**Bài tập**

***1. Transaction control***

create table Project (id number primary key,

pname varchar2(50),

cost number);

insert into project values (1, 'jupiter', 2000);

insert into project values (2, 'saturn', 1000);

insert into project values (3, 'mercury', 15000);

commit;

Viết kết quả thực thi vào cột output:

**1. Rollback**

| **Time** | **session** | **output** |
| --- | --- | --- |
| t0 | COMMIT; | Table PROJECT created.  1 row inserted.  1 row inserted.  1 row inserted.  Commit complete. |
| t1 | SET TRANSACTION NAME 'cost\_update'; | Transaction NAME succeeded. |
| t2 | SELECT XID, name, STATUS FROM V$TRANSACTION; | 05001B006C060000 ACTIVE |
| t3 | UPDATE project  SET cost = 8000  WHERE id = 1; | 1 row updated. |
| t4 | SELECT XID, name, STATUS FROM V$TRANSACTION; | 0A00160043310000 cost\_update ACTIVE  05001B006C060000 ACTIVE |
| t5 | SELECT \* FROM project; | 1 jupiter 8000  2 saturn 1000  3 mercury 15000 |
| t6 | ROLLBACK; | Rollback complete. |
| t7 | SELECT \* FROM project; | 1 jupiter 2000  2 saturn 1000  3 mercury 15000 |
| t8 | SELECT XID, name, STATUS FROM V$TRANSACTION; | 05001B006C060000 ACTIVE |

**2. Commit**

| **Time** | **session** | **output** |
| --- | --- | --- |
| t1 | COMMIT; | Commit complete. |
| t2 | SELECT XID, name, STATUS FROM V$TRANSACTION; | 05001B006C060000 ACTIVE |
| t3 | UPDATE project  SET cost = 6000  WHERE id = 2; | 1 row updated. |
| t4 | SELECT XID, name, STATUS FROM V$TRANSACTION; | 03000F0070060000 ACTIVE |
| t5 | Insert into project values (4, 'neptune', 19000); | 1 row inserted. |
| t6 | SELECT XID, name, STATUS FROM V$TRANSACTION; | 03000F0070060000 ACTIVE |
| t7 | COMMIT; | Commit complete. |
| t8 | SELECT \* FROM project; | 1 jupiter 2000  2 saturn 6000  3 mercury 15000  4 neptune 19000 |
| t9 | SELECT XID, name, STATUS FROM V$TRANSACTION; | No roll |

**3. Savepoint**

| **Time** | **session** | **output** |
| --- | --- | --- |
| t0 | COMMIT; | Commit complete. |
| t1 | Select \* from project; | 1 jupiter 2000  2 saturn 6000  3 mercury 15000  4 neptune 19000 |
| t2 | Update project set cost=400000  where pname='jupiter'; | 1 row updated. |
| t3 | SAVEPOINT after\_jupiter\_cost; | Savepoint created. |
| t4 | Update project set cost=130  where pname='jupiter'; | 1 row updated. |
| t5 | SAVEPOINT after\_mercury\_cost; | Savepoint created. |
| t6 | ROLLBACK TO SAVEPOINT after\_jupiter\_cost; | Rollback complete. |
| t7 | Select \* from project; | 1 jupiter 400000  2 saturn 6000  3 mercury 15000  4 neptune 19000 |
| t8 | Update project set cost=170  where pname='mercury'; | 1 row updated. |
| t9 | ROLLBACK; | Rollback complete. |
| t10 | Select \* from project; | 1 jupiter 2000  2 saturn 6000  3 mercury 15000  4 neptune 19000 |

**4. DDL**

| **Time** | **session** | **output** |
| --- | --- | --- |
| t0 | COMMIT; | Commit complete. |
| t1 | Select \* from project; | 1 jupiter 2000  2 saturn 6000  3 mercury 15000  4 neptune 19000 |
| t2 | SET TRANSACTION NAME 'cost\_update2'; | Transaction NAME succeeded. |
| t3 | Update project set cost=12300  where pname='jupiter'; | 1 row updated. |
| t4 | Select \* from project; | 1 jupiter 12300  2 saturn 6000  3 mercury 15000  4 neptune 19000 |
| t5 | --DDL statement  Create table test (id number); | Table TEST created. |
| t6 | Insert into test values (26); | 1 row inserted. |
| t7 | Rollback; | Rollback complete. |
| t8 | Select \* from project;  Select \* from test; | 1 jupiter 12300  2 saturn 6000  3 mercury 15000  4 neptune 19000  No row. |

From SQLDeveloper:

You can right click on a connection and chose **'Open SQL Worksheet'** it will create another window for the existing session. (Use Alt + F10and select the connection from the list).

If you need to create another **independent** session you can use Ctrl + Shift + N for an ongoing session.

Open two session (Ctrl + Shift + N) and do the following things.

