Logo, company name

Description automatically generated

School of Computer Science

Master of Applied computing (MAC)

Advanced Database Topic – COMP 8157 Section 4

**Lab 4: Concurrency Control**

Prepared by

Rohit Kumar (110088741)

Table of Contents

[Database: 3](#_Toc140000516)

[BankAccount table: 3](#_Toc140000517)

[Insert initial account balance: 4](#_Toc140000518)

[Methods for database: 5](#_Toc140000519)

[Withdrawal: 8](#_Toc140000520)

# Database:

## BankAccount table:

-- Create table

CREATE TABLE BankAccount

(

AccountNumber INT PRIMARY KEY,

Balance DECIMAL(10, 2)

);

A screenshot of a computer

Description automatically generated

## Insert initial account balance:

-- Set the initial account balance to $1000

-- Insert data

INSERT INTO BankAccount

(AccountNumber, Balance)

VALUES

(110088741,1000);

-- check balance

SELECT \* FROM BankAccount;

A screenshot of a computer

Description automatically generated

# Methods for database:

To simulate the concurrent withdrawals from account, I have executed two transactions and one of the transactions with wait for 30 sec. Before the transaction 1 finishes transaction 2 completed and it withdraw amount 50 and updated the account balance as 950. However, transaction1 took the initial amount 1000 and withdraw amount 150 and update the account balance as 850 which is wrong, since transaction 2 completed first therefore final amount should be 800 instead of 850. This is because Transaction 1 silently overwrites the update which is made by Transaction 2 which is causing Lost Update concurrency problem.

-- Winthdraw money transaction 1

BEGIN TRANSACTION

DECLARE @availablebalance1 INT

DECLARE @withdrawamount1 AS INT=150;

SELECT @availablebalance1 = Balance FROM BankAccount WHERE AccountNumber = 110088741

-- Transaction takes 30 seconds

WAITFOR DELAY '00:00:30'

IF(@availablebalance1-@withdrawamount1 >0)

UPDATE BankAccount SET Balance = @availablebalance1-@withdrawamount1 WHERE AccountNumber = 110088741;

ELSE

PRINT 'INSUFFICENT BALANCE';

COMMIT TRANSACTION

A screenshot of a computer

Description automatically generated

Transaction 1 trigger and it is still in execution mode.

-- Winthdraw money transaction 2

BEGIN TRANSACTION

DECLARE @availablebalance2 INT

DECLARE @withdrawamount2 AS INT=50;

SELECT @availablebalance2 = Balance FROM BankAccount WHERE AccountNumber = 110088741

IF(@availablebalance2-@withdrawamount2 >0)

UPDATE BankAccount SET Balance = @availablebalance2-@withdrawamount2 WHERE AccountNumber = 110088741;

ELSE

PRINT 'INSUFFICENT BALANCE';

COMMIT TRANSACTION

A screenshot of a computer

Description automatically generated

In between Transaction 2 triggered and update the balance amount as 950

A screenshot of a computer

Description automatically generated

Finally, transaction 1 completed and which updated the balance amount as 850

I have used isolation level repeatable read, as it would block the second transaction and let the first one finish first, in this way there will be no conflict, or lost updates, as one transaction completes first. Therefore, correct balance amount is reflected at the end which is 800.

## Withdrawal:

-- Winthdraw money transaction 1

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ

BEGIN TRANSACTION

DECLARE @availablebalance1 INT

DECLARE @withdrawamount1 AS INT=150;

SELECT @availablebalance1 = Balance FROM BankAccount WHERE AccountNumber = 110088741

-- Transaction takes 30 seconds

WAITFOR DELAY '00:00:30'

IF(@availablebalance1-@withdrawamount1 >0)

UPDATE BankAccount SET Balance = @availablebalance1-@withdrawamount1 WHERE AccountNumber = 110088741;

ELSE

PRINT 'INSUFFICENT BALANCE';

COMMIT TRANSACTION

A screenshot of a computer

Description automatically generated

-- Winthdraw money transaction 2

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ

BEGIN TRANSACTION

DECLARE @availablebalance2 INT

DECLARE @withdrawamount2 AS INT=50;

SELECT @availablebalance2 = Balance FROM BankAccount WHERE AccountNumber = 110088741

IF(@availablebalance2-@withdrawamount2 >0)

UPDATE BankAccount SET Balance = @availablebalance2-@withdrawamount2 WHERE AccountNumber = 110088741;

ELSE

PRINT 'INSUFFICENT BALANCE';

COMMIT TRANSACTION

A screenshot of a computer

Description automatically generated