TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP
KHOA ĐIỆN TỬ
BỘ MÔN: CÔNG NGHỆ THÔNG TIN

BÀI TẬP LỚN

HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU

Sinh viên: ……ĐÀO NGUYỄN PHÚ QUÝ……
Lớp: ……K57KMT……
Giáo viên hướng dẫn: .ĐỖ DUY COPS……

Thái Nguyên – 2024

TRƯỜNG ĐHKTCN

KHOA ĐIỆN TỬ

CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập - Tự do - Hạnh phúc

BÀI TẬP LỚN

MÔN HỌC.....

BỘ MÔN .....

Sinh viên:.....

Lớp: ..... *Ngành:* .....

Giáo viên hướng dẫn:.....

Ngày giao đề..... *Ngày hoàn thành* .....

Tên đề tài.....

.....

Yêu cầu.....

.....

.....

.....

.....

.....

.....

.....

GIÁO VIÊN HƯỚNG DẪN

(Ký và ghi rõ họ tên)

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

Thái Nguyên, ngày….tháng…..năm 20....

GIÁO VIÊN HƯỚNG DẪN

(Ký ghi rõ họ tên)

Lời Nói Đầu

Trong thời đại ngày nay, khi các giao dịch tài chính và hoạt động ngân hàng ngày càng trở nên phức tạp và đa dạng hơn, việc xây dựng một hệ thống quản lý hiệu quả và toàn diện là điều vô cùng quan trọng. Các ngân hàng cần một nền tảng vững chắc, bao gồm một cơ sở dữ liệu được thiết kế tối ưu và các chức năng phù hợp để đáp ứng nhu cầu quản lý và vận hành hàng ngày.

Báo cáo này nhằm cung cấp một giải pháp toàn diện cho việc thiết kế cơ sở dữ liệu và các chức năng liên quan đến một ứng dụng Quản lý Ngân hàng. Với mục tiêu tạo ra một hệ thống quản lý hiệu quả, đáng tin cậy và dễ sử dụng, báo cáo sẽ trình bày chi tiết các bước thiết kế cơ sở dữ liệu, bao gồm tạo cơ sở dữ liệu và các bảng cần thiết, định nghĩa các functions và stored procedures để xử lý nghiệp vụ, thêm dữ liệu mẫu, và kiểm tra chức năng của các stored procedures và functions.

Bằng cách tận dụng các công nghệ và kỹ thuật hiện đại trong lĩnh vực cơ sở dữ liệu, báo cáo này sẽ đưa ra một giải pháp toàn diện cho việc quản lý các hoạt động liên quan đến khách hàng, tài khoản, giao dịch, khoản vay, chi nhánh và nhân viên của ngân hàng. Các chức năng được cung cấp bởi các stored procedures và functions sẽ giúp xử lý dữ liệu một cách nhanh chóng và chính xác, đồng thời cung cấp các báo cáo và thống kê cần thiết cho việc quản lý ngân hàng.

Với mong muốn đóng góp vào sự phát triển và hiện đại hóa của hệ thống ngân hàng, báo cáo này sẽ là một tài liệu hữu ích cho các chuyên gia trong lĩnh vực công nghệ thông tin, các nhà quản lý ngân hàng, và bất kỳ ai quan tâm đến việc xây dựng một hệ thống quản lý ngân hàng hiệu quả và đáng tin cậy.

1. Giới thiệu

Báo cáo này cung cấp một tổng quan chi tiết về thiết kế cơ sở dữ liệu và các chức năng liên quan đến ứng dụng Quản lý Ngân hàng. Ứng dụng này được thiết kế để quản lý các hoạt động của một ngân hàng, bao gồm quản lý khách hàng, tài khoản, giao dịch, khoản vay, chi nhánh và nhân viên.

Trong báo cáo này, chúng ta sẽ tìm hiểu về các bước thiết kế cơ sở dữ liệu, bao gồm tạo cơ sở dữ liệu và các bảng cần thiết, định nghĩa các functions và stored procedures để xử lý nghiệp vụ, thêm dữ liệu mẫu, và kiểm tra chức năng của các stored procedures và functions.

2. Tạo Cơ sở Dữ liệu và Các Bảng

2.1. Tạo Cơ sở Dữ liệu

Đầu tiên, chúng ta tạo cơ sở dữ liệu "QuanLyNganHang" bằng câu lệnh sau:

```
CREATE DATABASE [QuanLyNganHang];
USE [QuanLyNganHang];
GO
```

2.2. Tạo Bảng Khách Hàng (Customers)

Bảng Khách Hàng (Customers) chứa thông tin cơ bản về khách hàng của ngân hàng, bao gồm mã khách hàng (CustomerID), tên, số điện thoại, địa chỉ, email và ngày sinh.

```
CREATE TABLE Customers (
    CustomerID VARCHAR(10) PRIMARY KEY,
    Name NVARCHAR(100),
    PhoneNumber VARCHAR(15),
    Address NVARCHAR(200),
    Email NVARCHAR(100),
    DateOfBirth DATE
);
```

2.3. Tạo Bảng Chi Nhánh (Branches)

Bảng Chi Nhánh (Branches) chứa thông tin về các chi nhánh của ngân hàng, bao gồm mã chi nhánh (BranchID), tên chi nhánh, địa chỉ và số điện thoại.

```
CREATE TABLE Branches (
    BranchID VARCHAR(10) PRIMARY KEY,
    BranchName NVARCHAR(100),
    BranchAddress NVARCHAR(200),
    PhoneNumber VARCHAR(15)
);
```

2.4. Tạo Bảng Tài Khoản (Accounts)

Bảng Tài Khoản (Accounts) chứa thông tin về các tài khoản của khách hàng tại ngân hàng, bao gồm mã tài khoản (AccountID), mã khách hàng (CustomerID), số dư (Balance), loại tài khoản (AccountType) và ngày tạo (CreatedDate).

```
CREATE TABLE Accounts (
    AccountID VARCHAR(10) PRIMARY KEY,
    CustomerID VARCHAR(10) FOREIGN KEY REFERENCES Customers(CustomerID),
    Balance DECIMAL(18, 2),
    AccountType NVARCHAR(50),
    CreatedDate DATE
);
```

2.5. Tạo Bảng Giao Dịch (Transactions)

Bảng Giao Dịch (Transactions) chứa thông tin về các giao dịch được thực hiện trên các tài khoản, bao gồm mã giao dịch (TransactionID), mã tài khoản (AccountID), số tiền (Amount), loại giao dịch (TransactionType), ngày giao dịch (TransactionDate) và mô tả (Description).

```
CREATE TABLE Transactions (
    TransactionID VARCHAR(10) PRIMARY KEY,
    AccountID VARCHAR(10) FOREIGN KEY REFERENCES Accounts(AccountID),
    Amount DECIMAL(18, 2),
    Trans actionType NVARCHAR(50), TransactionDate DATE, Description
NVARCHAR(200) );
```

2.6. Tạo Bảng Nhân Viên (Employees)

Bảng Nhân Viên (Employees) chứa thông tin về các nhân viên của ngân hàng, bao gồm mã nhân viên (EmployeeID), tên, vị trí công việc (Position), mã chi nhánh (BranchID), số điện thoại và email.

```
CREATE TABLE Employees (
    EmployeeID VARCHAR(10) PRIMARY KEY,
    Name NVARCHAR(100),
    Position NVARCHAR(50),
    BranchID VARCHAR(10) FOREIGN KEY REFERENCES Branches(BranchID),
    PhoneNumber VARCHAR(15),
    Email NVARCHAR(100)
);
```

2.7. Tạo Bảng Khoản Vay (Loans)

Bảng Khoản Vay (Loans) chứa thông tin về các khoản vay của khách hàng từ ngân hàng, bao gồm mã khoản vay (LoanID), mã khách hàng (CustomerID), số tiền vay (Amount), loại khoản vay (LoanType), ngày bắt đầu (StartDate), ngày kết thúc (EndDate) và lãi suất (InterestRate).

```
CREATE TABLE Loans (
    LoanID VARCHAR(10) PRIMARY KEY,
    CustomerID VARCHAR(10) FOREIGN KEY REFERENCES Customers(CustomerID),
    Amount DECIMAL(18, 2),
    LoanType NVARCHAR(50),
    StartDate DATE,
    EndDate DATE,
    InterestRate DECIMAL(5, 2)
);
```

