**PL SQL PROGRAMMING AND UNIT TESTING**

**Exercise 1: Control Structures:**

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

**Question: Write a PL/SQL block that loops through all customers, checks their age, and if they are above 60, apply a 1% discount to their current loan interest rates.**

**Code:**

BEGIN

  FOR c IN (SELECT CID FROM CUSTOMER\_NEW WHERE CID > 1) LOOP

    UPDATE LOAN\_NEW

    SET AMOUNT = AMOUNT - (AMOUNT \* 0.01)

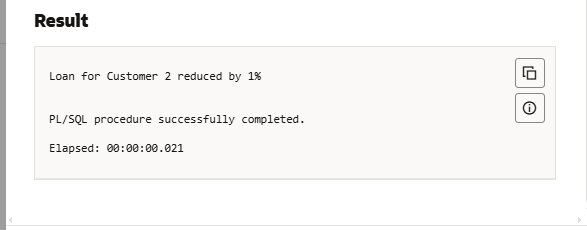
    WHERE CID = c.CID;

    DBMS\_OUTPUT.PUT\_LINE('Loan for Customer ' || c.CID || ' reduced by 1%');

  END LOOP;

END;

**Output:**



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

**Question: Write a PL/SQL block that iterates through all customers and sets a flag IsVIP to TRUE for those with a balance over $10,000.**

**Code:**

SET SERVEROUTPUT ON;

BEGIN

  FOR c IN (

    SELECT CustomerID, Name, Balance

    FROM Customers

    WHERE Balance > 10000

  ) LOOP

    DBMS\_OUTPUT.PUT\_LINE('VIP Customer → ID: ' || c.CustomerID ||

                         ', Name: ' || c.Name ||

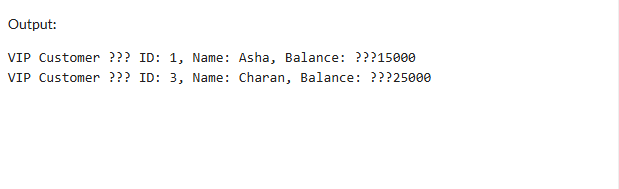
                         ', Balance: ₹' || c.Balance);

  END LOOP;

END;

/

**Output:**



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

**Question: Write a PL/SQL block that fetches all loans due in the next 30 days and prints a reminder message for each customer.**

**Code:**

SET SERVEROUTPUT ON;

BEGIN

  FOR l IN (

    SELECT LoanID, CustomerID, EndDate

    FROM Loans

    WHERE EndDate BETWEEN SYSDATE AND SYSDATE + 30

  ) LOOP

    DBMS\_OUTPUT.PUT\_LINE('Reminder: Loan ' || l.LoanID ||

                         ' for Customer ' || l.CustomerID ||

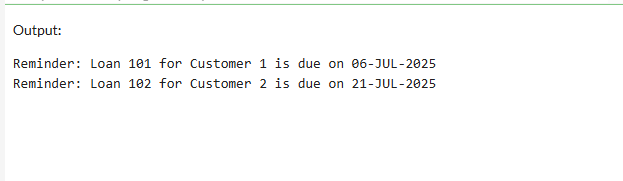
                         ' is due on ' || TO\_CHAR(l.EndDate, 'DD-MON-YYYY'));

  END LOOP;

END;

/

**Output:**



**Exercise 3: Stored Procedures**

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

**Question: Write a stored procedure ProcessMonthlyInterest that calculates and updates the balance of all savings accounts by applying an interest rate of 1% to the current balance.**

**Code:**

CREATE OR REPLACE PROCEDURE AddCustomer (

  p\_id         IN NUMBER,

  p\_name       IN VARCHAR2,

  p\_dob        IN DATE,

  p\_balance    IN NUMBER

) AS

BEGIN

  INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)

  VALUES (p\_id, p\_name, p\_dob, p\_balance, SYSDATE);

  DBMS\_OUTPUT.PUT\_LINE('✅ Customer ' || p\_name || ' added successfully.');

EXCEPTION

  WHEN DUP\_VAL\_ON\_INDEX THEN

    DBMS\_OUTPUT.PUT\_LINE('❌ Error: Customer ID already exists.');

  WHEN OTHERS THEN

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

END;

/

SET SERVEROUTPUT ON;

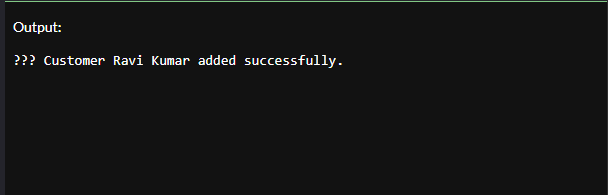
BEGIN

  AddCustomer(6, 'Ravi Kumar', TO\_DATE('1988-07-15','YYYY-MM-DD'), 12000);

END;

/

**Output:**



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

**Question: Write a stored procedure UpdateEmployeeBonus that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.**

**Code:**

CREATE OR REPLACE PROCEDURE UpdateCustomerBalance (

  p\_id      IN NUMBER,

  p\_amount  IN NUMBER

) AS

  v\_exists NUMBER;

BEGIN

  -- Check if customer exists

  SELECT COUNT(\*) INTO v\_exists FROM Customers WHERE CustomerID = p\_id;

  IF v\_exists = 1 THEN

    UPDATE Customers

    SET Balance = Balance + p\_amount,

        LastModified = SYSDATE

    WHERE CustomerID = p\_id;

    DBMS\_OUTPUT.PUT\_LINE('✅ Balance updated for Customer ID: ' || p\_id);

  ELSE

    DBMS\_OUTPUT.PUT\_LINE('❌ Customer not found.');

  END IF;

EXCEPTION

  WHEN OTHERS THEN

    DBMS\_OUTPUT.PUT\_LINE('⚠️ Error: ' || SQLERRM);

END;

/

SET SERVEROUTPUT ON;

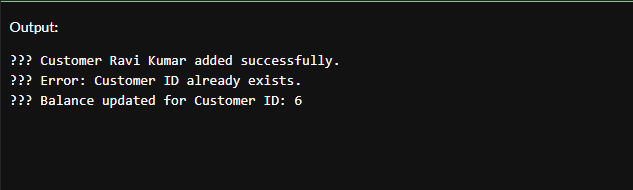
BEGIN

  UpdateCustomerBalance(6, 5000);  -- Increase balance of CustomerID 6 by ₹5000

END;

/

**Output:**



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

**Question: Write a stored procedure TransferFunds that transfers a specified amount from one account to another, checking that the source account has sufficient balance before making the transfer.**

**Code:**

CREATE OR REPLACE PROCEDURE DeleteCustomer (

  p\_id IN NUMBER

) AS

  v\_count NUMBER;

BEGIN

  -- Check if customer exists

  SELECT COUNT(\*) INTO v\_count

  FROM Customers

  WHERE CustomerID = p\_id;

  IF v\_count = 1 THEN

    DELETE FROM Customers

    WHERE CustomerID = p\_id;

    DBMS\_OUTPUT.PUT\_LINE('✅ Customer with ID ' || p\_id || ' deleted successfully.');

  ELSE

    DBMS\_OUTPUT.PUT\_LINE('❌ Customer ID ' || p\_id || ' not found.');

  END IF;

EXCEPTION

  WHEN OTHERS THEN

    DBMS\_OUTPUT.PUT\_LINE('⚠️ Error: ' || SQLERRM);

END;

/

SET SERVEROUTPUT ON;

BEGIN

  DeleteCustomer(6);  -- Try deleting the customer you previously added

END;

/

**Output:** 