Tipps & Tricks: AQ Remote

Bereich:	PL/SQL, DBA	Erstellung:	01/2004 MP
Versionsinfo:	9.2 -> 10.2, 10.2 -> 11.1	Letzte Überarbeitung:	05/2009 HA

Advanced Queuing auf Remote Datenbanken

Die Schnittstelle Advanced Queuing erfreut sich wachsender Beliebtheit. Leider sind die kryptischen Prozeduren für viele ein Stolperstein. In unserem Beispiel wollen wir eine Datenverbindung zwischen zwei Datenbanken herstellen. Wir arbeiten mit einer lokalen (bei uns 9.2) und einer Remote Datenbank (10g. Rel.2). Andere Kombinationen sind sicherlich auch möglich, werden aber nicht explizit erwähnt.

AQ Skript (optimiert für SQL*Plus)

```
REM ##################################################
REM  Einstellungen für Remote DB
REM ##################################################

REM Wenn Sie SQL*Plus verwenden, können Sie hier die Remote
REM Instanz eintragen
DEFINE remote=o10g2

CONNECT system/manager@&&remote
ALTER SYSTEM SET aq_tm_processes=5;
REM setzen Sie aq_tm_processes niedriger als 10, ansonsten kann
REM der QMNC-Prozess 100% CPU belegen (Bug: 5069930)
ALTER SYSTEM SET job_queue_processes=10;

CREATE USER aq IDENTIFIED BY aq;
GRANT EXECUTE ON DBMS_AQADM TO aq;
GRANT AQ_ADMINISTRATOR_ROLE TO aq;
GRANT CONNECT, RESOURCE TO aq;
GRANT EXECUTE ON dbms_aq TO aq;

CONNECT aq/aq@&&remote

REM Auf der Remote-Seite müssen auch die Grundlagen
REM für die Queues erstellt werden
CREATE OR REPLACE TYPE queue_type AS OBJECT
(subject     VARCHAR2(400),
 text         VARCHAR2(4000));
/

BEGIN
  DBMS_AQADM.CREATE_QUEUE_TABLE(
    queue_table        =>'Q_MULTI_TAB',
    queue_payload_type => 'QUEUE_TYPE',
    multiple_consumers => TRUE);

  DBMS_AQADM.CREATE_QUEUE(
    queue_table => 'Q_MULTI_TAB',
```

```
    queue_name  => 'Q_MULTI_REMOTE');

  DBMS_AQADM.START_QUEUE('Q_MULTI_REMOTE');
END;
/

CREATE OR REPLACE PROCEDURE RECEIVE
(p_timeout IN NUMBER DEFAULT 120)
IS
   v_options   DBMS_AQ.dequeue_options_t;
   v_props     DBMS_AQ.message_properties_t;
   v_handle    RAW(16);
   v_msg       queue_type;
   ex_timeout  EXCEPTION;
   PRAGMA EXCEPTION_INIT( ex_timeout, -25228);
BEGIN
   v_options.consumer_name := 'USER1';
   v_options.wait:= p_timeout;
   v_options.dequeue_mode:=dbms_aq.remove;
   DBMS_OUTPUT.PUT_LINE('Folgende Zeilen standen in der Queue:');
   LOOP
       DBMS_AQ.DEQUEUE('q_multi_remote',
                        v_options,
                        v_props,
                        v_msg,
                        v_handle);
       DBMS_OUTPUT.PUT_LINE(v_msg.subject||' '||v_msg.text);
       COMMIT;
   END LOOP;
EXCEPTION
   WHEN ex_timeout THEN -- keine weiteren Nachrichten
      NULL;
END;
/

REM ##################################################
REM Einstellungen für die Lokale DB
REM ##################################################

CONNECT system/manager
ALTER SYSTEM SET aq_tm_processes=5;
ALTER SYSTEM SET job_queue_processes=10;


CREATE USER aq IDENTIFIED BY aq;
GRANT EXECUTE ON dbms_aqadm TO aq;
GRANT EXECUTE ON dbms_aq TO aq;
GRANT AQ_ADMINISTRATOR_ROLE TO aq;
GRANT CONNECT, RESOURCE TO aq;
REM Ab 10g zusätzlich nötig:
GRANT CREATE DATABASE LINK TO aq;

REM Verbinden Sie sich auf die lokale Datenbank als AQ Benutzer
CONNECT aq/aq
```

```
REM Erstellen Sie einen DB-Link zur gewünschten Remote DB
REM Passen Sie hier bitte die SID und den Hostnamen an !
ALTER SESSION SET global_names=false;

REM ACHTUNG bei 11g als Ziel-Datenbank!
REM Hier sind Passwörter per Default case-sensitive
REM Deshalb Angabe in Hochkommata:
REM CREATE DATABASE LINK remote CONNECT TO AQ
REM IDENTIFIED BY "aq" USING '&&remote';

CREATE DATABASE LINK remote CONNECT TO aq
IDENTIFIED BY aq USING '&&remote';

REM Wenn Sie über keinen Eintrag zur Remote DB in der TNSNAMES.ORA
REM verfügen, können Sie auch diesen DB-Link verwenden:
REM CREATE DATABASE LINK remote CONNECT TO aq
REM IDENTIFIED BY aq USING
REM '(DESCRIPTION =(ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)
REM (HOST = localhost)(PORT = 1521)))
REM (CONNECT_DATA =(SID = o10g2)(SERVER = DEDICATED)))';

REM Testen Sie den DB-Link
SELECT * from global_name@remote;

REM Jetzt wird ein Objekttyp mit zwei Attributen (Spalten) angelegt
CREATE OR REPLACE TYPE queue_type AS OBJECT
(subject     VARCHAR2(400),
 text         VARCHAR2(4000));
/

REM Wir erstellen eine Queue Tabelle incl. Queue
REM und starten diese auch gleich
BEGIN
  DBMS_AQADM.CREATE_QUEUE_TABLE(
    queue_table        =>'Q_MULTI_TAB',
    queue_payload_type => 'QUEUE_TYPE',
    multiple_consumers => TRUE);

  DBMS_AQADM.CREATE_QUEUE(
    queue_table => 'Q_MULTI_TAB',
    queue_name  => 'Q_MULTI_REMOTE');

  DBMS_AQADM.START_QUEUE(
  queue_name => 'Q_MULTI_REMOTE',
  enqueue    => TRUE,
  dequeue    => TRUE);
END;
/

REM Remote Empfänger einrichten
BEGIN DBMS_AQADM.ADD_SUBSCRIBER (
     queue_name => 'Q_MULTI_REMOTE',
     subscriber => sys.aq$_agent('USER1','AQ.Q_MULTI_REMOTE@REMOTE',0));
END;
/
```

```
REM Zeitplan für die Übertragung einrichten (hier im Minutentakt)
BEGIN
   DBMS_AQADM.SCHEDULE_PROPAGATION(
      queue_name  => 'AQ.Q_MULTI_REMOTE',
      destination => 'REMOTE',
      start_time  => SYSDATE,
      duration    => '',
      next_time   => 'SYSDATE+1/(24*60)',
      latency     => '60');
END;
/

CREATE OR REPLACE PROCEDURE SEND(
   p_subject   IN VARCHAR2
        DEFAULT 'Zeit: '||TO_CHAR(SYSDATE,'DD HH24:MI:SS'),
   p_text      IN VARCHAR2 DEFAULT 'Ohne Worte :-) '
)
IS
   v_msg        queue_type:= queue_type(p_subject, p_text);
   v_options    DBMS_AQ.enqueue_options_t;
   v_props      DBMS_AQ.message_properties_t;
   v_handle     RAW(16);
BEGIN
   DBMS_AQ.ENQUEUE('q_multi_remote', v_options, v_props, v_msg, v_handle);
   DBMS_OUTPUT.PUT_LINE('Nachricht in Queue gestellt');
   COMMIT;
END;
/


SET SERVEROUTPUT ON
REM Aufruf der Sendeprozedur
EXEC send


CONNECT aq/aq@&&remote

SET SERVEROUTPUT ON
REM Aufruf der Empfangsprozedur
EXEC receive(2);
```

Es sollte nun von der lokalen Datenbank ein Datensatz zur Remote Datenbank übertragen werden. Wir wünschen viel Spaß bei weiteren Experimenten. Weitere Infos und Beispiele zu AQ erhalten Sie in unserem PL/SQL II Kurs. Ich hoffe wir sehen uns dort. :-)