

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM
KHOA CÔNG NGHỆ THÔNG TIN



HCMUTE

BỘ MÔN: HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU

QUẢN LÝ TRUNG TÂM NGOẠI NGỮ

GVHD: TS. Nguyễn Thành Sơn

Lớp HP: DBMS330284_22_1_02

Nhóm thực hiện: Nhóm 12

Học kỳ: 1

Năm học: 2022 – 2023

LỜI CẢM ƠN

Trước tiên chúng em xin bày tỏ sự trân trọng và lòng biết ơn đối với thầy giáo Nguyễn Thành Sơn, giảng viên bộ môn Hệ Quản Trị Cơ Sở Dữ Liệu – Khoa Công Nghệ Thông Tin – Trường Đại Học Sư Phạm Kỹ Thuật TP HCM. Trong suốt thời gian học và làm đồ án, thầy đã tận tâm chỉ bảo và định hướng cho nhóm em trong việc thực hiện đồ án.

Mặc dù đã rất cố gắng để hoàn thiện đồ án với tất cả nỗ lực, do tìm hiểu và xây dựng đồ án trong thời gian có hạn và kiến thức còn hạn chế, đồ án “Quản lí trung tâm ngoại ngữ” chắc chắn sẽ không thể tránh khỏi những sai sót. Chúng em rất mong nhận được sự quan tâm, thông cảm và những đóng góp quý báu của thầy và các bạn để đồ án ngày càng hoàn thiện hơn trong tương lai.

Sau cùng, chúng em xin kính chúc quý thầy cô thuộc khoa Công Nghệ Thông Tin dồi dào sức khỏe, niềm tin để tiếp tục thực hiện sứ mệnh cao đẹp của mình và truyền đạt kiến thức cho thế hệ mai sau.

Thủ Đức, tháng 11 năm 2022

Nhóm 12

LỜI MỞ ĐẦU

Hiện nay nhu cầu học ngoại ngữ của mọi người tại các trung tâm ngày càng tăng đặc biệt là trong một nền kinh tế toàn cầu. Nhận thấy sự khó khăn trong quá trình quản lý học viên, giảng viên cũng như là theo dõi bài học và việc giảng dạy tại trung tâm, hệ thống quản lý trung tâm ngoại ngữ này được ra đời trên nền Winform với mục đích hỗ trợ quản lý dễ dàng hơn và có thể thực hiện quản lý ở mọi nơi, mọi lúc. Xây dựng thành công hệ thống giúp trung tâm ngoại ngữ giảm được công sức người quản lý và công tác quản lý được thực hiện nhanh chóng, hiệu quả.

Đồ án gồm 5 chương:

Chương 1: Đặc tả đề tài

Chương 2: Phân tích thiết kế hệ thống

Chương 3: Thiết kế các chức năng

Chương 4: Tạo user và phân quyền

Chương 5: Giao diện hệ thống

DANH SÁCH THÀNH VIÊN NHÓM 12

STT	Họ và tên	MSSV
1	Lê Quang Tùng	20110746
2	Nguyễn Văn Lâm	20110668
3	Phạm Quỳnh Hương	20110141
4	Lê Minh Tường	20110280

MỤC LỤC

LỜI CẢM ƠN	
LỜI MỞ ĐẦU	
CHƯƠNG 1: ĐẶC TẢ ĐỀ TÀI	1
1.1 Mô tả bài toán.....	1
1.2 Mô tả chức năng của bài toán	2
CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ HỆ THỐNG.....	5
2.1 Thiết kế mô hình cơ sở dữ liệu mức quan niệm	5
2.2 Thiết kế mô hình cơ sở dữ liệu mức logic	6
2.3 Cài đặt CSDL và các ràng buộc.....	7
2.3.1 Tạo bảng.....	7
2.3.2 Các ràng buộc	10
2.3.3 Tạo Trigger	11
2.3.4 Tạo View.....	13
2.3.5 Tạo Function.....	16
2.3.6 Tạo Procedure	18
CHƯƠNG 3. THIẾT KẾ CÁC CHỨC NĂNG	35
3.1 Kết nối cơ sở dữ liệu.....	35
3.2 Quản lí nhân viên	36
3.3 Quản lí học viên	44
3.4 Đổi Password	48
3.5 Quản lí khóa học.....	50

3.6 Quản lí lớp học	52
3.7 Quản lý thanh toán.....	54
3.8 Xem danh sách lớp học với vai trò là giảng viên.....	59
3.9 Xem thời khóa biểu với vai trò là giảng viên.....	60
3.10 Xem thời khóa biểu với vai trò là học viên	61
3.11 Xem lịch sử giao dịch với vai trò là học viên	62
CHƯƠNG 4. TẠO USER VÀ PHÂN QUYỀN	63
4.1 Tạo role.....	63
4.2 Phân quyền.....	64
CHƯƠNG 5. GIAO DIỆN HỆ THỐNG	66
5.1 Form Login	66
5.2 Form HomePage.....	67
5.3 Form manage	70
5.4 Form views.....	73

CHƯƠNG 1: ĐẶC TẢ ĐỀ TÀI

1.1 Mô tả bài toán

Đề tài “Quản lý trung tâm ngoại ngữ” hướng đến các đối tượng sử dụng cụ thể là học viên (Students), giáo viên (Teachers), nhân viên (Staff). Những đối tượng này cần lưu trữ các thông tin như Mã số (ID) (được tạo tự động theo thứ tự tăng dần), Họ tên (Name), Email, Số điện thoại (Phone), Username, Password, và vị trí làm việc ở trung tâm (Position) đối với nhân viên.

Đối với tài khoản (Accounts) ta lưu trữ Username, Password (mặc định là Mtl@091202 lúc khởi tạo tài khoản) của từng người dùng. Để phân quyền là học viên, giảng viên hay nhân viên, ta lưu trữ RoleID của từng vai trò.

Chúng ta cũng cần lưu trữ các đối tượng là Khóa học (Courses) và Lớp học (Classes).

Đối với Khóa học (Courses), các dữ liệu cần lưu trữ là Mã khóa học (ID) (được tạo tự động theo thứ tự tăng dần), Tên khóa học (Name), Mục tiêu đầu ra của khóa học (Target), Số lượng bài học (No_Lessons) và Giá của khóa học (Price).

Đối với Lớp học (Classes), cần lưu trữ các thông tin về Mã lớp học (ID) (được tạo tự động theo thứ tự tăng dần), Tên lớp (Name), Ngày bắt đầu (Start_Date), Ngày kết thúc (End_Date), thông tin giảng viên dạy (Username), ID khóa học (Course ID), các ngày học trong tuần (WeekDays), Thời gian bắt đầu (Start_Time), Thời gian kết thúc (End_Time), Tên phòng học (ClassRoom), Số lượng học viên của lớp học đó (No_Students) được tự động cập nhật.

Để kiểm tra tình trạng đóng học phí của học viên, ta cần có bảng thống kê Thanh toán (Payments) của các học viên. Chúng ta cần lưu trữ các thông tin về Mã (ID) (được tạo tự động theo thứ tự tăng dần), Ngày thanh toán (Payment_Date), Số tiền (Amount), Phương thức thanh toán (Payment_Method), Username của học viên thực hiện thanh toán, Trạng thái thanh toán (Status) được cập nhật tự động, khi học viên đã hoàn thành tổng học phí thì sẽ được cập nhật là đã thanh toán.

1.2 Mô tả chức năng của bài toán

Chức năng đăng nhập, đổi mật khẩu của tài khoản và đăng ký tài khoản tự động khi khởi tạo thông tin cá nhân với Password được tạo tự động là Mtl@091202.

Đối với quản trị viên:

- Xem thông tin cá nhân.
- Quản trị nhân viên: quản trị viên có quyền thêm, cập nhật, xóa, tìm kiếm nhân viên theo tên nhân viên, vị trí công việc ở trung tâm và tìm kiếm trong tổng thể những thông tin được hiển thị.
- Quản trị học viên: quản trị viên có quyền thêm, cập nhật, xóa, tìm kiếm học viên theo tên học viên, tên lớp, tên khóa học, tên giảng viên của học viên đó và tìm kiếm trong tổng thể những thông tin được hiển thị.
- Quản trị giảng viên: quản trị viên có quyền thêm, cập nhật, xóa, tìm kiếm giảng viên theo tên giảng viên, tên lớp, tên khóa học mà giảng viên đó dạy và tìm kiếm trong tổng thể những thông tin được hiển thị.
- Quản lý khóa học: quản trị viên có quyền thêm, cập nhật, xóa, tìm kiếm khóa học theo tên khóa học, theo mức giá tối đa hoặc tối thiểu và tìm kiếm trong tổng thể những thông tin được hiển thị.
- Quản lý lớp học: quản trị viên có quyền thêm, cập nhật, xóa, tìm kiếm lớp học theo tên lớp học, tên phòng học, tên khóa học, tên giảng viên của lớp học đó và tìm kiếm trong tổng thể những thông tin được hiển thị.
- Quản lý thanh toán: quản trị viên có quyền thêm, cập nhật, xóa, tìm kiếm thông tin thanh toán của học viên theo tên học viên, số điện thoại, phương thức thanh toán, trạng thái thanh toán của học viên và tìm kiếm trong tổng thể những thông tin được hiển thị.

Đối với nhân viên:

- Xem thông tin cá nhân.

- Quản trị học viên: quản trị viên có quyền thêm, cập nhật, xóa, tìm kiếm học viên theo tên học viên, tên lớp, tên khóa học, tên giảng viên của học viên đó và tìm kiếm trong tổng thể những thông tin được hiển thị.
- Quản trị giảng viên: quản trị viên có quyền thêm, cập nhật, xóa, tìm kiếm giảng viên theo tên giảng viên, tên lớp, tên khóa học mà giảng viên đó dạy và tìm kiếm trong tổng thể những thông tin được hiển thị.
- Quản lý khóa học: quản trị viên có quyền thêm, cập nhật, xóa, tìm kiếm khóa học theo tên khóa học, theo mức giá tối đa hoặc tối thiểu và tìm kiếm trong tổng thể những thông tin được hiển thị.
- Quản lý lớp học: quản trị viên có quyền thêm, cập nhật, xóa, tìm kiếm lớp học theo tên lớp học, tên phòng học, tên khóa học, tên giảng viên của lớp học đó và tìm kiếm trong tổng thể những thông tin được hiển thị.
- Quản lý thanh toán: quản trị viên có quyền thêm, cập nhật, xóa, tìm kiếm thông tin thanh toán của học viên theo tên học viên, số điện thoại, phương thức thanh toán, trạng thái thanh toán của học viên và tìm kiếm trong tổng thể những thông tin được hiển thị.

Đối với giảng viên:

- Xem thông tin cá nhân.
- Xem lớp học đang dạy: giảng viên có quyền xem thông tin về lớp học mà giảng viên đó đang dạy
- Xem tất cả lớp học hiện có của trung tâm: giảng viên có quyền xem thông tin tất cả lớp học hiện có của trung tâm để tham khảo về việc đăng kí giảng dạy các lớp học đó, thực hiện tìm kiếm theo tên giảng viên, tên lớp học, tên khóa học và tìm kiếm trong tổng thể những thông tin được hiển thị.
- Xem thời khóa biểu: xem thời khóa biểu của các lớp mà giảng viên đó đang dạy.

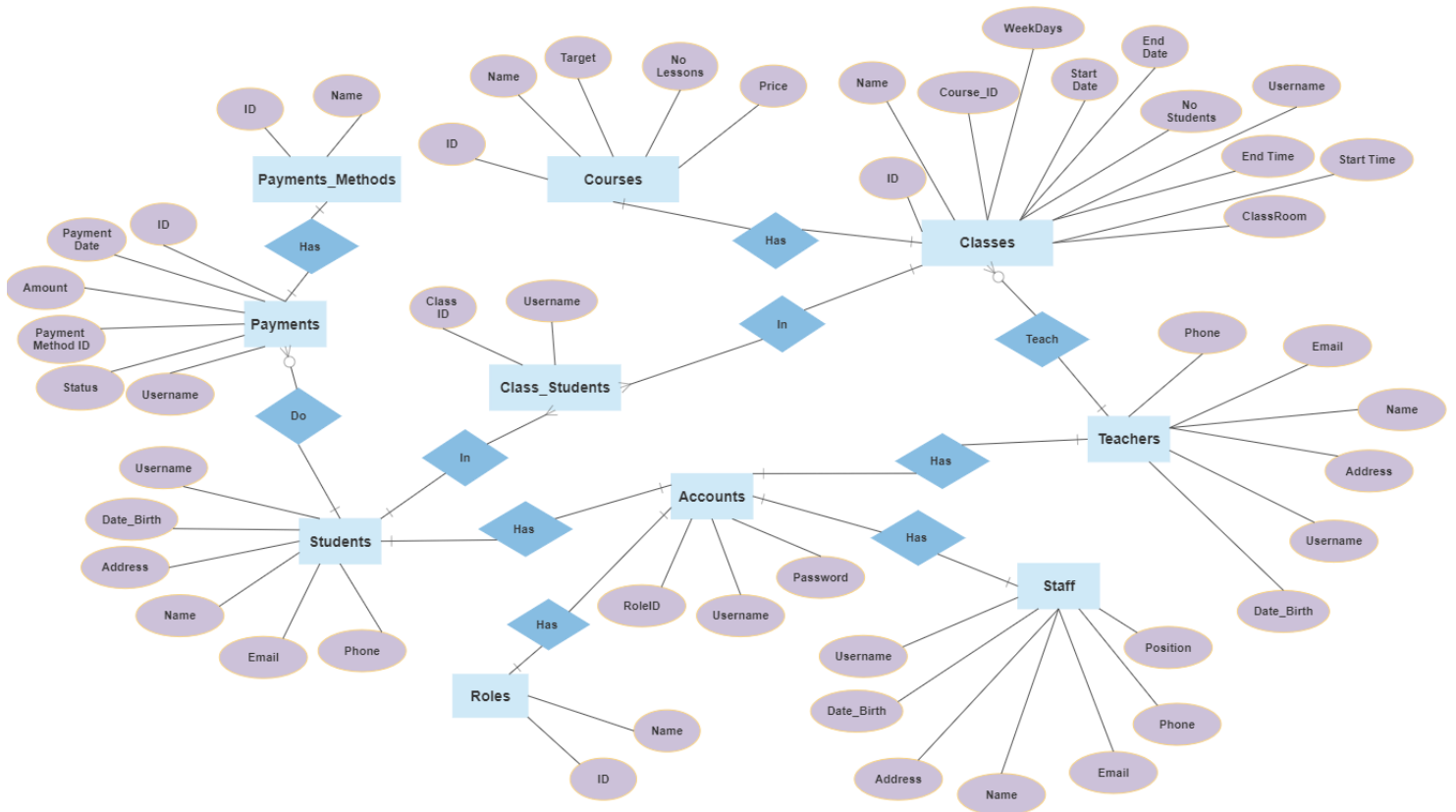
Đối với học viên:

- Xem thông tin cá nhân.

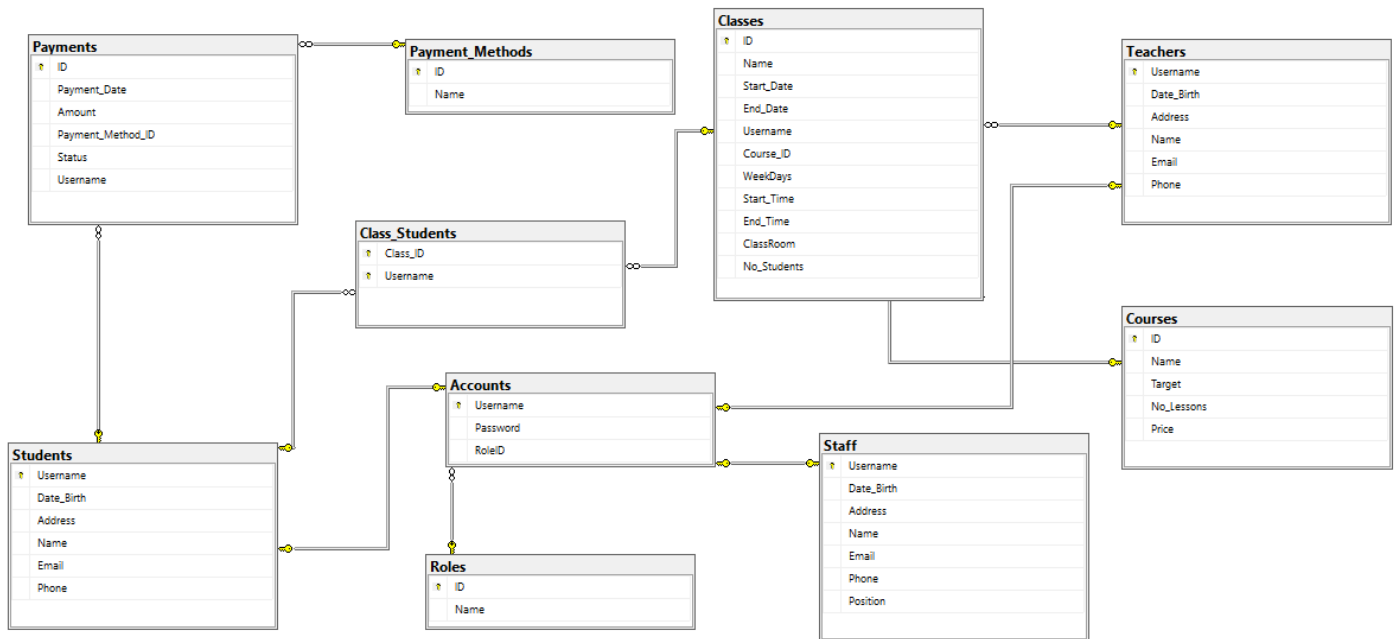
- Xem thời khóa biểu: học viên có quyền xem thời khóa biểu của mình chứa thông tin về lớp học, giảng viên đảm nhiệm và thời gian học.
- Xem lịch sử giao dịch: học viên có quyền xem thông tin thanh toán học phí của mình.

CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ HỆ THỐNG

2.1 Thiết kế mô hình cơ sở dữ liệu mức quan niệm



2.2 Thiết kế mô hình cơ sở dữ liệu mức logic



2.3 Cài đặt CSDL và các ràng buộc

2.3.1 Tạo bảng

Table Courses

```
create table Courses(  
    ID int IDENTITY(1,1) primary key,  
    Name nvarchar(40) not null,  
    Target float not null,  
    No_Lessons int not null check (No_Lessons >= 0),  
    Price int not null check(Price >= 0),  
);
```

Table Roles

```
create table Roles(  
    ID int IDENTITY(1,1) primary key,  
    Name nvarchar(50) not null,  
);
```

Table Accounts

```
create table Accounts (  
    Username varchar(100) primary key,  
    Password varchar(100) not null,  
    RoleID int not null,  
    foreign key (RoleID) references Roles(ID) ,  
);
```

Table Students

```
create table Students(  
    Username varchar(100) primary key,  
    Date_Birth date not null,  
    Address nvarchar(100) not null,  
    Name nvarchar(50) not null,  
    Email varchar(50) unique not null,  
    Phone varchar(11) unique not null,  
    check(10 = len(Phone) or len(Phone) = 11),  
    foreign key(Username) references Accounts(Username) on update cascade ,  
);
```

Table Teachers

```
create table Teachers(  
    Username varchar(100) primary key,  
    Date_Birth date not null,  
    Address nvarchar(100) not null,  
    Name nvarchar(50) not null,  
    Email varchar(50) unique not null,  
    Phone varchar(11) unique not null,  
    check(10 = len(Phone) or len(Phone) = 11),  
    foreign key(Username) references Accounts(Username) on update cascade,);
```

Table Staff

```
create table Staff(  
    Username varchar(100) primary key,  
    Date_Birth date not null,  
    Address nvarchar(100) not null,  
    Name nvarchar(50) not null,  
    Email varchar(50) unique not null,  
    Phone varchar(11) unique not null,  
    Position nvarchar(40) not null,  
    check(10 = len(Phone) or len(Phone) = 11),  
    foreign key(Username) references Accounts(Username) on update cascade,);
```

Table Classes

```
create table Classes(  
    ID int IDENTITY(1,1) primary key,  
    Name nvarchar(100) not null,  
    Start_Date date not null,  
    End_Date date not null,  
    Username varchar(100) not null,  
    Course_ID int not null,  
    WeekDays varchar(10) not null,  
    Start_Time time not null,  
    End_Time time not null,  
    Classroom varchar(20) not null,  
    No_Students int not null,  
    foreign key(Username) references Teachers(Username) on update cascade on delete  
cascade,  
    foreign key(Course_ID) references Courses(ID) ,);
```

Table Class_Students

```
create table Class_Students(  
    Class_ID int,  
    Username varchar(100),  
    primary key (Class_ID, Username),  
    foreign key(Class_ID) references Classes(ID) on delete cascade,  
    foreign key(Username) references Students(Username) on update cascade ,  
);
```

Table Payment_Methods

```
create table Payment_Methods(  
    ID int IDENTITY(1,1) primary key,  
    Name nvarchar(100) not null  
);
```

Table Payments

```
create table Payments(  
    ID int IDENTITY(1,1) primary key,  
    Payment_Date date not null,  
    Amount int not null check(Amount >= 0),  
    Payment_Method_ID int not null,  
    Status bit not null,  
    Username varchar(100) not null,  
    foreign key(Payment_Method_ID) references Payment_Methods(ID),  
    foreign key(Username) references Students(Username) on update cascade ,  
);
```

2.3.2 Các ràng buộc

Ràng buộc phạm vi Position của Staff gồm: Admin, HR, Marketing, Trainer, Sales, Receptionist:

```
alter table Staff
add constraint CheckPosition
check (Position like 'Admin' or Position like 'HR' or Position like 'Marketing' or
Position like 'Trainer' or Position like 'Sales' or Position like 'Receptionist');
```

Ràng buộc format của Password phải chứa chữ hoa, chữ thường, chữ số, một vài loại ký tự đặc biệt (! @ # \$ % ^ & * () - _ + = . , ; : ~) và có độ dài lớn hơn 6 ký tự:

```
alter table Accounts
add constraint CheckPassword_Accounts
check (Password like '%[0-9]%' and Password like '%[A-Z]%' collate Latin1_General_BIN2
and Password like '%[!@#$$%^&*()-_+=.,;:~]%' and len(Password)>6);
```

Ràng buộc format của Email chỉ được chứa số, ký tự, dấu chấm, dấu @, và đúng như format email bình thường:

```
alter table Students add constraint checkEmailStudent
check (Email like '%_@_%._%' and Email not like '% %' and PATINDEX('%[^a-z,0-9,@,.,,]%', Email) = 0);
alter table Staff add constraint checkEmailStaff
check (Email like '%_@_%._%' and Email not like '% %' and PATINDEX('%[^a-z,0-9,@,.,,]%', Email) = 0);
alter table Teachers add constraint checkEmailTeacher
check (Email like '%_@_%._%' and Email not like '% %' and PATINDEX('%[^a-z,0-9,@,.,,]%', Email) = 0);
```

Ràng buộc Phone chỉ chứa chữ số:

```
alter table Students add constraint checkPhoneStudent check (PATINDEX('%[^0-9]%', Phone) = 0);
alter table Staff add constraint checkPhoneStaff check (PATINDEX('%[^0-9]%', Phone) = 0);
alter table Teachers add constraint checkPhoneTeacher check (PATINDEX('%[^0-9]%', Phone) = 0);
```

Ràng buộc Weekdays theo format cố định:

```
alter table Classes add constraint checkDay check (PATINDEX('%[^2-7, -]%', Weekdays) = 0);
```

Ràng buộc về thời gian của khóa học (Date và Time):

```
alter table Classes add constraint checkDayStartEnd check (Start_Date < End_Date);
alter table Classes add constraint checkTimeStartEnd check (Start_Time < End_Time);
```


2.3.3 Tạo Trigger

Quy định thang điểm cố định của các loại khóa học khác nhau ở trung tâm:

```
create trigger checkTargetCourse on Courses
after update, insert as
declare @name nvarchar(40), @target float
select @name = ne.Name, @target = ne.Target
from inserted ne
begin
if @target < 0
rollback
else
begin
if @name like '%IELTS%'
if @target > 9
rollback
if @name like '%TOEIC%' and @target != CAST(@target as int)
rollback
if @name like '%TOEIC%' and @name like '%LR%' and @target > 990
rollback
if @name like '%TOEIC%' and @name like '%SW%' and @target >400
rollback
if @name like '%TOEIC%' and @name not like '%SW%' and @name not like '%LR%'
rollback
end
end
go
```

Chuyển đổi trạng thái thanh toán học phí, cộng dồn theo số tiền đóng và so sánh với tổng tiền của các khóa học đã đăng ký, sẽ là 1 nếu đã thanh toán đủ và là 0 nếu ngược lại:

```
create trigger Status_Payment
on Payments for insert, update
as
begin
declare @current_Amount int; declare @price int; declare @cid int;
select @current_Amount = sum(dbo.Payments.Amount) from dbo.Payments, inserted
where inserted.Username = Payments.Username
select @price = sum(c.Price)
from inserted, Class_Students cs inner join Classes cl on cs.Class_ID = cl.ID
inner join Courses c on cl.Course_ID = c.ID
where inserted.Username = cs.Username
if @current_Amount >= @price
begin
update dbo.Payments set dbo.Payments.Status = 1 from inserted,
dbo.Payments where dbo.Payments.ID = inserted.ID
end
else
begin
update dbo.Payments set dbo.Payments.Status = 0 from inserted,
dbo.Payments where dbo.Payments.ID = inserted.ID
end
end
go
```

Tăng tự động số lượng học viên trong từng lớp khi thêm học viên mới, số lượng tối đa học viên mỗi lớp là 10:

```
create trigger IncreaseNoStudent
on dbo.Class_Students for insert, update
as
begin
    declare @noStudents int;
    select @noStudents = dbo.Classes.No_Students from dbo.Classes, inserted where
inserted.Class_ID = dbo.Classes.ID
    if @noStudents = 10
    begin
        raiserror (N'This class is full of students!',16,1)
        rollback transaction
    end

    else
    begin
        update dbo.Classes set dbo.Classes.No_Students += 1 from inserted,
        dbo.Classes where inserted.Class_ID = dbo.Classes.ID
    end
end
go
```

Giảm tự động số lượng học viên trong từng lớp khi xóa học viên:

```
create trigger DecreaseNoStudent on dbo.Class_Students for delete, update as
        update dbo.Classes set dbo.Classes.No_Students -= 1 from deleted,
        dbo.Classes where deleted.Class_ID = dbo.Classes.ID
go
```

2.3.4 Tạo View

Chọn ra những thuộc tính kết hợp của 1 học viên:

```
create view Student_Info as
select Students.Username, Students.Name as StudentName, convert
(varchar(100),Students.Date_Birth, 103) as DateOfBirth,
Students.Address, Students.Email, Students.Phone, Courses.Name as CourseName,
Classes.Name as ClassName, Teachers.Name as TeacherName
from Students
inner join Class_Students on Students.Username=Class_Students.Username
inner join Classes on Class_Students.Class_ID=Classes.ID
inner join Teachers on Classes.Username=Teachers.Username
inner join Courses on Course_ID = Courses.ID;
go
```

Chọn ra những thuộc tính kết hợp của 1 giảng viên:

```
create view Teacher_Info as
select Teachers.Username, Teachers.Name as TeacherName,convert
(varchar(100),Date_Birth, 103) as DateOfBirth, Address, Email,
Phone, ISNULL(LUONG,0) as Salary from Teachers left join (select *,
dbo.TinhLuong(A.HSL) as LUONG from (select Username, count (*) as HSL from Classes
group by Username)A)B on
B.Username= Teachers.Username
```

Chọn ra những thuộc tính kết hợp của giảng viên bao gồm cả việc tính lương:

```
create view TeacherAndSalary as
select Teachers.Username, Teachers.Name as TeacherName,convert
(varchar(100),Date_Birth, 103) as DateOfBirth, Address, Email,
Phone, ISNULL(LUONG,0) as Salary, Classes.Name as ClassName, Courses.Name as CourseName
from Teachers left join (select *,
dbo.TinhLuong(A.HSL) as LUONG from (select Username, count (*) as HSL from Classes
group by Username)A)B on
B.Username= Teachers.Username inner join Classes on B.Username = Classes.Username inner
join Courses
on Classes.Course_ID=Courses.ID;
go
```

Thời khóa biểu của tất cả học viên:

```
create view StudentScheduleView
as
select cs.Username as Username, cl.ID as ClassID, cl.Name as ClassName,c.Name as
CourseName ,t.Name as TeacherName,
cl.WeekDays, CONCAT(SUBSTRING(convert(varchar, cl.Start_Time ,108),1,5), ' :
',SUBSTRING(convert(varchar, cl.End_Time ,108),1,5)) as Time, cl.ClassRoom as
ClassRoom
from Class_Students cs inner join Classes cl on cs.Class_ID = cl.ID inner join Courses
c on cl.Course_ID = c.ID inner join Teachers t on cl.Username = t.Username
go
```

Thời khóa biểu của tất cả giảng viên:

```
create view TeacherScheduleView
as
select t.Username as Username, cl.ID as ClassID, cl.Name as ClassName, cl.Course_ID as
CourseID, c.Name as CourseName ,
cl.WeekDays as Date, CONCAT(SUBSTRING(convert(varchar, cl.Start_Time ,108),1,5), ' :
' ,SUBSTRING(convert(varchar, cl.End_Time ,108),1,5)) as Time, cl.ClassRoom as
ClassRoom
from Classes cl inner join Courses c on cl.Course_ID = c.ID inner join Teachers t on
cl.Username = t.Username
go
```

Chọn ra những thuộc tính kết hợp của 1 nhân viên:

```
create view Staff_Info as
select Username, Name as StaffName, convert (varchar(100),Date_Birth, 103) as
DateOfBirth, Address, Email, Phone, Position, dbo.Tinh_Luong(Position) as Salary
from Staff
go
```

