CA Unified Infrastructure Management

CA UIM Database Best Practices for Oracle

Version 1

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Contents

Chapter 1: Introduction

This guide covers best practices for deploying, tuning, triaging, and maintaining the UIM Oracle database, also known as the Unified Infrastructure Management (UIM), as deployed on Oracle.

The guide includes new and existing information from UIM documentation, support articles, development tools and other internal and external sources. It organizes the information into the following main sections:

- General description of the UIM
- Prerequisites
- Best Practices
- Best Practices for Oracle
- Performance analysis
- Troubleshooting
- Updating Table Indexes
- Advanced NIS indexing
- SQL Tools and Scripts
- Monitoring the Instance Health with Reports

This document does not attempt to fully document the programming interfaces, theory of operation and structure of CA UIM. The documented is intended to be useful for practical issues in deploying, maintaining and tuning database.

Background on the Unified Infrastructure Management

The UIM database is integral and critical to overall CA UIM system operation and performance. The CA UIM solution requires a database to store the QoS, service level, configuration, alarm (optional) and other data that is collected, processed, and displayed by the system.

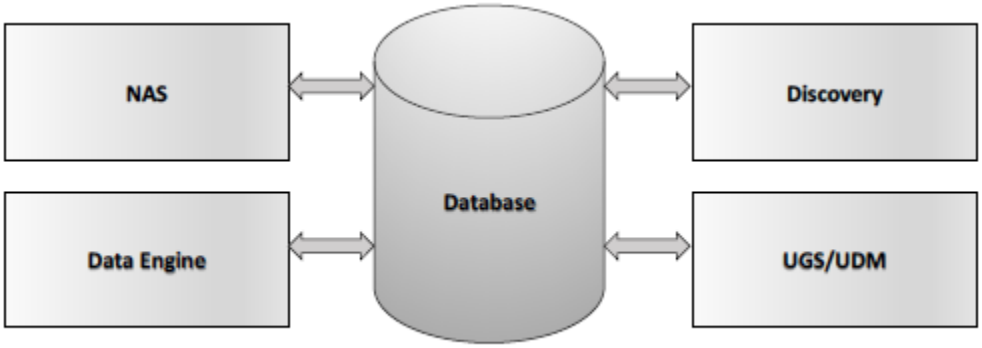
The UIM database was originally introduced into the CA UIM product to hold historical QoS data derived from raw data to enable Service Level Monitoring (SLM) features. With the introduction of expanded reporting and dashboard features in the Unified Management Portal (UMP), the SLM took on an expanded role and was re-named the Nimsoft Information Store, or *NIS*. During the CA UIM 8.0 release, the NIS became known as the UIM Database.

As with any OLTP (Online Transactional Processing) application, the volume of stored data increases over time. When scaled to the needs of large enterprise and managed service providers (MSPs), any database will require periodic maintenance and performance tuning. CA UIM customers, partners, and developers have generated a number of best practice procedures and tuning tips.

Chapter2: The UIM Database

These sections provide an overview of the schema and select components of the UIM Database.

UIM Database Within CA UIM (Simplified UIM Database Architecture)

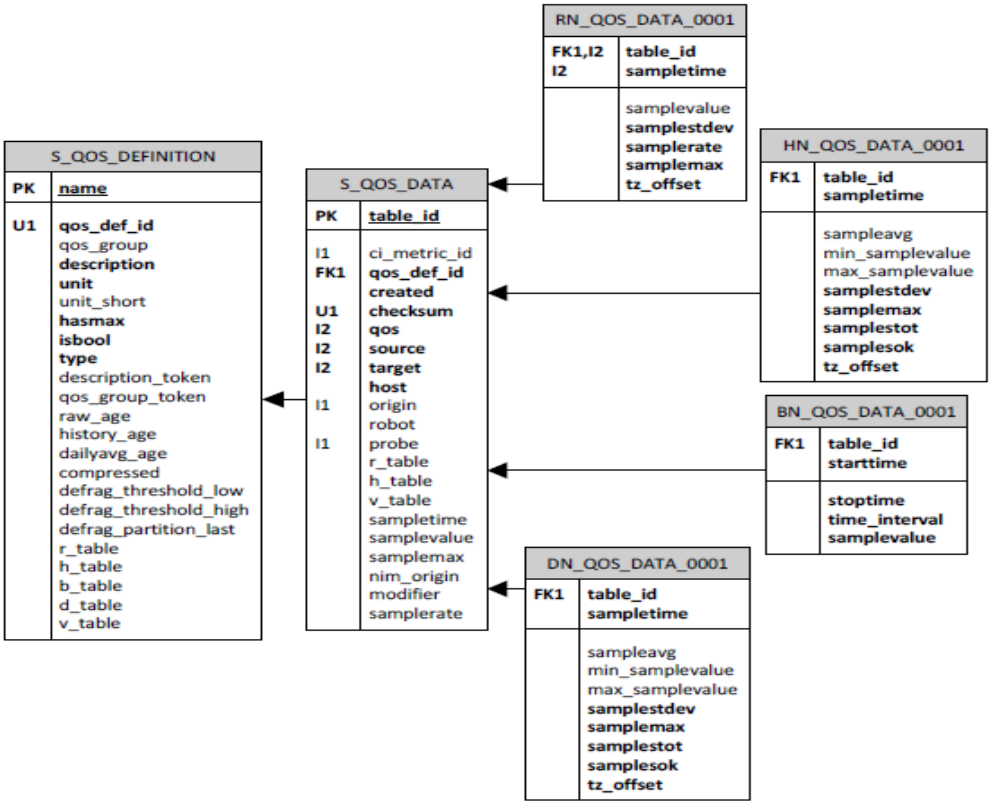


UIM Database Schema Overview

For a picture of how the UIM is structured, below is an abbreviated view from the SQL Developer tool, listing the tables in the UIM database. The full view is many pages long, with hundreds of tables listed.

Table	Children	Parents	Columns	Rows
Account_			16	1
Address			18	0
AnnouncementsDelivery			7	30
AnnouncementsEntry			17	0
AnnouncementsFlag			5	0
AssetCategory			14	0
AssetCategoryProperty			9	0
AssetEntries_AssetCategories			2	0
AssetEntries_AssetTags			2	0
AssetEntry			25	127
AssetLink			9	
AssetTag			9	0
AssetTagProperty				14,654
AssetTag[illegible]_0004			8	890,855
[illegible]QOS_DATA_0005			9	890,855
RN_QOS_DATA_0006			9	892,579
RN_QOS_DATA_0007			8	890,856
RN_QOS_DATA_0008			8	890,856
RN_QOS_DATA_0009			9	890,856
RN_QOS_DATA_0010			9	890,856
RN_QOS_DATA_0011			9	890,857
RN_QOS_DATA_0012			9	890,857
RN_QOS_DATA_001[illegible]			9	94,946
RN_QOS_DA[illegible]			[illegible]	94,946
RN_[illegible]site			7	
WikiNode			11	0
WikiPage			11	2
WikiPageResource			23	2
WorkflowDefinitionLink			4	2
WorkflowInstanceLink			10	0
WSRP_WSRPConsumer			10	0
WSRP_WSRPConsumerPortlet			9	0
WSRP_WSRPProducer			7	0
			7	1
410 Tables			**3,343**	**18,729,661**

Relationships Between Data Engine Tables



Database Tables

These tables are of primary interest:

Table name or pattern	Type of data held
S_QOS_DATA	QoS data used by the data_engine
RN_QOS_DATA_xxxx	Raw QoS data (one RN_table for each qos_def_id)
HN_QOS_DATA_xxxx	Aggregated (1hr interval) QoS data
DN_QOS_DATA_xxxx	Aggregated (1 day interval) QoS data
BN_QOS_DATA_xxxx	Baseline tables
CFG_*	ACE configuration information
CM_GROUP	NIS_Server/USM
CM_DEVICE	Discovery/configuration
CM_NETWORK	Discovery
CM_NIMBUS_*	Discovery/configuration
GRP_*	group_server, deprecated
ump* and QUARTZ_*	UMP and LifeRay
*_	Liferay
*_t_*	Tmp, dashboard_engine
NAS*	Alarms*
tbnLogging	Sp logging

tbnVersion	Version info
S_SLA_* and S_SLO_*	Service Level Agreement info

***Note:** Alarm data is held in a separate NAS database. If the NIS-bridge feature of NAS is enabled, alarms are replicated from NAS to NIS. UMP accesses alarm data from the NIS using the NIS-bridge feature.

RN_QOS_DATA_ Table Columns

The RN_QoS_Data_tables hold raw QoS data. This QoS data is written once and never updated. It is aggregated by 1 hour periods and stored in HN_QoS_Data_tables_xxxx. The table has the following columns:

TableID	Sampletime	Samplevalue	Samplestdev	Samplerate	Samplemax	Compressed	Tz_offset	inserttime

Column Name	Description
tableID	unique identifier; key for looking up time series data
Sampletime	time the sample was taken
Samplevalue	QoS value
Samplestdev	standard deviation of the sample
Samplerate	Rate of sampling
Samplemax	Maximum sample value (e.g. 100%)
Tz_offset	time zone offset

RN_tables Indexes

The default indexes on RN_tables are optimized for writing data:

Index	Description
Idx0	SAMPLETIME, TABLE_ID
Idx1	TABLE_ID, SAMPLETIME, TZ_OFFSET, SAMPLERATE

There is no primary key implemented on RN_QoS_DATA_tables as both tableID and sampletime can be duplicated.

About Table Partitioning

As RN_QoS_DATA_tables grow in size the time needed to order and index them increases, slowing performance. Subdividing tables into multiple partitions offers several benefits:

- Partitioning allows data loads, index creation and rebuilding, and backup/recovery to occur at the partition level rather than on the entire table.
- Partitioning improves query performance. In many cases, the results of a query can be achieved by accessing a subset of partitions rather than the entire table.
- Partitioning can significantly reduce the impact of scheduled downtime for maintenance operations.

UIM Database tables can be automatically partitioned if you are using Oracle Database. The partitioning scheme is a sliding window partition on sampletime with one partition per day. If partitioning is enabled, data is aged out of the RN tables by dropping the old partitions rather than deleting rows.

Manual Table Partitioning

Manual partitioning of the RN_QOS_DATA tables is achieved by running the data_engine partitioning stored procedure manually. You may wish to apply partitioning to your largest RN_QOS_DATA tables at first. We define large tables as those having over 100 million rows. Partitioning in a selective manner gives you more control over when tables are partitioned as this process can take considerable time to complete.

Below is the full process and query for manually partitioning the tables associated with a QoS definition:

IMPORTANT WARNING : Note that once you begin this process, you have 14 days to complete it. As soon as you begin the next step here there's no turning back. If the process is not completed within 14 days, then you risk losing 1 days' worth of data each day until the process is completed.

