PL SQL EXCERCISES(WEEK2\_MODULE)

EXCERCISE-1

CONTROL STRUCTURES:

CODE:

SET SERVEROUTPUT ON;

-- Create Tables (If not already created)

CREATE TABLE Customers (

CustomerID NUMBER PRIMARY KEY,

Name VARCHAR2(100),

DOB DATE,

Balance NUMBER,

LastModified DATE,

IsVIP VARCHAR2(10)

);

CREATE TABLE Loans (

LoanID NUMBER PRIMARY KEY,

CustomerID NUMBER,

LoanAmount NUMBER,

InterestRate NUMBER,

StartDate DATE,

EndDate DATE,

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

-- Insert Sample Data

INSERT INTO Customers VALUES (1, 'John Doe', TO\_DATE('1960-01-01', 'YYYY-MM-DD'), 12000, SYSDATE, 'FALSE');

INSERT INTO Customers VALUES (2, 'Jane Smith', TO\_DATE('1980-05-10', 'YYYY-MM-DD'), 8000, SYSDATE, 'FALSE');

INSERT INTO Loans VALUES (1, 1, 5000, 10, SYSDATE, SYSDATE + 20);

COMMIT;

-- Scenario 1: Senior Citizen Loan Discount

BEGIN

FOR loan\_rec IN (SELECT LoanID, InterestRate, CustomerID FROM Loans) LOOP

DECLARE

customer\_age NUMBER;

BEGIN

SELECT TRUNC(MONTHS\_BETWEEN(SYSDATE, DOB)/12)

INTO customer\_age

FROM Customers

WHERE CustomerID = loan\_rec.CustomerID;

IF customer\_age > 60 THEN

UPDATE Loans

SET InterestRate = InterestRate - 1

WHERE LoanID = loan\_rec.LoanID;

END IF;

END;

END LOOP;

COMMIT;

END;

/

-- Scenario 2: Promote VIP Customers

BEGIN

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

IF cust\_rec.Balance > 10000 THEN

UPDATE Customers

SET IsVIP = 'TRUE'

WHERE CustomerID = cust\_rec.CustomerID;

END IF;

END LOOP;

COMMIT;

END;

/

-- Scenario 3: Loan Due Reminder

BEGIN

FOR loan\_rec IN (

SELECT LoanID, CustomerID, EndDate

FROM Loans

WHERE EndDate BETWEEN SYSDATE AND SYSDATE + 30

) LOOP

DBMS\_OUTPUT.PUT\_LINE('Reminder: Loan ' || loan\_rec.LoanID || ' for Customer ' || loan\_rec.CustomerID || ' due on ' || TO\_CHAR(loan\_rec.EndDate, 'DD-MON-YYYY'));

END LOOP;

END;

/

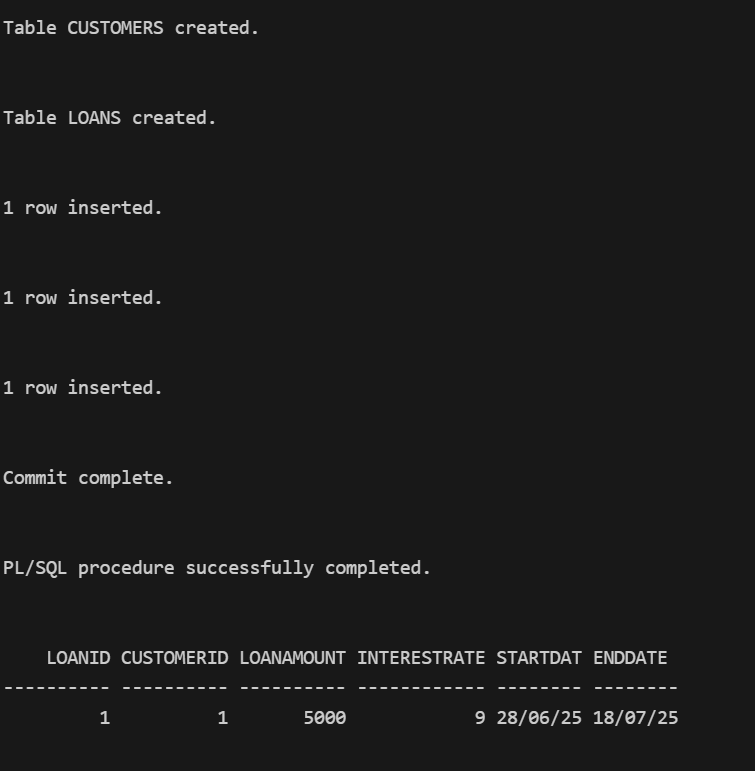
COMMANDS:

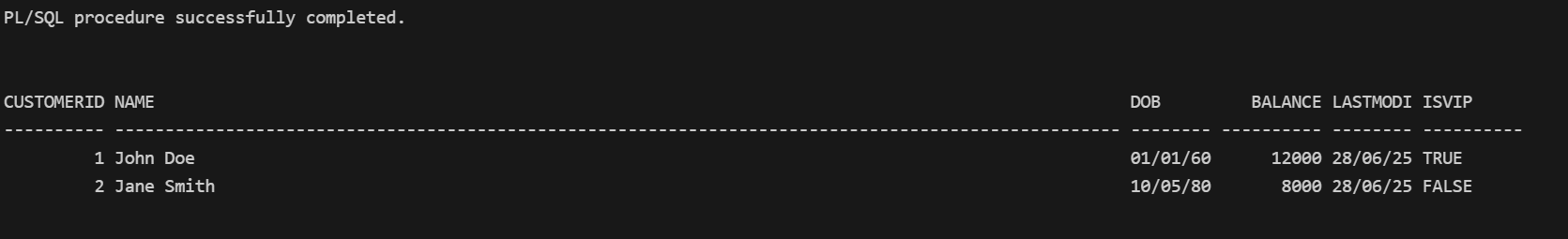
SELECT \* FROM Loans;

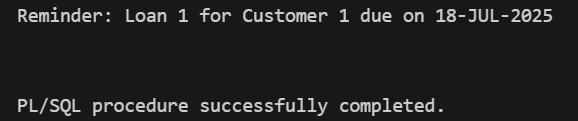
SELECT \* FROM Customers;

SET SERVEROUTPUT ON;

OUTPUT:







EXCERCISE-3:STORED PROCEDURES:

CODE:

SET SERVEROUTPUT ON;

-- Create Tables

CREATE TABLE Accounts (

AccountID NUMBER PRIMARY KEY,

CustomerID NUMBER,

AccountType VARCHAR2(20),

Balance NUMBER,

LastModified DATE

);

CREATE TABLE Employees (

EmployeeID NUMBER PRIMARY KEY,

Name VARCHAR2(100),

Position VARCHAR2(50),

Salary NUMBER,

Department VARCHAR2(50),

HireDate DATE

);

-- Insert Sample Data

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

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

INSERT INTO Employees VALUES (1, 'Alice', 'Manager', 50000, 'HR', SYSDATE);

INSERT INTO Employees VALUES (2, 'Bob', 'Developer', 60000, 'IT', SYSDATE);

COMMIT;

-- Procedure 1: Monthly Interest Process

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

FOR acc\_rec IN (

SELECT AccountID, Balance

FROM Accounts

WHERE AccountType = 'Savings'

) LOOP

UPDATE Accounts

SET Balance = Balance + (Balance \* 0.01)

WHERE AccountID = acc\_rec.AccountID;

END LOOP;

COMMIT;

END;

/

-- Procedure 2: Update Employee Bonus

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(

dept\_name IN VARCHAR2,

bonus\_percent IN NUMBER

) IS

BEGIN

UPDATE Employees

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

WHERE Department = dept\_name;

COMMIT;

END;

/

-- Procedure 3: Transfer Funds Between Accounts

CREATE OR REPLACE PROCEDURE TransferFunds(

from\_account IN NUMBER,

to\_account IN NUMBER,

amount IN NUMBER

) IS

source\_balance NUMBER;

BEGIN

SELECT Balance INTO source\_balance FROM Accounts WHERE AccountID = from\_account;

IF source\_balance >= amount THEN

UPDATE Accounts

SET Balance = Balance - amount

WHERE AccountID = from\_account;

UPDATE Accounts

SET Balance = Balance + amount

WHERE AccountID = to\_account;

COMMIT;

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

ELSE

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

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

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

END;

/

COMMANDS:

EXEC ProcessMonthlyInterest;

SELECT \* FROM Accounts;

EXEC UpdateEmployeeBonus('HR', 10);

SELECT \* FROM Employees;

SET SERVEROUTPUT ON;

EXEC TransferFunds(1, 2, 500);

SELECT \* FROM Accounts;

OUTPUTS:

