oracle 10g 研究ORACLE_HOME rdbms admin 下的脚本的功能（3）a0902000.sql

oracle 10g 研究ORACLE_HOME rdbms admin 下的脚本的功能（3）

```
#additional ANONYMOUS BLOCK dictionary upgrade.
#Upgrade Oracle RDBMS from 9.2.0 to the new release

Rem
Rem $Header: a0902000.sql 07-jun-2005.17:23:55 elu Exp $
Rem
Rem a0902000.sql
Rem
Rem Copyright (c) 1999, 2005, Oracle. All rights reserved.
Rem
Rem    NAME
Rem      a0902000.sql - additional ANONYMOUS BLOCK dictionary upgrade.
Rem                     Upgrade Oracle RDBMS from 9.2.0 to the new release
Rem
Rem
Rem    DESCRIPTION
Rem      Additional upgrade script to be run during the upgrade of an
Rem      9.2.0 database to the new release.
Rem
Rem      This script is called from u0902000.sql and a0900010.sql
Rem
Rem      Put any anonymous block related changes here.
Rem      Any dictionary create, alter, updates and deletes
Rem      that must be performed before catalog.sql and catproc.sql go
Rem      in c0902000.sql
Rem
Rem      The upgrade is performed in the following stages:
Rem        STAGE 1: steps to upgrade from 9.2.0 to 10.1
Rem        STAGE 2: upgrade from 9.2.0 to the new release
Rem
Rem    NOTES
Rem      * This script must be run using SQL*PLUS.
Rem      * You must be connected AS SYSDBA to run this script.
Rem
Rem    MODIFIED   (MM/DD/YY)
Rem    elu         06/07/05 - modify nocompress for aq tables
Rem    bpwang      09/15/04 - Bug 3880023: Remove orphaned entries in
Rem                           apply$_dest_obj_ops
Rem    rburns      01/07/04 - add calls to 10.1 scripts
Rem    elu         11/14/03 - nocompress AQ subscriber table
Rem    nbhatt      11/21/03 - add evaluation context to aq rules
Rem    gtarora     10/08/03 - move endian flag from a-script to c-script
Rem    jawilson    09/16/03 - AQ timestamp changes
Rem    htran       09/05/03 - fix subscriber address parsing
Rem    rburns      08/28/03 - cleanup
Rem    weiwang     07/19/03 - fix LRG 1543592
Rem    nbhatt      06/20/03 - add tables as dependents of queue table export
Rem    elu         05/16/03 - negative queue rule set
Rem    jawilson    05/09/03 - rename _NR IOT
Rem    liwong      06/18/03 - dml_handlers for virtual objects
Rem    weiwang     05/12/03 - remove rules upgrade code
Rem    rpang       05/15/03 - add new PL/SQL compiler parameters
Rem    htran       05/12/03 - change aq expact$ entries to 10.1 format
```

```
Rem    htran       05/02/03 - canonicalize address in subscriber table
Rem    nshodhan    04/24/03 - set streams_unsupported bit
Rem    weiwang     04/22/03 - patch queue tables during upgrade
Rem    tbgraves    03/27/03 - premerge SVRMGMT
Rem    tbgraves    03/12/03 - revert_updown_scripts
Rem    tbgraves    01/31/03 - pre-merge for sync-branch SVRMGMT
Rem                           based on MAIN_030117.0001
Rem    jawilson    01/14/03 - add buffered queue views
Rem    weiwang     01/23/03 - add IOTs for rule sets
Rem    rburns      01/18/03 - remove registry query
Rem    dvoss       01/14/03 - add logminer checkpoint upgrade
Rem    alakshmi    12/19/02 - turn on hotmining for RAC
Rem    htran       12/12/02 - expand ruleset_name in AQ subscriber tables and
Rem                           canonicalize
Rem    weiwang     12/19/02 - add upgrade script for queues
Rem    liwong      11/20/02 - Set source_dbname
Rem    htran       11/15/02 - canonicalize some Streams values
Rem    sslim       12/02/02 - lrg 1112873: logical standby support
Rem    elu         10/23/02 - queue negative rule sets
Rem    mxiao       08/23/02 - set values in mlog$ and cdc_change_tables$
Rem    gtarora     08/22/02 - lob flag
Rem    dvoss       09/05/02 - delete type 10 object from logmnrc_gtlo
Rem    rburns      04/10/02 - list components
Rem    rburns      03/27/02 - add 92 populate
Rem    rburns      03/17/02 - rburns_10i_updown_scripts
Rem    rburns      02/12/02 - created
Rem

Rem ========================================================================
Rem BEGIN STAGE 1: upgrade from 9.2.0 to 10.1
Rem ========================================================================

Rem Insert PL/SQL blocks here

Rem Set the values to the new columns in mlog$, cdc_change_tables$.
