Oracle Database Health Check

Overview

An Oracle database health check involves a thorough review of the database's performance, configuration, and security to ensure optimal operation. It typically includes examining system resource usage (CPU, memory, and storage), checking database logs for errors, and verifying backup and recovery processes. Performance metrics such as query response times, index efficiency, and I/O bottlenecks are analyzed to identify potential issues. A health check also assesses database security by reviewing user access, roles, and privileges. Regular checks help detect early signs of failure or inefficiency, ensuring that the database runs smoothly and securely.

Oracle database deployment and design are critical to ensuring the system is robust, scalable, and efficient. The deployment process involves installing the database on physical or virtual servers, configuring the storage, memory, and network settings for optimal performance, and setting up failover options like Oracle RAC or Data Guard for high availability.

Fundamentally, Oracle's architecture follows a multi-tiered approach with the **instance** (memory and background processes), the **database** (physical storage), and **clients** (applications) interacting with it. Key components include tablespaces, data files, redo logs, and control files, which manage data and recovery. Security, backup strategies, and efficient indexing are essential in the design phase to meet business continuity, performance, and security requirements.

The design should also account for scalability, by utilizing Oracle's partitioning, clustering, and load balancing features, while maintaining security through role-based access and encryption.

Improving Oracle Database performance involves several strategies. First, optimizing SQL queries can significantly reduce execution time. Second, proper indexing can enhance data retrieval speed. Third, regular database maintenance, such as statistics collection and defragmentation, ensures optimal performance. Fourth, using Oracle's built-in performance tuning tools, like Automatic Workload Repository (AWR) and SQL Tuning Advisor, helps identify and resolve bottlenecks. Lastly, scaling hardware resources, such as CPU and memory, can provide the necessary power for high-demand operations.

Below are some major categories involve to perform Health check of Oracle Database.

1. Oracle Database Architecture Health Check
2. Oracle Database Engine Health Check
3. Oracle Database Processes Level Health Check

Oracle Database Architecture Health Check

Oracle Database Architecture Health Check ensures optimal performance, much like reviewing the core components of a computer. The database's **memory** (SGA and PGA) must be balanced to handle workloads efficiently. **ASM (Automatic Storage Management)** is used to manage datafiles for efficient disk I/O. Critical components like **Redo Log files** are stored on fast, high-performance storage for quick recovery, while **Archive logs** are placed on separate storage for point-in-time recovery. Regular backups are stored on **tape drives** or other media, isolated from the main database storage, ensuring data protection.

- Check CPU Utilization

Top

```
top - 17:30:56 up 5 min,  1 user,  load average: 0.02, 0.31, 0.20
Tasks: 245 total,   1 running, 244 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.1 sy,  0.0 ni, 99.8 id,  0.0 wa,  0.1 hi,  0.1 si,  0.0 st
MiB Mem :   2624.4 total,   1555.9 free,    517.1 used,    551.5 buff/cache
MiB Swap:   2048.0 total,   2048.0 free,      0.0 used.   1929.5 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   1083 root      20   0  695856  30112  15752 S   0.7   1.1   0:02.58 tuned
   2508 oracle    20   0  141176   5660   4336 S   0.3   0.2   0:00.02 sshd
   2574 oracle    20   0  264228   4524   3644 R   0.3   0.2   0:00.30 top
      1 root      20   0  241336  14028   8904 S   0.0   0.5   0:03.69 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.05 kthreadd
```

cat /proc/stat

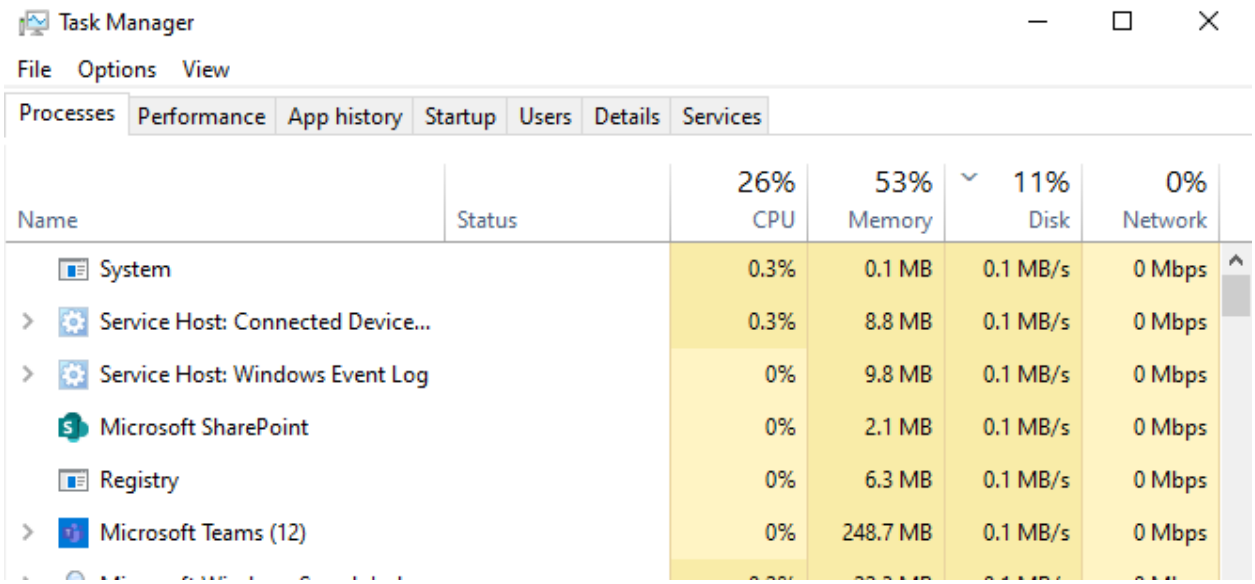
```
[root@dbsingle1 ~]# cat /proc/stat
cpu  3558 0 4379 321611 10133 931 389 0 0 0
cpu0 686 0 1135 81157 1968 268 56 0 0 0
cpu1 1395 0 1355 79745 2508 219 59 0 0 0
cpu2 710 0 1000 79193 3853 219 224 0 0 0
cpu3 765 0 887 81514 1802 224 48 0 0 0
intr 321160 38 9 0 0 0 0 0 0 0 0 0 0 15 0 0 1133 1314 2142 215 818 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 330 238 24 4 0 92 16 48 12 0
17 2579 3848 2149 1883 146 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
ctxt 342335
btime 1727267103
processes 2731
procs_running 1
procs_blocked 0
softirq 172670 1 36146 301 831 3422 0 204 50296 0 81469
```