1. Open the data_engine probe using Raw Configure mode (select the probe then hold down the SHIFT key and right-click to open in Raw Configure mode)
2. Change the value of data_management_partition from "no" to "yes" and then restart the data_engine.
3. This will commit a change to the database which is needed for the next steps.
4. Now open Raw Configure again, and set data_management_active to "no" and restart data_engine again.
5. From this point on, it's permissible to leave data_engine active – there is no need to deactivate it.
6. Determine which table to partition first - in the example below we'll use RN_QOS_DATA_0035 but this will depend on table size. (It's advisable to start with the largest tables as mentioned above.)
7. The number of this table (e.g. 0035) will be the number you will plug into the following query, for the "lQosDefId" parameter, to partition the desired table (strip the leading zeroes). This will also match the qos_def_id associated with this table in S_QOS_DEFINITION.

```
set serveroutput on size 30000;
declare
lQosDefId number default 35;   -- change this number to reflect which RN table you wish to partition
lMode varchar2(127) default 'all';
lStartTime timestamp default systimestamp;
lLogLevel number default 5;
lRawCompressedRows number;
lHistoryCompressedRows number;
lRawShrunkRows number;
lHistoryShrunkRows number;
lErrorMessage varchar2(1024);
```

```
lReturnCode number;
begin
spn_de_DataMaint(lQosDefId, lMode, lStartTime, lLogLevel, lRawCompressedRows,
lHistoryCompressedRows, lRawShrunkRows, lHistoryShrunkRows, lErrorMessage, lReturnCode);
dbms_output.put_line('lRawCompressedRows=' || to_char(lRawCompressedRows));
dbms_output.put_line('lHistoryCompressedRows=' || to_char(lHistoryCompressedRows));
dbms_output.put_line('lRawShrunkRows=' || to_char(lRawShrunkRows));
dbms_output.put_line('lHistoryShrunkRows=' || to_char(lHistoryShrunkRows));
dbms_output.put_line('lErrorMessage=' || lErrorMessage);
dbms_output.put_line('lReturnCode=' || to_char(lReturnCode));
end;
```

Once you have run this query against all the tables you wish to partition, you should re-enable data_engine maintenance as mentioned above, by setting data_management_active to "yes" again and restarting data_engine. On the next maintenance run, the data_engine will partition the remaining tables automatically.

Please find the below KB article about manual partitioning of tables. It is recommended to go through this reference before manually partitioning big tables.

https://comm.support.ca.com/kb/how-to-manually-partition-the-uim-database-for-oracle/KB000071942

Chapter 3: Prerequisites

This guide assumes that UIM Server and the UIM Database are installed and running. We recommend that you review the requirements and prerequisites for proper UIM Server and database installation. For more information, see the article on Pre-Installation Planning available at wiki.ca.com/uim .

Note: These links are for UIM 8.2. Please refer to the documentation that corresponds to your specific version.

Supported Database Versions

The following database and OS versions are supported:

Database	Supported Operation System (64 bit only)
Oracle 11g, Oracle 12c	Windows Server 2012 R2, Windows Server 2014, Windows Server 2016, Redhat, Ubuntu

Hardware Requirements

We recommend deploying the database on a dedicated physical server.

Database Server Requirements

While every situation is unique, the following deployment size categories give you a starting point for assessing your hardware requirements:

- **Small (One hub, fewer than 100 robots)** - Modest deployment, such as a proof-of-concept for a small business
- **Medium (Up to five hubs, fewer than 250 robots)** - Medium-scale deployment, such as a small government agency
- **Large (Up to twenty hubs, fewer than 500 robots)**
- **Major (Up to fifty hubs, fewer than 1000 robots)**
- **Over 50 hubs or over 1000 robots** - Consult with CA professional services or a CA UIM certified partner.

Deployment size	Processor *64-bit XEON-class, 2.0 GHz or better*	Memory	Storage recommendations
Small	One dual-core	8 GB	Obtain at least 1 TB storage for the database.
Medium	One or two quad-core	12 GB	Use RAID 10 (for speed and reliability).
Large	Two quad-core	12 GB to 18 GB	Spread database files across multiple disks to improve I/O.

			Choose drive subsystems with low latency/seek times, high spindle speeds, high interconnect bandwidth.
Major	Two quad- or eight-core	18 GB to 24 GB	Continually consider data redundancy, synchronization, and database growth.

About Database Performance

Relational database server performance is heavily affected by disk I/O performance and server bus bandwidth. Crowded VM hosts, clusters, or heavily shared storage in VM environments are not recommended for UIM database hosting.

CA recommends starting with at least 1TB of RAID 10 storage for the UIM Database. Also, consider spreading the database files across multiple disks (LUNs) to improve I/O performance. Choose drive subsystems with low latency and seek times, high spindle speeds, and high interconnect bandwidth.

The data redundancy/synchronization model needs to be considered on an on-going basis, taking into account the growth of the database. Selecting the right storage solution is beyond the scope of this document, we recommend that you discuss this with your storage vendor/VAR/consultant.

Chapter 4: General Best Practices

This section covers operational UIM Database best practices.

Best Practice	Comment
Read and observe documented pre-requisites and pre-install information	See the article on Pre-installation Planning available at wiki.ca.com/uim .
Always make a backup of your database before upgrading major CA UIM components (UIM Server and UMP)	Some upgrades contain a non-reversible upgrade script that changes the database structure of some tables.
Run get database information oracle on a regular basis	Establish a baseline so that system changes can be easily seen and CA UIM support can quickly respond to issues
Use v$session	This View provides information on status.
Set up periodic index maintenance	See the article on data_engine configuration and review the notes and warnings regarding the cost of setting up index maintenance.
Carefully consider the implications of database configuration settings	See the article on data_engine configuration.
Check database size	Use this query to check database size
Check for and correct index fragmentation on a regular basis.	Use get_index_fragmentation to check for index fragmentation. Automatic index maintenance can also be scheduled as described above.
Check for missing nodes in Dynamic Views	Find_missing_nodes_in_dynamic_views

Storage

This section covers storage considerations for UIM database.

Best Practice	Comments
Determine an overall storage strategy	Discuss a storage strategy with your storage vendor.

Backup

This section covers the UIM database backup best practices.

Best Practice	Comments
Plan and schedule regular backups of the database	
Test restore operation before it is needed	
Ensure sufficient disk capacity for backups	
Document backup and restore procedures	

Periodic Maintenance

This section covers the periodic maintenance of the UIM database.

Best Practice	Comment
Set up periodic index maintenance	See the article on data_engine configuration and review the notes and warnings regarding the cost of setting up index maintenance.
Identify skewed and outdated index and column statistics and make sure they are representative and current	Index statistics are used by the Oracle query optimizer to help it determine if and when an index should be used when executing a query.
Database and log file protection and management	
Temp data maintenance	
Data corruption detection	
Performance monitoring	

Chapter 5: Best Practices for Oracle

This section provides a checklist of values and properties to check when setting up and deploying Oracle database.

Checklist

This checklist is hierarchical in method, starting with hardware and OS settings, then Oracle instance, then the UIM database and its maintenance.

- Network Communication
- Cluster Environment
- Windows/Linux Configuration
- Disks configuration
- Oracle Instance Configuration
- Database Level Configuration
- Database Maintenance and Backup Strategy
- Additional Resources

Network Communication

This section covers the network communication best practices for UIM database.

For Windows:

Applicable Versions of Windows	Item	Recommendation
All	NIC full duplex	Network adapters and switch ports should have matching duplex levels or transfer speed settings. Full duplex provides better performance.
All	Network settings	Latest basic input/output system (BIOS) update for the server should be installed. Latest firmware update for the network adapter should be installed. Latest driver update for the network adapter MUST be installed.
All	NetBIOS and Server Message Block enabled	Disable NetBIOS and Server Message Block

		Important! Make sure NetBIOS is not in use.

Cluster Environment

This section covers the cluster environment best practices for UIM database.

Applicable Versions of OS and OS Version	Item	Recommendation
All	Cluster nodes hardware	Cluster nodes should have nearly identical hardware on all cluster nodes to simplify configuration and eliminate potential compatibility problems.
All	Memory adjustment	In an Active-Active environment, max memory for the Oracle instances should be set in a way that the total memory in the weakest node is split between the nodes. This will ensure that when all instances failover to one node, they will be able to that quickly and with no memory issues.

Windows/Linux Configuration

For more information about Windows Server performance, see Performance Tuning Guidelines for Windows Server 2012 R2.

Applicable Versions of OS	Item	Recommendation
All	Latest service pack	Implement the latest service pack and hotfixes.
All	64-bit hardware and software	Required
All	Paging file	Pagefile should be 1~1.5 times the amount of RAM and should NOT be placed on a drive that contains database files. **Important!** We recommend that you create multiple page files on different disk partitions beside C:\ (or even different

		disk subsystems) for performance reasons.
All Windows	System properties > Advanced setting	**Processor scheduling:** Select Background services. **Memory usage:** Select Programs
All	Unnecessary Services / applications	Number of running apps and services should be minimal. Unnecessary services should be stopped and disabled (Messenger, wireless configuration, etc.).
All	Anti-virus	The best Practice for Oracle is *not* installing anti-virus on a dedicated Oracle environment.

Disks Configuration

This section discusses the general configuration of the disk system (for detailed information regarding the placement of database files, refer to Database level configuration).

Applicable Versions of OS	File Type	Recommendation
All	Data Files	RAID 1+0 or 0+1 is the recommended RAID level for data files with high I/O requirements.
All	Undo Files	RAID 1+0 or 0+1 is the recommended RAID level for undo files in databases.
All	Archive Log Files	Always multiplex these files even when using RAID. RAID1 is recommended level for Archive Log files.
All	Online Redo Logs	Always multiplex these files even when using RAID. RAID1 is recommended level for Online Redo Log files.
All	Control Files	Always multiple Control Files we should be used whether we use RAID or not. RAID 1+0 or RAID 0

		are recommended for these files.

Oracle Instance Level Configuration

Note: This section describes the Oracle instance best practices.

Applicable Oracle Versions	Item	Recommendation
All	Latest service pack	Latest service pack and hotfixes.
All	Oracle installed on domain controller	Oracle should never be installed on a domain controller
All	Dedicated machine for Oracle	Best Practices imply that Oracle should be installed on its own dedicated host machine
All	Server allowed protocols (TCP/IP, VIA, NPs, etc.)	Limit the supported protocols.
All	Database files location setting	The database default *data* location should be changed to point to a dedicated disk for data files. The default location for *logs* should be changed to a dedicated location for log files that is optimized for write operations.

Oracle Database Level Configuration

This section describes database level configuration best practices for UIM database.

Applicable Oracle Versions	Item	Recommendation
All	Database files placement	Refer the Disks on SANS **Important!** At a minimum always separate data files from log files on separate physical disks.

