Advanced Customer Services

Report of Findings For

TWR

Prepared For

Prepared by Advanced Customer Services

Author:	Do Duc Thinh
Creation Date:	24 May 2016
Version:	1.0
Data Collection:	From 20-APR-16 01:00 to 22-APR-16 23:30

1. Contents

2. Executive Summary

Oracle Advanced Customer Services (ACS) conducted a technical assessment on the Techcombank TWR database hosted on Windows hosts named dc-card-db01. There were no changes made to the assessed system. This document recommends changes and identifies specific areas that require investigation that is more detailed.

The Engineered Assessment Performance (EAP) is a remote service that examines a customer system to make high-level performance recommendations and identify critical areas requiring immediate attention.
The goal of the technical assessment of the TWR environment, i.e., host, database, and I/O subsystem, was to identify factors that may be negatively affecting system performance and response time. Performance data was collected from 20-APR-16 01:00 to 22-APR-16 23:30.

Category	Information	Additional Comments
Database	Drop redundant indexes	See Redundant index
	Review SQL with high resource usage	See Resource Intensive SQL

Other areas were also identified that can be examined by reviewing the Findings and Recommendations Section in detail.

3. System Reviewed

The following system was reviewed during this engagement:

3.1. Host

Server Name	dc-card-db01
Running DB Instances	1
Purpose (Production, development, Q&A)	Production
Platform	Microsoft Windows x86 64-bit
Model	
Operating System	Windows
O/S Version and Release	Windows Server 2008 R2
Cluster Software / Version	N/A
# CPU	16
Processor / CPU Speed	
Memory	95 GB
Volume Manager / Version	ASM

3.2. Database

Database Name	TWR
Instance Name	twr1
RAC-Configuration	Yes
Machine Name	dc-card-db01
RDBMS Version/Release	11.2.0.4.0
Usage (OLTP, DSS, etc.)	OLTP
File System / raw devices	File system
AIO	Yes
Disk Space (of all db files)	Datafiles: 946 GB Tempfiles: 4.3 GB
Archiving Enabled?	Yes

4. Overview

Unless otherwise noted, all findings are based on data collected from 20-APR-16 01:00 to 22-APR-16 23:30.

4.1. Database Response Time

Within the scope of the RDBMS, Response Time in its simplest form consists of Service Time + Wait Time. Service Time equates to time the request is actively being processed on the CPU, while Wait Time encompasses everything else. Oracle tracks Service Time in views related to system statistics and Wait Time within a set of views collectively known as the Wait Interface. By taking snapshots of the relevant views, deltas can be calculated and analyzed to explain precisely where user response time is being spent.

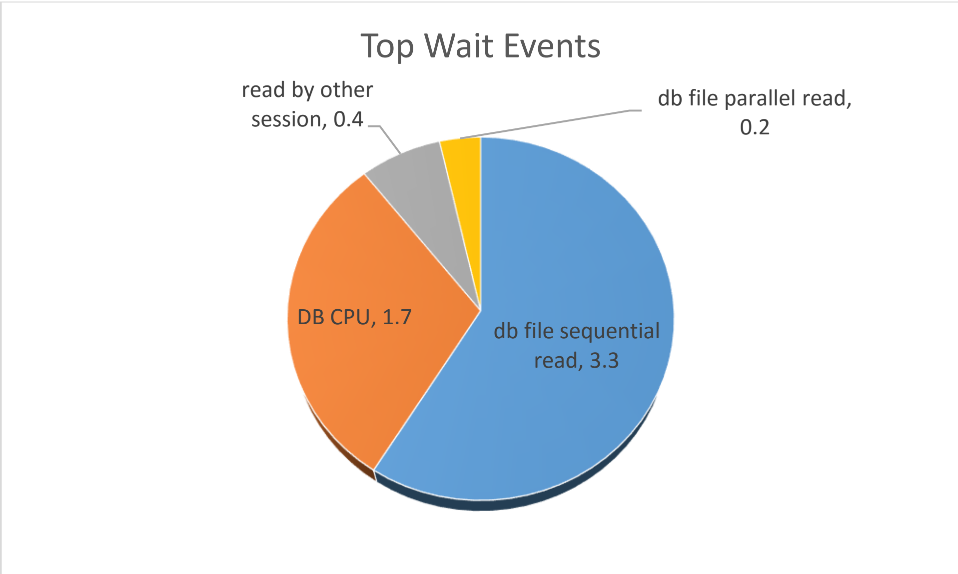

Figure 1: Top Foreground Waits for TWR

4.2. Connection Balance

It is advisable for Oracle RAC databases to have connections load balanced to make optimum utilization of resources. Given below is connection load balance graph for both the instances.

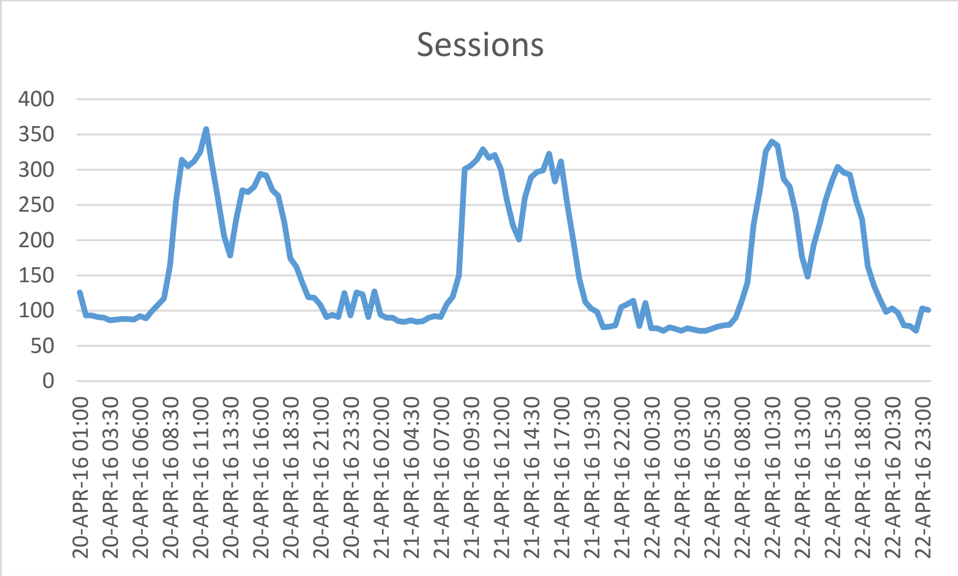

Figure 2: Average Database Sessions for TWR

Observation: As seen from the data, average number of session is evenly distributed on both the instances. No connection spikes or logon/logoff storms are observed. This indicates that connections are successfully load balanced across all nodes in the cluster.

