**Scenario 1: Handle Exceptions During Fund Transfers Between Accounts**

**Stored Procedure: SafeTransferFunds**

CREATE OR REPLACE PROCEDURE SafeTransferFunds(

p\_FromAccountID NUMBER,

p\_ToAccountID NUMBER,

p\_Amount NUMBER

) AS

v\_FromBalance NUMBER;

v\_ToBalance NUMBER;

BEGIN

SELECT Balance INTO v\_FromBalance

FROM Accounts

WHERE AccountID = p\_FromAccountID

FOR UPDATE;

IF v\_FromBalance < p\_Amount THEN

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

END IF;

UPDATE Accounts

SET Balance = Balance - p\_Amount, LastModified = SYSDATE

WHERE AccountID = p\_FromAccountID;

UPDATE Accounts

SET Balance = Balance + p\_Amount, LastModified = SYSDATE

WHERE AccountID = p\_ToAccountID;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Transfer successful from Account ' || p\_FromAccountID || ' to Account ' || p\_ToAccountID || ' of amount ' || p\_Amount);

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

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

END SafeTransferFunds;

/

**Scenario 2: Manage Errors When Updating Employee Salaries**

**Stored Procedure: UpdateSalary**

CREATE OR REPLACE PROCEDURE UpdateSalary(

p\_EmployeeID NUMBER,

p\_PercentIncrease NUMBER

) AS

v\_Salary NUMBER;

BEGIN

SELECT Salary INTO v\_Salary

FROM Employees

WHERE EmployeeID = p\_EmployeeID

FOR UPDATE;

v\_Salary := v\_Salary \* (1 + p\_PercentIncrease / 100);

UPDATE Employees

SET Salary = v\_Salary, HireDate = SYSDATE

WHERE EmployeeID = p\_EmployeeID;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Salary updated for Employee ID ' || p\_EmployeeID || '. New Salary: ' || v\_Salary);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Employee with ID ' || p\_EmployeeID || ' does not exist.');

WHEN OTHERS THEN

ROLLBACK;

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

END UpdateSalary;

/

**Scenario 3: Ensure Data Integrity When Adding a New Customer**

**Stored Procedure: AddNewCustomer**

CREATE OR REPLACE PROCEDURE AddNewCustomer(

p\_CustomerID NUMBER,

p\_Name VARCHAR2,

p\_DOB DATE,

p\_Balance NUMBER

) AS

BEGIN

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

VALUES (p\_CustomerID, p\_Name, p\_DOB, p\_Balance, SYSDATE);

COMMIT;

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

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

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

WHEN OTHERS THEN

ROLLBACK;

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

END AddNewCustomer;

/