

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC & KỸ THUẬT MÁY TÍNH



BÁO CÁO ASSIGNMENT 1

ĐỀ TÀI 1: TEACHING & LEARNING

GVHD: ThS. Đỗ Thanh Thái

SV thực hiện:	Tên	MSSV
	Đinh Tiến Khởi	2013537
	Trịnh Quang Minh	1914184

Tp. Hồ Chí Minh, Tháng 10/2022

MỤC LỤC

1	Overview	3
2	Requirements Analysis	3
2.1	Requirement	3
2.2	Constraint	3
3	Conceptual Design	4
3.1	Entity Relationship Diagram	4
3.2	Mô Tả Entity	4
3.3	Mô tả Relationship	5
3.4	Mô Tả Thuộc Tính Của Thực Thể	6
4	Logic Design	7
4.1	Relational Schema	7
4.2	Sử dụng công cụ để thiết kế Relational schema	7
4.2.1	Biểu Đồ Luận Lý	8
5	Normalization	9
5.1	Dạng chuẩn 1NF	9
5.2	Dạng chuẩn 2NF	9
5.3	Dạng chuẩn 3NF	9
5.4	Dạng chuẩn BCNF (Boyce Codd Normal Form)	10
6	Physical Design	10
6.1	Introduce Database Management System (DBMS)	10
6.1.1	Giới thiệu tổng quan về MySQL	10
6.1.2	Các ưu điểm của MySQL	10
6.1.3	Các nhược điểm của MySQL	11
6.1.4	MySQL trong bài tập lớn	11
6.2	Thiết lập bảng bằng MySQL	11
6.3	Table Relationships	15
6.4	Thêm dữ liệu - Insert Data	16
6.4.1	Tập lệnh	16
6.4.2	Kết Quả Thêm Dữ Liệu	16
7	Phân Quyền Và Kiểm Soát Truy Cập	17
7.1	Data Definition Language-DDL	17
7.1.1	Data Query Language-DQL	17
7.2	Data Manipulation Language-DML	17
7.3	Data Control Language-DCL	18
7.4	Định Nghĩa Trong Database	18
8	Trigger, Store Procedure	19
8.1	Trigger	19



8.2	Store Procedure	19
9	Indexing	19
10	Phát Triển Ứng Dụng	20
Tài Liệu Tham Khảo		23

listings

1 Overview

Bài tập lớn này sẽ thảo luận về việc thiết kế một hệ thống cơ sở dữ liệu quản lý các lớp học trong hệ thống giáo dục theo quy chế tín chỉ.

