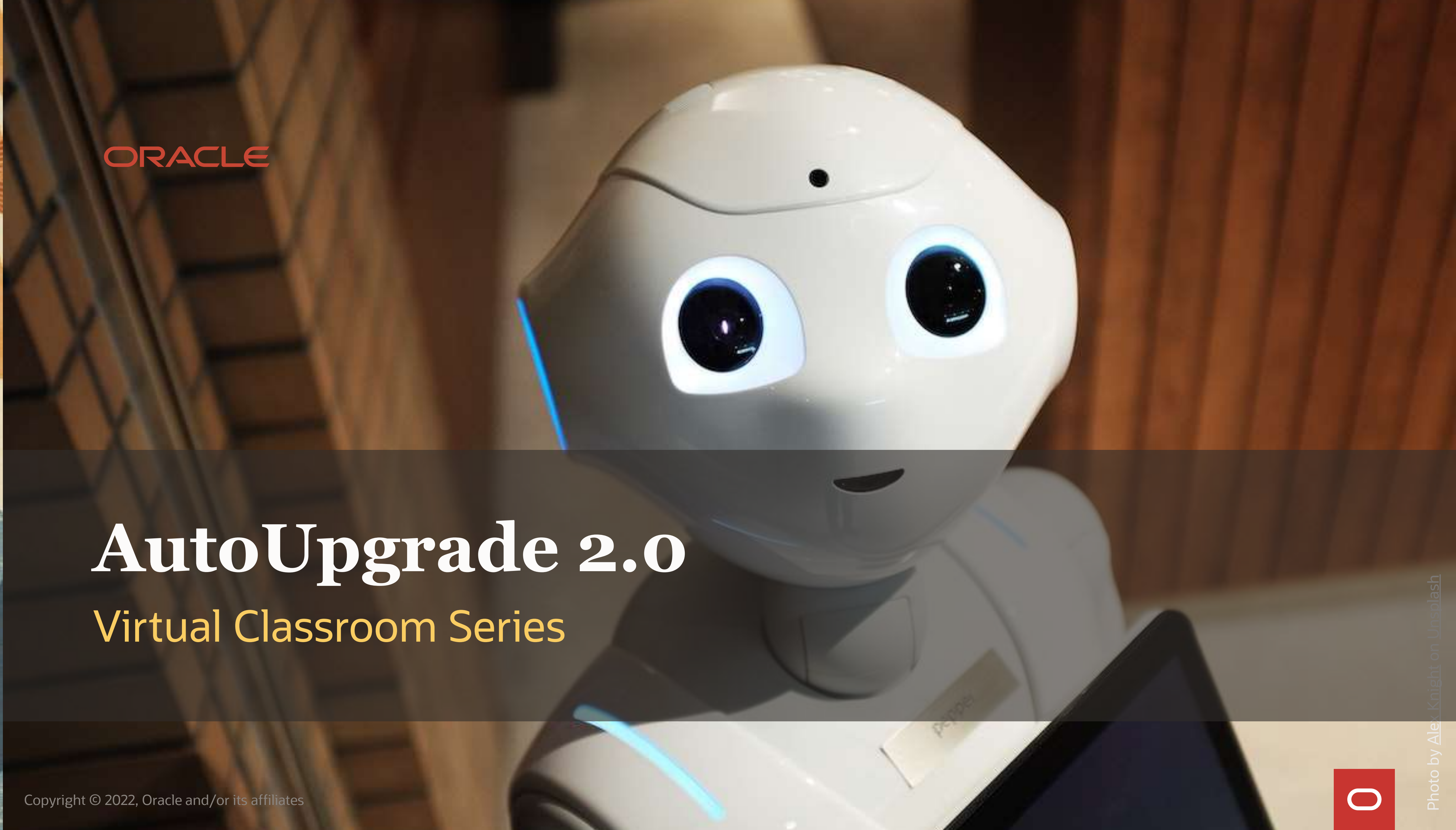


ORACLE

AutoUpgrade 2.0

Virtual Classroom Series

Photo by Alex Knight on Unsplash

ROY SWONGER
Vice President
Database Upgrade, Utilities & Patching

in royfswonger

@royfswonger

MIKE DIETRICH
Distinguished Product Manager
Database Upgrade and Migrations

in mikedietrich

@mikedietrichde

https://mikedietrichde.com

DANIEL OVERBY HANSEN
Senior Principal Product Manager
Cloud Migrations

dohdatabase

@dohdatabase

https://dohdatabase.com

RODRIGO JORGE
Senior Principal Product Manager
Database Patching and Upgrade

rodrigoaraujorge

@rodrigojorgedba

https://dbarj.com.br/en

Webinar | Get The Slides

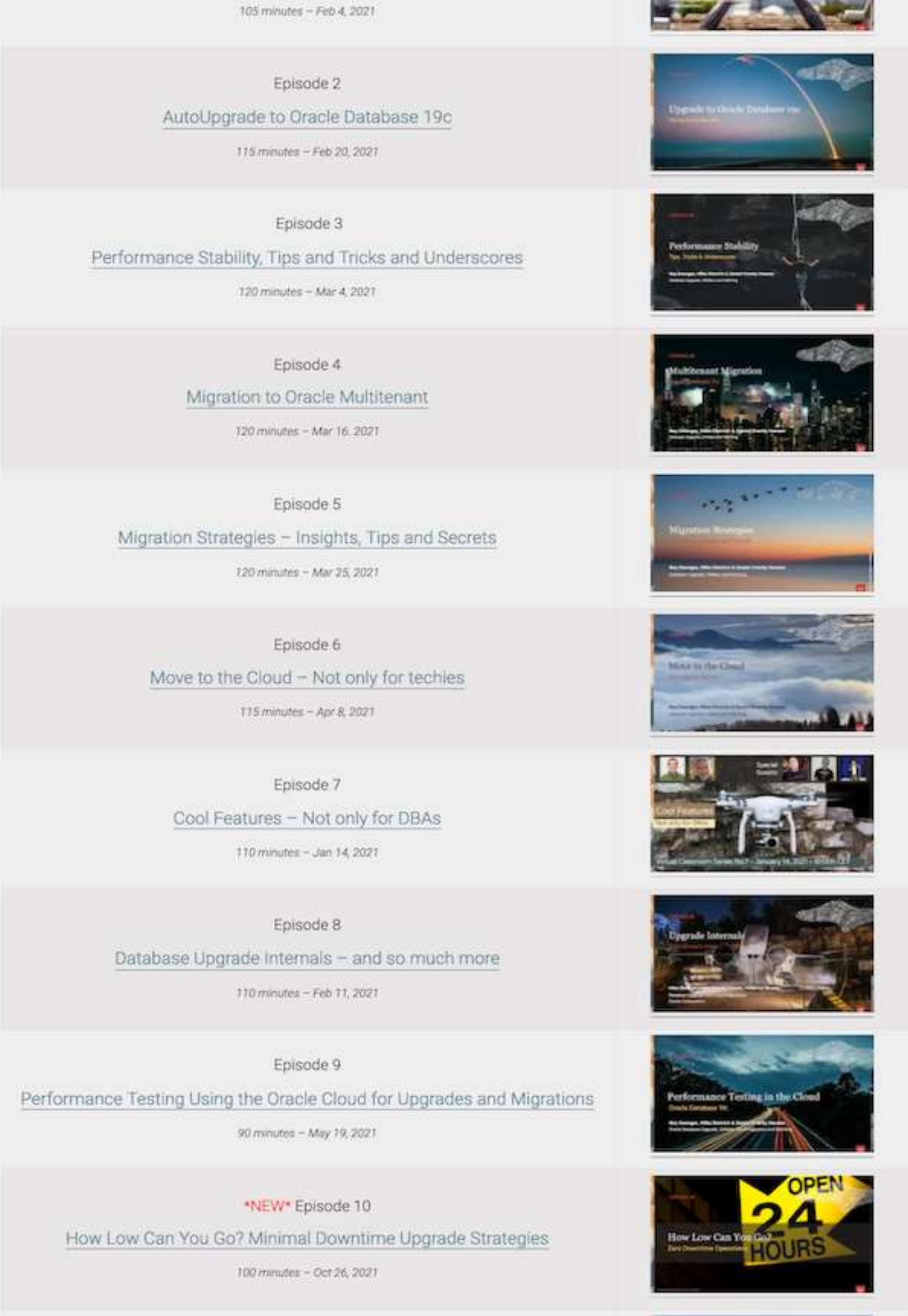
https://MikeDietrichDE.com/slides

ORACLE

AutoUpgrade 2.0

Virtual Classroom Series

Copyright © 2022, Oracle and/or its affiliates



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

Episode 7

Cool Features – Not only for DBAs

110 minutes – Jan 14, 2021

Episode 8

Database Upgrade Internals – and so much more

110 minutes – Feb 11, 2021

Episode 9

Performance Testing Using the Oracle Cloud for Upgrades and Migrations

90 minutes – May 19, 2021

NEW Episode 10

How Low Can You Go? Minimal Downtime Upgrade Strategies

100 minutes – Oct 26, 2021

Recorded Web Seminars

https://MikeDietrichDE.com/videos

ℹ Always use the latest version of AutoUpgrade

AutoUpgrade 2.0

Clone non-CDB PDB

TDE Support

REST API

Usability

Standby

CDB RAC

Refreshable Clone | Concept

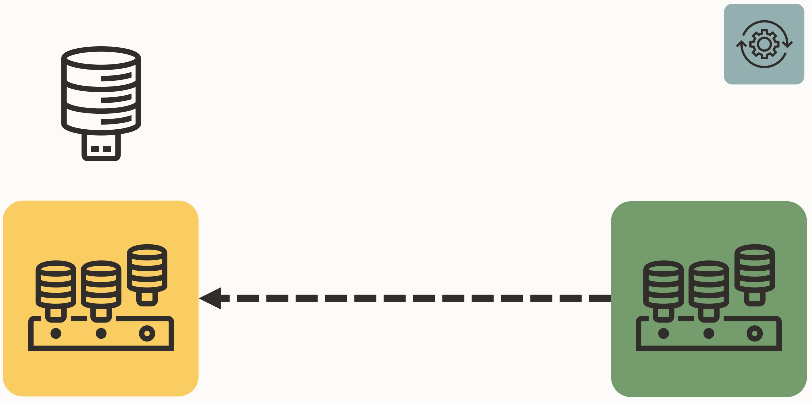
Create refreshable clone PDB

Refresh at will or automatic

Disconnect at will

Upgrade

Open



Pro tip: Works from Oracle Database 12.2

Clone Upgrade | Overview

AutoUpgrade supports refreshable clone upgrades across servers

- PDB Upgrade
  - Clone and upgrade
- non-CDB Migration
  - Clone and upgrade a non-CDB into a PDB

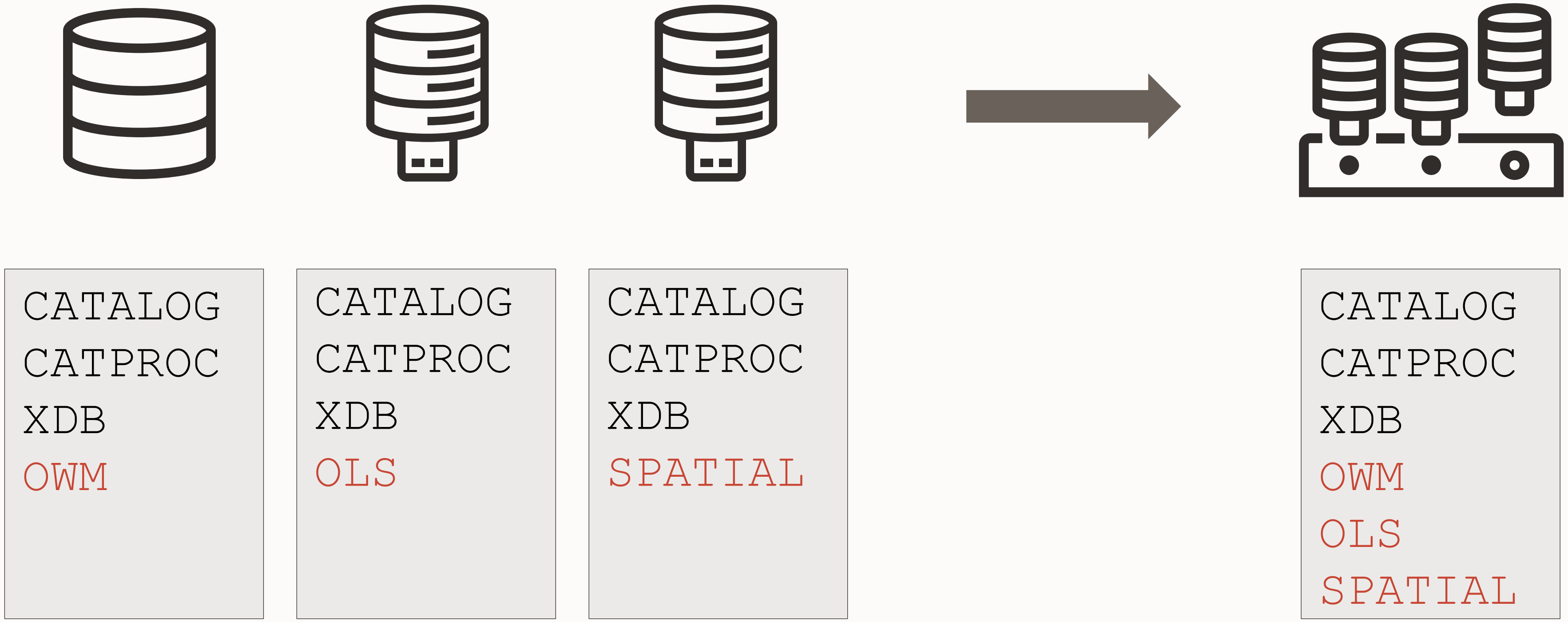
👍 You determine the point of synchronization

Parallel file copy is added automatically
Degree is adjusted according to resources available

Copy happens over SQL*Net
Optionally, use SQL*Net Encryption

Target CDB | Components

Target CDB$ROOT must contain a superset of non-CDBs and PDBs



CATALOG
CATPROC
XDB
OWM

CATALOG
CATPROC
XDB
OLS

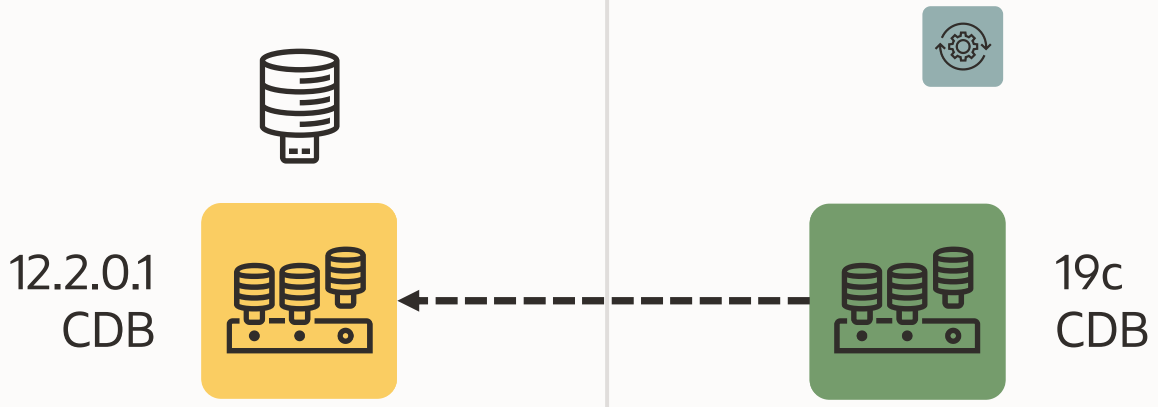
CATALOG
CATPROC
XDB
SPATIAL

CATALOG
CATPROC
XDB
OWM
OLS
SPATIAL

⚠ Target CDB$ROOT must be a component superset

PDB Upgrade | Concept

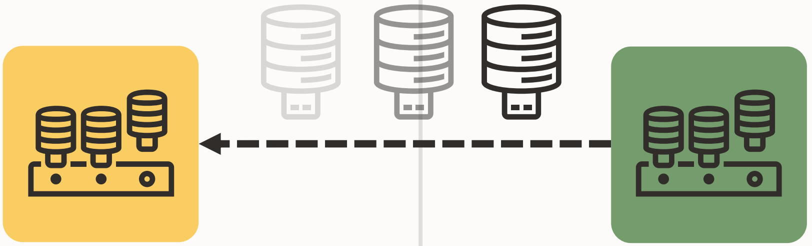
Unplug a PDB, plugin and upgrade



PDB Upgrade | Preparation

Source CDB

Target CDB



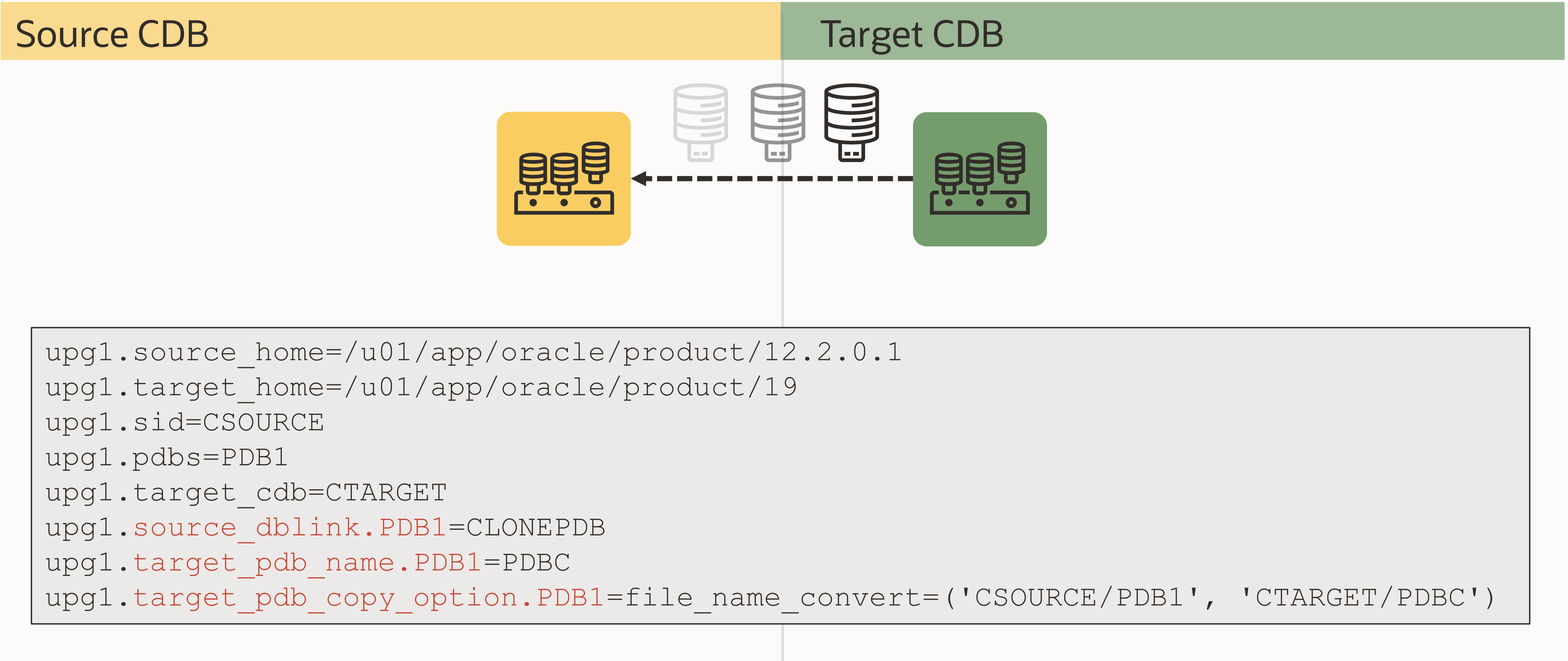
```
CREATE USER c##clone
       IDENTIFIED BY clone
       CONTAINER=ALL;

GRANT CREATE SESSION,
      CREATE PLUGGABLE DATABASE,
      SELECT_CATALOG_ROLE TO c##clone
      CONTAINER=ALL;

GRANT READ ON sys.enc$ TO c##clone
      CONTAINER=ALL;
```

```
CREATE DATABASE LINK CLONEPDB
       CONNECT TO c##clone
       IDENTIFIED BY clone
       USING 'CSOURCE';
```

Run fixups before initiating the clone operation

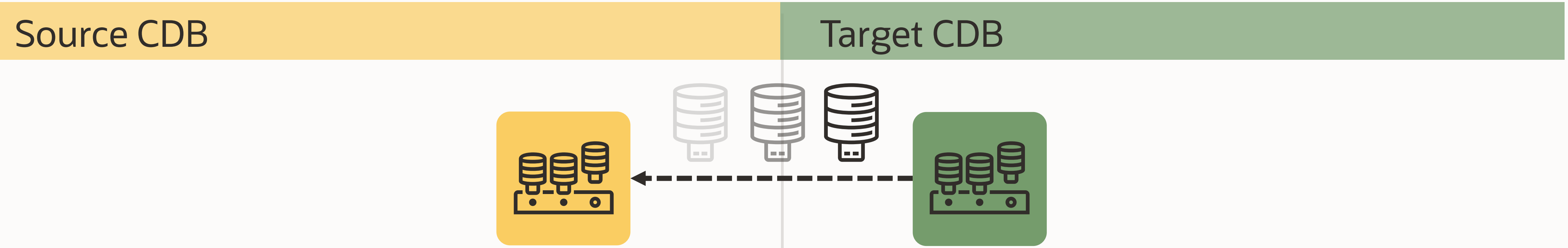
```
java -jar autoupgrade.jar -config PDB1.cfg -mode fixups
```

