEXT
CATALOG
CATPROC
XDB
SDO
CATALOG
CATPROC
XDB
OWM
CONTEXT
SDO

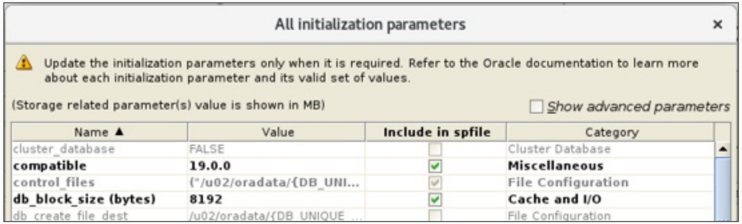
Create Container Database

1 Character set

2 Components

3 `COMPATIBLE`

- Keep at the default setting, *23.4.0*
- Unless you want the option of downgrade


All initialization parameters

Update the initialization parameters only when it is required. Refer to the Oracle documentation to learn more about each initialization parameter and its valid set of values.

(Storage related parameter(s) value is shown in MB)

☐ Show advanced parameters

Name ▲	Value	Include in spfile	Category
cluster_database	FALSE	☐	Cluster Database
compatible	19.0.0	✓	Miscellaneous
control_files	("/u02/oradata/{DB_UNI...	✓	File Configuration
db_block_size (bytes)	8192	✓	Cache and I/O
db_create_file_dest	/u02/oradata/{DB_UNIQUE ...	☐	File Configuration

```
-- Always set compatible to the default of a release
-- Use three digits only

alter system set compatible='23.4.0' scope=spfile;
```

```
-- Should I change compatible when patching?
-- No, this is a bad idea

alter system set compatible='23.5.0' scope=spfile;
```

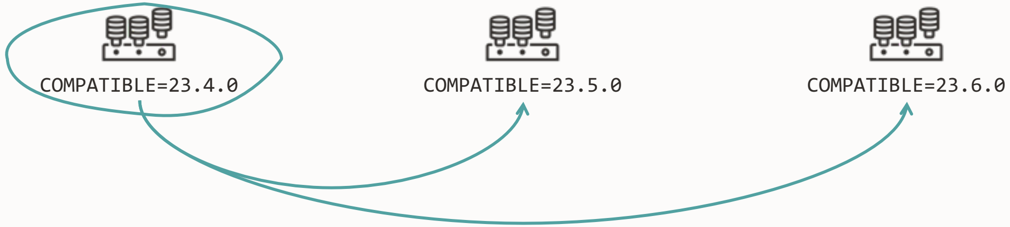
Compatible

- On plug-in, a PDB adopts `COMPATIBLE` of the CDB silently and without confirmation

- Changing `COMPATIBLE` is irreversible

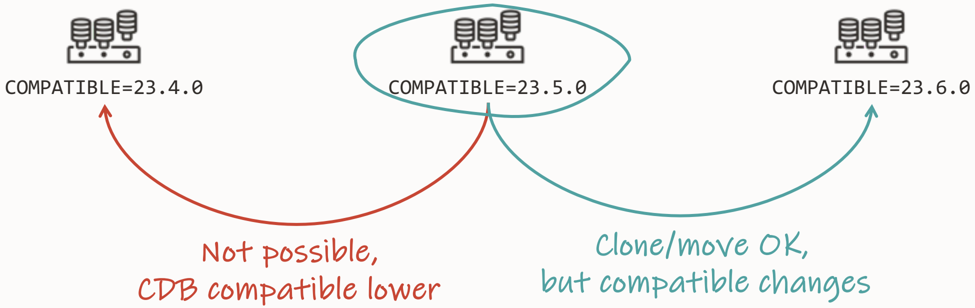
- Might prevent future move or clone of the PDB

Compatible



Clone/move OK,
but compatible changes

Compatible


COMPATIBLE=23.4.0
COMPATIBLE=23.5.0
COMPATIBLE=23.6.0
Not possible,
CDB compatible lower
Clone/move OK,
but compatible changes

Keep `COMPATIBLE` at the same setting in all CDBs on the same release

```
--Allows CDB views to include information on PDB$SEED objects.
--By default, such information is hidden.
--https://mikedietrichde.com/2017/07/21/why-exclude_seed_cdb_view-is-now-an-underscore-in-oracle-12-2/

alter system set "_exclude_seed_cdb_view"=false;
alter session set container=CDB$ROOT;


alter system set "_exclude_seed_cdb_view"=false;


select con_id, tablespace_name
from   cdb_tablespaces;
```

```
CON_ID TABLESPACE_NAME
------ ---------------
     1 SYSTEM
     1 SYSAUX
     1 UNDOTBS1
     1 TEMP
     1 USERS
     2 SYSTEM
     2 SYSAUX
     2 UNDOTBS1
     2 TEMP
     4 SYSTEM
     4 SYSAUX
     4 UNDOTBS1
     4 TEMP
```

Create Container Database

https://mikedietrichde.com/2018/08/08/creating-cdbs-non-cdbs-with-less-options/
https://mikedietrichde.com/2017/07/11/always-create-custom-database/
https://mikedietrichde.com/2017/07/26/remove-clean-components-oracle-11-2-12-2/

Migration

Migration to multitenant is a one-time operation that requires downtime

- No downtime when using Oracle GoldenGate

Migration

1 Plug in

2 Convert

Migration

1 Plug in

First, check if database is compatible with CDB

1. Generate manifest file in non-CDB
2. Check compatibility in CDB

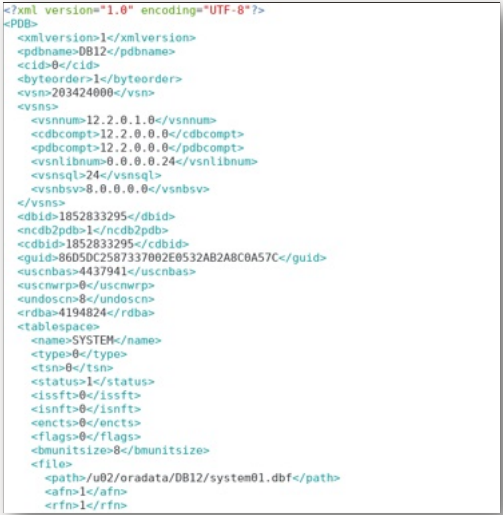
```
--In source, generate manifest file
--You can also generate a manifest file of a remote database using a db link
--You can generate a manifest file on a running database

exec dbms_pdb.describe('/tmp/DB19.xml');
```

Manifest File

What is a manifest file

- Data files
- Components
- Parameters
- Services
- Patch level
- Time zone

  ... and more



```
<?xml version="1.0" encoding="UTF-8"?>
<PDB>
  <xmlversion>1</xmlversion>
  <pdbname>DB12</pdbname>
  <cid>0</cid>
  <byteorder>1</byteorder>
  <vsn>203424000</vsn>
  <vsns>
    <vsnnum>12.2.0.1.0</vsnnum>
    <cdbcompt>12.2.0.0.0</cdbcompt>
    <pdbcompt>12.2.0.0.0</pdbcompt>
    <vsnlibnum>0.0.0.0.24</vsnlibnum>
    <vsnsql>24</vsnsql>
    <vsnbsv>8.0.0.0.0</vsnbsv>
  </vsns>
  <dbid>1852833295</dbid>
  <ncdb2pdb>1</ncdb2pdb>
  <cdbid>1852833295</cdbid>
  <guid>86D5DC2587337002E0532AB2A8C0A57C</guid>
  <uscnbas>4437941</uscnbas>
  <uscnwrp>0</uscnwrp>
  <undoscn>8</undoscn>
  <rdba>4194824</rdba>
  <tablespace>
    <name>SYSTEM</name>
    <type>0</type>
    <tsn>0</tsn>
    <status>1</status>
    <issft>0</issft>
    <isnft>0</isnft>
    <encts>0</encts>
    <flags>0</flags>
    <bmunitsize>8</bmunitsize>
    <file>
      <path>/u02/oradata/DB12/system01.dbf</path>
      <afn>1</afn>
      <rfn>1</rfn>
```

```
--In CDB, check compatibility
--If PDB name changes, add parameter to check_plug_compatibility

set serveroutput on
BEGIN
   IF dbms_pdb.check_plug_compatibility('/tmp/DB19.xml') THEN
      dbms_output.put_line('PDB compatible? ==> Yes');
   ELSE
      dbms_output.put_line('PDB compatible? ==> No');
   END IF;
END;
/
```

```
--Always check the details

select type, message
from   pdb_plug_in_violations
where  name='DB19' and status<>'RESOLVED';

TYPE       MESSAGE
______________________________________________________________________________________________________
ERROR      '19.9.0 Release_Update' is installed in the CDB but no release updates are installed in the PDB
ERROR      DBRU bundle patch 201020: Not installed in the CDB but installed in the PDB
ERROR      PDB's version does not match CDB's version: PDB's version 12.2.0.1.0. CDB's version 19.0.0.0.0.
WARNING    CDB parameter compatible mismatch: Previous '12.2.0' Current '19.0.0'
WARNING    PDB plugged in is a non-CDB, requires noncdb_to_pdb.sql be run.
```

Always check `PDB_PLUG_IN_VIOLATIONS` after any plug-in operation

- An error prevents the PDB from opening unrestricted

Migration

1 Plug in

Then, perform plug-in

1. Shut down non-CDB
2. Plug into CDB

Migration

1. Restart database in read-only mode

```
SQL> shutdown immediate
SQL> startup mount
SQL> alter database open read only;
```

2. Generate manifest file and shut down

```
SQL> exec dbms_pdb.describe('/tmp/DB19.xml');
SQL> shutdown immediate;
```

3. In CDB, create PDB from manifest file

```
SQL> create pluggable database DB19 using '/tmp/DB19.xml';
```

Migration

→

1 Plug in

2 Convert

Migration

2 Convert

1. Complete conversion with `noncdb_to_pdb.sql`
2. Requires downtime, but you run it only once
3. Irreversible

Migration

1. Open PDB

```
SQL> alter pluggable database DB19 open;
SQL> alter session set container=DB19;
```

2. Convert

```
SQL> @?/rdbms/admin/noncdb_to_pdb.sql
```

3. Restart PDB

```
SQL> alter pluggable database DB19 close;
SQL> alter pluggable database DB19 open;
```

Migration

4. Check plug-in violations

```
SQL> select type, message
     from   pdb_plug_in_violations
     where  name='DB19' and status<>'RESOLVED';
```

5. Ensure PDB is open `READ WRITE` and unrestricted

```
SQL> select open_mode, restricted from v$pdbs;
```

6. Configure PDB to auto-start

```
SQL> alter pluggable database DB19 save state;
```

Demo

Multitenant migration
19c non-CDB

Watch on YouTube

Downtime

How much downtime do we need?

`noncdb_to_pdb.sql`

- Runtime varies, typically 10-20 minutes
- Depends on dictionary complexity
- Not database size

Pre-tasks and post-tasks

- Follow best practices
- Data Guard

Expect total downtime 1-2 hours

- Simple databases even faster
- It's a migration, don't rush it

MIGRATION
options

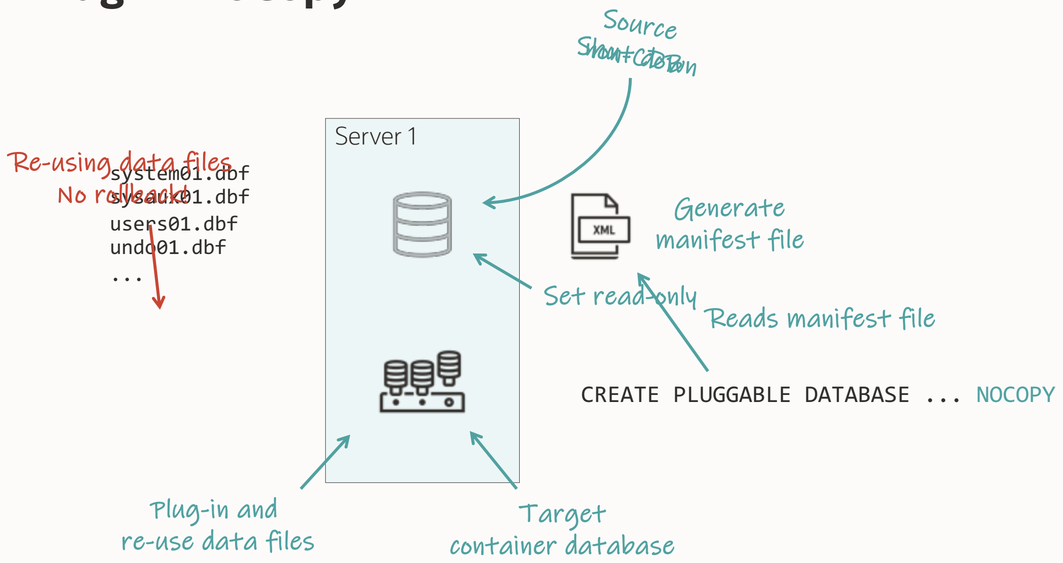
Other Options

Refreshable

Plug-in Copy

Plug-in NoCopy

Plug-in NoCopy



```
system01.dbf
sysaux01.dbf
users01.dbf
undo01.dbf
...
```

```
CREATE PLUGGABLE DATABASE ... NOCOPY
```

```
#AutoUpgrade config file
#Convert DB12 to PDB in CDB2 - re-use data files
upg1.source_home=/u01/app/oracle/product/19.22.0
upg1.target_home=/u01/app/oracle/product/19.23.0
upg1.sid=DB12
upg1.target_cdb=CDB2

#Fully automated migration
java -jar autoupgrade.jar -config db12.cfg -mode deploy
```

```
#AutoUpgrade config file
#Convert DB12 to PDB in CDB2 - re-use data files
#Upgrade from Oracle Database 19c to 23ai
upg1.source_home=/u01/app/oracle/product/19.22.0
upg1.target_home=/u01/app/oracle/product/23.4.0
upg1.sid=DB12
upg1.target_cdb=CDB2
```

Demo

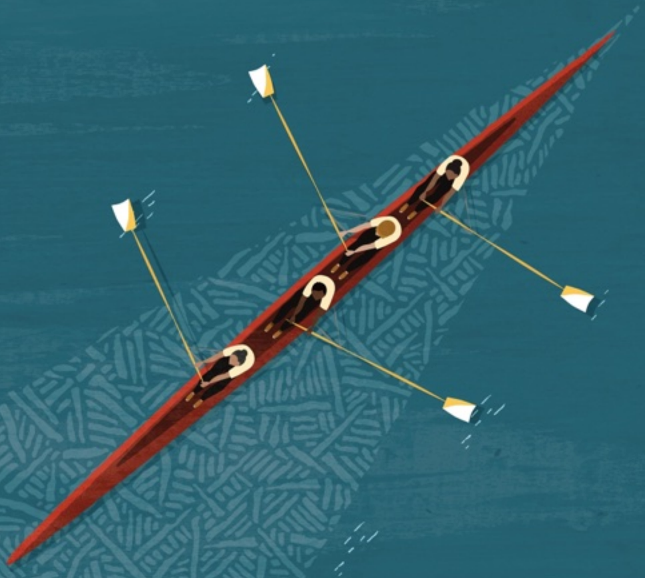
Multitenant migration
Including upgrade to Oracle Database 23ai
Using AutoUpgrade

Watch on YouTube

Consider using `MOVE` instead of `NOCOPY` to move files into proper directory

- Use `FILE_NAME_CONVERT` clause

MIGRATION
options



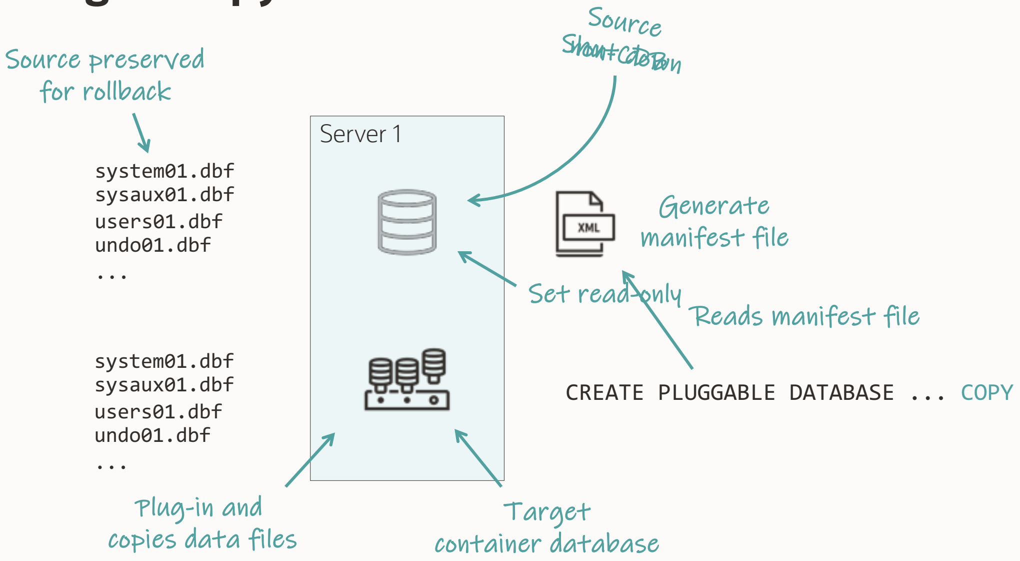
Other Options

Refreshable

Plug-in Copy

Plug-in NoCopy

Plug-in Copy



```
CREATE PLUGGABLE DATABASE ... COPY
```

```
#AutoUpgrade config file
#Convert DB12 to PDB in CDB2 - copy data files
upg1.source_home=/u01/app/oracle/product/19.22.0
upg1.target_home=/u01/app/oracle/product/19.23.0
upg1.sid=DB12
upg1.target_cdb=CDB2
upg1.target_pdb_copy_option.db12=file_name_convert('/u01', '/u02')

#Fully automated migration
java -jar autoupgrade.jar -config db12.cfg -mode deploy
```

```
#AutoUpgrade config file
#Convert DB12 to PDB in CDB2 - copy data files
#Generate OMF names - also for ASM
upg1.source_home=/u01/app/oracle/product/19.22.0
upg1.target_home=/u01/app/oracle/product/19.23.0
upg1.sid=DB12
upg1.target_cdb=CDB2
upg1.target_pdb_copy_option.db12=file_name_convert=NONE
```

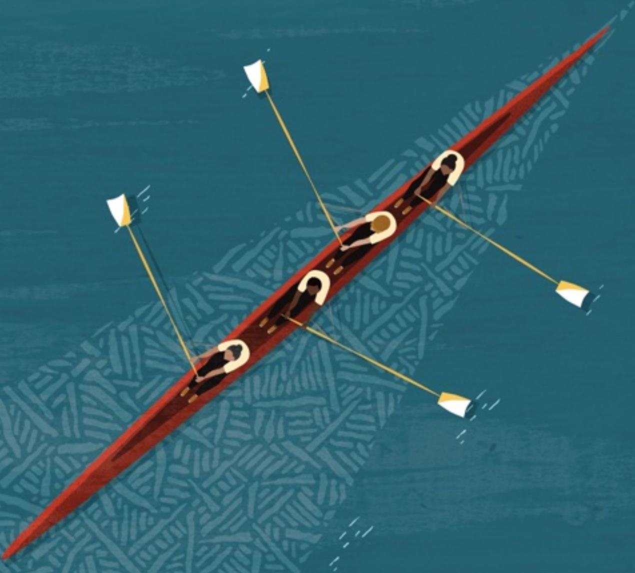
Generate OMF names
Also for ASM

Be sure to shut down the source database.

Use *`prefix`*`.close_source=yes`

MIGRATION

options



Other Options

Refreshable

Plug-in Copy

Plug-in NoCopy

Refreshable Clone

CREATE

Create PDB from non-CDB over a database link

REFRESH

Apply redo from non-CDB to keep PDB up-to-date

OUTAGE

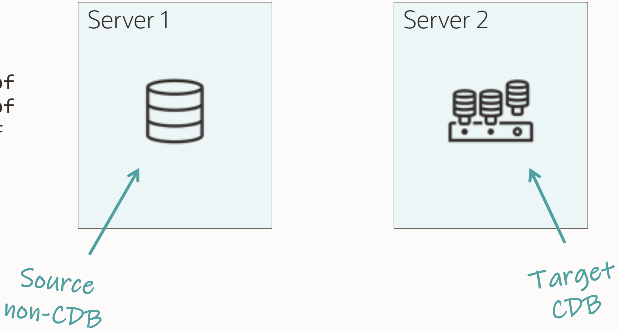
Disconnect users and refresh PDB for the last time

CONVERT

To become a proper PDB, it must be converted

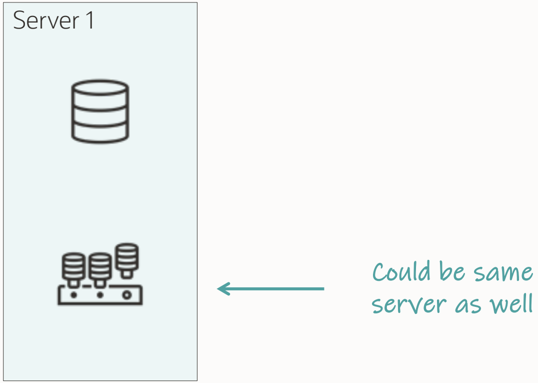
Refreshable Clone PDB

```
system01.dbf
sysaux01.dbf
users01.dbf
undo01.dbf
...
```

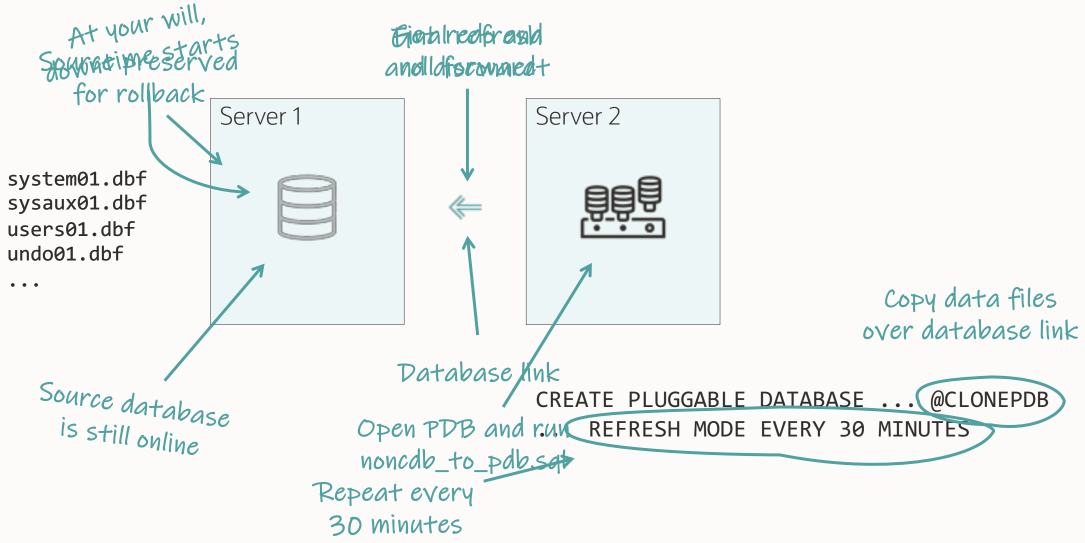


Refreshable Clone PDB

```
system01.dbf
sysaux01.dbf
users01.dbf
undo01.dbf
...
```



Refreshable Clone PDB


At your will, downtime starts
Source preserved for rollback
Final refresh and cutover
[illegible]
Server 1
Server 2
system01.dbf
sysaux01.dbf
users01.dbf
undo01.dbf
...
Copy data files over database link
Database link
Source database is still online
Open PDB and run noncdb_to_pdb.sql
Repeat every 30 minutes

```
CREATE PLUGGABLE DATABASE ... @CLONEPDB
... REFRESH MODE EVERY 30 MINUTES
```

Refreshable Clone

Source non-CDB | **Target CDB**

```
CREATE USER dblinkuser
       IDENTIFIED BY ... ;

GRANT CREATE SESSION,
      CREATE PLUGGABLE DATABASE,
      SELECT_CATALOG_ROLE TO dblinkuser;

GRANT READ ON sys.enc$ TO dblinkuser;
```

```
CREATE DATABASE LINK CLONEPDB
       CONNECT TO dblinkuser
       IDENTIFIED BY ...
       USING 'noncdb-alias';
```

Refreshable Clone

Source non-CDB	Target CDB

```
upg1.source_home=/u01/app/oracle/product/19
upg1.target_home=/u01/app/oracle/product/23
upg1.sid=NONCDB1
upg1.target_cdb=CDB1
upg1.source_dblink.NONCDB1=CLONEPDB
upg1.target_pdb_name.NONCDB1=PDB1

--Specify relative start time
--upg1.start_time=+1h30m
```

Refreshable Clone

Source non-CDB	Target CDB

```
upg1.source_home=/u01/app/oracle/product/19
upg1.target_home=/u01/app/oracle/product/23
upg1.sid=NONCDB1
upg1.target_cdb=CDB1
upg1.source_dblink.NONCDB1=CLONEPDB 300
upg1.target_pdb_name.NONCDB1=PDB1

--Specify relative start time
--upg1.start_time=+1h30m
```

Refreshable Clone

Source non-CDB	Target CDB

```
upg1.source_home=/u01/app/oracle/product/19
upg1.target_home=/u01/app/oracle/product/23
upg1.sid=NONCDB1
upg1.target_cdb=CDB1
upg1.source_dblink.NONCDB1=CLONEPDB 300
upg1.target_pdb_name.NONCDB1=PDB1
upg1.start_time=19/05/2024 02:00:00
--Specify relative start time
--upg1.start_time=+1h30m
```

Refreshable Clone

1

Run on source

```
autoupgrade.jar ... -mode analyze

autoupgrade.jar ... -mode fixups
```

2

Run on target

```
autoupgrade.jar ... -mode deploy
```

Refreshable Clone

1. PDB is created
2. Data files are copied
3. Redo is applied
4. Final refresh
5. Disconnect and convert

```
autoupgrade.jar ... -mode deploy
```

```
upg1.start_time=19/05/2024 02:00:00
```

The source non-CDB stays intact to allow rollback

Demo

Multitenant migration
Upgrade to Oracle Database 19c
Using Refreshable Clone PDBs

Watch on YouTube

Refreshable clone works only with deferred recovery on standby database

- You must restore the PDB on standby database after disconnect from non-CDB

Ensure archive logs are available on disk during migration

Cloning

CLONING

AutoUpgrade uses `CREATE PLUGGABLE DATABASE` statement with `PARALLEL` clause which clones the database using multiple parallel processes

PARALLEL

Based on system resources and current utilization the database automatically determines a proper parallel degree

TRANSFER

A new file transfer protocol that can bypass several layers in the database to achieve very high transfer rates

NETWORK

Watch out for network saturation. Control parallelism in the AutoUpgrade config file

The database applies automatic parallelism based on available system resources

- Control using ***prefix*.parallel_pdb_creation_clause**

```
SQL> select message, sofar, totalwork,time_remaining as remain, elapsed_seconds as ela
     from v$session_longops
     where opname='kpdbfCopyTaskCbk' and sofar != totalwork;
```

```
MESSAGE                                                                 SOFAR       TOTALWORK    REMAIN      ELA
kpdbfCopyTaskCbk: /u01/app/oracle/oradata/CDB2/EDA 3: 643199 out of 1310720 Blocks done          643199      1310720      134         129
kpdbfCopyTaskCbk: /u01/app/oracle/oradata/CDB2/EDA 3: 443007 out of 1310720 Blocks done          443007      1310720      213         109
kpdbfCopyTaskCbk: /u01/app/oracle/oradata/CDB2/EDA 3: 436351 out of 1310720 Blocks done          436351      1310720      216         108
kpdbfCopyTaskCbk: /u01/app/oracle/oradata/CDB2/EDA 3: 370431 out of 1310720 Blocks done          370431      1310720      256         101
```

```
SQL> select message, sofar, totalwork,time_remaining as remain, elapsed_seconds as ela
     from v$session_longops
     where opname='kpdbfCopyTaskCbk' and sofar != totalwork;
```

```
MESSAGE                                                                    SOFAR       TOTALWORK    REMAIN      ELA
kpdbfCopyTaskCbk: /u01/app/oracle/oradata/CDB2/EDA 3: 643199 out of 1310720 Blocks done          643199      1310720      134         129
kpdbfCopyTaskCbk: /u01/app/oracle/oradata/CDB2/EDA 3: 443007 out of 1310720 Blocks done          443007      1310720      213         109
kpdbfCopyTaskCbk: /u01/app/oracle/oradata/CDB2/EDA 3: 436351 out of 1310720 Blocks done          436351      1310720      216         108
kpdbfCopyTaskCbk: /u01/app/oracle/oradata/CDB2/EDA 3: 370431 out of 1310720 Blocks done          370431      1310720      256         101
```

```
SQL> select sql_text
     from v$sql s, v$session_longops l
     where s.sql_id=l.sql_id and l.opname='kpdbfCopyTaskCbk';
```

```
SQL_TEXT
/* SQL Analyze(256,0) */ SELECT /*+PARALLEL(4) NO_STATEMENT_QUEUING  */ * FROM X$KXFTASK /*kpdbfParallelCopyOrMove,PDB_FILE_COPY*/
```

Zürcher
Kantonalbank

Customer Case | Zürcher Kantonalbank

A reliable partner for over 150 years

- The bank for the people of Zurich since 1870
- With over 5'100 employees one of the largest employers in the canton of Zurich
- Globally networked full-service bank with strong regional and local roots

Customer Case | Zürcher Kantonalbank

Current situation

- Oracle databases on old OS and on Oracle Exadata
- 2023:
  - Migrate everything to Exadata until end of 2023
  - Consolidation to Multitenant and to the next long-term support release

Planned solution: AutoUpgrade

Customer Case | Zürcher Kantonalbank

Test setup

- 3 non-CDB databases of different size

Source	**Size / GB**
TEST40 (108)	165
TEST42 (107)	555
TEST41 (106)	18'496

- Exadata X6-2 compute node
- 7 storage cells (2x X6-2L / 3x X7-2L / 2x X8-2L)
- Oracle Database 19.15.0
- No additional options

Customer Case | Zürcher Kantonalbank

Cloning user

```
create user dblinkuser identified by Oracle_4UOracle_4U;
```

Permissions

```
grant CONNECT, RESOURCE, CREATE PLUGGABLE DATABASE,
      SELECT_CATALOG_ROLE to dblinkuser;
grant ALL ON SYS.ENC$ to dblinkuser;
```

Database link

```
create database link TEST42.DOMAIN connect to dblinkuser
identified by oracle_4uoracle_4u using ‘test42.domain’;
```

Customer Case | Zürcher Kantonalbank

Customer

Project

Constraints

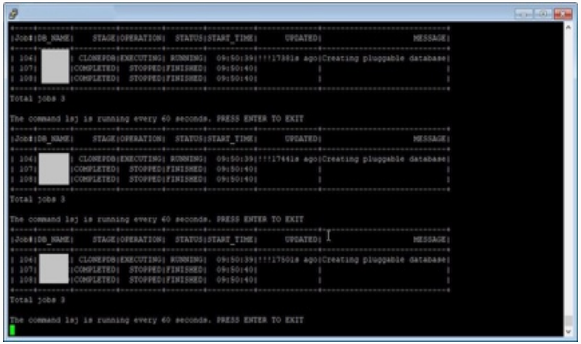
Preparation

Migration

Success?

