**Exercise 2: Error Handling**

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

CREATE OR REPLACE PROCEDURE SafeTransferFunds(

p\_from\_account\_id IN NUMBER,

p\_to\_account\_id IN NUMBER,

p\_amount IN NUMBER

)

IS

v\_balance NUMBER;

BEGIN

SELECT balance INTO v\_balance FROM accounts WHERE account\_id = p\_from\_account\_id;

IF v\_balance < p\_amount THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient funds for transfer.');

END IF;

UPDATE accounts SET balance = balance - p\_amount WHERE account\_id = p\_from\_account\_id;

UPDATE accounts SET balance = balance + p\_amount WHERE account\_id = p\_to\_account\_id;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Transfer of ' || p\_amount || ' completed successfully.');

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

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

END; /

**Output:**

Sample call:

EXEC SafeTransferFunds(101, 102, 500);

Transfer of 500 completed successfully.

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

CREATE OR REPLACE PROCEDURE UpdateSalary(

p\_emp\_id IN NUMBER,

p\_percent IN NUMBER

)

IS

v\_old\_salary NUMBER;

BEGIN

SELECT salary INTO v\_old\_salary FROM employees WHERE emp\_id = p\_emp\_id;

UPDATE employees

SET salary = salary + (salary \* p\_percent / 100)

WHERE emp\_id = p\_emp\_id;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Salary updated from ' || v\_old\_salary || ' to ' ||

(v\_old\_salary + (v\_old\_salary \* p\_percent / 100)));

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

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

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Unexpected error while updating salary: ' || SQLERRM);

END;

/

**Sample Call:**

EXEC UpdateSalary(201, 10);

**Output:**

Salary updated from 30000 to 33000

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

CREATE OR REPLACE PROCEDURE AddNewCustomer(

p\_customer\_id IN NUMBER,

p\_name IN VARCHAR2,

p\_age IN NUMBER,

p\_balance IN NUMBER

)

IS

BEGIN

INSERT INTO customers(customer\_id, name, age, balance)

VALUES(p\_customer\_id, p\_name, p\_age, p\_balance);

COMMIT;

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

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

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

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Unexpected error while adding customer: ' || SQLERRM);

END; /

**Output:**

**Sample Call:**

EXEC AddNewCustomer(301, 'Kiran', 35, 8000);

Customer Kiran added successfully.