3. Thiết kế Functions

3.1. Tính Số Dư của Một Tài Khoản (FN_GetAccountBalance)

Function này được sử dụng để tính số dư hiện tại của một tài khoản dựa trên mã tài khoản (AccountID) được cung cấp.

```
CREATE FUNCTION FN_GetAccountBalance
(
   @AccountID VARCHAR(10)
)
RETURNS DECIMAL(18, 2)
AS
BEGIN
   DECLARE @Balance DECIMAL(18, 2);
   SELECT @Balance = Balance FROM Accounts WHERE AccountID = @AccountID;
   RETURN @Balance;
END
GO
```

3.2. Tính Tổng Giao Dịch trong Một Ngày (FN_GetDailyTransactionTotal)

Function này được sử dụng để tính tổng số tiền giao dịch (gồm nạp tiền và rút tiền) của một tài khoản trong một ngày cụ thể.

```
CREATE FUNCTION FN_GetDailyTransactionTotal
(
   @AccountID VARCHAR(10),
   @TransactionDate DATE
)
RETURNS DECIMAL(18, 2)
AS
BEGIN
   DECLARE @Total DECIMAL(18, 2);
   SELECT @Total = SUM(Amount)
   FROM Transactions
   WHERE AccountID = @AccountID
   AND CONVERT(DATE,  TransactionDate) = @TransactionDate
   AND TransactionType IN ('Deposit', 'Withdrawal');

   RETURN @Total;
END
GO
```

3.3. Tính Tổng Số Dư của Tất Cả Tài Khoản (FN_totalBalance)

Function này được sử dụng để tính tổng số dư của tất cả các tài khoản trong ngân hàng.

```
CREATE FUNCTION FN_totalBalance()
RETURNS DECIMAL(18, 2)
AS
```

```
BEGIN
   DECLARE @totalBalance DECIMAL(18, 2);
   SELECT @totalBalance = SUM(Balance) FROM Accounts;
   RETURN @totalBalance;
END;
GO
```

3.4. Tính Tổng Giao Dịch trong Khoảng Thời Gian (FN_totalTransactions)

Function này được sử dụng để tính tổng số tiền giao dịch trong một khoảng thời gian nhất định.

```
CREATE FUNCTION FN_totalTransactions(@StartDate DATE, @EndDate DATE)
RETURNS DECIMAL(18, 2)
AS
BEGIN
   DECLARE @totalTransactions DECIMAL(18, 2);
   SELECT @totalTransactions = SUM(Amount)
   FROM Transactions
   WHERE TransactionDate BETWEEN @StartDate AND @EndDate;
   RETURN @totalTransactions;
END;
GO
```

3.5. Tính Tổng Khoản Vay của Một Khách Hàng (FN_totalLoanAmount)

Function này được sử dụng để tính tổng số tiền khoản vay của một khách hàng dựa trên mã khách hàng (CustomerID) được cung cấp.

```
CREATE FUNCTION FN_totalLoanAmount(@CustomerID VARCHAR(10))
RETURNS DECIMAL(18, 2)
AS
BEGIN
   DECLARE @totalLoanAmount DECIMAL(18, 2);
   SELECT @totalLoanAmount = SUM(Amount)
   FROM Loans
   WHERE CustomerID = @CustomerID;
   RETURN @totalLoanAmount;
END;
GO
```

4. Thiết kế Stored Procedures

4.1. Thêm Khách Hàng (SP_addCustomer)

Stored Procedure này được sử dụng để thêm một khách hàng mới vào bảng Customers.

```
CREATE PROCEDURE SP_addCustomer
   @CustomerID VARCHAR(10),
   @Name NVARCHAR(100),
   @PhoneNumber VARCHAR(15),
```

```
    @Address NVARCHAR(200),
    @Email NVARCHAR(100),
    @DateOfBirth DATE
AS
BEGIN
    INSERT INTO Customers (CustomerID, Name, PhoneNumber, Address, Email,
DateOfBirth)
    VALUES (@CustomerID, @Name, @PhoneNumber, @Address, @Email,
@DateOfBirth);
END;
GO
```

