# oracle 10g 研究ORACLE_HOME rdbms admin 下的脚本的功能（14）awrinfo.sql

oracle 10g 研究ORACLE_HOME rdbms admin 下的脚本的功能（14）awrinfo.sql

```
#Script to output general AWR information
#This script will output general Automatic Workload Repository
Rem      (AWR) information such as the size, data distribution, etc. in AWR
Rem      and SYSAUX. The intended use of this script is for diagnosing
Rem      abnormalities in AWR and not for diagnosing issues in the database
Rem      instance. Please look at addmrpt.sql and awrrpt.sql for diagnosing
Rem      database issues.
Rem
Rem      The following information will be displayed:
Rem
Rem         (I) AWR Snapshot Info gathering
Rem        (II) Advisor framework diagnostics
Rem       (III) AWR and ASH Usage Info Gathering
#

set echo off
Rem
Rem $Header: awrinfo.sql 27-aug-2004.09:40:32 mlfeng Exp $
Rem
Rem awrinfo.sql
Rem
Rem Copyright (c) 2003, 2004, Oracle. All rights reserved.
Rem
Rem    NAME
Rem      awrinfo.sql - Script to output general AWR information
Rem
Rem    DESCRIPTION
Rem      This script will output general Automatic Workload Repository
Rem      (AWR) information such as the size, data distribution, etc. in AWR
Rem      and SYSAUX. The intended use of this script is for diagnosing
Rem      abnormalities in AWR and not for diagnosing issues in the database
Rem      instance. Please look at addmrpt.sql and awrrpt.sql for diagnosing
Rem      database issues.
Rem
Rem      The following information will be displayed:
Rem
Rem         (I) AWR Snapshot Info gathering
Rem        (II) Advisor framework diagnostics
Rem       (III) AWR and ASH Usage Info Gathering
Rem
Rem    NOTES
Rem
Rem    MODIFIED   (MM/DD/YY)
Rem    mlfeng      08/06/04 - change wrh$_sqlbind
Rem    veeve       06/04/04 - fix object_space_usage()
Rem    veeve       04/15/04 - use :select_valid_rows in the Advisor sections
Rem    mlfeng      02/03/04 - read dbid from dbms_swrf_internal
Rem    mlfeng      01/16/04 - create/drop package
Rem    mlfeng      01/06/04 - sysaux occupants information
Rem    veeve       12/22/03 - add some advisor framework diagnostics
Rem    veeve       12/11/03 - warning on ash_eflush_status
Rem    mlfeng      12/05/03 - mlfeng_temp_orderby_statid
Rem    veeve       12/04/03 - removed ROLLSTAT, added more warnings
```

```
Rem    mlfeng      12/04/03 - changed wrm$_wr_control
Rem    veeve       12/03/03 - added warnings at the top
Rem    veeve       12/02/03 - coalesce related sections
Rem    veeve       12/01/03 - Some optimizations, more info in I.1, pretty print
Rem    mlfeng      11/26/03 - Ask for report name, output sections.
Rem    mlfeng      11/26/03 - Created
Rem

set feedback off verify off timing off echo off;

--
-- Prompt the User for the report file name (specify default),
-- then begin spooling
--
set termout off;
column dflt_name new_value dflt_name noprint;
select 'awrinfo.txt' dflt_name from dual;
set termout on;

prompt
prompt This script will report general AWR information
prompt ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
prompt
prompt
prompt Specify the Report File Name
prompt ~~~~~~~~~~~~~~~~~~~~~~~~~~~
prompt The default report file name is &dflt_name..  To use this name,
prompt press <return> to continue, otherwise enter an alternative.
prompt

set heading off;
set feedback off;
column report_name new_value report_name noprint;
select 'Using the report name ' || nvl('&&report_name','&dflt_name')
     , nvl('&&report_name','&dflt_name') report_name
  from sys.dual;

/*****************************************
 * Create package to be used by awrinfo.sql
 *****************************************/
create or replace package AWRINFO_UTIL as

  -----------------------------------------------
  -- get_perc_usage - Get the percent space usage
  -----------------------------------------------
  function  get_perc_usage( segment_owner varchar2,
                            segment_name varchar2,
                            segment_type varchar2,
                            partition_name varchar2 )
    return number;

  -----------------------------------------------
  -- get_type - Get the type for AWR table
  -----------------------------------------------
  function get_type( segment_name varchar2 )
    return varchar2;

  -----------------------------------------------
```

```
  -- classify_count - Get the classify count
  ------------------------------------------------
  function classify_count( cnt number )
    return varchar2;

end AWRINFO_UTIL;
/
show errors;

-- Create package body
create or replace package body AWRINFO_UTIL as

------------------------------------------------
-- get_perc_usage - Get the percent space usage
------------------------------------------------
function  get_perc_usage( segment_owner varchar2,
                          segment_name varchar2,
                          segment_type varchar2,
                          partition_name varchar2 )
return number
is
  space_used          number;
  space_allocated     number;
  chain_pcent         number;                          /* not used */
  sample_control      number := 100;
begin
    dbms_space.object_space_usage(segment_owner, segment_name, segment_type,
                                  sample_control, space_used, space_allocated,
                                  chain_pcent, partition_name);

