ORACLE

Hitchhiker's Guide for Upgrading to Oracle Database 19c

MakeIT 2023

Photo by Robert Katzki on Unsplash

Daniel Overby Hansen

Senior Principal Product Manager

Cloud Migration

dohdatabase

@dohdatabase

https://dohdatabase.com

AGENDA

10:00	**Introduction**
10:15	Upgrade to Oracle Database 19c
10:45	Break
11:00	Upgrade to Oracle Database 19c
12:00	Lunch
13:00	Ensure Performance Stability
14:00	Break
14:15	Hands-On Lab
16:00	End

Hands-On Lab | Create Your Lab

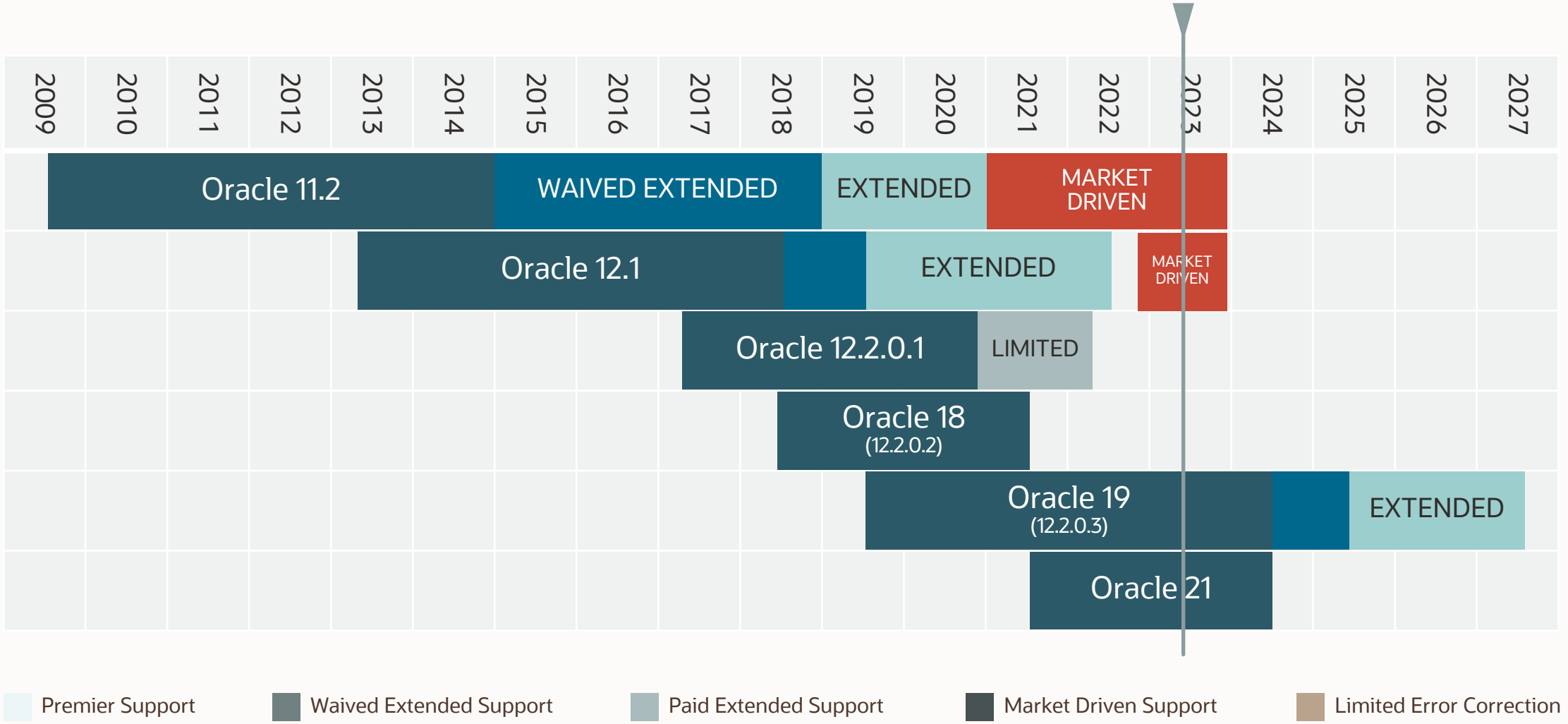
https://tinyurl.com/makeit2023hol

Workshop Code: 6868-JISO-UETI-ISAQ

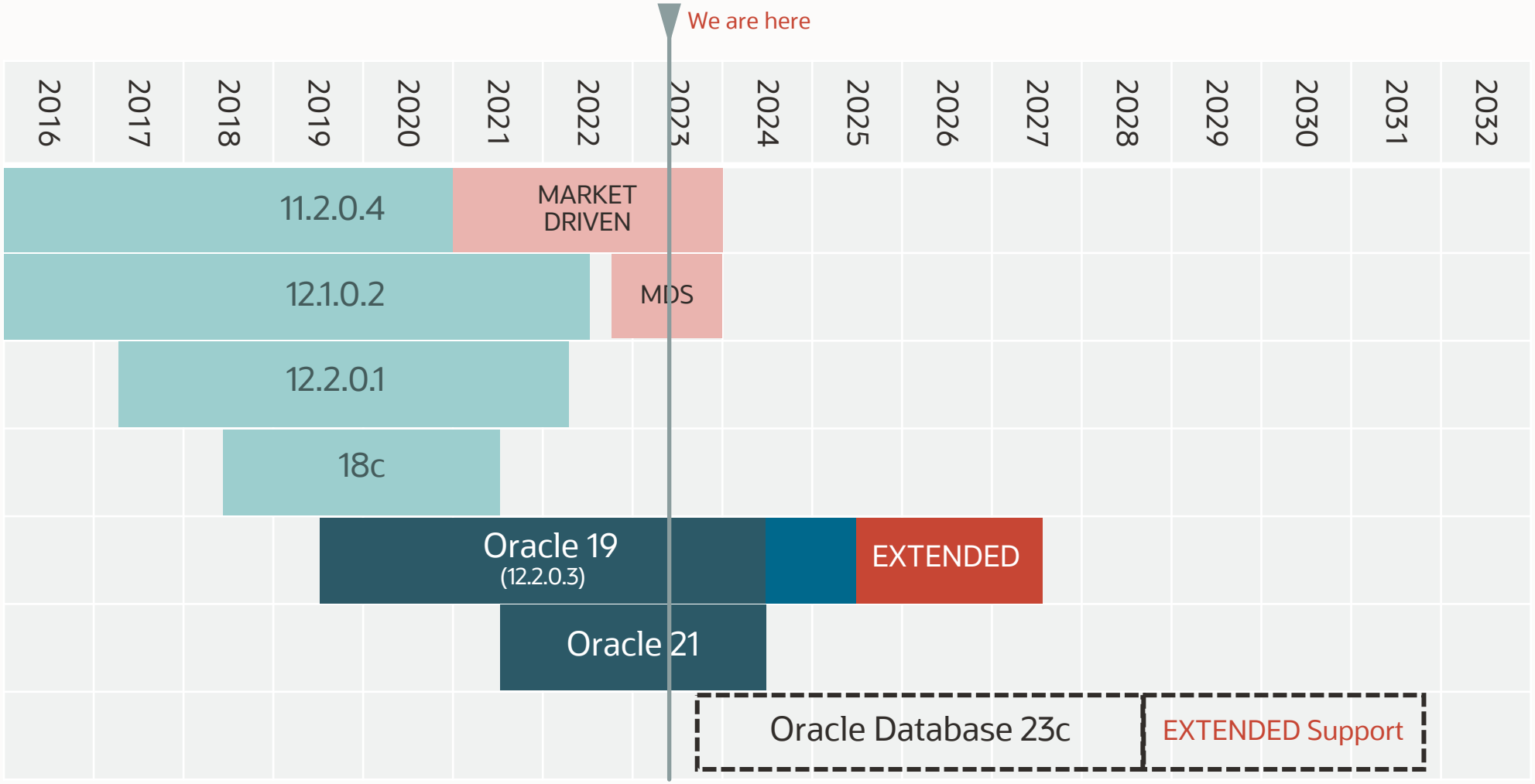
AGENDA

10:00	Introduction
10:15	**Upgrade to Oracle Database 19c**
10:45	Break
11:00	Upgrade to Oracle Database 19c
12:00	Lunch
13:00	Ensure Performance Stability
14:00	Break
14:15	Hands-On Lab
16:00	End

Lifetime Support Policy



Release Strategy | Make Your Plan



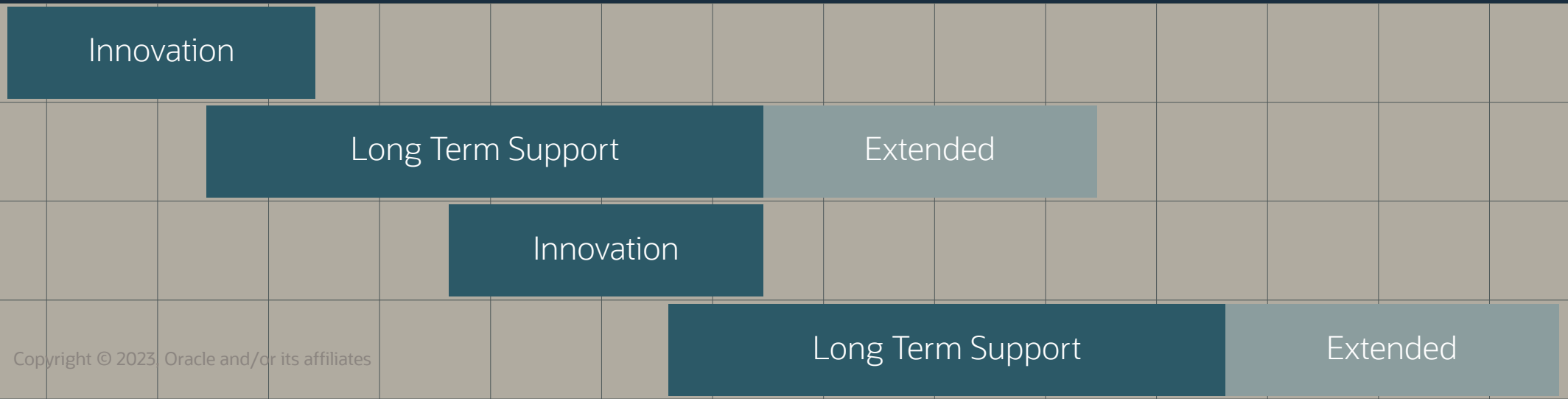
Release Types

LONG TERM SUPPORT

5+ years of Premier Support followed by
3+ years of Extended Support

INNOVATION

2 years of Premier Support
No Extended Support



Move production databases from one Long Term Support release to the next

Next Long Term Support release

Oracle Database 23c

Upgrade possible only from:

- Oracle Database 19c
- Oracle Database 21c

Do you want to upgrade?

Oracle Database 11.2.0.4
Oracle Database 12.1.0.2
Oracle Database 12.2.0.1
Oracle Database 18c

Oracle Database 11.2.0.4
Oracle Database 12.1.0.2
Oracle Database 12.2.0.1
Oracle Database 18c

Oracle Database 19c

Oracle Database 23c

Everybody must upgrade to Oracle Database 19c

- With or without Multitenant

```
--As of Oracle Database 19c you can create up to 3 PDBs without
--having the multitenant license. Applies to SE2 as well

SQL> alter system set max_pdbs=3;
```

Have at least a few environments in Oracle Database 19c using Oracle Multitenant

Your path to successful database upgrades

1

Install Oracle Home including RU and MRP

MOS Note: 2118136.2
MOS Note: 555.1
MOS Note: 2781612.2

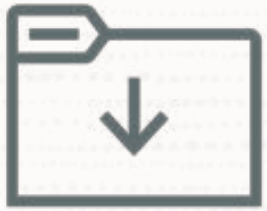
2

Download and deploy the most recent AutoUpgrade

MOS Note: 2485457.1

3

Collect performance information from current source and test thoroughly

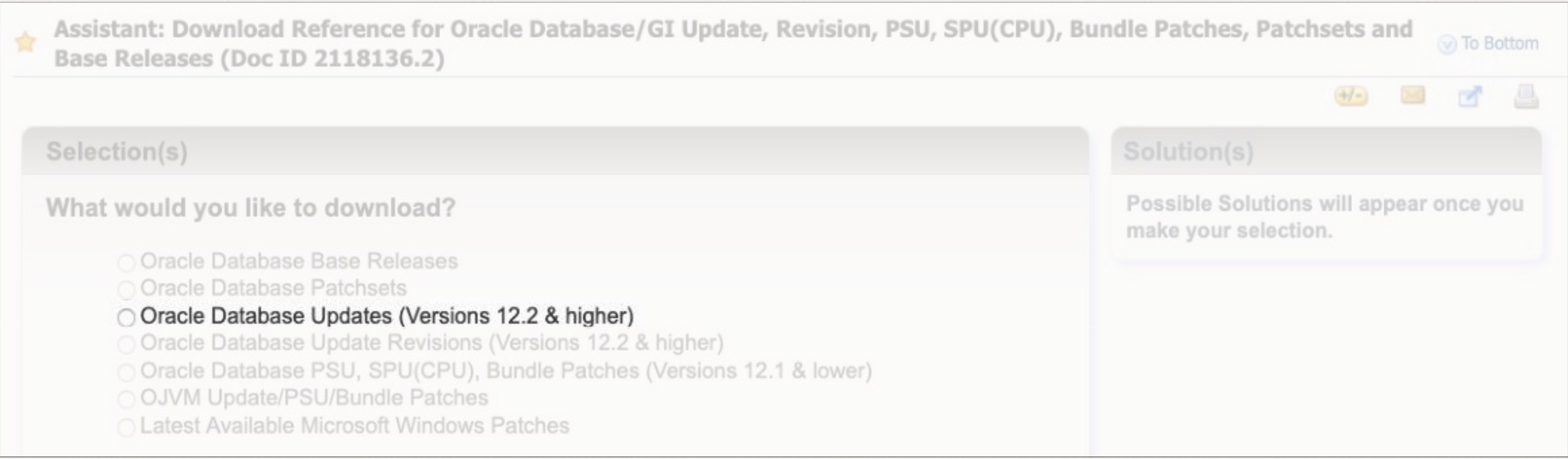


You always start with Oracle Database 19c base release

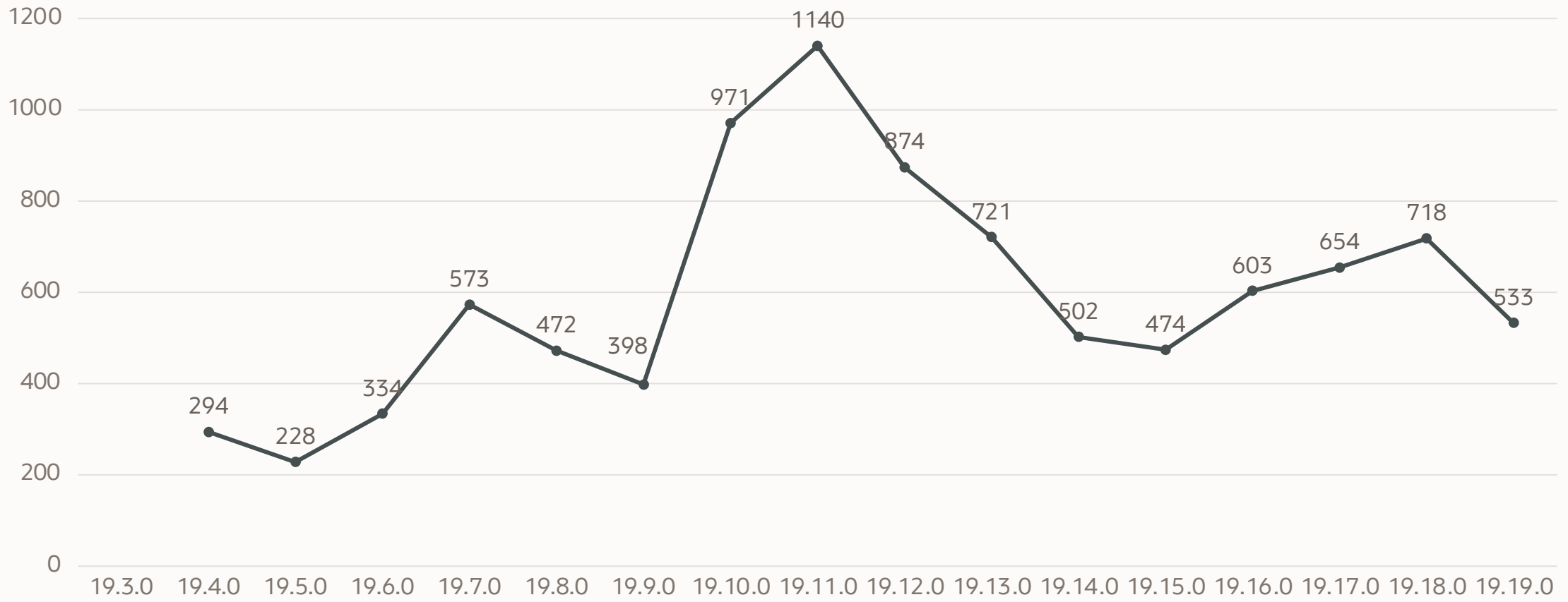
- Oracle Database 19.3.0

Before upgrading, apply the most recent RU

Use the Patch Download Assistant MOS Note: 2118136.2



Release Update Contents



Database 19 Release Updates and Revisions Bugs Fixed Lists (Doc ID 2523220.1)

You are missing out if you don't apply a recent Release Update

- Thousands of fixes
- 243 security fixes

Release Updates might be delayed

- Each quarter a Patch Availability Document is released with information on delayed patches
- Search for *Apr 2023 Patch Availability Document*

ORACLE MY ORACLE SUPPORT

PowerView is Off

Last Login: May 16, 2022

Dashboard | Knowledge | Service Requests | Patches & Updates | Community | Certifications | Managed Cloud | More...

Dashboard >

Give Feedback... There are no Knowledge Base results found. Expandi

KM Search Results

Knowledge Base | Archive | Community | Documentation | Bug | Patch | System Handbook

"Apr 2022 Patch Availability Document"

Start typing to select Product | Any Product Version | Any Platform | Last updated

Results: Knowledge Base | Archive | Community | Documentation | Bug

Tips

Try searching without using quotes or only quote the "important words" in your search.

Recommended Links

Why Can't I Download This Patch? - How Patches and Updates Entitlement Works (Doc ID 1369860.1)

Oracle Support Lifecycle Advisors (Doc ID 250.2)

Knowledge Base Search Results

Critical Patch Update (CPU) Program Apr 2022 Patch Availability Document (DB-only) (Doc ID 2844795.1)

Refine to All > Oracle Cloud > Oracle Infrastructure Cloud > Oracle Cloud at Customer > Gen 1 Exadata Cloud at Customer (Oracle Exadata Database

Apr 22, 2022

Critical Patch Update (CPU) Program Apr 2022 Patch Availability Document (EM-only) (Doc ID 2844807.1)

Refine to All > Enterprise Management > Enterprise Manager Products > Managing Databases using Enterprise Manager > Enterprise Manager for Or

Before upgrading, apply the most important patches

Always use Important Recommended One-Off Patches: MOS Note: 555.1

Recommended Patches for 19.18 DB Home

Below is the list of important patches to consider applying on top of 19.18. In addition to the relevant patches listed below, you should also review patches in Database PSU/BP/Update/Revision - Known Issues Primary Note(Doc ID 1227443.1) and Oracle Database Patches to Consider for 19c (Doc ID 2781612.2) which contains patches to consider for specific areas such as Data Pump, Golden gate etc.

Bug	Fixed in RU	Fixed in MRP	Description	Patches	NON ROLLING	Added
35037877 (replaces 20289608)			[SECURITY] EM patching may fail with ORA-4067	[list-patches]		20-APR-2023
32727143	19.19		[SQL EXECUTION] Transaction-level content isolation for transaction-duration global temporary tables	[list-patches]		20-APR-2023
34557500 (replaces 31544097)	19.19	DBMRP 19.17.0.0.230321, DBMRP 19.18.0.0.230321	[BLOCK TRACK] CTWR caused multiple instances to hung state on the RAC Standby DB	[list-patches]		20-APR-2023
34340632			[AQ] Smart Monitoring & Resiliency in AQ KGL Memory Usage To Help Message Cleanup And Prevent ORA-600 [KGL-HEAP-SIZE-EXCEEDED].	[list-patches]		14-APR-2023
35246710 (replaces 33803836)	19.19	DBMRP 19.18.0.0.230418	[BUFFER CACHE] High "Direct Path Read" Waits After 19.18 DBRU Patching	[list-patches]		08-APR-2023
34832725 (replaces			[SHRD CRSRS] ORA-4031 and / or High Shared Pool Latch Contention During Session Creation in	[list-		25-

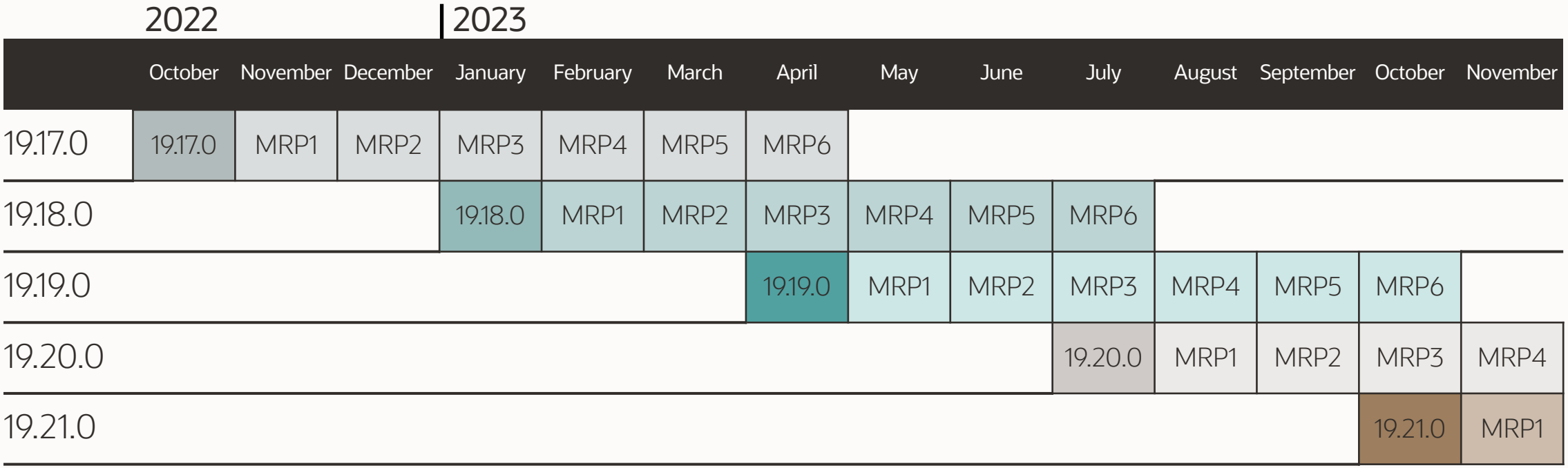
Monthly Recommended Patches

A collection of recommended one-off fixes
provided at monthly intervals
via a single downloadable patch

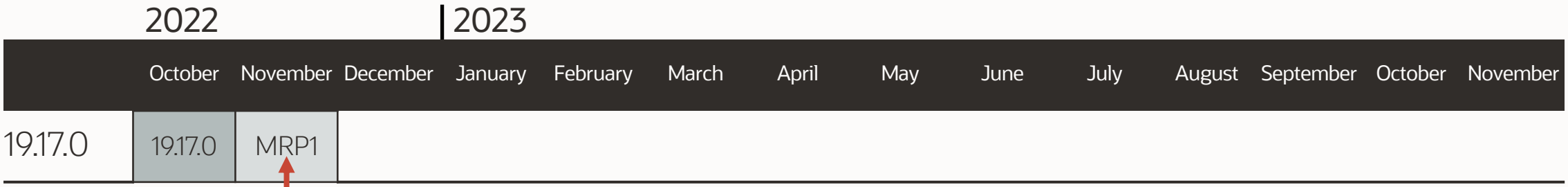
Timeline | Release Updates

	2021				2022				2023				2024		
	January	April	July	October	January	April	July	October	January	April	July	October	January	April	July
19c	19.10.0	19.11.0	19.12.0	19.13.0	19.14.0	19.15.0	19.16.0	19.17.0	19.18.0	19.19.0	19.20.0	19.21.0	19.22.0	19.23.0	19.24.0
21c		21.3.0	21.4.0	21.5.0	21.6.0	21.7.0	21.8.0	21.9.0	21.10.0	21.11.0	21.12.0	21.13.0	21.14.0	21.15.0	

Monthly Recommended Patches | Timeline

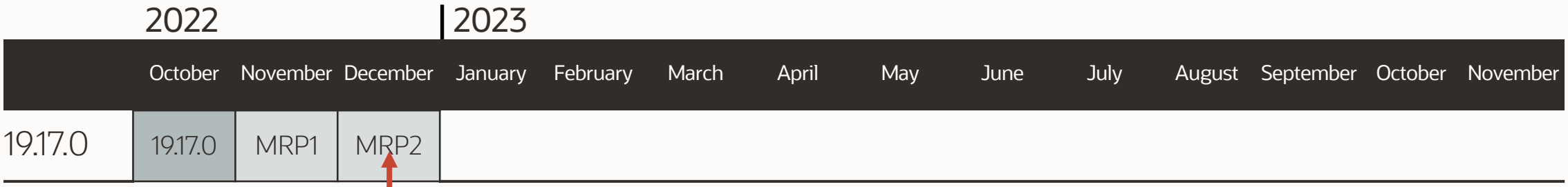


Monthly Recommended Patches | Step-by-Step



Bug	Fixed in RU	Fixed in MRP	Description	Patches	NON ROLLING	Added
33896423		DBMRP 19.17.0.0.221115	Flush Out Stale Antilocks and Convert kclcls_2 and kclantilock_17 to Soft Assert	[list-patches]		01-SEP-2022
34333986		DBMRP 19.17.0.0.221115	ORA-600 [KTUSCV1:CV Buf Too Big]/ [Kdbblkcheckerror] Block Corruption With Check Code 6127 After Migrating From Solaris to Linux	[list-patches]		28-JUL-2022
30691454		DBMRP 19.17.0.0.221115	DBHOME patching completely hung with kpdbhashtable_find multiple instance hang	[list-patches]		22-JUL-2022

Monthly Recommended Patches | Step-by-Step



Bug	Fixed in RU	Fixed in MRP	Description	Patches	NON ROLLING	Added
34649727			Linux: ORA-800 / Set Priority / DB Performance Merge Patch for 19.17 - 34286265 34318125	[list-patches]		18-OCT-2022
33896423		DBMRP 19.17.0.0.221115	Flush Out Stale Antilocks and Convert kclcls_2 and kclantilock_17 to Soft Assert	[list-patches]		01-SEP-2022
34333986		DBMRP 19.17.0.0.221115	ORA-600 [KTUSCV1:CV Buf Too Big]/ [Kdbblkcheckerror] Block Corruption With Check Code 6127 After Migrating From Solaris to Linux	[list-patches]		28-JUL-2022
30691454		DBMRP 19.17.0.0.221115	DBHOME patching completely hung with kpdbhashtable_find multiple instance hang	[list-patches]		22-JUL-2022

Monthly Recommended Patches | Step-by-Step

	2022			2023										
	October	November	December	January	February	March	April	May	June	July	August	September	October	November
19.17.0	19.17.0	MRP1	MRP2	MRP3										
19.18.0				19.18.0										

Monthly Recommended Patches | Step-by-Step

	2022			2023										
	October	November	December	January	February	March	April	May	June	July	August	September	October	November
19.17.0	19.17.0	MRP1	MRP2	MRP3	MRP4									
19.18.0				19.18.0	MRP1									

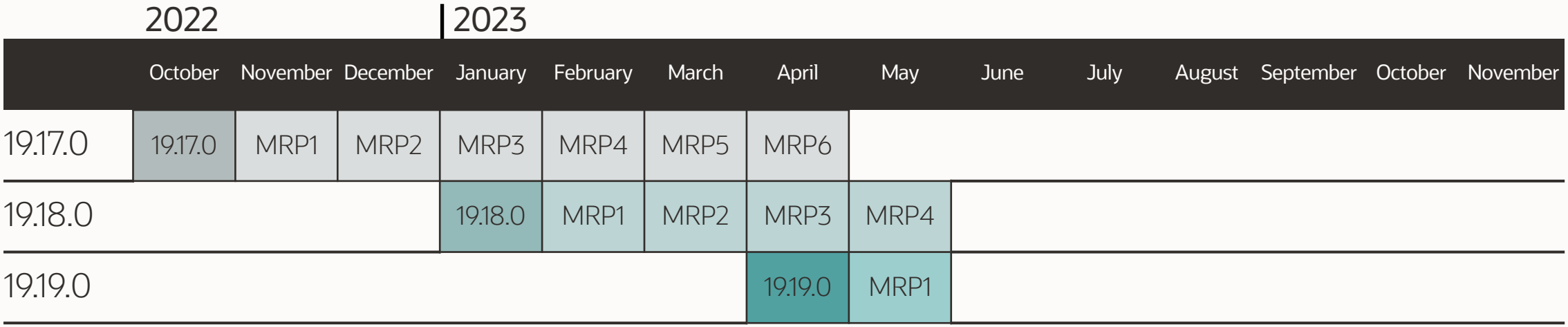
Monthly Recommended Patches | Step-by-Step

	2022			2023										
	October	November	December	January	February	March	April	May	June	July	August	September	October	November
19.17.0	19.17.0	MRP1	MRP2	MRP3	MRP4	MRP5								
19.18.0				19.18.0	MRP1	MRP2								

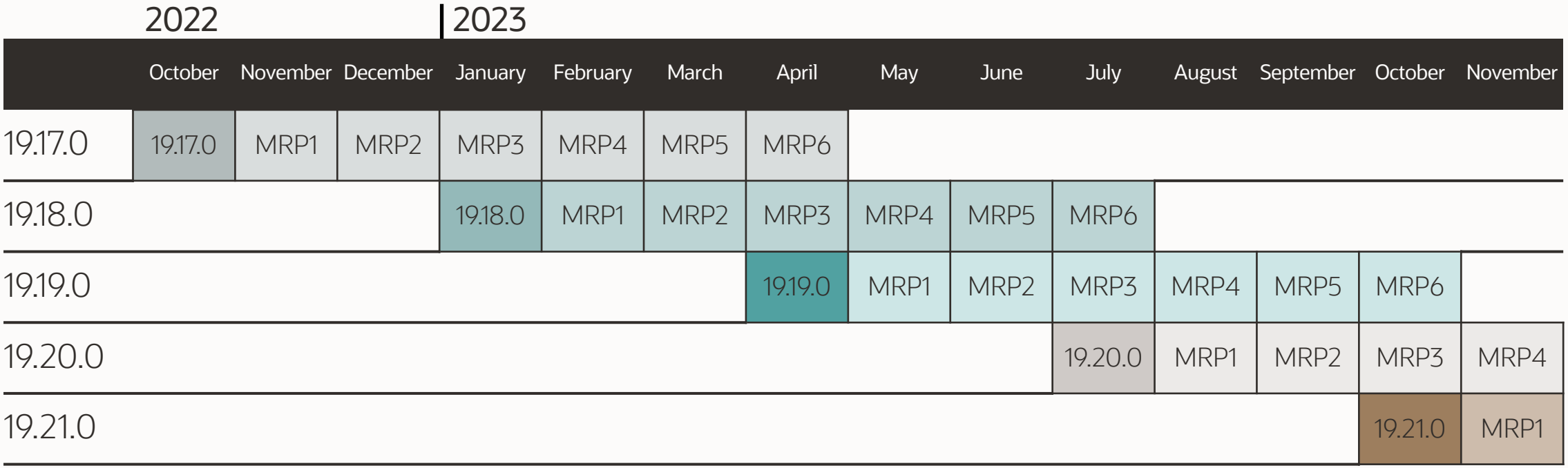
Monthly Recommended Patches | Step-by-Step

	2022			2023										
	October	November	December	January	February	March	April	May	June	July	August	September	October	November
19.17.0	19.17.0	MRP1	MRP2	MRP3	MRP4	MRP5	MRP6							
19.18.0				19.18.0	MRP1	MRP2	MRP3							
19.19.0							19.19.0							

Monthly Recommended Patches | Step-by-Step



Monthly Recommended Patches | Step-by-Step



MRP Facts | Contents

Monthly Recommended Patches get sourced mostly from MOS Note 555.1

Oracle Database 19c Important Recommended One-off Patches (Doc ID 555.1)

Through our review of service requests, we often find that issues encountered are the result of customers being on an older Release Update (RU) or Release Update Revision (RUR). Many issues will have been fixed in the latest updates, which we always recommend.

As noted in the update-specific tables below, fixes for known issues are targeted for inclusion in the first available RU or RUR. You can always find the latest RUs, RURs, other patches, lists of fixed bugs and known issues in Primary Note for Database Proactive Patch Program(Doc ID 888.1). Identifying and installing the latest updates (patches) helps ensure you are using the most current content for security, functional, regression and bug fixes, as well as minor enhancements and any emergency one-offs.

Note:
Beginning with the October 2022 patching cycle, 19c RURs will no longer be provided for 19.17.0 and above. No additional RURs will be delivered on any platform after the delivery of Oracle Database 19c RUR 19.16.2 in January, 2023.
Refer to Sunsetting of 19c RURs and FAQ (Doc ID 2898381.1) for further details.
To provide customers more frequent access to recommended and well-tested collections of patches, Oracle is pleased to introduce Monthly Recommended Patches (MRPs) starting Nov 2022. MRPs are supported only on Linux x86-64 platform.
Refer to Introducing Monthly Recommended Patches (MRPs) and FAQ (Doc ID 2898740.1) for further details.

In addition to the relevant patches listed below, you should apply patches based on the specific RU after reviewing the following My Oracle Support knowledge documents:

1. Database PSU/BP/Update/Revision - Known Issues Primary Note(Doc ID 1227443.1)
2. Oracle Database Patches to Consider for 19c (Doc ID 2781612.2) which contains patches to consider for specific areas such as Data Pump, Golden gate etc.

NOTE: Patches that start with "X8M" must be applied by Exadata customers running on X8M machines. Exadata customers who are not on X8M systems can ignore these patches. For details on X8M patches, refer to Exadata X8M (RoCE-based systems) Software Requirements and Recommendations(Doc ID 2724126.1)

NOTE: If you are an Applications Unlimited (for example, EBS) customer, you should follow any directions given in your product-specific documentation on applicability of Release Updates (RUs) and Release Update Revisions (RURs).

The information below lists any additional patches (both rolling and non-rolling) that are recommended for installation on top of each RU. Click the relevant link for details. Be aware that this document only shows patches for the latest four RUs.

NOTE: Latest four RUs.

1. 19.17 GI RU - Patch 34416665 (for Grid Infrastructure), 19.17 DB RU - Patch 34419443 (for DB Home)
2. 19.16 GI RU - Patch 34130714 (for Grid Infrastructure), 19.16 DB RU - Patch 34133642 (for DB Home)
3. 19.15 GI RU - Patch 33803476 (for Grid Infrastructure), 19.15 DB RU - Patch 33806152 (for DB Home)
4. 19.14 GI RU - Patch 33509923 (for Grid Infrastructure), 19.14 DB RU - Patch 33515361 (for DB Home)

Recommended Patches for 19.17 DB Home

Below is the list of important patches to consider applying on top of 19.17. In addition to the relevant patches listed below, you should also review patches in Database PSU/BP/Update/Revision - Known Issues Primary Note(Doc ID 1227443.1) and Oracle Database Patches to Consider for 19c (Doc ID 2781612.2) which contains patches to consider for specific areas such as Data Pump, Golden gate etc.

Bug	Fixed in RU	Fixed in MRP	Description	Patches	NON ROLLING	Added
34789241			Recommended RAC Merge for 19.17 - 33896423,34310304,34377917,34485380,34562023,34060122	[list-patches]		13-NOV-2022
34649727			Linux: ORA-800 / Set Priority / DB Performance Merge Patch for 19.17 - 34286265 34318125	[list-patches]		18-OCT-2022
33896423		DBMRP 19.17.0.0.221115	Flush Out Stale Antilocks and Convert kclds_2 and kclantilock_17 to Soft Assert	[list-patches]		01-SEP-2022
34333986		DBMRP 19.17.0.0.221115	ORA-600 [KTUSCV1:CV Buf Too Big]/ [Kdbblkcheckerror] Block Corruption With Check Code 6127 After Migrating From Solaris to Linux	[list-patches]		28-JUL-2022
30691454		DBMRP 19.17.0.0.221115	DBHOME patching completely hung with kpdbhashtable_find multiple instance hang	[list-patches]		22-JUL-2022
29213893			DBMS_STATS Failing With Error Ora-1422 When Gathering Stats for User$ Table	[list-patches]	YES	01-SEP-2021

Recommended Patches for 19.17 GI Home

Below is the list of important patches to consider applying on top of 19.17. In addition to the relevant patches listed below, you should also review patches in Database PSU/BP/Update/Revision - Known Issues Primary Note(Doc ID 1227443.1) and Oracle Database Patches to Consider for 19c (Doc ID 2781612.2) which contains patches to consider for specific areas such as Data Pump, Golden gate etc.
Only one OCW (Oracle Clusterware) patch should be applied to GI homes. Ensure you apply the patch starting with X8M if you are on an X8M system

Bug	Fixed in RU	Fixed in MRP	Description	Patches	NON ROLLING	Added
34649727			Linux: ORA-800 / Set Priority / DB Performance Merge Patch for 19.17 - 34286265 34318125	[list-patches]		18-OCT-2022

An MRP is a collection of several one-off patches

- Delivered via a merge patch
- Included patches must be RAC Rolling Installable

An MRP does not change the release number

- Like `v$instance.version_full`

MRP content is cumulative but only within one MRP line

- Example:
  19.17.0 MRP5 contains all previous MRPs done for Oracle 19.17.0
- MRPs are not bundle patches,
  so to install a newer MRP you must roll off previous MRPs

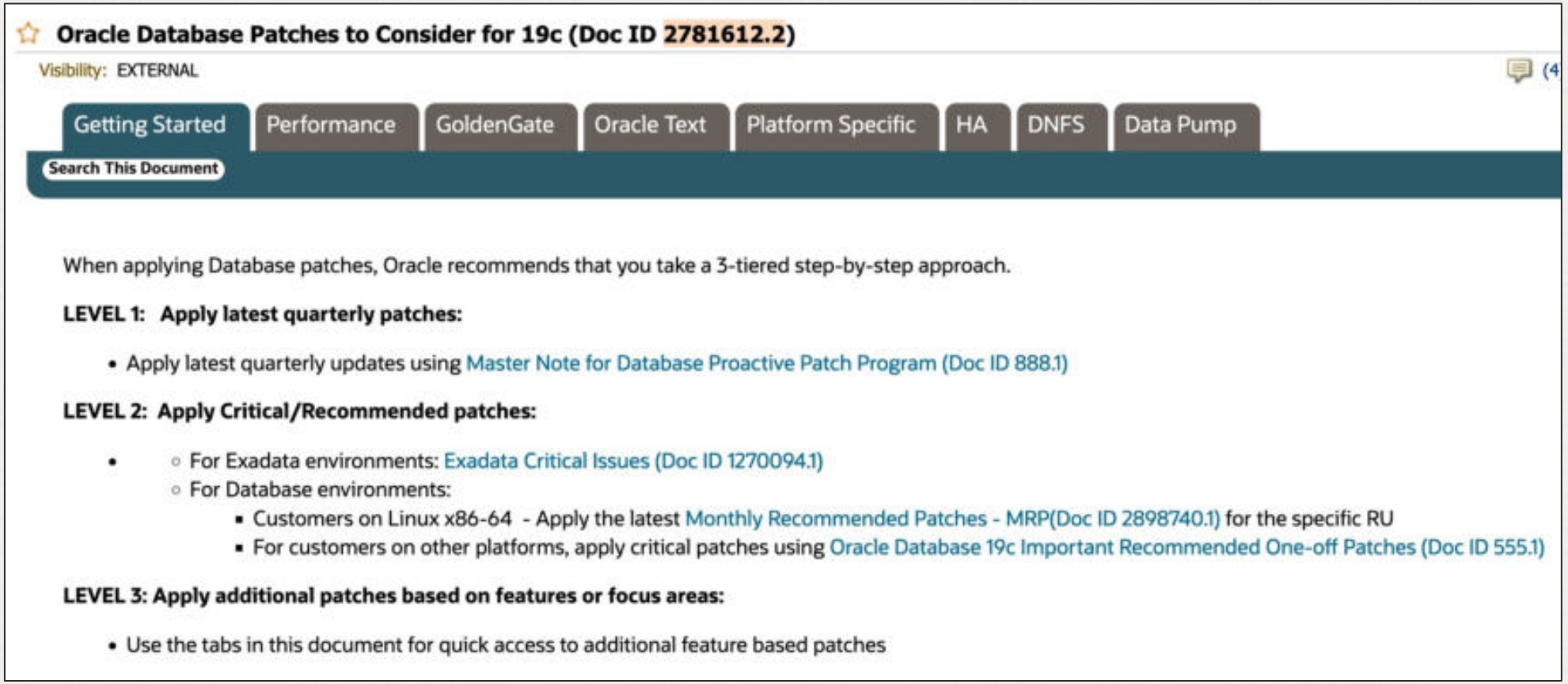
MRPs can contain security fixes

- Release Updates remain the primary security fix delivery mechanism

In OCI, include MRPs
by creating a *Database Software Image*

Before upgrading, apply the most important patches

In addition, use Patches to consider for 19c: MOS Note: 2781612.2



Oracle Database Patches to Consider for 19c (Doc ID 2781612.2)

Visibility: EXTERNAL

Getting Started | Performance | GoldenGate | Oracle Text | Platform Specific | HA | DNFS | Data Pump

Search This Document

When applying Database patches, Oracle recommends that you take a 3-tiered step-by-step approach.

LEVEL 1: Apply latest quarterly patches:

- Apply latest quarterly updates using Master Note for Database Proactive Patch Program (Doc ID 888.1)

LEVEL 2: Apply Critical/Recommended patches:

- For Exadata environments: Exadata Critical Issues (Doc ID 1270094.1)
- For Database environments:
  - Customers on Linux x86-64 - Apply the latest Monthly Recommended Patches - MRP(Doc ID 2898740.1) for the specific RU
  - For customers on other platforms, apply critical patches using Oracle Database 19c Important Recommended One-off Patches (Doc ID 555.1)

LEVEL 3: Apply additional patches based on features or focus areas:

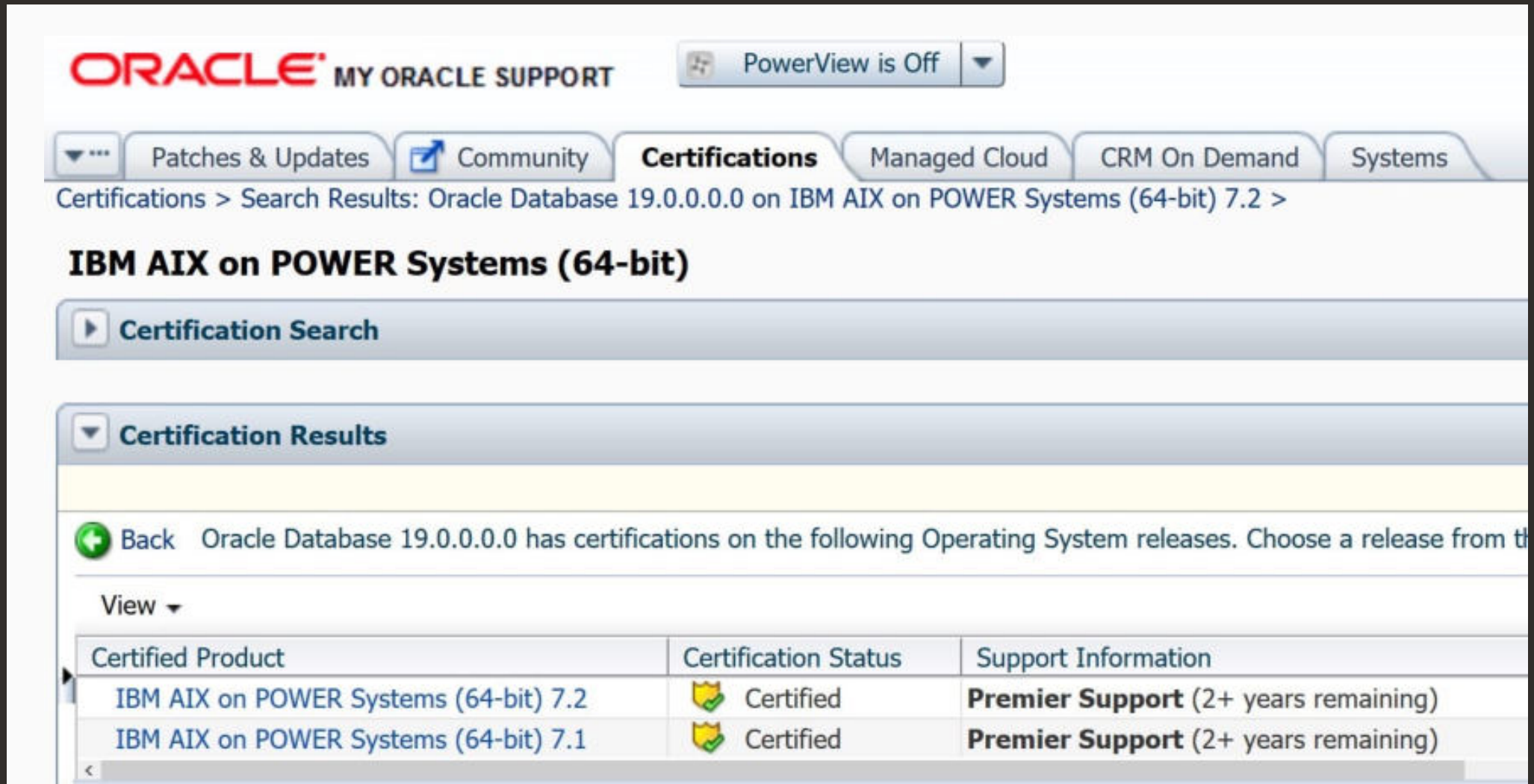
- Use the tabs in this document for quick access to additional feature based patches

Photo by Pablo Heimplatz on Unsplash

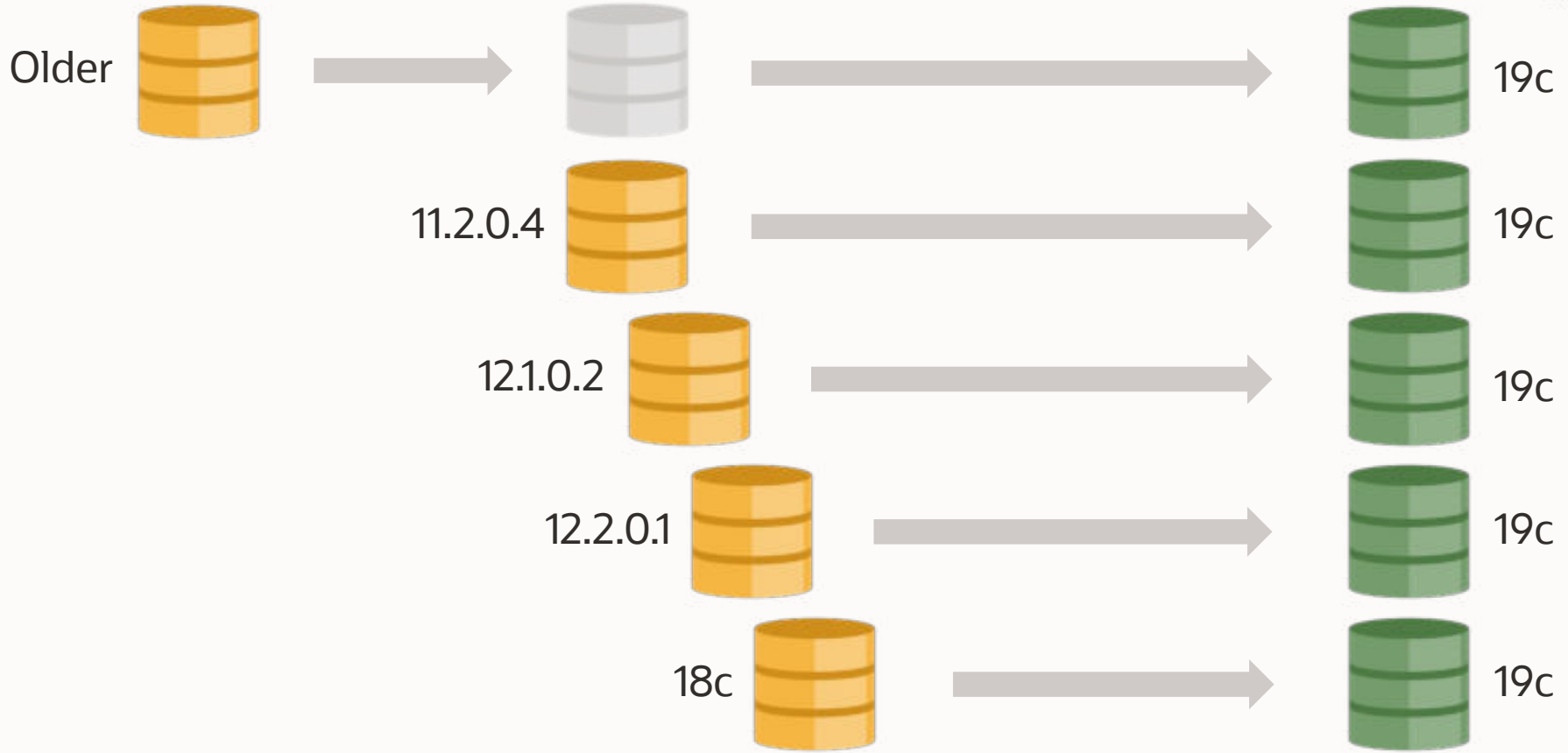
Check

Before Upgrade

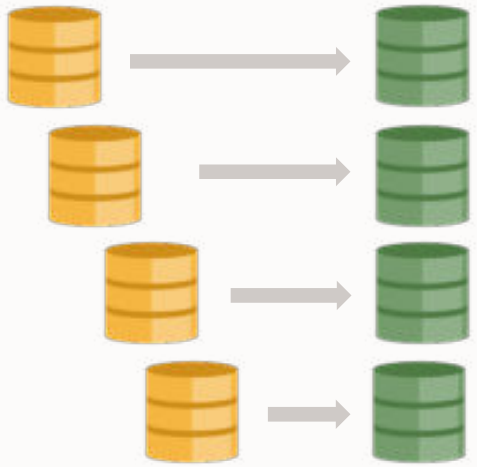
Supportability | OS Certification



Database Upgrade | Supported Releases



Database Upgrade | Supported Releases



Database / Oracle / Oracle Database / Release 19

Database Upgrade Guide

Oracle Database Releases That Support Direct Upgrade

Review the supported options for direct upgrades to the latest Oracle Database release.

You can perform a direct upgrade to the new release from the following releases:

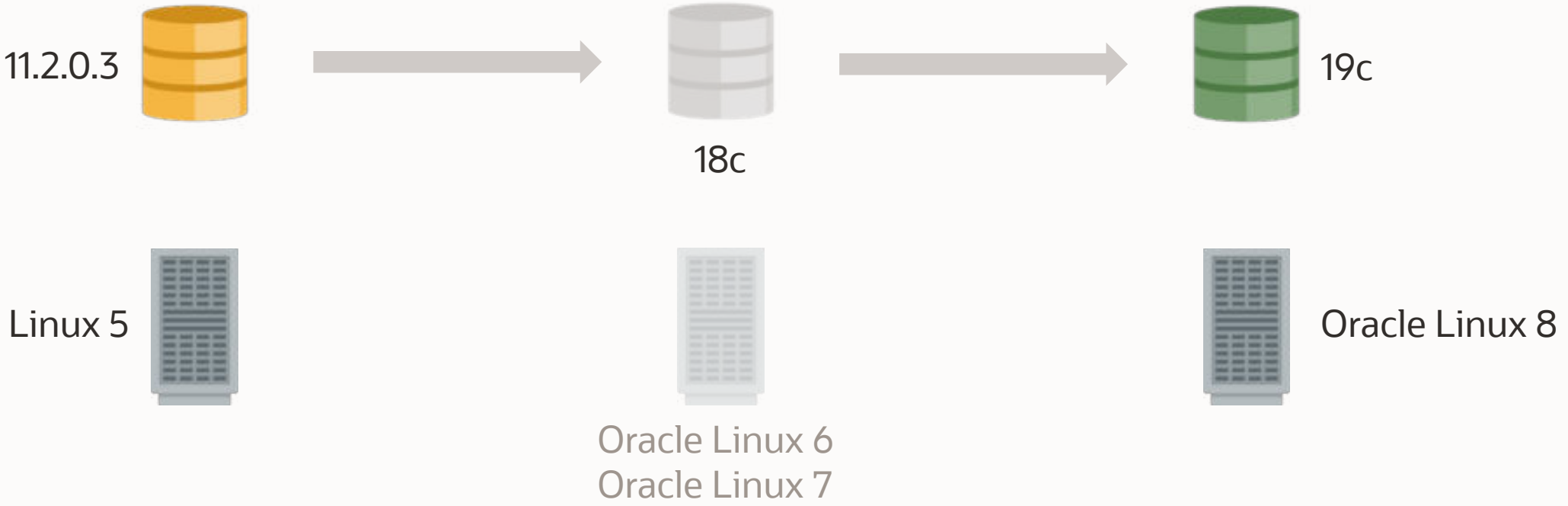
- 11.2.0.4
- 12.1.0.2
- 12.2.0.1
- 18

Database Upgrade Guide

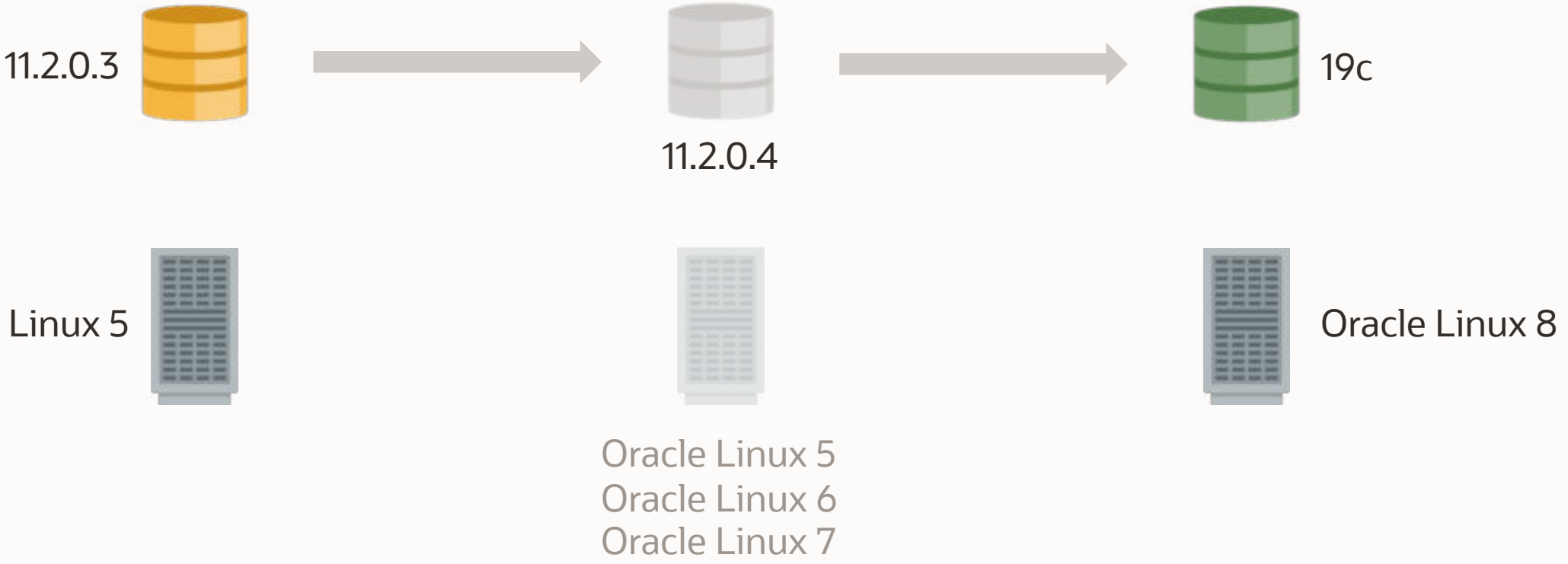
Database Upgrade | Intermediate Upgrades

- Ideally - upgrade as far as possible in the intermediate upgrade
- But often there is a platform upgrades as well
- So - it depends

Database Upgrade | Intermediate Upgrades



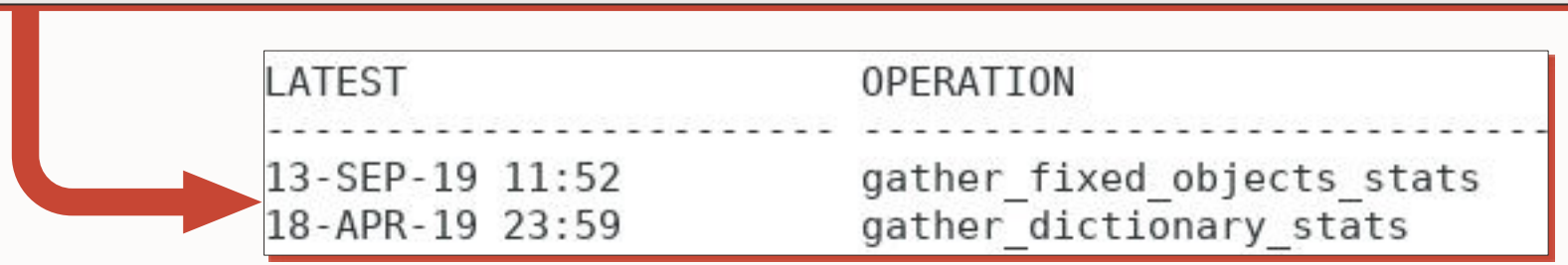
Database Upgrade | Intermediate Upgrades



Upgrade 19c | Speed it up

Check when dictionary stats have been gathered the last time

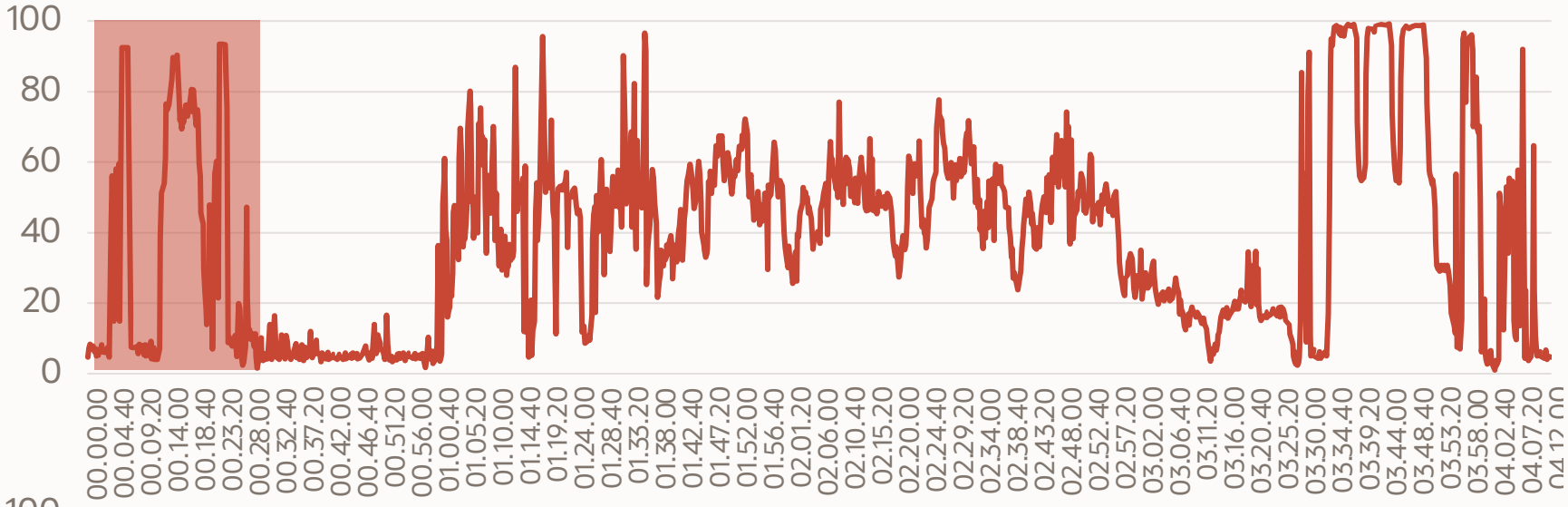
```
SELECT
   to_char(max(end_time),'dd-mon-yy hh24:mi') latest, operation
FROM
   dba_optstat_operations
WHERE
   operation in ('gather_dictionary_stats', 'gather_fixed_objects_stats')
GROUP BY
   operation;
```

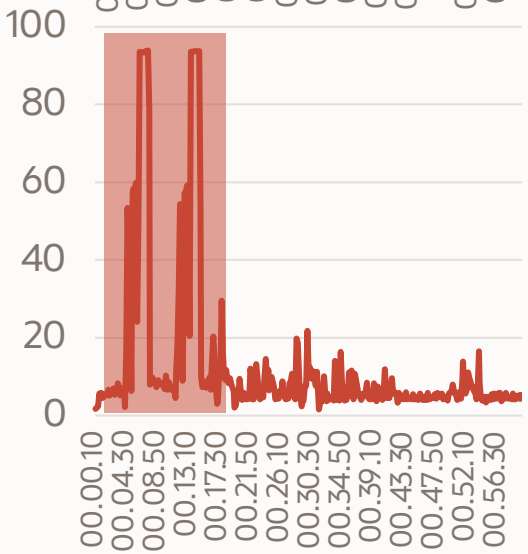


```
LATEST                    OPERATION
------------------------- ------------------------------
13-SEP-19 11:52           gather_fixed_objects_stats
18-APR-19 23:59           gather_dictionary_stats
```

Refresh stats a day before the upgrade

Upgrade 19c | Gather Stats In Advance





Gathering stats in advance saves 12 minutes

Operating System | Recommendations

Oracle Database (RDBMS) on Unix AIX,HP-UX,Linux,Solaris and MS Windows Operating Systems Installation and Configuration Requirements Quick Reference (12.1/12.2/18c/19c) (Doc ID 1587357.1)

19c

- Common Requirements
- Oracle Linux 7
- RHEL 7
- SLES 12
- SLES 15
- AIX 7.1
- AIX 7.2
- HP-UX
- Solaris 11

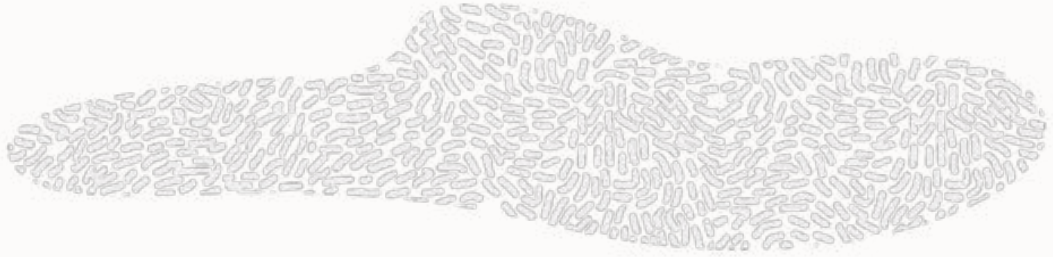
Oracle Linux 7

Oracle Linux 7

OS Version	Patches/Packages	Kernel settings
Oracle Linux 7.4 with the Unbreakable Enterprise Kernel 4: 4.1.12-124.19.2.el7uek.x86_64 or later Oracle Linux 7.4 with the Unbreakable Enterprise Kernel 5: 4.14.35-1818.1.6.el7uek.x86_64 or later Oracle Linux 7.5 with the Red Hat Compatible kernel: 3.10.0-862.11.6.el7.x86_64 or later	bc binutils compat-libcap1 compat-libstdc++ elfutils-libelf elfutils-libelf-devel fontconfig-devel glibc glibc-devel ksh libaio libaio-devel	semmsl 250 semmns 32000 semopm 100 semmni 128 shmall Greater than or equal to the value of shmmax, in pages. shmmax Half the size of physical memory in bytes shmmni 4096 panic_on_oops 1 file-max 6815744 aio-max-nr 1048576 ip_local_port_range Minimum:

For important databases, execute **database health checks** before upgrade

Health Checks



Health check script

- Download from MOS Note: 1366971
- In Multitenant, it must be run in each PDB separately

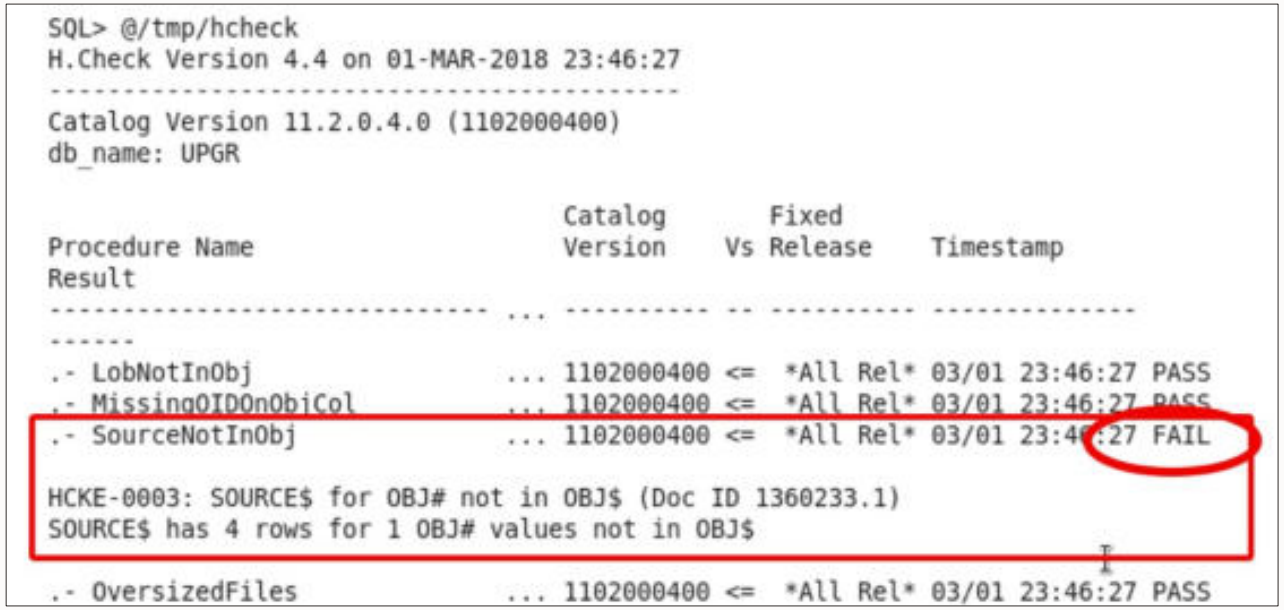
ORAchk Upgrade Readiness Assessment

- Part of Autonomous Health Framework (AHF)
- Download from MOS Note: 1457357.1
- Upgrade Readiness Check – MOS Note: 2550798.1

Health Check | hcheck.sql

If your database is highly important, do a health check

- Lightweight, non-intrusive script
- Checks consistency of selected dictionary relationships
- hcheck.sql - Script to Check for Known Problems (Doc ID 136697.1)



```
SQL> @/tmp/hcheck
H.Check Version 4.4 on 01-MAR-2018 23:46:27
----------------------------------------
Catalog Version 11.2.0.4.0 (1102000400)
db_name: UPGR

                                Catalog       Fixed
Procedure Name                  Version    Vs Release    Timestamp
Result
------------------------------ ... ---------- -- ---------- --------------
------
.- LobNotInObj                 ... 1102000400 <=  *All Rel* 03/01 23:46:27 PASS
.- MissingOIDOnObjCol          ... 1102000400 <=  *All Rel* 03/01 23:46:27 PASS
.- SourceNotInObj              ... 1102000400 <=  *All Rel* 03/01 23:46:27 FAIL

HCKE-0003: SOURCE$ for OBJ# not in OBJ$ (Doc ID 1360233.1)
SOURCE$ has 4 rows for 1 OBJ# values not in OBJ$

.- OversizedFiles              ... 1102000400 <=  *All Rel* 03/01 23:46:27 PASS
```

AGENDA

10:00	Introduction
10:15	Upgrade to Oracle Database 19c
10:45	**Break**
11:00	Upgrade to Oracle Database 19c
12:00	Lunch
13:00	Ensure Performance Stability
14:00	Break
14:15	Hands-On Lab
16:00	End

Upgrade to Oracle Database 19c

Photo by SpaceX on Unsplash

Upgrade using

AutoUpgrade

complete
AUTOMATION

rich on
FEATURES

fleet
SCALE

highly
RECOMMENDED

AutoUpgrade | Overview

11.2.0.4
12.1.0.2
12.2.0.1
18c
19c

ONE BUTTON APPROACH

12.2.0.1
18c
19c
21c

1. CONFIG FILE

2. ANALYZE

3. UPGRADE

Optionally,
convert to PDB

Get started | Quick Start Guide

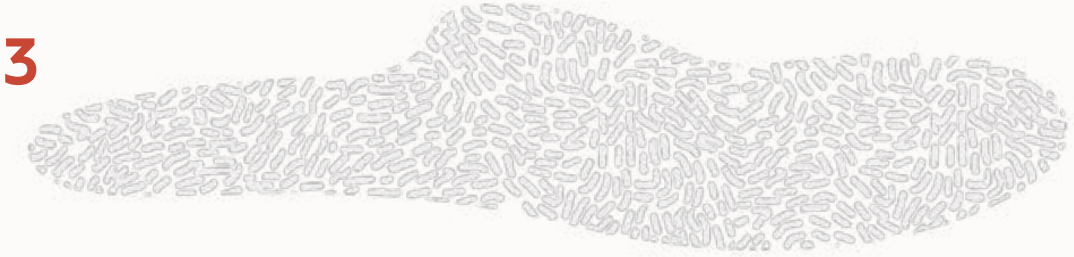
Simple overview
Read it, try it
Download from oracle.com

Photo by Joshua Sortino on Unsplash

AutoUpgrade

The ONLY recommended way to upgrade databases

Upgrade | AutoUpgrade - As Easy As 1-2-3

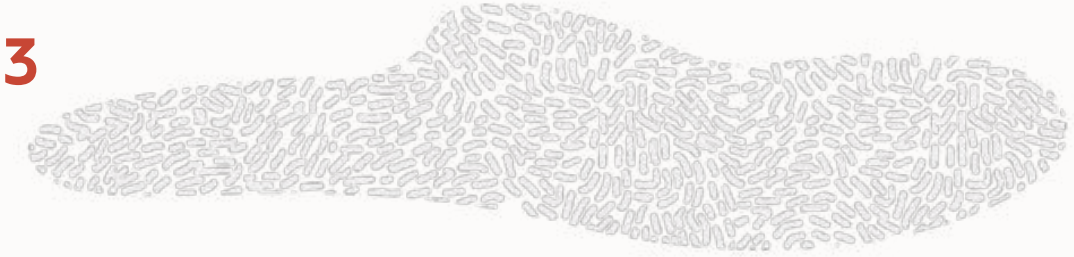


START	1. DOWNLOAD	2. CONFIG	3. DEPLOY	SUCCESS

Supported source releases
- 11.2.0.4
- 12.1.0.2
- 12.2.0.1
- 18
- 19

All architectures (CDB and non-CDB)
All supported operating systems
All editions (SE2, EE)
All types (single instance and RAC)

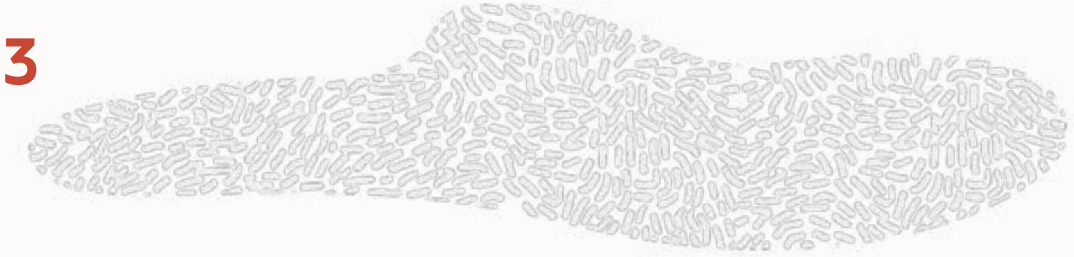
Upgrade | AutoUpgrade - As Easy As 1-2-3



START	1. DOWNLOAD	2. CONFIG	3. DEPLOY	SUCCESS

Download from My Oracle Support ID 2485457.1

Upgrade | AutoUpgrade - As Easy As 1-2-3

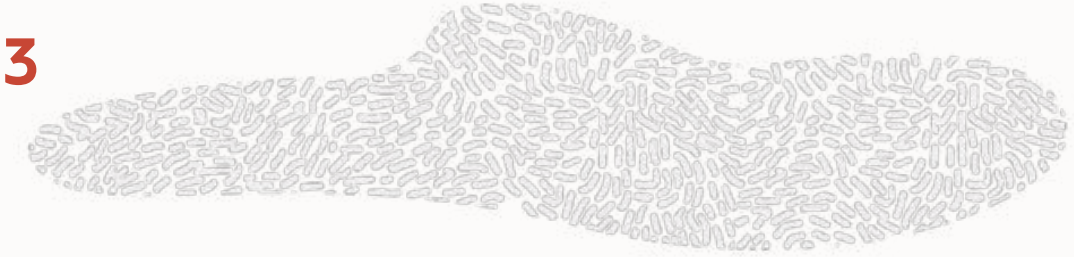


START	1. DOWNLOAD	2. CONFIG	3. DEPLOY	SUCCESS

Simple text file

```
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=CDB1
```

Upgrade | AutoUpgrade - As Easy As 1-2-3



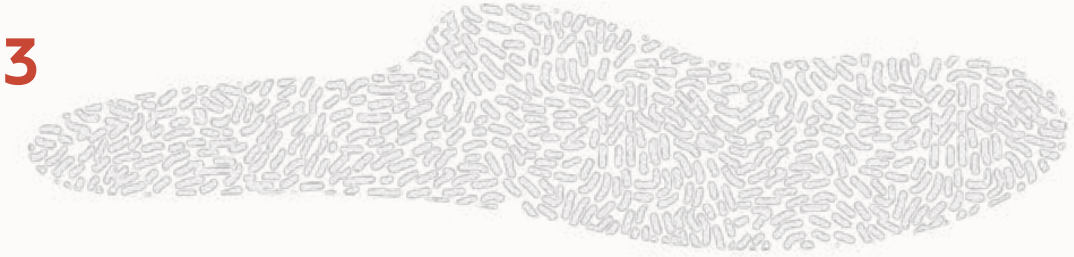
START	1. DOWNLOAD	2. CONFIG	3. DEPLOY	SUCCESS

One command

```
$ java -jar autoupgrade.jar -config cdb1.cfg -mode deploy
```

Advanced monitoring and logging

Upgrade | AutoUpgrade - As Easy As 1-2-3

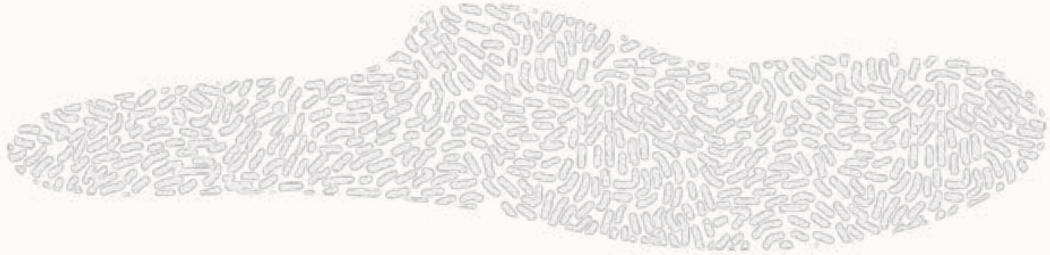


START	1. DOWNLOAD	2. CONFIG	3. DEPLOY	SUCCESS

Supported target releases

- 12.2.0.1
- 18
- 19
- 21
- Any future release

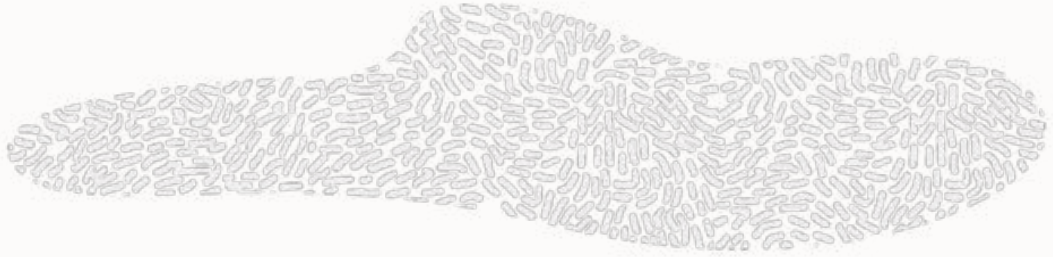
AutoUpgrade | Need And Don't Need



JAVA	JAR FILE	AGENTS	ENTERPRISE MANAGER	DBUA	EXTRA LICENSE

- Java 8 required
  - Part of Oracle Home since 12.1.0.2
- 3 MB jar file

AutoUpgrade | Need And Don't Need



JAVA	JAR FILE	AGENTS	ENTERPRISE MANAGER	DBUA	EXTRA LICENSE

- No agents to install
- Enterprise Manager not needed
- AutoUpgrade offers superior functionality
- No extra license

AutoUpgrade Essentials

Photo by Daniel Cheung on Unsplash

AutoUpgrade | Essentials

Download

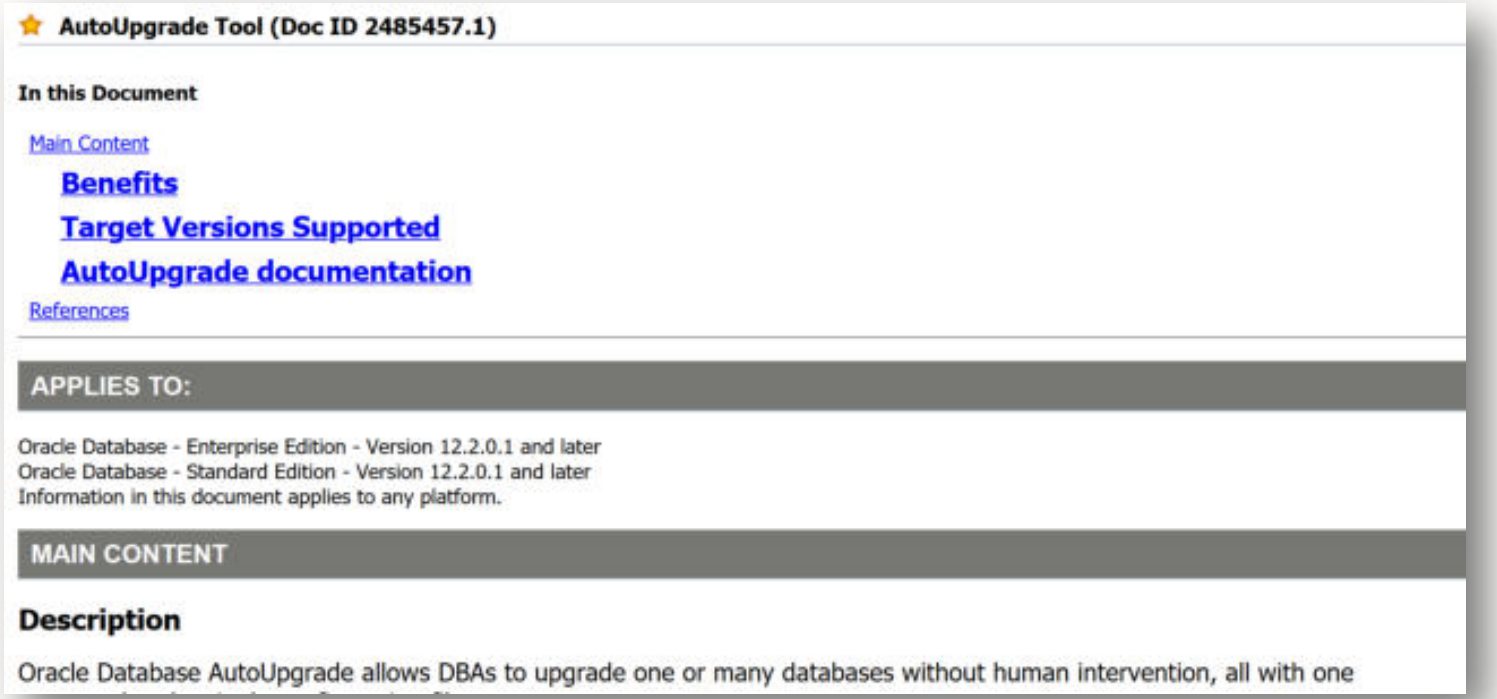
Configure

Analyze

Check

Upgrade

Always download latest version from MOS



AutoUpgrade Tool (Doc ID 2485457.1)

In this Document

Main Content

Benefits

Target Versions Supported

AutoUpgrade documentation

References

APPLIES TO:

Oracle Database - Enterprise Edition - Version 12.2.0.1 and later
Oracle Database - Standard Edition - Version 12.2.0.1 and later
Information in this document applies to any platform.

MAIN CONTENT

Description

Oracle Database AutoUpgrade allows DBAs to upgrade one or many databases without human intervention, all with one

AutoUpgrade | Essentials

Download
Configure
Analyze
Check
Upgrade

Check your version

```
$ java -jar autoupgrade.jar -version

build.version 22.5.221011
build.date 2022/10/11 14:23:59 -0400
build.hash e9428661
build.hash_date 2022/10/11 12:55:45 -0400
build.supported_target_versions 12.2,18,19,21
build.type production
```

AutoUpgrade | Essentials

Download

Configure

Analyze

Check

Upgrade

AutoUpgrade handles older releases as well

```
$ java -jar autoupgrade.jar -version

build.version 22.5.221011
build.date 2022/10/11 14:23:59 -0400
build.hash e9428661
build.hash_date 2022/10/11 12:55:45 -0400
build.supported_target_versions 12.2,18,19,21
build.type production
```

AutoUpgrade | Essentials

- Download
- **Configure**
- Analyze
- Check
- Upgrade

Shortest possible config file version

```
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=CDB1
```

Or, generate a sample config file

```
$ java -jar autoupgrade.jar -create_sample_file config

Created sample configuration file /home/oracle/sample_config.cfg
```

Pro tip: *upg1* is a prefix that you decide. Use it to define multiple databases

AutoUpgrade | Essentials

- Download
- Configure
- **Analyze**
- Check
- Upgrade

Analyze your database

```
$ java -jar autoupgrade.jar -config CDB1.cfg -mode analyze

...

upg> Job 100 completed

Please check the summary report at:
/u01/app/oracle/cfgtoollogs/autoupgrade/cfgtoollogs/upgrade/auto/status/status.html
/u01/app/oracle/cfgtoollogs/autoupgrade/cfgtoollogs/upgrade/auto/status/status.log
```

Pro tip: Analyze is similar to running `preupgrade.jar`

AutoUpgrade | Essentials

- Download
- Configure
- Analyze
- **Check**
- Upgrade

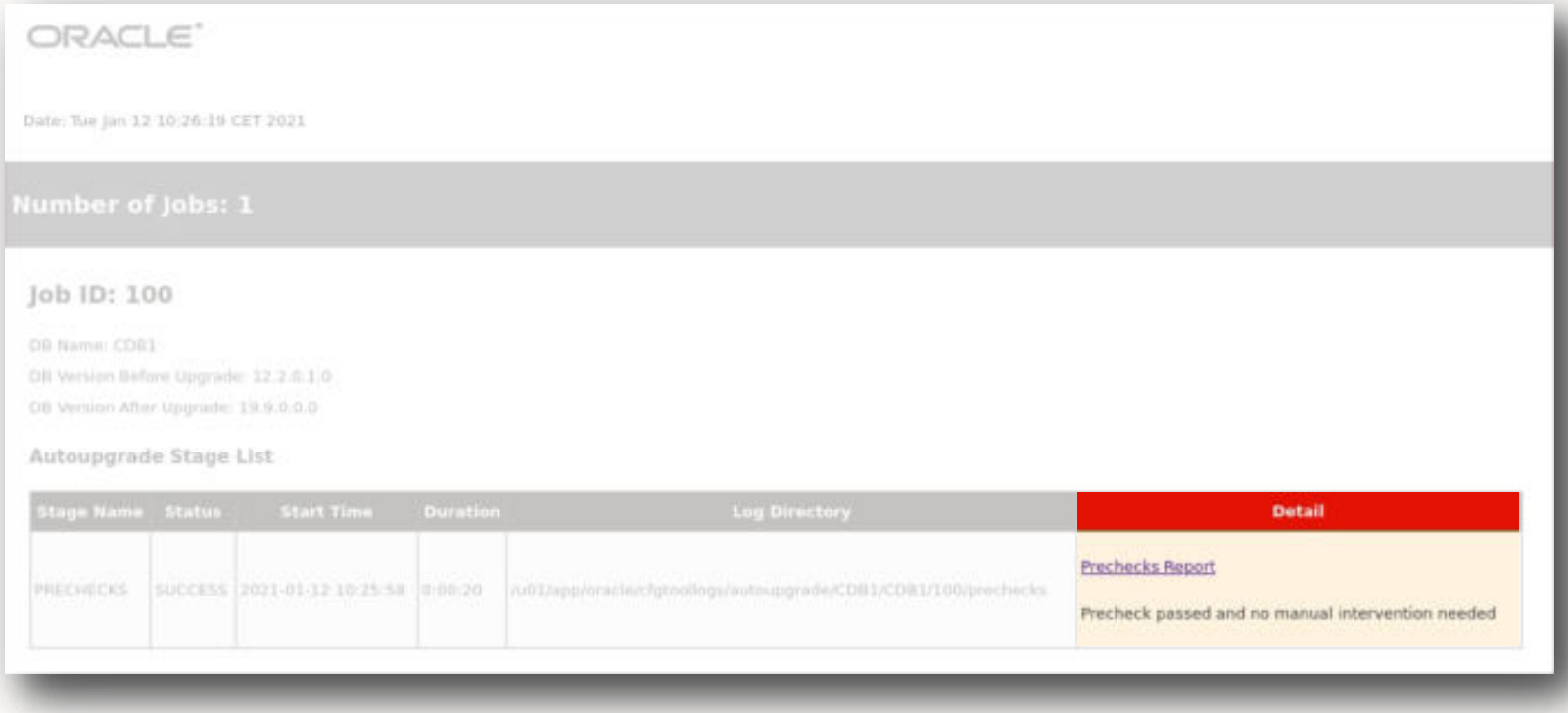
Summary report - text

```
=============================================
          Autoupgrade Summary Report
=============================================
[Date]           Tue Jan 12 10:26:19 CET 2021
[Number of Jobs] 1
=============================================
[Job ID] 100
=============================================
[DB Name]                   CDB1
[Version Before Upgrade]  12.2.0.1.0
[Version After Upgrade]   19.9.0.0.0
---------------------------------------------
[Stage Name]    PRECHECKS
[Status]        SUCCESS
[Start Time]    2021-01-12 10:25:58
[Duration]      0:00:20
[Log Directory] /u01/app/oracle/upg/CDB1/100/prechecks
[Detail]        /u01/app/oracle/upg/CDB1/100/prechecks/cdb1_preupgrade.log
                Precheck passed and no manual intervention needed
---------------------------------------------
```

AutoUpgrade | Essentials

- Download
- Configure
- Analyze
- **Check**
- Upgrade

Summary report - HTML


ORACLE

Date: Tue Jan 12 10:26:19 CET 2021

Number of Jobs: 1

Job ID: 100

DB Name: CDB1

DB Version Before Upgrade: 12.2.0.1.0

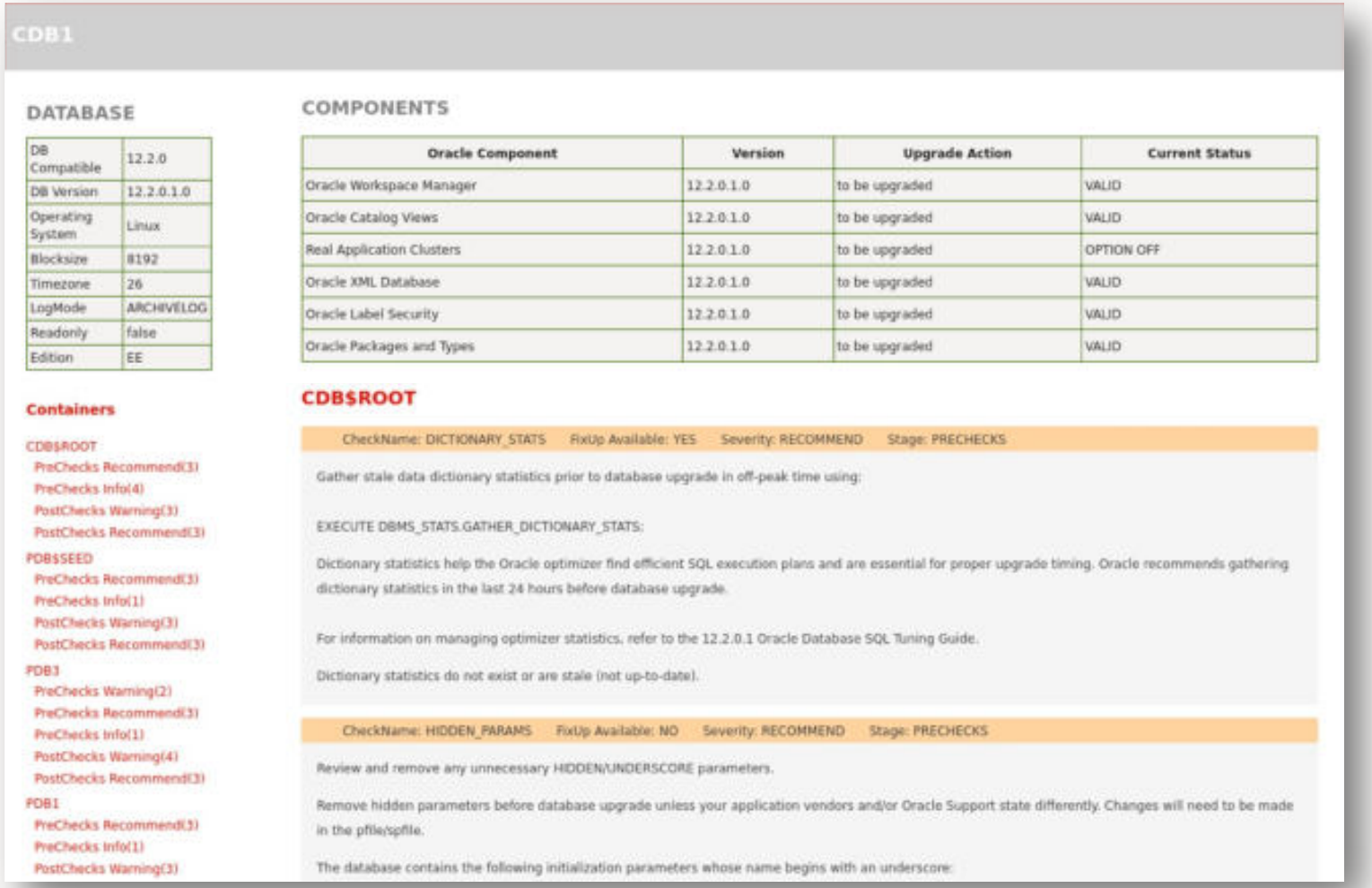
DB Version After Upgrade: 19.9.0.0.0

Autoupgrade Stage List

Stage Name	Status	Start Time	Duration	Log Directory	Detail
PRECHECKS	SUCCESS	2021-01-12 10:25:58	0:00:20	/u01/app/oracle/cfgtoollogs/autoupgrade/CDB1/CDB1/100/prechecks	Prechecks Report Precheck passed and no manual intervention needed

AutoUpgrade | Essentials

- Download
- Configure
- Analyze
- **Check**
- Upgrade



CDB1

DATABASE

DB Compatible	12.2.0
DB Version	12.2.0.1.0
Operating System	Linux
Blocksize	8192
Timezone	26
LogMode	ARCHIVELOG
Readonly	false
Edition	EE

Containers

CDB$ROOT
PreChecks Recommend(3)
PreChecks Info(4)
PostChecks Warning(3)
PostChecks Recommend(3)
PDB$SEED
PreChecks Recommend(3)
PreChecks Info(1)
PostChecks Warning(3)
PostChecks Recommend(3)
PDB3
PreChecks Warning(2)
PreChecks Recommend(3)
PreChecks Info(1)
PostChecks Warning(4)
PostChecks Recommend(3)
PDB1
PreChecks Recommend(3)
PreChecks Info(1)
PostChecks Warning(3)

COMPONENTS

Oracle Component	Version	Upgrade Action	Current Status
Oracle Workspace Manager	12.2.0.1.0	to be upgraded	VALID
Oracle Catalog Views	12.2.0.1.0	to be upgraded	VALID
Real Application Clusters	12.2.0.1.0	to be upgraded	OPTION OFF
Oracle XML Database	12.2.0.1.0	to be upgraded	VALID
Oracle Label Security	12.2.0.1.0	to be upgraded	VALID
Oracle Packages and Types	12.2.0.1.0	to be upgraded	VALID

CDB$ROOT

CheckName: DICTIONARY_STATS FixUp Available: YES Severity: RECOMMEND Stage: PRECHECKS

Gather stale data dictionary statistics prior to database upgrade in off-peak time using:

EXECUTE DBMS_STATS.GATHER_DICTIONARY_STATS;

Dictionary statistics help the Oracle optimizer find efficient SQL execution plans and are essential for proper upgrade timing. Oracle recommends gathering dictionary statistics in the last 24 hours before database upgrade.

For information on managing optimizer statistics, refer to the 12.2.0.1 Oracle Database SQL Tuning Guide.

Dictionary statistics do not exist or are stale (not up-to-date).

CheckName: HIDDEN_PARAMS FixUp Available: NO Severity: RECOMMEND Stage: PRECHECKS

Review and remove any unnecessary HIDDEN/UNDERSCORE parameters.

Remove hidden parameters before database upgrade unless your application vendors and/or Oracle Support state differently. Changes will need to be made in the pfile/spfile.

The database contains the following initialization parameters whose name begins with an underscore:

AutoUpgrade | Essentials

Download
Configure
Analyze
Check
Upgrade

Preupgrade report comes in:

- HTML
- Text
- JSON

AutoUpgrade | Essentials

- Download
- Configure
- Analyze
- Check
- **Upgrade**

Upgrade

```
$ java -jar autoupgrade.jar -config CDB1.cfg -mode deploy
```

Have a cup of coffee and wait, or …

AutoUpgrade | Essentials

- Download
- Configure
- Analyze
- Check
- **Upgrade**

Monitor

```
upg> lsj

+----+-------+---------+---------+-------+--------------+--------+---------------+
|Job#|DB_NAME|    STAGE|OPERATION| STATUS|    START_TIME| UPDATED|        MESSAGE|
+----+-------+---------+---------+-------+--------------+--------+---------------+
| 101|   CDB1|PREFIXUPS|EXECUTING|RUNNING|20/11/24 13:38|13:39:26|Remaining 12/13|
+----+-------+---------+---------+-------+--------------+--------+---------------+
```

AutoUpgrade | Essentials

- Download
- Configure
- Analyze
- Check
- **Upgrade**

All the details

```
upg> status -job 101

Progress
-----------------------------------
Start time:      20/11/24 13:38
Elapsed (min):   13
Last update:     2020-11-24T13:48:52.139
Stage:           DBUPGRADE
Operation:       EXECUTING
Status:          RUNNING
Stage summary:
    SETUP             <1 min
    GRP               <1 min
    PREUPGRADE        <1 min
    PRECHECKS         <1 min
    PREFIXUPS         8 min
    DRAIN             <1 min
    DBUPGRADE         3 min (IN PROGRESS)

Job Logs Locations
-----------------------------------
Logs Base:    /home/oracle/autoupg_default/CDB1/CDB1
Job logs:     /home/oracle/autoupg_default/CDB1/CDB1/101
Stage logs:   /home/oracle/autoupg_default/CDB1/CDB1/101/dbupgrade
TimeZone:     /home/oracle/autoupg_default/CDB1/CDB1/temp
```

AutoUpgrade | Essentials

- Download
- Configure
- Analyze
- Check
- **Upgrade**

All the details - continued

```
...

Additional information
----------------------------------
Details:
[Upgrading] is [0%] completed for [cdb1-cdb$root]
                 +---------+---------------+
                 |CONTAINER|     PERCENTAGE|
                 +---------+---------------+
                 | CDB$ROOT|   UPGRADE[12%]|
                 | PDB$SEED|UPGRADE PENDING|
                 |     PDB3|UPGRADE PENDING|
                 +---------+---------------+

Error Details:
None
```

AutoUpgrade | Essentials

- Download
- Configure
- Analyze
- Check
- **Upgrade**

Success

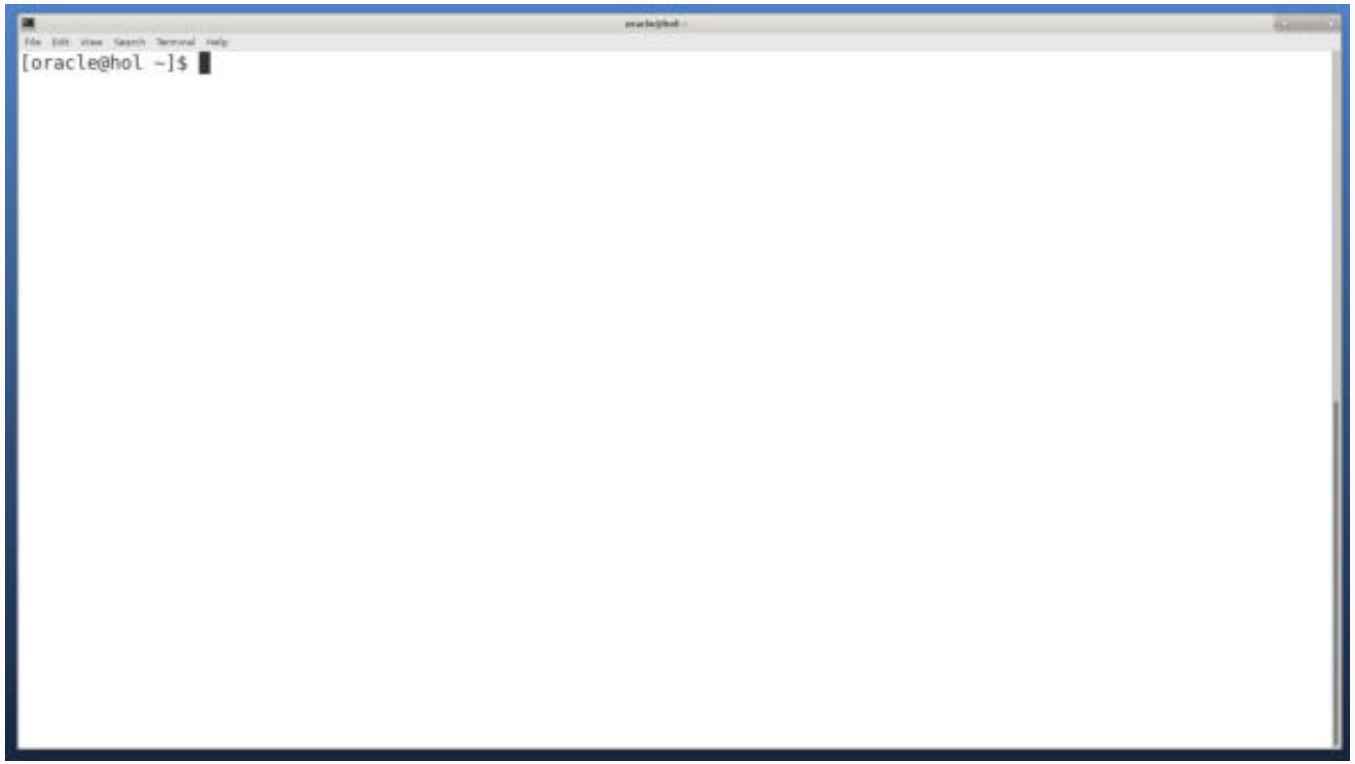
```
upg> Job 101 completed
------------------- Final Summary --------------------
Number of databases            [ 1 ]

Jobs finished successfully     [1]
Jobs failed                    [0]
Jobs pending                   [0]
------------- JOBS FINISHED SUCCESSFULLY -------------
Job 101 for CDB1

---- Drop GRP at your convenience once you consider it is no longer needed ----
Drop GRP from CDB1: drop restore point AUTOUPGRADE_9212_CDB1122010
```

And it includes:

- Recompilation (utlrp.sql)
- Time zone file upgrade
- Postupgrade fixups
- … and so much more

AutoUpgrade | Essentials

Download
Configure
Analyze
Check
Upgrade



Watch on YouTube

AutoUpgrade Advanced Options

Photo by Ciprian Boiciuc on Unsplash

AutoUpgrade | Advanced Options

Many Databases

- Different Servers
- PFILE
- Shell Scripts
- Restore Point
- Underscores
- Recompilation
- Time Zone
- Parallel
- Monitoring

Upgrade one or many databases

One

```
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=CDB1
```

Many

```
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=CDB1

upg2.source_home=/u01/app/oracle/product/11.2.0.4
upg2.target_home=/u01/app/oracle/product/19
upg2.sid=DB11204

...

upgn.source_home=/u01/app/oracle/product/12.1.0.2
upgn.target_home=/u01/app/oracle/product/19
upgn.sid=HR
```

Pro tip: You can also start multiple instances of AutoUpgrade at the same time

AutoUpgrade | Advanced Options

- Many Databases
- **Different Servers**
- PFILE
- Shell Scripts
- Restore Point
- Underscores
- Recompilation
- Time Zone
- Parallel
- Monitoring

Upgrade only when `upgrade_node` matches hostname

```
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.upgrade_node=test_server01.mycorp.net
upg1.sid=CDB1

upg2.source_home=/u01/app/oracle/product/12.2.0.1
upg2.target_home=/u01/app/oracle/product/19
upg2.upgrade_node=prod_server01.mycorp.net
upg2.sid=CDB2
```

- Database `upg1` will only be upgraded when AutoUpgrade gets executed on server `test_server01.mycorp.net`

AutoUpgrade | Advanced Options

Many Databases
Different Servers
PFILE
Shell Scripts
Restore Point
Underscores
Recompilation
Time Zone
Parallel
Monitoring

Update initialization parameters as part of the upgrade

You can:

- Add or remove parameters
- Before, during or after upgrade
- For a single or every database

AutoUpgrade | Advanced Options

- Many Databases
- Different Servers
- **PFILE**
- Shell Scripts
- Restore Point
- Underscores
- Recompilation
- Time Zone
- Parallel
- Monitoring

Remove a parameter during a specific upgrade

```
upg1.del_during_upgrade_pfile=/home/oracle/global_del_during.ora
```

```
# Example: global_del_during.ora

optimizer_features_enable
```

Add parameters to all databases after upgrade

```
global.add_after_upgrade_pfile=/home/oracle/global_add_after.ora
```

```
# Example: global_add_after.ora

deferred_segment_creation=false
_cursor_obsolete_threshold=1024
_sql_plan_directive_mgmt_control=0
_use_single_log_writer=true
```

AutoUpgrade | Advanced Options

- Many Databases
- Different Servers
- **PFILE**
- Shell Scripts
- Restore Point
- Underscores
- Recompilation
- Time Zone
- Parallel
- Monitoring

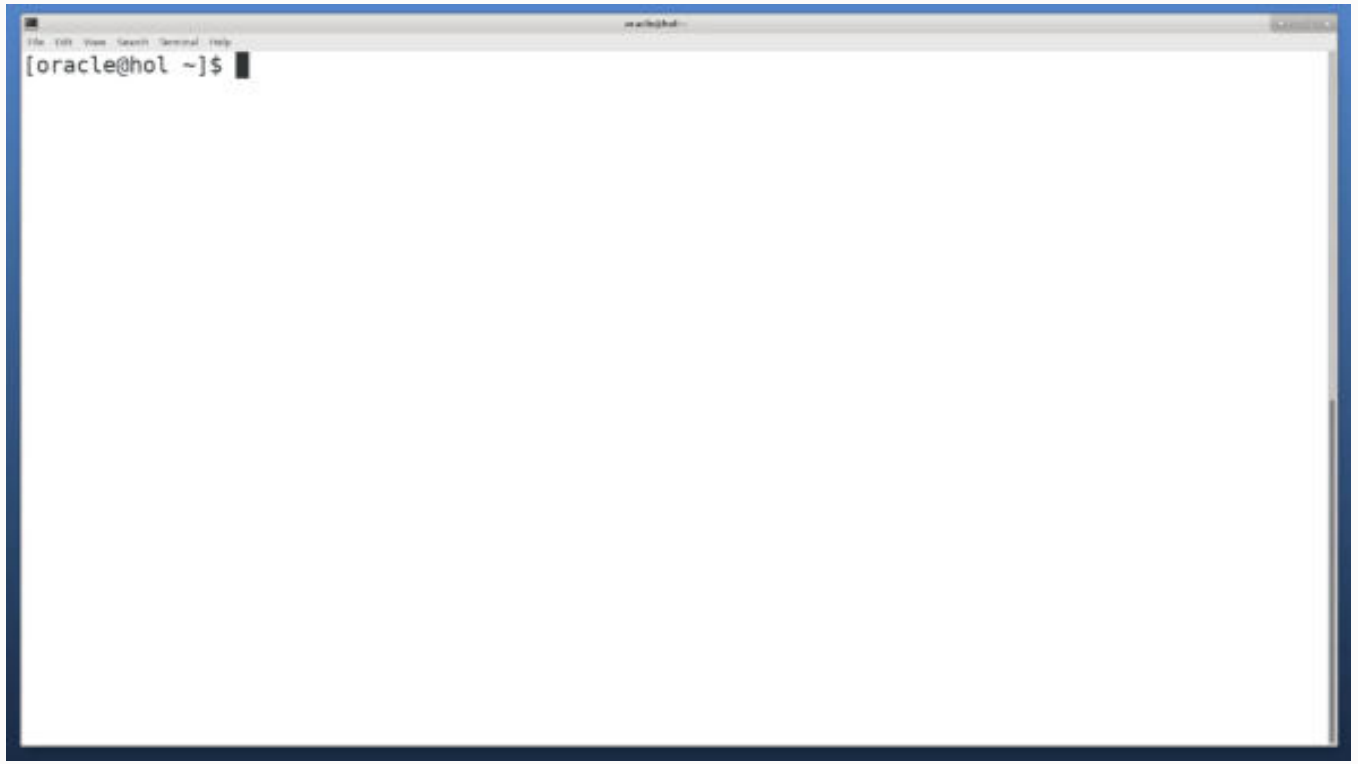
Batch-update parameters

```
global.del_during_upgrade_pfile=/home/oracle/global_del_during.ora
global.add_during_upgrade_pfile=/home/oracle/global_add_during.ora
global.del_after_upgrade_pfile=/home/oracle/global_del_during.ora
global.add_after_upgrade_pfile=/home/oracle/global_add_after.ora

upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=CDB1
upg1.add_after_upgrade_pfile=/home/oracle/upg1_add_after.ora

upg2.source_home=/u01/app/oracle/product/12.2.0.1
upg2.target_home=/u01/app/oracle/product/19
upg2.sid=CDB2
upg2.add_after_upgrade_pfile=/home/oracle/upg2_add_after.ora
```

```
# Example: global_add_after.ora

deferred_segment_creation=false
_cursor_obsolete_threshold=1024
_sql_plan_directive_mgmt_control=0
_use_single_log_writer=true
```

AutoUpgrade | Advanced Options

- Many Databases
- Different Servers
- **PFILE**
- Shell Scripts
- Restore Point
- Underscores
- Recompilation
- Time Zone
- Parallel
- Monitoring



Watch on YouTube

AutoUpgrade | Advanced Options

- Many Databases
- Different Servers
- PFILE
- **Shell Scripts**
- Restore Point
- Underscores
- Recompilation
- Time Zone
- Parallel
- Monitoring

Execute your own scripts as part of the upgrade

You can:

- Before and after upgrade
- Halt or continue on error
- For a single or every database

Ideas:

- Enterprise Manager configuration
- Backup configuration
- Interact with apps using the database

AutoUpgrade | Advanced Options

- Many Databases
- Different Servers
- PFILE
- **Shell Scripts**
- Restore Point
- Underscores
- Recompilation
- Time Zone
- Parallel
- Monitoring

Shell script execution

```
global.before_action=/database/scripts/set_blackout.sh

upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=CDB1
upg1.after_action=/database/scripts/start_level0.sh
```

- Permitted extension options:
  - Unix shell (.sh)
  - Microsoft Windows batch (.bat, .cmd)
  - Microsoft Windows PowerShell (.ps1)

Pro tip: If you want script execution for all upgrades use `global.before_action` and `global.after_action`

AutoUpgrade | Advanced Options

- Many Databases
- Different Servers
- PFILE
- **Shell Scripts**
- Restore Point
- Underscores
- Recompilation
- Time Zone
- Parallel
- Monitoring

Shell script execution

Default - AutoUpgrade does not react on return code

```
upg1.before_action=/database/scripts/run_this_on_UPG1_before.sh
```

Optionally - AutoUpgrade halts on non-zero return code

```
upg1.before_action=/database/scripts/run_this_on_UPG1_before.sh Y
```

Pro tip: Script output is captured and stored in *preupgrade* and *postupgrade* directory

AutoUpgrade | Advanced Options

Many Databases
Different Servers
PFILE
Shell Scripts
Restore Point
Underscores
Recompilation
Time Zone
Parallel
Monitoring

Environment variables:

- ORACLE_SID
- ORACLE_UNQNAME
- ORACLE_BASE
- ORACLE_HOME
- TNS_ADMIN

AutoUpgrade | Advanced Options

- Many Databases
- Different Servers
- PFILE
- Shell Scripts
- **Restore Point**
- Underscores
- Recompilation
- Time Zone
- Parallel
- Monitoring

Guaranteed Restore Points

```
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=CDB1
upg1.restoration=no

upg2.source_home=/u01/app/oracle/product/12.2.0.1
upg2.target_home=/u01/app/oracle/product/19
upg2.sid=CDB2
upg2.drop_grp_after_upgrade=yes
```

- Default behavior:
  - AutoUpgrade creates GRP except for
    - Standard Edition 2
    - `restoration=no`
  - GRP will be kept
  - GRP needs to be removed manually except for
    - `drop_grp_after_upgrade=yes` will only remove it when upgrade completed successfully

AutoUpgrade | Advanced Options

- Many Databases
- Different Servers
- PFILE
- Shell Scripts
- Restore Point
- **Underscores**
- Recompilation
- Time Zone
- Parallel
- Monitoring

Underscore parameters and events

```
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=CDB1
upg1.remove_underscore_parameters=yes
```

- Default behavior:
  - Underscores and events will be kept

AutoUpgrade | Advanced Options

- Many Databases
- Different Servers
- PFILE
- Shell Scripts
- Restore Point
- Underscores
- **Recompilation**
- Time Zone
- Parallel
- Monitoring

AutoUpgrade recompiles invalid Oracle-maintained objects after the upgrade

- To postpone the recompilation:

```
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=CDB1
upg1.run_utlrp=no
```

AutoUpgrade | Advanced Options

- Many Databases
- Different Servers
- PFILE
- Shell Scripts
- Restore Point
- Underscores
- **Recompilation**
- Time Zone
- Parallel
- Monitoring

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

AutoUpgrade | Advanced Options

- Many Databases
- Different Servers
- PFILE
- Shell Scripts
- Restore Point
- Underscores
- **Recompilation**
- Time Zone
- Parallel
- Monitoring

```
$ 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
```

AutoUpgrade | Advanced Options

- Many Databases
- Different Servers
- PFILE
- Shell Scripts
- Restore Point
- Underscores
- **Recompilation**
- Time Zone
- Parallel
- Monitoring

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

AutoUpgrade | Advanced Options

- Many Databases
- Different Servers
- PFILE
- Shell Scripts
- Restore Point
- Underscores
- Recompilation
- **Time Zone**
- Parallel
- Monitoring

Skip time zone upgrade

```
upg1.source_home=/u01/app/oracle/product/12.2.0.1
upg1.target_home=/u01/app/oracle/product/19
upg1.sid=CDB1
upg1.timezone_upg=no
```

- Default behavior:
  - Time zone adjustment happens post upgrade
  - Database will be restarted several times
  - Important when you use "Downgrade" as fallback strategy as time zone can't be downgraded

AutoUpgrade | Advanced Options

- Many Databases
- Different Servers
- PFILE
- Shell Scripts
- Restore Point
- Underscores
- Recompilation
- Time Zone
- **Parallel**
- Monitoring

CDB

```
upg1.catctl_options=-n 64 -N 8
```

-n	Total number of parallel processes (min 4, max unlimited, default `CPU_COUNT`)
-N	Number of parallel processes per PDB (min 1, max 8, default 2)

Concurrent PDB upgrades: n / N

AutoUpgrade | Advanced Options

- Many Databases
- Different Servers
- PFILE
- Shell Scripts
- Restore Point
- Underscores
- Recompilation
- Time Zone
- Parallel
- **Monitoring**

Monitor via browser:

`<au_global_log_dir>/cfgtoollogs/upgrade/auto/state.html`

Refreshes automatically

Photo by Joshua Fernandez on Unsplash

Compatible

Compatible | Recommendation

When should you change `COMPATIBLE`?

A week or two after the upgrade - requires a database restart

Caution: When you change `COMPATIBLE` you can't:

- Flashback to restore point
- Downgrade

Compatible | Recommendation

Which value should you use for `COMPATIBLE`?

- The default of the database release
- 11.2.0
- 12.1.0
- 12.2.0
- 18.0.0
- 19.0.0

Should you change `COMPATIBLE` when patching?

- NEVER!
  - Except for …

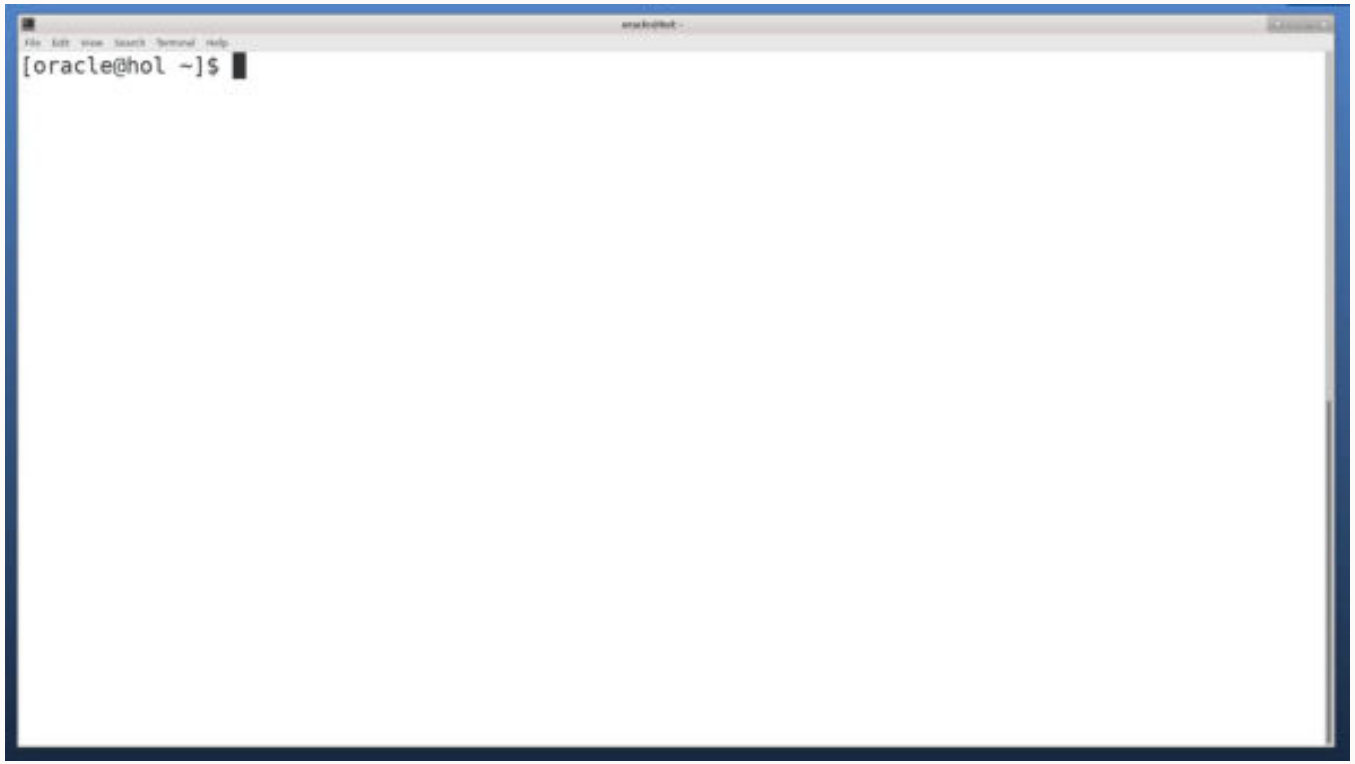
Compatible | AutoUpgrade

AutoUpgrade does not change `COMPATIBLE`

Unless you want it

```
upg1.drop_grp_after_upgrade=yes
upg1.raise_compatible=yes
```


Compatible | Demo



Watch on YouTube

AGENDA

10:00	Introduction
10:15	Upgrade to Oracle Database 19c
10:45	Break
11:00	Upgrade to Oracle Database 19c
12:00	**Lunch**
13:00	Ensure Performance Stability
14:00	Break
14:15	Hands-On Lab
16:00	End

AGENDA

10:00	Introduction
10:15	Upgrade to Oracle Database 19c
10:45	Break
11:00	Upgrade to Oracle Database 19c
12:00	Lunch
13:00	**Ensure Performance Stability**
14:00	Break
14:15	Hands-On Lab
16:00	End

“Help me - I have an upgrade problem ...”

In

95%

of all cases, "upgrade problem" in reality is a performance issue after upgrade. Or not database related.

There is exactly one way to mitigate the risk.

TESTING!

Photo by nikko macaspac on Unsplash

Testing | Typical Mistakes

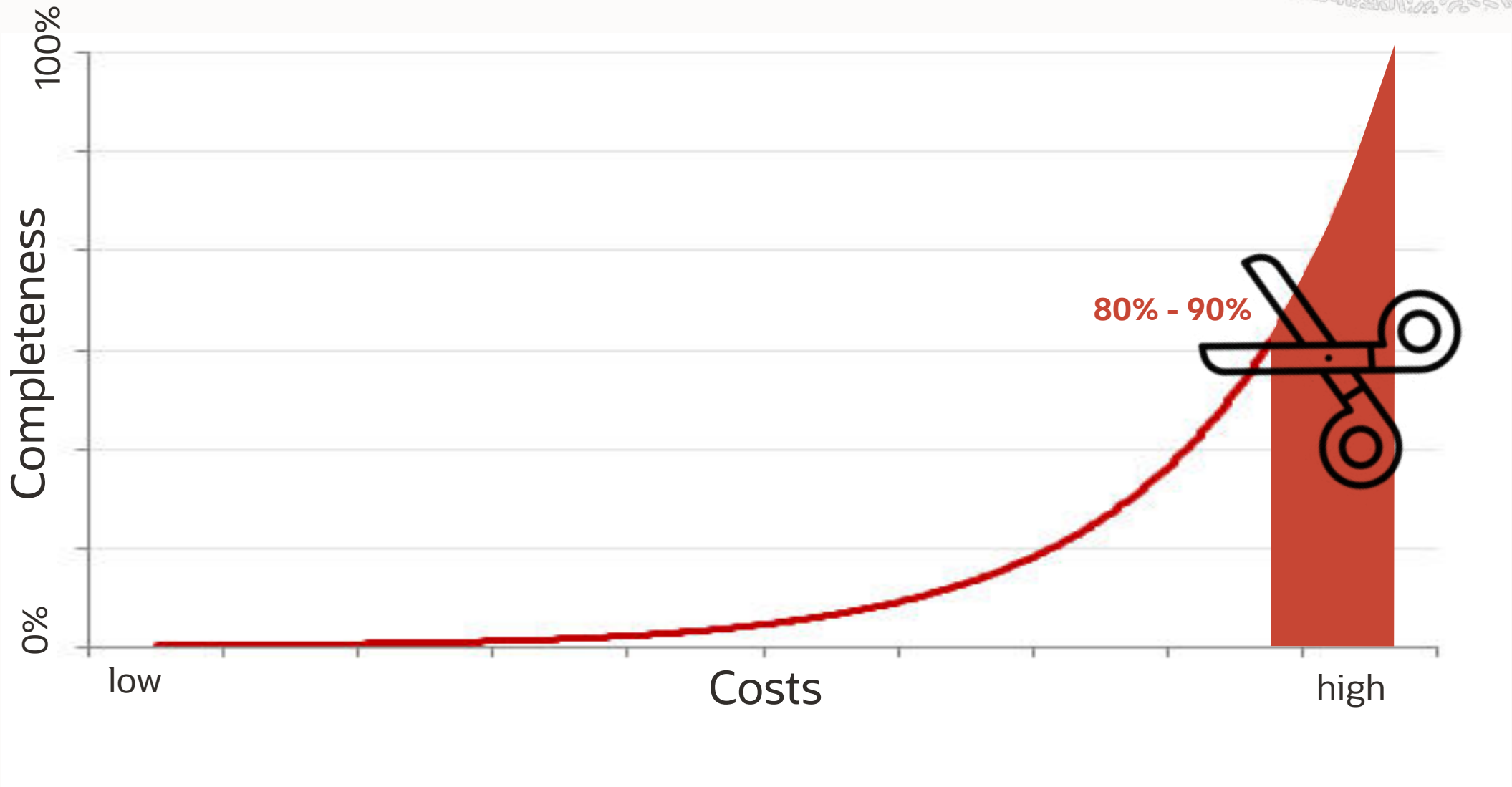
Only 10% of real data used
Artificially created data sets
Outdated data
Tests done on a laptop
No testing tools used
No stale statistics refreshed

”
Testing?? What a waste of time!

- *Real* experts fix it *after* go-live …

Photo by Elijah O'Donnell on Unsplash

Testing | Completeness versus Costs



General Performance Best Practices

Photo by Yannis Zaugg on Unsplash

Parameters | General Recommendations

Default

Deprecated/desupported

Underscores/events

Applications

The fewer parameters, the better

```
SQL> select name, value
     from v$parameter
     where isdefault='FALSE';

 NAME                                 VALUE
_____________________________________ ____________________________________________________
_bug27355984_xt_preproc_timeout       1000
_cursor_obsolete_threshold            1024
_exclude_seed_cdb_view                FALSE
_optimizer_aggr_groupby_elim          FALSE
_use_single_log_writer                TRUE
audit_file_dest                       /u01/app/oracle/admin/CDB2/adump
audit_trail                           NONE
compatible                            19.0.0
control_files                         /u02/fast_recovery_area/CDB2/control02.ctl
```

Parameters | General Recommendations

- Default
- **Deprecated/desupported**
- Underscores/events
- Applications

```
SQL> startup
ORA-32004: obsolete or deprecated parameter(s) specified for RDBMS instance
ORACLE instance started.

Total System Global Area       1577055360 bytes
Fixed Size                        9135232 bytes
Variable Size                   385875968 bytes
Database Buffers               1174405120 bytes
Redo Buffers                      7639040 bytes
Database mounted.
Database opened.
```

Pro tip: The Upgrade Guide contains a list of deprecated and desupported parameters

Parameters | General Recommendations

Default

Deprecated/desupported

Underscores/events

Applications

Use

- as few as possible
- not longer than needed

```
SQL> select name, value
     from v$parameter
     where substr(name, 0, 1) = '_' or name='event';
```

Create plan for remove it again

Pro tip: During upgrade it is recommended to remove all underscores and events

Parameters | General Recommendations

Default
Deprecated/desupported
Underscores/events
Applications

Follow application specific recommendations

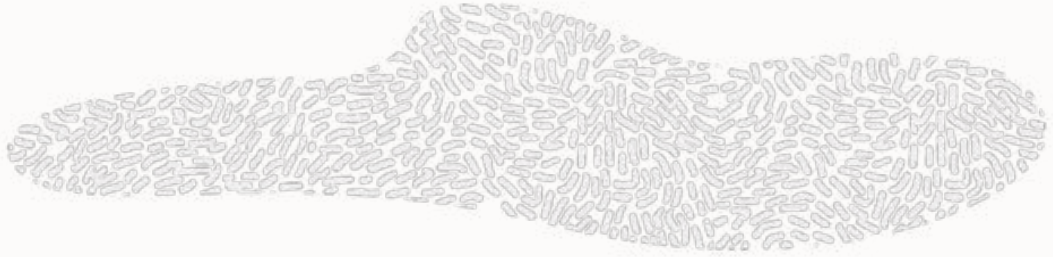
- E-Business Suite
- Siebel
- …

Database Initialization Parameters for Oracle E-Business Suite Release 12 (Doc ID 396009.1)

In This Document

The most current version of this document can be obtained in My Oracle Support Document 396009.1.

Parameters | Tracking Your Changes



Never implement a change without a comment

```
SQL> alter system set
     "_cursor_obsolete_threshold"=1024
     comment='04-03-2021 Daniel: MOS 2431353.1, evaluate after upgrade'
     scope=both;
```

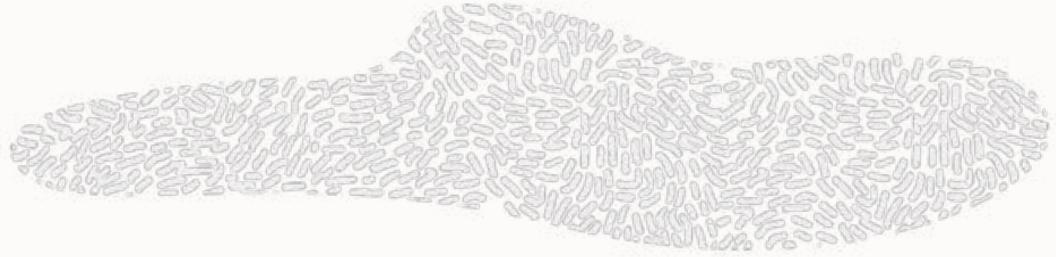
Or, in your PFile

```
*._cursor_obsolete_threshold=1024#04-03-2021 Daniel: MOS 2431353.1, evaluate after upgrade
```

View your comments

```
SQL> select value, update_comment from v$parameter where name='_cursor_obsolete_threshold';

VALUE    UPDATE_COMMENT
________ ____________________________________________________
1024     04-03-2021 Daniel: MOS 2431353.1, evaluate after upgrade
```

Parameters
COMPATIBLE vs OPTIMIZER_FEATURES_ENABLE



Fully independent from each other

- `COMPATIBLE`
  - Enables features
  - Always use the default value 19.0.0 in Oracle 19c
- `OPTIMIZER_FEATURES_ENABLE`
  - Just reverts to the parameters used in a previous release
  - Avoid using it if possible
  - This is not a Swiss Army knife!
  - You will turn off a lot of great features

> *Modifying the OPTIMIZER_FEATURES_ENABLE parameter generally is strongly discouraged and should only be used as a short term measure at the suggestion of Oracle Global Support.*

Use Caution if Changing the OPTIMIZER_FEATURES_ENABLE Parameter After an Upgrade (Doc ID 1362332.1)

Fix Control Persistence | DBMS_OPTIM_BUNDLE

Overview

Check

Enable

Output

Result

Info and Issues

Fix Control Persistence

- `DBMS_OPTIM_BUNDLE`
- Exists since 12.1.0.2 April 2017 (and earlier on Exadata)
- Idea:
  - Enable Optimizer behavior changing fixes at will
  - Fixes are installed but disabled by default

Fix Control Persistence | DBMS_OPTIM_BUNDLE

- Overview
- **Check**
- Enable
- Output
- Result
- Info and Issues

```
SQL> set serverout on
SQL> exec dbms_optim_bundle.GetBugsForBundle;

19.10.0.0.210119DBRU:
    Bug: 29487407,  fix_controls: 29487407
    Bug: 30998035,  fix_controls: 30998035
    Bug: 30786641,  fix_controls: 30786641
    Bug: 31444353,  fix_controls: 31444353
    Bug: 30486896,  fix_controls: 30486896
    Bug: 28999046,  fix_controls: 28999046
    Bug: 30902655,  fix_controls: 30902655
    Bug: 30681521,  fix_controls: 30681521
    Bug: 29302565,  fix_controls: 29302565
    Bug: 30972817,  fix_controls: 30972817
...
```

Fix Control Persistence | DBMS_OPTIM_BUNDLE

- Overview
- Check
- **Enable**
- Output
- Result
- Info and Issues

```
begin
  dbms_optim_bundle.enable_optim_fixes(
   action                         => 'ON',
   scope                          => 'BOTH',
   current_setting_precedence => 'YES');
end;
/
```


Fix Control Persistence | DBMS_OPTIM_BUNDLE

- Overview
- Check
- Enable
- **Output**
- Result
- Info and Issues

```
1) Current _fix_control setting for spfile:
None

2) Final _fix_control setting for spfile considering current_setting_precedence
is YES
29331066:1  28965084:1  28776811:1  28498976:1  28567417:1  28558645:1
29132869:1  29450812:1  29687220:1  29304314:1  29930457:1  27261477:1
31069997:1  31077481:1  28602253:1  29653132:0  29937655:1  30347410:1
30602828:1  30896685:0  29487407:1  30998035:1  30786641:1  31444353:0
30486896:1  28999046:1  30902655:1  30681521:1  29302565:1  30972817:1
30222669:1  31668694:1  31001490:1  30198239:7  30980115:1  30616738:0
31895670:0  19138896:1  31670824:0   9876287:1  30564898:1  32075777:0
30570982:1

3) Current _fix_control setting in memory:
29331066:0  28965084:0  28776811:0  28498976:0  28567417:0  28558645:0
29132869:0  29450812:0  29687220:0  29304314:0  29930457:0  27261477:0
31069997:0  31077481:0  28602253:0  29653132:0  29937655:0  30347410:0
30602828:0  30896685:0  29487407:0  30998035:0  30786641:0  31444353:0
30486896:0  28999046:0  30902655:0  30681521:0  29302565:0  30972817:0
30222669:0  31668694:0  31001490:0  30198239:0  30980115:0  30616738:0
31895670:0  19138896:0  31670824:0   9876287:0  30564898:0  32075777:0
30570982:0
```

Fix Control Persistence | DBMS_OPTIM_BUNDLE

Overview
Check
Enable
Output
Result
Info and Issues

In the SPFILE:

```
*._fix_control='29331066:1','28965084:1','28776811:1','28498976:1','285
67417:1','28558645:1','29132869:1','29450812:1','29687220:1','29304314:
1','29930457:1','27261477:1','31069997:1','31077481:1','28602253:1','29
653132:0','29937655:1','30347410:1','30602828:1','30896685:0','29487407
:1','30998035:1','30786641:1','31444353:0','30486896:1','28999046:1','3
0902655:1','30681521:1','29302565:1','30972817:1','30222669:1','3166869
4:1','31001490:1','30198239:7','30980115:1','30616738:0','31895670:0','
19138896:1','31670824:0','9876287:1','30564898:1','32075777:0','3057098
2:1'#added through dbms_optim_bundle package
```

- Restart necessary to take effect

Fix Control Persistence | DBMS_OPTIM_BUNDLE

- Overview
- Check
- Enable
- Output
- Result
- **Info and Issues**

MOS Note: 21470071 - Automatic Fix Control Persistence

Blog Post: DBMS_OPTIM_BUNDLE Package

Blog Post: You may need a one-off in 19.10.0

- Oracle 19.10.0 – Patch 31862593 required
- Otherwise, the 19.10 fixes can't be enabled

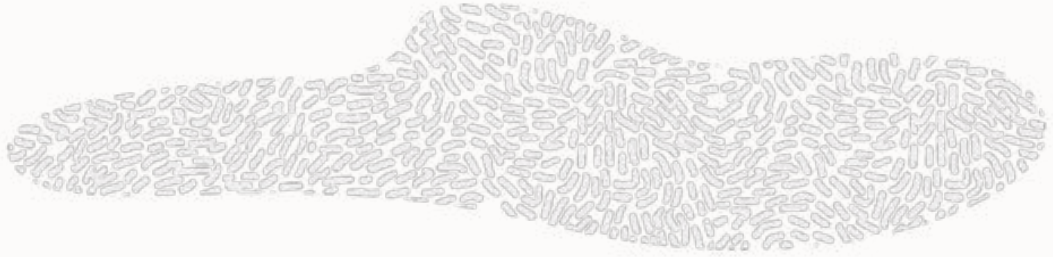
Blog Post: DBMS_OPTIM_BUNDLE is missing ... again?!

- Occasionally the package disappeared in several RUs

Photo by Veri Ivanova on Unsplash

Statistics

Dictionary Statistics | Overview



Statistics on SYS and other oracle maintained schemas

Gets executed by automatic optimizer statistics gathering

If disabled, consider instead to allow it to work only of dictionary stats

```
SQL> exec dbms_stats.set_global_prefs('autostats_target','oracle');
```

Dictionary Statistics | Gather

Refresh manually:

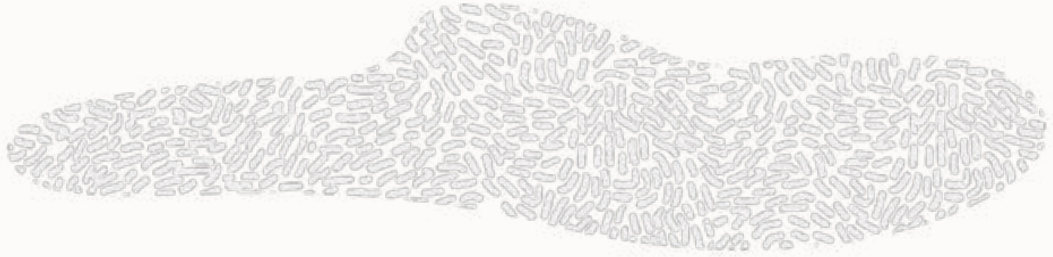
- Before and after upgrade
- Before (source) and after (target) logical migration
- After major application upgrades

Gather manually

```
SQL> BEGIN
        DBMS_STATS.GATHER_SCHEMA_STATS('SYS');
        DBMS_STATS.GATHER_SCHEMA_STATS('SYSTEM');
     END;
/
```

```
$ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl \
  -l /tmp \
  -b gatherstats -- \
  --x"begin dbms_stats.gather_schema_stats('SYS'); dbms_stats.gather_schema_stats('SYSTEM'); end;"
```

Fixed Objects Stats | Overview



> *After an upgrade, or after other database configuration changes, Oracle strongly recommends that you regather fixed object statistics after you have run representative workloads on Oracle Database.*

Database 19c Upgrade Guide, chapter 7

Never run it right after upgrade

Fixed Objects Stats | Definition

What is it?

```
SQL> SELECT owner, table_name
     FROM dba_tab_statistics
     WHERE object_type = 'FIXED TABLE';

OWNER    TABLE_NAME
________ _______________
SYS      X$KQFTA
SYS      X$KQFVI
SYS      X$KQFVT
SYS      X$KQFDT
SYS      X$KQFCO
SYS      X$KQFOPT
SYS      X$KYWMPCTAB
...
```

Pro tip: Dynamic statistics (sampling) are not used for X$ tables

Fixed Objects Stats | After Upgrade

Ask yourself: Do you remember this?

If not, DBMS_SCHEDULER to the rescue

Fixed Objects Stats | After Upgrade

1. Create a .sql script

```
BEGIN
   DBMS_SCHEDULER.CREATE_JOB (
      job_name => '"SYS"."GATHER_FIXED_OBJECTS_STATS_ONE_TIME"',
           job_type => 'PLSQL_BLOCK',
           job_action => 'BEGIN DBMS_STATS.GATHER_FIXED_OBJECTS_STATS; END;',
           start_date => SYSDATE+7,
           auto_drop => TRUE,
           comments => 'Gather fixed objects stats after upgrade - one time'
   );
   DBMS_SCHEDULER.ENABLE (
      name => '"SYS"."GATHER_FIXED_OBJECTS_STATS_ONE_TIME"'
   );
END;
/
```

Fixed Objects Stats | After Upgrade

2. Create a .sh script

```
$ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl \
   -n 4 -e \
   -C 'PDB$SEED' \
   -b sched_gfos -d /home/oracle/sched_gfos/ sched_gfos.sql
```

3. Execute .sh script after upgrade

```
upg1.after_action=/home/oracle/sched_gfos/sched_gfos.sh
```


Fixed Objects Stats | Other situations

Also gather fixed objects stats after:

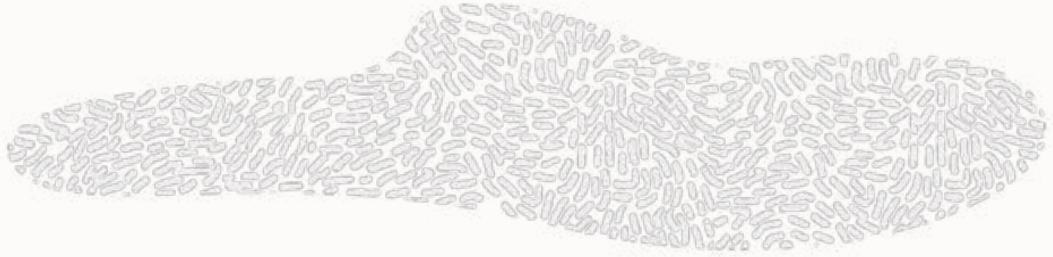
1. Major application upgrades
2. Using new functionality in the database
3. Major database configuration change

Always gather fixed objects stats when the system is warmed up - after your representative workload

Check out Best Practices for Gathering Optimizer Statistics with Oracle Database 19c

Pro tip: Automated stats gathering only gather fixed objects stats if they are completely missing

System Statistics | Overview

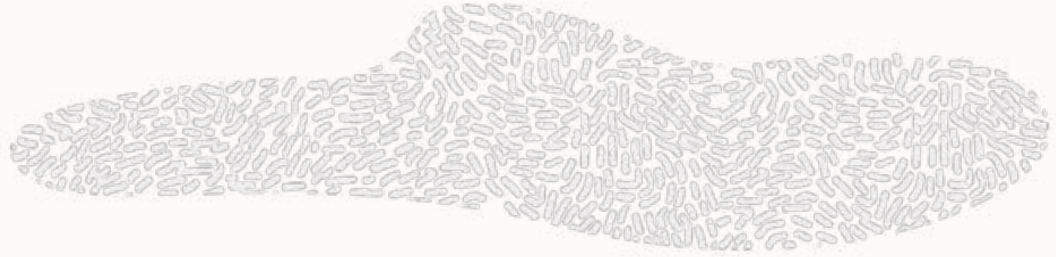


> *The system statistics describe hardware characteristics such as I/O and CPU performance and utilization.*
> *System statistics enable the query optimizer to more accurately estimate I/O and CPU costs when choosing execution plans.*

Database 19c SQL Tuning Guide, chapter 10

That sounds like a good idea

System Statistics | Recommendation



> *... in most cases you should use the defaults and not gather system statistics.*

Databases supporting a pure data warehouse workload on an Oracle Exadata Database Machine can benefit from system statistics gathered using the EXADATA option

... if the workload is mixed or you are not in a position to test the effect of using EXADATA system statistics, then stick to the defaults even on this platform.

Nigel Bayliss, Optimizer blog

System Statistics | Reference

To delete system statistics (and revert to defaults)

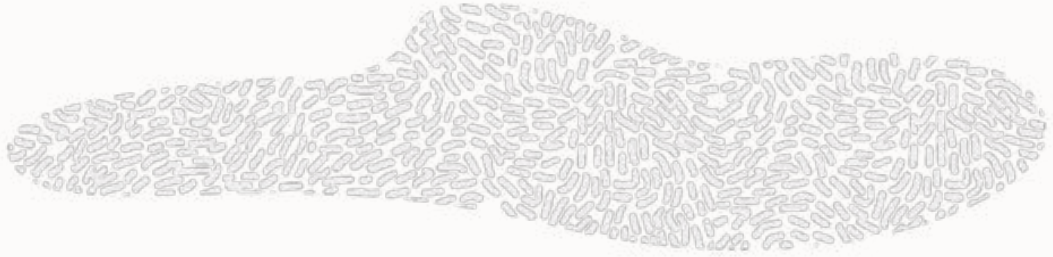
```
SQL> EXEC DBMS_STATS.DELETE_SYSTEM_STATS
```

References:

- Optimizer blog, Should You Gather System Statistics?
- SQL Tuning Guide, System Statistics
- SQL Tuning Guide, Guidelines for Gathering Optimizer Statistics Manually
- Database Performance Tuning Guide, Session and System Statistics

Performance Stability Prescription

SQL Tuning Set | Definition



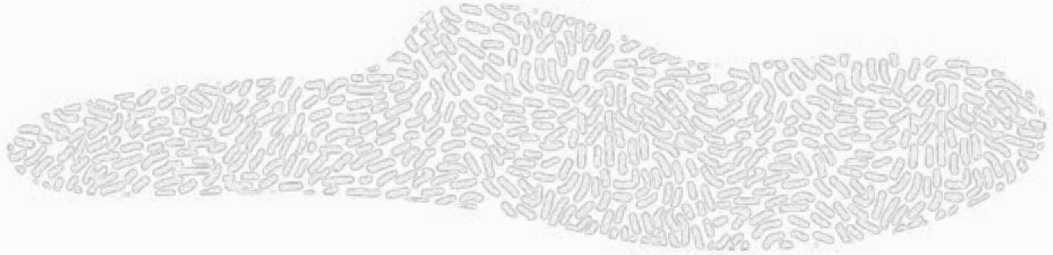
An SQL Tuning Set (STS) enables you to group SQL statements and related metadata in a single database object, which you can use to meet your tuning goals.

Specifically, SQL tuning sets achieve the following goals:

- *Providing input to the performance tuning advisors*
- *Transporting SQL between databases*

Database 19c SQL Tuning Guide, chapter 23

SQL Tuning Set | Definition

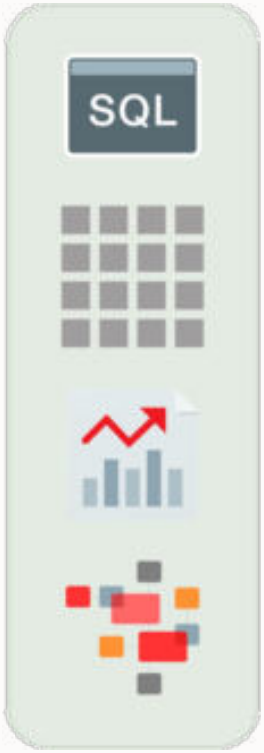


SQL statement

Context

Statistics

Plans



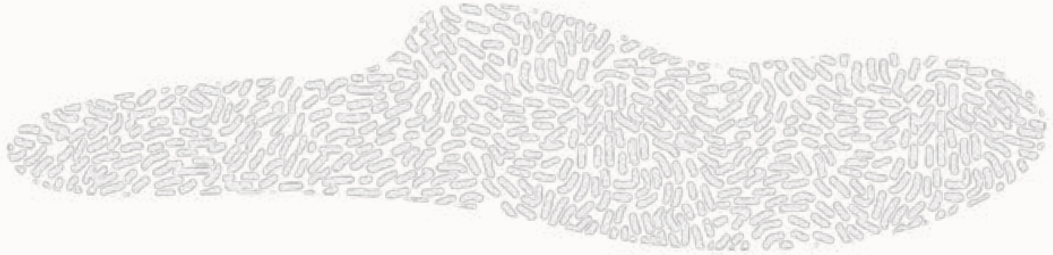
SQL Tuning Set | Create

First, create a SQL Tuning Set

```
SQL> BEGIN
  DBMS_SQLSET.CREATE_SQLSET (
    sqlset_name  => 'UPG_STS_1',
    description  => 'For upgrade - from source'
  );
END;
/
```

Pro tip: You can also use DBMS_SQLTUNE to create a SQL Tuning Set

SQL Tuning Set | Capture



Next, capture statements from AWR

```
SQL> DECLARE
   begin_id number;
   end_id number;
   cur sys_refcursor;
BEGIN
   SELECT min(snap_id), max(snap_id) INTO begin_id, end_id
   FROM dba_hist_snapshot;

open cur for
  select value(p) from table(dbms_sqltune.select_workload_repository(
       begin_snap       => begin_id,
       end_snap         => end_id,
       basic_filter     => 'parsing_schema_name not in (''SYS'')',
       ranking_measure1 => 'elapsed_time',
       result_limit     => 5000,
       attribute_list   => 'ALL')) p;

  dbms_sqltune.load_sqlset('UPG_STS_1', cur);

close cur;

END;
/
```

Pro tip: Consider excluding other internal schemas like *DBSNMP*, *ORACLE_OCM*, *LBACSYS*, *WMSYS*, *XDB*, *SYSTEM*

SQL Tuning Set | Capture

Optionally, capture statements from cursor cache

```
SQL> BEGIN
   DBMS_SQLSET.CAPTURE_CURSOR_CACHE_SQLSET(
      sqlset_name     => 'UPG_STS_1',
      time_limit      => 900,
      repeat_interval => 60,
      capture_option  => 'MERGE',
      capture_mode    => DBMS_SQLTUNE.MODE_ACCUMULATE_STATS,
      basic_filter    => 'parsing_schema_name not in (''SYS'')',
      sqlset_owner    => NULL,
      recursive_sql   => 'HAS_RECURSIVE_SQL');
   END;
/
```

Careful - puts load on your system

Pro tip: SQL Tuning Guide shows how to load all statements from a given schema

SQL Tuning Set | Transport

Pack into staging table on source database

```
SQL> BEGIN
   DBMS_SQLTUNE.CREATE_STGTAB_SQLSET (
      table_name          => 'UPG_STGTAB_1');
   DBMS_SQLTUNE.PACK_STGTAB_SQLSET (
      sqlset_name         => 'UPG_STS_1',
      staging_table_name  => 'UPG_STGTAB_1');
END;
```

Optionally, use `DBMS_SQLTUNE.REMAP_STGTAB_SQLSET` to remap between `CON_DBID`

Export with Data Pump

```
$ expdp user \
     directory=mydirectory
     dumpfile=upg_stgtab_1.dmp
     tables=UPG_STGTAB_1
```

SQL Tuning Set | Transport

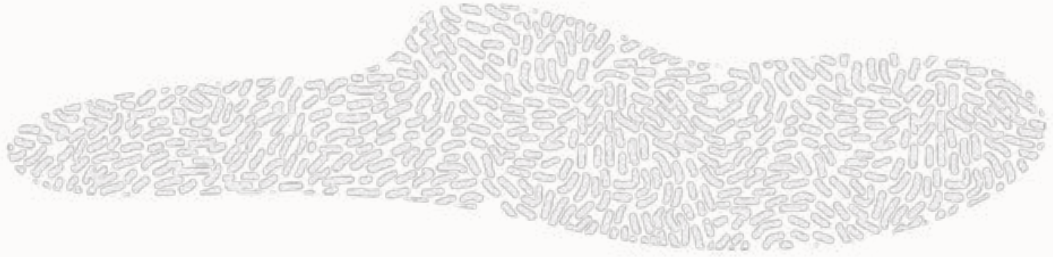
Import with Data Pump to target database

```
$ impdp user \
     directory=mydirectory
     dumpfile=upg_stgtab_1.dmp
     tables=UPG_STGTAB_1
```

Unpack staging table

```
SQL> BEGIN
   DBMS_SQLTUNE.UNPACK_STGTAB_SQLSET (
      sqlset_name         => '%',
      replace             => true,
      staging_table_name => 'UPG_STGTAB_1'
   );
END;
/
```

SQL Tuning Set | License



SQL Tuning Sets can also be accessed by way of database server APIs and command-line interfaces. Usage of any subprograms in the DBMS_SQLSET package to manage SQL Tuning Sets is part of the EE and EE-ES offerings.

In addition, the following subprograms, part of the DBMS_SQLTUNE package, provide an older interface to manage SQL Tuning Sets and are also part of the EE and EE-ES offerings:

ADD_SQLSET_REFERENCE
CREATE_STGTAB_SQLSET
LOAD_SQLSET
SELECT_CURSOR_CACHE
UNPACK_STGTAB_SQLSET
CAPTURE_CURSOR_CACHE_SQLSET
DELETE_SQLSET
PACK_STGTAB_SQLSET
SELECT_SQLSET
UPDATE_SQLSET
CREATE_SQLSET
DROP_SQLSET
REMOVE_SQLSET_REFERENCE
SELECT_WORKLOAD_REPOSITORY

Database 19c Database Licensing Information User Manual

SQL Tuning Set | Recommendation

Always capture workload data into SQL Tuning Sets

Collect at least
one full month of workload data
before any upgrade or migration

Workload Information

AWR – Automatic Workload Repository

Change the retention to a minimum of 40 days

```
exec
dbms_workload_repository.modify_snapshot_settings(
retention=>57600, interval=>30);
```

Collect SQL statements and plans

Use AWR as main source

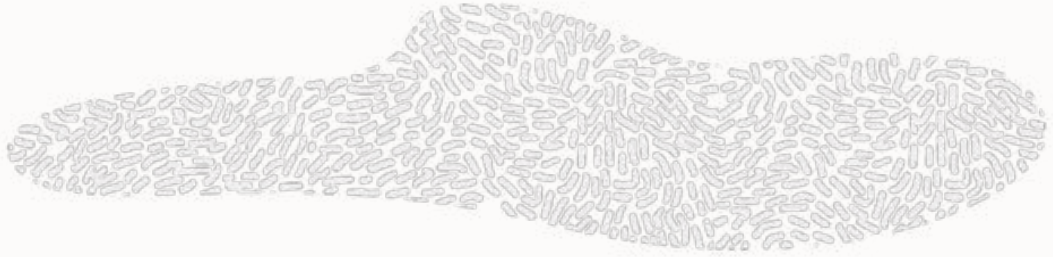
Capture from Cursor Cache for OLTP

Collect statements, plans and stats in SQL Tuning Sets

Performance Stability Prescription

1. Collect
2. **Compare**
3. Analyze
4. Tune
5. Manage
6. Test

AWR | Diff Report



Compare AWR report
from two different periods

1. AWR snapshot
2. Execute workload
3. AWR snapshot

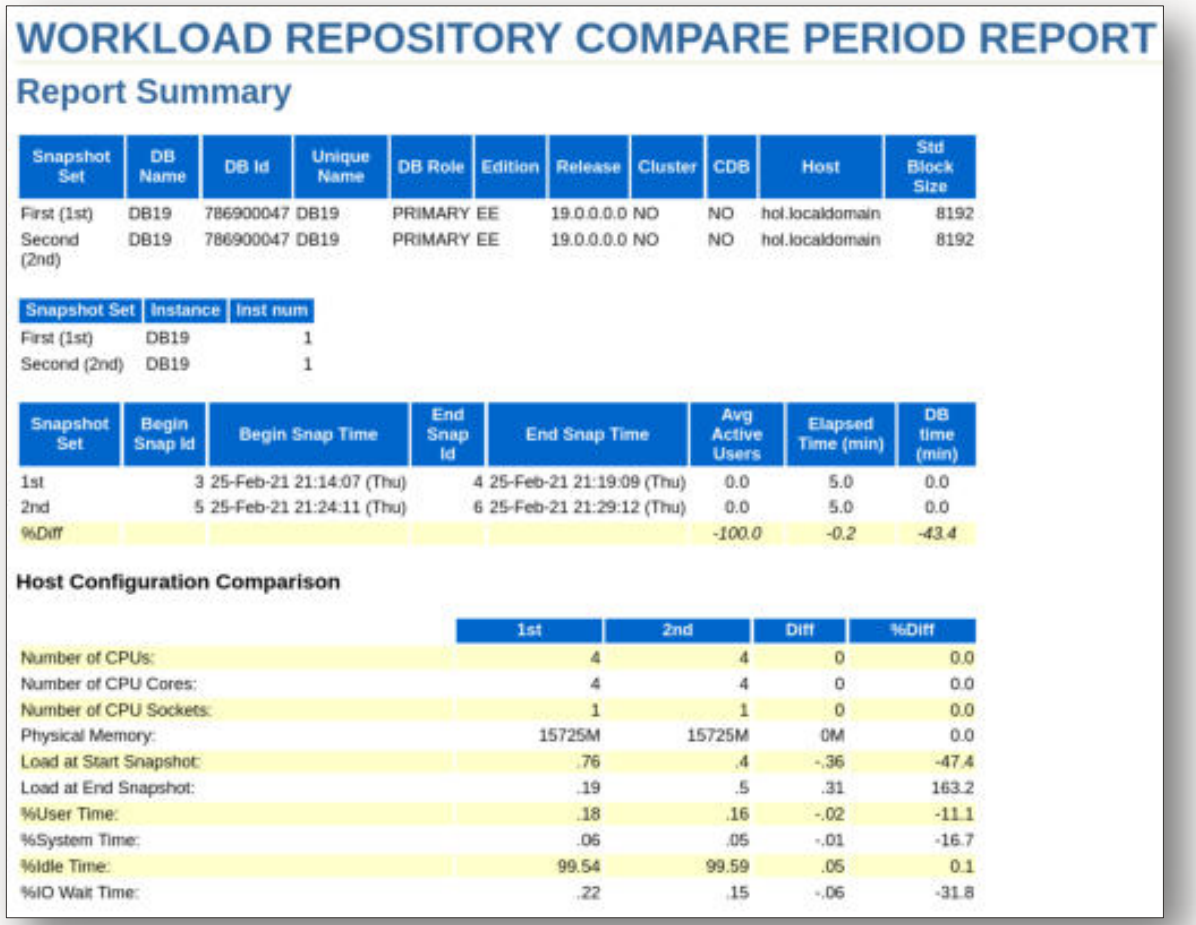
4. Upgrade

5. AWR snapshot
6. Execute workload
7. AWR snapshot

8. Compare

AWR | Diff Report

Use script `awrddrpt.sql`



WORKLOAD REPOSITORY COMPARE PERIOD REPORT

Report Summary

Snapshot Set	DB Name	DB Id	Unique Name	DB Role	Edition	Release	Cluster	CDB	Host	Std Block Size
First (1st)	DB19	786900047	DB19	PRIMARY	EE	19.0.0.0.0	NO	NO	hol.localdomain	8192
Second (2nd)	DB19	786900047	DB19	PRIMARY	EE	19.0.0.0.0	NO	NO	hol.localdomain	8192

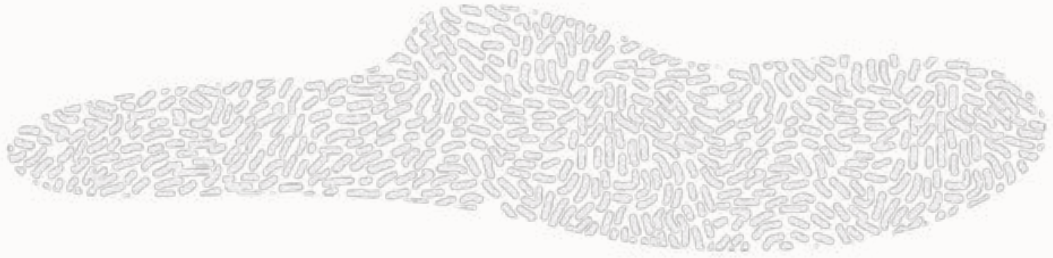
Snapshot Set	Instance	Inst num
First (1st)	DB19	1
Second (2nd)	DB19	1

Snapshot Set	Begin Snap Id	Begin Snap Time	End Snap Id	End Snap Time	Avg Active Users	Elapsed Time (min)	DB time (min)
1st	3	25-Feb-21 21:14:07 (Thu)	4	25-Feb-21 21:19:09 (Thu)	0.0	5.0	0.0
2nd	5	25-Feb-21 21:24:11 (Thu)	6	25-Feb-21 21:29:12 (Thu)	0.0	5.0	0.0
%Diff					-100.0	-0.2	-43.4

Host Configuration Comparison

	1st	2nd	Diff	%Diff
Number of CPUs:	4	4	0	0.0
Number of CPU Cores:	4	4	0	0.0
Number of CPU Sockets:	1	1	0	0.0
Physical Memory:	15725M	15725M	0M	0.0
Load at Start Snapshot:	.76	.4	-.36	-47.4
Load at End Snapshot:	.19	.5	.31	163.2
%User Time:	.18	.16	-.02	-11.1
%System Time:	.06	.05	-.01	-16.7
%Idle Time:	99.54	99.59	.05	0.1
%IO Wait Time:	.22	.15	-.06	-31.8

AWR | Diff Report



Use script `awrddrpt.sql`

Top Timed Events

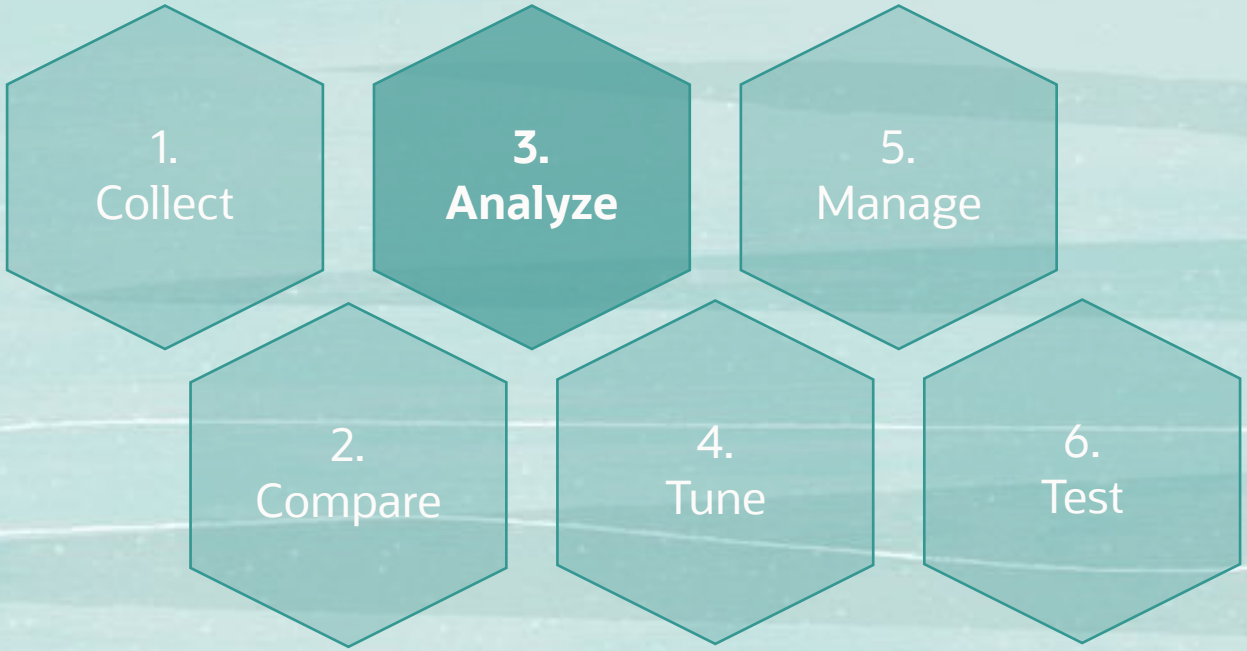
- Events with a "-" did not make the Top list in this set of snapshots, but are displayed for comparison purposes

1st						2nd					
Event	Wait Class	Waits	Time(s)	Avg Time(ms)	%DB time	Event	Wait Class	Waits	Time(s)	Avg Time(ms)	%DB time
CPU time			68,289.05		43.73	db file sequential read	User I/O	22,193,998	114,919.21	5.18	23.17
db file sequential read	User I/O	6,686,953	37,737.81	5.64	24.17	enq: SS - contention	Configuration	3,913	98,997.90	25,299.74	19.96
gc buffer busy	Cluster	12,508,244	23,886.55	1.91	15.30	CPU time			73,786.55		14.88
TCP Socket (KGAS)	Network	680,629	12,514.65	18.39	8.01	row cache lock	Concurrency	73,940	48,472.30	655.56	9.77
db file scattered read	User I/O	1,572,296	4,271.68	2.72	2.74	reliable message	Other	41,148	47,600.87	1,156.82	9.60

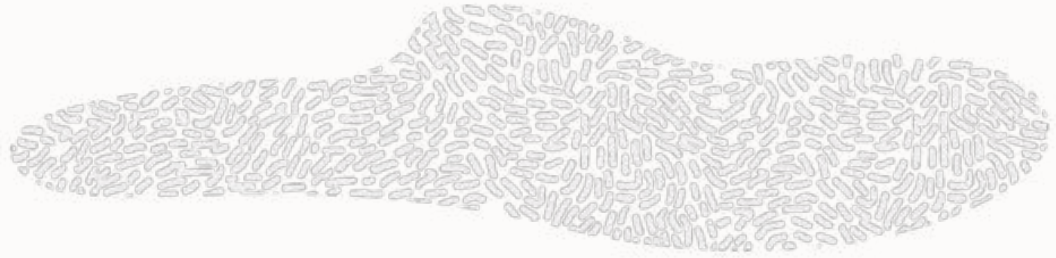
Requires Enterprise Edition + Diagnostic pack

Pro tip: For migrations, you can transport AWR data

Performance Stability Prescription



SQL Performance Analyzer | SPA

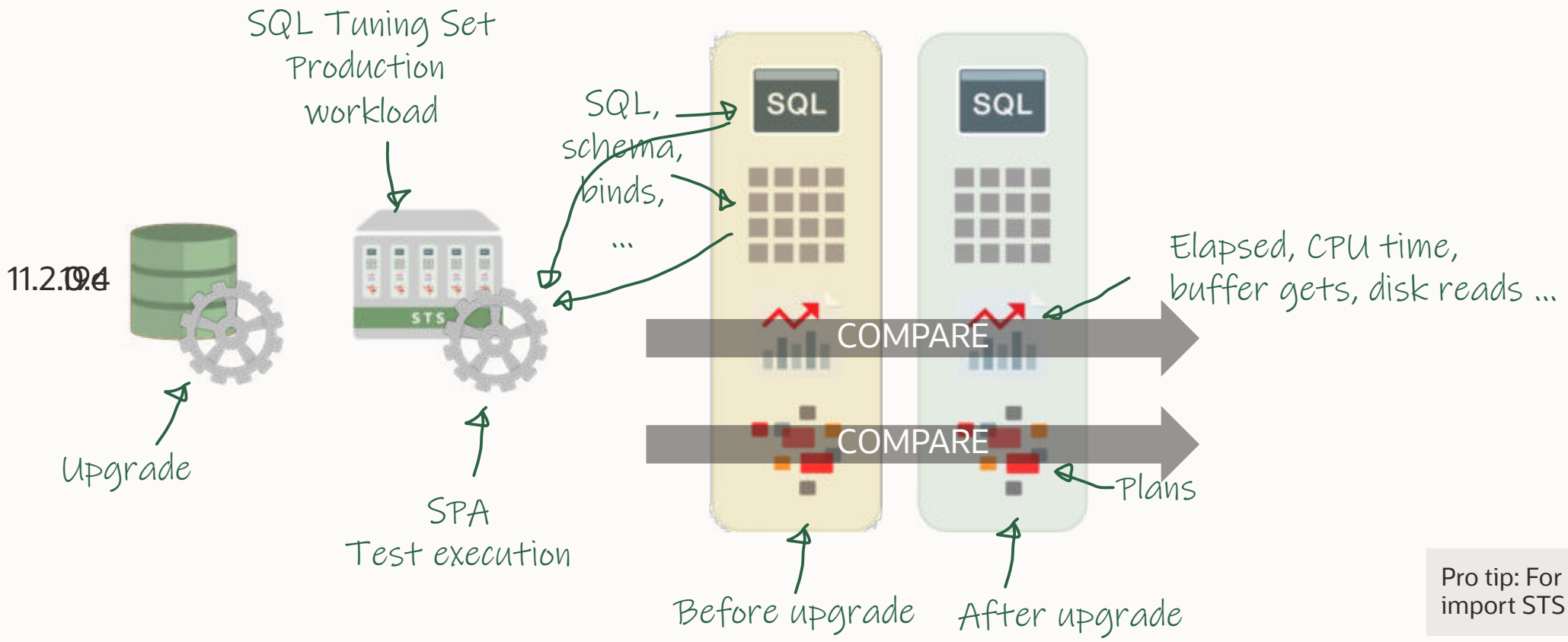


> *SPA provides fine-grained assessment of environment changes on SQL execution plans and statistics by running the SQL statements both in isolation and serially manner in before-change and after-change environments.*
>
> *SPA functionality is well integrated with existing SQL Tuning Set (STS), SQL Tuning Advisor, and SQL Plan Management functionality.*

Oracle Database Real Application Testing Data Sheet

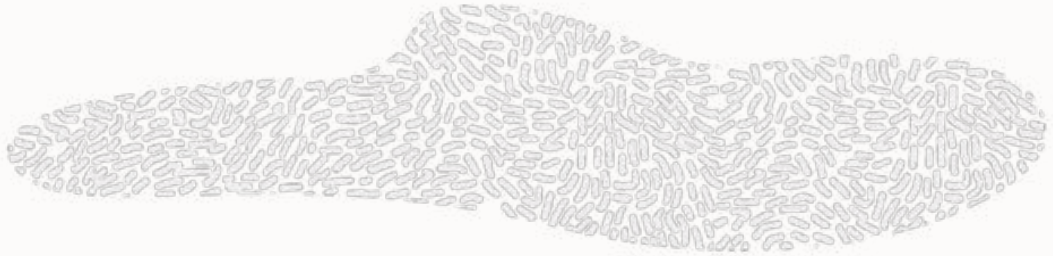
Requires Enterprise Edition + Real Application Testing

SPA | Concept



Pro tip: For migrations,
import STS into target database

SPA | Regressed Report



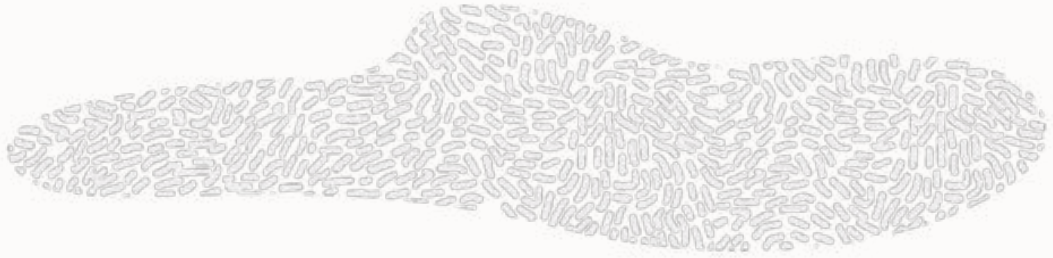
Regressed SQL Statements

	SQL ID	Net Impact on Workload (%)	Buffer Gets		Net Impact on SQL (%)	New Plan
			SQL Trial 1	SQL Trial 2		
⇩	3fv28gfu9y0aq	-0.050	26,504	29,573	-11.580	Y
⇩	czzzubf8fjz96	-0.030	1,410	1,981	-40.500	Y

From production workload

From test execution

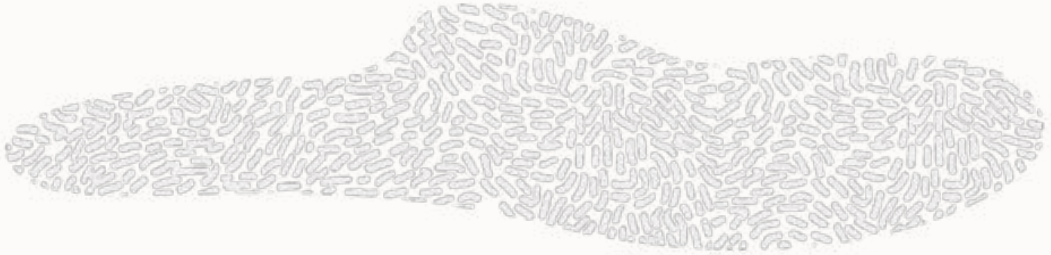
SPA | Regressed Report



Regressed SQL Statements

	SQL ID	Net Impact on Workload (%)	Buffer Gets		Net Impact on SQL (%)	New Plan
			SQL Trial 1	SQL Trial 2		
⇩	3fv28gfu9y0aq	-0.050	26,504	29,573	-11.580	Y
⇩	czzzubf8fjz96	-0.030	1,410	1,981	-40.500	Y

SPA | Regressed Report



Regressed SQL Statements

	SQL ID	Net Impact on Workload (%)	Buffer Gets		Net Impact on SQL (%)	New Plan
			SQL Trial 1	SQL Trial 2		
⇩	3fv28gfu9y0ag	-0.050	26,504	29,573	-11.580	Y
⇩	czzzubf8fjz96	-0.030	1,410	1,981	-40.500	Y

SQL Details: czzzubf8fjz96

Parsing Schema APPS Execution Frequency 3

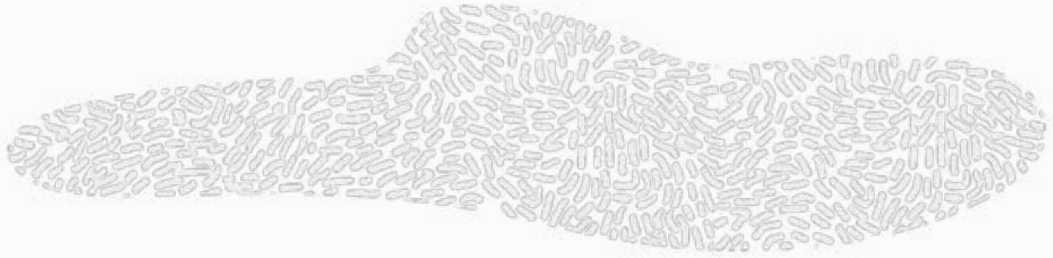
▷ SQL Text

```
SELECT /* my_query_21 */ /*+ ORDERED INDEX(t1) USE_HASH(t1) */ 'B' || t2.take_02 take_02, 'B' || t2.take_15
take_15, 'B' || t2.take_08 take_08, 'r' || t3.record_nr price_eur_id,...
```

Single Execution Statistics

	Execution Statistic Name	Net Impact on Workload (%)	Execution Statistic Collected		Net Impact on SQL (%)
			SQL Trial 1	SQL Trial 2	
⇩	Elapsed Time (sec)	-0.240	0.112	0.164	-46.170
⇧	Parse Time (sec)	0.220	0.001	0.001	14.490
⇩	CPU Time (sec)	-0.030	0.108	0.114	-5.040
⇨	User I/O Time (sec)	0.000	0.000	0.000	0.000
⇩	Buffer Gets	-0.030	1,410	1,981	-40.500

SPA | Regressed Report



Regressed SQL Statements

	SQL ID	Net Impact on Workload (%)	Buffer Gets		Net Impact on SQL (%)	New Plan
			SQL Trial 1	SQL Trial 2		
⇩	3fv28gfu9y0ag	-0.050	26,504	29,573	-11.580	Y
⇩	czzzubf8fjz96	-0.030	1,410	1,981	-40.500	Y

Plan Comparison

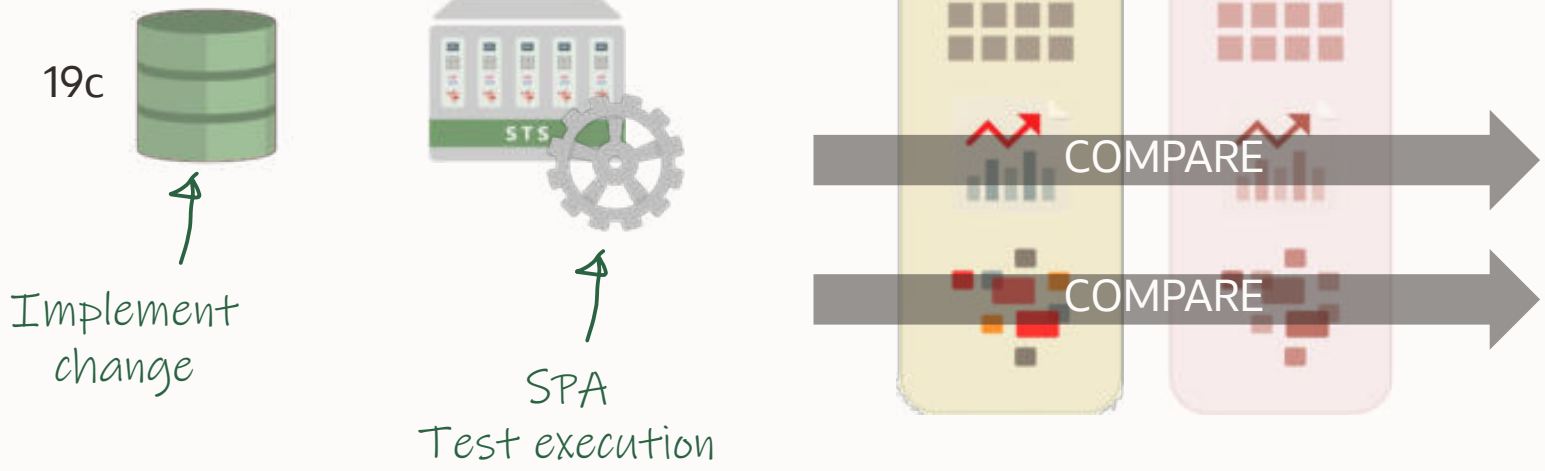
SQL_TRIAL_1353942463446

Plan Hash Value 1165613724

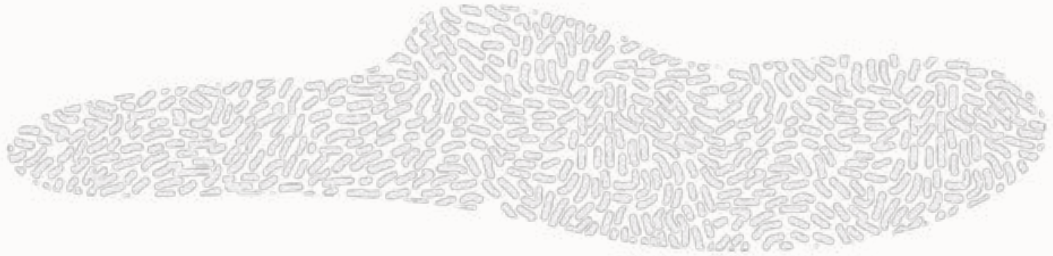
Expand All | Collapse All

Operation	Line ID	Object	Rows	Cost	Predicate
▽ SELECT STATEMENT	0		1	9,830	
▽ HASH GROUP BY	1		1	9,830	
▽ MERGE JOIN	2		1	9,829	
▽ SORT JOIN	3		8	9,795	
▽ HASH JOIN	4		8	9,794	"T1"."PERIOD_CODE"="T4"."FLYER...
INDEX RANGE SCAN	5	APPS.IDX$$_080F0004	1	2	"T4"."EXPORT_LIC_NR"=14659
▽ HASH JOIN	6		14,210	9,792	"T1"."SKU_NR"="T2"."SKU_NR" AN...

SPA | Continuous Improvement



SPA | Regressed Report



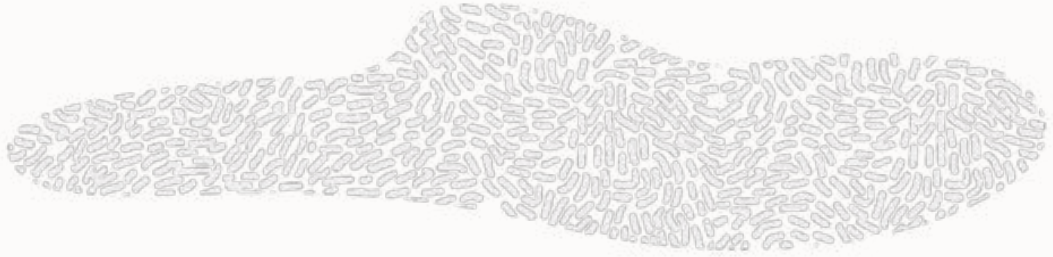
Regressed SQL Statements

	SQL ID	Net Impact on Workload (%)	Buffer Gets		Net Impact on SQL (%)	New Plan
			SQL Trial 1	SQL Trial 2		
⇩	3fv28gfu9y0aq	-0.050	26,504	29,573	-11.580	Y
⇩	czzzubf8fjz96	-0.030	1,410	1,981	-40.500	Y

Performance Stability Prescription

1. Collect
2. Compare
3. Analyze
4. **Tune**
5. Manage
6. Test

SQL Tuning Advisor



> *SQL Tuning Advisor is SQL diagnostic software in the Oracle Database Tuning Pack.*
>
> *...*
>
> *SQL Tuning Advisor is a mechanism for resolving problems related to suboptimally performing SQL statements.*

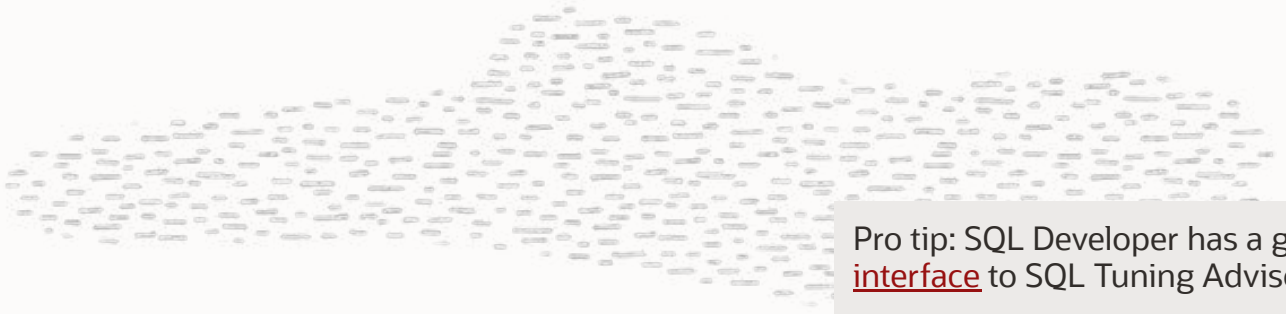
Database 19c SQL Tuning Guide, chapter 24

SQL Tuning Advisor | Findings

Types of findings:

1. Collection of object statistics
2. Creation of indexes
3. Rewriting SQL statements
4. Creation of SQL profiles

.... and more



Pro tip: SQL Developer has a good interface to SQL Tuning Advisor

6 simple steps with

SQL TUNING ADVISOR

can make a huge difference

1. Identify the problem
2. Select candidate statement
3. Get statement details
4. Execute tuning task
5. View report results
6. Apply recommendations

Real World Example | SQL Tuning Advisor in Action

1. Identify problem to be solved

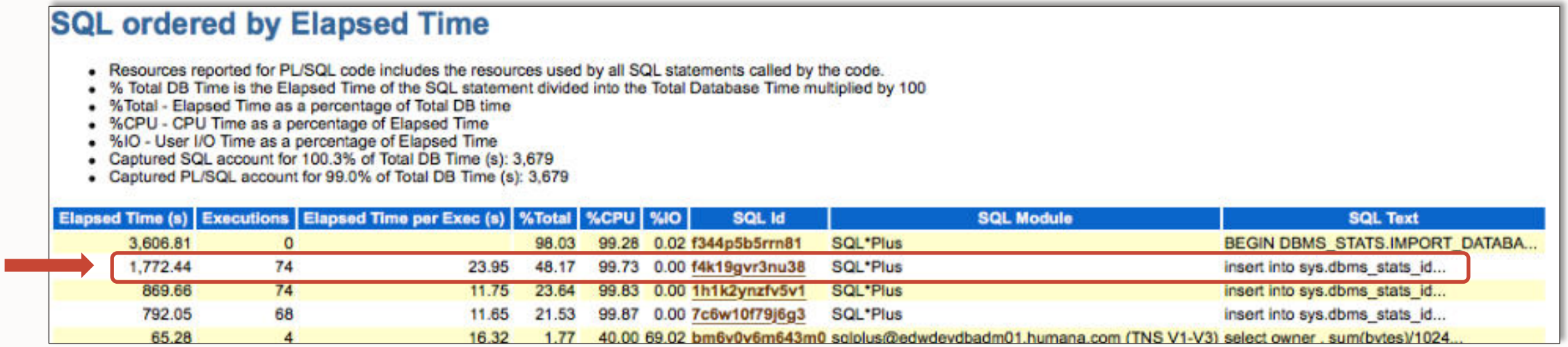
We should be trying to understand why the export of statistics to the stats table took 1.5 hrs, but the import took 48 hours to complete. Instead, it appears that the SR engineer wants to create some type of work-around situation.

I would think we would want to trace the import stats table process to determine why it is taking so long. The stats table that was created only contains 2.8 GBs of data, which should be able to be loaded in mins....

Your help to keep this on track is appreciated...

Real World Example | SQL Tuning Advisor in Action

2. Select candidate statement to tune
   - Generate an AWR Report



SQL ordered by Elapsed Time

- Resources reported for PL/SQL code includes the resources used by all SQL statements called by the code.
- % Total DB Time is the Elapsed Time of the SQL statement divided into the Total Database Time multiplied by 100
- %Total - Elapsed Time as a percentage of Total DB time
- %CPU - CPU Time as a percentage of Elapsed Time
- %IO - User I/O Time as a percentage of Elapsed Time
- Captured SQL account for 100.3% of Total DB Time (s): 3,679
- Captured PL/SQL account for 99.0% of Total DB Time (s): 3,679

Elapsed Time (s)	Executions	Elapsed Time per Exec (s)	%Total	%CPU	%IO	SQL Id	SQL Module	SQL Text
3,606.81	0		98.03	99.28	0.02	f344p5b5rrn81	SQL*Plus	BEGIN DBMS_STATS.IMPORT_DATABA...
1,772.44	74	23.95	48.17	99.73	0.00	f4k19gvr3nu38	SQL*Plus	insert into sys.dbms_stats_id...
869.66	74	11.75	23.64	99.83	0.00	1h1k2ynzfv5v1	SQL*Plus	insert into sys.dbms_stats_id...
792.05	68	11.65	21.53	99.87	0.00	7c6w10f79j6g3	SQL*Plus	insert into sys.dbms_stats_id...
65.28	4	16.32	1.77	40.00	69.02	bm6v0v6m643m0	sqlplus@edwdevdbadm01.humana.com (TNS V1-V3)	select owner , sum(bytes)/1024...

Real World Example | SQL Tuning Advisor in Action

3. Get SQL Statement Details

	DBMS_XPLAN.BUILD_PLAN_XML(TABLE_NAME=> gv$sql_plan , PLAN_TAG=> plan , FILTER_PREDS=>:B35 , FORMAT=> - PROJECTION +ALIAS +ADAPTIVE') ELSE NULL END XPLAN_XML FROM DUAL) V1) CONST_VIEW
dnak3w997p17j	update tabpart$ set dataobj# = :1, part# = :2, ts# = :3, file# = :4, block# = :5, pctfree$ = :6, pctused$ = :7, initrans = :8, maxtrans = :9, flags = :10, analyzetime = :11, samplesize = :12, rowcnt = :13, blkcnt = :14, empcnt = :15, avgspc = :16, chncnt = :17, avgrln = :18 where obj# = :19
f344p5b5rrn81	BEGIN DBMS_STATS.IMPORT_DATABASE_STATS(stattab => 'STATS'); END;
f4k19gvr3nu38	insert into sys.dbms_stats_id_map_tab (c5, c1, c2, cn) select distinct s.c5, s.c1, s.c2, d.partition_name cn from "SYSTEM"."STATS" s, (select u.name table_owner, op.name table_name, op.subname partition_name, tp.part# partition_position from user$ u, obj$ op, (select obj#, part# from tabpartv$ union all select obj#, part# from tabcompartv$) tp where u.user# = op.owner# and op.type# = 19 and op.obj# = tp.obj#) d where s.c5 = :1 and s.c1 = :2 and s.type in ('T', 'C', 'E', 'P', 'H', 'B', 't', 'c', 'M', 'U', 'G', 'L') and s.n13 is not null and s.c2 is not null and s.c3 is null and s.c5 = d.table_owner and s.c1 = d.table_name and s.n13 = d.partition_position and s.c2 != d.partition_name and s.statid is null
fcj8q52nqgfc5	update indcompart$ set part# = :1, subpartcnt = :2, flags = :3, defts# = :4, defpctfree = :5, definitrans = :6, defmaxtrans = :7, definiexts = :8, defextsize = :9, defminexts = :10, defmaxexts = :11, defextpct = :12, deflists = :13, defgroups = :14, defbufpool = :15, deflogging = :16, analyzetime = :17, samplesize = :18, rowcnt = :19, blevel = :20, leafcnt = :21, distkey = :22, lblkkey = :23, dblkkey = :24, clufac = :25, spare2 = :26, spare3 =:27, defmaxsize = :28 where obj# = :29
fdzqjmpvd6hvy	SELECT O.DATAOBJ# FROM SYS.OBJ$ O WHERE O.OBJ# = :B1

Real World Example | SQL Tuning Advisor in Action

4. Create and run SQL Tuning task

```
VARIABLE stmt_task VARCHAR2(64);

EXEC :stmt_task := DBMS_SQLTUNE.CREATE_TUNING_TASK(sql_id => 'f4k19gvr3nu38');
EXEC DBMS_SQLTUNE.EXECUTE_TUNING_TASK(:stmt_task);

SET LONG 10000
SET LONGCHUNKSIZE 10000
SET LINESIZE 100

SPOOL sql_tune_f4k19gvr3nu38.txt

SELECT dbms_sqltune.report_tuning_task( :stmt_task )FROM DUAL;

SPOOL OFF;
```

Real World Example | SQL Tuning Advisor in Action

5. View report results

```
DBMS_SQLTUNE.REPORT_TUNING_TASK(:STMT_TASK)
-------------------------------------------------------------------------------
Schema Name: SYS
SQL ID     : f4k19gvr3nu38
SQL Text   :       insert into sys.dbms_stats_id_map_tab (c5, c1, c2, cn)
                   select distinct s.c5, s.c1, s.c2, d.partition_name cn
                   from "SYSTEM"."STATS" s,
                      (select u.name table_owner, op.name table_name,
                         op.subname partition_name,
                         tp.part# partition_position
                       from user$ u, obj$ op,
                            (select obj#, part# from tabpartv$
                             union all
                             select obj#, part# from tabcompartv$
                            ) tp
                       where u.user# = op.owner# and op.type# = 19 and op.obj# = tp.obj#
                      ) d
                   where s.c5 = :1 and s.c1 = :2
                     and s.type in ('T','C','E','P','H','B','t','c','M','U','G','L')
                     and s.n13 is not null and s.c2 is not null and s.c3 is null
                     and s.c5 = d.table_owner and s.c1 = d.table_name
                     and s.n13 = d.partition_position
                     and s.c2 != d.partition_name and s.statid is null
Bind Variables :
 1 -  (VARCHAR2(32)):STG
 2 -  (VARCHAR2(32)):MTH_MBR_COV_PLAN_PCP_STG
```

Real World Example | SQL Tuning Advisor in Action

5. View report results

```
-------------------------------------------------------------------------------
FINDINGS SECTION (8 findings)
-------------------------------------------------------------------------------

1- Statistics Finding
---------------------
  Optimizer statistics for table "SYS"."TABPART$" and its indices are stale.

  Recommendation
  --------------
  - Consider collecting optimizer statistics for this table.
    execute dbms_stats.gather_table_stats(ownname => 'SYS', tabname =>
            'TABPART$', estimate_percent => DBMS_STATS.AUTO_SAMPLE_SIZE,
            method_opt => 'FOR ALL COLUMNS SIZE AUTO');

  Rationale
  ---------
    The optimizer requires up-to-date statistics for the table in order to
    select a good execution plan.
```

Real World Example | SQL Tuning Advisor in Action

5. View report results

```
6- SQL Profile Finding (see explain plans section below)
--------------------------------------------------------

  A potentially better execution plan was found for this statement.

  Recommendation (estimated benefit: 67.2%)
  -----------------------------------------
  - Consider accepting the recommended SQL profile.
    execute dbms_sqltune.accept_sql_profile(task_name => 'TASK_21944',
            task_owner => 'SYS', replace => TRUE);

  Validation results
  ------------------
  The SQL profile was tested by executing both its plan and the original plan
  and measuring their respective execution statistics. A plan may have been
  only partially executed if the other could be run to completion in less time.

                           Original Plan  With SQL Profile  % Improved
                           -------------  -----------------  ----------
  Completion Status:            COMPLETE           COMPLETE

  Elapsed Time (s):             2.588553           .802211         69 %
  CPU Time (s):                  2.57261           .799878       68.9 %
  User I/O Time (s):            .000557                 0        100 %
  Buffer Gets:                   182336             59805       67.2 %
  Physical Read Requests:             7                 0        100 %
  Physical Write Requests:            0                 0
  Physical Read Bytes:           114688                 0        100 %
  Physical Write Bytes:               0                 0
  Rows Processed:                     0                 0
  Fetches:                            0                 0
  Executions:                         1                 1
```

Real World Example | SQL Tuning Advisor in Action

5. View report results

```
7- Index Finding (see explain plans section below)
--------------------------------------------------
  The execution plan of this statement can be improved by creating one or more
  indices.


DBMS_SQLTUNE.REPORT_TUNING_TASK(:STMT_TASK)
-------------------------------------------------------------------------------

  Recommendation (estimated benefit: 88.23%)
  ------------------------------------------
  - Consider running the Access Advisor to improve the physical schema design
    or creating the recommended index.
    create index SYSTEM.IDX$$_55B80001 on SYSTEM.STATS("N13");

  Rationale
  ---------
    Creating the recommended indices significantly improves the execution plan
    of this statement. However, it might be preferable to run "Access Advisor"
    using a representative SQL workload as opposed to a single statement. This
    will allow to get comprehensive index recommendations which takes into
    account index maintenance overhead and additional space consumption.
```

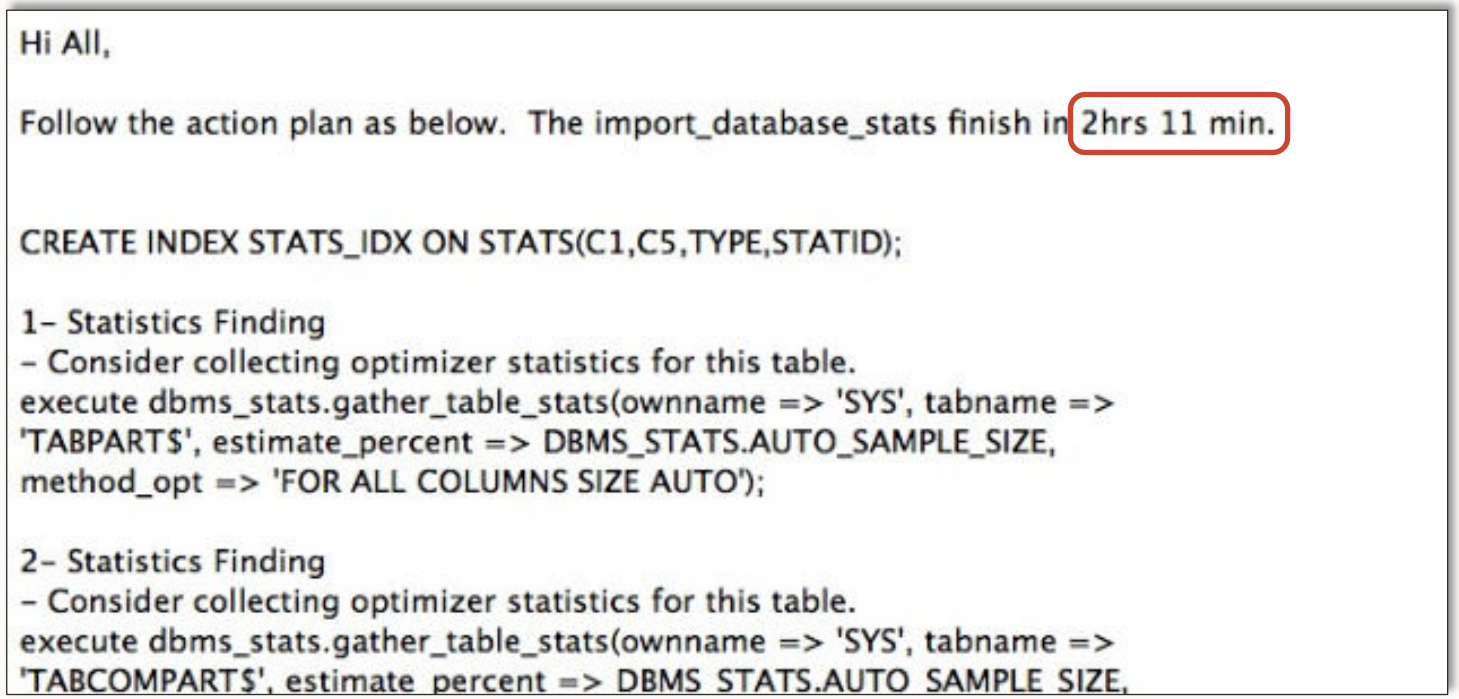
Real World Example | SQL Tuning Advisor in Action

5. View report results

```
8- Alternative Plan Finding
---------------------------
  Some alternative execution plans for this statement were found by searching
  the system's real-time and historical performance data.

  The following table lists these plans ranked by their average elapsed time.
  See section "ALTERNATIVE PLANS SECTION" for detailed information on each
  plan.

  id plan hash  last seen            elapsed (s)  origin          note
  -- ---------- -------------------- ------------ --------------- ----------------
   1 1434278210  2017-01-11/01:19:14        1.254 Cursor Cache    not reproducible
   2 1201858690  2017-01-11/01:19:23        1.355 Cursor Cache    not reproducible
   3 3167061724  2017-01-11/01:19:39        2.580 Cursor Cache    not reproducible
   4 2779611207  2017-01-11/01:19:30        2.855 Cursor Cache    not reproducible
   5 2908117100  2017-01-11/05:00:47        3.093 Cursor Cache    not reproducible

  Information
  -----------
  - All alternative plans other than the Original Plan could not be
    reproduced in the current environment.
  - The plan with id 1 could not be reproduced in the current environment.
    For this reason, a SQL plan baseline cannot be created to instruct the
    Oracle optimizer to pick this plan in the future.
  - The plan with id 2 could not be reproduced in the current environment.
    For this reason, a SQL plan baseline cannot be created to instruct the
    Oracle optimizer to pick this plan in the future.
  - The plan with id 3 could not be reproduced in the current environment.
    For this reason, a SQL plan baseline cannot be created to instruct the
```

Real World Example | SQL Tuning Advisor in Action

6. Act on findings
   - Follow 5 statistics recommendations to gather stats on 5 tables



Hi All,

Follow the action plan as below. The import_database_stats finish in 2hrs 11 min.

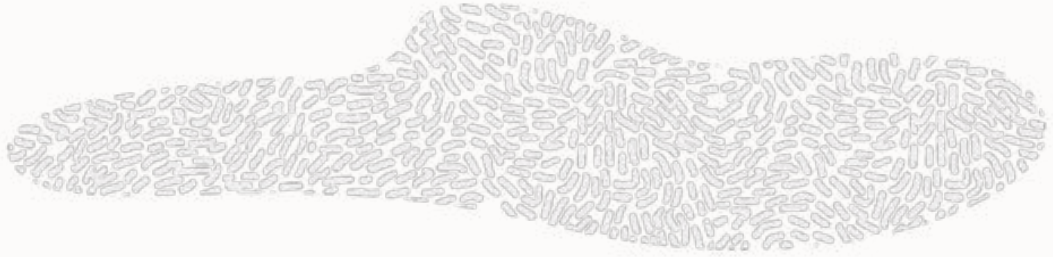
CREATE INDEX STATS_IDX ON STATS(C1,C5,TYPE,STATID);

1- Statistics Finding
- Consider collecting optimizer statistics for this table.
execute dbms_stats.gather_table_stats(ownname => 'SYS', tabname =>
'TABPART$', estimate_percent => DBMS_STATS.AUTO_SAMPLE_SIZE,
method_opt => 'FOR ALL COLUMNS SIZE AUTO');

2- Statistics Finding
- Consider collecting optimizer statistics for this table.
execute dbms_stats.gather_table_stats(ownname => 'SYS', tabname =>
'TABCOMPART$', estimate_percent => DBMS_STATS.AUTO_SAMPLE_SIZE,

   - Result: 20x improvement!

SQL Profiles

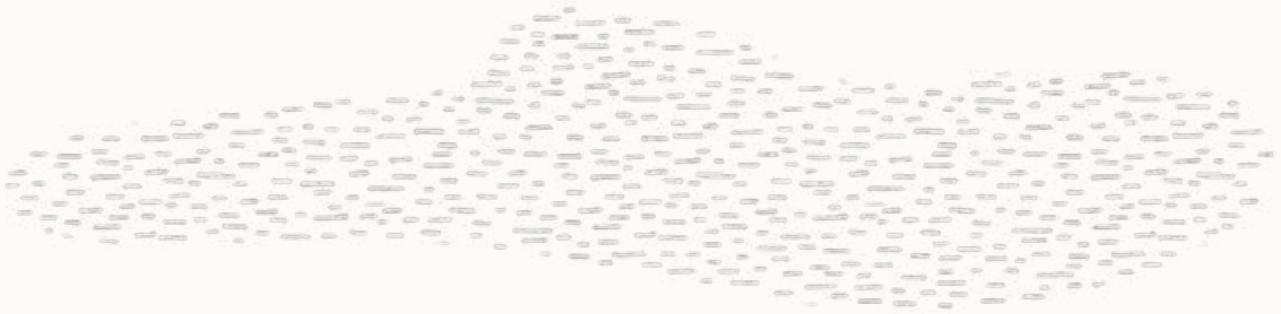


> *A SQL profile is a database object that contains auxiliary statistics specific to a SQL statement.*
>
> *...*
>
> *The corrected statistics in a SQL profile can improve optimizer cardinality estimates, which in turn leads the optimizer to select better plans.*

Database 19c SQL Tuning Guide, chapter 26

SQL Profiles | Facts

1. Part of Tuning Pack
   - Included in some cloud offerings
2. Stores a set of hints that causes the optimizer to select a plan
3. Affects one statement only



SQL Profiles | Facts

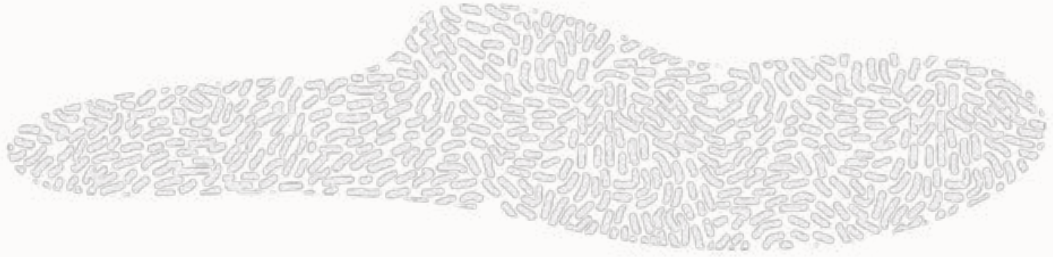
4. You can enable/disable a profile

5. Transparent to application
   - Does not require application changes

6. Persistent and transportable
   - Documentation

7. Useful with literals using `FORCE_MATCH=TRUE`

SQL Profile | Testing



1. Enable profile for selected environments only

```
SQL> exec :p_name := dbms_sqltune.accept_sql_profile(
   task_name=>'TASK_21944',
   name=>'XT_PROFILE',
   category=>'TEST_ENV');
```

2. Verify the profile – it doesn't get used by the optimizer in the live environment

```
SQL> alter session set sqltune_category='TEST_ENV';
```

3. Accept and make visible to all sessions (`'DEFAULT'`)

```
SQL> exec dbms_sqltune.alter_sql_profile(
   name=>'XT_PROFILE',
   attribute_name=>'CATEGORY',
   value=>'DEFAULT');
```

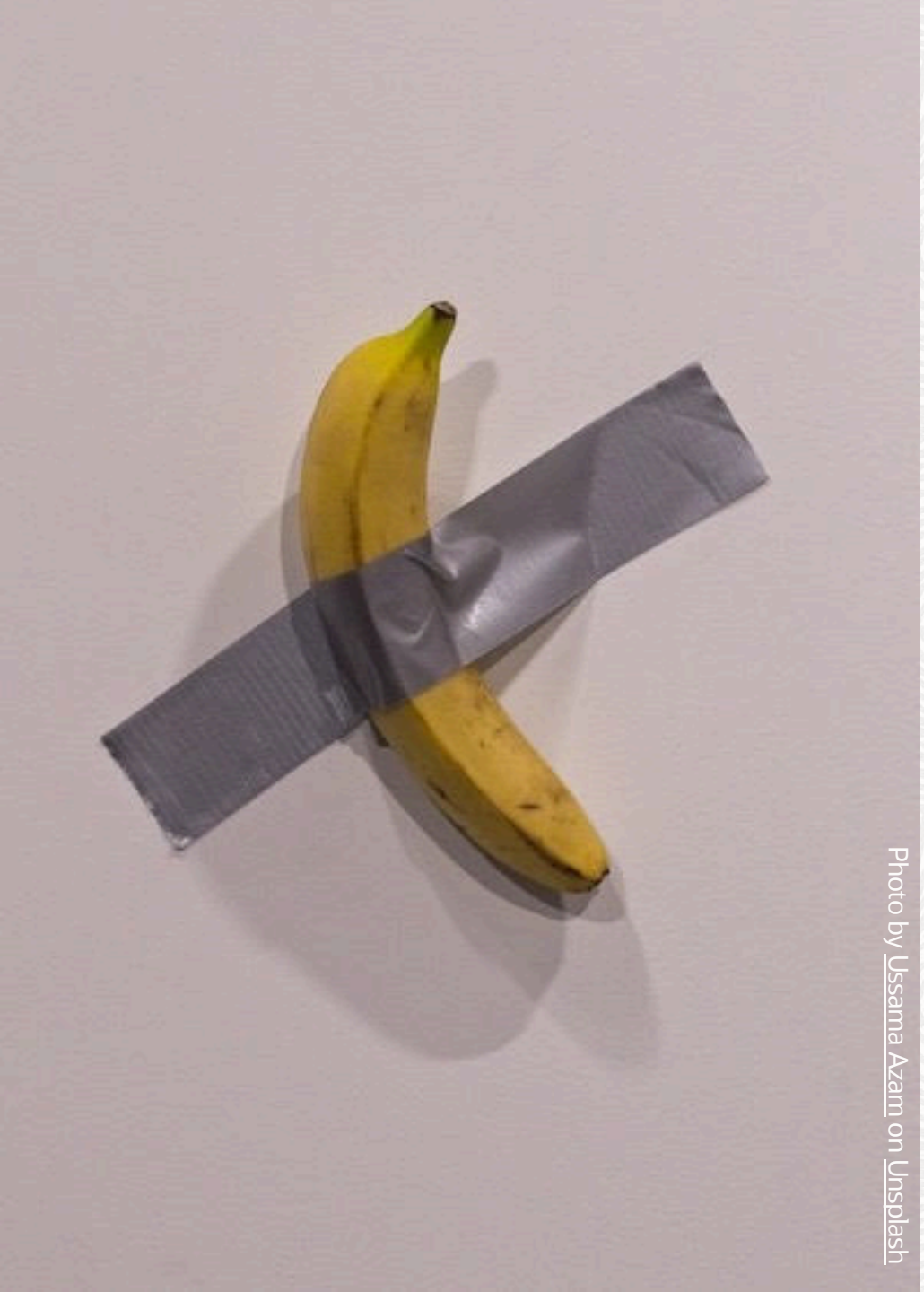


Photo by Ussama Azam on Unsplash

SQL Patch

SQL Patch | Overview

Repair SQL statements

- Add hints
- Transparent
- Persistent
- Introduced in Oracle 11g
  - Oracle 11g and 12.1: `DBMS_SQLDIAG_INTERNAL`
  - Oracle 12.2 and newer: `DBMS_SQLDIAG`
- Available in EE and SE2

- Documentation: https://docs.oracle.com/en/database/oracle/oracle-database/19/arpls/DBMS_SQLDIAG.html#GUID-0F29CD05-6BF3-4EEB-90F5-E2465865C255
- Useful scripts, e.g., create_sql_patch.sql: http://kerryosborne.oracle-guy.com/2013/06/06/sql-gone-bad-but-plan-not-changed/

SQL Patch | Version Differences

Oracle 11.2 and 12.1

- DBMS_SQLDIAG_INTERNAL
  (undocumented)

```
BEGIN
  SYS.DBMS_SQLDIAG_INTERNAL.i_create_patch(
    sql_text  => 'select * big_table',
    hint_text => 'PARALLEL(big_table,10)',
    name      => 'big_table_sql_patch');
END;
/
```

Oracle 12.2 and newer

- DBMS_SQLDIAG

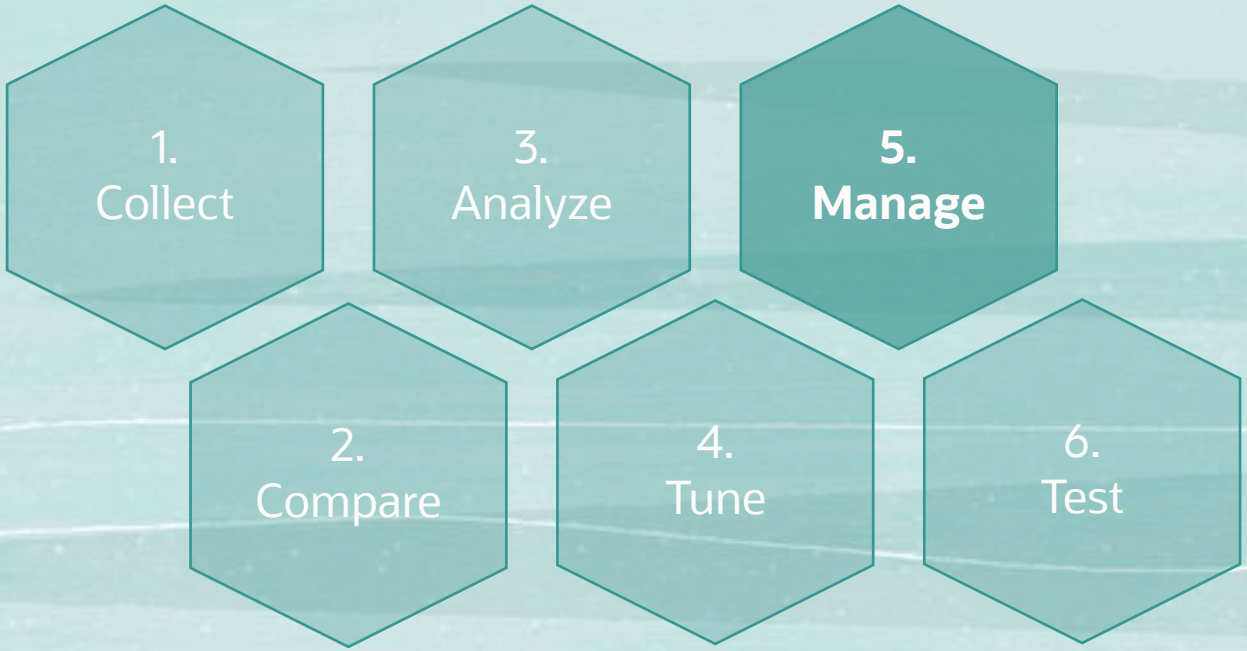
```
DECLARE
  l_patch_name  VARCHAR2(4000);
BEGIN
  l_patch_name :=
 SYS.DBMS_SQLDIAG.create_sql_patch(
    sql_text  => 'select * from big_table',
    hint_text => 'PARALLEL(big_table,10)',
    name      => 'big_table_sql_patch');
END;
/
```

Pro tip: You can use *SQL ID* instead of the full *SQL text*

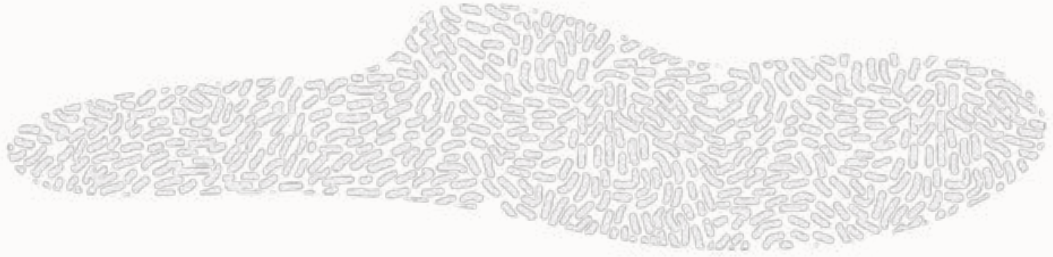
SQL Patch | Demo

Watch on YouTube

Performance Stability Prescription



SQL Plan Management | SPM

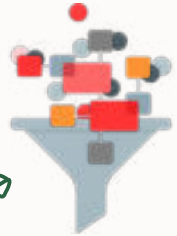


> *SQL plan management uses a mechanism called a **SQL plan baseline**, which is a set of accepted plans that the optimizer is allowed to use for a SQL statement.*
>
> *...*
>
> *SQL plan management prevents performance regressions caused by plan changes.*

Database 19c SQL Tuning Guide, chapter 27

SPM | Concept

Repeatable SQL



Filtering:

- Schema
- SQL text
- Module
- Action

Filter

Plan B

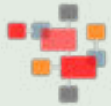
No plans in baseline

Plan A is used

Plan history

Dedicated part of SQL plan baseline

```
OPTIMIZER_USE_SQL_PLAN_BASELINES=TRUE
```

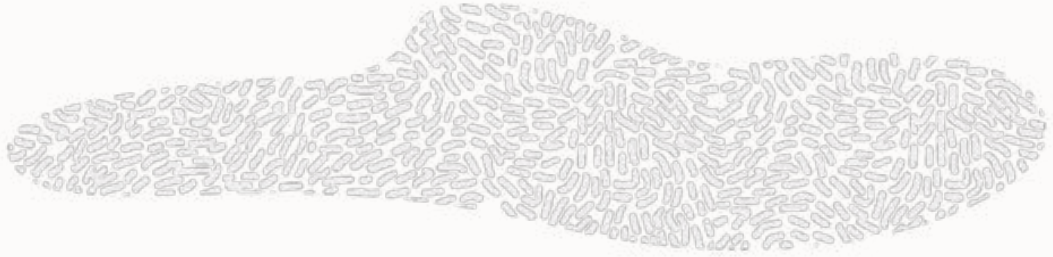


Plan A
Accepted

Plan B
Not accepted

Plan History

SPM | Plans



The plans in a SQL plan baseline can be:

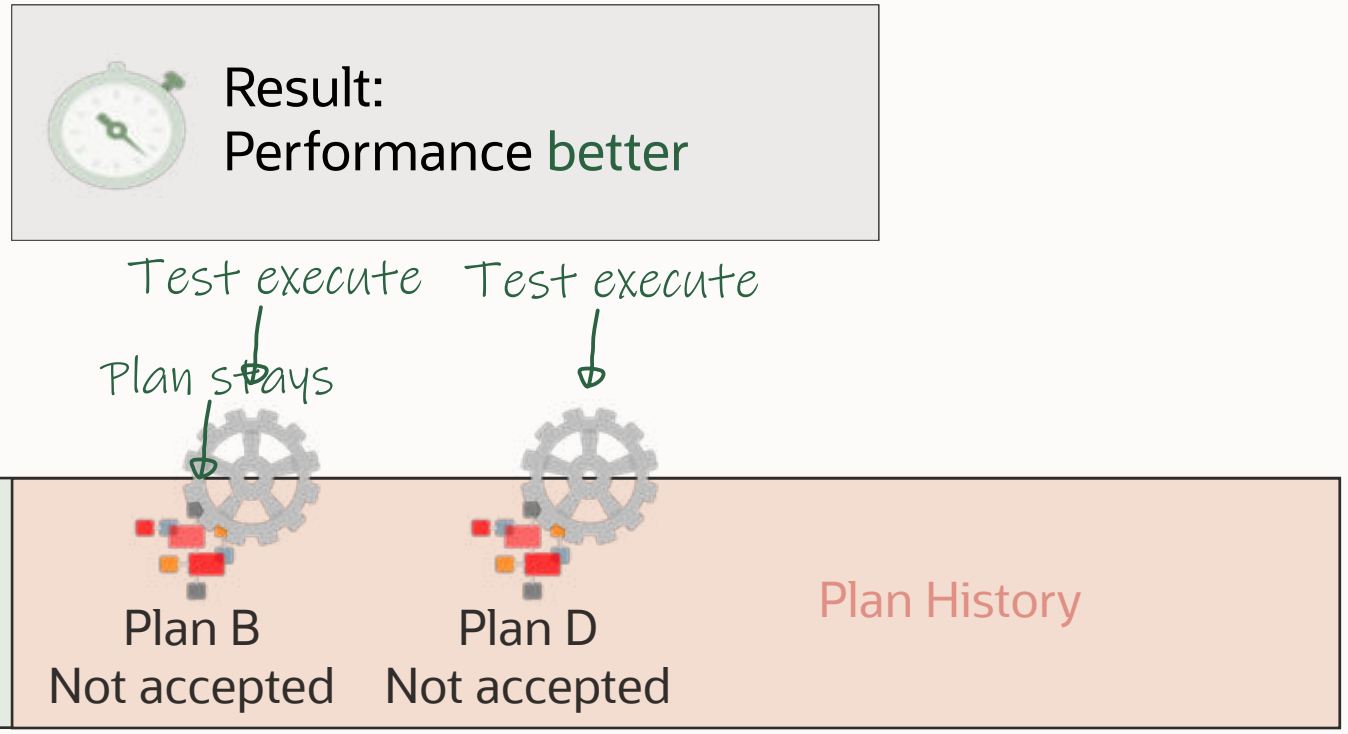
- Enabled
- Accepted
- Fixed

To change status use `DBMS_SPM.ALTER_SQL_PLAN_BASELINE`

You can also prevent plans from getting purged by setting the `autopurge` property.

Pro tip: The *Accepted* attribute can only be set by a test execution

SPM | Evolve



SPM | Evolve

Evolving happens in maintenance task SYS_AUTO_SPM_EVOLVE_TASK

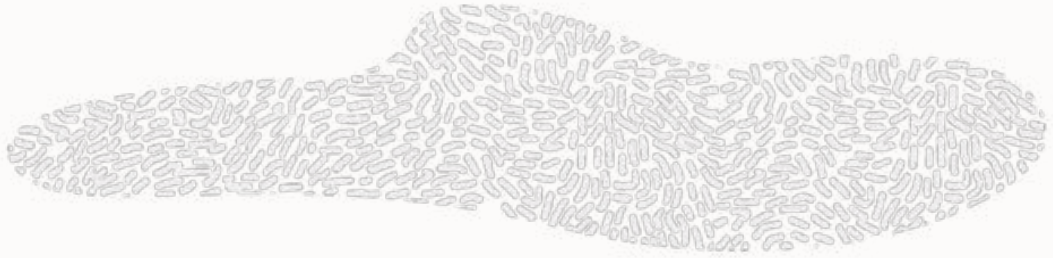
- Part of Automatic SQL Tuning Task

You decide whether recommendations are implemented automatically

```
SQL> BEGIN
   DBMS_SPM.SET_EVOLVE_TASK_PARAMETER(
      parameter => 'accept_plans',
      value     => 'true');
   END;
/
```

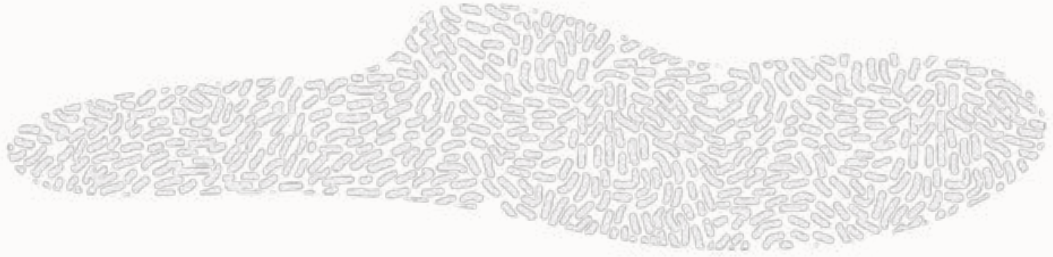
You can evolve plans manually

SPM | Management Base



- SQL Management Base is stored in SYSAUX tablespace
- Plans are stored in a LOB
- Unused plans are deleted after 53 weeks
- Space budget is 10 %

SPM | Management Base



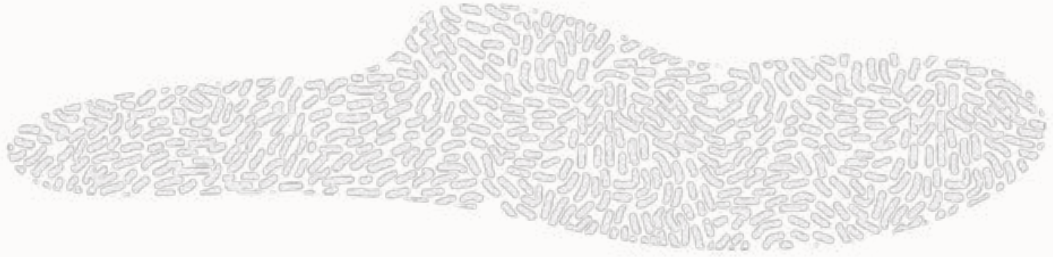
Check your settings

```
SQL> select parameter_name, parameter_value from dba_sql_management_config;

PARAMETER_NAME                          PARAMETER_VALUE
--------------------------------------- --------------------
AUTO_CAPTURE_ACTION
AUTO_CAPTURE_PARSING_SCHEMA_NAME
AUTO_CAPTURE_SQL_TEXT
AUTO_SPM_EVOLVE_TASK                    OFF
                                        [illegible]
PLAN_RETENTION_WEEKS                    53
SPACE_BUDGET_PERCENT                    10
```

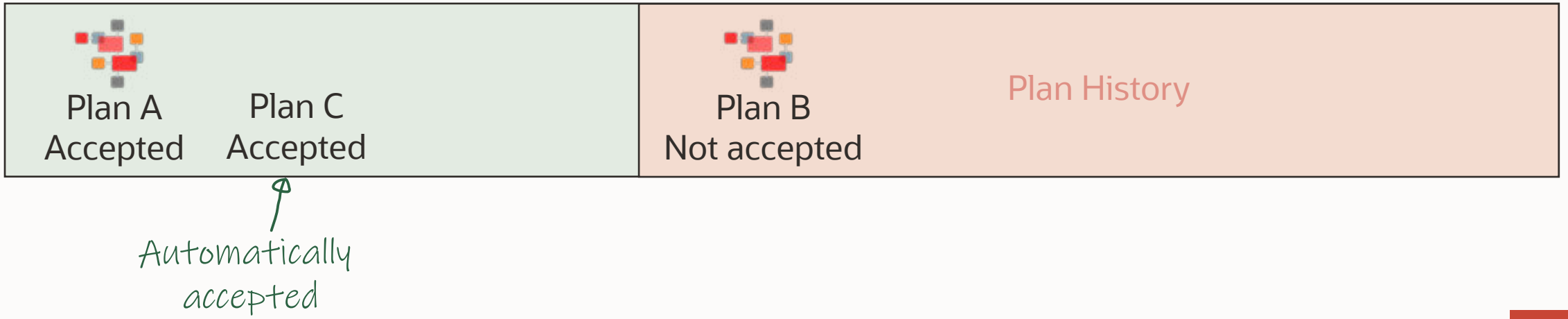
```
SQL> exec DBMS_SPM.CONFIGURE('plan_retention_weeks', 5);
```

```
SQL> exec DBMS_SPM.CONFIGURE('space_budget_percent', 5);
```

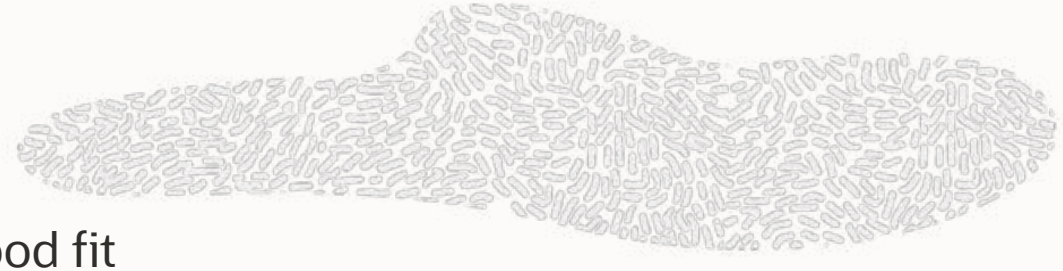
SPM | Load from STS




```
SQL> DECLARE
       cnt number;
   BEGIN
       cnt := DBMS_SPM.LOAD_PLANS_FROM_SQLSET('UPG_STS_1');
   END;
/
```



SPM | What if … literals

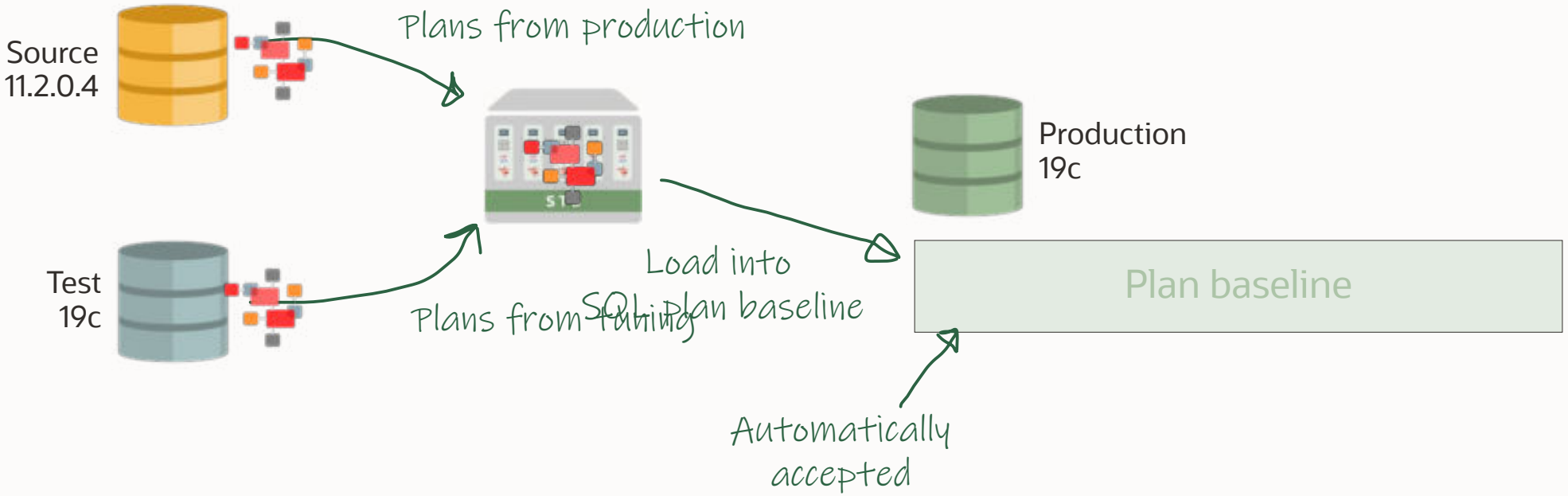


SQL Plan Management in a system with literals is not a good fit

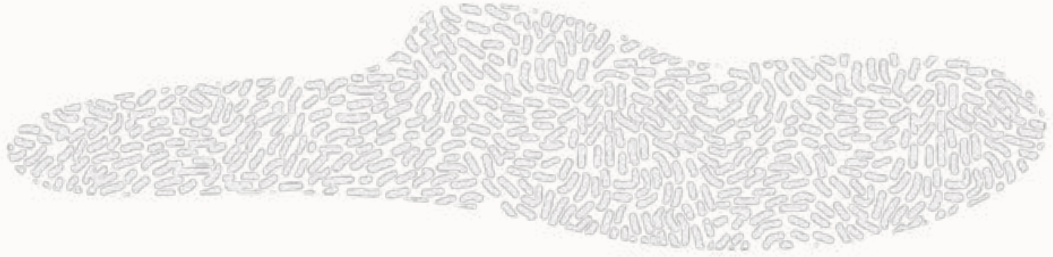
- Many distinct statements
- CURSOR_SHARING = FORCE? No!
- SQL profiles can do force matching

Optimal solution: Change your application to use bind variables

SPM | Use Case



SPM | Use Case




```
SQL> DECLARE
  plans_loaded NUMBER;
  filter VARCHAR2(255);
BEGIN
  filter := 'sql_id=''czzzubf8fjz96'' AND plan_hash_value=''1165613724''';

  plans_loaded := DBMS_SPM.LOAD_PLANS_FROM_SQLSET (
    sqlset_name  => 'UPG_STS_1',
    basic_filter => filter
  );
END;
/
```

Pro tip: The function `LOAD_PLANS_FROM_SQLSET` can also *fix* the plans

Photo by Julia Joppien on Unsplash

Transporting SQL Plan Baselines

Transporting | SQL Plan Baseline

Prepare

Extract

Transfer

Load

SQL Plan Baselines are stored in data dictionary

To transfer - information must be converted to a transportable format and stored in a staging table

```
SQL> BEGIN
   DBMS_SPM.CREATE_STGTAB_BASELINE (
      table_name   => 'SPB_STAGING',
      table_owner  => 'SPM');
   END;
/
```

Pro tip: Create the staging table in a schema and/or tablespace that you are migrating

Transporting | SQL Plan Baseline

Prepare

Extract

Transfer

Load

Select the baselines that you want to transfer

To extract the fixed and accepted plans

```
SQL> DECLARE
   l_count NUMBER;
BEGIN
   l_count := DBMS_SPM.PACK_STGTAB_BASELINE (
      table_name   => 'SPB_STAGING',
      table_owner  => 'SPM',
      enabled      => 'YES',
      fixed        => 'YES');
   END;
/
```

Pro tip: You can also use `dba_sql_plan_baselines` to find plans

Transporting | SQL Plan Baseline

Prepare
Extract
Transfer
Load

If the staging table is migrated together with the user data, you can skip this step

Use Data Pump to transfer that single table

```
SQL> CREATE DATABASE LINK src_link ... ;

$ impdp system network_link=src_link tables=SPM.SPB_STAGING ...
```

Pro tip: You can also import from dump file if there is no network connectivity to source database

Transporting | SQL Plan Baseline

- Prepare
- Extract
- Transfer
- **Load**

Finally, load the baselines from the staging table into the data dictionary

```
SQL> DECLARE
   l_count NUMBER;
BEGIN
   l_count := DBMS_SPM.UNPACK_STGTAB_BASELINE (
      table_name   => 'SPB_STAGING',
      table_owner  => 'SPM');
   END;
/
```

Pro tip: You can apply filters to limit the baselines to import

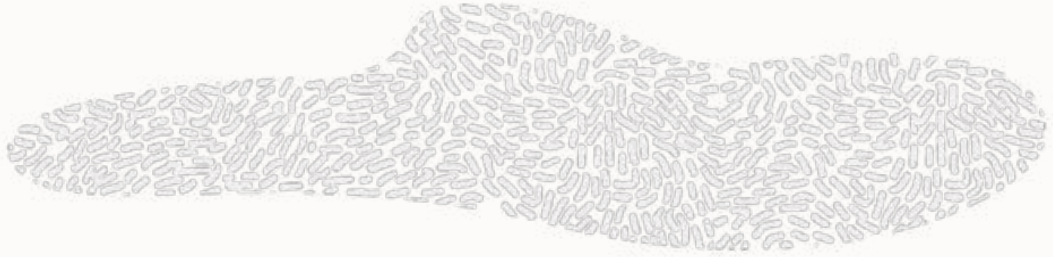
Different PLAN STABILITY options

	SQL Plan Management	SQL Profile	SQL Patch
Edition	EE (subset in SE2)	EE + Tuning	All
Method	Restrict plan usage	Improves cardinality estimates	Applies hints
Stores	Entire plan	Statistics / hints	Hints
Transportable	Yes	Yes	Yes
Plan guarantee	Yes	-	-
Maintenance	Automatic evolve	Manual inspection	Manual inspection

Performance Stability Prescription

1. Collect
2. Compare
3. Analyze
4. Tune
5. Manage
6. **Test**

Database Replay | Overview

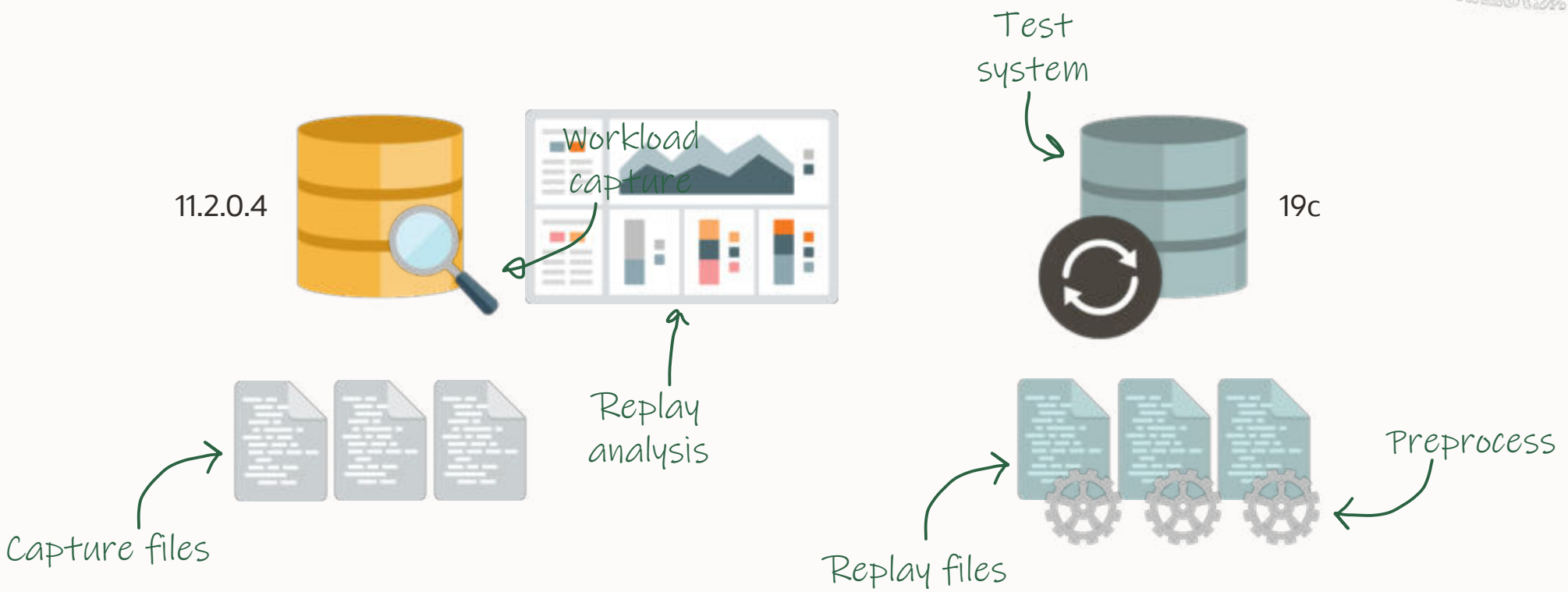


> *You can use Database Replay to capture a workload on the production system and replay it on a test system with the exact timing, concurrency, and transaction characteristics of the original workload.*
>
> *This enables you to test the effects of a system change without affecting the production system.*

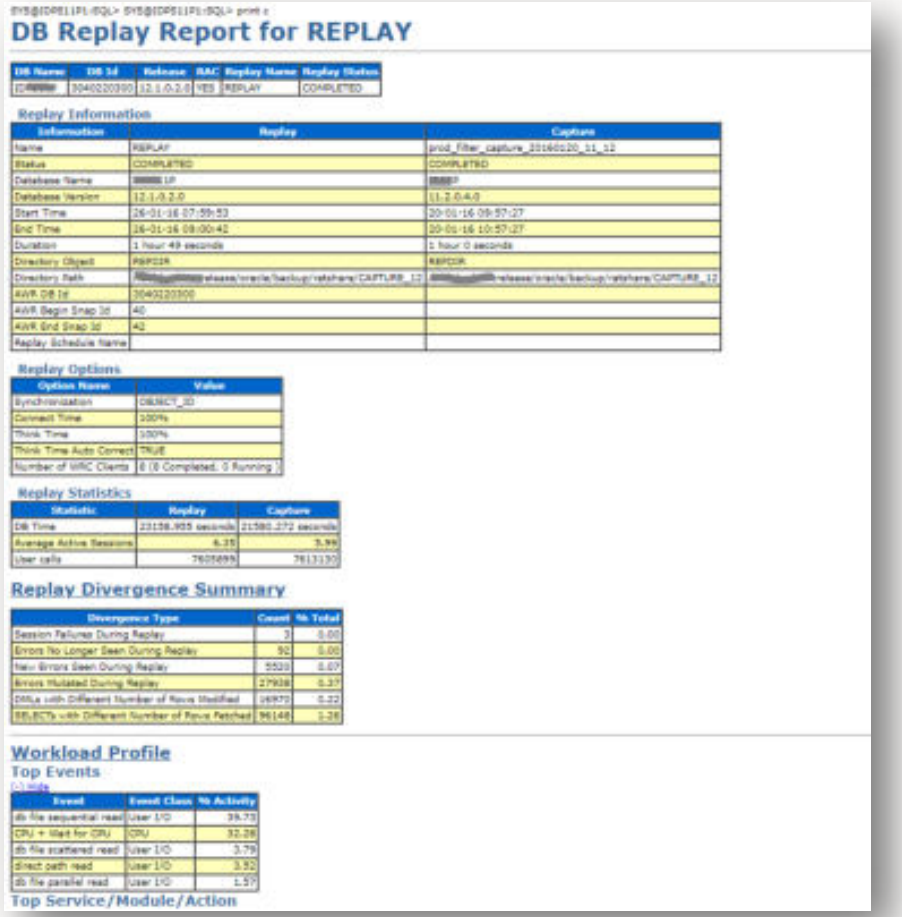
Database 19c Testing Guide, chapter 9

- Requires Enterprise Edition + Real Application Testing
- Available in Autonomous Database

Database Replay | Overview



Database Replay | Overview



DB Replay Report for REPLAY

Replay Information

Information	Replay	Capture
Name	REPLAY	prod_filter_capture_20160120_11_12
Status	COMPLETED	COMPLETED
Database Name	[redacted] UP	[redacted] P
Database Version	12.1.0.2.0	11.2.0.4.0
Start Time	26-01-16 07:59:53	20-01-16 09:57:27
End Time	26-01-16 08:00:42	20-01-16 10:57:27
Duration	1 hour 49 seconds	1 hour 0 seconds
Directory Object	RAPDIR	RAPDIR
Directory Path	[redacted]release/oracle/backup/rateshare/CAPTURE_12	[redacted]release/oracle/backup/rateshare/CAPTURE_12
AWR DB Id	3040320300	
AWR Begin Snap Id	40	
AWR End Snap Id	42	
Replay Schedule Name		

Replay Options

Option Name	Value
Synchronization	ORDER_D
Connect Time	100%
Think Time	100%
Think Time Auto Correct	TRUE
Number of WRC Clients	8 (8 Completed, 0 Running)

Replay Statistics

Statistic	Replay	Capture
DB Time	23138.935 seconds	21580.272 seconds
Average Active Sessions	6.35	5.99
User calls	7605699	7813130

Replay Divergence Summary

Divergence Type	Count	% Total
Session Failures During Replay	3	0.00
Errors No Longer Seen During Replay	92	0.00
New Errors Seen During Replay	5510	0.07
Errors Mutated During Replay	27908	0.37
DMLs with Different Number of Rows Modified	16972	0.22
SELECTs with Different Number of Rows Fetched	96148	1.26

Workload Profile

Top Events

Event	Event Class	% Activity
db file sequential read	User I/O	39.73
CPU + Wait for CPU	CPU	32.28
db file scattered read	User I/O	3.79
direct path read	User I/O	3.52
db file parallel read	User I/O	1.57

Top Service/Module/Action

Ulrike Schwinn on blogs.oracle.com

Database Replay | Overview

Compare Period Report: Capture vs. Replay

(+) General Information

(-) Replay Divergence

	Divergence Level	Percent of Calls That Diverged
Replay Divergence (compared to Capture)	HIGH	1.86%

(-) Main Performance Statistics

	Change in DB time	Capture total time	Replay total time	Capture % of DB time	Replay % of DB time
Database Time	-56.61%	51919.05 seconds	22530.42 seconds	100	100
CPU Time	-69.93%	14820.26 seconds	4395.94 seconds	28.18	19.51
User I/O Wait Time	106.92%	6208.74 seconds	12822.59 seconds	11.96	56.91
Cluster Wait Time	-84.02%	2151.54 seconds	343.9 seconds	4.14	1.53

(-) Top SQL/Call

(-) Top SQL by Change in DB Time

Force Matching Signature	example SQL_ID	Change in DB Time	Change in Average Response Time	Capture DB time	Replay DB time	example sql text
13059544831056126667	46km4ys2b7c0p	-3688.36 seconds	-100%	3688.36 seconds	0 seconds	(+) SELECT A2.3D FROM TEMP_APEX_MX_CHART_JOB S A1, TEMP_APEX_MX_C [...]

(-) Top Call by Change in DB Time

(-) Hardware Usage Comparison

(-) CPU Usage

system	CPU Sockets/Cores/Threads	Hosts' CPU Usage	Oracle Sessions on CPU	Oracle Run-queue Load
Capture	12/12/48	21.22%	4.14 active sessions	5.03 active sessions
Replay	16/16/64	12.99%	1.61 active sessions	1.39 active sessions

Ulrike Schwinn on blogs.oracle.com

Database Replay | Facts

1. Platform independent
2. RAC compliant - optionally, change number of nodes
3. Per-PDB capture/replay NEW IN 19c
4. Capture and replay across database releases

Database Replay | To Consider

1. Workload capture restrictions

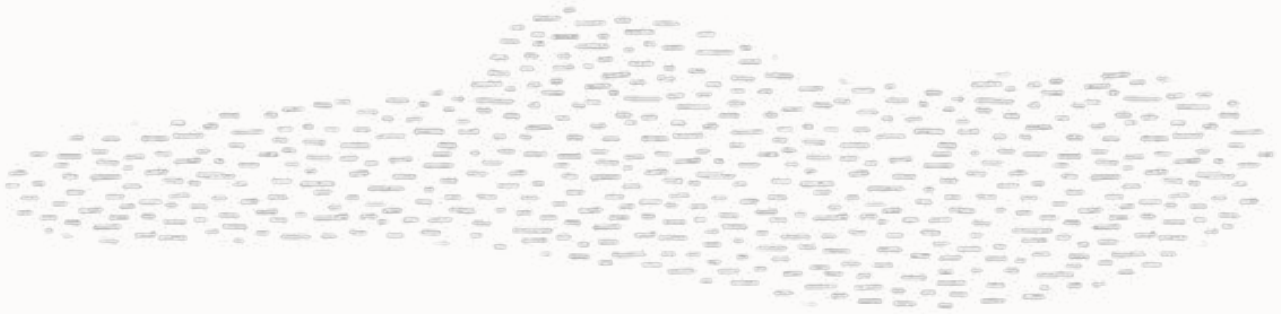
2. Not suitable with external dependencies
   - Database link, external tables, UTL_HTTP

3. Recommended to restart database before capture
   - Startup in restricted mode, capture automatically sets unrestricted mode

4. Work best from dedicated SCN
   - Data Pump `FLASHBACK_SCN` or restore to specific SCN

Database Replay | Info

- Start out with a small capture, then go full-scale
  - This allows you to iron-out issues and prepare for the full-scale capture
- Capture is a light-weight tracing, typically adds 4-5 % overhad
- Capture files are written to disk, disks must be fast
  - Otherwise it will affect the database
- Real-life example: 24h capture produced 4 TB of replay files
- Replay happens on a cold system
  - Use a "Replay Query Only" run to warm-up the system



Performance Stability Prescription

1. Collect
2. Compare
3. Analyze
4. Tune
5. Manage
6. Test

AGENDA

10:00	Introduction
10:15	Upgrade to Oracle Database 19c
10:45	Break
11:00	Upgrade to Oracle Database 19c
12:00	Lunch
13:00	Ensure Performance Stability
14:00	Break
14:15	**Hands-On Lab**
16:00	End

Hands-On Lab | Create Your Lab

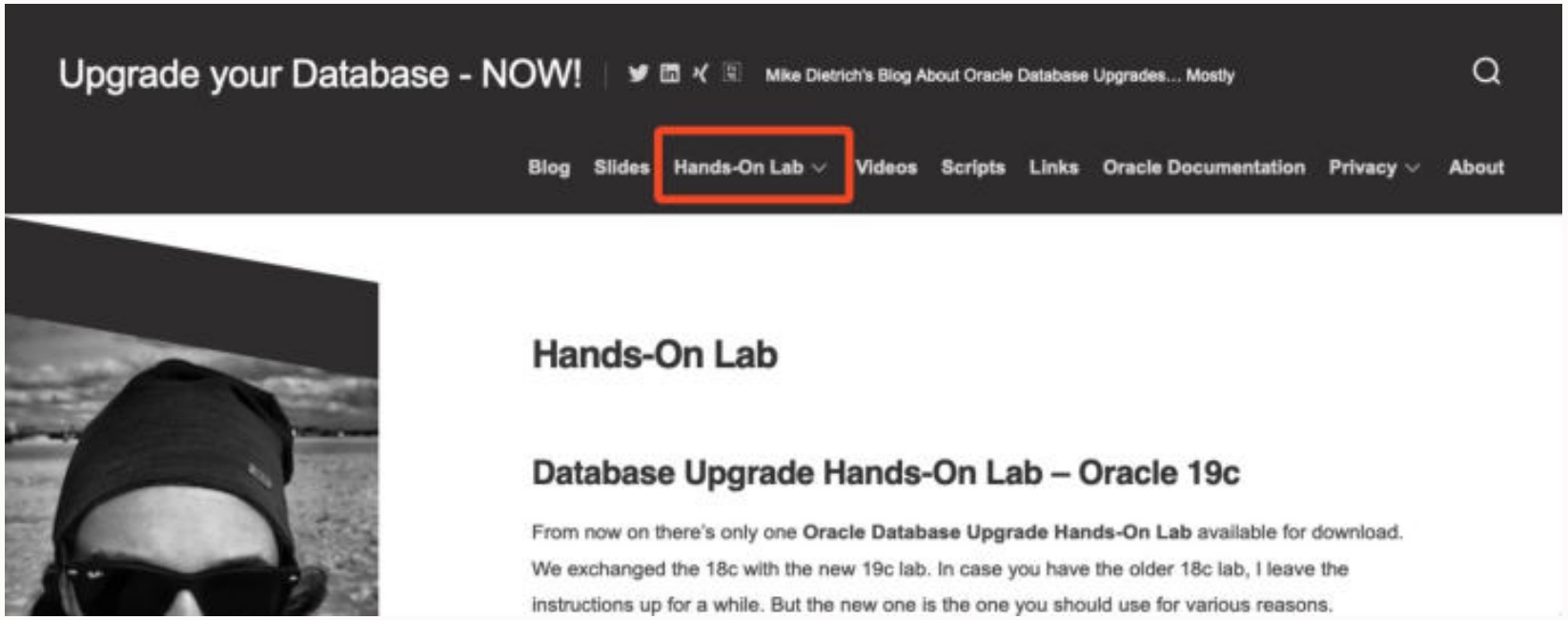
https://tinyurl.com/makeit2023hol

Workshop Code: 6868-JISO-UETI-ISAQ

Hands-On Lab | Create Your Lab

On Mike's blog (https://mikedietrichde.com) get

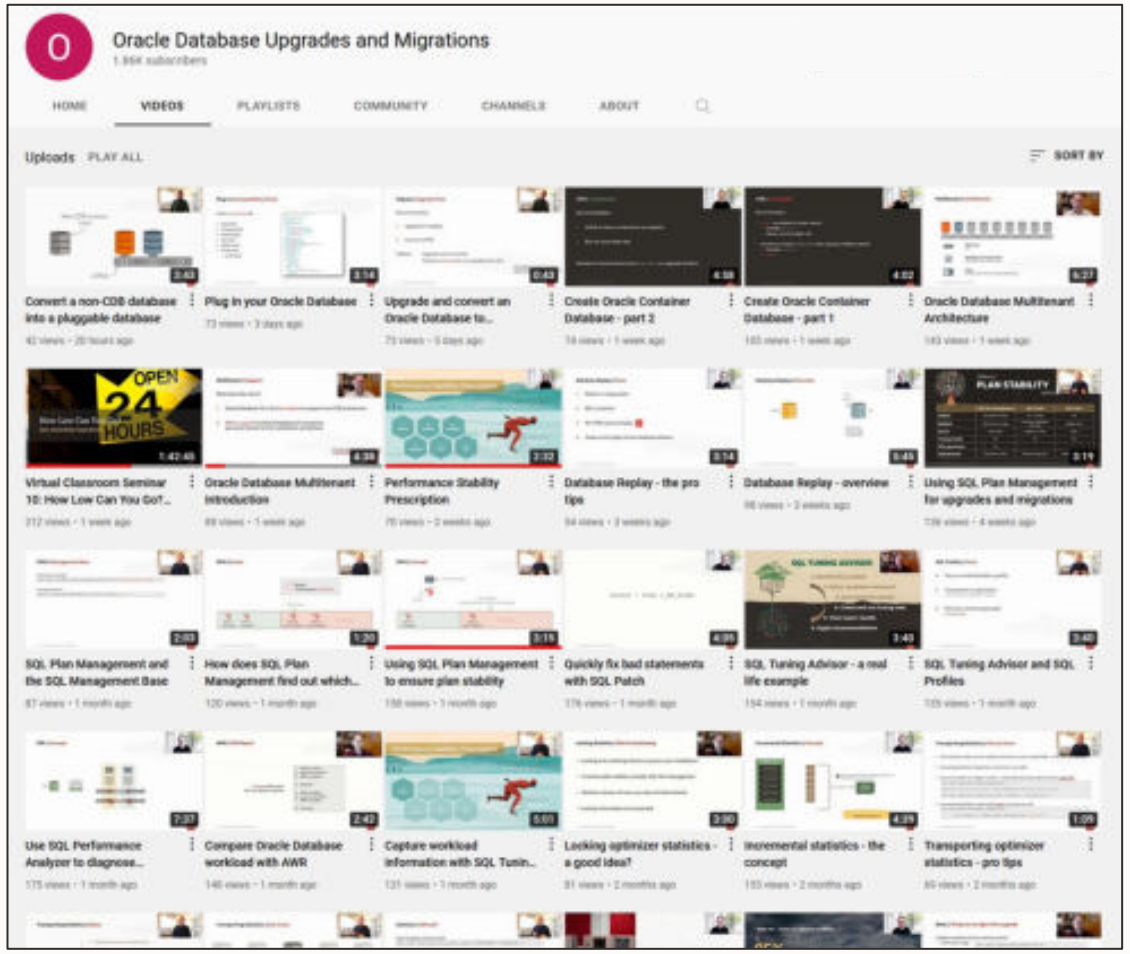
- the lab as Virtual Box image
- the instructions



AGENDA

10:00	Introduction
10:15	Upgrade to Oracle Database 19c
10:45	Break
11:00	Upgrade to Oracle Database 19c
12:00	Lunch
13:00	Ensure Performance Stability
14:00	Break
14:15	Hands-On Lab
16:00	**End**

YouTube | Oracle Database Upgrades and Migrations



Link

- 300+ 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

THANK YOU

Webinars:

https://MikeDietrichDE.com/videos

YouTube channel:

OracleDatabaseUpgradesandMigrations

THANK YOU

From SR to Patch

Insights into the Oracle Database Development Process

June 22, 2023 – 16:00 CEST

THANK YOU

Appendix

AutoUpgrade and TDE

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
```

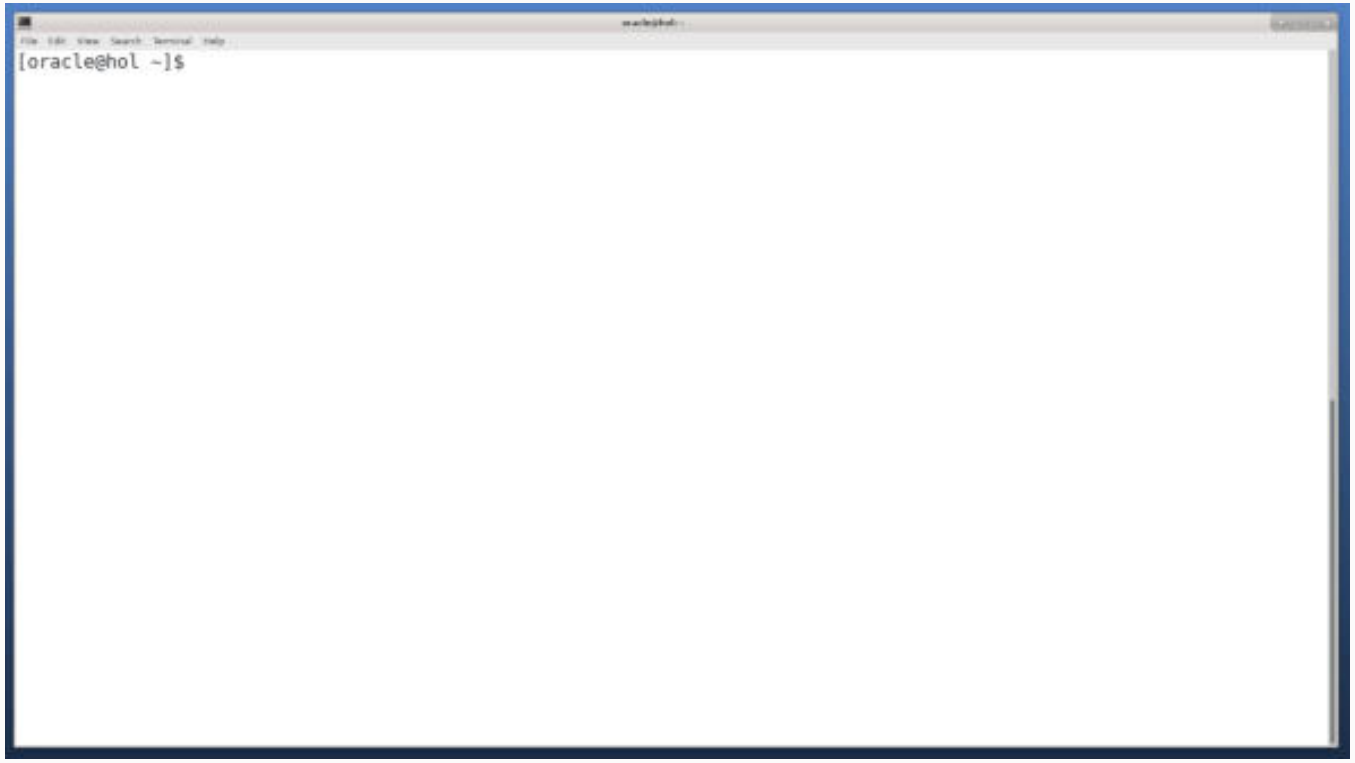

AutoUpgrade also supports converting an encrypted non-CDB to PDB

AutoUpgrade is compatible with Secure External Password Store

- Supported from Oracle Database 12.2

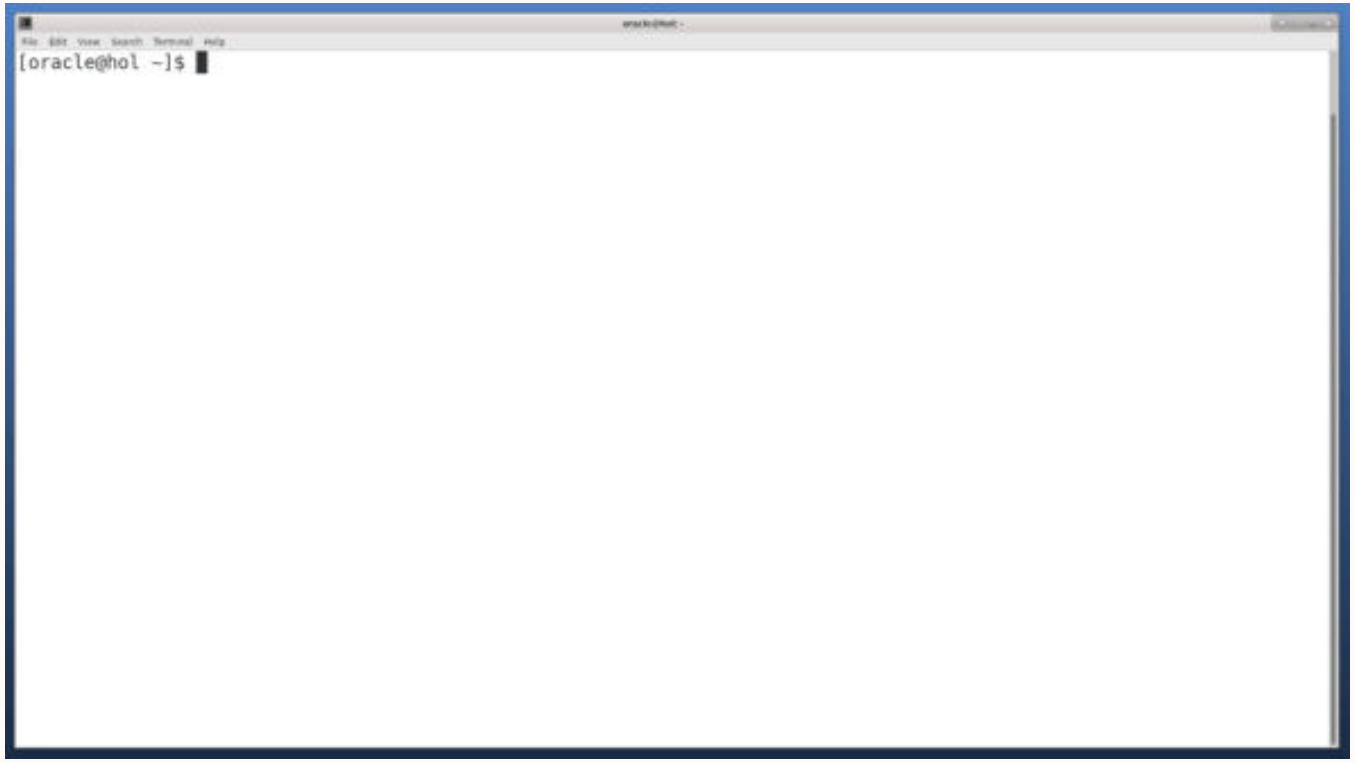
Pro tip: Get more details in blog post

TDE | Demo - Upgrading encrypted PDB



Watch on YouTube

TDE | Demo - Upgrading and converting to PDB



Watch on YouTube