Bank Management System

Team Name: ***Alpha***

Submitted by:

1.Name : Jannat Ara Simi

ID : 22201178

2. Name : Md. Sakib Hossaine

ID : 22201185

3. Name : Md Tabiur Rahman

ID : 22201171

4. Name : Mehedi Hasan Khan Riyad

ID : 22201172

Bank Management System

This project is a **Bank Management System** implemented using **MySQL**. It efficiently organizes and manages various aspects of a banking institution, such as customer details, accounts, loans, transactions, and credit cards. The system is designed to handle complex banking operations

Features

- Branch and banker management
- Customer account management
- Loan processing and payment tracking
- Transaction recording and analysis
- Credit card issuance and management

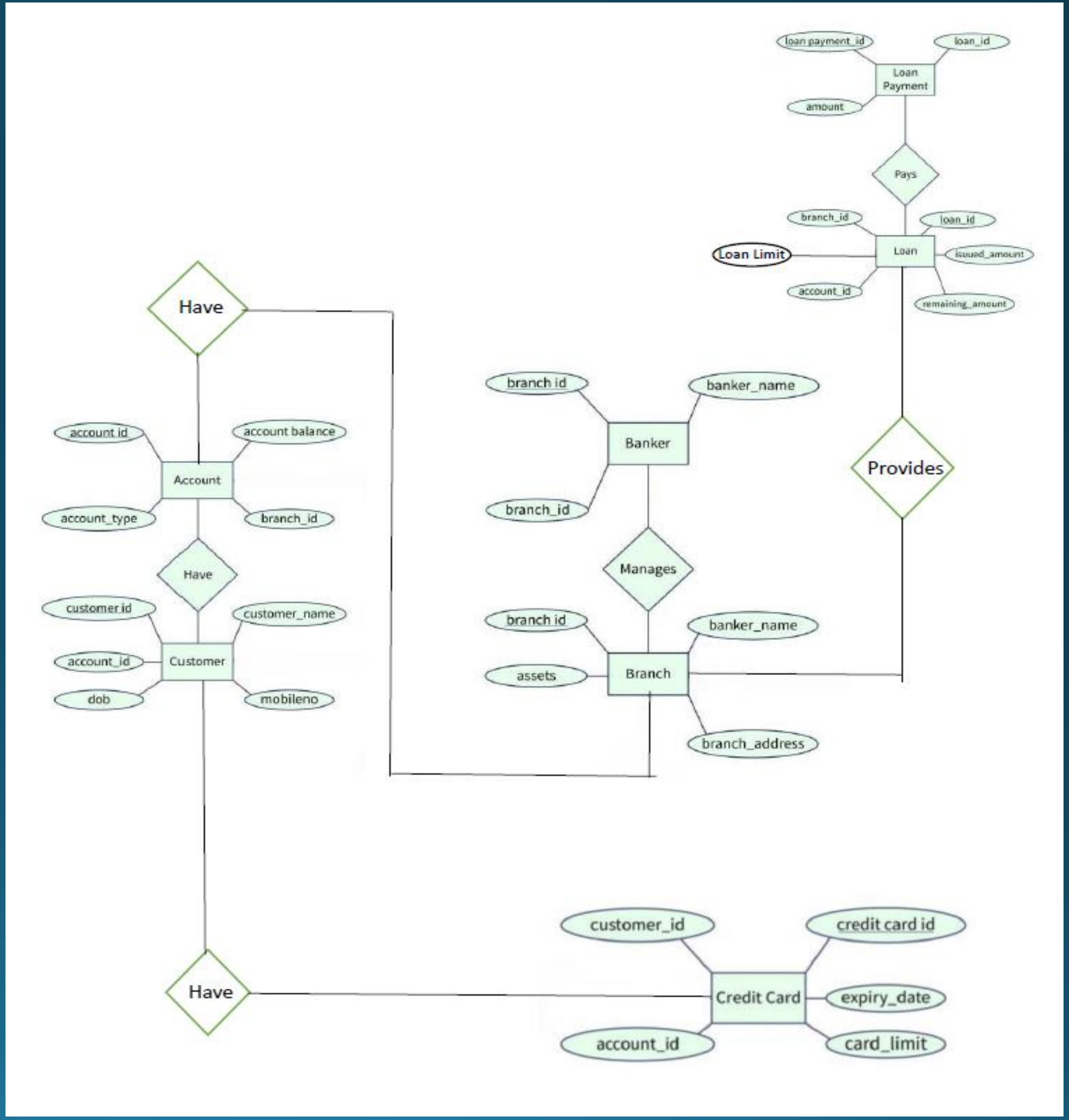
ER Diagram

1.Customer and Account:

- Each customer can have multiple accounts with attributes like account_type, account_balance, and branch_id.
- Customer details include customer_name, dob, and mobile_no.

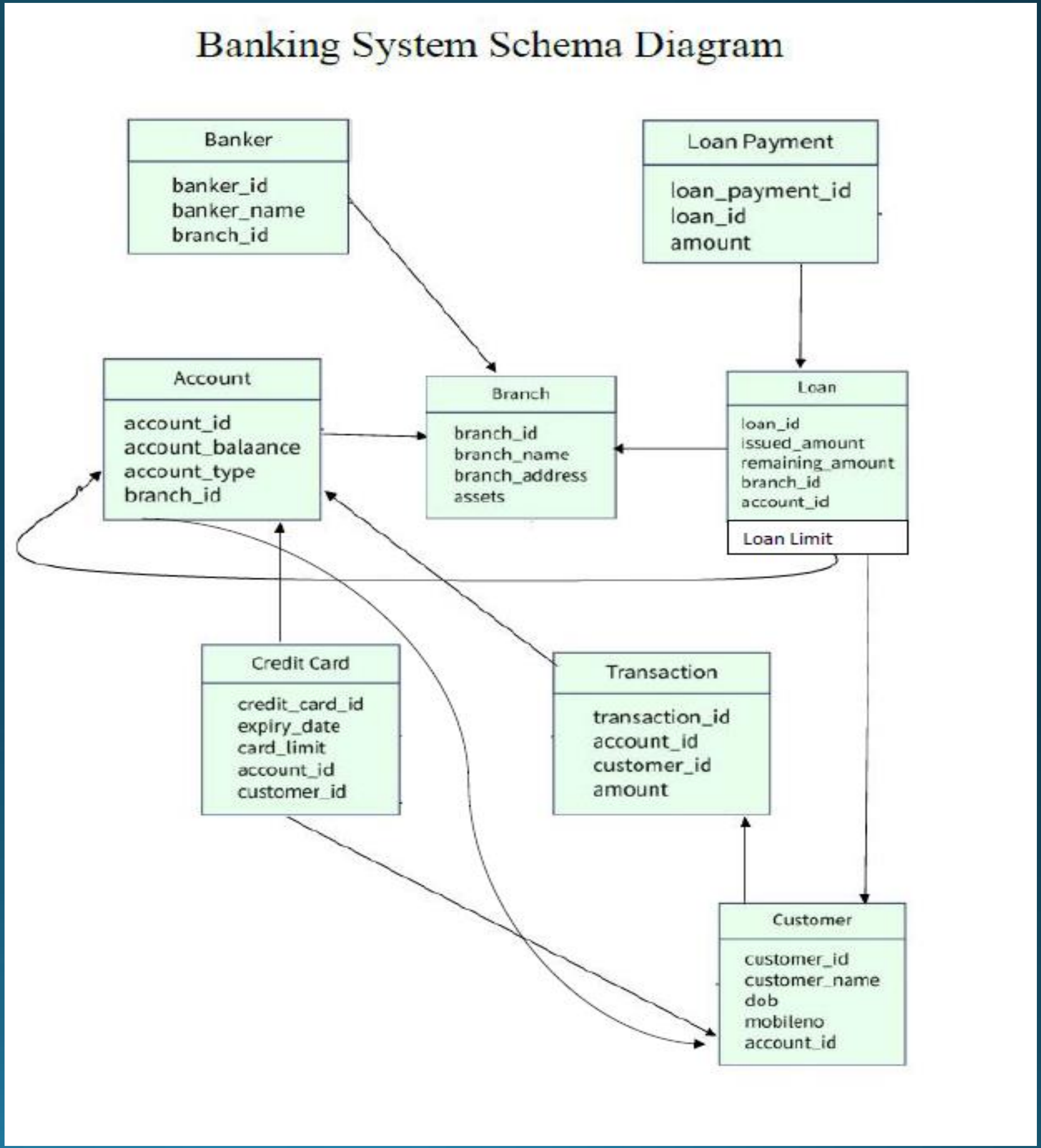
2.Branch and Banker:

- Branches are managed by bankers. Each branch has attributes like branch_id, branch_address, and assets.
- Bankers are linked to branches using branch_id and manage operations.



Schema Diagram

The relationships illustrate how these entities interact to manage banking operations like loans, accounts, and transactions.



Banking System Schema Diagram

Banker
banker_id
banker_name
branch_id

Loan Payment
loan_payment_id
loan_id
amount

Account
account_id
account_balaance
account_type
branch_id

Branch
branch_id
branch_name
branch_address
assets

Loan
loan_id
issued_amount
remaining_amount
branch_id
account_id
Loan Limit

Credit Card
credit_card_id
expiry_date
card_limit
account_id
customer_id

Transaction
transaction_id
account_id
customer_id
amount

Customer
customer_id
customer_name
dob
mobileno
account_id

Entities & Attributes

Entities:

- 1. **Branch** – Represents the physical location of the bank. Each branch offers services such as account creation, loan issuance, and credit card handling.
- 2. **Banker** – Employees assigned to manage customer relationships and oversee operations for specific branches.
- 3. **Customer** – Individuals who interact with the bank to open accounts, apply for loans, and manage their finances.

Attributes:

➢ ***Branch***

- branch_id (Primary Key)
- branch_name
- address

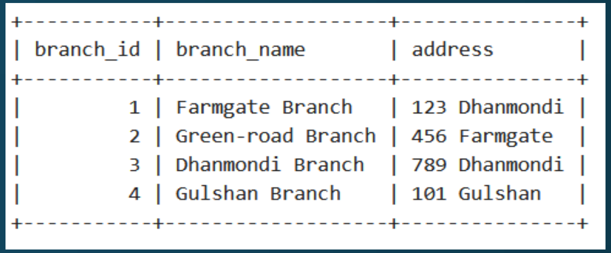
➢ ***Banker***

- banker_id (Primary Key)
- branch_id (Foreign Key referencing branch)
- banker_name

MY SQL QUERYS

1. Finding Branch table data.

SELECT * FROM branch;



```
+-----------+------------------+---------------+
| branch_id | branch_name      | address       |
+-----------+------------------+---------------+
|         1 | Farmgate Branch  | 123 Dhanmondi |
|         2 | Green-road Branch | 456 Farmgate  |
|         3 | Dhanmondi Branch | 789 Dhanmondi |
|         4 | Gulshan Branch   | 101 Gulshan   |
+-----------+------------------+---------------+
```

