SUPERSET: 6376594

PL/SQL PROGRAMMING EXERCISES(HANDS-ON)

Schema Created:

*CREATE TABLE Customers (*

*CustomerID NUMBER PRIMARY KEY,*

*Name VARCHAR2(100),*

*DOB DATE,*

*Balance NUMBER,*

*LastModified DATE*

*);*

*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*

*);*

Inserted Data:

**-- Customers (5 entries)**

INSERT INTO Customers VALUES (1, 'John Doe', TO\_DATE('1950-05-15', 'YYYY-MM-DD'), 9500, SYSDATE);

INSERT INTO Customers VALUES (2, 'Jane Smith', TO\_DATE('1980-07-20', 'YYYY-MM-DD'), 15000, SYSDATE);

INSERT INTO Customers VALUES (3, 'Robert King', TO\_DATE('1948-02-10', 'YYYY-MM-DD'), 12000, SYSDATE);

INSERT INTO Customers VALUES (4, 'Emily White', TO\_DATE('1995-11-25', 'YYYY-MM-DD'), 8000, SYSDATE);

INSERT INTO Customers VALUES (5, 'Michael Green', TO\_DATE('1972-03-30', 'YYYY-MM-DD'), 20000, SYSDATE);

**-- Accounts (5 entries)**

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

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

INSERT INTO Accounts VALUES (3, 3, 'Savings', 12000, SYSDATE);

INSERT INTO Accounts VALUES (4, 4, 'Current', 8000, SYSDATE);

INSERT INTO Accounts VALUES (5, 5, 'Savings', 20000, SYSDATE);

**-- Transactions (5 entries)**

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

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

INSERT INTO Transactions VALUES (3, 3, SYSDATE - 1, 500, 'Deposit');

INSERT INTO Transactions VALUES (4, 4, SYSDATE - 3, 1000, 'Withdrawal');

INSERT INTO Transactions VALUES (5, 5, SYSDATE - 2, 1500, 'Deposit');

**-- Loans (5 entries)**

INSERT INTO Loans VALUES (1, 1, 5000, 6, SYSDATE, SYSDATE + 15);

INSERT INTO Loans VALUES (2, 2, 8000, 5, SYSDATE, SYSDATE + 45);

INSERT INTO Loans VALUES (3, 3, 7000, 6.5, SYSDATE, SYSDATE + 10);

INSERT INTO Loans VALUES (4, 4, 9000, 5.5, SYSDATE, SYSDATE + 35);

INSERT INTO Loans VALUES (5, 5, 12000, 4.5, SYSDATE, SYSDATE + 25);

**-- Employees (5 entries)**

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'));

INSERT INTO Employees VALUES (3, 'Clara Evans', 'Analyst', 55000, 'Finance', TO\_DATE('2019-10-01', 'YYYY-MM-DD'));

INSERT INTO Employees VALUES (4, 'David Lee', 'Consultant', 65000, 'Marketing', TO\_DATE('2020-01-15', 'YYYY-MM-DD'));

INSERT INTO Employees VALUES (5, 'Nina Patel', 'Engineer', 72000, 'R&D', TO\_DATE('2018-07-10', 'YYYY-MM-DD'));

**Exercise 1: Control Structures**

**Scenario-1:**

BEGIN

  FOR cust IN (

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

    FROM Customers c

    JOIN Loans l ON c.CustomerID = l.CustomerID

  ) LOOP

    IF MONTHS\_BETWEEN(SYSDATE, cust.DOB) / 12 > 60 THEN

      UPDATE Loans

      SET InterestRate = InterestRate - 1

      WHERE LoanID = cust.LoanID;

    END IF;

  END LOOP;

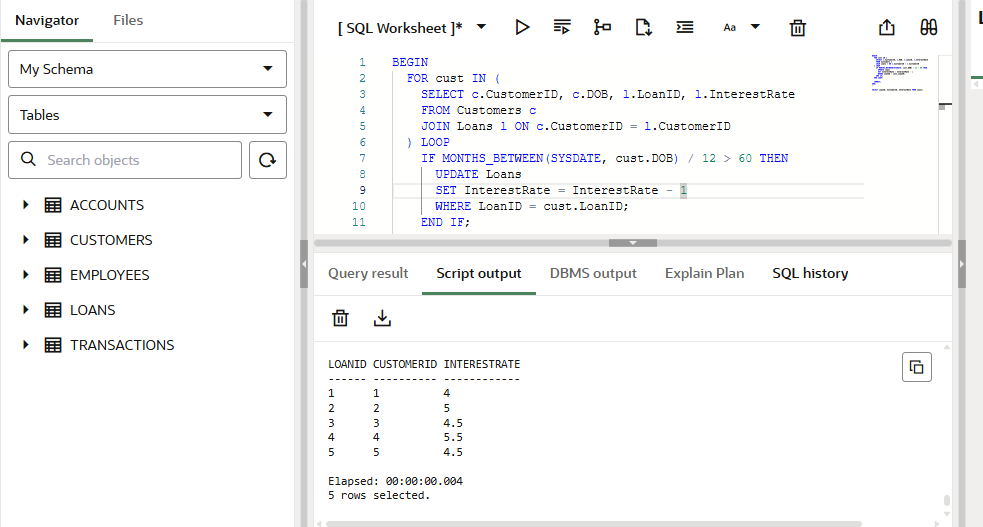
  COMMIT;

END;

/

SELECT LoanID, CustomerID, InterestRate FROM Loans;

**Output:**

****

**Scenario-2**

ALTER TABLE Customers ADD IsVIP VARCHAR2(5);

BEGIN

  FOR cust IN (SELECT CustomerID, Balance FROM Customers) LOOP

    IF cust.Balance > 10000 THEN

      UPDATE Customers

      SET IsVIP = 'TRUE'

      WHERE CustomerID = cust.CustomerID;

    ELSE

      UPDATE Customers

      SET IsVIP = 'FALSE'

      WHERE CustomerID = cust.CustomerID;

    END IF;

  END LOOP;

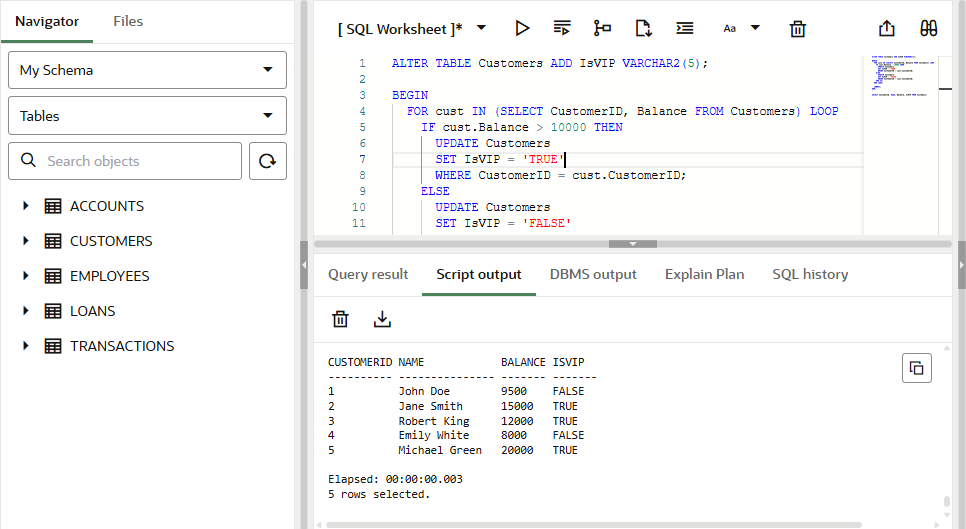
  COMMIT;

END;

/

SELECT CustomerID, Name, Balance, IsVIP FROM Customers;

**Output:**

****

**Scenario-3**BEGIN

  FOR l IN (

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

    FROM Loans l

    JOIN Customers c ON l.CustomerID = c.CustomerID

    WHERE l.EndDate <= SYSDATE + 30

  ) LOOP

    DBMS\_OUTPUT.PUT\_LINE(

      'Reminder: Loan ID ' || l.LoanID ||

      ' for customer "' || l.Name ||

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

    );

  END LOOP;

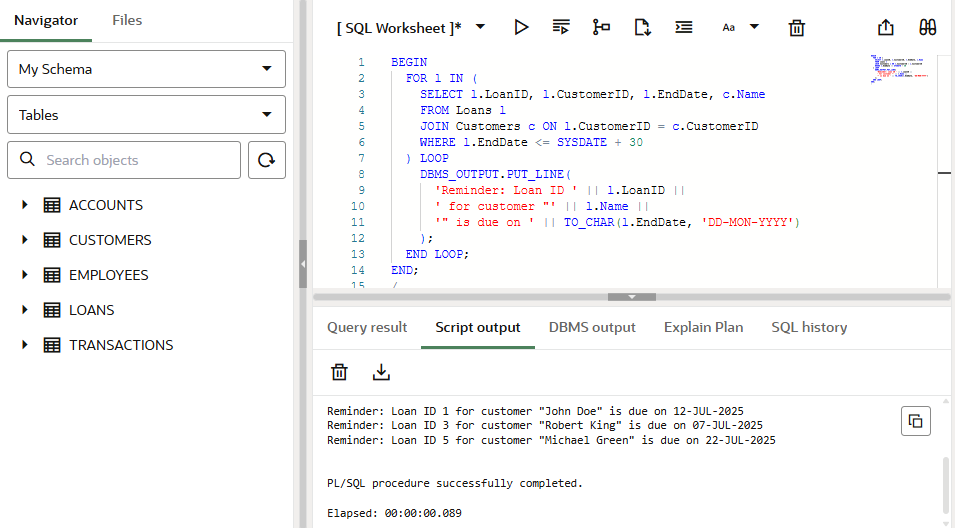
END;

/

**Expected Output:**

Reminder: Loan ID 1 for customer "John Doe" is due on 12-JUL-2025  
Reminder: Loan ID 3 for customer "Robert King" is due on 07-JUL-2025  
Reminder: Loan ID 5 for customer "Michael Green" is due on 22-JUL-2025

**Output:**

****

**Exercise 3: Stored Procedures**

**Scenario-1**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

  -- Loop through all savings accounts

  FOR acc IN (

    SELECT AccountID, Balance

    FROM Accounts

    WHERE AccountType = 'Savings'

  ) LOOP

    -- Update the balance with 1% interest

    UPDATE Accounts

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

        LastModified = SYSDATE

    WHERE AccountID = acc.AccountID;

  END LOOP;

  COMMIT; -- Save the changes to the database

END;

/

BEGIN

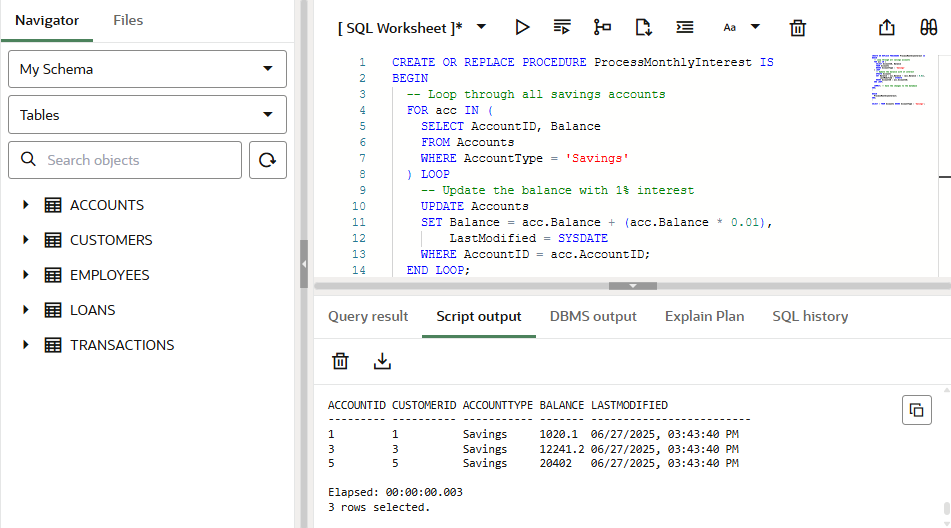
  ProcessMonthlyInterest;

END;

/

SELECT \* FROM Accounts WHERE AccountType = 'Savings';

**Output:**

****

**Scenario-2**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

  p\_department IN VARCHAR2,

  p\_bonus\_percent IN NUMBER

) IS

BEGIN

  -- Update salary for employees in the specified department

  UPDATE Employees

  SET Salary = Salary + (Salary \* (p\_bonus\_percent / 100))

  WHERE Department = p\_department;

  COMMIT;

END;

