PL/SQL Exercise 2 - Error Handling

# Scenario 1: Safe Fund Transfer with Error Handling

CREATE OR REPLACE PROCEDURE SafeTransferFunds (

p\_fromAccountID IN NUMBER,

p\_toAccountID IN NUMBER,

p\_amount IN NUMBER

)

IS

insufficient\_funds EXCEPTION;

v\_balance NUMBER;

BEGIN

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

IF v\_balance < p\_amount THEN

RAISE insufficient\_funds;

END IF;

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

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

COMMIT;

EXCEPTION

WHEN insufficient\_funds THEN

ROLLBACK;

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

WHEN OTHERS THEN

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE('Transfer failed: ' || SQLERRM);

END;

# Scenario 2: Update Employee Salary with Exception for Missing Employee

CREATE OR REPLACE PROCEDURE UpdateSalary (

p\_employeeID IN NUMBER,

p\_percentage IN NUMBER

)

IS

v\_salary NUMBER;

BEGIN

SELECT Salary INTO v\_salary FROM Employees WHERE EmployeeID = p\_employeeID;

UPDATE Employees

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

WHERE EmployeeID = p\_employeeID;

COMMIT;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Employee ID ' || p\_employeeID || ' not found.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Update failed: ' || SQLERRM);

END;

# Scenario 3: Add New Customer with Duplicate Check

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 (CustomerID, Name, DOB, Balance, LastModified)

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

COMMIT;

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('Insert failed: ' || SQLERRM);

END;

# Output:

