CSCI 5408

DATA MANAGEMENT AND WAREHOUSING

LAB - 3

Banner ID: B00984406

GitLab Assignment Link: N/A

Table of Contents

Designing a banking application database with the tables............................................................3

Inserting some sample/dummy data in the tables.……….............................................................7

Create transactions for the below two scenarios...........................................................................8

**Designing a banking application database with tables**

**Create a database called “lab3”:**

CREATE DATABASE IF NOT EXISTS lab3;

A screenshot of a computer

Description automatically generated  
Figure 1.1: Database lab3 created

**Create a table “customer\_details”:**

CREATE TABLE customer\_details (

customer\_id INT,

name VARCHAR(20),

address VARCHAR(50),

email VARCHAR(20),

phone\_number VARCHAR(15),

PRIMARY KEY (customer\_id)

);

A screenshot of a computer

Description automatically generated

Figure 1.2: customer\_details table created

**Create a table “account\_details”:**

CREATE TABLE account\_details(

account\_number INT,

account\_balance INT NOT NULL,

customer\_id INT,

PRIMARY KEY (account\_number),

FOREIGN KEY (customer\_id) REFERENCES customer\_details(customer\_id)

);

A screenshot of a computer

Description automatically generated

Figure 1.3: account\_details table created

**Create a table “customer\_transfer\_details”:**

CREATE TABLE account\_transfer\_details (

transfer\_id INT,

sender\_account\_number INT,

receiver\_account\_number INT,

transfer\_date DATETIME,

status ENUM ("waiting","accepted","declined"),

PRIMARY KEY (transfer\_id),

FOREIGN KEY (sender\_account\_number) REFERENCES

account\_details(account\_number),

FOREIGN KEY (receiver\_account\_number) REFERENCES

account\_details(account\_number)

);

A screenshot of a computer

Description automatically generated

Figure 1.4: account\_transfer\_details table created

**Inserting the sample/dummy data in tables**

**Inserting the data into the table “customer\_details”:**

INSERT INTO customer\_details(customer\_id, name, address, email, phone\_number) VALUES

(1, "jems", "811, 1881 Brunswick Street", "jems.patel@dal.ca", "+1-782-882-5653");

INSERT INTO customer\_details(customer\_id, name, address, email, phone\_number) VALUES

(2, "joli", "511, 2001 Brunswick Street", "joli1305@gmail.com", "+1-782-882-1305");

A screenshot of a computer

Description automatically generated

Figure 2.1: inserting data into the customer\_details table

**Inserting the data into the table “account\_details”:**

INSERT INTO account\_details(account\_number, account\_balance, customer\_id) VALUES

(125, 2500, 1);

INSERT INTO account\_details(account\_number, account\_balance, customer\_id) VALUES

(135, 5000, 2);

A screenshot of a computer

Description automatically generated

Figure 2.2: inserting data into the account\_details table

**Create transactions for the below two scenarios.**

3.A Scenario-1 (Transaction “Accepted” state): -

**Creating the transaction:**

SET AUTOCOMMIT=0;

START TRANSACTION;

UPDATE account\_details

SET account\_balance = account\_balance - 500

WHERE account\_number = 125;

INSERT INTO account\_transfer\_details (transfer\_id, sender\_account\_number, receiver\_account\_number, transfer\_date, status)

VALUES (1, 125, 135, NOW(), 'waiting');

-- Assuming the transaction has accepted (assumption)

UPDATE account\_details

SET account\_balance = account\_balance + 500

WHERE account\_number = 135;

UPDATE account\_transfer\_details

SET status = 'accepted'

WHERE transfer\_id = 1;

COMMIT;

**Explanation:**

* Initially, I disabled auto commit and initiated the transaction. Subsequently, I deducted 500 from the sender's account balance by specifying the account number.
* Following that, I inserted a new record into the account\_transfer\_details table with necessary details like sender's account number, receiver's account number, transfer date, and the transaction status set to "waiting".
* Assuming the transaction passed the security verification, I increased the receiver's account balance by 500 units.
* Finally, I changed the transaction status to "accepted" and committed the transaction.

**Screenshots:**

A screenshot of a computer

Description automatically generated

Figure 3.1: account\_details table after the successfully transaction

* It is observed that a total of 100 have been debited from account number 111 and credited to account number.

A screenshot of a computer

Description automatically generated

Figure 3.2: account\_transfer\_details table

* An additional entry has been made for the transaction, indicating it has been accepted.

3.B Scenario-2 (Transaction “Declined” state): -

SET AUTOCOMMIT=0;

START TRANSACTION;

UPDATE account\_details

SET account\_balance = account\_balance - 500

WHERE account\_number = 125;

INSERT INTO account\_transfer\_details (transfer\_id, sender\_account\_number, receiver\_account\_number, transfer\_date, status)

VALUES (2, 125, 135, NOW(), 'waiting');

SAVEPOINT before\_failure;

-- Assuming the transaction has failed (assumption)

ROLLBACK TO before\_failure;

UPDATE account\_details

SET account\_balance = account\_balance + 500

WHERE account\_number = 125;

UPDATE account\_transfer\_details

SET status = 'declined'

WHERE transfer\_id = 2;

COMMIT;

**Explanation:**

* Initially, I disabled auto commit and initiated the transaction. Subsequently, I deducted 500 from the sender's account balance by specifying the account number.
* Subsequent to that, I added a fresh entry into the account\_transfer\_details table with essential information such as the sender's account number, receiver's account number, transfer date, and the transaction status set as "waiting".
* Following this, I established a savepoint named before\_failure to enable reverting to this point in case of any transaction failure.
* In the event of a transaction failure (this is an assumption), I rolled back the transaction to the savepoint before\_failure, effectively reversing any modifications made post the savepoint.
* After rolling back, I restored the sender's account balance by adding back the 500 units that were previously deducted.
* Finally, I updated the transaction status to "declined" in the account\_transfer\_details table to indicate the failure of the transaction and then committed the transaction to finalize all changes.

**Screenshots:**

A screenshot of a computer

Description automatically generated

Figure 3.3: account\_details table after the failed transaction

* Both the sender and the receiver maintain their balance unchanged following the unsuccessful transaction.

A screenshot of a computer

Description automatically generated

Figure 3.4: account\_transfer\_details table

* An additional entry has been made for the transaction, indicating it has been declined.