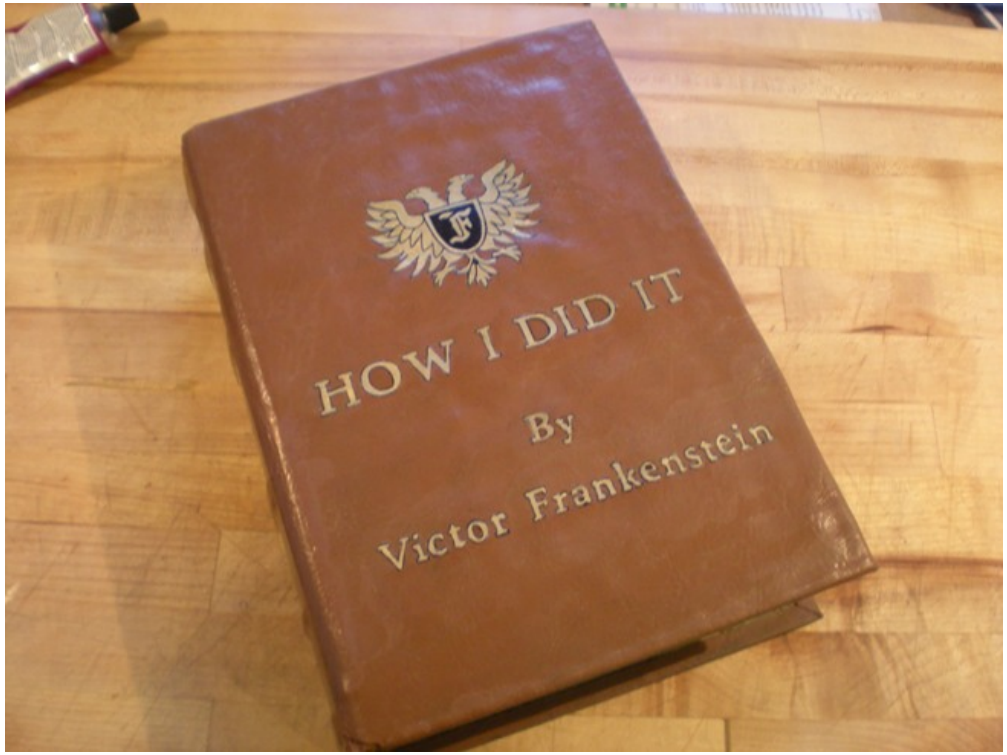
enkitec

My Favorite Scripts - 2010

by Kerry Osborne (a command line guy)

enkitec

My Favorite Scripts - 2010



whoami

Started working with Oracle in 1983
Primarily a developer for the first 5-6 years
Became a consultant around 1987
(through no fault of my own)
…

Never worked directly for Oracle
Not certified in anything (except Scuba Diving)
But I have attended the Hotsos Symposium 5 years in a row!

I am a command line guy!

enkitec

Philosophies

Believe nothing of what you hear, and only half of what you see. ~ Proverb

No hypothesis before analysis. ~ me

Working in Test is for amateurs. ~ me

If your elbow hurts, look at your elbow (not your ankle). ~ me

If I have seen farther it is by standing on the shoulders of giants. ~ Isaac Newton

There are no new notes, but plenty of new original songs. ~ me

I'll let you be in my dream if I can be in yours. ~ Bob Dylan

Why I Use Scripts

Because I'm a command line guy.

Because I like to know where the data came from.

Because sometimes I need (or want) something unique.

Because I'm not a good typer-ater.

Because I'm forgetful (occasionally).

Because I'm lazy.

Because people with problems are impatient!

Categories

Collecting Data:

- What's Going On?
- Finding SQL and Plans
- Finding Stats (table_stats, col_stats)

Prospecting:

- Identifying Periods of High Activity
- Looking for Significant Variation

Testing:

- Building Test Scripts
- Executing Test Scripts
- Flushing

Miscellaneous:

- Bits and Bobs

What's Going On?

First: Machine Level

```
vmstat, sar, top, topas, prstat, ps, uptime, realfreemem.sh, etc…
```

Second: Active SQL

```
-- as.sql
Select sid, substr(program,1,19) prog, b.sql_id, child_number child,
plan_hash_value, executions execs,
(elapsed_time/decode(nvl(executions,0),0,1,executions))/1000000 avg_etime,
sql_text
from v$session a, v$sql b
where status = 'ACTIVE'
and username is not null
and a.sql_id = b.sql_id
and a.sql_child_number = b.child_number
and sql_text not like 'select sid, substr(program,1,19) prog, b.sql_id%' -- don't show this
```

Third: AWR or Statspack

```
-- awrrpt.sql
   @$ORACLE_HOME/rdbms/admin/awrrpt
```

Statspack / AWR

Provides a lot of data

Very useful despite the aggregation of data

Snapshots collected once per hour by default (OK)

Retention is 7 days by default (not long enough)

Can report between any two snapshots

(but not across a bounce)

- **AWR / Statspack -** basically the same thing (except for the licensing fee)

Statspack / AWR

Contains Basic Workload Information
transactions, parses, db time, etc…

Contains a profile of how the database spent it's time
also known as a wait event profile

Contains most expensive SQL (along with %)
by Etime, CPU , Gets, Reads, Execs

List of Non-Default Parameters
including so called hidden parameters
(but not double top secret parameters)

Statspack / AWR - Warning

Not the tool for analyzing specific problems
- **aggregation of data can hide significant variation**
- **general issues may not apply to specific case**

On the Other hand …
- **it provides a good background context**

Finding SQL and Plans

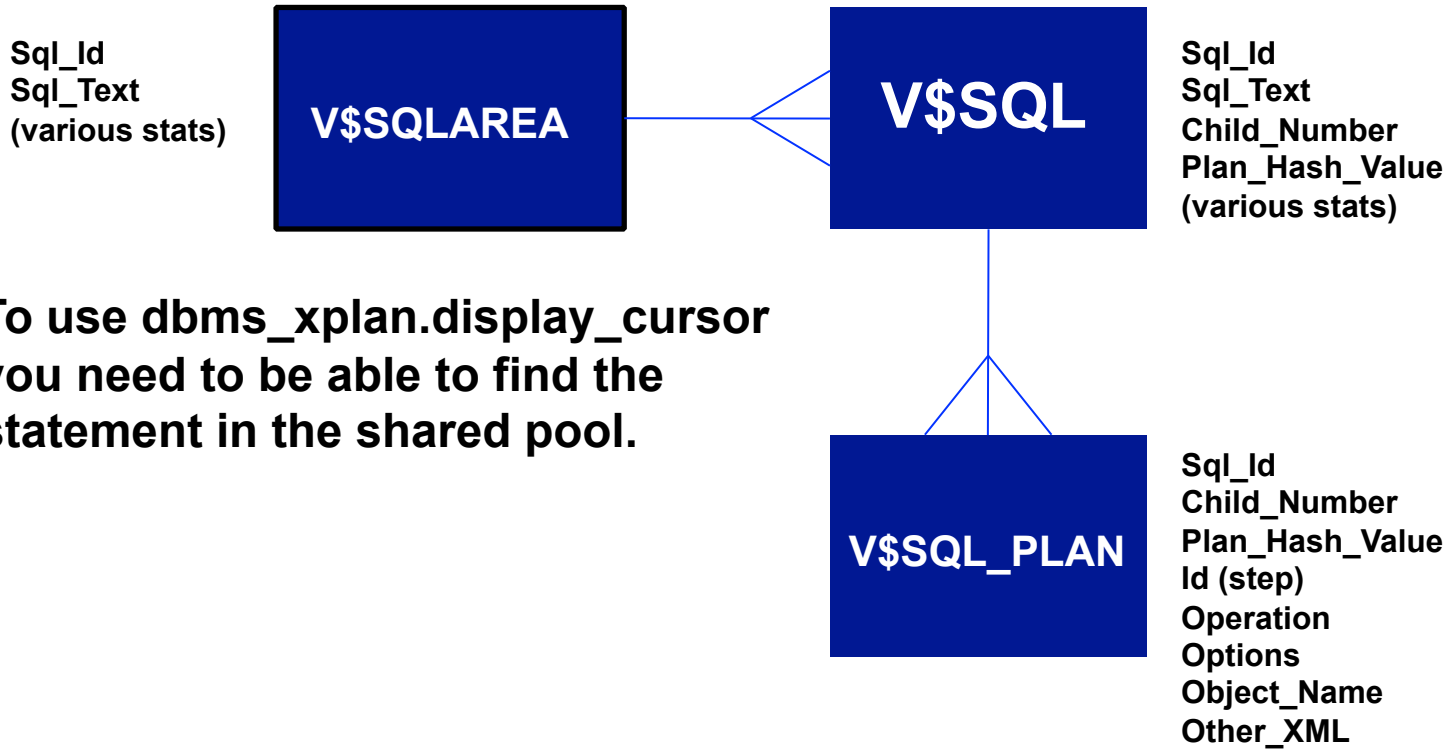
Finding SQL

- find_sql.sql (fs.sql) – find sql in V$SQL
- find_sql_awr.sql – find sql in DBA_HIST_SQLSTAT

Finding Plans

- dplan.sql – show plan for statement in V$SQL (uses dbms_xplan)
- dplan_awr.sql – show plan for statement in AWR history
- dplan_allstats.sql – show plan with extended stats

Shared Pool Layout (V$SQL…)



To use dbms_xplan.display_cursor you need to be able to find the statement in the shared pool.

Note: prior to 10g hash_value used as key (no sql_id)

Finding Statements in the Shared Pool

```
SQL> !cat find_sql.sql
select sql_id, child_number, plan_hash_value plan_hash, executions execs,
(elapsed_time/1000000)/decode(nvl(executions,0),0,1,executions) avg_etime,
buffer_gets/decode(nvl(executions,0),0,1,executions) avg_lio,
sql_text
from v$sql s
where upper(sql_text) like upper(nvl('&sql_text',sql_text))
and sql_text not like '%from v$sql where sql_text like nvl(%'
and sql_id like nvl('&sql_id',sql_id)
order by 1, 2, 3
/


SQL> @find_sql
Enter value for sql_text: %skew%
Enter value for sql_id:

SQL_ID         CHILD  PLAN_HASH  EXECS  AVG_ETIME      AVG_LIO SQL_TEXT
------------- ------ ---------- ------ ---------- ------------ ----------------------------------------------------
0qa98gcnnza7h      0  568322376      5      13.09      142,646 select avg(pk_col) from kso.skew where col1 > 0
0qa98gcnnza7h      1 3723858078      1       9.80    2,626,102 select avg(pk_col) from kso.skew where col1 > 0
```

DBMS_XPLAN

```
SQL> !cat dplan.sql
select * from table(dbms_xplan.display_cursor('&sql_id','&child_no','typical'))
/

SQL> @dplan
Enter value for sql_id: 3s1yukp05bzg6
Enter value for child_no: 0

PLAN_TABLE_OUTPUT
------------------------------------------------------------------------------------------------------------
SQL_ID  3s1yukp05bzg6, child number 0
-------------------------------------
select   schedule_mode, start_calibrate, num_votes,   synced_time, last_vote, status from
 WRI$_SCH_CONTROL where   schedule_id = :id

Plan hash value: 4159334603

------------------------------------------------------------------------------------------------------------
| Id  | Operation                    | Name                | Rows  | Bytes | Cost (%CPU)| Time     |
------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |                     |       |       |     1 (100)|          |
|   1 |  TABLE ACCESS BY INDEX ROWID| WRI$_SCH_CONTROL    |     1 |    28 |     1   (0)| 00:00:01 |
|*  2 |   INDEX UNIQUE SCAN          | WRM$_SCH_CONTROL_PK |     1 |       |     0   (0)|          |
------------------------------------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   2 - access("SCHEDULE_ID"=:ID)
```

DBMS_XPLAN

4 Display Functions:

- Display – plan table
- Display_Cursor – shared pool
- Display_AWR – AWR tables
- Display_Baseline – 11g

Options: (+,-)

- ALIAS
- ALLSTATS *
- IOSTATS
- MEMSTATS
- OUTLINE
- PEEKED_BINDS
- PREDICATE

See Rob van Wijk's blog for a very detailed set of examples
http://rwijk.blogspot.com/2008/03/dbmsxplandisplaycursor.html

DBMS_XPLAN - Options

allstats –

select /*+ gather_plan_statistics */ blah,blah,blah …
select * from table(dbms_xplan.display_cursor('&sql_id','&child_no','allstats'));

```
-----------------------------------------------------------------------------------------------------------------------
| Id  | Operation                           | Name                | Starts | E-Rows | A-Rows |   A-Time    | Buffers |
-----------------------------------------------------------------------------------------------------------------------
|*  1 |  COUNT STOPKEY                      |                     |      1 |        |      1 |00:00:00.01 |      44 |
|   2 |   PARTITION RANGE ALL               |                     |      1 |      1 |      1 |00:00:00.01 |      44 |
|   3 |    PARTITION HASH ALL               |                     |     10 |      1 |      1 |00:00:00.01 |      44 |
|   4 |     TABLE ACCESS BY LOCAL INDEX ROWID| DODA_TABLE          |     37 |      1 |      1 |00:00:00.01 |      44 |
|*  5 |      INDEX RANGE SCAN               | DODA_TABLE_IDX1     |     37 |   3403K|      1 |00:00:00.01 |      43 |
-----------------------------------------------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   1 - filter(ROWNUM<2)
   5 - access("COL1">SYSDATE@!)
```

Hint is not necessarily required:

alter session set statistics_level=all;
alter session set “_rowsource_execution_statistics”=true;

Finding Object Statistics

table_stats.sql – Shows table, column, index, stats for a table
col_stats.sql – Shows column stats including max and min

create_display_raw.sql – Used by col_stats.sql to convert raw max and min
diff_table_stats.sql – Shows what's changed

Prospecting

Busiest Time is Generally the Most Interesting:

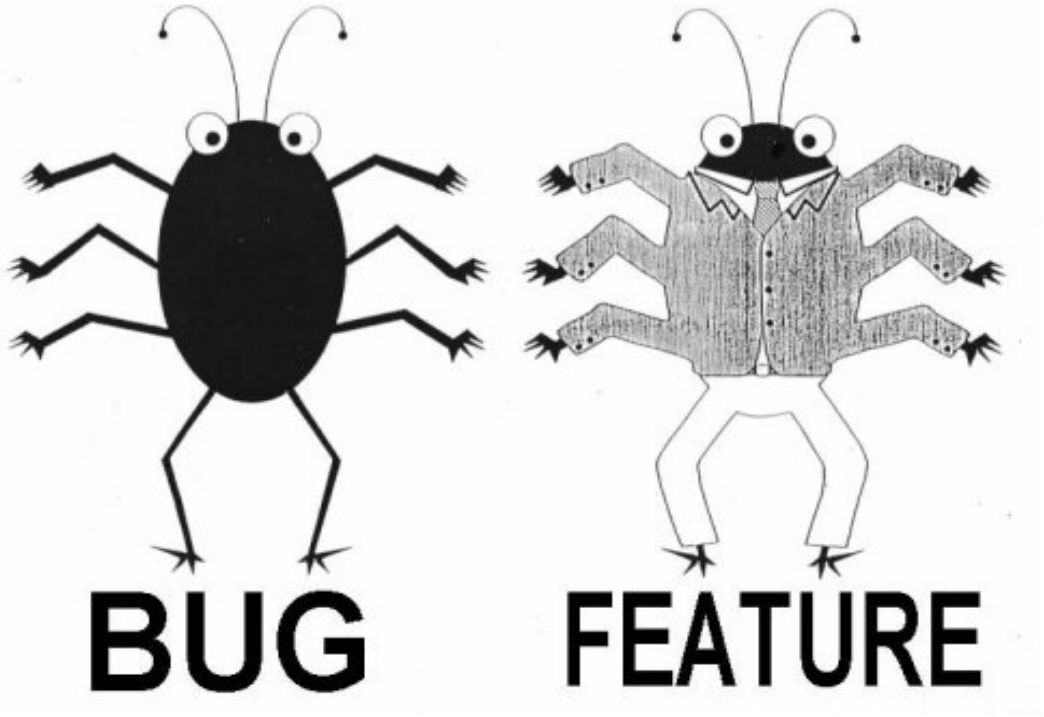
```
-- dbtime.sql - finds snapshots with highest dbtime
```

Looking for Large Variation:

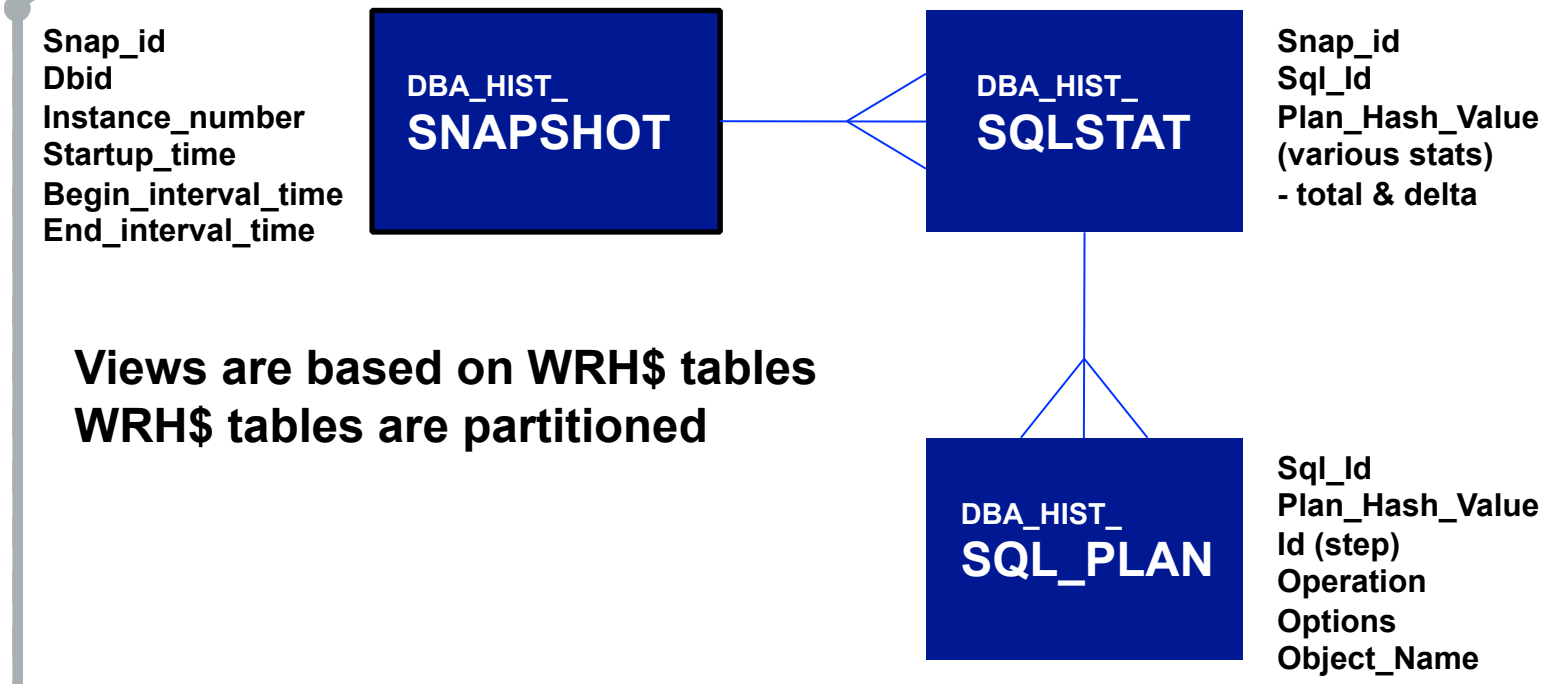
```
-- unstable_plans.sql - finds statements with significant elapsed time variation by plan
-- awr_plan_stats.sql - shows stats (avg_etime, avg_lio, etc) for statements by plan
-- awr_plan_change.sql - shows history of plan changes

-- whats_changed.sql - finds SQL with significant elapsed time variation after a point in time
```

Digression – Bind Variable Peeking

Drives Me Nuts!



AWR Layout (DBA_HIST_SQLSTAT…)



Views are based on WRH$ tables
WRH$ tables are partitioned

Testing

Getting the Statement (and variables):

```
-- build_bind_vars.sql - builds test script including peeked binds (OTHER_XML)
-- build_bind_vars2.sql - builds test script including binds from V$SQL_BIND_CAPTURE
-- build_bind_vars_awr.sql - builds test script including binds from AWR
```

Flushing:

```
-- flush_sql.sql - flush a single SQL statement using dbms_shared_pool.purge (11g & 10.2.0.4)
-- flush_sql10p.sql - creates and drops a Profile which as a side affect flushes SQL
```

Executing:

```
-- ss.sh - fires off a bunch of sessions executing a specified SQL script
```

Miscellaneous

```
-- os_pid.sql - find os pid from sid, or sid from os pid
-- sql_hints.sql - see the outline hints for a SQL statement
-- gps.sql - add gather_plan_statistics hint to a SQL statement via Profile
-- we.sql - show wait events for a SQL_ID or SQL_ID's that have waited for some event
-- parm.sql - show values of regular and hidden parameters
-- parm_mods.sql - show parameters that have changed
-- mismatch.sql - shows why new child created using V$SHARED_CURSOR
```

enkitec

Scripts in Zip

as.sql
awr_plan_change.sql
awr_plan_stats.sql
build_bind_vars.sql
build_bind_vars2.sql
build_bind_vars_awr.sql
col_stats.sql
create_display_raw.sql
dbtime.sql
diff_table_stats.sql
dplan.sql
dplan_allstats.sql
dplan_awr.sql
find_sql.sql
find_sql_awr.sql
flush_sql.sql
flush_sql10p.sql
gps.sql
mismatch.sql
my_favorite_scripts_2010.zip
os_pid.sql
parm_mods.sql
parms.sql
sql_hints.sql
ss.sh
table_stats.sql
table_stats2.sql
unstable_plans.sql
we.sql
whats_changed.sql

Note: Many of these scripts are discussed in detail on my blog.

Questions / Contact Information

Questions?

Contact Information : Kerry Osborne

kerry.osborne@enkitec.com
kerryosborne.oracle-guy.com
www.enkitec.com