**BÁO CÁO BÀI TẬP LỚN SỐ 2 MÔN HỆ CƠ SỞ DỮ LIỆU**

**HK231 - Lớp: L02**

**Nhóm**: SV58

**Danh sách thành viên**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stt | Họ tên | MSSV | Nội dung thực hiện | Ký tên xác nhận |
| 1 | Nguyễn Đinh Bằng | 2210298 | Tạo bảng, insert dữ liệu, function, procedure, trigger, backend kết nối database, web |  |
| 2 | Nguyễn Quốc Thịnh | 2213296 | Tạo bảng, thêm dữ liệu, sửa EERD, ánh xạ EERD sang lược đồ quan hệ, trigger, procedure, function |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Chủ đề**: Trang web bán khoá học online

**1. Tạo bảng và dữ liệu mẫu**

**1.1 Các câu lệnh tạo bảng và ràng buộc**

CREATE TABLE …

…

CREATE TRIGGER … (*các trigger liên quan đến kiểm tra ràng buộc dữ liệu, nếu có*)

**1.2 Câu lệnh thêm dữ liệu**

INSERT INTO …

(*Nếu dữ liệu được nhập trực tiếp vào DBMS hoặc giao diện ứng dụng thì không cần ghi nhóm câu lệnh INSERT trên. Tuy nhiên, khuyến khích các nhóm chuẩn bị sẵn các câu lệnh INSERT để dùng khi test dữ liệu hoặc khôi phục lại dữ liệu gốc khi cần.*)

Kết quả dữ liệu của các bảng trong CSDL.

A screen shot of a computer

Description automatically generated

A close up of a screen

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a message box

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**2. Trigger – Thủ tục – Hàm**

**2.1 Thủ tục INSERT / UPDATE / DELETE dữ liệu vào 1 bảng dữ liệu**

Bảng dữ liệu: Coupon (phiếu giảm giá)

1. Thủ tục INSERT:

**Chức năng của thủ tục**

Thủ tục InsertCoupon dùng để thêm một mã giảm giá (coupon) mới vào cơ sở dữ liệu, với các ràng buộc để đảm bảo tính hợp lệ của dữ liệu đầu vào.

**Use Case**

Thủ tục này được sử dụng trong các trường hợp sau:

* Khi cần thêm mã giảm giá cho các chiến dịch khuyến mãi.
* Đảm bảo rằng các thông tin của mã giảm giá (như tiêu đề, giá trị, loại giảm giá, ngày hiệu lực, v.v.) tuân theo các quy định về tính hợp lệ và không trùng lặp.

**Các thao tác chính**

1. **Kiểm tra dữ liệu đầu vào**:
   * Xác nhận rằng các trường bắt buộc không được để trống.
   * Kiểm tra các ràng buộc liên quan đến loại mã giảm giá (percent hoặc fixed), giá trị, và các ngày bắt đầu/kết thúc.
2. **Kiểm tra tính duy nhất**:
   * Xác nhận rằng tiêu đề mã giảm giá không bị trùng lặp.
3. **Thêm dữ liệu vào bảng**:
   * Thực hiện lệnh INSERT dữ liệu hợp lệ vào bảng Coupon trong cơ sở dữ liệu.
4. **Xử lý lỗi**:
   * Nếu có lỗi xảy ra, các thao tác sẽ được hoàn tác (rollback transaction), và thông báo lỗi sẽ được trả về.

Các ràng buộc được kiểm tra bao gồm:

* Các giá trị không được để trống.
* Loại mã giảm giá chỉ có thể là percent hoặc fixed.
* Giá trị phần trăm phải nằm trong khoảng từ 0 đến 100 (đối với loại percent).
* Giá trị giảm giá phải lớn hơn 0 (đối với loại fixed).
* Ngày bắt đầu không được ở quá khứ, và phải trước ngày hết hạn.
* Tiêu đề mã giảm giá không được trùng lặp với mã giảm giá đã tồn tại trong cơ sở dữ liệu.
* Giá trị giảm giá tối đa (nếu có) không được vượt quá giá trị giảm giá.

Input: Các thông số đầu vào của thủ tục bao gồm:

* @CouponTitle (NVARCHAR(255)): Tiêu đề mã giảm giá.
* @CouponValue (INT): Giá trị mã giảm giá (có thể là số tiền hoặc phần trăm).
* @CouponType (NVARCHAR(255)): Loại mã giảm giá (percent hoặc fixed).
* @CouponStartDate (DATE): Ngày bắt đầu hiệu lực của mã giảm giá.
* @CouponExpire (DATE): Ngày hết hạn của mã giảm giá.
* @CouponMaxDiscount (INT): Giá trị giảm giá tối đa (nếu có).
* @ErrorMessage (NVARCHAR(255), OUTPUT): Biến để lưu thông báo lỗi (nếu có).

Output:

* Nếu thành công: Thông báo "Thêm mã giảm giá thành công." được hiển thị.
* Nếu thất bại: Thông báo lỗi sẽ được trả về thông qua biến @ErrorMessage.

Câu lệnh tạo thủ tục:

*-- INSERT COUPON*

GO

CREATE PROCEDURE InsertCoupon

    @CouponTitle NVARCHAR(255),

    @CouponValue INT,

    @CouponType NVARCHAR(255),

    @CouponStartDate DATE,

    @CouponExpire DATE,

    @CouponMaxDiscount INT,

    @ErrorMessage NVARCHAR(255) OUTPUT

AS

BEGIN

    SET NOCOUNT ON;

    BEGIN TRY

*-- Validate nulls*

        IF @CouponTitle IS NULL OR @CouponValue IS NULL OR @CouponType IS NULL OR

           @CouponStartDate IS NULL OR @CouponExpire IS NULL OR @CouponMaxDiscount IS NULL

        BEGIN

            PRINT 'Lỗi: Dữ liệu không được để trống.'

            RAISERROR('Dữ liệu không được để trống.', 16, 1)

            RETURN

        END

*-- Validate title*

        IF LEN(TRIM(@CouponTitle)) = 0

        BEGIN

            PRINT 'Lỗi: Tiêu đề mã giảm giá không được để trống.'

            RAISERROR('Tiêu đề mã giảm giá không được để trống.', 16, 1)

            RETURN

        END

*-- Validate type*

        IF @CouponType NOT IN ('percent', 'fixed')

        BEGIN

            PRINT 'Lỗi: Loại mã giảm giá không hợp lệ.'

            RAISERROR('Loại mã giảm giá không hợp lệ.', 16, 1)

            RETURN

        END

*-- Validate value*

        IF @CouponValue <= 0

        BEGIN

            PRINT 'Lỗi: Giá trị mã giảm giá phải lớn hơn 0.'

            RAISERROR('Giá trị mã giảm giá phải lớn hơn 0.', 16, 1)

            RETURN

        END

*-- Validate dates*

        IF @CouponStartDate < GETDATE()

        BEGIN

            PRINT 'Lỗi: Ngày bắt đầu không được trong quá khứ.'

            RAISERROR('Ngày bắt đầu không được trong quá khứ.', 16, 1)

            RETURN

        END

        IF @CouponStartDate >= @CouponExpire

        BEGIN

            PRINT 'Lỗi: Ngày bắt đầu mã giảm giá phải trước ngày hết hạn.'

            RAISERROR('Ngày bắt đầu mã giảm giá phải trước ngày hết hạn.', 16, 1)

            RETURN

        END

*-- Validate max discount*

        IF @CouponMaxDiscount < 0

        BEGIN

            PRINT 'Lỗi: Giá trị giảm giá tối đa phải lớn hơn hoặc bằng 0.'

            RAISERROR('Giá trị giảm giá tối đa phải lớn hơn hoặc bằng 0.', 16, 1)

            RETURN

        END

*-- Validate percentage type*

        IF @CouponType = 'percent' AND (@CouponValue > 100 OR @CouponValue < 0)

        BEGIN

            PRINT 'Lỗi: Giá trị phần trăm phải nằm trong khoảng từ 0 đến 100.'

            RAISERROR('Giá trị phần trăm phải nằm trong khoảng từ 0 đến 100.', 16, 1)

            RETURN

        END

*-- Validate value type*

        IF @CouponType = 'fixed' AND @CouponValue <= 0

        BEGIN

            PRINT 'Lỗi: Giá trị giảm giá phải lớn hơn 0.'

            RAISERROR('Giá trị giảm giá phải lớn hơn 0.', 16, 1)

            RETURN

        END

*-- Check duplicate title*

        IF EXISTS (SELECT 1 FROM Coupon WHERE CouponTitle = @CouponTitle)

        BEGIN

            PRINT 'Lỗi: Tiêu đề mã giảm giá đã tồn tại.'

            RAISERROR('Tiêu đề mã giảm giá đã tồn tại.', 16, 1)

            RETURN

        END

*--nếu type = giá trị thì max discount phải <= value*

        IF @CouponType = 'fixed' AND @CouponMaxDiscount > @CouponValue

        BEGIN

            PRINT 'Lỗi: Giá trị giảm giá tối đa phải nhỏ hơn hoặc bằng giá trị giảm giá.'

            RAISERROR('Giá trị giảm giá tối đa phải nhỏ hơn hoặc bằng giá trị giảm giá.', 16, 1)

            RETURN

        END

*-- Insert with transaction*

        BEGIN TRANSACTION

            INSERT INTO Coupon (

                CouponTitle,

                CouponValue,

                CouponType,

                CouponStartDate,

                CouponExpire,

                CouponMaxDiscount

            )

            VALUES (

                @CouponTitle,

                @CouponValue,

                @CouponType,

                @CouponStartDate,

                @CouponExpire,

                @CouponMaxDiscount

            )

        COMMIT TRANSACTION

        PRINT 'Thêm mã giảm giá thành công.'

    END TRY

    BEGIN CATCH

        IF @@TRANCOUNT > 0

            ROLLBACK TRANSACTION

*-- PRINT 'Lỗi: ' + ERROR\_MESSAGE()*

        SET @ErrorMessage = ERROR\_MESSAGE()

        PRINT @ErrorMessage

        RETURN

    END CATCH

END

GO

Nhóm câu lệnh thực thi thủ tục mẫu:

DECLARE @ErrorMessage NVARCHAR(255);

EXEC InsertCoupon 'SUMMER2021', 20, 'percent', '2025-06-01', '2026-08-31', 100, @ErrorMessage OUTPUT;

Kết quả các màn hình hiển thị từ DBMS minh họa cho việc thực hiện thủ tục

A blue and white screen with red text

Description automatically generated

Trước khi thực hiện thủ tục:

A screenshot of a computer

Description automatically generated

Sau khi thực hiện thủ tục: Một coupon mới được thêm vào (SUMMER 2028, ID = 17)

A screenshot of a computer

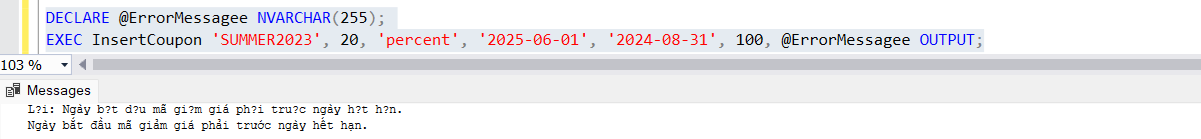
Description automatically generated

Khi ngày bắt đầu của mã giảm giá nhỏ hơn thời điểm hiện tại:

A white rectangular object with red numbers

Description automatically generated

Khi ngày bắt đầu nằm sau ngày hết hạn:



Khi tiêu đề mã giảm giá bị trùng

A blue and white rectangular object with red numbers

Description automatically generated

Khi loại mã giảm giá là “phần trăm” và giá trị mã giảm giá ngoài khoảng từ 0-100:

A white rectangular object with red text

Description automatically generated

Khi loại mã giảm giá là “giá trị” và giá trị giảm giá tối đa lớn hơn giá trị mã giảm giá:

A white text with red numbers

Description automatically generated

Khi insert một mã giảm giá có loại không phải là “fixed” hoặc “percent”:

A blue and white digital display

Description automatically generated with medium confidence

Ảnh chụp trước khi insert dữ liệu: Có 5 mã giảm giá.

A screenshot of a computer

Description automatically generated

Thực hiện insert một mã giảm giá mới: A screenshot of a computer

Description automatically generated

Sau khi đã insert: Có thêm mã giảm giá “WINTER”, tổng cộng có 6 mã giảm giá.A screenshot of a computer

Description automatically generated

Ảnh báo lỗi khi insert ngày bắt đầu trong quá khứ:

A screenshot of a computer

Description automatically generated

Ảnh báo lỗi khi ngày bắt đầu sau ngày kết thúc:

A screenshot of a computer

Description automatically generated

Ảnh báo lỗi khi loại mã giảm giá là “phần trăm” và giá trị lớn hơn 100 hoặc nhỏ hơn 0:

A screenshot of a computer

Description automatically generated

Ảnh báo lỗi khi loại mã giảm giá là “giá trị” và giá trị giảm giá tối đa lớn hơn giá trị mã giảm giá

A screenshot of a computer

Description automatically generated

1. Thủ tục UPDATE:

**Chức năng của thủ tục**

* Thủ tục UpdateCoupon được sử dụng để cập nhật thông tin của một mã giảm giá (coupon) đã tồn tại trong cơ sở dữ liệu, đồng thời đảm bảo các ràng buộc về tính hợp lệ của dữ liệu đầu vào.

**Use Case**

Thủ tục này được sử dụng trong các trường hợp:

* Khi cần thay đổi thông tin mã giảm giá, như tiêu đề, giá trị, loại giảm giá, ngày hiệu lực, hoặc giá trị giảm giá tối đa.
* Đảm bảo rằng các thông tin cập nhật không vi phạm các ràng buộc hoặc làm hỏng dữ liệu hiện có.

**Các thao tác chính**

1. **Kiểm tra tính tồn tại**:
   * Xác nhận rằng mã giảm giá cần cập nhật (CouponID) tồn tại trong cơ sở dữ liệu.
2. **Kiểm tra dữ liệu đầu vào**:
   * Ngày bắt đầu phải không nằm trong quá khứ và phải trước ngày hết hạn.
   * Giá trị giảm giá tối đa phải lớn hơn 0.
   * Loại mã giảm giá phải hợp lệ (percent hoặc fixed).
   * Đối với loại percent, giá trị phần trăm phải nằm trong khoảng từ 0 đến 100.
   * Đối với loại fixed, giá trị giảm giá phải lớn hơn 0.
3. **Cập nhật thông tin mã giảm giá**:
   * Thực hiện cập nhật các trường thông tin liên quan của mã giảm giá trong bảng Coupon.
4. **Xử lý lỗi**:
   * Nếu có lỗi xảy ra trong quá trình cập nhật, giao dịch sẽ được hoàn tác (rollback transaction), và thông báo lỗi sẽ được trả về.

***Mô tả thủ tục: UpdateCoupon***

***Chức năng của thủ tục***

*Thủ tục UpdateCoupon được sử dụng để cập nhật thông tin của một mã giảm giá (coupon) đã tồn tại trong cơ sở dữ liệu, đồng thời đảm bảo các ràng buộc về tính hợp lệ của dữ liệu đầu vào.*

***Use Case***

*Thủ tục này được sử dụng trong các trường hợp:*

* *Khi cần thay đổi thông tin mã giảm giá, như tiêu đề, giá trị, loại giảm giá, ngày hiệu lực, hoặc giá trị giảm giá tối đa.*
* *Đảm bảo rằng các thông tin cập nhật không vi phạm các ràng buộc hoặc làm hỏng dữ liệu hiện có.*

***Các thao tác chính***

1. ***Kiểm tra tính tồn tại****:*
   * *Xác nhận rằng mã giảm giá cần cập nhật (CouponID) tồn tại trong cơ sở dữ liệu.*
2. ***Kiểm tra dữ liệu đầu vào****:*
   * *Ngày bắt đầu phải không nằm trong quá khứ và phải trước ngày hết hạn.*
   * *Giá trị giảm giá tối đa phải lớn hơn 0.*
   * *Loại mã giảm giá phải hợp lệ (percent hoặc fixed).*
   * *Đối với loại percent, giá trị phần trăm phải nằm trong khoảng từ 0 đến 100.*
   * *Đối với loại fixed, giá trị giảm giá phải lớn hơn 0.*
3. ***Cập nhật thông tin mã giảm giá****:*
   * *Thực hiện cập nhật các trường thông tin liên quan của mã giảm giá trong bảng Coupon.*
4. ***Xử lý lỗi****:*
   * *Nếu có lỗi xảy ra trong quá trình cập nhật, giao dịch sẽ được hoàn tác (rollback transaction), và thông báo lỗi sẽ được trả về.*

**Input**

Các thông số đầu vào của thủ tục bao gồm:

* @CouponID (VARCHAR(20)): Mã định danh của mã giảm giá cần cập nhật.
* @CouponTitle (NVARCHAR(255)): Tiêu đề mã giảm giá mới.
* @CouponValue (INT): Giá trị mã giảm giá mới.
* @CouponType (VARCHAR(255)): Loại mã giảm giá mới (percent hoặc fixed).
* @CouponStartDate (DATE): Ngày bắt đầu hiệu lực mới của mã giảm giá.
* @CouponExpire (DATE): Ngày hết hạn mới của mã giảm giá.
* @CouponMaxDiscount (DECIMAL(10,2)): Giá trị giảm giá tối đa mới.
* @ErrorMessage (NVARCHAR(255), OUTPUT): Biến để lưu thông báo lỗi (nếu có)

**Output**

* **Kết quả trả về**:
  + Nếu thành công: Thông báo "Cập nhật mã giảm giá thành công." sẽ được hiển thị.
  + Nếu thất bại: Thông báo lỗi sẽ được trả về thông qua biến @ErrorMessage.

Các ràng buộc được kiểm tra trong thủ tục:

* Mã giảm giá cần cập nhật (CouponID) phải tồn tại trong bảng Coupon.
* Ngày bắt đầu không được ở quá khứ và phải trước ngày kết thúc.
* Loại mã giảm giá chỉ có thể là percent hoặc fixed.
* Giá trị phần trăm (percent) phải nằm trong khoảng từ 0 đến 100.
* Giá trị giảm giá (fixed) phải lớn hơn 0.
* Giá trị giảm giá tối đa phải lớn hơn 0.

