Scripts and Tools for Oracle Troubleshooting and Advanced Performance Analysis

Tanel Pöder

http://www.enkitec.com

http://blog.tanelpoder.com



Intro: About me



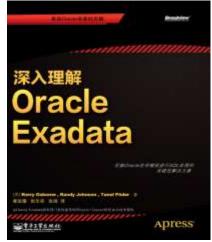


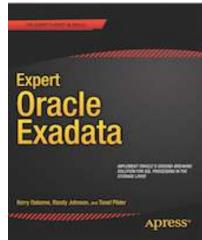


Tanel Pöder

Oracle Database Performance geek
Exadata Performance geek
Hadoop Performance geek

- Enkitec
 - Consultant
 - Researcher
 - Technology Evangelist







book (with Kerry Osborne and

Randy Johnson of Enkitec)



All scripts and tools are downloadable and free

- Download the scripts and tools from:
 - http://blog.tanelpoder.com/files/

- Updates (since I delivered this class in 2010):
 - I have shut the PlanViz website down so ignore some slides
 - I haven't updated the PerfSheet v3 tool since 2010
 - Luca Canali has built a new PerfSheet v4:
 - http://canali.web.cern.ch/canali/resources.htm



Intro: About this Training Day - 1

- 1. Your Performance
- Database Performance
- SQL Performance
- 4. Performance visualization, trending and advanced analysis
- We will talk about scripts, tools and a systematic approach to troubleshooting & tuning
 - Case studies!
- Download the scripts and tools from:
 - http://tech.e2sn.com/oracle-scripts-and-tools



Intro: About this Training Day - 2

- Productivity and speed of action
 - Both on Unix and Windows clients
- Scripts and Tools
 - Snapper v3.0
 - MOATS v1.0
 - LatchProf v2.0
 - And many more...:-)
- Trending & Visual Performance Analysis
 - Tools
 - Data Collection (even without Diag Pack licenses)
 - Visualization PerfSheet v3.0



Productivity and speed of action

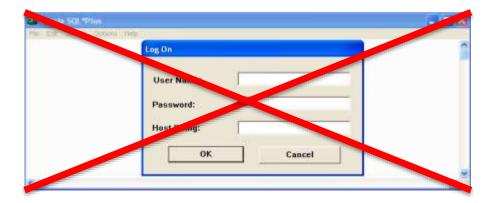
- Talking to Oracle
 - SQLPLUS still rocks [©]
 - ...especially in cases where you need to deviate from the usual path
 - You'll have a (searchable) history of output of your last commands
 - On the screen or in a spool/logfile...
 - You can run your scripts
 - It's more "pure" than GUI development tools (which try to be clever, do extra stuff, don't fetch all data etc)

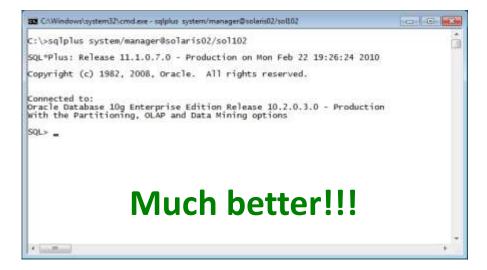


SQLPLUS on Windows

Configuration

- Do NOT use sqlplusw.exe ->
- This is what you get when running sqlplus from Start menu
- Luckily sqlplusw.exe is gone since 11g!
- Use CMD.exe and run sqlplus.exe ->



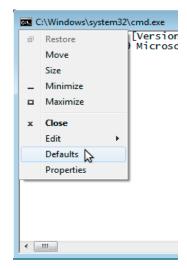


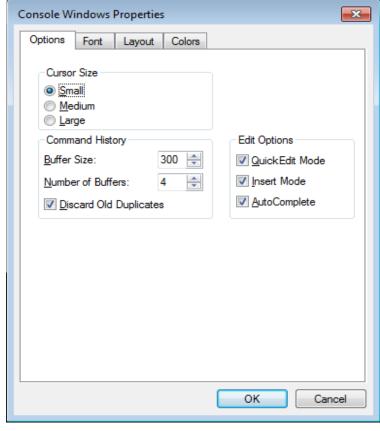


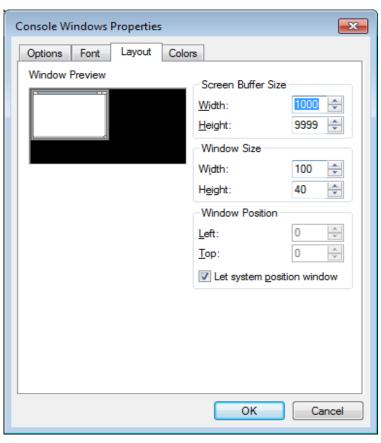
SQLPLUS on Windows – CMD.exe settings

- CMD.exe needs some tweaking...
 - You should check QuickEdit mode, Increase Screen Buffer size etc

1. 2. 3.





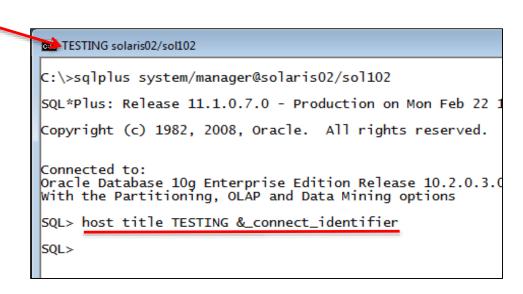




www.enkitec.com

Setting SQLPLUS Window title automatically

- Set cmd window title to include database session info
 - SQL> host title "test"
 - @i.sql queries session, instance info from V\$ views and sets title accordingly
 - i.sql is automatically called by login.sql if its in SQLPATH directory





www.enkitec.com 9

SQLPLUS on Windows

- Command line history
 - Up/Down arrow navigate between commands
 - CTRL+Left/Right arrow jump one word left/right
 - F7 Display a popup box with command line history
 - Use up/down arrows to navigate
 - ENTER to execute command
 - Right arrow to put the command to command line for editing
 - First letter of command to jump between commands starting with that letter (case sensitive)
 - F8 Retrieve command from history
 - Type a beginning of the command (for example "sel") and press F8
 - This will jump through all commands starting with "sel" (case sensitive)
 - List commands from history
 - SQL> host doskey /history /exename=sqlplus.exe
 - @h <text> history script showing old commands with text int them
 - @ha history all



www.enkitec.com

SQLPLUS on Windows

- Output formatting
 - @HTML
 - Uses SET MARKUP ON HTML
 - Runs the query in SQLPLUS buffer and spools the output to .HTML file
 - Launches that .HTML file with appropriate OS command
 - @XLS
 - Same as above, but spools to an .XLS file and launches it :-)
 - @CSV
 - Uses SET COLSEP "," and spools to a .CSV file



www.enkitec.com

SQLPLUS on Unix / Linux / MacOS

Configuration

- Setting xterm title
 - From shell
 - echo -ne "\033]0; your text here \007"
 - From SQLPLUS
 - host echo -ne "\033]0; your text here \007"
 - From vi
 - set title
 - set titlelen=20
- Decent terminal emulators
 - Putty on Windows (small, lightweight and works well!)
 - http://www.chiark.greenend.org.uk/~sgtatham/putty/
 - Terminator on MacOS, Unix, Linux (and apparently Windows)
 - http://software.jessies.org/terminator/



www.enkitec.com

SQLPLUS on Unix / Linux / MacOS

- Output formatting
 - Same as on Windows
 - Launching the HTML file is done with a different command:
 - MacOS open command launches the program associated with .HTML extension
 - Unix/Linux I just run firefox by default, but you can change it in the script
- Terminal with left-right scrolling capability?
 - I'm using Terminator
 - (Not the one you can install from Ubuntu repositories)
 - http://software.jessies.org/terminator/
 - However, it needs to install its own terminfo file to function properly!
 - This will be an issue if you need to SSH into a lot of servers in an enterprise
 - Good for demoing stuff though! ⁽²⁾
- SQL> set truncate on
 - Sqlplus allows you to truncate the columns which don't fit into the linesize instead of wrapping



www.enkitec.com

SQLPLUS on Unix / Linux / MacOS X

- Command line history using RLWRAP
 - Rlwrap (readline wrapper) adds GNU readline capability for any text mode application
 - BASH shell has the same functionality built in
 - Just run rlwrap sqlplus instead of just sqlplus (or any other utility)
 - Rlwrap gives also tab completion!
 - If you tell it to...
 - It can use a wordfile (with Oracle keywords, view, column names)
 - rlwrap -f
 - And it can remember every word it sees on the screen!
 - rlwrap -r



www.enkitec.com

SQLPLUS with RLWRAP – Keyboard shortcuts

- Keyboard shortcuts (Emacs mode):
 - Up/down arrows navigate command line history
 - CTRL+A go to beginning of line
 - CTRL+E go to end of line
 - CTRL+R search command history backwards
 - press CTRL+R again to move on to next match
 - CTRL+S search command history forward
 - go "back" from too many CTRL+R's
 - ALT+F / ALT+B Move Forward / Backward by one word
 - This works if ALT is used as the META key for emacs (check your terminal emulator preferences)
 - CTRL+K erase (kill) line from current position to the end of line
 - CTRL+X, BACKSPACE erase line from current position to the beginning
 - CTRL+W or ALT+BACKSPACE erase one word at a time
 - CTRL + _ undo command edit! (or CTRL+X, u)



