Connect to sys:

Sqlplus / as SYSDBA

Create a pluggable database:

CREATE PLUGGABLE DATABASE WEEK6PDB ADMIN USER lab IDENTIFIED BY userpass ROLES=(DBA) FILE\_NAME\_CONVERT=('PDBSEED','WEEK6PDB');

ALTER PLUGGABLE DATABASE WEEK6PDB OPEN;

--connect to system of week6pdb

CONNECT system/password@localhost/WEEK6PDB

CREATE USER week6 identified by userpass;

GRANT DBA TO week6;

Using sqldeveloper to connect to week6:

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)); |  |
|  | SELECT \* FROM TEST;  no rows selected | SELECT \* FROM TEST;  no rows selected |
|  | INSERT INTO TEST VALUES (1, 'HELLO', NULL);  1 row inserted |  |
|  |  | INSERT INTO TEST VALUES (1, 'GOODBYE', NULL); |
|  | --See what session is blocking other session?  Session B is blocked by session A | |
|  | COMMIT;  Commit complete | --See what’s happening in session B  Error report -  ORA-00001: unique constraint (C##GIAKIET.SYS\_C008032) violated |
|  | SELECT \* FROM TEST;  1 HELLO | SELECT \* FROM TEST;  1 HELLO |

|  |  |  |
| --- | --- | --- |
| **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?  Session B is blocked by session A | |
|  | ROLLBACK;  Rollback complete. | --See what’s happening in session B  Error report -  ORA-00001: unique constraint (C##GIAKIET.SYS\_C008032) violated |
|  |  | COMMIT;  Commit complete. |
|  | SELECT \* FROM TEST;  2 HELLO SESSION B  1 HELLO | SELECT \* FROM TEST;  2 HELLO SESSION B  1 HELLO |

1. Blocked Updates, Deletes

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

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

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

1. **SELECT FOR UPDATE**

|  |  |  |
| --- | --- | --- |
| **T** | **SESSION A** | **SESSION B** |
|  | DELETE FROM TEST;  INSERT INTO TEST VALUES (6, 'FAHASA', NULL);  COMMIT; |  |
|  | 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.  Session B is blocked |
|  | COMMIT;  Commit complete. |  |
|  |  | --See what’s happening in session B.  ID NAME NOTE  6 FAHASA  HELLO FAHASA |
|  | UPDATE TEST SET NOTE='UPDATED BY SESSION A' WHERE ID=6; |  |
|  |  | 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; |  |
|  | UPDATE TEST SET NOTE='HELLO LENOVO' WHERE ID=7;  1 row updated. |  |
|  |  | SELECT \* FROM TEST WHERE ID=7 FOR UPDATE NOWAIT; |
|  |  | --See what’s happening in session B.  ORA-00054: resource busy and acquire with NOWAIT specified or timeout expired  00054. 00000 - "resource busy and acquire with NOWAIT specified or timeout expired"  \*Cause: Interested resource is busy.  \*Action: Retry if necessary or increase timeout. |
|  | COMMIT;  Commit complete. |  |

1. TX Lock, TM Locks

|  |  |  |
| --- | --- | --- |
| **T** | **SESSION A** | **SESSION B** |
|  | INSERT INTO TEST VALUES (8, 'FORD', NULL);  COMMIT; |  |
|  | 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?) ca hai |  |
|  |  | ALTER TABLE TEST MODIFY NOTE VARCHAR2(200);  --See what’s happening in session B.  ALTER TABLE TEST MODIFY NOTE VARCHAR2(200)  Error report -  ORA-00054: resource busy and acquire with NOWAIT specified or timeout expired  00054. 00000 - "resource busy and acquire with NOWAIT specified or timeout expired"  \*Cause: Interested resource is busy.  \*Action: Retry if necessary or increase timeout. |
|  | 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.  Session B is blocked |
|  | COMMIT;  Commit complete. | --See what’s happening in session B.  1 row deleted. |
|  |  | COMMIT;  Commit complete. |

1. DEADLOCK

|  |  |  |
| --- | --- | --- |
| **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;  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;  DELETE FROM TEST WHERE ID=91  Error report -  ORA-00060: deadlock detected while waiting for resource | DELETE FROM TEST WHERE ID=90;  1 row deleted. |
|  | ROLLBACK;  Rollback complete. |  |
|  | --SEE WHAT’S HAPPENING? | |

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

|  |  |  |  |
| --- | --- | --- | --- |
| **Isolation Level** | **Dirty Read** | **Non-repeatable Read** | **Phantom Read** |
| Read uncommitted | Possible | Possible | Possible |
| Read committed | Not possible | Possible | Possible |
| Repeatable read | Not possible | Not possible | Possible |
| Serializable | Not possible | Not possible | Not possible |

Account Script:

|  |
| --- |
| 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, nhận xét

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

Run the Account script

|  |  |
| --- | --- |
| 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 |

Rerun the Account script

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

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

|  |  |  |
| --- | --- | --- |
| **T** | **Session 1** | **Session 2** |
|  | COMMIT;  select \* from accounts;  7715 5000  7720 5100  7740 3000 | COMMIT;  select \* from accounts;  7715 5000  7720 5100  7740 3000 |
|  | update accounts set balance=8000  where accid=7720;  1 row updated. | SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;  select \* from accounts;  Transaction ISOLATION succeeded.  7715 5000  7720 5100  7740 3000 |
|  | COMMIT;  Commit complete. |  |
|  |  | select \* from accounts;  7715 5000  7720 5100  7740 3000 |
|  |  | update accounts set balance=balance+40  where accid=7720;  Error report -  ORA-08177: can't serialize access for this transaction |
|  | select \* from accounts  7715 5000  7720 8000  7740 3000 | select \* from accounts  7715 5000  7720 5100  7740 3000 |
|  |  | ROLLBACK;  Rollback complete. |

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

|  |  |  |
| --- | --- | --- |
| **T** | **Session 1** | **Session 2** |
|  | select \* from accounts;  7715 5000  7720 8000  7740 3000 |  |
|  | INSERT INTO accounts VALUES (7750, 3000);  1 row inserted. |  |
|  |  | SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;  select \* from accounts;  Transaction ISOLATION succeeded.  7715 5000  7720 8000  7740 3000 |
|  |  |  |
|  | COMMIT;  Commit complete. |  |
|  | select \* from accounts;  7715 5000  7720 8000  7740 3000  7750 3000 | select \* from accounts;  7715 5000  7720 8000  7740 3000 |
|  |  | update accounts set balance=1111 where accid=**7750**;  **0 rows updated.** |
|  |  | INSERT INTO accounts VALUES (**7750**, 8000);  Error report -  ORA-00001: unique constraint (C##GIAKIET.SYS\_C008034) violated |
|  |  | COMMIT;  Commit complete. |

1. **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');

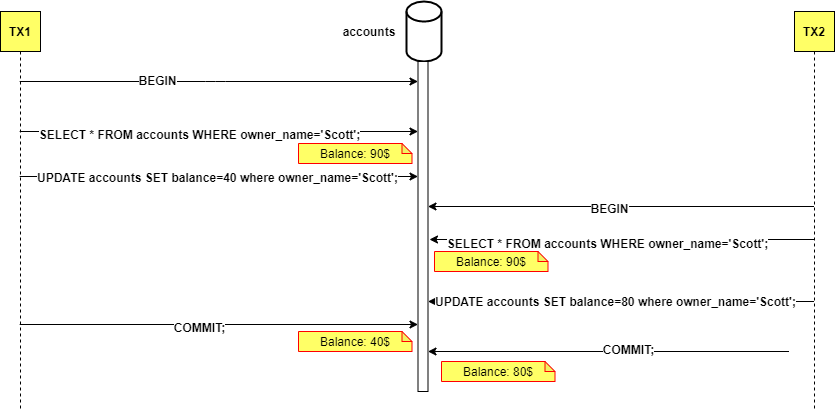
INSERT INTO accounts VALUES (7720, 5100, 'Tiger');