PDB Upgrade | AutoUpgrade



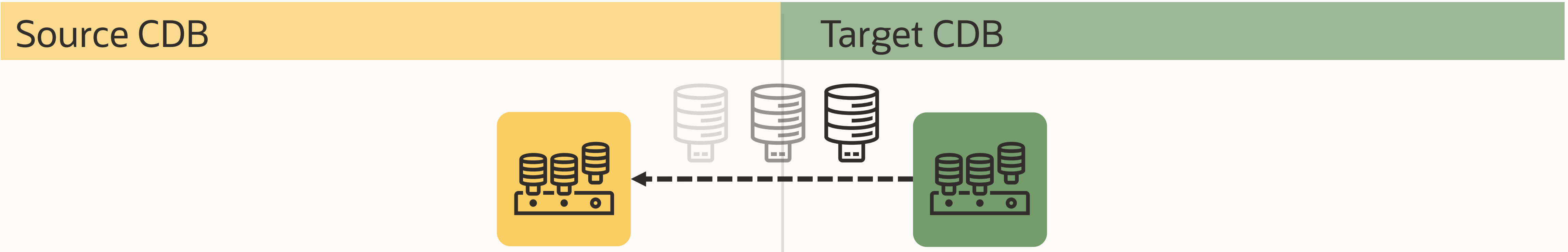
```
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=CSOURCE
upg1.pdbs=PDB1
upg1.target_cdb=CTARGET
upg1.source_dblink.PDB1=CLONEPDB
upg1.target_pdb_name.PDB1=PDBC
upg1.target_pdb_copy_option.PDB1=file_name_convert=('CSOURCE/PDB1', 'CTARGET/PDBC')
```

PDB Upgrade Refresh | AutoUpgrade



```
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=CSOURCE
upg1.pdbs=PDB1
upg1.target_cdb=CTARGET
upg1.source_dblink.PDB1=CLONEPDB 60
upg1.target_pdb_name.PDB1=PDBC
upg1.target_pdb_copy_option.PDB1=file_name_convert=('CSOURCE/PDB1', 'CTARGET/PDBC')
upg1.start_time=+1h30m
```

PDB Upgrade Refresh | alert.log



Source CDB

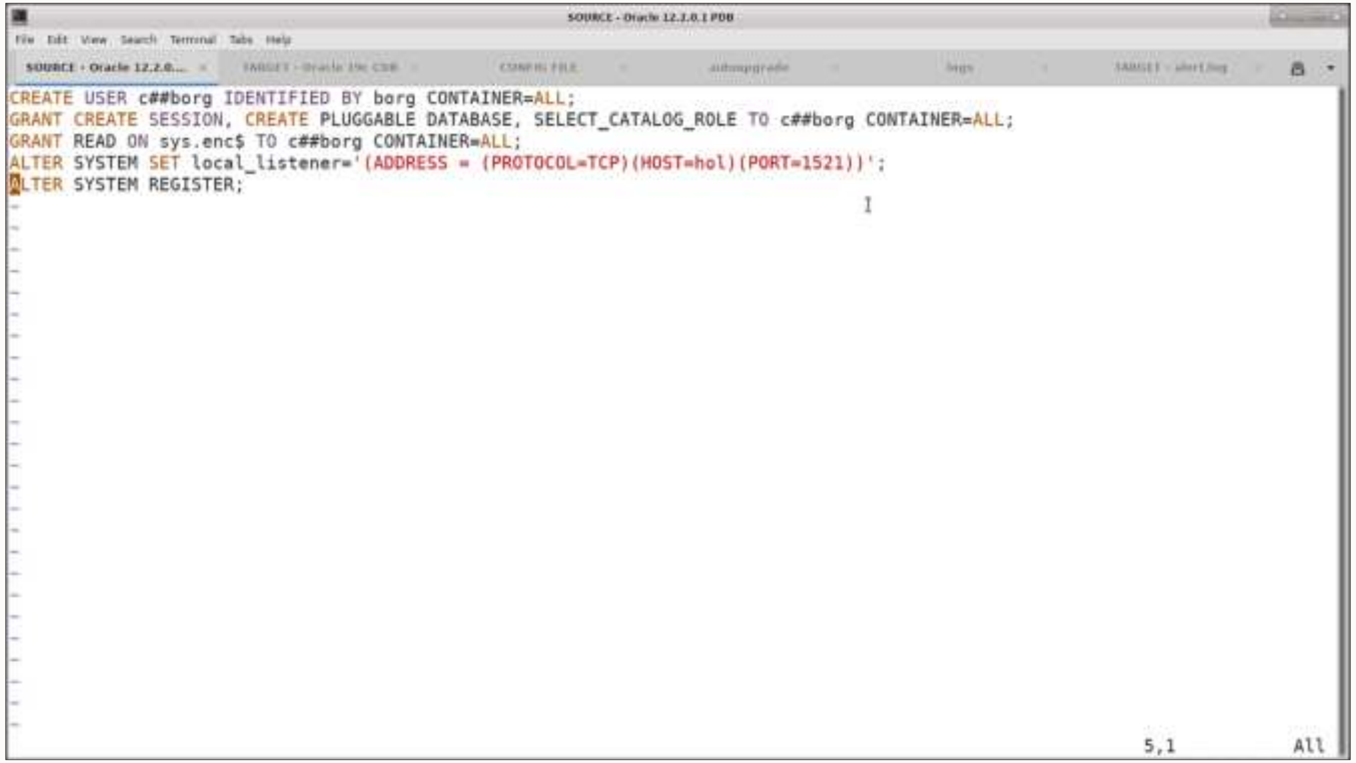
Target CDB

60 sec

```
2022-04-30T23:12:39.483201+02:00
PDB3(3):Media Recovery Complete (CDB2)
Completed: ALTER PLUGGABLE DATABASE PDB3 REFRESH

2022-04-30T23:13:36.973819+02:00
ALTER PLUGGABLE DATABASE PDB3 REFRESH

2022-04-30T23:13:39.371222+02:00
PDB3(3):Serial Media Recovery started

2022-04-30T23:13:39.527020+02:00
PDB3(3):Media Recovery Complete (CDB2)
Completed: ALTER PLUGGABLE DATABASE PDB3 REFRESH
```

👍 AutoUpgrade issues a final refresh of the PDB right before the upgrade starts

PDB Upgrade Refresh | Demo



SOURCE - Oracle 12.2.0.1 PDB
File Edit View Search Terminal Tabs Help
SOURCE - Oracle 12.2.0...
TARGET - Oracle 19c CDB
CONFIG FILE
autoupgrade
logs
TARGET - alert.log

```
CREATE USER c##borg IDENTIFIED BY borg CONTAINER=ALL;
GRANT CREATE SESSION, CREATE PLUGGABLE DATABASE, SELECT_CATALOG_ROLE TO c##borg CONTAINER=ALL;
GRANT READ ON sys.enc$ TO c##borg CONTAINER=ALL;
ALTER SYSTEM SET local_listener='(ADDRESS = (PROTOCOL=TCP)(HOST=hol)(PORT=1521))';
ALTER SYSTEM REGISTER;
```

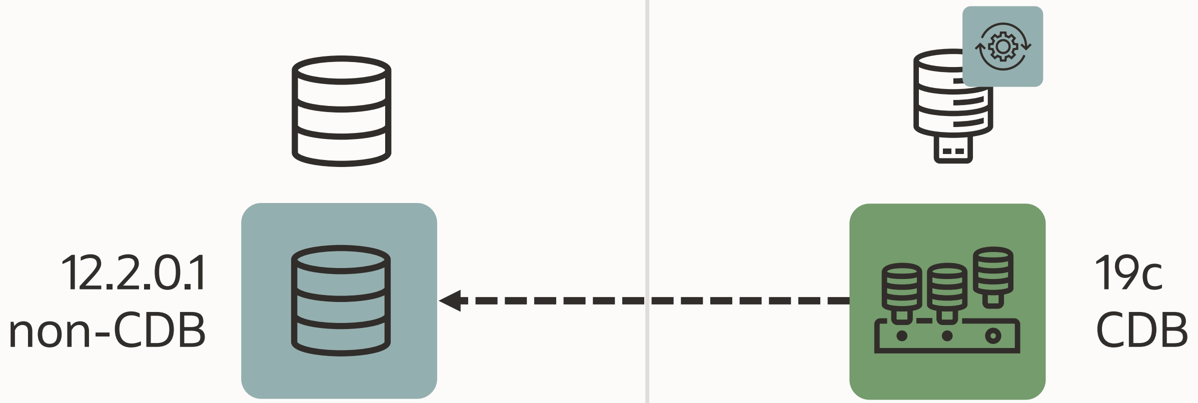
Watch on YouTube

👍 The source PDB stays intact to allow rollback

Non-CDB Migration | Concept

Migrate a non-CDB into a PDB

- Preupgrade
- Clone
- Upgrade
- Non-CDB to PDB



Non-CDB Migration | Preparation

Source non-CDB

Target CDB

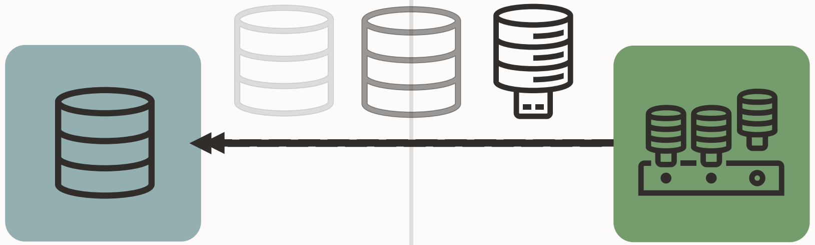
```
CREATE USER clone
       IDENTIFIED BY clone;

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

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

```
CREATE DATABASE LINK CLONENONCDB
       CONNECT TO clone
       IDENTIFIED BY clone
       USING 'NONCDB';
```

Non-CDB Migration | AutoUpgrade

Source non-CDB

Target CDB

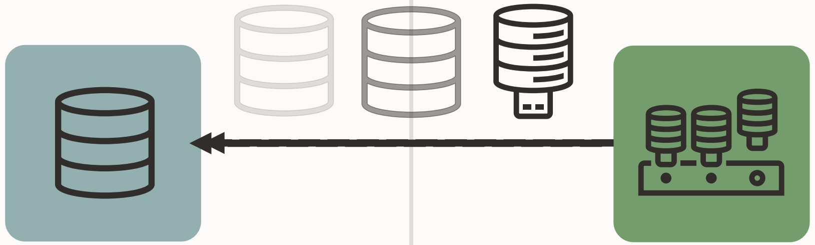


```
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=NONCDB
upg1.target_cdb=CTARGET
upg1.source_dblink.NONCDB=CLONENONCDB
upg1.target_pdb_name.NONCDB=PDBNONCDB
upg1.target_pdb_copy_option.NONCDB=file_name_convert=('NONCDB', 'CTARGET/PDBNONCDB')
```

Non-CDB Migration Refresh | AutoUpgrade

Source non-CDB

Target CDB



```
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=NONCDB
upg1.target_cdb=CTARGET
upg1.source_dblink.NONCDB=CLONENONCDB 300
upg1.target_pdb_name.NONCDB=PDBNONCDB
upg1.target_pdb_copy_option.NONCDB=file_name_convert=('NONCDB', 'CTARGET/PDBNONCDB')
upg1.start_time=+45m
```

👍 The source non-CDB stays intact to allow rollback

👍 Once the clone operations are completed, you can remove the clone user and the database link

⚠ Be VERY aware when you have a standby database

AutoUpgrade 2.0

Clone non-CDB PDB

TDE

REST API

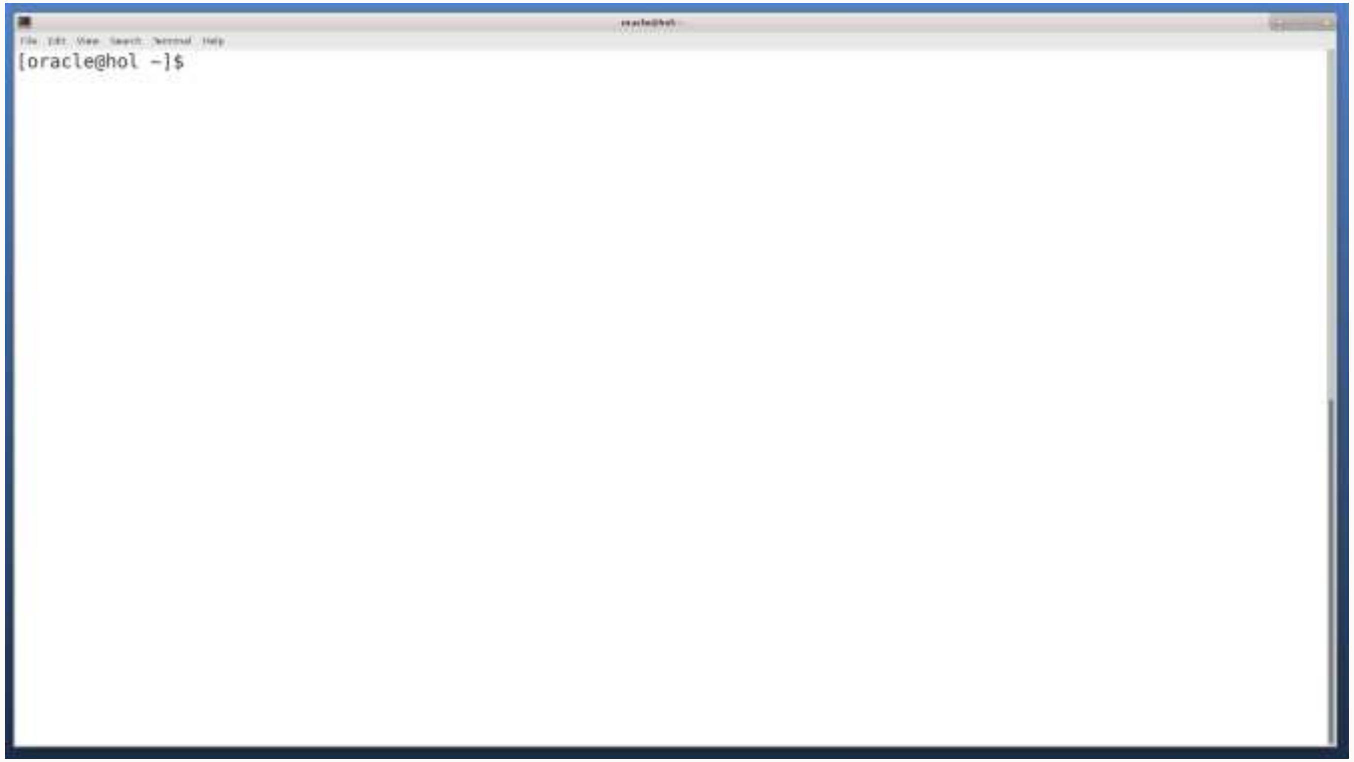
Usability

Standby

CDB RAC

⚠ Previously AutoUpgrade had no way of storing sensitive information such as keystore passwords ℹ Now, AutoUpgrade has its own keystore

TDE | Demo - Upgrading encrypted PDB



```
[oracle@hol ~]$
```

Watch on YouTube

AutoUpgrade fully supports Transparent Data Encryption

- Isolated keystore mode coming in a later version

TDE | Keystore

- New config file parameter: `global.keystore`
- Governs directory of AutoUpgrade keystore
- Password protected software keystore
- Optionally, an auto-open keystore

TDE | Keystore

```
$ cat DB12.cfg

global.keystore=/etc/oracle/keystores/autoupgrade/DB12
...


$ ls -l /etc/oracle/keystores/autoupgrade/DB12

-rw-------. 1 oracle dba 720 Mar 28 14:56 ewallet.p12
```

TDE | Keystore

AutoUpgrade keystore contains

- Database TDE keystore passwords (user-supplied)
- Passphrases or transport secrets (auto-generated)

TDE | Keystore

```
$ java -jar autoupgrade.jar -config DB12.cfg -load_password


TDE> add DB12

Enter your secret/Password:
Re-enter your secret/Password:
```

TDE | Keystore

In the TDE console, the following commands are available:

- `add`
- `delete`
- `list`
- `save`
- `help`
- `exit`

A password protects the AutoUpgrade keystore, unless you also create an auto-login keystore

TDE | Keystore

```
$ java -jar autoupgrade.jar -config DB12.cfg -load_password

TDE> save

Convert the keystore to auto-login [YES|NO] ?




$ ls -l /etc/oracle/keystores/autoupgrade/DB12

-rw-------. 1 oracle dba 765 Mar 28 14:56 cwallet.sso
-rw-------. 1 oracle dba 720 Mar 28 14:56 ewallet.p12
```

Protect the AutoUpgrade keystore like you protect any other keystore

- Apply restrictive file system permissions
- Audit access
- Back it up

TDE | Upgrade Non-CDB or CDB

To upgrade an encrypted non-CDB or entire CDB

- An auto-login TDE keystore must be present

```
SQL> -- LOCAL_AUTOLOGIN is also usable
SQL> select wallet_type from v$encryption_wallet;

AUTOLOGIN
```

You do not need an AutoUpgrade keystore

TDE | Upgrade Non-CDB or CDB

Workaround

- If database has issues finding the right keystore,
  you can override TNS_ADMIN location in config file:

```
upg1.source_tns_admin_dir=/u01/app/oracle/admin/DB12/tns_admin
upg1.target_tns_admin_dir=/u01/app/oracle/admin/DB12/tns_admin
```

⚠ Defining keystore location in *sqlnet.ora* is deprecated in Oracle Database 19c

👍 Use `WALLET_ROOT` parameter to define keystore location and use new TDE functionality

TDE | Upgrade Non-CDB or CDB

Use AutoUpgrade to switch to keystore configuration using `WALLET_ROOT`

Create text file with new initialization parameters:

```
$ cat /tmp/au-pfile-tde.txt

WALLET_ROOT='/etc/oracle/keystores/$ORACLE_SID'
TDE_CONFIGURATION='KEYSTORE_CONFIGURATION=FILE'
```

TDE | Upgrade Non-CDB or CDB

Instruct AutoUpgrade to add parameters during and after upgrade:

```
upg1.add_during_upgrade_pfile=/tmp/au-pfile-tde.txt
upg1.add_after_upgrade_pfile=/tmp/au-pfile-tde.txt
```

AutoUpgrade automatically copies keystore from previous location into location defined by `WALLET_ROOT`

Pro tip: Get more details in blog post

TDE | Upgrade Encrypted Non-CDB and Convert

To upgrade an encrypted non-CDB and afterwards convert it to a PDB:

Create config file

```
global.autoupg_log_dir=/u01/app/oracle/cfgtoollogs/autoupgrade
global.keystore=/u01/app/oracle/admin/autoupgrade/keystore

upg1.log_dir=/u01/app/oracle/cfgtoollogs/autoupgrade/DB12
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=DB12
upg1.target_cdb=CDB2
```

TDE | Upgrade Encrypted Non-CDB and Convert

Analyze the non-CDB for upgrade readiness

```
$ java -jar autoupgrade.jar -config DB12.cfg -mode analyze
```

Summary report will show which keystore passwords are needed:

```
REQUIRED ACTIONS
================
       1.  Perform the specified action ...
       ORACLE_SID                        Action Required
       ------------------------------    ------------------------
       DB12                              Add TDE password
       CDB2                              Add TDE password
```

TDE | Upgrade Encrypted Non-CDB and Convert

Start TDE console to load passwords

```
$ java -jar autoupgrade.jar -config DB12.cfg -load_password
```

Add database keystore passwords

```
TDE> add DB12

TDE> add CDB2
```

Start upgrade

```
$ java -jar autoupgrade.jar -config DB12.cfg -mode deploy
```

TDE | Upgrade Encrypted PDB

To upgrade an encrypted PDB using unplug-plug:

Create config file

```
global.autoupg_log_dir=/u01/app/oracle/cfgtoollogs/autoupgrade
global.keystore=/u01/app/oracle/admin/autoupgrade/keystore

upg1.log_dir=/u01/app/oracle/cfgtoollogs/autoupgrade/PDB1
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=CDB1
upg1.target_cdb=CDB2
upg1.pdbs=PDB1
```

TDE | Upgrade Encrypted PDB

Analyze the PDB 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
```

TDE | Upgrade Encrypted PDB

Start TDE console to load passwords

```
$ java -jar autoupgrade.jar -config PDB1.cfg -load_password
```

Add database keystore passwords

```
TDE> add CDB1

TDE> add CDB2
```

Start upgrade

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

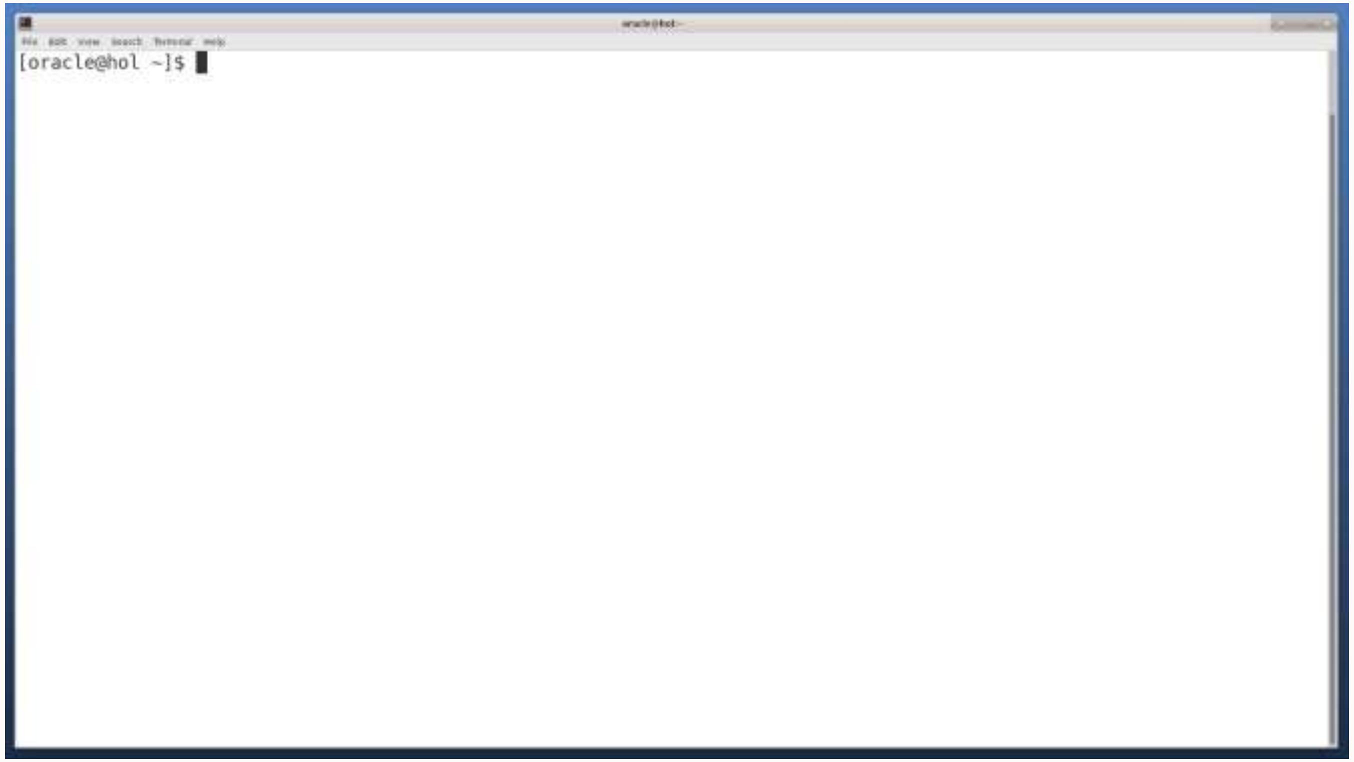
AutoUpgrade also supports
converting an encrypted non-CDB to PDB

