

The Oracle logo is displayed in a bold, red, sans-serif font. It is positioned in the upper left quadrant of the slide, partially overlapping a decorative graphic element consisting of a red and black patterned shape.

Introduction to Multitenant Architecture


Oracle Database 23c




Daniel Overby Hansen

Senior Principal Product Manager



 dohdatabase

 @dohdatabase

 <https://dohdatabase.com>

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 30 hours of technical content,
on-demand, anytime, anywhere





400+ technical experts helping peers globally

The **Oracle ACE Program** recognizes and rewards community members for their technical and community contributions to the Oracle community



3 membership tiers



For more details on Oracle ACE Program:
ace.oracle.com



Nominate

yourself or someone you know:

ace.oracle.com/nominate

Connect:  aceprogram_ww@oracle.com

 Facebook.com/OracleACEs

 [@oracleace](https://twitter.com/oracleace)





Architecture

Multitenant Evolution

2013

12^c

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

2018

18^c

- Snapshot Carousel
- Refreshable PDB Switchover
- CDB Fleet

2016

12^c

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

2019

19^c

- 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, 23c

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



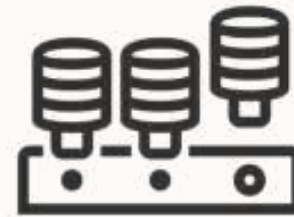
Generally, you don't need to change your application to use a pluggable database

Single vs. Multitenant



Single Tenant

One PDB
No extra license



Multitenant

Multiple PDBs
Extra license if more than 3 PDBs

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

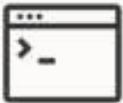


When you create a new CDB, it contains:

- 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

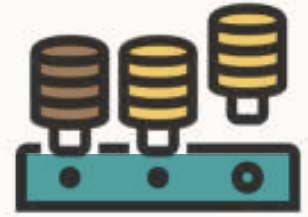
*CDB\$ROOT
is always 1*

*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
```

```
-----
```

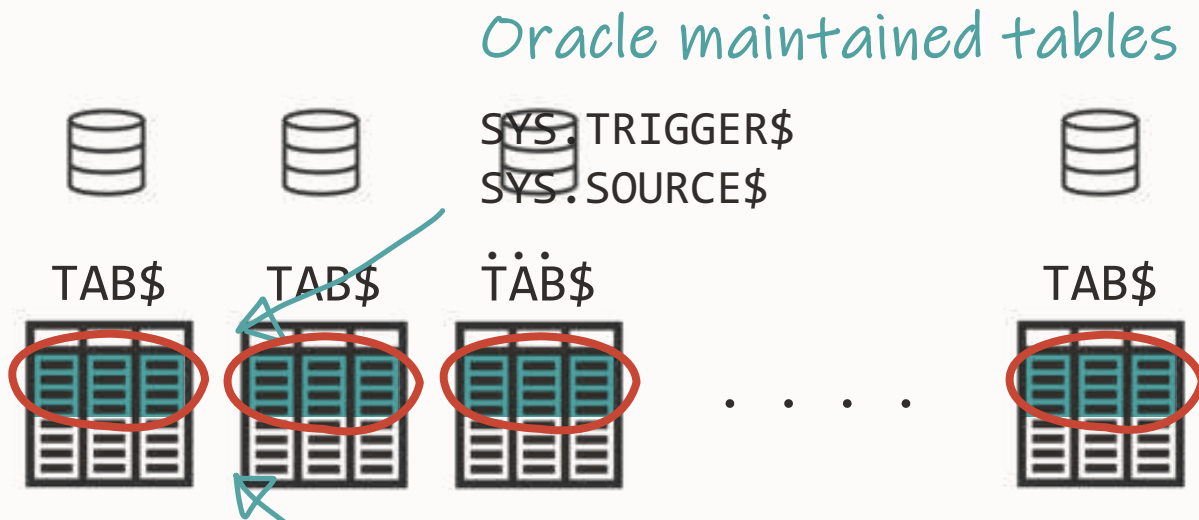
```
3
```




Multitenant implements data dictionary separation

Data Dictionary Architecture

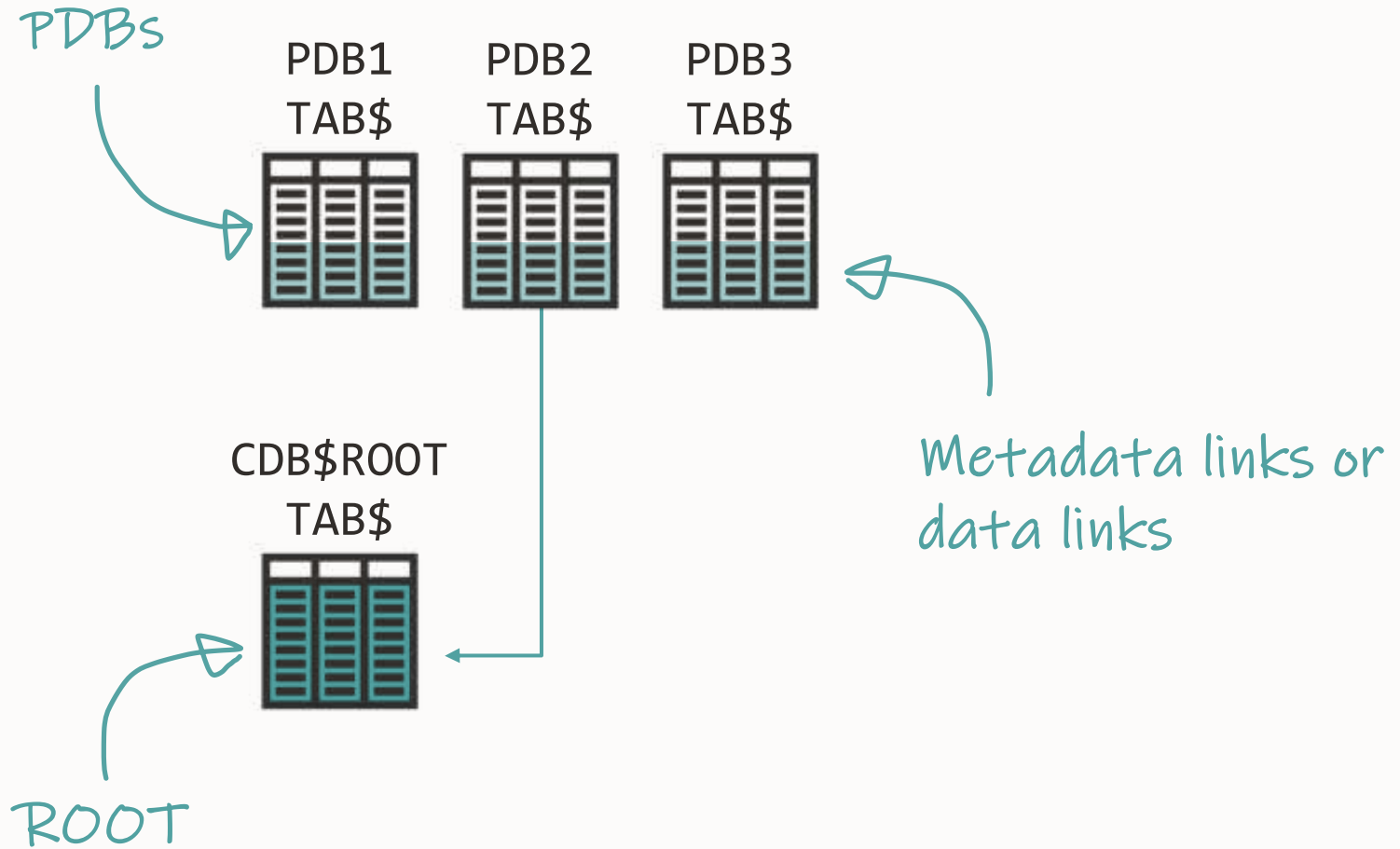
Non-CDB



Redundant data

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

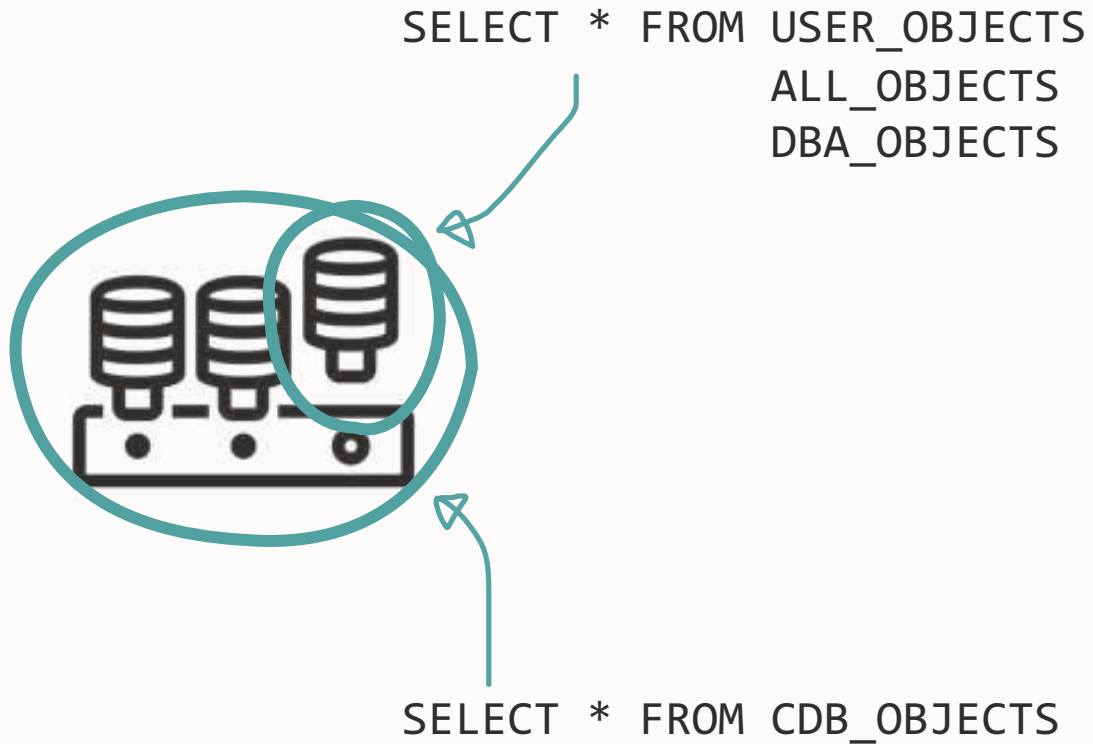
With much metadata stored in root, there is less work for a patch or upgrade.



CDB views describe the entire CDB including all PDBs

- Column `CON_ID` indicates the originating container

Data Dictionary Architecture



*Applies to any
data dictionary view*

CDB_ALL_TABLES
CDB_ANALYTIC_VIEW_ATTR_CLASS
.
.
.
CDB_XTERNAL_TAB_SUBPARTITIONS

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

```
select con_id, tablespace_name  
from cdb_tablespaces;
```

Originating
container

↓

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 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

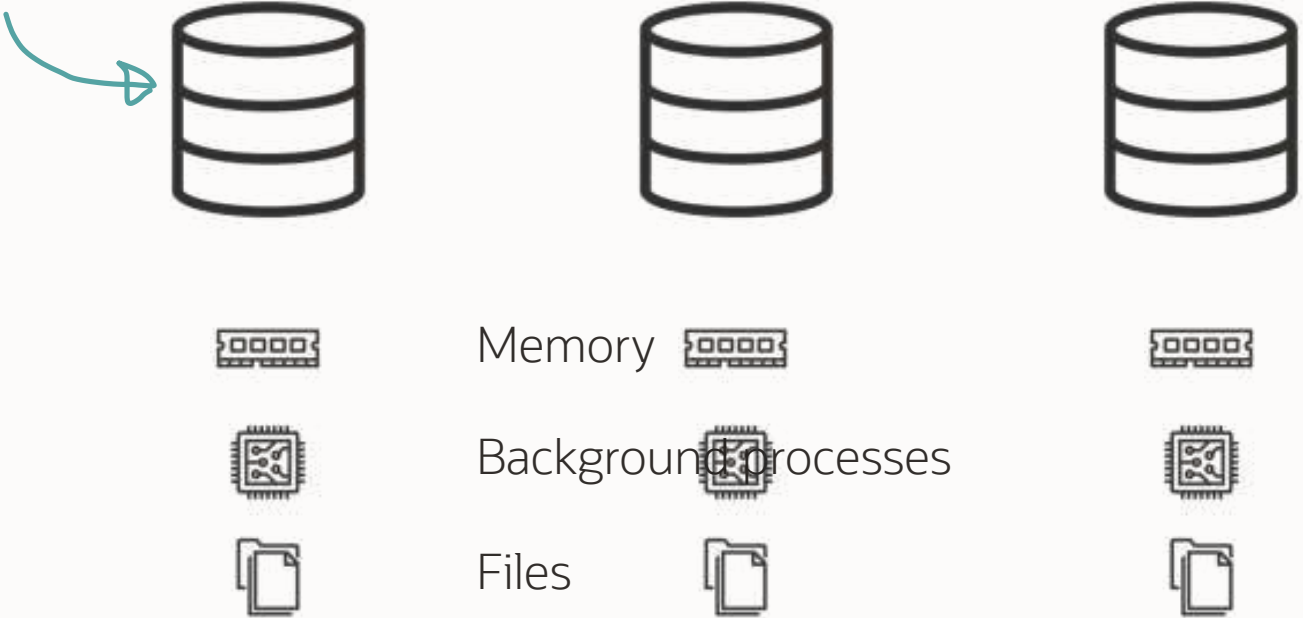
```
select name, value from v$system_parameter where con_id=<id>;
```



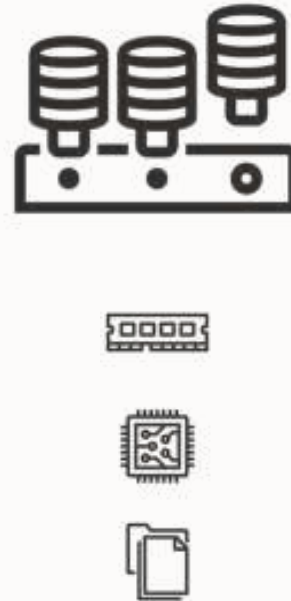
Share resources between PDBs

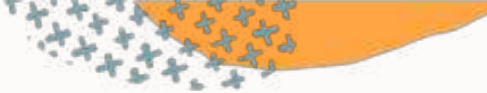
Resource Consolidation

Non-CDB
database

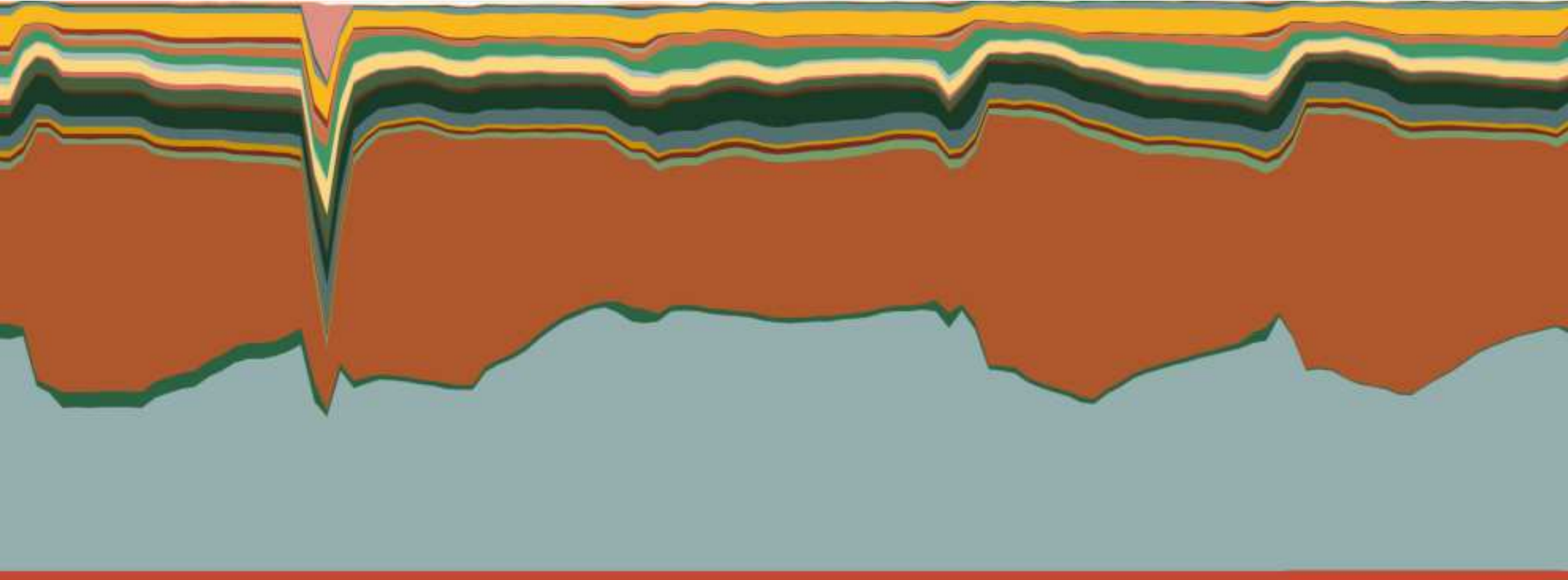


Resource Consolidation





Resource Consolidation



A US Health Care provider managed to

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

Consolidation Strategies?

There is no "*best*" strategy



Don't mix PDBs with different SLAs

Total memory consumption minus 20%-30%

Increase consolidation factor slowly



Be open to start with a completely new naming schema



Avoid noisy neighbors

- Allow sharing resources
but everyone must get a fair share

Step 1



Instance caging

- Define `CPU_COUNT` for each PDB
- Hard limit



Step 2



Memory allocation

- Use Automatic Shared Memory Management
- Set reasonable value for `SGA_TARGET`
- Optionally, also for memory pools
 - Shared pool (`SHARED_POOL_SIZE`)
 - Buffer cache (`DB_CACHE_SIZE`)

Step 3



Simple Resource Manager

- Enable CDB resource manager
- Set `RESOURCE_MANAGER_PLAN` at root level
- Allocate minimum shares
 - `CPU_MIN_COUNT`
 - `SGA_MIN_SIZE`



Step 4



Advanced Resource Manager

- Use directives instead of shares
- Exadata I/O Resource Management



You can still control resources inside a PDB
with Resource Manager





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

- Shares resources like with non-CDBs
- `CPU_COUNT` and `SGA_MAX_SIZE`

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*



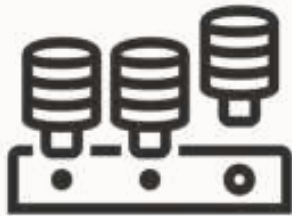
Many-as-one



Patch all containers
in one operation



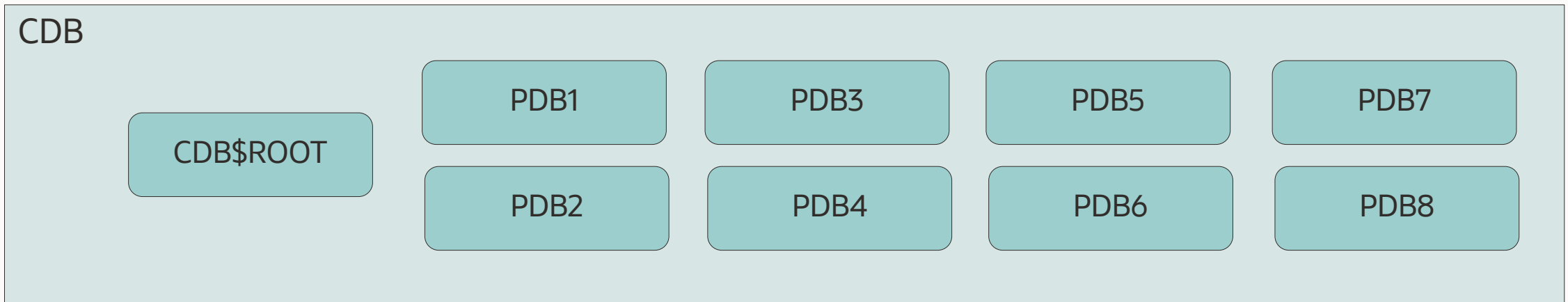
Many-as-one



Applies to:

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

Many-as-one



- Datapatch patches CDB\$ROOT and PDB\$SEED automatically
- Datapatch only patches open PDBs
- Datapatch determines parallel degree based on CPU count

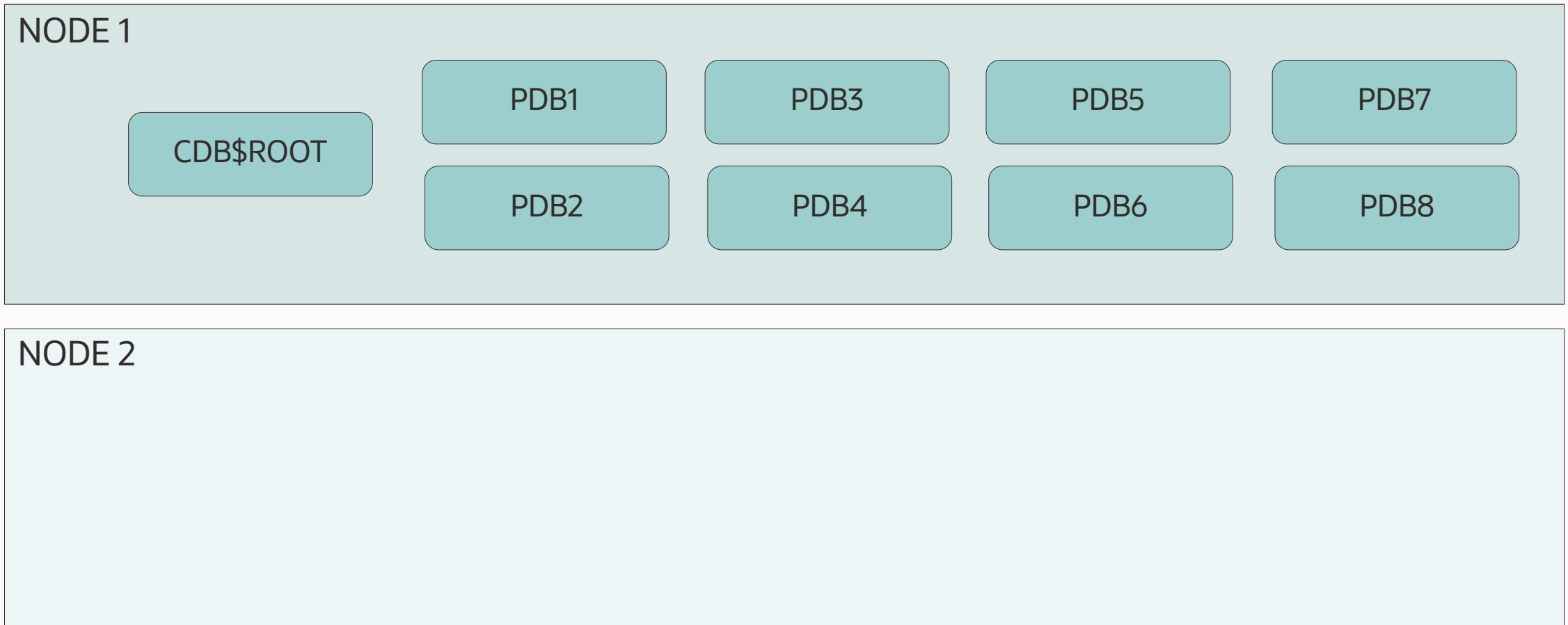




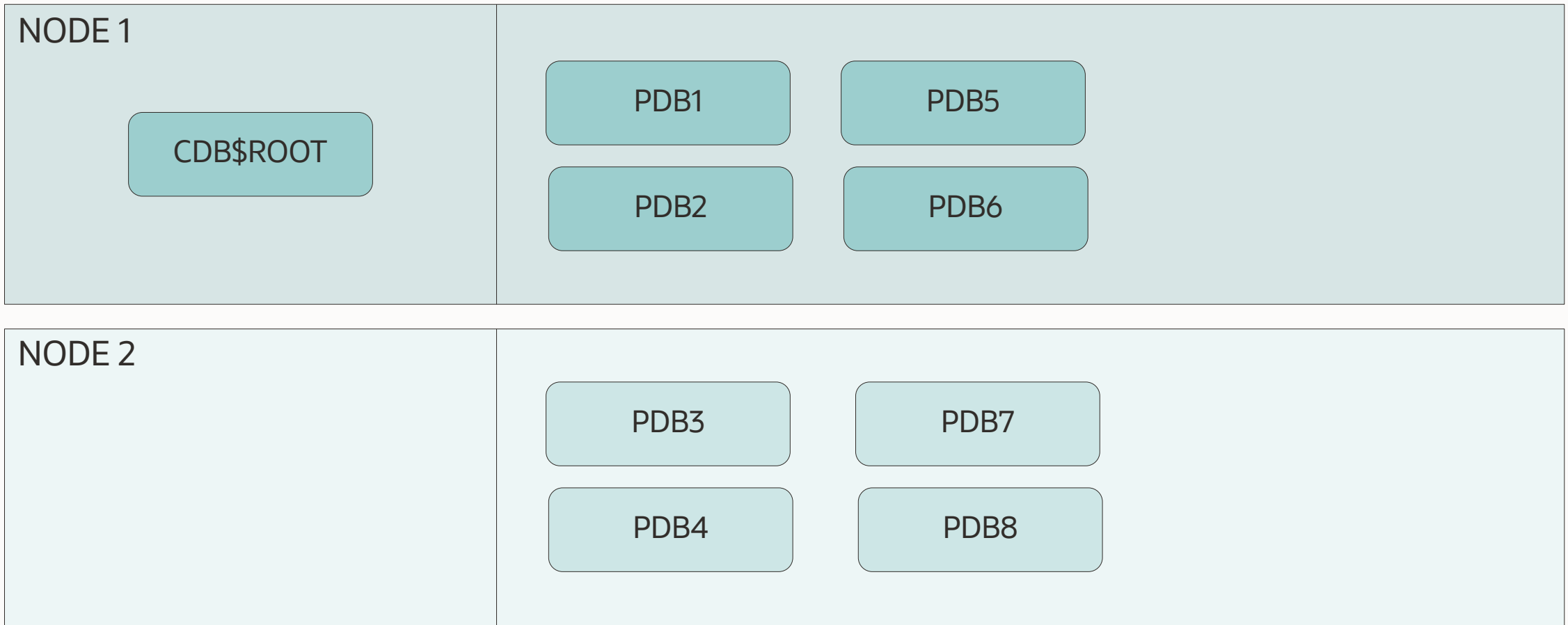
Significantly speed up patching using AutoUpgrade

- Applies to multitenant databases on RAC only

Distributed Patching



Distributed Patching



Distributed Patching

To enable distributed patching