All	Database files growth	Use the following general guidelines: • File growth should not be very large because user activity will wait for the file operation to complete. • File growth should not be very small to avoid issues with the file becoming full. • Growth should never be in terms of % but in terms of MBs.
All	Archive log Files	Automatic archiving should be done so that all filled redo log files will automatically archive.
All	Automatic Memory Management	Oracle strongly recommends to enable Automatic Memory Management to handle SGA and PGA memory components dynamically by the server.
All	Automatic Shared Memory Segment	If Automatic Memory Management is disabled, then Oracle recommends to enable Automatic Shared Memory Management to handle all SGA components dynamically by the server.

Database Maintenance and Backup Strategy

This section covers the backup strategy best practices for UIM database.

Applicable Oracle Versions	Item	Recommendation
All	Recovery models	If the archive log mode is not enabled and incremental backup is not enabled on database, then there is no point in time recovery. In case of a failure, such as database corruption, the database will need to be restored from the latest FULL

		database backup/differential backup. If the Archive log mode is enabled and we are following incremental backup, then make sure all files are backed up properly. For better performance, don't backup the databases or Archive logs to the same physical disk.
All	Index maintenance tasks	For better performance, it is strongly advised to rebuild indexes based on their size and fragmentation level.
All	Update statistics maintenance task	If Automatic Statistics Gathering option is enabled on in the database and there is no massive insertion/update of data, this task is not required (which is true in most cases).
All	Maintenance and backup files cleanup	Don't forget to clean up old backup files according to the database SLA and the amount of free space on the backup drives.
All	Database integrity checks	It is important to run integrity checks on the database. The earlier you find consistency issues, the better.

Additional Resources

https://docs.oracle.com/cd/E11882_01/server.112/e41573/memory.htm#PFGRF94254

https://docs.oracle.com/database/121/TGDBA/memory.htm#TGDBA293

https://docs.oracle.com/cd/E11882_01/server.112/e10803/config_storage.htm#HABPT4790

Chapter 6: Performance Analysis

This section provides a checklist for analyzing overall database performance, and can be used for continuous performance analysis and database tuning.

Checklist

Most issues listed in the right column link to a listing of the SQL query that will generate the desired output (valid for Oracle only):

Performance Analysis Area	Issue to identify
Analyze Wait Statistics	Wait times (see example below)
Perform Index Analysis, identifying top issues	Most Costly Unused Indexes Top Costly Missing Indexes Tables without Primary key Tables with no indexes
Identify top SQL queries according to high resource utilization	Top SQL with Highest CPU Top SQL with Highest I/O Top SQL with Highest Duration Top DISK intensive queries

Example Query (Wait Statistics)

This query provides wait statistics that give a good indication of resource bottlenecks from an Oracle perspective:

```
SELECT * FROM

(SELECT w.event AS "event", w.time_waited as "Time Waited", round(w.time_waited/tw.twt*100,2) AS wait_pct

FROM

 gv$system_event w, (select inst_id, sum(time_waited) twt from gv$system_event

 WHERE time_waited>0 AND

event NOT IN ('Null event', 'client message', 'rdbms ipc reply', 'smon timer', 'rdbms ipc message', 'PX Idle

Wait', 'PL/SQL lock timer', 'file open', 'pmon timer', 'WMON goes to sleep', 'virtual circuit status',

'dispatcher timer', 'SQL*Net message from client', 'parallel query dequeue wait', 'pipe get')

  group by inst_id

  ) tw

WHERE

w.inst_id = tw.inst_id and w.time_waited>0 and

round(w.time_waited/tw.twt*100,2) > 1 and
```

w.event NOT IN ('Null event', 'client message', 'rdbms ipc reply', 'smon timer', 'rdbms ipc message', 'PX Idle Wait', 'PL/SQL lock timer', 'file open', 'pmon timer', 'WMON goes to sleep', 'virtual circuit status', 'dispatcher timer', 'SQL*Net message from client', 'parallel query dequeue wait', 'pipe get')

ORDER by "Time Waited" desc

)

WHERE rownum <= 10;

Example Results

	event	Time Waited	WAIT_PCT
1	PX Idle Wait	12013323186	54.99
2	Space Manager: slave idle wait	5602715462	25.65
3	DIAG idle wait	711605621	3.26
4	LGWR worker group idle	711251419	3.26
5	Streams AQ: qmn slave idle wait	355923452	1.63
6	VKRM Idle	355909805	1.63
7	shared server idle wait	355907746	1.63
8	AQPC idle	355906000	1.63
9	lreg timer	355903822	1.63
10	heartbeat redo informer	355902852	1.63

Chapter 7: Troubleshooting

This section provides best practices to triage and troubleshoot a distressed database.

Best Practice	Comments
Recognize a distressed database, learn the signs	One symptom is the data_engine queue backing up
Investigate what other processes are running	Use V$session
Investigate index fragmentation	Use get_index_fragmentation to check index fragmentation
Check disk subsystem(s) and drive failure	Performance impact to RAID Check RAID manager console
Resource constrained?	Task Manger (Windows) Top command in Linux and Solaris

Troubleshooting High CPU Issues

If UIM Server is facing slowness issue and identify the problem is with CPU (Identified using Task Manager in Windows and top or vmstat in Linux), perform the following checks:

High CPU consumption can occur because of following two reasons:

1. **Server side CPU bottleneck**

Check the below counters to make sure server CPU is the bottleneck:

Windows

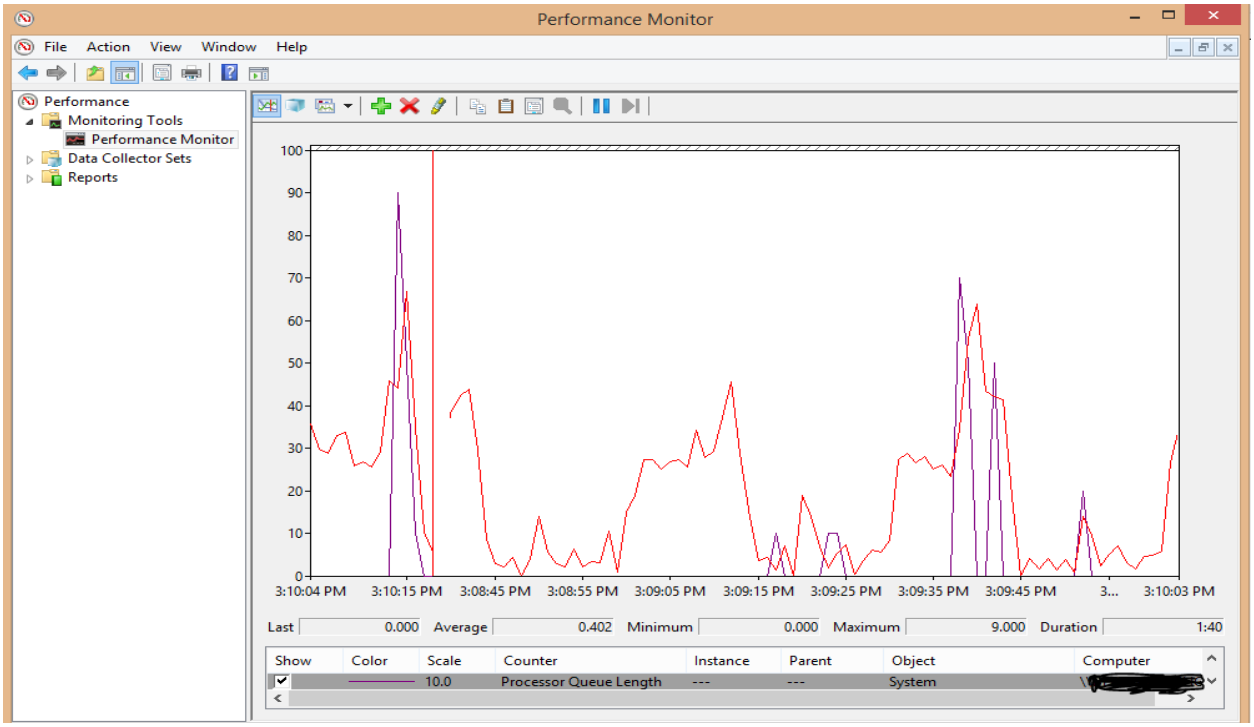
Check **processor queue length** counter in Windows server. If this number is greater than 0, that means there are more requests per core than the system can handle. This can be a cause for significant performance issues in oracle.

You can launch Performance Monitor a few different ways.

Method 1: Start, Control Panel, Administrative Tools, Performance Monitor.

Method 2: Start, search PerfMon.exe

When Performance Monitor is launched, an interface similar to the following is displayed:



Linux:

Check run queue (r value) **cpu_count** in Linux. To find the oracle server has the CPU bottleneck check the run queue value (per vmstat). If this value Exceed the number of processors on the server (cpu_count) then server has the CPU bottleneck. The following example screenshot shows the vmstat output.

```
[root@tuuja01-I7494 ~]# vmstat
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 1  0 1596840 168080      0 834300    0    0    12     1    3    1  0  0 100  0
 0
[root@tuuja01-I7494 ~]#
```

Also, If "**resmgr:cpu quantum**" event appears in top 5 timed events on a AWR.

If the processor queue length or r value in vmstat command is lower than the CPU cores, then the problem is on the application side. If this value is always more than CPU cores, increase the CPU cores in the database server.

2. **Application-Level CPU Bottleneck**

If server has enough CPU then consider tuning the database.

Checking database server side bottleneck:

- Find the queries that are running when the CPU is at its 100% utilization.
- Update the statistics and check the fragmentation on the tables.
- Check for any big table scans and create appropriate indexes to avoid scans.
- Tune the queries that are taking longer duration by using the Index Tuning Advisor, and apply the recommendations.
- Generate the AWR and ADDM reports to get the recommendations based on the statistics.

Troubleshooting Memory Issues

If UIM Server is facing slowness issue and if we identify the problem is with Memory (Identified using Task manager in windows and top or vmstat in Linux), Then perform the following checks:

1. Check the database server has enough memory to handle the workload or not

The buffer cache hit ratio calculates how often a requested data block has been found in the buffer cache without requiring physical disk access.

To check the memory bottleneck on the server, find the buffer cache hit ratio using below query:

```
SELECT   Sum(Decode(a.name, 'consistent gets', a.value, 0)) "Consistent Gets",
         Sum(Decode(a.name, 'db block gets', a.value, 0)) "DB Block Gets",
         Sum(Decode(a.name, 'physical reads', a.value, 0)) "Physical Reads",
         Round(((Sum(Decode(a.name, 'consistent gets', a.value, 0)) +
         Sum(Decode(a.name, 'db block gets', a.value, 0)) -
         Sum(Decode(a.name, 'physical reads', a.value, 0)))/
         (Sum(Decode(a.name, 'consistent gets', a.value, 0)) +
         Sum(Decode(a.name, 'db block gets', a.value, 0))))
        *100,2) "Hit Ratio %"
FROM   v$sysstat a;
```

The buffer cache hit percentage should be higher than 90 percent. If not, then there is an indication that the server has less memory than required.

2. Check if there is a constant swapping on the server or not.

If si and so columns of the vmstat output shows constant swapping, then server is facing memory issue.

Tuning oracle memory

There are two memory structures called SGA (System Global Area) and PGA (Program Global Area). Oracle strongly recommends to enable the method called “Automatic Memory Management”.

Automatic Memory Management

If you enable the Automatic memory management, oracle dynamically manages the SGA and PGA memories. If you disable the Automatic memory management and if you want to manage SGA memory components, then consider enabling “Automatic shared memory management”.

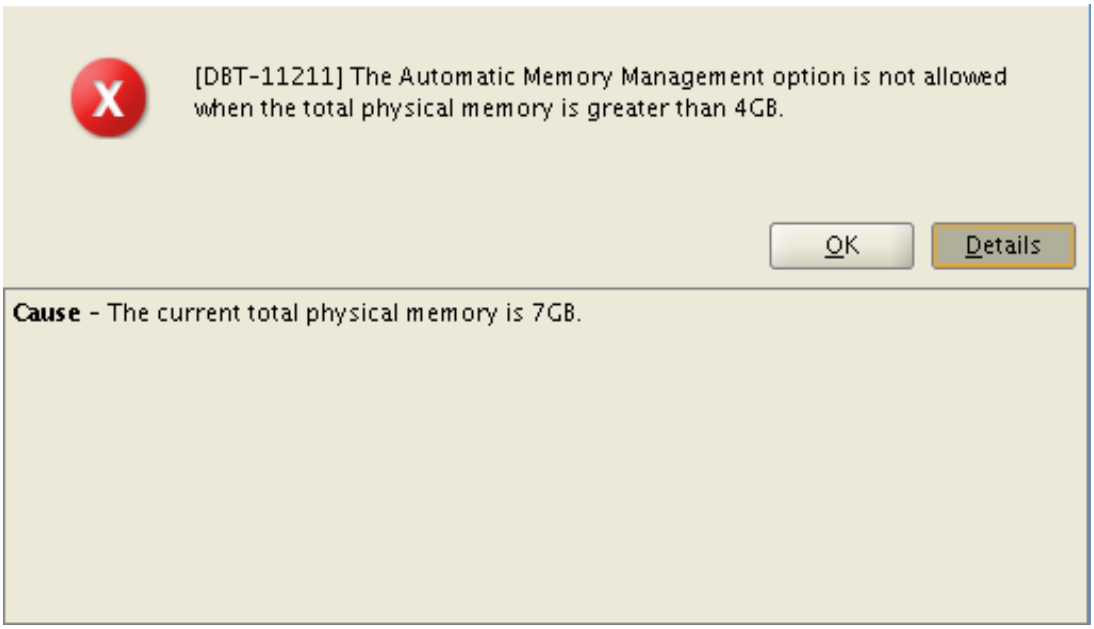
If you want to enable Automatic memory management set “MEMORY_TARGET” and “MEMORY_MAX_TARGET”. If you want tuning advice for this value, then use V$MEMORY_TARGET_ADVICE. Use below commands to change the memory targets:

ALTER SYSTEM SET MEMORY_TARGET = nM SCOPE = SPFILE;

ALTER SYSTEM SET MEMORY_MAX_TARGET = nM SCOPE = SPFILE;

Recommendation:

If the database is created using Database Configuration Assistance (DBCA) and system memory is lower than 4GB, Automatic Memory Management is enabled by default. Automatic Memory Management is not recommended if the system memory is higher than 4GB. In fact, in Oracle 12.2, if you enable Automatic Memory Management if the system memory is higher than 4GB, the installation is stopped and the following error is displayed:



Therefore, it is recommended to enable Automatic Shared Memory Management if the system memory is higher than 4GB.

Automatic Shared Memory Management

If Automatic memory management is disabled, then oracle database uses Automatic shared memory management to manage SGA Memory. If you want to enable Automatic shared memory management then set “SGA_TARGET” and “SGA_MAX_SIZE”. Use below commands to change SGA target memory:

ALTER SYSTEM SET SGA_TARGET = nM SCOPE = SPFILE;

ALTER SYSTEM SET SGA_MAX_SIZE = nM SCOPE = SPFILE;

Recommendation:

If Automatic Memory Management is disabled or If we have a system memory greater than 4GB, it is recommended to enable the Automatic Shared Memory management.

Sizing SGA and PGA manually

SGA has many components like Buffer cache, Shared pool, Java pool etc.

Below are the dynamic views and their recommendations:

Database Cache Advice

Predicts how changes to the buffer cache will affect physical reads.

Query to find best buffer cache value:

SELECT size_for_estimate, buffers_for_estimate, estd_physical_read_factor, estd_physical_reads

FROM V$DB_CACHE_ADVICE

WHERE name = 'DEFAULT'

AND block_size = (SELECT value FROM V$PARAMETER WHERE name = 'db_block_size')

AND advice_status = 'ON';

Recommendation:

Increase the buffer cache (DB_CACHE_SIZE) to the value recommended by the size_for_estimate column based on the estd_physical_reads_factor value.

Note: If SGA_TARGET is set then the default value of DB_CACHE_SIZE is 0 (internally determined by the Oracle Database). If the parameter is specified, then the user-specified value indicates a minimum value for the memory pool.

Use below query to check DB_CACHE_SIZE:

SHOW PARAMETER DB_CACHE_SIZE

Tuning Shared Pool Size

Query to find shared pool value:

```
SELECT SUM(gets) “Data Dict. Gets", SUM(getmisses) “Data Dict. Cache Misses"
 , TRUNC((1-(sum(getmisses)/SUM(gets))) *100, 2) "Data Dict Cache Hit Ratio"
 , TRUNC(SUM(getmisses)*100/SUM(gets), 2) “% Missed"
FROM v$rowcache;
```

Recommendation:

The Dictionary Cache Hit% should be greater than 90% and missed% should be lesser than 15%. If not consider increase SHARED_POOL_SIZE.

Tuning Redo Log Buffer

Query to find Redolog space ratio:

```
 SELECT e. value "redolog space request",
 s.value "redolog space wait time",
Round(e.value/s.value,2) "Redolog space ratio"
FROM
v$sysstat s, v$sysstat e
WHERE s.name = 'redo log space requests'
AND e.name = 'redo entries';
```

Recommendation:

If the ratio of redo log space is less than 5000 then increase redo log buffer until this ratio stops falling. There should be no log buffer space waits. Making log file bigger or moving the log file to faster disk will help.

PGA Advice

Query to get the best PGA_AGGREGATE_TARGET value:

```
SELECT ROUND(pga_target_for_estimate/1024/1024) target_mb,
 estd_pga_cache_hit_percentage cache_hit_perc,
 estd_overalloc_count
```

FROM v$pga_target_advice;

Recommendation:

Increase the PGA_AGGREGATE_TARGET to the value recommended by the target_mb column based on the cache_hit_perc value.

If Buffer Cache Hit % is more than 90% and there is swapping is happening in the server then problem is at the application side.

Do the following to fix the memory related issues:

1. Find the queries that are consuming more memory and tune them by creating all required indexes.
2. Update the statistics and check the fragmentation on the tables.
3. Check for any big table scans and create appropriate indexes to avoid scans.
4. Tune the queries that are taking longer duration by using the Index Tuning Advisor and apply the recommendations.
5. Generate the AWR and ADDM reports to get the recommendations based on the statistics.

NLS Parameters for Slow Performance

If there is a slow performance at UIM probe with Oracle UIM DB, check this.

SELECT * FROM NLS_INSTANCE_PARAMETERS WHERE parameter IN ('NLS_COMP', 'NLS_SORT');

If values for these parameters are not BINARY, then suggest

ALTER SYSTEM SET NLS_COMP=BINARY SCOPE=SPFILE;

ALTER SYSTEM SET NLS_SORT=BINARY SCOPE=SPFILE;

According to Oracle doc, BINARY give us the best performance.

Additional Resources

https://docs.oracle.com/cd/B28359_01/server.111/b28310/memory003.htm#ADMIN11201

https://docs.oracle.com/cd/E11882_01/server.112/e25494/memory.htm#ADMIN11234

https://docs.oracle.com/database/121/TGDBA/tune_sga.htm#TGDBA336

Appendix A: SQL Tools and Scripts

This section lists SQL code to perform the following tasks:

Get Database Size Information

This script provides a listing of data tables, sorted by size.

```
SELECT
segment_name AS "Table Name",
tablespace_name AS "Tablespace Name",
bytes/1048576 AS "Size MB"
FROM
DBA_SEGMENTS
WHERE
OWNER = 'SYSTEM' AND
SEGMENT_TYPE = 'TABLE'
ORDER BY bytes/1048576 desc;
```

Get Database Information (Oracle)

This script provides a summary of the UIM Database deployment size, what is being monitored, probe versions, frequently used probes, database size, data location, and information on specific data tables.

```
-- summary of deployment size

select '1. # qos definitions' as item, COUNT(*) as cnt from S_QOS_DEFINITION

union

select '2. # qos objects', COUNT(*) from S_QOS_DATA

union

select '3. # robots', COUNT(*) from CM_NIMBUS_ROBOT where is_hub = 0 and alive_time >
TO_CHAR(SYSDATE -(1/24), 'dd-mon-yy hh24:mi:ss')

union

select '4. # hubs', COUNT(*) from CM_NIMBUS_ROBOT where is_hub = 1 and alive_time >
TO_CHAR(SYSDATE -(1/24), 'dd-mon-yy hh24:mi:ss')

union

select '5. # computer systems', COUNT(*) from CM_COMPUTER_SYSTEM where alive_time >
TO_CHAR(SYSDATE -(1/24), 'dd-mon-yy hh24:mi:ss')

union

select '4. # hubs', COUNT(*) from CM_NIMBUS_ROBOT where is_hub = 1 and alive_time >
TO_CHAR(SYSDATE -(1/24), 'dd-mon-yy hh24:mi:ss')

union

select '5. # computer systems', COUNT(*) from CM_COMPUTER_SYSTEM where alive_time >
TO_CHAR(SYSDATE -(1/24), 'dd-mon-yy hh24:mi:ss')

-- whats being monitored

select probe, COUNT(distinct qos) as QOS, COUNT(distinct source) as sources, COUNT(distinct target) as
targets

from S_QOS_DATA

group by probe

order by targets desc, QOS desc

-- is everything running the same versions?

select probe_name, pkg_version, COUNT(*) as cnt from CM_NIMBUS_PROBE

where active = 1
```

and probe_name in ('controller', 'hub')

group by probe_name, pkg_version;

-- most frequently used probes

select probe_name, count(*) as Cnt

from CM_NIMBUS_PROBE

group by probe_name

order by Cnt desc

Get Index Fragmentation

This script will be used to analyze and provide the index fragmentation

ANALYZE INDEX CM_COMPUTER_SYSTEM_PK__CM_COMP VALIDATE STRUCTURE;

SELECT name, height,lf_rows,lf_blks,del_lf_rows,(del_lf_rows/lf_rows)

*100 as ratio FROM INDEX_STATS;

Recommendation:

Follow these rules to determine whether you need to rebuild the index:

1) If the index has height greater than four, rebuild the index.

2) The deleted leaf rows should be lesser than 20%.

Syntax:

Sql > ALTER INDEX index_name REBUILD;

Find Missing Nodes in Dynamic Views

-- # of origins not matching

select cs.origin as cmOrigin, d.origin as sqdOrigin , count(*) from CM_COMPUTER_SYSTEM cs inner join CM_NIMBUS_ROBOT r on cs.ip = r.ip and cs.origin = r.origin inner join S_QOS_DATA d on cs.ip = d.host where d.origin <> r.origin group by cs.origin, d.origin order by count(*) desc

-- query to see if origins match up between CM_COMPUTER_SYSTEM, CM_NIMBUS_ROBOT, S_QOS_DATA

select cs.origin as cmOrigin, r.origin as robotOrigin, d.origin as sqdOrigin ,

len(cs.origin) as cmOriginLen, len(r.origin) as robotOriginLen, len(d.origin) as sqdOriginLen ,

```
*
from CM_COMPUTER_SYSTEM cs
inner join CM_NIMBUS_ROBOT r
on cs.ip = r.ip
and cs.origin = r.origin
inner join S_QOS_DATA d
on cs.ip = d.host
where d.origin <> r.origin

-- query to see if origins matchup between CM_COMPUTER_SYSTEM and CM_NIMBUS_ROBOT

select cs.origin as cmOrigin, r.origin as robotOrigin, len(r.origin) as robotOriginLen, len(cs.origin) as cmOriginLen ,* from CM_COMPUTER_SYSTEM cs inner join CM_NIMBUS_ROBOT r on cs.ip = r.ip where cs.origin <> r.origin

-- query to update origins where S_QOS_DATA doesn't match the others

begin transaction
update d set d.origin = r.origin
from S_QOS_DATA d
inner join CM_COMPUTER_SYSTEM cs
on cs.ip = d.host
inner join CM_NIMBUS_ROBOT r
on cs.ip = r.ip
and cs.origin = r.origin
where d.origin <> r.origin
commit transaction

-- looking for the device from S_QOS_DATA

select * from S_QOS_DATA d
left join CM_CONFIGURATION_ITEM_METRIC m
```

```
on d.ci_metric_id = m.ci_metric_id
left join CM_CONFIGURATION_ITEM i
on m.ci_id = i.ci_id
left join CM_DEVICE c
on i.dev_id = c.dev_id
where d.probe = 'cdm' and
d.robot = ''

-- device info

select * from CM_DEVICE d
where d.cs_id = ''
or d.dev_id = ''

-- looking for the device from CM_COMPUTER_SYSTEM

select * From CM_COMPUTER_SYSTEM s
left join CM_GROUP_MEMBER cm
on s.cs_id = cm.cs_id
left join CM_GROUP cg
on cg.grp_id = cm.grp_id
where s.ip = ''
or s.name = ''
or s.cs_id = ''

-- all left joins to see where things break down

select distinct
c.dev_id,
r.address nimbus_address,
r.ip robotip,
r.domain,
```

```
r.hub hubname,
s.name robotname,
cg.name groupname,
s.nimbus_type,
d.source source,
d.origin,
s.os_type os_major,
s.os_name os_minor,
s.os_version,
s.os_description,
d.ci_metric_id,
d.qos,
d.target,
d.r_table,
d.probe,
d.table_id,
d.samplevalue value
from S_QOS_DATA d
left join CM_CONFIGURATION_ITEM_METRIC m
on m.ci_metric_id=d.ci_metric_id
left join CM_CONFIGURATION_ITEM i
on i.ci_id = m.ci_id
left join CM_DEVICE c
on c.dev_id = i.dev_id
left join CM_COMPUTER_SYSTEM s
on c.cs_id = s.cs_id
left join CM_GROUP_MEMBER cm
on c.cs_id = cm.cs_id
left join CM_GROUP cg
```

```
on cg.grp_id = cm.grp_id
left join CM_NIMBUS_ROBOT r
on s.ip = r.ip and
r.origin = d.origin
where
d.probe = 'cdm'
-- and d.robot = ''
-- and d.origin = ''
-- query used by dynamic views to build the tree nodes
select distinct
c.dev_id,
r.address nimbus_address,
r.ip robotip,
r.domain,
r.hub hubname,
s.name robotname,
cg.name groupname,
s.nimbus_type,
d.source source,
d.origin,
s.os_type os_major,
s.os_name os_minor,
s.os_version,
s.os_description,
d.ci_metric_id,
d.qos,
d.target,
d.r_table,
d.probe,
```

```
d.table_id,
d.samplevalue value
from S_QOS_DATA d,
CM_CONFIGURATION_ITEM_METRIC m,
CM_CONFIGURATION_ITEM i,
CM_DEVICE c,
CM_COMPUTER_SYSTEM s,
CM_GROUP_MEMBER cm,
CM_GROUP cg,
CM_NIMBUS_ROBOT r
where
m.ci_metric_id=d.ci_metric_id and
i.ci_id = m.ci_id and
c.dev_id = i.dev_id and
c.cs_id = s.cs_id and
c.cs_id = cm.cs_id and
cg.grp_id = cm.grp_id and
s.ip = r.ip and
r.origin = d.origin and
d.probe = 'cdm'
UNION
select distinct
c.dev_id,
r.address nimbus_address,
r.ip robotip,
r.domain,
r.hub hubname,
s.name robotname,
cg.name groupname,
```

```
s.nimbus_type,
d.source source,
d.origin,
s.os_type os_major,
s.os_name os_minor,
s.os_version,
s.os_description,
d.ci_metric_id,
d.qos,
d.target,
d.r_table,
d.probe,
d.table_id,
d.samplevalue value
from S_QOS_DATA d,
CM_CONFIGURATION_ITEM_METRIC m,
CM_CONFIGURATION_ITEM i,
CM_DEVICE c,
CM_COMPUTER_SYSTEM s,
CM_GROUP_MEMBER cm,
CM_GROUP cg,
CM_NIMBUS_ROBOT r
where
m.ci_metric_id=d.ci_metric_id and
i.ci_id = m.ci_id and
c.dev_id = i.dev_id and
c.cs_id = s.cs_id and
c.cs_id = cm.cs_id and
cg.grp_id = cm.grp_id and
```

```
r.origin = d.origin and
d.probe = 'RSP'
```

Find Most Costly Unused Indexes

These are the indexes which are not useful in any query execution and consuming space on the disk unnecessarily.

Query to find unused Indexes:

```
SELECT owner,
    index_name
FROM   dba_indexes
WHERE owner = 'schema_name'
AND    status NOT IN ('VALID', 'N/A')
ORDER BY owner, index_name;
```

Recommendation:

Analyze all the unused indexes. Delete the unused indexes if they are not required.

Find Top Costly Missing Indexes

This script will be used to find missing indexes:

```
select * from (
select 'the column ' || c.name || ' of the table ' || us.name || '.' || o.name || ' was used ' ||
u.equality_preds || ' times in an equality predicate and ' || u.equijoin_preds || ' times in an equijoin
predicate and is not indexed' as colum_to_index
 from sys.col_usage$ u JOIN
    sys.obj$ o ON u.obj# = o.obj# JOIN
    sys.col$ c ON u.obj# = c.obj# JOIN
    sys.user$ us ON  us.user# = o.owner# and   u.intcol# = c.col#
WHERE
 us.name='schema_name'
 and   c.name not in (select column_name from dba_ind_columns where index_owner
='schema_name')
```

```
 and (u.equality_preds > 100 OR u.equijoin_preds > 100)
 order by u.equality_preds+u.equijoin_preds desc)
WHERE rownum <11;
```

Recommendation: Create all missing indexes only if they are required.

Find Tables without Primary Key

Use the below query to find tables with primary key.

```
SELECT table_name
FROM user_tables
WHERE table_name
NOT IN (SELECT table_name FROM user_constraints WHERE constraint_type ='P' ) ORDER BY
table_name;
```

Find Objects with No Indexes

Use below query to find tables with no indexes.

```
SELECT t.TABLE_NAME
 FROM USER_TABLES t
 LEFT OUTER JOIN (SELECT DISTINCT TABLE_NAME
                  FROM USER_INDEXES) i
  ON i.TABLE_NAME = t.TABLE_NAME
 WHERE i.TABLE_NAME IS NULL
 ORDER BY t.TABLE_NAME;
```

Find Top SQL with Highest CPU

Below is the query to find top 20 CPU and Disk intensive queries.

```
SELECT * from
(SELECT sql_text,
cpu_time/1000000 cpu_time,
elapsed_time/1000000 elapsed_time,
disk_reads,
buffer_gets,
rows_processed
```

```
FROM v$sqlarea

ORDER BY cpu_time desc, disk_reads desc

)

WHERE rownum < 21
```

Recommendation: Analyze the queries and create all appropriate indexes on the tables involved in the queries listed above. If all the indexes are in place and still query is taking more CPU then try to increase the number of cores.

Find Top SQL with Highest I/O

Below query finds the top 10 queries information with more number of disk reads.

```
SELECT * FROM (SELECT Substr(a.sql_text,1,50) sql_text,

Trunc(a.disk_reads/Decode(a.executions,0,1,a.executions)) reads_per_exe,

a.buffer_gets, a.disk_reads, a.executions "exe", a.sorts "sorts", a.address "address"

FROM   v$sqlarea a

ORDER BY 2 DESC)

WHERE rownum <= 10;
```

Recommendations:

Analyze the queries and try to decrease the number of disk reads by creating proper Indexes and updating statistics on the tables involved in the queries listed above.

Find Statistics Update Time

To check for table statistics, use the following query.

```
SELECT owner,

table_name,

num_rows,

sample_size,

last_analyzed,

tablespace_name

FROM dba_tables

WHERE owner='<SCHEMANAME>'

ORDER BY owner
```

To check for index statistics, use the following query:

```
SELECT
 index_name,
table_name,
num_rows,
sample_size,
distinct_keys,
last_analyzed,
status
FROM
 user_indexes
WHERE table_owner='<SCHEMANAME>'
ORDER BY
 table_owner;
```

Update Statistics Queries

Query to update the statistics on the table:

```
EXECUTE DBMS_STATS.GATHER_TABLE_STATS('CA_UIM','S_QOS_DATA');
```

Query to update the statistics on the index:

```
EXECUTE dbms_stats.gather_index_stats('CA_UIM', 'S_QOS_DATA_IDX1');
```

Query to update the statistics on the partition:

```
EXECUTE dbms_stats.gather_index_stats('CA_UIM',
'BN_QOS_DATA_0001_IDX1','BN_QOS_DATA_0001$1522454400');
```

Query to check “auto optimizer statistics collection” is on or not:

```
SELECT client_name, status FROM dba_autotask_operation;
```

By default, Automatic Statistics Collection is enabled in Oracle. If it is disabled, you can enable it using the below script:

```
BEGIN
  DBMS_AUTO_TASK_ADMIN.ENABLE(
    client_name => 'auto optimizer stats collection',
```

```
   operation => NULL,
   window_name => NULL);
END;
/
```

Recommendation:

Table and index statistics should be up-to-date. If statistics are not up-to-date, then Optimizer will select wrong query execution plan.

