Employee Management System SQL Server Exercises

Database Schema

```
CREATE TABLE Departments (
  DepartmentID INT PRIMARY KEY,
  DepartmentName VARCHAR(100) NOT NULL
);

CREATE TABLE Employees (
  EmployeeID INT PRIMARY KEY,
  FirstName VARCHAR(50),
  LastName VARCHAR(50),
  Email VARCHAR(100) UNIQUE,
  Salary DECIMAL(10, 2),
  DepartmentID INT,
  FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID)
);

CREATE TABLE AuditLog (
  LogID INT IDENTITY(1,1) PRIMARY KEY,
  Action VARCHAR(100),
  ErrorMessage VARCHAR(4000),
  ActionDate DATETIME DEFAULT GETDATE()
);
```

Question 1: Basic TRY...CATCH for Error Logging

Scenario: You want to insert a new employee, but sometimes the email might already exist.

Task:
1. Write a stored procedure AddEmployee that:
   - Accepts employee details.
   - Tries to insert into Employees.
   - If an error occurs, catches it and logs the error message into AuditLog.

Learning Outcome: Understand basic TRY...CATCH and error logging.

Question 2: Using THROW to Re-raise Errors

Scenario: You want to log the error but also let the application know something went wrong.

Task:
1. Modify the AddEmployee procedure:
   - After logging the error in AuditLog, use THROW to re-raise the error.

Learning Outcome: Learn how to propagate errors after handling them.

Question 3: Custom Error with RAISERROR

Scenario: You want to validate that salary must be greater than 0.

Task:
1. In the AddEmployee procedure:
   - Before inserting, check if Salary <= 0.
   - If so, use RAISERROR to raise a custom error: 'Salary must be greater than zero.'

Learning Outcome: Use RAISERROR for custom business rule enforcement.

Question 4: Nested TRY...CATCH with RAISERROR

Scenario: You want to simulate a nested error handling scenario.

Task:
1. Create a procedure TransferEmployee that:
   - Updates an employee's department.
   - If the department doesn't exist, raise a custom error.
   - Use nested TRY...CATCH to catch and log the error, then re-raise it.

Learning Outcome: Understand nested error handling and control flow.

Question 5: Logging All Errors in a Transaction

Scenario: You want to ensure that either all changes succeed or none do.

Task:
1. Create a procedure BatchInsertEmployees:

- Accepts multiple employee records (simulate with multiple inserts).
- Wrap the inserts in a transaction.
- If any insert fails, roll back and log the error.

Learning Outcome: Combine transactions with TRY...CATCH and error logging.

Question 6: Dynamic RAISERROR with Severity and State

Scenario: You want to raise different errors based on different conditions.

Task:
1. Modify AddEmployee:
   - If salary is too low (< 1000), raise a warning (severity 10).
   - If salary is negative, raise an error (severity 16).
   - Use RAISERROR with different severity levels.

Learning Outcome: Learn how severity and state affect error behavior.