```
$ cat RACCDB.cfg  
  
upg1.source_home=/u01/app/oracle/product/19/dbhome_19_18  
upg1.target_home=/u01/app/oracle/product/19/dbhome_19_19  
upg1.sid=RACCDB  
upg1.tune_setting=proactive_fixups=true,distributed_upgrade=true  
  
$ java -jar autoupgrade.jar -config RACCDB.cfg -mode deploy
```

41%

—
Time saved by using distributed patching





Benefits



The **multitenant** architecture enables

- 1** Self-contained PDBs
- 2** Common and easier operations
- 3** Resource sharing and consolidation





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

Migration



1 Plug in

Then, perform plug-in

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

Migration



2

Convert

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



Convert-on-open automatically converts a non-CDB when plugged in

- Oracle Database 23c
- Also *upgrade-on-open*

MIGRATION

options

Other Options

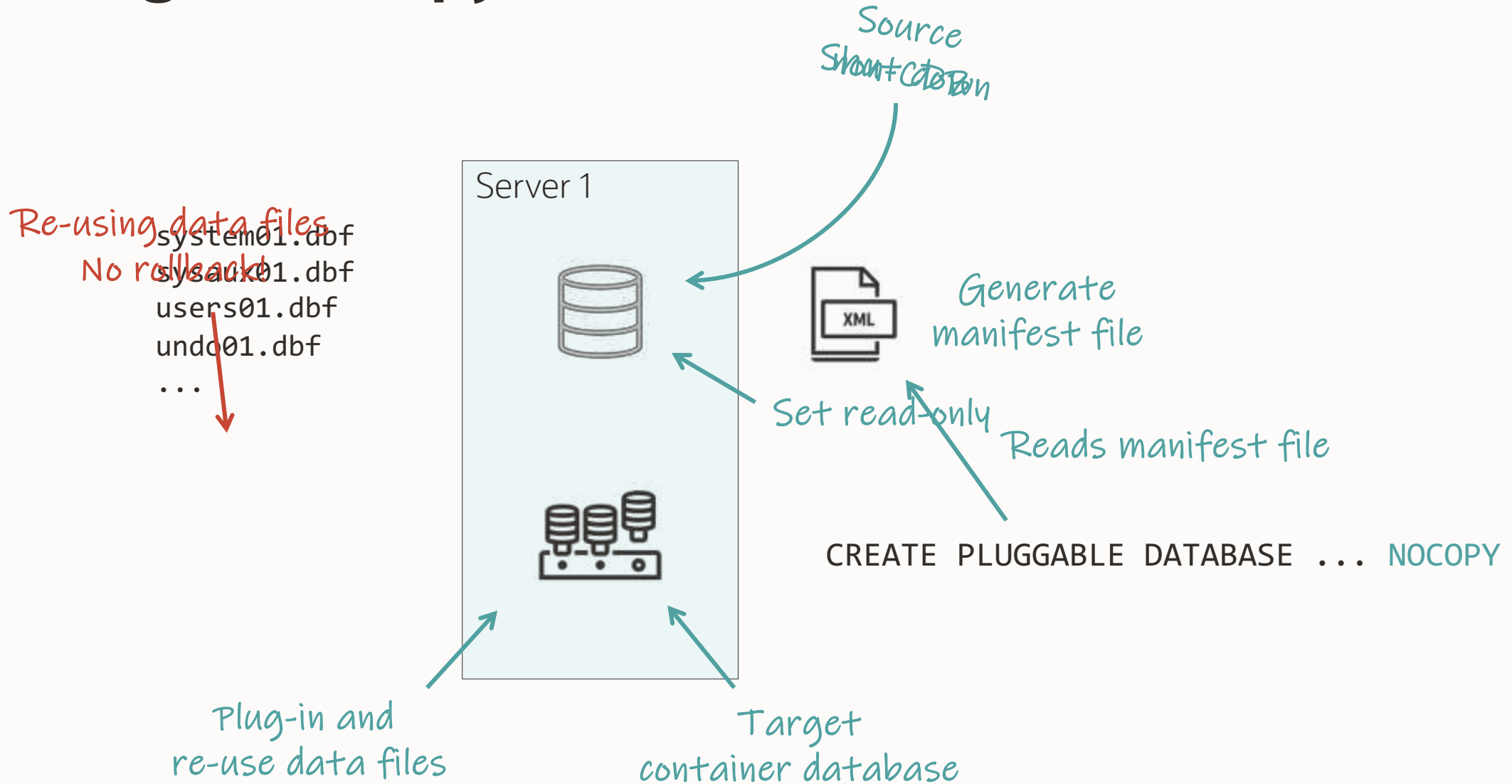
Refreshable

Plug-in Copy

Plug-in NoCopy



Plug-in NoCopy



MIGRATION

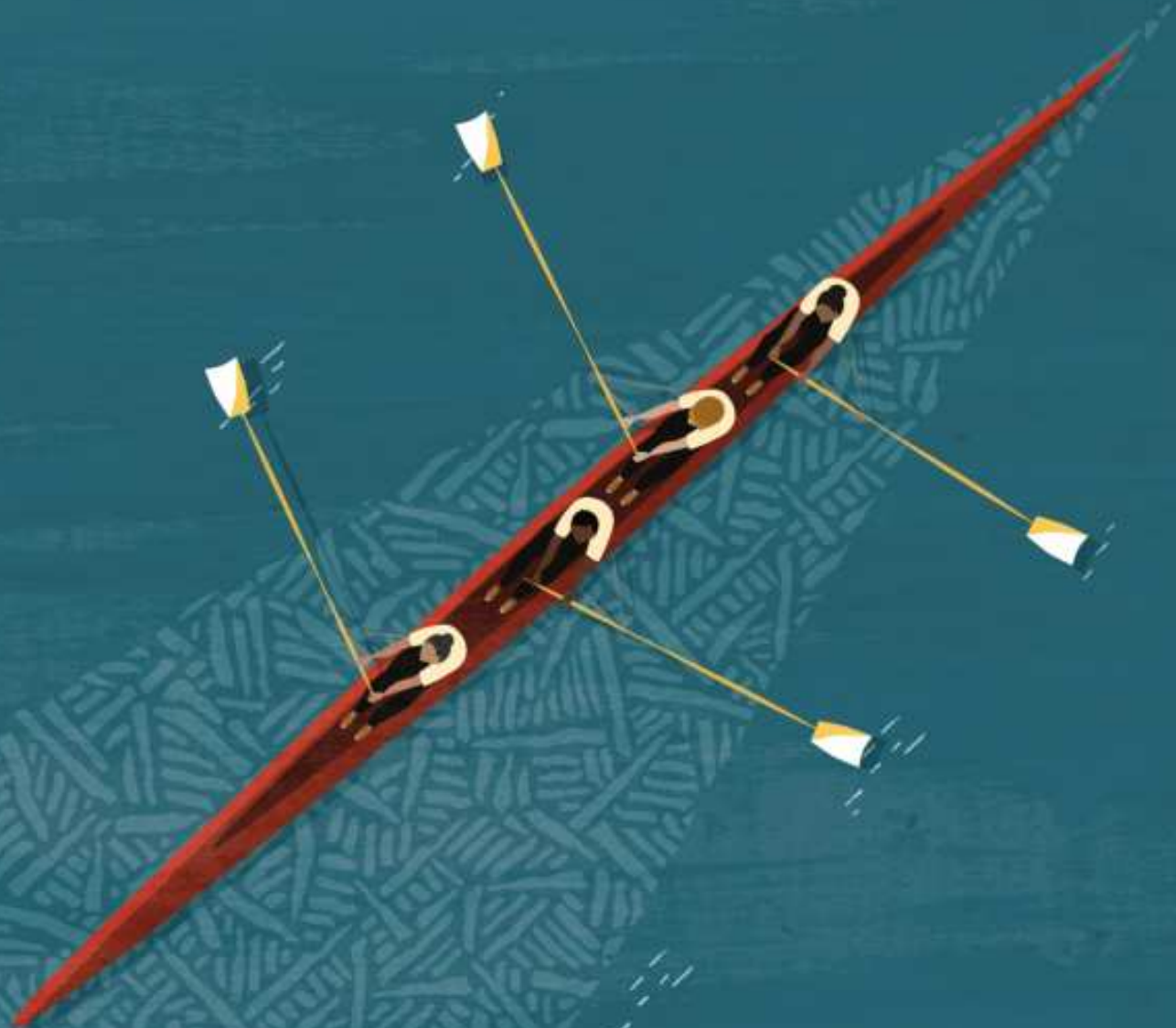
options

Other Options

Refreshable

Plug-in Copy

Plug-in NoCopy



Plug-in Copy

Source preserved
for rollback

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

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

Plug-in and
copies data files



Target
container database

Source
Snapshot



Generate
manifest file

Set read-only

Reads manifest file

CREATE PLUGGABLE DATABASE ... COPY

MIGRATION

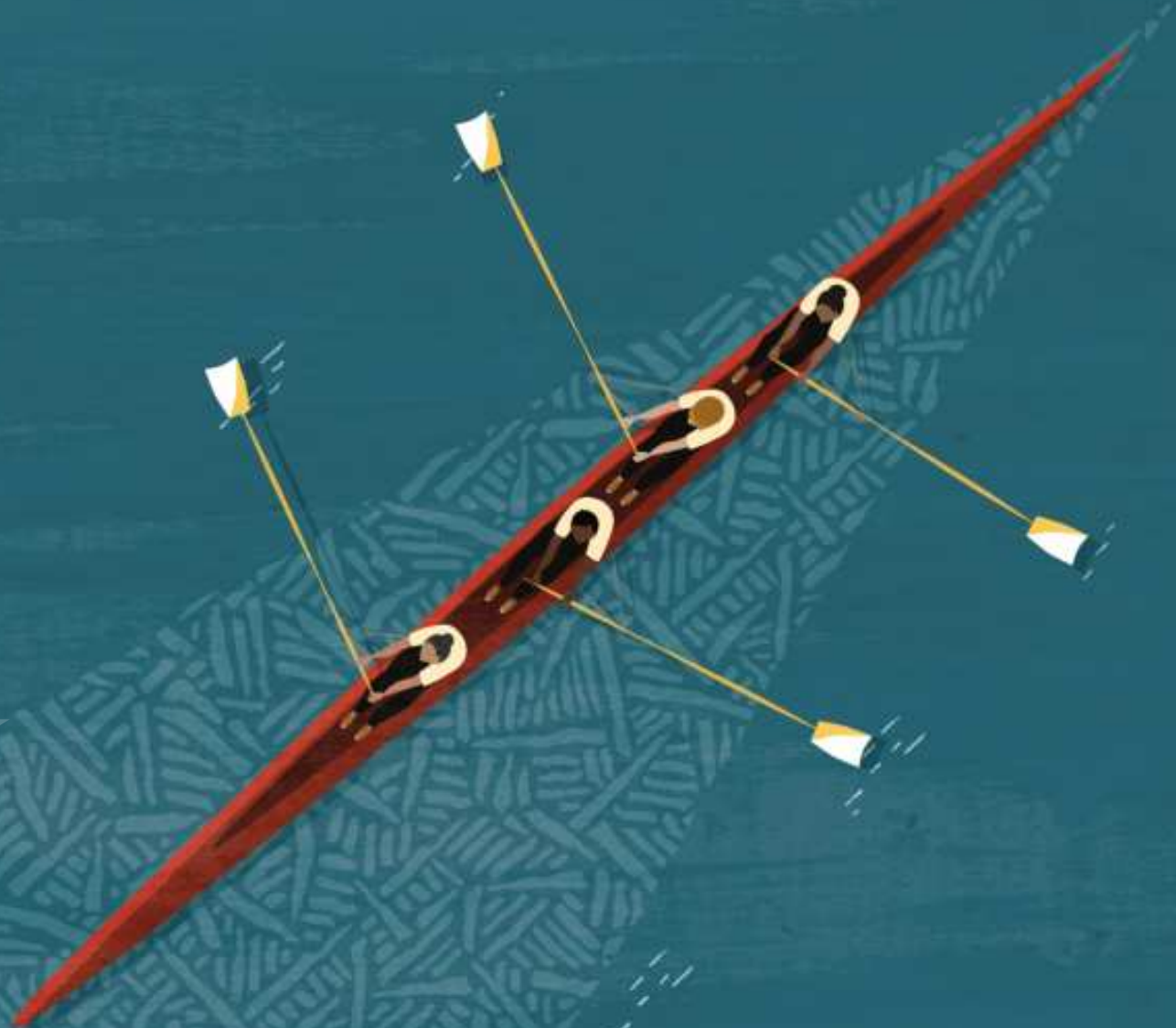
options

Other Options

Refreshable

Plug-in Copy

Plug-in NoCopy



MIGRATION

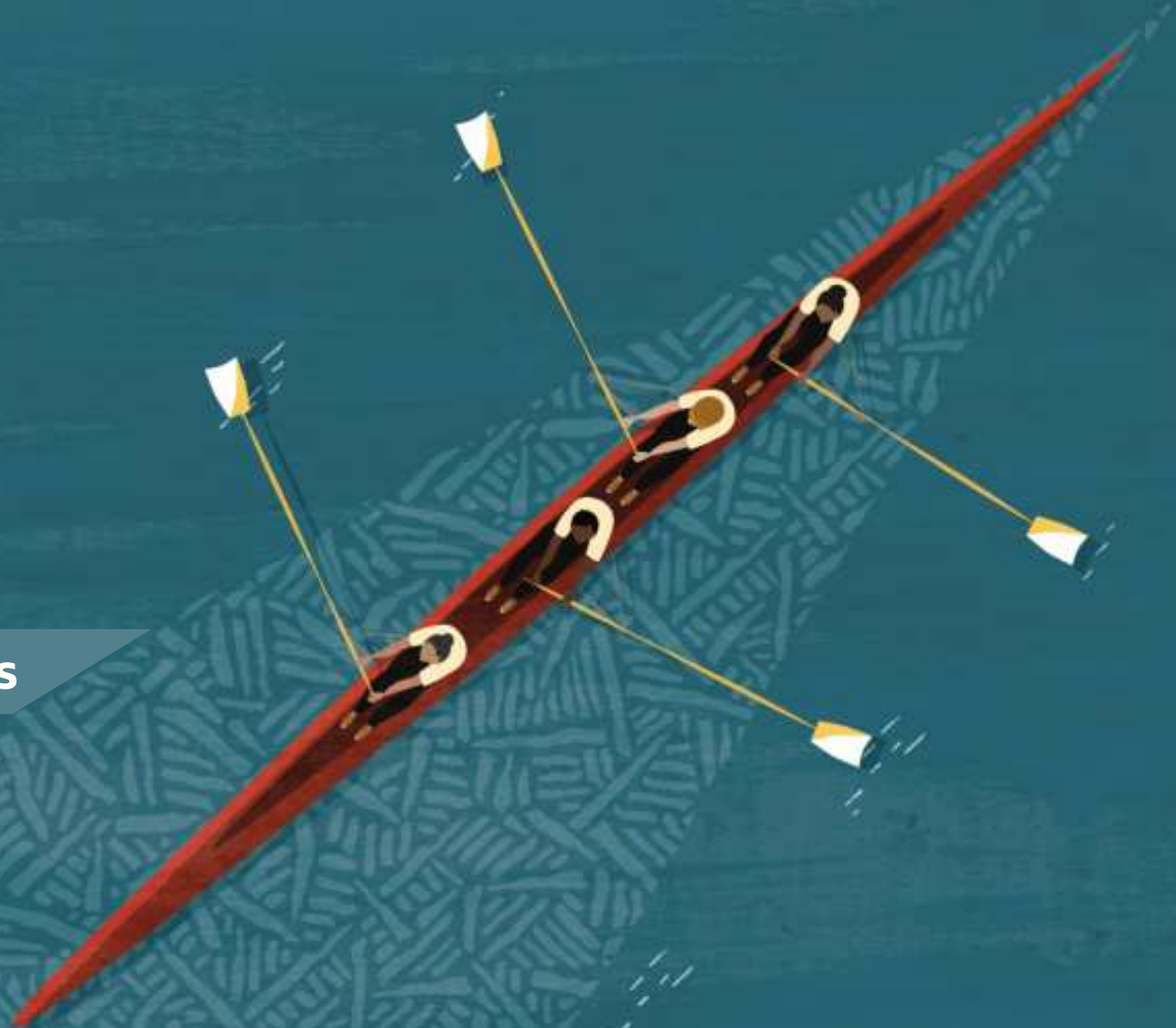
options

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



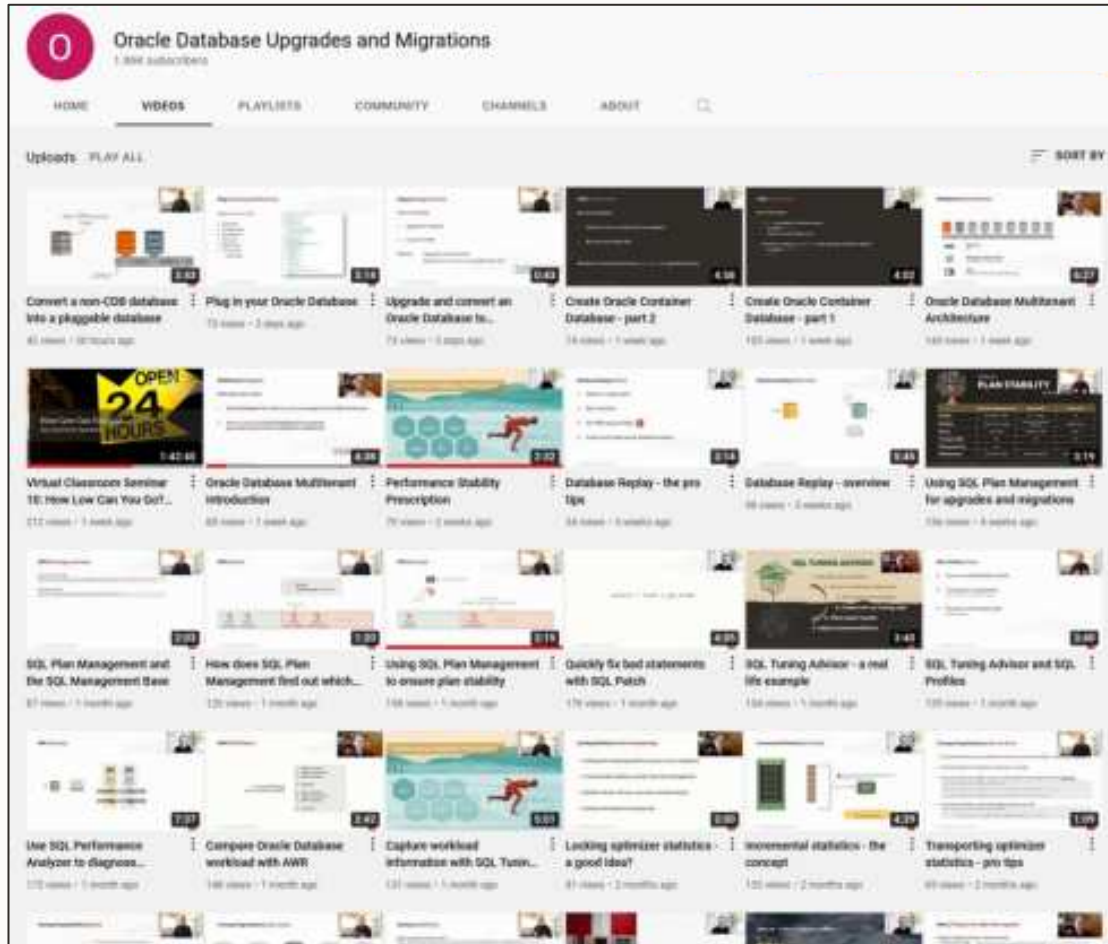
Other Options

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

YouTube | Oracle Database Upgrades and Migrations



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



Thank You