AutoUpgrade is compatible with Secure External Password Store

- Supported from Oracle Database 12.2

Pro tip: Get more details in blog post

TDE | Demo - Upgrading and converting to PDB



```
[oracle@hol ~]$ 
```

Watch on YouTube

TDE | Additional Information

- Blog post series
- Configuring an External Store for a Keystore Password

AutoUpgrade 2.0

Clone non-CDB PDB

TDE Support

REST API

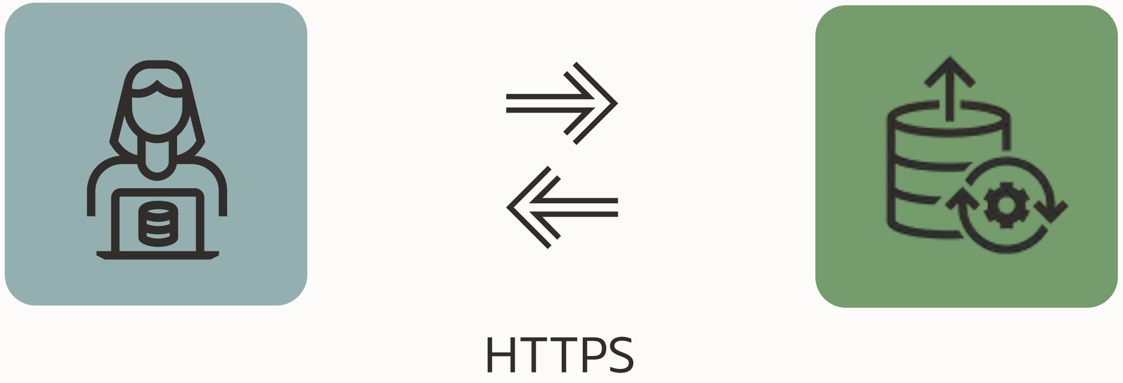
Usability

Standby

CDB RAC

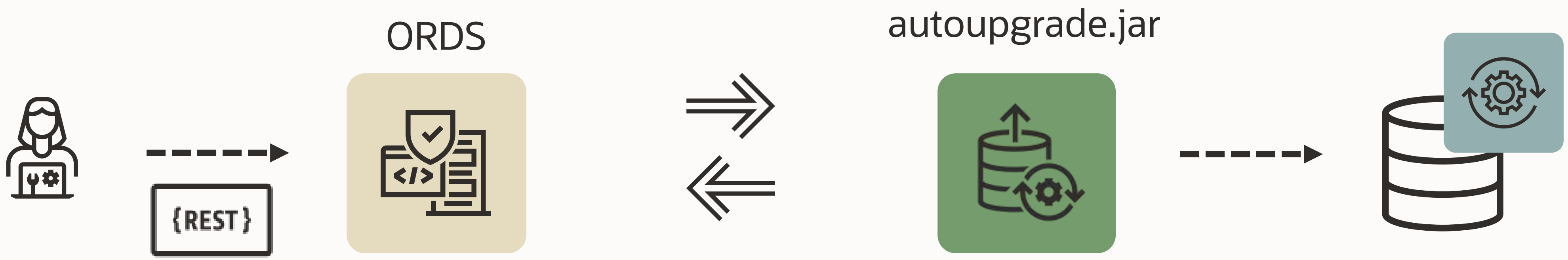
REST API | Why use it?

- Well-known API
- Flexibility
- Simplicity
- Upgrade-on-demand



REST API | How it works

AutoUpgrade uses ORDS to handle the Java calls



Requirement:

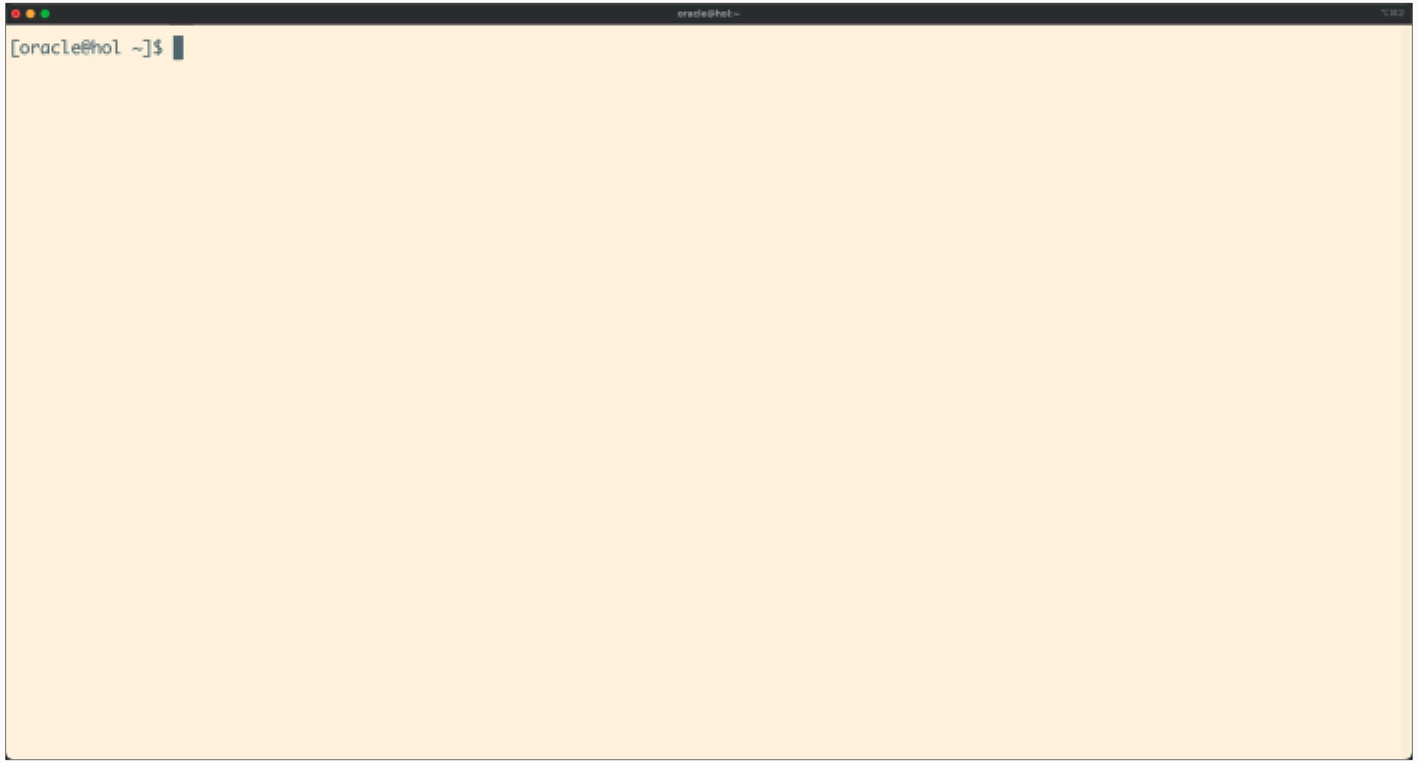
- Oracle REST Data Services (ORDS) 22.1.0 or later

REST API | Enable

```
$ #Enable the AutoUpgrade API
$ java -jar ords.war set-property autoupgrade.api.enabled true

$ #Set the location of AutoUpgrade log files
$ java -jar ords.war set-property autoupgrade.api.loglocation /u01/autoupgrade_logs

$ #Which AutoUpgrade.jar to use
$ java -jar ords.war set-property autoupgrade.api.aulocation /u01/autoupgrade.jar

$ #Which Java to use
$ java -jar ords.war set-property autoupgrade.api.jvmlocation /bin/java
```

Always use a separate instance of ORDS for AutoUpgrade APIs ℹ Always protect the REST APIs endpoints with a firewall

REST API | Demo - Install



```
[oracle@hol ~]$ 
```

Watch on YouTube

REST API | Config file vs. JSON

Config File

```
global.autoupg_log_dir=/home/oracle/logs
upg1.source_home=/u01/app/product/11
upg1.target_home=/u01/app/product/19
upg1.sid=UPGR
upg1.log_dir=/home/oracle/logs
upg1.restoration=no
```

JSON for REST API

```
{
  "global": {
    "autoupg_log_dir": "/home/oracle/logs"
  },
  "jobs": [{
    "source_home": "/u01/app/product/11",
    "target_home": "/u01/app/product/19",
    "sid": "UPGR",
    "log_dir": "/home/oracle/logs",
    "restoration": "no"
  }]
}
```

REST API | Config file vs. JSON

Config File

```
global.autoupg_log_dir=/home/oracle/logs
upg1.source_home=/u01/app/product/11
upg1.target_home=/u01/app/product/19
upg1.sid=UPGR
upg1.log_dir=/home/oracle/logs
upg1.restoration=no
```

JSON for REST API

```
{
  "global": {
    "autoupg_log_dir": "/home/oracle/logs"
  },
  "jobs": [{
    "source_home": "/u01/app/product/11",
    "target_home": "/u01/app/product/19",
    "sid": "UPGR",
    "log_dir": "/home/oracle/logs",
    "restoration": "no"
  }]
}
```

REST API | Config file vs. JSON

Config File

```
global.autoupg_log_dir=/home/oracle/logs
upg1.source_home=/u01/app/product/11
upg1.target_home=/u01/app/product/19
upg1.sid=UPGR
upg1.log_dir=/home/oracle/logs
upg1.restoration=no
```

JSON for REST API

```
{
  "global": {
    "autoupg_log_dir": "/home/oracle/logs"
  },
  "jobs": [{
    "source_home": "/u01/app/product/11",
    "target_home": "/u01/app/product/19",
    "sid": "UPGR",
    "log_dir": "/home/oracle/logs",
    "restoration": "no"
  }]
}
```

REST API | Config file vs. JSON

Config File

```
global.autoupg_log_dir=/home/oracle/logs
upg1.source_home=/u01/app/product/11
upg1.target_home=/u01/app/product/19
upg1.sid=UPGR
upg1.log_dir=/home/oracle/logs
upg1.restoration=no
```

JSON for REST API

```
{
  "global": {
    "autoupg_log_dir": "/home/oracle/logs"
  },
  "jobs": [{
    "source_home": "/u01/app/product/11",
    "target_home": "/u01/app/product/19",
    "sid": "UPGR",
    "log_dir": "/home/oracle/logs",
    "restoration": "no"
  }]
}
```

REST API | Config file vs. JSON

Config File

```
global.autoupg_log_dir=/home/oracle/logs
upg1.source_home=/u01/app/product/11
upg1.target_home=/u01/app/product/19
upg1.sid=UPGR
upg1.log_dir=/home/oracle/logs
upg1.restoration=no
```

JSON for REST API

```
{
  "global": {
    "autoupg_log_dir": "/home/oracle/logs"
  },
  "jobs": [{
    "source_home": "/u01/app/product/11",
    "target_home": "/u01/app/product/19",
    "sid": "UPGR",
    "log_dir": "/home/oracle/logs",
    "restoration": "no"
  }]
}
```

REST API | Methods

Methods available in the REST API

task	(GET / POST)
tasks	(GET)
status	(GET)
progress	(GET)
console	(GET)
log	(GET)

Only API with POST method

Pro tip: Read more about REST APIs

REST API | Create a New AutoUpgrade Task

```
$ curl -k --data-binary "@UPGR.json" -X POST --header "Content-Type:application/json"
'https://localhost:8443/ords/autoupgrade/task?mode=analyze'
```

Ignore verification of the certificate

JSON file

API method

Input file type

Endpoint API

Equal

AutoUpgrade configuration

```
{
    "taskid": "job_2022_04_27_05.17.24.146_0",
    "status": "submitted",
    "message": "",
    "link": "https://localhost:8443/ords/autoupgrade/task?taskid=job_2022_04_27_05.17.24.146_0",
    "config": {
        "global": {
```

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

```
                "source_home": "/u01/app/oracle/product/11.2.0.4",
                "target_home": "/u01/app/oracle/product/19",
                "sid": "UPGR",
                "log_dir": "/home/oracle/logs",
                "restoration": "no"
            }
        ]
    }
}
```

REST API | List all Tasks

```
$ curl -k https://localhost:8443/ords/autoupgrade/tasks
```

```
{
    "total_tasks": 1,
    "tasks": [
        {
            "mode": "analyze",
            "taskid": "job_2022_04_27_05.17.24.146_0",
            "config": {
                "jobs": [
                    {
                        "source_home": "/u01/app/oracle/product/11.2.0.4",
                        "sid": "UPGR"
                    }
                ]
            },
            "link": "https://localhost:8443/ords/autoupgrade/task?taskid=job_2022_04_27_05.17.24.146_0"
        }
    ]
}
```

Task Identifier

REST API | Get Specific Task

```
$ curl -k 'https://localhost:8443/ords/autoupgrade/task?taskid=job_2022_04_27_05.17.24.146_0'
```

Task Identifier

```
{
    "taskid": "job_2022_04_27_05.17.24.146_0",
    "status": "finished",
    "message": "",
    "link": "https://localhost:8443/ords/autoupgrade/task?taskid=job_2022_04_27_05.17.24.146_0",
    "config": {
        "global": {
            "autoupg_log_dir": "/home/oracle/logs"
        },
        "jobs": [
            {
                "source_home": "/u01/app/oracle/product/11.2.0.4",
                "target_home": "/u01/app/oracle/product/19",
                "sid": "UPGR",
                "log_dir": "/home/oracle/logs",
                "restoration": "no"
            }
        ]
    }
}
```

Task Status

REST API | Get console output for Job

```
$ curl -k 'https://localhost:8443/ords/autoupgrade/console?taskid=job_2022_04_27_05.17.24.146_0'
```

```
AutoUpgrade is not fully tested on OpenJDK 64-Bit Server VM, Oracle recommends to use Java HotSpot(TM)
AutoUpgrade 22.2.220324 launched with default internal options
Processing config file ...
+--------------------------------+
| Starting AutoUpgrade execution |
+--------------------------------+
1 Non-CDB(s) will be analyzed
Job 100 database upgr
Job 100 completed
------------------- Final Summary --------------------
Number of databases            [ 1 ]

Jobs finished                  [1]
Jobs failed                    [0]

Please check the summary report at:
/u01/AU_REST/autoupgrade_logs/job_2022_04_27_05.17.24.146_0/cfgtoollogs/upgrade/auto/status/status.html
/u01/AU_REST/autoupgrade_logs/job_2022_04_27_05.17.24.146_0/cfgtoollogs/upgrade/auto/status/status.log
```

REST API | Resubmit in deploy mode

```
$ curl -k -X POST 'https://localhost:8443/ords/autoupgrade/task?taskid=job_2022_04_27_05.17.24.146_0&mode=deploy'
```

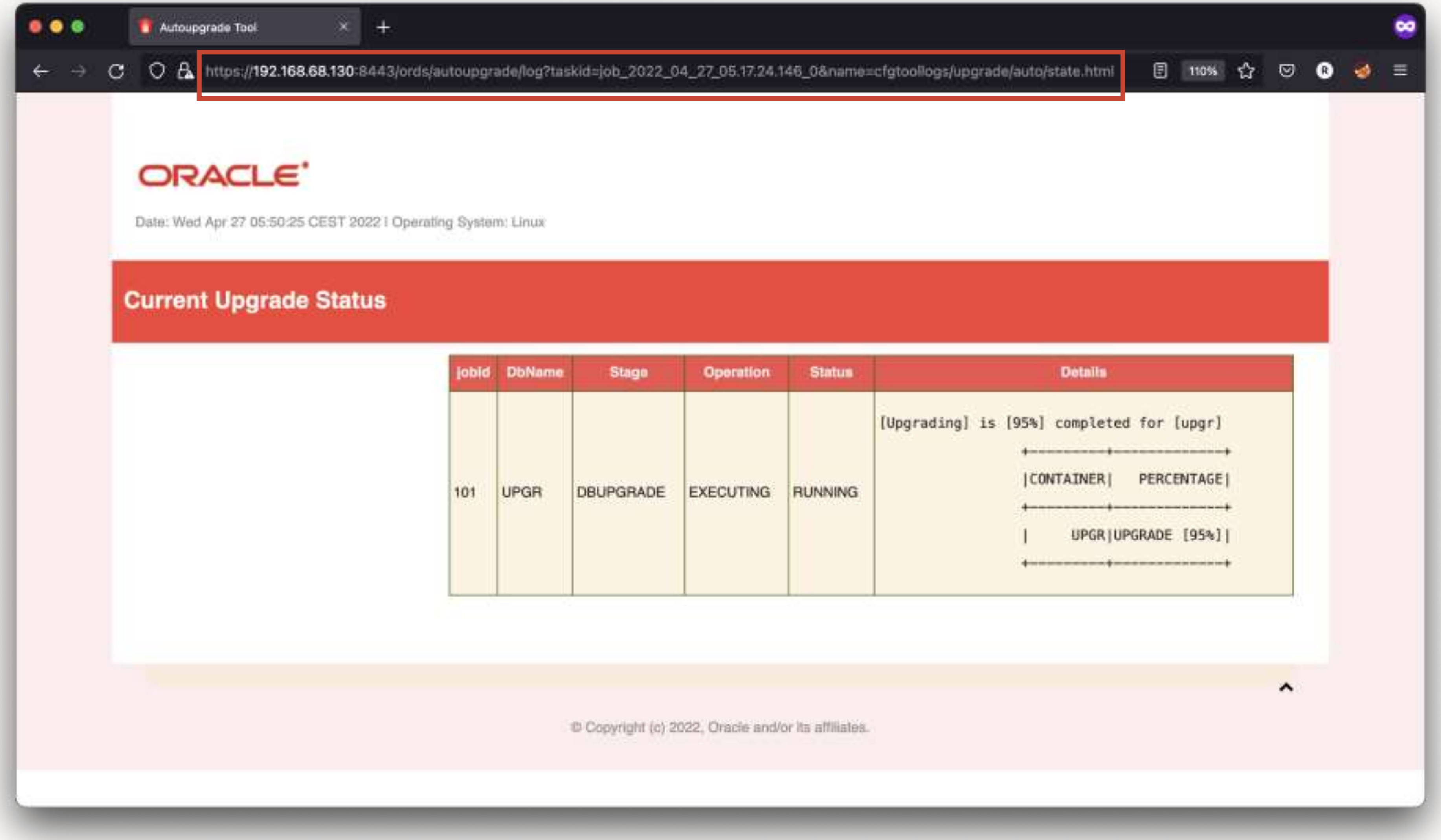
```
{
    "taskid": "job_2022_04_27_05.17.24.146_0",
    "status": "submitted",
    "message": "",
    "link": "https://localhost:8443/ords/autoupgrade/task?taskid=job_2022_04_27_05.17.24.146_0",
    "config": {
        "global": {
            "autoupg_log_dir": "/home/oracle/logs"
        },
        "jobs": [
            {
                "source_home": "/u01/app/oracle/product/11.2.0.4",
                "target_home": "/u01/app/oracle/product/19",
                "sid": "UPGR",
                "log_dir": "/home/oracle/logs",
                "restoration": "no"
            }
        ]
    }
}
```

REST API | List all files created by task

```
$ curl -k 'https://localhost:8443/ords/autoupgrade/log?taskid=job_2022_04_27_05.17.24.146_0'
```

```
{
    "logs": [
        ...,
        {
            "filename": "cfgtoollogs/upgrade/auto/status/status.html",
            "link":
"https://localhost:8443/ords/autoupgrade/log?taskid=job_2022_04_27_05.17.24.146_0&name=cfgtoollogs/upgrade/auto/status/status.html"
        },
        {
            "filename": "cfgtoollogs/upgrade/auto/status/status.log",
            "link":
"https://localhost:8443/ords/autoupgrade/log?taskid=job_2022_04_27_05.17.24.146_0&name=cfgtoollogs/upgrade/auto/status/status.log"
        },
        {
            "filename": "cfgtoollogs/upgrade/auto/status/progress.json",
            "link":
"https://localhost:8443/ords/autoupgrade/log?taskid=job_2022_04_27_05.17.24.146_0&name=cfgtoollogs/upgrade/auto/status/progress.json"
        }
    ]
}
```

REST API | State.html



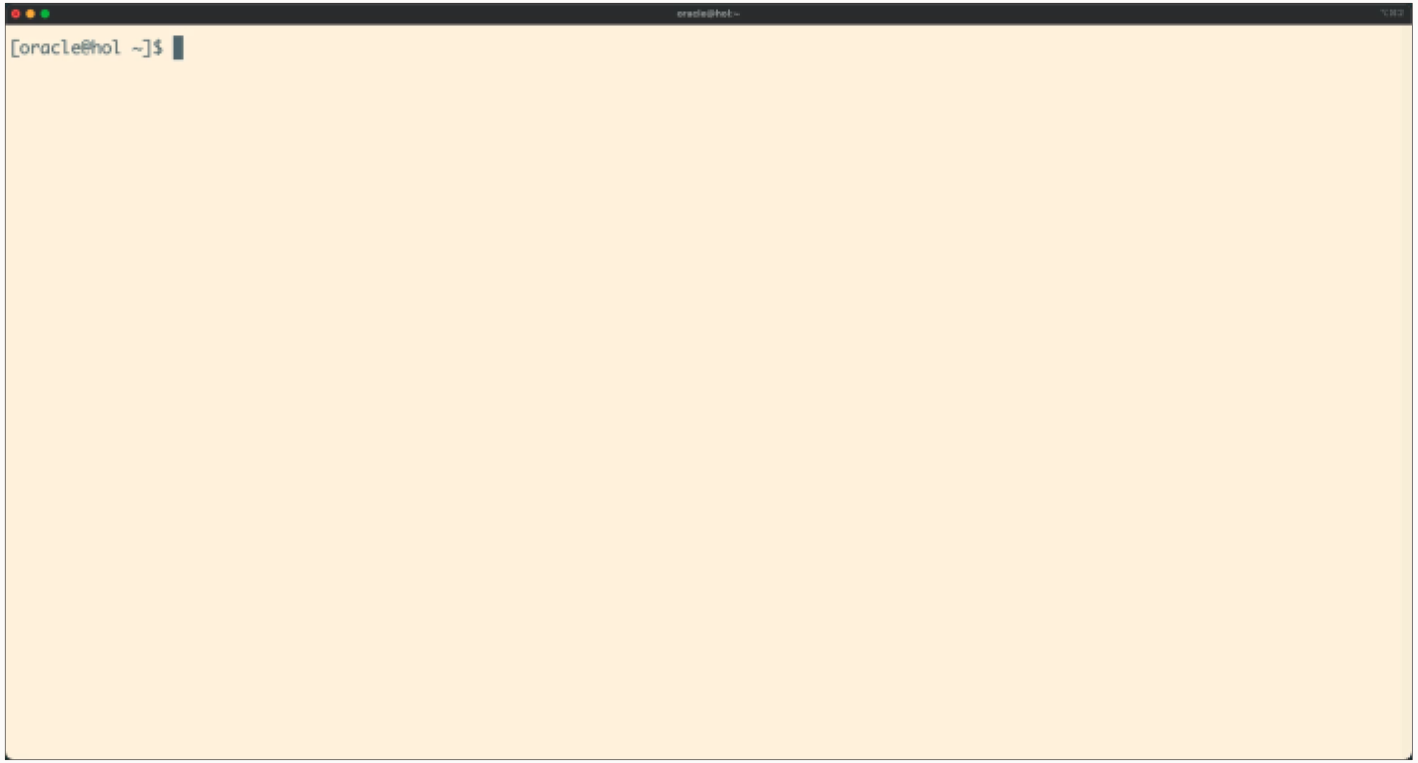
https://192.168.68.130:8443/ords/autoupgrade/log?taskid=job_2022_04_27_05.17.24.146_0&name=cfgtoollogs/upgrade/auto/state.html

Date: Wed Apr 27 05:50:25 CEST 2022 I Operating System: Linux

Current Upgrade Status

<table>
<tr><th>jobid</th><th>DbName</th><th>Stage</th><th>Operation</th><th>Status</th><th>Details</th></tr>
<tr><td>101</td><td>UPGR</td><td>DBUPGRADE</td><td>EXECUTING</td><td>RUNNING</td><td>[Upgrading] is [95%] completed for [upgr]
+---------+-------------+
|CONTAINER| PERCENTAGE|
+---------+-------------+
| UPGR|UPGRADE [95%]|
+---------+-------------+</td></tr>
</table>

REST API | Demo - Upgrade



```
[oracle@hol ~]$
```

Watch on YouTube

REST API | More details

- Official Documentation
- Blog post

AutoUpgrade 2.0

Clone non-CDB PDB

TDE Support

REST API

Usability

Standby

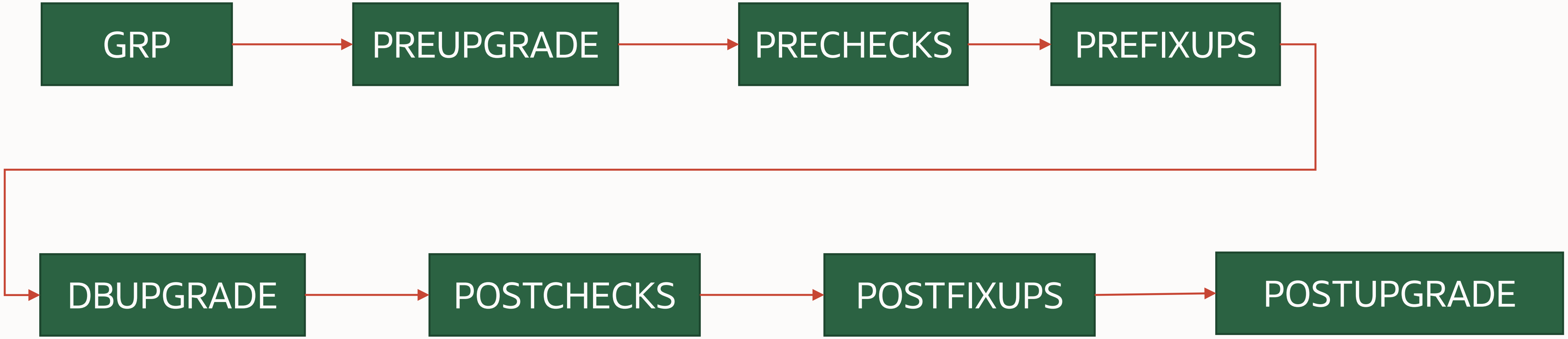
CDB RAC

Proactive Fixups result in
faster upgrades of CDBs with many PDBs

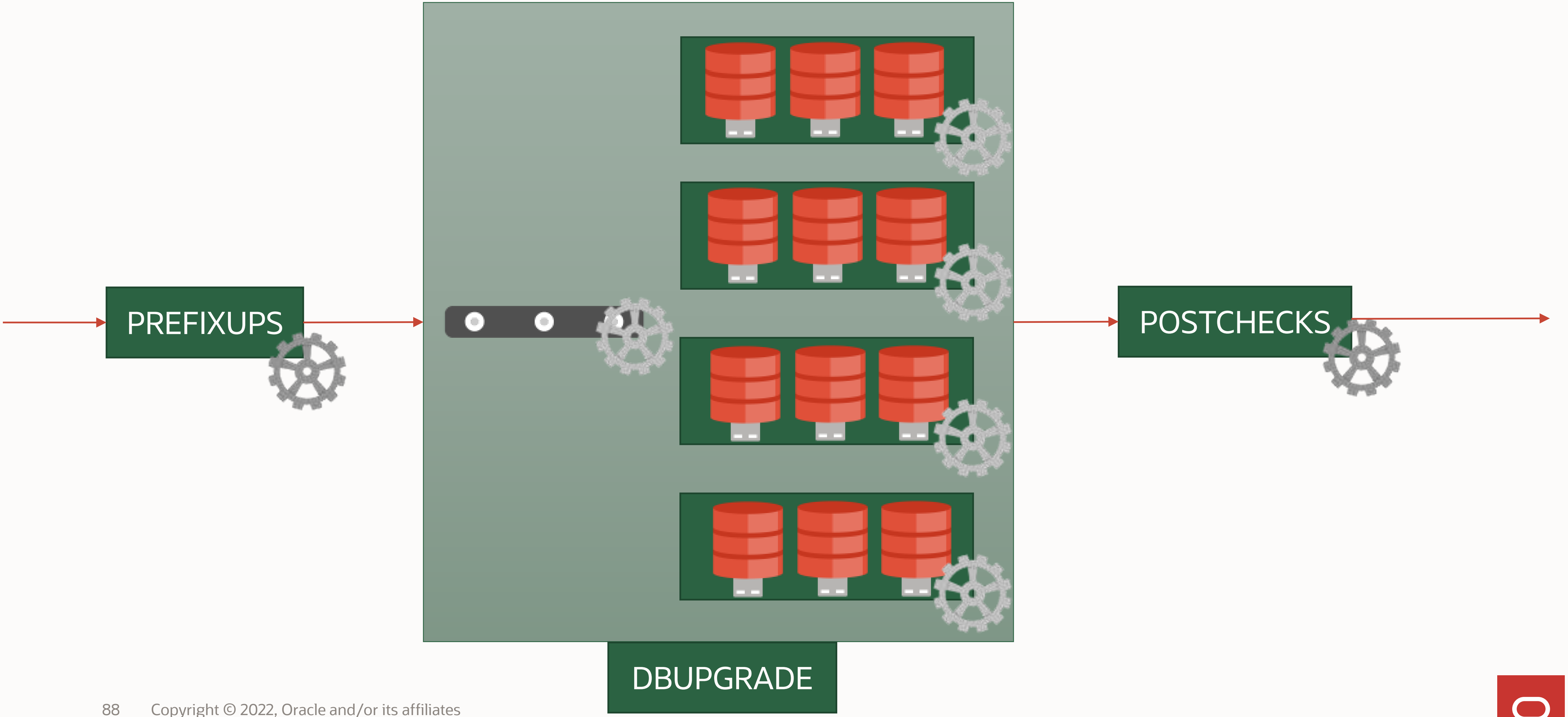
Proactive Fixups | What is it?

- Performance feature
- Changes only the order of the tasks of AutoUpgrade workflow
- Isolates errors in PDBs
- Valid for CDB upgrades only

Proactive Fixups | Classic Flow



Proactive Fixups | Classic Flow

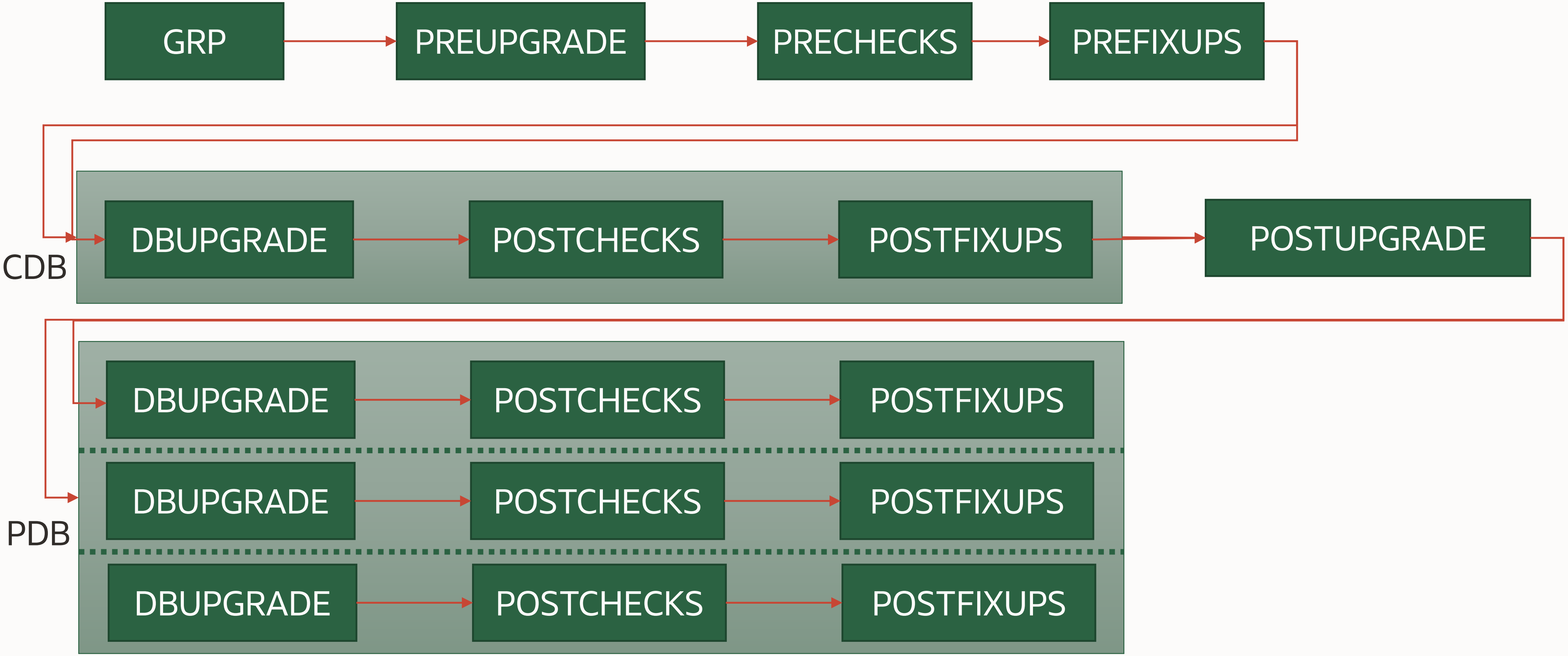


PREFIXUPS

DBUPGRADE

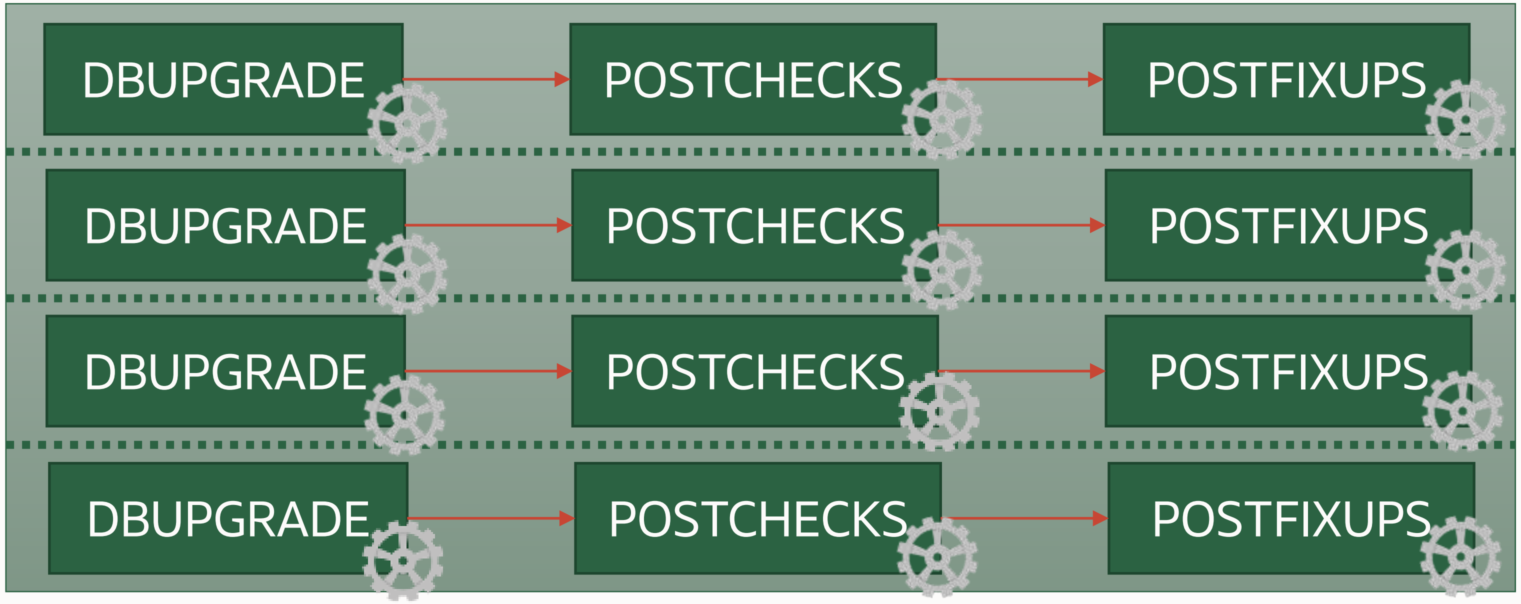
POSTCHECKS

Proactive Fixups | New Flow



Proactive Fixups | New Flow

PDBSUPG STAGE



DBUPGRADE → POSTCHECKS → POSTFIXUPS

DBUPGRADE → POSTCHECKS → POSTFIXUPS

DBUPGRADE → POSTCHECKS → POSTFIXUPS

DBUPGRADE → POSTCHECKS → POSTFIXUPS

Proactive Fixups | New Flow

PDBSUPG STAGE

```
Stage-Progress Per Container

        +--------+----------+--------+
        |Database|     Stage|Progress|
        +--------+----------+--------+
        |PDB$SEED| DBUPGRADE|    91 %|
        |   PDB01|POSTFIXUPS|     0 %|
        |   PDB02| DBUPGRADE|    20 %|
        |   PDB03|POSTFIXUPS|    25 %|
        |   PDB04|POSTFIXUPS|    75 %|
        |   PDB05|POSTFIXUPS|    10 %|
        |   PDB06| DBUPGRADE|     6 %|
        |   PDB07| DBUPGRADE|    91 %|
        |   PDB08| DBUPGRADE|    91 %|
        |   PDB09| DBUPGRADE|    91 %|
        +--------+----------+--------+
```

Proactive Fixups | Gain

4 PDBs + ROOT | 4 Cores

Default   Proactive Fixups

```
INFO     PREUPGRADE      <1 min    | INFO     PREUPGRADE      <1 min
INFO     PRECHECKS        1 min    | INFO     PRECHECKS        1 min
INFO     PREFIXUPS        8 min    | INFO     PREFIXUPS        7 min
INFO     DRAIN           <1 min    | INFO     DRAIN           <1 min
INFO     DBUPGRADE      143 min    | INFO     DBUPGRADE      130 min
INFO     POSTCHECKS       2 min    | INFO     POSTCHECKS      <1 min
INFO     POSTFIXUPS      34 min    | INFO     POSTFIXUPS      <1 min
INFO     POSTUPGRADE      1 min    | INFO     POSTUPGRADE      1 min
                                   |
                 TOTAL  179 min    |                  TOTAL  130 min
```

Proactive Fixups | Gain

16 PDBs + ROOT | 8 Cores | Defaults

Default **Proactive Fixups**

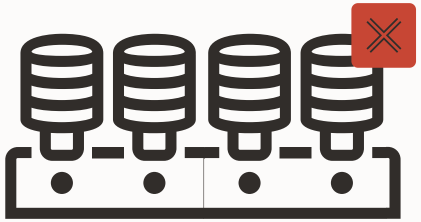
```
INFO      PREUPGRADE        <1  min      |  INFO      PREUPGRADE        <1  min
INFO      PRECHECKS         <1  min      |  INFO      PRECHECKS         <1  min
INFO      PREFIXUPS         <1  min      |  INFO      PREFIXUPS         14  min
INFO      DRAIN              2  min      |  INFO      DRAIN              2  min
INFO      DBUPGRADE        210  min      |  INFO      DBUPGRADE        195  min
INFO      POSTCHECKS         3  min      |  INFO      POSTCHECKS        <1  min
INFO      POSTFIXUPS        46  min      |  INFO      POSTFIXUPS        <1  min
INFO      POSTUPGRADE       <1  min      |  INFO      POSTUPGRADE        1  min

                    TOTAL  259  min      |                      TOTAL  195  min
```

The more PDBs, the greater the benefit

Proactive Fixups isolates each PDB
Errors in a PDB does not affect others

Proactive Fixups | Isolation

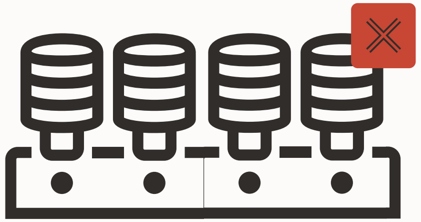
DEFAULT



Error in a PDB upgrade:

- Entire job halts
- Job can't complete

PROACTIVE FIXUPS



Error in a PDB upgrade:

- Other upgrades continue
- Job completes

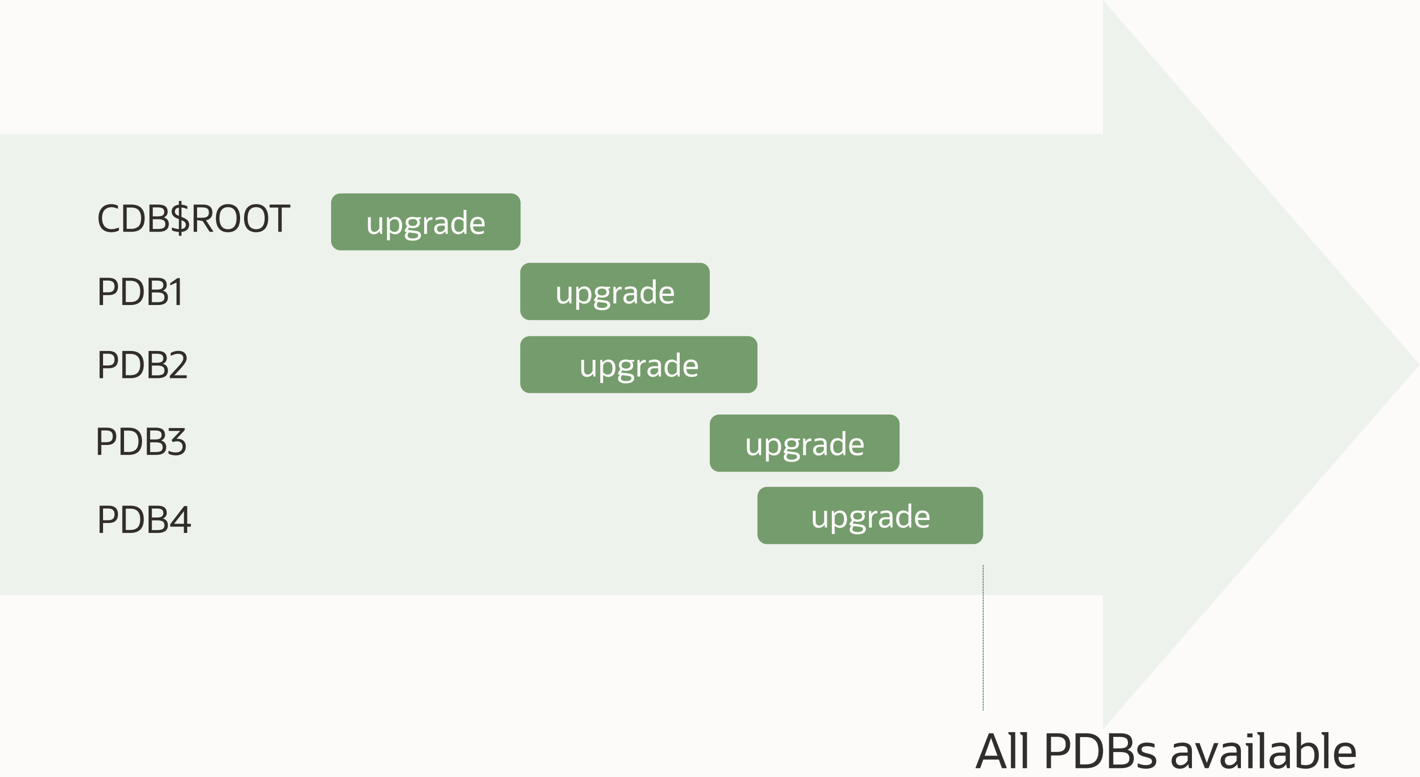
⚠

Restore point protects on CDB level only
Only entire CDB can be flashed back

Proactive Fixups | Availability

DEFAULT

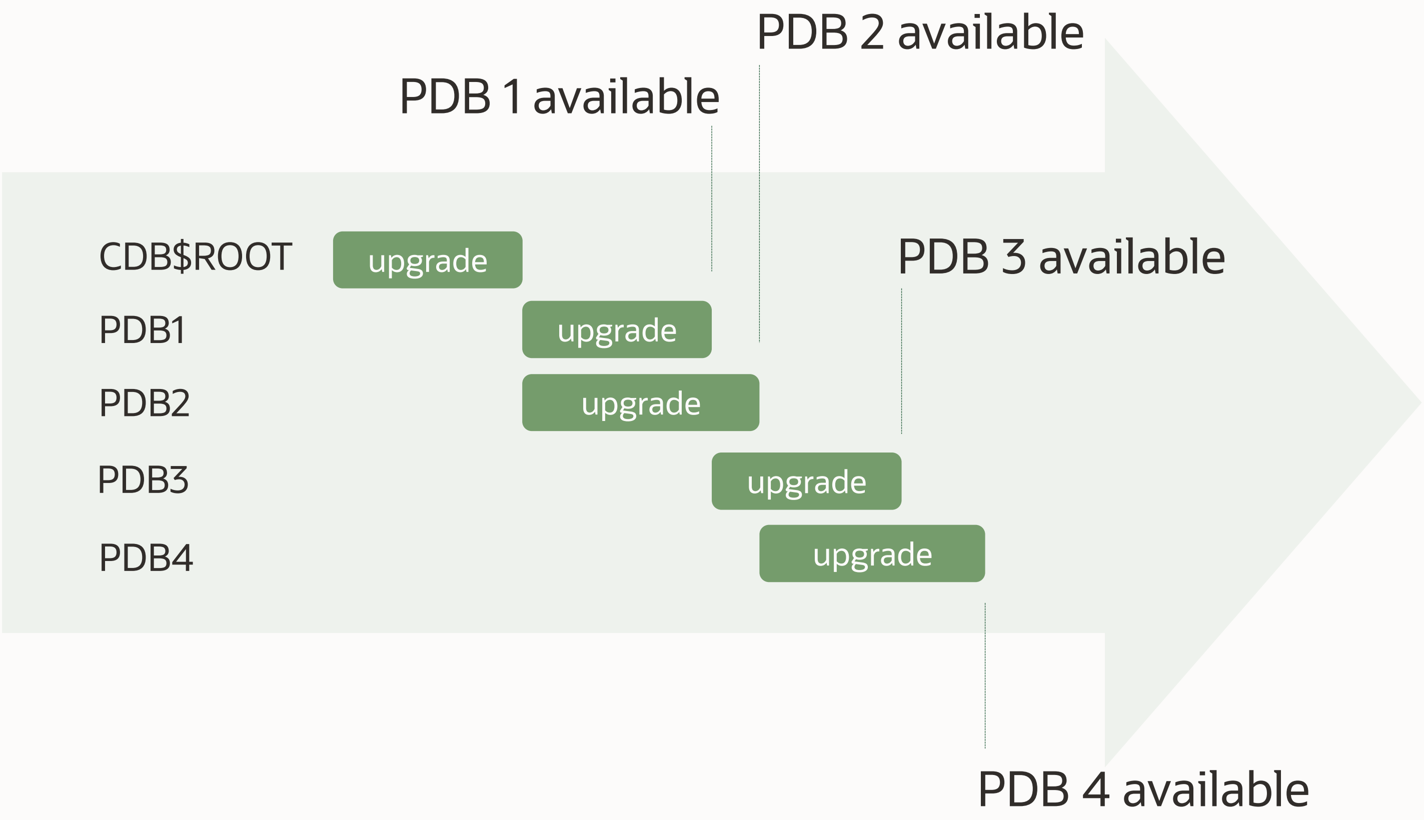
```
make_pdbs_available=false
```



Proactive Fixups | Availability

IMMEDIATELY AVAILABLE

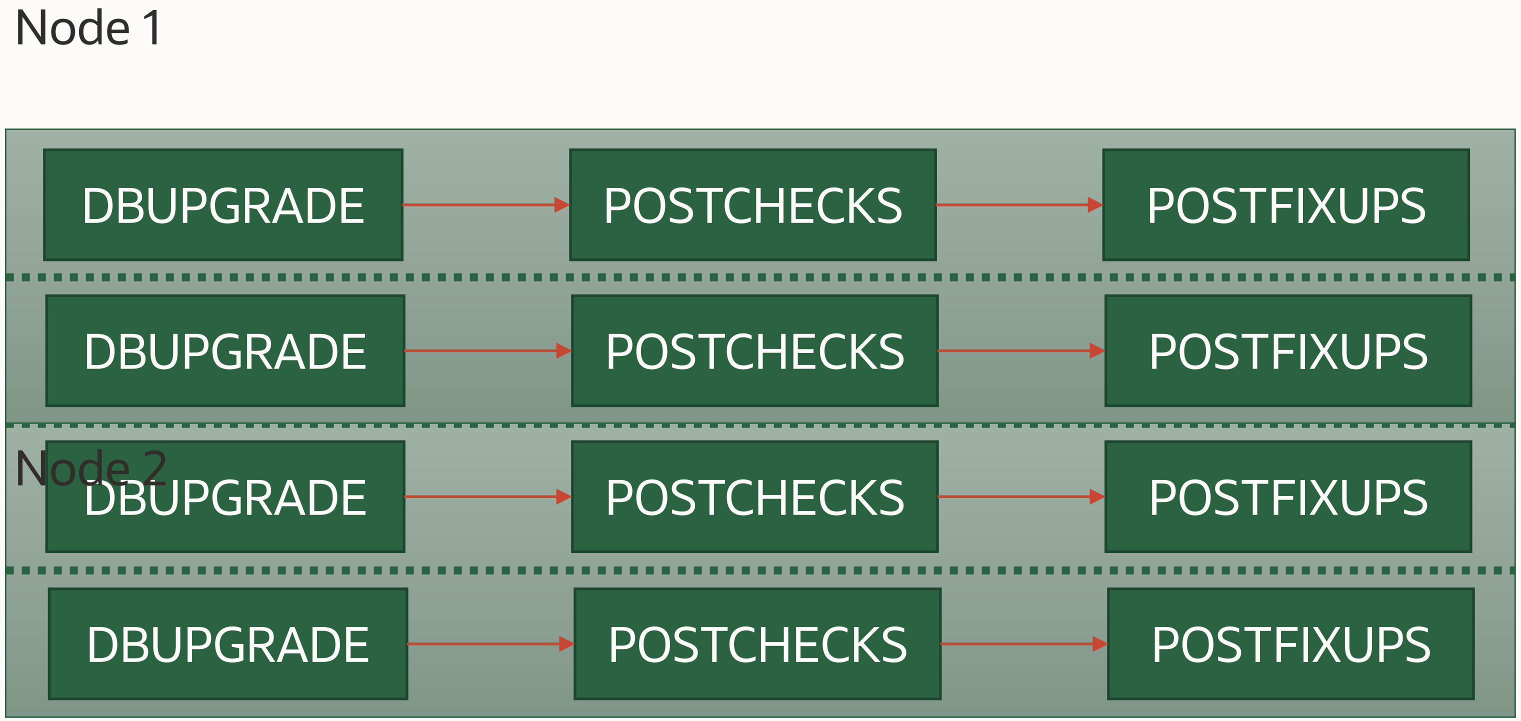
`make_pdbs_available=true`



Distributed upgrade uses all nodes in a cluster resulting in faster upgrades of CDBs

- Applies to RAC only
- Requires Proactive Fixups

Distributed Upgrade | Concept



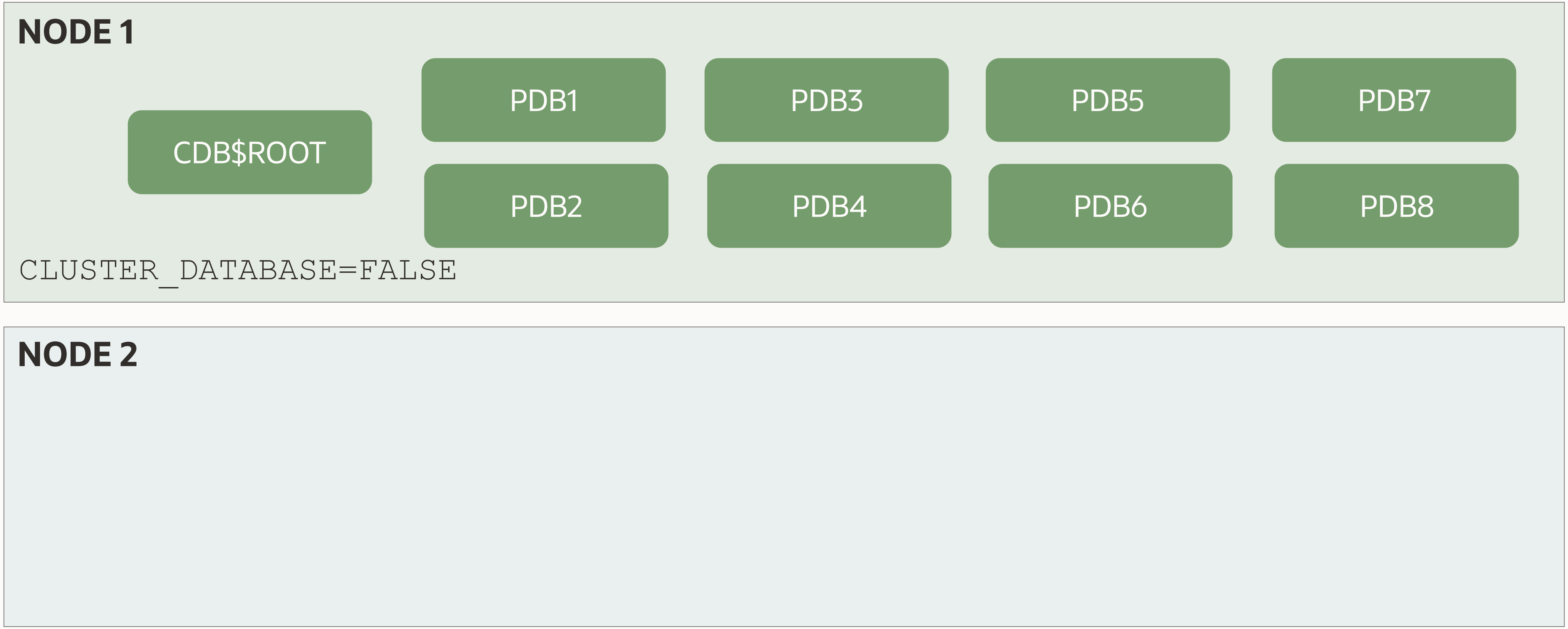
Distributed Upgrade | What is it?

- Performance feature
- Valid for CDB upgrades on RAC only
- First, CDB$ROOT upgrades on local node

  ```
  CLUSTER_DATABASE=FALSE
  ```
- Then, leverage resources on all nodes to upgrade PDBs

  ```
  CLUSTER_DATABASE=TRUE
  ```

Distributed Upgrade | Before



NODE 1

CDB$ROOT

PDB1

PDB2

PDB3

PDB4

PDB5

PDB6

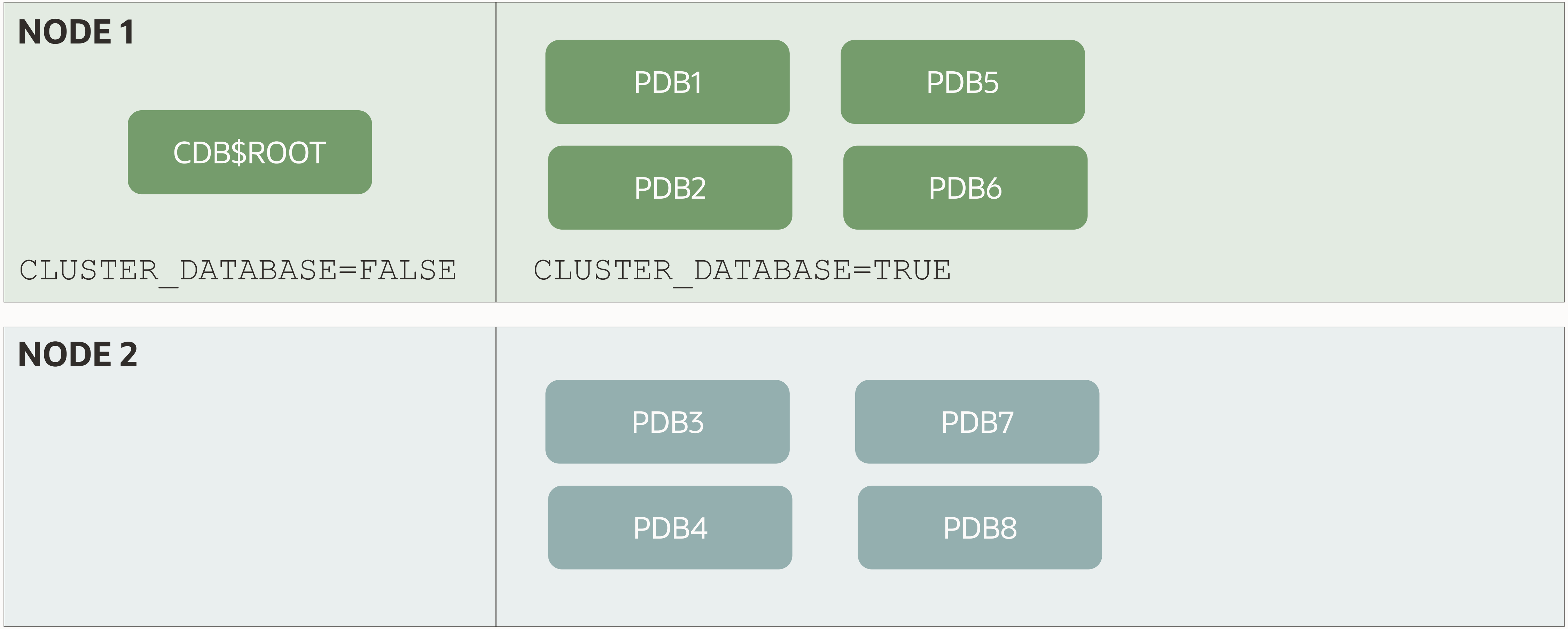
PDB7

PDB8

CLUSTER_DATABASE=FALSE

NODE 2

Distributed Upgrade | After



Distributed Upgrade | Console Message

```
Stage-Progress Per Container

       +--------+----------+--------+----+
       |Database|     Stage|Progress|Node|
       +--------+----------+--------+----+
       |PDB$SEED| DBUPGRADE|    91 %| au1|
       |   PDB01|POSTFIXUPS|     0 %| au1|
       |   PDB03|POSTFIXUPS|     0 %| au1|
       |   PDB04|POSTFIXUPS|     0 %| au1|
       |   PDB05|POSTFIXUPS|     0 %| au1|
       |   PDB02| DBUPGRADE|    91 %| au2|
       |   PDB06| DBUPGRADE|    91 %| au2|
       |   PDB07| DBUPGRADE|    91 %| au2|
       |   PDB08| DBUPGRADE|    91 %| au2|
       |   PDB09| DBUPGRADE|    91 %| au2|
       +--------+----------+--------+----+
```

Distributed Upgrade | Use

To enable distributed upgrade:

```
$ cat RACDB.cfg

global.autoupg_log_dir=/u01/app/oracle/cfgtoollogs/autoupgrade
upg1.log_dir=/u01/app/oracle/cfgtoollogs/autoupgrade/ RACDB
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid= RACDB
upg1.tune_setting=proactive_fixups=true,distributed_upgrade=true

$ java -jar autoupgrade.jar -config RACDB.cfg -mode deploy
```

Distributed Upgrade | Use

Under the hood

1. **AutoUpgrade creates a special config file**

2. AutoUpgrade spawns itself on all nodes

```
global.autoupg_log_dir=/u01/app/oracle/cfgtoollogs/...
# Databases section
# Database Batch 1
batch1.sid=RACDB1
batch1.source_home=/u01/app/oracle/product/12.2.0.1
batch1.target_home=/u01/app/oracle/product/19
batch1.upgrade_node=boston1
batch1.pdbs=PDB$SEED,PDB01,PDB03,PDB04,PDB05
batch1.tune_setting=DISTRIBUTED_UPGRADE=true,...
# Database Batch 2
batch2.sid=RACDB2
batch2.source_home=/u01/app/oracle/product/12.2.0.1
batch2.target_home=/u01/app/oracle/product/19
batch2.upgrade_node=boston2
batch2.pdbs=PDB02,PDB06,PDB07,PDB08,PDB09
batch2.tune_setting=DISTRIBUTED_UPGRADE=true,...
```

Distributed Upgrade | Use

Under the hood

1. AutoUpgrade creates a special config file

2. **AutoUpgrade spawns itself on all nodes**

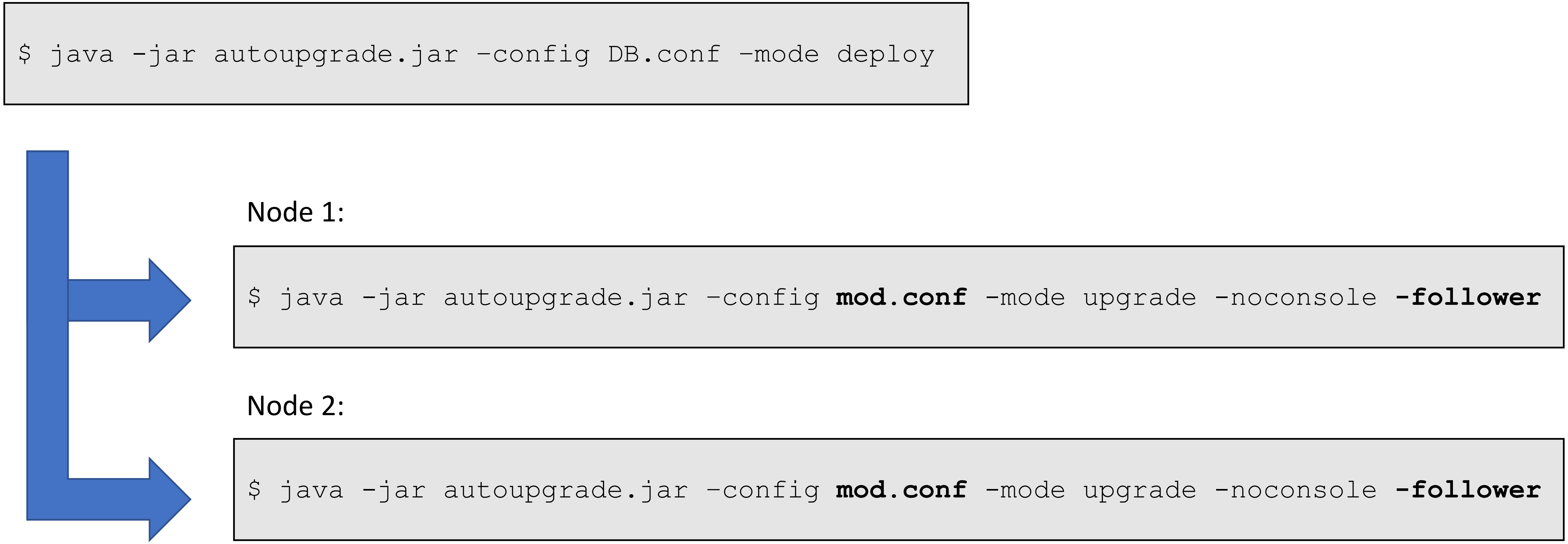
Node 1

```
$ java -jar autoupgrade.jar –config mod.conf \
     -mode upgrade -noconsole -follower
```

Node 2

```
$ java -jar autoupgrade.jar –config mod.conf \
     -mode upgrade -noconsole -follower
```

Distributed Upgrade | Architecture



```
$ java -jar autoupgrade.jar -config DB.conf -mode deploy
```

Node 1:

```
$ java -jar autoupgrade.jar -config mod.conf -mode upgrade -noconsole -follower
```

Node 2:

```
$ java -jar autoupgrade.jar -config mod.conf -mode upgrade -noconsole -follower
```