Remarks

Migration in progress

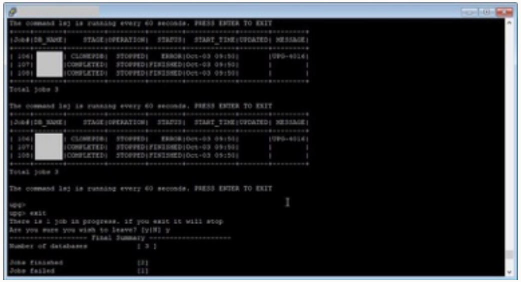


```
+----+-------+---------+---------+--------+----------+-------------+------------------------+
|Job#|DB_NAME|    STAGE|OPERATION|  STATUS|START_TIME|      UPDATED|                 MESSAGE|
+----+-------+---------+---------+--------+----------+-------------+------------------------+
| 106|[redacted]| CLONEPDB|EXECUTING| RUNNING|  09:50:39|!!!17381s ago|Creating pluggable database|
| 107|       |COMPLETED|  STOPPED|FINISHED|  09:50:40|             |                        |
| 108|       |COMPLETED|  STOPPED|FINISHED|  09:50:40|             |                        |
+----+-------+---------+---------+--------+----------+-------------+------------------------+
Total jobs 3

The command lsj is running every 60 seconds. PRESS ENTER TO EXIT
+----+-------+---------+---------+--------+----------+-------------+------------------------+
|Job#|DB_NAME|    STAGE|OPERATION|  STATUS|START_TIME|      UPDATED|                 MESSAGE|
+----+-------+---------+---------+--------+----------+-------------+------------------------+
| 106|[redacted]| CLONEPDB|EXECUTING| RUNNING|  09:50:39|!!!17441s ago|Creating pluggable database|
| 107|       |COMPLETED|  STOPPED|FINISHED|  09:50:40|             |                        |
| 108|       |COMPLETED|  STOPPED|FINISHED|  09:50:40|             |                        |
+----+-------+---------+---------+--------+----------+-------------+------------------------+
Total jobs 3

The command lsj is running every 60 seconds. PRESS ENTER TO EXIT
+----+-------+---------+---------+--------+----------+-------------+------------------------+
|Job#|DB_NAME|    STAGE|OPERATION|  STATUS|START_TIME|      UPDATED|                 MESSAGE|
+----+-------+---------+---------+--------+----------+-------------+------------------------+
| 106|[redacted]| CLONEPDB|EXECUTING| RUNNING|  09:50:39|!!!17501s ago|Creating pluggable database|
| 107|       |COMPLETED|  STOPPED|FINISHED|  09:50:40|             |                        |
| 108|       |COMPLETED|  STOPPED|FINISHED|  09:50:40|             |                        |
+----+-------+---------+---------+--------+----------+-------------+------------------------+
Total jobs 3

The command lsj is running every 60 seconds. PRESS ENTER TO EXIT
```

Source	Runtime/Min
TEST40 (108)	26
TEST42 (107)	ongoing
TEST41 (106)	ongoing

Customer Case | Zürcher Kantonalbank

Migration completed



```
The command lsj is running every 60 seconds. PRESS ENTER TO EXIT
+----+-------+---------+---------+-------+--------------+-------+--------+
|Job#|DB_NAME|    STAGE|OPERATION| STATUS|    START_TIME|UPDATED| MESSAGE|
+----+-------+---------+---------+-------+--------------+-------+--------+
| 106|[redacted]| CLONEPDB|  STOPPED|  ERROR|Oct-03 09:50|       |UPG-4014|
| 107|         |COMPLETED|  STOPPED|FINISHED|Oct-03 09:50|      |        |
| 108|         |COMPLETED|  STOPPED|FINISHED|Oct-03 09:50|      |        |
+----+-------+---------+---------+-------+--------------+-------+--------+
Total jobs 3

The command lsj is running every 60 seconds. PRESS ENTER TO EXIT
+----+-------+---------+---------+-------+--------------+-------+--------+
|Job#|DB_NAME|    STAGE|OPERATION| STATUS|    START_TIME|UPDATED| MESSAGE|
+----+-------+---------+---------+-------+--------------+-------+--------+
| 106|[redacted]| CLONEPDB|  STOPPED|  ERROR|Oct-03 09:50|       |UPG-4014|
| 107|         |COMPLETED|  STOPPED|FINISHED|Oct-03 09:50|      |        |
| 108|         |COMPLETED|  STOPPED|FINISHED|Oct-03 09:50|      |        |
+----+-------+---------+---------+-------+--------------+-------+--------+
Total jobs 3

The command lsj is running every 60 seconds. PRESS ENTER TO EXIT

upg>
upg> exit
There is 1 job in progress. if you exit it will stop
Are you sure you wish to leave? [y|N] y
-------------------- Final Summary --------------------
Number of databases            [ 3 ]

Jobs finished                  [2]
Jobs failed                    [1]
```

Source	Runtime/Min
TEST40 (108)	26
TEST42 (107)	226 (~3.5h)
TEST41 (106)	1770 (29h)

Customer Case | Zürcher Kantonalbank

First non-CDBs migrated successfully

- Project is ongoing

Customer Case | Zürcher Kantonalbank

For large databases, make sure archives aren't cleaned up

- Solution: restore archivelogs from backup

User profile with IDLE_TIME lead to kill of the session

- Solution: assign a different profile to the clone user

Summary

- Very comfortable to use
  - Everything happens automatically
  - Does not require user interaction
- Simple syntax
- No license costs associated
- Perfect for pre-migration test

- Very Stable

MIGRATION

options

Other Options

Refreshable

Plug-in Copy

Plug-in NoCopy

Other Options

It is also possible to migrate using

1 Data Pump

2 Transportable Tablespaces

3 GoldenGate

- Well-known and proven method
- Extremely flexible
- Migrate from lower version
- Migrate from cross-Endian
- Preserves source database for fallback

Other Options

It is also possible to migrate using

1. Data Pump
2. Transportable Tablespaces
3. GoldenGate

- Faster for larger databases
- Migrate from lower version
- Migrate from cross-Endian
- Preserves source database for fallback

Other Options

It is also possible to migrate using

1 Data Pump

2 Transportable Tablespaces

3 GoldenGate

- Only zero downtime option
- Migrate from lower version
- Migrate from cross-Endian
- Preserves source database for fallback
- Active-active replication for ultimate solution

comparing **MIGRATION** options

	Plug-in NoCopy	Plug-in Copy	Refreshable Clone PDB	Data Pump	Transportable	GoldenGate
Downtime	Less	Considerable	Less	Considerable	Minimal	None
Rollback	No	Yes	Yes	Yes	Yes	Yes
Cross-platform (same Endian)	Yes	Yes	Yes	Yes	Yes	Yes
Cross-Endian	No	No	No	Yes	Yes	Yes
Cross-version	No	No	Yes	Yes	Yes	Yes
Complexity	Low	Low	Low	Medium	Medium	High

Best Practices for multitenant migration

Gather dictionary stats before migration

- Preferably also after migration

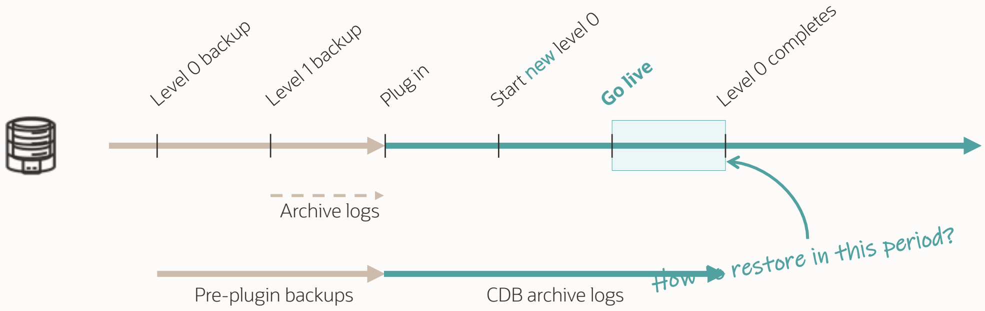
Perform a dictionary check before migration

- Use `DBMS_DICTIONARY_CHECK`

Backup your database after migration

- Level 0

Recovery Strategy



Practice restore with pre-plugin backups

- Check `DBMS_PDB.EXPORTRMANBACKUP`

What about Oracle RAC Database

RAC

- Nothing special, it works seamlessly
- Connect to one instance and plug in
- No need to start with `CLUSTER_DATABASE=FALSE`

What happens during plug-in with Oracle Data Guard

Data Guard

Plug-in on primary propagates to standby database via redo

1 Enabled recovery

2 Deferred recovery

Data Guard

1

Enabled recovery

```
create pluggable database ... standbys=all
```

Standby records PDB creation

Standby locates data files

MRP applies redo to PDB

PDB is immediately protected

Default

2

Deferred recovery

Data Guard

1

Enabled recovery

```
create pluggable database ... standbys=all
```

Standby records PDB creation

Standby locates data files

MRP applies redo to PDB

PDB is immediately protected

2

Deferred recovery

```
create pluggable database ... standbys=none
```

Standby records PDB creation

Standby ignores data files

MRP skips redo

PDB protected after restore

You can specify recovery mode for each standby database

- `CREATE PLUGGABLE DATABASE ... STANDBYS=STDBY1,STDBY3`
- `CREATE PLUGGABLE DATABASE ... STANDBYS=ALL EXCEPT STDBY2`

```
--Check the recovery mode of each PDBs
--Possible values: ENABLED, DISABLED

select name, recovery_status from v$pdbs;
```

In AutoUpgrade, specify recovery mode using `*prefix*.manage_standbys_clause`

- Value is inserted directly into `STANDBYS` clause in `CREATE PLUGGABLE DATABASE` statement

Enabled Recovery

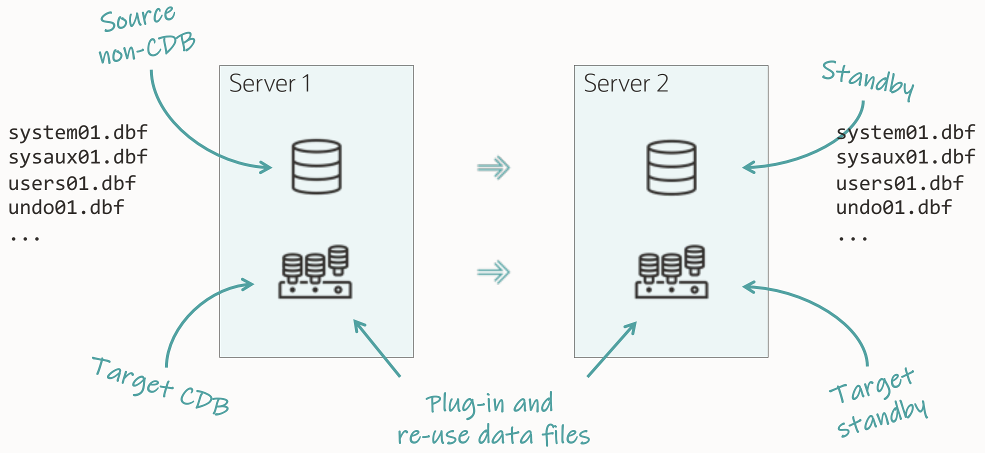
- A more complex approach
- Requires additional work before and during plug-in
- Standby database protects the PDB immediately after plug-in
- When you use `STANDBYS=ALL` or a list
- Default

Enabled Recovery

Enabled recovery applies to:

- All standby databases when using `STANDBYS=ALL`
- Or, those standby databases mention in `STANDBYS=<list>`
- Or, those not mentioned in `STANDBYS=ALL EXCEPT <list>`

Enabled Recovery



All data files on primary and standby must be at the same SCN

Reusing Data Files

Primary

```
SQL> shutdown immediate
SQL> startup mount
SQL> alter system
     flush redo to stdby no confirm apply;
SQL> alter database open read only;
SQL> select checkpoint_change#
     from v$datafile_header where file#=1;
```

MUST MATCH!

```
SQL> exec dbms_pdb.describe('/home/oracle/prmy.xml');
SQL> shutdown immediate
```

Standby

```
DGMGRL> edit database stdby set state='APPLY-OFF';
SQL> shutdown immediate
SQL> startup
SQL> alter database
     recover managed standby database cancel;
```

Replace with checkpoint_change#

```
SQL> alter database recover managed standby database
     until change 2319950;
SQL> select checkpoint_change#
     from v$datafile_header where file#=1;
```

```
SQL> shutdown immediate
```

Enabled Recovery

- The plug-in happens on the primary database
- The plug-in uses the manifest file
- The manifest file contains information on data files from the primary database only

How does the standby database know which files to plug in?

Enabled Recovery

How does the standby database know which files to plug in?

1. Regular files
2. OMF in regular file system
3. ASM

Enabled Recovery

1 Regular files

- Standby search for data files at the same location as the primary
- Override with `DB_FILE_NAME_CONVERT`
- Or, override with `STANDBY_PDB_SOURCE_FILE_DIRECTORY`

Enabled Recovery

2 OMF in regular file system

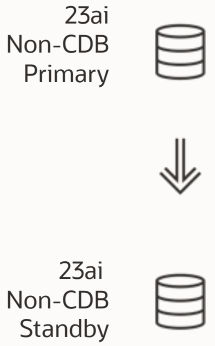
- Standby search for data files at the OMF location (`DB_CREATE_FILE_DEST`)
- Move data files from non-CDB location into OMF location
- Or, create soft links in OMF location pointing to data file location

Enabled Recovery

3 ASM

- Standby search for data files at the OMF location (`DB_CREATE_FILE_DEST`)
- Now, it becomes a little tricky…

Enabled Recovery | ASM

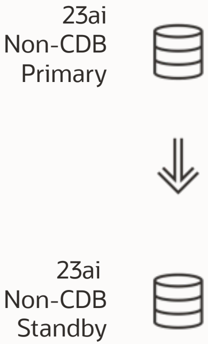


```
SQL> select name from v$datafile;

NAME
------------------------------------------------------------
+DATA/DB_BOSTON/DATAFILE/system.269.1103046537
+DATA/DB_BOSTON/DATAFILE/sysaux.270.1103046537
+DATA/DB_BOSTON/DATAFILE/users.273.1103046827
```

```
SQL> select name from v$datafile;

NAME
------------------------------------------------------------
+DATA/DB_CHICAGO/DATAFILE/system.265.1103050007
+DATA/DB_CHICAGO/DATAFILE/sysaux.266.1103050007
+DATA/DB_CHICAGO/DATAFILE/users.269.1103050009
```

Same file,
but different name

Enabled Recovery | ASM



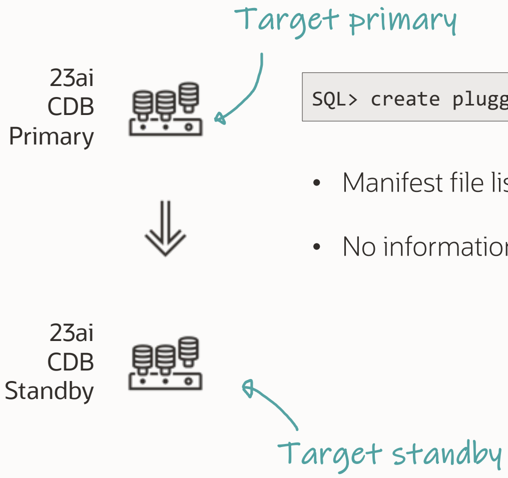
The manifest file contains

- File path on primary database only
- Not standby database

```
SQL> exec dbms_pdb.describe('/tmp/manifest_DB.xml');
```

```
<?xml version="1.0" encoding="UTF-8"?>
<PDB>
  <xmlversion>1</xmlversion>
  <pdbname>PDB1</pdbname>
  ...
  <guid>DDB49CFEFD8ED4FCE053E801000A078C</guid>
  ...
  <tablespace>
    <name>USERS</name>
    ...
    <file>
      <path>+DATA/DB_BOSTON/DATAFILE/users.273.1103046827</path>
```

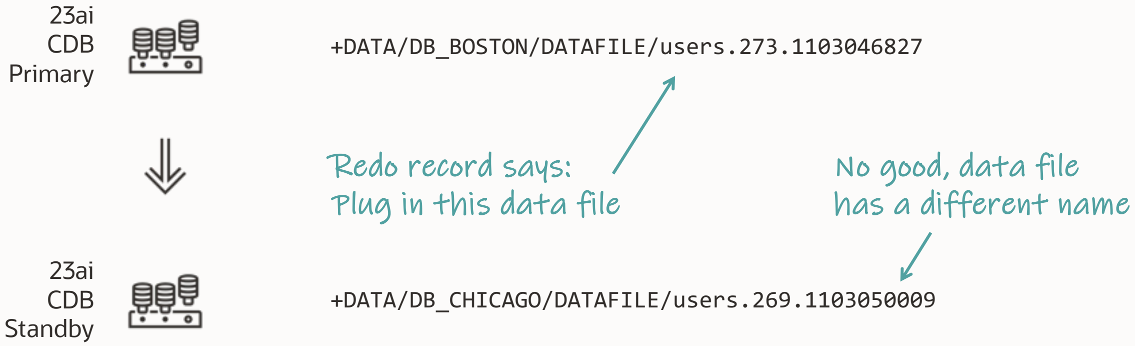
Enabled Recovery | ASM



```
SQL> create pluggable database PDB1 using '/tmp/manifest_DB.xml' ... ;
```

- Manifest file lists the location of data files on primary
- No information about standby databases

Enabled Recovery | ASM



```
+DATA/DB_BOSTON/DATAFILE/users.273.1103046827
```

```
+DATA/DB_CHICAGO/DATAFILE/users.269.1103050009
```

Enabled Recovery | ASM

23ai
CDB
Primary

```
+DATA/DB_BOSTON/DATAFILE/users.273.1103046827
```

OK, let's check the OMF directory

23ai
CDB
Standby

```
+DATA/DB_CHICAGO/DATAFILE/users.269.1103050009
+DATA/CDB1_CHICAGO/<PDB_GUID>/DATAFILE
```

It's empty

Enabled Recovery | ASM

23ai
CDB
Primary

+DATA/DB_BOSTON/DATAFILE/users.273.1103046827

⇓

OK, let's check the OMF directory

23ai
CDB
Standby

+DATA/DB_CHICAGO/DATAFILE/users.269.1103050009

+DATA/CDB1_CHICAGO/<PDB_GUID>/DATAFILE

It's empty

I'll just move the file in ASM

There's no move command in ASM.
How about copying?

```
ASMCMD> cp users.269.1103050009
           +DATA/DB_CHICAGO/.../users.273.1103046827

ASMCMD-8016: copy source '+DATA/DB_BOSTON/.../users.269.1103050009' and target
'+DATA/DB_CHICAGO/.../users.273.1103046827' failed
ORA-15056: additional error message
ORA-15046: ASM file name 'users.273.1103046827' is not in single-file creation form
ORA-06512: at "SYS.X$DBMS_DISKGROUP", line 617
ORA-06512: at line 3 (DBD ERROR: OCIStmtExecute)
```

Only a database can produce files with ASM/OMF data file names

There's no `move` command in ASM. But you can create *aliases*

- Similar to file system soft links

```
SQL> alter diskgroup data add alias
     '+DATA/DB_CHICAGO/DATAFILE/users.269.1103050009'
     for
     '+DATA/CDB1_CHICAGO/<PDB_GUID>/DATAFILE/users.269.1103050009':
```

Non-CDB standby data file

```
SQL> alter diskgroup data add alias
     '+DATA/DB_CHICAGO/DATAFILE/users.269.1103050009'
     for
     '+DATA/CDB1_CHICAGO/<PDB_GUID>/DATAFILE/users.269.1103050009';
```

```
SQL> alter diskgroup data add alias
     '+DATA/DB_CHICAGO/DATAFILE/users.269.1103050009'
     for
     '+DATA/CDB1_CHICAGO/<PDB_GUID>/DATAFILE/users.269.1103050009';
```

↑

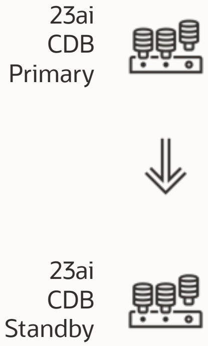
Standby PDB OMF location

```
SQL> alter diskgroup data add alias
     '+DATA/DB_CHICAGO/DATAFILE/users.269.1103050009'
     for
     '+DATA/CDB1_CHICAGO/<PDB_GUID>/DATAFILE/users.269.1103050009';
```

$\uparrow$

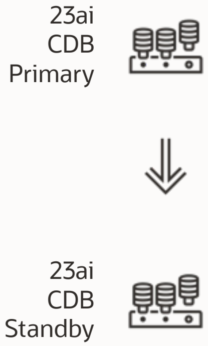
Name does not matter.
Standby scans all files in OMF directory

Data Guard | Re-use Data Files



- Standby database scans its own OMF directory for data files
- Standby ignores file names and look at file headers
- Standby will find aliases and find the real file locations

Data Guard | Re-use Data Files



Looking for file like on primary

```
Recovery scanning directory +DATA/DB_BOSTON/... for any matching files
Deleted Oracle managed file +DATA/DB_BOSTON/...
Successfully added datafile 37 to media recovery
Datafile #37: +DATA/DB_CHICAGO/DATAFILE/users.269.1103050009
```

Follows alias and finds the real file

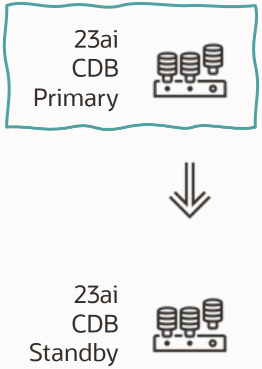
Demo

Multitenant migration
19c non-CDB to 23ai
With Data Guard and re-using data files