/

BEGIN

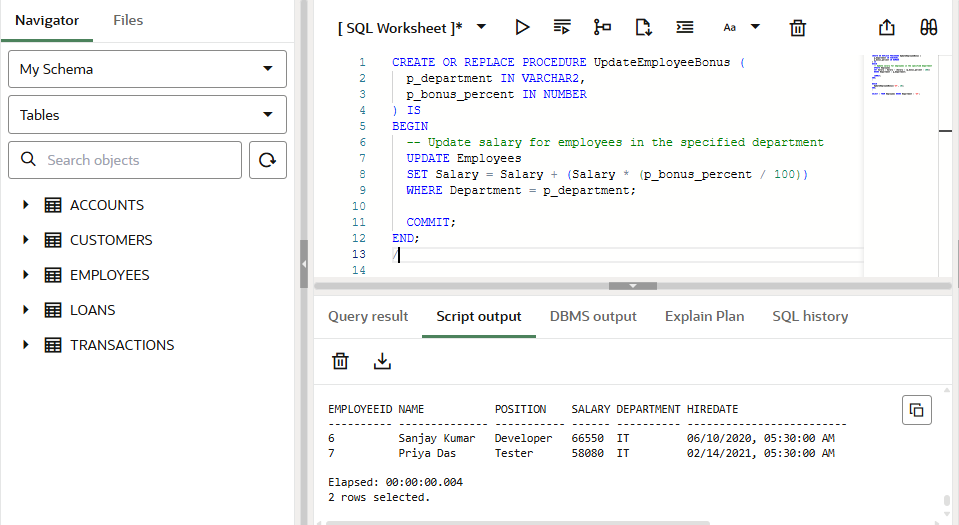
  UpdateEmployeeBonus('IT', 10);

END;

/

SELECT \* FROM Employees WHERE Department = 'IT';

**Output:**

****

**Scenario-3**

CREATE OR REPLACE PROCEDURE TransferFunds (

  p\_from\_account IN NUMBER,

  p\_to\_account IN NUMBER,

  p\_amount IN NUMBER

) IS

  v\_balance NUMBER;

BEGIN

  -- Step 1: Get balance of source account

  SELECT Balance INTO v\_balance

  FROM Accounts

  WHERE AccountID = p\_from\_account;

  -- Step 2: Check for sufficient funds

  IF v\_balance < p\_amount THEN

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

  END IF;

  -- Step 3: Deduct from source account

  UPDATE Accounts

  SET Balance = Balance - p\_amount,

      LastModified = SYSDATE

  WHERE AccountID = p\_from\_account;

  -- Step 4: Add to target account

  UPDATE Accounts

  SET Balance = Balance + p\_amount,

      LastModified = SYSDATE

  WHERE AccountID = p\_to\_account;

  COMMIT;

END;

/

BEGIN

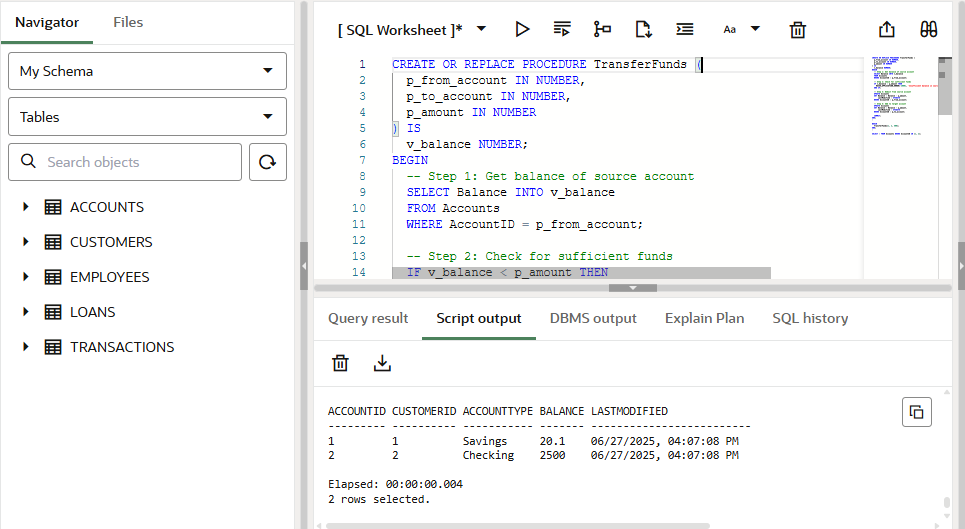
  TransferFunds(1, 2, 500);

END;

/

SELECT \* FROM Accounts WHERE AccountID IN (1, 2);

**Output:**

****