

ORACLE



Performance Stability

After upgrade and migration

Daniel Overby Hansen

Senior Principal Product Manager





Daniel Overby Hansen

Senior Principal Product Manager
Cloud Migration

 <https://dohdatabase.com>

 @dohdatabase

 dohdatabase



Statistics

Statistics Advisor | Overview



New in Oracle Database 12.2

Give it a try, but ...

Be aware - potentially it will **eat** your SYSAUX tablespace

Statistics Advisor | Check



How much space is used?

```
SQL> select occupant_name, space_usage_kbytes
       from v$sysaux_occupants;
```

OCCUPANT_NAME	SPACE_USAGE_KBYTES
SM/ADVISOR	5901376
...	

```
SQL> select * from (
       select segment_name, owner, tablespace_name, bytes/1024/1024 "size(mb)", segment_type
       from dba_segments
       where tablespace_name='SYSAUX'
       order by bytes desc)
       where rownum <= 10;
```

SEGMENT_NAME	OWNER	TABLESPACE	SIZE (MB)	SEGMENT_TYPE
WRI\$_ADV_OBJECTS	SYS	SYSAUX	3600	TABLE
WRI\$_ADV_OBJECTS_IDX_01	SYS	SYSAUX	1400	INDEX



Statistics Advisor | Disable



If you want to disable the automatic statistics advisor job

1. In 21c, disable the auto task

```
SQL> exec dbms_stats.set_global_prefs('AUTO_STATS_ADVISOR_TASK','FALSE');
```

2. 19c, request backport of bug 26749785 and then disable

3. Or, disable with workaround

```
SQL> begin
    dbms_advisor.set_task_parameter('AUTO_STATS_ADVISOR_TASK','_AUTO_MMON_INTERVAL',2147483647);
    dbms_advisor.set_task_parameter('AUTO_STATS_ADVISOR_TASK','_AUTO_STATS_INTERVAL',2147483647);
end;
/
```

Pro tip: If you disable the automatic statistics advisor job, you can still do manual executions



Statistics Advisor | Purge



Refer to these two MOS notes:

1. [SYSAUX Tablespace Grows Rapidly After Upgrading Database to 12.2.0.1 or Above Due To Statistics Advisor \(Doc ID 2305512.1\)](#)
2. [How To Purge Optimizer Statistics Advisor Old Records From 12.2 Onwards \(Doc ID 2660128.1\)](#)



Statistics Advisor | References



- Mike Dietrich blog post: [Oracle Optimizer Statistics Advisor in Oracle Database 12.2.0.1](#)
- MOS note: [SYSAUX Tablespace Grows Rapidly After Upgrading Database to 12.2.0.1 or Above Due To Statistics Advisor \(Doc ID 2305512.1\)](#)
- MOS note: [Optimizer Statistics Advisor In 12.2 \(Quick Overview\) \(Doc ID 2259398.1\)](#)
- Oracle Database 19c SQL Tuning Guide, [Analyzing Statistics Using Optimizer Statistics Advisor](#)