2. Find all customers with a savings account.

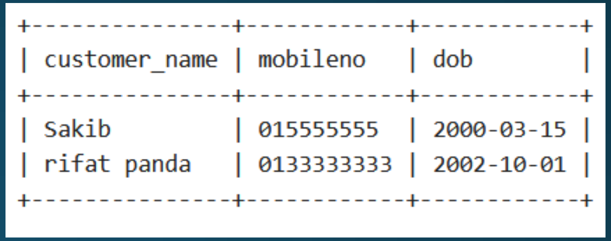
SELECT customer_name, mobileno, dob

FROM customer

INNER JOIN account ON

customer.account_id = account.account_id

WHERE account_type = 'savings';



```
+---------------+------------+------------+
| customer_name | mobileno   | dob        |
+---------------+------------+------------+
| Sakib         | 015555555  | 2000-03-15 |
| rifat panda   | 0133333333 | 2002-10-01 |
+---------------+------------+------------+
```

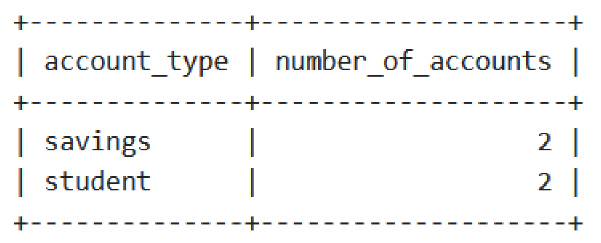
3. Count the number of accounts per account type (student or savings).

SELECT account_type, COUNT(account_id)

AS number_of_accounts

GROUP BY account_type;

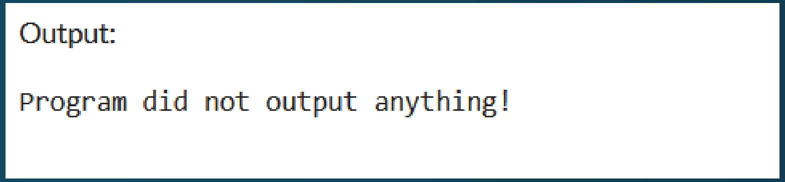
FROM account



```
+--------------+--------------------+
| account_type | number_of_accounts |
+--------------+--------------------+
| savings      |                  2 |
| student      |                  2 |
+--------------+--------------------+
```

4. Show all bankers working at a specific branch

```
SELECT banker.banker_name
FROM banker
INNER JOIN branch ON banker.branch_id = branch.branch_id
WHERE branch.branch_name = 'Main Branch';
```

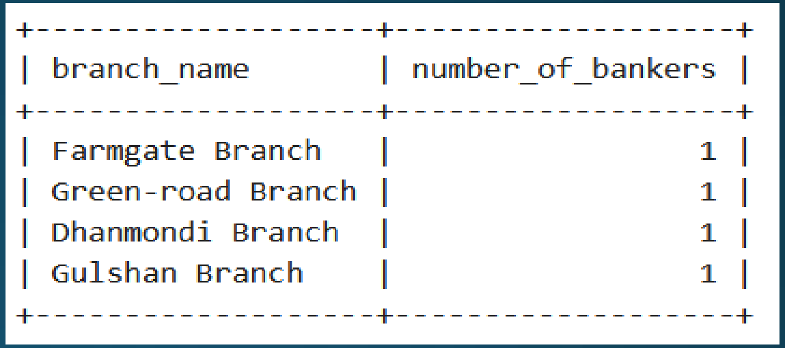


Output:

```
Program did not output anything!
```

5. List all branches with the number of bankers working in each

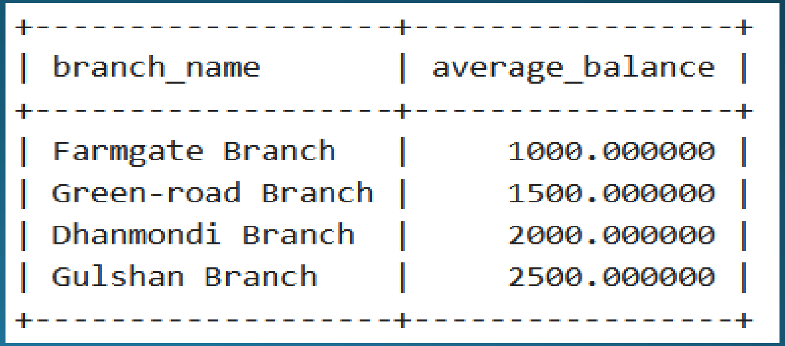
```
SELECT branch.branch_name, COUNT(banker.banker_id)
AS  number_of_bankers
FROM branch
LEFT JOIN banker ON branch.branch_id = banker.branch_id
GROUP BY branch.branch_name;
```



```
+-------------------+-------------------+
| branch_name       | number_of_bankers |
+-------------------+-------------------+
| Farmgate Branch   |                 1 |
| Green-road Branch |                 1 |
| Dhanmondi Branch  |                 1 |
| Gulshan Branch    |                 1 |
+-------------------+-------------------+
```

6. Calculate the average balance of accounts for each branch

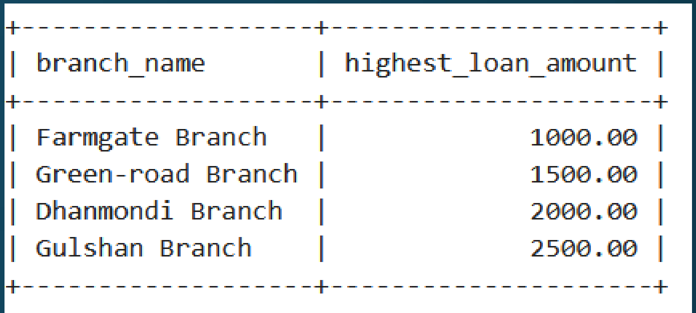
```
SELECT branch.branch_name, AVG(account.balance) AS
average_balance
FROM branch
INNER JOIN account ON branch.branch_id = account.branch_id
GROUP BY branch.branch_name;
```



```
+-------------------+-----------------+
| branch_name       | average_balance |
+-------------------+-----------------+
| Farmgate Branch   |     1000.000000 |
| Green-road Branch |     1500.000000 |
| Dhanmondi Branch  |     2000.000000 |
| Gulshan Branch    |     2500.000000 |
+-------------------+-----------------+
```

7. Retrieve the highest loan amount issued at each branch

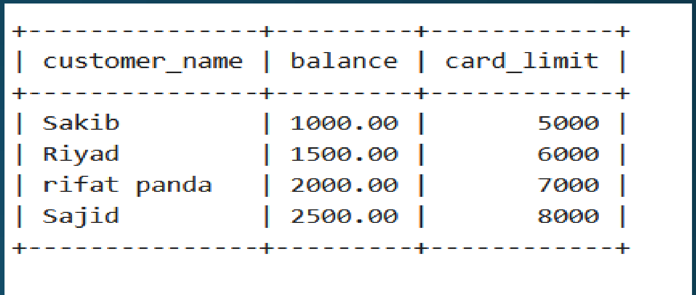
```
SELECT branch.branch_name, MAX(loan.amount) AS highest_loan_amount
                 FROM loan
    INNER JOIN branch ON loan.branch_id = branch.branch_id
                GROUP BY branch.branch_name;
```



```
+------------------+---------------------+
| branch_name      | highest_loan_amount |
+------------------+---------------------+
| Farmgate Branch  |             1000.00 |
| Green-road Branch |            1500.00 |
| Dhanmondi Branch |             2000.00 |
| Gulshan Branch   |             2500.00 |
+------------------+---------------------+
```