4.3. Average Active Session

The number of active session show how many users are waiting for Oracle to process it's task. The higher, the more load are put on database server.

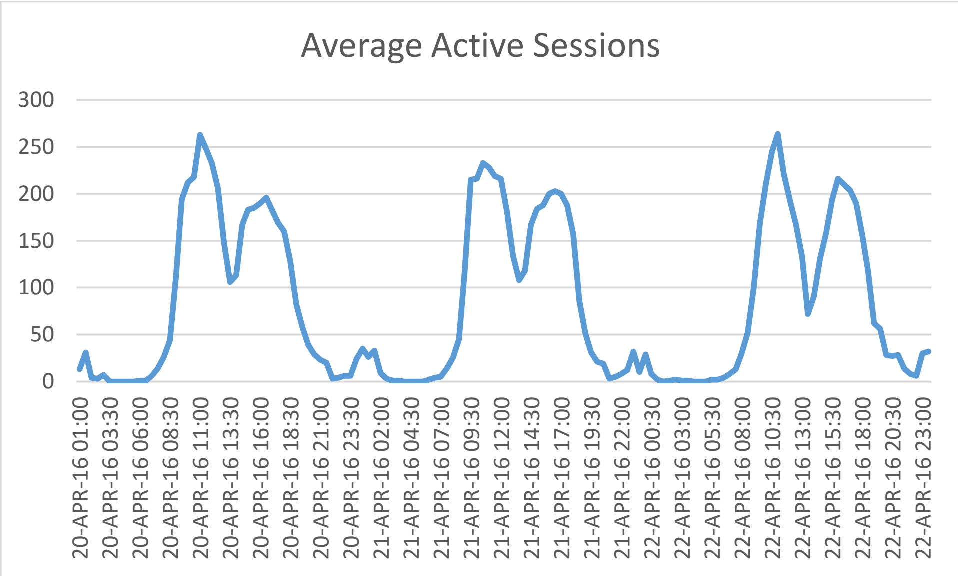

Figure 3: Average Active Sessions for TWR

Observation: Average active session is within system capacity

4.4. CPU Utilization

CPU capacity is a critical resource that should remain below a sustained rate of seventy percent at nearly all times. Whenever CPU utilization is over this amount, response time and throughput suffer, particularly if the saturation is sustained. This system was at all times well below seventy percent.

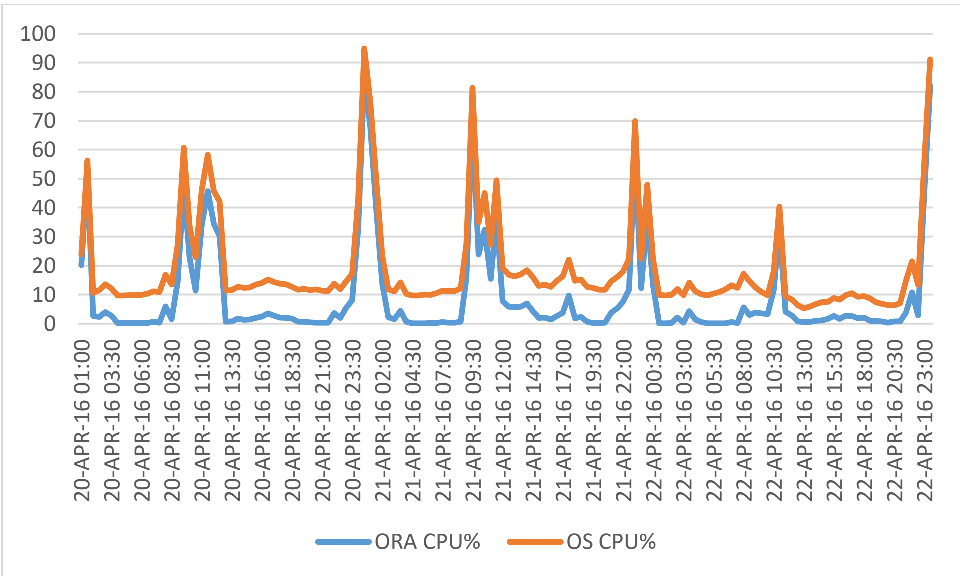

Figure 4: CPU Break Down for TWR

Observation:

The CPU utilization is well under server capacity.

4.5. Disk IO Activity

Physical Disk IO involves the transfer of data to or from the physical hardware. If a disk is more than 60% busy over sustained periods of time, this can indicate overuse of that resource.

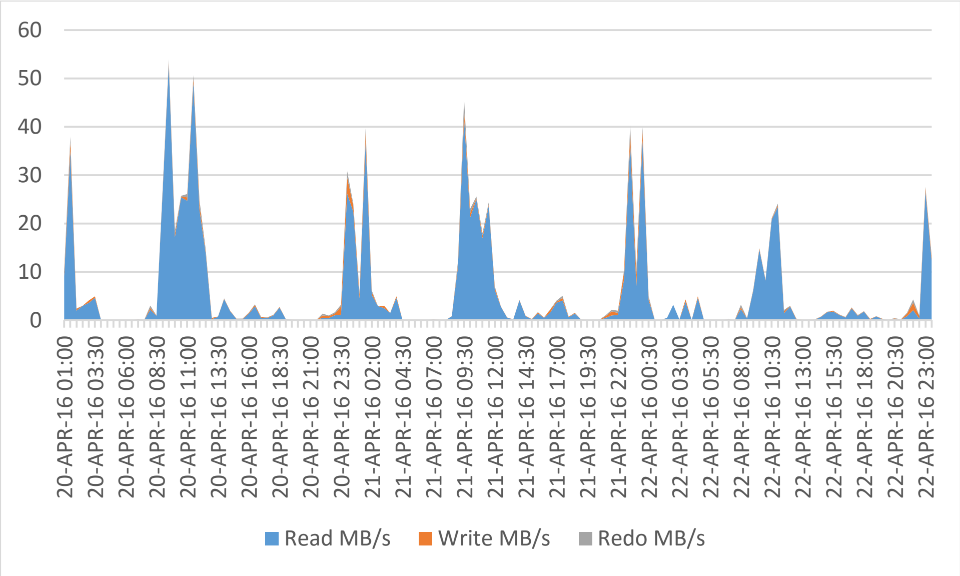

Figure 5: Disk IO for TWR

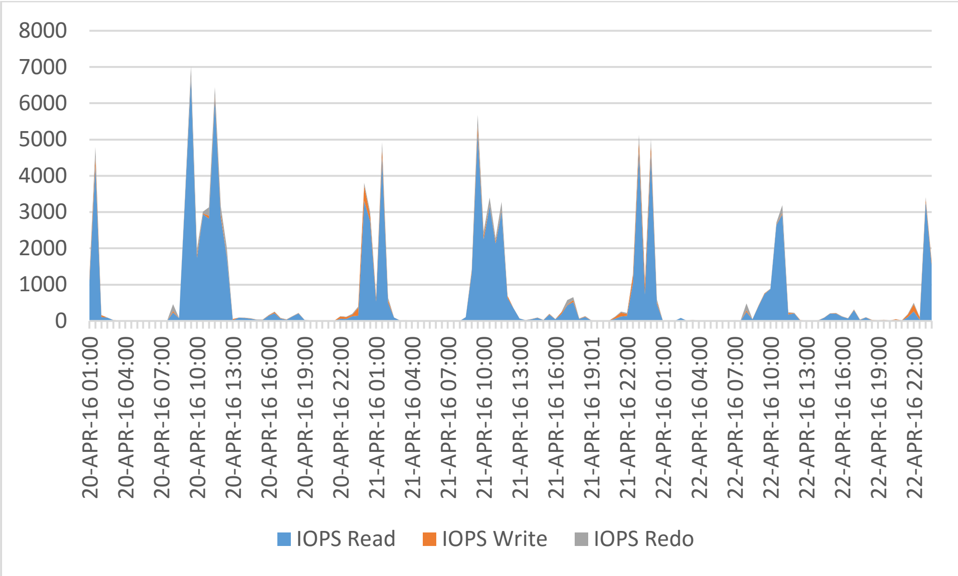

Figure 6: Read , Write Operations per Second (IOPS) for TWR

4.6. Logical IO

Oracle Logical I/O is defined as whenever the Oracle kernel requests access to an Oracle block in the database buffer cache. If the kernel cannot find a specified Oracle block in the database buffer cache, then the Logical I/O causes physical I/O. Because of this, Logical Reads is a better measurement of internal database activity than Physical Reads. Additionally, Logical Reads do require resources and affect response time to a much greater extent than once was thought.

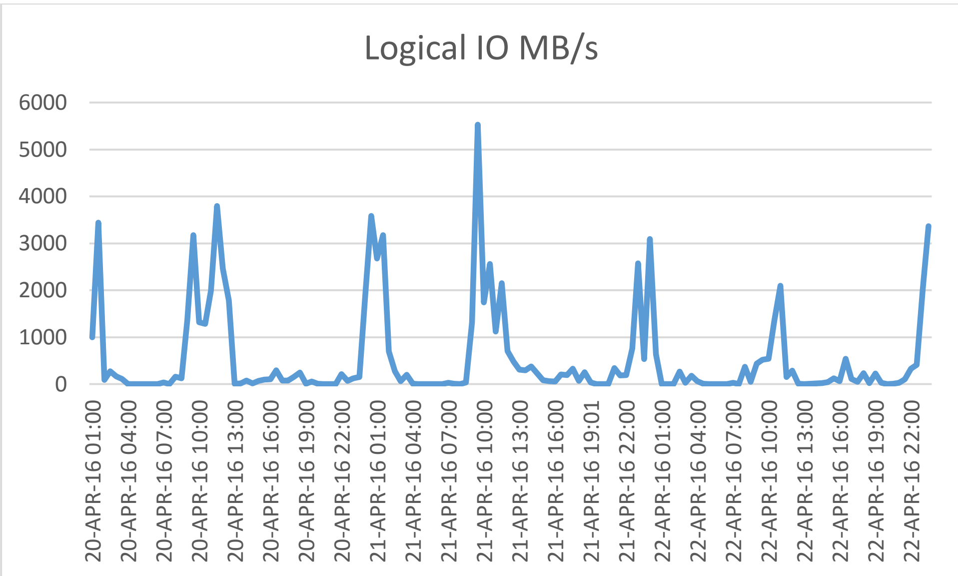

Figure 7: Logical IO for TWR

4.7. Transactions Rate and User Calls

Peak periods are often defined by the transaction arrival rate. In Oracle, a transaction is defined by a series of operations that result in either a COMMIT operation, or a ROLLBACK operation.

