**EXERCISE 2 : ERROR HANDLING**

USE bankdb;

-- ==========================================

-- Scenario 1: SafeTransferFunds Stored Procedure

-- ==========================================

DELIMITER //

CREATE PROCEDURE SafeTransferFunds(

IN from\_account\_id INT,

IN to\_account\_id INT,

IN transfer\_amount DECIMAL(10,2)

)

BEGIN

DECLARE from\_balance DECIMAL(10,2);

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

SELECT 'Transfer failed. Transaction rolled back due to an error.' AS ErrorMessage;

END;

START TRANSACTION;

-- Get source balance

SELECT Balance INTO from\_balance

FROM Accounts

WHERE AccountID = from\_account\_id

FOR UPDATE;

-- Check for sufficient funds

IF from\_balance < transfer\_amount THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Insufficient funds in source account';

END IF;

-- Perform transfer

UPDATE Accounts SET Balance = Balance - transfer\_amount WHERE AccountID = from\_account\_id;

UPDATE Accounts SET Balance = Balance + transfer\_amount WHERE AccountID = to\_account\_id;

COMMIT;

SELECT 'Transfer successful.' AS Message;

END;

//

-- ==========================================

-- Scenario 2: UpdateSalary Stored Procedure

-- ==========================================

CREATE PROCEDURE UpdateSalary(

IN emp\_id INT,

IN percent\_increase DECIMAL(5,2)

)

BEGIN

DECLARE emp\_count INT;

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

SELECT CONCAT('Error: Failed to update salary for EmployeeID ', emp\_id) AS ErrorMessage;

END;

-- Check if employee exists

SELECT COUNT(\*) INTO emp\_count FROM Employees WHERE EmployeeID = emp\_id;

IF emp\_count = 0 THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Employee does not exist';

END IF;

-- Update salary

UPDATE Employees

SET Salary = Salary + (Salary \* percent\_increase / 100)

WHERE EmployeeID = emp\_id;

SELECT 'Salary updated successfully.' AS Message;

END;

//

-- ==========================================

-- Scenario 3: AddNewCustomer Stored Procedure

-- ==========================================

CREATE PROCEDURE AddNewCustomer(

IN cust\_id INT,

IN cust\_name VARCHAR(100),

IN dob DATE,

IN balance DECIMAL(10,2)

)

BEGIN

DECLARE cust\_count INT;

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

SELECT 'Error inserting new customer. Possible duplicate ID.' AS ErrorMessage;

END;

-- Check if customer already exists

SELECT COUNT(\*) INTO cust\_count

FROM Customers

WHERE CustomerID = cust\_id;

IF cust\_count > 0 THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Customer already exists with this ID';

END IF;

-- Insert new customer

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

VALUES (cust\_id, cust\_name, dob, balance, NOW());

SELECT 'Customer added successfully.' AS Message;

END;

//

DELIMITER ;

**OUTPUT:**

USE bankdb;

SELECT \* FROM Accounts;

SELECT \* FROM Employees;

SELECT \* FROM Customers;

CREATE TABLE ErrorLog (ErrorMsg TEXT, LoggedAt DATETIME DEFAULT NOW());

SELECT \* FROM ErrorLog;







