ORACLE

Oracle Database In-Memory

Ask TOM Office Hours – SQL Monitor Updates

Andy Rivenes

Database In-Memory Product Management

Oracle Corporation

Email: andy.rivenes@oracle.com

Twitter: @TheInMemoryGuy

Blog: blogs.oracle.com/in-memory

SQL Monitor Update - Agenda

- See previous session SQL Monitor with Database In-Memory from December 18, 2019
  - This session will be a little bit of review and a little bit of new material
- There are some new features in the Active report
- We will dive a little deeper into the details
- Hopefully will convince you to only use the Active report and to not bother with the TEXT or HTML options

Explain Plan

- Please don't expect an "`EXPLAIN PLAN`" to be of much use
- From the SQL Tuning Guide:

  "EXPLAIN PLAN output shows how the database would have run the SQL statement when the statement was explained. Because of differences in the execution environment and explain plan environment, the explained plan can differ from the actual plan used during statement execution."
- A simple EXPLAIN PLAN:

```
---------------------------------------------------------------------------------
| Id  | Operation           | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
---------------------------------------------------------------------------------
|   0 | SELECT STATEMENT    |           |     1 |    43 | 26813   (1)| 00:00:02 |
|   1 |  SORT AGGREGATE     |           |     1 |    43 |            |          |
|*  2 |   HASH JOIN         |           |   418 | 17974 | 26813   (1)| 00:00:02 |
|*  3 |    TABLE ACCESS FULL| DATE_DIM  |     1 |    25 |  2222   (1)| 00:00:01 |
|*  4 |    TABLE ACCESS FULL| LINEORDER |  1004K|   17M| 24588   (1)| 00:00:01 |
---------------------------------------------------------------------------------
```

- Now imagine trying to parse something like that over hundreds of lines
- Where do you start?

Displaying The Actual Execution Plan

```
alter session set statistics_level = all;
Select sum(lo_extendedprice * lo_discount) revenue
From   LINEORDER l, DATE_DIM d
Where  l.lo_orderdate = d.d_datekey
And    l.lo_discount between 2 and 3
And    l.lo_quantity < 24
And    d.d_date='December 24, 1996';
select * from table(dbms_xplan.display_cursor(NULL, NULL, 'TYPICAL IOSTATS LAST'));
```

```
------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation           | Name      | Starts | E-Rows |E-Bytes| Cost (%CPU)| E-Time   | A-Rows |   A-Time   | Buffers | Reads  |
------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT    |           |      1 |        |       | 26813 (100)|          |      1 |00:00:00.96 |   89745 |  89701 |
|   1 |  SORT AGGREGATE     |           |      1 |      1 |    43 |            |          |      1 |00:00:00.96 |   89745 |  89701 |
|*  2 |   HASH JOIN         |           |      1 |    418 | 17974 | 26813   (1)| 00:00:02 |    440 |00:00:00.95 |   89745 |  89701 |
|*  3 |    TABLE ACCESS FULL| DATE_DIM  |      1 |      1 |    25 |  2222   (1)| 00:00:01 |      1 |00:00:00.01 |      83 |     47 |
|*  4 |    TABLE ACCESS FULL| LINEORDER |      1 |   1004K|    17M| 24588   (1)| 00:00:01 |   1004K|00:00:00.91 |   89662 |  89654 |
------------------------------------------------------------------------------------------------------------------------------------
```

- An improvement
- At least we can see where time was spent and the actual rows returned, but we can do a lot better

SQL Monitor Report Types

- There are three types of SQL Monitor reports
  - Text – a text report, this is the default
    - `dbms_sql_monitor.report_sql_monitor(report_level=>'ALL',type=>'TEXT')`
  - HTML – simple HTML report
    - `dbms_sql_monitor.report_sql_monitor(report_level=>'ALL',type=>'HTML')`
  - Active – database active report
    - `dbms_sql_monitor.report_sql_monitor(report_level=>'ALL',type=>'ACTIVE')`

SQL Monitor Active Reports

- For Database In-Memory please, only use the “**active**” report
- This is what we will ask for and if you give us something else then we will probably ask again ☹
- All other SQL Monitor reports lack any data in the “Activity” Bar. This is a serious limitation when trying to find out how time was spent

- SQL Monitor has a text based version if you need text
  - You will lose the Activity section
  - The advantage over dbms_xplan is that it will display Parallel Query and RAC information from remote execution

- If you really need a character based version then dbms_xplan can still be useful:
  - alter session set statistics_level = all;
  - Run your SQL
  - select * from table(dbms_xplan.display_cursor(NULL,NULL,'TYPICAL IOSTATS LAST’));

Caveats

- Required to use SQL Monitor:
  - A Diagnostics and Tuning Pack License
  - STATISTICS_LEVEL parameter to be set to 'TYPICAL' or 'ALL'
  - CONTROL_MANAGEMENT_PACK_ACCESS parameter set to 'DIAGNOSTIC+TUNING'

- Database Reference - Oracle Database monitors simple database operations, which are top SQL statements and PL/SQL subprograms, when any of the following conditions is true:
  - The operations run in parallel
    - How to Collect SQL Monitor Output For Parallel Query (Doc ID 1604469.1)
  - The operations have consumed at least 5 seconds of CPU or I/O time in a single execution
  - Tracking for the operations is forced by the /*+ MONITOR */ hint

SQL Monitor Limits

- SQL query line limit

  How to Monitor SQL Statements with Large Plans Using Real-Time SQL Monitoring? (Doc ID 1613163.1)

  The threshold for plan monitoring is controlled by a hidden parameter _SQLMON_MAX_PLANLINES (which has a default of 300). This can be changed to a higher value, for example 500 using a command like:

  ```
  ALTER SYSTEM SET "_sqlmon_max_planlines"=500 SCOPE=BOTH;
  or
  ALTER SESSION SET "_sqlmon_max_planlines"=500 SCOPE=BOTH;
  ```

  This would mean that explain plans with up 500 lines in them would now be monitored

- SQL Monitor run time threshold limit
  - You can change the default run time threshold of 5 seconds to a different value with the parameter _SQLMON_THRESHOLD
  - However, you should probably use the MONITOR hint instead

Report format changes – Flash to Jet

- Oracle Real-time SQL Monitoring Active Reports in Enterprise Manager and the Adobe Flash End-of-Life (Doc ID 2734333.1)
  - Before Oracle database 19c, the ‘active’ reports were Adobe Flash based
  - Rendering the Flash ‘active’ report required some SWF files to be downloaded from the Oracle Technology Network (OTN).

- Starting in database 19c, the SQL Monitor reports are rendered using Oracle JET UI technology.
  - After the Flash EOL, the older saved Flash ‘active’ reports would no longer be able to render. Oracle is updating the necessary files on OTN so that any older Flash based reports will render automatically in the new Oracle JET UI.
  - Oracle Technology Network (OTN) will be updated with the newer JET UI files on December 9, 2020.
    After that, all older saved Flash ‘active’ reports will be viewable as JET reports.

New Features

- **19c New Feature: Real-time SQL Monitoring for Developers (Doc ID 2480461.1)**
  - Starting in Oracle Database 19c, Oracle allow database user to view their own Real Time SQL Monitoring reports without requiring DBA privileges or SELECT_CATALOG_ROLE

- New “More Columns” pull down lets you select/unselect columns to display in the report
  - **Cost** is now available

- The Rows column will highlight in red cardinalities that have large variations from the estimate

- Optimize Environment section to show optimizer parameter settings

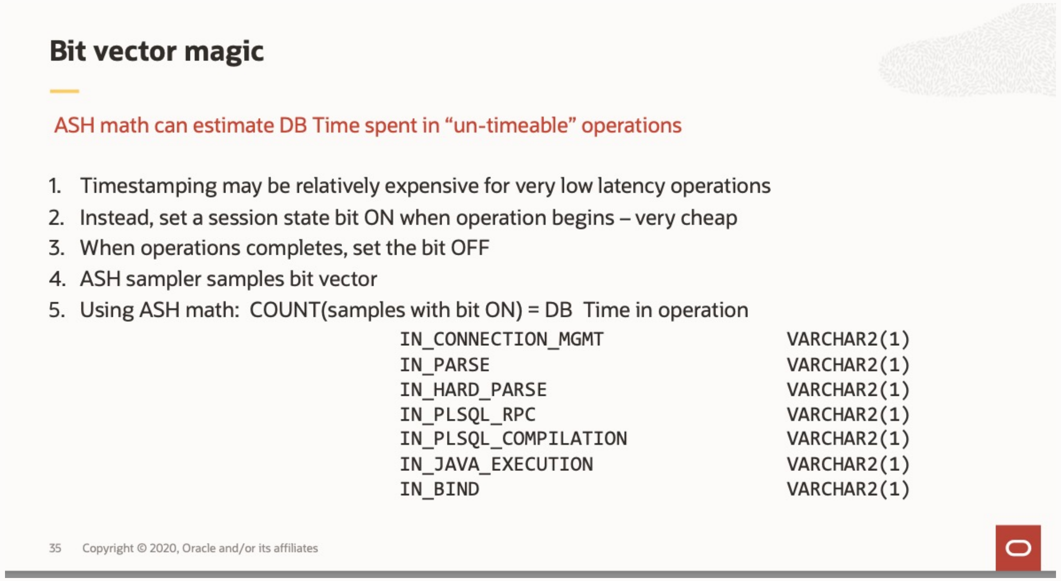
- Outline section to show outline hints

- Full screen mode for the Plan Statistics section

Activity Column

- The activity column in the Plan Statistics section is only populated with data in the Active report
  - It doesn’t look like this will ever change
- The Activity column provides you with a breakdown of where time was spent for each execution line
  - If it is blank then no time was recorded for that line
  - This data comes from ASH data
  - This time diffentiation is available in the Response Time column and in the Time & Wait and I/O sections of the report as well
- Where does this time come from?
  - v$active_session_history
  - For Database In-Memory there are multiple “IN_MEMORY%” columns
  - Since it is ASH data it is sampled

Bit vector magic



Bit vector magic

ASH math can estimate DB Time spent in “un-timeable” operations

1. Timestamping may be relatively expensive for very low latency operations
2. Instead, set a session state bit ON when operation begins – very cheap
3. When operations completes, set the bit OFF
4. ASH sampler samples bit vector
5. Using ASH math: COUNT(samples with bit ON) = DB Time in operation

```
IN_CONNECTION_MGMT                VARCHAR2(1)
IN_PARSE                          VARCHAR2(1)
IN_HARD_PARSE                     VARCHAR2(1)
IN_PLSQL_RPC                      VARCHAR2(1)
IN_PLSQL_COMPILATION              VARCHAR2(1)
IN_JAVA_EXECUTION                 VARCHAR2(1)
IN_BIND                           VARCHAR2(1)
```

35

- Do a Google search on “DB Time, Average Active Sessions, and ASH Math” by John Beresniewicz
- I saw him present at the 2020 RMOUG Training Days conference
- It’s a very good presentation on how and why Oracle added this information to ASH

Risks With Very Short Run Times

- Relies on ASH data which is sampled data
- "… statistics in V$SQL_MONITOR are generally refreshed in near real time, **once every second**." – MOS Note: How to Collect SQL Monitor Output For Parallel Query (Doc ID 1604469.1)
- The accuracy is not as good as extended SQL trace data, but in most cases that is OK
- Plus, the CPU breakdown between time spent on in-memory processing versus other CPU time is not present in SQL trace data

Time & Wait Section

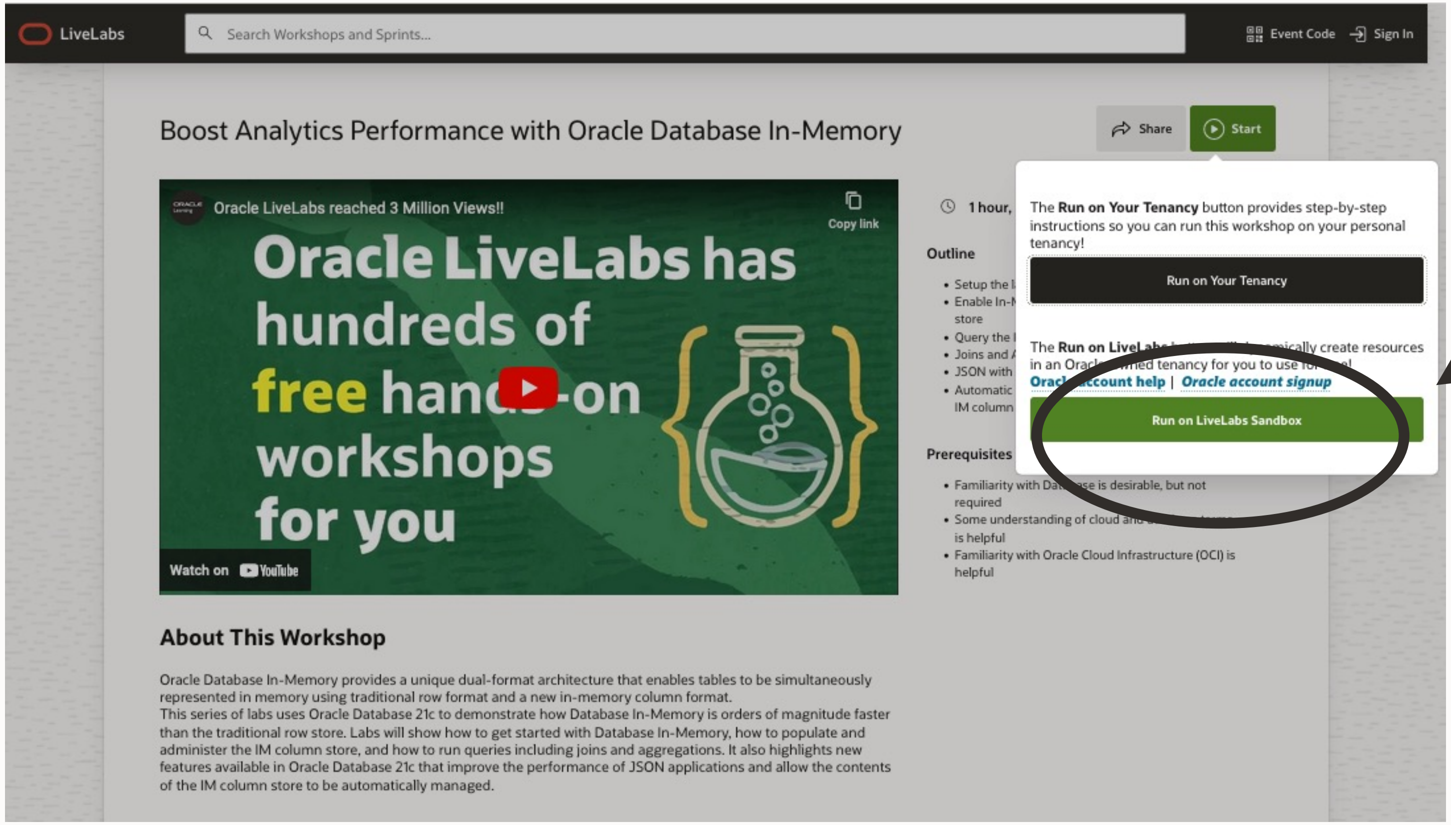
- Differences between Duration and Database Time
  - Parse time is not accounted for in SQL Monitor active reports
  - Non-idle wait events are not accounted for in wait times
  - Parallel Query - Database Time can be much larger than Duration, but that is the point
- No session stats are available
  - This might be a reason to use dbms_xplan and session stats like we do in LiveLabs

Additional Resources

- How to generate a SQL Monitor Report - Updated for 19c!
- SQL Maria - Getting the most out of Oracle SQL Monitor
- SQL Tuning Guide
- MOS Notes
  - Monitoring SQL Statements with Real-Time SQL Monitoring (Doc ID 1380492.1)
  - How to Collect SQL Monitor Output For Parallel Query (Doc ID 1604469.1)
  - How to Monitor SQL Statements with Large Plans Using Real-Time SQL Monitoring? (Doc ID 1613163.1)
  - Oracle Real-time SQL Monitoring Active Reports in Enterprise Manager and the Adobe Flash End-of-Life (Doc ID 2734333.1)

Boost Analytics Performance with Database In-Memory

http://bit.ly/golivelabs


LiveLabs
Search Workshops and Sprints...
Event Code
Sign In
Share
Start
Oracle LiveLabs reached 3 Million Views!!
Copy link
Oracle LiveLabs has hundreds of free hands-on workshops for you
Watch on YouTube

Boost Analytics Performance with Oracle Database In-Memory

1 hour,

The **Run on Your Tenancy** button provides step-by-step instructions so you can run this workshop on your personal tenancy!

Run on Your Tenancy

The **Run on LiveLabs** ... mically create resources in an Oracl... ned tenancy for you to use ... !

Oracl... ccount help | *Oracle account signup*

Run on LiveLabs Sandbox

Outline

- Setup the l
- Enable In-M store
- Query the I
- Joins and A
- JSON with
- Automatic IM column

Prerequisites

- Familiarity with Da... se is desirable, but not required
- Some understanding of cloud and ... is helpful
- Familiarity with Oracle Cloud Infrastructure (OCI) is helpful

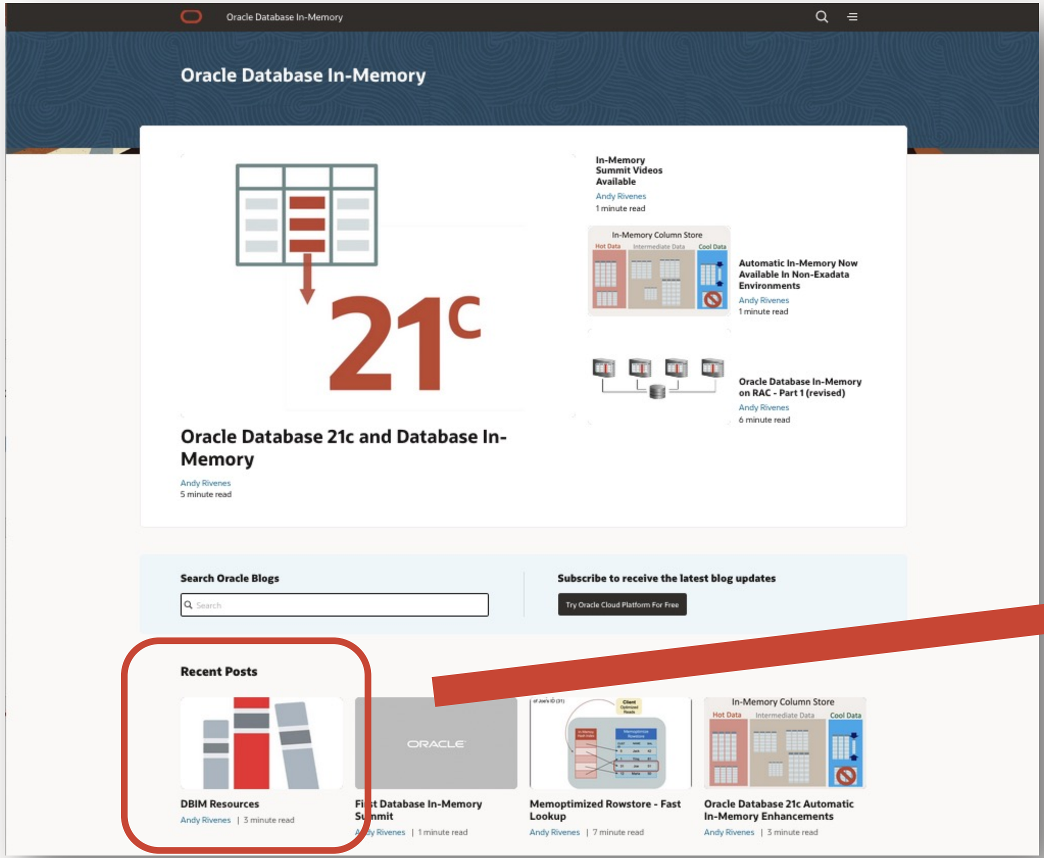
About This Workshop

Oracle Database In-Memory provides a unique dual-format architecture that enables tables to be simultaneously represented in memory using traditional row format and a new in-memory column format.
This series of labs uses Oracle Database 21c to demonstrate how Database In-Memory is orders of magnitude faster than the traditional row store. Labs will show how to get started with Database In-Memory, how to populate and administer the IM column store, and how to run queries including joins and aggregations. It also highlights new features available in Oracle Database 21c that improve the performance of JSON applications and allow the contents of the IM column store to be automatically managed.

Free Tenancy Reservation

No requirement to sign up for the Cloud!

https://blogs.oracle.com/in-memory/dbim-resources



Oracle Database In-Memory

Oracle Database In-Memory

Oracle Database 21c and Database In-Memory
Andy Rivenes
5 minute read

In-Memory Summit Videos Available
Andy Rivenes
1 minute read

Automatic In-Memory Now Available In Non-Exadata Environments
Andy Rivenes
1 minute read

Oracle Database In-Memory on RAC - Part 1 (revised)
Andy Rivenes
6 minute read

Search Oracle Blogs

Search

Subscribe to receive the latest blog updates

Try Oracle Cloud Platform For Free

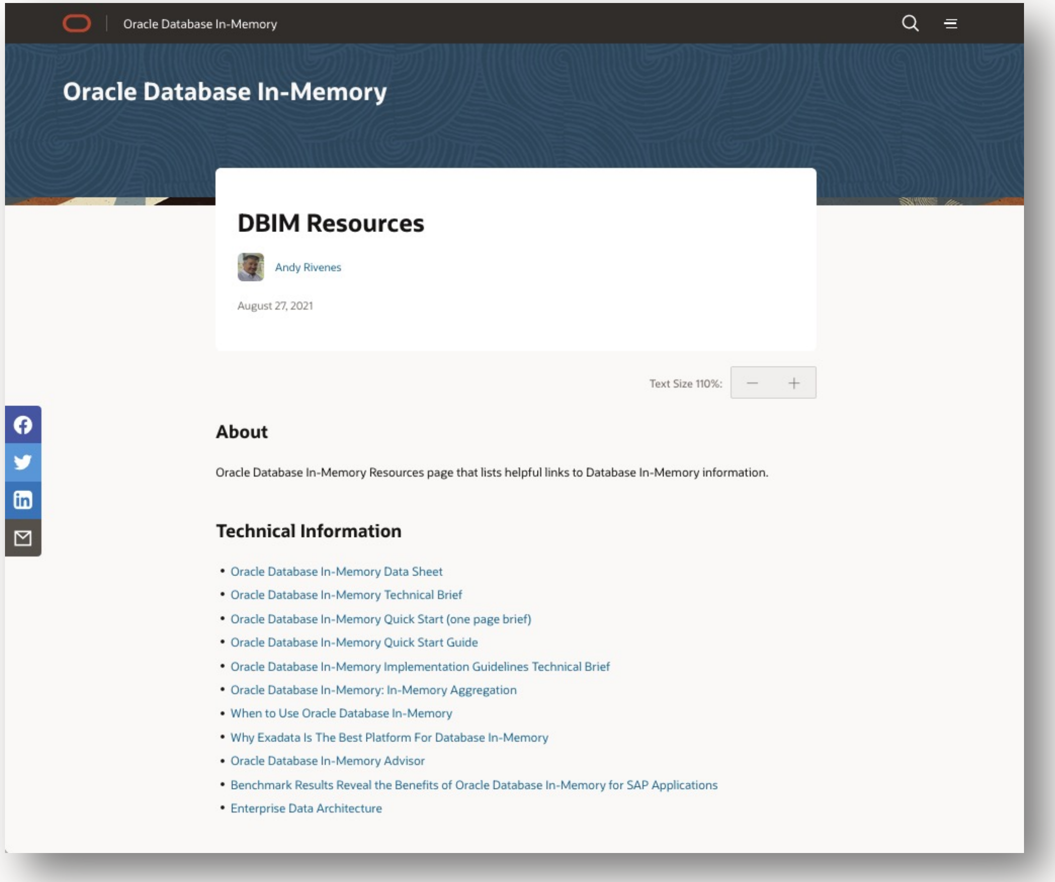
Recent Posts

DBIM Resources
Andy Rivenes | 3 minute read

First Database In-Memory Summit
Andy Rivenes | 1 minute read

Memoptimized Rowstore - Fast Lookup
Andy Rivenes | 7 minute read

Oracle Database 21c Automatic In-Memory Enhancements
Andy Rivenes | 3 minute read



Oracle Database In-Memory

Oracle Database In-Memory

DBIM Resources

Andy Rivenes

August 27, 2021

Text Size 110%: − +

About

Oracle Database In-Memory Resources page that lists helpful links to Database In-Memory information.

Technical Information

- Oracle Database In-Memory Data Sheet
- Oracle Database In-Memory Technical Brief
- Oracle Database In-Memory Quick Start (one page brief)
- Oracle Database In-Memory Quick Start Guide
- Oracle Database In-Memory Implementation Guidelines Technical Brief
- Oracle Database In-Memory: In-Memory Aggregation
- When to Use Oracle Database In-Memory
- Why Exadata Is The Best Platform For Database In-Memory
- Oracle Database In-Memory Advisor
- Benchmark Results Reveal the Benefits of Oracle Database In-Memory for SAP Applications
- Enterprise Data Architecture