Rem The columns were added in c0902000.sql
DECLARE
   oldest_time    DATE;
   oldest_scn     NUMBER;
   cursor c1 IS
     SELECT mowner, master, flag, oldest, oldest_pk, oldest_oid
       FROM sys.mlog$
       ORDER BY mowner, master;
   cursor c2 IS
     SELECT obj#, mvl_flag, mvl_oldest_rid, mvl_oldest_pk, mvl_oldest_oid,
       mvl_oldest_rid_time, mvl_oldest_pk_time, mvl_oldest_oid_time
       FROM sys.cdc_change_tables$
       WHERE bitand(mvl_flag, 128) = 128
       ORDER BY obj#;
BEGIN
   FOR c1_rec IN c1 LOOP
      IF bitand(c1_rec.flag, 1024) = 1024 THEN
         oldest_time := c1_rec.oldest;
         IF oldest_time > c1_rec.oldest_pk THEN
            oldest_time := c1_rec.oldest_pk;
         END IF;
         IF oldest_time > c1_rec.oldest_oid THEN
            oldest_time := c1_rec.oldest_oid;
```

```
        END IF;
        UPDATE sys.mlog$ SET oldest_seq = oldest_time
          WHERE mowner = c1_rec.mowner AND master = c1_rec.master;
      ELSE
        UPDATE sys.mlog$ SET oldest_seq =
          TO_DATE('4000-01-01:00:00:00', 'YYYY-MM-DD:HH24:MI:SS')
          WHERE mowner = c1_rec.mowner AND master = c1_rec.master;
      END IF;
   END LOOP;
   FOR c2_rec IN c2 LOOP
      IF bitand(c2_rec.mvl_flag, 1024) = 1024 THEN
        oldest_scn  := c2_rec.mvl_oldest_rid;
        oldest_time := c2_rec.mvl_oldest_rid_time;
        IF oldest_scn > c2_rec.mvl_oldest_pk THEN
           oldest_scn  := c2_rec.mvl_oldest_pk;
           oldest_time := c2_rec.mvl_oldest_pk_time;
        END IF;
        IF oldest_scn > c2_rec.mvl_oldest_oid THEN
           oldest_scn  := c2_rec.mvl_oldest_oid;
           oldest_time := c2_rec.mvl_oldest_oid_time;
        END IF;
        UPDATE sys.cdc_change_tables$
          SET mvl_oldest_seq = oldest_scn, mvl_oldest_seq_time = oldest_time
          WHERE obj# = c2_rec.obj#;
      ELSE
        UPDATE sys.cdc_change_tables$
          SET mvl_oldest_seq = 2.8147E+14, mvl_oldest_seq_time =
              TO_DATE('4000-01-01:00:00:00', 'YYYY-MM-DD:HH24:MI:SS')
          WHERE obj# = c2_rec.obj#;
      END IF;
   END LOOP;
END;
/

Rem Add components to registry that were not picked up for 9.2
execute dbms_registry_sys.populate_92


Rem Delete type 10 objects from Logminer cache
declare
  no_such_table exception;
  pragma exception_init(no_such_table, -942);
begin
  execute immediate ' delete from system.logmnrc_gtlo ' ||
                    ' where lvl0type# = 10 ';
  commit;
  exception when no_such_table then
    null;
end;
/

Rem Upgrade Logminer Checkpoint Data to 10.1
execute dbms_logmnr_internal.upgrade_ckpt;

Rem ============ Beginning of STREAMS upgrade ========================

declare
  cursor qt_cur is select schema, name, flags, sort_cols from
```

```
                        system.aq$_queue_tables;
  sort_by_eqt      BOOLEAN;
begin
  for qt in qt_cur loop
    begin

      -- patch up time columns
      IF qt.sort_cols = 2 or qt.sort_cols = 3 or qt.sort_cols = 7 THEN
        sort_by_eqt := TRUE;
      ELSE
        sort_by_eqt := FALSE;
      END IF;

      dbms_aqadm_sys.patch_queue_table(qt.schema, qt.name, qt.flags,
                                       sort_by_eqt, TRUE);

      if (sys.dbms_aqadm_sys.mcq_8_1(qt.flags)) then
        sys.dbms_aqadm_sys.create_buffer_view(qt.schema, qt.name);
        sys.dbms_prvtaqim.create_base_view(qt.schema, qt.name, qt.flags);
      else
        sys.dbms_aqadm_sys.create_base_view(qt.schema, qt.name, qt.flags);
      end if;
    exception
      when others then
        null;
    end;
  end loop;
end;
/

Rem ===== AQ related upgrade =============
declare
  cursor qt_cur is select schema, name, flags from system.aq$_queue_tables;
  stmt_buf VARCHAR2(128);
begin
  for qt in qt_cur loop
    if (sys.dbms_aqadm_sys.mcq_8_1(qt.flags)) then
      begin
        sys.dbms_aq_sys_exp_internal.deregister_procedural_action(
          'AQ$_' || qt.name || '_NR', qt.schema);
        stmt_buf := 'DROP TABLE ' || qt.schema || '.AQ$_' || qt.name || '_NR';
        dbms_aqadm_syscalls.kwqa_3gl_ExecuteStmt(stmt_buf);
        sys.dbms_prvtaqim.create_signature_iot(qt.schema, qt.name);
      exception
        when others then
          null;
      end;
    end if;
