Exercise – 1

DELIMITER $$

CREATE PROCEDURE ApplySeniorLoanDiscount()

BEGIN

DECLARE done INT DEFAULT FALSE;

DECLARE v\_LoanID INT;

DECLARE v\_InterestRate DECIMAL(5,2);

DECLARE v\_DOB DATE;

DECLARE v\_CustomerID INT;

DECLARE cur CURSOR FOR

SELECT l.LoanID, l.InterestRate, c.DOB, c.CustomerID

FROM Loans l

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

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

OPEN cur;

read\_loop: LOOP

FETCH cur INTO v\_LoanID, v\_InterestRate, v\_DOB, v\_CustomerID;

IF done THEN

LEAVE read\_loop;

END IF;

IF TIMESTAMPDIFF(YEAR, v\_DOB, CURDATE()) > 60 THEN

UPDATE Loans

SET InterestRate = v\_InterestRate - 1

WHERE LoanID = v\_LoanID;

END IF;

END LOOP;

CLOSE cur;

END$$

DELIMITER ;

-- To run:

CALL ApplySeniorLoanDiscount();

A screenshot of a computer

AI-generated content may be incorrect.

DELIMITER $$

CREATE PROCEDURE PromoteVIPCustomers()

BEGIN

UPDATE Customers

SET IsVIP = TRUE

WHERE Balance > 10000;

UPDATE Customers

SET IsVIP = FALSE

WHERE Balance <= 10000;

END$$

DELIMITER ;

-- To run:

CALL PromoteVIPCustomers();

A screenshot of a computer

AI-generated content may be incorrect.

DELIMITER $$

CREATE PROCEDURE SendLoanReminders()

BEGIN

DECLARE done INT DEFAULT FALSE;

DECLARE v\_Name VARCHAR(100);

DECLARE v\_EndDate DATE;

DECLARE v\_LoanID INT;

DECLARE cur CURSOR FOR

SELECT c.Name, l.EndDate, l.LoanID

FROM Loans l

JOIN Customers c ON l.CustomerID = c.CustomerID

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

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

OPEN cur;

read\_loop: LOOP

FETCH cur INTO v\_Name, v\_EndDate, v\_LoanID;

IF done THEN

LEAVE read\_loop;

END IF;

SELECT CONCAT('Reminder: Loan ID ', v\_LoanID, ' for ', v\_Name,

' is due on ', v\_EndDate) AS ReminderMessage;

END LOOP;

CLOSE cur;

END$$

DELIMITER ;

-- To run:

CALL SendLoanReminders();

A screenshot of a computer

AI-generated content may be incorrect.