ORACLE

Performance and Migrations

A Practical Guide

Oracle

DBAs

run the world

Daniel Overby Hansen

Distinguished Product Manager

dohdatabase

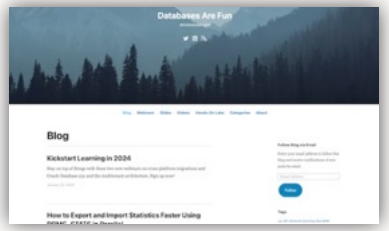
@dohdatabase.com

https://dohdatabase.com

Find Slides and Much More on Our Blogs

MikeDietrichDE.com

Mike.Dietrich@oracle.com



dohdatabase.com

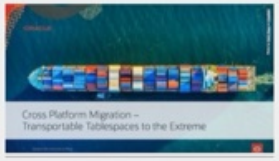
Daniel.Overby.Hansen@oracle.com

DBArj.com.br

Rodrigo.R.Jorge@oracle.com

AlexZaballa.com

Alex.Zaballa@oracle.com

Virtual Classroom Seminars	Slides
Episode 16 (replaces Episode 1 from Feb 2021) Oracle Database Release and Patching Strategy for 19c and 23c *115 minutes – May 10, 2023*	
Episode 17 From SR to Patch – Insights into the Oracle Database Development process *55 minutes – June 22, 2023*	
Episode 18 Cross Platform Migration – Transportable Tablespaces to the Extreme *145 min – February 22, 2024*	
Episode 19 Move to Oracle Database 23ai – Everything you need to know about Multitenant PART 1 *145 min – May 16, 2024*	
Episode 20 Move to Oracle Database 23ai – Everything you need to know about Multitenant PART 2 *100 min – June 28, 2024*	

Recorded Web Seminars

https://MikeDietrichDE.com/videos

More than 40 hours of technical content, on-demand, anytime, anywhere

What's the Worst That Can Happen?


Photo by Urip Dunker on Unsplash

Photo by Dave Hoefler on Unsplash

What's the worst that can happen after the migration?

Photo by Craig Pattenaude on Unsplash

What if we had a *one-button* tool that points out the bad statements…

SQL Performance Analyzer

Database Licensing Information User Manual

1 Licensing Information

1.1 Introduction

This Licensing Information document is a part of the product or program documentation under the terms of your Oracle license agreement and is intended to help you understand the program editions, entitlements, restrictions, prerequisites, special license rights, and/or separately licensed third party technology terms associated with the Oracle software program(s) covered by this document (the "Program(s)"). Entitled or restricted use products or components identified in this document that are not provided with the particular Program may be obtained from the Oracle Software Delivery Cloud website (https://edelivery.oracle.com) or from media Oracle may provide. If you have a question about your license rights and obligations, please contact your Oracle sales representative, review the information provided in Oracle's Software Investment Guide (http://www.oracle.com/us/

Oracle Real Application Testing	Extra cost option: **EE-ES** Included option: **BaseDB EE**, **BaseDB EE-HP**, **BaseDB EE-EP**, **ExaDB**	Oracle Real Application Testing includes the following features: • Database Replay • SQL Performance Analyzer (SPA) • Database Migration Planner • Database Migration Workbench **Database Replay** • The Oracle Real Application Testing license is required on both capture and replay systems for Database Replay and is charged by the total number of CPUs on those systems. Licensing is also charged by the total number of CPUs on both systems when the capture is done on any read-only standby database and the workload is replayed on a True Cache. • Use of Capture and Replay ASH Analytics Reports, Compare Period ADDM Reports, and Replay Compare Period Reports also requires an Oracle Diagnostics Pack license. • An Oracle Real Application Testing license permits you to access Database Replay functionality through Oracle Enterprise Manager, as well as through the following database server command-line APIs: `DBMS_WORKLOAD_CAPTURE` package and `DBMS_WORKLOAD_REPLAY` package. The use of the `DBMS_WORKLOAD_REPLAY.COMPARE_PERIOD_REPORT()` function also requires a license for Oracle Diagnostics Pack. **SQL Performance Analyzer (SPA)** An Oracle Real Application Testing license permits you to access SQL Performance Analyzer functionality through Oracle Enterprise Manager, as w as through the following database server command-line API: `DBMS_SQLPA`

PERFORMANCE STABILITY

1

CAPTURE

2

ANALYZE

3

FIX

4

REMEDY

1

CAPTURE

Capture workload information into SQL Tuning Set

2

ANALYZE

3

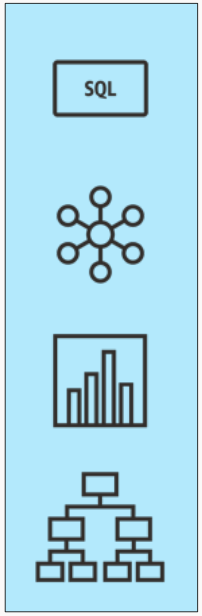
FIX

4

REMEDY

SQL Tuning Set | Definition

SQL Tuning Set



SQL statement

Context

Statistics

Plans

```
--Capture directly from cursor cache at regular intervals
exec DBMS_SQLSET.CAPTURE_CURSOR_CACHE ( ... );

--One time sample from cursor cache
exec DBMS_SQLSET.SELECT_CURSOR_CACHE ( ... );
```


```
--Capture directly from cursor cache at regular intervals
exec DBMS_SQLSET.CAPTURE_CURSOR_CACHE ( ... );

--One time sample from cursor cache
exec DBMS_SQLSET.SELECT_CURSOR_CACHE ( ... );

--Capture from AWR
SQL> exec DBMS_SQLSET.SELECT_WORKLOAD_REPOSITORY ( ... );
```


```
SQL> select name, statement_count from dba_sqlset;

NAME                             STATEMENT_COUNT
_______________________________ _________________
SALES_APP_WORKLOAD                            43
```

```
SQL> select view_name from dba_views where view_name like 'DBA%SQLSET%';

VIEW_NAME

_____________________________
DBA_SQLSET
DBA_SQLSET_REFERENCES
DBA_SQLSET_STATEMENTS
DBA_SQLSET_BINDS
DBA_SQLSET_PLANS
```


Automatic SQL Tuning Set captures statements at regular intervals

- Off by default

```
SQL> select name, statement_count from dba_sqlset;

NAME                           STATEMENT_COUNT
------------------------------ ---------------
SALES_APP_WORKLOAD                          43
SYS_AUTO_STS                              3320
```

Gather at least a full month of workload data

- Assist in testing your database
- Useful in solving post-migration performance problems

```
EXEC DBMS_SQLSET.PACK_STGTAB ( ... );

...

EXEC DBMS_SQLSET.UNPACK_STGTAB ( ... );
```

```
-- If you have many bind variables in your queries, increase the number of binds
-- to capture from 400 (default) to 3999 (max)

alter system set "_cursor_bind_capture_area_size"=3999;
```

Further Information

SQL Tuning Set

- Blog post: Oracle SQL Tuning Sets (STS) – The foundation for SQL Tuning
- Blog post: What is the automatic SQL tuning set?

1

CAPTURE

2

ANALYZE

Analyze performance after migration using SQL Performance Analyzer

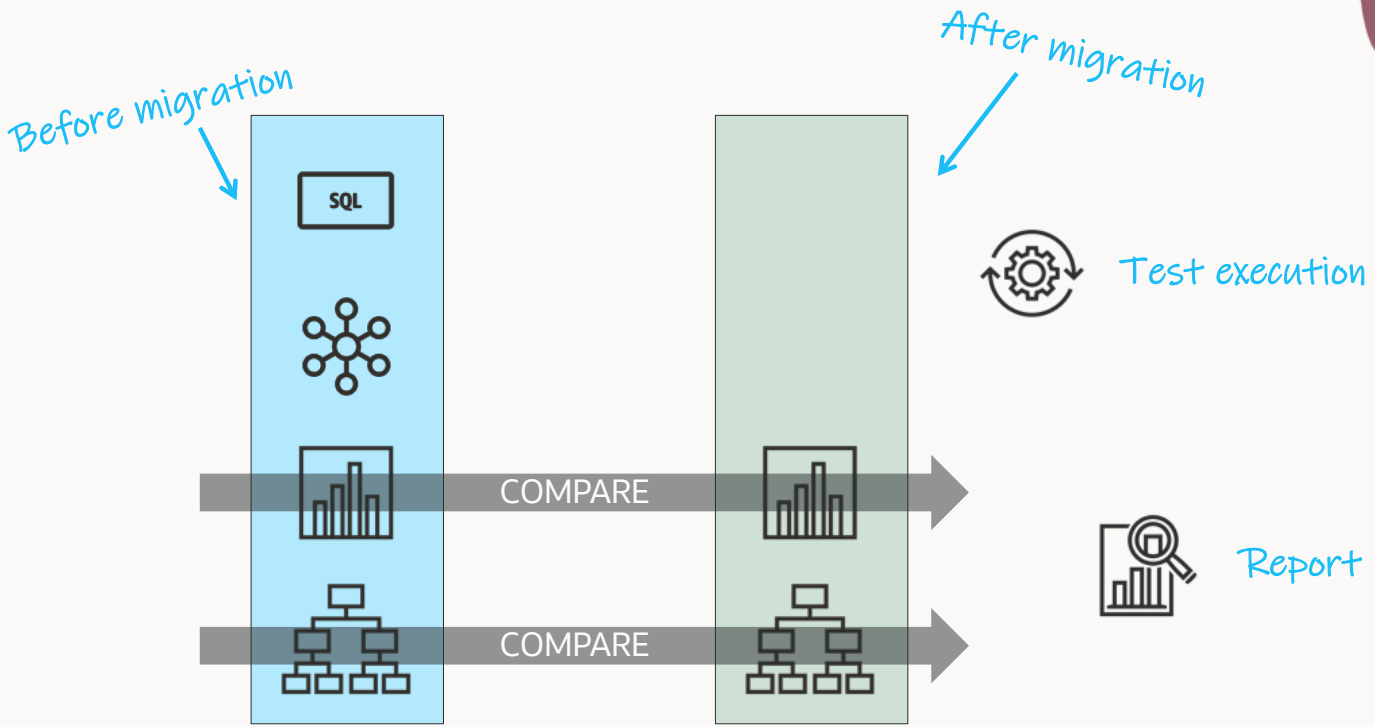
3

FIX

4

REMEDY

SQL Performance Analyzer



Report Summary

Projected Workload Change Impact:

Overall Impact	**4.64%**
Improvement Impact	**30.98%**
Regression Impact	**-26.35%**

SQL Statement Count

SQL Category	SQL Count	Plan Change Count
Overall	43	5
Improved	1	1
Regressed	1	1
Unchanged	30	3
Unsupported	11	0

Top 32 SQL Sorted by Absolute Value of Change Impact on the Workload

object_id	sql_id	Impact on Workload	Execution Frequency	Metric Before	Metric After	Impact on SQL	Plan Change
83	f90zn75aphu4w	30.98%	2828	139319.505304102	41950	69.89%	y
47	0cwuxyv314wcg	-26.35%	18254	981.459680070122	13809	-1306.99%	y
80	csv0xdm9c394t	-.36%	2734	4689.26664228237	5868	-25.14%	n
60	4hbzjyh4p336s	.21%	2818	668.862668559262	10	98.5%	n
76	a8ntu3081hfgw	-.18%	2818	262.609297374024	828	-215.3%	y

Report Summary

Projected Workload Change Impact:

Overall Impact : 4.64%
Improvement Impact : 30.98%
Regression Impact : -26.35%

SQL Statement Count

SQL Category	SQL Count	Plan Change Count
Overall	43	5
Improved	1	1
Regressed	1	1
Unchanged	30	3
Unsupported	11	0

Top 32 SQL Sorted by Absolute Value of Change Impact on the Workload

object_id	sql_id	Impact on Workload	Execution Frequency	Metric Before	Metric After	Impact on SQL	Plan Change
83	f90zn75aphu4w	30.98%	2828	139319.505304102	41950	69.89%	y
47	0cwuxyv314wcg	-26.35%	18254	981.459680070122	13809	-1306.99%	y
80	csv0xdm9c394t	-.36%	2734	4689.26664228237	5868	-25.14%	n
60	4hbzjyh4p336s	.21%	2818	668.862668559262	10	98.5%	n
76	a8ntu3081hfgw	-.18%	2818	262.609297374024	828	-215.3%	y

Report Summary

Projected Workload Change Impact:

Overall Impact	: 4.64%
Improvement Impact	: 30.98%
Regression Impact	: -26.35%

SQL Statement Count

SQL Category	SQL Count	Plan Change Count
Overall	43	5
Improved	1	1
Regressed	1	1
Unchanged	30	3
Unsupported	11	0

Top 32 SQL Sorted by Absolute Value of Change Impact on the Workload

object_id	sql_id	Impact on Workload	Execution Frequency	Metric Before	Metric After	Impact on SQL	Plan Change
83	f90zn75aphu4w	30.98%	2828	139319.505304102	41950	69.89%	y
47	0cwuxyv314wcg	-26.35%	18254	981.459680070122	13809	-1306.99%	y
80	csv0xdm9c394t	-.36%	2734	4689.26664228237	5868	-25.14%	n
60	4hbzjyh4p336s	.21%	2818	668.862668559262	10	98.5%	n
76	a8ntu3081hfgw	-.18%	2818	262.609297374024	828	-215.3%	y

SQL Details:

Object ID	: 47
Schema Name	: TPCC
Container Name	: Unknown (con_dbid: 344460545)
SQL ID	: 0cwuxyv314wcg
Execution Frequency	: 18254
SQL Text	: SELECT ROWID FROM CUSTOMER WHERE C_W_ID = :B3 AND C_D_ID = :B2 AND C_LAST = :B1

Bind Variables:

1 - (NUMBER): 3
2 - (NUMBER): 7
3 - (VARCHAR2): ESEEINGOUGHT

Execution Statistics:

Stat Name	Impact on Workload	Value Before	Value After	Impact on SQL
elapsed_time	-26.35%	.000981	.013809	-1306.99%
parse_time			.000477	
cpu_time	-106.5%	.000494	.013743	-2681.43%
user_io_time			0	
buffer_gets	-130.82%	253	7252	-2759.03%
cost	-745969.19%	255	1982	-677.25%
reads	0%	0	0	0%
writes	0%	0	0	0%

SQL Details:

Object ID : 47
Schema Name : TPCC
Container Name : Unknown (con_dbid: 344460545)
SQL ID : 0cwuxyv314wcg
Execution Frequency : 18254
SQL Text : SELECT ROWID FROM CUSTOMER WHERE C_W_ID = :B3 AND C_D_ID = :B2 AND C_LAST = :B1

Bind Variables:

1 - (NUMBER): 3
2 - (NUMBER): 7
3 - (VARCHAR2): ESEEINGOUGHT

Execution Statistics:

Stat Name	Impact on Workload	Value Before	Value After	Impact on SQL
elapsed_time	-26.35%	.000981	.013809	-1306.99%
parse_time			.000477	
cpu_time	-106.5%	.000494	.013743	-2681.43%
user_io_time			0	
buffer_gets	-130.82%	253	7252	-2759.03%
cost	-745969.19%	255	1982	-677.25%
reads	0%	0	0	0%
writes	0%	0	0	0%

Execution Plan Before Change:

Plan Hash Value : 612465046

Id	Operation	Name	Rows	Bytes	Cost	Time
0	SELECT STATEMENT				255	
1	SORT ORDER BY		2	92	255	00:00:01
2	TABLE ACCESS BY INDEX ROWID BATCHED	CUSTOMER	2	92	254	00:00:01
3	INDEX RANGE SCAN	CUSTOMER_I1	3000		10	00:00:01

Execution Plan After Change:

Plan Id : 545
Plan Hash Value : 4040750106

Id	Operation	Name	Rows	Bytes	Cost	Time
0	SELECT STATEMENT		2	92	1982	00:00:01
1	SORT ORDER BY		2	92	1982	00:00:01
* 2	TABLE ACCESS FULL	CUSTOMER	2	92	1981	00:00:01

Predicate Information (identified by operation id):

- 2 - filter("C_LAST"=:B1 AND "C_D_ID"=:B2 AND "C_W_ID"=:B3)

Execution Plan Before Change:

Plan Hash Value : 612465046

Id	Operation	Name	Rows	Bytes	Cost	Time
0	SELECT STATEMENT				255	
1	SORT ORDER BY		2	92	255	00:00:01
2	TABLE ACCESS BY INDEX ROWID BATCHED	CUSTOMER	2	92	254	00:00:01
3	INDEX RANGE SCAN	CUSTOMER_I1	3000		10	00:00:01

Execution Plan After Change:

Plan Id : 545
Plan Hash Value : 4040750106

Id	Operation	Name	Rows	Bytes	Cost	Time
0	SELECT STATEMENT		2	92	1982	00:00:01
1	SORT ORDER BY		2	92	1982	00:00:01
* 2	TABLE ACCESS FULL	CUSTOMER	2	92	1981	00:00:01

Predicate Information (identified by operation id):

- 2 - filter("C_LAST"=:B1 AND "C_D_ID"=:B2 AND "C_W_ID"=:B3)

You don't need to connect your app to use SQL Performance Analyzer

You can use SQL Performance Analyzer to test any change to your database

Schema changes may interfere with SQL Performance Analyzer

Further Information

SQL Performance Analyzer

- Blog post: Smooth transition to Autonomous Database using SPA

1

CAPTURE

2

ANALYZE

3

FIX

Fix regressing statements

4

REMEDY

Fixing Statements

Most cloud offerings have access to a number of tools:

- SQL Tuning Advisor
- Real-time SQL Monitoring
- SQL Access Advisor

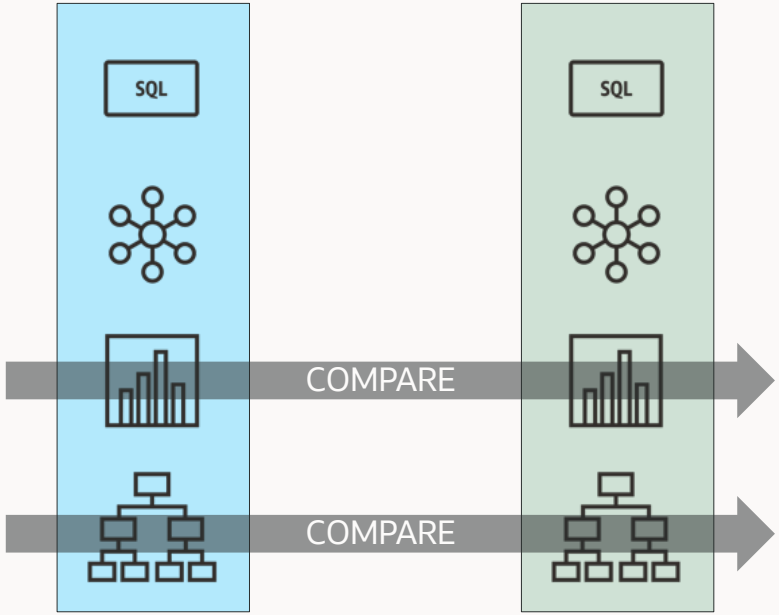
Perhaps you even have access to a DBA

You could also ask ChatGPT

What is the best fix?

- Does it have any side effects?

Continuous Improvement



Continuous Improvement

Implement change

Test execute

Check outcome

Repeat

1

CAPTURE

2

ANALYZE

3

FIX

4

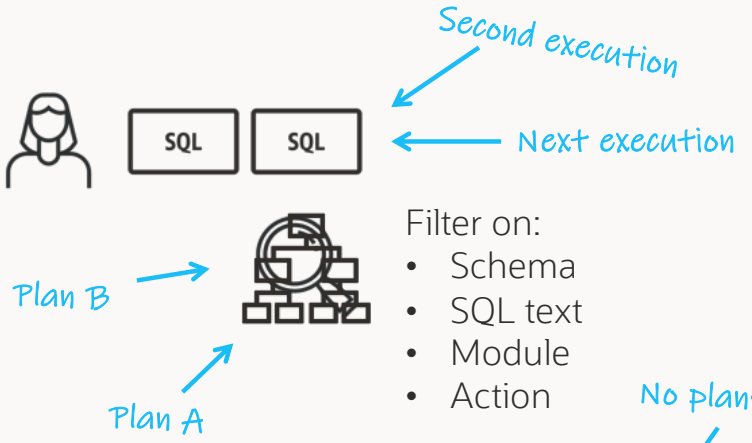
REMEDY

Stabilize performance
using
SQL Plan Management

SQL Plan Management is the best tool to ensure plan stability

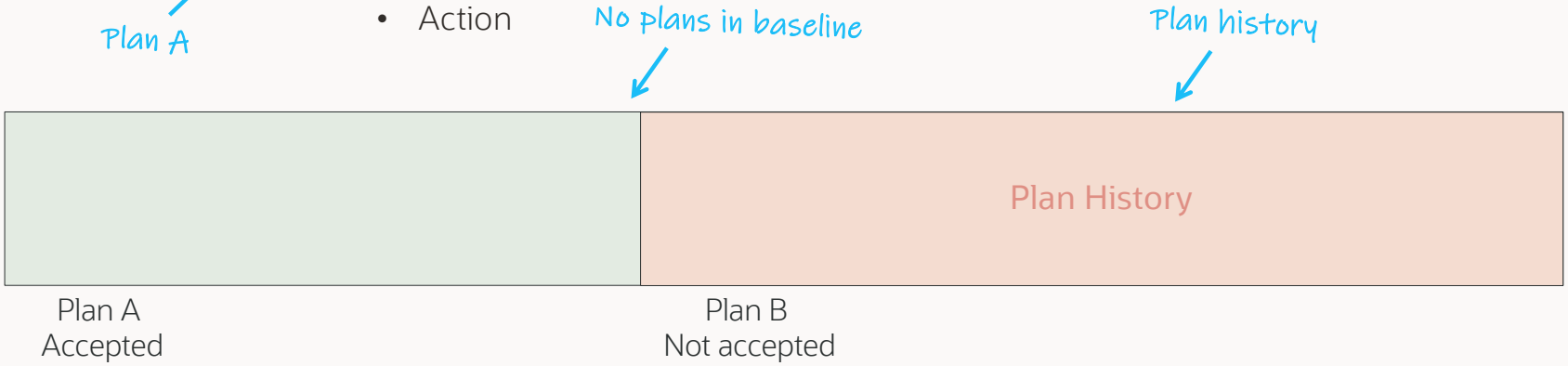
```
-- Toggles creation of SQL plan baselines for all repeatable statements
-- Usually, not recommended to capture and create baselines for all statements

alter system set optimizer_capture_sql_plan_baselines=true;
```

SQL Plan Management



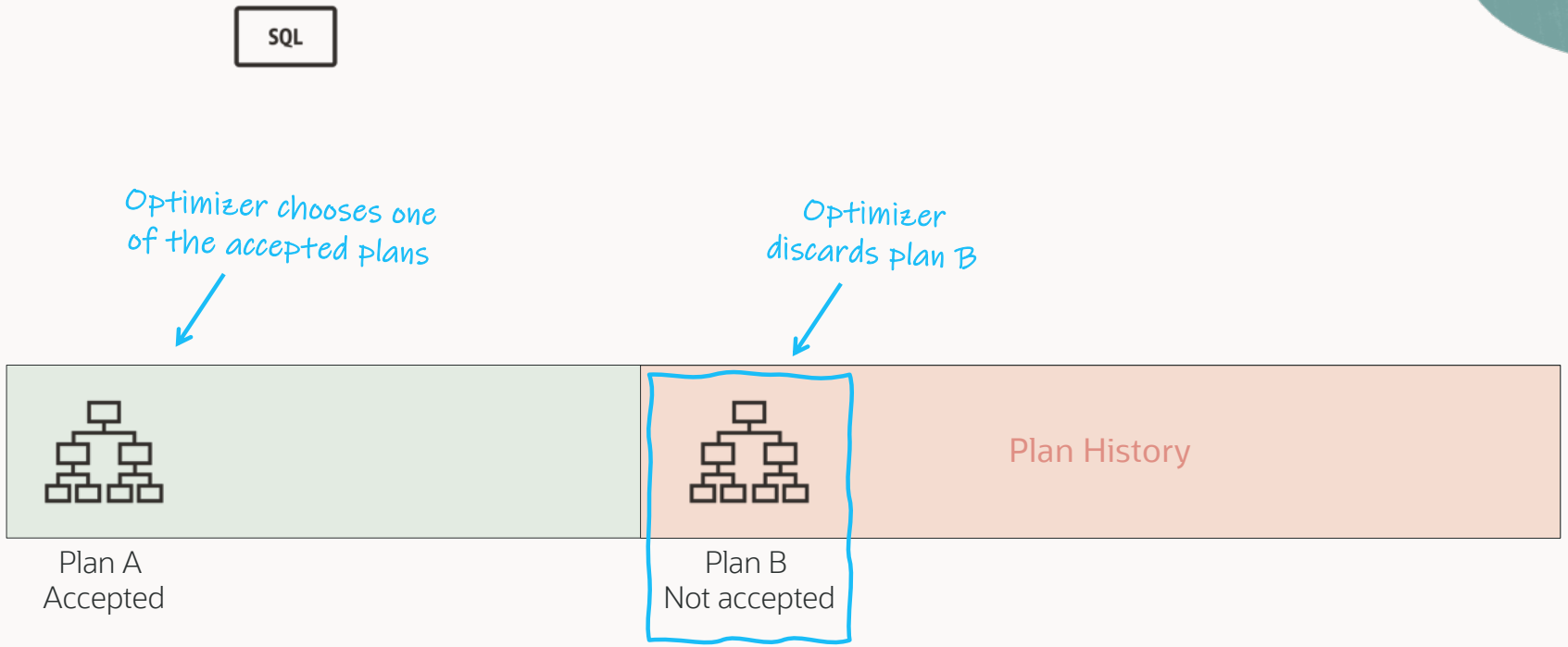
Filter on:
- Schema
- SQL text
- Module
- Action

Something changed:
- New statistics
- New parameters
- Upgrade

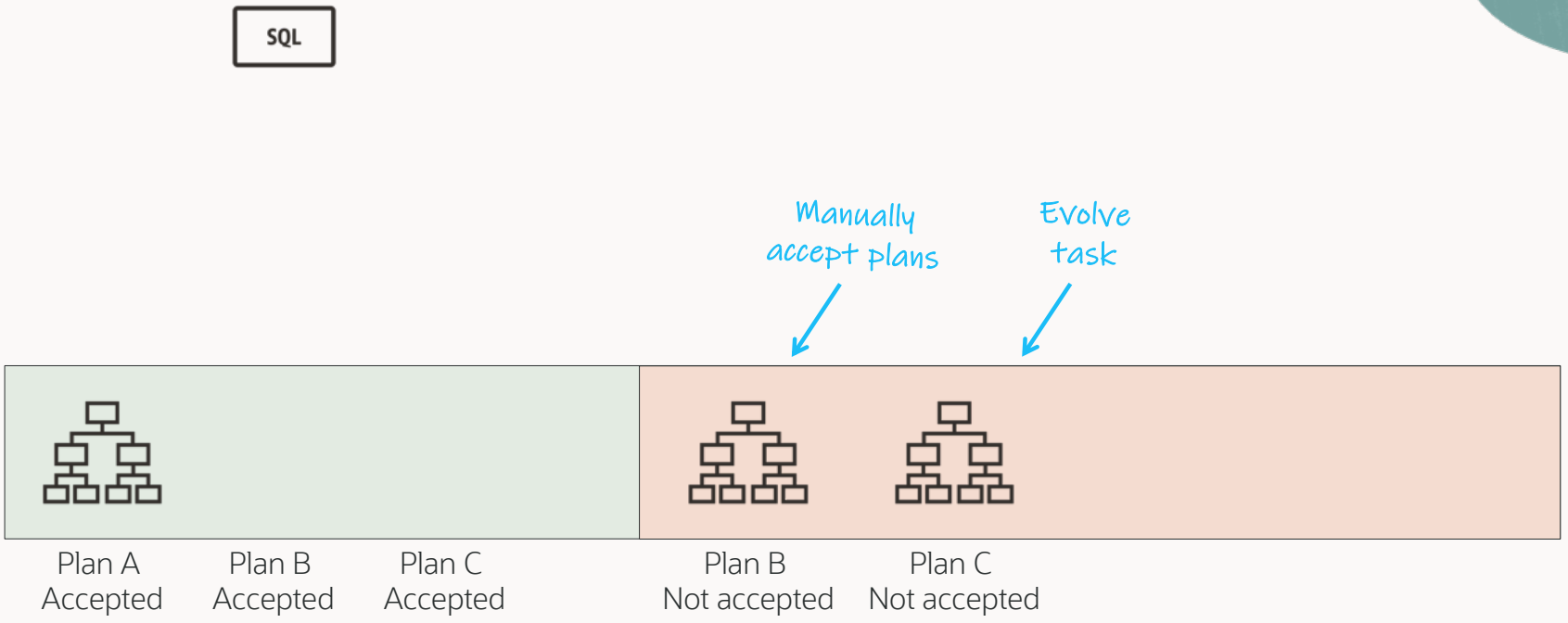


```
-- Restricts the optimizer to only use plans that are accepted
-- This is the default value

alter system set optimizer_use_sql_plan_baselines=true;
```

