

CSCI 5408

DATA MANAGEMENT AND WAREHOUSING



LAB ASSIGNMENT - 3

Submitted By: Kenil Shaileshkumar Patel
(kenil.patel@dal.ca)
Banner ID: B00954251
Submitted On: October 7, 2023

Gitlab Repository Link

https://git.cs.dal.ca/kenil/csci5408_f23_b00954251_kenil_patel/-/tree/main/Lab3

Table of Contents

Sr. No	Title	Page No.
1.	Creating Database for the Banking System	3
2.	Transaction	5
3.	References	7

Creating Database for Banking System

Creating a schema named BankDatabase with the following SQL query:

```
CREATE SCHEMA IF NOT EXISTS BankDatabase;  
USE BankDatabase
```

Further creating the tables for the database using the SQL query:

The first table is CustomerDetail:

```
CREATE TABLE IF NOT EXISTS CustomerDetail (  
    CustomerID INT PRIMARY KEY AUTO_INCREMENT,  
    Name VARCHAR(100) NOT NULL,  
    MailAddress VARCHAR(200) NOT NULL,  
    PermanentAddress VARCHAR(200) NOT NULL,  
    Email VARCHAR(100) UNIQUE NOT NULL,  
    PhoneNumber VARCHAR(20) UNIQUE NOT NULL  
);
```

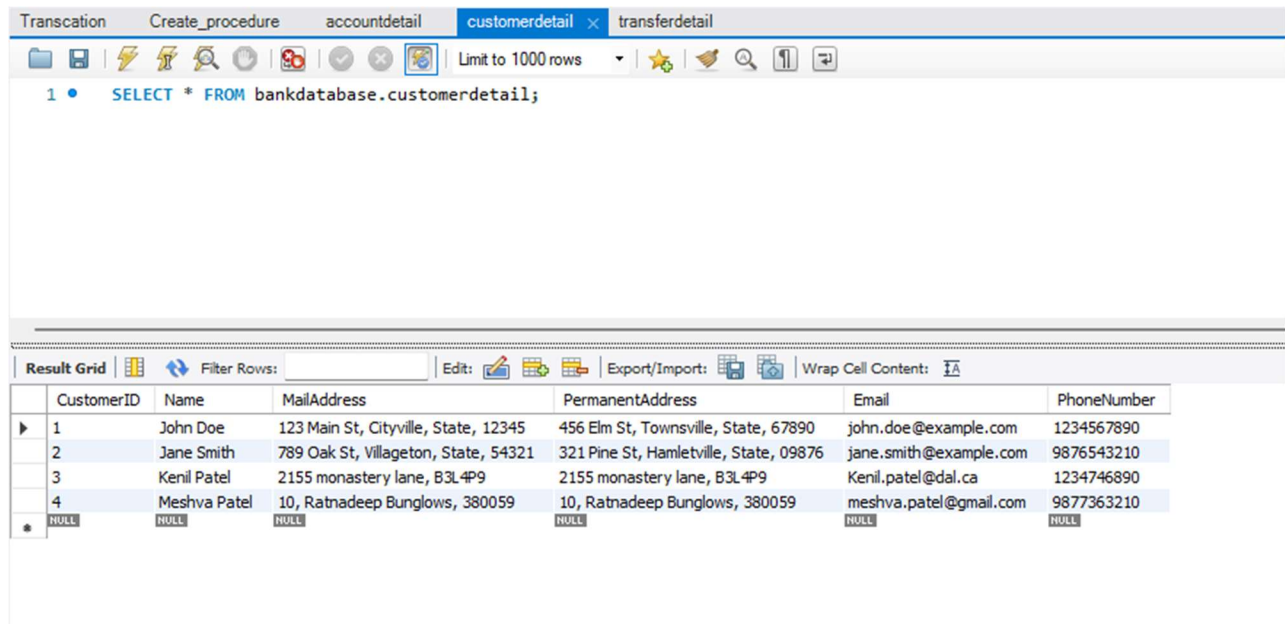
The second table is named as AccountDetail:

```
CREATE TABLE IF NOT EXISTS AccountDetail (  
    AccNumber INT PRIMARY KEY AUTO_INCREMENT,  
    AccBalance DECIMAL(15, 2) NOT NULL,  
    CustomerID INT,  
    FOREIGN KEY (CustomerID) REFERENCES CustomerDetail(CustomerID)  
);
```

Third table is TransferDetail:

```
CREATE TABLE IF NOT EXISTS TransferDetail (  
    TransferID INT PRIMARY KEY AUTO_INCREMENT,  
    DateOfTransfer DATE NOT NULL,  
    RecipientName VARCHAR(100) NOT NULL,  
    Status VARCHAR(20),  
    AccNumber INT NOT NULL,  
    FOREIGN KEY (AccNumber) REFERENCES AccountDetail(AccNumber)  
);
```

Output:



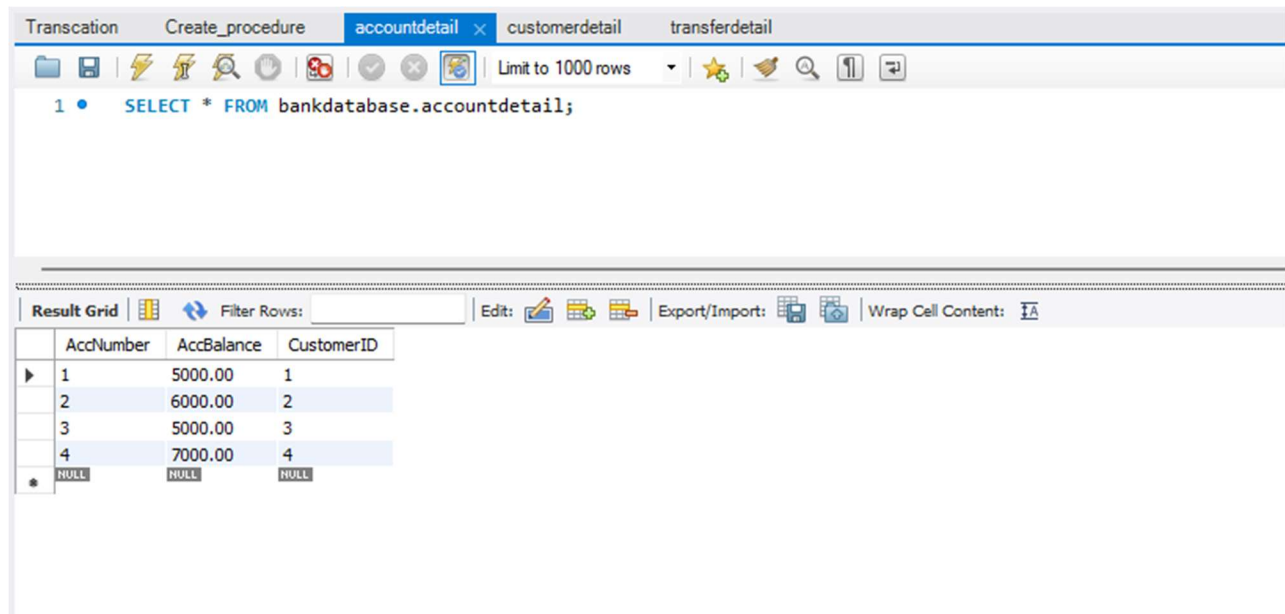
Transcation Create_procedure accountdetail **customerdetail** transferdetail

Limit to 1000 rows

1 • `SELECT * FROM bankdatabase.customerdetail;`

	CustomerID	Name	MailAddress	PermanentAddress	Email	PhoneNumber
▶	1	John Doe	123 Main St, Cityville, State, 12345	456 Elm St, Townsville, State, 67890	john.doe@example.com	1234567890
	2	Jane Smith	789 Oak St, Villagetown, State, 54321	321 Pine St, Hamletville, State, 09876	jane.smith@example.com	9876543210
	3	Kenil Patel	2155 monastery lane, B3L4P9	2155 monastery lane, B3L4P9	Kenil.patel@dal.ca	1234746890
	4	Meshva Patel	10, Ratnadeep Bunglows, 380059	10, Ratnadeep Bunglows, 380059	meshva.patel@gmail.com	9877363210
*	NULL	NULL	NULL	NULL	NULL	NULL

Figure 1: Customer Details.



Transcation Create_procedure **accountdetail** customerdetail transferdetail

Limit to 1000 rows

1 • `SELECT * FROM bankdatabase.accountdetail;`

	AccNumber	AccBalance	CustomerID
▶	1	5000.00	1
	2	6000.00	2
	3	5000.00	3
	4	7000.00	4
*	NULL	NULL	NULL

Figure 2: Account Details.

Transactions

I have made a procedure which has transaction for the debit of money from the account. Below is the transaction:

```
DELIMITER //

CREATE PROCEDURE Transfer(IN debitAmount INT, IN accountid INT, IN status
VARCHAR(20))
BEGIN
    START TRANSACTION;

    SAVEPOINT IntialSavepoint;

    UPDATE AccountDetail
    SET AccBalance = AccBalance - debitAmount
    WHERE AccNumber = accountid;

    INSERT INTO TransferDetail (AccNumber, DateOfTransfer, RecipientName, Status)
    VALUES (accountid, CURDATE(), 'Manish', 'waiting');

    IF status = 'accept' THEN
        UPDATE TransferDetail
        SET Status = 'accept'
        WHERE AccNumber = accountid;
        COMMIT;
    ELSEIF status = 'decline' THEN
        ROLLBACK TO SAVEPOINT IntialSavepoint;
    END IF;
END //

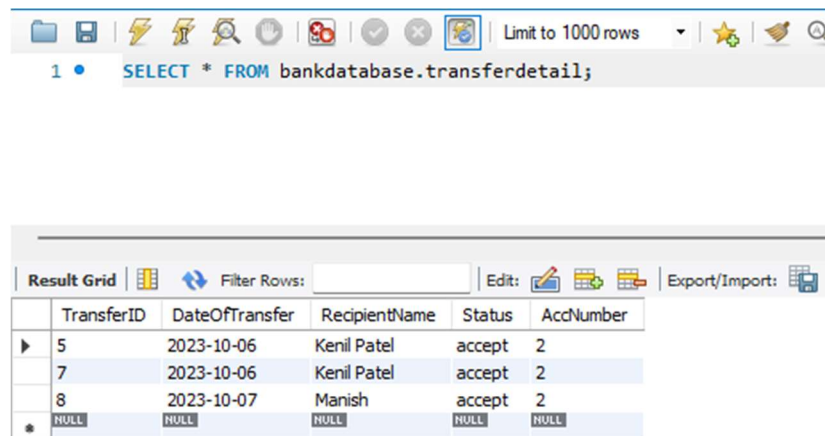
DELIMITER ;
```

In the above transaction, it will initially deduct money from the account whose account ID is given then it will make a new entry in the transferDetails table and at this point, the status will be “waiting”. But when it passes through some business logic and the status changes to the accepted it will update the status to “accept” in the transferDetail table. Otherwise, if the status is decline then it will roll back to the save point and there will be no update in the AccountDetails table.

We can call this procedure using the below sql script:

```
SET @status = 'accept';
SET @debitAmount = 500;
SET @accountid = 2;

CALL Transfer(@debitAmount, @accountid, @status);
```



The screenshot shows a database query tool interface. At the top, a toolbar contains icons for file operations, execution, and search. Below the toolbar, a text area displays the SQL query: `SELECT * FROM bankdatabase.transferdetail;`. The query is numbered '1'. Below the query, a 'Result Grid' is visible, showing a table with 6 columns: TransferID, DateOfTransfer, RecipientName, Status, and AccNumber. The table contains three rows of data, all with a status of 'accept'. The first row has TransferID 5, DateOfTransfer 2023-10-06, RecipientName Kenil Patel, and AccNumber 2. The second row has TransferID 7, DateOfTransfer 2023-10-06, RecipientName Kenil Patel, and AccNumber 2. The third row has TransferID 8, DateOfTransfer 2023-10-07, RecipientName Manish, and AccNumber 2. A fourth row is partially visible with NULL values.

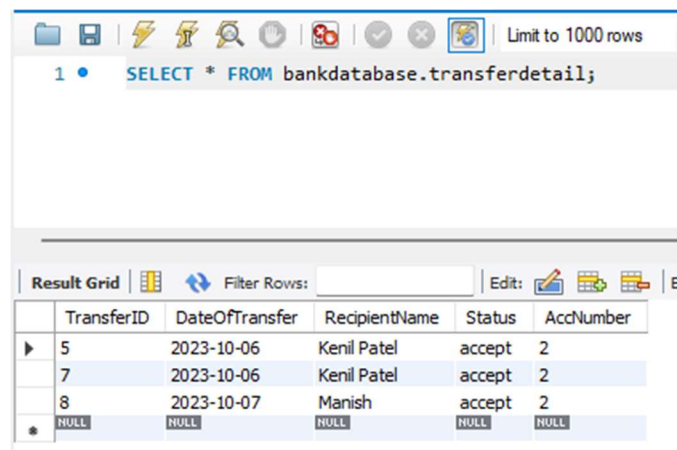
TransferID	DateOfTransfer	RecipientName	Status	AccNumber
5	2023-10-06	Kenil Patel	accept	2
7	2023-10-06	Kenil Patel	accept	2
8	2023-10-07	Manish	accept	2
NULL	NULL	NULL	NULL	NULL

Figure 3: Transfer Details when it is accepted.

```
SET @status = 'decline';
SET @debitAmount = 500;
SET @accountid = 2;

CALL Transfer(@debitAmount, @accountid, @status);
```

When we pass the status decline then it will not change the transfer table and roll back to the savepoint. Thus, there will be no changes in the transfer table.



This screenshot is identical to Figure 3, showing the same SQL query and result grid. The query is `SELECT * FROM bankdatabase.transferdetail;` and the result grid shows the same three rows of accepted transfers. This indicates that the 'decline' status passed in the preceding code block did not affect the data in the transfer table.

TransferID	DateOfTransfer	RecipientName	Status	AccNumber
5	2023-10-06	Kenil Patel	accept	2
7	2023-10-06	Kenil Patel	accept	2
8	2023-10-07	Manish	accept	2
NULL	NULL	NULL	NULL	NULL

Figure 4: Transfer table when it declines.

References

1. MySQL, "MySQL Workbench," *Mysql.com*, 2019.
<https://www.mysql.com/products/workbench/>