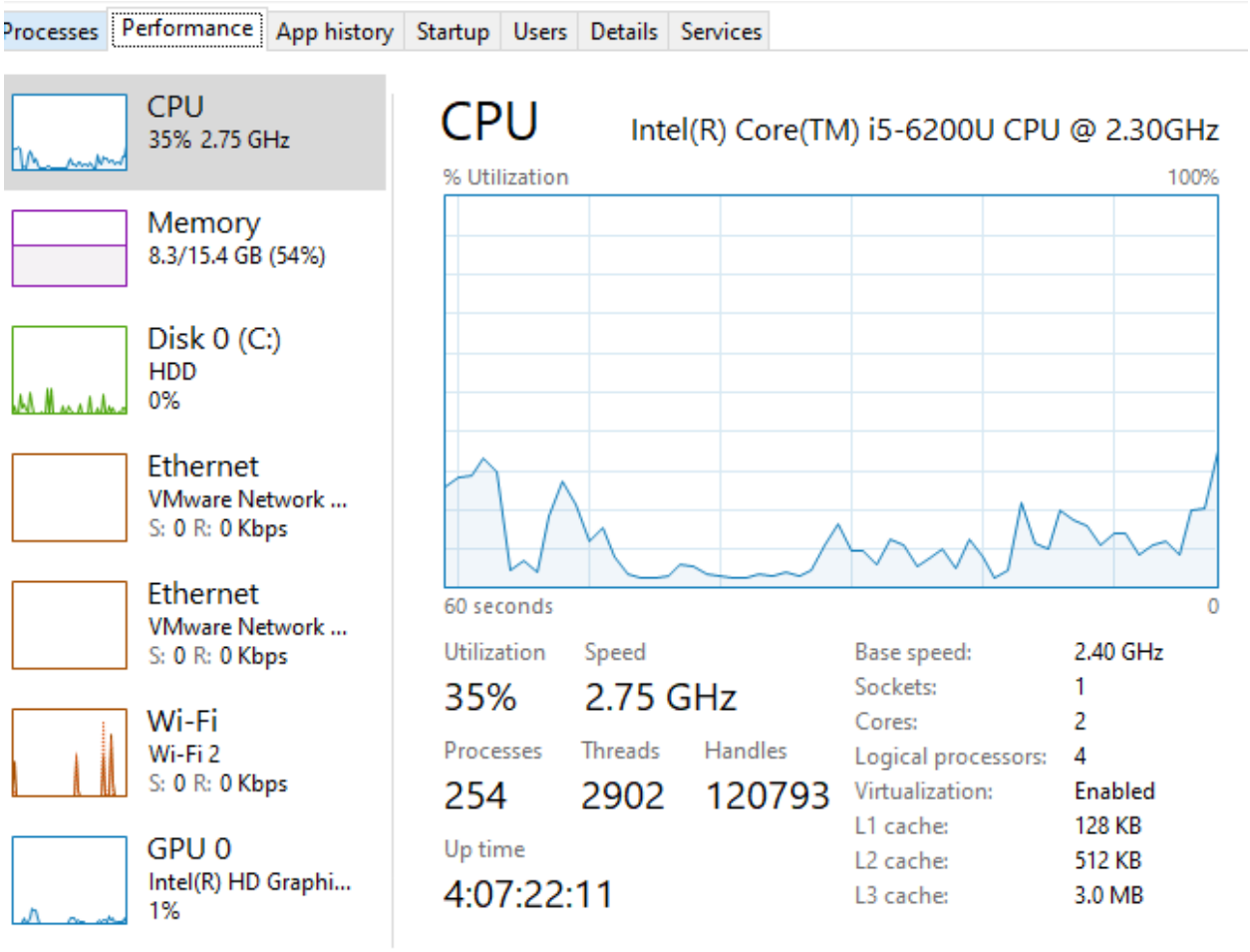
yum install sysstat

mpstat

Oracle Database Health Check

```
05:36:44 PM  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest
  %gnice   %idle
05:36:44 PM  all    1.27    0.00    1.56    3.64    0.32    0.13    0.00    0.00
    0.00   93.09
```



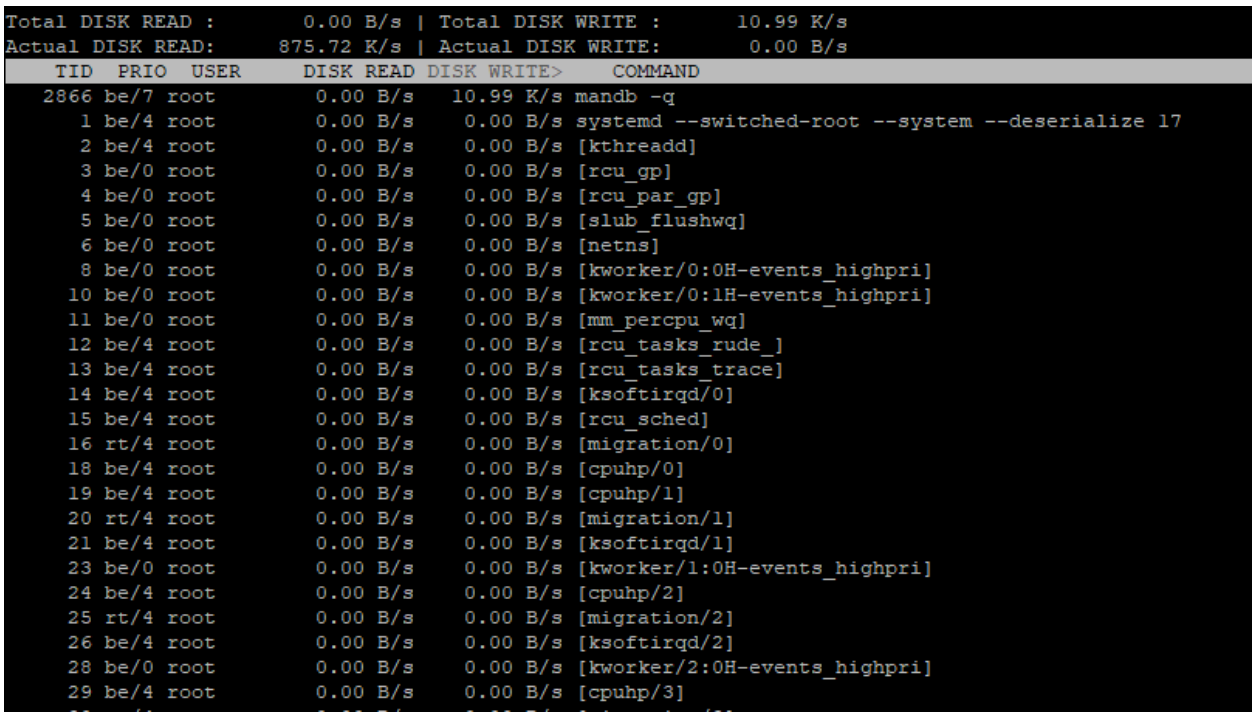


- Check Storage I/O

```
iostat
```

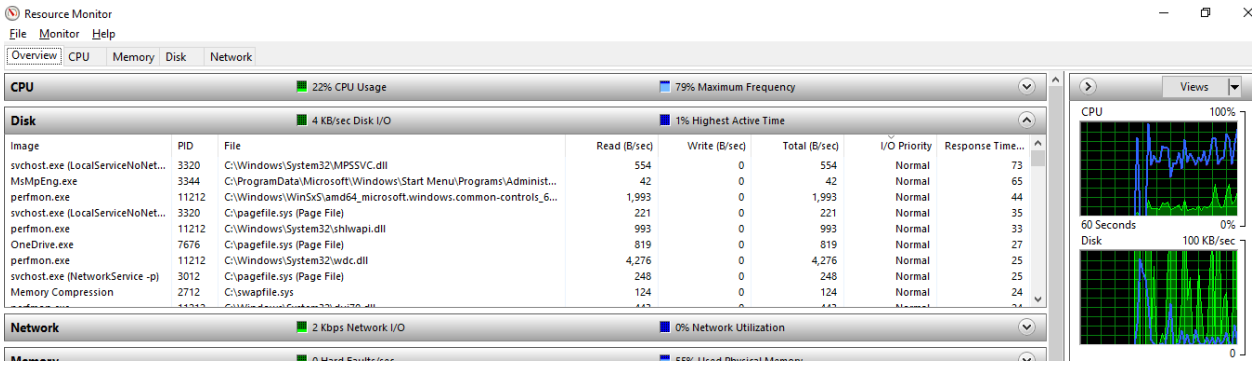
```
[root@dbsingle1 ~]# iostat
Linux 5.15.0-200.131.27.el8uek.x86_64 (dbsingle1)       09/25/2024      _x86_64_        (4 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.65    0.00    1.08    1.81    0.00   96.45

Device             tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
nvme0n1           8.65       503.31        21.14     707083      29701
scd0              0.12         9.75         0.00      13702          0
dm-0              8.19       462.77        19.61     650130      27553
dm-1              0.07         1.58         0.00       2220          0
sdc               0.11         2.87         0.01       4029          8
sdb               5.91        29.35         0.10      41233        144
sdd               0.07         1.68         0.00       2365          4
sda               0.07         1.78         0.00       2500          0

[root@dbsingle1 ~]#
```

yum install iotop
iotop



Using Resource Monitor:

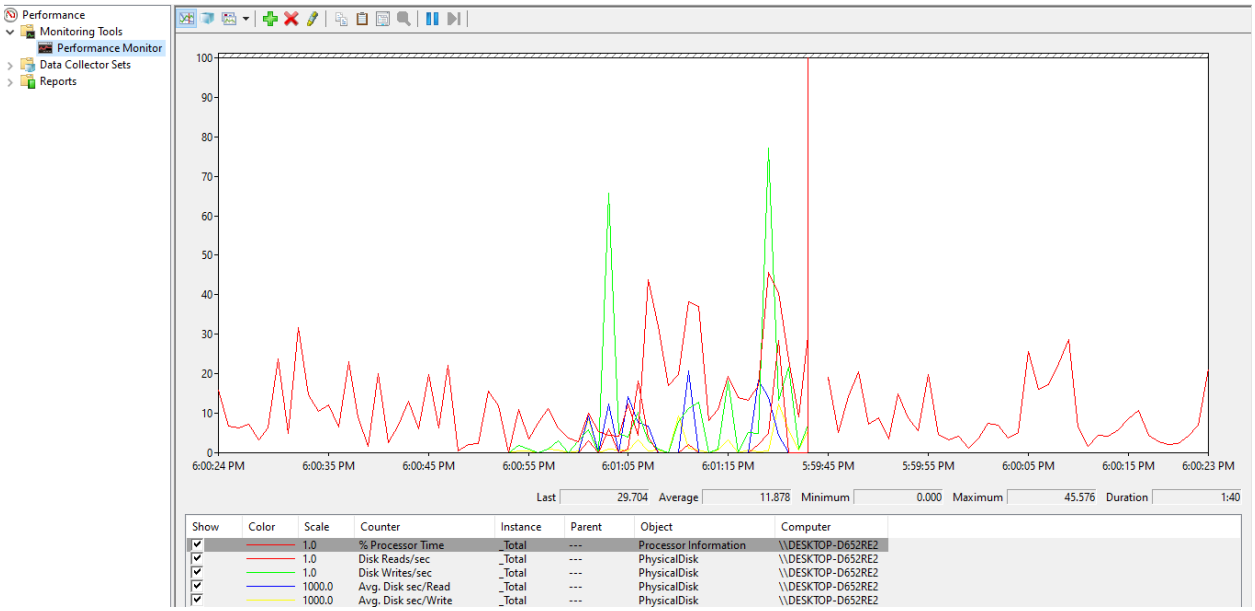
- Press `Ctrl + Shift + Esc` to open **Task Manager**.
- Go to the **Performance** tab and click **Open Resource Monitor** at the bottom.
- Switch to the **Disk** tab in Resource Monitor.
- Here, you can monitor disk I/O statistics, including:
  - **Disk Reads/sec**: Number of read operations per second.
  - **Disk Writes/sec**: Number of write operations per second.
  - **Disk Queue Length**: The length of the queue of I/O operations waiting for disk access.



Using Performance Monitor (PerfMon):

- Press `Windows + R`, type `perfmon`, and press **Enter**.
- In **Performance Monitor**, add counters for disk-related metrics:
  - Right-click on **Performance Monitor** in the left panel.
  - Click **Add Counters**.
  - Choose **PhysicalDisk** and select counters such as:
    - **Disk Reads/sec**
    - **Disk Writes/sec**
    - **Avg. Disk sec/Read**
    - **Avg. Disk sec/Write**
    - **Disk Queue Length**

    These counters give detailed real-time data about I/O performance.



