**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT THÀNH PHỐ HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BỘ MÔN CÔNG NGHỆ PHẦN MỀM**

A blue logo with a ball and a black background

Description automatically generated-----**🙞🕮🙜**-----

**ĐỀ TÀI CUỐI KỲ MÔN HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU**

**TÊN ĐỀ TÀI:**

**QUẢN LÝ ĐIỂM SINH VIÊN KHOA CÔNG NGHỆ THÔNG TIN**

**GVHD:** *T.S. Nguyễn Thành Sơn*

***Lớp HP:*** *DBMS Sáng thứ 6*

***Nhóm thực hiện:*** *Nhóm 3*

***Học kỳ:*** *I*

***Năm học:*** *2024 – 2025*

Thành phố Hồ Chí Minh, ngày … tháng … năm 2024

**DANH SÁCH CÁC THÀNH VIÊN**

**HỌC KỲ I, NĂM HỌC: 2024 – 2025**

**Nhóm 3**

**Đề tài: Quản lý điểm sinh viên khoa Công nghệ thông tin.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Họ và tên** | **MSSV** | **Tỉ lệ % hoàn thành** | **SĐT** |
| 1 | Nguyễn Hoàng Phúc | 22110400 | 100% |  |
| 2 | Hồ Lê Tấn Lợi | 22110370 | 100% |  |
| 3 | Lê Xuân Thịnh | 22110427 | 100% |  |
| 4 | Nguyễn Văn Luân | 22110373 | 100% |  |

**Ghi chú:**

Tỉ lệ % = 100%

Trưởng nhóm: Nguyễn Hoàng Phúc

**Nhận xét của giảng viên**

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

Ngày …… Tháng …… Năm ………

*Giáo viên chấm điểm*

**Mục lục**

[CHƯƠNG 1: ĐẶC TẢ BÀI TOÁN 1](#_Toc182460146)

[1.1. Đặc tả bài toán 1](#_Toc182460147)

[CHƯƠNG 2: THIẾT KẾ CƠ SỞ DỮ LIỆU 2](#_Toc182460148)

[2.1. Thiết kế cơ sở dữ liệu mức quan niệm 2](#_Toc182460149)

[2.2. Thiết kế cơ sở dữ liệu mức logic 2](#_Toc182460150)

[2.2.1. Các ràng buộc cần có 3](#_Toc182460151)

[2.3. Cài đặt cơ sở dữ liệu 5](#_Toc182460152)

[2.4. Các trigger 6](#_Toc182460153)

[2.4.1. Đảm bảo dữ liệu đúng khi thêm sinh viên 6](#_Toc182460154)

[2.4.2. Đảm bảo dữ liệu đúng khi thêm giảng viên 8](#_Toc182460155)

[2.4.3. Đảm bảo dữ liệu đúng khi thêm môn học 9](#_Toc182460156)

[2.4.4. Đảm bảo dữ liệu đúng khi thêm lớp học 10](#_Toc182460157)

[2.4.5. Đảm bảo dữ liệu đúng khi thêm đăng ký 11](#_Toc182460158)

[2.4.6. Số lượng sinh viên không vượt quá sức chứa của phòng sau khi đăng ký 12](#_Toc182460159)

[2.4.7. Gửi mail cho sinh viên bị rớt môn 12](#_Toc182460160)

[2.5. Các view 13](#_Toc182460161)

[2.5.1. Danh sách sinh viên 13](#_Toc182460162)

[2.5.2. Danh sách giảng viên 14](#_Toc182460163)

[2.5.3. Danh sách môn học 15](#_Toc182460164)

[2.5.4. Danh sách lớp học 17](#_Toc182460165)

[2.5.5. Những lớp học giảng viên phụ trách 18](#_Toc182460166)

[2.5.6. Các lớp học còn chỗ trống 20](#_Toc182460167)

[2.5.7. Điểm trung bình của các sinh viên 21](#_Toc182460168)

[2.5.8. Điểm trung bình tích lũy của các sinh viên 23](#_Toc182460169)

[2.5.9. Bảng xếp hạng các sinh viên bằng điểm trung bình tích lũy 24](#_Toc182460170)

[CHƯƠNG 3: CÀI ĐẶT CHỨC NĂNG 26](#_Toc182460171)

[3.1. Kết nối cơ sở dữ liệu 26](#_Toc182460172)

[3.2. Thủ tục 27](#_Toc182460173)

[3.2.1. Thêm sinh viên 27](#_Toc182460174)

[3.2.2. Thêm giảng viên 28](#_Toc182460175)

[3.2.3. Thêm môn học 29](#_Toc182460176)

[3.2.4. Thêm lớp học 31](#_Toc182460177)

[3.2.5. Xóa sinh viên 33](#_Toc182460178)

[3.2.6. Xóa giảng viên 34](#_Toc182460179)

[3.2.7. Xóa môn học 35](#_Toc182460180)

[3.2.8. Xóa lớp học 36](#_Toc182460181)

[3.2.9. Sửa thông tin sinh viên 37](#_Toc182460182)

[3.2.10. Sửa thông tin giảng viên 39](#_Toc182460183)

[3.2.11. Sửa thông tin môn học 40](#_Toc182460184)

[3.2.12. Sửa lớp học 42](#_Toc182460185)

[3.2.13. Đăng ký sinh viên vào lớp học 44](#_Toc182460186)

[3.2.14. Cập nhật điểm sinh viên 46](#_Toc182460187)

[3.2.15. Xem kết quả học của các sinh viên tại một lớp 48](#_Toc182460188)

[3.2.16. Xem thời khóa biểu một sinh viên 50](#_Toc182460189)

[3.2.17. Giảng viên dạy môn học với nhiều sinh viên nhất 52](#_Toc182460190)

[3.3. Hàm 53](#_Toc182460191)

[3.3.1. Kiểm tra một sinh viên có qua môn 53](#_Toc182460192)

[3.3.2. Xem học lực một sinh viên 55](#_Toc182460193)

[3.3.3. Tính số tín chỉ đã hoàn thành của một sinh viên (thieu comment) 57](#_Toc182460194)

[3.3.4. Đếm số lượng sinh viên của một lớp học 58](#_Toc182460195)

[3.3.5. Đếm số lớp học một giảng viên đang phụ trách 59](#_Toc182460196)

[3.3.6. Tính phần trăm qua môn trong một lớp học 61](#_Toc182460197)

[3.3.7. Lấy danh sách sinh viên với điểm trong một lớp học 62](#_Toc182460198)

[3.3.8. Xem bảng điểm của một sinh viên 64](#_Toc182460199)

[3.3.9. Tính điểm trung bình một sinh viên 66](#_Toc182460200)

[3.3.10. Tính điểm trung bình tích lũy của một sinh viên 68](#_Toc182460201)

[3.3.11. Đếm số lượng lớp một sinh viên đăng ký 69](#_Toc182460202)

[3.4. Transactions 71](#_Toc182460203)

[3.4.1. Chuyển lớp học cho sinh viên 71](#_Toc182460204)

[CHƯƠNG 4: TẠO USER VÀ PHÂN QUYỀN 73](#_Toc182460205)

[4.1. Các vai trò của ứng dụng 73](#_Toc182460206)

[4.2. Tạo login trên server 73](#_Toc182460207)

[4.3. Tạo user trong cơ sở dữ liệu 74](#_Toc182460208)

[4.4. Tạo role 74](#_Toc182460209)

[4.5. Thêm quyền cho các Role 74](#_Toc182460210)

[4.6. Thêm user vào role 75](#_Toc182460211)

[4.7. Cấp quyền thực thi hàm hệ thống 75](#_Toc182460212)

[CHƯƠNG 5: CÀI ĐẶT GIAO DIỆN 75](#_Toc182460213)

[5.1. Trang đăng nhập 75](#_Toc182460214)

[5.2. Trang chính 76](#_Toc182460215)

[5.3. Trang quản lý sinh viên 77](#_Toc182460216)

[5.4. Trang quản lý giảng viên 77](#_Toc182460217)

[5.5. Trang quản lý môn học 78](#_Toc182460218)

[5.6. Trang quản lý lớp học 78](#_Toc182460219)

# ĐẶC TẢ BÀI TOÁN

## Đặc tả bài toán

Điểm số của các sinh viên trong khoa Công nghệ thông tin của một trường học sẽ cần được quản lý bằng một hệ thống quản lý điểm sinh viên. Một sinh viên sẽ có mã sinh viên, họ và tên, email, số điện thoại, và quê quán. Một môn học sẽ có mã môn học, tên môn học, và số tín chỉ. Một giảng viên sẽ có mã giảng viên, họ và tên, email, và số điện thoại. Một lớp học sẽ có mã lớp học, thứ, tiết bắt đầu và tiết kết thúc. Ví dụ, nếu một lớp học diễn ra vào thứ sáu và học 3 tiết từ tiết 2 đến tiết 4 thì thông tin của lớp học đó sẽ là thứ sáu, tiết bắt đầu là 2, và tiết kết thúc là 4. Có 7 thứ từ chủ nhật đến thứ hai và trong một ngày sẽ có 12 tiết từ tiết 1 đến tiết 12. Một lớp học bắt buộc dạy duy nhất một môn học, tuy nhiên một môn học có thể được dạy ở nhiều lớp khác nhau. Một lớp học phải được phụ trách bởi duy nhất một giảng viên, trong khi một giảng viên có thể phụ trách nhiều lớp học khác nhau hoặc không phụ trách lớp nào. Phòng học là nơi lớp học được tổ chức, mỗi phòng học sẽ có mã phòng học và sức chứa của nó. Sức chứa của một phòng học không được bé hơn 10 hoặc lớn hơn 100 sinh viên. Một lớp học sẽ chỉ được tổ chức tại duy nhất một phòng học, trong khi một phòng học có thể tổ chức nhiều lớp học khác nhau mặc dù có thể có phòng bị trống; đồng thời cũng cần đảm bảo không có sự trùng lặp về thứ và khoảng thời gian từ tiết bắt đầu đến tiết kết thúc đối với các lớp học được tổ chức chung phòng. Ngoài ra, sinh viên sẽ cần đăng ký lớp học. Một sinh viên có thể không đăng ký hoặc đăng ký nhiều lớp học khác nhau miễn là không bị trùng môn học, trong khi một lớp học phải có ít nhất một hoặc nhiều sinh viên khác nhau cùng đăng ký. Sinh viên sẽ có điểm cho môn học được dạy tại lớp mà sinh viên đã đăng ký. Điểm của một môn học gồm 2 thành phần là điểm quá trình và điểm cuối kỳ. Điểm quá trình hoặc điểm cuối kỳ có thể trống nếu như chưa được nhập, điểm phải được nhập đầy đủ mới có thể đánh giá và xếp loại. Môn học của sinh viên được đánh giá là qua môn nếu như trung bình cộng của điểm quá trình và điểm cuối kỳ của sinh viên đó lớn hơn hoặc bằng 5.0 và điểm cuối kỳ không dưới 3.0. Trung bình cộng của điểm quá trình và điểm cuối kỳ của một môn được gọi là điểm trung bình môn của môn đó. Trung bình cộng của các điểm trung bình môn của một sinh viên được gọi là điểm trung bình, điểm này sẽ được dùng để xếp loại học lực của sinh viên đó. Cụ thể, một điểm số nếu như muốn xếp loại sẽ được xếp loại giỏi nếu không dưới 8.0, xếp loại khá nếu bé hơn 8.0 và không dưới 6.5, loại trung bình nếu bé hơn 6.5 và không dưới 5.0, và loại yếu cho trường hợp bé hơn 5.0 hoặc rớt môn. Điểm trung bình tích lũy của sinh viên là điểm trung bình của sinh viên nhưng không tính các môn học bị rớt.

Hệ thống quản lý điểm sinh viên sẽ cung cấp các chức năng cơ bản như thêm, xóa, cập nhật thông tin sinh viên, giảng viên, môn học, và lớp học. Đăng ký sinh viên vào lớp học và cập nhật điểm cho các sinh viên của một lớp. Phụ trách giảng viên dạy vào dạy lớp học. Xem kết quả học tập của các sinh viên với môn học được dạy tại lớp học. Kiểm tra một sinh viên có qua môn hay không. Xem học lực của một sinh viên. Tính số tín chỉ đã hoàn thành của một sinh viên. Đếm số lượng sinh viên của một lớp. Tính tỷ lệ phần trăm qua môn trong một lớp học. Đếm số lớp học mà một giảng viên đang phụ trách. Lấy danh sách các sinh viên trong một lớp học. Xem bảng điểm của một sinh viên, bảng này gồm các điểm và xếp loại tương ứng của các môn đã học. Tính điểm trung bình và điểm trung bình tích lũy của một sinh viên.

Giao diện hệ thống sẽ được thiết kế với trang tổng quan hiển thị bảng tóm tắt về số lượng sinh viên, môn học, giảng viên, lớp học, và phòng học. Cùng các phím điều hướng đến các trang chức năng cơ bản thêm, xóa, cập nhật cho sinh viên, môn học, giảng viên, và phòng học. Trong chức năng quản lý sinh viên sẽ liệt kê danh sách sinh viên, cùng các nút thêm, xóa sinh viên thuận tiện. Các nút được nhấn sẽ hiện ra các trang nhập thông tin hoặc cảnh báo xóa dữ liệu tương ứng, và cập nhật thông tin của sinh viên; tương tự với trang quản lý môn học, giảng viên, hay phòng. Tiếp theo là trang quản lý điểm số sẽ được bố trí theo lớp học cùng các thông tin cơ bản như mã lớp, môn học, số lượng sinh viên. Các nút chức năng chung liên quan đến điểm số như như xếp loại điểm cho các lớp, xếp loại môn theo điểm của sinh viên, tổng kết kết quả giảng dạy của giảng viên,… cũng được trình bày. Người dùng nhấn vào lớp học để xem thông tin thêm và truy cập vào các chức năng tính toán và thống kê điểm số cho lớp đó. Người dùng cũng có thể xem thông tin của giảng viên phụ trách, cùng danh sách sinh viên của lớp học, điểm trung bình của các sinh viên của lớp đó. Có nút xem chi tiết thông tin để có thể xem chi tiết hơn về thông tin cá nhân và điểm mỗi sinh viên, bao gồm điểm quá trình, điểm cuối kỳ, và điểm trung bình các môn khác.

# THIẾT KẾ CƠ SỞ DỮ LIỆU

## Thiết kế cơ sở dữ liệu mức quan niệm

Từ đặc tả của bài toán, ta được sơ đồ thực thể - quan hệ ERD.

**A diagram of a diagram

Description automatically generated with medium confidence**

## Thiết kế cơ sở dữ liệu mức logic

Từ mô hình thực thể - quan hệ ERD, ta được lược đồ sơ sở dữ liệu quan hệ.

SinhVien(MaSinhVien, HoVaTen, Email, SoDienThoai, QueQuan)

MonHoc(MaMonHoc, TenMonHoc, SoTinChi)

GiangVien(MaGiangVien, HoVaTen, Email, SoDienThoai)

PhongHoc(MaPhongHoc, SucChua)

LopHoc(MaLopHoc, Thu, TietBatDau, TietKetThuc, MaPhongHoc, MaGiangVien, MaMonHoc)

DangKy(MaSinhVien, MaLopHoc, DiemQuaTrinh, DiemCuoiKy)

### Các ràng buộc cần có

|  |  |  |  |
| --- | --- | --- | --- |
| **Bảng** | **Thuộc tính** | **Kiểu dữ liệu** | **Ràng buộc** |
| SinhVien | MaSinhVien | INT | PK\_SinhVien PRIMARY KEY |
|  | HoVaTen | NVARCHAR(100) | NOT NULL |
|  | Emai | VARCHAR(100) | UNIQUE NOT NULL |
|  | SoDienThoai | VARCHAR(20) | UNIQUE NOT NULL |
|  | QueQuan | NVARCHAR(100) | NOT NULL |
| MonHoc | MaMonHoc | INT | PK\_MonHoc PRIMARY KEY |
|  | TenMonHoc | NVARCHAR(100) | NOT NULL |
|  | SoTinChi | TINYINT | NOT NULL CHK\_SoTinChiHopLe CHECK (SoTinChi>=1 AND SoTinChi <= 5) |
| GiangVien | MaGiangVien | INT | PK\_GiangVien PRIMARY KEY |
|  | HoVaTen | NVARCHAR(100) | NOT NULL |
|  | Email | VARCHAR(100) | UNIQUE NOT NULL |
|  | SoDienThoai | VARCHAR(20) | UNIQUE NOT NULL |
| PhongHoc | MaPhongHoc | INT | PK\_PhongHoc PRIMARY KEY |
|  | SucChua | TINYINT | NOT NULL CHK\_SucChuaHopLe CHECK (SucChua >= 10 AND SucChua <= 100) |
| LopHoc | MaLopHoc | INT | PK\_LopHoc PRIMARY KEY |
|  | Thu | TINYINT | NOT NULL CHK\_ThuHopLe CHECK(Thu >= 1 AND Thu <= 7)  Ghi chú: 1 là chủ nhật, 2 là thứ hai, cho đến 7 là thứ bảy |
|  | TietBatDau | TINYINT | NOT NULL CHK\_TietHopLe CHECK(TietBatDau >= 1 AND TietBatDau <= 12 AND TietKetThuc >= 1 AND TietKetThuc <= 12 AND TietBatDau <= TietKetThuc) |
|  | TietKetThuc | TINYINT | NOT NULL |
|  | MaPhongHoc | INT | FK\_LopHoc\_PhongHoc FOREIGN KEY REFERENCES PhongHoc(MaPhongHoc) ON UPDATE CASCADE ON DELETE CASCADE |
|  | MaGiangVien | INT | FK\_LopHoc\_GiangVien FOREIGN KEY REFERENCES GiangVien(MaGiangVien) ON UPDATE CASCADE ON DELETE CASCADE |
|  | MaMonHoc | INT | FK\_LopHoc\_MonHoc FOREIGN KEY REFERENCES MonHoc(MaMonHoc) ON UPDATE CASCADE |
| DangKy | MaSinhVien | INT | NOT NULL FK\_DangKy\_SinhVien FOREIGN KEY (MaSinhVien) REFERENCES SinhVien(MaSinhVien) ON UPDATE CASCADE |
|  | MaLopHoc | INT | NOT NULL FK\_DangKy\_MonHoc FOREIGN KEY (MaMonHoc) REFERENCES MonHoc(MaMonHoc) ON UPDATE CASCADE |
|  | DiemQuaTrinh | DECIMAL(4, 2) | CHK\_DiemQuaTrinhHopLe CHECK (DiemQuaTrinh >= 0 AND DiemQuaTrinh <= 10) |
|  | DiemCuoiKy | DECIMAL(4, 2) | CHK\_DiemCuoiKyHopLe CHECK (DiemCuoiKy >= 0 AND DiemCuoiKy <= 10) |
|  |  |  | PK\_DangKy PRIMARY KEY (MaSinhVien, MaLopHoc) |