Watch on YouTube

Move data files into proper OMF location and get rid of aliases

Data File Location



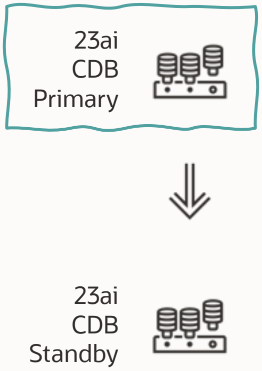
```
SQL> select file#, name from v$datafile;

FILE# NAME
----- ------------------------------------------------
14      +DATA/DB_BOSTON/DATAFILE/system.269.1103046537
15      +DATA/DB_BOSTON/DATAFILE/sysaux.270.1103046537
16      +DATA/DB_BOSTON/DATAFILE/users.273.1103046827
```

Non-CDB OMF location

The database does not care.
But humans might care

- A database can use files from a non-OMF location

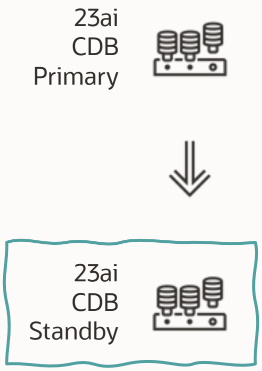
Online Data File Move



```
SQL> ALTER DATABASE MOVE DATAFILE 14;

SQL> select file#, name from v$datafile;

FILE# NAME
----- -----------------------------------------------------------
14      +DATA/CDB_BOSTON/<PDB GUID>/DATAFILE/system.269.1103046537
15      +DATA/DB_BOSTON/DATAFILE/sysaux.270.1103046537
16      +DATA/DB_BOSTON/DATAFILE/users.273.1103046827
```

- The database copies the file, then drops the alias and original file
- Requires additional disk space
- Online operation

Online Data File Move



- On standby, online datafile move works only when CDB and PDB are open
- Stop redo apply before opening, unless you have a license for Active Data Guard
- Requires time and disk space to hold a copy of the data file
- Removes ASM alias and original file upon completion

```
DGMGRL> edit database <STANDBY_UNQNAME> set state='apply-off';
DGMGRL> edit database <STANDBY_UNQNAME> set state='apply-off';

SQL> alter database open read only;
SQL> alter pluggable database PDB1 open read only;
SQL> alter session set container=PDB1;
SQL> alter database move datafile <file#>;
SQL> alter database move datafile <file#>;
SQL> ...
SQL> alter database move datafile <file#>;
```

```
DGMGRL> edit database <STANDBY_UNQNAME> set state='apply-off';

SQL> alter database open read only;
SQL> alter pluggable database PDB1 open read only;
SQL> alter session set container=PDB1;
SQL> alter database move datafile <file#>;
SQL> alter database move datafile <file#>;
SQL> ...
SQL> alter database move datafile <file#>;

srvctl stop database -d $ORACLE_UNQNAME -o immediate
srvctl start database -d $ORACLE_UNQNAME -o mount
```

```
DGMGRL> edit database <STANDBY_UNQNAME> set state='apply-off';

SQL> alter database open read only;
SQL> alter pluggable database PDB1 open read only;
SQL> alter session set container=PDB1;
SQL> alter database move datafile <file#>;
SQL> alter database move datafile <file#>;
SQL> ...
SQL> alter database move datafile <file#>;

srvctl stop database -d $ORACLE_UNQNAME -o immediate
srvctl start database -d $ORACLE_UNQNAME -o mount

DGMGRL> edit database <STANDBY_UNQNAME> set state='apply-on';
DGMGRL> edit database <STANDBY_UNQNAME> set state='apply-off';

SQL> alter database open read only;
SQL> alter pluggable database PDB1 open read only;
SQL> alter session set container=PDB1;
SQL> alter database move datafile <file#>;
SQL> alter database move datafile <file#>;
SQL> ...
SQL> alter database move datafile <file#>;

srvctl stop database -d $ORACLE_UNQNAME -o immediate
srvctl start database -d $ORACLE_UNQNAME -o mount

DGMGRL> edit database <STANDBY_UNQNAME> set state='apply-on';
```

Just move data files,
if Active Data Guard

While relocating data files, apply lag increases if redo apply is off

- Redo transport is still active
- Switchover/failover time increases

What happens with enabled recovery if the standby fails to find the data files?

Enabled Recovery | Missing Data Files

What if a standby database fails to find data files?

- If Active Data Guard and PDB Recovery Isolation is turned on
  - New feature in Oracle Database 21c
  - Recovery disabled for PDB
  - Recovery proceeds in the entire CDB, except in specific PDB
  - Standby automatically restores data files from primary and re-enables recovery afterward
  - PDB protected after auto-restore
- If not, recovery halts in the **entire** CDB
  - **This is a critical situation**

What about AutoUpgrade and enabled recovery?

Enabled Recovery | AutoUpgrade

The current version (24.1) does not support plugging in with enabled recovery

- Enabled recovery requires work on both primary and standby hosts
- You must execute commands at specific times
- It's complicated - but we're working on it

What about **`AS CLONE`** clause

- `CREATE PLUGGABLE DATABASE ... AS CLONE`

Enabled Recovery | As Clone

- When you plug in a non-CDB the GUID doesn't change
  - Get GUID from the manifest file or `V$CONTAINERS`
  - GUID only changes when you use `AS CLONE` keyword

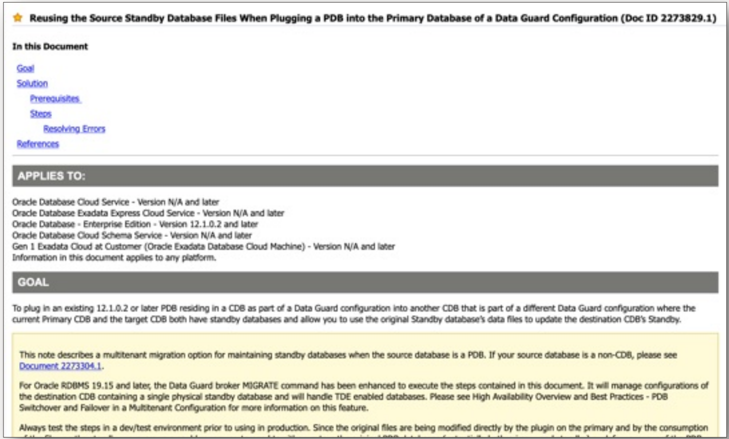
- Impossible to re-use standby data files when
  - Using OMF and `CREATE PLUGGABLE DATABASE ... AS CLONE`
  - Regardless of whether you use regular file system or ASM
  - Only works with regular files and non-OMF

Enabled Recovery | As Clone

- Regular file system and non-OMF
  - Put data files at the same location as primary data files
  - Take `FILE_NAME_CONVERT` into account (`CREATE PLUGGABLE DATABASE`)
  - Adjust according to `DB_FILE_NAME_CONVERT`

Data Guard | Enabled Recovery

Reusing the Source Standby Database Files When Plugging a PDB into the Primary Database of a Data Guard Configuration (Doc ID 2273829.1)



Reusing the Source Standby Database Files When Plugging a PDB into the Primary Database of a Data Guard Configuration (Doc ID 2273829.1)

In this Document

APPLIES TO:

Oracle Database Cloud Service - Version N/A and later
Oracle Database Exadata Express Cloud Service - Version N/A and later
Oracle Database - Enterprise Edition - Version 12.1.0.2 and later
Oracle Database Cloud Schema Service - Version N/A and later
Gen 1 Exadata Cloud at Customer (Oracle Exadata Database Cloud Machine) - Version N/A and later
Information in this document applies to any platform.

GOAL

To plug in an existing 12.1.0.2 or later PDB residing in a CDB as part of a Data Guard configuration into another CDB that is part of a different Data Guard configuration where the current Primary CDB and the target CDB both have standby databases and allow you to use the original Standby database's data files to update the destination CDB's Standby.

This note describes a multitenant migration option for maintaining standby databases when the source database is a PDB. If your source database is a non-CDB, please see Document 2273304.1.

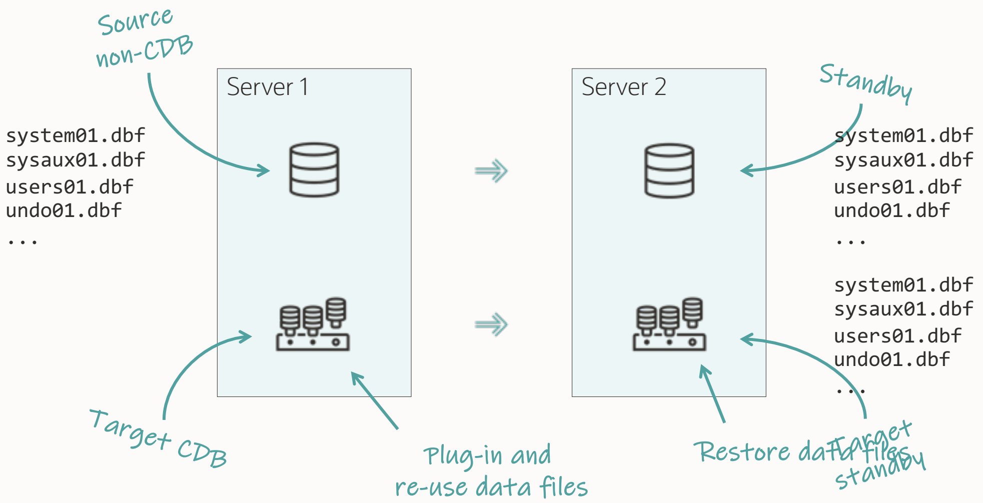
For Oracle RDBMS 19.15 and later, the Data Guard broker MIGRATE command has been enhanced to execute the steps contained in this document. It will manage configurations of the destination CDB containing a single physical standby database and will handle TDE enabled databases. Please see High Availability Overview and Best Practices - PDB Switchover and Failover in a Multitenant Configuration for more information on this feature.