  end loop;
end;
/

DECLARE
  gname     VARCHAR2(128);
BEGIN
  SELECT global_name INTO gname FROM global_name;
  UPDATE streams$_capture_process
    SET source_dbname = gname,
```

```
        use_dblink = 0;
  COMMIT;
END;
/

Rem canonicalize the procedure names in Streams tables
Rem only upper case names worked correctly in 9.2, so don't worry
Rem about double canonicalization or preserving original case
declare
  canon_value1 varchar(98);
  canon_value2 varchar(98);
  cursor ap_cur is select message_handler, ddl_handler
     from sys.streams$_apply_process
     for update of message_handler, ddl_handler;
  cursor doo_cur is select user_apply_procedure from sys.apply$_dest_obj_ops
    for update of user_apply_procedure;
begin
  -- update streams$_apply_process table
  for ap_rec in ap_cur loop
    dbms_utility.canonicalize(ap_rec.message_handler, canon_value1, 98);
    dbms_utility.canonicalize(ap_rec.ddl_handler, canon_value2, 98);
    update sys.streams$_apply_process
      set message_handler = canon_value1, ddl_handler = canon_value2
      where current of ap_cur;
  end loop;

  -- update apply$_dest_obj_ops table
  for doo_rec in doo_cur loop
    dbms_utility.canonicalize(doo_rec.user_apply_procedure, canon_value1, 98);
    update sys.apply$_dest_obj_ops
      set user_apply_procedure = canon_value1 where current of doo_cur;
  end loop;
end;
/

Rem Add negative rule set column to AQ subscriber tables
Rem expand and canonicalize ruleset_name
Rem canonicalize 8.1 subscriber addresses that contain queue names
DECLARE
  qt_schema        VARCHAR2(30);
  qt_name          VARCHAR2(30);
  qt_flags         NUMBER;
  CURSOR find_qt_c IS SELECT schema, name, flags, objno, sort_cols
                   FROM system.aq$_queue_tables;
  add_col_sql      VARCHAR2(300);
  expand_col_sql   VARCHAR2(300);
  rs_sel_sql       VARCHAR2(300);
  rs_upd_sql       VARCHAR2(300);
  a_sel_sql        VARCHAR2(512);
  a_upd_sql        VARCHAR2(300);
  no_cmprs_sql     VARCHAR2(300);
  no_cmprs_sql2    VARCHAR2(300);
  rebuild_idx_sql  VARCHAR2(300);
  TYPE rs_cur_type IS REF CURSOR;
  rs_cv            RS_CUR_TYPE;
  orig_rs_name     VARCHAR2(61);
  canon_rs_name    VARCHAR2(65);
  sub_id           NUMBER;
```

```
  a_cv             RS_CUR_TYPE;
  old_address      VARCHAR2(1024);
  new_address      VARCHAR2(1024);
  sub_name         VARCHAR2(30);
  schema_canon     VARCHAR2(30);
  name_canon       VARCHAR2(30);
  db_name          VARCHAR2(128);
  db_dom           VARCHAR2(128);
  CURSOR nqrs_c(obj_num NUMBER) IS
    SELECT name FROM system.aq$_queues
      WHERE table_objno = obj_num AND usage = 0;
  sort_by_eqt      BOOLEAN;
  tab_sp1          NUMBER;
  CURSOR rbi_c(tab_owner VARCHAR2, tab_name VARCHAR2) IS
    SELECT owner, index_name
      FROM dba_indexes
      WHERE table_owner=tab_owner
        AND table_name=tab_name
        AND status= 'UNUSABLE';
BEGIN
  FOR q_rec IN find_qt_c LOOP         -- iterate all queue tables
    qt_schema := q_rec.schema;                      -- get queue table schema
    qt_name   := q_rec.name;                        -- get queue table name
    qt_flags  := q_rec.flags;                       -- get queue flags

    IF ((bitand(qt_flags, 8) = 8)) THEN

     -- Cannot add/drop column from compressed table if 9.2 compatible,
     -- so first uncompress the table if it is compressed.
     SELECT t.spare1 INTO tab_sp1
        FROM tab$ t, obj$ o, user$ u
        WHERE t.obj# = o.obj#
          AND u.user# = o.owner#
          AND o.name = qt_name
          AND u.name = qt_schema;
      IF BITAND(tab_sp1, 131072)=131072 THEN             -- table is compressed
        no_cmprs_sql := 'ALTER TABLE "' || qt_schema || '"."' || qt_name
                        || '" MOVE NOCOMPRESS';
       EXECUTE IMMEDIATE no_cmprs_sql;                 -- uncompress the table
        no_cmprs_sql2 := 'ALTER TABLE "' || qt_schema || '"."' || qt_name
                        || '" NOCOMPRESS';
       EXECUTE IMMEDIATE no_cmprs_sql2;                -- uncompress the table

        -- rebuild any unusable indexes for the table
        FOR r IN rbi_c(qt_schema,qt_name) LOOP
          rebuild_idx_sql := 'ALTER INDEX "' || r.owner || '"."'
                             || r.index_name || '" REBUILD';
          EXECUTE IMMEDIATE rebuild_idx_sql;
        END LOOP;
      END IF;

      add_col_sql := 'ALTER TABLE "' || qt_schema || '"."' || qt_name
                     || '" ADD (user_prop sys.anydata)';

     BEGIN
        EXECUTE IMMEDIATE add_col_sql;
      EXCEPTION
        WHEN OTHERS THEN
```

```
    IF sqlcode = -1430 THEN NULL;
    ELSE RAISE;
    END IF;
  END;
END IF;

-- patch up time columns
IF q_rec.sort_cols = 2 or q_rec.sort_cols = 3 or q_rec.sort_cols = 7 THEN
  sort_by_eqt := TRUE;
ELSE
  sort_by_eqt := FALSE;
END IF;

dbms_aqadm_sys.patch_queue_table(qt_schema, qt_name, qt_flags,
                                 sort_by_eqt, TRUE);


IF ((bitand(qt_flags, 8) = 8) AND (bitand(qt_flags, 1) = 1)) THEN
  no_cmprs_sql := 'ALTER TABLE '
                  || '"'||qt_schema||'"' || '.' || '"'||'AQ$_'|| qt_name
                  ||'_S'||'"'
                  || ' MOVE NOCOMPRESS';
  no_cmprs_sql2 := 'ALTER TABLE '
                  || '"'||qt_schema||'"' || '.' || '"'||'AQ$_'|| qt_name
                  ||'_S'||'"'
                  || ' NOCOMPRESS';
  add_col_sql := 'ALTER TABLE '
                  || '"'||qt_schema||'"' || '.' || '"'||'AQ$_'|| qt_name
                  ||'_S'||'"'
                  || ' ADD (negative_ruleset_name VARCHAR2(65))';
  expand_col_sql := 'ALTER TABLE '
                  || '"'||qt_schema||'"' || '.' || '"'||'AQ$_'|| qt_name
                  ||'_S'||'"'
                  || ' MODIFY (ruleset_name VARCHAR2(65))';

  rs_sel_sql := 'SELECT ruleset_name, subscriber_id FROM '
                  || '"'||qt_schema||'"' || '.' || '"'||'AQ$_'|| qt_name
                  ||'_S'||'"'|| ' for update';

  rs_upd_sql := 'UPDATE '
                  || '"'||qt_schema||'"' || '.' || '"'||'AQ$_'|| qt_name
                  ||'_S'||'"'|| ' set ruleset_name = :1 where '
                  ||'subscriber_id = :2';

  -- selecting addresses that need to be canonicalized
  a_sel_sql := 'SELECT name, address, subscriber_id FROM '
                  || '"'||qt_schema||'"' || '.' || '"'||'AQ$_'|| qt_name
                  ||'_S'||'"'|| ' WHERE (protocol is NULL '
                  ||'OR protocol = 0) AND address IS NOT NULL '
                  ||'AND (bitand(subscriber_type, 4) = 4 '
                  ||'  OR bitand(subscriber_type, 2) = 2 '
                  ||'  OR bitand(subscriber_type, 1) = 1) FOR UPDATE';
  -- for updating the address
  a_upd_sql := 'UPDATE '
                  || '"'||qt_schema||'"' || '.' || '"'||'AQ$_'|| qt_name
                  ||'_S'||'"'|| ' set address = :1 WHERE '
                  ||'subscriber_id = :2';
```

```
-- Cannot add/drop column from compressed table if 9.2 compatible,
-- so first uncompress the table if it is compressed.
SELECT t.spare1 INTO tab_sp1
   FROM tab$ t, obj$ o, user$ u
   WHERE t.obj# = o.obj#
     AND u.user# = o.owner#
     AND o.name = 'AQ$_'||qt_name||'_S'
     AND u.name = qt_schema;
 IF BITAND(tab_sp1, 131072)=131072 THEN             -- table is compressed
   EXECUTE IMMEDIATE no_cmprs_sql;                  -- uncompress the table

   EXECUTE IMMEDIATE no_cmprs_sql2;                 -- uncompress the table

   -- rebuild any unusable indexes for the table
   FOR r IN rbi_c(qt_schema,'AQ$_'||qt_name||'_S') LOOP
     rebuild_idx_sql := 'ALTER INDEX "' || r.owner || '"."'