www.enkitec.com

Installing RLWRAP

- Linux
 - RHEL/OEL/CentOS Get from here: http://fedoraproject.org/wiki/EPEL
 - Ubuntu: apt-get install rlwrap
- MacOS X Get it from an open source package site for MacOS
 - MacPorts http://www.macports.org/
 - DarwinPorts http://darwinports.com/
 - Fink http://www.finkproject.org/
- Solaris (SPARC and Intel)
 - http://www.sunfreeware.com/
- On HP-UX there's a similar built in command: ied
- Should also work on Windows with Cygwin
 - http://blog.tanelpoder.com/2010/05/07/flexible-sqlplus-command-line-history-with-rlwrap/



www.enkitec.com

Setting up TPT scripts - 1

- Tanel's Performance Tuning scripts :-)
 - 500+ scripts which I use "every day"
 - Most scripts do one thing only (such as show parameter values etc)
 - Also latest versions of the more sophisticated scripts (Snapper, LatchProfX etc) are in there
 - Download the .zip file and unzip to a directory:
 - http://tech.e2sn.com/oracle-scripts-and-tools
- Set SQLPATH variable to point to the script directory
 - Unix:
 - export SQLPATH=\$HOME/tpt
 - Windows:
 - set SQLPATH=C:\users\tanel\tpt
 - Or Right click on My Computer -> Properties -> Advanced Settings -> Environment Variables; and add it there



<u>www.enkitec.com</u> 17

Setting up TPT scripts - 2

- Edit init.sql (init.sql sets up various sqlplus variables)
 - Uncomment the appropriate parameters in the SETUP section
 - It tells sqlplus whether you're running it on Windows, Unix or MacOS

```
-- SETUP SECTION -
-- Uncomment the following lines if you run your sqlplus on Windows
-- def SQLPATH=%SQLPATH%
-- def _start=start
-- def _delete=del
-- host doskey /exename=sqlplus.exe desc=set lines 80 sqlprompt ""$Tdescribe $*$Tset lines 299 sqlprompt "SQL> "
```

- Edit **i.sql** (i.sql is the "who am I" script which also sets window title)
 - Uncomment the appropriate title command in the setup section (EOF)

```
host title &_i_user@&_i_conn [sid=&mysid ser#=&_i_serial spid=&_i_spid ... -- host echo -ne "\033]0;&_i_user@&_i_inst sid=&mysid,&_i_serial\007"
```



www.enkitec.com

Database Performance Tools



What is Oracle Database activity about?

1. A bunch of *sessions*

You need a session for doing anything in Oracle

1. Applications use these sessions to execute (PL)SQL cursors

- Any database call requires use of a cursor
- Including PL/SQL stored procedure calls

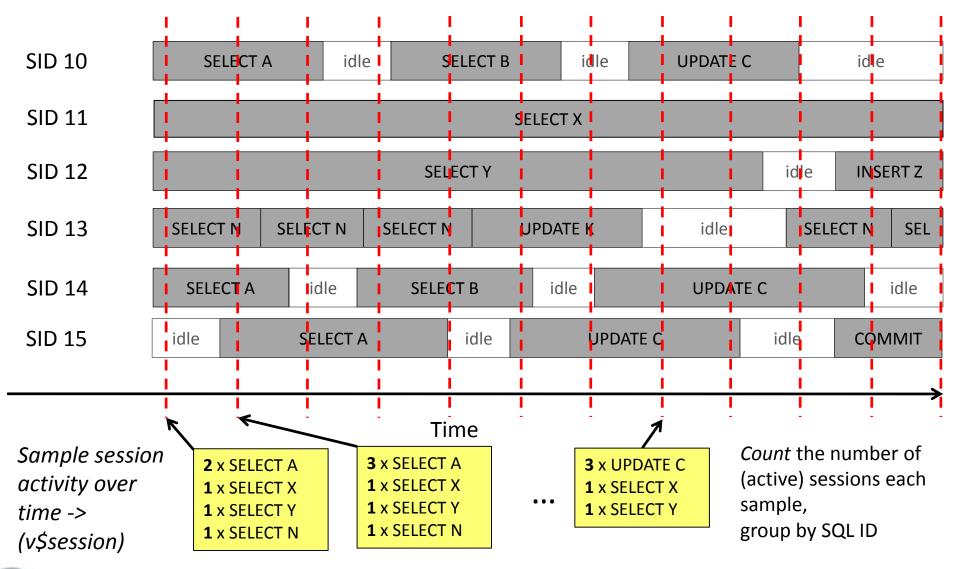
The cursor execution will do some work

- During the execution, the processes either:
 - Work on CPU
 - Or at least want to work on CPU, but OS scheduler doesn't allow them
 - Do not Work on CPU
 - They have been put to sleep by OS because of a blocking system call
 - They have gone to sleep voluntarily, waiting for some event completion



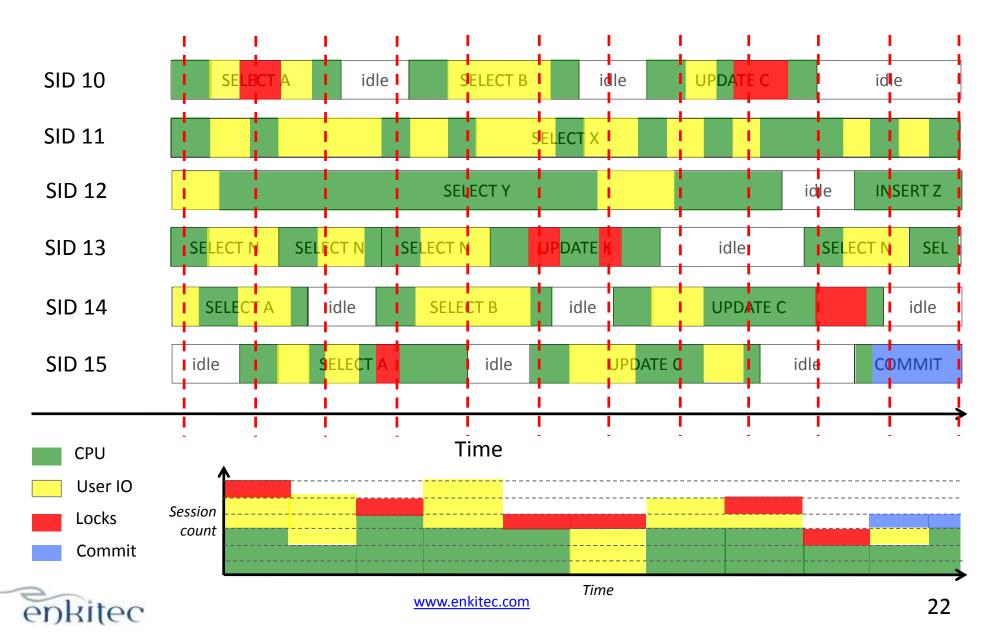
<u>www.enkitec.com</u>

Sampling session activity details over time – Which SQL?





Sampling session activity – Working or Waiting?



Understanding Database Performance

Performance diagnosis data should have at least these details

Which Sessions? -> v\$session.sid

Which SQL? -> v\$session.sql_id, plsql_(sub)program_id

• Work or wait? -> v\$session.state, v\$session.event, p1,p2,p3

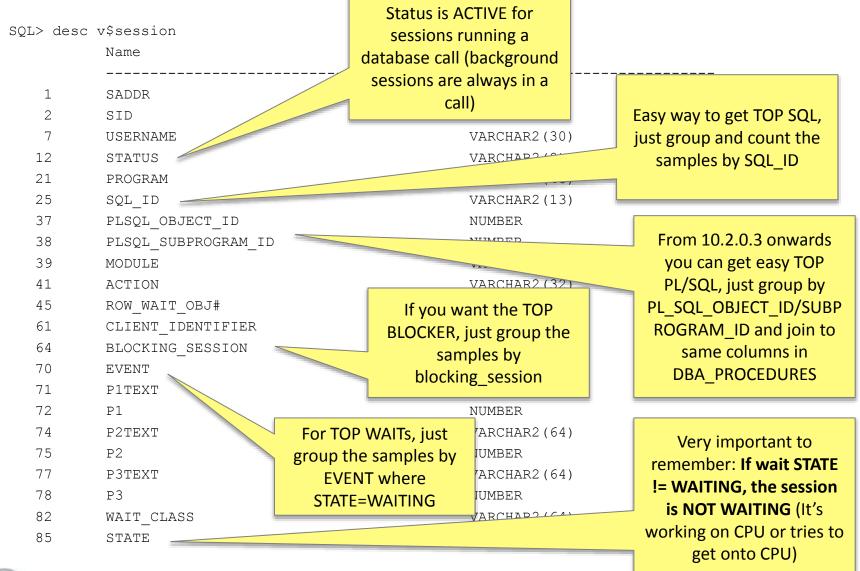
When? -> Time when samples were taken

- This data can be grouped and summarized by any dimension depending on the "performance question" asked!
 - The sampled session activity dataset is like a FACT table in DW's
 - The existence of a session sample shows that the session was active
 - The dimension attributes show a large variety of extra details
 - Which SQL was running, Working or Waiting, Which event, P1,P2...



www.enkitec.com

A selective list of dimensions for performance analysis





Instance Activity overview

- What is the database (instance) doing right now?
 - @a.sql Instance Activity SQL
 - It takes one ASH style sample of V\$SESSION and shows TOP sql/wait
 - I wrote it to show how easy it is to get an overview of instance activity
 - But it's also very short to type I actually use it regularly :-)
 - I just run it couple of times in a row!

```
select
    count(*)
  , sql id
  , case state when 'WAITING' then 'WAITING' else 'ON CPU' end state
  , case state when 'WAITING' then event else 'On CPU / runqueue' end event
from
    v$session
where
    status='ACTIVE'
and wait class != 'Idle'
and sid != (select sid from v$mystat where rownum=1)
group by
    sql id
  , case state when 'WAITING' then 'WAITING' else 'ON CPU' end
  , case state when 'WAITING' then event else 'On CPU / runqueue' end
order by
    count(*) desc
```



<u>www.enkitec.com</u>

Snapper v3 - Overview

- Major improvement ASH style session activity sampling!
 - From V\$SESSION / V\$SESSION_WAIT (no separate licenses needed)
 - @snapper ash <total_seconds> <sampling_interval> <SIDs>
- Still works all the way from Oracle 9.2 to latest Oracle version
- Still NO installation/database changes are needed!
 - It's just an anonymous PL/SQL block, parsed on the fly...
 - Look into the code to see some cool challenges involved!!!
- Still free and still has a picture of a real Snapper (fish) in it! ;-)
 - ...and you can still run it directly from web if you feel adventurous
 - SQL> @http://www.tanelpoder.com/files/scripts/snapper.sql



<u>www.enkitec.com</u>

Snapper v3 – Session Activity Sampling

Example 1a: Single Session troubleshooting:

```
SQL> @snapper ash, gather=n 5 1 32
-- Session Snapper v3.01 by Tanel Poder ( http://www.tanelpoder.com )
  -- Act% | Wait Event
                                 | Wait Class
                                                          | SQL ID
   84% | db file scattered read | User I/O
                                                         g723q6scvr9h8
                                                          | q723q6scvr9h8
   11% | ON CPU
                                | ON CPU
    3% | db file sequential read | User I/O
         ON CPU
                                 I ON CPU
                                                          | c0j6cx9kzjf7q
    3% |
                                           As we measure a single
SQL> @sqlid q723q6scvr9h8
                                           session, then the max
                                         Activity% can be 100% (allow
HASH VALUE CH# SQL TEXT
                                          for some rounding errors).
             0 /* SQL Analyze(0) */ sele
 431728136
                                                                 (t)
                                         If the activity is measured less
               no parallel index(t) dbms
                                                                 t use weak name resl
                                         than 100%, it means that the
               dynamic sampling(0) no mo
                                                                 to char(count
                                          session was not active the
                                                                 0009$"),16,0,32),
               ("SYS NC00009$")), to char
                                         remaining time (it was idle).
```



www.enkitec.com

Snapper v3 – Session Activity and Performance Counters

Example 1b: Single Session troubleshooting with perf counters

```
SQL> @snapper ash, stats 5 1 146
-- Session Snapper v3.01 by Tanel Poder ( http://www.tanelpoder.com )
SID, USERNAME , TYPE, STATISTIC
                                                          DELTA, HDELTA/SEC, %TIME,
146, SYS
          , STAT, session logical reads
                                                          309970, 61.99k,
                                                          309966, 61.99k,
           , STAT, consistent gets
146, SYS
                                                          309967, 61.99k,
           , STAT, consistent gets from cache
146, SYS
                                                          309968, 61.99k,
146, SYS
           , STAT, consistent gets from cache (fastpath)
146, SYS
           , STAT, calls to kcmgcs
                                                           1660,
                                                                    332,
                                                         308388, 61.68k,
           , STAT, no work - consistent read gets
146, SYS
146, SYS
           , STAT, table scans (short tables)
                                                           332,
                                                                    66.4,
           , STAT, table scan rows gotten
146, SYS
                                                         25006755,
                                                                      5M,
                                                          308415, 61.68k,
           , STAT, table scan blocks gotten
146, SYS
146, SYS
           , TIME, DB CPU
                                                          3960000,
                                                                    792ms,
                                                                              79.2%
                                                          4059947, 811.99ms,
           , TIME, sql execute elapsed time
146, SYS
                                                                              81.2%
           , TIME, DB time
146, SYS
                                                          4059947, 811.99ms, 81.2%
-- End of snap 1, end=2010-03-01 11:31:10, seconds=5
------ Sampled Session Activity Profile ------
-- Act% | Wait Event
                           | Wait Class
                   | ON CPU
  100% | ON CPU
                                         | 7vxd09448zcrx
```



www.enkitec.com

Snapper v3 – Session Activity Sampling

- Example 2: Whole instance overview
 - You can write any query instead of the list of SIDs:

```
SQL> @snapper ash, stats, gather=n 30 1 "select sid from v$session"
-- Session Snapper v3.01 by Tanel Poder (http://www.tanelpoder.com)
  ------ Sampled Session Activity Profile ------
-- Act% | Wait Event
                                     | Wait Class
                                                                  | SQL ID
                                                                 | 3jbwa65aqmkvm
   164% | read by other session
                                     | User I/O
    99% | ON CPU
                                                                   7vxd09448zcrx
                                     I ON CPU
    99% | ON CPU
                                     I ON CPU
                                                                   bt4txmkqf7u89
    98% | db file scattered read
                                   | User I/O
                                                                   3jbwa65aqmkvm
                                                                   3jbwa65aqmkvm
    98% | direct path read
                                    | User I/O
    61% | db file scattered read
                                                                   dw2zgaapax1sq
                                   | User I/O
    41% | control file parallel wri | System I/O
                                      Systam I/A
    34% | db file async I/O submit
    27% | log file parallel write
                                          As we are measuring more
    24% | db file sequential read
                                                                   Obzhqhhj9mpaa
                                          than one session, then it's ok
                                          to see over 100% of activity.
                                          100% stands for all response
                                           time of a single session!
```



www.enkitec.com 29

Snapper v3 – Performance Counters

Example 3a - Find top REDO geneators

```
SQL> @snapper ash, stats, gather=s, sinclude=redo size 5 1 "select sid from v$session"
-- Session Snapper v3.01 by Tanel Poder ( http://www.tanelpoder.com )
  SID, USERNAME , TYPE, STATISTIC
                                                  , DELTA, HDELTA/SEC,
   20, SYS , STAT, redo size
                                                       13376076, 2.68M,
                                                  , 18232, 3.65k,
   31, SYS , STAT, redo size
  143, SYS , STAT, redo size
                                                         26796, 5.36k,
-- End of snap 1, end=2010-03-01 11:52:47, seconds=5
 -- Act% | Wait Event | Wait Class
                                               | SQL_ID
   69% | log file parallel write | System I/O
   50% | ON CPU
                         I ON CPU
                                             | crwqs334cz65x
                                               | crwqs334cz65x
  27% | log buffer space | Configuration
  12% | log buffer space | Configuration
                                           | f6f6wxb0st0yp
             I ON CPU
   8% | ON CPU
   8% | db file sequential read | User I/O
                                               | gvt8zu1k1tsff
```



www.enkitec.com 30

Snapper v3 – Performance Counters

- Example 3b:
 - Find top IO throughput (MB) users

```
SQL> @snapper stats, gather=s, sinclude=physical%bytes 5 1 "select sid from v$session"
-- Session Snapper v3.01 by Tanel Poder ( http://www.tanelpoder.com )
   SID, USERNAME , TYPE, STATISTIC
             _____
    6, (CKPT) , STAT, physical read total bytes
                                                              16384, 3.28k,
                                                             32768, 6.55k,
    6, (CKPT) , STAT, physical write total bytes
    6, (CKPT) , STAT, cell physical IO interconnect bytes , 49152, 9.83k,
                                                             29360128, 5.87M,
    20, SYS , STAT, physical read total bytes
   20, SYS , STAT, cell physical IO interconnect bytes
                                                             29360128,
                                                                        5.87M,
                                                             29360128,
   20, SYS
              , STAT, physical read bytes
                                                                        5.87M,
   139, SOE
             , STAT, physical read total bytes
                                                                       367k,
                                                             1835008,
   139, SOE , STAT, cell physical IO interconnect bytes
                                                             1835008, 367k,
   139, SOE , STAT, physical read bytes
                                                             1835008, 367k,
   End of snap 1, end=2010-03-01 12:00:35, seconds=5
```

- You can use the same trick with any performance counter
 - IOPS @snappiops
 - CPU @snapcpu



Snapper v3 – Write output to a tracefile

Example 4:

- SQL> @snapper ash,trace 30 999999 all
- Uses DBMS_SYSTEM.KSDWRT() for writing to the tracefile
- In above example, a Snapper snapshot is written to a tracefile every 30 seconds (and 999999 times in a row unless you cancel)
- You can run tail -f in OS for realtime Snapper output monitoring
- "all" option instead of a SID stands for all sessions in the database
- Note that gathering V\$SESSTAT stats for a lot of sessions frequently is CPUexpensive
 - That's why the "all" option disables V\$SESSTAT gathering in Snapper v3.0 by default
 - You can still enable it by adding the gather=... option to collect the stats you want (read the script header for documentation)



www.enkitec.com

Snapper v3 – Advanced Performance Counters

- Over 600 V\$SESSTAT performance counters in Oracle 11.2
 - One separate set for each session!
- Where are the performance counters documented?
 - Oracle Database Reference "Statistics Descriptions"
 - http://download.oracle.com/docs/cd/E14072 01/server.112/e10820/stats002.htm
 - Note that the above URL may change when Oracle releases new docs
 - tech.E2SN:
 - http://tech.e2sn.com/oracle/performance/reference
 - I'll be updating this one over time...



www.enkitec.com

Case study – Bad index build performance - 1

- An index build was taking too long time
 - Function based index build on a busy table
 - Build, not re-build
 - Ran with ONLINE option
 - Executed in parallel
- The parallel slaves were doing the full table scan
 - But they were waiting for db file sequential read for over 80% of their time
 - Why would be the parallel slaves (which did a full table scan) be doing single block reads so much?



Case study – Bad index build performance - 2

```
Total single block IO requests
-- SID, TYPE, STATISTIC
                                                                                    SEC
                                                               total IO requests -
                                                            total multi block requests =
  3768, STAT, session logical reads
                                                                                     76
                                                                 246 – 51 = 195
  3768, STAT, physical read total IO requests
                                                                                     . 35
  3768, STAT, physical read total multi block requests,
                                                                       877,
                                                                                  51.59
  3768, STAT, physical read total bytes
                                                                                  4.35M
  3768, STAT, consistent gets
                                                                                 679.94
                                                physical reads is a misleading
  3768, STAT, consistent gets from cache
                                                                                 679.76
                                                name – it's really the number
  3768, STAT, consistent gets - examinati
                                                                                  36.18
                                                   of blocks read via PIOs
  3768, STAT, physical reads
                                                                                 528.71
  3768, STAT, physical reads cache
                                                                                 528.71
                                                                      8988,
                                                                                 246.88
  3768, STAT, physical read IO requests
                                                                      4197,
  3768, STAT, physical read bytes
                                                                  73637888.
                                                                                  4.33M
  3768, STAT, physical reads cache prefetch
                                                                                 279.53
                                                       single block reads due
  3768, STAT, table scan rows gotten
                                                                                  1.65k
                                                      chained / migrated rows!
  3768, STAT, table scan blocks gotten
                                                                                 435.24
                                                                      3335,
                                                                                 196.18
  3768, STAT, table fetch continued row
                                                                                  11.29
  3768, STAT, PX local messages sent
                                                                       192,
  3768, WAIT, db file sequential read
                                                                  14393221,
                                                                              846.66ms
  3768, WAIT, db file scattered read
                                                                  1040277,
                                                                                61.19ms
```



Snapper v3 - Summary

 Oracle Session Snapper is a powerful ad-hoc session-level performance diagnosis tool which does not require any installation or database changes!

Features:

- ASH style session activity sampling
- Reports top SQL, top Waits, etc
- Very useful for drilling down to details when the plain wait interface and SQL trace do not give enough details
- Takes snapshots of V\$ views (V\$SESSTAT, etc) and reports deltas
 - Report all stats of some sessions (like the advanced diagnosis example)
 - Report some stats of all sessions (like the redo or IO example)
- Flexible!



Beyond Oracle's instrumentation

- V\$ views, ASH, SQL trace etc will tell you what Oracle thinks is happening
 - They all are based on Oracle's built-in instrumentation...
 - ...Which isn't measuring everything and may have bugs
 - If Oracle's instrumentation lies, all tools on top of that lie



Oracle troubleshooting "drilldown" sequence

- 1. Which sessions, what SQL, wait or not?
 - V\$SESSION, ASH, SQL Trace, Snapper
 - Where is the response time spent?
- 2. Which performance counters increase?
 - V\$SESSTAT, Snapper
 - What kind of work Oracle reports to be doing internally?
- 3. Which Oracle kernel functions are executed
 - pstack, DTrace, os_watcher
 - Exactly what work is Oracle doing internally?



- 1. DW on Oracle 10.1.0.3 a few years ago...
- 2. Inserts in ETL process *occasionally* very slow
 - The wait interface showed log file sync as the main wait event!
 - Why would plain INSERT statements wait for log file sync???
- 3. Perhaps some recursive (autonomous) transactions committing?
 - There were recursive calls because of triggers, but...
 - ...I ran Snapper on the session user commits was not increasing!
 - (...and triggers running in the context of parent transaction can't commit anyway)
- 4. I went through the stats which did increment in V\$SESSTAT
 - Incrementing (or decrementing) stats show what kind of operations Oracle is doing (and how many of them)



39

 SID,	TYPE,	STATISTIC		,	DEL	 тА,	HDELTA/S
 319,	STAT,	recursive calls		,	2	08,	69.33
319,	STAT,	session logical reads		,	40	02,	1.33k
319,	STAT,	db block gets				59,	1.12k
319,	STAT,	db block gets from cache	86 transa	action ro	llbacks	59,	1.12k
319,	STAT,	consistent gets	per se	cond, bu	ıt no	43,	214.33
319,	STAT,	consistent gets from cache	user	rollback	s!!!	43,	214.33
319,	STAT,	consistent gets - examination				31,	177
319,	STAT,	db block changes	7	,	25	03,	834.33
319,	STAT,	redo synch writes		,	2	57,	85.67
319,	STAT,	redo entries		,	12	66,	422
319,	STAT,	redo size		,	4011	84,	133.73k
319,	STAT,	redo ordering marks		,	2	13,	71
319,	STAT,	undo change vector size		,	1223	76,	40.79k
319,	STAT,	rollback changes - undo refords	applied	,	2	57,	85.67
319,	STAT,	transaction rollbacks		,	2	57,	85.67
319,	STAT,	commit txn count during cleanout		,		3,	1
319,	STAT,	active txn count during cleanout		,	2	71,	90.33
319,	STAT,	cleanout - number of ktugct call	S	,	2	68,	89.33
319,	STAT,	execute count		,	2	08,	69.33
319,	WAIT,	log file sync		,	23447	49,	781.58ms
319,	WAIT,	db file sequential read		,	283	94,	9.46ms
 End	of sna	ap 1					



- Metric definitions from "Statistics Descriptions" @ Reference Guide:
 - transaction rollbacks
 - Number of transactions being successfully rolled back
 - user rollbacks
 - Number of times users manually issue the ROLLBACK statement or an error occurs during a user's transactions
- Time for a search in Metalink:
 - I searched for "transaction rollbacks user rollbacks log file sync"
 - ASSM Bug # 4664513 Slow Insert Waiting For 'log File Sync' High 'transaction Rollback' Stat (Base bug# 4288876)
 - ASSM space manager failed to find free space in a block thought to be empty enough and had to roll back its internal transaction
 - Rollbacks also wait for log file sync!!!



- Long term solution was to patch the database
- Short term workaround was to rebuild the fact table indexes to FREELIST managed tablespaces
 - We started from the latest partitions only as these were where the most inserts were done

- Note that even though this time the V\$SESSTAT stats gave us a good match in Metalink, I wanted to be sure:
 - And I looked into the ultimate (but microscopic) source of information
 - The process stack!
 - Remember, Oracle is just a program, written in C, so whatever it is doing, it
 must be doing it so because of some C function in its kernel



Read Metalink note# 175982.1 for more info about Oracle kernel modules and functions

```
$ pstack 26490
                                   semsys – sleep and wake
                                                        ROTOCOL=beq)))
26490: oracleXYZ (DESCRIPTION=
                                   up when someone posts
 ffffffff7c9a5288 semsys
                                     us via a semaphore
                                                            kslwait is used for voluntarily going
 fffffffffc9a071c semtimedop
                                                             to sleep waiting for someone to
 000000102f78238 sskgpwwait (fffffffffffdd910, f423f,
                                                            wake us up. Event 0x64 was log file
 000000100afece0 ksliwat (10506aae0, 87f323ce8, 0,
                                                                       sync in
 000000100aff7a0 kslwaitns (64, 1, 121, 202coc
                                                              v$event name in that version
 0000000100c19198 kskthbwt (64, 1, 121 JCCC9, d76, 0) + 98
 0000000100aff75c kslwait (64, 121, 202cc9, 7e7, 908, 0) + 5
                                                                   kcrf commit force is
 0000001003d6c4c kcrf commit force (38<del>0016a13,</del>
                                                                 actually used for sync'ing
 000000100e86fd0 kcbchq1 (100e84, 100e89000, dc7c614667b50,
                                                                    both commits and
 000000100e84680 kcbchg (ffffffffffffde490, 800000000, a0dd0
                                                                       rollbacks!
 0000000100a261c8 ktucmt (874a769f0, ffffffffffb9d1ea8, 0, 0, 0, 3) + 5a8
 0000001009f8728 ktuabt (1, 2 1 1 874-7605)
                                                   Kernel Transaction Undo
 0000001009bfb88 ktcrab (874a769f0, 0, 200000)
                                                                        + 128
                                                     Abort transaction
 000000101ec4984 kdisdelete (0, 1131, 104c00,
                                                                        f20, 0) + 44
 000000101e8ea64 kdisnew bseq srch cbk (ffffffttt/tte/368
                                                              Kernel Transaction Space
 000000010067dbe0 ktspfpblk (ffffffff
                                                              2378, 3, ffffffff7b9d
                                         scan BitMap Block – this
 000000010067e160 ktspfsrch (5, 3b45ba
                                                              20, fffffffffffe02a8,
                                            function was also
 000000100678bb8 ktspscan bmb (9 9
                                                               19, 10) + b8
                                            mentioned in the
 0000001006770b4 ktspgsp cbk (0, ec
                                                               1) + 174
                                             Metalink note
 000000101e8f890 kdisnew (101e8d,
                                                               101e8d000, 2, fffffff
```

```
0000000101e91aec kdisnewle (fffffffffb9d7fd8, 3b447592, fffffffffbb7a9b8, fffff
0000000101eb1d00 kdisle (b5, 0, 55, 0, 2, fffffffffffffe6dc0) + 1500
0) +
                                                       kdi = Kernel Data Index
000000101e2ca40 kdiins (8<del>31cde110, 0, 0, 111111111111</del>
                                                                            ff)
000000101cbe2cc kauxsin (105068000, 10506f, 105000,
                                                           management
                                                                            + 88
0000000102388d8c insidx (1, 0, 0, 0, 0, 0) + 2ec
000000102385660 insrow (8678f63c8, ffffffffffffffe40
                                                       40000681
                                                                 ffffffff7bb6eea
000000102384910 insdry (4f, 40000001,
                                                                      6460, 4000
00000001023839fc inscovexe (40000681, 1c01, 1050
                                                                      8, fffffff
                                                       insert a row
0000000102395710 insExecStmtExecIniEngine (86781
                                                                      ffff7fff76
0000000102384148 insexe (105074, 7f1, 2e5, 10506
                                                                      ffff7ba19e
000000101acf090 opiexe (e, fffffffffbc68ee8, ffffffffffffff6fb0, 0, 1, fffffffff
0000000101ad7be8 opipls (1, 20, fffin.
                                                                        ffffffff
0000001002d0058 opiodr (6, 10506ae10, 10434cm
                                                  Oracle Program Interface
                                                                      000) + 598
0000001002d4ec0 rpidrus (fffffffffff8810, 1050
                                                                      5cb288, 86
                                                     execute cursor
0000000102f615e4 skgmstack (fffffffffff8a38, ff
                                                                      fff7fff897
00000001002d504c rpidru (fffffffffffffffff9130, 10422b000, 10422a918, 104229598, 410
0000001002d4808 rpiswu2 (0, 10)
                                                                            ffff
                                                  Recursive Program Interface - set
...some functions snipped...
                                                     up a recursive call. This is
0000001002cc174 opidry (0, 4, 10506a, 10507145)
                                                    essentially what a "PL/SQL
0000001002c9828 sou2o (ffffffffffffe6a8, 3c, 4,
                                                   context switch" also means – a
                                                                            00)
0000001002a7b34 main (2, ffffffffffffe788, ffff
                                                         recursive call.
00000001002a7a7c start (0, 0, 0, 0, 0, 0) + 17d
```



MOATS v1.0 - Overview

- MOATS stands for the Mother Of All Tuning Scripts :-)
 - While Snapper is an ad-hoc, "no footprint", simple troubleshooting utility...
 - ...MOATS requires installation of a PL/SQL package and optionally tables
 - MOATS is (will be) more powerful
 - MOATS automatically displays related performance data
 - MOATS can be used for constant performance monitoring
 - MOATS runs on Oracle 10g+
 - C'mon people, 9i is ancient ;-)
 - Snapper still works on 9.2 too...



MOATS v1.0 – Overview

- Written by:
 - Adrian Billington a PL/SQL guru (http://www.oracle-developer.net)
 - And me an Oracle performance geek
 - Downloadable from: http://tech.e2sn.com/oracle-scripts-and-tools
- The 1st goal is to write a decent TOP utility for Oracle
 - Which should run inside Oracle of course I mean sqlplus!
 - Coming in v2.0:
 - Persist session activity samples
 - Yet-another-ASH clone based on V\$SESSION / X\$KSUSE (but better ;-)
 - Persistent V\$LATCHHOLDER data
 - Graphical visualization with PerfSheet



MOATS v1.0 – Oracle Top output

```
SQL> set arraysize 72
SQL> select * from table(moats.top);
MOATS: The Mother Of All Tuning Scripts v1.0 by Tanel Poder & Adrian Billington
    http://www.e2sn.com & http://www.oracle-developer.net
| Cur Time: 01-Mar 09:38:44 | Calls/s: 20.5 | hParse/s: 4.9 | PhyRD/s: 11.0 | Write MB/s: 0.3 |
| History: 0h 1m 17s | Commits/s: 0.4 | ccHits/s: 25.8 | PhyWR/s: 3.7 | Redo MB/s:
                                                                        0.0 1
| 100% | 3jbwa65aqmkvm (0) | 30,28,154,142,14 | | 80% | direct path write temp | User I/O |
                                       | 20% | log file parallel write | System I/O |
 40% | (0)
                                       | 20% | db file sequential read | User I/O
 20% | 13sp89mx21tab (0) | 32
 20% | cq8hzxwd7mk8d (0) | 143
                                  | 20% | control file parallel write | System I/O
                                  | 20% | direct path read | User I/O
+ TOP SQL ID -- PLAN HASH --- SQL TEXT -----+
                     SELECT O.ORDER ID, LINE ITEM ID, PRODUCT ID, UNIT PRICE, QUANTITY, ORDER MODE, O
3jbwa65aqmkvm 2645209246
                     RDER STATUS, ORDER TOTAL, SALES REP ID, PROMOTION ID, C.CUSTOMER ID, CUST FIRST
13sp89mx21tab 644658511
                     SELECT MOATS ASH OT ( SYSTIMESTAMP, SADDR, SID, SERIAL#, AUDSID, PADDR, USER#, US
                     ERNAME, COMMAND, OWNERID, TADDR, LOCKWAIT, STATUS, SERVER, SCHEMA#, SCHEMANAME,
cq8hzxwd7mk8d 415205717
                     SELECT VALUE FROM NLS DATABASE PARAMETERS WHERE PARAMETER = 'NLS NCHAR CHARACTER
                     SET'
```



MOATS v1.0 - Architecture

- It's running in plain old SQLPLUS!
- You heard right, no Perl, Python, Java etc
- Moats.top() is a PL/SQL pipelined function
 - Returning 72 rows at a time, then sleeping, returning more rows, etc.
 - SQLPLUS default arraysize is 15, so use SET ARRAYSIZE 72 first
- We've even created a view on top of the pipelined function:
 - CREATE OR REPLACE VIEW top AS SELECT * FROM TABLE(moats.top)
- So you can just log on via sqlplus and run:
 - SELECT * FROM top;



www.enkitec.com

48

LatchProf and LatchProfX v2.0

Lath Holder Profiler

- Troubleshoot latch contention by measuring who holds the latch the most
- Note that you should troubleshoot latch contention only when you see significant waiting for latches!
 - By significant I mean a significant part of your query response time
 - Some latch contention in (highly concurrent) systems is fine (and healthy)

Version 2.0 gives 2 major improvements:

- Report SQL_ID (or hash_value in 9i) of the statements causing latches to be held
- A package for collecting and storing latch holder data over time
- Just like ASH, but not for active sessions, but for "active" latches



LatchProf 2.0 Syntax and example

Syntax:

- @latchprof <what_columns> <sid> <latch_name> <samples>
- @latchprofx <what_columns> <sid> <latch_name> <samples>
- Look inside the scripts for full reference

Example

Show top SID and SQL_ID holding a latch (break down by name)

```
SQL> @latchprof sid,name,sqlid % % 100000
```

```
-- LatchProf 2.01 by Tanel Poder ( http://www.tanelpoder.com )
```

SID	NAME	SQLID	Held	Gets	Held %	Held ms	Avg hold ms
143	cache buffers chains	bt4txmkqf7u89	1586	1468	1.59	19.984	.014
32	shared pool		529	529	.53	6.665	.013
143	simulator lru latch	bt4txmkqf7u89	89	89	.09	1.121	.013
32	shared pool	cumzymmfh1uvr	50	50	.05	.630	.013
132	row cache objects		1	1	.00	.013	.013
32	shared pool	0000000000000	1	1	.00	.013	.013
130	redo writing		1	1	.00	.013	.013



A little detour to hangs, systemstate dumps (if we have time)

- Interpreting hanganalyze dump
 - \$ grep LEAF tracefile.trc
 - LEAF leaf node in the hang chain
 - Fix (or kill) the leaf and the hang goes away
 - NLEAF Not a Leaf
 - LEAF_NW leaf NoWait (on CPU)
- V\$WAIT_CHAINS in 11g
 - RAC aware, "free" no Diagnostics Pack licenses needed
 - Query V\$ view or if you can't log in even as SYSDBA then:
 - \$ sqlplus –prelim "/as sysdba"
 - SQL> ORADEBUG DIRECT_ACCESS select * from x\$ksdhng_chains
- Systemstate dump explorer it's very simple! ;-)
 - ssexplorer.sh <tracefile>
 - Open the output with a fast browser (preferrably Chrome) as systemstate dumps are large. In some cases pre-filtering is needed to reduce dump size!



A set of random little (but useful) scripts

- @pvalid <parameter_name>
 - Shows valid values of a multioption parameter
 - Uses X\$KSPVLD_VALUES -- 10g+
 - V\$PARAMETER_VALID_VALUES does not show undocumented params
- @hint <name>
 - Shows which hints are available from which database version
 - Uses V\$SQL_HINT 11g+
- @lt <type>
 - Shows the descriptions of enqueue lock types (like TX,TM,KO, etc)
 - Shows what these enqueues ID1, ID2 values mean!
 - V\$LOCK_TYPE --10g+



www.enkitec.com

52

SQL Performance Tools



Getting execution plans

- How to not get an execution plan
 - Forget about the EXPLAIN PLAN FOR command *
 - 1. It does not peek bind variable values
 - It assumes all bind variables are VARCHAR2
 - This causes implicit datatype conversions to be added to the plan
 - And this may cause the explained plan to be different from the real plan
 - 3. It uses *current* optimizer stats and *current session*'s optimizer environment
 - ...these may be different from the stats and environment used at the time of the problem SQL plan compilation
- Read the plan directly from the child cursor instead
 - V\$SQL_PLAN (v9.2+)
 - DBMS_XPLAN.DISPLAY_CURSOR (v10.1+) uses also V\$SQL_PLAN
 - Or grep for STAT# lines in a SQL trace file



Cases where we still may need to use the old explain plan

- Estimating DDL execution plans
 - EXPLAIN PLAN FOR CREATE TABLE AS SELECT ...
 - However the limitations mentioned in previous slide still apply
 - EXPLAIN PLAN FOR ALTER INDEX REBUILD ...
- Testing DML plans in production
 - Did I just say that?!;-)
 - Do it it you want to spice up your work life
 - Or perhaps even get a new job ;-)
 - But if you really do need to do it for whatever reason, then you can still see the real plan without actually running the DML...



Testing in production is a bad idea!

SQL> set transaction read only;
Transaction set.

WARNING! This trick does not work if you are connected as SYS, as for SYS this command doesn't do anything, so your DML would actually run!

```
SQL> delete t where owner not in (select username from all users);
delete t where owner not in (select username from all users)
ERROR at line 1:
ORA-01456: may not perform insert/delete/update operation inside a READ ONLY transaction
SQL> select * from table(dbms xplan.display cursor);
| Id | Operation | Name | Rows | Bytes | Cost (%CPU) | Time |
  0 | DELETE STATEMENT | | 1636 (100) |
               | T | | |
  1 | DELETE
|* 2 | FILTER | | |
 3 | TABLE ACCESS FULL | T | 299K| 8781K| 1631 (1) | 00:00:20 |
 4 | NESTED LOOPS | | 1 | 26 | 5 (0) | 00:00:01 |
  5 | NESTED LOOPS | 1 | 23 | 4 (0) | 00:00:01 |
|* 6 | TABLE ACCESS FULL | USER$ | 1 | 20 | 3 (0) | 00:00:01 |
  7 | TABLE ACCESS CLUSTER | TS$ | 1 | 3 | 1 (0) | 00:00:01 |
|* 8 | INDEX UNIQUE SCAN | I_TS# | 1 | 0 (0) | |
  9 | TABLE ACCESS CLUSTER | TS$ | 1 | 3 | 1 (0) | 00:00:01 |
|* 10 | INDEX UNIQUE SCAN | I TS# | 1 | 0 (0)|
```



Viewing the parameters used for plan compilation

- Optimizer environment is stored inside every child cursor
 - V\$SQL
 - optimizer_envRAW(839)
 - optimizer_env_hash_value NUMBER
 - V\$SQL_OPTIMIZER_ENV
 - V\$SES_OPTIMIZER_ENV
 - V\$SYS_OPTIMIZER_ENV
- Scripts to query the underlying X\$ tables
 - @sqlopt <hash_value> <child#> <%parameter%>
 - @sesopt <sid> <%parameter%>
 - @sysopt <%parameter%>



Using optimizer_features_enable for troubleshooting

- Very useful for reverting back to "old" behavior after upgrade
 - At session / application level
 - With ALTER SESSION
 - Or at SQL statement level
 - Using OPTIMIZER_FEATURES_ENABLE('<version>') hint
- But also very useful for troubleshooting SQL issues after upgrade easy way to narrow down some issues
 - The first things I check when some SQL runs badly after upgrade
 - The execution plan (obviously)
 - 2. The differences in (undocumented) optimizer parameters
 - Using sqlopt.sql for a specific SQL (need to run it on both old and new db)
 - Or a script which gives me the optimizer parameter changes between any Oracle database release (see next slide)



www.enkitec.com

58

Compare optimizer_features_enable impact

- Create the opt_param_matrix table
 - SQL> @tools/optimizer/optimizer_features_matrix.sql
- Report parameter differences with @cofep.sql

```
SQL> @cofep.sql 10.2.0.4 11.1.0.7
Compare Optimizer Features Enable Parameter differences for values
10.2.0.4 and 11.1.0.7
                                         '10.2.0.4' '11.1.0.7'
PARAMETER
bloom folding enabled
                                                    TRUE
                                         FALSE
bloom pruning enabled
                                                    TRUE
                                         FALSE
first k rows dynamic proration
                                         FALSE
                                                    TRUE
nlj batching enabled
optimizer adaptive cursor sharing
                                         FALSE
                                                    TRUE
optimizer enable density improvements
                                         FALSE
                                                    TRUE
optimizer enable extended stats
                                         FALSE
                                                    TRUE
optimizer extend jppd view types
                                         FALSE
                                                    TRUE
optimizer extended cursor sharing rel
                                         NONE
                                                    SIMPLE
optimizer extended stats usage control
                                         255
                                                    2.2.4
```



www.enkitec.com

59

Reading execution plans – on one slide

Execution plan is a tree

- Tree of row source operators (which are internally just C functions)
- Root of the tree is where all "matching" rows are pulled from the leaves
 - Root is the execution plan line with ID=0 (top line in the plan output)
 - For SELECT STATEMENT the root operation passes rows to OPI Fetch which sends the rows back to client
 - For DML, the root operation will modify the rows returned to it from branches under it
- Tree hierarchy
 - Every row source in the tree can get input only from its direct children
 - CBO can push predicates deeper towards the leaves of the tree
 - Data access starts from the first leaf operation of the tree *
 - Joins can join only two child row sources together at a time



Flexible DBMS_XPLAN.DISPLAY_CURSOR alternatives

- XM (eXplain from Memory) scripts
 - @xmsh <hash_value> <child#>
 - eXplain from Memory with Statistics by hash_value)
 - @xmsi <sqlid> <child#>
 - eXplain from Memory with Statistics by SQL ID)
 - @xms
 - eXplain from Memory with statistics last SQL executed in current session)
- They use V\$SQL_PLAN% views
 - just like DBMS_XPLAN.DISPLAY_CURSOR does
 - They work from Oracle 9.2
 - They report better detail than DBMS_XPLAN.DISPLAY_CURSOR does
 - DML example report consistent gets vs current gets



Profiling execution plans - 1

- Just like you can profile which SQLs take most of the application response time, you can profile which SQL execution plan lines take most of a SQL statement response time
 - alter session set statistics_level = all
 - Or _rowsource_execution_statistics = true
 - Works since 9.2
 - gather_plan_statistics hint
 - Works from 10.2
 - SQL_TRACE enables plan profiling too for newly parsed cursors!
 - You can verify it by looking into the sqlstat_enabled compilation environment parameter (using @sqlopt.sql)



Profiling execution plans - 2

- The V\$SQL_PLAN_STATISTICS[_ALL] views are updated at the end of the call (fetch for example)
 - If the statement is already overrunning its time, you can cancel it with CTRL+C, resource manager or even kill the session (but not the process)
 - And the statistics up to the query cancellation will be loaded to V\$SQL_PLAN_STATISTICS
- Since 11g, ASH samples current execution plan line info too!
 - _rowsource_profiling_statistics = true by default
 - Populates new columns in V\$ACTIVE_SESSION_HISTORY:
 - SQL_PLAN_LINE_ID corresponds to the ID in V\$SQL_PLAN
 - SQL_PLAN_OPERATION for example: TABLE ACCESS
 - SQL PLAN OPTIONS for example: FULL



Profiling execution plans - 3

- Starting from 10g, the rowsource function elapsed times are sampled (not every row source call is timed)
 - The sampled times are extrapolated and accounted to the rowsource which was active during the sample
 - This may skew the profile stats in some cases
 - You can set the _rowsource_statistics_sampfreq lower for test run
 - Must be power of 2 (default is 128)
 - Note that the CPU usage goes much higher (especially with NL joins)

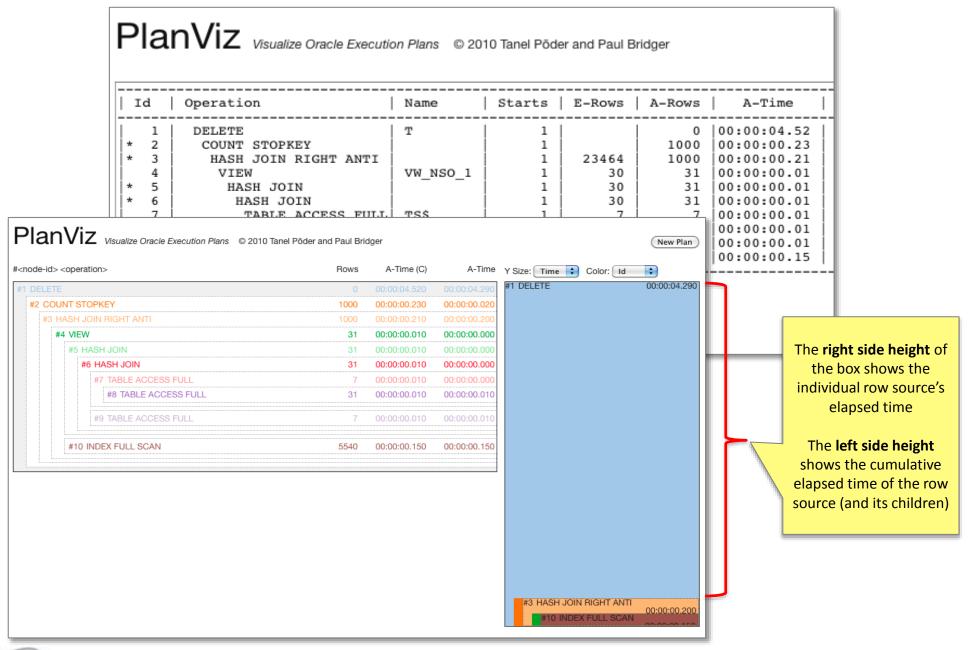
Id	Operation	Name	St	arts I	E-Rows	A-Rows	A-Time	
0	SELECT STATEMENT			1		1	00:00:01.10	1
1	SORT AGGREGATE	1	1	1	1	1	00:00:01.10	\perp
2	VIEW	DBA SOURCE	1	1	589K	624K	00:00:03.58	1
3	UNION-ALL	_	1	1	1	624K	00:00:03.02	
* 4	FILTER	1	1	1	1	624K	00:00:01.91	
* 5	HASH JOIN	1	1	1	589K	624K	00:00:01.09	
* 6	HASH JOIN		1	1	6446	6899	00:00:00.04	
7	INDEX FULL SCAN	I_USER2	1	1	100	101	00:00:00.01	1
* 8	HASH JOIN			1	6446	6899	00:00:00.03	1



Visualizing execution plan profiles

- Challenge
 - The elapsed time figures reported by SQL plan profiling are cumulative
 - They include time spent:
 - In the execution plan line (rowsource) itself
 - All child rowsources "under" it in the execution plan tree
 - It's not convenient to manually calculate the real non-cumulative runtimes
 - Especially for large plans
- PlanViz http://tech.e2sn.com/apps/planviz/
 - Free online execution plan hierarchy and response time profile visualizer
 - Parses DBMS_XPLAN.DISPLAY(_CURSOR) output
 - Calculates the non-cumulative A-Time values
 - Visualizes the plan profile for quick drilldown

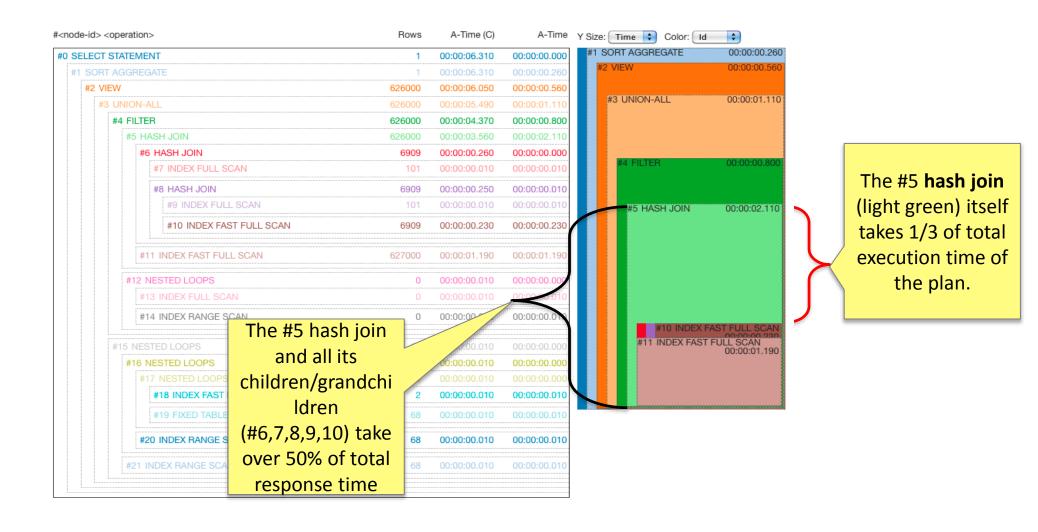






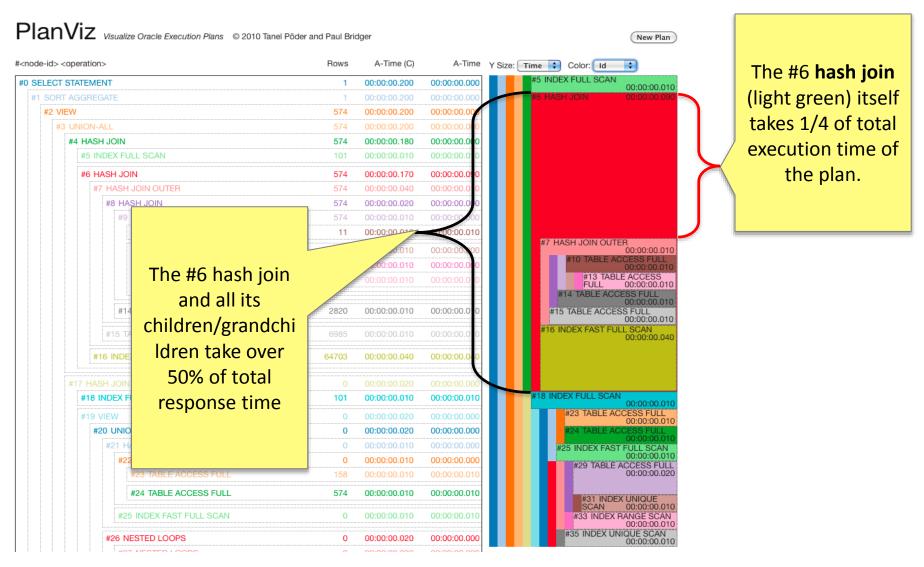
<u>www.enkitec.com</u>

Reading PlanViz plan output - 1





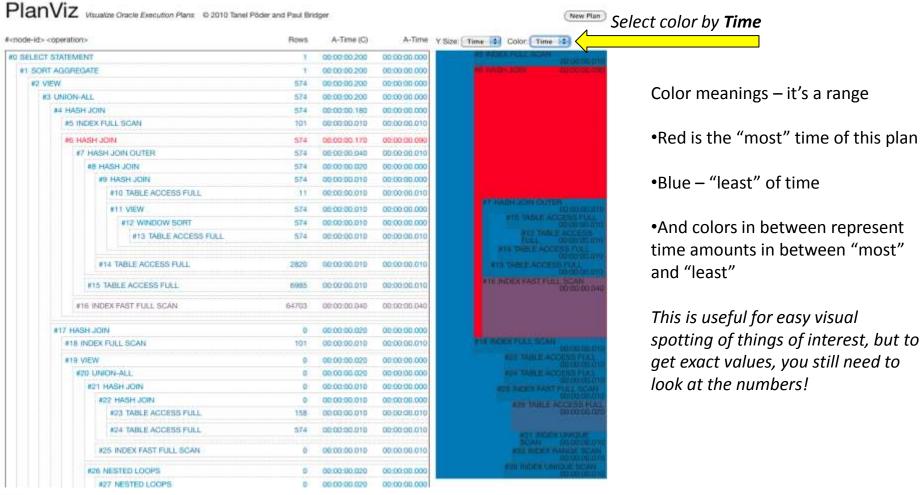
Reading PlanViz plan output- 2





Reading PlanViz plan output – 3 (continued)

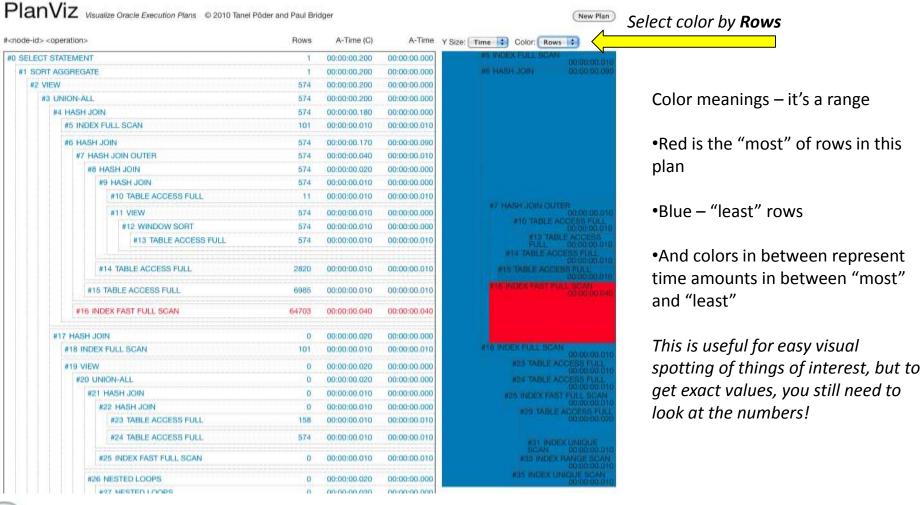
Color row sources by time spent in them



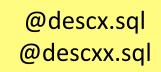


Reading PlanViz plan output – 4 (continued)

Color row sources by number of rows they return to parent







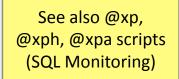
A better describe command

- @desc
 - Shows basic optimizer statistics (also column low/high values)

SQL> @desc obj\$

Col#	Column Name	Nu	11?	Type	NUM_DISTINCT	DENSITY	NUM_NULLS	NUM_BUCKETS
1	OBJ#	NOT	NULL	NUMBER(,)	74760	.000013376	0	1
2	DATAOBJ#			NUMBER(,)	9657	.000103552	65059	1
3	OWNER#	NOT	NULL	NUMBER(,)	35	6.6338E-06	0	28
4	NAME	NOT	NULL	VARCHAR2(30)	44960	.000040124	0	254
5	NAMESPACE	NOT	NULL	NUMBER(,)	21	6.6338E-06	0	15
6	SUBNAME			VARCHAR2(30)	577	.00205538	73440	254
7	TYPE#	NOT	NULL	NUMBER(,)	46	6.6338E-06	0	33
8	CTIME	NOT	NULL	DATE (7)	2671	.001430615	0	254
9	MTIME	NOT	NULL	DATE (7)	2702	.001455604	0	254
10	STIME	NOT	NULL	DATE (7)	2776	.001432665	0	254
11	STATUS	NOT	NULL	NUMBER(,)	2	6.6338E-06	0	1
12	REMOTEOWNER			VARCHAR2(30)	0	0	74760	0
13	LINKNAME			VARCHAR2 (128)	0	0	74760	0
14	FLAGS			NUMBER(,)	10	6.6338E-06	0	9
15	OID\$			RAW(16)	3268	.000305998	71492	1
16	SPARE1			NUMBER(,)	2	.5	0	1
17	SPARE2			NUMBER(,)	11	.090909091	0	1
18	SPARE3			NUMBER(,)	35	6.6338E-06	0	28
19	SPARE4			VARCHAR2 (1000)	0	0	74760	0
20	SPARE5			VARCHAR2(1000)	0	0	74760	0
21	SPARE6			DATE (7)	0	0	74760	0





Parallel Execution slave communication

- @tq
 - Shows how many rows PX slaves have consumed from communications buffers (Table Queues) in different parallel execution plan phases (TQ_IDs)

SQL> @tq Show PX Table Queue statistics from last Parallel Execution in this session...

TQ_ID (DFO,SET)	SERVER_TYPE	NUM_ROWS	BYTES	WAITS	TIMEOUTS	PROCESS	INSTANCE	DFO_NUMBER	TQ_ID
:TQ10000	Produced	376	1289	0	0	QC	1	1	0
	Consumed	94	399	6		P007	1		
	Consumed	94	399	5	1	P005	1		
	Consumed	94	399	4	0	P004	1		
	Consumed	94	399	6	1	P006	1		
:TQ10001	Produced	376	2562	0	0	QC	1	1	1
	Consumed	94	1189	3	0	P007	1		
	Consumed	94	1189	5	1	P005	1		
	Consumed	94	1189	4	0	P004	1		
	Consumed	94	1189	6	1	P006	1		
:TQ10002	Produced	6968	132404	3	2	P007	1	1	2
	Produced	7956	150304	8	2	P006	1		
	Produced	9172	171052	2	0	P005	1		
	Produced	3884	72856	8	1	P004	1		
	Consumed	6995	131654	4	1	P002	1		
	Consumed	6995	131654	4	1	P000	1		



www.enkitec.com

72

Visualization Tools for Performance Analysis



PerfSheet v3.0 - Introduction

Problem:

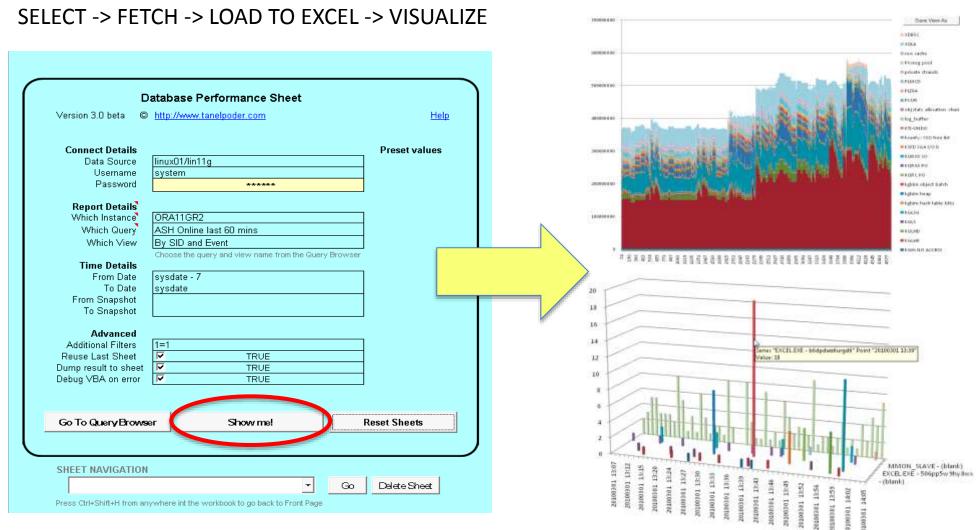
- Visualize (performance) data
- Do it easily...
- Not manually!
 - The manual cycle:
 - SELECT -> SPOOL -> FIX -> LOAD TO EXCEL -> FIX -> CREATE CHART

Solution:

- PerfSheet
- Automates the manual cycle above
- Nothing more
- You still need to write the queries (if the existing ones in there aren't enough)
- PerfSheet (now v3.0) greatly speeds up graphical data analysis work!



PerfSheet v3.0 - Demo





PerfSheet v3.0 - Architecture

- Requires Excel 2002+
 - Does not work on the MacOS Office (No VBA support in Excel 2008)
- Uses MS ADODB
 - Oracle (instant) client drivers must be installed and available
 - No need for manual ODBC data source setup
 - No need for TNSNAMES.ORA or naming configuration!
- When you select a query and hit "Show me!" then PerfSheet will:
 - Go to Queries Tab and get the SQL text corresponding to the query name you selected
 - 2. Log on to database you selected, run the SQL and fetch all data
 - 3. Create a PivotChart out of the data and visualize it as defined in the selected View (visualization rules are defined on the Views tab)



PerfSheet v3.0 – Setting up your own queries

- Go to Queries sheet
- 2. Add new query to the end of the list (follow other examples)
- 3. Go to front page, select your query in query browser and click "Show me"
- Drag & Drop the columns you want to appropriate axis on the chart
- 5. Click "Save View As" and type in a view name
 - View is just a way to visualize the queries data
 - You can have multiple views per query to show the same date in a different way
- 6. You're all done
 - You need to save the PerfSheet.xls of course



www.enkitec.com

77

"Free" AWR views in 11g

- Since 11g you may query following AWR views even if you don't have Diagnostics Pack license
 - DBA_HIST_SEG_STAT
 - DBA HIST SEG STAT OBJ
 - DBA_HIST_UNDOSTAT
 - Documented in Oracle Licensing Guide
 - http://download.oracle.com/docs/cd/B28359 01/license.111/b28287/options.htm#st href69 (or google for "Free AWR views", you'll land on my blog article)
- In 10g you'd need to stick with StatsPack level 7
 - Or modify the snap() procedure to include segment stats without the execution plan snapshots (level 6)



www.enkitec.com

78

Demo - Identifying trends inside execution plans

- 11g + ASH only, unfortunately
 - Or on Solaris 10 with Dtrace ;-)
 - ASH has new SQL_PLAN_LINE_ID column

