Sessions

Total user count on database

Users Details Session

v$sql_monitor

Sql-Monitor report for a sql_id (Like OEM report)

Sid Level Debug

Long Running query - long ops

Wait event for a SID

RealTime Monitoring for sid

Sql details

Currently running SQL query's

RealTime Monitoring for sql_id

SQL History

Identify

statistics of objects of a specific sql id

Sql id current rows, I/O , read/wrire details

Check the progress of DML statements

Detailed history of SQL_ID

Sql Hanging or not

SQL to show the full SQL executing for active sessions

wait events

Database level Wait ratios/Events

Events Drilled down %

wait event for past 5 mins / 30 mins/ 1 hour

wait events for a sql_id

Wait event for objects in active session history

Explain Plan

explain plan for manual session

explain plan for sql id present/history

Process

Finding SID for a PID

Finding OS pid for a db sid

Finding db sid from OS spid

Finding own sid & serial number

Killing Sessions

Killing Own session

kill a sid

Killing old session (1 day old)   (or)   4 hours

OS level killing old session for database

Determine If Killed Session Is Rolling Back Transactions

Extra

CPU/REDO/PGA/READ/WRITE used by a SID

Cursor Value for a SID (All Sql query executed by SID)

Maximum no of session in the database
Cheking all sql from a user

wait event currently- ON CPU

command to purge explain plan for a specific program...

Enable Parallelism in session level

parallelism Hints

Uncommited transactions in the database

SQL Profiles & Baselines

sql patch

Sql Profiles

Sql Baselines

extract hint from all profile/baseline/patch - single script

Fixing the PLAN HASH VALUE

purge the old plan hash value from memory

Sql_doctor script

break on report compute SUM of tot on report compute SUM of active on report compute SUM of inactive on report col username for a50 select DECODE(username,NULL,'INTERNAL',USERNAME) Username,	
set linesize 750 pages 9999 column box format a30 column spid format a10 column username format a30 column program format a30	<-- For cluster for standalone -->
column sid format 9999 column serial for 999999 column status format a15 column username format a10 column sql_text format a80	overall --> <-- ONLY EXECUTING
set lines 750 pages 9999 set long 20000 longchunksize 20000 select dbms_sqltune.report_sql_monitor_list() text_line from dual;	trimspool on trim on long 2000000 longchunksize 2000000 select DBMS_SQLTUNE.REPORT_SQL_MONITOR(sql_id=>'&sql_id',
gv$session	gv$sql_monitor
set linesize 750 pages 9999 column box format a30 column spid format a10 column username format a20 column program format a30	set lines 1000 pages 9999 column sid format 9999 column serial for 999999 column status format a15 column username format a20

SELECT
a.sid,RPAD(a.opname,30),a.sofar,a.totalwork,a.ELAPSED_SECONDS,ROUND(((a.sofar)*100)/a.totalwork,3) "%_COMPLETED",
RPAD(a.username,10)
	set pages 50000 lines 32767
	col OPNAME for a10
	col SID form 9999
	col SERIAL form 9999999
	col PROGRAM for a10

Currently waiting ?	Overall Waits
col WAIT_CLASS for a10 SELECT sw.inst_id,NVL(s.username, '(oracle)') AS username, s.sid, s.serial#,	COLUMN username FORMAT A20 COLUMN sid FORMAT 9999 COLUMN serial# FORMAT 9999 COLUMN event FORMAT A40
Elapsed/CPU/Read/Write MB	Each Layer time spend
SELECT FROM (SELECT status, --username, sql_id,	SELECT ROUND(elapsed_time /1000000) AS "Elapsed (s)", ROUND(cpu_time /1000000,3) AS "CPU (s)", ROUND(queuing_time /1000000,3) AS
v$sqlarea/v$sql	ASH

column sid format 9999 column username format a15 column PARSING_SCHEMA_NAME format a15 column SQL_EXEC_START for a21	column my_ser format 99999 column my_state format a30 column my_blkr format 999 select to_char(a.sample_time, 'HH24:MI:SS')
Sql_monitor report	Elapsed/CPU/Read/Write MB
trimspool on trim on long 2000000 longchunksize 2000000 select DBMS_SQLTUNE.REPORT_SQL_MONITOR(	FROM (SELECT status, --username, sql_id,
All in one	AWR data

