***WEEK 2***

***MANDATORY HANDS-ON***

***PL SQL***

**-- Customers**

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

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

INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified) VALUES (3, 'GDC', TO\_DATE('1905-05-15', 'YYYY-MM-DD'), 2000, SYSDATE);

INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified) VALUES (99, 'Senior Citizen', TO\_DATE('1950-01-01','YYYY-MM-DD'), 225000, SYSDATE);

INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified) VALUES (199, 'Virat Kohli', TO\_DATE('1950-01-01','YYYY-MM-DD'), 225000, SYSDATE);

INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified) VALUES (777, 'Ravi Teja', TO\_DATE('1970-10-15', 'YYYY-MM-DD'), 7000, SYSDATE);

**-- Accounts**

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);

**-- Transactions**

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');

**-- Loans**

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 (99, 99, 10000, 7, SYSDATE, ADD\_MONTHS(SYSDATE, 60));

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

VALUES (101, 199, 12000, 6.5, SYSDATE, SYSDATE + 10);

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

VALUES (777, 777, 15000, 8, SYSDATE, SYSDATE + 10);

-- Employees

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;

**Exercise 1: Control Structures**

***--Scenario 1: Senior Citizens (>60 yrs) Get 1% Reduced Loan Interest***

SET SERVEROUTPUT ON;

BEGIN

FOR c IN (SELECT \* FROM Customers c JOIN Loans l ON c.CustomerID = l.CustomerID) LOOP

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

UPDATE Loans

SET InterestRate = InterestRate - 1

WHERE LoanID = c.LoanID;

END IF;

END LOOP;

END;

/

***-- View affected records***

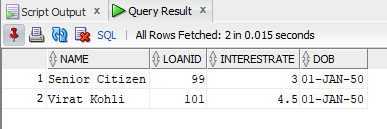
SELECT c.Name, l.LoanID, l.InterestRate, c.DOB

FROM Customers c

JOIN Loans l ON c.CustomerID = l.CustomerID

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

***OUTPUT***



**Exercise 3: Stored Procedures**

***-- Scenario 1: Process monthly interest***

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

UPDATE Accounts

SET Balance = Balance + (Balance \* 0.01)

WHERE AccountType = 'Savings';

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

COMMIT;

END;

/

***-- updates balances of Savings accounts by adding 1% interest.***

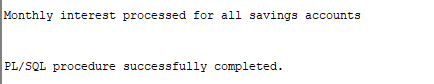
BEGIN

ProcessMonthlyInterest;

END;

/

***OUTPUT***



***-- Scenario 2: Update employee bonus***

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(p\_dept VARCHAR2, p\_bonus NUMBER) IS

BEGIN

UPDATE Employees

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

WHERE Department = p\_dept;

DBMS\_OUTPUT.PUT\_LINE('Bonus applied to department ' || p\_dept);

COMMIT;

END;

/

***--updates salaries of employees in a department.***

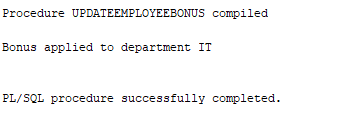
BEGIN

UpdateEmployeeBonus('IT', 15); -- Add 15% bonus to IT department

END;

/

***OUTPUT***

****

***-- Scenario 3: Transfer funds***

CREATE OR REPLACE PROCEDURE TransferFunds(p\_from NUMBER, p\_to NUMBER, p\_amount NUMBER) IS

v\_balance NUMBER;

BEGIN

SELECT Balance INTO v\_balance FROM Accounts WHERE AccountID = p\_from;

IF v\_balance >= p\_amount THEN

UPDATE Accounts SET Balance = Balance - p\_amount WHERE AccountID = p\_from;

UPDATE Accounts SET Balance = Balance + p\_amount WHERE AccountID = p\_to;

DBMS\_OUTPUT.PUT\_LINE('Transfer successful!');

COMMIT;

ELSE

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

END IF;

END;

/

***--if enough balance is available.***

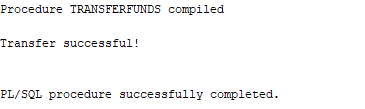
BEGIN

TransferFunds(1, 2, 50); -- Transfers ₹50 from account 1 to 2

END;

/

***OUTPUT***



***--Fails (shows output)***

BEGIN

TransferFunds(1, 2, 100000); -- Too much, should fail

END;

/

***OUTPUT***