Các câu lệnh tạo thủ tục:

|  |
| --- |
| *--UPDATE COUPON*  CREATE PROCEDURE UpdateCoupon      @CouponID VARCHAR(20),      @CouponTitle NVARCHAR(255),      @CouponValue INT,      @CouponType VARCHAR(255),      @CouponStartDate DATE,      @CouponExpire DATE,      @CouponMaxDiscount DECIMAL(10,2),      @ErrorMessage NVARCHAR(255) OUTPUT  AS  BEGIN      BEGIN TRY  *-- Validate input*          IF @CouponStartDate < GETDATE()          BEGIN              RAISERROR('Ngày bắt đầu không được trong quá khứ', 16, 1)              RETURN          END          IF @CouponStartDate > @CouponExpire          BEGIN              RAISERROR('Ngày bắt đầu không được lớn hơn ngày kết thúc', 16, 1)              RETURN          END            IF @CouponMaxDiscount <= 0          BEGIN              RAISERROR('Giá trị giảm giá phải lớn hơn 0', 16, 1)              RETURN          END  *--Validate coupon type*          IF @CouponType NOT IN ('percent', 'fixed')          BEGIN              RAISERROR('Loại mã giảm giá không hợp lệ', 16, 1)              RETURN          END  *--Validate coupon percentage range*          IF @CouponType = 'percent' AND (@CouponValue <= 0 OR @CouponValue > 100)          BEGIN              RAISERROR('Giá trị phần trăm phải nằm trong khoảng từ 0 đến 100', 16, 1)              RETURN          END  *--Validate coupon value*          IF @CouponType = 'fixed' AND @CouponValue <= 0          BEGIN              RAISERROR('Giá trị giảm giá phải lớn hơn 0', 16, 1)              RETURN          END  *--check if coupon exists*          IF NOT EXISTS (SELECT 1 FROM Coupon WHERE CouponID = @CouponID)          BEGIN              RAISERROR(N'Không tìm thấy mã giảm giá', 16, 1)              RETURN          END            BEGIN TRANSACTION              UPDATE Coupon              SET                  CouponTitle = @CouponTitle,                  CouponValue = @CouponValue,                  CouponType = @CouponType,                  CouponStartDate = @CouponStartDate,                  CouponExpire = @CouponExpire,                  CouponMaxDiscount = @CouponMaxDiscount              WHERE CouponID = @CouponID              IF @@ROWCOUNT = 0              BEGIN                  RAISERROR('Không tìm thấy mã giảm giá', 16, 1)                  ROLLBACK TRANSACTION                  RETURN              END          COMMIT TRANSACTION          PRINT 'Cập nhật mã giảm giá thành công.'      END TRY      BEGIN CATCH          IF @@TRANCOUNT > 0              ROLLBACK TRANSACTION          PRINT 'Lỗi: ' + ERROR\_MESSAGE()          SET @ErrorMessage = ERROR\_MESSAGE()          RETURN      END CATCH  END  GO |

Nhóm câu lệnh thực thi thủ tục mẫu:

|  |
| --- |
| *DECLARE @ErrorMessage2 NVARCHAR(255);*  *EXEC UpdateCoupon '1', 'SUMMER2022', 25, 'percent', '2025-06-01', '2026-08-31', 100, @ErrorMessage2 OUTPUT;* |

Màn hình DBMS minh hoạ việc thực hiện thủ tục: Sửa đổi mã giảm giá có ID = 1 thành “SUMMER2022”:

A screenshot of a computer

Description automatically generated

Báo lỗi khi loại mã giảm giá là “phần trăm” và giá trị ngoài khoảng 0-100:

A blue and white bar with red numbers

Description automatically generated

Khi chỉnh sửa mã giảm giá và sửa loại mã giảm giá thành một chuỗi khác ngoài “fixed” và “percent”:

A close-up of a computer screen

Description automatically generated

Khi loại mã giảm giá là “giá trị” và giá trị giảm giá tối đa lớn hơn giá trị mã giảm giá:



Minh hoạ khi kết nối với web:

Trước khi chỉnh sửa:

A screenshot of a computer

Description automatically generated

Bấm vào “Edit” và chỉnh sửa tiêu đề thành “SUMMER2025”:

A screenshot of a coupon

Description automatically generated

Nhấn “Update”: Tiêu đề của mã giảm giá đã đổi thành SUMMER2025

A screenshot of a computer

Description automatically generated

Minh hoạ báo lỗi khi chỉnh sửa mã giảm giá vi phạm các điều kiện:

A screenshot of a computer

Description automatically generated

1. Thủ tục DELETE:

**Chức năng của thủ tục**

Thủ tục DeleteCoupon được sử dụng để xóa một mã giảm giá (coupon) khỏi cơ sở dữ liệu, đồng thời xóa các dữ liệu liên quan trong bảng phụ thuộc (Edit).

**Use Case**

Thủ tục này được sử dụng trong các trường hợp:

* Khi cần hủy bỏ mã giảm giá không còn sử dụng.
* Đảm bảo dữ liệu liên quan (như các chỉnh sửa lịch sử) được xóa đồng thời để duy trì tính toàn vẹn dữ liệu.

Thủ tục đảm bảo rằng:

* Mã giảm giá chỉ được xóa khi nó tồn tại trong cơ sở dữ liệu.
* Các dữ liệu liên quan (chẳng hạn như lịch sử chỉnh sửa) cũng được xóa để đảm bảo không còn dữ liệu "mồ côi" (orphaned records).
* Sử dụng giao dịch (transaction) để đảm bảo tính toàn vẹn dữ liệu, đặc biệt khi có nhiều bảng phụ thuộc.

**Các thao tác chính**

1. **Kiểm tra tính tồn tại**:
   * Xác nhận rằng mã giảm giá cần xóa (CouponID) tồn tại trong bảng Coupon.
2. **Xóa dữ liệu liên quan**:
   * Xóa các bản ghi trong bảng phụ thuộc (Edit) có liên kết với mã giảm giá.
3. **Xóa mã giảm giá**:
   * Thực hiện xóa mã giảm giá khỏi bảng Coupon.
4. **Xử lý lỗi**:
   * Nếu có lỗi xảy ra trong quá trình xóa, giao dịch sẽ được hoàn tác (rollback transaction), và thông báo lỗi sẽ được hiển thị.

**Input**

Các thông số đầu vào của thủ tục bao gồm:

* @CouponID (VARCHAR(20)): Mã định danh của mã giảm giá cần xóa.

**Output**

* **Kết quả trả về**:
  + Nếu thành công: Thông báo "Xóa mã giảm giá thành công." sẽ được hiển thị.
  + Nếu thất bại: Thông báo lỗi sẽ được hiển thị qua cơ chế RAISERROR.

**Các bước thực hiện**

1. **Kiểm tra tính tồn tại của mã giảm giá**:
   * Nếu không tìm thấy mã giảm giá, thủ tục sẽ kết thúc và thông báo lỗi "Không tìm thấy mã giảm giá".
2. **Xóa dữ liệu liên quan trong bảng Edit**:
   * Xóa các bản ghi có liên kết với mã giảm giá thông qua cột EditCouponID.
3. **Xóa mã giảm giá trong bảng Coupon**:
   * Xóa bản ghi có mã định danh trùng với @CouponID.
   * Nếu không thể xóa, thông báo lỗi "Không thể xóa mã giảm giá" sẽ được hiển thị.
4. **Xử lý giao dịch**:
   * Nếu tất cả các bước thành công, giao dịch sẽ được hoàn tất (commit transaction).
   * Nếu xảy ra lỗi trong bất kỳ bước nào, giao dịch sẽ được hoàn tác (rollback transaction).

Các câu lệnh tạo thủ tục:

|  |
| --- |
| *--DELETE COUPON*  CREATE PROCEDURE DeleteCoupon      @CouponID VARCHAR(20)  AS  BEGIN      BEGIN TRY          BEGIN TRANSACTION  *-- Check if coupon exists*              IF NOT EXISTS (SELECT 1 FROM Coupon WHERE CouponID = @CouponID)              BEGIN                  RAISERROR('Không tìm thấy mã giảm giá', 16, 1)                  RETURN              END  *--DELETE Edit*              DELETE FROM Edit              WHERE EditCouponID = @CouponID  *-- Delete the coupon*              DELETE FROM Coupon              WHERE CouponID = @CouponID              IF @@ROWCOUNT > 0                  PRINT 'Xóa mã giảm giá thành công.'              ELSE                  RAISERROR('Không thể xóa mã giảm giá', 16, 1)          COMMIT TRANSACTION      END TRY      BEGIN CATCH          IF @@TRANCOUNT > 0              ROLLBACK TRANSACTION          PRINT 'Lỗi: ' + ERROR\_MESSAGE()      END CATCH  END  GO |

Minh hoạ việc sử dụng thủ tục:

*EXEC DeleteCoupon '1';*

Màn hình DBMS trước khi thực hiện thủ tục delete:

*A screenshot of a computer

Description automatically generated*

Sau khi thực hiện thủ tục delete với id = 7:

*A screenshot of a computer

Description automatically generated*

Nếu input là một coupon id không tồn tại:

A screenshot of a computer

Description automatically generated

Minh hoạ việc kết nối với web: Trước khi nhấn “Delete” coupon id = 6:

**A screenshot of a computer

Description automatically generated**

Sau khi nhấn vào Delete coupon id = 6, coupon đó biến mất: A screenshot of a computer

Description automatically generated

**2.2 TRIGGER**

*Lưu ý: các trigger để kiểm tra các ràng buộc dữ liệu mà có thể định nghĩa được trong câu lệnh CREATE TABLE sẽ không được tính điểm.*

*Ví dụ: kiểm tra thuộc tính MSSV có duy nhất không (Unique), kiểm tra email có đúng định dạng không (dùng REGULAR EXPRESSION hoặc Like), số điện thoại sinh viên không được để trống (Not null)*

1. **Trigger 1**

**Mô tả nhóm trigger**

Đây là nhóm gồm 2 trigger phối hợp với nhau để tự động tính toán và cập nhật tổng giá trị đơn hàng khi có bất kỳ thay đổi nào liên quan đến đơn hàng, bao gồm:

* TR\_Orders\_Calculate\_Total: Kích hoạt khi có thay đổi trong bảng Orders (thêm đơn hàng mới hoặc cập nhật mã giảm giá)
* TR\_CourseOrder\_Calculate\_Total: Kích hoạt khi có thay đổi trong bảng CourseOrder (thêm/sửa/xóa khóa học trong đơn hàng)

Nhóm trigger này được sử dụng trong các trường hợp:

* Khi tạo đơn hàng mới
* Khi thêm hoặc xóa khóa học khỏi đơn hàng
* Khi thay đổi mã giảm giá áp dụng cho đơn hàng
* Đảm bảo tổng giá trị đơn hàng luôn được cập nhật tự động và chính xác

**Các thao tác chính**

1. Thu thập thông tin đơn hàng:
   * Xác định OrderID cần cập nhật
   * Lấy thông tin về mã giảm giá (nếu có)
   * Tính tổng giá các khóa học trong đơn hàng
2. Nếu thay đổi liên quan đến đơn hàng đã được thanh toán thì sẽ báo lỗi
3. Xử lý mã giảm giá:
   * Kiểm tra tính hợp lệ và thời hạn của mã giảm giá
   * Tính toán giá trị giảm dựa trên loại giảm giá:
     + Phần trăm: Tính % trên tổng giá trị
     + Giá trị cố định: Áp dụng trực tiếp số tiền giảm
4. Áp dụng các ràng buộc giảm giá:
   * Đảm bảo giá trị giảm không vượt quá giới hạn tối đa
   * Với giảm giá cố định: không vượt quá tổng giá trị đơn hàng
   * Với giảm giá phần trăm: tính chính xác số tiền được giảm
5. Cập nhật đơn hàng:
   * Tính tổng giá trị cuối cùng sau khi áp dụng giảm giá
   * Cập nhật vào trường TotalAmount của bảng Orders

Câu lệnh tạo trigger:

|  |
| --- |
| CREATE TRIGGER TR\_Orders\_Calculate\_Total  ON Orders  AFTER UPDATE  AS  BEGIN  -- Declare variables  DECLARE @OrderID INT  DECLARE @PaymentStatus NVARCHAR(255)  -- Get OrderID and payment status from inserted record  SELECT @OrderID = OrderID, @PaymentStatus = OrderPaymentStatus  FROM inserted  -- Check if order is already paid  IF EXISTS (  SELECT 1  FROM Orders  WHERE OrderID = @OrderID  AND OrderPaymentStatus = 'paid'  )  BEGIN  RAISERROR ('Không thể thay đổi đơn hàng đã thanh toán.', 16, 1)  ROLLBACK TRANSACTION  RETURN  END  -- Declare variables  DECLARE @CouponID INT  DECLARE @SubTotal FLOAT  DECLARE @Discount FLOAT  DECLARE @FinalTotal FLOAT  DECLARE @CouponType NVARCHAR(255)  DECLARE @CouponValue INT  DECLARE @CouponMaxDiscount INT  DECLARE @CurrentDate DATE    -- Get current date  SET @CurrentDate = GETDATE()  -- Get the affected OrderID from inserted records  SELECT @OrderID = OrderID, @CouponID = CouponID  FROM inserted  -- Calculate subtotal by summing up course prices  SELECT @SubTotal = COALESCE(SUM(c.CoursePrice), 0)  FROM CourseOrder co  JOIN Course c ON co.CourseID = c.CourseID  WHERE co.OrderID = @OrderID  -- If there's a coupon, calculate discount  IF @CouponID IS NOT NULL  BEGIN  -- Get coupon details with date validation  SELECT  @CouponType = CouponType,  @CouponValue = CouponValue,  @CouponMaxDiscount = CouponMaxDiscount  FROM Coupon  WHERE CouponID = @CouponID  AND CouponStartDate <= @CurrentDate  AND CouponExpire >= @CurrentDate  -- Only calculate discount if coupon is valid (not NULL from previous query)  IF @CouponType IS NOT NULL  BEGIN  -- Calculate discount based on coupon type  IF @CouponType = 'percent'  BEGIN  SET @Discount = (@SubTotal \* @CouponValue) / 100.0  -- Check if discount exceeds max discount  IF @Discount > @CouponMaxDiscount  SET @Discount = @CouponMaxDiscount  END  ELSE -- Fixed value  BEGIN  SET @Discount = @CouponValue  -- Check if discount exceeds max discount  IF @Discount > @CouponMaxDiscount  SET @Discount = @CouponMaxDiscount  -- Check if discount exceeds subtotal  IF @Discount > @SubTotal  SET @Discount = @SubTotal  END  END  ELSE  BEGIN  SET @Discount = 0 -- Coupon is not valid due to dates  END  END  ELSE  BEGIN  SET @Discount = 0  END  -- Calculate final total  SET @FinalTotal = @SubTotal - @Discount  -- Update the TotalAmount in Orders table  UPDATE Orders  SET TotalAmount = @FinalTotal  WHERE OrderID = @OrderID  END; |

|  |
| --- |
| CREATE TRIGGER TR\_CourseOrder\_Calculate\_Total  ON CourseOrder  AFTER INSERT, UPDATE, DELETE  AS  BEGIN  IF EXISTS (  SELECT 1  FROM Orders o  WHERE o.OrderPaymentStatus = 'paid'  AND o.OrderID IN (  SELECT OrderID FROM inserted  UNION  SELECT OrderID FROM deleted  )  )  BEGIN  RAISERROR ('Không thể thay đổi đơn hàng đã thanh toán.', 16, 1)  ROLLBACK TRANSACTION  RETURN  END  DECLARE @CurrentDate DATE  SET @CurrentDate = GETDATE()  UPDATE Orders  SET TotalAmount = (  SELECT COALESCE(SUM(c.CoursePrice), 0) -  CASE  WHEN cp.CouponType = N'percent'  AND cp.CouponStartDate <= @CurrentDate  AND cp.CouponExpire >= @CurrentDate THEN  CASE  WHEN (COALESCE(SUM(c.CoursePrice), 0) \* cp.CouponValue / 100.0) > cp.CouponMaxDiscount  THEN cp.CouponMaxDiscount  ELSE (COALESCE(SUM(c.CoursePrice), 0) \* cp.CouponValue / 100.0)  END  WHEN cp.CouponType = N'value'  AND cp.CouponStartDate <= @CurrentDate  AND cp.CouponExpire >= @CurrentDate THEN  CASE  WHEN cp.CouponValue > cp.CouponMaxDiscount THEN cp.CouponMaxDiscount  WHEN cp.CouponValue > COALESCE(SUM(c.CoursePrice), 0) THEN COALESCE(SUM(c.CoursePrice), 0)  ELSE cp.CouponValue  END  ELSE 0  END  FROM CourseOrder co2  JOIN Course c ON co2.CourseID = c.CourseID  LEFT JOIN Orders o ON co2.OrderID = o.OrderID  LEFT JOIN Coupon cp ON o.CouponID = cp.CouponID  WHERE co2.OrderID = Orders.OrderID  GROUP BY co2.OrderID, cp.CouponType, cp.CouponValue, cp.CouponMaxDiscount,  cp.CouponStartDate, cp.CouponExpire  )  WHERE OrderID IN (  SELECT OrderID FROM inserted  UNION  SELECT OrderID FROM deleted  );  END; |

**Testcase:**

Tạo đơn hàng mới có OrderID là 103, lấy ngày hiện tại bằng hàm GETDATE(), tổng giá trị đơn hàng ban đầu là 0 và chưa có mã giảm giá nào

|  |
| --- |
| INSERT INTO Orders (OrderID, OrderPaymentStatus, OrderDate, StudentID,TotalAmount,CouponID)  VALUES  (103, 'unpaid', GETDATE(), 15,0,NULL); |

Kiểm tra đơn hàng có OrderID là 103

|  |
| --- |
| SELECT \*  FROM Orders  WHERE OrderID=103 |

Kết quả như sau, TotalAmmount bằng 0 vì đơn hàng này chưa có khóa học nào



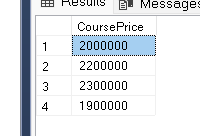
Thêm 2 khóa học có CourseID là 1 và 2 vào đơn hàng có id là 103 trong bảng CourseOrder

|  |
| --- |
| INSERT INTO CourseOrder( CourseID,OrderID)  VALUES  (1,103),  (2,103); |

Kiểm tra giá tiền của 2 khóa học có id là 1 , 2, 3 và 4

|  |
| --- |
| SELECT CoursePrice  FROM Course  WHERE CourseID IN(1,2,3,4) |

Kết quả như sau:



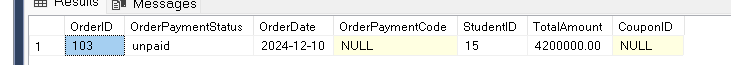
Khóa học 1 có giá là 2 000 000

Khóa học 2 có giá là 2 200 000

Khóa học 3 có giá 2 300 000

Khóa học 4 có giá 1 900 000

Ta kiểm tra lại thông tin đơn hàng có ID là 103, kết quả như sau:



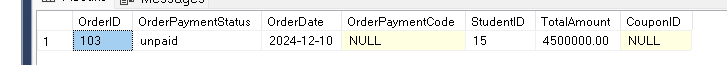
Vậy trigger TR\_CourseOrder\_Calculate\_Total đã được chạy đẻ tính toán tổng tiền khóa học khi có thay đổi trên bảng CourseOrder là 2 000 000 + 2 200 000 = 4 200 000

Thử UPDATE bảng CourseOrder có OrderID là 103 , có khóa học 1 thay thành khóa học 3

|  |
| --- |
| UPDATE CourseOrder  SET CourseID=3  WHERE CourseID=1 AND OrderID=103 |

Khi đó đơn hàng có OrderID là 103 sẽ có khóa học 2 và khóa học 3, tổng tiền là:

2 200 000 + 2 300 000 = 4 500 000



Thử DELETE bảng CourseOrder có id đơn hàng là 103, khóa học 2

|  |
| --- |
| DELETE FROM CourseOrder  WHERE CourseID=2 AND OrderID=103 |

Khi đó đơn hàng có OrderID là 103 chỉ còn khóa học 3, tổng tiền sẽ là 2 300 000



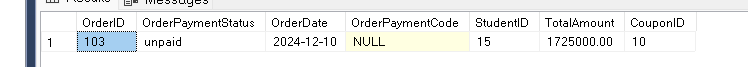
INSERT phiếu giảm giá vào:

|  |
| --- |
| INSERT INTO Coupon (CouponID,CouponTitle, CouponValue, CouponType, CouponStartDate, CouponExpire, CouponMaxDiscount)  VALUES  -- Giảm theo phần trăm  (9,N'Giảm 15% khóa học mới', 15,'percent' , '2024-01-01', '2025-04-28', 1500000),  (10,N'Giảm 25% khóa học sql', 25, 'percent', '2024-05-01', '2025-03-28', 1500000),  (11,N'Giảm 300k nhân diệp giáng sinh', 300000, 'value', '2024-01-15', '2024-12-30', 1500000); |

Thử UPDATE đơn hàng có OrderID là 103 , gắn phiếu giảm giá có CouponID là 10 cho đơn hàng

|  |
| --- |
| UPDATE Orders  SET CouponID=10  WHERE OrderID=103 |

Khi đó đơn hàng có OrderID là 103 sẽ giảm giá 25%, giá trị giảm là 2 300 000 x 25%= 575 000 , vì giá trị giảm bé hơn 1 500 000 nên giữ nguyên giá trị giảm này, khi đó giá trị đơn hàng còn : 2 300 000 – 575 000 = 1 725 000



Ta cập nhật CourseOrder để đơn hàng có OrderID là 103 có thêm khóa học 1, 2 và 4

|  |
| --- |
| INSERT INTO CourseOrder( CourseID,OrderID)  VALUES  (1,103),  (2,103),  (4,103); |

Khi đó tổng giá trị khóa học sẽ là

2 000 000 + 2 200 00 + 2 300 000 + 1 900 000 = 8 400 000

Vì áp dụng phiếu giảm giá 25%, nên giá trị giảm sẽ là 8 400 000 x 25%=2 100 000, vì giá trị giảm này lớn hơn 1 500 000 nên chỉ được giảm 1 500 000, tổng giá tiền khóa học sẽ còn:

8 400 000 – 1 500 000= 6 900 000



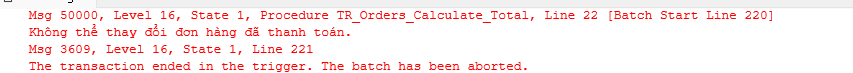
Ta thêm 1 đơn hàng có ID là 104, với trạng thái thanh toán là “paid”

|  |
| --- |
| INSERT INTO Orders (OrderID, OrderPaymentStatus, OrderDate, StudentID,TotalAmount,CouponID)  VALUES  (104, 'paid', GETDATE(), 15,0,NULL); |

Nếu ta thử gắn phiếu giảm giá cho đơn hàng này

|  |
| --- |
| UPDATE Orders  SET CouponID=10  WHERE OrderID=104 |

Sẽ in ra lỗi, vì đơn hàng đã thanh toán rồi không thay đổi được nữa



**b. Trigger 2**

**Mô tả trigger:**

Chức năng của trigger Trigger TR\_Review\_Update\_Course\_Rating được sử dụng để tự động tính toán và cập nhật điểm đánh giá trung bình của khóa học (CourseAverageRating) mỗi khi có thay đổi trong bảng Review. Trigger được kích hoạt (fire) sau khi có bất kỳ thao tác thêm, sửa, xóa đánh giá (AFTER INSERT, UPDATE, DELETE).

Trigger này được sử dụng trong các trường hợp:

* Khi học viên thêm đánh giá mới cho khóa học
* Khi học viên chỉnh sửa đánh giá đã tồn tại
* Khi đánh giá bị xóa khỏi hệ thống
* Đảm bảo điểm đánh giá trung bình của khóa học luôn được cập nhật tự động và chính xác

**Các thao tác chính**

1. Thu thập các CourseID bị ảnh hưởng:
   * Lấy danh sách CourseID từ cả các đánh giá mới/được cập nhật (inserted)
   * Lấy danh sách CourseID từ các đánh giá bị xóa/thay đổi (deleted)
   * Kết hợp (UNION) các CourseID để có danh sách đầy đủ các khóa học cần cập nhật
2. Tính toán điểm trung bình:
   * Tính trung bình cộng của tất cả ReviewScore cho mỗi khóa học
   * Sử dụng CAST để chuyển đổi điểm số thành kiểu FLOAT để có kết quả chính xác
   * Nhóm theo CourseID để tính riêng cho từng khóa học
3. Cập nhật khóa học:
   * Cập nhật giá trị CourseAverageRating trong bảng Course
   * Chỉ cập nhật các khóa học bị ảnh hưởng bởi thay đổi

Câu lệnh tạo (các) trigger:

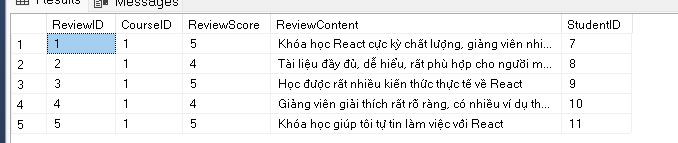
|  |
| --- |
| CREATE OR ALTER TRIGGER TR\_Review\_Update\_Course\_Rating  ON Review  AFTER INSERT, UPDATE, DELETE  AS  BEGIN  -- Update Course average rating for all affected courses  UPDATE Course  SET CourseAverageRating = (  SELECT AVG(CAST(ReviewScore AS FLOAT))  FROM Review r  WHERE r.CourseID = Course.CourseID  GROUP BY r.CourseID  )  WHERE CourseID IN (  -- Get affected CourseIDs from both inserted and deleted  SELECT CourseID FROM inserted  UNION  SELECT CourseID FROM deleted  );  END; |

**Test case**

Đầu tiên ta kiểm tra dữ liệu mẫu bảng Review, kiểm tra các đánh giá cho các khóa học có CourseID=1

|  |
| --- |
| SELECT \*  FROM Review  WHERE CourseID=1 |

Kết quả:



Vậy ban đầu Course 1 có 5 đánh giá, điểm số đánh giá lần lượt là 5, 4, 5, 4, 5

Khi thêm một đánh giá cho Course 1 bằng câu lệnh như sau:

|  |
| --- |
| INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES  (6, 1, 1, N'Khóa học này quá mông lung', 15); |

Vậy đánh giá trung bình sau khi thêm dữ liệu vô là: (5 + 4 + 5 + 4 + 5 + 1)/6 =4

Ta kiểm tra CourseAverageRating của Course có CourseID là 1

|  |
| --- |
| SELECT CourseAverageRating  FROM Course  WHERE CourseID=1 |

Kết quả như sau:



Ta xóa đi một hàng trong Review có ReviewID là 1 và CourseID là 1

Khi đó đánh giá trung bình sẽ là : (4+5+4+5+1)/5=3.8

|  |
| --- |
| DELETE FROM Review  WHERE ReviewID=1 AND CourseID=1 |

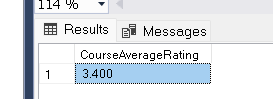
Ta kiểm tra lại CourseAverageRating của Course có CourseID là 1



Ta cập nhật Review có ReviewID là 2 và CourseID là 1, cập nhật lại ReviewScore là 2

|  |
| --- |
| UPDATE Review  SET ReviewScore=2  WHERE ReviewID=2 AND CourseID=1 |

Khi đó đánh giá trung bình sẽ là : (2+5+4+5+1)/5=3.4



**2.3 THỦ TỤC**

1. **Thủ tục 1**

**Mô tả thủ tục:** Thủ tục GetCoursesInCategoryByMinRating được sử dụng để tìm kiếm và hiển thị thông tin các khóa học trong một danh mục cụ thể, có điểm đánh giá trung bình cao hơn hoặc bằng một mức điểm cho trước.

Thủ tục này được sử dụng trong các trường hợp:

* Khi người dùng muốn tìm kiếm khóa học theo danh mục
* Khi cần lọc khóa học theo mức đánh giá tối thiểu
* Khi cần xem thống kê về các khóa học chất lượng cao trong một danh mục
* Hỗ trợ việc đề xuất khóa học cho học viên dựa trên đánh giá

**Các thao tác chính:**

1. Kiểm tra tính hợp lệ của đầu vào:
   * Rating phải nằm trong khoảng từ 1 đến 5
2. Tìm kiếm khóa học theo tiêu chí:
   * Tên danh mục chứa chuỗi tìm kiếm
   * Điểm đánh giá trung bình đạt ngưỡng tối thiểu
   * Có ít nhất một đánh giá
3. Trả về thông tin chi tiết:
   * Tên danh mục và khóa học
   * Số lượng đánh giá
   * Điểm trung bình
   * Giá và thời gian khóa học

Input:

 @CategoryName (NVARCHAR(255)): Tên danh mục cần tìm kiếm

 @MinRating (DECIMAL(10,3)): Điểm đánh giá tối thiểu

Output:

Kết quả trả về bảng gồm các cột:

* CategoryName: Tên danh mục
* CourseName: Tên khóa học
* TotalReviews: Tổng số đánh giá
* CourseAverageRating: Điểm trung bình
* CoursePrice: Giá khóa học
* CourseStartDate: Ngày bắt đầu
* CourseEndDate: Ngày kết thúc

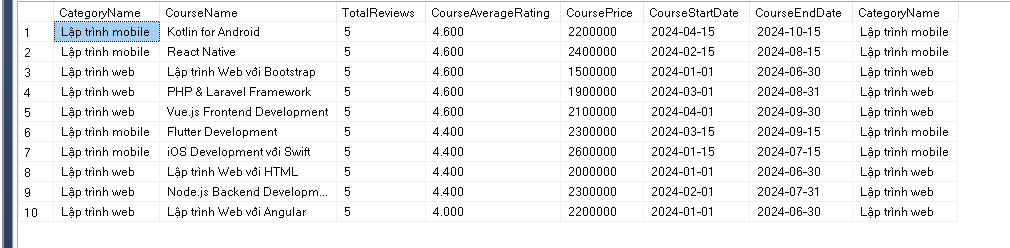
Câu lệnh tạo thủ tục:

|  |
| --- |
| CREATE PROCEDURE GetCoursesInCategoryByMinRating  @CategoryName NVARCHAR(255),  @MinRating DECIMAL(10,3)  AS  BEGIN  -- Kiểm tra tham số đầu vào  IF @MinRating < 1 OR @MinRating > 5  BEGIN  RAISERROR('Rating phải nằm trong khoảng từ 1 đến 5', 16, 1)  RETURN  END    SELECT  cat.CategoryName,  c.CourseName,  COUNT(r.ReviewID) as TotalReviews,  c.CourseAverageRating,  c.CoursePrice,  c.CourseStartDate,  c.CourseEndDate,  cat.CategoryName  FROM Course c  INNER JOIN Category cat ON c.CategoryID = cat.CategoryID  LEFT JOIN Review r ON c.CourseID = r.CourseID  WHERE  cat.CategoryName LIKE N'%' + @CategoryName + '%'  AND c.CourseAverageRating >= @MinRating  GROUP BY  cat.CategoryName,  c.CourseName,  c.CourseAverageRating,  c.CoursePrice,  c.CourseStartDate,  c.CourseEndDate,  cat.CategoryName  HAVING  COUNT(r.ReviewID) > 0 -- Thay đổi điều kiện HAVING  ORDER BY  c.CourseAverageRating DESC;  END; |

Nhóm câu lệnh thực thi thủ tục mẫu:

|  |
| --- |
| EXEC GetCoursesInCategoryByMinRating @CategoryName = N'Lập trình', @MinRating = 4.0 |

Kết quả các màn hình hiển thị từ DBMS minh họa cho việc thực hiện thủ tục



1. **Thủ tục 2**

**Mô tả thủ tục:** Thủ tục sp\_GetValidCoupons được sử dụng để lấy danh sách các mã giảm giá còn hiệu lực tại thời điểm hiện tại, kèm theo thống kê số lượng các khóa học đã áp dụng mã giảm giá đó**.** Thủ tục này được sử dụng trong các trường hợp:

* Kiểm tra các mã giảm giá còn hiệu lực
* Thống kê mức độ sử dụng của các mã giảm giá
* Quản lý và theo dõi hiệu quả của các chương trình khuyến mãi
* Hỗ trợ việc ra quyết định về gia hạn hoặc kết thúc các mã giảm giá

**Các thao tác chính:**

1. Lấy thời điểm hiện tại:
   * Sử dụng hàm GETDATE() để lấy ngày hiện tại
   * Lưu vào biến @CurrentDate
2. Kiểm tra thời hạn mã giảm giá:
   * So sánh với ngày hiện tại (@CurrentDate)
   * Lọc các mã giảm giá còn hiệu lực
3. Thống kê số lượng khóa học:
   * Đếm số khóa học đã áp dụng mỗi mã giảm giá
4. Sắp xếp kết quả:
   * Theo số lượng khóa học giảm dần
   * Theo ngày hết hạn tăng dần

**Input:** Không có tham số đầu vào

**Output:** Kết quả trả về bảng gồm các cột:

* CouponID: Mã định danh
* CouponTitle: Tiêu đề mã giảm giá
* CouponValue: Giá trị giảm giá
* CouponType: Loại giảm giá
* CouponStartDate: Ngày bắt đầu
* CouponExpire: Ngày hết hạn
* CouponMaxDiscount: Giảm giá tối đa
* TotalAffectedCourses: Số khóa học đã áp dụng

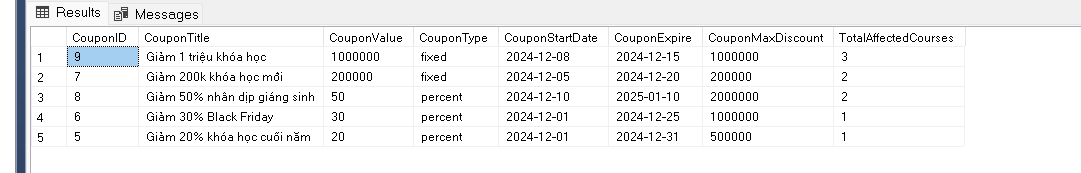
Câu lệnh tạo thủ tục:

|  |
| --- |
| GO  CREATE ALTER PROCEDURE sp\_GetValidCoupons  AS  BEGIN  DECLARE @CurrentDate DATE = GETDATE();    SELECT  c.CouponID,  c.CouponTitle,  c.CouponValue,  c.CouponType,  c.CouponStartDate,  c.CouponExpire,  c.CouponMaxDiscount,  (  SELECT COUNT(DISTINCT co.CourseID)  FROM Orders o, CourseOrder co  WHERE o.CouponID = c.CouponID  AND co.OrderID = o.OrderID  ) as TotalAffectedCourses  FROM Coupon c  WHERE c.CouponStartDate <= @CurrentDate  AND c.CouponExpire >= @CurrentDate  ORDER BY TotalAffectedCourses DESC, c.CouponExpire ASC;  END; |

Nhóm câu lệnh thực thi thủ tục mẫu:

|  |
| --- |
| EXEC sp\_GetValidCoupons; |

Kết quả các màn hình hiển thị từ DBMS minh họa cho việc thực hiện thủ tục:



**2.4 HÀM**

**a. Mô tả hàm: CalculateTotalRevenue**

**Chức năng của hàm**

Hàm CalculateTotalRevenue được sử dụng để tính tổng doanh thu từ các đơn hàng (orders) của một khóa học cụ thể, dựa trên thông tin trong bảng CourseOrder và Course.

**Use Case**

Hàm này được sử dụng trong các trường hợp:

* Khi cần tính tổng doanh thu của một khóa học cụ thể để phục vụ báo cáo doanh thu hoặc phân tích dữ liệu.
* Đảm bảo rằng doanh thu được tính toán chính xác dựa trên giá của khóa học và số lượng đơn hàng liên quan.

**Các thao tác chính**

1. **Kiểm tra tính hợp lệ**:
   * Kiểm tra xem CourseID có tồn tại trong bảng Course hay không.
   * Nếu không tồn tại, hàm trả về -1 để báo lỗi.
2. **Duyệt qua các đơn hàng**:
   * Sử dụng con trỏ (cursor) để duyệt qua từng đơn hàng liên quan đến khóa học trong bảng CourseOrder.
3. **Tính tổng doanh thu**:
   * Lấy giá của khóa học từ bảng Course và cộng dồn vào tổng doanh thu.
4. **Hoàn tất và trả về kết quả**:
   * Đóng con trỏ sau khi duyệt qua toàn bộ các đơn hàng.
   * Trả về tổng doanh thu tính được.

**Input**

Hàm nhận đầu vào là:

* @CourseID (INT): Mã định danh của khóa học cần tính tổng doanh thu.

**Output**

* **Kết quả trả về**:
  + Tổng doanh thu (INT) của các đơn hàng liên quan đến khóa học có CourseID.
  + Trả về -1 nếu CourseID không tồn tại trong bảng Course.

**Các bước thực hiện**

1. **Kiểm tra tính hợp lệ của CourseID**:
   * Nếu CourseID không tồn tại trong bảng Course, hàm sẽ trả về -1 để báo lỗi.
2. **Sử dụng con trỏ để duyệt qua đơn hàng**:
   * Lấy danh sách các OrderID liên quan đến khóa học từ bảng CourseOrder.
   * Duyệt qua từng đơn hàng bằng con trỏ.
3. **Tính tổng doanh thu**:
   * Lấy giá của khóa học từ bảng Course.
   * Cộng dồn giá trị vào biến @TotalRevenue.
4. **Hoàn tất và trả về**:
   * Đóng con trỏ và thu hồi bộ nhớ.
   * Trả về giá trị của @TotalRevenue.

**Ví dụ sử dụng**

1. **Kịch bản hợp lệ**:
   * @CourseID = 1: Khóa học tồn tại với nhiều đơn hàng.
   * Kết quả trả về là tổng doanh thu từ tất cả các đơn hàng liên quan.
2. **Kịch bản không hợp lệ**:
   * @CourseID = 999: Khóa học không tồn tại.
   * Kết quả trả về -1.

*-- Hàm tính tổng doanh thu từ các đơn hàng của một khóa học*

GO

CREATE FUNCTION CalculateTotalRevenue

(

    @CourseID INT

)

RETURNS INT

AS

BEGIN

    DECLARE @TotalRevenue INT = 0;

    DECLARE @OrderID INT;

    DECLARE @CoursePrice INT;

*-- Kiểm tra xem CourseID có tồn tại trong bảng*

    IF NOT EXISTS (SELECT 1 FROM Course WHERE CourseID = @CourseID)

    BEGIN

        RETURN -1; *-- Trả về -1 nếu CourseID không hợp lệ*

    END

*-- Khai báo con trỏ để lặp qua các đơn hàng liên quan đến khóa học*

    DECLARE OrderCursor CURSOR FOR

    SELECT OrderID FROM CourseOrder WHERE CourseID = @CourseID;

*-- Mở con trỏ để lấy OrderID đầu tiên*

    OPEN OrderCursor;

    FETCH NEXT FROM OrderCursor INTO @OrderID; *-- Lấy OrderID đầu tiên*

*-- Lặp qua các đơn hàng để tính tổng doanh thu*

    WHILE @@FETCH\_STATUS = 0 *-- Nếu còn dữ liệu*

    BEGIN

*-- Lấy giá khoá học từ bảng Course*

        SELECT @CoursePrice = CoursePrice FROM Course WHERE CourseID = @CourseID;

*-- Cộng vào tổng doanh thu*

        SET @TotalRevenue = @TotalRevenue + @CoursePrice;

*-- Lấy OrderID tiếp theo*

        FETCH NEXT FROM OrderCursor INTO @OrderID;

    END

*-- Đóng con trỏ và thu hồi bộ nhớ*

    CLOSE OrderCursor;

    DEALLOCATE OrderCursor;

*-- Trả về tổng doanh thu*

    RETURN @TotalRevenue;

END;

GO

Nhóm câu lệnh thực thi hàm:

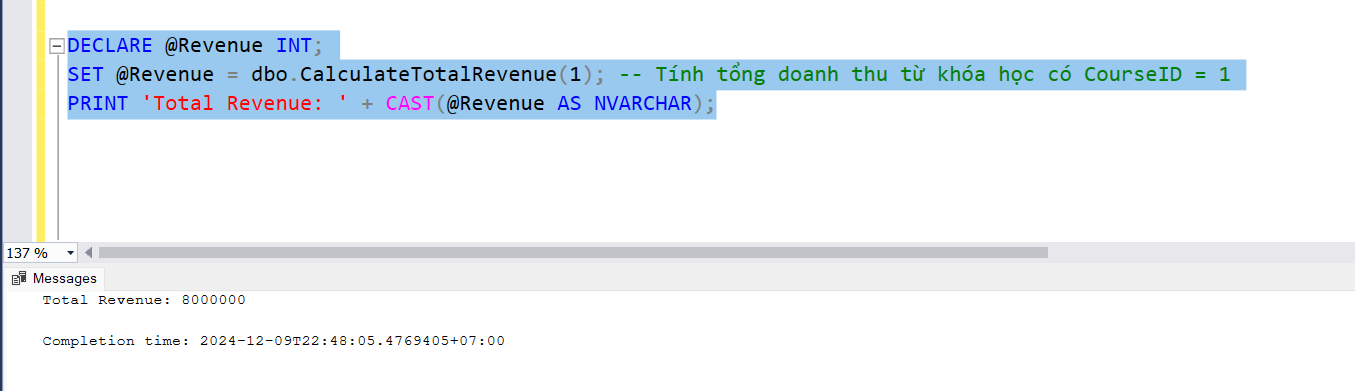
DECLARE @Revenue INT;

SET @Revenue = dbo.CalculateTotalRevenue(1); *-- Tính tổng doanh thu từ khóa học có CourseID = 1*

PRINT 'Total Revenue: ' + CAST(@Revenue AS NVARCHAR);

Kết quả các màn hình hiển thị từ DBMS minh họa cho việc thực hiện hàm

Khi course ID tồn tại:



Khi course ID không tồn tại: Trả về -1A computer screen shot of a message

Description automatically generated

**Mô tả hàm: CalculateStudentTestScore**

**Chức năng của hàm** Hàm CalculateStudentTestScore được sử dụng để tính điểm trung bình các bài kiểm tra của một học sinh, bằng cách:

* Tính tổng điểm từ các bài kiểm tra mà học sinh đã làm
* Xác định số điểm cho mỗi câu hỏi dựa trên tổng số câu trong bài kiểm tra
* Kiểm tra câu trả lời của học sinh với đáp án đúng để tính điểm

**Use Case** Hàm này được sử dụng trong các trường hợp:

* Khi cần tính toán điểm số của học sinh sau khi hoàn thành bài kiểm tra
* Khi cần đánh giá kết quả học tập của học sinh
* Khi cần xuất báo cáo điểm số cho học sinh hoặc phụ huynh
* Hỗ trợ việc theo dõi tiến độ học tập của học sinh

**Các thao tác chính**

1. **Kiểm tra tính hợp lệ**:
   * Xác nhận StudentID tồn tại trong bảng Student
   * Nếu không tồn tại, trả về -1 để báo lỗi
2. **Duyệt qua các bài kiểm tra**:
   * Sử dụng cursor để duyệt qua từng bài kiểm tra của học sinh
   * Lấy thông tin về số câu hỏi và điểm số mỗi câu
3. **Tính điểm cho mỗi bài kiểm tra**:
   * Đếm số câu trả lời đúng
   * Tính điểm dựa trên tỷ lệ câu đúng và tổng số câu
   * Cập nhật tổng điểm và số lượng bài kiểm tra
4. **Tính điểm trung bình**:
   * Tính điểm trung bình từ tổng điểm và số lượng bài kiểm tra
   * Trả về kết quả cuối cùng

**Input**

* @StudentID (INT): Mã định danh của học sinh cần tính điểm trung bình

**Output**

* Điểm trung bình (FLOAT) của tất cả các bài kiểm tra
* Trả về -1 nếu StudentID không tồn tại
* Trả về 0 nếu học sinh chưa làm bài kiểm tra nào

**Ví dụ sử dụng**

1. Kịch bản thành công:
   * @StudentID = 7 (học sinh tồn tại và có làm bài kiểm tra)
   * Kết quả: Trả về điểm trung bình các bài kiểm tra
2. Kịch bản thất bại:
   * @StudentID = 999 (học sinh không tồn tại)
   * Kết quả: Trả về -1
3. Kịch bản không có dữ liệu:
   * @StudentID = 20 (học sinh tồn tại nhưng chưa làm bài kiểm tra)
   * Kết quả: Trả về 0

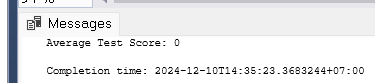
|  |
| --- |
| CREATE OR ALTER FUNCTION CalculateStudentTestScore  (  @StudentID INT  )  RETURNS FLOAT  AS  BEGIN  -- Variables for calculations  DECLARE @TotalScore FLOAT = 0;  DECLARE @TestCount INT = 0;  DECLARE @CurrentTestOrder INT;  DECLARE @CurrentCourseID INT;  DECLARE @CurrentChapterName NVARCHAR(255);  DECLARE @QuestionCount INT;  DECLARE @PointsPerQuestion FLOAT;  DECLARE @CorrectAnswers INT;  DECLARE @TestScore FLOAT;  -- Validate input  IF NOT EXISTS (SELECT 1 FROM Student WHERE StudentID = @StudentID)  RETURN -1; -- Invalid StudentID  -- Cursor to iterate through all tests taken by the student  DECLARE TestCursor CURSOR FOR  SELECT DISTINCT  st.TestOrder,  st.CourseID,  st.ChapterName  FROM Student\_Test st  WHERE st.StudentID = @StudentID;  OPEN TestCursor;  FETCH NEXT FROM TestCursor INTO @CurrentTestOrder, @CurrentCourseID, @CurrentChapterName;  WHILE @@FETCH\_STATUS = 0  BEGIN  -- Get total number of questions in this test  SELECT @QuestionCount = COUNT(\*)  FROM QuestionTest qt  WHERE qt.TestOrder = @CurrentTestOrder  AND qt.CourseID = @CurrentCourseID  AND qt.ChapterName = @CurrentChapterName;  -- Calculate points per question (10 points total per test)  SET @PointsPerQuestion = 10.0 / @QuestionCount;  -- Count correct answers for this test  SELECT @CorrectAnswers = COUNT(\*)  FROM StudentOption so  JOIN Options o ON so.QuestionID = o.QuestionID AND so.OptionName = o.OptionName  JOIN QuestionTest qt ON qt.QuestionID = o.QuestionID  WHERE so.StudentID = @StudentID  AND qt.TestOrder = @CurrentTestOrder  AND qt.CourseID = @CurrentCourseID  AND qt.ChapterName = @CurrentChapterName  AND o.IsCorrect = 1;  -- Calculate score for this test  SET @TestScore = @CorrectAnswers \* @PointsPerQuestion;  -- Add to total  SET @TotalScore = @TotalScore + @TestScore;  SET @TestCount = @TestCount + 1;  FETCH NEXT FROM TestCursor INTO @CurrentTestOrder, @CurrentCourseID, @CurrentChapterName;  END;  CLOSE TestCursor;  DEALLOCATE TestCursor;  -- Return average score  RETURN CASE  WHEN @TestCount > 0 THEN @TotalScore / @TestCount  ELSE 0  END;  END; |

Nhóm câu lệnh thực thi hàm:

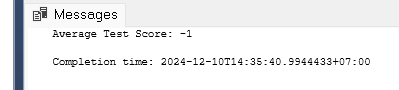
|  |
| --- |
| DECLARE @AvgScore FLOAT;  SET @AvgScore = dbo.CalculateStudentTestScore(8); -- Tính điểm trung bình của học sinh có StudentID = 8  PRINT 'Average Test Score: ' + CAST(@AvgScore AS NVARCHAR); |

Kết quả:



Nếu học sinh chưa làm bài kiểm tra nào:  


Nếu học sinh không tồn tại:



**3. ỨNG DỤNG KẾT NỐI VỚI CSDL**

Giao diện ứng dụng: Gồm các nút bấm tương ứng với các bảng trong cơ sở dữ liệu. Khi bấm vào thì bảng dữ liệu đó sẽ hiện ra:

http://localhost:3500

A screenshot of a computer

Description automatically generated

http://localhost:3500/coupon

Giao diện khi bấm vào bảng “Coupon”: Bảng coupon hiện ra với các dữ liệu mẫu tương ứng.

Trang này gồm:

* Nút “Go Back” để trở về trang <http://localhost:3500>
* Nút “Create New Coupon” để tạo mới một mã giảm giá (Thủ tục insert trong phần 2.1)
* Thanh search để lọc ra các coupon có tiêu đề tương ứng.
* Nút sort để sắp xếp các dữ liệu trong bảng coupon theo các tiêu chí như CouponID, CouponTitle, CouponValue…

A screenshot of a coupon id

Description automatically generated

* Mỗi hàng tương ứng với coupon sẽ có thêm các nút Edit để chỉnh sửa coupon và Delete để xoá coupon tương ứng (Thủ tục sửa/xoá trong phần 2.1)
* Nút “View Coupons can be appied now” gọi thủ tục sp\_GetValidCoupons ở phần 2.3 để tìm ra các mã giảm giá có thể áp dụng ở thời điểm hiện tại.

**A screenshot of a computer

Description automatically generated**

http://localhost:3500/coupon/edit/1

Giao diện khi nhấn vào “Edit” một coupon nào đó: Khi nhập dữ liệu và nhấn “Update”, trang web sẽ dùng thủ tục ở phần 2.1 để validate và báo lỗi hoặc cập nhật (nếu dữ liệu hợp lệ).

**A screenshot of a computer

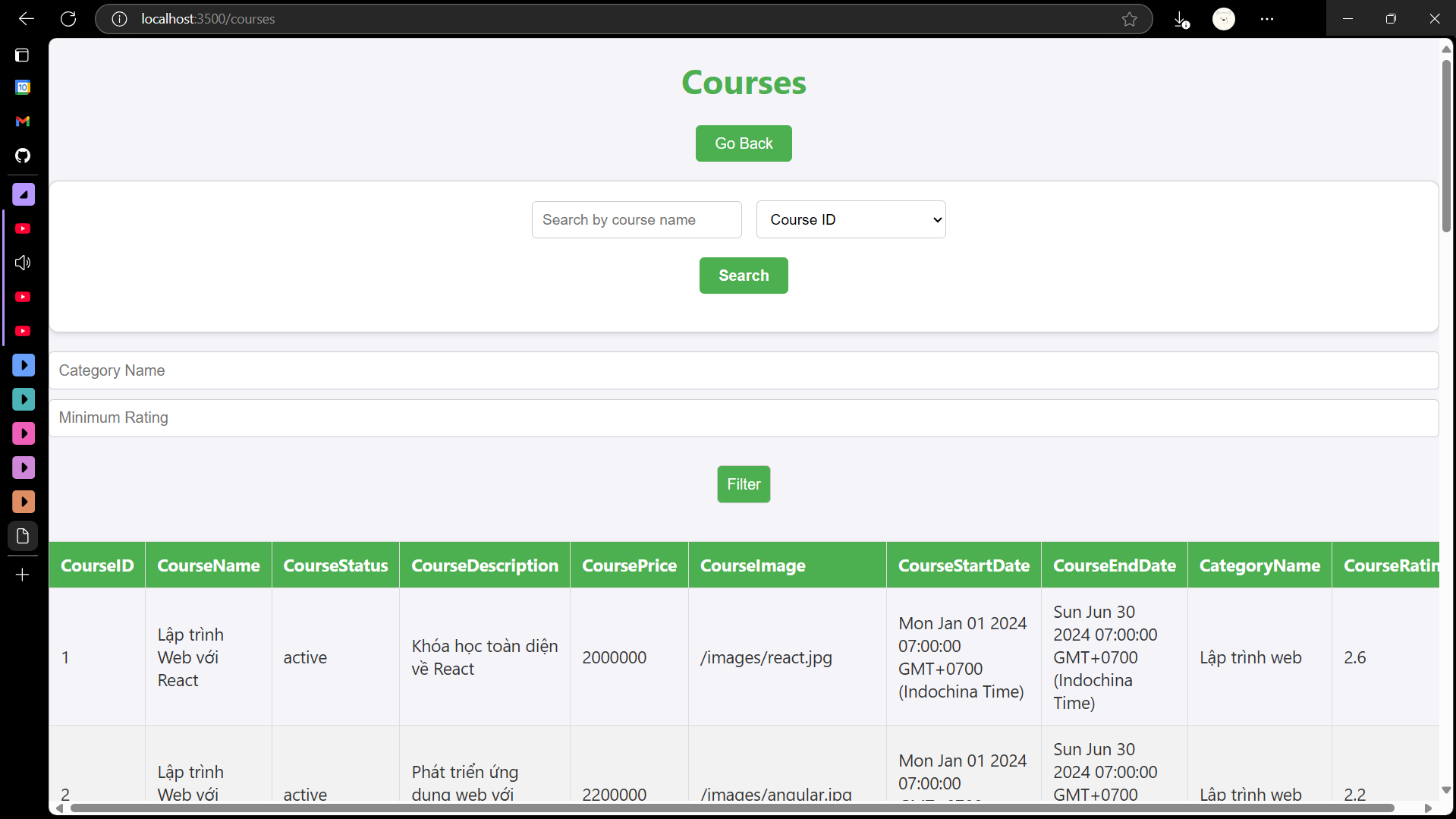
Description automatically generated**

Khi nhập dữ liệu không hợp lệ:A screenshot of a coupon

Description automatically generated

**http://localhost:3500/courses**

Giao diện khi nhấn vào bảng Course, bao gồm các nút tìm kiếm theo tên khoá học, sắp xếp, lọc (filter) theo tên danh mục và đánh giá trung bình (average rating) (các thủ tục ở phần 2.3)

****

**4. Thiết kế ERD cập nhật mới nhất**

A diagram of a football game

Description automatically generated

A close-up of a white sheet

Description automatically generated

**Lưu ý chung:**

- Xem kỹ lại các yêu cầu trong file mô tả BTL2.

- Các câu lệnh đưa vào báo cáo phải ở dạng TEXT, không dùng ảnh chụp màn hình.

- Chỉ những nội dung được đưa vào cuốn báo cáo mới được tính điểm, những phần sửa chữa, bổ sung tại thời điểm báo cáo sẽ không tính.

- Tại buổi báo cáo, các nhóm cần thực hiện tạo sẵn các bảng dữ liệu, thêm sẵn dữ liệu, và tạo sẵn trigger, thủ tục, hàm. KHÔNG phải đến lúc báo cáo mới chạy các câu lệnh trên.

- Các nhóm cần chuẩn bị dữ liệu mẫu, câu lệnh mẫu để demo các nội dung trong buổi báo cáo.

1.1

*-------------------------------- CREATE DATABASE --------------------------------*

CREATE TABLE Users(

    UsernameID INT IDENTITY (1,1) PRIMARY KEY,

    UserEmail NVARCHAR(255),

    UserPhone VARCHAR(10),

    UserPassword NVARCHAR(255),

    UserAddress NVARCHAR(255),

    UserImage NVARCHAR(255),

    Username NVARCHAR(255),

)

*--Bảng Admin*

CREATE TABLE Admin(

    AdminID INT PRIMARY KEY,

    FOREIGN KEY (AdminID) REFERENCES Users(UsernameID),

);

*--Bảng giảng viên*

CREATE TABLE Teacher(

    TeacherID INT PRIMARY KEY,

    TeacherDescription NVARCHAR(255),

    FOREIGN KEY (TeacherID) REFERENCES Users(UsernameID),

)

*--Bảng danh mục*

CREATE TABLE Category(

    CategoryID INT PRIMARY KEY,

    CategoryName NVARCHAR(255),

    CategoryDescription NVARCHAR(255),

);

*--Bảng khoá học*

CREATE TABLE Course(

    CourseID INT PRIMARY KEY,

    CourseName NVARCHAR(255),

    CourseStatus NVARCHAR(255),

    CourseDescription NVARCHAR(255),

    CoursePrice INT,

    CourseImage NVARCHAR(255),

    CourseStartDate DATE,

    CourseEndDate DATE,

    CourseAverageRating DECIMAL(10,3),

    CategoryID INT,

    FOREIGN KEY (CategoryID) REFERENCES Category(CategoryID),

);

*--Chương*

CREATE TABLE Chapter(

    ChapterName NVARCHAR(255),

    CourseID INT,

    PRIMARY KEY (ChapterName, CourseID),

    FOREIGN KEY (CourseID) REFERENCES Course(CourseID)

);

*--Tài liệu*

CREATE TABLE Document(

    DocumentID INT IDENTITY (1,1) PRIMARY KEY,

    CourseID INT,

    ChapterName NVARCHAR(255),

    DocumentAuthor NVARCHAR(255),

    DocumentTitle NVARCHAR(255),

    DocumentSize INT,

    DocumentType NVARCHAR(255),

    DocumentContent NVARCHAR(255),

    FOREIGN KEY (ChapterName, CourseID) REFERENCES Chapter(ChapterName, CourseID)

);

*--Bài học*

CREATE TABLE Lesson(

    LessonOrder INT,

    CourseID INT,

    ChapterName NVARCHAR(255),

    LessonTitle NVARCHAR(255),

    LessonContent NVARCHAR(255),

    LessonDuraion TIME,

    PRIMARY KEY (LessonOrder, CourseID, ChapterName),

    FOREIGN KEY (ChapterName, CourseID) REFERENCES Chapter(ChapterName, CourseID),

)

*--Bài tập*

CREATE TABLE Exercise(

    ExerciseOrder INT,

    LessonOrder INT,

    CourseID INT,

    ChapterName NVARCHAR(255),

    ExerciseTitle NVARCHAR(255),

    ExerciseAnswer NVARCHAR(255),

    ExerciseCorrectAnswerNumber INT,

    ExerciseContent NVARCHAR(255),

    PRIMARY KEY (ExerciseOrder, LessonOrder, CourseID, ChapterName),

    FOREIGN KEY (LessonOrder, CourseID, ChapterName) REFERENCES Lesson(LessonOrder, CourseID, ChapterName),

)

*--Bài kiểm tra*

CREATE TABLE Test(

    TestOrder INT,

    CourseID INT,

    ChapterName NVARCHAR(255),

    TestDuration TIME,

    PRIMARY KEY (TestOrder, CourseID, ChapterName),

    FOREIGN KEY (ChapterName, CourseID) REFERENCES Chapter(ChapterName, CourseID),

)

*--Câu hỏi*

CREATE TABLE Question(

    QuestionID INT IDENTITY (1,1) PRIMARY KEY,

    QuestionScore INT,

    QuestionContent NVARCHAR(255),

)

*--Lựa chọn*

CREATE TABLE Options(

    QuestionID INT,

    OptionName NVARCHAR(255) UNIQUE,

    IsCorrect BIT, *--Đúng hay sai*

    OptionContent NVARCHAR(255), *--Nội dung lựa chọn*

    PRIMARY KEY (QuestionID, OptionName),

    FOREIGN KEY (QuestionID) REFERENCES Question(QuestionID),

)

*--Câu hỏi thuộc bài kiểm tra*

CREATE TABLE QuestionTest(

    QuestionID INT,

    TestOrder INT,

    CourseID INT,

    ChapterName NVARCHAR(255),

    PRIMARY KEY (QuestionID, TestOrder, CourseID, ChapterName),

    FOREIGN KEY (QuestionID) REFERENCES Question(QuestionID),

    FOREIGN KEY (TestOrder, CourseID, ChapterName) REFERENCES Test(TestOrder, CourseID, ChapterName),

)

*--Bảng học viên*

CREATE TABLE Student(

    StudentID INT IDENTITY (1,1) PRIMARY KEY,

    OptionName NVARCHAR(255),

    QuestionID INT,

    FOREIGN KEY (StudentID) REFERENCES Users(UsernameID),

    FOREIGN KEY (OptionName) REFERENCES Options(OptionName),

    FOREIGN KEY (QuestionID) REFERENCES Question(QuestionID),

)

CREATE TABLE Student\_Test(

    StudentID INT,

    CourseID INT,

    TestOrder INT,

    ChapterName NVARCHAR(255),

    TestScore FLOAT,  *-- Điểm số của sinh viên cho bài kiểm tra*

*-- Khóa chính gồm 4 trường*

    PRIMARY KEY (StudentID, CourseID, TestOrder, ChapterName),

*-- Khóa ngoại tới bảng Student*

    FOREIGN KEY (StudentID) REFERENCES Student(StudentID),

*-- Khóa ngoại tới bảng Test (gồm 3 trường CourseID, TestOrder, ChapterName)*

    FOREIGN KEY (TestOrder, CourseID, ChapterName)

        REFERENCES Test(TestOrder, CourseID, ChapterName)

);

*--Người đại diện*

CREATE TABLE Guardian(

    GuardianID INT IDENTITY (1,1) PRIMARY KEY,

    StudentID INT,

    GuardianName NVARCHAR(255),

    GuardianEmail NVARCHAR(255),

    GuardianPhone VARCHAR(10),

    FOREIGN KEY (StudentID) REFERENCES Student(StudentID),

)

*--Giảng viên dạy khoá học*

CREATE TABLE TeacherCourse(

    TeacherID INT,

    CourseID INT,

    PRIMARY KEY (TeacherID, CourseID),

    FOREIGN KEY (TeacherID) REFERENCES Teacher(TeacherID),

    FOREIGN KEY (CourseID) REFERENCES Course(CourseID),

)

*--Phiếu giảm giá*

CREATE TABLE Coupon(

    CouponID INT IDENTITY (1,1) PRIMARY KEY,

    CouponTitle NVARCHAR(255),

    CouponValue INT,

    CouponType NVARCHAR(255),

    CouponStartDate DATE,

    CouponExpire DATE,

    CouponMaxDiscount INT,

)

*--Bảng Chỉnh sửa*

CREATE TABLE Edit (

    EditID INT IDENTITY (1,1) PRIMARY KEY,

    EditTime TIME,

    EditDescription NVARCHAR(255),

    EditAdminID INT,

    EditCouponID INT,

    EditCourseID INT,

    FOREIGN KEY (EditCourseID) REFERENCES Course(CourseID),

    FOREIGN KEY (EditAdminID) REFERENCES Admin(AdminID),

    FOREIGN KEY (EditCouponID) REFERENCES Coupon(CouponID),

);

*--Khoá học có học viên*

CREATE TABLE CourseStudent(

    CourseID INT,

    StudentID INT,

    PRIMARY KEY (CourseID, StudentID),

    FOREIGN KEY (CourseID) REFERENCES Course(CourseID),

    FOREIGN KEY (StudentID) REFERENCES Student(StudentID),

)

*--Phương thức thanh toán*

CREATE TABLE PaymentMethod(

    PaymentCode NVARCHAR(255) PRIMARY KEY, *--mã thanh toán*

    PayerName NVARCHAR(255),

    PaymentDate DATE,

)

*--Momo*

CREATE TABLE Momo(

    PaymentCode NVARCHAR(255) PRIMARY KEY, *--mã thanh toán*

    PhoneNumber VARCHAR(10),

    FOREIGN KEY (PaymentCode) REFERENCES PaymentMethod(PaymentCode),

)

*--Internet Banking*

CREATE TABLE InternetBanking(

    PaymentCode NVARCHAR(255) PRIMARY KEY, *--mã thanh toán*

    BankName NVARCHAR(255),

    FOREIGN KEY (PaymentCode) REFERENCES PaymentMethod(PaymentCode),

)

*--VISA*

CREATE TABLE VISA(

    PaymentCode NVARCHAR(255) PRIMARY KEY, *--mã thanh toán*

    CardNumber VARCHAR(16),

    ExpireDate DATE,

    FOREIGN KEY (PaymentCode) REFERENCES PaymentMethod(PaymentCode),

)

*--Đơn hàng*

CREATE TABLE Orders(

    OrderID INT PRIMARY KEY,

    OrderPaymentStatus NVARCHAR(255),

    OrderDate DATE,

    OrderPaymentCode NVARCHAR(255),

    StudentID INT,

    TotalAmount DECIMAL(10,2),

    CouponID INT,

    FOREIGN KEY (StudentID) REFERENCES Student(StudentID),

    FOREIGN KEY (OrderPaymentCode) REFERENCES PaymentMethod(PaymentCode),

    FOREIGN KEY (CouponID) REFERENCES Coupon(CouponID),

)

*--Khoá học thêm vào đơn hàng*

CREATE TABLE CourseOrder(

    CourseID INT,

    OrderID INT,

    PRIMARY KEY (CourseID, OrderID),

    FOREIGN KEY (CourseID) REFERENCES Course(CourseID),

    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),

)

*--bảng review*

CREATE TABLE Review(

    ReviewID INT,

    CourseID INT,

    ReviewScore INT,

    ReviewContent NVARCHAR(255),

    StudentID INT,

    PRIMARY KEY(ReviewID, CourseID),  *-- Review là thực thể yếu của Course nên khóa chính gồm ReviewID và CourseID*

    FOREIGN KEY (CourseID) REFERENCES Course(CourseID),

    FOREIGN KEY (StudentID) REFERENCES Student(StudentID)

);

*---TRIGGER 1*

GO

CREATE TRIGGER TR\_Orders\_Calculate\_Total

ON Orders

AFTER UPDATE

AS

BEGIN

*-- Declare variables*

    DECLARE @OrderID INT

    DECLARE @PaymentStatus NVARCHAR(255)

*-- Get OrderID and payment status from inserted record*

    SELECT @OrderID = OrderID, @PaymentStatus = OrderPaymentStatus

    FROM inserted

*-- Check if order is already paid*

    IF EXISTS (

        SELECT 1

        FROM Orders

        WHERE OrderID = @OrderID

        AND OrderPaymentStatus = 'paid'

    )

    BEGIN

        RAISERROR ('Không thể thay đổi đơn hàng đã thanh toán.', 16, 1)

        ROLLBACK TRANSACTION

        RETURN

    END

*-- Declare variables*

    DECLARE @CouponID INT

    DECLARE @SubTotal FLOAT

    DECLARE @Discount FLOAT

    DECLARE @FinalTotal FLOAT

    DECLARE @CouponType NVARCHAR(255)

    DECLARE @CouponValue INT

    DECLARE @CouponMaxDiscount INT

    DECLARE @CurrentDate DATE

*-- Get current date*

    SET @CurrentDate = GETDATE()

*-- Get the affected OrderID from inserted records*

    SELECT @OrderID = OrderID, @CouponID = CouponID

    FROM inserted

*-- Calculate subtotal by summing up course prices*

    SELECT @SubTotal = COALESCE(SUM(c.CoursePrice), 0)

    FROM CourseOrder co

    JOIN Course c ON co.CourseID = c.CourseID

    WHERE co.OrderID = @OrderID

*-- If there's a coupon, calculate discount*

    IF @CouponID IS NOT NULL

    BEGIN

*-- Get coupon details with date validation*

        SELECT

@CouponType = CouponType,

            @CouponValue = CouponValue,

            @CouponMaxDiscount = CouponMaxDiscount

        FROM Coupon

        WHERE CouponID = @CouponID

        AND CouponStartDate <= @CurrentDate

        AND CouponExpire >= @CurrentDate

*-- Only calculate discount if coupon is valid (not NULL from previous query)*

        IF @CouponType IS NOT NULL

        BEGIN

*-- Calculate discount based on coupon type*

            IF @CouponType = 'percent'

            BEGIN

                SET @Discount = (@SubTotal \* @CouponValue) / 100.0

*-- Check if discount exceeds max discount*

                IF @Discount > @CouponMaxDiscount

                    SET @Discount = @CouponMaxDiscount

            END

            ELSE *-- Fixed value*

            BEGIN

                SET @Discount = @CouponValue

*-- Check if discount exceeds max discount*

                IF @Discount > @CouponMaxDiscount

                    SET @Discount = @CouponMaxDiscount

*-- Check if discount exceeds subtotal*

                IF @Discount > @SubTotal

                    SET @Discount = @SubTotal

            END

        END

        ELSE

        BEGIN

            SET @Discount = 0 *-- Coupon is not valid due to dates*

        END

    END

    ELSE

    BEGIN

        SET @Discount = 0

    END

*-- Calculate final total*

    SET @FinalTotal = @SubTotal - @Discount

*-- Update the TotalAmount in Orders table*

    UPDATE Orders

    SET TotalAmount = @FinalTotal

    WHERE OrderID = @OrderID

END;

GO

GO

CREATE TRIGGER TR\_CourseOrder\_Calculate\_Total

ON CourseOrder

AFTER INSERT, UPDATE, DELETE

AS

BEGIN

    IF EXISTS (

        SELECT 1

FROM Orders o

        WHERE o.OrderPaymentStatus = 'paid'

        AND o.OrderID IN (

            SELECT OrderID FROM inserted

            UNION

            SELECT OrderID FROM deleted

        )

    )

    BEGIN

        RAISERROR ('Không thể thay đổi đơn hàng đã thanh toán.', 16, 1)

        ROLLBACK TRANSACTION

        RETURN

    END

    DECLARE @CurrentDate DATE

    SET @CurrentDate = GETDATE()

    UPDATE Orders

    SET TotalAmount = (

        SELECT COALESCE(SUM(c.CoursePrice), 0) -

            CASE

                WHEN cp.CouponType = N'percent'

                AND cp.CouponStartDate <= @CurrentDate

                AND cp.CouponExpire >= @CurrentDate THEN

                    CASE

                        WHEN (COALESCE(SUM(c.CoursePrice), 0) \* cp.CouponValue / 100.0) > cp.CouponMaxDiscount

                        THEN cp.CouponMaxDiscount

                        ELSE (COALESCE(SUM(c.CoursePrice), 0) \* cp.CouponValue / 100.0)

                    END

                WHEN cp.CouponType = N'value'

                AND cp.CouponStartDate <= @CurrentDate

                AND cp.CouponExpire >= @CurrentDate THEN

                    CASE

                        WHEN cp.CouponValue > cp.CouponMaxDiscount THEN cp.CouponMaxDiscount

                        WHEN cp.CouponValue > COALESCE(SUM(c.CoursePrice), 0) THEN COALESCE(SUM(c.CoursePrice), 0)

                        ELSE cp.CouponValue

                    END

                ELSE 0

            END

        FROM CourseOrder co2

        JOIN Course c ON co2.CourseID = c.CourseID

        LEFT JOIN Orders o ON co2.OrderID = o.OrderID

        LEFT JOIN Coupon cp ON o.CouponID = cp.CouponID

        WHERE co2.OrderID = Orders.OrderID

        GROUP BY co2.OrderID, cp.CouponType, cp.CouponValue, cp.CouponMaxDiscount,

                cp.CouponStartDate, cp.CouponExpire

    )

    WHERE OrderID IN (

        SELECT OrderID FROM inserted

        UNION

        SELECT OrderID FROM deleted

    );

END;

GO

*-- TRIGGER 2*

GO

CREATE OR ALTER TRIGGER TR\_Review\_Update\_Course\_Rating

ON Review

AFTER INSERT, UPDATE, DELETE

AS

BEGIN

*-- Update Course average rating for all affected courses*

    UPDATE Course

    SET CourseAverageRating = (

        SELECT AVG(CAST(ReviewScore AS FLOAT))

        FROM Review r

        WHERE r.CourseID = Course.CourseID

        GROUP BY r.CourseID

    )

    WHERE CourseID IN (

*-- Get affected CourseIDs from both inserted and deleted*

        SELECT CourseID FROM inserted

        UNION

        SELECT CourseID FROM deleted

    );

END;

GO

1.2

*--------------------------- INSERT ---------------------------*

*-- Insert Users*

INSERT INTO Users (UserEmail, UserPhone, UserPassword, UserAddress, UserImage, Username) VALUES

*-- Admins*

('admin1@edu.com', '0123456789', 'admin123', N'Hà Nội', '/images/admin1.jpg', N'Admin Nguyễn Văn A'),

('admin2@edu.com', '0123456788', 'admin456', N'Hồ Chí Minh', '/images/admin2.jpg', N'Admin Trần Thị B'),

*-- Teachers*

('teacher1@edu.com', '0123456787', 'teacher123', N'Hà Nội', '/images/teacher1.jpg', N'GV. Phạm Văn C'),

('teacher2@edu.com', '0123456786', 'teacher456', N'Đà Nẵng', '/images/teacher2.jpg', N'GV. Lê Thị D'),

('teacher3@edu.com', '0123456785', 'teacher789', N'Hải Phòng', '/images/teacher3.jpg', N'GV. Hoàng Văn E'),

('teacher4@edu.com', '0123456784', 'teacher012', N'Cần Thơ', '/images/teacher4.jpg', N'GV. Ngô Thị F'),

*-- Students (14 students)*

('student1@gmail.com', '0123456783', 'student123', N'Hà Nội', '/images/student1.jpg', N'Trần Văn G'),

('student2@gmail.com', '0123456782', 'student456', N'Hồ Chí Minh', '/images/student2.jpg', N'Nguyễn Thị H'),

('student3@gmail.com', '0123456781', 'student789', N'Đà Nẵng', '/images/student3.jpg', N'Lê Văn I'),

('student4@gmail.com', '0123456780', 'student012', N'Hải Phòng', '/images/student4.jpg', N'Phạm Thị K'),

('student5@gmail.com', '0123456779', 'student345', N'Cần Thơ', '/images/student5.jpg', N'Hoàng Văn L'),

('student6@gmail.com', '0123456778', 'student678', N'Hà Nội', '/images/student6.jpg', N'Ngô Thị M'),

('student7@gmail.com', '0123456777', 'student901', N'Hồ Chí Minh', '/images/student7.jpg', N'Vũ Văn N'),

('student8@gmail.com', '0123456776', 'student234', N'Đà Nẵng', '/images/student8.jpg', N'Đặng Thị O'),

('student9@gmail.com', '0123456775', 'student567', N'Hải Phòng', '/images/student9.jpg', N'Bùi Văn P'),

('student10@gmail.com', '0123456774', 'student890', N'Cần Thơ', '/images/student10.jpg', N'Mai Thị Q'),

('student11@gmail.com', '0123456773', 'student111', N'Hà Nội', '/images/student11.jpg', N'Lý Văn R'),

('student12@gmail.com', '0123456772', 'student222', N'Hồ Chí Minh', '/images/student12.jpg', N'Trịnh Thị S'),

('student13@gmail.com', '0123456771', 'student333', N'Đà Nẵng', '/images/student13.jpg', N'Đinh Văn T'),

('student14@gmail.com', '0123456770', 'student444', N'Hải Phòng', '/images/student14.jpg', N'Cao Thị U');

*-- Insert Admin*

INSERT INTO Admin (AdminID) VALUES (1), (2);

*-- Insert Teacher*

INSERT INTO Teacher (TeacherID, TeacherDescription) VALUES

(3, N'Chuyên gia lập trình với 10 năm kinh nghiệm'),

(4, N'Thạc sĩ Khoa học máy tính'),

(5, N'Tiến sĩ An toàn thông tin'),

(6, N'Chuyên gia về AI và Machine Learning');

*-- Insert Category*

INSERT INTO Category (CategoryID,CategoryName, CategoryDescription) VALUES

(1,N'Lập trình web', N'Các khóa học về phát triển web'),

(2,N'Lập trình mobile', N'Các khóa học về phát triển ứng dụng di động'),

(3,N'Cơ sở dữ liệu', N'Các khóa học về quản trị và thiết kế database'),

(4,N'AI và Machine Learning', N'Các khóa học về trí tuệ nhân tạo');

*-- Insert Course*

INSERT INTO Course (CourseID,CourseName, CourseStatus, CourseDescription, CoursePrice, CourseImage, CourseStartDate, CourseEndDate, CategoryID) VALUES

*-- Category 1: Lập trình web (Thêm 7 khóa)*

(1,N'Lập trình Web với React', 'active', N'Khóa học toàn diện về React', 2000000, '/images/react.jpg', '2024-01-01', '2024-06-30', 1),

(2,N'Lập trình Web với Angular', 'active', N'Phát triển ứng dụng web với Angular', 2200000, '/images/angular.jpg', '2024-01-01', '2024-06-30', 1),

(3,N'Node.js Backend Development', 'active', N'Xây dựng backend với Node.js', 2300000, '/images/nodejs.jpg', '2024-02-01', '2024-07-31', 1),

(4,N'PHP & Laravel Framework', 'active', N'Phát triển web với PHP và Laravel', 1900000, '/images/laravel.jpg', '2024-03-01', '2024-08-31', 1),

(5,N'Vue.js Frontend Development', 'active', N'Xây dựng giao diện với Vue.js', 2100000, '/images/vuejs.jpg', '2024-04-01', '2024-09-30', 1),

(6,N'Lập trình Web với HTML', 'active', N'Khóa học toàn diện về HTML', 2000000, '/images/react.jpg', '2024-01-01', '2024-06-30', 1),

(7,N'Lập trình Web với Bootstrap', 'active', N'Khóa học toàn diện về Bootstrap', 1500000, '/images/react.jpg', '2024-01-01', '2024-06-30', 1),

*-- Category 2: Lập trình mobile (Thêm 4 khóa)*

(8,N'iOS Development với Swift', 'active', N'Phát triển ứng dụng iOS', 2600000, '/images/swift.jpg', '2024-01-15', '2024-07-15', 2),

(9,N'React Native', 'active', N'Phát triển ứng dụng đa nền tảng với React Native', 2400000, '/images/react-native.jpg', '2024-02-15', '2024-08-15', 2),

(10,N'Flutter Development', 'active', N'Xây dựng ứng dụng mobile với Flutter', 2300000, '/images/flutter.jpg', '2024-03-15', '2024-09-15', 2),

(11,N'Kotlin for Android', 'active', N'Lập trình Android với Kotlin', 2200000, '/images/kotlin.jpg', '2024-04-15', '2024-10-15', 2),

*-- Category 3: Cơ sở dữ liệu (Thêm 4 khóa)*

(12,N'MongoDB Master', 'active', N'Làm chủ MongoDB', 1900000, '/images/mongodb.jpg', '2024-02-01', '2024-05-31', 3),

(13,N'PostgreSQL Advanced', 'active', N'PostgreSQL nâng cao', 2000000, '/images/postgresql.jpg', '2024-03-01', '2024-06-30', 3),

(14,N'MySQL Performance', 'active', N'Tối ưu hiệu suất MySQL', 1950000, '/images/mysql.jpg', '2024-04-01', '2024-07-31', 3),

(15,N'Oracle Database', 'active', N'Quản trị Oracle Database', 2100000, '/images/oracle.jpg', '2024-05-01', '2024-08-31', 3),

*-- Category 4: AI và Machine Learning (Thêm 4 khóa)*

(16,N'Deep Learning Cơ bản', 'active', N'Nhập môn Deep Learning', 3200000, '/images/deep-learning.jpg', '2024-02-15', '2024-08-15', 4),

(17,N'Computer Vision', 'active', N'Xử lý ảnh và thị giác máy tính', 3100000, '/images/cv.jpg', '2024-03-15', '2024-09-15', 4),

(18,N'Natural Language Processing', 'active', N'Xử lý ngôn ngữ tự nhiên', 3300000, '/images/nlp.jpg', '2024-04-15', '2024-10-15', 4),

(19,N'Data Science với Python', 'active', N'Khoa học dữ liệu với Python', 2900000, '/images/data-science.jpg', '2024-05-15', '2024-11-15', 4);

*-- Insert TeacherCourse*

INSERT INTO TeacherCourse (TeacherID, CourseID) VALUES

(3, 1), (3, 2),

(4, 2), (4, 3),

(5, 3), (5, 4),

(6, 4), (6, 1);

*-- Insert Student (assigning UsernameID 7-20 as students)*

SET IDENTITY\_INSERT Student ON;

INSERT INTO Student (StudentID, OptionName, QuestionID) VALUES

(7, NULL, NULL),

(8, NULL, NULL),

(9, NULL, NULL),

(10, NULL, NULL),

(11, NULL, NULL),

(12, NULL, NULL),

(13, NULL, NULL),

(14, NULL, NULL),

(15, NULL, NULL),

(16, NULL, NULL),

(17, NULL, NULL),

(18, NULL, NULL),

(19, NULL, NULL),

(20, NULL, NULL);

SET IDENTITY\_INSERT Student OFF;

*-- Insert PaymentMethod*

INSERT INTO PaymentMethod (PaymentCode, PayerName, PaymentDate) VALUES

('PAY001', N'Trần Văn G', '2024-01-05'),

('PAY002', N'Nguyễn Thị H', '2024-01-10'),

('PAY003', N'Lê Văn I', '2024-01-15'),

('PAY004', N'Phạm Thị K', '2024-01-20'),

('PAY005', N'Hoàng Văn L', '2024-01-25');

*-- Thêm dữ liệu cho Momo (2 giao dịch)*

INSERT INTO Momo (PaymentCode, PhoneNumber) VALUES

('PAY001', '0123456783'),  *-- Số điện thoại của student 7*

('PAY002', '0123456782');  *-- Số điện thoại của student 8*

*-- Thêm dữ liệu cho Internet Banking (2 giao dịch)*

INSERT INTO InternetBanking (PaymentCode, BankName) VALUES

('PAY003', N'VietcomBank'),  *-- Của student 9*

('PAY004', N'BIDV');         *-- Của student 10*

*-- Thêm dữ liệu cho VISA (1 giao dịch)*

INSERT INTO VISA (PaymentCode, CardNumber, ExpireDate) VALUES

('PAY005', '4111111111111111', '2025-12-31');  *-- Của student 11*

*-- Insert Coupon data*

INSERT INTO Coupon (CouponTitle, CouponValue, CouponType, CouponStartDate, CouponExpire, CouponMaxDiscount) VALUES

*-- Giảm theo phần trăm*

(N'Giảm 15% khóa học mới', 15, N'Phần trăm', '2024-01-01', '2024-03-31', 500000),*--1*

(N'Giảm 25% khóa học IT', 25, N'Phần trăm', '2024-02-01', '2024-04-30', 1000000),*--2*

*-- Giảm theo giá trị cố định*

(N'Giảm 300k khóa học đầu tiên', 300000, N'Giá trị', '2024-01-15', '2024-03-15', 300000),*--3*

(N'Giảm 500k khóa học nâng cao', 500000, N'Giá trị', '2024-02-15', '2024-05-15', 500000),*--4*

(N'Giảm 20% khóa học cuối năm', 20, N'percent', '2024-12-01', '2024-12-31', 500000),*--5*

(N'Giảm 30% Black Friday', 30, N'percent', '2024-12-01', '2024-12-25', 1000000),*--6*

(N'Giảm 200k khóa học mới', 200000, N'fixed', '2024-12-05', '2024-12-20', 200000),*--7*

(N'Giảm 50% nhân dịp giáng sinh', 50, N'percent', '2024-12-10', '2025-01-10', 2000000),*--8*

(N'Giảm 1 triệu khóa học', 1000000, N'fixed', '2024-12-08', '2024-12-15', 1000000);*--9*

*-- Insert Orders*

INSERT INTO Orders (OrderID,OrderPaymentStatus, OrderDate, OrderPaymentCode, StudentID,TotalAmount,CouponID) VALUES

(1,'paid', '2024-01-05', 'PAY001', 7,4500000,NULL), *--đơn 1 của student 7 (4,5 TRIEU)*

(2,'paid', '2024-01-10', 'PAY002', 8,3800000,NULL), *--đơn 2 của student 8 (3,8 trieu)*

(3,'paid', '2024-01-15', 'PAY003', 9,5500000,NULL),*--đơn 3 của student 9 (5,5 trieu)*

(4,'paid', '2024-01-20', 'PAY004', 10,1800000,NULL),*--đơn 4 của student 10  (1,8 trieu)*

(5,'paid', '2024-01-25', 'PAY005', 11,3000000,NULL), *-- đơn 5 của student 11 (3 trieu)*

(6,'unpaid', NULL, NULL, 12,NULL,1),  *-- đơn 6 học sinh 12 (9.3 trieu)*

(7,'unpaid', NULL, NULL, 13,NULL,5),  *--đơn 7 học sinh 13*

(8,'unpaid', NULL, NULL, 14,NULL,6),  *-- đơn 8 học sinh 14*

(9,'unpaid', NULL, NULL, 15,NULL,7),  *-- đơn 9 học sinh 15*

(10,'unpaid', NULL, NULL, 16,NULL,8),  *-- đơn 10 học sinh 16*

(11, 'unpaid',NULL,NULL,17,NULL, 9), *-- đơn 11 học sinh 17*

(12, 'unpaid', NULL,NULL,18,NULL, 7), *-- đơn 12 học sinh 18*

(13, 'unpaid',NULL,NULL,19,NULL,8);

INSERT INTO CourseOrder (CourseID, OrderID) VALUES

(1, 6),  *-- đơn 6 student 12 đăng ký course 1 (React)*

(2, 6),  *-- đơn 6 student 12 đăng ký course 3 (SQL)*

(3, 6),  *-- đơn 6 student 12 đăng ký course 3 (SQL)*

(4, 6),  *-- đơn 6 student 12 đăng ký course 3 (SQL)*

(2, 7),  *-- đơn 7 student 13 đăng ký course 2 (Android)*

(4, 8),  *-- đơn 8 student 14 đăng ký course 4 (ML)*

(1, 9),  *-- đơn 9 student 15 đăng ký course 1 (React)*

(2, 9),  *-- đơn 9 student 15 đăng ký course 2 (Android)*

(4, 10), *-- đơn 10 student 16 đăng ký course 4 (ML)*

(2,11), *--đơn 11 course 2*

(3,11), *--đơn 11 course 3*

(4,11), *--đơn 11 course 4*

(1,12), *--đơn 12*

(2,12),

(3,13);

INSERT INTO CourseStudent (CourseID, StudentID)

VALUES

(1, 7),  *--student 7 có course 1*

(2, 7),*--student 7 có course 2*

(1, 8), *--student 8 course 1*

(3, 8),*--student 8 course 3*

(2, 9),*--student 9 course 2*

(4, 9),*--student 9 course 4*

(3, 10),*--student 10 course 3*

(4, 11);*--student 11 course 4*

INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES

*-- Course 1 (React) - ReviewID phải khác nhau trong cùng CourseID*

(1, 1, 2, N'Khóa học React cực kỳ chất lượng, giảng viên nhiệt tình', 7),

(2, 1, 3, N'Tài liệu đầy đủ, dễ hiểu, rất phù hợp cho người mới bắt đầu', 8),

(3, 1, 2, N'Học được rất nhiều kiến thức thực tế về React', 9),

(4, 1, 5, N'Giảng viên giải thích rất rõ ràng, có nhiều ví dụ thực tế', 10),

(5, 1, 1, N'Khóa học giúp tôi tự tin làm việc với React', 11),

*--Course 2*

(1, 2, 1, N'Khóa học Angular rất chi tiết và chuyên sâu', 12),

(2, 2, 2, N'Nội dung được cập nhật theo phiên bản mới nhất của Angular', 13),

(3, 2, 1, N'Rất hài lòng với chất lượng giảng dạy', 14),

(4, 2, 3, N'Bài tập thực hành giúp hiểu sâu về Angular', 15),

(5, 2, 4, N'Giảng viên rất tận tâm, support nhiệt tình', 16),

*-- Course 3 (SQL)*

(1, 3, 5, N'Khóa học Node.js rất thực tế và hữu ích', 7),

(2, 3, 2, N'Được học cách xây dựng API một cách chuyên nghiệp', 8),

(3, 3, 3, N'Tài liệu phong phú, nhiều ví dụ thực tế', 9),

(4, 3, 1, N'Giảng viên có nhiều kinh nghiệm thực tế', 10),

(5, 3, 2, N'Rất hài lòng với nội dung khóa học', 11),

*-- Course 4 (ML)*

(1, 4, 5, N'Khóa học Laravel rất chất lượng', 12),

(2, 4, 4, N'Được học từ cơ bản đến nâng cao về Laravel', 13),

(3, 4, 5, N'Giảng viên hướng dẫn tận tình', 14),

(4, 4, 4, N'Nội dung khóa học được cập nhật thường xuyên', 15),

(5, 4, 5, N'Rất hữu ích cho công việc', 16);

INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES

(1, 5, 4, N'Vue.js được giảng dạy rất dễ hiểu', 7),

(2, 5, 3, N'Khóa học giúp tôi nắm vững Vue.js', 8),

(3, 5, 4, N'Tuyệt vời, đã hiểu rõ về Vue Composition API', 9),

(4, 5, 3, N'Rất thích cách trình bày của giảng viên', 10),

(5, 5, 3, N'Khóa học cập nhật những tính năng mới nhất của Vue', 11)

*-- Reviews for Course 6 (HTML)*

INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES

(1, 6, 3, N'Khóa học HTML cơ bản nhưng đầy đủ', 12),

(2, 6, 4, N'Rất phù hợp cho người mới bắt đầu', 13),

(3, 6, 4, N'Giảng viên giải thích rất rõ ràng', 14),

(4, 6, 4, N'Nhiều ví dụ thực tế hữu ích', 15),

(5, 6, 5, N'Tài liệu học tập phong phú', 16);

*-- Reviews for Course 7 (Bootstrap)*

INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES

(1, 7, 1, N'Học Bootstrap rất thú vị', 7),

(2, 7, 2, N'Khóa học giúp tôi làm chủ Bootstrap', 8),

(3, 7, 1, N'Rất nhiều project thực tế', 9),

(4, 7, 1, N'Giảng viên hướng dẫn chi tiết', 10),

(5, 7, 1, N'Tài liệu cập nhật với Bootstrap 5', 11);

*-- Reviews for Course 8 (iOS Development)*

INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES

(1, 8, 1, N'Khóa học iOS rất chuyên nghiệp', 12),

(2, 8, 2, N'Được học Swift từ cơ bản đến nâng cao', 13),

(3, 8, 3, N'Nhiều kinh nghiệm thực tế từ giảng viên', 14),

(4, 8, 4, N'Tài liệu đầy đủ và chi tiết', 15),

(5, 8, 5, N'Rất hài lòng với khóa học', 16);

*-- Reviews for Course 9 (React Native)*

INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES

(1, 9, 1, N'React Native được giảng dạy rất hay', 7),

(2, 9, 1, N'Dễ hiểu và thực tế', 8),

(3, 9, 3, N'Giảng viên nhiệt tình support', 9),

(4, 9, 2, N'Học được nhiều kinh nghiệm thực tế', 10),

(5, 9, 5, N'Khóa học rất đáng giá', 11);

*-- Reviews for Course 10 (Flutter)*

INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES

(1, 10, 5, N'Flutter được giảng dạy rất chuyên nghiệp', 12),

(2, 10, 5, N'Tài liệu cập nhật với phiên bản mới nhất', 13),

(3, 10, 4, N'Nhiều ví dụ thực tế hữu ích', 14),

(4, 10, 5, N'Giảng viên có nhiều kinh nghiệm', 15),

(5, 10, 4, N'Rất hài lòng với chất lượng', 16);

*-- Reviews for Course 11 (Kotlin)*

INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES

(1, 11, 1, N'Khóa học Kotlin rất chất lượng', 7),

(2, 11, 4, N'Được học nhiều tính năng mới của Kotlin', 8),

(3, 11, 2, N'Giảng viên giải thích rất rõ ràng', 9),

(4, 11, 5, N'Tài liệu phong phú và cập nhật', 10),

(5, 11, 5, N'Rất thích cách trình bày', 11);

*-- Reviews for Course 12 (MongoDB)*

INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES

(1, 12, 4, N'MongoDB được giảng dạy rất chi tiết', 12),

(2, 12, 2, N'Hiểu rõ về NoSQL database', 13),

(3, 12, 1, N'Nhiều bài tập thực hành', 14),

(4, 12, 5, N'Giảng viên rất nhiệt tình', 15),

(5, 12, 5, N'Khóa học rất hữu ích', 16);

*-- Reviews for Course 13 (PostgreSQL)*

INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES

(1, 13, 1, N'PostgreSQL được giảng dạy rất chuyên sâu', 7),

(2, 13, 1, N'Nhiều kiến thức nâng cao hữu ích', 8),

(3, 13, 2, N'Giảng viên có nhiều kinh nghiệm thực tế', 9),

(4, 13, 2, N'Tài liệu đầy đủ và chi tiết', 10),

(5, 13, 3, N'Rất hài lòng với khóa học', 11);

*-- Reviews for Course 14 (MySQL)*

INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES

(1, 14, 5, N'MySQL Performance được giảng dạy rất hay', 12),

(2, 14, 5, N'Học được nhiều kỹ thuật tối ưu', 13),

(3, 14, 5, N'Giảng viên chia sẻ nhiều kinh nghiệm', 14),

(4, 14, 5, N'Nội dung khóa học rất thực tế', 15),

(5, 14, 5, N'Đáng giá từng đồng', 16);

*-- Reviews for Course 15 (Oracle)*

INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES

(1, 15, 5, N'Oracle Database được giảng dạy rất chuyên nghiệp', 7),

(2, 15, 4, N'Tài liệu đầy đủ và chi tiết', 8),

(3, 15, 5, N'Giảng viên rất tận tâm', 9),

(4, 15, 1, N'Nhiều bài tập thực hành hữu ích', 10),

(5, 15, 2, N'Khóa học rất chất lượng', 11);

*-- Reviews for Course 16 (Deep Learning)*

INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES

(1, 16, 3, N'Deep Learning cơ bản được giảng dạy dễ hiểu', 12),

(2, 16, 3, N'Giảng viên giải thích rõ ràng các khái niệm', 13),

(3, 16, 3, N'Nhiều ví dụ thực tế', 14),

(4, 16, 3, N'Tài liệu học tập phong phú', 15),

(5, 16, 2, N'Rất hài lòng với khóa học', 16);

*-- Reviews for Course 17 (Computer Vision)*

INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES

(1, 17, 1, N'Computer Vision được giảng dạy rất hay', 7),

(2, 17, 2, N'Nhiều project thực tế thú vị', 8),

(3, 17, 1, N'Giảng viên có kiến thức sâu rộng', 9),

(4, 17, 1, N'Tài liệu cập nhật với công nghệ mới', 10),

(5, 17, 1, N'Khóa học rất đáng giá', 11);

*-- Reviews for Course 18 (NLP)*

INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES

(1, 18, 2, N'NLP được giảng dạy rất chuyên sâu', 12),

(2, 18, 2, N'Được học các kỹ thuật xử lý ngôn ngữ hiện đại', 13),

(3, 18, 2, N'Giảng viên nhiệt tình hỗ trợ', 14),

(4, 18, 5, N'Nội dung khóa học rất thực tế', 15),

(5, 18, 5, N'Rất hài lòng với chất lượng', 16);

*-- Reviews for Course 19 (Data Science)*

INSERT INTO Review (ReviewID, CourseID, ReviewScore, ReviewContent, StudentID) VALUES

(1, 19, 1, N'Data Science với Python rất thực tế', 7),

(2, 19, 5, N'Được học nhiều thư viện phân tích dữ liệu', 8),

(3, 19, 5, N'Giảng viên có nhiều kinh nghiệm thực tế', 9),

(4, 19, 5, N'Tài liệu đầy đủ và cập nhật', 10),

(5, 19, 5, N'Khóa học rất chất lượng', 11);

INSERT INTO Chapter (ChapterName, CourseID) VALUES

(N'Chương 1: Giới thiệu về ReactJS', 1),          *-- Chapter 1 của React*

(N'Chương 2: JSX và Components', 1),              *-- Chapter 2 của React*

(N'Chương 3: State và Props', 1),                 *-- Chapter 3 của React*

(N'Chương 4: React Hooks', 1);                    *-- Chapter 4 của React*

*-- Thêm Chapter cho Course 2 (Android)*

INSERT INTO Chapter (ChapterName, CourseID) VALUES

(N'Chương 1: Cơ bản về Android', 2),             *-- Chapter 1 của Android*

(N'Chương 2: Layout và UI', 2),                  *-- Chapter 2 của Android*

(N'Chương 3: Activities và Intents', 2),         *-- Chapter 3 của Android*

(N'Chương 4: Data Storage', 2);                  *-- Chapter 4 của Android*

*-- Thêm Chapter cho Course 3 (SQL)*

INSERT INTO Chapter (ChapterName, CourseID) VALUES

(N'Chương 1: Cơ sở về SQL', 3),                  *-- Chapter 1 của SQL*

(N'Chương 2: Truy vấn cơ bản', 3),              *-- Chapter 2 của SQL*

(N'Chương 3: Joins và Subqueries', 3),          *-- Chapter 3 của SQL*

(N'Chương 4: Stored Procedures', 3);            *-- Chapter 4 của SQL*

*-- Thêm Chapter cho Course 4 (ML)*

INSERT INTO Chapter (ChapterName, CourseID) VALUES

(N'Chương 1: Giới thiệu Machine Learning', 4),   *-- Chapter 1 của ML*

(N'Chương 2: Supervised Learning', 4),           *-- Chapter 2 của ML*

(N'Chương 3: Unsupervised Learning', 4),        *-- Chapter 3 của ML*

(N'Chương 4: Deep Learning', 4);                *-- Chapter 4 của ML*

INSERT INTO Test (TestOrder, CourseID, ChapterName, TestDuration) VALUES

*-- Tests for React Course (Course 1)*

(1, 1, N'Chương 1: Giới thiệu về ReactJS', '00:30:00'),    *-- Test 1 của Chương 1 Course React*

(2, 1, N'Chương 1: Giới thiệu về ReactJS', '00:30:00'),    *-- Test 2 của Chương 1 Course React*

(1, 1, N'Chương 2: JSX và Components', '00:30:00'),        *-- Test 1 của Chương 2 Course React*

(2, 1, N'Chương 2: JSX và Components', '00:30:00'),        *-- Test 2 của Chương 2 Course React*

(1, 1, N'Chương 3: State và Props', '00:30:00'),           *-- Test 1 của Chương 3 Course React*

(2, 1, N'Chương 3: State và Props', '00:30:00'),           *-- Test 2 của Chương 3 Course React*

(1, 1, N'Chương 4: React Hooks', '00:30:00'),              *-- Test 1 của Chương 4 Course React*

(2, 1, N'Chương 4: React Hooks', '00:30:00'),              *-- Test 2 của Chương 4 Course React*

*-- Tests for Android Course (Course 2)*

(1, 2, N'Chương 1: Cơ bản về Android', '00:30:00'),        *-- Test 1 của Chương 1 Course Android*

(2, 2, N'Chương 1: Cơ bản về Android', '00:30:00'),        *-- Test 2 của Chương 1 Course Android*

(1, 2, N'Chương 2: Layout và UI', '00:30:00'),             *-- Test 1 của Chương 2 Course Android*

(2, 2, N'Chương 2: Layout và UI', '00:30:00'),             *-- Test 2 của Chương 2 Course Android*

(1, 2, N'Chương 3: Activities và Intents', '00:30:00'),    *-- Test 1 của Chương 3 Course Android*

(2, 2, N'Chương 3: Activities và Intents', '00:30:00'),    *-- Test 2 của Chương 3 Course Android*

(1, 2, N'Chương 4: Data Storage', '00:30:00'),             *-- Test 1 của Chương 4 Course Android*

(2, 2, N'Chương 4: Data Storage', '00:30:00'),             *-- Test 2 của Chương 4 Course Android*

*-- Tests for SQL Course (Course 3)*

(1, 3, N'Chương 1: Cơ sở về SQL', '00:30:00'),             *-- Test 1 của Chương 1 Course SQL*

(2, 3, N'Chương 1: Cơ sở về SQL', '00:30:00'),             *-- Test 2 của Chương 1 Course SQL*

(1, 3, N'Chương 2: Truy vấn cơ bản', '00:30:00'),          *-- Test 1 của Chương 2 Course SQL*

(2, 3, N'Chương 2: Truy vấn cơ bản', '00:30:00'),          *-- Test 2 của Chương 2 Course SQL*

(1, 3, N'Chương 3: Joins và Subqueries', '00:30:00'),      *-- Test 1 của Chương 3 Course SQL*

(2, 3, N'Chương 3: Joins và Subqueries', '00:30:00'),      *-- Test 2 của Chương 3 Course SQL*

(1, 3, N'Chương 4: Stored Procedures', '00:30:00'),        *-- Test 1 của Chương 4 Course SQL*

(2, 3, N'Chương 4: Stored Procedures', '00:30:00'),        *-- Test 2 của Chương 4 Course SQL*

*-- Tests for ML Course (Course 4)*

(1, 4, N'Chương 1: Giới thiệu Machine Learning', '00:30:00'), *-- Test 1 của Chương 1 Course ML*

(2, 4, N'Chương 1: Giới thiệu Machine Learning', '00:30:00'), *-- Test 2 của Chương 1 Course ML*

(1, 4, N'Chương 2: Supervised Learning', '00:30:00'),         *-- Test 1 của Chương 2 Course ML*

(2, 4, N'Chương 2: Supervised Learning', '00:30:00'),         *-- Test 2 của Chương 2 Course ML*

(1, 4, N'Chương 3: Unsupervised Learning', '00:30:00'),      *-- Test 1 của Chương 3 Course ML*

(2, 4, N'Chương 3: Unsupervised Learning', '00:30:00'),      *-- Test 2 của Chương 3 Course ML*

(1, 4, N'Chương 4: Deep Learning', '00:30:00'),              *-- Test 1 của Chương 4 Course ML*

(2, 4, N'Chương 4: Deep Learning', '00:30:00');              *-- Test 2 của Chương 4 Course ML*

INSERT INTO Question (QuestionScore, QuestionContent) VALUES

*-- React Questions (1-5)*

(2, N'React là gì?'),                                          *-- Question 1*

(2, N'JSX trong React là gì?'),                               *-- Question 2*

(2, N'Props trong React được sử dụng để làm gì?'),            *-- Question 3*

(2, N'State trong React là gì?'),                             *-- Question 4*

(2, N'React Hooks được giới thiệu từ phiên bản nào?'),        *-- Question 5*

*-- Android Questions (6-10)*

(2, N'Android Studio là gì?'),                                *-- Question 6*

(2, N'Layout trong Android là gì?'),                          *-- Question 7*

(2, N'Activity trong Android là gì?'),                        *-- Question 8*

(2, N'Intent trong Android được sử dụng để làm gì?'),         *-- Question 9*

(2, N'SQLite trong Android được sử dụng để làm gì?'),         *-- Question 10*

*-- SQL Questions (11-15)*

(2, N'Primary Key là gì?'),                                   *-- Question 11*

(2, N'JOIN trong SQL là gì?'),                               *-- Question 12*

(2, N'Stored Procedure là gì?'),                             *-- Question 13*

(2, N'Transaction trong SQL Server là gì?'),                 *-- Question 14*

(2, N'Index trong SQL Server dùng để làm gì?'),              *-- Question 15*

*-- ML Questions (16-20)*

(2, N'Machine Learning là gì?'),                             *-- Question 16*

(2, N'Supervised Learning là gì?'),                          *-- Question 17*

(2, N'Unsupervised Learning là gì?'),                        *-- Question 18*

(2, N'Deep Learning là gì?'),                                *-- Question 19*

(2, N'Neural Network là gì?');                               *-- Question 20*

*-- Insert Options for each Question*

INSERT INTO Options (QuestionID, OptionName, OptionContent, IsCorrect) VALUES

*-- Options for Question 1 (React là gì?)*

(1, 'A1', N'Một framework JavaScript', 0),

(1, 'B1', N'Một thư viện JavaScript để xây dựng giao diện người dùng', 1),

(1, 'C1', N'Một ngôn ngữ lập trình', 0),

(1, 'D1', N'Một cơ sở dữ liệu', 0),

*-- Options for Question 2 (JSX trong React là gì?)*

(2, 'A2', N'Một extension của JavaScript cho phép viết HTML trong JavaScript', 1),