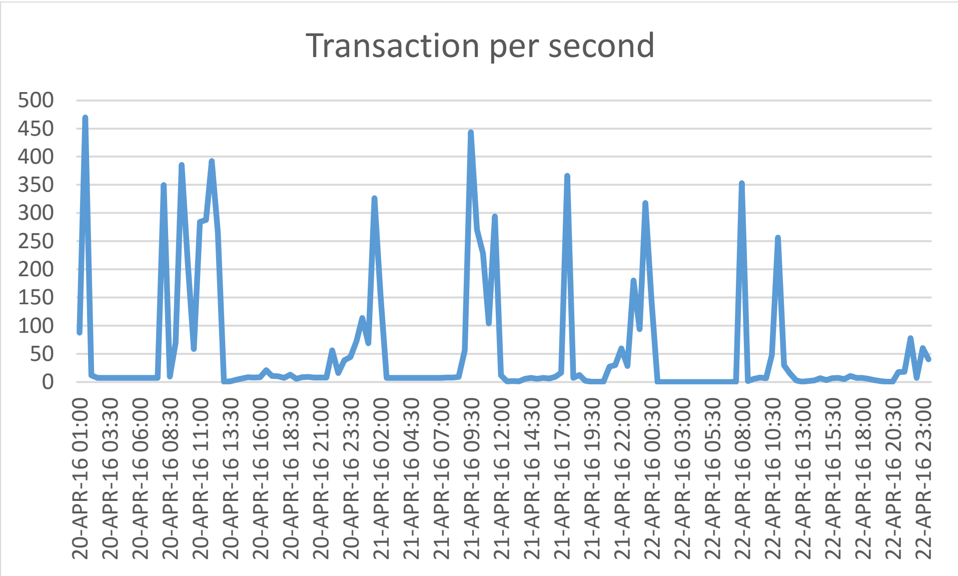

Figure 8: Transactions per Second for TWR

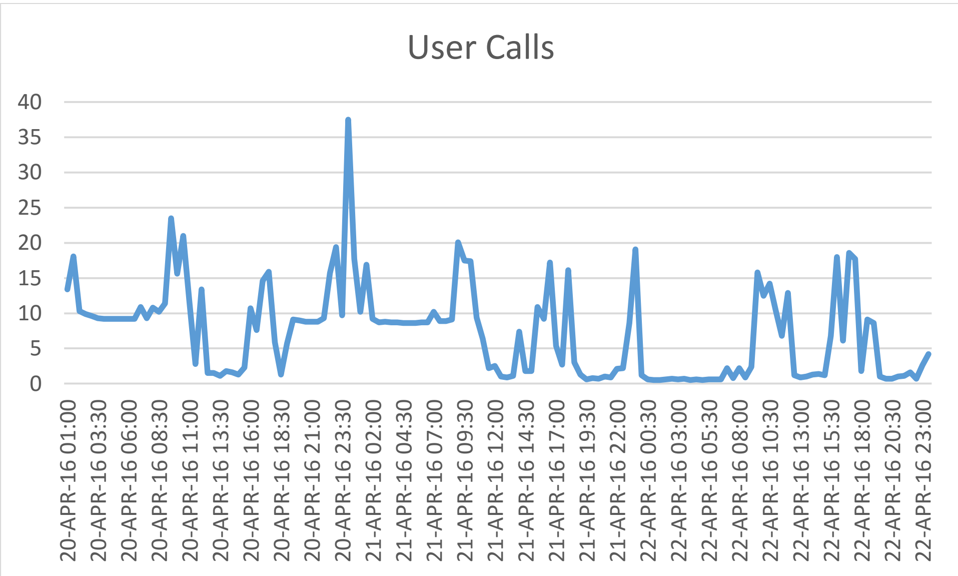

Figure 9: User call per seconds for TWR

4.8. PGA Memory Statistics

The Program Global Area (PGA) is a memory buffer that contains data and control information for a server process. A PGA is created by Oracle when a server process is started. The information in a PGA depends on the Oracle configuration.
Access to it is exclusive to that server process and is read from and written to only by the Oracle code acting on behalf of it. An example of such information is the runtime area of a cursor. Each time a cursor is executed, a new runtime area is created for that cursor in the PGA memory region of the server process executing that cursor. Analyze this section helps when using the new model to allocate PGA.

The goal is to have most work areas running with an optimal size (for example, more than 90% or even 100% for pure OLTP systems), while a smaller fraction of them are running with a one-pass size (for example, less than 10%). Multi-pass execution should be avoided. Even for DSS systems running large sorts and hash-joins, the memory requirement for the one-pass executions is relatively small. A system configured with a reasonable amount of PGA memory should not need to perform multiple passes over the input data.

Under automatic PGA memory management mode, Oracle honors the PGA_AGGREGATE_TARGET limit by controlling dynamically the amount of PGA memory allotted to SQL database areas. At the same time, Oracle maximizes the performance of all the memory-intensive SQL operators by maximizing the number of database areas that are using an optimal amount of PGA memory (cache memory). The rest of the database areas are executed in one-pass mode, unless the PGA memory limit set by PGA_AGGREGATE_TARGET is so low that multipass execution is required to reduce even more the consumption of PGA memory and honor the PGA target limit.

In 11g, PGA_AGGREGATE_TARGET controls work areas allocated by both dedicated and shared connections.
This metric is computed by Oracle to reflect the performance of the PGA memory component. It is cumulative from instance start-up. A value of 100% means that all work areas executed by the system since instance start-up have used an optimal amount of PGA memory. This is, of course, ideal but rarely happens except maybe for pure OLTP systems. In reality, some work areas run one-pass or even multi-pass, depending on the overall size of the PGA memory. When a work area cannot run optimally, one or more extra passes is performed over the input data. This reduces the cache-hit percentage in proportion to the size of the input data and the number of extra passes performed.

