**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.

**MY ANSWERS :**

CREATE OR REPLACE PROCEDURE SafeTransferFunds (

p\_fromAcct IN NUMBER,

p\_toAcct IN NUMBER,

p\_amount IN NUMBER

) IS

e\_insufficient\_funds EXCEPTION;

v\_fromBal NUMBER;

BEGIN

SELECT Balance

INTO v\_fromBal

FROM Accounts

WHERE AccountID = p\_fromAcct

FOR UPDATE;

IF v\_fromBal < p\_amount THEN

RAISE e\_insufficient\_funds;

END IF;

UPDATE Accounts

SET Balance = Balance - p\_amount

WHERE AccountID = p\_fromAcct;

UPDATE Accounts

SET Balance = Balance + p\_amount

WHERE AccountID = p\_toAcct;

COMMIT;

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

EXCEPTION

WHEN e\_insufficient\_funds THEN

ROLLBACK;

INSERT INTO ErrorLog (ProcName, ErrMsg)

VALUES ('SafeTransferFunds', 'Insufficient funds in source account');

DBMS\_OUTPUT.PUT\_LINE(' Transfer failed – insufficient funds.');

WHEN OTHERS THEN

ROLLBACK;

INSERT INTO ErrorLog (ProcName, ErrMsg)

VALUES ('SafeTransferFunds', SQLERRM);

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

END;

/

**TestCase with output :**

SET SERVEROUTPUT ON;

🡪Success case: from 1 → 2, amount 200

EXEC SafeTransferFunds (1, 2, 200);

🡪Output: Transfer successful.

🡪 Failure case: try 2 000 from acct 1 (only 800 left!)

EXEC SafeTransferFunds (1, 2, 2000);

🡪 Output: Transfer failed – insufficient funds.

**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.

**My Answers :**

CREATE OR REPLACE PROCEDURE UpdateSalary (

p\_empID IN Employees.EmployeeID%TYPE,

p\_percent IN NUMBER -- e.g. 10 for +10 %

) IS

v\_dummy NUMBER;

BEGIN

SELECT 1 INTO v\_dummy

FROM Employees

WHERE EmployeeID = p\_empID;

UPDATE Employees

SET Salary = Salary \* (1 + p\_percent/100)

WHERE EmployeeID = p\_empID;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Salary updated for Employee '||p\_empID);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

ROLLBACK;

INSERT INTO ErrorLog (ProcName, ErrMsg)

VALUES ('UpdateSalary', 'Employee ID not found: '||p\_empID);

DBMS\_OUTPUT.PUT\_LINE('Error: employee not found.');

WHEN OTHERS THEN

ROLLBACK;

INSERT INTO ErrorLog (ProcName, ErrMsg)

VALUES ('UpdateSalary', SQLERRM);

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

END;

/

**TestCase with output :**

🡪 Good ID

EXEC UpdateSalary(1, 5);

🡪 Output: Salary updated for Employee 1

🡪 Bad ID

EXEC UpdateSalary(99, 5);

🡪 Output: Error: employee not found.

**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.

**My Answers :**

CREATE OR REPLACE PROCEDURE AddNewCustomer (

p\_custID IN Customers.CustomerID%TYPE,

p\_name IN Customers.Name%TYPE,

p\_dob IN DATE,

p\_balance IN NUMBER

) IS

BEGIN

INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)

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

COMMIT;

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

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

ROLLBACK;

INSERT INTO ErrorLog (ProcName, ErrMsg)

VALUES ('AddNewCustomer', 'Duplicate CustomerID: '||p\_custID);

DBMS\_OUTPUT.PUT\_LINE(' Error: customer already exists.');

WHEN OTHERS THEN

ROLLBACK;

INSERT INTO ErrorLog (ProcName, ErrMsg)

VALUES ('AddNewCustomer', SQLERRM);

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

END;

/

**TestCase with output :**

🡪 Insert new customer OK

EXEC AddNewCustomer(5,'New Guy',DATE '1995-01-01',500);

🡪 Output: Customer added: New Guy

🡪 Try same ID again

EXEC AddNewCustomer(5,'Clash',DATE '1990-01-01',700);

🡪 Output: Error: customer already exists.