Always test the steps in a dev/test environment prior to using in production. Since the original files are being modified directly by the plugin on the primary and by the consumption

Deferred Recovery

- The simplest approach
- Requires additional work after plug-in
- You must restore the PDB and re-enabledrecovery
- Standby database protects the PDB after restore
- When you use `STANDBYS=NONE`

Deferred Recovery

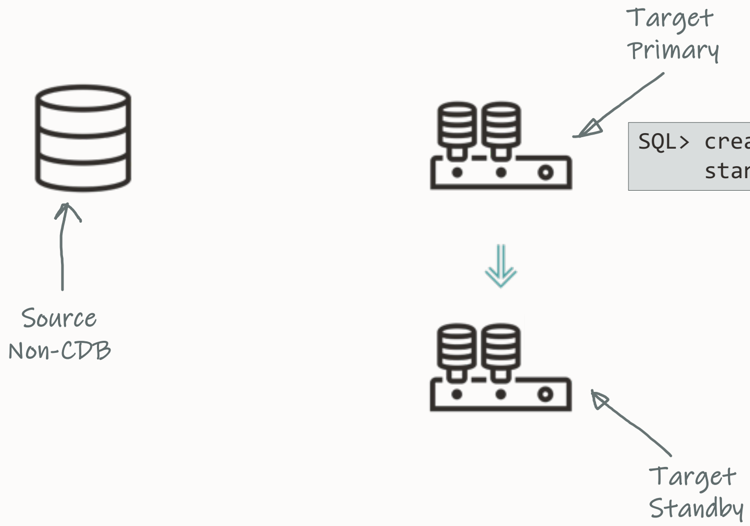


Deferred Recovery

Deferred recovery applies to:

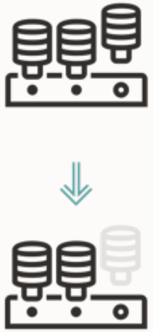
- All standby databases when using `STANDBYS=NONE`
- Or, those standby databases mention in `STANDBYS=ALL EXCEPT <list>`
- Or, those not mentioned in `STANDBYS=<list>`

Deferred Recovery



```
SQL> create pluggable database ...
     standbys=none;
```

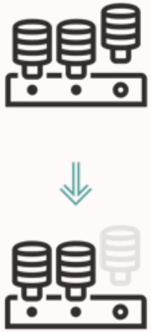
Deferred Recovery



```
SQL> create pluggable database ...
     standbys=none;
```

PDB created
Data files missing

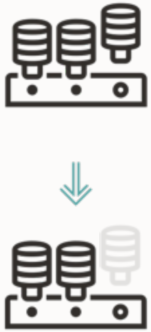
Deferred Recovery



```
SQL> show pdbs

CON_NAME    OPEN MODE
PDB1        READ WRITE
```

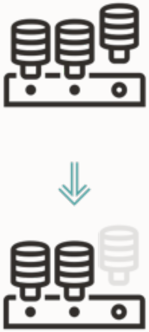
```
SQL> show pdbs

CON_NAME    OPEN MODE
PDB1        MOUNTED
```

Deferred Recovery



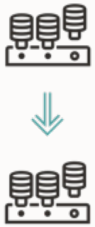
```
SQL> select name, recovery_status
     from v$pdbs;

NAME    RECOVERY_STATUS
PDB1    DISABLED
```

Deferred Recovery



```
RMAN> restore pluggable database
      ... from service ... ;

SQL> alter pluggable database
     enable recovery;
SQL> alter database datafile
     ... online;
```

Deferred Recovery

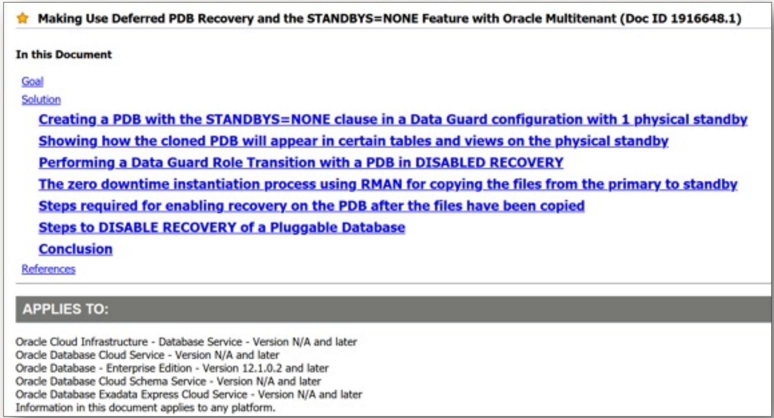


```
RMAN> restore pluggable database
      ... from service ... ;

SQL> alter pluggable database
     enable recovery;
SQL> alter database datafile
     ... online;
```

- Automated process in Oracle Database 21c
- PDB Recovery Isolation
- Requires Active Data Guard

Data Guard | Deferred Recovery

Making Use Deferred PDB Recovery and the STANDBYS=NONE Feature with Oracle Multitenant (Doc ID 1916648.1)



Making Use Deferred PDB Recovery and the STANDBYS=NONE Feature with Oracle Multitenant (Doc ID 1916648.1)

In this Document

APPLIES TO:

Oracle Cloud Infrastructure - Database Service - Version N/A and later
Oracle Database Cloud Service - Version N/A and later
Oracle Database - Enterprise Edition - Version 12.1.0.2 and later
Oracle Database Cloud Schema Service - Version N/A and later
Oracle Database Exadata Express Cloud Service - Version N/A and later
Information in this document applies to any platform.

Data Guard | Additional Information

Data Guard Impact on Oracle Multitenant Environments (Doc ID 2049127.1)

The physical standby database and redo apply will normally expect a new PDB's datafiles to have been pre-copied to the standby site and be in such a state that redo received from the primary database can be immediately applied. The standby database ignores any file name conversion specification on the CREATE PLUGGABLE DATABASE statement and relies solely on the standby database's initialization parameter settings for DB_CREATE_FILE_DEST and DB_FILE_NAME_CONVERT for locations and file naming.

For these cases, Oracle recommends deferring recovery of the PDB using the STANDBYS=NONE clause on the CREATE PLUGGABLE DATABASE statement. Recovery of the PDB can be enabled at some point in the future once the PDB's data files have been copied from the primary database to the standby database in a manner similar to that documented in Document 1916648.1.

Don't jeopardize your Data Guard

- Test the procedure and verify before go-live

How a customer handled the migration

- Customer case

Customer Case

Customer → Project → Result → Learnings

- **Swisscom** - Switzerland's leading telco
- One of the leading IT companies in Switzerland
- One of Switzerland’s most sustainable and innovative companies

Customer Case

Customer Case

Customer | **Project** | Result | Learnings

Oracle Siebel CRM

Non-CDB

Single-tenant architecture

- Database size: 18 TB
- Release: 19.17.0
- Average active users: 3000

Customer Case

Customer | **Project** | Result | Learnings

Oracle Siebel CRM

Non-CDB

- 6,000 tables
- 29,000 indexes
- Partitioning
- LOBs
- 51 bigfile tablespaces

Single-tenant architecture

Customer Case

Data Guard with five standby databases

Each database is a 4-node RAC database

Running on Exadata Database Machine

Streaming data to microservices using GoldenGate

After consulting the business,
a plan was made for the standby databases

Customer Case

Standby 1: DR	Re-use data files, ASM alias
Standby 2-3: Auxiliary DR	Restore data files
Standby 4-5: Reporting	Restore data files

Customer Case

Customer | Project | **Result** | Learnings

- Total downtime was 3 hours 30 minutes
  - Planned maintenance window was 4 hours
  - Most time spent on application work and GoldenGate configuration

- Database migration
  - `noncdb_to_pdb.sql`: 18 minutes

Customer Case

Customer → Project → Result → **Learnings**

1. Test, test, and test
2. Create a detailed runbook
3. Remove complexity from the project to the extent possible
4. Work together and double-check all actions

Converting on Exadata Database Service and Exadata Cloud@Customer

Exadata Database Service

1. Use tooling to create a new CDB - or use an existing one
2. Plug in and convert using a method of choice
3. Tooling automatically adapts the new PDB after a while

Converting Oracle E-Business Suite

E-Business Suite

- Oracle E-Business Suite Release 12.2 and 12.1.3 support multitenant architecture

- But only in single-tenant architecture

- Useful MOS notes
  - FAQ: Oracle E-Business Suite and the Oracle Multitenant Architecture (Doc ID 2567105.1)
  - Interoperability Notes: Oracle E-Business Suite Release 12.2 with Oracle Database 19c (Doc ID 2552181.1)
  - Getting Started with Oracle E-Business Suite on Oracle Cloud Infrastructure (Doc ID 2517025.1)
  - Using Oracle 19c Oracle RAC Multitenant (Single PDB) with Oracle E-Business Suite Release 12.2 (Doc ID 2530665.1)

Plugging in copies of the same database

As Clone

Each PDB has a unique GUID
- Check `V$CONTAINERS`

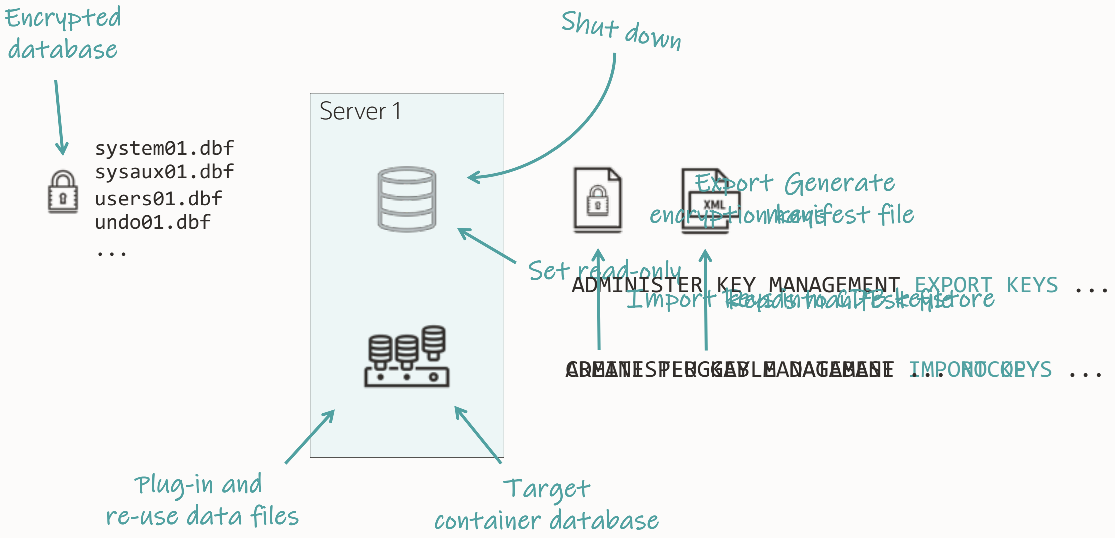
If you plug in the same database multiple times, there are conflicting GUIDs

Use `CREATE PLUGGABLE DATABASE ... AS CLONE` to generate new GUIDs on plug-in

Migrating encrypted databases

- TDE Tablespace Encryption

Encrypted Database



```
ADMINISTER KEY MANAGEMENT EXPORT KEYS ...
ADMINISTER KEY MANAGEMENT IMPORT KEYS ...
CREATE PLUGGABLE DATABASE ... NOCOPY ...
```

AutoUpgrade fully supports encrypted databases

Encrypted Database

Certain database operations require passwords or secrets

```
CREATE PLUGGABLE DATABASE ... KEYSTORE IDENTIFIED BY <password>

ALTER PLUGGABLE DATABASE ... UNPLUG INTO ... ENCRYPT USING <secret>

CREATE PLUGGABLE DATABASE ... DECRYPT USING <secret>

ADMINISTER KEY MANAGEMENT ... KEYSTORE IDENTIFIED BY <password>
```

Secure External Password Store

Operator stores database keystore password in a Secure External Password Store

AutoUpgrade Keystore

Operator loads database keystore password into AutoUpgrade keystore ahead of upgrade

```
# Configure AutoUpgrade to work on encrypted databases
# Specify path for AutoUpgrade keystore

global.keystore=/etc/oracle/keystores/autoupgrade/DB12
global.autoupg_log_dir=/u01/app/oracle/cfgtoollogs/autoupgrade
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=DB12
```

Encrypted Database

Analyze the database for upgrade readiness

```
$ java -jar autoupgrade.jar -config PDB1.cfg -mode analyze
```

Summary report will show which keystore passwords are needed

```
REQUIRED ACTIONS
================
      1.  Perform the specified action ...
      ORACLE_SID                    Action Required
      ------------------------------  ------------------------
      CDB1                          Add TDE password
      CDB2                          Add TDE password
```

Demo

Multitenant migration
Including upgrade to Oracle Database 19c
Using AutoUpgrade

Watch on YouTube

In the unlikely event of ...