(2, 'B2', N'Một ngôn ngữ lập trình mới', 0),

(2, 'C2', N'Một framework', 0),

(2, 'D2', N'Một thư viện độc lập', 0),

*-- Options for Question 3 (Props trong React)*

(3, 'A3', N'Để lưu trữ state của component', 0),

(3, 'B3', N'Để truyền dữ liệu giữa các component', 1),

(3, 'C3', N'Để tạo routing', 0),

(3, 'D3', N'Để kết nối database', 0),

*-- Options for Question 4 (State trong React)*

(4, 'A4', N'Dữ liệu tĩnh không thay đổi', 0),

(4, 'B4', N'Dữ liệu động có thể thay đổi trong component', 1),

(4, 'C4', N'Thuộc tính CSS', 0),

(4, 'D4', N'Tên component', 0),

*-- Options for Question 5 (React Hooks)*

(5, 'A5', N'React 15.0', 0),

(5, 'B5', N'React 16.8', 1),

(5, 'C5', N'React 17.0', 0),

(5, 'D5', N'React 18.0', 0),

*-- Options for Question 6 (Android Studio là gì?)*

(6, 'A6', N'IDE chính thức để phát triển ứng dụng Android', 1),

(6, 'B6', N'Một trình duyệt web', 0),

(6, 'C6', N'Một hệ điều hành', 0),

(6, 'D6', N'Một ngôn ngữ lập trình', 0),

*-- Options for Question 7 (Layout trong Android là gì?)*

(7, 'A7', N'Một dạng cơ sở dữ liệu', 0),

(7, 'B7', N'Cấu trúc sắp xếp các thành phần giao diện người dùng', 1),

(7, 'C7', N'Một kiểu biến trong Android', 0),

(7, 'D7', N'Một công cụ debug', 0),

*-- Options for Question 8 (Activity trong Android là gì?)*

(8, 'A8', N'Một file cấu hình', 0),

(8, 'B8', N'Một component đại diện cho một màn hình trong ứng dụng', 1),

(8, 'C8', N'Một thư viện đồ họa', 0),

(8, 'D8', N'Một loại database', 0),

*-- Options for Question 9 (Intent trong Android)*

(9, 'A9', N'Một cơ chế để chuyển đổi giữa các Activity', 1),

(9, 'B9', N'Một kiểu dữ liệu', 0),

(9, 'C9', N'Một framework testing', 0),

(9, 'D9', N'Một công cụ build', 0),

*-- Options for Question 10 (SQLite trong Android)*

(10, 'A10', N'Một framework UI', 0),

(10, 'B10', N'Một hệ quản trị cơ sở dữ liệu nhẹ tích hợp trong Android', 1),

(10, 'C10', N'Một công cụ debug', 0),

(10, 'D10', N'Một thư viện đồ họa', 0),

*-- Options for Question 11 (Primary Key là gì?)*

(11, 'A11', N'Khóa dùng để mở database', 0),

(11, 'B11', N'Trường duy nhất định danh cho mỗi bản ghi trong bảng', 1),

(11, 'C11', N'Một loại index', 0),

(11, 'D11', N'Một công cụ tối ưu', 0),

*-- Options for Question 12 (JOIN trong SQL là gì?)*

(12, 'A12', N'Lệnh để tạo bảng mới', 0),

(12, 'B12', N'Phương thức kết hợp dữ liệu từ nhiều bảng', 1),

(12, 'C12', N'Công cụ backup database', 0),

(12, 'D12', N'Một loại constraint', 0),

*-- Options for Question 13 (Stored Procedure là gì?)*

(13, 'A13', N'Một tập hợp các câu lệnh SQL được biên dịch sẵn', 1),

(13, 'B13', N'Một loại table', 0),

(13, 'C13', N'Một công cụ quản lý user', 0),

(13, 'D13', N'Một kiểu dữ liệu', 0),

*-- Options for Question 14 (Transaction trong SQL Server)*

(14, 'A14', N'Một công cụ backup', 0),

(14, 'B14', N'Một đơn vị công việc đảm bảo tính toàn vẹn dữ liệu', 1),

(14, 'C14', N'Một loại index', 0),

(14, 'D14', N'Một kiểu join', 0),

*-- Options for Question 15 (Index trong SQL Server)*

(15, 'A15', N'Một cấu trúc để tối ưu hóa truy vấn', 1),

(15, 'B15', N'Một loại bảng', 0),

(15, 'C15', N'Một công cụ backup', 0),

(15, 'D15', N'Một kiểu constraint', 0),

*-- Options for Question 16 (Machine Learning là gì?)*

(16, 'A16', N'Một ngôn ngữ lập trình', 0),

(16, 'B16', N'Khả năng máy tính học từ dữ liệu mà không cần lập trình tường minh', 1),

(16, 'C16', N'Một hệ điều hành', 0),

(16, 'D16', N'Một công cụ phần mềm', 0),

*-- Options for Question 17 (Supervised Learning là gì?)*

(17, 'A17', N'Học có giám sát với dữ liệu được gán nhãn', 1),

(17, 'B17', N'Học không giám sát', 0),

(17, 'C17', N'Học tăng cường', 0),

(17, 'D17', N'Deep Learning', 0),

*-- Options for Question 18 (Unsupervised Learning là gì?)*

(18, 'A18', N'Học có giám sát', 0),

(18, 'B18', N'Học không giám sát với dữ liệu không gán nhãn', 1),

(18, 'C18', N'Học tăng cường', 0),

(18, 'D18', N'Transfer Learning', 0),

*-- Options for Question 19 (Deep Learning là gì?)*

(19, 'A19', N'Một thuật toán Machine Learning', 0),

(19, 'B19', N'Một phương pháp học sâu sử dụng nhiều lớp neural network', 1),

(19, 'C19', N'Một công cụ phần mềm', 0),

(19, 'D19', N'Một ngôn ngữ lập trình', 0),

*-- Options for Question 20 (Neural Network là gì?)*

(20, 'A20', N'Một thuật toán đơn giản', 0),

(20, 'B20', N'Một mô hình học máy lấy cảm hứng từ não người', 1),

(20, 'C20', N'Một cơ sở dữ liệu', 0),

(20, 'D20', N'Một công cụ lập trình', 0);

*-- Link Questions to Tests (5 questions per test)*

INSERT INTO QuestionTest (QuestionID, TestOrder, CourseID, ChapterName) VALUES

*-- Questions for React Course (Course 1)*

*-- Chương 1: Giới thiệu về ReactJS*

(1, 1, 1, N'Chương 1: Giới thiệu về ReactJS'),    *-- Test 1 của Chương 1 React Question 1*

(2, 1, 1, N'Chương 1: Giới thiệu về ReactJS'),    *-- Test 1 của Chương 1 React Question 2*

(3, 1, 1, N'Chương 1: Giới thiệu về ReactJS'),    *-- Test 1 của Chương 1 React Question 3*

(4, 1, 1, N'Chương 1: Giới thiệu về ReactJS'),    *-- Test 1 của Chương 1 React Question 4*

(5, 1, 1, N'Chương 1: Giới thiệu về ReactJS'),    *-- Test 1 của Chương 1 React Question 5*

*-- Test 2 của Chương 1 React*

(6, 2, 1, N'Chương 1: Giới thiệu về ReactJS'),    *-- Test 2 của Chương 1 React Question 6*

(7, 2, 1, N'Chương 1: Giới thiệu về ReactJS'),    *-- Test 2 của Chương 1 React Question 7*

(8, 2, 1, N'Chương 1: Giới thiệu về ReactJS'),    *-- Test 2 của Chương 1 React Question 8*

(9, 2, 1, N'Chương 1: Giới thiệu về ReactJS'),    *-- Test 2 của Chương 1 React Question 9*

(10, 2, 1, N'Chương 1: Giới thiệu về ReactJS'),   *-- Test 2 của Chương 1 React Question 10*

*-- Chương 2: JSX và Components*

(11, 1, 1, N'Chương 2: JSX và Components'),        *-- Test 1 của Chương 2 React Question 11*

(12, 1, 1, N'Chương 2: JSX và Components'),        *-- Test 1 của Chương 2 React Question 12*

(13, 1, 1, N'Chương 2: JSX và Components'),        *-- Test 1 của Chương 2 React Question 13*

(14, 1, 1, N'Chương 2: JSX và Components'),        *-- Test 1 của Chương 2 React Question 14*

(15, 1, 1, N'Chương 2: JSX và Components'),        *-- Test 1 của Chương 2 React Question 15*

*-- Test 2 của Chương 2 React*

(16, 2, 1, N'Chương 2: JSX và Components'),        *-- Test 2 của Chương 2 React Question 16*

(17, 2, 1, N'Chương 2: JSX và Components'),        *-- Test 2 của Chương 2 React Question 17*

(18, 2, 1, N'Chương 2: JSX và Components'),        *-- Test 2 của Chương 2 React Question 18*

(19, 2, 1, N'Chương 2: JSX và Components'),        *-- Test 2 của Chương 2 React Question 19*

(20, 2, 1, N'Chương 2: JSX và Components'),        *-- Test 2 của Chương 2 React Question 20*

*-- Chương 3: State và Props*

(1, 1, 1, N'Chương 3: State và Props'),          *-- Test 1 của Chương 3 React*

(2, 1, 1, N'Chương 3: State và Props'),

(3, 1, 1, N'Chương 3: State và Props'),

(4, 1, 1, N'Chương 3: State và Props'),

(5, 1, 1, N'Chương 3: State và Props'),

(6, 2, 1, N'Chương 3: State và Props'),          *-- Test 2 của Chương 3 React*

(7, 2, 1, N'Chương 3: State và Props'),

(8, 2, 1, N'Chương 3: State và Props'),

(9, 2, 1, N'Chương 3: State và Props'),

(10, 2, 1, N'Chương 3: State và Props'),

*-- Chương 4: React Hooks*

(11, 1, 1, N'Chương 4: React Hooks'),            *-- Test 1 của Chương 4 React*

(12, 1, 1, N'Chương 4: React Hooks'),

(13, 1, 1, N'Chương 4: React Hooks'),

(14, 1, 1, N'Chương 4: React Hooks'),

(15, 1, 1, N'Chương 4: React Hooks'),

(16, 2, 1, N'Chương 4: React Hooks'),            *-- Test 2 của Chương 4 React*

(17, 2, 1, N'Chương 4: React Hooks'),

(18, 2, 1, N'Chương 4: React Hooks'),

(19, 2, 1, N'Chương 4: React Hooks'),

(20, 2, 1, N'Chương 4: React Hooks'),

*-- COURSE 2: ANDROID*

*-- Chương 1: Cơ bản về Android*

(1, 1, 2, N'Chương 1: Cơ bản về Android'),       *-- Test 1 của Chương 1 Android*

(2, 1, 2, N'Chương 1: Cơ bản về Android'),

(3, 1, 2, N'Chương 1: Cơ bản về Android'),

(4, 1, 2, N'Chương 1: Cơ bản về Android'),

(5, 1, 2, N'Chương 1: Cơ bản về Android'),

(6, 2, 2, N'Chương 1: Cơ bản về Android'),       *-- Test 2 của Chương 1 Android*

(7, 2, 2, N'Chương 1: Cơ bản về Android'),

(8, 2, 2, N'Chương 1: Cơ bản về Android'),

(9, 2, 2, N'Chương 1: Cơ bản về Android'),

(10, 2, 2, N'Chương 1: Cơ bản về Android'),

*-- Chương 2: Layout và UI*

(11, 1, 2, N'Chương 2: Layout và UI'),           *-- Test 1 của Chương 2 Android*

(12, 1, 2, N'Chương 2: Layout và UI'),

(13, 1, 2, N'Chương 2: Layout và UI'),

(14, 1, 2, N'Chương 2: Layout và UI'),

(15, 1, 2, N'Chương 2: Layout và UI'),

(16, 2, 2, N'Chương 2: Layout và UI'),           *-- Test 2 của Chương 2 Android*

(17, 2, 2, N'Chương 2: Layout và UI'),

(18, 2, 2, N'Chương 2: Layout và UI'),

(19, 2, 2, N'Chương 2: Layout và UI'),

(20, 2, 2, N'Chương 2: Layout và UI'),

*-- Chương 3: Activities và Intents*

(1, 1, 2, N'Chương 3: Activities và Intents'),    *-- Test 1 của Chương 3 Android*

(2, 1, 2, N'Chương 3: Activities và Intents'),

(3, 1, 2, N'Chương 3: Activities và Intents'),

(4, 1, 2, N'Chương 3: Activities và Intents'),

(5, 1, 2, N'Chương 3: Activities và Intents'),

(6, 2, 2, N'Chương 3: Activities và Intents'),    *-- Test 2 của Chương 3 Android*

(7, 2, 2, N'Chương 3: Activities và Intents'),

(8, 2, 2, N'Chương 3: Activities và Intents'),

(9, 2, 2, N'Chương 3: Activities và Intents'),

(10, 2, 2, N'Chương 3: Activities và Intents'),

*-- Chương 4: Data Storage*

(11, 1, 2, N'Chương 4: Data Storage'),            *-- Test 1 của Chương 4 Android*

(12, 1, 2, N'Chương 4: Data Storage'),

(13, 1, 2, N'Chương 4: Data Storage'),

(14, 1, 2, N'Chương 4: Data Storage'),

(15, 1, 2, N'Chương 4: Data Storage'),

(16, 2, 2, N'Chương 4: Data Storage'),            *-- Test 2 của Chương 4 Android*

(17, 2, 2, N'Chương 4: Data Storage'),

(18, 2, 2, N'Chương 4: Data Storage'),

(19, 2, 2, N'Chương 4: Data Storage'),

(20, 2, 2, N'Chương 4: Data Storage'),

*-- COURSE 3: SQL*

*-- Chương 1: Cơ sở về SQL*

(1, 1, 3, N'Chương 1: Cơ sở về SQL'),             *-- Test 1 của Chương 1 SQL*

(2, 1, 3, N'Chương 1: Cơ sở về SQL'),

(3, 1, 3, N'Chương 1: Cơ sở về SQL'),

(4, 1, 3, N'Chương 1: Cơ sở về SQL'),

(5, 1, 3, N'Chương 1: Cơ sở về SQL'),

(6, 2, 3, N'Chương 1: Cơ sở về SQL'),             *-- Test 2 của Chương 1 SQL*

(7, 2, 3, N'Chương 1: Cơ sở về SQL'),

(8, 2, 3, N'Chương 1: Cơ sở về SQL'),

(9, 2, 3, N'Chương 1: Cơ sở về SQL'),

(10, 2, 3, N'Chương 1: Cơ sở về SQL'),

*-- Chương 2: Truy vấn cơ bản*

(11, 1, 3, N'Chương 2: Truy vấn cơ bản'),         *-- Test 1 của Chương 2 SQL*

(12, 1, 3, N'Chương 2: Truy vấn cơ bản'),

(13, 1, 3, N'Chương 2: Truy vấn cơ bản'),

(14, 1, 3, N'Chương 2: Truy vấn cơ bản'),

(15, 1, 3, N'Chương 2: Truy vấn cơ bản'),

(16, 2, 3, N'Chương 2: Truy vấn cơ bản'),         *-- Test 2 của Chương 2 SQL*

(17, 2, 3, N'Chương 2: Truy vấn cơ bản'),

(18, 2, 3, N'Chương 2: Truy vấn cơ bản'),

(19, 2, 3, N'Chương 2: Truy vấn cơ bản'),

(20, 2, 3, N'Chương 2: Truy vấn cơ bản'),

*-- Chương 3: Joins và Subqueries*

(1, 1, 3, N'Chương 3: Joins và Subqueries'),      *-- Test 1 của Chương 3 SQL*

(2, 1, 3, N'Chương 3: Joins và Subqueries'),

(3, 1, 3, N'Chương 3: Joins và Subqueries'),

(4, 1, 3, N'Chương 3: Joins và Subqueries'),

(5, 1, 3, N'Chương 3: Joins và Subqueries'),

(6, 2, 3, N'Chương 3: Joins và Subqueries'),      *-- Test 2 của Chương 3 SQL*

(7, 2, 3, N'Chương 3: Joins và Subqueries'),

(8, 2, 3, N'Chương 3: Joins và Subqueries'),

(9, 2, 3, N'Chương 3: Joins và Subqueries'),

(10, 2, 3, N'Chương 3: Joins và Subqueries'),

*-- Chương 4: Stored Procedures*

(11, 1, 3, N'Chương 4: Stored Procedures'),        *-- Test 1 của Chương 4 SQL*

(12, 1, 3, N'Chương 4: Stored Procedures'),

(13, 1, 3, N'Chương 4: Stored Procedures'),

(14, 1, 3, N'Chương 4: Stored Procedures'),

(15, 1, 3, N'Chương 4: Stored Procedures'),

(16, 2, 3, N'Chương 4: Stored Procedures'),        *-- Test 2 của Chương 4 SQL*

(17, 2, 3, N'Chương 4: Stored Procedures'),

(18, 2, 3, N'Chương 4: Stored Procedures'),

(19, 2, 3, N'Chương 4: Stored Procedures'),

(20, 2, 3, N'Chương 4: Stored Procedures'),

*-- COURSE 4: MACHINE LEARNING*

*-- Chương 1: Giới thiệu Machine Learning*

(1, 1, 4, N'Chương 1: Giới thiệu Machine Learning'), *-- Test 1 của Chương 1 ML*

(2, 1, 4, N'Chương 1: Giới thiệu Machine Learning'),

(3, 1, 4, N'Chương 1: Giới thiệu Machine Learning'),

(4, 1, 4, N'Chương 1: Giới thiệu Machine Learning'),

(5, 1, 4, N'Chương 1: Giới thiệu Machine Learning'),

(6, 2, 4, N'Chương 1: Giới thiệu Machine Learning'), *-- Test 2 của Chương 1 ML*

(7, 2, 4, N'Chương 1: Giới thiệu Machine Learning'),

(8, 2, 4, N'Chương 1: Giới thiệu Machine Learning'),

(9, 2, 4, N'Chương 1: Giới thiệu Machine Learning'),

(10, 2, 4, N'Chương 1: Giới thiệu Machine Learning'),

*-- Chương 2: Supervised Learning*

(11, 1, 4, N'Chương 2: Supervised Learning'),       *-- Test 1 của Chương 2 ML*

(12, 1, 4, N'Chương 2: Supervised Learning'),

(13, 1, 4, N'Chương 2: Supervised Learning'),

(14, 1, 4, N'Chương 2: Supervised Learning'),

(15, 1, 4, N'Chương 2: Supervised Learning'),

(16, 2, 4, N'Chương 2: Supervised Learning'),       *-- Test 2 của Chương 2 ML*

(17, 2, 4, N'Chương 2: Supervised Learning'),

(18, 2, 4, N'Chương 2: Supervised Learning'),

(19, 2, 4, N'Chương 2: Supervised Learning'),

(20, 2, 4, N'Chương 2: Supervised Learning'),

*-- Chương 3: Unsupervised Learning*

(1, 1, 4, N'Chương 3: Unsupervised Learning'),     *-- Test 1 của Chương 3 ML*

(2, 1, 4, N'Chương 3: Unsupervised Learning'),

(3, 1, 4, N'Chương 3: Unsupervised Learning'),

(4, 1, 4, N'Chương 3: Unsupervised Learning'),

(5, 1, 4, N'Chương 3: Unsupervised Learning'),

(6, 2, 4, N'Chương 3: Unsupervised Learning'),     *-- Test 2 của Chương 3 ML*

(7, 2, 4, N'Chương 3: Unsupervised Learning'),

(8, 2, 4, N'Chương 3: Unsupervised Learning'),

(9, 2, 4, N'Chương 3: Unsupervised Learning'),

(10, 2, 4, N'Chương 3: Unsupervised Learning'),

*-- Chương 4: Deep Learning*

(11, 1, 4, N'Chương 4: Deep Learning'),            *-- Test 1 của Chương 4 ML*

(12, 1, 4, N'Chương 4: Deep Learning'),

(13, 1, 4, N'Chương 4: Deep Learning'),

(14, 1, 4, N'Chương 4: Deep Learning'),

(15, 1, 4, N'Chương 4: Deep Learning'),

(16, 2, 4, N'Chương 4: Deep Learning'),            *-- Test 2 của Chương 4 ML*

(17, 2, 4, N'Chương 4: Deep Learning'),

(18, 2, 4, N'Chương 4: Deep Learning'),

(19, 2, 4, N'Chương 4: Deep Learning'),

(20, 2, 4, N'Chương 4: Deep Learning');

*-- Student 7 - Course 1 (React)*

INSERT INTO Student\_Test (StudentID, CourseID, TestOrder, ChapterName, TestScore)

VALUES

(7, 1, 1, N'Chương 1: Giới thiệu về ReactJS', 0),

(7, 1, 2, N'Chương 1: Giới thiệu về ReactJS', 0),

(7, 1, 1, N'Chương 2: JSX và Components', 0),

(7, 1, 2, N'Chương 2: JSX và Components', 0),

(7, 1, 1, N'Chương 3: State và Props', 0),

(7, 1, 2, N'Chương 3: State và Props', 0),

(7, 1, 1, N'Chương 4: React Hooks', 0),

(7, 1, 2, N'Chương 4: React Hooks', 0);

*-- Student 7 - Course 2 (Android)*

INSERT INTO Student\_Test (StudentID, CourseID, TestOrder, ChapterName, TestScore)

VALUES

(7, 2, 1, N'Chương 1: Cơ bản về Android', 0),

(7, 2, 2, N'Chương 1: Cơ bản về Android', 0),

(7, 2, 1, N'Chương 2: Layout và UI', 0),

(7, 2, 2, N'Chương 2: Layout và UI', 0),

(7, 2, 1, N'Chương 3: Activities và Intents', 0),

(7, 2, 2, N'Chương 3: Activities và Intents', 0),

(7, 2, 1, N'Chương 4: Data Storage', 0),

(7, 2, 2, N'Chương 4: Data Storage', 0);

*-- Student 8 - Course 1 (React)*

INSERT INTO Student\_Test (StudentID, CourseID, TestOrder, ChapterName, TestScore)

VALUES

(8, 1, 1, N'Chương 1: Giới thiệu về ReactJS', 0),

(8, 1, 2, N'Chương 1: Giới thiệu về ReactJS', 0),

(8, 1, 1, N'Chương 2: JSX và Components', 0),

(8, 1, 2, N'Chương 2: JSX và Components', 0),

(8, 1, 1, N'Chương 3: State và Props', 0),

(8, 1, 2, N'Chương 3: State và Props', 0),

(8, 1, 1, N'Chương 4: React Hooks', 0),

(8, 1, 2, N'Chương 4: React Hooks', 0);

*-- Student 8 - Course 3 (SQL)*

INSERT INTO Student\_Test (StudentID, CourseID, TestOrder, ChapterName, TestScore)

VALUES

(8, 3, 1, N'Chương 1: Cơ sở về SQL', 0),

(8, 3, 2, N'Chương 1: Cơ sở về SQL', 0),

(8, 3, 1, N'Chương 2: Truy vấn cơ bản', 0),

(8, 3, 2, N'Chương 2: Truy vấn cơ bản', 0),

(8, 3, 1, N'Chương 3: Joins và Subqueries', 0),

(8, 3, 2, N'Chương 3: Joins và Subqueries', 0),

(8, 3, 1, N'Chương 4: Stored Procedures', 0),

(8, 3, 2, N'Chương 4: Stored Procedures', 0);

*-- Student 9 - Course 2 (Android)*

INSERT INTO Student\_Test (StudentID, CourseID, TestOrder, ChapterName, TestScore)

VALUES

(9, 2, 1, N'Chương 1: Cơ bản về Android', 0),

(9, 2, 2, N'Chương 1: Cơ bản về Android', 0),

(9, 2, 1, N'Chương 2: Layout và UI', 0),

(9, 2, 2, N'Chương 2: Layout và UI', 0),

(9, 2, 1, N'Chương 3: Activities và Intents', 0),

(9, 2, 2, N'Chương 3: Activities và Intents', 0),

(9, 2, 1, N'Chương 4: Data Storage', 0),

(9, 2, 2, N'Chương 4: Data Storage', 0);

*-- Student 9 - Course 4 (ML)*

INSERT INTO Student\_Test (StudentID, CourseID, TestOrder, ChapterName, TestScore)

VALUES

(9, 4, 1, N'Chương 1: Giới thiệu Machine Learning', 0),

(9, 4, 2, N'Chương 1: Giới thiệu Machine Learning', 0),

(9, 4, 1, N'Chương 2: Supervised Learning', 0),

(9, 4, 2, N'Chương 2: Supervised Learning', 0),

(9, 4, 1, N'Chương 3: Unsupervised Learning', 0),

(9, 4, 2, N'Chương 3: Unsupervised Learning', 0),

(9, 4, 1, N'Chương 4: Deep Learning', 0),

(9, 4, 2, N'Chương 4: Deep Learning', 0);

*-- Student 10 - Course 3 (SQL)*

INSERT INTO Student\_Test (StudentID, CourseID, TestOrder, ChapterName, TestScore)

VALUES

(10, 3, 1, N'Chương 1: Cơ sở về SQL', 0),

(10, 3, 2, N'Chương 1: Cơ sở về SQL', 0),

(10, 3, 1, N'Chương 2: Truy vấn cơ bản', 0),

(10, 3, 2, N'Chương 2: Truy vấn cơ bản', 0),

(10, 3, 1, N'Chương 3: Joins và Subqueries', 0),

(10, 3, 2, N'Chương 3: Joins và Subqueries', 0),

(10, 3, 1, N'Chương 4: Stored Procedures', 0),

(10, 3, 2, N'Chương 4: Stored Procedures', 0);

*-- Student 11 - Course 4 (ML)*

INSERT INTO Student\_Test (StudentID, CourseID, TestOrder, ChapterName, TestScore)

VALUES

(11, 4, 1, N'Chương 1: Giới thiệu Machine Learning', 0),

(11, 4, 2, N'Chương 1: Giới thiệu Machine Learning', 0),

(11, 4, 1, N'Chương 2: Supervised Learning', 0),

(11, 4, 2, N'Chương 2: Supervised Learning', 0),

(11, 4, 1, N'Chương 3: Unsupervised Learning', 0),

(11, 4, 2, N'Chương 3: Unsupervised Learning', 0),

(11, 4, 1, N'Chương 4: Deep Learning', 0),

(11, 4, 2, N'Chương 4: Deep Learning', 0);

INSERT INTO Lesson (LessonOrder, CourseID, ChapterName, LessonTitle, LessonContent, LessonDuraion) VALUES

*-- Lessons của Chapter 1 React*

(1, 1, N'Chương 1: Giới thiệu về ReactJS', N'Bài 1: Tổng quan về ReactJS', N'Giới thiệu về ReactJS và ứng dụng', '01:30:00'),

(2, 1, N'Chương 1: Giới thiệu về ReactJS', N'Bài 2: Cài đặt môi trường', N'Hướng dẫn cài đặt NodeJS và create-react-app', '01:00:00'),

*-- Lessons của Chapter 2 React*

(1, 1, N'Chương 2: JSX và Components', N'Bài 1: JSX là gì', N'Tìm hiểu về JSX và cách sử dụng', '02:00:00'),

(2, 1, N'Chương 2: JSX và Components', N'Bài 2: Components', N'Các loại components trong React', '01:45:00');

*-- Thêm Exercise cho các Lesson của React*

INSERT INTO Exercise (ExerciseOrder, LessonOrder, CourseID, ChapterName, ExerciseTitle, ExerciseContent, ExerciseAnswer, ExerciseCorrectAnswerNumber) VALUES

*-- Exercise cho Lesson 1 Chapter 1 React*

(1, 1, 1, N'Chương 1: Giới thiệu về ReactJS', N'React là gì?', N'Chọn định nghĩa đúng về React: A. Framework, B. Library, C. Programming Language', 'B', 2),

(2, 1, 1, N'Chương 1: Giới thiệu về ReactJS', N'Ai phát triển React?', N'Ai phát triển React: A. Google, B. Facebook, C. Microsoft', 'B', 2),

*-- Exercise cho Lesson 2 Chapter 1 React*

(1, 2, 1, N'Chương 1: Giới thiệu về ReactJS', N'NodeJS version', N'Version tối thiểu của NodeJS để chạy React là: A. 10.x, B. 12.x, C. 14.x', 'C', 3);

*-- Thêm Document cho các Course*

INSERT INTO Document (CourseID, ChapterName, DocumentAuthor, DocumentTitle, DocumentSize, DocumentType, DocumentContent) VALUES

*-- Documents cho React Course*

(1, N'Chương 1: Giới thiệu về ReactJS', N'React Team', N'React Fundamentals.pdf', 1024, 'pdf', N'Link to React Fundamentals PDF'),

(1, N'Chương 2: JSX và Components', N'React Team', N'JSX Guide.pdf', 2048, 'pdf', N'Link to JSX Guide PDF'),

(1, N'Chương 3: State và Props', N'React Team', N'State Management.pdf', 1536, 'pdf', N'Link to State Management PDF'),

*-- Documents cho Android Course*

(2, N'Chương 1: Cơ bản về Android', N'Android Team', N'Android Basics.pdf', 2048, 'pdf', N'Link to Android Basics PDF'),

(2, N'Chương 2: Layout và UI', N'Android Team', N'UI Design Guide.pdf', 1536, 'pdf', N'Link to UI Design PDF'),

*-- Documents cho SQL Course*

(3, N'Chương 1: Cơ sở về SQL', N'SQL Team', N'SQL Fundamentals.pdf', 1024, 'pdf', N'Link to SQL Fundamentals PDF'),

(3, N'Chương 2: Truy vấn cơ bản', N'SQL Team', N'Basic Queries.pdf', 1536, 'pdf', N'Link to Basic Queries PDF'),

*-- Documents cho ML Course*

(4, N'Chương 1: Giới thiệu Machine Learning', N'ML Team', N'ML Introduction.pdf', 2048, 'pdf', N'Link to ML Introduction PDF'),

(4, N'Chương 2: Supervised Learning', N'ML Team', N'Supervised Learning Guide.pdf', 1536, 'pdf', N'Link to Supervised Learning PDF');

INSERT INTO Guardian (StudentID, GuardianName, GuardianEmail, GuardianPhone) VALUES

*-- Guardian cho student 7 (Trần Văn G)*

(7, N'Trần Văn Minh', 'tvanminh@gmail.com', '0912345678'),

*-- Guardian cho student 8 (Nguyễn Thị H)*

(8, N'Nguyễn Văn Nam', 'nvnam@gmail.com', '0923456789'),

*-- Guardian cho student 9 (Lê Văn I)*

(9, N'Lê Thị Hương', 'lthuong@gmail.com', '0934567890'),

*-- Guardian cho student 10 (Phạm Thị K)*

(10, N'Phạm Văn Đức', 'pvduc@gmail.com', '0945678901'),

*-- Guardian cho student 11 (Hoàng Văn L)*

(11, N'Hoàng Thị Mai', 'htmai@gmail.com', '0956789012'),

*-- Guardian cho student 12 (Ngô Thị M)*

(12, N'Ngô Văn Hùng', 'nvhung@gmail.com', '0967890123'),

*-- Guardian cho student 13 (Vũ Văn N)*

(13, N'Vũ Thị Lan', 'vtlan@gmail.com', '0978901234'),

*-- Guardian cho student 14 (Đặng Thị O)*

(14, N'Đặng Văn Thành', 'dvthanh@gmail.com', '0989012345'),

*-- Guardian cho student 15 (Bùi Văn P)*

(15, N'Bùi Thị Hà', 'btha@gmail.com', '0990123456'),

*-- Guardian cho student 16 (Mai Thị Q)*

(16, N'Mai Văn Long', 'mvlong@gmail.com', '0901234567'),

*-- Guardian cho student 17 (Lý Văn R)*

(17, N'Lý Thị Ngọc', 'ltngoc@gmail.com', '0912345670'),

*-- Guardian cho student 18 (Trịnh Thị S)*

(18, N'Trịnh Văn Tùng', 'tvtung@gmail.com', '0923456781'),

*-- Guardian cho student 19 (Đinh Văn T)*

(19, N'Đinh Thị Thảo', 'dtthao@gmail.com', '0934567892'),

*-- Guardian cho student 20 (Cao Thị U)*

(20, N'Cao Văn Phong', 'cvphong@gmail.com', '0945678903');

INSERT INTO Edit (EditTime, EditDescription, EditAdminID, EditCouponID, EditCourseID) VALUES

*-- Chỉnh sửa khóa học*

('13:00:00', N'Cập nhật mô tả khóa học React', 1, NULL, 1),

('14:30:00', N'Thay đổi giá khóa học Android', 1, NULL, 2),

('15:45:00', N'Cập nhật nội dung khóa học SQL', 2, NULL, 3),

('16:20:00', N'Điều chỉnh thời gian học ML', 2, NULL, 4),

*-- Chỉnh sửa coupon*

('09:00:00', N'Tạo mới coupon giảm 15%', 1, 1, NULL),

('09:30:00', N'Điều chỉnh hạn sử dụng coupon 25%', 1, 2, NULL);

*--------------- TRIGGER để check trước khi insert dữ liệu ----------------*

*-- Trigger khi insert dữ liệu vào bảng Users*

GO

CREATE TRIGGER trg\_Users\_Insert

ON Users

INSTEAD OF INSERT

AS

BEGIN

    DECLARE @UserEmail NVARCHAR(255), @UserPhone VARCHAR(10);

    SELECT @UserEmail = UserEmail, @UserPhone = UserPhone

    FROM inserted;

*-- Validate UserEmail*

    IF @UserEmail IS NULL OR @UserEmail NOT LIKE '%\_@\_\_%.\_\_%'

    BEGIN

        RAISERROR ('Invalid email format', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- Validate UserPhone*

    IF LEN(@UserPhone) <> 10 OR @UserPhone NOT LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'

    BEGIN

        RAISERROR ('Invalid phone number', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- Validate unique UserEmail*

    IF EXISTS (SELECT 1 FROM Users WHERE UserEmail = @UserEmail)

    BEGIN

        RAISERROR ('UserEmail must be unique', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- Validate unique UserPhone*

    IF EXISTS (SELECT 1 FROM Users WHERE UserPhone = @UserPhone)

    BEGIN

        RAISERROR ('UserPhone must be unique', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- nếu không có lỗi thì insert dữ liệu*

    INSERT INTO Users (UserEmail, UserPhone, UserPassword, UserAddress, UserImage, Username)

    SELECT UserEmail, UserPhone, UserPassword, UserAddress, UserImage, Username

    FROM inserted;

END;

*--trigger cho bảng teacher*

GO

CREATE TRIGGER trg\_Teacher\_Insert

ON Teacher

INSTEAD OF INSERT

AS

BEGIN

    DECLARE @TeacherDescription NVARCHAR(255);

    SELECT @TeacherDescription = TeacherDescription

    FROM inserted;

*-- Validate TeacherDescription*

    IF @TeacherDescription IS NULL

    BEGIN

        RAISERROR ('TeacherDescription cannot be null', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- nếu không có lỗi thì insert dữ liệu*

    INSERT INTO Teacher (TeacherID, TeacherDescription)

    SELECT TeacherID, TeacherDescription

    FROM inserted;

END;

*-- Trigger khi insert dữ liệu vào bảng Admin*

GO

CREATE TRIGGER trg\_Admin\_Insert

ON Admin

INSTEAD OF INSERT

AS

BEGIN

    DECLARE @AdminID INT;

    SELECT @AdminID = AdminID

    FROM inserted;

*-- Validate AdminID exists in Users table*

    IF NOT EXISTS (SELECT 1 FROM Users WHERE UsernameID = @AdminID)

    BEGIN

        RAISERROR ('AdminID does not exist in Users table', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- If validation passes, insert the data*

    INSERT INTO Admin (AdminID)

    SELECT AdminID

    FROM inserted;

END;

*-- Trigger khi insert dữ liệu vào bảng Student*

GO

CREATE TRIGGER trg\_Student\_Insert

ON Student

INSTEAD OF INSERT

AS

BEGIN

    DECLARE @OptionName NVARCHAR(255), @QuestionID INT, @StudentID INT;

    SELECT @OptionName = OptionName, @QuestionID = QuestionID, @StudentID = StudentID

    FROM inserted;

*-- Validate OptionName*

    IF @OptionName IS NOT NULL AND NOT EXISTS (SELECT 1 FROM Options WHERE OptionName = @OptionName)

    BEGIN

        RAISERROR ('OptionName does not exist in Options table', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- Validate QuestionID*

    IF @QuestionID IS NOT NULL AND NOT EXISTS (SELECT 1 FROM Question WHERE QuestionID = @QuestionID)

    BEGIN

        RAISERROR ('QuestionID does not exist in Question table', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- If validation passes, insert the data*

    INSERT INTO Student (StudentID, OptionName, QuestionID)

    SELECT StudentID, OptionName, QuestionID

    FROM inserted;

END;

GO

*-- Trigger khi insert dữ liệu vào bảng Category*

GO

CREATE TRIGGER trg\_Category\_Insert

ON Category

INSTEAD OF INSERT

AS

BEGIN

    DECLARE @CategoryName NVARCHAR(255), @CategoryDescription NVARCHAR(255);

    SELECT @CategoryName = CategoryName, @CategoryDescription = CategoryDescription

    FROM inserted;

*-- Validate CategoryName*

    IF @CategoryName IS NULL

    BEGIN

        RAISERROR ('CategoryName cannot be null', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- Validate CategoryDescription*

    IF @CategoryDescription IS NULL

    BEGIN

        RAISERROR ('CategoryDescription cannot be null', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- If validation passes, insert the data*

    INSERT INTO Category (CategoryName, CategoryDescription)

    SELECT CategoryName, CategoryDescription

    FROM inserted;

END;

*-- Trigger khi insert dữ liệu vào bảng Course*

GO

CREATE TRIGGER trg\_Course\_Insert

ON Course

INSTEAD OF INSERT

AS

BEGIN

    DECLARE @CourseName NVARCHAR(255), @CourseStatus NVARCHAR(255), @CourseDescription NVARCHAR(255),

            @CoursePrice INT, @CourseImage NVARCHAR(255), @CourseStartDate DATE, @CourseEndDate DATE,

            @CategoryID INT;

    SELECT @CourseName = CourseName, @CourseStatus = CourseStatus, @CourseDescription = CourseDescription,

            @CoursePrice = CoursePrice, @CourseImage = CourseImage, @CourseStartDate = CourseStartDate,

            @CourseEndDate = CourseEndDate, @CategoryID = CategoryID

    FROM inserted;

*-- Validate CourseName*

    IF @CourseName IS NULL

    BEGIN

        RAISERROR ('CourseName cannot be null', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- Validate CourseStatus*

    IF @CourseStatus IS NULL

    BEGIN

        RAISERROR ('CourseStatus cannot be null', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- Validate CourseDescription*

    IF @CourseDescription IS NULL

    BEGIN

        RAISERROR ('CourseDescription cannot be null', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- Validate CoursePrice*

    IF @CoursePrice IS NULL

    BEGIN

        RAISERROR ('CoursePrice cannot be null', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- Validate CourseImage*

    IF @CourseImage IS NULL

    BEGIN

        RAISERROR ('CourseImage cannot be null', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- Validate CourseStartDate*

    IF @CourseStartDate IS NULL

    BEGIN

        RAISERROR ('CourseStartDate cannot be null', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- Validate CourseEndDate*

    IF @CourseEndDate IS NULL

    BEGIN

        RAISERROR ('CourseEndDate cannot be null', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- Validate CategoryID*

    IF NOT EXISTS (SELECT 1 FROM Category WHERE CategoryID = @CategoryID)

    BEGIN

        RAISERROR ('CategoryID does not exist in Category table', 16, 1);

        ROLLBACK TRANSACTION;

        RETURN;

    END

*-- If validation passes, insert the data*

    INSERT INTO Course (CourseName, CourseStatus, CourseDescription, CoursePrice, CourseImage, CourseStartDate, CourseEndDate, CategoryID)

    SELECT CourseName, CourseStatus, CourseDescription, CoursePrice, CourseImage, CourseStartDate, CourseEndDate, CategoryID

    FROM inserted;

END;