Distributed Upgrade | Architecture

Custom config file for 1 CDB + 10 PDBs upgrade

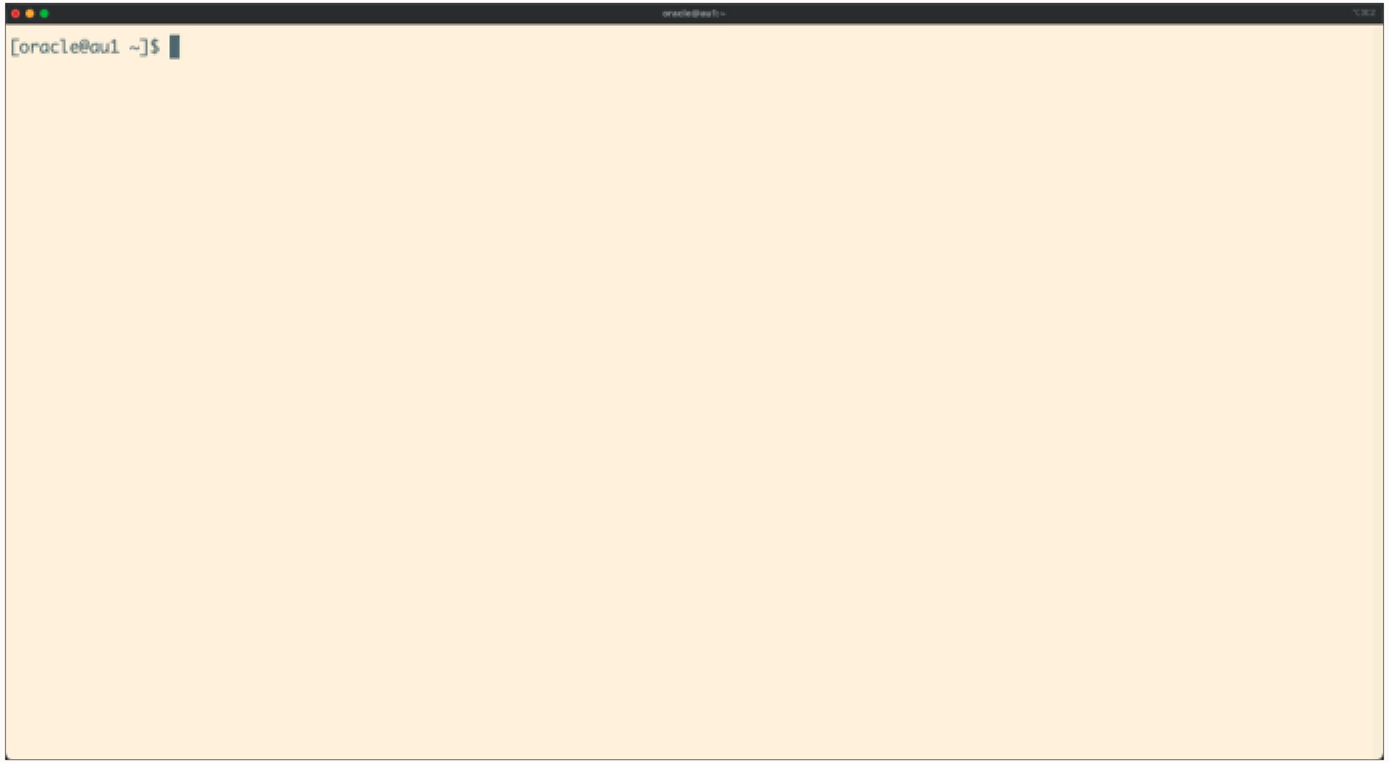
```
global.autoupg_log_dir=/u01/app/oracle/cfgtoollogs/autoupgrade/CDB12_iad1d7/au_logs
# Databases section
# Database Batch 1
batch1.sid=CDB121
batch1.source_home=/u01/app/oracle/product/12.2.0.1/dbhome_1
batch1.target_home=/u01/app/oracle/product/19/dbhome_2
batch1.upgrade_node=au1
batch1.pdbs=PDB$SEED,PDB01,PDB03,PDB04,PDB05
batch1.tune_setting=DISTRIBUTED_UPGRADE=true,PROACTIVE_FIXUPS=true
# Database Batch 2
batch2.sid=CDB122
batch2.source_home=/u01/app/oracle/product/12.2.0.1/dbhome_1
batch2.target_home=/u01/app/oracle/product/19/dbhome_2
batch2.upgrade_node=au2
batch2.pdbs=PDB02,PDB06,PDB07,PDB08,PDB09
batch2.tune_setting=DISTRIBUTED_UPGRADE=true,PROACTIVE_FIXUPS=true
```

Uses ORACLE_BASE

Node 1 (`au1`) receives 5 PDBs

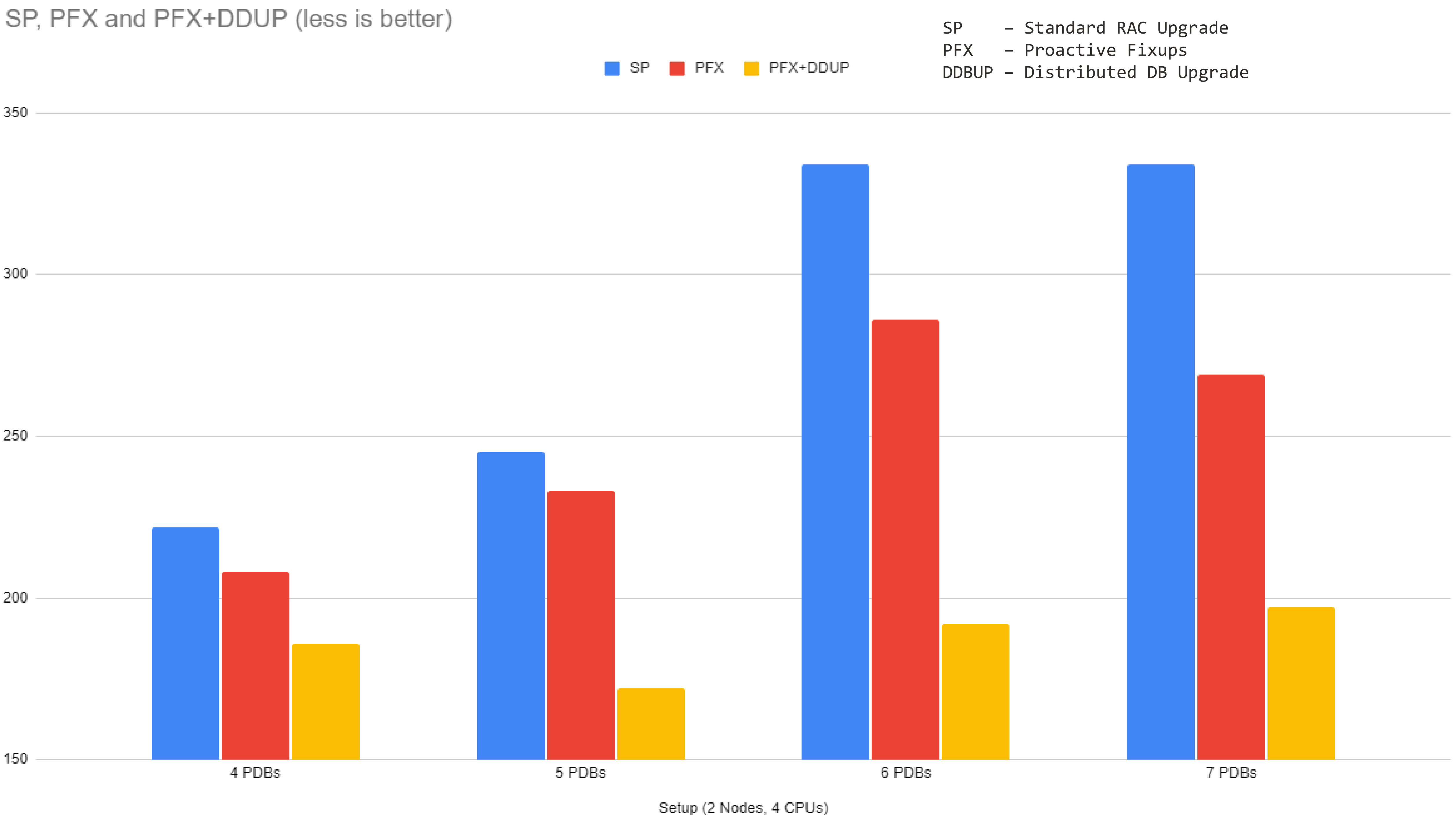
Node 2 (`au2`) also receives another 5 PDBs

Distributed Upgrade | Demo



```
[oracle@au1 ~]$ 
```

Watch on YouTube

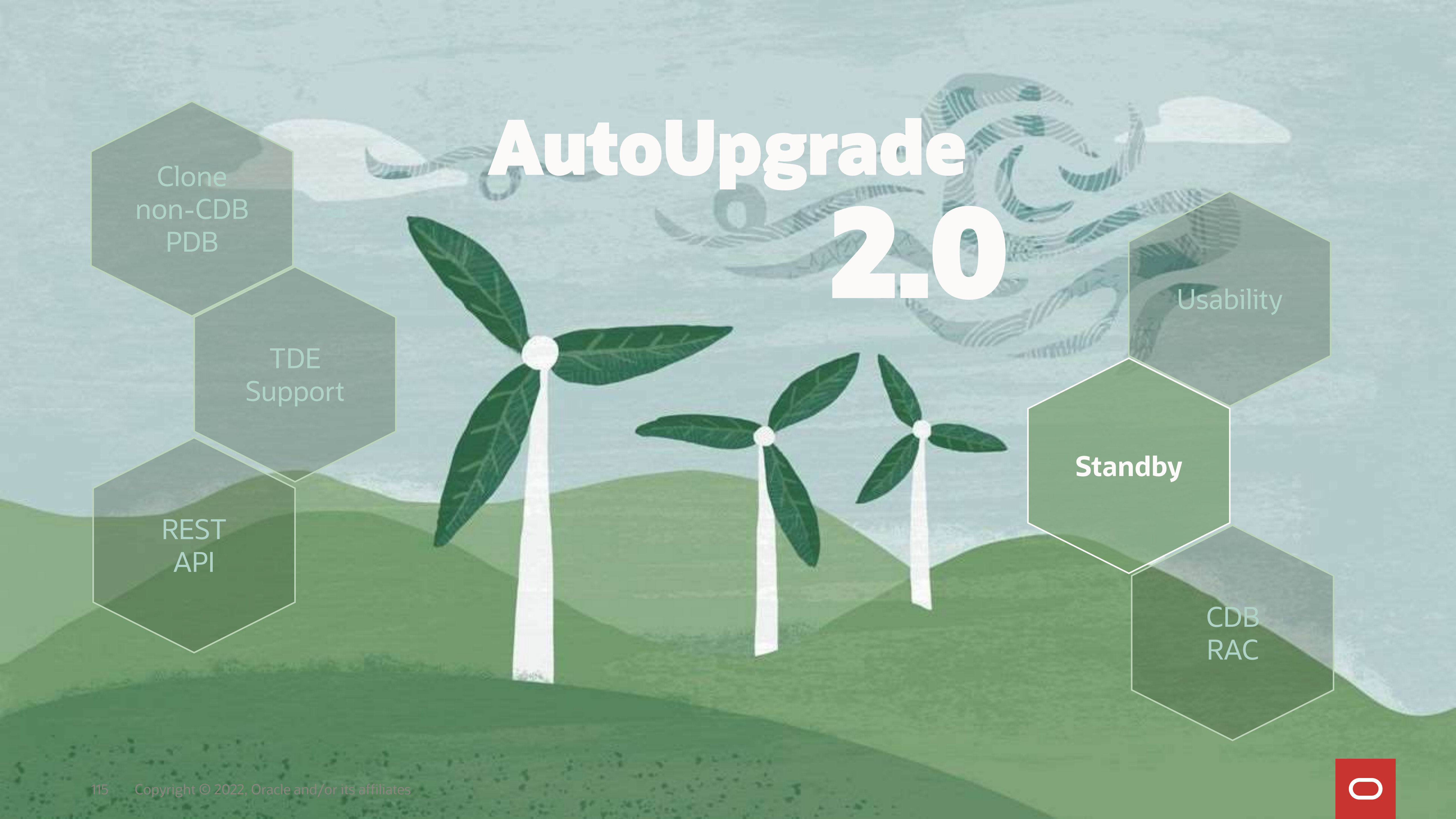


```
SP    – Standard RAC Upgrade
PFX   – Proactive Fixups
DDBUP – Distributed DB Upgrade
```

👍 By default, AutoUpgrade uses two nodes

You can control how many nodes are being used

```
upg1.tune_setting=distributed_upgrade=true,active_nodes_limit=n
```



AutoUpgrade 2.0

Clone non-CDB PDB

TDE Support

REST API

Usability

Standby

CDB RAC

⚠ Default for `defer_standby_log_shipping` changed from `YES` to `NO`

By default, AutoUpgrade no longer change your redo transport configuration during upgrade

- Redo log transport is no longer deferred

Redo is applied on standby databases continuously during upgrade

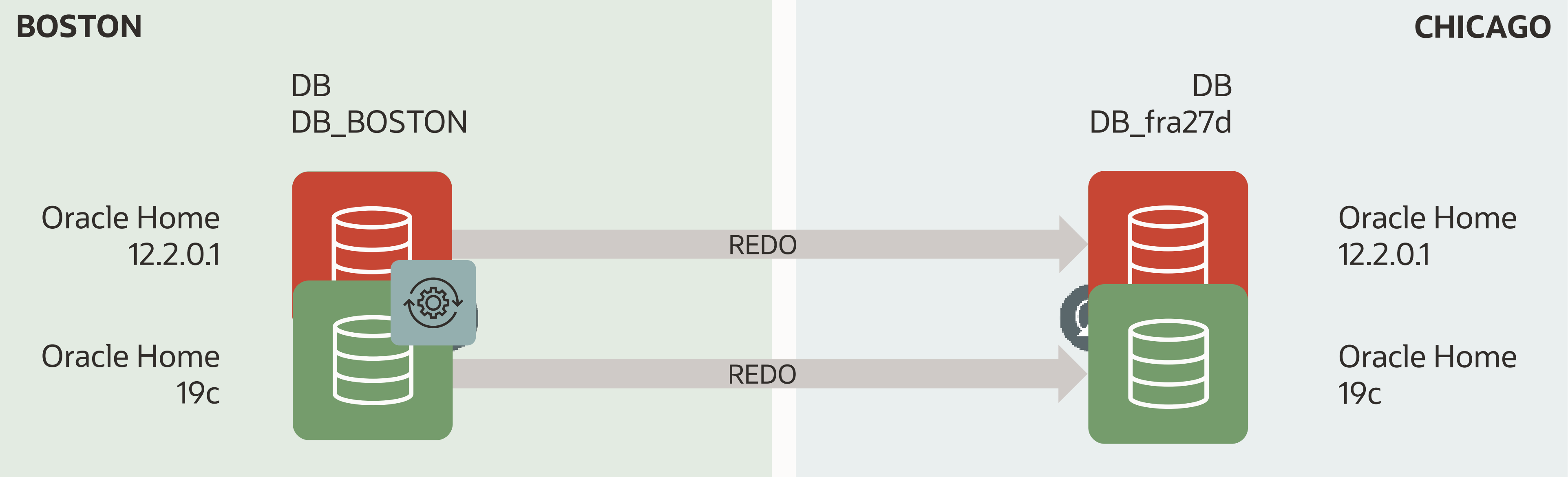
- Complies with MAA recommendations

Data Guard | Comparison

BEFORE `defer_standby_log_shipping=yes`	**AFTER** `defer_standby_log_shipping=no`
Maximum protection	Minimum downtime
Upgrade team recommendation	MAA recommendation
Redo log transport deferred	Redo log transport enabled
Redo apply stopped	Redo apply active
Protected by disconnected standby and guaranteed restore point	Protected by guaranteed restore point
	DEFAULT

When upgrading to Oracle Database 19c keep Data Guard broker running

Data Guard | Concept

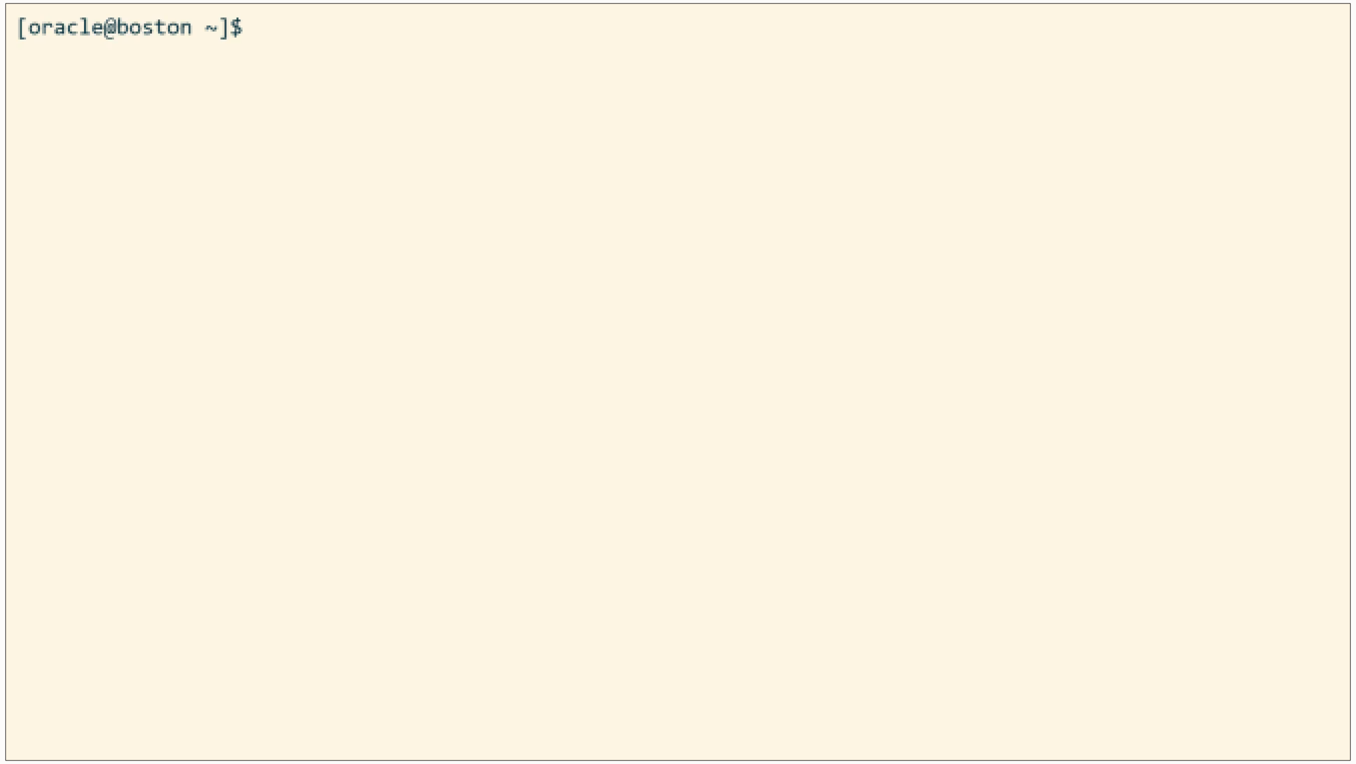


```
$ java -jar autoupgrade.jar ... -mode deploy
```

- Restore point
- Start in new Oracle Home
- Upgrade

```
$ srvctl stop database -d $ORACLE_UNQNAME

$ #switch to new Oracle Home
$ srvctl upgrade database -d $ORACLE_UNQNAME
$ srvctl start database -d $ORACLE_UNQNAME \
      -startoption mount
```

Data Guard | Demo



```
[oracle@boston ~]$
```

Watch on YouTube

Data Guard | Deferring Log Transport

A word of advice:
If `defer_standby_log_shipping=yes`,
all remote log archive destinations are deferred

A log archive destination can be used for:

- Standby databases
- GoldenGate downstream capture
- Per PDB Data Guard
- ZDLRA real-time redo transport

⚠ When a CDB with Data Guard receives a new PDB, special attention is needed

⚠

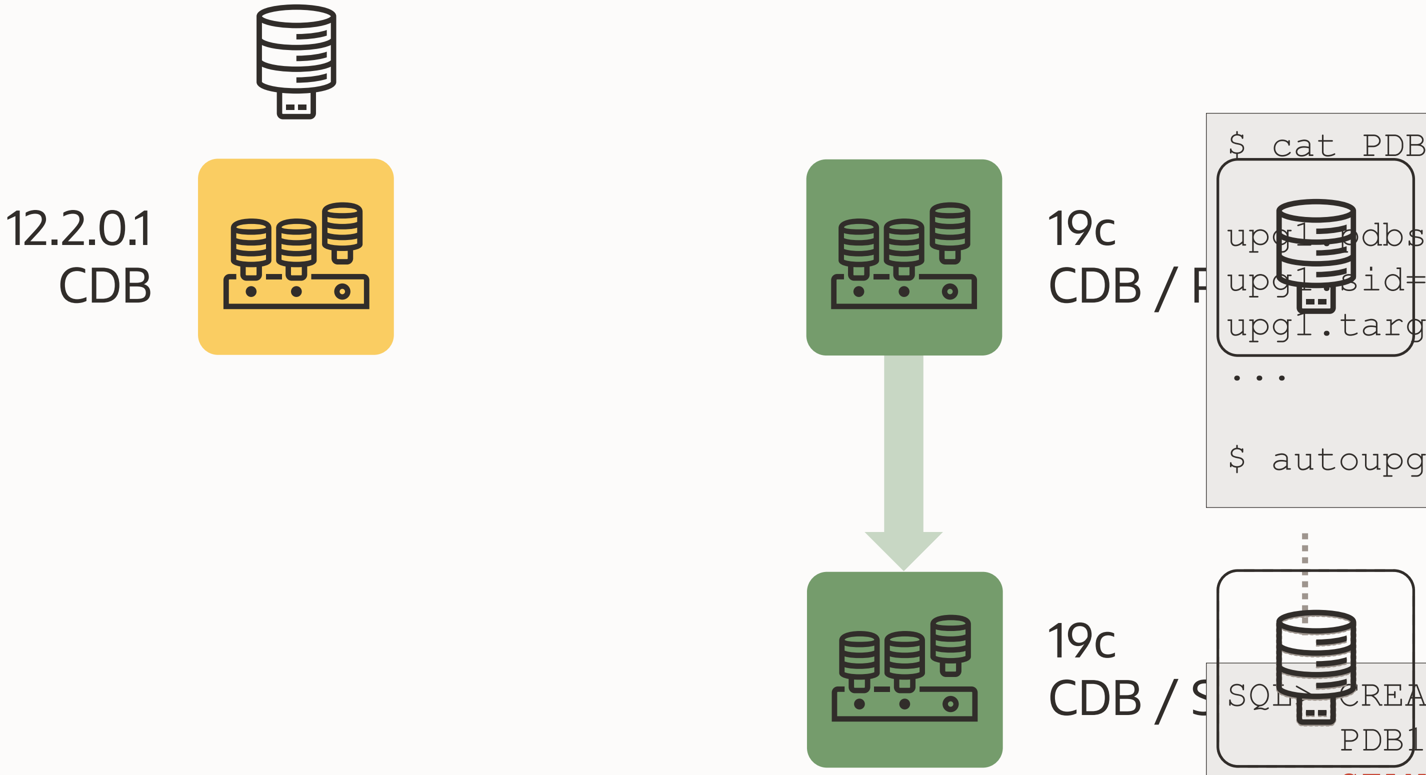
PDB data files must be in exact same location on primary and standby database, otherwise, MRP process will crash

👍

New AutoUpgrade config file parameter `manage_standbys_clause` defaults to `NONE`

AutoUpgrade will create PDBs using `STANDBYS=NONE` clause

Data Guard | Plug-in on standby



```
$ cat PDB1.cfg

upg1.pdbs=PDB1
upg1.sid=CDB122
upg1.target_sid
...

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

```
SQL> show pdbs

CON_NAME     OPEN MODE
PDB1         READ WRITE
```

```
SQL> CREATE PLU
     PDB1 USING
     STANDBYS=N
```

```
RMAN> restore pluggable database
      pdb1 from service .... ;

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

Data Guard | Plug-in on standby

For:

- Non-CDB to PDB conversion
- Unplug-plug upgrade

PDB is available on primary database only

- For a period, PDB is not protected by Data Guard
- Restore and recover data files to standby database
- Making Use Deferred PDB Recovery and the STANDBYS=NONE Feature with Oracle Multitenant (Doc ID 1916648.1)

You can re-use the PDB data files on the standby database, but special attention is needed

- Use AutoUpgrade config file parameter `manage_standbys_clause=all`

Data Guard | Re-use data files

To re-use data files and keep standby database intact

- Including making PDB available on standby database immediately

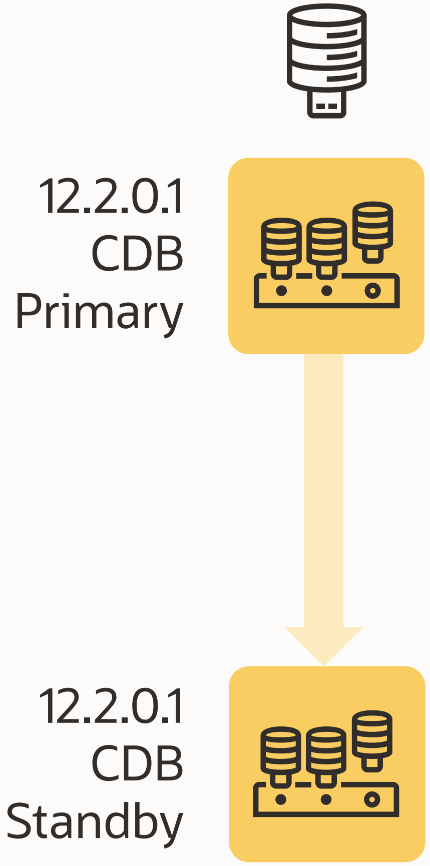
Special care is required

- Data files on standby database must be in exact same location as on primary database

For ASM and OMF:

- Reusing the Source Standby Database Files When Plugging a non-CDB as a PDB into the Primary Database of a Data Guard Configuration (Doc ID 2273304.1)

Data Guard | Re-use data files



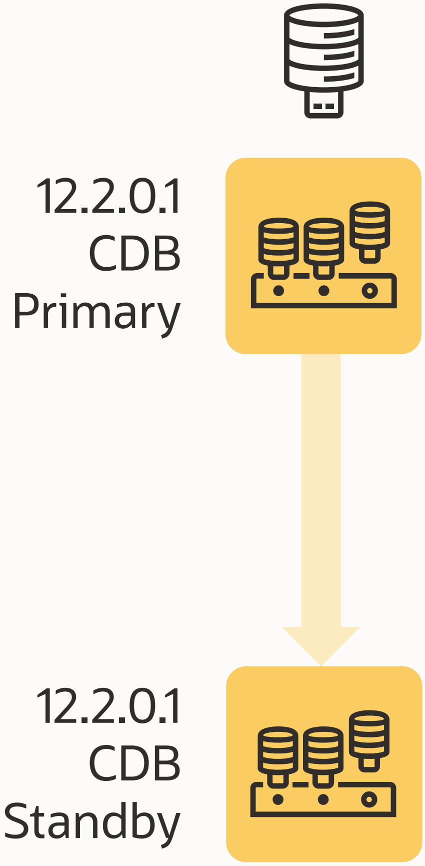
12.2.0.1
CDB
Primary

```
SQL> select name from v$datafile where con_id=3;

NAME
--------------------------------------------------------------------------------
+DATA/DB_BOSTON/DD934E8207292138E053E801000A8351/DATAFILE/system.269.1103046537
+DATA/DB_BOSTON/DD934E8207292138E053E801000A8351/DATAFILE/sysaux.270.1103046537
+DATA/DB_BOSTON/DD934E8207292138E053E801000A8351/DATAFILE/undotbs1.268.1103046537
+DATA/DB_BOSTON/DD934E8207292138E053E801000A8351/DATAFILE/users.273.1103046827
```

12.2.0.1
CDB
Standby

```
SQL> select name from v$datafile where con_id=3;

NAME
--------------------------------------------------------------------------------
+DATA/DB_FRA27D/DD934E8207292138E053E801000A8351/DATAFILE/system.265.1103050007
+DATA/DB_FRA27D/DD934E8207292138E053E801000A8351/DATAFILE/sysaux.266.1103050007
+DATA/DB_FRA27D/DD934E8207292138E053E801000A8351/DATAFILE/undotbs1.267.1103050009
+DATA/DB_FRA27D/DD934E8207292138E053E801000A8351/DATAFILE/users.269.1103050009
```

Data Guard | Re-use data files



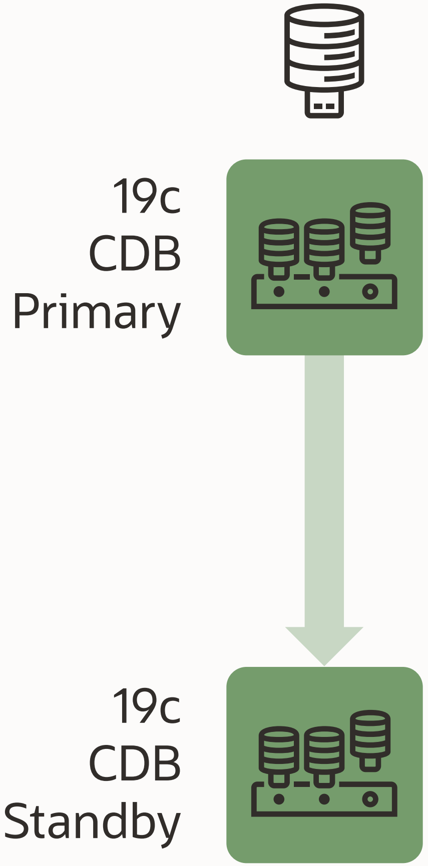
The manifest file contains

- File path on primary database only
- Not standby database

```
SQL> alter pluggable database PDB1 unplug into '/tmp/manifest_PDB1.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/DD934E8207292138E053E801000A8351/DATAFILE/users.273.1103046827</path>
```

Data Guard | Re-use data files



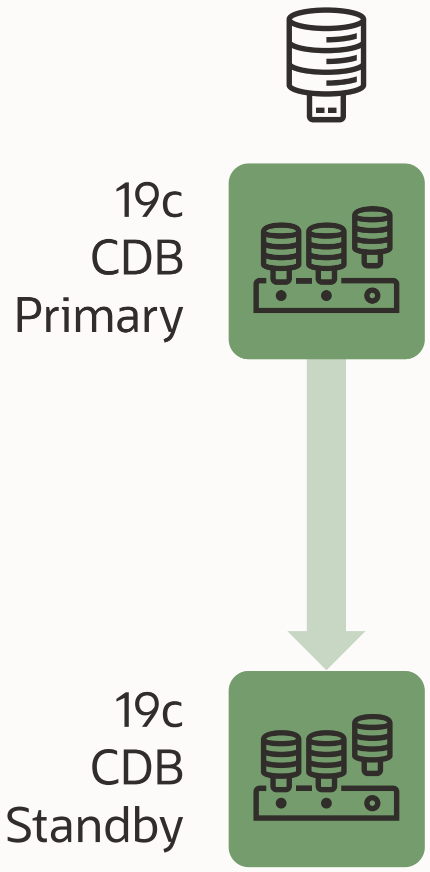
19c
CDB
Primary

19c
CDB
Standby

```
SQL> create pluggable database PDB1 using '/tmp/manifest_PDB1.xml' ... ;
```

- Manifest file lists the location of data files on primary
- No information about standby databases
- Standby database expect data files at the same location as on primary

Data Guard | Re-use data files

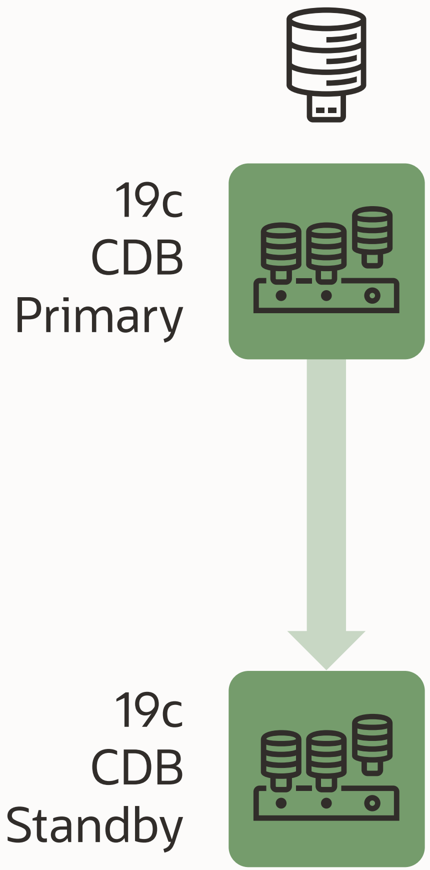


I will just move the files in ASM!

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

ASMCMD-8016: copy source '+DATA/DB_FRA27D/.../users.269.1103050009' and
target '+DATA/DB_BOSTON/.../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

Data Guard | Re-use data files



ASM alias to the rescue!

- On standby, create aliases for the primary data files

```
ASMCMD> alter diskgroup data add alias '...' for '...' ;
```

- Plug in PDB, standby will find aliases and find the real file locations
  From alert log

```
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_FRA27D/.../DATAFILE/users.269.1103050009'
```

Don't jeopardize your Data Guard!
Test the procedure and
verify your environment afterwards

AutoUpgrade 2.0

Clone non-CDB PDB

TDE Support

REST API

Usability

Standby

CDB RAC

New console commands

Usability | Console commands

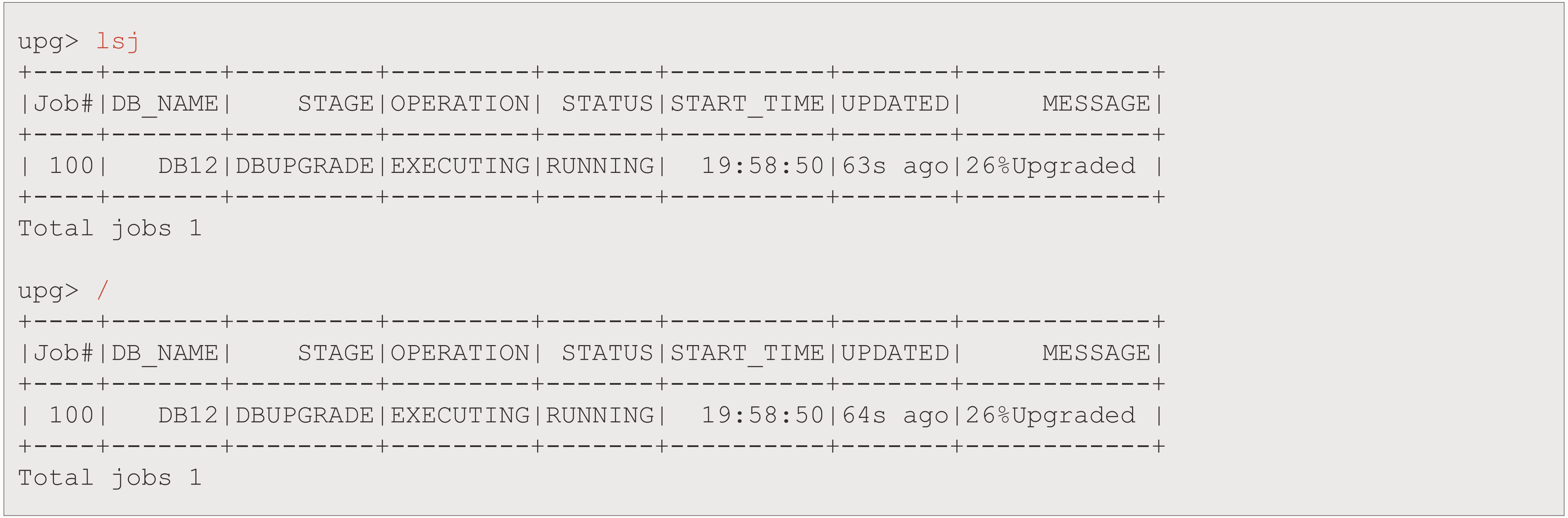
Repeat `lsj` or `status` command every *n* second

```
upg> lsj -a 10
+----+-------+---------+---------+-------+----------+-------+------------+
|Job#|DB_NAME|    STAGE|OPERATION| STATUS|START_TIME|UPDATED|     MESSAGE|
+----+-------+---------+---------+-------+----------+-------+------------+
| 100|   DB12|DBUPGRADE|EXECUTING|RUNNING|  19:58:50|70s ago|21%Upgraded |
+----+-------+---------+---------+-------+----------+-------+------------+
Total jobs 1

The command lsj is running every 10 seconds. PRESS ENTER TO EXIT
```

Pro tip: Repeat interval (`-a`) can be from 7 to 1200 seconds

Usability | Console commands

/ repeats the last command



```
upg> lsj
+----+-------+---------+---------+-------+----------+-------+------------+
|Job#|DB_NAME|    STAGE|OPERATION| STATUS|START_TIME|UPDATED|     MESSAGE|
+----+-------+---------+---------+-------+----------+-------+------------+
| 100|   DB12|DBUPGRADE|EXECUTING|RUNNING|  19:58:50|63s ago|26%Upgraded |
+----+-------+---------+---------+-------+----------+-------+------------+
Total jobs 1

upg> /
+----+-------+---------+---------+-------+----------+-------+------------+
|Job#|DB_NAME|    STAGE|OPERATION| STATUS|START_TIME|UPDATED|     MESSAGE|
+----+-------+---------+---------+-------+----------+-------+------------+
| 100|   DB12|DBUPGRADE|EXECUTING|RUNNING|  19:58:50|64s ago|26%Upgraded |
+----+-------+---------+---------+-------+----------+-------+------------+
Total jobs 1
```

Usability | Console commands

`h` or `hist` shows the last commands, use `/n` to run the command

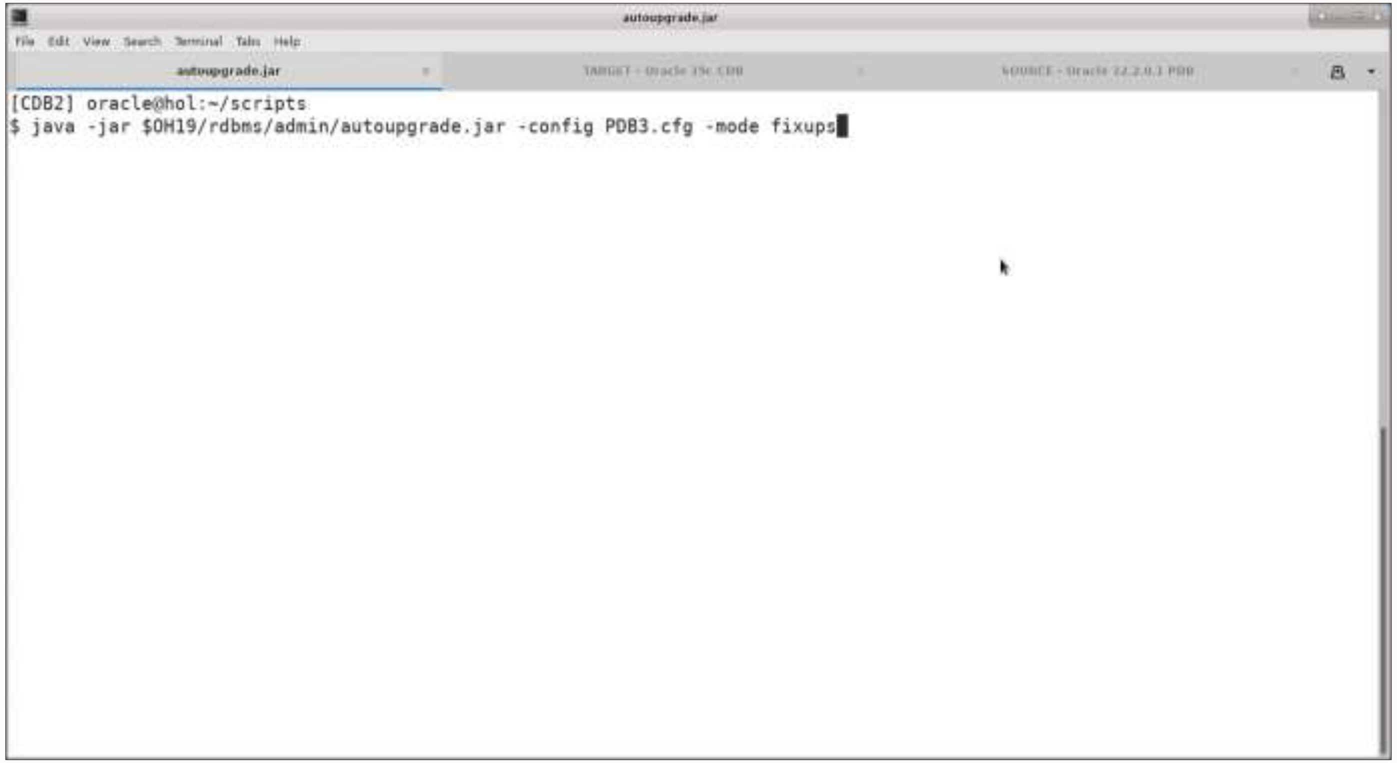
```
upg> h
 0  -> lsj -a 10
 1  -> lsj
 2  -> status -job 100 -a 10
 3  -> tasks
 4  -> help
upg> /1
+----+-------+---------+---------+-------+----------+-------+------------+
|Job#|DB_NAME|    STAGE|OPERATION| STATUS|START_TIME|UPDATED|     MESSAGE|
+----+-------+---------+---------+-------+----------+-------+------------+
| 100|   DB12|DBUPGRADE|EXECUTING|RUNNING|  19:58:50|87s ago|39%Upgraded |
+----+-------+---------+---------+-------+----------+-------+------------+
Total jobs 1
```

Usability | Console commands

`status -config` displays information about the current job

```
upg> status -config

Config

         Oracle SID        [DB12]
         Source Home       [/u01/app/oracle/product/12.2.0.1]
         Target Home       [/u01/app/oracle/product/19]
         Is RU Apply       [false]
         is CDB            [false]
         CPU Count         [4]
         Threads p/core    [2]
         Target CDB        [N/A]
         Custom Env        [N/A]
         DB Type           [STANDALONE]
         Tune Settings     [N/A]

System Parameters

         DB_UPGRADE_FATAL_ERRORS                  [ORA-00600,ORA-07445]
         SYSTEM_CHECKS_ORACLE_HOME_REQ_SPACE      [6g]
         HEARTBEAT_HEARTBEAT_SLEEP                [1]
```

Usability | Demo - 1



```
[CDB2] oracle@hol:~/scripts
$ java -jar $OH19/rdbms/admin/autoupgrade.jar -config PDB3.cfg -mode fixups
```

