|  |  |
| --- | --- |
| Customer\_Types | |
| Customer\_Types\_code | INT(10) (PK) |
| Customer\_Types\_description | VARCHAR (50) |

|  |  |
| --- | --- |
| Customers | |
| Customers\_ID | INT(10) (PK) |
| Customers\_name | VARCHAR(50) |
| Customers\_phone | INT(10) |
| Customers\_email | VARCHAR(50) |
| Date\_became\_customer | DATE |
| Login | VARCHAR(50) |
| Password | VARCHAR(50) |
| Other\_details | VARCHAR(50) |
| Customer\_types\_code | INT(10) (FK) |
| Nominee\_name | VARCHAR(50) |
| Nominee\_relation | VARCHAR(50) |
| Nominee\_phone | INT(10) |
| Nominee\_address | VARCHAR(50) |

|  |  |
| --- | --- |
| Account\_Types | |
| Account\_Types\_code | VARCHAR(15) (PK) |
| Account\_Types\_description | VARCHAR(50) |

|  |  |
| --- | --- |
| Accounts | |
| Account\_ID | INT(10) (PK) |
| Accounts\_name | VARCHAR(50) |
| Date\_opened | DATE |
| Other\_account\_details | VARCHAR(50) |
| Account\_types\_code | INT(10) (FK) |
| Customer\_ID | INT(10) (FK) |
| Debit | FLOAT (8,1) (FK) |
| Credit | FLOAT (8,1) (FK) |
| Balance | FLOAT (8,1) (FK) |

|  |  |
| --- | --- |
| Admin\_user | |
| Admin\_user\_Login | VARCHAR(50) (PK) |
| Admin\_user\_ Password | VARCHAR(50) |

|  |  |
| --- | --- |
| Transactions | |
| Transaction\_ID | INT(10) (PK) |
| Date | DATE |
| Debit | FLOAT (8,1) |
| Credit | FLOAT (8,1) |
| Balance | FLOAT (8,1) |
| Other\_details | VARCHAR(50) |
| Purchase\_ID | INT(10) (FK) |
| Account\_ID | INT(10) (FK) |
| Transaction\_types\_code | VARCHAR(50) (FK) |

CREATE DATABASE BankingDatabase;

GO

USE BankingDatabase;

GO

CREATE TABLE Customer\_Types (

Customer\_Types\_code INT PRIMARY KEY,

Customer\_Types\_description VARCHAR(50)

);

GO

CREATE TABLE Customers (

Customers\_ID INT PRIMARY KEY,

Customers\_name VARCHAR(50),

Customers\_phone BIGINT, -- Changed data type to BIGINT

Customers\_email VARCHAR(50),

Date\_became\_customer DATE,

Login VARCHAR(50),

Password VARCHAR(50),

Other\_details VARCHAR(50),

Customer\_types\_code INT,

Nominee\_name VARCHAR(50),

Nominee\_relation VARCHAR(50),

Nominee\_phone BIGINT, -- Changed data type to BIGINT

Nominee\_address VARCHAR(50),

FOREIGN KEY (Customer\_types\_code) REFERENCES Customer\_Types(Customer\_Types\_code)

);

GO

CREATE TABLE Account\_Types (

Account\_Types\_code VARCHAR(15) PRIMARY KEY,

Account\_Types\_description VARCHAR(50)

);

GO

CREATE TABLE Accounts (

Account\_ID INT PRIMARY KEY,

Accounts\_name VARCHAR(50),

Date\_opened DATE,

Other\_account\_details VARCHAR(50),

Account\_types\_code VARCHAR(15),

Customer\_ID INT,

Debit DECIMAL(8,1),

Credit DECIMAL(8,1),

Balance DECIMAL(8,1),

FOREIGN KEY (Account\_types\_code) REFERENCES Account\_Types(Account\_Types\_code),

FOREIGN KEY (Customer\_ID) REFERENCES Customers(Customers\_ID)

);

GO

CREATE TABLE Admin\_user (

Admin\_user\_Login VARCHAR(50) PRIMARY KEY,

Admin\_user\_Password VARCHAR(50)

);

GO

CREATE TABLE Transaction\_Types (

Transaction\_Types\_code VARCHAR(50) PRIMARY KEY,

Transaction\_Types\_description VARCHAR(50)

);