Find Active sessions

Query to find active sessions information

```
SELECT sesion.sid,
        sesion.username,
        optimizer_mode,
        hash_value,
        address,
        cpu_time,
        elapsed_time,
         sql_text
  FROM v$sqlarea sqlarea, v$session sesion
  WHERE sesion.sql_hash_value = sqlarea.hash_value
  AND sesion.sql_address    = sqlarea.address
  AND sesion.username is not null
  AND sesion.status='ACTIVE';
```

Find Top DISK- Intensive Queries

Use the below query to find the Top 20 disk intensive queries

```
SELECT
     sql_text, username, disk_reads_per_exec, buffer_gets, disk_reads, parse_calls, sorts, executions,
     rows_processed, hit_ratio, first_load_time, sharable_mem, persistent_mem,runtime_mem,
     cpu_time, elapsed_time, address,  hash_value
FROM
```

```
(SELECT sql_text ,  b.username , round((a.disk_reads/decode(a.executions,0,1, a.executions)),2)
 disk_reads_per_exec,  a.disk_reads , a.buffer_gets , a.parse_calls , a.sorts , a.executions ,
 a.rows_processed , 100 - round(100 * a.disk_reads/greatest(a.buffer_gets,1),2) hit_ratio,
 a.first_load_time , sharable_mem , persistent_mem ,  runtime_mem, cpu_time, elapsed_time,
 address, hash_value
 FROM
 sys.v_$sqlarea a,
 sys.all_users b
WHERE
a.parsing_user_id=b.user_id AND
b.username in ('schema_name')
ORDER BY 3 desc)
WHERE rownum < 21
```

Recommendation:

Analyze the queries and create the appropriate indexes on the tables on the tables involved in the queries listed above. If all the indexes exist, then use the DISK with fast head movement.

Appendix B: Monitoring the Instance Health with Reports

You can use the set of reports to monitor the Oracle instance server health. These reports monitor the health and give recommendations, if necessary.

Automatic Database Diagnostic Monitor (ADDM)

When a problem occurs in oracle database, proper diagnosis has to be done to know the exact problem. ADDM reports will be used to identify and fix the issues. These reports will generate the root cause of the problem and also gives us the recommendations to the problem.

ADDM reports will use the statistical data stored in Automatic Workload Repository for proper diagnosis of the problem. These reports will analyze the AWR data on regular basis, identifies the root cause and provides the best suitable solutions in the form of recommendations to apply on database. To generate the ADDM reports the **STATISTICAL_LEVEL** parameter has to be set as **TYPICAL** or **ALL**.

The types of problems analyzed by ADMM reports are:

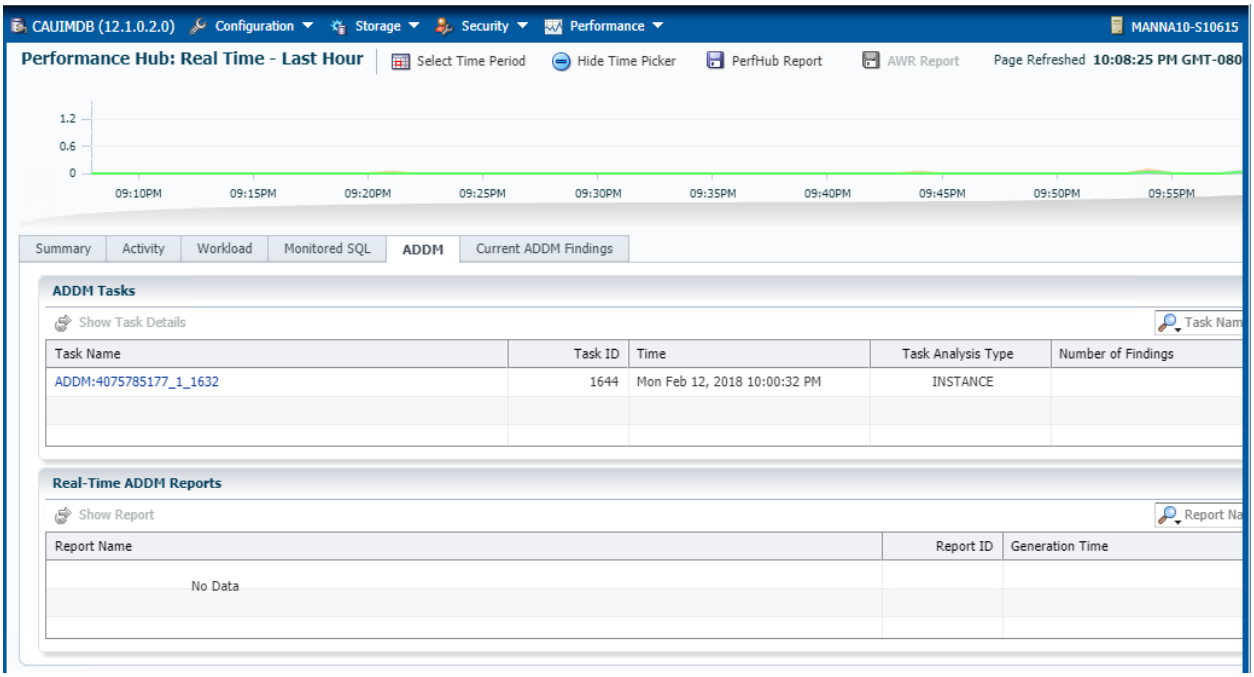
- CPU bottlenecks
- Memory structures
- I/O capacity usage
- High load SQL and PL/SQL.
- Database configuration issues.
- Concurrency issues like buffer busy waits
- Object contention issues.

The types of recommendations in ADDM reports may include:

- Hardware changes
- Database configuration changes
- Schema changes
- Application changes
- Ways to Generate ADDM Reports:

Using Enterprise Manager

To access ADDM reports in Enterprise Manager click on Performance->Performance Hub from home page. You will get the below screen:



Click on ADDM tab and select Task under Task Name to analyze tasks. Select Report name under Real-Time ADDM Report section to see the ADDM report for certain period.

Using addmrpt.sql Script

You can use the **addmrpt.sql** script to create an ADDM report from SQL*Plus. The script is called as follows.

-- UNIX

@ORACLE_HOME/db_home1/rdbms/admin/addmrpt.sql

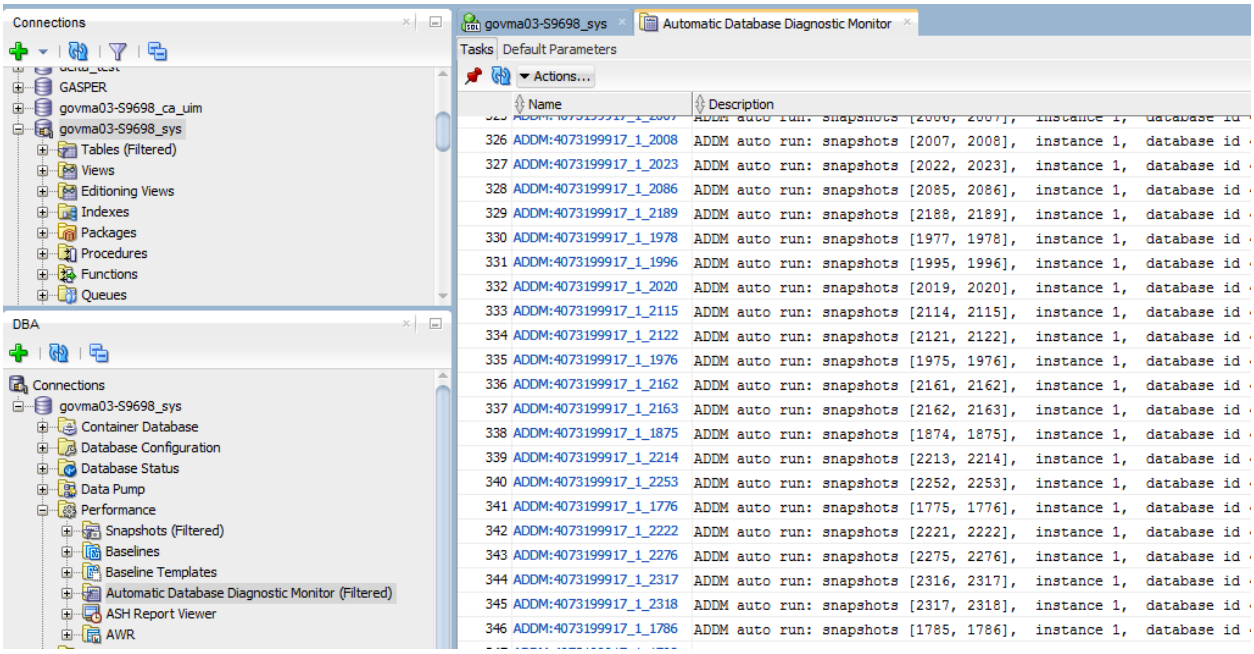
-- Windows

@ORACLE_HOME\db_home1\rdbms\admin\addmrpt.sql

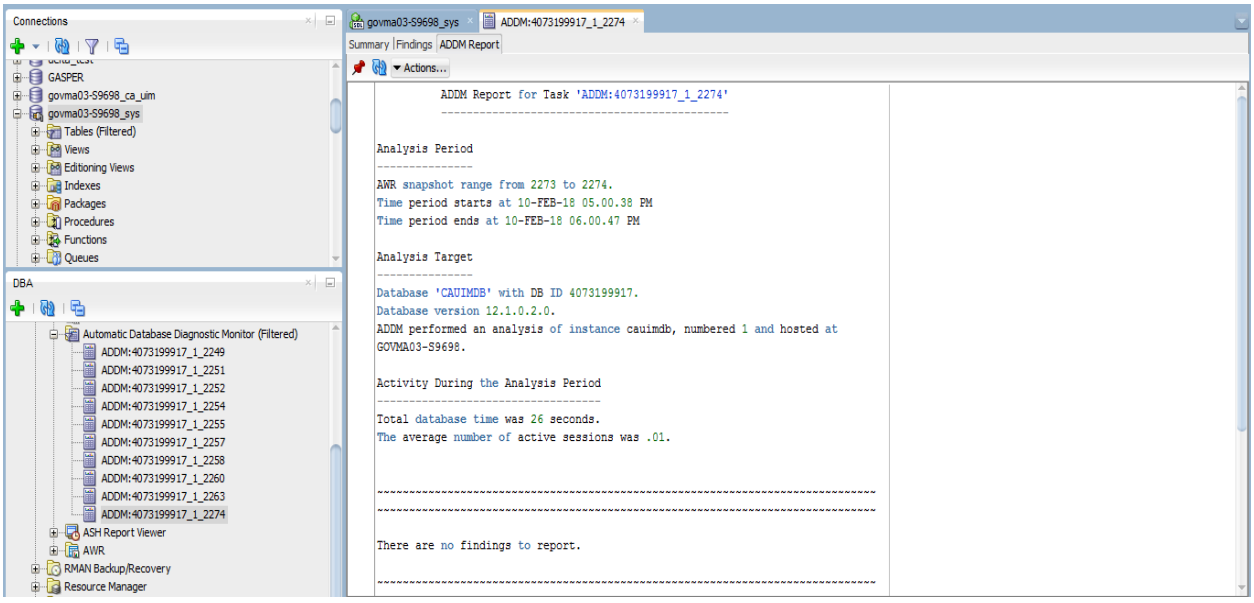
While running the script it will ask for begin and end snapshots to generate the ADDM report.

Using SQL Developer and ADDM Reports

If you are using SQL Developer 4 or later, you can view ADDM reports directly from SQL Developer. To see the reports, select View->DBA. Expand the connection and expand the performance node. Right click on the **Automatic Database Diagnostic Monitor (Filtered)** and click on **open** to see older reports. If you want to select new report click on **run ADDM**.



After running the ADDM to see the report expand Automatic Database Diagnostic Monitor(Filtered) and select the report name and open. It will give you report as shown in below screenshot:



Recommendation: Analyze and apply all the recommendations suggested in the ADDM report as they will read and analyze AWR reports.

Please find the below links to know more about ADDM Reports:

https://docs.oracle.com/database/121/TGDBA/pfgrf_diag.htm#TGDBA02602

Automatic Workload Repository (AWR) Reports

Oracle generates snapshots (sets of historical data) for specific time periods that will be used in identifying performance problems. By default, oracle generates snapshots for every hour and stores the statistics in Automatic Workload Repository (AWR) for 8 days. AWR collects, process and maintain performance statistics for problem detection and tuning purpose.

The AWR statistics include the following:

- Object statistics
- Time usage for activities
- System and session statistics
- SQL Statements that are consuming lot of Memory, I/O and CPU
- Active Session History statistics

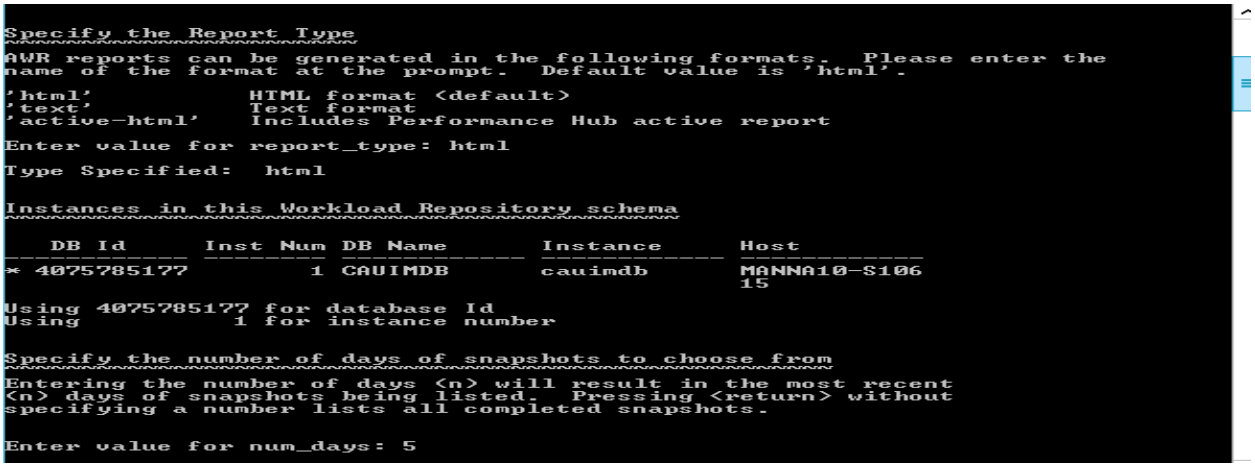
Using Workload Repository Reports Script

Oracle provides two script files to generate AWR reports manually. AWR reports can be generated either in HTML format or Text format. The reports can be generated as follows:

@$ORACLE_HOME/rdbms/admin/awrrpt.sql

@$ORACLE_HOME/rdbms/admin/awrrpti.sql

While running the script, it will prompt for file format, begin snapshot, end snapshot and file name as shown in below screenshots:

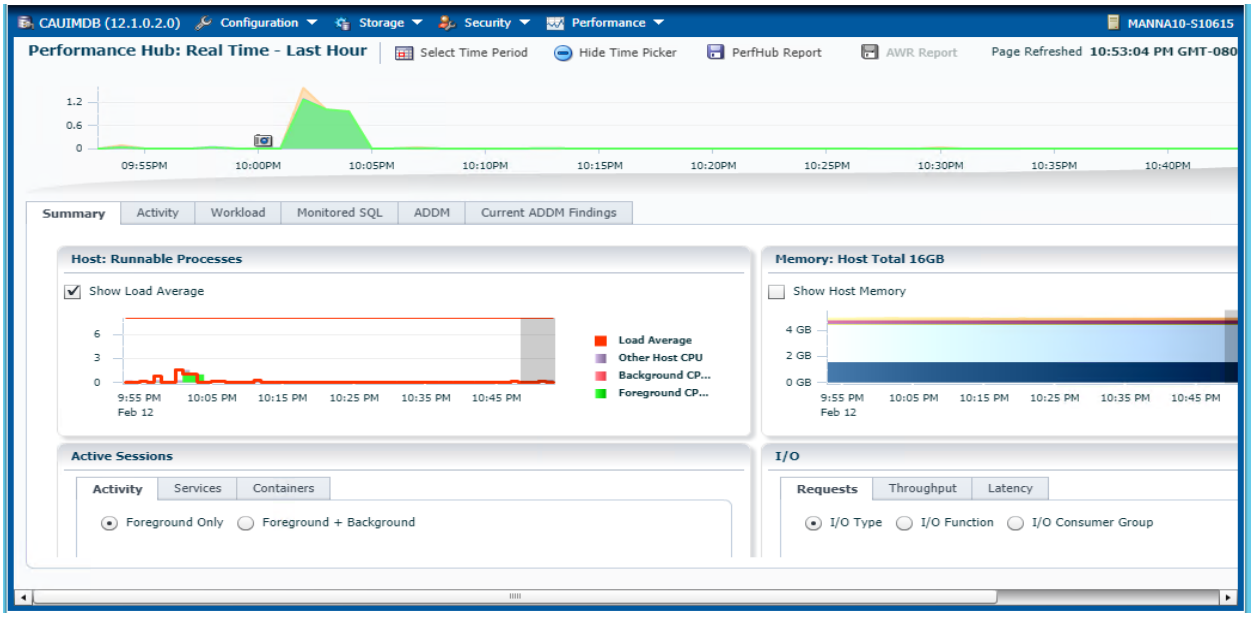


```
                                  1618  12  Feb 2018  08:00       1
                                  1619  12  Feb 2018  09:00       1
                                  1620  12  Feb 2018  10:00       1
                                  1621  12  Feb 2018  11:00       1
                                  1622  12  Feb 2018  12:00       1
                                  1623  12  Feb 2018  13:00       1
                                  1624  12  Feb 2018  14:00       1
                                  1625  12  Feb 2018  15:00       1

                                                                  Snap
Instance      DB Name        Snap Id     Snap Started            Level
------------ ------------ ---------- ------------------- -----
cauimdb       CAUIMDB            1626  12  Feb 2018  16:00       1
                                 1627  12  Feb 2018  17:00       1
                                 1628  12  Feb 2018  18:00       1
                                 1629  12  Feb 2018  19:00       1
                                 1630  12  Feb 2018  20:00       1
                                 1631  12  Feb 2018  21:00       1
                                 1632  12  Feb 2018  22:00       1


Specify the Begin and End Snapshot Ids
Enter value for begin_snap: _
```

It will list all snapshots and its time in 5 days. Select begin and end snapshots and give file name to generate.

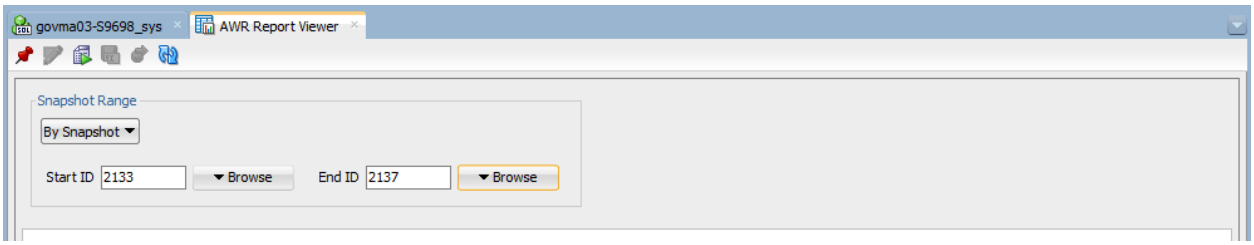
Using Enterprise Manager

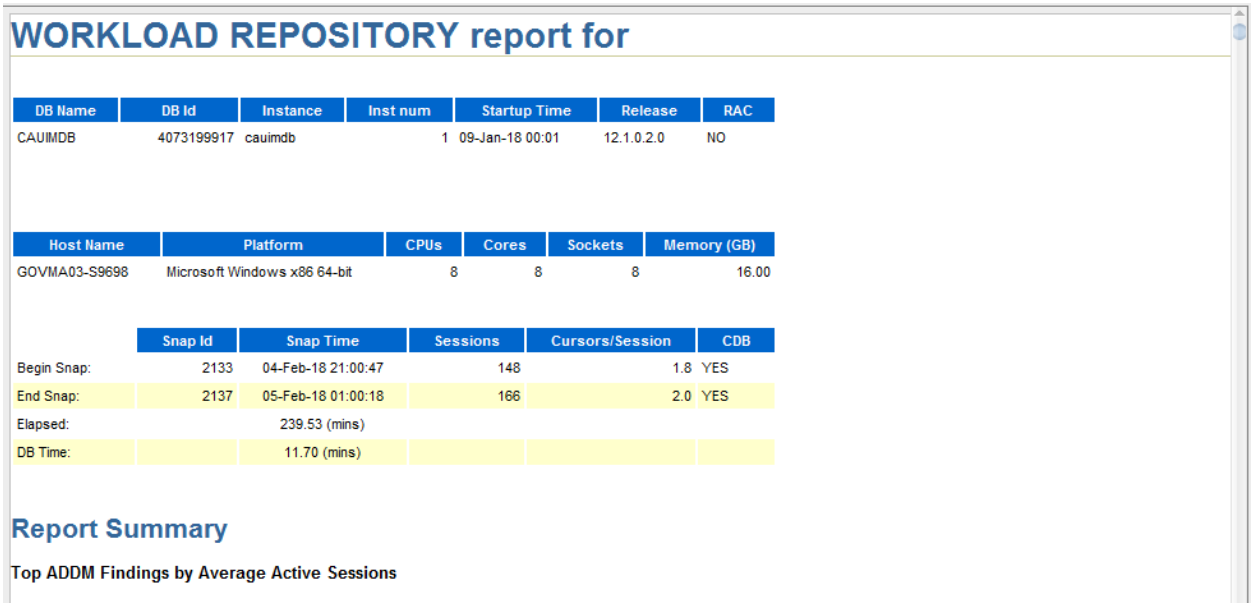
The AWR reports can be generated by Enterprise Manager by clicking Performance Hub under Performance Tab. Click on AWR Report icon to see the AWR report as shown in below screenshot.



Using SQL Developer and AWR Reports

If you are using SQL Developer 4 or later, you can view AWR reports directly from SQL Developer. To see the reports, select **View->DBA**. Expand the connection and expand the performance node. Expand the **AWR** section and click on **AWR Report Viewer**. Select start ID, End ID and click on generate Report as shown in below screenshot.





WORKLOAD REPOSITORY report for

DB Name	DB Id	Instance	Inst num	Startup Time	Release	RAC
CAUIMDB	4073199917	cauimdb	1	09-Jan-18 00:01	12.1.0.2.0	NO

Host Name	Platform	CPUs	Cores	Sockets	Memory (GB)
GOVMA03-S9698	Microsoft Windows x86 64-bit	8	8	8	16.00

	Snap Id	Snap Time	Sessions	Cursors/Session	CDB
Begin Snap:	2133	04-Feb-18 21:00:47	148	1.8	YES
End Snap:	2137	05-Feb-18 01:00:18	166	2.0	YES
Elapsed:		239.53 (mins)			
DB Time:		11.70 (mins)			

Report Summary

Top ADDM Findings by Average Active Sessions

Please find the below reference to know more details about AWR Reports:

https://docs.oracle.com/database/121/TGDBA/gather_stats.htm#TGDBA169

SQL Tuning Advisor

SQL Tuning Advisor will be used to tune and get the recommendations of the query. You can submit one or more SQL statements as input to the tuning advisor and it will give you the recommendations for how to tune the statements, along with a rationale and expected benefit.

Tuning recommendations include:

- Collection of object statistics
- Creation of indexes
- Rewriting SQL statements
- Creation of SQL profiles
- Creation of SQL plan baselines

Using Create Tuning API

To access the SQL tuning advisor API, a user must have the `ADVISOR` privilege.

CONN sys/password AS SYSDBA

GRANT ADVISOR TO user_name;

GRANT ADMINISTER SQL MANAGEMENT OBJECT TO user_name;

CONN user_name/password

CREATE_TUNING_TASK is a function which can be used to create a tuning task for a statement. The statements or query to be analyzed can be retrieved from the Automatic Workload Repository (AWR) and from different statistical views.

Pass the query which you want to analyze in below script:

```
DECLARE
 l_sql             VARCHAR2(500);
 l_sql_tune_task_id  VARCHAR2(100);
BEGIN
 l_sql := 'SQL Query';
 l_sql_tune_task_id := DBMS_SQLTUNE.create_tuning_task (
                  sql_text    => l_sql,
                  bind_list   => sql_binds(anydata.ConvertNumber(100)),
                  user_name   => 'user_name',
                  scope       => DBMS_SQLTUNE.scope_comprehensive,
                  time_limit  => 60,
                  task_name   => 'tuning_task_name',
                  description => 'Tuning task for Advisor.');
 DBMS_OUTPUT.put_line('l_sql_tune_task_id: ' || l_sql_tune_task_id);
END;
/
```

After creating the tuning task by using above script execute the script by using below query:

```
EXEC DBMS_SQLTUNE.execute_tuning_task(task_name => 'tuning_task_name');
```

The status of the tuning task can be monitored using the DBA_ADVISOR_LOG view.

```
SELECT task_name, status FROM dba_advisor_log WHERE owner = 'user_name';

TASK_NAME                      STATUS
-------------------------      -----------
```

tuning_task_name COMPLETED

Once the tuning task has executed successfully the recommendations can be displayed using the **REPORT_TUNING_TASK** function.

SELECT DBMS_SQLTUNE.report_tuning_task('tuning_task_name') AS recommendations FROM dual;

The above query will generate details about the query including Execution Plan, Findings, Recommendations etc.

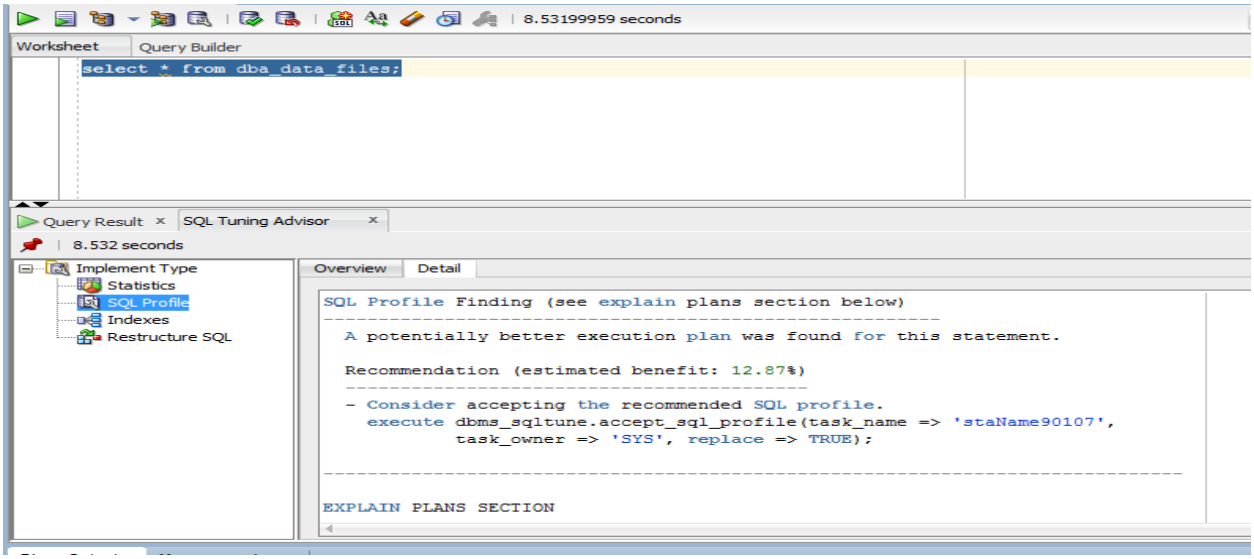
Using SQL Tuning Advisor by SQL Developer

SQL Tuning Advisor performs four types of analysis on the SQL statement. All the recommendations are displayed in the Overview. You can also view each recommendation individually.

Write the SQL query to be analyzed in the SQL Worksheet, and click the **SQL Tuning Advisor** (Ctrl+F12) icon in the SQL Worksheet toolbar.

- The SQL Tuning Advisor task is initiated based on the current statement in the SQL Worksheet.
- After the task completes, the SQL Tuning Advisor page opens up with performance tuning suggestions related to Statistics, SQL Profile, Indexes. For each category, you can see an overview as well as detail information.
- Tuning advisor also reconstruct the query if needed.

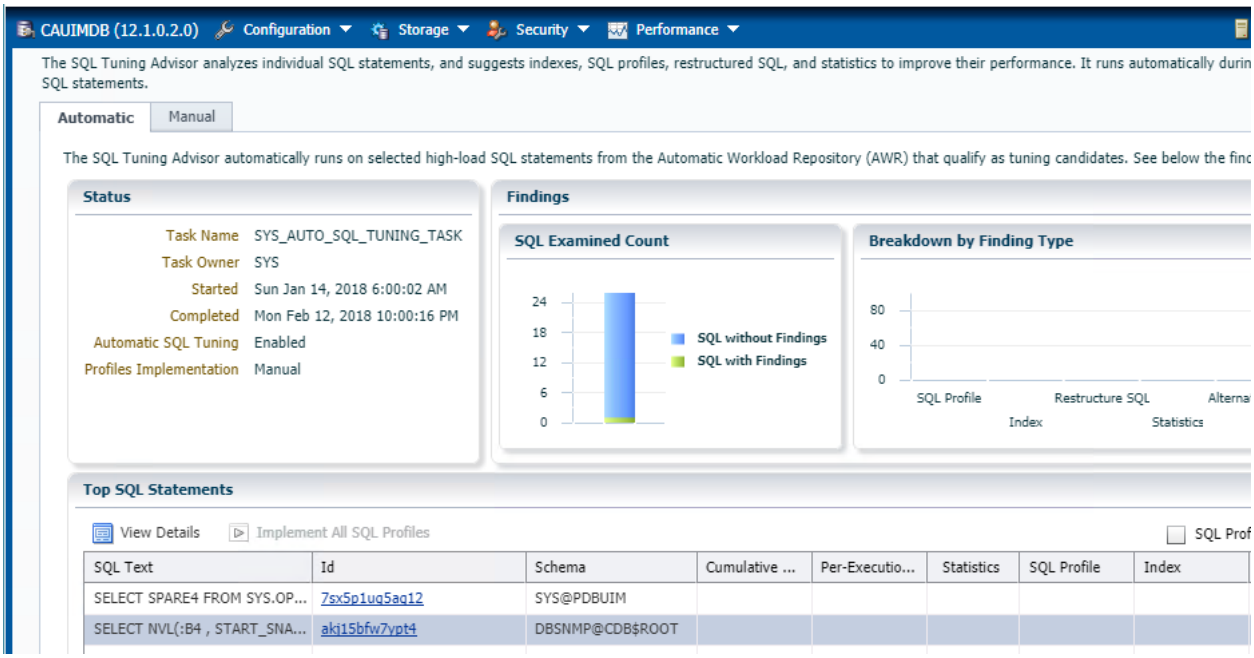
Please find the below screenshot for running tuning advisor:



On each page, you can click the **Run SQL Script (Alt+F11)** button, to implement the recommended changes. Alternatively, you can click the **Open SQL Script Worksheet (Alt+F12)** button, to review the recommendations, and make changes before implementing it.

Using Enterprise Manager

Enterprise Manager also has the SQL tuning Advisor feature. To access the Tuning advisor, click on **Performance Tab** and Select **SQL Tuning Advisor.** This will give the dashboard with all the Top SQL statements, status, execution count etc. Click on individual query under **Top SQL Statements** section to see the recommendations as shown in below screenshot:



It will give Findings and % Benefit with that recommendation. Analyze the recommendation and apply based on requirement.

Please find the below references for more details on SQL Tuning Advisor:

https://docs.oracle.com/en/database/oracle/oracle-database/12.2/arpls/DBMS_SQLTUNE.html#GUID-02C23DDE-6304-49E4-9DFC-EA39D44F943C