4.2. Thêm Tài Khoản (SP_addAccount)

Stored Procedure này được sử dụng để thêm một tài khoản mới vào bảng Accounts.

```
CREATE PROCEDURE SP_addAccount
    @AccountID VARCHAR(10),
    @CustomerID VARCHAR(10),
    @Balance DECIMAL(18, 2),
    @AccountType NVARCHAR(50),
    @CreatedDate DATE
AS
BEGIN
    INSERT INTO Accounts (AccountID, CustomerID, Balance, AccountType,
CreatedDate)
    VALUES (@AccountID, @CustomerID, @Balance, @AccountType, @CreatedDate);
END;
GO
```

4.3. Thêm Giao Dịch (SP_addTransaction)

Stored Procedure này được sử dụng để thêm một giao dịch mới vào bảng Transactions.

```
CREATE PROCEDURE SP_addTransaction
    @TransactionID VARCHAR(10),
    @AccountID VARCHAR(10),
    @Amount DECIMAL(18, 2),
    @TransactionType NVARCHAR(50),
    @TransactionDate DATE,
    @Description NVARCHAR(200)
AS
BEGIN
    INSERT INTO Transactions (TransactionID, AccountID, Amount, TransactionType,
TransactionDate, Description)
    VALUES (@TransactionID, @AccountID, @Amount, @TransactionType,
@TransactionDate, @Description);
END;
GO
```

4.4. Thêm Chi Nhánh (SP_addBranch)

Stored Procedure này được sử dụng để thêm một chi nhánh mới vào bảng Branches.

```
CREATE PROCEDURE SP_addBranch
   @BranchID VARCHAR(10),
   @BranchName NVARCHAR(100),
   @ BranchAddress NVARCHAR(200), @PhoneNumber VARCHAR(15) AS BEGIN
INSERT INTO Branches (BranchID, BranchName, BranchAddress, PhoneNumber)
VALUES (@BranchID, @BranchName, @BranchAddress, @PhoneNumber);
END;
GO
```

4.5. Thêm Nhân Viên (SP_addEmployee)

Stored Procedure này được sử dụng để thêm một nhân viên mới vào bảng Employees.

```
CREATE PROCEDURE SP_addEmployee
@EmployeeID VARCHAR(10),
@Name NVARCHAR(100),
@Position NVARCHAR(50),
@BranchID VARCHAR(10),
@PhoneNumber VARCHAR(15),
@Email NVARCHAR(100)
AS
BEGIN
INSERT INTO Employees (EmployeeID, Name, Position, BranchID, PhoneNumber,
Email) VALUES (@EmployeeID, @Name, @Position, @BranchID, @PhoneNumber,
@Email);
END;
GO
```

4.6. Thêm Khoản Vay (SP_addLoan)

Stored Procedure này được sử dụng để thêm một khoản vay mới vào bảng Loans.

```
CREATE PROCEDURE SP_addLoan
   @LoanID VARCHAR(10),
   @CustomerID VARCHAR(10),
   @Amount DECIMAL(18, 2),
   @LoanType NVARCHAR(50),
   @StartDate DATE,
   @EndDate DATE,
   @InterestRate DECIMAL(5, 2)
AS
BEGIN
   INSERT INTO Loans (LoanID, CustomerID, Amount, LoanType, StartDate, EndDate,
InterestRate)
   VALUES (@LoanID, @CustomerID, @Amount, @LoanType, @StartDate, @EndDate,
@InterestRate);
END;
GO
```

4.7. Nạp Tiền (SP_DepositMoney)

Stored Procedure này được sử dụng để nạp tiền vào một tài khoản và ghi lại giao dịch nạp tiền trong bảng Transactions.

```
CREATE PROCEDURE SP_DepositMoney
   @TransactionID VARCHAR(10),
   @AccountID VARCHAR(10),
   @Amount DECIMAL(18, 2),
   @Description NVARCHAR(500)
AS
BEGIN
   UPDATE Accounts
   SET Balance = Balance + @Amount
   WHERE AccountID = @AccountID;

   INSERT INTO Transactions (TransactionID, AccountID, Amount, TransactionType,
TransactionDate, Description)
   VALUES (@TransactionID, @AccountID, @Amount, 'Deposit', GETDATE(),
@Description);
END
GO
```