8. Find all credit card holders with their account balances and credit limits

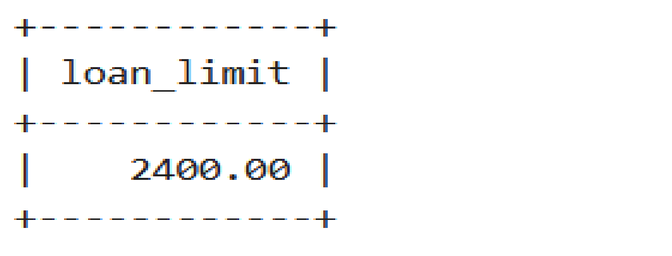
```
           SELECT customer.customer_name,
account.balance,customer_credit_card.card_limit
                                FROM customer
    INNER JOIN account ON customer.account_id = account.account_id
        INNER JOIN customer_credit_card ON customer.customer_id
                =  customer_credit_card.customer_id;
```



```
+---------------+---------+------------+
| customer_name | balance | card_limit |
+---------------+---------+------------+
| Sakib         | 1000.00 |       5000 |
| Riyad         | 1500.00 |       6000 |
| rifat panda   | 2000.00 |       7000 |
| Sajid         | 2500.00 |       8000 |
+---------------+---------+------------+
```

9. "Find Loan Limit for Sakib"

```
    SELECT l.loan_limitFROM customer cJOIN account a ON c.account_id =
a.account_idJOIN loan l ON a.account_id = l.account_idWHERE
c.customer_name = 'rifat panda';
```

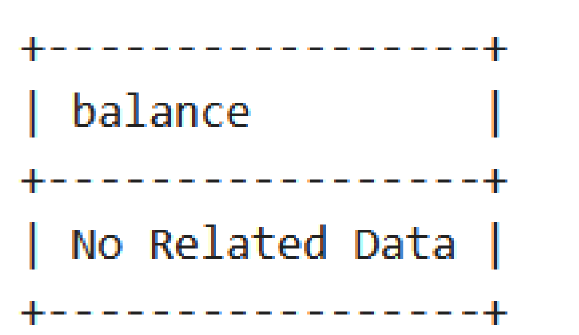


```
+------------+
| loan_limit |
+------------+
|    2400.00 |
+------------+
```

Corner Case

10. Retrieve the balances of all accounts with a balance greater than 5000. If no such accounts exist, display 'No Related Data

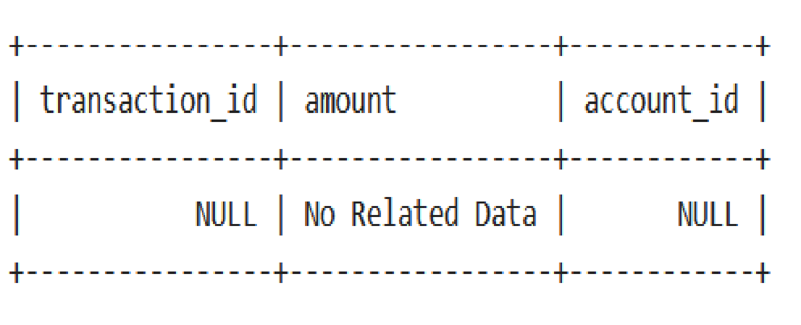
```
         SELECT CAST(balance AS CHAR) AS balance
         FROM account WHERE balance > 5000
         UNION
      SELECT 'No Related Data' WHERE NOT EXISTS (
           SELECT 1 FROM account
            WHERE balance > 5000 );
```



```
+------------------+
| balance          |
+------------------+
| No Related Data  |
+------------------+
```

11. Find all transactions for a specific customer

```
SELECT transaction.transaction_id, transaction.amount, transaction.account_id
         FROM transaction
     INNER JOIN customer ON transaction.customer_id =   customer.customer_id
     WHERE customer.customer_name = 'nonexistent_customer'
           UNION
          SELECT NULL, 'No Related Data', NULL
     WHERE NOT EXISTS (   SELECT 1  FROM transaction
   INNER JOIN customer ON transaction.customer_id = customer.customer_id
    WHERE customer.customer_name = 'nonexistent_customer'  );
```



```
+----------------+-----------------+------------+
| transaction_id | amount          | account_id |
+----------------+-----------------+------------+
|           NULL | No Related Data |       NULL |
+----------------+-----------------+------------+
```

Thank you!