## Cài đặt cơ sở dữ liệu

Bảng sinh viên:

CREATE TABLE SinhVien (

MaSinhVien INT,

HoVaTen NVARCHAR(100) NOT NULL,

Email VARCHAR(100) NOT NULL UNIQUE,

SoDienThoai VARCHAR(20) NOT NULL UNIQUE,

QueQuan NVARCHAR(100) NOT NULL,

CONSTRAINT PK\_SinhVien PRIMARY KEY (MaSinhVien)

);

Bảng môn học:

CREATE TABLE MonHoc (

MaMonHoc INT,

TenMonHoc NVARCHAR(100) NOT NULL,

SoTinChi TINYINT NOT NULL,

CONSTRAINT PK\_MonHoc PRIMARY KEY (MaMonHoc),

CONSTRAINT CHK\_SoTinChiHopLe CHECK (SoTinChi >= 1 AND SoTinChi <= 5)

);

Bảng giảng viên:

CREATE TABLE GiangVien (

MaGiangVien INT,

HoVaTen NVARCHAR(100) NOT NULL,

Email VARCHAR(100) NOT NULL UNIQUE,

SoDienThoai VARCHAR(20) NOT NULL UNIQUE,

CONSTRAINT PK\_GiangVien PRIMARY KEY (MaGiangVien)

);

Bảng phòng học:

CREATE TABLE PhongHoc (

MaPhongHoc INT,

SucChua TINYINT NOT NULL,

CONSTRAINT PK\_PhongHoc PRIMARY KEY (MaPhongHoc),

CONSTRAINT CHK\_SucChuaHopLe CHECK (SucChua >= 10 AND SucChua <= 100)

);

Bảng lớp học:

CREATE TABLE LopHoc (

MaLopHoc INT,

Thu TINYINT NOT NULL,

TietBatDau TINYINT NOT NULL,

TietKetThuc TINYINT NOT NULL,

MaPhongHoc INT,

MaGiangVien INT,

MaMonHoc INT,

CONSTRAINT PK\_LopHoc PRIMARY KEY (MaLopHoc),

CONSTRAINT CHK\_ThuHopLe CHECK (Thu >= 1 AND Thu <= 7),

CONSTRAINT CHK\_TietHopLe CHECK (TietBatDau >= 1 AND TietBatDau <= 12 AND TietKetThuc >= 1 AND TietKetThuc <= 12 AND TietBatDau <= TietKetThuc),

CONSTRAINT FK\_LopHoc\_PhongHoc FOREIGN KEY (MaPhongHoc) REFERENCES PhongHoc(MaPhongHoc) ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT FK\_LopHoc\_GiangVien FOREIGN KEY (MaGiangVien) REFERENCES GiangVien(MaGiangVien) ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT FK\_LopHoc\_MonHoc FOREIGN KEY (MaMonHoc) REFERENCES MonHoc(MaMonHoc) ON UPDATE CASCADE

);

Bảng đăng ký:

CREATE TABLE DangKy (

MaSinhVien INT NOT NULL,

MaLopHoc INT NOT NULL,

DiemQuaTrinh DECIMAL(4, 2),

DiemCuoiKy DECIMAL(4, 2),

CONSTRAINT PK\_DangKy PRIMARY KEY (MaSinhVien, MaLopHoc),

CONSTRAINT CHK\_DiemQuaTrinhHopLe CHECK (DiemQuaTrinh >= 0 AND DiemQuaTrinh <= 10),

CONSTRAINT CHK\_DiemCuoiKyHopLe CHECK (DiemCuoiKy >= 0 AND DiemCuoiKy <= 10),

CONSTRAINT FK\_DangKy\_SinhVien FOREIGN KEY (MaSinhVien) REFERENCES SinhVien(MaSinhVien) ON UPDATE CASCADE,

CONSTRAINT FK\_DangKy\_LopHoc FOREIGN KEY (MaLopHoc) REFERENCES LopHoc(MaLopHoc) ON UPDATE CASCADE

);

Sơ đồ quan hệ:

A screenshot of a computer

Description automatically generated

## Các trigger

### Đảm bảo dữ liệu đúng khi thêm sinh viên

CREATE TRIGGER trg\_ThayViThemSinhVien ON SinhVien

INSTEAD OF INSERT

AS

BEGIN

DECLARE

@MaSinhVien INT,

@HoVaTen NVARCHAR(100),

@Email VARCHAR(100),

@SoDienThoai VARCHAR(20),

@QueQuan NVARCHAR(100);

-- Lấy dữ liệu từ bảng inserted

SELECT

@MaSinhVien = i.MaSinhVien,

@HoVaTen = i.HoVaTen,

@Email = i.Email,

@SoDienThoai = i.SoDienThoai,

@QueQuan = i.QueQuan

FROM inserted i;

-- Kiểm tra nếu mã sinh viên đã tồn tại

IF EXISTS (

SELECT 1

FROM SinhVien sv

WHERE sv.MaSinhVien = @MaSinhVien

)

BEGIN

RAISERROR('Mã sinh viên %d đã tồn tại.', 16, 1, @MaSinhVien);

RETURN;

END;

-- Kiểm tra điều kiện rỗng và định dạng

IF @HoVaTen = N''

BEGIN

RAISERROR('Họ và tên không được để trống.', 16, 1);

RETURN;

END;

IF @SoDienThoai = ''

BEGIN

RAISERROR('Số điện thoại không được để trống.', 16, 1);

RETURN;

END;

IF @QueQuan = N''

BEGIN

RAISERROR('Quê quán không được để trống.', 16, 1);

RETURN;

END;

IF @Email NOT LIKE '\_%@\_%.\_%'

BEGIN

RAISERROR('Email không đúng định dạng.', 16, 1);

RETURN;

END;

IF @SoDienThoai LIKE '%[^0-9]%'

BEGIN

RAISERROR('Số điện thoại không đúng định dạng.', 16, 1);

RETURN;

END;

-- Kiểm tra trùng lặp

IF EXISTS (SELECT 1 FROM SinhVien WHERE Email = @Email)

BEGIN

RAISERROR(N'Email %s đã được sử dụng.', 16, 1, @Email);

RETURN;

END

IF EXISTS (SELECT 1 FROM SinhVien WHERE SoDienThoai = @SoDienThoai)

BEGIN

RAISERROR(N'Số điện thoại %s đã được sử dụng.', 16, 1, @SoDienThoai);

RETURN;

END

-- Chèn dữ liệu hợp lệ vào bảng SinhVien

INSERT INTO SinhVien (MaSinhVien, HoVaTen, Email, SoDienThoai, QueQuan)

VALUES (@MaSinhVien, @HoVaTen, @Email, @SoDienThoai, @QueQuan);

END;

### Đảm bảo dữ liệu đúng khi thêm giảng viên

CREATE TRIGGER trg\_ThayViThemGiangVien

ON GiangVien

INSTEAD OF INSERT

AS

BEGIN

DECLARE

@MaGiangVien INT,

@HoVaTen NVARCHAR(100),

@Email VARCHAR(100),

@SoDienThoai VARCHAR(20);

-- Lấy dữ liệu từ bảng inserted

SELECT

@MaGiangVien = i.MaGiangVien,

@HoVaTen = i.HoVaTen,

@Email = i.Email,

@SoDienThoai = i.SoDienThoai

FROM inserted i;

-- Kiểm tra nếu mã giảng viên đã tồn tại

IF EXISTS (

SELECT 1

FROM GiangVien gv

WHERE gv.MaGiangVien = @MaGiangVien

)

BEGIN

RAISERROR('Mã giảng viên %d đã tồn tại.', 16, 1, @MaGiangVien);

RETURN;

END;

-- Kiểm tra điều kiện rỗng và định dạng

IF @HoVaTen = N''

BEGIN

RAISERROR('Họ và tên không được để trống.', 16, 1);

RETURN;

END;

IF @SoDienThoai = ''

BEGIN

RAISERROR('Số điện thoại không được để trống.', 16, 1);

RETURN;

END;

IF @Email NOT LIKE '\_%@\_%.\_%'

BEGIN

RAISERROR('Email không đúng định dạng.', 16, 1);

RETURN;

END;

IF @SoDienThoai LIKE '%[^0-9]%'

BEGIN

RAISERROR('Số điện thoại không đúng định dạng.', 16, 1);

RETURN;

END;

-- Kiểm tra trùng lặp

IF EXISTS (SELECT 1 FROM GiangVien WHERE Email = @Email)

BEGIN

RAISERROR(N'Email %s đã được sử dụng.', 16, 1, @Email);

RETURN;

END

IF EXISTS (SELECT 1 FROM GiangVien WHERE SoDienThoai = @SoDienThoai)

BEGIN

RAISERROR(N'Số điện thoại %s đã được sử dụng.', 16, 1, @SoDienThoai);

RETURN;

END

-- Chèn dữ liệu hợp lệ vào bảng GiangVien

INSERT INTO GiangVien (MaGiangVien, HoVaTen, Email, SoDienThoai)

VALUES (@MaGiangVien, @HoVaTen, @Email, @SoDienThoai);

END;

### Đảm bảo dữ liệu đúng khi thêm môn học

CREATE TRIGGER trg\_ThayViThemMonHoc

ON MonHoc

INSTEAD OF INSERT

AS

BEGIN

DECLARE

@MaMonHoc INT,

@TenMonHoc NVARCHAR(100),

@SoTinChi TINYINT;

-- Lấy dữ liệu từ bảng inserted

SELECT

@MaMonHoc = i.MaMonHoc,

@TenMonHoc = i.TenMonHoc,

@SoTinChi = i.SoTinChi

FROM inserted i;

-- Kiểm tra nếu mã môn học đã tồn tại

IF EXISTS (

SELECT 1

FROM MonHoc mh

WHERE mh.MaMonHoc = @MaMonHoc

)

BEGIN

RAISERROR('Mã môn học %d đã tồn tại.', 16, 1, @MaMonHoc);

RETURN;

END;

-- Kiểm tra điều kiện hợp lệ cho tên môn học và số tín chỉ

IF @TenMonHoc = N''

BEGIN

RAISERROR('Tên môn học không được để trống.', 16, 1);

RETURN;

END;

-- Kiểm tra điều kiện hợp lệ cho tên môn học và số tín chỉ

IF @SoTinChi < 1

OR @SoTinChi > 5

BEGIN

RAISERROR('Số tín chỉ phải từ 1 đến 5.', 16, 1);

RETURN;

END;

-- Chèn dữ liệu hợp lệ vào bảng MonHoc

INSERT INTO MonHoc (MaMonHoc, TenMonHoc, SoTinChi)

VALUES (@MaMonHoc, @TenMonHoc, @SoTinChi);

END;

### Đảm bảo dữ liệu đúng khi thêm lớp học

CREATE TRIGGER trg\_ThayViThemLopHoc ON LopHoc

INSTEAD OF INSERT

AS

BEGIN

DECLARE @MaLopHoc INT, @Thu TINYINT, @TietBatDau TINYINT, @TietKetThuc TINYINT, @MaPhongHoc INT;

SELECT

@MaLopHoc = i.MaLopHoc,

@Thu = i.Thu,

@TietBatDau = i.TietBatDau,

@TietKetThuc = i.TietKetThuc,

@MaPhongHoc = i.MaPhongHoc

FROM inserted i;

-- Kiểm tra trùng mã lớp học

IF EXISTS (

SELECT 1

FROM LopHoc lh

WHERE lh.MaLopHoc = @MaLopHoc

)

BEGIN

RAISERROR ('Mã lớp học %d đã tồn tại.', 16, 1, @MaLopHoc);

RETURN;

END

-- Kiểm tra điều kiện hợp lệ cho thứ

IF @Thu < 1 OR @Thu > 7

BEGIN

RAISERROR('Thứ phải từ 1 đến 7.', 16, 1);

RETURN;

END;

-- Kiểm tra điều kiện hợp lệ cho tiết học

IF @TietBatDau < 1 OR @TietBatDau > 12

OR @TietKetThuc < 1 OR @TietKetThuc > 12

BEGIN

RAISERROR('Tiết bắt đầu và kết thúc phải từ 1 đến 12.', 16, 1);

RETURN;

END;

IF @TietBatDau > @TietKetThuc

BEGIN

RAISERROR('Tiết bắt đầu không được lớn hơn tiết kết thúc.', 16, 1);

RETURN;

END;

-- Kiểm tra trùng lịch học với lớp học khác

DECLARE @MaLopHocTrung INT;

SELECT @MaLopHocTrung = lh.MaLopHoc

FROM LopHoc lh

WHERE lh.MaLopHoc != @MaLopHoc AND lh.MaPhongHoc = @MaPhongHoc AND lh.Thu = @Thu AND

(

(@TietKetThuc BETWEEN lh.TietBatDau AND lh.TietKetThuc)OR

(@TietBatDau BETWEEN lh.TietBatDau AND lh.TietKetThuc)

)

IF @MaLopHocTrung IS NOT NULL

BEGIN

RAISERROR ('Lớp học %d cần được thêm bị trùng lịch với lớp học %d cùng phòng.', 16, 1, @MaLopHoc, @MaLopHocTrung);

RETURN;

END

-- Chèn lớp học mới

INSERT INTO LopHoc (MaLopHoc, Thu, TietBatDau, TietKetThuc, MaPhongHoc, MaGiangVien, MaMonHoc)

SELECT i.MaLopHoc, i.Thu, i.TietBatDau, i.TietKetThuc, i.MaPhongHoc, i.MaGiangVien, i.MaMonHoc

FROM inserted i;

END;

### Đảm bảo dữ liệu đúng khi thêm đăng ký

CREATE TRIGGER trg\_ThayViThemDangKy ON DangKy

INSTEAD OF INSERT

AS

BEGIN

-- Kiểm tra xem sinh viên có tồn tại không

IF NOT EXISTS (

SELECT 1

FROM SinhVien sv

JOIN inserted i ON sv.MaSinhVien = i.MaSinhVien

)

BEGIN

RAISERROR(N'Không tồn tại mã sinh viên muốn chèn.', 16, 1);

RETURN;

END

-- Kiểm tra xem lớp học có tồn tại không

IF NOT EXISTS (

SELECT 1

FROM LopHoc lh

JOIN inserted i ON lh.MaLopHoc = i.MaLopHoc)

BEGIN

RAISERROR(N'Không tồn tại mã lớp học muốn chèn.', 16, 1);

RETURN;

END

-- Lấy mã sinh viên, mã lớp học, và mã môn học sinh viên muốn đăng ký

DECLARE @MaSinhVien INT, @MaLopHoc INT, @MaMonHocSinhVienDangKy INT;

SELECT @MaSinhVien = i.MaSinhVien, @MaLopHoc = lh.MaLopHoc, @MaMonHocSinhVienDangKy = lh.MaMonHoc

FROM inserted i

INNER JOIN LopHoc lh ON i.MaLopHoc = lh.MaLopHoc;

-- Tìm trong những lớp học mà sinh viên đã đăng ký, kiểm tra có lớp học nào đang dạy môn mà sinh viên muốn đăng ký hay không

DECLARE @MaLopHocBiTrung INT;

SELECT @MaLopHocBiTrung = dk.MaLopHoc

FROM DangKy dk

INNER JOIN LopHoc lh ON dk.MaLopHoc = lh.MaLopHoc

WHERE dk.MaSinhVien = @MaSinhVien AND lh.MaMonHoc = @MaMonHocSinhVienDangKy;

-- Kiểm tra trùng môn

-- Nếu tìm thấy lớp học bị trùng, ném lỗi

IF @MaLopHocBiTrung IS NOT NULL

BEGIN

RAISERROR ('Sinh viên (MaSinhVien = %d) muốn đăng ký môn học (MaMonHoc = %d), nhưng đã bị trùng với lớp học (MaLopHoc = %d).', 16, 1, @MaSinhVien, @MaMonHocSinhVienDangKy, @MaLopHocBiTrung);

RETURN;

END

-- Chèn dữ liệu nếu không có lỗi

INSERT INTO DangKy (MaSinhVien, MaLopHoc, DiemQuaTrinh, DiemCuoiKy)

SELECT i.MaSinhVien, i.MaLopHoc, i.DiemQuaTrinh, i.DiemCuoiKy

FROM inserted i;

END;

### Số lượng sinh viên không vượt quá sức chứa của phòng sau khi đăng ký

CREATE TRIGGER trg\_SauKhiThemDangKy ON DangKy

AFTER INSERT

AS

BEGIN

-- Nếu như phòng học bị vượt quá sức chứa thì sẽ hủy việc đăng ký vừa thực hiện

DECLARE @MaLopHoc INT, @SoLuongSinhVien INT, @SucChua INT;

SELECT @MaLopHoc = i.MaLopHoc, @SucChua = ph.SucChua

FROM inserted i

INNER JOIN LopHoc lh ON i.MaLopHoc = lh.MaLopHoc

INNER JOIN PhongHoc ph on lh.MaPhongHoc = ph.MaPhongHoc

SELECT @SoLuongSinhVien = COUNT(dk.MaSinhVien)

FROM DangKy dk

WHERE dk.MaLopHoc = @MaLopHoc;

IF @SoLuongSinhVien > @SucChua

BEGIN

RAISERROR ('Không thể đăng ký vì lớp học (MaLopHoc = %d) đã đạt sức chứa tối đa', 16, 1, @MaLopHoc);

ROLLBACK TRANSACTION;

END

END;

### Gửi mail cho sinh viên bị rớt môn

CREATE TRIGGER trg\_SauKhiSuaDangKy ON DangKy

AFTER UPDATE

AS

BEGIN

DECLARE @MaSinhVien INT, @MaLopHoc INT, @DiemQuaTrinh DECIMAL(4, 2), @DiemCuoiKy DECIMAL(4, 2);

-- Lấy dữ liệu từ bảng inserted

SELECT

@MaSinhVien = i.MaSinhVien,

@MaLopHoc = i.MaLopHoc,

@DiemQuaTrinh = i.DiemQuaTrinh,

@DiemCuoiKy = i.DiemCuoiKy

FROM inserted i;

DECLARE @MaMonHoc INT;

SELECT @MaMonHoc = lh.MaMonHoc

FROM DangKy dk

JOIN LopHoc lh ON dk.MaLopHoc = lh.MaLopHoc

WHERE lh.MaLopHoc = @MaLopHoc

DECLARE @QuaMon INT;

SELECT @QuaMon = dbo.fn\_KiemTraQuaMon(@MaSinhVien, @MaMonHoc)

-- Nếu sinh viên rớt môn thì gửi mail

IF @QuaMon = 0

BEGIN

DECLARE @EmailSinhVien VARCHAR(100), @TenMonHoc NVARCHAR(100);

-- Lấy thông tin email của sinh viên rớt môn

SELECT @EmailSinhVien = sv.Email

FROM SinhVien sv

WHERE sv.MaSinhVien = @MaSinhVien

-- Lấy thêm thông tin môn học bị rớt

SELECT @TenMonHoc = mh.TenMonHoc

FROM MonHoc mh

WHERE mh.MaMonHoc = @MaMonHoc;

DECLARE @EmailBody NVARCHAR(300);

-- Tạo nội dung email trong biến @EmailBody

SET @EmailBody = N'Dear Student,' + CHAR(13) + CHAR(10) +

N'You have failed a class.' + CHAR(13) + CHAR(10) +

N'Class ID: ' + CAST(@MaLopHoc AS NVARCHAR(10)) + CHAR(13) + CHAR(10) +

N'Subject ID: ' + CAST(@MaMonHoc AS NVARCHAR(10)) + CHAR(13) + CHAR(10) +

N'Subject Name: ' + @TenMonHoc + CHAR(13) + CHAR(10) +

N'If you consider this a misunderstanding, please contact your advisor for more details.';

-- Gọi hàm gửi email

EXEC msdb.dbo.sp\_send\_dbmail

@profile\_name = 'nhp\_profile',

@recipients = @EmailSinhVien,

@subject = 'Notice of Failure',

@body = @EmailBody;

END

END;

## Các view

### Danh sách sinh viên

Code SQL:

-- Tạo view vw\_SinhVien để truy xuất thông tin sinh viên từ bảng SinhVien.

-- View này bao gồm các trường: MaSinhVien, HoVaTen, Email, SoDienThoai và QueQuan.

CREATE VIEW vw\_SinhVien AS

SELECT sv.MaSinhVien, sv.HoVaTen, sv.Email, sv.SoDienThoai, sv.QueQuan

FROM SinhVien sv

Code C#:

/// <summary>

/// Lấy danh sách sinh viên từ view vw\_SinhVien.

/// </summary>

/// <returns>

/// Một bảng chứa thông tin của toàn bộ sinh viên gồm các cột:

/// <br>- MaSinhVien</br>

/// <br>- HoVaTen</br>

/// <br>- Email</br>

/// <br>- SoDienThoai</br>

/// <br>- QueQuan</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public DataTable GetViewSinhVien()

{

    try

    {

        DataTable danhSachSinhVien = new DataTable();

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            string query = @"SELECT MaSinhVien, HoVaTen, Email, SoDienThoai, QueQuan FROM vw\_SinhVien";

            using (SqlCommand cmd = new SqlCommand(query, connection))

            {

                using (SqlDataAdapter adapter = new SqlDataAdapter(cmd))

                {

                    adapter.Fill(danhSachSinhVien);

                }

            }

        }

        return danhSachSinhVien;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Danh sách giảng viên

Code SQL:

-- Tạo view vw\_GiangVien để truy xuất thông tin giảng viên từ bảng GiangVien.

-- View này bao gồm các trường: MaGiangVien, HoVaTen, Email, SoDienThoai.

CREATE VIEW vw\_GiangVien AS

SELECT gv.MaGiangVien, gv.HoVaTen, gv.Email, gv.SoDienThoai

FROM GiangVien gv

Code C#:

/// <summary>

/// Lấy danh sách giảng viên từ view vw\_GiangVien.

/// </summary>

/// <returns>

/// Một bảng chứa thông tin của toàn bộ giảng viên gồm các cột:

/// <br>- MaGiangVien</br>

/// <br>- HoVaTen</br>

/// <br>- Email</br>

/// <br>- SoDienThoai</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public DataTable GetViewGiangVien()

{

    try

    {

        DataTable danhSachGiangVien = new DataTable();

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            string query = @"SELECT MaGiangVien, HoVaTen, Email, SoDienThoai FROM vw\_GiangVien";

            using (SqlCommand cmd = new SqlCommand(query, connection))

            {

                using (SqlDataAdapter adapter = new SqlDataAdapter(cmd))

                {

                    adapter.Fill(danhSachGiangVien);

                }

            }

        }

        return danhSachGiangVien;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Danh sách môn học

Code SQL:

-- Tạo view vw\_MonHoc để truy xuất thông tin môn học từ bảng MonHoc.

-- View này bao gồm các trường: MaMonHoc, TenMonHoc, SoTinChi.

CREATE VIEW vw\_MonHoc AS

SELECT mh.MaMonHoc, mh.TenMonHoc, mh.SoTinChi

FROM MonHoc mh

Code C#:

/// <summary>

/// Lấy danh sách môn học từ view vw\_MonHoc.

/// </summary>

/// <returns>

/// Một bảng chứa thông tin của toàn bộ môn học gồm các cột:

/// <br>- MaMonHoc</br>

/// <br>- TenMonHoc</br>

/// <br>- SoTinChi</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public DataTable GetViewMonHoc()

{

    try

    {

        DataTable danhSachMonHoc = new DataTable();

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            string query = @"SELECT MaMonHoc, TenMonHoc, SoTinChi FROM vw\_MonHoc";

            using (SqlCommand cmd = new SqlCommand(query, connection))

            {

                using (SqlDataAdapter adapter = new SqlDataAdapter(cmd))

                {

                    adapter.Fill(danhSachMonHoc);

                }

            }

        }

        return danhSachMonHoc;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Danh sách lớp học

Code SQL:

-- Tạo view vw\_LopHoc để truy xuất thông tin môn học từ bảng LopHoc.

-- View này bao gồm các trường: MaMonHoc, TenMonHoc, SoTinChi

CREATE VIEW vw\_LopHoc AS

SELECT

MaLopHoc,

Thu,

TietBatDau,

TietKetThuc,

MaPhongHoc,

MaGiangVien,

MaMonHoc

FROM LopHoc;

Code C#:

/// <summary>

/// Lấy danh sách lớp học từ view vw\_LopHoc.

/// </summary>

/// <returns>

/// Một bảng chứa thông tin của toàn bộ lớp học gồm các cột:

/// <br>- MaLopHoc</br>

/// <br>- Thu</br>

/// <br>- TietBatDau</br>

/// <br>- TietKetThuc</br>

/// <br>- MaPhongHoc</br>

/// <br>- MaGiangVien</br>

/// <br>- MaMonHoc</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public DataTable GetViewLopHoc()

{

    try

    {

        DataTable danhSachLopHoc = new DataTable();

        // Thiết lập kết nối với cơ sở dữ liệu

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open(); // Mở kết nối đến cơ sở dữ liệu

            // Truy vấn để lấy dữ liệu từ view vw\_LopHoc

            string query = @"SELECT

                                MaLopHoc,

                                Thu,

                                TietBatDau,

                                TietKetThuc,

                                MaPhongHoc,

                                MaGiangVien,

                                MaMonHoc

                            FROM vw\_LopHoc";

            // Tạo đối tượng SqlCommand để thực thi truy vấn

            using (SqlCommand cmd = new SqlCommand(query, connection))

            {

                // Sử dụng SqlDataAdapter để điền dữ liệu vào DataTable

                using (SqlDataAdapter adapter = new SqlDataAdapter(cmd))

                {

                    adapter.Fill(danhSachLopHoc); // Điền dữ liệu vào DataTable

                }

            }

        }

        return danhSachLopHoc; // Trả về DataTable chứa danh sách lớp học

    }

    catch (SqlException)

    {

        throw; // Ném lại ngoại lệ SqlException nếu có lỗi xảy ra

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}"); // Ném lại ngoại lệ khác

    }

}

### Những lớp học giảng viên phụ trách

Code SQL:

-- Tạo view vw\_LopHocGiangVienPhuTrach để truy xuất thông tin lớp học cùng với giảng viên phụ trách.

-- View này bao gồm các trường:

-- MaLopHoc, Thu, TietBatDau, TietKetThuc, MaPhongHoc, MaMonHoc, MaGiangVien, HoVaTen, Email, SoDienThoai.

CREATE VIEW vw\_LopHocGiangVienPhuTrach AS

SELECT

lh.MaLopHoc,

lh.Thu,

lh.TietBatDau,

lh.TietKetThuc,

lh.MaPhongHoc,

lh.MaMonHoc,

gv.MaGiangVien,

gv.HoVaTen,

gv.Email,

gv.SoDienThoai

FROM LopHoc lh

INNER JOIN GiangVien gv ON lh.MaGiangVien = gv.MaGiangVien;

Code C#:

/// <summary>

/// Lấy danh sách lớp học và giảng viên phụ trách từ view vw\_LopHocGiangVienPhuTrach.

/// </summary>

/// <returns>

/// Một bảng chứa thông tin của toàn bộ lớp học cùng với giảng viên phụ trách gồm các cột:

/// <br>- MaLopHoc</br>

/// <br>- Thu</br>

/// <br>- TietBatDau</br>

/// <br>- TietKetThuc</br>

/// <br>- MaPhongHoc</br>

/// <br>- MaMonHoc</br>

/// <br>- MaGiangVien</br>

/// <br>- HoVaTen</br>

/// <br>- Email</br>

/// <br>- SoDienThoai</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public DataTable GetViewLopHocGiangVienPhuTrach()

{

    try

    {

        DataTable danhSachLopHocGiangVien = new DataTable();

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            string query = @"SELECT

                                MaLopHoc,

                                Thu,

                                TietBatDau,

                                TietKetThuc,

                                MaPhongHoc,

                                MaMonHoc,

                                MaGiangVien,

                                HoVaTen,

                                Email,

                                SoDienThoai

                            FROM vw\_LopHocGiangVienPhuTrach";

            using (SqlCommand cmd = new SqlCommand(query, connection))

            {

                using (SqlDataAdapter adapter = new SqlDataAdapter(cmd))

                {

                    adapter.Fill(danhSachLopHocGiangVien);

                }

            }

        }

        return danhSachLopHocGiangVien;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Các lớp học còn chỗ trống

Code SQL:

-- Tạo view vw\_LopHocConTrong để xem thông tin những lớp học còn trống.

-- View này bao gồm các trường:

-- MaLopHoc, MaPhongHoc, SucChua, SoSinhVienDaDangKy, SoLuongConTrong.

CREATE VIEW vw\_LopHocConTrong AS

SELECT

lh.MaLopHoc,

ph.MaPhongHoc,

ph.SucChua,

COUNT(dk.MaSinhVien) AS SoSinhVienDaDangKy,

(ph.SucChua - COUNT(dk.MaSinhVien)) AS SoLuongConTrong

FROM LopHoc lh

INNER JOIN PhongHoc ph ON lh.MaPhongHoc = ph.MaPhongHoc

LEFT JOIN DangKy dk ON lh.MaLopHoc = dk.MaLopHoc

GROUP BY

lh.MaLopHoc, ph.MaPhongHoc, ph.SucChua

HAVING COUNT(dk.MaSinhVien) < ph.SucChua;

Code C#:

/// <summary>

/// Lấy danh sách các lớp học còn trống và số lượng chỗ trống.

/// </summary>

/// <returns>

/// Trả về một DataTable chứa thông tin các lớp học còn trống sinh viên, bao gồm các cột:

/// <br>- MaLopHoc</br>

/// <br>- MaPhongHoc</br>

/// <br>- SucChua</br>

/// <br>- SoSinhVienDaDangKy</br>

/// <br>- SoLuongConTrong</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public DataTable GetDanhSachLopHocConTrong()

{

    try

    {

        DataTable danhSachLopHocConTrong = new DataTable();

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            string query = @"SELECT

                                MaLopHoc,

                                MaPhongHoc,

                                SucChua,

                                SoSinhVienDaDangKy,

                                SoLuongConTrong

                             FROM vw\_LopHocConTrong";

            using (SqlCommand cmd = new SqlCommand(query, connection))

            {

                using (SqlDataAdapter adapter = new SqlDataAdapter(cmd))

                {

                    adapter.Fill(danhSachLopHocConTrong);

                }

            }

        }

        return danhSachLopHocConTrong;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Điểm trung bình của các sinh viên

Code SQL:

-- Tạo view vw\_DiemTrungBinhSinhVien để truy xuất thông tin điểm trung bình của sinh viên.

-- View này bao gồm các trường:

-- MaSinhVien, HoVaTen, DiemTrungBinh.

CREATE VIEW vw\_DiemTrungBinhSinhVien AS

SELECT

sv.MaSinhVien,

sv.HoVaTen,

AVG((dk.DiemQuaTrinh + dk.DiemCuoiKy) / 2.0) AS DiemTrungBinh

FROM SinhVien sv

INNER JOIN DangKy dk ON sv.MaSinhVien = dk.MaSinhVien

WHERE dk.DiemQuaTrinh IS NOT NULL AND dk.DiemCuoiKy IS NOT NULL

GROUP BY sv.MaSinhVien, sv.HoVaTen;

Code C#:

/// <summary>

/// Lấy danh sách điểm trung bình của sinh viên từ view vw\_DiemTrungBinhSinhVien.

/// </summary>

/// <returns>

/// Một bảng chứa thông tin về điểm trung bình của sinh viên gồm các cột:

/// <br>- MaSinhVien</br>

/// <br>- HoVaTen</br>

/// <br>- DiemTrungBinh</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public DataTable GetViewDiemTrungBinhSinhVien()

{

    try

    {

        DataTable danhSachDiemTrungBinh = new DataTable();

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            string query = @"SELECT MaSinhVien, HoVaTen, DiemTrungBinh

                     FROM vw\_DiemTrungBinhSinhVien";

            using (SqlCommand cmd = new SqlCommand(query, connection))

            {

                using (SqlDataAdapter adapter = new SqlDataAdapter(cmd))

                {

                    adapter.Fill(danhSachDiemTrungBinh);

                }

            }

        }

        return danhSachDiemTrungBinh;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Điểm trung bình tích lũy của các sinh viên

Code SQL:

-- Tạo view vw\_DiemTrungBinhTichLuySinhVien để truy xuất thông tin điểm trung bình tích lũy của sinh viên.

-- View này bao gồm các trường:

-- MaSinhVien, HoVaTen, DiemTrungBinhTichLuy.

CREATE VIEW vw\_DiemTrungBinhTichLuySinhVien AS

SELECT

sv.MaSinhVien,

sv.HoVaTen,

AVG((dk.DiemQuaTrinh + dk.DiemCuoiKy) / 2.0) AS DiemTrungBinhTichLuy

FROM SinhVien sv

INNER JOIN DangKy dk ON sv.MaSinhVien = dk.MaSinhVien

WHERE

dk.DiemQuaTrinh IS NOT NULL

AND dk.DiemCuoiKy IS NOT NULL

AND dk.DiemCuoiKy >= 3.0

AND ((dk.DiemQuaTrinh + dk.DiemCuoiKy) / 2.0) >= 5.0

GROUP BY

sv.MaSinhVien, sv.HoVaTen;

Code C#:

/// <summary>

/// Lấy danh sách điểm trung bình tích lũy của sinh viên từ view vw\_DiemTrungBinhTichLuySinhVien.

/// </summary>

/// <returns>

/// Một bảng chứa thông tin về điểm trung bình tích lũy của sinh viên gồm các cột:

/// <br>- MaSinhVien</br>

/// <br>- HoVaTen</br>

/// <br>- DiemTrungBinhTichLuy</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public DataTable GetViewDiemTrungBinhTichLuySinhVien()

{

    try

    {

        DataTable danhSachDiemTrungBinhTichLuy = new DataTable();

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            string query = @"SELECT

                        MaSinhVien,

                        HoVaTen,

                        DiemTrungBinhTichLuy

                     FROM vw\_DiemTrungBinhTichLuySinhVien";

            using (SqlCommand cmd = new SqlCommand(query, connection))

            {

                using (SqlDataAdapter adapter = new SqlDataAdapter(cmd))

                {

                    adapter.Fill(danhSachDiemTrungBinhTichLuy);

                }

            }

        }

        return danhSachDiemTrungBinhTichLuy;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Bảng xếp hạng các sinh viên bằng điểm trung bình tích lũy

Code SQL:

-- Tạo view vw\_XepHangSinhVienBangDiemTBTichLuy để truy xuất bảng xếp hạng các sinh viên theo điểm trung bình tích lũy.

-- View này bao gồm các trường:

-- XepHang, MaSinhVien, HoVaTen, DiemTrungBinhTichLuy.

CREATE VIEW vw\_XepHangSinhVienBangDiemTBTichLuy AS

SELECT

ROW\_NUMBER() OVER (ORDER BY ROUND(SUM(((dk.DiemQuaTrinh + dk.DiemCuoiKy) / 2) \* mh.SoTinChi) / SUM(mh.SoTinChi), 2) DESC) AS XepHang,

sv.MaSinhVien,

sv.HoVaTen,

ROUND(SUM(((dk.DiemQuaTrinh + dk.DiemCuoiKy) / 2) \* mh.SoTinChi) / SUM(mh.SoTinChi), 2) AS DiemTrungBinhTichLuy

FROM

SinhVien sv

JOIN

DangKy dk ON sv.MaSinhVien = dk.MaSinhVien

JOIN

LopHoc lh ON dk.MaLopHoc = lh.MaLopHoc

JOIN

MonHoc mh ON lh.MaMonHoc = mh.MaMonHoc

WHERE

dk.DiemQuaTrinh IS NOT NULL

AND dk.DiemCuoiKy IS NOT NULL

AND dk.DiemCuoiKy >= 3

AND (dk.DiemQuaTrinh + dk.DiemCuoiKy) >= 5

GROUP BY

sv.MaSinhVien, sv.HoVaTen;

Code C#:

/// <summary>

/// Lấy danh sách xếp hạng sinh viên từ view vw\_XepHangSinhVienBangDiemTBTichLuy.

/// </summary>

/// <returns>

/// Trả về một DataTable chứa danh sách xếp hạng sinh viên gồm các cột như:

/// <br>- MaSinhVien: Mã sinh viên</br>

/// <br>- HoVaTen: Họ và tên sinh viên</br>

/// <br>- SoTinChiDaHoanThanh: Số tín chỉ đã hoàn thành</br>

/// <br>- DiemTrungBinh: Điểm trung bình tích lũy</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception

public DataTable GetViewXepHangSinhVien()

{

    try

    {

        DataTable DanhSachXepHang = new DataTable();

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("select \* from vw\_XepHangSinhVienBangDiemTBTichLuy", connection))

            {

                using (SqlDataAdapter adapter = new SqlDataAdapter(command))

                {

                    adapter.Fill(DanhSachXepHang);

                }

            }

        }

        return DanhSachXepHang;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception(ex.Message);

    }

}

# CÀI ĐẶT CHỨC NĂNG

## Kết nối cơ sở dữ liệu

internal class QLDSVCNTTConnection

{

    public static string connectionString = "";

    public QLDSVCNTTConnection(string username= "mainad1", string password="1234")

    {

        connectionString = $@"Data Source=xichxo;Initial Catalog=QLDiemSVKhoaCNTT;User ID={username};Password={password};TrustServerCertificate=True";

    }

    /// <summary>

    /// Test the database connection.

    /// </summary>

    /// <returns>0 nếu kết nối thành công.</returns>

    /// <exception cref="SqlException">

    /// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

    /// </exception>

    /// <exception cref="Exception">

    /// Ném ra khi có lỗi khác không xác định xảy ra.

    /// </exception>

    public static int TestConnection()

    {

        try

        {

            // Sử dụng chuỗi kết nối đã định nghĩa để tạo kết nối tới cơ sở dữ liệu

            string connectionString = QLDSVCNTTConnection.connectionString;

            // Tạo đối tượng SqlConnection với chuỗi kết nối

            using (SqlConnection connection = new SqlConnection(connectionString))

            {

                // Mở kết nối tới cơ sở dữ liệu

                connection.Open();

                // Nếu mở kết nối thành công, trả về 0

                return 0; // Có thể trả về mã 0 nếu kết nối thành công

            }

        }

        catch (SqlException)

        {

            // Ném lại lỗi SQL để xử lý ở nơi gọi hàm

            throw;

        }

        catch (Exception ex)

        {

            // Ném lỗi chung với thông điệp cụ thể

            throw new Exception($"Error: {ex.Message}");

        }

    }

}

## Thủ tục

### Thêm sinh viên

Code SQL:

CREATE PROCEDURE proc\_ThemSinhVien

@MaSinhVien INT, -- Mã sinh viên

@HoVaTen NVARCHAR(100), -- Họ và tên sinh viên

@Email VARCHAR(100), -- Email sinh viên

@SoDienThoai VARCHAR(20), -- Số điện thoại sinh viên

@QueQuan NVARCHAR(100) -- Quê quán của sinh viên

AS

BEGIN

BEGIN TRY

INSERT INTO SinhVien (MaSinhVien, HoVaTen, Email, SoDienThoai, QueQuan)

VALUES (@MaSinhVien, @HoVaTen, @Email, @SoDienThoai, @QueQuan);

PRINT N'Sinh viên ' + CAST(@MaSinhVien AS NVARCHAR) + N' đã được thêm thành công.';

END TRY

BEGIN CATCH

-- Bắt thông báo lỗi từ trigger

DECLARE @ErrorMessage NVARCHAR(4000) = ERROR\_MESSAGE();

RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END;

Code C#:

public int ThemSinhVien(int maSinhVien, string hoVaTen, string email, string soDienThoai, string queQuan)

{

    try

    {

        string connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_ThemSinhVien", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.AddWithValue("@MaSinhVien", maSinhVien);

                command.Parameters.AddWithValue("@HoVaTen", hoVaTen);

                command.Parameters.AddWithValue("@Email", email);

                command.Parameters.AddWithValue("@SoDienThoai", soDienThoai);

                command.Parameters.AddWithValue("@QueQuan", queQuan);

                return command.ExecuteNonQuery();

            }

        }

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        // Handle other exceptions

        throw new Exception($"Error: {ex.Message}");

    }

}

### Thêm giảng viên

Code SQL:

CREATE PROCEDURE proc\_ThemGiangVien

@MaGiangVien INT, -- Mã giảng viên

@HoVaTen NVARCHAR(100), -- Họ và tên giảng viên

@Email NVARCHAR(100), -- Email của giảng viên

@SoDienThoai NVARCHAR(20) -- Số điện thoại của giảng viên

AS

BEGIN

BEGIN TRY

INSERT INTO GiangVien (MaGiangVien, HoVaTen, Email, SoDienThoai)

VALUES (@MaGiangVien, @HoVaTen, @Email, @SoDienThoai);

PRINT N'Giảng viên ' + CAST(@MaGiangVien AS NVARCHAR) + N' đã được thêm thành công.';

END TRY

BEGIN CATCH

-- Bắt thông báo lỗi từ trigger

DECLARE @ErrorMessage NVARCHAR(4000) = ERROR\_MESSAGE();

RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END;

Code C#:

/// <summary>

/// Thêm một giảng viên mới vào cơ sở dữ liệu.

/// </summary>

/// <param name="maGiangVien">Mã giảng viên cần thêm.</param>

/// <param name="hoVaTen">Họ và tên của giảng viên.</param>

/// <param name="email">Email của giảng viên.</param>

/// <param name="soDienThoai">Số điện thoại của giảng viên.</param>

/// <returns>Trả về số lượng bản ghi bị ảnh hưởng (thông thường là 1 nếu thành công).</returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public int ThemGiangVien(int maGiangVien, string hoVaTen, string email, string soDienThoai)

{

    try

    {

        string connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_ThemGiangVien", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                // Thêm các tham số vào Stored Procedure

                command.Parameters.AddWithValue("@MaGiangVien", maGiangVien);

                command.Parameters.AddWithValue("@HoVaTen", hoVaTen);

                command.Parameters.AddWithValue("@Email", email);

                command.Parameters.AddWithValue("@SoDienThoai", soDienThoai);

                // Thực thi lệnh và trả về số dòng bị ảnh hưởng

                return command.ExecuteNonQuery();

            }

        }

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Thêm môn học

Code SQL:

CREATE PROCEDURE proc\_ThemMonHoc

@MaMonHoc INT, -- Mã môn học cần thêm

@TenMonHoc NVARCHAR(100), -- Tên môn học cần thêm

@SoTinChi TINYINT -- Số tín chỉ

AS

BEGIN

BEGIN TRY

INSERT INTO MonHoc(MaMonHoc, TenMonHoc, SoTinChi)

VALUES (@MaMonHoc, @TenMonHoc, @SoTinChi)

PRINT N'Môn học ' + CAST(@MaMonHoc AS NVARCHAR) + N' đã được thêm thành công.';

END TRY

BEGIN CATCH

-- Bắt thông báo lỗi từ trigger

DECLARE @ErrorMessage NVARCHAR(4000) = ERROR\_MESSAGE();

RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END;

Code C#:

/// <summary>

/// Thêm một môn học mới vào cơ sở dữ liệu thông qua stored procedure "proc\_ThemMonHoc".

/// </summary>

/// <param name="maMonHoc">

/// Mã môn học của môn học cần thêm.

/// </param>

/// <param name="tenMonHoc">

/// Tên môn học của môn học cần thêm.

/// </param>

/// <param name="soTinChi">

/// Số tín chỉ của môn học cần thêm.

/// </param>

/// <returns>

/// Trả về số dòng bị ảnh hưởng sau khi thực thi stored procedure.

/// <br>- Nếu trả về giá trị > 0, nghĩa là thêm môn học thành công.</br>

/// <br>- Nếu trả về 0, nghĩa là không có bản ghi nào được thêm.</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc thực thi câu lệnh SQL.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi không xác định khác.

/// </exception>

public int ThemMonHoc(int maMonHoc, string tenMonHoc, byte soTinChi)

{

    try

    {

        string connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_ThemMonHoc", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.AddWithValue("@MaMonHoc", maMonHoc);

                command.Parameters.AddWithValue("@TenMonHoc", tenMonHoc);

                command.Parameters.AddWithValue("@SoTinChi", soTinChi);

                return command.ExecuteNonQuery();

            }

        }

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Thêm lớp học

Code SQL:

CREATE PROCEDURE proc\_ThemLopHoc

@MaLopHoc INT, -- Mã lớp học cần thêm

@Thu TINYINT, -- Ngày trong tuần

@TietBatDau TINYINT, -- Tiết bắt đầu

@TietKetThuc TINYINT, -- Tiết kết thúc

@MaPhongHoc INT, -- Mã phòng học

@MaGiangVien INT, -- Mã giảng viên

@MaMonHoc INT -- Mã môn học

AS

BEGIN

BEGIN TRY

INSERT INTO LopHoc(MaLopHoc, Thu, TietBatDau, TietKetThuc, MaPhongHoc, MaGiangVien, MaMonHoc)

VALUES (@MaLopHoc, @Thu, @TietBatDau, @TietKetThuc, @MaPhongHoc, @MaGiangVien, @MaMonHoc);

PRINT N'Lớp học ' + CAST(@MaLopHoc AS NVARCHAR) + N' đã được thêm thành công.';

END TRY

BEGIN CATCH

-- Bắt thông báo lỗi từ trigger

DECLARE @ErrorMessage NVARCHAR(4000) = ERROR\_MESSAGE();

RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END;

Code C#:

/// <summary>

/// Thêm một lớp học mới vào cơ sở dữ liệu thông qua stored procedure "proc\_ThemLopHoc".

/// </summary>

/// <param name="maLopHoc">

/// Mã lớp học của lớp học cần thêm.

/// </param>

/// <param name="thu">

/// Ngày trong tuần của lớp học cần thêm (1-7).

/// </param>

/// <param name="tietBatDau">

/// Tiết bắt đầu của lớp học cần thêm.

/// </param>

/// <param name="tietKetThuc">

/// Tiết kết thúc của lớp học cần thêm.

/// </param>

/// <param name="maPhongHoc">

/// Mã phòng học của lớp học cần thêm.

/// </param>

/// <param name="maGiangVien">

/// Mã giảng viên phụ trách lớp học cần thêm.

/// </param>

/// <param name="maMonHoc">

/// Mã môn học của lớp học cần thêm.

/// </param>

/// <returns>

/// Trả về số dòng bị ảnh hưởng sau khi thực thi stored procedure.

/// <br>- Nếu trả về giá trị > 0, nghĩa là thêm lớp học thành công.</br>

/// <br>- Nếu trả về 0, nghĩa là không có bản ghi nào được thêm.</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc thực thi câu lệnh SQL.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi không xác định khác.

/// </exception>

public int ThemLopHoc(int maLopHoc, byte thu, byte tietBatDau, byte tietKetThuc, int maPhongHoc, int maGiangVien, int maMonHoc)

{

    try

    {

        string connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_ThemLopHoc", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.AddWithValue("@MaLopHoc", maLopHoc);

                command.Parameters.AddWithValue("@Thu", thu);

                command.Parameters.AddWithValue("@TietBatDau", tietBatDau);

                command.Parameters.AddWithValue("@TietKetThuc", tietKetThuc);

                command.Parameters.AddWithValue("@MaPhongHoc", maPhongHoc);

                command.Parameters.AddWithValue("@MaGiangVien", maGiangVien);

                command.Parameters.AddWithValue("@MaMonHoc", maMonHoc);

                return command.ExecuteNonQuery();

            }

        }

    }

    catch (SqlException)

    {

        throw; // Ném lại ngoại lệ SqlException nếu có lỗi xảy ra

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}"); // Ném lại ngoại lệ khác

    }

}

### Xóa sinh viên

Code SQL:

CREATE PROCEDURE proc\_XoaSinhVien

@MaSinhVien INT -- Mã sinh viên cần xóa

AS

BEGIN

BEGIN TRY

DELETE FROM SinhVien

WHERE MaSinhVien = @MaSinhVien;

PRINT N'Sinh viên ' + CAST(@MaSinhVien AS NVARCHAR) + N' đã được xóa thành công.';

END TRY

BEGIN CATCH

-- Bắt thông báo lỗi từ trigger

DECLARE @ErrorMessage NVARCHAR(4000) = ERROR\_MESSAGE();

RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END;

Code C#:

/// <summary>

/// Xóa sinh viên khỏi cơ sở dữ liệu theo mã sinh viên.

/// </summary>

/// <param name="maSinhVien">Mã sinh viên cần xóa.</param>

/// <returns>Trả về số lượng bản ghi bị ảnh hưởng (0 hoặc 1).</returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public int XoaSinhVien(int maSinhVien)

{

    try

    {

        string connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_XoaSinhVien", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                // Thêm tham số mã sinh viên vào Stored Procedure

                command.Parameters.AddWithValue("@MaSinhVien", maSinhVien);

                // Thực thi lệnh và trả về số dòng bị ảnh hưởng

                return command.ExecuteNonQuery();

            }

        }

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Xóa giảng viên

Code SQL:

CREATE PROCEDURE proc\_XoaGiangVien

@MaGiangVien INT -- Mã giảng viên cần xóa

AS

BEGIN

BEGIN TRY

DELETE FROM GiangVien

WHERE MaGiangVien = @MaGiangVien;

PRINT N'Giảng viên ' + CAST(@MaGiangVien AS NVARCHAR) + N' đã được xóa thành công.';

END TRY

BEGIN CATCH

-- Bắt thông báo lỗi từ trigger

DECLARE @ErrorMessage NVARCHAR(4000) = ERROR\_MESSAGE();

RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END;

Code C#:

/// <summary>

/// Xóa giảng viên khỏi cơ sở dữ liệu theo mã giảng viên.

/// </summary>

/// <param name="maGiangVien">Mã giảng viên cần xóa.</param>

/// <returns>Trả về số lượng bản ghi bị ảnh hưởng (0 hoặc 1).</returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public int XoaGiangVien(int maGiangVien)

{

    try

    {

        string connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_XoaGiangVien", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.AddWithValue("@MaGiangVien", maGiangVien);

                return command.ExecuteNonQuery();

            }

        }

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Xóa môn học

Code SQL:

CREATE PROCEDURE proc\_XoaMonHoc

@MaMonHoc INT -- Mã môn học cần xóa

AS

BEGIN

BEGIN TRY

DELETE FROM MonHoc

WHERE MaMonHoc = @MaMonHoc;

PRINT N'Môn học ' + CAST(@MaMonHoc AS NVARCHAR) + N' đã được xóa thành công.';

END TRY

BEGIN CATCH

-- Bắt thông báo lỗi từ trigger

DECLARE @ErrorMessage NVARCHAR(4000) = ERROR\_MESSAGE();

RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END;

Code C#:

/// <summary>

/// Xóa môn học khỏi cơ sở dữ liệu theo mã môn học.

/// </summary>

/// <param name="maMonHoc">Mã môn học cần xóa.</param>

/// <returns>Trả về số lượng bản ghi bị ảnh hưởng (0 hoặc 1).</returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public int XoaMonHoc(int maMonHoc)

{

    try

    {

        string connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_XoaMonHoc", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                // Thêm tham số mã môn học vào Stored Procedure

                command.Parameters.AddWithValue("@MaMonHoc", maMonHoc);

                // Thực thi lệnh và trả về số dòng bị ảnh hưởng

                return command.ExecuteNonQuery();

            }

        }

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Xóa lớp học

Code SQL:

CREATE PROCEDURE proc\_XoaLopHoc

@MaLopHoc INT -- Mã lớp học cần xóa

AS

BEGIN

BEGIN TRY

DELETE FROM LopHoc WHERE MaLopHoc = @MaLopHoc;

PRINT N'Lớp học ' + CAST(@MaLopHoc AS NVARCHAR) + N' đã được xóa thành công.';

END TRY

BEGIN CATCH

-- Bắt thông báo lỗi từ trigger

DECLARE @ErrorMessage NVARCHAR(4000) = ERROR\_MESSAGE();

RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END;

Code C#:

/// <summary>

/// Xóa một lớp học khỏi cơ sở dữ liệu thông qua stored procedure "proc\_XoaLopHoc".

/// </summary>

/// <param name="maLopHoc">

/// Mã lớp học của lớp học cần xóa.

/// </param>

/// <returns>

/// Trả về số dòng bị ảnh hưởng sau khi thực thi stored procedure.

/// <br>- Nếu trả về giá trị > 0, nghĩa là xóa lớp học thành công.</br>

/// <br>- Nếu trả về 0, nghĩa là không có bản ghi nào được xóa.</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc thực thi câu lệnh SQL.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi không xác định khác.

/// </exception>

public int XoaLopHoc(int maLopHoc)

{

    try

    {

        string connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_XoaLopHoc", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.AddWithValue("@MaLopHoc", maLopHoc);

                return command.ExecuteNonQuery();

            }

        }

    }

    catch (SqlException)

    {

        throw; // Ném lại ngoại lệ SqlException nếu có lỗi xảy ra

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}"); // Ném lại ngoại lệ khác

    }

}

### Sửa thông tin sinh viên

Code SQL:

CREATE PROCEDURE proc\_SuaSinhVien

@MaSinhVien INT, -- Mã sinh viên cần sửa

@HoVaTen NVARCHAR(100), -- Họ và Tên mới

@Email VARCHAR(100), -- Email mới

@SoDienThoai VARCHAR(20), -- Số điện thoại mới

@QueQuan NVARCHAR(100) -- Quê quán mới

AS

BEGIN

BEGIN TRY

UPDATE SinhVien

SET HoVaTen = @HoVaTen,

Email = @Email,

SoDienThoai = @SoDienThoai,

QueQuan = @QueQuan

WHERE MaSinhVien = @MaSinhVien

PRINT N'Sửa sinh viên ' + CAST(@MaSinhVien AS NVARCHAR) + N'thành công'

END TRY

BEGIN CATCH

-- Bắt thông báo lỗi từ trigger

DECLARE @ErrorMessage NVARCHAR(4000) = ERROR\_MESSAGE();

RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END;

Code C#:

/// <summary>

/// Cập nhật thông tin sinh viên trong cơ sở dữ liệu thông qua stored procedure "proc\_SuaSinhVien".