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;
/
```



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



Statistics | Gather Statistics Before Upgrade

Check when dictionary stats have been gathered the last time

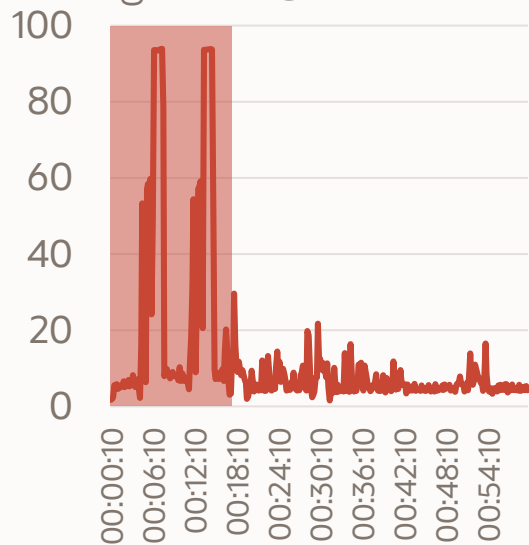
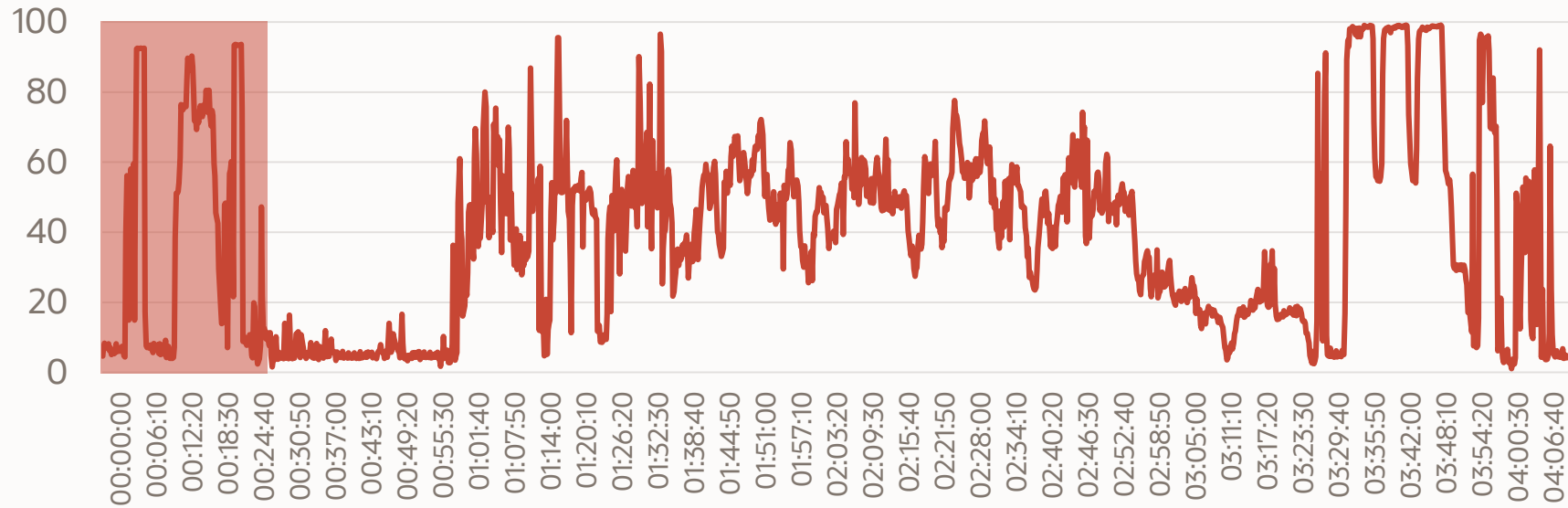
```
SELECT
  to_char(max(end_time), 'dd-mon-yy hh24:mi') latest, operation
FROM
  dba_optstat_operations
WHERE
  operation in ('gather_dictionary_stats', 'gather_fixed_objects_stats')
GROUP BY
  operation;
```



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**

Statistics | Gather Statistics Before Upgrade



Gathering stats in advance saves **12 minutes**
Dictionary and fixed objects

Statistics | Effect of Having Good Stats During Upgrade

- The larger the dictionary, the bigger the effect

	Duration	Reduction
No dictionary and fixed objects stats	15 min 55 sec	
Gathered dictionary and fixed objects stats	14 min 10 sec	11 %
Gathered schema and cluster index stats	13 min 41 sec	3.4 % to previous
Total downtime saved	2 min 14 sec	14 % overall

- This example has been done with one of the tiny Hands-On Lab databases

Statistics | Effect of Having Good Stats During Upgrade

- Upgrade duration for Oracle E-Business Suite

	Duration	Reduction
No dictionary and fixed objects stats	10 hrs 56 min 52 sec	
Gathered dictionary and fixed objects stats	52 min 42 sec	93 %
Gathered schema and cluster index stats	52 min 25 sec	0.5 % to previous
Total downtime saved	10 hrs 4 min 14 sec	93.5 % overall

Locking Statistics | Overview

You **finally** did it. You produced the **ultimate stats**.

It is time to **lock** 'em!

Locking Statistics | Use Cases

”

You can lock statistics to prevent them from changing.

[Database 19c SQL Tuning Guide, chapter 15](#)

- Certain static environments
- Highly volatile tables
- Enable use of dynamic statistics
- ... and all the exceptions

Locking Statistics | Show it

Lock table statistics

```
SQL> EXEC DBMS_STATS.LOCK_TABLE_STATS(ownname=>'MYAPP', tabname=>'MY_VOLATILE_TAB1');
```

You can also lock at:

- Schema-level
- Partition-level

You can also unlock statistics

Pro tip: Locking table statistics also lock index and partition statistics



Locking Statistics | **Worth mentioning**

- Locking and unlocking statistics causes cursor invalidation
- To achieve plan stability consider SQL Plan Management
- Statistics advisor will warn you about locked statistics
- Locking information is not exported



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

```
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 Best Practices

A prescription to ensure performance stability

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

- [Using This Document](#)
- [Section 1: Common Database Initialization Parameters For All Releases](#)
- [Section 2: Release-Specific Database Initialization Parameters For Oracle 11g Release 2](#)
- [Section 3: Release-Specific Database Initialization Parameters For Oracle 12c Release 1](#)
- [Section 4: Release-Specific Database Initialization Parameters For Oracle 19c](#)
- [Section 5: Additional Database Initialization Parameters For Oracle E-Business Suite Release 12.2](#)
- [Section 6: Using System Managed Undo \(SMU\)](#)
- [Section 7: Temporary Tablespace Setup](#)
- [Section 8: Database Initialization Parameter Sizing](#)

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



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

In of all cases

95%

an "upgrade problem" in **reality** is a **performance issue after upgrade**, or **not** a database **upgrade** topic.

There is exactly one way to mitigate the risk:

TESTING!

Testing | Typical Mistakes

- Only 10% of real data used
- Data sets artificially created
- Tests done on a laptop
- No testing tools used
- Data from 6 months ago
- No statistics refreshed
- Testing?? Waste of time!
 - Real **experts** fix it **after** go-live ...





Photo by [Alex Motoc](#) on [Unsplash](#)

Testing

Need a new test environment?

Test Environments | Ideas



Snapshot standby database

- Leverage existing standby databases
- Increase RTO a little - and gain a *free* test environment



Test Environments | Ideas



Hybrid Data Guard in Oracle Cloud Infrastructure

- Create as many as you like
- Pay-as-you-go



Test Environments | Ideas



CloneDB

- Copy-on-write
- Uses image copies of data files stored on NFS, delta is written locally



Test Environments | Ideas



Snapshot Copy PDBs

- Requires compatible storage system
- Or, use CloneDB functionality (requires source PDB is read-only)



Test Environments | Ideas



Split Mirror Clone PDBs

- Requires ASM and Oracle Database 18c
- A point-in-time version of a PDB



Test Environments | Ideas



Exadata Sparse Snapshots

- Space savings - fast provisioning
- Clone still has access to Exadata storage features



Best Practice | **COMPATIBLE** vs **Optimizer**

- `COMPATIBLE` and `OPTIMIZER_FEATURES_ENABLE`
 - Fully independent from each other
 - Set `COMPATIBLE` to the default of the release, e.g. **19.0.0**
 - Change `OPTIMIZER_FEATURES_ENABLE` only when you have to
 - Avoid it if possible
 - This is **not** a Swiss Army knife!
 - You will turn off a lot of great features

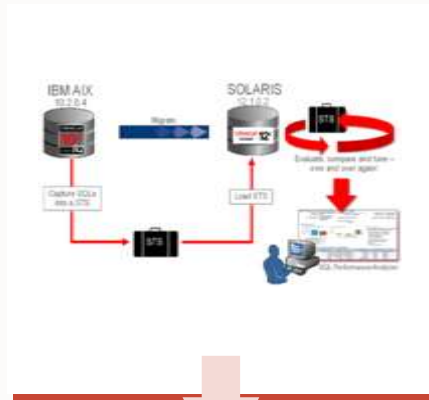
Testing Tools | Workflow



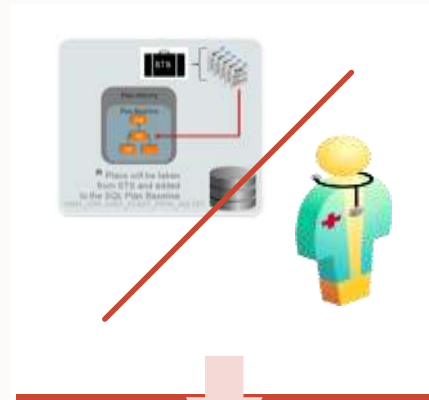
Collect execution plans before upgrade



Compare AWR Snapshots



Verify them with **SQL Performance Analyzer**



Regressed plans?
SQL Plan Management
SQL Tuning Advisor



Verify functionality and performance with **Database Replay**

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](#)

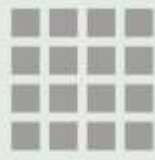
SQL Tuning Set | Definition



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_SOLTUNE](#) 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*

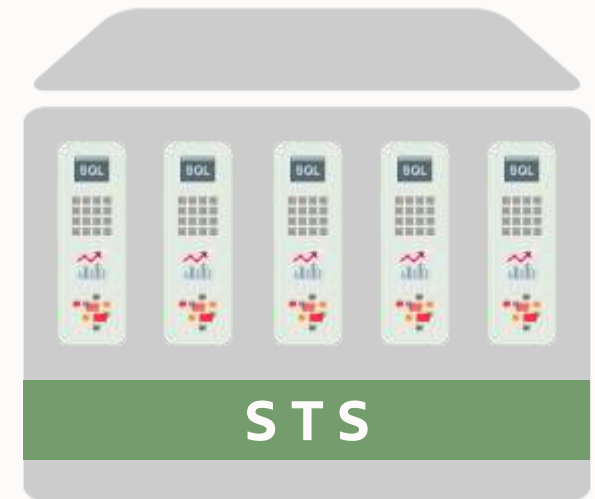


SQL Tuning Set | Capture



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:

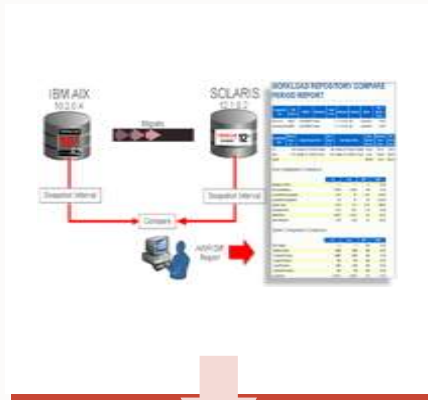
<i>ADD_SQLSET_REFERENCE</i>	<i>CAPTURE_CURSOR_CACHE_SQLSET</i>
<i>CREATE_SQLSET</i>	<i>CREATE_STGTAB_SQLSET</i>
<i>DELETE_SQLSET</i>	<i>DROP_SQLSET</i>
<i>LOAD_SQLSET</i>	<i>PACK_STGTAB_SQLSET</i>
<i>REMOVE_SQLSET_REFERENCE</i>	<i>SELECT_CURSOR_CACHE</i>
<i>SELECT_SQLSET</i>	<i>SELECT_WORKLOAD_REPOSITORY</i>
<i>UNPACK_STGTAB_SQLSET</i>	<i>UPDATE_SQLSET</i>

[Database 19c Database Licensing Information User Manual](#)

