ORACLE

Oracle Database In-Memory Advisor

Best Practices

Purpose statement

This document provides best practices for the use of the Oracle Database In-Memory Advisor. It is intended solely to give you some general guidelines so that you can determine how best to use Database In-Memory in your environment.

Disclaimer

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

ORACLE

Table of contents

ORACLE

Intended Audience

Readers are assumed to have hands-on experience with Oracle Database technologies from the perspective of a DBA or performance specialist.

Introduction

Oracle Database 12.1.0.2 introduced Oracle Database In-Memory allowing a single database to efficiently support mixed analytic and transactional workloads. An Oracle Database configured with Database In-Memory delivers optimal performance for transactions while simultaneously supporting real-time analytics and reporting. This is possible due to a unique "dual-format" architecture that enables data to be maintained in both the existing Oracle row format, for OLTP operations, and a new purely in-memory column format, optimized for analytical processing. Database In-Memory also enables both data marts and data warehouses to provide more ad-hoc analytics, giving end-users the ability to ask multiple business driving queries in the same time it takes to run just one now.

For complete details about Database In-Memory, see the Oracle Database In-Memory technical brief and the Oracle Database In-Memory Page on oracle.com.

The Oracle Database In-Memory Advisor analyzes your workload and makes specific recommendations regarding how to size Database In-Memory and which objects would render the greatest benefit to your system when placed In-Memory. See the In-Memory Advisor whitepaper for details on the In-Memory Advisor.

This paper discusses best practices for using the In-Memory Advisor.

ORACLE

In-Memory Advisor Description

The goal of Database In-Memory is to optimize analytical processing in the database. The In-Memory Advisor analyzes the analytical processing workload present in your database to determine an estimated benefit for the database as a whole if that analytical workload is optimized.

The In-Memory Advisor differentiates analytics processing from other database activity based upon SQL plan cardinality, Active Session History (ASH), use of parallel query, and other statistics.

The In-Memory Advisor estimates the In-Memory size of objects based upon statistics and heuristic compression factors and, optionally, the DBMS_COMPRESSION package (in Oracle Database 12.1.0.2 and above).

The In-Memory Advisor estimates analytic processing performance improvement factors based upon the following:

- Elimination of user I/O waits, cluster transfer waits, buffer cache latch waits, etc.
- Certain query processing advantages related to specific compression types.
- Decompression cost heuristics per specific compression types.
- SQL plan selectivity, number of columns in the result set, etc.

The In-Memory Advisor produces a recommendation report optimized for the In-Memory size you specify. If you omit the In-Memory size, it defaults to the largest In-Memory size recommended by the In-Memory Advisor based on its analysis. Independent of the In-Memory size you specify, the top of the report lists a number of In-Memory sizes with estimated performance benefits. With this information, you may choose a different In-Memory size. The rest of the report shows recommendations optimized for the In-Memory size you choose.

In-Memory Advisor Best Practices

Below are some best practices to help you use the In-Memory Advisor most effectively in your environment.

Use SQL Performance Analyzer to Validate Recommendations

SQL Performance Analyzer (SPA) allows you to validate the change in performance of individual SQL statements in your database. SPA can be used to measure the impact of any change to your database that could affect performance. Therefore, SPA is ideally suited for evaluating the impact of placing specific database objects in the In-Memory column store.

The input to SPA is a SQL Tuning Set (STS). A SQL Tuning Set is a database object that includes one or more SQL statements, along with their execution statistics and execution context. SQL statements can be loaded into a SQL Tuning Set from different sources, including the cursor cache, Automatic Workload Repository (AWR), and existing SQL Tuning Sets.

SPA creates two trials by executing every SQL in the SQL Tuning Set twice. SPA first executes every SQL in the pre-change environment. After the change has been made that will impact performance, SPA executes the second trial.

In this case, trial 1 will be executed using the traditional database access structures. Trial 2 will be executed with database objects in the In-Memory column store. After the two trials have been run, SPA produces a report highlighting the performance impact of the change – both for the workload as a whole and also the impact on each individual SQL in the SQL Tuning Set.

You can execute the various tasks for SPA from Enterprise Manager 12c or from PL/SQL. Both methods are documented in the Database Testing Guide available in the Oracle database documentation library.

Create a SQL Tuning Set

The first step is to create the SQL Tuning Set (STS) that contains the set of SQL to be tested for performance improvements. An STS can be created from polling the cursor cache, from existing AWR snapshots, or as a subset of an existing STS.

ORACLE

Database In-Memory improves the performance of analytic queries – queries that generally take a significant amount of time to execute. To measure the benefit of Database In-Memory, create an STS that consists of long running queries.

To create an STS from the cursor cache using PL/SQL, you use the DBMS_SQLTUNE.CAPTURE_CURSOR_CACHE_SQLSET procedure. You can specify a SQL Tuning Set name, the duration of time in seconds for the capture to run and the polling interval. For example:

```
EXEC DBMS_SQLTUNE.CAPTURE_CURSOR_CACHE_SQLSET( -
sqlset_name => 'IM_STS', -
time_limit => 300, -
repeat_interval => 10);
```

This would create an STS named IM_STS by polling the cursor cache for SQL every 10 seconds over the next 300 seconds.

You can also create an STS from SQL captured in Automatic Workload Repository (AWR) snapshots. AWR automatically captures high load SQL at regular intervals in your database. For testing Database In-Memory, select long running queries based on elapsed time. For example, the following code snippet creates an STS named IM_STS. It then loads that STS with SQL from AWR snapshots 1000 through 1010 that have elapsed time greater than 60 seconds.

```
EXEC DBMS_SQLTUNE.CREATE_SQLSET (sqlset_name => 'IM_STS'); DECLARE
  cur DBMS_SQLTUNE.SQLSET_CURSOR;
BEGIN
  OPEN cur FOR
    SELECT VALUE(P)
      FROM table(
        DBMS_SQLTUNE.SELECT_WORKLOAD_REPOSITORY(1000,1010,
                        'elapsed_time > 60000000') ) P;
  DBMS_SQLTUNE.LOAD_SQLSET(sqlset_name => 'IM_STS',
                        populate_cursor => cur);
END;
/
```

Configure Database In-Memory and Load Database Objects

To test Database In-Memory, SPA will execute the SQL in the STS using the standard database structures and then using Database In-Memory. The first step is to implement the In-Memory column store and populate it with the objects recommended by the In-Memory Advisor.

There are two steps to populating the In-Memory Column Store with database objects, such as tables and partitions.

The first step is to designate objects that should be placed In-Memory with the INMEMORY clause. In addition, you can specify a priority for the objects to be populated In-Memory with the PRIORITY clause. A priority of CRITICAL means that the object should be loaded In-Memory immediately after the database is restarted and before objects with a lower priority. A priority of NONE, which is the default, means the object will be loaded upon access, and after any other objects of higher priority have been loaded. For example, the following DML would specify the LINEITEM table should be populated immediately after the database has been restarted:

```
alter table lineitem inmemory priority critical;
```

The In-Memory Advisor produces a script which modifies the recommended tables and partitions so they will be placed in the In-Memory column store. Execute that script to appropriately modify your key database objects so they will be loaded into In-Memory.

ORACLE

The second step to configure the In-Memory Column Store is to set the INMEMORY_SIZE initialization parameter to the size suggested by the In-Memory Advisor. To set the INMEMORY_SIZE parameter the database must be shutdown and restarted.

For example:

```
alter system set inmemory_size=10G scope=spfile;
```

Then shutdown and restart the database.

Allow sufficient time after the database has restarted for the In-Memory column store to be populated. If some objects have priority NONE, you may have to run a full table scan of the object for it to be populated into the In-Memory column store.

For example:

```
select /*+ full(orders) */ count(*) from orders;
```

You can monitor the population of the In-Memory column store from the Enterprise Manager In-Memory Central page or by querying the V$IM_SEGMENTS view in the database.

Run the SPA trials

Once the column store is populated, run the SPA test. To test Database In-Memory, SPA executes the SQL in the STS, in one trial without using the In-Memory column store, and then a second time using the In-Memory column store. After the completion of the trials, SPA produces a report comparing the performance difference between the two.

On the SPA home page in Enterprise Manager, there are a number of predefined workflows, along with a 'Guided Workflow'.

For testing Database In-Memory, the easiest workflow to use is the 'Parameter Change' workflow. When you choose the 'Parameter Change' workflow, you will be prompted for a few inputs:

- A name for the SPA Task

- The STS you've created containing the SQL to be tested with Database In-Memory

- The parameter you wish to test – INMEMORY_QUERY

- The two values of the parameter – DISABLE and ENABLE.

- The metric to be used for comparison between the two trials. Database In-Memory improves analytic query performance. A good metric to compare analytic query performance is 'Elapsed Time', since analytic queries typically take a substantial amount of time. Choose 'Elapsed Time' as the metric.

Your input screen should look similar to Figure 1, below:

ORACLE

Figure 1. SPA Task Input

Click on the 'Submit' button to initiate the SPA task.

The SPA task should take some time to complete. You can monitor the progress of the SPA task from the SPA home page.

Evaluate SPA Results

When the two trials are completed, click on the 'View Latest Report' from the SPA home page. The report is a graphical report giving the results of the two trials.

In the top left of the report, is a summary result of the two trials, based on the chosen metric, along with the computed benefit of the change. To the right of the summary, is a bar chart showing the number of SQLs with improved, regressed and unchanged performance. The bar chart also indicates whether the SQLs have a new or unchanged execution plan.

Figure 2 below shows an example of the summary section:

ORACLE
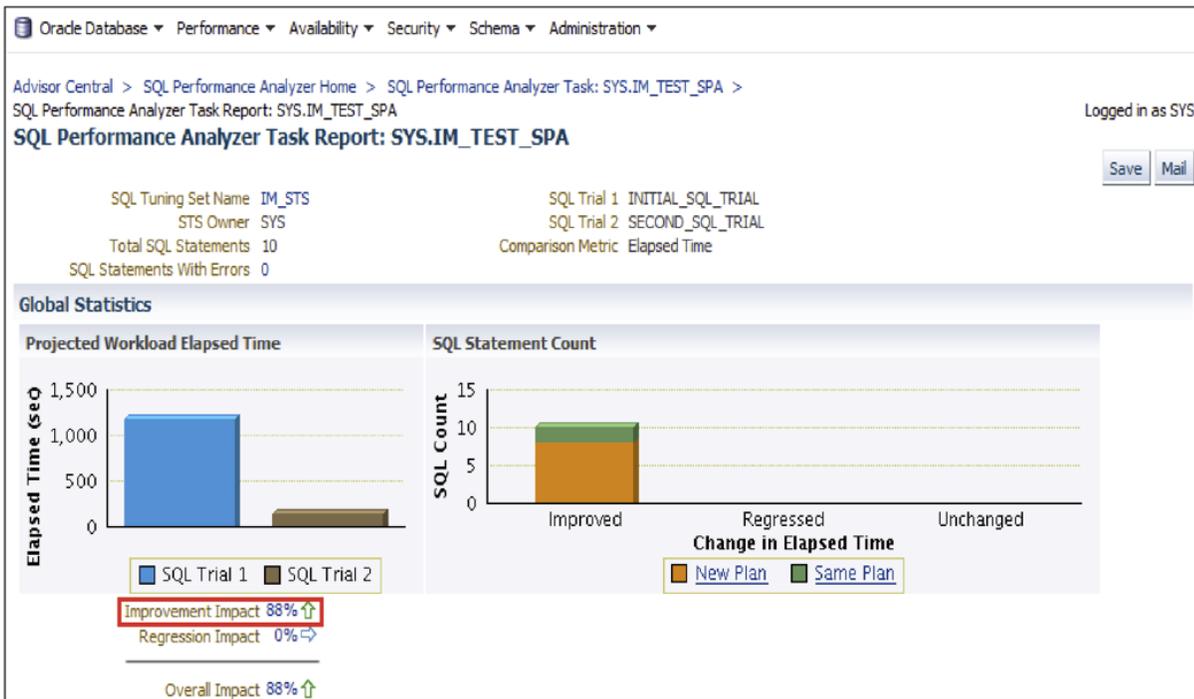
Figure 2. Summary of SPA results

Below the summary is a list of the top SQL that have the greatest impact on the workload. For each SQL, two measurements are listed. The first measurement is the change in performance of the individual SQL and the second is the net impact on the entire workload of the change in performance of this SQL. Figure 3 below shows an example of this section of the report:

	SQL ID	Net Impact on Workload (%)	Elapsed Time (sec)		Net Impact on SQL (%)	New Plan
			SQL Trial 1	SQL Trial 2		
⇑	082r0f7j3sngb	14.310	13.859	1.024	92.610	Y
⇑	0cscg7xwjtu03	11.340	14.791	1.569	89.390	Y
⇑	9chrkvvc8vyy6	9.540	10.539	0.428	95.940	Y
⇑	d6zh34wf4d9cb	9.510	15.451	1.585	89.740	Y
⇑	gr4kz0d4h0trw	9.290	12.810	2.965	76.860	Y

Figure 3. High impact SQL summary

You can drill down into more details of each SQL by clicking its SQL ID link. That drill down will give you the execution plans and all metric details for that SQL.

To save the report to a shareable html format, click the 'Save' button at the top of the page.

Conclusion

The In-Memory Advisor makes specific, actionable recommendations about how to configure Database In-Memory and which database objects to place in the IM column store. Use the SQL Performance Analyzer for quick and easy validation that your workload will effectively use and benefit from Database In-Memory.

Business / Technical Brief / Oracle Database In-Memory Advisor / Version 2.1

Copyright © 2022, Oracle and/or its affiliates / Public

ORACLE

Connect with us

Call +**1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at: **oracle.com/contact**.

[B] blogs.oracle.com   [f] facebook.com/oracle   [t] twitter.com/oracle