**Exercise 2: Error Handling**

**Scenario 1:** Handle exceptions during fund transfers between accounts.

* + **Question:** Write a stored procedure **SafeTransferFunds** that transfers funds between two accounts. Ensure that if any error occurs (e.g., insufficient funds), an appropriate error message is logged and the transaction is rolled back.

CREATE OR REPLACE PROCEDURE SafeTransferFunds

(p\_fromAccountID IN NUMBER,p\_toAccountID IN NUMBER,p\_amount IN NUMBER)

IS

    v\_balance NUMBER;

BEGIN

    -- Check balance of fromAccount

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

    IF v\_balance < p\_amount THEN

        RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient funds in the source account.');

    END IF;

    -- Deduct amount from fromAccount

    UPDATE Accounts SET Balance = Balance - p\_amount, LastModified = SYSDATE

    WHERE AccountID = p\_fromAccountID;

    -- Add amount to toAccount

    UPDATE Accounts SET Balance = Balance + p\_amount, LastModified = SYSDATE

    WHERE AccountID = p\_toAccountID;

EXCEPTION

    WHEN OTHERS THEN

        ROLLBACK;

        DBMS\_OUTPUT.PUT\_LINE('An error occurred: ' || SQLERRM);

        RAISE;

END SafeTransferFunds;

/

**Scenario 2:** Manage errors when updating employee salaries.

* + **Question:** Write a stored procedure **UpdateSalary** that increases the salary of an employee by a given percentage. If the employee ID does not exist, handle the exception, and log an error message.

CREATE OR REPLACE PROCEDURE UpdateSalary (emp\_ID NUMBER,per NUMBER)

IS

BEGIN

    UPDATE Employees

    SET Salary = Salary \* (1 + per / 100) WHERE EmployeeID = emp\_id;

    DBMS\_OUTPUT.PUT\_LINE('Updated salary of Employee ID '||emp\_id);

  IF SQL%ROWCOUNT = 0 THEN

        RAISE no\_data\_found;

  END IF;

  EXCEPTION

        WHEN no\_data\_found THEN

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

  WHEN OTHERS THEN

          DBMS\_OUTPUT.PUT\_LINE('An error occurred: ' || SQLERRM);

END UpdateSalary;

/

**Scenario 3:** Ensure data integrity when adding a new customer.

* + **Question:** Write a stored procedure **AddNewCustomer** that inserts a new customer into the Customers table. If a customer with the same ID already exists, handle the exception by logging an error and preventing the insertion.

CREATE OR REPLACE PROCEDURE AddNewCustomer (

    p\_customerID IN NUMBER,

    p\_name IN VARCHAR2,

    p\_dob IN DATE,

    p\_balance IN NUMBER

) IS

BEGIN

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

    EXCEPTION

        WHEN DUP\_VAL\_ON\_INDEX THEN

            DBMS\_OUTPUT.PUT\_LINE('Error: Customer ID ' || p\_customerID || ' already exists.');

        WHEN OTHERS THEN

            DBMS\_OUTPUT.PUT\_LINE('An error occurred: ' || SQLERRM);

END AddNewCustomer;

/