                        || r.index_name || '" REBUILD';
     EXECUTE IMMEDIATE rebuild_idx_sql;
   END LOOP;
 END IF;

 BEGIN
   EXECUTE IMMEDIATE add_col_sql;
 EXCEPTION
   WHEN OTHERS THEN
    IF sqlcode = -1430 THEN NULL;
    ELSE RAISE;
    END IF;
 END;

 -- expand ruleset_name column;
 EXECUTE IMMEDIATE expand_col_sql;

 -- canonicalize ruleset_name values
 -- Only upper case names would have worked in 9.2
 OPEN rs_cv FOR rs_sel_sql;
 LOOP
   FETCH rs_cv into orig_rs_name, sub_id;
   EXIT WHEN rs_cv%NOTFOUND;
   dbms_utility.canonicalize(orig_rs_name, canon_rs_name, 65);
   -- do update
   execute immediate rs_upd_sql using canon_rs_name, sub_id;
 END LOOP;
 CLOSE rs_cv;

 -- for all normal queues in the 81 queue table create a negative rule set
 FOR r IN nqrs_c(q_rec.objno) LOOP
   dbms_prvtaqis.create_queue_rule_set(
     '"'||qt_schema||'"."'||r.name||'_N"', qt_schema , qt_name);
 END LOOP;

 -- add aq evaluation context to all AQ rules
 --
 dbms_prvtaqis.upgrade_rule_10i(qt_schema, qt_name, null);

 -- canonicalize address
 OPEN a_cv FOR a_sel_sql;
 LOOP
```

```
        FETCH a_cv into sub_name, old_address, sub_id;
        EXIT WHEN a_cv%NOTFOUND;

        -- warn user if subscriber name contains quotes
        IF INSTR(sub_name, '"') > 0 THEN
          dbms_system.ksdwrt(dbms_system.alert_file,
            'Subscriber name'|| sub_name ||' contains double quotes. '||
            'Quoted subscribers created when database compatibility level '||
            'is below 10.0 do not work correctly and will cease to work '||
            'once compatibility has been increased to 10.0. ');
        END IF;

        BEGIN
          dbms_aqadm_sys.parse_name(
            dbms_aqadm_sys.AQ_SUBSCRIBER_ADDRESS, old_address, schema_canon,
            name_canon, db_name, db_dom, FALSE);

          new_address := '"'||schema_canon||'"' || '.' ||
                         '"'||name_canon||'"';
          IF db_name IS NOT NULL THEN
            new_address := new_address || '@' || db_name;
            IF db_dom IS NOT NULL THEN
              new_address := new_address || '.' || db_dom;
            END IF;
          END IF;
        EXCEPTION WHEN others THEN
          -- parsing failed, so use old address and write an alert
          new_address := old_address;
          dbms_system.ksdwrt(dbms_system.alert_file,
            'Subscriber '|| sub_name ||' has invalid address '||
            old_address ||'.  Please remove this subscriber since it '||
            'has an unusable address');
        END;

        -- do update
        EXECUTE IMMEDIATE a_upd_sql USING new_address, sub_id;
      END LOOP;
      CLOSE a_cv;

      COMMIT;
    END IF;

  END LOOP;
END;
/

Rem canonicalize subscriber addresses that contain queue names for 8.0 queues.
Rem If address can't be parsed then just leave old value.
DECLARE
  CURSOR subs_cur IS SELECT subscribers FROM system.aq$_queues
                       WHERE subscribers IS NOT NULL FOR UPDATE;
  subs             AQ$_SUBSCRIBERS;
  new_address      VARCHAR2(1024);
  schema_canon     VARCHAR2(30);
  name_canon       VARCHAR2(30);
  db_name          VARCHAR2(128);
  db_dom           VARCHAR2(128);
BEGIN
```

```
  FOR subs_rec IN subs_cur LOOP
    subs := subs_rec.subscribers;
    FOR i IN 1..subs.COUNT LOOP
      -- if a subscriber with an address is found and protocol is null or 0
      -- then put the address into 10.1 canonical form.
      IF subs.EXISTS(i) AND subs(i) IS NOT NULL
         AND subs(i).address IS NOT NULL
         AND (subs(i).protocol IS NULL OR subs(i).protocol = 0) THEN
        -- parse name to get components and form new name.
        -- catch exceptions and leave value alone
        BEGIN
          dbms_aqadm_sys.parse_name(
            dbms_aqadm_sys.AQ_SUBSCRIBER_ADDRESS, subs(i).address,
            schema_canon, name_canon, db_name, db_dom, FALSE);

            new_address := '"'||schema_canon||'"' || '.' ||
                           '"'||name_canon||'"';
            IF db_name IS NOT NULL THEN
              new_address := new_address || '@' || db_name;
              IF db_dom IS NOT NULL THEN
                new_address := new_address || '.' || db_dom;
              END IF;
            END IF;
            -- assign new address
            subs(i).address := new_address;
         EXCEPTION WHEN others THEN
           dbms_system.ksdwrt(dbms_system.alert_file,
             'Subscriber '|| subs(i).name ||' with address '||
             subs(i).address ||' has an invalid address.' ||
             'Please remove this subscriber since it has an unusable address');
         END;
      END IF;
    END LOOP;
    -- update with new subscribers entry
    UPDATE system.aq$_queues SET subscribers = subs
      WHERE CURRENT OF subs_cur;
  END LOOP;
  COMMIT;
END;
/

Rem change expact$ params to 10.1 format
DECLARE
  -- cursor to select non-subscriber AQ expact$ entries
  CURSOR uargs1_cur IS SELECT user_arg FROM sys.expact$ WHERE
                             func_schema = 'SYS'
                             AND func_package = 'DBMS_AQ_IMPORT_INTERNAL'
                             AND func_proc != 'AQ_EXPORT_SUBSCRIBER'
                       FOR UPDATE;
  -- cursor to select subscriber AQ expact$ entries
  CURSOR uargs2_cur IS SELECT user_arg FROM sys.expact$ WHERE
                             func_schema = 'SYS'
                             AND func_package = 'DBMS_AQ_IMPORT_INTERNAL'
                             AND func_proc = 'AQ_EXPORT_SUBSCRIBER'
                       FOR UPDATE;
  dqt               CONSTANT VARCHAR2(1) := '"';
  new_uargs         VARCHAR2(2000);
```

```
-- this procedure parses the expact entry
PROCEDURE aq_comma_to_table(arg_list IN     VARCHAR2,
                            argc     OUT    BINARY_INTEGER,
                            argv     OUT    DBMS_UTILITY.UNCL_ARRAY) IS

c_pos     BINARY_INTEGER := 1;
c_old_pos BINARY_INTEGER := 1;

BEGIN
  argc := 1;
  LOOP
    c_pos := INSTR(arg_list, ',', c_pos);

    IF c_pos != 0 THEN
      argv(argc) := SUBSTR(arg_list, c_old_pos, c_pos-c_old_pos);
    ELSE
      argv(argc) := SUBSTR(arg_list, c_old_pos,
                           LENGTH(arg_list)-c_old_pos+1);
    END IF;
    EXIT WHEN c_pos = 0;

    c_pos     := c_pos + 1;
    c_old_pos := c_pos;
    argc      := argc + 1;

  END LOOP;
END aq_comma_to_table;

-- this function returns the new expact string
FUNCTION get_new_sub_expact_string(arg_list IN VARCHAR2) RETURN VARCHAR2 IS
  argv           dbms_utility.uncl_array;
  argc           BINARY_INTEGER;
  queue_name     VARCHAR2(30);
  sub_name       VARCHAR2(30);
  sub_dest       VARCHAR2(1024);
  sub_pro        VARCHAR2(30);
  canon_sub_dest VARCHAR2(1024);
  new_str        VARCHAR2(2000);             -- the 10.1 format expact string
  schema_canon   VARCHAR2(30);
  name_canon     VARCHAR2(30);
  db_name        VARCHAR2(128);
  db_dom         VARCHAR2(128);
BEGIN
  aq_comma_to_table(arg_list, argc, argv);

  -- extract the subscriber fields from the argument
  queue_name := argv(1);
  IF (LENGTH(argv(2)) > 4) THEN
    --all the data after the first 4 char
    sub_name   :=  SUBSTR(argv(2), 5);
  ELSE
    sub_name   := NULL;
  END IF;

  IF (LENGTH(argv(3)) > 4) THEN
    --all the data after the first 4 char
    sub_dest   := SUBSTR(argv(3), 5);
  ELSE
```

```
      sub_dest   := NULL;
    END IF;

    IF (LENGTH(argv(4)) > 4) THEN
      --all the data after the first 4 char
      sub_pro   := TO_NUMBER(SUBSTR(argv(4), 5));
    ELSE
      sub_pro   := NULL;
    END IF;

    canon_sub_dest := sub_dest;
    -- only parse the address if internal protocol 0 and address is not NULL
    IF sub_pro = 0 AND sub_dest IS NOT NULL THEN
      BEGIN
        dbms_aqadm_sys.parse_name(
          dbms_aqadm_sys.AQ_SUBSCRIBER_ADDRESS, sub_dest, schema_canon,
          name_canon, db_name, db_dom, FALSE);

        canon_sub_dest := '"'||schema_canon||'"' || '.' ||
                          '"'||name_canon||'"';
        IF db_name IS NOT NULL THEN
          canon_sub_dest := canon_sub_dest|| '@' || db_name;
          IF db_dom IS NOT NULL THEN
             canon_sub_dest := canon_sub_dest || '.' || db_dom;
           END IF;
        END IF;
      EXCEPTION WHEN others THEN
        -- parsing failed, so use old address and write an alert
        canon_sub_dest := sub_dest;
        dbms_system.ksdwrt(dbms_system.alert_file,
          'Subscriber '|| sub_name ||' has invalid address '||
           sub_dest ||'.  Please remove this subscriber since it '||
          'has an unusable address');
      END;
    END IF;

    new_str := dqt||queue_name||dqt || ',' || dqt||'NAME'||sub_name||dqt ||
               ',' || dqt||'ADDR'||canon_sub_dest||dqt||
               ',' || dqt||'PROT'||TO_CHAR(sub_pro)||dqt;

    RETURN new_str;
  END get_new_sub_expact_string;
BEGIN
  -- convert non-subscriber entries
  FOR uargs1_rec IN uargs1_cur LOOP
    -- for idempotence, don't change string if it starts with a double quote
    IF SUBSTR(uargs1_rec.user_arg, 1, 1) != '"' THEN
      -- replace ',' with '",",' and add a quote at begining and end
      new_uargs := REPLACE(uargs1_rec.user_arg, ',', '",",');
      new_uargs := '"'||new_uargs||'"';
      UPDATE sys.expact$ SET user_arg = new_uargs WHERE CURRENT OF uargs1_cur;
    END IF;
  END LOOP;

  -- convert subscriber entries
  FOR uargs2_rec IN uargs2_cur LOOP
    -- for idempotence, only put string in new format if it does not
    -- start with a double quote.
```

```
    IF SUBSTR(uargs2_rec.user_arg, 1, 1) != '"' THEN
      new_uargs := get_new_sub_expact_string(uargs2_rec.user_arg);
      UPDATE sys.expact$ SET user_arg = new_uargs WHERE CURRENT OF uargs2_cur;
    END IF;
  END LOOP;

  COMMIT;
END;
/

Rem ===== add other tables as dependent on queue table
declare
  cursor qt_cur is select schema, name, flags
   from system.aq$_queue_tables where schema != 'SYS';
  stmt_buf VARCHAR2(128);
begin
  for qt in qt_cur loop
    if (sys.dbms_aqadm_sys.mcq_8_1(qt.flags)) then
      begin
        -- Export subscriber table with queue table
        dbms_aqadm_sys.add_qtab_expdep(qt.schema, qt.name,
                                       'AQ$_'||qt.name||'_S');
        -- Export dequeue iot with queue table
        dbms_aqadm_sys.add_qtab_expdep(qt.schema, qt.name,
                                       'AQ$_'||qt.name||'_I');
        -- Export history iot with queue table
        dbms_aqadm_sys.add_qtab_expdep(qt.schema, qt.name,
                                       'AQ$_'||qt.name||'_H');
        -- Export timemanager iot with queue table
        dbms_aqadm_sys.add_qtab_expdep(qt.schema, qt.name,
                                       'AQ$_'||qt.name||'_T');
        -- Export signature iot with queue table
        dbms_aqadm_sys.add_qtab_expdep(qt.schema, qt.name,
                                       'AQ$_'||qt.name||'_G');
      exception
        when others then
          null;
      end;
    end if;
  end loop;
end;
/

Rem Update first_scn in the capture process
UPDATE streams$_capture_process c
  SET c.first_scn = (SELECT l.first_scn
                     FROM dba_logmnr_session l
                     WHERE c.logmnr_sid = l.id);
COMMIT;

Rem Turn on hotmining for RAC
BEGIN
  IF dbms_utility.is_cluster_database THEN
    UPDATE system.logmnr_session$ x
      SET session_attr = DECODE(BITAND(session_attr, 8388608), 8388608,
                                session_attr, session_attr+8388608)
      WHERE EXISTS (SELECT c.logmnr_sid
                    FROM sys.streams$_capture_process c
```

```
                    WHERE c.LOGMNR_SID = x.session#);
    COMMIT;
  END IF;
END;
/

Rem ============ Beginning of PL/SQL upgrade ========================
Rem
Rem Create new PL/SQL compiler paramters from the old plsql_compiler_flags
Rem

DECLARE
  plsql_debug     VARCHAR2(32);
  plsql_code_type VARCHAR2(32);

  PROCEDURE parse_compiler_flags(plsql_compiler_flags IN OUT nocopy VARCHAR2,
                                 plsql_code_type      OUT VARCHAR2,
                                 plsql_debug          OUT VARCHAR2) AS
    i pls_integer;
    v VARCHAR2(256);
    d VARCHAR2(32);
  BEGIN
    plsql_debug     := 'FALSE';
    plsql_code_type := 'INTERPRETED';
    LOOP
      plsql_compiler_flags := ltrim(plsql_compiler_flags);
      EXIT WHEN plsql_compiler_flags IS NULL;
      i := instr(plsql_compiler_flags, ',');
      IF (i > 0) THEN
        v := substr(plsql_compiler_flags, 1, i-1);
        plsql_compiler_flags := substr(plsql_compiler_flags, i+1);
      ELSE
        v := plsql_compiler_flags;
        plsql_compiler_flags := NULL;
      END IF;
      v := upper(rtrim(v));

      CASE v
        WHEN 'DEBUG' THEN
          plsql_debug := 'TRUE';  d := v;
        WHEN 'NON_DEBUG' THEN
          plsql_debug := 'FALSE'; d := v;
        WHEN 'INTERPRETED' THEN
          plsql_code_type := v;
        WHEN 'NATIVE' THEN
          plsql_code_type := v;
        ELSE
          NULL;
      END CASE;
    END LOOP;
    plsql_compiler_flags := plsql_code_type || ', ' || d;
  END;

BEGIN

  DELETE FROM settings$ WHERE param IN ('plsql_debug', 'plsql_code_type');

  FOR r IN (SELECT * FROM settings$ WHERE param = 'plsql_compiler_flags') LOOP
```

```
    parse_compiler_flags(r.value, plsql_code_type, plsql_debug);
    INSERT INTO settings$ (obj#, param, value)
      VALUES(r.obj#, 'plsql_debug', plsql_debug);
    INSERT INTO settings$ (obj#, param, value)
      VALUES(r.obj#, 'plsql_code_type', plsql_code_type);
    UPDATE settings$ SET value = r.value
      WHERE obj# = r.obj# AND param = 'plsql_compiler_flags';
  END LOOP;

  COMMIT;

END;
/
Rem ============================== End of PL/SQL upgrade ================

Rem Set streams unsupported bit in tab$.trigflag if required
Rem 0x10000000 flag in tab$.trigflag indicates streams unsupported table
DECLARE
  -- find a list of updatable snapshot logs
  CURSOR uslog_tabs IS
    SELECT o.obj#
    FROM   snap$ s, obj$ o, user$ u
    WHERE  u.name = s.sowner
    AND    o.name = s.uslog
    AND    o.owner# = u.user#
    AND    BITAND(s.flag, 2) = 2;

  -- find a list of mv logs
  CURSOR mvlog_tabs IS
    SELECT o.obj#
    FROM   mlog$ m, obj$ o, user$ u
    WHERE  u.name = m.mowner
    AND    o.name = m.log
    AND    o.owner# = u.user#;

BEGIN
  -- mv logs
  FOR mvlog_objn IN mvlog_tabs LOOP
    UPDATE sys.tab$ t
    SET    t.trigflag = (t.trigflag + 268435456)
    WHERE  t.obj# = mvlog_objn.obj#
    AND    BITAND(t.trigflag, 268435456) != 268435456;
  END LOOP;

  COMMIT;

  -- updatable snapshot logs
  FOR uslog_objn IN uslog_tabs LOOP
    UPDATE sys.tab$ t
    SET    t.trigflag = (t.trigflag + 268435456)
    WHERE  t.obj# = uslog_objn.obj#
    AND    BITAND(t.trigflag, 268435456) != 268435456;
  END LOOP;

  COMMIT;

  -- TODO : cdc logs
END;
```

```
/

-- Populate sname and oname properly.
BEGIN
  FOR rec IN (select rid, sname, oname from
                (select do.rowid rid, u.username sname, o.name oname
                   from sys.obj$ o, apply$_dest_obj_ops do, dba_users u
                     where o.obj# = do.object_number
                       and o.owner# = u.user_id
                       and o.remoteowner is NULL
               union
                 select do.rowid rid, o.remoteowner sname, o.name oname
                   from sys.obj$ o, apply$_dest_obj_ops do
                     where o.obj# = do.object_number
                       and o.remoteowner is not null)) LOOP
    UPDATE sys.apply$_dest_obj_ops a
      SET a.sname = rec.sname, a.oname = rec.oname
        WHERE a.rowid = rec.rid;
  END LOOP;
  COMMIT;
  -- Bug 3880023:  Some obj#s will not exist (if the table was dropped),
  -- delete these rows as they are meaningless in 10.1+
  DELETE FROM sys.apply$_dest_obj_ops WHERE sname IS NULL AND oname IS NULL;
  COMMIT;
END;
/

-- ensure those columns are not null
alter table sys.apply$_dest_obj_ops modify
(sname varchar2(30) not null,
 oname varchar2(30) not null)
/

drop index i_apply_dest_obj_ops1;

create unique index i_apply_dest_obj_ops1 on
  apply$_dest_obj_ops (sname, oname, apply_operation, apply_name);

Rem ============================== End of STREAMS upgrade ===============

Rem
Rem Complete Logical Standby upgrade with migration of metadata to SYSAUX
Rem
execute dbms_logstdby.set_tablespace('SYSAUX');

Rem =======================================================================
Rem END STAGE 1: upgrade from 9.2.0 to 10.1
Rem =======================================================================

Rem =======================================================================
Rem BEGIN STAGE 2: invoke script for subsequent release
Rem =======================================================================
Rem

@@a1001000

Rem =======================================================================
Rem END STAGE 2: invoke script for subsequent release
```

```
Rem ========================================================================

Rem ************************************************************************
Rem END a0902000.sql
Rem ************************************************************************
```