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

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

```
#A chunk of common code used for SWRF reports and ADDM.
#This script gets the dbid,eid,filename,etc from the user
#for both components to use.


Rem
Rem $Header: awrddinp.sql 22-may-2005.14:37:08 mlfeng Exp $
Rem
Rem awrddinp.sql
Rem
Rem Copyright (c) 2004, 2005, Oracle. All rights reserved.
Rem
Rem    NAME
Rem      awrinput.sql - <one-line expansion of the name>
Rem
Rem    DESCRIPTION
Rem      A chunk of common code used for SWRF reports and ADDM.
Rem      This script gets the dbid,eid,filename,etc from the user
Rem      for both components to use.
Rem
Rem    NOTES
Rem      This script could leave a few other SQL*Plus substitution and/or
Rem      bind variables defined at the end.
Rem
Rem    MODIFIED   (MM/DD/YY)
Rem    mlfeng      05/22/05 - remove leading blank from date conversion
Rem    mramache    06/17/04 - mramache_diff_diff
Rem    mramache    05/25/04 - Fixing a problem with number of days for second
Rem                           snapshot.
Rem    ilistvin    05/25/04 - created
Rem



-- Script Parameters:
--   First Param (&1) : file prefix e.g. 'swrfrpt_'
--   Second Param (&2) : file extension e.g. '.html', '.lst'
--     **** IMPORTANT - the second parameter must be non-null, or else SQL plus
--          adds an awkward prompt when we try to use it

-- After executing, this module leaves the substitution variable
-- &report_name defined.  Issue the command spool &report_name to
-- spool your report to a file, and then undefine report_name when you're
-- done with it.

-- The following list of SQL*Plus bind variables will be defined and assigned a value
-- by this SQL*Plus script:
--    variable dbid      number     - Database id for first pair of snapshots
--    variable inst_num  number     - Instance number for first pair of snapshots
--    variable bid       number     - Begin snapshot id for first pair of snapshots
--    variable eid       number     - End snapshot id for first pair of snapshots
--    variable dbid2      number     - Database id for second pair of snapshots
--    variable inst_num2  number     - Instance number for second pair of snapshots
--    variable bid2       number     - Begin snapshot id for second pair of snapshots
--    variable eid2       number     - End snapshot id for second pair of snapshots
```

```
clear break compute;
repfooter off;
ttitle off;
btitle off;

set heading on;
set timing off veri off space 1 flush on pause off termout on numwidth 10;
set echo off feedback off pagesize 60 linesize 80 newpage 1 recsep off;
set trimspool on trimout on define "&" concat "." serveroutput on;
set underline on;

--
-- Request the DB Id and Instance Number, if they are not specified

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 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
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 Database Id and Instance Number for the First Pair of Snapshots
prompt ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
prompt Using &&dbid for Database Id for the first pair of snapshots
prompt Using &&inst_num for Instance Number for the first pair of snapshots



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

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

```
/

--
--  Error reporting

whenever sqlerror exit;
variable max_snap_time char(10);
declare

  cursor cidnum is
     select 'X'
       from dba_hist_database_instance
      where instance_number = :inst_num
        and dbid            = :dbid;

  cursor csnapid is
     select to_char(max(end_interval_time),'dd/mm/yyyy')
       from dba_hist_snapshot
      where instance_number = :inst_num
        and dbid            = :dbid;

  vx     char(1);

begin

  -- Check Database Id/Instance Number is a valid pair
  open cidnum;
  fetch cidnum into vx;
  if cidnum%notfound then
    raise_application_error(-20200,
      'Database/Instance ' || :dbid || '/' || :inst_num ||
      ' does not exist in DBA_HIST_DATABASE_INSTANCE');
  end if;
  close cidnum;

