**Simple Bank Database**

(Using SQL)



# Project Aim and Objectives

**Aim:**

To design and implement a relational database system for a bank that efficiently manages customers, accounts, loans, branches, transactions, and employees using SQL.

**Objectives:**

• Store and manage bank customer data securely.

• Keep track of customer accounts and balances.

• Manage transactions and their history for auditing.

• Record loan details and repayments.

• Maintain branch-level details.

• Manage bank employees and salaries.

• Enable data retrieval through SQL queries and reports.

ER Diagram of bank database.

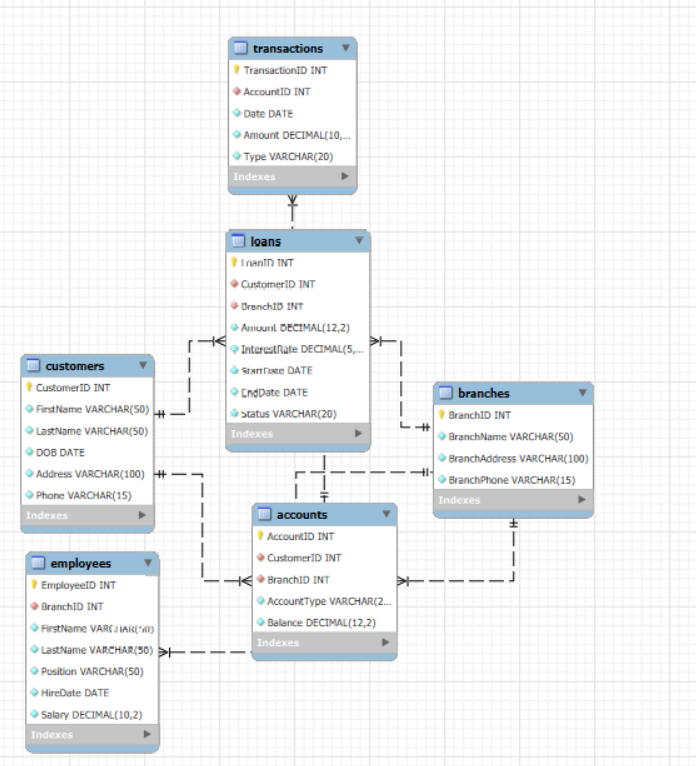
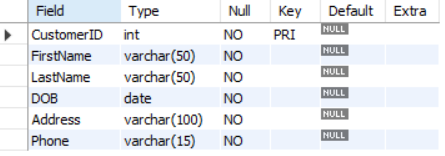
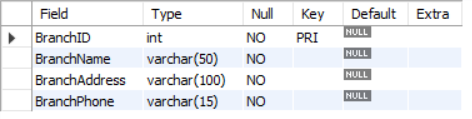


TABLE DESCRIPTION

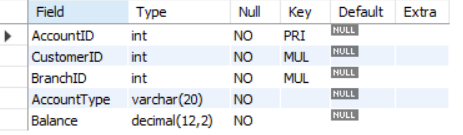
1. CUSTOMERS:



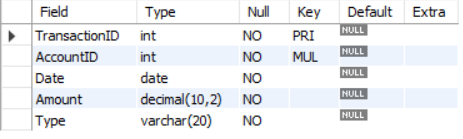
1. BRANCHES:



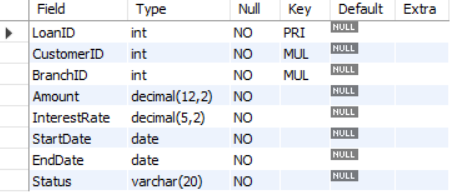
1. ACCOUNTS:



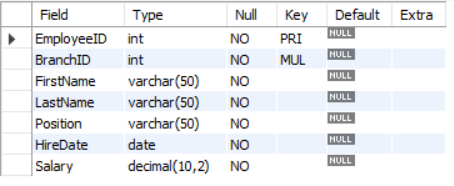
1. TRANSACTIONS:



1. LOANS:



1. EMPLOYEES:



* **CREATING DATABASE:-**

Create database bank database;

Use bank database;

Table Creation and Insertion Commands:

1. **CREATE TABLE Customers (**

CustomerID INT PRIMARY KEY**,**

FirstName VARCHAR(50) NOT NULL,

LastName VARCHAR(50) NOT NULL,

DOB DATE NOT NULL,

Address VARCHAR(100) NOT NULL,

Phone VARCHAR(15) NOT NULL

)

**INSERT INTO Customers** (CustomerID, FirstName, LastName, DOB, Address, Phone) VALUES

(1, 'John', 'Smith', '1985-02-15', '123 Elm St', '555-1001'),

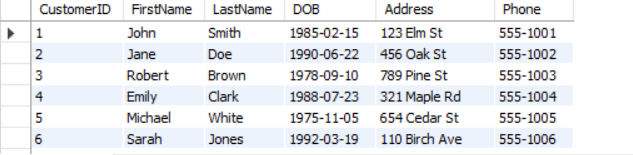
(2, 'Jane', 'Doe', '1990-06-22', '456 Oak St', '555-1002'),

(3, 'Robert', 'Brown', '1978-09-10', '789 Pine St', '555-1003'),

(4, 'Emily', 'Clark', '1988-07-23', '321 Maple Rd', '555-1004'),

(5, 'Michael', 'White', '1975-11-05', '654 Cedar St', '555-1005'),etc.

**Select \* from Customers;**



**2).CREATE TABLE Branches** (

BranchID INT PRIMARY KEY,

BranchName VARCHAR(50) NOT NULL,

BranchAddress VARCHAR(100) NOT NULL,

BranchPhone VARCHAR(15) NOT NULL

);

**INSERT INTO Branches** (BranchID, BranchName, BranchAddress, BranchPhone) VALUES

(1, 'Main Branch', '100 Main St', '555-2001'),

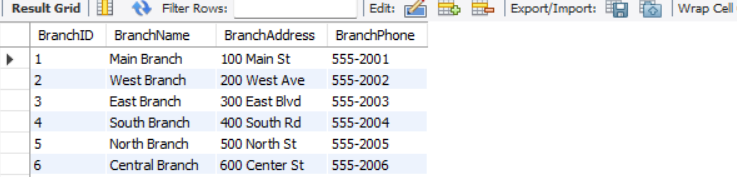
(2, 'West Branch', '200 West Ave', '555-2002'),

(3, 'East Branch', '300 East Blvd', '555-2003'),

(4, 'South Branch', '400 South Rd', '555-2004'),

(5, 'North Branch', '500 North St', '555-2005')

**Select \* from Branches;**



**3).REATE TABLE Accounts** (

AccountID INT PRIMARY KEY,

CustomerID INT NOT NULL,

BranchID INT NOT NULL,

AccountType VARCHAR(20) NOT NULL,

Balance DECIMAL(12,2) NOT NULL,

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),

FOREIGN KEY (BranchID) REFERENCES Branches(BranchID)

);

**INSERT INTO Accounts** (AccountID, CustomerID, BranchID, AccountType, Balance) VALUES

(1, 1, 1, 'Savings', 1200.50),

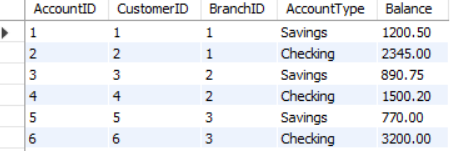
(2, 2, 1, 'Checking', 2345.00),

(3, 3, 2, 'Savings', 890.75),

(4, 4, 2, 'Checking', 1500.20),

(5, 5, 3, 'Savings', 770.00)

**Select \* from accounts;**



**4). CREATE TABLE Transactions** (

TransactionID INT PRIMARY KEY,

AccountID INT NOT NULL,

Date DATE NOT NULL,

Amount DECIMAL(10,2) NOT NULL,

Type VARCHAR(20) NOT NULL,

FOREIGN KEY (AccountID) REFERENCES Accounts(AccountID)

);

**INSERT INTO Transactions** (TransactionID, AccountID, Date, Amount, Type) VALUES

(1, 1, '2025-01-10', 100.00, 'Deposit'),

(2, 1, '2025-01-20', 50.00, 'Withdrawal'),

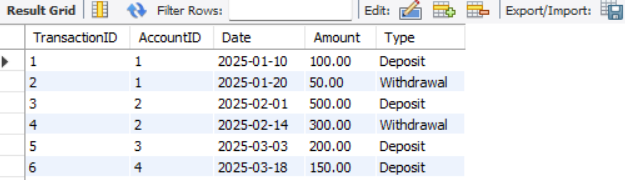
(3, 2, '2025-02-01', 500.00, 'Deposit'),

(4, 2, '2025-02-14', 300.00, 'Withdrawal'),

(5, 3, '2025-03-03', 200.00, 'Deposit'),

(6, 4, '2025-03-18', 150.00, 'Deposit'),etc.

**Select \* from accounts;**



**5).CREATE TABLE Loans** (

LoanID INT PRIMARY KEY,

CustomerID INT NOT NULL,

BranchID INT NOT NULL,

Amount DECIMAL(12,2) NOT NULL,

InterestRate DECIMAL(5,2) NOT NULL,

StartDate DATE NOT NULL,

EndDate DATE NOT NULL,

Status VARCHAR(20) NOT NULL,

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),

FOREIGN KEY (BranchID) REFERENCES Branches(BranchID)

);

**INSERT INTO Loans** (LoanID, CustomerID, BranchID, Amount, InterestRate, StartDate, EndDate, Status) VALUES

(1, 1, 1, 5000.00, 5.50, '2025-01-10', '2028-01-10', 'Active'),

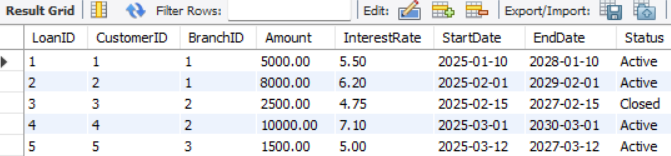
(2, 2, 1, 8000.00, 6.20, '2025-02-01', '2029-02-01', 'Active'),

(3, 3, 2, 2500.00, 4.75, '2025-02-15', '2027-02-15', 'Closed'),

(4, 4, 2, 10000.00, 7.10, '2025-03-01', '2030-03-01', 'Active'),

(5, 5, 3, 1500.00, 5.00, '2025-03-12', '2027-03-12', 'Active'),etc.

