Exercise 1: Setting Up Junit

**Scenario 1:**

BEGIN

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

IF customer\_rec.Age > 60 THEN

UPDATE Customers

SET LoanInterestRate = LoanInterestRate - 1

WHERE CustomerID = customer\_rec.CustomerID;

DBMS\_OUTPUT.PUT\_LINE('dis to Customer ID: ' || customer\_rec.CustomerID);

END IF;

END LOOP;

COMMIT;

END;

/

Output:

dis to Customer ID: 23

dis to Customer ID: 45

dis to Customer ID: 34

**Scenario 2:**

BEGIN

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

IF customer\_rec.Balance > 10000 THEN

UPDATE Customers

SET IsVIP = 'TRUE'

WHERE CustomerID = customer\_rec.CustomerID;

DBMS\_OUTPUT.PUT\_LINE('Promoted to VIP: Customer ID: ' || customer\_rec.CustomerID);

END IF;

END LOOP;

COMMIT;

END;

/

Output:

Promoted to VIP: Customer ID: 12

Promoted to VIP: Customer ID: 35

Promoted to VIP: Customer ID: 45  
  
**Scenario 3:**

BEGIN

FOR loan\_rec IN (SELECT CustomerID, LoanID, DueDate FROM Loans WHERE DueDate <= SYSDATE + 30) LOOP

DBMS\_OUTPUT.PUT\_LINE('Reminder: Loan ' || loan\_rec.LoanID || ' for Customer ' || loan\_rec.CustomerID || ' is due on ' || TO\_CHAR(loan\_rec.DueDate, 'DD-MON-YYYY'));

END LOOP;

END;

/  
  
Output:

Reminder: Loan L201 for Customer 101 is due on 15-JUL-2025

Reminder: Loan L205 for Customer 106 is due on 22-JUL-2025

Reminder: Loan L210 for Customer 109 is due on 23-JUL-2025

Exercise 3: Stored Procedures

**Scenario 1:**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

FOR acc\_rec IN (SELECT AccountID, Balance FROM Accounts WHERE AccountType = 'SAVINGS') LOOP

UPDATE Accounts

SET Balance = Balance + (Balance \* 0.01)

WHERE AccountID = acc\_rec.AccountID;

DBMS\_OUTPUT.PUT\_LINE('Interest applied to Account ID: ' || acc\_rec.AccountID);

END LOOP;

COMMIT;

END;

/

Output:

Interest applied to Account ID: 1001

Interest applied to Account ID: 1003

Interest applied to Account ID: 1007

Interest applied to Account ID: 1010

**Scenario 2:**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

dept\_id IN NUMBER,

bonus\_percent IN NUMBER

) IS

BEGIN

FOR emp\_rec IN (SELECT EmployeeID, Salary FROM Employees WHERE DepartmentID = dept\_id) LOOP

UPDATE Employees

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

WHERE EmployeeID = emp\_rec.EmployeeID;

DBMS\_OUTPUT.PUT\_LINE('Bonus applied to Employee ID: ' || emp\_rec.EmployeeID);

END LOOP;

COMMIT;

END;

/

Output:

Bonus applied to Employee ID: 201

Bonus applied to Employee ID: 203

Bonus applied to Employee ID: 207

**Scenario 3:**

CREATE OR REPLACE PROCEDURE TransferFunds (

from\_account\_id IN NUMBER,

to\_account\_id IN NUMBER,

transfer\_amount IN NUMBER

) IS

from\_balance NUMBER;

BEGIN

-- Check balance of source account

SELECT Balance INTO from\_balance FROM Accounts WHERE AccountID = from\_account\_id FOR UPDATE;

IF from\_balance >= transfer\_amount THEN

-- Deduct from source account

UPDATE Accounts

SET Balance = Balance - transfer\_amount

WHERE AccountID = from\_account\_id;

-- Add to destination account

UPDATE Accounts

SET Balance = Balance + transfer\_amount

WHERE AccountID = to\_account\_id;

DBMS\_OUTPUT.PUT\_LINE('Transferred ' || transfer\_amount || ' from Account ' || from\_account\_id || ' to Account ' || to\_account\_id);

COMMIT;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Insufficient balance in Account ' || from\_account\_id);

ROLLBACK;

END IF;

END;

/

Output:

Transferred 500 from Account 1001 to Account 1002