**PL SQL QUERIES**

**Exercise 1: Control Structures**

-- Drop tables if they already exist (optional cleanup)

BEGIN

  EXECUTE IMMEDIATE 'DROP TABLE Loans';

  EXECUTE IMMEDIATE 'DROP TABLE Customers';

EXCEPTION

  WHEN OTHERS THEN NULL;

END;

/

-- Step 1: Create Tables

CREATE TABLE Customers (

  CustomerID   NUMBER PRIMARY KEY,

  Name         VARCHAR2(100),

  Age          NUMBER,

  Balance      NUMBER,

  IsVIP        VARCHAR2(5)

);

CREATE TABLE Loans (

  LoanID       NUMBER PRIMARY KEY,

  CustomerID   NUMBER,

  InterestRate NUMBER(5,2),

  DueDate      DATE,

  FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

-- Step 2: Insert Data

INSERT INTO Customers VALUES (1, 'Alice', 65, 12000, 'FALSE');

INSERT INTO Customers VALUES (2, 'Bob', 45, 8000, 'FALSE');

INSERT INTO Customers VALUES (3, 'Carol', 70, 15000, 'FALSE');

INSERT INTO Loans VALUES (101, 1, 7.5, SYSDATE + 10);

INSERT INTO Loans VALUES (102, 2, 8.0, SYSDATE + 40);

INSERT INTO Loans VALUES (103, 3, 6.5, SYSDATE + 5);

COMMIT;

-- Step 3: VIP Promotion PL/SQL Block

BEGIN

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

    UPDATE Customers

    SET IsVIP = 'TRUE'

    WHERE CustomerID = cust.CustomerID;

  END LOOP;

  COMMIT;

END;

/

-- Step 4: Loan Reminder Block

BEGIN

  FOR loan IN (

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

    FROM Loans l

    JOIN Customers c ON l.CustomerID = c.CustomerID

    WHERE l.DueDate <= SYSDATE + 30

  ) LOOP

    DBMS\_OUTPUT.PUT\_LINE('Reminder: Dear ' || loan.Name ||

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

                         ') is due on ' || TO\_CHAR(loan.DueDate, 'DD-Mon-YYYY'));

  END LOOP;

END;

/

-- Step 5: Check Results

SELECT \* FROM Customers;

SELECT \* FROM Loans;

Dbms output :

Reminder: Dear Alice, your loan (ID: 101) is due on 06-Jul-2025

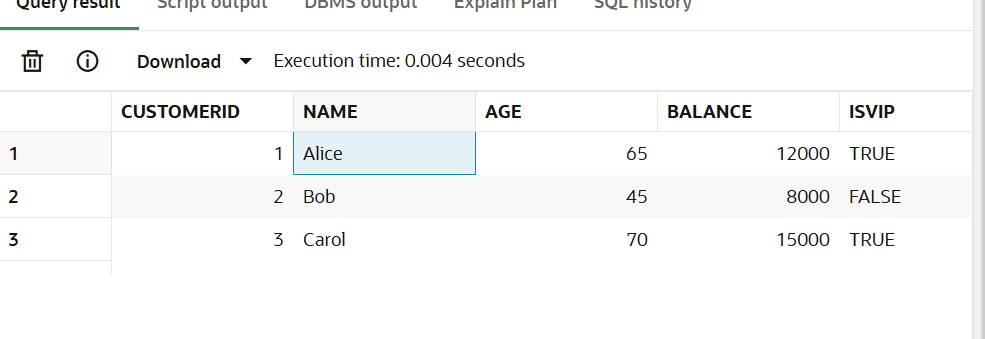
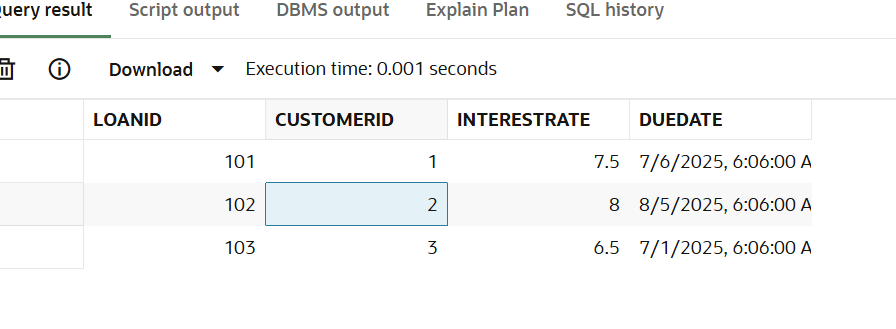
Reminder: Dear Carol, your loan (ID: 103) is due on 01-Jul-2025

"CUSTOMERID" "NAME" "AGE" "BALANCE" "ISVIP"

1 "Alice" 65 12000 "TRUE"

Query Result :

"Bob" 45 8000 "FALSE"

3 "CUSTOMERID" "NAME" "AGE" " BALANCE" "ISVIP"

**Exercise 3: Stored Procedures**

CREATE TABLE Accounts (

  AccountID   NUMBER PRIMARY KEY,

  CustomerID  NUMBER,

  Balance     NUMBER,

  AccountType VARCHAR2(20) -- e.g., 'Savings', 'Checking'

);

CREATE TABLE Employees (

  EmployeeID   NUMBER PRIMARY KEY,

  Name         VARCHAR2(100),

  DepartmentID NUMBER,

  Salary       NUMBER

);

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS

BEGIN

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

    UPDATE Accounts

    SET Balance = Balance + (acc.Balance \* 0.01)

    WHERE AccountID = acc.AccountID;

  END LOOP;

  COMMIT;

END;

/

CREATE OR REPLACE PROCEDURE TransferFunds(

  from\_acc IN NUMBER,

  to\_acc IN NUMBER,

  amount IN NUMBER

) AS

  from\_balance NUMBER;

  cnt NUMBER;

BEGIN

  -- Check if source account exists

  SELECT COUNT(\*) INTO cnt FROM Accounts WHERE AccountID = from\_acc;

  IF cnt = 0 THEN

    RAISE\_APPLICATION\_ERROR(-20002, 'Source account does not exist');

  END IF;

  -- Check if destination account exists

  SELECT COUNT(\*) INTO cnt FROM Accounts WHERE AccountID = to\_acc;

  IF cnt = 0 THEN

    RAISE\_APPLICATION\_ERROR(-20003, 'Destination account does not exist');

  END IF;

  -- Get source balance

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

  -- Check if balance is sufficient

  IF from\_balance < amount THEN

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

  END IF;

  -- Transfer funds

  UPDATE Accounts

  SET Balance = Balance - amount

  WHERE AccountID = from\_acc;

  UPDATE Accounts

  SET Balance = Balance + amount

  WHERE AccountID = to\_acc;

  COMMIT;

END;

/

INSERT INTO Accounts VALUES (1001, 1, 5000, 'Savings');

INSERT INTO Accounts VALUES (1002, 2, 3000, 'Savings');

COMMIT;

SELECT \* FROM Accounts;

BEGIN

  TransferFunds(1001, 1002, 1000);

END;

Output:

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

"CUSTOMERID" "NAME" "AGE" "BALANCE" "ISVIP"

1 "Alice" 65 12000 "TRUE"

2 "Bob" 45 8000 "FALSE"

3 "Carol" 70 15000 "TRUE"