  -- Check Snapshots exist for Database Id/Instance Number
  open csnapid;
  fetch csnapid into :max_snap_time;
  if csnapid%notfound then
    raise_application_error(-20200,
      'No snapshots exist for Database/Instance '||:dbid||'/'||:inst_num);
  end if;
  close csnapid;

end;
/
whenever sqlerror continue;



--
--  Ask how many days of snapshots to display

set termout on;
column instart_fmt noprint;
column inst_name   format a12  heading 'Instance';
column db_name     format a12  heading 'DB Name';
column snap_id     format 99999990 heading 'Snap Id';
column snapdat     format a18  heading 'Snap Started' just c;
column lvl         format 99   heading 'Snap|Level';

prompt
prompt
prompt Specify the number of days of snapshots to choose from
prompt ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
prompt Entering the number of days (n) will result in the most recent
prompt (n) days of snapshots being listed.  Pressing <return> without
prompt specifying a number lists all completed snapshots.
prompt
prompt

set heading off;
column num_days new_value num_days noprint;
select    'Listing '
       || decode( nvl('&&num_days', 3.14)
                , 0    , 'no snapshots'
                , 3.14 , 'all Completed Snapshots'
                , 1    , 'the last day''s Completed Snapshots'
                , 'the last &num_days days of Completed Snapshots')
     , nvl('&&num_days', 3.14)  num_days
  from sys.dual;
set heading on;



--
-- List available snapshots

break on inst_name on db_name on host on instart_fmt skip 1;

ttitle off;

select to_char(s.startup_time,'dd Mon "at" HH24:mi:ss') instart_fmt
     , di.instance_name                                  inst_name
     , di.db_name                                        db_name
     , s.snap_id                                         snap_id
     , to_char(s.end_interval_time,'dd Mon YYYY HH24:mi') snapdat
     , s.snap_level                                      lvl
  from dba_hist_snapshot s
     , dba_hist_database_instance di
 where s.dbid              = :dbid
   and di.dbid             = :dbid
   and s.instance_number   = :inst_num
   and di.instance_number  = :inst_num
   and di.dbid             = s.dbid
   and di.instance_number  = s.instance_number
   and di.startup_time     = s.startup_time
   and s.end_interval_time >= decode( &num_days
                                   , 0   , to_date('31-JAN-9999','DD-MON-YYYY')
                                   , 3.14, s.end_interval_time
                                   , to_date(:max_snap_time,'dd/mm/yyyy') - (&num_days-1))
 order by db_name, instance_name, snap_id;

clear break;
ttitle off;


--
```

```
--  Ask for the snapshots Id's which are to be compared

prompt
prompt
prompt Specify the First Pair of Begin and End Snapshot Ids
prompt ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
prompt First Begin Snapshot Id specified: &&begin_snap
prompt
prompt First End   Snapshot Id specified: &&end_snap
prompt



--
--  Set up the snapshot-related binds

variable bid        number;
variable eid        number;
begin
  :bid       :=  &begin_snap;
  :eid       :=  &end_snap;
end;
/

prompt



--
--  Error reporting

whenever sqlerror exit;
declare

  cursor cspid(vspid dba_hist_snapshot.snap_id%type) is
     select end_interval_time
          , startup_time
       from dba_hist_snapshot
      where snap_id         = vspid
        and instance_number = :inst_num
        and dbid            = :dbid;

  bsnapt  dba_hist_snapshot.end_interval_time%type;
  bstart  dba_hist_snapshot.startup_time%type;
  esnapt  dba_hist_snapshot.end_interval_time%type;
  estart  dba_hist_snapshot.startup_time%type;

begin

  -- Check Begin Snapshot id is valid, get corresponding instance startup time
  open cspid(:bid);
  fetch cspid into bsnapt, bstart;
  if cspid%notfound then
    raise_application_error(-20200,
      'Begin Snapshot Id '||:bid||' does not exist for this database/instance');
  end if;
  close cspid;

  -- Check End Snapshot id is valid and get corresponding instance startup time
  open cspid(:eid);
```

```
  fetch cspid into esnapt, estart;
  if cspid%notfound then
    raise_application_error(-20200,
      'End Snapshot Id '||:eid||' does not exist for this database/instance');
  end if;
  if esnapt <= bsnapt then
    raise_application_error(-20200,
      'End Snapshot Id '||:eid||' must be greater than Begin Snapshot Id '||:bid);
  end if;
  close cspid;

  -- Check startup time is same for begin and end snapshot ids
  if ( bstart != estart) then
    raise_application_error(-20200,
      'The instance was shutdown between snapshots '||:bid||' and '||:eid);
  end if;

end;
/
whenever sqlerror continue;



--
--  Get the database info to display in the report

set termout off;
column para       new_value para;
column versn      new_value versn;
column host_name  new_value host_name;
column db_name    new_value db_name;
column inst_name  new_value inst_name;
column btime      new_value btime;
column etime      new_value etime;

select parallel       para
     , version        versn
     , host_name      host_name
     , db_name        db_name
     , instance_name  inst_name
     , to_char(end_interval_time,'YYYYMMDD HH24:MI:SS')  btime
  from dba_hist_database_instance di
     , dba_hist_snapshot          s
 where s.snap_id          = :bid
   and s.dbid             = :dbid
   and s.instance_number  = :inst_num
   and di.dbid            = s.dbid
   and di.instance_number = s.instance_number
   and di.startup_time    = s.startup_time;

select to_char(end_interval_time,'YYYYMMDD HH24:MI:SS')  etime
  from dba_hist_snapshot     s
 where s.snap_id          = :eid
   and s.dbid             = :dbid
   and s.instance_number  = :inst_num;

variable para       varchar2(9);
variable versn      varchar2(10);
variable host_name  varchar2(64);
variable db_name    varchar2(20);
variable inst_name  varchar2(20);
variable btime      varchar2(25);
variable etime      varchar2(25);
begin
  :para      := '&para';
  :versn     := '&versn';
  :host_name := '&host_name';
  :db_name   := '&db_name';
  :inst_name := '&inst_name';
  :btime     := '&btime';
  :etime     := '&etime';
end;
/



clear break compute;
repfooter off;
ttitle off;
btitle off;

set heading on;
set timing off veri off space 1 flush on pause off termout on numwidth 10;
set echo off feedback off pagesize 60 linesize 80 newpage 1 recsep off;
set trimspool on trimout on define "&" concat "." serveroutput on;
set underline on;

--
-- Request the DB Id and Instance Number, if they are not specified

prompt
prompt
prompt Instances in this Workload Repository schema
prompt ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
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;
--
-- Set up dbid and instance number for the first pair of snapshots
-- as defaults for the second pair of snapshots
--
column dbid1 new_value dbid1 noprint;
column instnum1 new_value instnum1 noprint;
select :dbid as dbid1, :inst_num as instnum1 from dual;

prompt
prompt Database Id and Instance Number for the Second Pair of Snapshots
prompt ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

```
prompt
prompt Using &&dbid2 for Database Id for the second pair of snapshots
prompt Using &&inst_num2 for Instance Number for the second pair of snapshots

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

variable dbid2       number;
variable inst_num2   number;
begin
  :dbid2      :=  &dbid2;
  :inst_num2  :=  &inst_num2;
end;
/




--
--  Ask how many days of snapshots to display

set termout on;
column instart_fmt noprint;
column inst_name   format a12  heading 'Instance';
column db_name     format a12  heading 'DB Name';
column snap_id     format 99999990 heading 'Snap Id';
column snapdat     format a18  heading 'Snap Started' just c;
column lvl         format 99   heading 'Snap|Level';

prompt
prompt
prompt Specify the number of days of snapshots to choose from
prompt ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
prompt Entering the number of days (n) will result in the most recent
prompt (n) days of snapshots being listed.  Pressing <return> without
prompt specifying a number lists all completed snapshots.
prompt
prompt

set heading off;
column num_days2 new_value num_days2 noprint;
select    'Listing '
       || decode( nvl('&&num_days2', 3.14)
                , 0    , 'no snapshots'
                , 3.14 , 'all Completed Snapshots'
                , 1    , 'the last day''s Completed Snapshots'
                , 'the last &num_days2 days of Completed Snapshots')
     , nvl('&&num_days2', 3.14)  num_days2
  from sys.dual;
set heading on;



--
-- List available snapshots

break on inst_name on db_name on host on instart_fmt skip 1;

ttitle off;
```

```
select to_char(s.startup_time,'dd Mon "at" HH24:mi:ss') instart_fmt
     , di.instance_name                                  inst_name
     , di.db_name                                        db_name
     , s.snap_id                                         snap_id
     , to_char(s.end_interval_time,'dd Mon YYYY HH24:mi') snapdat
     , s.snap_level                                      lvl
  from dba_hist_snapshot s
     , dba_hist_database_instance di
 where s.dbid              = :dbid2
   and di.dbid             = :dbid2
   and s.instance_number   = :inst_num2
   and di.instance_number  = :inst_num2
   and di.dbid             = s.dbid
   and di.instance_number  = s.instance_number
   and di.startup_time     = s.startup_time
   and s.end_interval_time >= decode( &num_days2
                                   , 0   , to_date('31-JAN-9999','DD-MON-YYYY')
                                   , 3.14, s.end_interval_time
                                   , to_date(:max_snap_time,'dd/mm/yyyy') - (&num_days2-1))
 order by db_name, instance_name, snap_id;

clear break;
ttitle off;



--
--  Error reporting

whenever sqlerror exit;
-- variable max_snap_time char(10);
declare

  cursor cidnum2 is
     select 'X'
       from dba_hist_database_instance
      where instance_number = :inst_num2
        and dbid            = :dbid2;

  cursor csnapid2 is
     select to_char(max(end_interval_time),'dd/mm/yyyy')
       from dba_hist_snapshot
      where instance_number = :inst_num2
        and dbid            = :dbid2;

  vx     char(1);

begin

  -- Check Database Id/Instance Number is a valid pair
  open cidnum2;
  fetch cidnum2 into vx;
  if cidnum2%notfound then
    raise_application_error(-20200,
      'Database/Instance ' || :dbid2 || '/' || :inst_num2 ||
      ' does not exist in DBA_HIST_DATABASE_INSTANCE');
  end if;
  close cidnum2;
  -- Check Snapshots exist for Database Id/Instance Number
  open csnapid2;
  fetch csnapid2 into :max_snap_time;
  if csnapid2%notfound then
    raise_application_error(-20200,
      'No snapshots exist for Database/Instance '||:dbid2||'/'||:inst_num2);
  end if;
  close csnapid2;

end;
/
whenever sqlerror continue;



--
--  Ask how many days of snapshots to display

set termout on;

--
--  Ask for the snapshots Id's which are to be compared

prompt
prompt
prompt Specify the Second Pair of Begin and End Snapshot Ids
prompt ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
prompt Second Begin Snapshot Id specified: &&begin_snap2
prompt
prompt Second End   Snapshot Id specified: &&end_snap2
prompt



--
--  Set up the snapshot-related binds

variable bid2        number;
variable eid2       number;
begin
  :bid2       :=  &begin_snap2;
  :eid2       :=  &end_snap2;
end;
/

prompt



--
--  Error reporting

whenever sqlerror exit;
declare

  cursor cspid2(vspid dba_hist_snapshot.snap_id%type) is
     select end_interval_time
          , startup_time
       from dba_hist_snapshot
      where snap_id         = vspid
        and instance_number = :inst_num2
```

```
        and dbid             = :dbid2;

  bsnapt  dba_hist_snapshot.end_interval_time%type;
  bstart  dba_hist_snapshot.startup_time%type;
  esnapt  dba_hist_snapshot.end_interval_time%type;
  estart  dba_hist_snapshot.startup_time%type;

begin

  -- Check Begin Snapshot id is valid, get corresponding instance startup time
  open cspid2(:bid2);
  fetch cspid2 into bsnapt, bstart;
  if cspid2%notfound then
    raise_application_error(-20200,
      'Begin Snapshot Id '||:bid2||' does not exist for this database/instance');
  end if;
  close cspid2;

  -- Check End Snapshot id is valid and get corresponding instance startup time
  open cspid2(:eid2);
  fetch cspid2 into esnapt, estart;
  if cspid2%notfound then
    raise_application_error(-20200,
      'End Snapshot Id '||:eid2||' does not exist for this database/instance');
  end if;
  if esnapt <= bsnapt then
    raise_application_error(-20200,
      'End Snapshot Id '||:eid2||' must be greater than Begin Snapshot Id '||:bid2);
  end if;
  close cspid2;

  -- Check startup time is same for begin and end snapshot ids
  if ( bstart != estart) then
    raise_application_error(-20200,
      'The instance was shutdown between snapshots '||:bid2||' and '||:eid2);
  end if;

end;
/
whenever sqlerror continue;



--
--  Get the database info to display in the report

set termout off;
column para       new_value para2;
column versn      new_value versn2;
column host_name  new_value host_name2;
column db_name    new_value db_name2;
column inst_name  new_value inst_name2;
column btime      new_value btime2;
column etime      new_value etime2;

select parallel       para
     , version        versn
     , host_name      host_name
     , db_name        db_name
     , instance_name  inst_name
     , to_char(end_interval_time, 'YYYYMMDD HH24:MI:SS')  btime
  from dba_hist_database_instance di
     , dba_hist_snapshot          s
 where s.snap_id          = :bid2
   and s.dbid             = :dbid2
   and s.instance_number  = :inst_num2
   and di.dbid            = s.dbid
   and di.instance_number = s.instance_number
   and di.startup_time    = s.startup_time;

select to_char(end_interval_time, 'YYYYMMDD HH24:MI:SS')  etime
  from dba_hist_snapshot     s
 where s.snap_id          = :eid2
   and s.dbid             = :dbid2
   and s.instance_number  = :inst_num2;

variable para2      varchar2(9);
variable versn2     varchar2(10);
variable host_name2  varchar2(64);
variable db_name2    varchar2(20);
variable inst_name2  varchar2(20);
variable btime2      varchar2(25);
variable etime2      varchar2(25);
begin
  :para2      := '&para2';
  :versn2     := '&versn2';
  :host_name2 := '&host_name2';
  :db_name2   := '&db_name2';
  :inst_name2 := '&inst_name2';
  :btime2     := '&btime2';
  :etime2     := '&etime2';
end;
/
set termout on;



--
-- Use report name if specified, otherwise prompt user for output file
-- name (specify default), then begin spooling
--
set termout off;
column dflt_name new_value dflt_name noprint;
select '&&1'||:inst_num||'_'||:bid||'_'||:inst_num2||'_'||:bid2||'&&2' dflt_name from dual;
set termout on;

prompt
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.
prompt

set heading 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;
```

```
set heading off;
set pagesize 50000;
set echo off;
set feedback off;

undefine dbid
undefine inst_num
undefine num_days
undefine begin_snap
undefine end_snap

undefine para;
undefine versn;
undefine host_name;
undefine db_name;
undefine inst_name;
undefine btime
undefine etime

undefine dbid2
undefine inst_num2
undefine num_days2
undefine begin_snap2
undefine end_snap2

undefine para2;
undefine versn2;
undefine host_name2;
undefine db_name2;
undefine inst_name2;
undefine btime2
undefine etime2

undefine dflt_name2

undefine 1
undefine 2
```