Quản lý thanh toán chung của tất cả học viên:

```
create view PaymentsView
as
select ID ,Students.Username,Students.Name as StudentName, Students.Email,
Students.Phone as Phone,
Payments.Payment_Date as PaymentDate, Amount as Amount ,
[dbo].PaymentMethodName_byId(Payments.Payment_Method_ID) as PaymentMethod,
[dbo].TrangThaiThanhToan(Payments.Status) as PaymentStatus
from Payments inner join Students on Students.Username = Payments.Username
go
```

Chọn ra những thuộc tính kết hợp của 1 lớp học:

```
create view Class_Info as
select Classes.ID, Classes.Name as ClassName, convert (varchar(100),Start_Date, 103) as
StartDate, convert (varchar(100),End_Date, 103) as EndDate, WeekDays, convert
(varchar(100),Start_Time, 108) as StartTime,
convert (varchar(100),End_Time, 108) as EndTime, ClassRoom, No_Students as NoStudents,
Courses.Name as CourseName, ROUND(Target,1) as Target, Teachers.Name as TeacherName
from Classes left join Courses
on Classes.Course_ID=Courses.ID left join Teachers on
Teachers.Username=Classes.Username;
go
```

Chọn ra những thuộc tính của 1 khóa học:

```
create view Course_Info as
select ID, Name as CourseName, ROUND(Target,1) as Target, No_Lessons as NoLessons,
Price from Courses
go
```

Chọn ra những thuộc tính của lớp học:

```
create view ClassesView
as
select Classes.ID, Classes.Name as ClassName, Classes.Start_Date as StartDate,
Classes.End_Date as EndDate, [dbo].getTeacherName_byUsername(Classes.Username)
as TeacherName, [dbo].getCourseName_byID(Classes.Course_ID) as CourseName,
Classes.WeekDays, Classes.Start_Time, Classes.End_Time, Classes.ClassRoom,
Classes.No_Students
from Classes
go
```

Lịch sử giao dịch của tất cả học viên:

```
create view PaymentView
as
select st.Username, pa.Payment_Date as PaymentDate, pa.Amount as Amount, pm.Name as
PaymentMethod, pa.Status as PaymentStatus
from Students st inner join Payments pa on st.Username = pa.Username inner join
Payment_Methods pm on pa.Payment_Method_ID = pm.ID
go
```

2.3.5 Tạo Function

Gán lương cứng cho các vị trí của nhân viên của trung tâm:

```
create function Tinh_Luong (@position as nvarchar(40))
returns int
as begin
declare @result int
if @position = 'Admin' set @result = 12000000
if @position = 'HR' set @result = 10000000
if @position = 'Marketing' set @result = 9000000
if @position = 'Sales' or @position = 'Trainer' set @result = 7000000
if @position = 'Receptionist' set @result = 4000000
return @result
end;
go
```

Trả về tên của phương thức thanh toán bằng ID:

```
create function PaymentMethodName_byId (@id int)
returns nvarchar(20)
as begin
declare @name nvarchar(20)
set @name = (select Payment_Methods.Name from Payment_Methods where Payment_Methods.ID
= 1)
return @name
end
```

Trả về thông tin trạng thái thanh toán theo dạng chữ:

```
create function TrangThaiThanhToan (@status bit)
returns nvarchar(30)
as begin
declare @status_name nvarchar(30)
set @status_name = ''
if @status = 1
begin
set @status_name = N'Paid'
end
if @status = 0
begin
set @status_name = N'Unpaid'
end
return @status_name
end
go
```

Tính lương của giảng viên theo công thức: $5000000 * (\text{số lớp giảng viên dạy})$

```
create function TinhLuong (@noclass as int)
returns int
as begin
declare @result int = 5000000*@noclass
if @noclass is null
set @result =0
return @result
end;
go
```

Tìm tên khóa học bằng ID:

```
create function getCourseName_byID (@id int)
returns nvarchar(30)
as begin
declare @name nvarchar(30)
set @name = (select Courses.Name from Courses where Courses.ID = 1)
return @name
end
```

Tìm tên giảng viên bằng username:

```
create function getTeacherName_byUsername (@username nvarchar(30))
returns nvarchar(30)
as begin
declare @name nvarchar(30)
set @name = (select Teachers.Name from Teachers where Teachers.Username = @username)
return @name
end
```

Trả về bảng chứa thông tin thanh toán:

```
create function getPayments_func()
returns table
as
return (
select ID ,Students.Username,Students.Name as StudentName, Students.Email,
Students.Phone as Phone,
Payments.Payment_Date as PaymentDate, Amount as Amount,
[dbo].PaymentMethodName_byId(Payments.Payment_Method_ID) as PaymentMethod,
[dbo].TrangThaiThanhToan(Payments.Status) as PaymentStatus
from Payments inner join Students on Students.Username = Payments.Username
)
go
```

2.3.6 Tạo Procedure

Thêm thông tin khóa học:

```
create procedure AddCourse @coursename nvarchar(40), @target float, @nolessons int,
@price int as
begin
begin try
insert into Courses values (@coursename, @target, @nolessons, @price);
end try
begin catch
DECLARE @CustomMessage VARCHAR(1000),
        @CustomError INT,
        @CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
SET @CustomError = 54321;
SET @CustomState = 1;
THROW @CustomError, @CustomMessage, @CustomState;
end catch
end
go
```

Cập nhật thông tin khóa học:

```
create procedure UpdateCourse @courseid int , @coursename nvarchar(40), @target float,
@nolessons int, @price int as
begin
    if exists (select Courses.ID from Courses where Courses.ID = @courseid)
        begin
            update Courses
            set Name = @coursename, Target = @target, No_Lessons=@nolessons,
Price=@price where ID = @courseid
        end
    else
        begin
            DECLARE @CustomMessage VARCHAR(1000),
                    @CustomError INT,
                    @CustomState INT;
            SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
            SET @CustomError = 54321;
            SET @CustomState = 1;
            THROW @CustomError, @CustomMessage, @CustomState;
        end
    end
end
go
```


Procedure + transaction xóa thông tin khóa học:

```
create procedure deleteCOURSE_sequentially @id int
as
begin try
    begin transaction
        delete from Classes where Classes.Course_ID = @id;
        delete from Courses where Courses.ID = @id;
        commit transaction
    end try
begin catch
    DECLARE @CustomMessage VARCHAR(1000),
            @CustomError INT,
            @CustomState INT;
    SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
    SET @CustomError = 54321;
    SET @CustomState = 1;
    rollback;
    THROW @CustomError, @CustomMessage, @CustomState;
end catch
go
```

Lấy ra thông tin khóa học thông qua tên khóa học:

```
create procedure GetCourseByCourseName @coursename nvarchar(40) as
select * from Course_Info where CourseName= @coursename;
go
```

Lấy ra thông tin khóa học thông qua mức học phí cao nhất được nhập:

```
create procedure GetCourseByMaxPrice @price int as
begin try
    select * from Course_Info where Price <= @price;
end try
begin catch
    DECLARE @CustomMessage VARCHAR(1000),
            @CustomError INT,
            @CustomState INT;
    SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
    SET @CustomError = 54321;
    SET @CustomState = 1;
    THROW @CustomError, @CustomMessage, @CustomState;
end catch
go
```

Lấy ra thông tin khóa học thông qua mức học phí thấp nhất được nhập:

```
create procedure GetCourseByMinPrice @price int as
begin try
select * from Course_Info where Price >= @price;
end try
begin catch
DECLARE @CustomMessage VARCHAR(1000),
        @CustomError INT,
        @CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
SET @CustomError = 54321;
SET @CustomState = 1;
THROW @CustomError, @CustomMessage, @CustomState;
end catch
go
```

Thêm giảng viên mới vào SQL Server:

```
create procedure addTeacherAccountToServer @username varchar(30)
as begin
    begin try
        declare @login nvarchar(4000)
        set @login = N'CREATE LOGIN ' + QUOTENAME(@username) + ' WITH PASSWORD =
' + QUOTENAME('Mtl@091202', '') + ', default_database = ' +
QUOTENAME('LanguageCenter')
        exec(@login)
        declare @user nvarchar(4000)
        set @user = N'CREATE USER ' + QUOTENAME(@username) + ' FOR LOGIN ' +
QUOTENAME(@username)
        exec(@user)
        exec sp_addrolemember 'Teacher', @username
    end try
    begin catch
        rollback
    end catch
end
go
```

Xóa User khỏi SQL Server:

```
create procedure DeleleUserOnServer @username varchar(30)
as begin
    begin try
        declare @user nvarchar(4000)
        set @user = N'DROP USER ' + QUOTENAME(@username) + ';
        exec(@user)
        declare @login nvarchar(4000)
        set @login = N'DROP LOGIN ' + QUOTENAME(@username) + ';
        exec(@login)
    end try
    begin catch
        rollback
    end catch
end
```

Xóa thông tin giảng viên:

```
create procedure deleteACCOUNT_TEACHER_sequentially @username nvarchar(50)
as
if exists (select Accounts.Username from Accounts where Username = @username)
begin
begin transaction
delete from Teachers where Username=@username;
delete from Accounts where Username = @username;
exec DeleleUserOnServer @username;
commit transaction
end
else
begin
DECLARE @CustomMessage VARCHAR(1000),
        @CustomError INT,
        @CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
SET @CustomError = 54321;
SET @CustomState = 1;
rollback;
THROW @CustomError, @CustomMessage, @CustomState;
end
go
```

Lấy ra thông tin giảng viên thông qua tên giảng viên:

```
create procedure GetTeacherByTeacherName @name nvarchar(50)
as select distinct * from Teacher_Info where TeacherName= @name
go
```

Lấy ra thông tin giảng viên thông qua tên khóa học:

```
create procedure GetTeacherByCourseName @name nvarchar(50)
as select distinct Username, TeacherName, DateOfBirth, Address, Email, Phone, Salary
from TeacherAndSalary where CourseName= @name
go
```

Lấy ra thông tin giảng viên thông qua tên lớp học:

```
create procedure GetTeacherByClassName @name nvarchar(50)
as
begin
select distinct Username, TeacherName, DateOfBirth, Address, Email, Phone, Salary from
TeacherAndSalary where ClassName= @name
end
go
```

Thêm thông tin giảng viên:

```
create procedure AddTeacher @username varchar(100), @name nvarchar(50), @dateofbirth
date, @address nvarchar(100),
@email varchar(50), @phone varchar(11)
as
begin
    begin try
        if IS_MEMBER('sysadmin') = 0
            begin
                EXEC master..sp_addsrvrolemember @loginame =
@username, @rolename = N'sysadmin'
            end
            begin transaction
                insert into Accounts values (@username, 'Mtl@091202', 2);
                insert into Teachers values (@username, @dateofbirth, @address, @name,
@email, @phone);
                exec addTeacherAccountToServer @username
                EXEC master..sp_dropsrvrolemember @loginame = @username, @rolename =
N'sysadmin'
            commit transaction
        end try
        begin catch
            DECLARE @CustomMessage VARCHAR(1000),
                @CustomError INT,
                @CustomState INT;
            SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
            SET @CustomError = 54321;
            SET @CustomState = 1;
            ROLLBACK;
            THROW @CustomError, @CustomMessage, @CustomState;
        end catch
    end
```

Cập nhật thông tin giảng viên:

```
create procedure UpdateTeacher @username varchar(100), @name nvarchar(50), @dateofbirth
date, @address nvarchar(100), @email varchar(50), @phone varchar(11)
as
begin
    if exists (select Accounts.Username from Accounts where Username = @username)
        begin
            update Teachers
                set Name=@name, Date_Birth=@dateofbirth, Address=@address, Email =
@email, Phone=@phone where Username=@username
        end
    else
        begin
            DECLARE @CustomMessage VARCHAR(1000),
                @CustomError INT,
                @CustomState INT;
            SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
            SET @CustomError = 54321;
            SET @CustomState = 1;
            THROW @CustomError, @CustomMessage, @CustomState;
        end
    end
go
```

Thêm thông tin lớp học:

```
create procedure AddClass @classname nvarchar(100), @startdate date, @enddate date,
@weekdays varchar(10),
@starttime time(7), @endtime time(7), @classroom varchar(20), @coursename nvarchar(40),
@target float, @teachername nvarchar(50) as
begin
    declare @courseid int, @teacherusername varchar(100)
    set @courseid = (select Courses.ID from Courses where Name = @coursename and
Target = @target)
    set @teacherusername = (select Username from Teachers where Name = @teachername)
    begin try
        insert into Classes values (@classname, @startdate, @enddate,
@teacherusername, @courseid, @weekdays, @starttime, @endtime, @classroom, 0);
    end try
    begin catch
        DECLARE @CustomMessage VARCHAR(1000),
                @CustomError INT,
                @CustomState INT;
        SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
        SET @CustomError = 54321;
        SET @CustomState = 1;
        THROW @CustomError, @CustomMessage, @CustomState;
    end catch
end
go
```

Cập nhật thông tin lớp học:

```
create procedure UpdateClass @classid int, @classname nvarchar(100), @startdate date,
@enddate date, @weekdays varchar(10), @starttime time(7), @endtime time(7),
@classroom varchar(20), @coursename nvarchar(40), @target float, @teachername
nvarchar(50) as
begin
    declare @courseid int, @teacherusername varchar(100)
    set @courseid = (select Courses.ID from Courses where Name = @coursename and
Target = @target)
    set @teacherusername = (select Username from Teachers where Name = @teachername)
    begin try
        update Classes
        set Name = @classname, Start_Date=@startdate, End_Date=@enddate,
Username=@teacherusername, Course_ID=@courseid, WeekDays=@weekdays,
Start_Time=@starttime,
        End_Time = @endtime, ClassRoom=@classroom where Classes.ID = @classid
    end try

    begin catch
        DECLARE @CustomMessage VARCHAR(1000),
                @CustomError INT,
                @CustomState INT;
        SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
        SET @CustomError = 54321;
        SET @CustomState = 1;
        THROW @CustomError, @CustomMessage, @CustomState;
    end catch
end
go
```

Xóa thông tin lớp học:

```
create procedure deleteCLASS_sequentially @id int
as
delete from Classes where ID = @id;
```

Lấy ra thông tin lớp học thông qua tên lớp học:

```
create procedure GetClassBYClassName @classname nvarchar(100) as
select * from Class_Info where ClassName=@classname
go
```

Lấy ra thông tin lớp học thông qua tên phòng học:

```
create procedure GetClassBYClassRoom @classroom varchar(20) as
select * from Class_Info where ClassRoom=@classroom
go
```

Lấy ra thông tin lớp học thông qua tên khóa học:

```
create procedure GetClassBYCourseName @courseName nvarchar(40) as
select * from Class_Info where CourseName=@courseName
go
```

Lấy ra thông tin lớp học thông qua tên giảng viên:

```
create procedure GetClassBYTeacherName @teacherName nvarchar(100) as
select * from Class_Info where TeacherName=@teacherName
go
```

Procedure + transaction xóa thông tin học viên ở 1 lớp học cụ thể:

```
create procedure deleteSTUDENT_View @username nvarchar(100), @classname nvarchar(100)
as
begin try
declare @classid int set @classid = (select ID from Classes where Name = @classname)
delete from Class_Students where Username = @username and Class_ID = @classid;
end try
begin catch
DECLARE @CustomMessage VARCHAR(1000),
        @CustomError INT,
        @CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
SET @CustomError = 54321;
SET @CustomState = 1;
THROW @CustomError, @CustomMessage, @CustomState;
end catch
go
```

Xóa thông tin học viên:

```
create procedure deleteACCOUNT_STUDENT_sequentially @username nvarchar(50)
as
if exists (select Accounts.Username from Accounts where Username = @username)
begin
begin transaction
delete from Class_Students where Class_Students.Username = @username ;
delete from Payments where Payments.Username = @username;
delete from Students where Students.Username = @username;
delete from Accounts where Accounts.Username = @username;
exec DeleleUserOnServer @username;
commit transaction
end
else
begin
DECLARE @CustomMessage VARCHAR(1000),
        @CustomError INT,
        @CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
SET @CustomError = 54321;
SET @CustomState = 1;
rollback;
THROW @CustomError, @CustomMessage, @CustomState;
end
go
```

Liệt kê một vài thông tin của học viên:

```
create procedure GetListStudent as
select Username, StudentName, DateOfBirth, Address, Email, Phone, ClassName
from Student_Info;
go
```

Lấy thông tin học viên thông qua tên học viên:

```
create procedure GetStudentByStudentName @name nvarchar(50)
as begin
select distinct Username, StudentName, DateOfBirth, Address, Email, Phone, ClassName
from Student_Info
where StudentName= @name
end
go
```

Lấy thông tin học viên thông qua tên giảng viên của học viên đó:

```
create procedure GetStudentByTeacherName @name nvarchar(50)
as begin
select distinct Username, StudentName, DateOfBirth, Address, Email, Phone, ClassName
from Student_Info
where Student_Info.TeacherName = @name
end
go
```

Lấy thông tin học viên thông qua tên khóa học:

```
create procedure GetStudentByCourseName @name nvarchar(50)
as
begin
select distinct Username, StudentName, DateOfBirth, Address, Email, Phone, ClassName
from Student_Info
where Student_Info.CourseName = @name
end
go
```

Lấy thông tin học viên thông qua tên lớp học:

```
create procedure GetStudentByClassName @name nvarchar(50)
as
begin
select distinct Username, StudentName, DateOfBirth, Address, Email, Phone from
Student_Info
where Student_Info.ClassName = @name
end
goc
```


Thêm học viên vào SQL Server:

```
create procedure addStudentAccountToServer @username varchar(30) as begin
    declare @login nvarchar(4000)
    set @login = N'CREATE LOGIN ' + QUOTENAME(@username) + ' WITH PASSWORD = ' +
    QUOTENAME('Mtl@091202', '') + ', default_database = ' + QUOTENAME('LanguageCenter')
    exec(@login)
    declare @user nvarchar(4000)
    set @user = N'CREATE USER '+QUOTENAME(@username)+' FOR LOGIN '+
    QUOTENAME(@username)
    exec(@user)
    exec sp_addrolemember 'Student', @username
end
go
```

Thêm thông tin học viên kèm thông tin đăng ký học lớp học:

```
create procedure AddStudent @username varchar(100), @name nvarchar(50), @dateofbirth
date, @address nvarchar(100),
@email varchar(50), @phone varchar(11), @classname nvarchar(100)
as
begin
declare @check bit = 0
if IS_MEMBER('sysadmin') = 0
begin
set @check = 1
EXEC master..sp_addsrvrolemember @loginame = @username, @rolename =
N'sysadmin'
end
declare @classid int set @classid = (select ID from Classes where Name = @classname)
if not exists (select Username from Students where Username=@username) and not exists
(select Username from Accounts where Username=@username)
begin
begin transaction
insert into Accounts values (@username, 'Mtl@091202', 3);
insert into Students values (@username, @dateofbirth, @address, @name, @email, @phone);
insert into Class_Students values (@classid, @username);
insert into Payments values (getdate(), 0, 1, 0, @username);
exec addStudentAccountToServer @username
commit transaction
end
else
begin
if not exists (select * from Class_Students where Class_ID = @classid and
Username=@username)
begin
begin transaction
insert into Class_Students values (@classid, @username);
insert into Payments values (GETDATE(), 0, 1, 0, @username);
commit transaction
end
else
begin
DECLARE @CustomMessage VARCHAR(1000),
@CustomError INT,
@CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
SET @CustomError = 54321;
SET @CustomState = 1;
rollback;
THROW @CustomError, @CustomMessage, @CustomState;
end
end
if @check = 1
EXEC master..sp_dropsrvrolemember @loginame = @username, @rolename = N'sysadmin'
end
go
```

Cập nhật thông tin học viên:

```
create procedure UpdateStudent @username varchar(100), @name nvarchar(50), @dateofbirth
date, @address nvarchar(100), @email varchar(50), @phone varchar(11)
as
begin
if exists (select Accounts.Username from Accounts where Username = @username)
begin
begin transaction
update Students
set Name=@name, Date_Birth=@dateofbirth, Address=@address, Email = @email, Phone=@phone
where Username=@username
commit transaction
end
else
begin
DECLARE @CustomMessage VARCHAR(1000),
        @CustomError INT,
        @CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
SET @CustomError = 54321;
SET @CustomState = 1;
THROW @CustomError, @CustomMessage, @CustomState;
end
end
go
```

Procedure + transaction xóa thông tin nhân viên:

```
create procedure deleteACCOUNT_STAFF_sequentially @username nvarchar(50)
as
begin try
begin transaction
delete from Staff where Username=@username;
delete from Accounts where Username = @username;
exec DeleleUserOnServer @username;
commit transaction
end try
begin catch
DECLARE @CustomMessage VARCHAR(1000),
        @CustomError INT,
        @CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
SET @CustomError = 54321;
SET @CustomState = 1;
THROW @CustomError, @CustomMessage, @CustomState;
end catch
go
```

Lấy ra thông tin nhân viên thông qua tên nhân viên:

```
create procedure GetStaffByStaffName @name nvarchar(50) as select * from Staff_Info
where StaffName =@name
go
```

Lấy thông tin nhân viên thông qua vị trí làm việc ở trung tâm:

```
create procedure GetStaffByPosition @name nvarchar(50)
as select * from Staff_Info where Position =@name
go
```

Thêm nhân viên vào SQL Server:

```
create procedure addStaffAccountToServer @username varchar(30), @position varchar(30)
as begin
    declare @login nvarchar(4000)
    set @login = N'CREATE LOGIN ' + QUOTENAME(@username) + ' WITH PASSWORD = ' +
    QUOTENAME('Mtl@091202', '') + ', default_database = ' + QUOTENAME('LanguageCenter')
    exec(@login)
    declare @user nvarchar(4000)
    set @user = N'CREATE USER ' + QUOTENAME(@username) + ' FOR LOGIN ' +
    QUOTENAME(@username)
    exec(@user)
    if @position = 'Admin'
        begin
            EXEC sp_addrolemember 'Administrator', @username
            EXEC master..sp_addsrvrolemember @loginame = @username, @rolename =
            N'sysadmin'
        end
    else
        EXEC sp_addrolemember 'Staff', @username
end
go
```

Thêm thông tin nhân viên:

```
create procedure AddStaff @username varchar(100), @name nvarchar(50), @dateofbirth
date, @address nvarchar(100),
@email varchar(50), @phone varchar(11), @position nvarchar(40)
as
begin try
begin transaction
insert into Accounts values (@username, 'Mtl@091202', 1);
insert into Staff values (@username, @dateofbirth, @address, @name, @email, @phone,
@position);
exec addStaffAccountToServer @username, @position
commit transaction
end try
begin catch
DECLARE @CustomMessage VARCHAR(1000),
        @CustomError INT,
        @CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
SET @CustomError = 54321;
SET @CustomState = 1;
rollback;
THROW @CustomError, @CustomMessage, @CustomState;
end catch
```

Cập nhật thông tin nhân viên:

```
create procedure UpdateStaff @username varchar(100), @name nvarchar(50), @dateofbirth
date, @address nvarchar(100), @email varchar(50), @phone varchar(11),
@position nvarchar(40) as
begin
if exists (select Accounts.Username from Accounts where Username = @username)
begin
update Staff
set Name=@name, Date_Birth=@dateofbirth, Address=@address, Email = @email,
Phone=@phone, Position=@position where Username=@username
end
else
begin
DECLARE @CustomMessage VARCHAR(1000),
        @CustomError INT,
        @CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
SET @CustomError = 54321;
SET @CustomState = 1;
THROW @CustomError, @CustomMessage, @CustomState;
end
end
go
```

Lấy thông tin thanh toán:

```
create procedure getPayments
as begin
select * from PaymentsView
end
```

Thêm thông tin thanh toán:

```
create procedure InsertPayment @payment_date date, @amount int, @method_id int, @status
int, @username nvarchar(30)
as
begin try
insert into Payments values(@payment_date, @amount, @method_id, @status, @username)
end try
begin catch
DECLARE @CustomMessage VARCHAR(1000),
        @CustomError INT,
        @CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
SET @CustomError = 54321;
SET @CustomState = 1;
THROW @CustomError, @CustomMessage, @CustomState;
end catch
go
```

Cập nhật thông tin thanh toán:

```
create procedure updatePayment @id int ,@payment_date date, @amount int, @method_id
int, @username nvarchar(30)
as
begin try
update Payments
set Payment_Date = @payment_date, Amount = @amount, Payment_Method_ID = @method_id,
Username = @username
where ID = @id
end try
begin catch
DECLARE @CustomMessage VARCHAR(1000),
        @CustomError INT,
        @CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
SET @CustomError = 54321;
SET @CustomState = 1;
THROW @CustomError, @CustomMessage, @CustomState;
end catch
go
```

Xóa thông tin thanh toán:

```
create procedure deletePayment @id int
as begin
delete from Payments where Payments.ID = @id
end
go
```

Lấy thông tin tất cả lớp học của giảng viên:

```
create procedure getAllClasses
as
select * from Class_Info
go
```

Lấy ra thông tin lớp học thông qua ID lớp học:

```
create procedure GetClassByClassID @id int
as begin
select * from ClassesView where ClassesView.ID = @id
end
go
```

Lấy thông tin thời khóa biểu của học viên cụ thể:

```
create procedure getScheduleStudent (@name varchar(100))
as begin
select * from StudentScheduleView where StudentScheduleView.Username= @name
end
go
```

Lấy thông tin thời khóa biểu của giảng viên cụ thể:

```
create procedure getScheduleTeacher (@name varchar(100))
as begin select * from TeacherScheduleView where TeacherScheduleView.Username = @name
end
go
```

Lấy thông tin lịch sử giao dịch của học viên cụ thể:

```
create procedure GetTransactionHistory (@name varchar(100))
as begin select * from PaymentView where PaymentView.Username = @name
end
go
```

Lấy thông tin lịch sử giao dịch của học viên cụ thể thông qua tên học viên đó:

```
create procedure GetPaymentBYStudentName @name nvarchar(30)
as begin select * from PaymentsView where PaymentsView.StudentName = @name
end
go
```

Lấy thông tin lịch sử giao dịch của học viên cụ thể thông qua số điện thoại:

```
create procedure GetPaymentBYPhone @phone nvarchar(10)
as begin select * from PaymentsView where PaymentsView.Phone = @phone
end
go
```

Lấy thông tin lịch sử giao dịch của học viên cụ thể thông qua phương thức thanh toán:

```
create procedure GetPaymentBYPaymentMethod @method nvarchar(30)
as begin select * from PaymentsView where PaymentsView.PaymentMethod = @method
end
go
```

Lấy thông tin lịch sử giao dịch của học viên cụ thể thông qua trạng thái thanh toán:

```
create procedure GetPaymentByPaymentStatus @status nvarchar(30)
as begin
if(@status = N'Paid' or @status = N'Unpaid')
select * from PaymentsView where PaymentsView.PaymentStatus = @status
else
begin
DECLARE @CustomMessage VARCHAR(1000), CustomError INT, @CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE(); SET @CustomError = 54321;
SET @CustomState = 1; THROW @CustomError, @CustomMessage, @CustomState;
end
end
go
```

Đổi Password của tài khoản:

```
create procedure ChangePassword @username varchar(100), @pass varchar(100)
as
begin try
    declare @check int
    set @check = 0
    declare @oldPass varchar(100)
    select @oldPass = Accounts.Password from Accounts where Username = @username
    update Accounts set Accounts.Password = @pass where Accounts.Username =
@username
    if IS_MEMBER('sysadmin') = 0
        begin
            set @check = 1
            EXEC master..sp_addsrvrolemember @loginame = @username, @rolename =
N'sysadmin'
        end
    declare @query varchar(100)
    set @query = 'ALTER LOGIN ' + QUOTENAME(@username) + ' WITH PASSWORD = ' + ''' +
@pass + ''' + ' OLD_PASSWORD = ' + ''' + @oldPass + ''';
    print @query
    exec(@query)
    if @check = 1
        EXEC master..sp_dropsrvrolemember @loginame = @username, @rolename =
N'sysadmin'
end try
begin catch
    declare @err_mess varchar(1000);
    set @err_mess = 'Error ' + ERROR_MESSAGE();
    PRINT @err_mess;
    THROW;
    rollback
end catch
go
```


CHƯƠNG 3. THIẾT KẾ CÁC CHỨC NĂNG

3.1 Kết nối cơ sở dữ liệu

```
private static SqlConnection conn = null;

public static SqlConnection getConnection()
{
    if (conn == null)
    {
        try
        {
            string userName = Login.username;
            string passWord = Login.password;
            string connectionString = string.Format(@"Data
Source=LEMINHTUONG;Initial Catalog=LanguageCenter;User ID={0};Password={1}",
userName, passWord);
            conn = new SqlConnection(connectionString);
            conn.Open();
        }
        catch (Exception){}
    }
    return conn;
}
```

3.2 Quản lí nhân viên

Procedure + transaction xóa thông tin nhân viên:

```
create procedure deleteACCOUNT_STAFF_sequentially @username nvarchar(50)
as
begin try
    begin transaction
        delete from Staff where Username=@username;
        delete from Accounts where Username = @username;
        exec DeleleUserOnServer @username;
        commit transaction
    end try
    begin catch
        DECLARE @CustomMessage VARCHAR(1000),
                @CustomError INT,
                @CustomState INT;
        SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
        SET @CustomError = 54321;
        SET @CustomState = 1;
        THROW @CustomError, @CustomMessage, @CustomState;
    end catch
go
```

Lấy ra thông tin nhân viên thông qua tên nhân viên:

```
create procedure GetStaffByStaffName @name nvarchar(50)
as
select * from Staff_Info where StaffName =@name
go
```

Lấy thông tin nhân viên thông qua vị trí làm việc ở trung tâm:

```
create procedure GetStaffByPosition @name nvarchar(50)
as
select * from Staff_Info where Position =@name
go
```

Thêm nhân viên vào SQL Server:

```
create procedure addStaffAccountToServer @username varchar(30), @position varchar(30)
as begin
    declare @login nvarchar(4000)
    set @login = N'CREATE LOGIN ' + QUOTENAME(@username) + ' WITH PASSWORD = ' +
    QUOTENAME('Mtl@091202', ''') + ', default_database = ' + QUOTENAME('LanguageCenter')
    exec(@login)
    declare @user nvarchar(4000)
    set @user = N'CREATE USER ' + QUOTENAME(@username) + ' FOR LOGIN ' +
    QUOTENAME(@username)
    exec(@user)

    if @position = 'Admin'
        begin
            EXEC sp_addrolemember 'Administrator', @username
            EXEC master..sp_addsrvrolemember @loginame = @username, @rolename =
            N'sysadmin'
        end
    else
        EXEC sp_addrolemember 'Staff', @username
    end
go
```

Thêm thông tin nhân viên:

```
create procedure AddStaff @username varchar(100), @name nvarchar(50), @dateofbirth
date, @address nvarchar(100),
@email varchar(50), @phone varchar(11), @position nvarchar(40)
as
begin try
    begin transaction
        insert into Accounts values (@username, 'Mtl@091202', 1);
        insert into Staff values (@username, @dateofbirth, @address, @name, @email,
        @phone, @position);
        exec addStaffAccountToServer @username, @position
        commit transaction
    end try
    begin catch
        DECLARE @CustomMessage VARCHAR(1000),
                @CustomError INT,
                @CustomState INT;
        SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
        SET @CustomError = 54321;
        SET @CustomState = 1;
        rollback;
        THROW @CustomError, @CustomMessage, @CustomState;
    end catch
go
```

Cập nhật thông tin nhân viên:

```
create procedure UpdateStaff @username varchar(100), @name nvarchar(50), @dateofbirth
date, @address nvarchar(100), @email varchar(50), @phone varchar(11),
@position nvarchar(40) as
begin
    if exists (select Accounts.Username from Accounts where Username = @username)
        begin try
            update Staff
            set Name=@name, Date_Birth=@dateofbirth, Address=@address, Email =
@email, Phone=@phone, Position=@position where Username=@username
        end try
        begin catch
            DECLARE @CustomMessage VARCHAR(1000),
                    @CustomError INT,
                    @CustomState INT;
            SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
            SET @CustomError = 54321;
            SET @CustomState = 1;
            THROW @CustomError, @CustomMessage, @CustomState;
        end catch
    else
        THROW 51000, 'Username does not exist.', 1;
end
```

Gán lương cứng cho các vị trí của nhân viên của trung tâm:

```
create function Tinh_Luong (@position as nvarchar(40))
returns int
as begin
    declare @result int
    if @position = 'Admin'
        set @result = 12000000
    if @position = 'HR'
        set @result = 10000000
    if @position = 'Marketing'
        set @result = 9000000
    if @position = 'Sales' or @position = 'Trainer'
        set @result = 7000000
    if @position = 'Receptionist'
        set @result = 4000000
    return @result
end;
```

Chọn ra những thuộc tính kết hợp của 1 nhân viên:

```
create view Staff_Info as
select Username, Name as StaffName, convert (varchar(100),Date_Birth, 103) as
DateOfBirth, Address, Email, Phone, Position, dbo.Tinh_Luong(Position) as Salary
from Staff
go
```

Xóa User khỏi SQL Server:

```
create procedure DeleleUserOnServer @username varchar(30)
as begin
    begin try
        declare @user nvarchar(4000)
        set @user = N'DROP USER' + QUOTENAME(@username) + ';'
        exec(@user)
        declare @login nvarchar(4000)
        set @login = N'DROP LOGIN ' + QUOTENAME(@username) + ';'
        exec(@login)
    end try
    begin catch
        rollback
    end catch
end
```

3.3 Quản lý giảng viên:

Chọn ra những thuộc tính kết hợp của 1 giảng viên:

```
create view Teacher_Info as
select Teachers.Username, Teachers.Name as TeacherName,convert
(varchar(100),Date_Birth, 103) as DateOfBirth, Address, Email,
Phone, ISNULL(LUONG,0) as Salary from Teachers left join (select *,
dbo.TinhLuong(A.HSL) as LUONG from (select Username, count (*) as HSL from Classes
group by Username)A)B on
B.Username= Teachers.Username
--
```

Chọn ra những thuộc tính kết hợp của giảng viên bao gồm cả việc tính lương:

```
create view TeacherAndSalary as
select Teachers.Username, Teachers.Name as TeacherName,convert
(varchar(100),Date_Birth, 103) as DateOfBirth, Address, Email,
Phone, ISNULL(LUONG,0) as Salary, Classes.Name as ClassName, Courses.Name as CourseName
from Teachers left join (select *,
dbo.TinhLuong(A.HSL) as LUONG from (select Username, count (*) as HSL from Classes
group by Username)A)B on
B.Username= Teachers.Username inner join Classes on B.Username = Classes.Username inner
join Courses
on Classes.Course_ID=Courses.ID;
go
```

Tính lương của giảng viên theo công thức: $5000000 * (\text{số lớp giảng viên dạy})$

```
create function TinhLuong (@noclass as int)
returns int
as begin
declare @result int = 5000000*@noclass
if @noclass is null
set @result =0
return @result
end;
go
```

Thêm giảng viên mới vào SQL Server:

```
create procedure addTeacherAccountToServer @username varchar(30)
as begin
    begin try
        declare @login nvarchar(4000)
        set @login = N'CREATE LOGIN ' + QUOTENAME(@username) + ' WITH PASSWORD =
' + QUOTENAME('Mtl@091202', '') + ', default_database = ' +
QUOTENAME('LanguageCenter')
        exec(@login)
        declare @user nvarchar(4000)
        set @user = N'CREATE USER ' + QUOTENAME(@username) + ' FOR LOGIN ' +
QUOTENAME(@username)
        exec(@user)
        exec sp_addrolemember 'Teacher', @username
    end try
    begin catch
        rollback
    end catch
end
go
```

Xóa User khỏi SQL Server:

```
create procedure DeleleUserOnServer @username varchar(30)
as begin
    declare @user nvarchar(4000)
    set @user = N'DROP USER ' + QUOTENAME(@username) + ';'
    exec(@user)
    declare @login nvarchar(4000)
    set @login = N'DROP LOGIN ' + QUOTENAME(@username) + ';'
    exec(@login)
end
go
```

Xóa thông tin giảng viên:

```
create procedure deleteACCOUNT_TEACHER_sequentially @username nvarchar(50)
as
if exists (select Accounts.Username from Accounts where Username = @username)
begin
    begin transaction
    delete from Teachers where Username=@username;
    delete from Accounts where Username = @username;
    exec DeleleUserOnServer @username;
    commit transaction
end
else
```

```

begin
DECLARE @CustomMessage VARCHAR(1000),
        @CustomError INT,
        @CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
SET @CustomError = 54321;
SET @CustomState = 1;
rollback;
THROW @CustomError, @CustomMessage, @CustomState;
end
go

```

Lấy ra thông tin giảng viên thông qua tên giảng viên:

```

create procedure GetTeacherByTeacherName @name nvarchar(50)
as
begin
select distinct * from Teacher_Info where TeacherName= @name
end
go

```

Lấy ra thông tin giảng viên thông qua tên khóa học:

```

create procedure GetTeacherByCourseName @name nvarchar(50)
as
begin
select distinct Username, TeacherName, DateOfBirth, Address, Email, Phone, Salary from
TeacherAndSalary where CourseName= @name
end
go

```

Lấy ra thông tin giảng viên thông qua tên lớp học:

```

create procedure GetTeacherByClassName @name nvarchar(50)
as
begin
select distinct Username, TeacherName, DateOfBirth, Address, Email, Phone, Salary from
TeacherAndSalary where ClassName= @name
end
go

```


Thêm thông tin giảng viên:

```
create procedure AddTeacher @username varchar(100), @name nvarchar(50), @dateofbirth
date, @address nvarchar(100),
@email varchar(50), @phone varchar(11)
as
begin
    begin try
        if IS_MEMBER('sysadmin') = 0
            begin
                EXEC master..sp_addsrvrolemember @loginame =
@username, @rolename = N'sysadmin'
            end
        begin transaction
            insert into Accounts values (@username, 'Mt1@091202', 2);
            insert into Teachers values (@username, @dateofbirth, @address, @name,
@email, @phone);
            exec addTeacherAccountToServer @username
            EXEC master..sp_dropsrvrolemember @loginame = @username, @rolename =
N'sysadmin'
            commit transaction
        end try
        begin catch
            DECLARE @CustomMessage VARCHAR(1000),
                @CustomError INT,
                @CustomState INT;
            SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
            SET @CustomError = 54321;
            SET @CustomState = 1;
            ROLLBACK;
            THROW @CustomError, @CustomMessage, @CustomState;
        end catch
    end
go
```

Cập nhật thông tin giảng viên:

```
create procedure UpdateTeacher @username varchar(100), @name nvarchar(50), @dateofbirth
date, @address nvarchar(100), @email varchar(50), @phone varchar(11)
as
begin
    if exists (select Accounts.Username from Accounts where Username = @username)
        begin
            update Teachers
            set Name=@name, Date_Birth=@dateofbirth, Address=@address, Email = @email, Phone=@phone
            where Username=@username
        end
    else begin
        DECLARE @CustomMessage VARCHAR(1000),
            @CustomError INT,
            @CustomState INT;
        SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
        SET @CustomError = 54321;
        SET @CustomState = 1;
        THROW @CustomError, @CustomMessage, @CustomState;
    end
end
go
```

3.4 Quản lí học viên

Xóa User khỏi SQL Server:

```
create procedure DeleleUserOnServer @username varchar(30)
as begin
    begin try
        declare @user nvarchar(4000)
        set @user = N'DROP USER' + QUOTENAME(@username) + ';'
        exec(@user)
        declare @login nvarchar(4000)
        set @login = N'DROP LOGIN ' + QUOTENAME(@username) + ';'
        exec(@login)
    end try
    begin catch
        rollback
    end catch
end
go
```

Chọn ra những thuộc tính kết hợp của 1 học viên:

```
create view Student_Info as
select Students.Username, Students.Name as StudentName, convert
(varchar(100),Students.Date_Birth, 103) as DateOfBirth,
Students.Address, Students.Email, Students.Phone, Courses.Name as CourseName,
Classes.Name as ClassName, Teachers.Name as TeacherName
from Students
inner join Class_Students on Students.Username=Class_Students.Username
inner join Classes on Class_Students.Class_ID=Classes.ID
inner join Teachers on Classes.Username=Teachers.Username
inner join Courses on Course_ID = Courses.ID;
go
```

Xóa thông tin học viên:

```
create procedure deleteACCOUNT_STUDENT_sequentially @username nvarchar(50)
as
if exists (select Accounts.Username from Accounts where Username = @username)
begin
    begin transaction
    delete from Class_Students where Class_Students.Username = @username ;
    delete from Payments where Payments.Username = @username;
    delete from Students where Students.Username = @username;
    delete from Accounts where Accounts.Username = @username;
    exec DeleleUserOnServer @username;
    commit transaction
end
```

```

else
    begin
        DECLARE @CustomMessage VARCHAR(1000),
                @CustomError INT,
                @CustomState INT;
        SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
        SET @CustomError = 54321;
        SET @CustomState = 1;
        rollback;
        THROW @CustomError, @CustomMessage, @CustomState;
    end
go

```

Liệt kê một vài thông tin của học viên:

```

create procedure GetListStudent as
select Username, StudentName, DateOfBirth, Address, Email, Phone, ClassName
from Student_Info;
go

```

Lấy thông tin học viên thông qua tên học viên:

```

create procedure GetStudentByStudentName @name nvarchar(50)
as begin
select distinct Username, StudentName, DateOfBirth, Address, Email, Phone, ClassName
from Student_Info
where StudentName= @name
end
go

```

Lấy thông tin học viên thông qua tên giảng viên của học viên đó:

```

create procedure GetStudentByTeacherName @name nvarchar(50)
as begin
select distinct Username, StudentName, DateOfBirth, Address, Email, Phone, ClassName
from Student_Info
where Student_Info.TeacherName = @name
end
go

```

Lấy thông tin học viên thông qua tên khóa học:

```

create procedure GetStudentByCourseName @name nvarchar(50)
as
begin
select distinct Username, StudentName, DateOfBirth, Address, Email, Phone, ClassName
from Student_Info
where Student_Info.CourseName = @name
end
go

```

Lấy thông tin học viên thông qua tên lớp học:

```
create procedure GetStudentByClassName @name nvarchar(50)
as
begin
select distinct Username, StudentName, DateOfBirth, Address, Email, Phone from
Student_Info
where Student_Info.ClassName = @name
end
goc
```

Thêm thông tin học viên:

```
create procedure AddStudent @username varchar(100), @name nvarchar(50), @dateofbirth
date, @address nvarchar(100),
@email varchar(50), @phone varchar(11), @classname nvarchar(100)
as
begin
declare @check bit = 0
if IS_MEMBER('sysadmin') = 0
begin
set @check = 1
EXEC master..sp_addsrvrolemember @loginame = @username, @rolename =
N'sysadmin'
end
declare @classid int set @classid = (select ID from Classes where Name = @classname)
if not exists (select Username from Students where Username=@username) and not exists
(select Username from Accounts where Username=@username)
begin
begin transaction
insert into Accounts values (@username, 'Mtl@091202', 3);
insert into Students values (@username, @dateofbirth, @address, @name, @email, @phone);
insert into Class_Students values (@classid, @username);
insert into Payments values (getdate(), 0, 1, 0, @username);
exec addStudentAccountToServer @username
commit transaction
end
else
begin
if not exists (select * from Class_Students where Class_ID = @classid and
Username=@username)
begin
begin transaction
insert into Class_Students values (@classid, @username);
insert into Payments values (GETDATE(), 0, 1, 0, @username);
commit transaction
end
```

```

else
begin
DECLARE @CustomMessage VARCHAR(1000),
        @CustomError INT,
        @CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
SET @CustomError = 54321;
SET @CustomState = 1;
rollback;
THROW @CustomError, @CustomMessage, @CustomState;
end
end
if @check = 1
    EXEC master..sp_dropsrvrolemember @loginame = @username, @rolename = N'sysadmin'
end
go

```

Cập nhật thông tin học viên:

```

create procedure UpdateStudent @username varchar(100), @name nvarchar(50), @dateofbirth
date, @address nvarchar(100), @email varchar(50), @phone varchar(11), @classname
varchar(100)
as
begin
    if exists (select Accounts.Username from Accounts where Username = @username)
        begin try
            begin transaction
                declare @classid int = (select ID from Classes where Name =
@classname);
                update Class_Students
                set Class_ID= @classid where Class_Students.Username=@username
                update Students
                set Name=@name, Date_Birth=@dateofbirth, Address=@address, Email =
@email, Phone=@phone where Username=@username
                commit transaction
            end try
            begin catch
                DECLARE @CustomMessage VARCHAR(1000),
                        @CustomError INT,
                        @CustomState INT;
                SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
                SET @CustomError = 54321;
                SET @CustomState = 1;
                rollback;
                THROW @CustomError, @CustomMessage, @CustomState;
            end catch
        else
            THROW 51000, ' Username does not exist.', 1;
        end
end
go

```

Procedure + transaction xóa thông tin học viên ở 1 lớp học cụ thể:

```
create procedure deleteSTUDENT_View @username nvarchar(100), @classname nvarchar(100)
as
begin try
declare @classid int set @classid = (select ID from Classes where Name = @classname)
delete from Class_Students where Username = @username and Class_ID = @classid;
end try
begin catch
DECLARE @CustomMessage VARCHAR(1000), CustomError INT, CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE(); SET @CustomError = 54321;
SET @CustomState = 1; THROW @CustomError, @CustomMessage, @CustomState;
end catch
go
```

Thêm học viên vào SQL Server:

```
create procedure addStudentAccountToServer @username varchar(30)
as begin
    begin try
        declare @login nvarchar(4000)
        set @login = N'CREATE LOGIN ' + QUOTENAME(@username) + ' WITH PASSWORD = '
        + QUOTENAME('Mt1@091202', '') + ', default_database = ' + QUOTENAME('LanguageCenter')
        exec(@login)
        declare @user nvarchar(4000)
        set @user = N'CREATE USER ' + QUOTENAME(@username) + ' FOR LOGIN ' +
        QUOTENAME(@username)
        exec(@user)
        exec sp_addrolemember 'Student', @username
    end try
    begin catch
        rollback
    end catch
end
```

3.5 Đổi Password

Đổi Password của tài khoản:

```
create procedure ChangePassword @username varchar(100), @pass varchar(100)
as
begin try
    declare @check int
    set @check = 0
    declare @oldPass varchar(100)
    select @oldPass = Accounts.Password from Accounts where Username = @username
    update Accounts set Accounts.Password = @pass where Accounts.Username =
    @username
    if IS_MEMBER('sysadmin') = 0
        begin
            set @check = 1
            EXEC master..sp_addsrvrolemember @loginame = @username, @rolename =
            N'sysadmin'
        end
end
```

```

        declare @query varchar(100)
        set @query = 'ALTER LOGIN ' + QUOTENAME(@username) + ' WITH PASSWORD = ' + @pass + @oldPass + ' OLD_PASSWORD = ' + @oldPass;
        print @query
        exec(@query)
        if @check = 1
            EXEC master..sp_dropsrvrolemember @loginame = @username, @rolename =
N'sysadmin'
    end try
    begin catch
        declare @err_mess varchar(1000);
        set @err_mess = 'Error ' + ERROR_MESSAGE();
        PRINT @err_mess;
        THROW;
        rollback
    end catch
go

```

3.6 Quản lí khóa học

Thêm thông tin khóa học:

```
create procedure AddCourse @coursename nvarchar(40), @target float, @nolessons int,
@price int as
begin
begin try
insert into Courses values (@coursename, @target, @nolessons, @price);
end try
begin catch
DECLARE @CustomMessage VARCHAR(1000), CustomError INT, CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE(); SET @CustomError = 54321;
SET @CustomState = 1; THROW @CustomError, @CustomMessage, @CustomState;
end catch
end
go
```

Cập nhật thông tin khóa học:

```
create procedure UpdateCourse @courseid int , @coursename nvarchar(40), @target float,
@nolessons int, @price int as
begin
    if exists (select Courses.ID from Courses where Courses.ID = @courseid)
        begin
            update Courses
            set Name = @coursename, Target = @target, No_Lessons=@nolessons,
Price=@price where ID = @courseid
        end
    else
        begin
            DECLARE @CustomMessage VARCHAR(1000), @CustomError INT,
@CustomState INT;
            SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE(); SET
@CustomError = 54321; SET @CustomState = 1;
            THROW @CustomError, @CustomMessage, @CustomState;
        end
    end
end
go
```

Procedure + transaction xóa thông tin khóa học:

```
create procedure deleteCOURSE_sequentially @id int
as
begin try
begin transaction
delete from Classes where Classes.Course_ID = @id;
delete from Courses where Courses.ID = @id;
commit transaction
end try
begin catch
DECLARE @CustomMessage VARCHAR(1000), CustomError INT, CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE(); SET @CustomError = 54321;
SET @CustomState = 1; THROW @CustomError, @CustomMessage, @CustomState;
end catch
go
```


Lấy ra thông tin khóa học thông qua tên khóa học:

```
create procedure GetCourseByCourseName @courseName nvarchar(40) as
select * from Course_Info where CourseName= @courseName;
go
```

Lấy ra thông tin khóa học thông qua mức học phí cao nhất được nhập:

```
create procedure GetCourseByCourseName @courseName nvarchar(40) as
select * from Course_Info where CourseName= @courseName;
go
```

Lấy ra thông tin khóa học thông qua mức học phí thấp nhất được nhập:

```
create procedure GetCourseByMinPrice @price int as
begin try
    select * from Course_Info where Price >= @price;
end try
begin catch
    DECLARE @CustomMessage VARCHAR(1000),
            @CustomError INT,
            @CustomState INT;
    SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
    SET @CustomError = 54321;
    SET @CustomState = 1;
    THROW @CustomError, @CustomMessage, @CustomState;
end catch
go
```

Chọn ra những thuộc tính của 1 khóa học:

```
create view Course_Info as
select ID, Name as CourseName, ROUND(Target,1) as Target, No_Lessons as NoLessons,
Price from Courses
go
```

3.7 Quản lí lớp học

Chọn ra những thuộc tính kết hợp của 1 lớp học:

```
create view Class_Info as
select Classes.ID, Classes.Name as ClassName, convert (varchar(100),Start_Date, 103) as
StartDate, convert (varchar(100),End_Date, 103) as EndDate, WeekDays, convert
(vvarchar(100),Start_Time, 108) as StartTime,
convert (varchar(100),End_Time, 108) as EndTime, Classroom, No_Students as NoStudents,
Courses.Name as CourseName, ROUND(Target,1) as Target, Teachers.Name as TeacherName
from Classes left join Courses
on Classes.Course_ID=Courses.ID left join Teachers on
Teachers.Username=Classes.Username;
go
```

Thêm thông tin lớp học:

```
create procedure AddClass @classname nvarchar(100), @startdate date, @enddate date,
@weekdays varchar(10),
@starttime time(7), @endtime time(7), @classroom varchar(20), @coursename nvarchar(40),
@target float, @teachername nvarchar(50) as
begin
    declare @courseid int, @teacherusername varchar(100)
    set @courseid = (select Courses.ID from Courses where Name = @coursename and
Target = @target)
    set @teacherusername = (select Username from Teachers where Name = @teachername)
    begin try
        insert into Classes values (@classname, @startdate, @enddate,
@teacherusername, @courseid, @weekdays, @starttime, @endtime, @classroom, 0);
    end try
    begin catch
        DECLARE @CustomMessage VARCHAR(1000),
                @CustomError INT,
                @CustomState INT;
        SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
        SET @CustomError = 54321;
        SET @CustomState = 1;
        THROW @CustomError, @CustomMessage, @CustomState;
    end catch
end
go
```

Lấy ra thông tin lớp học thông qua tên khóa học:

```
create procedure GetClassBYCourseName @coursename nvarchar(40) as
select * from Class_Info where CourseName=@coursename
go
```

Lấy ra thông tin lớp học thông qua tên giảng viên:

```
create procedure GetClassBYTeacherName @teachername nvarchar(100) as
select * from Class_Info where TeacherName=@teachername
go
```

Cập nhật thông tin lớp học:

```
create procedure UpdateClass @classid int, @classname nvarchar(100), @startdate date,
@enddate date, @weekdays varchar(10), @starttime time(7), @endtime time(7),
@classroom varchar(20), @coursename nvarchar(40), @target float, @teachername
nvarchar(50) as
begin
    declare @courseid int, @teacherusername varchar(100)
    set @courseid = (select Courses.ID from Courses where Name = @coursename and
Target = @target)
    set @teacherusername = (select Username from Teachers where Name = @teachername)
    begin try
        update Classes
        set Name = @classname, Start_Date=@startdate, End_Date=@enddate,
Username=@teacherusername, Course_ID=@courseid, WeekDays=@weekdays,
Start_Time=@starttime,
        End_Time = @endtime, ClassRoom=@classroom where Classes.ID = @classid
    end try

    begin catch
        DECLARE @CustomMessage VARCHAR(1000),
                @CustomError INT,
                @CustomState INT;
        SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
        SET @CustomError = 54321;
        SET @CustomState = 1;
        THROW @CustomError, @CustomMessage, @CustomState;
    end catch
end
go
```

Xóa thông tin lớp học:

```
create procedure deleteCLASS_sequentially @id int
as
delete from Classes where ID = @id;
go
```

Lấy ra thông tin lớp học thông qua tên lớp học:

```
create procedure GetClassBYClassName @classname nvarchar(100) as
select * from Class_Info where ClassName=@classname
go
```

Lấy ra thông tin lớp học thông qua tên phòng học:

```
create procedure GetClassBYClassRoom @classroom varchar(20) as
select * from Class_Info where ClassRoom=@classroom
go
```

3.8 Quản lý thanh toán

Trả về tên của phương thức thanh toán bằng ID:

```
create function PaymentMethodName_byId (@id int)
returns nvarchar(20)
as begin
    declare @name nvarchar(20)
    set @name = (select Payment_Methods.Name from Payment_Methods where
Payment_Methods.ID = @id)
    return @name
end
go
```

Trả về thông tin trạng thái thanh toán theo dạng chữ:

```
create function TrangThaiThanhToan (@status bit)
returns nvarchar(30)
as begin
    declare @status_name nvarchar(30)
    set @status_name = ''
    if @status = 1
        begin
            set @status_name = N'Paid'
        end
    if @status = 0
        begin
            set @status_name = N'Unpaid'
        end
    return @status_name
end
go
```

Thêm thông tin thanh toán:

```
create procedure InsertPayment @payment_date date, @amount int, @method_id int, @status
int, @username nvarchar(30)
as
begin try
    insert into Payments values(@payment_date, @amount, @method_id, @status,
@username)
end try
begin catch
    DECLARE @CustomMessage VARCHAR(1000),
            @CustomError INT,
            @CustomState INT;
    SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
    SET @CustomError = 54321;
    SET @CustomState = 1;
    THROW @CustomError, @CustomMessage, @CustomState;
end catch
go
```

Cập nhật thông tin thanh toán:

```
create procedure updatePayment @id int ,@payment_date date, @amount int, @method_id
int, @username nvarchar(30)
as
begin try
    update Payments
    set Payment_Date = @payment_date, Amount = @amount, Payment_Method_ID =
@method_id, Username = @username
    where ID = @id
end try
begin catch
    DECLARE @CustomMessage VARCHAR(1000),
            @CustomError INT,
            @CustomState INT;
    SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
    SET @CustomError = 54321;
    SET @CustomState = 1;
    THROW @CustomError, @CustomMessage, @CustomState;
end catch
go
```

Xóa thông tin thanh toán:

```
create procedure deletePayment @id int
as begin
delete from Payments where Payments.ID = @id
end
go
```

Lấy thông tin thanh toán:

```
create procedure getPayments
as begin
select * from PaymentsView
end
go
```

Quản lý thanh toán chung của tất cả học viên:

```
create view PaymentsView
as
select ID ,Students.Username,Students.Name as StudentName, Students.Email,
Students.Phone as Phone,
Payments.Payment_Date as PaymentDate, Amount as Amount ,
[dbo].PaymentMethodName_byId(Payments.Payment_Method_ID) as PaymentMethod,
[dbo].TrangThaiThanhToan(Payments.Status) as PaymentStatus
from Payments inner join Students on Students.Username = Payments.Username
go
```


Lấy thông tin lịch sử giao dịch của học viên cụ thể thông qua tên học viên đó:

```
create procedure GetPaymentBYStudentName @name nvarchar(30)
as begin
select * from PaymentsView where PaymentsView.StudentName = @name
end
go
```

Lấy thông tin lịch sử giao dịch của học viên cụ thể thông qua số điện thoại:

```
create procedure GetPaymentBYPhone @phone nvarchar(10)
as begin
select * from PaymentsView where PaymentsView.Phone = @phone
end
go
```

Lấy thông tin lịch sử giao dịch của học viên cụ thể thông qua phương thức thanh toán:

```
create procedure GetPaymentBYPaymentMethod @method nvarchar(30)
as begin
select * from PaymentsView where PaymentsView.PaymentMethod = @method
end
go
```

Lấy thông tin lịch sử giao dịch của học viên cụ thể thông qua trạng thái thanh toán:

```
create procedure GetPaymentByPaymentStatus @status nvarchar(30)
as begin
if(@status = N'Paid' or @status = N'Unpaid')
begin
select * from PaymentsView where PaymentsView.PaymentStatus = @status
end
else
begin
DECLARE @CustomMessage VARCHAR(1000),
        @CustomError INT,
        @CustomState INT;
SET @CustomMessage = 'My Custom Text ' + ERROR_MESSAGE();
SET @CustomError = 54321;
SET @CustomState = 1;
THROW @CustomError, @CustomMessage, @CustomState;
end
end
go
```

Tìm tên khóa học bằng ID:

```
create function getCourseName_byID (@id int)
returns nvarchar(30)
as begin
declare @name nvarchar(30)
set @name = (select Courses.Name from Courses where Courses.ID = 1)
return @name
end
```

Tìm tên giảng viên bằng username:

```
create function getTeacherName_byUsername (@username nvarchar(30))
returns nvarchar(30)
as begin
declare @name nvarchar(30)
set @name = (select Teachers.Name from Teachers where Teachers.Username = @username)
return @name
end
```

Trả về bảng chứa thông tin thanh toán:

```
create function getPayments_func()
returns table
as
return (
    select ID ,Students.Username,Students.Name as StudentName, Students.Email,
    Students.Phone as Phone,
    Payments.Payment_Date as PaymentDate, Amount as Amount,
    [dbo].PaymentMethodName_byId(Payments.Payment_Method_ID) as PaymentMethod,
    [dbo].TrangThaiThanhToan(Payments.Status) as PaymentStatus
    from Payments inner join Students on Students.Username = Payments.Username
)
go
```


3.9 Xem danh sách lớp học với vai trò là giảng viên

Lấy thông tin tất cả lớp học của giảng viên:

```
create procedure getAllClasses
as
select * from Class_Info
go
```

Lấy ra thông tin lớp học thông qua ID lớp học:

```
create procedure GetClassByClassID @id int
as begin
select * from ClassesView where ClassesView.ID = @id
end
go
```

Lấy ra thông tin lớp học thông qua tên giảng viên:

```
create procedure GetClassBYTeacherName @teachername nvarchar(100) as
select * from Class_Info where TeacherName=@teachername
go
```

Lấy ra thông tin lớp học thông qua tên lớp học:

```
create procedure GetClassBYClassName @classname nvarchar(100) as
select * from Class_Info where ClassName=@classname
go
```

Lấy ra thông tin lớp học thông qua tên khóa học:

```
create procedure GetClassBYCourseName @coursename nvarchar(40) as
select * from Class_Info where CourseName=@coursename
go
```

Chọn ra những thuộc tính của lớp học:

```
create view ClassesView
as
select Classes.ID, Classes.Name as ClassName, Classes.Start_Date as StartDate,
Classes.End_Date as EndDate, [dbo].getTeacherName_byUsername(Classes.Username)
as TeacherName, [dbo].getCourseName_byID(Classes.Course_ID) as CourseName,
Classes.WeekDays, Classes.Start_Time, Classes.End_Time, Classes.ClassRoom,
Classes.No_Students
from Classes
go
```

3.10 Xem thời khóa biểu với vai trò là giảng viên

Lấy thông tin thời khóa biểu của giảng viên cụ thể:

```
create procedure getScheduleTeacher (@name varchar(100))
as begin
select * from TeacherScheduleView where TeacherScheduleView.Username = @name
end
go
```

Thời khóa biểu của tất cả giảng viên:

```
create view TeacherScheduleView
as
select t.Username as Username, cl.ID as ClassID, cl.Name as ClassName, cl.Course_ID as
CourseID, c.Name as CourseName ,
cl.WeekDays as Date, CONCAT(SUBSTRING(convert(varchar, cl.Start_Time ,108),1,5), ' :
' ,SUBSTRING(convert(varchar, cl.End_Time ,108),1,5)) as Time, cl.ClassRoom as
ClassRoom
from Classes cl inner join Courses c on cl.Course_ID = c.ID inner join Teachers t on
cl.Username = t.Username
go
```

3.11 Xem thời khóa biểu với vai trò là học viên

Thời khóa biểu của tất cả học viên:

```
create view StudentScheduleView
as
select cs.Username as Username, cl.ID as ClassID, cl.Name as ClassName, c.Name as
CourseName , t.Name as TeacherName,
cl.WeekDays, CONCAT(SUBSTRING(convert(varchar, cl.Start_Time ,108),1,5), ' :
' ,SUBSTRING(convert(varchar, cl.End_Time ,108),1,5)) as Time, cl.ClassRoom as
ClassRoom
from Class_Students cs inner join Classes cl on cs.Class_ID = cl.ID inner join Courses
c on cl.Course_ID = c.ID inner join Teachers t on cl.Username = t.Username
go
```

Lấy thông tin thời khóa biểu của học viên cụ thể:

```
create procedure getScheduleStudent (@name varchar(100))
as begin
select * from StudentScheduleView where StudentScheduleView.Username= @name
end
go
```

3.12 Xem lịch sử giao dịch với vai trò là học viên

Lịch sử giao dịch của tất cả học viên:

```
create view PaymentView
as
select st.Username, pa.Payment_Date as PaymentDate, pa.Amount as Amount, pm.Name as
PaymentMethod, pa.Status as PaymentStatus
from Students st inner join Payments pa on st.Username = pa.Username inner join
Payment_Methods pm on pa.Payment_Method_ID = pm.ID
go
```

Lấy thông tin lịch sử giao dịch của học viên cụ thể:

```
create procedure GetTransactionHistory (@name varchar(100))
as begin
select * from PaymentView where PaymentView.Username = @name
end
go
```

CHƯƠNG 4. TẠO USER VÀ PHÂN QUYỀN

4.1 Tạo role

Khởi tạo tài khoản sa → tài khoản admin ban đầu

```
insert into Accounts values ('sa', 'Mtl@091202', 1);  
insert into Staff values ('sa', '2012-10-25', 'Tien Giang', 'Le Minh Tuong',  
'lmt2002@gmail.com', '0834091202', 'Admin');
```

Tạo các Role

```
create role Administrator  
create role Staff  
create role Teacher  
create role Student
```

4.2 Phân quyền

Phân toàn quyền quản lý toàn hệ thống cho quản trị viên

```
GRANT ALTER, VIEW DEFINITION, EXECUTE TO Administrator
```

Phân quyền toàn bộ cho nhân viên

```
GRANT ALTER, VIEW DEFINITION, EXECUTE TO Staff
```

Phân quyền mức procedure cho nhân viên

```
REVOKE EXECUTE on dbo.deleteACCOUNT_STAFF_sequentially to Staff
REVOKE EXECUTE on dbo.GetStaffByStaffName to Staff
REVOKE EXECUTE on dbo.GetStaffByPosition to Staff
REVOKE EXECUTE on dbo.addStaffAccountToServer to Staff
REVOKE EXECUTE on dbo.AddStaff to Staff
REVOKE EXECUTE on dbo.UpdateStaff to Staff
REVOKE EXECUTE on dbo.Tinh_Luong to Staff
GRANT ALTER, VIEW DEFINITION, SELECT on dbo.Staff_Info to Staff
```

Phân quyền trên các view cho giảng viên

```
GRANT SELECT on dbo.Class_Students to Teacher
GRANT SELECT on dbo.Classes to Teacher
GRANT SELECT on dbo.Courses to Teacher
GRANT SELECT on dbo.Roles to Teacher
GRANT SELECT on dbo.Students to Teacher
GRANT SELECT on dbo.Teachers to Teacher
GRANT SELECT on dbo.Accounts to Teacher
GRANT ALTER, VIEW DEFINITION, SELECT on dbo.TeacherScheduleView to Teacher
```

Phân quyền trên các procedures cho giảng viên

```
GRANT EXECUTE on dbo.getScheduleTeacher to Teacher
GRANT EXECUTE on dbo.ChangePassword to Teacher
GRANT EXECUTE on dbo.getAllClasses to Teacher
GRANT EXECUTE on dbo.GetClassBYTeacherName to Teacher
GRANT EXECUTE on dbo.GetClassByClassID to Teacher
GRANT EXECUTE on dbo.GetClassByClassName to Teacher
GRANT EXECUTE on dbo.GetClassByCourseName to Teacher
```

Phân quyền trên các view cho học viên

```
GRANT SELECT on dbo.Class_Students to Student
GRANT SELECT on dbo.Accounts to Student
GRANT SELECT on dbo.Classes to Student
GRANT SELECT on dbo.Courses to Student
GRANT SELECT on dbo.Roles to Student
GRANT SELECT on dbo.Students to Student
GRANT SELECT on dbo.Teachers to Student
GRANT SELECT on dbo.Payment_Methods to Student
GRANT SELECT on dbo.Payments to Student
GRANT ALTER, VIEW DEFINITION, SELECT on dbo.StudentScheduleView to Student
```

Phân quyền trên các procedures cho học viên


```
GRANT EXECUTE on dbo.getScheduleStudent to Student
GRANT EXECUTE on dbo.GetTransactionHistory to Student
GRANT EXECUTE on dbo.ChangePassword to Student
```


CHƯƠNG 5. GIAO DIỆN HỆ THỐNG

5.1 Form Login

Login

×

**LANGUAGE CENTER**



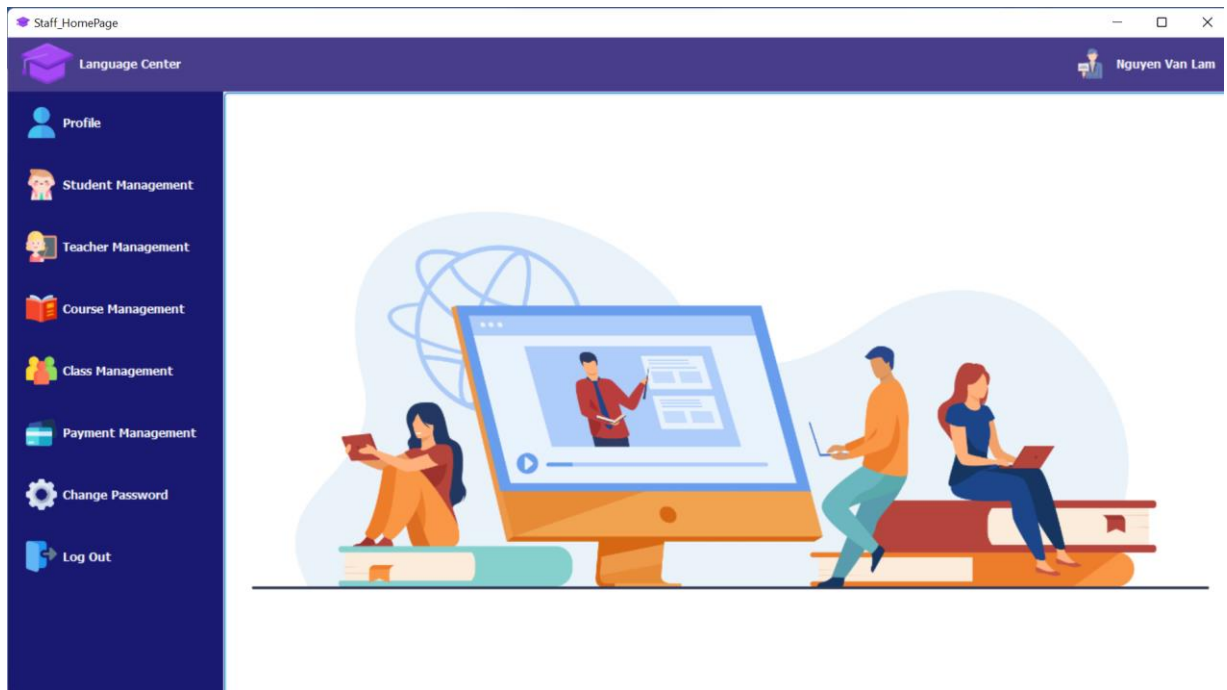
Username:

Password:

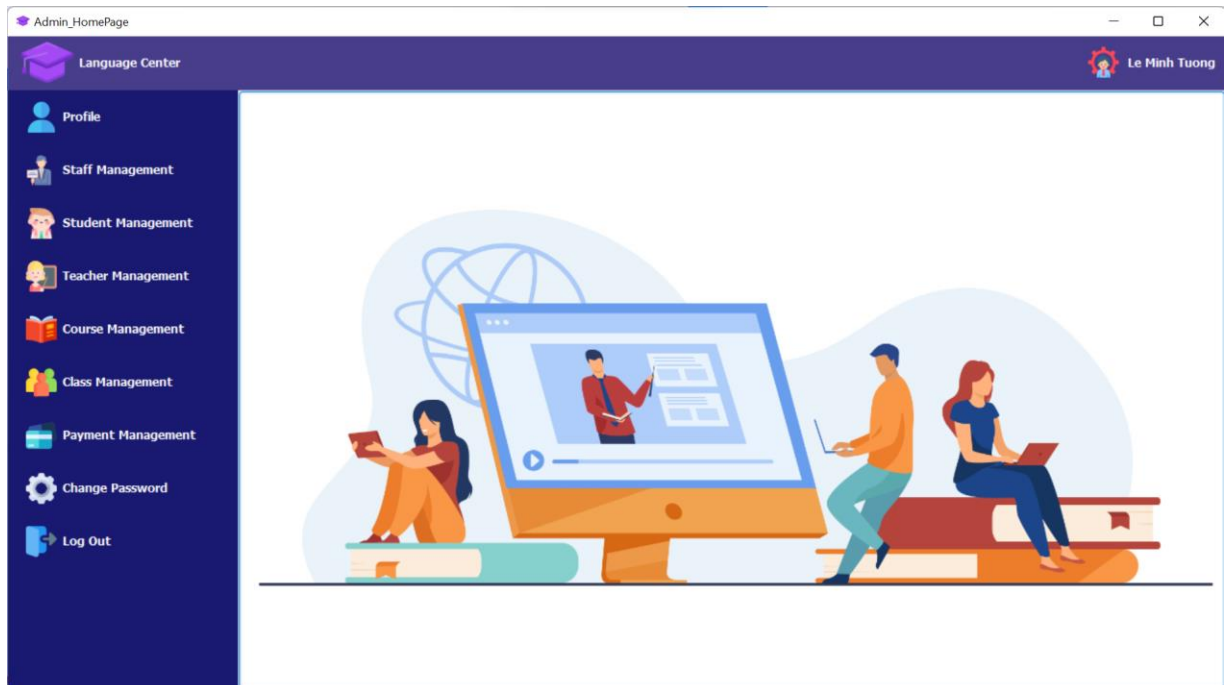
Log In

5.2 Form HomePage

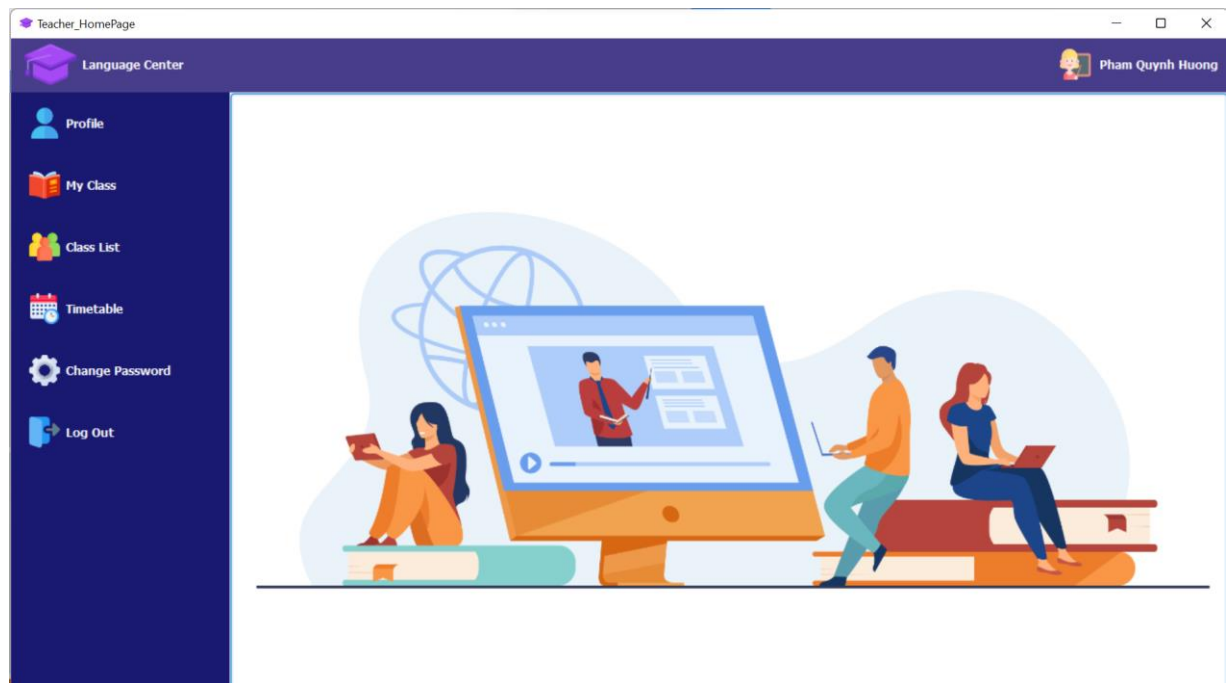
Form giao diện nhân viên:



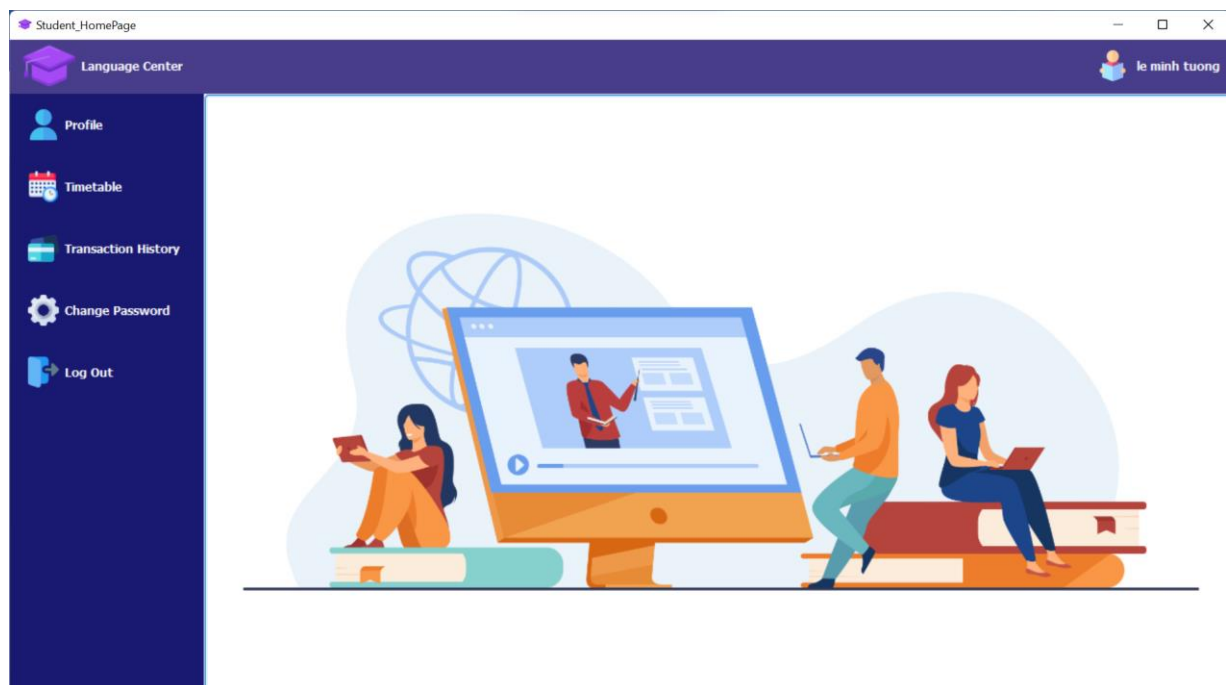
Form giao diện admin:




Form giao diện giảng viên:





Form giao diện học viên:



Đổi mật khẩu:

 Change password

 LANGUAGE CENTER



Username:

New Password:

Repeat Password:

Save

5.3 Form manage

Quản lí nhân viên:

Admin_HomePage

Language Center

Le Minh Tuong

Profile

Staff Management

Student Management

Teacher Management

Course Management

Class Management

Payment Management

Change Password

Log Out

Staff Management

Search By:

Username	StaffName	DateOfBirth	Address	Email	Phone	Position	Salary
admin1	Le Minh Tuong	09/12/2002	Hoang Dieu 2	admin1@gmail.com	0834091899	Admin	12000000
sa	Le Minh Tuong	25/10/2012	Tien Giang	lmt2002@gmail.com	0834091202	Admin	12000000
staff1	Le Minh Tuong	09/12/2002	Hoang Dieu 2	staff1@gmail.com	0834091990	HR	10000000
staff2	Pham Quynh Huong	31/01/2002	Dang Van Bi	staff2@gmail.com	0941156762	Marketing	9000000
staff3	Le Quang Tung	01/01/2002	Quan 9	staff3@gmail.com	0877600192	Trainer	7000000
staff4	Nguyen Van Lam	01/01/2002	To Ngoc Van	staff4@gmail.com	0762541389	Sales	7000000
staff5	Le Minh Tuong	09/12/2002	Hoang Dieu 2	staff5@gmail.com	0872616381	Receptionist	4000000

Username: sa Address: Tien Giang Add Refresh

Name: Le Minh Tuong Email: lmt2002@gmail.com Update Delete

DateOfBirth: 25-Oct-12 Phone: 0834091202

Position: Admin Salary: 12000000

Quản lí học viên:

Admin_HomePage

Language Center

Le Minh Tuong

Profile

Staff Management

Student Management

Teacher Management

Course Management

Class Management

Payment Management

Change Password

Log Out

Student Management

Search By:

Username	StudentName	DateOfBirth	Address	Email	Phone	Classname
student1	le minh tuong	09/12/2002	hoang dieu 2	lmt@gmail.com	0941144562	Lop 01
student3	le quang tung	01/01/2002	quan 9	lqt@gmail.com	0983625172	Lop 01
student4	pham quynh huong	31/01/2002	dang van bi	pqh@gmail.com	0143892012	Lop 01
student1	le minh tuong	09/12/2002	hoang dieu 2	lmt@gmail.com	0941144562	Lop 02
student2	nguyen van lam	07/07/2002	to ngoc van	nvi@gmail.com	0998765403	Lop 02
student1	le minh tuong	09/12/2002	hoang dieu 2	lmt@gmail.com	0941144562	Lop 03
student3	le quang tung	01/01/2002	quan 9	lqt@gmail.com	0983625172	Lop 03
student4	pham quynh huong	31/01/2002	dang van bi	pqh@gmail.com	0143892012	Lop 03
student4	pham quynh huong	31/01/2002	dang van bi	pqh@gmail.com	0143892012	Lop 04
student2	nguyen van lam	07/07/2002	to ngoc van	nvi@gmail.com	0998765403	Lop 05
student3	le quang tung	01/01/2002	quan 9	lqt@gmail.com	0983625172	Lop 06
student4	pham quynh huong	31/01/2002	dang van bi	pqh@gmail.com	0143892012	Lop 06

Username: Address: Add Refresh

Name: Email: Update Delete

DateOfBirth: 02-Dec-22 Phone:

Class Name:

Quản lý giảng viên:

QUẢN LÝ GIẢNG VIÊN

Admin_HomePage

Language Center

Le Minh Tuong

Quản lý giảng viên

Tìm theo:

Username	TeacherName	DateOfBirth	Address	Email	Phone	Salary
teacher1	Pham Quynh Huong	31/01/2002	Vung Tau	huong@gmail.com	0912334898	1000000
teacher2	Nguyen Van Lam	01/01/2002	Quy Nhon	lam@gmail.com	0912334777	500000
teacher3	Le Quang Tung	01/01/2002	Binh Dinh	tung@gmail.com	0912334090	1000000
teacher4	Le Minh Tuong	09/12/2002	Tien Giang	tuong@gmail.com	0912334121	500000

Username: teacher4 Địa chỉ: Tien Giang
Họ tên: Le Minh Tuong Email: tuong@gmail.com
Ngày sinh: 09-Dec-02 Số điện thoại: 0912334121
Salary: 500000

Thêm Làm mới
Sửa Xóa

Quản lý khóa học:

Admin_HomePage

Language Center

Le Minh Tuong

Course Management

Search By:

ID	CourseName	Target	NoLessons	Price
1	TOEIC LR	650	30	3000000
2	TOEIC LR	900	60	7000000
3	TOEIC SW	250	40	3000000
4	TOEIC SW	400	70	7000000
5	IELTS	6.5	70	10000000
6	IELTS	9	100	12000000

Course ID: 1 NoLessons: 30
Course Name: TOEIC LR Price: 3000000
Target: 650

Add Refresh
Update Delete

Quản lí lớp học:

Admin_HomePage

Language Center

Le Minh Tuong

Profile

Staff Management

Student Management

Teacher Management

Course Management

Class Management

Payment Management

Change Password

Log Out

Class Management

Search By:

ID	ClassName	StartDate	EndDate	WeekDays	StartTime	EndTime	ClassRoom	NoStudents	CourseName	Target	TeacherName
1	Lop 01	24/10/2022	30/11/2022	2-4-6	17:00:00	19:00:00	P01	3	TOEIC LR	650	Pham Quynh Huong
2	Lop 02	24/10/2022	30/11/2022	3-5-7	17:00:00	19:00:00	P02	2	TOEIC LR	900	Pham Quynh Huong
3	Lop 03	24/10/2022	30/11/2022	2-4-6	17:00:00	19:00:00	P03	3	TOEIC SW	250	Le Quang Tung
4	Lop 04	24/10/2022	30/11/2022	3-5-7	17:00:00	19:00:00	P04	1	TOEIC SW	400	Le Quang Tung
5	Lop 05	24/10/2022	30/11/2022	2-4-6	17:00:00	19:00:00	P05	1	IELTS	9	Nguyen Van Lam
6	Lop 06	24/10/2022	30/11/2022	3-5-7	17:00:00	19:00:00	P06	2	IELTS	6.5	Le Minh Tuong

ID: 1

Classname: Lop 01

StartDate: 24-Oct-22

EndDate: 30-Nov-22

Weekdays: 2-4-6

Classroom: P01

NoStudents: 3

StartTime: 6:00:00 PM

EndTime: 7:00:00 PM

CourseName: TOEIC LR

Target: 650

TeacherName: Pham Quynh Huong

Add

Refresh

Update

Delete

Quản lí thanh toán:

Admin_HomePage

Language Center

Le Minh Tuong

Profile

Staff Management

Student Management

Teacher Management

Course Management

Class Management

Payment Management

Change Password

Log Out

Payment Management

Search By:

ID	Username	StudentName	Email	Phone	PaymentDate	Amount	PaymentMethod	PaymentStatus
1	student1	le minh tuong	lmt@gmail.com	0941144562	09-Sep-22	8000000	Mobile Banking	Unpaid
2	student1	le minh tuong	lmt@gmail.com	0941144562	11-Sep-22	5000000	Mobile Banking	Paid
3	student2	nguyen van lam	nvl@gmail.com	0998765403	09-Sep-22	7000000	Mobile Banking	Unpaid
4	student2	nguyen van lam	nvl@gmail.com	0998765403	10-Sep-22	10000000	Mobile Banking	Unpaid
5	student2	nguyen van lam	nvl@gmail.com	0998765403	11-Sep-22	2000000	Mobile Banking	Paid
6	student3	le quang tung	lqt@gmail.com	0983625172	12-Sep-22	16000000	Mobile Banking	Paid
7	student4	pham quynh huong	pqh@gmail.com	0143892012	11-Sep-22	10000000	Mobile Banking	Unpaid
8	student4	pham quynh huong	pqh@gmail.com	0143892012	12-Sep-22	13000000	Mobile Banking	Paid

ID: 1

Username: student1

Payment Date: 09-Sep-22

Amount: 8000000

Method: Mobile Banking

Add

Refresh

Update

Delete

5.4 Form views

Xem thông tin cá nhân của từng vai trò

STAFF INFO

Name: Le Minh Tuong
Username: admin1
DateOfBirth: 09/12/2002
Address: Hoang Dieu 2
Email: admin1@gmail.com
Phone: 0834091899
Position: Admin

STAFF INFO

Name: Le Quang Tung
Username: staff3
DateOfBirth: 01/01/2002
Address: Quan 9
Email: staff3@gmail.com
Phone: 0877600192
Position: Trainer

TEACHER INFO

Name: Pham Quynh Huong
DateOfBirth: 31/01/2002
Address: Vung Tau
Email: huong@gmail.com
Phone: 0912334898

STUDENT INFO

Name: nguyen van lam
Username: student2
DateOfBirth: 07/07/2002
Address: to ngoc van
Email: nvl@gmail.com
Phone: 0998765403

Xem lớp học đang dạy của giảng viên đó

The screenshot shows a web application interface for a teacher. The top header is purple with 'Language Center' on the left and a user profile 'Pham Quynh Huong' on the right. A dark blue sidebar on the left contains navigation links: Profile, My Class, Class List, Timetable, Change Password, and Log Out. The main content area is titled 'MY CLASS' and contains a table with the following data:

ID	ClassName	StartDate	EndDate	WeekDays	StartTime	EndTime	ClassRoom	NoStudents	CourseName	Target	TeacherName
1	Lop 01	24/10/2022	30/11/2022	2-4-6	17:00:00	19:00:00	P01	3	TOEIC LR	650	Pham Quynh Huong
2	Lop 02	24/10/2022	30/11/2022	3-5-7	17:00:00	19:00:00	P02	2	TOEIC LR	900	Pham Quynh Huong

Xem danh sách tất cả lớp học với vai trò giảng viên

Teacher_HomePage

Language Center

Pham Quynh Huong

Profile

My Class

Class List

Timetable

Change Password

Log Out

ALL CLASSES

Search By:

ID	ClassName	StartDate	EndDate	WeekDays	StartTime	EndTime	ClassRoom	NoStudents	CourseName	Target	TeacherName
1	Lop 01	24/10/2022	30/11/2022	2-4-6	17:00:00	19:00:00	P01	3	TOEIC LR	650	Pham Quynh Huong
2	Lop 02	24/10/2022	30/11/2022	3-5-7	17:00:00	19:00:00	P02	2	TOEIC LR	900	Pham Quynh Huong
3	Lop 03	24/10/2022	30/11/2022	2-4-6	17:00:00	19:00:00	P03	3	TOEIC SW	250	Le Quang Tung
4	Lop 04	24/10/2022	30/11/2022	3-5-7	17:00:00	19:00:00	P04	1	TOEIC SW	400	Le Quang Tung
5	Lop 05	24/10/2022	30/11/2022	2-4-6	17:00:00	19:00:00	P05	1	IELTS	9	Nguyen Van Lam
6	Lop 06	24/10/2022	30/11/2022	3-5-7	17:00:00	19:00:00	P06	2	IELTS	6.5	Le Minh Tuong

Xem thời khóa biểu dạy của giảng viên đó

Teacher_HomePage

Language Center

Pham Quynh Huong

Profile

My Class

Class List

Timetable

Change Password

Log Out

TIME TABLE

Username	ClassID	ClassName	CourseID	CourseName	Date	Time	ClassRoom
teacher1	1	Lop 01	1	TOEIC LR	2-4-6	17:00 : 19:00	P01
teacher1	2	Lop 02	2	TOEIC LR	3-5-7	17:00 : 19:00	P02

Xem thời khóa biểu của học viên:

The screenshot shows a web application window titled "Student_HomePage". The interface has a dark blue sidebar on the left with a "Language Center" header and a user profile "le minh tuong". The sidebar contains links for Profile, Timetable, Transaction History, Change Password, and Log Out. The main content area displays a "TIME TABLE" table with the following data:

Username	ClassID	ClassName	CourseName	TeacherName	WeekDays	Time	ClassRoom
student1	1	Lop 01	TOEIC LR	Pham Quynh Huong	2-4-6	17:00 : 19:00	P01
student1	2	Lop 02	TOEIC LR	Pham Quynh Huong	3-5-7	17:00 : 19:00	P02
student1	3	Lop 03	TOEIC SW	Le Quang Tung	2-4-6	17:00 : 19:00	P03

Xem lịch sử giao dịch của học viên

The screenshot shows the same web application window, but the "Transaction History" link in the sidebar is selected. The main content area displays a "TRANSACTION HISTORY" table with the following data:

Username	PaymentDate	Amount	PaymentMethod	PaymentStatus
student1	09-Sep-22	8000000	Mobile Banking	<input type="checkbox"/>
student1	11-Sep-22	5000000	Cash	<input checked="" type="checkbox"/>
				<input type="checkbox"/>