1. Compare data at time t3 and t5

| **Time** | **Session1** | **Output1** | **Session2** | **Output2** |
| --- | --- | --- | --- | --- |
| t0 | SET TRANSACTION NAME 'cost\_update3'; | Transaction NAME succeeded. |  |  |
| t1 | Select \* from project; | 1 jupiter 12300  2 saturn 6000  3 mercury 15000  4 neptune 19000 |  |  |
| t2 | Update project set cost=467  where pname='jupiter'; | 1 row updated. |  |  |
| t3 | Select \* from project; | 1 jupiter 467  2 saturn 6000  3 mercury 15000  4 neptune 19000 | Select \* from project; | 1 jupiter 12300  2 saturn 6000  3 mercury 15000  4 neptune 19000 |
| t4 | Rollback; | Rollback complete. |  |  |
| t5 | Select \* from project; | 1 jupiter 12300  2 saturn 6000  3 mercury 15000  4 neptune 19000 | Select \* from project; | 1 jupiter 12300  2 saturn 6000  3 mercury 15000  4 neptune 19000 |

| **Time** | **Session1** | **Output1** | **Session2** | **Output2** |
| --- | --- | --- | --- | --- |
| t1 | SET TRANSACTION NAME 'cost\_update5'; | Transaction NAME succeeded. |  |  |
| t2 | Update project set cost=1900  where pname='jupiter'; | 1 row updated. |  |  |
| t3 | Select \* from project; | 1 jupiter 1900  2 saturn 6000  3 mercury 15000  4 neptune 19000 | Select \* from project; | 1 jupiter 12300  2 saturn 6000  3 mercury 15000  4 neptune 19000 |
| t4 | Commit; | Commit complete. |  |  |
| t5 | Select \* from project; | 1 jupiter 1900  2 saturn 6000  3 mercury 15000  4 neptune 19000 | Select \* from project; | 1 jupiter 1900  2 saturn 6000  3 mercury 15000  4 neptune 19000 |

1. Compare data at time t3, t5 and t8

| **Time** | **Session1** | **Output1** | **Session2** | **Output2** |
| --- | --- | --- | --- | --- |
| t0 | SET TRANSACTION NAME 'cost\_update6'; | Transaction NAME succeeded. | SET TRANSACTION NAME 'cost\_update7'; | Transaction NAME succeeded. |
| t1 | Select \* from project; | 1 jupiter 1900  2 saturn 6000  3 mercury 15000  4 neptune 19000 | Select \* from project; | 1 jupiter 1900  2 saturn 6000  3 mercury 15000  4 neptune 19000 |
| t2 | Update project set cost=3456  where pname='mercury'; | 1 row updated. |  |  |
| t3 | Select \* from project; | 1 jupiter 1900  2 saturn 6000  3 mercury 3456  4 neptune 19000 | Select \* from project; | 1 jupiter 1900  2 saturn 6000  3 mercury 15000  4 neptune 19000 |
| t4 |  |  | Insert into project values (5, 'mars', 14500); | 1 row inserted. |
| t5 | Select \* from project; | 1 jupiter 1900  2 saturn 6000  3 mercury 3456  4 neptune 19000 | Select \* from project; | 1 jupiter 1900  2 saturn 6000  3 mercury 15000  4 neptune 19000  5 mars 14500 |
| t6 | Rollback; | Rollback complete. |  |  |
| t7 |  |  | Commit; | Commit complete. |
| t8 | Select \* from project; | 1 jupiter 1900  2 saturn 6000  3 mercury 15000  4 neptune 19000  5 mars 14500 | Select \* from project; | 1 jupiter 1900  2 saturn 6000  3 mercury 15000  4 neptune 19000  5 mars 14500 |

| **Time** | **Session1** | **Output1** | **Session2** | **Output2** |
| --- | --- | --- | --- | --- |
| t0 | SET TRANSACTION NAME 'cost\_update8'; | Transaction NAME succeeded. | SET TRANSACTION NAME 'cost\_update9'; | Transaction NAME succeeded. |
| t1 | Update project set cost=3490  where pname='mercury'; | 1 row updated. |  |  |
| t2 | Select \* from project; | 1 jupiter 1900  2 saturn 6000  3 mercury 3490  4 neptune 19000  5 mars 14500 | Select \* from project; | 1 jupiter 1900  2 saturn 6000  3 mercury 15000  4 neptune 19000  5 mars 14500 |
| t3 |  |  | Update project set cost=298  Where pname='saturn'; | 1 row updated. |
| t4 | Select \* from project; | 1 jupiter 1900  2 saturn 6000  3 mercury 3490  4 neptune 19000  5 mars 14500 | Select \* from project; | 1 jupiter 1900  2 saturn 298  3 mercury 15000  4 neptune 19000  5 mars 14500 |
| t5 | Create table test1 (id number); | Table TEST1 created. |  |  |
| t6 | Rollback; | Rollback complete. |  |  |
| t7 | Select \* from project; | 1 jupiter 1900  2 saturn 6000  3 mercury 3490  4 neptune 19000  5 mars 14500 | Select \* from project; | 1 jupiter 1900  2 saturn 298  3 mercury 3490  4 neptune 19000  5 mars 14500 |
| t8 |  |  | Commit; | Commit complete. |
| t9 | Select \* from project; | 1 jupiter 1900  2 saturn 298  3 mercury 3490  4 neptune 19000  5 mars 14500 | Select \* from project; | 1 jupiter 1900  2 saturn 298  3 mercury 3490  4 neptune 19000  5 mars 14500 |

**Transaction Processing in PL/SQL**

CREATE TABLE accounts (account\_id NUMBER(6), balance NUMBER (10,2),

check (balance>=0));

INSERT INTO accounts VALUES (7715, 6350.00);

INSERT INTO accounts VALUES (7720, 5100.50);

COMMIT;

Following is the difference circumstance of transferring money

1. Example: transfer money ($250) from account 7715 to 7720

|  |  |  |
| --- | --- | --- |
| **Time** | **Session** | **output** |
| t0 | SELECT \* FROM accounts; | 7715 6350  7720 5100.5 |
| t1 | DECLARE  transfer NUMBER(8,2) := 250;  BEGIN  UPDATE accounts SET balance = balance - transfer WHERE account\_id = 7715;  UPDATE accounts SET balance = balance + transfer WHERE account\_id = 7720;  COMMIT;  END; | PL/SQL procedure successfully completed. |
| t2 | SELECT \* FROM accounts; | 7715 6100  7720 5350.5 |

1. Example: Transfer money ($9000) from account 7715 to 7720

|  |  |  |
| --- | --- | --- |
| **Time** | **Session** | **output** |
| t0 | SELECT \* FROM accounts; | 7715 6100  7720 5350.5 |
| t1 | DECLARE  transfer NUMBER(8,2) := 9000;  BEGIN  UPDATE accounts SET balance = balance - transfer WHERE account\_id = 7715;  UPDATE accounts SET balance = balance + transfer WHERE account\_id = 7720;  COMMIT;  END; | ERROR at line 1:  ORA-02290: check constraint (C##QUANDZ.SYS\_C007946) violated  ORA-06512: at line 4 |
| t2 | SELECT \* FROM accounts; | 7715 6100  7720 5350.5 |

1. Example: Transfer money ($9000) from account 7715 to 7720

|  |  |  |
| --- | --- | --- |
| **Time** | **Session** | **output** |
| t0 | SELECT \* FROM accounts; | 7715 6100  7720 5350.5 |
| t1 | DECLARE  transfer NUMBER(8,2) := 9000;  BEGIN  UPDATE accounts SET balance = balance + transfer WHERE account\_id = 7720;  UPDATE accounts SET balance = balance - transfer WHERE account\_id = 7715;  COMMIT;  END; | ERROR at line 1:  ORA-02290: check constraint (C##QUANDZ.SYS\_C007946) violated  ORA-06512: at line 5 |
| t2 | SELECT \* FROM accounts; | 7715 6100  7720 5350.5 |

1. Example: Transfer money ($9000) from account 7715 to 7720

|  |  |  |
| --- | --- | --- |
| **Time** | **Session** | **output** |
| t0 | SELECT \* FROM accounts; | 7715 6100  7720 5350.5 |
| t1 | DECLARE  transfer NUMBER(8,2) := 9000;  BEGIN  UPDATE accounts SET balance = balance + transfer WHERE account\_id = 7720;  COMMIT;  UPDATE accounts SET balance = balance - transfer WHERE account\_id = 7715;  COMMIT;  END; | ERROR at line 1:  ORA-02290: check constraint (C##QUANDZ.SYS\_C007946) violated  ORA-06512: at line 6 |
| t2 | SELECT \* FROM accounts; | 7715 6100  7720 14350.5 |

1. Example: PL/SQL WITH EXCEPTION

|  |  |  |
| --- | --- | --- |
| **Time** | **Session** | **output** |
| t0 | SELECT \* FROM accounts; | 7715 6100  7720 14350.5 |
| t1 | SET SERVEROUTPUT ON  DECLARE  transfer NUMBER(8,2) := 9000;  BEGIN  UPDATE accounts SET balance = balance + transfer WHERE account\_id = 7720;  UPDATE accounts SET balance = balance - transfer WHERE account\_id = 7715;  COMMIT;  EXCEPTION WHEN **OTHERS** THEN  **Dbms\_output.put\_line ('error!!!!!!!!! ');**  END; | PL/SQL procedure successfully completed.  error!!!!!!!!!  PL/SQL procedure successfully completed. |
| t2 | SELECT \* FROM accounts; | 7715 6100  7720 23350.5 |

1. Example: PL/SQL WITH EXCEPTION

|  |  |  |
| --- | --- | --- |
| **Time** | **Session** | **output** |
| t0 | SELECT \* FROM accounts; | 7715 6100  7720 23350.5 |
| t1 | SET SERVEROUTPUT ON  DECLARE  transfer NUMBER(8,2) := 9000;  BEGIN  UPDATE accounts SET balance = balance + transfer WHERE account\_id = 7720;  UPDATE accounts SET balance = balance - transfer WHERE account\_id = 7715;  COMMIT;  EXCEPTION WHEN OTHERS THEN  **RAISE;**  END; | ERROR at line 1:  ORA-02290: check constraint (C##QUANDZ.SYS\_C007946) violated  ORA-06512: at line 8  ORA-06512: at line 5 |
| t2 | SELECT \* FROM accounts; | 7715 6100  7720 23350.5 |