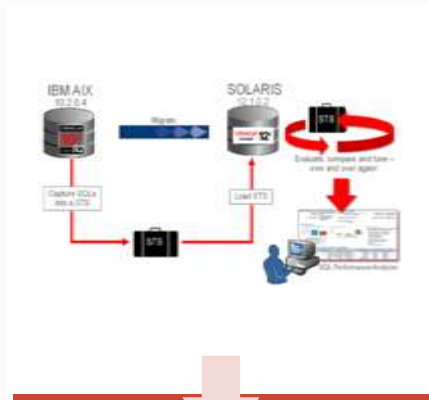
Testing Tools | SQL Plan Management



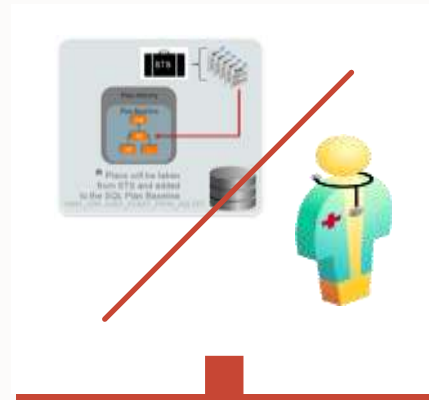
Collect execution plans before upgrade



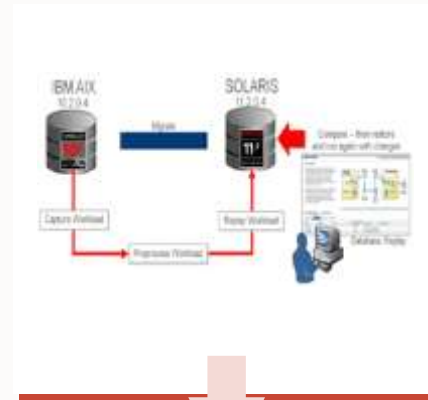
Compare AWR Snapshots



Verify them with **SQL Performance Analyzer**



Regressed plans?
SQL Plan Management



Verify functionality and performance with **Database Replay**

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



SQL SQL

Repeatable SQL

No plans in baseline

Plan A Filter



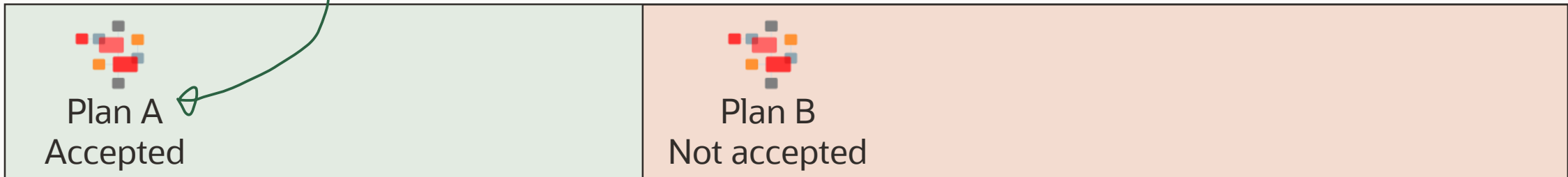
New plans in plan history

Plan A is used

Filtering changed

- Slow statistics
- SQL text parameters
- SQL plan history
- Action

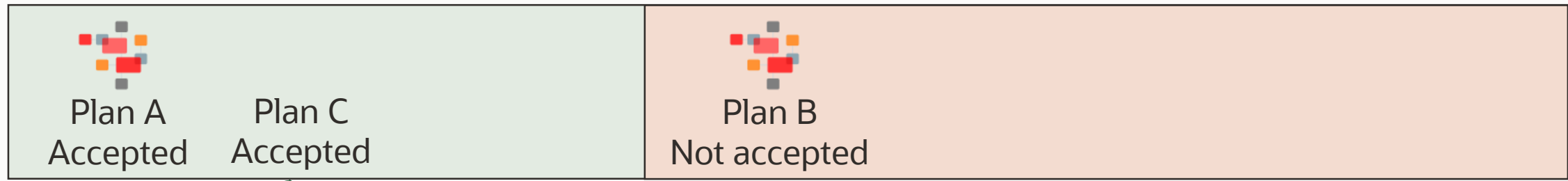
```
OPTIMIZER_USE_SQL_PLAN_BASELINES=TRUE
```



SPM | Load from STS



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




Automatically
accepted

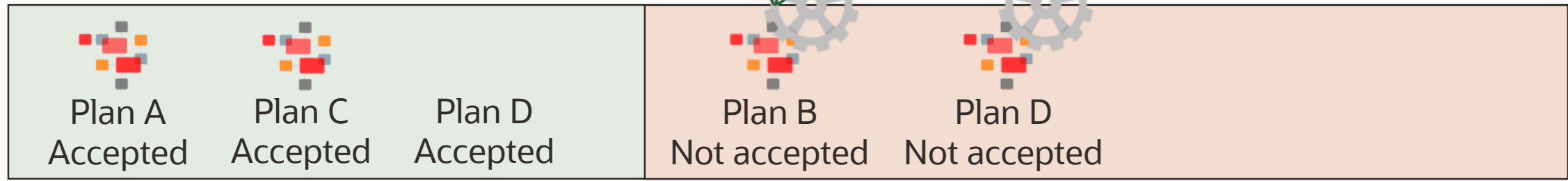


SPM | Evolve



 **Result:**
Performance **better**

Test execute Test execute
Plan stays



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 | 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 | 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	

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

AUTO_CAPTURE_PARSING_SCHEMA_NAME	
AUTO_CAPTURE_SQL_TEXT	
AUTO_SPM_EVOLVE_TASK	OFF

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

PLAN_RETENTION_WEEKS	53
SPACE_BUDGET_PERCENT	10



SQL Plan Management | Plan Stability

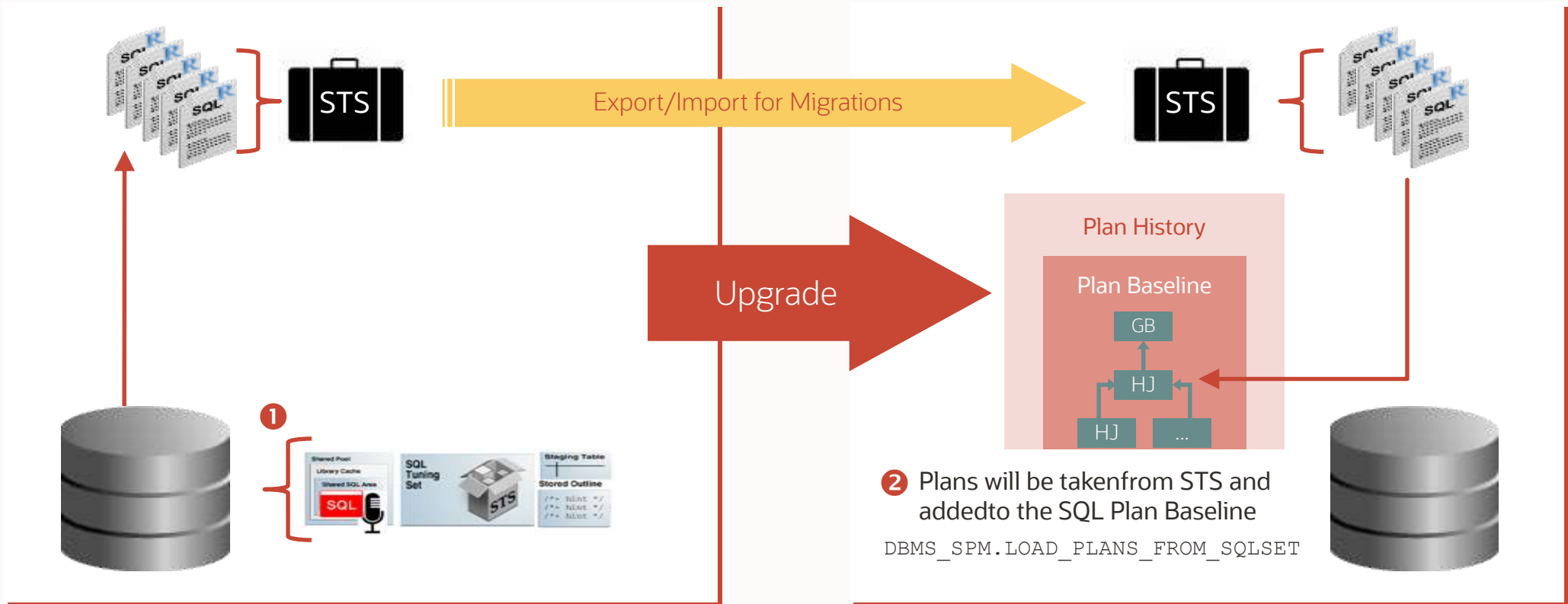




Photo by andre spilborghs on Unsplash

The "hidden" gems

Fix it when you upgrade

Automatic SQL Tuning Set

In Oracle 19.7.0, a **Automatic SQL Tuning Set** gets populated

- Some customers reported high growth and consumption in SYSAUX

```
select to_char(max(last_schedule_time), 'DD-MON-YY hh24:mi') LATEST, task_name, status, enabled from
dba_autotask_schedule_control group by task_name, status, enabled ;
```

LATEST	TASK_NAME	STATUS	ENABLED
27-MAY-20 20:00	Auto STS Capture Task	SUCCEEDED	TRUE
15-APR-20 00:16	Auto SPM Task	SUCCEEDED	FALSE

- See: https://mikedietrichde.com/2020/05/28/do-you-love-unexpected-surprises-sys_auto_sts-in-oracle-19-7-0/
- If you want to disable it:
 - BEGIN
DBMS_AUTO_TASK_ADMIN.DISABLE(client_name=>'Auto STS Capture Task', operation=>NULL, window_name=>NULL);
END;
/
• Task is not enabled by default from Oracle 19.8.0 on

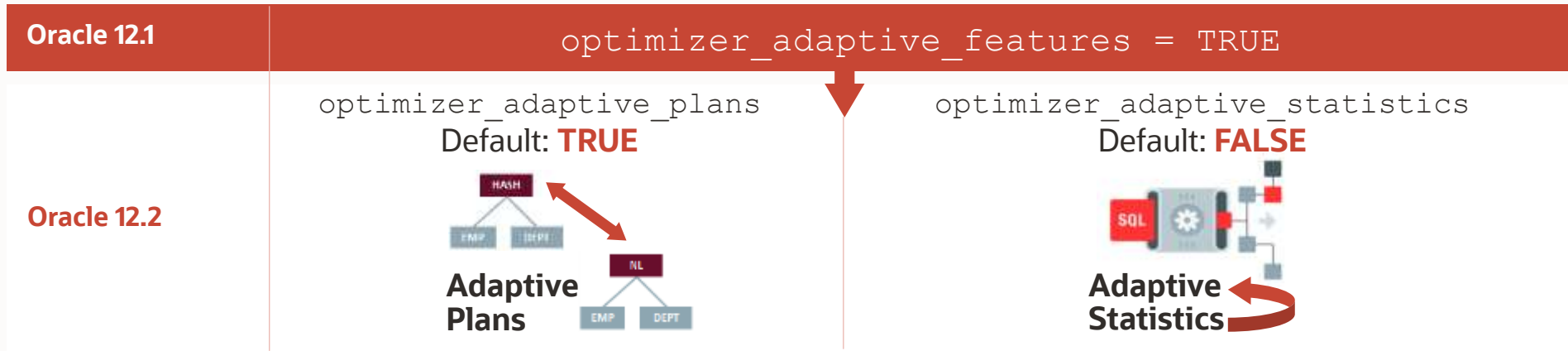
Oracle 19c | `_cursor_obsolete_threshold`

- [MOS Note: 2431353.1](#)
High Version Counts for SQL statements (>1024) post upgrade To 12.2 and above causing database slow performance
 - Since Oracle 12.2.0.1, the old default of **1024** has been raised to **8192**
 - This may lead to various drastic performance issues
 - Recommendation:
 - **Set it to the old default to avoid** mutex concurrency and other issues:
`_cursor_obsolete_threshold=1024`

Oracle 19c | `deferred_segment_creation`

- [MOS Note: 1216282.1](#)
All sorts of issues and bugs - from corruption to mutex contention to customer scripts not operating correctly anymore
 - Recommendation:
Set it to
`deferred_segment_creation=false`
unless you have a valid reason

Oracle 19c | Adaptive Features



- Recommendation: Set `optimizer_adaptive_statistics` explicitly in your SPFILE
- See:
 - <https://mikedietrichde.com/2018/01/18/additional-info-adaptive-features-fixes-oracle-12-1-0-2/>
 - [MOS Note: 2187449.1 - Recommendations for Adaptive Features in Oracle Database 12.1](#)

Oracle 19c | `_sql_plan_directive_mgmt_control`

- [MOS Note: 2209560.1 - How To Disable SQL Plan Directive \(SPD\)](#)
- In order to **fully disable** SQL Plan Directives, you need to set:
 - `_sql_plan_directive_mgmt_control=0`
- Otherwise the database collects SPDs in the background, but won't use it
 - Having `optimizer_adaptive_statistics=false` - which is the default - disables **only** the usage of SPDs but **not** their creation

Now RELAX ... Stay Calm ...

And open an SR with
Oracle Support in case
of real trouble



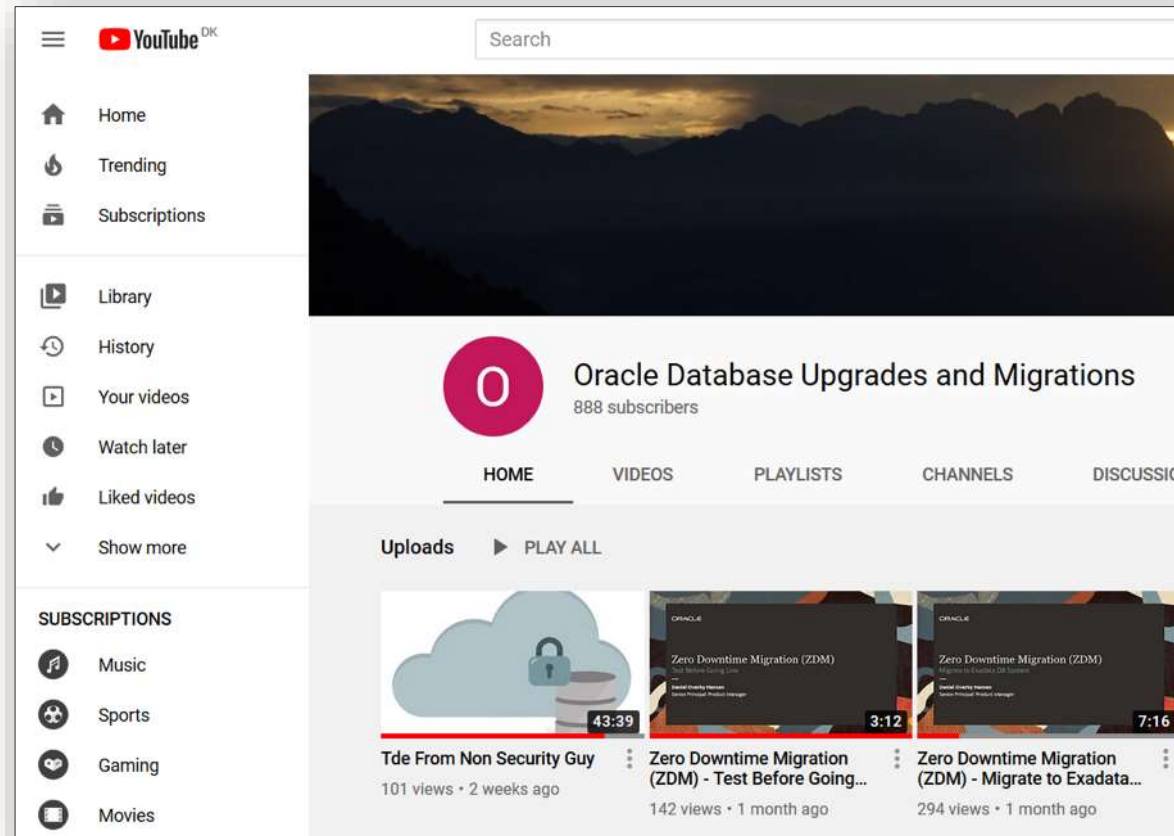
Photo by [Jonathan Velasquez](#) on [Unsplash](#)

Webinar **Performance Stability, tips, tricks & underscores**

Thursday 4 March 2021
19:00 - 21:00 CET


[Sign up](#)

YouTube | Oracle Database Upgrades and Migrations



[YouTube Channel](#)



Upgrade your Database - NOW!  Mike Dietrich's Blog About Oracle Database Upgrades... Mostly

Blog Slides **Hands-On Lab** Events Papers / Docs Videos Scripts Links Oracle Documents

HOL 19c – Main Index Page

« PREVIOUS » MAIN INDEX HOL 19c » NEXT »

[Download the Oracle 19c Hands-On Lab](#)

You can use this page as an index page for the 19c lab!

- Setup
- Load
- Capture and Preserve SQL
- AutoUpgrade

- 1. Setup
- 2. Load
- 3. Capture and Preserve
- 4. AutoUpgrade
- 5. AWR Diff
- 6. SQL Perf Analyzer
- 7. SQL Plan Mgmt
- 8. SQL Tuning Advisor
- 10. Plugin UPGR => CDB2
- 11. Migrate FTEX
- 12. Un-/ Plug / Upgrade
- 13. Fallback Strategies
- 15. AutoUpgrade – Special



Thank you!