/// </summary>

/// <param name="maSV">

/// Mã sinh viên cần cập nhật.

/// </param>

/// <param name="hoten">

/// Họ và tên mới của sinh viên.

/// </param>

/// <param name="email">

/// Email mới của sinh viên.

/// </param>

/// <param name="sdt">

/// Số điện thoại mới của sinh viên.

/// </param>

/// <param name="quequan">

/// Quê quán mới của sinh viên.

/// </param>

/// <returns>

/// Trả về số dòng bị ảnh hưởng sau khi thực thi stored procedure.

/// <br>- Nếu trả về giá trị > 0, nghĩa là cập nhật sinh viên thành công.</br>

/// <br>- Nếu trả về 0, nghĩa là không có bản ghi nào được cập nhật.</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc thực thi câu lệnh SQL.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi không xác định khác.

/// </exception>

public int SuaSinhVien(int maSV, string hoten, string email, string sdt, string quequan)

{

    try

    {

        string connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_SuaSinhVien", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.Add(new SqlParameter("@MaSinhVien", maSV));

                command.Parameters.Add(new SqlParameter("@HoVaTen", hoten));

                command.Parameters.Add(new SqlParameter("@Email", email));

                command.Parameters.Add(new SqlParameter("@SoDienThoai", sdt));

                command.Parameters.Add(new SqlParameter("@QueQuan", quequan));

                return command.ExecuteNonQuery();

            }

        }

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Sửa thông tin giảng viên

Code SQL:

CREATE PROCEDURE proc\_SuaGiangVien

@MaGiangVien INT, -- Mã giảng viên cần sửa

@HoVaTen NVARCHAR(100), -- Họ và tên mới của giảng viên

@Email VARCHAR(100), -- Email mới của giảng viên

@SoDienThoai VARCHAR(20) -- Số điện thoại mới của giảng viên

AS

BEGIN

BEGIN TRY

UPDATE GiangVien

SET HoVaTen = @HoVaTen, Email = @Email, SoDienThoai = @SoDienThoai

WHERE MaGiangVien = @MaGiangVien;

PRINT N'Sửa giảng viên ' + CAST(@MaGiangVien AS NVARCHAR) + N' thành công.';

END TRY

BEGIN CATCH

-- Bắt thông báo lỗi từ trigger

DECLARE @ErrorMessage NVARCHAR(4000) = ERROR\_MESSAGE();

RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END;

Code C#:

/// <summary>

/// Cập nhật thông tin giảng viên trong cơ sở dữ liệu.

/// </summary>

/// <param name="maGiangVien">Mã giảng viên cần cập nhật.</param>

/// <param name="hoVaTen">Họ và tên mới của giảng viên.</param>

/// <param name="email">Email mới của giảng viên.</param>

/// <param name="soDienThoai">Số điện thoại mới của giảng viên.</param>

/// <returns>Trả về số lượng bản ghi bị ảnh hưởng (0 hoặc 1).</returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public int SuaGiangVien(int maGiangVien, string hoVaTen, string email, string soDienThoai)

{

    try

    {

        string connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_SuaGiangVien", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                // Thêm tham số cho thủ tục

                command.Parameters.AddWithValue("@MaGiangVien", maGiangVien);

                command.Parameters.AddWithValue("@HoVaTen", hoVaTen);

                command.Parameters.AddWithValue("@Email", email);

                command.Parameters.AddWithValue("@SoDienThoai", soDienThoai);

                // Thực thi lệnh và trả về số dòng bị ảnh hưởng

                return command.ExecuteNonQuery();

            }

        }

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Sửa thông tin môn học

Code SQL:

CREATE PROCEDURE proc\_SuaMonHoc

@MaMonHoc INT, -- Mã môn học cần sửa

@TenMonHoc NVARCHAR(100), -- Tên môn học mới

@SoTinChi TINYINT -- Số tín chỉ mới

AS

BEGIN

BEGIN TRY

UPDATE MonHoc

SET TenMonHoc = @TenMonHoc, SoTinChi = @SoTinChi

WHERE MaMonHoc = @MaMonHoc

PRINT N'Sửa môn học ' + CAST(@MaMonHoc AS NVARCHAR) + N' thành công.'

END TRY

BEGIN CATCH

-- Bắt thông báo lỗi từ trigger

DECLARE @ErrorMessage NVARCHAR(4000) = ERROR\_MESSAGE();

RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END;

Code C#:

/// <summary>

/// Cập nhật thông tin môn học trong cơ sở dữ liệu.

/// </summary>

/// <param name="maMonHoc">Mã môn học cần cập nhật.</param>

/// <param name="tenMonHoc">Tên mới của môn học.</param>

/// <param name="soTinChi">Số tín chỉ mới của môn học.</param>

/// <returns>Trả về số lượng bản ghi bị ảnh hưởng (0 hoặc 1).</returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public int SuaMonHoc(int maMonHoc, string tenMonHoc, byte soTinChi)

{

    try

    {

        string connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_SuaMonHoc", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.AddWithValue("@MaMonHoc", maMonHoc);

                command.Parameters.AddWithValue("@TenMonHoc", tenMonHoc);

                command.Parameters.AddWithValue("@SoTinChi", soTinChi);

                return command.ExecuteNonQuery();

            }

        }

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Sửa lớp học

Code SQL:

CREATE PROCEDURE proc\_SuaLopHoc

@MaLopHoc INT, -- Mã lớp học cần sửa

@Thu TINYINT, -- Ngày trong tuần

@TietBatDau TINYINT, -- Tiết bắt đầu

@TietKetThuc TINYINT, -- Tiết kết thúc

@MaPhongHoc INT, -- Mã phòng học

@MaGiangVien INT, -- Mã giảng viên

@MaMonHoc INT -- Mã môn học

AS

BEGIN

BEGIN TRY

-- Kiểm tra trùng mã lớp học

IF NOT EXISTS (

SELECT 1

FROM LopHoc lh

WHERE lh.MaLopHoc = @MaLopHoc

)

BEGIN

RAISERROR ('Mã lớp học %d không tồn tại.', 16, 1, @MaLopHoc);

RETURN;

END

-- Kiểm tra điều kiện hợp lệ cho thứ

IF @Thu < 1 OR @Thu > 7

BEGIN

RAISERROR('Thứ phải từ 1 đến 7.', 16, 1);

RETURN;

END;

-- Kiểm tra điều kiện hợp lệ cho tiết học

IF @TietBatDau < 1 OR @TietBatDau > 12

OR @TietKetThuc < 1 OR @TietKetThuc > 12

BEGIN

RAISERROR('Tiết bắt đầu và kết thúc phải từ 1 đến 12.', 16, 1);

RETURN;

END;

IF @TietBatDau > @TietKetThuc

BEGIN

RAISERROR('Tiết bắt đầu không được lớn hơn tiết kết thúc.', 16, 1);

RETURN;

END;

-- Kiểm tra trùng lịch học với lớp học khác

DECLARE @MaLopHocTrung INT;

SELECT @MaLopHocTrung = lh.MaLopHoc

FROM LopHoc lh

WHERE lh.MaLopHoc != @MaLopHoc AND lh.MaPhongHoc = @MaPhongHoc AND lh.Thu = @Thu AND

(

(@TietKetThuc BETWEEN lh.TietBatDau AND lh.TietKetThuc)OR

(@TietBatDau BETWEEN lh.TietBatDau AND lh.TietKetThuc)

)

IF @MaLopHocTrung IS NOT NULL

BEGIN

RAISERROR ('Lớp học %d cần sửa bị trùng lịch với lớp học %d cùng phòng.', 16, 1, @MaLopHoc, @MaLopHocTrung);

RETURN;

END

-- Cập nhật lớp học

UPDATE LopHoc

SET

Thu = @Thu,

TietBatDau = @TietBatDau,

TietKetThuc = @TietKetThuc,

MaPhongHoc = @MaPhongHoc,

MaGiangVien = @MaGiangVien,

MaMonHoc = @MaMonHoc

WHERE MaLopHoc = @MaLopHoc;

PRINT N'Lớp học ' + CAST(@MaLopHoc AS NVARCHAR) + N' đã được sửa thành công.';

END TRY

BEGIN CATCH

-- Bắt thông báo lỗi từ trigger

DECLARE @ErrorMessage NVARCHAR(4000) = ERROR\_MESSAGE();

RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END;

Code C#:

/// <summary>

/// Sửa thông tin của một lớp học trong cơ sở dữ liệu thông qua stored procedure "proc\_SuaLopHoc".

/// </summary>

/// <param name="maLopHoc">

/// Mã lớp học của lớp học cần sửa.

/// </param>

/// <param name="thu">

/// Ngày trong tuần của lớp học cần sửa (1-7).

/// </param>

/// <param name="tietBatDau">

/// Tiết bắt đầu của lớp học cần sửa.

/// </param>

/// <param name="tietKetThuc">

/// Tiết kết thúc của lớp học cần sửa.

/// </param>

/// <param name="maPhongHoc">

/// Mã phòng học của lớp học cần sửa.

/// </param>

/// <param name="maGiangVien">

/// Mã giảng viên phụ trách lớp học cần sửa.

/// </param>

/// <param name="maMonHoc">

/// Mã môn học của lớp học cần sửa.

/// </param>

/// <returns>

/// Trả về số dòng bị ảnh hưởng sau khi thực thi stored procedure.

/// <br>- Nếu trả về giá trị > 0, nghĩa là sửa lớp học thành công.</br>

/// <br>- Nếu trả về 0, nghĩa là không có bản ghi nào được sửa.</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc thực thi câu lệnh SQL.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi không xác định khác.

/// </exception>

public int SuaLopHoc(int maLopHoc, byte thu, byte tietBatDau, byte tietKetThuc, int maPhongHoc, int maGiangVien, int maMonHoc)

{

    try

    {

        string connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_SuaLopHoc", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.AddWithValue("@MaLopHoc", maLopHoc);

                command.Parameters.AddWithValue("@Thu", thu);

                command.Parameters.AddWithValue("@TietBatDau", tietBatDau);

                command.Parameters.AddWithValue("@TietKetThuc", tietKetThuc);

                command.Parameters.AddWithValue("@MaPhongHoc", maPhongHoc);

                command.Parameters.AddWithValue("@MaGiangVien", maGiangVien);

                command.Parameters.AddWithValue("@MaMonHoc", maMonHoc);

                return command.ExecuteNonQuery();

            }

        }

    }

    catch (SqlException)

    {

        throw; // Ném lại ngoại lệ SqlException nếu có lỗi xảy ra

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}"); // Ném lại ngoại lệ khác

    }

}

### Đăng ký sinh viên vào lớp học

Code SQL:

CREATE PROCEDURE proc\_DangKySinhVienVaoLop

@MaSinhVien INT, -- Input: Mã sinh viên

@MaLopHoc INT -- Input: Mã lớp học

AS

BEGIN

BEGIN TRY

-- Thực hiện đăng ký sinh viên vào lớp học

INSERT INTO DangKy (MaSinhVien, MaLopHoc)

VALUES (@MaSinhVien, @MaLopHoc);

-- Thông báo thành công

PRINT N'Sinh viên ' + CAST(@MaSinhVien AS NVARCHAR) + N' đã đăng ký vào lớp học ' + CAST(@MaLopHoc AS NVARCHAR) + N' thành công.';

END TRY

BEGIN CATCH

-- Nếu có lỗi từ trigger, hiển thị thông báo lỗi

DECLARE @ErrorMessage NVARCHAR(4000) = ERROR\_MESSAGE();

RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END;

Code C#:

/// <summary>

/// Đăng ký sinh viên vào lớp học bằng cách gọi thủ tục proc\_DangKySinhVienVaoLop.

/// </summary>

/// <param name="maSinhVien">Mã sinh viên cần đăng ký.</param>

/// <param name="maLopHoc">Mã lớp học mà sinh viên muốn đăng ký.</param>

/// <returns>

/// Trả về true nếu đăng ký thành công, false nếu đã đăng ký hoặc xảy ra lỗi.

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public bool DangKySinhVienVaoLop(int maSinhVien, int maLopHoc)

{

    try

    {

        string connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_DangKySinhVienVaoLop", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                // Thêm tham số vào thủ tục

                command.Parameters.AddWithValue("@MaSinhVien", maSinhVien);

                command.Parameters.AddWithValue("@MaLopHoc", maLopHoc);

                // Thực thi lệnh

                command.ExecuteNonQuery();

                return true;

            }

        }

    }

    catch (SqlException ex)

    {

        // Nếu gặp lỗi, có thể kiểm tra xem lỗi do sinh viên đã đăng ký lớp học này trước đó hay không

        if (ex.Message.Contains("Sinh viên đã đăng ký lớp học này"))

        {

            return false;

        }

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Cập nhật điểm sinh viên

Code SQL:

CREATE PROCEDURE proc\_CapNhatDiemSinhVien

@MaSinhVien INT, -- Mã số sinh viên cần cập nhật điểm

@MaLopHoc INT, -- Mã môn học mà sinh viên đăng ký

@DiemQuaTrinh DECIMAL(4,2), -- Điểm quá trình cần cập nhật

@DiemCuoiKy DECIMAL(4,2) -- Điểm cuối kỳ cần cập nhật

AS

BEGIN

BEGIN TRY

-- Kiểm tra xem sinh viên có tồn tại không

IF NOT EXISTS (

SELECT 1

FROM SinhVien sv

WHERE sv.MaSinhVien = @MaSinhVien

)

BEGIN

RAISERROR(N'Không tồn tại mã sinh viên %d trong thông tin đăng ký cần sửa.', 16, 1, @MaSinhVien);

RETURN;

END

-- Kiểm tra lớp học có tồn tại không

IF NOT EXISTS (

SELECT 1

FROM LopHoc lh

WHERE lh.MaLopHoc = @MaLopHoc

)

BEGIN

RAISERROR ('Không tồn tại mã lớp học %d trong thông tin đăng ký cần sửa.', 16, 1, @MaLopHoc);

RETURN;

END

-- Kiểm tra sinh viên và lớp học có tồn tại trong bảng DangKy hay không

IF NOT EXISTS (SELECT 1 FROM DangKy where MaLopHoc = @MaLopHoc and MaSinhVien = @MaSinhVien)

BEGIN

-- Thông báo sinh viên chưa đăng ký lớp học

RAISERROR(N'Sinh viên chưa đăng ký lớp học này.', 16, 1);

RETURN;

END

-- Kiểm tra điểm hợp lệ

IF @DiemQuaTrinh < 0.0 OR

@DiemQuaTrinh > 10.0 OR

@DiemCuoiKy < 0.0 OR

@DiemCuoiKy > 10.0

BEGIN

RAISERROR(N'Điểm cần cập nhật không hợp lệ.', 16, 1);

RETURN;

END

-- Cập nhật điểm quá trình và điểm cuối kỳ cho sinh viên

UPDATE DangKy

SET DiemQuaTrinh = @DiemQuaTrinh,

DiemCuoiKy = @DiemCuoiKy

WHERE MaLopHoc = @MaLopHoc and MaSinhVien = @MaSinhVien

PRINT N'Cập nhật điểm thành công.'

END TRY

BEGIN CATCH

-- Bắt thông báo lỗi từ trigger

DECLARE @ErrorMessage NVARCHAR(4000) = ERROR\_MESSAGE();

RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END;

Code C#:

/// <summary>

/// Cập nhật điểm quá trình và điểm cuối kỳ cho sinh viên trong lớp học.

/// </summary>

/// <param name="maSV">Mã số sinh viên cần cập nhật điểm.</param>

/// <param name="maLH">Mã lớp học mà sinh viên đã đăng ký.</param>

/// <param name="diemGK">Điểm quá trình (giữa kỳ) cần cập nhật.</param>

/// <param name="diemCK">Điểm cuối kỳ cần cập nhật.</param>

/// <returns>

/// Trả về số lượng dòng bị ảnh hưởng (nếu cập nhật thành công).

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc thực thi câu lệnh SQL.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi chung khác.

/// </exception>

public int CapNhatDiem(int maSV, int maLH, decimal diemGK, decimal diemCK)

{

    try

    {

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_CapNhatDiemSinhVien", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.Add(new SqlParameter("@MaSinhVien", maSV));

                command.Parameters.Add(new SqlParameter("@MaLopHoc", maLH));

                command.Parameters.Add(new SqlParameter("@DiemQuaTrinh", diemGK));

                command.Parameters.Add(new SqlParameter("@DiemCuoiKy", diemCK));

                return command.ExecuteNonQuery();

            }

        }

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception(ex.Message);

    }

}

### Xem kết quả học của các sinh viên tại một lớp

Code SQL:

CREATE PROCEDURE proc\_XemKetQuaHocTapCuaLop

@MaLopHoc INT

AS

BEGIN

-- Input:

-- Mã lớp học cần xem danh sách sinh viên và xếp loại

-- Output:

-- Trả về danh sách sinh viên trong lớp học có mã @MaLopHoc. Bao gồm:

-- - MaSinhVien: Mã sinh viên

-- - HoVaTen: Họ và tên của sinh viên

-- - DiemQuaTrinh: Điểm quá trình của sinh viên

-- - DiemCuoiKy: Điểm cuối kỳ của sinh viên

-- - DiemTrungBinhMon: Điểm trung bình môn (trung bình cộng của điểm quá trình và điểm cuối kỳ)

-- - XepLoai: Xếp loại học lực của sinh viên dựa trên điểm trung bình môn

-- + 'Giỏi' nếu DiemTrungBinhMon >= 8.0

-- + 'Khá' nếu DiemTrungBinhMon từ 6.5 đến dưới 8.0

-- + 'Trung Bình' nếu DiemTrungBinhMon từ 5.0 đến dưới 6.5

-- + 'Yếu' nếu DiemTrungBinhMon dưới 5.0

-- + 'Chưa biết' nếu DiemTrungBinhMon chưa có do thiếu điểm

-- - TrangThaiQuaMon: Trạng thái qua môn của sinh viên

-- + 'QUA' nếu DiemTrungBinhMon >= 5.0 và DiemCuoiKy >= 3.0

-- + 'RỚT' nếu không đạt điều kiện qua môn

-- + 'Chưa biết' nếu DiemTrungBinhMon chưa xác định do thiếu điểm

SELECT MaSinhVien,

HoVaTen,

DiemQuaTrinh,

DiemCuoiKy,

DiemTrungBinhMon,

CASE

WHEN DiemTrungBinhMon IS NULL THEN N'Chưa biết'

WHEN DiemTrungBinhMon >= 8.0 THEN N'Giỏi'

WHEN DiemTrungBinhMon < 8.0 AND DiemTrungBinhMon >= 6.5 THEN N'Khá'

WHEN DiemTrungBinhMon < 6.5 AND DiemTrungBinhMon >= 5.0 THEN N'Trung Bình'

ELSE N'Yếu'

END AS XepLoai,

CASE

WHEN DiemTrungBinhMon IS NULL THEN N'Chưa biết'

WHEN DiemTrungBinhMon >= 5.0 AND DiemCuoiKy >= 3.0 THEN N'QUA'

ELSE N'RỚT'

END AS TrangThaiQuaMon

FROM (

SELECT sv.MaSinhVien,

sv.HoVaTen,

dk.DiemQuaTrinh,

dk.DiemCuoiKy,

CASE

WHEN dk.DiemQuaTrinh IS NULL OR dk.DiemCuoiKy IS NULL THEN NULL

ELSE (dk.DiemQuaTrinh + dk.DiemCuoiKy) / 2

END AS DiemTrungBinhMon

FROM SinhVien sv

INNER JOIN DangKy dk ON sv.MaSinhVien = dk.MaSinhVien

INNER JOIN LopHoc lh ON dk.MaLopHoc = lh.MaLopHoc

WHERE lh.MaLopHoc = @MaLopHoc

) AS Subquery;

END;

Code C#:

/// <summary>

/// Gọi thủ tục proc\_XemKetQuaHocTapCuaLop để lấy danh sách kết quả học tập môn học được dạy tại lớp học, kết quả này là bảng điểm của sinh viên với môn học được dạy.

/// và xếp loại học lực dựa trên điểm trung bình môn.

/// </summary>

/// <param name="maLopHoc">Mã lớp học cần xem danh sách sinh viên</param>

/// <returns>

/// Danh sách các sinh viên bao gồm:

/// <br>- MaSinhVien: Mã sinh viên.</br>

/// <br>- HoVaTen: Họ và tên sinh viên.</br>

/// <br>- DiemQuaTrinh: Điểm quá trình.</br>

/// <br>- DiemCuoiKy: Điểm cuối kỳ.</br>

/// <br>- DiemTrungBinhMon: Điểm trung bình môn (trung bình cộng của điểm quá trình và điểm cuối kỳ).</br>

/// <br>- XepLoai: Xếp loại học lực (Giỏi, Khá, Trung Bình, Yếu, hoặc Chưa biết nếu thiếu điểm).</br>

/// <br>- TrangThaiQuaMon: Trạng thái qua môn (QUA, RỚT hoặc Chưa biết nếu thiếu điểm).</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public DataTable XemKetQuaHocTapCuaLop(int maLopHoc)

{

    try

    {

        DataTable ketQuaXepLoaiDiem = new DataTable();

        String connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_XemKetQuaHocTapCuaLop", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.AddWithValue("@MaLopHoc", maLopHoc);

                using (SqlDataReader reader = command.ExecuteReader())

                {

                    ketQuaXepLoaiDiem.Load(reader);

                    return ketQuaXepLoaiDiem;

                }

            }

        }

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Xem thời khóa biểu một sinh viên

Code SQL:

CREATE PROCEDURE proc\_XemThoiKhoaBieuSinhVien

@MaSinhVien INT

AS

BEGIN

-- Xem thời khóa biểu của một sinh viên.

-- Input: Mã sinh viên cần xem thời khóa biểu

-- Output:

-- Một bảng có các thuộc tính sau:

-- Thu: thứ của lớp học.

-- TietBatDau: Tiết bắt đầu.

-- TietKetThuc: Tiết kết thúc.

-- TenMonHoc: Tên môn học được dạy tại lớp.

-- SoTinChi: Số tín chỉ của môn học.

IF NOT EXISTS (SELECT 1 FROM SinhVien WHERE MaSinhVien = @MaSinhVien)

BEGIN

RAISERROR('Sinh viên không tồn tại.', 16, 1)

RETURN

END

-- Truy vấn thông tin thời khóa biểu của sinh viên

SELECT

lh.Thu,

lh.TietBatDau,

lh.TietKetThuc,

mh.TenMonHoc,

mh.SoTinChi,

gv.HoVaTen AS TenGiangVien,

ph.MaPhongHoc

FROM

DangKy dk

JOIN LopHoc lh ON dk.MaLopHoc = lh.MaLopHoc

JOIN MonHoc mh ON lh.MaMonHoc = mh.MaMonHoc

JOIN GiangVien gv ON lh.MaGiangVien = gv.MaGiangVien

JOIN PhongHoc ph ON lh.MaPhongHoc = ph.MaPhongHoc

WHERE

dk.MaSinhVien = @MaSinhVien

ORDER BY

lh.Thu, lh.TietBatDau;

END;

Code C#:

/// <summary>

/// Gọi thủ tục proc\_XemThoiKhoaBieuSinhVien để lấy thời khóa biểu của sinh viên.

/// </summary>

/// <param name="maSinhVien">Mã sinh viên cần xem thời khóa biểu.</param>

/// <returns>

/// Danh sách các sinh viên bao gồm:

/// <br>- Thu: Mã sinh viên.</br>

/// <br>- TietBatDau: Họ và tên sinh viên.</br>

/// <br>- TietKetThuc: Điểm quá trình.</br>

/// <br>- TenMonHoc: Điểm cuối kỳ.</br>

/// <br>- SoTinChi: Điểm trung bình môn (trung bình cộng của điểm quá trình và điểm cuối kỳ).</br>

/// <br>- TenGiangVien: Xếp loại học lực (Giỏi, Khá, Trung Bình, Yếu, hoặc Chưa biết nếu thiếu điểm).</br>

/// <br>- MaPhongHoc: Trạng thái qua môn (QUA, RỚT hoặc Chưa biết nếu thiếu điểm).</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public DataTable XemThoiKhoaBieu1SV(int maSinhVien)

{

    try

    {

        DataTable thoiKhoaBieu = new DataTable();

        String connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_XemThoiKhoaBieuSinhVien", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.AddWithValue("@MaSinhVien", maSinhVien);

                using (SqlDataReader reader = command.ExecuteReader())

                {

                   thoiKhoaBieu.Load(reader);

                    return thoiKhoaBieu;

                }

            }

        }

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Giảng viên dạy môn học với nhiều sinh viên nhất

Code SQL:

CREATE PROCEDURE proc\_GiangVienDayMonHocVoiSoSVDangKyCaoNhat

AS

BEGIN

-- Tìm thông tin giảng viên, môn học và số lượng sinh viên đăng ký

SELECT

gv.MaGiangVien,

gv.HoVaTen AS TenGiangVien,

mh.TenMonHoc,

COUNT(dk.MaSinhVien) AS SoLuongSinhVien

FROM GiangVien gv

INNER JOIN LopHoc lh ON gv.MaGiangVien = lh.MaGiangVien

INNER JOIN MonHoc mh ON lh.MaMonHoc = mh.MaMonHoc

INNER JOIN DangKy dk ON lh.MaLopHoc = dk.MaLopHoc

GROUP BY gv.MaGiangVien, gv.HoVaTen, mh.TenMonHoc

HAVING COUNT(dk.MaSinhVien) = (

-- Truy vấn con để lấy số lượng sinh viên đăng ký cao nhất

SELECT MAX(SoLuongSinhVien)

FROM (

SELECT

COUNT(dk.MaSinhVien) AS SoLuongSinhVien

FROM GiangVien gv

INNER JOIN LopHoc lh ON gv.MaGiangVien = lh.MaGiangVien

INNER JOIN MonHoc mh ON lh.MaMonHoc = mh.MaMonHoc

INNER JOIN DangKy dk ON lh.MaLopHoc = dk.MaLopHoc

GROUP BY gv.MaGiangVien, gv.HoVaTen, mh.TenMonHoc

) AS SoLuongMax

);

END;

Code C#:

/// <summary>

/// Gọi thủ tục proc\_GiangVienDayMonHocVoiSoSVDangKyCaoNhat để lấy danh sách giảng viên dạy môn học với số lượng sinh viên đăng ký cao nhất.

/// </summary>

/// <returns>

/// Danh sách giảng viên dạy môn học có số lượng sinh viên đăng ký cao nhất, bao gồm:

/// <br>- MaGiangVien: Mã giảng viên.</br>

/// <br>- TenGiangVien: Tên giảng viên.</br>

/// <br>- TenMonHoc: Tên môn học giảng viên dạy.</br>

/// <br>- SoLuongSinhVien: Số lượng sinh viên đăng ký môn học.</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public DataTable GetGiangVienDayMonHocVoiSoSVDangKyCaoNhat()

{

    try

    {

        DataTable result = new DataTable();

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            using (SqlCommand cmd = new SqlCommand("proc\_GiangVienDayMonHocVoiSoSVDangKyCaoNhat", connection))

            {

                cmd.CommandType = CommandType.StoredProcedure;

                using (SqlDataAdapter adapter = new SqlDataAdapter(cmd))

                {

                    adapter.Fill(result);

                }

            }

        }

        return result;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

## Hàm

### Kiểm tra một sinh viên có qua môn

Code SQL:

CREATE FUNCTION dbo.fn\_KiemTraQuaMon (

@MaSinhVien INT,

@MaMonHoc INT

)

RETURNS TINYINT

AS

BEGIN

-- Input:

-- @MaSinhVien: Mã sinh viên cần kiểm tra

-- @MaMonHoc: Mã môn học của sinh viên cần kiểm tra

-- Output:

-- 1 nếu qua môn, DiemCuoiKy >= 3.0 và điểm trung bình môn (DiemQuaTrinh + DiemCuoiKy) / 2 >= 5.0

-- 0 nếu rớt môn, DiemCuoiKy < 3.0 hoặc điểm trung bình môn < 5.0

-- 2 chưa nhập đủ điểm

-- 3 sinh viên không tồn tại

-- 4 sinh viên chưa đăng ký môn cần kiểm tra

-- Kiểm tra xem sinh viên có tồn tại trong bảng DangKy không

IF NOT EXISTS (SELECT 1 FROM DangKy dk WHERE dk.MaSinhVien = @MaSinhVien)

BEGIN

RETURN 3

END

-- Kiểm tra xem sinh viên có học môn này không

IF NOT EXISTS (SELECT 1

FROM DangKy dk

INNER JOIN LopHoc lh ON dk.MaLopHoc = lh.MaLopHoc

WHERE dk.MaSinhVien = @MaSinhVien AND lh.MaMonHoc = @MaMonHoc)

BEGIN

RETURN 4

END

DECLARE @DiemQuaTrinh DECIMAL(4, 2), @DiemCuoiKy DECIMAL(4, 2), @DiemTrungBinhMon DECIMAL(4, 2)

SELECT @DiemQuaTrinh = dk.DiemQuaTrinh, @DiemCuoiKy = dk.DiemCuoiKy

FROM DangKy dk

INNER JOIN LopHoc lh ON dk.MaLopHoc = lh.MaLopHoc

WHERE dk.MaSinhVien = @MaSinhVien AND lh.MaMonHoc = @MaMonHoc

IF @DiemQuaTrinh IS NULL OR @DiemCuoiKy IS NULL

BEGIN

RETURN 2

END

SET @DiemTrungBinhMon = (@DiemQuaTrinh + @DiemCuoiKy) / 2.0

IF @DiemCuoiKy >= 3.0 AND @DiemTrungBinhMon >= 5.0

BEGIN

RETURN 1

END

RETURN 0

END;

Code C#:

/// <summary>

/// Kiểm tra tình trạng qua môn học của sinh viên dựa trên mã sinh viên và mã môn học.

/// </summary>

/// <param name="maSinhVien">Mã sinh viên cần kiểm tra.</param>

/// <param name="maMonHoc">Mã môn học của sinh viên cần kiểm tra.</param>

/// <returns>Trả về tình trạng qua môn dưới dạng byte: 1 (QUA), 0 (RỚT), 2 (Chưa nhập đủ điểm), 3 (Sinh viên không tồn tại), hoặc 4 (Sinh viên chưa đăng ký môn học).</returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public byte KiemTraQuaMon(int maSinhVien, int maMonHoc)

{

    try

    {

        byte ketQua; // Biến để lưu kết quả trả về từ hàm SQL

        string connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            string query = "SELECT dbo.fn\_KiemTraQuaMon(@MaSinhVien, @MaMonHoc)";

            using (SqlCommand command = new SqlCommand(query, connection))

            {

                command.Parameters.AddWithValue("@MaSinhVien", maSinhVien);

                command.Parameters.AddWithValue("@MaMonHoc", maMonHoc);

                ketQua = (byte)command.ExecuteScalar();

            }

        }

        return ketQua;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Xem học lực một sinh viên

Code SQL:

CREATE FUNCTION dbo.fn\_XemHocLucSinhVien (

@MaSinhVien INT

)

RETURNS NVARCHAR(50)

AS

BEGIN

-- Input:

-- @MaSinhVien: Mã sinh viên cần xem học lực

-- Output:

-- Trả về 1 chuỗi kí tự là học lực của sinh viên, bao gồm: Giỏi, Khá, Trung Bình, Yếu.

DECLARE @DiemTrungBinh DECIMAL(4, 2);

-- Tính điểm trung bình của tất cả các môn mà sinh viên đã đăng ký

SELECT @DiemTrungBinh = AVG((dk.DiemQuaTrinh + dk.DiemCuoiKy) / 2.0)

FROM DangKy dk

WHERE dk.MaSinhVien = @MaSinhVien;

-- Trả về học lực dựa vào điểm trung bình

RETURN (

CASE

WHEN @DiemTrungBinh >= 8.0 THEN N'Giỏi'

WHEN @DiemTrungBinh >= 6.5 THEN N'Khá'

WHEN @DiemTrungBinh >= 5.0 THEN N'Trung Bình'

ELSE N'Yếu'

END

);

END;

Code C#:

/// <summary>

/// Xem học lực của một sinh viên dựa vào điểm trung bình các môn đã đăng ký.

/// </summary>

/// <param name="maSinhVien">Mã sinh viên cần xem học lực.</param>

/// <returns>

/// Trả về chuỗi học lực của sinh viên:

/// <br>- Giỏi</br>

/// <br>- Khá</br>

/// <br>- Trung Bình</br>

/// <br>- Yếu</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public string XemHocLucSinhVien(int maSinhVien)

{

    try

    {

        string hocLuc = string.Empty;

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            string query = @"SELECT dbo.fn\_XemHocLucSinhVien(@MaSinhVien) AS HocLuc";

            using (SqlCommand cmd = new SqlCommand(query, connection))

            {

                cmd.Parameters.AddWithValue("@MaSinhVien", maSinhVien);

                object result = cmd.ExecuteScalar();

                if (result != null)

                {

                    hocLuc = result.ToString();

                }

            }

        }

        return hocLuc;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Tính số tín chỉ đã hoàn thành của một sinh viên (thieu comment)

Code SQL:

CREATE FUNCTION dbo.fn\_SoTinChiDaHoanThanh(@MaSinhVien INT)

RETURNS INT

AS

BEGIN

-- Input:

-- @MaSinhVien: Mã sinh viên cần tính số tính chỉ

-- Output:

-- Trả về số tín chỉ đã hoàn thành của sinh viên có mã @MaSinhVien

DECLARE @SoTCHoanThanh INT;

SELECT @SoTCHoanThanh = SUM(MH.SoTinChi)

FROM

DangKy DK

INNER JOIN LopHoc LH ON DK.MaLopHoc = LH.MaLopHoc

INNER JOIN MonHoc MH ON LH.MaMonHoc = MH.MaMonHoc

WHERE

DK.MaSinhVien = @MaSinhVien

AND DiemQuaTrinh >= 3

AND (DiemQuaTrinh + DiemCuoiKy)/2 >= 5

RETURN ISNULL(@SoTCHoanThanh, 0)

END;

Code C#:

/// <summary>

/// Hàm lấy số tín chỉ mà sinh viên đã hoàn thành dựa trên mã sinh viên.

/// </summary>

/// <param name="maSV">Mã sinh viên cần tính số tín chỉ hoàn thành.</param>

/// <returns>Trả về tổng số tín chỉ đã hoàn thành. Nếu có lỗi xảy ra, trả về 0.</returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi không xác định xảy ra.

/// </exception>

public int SoTinChiHoanThanh(int maSV)

{

    try

    {

        int soTinChi = 0;

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            string querry = "select dbo.fn\_SoTinChiDaHoanThanh(" + maSV + ")";

            using (SqlCommand command = new SqlCommand(querry, connection))

            {

                soTinChi = (int)command.ExecuteScalar();

            }

            connection.Close();

        }

        return soTinChi;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Đếm số lượng sinh viên của một lớp học

Code SQL:

CREATE FUNCTION dbo.fn\_DemSoLuongSinhVienCuaLop(@MaLopHoc INT)

RETURNS INT

AS

BEGIN

-- Input:

-- @MaLopHoc: Mã lớp học cần đếm số lượng sinh viên

-- Output:

-- Trả về số lượng sinh viên đã đăng ký vào lớp học có mã @MaLopHoc

DECLARE @SoLuongSinhVien INT;

SELECT @SoLuongSinhVien = COUNT(dk.MaSinhVien)

FROM DangKy dk

WHERE dk.MaLopHoc = @MaLopHoc;

RETURN @SoLuongSinhVien;

END;

Code C#:

/// <summary>

/// Trả về số lượng sinh viên của một lớp học.

/// </summary>

/// <param name="maLopHoc">Mã lớp học của lớp cần được đếm sinh viên.</param>

/// <returns>Sô lượng sinh viên của lớp cần đếm.</returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public int DemSoLuongSinhVienCuaLop(int maLopHoc)

{

    try

    {

        int soLuongSinhVien = 0;

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            using (SqlCommand cmd = new SqlCommand("SELECT dbo.fn\_DemSoLuongSinhVienCuaLop(@MaLopHoc)", connection))

            {

                cmd.Parameters.AddWithValue("@MaLopHoc", maLopHoc);

                soLuongSinhVien = (int)cmd.ExecuteScalar();

            }

        }

        return soLuongSinhVien;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Đếm số lớp học một giảng viên đang phụ trách

Code SQL:

CREATE FUNCTION dbo.fn\_DemSoLopGiangVienPhuTrach(@MaGiangVien INT)

RETURNS INT

AS

BEGIN

-- Input:

-- @MaGiangVien: Mã giảng viên cần đếm số lượng lớp học mà họ phụ trách.

-- Output:

-- Trả về số lượng lớp học mà giảng viên có mã @MaGiangVien phụ trách.

DECLARE @SoLuongLopHoc INT;

SELECT @SoLuongLopHoc = COUNT(lh.MaLopHoc)

FROM LopHoc lh

WHERE lh.MaGiangVien = @MaGiangVien;

RETURN @SoLuongLopHoc;

END;

Code C#:

/// <summary>

/// Đếm số lớp một giảng viên đang phụ trách.

/// </summary>

/// <param name="maGiangVien">Mã giảng viên cần kiểm tra.</param>

/// <returns>Sô lượng lớp mà giảng viên phụ trách.</returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public int DemSoLopGiangVienPhuTrach(int maGiangVien)

{

    try

    {

        int soLopGiangVienPhuTrach = 0;

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            using (SqlCommand cmd = new SqlCommand("SELECT dbo.fn\_DemSoLopGiangVienPhuTrach(@MaGiangVien)", connection))

            {

                cmd.Parameters.AddWithValue("@MaGiangVien", maGiangVien);

                soLopGiangVienPhuTrach = (int)cmd.ExecuteScalar();

            }

        }

        return soLopGiangVienPhuTrach;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Tính phần trăm qua môn trong một lớp học

Code SQL:

CREATE FUNCTION dbo.fn\_TinhPhanTramQuaMon(@MaLopHoc INT)

RETURNS DECIMAL(5, 2)

AS

BEGIN

-- Input:

-- @MaLopHoc: Mã lớp học cần tính

-- Output:

-- Phần trăm sinh viên qua môn, nhỏ nhất là 0 và lớn nhất là 100

DECLARE @SoLuongSinhVien INT;

DECLARE @SoLuongQua INT;

-- Đếm số lượng sinh viên đã đăng ký vào lớp

SELECT @SoLuongSinhVien = COUNT(\*)

FROM DangKy

WHERE MaLopHoc = @MaLopHoc;

-- Đếm số lượng sinh viên qua môn

SELECT @SoLuongQua = COUNT(\*)

FROM DangKy dk

INNER JOIN LopHoc lh ON dk.MaLopHoc = lh.MaLopHoc

WHERE dk.MaLopHoc = @MaLopHoc AND dbo.fn\_KiemTraQuaMon(dk.MaSinhVien, lh.MaMonHoc) = 1;

-- Kiểm tra tránh chia cho 0

IF @SoLuongSinhVien = 0

BEGIN

RETURN 0.00; -- Nếu không có sinh viên, trả về 0%

END

-- Tính phần trăm qua môn

RETURN (@SoLuongQua \* 100.0 / @SoLuongSinhVien);

END;

Code C#:

/// <summary>

/// Tính phần trăm sinh viên qua môn trong một lớp học.

/// </summary>

/// <param name="maLopHoc">Mã lớp học cần tính phần trăm qua môn.</param>

/// <returns>

/// Trả về phần trăm qua môn (decimal, từ 0 đến 100) của lớp học.

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public decimal TinhPhanTramQuaMon(int maLopHoc)

{

    try

    {

        decimal phanTramQuaMon = 0;

        string connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            string query = "SELECT dbo.fn\_TinhPhanTramQuaMon(@MaLopHoc)";

            using (SqlCommand cmd = new SqlCommand(query, connection))

            {

                cmd.Parameters.AddWithValue("@MaLopHoc", maLopHoc);

                object result = cmd.ExecuteScalar();

                // Kiểm tra giá trị trả về không null

                if (result != DBNull.Value)

                {

                    phanTramQuaMon = Convert.ToDecimal(result);

                }

            }

        }

        return phanTramQuaMon;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Lấy danh sách sinh viên với điểm trong một lớp học

Code SQL:

CREATE FUNCTION dbo.fn\_LayDanhSachSinhVienTrongLop (

@MaLopHoc INT

)

RETURNS TABLE

AS

RETURN

(

-- Input:

-- @MaLopHoc: Mã lớp học cần lấy danh sách.

-- Output:

-- Trả về danh sách sinh viên đăng ký trong lớp học có mã @MaLopHoc. Bao gồm:

-- - MaSinhVien: Mã sinh viên

-- - HoVaTen: Họ và tên của sinh viên

-- - Email: Email của sinh viên

-- - SoDienThoai: Số điện thoại của sinh viên

-- - QueQuan: Quê của sinh viên

-- - DiemQuaTrinh: Điểm quá trình của sinh viên

-- - DiemCuoiKy: Điểm cuối kỳ của sinh viên

SELECT

sv.MaSinhVien,

sv.HoVaTen,

sv.Email,

sv.SoDienThoai,

sv.QueQuan,

dk.DiemQuaTrinh,

dk.DiemCuoiKy

FROM DangKy dk

INNER JOIN SinhVien sv ON dk.MaSinhVien = sv.MaSinhVien

WHERE dk.MaLopHoc = @MaLopHoc

);

Code C#:

/// <summary>

/// Lấy danh sách sinh viên trong lớp học có mã lớp học được chỉ định.

/// </summary>

/// <param name="maLopHoc">Mã lớp học cần lấy danh sách sinh viên.</param>

/// <returns>

/// Trả về một DataTable chứa danh sách sinh viên trong lớp, bao gồm các cột:

/// <br>- MaSinhVien: Mã sinh viên.</br>

/// <br>- HoVaTen: Họ và tên sinh viên.</br>

/// <br>- Email: Địa chỉ email.</br>

/// <br>- SoDienThoai: Số điện thoại.</br>

/// <br>- QueQuan: Quê quán của sinh viên.</br>

/// <br>- DiemQuaTrinh: Điểm quá trình của sinh viên.</br>

/// <br>- Điểm cuối kỳ: Điểm cuối kỳ của sinh viên.</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public DataTable LayDanhSachSinhVienTrongLop(int maLopHoc)

{

    try

    {

        DataTable danhSachSinhVien = new DataTable();

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            string query = @"

            SELECT

                sv.MaSinhVien,

                sv.HoVaTen,

                sv.Email,

                sv.SoDienThoai,

                sv.QueQuan,

                sv.DiemQuaTrinh,

                sv.DiemCuoiKy

            FROM

                dbo.fn\_LayDanhSachSinhVienTrongLop(@MaLopHoc) AS sv";

            using (SqlCommand cmd = new SqlCommand(query, connection))

            {

                cmd.Parameters.AddWithValue("@MaLopHoc", maLopHoc);

                using (SqlDataAdapter adapter = new SqlDataAdapter(cmd))

                {

                    adapter.Fill(danhSachSinhVien);

                }

            }

        }

        return danhSachSinhVien;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Xem bảng điểm của một sinh viên

Code SQL:

CREATE FUNCTION dbo.fn\_LayBangDiemSinhVien (

@MaSinhVien INT

)

RETURNS @BangDiem TABLE (

MaMonHoc INT,

TenMonHoc NVARCHAR(255),

DiemQuaTrinh DECIMAL(4, 2),

DiemCuoiKy DECIMAL(4, 2),

DiemTrungBinh DECIMAL(4, 2),

XepLoaiMon NVARCHAR(50)

)

AS

BEGIN

-- Input:

-- @MaSinhVien: Mã sinh viên cần lấy bảng điểm.

-- Output:

-- Trả về bảng điểm bao gồm:

-- - MaMonHoc: Mã môn học

-- - TenMonHoc: Tên môn học

-- - DiemQuaTrinh: Điểm quá trình của môn học

-- - DiemCuoiKy: Điểm cuối kỳ của môn học

-- - DiemTrungBinh: Điểm trung bình môn (tính trung bình cộng của điểm quá trình và điểm cuối kỳ)

-- - XepLoaiMon: Xếp loại từng môn dựa trên điểm trung bình

-- Chèn dữ liệu bảng điểm chi tiết từng môn

INSERT INTO @BangDiem

SELECT

mh.MaMonHoc,

mh.TenMonHoc,

dk.DiemQuaTrinh,

dk.DiemCuoiKy,

(dk.DiemQuaTrinh + dk.DiemCuoiKy) / 2.0 AS DiemTrungBinh,

CASE

WHEN (dk.DiemQuaTrinh + dk.DiemCuoiKy) / 2.0 >= 8.0 THEN N'Giỏi'

WHEN (dk.DiemQuaTrinh + dk.DiemCuoiKy) / 2.0 >= 6.5 THEN N'Khá'

WHEN (dk.DiemQuaTrinh + dk.DiemCuoiKy) / 2.0 >= 5.0 THEN N'Trung Bình'

ELSE N'Yếu'

END AS XepLoaiMon

FROM DangKy dk

INNER JOIN LopHoc lh ON dk.MaLopHoc = lh.MaLopHoc

INNER JOIN MonHoc mh ON lh.MaMonHoc = mh.MaMonHoc

WHERE dk.MaSinhVien = @MaSinhVien;

RETURN;

END;

Code C#:

/// <summary>

/// Lấy bảng điểm của sinh viên, bao gồm các môn học, điểm quá trình, điểm cuối kỳ, điểm trung bình môn và xếp loại.

/// </summary>

/// <param name="maSinhVien">Mã sinh viên cần lấy bảng điểm.</param>

/// <returns>

/// Trả về một DataTable chứa bảng điểm của sinh viên, bao gồm:

/// <br>- MaMonHoc: Mã môn học.</br>

/// <br>- TenMonHoc: Tên môn học.</br>

/// <br>- DiemQuaTrinh: Điểm quá trình.</br>

/// <br>- DiemCuoiKy: Điểm cuối kỳ.</br>

/// <br>- DiemTrungBinh: Điểm trung bình môn.</br>

/// <br>- XepLoaiMon: Xếp loại môn học.</br>

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public DataTable LayBangDiemSinhVien(int maSinhVien)

{

    try

    {

        DataTable bangDiem = new DataTable();

        string connectionString = QLDSVCNTTConnection.connectionString;

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            string query = @"

            SELECT

                MaMonHoc,

                TenMonHoc,

                DiemQuaTrinh,

                DiemCuoiKy,

                DiemTrungBinh,

                XepLoaiMon

            FROM dbo.fn\_LayBangDiemSinhVien(@MaSinhVien)";

            using (SqlCommand cmd = new SqlCommand(query, connection))

            {

                cmd.Parameters.AddWithValue("@MaSinhVien", maSinhVien);

                using (SqlDataAdapter adapter = new SqlDataAdapter(cmd))

                {

                    adapter.Fill(bangDiem);

                }

            }

        }

        return bangDiem;

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Tính điểm trung bình một sinh viên

Code SQL:

CREATE FUNCTION dbo.fn\_TinhDiemTrungBinh (

@MaSinhVien INT

)

RETURNS DECIMAL(4, 2)

AS

BEGIN

-- Input:

-- @MaSinhVien: Mã sinh viên cần tính điểm trung bình

-- Output:

-- Điểm trung bình của sinh viên

DECLARE @DiemTrungBinh DECIMAL(4, 2);

-- Tính điểm trung bình học kỳ: Trung bình cộng điểm quá trình và điểm cuối kỳ của tất cả các môn

SELECT @DiemTrungBinh = AVG((dk.DiemQuaTrinh + dk.DiemCuoiKy) / 2.0)

FROM DangKy dk

INNER JOIN LopHoc lh ON dk.MaLopHoc = lh.MaLopHoc

INNER JOIN MonHoc mh ON lh.MaMonHoc = mh.MaMonHoc

WHERE dk.MaSinhVien = @MaSinhVien;

-- Trả về kết quả tính trung bình

RETURN @DiemTrungBinh;

END;

Code C#:

/// <summary>

/// Lấy điểm trung bình của sinh viên dựa trên mã sinh viên.

/// </summary>

/// <param name="maSinhVien">Mã sinh viên cần tính điểm trung bình.</param>

/// <returns>

/// Trả về điểm trung bình (decimal) của sinh viên.

/// </returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public decimal LayDiemTrungBinhSinhVien(int maSinhVien)

{

    try

    {

        decimal diemTrungBinh = 0; // Khởi tạo biến để lưu kết quả

        string connectionString = QLDSVCNTTConnection.connectionString; // Chuỗi kết nối đến SQL Server

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            // Truy vấn để gọi hàm TinhDiemTrungBinh từ SQL Server

            string query = "SELECT dbo.fn\_TinhDiemTrungBinh(@MaSinhVien)";

            using (SqlCommand cmd = new SqlCommand(query, connection))

            {

                // Thêm tham số maSinhVien

                cmd.Parameters.AddWithValue("@MaSinhVien", maSinhVien);

                // Thực thi lệnh và lấy giá trị trả về của function

                object result = cmd.ExecuteScalar();

                // Kiểm tra giá trị trả về không null

                if (result != DBNull.Value)

                {

                    diemTrungBinh = Convert.ToDecimal(result);

                }

            }

        }

        return diemTrungBinh; // Trả về điểm trung bình

    }

    catch (SqlException)

    {

        throw;

    }

    catch (Exception ex)

    {

        throw new Exception($"Error: {ex.Message}");

    }

}

### Tính điểm trung bình tích lũy của một sinh viên

Code SQL:

CREATE FUNCTION dbo.fn\_TinhDiemTrungBinhTichLuy (

@MaSinhVien INT

)

RETURNS DECIMAL(4, 2)

AS

BEGIN

-- Input:

-- @MaSinhVien: Mã sinh viên cần tính điểm trung bình

-- Output:

-- Điểm trung bình của sinh viên

DECLARE @DiemTrungBinhTichLuy DECIMAL(4, 2);

-- Tính điểm trung bình tích lũy: Trung bình cộng điểm quá trình và điểm cuối kỳ của tất cả các môn

SELECT @DiemTrungBinhTichLuy = AVG((dk.DiemQuaTrinh + dk.DiemCuoiKy) / 2.0)

FROM DangKy dk

INNER JOIN LopHoc lh ON dk.MaLopHoc = lh.MaLopHoc

INNER JOIN MonHoc mh ON lh.MaMonHoc = mh.MaMonHoc

WHERE

dk.MaSinhVien = @MaSinhVien

AND dk.DiemQuaTrinh IS NOT NULL

AND dk.DiemCuoiKy IS NOT NULL

AND dk.DiemCuoiKy >= 3.0

AND ((dk.DiemQuaTrinh + dk.DiemCuoiKy) / 2.0) >= 5.0;

-- Trả về điểm trung bình tích lũy

RETURN @DiemTrungBinhTichLuy;

END;

Code C#:

/// <summary>

/// Hàm tính điểm trung bình tích lũy của một sinh viên dựa trên mã sinh viên.

/// </summary>

/// <param name="maSinhVien">Mã sinh viên cần tính điểm trung bình tích lũy.</param>

/// <returns>Trả về điểm trung bình tích lũy của sinh viên. Nếu có lỗi xảy ra, trả về 0.</returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public decimal TinhDiemTrungBinhTichLuy(int maSinhVien)

{

    try

    {

        decimal diemTrungBinhTichLuy = 0; // Khởi tạo giá trị ban đầu

        // Chuỗi kết nối đến cơ sở dữ liệu

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            // Tạo lệnh gọi hàm fn\_TinhDiemTrungBinhTichLuy

            using (SqlCommand cmd = new SqlCommand("SELECT dbo.fn\_TinhDiemTrungBinhTichLuy(@MaSinhVien)", connection))

            {

                // Thêm tham số mã sinh viên

                cmd.Parameters.AddWithValue("@MaSinhVien", maSinhVien);

                // Thực thi lệnh và lấy kết quả trả về

                object result = cmd.ExecuteScalar();

                // Nếu kết quả không null, gán vào biến điểm trung bình tích lũy

                if (result != null && result != DBNull.Value)

                {

                    diemTrungBinhTichLuy = Convert.ToDecimal(result);

                }

            }

        }

        return diemTrungBinhTichLuy;

    }

    catch (SqlException ex)

    {

        // Xử lý lỗi SQL

        throw new Exception($"SQL Error: {ex.Message}");

    }

    catch (Exception ex)

    {

        // Xử lý các lỗi khác

        throw new Exception($"Error: {ex.Message}");

    }

}

### Đếm số lượng lớp một sinh viên đăng ký

Code SQL:

CREATE FUNCTION dbo.fn\_DemSoLuongLopDangKy (

@MaSinhVien INT

)

RETURNS INT

AS

BEGIN

-- Hàm fn\_DemSoLuongLopDangKy

-- Đếm số lượng lớp mà một sinh viên đã đăng ký trong bảng DangKy.

-- Input:

-- @MaSinhVien INT - Mã số của sinh viên cần đếm số lượng lớp đăng ký.

-- Output:

-- INT - Trả về số lượng lớp mà sinh viên đã đăng ký.

DECLARE @SoLuongLop INT;

-- Đếm số lượng lớp mà sinh viên đã đăng ký

SELECT @SoLuongLop = COUNT(\*)

FROM DangKy

WHERE MaSinhVien = @MaSinhVien;

RETURN @SoLuongLop;

END;

Code C#:

/// <summary>

/// Hàm đếm số lượng lớp mà sinh viên đã đăng ký.

/// </summary>

/// <param name="maSinhVien">Mã sinh viên cần đếm số lượng lớp đăng ký.</param>

/// <returns>Trả về số lượng lớp mà sinh viên đã đăng ký. Nếu có lỗi xảy ra, trả về 0.</returns>

/// <exception cref="SqlException">

/// Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.

/// </exception>

/// <exception cref="Exception">

/// Ném ra khi có lỗi khác không xác định xảy ra.

/// </exception>

public int SoLuongLopSVDangKy(int maSinhVien)

{

    try

    {

        int soLuongLop = 0;

        using (SqlConnection connection = new SqlConnection(QLDSVCNTTConnection.connectionString))

        {

            connection.Open();

            using (SqlCommand cmd = new SqlCommand("SELECT dbo.fn\_DemSoLuongLopDangKy(@MaSinhVien)", connection))

            {

                cmd.Parameters.AddWithValue("@MaSinhVien", maSinhVien);

                soLuongLop = (int)cmd.ExecuteScalar();

            }

        }

        return soLuongLop;

    }

    catch (SqlException ex)

    {

        // Xử lý lỗi SQL

        throw new Exception($"SQL Error: {ex.Message}");

    }

    catch (Exception ex)

    {

        // Xử lý các lỗi khác

        throw new Exception($"Error: {ex.Message}");

    }

}

## Transactions

### Chuyển lớp học cho sinh viên

Code SQL:

GO

CREATE PROCEDURE proc\_ChuyenSinhVienSangLopKhac

@MaSinhVien INT, -- Mã sinh viên cần chuyển

@MaLopHocCu INT, -- Mã lớp học hiện tại

@MaLopHocMoi INT -- Mã lớp học đích

AS

BEGIN

BEGIN TRY

-- Bắt đầu transaction

BEGIN TRANSACTION;

-- Kiểm tra sự tồn tại của sinh viên trong lớp cũ

IF NOT EXISTS (SELECT 1 FROM DangKy WHERE MaSinhVien = @MaSinhVien AND MaLopHoc = @MaLopHocCu)

BEGIN

RAISERROR(N'Sinh viên chưa từng đăng ký lớp học %d.', 16, 1, @MaLopHocCu);

ROLLBACK TRANSACTION;

RETURN;

END

-- Kiểm tra lớp mới có dạy giống môn của lớp cũ hay không

IF NOT EXISTS (SELECT 1 FROM LopHoc lh WHERE lh.MaLopHoc = @MaLopHocMoi)

BEGIN

RAISERROR(N'Lớp %d học đích không tồn tại.', 16, 1, @MaLopHocMoi);

ROLLBACK TRANSACTION;

RETURN;

END

-- Kiểm tra lớp mới có dạy giống môn cũ hay không

DECLARE @MaMonCu INT, @MaMonMoi INT;

SELECT @MaMonCu = lh.MaMonHoc

FROM DangKy dk

JOIN LopHoc lh ON dk.MaLopHoc = lh.MaLopHoc

WHERE lh.MaLopHoc = @MaLopHocCu;

SELECT @MaMonMoi = lh.MaMonHoc

FROM DangKy dk

JOIN LopHoc lh ON dk.MaLopHoc = lh.MaLopHoc

WHERE lh.MaLopHoc = @MaLopHocMoi;

IF (@MaMonCu != @MaMonMoi)

BEGIN

RAISERROR(N'Lớp mới dạy khác môn mà lớp cũ dạy.', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

-- Kiểm tra sức chứa của lớp mới và số lượng sinh viên đã đăng ký

DECLARE @SucChua INT, @SoSinhVienDangKy INT;

SELECT @SucChua = ph.SucChua

FROM LopHoc lh

INNER JOIN PhongHoc ph ON lh.MaPhongHoc = ph.MaPhongHoc

WHERE lh.MaLopHoc = @MaLopHocMoi;

SELECT @SoSinhVienDangKy = COUNT(\*)

FROM DangKy

WHERE MaLopHoc = @MaLopHocMoi;

IF @SoSinhVienDangKy >= @SucChua

BEGIN

RAISERROR(N'Lớp học đích đã đạt sức chứa tối đa.', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

-- Kiểm tra nếu sinh viên đã đăng ký lớp học mới

IF (@MaLopHocCu = @MaLopHocMoi)

BEGIN

RAISERROR(N'Sinh viên đã ở trong lớp cũ rồi.', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

UPDATE DangKy

SET MaLopHoc = @MaLopHocMoi

WHERE MaSinhVien = @MaSinhVien AND MaLopHoc = @MaLopHocCu;

-- Commit transaction nếu không có lỗi

COMMIT TRANSACTION;

PRINT N'Chuyển lớp thành công!';

END TRY

BEGIN CATCH

-- Rollback transaction nếu có lỗi

ROLLBACK TRANSACTION;

DECLARE @ErrorMessage NVARCHAR(4000) = ERROR\_MESSAGE();

RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END;

Code C#:

/// <summary>

/// Chuyển sinh viên sang lớp khác bằng cách gọi thủ tục proc\_ChuyenSinhVienSangLopKhac.

/// </summary>

/// <param name="maSinhVien">Mã sinh viên cần chuyển lớp.</param>

/// <param name="maLopHocCu">Mã lớp học hiện tại của sinh viên.</param>

/// <param name="maLopHocMoi">Mã lớp học mới mà sinh viên sẽ được chuyển đến.</param>

/// <returns>Số dòng bị ảnh hưởng trong cơ sở dữ liệu khi thực hiện chuyển lớp.</returns>

/// <exception cref="SqlException">Ném ra khi có lỗi xảy ra trong quá trình kết nối hoặc truy vấn cơ sở dữ liệu.</exception>

/// <exception cref="Exception">Ném ra khi có lỗi không xác định khác.</exception>

public int ChuyenSinhVienSangLopKhac(int maSinhVien, int maLopHocCu, int maLopHocMoi)

{

    try

    {

        string connectionString = QLDSVCNTTConnection.connectionString; // Chuỗi kết nối đến cơ sở dữ liệu

        using (SqlConnection connection = new SqlConnection(connectionString))

        {

            connection.Open();

            using (SqlCommand command = new SqlCommand("proc\_ChuyenSinhVienSangLopKhac", connection))

            {

                command.CommandType = CommandType.StoredProcedure;

                // Thêm tham số vào stored procedure

                command.Parameters.AddWithValue("@MaSinhVien", maSinhVien);

                command.Parameters.AddWithValue("@MaLopHocCu", maLopHocCu);

                command.Parameters.AddWithValue("@MaLopHocMoi", maLopHocMoi);

                // Thực thi stored procedure và trả về số dòng bị ảnh hưởng

                return command.ExecuteNonQuery();

            }

        }

    }

    catch (SqlException)

    {

        throw;  // Ném lại lỗi SQL nếu có

    }

    catch (Exception ex)

    {

        // Ném lại lỗi với thông điệp chi tiết nếu có lỗi khác

        throw new Exception($"Error: {ex.Message}");

    }

}

# TẠO USER VÀ PHÂN QUYỀN

## Các vai trò của ứng dụng

Ứng dụng có 5 vai trò là: người quản lý sinh viên, người quản lý giảng viên, người quản lý môn học, người quản lý lớp học, và người quản lý chính. Những người quản lý chỉ được quyền quản lý một loại tài nguyên sinh viên, giảng viên, môn học, hoặc lớp học tương ứng với họ, còn người quản lý chính có toàn quyền quản lý. Các tài khoản sau khi được tạo sẽ được gán cho một vai trò để có thể sử dụng cơ sở dữ liệu của ứng dụng.

## Tạo login trên server

USE master;

CREATE LOGIN adsv1 WITH PASSWORD = '1234';

CREATE LOGIN adgv1 WITH PASSWORD = '1234';

CREATE LOGIN admh1 WITH PASSWORD = '1234';

CREATE LOGIN adlh1 WITH PASSWORD = '1234';

CREATE LOGIN mainad1 WITH PASSWORD = '1234';

## Tạo user trong cơ sở dữ liệu

-- Tạo các user trong cơ sở dữ liệu QLDiemSVKhoaCNTT

USE QLDiemSVKhoaCNTT;

CREATE USER adsv1 FOR LOGIN adsv1;

CREATE USER adgv1 FOR LOGIN adgv1;

CREATE USER admh1 FOR LOGIN admh1;

CREATE USER adlh1 FOR LOGIN adlh1;

CREATE USER mainad1 FOR LOGIN mainad1;

## Tạo role

-- Tạo các Role

CREATE ROLE AdminSVRole;

CREATE ROLE AdminGVRole;

CREATE ROLE AdminMHRole;

CREATE ROLE AdminLHRole;

CREATE ROLE MainAdminRole;

## Thêm quyền cho các Role

-- admin sinh viên

GRANT SELECT ON dbo.vw\_SinhVien TO AdminSVRole;

GRANT SELECT ON dbo.vw\_XepHangSinhVienBangDiemTBTichLuy TO AdminSVRole;

GRANT SELECT ON dbo.vw\_LopHoc TO AdminSVRole;

GRANT SELECT ON dbo.fn\_LayDanhSachSinhVienTrongLop TO AdminSVRole;

GRANT SELECT ON dbo.fn\_LayBangDiemSinhVien TO AdminSVRole;

GRANT EXECUTE ON dbo.fn\_DemSoLuongLopDangKy TO AdminSVRole;

GRANT EXECUTE ON dbo.proc\_XemThoiKhoaBieuSinhVien TO AdminSVRole;

GRANT EXECUTE ON dbo.proc\_ThemSinhVien TO AdminSVRole;

GRANT EXECUTE ON dbo.proc\_XoaSinhVien TO AdminSVRole;

GRANT EXECUTE ON dbo.proc\_SuaSinhVien TO AdminSVRole;

GRANT EXECUTE ON dbo.proc\_CapNhatDiemSinhVien TO AdminSVRole;

GRANT EXECUTE ON dbo.fn\_TinhDiemTrungBinh TO AdminSVRole;

GRANT EXECUTE ON dbo.fn\_TinhDiemTrungBinhTichLuy TO AdminSVRole;

GRANT EXECUTE ON dbo.fn\_SoTinChiDaHoanThanh TO AdminSVRole;

-- admin giang vien

GRANT SELECT ON dbo.vw\_GiangVien TO AdminGVRole;

GRANT EXECUTE ON dbo.proc\_XoaGiangVien TO AdminGVRole;

GRANT EXECUTE ON dbo.proc\_ThemGiangVien TO AdminGVRole;

GRANT EXECUTE ON dbo.proc\_SuaGiangVien TO AdminGVRole;

GRANT EXECUTE ON dbo.fn\_KiemTraQuaMon TO AdminGVRole;

GRANT EXECUTE ON dbo.fn\_XemHocLucSinhVien TO AdminGVRole;

GRANT EXECUTE ON dbo.proc\_GiangVienDayMonHocVoiSoSVDangKyCaoNhat TO AdminGVRole;

-- admin môn học

GRANT SELECT ON dbo.vw\_MonHoc TO AdminMHRole;

GRANT EXECUTE ON dbo.proc\_ThemMonHoc TO AdminMHRole;

GRANT EXECUTE ON dbo.proc\_XoaMonHoc TO AdminMHRole;

GRANT EXECUTE ON dbo.proc\_SuaMonHoc TO AdminMHRole;

-- admin lop hoc

GRANT SELECT ON dbo.vw\_GiangVien TO AdminLHRole;

GRANT SELECT ON dbo.vw\_MonHoc TO AdminLHRole;

GRANT SELECT ON dbo.vw\_LopHoc TO AdminLHRole;

GRANT SELECT ON dbo.vw\_SinhVien TO AdminLHRole;

GRANT SELECT ON dbo.PhongHoc TO AdminLHRole;

GRANT SELECT ON dbo.vw\_LopHocConTrong TO AdminLHRole;

GRANT SELECT ON dbo.vw\_LopHocGiangVienPhuTrach TO AdminLHRole;

GRANT SELECT ON dbo.fn\_LayDanhSachSinhVienTrongLop TO AdminLHRole;

GRANT EXECUTE ON proc\_ThemLopHoc TO AdminLHRole;

GRANT EXECUTE ON proc\_XoaLopHoc TO AdminLHRole;

GRANT EXECUTE ON proc\_SuaLopHoc TO AdminLHRole;

GRANT EXECUTE ON proc\_DangKySinhVienVaoLop TO AdminLHRole;

GRANT EXECUTE ON proc\_XemKetQuaHocTapCuaLop TO AdminLHRole;

GRANT EXECUTE ON dbo.fn\_DemSoLopGiangVienPhuTrach TO AdminLHRole;

GRANT EXECUTE ON dbo.fn\_DemSoLuongSinhVienCuaLop TO AdminLHRole;

GRANT EXECUTE ON dbo.fn\_TinhPhanTramQuaMon TO AdminLHRole;

GRANT EXECUTE ON dbo.proc\_ChuyenSinhVienSangLopKhac TO AdminLHRole;

-- main admin

ALTER ROLE db\_owner ADD MEMBER MainAdminRole;

## Thêm user vào role

ALTER ROLE AdminSVRole ADD MEMBER adsv1;

ALTER ROLE AdminGVRole ADD MEMBER adgv1;

ALTER ROLE AdminMHRole ADD MEMBER admh1;

ALTER ROLE AdminLHRole ADD MEMBER adlh1;

ALTER ROLE MainAdminRole ADD MEMBER mainad1;

## Cấp quyền thực thi hàm hệ thống

-- cấp quyền thực thi hàm hệ thống msdb.dbo.sp\_send\_dbmail

USE msdb; -- dùng cơ sở dữ liệu chứa hàm hệ thống

-- tạo user cho cơ sở dữ liệu

CREATE USER adsv1 FOR LOGIN adsv1;

CREATE USER mainad1 FOR LOGIN mainad1;

-- tạo role tương ứng cho các user

CREATE ROLE AdminSVRole;

CREATE ROLE MainAdminRole;

-- cấp quyền cho các role

GRANT EXECUTE ON dbo.sp\_send\_dbmail TO AdminSVRole;

GRANT EXECUTE ON dbo.sp\_send\_dbmail TO MainAdminRole;

-- thêm user vào các role

ALTER ROLE AdminSVRole ADD MEMBER adsv1;

ALTER ROLE MainAdminRole ADD MEMBER mainad1;

# CÀI ĐẶT GIAO DIỆN

## Trang đăng nhập

Nhập mật khẩu và tài khoản được cấp để đăng nhập, một số tính năng sẽ không truy cập được nếu như quyền của tài khoản đó không đủ cao.

A screenshot of a login box

Description automatically generated

## Trang chính

Trang chính hiện 4 nút chức năng quản lý.

A screenshot of a computer

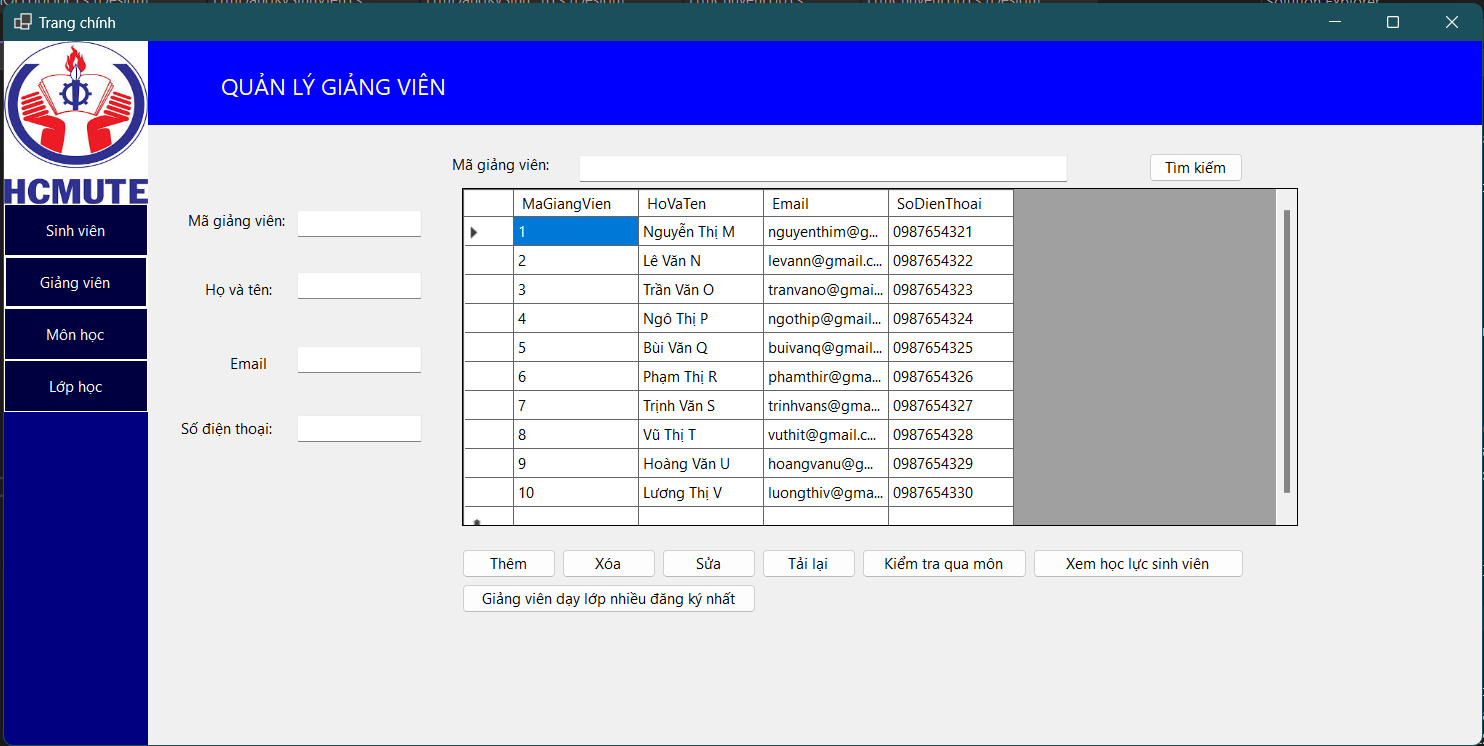
Description automatically generated

## Trang quản lý sinh viên

A screenshot of a computer

Description automatically generated

## Trang quản lý giảng viên



## Trang quản lý môn học

A screenshot of a computer

Description automatically generated

## Trang quản lý lớp học

A screenshot of a computer

Description automatically generated

**TỔNG KẾT**

**LỜI CẢM ƠN**

Tập thể nhóm em xin gửi lời cảm ơn chân thành nhất tới thầy Nguyễn Thành Sơn vì những kiến thức quý báu và sự hướng dẫn tận tình trong suốt quá trình học môn Hệ Quản Trị Cơ Sở Dữ Liệu, cả về kiến thức chuyên môn và kỹ năng mềm, tác phong trong học tập và làm việc sau này. Những bài giảng của thầy không chỉ giúp em hiểu rõ hơn về các khái niệm lý thuyết mà còn truyền cảm hứng và động lực cho em trong việc ứng dụng các kiến thức này vào thực tế, là nền tảng cốt lõi của mọi sinh viên Công nghệ thông tin. Nhóm em sẽ luôn ghi nhớ những bài học quý giá từ thầy và cố gắng vận dụng chúng trong công việc và học tập sau này.