**⇒ Mandatory Hands-On**

**Exercise 1: Control Structures**

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

SET SERVEROUTPUT ON;

DECLARE

    CURSOR cust\_cursor IS

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

        FROM Customers c

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

    v\_CustomerID   Customers.CustomerID%TYPE;

    v\_DOB          Customers.DOB%TYPE;

    v\_LoanID       Loans.LoanID%TYPE;

    v\_InterestRate Loans.InterestRate%TYPE;

    v\_Age          NUMBER;

BEGIN

    FOR rec IN cust\_cursor LOOP

        v\_CustomerID   := rec.CustomerID;

        v\_DOB          := rec.DOB;

        v\_LoanID       := rec.LoanID;

        v\_InterestRate := rec.InterestRate;

        v\_Age := FLOOR(MONTHS\_BETWEEN(SYSDATE, v\_DOB) / 12);

        IF v\_Age > 60 THEN

            UPDATE Loans

            SET InterestRate = v\_InterestRate - 1

            WHERE LoanID = v\_LoanID;

            DBMS\_OUTPUT.PUT\_LINE('✅ Discount applied | Customer ID: ' || v\_CustomerID ||

                                 ' | Loan ID: ' || v\_LoanID ||

                                 ' | Age: ' || v\_Age ||

                                 ' | New Interest Rate: ' || TO\_CHAR(v\_InterestRate - 1));

        ELSE

            DBMS\_OUTPUT.PUT\_LINE('ℹ️ No discount | Customer ID: ' || v\_CustomerID ||

                                 ' | Loan ID: ' || v\_LoanID ||

                                 ' | Age: ' || v\_Age);

        END IF;

    END LOOP;

    COMMIT;

END;

/

Result



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

ALTER TABLE Customers ADD IsVIP VARCHAR2(5);

SET SERVEROUTPUT ON;

DECLARE

    CURSOR cust\_cursor IS

        SELECT CustomerID, Balance

        FROM Customers;

    v\_CustomerID Customers.CustomerID%TYPE;

    v\_Balance    Customers.Balance%TYPE;

BEGIN

    FOR rec IN cust\_cursor LOOP

        v\_CustomerID := rec.CustomerID;

        v\_Balance    := rec.Balance;

        IF v\_Balance > 10000 THEN

            UPDATE Customers

            SET IsVIP = 'TRUE'

            WHERE CustomerID = v\_CustomerID;

            DBMS\_OUTPUT.PUT\_LINE('✅ Promoted to VIP | Customer ID: ' || v\_CustomerID ||

                                 ' | Balance: ' || v\_Balance);

        ELSE

            DBMS\_OUTPUT.PUT\_LINE('ℹ️ Not eligible for VIP | Customer ID: ' || v\_CustomerID ||

                                 ' | Balance: ' || v\_Balance);

        END IF;

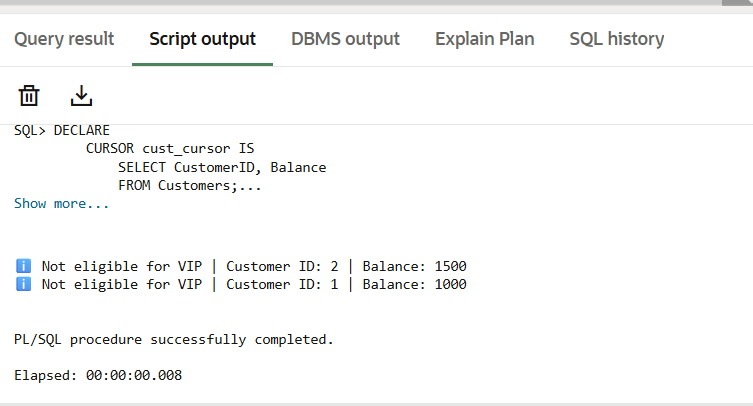
    END LOOP;

    COMMIT;

END;

/

Result



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

SET SERVEROUTPUT ON;

UPDATE Loans

SET EndDate = SYSDATE + 10

WHERE LoanID = 1;

COMMIT;

