Banking System Database Documentation

# Overview

This project involves designing and implementing a database schema for a banking system capable of managing customer information, accounts, transactions, and other relevant data. The database schema follows best practices for data integrity, security, and performance.

# Database Schema

## 1. Customers Table

Description: Stores information about the customers of the banking system.

Columns:  
- CustomerID (INT, AUTO\_INCREMENT, PRIMARY KEY): Unique identifier for each customer.  
- FirstName (VARCHAR(50)): Customer's first name.  
- LastName (VARCHAR(50)): Customer's last name.  
- Email (VARCHAR(100)): Customer's email address.  
- Phone (VARCHAR(15)): Customer's phone number.  
- Address (VARCHAR(255)): Customer's physical address.

## 2. Accounts Table

Description: Stores information about the accounts held by customers.

Columns:  
- AccountID (INT, AUTO\_INCREMENT, PRIMARY KEY): Unique identifier for each account.  
- CustomerID (INT, FOREIGN KEY): Identifier linking the account to a customer in the Customers table.  
- AccountNumber (VARCHAR(20), UNIQUE): Unique account number.  
- AccountType (VARCHAR(20)): Type of the account (e.g., Savings, Checking).  
- Balance (DECIMAL(18, 2), DEFAULT 0.00): Current balance of the account.  
- CreatedDate (TIMESTAMP, DEFAULT CURRENT\_TIMESTAMP): Date and time when the account was created.

## 3. Transactions Table

Description: Stores information about the transactions performed on accounts.

Columns:  
- TransactionID (INT, AUTO\_INCREMENT, PRIMARY KEY): Unique identifier for each transaction.  
- AccountID (INT, FOREIGN KEY): Identifier linking the transaction to an account in the Accounts table.  
- TransactionType (VARCHAR(50)): Type of transaction (e.g., Deposit, Withdrawal, Transfer).  
- Amount (DECIMAL(18, 2)): Amount involved in the transaction.  
- TransactionDate (TIMESTAMP, DEFAULT CURRENT\_TIMESTAMP): Date and time when the transaction occurred.

SQL:

*```sql  
-- Create database*

*CREATE DATABASE BankingSystem;*

*USE BankingSystem;*

*-- Table for customer information*

*CREATE TABLE Customers (*

*CustomerID INT AUTO\_INCREMENT PRIMARY KEY,*

*FirstName VARCHAR(50),*

*LastName VARCHAR(50),*

*Email VARCHAR(100),*

*Phone VARCHAR(15),*

*Address VARCHAR(255)*

*);*

*-- Table for account details*

*CREATE TABLE Accounts (*

*AccountID INT AUTO\_INCREMENT PRIMARY KEY,*

*CustomerID INT,*

*AccountNumber VARCHAR(20) UNIQUE,*

*AccountType VARCHAR(20),*

*Balance DECIMAL(18, 2) DEFAULT 0.00,*

*CreatedDate TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,*

*FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)*

*);*

*-- Table for transaction history*

*CREATE TABLE Transactions (*

*TransactionID INT AUTO\_INCREMENT PRIMARY KEY,*

*AccountID INT,*

*TransactionType VARCHAR(50),*

*Amount DECIMAL(18, 2),*

*TransactionDate TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,*

*FOREIGN KEY (AccountID) REFERENCES Accounts(AccountID)*

*);  
```*

# Stored Procedures

## 1. sp\_CreateCustomer

Description: Inserts a new customer into the Customers table.

Parameters:  
- FirstName (VARCHAR(50)): Customer's first name.  
- LastName (VARCHAR(50)): Customer's last name.  
- Email (VARCHAR(100)): Customer's email address.  
- Phone (VARCHAR(15)): Customer's phone number.  
- Address (VARCHAR(255)): Customer's physical address.

SQL:

```sql  
DELIMITER $$  
CREATE PROCEDURE sp\_CreateCustomer (  
 IN FirstName VARCHAR(50),  
 IN LastName VARCHAR(50),  
 IN Email VARCHAR(100),  
 IN Phone VARCHAR(15),  
 IN Address VARCHAR(255)  
)  
BEGIN  
 INSERT INTO Customers (FirstName, LastName, Email, Phone, Address)  
 VALUES (FirstName, LastName, Email, Phone, Address);  
END $$  
DELIMITER ;  
```

## 2. sp\_OpenAccount

Description: Opens a new account for a customer.

Parameters:  
- CustomerID (INT): Identifier of the customer.  
- AccountNumber (VARCHAR(20)): Unique account number.  
- AccountType (VARCHAR(20)): Type of the account (e.g., Savings, Checking).

SQL:

```sql  
DELIMITER $$  
CREATE PROCEDURE sp\_OpenAccount (  
 IN CustomerID INT,  
 IN AccountNumber VARCHAR(20),  
 IN AccountType VARCHAR(20)  
)  
BEGIN  
 INSERT INTO Accounts (CustomerID, AccountNumber, AccountType)  
 VALUES (CustomerID, AccountNumber, AccountType);  
END $$  
DELIMITER ;  
```

## 3. sp\_DepositMoney

Description: Deposits money into an account.

Parameters:  
- AccountID (INT): Identifier of the account.  
- Amount (DECIMAL(18, 2)): Amount to be deposited.

SQL:

```sql  
DELIMITER $$  
CREATE PROCEDURE sp\_DepositMoney (  
 IN AccountID INT,  
 IN Amount DECIMAL(18, 2)  
)  
BEGIN  
 UPDATE Accounts  
 SET Balance = Balance + Amount  
 WHERE AccountID = AccountID;  
  
 INSERT INTO Transactions (AccountID, TransactionType, Amount)  
 VALUES (AccountID, 'Deposit', Amount);  
END $$  
DELIMITER ;  
```

## 4. sp\_WithdrawMoney

Description: Withdraws money from an account.

Parameters:  
- AccountID (INT): Identifier of the account.  
- Amount (DECIMAL(18, 2)): Amount to be withdrawn.

SQL:

```sql  
DELIMITER $$  
CREATE PROCEDURE sp\_WithdrawMoney (  
 IN AccountID INT,  
 IN Amount DECIMAL(18, 2)  
)  
BEGIN  
 UPDATE Accounts  
 SET Balance = Balance - Amount  
 WHERE AccountID = AccountID;  
  
 INSERT INTO Transactions (AccountID, TransactionType, Amount)  
 VALUES (AccountID, 'Withdrawal', Amount);  
END $$  
DELIMITER ;  
```

## 5. sp\_TransferMoney

Description: Transfers money from one account to another.

Parameters:  
- FromAccountID (INT): Identifier of the source account.  
- ToAccountID (INT): Identifier of the destination account.  
- Amount (DECIMAL(18, 2)): Amount to be transferred.

SQL:

```sql  
DELIMITER $$  
CREATE PROCEDURE sp\_TransferMoney (  
 IN FromAccountID INT,  
 IN ToAccountID INT,  
 IN Amount DECIMAL(18, 2)  
)  
BEGIN  
 START TRANSACTION;  
  
 UPDATE Accounts  
 SET Balance = Balance - Amount  
 WHERE AccountID = FromAccountID;  
  
 UPDATE Accounts  
 SET Balance = Balance + Amount  
 WHERE AccountID = ToAccountID;  
  
 INSERT INTO Transactions (AccountID, TransactionType, Amount)  
 VALUES (FromAccountID, 'Transfer Out', Amount),  
 (ToAccountID, 'Transfer In', Amount);  
  
 COMMIT;  
END $$  
DELIMITER ;  
```

## 6. sp\_ViewTransactionHistory

Description: Retrieves the transaction history for a specific account.

Parameters:  
- AccountID (INT): Identifier of the account.

SQL:

```sql  
DELIMITER $$  
CREATE PROCEDURE sp\_ViewTransactionHistory (  
 IN AccountID INT  
)  
BEGIN  
 SELECT TransactionID, AccountID, TransactionType, Amount, TransactionDate  
 FROM Transactions  
 WHERE AccountID = AccountID  
 ORDER BY TransactionDate DESC;  
END $$  
DELIMITER ;  
```