    return (space_used*100/space_allocated);
exception
  /* skip table if error encountered */
  when others then
    return null;
end get_perc_usage;


------------------------------------------------
-- get_type - Get the type for AWR table
------------------------------------------------
function get_type( segment_name varchar2 )
return varchar2
is
begin
  return  (case when segment_name like '%ACTIVE_SESSION_HIST%' then 'ASH'
                when segment_name like 'WRH$_ASH_BL_PK' then 'ASH'

                when segment_name like '%EVENT%' then 'EVENTS'
                when segment_name like '%ENQ%' then 'EVENTS'
                when segment_name like '%LIBRARYCACHE%' then 'EVENTS'

                when segment_name like '%FILE%' then 'SPACE'
                when segment_name like '%TEMP%' then 'SPACE'
                when segment_name like '%TABLESPACE%' then 'SPACE'
                when segment_name like '%SEG%' then 'SPACE'

                when segment_name like '%SQL_BIND%' then 'SQLBIND'
```

```
                when segment_name like '%SQLTEXT%' then 'SQLTEXT'

                when segment_name like '%SQL_PLAN%' then 'SQLPLAN'

                when segment_name like '%SQL%' then 'SQL'
                when segment_name like '%OPTIMIZER%' then 'SQL'

                when segment_name like '%DLM%' then 'RAC'
                when segment_name like '%CR_BLOCK_SERVER%' then 'RAC'
                when segment_name like '%CURRENT_BLOCK_SERVER%' then 'RAC'
                when segment_name like '%CLASS_CACHE_TRANSFER%' then 'RAC'

                else 'FIXED'
            end);
end get_type;

-------------------------------------------
-- classify_count - Get the classify count
-------------------------------------------
function classify_count( cnt number )
return varchar2
is
  blim   number;
  elim   number;
begin

  blim := trunc(cnt/5) * 5;
  elim := (trunc((cnt + 5)/5) * 5) - 1;

  return( trim(to_char(blim, '0990')) || ' - ' || trim(to_char(elim, '0990')) );

end classify_count;

end;
/
show errors;

set termout on;
spool &report_name

set linesize 110
set pagesize 50

set serveroutput on
execute dbms_output.enable(1000000);

alter session set nls_date_format      = 'HH24:MI:SS (MM/DD)';
alter session set nls_timestamp_format = 'HH24:MI:SS (MM/DD)';

set echo off
variable dbid  number;
variable instid number;

begin
  :dbid := dbms_swrf_internal.get_awr_dbid;

  select instance_number into :instid from v$instance;
```

```
end;
/

set echo off
column db_id format a12 just r;
column name format a20
column platform_name format a30
column host_platform format a40 wrap
column startup_time format a17
column inst format 9999

prompt ~~~~~~~~~~~~~~~
prompt AWR INFO Report
prompt ~~~~~~~~~~~~~~~

set heading off
select 'Report generated at', to_char(systimestamp, 'HH24:MI:SS "on" Mon DD, YYYY ( ')
                        || trim(to_char(systimestamp, 'Day'))
                        || to_char(systimestamp, ' ) "in Timezone" TZR')
from dual;

col attr_nl newline;
select 'Warning: CATPROC Not Valid! ',
       '--------------------------------------------------------------------------------' attr_nl,
       rpad('Status: ', 30) || r.status || ' (1 => VALID) ' attr_nl,
       (case when (r.version <> i.version)
             then rpad('Version: ', 30) || 'Mismatch - CATPROC Version ' || r.version
                  || ', V$INSTANCE Version ' || i.version
             else rpad('Version: ', 30) || r.version
        end) attr_nl,
       rpad('Flags: ', 30) || r.flags attr_nl,
       rpad('Schema#/Invoker#:', 30) || r.schema# || '/' || r.invoker# attr_nl,
       rpad('Last Modified: ', 30) || r.modified attr_nl,
       rpad('Loading/Loaded:',30) || r.date_loading || '/' || r.date_loaded attr_nl,
       rpad('Upgrading/Upgraded:',30) || r.date_upgrading || '/' || r.date_upgraded attr_nl,
       rpad('Downgrading/Downgraded:',30) || r.date_downgrading || '/' || r.date_downgraded attr_nl,
       rpad('Removing/Removed:', 30) || r.date_removing || '/' || r.date_removed attr_nl,
       rpad('Valid/Invalid:', 30) || r.date_valid || '/' || r.date_invalid attr_nl,
       rpad('Version Original/Previous:', 30) || r.org_version || '/' || r.prv_version attr_nl
from   sys.registry$ r, v$instance i
where  r.cid = 'CATPROC'
  and  r.status <> 1;


col attr_nl newline;
select 'Warning: Non Standard SYSAUX Configuration! ',
       '--------------------------------------------------------------------------------' attr_nl,
       rpad('Status: ', 30) || t.status attr_nl,
       rpad('Contents: ', 30) || t.contents attr_nl,
       rpad('Block Size: ', 30) || t.block_size attr_nl,
       rpad('Logging: ', 30) || t.logging attr_nl,
       rpad('Extent Management: ', 30) || t.extent_management attr_nl,
       rpad('Segment Space Management: ', 30) || t.segment_space_management attr_nl
from   dba_tablespaces t
where  t.tablespace_name = 'SYSAUX'
  and  (t.extent_management <> 'LOCAL'
        or t.segment_space_management <> 'AUTO'
        or t.status <> 'ONLINE'
```

```
       or t.contents <> 'PERMANENT');
set heading on

col sysaux_df_not_avail  format a80 heading 'Warning: SYSAUX datafiles not AVAILABLE';
define  file_name_max = 40;

select 'File ' ||
       (case when length(file_name) > &file_name_max
             then substr(file_name, 1, &file_name_max/2)
                  || '...'