Watch on YouTube

Usability | Console commands

`fxlist` displays the fixups

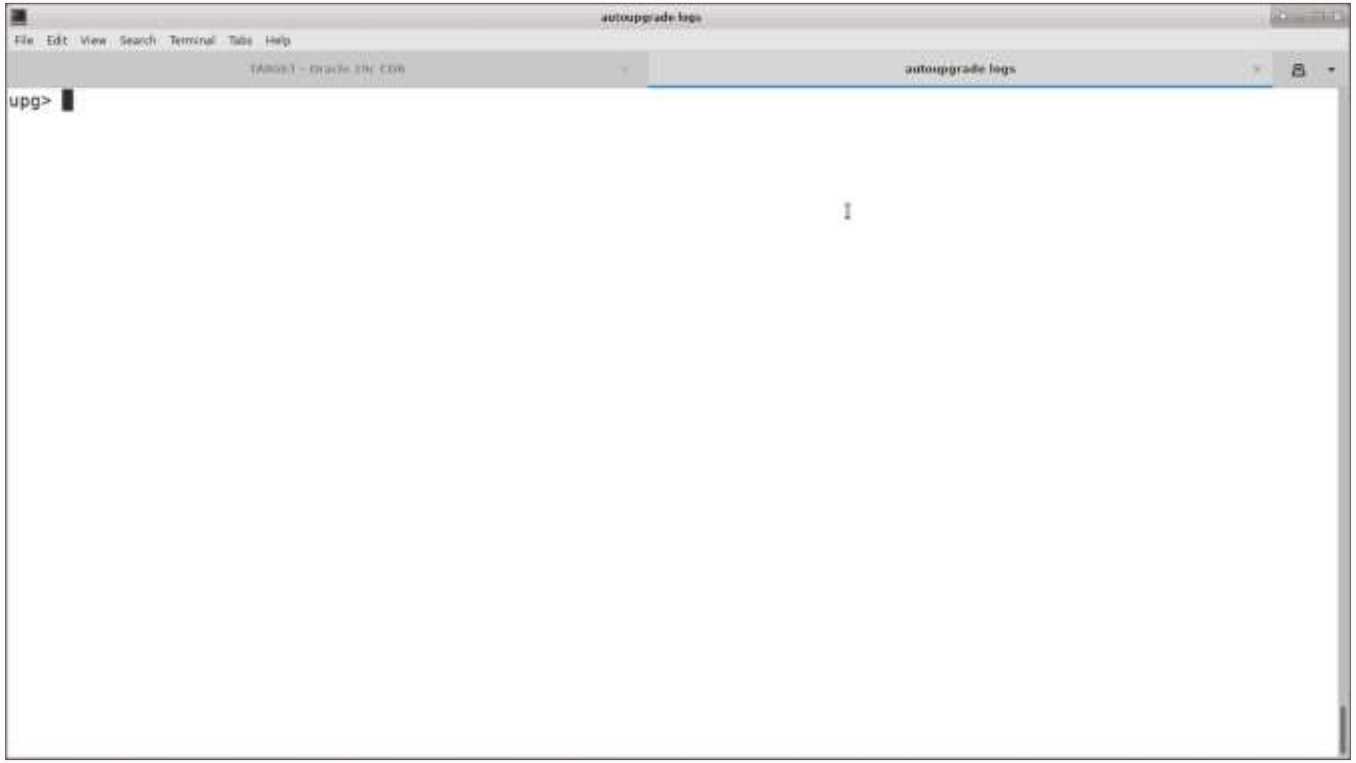
```
upg> fxlist -job 100

...

PostFixUps of Job 100

        Database DB12
                 +--------------------+---------+--------+
                 |          FixUp Name| Severity|Run Fix?|
                 +--------------------+---------+--------+
                 |OLD_TIME_ZONES_EXIST|  WARNING|     YES|
                 |     POST_DICTIONARY|RECOMMEND|     YES|
                 |          POST_UTLRP|RECOMMEND|     YES|
                 |  TIMESTAMP_MISMATCH|  WARNING|     YES|
                 +--------------------+---------+--------+
```

Usability | Console commands

Change fixup execution using `fxlist` (`yes`, `no`, `skip`)

```
upg> fxlist -job 100 -c DB12 alter OLD_TIME_ZONES_EXIST run no

...

PostFixUps of Job 100

         Database DB12
                 +--------------------+---------+--------+
                 |          FixUp Name| Severity|Run Fix?|
                 +--------------------+---------+--------+
                 |OLD_TIME_ZONES_EXIST|  WARNING|      NO|
                 |     POST_DICTIONARY|RECOMMEND|     YES|
                 |          POST_UTLRP|RECOMMEND|     YES|
                 |  TIMESTAMP_MISMATCH|  WARNING|     YES|
                 +--------------------+---------+--------+
```

Usability | Demo - 2



```
upg>
```

Watch on YouTube

Multitenant: Restrict resources consumed during recompilation phase (`utlrp`)

Usability | Recompilation

During multitenant upgrades AutoUpgrade:

- Recompiles in many PDBs at the same time (`CPU_COUNT`/3)
- Recompilation in a PDB runs with three threads
- Recompilation is very CPU intensive

Usability | Recompilation

```
$ sar -u 10 10

01:08:34 PM     CPU     %user     %nice   %system   %iowait    %steal     %idle
01:08:44 PM     all     95.09      0.00      2.18      0.01      0.00      2.72
01:08:54 PM     all     96.62      0.00      2.14      0.01      0.00      1.23
01:09:04 PM     all     96.75      0.00      2.30      0.03      0.00      0.92
01:09:14 PM     all     96.31      0.00      3.14      0.00      0.00      0.55
01:09:24 PM     all     95.72      0.03      4.07      0.00      0.00      0.18
01:09:34 PM     all     97.84      0.00      1.87      0.00      0.00      0.28
01:09:44 PM     all     97.12      0.00      2.06      0.01      0.00      0.81
01:09:54 PM     all     95.67      0.00      1.85      0.01      0.00      2.47
01:10:04 PM     all     95.39      0.00      2.95      0.01      0.00      1.65
01:10:14 PM     all     95.23      0.00      2.46      0.00      0.00      2.31
Average:        all     96.17      0.00      2.50      0.01      0.00      1.31
```

👍 Very nice when there is only one database on the host

⚠ Potential problem when there are more databases on the host

Usability | Recompilation

Two new *tune settings* to control recompilation

- `utlrp_pdb_in_parallel`
- `utlrp_threads_per_pdb`

Example:

```
upg1.tune_setting=utlrp_pdb_in_parallel=3,utlrp_threads_per_pdb=4
```

AutoUpgrade will recompile:

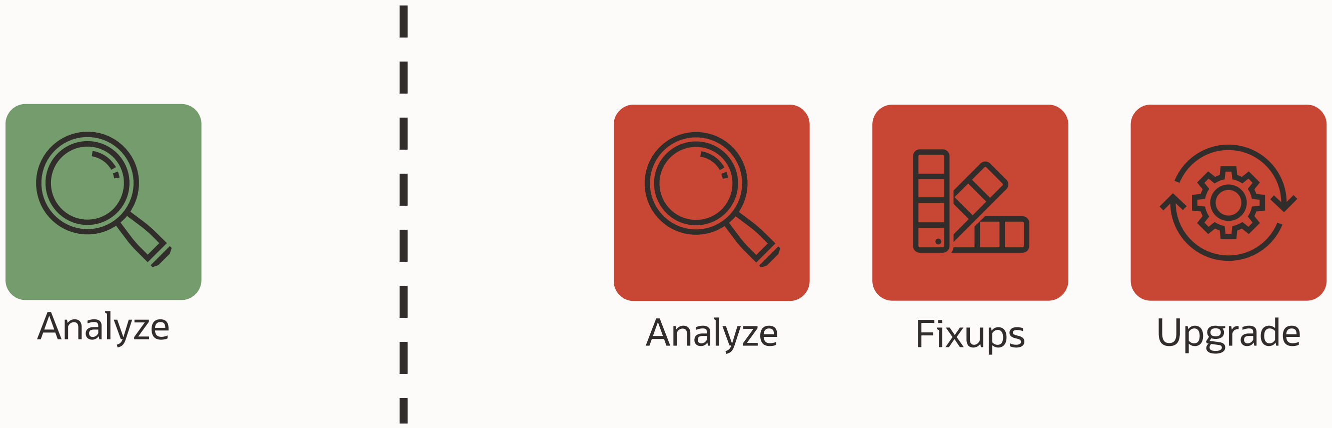
- Three PDBs at a time
- Use four threads per PDB

CPU consumption will use a maximum of 12 cores

Pro tip: Non-CDB and CDB$ROOT recompile with eight threads

👍 Fast Deploy reduces downtime

Usability | Fast Deploy



Analyze

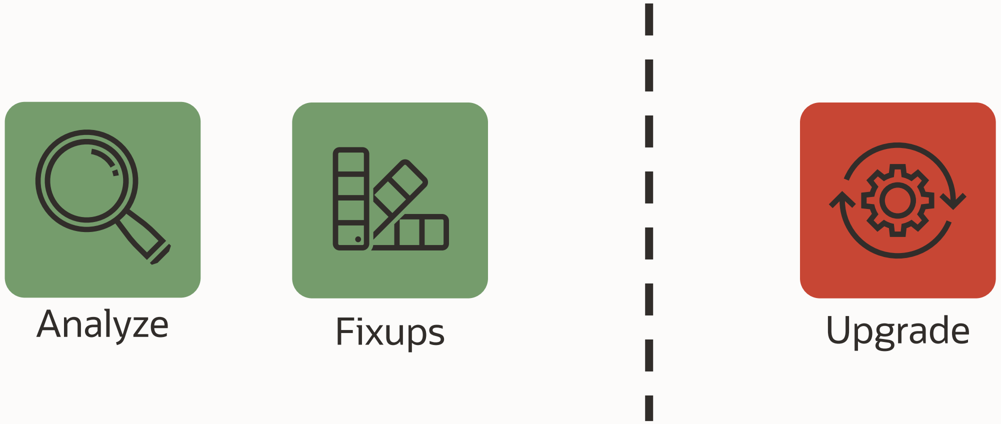
Analyze

Fixups

Upgrade

```
$ java -jar autoupgrade.jar -mode analyze

$ java -jar autoupgrade.jar -mode deploy
```

Usability | Fast Deploy



Analyze

Fixups

Upgrade

```
$ java -jar autoupgrade.jar -mode analyze
$ java -jar autoupgrade.jar -mode fixups

$ java -jar autoupgrade.jar -mode upgrade
```

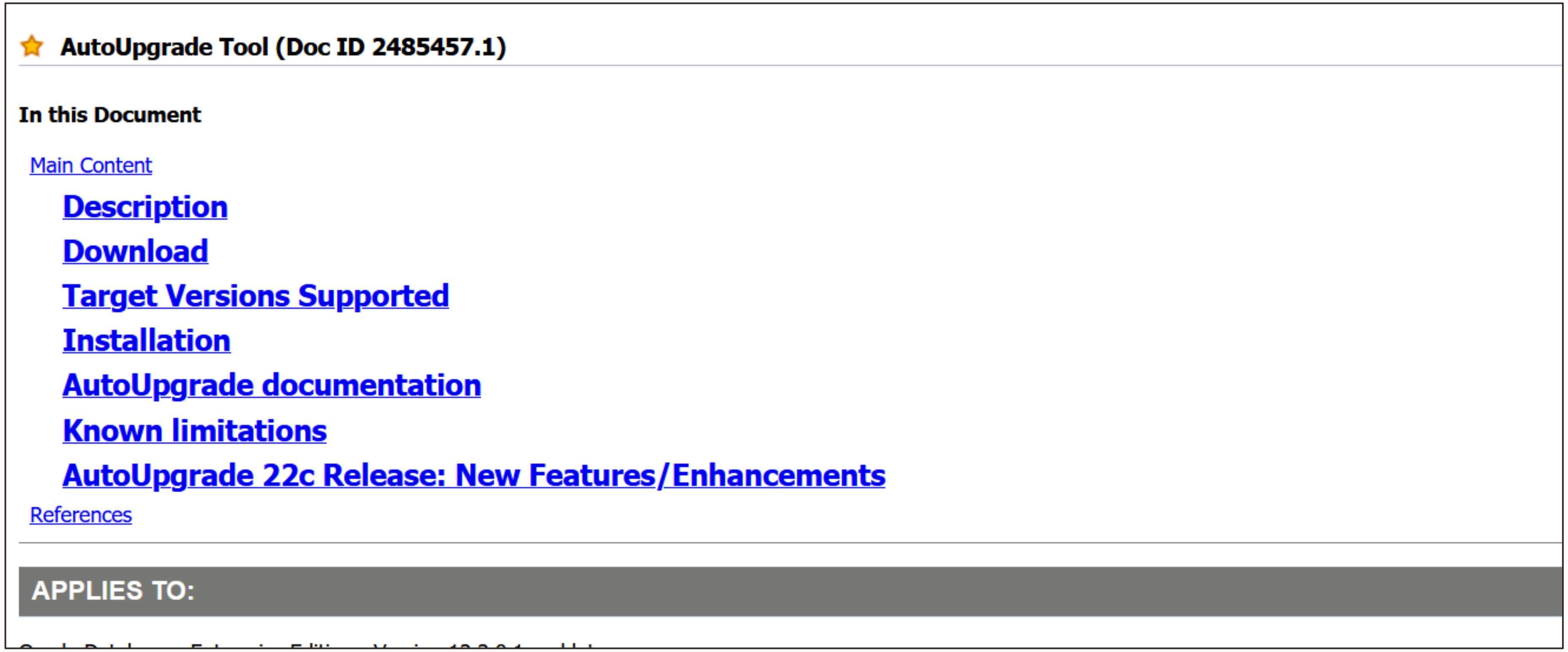
Usability | Fast Deploy

⚠ Between fixups and downtime there is a risk that new, undetected issues are introduced

Pro tip: Blog post with more details

AutoUpgrade 2.0

AutoUpgrade | Latest version



AutoUpgrade Tool (Doc ID 2485457.1)

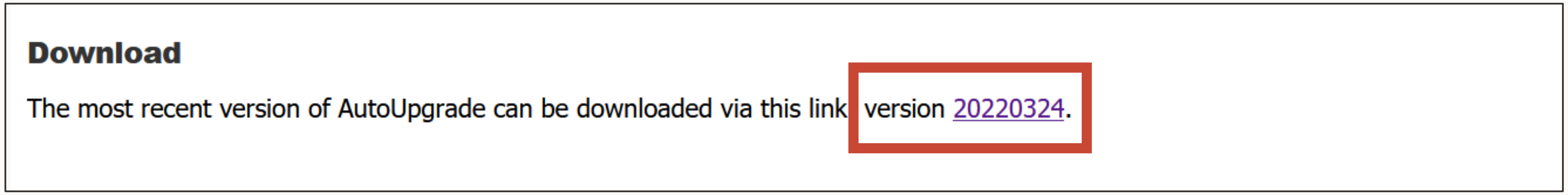
In this Document

Main Content

- Description
- Download
- Target Versions Supported
- Installation
- AutoUpgrade documentation
- Known limitations
- AutoUpgrade 22c Release: New Features/Enhancements

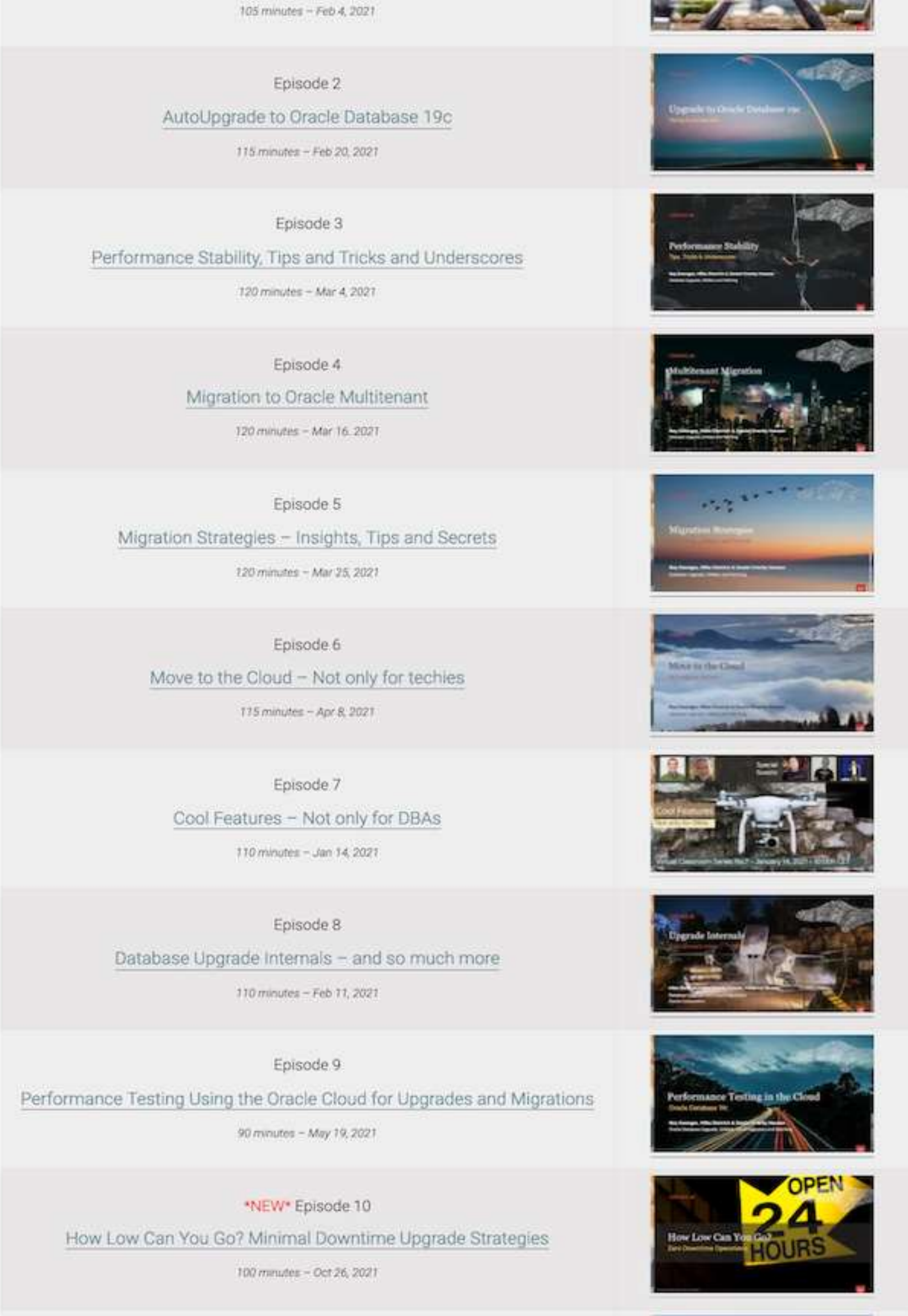
References

APPLIES TO:



Download

The most recent version of AutoUpgrade can be downloaded via this link version 20220324.



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

Episode 7
Cool Features – Not only for DBAs
110 minutes – Jan 14, 2021

Episode 8
Database Upgrade Internals – and so much more
110 minutes – Feb 11, 2021

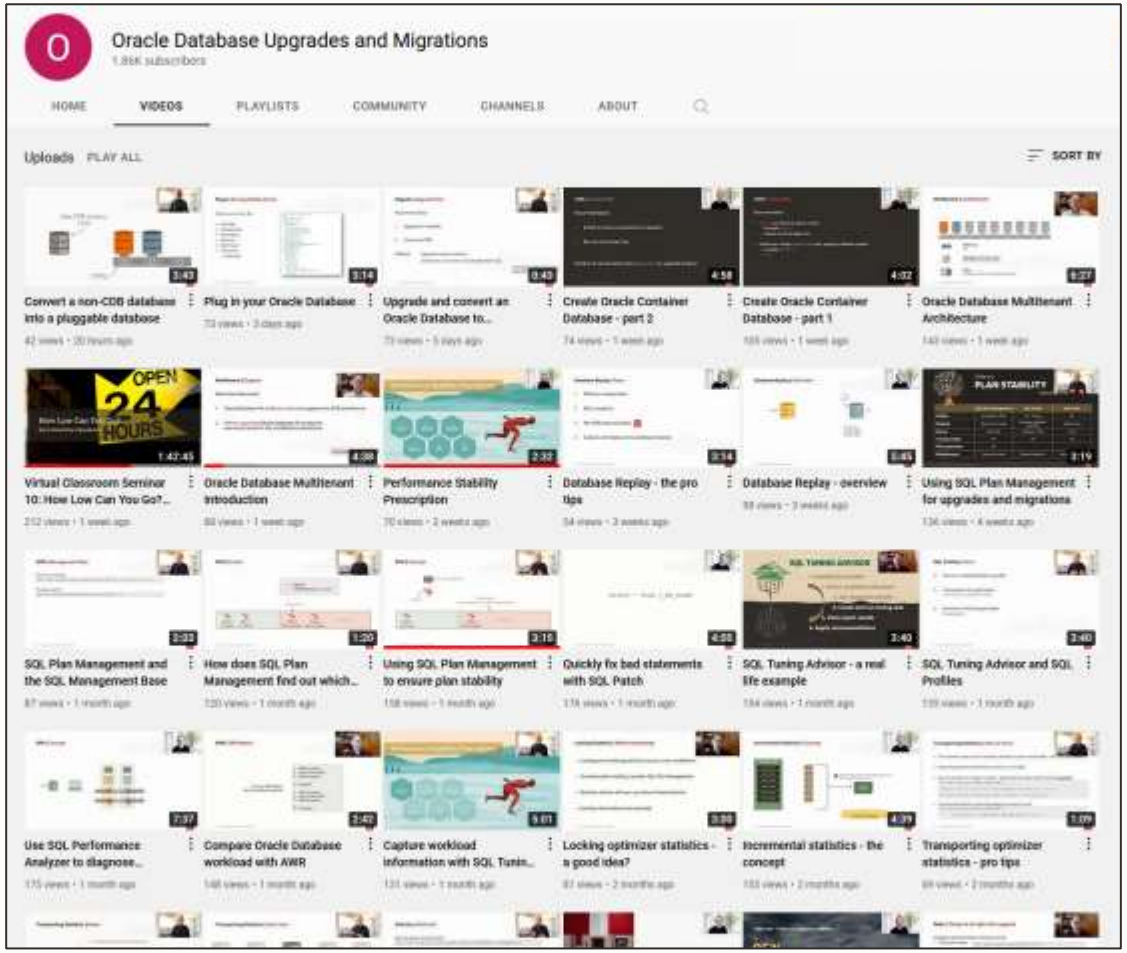
Episode 9
Performance Testing Using the Oracle Cloud for Upgrades and Migrations
90 minutes – May 19, 2021

NEW Episode 10
How Low Can You Go? Minimal Downtime Upgrade Strategies
100 minutes – Oct 26, 2021

Recorded Web Seminars

https://MikeDietrichDE.com/videos

YouTube | Oracle Database Upgrades and Migrations



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
80 views • 1 week ago

Performance Stability Prescription
70 views • 2 weeks ago

Database Replay - the pro tips
54 views • 2 weeks ago

Database Replay - overview
59 views • 3 weeks ago

Using SQL Plan Management for upgrades and migrations
134 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
178 views • 1 month ago

SQL Tuning Advisor - a real life example
184 views • 1 month ago

SQL Tuning Advisor and SQL Profiles
139 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?
87 views • 2 months ago

Incremental statistics - the concept
153 views • 2 months ago

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

Link

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

SCAN ME

THANK YOU

Visit our blogs:

https://MikeDietrichDE.com

https://DOHdatabase.com

https://www.dbarj.com.br/en

THANK YOU

Webinars:

https://MikeDietrichDE.com/videos

YouTube channel:

OracleDatabaseUpgradesandMigrations

THANK YOU

DATA PUMP

Best of Features and Use Cases

November/December 2022

THANK YOU