УО «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра информационных систем и технологий

Лабораторная работа №3

РАСПРЕДЕЛЕННЫЕ ТРАНЗАКЦИИ В ORACLE

Выполнили студенты

3 курса 1 группы факультета ИТ:

Литягин А.А.

Божко Д.В.

Проверила:

Преподаватель-стажер

Комарова Е.И.

Минск 2022

Цель работы: смоделировать распределенную транзакцию, которая блокируется и будет ожидать освобождение ресурса на удаленном сервере; смоделировать распределенную транзакцию, у которой нарушается ограничение целостности на стороне уделенного сервера.

Листинг кода:

|  |
| --- |
| create user LinkUser identified by Pa$$w0rd;  GRANT create any table, connect, create public database link, SELECT ANY SEQUENCE to LinkUser;  alter user LinkUser quota unlimited on Users;  GRANT restricted session to LinkUser;  -- db1  CREATE PUBLIC DATABASE LINK RIS\_DBLINK  CONNECT TO LinkUser IDENTIFIED BY Pa$$w0rd  USING 'orcl3';  -- db2  drop public database link RIS\_DBLINK;  CREATE PUBLIC DATABASE LINK RIS\_DBLINK  CONNECT TO LinkUser IDENTIFIED BY Pa$$w0rd  USING '(DESCRIPTION=    (ADDRESS=(PROTOCOL=TCP)               (HOST=192.168.155.10)(PORT=1521))    (CONNECT\_DATA=(SERVICE\_NAME=orcl3)))';  drop table LinkUser.t\_linkuser;  create table LinkUser.t\_linkuser (      id int generated by default as identity,      payload int check (payload > 0),      CONSTRAINT T1\_PK primary key(id)  ) tablespace users;  create or replace procedure LinkUser.test\_dblink1 as  begin      insert into LinkUser.t\_linkuser (payload) values (1);      insert into LinkUser.t\_linkuser@RIS\_DBLINK (payload) values (1);        update LinkUser.t\_linkuser set payload = 2 where payload = 1;      update LinkUser.t\_linkuser@RIS\_DBLINK set payload = 2 where payload = 1;      commit;  EXCEPTION      WHEN OTHERS THEN begin ROLLBACK; RAISE; end;  end;  /  create or replace procedure LinkUser.test\_dblink2 as  begin        delete LinkUser.t\_linkuser where payload = 2;      delete LinkUser.t\_linkuser@RIS\_DBLINK where payload = 2;      commit;  EXCEPTION      WHEN OTHERS THEN begin ROLLBACK; RAISE; end;  end;  create or replace procedure LinkUser.test\_dblink3 as  begin        insert into LinkUser.t\_linkuser (payload) values (1);      insert into LinkUser.t\_linkuser@RIS\_DBLINK (payload) values (-1);        commit;  EXCEPTION      WHEN OTHERS THEN begin ROLLBACK; RAISE; end;  end;    exec LinkUser.test\_dblink1;  select \* from LinkUser.t\_linkuser@RIS\_DBLINK;  exec LinkUser.test\_dblink2;  select \* from LinkUser.t\_linkuser@RIS\_DBLINK;  exec LinkUser.test\_dblink3;  select \* from LinkUser.t\_linkuser@RIS\_DBLINK;  select \* from LinkUser.t\_linkuser;  connect sys as sysdba;  GRANT execute ON DBMS\_LOCK TO LinkUser;  create or replace procedure LinkUser.test\_dblink4 as  begin      LOCK TABLE LinkUser.t\_linkuser IN EXCLUSIVE MODE;      LOCK TABLE LinkUser.t\_linkuser@RIS\_DBLINK IN EXCLUSIVE MODE;  --    delete LinkUser.t\_linkuser;  --    delete LinkUser.t\_linkuser@RIS\_DBLINK;      --DBMS\_LOCK.SLEEP(10);  --    insert into LinkUser.t\_linkuser (payload) values (1);  --    insert into LinkUser.t\_linkuser@RIS\_DBLINK (payload) values (1);      commit;  end;  commit;  connect LinkUser/Pa$$w0rd;  rollback;  SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;  exec LinkUser.test\_dblink4; |

1. На обоих серверах создаем пользователей в рамках PDBORCL.

|  |
| --- |
| create user LinkUser identified by Pa$$w0rd;  GRANT create any table, connect, create public database link, SELECT ANY SEQUENCE to LinkUser;  alter user LinkUser quota unlimited on Users;  GRANT restricted session to LinkUser; |

1. Создаем две таблицы на разных серверах в схемах созданных пользователей.

|  |
| --- |
| create table LinkUser.t\_linkuser (      id int generated by default as identity,      payload int check (payload > 0),      CONSTRAINT T1\_PK primary key(id)  ) tablespace users; |

1. Устанавливаем DBLINK типа USER1-USER1 между пользователями, расположенными на двух серверах.

|  |
| --- |
| CREATE PUBLIC DATABASE LINK RIS\_DBLINK  CONNECT TO LinkUser IDENTIFIED BY Pa$$w0rd  USING '(DESCRIPTION=    (ADDRESS=(PROTOCOL=TCP)               (HOST=192.168.155.10)(PORT=1521))    (CONNECT\_DATA=(SERVICE\_NAME=orcl3)))'; |

1. Разрабатываем скрипт, пытающийся добавить с таблицу недопустимое значение. Скрипт работает на разных серверах. При выполнении выдает ошибку, представленную на рисунке 1.

|  |
| --- |
| create or replace procedure LinkUser.test\_dblink3 as  begin        insert into LinkUser.t\_linkuser (payload) values (1);      insert into LinkUser.t\_linkuser@RIS\_DBLINK (payload) values (-1);        commit;  EXCEPTION      WHEN OTHERS THEN begin ROLLBACK; RAISE; end;  end; |

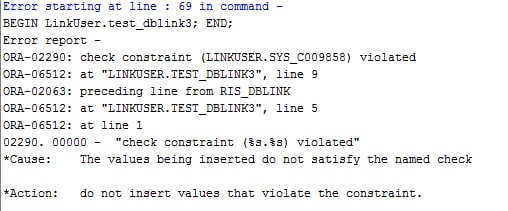


Рисунок 1 – Консоль при выполнении скрипта добавления недопустимого значения

1. Разрабатываем распределенную транзакцию, по условиям которой выполнение скрипта вторым сервером невозможно, пока задачу не закончит первый. При выполнении данной транзакции происходит ожидание, продемонстрированное на рисунке 2.

|  |
| --- |
| create or replace procedure LinkUser.test\_dblink4 as  begin      LOCK TABLE LinkUser.t\_linkuser IN EXCLUSIVE MODE;      LOCK TABLE LinkUser.t\_linkuser@RIS\_DBLINK IN EXCLUSIVE MODE;  --    delete LinkUser.t\_linkuser;  --    delete LinkUser.t\_linkuser@RIS\_DBLINK;      --DBMS\_LOCK.SLEEP(10);  --    insert into LinkUser.t\_linkuser (payload) values (1);  --    insert into LinkUser.t\_linkuser@RIS\_DBLINK (payload) values (1);      commit;  end; |

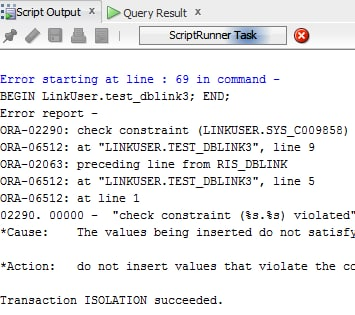


Рисунок 2 – Консоль второго сервера при выполнении распределенной транзакции

Вывод: в ходе лабораторной работы мы смоделировали распределенную транзакцию, которая блокируется и будет ожидать освобождение ресурса на удаленном сервере, смоделировали распределенную транзакцию, у которой нарушается ограничение целостности на стороне уделенного сервера. По итогу можно сказать, что распределенная транзакция требует стабильной работы и подключения обоих серверов и эффективна в случаях, когда несколько пользователей должны взаимодействовать с одним и тем же ресурсом.