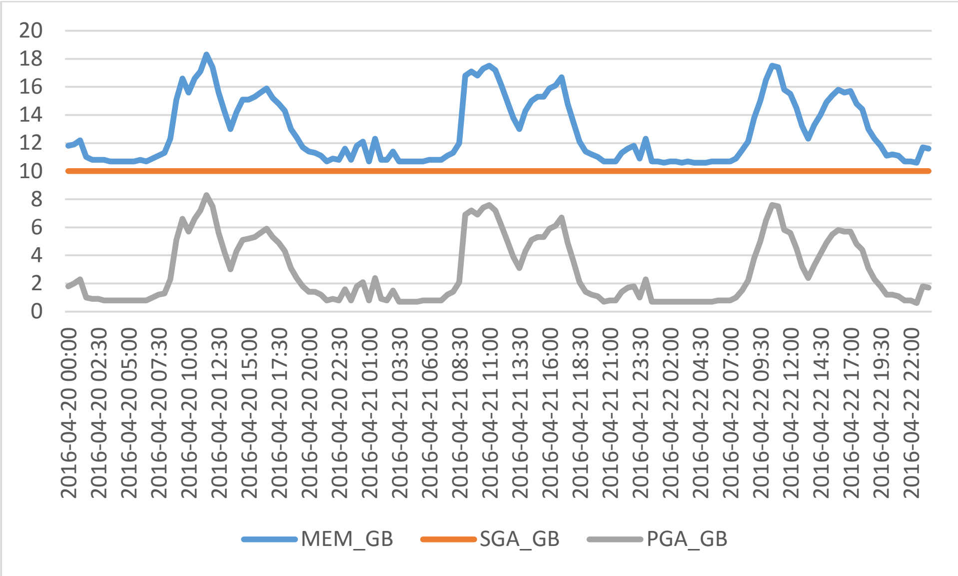

Figure 10: Memory allocated for TWR

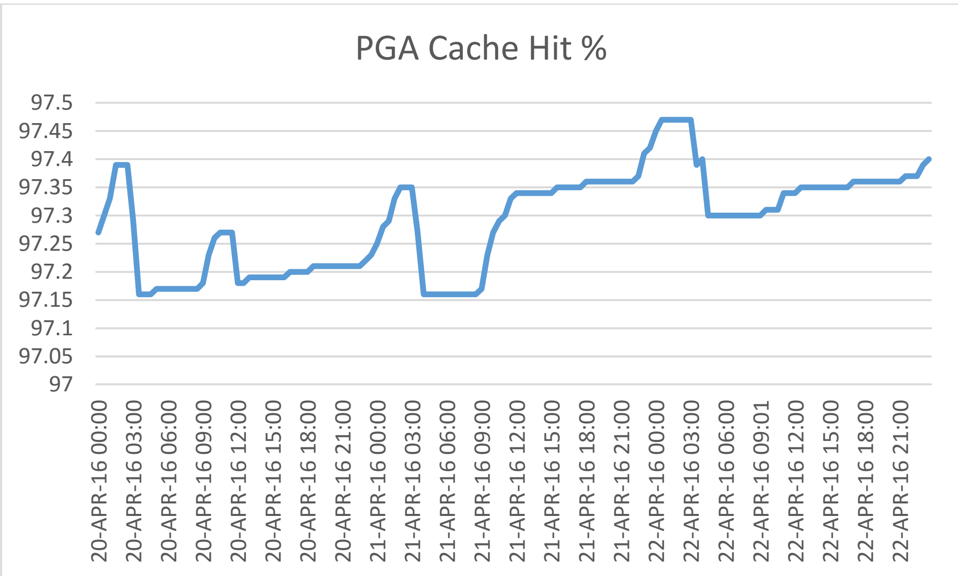

Figure 11: PGA cache hit % for TWR

Observation:

The current size of PGA is already optimal.

4.9. Redo Transaction Activity

Redo logs contain the transaction data that is created when users submit changes to the database. As each redo log fills to capacity with these changes, it "switches" to the next redo log to continue, while the just filled redo log is then copied by the archiving process. If redo log switching is occurring too frequently, slowdowns may be experienced while waiting on the archiver process to finish, or for redo log space manipulation to occur. Increasing or decreasing the size of the redo logs can easily adjust the rate of switching.

#	GROUP#	MEMBER	SIZE (GB)
1	1	+DATA02/twrdc/onlinelog/group_1.445.908894179	1
2	1	+FRA02/twrdc/onlinelog/group_1.702.908894181	1
3	2	+DATA02/twrdc/onlinelog/group_2.446.908894183	1
4	2	+FRA02/twrdc/onlinelog/group_2.510.908894185	1
5	3	+DATA02/twrdc/onlinelog/group_3.447.908894187	1
6	3	+DATA02/twrdc/onlinelog/group_3.447.908894187	1

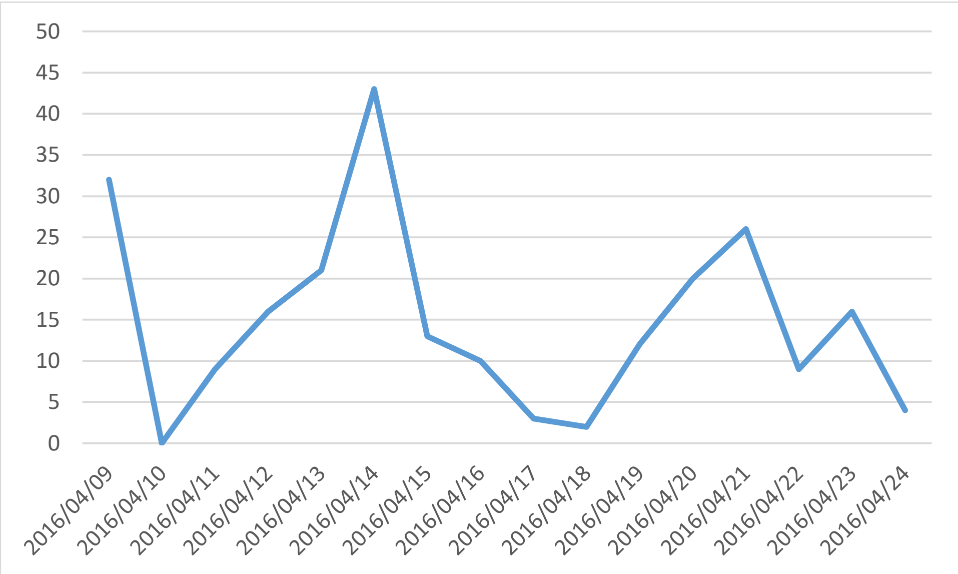

Figure 12: Redo log per day (GB) for TWR

Recommendation: Add multiplex redo log member - Oracle recommend 2 members in each redo log group, each belongs to different mount point/disk group.

5. Findings

5.1. Redundant index

These indexes has other superset index which means Oracle can use the superset index:

#	TABLE OWNER	TABLE NAME	REDUNDANT INDEX	SUPERSET INDEX
1	A4M	TACQFIID4TWA	IACQFIID4TWAM (BRANCH:MODULECODE)	PK_ACQFIID4TWA (BRANCH:MODULECODE:FICODE)
2	A4M	TADJCONTRACTNOCFG	IADJCONTRCFG_CNO2CODE (BRANCH:CODE)	PK_ADJCONTRACTNOCFG (BRANCH:CODE:CONTRACTNO)
3	A4M	TANALYZEACQGROUP	IANALYZEACQGROUPM (BRANCH:MODULECODE)	PK_ANALYZEACQGROUP (BRANCH:MODULECODE:GROUPCODE)
4	A4M	TAZERICARDACCOUNTLOG	I_TAZERICARDACCOUNTLOG (BRANCH:ACCOUNTNO)	I_TAZERICARDACCOUNTLOGIDCLIENT (BRANCH:ACCOUNTNO:IDCLIENT)
5	A4M	TAZERICARDUNLOADEDPAYMENTS	I_TAZERICARDUNLOADEDPAYMENTS (DOCNO)	PK_TAZERICARDUNLOADEDPAYMENTS (DOCNO:NO)
6	A4M	TCARDCONTRACT	ICARDCONTRACTCLIENT (BRANCH:IDCLIENT)	ICARDCONTRACTNO (BRANCH:IDCLIENT:CONTRACT)
7	A4M	TCARDTASKIMPORTREC	ICARDTASKIMPORTRECPACK (BRANCH:PACKNO:RECTYPE)	UK_CARDTASKIMPORTREC (BRANCH:PACKNO:RECTYPE:RECNO)
8	A4M	TCLERKTOSTATIONS	ICLERKTOSTATIONS_CLERK (BRANCH:CLERKCODE)	PK_TCLERKTOSTATIONS (BRANCH:CLERKCODE:STATION)
9	A4M	TCLIENTBANKCORRACCOUNT	ICLIENTBANKCORRACCOUNT_ID (BRANCH:IDCLIENT)	UK_CLIENTBANKCORRACCOUNT (BRANCH:IDCLIENT:BIC:ACCOUNT:CORRACC)

