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

**HK231 - Lớp: L0x**

**Nhóm**: <tên nhóm>

**Danh sách thành viên**:

|  |  |  |
| --- | --- | --- |
| Stt | Họ tên | MSSV |
| 1 |  |  |
| 2 |  |  |

**Chủ đề**: <chủ đề mà nhóm chọn>

**1. Thiết kế ERD cập nhật mới nhất**

A diagram of a network

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: <course>**

1. Thủ tục INSERT: **InsertCourse**

Thủ tục này được sử dụng để thêm một khóa học mới vào bảng Course trong cơ sở dữ liệu.

**Usecase:** Sử dụng trong các trường hợp cần thêm một khóa học mới vào hệ thống quản lý khóa học.

**Thao tác chính:**

1. Thêm dữ liệu vào bảng Course với các giá trị được truyền vào qua các biến @p\_Course\_ID, @p\_Course\_Name, @p\_Course\_Status, @p\_Course\_Description, @p\_Course\_Price, @p\_Course\_Image, @p\_Course\_Start\_Date, @p\_Course\_End\_Date, @p\_Course\_Categories\_ID.

2. Lấy giá trị Course\_ID vừa được tạo bằng cách sử dụng SCOPE\_IDENTITY().

3. Trả về thông báo xác nhận thành công với giá trị Course\_ID vừa được tạo.

**Input:**

\* @p\_Course\_ID: ID của khóa học.

\* @p\_Course\_Name: Tên của khóa học.

\* @p\_Course\_Status: Trạng thái của khóa học.

\* @p\_Course\_Description: Mô tả khóa học.

\* @p\_Course\_Price: Giá của khóa học.

\* @p\_Course\_Image: Đường dẫn hình ảnh của khóa học.

\* @p\_Course\_Start\_Date: Ngày bắt đầu của khóa học.

\* @p\_Course\_End\_Date: Ngày kết thúc của khóa học.

\* @p\_Course\_Categories\_ID: ID của danh mục khóa học.

**Output:**

Thông báo xác nhận thành công với giá trị Course\_ID vừa được tạo, ví dụ: "12345 Khóa học "Lập trình web" đã được thêm thành công với Course\_ID = 67890!".

1. Thủ tục UPDATE: UpdateCourse

Đây là một thủ tục lưu trữ (stored procedure) được sử dụng để cập nhật thông tin của một khóa học trong cơ sở dữ liệu. Nó thực hiện việc kiểm tra tính hợp lệ của dữ liệu đầu vào và cập nhật thông tin khóa học vào bảng Course nếu dữ liệu hợp lệ.

**Usecase**: Sử dụng trong các trường hợp cần cập nhật thông tin của một khóa học đã tồn tại trong hệ thống quản lý khóa học.

**Thao tác chính**:

1. Kiểm tra xem khóa học có tồn tại không.
2. Kiểm tra xem danh mục có tồn tại không.
3. Cập nhật thông tin khóa học nếu cả khóa học và danh mục đều tồn tại, bao gồm:
   * Cập nhật tên khóa học (Course\_Name).
   * Cập nhật trạng thái khóa học (Course\_Status).
   * Cập nhật mô tả khóa học (Course\_Description).
   * Cập nhật giá khóa học (Course\_Price).
   * Cập nhật hình ảnh khóa học (Course\_Image).
   * Cập nhật ngày bắt đầu của khóa học (Course\_Start\_Date).
   * Cập nhật ngày kết thúc của khóa học (Course\_End\_Date).
   * Cập nhật ID danh mục của khóa học (Course\_Categories\_ID).
4. Trả về thông báo xác nhận thành công.

**Input**:

* @p\_Course\_ID: ID của khóa học cần cập nhật.
* @p\_Course\_Name: Tên của khóa học.
* @p\_Course\_Status: Trạng thái của khóa học.
* @p\_Course\_Description: Mô tả khóa học.
* @p\_Course\_Price: Giá của khóa học.
* @p\_Course\_Image: Đường dẫn hình ảnh của khóa học.
* @p\_Course\_Start\_Date: Ngày bắt đầu của khóa học.
* @p\_Course\_End\_Date: Ngày kết thúc của khóa học.
* @p\_Course\_Categories\_ID: ID của danh mục khóa học.

**Output**:

* Thông báo xác nhận thành công, ví dụ: "Khóa học "Lập trình C++ cơ bản" đã được cập nhật thành công!".

**Lỗi**: Hàm sẽ trả về lỗi "Khóa học không tồn tại!" hoặc "Danh mục không tồn tại!" nếu khóa học hoặc danh mục không được tìm thấy trong cơ sở dữ liệu.

**Ví dụ**: Dòng lệnh EXEC UpdateCourse 501, 'Lập trình C++ cơ bản', 'Open', 'Khóa học giúp bạn nắm vững các kiến thức cơ bản về C++.', 500, 'c++\_basic.jpg', '2024-01-01', '2024-06-30', 1; sẽ cập nhật thông tin của khóa học có ID là 501 với các giá trị mới được cung cấp.

1. **Thủ tục DELETE: DeleteCourse**

Đây là một thủ tục được dùng để xóa một khóa học khỏi bảng Course trong cơ sở dữ liệu. Nó thực hiện việc kiểm tra xem khóa học có tồn tại không trước khi xóa.

**Usecase**: Sử dụng trong các trường hợp cần xóa một khóa học không còn được sử dụng khỏi hệ thống quản lý khóa học.

**Thao tác chính**:

1. Kiểm tra xem khóa học có tồn tại không.
2. Xóa khóa học khỏi bảng Course nếu khóa học tồn tại.
3. Trả về thông báo xác nhận thành công.

**Input**:

* @p\_Course\_ID: ID của khóa học cần xóa.

**Output**:

* Thông báo xác nhận thành công, ví dụ: "Khóa học có Course\_ID = 501 đã được xóa thành công!".

**Lỗi**: Hàm sẽ trả về lỗi "Khóa học không tồn tại!" nếu khóa học không được tìm thấy trong cơ sở dữ liệu.

**Ví dụ**: Dòng lệnh EXEC DeleteCourse 501; sẽ xóa khóa học có ID là 501 khỏi bảng Course.

**2.2 TRIGGER**

1. Trigger 1

**Mô tả Trigger:**

* **Tên trigger:** tr\_CheckCourseDates
* **Chức năng:** Trigger này được thiết kế để đảm bảo thời gian bắt đầu của khóa học luôn nhỏ hơn thời gian kết thúc của khóa học.
* **Sử dụng:** Trigger này được sử dụng trong tình huống thêm khóa học mới (insert) vào bảng Course.
* **Cách kích hoạt:** Trigger được kích hoạt (fire) *ngay trước* khi câu lệnh INSERT vào bảng Course được thực thi.
* **Thao tác chính:**
  + Kiểm tra xem có bất kỳ khóa học nào trong dữ liệu được thêm vào (inserted) có thời gian bắt đầu Course\_Start\_Date lớn hơn hoặc bằng thời gian kết thúc Course\_End\_Date hay không.
  + Nếu tìm thấy trường hợp này, trigger sẽ raise lỗi RAISERROR với thông báo "Thời gian bắt đầu phải nhỏ hơn thời gian kết thúc của khóa học.".
  + Nếu dữ liệu được thêm vào hợp lệ (thời gian bắt đầu nhỏ hơn thời gian kết thúc), trigger sẽ thực hiện việc thêm dữ liệu vào bảng Course bằng cách copy dữ liệu từ biến inserted vào bảng.
  + Cuối cùng, trigger sẽ thông báo "Khóa học đã được thêm thành công!" nếu thao tác insert thành công.

1. Trigger 2

**Mô tả Trigger:**

* **Tên trigger:** tr\_CheckCouponValue
* **Chức năng:** Trigger này được thiết kế để kiểm tra xem thông tin của phiếu giảm giá khi được thêm vào có hợp lệ hay không.
* **Sử dụng:** Trigger này được sử dụng trong tình huống thêm phiếu giảm giá mới (insert) vào bảng Coupon.
* **Cách kích hoạt:** Trigger được kích hoạt (fire) *ngay trước* khi câu lệnh INSERT vào bảng Coupon được thực thi.
* **Thao tác chính:**
  + Kiểm tra xem có bất kỳ phiếu giảm giá nào trong dữ liệu được thêm vào (inserted) có giá trị giảm giá Coupon\_Value âm hay không. Nếu có, trigger sẽ báo lỗi "Giá trị giảm giá phải là số không âm.".
  + Kiểm tra xem có bất kỳ phiếu giảm giá nào trong dữ liệu được thêm vào (inserted) có loại giảm giá Coupon\_Type không phải là "Percent" hoặc "Fixed" hay không. Nếu có, trigger sẽ báo lỗi "Loại giảm giá phải là "Percent" hoặc "Fixed".".
  + Kiểm tra xem có bất kỳ phiếu giảm giá nào trong dữ liệu được thêm vào (inserted) có ngày bắt đầu Coupon\_Start\_Date lớn hơn hoặc bằng ngày kết thúc Coupon\_Expiry\_Date hay không. Nếu có, trigger sẽ báo lỗi "Ngày bắt đầu phải nhỏ hơn ngày kết thúc.".
  + Kiểm tra xem có bất kỳ phiếu giảm giá nào trong dữ liệu được thêm vào (inserted) có giá trị giảm giá tối đa Coupon\_Max\_Discount âm hay không. Nếu có, trigger sẽ báo lỗi "Giá trị giảm giá tối đa phải là số không âm.".
  + Nếu tất cả dữ liệu hợp lệ, trigger sẽ thêm phiếu giảm giá mới vào bảng Coupon bằng cách copy dữ liệu từ biến inserted vào bảng.
  + Cuối cùng, trigger sẽ thông báo "Phiếu giảm giá đã được thêm thành công!" nếu thao tác insert thành công.

**2.3 THỦ TỤC**

1. Thủ tục 1

Mô tả thủ tục: <*thủ tục này dùng để làm gì, sử dụng trong trường hợp – usecase – nào, thủ tục này bao gồm những thao tác chính nào*>

Input: <*các thông số nhập, nếu có*>

Output: <*kết quả trả về, nếu có*>

1. Thủ tục 2

*Trình bày tương tự thủ tục 1*

**2.4 HÀM**

1. Hàm 1

Mô tả hàm: <*hàm này dùng để làm gì, sử dụng trong trường hợp – usecase – nào, bao gồm những thao tác chính nào*>

Input: <*các thông số nhập, nếu có*>

Output: <*kết quả trả về, nếu có*>

1. Hàm 2

*Trình bày tương tự hàm 1*