**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.

**CODE**

DELIMITER //

CREATE PROCEDURE ProcessMonthlyInterest()

BEGIN

    DECLARE done INT DEFAULT 0;

    DECLARE account\_id INT;

    DECLARE balance DECIMAL(10,2);

    DECLARE cur CURSOR FOR SELECT account\_id, balance FROM SavingsAccounts;

    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;

    OPEN cur;

    read\_loop: LOOP

        FETCH cur INTO account\_id, balance;

        IF done THEN

            LEAVE read\_loop;

        END IF;

        -- Apply 1% interest to the current balance

        SET balance = balance + (balance \* 0.01);

        -- Update the account with the new balance

        UPDATE SavingsAccounts SET balance = balance WHERE account\_id = account\_id;

    END LOOP;

    CLOSE cur;

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.

**CODE**

DELIMITER //

CREATE PROCEDURE UpdateEmployeeBonus(IN department\_name VARCHAR(100), IN bonus\_percentage DECIMAL(5,2))

BEGIN

    -- Update the salary by adding the bonus percentage

    UPDATE Employees

    SET salary = salary + (salary \* bonus\_percentage / 100)

    WHERE department = department\_name;

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.

**CODE**

DELIMITER //

CREATE PROCEDURE TransferFunds(IN source\_account\_id INT, IN destination\_account\_id INT, IN amount DECIMAL(10,2))

BEGIN

    DECLARE current\_balance DECIMAL(10,2);

    -- Check the balance of the source account

    SELECT balance INTO current\_balance FROM Accounts WHERE account\_id = source\_account\_id;

    -- Check if the source account has sufficient balance

    IF current\_balance >= amount THEN

        -- Deduct the amount from the source account

        UPDATE Accounts SET balance = balance - amount WHERE account\_id = source\_account\_id;

        -- Add the amount to the destination account

        UPDATE Accounts SET balance = balance + amount WHERE account\_id = destination\_account\_id;

    ELSE

        -- Insufficient funds, raise an error

        SIGNAL SQLSTATE '5000' SET MESSAGE\_TEXT = 'Insufficient funds in the source account';

    END IF;

END //

DELIMITER ;