

OraPerf.com:  
Real Life Performance Data  
Session #1493

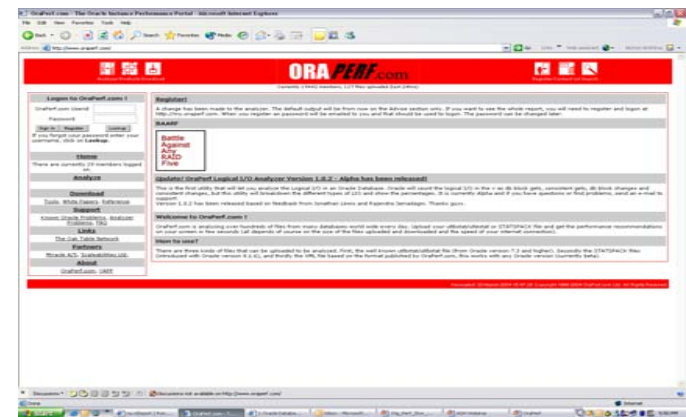
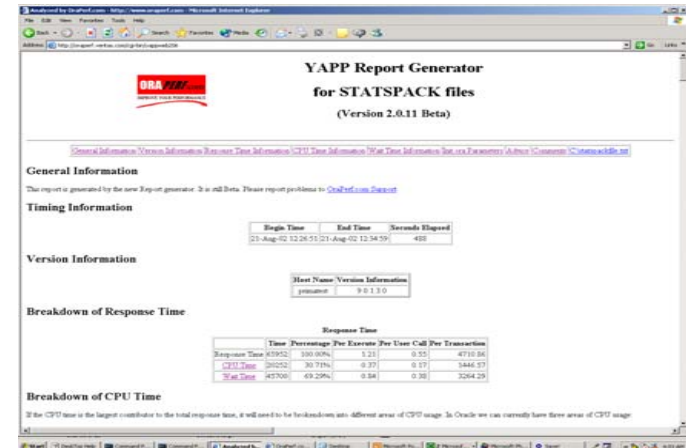
Anjo Kolk

APM Division

Veritas Software

# My background

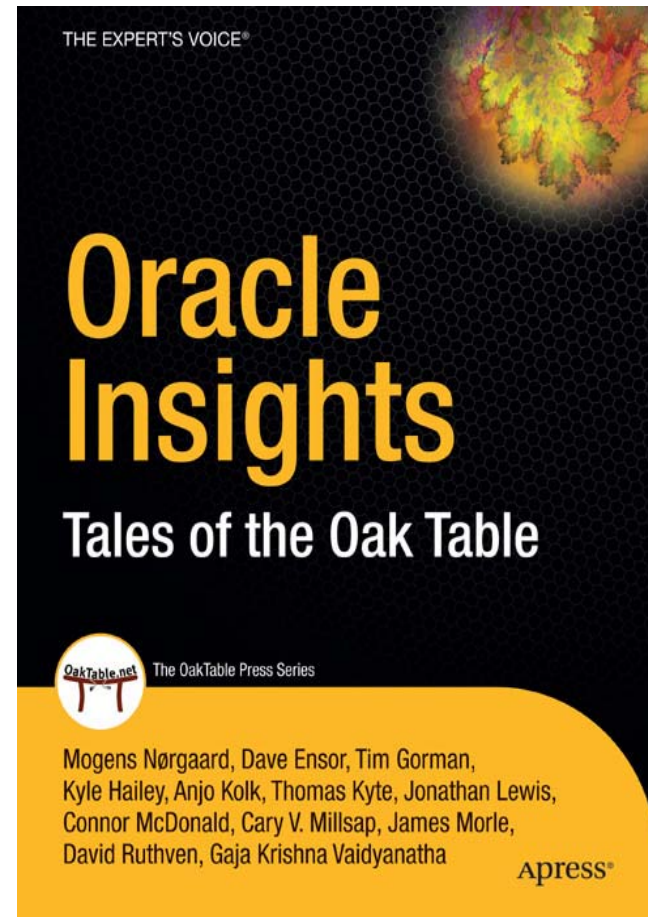
- Started working with Oracle version 4
- Developer, Consultant and Trouble shooter in the past 19 years
- Oracle7 Wait Events and Enqueues White paper
- Yet Another Performance Profiling (YAPP) White Paper: Response Time Tuning for Oracle
- Founding member of the Oaktable Network
- OraPerf.com website: owner and developer  
OraPerf.com website: owner and developer



# Oracle Insights

---

- Oracle Insights : Tales of the Oaktable
- A 11-author Book Project
  - Dave Ensor
  - Tim Gorman
  - Kyle Hailey
  - Anjo Kolk
  - Jonathan Lewis
  - Connor McDonald
  - Cary Millsap
  - James Morle
  - Mogens Norgaard
  - David Ruthven
  - Gaja Krishna Vaidyanatha
- Published by Apress  
(<http://www.apress.com>)



# History of OraPerf.com

---

- Started in December 1998, in the basement
- Cable modem introduction
- Redhat Linux
- 2 CPU (366 Mhz Celeron, over clocked)
- Based on YAPP
- Initially supported only Utlbstat/Utlestat

# YAPP

---

- OraPerf.com is based on YAPP
- YAPP = Yet Another Performance Profiling Methodology
- Developed in 1996 as reaction to the Check list/Ratio tuning
- Solved a real (large) customer performance problem, and they still use it today.

## So what is YAPP?

---

- Basically (Response) Time based tuning (or taking a Holistic View of your system)
- $R = C + W$ 
  - R = (Response) Time
  - C = CPU/Service Time
  - W = Wait Time
- The basic idea is to look at all the time that is spent inside the database.
- Developed for sessions, but will also work on instance or SQL statement level

## So what is YAPP?

---

- Checklist tuning is basically going down your list of ratios:
  - Buffer cache hit ratio
  - Latch miss ratio
  - Parse/execute ratio, etc.
- Ratios should never be used as the starting point of your tuning process!
- And what are good percentages for your ratios?

# Oracle Buffer Cache Hit Ratio

---

- Use the expert guidelines:
  - 94 – 97% in Oracle Applications (Source: Oracle)
  - 80% (UNIX files) or > 90% (raw devices) (Source: Oracle)
  - 95% online day and > 85% batch (Source: Gurry & Corrigan)
  - 99.999% (Source: Richard Niemic)
- So even the experts don't agree!

# End User focus

---

- End users complain about
  - Response time problems
    - Noticed directly by end users
  - Throughput problems
    - Noticed by management and solved by throwing hardware at the problem. Most of the time the throughput will improve but not the response time.
- Should be used as the starting point of your tuning process, not a bad ratio

# Improving TCO

---

- Another reason for starting the tuning process can be to reduce the Total Cost of Ownership of a system
  - End users are happy, but can we provide the same performance/functionality with less?
    - Ratios can never help here.
- Looking at where you spend your time, you can!
  - Reduce CPU usage
  - Reduce Wait time

## Why utlBstat/utlEstat?

---

- Run utlBstat and some time later utlEstat to produce a file called report.txt
- Shipped with Oracle Release
  - utlBstat – Begin Statistics
  - utlEstat – End Statistics
- Many DBAs use this to monitor their database(s)
- But ....
  - Don't know how to interpret report.txt

# Problems with Utlbstat/Utlestat

---

- Create/drop tables
- Doesn't support OPS/RAC
- Only Instance statistics
- Not all interesting statistics are collected and reported

# Interesting stats that are missing

---

- For Example:
  - SQL statements
    - Buffer gets
    - Disk reads
    - Parse calls
    - Executes
    - Version count
  - Enqueue stats
  - Detailed latch statistics

## STATSPACK since 8.1.6

---

- Much better than utlstat/utlestat, but requires a bit more administration.
- The report/file generated contains much more info, but like report.txt it is difficult to read (file can be large)
- So lets sign up with OraPerf.com and have the file analyzed.

# OraPerf

---

Logon or Register and then logon. After login on, you will enter your personal workspace.

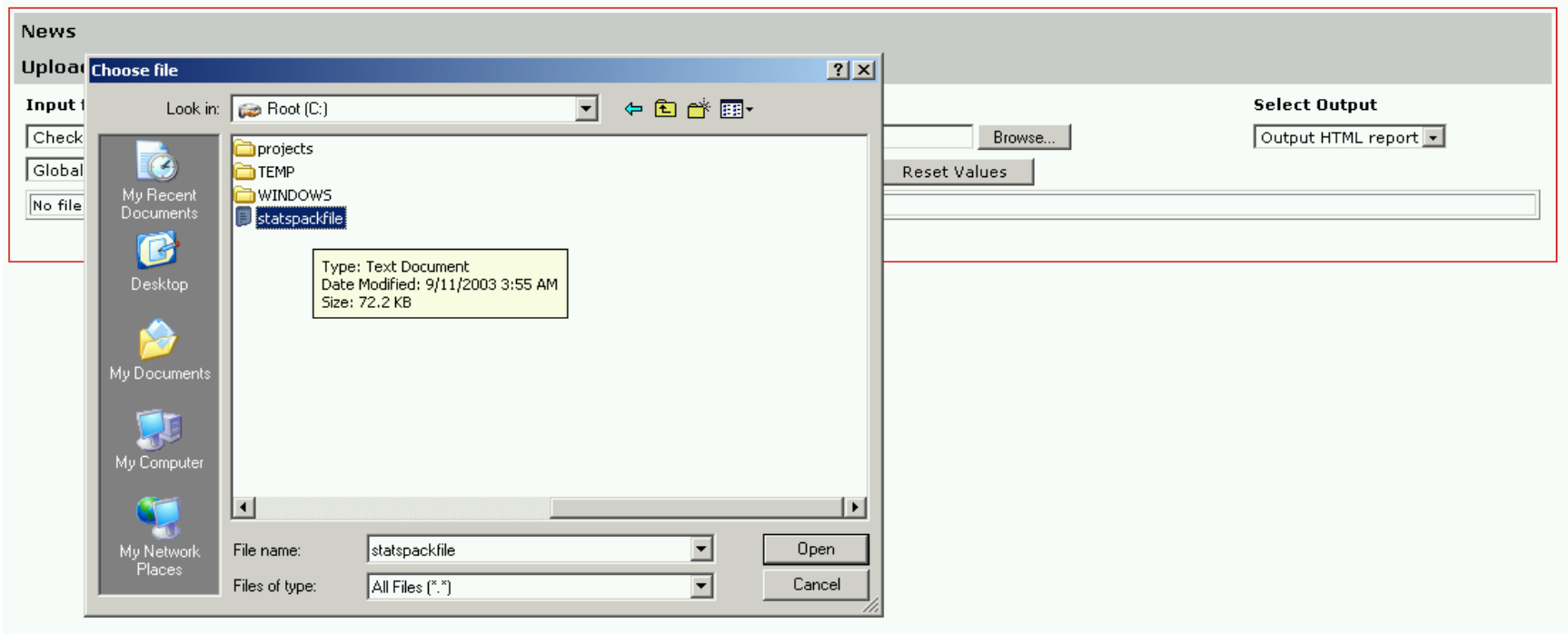
**News**

**Upload your file!**

<b>Input file type</b>	<b>Statspack version used</b>	<b>Report File</b>	<b>Select Output</b>
<input type="text" value="Check file type"/>	<input type="text" value="8.1.6"/>	<input type="text"/> <input type="button" value="Browse..."/>	<input type="text" value="Output HTML report"/>
<input type="text" value="Global Profile"/>		<input type="button" value="Upload File(s)!"/> <input type="button" value="Reset Values"/>	

# OraPerf

Click on the BROWSE button to start browsing for the file that you want to upload.

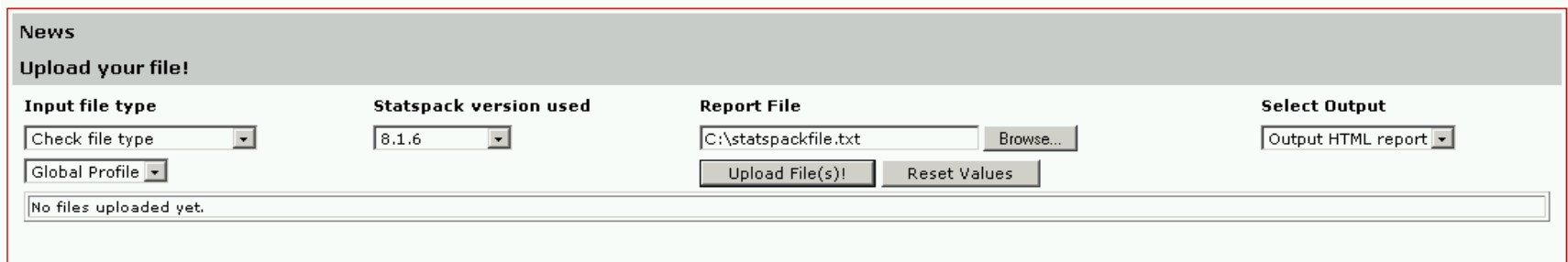


Then click on OPEN.

# OraPerf

---

If the file has been located, hit the **UPLOAD FILE(S)** button.



The screenshot shows a web interface for uploading files. At the top, there is a 'News' section. Below it, the main heading is 'Upload your file!'. The interface is divided into four columns: 'Input file type', 'Statspack version used', 'Report File', and 'Select Output'. Under 'Input file type', there is a dropdown menu set to 'Check file type' and another dropdown set to 'Global Profile'. Under 'Statspack version used', there is a dropdown menu set to '8.1.6'. Under 'Report File', there is a text input field containing 'C:\statspackfile.txt' and a 'Browse...' button. Below this, there are two buttons: 'Upload File(s)!' and 'Reset Values'. Under 'Select Output', there is a dropdown menu set to 'Output HTML report'. At the bottom of the form, there is a text area containing the message 'No files uploaded yet.'


One can take the default settings of the form, but incase the file is not recognized set the file type (“check file type”) or the version of STATSPACK that is used (not the RDBMS version).

