-- Scenario 1: Handle exceptions during fund transfers between accounts

CREATE PROCEDURE SafeTransferFunds(from\_account\_id NUMBER, to\_account\_id NUMBER, amount NUMBER) AS

BEGIN

    BEGIN

        UPDATE Accounts

        SET Balance = Balance - amount

        WHERE AccountID = from\_account\_id;

        UPDATE Accounts

        SET Balance = Balance + amount

        WHERE AccountID = to\_account\_id;

    EXCEPTION

        WHEN OTHERS THEN

            ROLLBACK;

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

    END;

END;

-- Scenario 2: Manage errors when updating employee salaries

CREATE PROCEDURE UpdateSalary(employee\_id NUMBER, percentage NUMBER) AS

BEGIN

    BEGIN

        UPDATE Employees

        SET Salary = Salary \* (1 + percentage / 100)

        WHERE EmployeeID = employee\_id;

    EXCEPTION

        WHEN NO\_DATA\_FOUND THEN

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

    END;

END;

-- Scenario 3: Ensure data integrity when adding a new customer

CREATE PROCEDURE AddNewCustomer(customer\_id NUMBER, name VARCHAR2, dob DATE) AS

BEGIN

    BEGIN

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

        VALUES (customer\_id, name, dob, 0, SYSDATE);

    EXCEPTION

        WHEN DUP\_VAL\_ON\_INDEX THEN

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

    END;

END;