10	A4M	TCLIENTCNS	ICNSCLIENTIDCNS (BRANCH:IDCNS)	UK_CLIENTCNS (BRANCH:IDCNS:CHANNEL:ADDRESS)
11	A4M	TCONTRACTDELINQSETUP	ICONTRACTDELINQSETUP (BRANCH:CONTRACTTYPE:PERIOD)	PK_TCONTRACTDELINQSETUP (BRANCH:CONTRACTTYPE:PERIOD:OVERLIMIT)
12	A4M	TCONTRACTDELINQSETUP	ICONTRACTDELINQSETUP_CTYPE (BRANCH:CONTRACTTYPE)	ICONTRACTDELINQSETUP (BRANCH:CONTRACTTYPE:PERIOD)
13	A4M	TCONTRACTDELINQSETUP	ICONTRACTDELINQSETUP_CTYPE (BRANCH:CONTRACTTYPE)	PK_TCONTRACTDELINQSETUP (BRANCH:CONTRACTTYPE:PERIOD:OVERLIMIT)
14	A4M	TCONTRACTREDINTSETTINGS	ICTREDINTSETTINGS_PROFILEID (BRANCH:PROFILEID)	PK_TCONTRACTREDINTSETTINGS (BRANCH:PROFILEID:GROUPID:PRIORITY)
15	A4M	TCONTRACTSTMINPAYMENTDATA	ICONTRSTMINPAYDATA_BR_RECNO (BRANCH:SCRECNO)	ICONTRACTSTMPDATA_RECNO (BRANCH:SCRECNO:CURRENCYNUMBER)
16	A4M	TCONTRACTTYPELINKPARAMS	ICONTRACTTYPELINKID (LINKID)	ICONTRACTTYPELINKP2P (LINKID:PARAMTYPE:MAINPARAMCODE:LINKPARAMCODE)
17	A4M	TDISPMANREASONCODE	IDX_TDISPMANREASONCODE_PS (BRANCH:PAYMENTSYSTEM)	IDX_TDISPMANREASONCODE_PSRC (BRANCH:PAYMENTSYSTEM:REASONCODE)
18	A4M	TDISPMANREASONCODE	IDX_TDISPMANREASONCODE_PS (BRANCH:PAYMENTSYSTEM)	PK_TDISPMANREASONCODE (BRANCH:PAYMENTSYSTEM:REASONCODE:SUBREASONCODE)
19	A4M	TDISPMANREASONCODE	IDX_TDISPMANREASONCODE_PSRC (BRANCH:PAYMENTSYSTEM:REASONCODE)	PK_TDISPMANREASONCODE (BRANCH:PAYMENTSYSTEM:REASONCODE:SUBREASONCODE)
20	A4M	TDISPUTEDATA	I_DISPUTEDATAPS (BRANCH:PSYSTEM)	I_DISPUTEDATAPSMSG (BRANCH:PSYSTEM:MSGCODE)

21	A4M	TDISPUTEDATA	I_DISPUTEDATAPS (BRANCH:PSYSTEM)	I_DISPUTEDATAPSPAR (BRANCH:PSYSTEM:PARENTAL)
22	A4M	TDYNASQLSOURCE	I_DYNASQLSOURCE (CODE)	PK_DYNASQLSOURCE (CODE:LINE)
23	A4M	TEPAYTRAN2TRAN	ITEPAYTRAN2TRAN (BRANCH:MASTERTRAN)	PK_EPAYTRAN2TRAN (BRANCH:MASTERTRAN:DETAILTRAN)
24	A4M	TEUROPAYCONFIG	ITEUROPAYCONFIG (BRANCH:OPTIONOWNER)	PK_EUROPAYCONFIG (BRANCH:OPTIONOWNER:OPTIONTYPE)
25	A4M	TEUROPAYTERMPROGRAMME	IEUROPAYTERMPROGRAMME (BRANCH:IFP)	UK_TEUROPAYTERMPROGRAMME (BRANCH:IFP:TERMCODE:TIER)
26	A4M	TEXMASKACCODB	I_TEXMASKACCODB_TYPE (BRANCH:MODULECODE:PACKETTYPE)	PK_TEXMASKACCODB (BRANCH:MODULECODE:PACKETTYPE:ACCOUNTMASK)
27	A4M	TEXOUTISSFIID	IEXOUTISSFIID_2 (BRANCH:CODE_FIID)	PK_EXOUTISSFIID (BRANCH:CODE_FIID:CODE_MODULE)
28	A4M	TEXPAYODB	I_TEXPAYODB_ADM (BRANCH:ISADMIN)	I_TEXPAYODB_OPDATE (BRANCH:ISADMIN:OPDATE)
29	A4M	TFINSTATECLIENT	IFINSTATECLIENT_IDCLIENT (BRANCH:IDCLIENT)	IFINSTATECLIENT_IDCLIENT_CODE (BRANCH:IDCLIENT:CODE)
30	A4M	TGUTAACCOUNTLOG	I_TGUTAACCOUNTLOG (BRANCH:ACCOUNTNO)	I_TGUTAACCOUNTLOGIDCLIENT (BRANCH:ACCOUNTNO:IDCLIENT)
31	A4M	TISSCLIENT	IISSCLIENT_PACKET (BRANCH:PACKETNO)	PK_ISSCLIENT (BRANCH:PACKETNO:RECORDNO)
32	A4M	TLOYALTYPRIZE_PROG2ITEM	IPROG2ITEM_PROG (BRANCH:PROGCODE)	PK_LOYALTYPRIZE_PROG2ITEM (BRANCH:PROGCODE:ITEMCODE)
33	A4M	TMASKCONTAINERFIELDS	I_TMASKCONTAINERFIELDS (BRANCH:MASKTYPE)	UK_MASKTYPE_FIELDS (BRANCH:MASKTYPE:FIELDORDER)
34	A4M	TOWNFIID4TWA	IOWNFIID4TWAM (BRANCH:MODULECODE)	PK_OWNFIID4TWA (BRANCH:MODULECODE:FICODE)

35	A4M	TPLANACCOUNT	IPLANACCOUNT_LEDGER (BRANCH:LEDGER)	IPLANACCOUNT_SUBLEDGER (BRANCH:LEDGER:SUBLEDGER)
36	A4M	TREFERENCEHOLIDAYS	I_TREFERENCEHOLIDAYS (BRANCH:ID_CLND)	UK_TREFERENCEHOLIDAYS (BRANCH:ID_CLND:DAYS)
37	A4M	TREGULARPROCESSED	IDX_REGULARPROCESSED_2 (BRANCH:PAYMENTCODE:PAYMENTDATE)	PK_REGULARPROCESSED (BRANCH:PAYMENTCODE:PAYMENTDATE:PROCPACKNO)
38	A4M	TRETAILERCUSTOMER	I_RETAILERCUSTOMER1 (BRANCH:RETAILERCODE)	UK_RETAILERCUSTOMER (BRANCH:RETAILERCODE:CUSTOMERID)
39	A4M	TRETAILEREVENTLOG	I_NO__TRETAILEREVENTLOG (BRANCH:NO)	I_TRETAILEREVENTLOG (BRANCH:NO:RETCODE:EVENTTYPE:ATTRIBUTE:EVENTOPDATE)
40	A4M	TTERMINALEVENTLOG	I_NO_TTERMINALEVENTLOG (BRANCH:NO)	I_TTERMINALEVENTLOG (BRANCH:NO:TERMCODE:EVENTTYPE:ATTRIBUTE:EVENTOPDATE)
41	A4M	TUCSACCOUNTLOG	I_TUCSACCOUNTLOG (BRANCH:ACCOUNTNO)	I_TUCSACCOUNTLOGIDCLIENT (BRANCH:ACCOUNTNO:IDCLIENT)
42	A4M	TURALSIBACCOUNTLOG	I_TURALSIBACCOUNTLOG (BRANCH:ACCOUNTNO)	I_TURALSIBACCOUNTLOGIDCLIENT (BRANCH:ACCOUNTNO:IDCLIENT)
43	A4M	TVTB24ACCOUNTLOG	I_TVTB24ACCOUNTLOG (BRANCH:ACCOUNTNO)	I_TVTB24ACCOUNTLOGIDCLIENT (BRANCH:ACCOUNTNO:IDCLIENT)
44	A4M	TWFREFERENCEWFTYPEPROP	IWFREFWFTYPEPROP_WFTYPE (BRANCH:WFTYPE)	PK_TWFREFERENCEWFTYPEPROP (BRANCH:WFTYPE:WFTYPEPROP)
45	TWI	PANRANGE	IDX_PANRANGE_IINID (IINID)	IDX_PANRANGE_IINIDMINRANGE (IINID:MINRANGE)

Recommendation: Drop redundant index to reduce overhead on DML & release storage space.

5.2. Resource Intensive SQL

Any effort of improving performance must involve reviewing the actual SQL code that is submitted in transactions to determine if it can be optimized. The largest performance improvements usually come from tuning the actual code, not from adjusting database parameters. It is important to periodically extract the most resource intensive SQL and examine it for improvement opportunities.

Following SQL statements should be tuned to gain maximum benefits.

5.2.1. SQL ID 3xb8kjfcwmm8r - Module cpi2_CardCenter.exe

```
SELECT NO
  FROM TCONTRACTITEM
 WHERE KEY LIKE :B1
```

```
Plan hash value: 4164302988

-------------------------------------------
| Id  | Operation          | Name         |
-------------------------------------------
|   0 | SELECT STATEMENT   |              |
|   1 |  TABLE ACCESS FULL| TCONTRACTITEM |
-------------------------------------------
```

Stat Name	Statement Total	Per Execution	% Snap Total
Elapsed Time (ms)	8,152,413	160.90	0.04
CPU Time (ms)	8,144,656	160.74	2.20
Executions	50,669		
Buffer Gets	357,642,990	7,058.42	1.96
Disk Reads	463	0.01	0.00
Parse Calls	2	0.00	0.00
Rows	50,672	1.00	

Recommendation: Create a new index on KEY column.

5.2.2. SQL ID 03c0kqjh3nstd - Module cpi2_VIP.exe

```
SELECT COUNT (1)
  FROM TREFERENCETERMINAL T, TREFERENCERETAILER A, TACCGROUP B
 WHERE      T.BRANCH = 1
       AND A.BRANCH = 1
       AND B.BRANCH = 1
```

```
AND T.FICODE = 1
AND A.FICODE = 1
AND T.RETAILERCODE = A.CODE
AND A.ACCOUNTGROUPCODE = B.CODE
AND T.IDENT = :B2
AND B.CURRENCY = :B1
```

```
Plan hash value: 4171129957

-------------------------------------------------------------------------
| Id  | Operation                        | Name                         |
-------------------------------------------------------------------------
|   0 | SELECT STATEMENT                 |                              |
|   1 |  SORT AGGREGATE                  |                              |
|   2 |   NESTED LOOPS                   |                              |
|   3 |    NESTED LOOPS                  |                              |
|   4 |     TABLE ACCESS BY INDEX ROWID| TREFERENCERETAILER           |
|   5 |      INDEX RANGE SCAN            | RETAILER_SKEY                |
|   6 |     INDEX UNIQUE SCAN            | UK_TACCGROUP_CODECURRENCY |
|   7 |    INDEX UNIQUE SCAN             | TERMINAL_SKEY                |
-------------------------------------------------------------------------
```

Stat Name	Statement Total	Per Execution	% Snap Total
Elapsed Time (ms)	121,748	14.76	0.00
CPU Time (ms)	116,080	14.08	0.03
Executions	8,246		
Buffer Gets	102,989,873	12,489.68	0.56
Disk Reads	569	0.07	0.00
Parse Calls	11	0.00	0.00
Rows	8,246	1.00	

Table TREFERENCETERMINAL has 6,280 rows.
Column *IDENT* has 6265 distinct values. This column was indexed on TERMINAL_SKEY(BRANCH, FICODE, RETAILERCODE, IDENT) but it is ineffective to use because it is on 4th position. Create a separate index on it to make it more efficient to retrieve data.

Recommendation: Create a new index on IDENT column of TREFERENCETERMINAL table.

Appendix 1 – Other Documentation

MTS and Large Pool:

Note:62140.1	Fundamentals of the Large Pool
Note:268581.1	Obsolete / Deprecated Initialization Parameters in 10G

Checkpoints:

Note:265831.1	Automatic Checkpoint Tuning in 10g
Note:274264.1	REDO LOGS SIZING ADVISORY in 10g

Statistics:

Note: 266040.1	Automatic statistics Gathering in oracle 10G
Note: 252597.1	Relation between Table Monitoring and STATISTICS_LEVEL parameter in 10g
Note:281790.1	Oracle Database 10g DBMS_STATS Package FORCE argument
Note: 283890.1	Oracle Database 10g Locking Statistics

Locally Managed Tablespaces:

Note:93771.1	Introduction to Locally-Managed Tablespaces
Note:262472.1	10g: BIGFILE Type Tablespaces Versus SMALLFILE Type

Tuning CPU Resources

Note: 33824.1	Statistic - recursive cpu usage
Note: 164768.1	Diagnosing High CPU Utilization
Note: 33828.1	Statistic - cpu used by this session (Reference Note)
Note: 33854.1	Statistic - parse time elapsed (Reference Note)
Note: 33853.1	Statistic - parse time cpu (Reference Note)
Note: 33828.1	Statistic - cpu used by this session (Reference Note)
Note: 276103.1	PERFORMANCE TUNING USING 10g ADVISORS AND MANAGEABILITY FEATURES

MTS and Large Pool:

Tuning I/O

Note: 30286.1	I/O Tuning with Different RAID Configurations
Note: 1037322.6	WHAT IS THE DB_FILE_MULTIBLOCK_READ_COUNT PARAMETER
Note: 148342.1	Avoiding I/O Disk Contention
Note: 245055.1	Oracle Database 10g Enhanced wait model
Note: 272360.1	Tablespace Groups for SQL Operations in 10g
Note: 242090.1	10g NEW FEATURE on SEGMENT SHRINK

Optimizing SQL Statements

Note: 10585.1	Query and Application Tuning using Explain and TKProf
Note: 163563.1	START POINT: My Query runs slowly
Note: 33089.1	TROUBLESHOOTING GUIDE: SQL Tuning
Note: 67522.1	Why is my index not used?
Note: 69992.1	Why is my hint ignored?
Note: 34558.1	Waitevent - db file scattered read (Reference Note)
Note: 34396.1	Waitevent - SQL*Net message from dblink (Reference Note)
Note: 34559.1	Waitevent - db file sequential read (Reference Note)
Note 259188.1	Oracle10g: Using SQLAccess Advisor (DBMS_ADVISOR) with the Automatic Workload Repository
Note: 262687.1	How to use the Sql Tuning Advisor
Note:244192.1	10g NEW FEATURE Automatic Database Diagnostic Monitor (ADDM)
Note:250655.1	How to use the Automatic Database Diagnostic Monitor
Note: 290027.1	Computationally intensive PL/SQL programs run fast on 10G as compared to 9i

Tuning Network Resources

Note: 44694.1	SQL*Net Packet Sizes (SDU and TDU Parameters)
Note: 1005123.6	Tuning SQL*Net for better performance

MTS and Large Pool:

Tuning Memory

Note: 257643.1	Oracle Database 10g Automated SGA Memory Tuning
Note: 295626.1	How To Use Automatic Shared Memory Management (ASMM) In Oracle10g

Appendix 2 - Methodology

A number of tools and methods were involved in examining and reviewing the systems, and providing the recommendations:

Statspack

Statspack is an Oracle provided utility that collects information and stores the performance statistics data permanently in Oracle tables, which can later be used for reporting and analysis. The data collected can be analyzed using the report provided, which includes an "instance health and load" summary page, high resource SQL statements, as well as the traditional wait events and initialization parameters.

High-Water Mark Viewer

This Excel spreadsheet connects to a local Oracle database via OO4O and extracts high-water mark information on all tables. It then generates an Excel chart that depicts the impact on full table scans.

Oracle Trender

This Excel spreadsheet connects to a local Oracle database via OO4O and extracts information about wait events, statistics etc. Several Excel charts are automatically generated that depict the impact of these events on the overall system.

KM Repository on Oracle Support

KM Repository on Oracle Support is Oracle's electronic delivery of support information. A premier service for all Oracle-supported customers, it has a wealth of white papers, technical bulletins, user forums, Oracle documentation, and is used to open and work Technical Assistance Requests, verify supported configurations, etc. http://support.oracle.com

SAR

SAR is a performance data collection program found on most Unix platforms. It is configurable for interval and duration, and can capture critical CPU and disk performance data.

Remote Diagnostic Agent

The Remote Diagnostic Agent is an Oracle-developed tool that is designed to collect significant amounts of configuration information from both hosts and databases. Primarily used to diagnose problem issues, the information can also be used proactively.

Appendix 3 - Caveats

There are several potential issues with a health check of this nature.

- The data held internally in SYS or SYSTEM tables and views can be tainted by issues that are now rectified. For example, a database has been running for 4 weeks with a frequently executed query resulting in an expensive full table scan against a 1 million block table. Just prior to the health check a unique index was added to this table resulting in all queries using a unique index lookup. V$SYSTEM_EVENT would probably still show that too many db file scattered read' waits had occurred, and the original queries against that table are likely in the top x queries ordered by buffer gets/physical reads, yet the issue has now been rectified.

- Several of the findings in this document are based on ratios. These are not always a solid basis for analyzing a particular component's performance. Using the frequently quoted buffer cache example, a database can have 5 sql statements that constitute 90% of the load. These queries result in full table scans of a relatively large table and are executed frequently so that each full table scan results in mostly logical rather than physical I/Os. The outcome of this scenario might be that the Buffer Cache Hit Ratio is a very nice 99.99%. Say changing the statements to use indexes and altering the application to execute the sql less often resulted in an 80% drop in overall I/Os. Likely the Buffer Cache Hit Ratio has now dropped but the database is much healthier as a result!

- Most importantly, the Performance Review is based on the difference between two snapshots of how the database has been performing at particular points in time, based on its load and sql at those times. Reducing the load on the database, for example by reducing logical and physical I/O, or reducing parsing, may negate the need to implement some of the recommendations outlined below. To give an example, the Shared Pool Advisor may be indicating that the shared pool size needs to be increased. However, changing the code to use bind variables instead of literals may increase cursor shareability and therefore free space in the shared pool. Result, shared pool no longer needs resizing so the recommendation now becomes redundant.

- In some cases the 'av rd (ms)' columns in a statspack report can show unusually high numbers - in some cases this can be converted to much more time than there is between snapshots. If this is the case in your existing statspack snapshots, you may need to apply patch 4942939.