**PL/SQL HANDS-ON**

**CREATING TABLE**

CREATE TABLE Customers (

    CustomerID NUMBER PRIMARY KEY,

    Name VARCHAR2(100),

    DOB DATE,

    Balance NUMBER,

    LastModified DATE,

    IsVIP CHAR(1) DEFAULT 'N'  -- Added for VIP status

);

CREATE TABLE Accounts (

    AccountID NUMBER PRIMARY KEY,

    CustomerID NUMBER,

    AccountType VARCHAR2(20),

    Balance NUMBER,

    LastModified DATE,

    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

CREATE TABLE Transactions (

    TransactionID NUMBER PRIMARY KEY,

    AccountID NUMBER,

    TransactionDate DATE,

    Amount NUMBER,

    TransactionType VARCHAR2(10),

    FOREIGN KEY (AccountID) REFERENCES Accounts(AccountID)

);

CREATE TABLE Loans (

    LoanID NUMBER PRIMARY KEY,

    CustomerID NUMBER,

    LoanAmount NUMBER,

    InterestRate NUMBER,

    StartDate DATE,

    EndDate DATE,

    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

CREATE TABLE Employees (

    EmployeeID NUMBER PRIMARY KEY,

    Name VARCHAR2(100),

    Position VARCHAR2(50),

    Salary NUMBER,

    Department VARCHAR2(50),

    HireDate DATE

);

**INSERTING DATA:**

INSERT INTO Customers VALUES (1, 'John Doe', TO\_DATE('1960-05-15','YYYY-MM-DD'), 12000, SYSDATE, 'N');

INSERT INTO Customers VALUES (2, 'Jane Smith', TO\_DATE('1990-07-20','YYYY-MM-DD'), 8000, SYSDATE, 'N');

INSERT INTO Accounts VALUES (1, 1, 'Savings', 1000, SYSDATE);

INSERT INTO Accounts VALUES (2, 2, 'Checking', 1500, SYSDATE);

INSERT INTO Transactions VALUES (1, 1, SYSDATE, 200, 'Deposit');

INSERT INTO Transactions VALUES (2, 2, SYSDATE, 300, 'Withdrawal');

INSERT INTO Loans VALUES (1, 1, 5000, 5, SYSDATE, ADD\_MONTHS(SYSDATE, 20));

INSERT INTO Employees VALUES (1, 'Alice Johnson', 'Manager', 70000, 'HR', TO\_DATE('2015-06-15', 'YYYY-MM-DD'));

INSERT INTO Employees VALUES (2, 'Bob Brown', 'Developer', 60000, 'IT', TO\_DATE('2017-03-20', 'YYYY-MM-DD'));

**EXERCISE 1:CONTROL STRUCTURES**

**SCENERIO 1**

BEGIN

    FOR rec IN (SELECT l.LoanID, l.InterestRate

                FROM Loans l JOIN Customers c ON l.CustomerID = c.CustomerID

                WHERE MONTHS\_BETWEEN(SYSDATE, c.DOB)/12 > 60) LOOP

        UPDATE Loans

        SET InterestRate = rec.InterestRate - 1

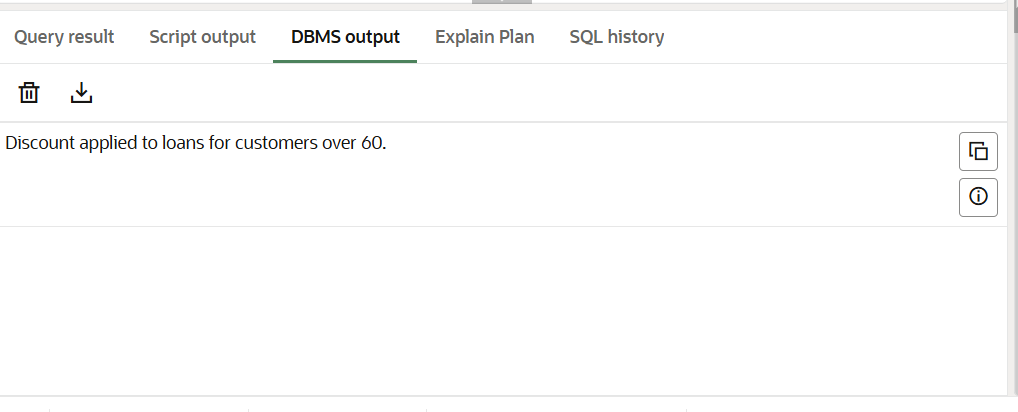
        WHERE LoanID = rec.LoanID;

    END LOOP;

    DBMS\_OUTPUT.PUT\_LINE('Discount applied to loans for customers over 60.');

END;

/



**SCENERIO 2:**

BEGIN

FOR cust IN (SELECT CustomerID FROM Customers WHERE Balance > 10000) LOOP

UPDATE Customers

SET IsVIP = 'Y'

WHERE CustomerID = cust.CustomerID;

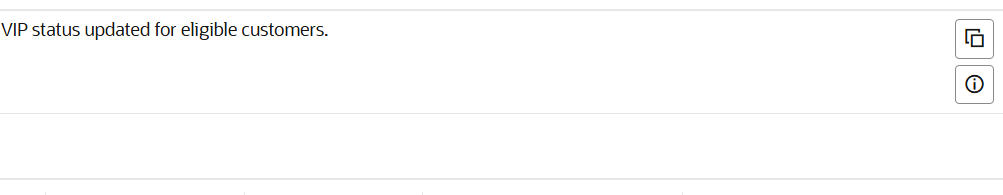
END LOOP;

DBMS\_OUTPUT.PUT\_LINE('VIP status updated for eligible customers.');

END;

/

**OUTPUT**

****

**SCENERIO 3**

**INSERTING VALUES:**

INSERT INTO Loans (LoanID, CustomerID, LoanAmount, InterestRate, StartDate, EndDate)

VALUES (2, 2, 3000, 4.5, SYSDATE, SYSDATE + 15);

BEGIN

    FOR rec IN (SELECT c.Name, l.EndDate

                FROM Loans l JOIN Customers c ON l.CustomerID = c.CustomerID

                WHERE l.EndDate BETWEEN SYSDATE AND SYSDATE + 30) LOOP

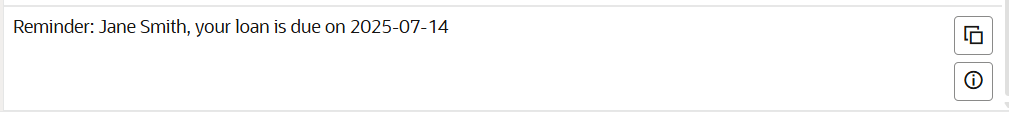
        DBMS\_OUTPUT.PUT\_LINE('Reminder: ' || rec.Name || ', your loan is due on ' || TO\_CHAR(rec.EndDate, 'YYYY-MM-DD'));

    END LOOP;

END;

/

**OUTPUT**

****

**EXERCISE 3:STORED PROCEDURES**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

    FOR acc IN (SELECT AccountID, Balance

                FROM Accounts

                WHERE AccountType = 'Savings') LOOP

        UPDATE Accounts

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

            LastModified = SYSDATE

        WHERE AccountID = acc.AccountID;

    END LOOP;

    DBMS\_OUTPUT.PUT\_LINE('Monthly interest processed for all savings accounts.');

END;

/

**EXAMPLE CALL**

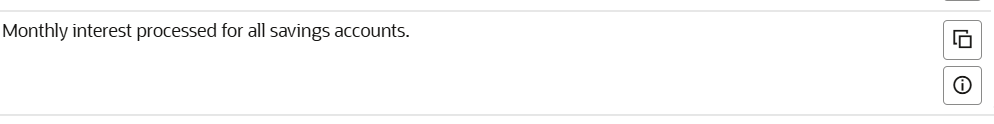
BEGIN

    ProcessMonthlyInterest;

END;

/

**OUTPUT**

****

**SCENERIO 2**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

    dept\_name IN VARCHAR2,

    bonus\_pct IN NUMBER

) IS

BEGIN

    UPDATE Employees

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

    WHERE Department = dept\_name;

    DBMS\_OUTPUT.PUT\_LINE('Bonus applied to ' || dept\_name || ' department.');

END;

/

**SAMPLE CALL**

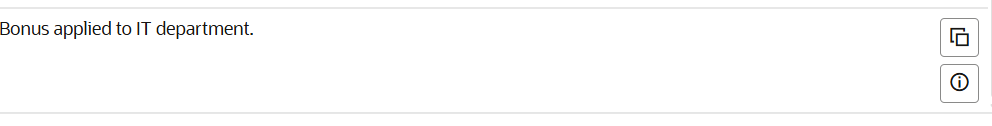
BEGIN

    UpdateEmployeeBonus('IT', 10);

END;

/

**OUTPUT**

****

**SCENERIO 3**

CREATE OR REPLACE PROCEDURE TransferFunds (

    from\_account IN NUMBER,

    to\_account IN NUMBER,

    amount IN NUMBER

) IS

    insufficient\_funds EXCEPTION;

BEGIN

    DECLARE

        from\_balance NUMBER;

    BEGIN

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

        IF from\_balance < amount THEN

            RAISE insufficient\_funds;

        END IF;

        UPDATE Accounts

        SET Balance = Balance - amount,

            LastModified = SYSDATE

        WHERE AccountID = from\_account;

        UPDATE Accounts

        SET Balance = Balance + amount,

            LastModified = SYSDATE

        WHERE AccountID = to\_account;

        DBMS\_OUTPUT.PUT\_LINE('Transfer of $' || amount || ' successful.');

    EXCEPTION

        WHEN insufficient\_funds THEN

            DBMS\_OUTPUT.PUT\_LINE('Error: Insufficient funds.');

    END;

END;

/

**SAMPLE CALL:**

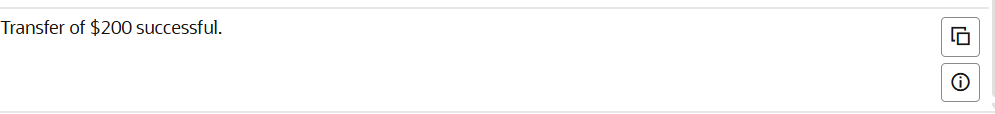
BEGIN

    TransferFunds(1, 2, 200);

END;

/

**OUTPUT:**

****