1. Example: PL/SQL WITH EXCEPTION

|  |  |  |
| --- | --- | --- |
| **Time** | **Session** | **output** |
| t0 | SELECT \* FROM accounts; | 7715 6100  7720 23350.5 |
| t1 | SET SERVEROUTPUT ON  DECLARE  transfer NUMBER(8,2) := 9000;  BEGIN  UPDATE accounts SET balance = balance + transfer WHERE account\_id = 7720;  COMMIT;  UPDATE accounts SET balance = balance - transfer WHERE account\_id = 7715;    EXCEPTION WHEN OTHERS THEN  **RAISE;**  END; | ORA-02290: check constraint (C##QUANDZ.SYS\_C007946) violated  ORA-06512: at line 9  ORA-06512: at line 6 |
| t2 | SELECT \* FROM accounts; | 7715 6100  7720 32350.5 |

***2. Điều khiển đồng thời***

Connect to sys:

Sqlplus / as SYSDBA

Create a pluggable database:

CREATE PLUGGABLE DATABASE WEEK5PDB ADMIN USER lab IDENTIFIED BY userpass ROLES=(DBA) FILE\_NAME\_CONVERT=('PDBSEED','WEEK5PDB');

ALTER PLUGGABLE DATABASE WEEK5PDB OPEN;

--connect to system of week5pdb

CONNECT system/password@localhost/WEEK5PDB

CREATE USER week5 identified by userpass;

GRANT DBA TO week5;

Using sqldeveloper to connect to week5:

From Sqldeveloper, open two **independent** session (Ctrl + Shift + N), called session A and session B.

1. **Locks, blocks and deadlocks**
2. Blocked Inserts

|  |  |  |
| --- | --- | --- |
| **T** | **Session A** | **Session B** |
|  | CREATE TABLE TEST (ID NUMBER **PRIMARY KEY**, NAME VARCHAR2(50), NOTE VARCHAR2(1000));  Table TEST created. |  |
|  | SELECT \* FROM TEST;  NO ROW SELECTED. | SELECT \* FROM TEST;  NO ROW SELECTED. |
|  | INSERT INTO TEST VALUES (1, 'HELLO', NULL);  1 row inserted. |  |
|  |  | INSERT INTO TEST VALUES (1, 'GOODBYE', NULL);  Câu truy vấn đợi mãi |
|  | --See what session is blocking other session?  Có. | |
|  | COMMIT; | --See what’s happening in session B  Error starting at line : 4 in command -  INSERT INTO TEST VALUES (1, 'GOODBYE', NULL)  Error report - |
|  | SELECT \* FROM TEST;  1 HELLO NULL | SELECT \* FROM TEST;  1 HELLO NULL |

|  |  |  |
| --- | --- | --- |
| **T** | **Session A** | **Session B** |
|  | INSERT INTO TEST VALUES (2, 'HELLO SESSION A', NULL);  1 row inserted. |  |
|  |  | INSERT INTO TEST VALUES (2, 'HELLO SESSION B', NULL); |
|  | --See what session is blocking other session?  Có | |
|  | ROLLBACK;  Rollback complete. | --See what’s happening in session B  1 row inserted. |
|  |  | COMMIT;  Commit complete. |
|  | SELECT \* FROM TEST;  1 HELLO  2 HELLO SESSION B | SELECT \* FROM TEST;  1 HELLO  2 HELLO SESSION B |

1. Blocked Updates, Deletes

|  |  |  |
| --- | --- | --- |
| **T** | **Session A** | **Session B** |
|  | INSERT INTO TEST VALUES (3, 'JOHN', NULL);  COMMIT;  1 row inserted.  Commit complete. |  |
|  | UPDATE TEST SET NOTE='UPDATED BY SESSION A' WHERE ID=3;  1 row updated. |  |
|  |  | UPDATE TEST SET NOTE='UPDATED BY SESSION B' WHERE ID=3;  BLOCKING |
|  | --See what session is blocking other session? | |
|  | COMMIT;  Commit complete. | --See what’s happening in session B  1 row updated. |
|  |  | COMMIT;  Commit complete. |
|  | SELECT \* FROM TEST;  1 HELLO  3 JOHN UPDATED BY SESSION B  2 HELLO SESSION B | SELECT \* FROM TEST;  1 HELLO  3 JOHN UPDATED BY SESSION B  2 HELLO SESSION B |

|  |  |  |
| --- | --- | --- |
| **T** | **SESSION A** | **SESSION B** |
|  | INSERT INTO TEST VALUES (4, 'SAMSUNG', NULL);  COMMIT;  1 row inserted.  Commit complete. |  |
|  | SELECT \* FROM TEST;  1 HELLO  3 JOHN UPDATED BY SESSION B  2 HELLO SESSION B  4 SAMSUNG | SELECT \* FROM TEST;  1 HELLO  3 JOHN UPDATED BY SESSION B  2 HELLO SESSION B  4 SAMSUNG |
|  | UPDATE TEST SET NAME='APPLE' WHERE ID=4;  1 row updated. |  |
|  |  | DELETE FROM TEST WHERE ID=4;  blocking |
|  | --See what session is blocking other session? | |
|  | COMMIT;  Commit complete. | --See what’s happening in session B  1 row deleted. |
|  |  | COMMIT;  Commit complete. |
|  | SELECT \* FROM TEST;  1 HELLO  3 JOHN UPDATED BY SESSION B  2 HELLO SESSION B | SELECT \* FROM TEST;  1 HELLO  3 JOHN UPDATED BY SESSION B  2 HELLO SESSION B |

|  |  |  |
| --- | --- | --- |
| **T** | **SESSION A** | **SESSION B** |
|  | INSERT INTO TEST VALUES (5, 'TIKI', NULL);  COMMIT;  1 row inserted.  Commit complete. |  |
|  | SELECT \* FROM TEST;  1 HELLO  3 JOHN UPDATED BY SESSION B  2 HELLO SESSION B  5 TIKI | SELECT \* FROM TEST;  1 HELLO  3 JOHN UPDATED BY SESSION B  2 HELLO SESSION B  5 TIKI |
|  | DELETE FROM TEST WHERE ID=5;  1 row deleted. |  |
|  |  | UPDATE TEST SET NAME='LAZADA' WHERE ID=5;  blocking |
|  | --See what session is blocking other session? | |
|  | ROLLBACK;  Rollback complete. | --See what’s happening in session B  1 row updated. |
|  |  | COMMIT;  Commit complete. |
|  | SELECT \* FROM TEST;  1 HELLO  3 JOHN UPDATED BY SESSION B  2 HELLO SESSION B  5 LAZADA | SELECT \* FROM TEST;  1 HELLO  3 JOHN UPDATED BY SESSION B  2 HELLO SESSION B  5 LAZADA |

1. **SELECT FOR UPDATE**

|  |  |  |
| --- | --- | --- |
| **T** | **SESSION A** | **SESSION B** |
|  | DELETE FROM TEST;  INSERT INTO TEST VALUES (6, 'FAHASA', NULL);  COMMIT;  4 rows deleted.  1 row inserted.  Commit complete. |  |
|  | SELECT \* FROM TEST;  6 FAHASA | SELECT \* FROM TEST;  6 FAHASA |
|  | UPDATE TEST SET NOTE='HELLO FAHASA' WHERE ID=6;  1 row updated. |  |
|  |  | SELECT \* FROM TEST WHERE ID=6 **FOR UPDATE**; |
|  |  | --See what’s happening in session B.  blocking |
|  | COMMIT;  Commit complete. |  |
|  |  | --See what’s happening in session B.  6 FAHASA HELLO FAHASA |
|  | UPDATE TEST SET NOTE='UPDATED BY SESSION A' WHERE ID=6; blocking |  |
|  |  | COMMIT;  Commit complete. |
|  | --See what’s happening in session A.  1 row updated. |  |
|  | COMMIT;  Commit complete. |  |

1. **SELECT FOR UPDATE NOWAIT**

|  |  |  |
| --- | --- | --- |
| **T** | **SESSION A** | **SESSION B** |
|  | INSERT INTO TEST VALUES (7, 'LENOVO', NULL);  COMMIT;  1 row inserted.  Commit complete. |  |
|  | UPDATE TEST SET NOTE='HELLO LENOVO' WHERE ID=7;  1 row updated. |  |
|  |  | SELECT \* FROM TEST WHERE ID=7 FOR UPDATE NOWAIT;  ORA-00054: resource busy and acquire with NOWAIT specified or timeout expired |
|  |  | --See what’s happening in session B. |
|  | COMMIT;  Commit complete. | 7 LENOVO HELLO LENOVO |

1. TX Lock, TM Locks

|  |  |  |
| --- | --- | --- |
| **T** | **SESSION A** | **SESSION B** |
|  | INSERT INTO TEST VALUES (8, 'FORD', NULL);  COMMIT;  Commit complete. |  |
|  | SELECT \* FROM TEST;  7 LENOVO HELLO LENOVO  8 FORD  6 FAHASA UPDATED BY SESSION A | SELECT \* FROM TEST;  7 LENOVO HELLO LENOVO  8 FORD  6 FAHASA UPDATED BY SESSION A |
|  | UPDATE TEST SET NOTE='HELLO FORD' WHERE ID=8;  1 row updated.  --(WHICH LOCK HERE: TX? TM LOCK?) |  |
|  |  | ALTER TABLE TEST MODIFY NOTE VARCHAR2(200);  ALTER TABLE TEST MODIFY NOTE VARCHAR2(200)  Error report -  ORA-00054: resource busy and acquire with NOWAIT specified or timeout expired  --See what’s happening in session B. |
|  | COMMIT;  Commit complete. | --See what’s happening in session B. |
|  | UPDATE TEST SET NOTE='HELLO FORD' WHERE ID=8;  1 row updated. |  |
|  |  | DELETE FROM TEST WHERE ID=8;  --See what’s happening in session B. |
|  | COMMIT;  Commit complete. | --See what’s happening in session B.  Notthing |
|  |  | COMMIT; |

1. Deadlook

|  |  |  |
| --- | --- | --- |
| **T** | **SESSION A** | **SESSION B** |
|  | DELETE FROM TEST;  INSERT INTO TEST VALUES (90, 'RESOURCE 1', NULL);  INSERT INTO TEST VALUES (91, 'RESOURCE 2', NULL);  COMMIT;  2 rows deleted.  1 row inserted.  1 row inserted.  Commit complete. |  |
|  | SELECT \* FROM TEST;  90 RESOURCE 1  91 RESOURCE 2 | SELECT \* FROM TEST;  90 RESOURCE 1  91 RESOURCE 2 |
|  | UPDATE TEST SET NOTE='HELLO RESOURCE 1' WHERE ID=90;  1 row updated. | UPDATE TEST SET NOTE='HELLO RESOURCE 2' WHERE ID=91;  1 row updated. |
|  | DELETE FROM TEST WHERE ID=91;  blocking | DELETE FROM TEST WHERE ID=90;  blocking |
|  | --SEE WHAT’S HAPPENING?  Error report -  SQL Error: ORA-00060: deadlock detected while waiting for resource | |

**[Isolation levels](https://stackoverflow.com/questions/13647604/isolation-levels-in-oracle)**

DROP TABLE accounts;

CREATE TABLE accounts (accid NUMBER(6) primary key,

balance NUMBER (10,2),

check (balance>=0));

INSERT INTO accounts VALUES (7715, 7000);

INSERT INTO accounts VALUES (7720, 5100);

COMMIT;

Tạo hai transaction, chạy thử theo các tình huống sau và ghi kết quả.

1. Read committed: **Non-repeatable Read**

|  |  |
| --- | --- |
| Accid | balance |
| 7715 | 7000 |
| 7720 | 5100 |

|  |  |  |
| --- | --- | --- |
| **T** | **Session 1** | **Session 2** |
|  | select \* from accounts;  7715 7000  7720 5100 | select \* from accounts;  7715 7000  7720 5100 |
|  | update accounts  set balance=balance-2000  where accid=7715;  1 row updated. |  |
|  |  | select \* from accounts;  7715 7000  7720 5100 |
|  | Commit;  Commit complete. |  |
|  | select \* from accounts;  7715 5000  7720 5100 | select \* from accounts;  7715 5000  7720 5100 |

1. Read committed: **Phantom Read**

|  |  |  |
| --- | --- | --- |
| **T** | **Session 1** | **Session 2** |
|  | select \* from accounts  where balance>100;  7715 5000  7720 5100 | select \* from accounts  where balance>100;  7715 5000  7720 5100 |
|  | INSERT INTO accounts VALUES (7740, 3000);  Commit;  1 row inserted.  Commit complete. |  |
|  |  | select \* from accounts  where balance>100;  7715 5000  7720 5100  7740 3000 |

Run the Account script:

DELETE FROM ACCOUNTS;

INSERT INTO accounts VALUES (7715, 7000);

INSERT INTO accounts VALUES (7720, 5100);

COMMIT;

|  |  |
| --- | --- |
| Accid | balance |
| 7715 | 7000 |
| 7720 | 5100 |

1. Serializable Isolation Level: **repeatable Read**

|  |  |  |
| --- | --- | --- |
| **T** | **Session 1** | **Session 2** |
|  | COMMIT;  select \* from accounts;  7715 7000  7720 5100 | COMMIT;  select \* from accounts;  7715 7000  7720 5100 |
|  | update accounts set balance=8000  where accid=7720;  1 row updated. | SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;  select \* from accounts;  7715 7000  7720 5100 |
|  | COMMIT;  Commit complete. |  |
|  |  | select \* from accounts;  7715 7000  7720 5100 |
|  |  | update accounts set balance=40  where accid=7720;  Error report -  SQL Error: ORA-08177: can't serialize access for this transaction |
|  | select \* from accounts  7715 7000  7720 8000 | select \* from accounts  7715 7000  7720 5100 |
|  |  | ROLLBACK;  Rollback complete. |

1. Serializable Isolation Level**: non-Phantom Read**

|  |  |  |
| --- | --- | --- |
| **T** | **Session 1** | **Session 2** |
|  | select \* from accounts;  7715 7000  7720 8000 |  |
|  | INSERT INTO accounts VALUES (7750, 3000);  1 row inserted. |  |
|  |  | SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;  select \* from accounts;  7715 7000  7720 8000 |
|  |  |  |
|  | COMMIT;  Commit complete. |  |
|  | select \* from accounts;  7750 3000  7715 7000  7720 8000 | select \* from accounts;  7715 7000  7720 8000 |
|  |  | update accounts set balance=1111 where accid=**7750**;  **0 rows updated.** |
|  |  | INSERT INTO accounts VALUES (**7750**, 8000);  INSERT INTO accounts VALUES (7750, 8000)  Error report -  ORA-00001: unique constraint (WEEK5.SYS\_C007432) violated |
|  |  | COMMIT;  Commit complete. |

**Các vấn đề trong xử lý đồng thời**

DROP TABLE accounts;

CREATE TABLE accounts (accid NUMBER(6) primary key,

balance NUMBER (10,2),

owner\_name varchar2(30),

check (balance>=0));

INSERT INTO accounts VALUES (7715, 90, 'Scott');

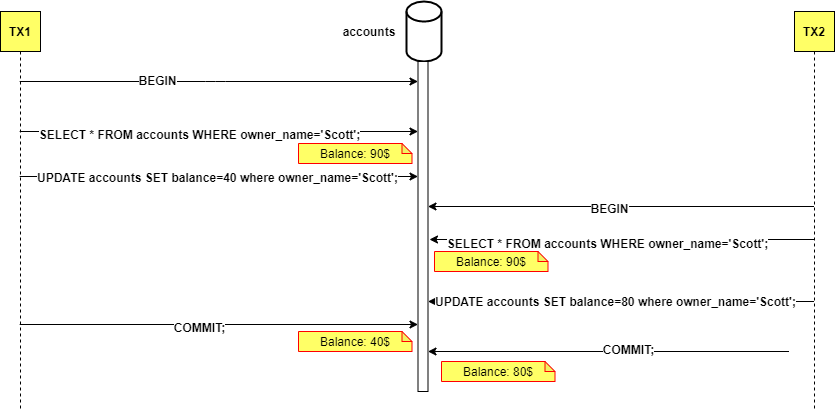
INSERT INTO accounts VALUES (7720, 5100, 'Tiger');

INSERT INTO accounts VALUES (7725, 20, 'Helen');

INSERT INTO accounts VALUES (7730, 49, 'John');

COMMIT;

1. **Lost update**



Viết hai transaction rút tiền thể hiện tình trạng lost update theo như sơ đồ trên. Nêu các cách tránh lost update.

### **Increase transaction isolation level**

### **Pessimistic Locking**

### **Optimistic Locking**

***Giải***

* Hai giao dịch rút tiền thể hiện tình trạng Lost Update theo sơ đồ:

**TX1:**

BEGIN;

SELECT \* FROM accounts WHERE owner\_name='Scott'; (Balance: 90$)

UPDATE accounts SET balance=40 WHERE owner\_name='Scott';

COMMIT; (Balance: 40$)

**TX2:**

BEGIN;

SELECT \* FROM accounts WHERE owner\_name='Scott'; (Balance: 90$) - Đọc giá trị cũ trước khi TX1 commit

UPDATE accounts SET balance=80 WHERE owner\_name='Scott';

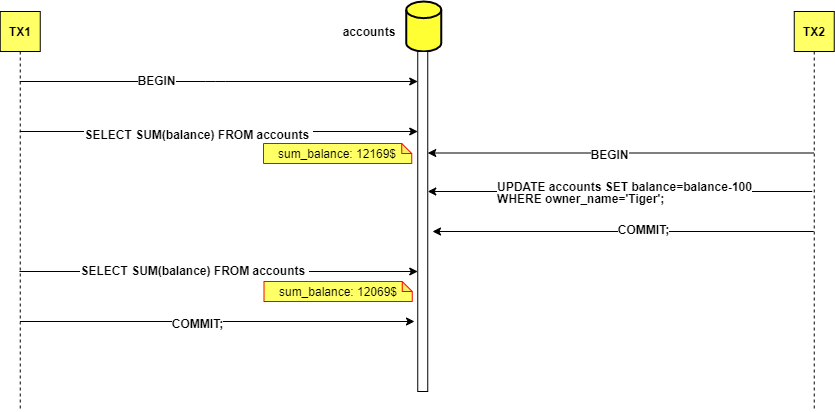
COMMIT; (Balance: 80$) - Ghi đè lên kết quả của TX1

**Kết quả Lost Update:** Sau cả hai giao dịch, số dư cuối cùng là 80$, trong khi lẽ ra phải là 30$ (90−50 (từ TX1) −10 (từ TX2)).

**Các cách tránh Lost Update:**

* 1. Increase transaction isolation level:
* Sử dụng mức độ cô lập từ REPEATABLE READ trở lên (ví dụ: SERIALIZABLE).
  1. Pessimistic Locking:
* Sử dụng SELECT ... FOR UPDATE (hoặc cú pháp tương tự tùy theo DBMS) để khóa hàng dữ liệu ngay khi đọc, ngăn chặn các giao dịch khác sửa đổi.
* Ví dụ: SELECT \* FROM accounts WHERE owner\_name='Scott' FOR UPDATE;
  1. Optimistic Locking:
* Thêm một trường version hoặc timestamp vào bảng.
* Khi đọc, lưu lại giá trị version.
* Khi cập nhật, kiểm tra version trong WHERE clause và tăng version lên. Nếu version đã thay đổi, thao tác sẽ thất bại và cần thử lại.
* Ví dụ: UPDATE accounts SET balance=..., version=version+1 WHERE owner\_name='Scott' AND version=<old\_version\_read>;

1. **Non-repeatable read**



Viết hai transaction, một transaction thể hiện việc truy vấn dữ liệu để viết báo cáo, transaction còn lại thực hiện việc rút tiền.

Nêu phương pháp để giải quyết non-repeatable read.

**Giải**

• Hai Transaction:

Transaction báo cáo (TX1):

BEGIN;

SELECT SUM(balance) FROM accounts; (Kết quả: 121695)

SELECT SUM(balance) FROM accounts; (Kết quả: 120695 - Giá trị khác do TX2 đã commit)

COMMIT;

Transaction rút tiền (TX2):

BEGIN;

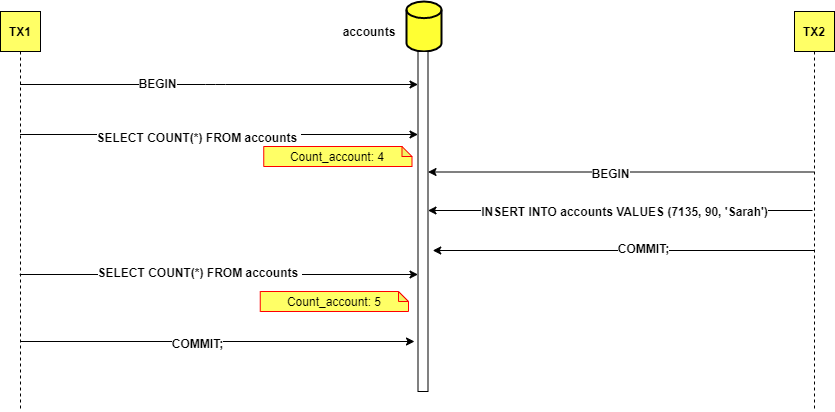
UPDATE accounts SET balance=balance-100 WHERE owner\_name='Tiger';

COMMIT;

• Mô tả Non-Repeatable Read: TX1 đọc SUM(balance) hai lần trong cùng một giao dịch nhưng nhận được kết quả khác nhau (121695 rồi 120695). Điều này xảy ra vì TX2 đã thay đổi dữ liệu và commit giữa hai lần đọc của TX1.

• Phương pháp giải quyết Non-Repeatable Read: tăng mức độ cô lập của giao dịch (Transaction Isolation Level) lên REPEATABLE READ hoặc SERIALIZABLE.

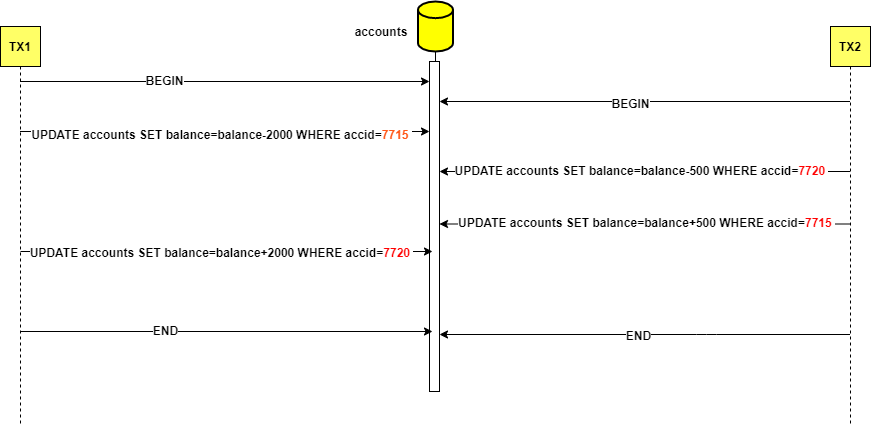
1. **Phantom read**



Viết hai transaction, một transaction thể hiện việc truy vấn dữ liệu để viết báo cáo, transaction còn lại thực hiện việc thêm tài khoản vào DB.

Nêu phương pháp để giải quyết phantom read.

**Deadlock**



Viết hai transaction thể hiện việc chuyển tiền theo sơ đồ như trên. (Gây nên tình trạng deadlock)

Nêu phương pháp giải quyết deadlock.

***Giải***

* ***Hai Transaction:***
* **Transaction báo cáo (TX1):**

BEGIN;

SELECT COUNT(\*) FROM accounts; (Kết quả: 4)

SELECT COUNT(\*) FROM accounts; (Kết quả: 5 - Có thêm hàng mới)

COMMIT;

* **Transaction thêm tài khoản (TX2):**

BEGIN;

INSERT INTO accounts VALUES (7135, 90, 'Sarah');

COMMIT;

* **Mô tả Phantom Read:** TX1 thực hiện SELECT COUNT(\*) hai lần trong cùng giao dịch nhưng nhận kết quả khác nhau (4 rồi 5) do TX2 đã thêm một hàng mới và commit giữa hai lần đọc.
* **Phương pháp giải quyết Phantom Read:** tăng mức độ cô lập của giao dịch lên SERIALIZABLE.