- Rollback and fallback options

PDB conversion is irreversible

- Not even Flashback Database can help

Rollback Options | Before Go-Live

1 Leave a copy

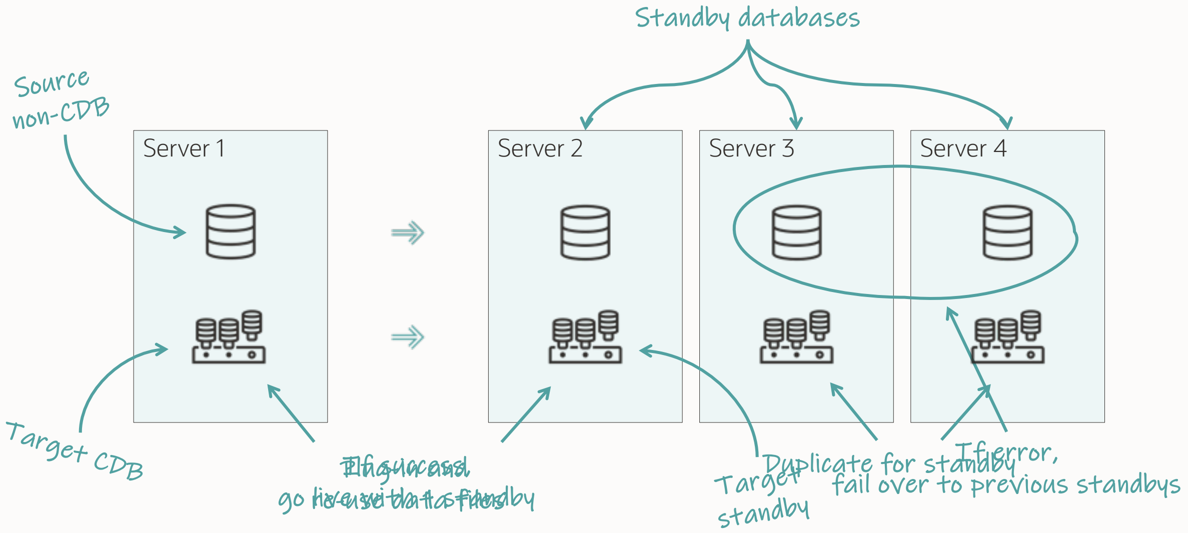
2 RMAN Restore

Rollback Options | Before Go-Live

1 Leave a copy

- `CREATE PLUGGABLE DATABASE ... COPY`
- Refreshable clone PDBs
- Incrementally roll forward data files copies
- Leave a standby database behind

Rollback Options | Before Go-Live



Rollback Options | Before Go-Live

2 RMAN Restore

- Time-consuming
- May not satisfy business requirements

Fallback Options | After Go-Live

1 Back to non-CDB

- Data Pump
- Transportable Tablespaces
- GoldenGate

An alternative option to fall back from upgrade and PDB conversion

Fallback Options | After Go-Live

If you upgraded and converted

- From Oracle Database 19c to 23ai

1 Back to 19c, stay multitenant

- Downgrade
- `COMPATIBLE` must be *19.0.0* in 23ai CDB

2 Back to 19c, back to non-CDB

- Follow 1
- Transportable tablespace back into non-CDB
- Alternatively, Data Pump from 23ai directly to 19c non-CDB

Wrapping Up

Words of Advice

1 Start with simple databases

2 Leave time to learn and adjust

3 Proceed with bigger databases

Further Reading

Oracle Support:

- Oracle Multitenant: Frequently Asked Questions (Doc ID 1511619.1)
- How to migrate a non pluggable database that uses TDE to pluggable database ? (Doc ID 1678525.1)

Blog posts:

- Database Migration from non-CDB to PDB – Typical Plugin Issues and Workarounds
- Upgrade & Plug In: With ASM, Data Guard, TDE and no Keystore Password

There are many benefits to explore

- Application containers
- Faster cloning
- Faster provisioning
- Faster redeployment
- Sparse clones
- Resource consolidation
- Save resources
- Improved functionality
- Better management
- More secure
- Separation of duties
- Easier operations
- Many-as-one
- Faster updates
- Faster patching
- Enables self-service

It's better to fail in our lab, than in production

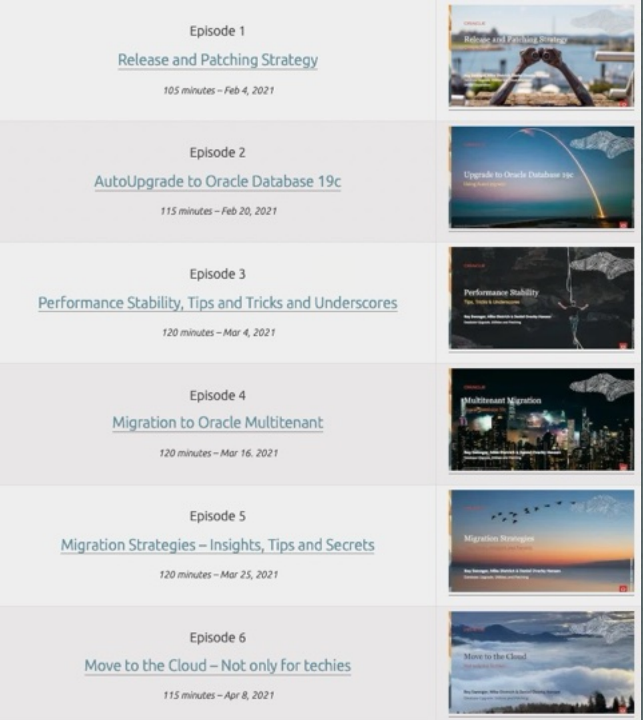
Try multitenant migration in our Hands-On Lab

For free using Oracle LiveLabs

Part 2

Move to Oracle Database 23ai
- Everything you need to know about Oracle Multitenant

Live on June 27, 14:00 CEST
Sign up



Episode 1

Release and Patching Strategy

105 minutes – Feb 4, 2021

Episode 2

AutoUpgrade to Oracle Database 19c

115 minutes – Feb 20, 2021

Episode 3

Performance Stability, Tips and Tricks and Underscores

120 minutes – Mar 4, 2021

Episode 4

Migration to Oracle Multitenant

120 minutes – Mar 16. 2021

Episode 5

Migration Strategies – Insights, Tips and Secrets

120 minutes – Mar 25, 2021

Episode 6

Move to the Cloud – Not only for techies

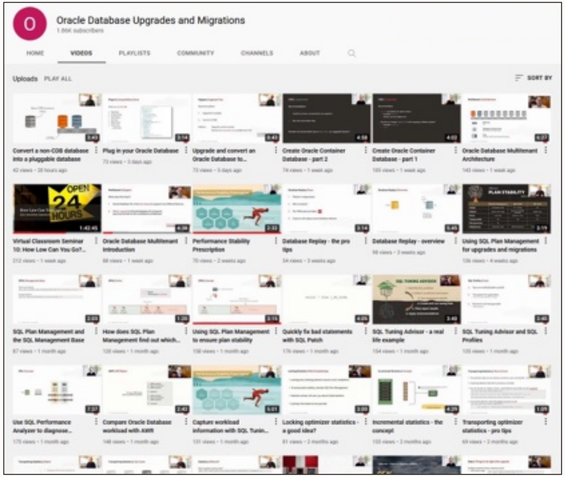
115 minutes – Apr 8, 2021

Recorded Web Seminars

https://MikeDietrichDE.com/videos

More than 35 hours of technical content, on-demand, anytime, anywhere

YouTube | @UpgradeNow



Oracle Database Upgrades and Migrations
1.86K subscribers

Convert a non-CDB database into a pluggable database
42 views · 20 hours ago

Plug in your Oracle Database
73 views · 3 days ago

Upgrade and convert an Oracle Database to...
73 views · 5 days ago

Create Oracle Container Database - part 2
74 views · 1 week ago

Create Oracle Container Database - part 1
105 views · 1 week ago

Oracle Database Multitenant Architecture
143 views · 1 week ago

Virtual Classroom Seminar 10: How Low Can You Go?...
212 views · 1 week ago

Oracle Database Multitenant Introduction
88 views · 1 week ago

Performance Stability Prescription
70 views · 2 weeks ago

Database Replay - the pro tips
54 views · 3 weeks ago

Database Replay - overview
98 views · 3 weeks ago

Using SQL Plan Management for upgrades and migrations
106 views · 4 weeks ago

SQL Plan Management and the SQL Management Base
87 views · 1 month ago

How does SQL Plan Management find out which...
120 views · 1 month ago

Using SQL Plan Management to ensure plan stability
158 views · 1 month ago

Quickly fix bad statements with SQL Patch
176 views · 1 month ago

SQL Tuning Advisor - a real life example
154 views · 1 month ago

SQL Tuning Advisor and SQL Profiles
135 views · 1 month ago

Use SQL Performance Analyzer to diagnose...
175 views · 1 month ago

Compare Oracle Database workload with AWR
148 views · 1 month ago

Capture workload information with SQL Tunin...
131 views · 1 month ago

Locking optimizer statistics - a good idea?
41 views · 2 months ago

Incremental statistics - the concept
155 views · 2 months ago

Transporting optimizer statistics - pro tips
69 views · 2 months ago

Link

- 300+ videos
- New videos every week
- No marketing
- No buzzwords
- All tech

SCAN ME

Thank You