- Check Memory Utilization

  Free -g

```
[oracle@dbsingle1 ~]$ free -g
              total        used        free      shared  buff/cache   available
Mem:              2           0           0           0           1           1
Swap:             1           0           1
[oracle@dbsingle1 ~]$
```

```
cat /proc/meminfo
```

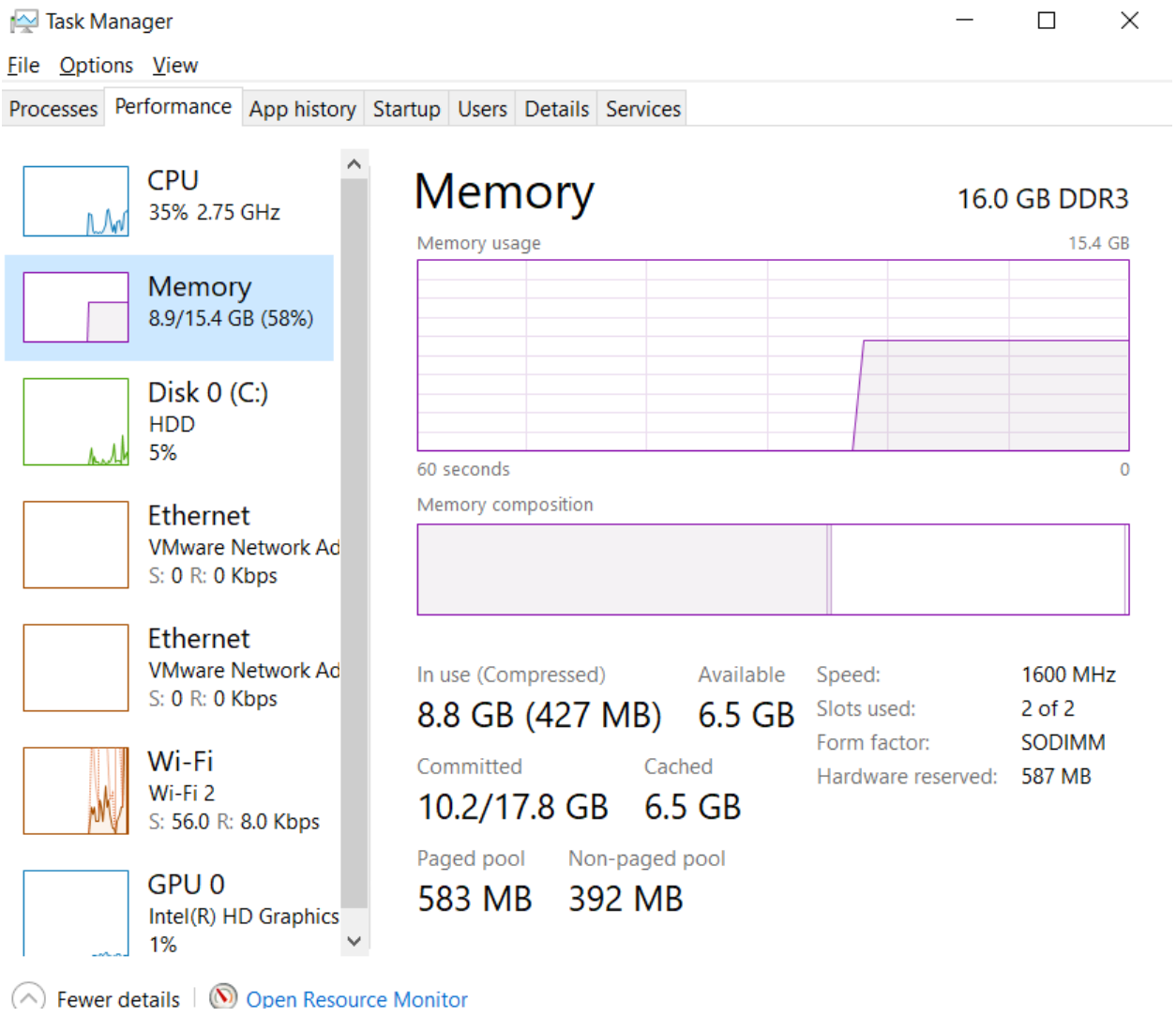
```
[oracle@dbsingle1 ~]$ cat /proc/meminfo
MemTotal:        2687428 kB
MemFree:          830484 kB
MemAvailable:    1817240 kB
Buffers:           47384 kB
Cached:          1044660 kB
SwapCached:            0 kB
Active:           205920 kB
Inactive:        1260872 kB
Active(anon):       2276 kB
Inactive(anon):   394012 kB
Active(file):     203644 kB
Inactive(file):   866860 kB
Unevictable:       12396 kB
Mlocked:           12396 kB
SwapTotal:       2097148 kB
SwapFree:        2097148 kB
Dirty:                 0 kB
Writeback:             0 kB
AnonPages:        386160 kB
Mapped:           192880 kB
Shmem:             10936 kB
KReclaimable:     100664 kB
Slab:             191204 kB
SReclaimable:     100664 kB
SUnreclaim:        90540 kB
KernelStack:        6048 kB
```

vmstat 1

```
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      0 829980  47384 1145364    0    0     0     0   93  156  0  0 100  0  0
 0  0      0 829980  47384 1145364    0    0     0     0  101  177  0  0 100  0  0
 0  0      0 829980  47384 1145364    0    0     0     0   88  154  0  0 100  0  0
 0  0      0 829980  47384 1145364    0    0     0     0   94  169  0  0 100  0  0
 0  0      0 829980  47384 1145364    0    0     0     0   94  163  0  0 100  0  0
 0  0      0 829980  47384 1145364    0    0     0     0  288  289  2  1 97  0  0
 0  0      0 829980  47384 1145364    0    0     0     0   84  147  0  0 100  0  0
 0  0      0 829980  47384 1145364    0    0     0     0   71  135  0  0 100  0  0
 0  0      0 829980  47384 1145364    0    0     0     0   71  136  0  0 100  0  0
 0  0      0 829980  47384 1145364    0    0     0     0   77  145  0  0 100  0  0
 0  0      0 829980  47384 1145364    0    0     0     0   96  177  0  0 100  0  0
 0  0      0 829980  47384 1145364    0    0     0     0  108  200  0  0 100  0  0
```

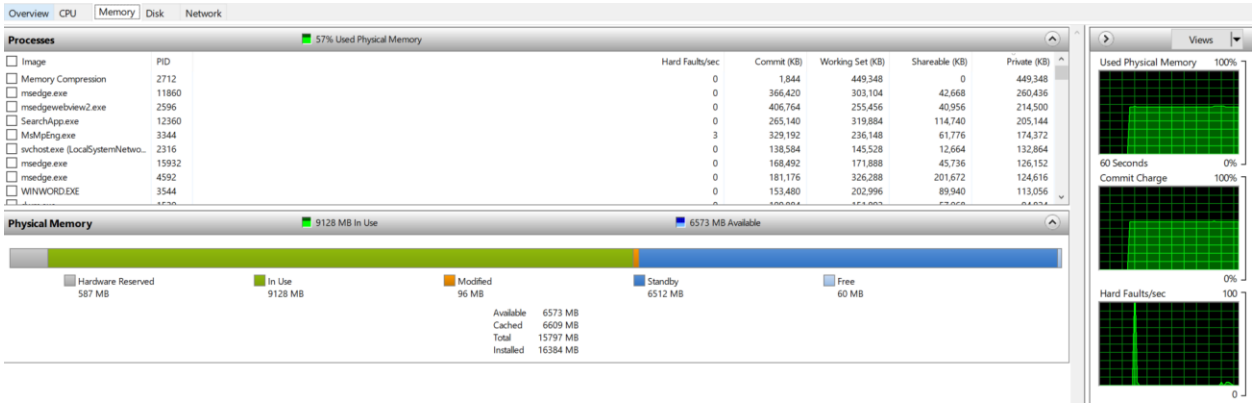
Using Task Manager:

- Press `Ctrl + Shift + Esc` to open **Task Manager**.
- Go to the **Performance** tab, and select **Memory** on the left.



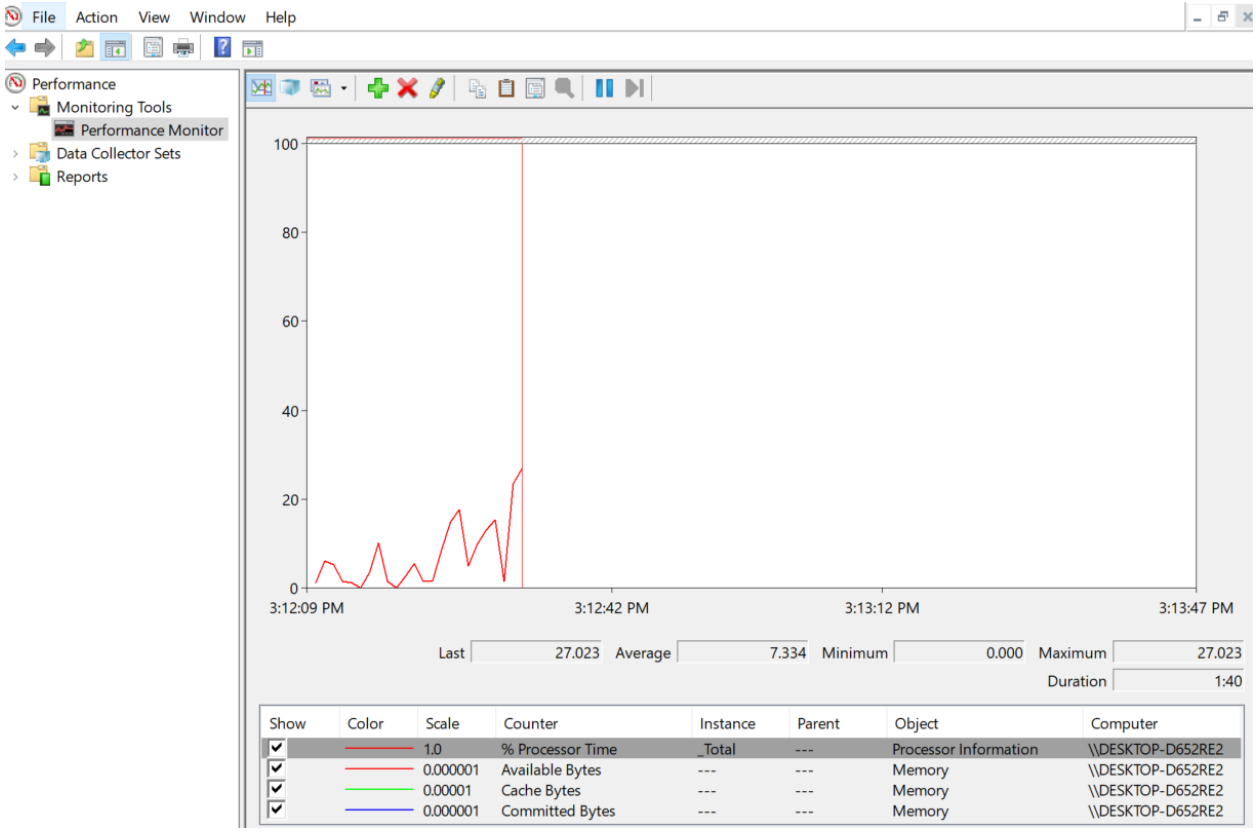
Using Resource Monitor:

- Open Task Manager (`Ctrl + Shift + Esc`), go to the **Performance** tab, and click on **Open Resource Monitor**.
- Switch to the **Memory** tab to get detailed information about memory utilization.



Using Performance Monitor (PerfMon):

- Press `Windows + R`, type `perfmon`, and press **Enter**.
- In **Performance Monitor**, you can add memory-related counters to track usage.
  - Right-click in the window, choose **Add Counters**.
  - Select **Memory**, and choose counters like:
    - **Available MBytes**
    - **Committed Bytes**
    - **Cache Faults/sec**
    - **Page Faults/sec**



- Redo Log File Should be on Fast Storage

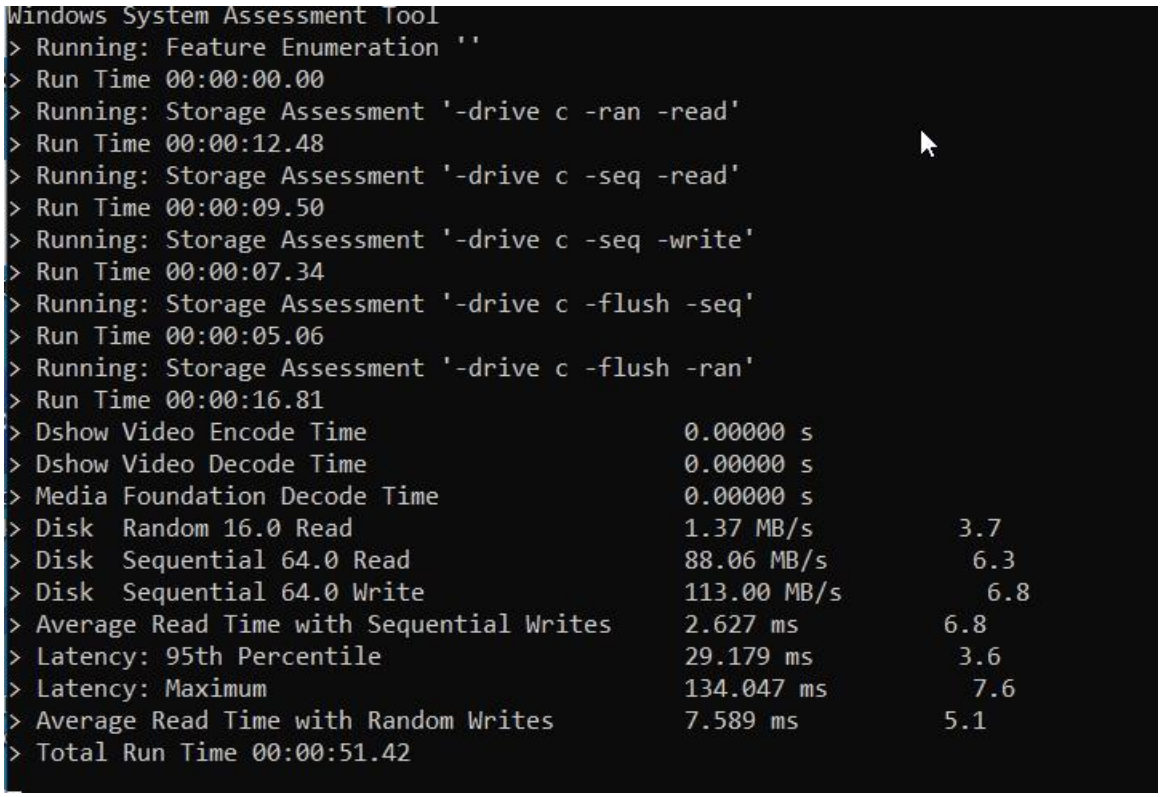
  dd if=/dev/zero of=testfile bs=1G count=1 oflag=direct

```
[root@dbsingle1 u01]# dd if=/dev/zero of=testfile bs=1G count=1 oflag=direct
1+0 records in
1+0 records out
1073741824 bytes (1.1 GB, 1.0 GiB) copied, 5.53857 s, 194 MB/s
```

  dd if=testfile of=/dev/null bs=1G count=1 iflag=direct

```
[root@dbsingle1 u01]# dd if=testfile of=/dev/null bs=1G count=1 iflag=direct
1+0 records in
1+0 records out
1073741824 bytes (1.1 GB, 1.0 GiB) copied, 1.17953 s, 910 MB/s
```

winsat disk -drive c



- Backups Should Be Copy External Storage Like Tape Drive or Other Storage.

Oracle Database Engine Health Check

An Oracle Database Engine Health Check is essential for maintaining optimal performance and involves monitoring several critical components. The System Global Area (SGA) should be appropriately sized to minimize memory contention, while the Program Global Area (PGA) needs careful management to ensure efficient memory allocation for session data and sorting operations. Additionally, the size of the redo logs plays a significant role in recovery performance, and resizing them can prevent frequent log switches that may hinder throughput. Monitoring log switches is crucial, as frequent occurrences can indicate transaction volume issues, and analyzing wait events provides insights into resource contention, helping identify bottlenecks and optimize the database environment effectively.

- **System Global Area (SGA)**: This shared memory area contains data and control information for the Oracle instance, and its size should be tuned to accommodate the workload, minimizing memory contention.

- **Program Global Area (PGA)**: This memory region is allocated to each Oracle process, and monitoring its usage helps ensure efficient memory allocation for sorting, hashing, and managing session data.

- **Redo Log Size**: The size of the redo log files impacts recovery performance; appropriately sizing these logs can enhance database recovery speed and prevent frequent log switches.

- **Log Switches**: Frequent log switches can indicate issues with transaction volume or log size, and monitoring this helps in optimizing the log configuration for better throughput.

Date	Day	Total	Avg per Hour	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23
9/25/2024	Wed	1	0.07													1											
9/24/2024	Tue	5	0.21													2	1			2							
5/24/2024	Fri	3	0.13													1	2										
5/15/2024	Wed	8	0.33											1	1	1	1		4								
4/18/2024	Thu	2	0.08	1																						1	
4/13/2024	Sat	2	0.08												1												1
4/12/2024	Fri	1	0.04																							1	
4/11/2024	Thu	1	0.04																								1
4/10/2024	Wed	1	0.04																			1					
3/30/2024	Sat	1	0.04																	1							
3/29/2024	Fri	44	1.83	14	29										1												
3/28/2024	Thu	70	2.92	50	5																						15
3/27/2024	Wed	135	5.63	4	1								4	24	10	1			1			23	35				32
3/26/2024	Tue	44	1.83												2	29		10	2								1
3/25/2024	Mon	1	0.04											1													
3/19/2024	Tue	1	0.04															1									

- **Wait Events**: Analyzing wait events provides insights into resource contention, helping identify bottlenecks and performance issues, which is essential for tuning and optimizing the database environment.

- **Shared Server and Dedicated Server Process**

  For Configure Shared Server

  select name,value from gv$parameter where name in ('shared_servers','dispatchers','max_shared_servers','shared_server_sessions','max_dispatchers');
  SELECT * FROM V$SHARED_SERVER;
  SELECT * FROM V$DISPATCHER;
  ALTER SYSTEM SET SHARED_SERVERS = 5;
  ALTER SYSTEM SET DISPATCHERS = '(PROTOCOL=TCP)(DISPATCHERS=3)';
  ALTER SYSTEM SET MAX_SHARED_SERVERS = 10;
  ALTER SYSTEM SET SHARED_SERVER_SESSIONS = 100;
  ALTER SYSTEM SET MAX_DISPATCHERS = 5;
  SELECT * FROM V$SHARED_SERVER;
  SELECT * FROM V$DISPATCHER;

- **Monitor Block Tracking File Size which use for Incremental Backup.**

  ALTER DATABASE Disable BLOCK CHANGE TRACKING;
  ALTER DATABASE ENABLE BLOCK CHANGE TRACKING USING FILE '/u01/blocktracking.ora';
  SELECT filename, status, bytes FROM v$block_change_tracking;

- **Configure Keep_Pool and Move Frequently Used Objects in Keep Pool**

  select table_name,BUFFER_POOL from dba_tables where OWNER='T24' AND not BUFFER_POOL='DEFAULT';
  alter TABLE T24.F_BATCH storage (buffer_pool keep);

Oracle Database Processes Execution Level Health Check

An Oracle Database Execution Level Health Check focuses on monitoring and optimizing SQL query performance and execution plans. It involves analyzing SQL statements using tools like **EXPLAIN PLAN**, **SQL Trace**, and **TKPROF** to identify inefficient queries, missing indexes, or suboptimal joins. The check also reviews **bind variable usage**, **index efficiency**, and **parallel execution** settings to enhance performance. By detecting long-running queries or heavy resource-consuming statements, this health check ensures that database operations are executed optimally, improving overall query performance and resource utilization.

Operational Health Check

- **Archive Log Location Space Monitoring**

```
SELECT TRUNC (first_time) "Date", inst_id, TO_CHAR (first_time, 'Dy') "Day",
COUNT (1) "Total",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '00', 1, 0)) "h0",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '01', 1, 0)) "h1",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '02', 1, 0)) "h2",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '03', 1, 0)) "h3",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '04', 1, 0)) "h4",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '05', 1, 0)) "h5",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '06', 1, 0)) "h6",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '07', 1, 0)) "h7",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '08', 1, 0)) "h8",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '09', 1, 0)) "h9",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '10', 1, 0)) "h10",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '11', 1, 0)) "h11",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '12', 1, 0)) "h12",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '13', 1, 0)) "h13",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '14', 1, 0)) "h14",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '15', 1, 0)) "h15",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '16', 1, 0)) "h16",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '17', 1, 0)) "h17",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '18', 1, 0)) "h18",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '19', 1, 0)) "h19",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '20', 1, 0)) "h20",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '21', 1, 0)) "h21",
```

```
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '22', 1, 0)) "h22",
SUM (DECODE (TO_CHAR (first_time, 'hh24'), '23', 1, 0)) "h23",
ROUND (COUNT (1) / 24, 2) "Avg"
FROM gv$log_history
WHERE thread# = inst_id
AND first_time > sysdate -7
GROUP BY TRUNC (first_time), inst_id, TO_CHAR (first_time, 'Dy')
ORDER BY 1,2;
```

- **Monitor Flash Recover Area**

```
select   name,  floor(space_limit / 1024 / 1024/1024) "Size GB", ceil(space_used / 1024 /
1024/1024) "Used GB" from   v$recovery_file_dest order by  name;
```

- **Check Tablespace Size**

```
SELECT ts.tablespace_name,
    size_info.megs_alloc,
    size_info.megs_free/1024,
    size_info.megs_used,
    size_info.pct_free,
    size_info.pct_used,
    size_info.MAX,
    size_info.Available /1024 Available_GB,
    Available_Percentage
  FROM (SELECT a.tablespace_name,
          ROUND (a.bytes_alloc / 1024 / 1024) megs_alloc,
          ROUND (NVL (b.bytes_free, 0) / 1024 / 1024) megs_free,
          ROUND ( (a.bytes_alloc - NVL (b.bytes_free, 0)) / 1024 / 1024)
            megs_used,
          ROUND ( (NVL (b.bytes_free, 0) / a.bytes_alloc) * 100)
            Pct_Free,
          100 - ROUND ( (NVL (b.bytes_free, 0) / a.bytes_alloc) * 100)
            Pct_used,
          ROUND (maxbytes / 1048576) MAX,
         ((ROUND (maxbytes / 1048576))- (ROUND ( (a.bytes_alloc - NVL (b.bytes_free, 0)) / 1024 /
1024)))  as Available
         , (((ROUND (maxbytes / 1048576))- (ROUND ( (a.bytes_alloc - NVL (b.bytes_free, 0)) / 1024 /
1024))) /(ROUND (maxbytes / 1048576)))*100      as Available_Percentage
       FROM (  SELECT f.tablespace_name,
                 SUM (f.bytes) bytes_alloc,
                 SUM (
                   DECODE (f.autoextensible,
                         'YES', f.maxbytes,
```

```
                    'NO', f.bytes))
                maxbytes
           FROM dba_data_files f
         GROUP BY tablespace_name) a,
        (  SELECT f.tablespace_name, SUM (f.bytes) bytes_free
           FROM dba_free_space f
         GROUP BY tablespace_name) b
  WHERE a.tablespace_name = b.tablespace_name(+)
 UNION ALL
  SELECT h.tablespace_name,
         ROUND (SUM (h.bytes_free + h.bytes_used) / 1048576) megs_alloc,
         ROUND (
            SUM (
              (h.bytes_free + h.bytes_used) - NVL (p.bytes_used, 0))
           / 1048576)
           megs_free,
         ROUND (SUM (NVL (p.bytes_used, 0)) / 1048576) megs_used,
         ROUND (
            (  SUM (
                 (h.bytes_free + h.bytes_used)
               - NVL (p.bytes_used, 0))
             / SUM (h.bytes_used + h.bytes_free))
           * 100)
           Pct_Free,
          100
         - ROUND (
              (  SUM (
                   (h.bytes_free + h.bytes_used)
                 - NVL (p.bytes_used, 0))
               / SUM (h.bytes_used + h.bytes_free))
             * 100)
           pct_used,
         ROUND (
           SUM (
              DECODE (f.autoextensible,
                   'YES', f.maxbytes,
                   'NO', f.bytes)
             / 1048576))
           MAX,
          (((ROUND (SUM (DECODE (f.autoextensible,'YES', f.maxbytes,'NO', f.bytes) / 1048576))) -
(ROUND (SUM (NVL (p.bytes_used, 0)) / 1048576)))/1024) as Available_GB
           ,((((ROUND (SUM (DECODE (f.autoextensible,'YES', f.maxbytes,'NO', f.bytes) / 1048576))) -
(ROUND (SUM (NVL (p.bytes_used, 0)) / 1048576)))) /(ROUND (SUM (DECODE (f.autoextensible,'YES',
f.maxbytes,'NO', f.bytes) / 1048576)))) * 100 as Available_Percentage
      FROM sys.v_$TEMP_SPACE_HEADER h,
         sys.v_$Temp_extent_pool p,
```

```
            dba_temp_files f
        WHERE     p.file_id(+) = h.file_id
            AND p.tablespace_name(+) = h.tablespace_name
            AND f.file_id = h.file_id
            AND f.tablespace_name = h.tablespace_name
      GROUP BY h.tablespace_name) size_info,
     sys.dba_tablespaces ts,
     sys.dba_tablespace_groups tsg
WHERE     ts.tablespace_name = size_info.tablespace_name
     AND ts.tablespace_name = tsg.tablespace_name(+)
     order by Available_Percentage asc;
```

- **Monitor Database Backups**

```
SELECT  status status,
        object_type object_type,
        input_bytes / 1024 / 1024 / 1024 input_bytes,
        output_bytes / 1024 / 1024 / 1024 output_bytes,
        TO_CHAR (start_time, 'mm/dd/yyyy:hh:mi:ss') AS start_time,
        TO_CHAR (end_time, 'mm/dd/yyyy:hh:mi:ss') AS end_time
    FROM v$rman_status
   WHERE start_time > SYSDATE - 1 AND operation = 'BACKUP';
```

- **Monitor Database Alert Log**

Performance Related Health Check

Check Database Session Locking

```
select  blocking_session, sid, serial#, wait_class,   seconds_in_wait
from
   v$session where blocking_session is not NULL order by blocking_session;
select username, sid,serial#,sql_id , wait_time from v$session where (sid ) in
(
select blocking_session from v$session
);
select inst_id,last_call_et,username,machine,event,status from gv$session where username='T24'
and lower(event) like '%contention%';
select * from gv$session a , gv$lock b where a.sid=b.sid and A.INST_ID=b.inst_id and
username='T24' and lower(event) like '%contention%';

select * from dba_blockers;
```

Check the Queries which last_call_et column value high and increasing in your Database

```
select event,last_call_et,username,module,program,machine,sql_id,prev_sql_id,
((select max(sql_text) from gv$sql a where a.sql_id=s.sql_id)) as sql1,
((select max(sql_text) from gv$sql a where a.sql_id=s.prev_sql_id)) as sql2
from gv$session s where status='ACTIVE' and not username is null order by last_call_et desc;
```

Check Active / In Active Database Sessions

```
select inst_id,sid,last_call_et,username,machine,event,status from gv$session where
username='T24' order by last_call_et desc;
select inst_id,sid,last_call_et,username,machine,event,status from gv$session where
username='T24' and status='ACTIVE' order by last_call_et desc;
```

Check User Level Session Locking.

```
SELECT se.inst_id,se.last_call_et,
lk.SID, se.username, se.OSUser, se.Machine,DECODE (lk.TYPE,'TX', 'Transaction','TM',
'DML','UL', 'PL/SQL User Lock',lk.TYPE)
lock_type, DECODE (lk.lmode, 0, 'None', 1, 'Null',
2, 'Row-S (SS)',
3, 'Row-X (SX)',
4, 'Share',
5, 'S/Row-X (SSX)',
6, 'xclusive',
TO_CHAR (lk.lmode))
mode_held,
DECODE (lk.request,
0, 'None',
1, 'Null',
2, 'Row-S (SS)',
3, 'Row-X (SX)',
4, 'Share',
5, 'S/Row-X (SSX)',
6, 'Exclusive',
TO_CHAR (lk.request))
mode_requested,
TO_CHAR (lk.id1) lock_id1,
```

```
TO_CHAR (lk.id2) lock_id2,
ob.owner,
ob.object_type,
ob.object_name,
DECODE (lk.Block, 0, 'No', 1, 'Yes', 2, 'Global') block,
se.lockwait
FROM GV$lock lk, dba_objects ob, GV$session se
WHERE lk.TYPE IN ('TX', 'TM', 'UL')
AND lk.SID = se.SID
AND lk.id1 = ob.object_id(+)
AND lk.inst_id = se.inst_id
and last_call_et > 100;
```

Generate AWR Report

Check "DB Time"

	Snap Id	Snap Time	Sessions	Cursors/Session	Instances
Begin Snap:	47761	20-Dec-23 14:00:13	408	15.9	2
End Snap:	47762	20-Dec-23 15:00:18	399	16.3	2
Elapsed:		60.08 (mins)			
DB Time:		292.35 (mins)			

Check "SQL ordered by Elapsed Time"

SQL ordered by Elapsed Time

- Resources reported for PL/SQL code includes the resources used by all SQL statements called by the code.
- % Total DB Time is the Elapsed Time of the SQL statement divided into the Total Database Time multiplied by 100
- %Total - Elapsed Time as a percentage of Total DB time
- %CPU - CPU Time as a percentage of Elapsed Time
- %IO - User I/O Time as a percentage of Elapsed Time
- Captured SQL account for 93.5% of Total DB Time (s): 17,541
- Captured PL/SQL account for 0.4% of Total DB Time (s): 17,541

Elapsed Time (s)	Executions	Elapsed Time per Exec (s)	%Total	%CPU	%IO	SQL Id	SQL Module	SQL Text
2,740.68	12,758	0.21	15.62	22.47	71.79	3gd400tz9ra7b	CM-ESTMT	SELECT IMD.D1_TO_DTTM as D1_TO...
2,534.60	185	13.70	14.45	0.01	0.00	1vtwyftsq8m5w	D1-IMD	UPDATE CI_BATCH_CTRL SET LAST_...
2,288.13	317	7.22	13.04	83.72	1.76	65xtjmhy3bk76	CILQTDSP	select todoentry_0_.TD_PRIORIT...
2,137.54	45,674	0.05	12.19	47.51	5.38	8fp2704uvcjyz	CM-ESTMT	select f1_gtt_col1_dttm F1_GTT...
1,725.57	87	19.83	9.84	93.77	0.00	f92xg1zmjaqzn	CILQTDSP	SELECT MIN(td.TD_PRIORITY_FLG)...
1.334.52	48.593.913	0.00	7.61	27.93	0.00	9kv7b6maf4cmc	D1-IMD	insert into F1 GENERIC GTT (F1...

Check Top 10 Foreground Events by Total Wait Time

Top 10 Foreground Events by Total Wait Time

Event	Waits	Total Wait Time (sec)	Avg Wait	% DB time	Wait Class
DB CPU		10.7K		61.3	
enq: TX - row lock contention	259	2545.6	9.83 s	14.5	Application
cell single block physical read: pmem cache	17,900,482	2278.9	127.31us	13.0	Other
cell single block physical read: flash cache	5,830,804	1805.8	309.71us	10.3	User I/O

Wait Classes by Total Wait Time

Wait Classes by Total Wait Time

Wait Class	Waits	Total Wait Time (sec)	Avg Wait Time	% DB time	Avg Active Sessions
DB CPU		10,745		61.3	3.0
User I/O	9,398,732	2,760	293.63us	15.7	0.8
Application	4,517	2,554	565.37ms	14.6	0.7
Other	22,984,546	2,413	104.99us	13.8	0.7
Cluster	24,420,322	1,250	51.20us	7.1	0.3

Check Table Stats

```
select max(LAST_ANALYZED),'execute dbms_stats.gather_table_stats(ownname=>' || '''' || 't24' || '''' || ',' || 'tabname=>' || '''' || table_name || '''' || ',DEGREE=> 10, estimate_percent=> 10 );' from dba_tab_statistics where owner='T24' and stale_stats ='YES'
group by table_name order by max(LAST_ANALYZED) asc;
select max(LAST_ANALYZED),'execute dbms_stats.gather_table_stats(ownname=>' || '''' || 't24' || '''' || ',' || 'tabname=>' || '''' || table_name || '''' || ',DEGREE=> 10, estimate_percent=> 10 );' from dba_tab_statistics where owner='T24' and stale_stats ='YES' group by table_name order by max(LAST_ANALYZED) asc;

select owner,table_name,LAST_ANALYZED from dba_tables;
```

Check Invalid Objects

```
select * from dba_ind_partitions where index_owner='T24' and not status='USABLE' ;
select index_name,status from dba_indexes where not status='VALID' and owner='T24';
```

Check Execution Plan for Query

```
explain plan for insert into test1 select * from test1;
select * from table(dbms_xplan.display);
```

Check Usage of Indexes

```
ALTER INDEX abc1 MONITORING USAGE;
SELECT index_name, table_name, monitoring, used FROM V$OBJECT_USAGE;
ALTER INDEX abc1 NO MONITORING USAGE;
```

Check List of Queries Perform Full Scan

```
SELECT sql_id,
      child_number,
      object_owner,
      object_name,
      operation,
      options
FROM   v$sql_plan
WHERE  operation = 'TABLE ACCESS'
AND    options = 'FULL' ;
```