DECLARE

    CURSOR due\_loans IS

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

        FROM Loans l

        JOIN Customers c ON l.CustomerID = c.CustomerID

        WHERE l.EndDate BETWEEN SYSDATE AND SYSDATE + 30;

    v\_LoanID     Loans.LoanID%TYPE;

    v\_EndDate    Loans.EndDate%TYPE;

    v\_CustomerID Customers.CustomerID%TYPE;

    v\_Name       Customers.Name%TYPE;

    v\_found BOOLEAN := FALSE;

BEGIN

    FOR rec IN due\_loans LOOP

        v\_found := TRUE;

        v\_LoanID     := rec.LoanID;

        v\_EndDate    := rec.EndDate;

        v\_CustomerID := rec.CustomerID;

        v\_Name       := rec.Name;

        DBMS\_OUTPUT.PUT\_LINE('📢 Reminder: Loan ' || v\_LoanID ||

                             ' for customer ' || v\_Name ||

                             ' (Customer ID: ' || v\_CustomerID || ')' ||

                             ' is due on ' || TO\_CHAR(v\_EndDate, 'YYYY-MM-DD'));

    END LOOP;

    IF NOT v\_found THEN

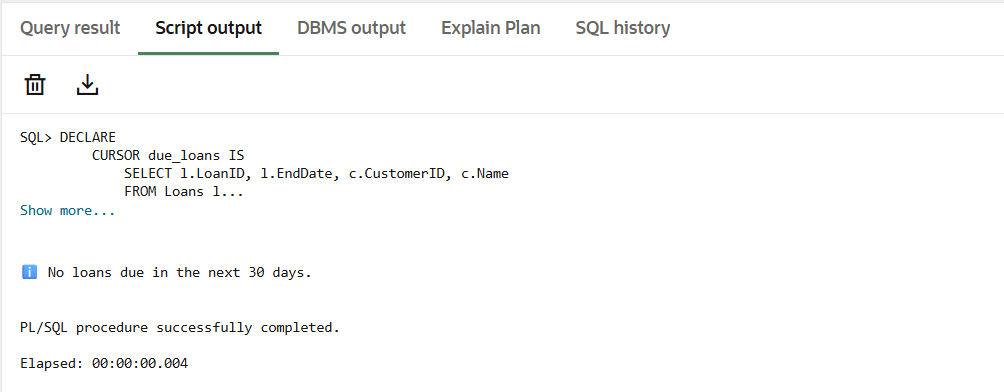
        DBMS\_OUTPUT.PUT\_LINE('ℹ️ No loans due in the next 30 days.');

    END IF;

END;

/

Result



**Exercise 3: Stored Procedures**

**Scenario 1:** The bank needs to process monthly interest for all savings accounts.

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS

BEGIN

FOR acc IN (

SELECT AccountID, Balance

FROM Accounts

WHERE AccountType = 'Savings'

) LOOP

UPDATE Accounts

SET Balance = Balance + (Balance \* 0.01)

WHERE AccountID = acc.AccountID;

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

END LOOP;

COMMIT;

END;

/

SET SERVEROUTPUT ON;

BEGIN

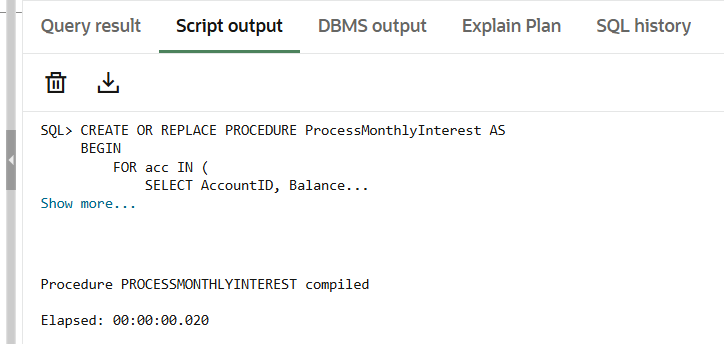
ProcessMonthlyInterest;

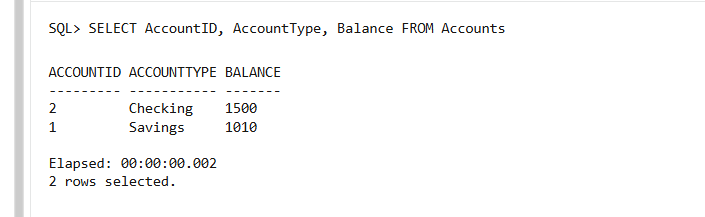
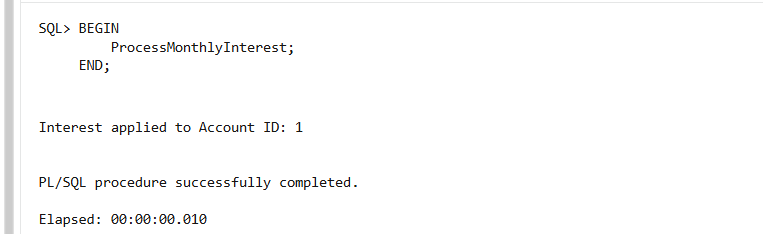
END;

/

SELECT AccountID, AccountType, Balance FROM Accounts;

Result





**Scenario 2:** The bank wants to implement a bonus scheme for employees based on their performance.

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

p\_Department IN VARCHAR2,

p\_BonusPercent IN NUMBER

) AS

BEGIN

FOR emp IN (

SELECT EmployeeID, Salary

FROM Employees

WHERE Department = p\_Department

) LOOP

UPDATE Employees

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

WHERE EmployeeID = emp.EmployeeID;

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

END LOOP;

COMMIT;

END;

/

SET SERVEROUTPUT ON;

BEGIN

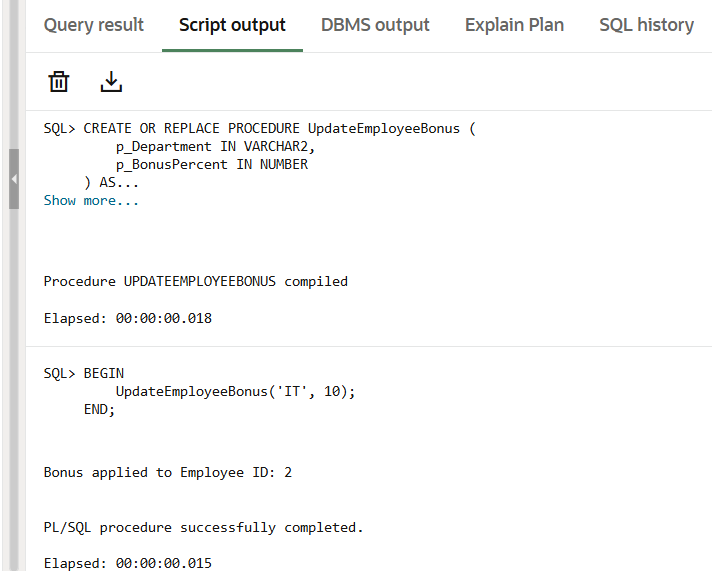
UpdateEmployeeBonus('IT', 10);

END;

/

SELECT EmployeeID, Name, Department, Salary FROM Employees;

Result



**Scenario 3:** Customers should be able to transfer funds between their accounts.

CREATE OR REPLACE PROCEDURE TransferFunds (

p\_FromAccountID IN NUMBER,

p\_ToAccountID IN NUMBER,

p\_Amount IN NUMBER

) AS

v\_FromBalance NUMBER;

BEGIN

SELECT Balance INTO v\_FromBalance

FROM Accounts

WHERE AccountID = p\_FromAccountID;

IF v\_FromBalance >= p\_Amount THEN

UPDATE Accounts

SET Balance = Balance - p\_Amount

WHERE AccountID = p\_FromAccountID;

UPDATE Accounts

SET Balance = Balance + p\_Amount

WHERE AccountID = p\_ToAccountID;

DBMS\_OUTPUT.PUT\_LINE('Transferred ' || p\_Amount || ' from Account ' || p\_FromAccountID || ' to Account ' || p\_ToAccountID);

COMMIT;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Insufficient balance in Account ' || p\_FromAccountID);

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('One of the accounts does not exist.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Unexpected error: ' || SQLERRM);

END;

/

SET SERVEROUTPUT ON;

BEGIN

TransferFunds(2, 1, 200);

END;

/

SELECT AccountID, Balance FROM Accounts;

Result:

