oracle 10g 研究ORACLE_HOME rdbms admin 下的脚本的功能（9）ashrpti.sql

oracle 10g 研究ORACLE_HOME rdbms admin 下的脚本的功能（9）ashrpti.sql

#SQL*Plus script to generate ASH Report

```
#SQL*Plus script to generate ASH Report
Rem
Rem $Header: ashrpti.sql 04-may-2005.20:06:38 veeve Exp $
Rem
Rem ashrpti.sql
Rem
Rem Copyright (c) 2004, 2005, Oracle. All rights reserved.
Rem
Rem    NAME
Rem      ashrpti.sql - SQL*Plus script to generate ASH Report
Rem
Rem    DESCRIPTION
Rem
Rem    NOTES
Rem
Rem    MODIFIED   (MM/DD/YY)
Rem    veeve       05/11/05 - add support for slot_width input
Rem    veeve       05/04/05 - fixed slow query in get btime/etime
Rem    veeve       06/24/04 - flexible input formats for begin_time
Rem    veeve       06/10/04 - veeve_ash_report_r2
Rem    veeve       06/04/04 - Created
Rem

set echo off verify off timing off feedback off trimspool on trimout on
set long 1000000 pagesize 6000 linesize 80

Rem
Rem Customizable parameters
Rem =======================

define default_report_type         = 'html';
define default_report_duration     = 15;
define default_report_name_prefix  = 'ashrpt';
define default_report_name_suffix  = 'MMDD_HH24MI';

Rem
Rem End of Customizable parameters
Rem ==============================

Rem
Rem Get Report Type
Rem ===============

prompt
prompt Specify the Report Type
prompt ~~~~~~~~~~~~~~~~~~~~~~~
prompt Enter 'html' for an HTML report, or 'text' for plain text
prompt Defaults to '&&default_report_type'

column report_type new_value report_type;
set heading off
```

```
select 'Type Specified: ',
       lower( (case when '&&report_type' IS NULL
                    then '&&default_report_type'
                    when '&&report_type' <> 'text'
                    then 'html'
                    else '&&report_type' end) ) report_type
from dual;
set heading on

Rem
Rem Get dbid and instid (if not already specified)
Rem ===================

column instt_num  heading "Inst Num"  format 99999;
column instt_name heading "Instance"  format a12;
column dbb_name   heading "DB Name"   format a12;
column dbbid      heading "DB Id"     format a12 just c;
column host       heading "Host"      format a12;

prompt
prompt
prompt Instances in this Workload Repository schema
prompt ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
set heading on
select distinct
       (case when cd.dbid = wr.dbid and
                  cd.name = wr.db_name and
                  ci.instance_number = wr.instance_number and
                  ci.instance_name   = wr.instance_name
             then '* '
             else '  '
        end) || wr.dbid   dbbid
     , wr.instance_number instt_num
     , wr.db_name         dbb_name
     , wr.instance_name   instt_name
     , wr.host_name       host
  from dba_hist_database_instance wr, v$database cd, v$instance ci;

prompt
prompt Defaults to current database
column dbid new_value dbid;
set heading off
select 'Using database id:',
       (case when '&&dbid' IS NULL
             then d.dbid
             else to_number('&&dbid') end) || ' ' as dbid
from v$database d;
set heading on

prompt
prompt Defaults to current instance
column inst_num new_value inst_num;
set heading off
select 'Using instance number:',
       (case when '&&inst_num' IS NULL
             then ci.instance_number
             else to_number('&&inst_num') end) || ' ' as inst_num
from v$instance ci;
```

```
set heading on

--
--  Set up the binds for dbid and instance_number

variable dbid       number;
variable inst_num   number;
begin
  :dbid      :=  &dbid;
  :inst_num  :=  &inst_num;
end;
/

Rem
Rem Get btime and etime
Rem ===================

-- First, show the oldest and the latest ASH samples available
define   ash_time_format = 'DD-Mon-YY HH24:MI:SS';
variable oldest_sample   varchar2(30);
variable latest_sample   varchar2(30);

whenever sqlerror exit;
declare
  oldest_snap      number;
  latest_snap      number;
  mydbid           number;
  myinst_num       number;

  oldest_mem       date := NULL;
  latest_mem       date := NULL;
  oldest_disk      date := NULL;
  latest_disk      date := NULL;

begin
  select min(snap_id), max(snap_id)
  into   oldest_snap, latest_snap
  from   dba_hist_snapshot
  where  dbid = :dbid
    and  instance_number = :inst_num;

  select dbid into mydbid from v$database;
  select instance_number into myinst_num from v$instance;

  select min(sample_time), max(sample_time)
  into   oldest_disk, latest_disk
  from   dba_hist_active_sess_history
  where  dbid = :dbid
    and  instance_number = :inst_num
    and  snap_id in (oldest_snap, latest_snap);

  if (mydbid = :dbid AND myinst_num = :inst_num) then
    select min(sample_time), max(sample_time)
    into   oldest_mem, latest_mem
    from   v$active_session_history;
  end if;

  if (oldest_disk is null AND oldest_mem is null) then
```

```
    raise_application_error(-20200,
      'No ASH samples exist for Database/Instance '||:dbid||'/'||:inst_num);
  end if;

  -- Put the min(oldest_disk, oldest_mem) in oldest_disk
  -- Take care of NULLs
  case
    when (oldest_disk IS NOT NULL AND oldest_mem IS NOT NULL)
    then oldest_disk := least(oldest_disk, oldest_mem);
    when (oldest_disk IS NULL)
    then oldest_disk := oldest_mem;
    else oldest_disk := oldest_disk;
  end case;

  -- Put the max(latest_disk, latest_mem) in latest_disk
  -- Take care of NULLs
  case
    when (latest_disk IS NOT NULL AND latest_mem IS NOT NULL)
    then latest_disk := greatest(latest_disk, latest_mem);
    when (latest_disk IS NULL)
    then latest_disk := latest_mem;
    else latest_disk := latest_disk;
  end case;

  :oldest_sample := to_char(oldest_disk, '&&ash_time_format');
  :latest_sample := to_char(latest_disk, '&&ash_time_format');

end;
/
whenever sqlerror continue;

column ash_sample  format a20;
column min_past    format a6;
prompt
prompt
prompt ASH Samples in this Workload Repository schema
prompt ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
set heading off
select 'Oldest ASH sample available: ', :oldest_sample as ash_sample,
       '[',
       (to_char((sysdate - to_date(:oldest_sample, '&&ash_time_format'))*1440,
                '99999')) as min_past,
       'mins in the past]',
       'Latest ASH sample available: ', :latest_sample as ash_sample,
       '[',
       (to_char((sysdate - to_date(:latest_sample, '&&ash_time_format'))*1440,
                '99999')) as min_past,
       'mins in the past]'
from   dual;
set heading on

prompt
prompt Specify the timeframe to generate the ASH report
prompt ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~



Rem
Rem Get btime
```

```
Rem =========
prompt Enter begin time for report:
prompt
prompt --    Valid input formats:
prompt --      To specify absolute begin time:
prompt --        [MM/DD[/YY]] HH24:MI[:SS]
prompt --        Examples: 02/23/03 14:30:15
prompt --                  02/23 14:30:15
prompt --                  14:30:15
prompt --                  14:30
prompt --      To specify relative begin time: (start with '-' sign)
prompt --        -[HH24:]MI
prompt --        Examples: -1:15  (SYSDATE - 1 Hr 15 Mins)
prompt --                  -25    (SYSDATE - 25 Mins)
prompt
prompt Defaults to -&&default_report_duration mins
prompt Report begin time specified: &&begin_time
prompt

--
--  Set up the binds for btime
whenever sqlerror exit;
variable btime varchar2(30);
declare
  lbtime_in         varchar2(100);
  begin_time        date;

  FUNCTION get_time_from_begin_time( btime_in IN VARCHAR2 )
    RETURN DATE
  IS
    first_char    VARCHAR2(2);
    in_str        VARCHAR2(100);

    past_hrs      NUMBER;
    past_mins     NUMBER;
    pos           NUMBER;

    num_slashes   NUMBER := 0;
    num_colons    NUMBER := 0;
    date_part     VARCHAR2(100);
    time_part     VARCHAR2(100);

    my_fmt        VARCHAR2(100) := 'MM/DD/YY HH24:MI:SS';
  BEGIN
    in_str := TRIM(btime_in);
    first_char := SUBSTR(in_str, 1, 1);

    /* Handle relative input format starting with a -ve sign, first */
    IF (first_char = '-') THEN
      in_str := SUBSTR(in_str, 2);
      pos := INSTR(in_str,':');
      IF (pos = 0) THEN
        past_hrs := 0;
        past_mins := TO_NUMBER(in_str);
      ELSE
        past_hrs := TO_NUMBER(SUBSTR(in_str,1,pos-1));
        past_mins := TO_NUMBER(SUBSTR(in_str,pos+1));
      END IF;
      IF (past_mins = 0 AND past_hrs = 0) THEN
        /* Invalid input */
        raise_application_error( -20500,
                                 'Invalid input! Cannot recognize ' ||
                                 'input format for begin_time ' || '"' ||
                                 TRIM(btime_in) || '"' );
        RETURN NULL;
      END IF;

      RETURN (sysdate - past_hrs/24 - past_mins/1440);
    END IF;

    /* Handle absolute input format now.
       Fill out all the missing optional parts of the input string
       to make it look like 'my_fmt' first. Then just do "return to_date()".
     */
    FOR pos in 1..LENGTH(in_str) LOOP
      IF (SUBSTR(in_str,pos,1) = '/') THEN
        num_slashes := num_slashes + 1;
      END IF;
      IF (SUBSTR(in_str,pos,1) = ':') THEN
        num_colons := num_colons + 1;
      END IF;
    END LOOP;

    IF (num_slashes > 0) THEN
      pos := INSTR(in_str,' ');
      date_part := TRIM(SUBSTR(in_str,1,pos-1));
      time_part := TRIM(SUBSTR(in_str,pos+1));

      IF (num_slashes = 1) THEN
        date_part := date_part || '/' || TO_CHAR(sysdate,'YY');
      END IF;
    ELSE
      date_part := TO_CHAR(sysdate,'MM/DD/YY');
      time_part := in_str;
    END IF;

    IF (num_colons > 0) THEN
      IF (num_colons = 1) THEN
        time_part := time_part || ':00';
      END IF;
      in_str := date_part || ' ' || time_part;
      begin
        RETURN TO_DATE(in_str, my_fmt);
      exception
        when others then
        /* Invalid input */
        raise_application_error( -20500,
                                 'Invalid input! Cannot recognize ' ||
                                 'input format for begin_time ' || '"' ||
                                 TRIM(btime_in) || '"' );
      end;
    END IF;

    /* Invalid input */
    raise_application_error( -20500,
                             'Invalid input! Cannot recognize ' ||
                             'input format for begin_time ' || '"' ||
                             TRIM(btime_in) || '"' );
    RETURN NULL;

  END get_time_from_begin_time;

begin
  lbtime_in  := nvl('&&begin_time', '-' || &&default_report_duration);
  begin_time := get_time_from_begin_time(lbtime_in);
  :btime := to_char( begin_time, '&&ash_time_format' );
end;
/
whenever sqlerror continue;

Rem
Rem Get etime
Rem =========
prompt Enter duration in minutes starting from begin time:
prompt Defaults to SYSDATE - begin_time
prompt Press Enter to analyze till current time
prompt Report duration specified:   &&duration

--
--  Set up the binds for etime
variable etime varchar2(30);
declare
  duration          number;
  since_begin_time  number;
  begin_time        date;
  end_time          date;
begin
  -- First calculate minutes since begin_time
  begin_time := to_date( :btime, '&&ash_time_format' );
  since_begin_time := (sysdate - begin_time)*1440;

  -- Default to since_begin_time
  duration   := nvl('&&duration', since_begin_time);

  -- Put upper bound on user input to not go into the future
  if (duration > since_begin_time) then
    duration := since_begin_time;
  end if;

  -- Calculate end_time and :etime
  end_time := begin_time + duration/1440;
  :etime := to_char( end_time, '&&ash_time_format' );
end;
/

column nl80 format a80 newline;
set heading off
select 'Using ' || :btime || ' as report begin time' as nl80,
       'Using ' || :etime || ' as report end time' as nl80
from   dual;
set heading on

Rem
```

```
Rem Get Slot Width for the 'Activity Over Time' section
Rem ===================================================

prompt
prompt Specify Slot Width (using ashrpti.sql) for 'Activity Over Time' section
prompt ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
prompt
prompt -- Explanation:
prompt --   In the 'Activity Over Time' section of the ASH report,
prompt --   the analysis period is divided into smaller slots
prompt --   and top wait events are reported in each of those slots.
prompt
prompt -- Default:
prompt --   The analysis period will be automatically split upto 10 slots
prompt --   complying to a minimum slot width of
prompt --     1 minute,  if the source is V$ACTIVE_SESSION_HISTORY or
prompt --     5 minutes, if the source is DBA_HIST_ACTIVE_SESS_HISTORY.
prompt

prompt
prompt Specify Slot Width in seconds to use in the 'Activity Over Time' section:
prompt Defaults to a value as explained above:
prompt Slot Width specified: &&slot_width
prompt



Rem
Rem Get Special Report Targets
Rem ==========================

prompt
prompt Specify Report Targets (using ashrpti.sql) to generate the ASH report
prompt ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
prompt
prompt -- Explanation:
prompt --   ASH Report can accept "Report Targets",
prompt --   like a particular SQL statement, or a particular SESSION,
prompt --   to generate the report on. If one or more report targets are
prompt --   specified, then the data used to generate the report will only be
prompt --   the ASH samples that pertain to ALL the specified report targets.
prompt
prompt -- Default:
prompt --   If none of the report targets are specified,
prompt --   then the target defaults to all activity in the database instance.
prompt

prompt
prompt Specify SESSION_ID (eg: from V$SESSION.SID) report target:
prompt Defaults to NULL:
prompt SESSION report target specified: &&target_session_id
prompt

prompt
prompt Specify SQL_ID (eg: from V$SQL.SQL_ID) report target:
prompt Defaults to NULL: (% and _ wildcards allowed)
prompt SQL report target specified: &&target_sql_id
prompt
```

```
prompt
prompt Specify WATI_CLASS name (eg: from V$EVENT_NAME.WAIT_CLASS) report target:
prompt [Enter 'CPU' to investigate CPU usage]
prompt Defaults to NULL: (% and _ wildcards allowed)
prompt WAIT_CLASS report target specified: &&target_wait_class
prompt

prompt
prompt Specify SERVICE_HASH (eg: from V$ACTIVE_SERVICES.NAME_HASH) report target:
prompt Defaults to NULL:
prompt SERVICE report target specified: &&target_service_hash
prompt

prompt
prompt Specify MODULE name (eg: from V$SESSION.MODULE) report target:
prompt Defaults to NULL: (% and _ wildcards allowed)
prompt MODULE report target specified: &&target_module_name
prompt

prompt
prompt Specify ACTION name (eg: from V$SESSION.ACTION) report target:
prompt Defaults to NULL: (% and _ wildcards allowed)
prompt ACTION report target specified: &&target_action_name
prompt

prompt
prompt Specify CLIENT_ID (eg: from V$SESSION.CLIENT_IDENTIFIER) report target:
prompt Defaults to NULL: (% and _ wildcards allowed)
prompt CLIENT_ID report target specified: &&target_client_id
prompt



Rem
Rem Get Report Name
Rem ===============

-- set the extension based on the report_type
set termout off;
column ext new_value ext;
select '.html' ext from dual where lower('&&report_type') <> 'text';
select '.txt' ext from dual where lower('&&report_type') = 'text';
set termout on;

set termout off;
column dflt_name new_value dflt_name noprint;
select '&&default_report_name_prefix' || '_'
       || :inst_num || '_'
       || to_char( to_date(:etime, '&&ash_time_format'),
                   '&&default_report_name_suffix' )
       || '&&ext' dflt_name
from dual;
set termout on;

prompt Specify the Report Name
prompt ~~~~~~~~~~~~~~~~~~~~~~~
prompt The default report file name is &&dflt_name..  To use this name,
prompt press <return> to continue, otherwise enter an alternative.
```

```
column report_name_msg new_value report_name_msg;
column report_name new_value report_name;
set heading off;
select 'Using the report name'
       as report_name_msg,
       nvl('&&report_name','&dflt_name') as report_name
  from sys.dual;
set heading on;

column nl80 format a80 newline;
set heading off;
select 'Summary of All User Input',
       '-------------------------' as nl80,
       'Format         : ' || upper('&&report_type') as nl80,
       'DB Id          : ' || :dbid as nl80,
       'Inst num       : ' || :inst_num as nl80,
       'Begin time     : ' || :btime as nl80,
       'End time       : ' || :etime as nl80,
       'Slot width     : ' || decode('&&slot_width',
                             NULL, 'Default',
                             '&&slot_width seconds') as nl80,
       'Report targets : ' || (decode(q'^&&target_session_id^',
                             NULL, 0, 1)
                      + decode(q'^&&target_sql_id^',
                             NULL, 0, 1)
                      + decode(q'^&&target_wait_class^',
                             NULL, 0, 1)
                      + decode(q'^&&target_service_hash^',
                             NULL, 0, 1)
                      + decode(q'^&&target_module_name^',
                             NULL, 0, 1)
                      + decode(q'^&&target_action_name^',
                             NULL, 0, 1)
                      + decode(q'^&&target_client_id^',
                             NULL, 0, 1)) as nl80,
       'Report name    : ' || '&&report_name'
from   dual;
set heading on;



Rem
Rem Set function name and linesize
Rem =====================================

set termout off;
column fn_name new_value fn_name noprint;
select 'ash_report_text' fn_name
  from dual
 where lower('&report_type') = 'text';
select 'ash_report_html' fn_name
  from dual
 where lower('&report_type') <> 'text';

column lnsz new_value lnsz noprint;
select '80' lnsz from dual where lower('&report_type') = 'text';
select '500' lnsz from dual where lower('&report_type') <> 'text';
set termout on;
set linesize &lnsz;

Rem
Rem Spool out the report
Rem ====================

set termout on heading off;
spool &report_name;
select output from table(dbms_workload_repository.&fn_name( :dbid,
                             :inst_num,
                             to_date(:btime, '&&ash_time_format'),
                             to_date(:etime, '&&ash_time_format'),
                             0,
                             to_number(nvl('&&slot_width', 0)),
                             to_number(q'^&&target_session_id^'),
                             q'^&&target_sql_id^',
                             q'^&&target_wait_class^',
                             to_number(q'^&&target_service_hash^'),
                             q'^&&target_module_name^',
                             q'^&&target_action_name^',
                             q'^&&target_client_id^'
                         ));

spool off;

prompt Report written to &report_name.

-- cleanup
clear columns sql;

ttitle off;
btitle off;
repfooter off;
set linesize 78 termout on feedback 6 heading on;

-- Undefine all 'define's
undefine default_report_type;
undefine default_report_duration;
undefine default_report_name_prefix;
undefine default_report_name_suffix;
undefine ash_time_format

-- Undefine all 'new_value's
undefine report_type
undefine ext
undefine dflt_name
undefine report_name_msg
undefine report_name
undefine fn_name
undefine lnsz

-- Undefine all 'input variables'
undefine dbid
undefine inst_num
undefine begin_time
undefine duration
undefine slot_width
undefine target_session_id
undefine target_sql_id
undefine target_wait_class
undefine target_service_hash
undefine target_module_name
undefine target_action_name
undefine target_client_id

whenever sqlerror continue;
--
-- End of script file
```