**PL/SQL PROGRAMMING**

* **Exercise 1: Control Structures**

CREATE TABLE Customers (

CustomerID NUMBER PRIMARY KEY,

Name VARCHAR2(100),

DOB DATE,

Balance NUMBER,

LastModified DATE,

IsVIP CHAR(1)

);

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

);

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

VALUES (1, 'John Doe', TO\_DATE('1955-05-15', 'YYYY-MM-DD'), 1000, SYSDATE, 'N');

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

VALUES (2, 'Jane Smith', TO\_DATE('1940-07-20', 'YYYY-MM-DD'), 12000, SYSDATE, 'N');

INSERT INTO Accounts (AccountID, CustomerID, AccountType, Balance, LastModified)

VALUES (1, 1, 'Savings', 1000, SYSDATE);

INSERT INTO Accounts (AccountID, CustomerID, AccountType, Balance, LastModified)

VALUES (2, 2, 'Checking', 1500, SYSDATE);

INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

VALUES (1, 1, SYSDATE, 200, 'Deposit');

INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

VALUES (2, 2, SYSDATE, 300, 'Withdrawal');

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

VALUES (1, 1, 5000, 5, SYSDATE, ADD\_MONTHS(SYSDATE, 60));

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

VALUES (2, 2, 7000, 6, SYSDATE, SYSDATE + 20);

INSERT INTO Employees (EmployeeID, Name, Position, Salary, Department, HireDate)

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

INSERT INTO Employees (EmployeeID, Name, Position, Salary, Department, HireDate)

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

COMMIT;

BEGIN

FOR cust IN (

SELECT CustomerID FROM Customers

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

) LOOP

UPDATE Loans

SET InterestRate = InterestRate - 1

WHERE CustomerID = cust.CustomerID;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Scenario 1: Interest rate discount applied for customers above 60.');

END;

/

BEGIN

FOR cust IN (

SELECT CustomerID, Name FROM Customers

WHERE Balance > 10000

) LOOP

UPDATE Customers

SET IsVIP = 'Y'

WHERE CustomerID = cust.CustomerID;

DBMS\_OUTPUT.PUT\_LINE('Scenario 2: ' || cust.Name || ' promoted to VIP.');

END LOOP;

END;

/

BEGIN

FOR rec IN (

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

FROM Loans l

JOIN Customers c ON l.CustomerID = c.CustomerID

WHERE l.EndDate BETWEEN SYSDATE AND SYSDATE + 30

) LOOP

DBMS\_OUTPUT.PUT\_LINE('Scenario 3: Reminder - ' || rec.Name ||

', your loan (ID: ' || rec.LoanID || ') is due on ' ||

TO\_CHAR(rec.EndDate, 'DD-MON-YYYY'));

END LOOP;

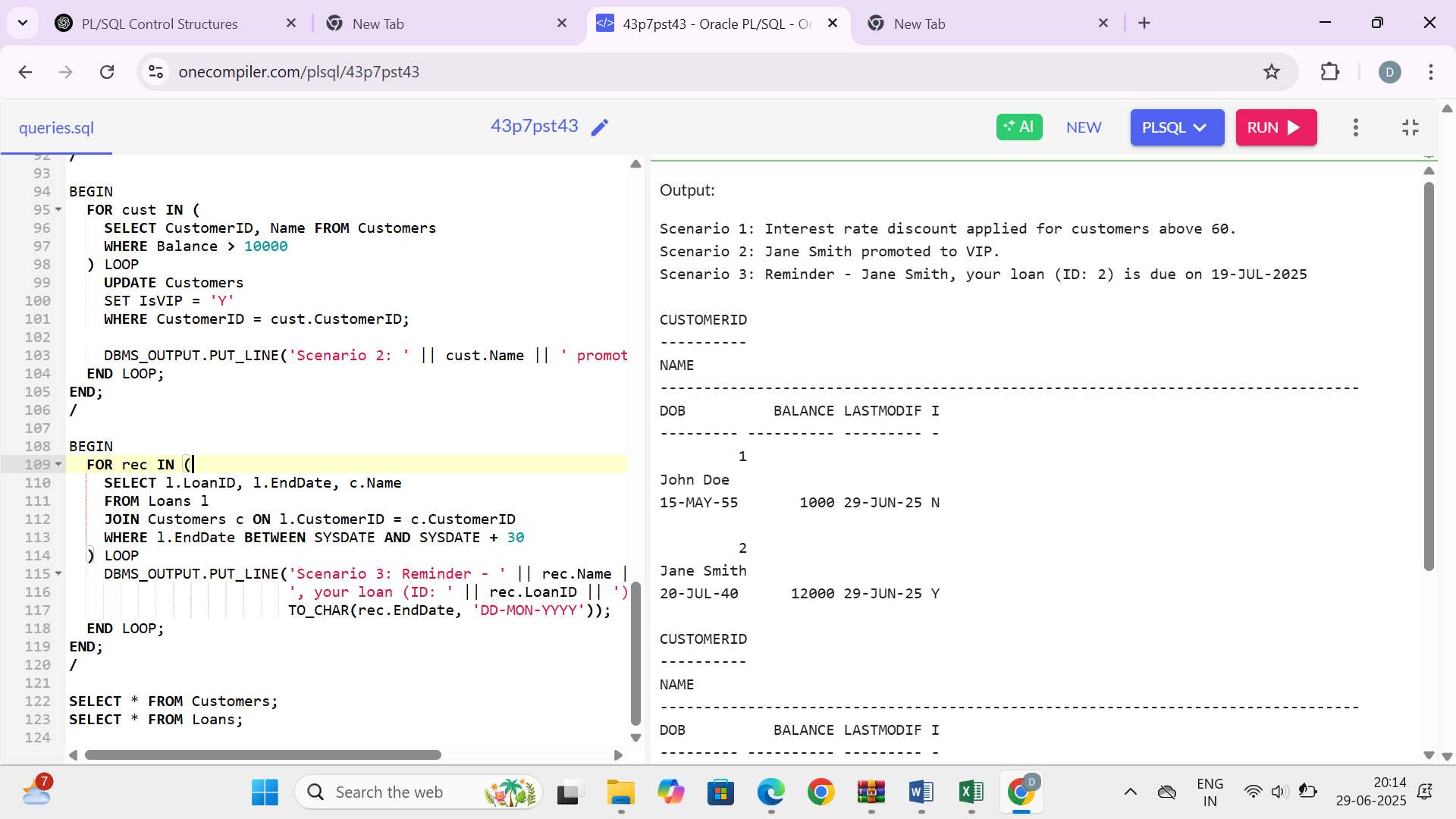
END;

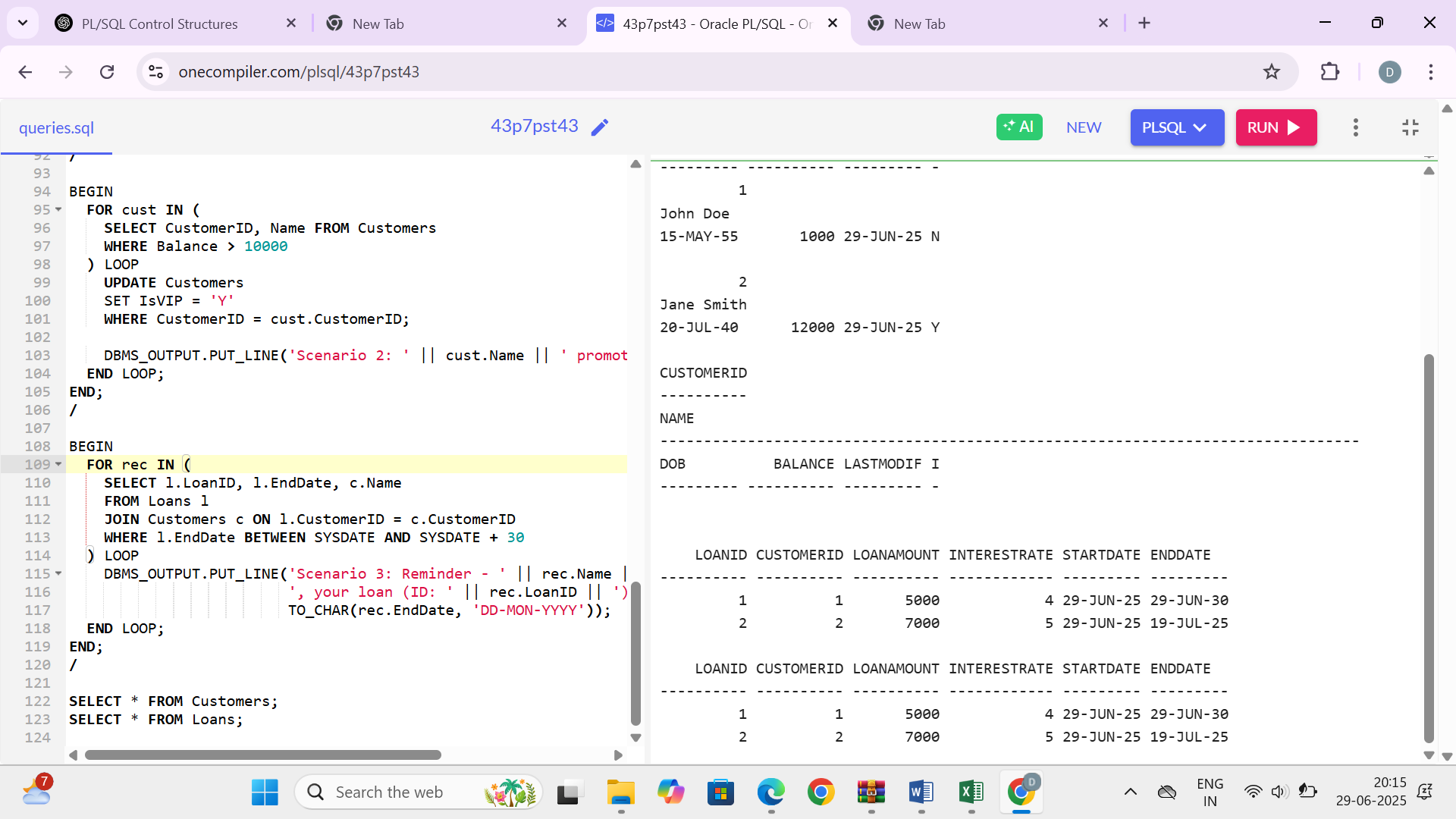
/

SELECT \* FROM Customers;

SELECT \* FROM Loans;

**OUTPUT**





* **Exercise 3: Stored Procedures**

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(20),

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

);

INSERT INTO Customers VALUES (1, 'John Doe', TO\_DATE('1985-05-15', 'YYYY-MM-DD'), 1000, SYSDATE);

INSERT INTO Customers VALUES (2, 'Jane Smith', TO\_DATE('1990-07-20', 'YYYY-MM-DD'), 1500, SYSDATE);

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, 60));

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'));

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;

END;

/

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(dept IN VARCHAR2, bonus\_pct IN NUMBER) IS

BEGIN

UPDATE Employees

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

WHERE Department = dept;

END;

/

CREATE OR REPLACE PROCEDURE TransferFunds(from\_acc IN NUMBER, to\_acc IN NUMBER, amount IN NUMBER) IS

from\_bal NUMBER;

txn\_id NUMBER;

BEGIN

SELECT Balance INTO from\_bal FROM Accounts WHERE AccountID = from\_acc;

IF from\_bal < amount THEN

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

END IF;

UPDATE Accounts SET Balance = Balance - amount, LastModified = SYSDATE WHERE AccountID = from\_acc;

UPDATE Accounts SET Balance = Balance + amount, LastModified = SYSDATE WHERE AccountID = to\_acc;

SELECT NVL(MAX(TransactionID), 0) + 1 INTO txn\_id FROM Transactions;

INSERT INTO Transactions VALUES (txn\_id, from\_acc, SYSDATE, amount, 'Transfer-Out');

INSERT INTO Transactions VALUES (txn\_id + 1, to\_acc, SYSDATE, amount, 'Transfer-In');

END;

/

BEGIN

DBMS\_OUTPUT.PUT\_LINE('--- Running ProcessMonthlyInterest ---');

ProcessMonthlyInterest;

DBMS\_OUTPUT.PUT\_LINE('--- Giving 10% bonus to HR ---');

UpdateEmployeeBonus('HR', 10);

DBMS\_OUTPUT.PUT\_LINE('--- Transferring $300 from Account 2 to 1 ---');

TransferFunds(2, 1, 300);

END;

/

DECLARE

CURSOR acc\_cur IS SELECT \* FROM Accounts;

CURSOR txn\_cur IS SELECT \* FROM Transactions;

CURSOR emp\_cur IS SELECT EmployeeID, Name, Salary, Department FROM Employees;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('--- Final Account Balances---');

FOR r IN acc\_cur LOOP

DBMS\_OUTPUT.PUT\_LINE('AccountID: ' || r.AccountID || ', CustomerID: ' || r.CustomerID ||

', Type: ' || r.AccountType || ', Balance: $' || r.Balance);

END LOOP;

DBMS\_OUTPUT.PUT\_LINE(CHR(10) || '--- Final Transactions ---');

FOR t IN txn\_cur LOOP

DBMS\_OUTPUT.PUT\_LINE('TxnID: ' || t.TransactionID || ', AccountID: ' || t.AccountID ||

', Amount: $' || t.Amount || ', Type: ' || t.TransactionType);

END LOOP;

DBMS\_OUTPUT.PUT\_LINE(CHR(10) || '--- Final Employee Salaries ---');

FOR e IN emp\_cur LOOP

DBMS\_OUTPUT.PUT\_LINE('EmpID: ' || e.EmployeeID || ', Name: ' || e.Name ||

', Salary: $' || e.Salary || ', Dept: ' || e.Department);

END LOOP;

END;

/

**OUTPUT**