SQL Plan Management



SQL Plan Management



```
-- The database stores plans in the plan history for 53 weeks.
-- This might consume a lot of space. Consider lowering the limit.

exec dbms_spm.configure('plan_retention_weeks', 5);
```

You can manually create plan baselines for specific statements

- Don't capture all plans

```
-- Load all plans from a SQL tuning set into plan baselines
-- Plans are accepted automatically without test execution

var cnt number;
exec :cnt := DBMS_SPM.LOAD_PLANS_FROM_SQLSET(
                basic_filter => 'sql_id=''0cwuxyv314wcg''',
                ...
             );
```

Plan baselines are transportable; create in test, use in production

```
-- Pack baselines from "problematic" statements into staging table
exec DBMS_SPM.PACK_STGTAB_BASELINE(sql_handle => SQL_130f372d9ffe4df9, ...);
```

```
-- Pack baselines from "problematic" statements into staging table
exec DBMS_SPM.PACK_STGTAB_BASELINE(sql_handle => SQL_130f372d9ffe4df9, ...);
exec DBMS_SPM.PACK_STGTAB_BASELINE(sql_handle => SQL_51dc7232adc62849, ...);
exec DBMS_SPM.PACK_STGTAB_BASELINE(sql_handle => SQL_3f38bc33ae7086c9, ...);
```

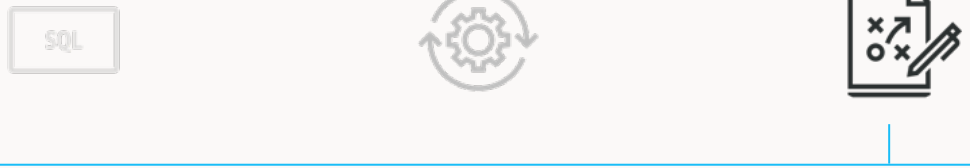

```
-- Pack baselines from "problematic" statements into staging table
exec DBMS_SPM.PACK_STGTAB_BASELINE(sql_handle => SQL_130f372d9ffe4df9, ...);
exec DBMS_SPM.PACK_STGTAB_BASELINE(sql_handle => SQL_51dc7232adc62849, ...);
exec DBMS_SPM.PACK_STGTAB_BASELINE(sql_handle => SQL_3f38bc33ae7086c9, ...);

...

-- After production migration, import plan baselines to fix regressions
exec DBMS_SPM.UNPACK_STGTAB_BASELINE( ... );
```

Let's automate it

Automatic SQL Plan Management

- Background job (30 min)
- Finds resource-intensive SQLs
- Evaluate execution statistics against previous executions

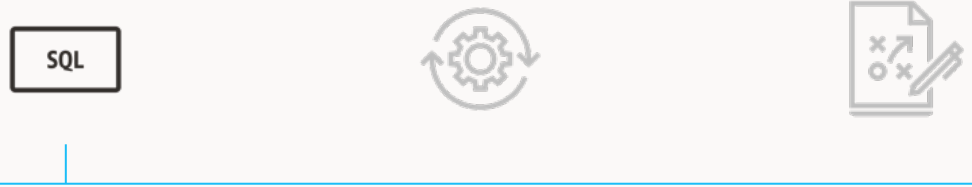
Automatic SQL Plan Management



- Creates a plan baseline with the best plan
- Next execution uses an accepted plan

- Oracle Database 19c Enterprise Edition, RU 19.22
- Exadata and Autonomous AI Database

Real-Time SQL Plan Management

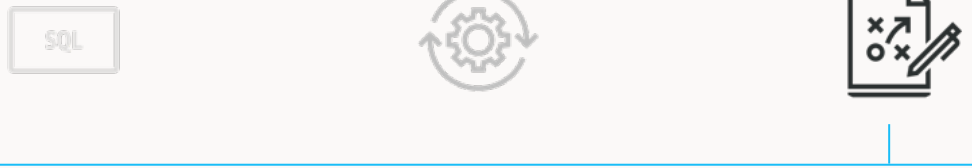


- User executes SQL
- Optimizer chooses a different execution plan
- Executes with new plan

Real-Time SQL Plan Management

- Compares execution statistics with history
- Comparison is performed in the user foreground session

Real-Time SQL Plan Management



- Creates a plan baseline with the best plan
- Next execution uses an accepted plan

- Oracle AI Database 26ai Enterprise Edition
- Autonomous AI Database 19c

Here's one for the Top Gun DBA

```
--Loads all known plans from cursor cache, AWR,
--and automatic SQL tuning sets into a SQL plan baseline.
--Use Evolve Advisor to find the best plan and mark that as accepted.

var report clob;
exec :report := dbms_spm.add_verified_sql_plan_baseline('<sql_id>');
select :report report from dual;
```

Who said literals?