4.8. Rút Tiền (SP_WithdrawMoney)

Stored Procedure này được sử dụng để rút tiền từ một tài khoản và ghi lại giao dịch rút tiền trong bảng Transactions. Nó cũng kiểm tra xem số dư trong tài khoản có đủ để rút tiền hay không.

```
CREATE PROCEDURE SP_WithdrawMoney
   @TransactionID VARCHAR(10),
   @AccountID VARCHAR(10),
   @Amount DECIMAL(18, 2),
   @Description NVARCHAR(500)
AS
BEGIN
   DECLARE @CurrentBalance DECIMAL(18, 2);
   SELECT @CurrentBalance = Balance FROM Accounts WHERE AccountID =
@AccountID;

   IF @CurrentBalance >= @Amount
   BEGIN
      UPDATE Accounts
      SET Balance = Balance - @Amount
      WHERE AccountID = @AccountID;

      INSERT INTO Transactions (TransactionID, AccountID, Amount, TransactionType,
TransactionDate, Description)
      VALUES (@TransactionID, @AccountID, @Amount, 'Withdrawal', GETDATE(),
@Description);
   END
   ELSE
   BEGIN
      RAISERROR ('Insufficient funds', 16, 1);
   END
END
```

```
GO
```

5. Thêm Dữ Liệu Mẫu

5.1. Thêm Dữ Liệu Mẫu vào Bảng Khách Hàng

```
INSERT INTO Customers (CustomerID, Name, PhoneNumber, Address, Email,
DateOfBirth)
VALUES
   ('C001', 'Nguyen Van A', '0912345678', '123 Phan Dinh Phung, Hanoi',
'a@example.com', '1990-01-01'),
   ('C002', 'Tran Thi B', '0987654321', '456 Le Loi, Ho Chi Minh City', 'b@example.com',
'1985-05-10'),
   ('C003', 'Pham Van C', '0911223344', '789 Nguyen Hue, Da Nang', 'c@example.com',
'1995-12-20');
GO
```

5.2. Thêm Dữ Liệu Mẫu vào Bảng Tài Khoản

```
INSERT INTO Accounts (AccountID, CustomerID, Balance, AccountType, CreatedDate)
VALUES
   ('A001', 'C001', 1000000, 'Saving', '2024-06-19'),
   ('A002', 'C002', 500000, 'Checking', '2024-06-20'),
   ('A003', 'C003', 2000000, 'Saving', '2024-06-21');
GO
```

5.3. Thêm Dữ Liệu Mẫu vào Bảng Giao Dịch

```
INSERT INTO Transactions (TransactionID, AccountID, Amount, TransactionType,
TransactionDate, Description)
VALUES
   ('T001', 'A001', 500000, 'Deposit', '2024-06-19', 'Initial deposit'),
   ('T002', 'A002', 200000, 'Withdrawal', '2024-06-20', 'ATM withdrawal'),
   ('T003', 'A003', 1000000, 'Transfer', '2024-06-21', 'Transfer to another account');
GO
```

5.4. Thêm Dữ Liệu Mẫu vào Bảng Chi Nhánh

```
INSERT INTO Branches (BranchID, BranchName, BranchAddress, PhoneNumber)
VALUES
   ('B001', 'Branch 1', '123 Tran Hung Dao, Hanoi', '0911223344'),
   ('B002', 'Branch 2', '456 Le Loi, Ho Chi Minh City', '0987654321'),
   ('B003', 'Branch 3', '789 Nguyen Hue, Da Nang', '0912345678');
GO
```

5.5. Thêm Dữ Liệu Mẫu vào Bảng Nhân Viên

```
INSERT INTO Employees (EmployeeID, Name, Position, BranchID, PhoneNumber,
Email)
```

```
VALUES
   ('E001', 'Le Van B', 'Manager', 'B001', '0911555777', 'b@example.com'),
   ('E002', 'Tran Thi D', 'Teller', 'B002', '0988777666', 'd@example.com'),
   ('E003', 'Nguyen Van E', 'Customer Service', 'B003', '0911222333', 'e@example.com');
```

5.6. Thêm Dữ Liệu Mẫu vào Bảng Khoản Vay

```
INSERT INTO Loans (LoanID, CustomerID, Amount, LoanType, StartDate, EndDate,
InterestRate)
VALUES
   ('L001', 'C001', 50000000, 'Home Loan', '2024-06-19', '2034-06-19', 5.5),
   ('L002', 'C002', 20000000, 'Car Loan', '2024-06-20', '2030-06-20', 4.75),
   ('L003', 'C003', 10000000, 'Personal Loan', '2024-06-21', '2029-06-21', 6.0);
GO
```

6. Kiểm tra Chức Năng

6.1. Kiểm tra Stored Procedures

Sau đây là các script để kiểm tra chức năng của các Stored Procedures:

```
-- Test SP_addCustomer
EXEC SP_addCustomer 'C003', 'Nguyen Van C', '0912345689', '123 Hanoi',
'c@example.com', '1990-05-01';
SELECT * FROM Customers WHERE CustomerID = 'C003';
 -- Test SP_addAccount
EXEC SP_addAccount 'A004', 'C002', 1000000, 'Saving', '2024-06-19';
SELECT * FROM Accounts WHERE AccountID = 'A004';
-- Test SP_addTransaction
EXEC SP_addTransaction 'T001', 'A001', 500000, 'Deposit', '2024-06-19', 'Initial deposit';
SELECT * FROM Transactions WHERE TransactionID = 'T001';
-- Test SP_addBranch
EXEC SP_addBranch 'B004', 'Branch 1', '123 Tran Hung Dao, Hanoi', '0911223344';
SELECT * FROM Branches WHERE BranchID = 'B004';
-- Test SP_addEmployee
EXEC SP_addEmployee 'E004', 'Le Van D', 'Manager', 'B004', '0911555777',
'd@example.com';
SELECT * FROM Employees WHERE EmployeeID = 'E004';
-- Test SP_addLoan
EXEC SP_addLoan 'L004', 'C003', 50000000, 'Home Loan', '2024-06-19', '2029-06-19',
5.5;
SELECT * FROM Loans WHERE LoanID = 'L004';
-- Test SP_DepositMoney EXEC dbo.SP_DepositMoney 'T002', 'A001', 1000.00, 'Nap tien
vao tai khoan';
-- Test SP_WithdrawMoney
EXEC dbo.SP_WithdrawMoney 'T003', 'A001', 52000.00, 'Rut tien tu tai khoan';
SELECT * FROM Transactions WHERE AccountID = 'A001';
```

6.2. Kiểm tra Functions

Sau đây là các script để kiểm tra chức năng của các Functions:

```
-- Test FN_GetAccountBalance
SELECT dbo.FN_GetAccountBalance('A001') AS CurrentBalance;

-- Test FN_GetDailyTransactionTotal
SELECT dbo.FN_GetDailyTransactionTotal('A001', '2024-06-19') AS
DailyTransactionTotal;

-- Test FN_totalBalance
SELECT dbo.FN_totalBalance() AS TotalBalance;

-- Test FN_totalTransactions
SELECT dbo.FN_totalTransactions('2024-06-01', '2024-06-30') AS TotalTransactions;

-- Test FN_totalLoanAmount
SELECT dbo.FN_totalLoanAmount('C001') AS TotalLoanAmount;
```

7. Kết luận

Báo cáo này cung cấp một tổng quan chi tiết về thiết kế cơ sở dữ liệu và các chức năng liên quan đến ứng dụng Quản lý Ngân hàng. Bao gồm các bước tạo cơ sở dữ liệu và các bảng cần thiết, định nghĩa các functions và stored procedures để xử lý nghiệp vụ, thêm dữ liệu mẫu, và kiểm tra chức năng của các stored procedures và functions.
Với thiết kế này, ứng dụng Quản lý Ngân hàng có thể quản lý hiệu quả các hoạt động liên quan đến khách hàng, tài khoản, giao dịch, khoản vay, chi nhánh và nhân viên. Các chức năng được cung cấp bởi các stored procedures và functions sẽ giúp xử lý dữ liệu một cách nhanh chóng và chính xác, đồng thời cung cấp các báo cáo và thống kê cần thiết cho việc quản lý ngân hàng.

8. Mã QR để tới trang github