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

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

COMMIT;

1. **Lost update**



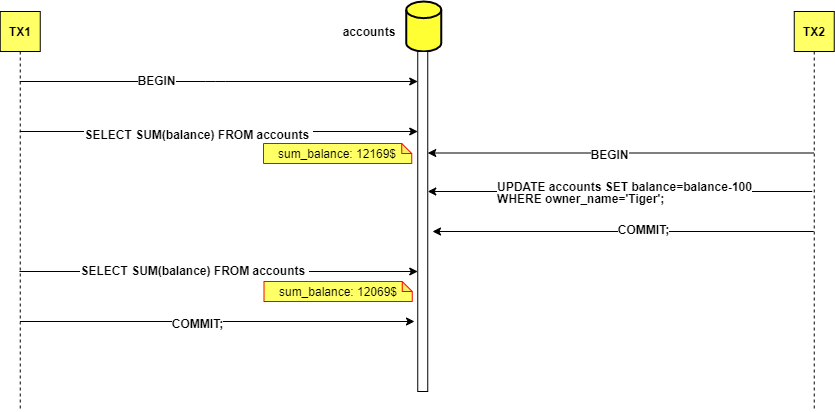
Viết hai transaction rút tiền thể hiện tình trạng lost update theo như sơ đồ trên. Nêu các cách tránh lost update.

### **Increase transaction isolation level**

### **Pessimistic Locking**

### **Optimistic Locking**

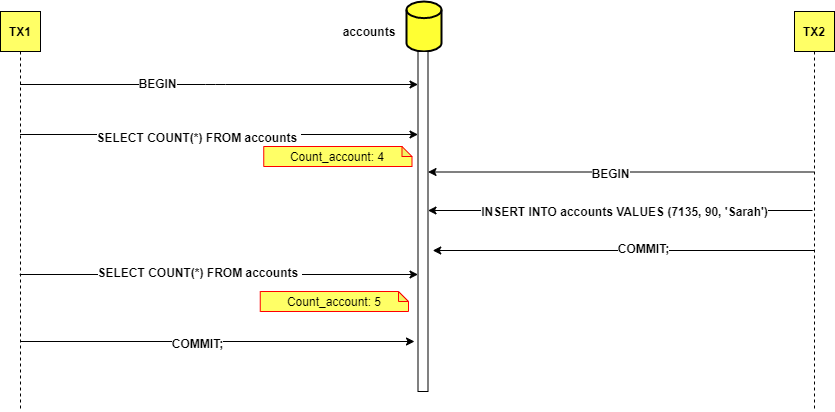
1. **Non-repeatable read**



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.

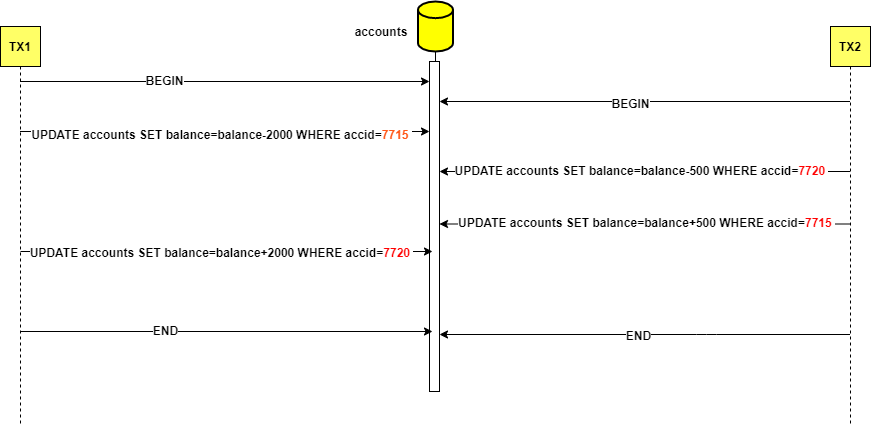
1. **Phantom read**



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.