- SQL Plan Management is not a good fit for an application that doesn't use bind variables

```
--Application using literals creates many distinct statements
--You'd get 4 plan baselines
select * from sales where order_id=42;
select * from sales where order_id=56;
select * from sales where order_id=101;
select * from sales where order_id=220;
```


```
--Application using literals creates many distinct statements
--You'd get 4 plan baselines
select * from sales where order_id=42;
select * from sales where order_id=56;
select * from sales where order_id=101;
select * from sales where order_id=220;


--Ideally change the application to use literals
--You'd get only 1 plan baseline
select * from sales where order_id=:b1;
```


Generally, avoid setting CURSOR_SHARING=FORCE

- Advice from Real-World Performance Group

Use SQL Profiles

- Part of Tuning Pack included in most cloud offerings

Further Information

SQL Plan Management

- Blog post: SQL Plan Management Cheat Sheet – Part 1
- Blog post: SQL Plan Management Cheat Sheet – Part 2
- Blog post: What is automatic SQL plan management and why should you care?
- Blog post: What is Real-time SQL plan Management?
- My Oracle Support: Things to Consider to Avoid SQL Plan Management (SPM) Related Problems on 19c (KB139467)

1

CAPTURE

2

ANALYZE

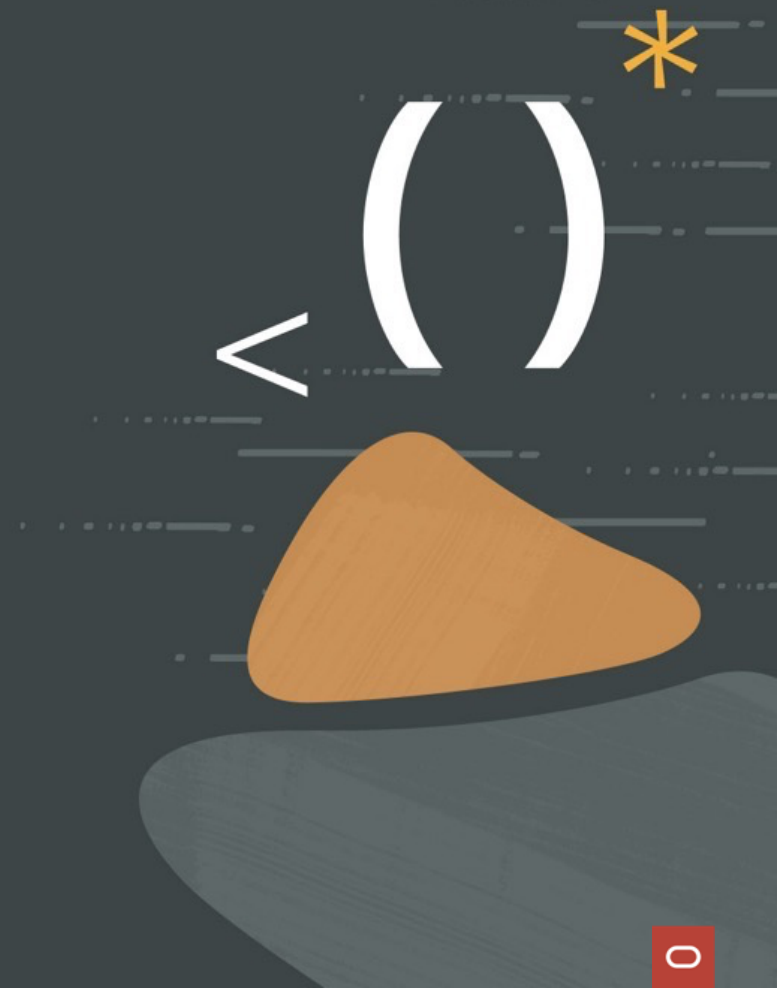
3

FIX

4

REMEDY

A Word About Statistics



PHYSICAL

LOGICAL

PHYSICAL

LOGICAL

Statistics

Object Statistics

Statistics Preferences

Column Usage Information

Object Statistics

Statistics  Statistics Preferences

Column Usage Information

```
BEGIN
   DBMS_STATS.SET_TABLE_PREFS (
      OWNNAME => 'APPUSER',
      TABNAME => 'SALES',
      PNAME   => 'TABLE_CACHED_BLOCKS',
      PVALUE  => '42'
   );
END;
```


Table 171-131 SET_TABLE_PREFS Procedure Parameters

Parameter	Description
ownname	Owner name
tabname	Table name
pname	Preference name. You can set the default value for following preferences: • APPROXIMATE_NDV_ALGORITHM • AUTO_STAT_EXTENSIONS • CASCADE • DEGREE • ESTIMATE_PERCENT • GRANULARITY • INCREMENTAL • INCREMENTAL_LEVEL • INCREMENTAL_STALENESS • METHOD_OPT • NO_INVALIDATE • OPTIONS • PREFERENCE_OVERRIDES_PARAMETER • PUBLISH • STALE_PERCENT • TABLE_CACHED_BLOCKS
pvalue	Preference value. If NULL is specified, it will set the Oracle default value.