**Select \* from Loans;**



**6).** **CREATE TABLE** Employees (

EmployeeID INT PRIMARY KEY,

BranchID INT NOT NULL,

FirstName VARCHAR(50) NOT NULL,

LastName VARCHAR(50) NOT NULL,

Position VARCHAR(50) NOT NULL,

HireDate DATE NOT NULL,

Salary DECIMAL(10,2) NOT NULL,

FOREIGN KEY (BranchID) REFERENCES Branches(BranchID)

);

**INSERT INTO Employees** (EmployeeID, BranchID, FirstName, LastName, Position, HireDate, Salary) VALUES

(1, 21, 'Lisa', 'Lopez', 'Manager', '2017-03-30', 36125.48),

(2, 9, 'Anthony', 'Thompson', 'Teller', '2017-12-10', 45044.48),

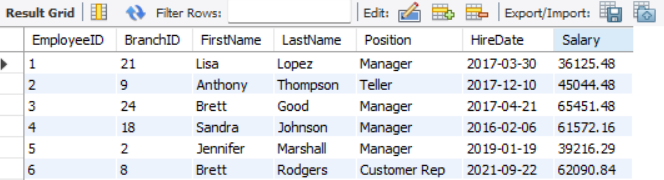
(3, 24, 'Brett', 'Good', 'Manager', '2017-04-21', 65451.48),

(4, 18, 'Sandra', 'Johnson', 'Manager', '2016-02-06', 61572.16),

(5, 2, 'Jennifer', 'Marshall', 'Manager', '2019-01-19', 39216.29),

(6, 8, 'Brett', 'Rodgers', 'Customer Rep', '2021-09-22', 62090.84),etc.

**Select \* from Employees;**



**BASIC QUERIES -SQL**

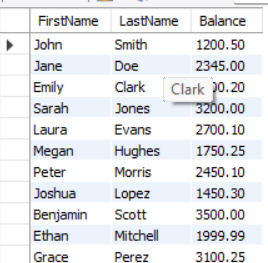
1. **find all customers who have balance over 1000/-**

SELECT DISTINCT c.FirstName, c.LastName, a.Balance

FROM Customers c

JOIN Accounts a ON c.CustomerID = a.CustomerID

WHERE a.Balance > 1000;

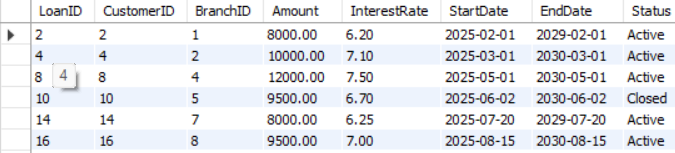


1. **list loans with interest rate avove 6%**

SELECT \*

FROM Loans

WHERE InterestRate > 6.00;

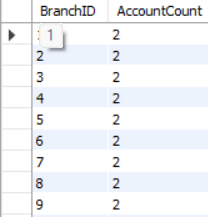


1. **count number of accounts per branch**

SELECT BranchID, COUNT(\*) AS AccountCount

FROM Accounts

GROUP BY BranchID;



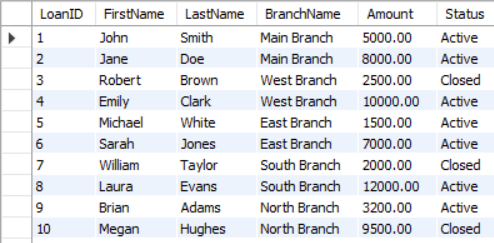
1. **show loans with customer names and branch names**

SELECT l.LoanID, c.FirstName, c.LastName, b.BranchName, l.Amount, l.Status

FROM Loans l

JOIN Customers c ON l.CustomerID = c.CustomerID

JOIN Branches b ON l.BranchID = b.BranchID;

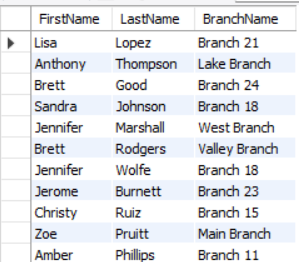


**5) joine employees and branches to list employee names with branch names**

SELECT e.FirstName, e.LastName, b.BranchName

FROM Employees e

JOIN Branches b ON e.BranchID = b.BranchID;

****

**CONCLUSION:**

The Simple Bank Database project successfully demonstrates how to design and implement a relational database system for a banking environment using SQL. By defining well-structured tables for customers, accounts, loans, transactions, branches, and employees, the system ensures secure and organized storage of essential banking data.

Through carefully constructed SQL queries—including complex joins and data manipulations—the project highlights how such a database can support daily banking operations like managing customer information, tracking transactions, processing loans, and generating insightful reports.

Overall, the Simple Bank Database provides a practical and efficient approach to handling core banking data requirements, ensuring both data integrity and ease of information retrieval for decision-making and customer service.