Analyzed by OraPerf.com - http://www.oraperf.com - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites Media

Address http://oraperf.veritas.com/cgi-bin/yappweb206 Go Links



## YAPP Report Generator for STATSPACK files (Version 2.0.11 Beta)

---

[General Information](#)
[Version Information](#)
[Reponse Time Information](#)
[CPU Time Information](#)
[Wait Time Information](#)
[Init.ora Parameters](#)
[Advice](#)
[Comments](#)
[C:\statspackfile.txt](#)

### General Information

This report is generated by the new Report generator. It is still Beta. Please report problems to [OraPerf.com Support](#)

### Timing Information

Begin Time	End Time	Seconds Elapsed
21-Aug-02 12:26:51	21-Aug-02 12:34:59	488

### Version Information

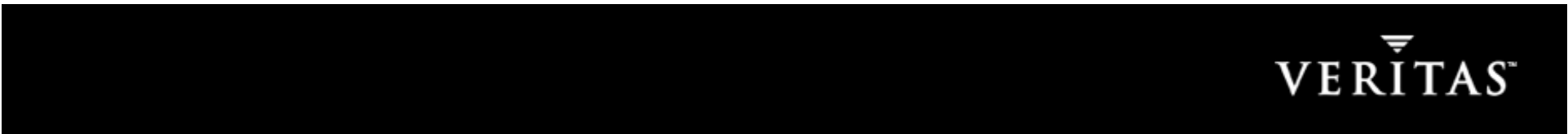
Host Name	Version Information
primatest	9.0.1.3.0

### Breakdown of Response Time

	Time	Percentage	Per Execute	Per User Call	Per Transaction
Response Time	65952	100.00%	1.21	0.55	4710.86
<a href="#">CPU Time</a>	20252	30.71%	0.37	0.17	1446.57
<a href="#">Wait Time</a>	45700	69.29%	0.84	0.38	3264.29

### Breakdown of CPU Time

If the CPU time is the largest contributor to the total response time, it will need to be brokendown into different areas of CPU usage. In Oracle we can currently have three areas of CPU usage:



Analyzed by OraPerf.com - http://www.oraperf.com - Microsoft Internet Explorer

Address: http://oraperf.veritas.com/cgi-bin/yappweb206

### Advice

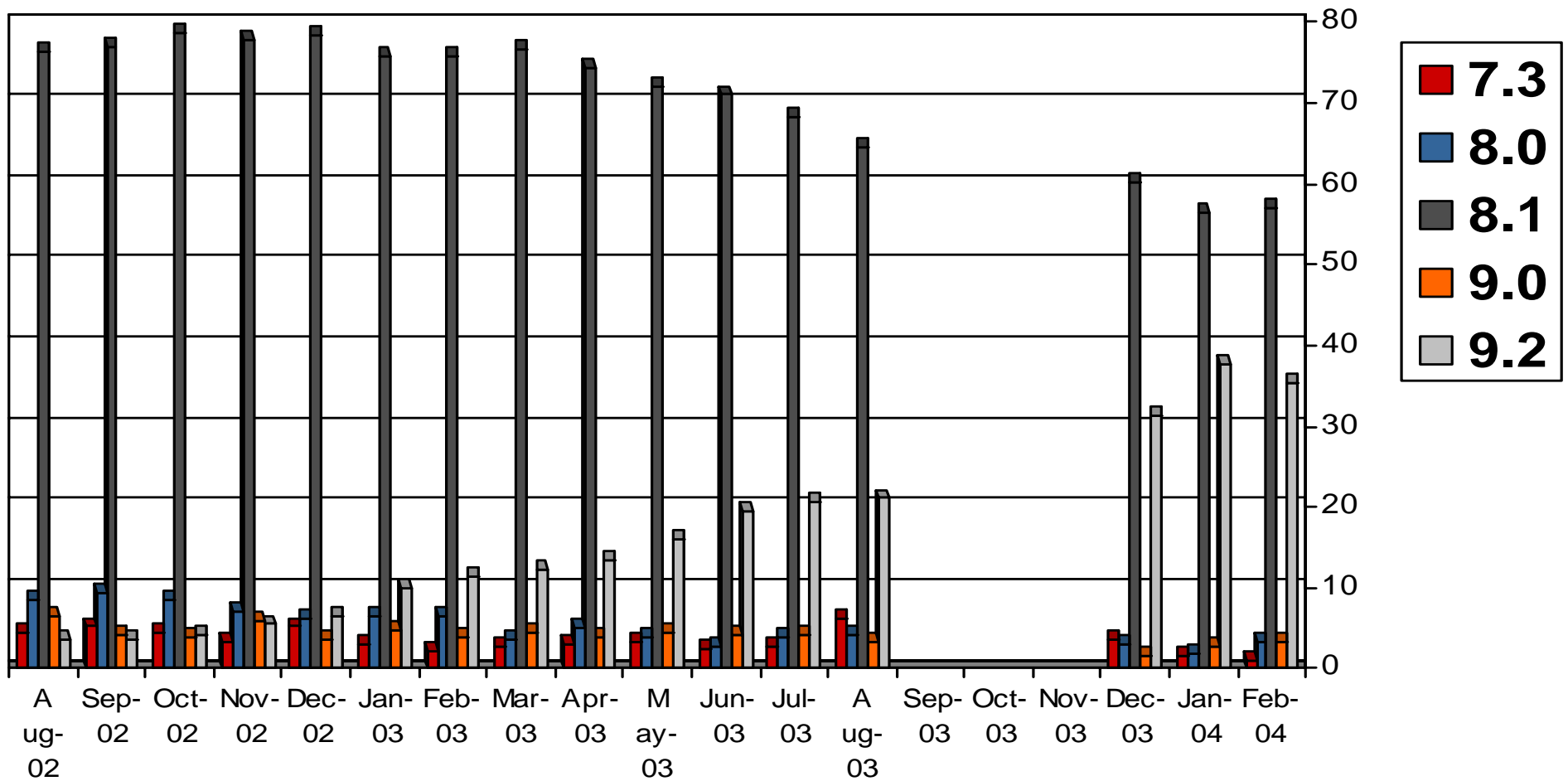
The advice is given in the order of the most impact of the total reponse time. The percentage gain is taken of the reponse time.

Maximum Gain (%)	What	
0	Check SQL*Net Configuration for SDU and TDU settings	Also check the usage of array inserts.. Also check the usage of array fetch
1	Tune the <b>direct path write</b> event.	No detailed information is available yet.
2	Tune the <b>log file sync</b>	<p>A number of things can be tried:</p> <ul style="list-style-type: none"> <li>• Reduce the init.ora parameter <b>processes</b></li> <li>• Stripe the redo log file(s) over multiple disks with a small stripe width (8-32K)</li> </ul>
3	Reduce the number of buffer gets or executions	<pre> SELECT M.CS_NUM_PIECE_MAT,SUBSTR(M.CS_NUM_CHRONO,5,8),M.CS_COD_TYP_PIECE_MAT,CS_COD_ARTICLE,CS_QTE_LIVRE_PLEIN,CS_QTE value 3781515447). SELECT T3.CS_COD_SITE_FACT,T3.CS_COD_ACTIVITE,T2.CS_NUM_CHRONO,T1.CS_NUM_LIGNE_FACT,T2.CS_COD_TYP_PIECE_VAL,T1.CS_COD_ARTICLE (hash value 2776196258). SELECT T2.CS_COD_SITE_FACT,T2.CS_COD_ACTIVITE,CS_NUM_PIECE_VAL,T1.CS_NUM_CHRONO,CS_DAT_PIECE,T1.CS_CC (T1.CS_COD_MONNAIE,'FRF'),CS_COD_MOTIF_DECONS,CS_COD_TYP_CONS,CS_LIB_NOM_USAGER,CS_LIB_ (hash value 871724077). SELECT C.CS_C (CS_MNT_ECHEANCE,0),E.CS_NUM_CHRONO,E.CS_DAT_ECHEANCE,D.CS_COD_SITE_FACT,D.CS_COD_ACTIVITE,D.CS_LIB_NOM_FACT,E.CS_NUM C.CS_COD_CLI,C.CS_LIB_NOM_TIRE,C.CS_LIB_NOM_BOE,C.CS_COD_GUICHET,C.CS_NUM_CPTE,C.CS_COD_ETS,NVL(CS_MNT_ECHEANCE,0),E.CS (hash value 570483654). SELECT T1.CS_COD_CLI,T1.CS_NUM_CHRONO_CLI,T1.CS_DAT_PIECE,T1.CS_MNT_ECHEANCE,T1.CS_DAT_ECHEANCE,NVL(T T2.CS_NUM_CHRONO_CLI AND T1.CS_NUM_C (hash value 2981666050). DELETE FROM BAREME_CLIENT WHERE CS_COD_REF_TARIF IN (SELECT CS CS_COD_SS_TYP_PIECE_VAL WHERE CS_COD_TYP_PIECE_VAL IN ('04','54') AND CS_NUM_JE IS NULL AND CS_DAT_ENVOI IS NULL AND CS_DAT (T1.CS_COD_MONNAIE,'EUR'),T3.CS_COD_SITE_FACT,T3.CS_COD_ACTIVITE,T1.CS_COD_TYP_PRESTATION,T1.CS_NUM_CHRONO,T1.CS_IND_REM C.CS_NUM_CHRONO_CLI,C.CS_COD_CLI,C.CS_COD_ETS,C.CS_COD_GUICHET,C.CS_NUM_CPTE,C.CS_COD_CLE_RIB FROM ECHEANCIER E,DOM_I DELETE FROM PRIX_BAREME WHERE CS_COD_REF_TARIF IN (SELECT CS_COD_REF_TARIF FROM REFERENCE_TARIF WHERE CS_COD_PRIORITE_ T2.CS_COD_SITE_FACT,T2.CS_COD_ACTIVITE,T1.CS_NUM_CHRONO,T1.CS_COD_TYP_PIECE_VAL,T1.CS_COD_SENS,T1.CS_MNT_TOT_VAL_TTC_S_ (hash value 3750526490). UPDATE TNORA.CSTPRIXBAREME SET CS_DAT_FIN_EFFET=b1 WHERE CS_COD_REF_TARIF = b2 AND CS_COD_ART_PROD = CS_DAT_ENVOI IS NULL AND CS_DAT_JOUR_CLOTURE IS NOT NULL ORDER BY CS_NUM_CHRONO (hash value 3494303593). DELETE FROM PRIX_BA </pre>
51	Tune the Multi Threaded Server	Running the Multi Threaded Server will allow for more sessions to run on your system, but will add some overhead for each session.

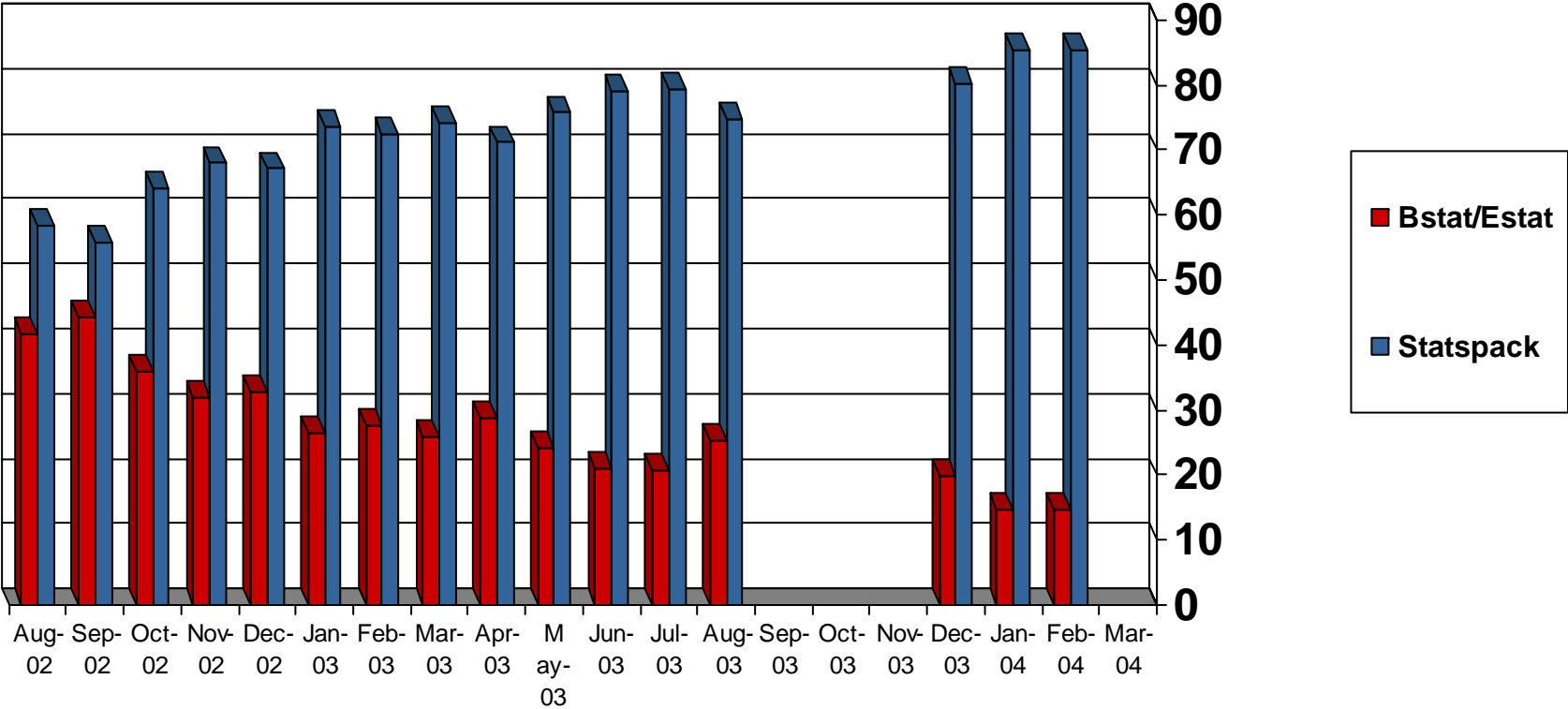
### Comments

Start | DeskTop Help | Command P... | Command P... | Analyzed b... | OraPerf.co... | Desktop | Microsoft Po... | 2 Microsof... | Microsoft Ph... | Save! | 4:05 AM

# Oracle Versions Uploaded



# Is STATSPACK getting popular?



# Common Problems

---

- Different databases and versions have the symptoms/problems:
  - Too many logical I/O
  - Slow physical I/O
  - Locking and Latching problems
- Why? There must be a common cause
  - Problem/limitations in Oracle kernel?
  - Oracle documentation not good (enough)?
  - Oracle books and courses not good (enough)?

## Symptom: Too many Logical I/O

---

- Logical I/O or buffer get
- Relation between logical I/O and CPU usage
- Buffer cache hit ratio
  - $(1 - (\text{physical reads} / (\text{db buffer gets} + \text{consistent gets})))$
- Too many logical I/O will give a good buffer cache hit ratio
- Reduce number of executes or logical I/O's per execute
- High logical I/O count means normally inefficient SQL

# From DUAL

---

- Many selects from DUAL
  - CURVAL, NEXTVAL from SEQUENCES
  - PL/SQL related functions
  - SYSDATE
- Interesting constructs
  - ... where rownum = 1

## Common Block and Buffer cache sizes

---

- `Db_block_size`
  - 8192 most common,
- `Db_block_buffers/db_cache_size`
  - `Db_block_buffers` = 7,100,000 (8K)

# From DUAL

---

- Many selects from DUAL
  - CURVAL, NEXTVAL from SEQUENCES
  - PL/SQL related functions
  - SYSDATE
- Interesting constructs
  - ... where rownum = 1

## Symptom 2: Too Slow Physical I/O

---

- Disk capacity increases rapidly
- Number of random I/O operations doesn't increase as quickly
- Full seek time is biggest component
- Prices of disks keep dropping
- Disk Array with large Cache

## Symptom 2: Too Slow Physical I/O

---

- More cost effective to buy large disks
- So database size roughly decides the number of disks needed
- OLTP databases should be sized based on the number of reads/writes per transaction
- Total number of reads + writes will decide the number of disks needed

## Symptom 2: Too Slow Physical I/O

---

- Heavily accessed data files should be striped over enough disks
- Caching data is important, but where
  - Oracle buffer cache
  - File System buffer cache
  - Disk array cache

# SAN has a big impact on performance

---

- Following example is from Hennessy and Patterson
  - Ultra3 SCSI controller 0.3 msec overhead, 160 MB/sec
  - Consider Oracle I/O
    - Single block reads (4K, 8K, 16K)
    - Multiblock reads (32K, 64K, 128)
    - Single block writes (4K, 8K, 16K)
    - Multiblock writes (32K, 64K, 128K)

# How many IOPS can we do?

---

- How many I/O can we do?
  - Transfer time
    - 4K / 160MB = 0.025 msec
    - 8K / 160MB = 0.05 msec
    - 16K / 160MB = 0.1 msec
    - 32K / 160MB = 0.2 msec
    - 64K / 160MB = 0.4 msec
  - Controller Overhead + Transfer Time
    - 64K  $1/(0.3 + 0.4)\text{ms} = 1428$  IOPS
    - 32K  $1/(0.3 + 0.2)\text{ms} = 2000$  IOPS
    - 16K  $1/(0.3 + 0.1)\text{ms} = 2500$  IOPS
    - 8K  $1/(0.3 + 0.05)\text{ms} = 2857$  IOPS
    - 4K  $1/(0.3 + 0.025)\text{ms} = 3076$  IOPS

## How many IOPS can we do?

---

- The number of IOPS per controller varies depending on the I/O block size, between 1400 to 3100.
- This means that the SCSI bus is only utilized for
  - $1400 * 64K = 87.5 \text{ MB} / 160 \text{ MB} = 54.68\%$
  - $3100 * 4K = 12.1 \text{ MB} / 160 \text{ MB} = 7.56 \%$
  - (Ignoring SCSI protocol overhead)

# How many I/O can a disk do?

---

- Average Time for a disk I/O:
  - 15000 RPM
  - 5 ms average seek time
  - 40 MB/sec transfer rate
- I/O time is
  - $5 \text{ ms} + (\text{rotational delay})/15000 + (\text{blocksize}/(40 \text{ MB/sec}))$
  - $5 \text{ ms} + 0.5 / 15000 + 32\text{K}/40\text{MB} = 7.8 \text{ msec}$
  - $1 / 7.8 \text{ ms} = 128 \text{ IOPS}$
- Example from page 745, Computer Architecture: A Quantitative Approach, 3<sup>rd</sup> Edition

# Different block sizes

---

- 4K (1/7.1 ms = 140 IOPS)
  - $5\text{ms} + (0.5/15000\text{RPM}) + 4\text{K}/40\text{MB} = 5 + 2 + 0.1 = 7.1$
- 8k (1/7.2 ms = 139 IOPS)
  - $5\text{ms} + (0.5/15000\text{RPM}) + 8\text{K}/40\text{MB} = 5 + 2 + 0.2 = 7.2$
- 16K (1/7.4 ms = 135 IOPS)
  - $5\text{ms} + (0.5/15000\text{RPM}) + 16\text{K}/40\text{MB} = 5 + 2 + 0.4 = 7.4$
- 32K (1/7.8 ms = 128 IOPS)
  - $5\text{ms} + (0.5/15000\text{RPM}) + 32\text{K}/40\text{MB} = 5 + 2 + 0.8 = 7.8$
- 64K (1/8.6 ms = 116 IOPS)
  - $5\text{ms} + (0.5/15000\text{RPM}) + 64\text{K}/40\text{MB} = 5 + 2 + 1.6 = 8.6$

# Optimizing I/O Response Times

---

- M/M/1
  - Server utilization = arrival rate x  $\text{Time}_{\text{Server}}$
  - $\text{Time}_{\text{System}} = \text{Time}_{\text{Server}} + \text{Time}_{\text{Queue}}$
- 8K blocksize (135 IOPS, 7.2 ms)
  - 135  $\Rightarrow$  240.0 ms
  - 105  $\Rightarrow$  29.5 ms
  - 75  $\Rightarrow$  15.7 ms
  - 45  $\Rightarrow$  10.6 ms

# Optimizing Response Time

---

- 64K blocksize (116 IOPS, 8.6 ms)
  - 135 => Not Possible
  - 105 => 88.6 ms
  - 75 => 24.6 ms
  - 45 => 14.6 ms

# Some Conclusions

---

- Seek Time still biggest component
  - 5 ms out of 7.1 to 8.6 (58 to 70 percent)
  - Disks should not be fully utilized, but may be only for 60 percent of the Maximum IOPS
    - This is of course application and end user dependent.
  - Buy more disks for IOPS not for storage
  - Like with all critical resources don't aim for 100 percent utilization!
    - IOPS
    - Storage

# Average Read Time

---

- Single block I/O
  - Average 1-10 msec
  - Maximum 500 msec
- Multi block I/O
  - Average 1-50 msec
  - Maximum 500+ msec
- Direct Path I/O
  - Average 1-50 msec
  - Maximum 500+ msec

## Symptom 3: Latch

---

- Synchronization means
  - Latch contention
  - Enqueue contention
- Always a symptom
- SLEEPS column in V\$LATCH indication of contention
- Generally Oracle will sleep 1 centi second
- Faster CPU means that we spend a lot of time waiting
- So either don't spin at all or spin more

# Examples of Slow Physical I/O

---

## Symptom 3: Latch contention

---

- Every logical I/O will result in a latch operation
- Cache buffer chain latch get
  - So many latch 'gets' and 'sleeps' on this
- Latch could be a symptom of too many buffer gets (logical I/Os)
  - Symptom of inefficient SQL

# Symptom 3: Latch contention

---

- For Example Symptom of
  - Hard parsing (not sharing SQL)
  - Soft parsing (re-parsing the same SQL statements over and over again)
    - Session\_cached\_cursors init.ora parameter
  - Too many Logical I/Os
- Hard parsing
  - Cursor\_sharing init.ora parameter (> 8.1.5)
  - Exact, force, similar

## Symptom 3: Enqueue contention

---

- Sequence problems
  - SQ enqueue
  - Need to increase cache size or check for sequences with order option
- High water mark problems
  - HW enqueue
  - Add free lists to the object

# Lock Contention Examples

---

# Some interesting statistics

---

- Shared Pool Size

- Largest shared\_pool\_size: 44795166720 bytes

- Yes, 41.7 GB

- Oracle9 Release 2
    - Redo/sec 1.1 MByte, 6.6 KByte/tx, 192 TX/sec
    - Approx. 25000 executes/sec, approx. 3800 parses/sec
    - Approx. 5700 blocks read/sec
    - Approx. 2700 blocks writes/sec
    - Approx. 50% of elapse time is “latch free” wait mostly for “library cache latch”
    - 8K blocksize, 790 Mbyte buffer cache

# Conclusions

---

- Many databases have the same mistakes and problems
  - This points to a fundamental problem
- CPUs are getting faster and will help to hide a lot of the bad SQL problems
- IOPS are growing not that fast, and SANs are getting more and more popular
  - IO will be the performance problem of the future
- Tuning SQL is always a good thing 😊

## References

---

- Computer Architecture: A Quantitative Approach, John L. Hennessy & David A. Patterson, ISBN 1558607242
- Sane SAN, James Morle, Scalabilities, <http://www.scaleabilities.co.uk>
- Oaktable (<http://www.oaktable.net>)
- Jonathan Lewis (<http://jlcomp.demon.co.uk>)
- Tuning I/O for Better Performance, Wei Wuhan, <http://www.oraperf.com>

## Q&A

---

Please don't forget to fill out the review forms

This is session #1493