PL/SQL Packages and Types Reference, Oracle Database 19c

```
EXEC DBMS_STATS.EXPORT_TABLE_PREFS('APPUSER', 'SALES', 'STAGING_TAB');

...

EXEC DBMS_STATS.IMPORT_TABLE_PREFS('APPUSER', 'SALES', 'STAGING_TAB');
```


It is not possible to export or transport global statistics preference

- Define manually using `DBMS_STATS.SET_GLOBAL_PREFS`

You often use statistics preferences to solve a particular problem

- Evaluate whether that problem exists in the target environment

Statistics

Object Statistics

Statistics Preferences

Column Usage Information

Column Usage Information

- Information on how you join tables
- Stored internally in `SYS.COL_USAGE$`
- Used by the optimizer to determine when to create histograms
  `METHOD_OPT => ... SIZE AUTO`
- When missing, statistics gathering creates no or few histograms

1 Include statistics in Data Pump

2 Exclude statistics in Data Pump
Regather statistics after import

3 Exclude statistics in Data Pump
Import statistics using `DBMS_STATS`

Importing statistics using Data Pump is a convenient, but slow option

- Includes object statistics, table-level preferences and column usage information
- Not recommended

1 Include statistics in Data Pump

2 Exclude statistics in Data Pump
Regather statistics after import

3 Exclude statistics in Data Pump
Import statistics using DBMS_STATS

Be sure to gather statistics fast

- Use **DEGREE** and **CONCURRENT** options
- Blog post by Nigel Bayliss

You're missing preferences and column usage information

EXCLUDE

`EXCLUDE=STATISTICS`

`COL_USAGE$` empty

REGATHER

First time only

```
METHOD_OPT =>
SIZE SKEWONLY
```

GO LIVE

Column usage information is updated

REGATHER

Use default

```
METHOD_OPT =>
SIZE AUTO
```

Further Information

Transporting statistics

- Blog post: How to Export and Import Statistics Faster Using DBMS_STATS in Parallel
- Blog post: If Importing Statistics Using DBMS_STATS Is Slow
- Blog post: Does Exporting Database Statistics Include the Dictionary Statistics?
- Video: Transporting optimizer statistics - pro tips
- Video: Transporting optimizer statistics - demo
- Video: Transporting optimizer statistics - the concept

1 Include statistics in Data Pump

2 Exclude statistics in Data Pump
Regather statistics after import

3 Exclude statistics in Data Pump
Import statistics using `DBMS_STATS`

```
EXEC DBMS_STATS.EXPORT_SCHEMA_STATS('APPUSER', 'STAGING_TAB');

...

EXEC DBMS_STATS.IMPORT_SCHEMA_STATS('APPUSER', 'STAGING_TAB');
```


The database considers imported statistics as *current*

- Be sure those statistics are accurate

Importing or transporting statistics might be a bad idea…

When source and target database do not match

Statistics | When Importing Stats Is Bad

Fragmented table

Blocks	12000
Leaf blocks	11000
B-level	4
Clustering factor	10000

Compacted table

Blocks	12000
Leaf blocks	11000
B-level	4
Clustering factor	10000

```
DBMS_STATS.GATHER_TABLE_STATS(...
```

Blocks	5000
Leaf blocks	4000
B-level	2
Clustering factor	20000

Statistics | When Importing Stats Is Bad

- Potentially a problem
  - Fragmented tables
  - Changing block size
  - Changing character set
  - Compress or decompress
  - …
- Only a problem for table and index base statistics, column statistics remain accurate

Comparing STATISTICS options

	Import with Data Pump	Regather	Import with DBMS_STATS
Time	Significant	Significant	Short
Column usage information	Included	Missing	Missing
Accuracy	Potentially inaccurate	Accurate	Potentially inaccurate
Statistics preferences	Included	Missing	Optional

Daniel's Favorite

1. Perform test migration
   - Regather in test database
   - Test and adjust
   - Export statistics using `DBMS_STATS`

2. Perform production migration
   - Import from test migration

Autonomous AI Database gathers statistics differently

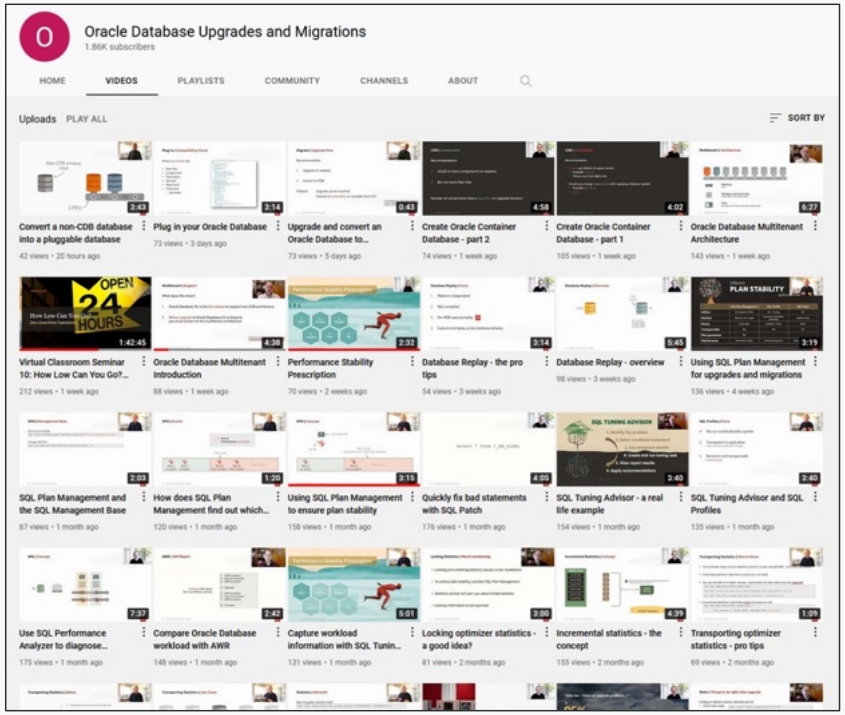
It's better to fail in our lab, than in production

Oracle LiveLabs:
Hitchhiker's Guide for Upgrading to Oracle AI Database

Key Learnings

1. Gather workload information
2. Analyze with SQL Performance Analyzer
3. Use SQL Plan Management

YouTube | Oracle Database Upgrades and Migrations



https://www.youtube.com/@upgradenow

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

SCAN ME

aiOug

MULTICLOUD AI WORLD

HYDERABAD & BENGALURU

Conference Session:

Performance and Migrations: A Practical Guide

Your Voice Matters
Let Me Know
How I Did

Daniel Overby Hansen

Distinguished Product Manager

Oracle Denmark


ORACLE