**Exercise 3: Stored Procedures**

**Scenario 1:** The bank needs to process monthly interest for all savings accounts.

**Question:** Write a stored procedure **ProcessMonthlyInterest** that calculates and updates the balance of all savings accounts by applying an interest rate of 1% to the current balance.

**Answer:**

DELIMITER //

CREATE PROCEDURE ProcessMonthlyInterest()

BEGIN

UPDATE Accounts

SET Balance = Balance \* 1.01

WHERE AccountType = 'Savings';

END //

DELIMITER ;

**Scenario 2:** The bank wants to implement a bonus scheme for employees based on their performance.

**Question:** Write a stored procedure **UpdateEmployeeBonus** that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.

**Answer:**

DELIMITER //

CREATE PROCEDURE UpdateEmployeeBonus(

IN departmentName VARCHAR(50),

IN bonusPercentage DECIMAL(5,2)

)

BEGIN

UPDATE Employee

SET Salary = Salary + (Salary \* (bonusPercentage / 100))

WHERE Department = departmentName;

END //

DELIMITER ;

**Scenario 3:** Customers should be able to transfer funds between their accounts.

**Question:** Write a stored procedure **TransferFunds** that transfers a specified amount from one account to another, checking that the source account has sufficient balance before making the transfer.

**Answer:**

DELIMITER //

CREATE PROCEDURE TransferFunds(

IN sourceAccountID INT,

IN targetAccountID INT,

IN amount DECIMAL(10,2)

)

BEGIN

DECLARE sourceBalance DECIMAL(10,2);

START TRANSACTION;

SELECT Balance INTO sourceBalance

FROM Accounts

WHERE AccountID = sourceAccountID;

IF sourceBalance < amount THEN

-- Rollback transaction and signal error

ROLLBACK;

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Insufficient funds';

ELSE

UPDATE Accounts

SET Balance = Balance - amount

WHERE AccountID = sourceAccountID;

UPDATE Accounts

SET Balance = Balance + amount

WHERE AccountID = targetAccountID;

COMMIT;

END IF;

END //

DELIMITER ;