                  || substr(file_name, length(file_name) - &file_name_max/2 + 1, &file_name_max)
             else file_name end)
       || ' with ID ' || file_id
       || ' has status ' || status as sysaux_df_not_avail
from   dba_data_files
where  tablespace_name = 'SYSAUX'
  and  status <> 'AVAILABLE';


variable  stat_lvl_prm_hash  number;

begin
  select hash into :stat_lvl_prm_hash
  from   v$parameter
  where  lower(name) = 'statistics_level';
end;
/

col stat_lvl_alter  format a80 heading 'Warning: STATISTICS_LEVEL Altered!';
select 'DB ' || dbid || ' Instance ' || instid || ':'
       || ' From ' || rpad(value,7) || ' in snap ' || snap_id
       || ' To '   || rpad(lead_value,7) || ' in snap ' || lead_snap_id stat_lvl_alter
from   (select value, lead(value, 1) over (order by dbid, instance_number, snap_id) as lead_value,
               rank() over (partition by dbid, instance_number order by snap_id desc) as snap_rank,
               snap_id, lead(snap_id, 1) over (order by dbid, instance_number, snap_id) as lead_snap_id,
               dbid,  lead(dbid, 1)  over (order by dbid, instance_number, snap_id) as lead_dbid,
               instance_number instid,  lead(instance_number, 1)  over (order by dbid, instance_number, snap_id)
               as lead_instid
        from   dba_hist_parameter
        where  dbid = :dbid
          and  parameter_hash = :stat_lvl_prm_hash)
where  lower(value) <> lower(lead_value)
  and  dbid = lead_dbid
  and  instid = lead_instid
  and  lead_value is not null;

col nonstd_params format a80 heading 'Warning: Few Underscore Parameters Altered!';
select 'Parameter ' || n.ksppinm || ' set to ' || v.ksppstdvl nonstd_params
from   x$ksppi n, x$ksppsv v
where  n.indx = v.indx
  and  (lower(n.ksppinm) like '\_addm%' escape '\'
        or  lower(n.ksppinm) like '\_ash%'  escape '\'
        or  lower(n.ksppinm) like '\_awr%'  escape '\'
        or  lower(n.ksppinm) like '\_swrf%'  escape '\')
  and  (v.ksppstdf <> 'TRUE'
        or bitand(ksppstvf,7) = 1);

col awr_status  format a80 heading 'Warning: Non Default AWR Setting!';
```

```
select  'Snapshot interval is ' || round(w.snapint_num/60) || ' minutes and '
        || 'Retention is ' || round(w.retention_num/86400) || ' days' as awr_status
from    wrm$_wr_control w
where   w.dbid = :dbid
  and   (w.snapint_num <> 3600
         or w.retention_num <> 604800);

col ash_status  format a80 heading 'Warning: Abnormal ASH Status!';
select status ash_status
from   (select (case  when (bitand(m.flags, 16) = 0)  then 'Disabled (No signal from run-once MMON)'
                      when (bitand(m.flags,  1) <> 0) then 'Disabled (Using _ASH_ENABLE)'
                      when (bitand(m.flags,  2) <> 0) then 'Enabled (Sample all connected sessions)'
                      when (bitand(m.flags,  4) <> 0) then 'Enabled (Flushing is disabled)'
                      when (bitand(m.flags,  8) <> 0) then 'Enabled (Latest flush had errors)'
                      else  'Enabled' end) status
        from   x$kewam m)
where  status <> 'Enabled';

col ash_eflush_status  format a80 heading 'Interesting Info: Significant Number of ASH On-demand Flushes!';
select 'ASH On-demand Flushing % = ( '
       || m.emergency_flusher_count || '/' || m.total_flusher_count
       || ' ) =' || to_char(m.emergency_flusher_count * 100 / m.total_flusher_count, '990.0') || '%'
       ash_eflush_status
from   x$kewam m
where  m.total_flusher_count > 0
  and  m.emergency_flusher_count * 10 > m.total_flusher_count;

prompt
select (a.cur || a.dbid) as db_id, a.db_name, a.host_platform,
       a.inst, a.startup_time, wdi.last_ash_sample_id as last_ash_sid,
       a.parallel
from   (select (case when awr.dbid = :dbid and
                          awr.instance_number = :instid then '* '
                else '  ' end ) as Cur,
                awr.dbid, max(awr.db_name) as db_name,
                max(awr.host_name) || ' - ' ||
                (case when awr.dbid = :dbid then max(d.platform_name)
                      else '' end ) as host_platform,
                awr.instance_number inst,
                max(awr.startup_time) as startup_time, max(awr.parallel) as parallel
        from    wrm$_database_instance awr, v$database d
        group   by awr.dbid, awr.instance_number) a,
       wrm$_database_instance wdi
where  wdi.dbid = a.dbid
  and  wdi.instance_number = a.inst
  and  wdi.startup_time = a.startup_time
order by a.dbid;

prompt
prompt ############################################################
prompt  (I) AWR Snapshots Information
prompt ############################################################

prompt
prompt ***************************************************
prompt  (1a) SYSAUX usage - Schema breakdown (dba_segments)
prompt ***************************************************
```

```
column size_M format 999,990.0

Rem all following sizes are in MB
variable sysaux_size     number;
variable sysaux_maxsize  number;
variable sysaux_numfiles number;
variable sysaux_nf_autoext number;
variable sysaux_sz_autoext number;
variable sysaux_sys_size number;
variable sysaux_system_size number;
variable awr_size        number;
variable awr_size_avg    number;
variable awr_size_past24 number;
variable adv_size        number;
variable opt_size        number;
variable oth_size        number;
variable mis_size        number;
variable snap_size       number;
variable prevday_size    number;

variable num_inst        number;

variable all_snaps       number;
variable good_snaps      number;
variable today_snaps     number;
variable est_today_snaps number;
variable num_days        number;

variable snap_intr       number;
variable ret_intr        number;

variable align           number;
variable mb_format       varchar2(30);
variable kb_format       varchar2(30);
variable perc_format     varchar2(30);

variable other_sysaux    varchar2(2000);
variable occ_space       number;
variable occ_sys_space   number;
variable occ_system_space number;

set heading off
declare
  cursor sysaux_usage is
  select owner, sum(bytes)/1024/1024 as owner_size,
         (sum(sum(bytes)) over ())/1024/1024 as total_size
  from   dba_segments
  where  tablespace_name = 'SYSAUX'
  group  by owner
  order  by sum(bytes) desc;

  i      number;

  cursor sysaux_occ_usage is
  select occupant_name, schema_name, space_usage_kbytes/1024 space_usage_mb
  from v$sysaux_occupants
  order by space_usage_kbytes desc, occupant_name;
```

```
  in_syxocc number;
  newlinech varchar2(1) := '
';

  occ_fmt number;
  sch_fmt number;
  space_fmt number;

  total_unacc_unreg  number;

begin

  :align := 35;
  :mb_format := '99,999,990.0';
  :kb_format := '999,999,990';
  :perc_format := '990.0';

  i := 0;
  total_unacc_unreg := 0;

  :other_sysaux := '| This section displays schemas that are not registered ' ||
                   newlinech ||
                 '| in V$SYSAUX_OCCUPANTS ' || newlinech ||
                 '| ' || newlinech;

  select sum(case when autoextensible = 'YES' then (f.maxbytes)/1024/1024
              else (f.user_bytes)/1024/1024 end),
         count(*),
         sum(case when autoextensible = 'YES' then 1
                  else 0 end),
         sum(case when autoextensible = 'YES' then maxbytes/1024/1024
                  else 0 end)
  into   :sysaux_maxsize, :sysaux_numfiles, :sysaux_nf_autoext, :sysaux_sz_autoext
  from   dba_data_files f
  where  f.tablespace_name = 'SYSAUX';

  for sysaux_rec in sysaux_usage loop
    if (i = 0) then
      :sysaux_size := sysaux_rec.total_size;
      dbms_output.put_line( '|' );
      dbms_output.put( rpad('| Total SYSAUX size ', :align)
                       || to_char(sysaux_rec.total_size, :mb_format) || ' MB ( ' );

      if ( :sysaux_maxsize = 0) then
        -- just in case the maxsize is zero, finish writing the line */
        dbms_output.put_line(')');

      elsif ( :sysaux_nf_autoext > 0 ) then
        -- AUTOEXTEND ON
        if ( :sysaux_nf_autoext = :sysaux_numfiles ) then
          -- All files are with AUTOEXTEND ON
          dbms_output.put_line( round(:sysaux_size*100/:sysaux_maxsize) || '% of '
                                || trim(to_char(:sysaux_maxsize, :mb_format)) || ' MB MAX with AUTOEXTEND ON )' );
        else
          -- Some files with AUTOEXTEND ON
          dbms_output.put_line( 'AUTOEXTEND ON for ' || :sysaux_nf_autoext
                                || ' out of ' || :sysaux_numfiles || ' files )' );
          dbms_output.put_line( rpad('| Fixed limit  ', :align )
```

```
                            || to_char(:sysaux_maxsize - :sysaux_sz_autoext, :mb_format) || ' MB' );
        dbms_output.put_line( rpad('| Auto Extent limit ', :align )
                            || to_char(:sysaux_sz_autoext, :mb_format) || ' MB' );
      end if;
    else
      -- AUTOEXTEND OFF
      dbms_output.put_line( round(:sysaux_size*100/:sysaux_maxsize) || '% of '
                          || trim(to_char(:sysaux_maxsize, :mb_format)) || ' MB MAX with AUTOEXTEND OFF )' );
    end if;

    dbms_output.put_line( '|' );
    i := 1;
  end if;
  dbms_output.put_line( rpad('| Schema  ' || rpad(sysaux_rec.owner, 12) || ' occupies ', :align)
                      || to_char(sysaux_rec.owner_size, :mb_format) || ' MB ('
                      || to_char(sysaux_rec.owner_size*100/:sysaux_size, :perc_format) || '% )' );
  if (sysaux_rec.owner = 'SYS') then
    :sysaux_sys_size := sysaux_rec.owner_size;
  elsif (sysaux_rec.owner = 'SYSTEM') then
    :sysaux_system_size := sysaux_rec.owner_size;
  end if;

  /* run query to see if the schema belongs in v$sysaux_occupants */
  select count(*) into in_syxocc
    from dual
    where sysaux_rec.owner in (select schema_name from v$sysaux_occupants);

  /* remember those schemas that are not registered as occupants */
  if (in_syxocc = 0) then
    :other_sysaux := :other_sysaux ||
                     rpad('| Schema  ' || rpad(sysaux_rec.owner, 12) ||
                          ' occupies ', :align) ||
                     to_char(sysaux_rec.owner_size, :mb_format) ||
                     ' MB ' || newlinech;

    total_unacc_unreg := total_unacc_unreg + sysaux_rec.owner_size;

  end if;

end loop;

:other_sysaux := :other_sysaux || '| ' || newlinech;

:other_sysaux := :other_sysaux || rpad('| Total space', :align)
                    || to_char(total_unacc_unreg, :mb_format)
                    || ' MB ' || newlinech;

:other_sysaux := :other_sysaux || '| ';

-- set up the formatting
occ_fmt   := 20;
sch_fmt   := 20;
space_fmt := 16;

dbms_output.put_line( '| ' );
dbms_output.put_line( '**************************************************** ');
dbms_output.put_line( ' (1b) SYSAUX occupants space usage (v$sysaux_occupants)');
dbms_output.put_line( '**************************************************** ');
```

```
  dbms_output.put_line('|   ');
  dbms_output.put_line('| ' || rpad('Occupant Name', occ_fmt) || ' '
                            || rpad('Schema Name',   sch_fmt) || ' '
                            || lpad('Space Usage',   space_fmt));
  dbms_output.put_line('| ' || rpad('-', occ_fmt,   '-') || ' '
                            || rpad('-', sch_fmt, '-') || ' '
                            || lpad('-', space_fmt,   '-'));

  -- init occupant space
  :occ_space        := 0;
  :occ_sys_space    := 0;
  :occ_system_space := 0;

  -- loop through all the occupants
  for occ_rec in sysaux_occ_usage loop

    dbms_output.put_line('| ' || rpad(occ_rec.occupant_name, occ_fmt) || ' '
                              || rpad(occ_rec.schema_name,   sch_fmt) || ' '
                              || to_char(occ_rec.space_usage_mb, :mb_format) || ' '
                              || 'MB');

    -- increment occupant space
    :occ_space := :occ_space + occ_rec.space_usage_mb;

    -- remember space for occupants in SYS, SYSTEM
    if (occ_rec.schema_name = 'SYS') then
      :occ_sys_space := :occ_sys_space + occ_rec.space_usage_mb;
    elsif (occ_rec.schema_name = 'SYSTEM') then
      :occ_system_space := :occ_system_space + occ_rec.space_usage_mb;
    end if;

    -- check if occupant is AWR
    if (occ_rec.occupant_name = 'SM/AWR') then
      :awr_size := occ_rec.space_usage_mb;
    end if;

  end loop;

  dbms_output.put_line('| ' );

  dbms_output.put_line('| ' || rpad('Others (Unaccounted space)',
                                    occ_fmt + sch_fmt + 1) || ' '
                           || to_char(:sysaux_size-:occ_space, :mb_format) || ' '
                           || 'MB');

  dbms_output.put_line('| ' );

end;
/

prompt
prompt ****************************************
prompt  (1c) SYSAUX usage - Unregistered Schemas
prompt ****************************************

set long 10000;
select :other_sysaux from dual;
```

```
prompt ************************************************************
prompt  (1d) SYSAUX usage - Unaccounted space in registered schemas
prompt ************************************************************
prompt |
prompt | This section displays unaccounted space in the registered
prompt | schemas of V$SYSAUX_OCCUPANTS.


declare
  unacc_sys_system number;
  total_unacc_reg  number;

  cursor unaccounted_usage is
    select occ.schema_name schema_name,
           seg.owner_size - occ.space_usage_mb size_diff
      from
        (select schema_name, sum(space_usage_kbytes)/1024 space_usage_mb
           from v$sysaux_occupants
           group by schema_name) occ,
        (select owner, sum(bytes)/1024/1024 as owner_size
           from   dba_segments
           where  tablespace_name = 'SYSAUX'
           group  by owner) seg
      where
        occ.schema_name = seg.owner and
        occ.schema_name not in ('SYS', 'SYSTEM') and
        (seg.owner_size - occ.space_usage_mb) != 0;

begin

  dbms_output.put_line('|');

  total_unacc_reg := 0;

  -- unaccounted usage for SYS/SYSTEM
  unacc_sys_system := :sysaux_sys_size + :sysaux_system_size -
                      (:occ_sys_space + :occ_system_space);

  -- increment total unaccounted registered
  total_unacc_reg := total_unacc_reg + unacc_sys_system;

  dbms_output.put_line(rpad('| Unaccounted space in SYS/SYSTEM', :align)
                       || to_char(unacc_sys_system, :mb_format) || ' '
                       || 'MB');


  -- ##################################################
  -- Right now, there are no other schemas that will have unaccounted
  -- space inside the schema. This logic will be disabled to avoid
  -- an extra scan on dba_segments.  Enable this logic if there exists
  -- other schemas that have unaccounted for space inside a schema.
  -- ##################################################
  if (FALSE) then
    -- loop through the unaccounted usages for the other registered schemas
    for unacc_rec in unaccounted_usage loop

      dbms_output.put_line(rpad('| Unaccounted space in ' ||
```

```
                              unacc_rec.schema_name, :align)
                            || to_char(unacc_rec.size_diff, :mb_format) || ' '
                            || 'MB');

      total_unacc_reg := total_unacc_reg + unacc_rec.size_diff;
    end loop;
  end if;

  dbms_output.put_line('|');

  dbms_output.put_line(rpad('| Total space', :align)
                        || to_char(total_unacc_reg, :mb_format) || ' '
                        || 'MB');

  dbms_output.put_line('|');

end;
/


begin

  select sum(all_snaps), sum(good_snaps), sum(today_snaps),
         sysdate - min(begin_interval_time)
  into   :all_snaps, :good_snaps, :today_snaps, :num_days
  from   (select 1 as all_snaps,
                 (case when s.status = 0 then 1 else 0 end) as good_snaps,
                 (case when (s.end_interval_time > sysdate - 1)
                       then 1 else 0 end)
                 as today_snaps,
                 cast(s.begin_interval_time as date) as begin_interval_time
          from   wrm$_snapshot s
          where  dbid = :dbid);

  select count(distinct instance_number) into :num_inst
  from   wrm$_database_instance
  where  dbid = :dbid;

  :today_snaps := :today_snaps / :num_inst;

  select m.snapint_num, m.retention_num
  into   :snap_intr, :ret_intr
  from   wrm$_wr_control m
  where  m.dbid = :dbid;

  :snap_size := :awr_size/:all_snaps;
  :awr_size_avg := :snap_size*86400/:snap_intr;

  dbms_output.put_line('************************************');
  dbms_output.put_line(' (2) Size estimates for AWR snapshots');
  dbms_output.put_line('************************************');
  dbms_output.put_line('|   ');
  dbms_output.put_line('| Estimates based on '
                       || round(:snap_intr/60) || ' mins snapshot INTERVAL:');

  dbms_output.put_line( rpad('|    AWR size/day ', :align)
                       || to_char(:awr_size_avg, :mb_format)
                       || ' MB (' || trim(to_char(:snap_size*1024, :kb_format)) || ' K/snap * '
```

```
                        || round(86400/:snap_intr) || ' snaps/day)' );

  dbms_output.put_line( rpad('|    AWR size/wk ', :align)
                        || to_char(:awr_size_avg * 7, :mb_format)
                        || ' MB (size_per_day * 7) per instance' );

  if (:num_inst > 1) then
    dbms_output.put_line( rpad('|    AWR size/wk ', :align)
                          || to_char(:awr_size_avg * 7 * :num_inst, :mb_format)
                          || ' MB (size_per_day * 7) per database' );
  end if;

  if (:num_days < 1) then
    :est_today_snaps := round(:today_snaps / :num_days);
  else
    :est_today_snaps := :today_snaps;
  end if;

  :awr_size_past24 := :snap_size * :est_today_snaps;

  dbms_output.put_line('|  ');
  dbms_output.put_line('| Estimates based on '
                        || round(:today_snaps) || ' snaps in past 24 hours:' );

  dbms_output.put_line( rpad('|    AWR size/day ', :align)
                        || to_char(:awr_size_past24, :mb_format) || ' MB ('
                        || trim(to_char(:snap_size*1024, :kb_format)) || ' K/snap and '
                        || round(:today_snaps) || ' snaps in past '
                        || round(least(:num_days*24,24),1) || ' hours)' );

  dbms_output.put_line( rpad('|    AWR size/wk ', :align)
                        || to_char(:awr_size_past24 * 7, :mb_format)
                        || ' MB (size_per_day * 7) per instance' );

  if (:num_inst > 1) then
    dbms_output.put_line( rpad('|    AWR size/wk ', :align)
                          || to_char(:awr_size_past24 * 7 * :num_inst, :mb_format)
                          || ' MB (size_per_day * 7) per database' );
  end if;

  dbms_output.put_line('|  ');

end;
/

set heading on

prompt
prompt **********************************
prompt (3a) Space usage by AWR components (per database)
prompt **********************************

col component format a9;
col mb format 99,990.0 wrap;
col ti_bytes format a16 heading 'TABLE% : INDEX%';
col kb_per_snap format 999,999,990 wrap;
col mb_per_day format 999,990.0 wrap;
col mb_per_week format 999,990.0 wrap;
```

```
col segment_name format a69 wrap;
col segnm_pct_spc_used format a69 wrap heading 'SEGMENT_NAME - % SPACE_USED';
col segment_type format a15 wrap;
col perc format 990.0;
col awr_perc format 990.0 heading '% AWR';

select  component,
        sum(bytes)/1024/1024 as MB,
        sum(bytes)/1024/1024*100/:awr_size as awr_perc,
        sum(bytes)/1024/:all_snaps as kb_per_snap,
        sum(bytes)/:all_snaps*:est_today_snaps/1024/1024 as mb_per_day,
        sum(bytes)/:all_snaps*:est_today_snaps*7/1024/1024 as mb_per_week,
        lpad( round(sum(tbytes)*100/sum(bytes)), 5) || '% : '
             || round(sum(ibytes)*100/sum(bytes)) || '%' as ti_bytes
from    (select  awrinfo_util.get_type(segment_name) as component, bytes,
                 (case when segment_type like '%TABLE%' then bytes else 0 end) as tbytes,
                 (case when segment_type like '%INDEX%' then bytes else 0 end) as ibytes
          from   dba_segments
          where  (segment_name like 'WRH%' or segment_name like 'WRM%')
            and  tablespace_name = 'SYSAUX'
            and  owner = 'SYS')
group by component
order by sum(bytes) desc;

prompt
prompt ********************************
prompt (3b) Space usage within AWR Components (> 500K)
prompt ********************************

select  component, bytes/1024/1024 as MB,
        rpad( segment_name || (case when partition_name is null then ''
                                    else '.' || partition_name
                               end), 61 )
        || ' -'
        || to_char(awrinfo_util.get_perc_usage( owner, segment_name,
                                                segment_type, partition_name),
                   '990')
        || '%'
        as segnm_pct_spc_used,
        segment_type
from    (select owner, segment_name, partition_name, segment_type,
                awrinfo_util.get_type(segment_name) as component, bytes,
                rank() over (partition by awrinfo_util.get_type(segment_name)
                             order by bytes desc, segment_name asc, partition_name asc) as rnk,
                sum(bytes) over
                  (partition by awrinfo_util.get_type(segment_name)) as grp_bytes
          from   dba_segments
          where  (segment_name like 'WRH%' or segment_name like 'WRM%')
            and  tablespace_name = 'SYSAUX'
            and  owner = 'SYS')
where   rnk <= 30
  and   bytes > 500000
order   by grp_bytes desc, component asc, bytes desc;

prompt
prompt ********************************
prompt  (4) Space usage by non-AWR components (> 500K)
prompt ********************************
```

```
select 'NON_AWR  ' as component, s.size_m as mb,
       s.owner || '.' || s.segment_name
        || (case when s.partition_name is null then ''
                 else '.' || s.partition_name end) as segment_name,
       s.segment_type
from
  (select segment_name, partition_name, owner, bytes/1024/1024 size_M, segment_type
     from dba_segments
     where tablespace_name = 'SYSAUX'
       and not (segment_name like 'WRH%' or segment_name like 'WRM%')
       and bytes > 500000
     order by size_M desc) s
  where rownum <= 100;

prompt
prompt **********************************
prompt   (5a) AWR snapshots - last 50
prompt **********************************

set heading off
select 'Total snapshots in DB ' || dbid || ' Instance ' || instance_number || ' = ' || count(*)
from wrm$_snapshot
group by dbid, instance_number;
set heading on
column flush_elapsed format a20
column endtm         format a17
column startup_time  format a17
column status format 99
column errcnt format 9999

select dbid, snap_id, inst, flush_elapsed, endtm, startup_time, status, errcnt
from
  (select dbid, snap_id, instance_number as inst, flush_elapsed,
          end_interval_time endtm, startup_time,
          status, error_count errcnt,
          rank() over (partition by dbid, instance_number order by end_interval_time desc) rnk
   from wrm$_snapshot)
where rnk <= 50 order by dbid, snap_id, inst;

prompt
prompt **********************************
prompt   (5b) AWR snapshots with errors or invalid
prompt **********************************

set feedback 1000
column status format 9999
column count  format 9999

select sn.dbid, sn.snap_id, sn.instance_number inst,
       sn.end_interval_time endtm,
       sn.status, sn.error_count count, err.table_name, err.error_number errnum
       from wrm$_snapshot sn, wrm$_snap_error err
       where sn.snap_id = err.snap_id
         and sn.dbid = err.dbid
         and sn.instance_number = err.instance_number
order by sn.dbid, sn.snap_id, sn.instance_number;
```

```
set feedback off

prompt
prompt **********************************
prompt   (5c) AWR snapshots -- OLDEST Non-Baselined snapshots
prompt **********************************

select s1.dbid, s1.instance_number inst, s1.snap_id,
       s1.end_interval_time endtm,
       s1.status, s1.error_count
  from wrm$_snapshot s1,
       (select s2.dbid, min(s2.instance_number) inst_id,
               min(s2.snap_id) snid
          from wrm$_snapshot s2
          where not exists (select 1 from wrm$_baseline bl
                            where s2.dbid = bl.dbid
                              and s2.snap_id between bl.start_snap_id
                                                 and bl.end_snap_id)
          group by dbid) smin
  where s1.dbid = smin.dbid
    and s1.instance_number = smin.inst_id
    and s1.snap_id = smin.snid
  order by s1.dbid, inst;



prompt
prompt **********************************
prompt   (6) AWR Control Settings - interval, retention
prompt **********************************

-- wrm$_wr_control
column dbid           format 9999999999
column lsnapid        format 9999999
column lsplitid       format 9999999
column lsnaptime        format a14
column lpurgetime       format a14
column flag           format 9999
column interval       format a17
column retention      format a17
column vrsn            format 999

-- Display WR Control record
select dbid, MOST_RECENT_SNAP_ID lsnapid, MOST_RECENT_SPLIT_ID lsplitid,
  to_char(MOST_RECENT_SNAP_TIME, 'MM/DD hh24:mi:ss') lsnaptime,
  to_char(MOST_RECENT_PURGE_TIME, 'MM/DD hh24:mi:ss') lpurgetime,
  STATUS_FLAG flag, snap_interval Interval, retention,
  SWRF_VERSION vrsn
from   wrm$_wr_control
order by dbid;



prompt
prompt **********************************
prompt   (7a) AWR Contents - row counts for each snapshots
prompt **********************************

select snap_id, inst, ash, sql, sqbnd, files, segst, sysevt
  from (select sn.dbid, sn.instance_number inst, sn.snap_id,
```

```
          nvl(ash.cnt,0) ash, nvl(sql.cnt,0) sql,
          nvl(sqbnd.cnt,0) sqbnd, nvl(files.cnt,0) files,
          nvl(seg.cnt,0) segst,
          nvl(evt.cnt,0) sysevt
    from wrm$_snapshot sn,
      (select dbid, instance_number, snap_id, count(*) cnt
         from wrh$_active_session_history  where dbid = :dbid
         group by dbid, snap_id, instance_number) ash,
      (select dbid, instance_number, snap_id, count(*) cnt
         from wrh$_sqlstat  where dbid = :dbid
         group by dbid, snap_id, instance_number) sql,
      (select dbid, instance_number, snap_id, count(*) cnt
         from dba_hist_sqlbind  where dbid = :dbid
         group by dbid, snap_id, instance_number) sqbnd,
      (select dbid, instance_number, snap_id, count(*) cnt
         from wrh$_filestatxs  where dbid = :dbid
         group by dbid, snap_id, instance_number) files,
      (select dbid, instance_number, snap_id, count(*) cnt
         from wrh$_seg_stat  where dbid = :dbid
         group by dbid, snap_id, instance_number) seg,
      (select dbid, instance_number, snap_id, count(*) cnt
         from wrh$_system_event  where dbid = :dbid
         group by dbid, snap_id, instance_number) evt
     where (sn.snap_id = ash.snap_id(+)
           and sn.instance_number = ash.instance_number(+))
       and (sn.snap_id = sql.snap_id(+)
           and sn.instance_number = sql.instance_number(+))
       and (sn.snap_id = sqbnd.snap_id(+)
           and sn.instance_number = sqbnd.instance_number(+))
       and (sn.snap_id = files.snap_id(+)
           and sn.instance_number = files.instance_number(+))
       and (sn.snap_id = seg.snap_id(+)
           and sn.instance_number = seg.instance_number(+))
       and (sn.snap_id = evt.snap_id(+)
           and sn.instance_number = evt.instance_number(+))
       and sn.dbid = :dbid
     order by snap_id desc, inst desc) s1
  where rownum <= 50 order by snap_id asc, inst asc;


prompt
prompt **********************************
prompt   (7b) AWR Contents - average row counts per snapshot
prompt **********************************


select  count(*) snap_count, s1.inst,
        round(avg(ash),2) ASH, round(avg(sql),2) SqlStat, round(avg(sqbnd),2) SQLBind,
        round(avg(files),2) Files,
        round(avg(segst),2) SegStat, round(avg(sysevt),2) SysEvent
  from (select sn.dbid, sn.instance_number inst, sn.snap_id,
          nvl(ash.cnt,0) ash, nvl(sql.cnt,0) sql,
          nvl(sqbnd.cnt,0) sqbnd, nvl(files.cnt,0) files,
          nvl(seg.cnt,0) segst,
          nvl(evt.cnt,0) sysevt
        from wrm$_snapshot sn,
             (select dbid, instance_number, snap_id, count(*) cnt
                from wrh$_active_session_history  where dbid = :dbid
```

```
                group by dbid, snap_id, instance_number) ash,
              (select dbid, instance_number, snap_id, count(*) cnt
                from wrh$_sqlstat  where dbid = :dbid
                group by dbid, snap_id, instance_number) sql,
              (select dbid, instance_number, snap_id, count(*) cnt
                from dba_hist_sqlbind  where dbid = :dbid
                group by dbid, snap_id, instance_number) sqbnd,
              (select dbid, instance_number, snap_id, count(*) cnt
                from wrh$_filestatxs  where dbid = :dbid
                group by dbid, snap_id, instance_number) files,
              (select dbid, instance_number, snap_id, count(*) cnt
                from wrh$_seg_stat  where dbid = :dbid
                group by dbid, snap_id, instance_number) seg,
              (select dbid, instance_number, snap_id, count(*) cnt
                from wrh$_system_event  where dbid = :dbid
                group by dbid, snap_id, instance_number) evt
        where (sn.snap_id = ash.snap_id(+)
                  and sn.instance_number = ash.instance_number(+))
            and (sn.snap_id = sql.snap_id(+)
                  and sn.instance_number = sql.instance_number(+))
            and (sn.snap_id = sqbnd.snap_id(+)
                  and sn.instance_number = sqbnd.instance_number(+))
            and (sn.snap_id = files.snap_id(+)
                  and sn.instance_number = files.instance_number(+))
            and (sn.snap_id = seg.snap_id(+)
                  and sn.instance_number = seg.instance_number(+))
            and (sn.snap_id = evt.snap_id(+)
                  and sn.instance_number = evt.instance_number(+))
            and sn.dbid = :dbid
        order by snap_id desc, inst desc) s1
  group by dbid, inst;


prompt
prompt ********************************
prompt  (7c) AWR total item counts - names, text, plans
prompt ********************************

select sqtxt.cnt sqltext, sqpln.cnt sqlplan,
       sqlbmeta.cnt sqlbmeta, segob.cnt segobj,
       datafile.cnt datafile, tempfile.cnt tempfile
  from (select count(*) cnt
          from wrh$_sqltext  where dbid = :dbid) sqtxt,
       (select count(*) cnt
          from wrh$_sql_plan  where dbid = :dbid) sqpln,
       (select count(*) cnt
          from wrh$_sql_bind_metadata where dbid = :dbid) sqlbmeta,
       (select count(*) cnt
          from wrh$_datafile  where dbid = :dbid) datafile,
       (select count(*) cnt
          from wrh$_tempfile  where dbid = :dbid) tempfile,
       (select count(*) cnt
          from wrh$_seg_stat_obj  where dbid = :dbid) segob;

set feedback off verify off timing off echo off;
set pagesize 1000;
```

```
prompt
prompt
prompt ############################################################
prompt (II) Advisor Framework Info
prompt ############################################################

Rem
Rem Set this variable to FALSE in tests so that no rows will
Rem be selected in any of the advisor framework sections.
variable  select_valid_rows   number;

declare
  local_dbid number;
begin

  select dbid into local_dbid from V$DATABASE;

  if (:dbid = local_dbid) then
    :select_valid_rows := 1;
  else
    :select_valid_rows := 0;
    dbms_output.put_line('Warning: Using test settings.');
    dbms_output.put_line('~~~~~~~  No rows will be selected ' ||
                         'in the following advisor framework sections.');
  end if;

end;
/

prompt
prompt ********************************
prompt   (1) Advisor Tasks - Last 50
prompt ********************************
col  owner_adv     format  a14  trunc heading  'OWNER/ADVISOR';
col  task_id_name  format  a32        heading  'TASK_ID/NAME';
col  exe_duratn    format  9,999,990;
col  exe_creatn    format  9,999,990;
col  how_created   format  a5;
col  status        format  a12;
select owner || '/' || advisor_name as owner_adv
        , task_id || '/' || task_name as task_id_name
        , created
        , (execution_end - execution_start)*86400 as exe_duratn
        , (execution_end - created) * 86400 as exe_creatn
        , how_created
        , status
from   (select t.*, rank() over (order by execution_end desc) rnk
        from   dba_advisor_tasks t) dat
where  dat.rnk <= 50
  and  :select_valid_rows = 1
order  by created;

prompt
prompt ********************************
prompt   (2) Advisor Task - Oldest 5
prompt ********************************
select owner || '/' || advisor_name as owner_adv
        , task_id || '/' || task_name as task_id_name
```

```
        , created
        , (execution_end - execution_start)*86400 as exe_duratn
        , (execution_end - created) * 86400 as exe_creatn
        , how_created
        , status
from   (select t.*, rank() over (order by execution_end asc) rnk
        from   dba_advisor_tasks t) dat
where  dat.rnk <= 5
  and  :select_valid_rows = 1
order  by created;

prompt
prompt ********************************
prompt   (3) Advisor Tasks With Errors - Last 50
prompt ********************************
set  feedback  10000
col  task_desc      format  a110 newline;
col  error_msg      format  a110 newline;
select owner || '/' || advisor_name as owner_adv
        , task_id || '/' || task_name as task_id_name
        , created
        , (execution_end - execution_start)*86400 as exe_duratn
        , (execution_end - created) * 86400 as exe_creatn
        , how_created
        , status
        , 'Description: ' || description as task_desc
        , 'Error Msg  : ' || error_message as error_msg
from   (select t.*, rank() over (order by execution_end desc) rnk
        from   dba_advisor_tasks t
        where  status <> 'COMPLETED') dat
where  dat.rnk <= 50
  and  :select_valid_rows = 1
order by created;
set  feedback off


prompt
prompt
prompt ############################################################
prompt (III) ASH Usage Info
prompt ############################################################


col num_active_sessions format a20;
col num_samples format 999,999,990;

Rem   Define how far back you want to query ASH in days.
define    ash_long_hist   = 3;
define    ash_short_hist  = 1;

variable  ash_long_hist_snap    number;
variable  ash_short_hist_snap   number;
variable  ash_long_hist         number;
variable  ash_short_hist        number;

declare
  local_dbid number;
```

```
begin

  select dbid into local_dbid from V$DATABASE;

  if (:dbid = local_dbid) then
    :ash_long_hist  := &ash_long_hist;
    :ash_short_hist := &ash_short_hist;
  else
    :ash_long_hist  := 36500;                              /* 100 years */
    :ash_short_hist := 36500;                              /* 100 years */
    dbms_output.put_line('Warning: Using test ASH settings.');
  end if;

  select min(snap_id)
  into   :ash_long_hist_snap
  from   wrm$_snapshot s
  where  dbid = :dbid
    and  (sysdate - :ash_long_hist) <= end_interval_time;

  select min(snap_id)
  into   :ash_short_hist_snap
  from   wrm$_snapshot s
  where  dbid = :dbid
    and  (sysdate - :ash_short_hist) <= end_interval_time;
end;
/

prompt
prompt ********************************
prompt (1a) ASH histogram (past &ash_long_hist days)
prompt ********************************

set heading on
select awrinfo_util.classify_count(grp) as num_active_sessions,
       count(*) as num_samples
from   (select trunc(count(*)/5) * 5 as grp
        from   wrh$_active_session_history
        where  dbid = :dbid
          and  snap_id > :ash_long_hist_snap
          and  sample_time > sysdate - :ash_long_hist
        group  by dbid, instance_number, sample_id)
group by grp
order by grp asc;

prompt
prompt ********************************
prompt (1b) ASH histogram (past &ash_short_hist day)
prompt ********************************

set heading on
select awrinfo_util.classify_count(grp) as num_active_sessions,
       count(*) as num_samples
from   (select trunc(count(*)/5) * 5 as grp
        from   wrh$_active_session_history
        where  dbid = :dbid
          and  snap_id > :ash_short_hist_snap
          and  sample_time > sysdate - :ash_short_hist
        group  by dbid, instance_number, sample_id)
```

```
group by grp
order by grp asc;

col avg_active format 9,990.90 wrap;

prompt
prompt **********************************
prompt (2a) ASH details (past &ash_long_hist days)
prompt **********************************

set heading on

col inst format 999;
col num_rows format 999,999,990;

select instance_number as inst,
       min(sample_time) as min_time, max(sample_time) as max_time,
       (max(sample_id) - min(sample_id) + 1)/10 as num_samples, sum(cnt) as num_rows,
       sum(cnt)/(max(sample_id)-min(sample_id)+1)*10 as avg_active
from   (select sample_id, cast(max(sample_time) as date) as sample_time,
               dbid, instance_number, count(*) as cnt
        from   wrh$_active_session_history
        where  dbid = :dbid
          and  snap_id > :ash_long_hist_snap
          and  sample_time > sysdate - :ash_long_hist
        group  by dbid, instance_number, sample_id)
group  by dbid, instance_number;

prompt
prompt **********************************
prompt (2b) ASH details (past &ash_short_hist day)
prompt **********************************

set heading on
select instance_number as inst,
       min(sample_time) as min_time, max(sample_time) as max_time,
       (max(sample_id) - min(sample_id) + 1)/10 as num_samples, sum(cnt) as num_rows,
       sum(cnt)/(max(sample_id)-min(sample_id)+1)*10 as avg_active
from   (select sample_id, cast(max(sample_time) as date) as sample_time,
               dbid, instance_number, count(*) as cnt
        from   wrh$_active_session_history
        where  dbid = :dbid
          and  snap_id > :ash_short_hist_snap
          and  sample_time > sysdate - :ash_short_hist
        group  by dbid, instance_number, sample_id)
group  by dbid, instance_number;

prompt
prompt **********************************
prompt (2c) ASH sessions (Fg Vs Bg) (past &ash_short_hist day across all instances in RAC)
prompt **********************************

col   sess_type    format a20 newline;

set heading off
select 'Foreground %' sess_type, sum(fg_cnt)*100/sum(tot_cnt) as perc,
       'Background %' sess_type, sum(bg_cnt)*100/sum(tot_cnt) as perc,
       'MMNL %' sess_type, sum(mmnl_cnt)*100/sum(tot_cnt) as perc
```

```
from   (select 1 as tot_cnt,
            (case when session_type = 1 then 1 else 0 end) as fg_cnt,
            (case when session_type = 2 then 1 else 0 end) as bg_cnt,
            (case when program like '%MMNL%' then 1 else 0 end) as mmnl_cnt
        from   wrh$_active_session_history
        where  dbid = :dbid
          and  snap_id > :ash_short_hist_snap
          and  sample_time > sysdate - :ash_short_hist);
set heading on

prompt End of Report

spool off;

prompt Report written to &report_name.

drop package AWRINFO_UTIL;

set termout off;
clear columns sql;
set linesize 78 termout on feedback 6 heading on;

undefine report_name
undefine ash_long_hist;
undefine ash_short_hist;

whenever sqlerror continue;
```