CREATE TABLE Transactions (

Transaction\_ID INT PRIMARY KEY,

Date DATE,

Debit DECIMAL(8,1),

Credit DECIMAL(8,1),

Balance DECIMAL(8,1),

Other\_details VARCHAR(50),

Purchase\_ID INT,

Account\_ID INT,

Transaction\_types\_code VARCHAR(50),

FOREIGN KEY (Purchase\_ID) REFERENCES Transactions(Transaction\_ID),

FOREIGN KEY (Account\_ID) REFERENCES Accounts(Account\_ID),

FOREIGN KEY (Transaction\_types\_code) REFERENCES Transaction\_Types(Transaction\_Types\_code)

);

-- Retrieve data from Customer\_Types table

SELECT \* FROM Customer\_Types;

-- Retrieve data from Customers table

SELECT \* FROM Customers;

-- Retrieve data from Account\_Types table

SELECT \* FROM Account\_Types;

-- Retrieve data from Accounts table

SELECT \* FROM Accounts;

-- Retrieve data from Admin\_user table

SELECT \* FROM Admin\_user;

-- Retrieve data from Transactions table

SELECT \* FROM Transactions;

-- Retrieve data from Transaction\_Types table

SELECT \* FROM Transaction\_Types;

-- Insert sample data into Customer\_Types table

INSERT INTO Customer\_Types (Customer\_Types\_code, Customer\_Types\_description)

VALUES (1, 'Type 1'), (2, 'Type 2'), (3, 'Type 3');

-- Insert sample data into Customers table

INSERT INTO Customers (Customers\_ID, Customers\_name, Customers\_phone, Customers\_email, Date\_became\_customer, Login, Password, Other\_details, Customer\_types\_code, Nominee\_name, Nominee\_relation, Nominee\_phone, Nominee\_address)

VALUES

(1, 'John Doe', 1234567890, 'john.doe@example.com', '2023-01-01', 'johndoe', 'password1', 'Details 1', 1, 'Jane Doe', 'Spouse', 9876543210, 'Address 1'),

(2, 'Jane Smith', 9876543210, 'jane.smith@example.com', '2023-02-01', 'janesmith', 'password2', 'Details 2', 2, 'John Smith', 'Spouse', 1234567890, 'Address 2'),

(3, 'David Johnson', 5555555555, 'david.johnson@example.com', '2023-03-01', 'davidjohnson', 'password3', 'Details 3', 3, 'Sarah Johnson', 'Spouse', 1111111111, 'Address 3');

-- Insert sample data into Account\_Types table

INSERT INTO Account\_Types (Account\_Types\_code, Account\_Types\_description)

VALUES ('Type A', 'Account Type A'), ('Type B', 'Account Type B'), ('Type C', 'Account Type C');

-- Insert sample data into Accounts table

INSERT INTO Accounts (Account\_ID, Accounts\_name, Date\_opened, Other\_account\_details, Account\_types\_code, Customer\_ID, Debit, Credit, Balance)

VALUES

(1, 'Account 1', '2023-01-01', 'Details 1', 'Type A', 1, 100.5, 0, 100.5),

(2, 'Account 2', '2023-02-01', 'Details 2', 'Type B', 2, 0, 50.2, -50.2),

(3, 'Account 3', '2023-03-01', 'Details 3', 'Type C', 3, 10.5, 20.3, -9.8);

-- Insert sample data into Admin\_user table

INSERT INTO Admin\_user (Admin\_user\_Login, Admin\_user\_Password)

VALUES ('admin1', 'adminpassword1'), ('admin2', 'adminpassword2'), ('admin3', 'adminpassword3');

-- Insert sample data into Transaction\_Types table

INSERT INTO Transaction\_Types (Transaction\_Types\_code, Transaction\_Types\_description)

VALUES ('Type 1', 'Transaction Type 1'), ('Type 2', 'Transaction Type 2'), ('Type 3', 'Transaction Type 3');

-- Insert sample data into Transactions table

INSERT INTO Transactions (Transaction\_ID, Date, Debit, Credit, Balance, Other\_details, Purchase\_ID, Account\_ID, Transaction\_types\_code)

VALUES

(1, '2023-01-01', 10.5, 0, 10.5, 'Details 1', NULL, 1, 'Type 1'),

(2, '2023-02-01', 0, 20.3, -9.8, 'Details 2', NULL, 2, 'Type 2'),

(3, '2023-03-01', 5.5, 5.5, 0, 'Details 3', NULL, 3, 'Type 3');

-- Retrieve data from Customer\_Types table

SELECT \* FROM Customer\_Types;

go

-- Retrieve data from Account\_Types table

SELECT \* FROM Account\_Types;

go

-- Retrieve data from Accounts table

SELECT \* FROM Accounts;

go

-- Retrieve data from Customers table

SELECT \* FROM Customers;

go

-- Retrieve data from Admin\_user table

SELECT \* FROM Admin\_user;

go

-- Retrieve data from Transactions table

SELECT \* FROM Transactions;

go

-- Retrieve data from Transaction\_Types table

SELECT \* FROM Transaction\_Types;

go

USE BankingDatabase;

GO

-- Create the view to retrieve account details for a customer

CREATE VIEW CustomerAccountDetails AS

SELECT

C.Customers\_ID,

C.Customers\_name,

A.Account\_ID,

A.Accounts\_name,

A.Balance

FROM

Customers AS C

JOIN

Accounts AS A ON C.Customers\_ID = A.Customer\_ID;

go

SELECT \* FROM CustomerAccountDetails;

go

USE BankingDatabase;

GO

USE BankingDatabase;

GO

-- Create a trigger to update the balance of the account after every transaction

CREATE TRIGGER UpdateAccountBalance

ON Transactions

AFTER INSERT, UPDATE

AS

BEGIN

-- Update the balance in the Accounts table based on the transaction

UPDATE A

SET

A.Balance = A.Balance + (I.Credit - I.Debit)

FROM

Accounts AS A

JOIN

inserted AS I ON A.Account\_ID = I.Account\_ID;

END;

go

SELECT \* FROM Accounts;

go

USE BankingDatabase;

GO

-- Create a stored procedure to get the account balance for a specific customer

CREATE PROCEDURE GetAccountBalance

@CustomerID INT

AS

BEGIN

SET NOCOUNT ON;

SELECT

A.Account\_ID,

A.Accounts\_name,

A.Balance

FROM

Accounts AS A

WHERE

A.Customer\_ID = @CustomerID;

END;

go

USE BankingDatabase;

GO

-- Execute the stored procedure to get account balances for a specific customer

EXEC GetAccountBalance @CustomerID = 1;

go

DECLARE @name VARCHAR(50) -- database name

DECLARE @path VARCHAR(256) -- path for backup files

DECLARE @fileName VARCHAR(256) -- filename for backup

DECLARE @fileDate VARCHAR(20) -- used for file name

SET @path = 'C:\Backup\'

SELECT @fileDate = CONVERT(VARCHAR(20), GETDATE(), 112)

DECLARE db\_cursor CURSOR FOR

SELECT name

FROM MASTER.dbo.sysdatabases

WHERE name NOT IN ('master','model','msdb','tempdb')

OPEN db\_cursor

FETCH NEXT FROM db\_cursor INTO @name

WHILE @@FETCH\_STATUS = 0

BEGIN

SET @fileName = @path + @name + '\_' + @fileDate + '.BAK'

BACKUP DATABASE @name TO DISK = @fileName

FETCH NEXT FROM db\_cursor INTO @name

END

CLOSE db\_cursor

DEALLOCATE db\_cursor

go

USE master; -- Switch to the master database

-- Specify the restore file path and name

DECLARE @backupFilePath NVARCHAR(500) = 'C:\Backup\YourBackupFile.bak';

-- Specify the logical and physical file names for the database

DECLARE @databaseName NVARCHAR(128) = 'YourDatabaseName';

DECLARE @dataFilePath NVARCHAR(500) = 'C:\Data\YourDatabaseName.mdf';

DECLARE @logFilePath NVARCHAR(500) = 'C:\Logs\YourDatabaseName.ldf';

-- Restore the database

RESTORE DATABASE @databaseName

FROM DISK = @backupFilePath

WITH

MOVE 'LogicalDataFileName' TO @dataFilePath,

MOVE 'LogicalLogFileName' TO @logFilePath,

REPLACE; -- Use REPLACE to overwrite the existing database if it already exists

go

--- Remote login

EXECUTE AS LOGIN = 'RemoteLoginName'; -- Replace 'RemoteLoginName' with the actual remote login name

--- To show the current transaction in SQL, you can use the @@TRANCOUNT system variable.

SELECT @@TRANCOUNT;

go

---To rollback a transaction in SQL, you can use the ROLLBACK statement.

ROLLBACK;

go

---To commit a transaction in SQL, you can use the COMMIT statement.

COMMIT;