Week-2

PL SQL programming

**Exercise 2: Error Handling**

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE Transactions';

EXCEPTION

WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF;

END;

/

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE Accounts';

EXCEPTION

WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF;

END;

/

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE Loans';

EXCEPTION

WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF;

END;

/

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE Employees';

EXCEPTION

WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF;

END;

/

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE Customers';

EXCEPTION

WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF;

END;

/

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

);

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'));

COMMIT;

SET SERVEROUTPUT ON SIZE 1000000;

-- Exercise 2: Error Handling

-- Scenario 1: Handle exceptions during fund transfers between accounts

CREATE OR REPLACE PROCEDURE SafeTransferFunds(from\_acc NUMBER, to\_acc NUMBER, amt NUMBER) IS

BEGIN

UPDATE Accounts SET Balance = Balance - amt WHERE AccountID = from\_acc;

IF SQL%ROWCOUNT = 0 THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Source account not found.');

END IF;

UPDATE Accounts SET Balance = Balance + amt WHERE AccountID = to\_acc;

IF SQL%ROWCOUNT = 0 THEN

RAISE\_APPLICATION\_ERROR(-20002, 'Target account not found.');

END IF;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE('Error in SafeTransferFunds: '||SQLERRM);

END;

/

-- Scenario 2: Manage errors when updating employee salaries

CREATE OR REPLACE PROCEDURE UpdateSalary(emp\_id NUMBER, percent\_increase NUMBER) IS

BEGIN

UPDATE Employees SET Salary = Salary + (Salary \* percent\_increase / 100) WHERE EmployeeID = emp\_id;

IF SQL%ROWCOUNT = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Employee not found with ID: '||emp\_id);

ELSE

DBMS\_OUTPUT.PUT\_LINE('Salary updated for EmployeeID: '||emp\_id);

END IF;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error in UpdateSalary: '||SQLERRM);

END;

/

-- Scenario 3: Ensure data integrity when adding a new customer

CREATE OR REPLACE PROCEDURE AddNewCustomer(p\_id NUMBER, p\_name VARCHAR2, p\_dob DATE, p\_balance NUMBER) IS

BEGIN

INSERT INTO Customers VALUES (p\_id, p\_name, p\_dob, p\_balance, SYSDATE);

DBMS\_OUTPUT.PUT\_LINE('Customer inserted: '||p\_name);

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('Customer with ID '||p\_id||' already exists.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error in AddNewCustomer: '||SQLERRM);

END;

/

BEGIN

-- Test SafeTransferFunds

SafeTransferFunds(1, 2, 100);

END;

/

BEGIN

-- Test UpdateSalary

UpdateSalary(1, 10); -- increase Alice's salary by 10%

END;

/

BEGIN

-- Test AddNewCustomer

AddNewCustomer(3, 'Sam Wilson', TO\_DATE('1995-04-12', 'YYYY-MM-DD'), 2000);

END;

/