COL instance_number FOR 9999 HEA 'Inst';
COL end_time HEA 'End Time';
COL plan_hash_value HEA 'Plan|Hash Value';
COL executions_total FOR 999,999 HEA 'Execs|Total';
COL rows_per_exec HEA 'Rows Per Exec';
	COL instance_number FOR 9999 HEA 'Inst';
	COL end_time HEA 'End Time';
	COL plan_hash_value HEA 'Plan|Hash Value';
	COL executions_total FOR 999,999 HEA 'Execs|Total';

sql_id	sql_text
99999999999999999999999999 col sql_text for a50 select sql_id, exact_matching_signature, SQL_TEXT from v$sqlarea where UPPER(sql_text) like '%DUMMY%' order by UPPER(sql_text);	set lines 750 pages 9999 select sql_text from dba_hist_sqltext where sql_id = '&SQL_ID'; set long 20000
set lines 300 set pages 300 col table_name for a40 col owner for a30 select distinct owner, table_name, STALE_STATS,	col index_name for a50 SELECT owner, index_name, table_name,last_analyzed, sample_size, num_rows,
SQL_ID,PLAN_HASH_VALUE,ELAPSED_TIME,PHYSICAL_READ_BYTES,PHYSICAL_WRITE_BYTES,DISK_READS,DIRECT_WRITES,BUFFER_GETS,ROWS_PROCESSED from v$sqlstats where sql_id='&sql_id';	select sql_text,rows_processed from v$sql where USERS_EXECUTING>0;
col sql_text for a60 SELECT rows_processed "Total Rows Processed", ROUND((SYSDATE - TO_DATE(first_load_time, 'yyyy-mm-dd hh24:mi:ss')) * 24 * 60, 1) "Total Time (Min)", TRUNC(rows_processed /((SYSDATE -	
select instance_number inst_id,SESSION_ID,USER_ID,PROGRAM,sql_id,SQL_CHILD_NUMBER,sql_plan_hash_value,to_char (sql_exec_start, 'dd-Mon-yyyy hh24:mi:ss') sql_exec_start from	select sql_id, starting_time, end_time, (EXTRACT(HOUR FROM run_time) * 3600 + EXTRACT(MINUTE FROM run_time)
sess_io.sid, sesion.sql_id, sess_io.block_gets,	sess_io.sid, sess_io.block_gets, sess_io.consistent_gets,
select sesion.sid,sql_text from v$sqltext sqltext, v$session sesion where sesion.sql_hash_value =	select a.sid,b.sql_fulltext from V$Session a, V$SQLAREA b where a.sql_id=b.sql_id and

Wait Ratio	Wait Percentage % (Only wait events-NO CPU)

SELECT METRIC_NAME, VALUE
FROM V$SYSMETRIC
WHERE METRIC_NAME IN ('Database CPU Time Ratio',
'Database Wait Time Ratio') AND
INTSIZE_CSEC =
(select max(INTSIZE_CSEC) from V$SYSMETRIC);
	SELECT WAIT_CLASS,
	TOTAL_WAITS,
	round(100 * (TOTAL_WAITS / SUM_WAITS),2)
	PCT_TOTWAITS,
	ROUND((TIME_WAITED / 100),2)
	TOT_TIME_WAITED,
	round(100 * (TIME_WAITED / SUM_TIME),2)

	SET lines 750 pages 10000 COLUMN wait_class format a30 COLUMN event format a60 COLUMN total_waits format 999999 COLUMN total_us format 999999999 COLUMN pct_time format 99.99 COLUMN avg_us format 999999.99

SELECT NVL(a.event, 'ON CPU') AS event,
COUNT(*) AS total_wait_time
FROM v$active_session_history a
WHERE a.sample_time > SYSDATE - 5/(24*60) -- 5 mins
GROUP BY a.event
ORDER BY total_wait_time DESC;
	SELECT NVL(a.event, 'ON CPU') AS event,
	COUNT(*) AS total_wait_time
	FROM v$active_session_history a
	WHERE a.sample_time > SYSDATE - 60/2880 -- 30 mins
	GROUP BY a.event
	ORDER BY total_wait_time DESC;

select event, time_waited "time_waited(s)", case when time_waited = 0 then 0	<-- shared pool (current) From AWR -->
set lines 750 pages 9999 col OBJECT_NAME for a35 select * from (SELECT	

" @?/rdbms/admin/utlxplp.sql
(or)
SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY('','','+COST +BYTES -PREDICATE'));

SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY('','','ALLSTATS LAST +OUTLINE +PEEKED_BINDS +PROJECTION +ALIAS +PREDICATE +COST +BYTES'));

TABLE(DBMS_XPLAN.DISPLAY_CURSOR('&sql_id',&
childnumber,'ALLSTATS LAST +PEEKED_BINDS +PROJECTION +ALIAS +OUTLINE +PREDICATE +COST +BYTES'));
(OR)
select * from table(dbms_xplan.display_cursor('&sql_id',&childnumber,'ADVANCED'));

<-- shared pool (current)

From AWR -->

v$session s, v$process p where paddr=addr and p.pid=30849 order by p.pid;

select spid "host-pid",p.pid, s.sid, s.serial#, p.program, s.machine from gv$session s, gv$process p where paddr=addr and s.sid=&sid order by p.pid;

<-- Killing a session (Good Query)

select s.sid, s.serial#, s.username,
to_char(s.logon_time,'DD-MON HH24:MI:SS') logon_time,
p.pid oraclepid, p.spid "ServerPID", s.process

select sys_context ('USERENV' , 'SID') OwnSID from dual;
select sys_context('USERENV','SESSION_USER') current_user,sys_context('USERENV','SESSION_SCH

select distinct sid OwnSID from v$mystat;

alter session set events 'immediate crash';

select 'alter system kill session ' || '''' || sid || ',' || serial# ||',@'|| inst_id || '''' || ' immediate;' from gv$session where sid='&sid';

select ' alter system kill session '''||sid||''','''||serial#||''',''@'||inst_id||''' immediate; ' from gv$session where username in ('SCHEMA1','SCHEMA2') and logon_time < sysdate -1 and status='INACTIVE';

select ' alter system kill session '''||sid||','||serial#||',@'||inst_id||''' immediate; ' -- 4 hours
from gv$session where username in ('SCHEMA1','SCHEMA2','SCHEMA3') and status='INACTIVE' and last_call_et > 4*60*60;

```
ps -ef | grep "oracleinstname (LOCAL=NO)"
```

Then kill the old sessions using kill -9 spid

```
kill -9 `ps -ef|grep LOCAL=NO|grep oratst1|awk
'{print $2}'`
```

```
SELECT a.sid, a.username, b.xidusn rollback_seg_no,
b.used_urec undo_records, b.used_ublk undo_blocks
FROM gv$session a, gv$transaction b
WHERE a.saddr = b.ses_addr;
```

```
SELECT Logon_time,
       (SELECT ROUND (VALUE / 1024 / 1024, 2)
          FROM v$sesstat
SELECT o.sid,
      o.sql_text,
      o.address,
      o.hash_value,
      o.user_name,
      s.schemaname,
      o.sql_id
  FROM v$open_cursor o, v$session s
 WHERE o.saddr = s.saddr AND o.sid = s.sid AND
(O.SID = &sid)
"ID",B.BEGIN_INTERVAL_TIME,B.END_INTERVAL_T
IME,A.RESOURCE_NAME,
CURRENT_UTILIZATION
"CURRENT",MAX_UTILIZATION "MAX"
FROM WRH$_RESOURCE_LIMIT A,
WRM$_SNAPSHOT B
SELECT SQL_ID, SQL_FULLTEXT, PLAN_HASH_VALUE, PARSING_SCHEMA_NAME, ELAPSED_TIME FR
select
      count(*),
      CASE WHEN state != 'WAITING' THEN
'WORKING'
          ELSE 'WAITING'
```

```
-- Purge explain plan from shared pool, must be done on each RAC node
select inst_id, sql_id, address, hash_value, plan_hash_value, sql_text from gv$sqlarea where sql_text like 'SELECT COUNT(*) FROM V$SESSION A WHERE A.AUDSID IN%';
select * from gv$sqlarea where sql_text like '%XLA_AE_LINES_GT%';
select inst_id, sql_id, address, hash_value, plan_hash_value, sql_text from gv$sqlarea where sql_id = 'amd3xmgb8cnuz';
alter session force parallel query;

alter session enable parallel DML;
/*+ PARALLEL */
/*+ PARALLEL, 8 */
/*+ NOPARALLEL */

select xidusn, xidsqn from v$transaction;
```

```
declare
   v_sql CLOB;
begin
   select sql_text into v_sql from dba_hist_sqltext where sql_id='5273fz2cqkk80';
   sys.dbms_sqldiag_internal.i_create_patch(
      sql_text  => v_sql,
      hint_text => 'DYNAMIC_SAMPLING(4)',
      name      => '5273fz2cqkk80_patch');
end;
```

https://avdeo.com/2012/12/14/oracle-sql-patch-i/

check	ENABLE/DISABLE/DROP

select NAME,SIGNATURE ,STATUS,FORCE_MATCH	EXEC DBMS_SQLTUNE.ALTER_SQL_PROFILE('coe_5273fz2cqkk80_3455548535','STATUS','DISABLED');
	
	exec dbms_sqltune.drop_sql_profile('coe_5273fz2cqkk80_3455548535');

check	Drop

select SQL_HANDLE, PLAN_NAME, ENABLED, ACCEPTED, FIXED ,sql_text   from dba_sql_plan_baselines;
 	
select * from table(dbms_xplan.display_sql_plan_baseline(plan_name=>'SQL_PLAN_b7pnq6yp8da6a29d0d9b7'));
	DECLARE
	 i NATURAL;
	BEGIN
	  i := dbms_spm.drop_sql_plan_baseline('SQL_b3d69637aa86a8ca');
	  dbms_output.put_line(i);
	END;
	/

All in one script	**Taking SQL HINTS from memory**

```
set pagesize 60
set linesize 180
set trimspool on

column  plan_name format a32
column  signature format
999,999,999,999,999,999,999
column  category  format a10
column  hint format a70 wrap word
```

Resolution normally, is to fix the execution plan in 11g by running

```
variable x number
begin

BEGIN
 FOR i IN (SELECT address, hash_value
 FROM gv$sqlarea WHERE sql_id = '&sql_id.')
 LOOP

SYS.DBMS_SHARED_POOL.PURGE(i.address||','||i.h
ash_value, 'C');
 END LOOP;
END;
/
```

```
SELECT
chr(9)||chr(9)||''''||regexp_replace(extractvalue(value(d
), '/hint'),'''','''''')||''','
 from
 xmltable('/*/outline_data/hint'
 passing (
 select
 xmltype(other_xml) as xmlval
 from
```

To see the hints from sql_profile

```
select hint from (
select s.sql_id, sd.obj_type,
row_number() over (partition by sd.signature,
sd.category order by sd.signature) row_num,
```

http://expertoracle.com/2015/07/08/flush-bad-sql-plan-f

```
---------------------------------------------------------------------
---------------------------------------------------------------------
------
---
---   one script to Identify all performace issue related
to sql query
```

```
set pages 500
set linesize 750
column box format a30
column spid format a10
column username format a30
column sid format 9999
column serial for 999999
column status format a15
column username format a10
column sql_text format a80
```

```
set head off
set verify off
set echo off
set pages 1500
set linesize 100
```

Missing statements in SQL Monitoring -->

```
ALTER SESSION SET
"_SQLMON_MAX_PLAN"=4020;
ALTER SESSION SET
"_SQLMON_MAX_PLANLINES"=4000;
```

```
COLUMN sid FORMAT 99999
COLUMN serial# FORMAT 9999999
COLUMN machine FORMAT A30
COLUMN progress_pct FORMAT 99999999.00
COLUMN elapsed FORMAT A10
```

```
SELECT SID, SERIAL#, OPNAME,
TARGET, SOFAR, TOTALWORK,
UNITS,
TO_CHAR(START_TIME,'DD/MON/YYY
Y HH24:MI:SS') START_TIME,
```

Time model

```
select stat_name, value
from V$SESS_TIME_MODEL
where sid = &sid
order by value desc;
```

Stats

```
 select vsn.name, vst.value
from v$sesstat vst, v$statname vsn
where vsn.statistic# = vst.statistic#
and vst.value  != 0
and vst.sid = &sid
```

Explain Plan waiting steps

```
col PLAN FOR a150
SELECT
    RPAD('(' || p.plan_line_ID || ' ' ||
NVL(p.plan_parent_id,'0') || ')',8) || '|' ||
    RPAD(LPAD (' ', 2*p.plan_DEPTH) ||
```

https://blog.yannickjaquier.com/oracle/real-time-sql-monitoring.html

AWR

Time based

SELECT s.snap_id,TO_CHAR(s.begin_interval_time, 'DD-MON HH24:MI') snap_time,ss.sql_id,ss.plan_hash_value,	s.sql_id, sum(case when begin_interval_time = to_date('14-nov-2017 1100','dd-mon-yyyy hh24mi')
Each Layer time spend	Explain Plan waiting steps

"Elapsed (s)",
    ROUND(cpu_time   /1000000,3)   AS
"CPU (s)",
    ROUND(queuing_time   /1000000,3)   AS
	SELECT
	    RPAD('(' || p.plan_line_ID || ' ' ||
	NVL(p.plan_parent_id,'0') || ')',8) || '|' ||
	    RPAD(LPAD (' ', 2*p.plan_DEPTH) ||

AWR-LIO	Current Memory
col avg_etime for 999,999 col avg_lio for 999,999,999 col avg_pio for 999,999,999 col begin_interval_time for a30 col node for 99999	col avg_et_secs justify right format 9999999.99 col cost justify right format 9999999999 col timestamp justify center format a25 col parsing_schema_name justify center
bind_variable	

col VALUE_STRING for a50
SELECT
NAME,POSITION,DATATYPE_STRING,VALUE_STRING FROM gv$sql_bind_capture WHERE
sql_id='&sql_id';	

s.program, t.xidusn, t.used_ublk, t.used_urec, sa.sql_text from v$process p,v$session s, v$sqlarea sa, v$transaction t	dbms_stats.flush_database_monitoring_info; select inserts,updates,deletes from

Wait Percentage % (Only wait events-**WITH CPU**)

```
col time_cat format a20 heading "Time category"
col time_secs format 999,999.99 "Time (s)"
col pct format 99.99 "Time|pct"
set lines 750 pagesize 10000
```

```
SELECT wait_class time_cat, ROUND((time_secs),
```

```
SELECT   event, total_waits,
        ROUND (time_waited_micro / 1000000) AS
time_waited_secs,
        ROUND (time_waited_micro * 100 /
           SUM (time_waited_micro) OVER (),2) AS
pct_time
    FROM (SELECT event, total_waits,
```

```
set lines 750 pages 9999
select * from (
 select
   WAIT_CLASS ,
   EVENT,
   count(sample_time) as EST_SECS_IN_WAIT
 from v$active_session_history
```

```
    event,
    time_waited "time_waited(s)",
    case when time_waited = 0 then
        0
```

http://guyharrison.squarespace.com/opsgsamples/

```
select * from (
select EVENT, WAIT_CLASS,
SUM(TOTAL_WAITS),round
(sum(TIME_WAITED_micro)/1000000,0)
as time_waited_secs  from
V$SYSTEM_EVENT
where wait_class != 'Idle' group by
```

```
select expl.*
from
gv$sql sql, v$session ses,
TABLE(dbms_xplan.display_cursor(sql.sql_id,
sql.child_number,format=>'typical +predicate')) expl
where ses.sql_address = sql.address and ses.sid
= &&1
```

http://www.centroid.com/blog/monitoring-exadata-smart-scan

```
select * from
table(dbms_xplan.display_awr('&sql_id', null, null,
'ALLSTATS LAST'));
```

https://hoopercharles.wordpress.com/2010/03/01/dbms_xplan-format-parameters/

```
select vs.sid, vs.username, vs.osuser, vs.process
fg_pid,
vp.spid bg_pid
from v$session vs, v$process vp
```

```
select sys_context('USERENV', 'IP_ADDRESS')
from dual;
```

```
begin
 for x in (select
SID,Serial#,status,last_call_et,username from
v$session
where type='USER'
and status='INACTIVE'
and username = 'TESTER'
and last_call_et > 1200
             ) loop
        execute 'Alter System Kill Session '''|| x.Sid
                     || ',' || x.Serial# || '''';
    end loop;
end;
/
```

```
select 'ALTER SYSTEM KILL SESSION
'''||SID||','||SERIAL#||'''IMMEDIATE;' from
v$session where username = 'A' and
STATUS = 'ACTIVE';
```

```
for i in `ps -ef |grep "oracledb10g1 (LOCAL=NO)"
|grep -v grep | awk '{print $2}'`
do
 echo  kill -9  $i
done
```

```
set linesize 30
spool kill_old_sess.sh
select '#!/bin/ksh' from dual;
select 'kill -9 ' || spid
from v$process p, v$SESSTAT
t,v$sess_io i ,v$session s
where i.sid=s.sid
        and p.addr=paddr(+)
```

```
OM V$SQL WHERE UPPER(PARSING_SCHEMA_NAME) ='SYS' ORDER BY ELAPSED_TIME
```

VAR signature NUMBER;
VAR signaturef NUMBER;
REM
DECLARE
sql_txt CLOB;
output   varchar2(1000);
h   SYS.SQLPROF_ATTR;
PROCEDURE wa (p_line IN VARCHAR2) IS
BEGIN
DBMS_LOB.WRITEAPPEND(sql_txt,
	<= My own sql Patch

sql Profiles for a sql_id	
set lines 1000 pages 9999 col name for a30 col task_exec_name for a16 col category for a10 col created for a30 col sql_text for a150 col signature for 999999999999999999999999	
load from cursor	Enable/disable

variable sqlid number;
execute :sqlid :=DBMS_SPM.LOAD_PLANS_FROM_CURSOR_CACHE(sql_id=>'5qbbnv0abm2vx', PLAN_HASH_VALUE=> 4197102931, SQL_HANDLE => 'SQL_d3318f33dfac7bc2');
	DECLARE
	l_plans_altered PLS_INTEGER;
	BEGIN
	l_plans_altered := DBMS_SPM.alter_sql_plan_baseline(
	sql_handle => 'SYS_SQL_1447ba3a1d83920f',
	plan_name => 'SYS_SQL_PLAN_1d83920fae82cf72',

Taking SQL HINTS from AWR	**Taking SQL HINTS from PROFILES**

```
SELECT
chr(9)||chr(9)||''''||regexp_replace(extractvalue(value(
d), '/hint'),'''','''''')||''','
 from
 xmltable('/*/outline_data/hint'
 passing (
 select
 xmltype(other_xml) as xmlval
 from
```

```
  select hint as outline_hints
   from (select p.name, p.signature,
p.category, row_number()
         over (partition by sd.signature,
sd.category order by sd.signature)
row_num,
         extractValue(value(t), '/hint') hint
   from sqlobj$data sd, dba_sql_profiles
```

http://intermediatesql.com/oracle/what-are-sql-profiles-and-why-do-we-need-them/

http://www.alfredokriegdba.com/2015/02/

from-shared-pool/

http://allappsdba.blogspot.com/2012/04/queries-to-get-session-information.html

```
NO_XML_QUERY_REWRITE */
t.report_id, x1.sql_id, x1.plan_hash,
x1.sql_exec_id,
x1.elapsed_time/1000000 ELAP_SEC
FROM dba_hist_reports t
```

```
report_id,
key1
sql_id,
key2
sql_exec_
```

uro-pagano.com/2015/05/04/historical-sql-

```
column username format a20
Select s.username, s.Sid, s.serial#,
S.Sql_Id, round((Sysdate-
Sql_Exec_Start)*24*60*60/60,0)
MINUTES, Sql_Text
```

```
(
   select
      sql_id,
      sql_plan_hash_value,
```

https://blog.yannickjaquier.com/oracle/real-time-sql-monitoring.html

http://www.tek-tips.com/viewthread.cfm?qid=1572803

DESC

Fix baseline of one sql_id to another

```
variable sqlid number;
execute :sqlid
:=DBMS_SPM.LOAD_PLANS_FROM_
CURSOR_CACHE(sql_id=>'31pux6bym
f1d4');


SQL> select sql_handle, plan_name,
enabled from dba_sql_plan_baselines;
```

Taking SQL HINTS from PLAN_TABLE

```
SELECT
regexp_replace(extractvalue(value(d),
'/hint'),'''','''''') plan_hint
        from
        xmltable('/*/outline_data/hint'
                passing (
                        select
                                xmltype(other_xml)
as xmlval                                         https://jonathanlewis.wordpress.com/2017/06/12/dbms_sqldiag/
```