**Exercise 1: Control Structures**

**Solution:**

Scenario 1:

DECLARE

CURSOR cur\_customers IS

SELECT CustomerID, InterestRate

FROM Loans

WHERE CustomerID IN (SELECT CustomerID FROM Customers WHERE Age > 60);

BEGIN

FOR rec IN cur\_customers LOOP

UPDATE Loans

SET InterestRate = InterestRate - 1

WHERE CustomerID = rec.CustomerID;

END LOOP;

COMMIT;

END;

Scenario 2:

DECLARE

CURSOR cur\_customers IS

SELECT CustomerID

FROM Customers

WHERE Balance > 10000;

BEGIN

FOR rec IN cur\_customers LOOP

UPDATE Customers

SET IsVIP = 'TRUE'

WHERE CustomerID = rec.CustomerID;

END LOOP;

COMMIT;

END;

Scenario 3:

DECLARE

CURSOR cur\_due\_loans IS

SELECT LoanID, CustomerID, DueDate

FROM Loans

WHERE DueDate BETWEEN SYSDATE AND SYSDATE + 30;

v\_customer\_name VARCHAR2(100);

BEGIN

FOR rec IN cur\_due\_loans LOOP

SELECT Name INTO v\_customer\_name FROM Customers WHERE CustomerID = rec.CustomerID;

DBMS\_OUTPUT.PUT\_LINE('Reminder: Dear ' || v\_customer\_name ||

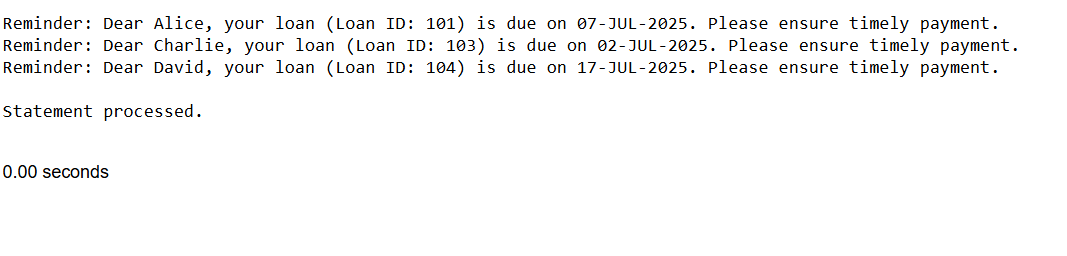
', your loan (Loan ID: ' || rec.LoanID ||

') is due on ' || TO\_CHAR(rec.DueDate, 'DD-MON-YYYY') || '. Please ensure timely payment.');

END LOOP;

END;

**Output:**

****

**Exercise 3: Stored Procedures**

**Solution:**

Scenario 1:

*ProcessedMonthlyInterest*:

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS

BEGIN

UPDATE SavingsAccounts

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

COMMIT;

END;

Scenario 2:

*UpdateEmployeeBonus:*

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

p\_DepartmentID IN NUMBER,

p\_BonusPercent IN NUMBER

) AS

BEGIN

UPDATE Employees

SET Salary = Salary + (Salary \* p\_BonusPercent / 100)

WHERE DepartmentID = p\_DepartmentID;

COMMIT;

END;

Scenario 3:

*Transfer Funds:*

CREATE OR REPLACE PROCEDURE TransferFunds (

p\_SourceAccountID IN NUMBER,

p\_DestinationAccountID IN NUMBER,

p\_Amount IN NUMBER

) AS

v\_SourceBalance NUMBER;

BEGIN

SELECT Balance INTO v\_SourceBalance

FROM Accounts

WHERE AccountID = p\_SourceAccountID

FOR UPDATE;

IF v\_SourceBalance >= p\_Amount THEN

UPDATE Accounts

SET Balance = Balance - p\_Amount

WHERE AccountID = p\_SourceAccountID;

UPDATE Accounts

SET Balance = Balance + p\_Amount

WHERE AccountID = p\_DestinationAccountID;

COMMIT;

ELSE

RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient balance in source account.');

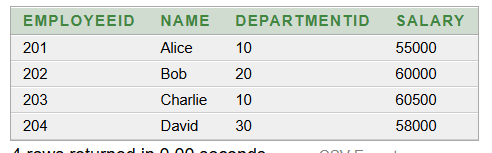
END IF;

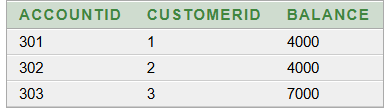
END;

**Output:**

**Scenario 1:**

****

**Scenario 2:**

**Scenario 3:**