**Exercise 1: Control Structures**

**Scenario 1: The bank wants to apply a discount to loan interest rates for customers above 60 years old.**

* + **Question: Write a PL/SQL block that loops through all customers, checks their age, and if they are above 60, apply a 1% discount to their current loan interest rates.**

**Scenario 2: A customer can be promoted to VIP status based on their balance.**

* + **Question: Write a PL/SQL block that iterates through all customers and sets a flag IsVIP to TRUE for those with a balance over $10,000.**

**Scenario 3: The bank wants to send reminders to customers whose loans are due within the next 30 days.**

* + **Question: Write a PL/SQL block that fetches all loans due in the next 30 days and prints a reminder message for each customer.**

**ANSWER**

=

**SCENARIO 1:**

**BEGIN**

**FOR cust\_rec IN (SELECT CustomerID, Age, LoanInterestRate FROM Customers) LOOP**

**IF cust\_rec.Age > 60 THEN**

**UPDATE Customers**

**SET LoanInterestRate = LoanInterestRate - (LoanInterestRate \* 0.01)**

**WHERE CustomerID = cust\_rec.CustomerID;**

**END IF;**

**END LOOP;**

**COMMIT;**

**END;**

**/**

**IMPLEMENTATION ON WORKBENCH AS PROCEDURE:**

DELIMITER $$

CREATE PROCEDURE ApplyLoanDiscountForSeniors()

BEGIN

DECLARE done INT DEFAULT 0;

DECLARE custId INT;

DECLARE custAge INT;

DECLARE interestRate DECIMAL(5,2);

DECLARE cur CURSOR FOR

SELECT CustomerID, Age, LoanInterestRate FROM Customers;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;

OPEN cur;

read\_loop: LOOP

FETCH cur INTO custId, custAge, interestRate;

IF done THEN

LEAVE read\_loop;

END IF;

IF custAge > 60 THEN

UPDATE Customers

SET LoanInterestRate = interestRate - (interestRate \* 0.01)

WHERE CustomerID = custId;

END IF;

END LOOP;

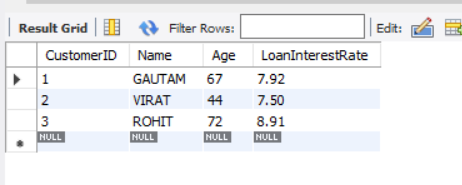
CLOSE cur;

END$$

DELIMITER ;

CALL ApplyLoanDiscountForSeniors();

**OUTPUT:**



**SCENARIO 2**

**BEGIN**

**FOR vip\_rec IN (SELECT CustomerID, Balance FROM Customers) LOOP**

**IF vip\_rec.Balance > 10000 THEN**

**UPDATE Customers**

**SET IsVIP = 'TRUE'**

**WHERE CustomerID = vip\_rec.CustomerID;**

**END IF;**

**END LOOP;**

**COMMIT;**

**END;**

**/**

**IMPLEMENTATION IN WORKBENCH AS PROCEDURE:**

DROP PROCEDURE IF EXISTS PromoteVIPCustomers;

DELIMITER $$

CREATE PROCEDURE PromoteVIPCustomers()

BEGIN

DECLARE done INT DEFAULT 0;

DECLARE custId INT;

DECLARE custBalance DECIMAL(10,2);

DECLARE cur CURSOR FOR

SELECT CustomerID, Balance FROM Customers;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;

OPEN cur;

read\_loop: LOOP

FETCH cur INTO custId, custBalance;

IF done THEN

LEAVE read\_loop;

END IF;

IF custBalance > 10000 THEN

UPDATE Customers

SET IsVIP = TRUE

WHERE CustomerID = custId;

END IF;

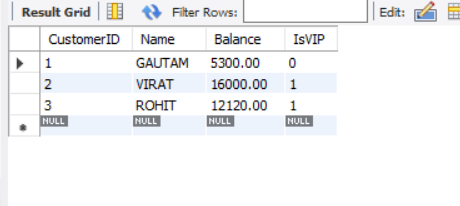
END LOOP;

CLOSE cur;

END$$

DELIMITER ;

**OUTPUT:**



**SCENARIO 3**

**BEGIN**

**FOR loan\_rec IN (**

**SELECT l.LoanID, c.Name, l.DueDate**

**FROM Loans l**

**JOIN Customers c ON l.CustomerID = c.CustomerID**

**WHERE l.DueDate BETWEEN SYSDATE AND SYSDATE + 30**

**) LOOP**

**DBMS\_OUTPUT.PUT\_LINE('📢 Reminder: Loan #' || loan\_rec.LoanID || ' for ' || loan\_rec.Name ||**

**' is due on ' || TO\_CHAR(loan\_rec.DueDate, 'DD-Mon-YYYY'));**

**END LOOP;**

**END;**

**/**

**IMPLEMENTATION ON WORKBENCH :**

SELECT

l.LoanID,

c.Name AS CustomerName,

l.DueDate,

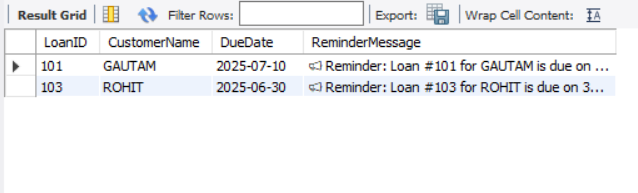
CONCAT('📢 Reminder: Loan #', l.LoanID, ' for ', c.Name,

' is due on ', DATE\_FORMAT(l.DueDate, '%d-%b-%Y')) AS ReminderMessage

FROM Loans l

JOIN Customers c ON l.CustomerID = c.CustomerID

WHERE l.DueDate BETWEEN CURDATE() AND CURDATE() + INTERVAL 30 DAY;

****

**AS I AM WORKING ON MYSQL WORKBENCH SO I HAVE USD SQL EQUIVALENT CODE**

**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.**

**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.**

**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:**

**SCENARIO 1:**

DELIMITER $$

CREATE PROCEDURE ProcessMonthlyInterest()

BEGIN

DECLARE done INT DEFAULT 0;

DECLARE accId INT;

DECLARE accType VARCHAR(20);

DECLARE bal DECIMAL(10,2);

DECLARE cur CURSOR FOR

SELECT AccountID, AccountType, Balance FROM Accounts;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;

OPEN cur;

read\_loop: LOOP

FETCH cur INTO accId, accType, bal;

IF done THEN

LEAVE read\_loop;

END IF;

IF accType = 'Savings' THEN

UPDATE Accounts

SET Balance = bal + (bal \* 0.01)

WHERE AccountID = accId;

END IF;

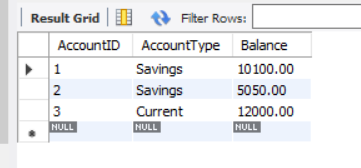
END LOOP;

CLOSE cur;

END$$

DELIMITER ;

**OUTPUT:**



**SCENARIO 2:**

DELIMITER $$

CREATE PROCEDURE UpdateEmployeeBonus(

IN dept\_name VARCHAR(50),

IN bonus\_percent DECIMAL(5,2)

)

BEGIN

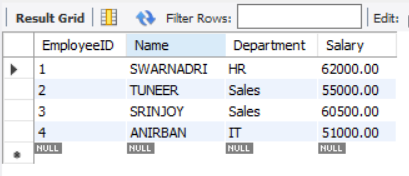
UPDATE Employees

SET Salary = Salary + (Salary \* bonus\_percent / 100)

WHERE Department = dept\_name;

END$$

DELIMITER ;

****

**SCENARIO 3:**

DELIMITER $$

CREATE PROCEDURE TransferFunds(

IN source\_account\_id INT,

IN target\_account\_id INT,

IN amount\_to\_transfer DECIMAL(10,2)

)

BEGIN

DECLARE source\_balance DECIMAL(10,2);

SELECT Balance INTO source\_balance

FROM Accounts

WHERE AccountID = source\_account\_id;

IF source\_balance >= amount\_to\_transfer THEN

UPDATE Accounts

SET Balance = Balance - amount\_to\_transfer

WHERE AccountID = source\_account\_id;

UPDATE Accounts

SET Balance = Balance + amount\_to\_transfer

WHERE AccountID = target\_account\_id;

ELSE

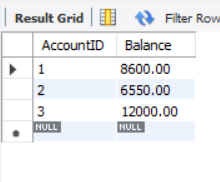
SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = ' Transfer failed: Insufficient funds.';

END IF;

END$$

DELIMITER ;

****