2 Requirements Analysis

2.1 Requirement

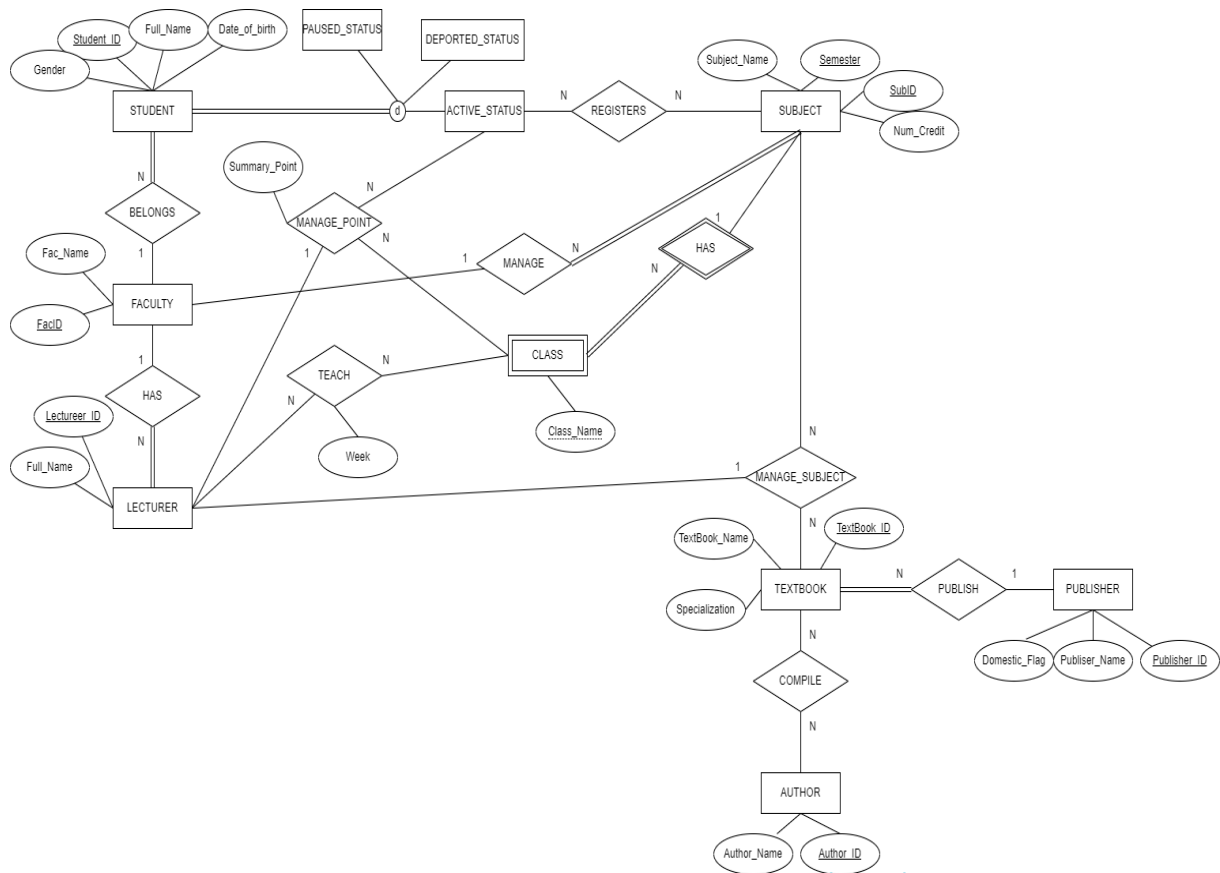
- Sinh viên (STUDENT) có trạng thái học tập bình thường (ACTIVE STATUS) có thể đăng ký một hoặc nhiều môn học, những sinh viên có trạng thái tạm dừng (PAUSED STATUS) hoặc buộc thôi học (DEPORTED STATUS) sẽ không thể đăng ký bất kỳ môn học nào ở học kỳ đó.
- Một môn học (SUBJECT) được mở và triển khai gồm một hoặc nhiều lớp tùy vào số lượng sinh viên đăng ký ở mỗi học kỳ.
- Mỗi lớp (CLASS) do một hoặc nhiều giảng viên (LECTURER) phụ trách ở các tuần học khác nhau, trong đó có một giảng viên phụ trách chính sẽ ghi điểm cho các sinh viên vào cuối học kỳ.
- Một hoặc nhiều giáo trình (TEXTBOOK) sẽ được sử dụng cho mỗi môn học.
- Mỗi giáo trình có một nhà xuất bản (PUBLISHER), có thể là nhà xuất bản trong nước hoặc ngoài nước, nhiều loại giáo trình có thể được mua chung từ một nhà xuất bản. Nhiều giáo trình thuộc về cùng một lĩnh vực (Specialization) có thể được mua từ các nhà xuất bản khác nhau.
- Mỗi giáo trình được biên soạn bởi một hay nhiều tác giả (AUTHOR) và một tác giả cũng có thể biên soạn một hoặc nhiều giáo trình.
- Mỗi giảng viên hay mỗi sinh viên chỉ thuộc về một khoa nhất định (FACULTY).
- Mỗi khoa quản lý danh sách các môn học thuộc chương trình đào tạo của khoa đó và giảng viên quản lý các môn học và các giáo trình cho môn học đó.

2.2 Constraint

- Mỗi môn học tối đa 3 tín chỉ.
- Mỗi sinh viên chỉ được đăng ký tối đa 18 tín chỉ cho một học kỳ.
- Mỗi lớp học có tối đa 60 sinh viên đăng ký.
- Mỗi môn học có tối đa 3 giáo trình chính do giảng viên phụ trách chính chỉ định.
- Mỗi giáo trình chính có thời gian kể từ lúc xuất bản không được quá 10 năm.

3 Conceptual Design

3.1 Entity Relationship Diagram



Hình 1: Entity-Relationship Diagram (ERD)

3.2 Mô Tả Entity

Kiểu thực thể	Mô tả
STUDENT	Lưu trữ thông tin của tất cả sinh viên, được chia làm 3 loại dựa theo trạng thái học tập của sinh viên đó: bình thường (Active), ngừng học (Paused) và buộc thôi học (Deported).
LECTURER	Lưu trữ thông tin của tất cả giảng viên trong trường.
FACULTY	Lưu trữ thông tin các khoa của trường. Các khoa có vai trò quản lý các sinh viên và giảng viên thuộc về khoa đó.
SUBJECT	Lưu trữ thông tin tất cả các môn học thuộc chương trình đào tạo của các khoa.
TEXTBOOK	Lưu trữ thông tin tất cả giáo trình mà các môn học sử dụng.
PUBLISHER	Lưu trữ thông tin các nhà xuất bản của các giáo trình.
AUTHOR	Lưu trữ thông tin của các tác giả đã biên soạn giáo trình

Class - Weak Entity	Lưu trữ thông tin các lớp được mở trong một học kỳ của tất cả các môn học.
---------------------	--

3.3 Mô tả Relationship

Relationship	Mô tả
BELONGS	Biểu Hiện cho mối quan hệ giữa: Khoa và Sinh Viên (FACULTY - STUDENT). Thông qua mối liên hệ này có thể biết sinh viên thuộc khoa nào và một khoa có những sinh viên nào.
HAS	Biểu thị mối quan hệ giữa: Khoa và Giảng Viên (FACULTY - LECTURER); Môn Học và Lớp Học (SUBJECT - CLASS) + FACULTY - LECTURER: Thông qua mối liên hệ có thể biết giảng viên thuộc khoa nào và khoa có những giảng viên nào. + SUBJECT - CLASS: Thông qua mối liên hệ có thể biết môn học mở những lớp nào và một lớp thuộc về môn học nào.
REGISTERS	Biểu thị mối quan hệ giữa: Sinh Viên ở trạng thái bình thường và Môn Học (ACTIVE_STATUS - SUBJECT). Thông qua mối liên hệ có thể biết sinh viên đăng ký những môn nào và danh sách sinh viên đăng ký một môn học nhất định.
MANAGE	Biểu hiện mối quan hệ giữa: Khoa và Môn học (FACULTY - SUBJECT). Thông qua mối quan hệ có thể biết được một khoa sẽ mở những môn học nào và một môn học thuộc về khoa nào quản lý.
TEACH	Biểu thị cho mối quan hệ giữa Giảng Viên và Lớp học (LECTURER - CLASS). Thông qua mối liên hệ có thể xác định giảng viên dạy những lớp nào và một lớp được phân định cho giảng viên nào.
MANAGE_POINT	Biểu thị mối quan hệ giữa Giảng Viên, Sinh Viên và Lớp học. Thông qua mối liên hệ, giảng viên có thể ghi điểm cho sinh viên ở một lớp nhất định, sinh viên có thể xem điểm của mình. Quan hệ sở hữu thuộc tính điểm tổng kết cho sinh viên.
COMPILE	Biểu thị mối quan hệ giữa Tác Giả và Giáo Trình (AUTHOR - TEXTBOOK). Thông qua mối quan hệ có thể xác định tác giả của một giáo trình nhất định và danh sách giáo trình mà tác giả viết.
PUBLISH	Biểu thị mối quan hệ giữa Nhà Xuất Bản và Giáo Trình (PUBLISHER - TEXTBOOK). Thông qua mối quan hệ có thể xác định nhà xuất bản của một giáo trình nhất định và danh sách giáo trình mà doanh nghiệp in ấn.
MANAGE_SUBJECT	Biểu thị mối quan hệ giữa Môn Học, Giảng Viên và Giáo Trình (SUBJECT - LECTURER - TEXTBOOK). Thông qua mối quan hệ có thể xác định giảng viên sử dụng những giáo trình nào và cho môn học nào.

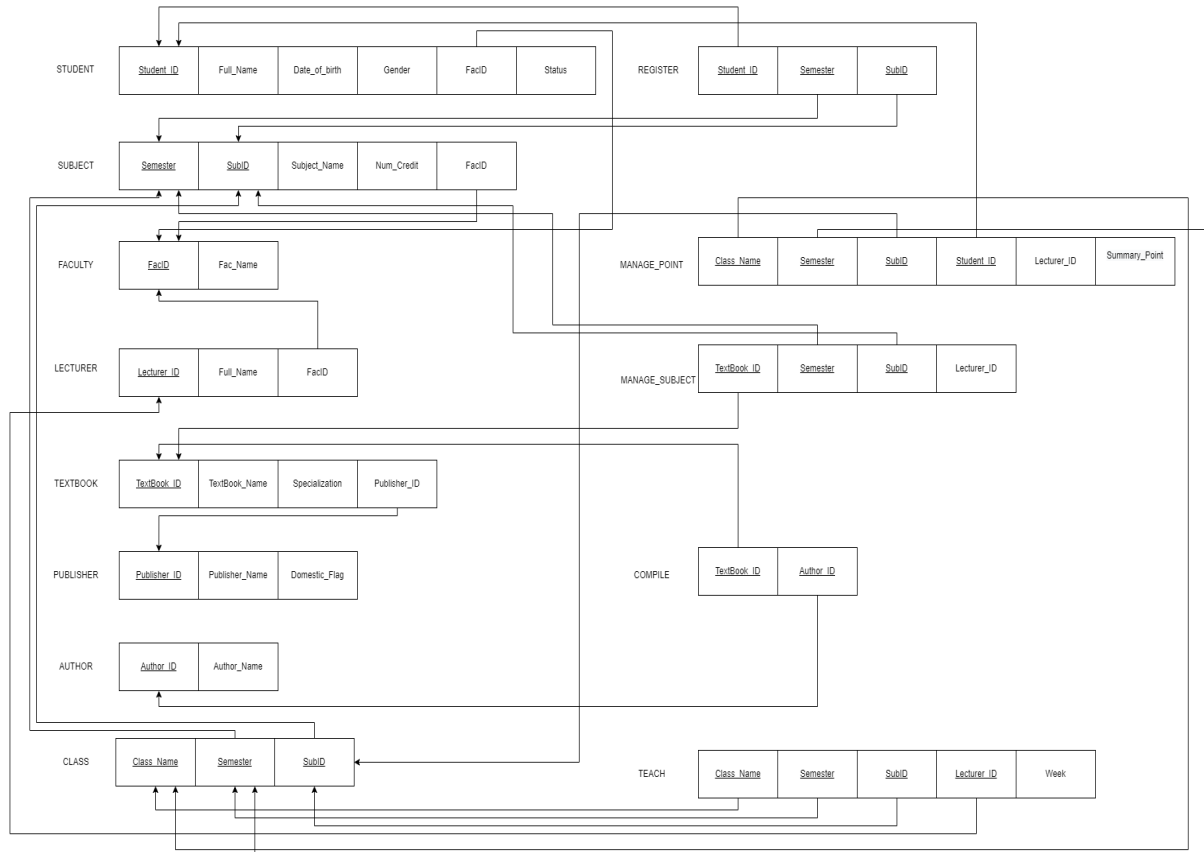
3.4 Mô Tả Thuộc Tính Của Thực Thể

Kiểu thực thể	Primary Key	Attributes
STUDENT	Student_ID	+ Student_ID: Mã số sinh Viên. + Full_Name: Họ Và Tên sinh viên. + Date_of_birth: Ngày tháng năm sinh của sinh viên. + Gender: Giới tính.
LECTURER	Lecturer_ID	+ Lecturer_ID: Mã số giảng viên. + Full_Name: Họ và tên giảng viên.
FACULTY	FacID	+ FacID: Mã số phòng khoa. + Fac_Name: Tên khoa.
SUBJECT	SubID, Semester	+ SubID: mã môn học. + Subject_Name: Tên môn học + Num_Credit: Số tín chỉ của môn học. + Semester: Học kỳ mở môn học.
TEXTBOOK	TextBook_ID	+ TextBook_ID: Mã số giáo trình + TextBook_Name: Tên giáo trình. + Specialization: Chuyên ngành cuốn sách hướng đến.
PUBLISHER	Publisher_ID	+ Publisher_ID: Mã số nhà xuất bản. + Publisher_Name: Tên nhà xuất bản. + Domestic_Flag: Xác định sản xuất nội địa.
AUTHOR	Author_ID	+ Author_ID: Mã tác giả. + Author_Name: Tên tác giả.
CLASS	Class_Name: Khóa riêng phần phụ thuộc vào thực thể mạnh SUBJECT.	+ Class_Name: Tên lớp thuộc môn học nhất định.

4 Logic Design

4.1 Relational Schema

Sau khi ánh xạ:

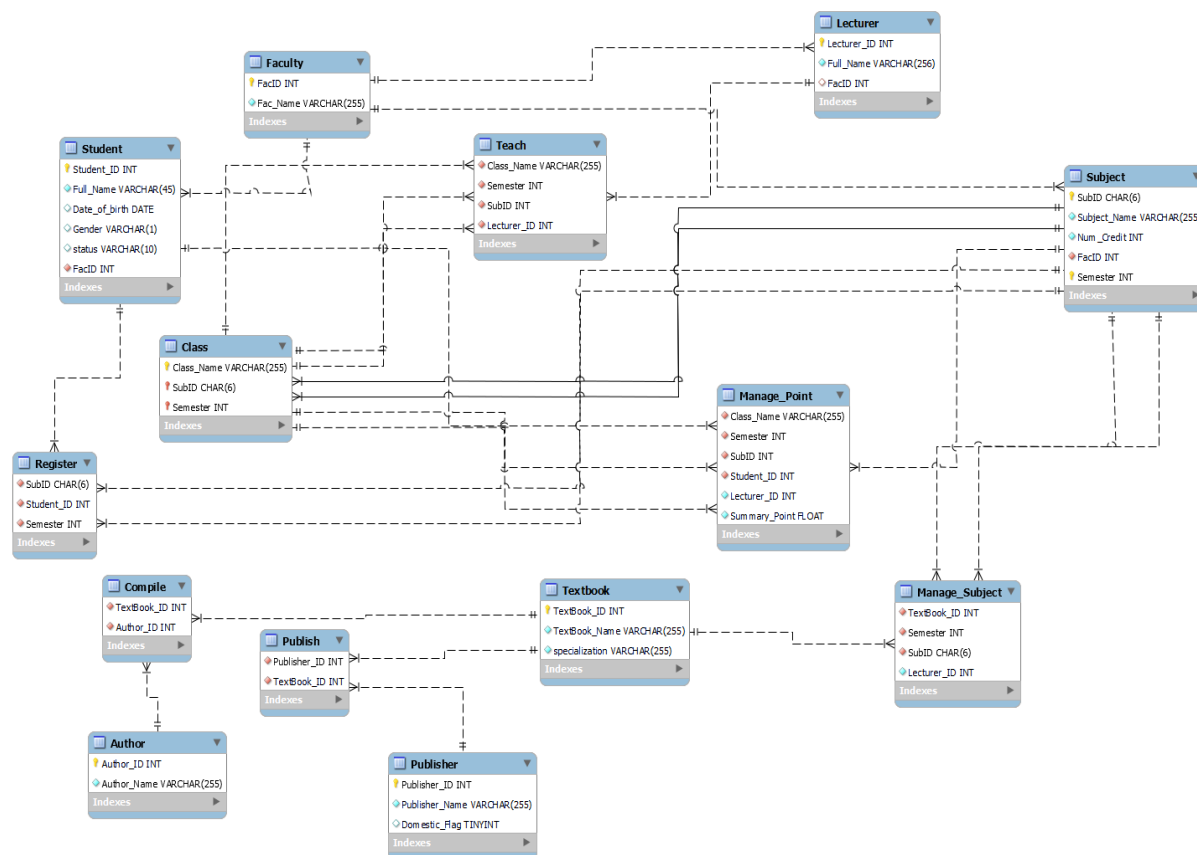


Hình 2: Relational Schema

4.2 Sử dụng công cụ để thiết kế Relational schema

Sau khi tìm hiểu các công cụ dùng để thiết kế sơ đồ quan hệ của các cơ sở dữ liệu như StartUML, Vertabelo, ERWin, RationalRose, Workbench... nhóm quyết định chọn công cụ Workbench để thiết kế.

4.2.1 Biểu Đồ Luận Lý



Hình 3: Lược Đồ Ảnh Xạ

5 Normalization

- Chuẩn hoá là quá trình tách bảng thành các bảng nhỏ hơn dựa vào các phụ thuộc hàm. Các dạng chuẩn là các chỉ dẫn để thiết kế các bảng trong CSDL.
- Mục đích của chuẩn hoá là loại bỏ các dư thừa dữ liệu và các lỗi khi thao tác dư thừa và các lỗi khi thao tác dữ liệu (Insert, Delete, Update). Nhưng chuẩn hoá làm tăng thời gian truy vấn.

5.1 Dạng chuẩn 1NF

- Một bảng (quan hệ) được gọi là ở dạng chuẩn 1NF nếu và chỉ nếu toàn bộ các miền giá trị của các cột có mặt trong bảng (quan hệ) đều chỉ chứa các giá trị nguyên tử (nguyên tố), nghĩa là mọi thuộc tính trong bảng quan hệ đó đều là thuộc tính đơn trị và không có bất kỳ thuộc tính nào là thuộc tính hỗn hợp.
- 1NF là yêu cầu tối thiểu của một mô hình dữ liệu quan hệ để nó có thể được triển khai trong ngôn ngữ truy vấn có cấu trúc (SQL). Trong mô hình Teaching-Learning đang thiết kế, không có vi phạm dạng chuẩn 1NF.

5.2 Dạng chuẩn 2NF

- Định nghĩa một quan hệ ở dạng chuẩn 2NF nếu quan hệ đó:
 - + Là 1NF
 - + Các thuộc tính không khoá phải phụ thuộc hàm đầy đủ vào khoá chính
- Một quan hệ ở dạng chuẩn 2NF nếu thỏa mãn 1 trong các điều kiện sau: Khoá chính chỉ gồm một thuộc tính. Bảng không có các thuộc tính không khoá. Tất cả các thuộc tính không khoá phụ thuộc hoàn toàn vào tập các thuộc tính khoá chính.
- Trong mô hình đang thiết kế, không có vi phạm chuẩn 2NF

5.3 Dạng chuẩn 3NF

- Một quan hệ ở dạng chuẩn 3NF nếu quan hệ đó:
 - + Là 2NF
 - + Các thuộc tính không khoá phải phụ thuộc trực tiếp vào khoá chính, nghĩa là không có phụ thuộc bắc cầu của các thuộc tính không phải nguyên tố.
- Một quan hệ ở dạng 3NF nếu có ít nhất một trong các điều kiện sau đây trong mọi phụ thuộc hàm không tầm thường $X \rightarrow Y$
 - + X là một siêu khóa.
 - + Y là thuộc tính nguyên tố (mỗi phần tử của Y là một phần của khóa ứng viên nào đó).
- Sau khi hoàn thiện, các mối quan hệ trong mô hình thiết kế đều đáp ứng các điều kiện của dạng chuẩn 3NF

5.4 Dạng chuẩn BCNF (Boyce Codd Normal Form)

- Một quan hệ ở dạng chuẩn BCNF nếu quan hệ đó:
 - + Là 3NF
 - + Không có thuộc tính khoá mà phụ thuộc hàm vào thuộc tính không khoá.
- Một quan hệ thỏa mãn BCNF nếu trong mọi phụ thuộc hàm không tầm thường $X \rightarrow Y$, X là siêu khoá.
- Nhìn chung, mô hình mà nhóm thiết kế đã đáp ứng đầy đủ 4 dạng chuẩn hóa về yêu cầu dữ liệu.

6 Physical Design

6.1 Introduce Database Management System (DBMS)

- Database Management System(DBMS) là một phần mềm hướng tới các tác vụ khởi tạo và quản lý các databases. Do vậy, DBMS cung cấp các tác vụ lưu trữ hoặc truy hồi thông tin từ database một cách hiệu quả và thuận tiện.
- Trong phạm vi môn học, nhóm em quyết định sử dụng công cụ MySQL để tạo cơ sở dữ liệu.

6.1.1 Giới thiệu tổng quan về MySQL

- MySQL là hệ quản trị cơ sở dữ liệu mã nguồn mở phổ biến hàng đầu trên thế giới và đặc biệt được ưa chuộng trong quá trình xây dựng, phát triển ứng dụng. Đây là hệ quản trị cơ sở dữ liệu tốc độ cao, ổn định và dễ sử dụng, được tin dùng từ các doanh nghiệp nhỏ đến những doanh nghiệp toàn cầu.
- MySQL được phát triển, thương mại hóa và hỗ trợ bởi MySQL AB, một công ty tại Thụy Điển.

6.1.2 Các ưu điểm của MySQL

- MySQL được phát hành trên danh nghĩa là mã nguồn mở, vậy nên có thể coi nó là miễn phí.
- MySQL rất mạnh mẽ trong việc xử lý một tập lớn các phương thức liên quan đến các databases packages.
- MySQL tương thích với hầu hết các hệ điều hành phổ biến cùng với đó là đa dạng ngôn ngữ lập trình như PHP, PERL, C, C++, JAVA, etc.
- MySQL hỗ trợ khả năng lưu trữ khá tốt, lên tới 50 triệu dòng trên mỗi bảng và có thể hơn. Giới hạn mặc định về kích thước file của một bảng lên đến 4GB nhưng nếu hệ điều hành cho phép sửa đổi, con số đó có thể lên đến 8 triệu TB, về mặt lý thuyết. Đồng thời, tốc độ xử lý nhanh và đảm bảo tính chính xác cũng là một điểm mạnh của MySQL.
- MySQL có thể được cấu hình riêng biệt bởi các lập trình viên sao cho phù hợp với mục đích nhất định

6.1.3 Các nhược điểm của MySQL

- MySQL bị hạn chế dung lượng, cụ thể, khi số bản ghi của người dùng lớn dần, sẽ gây khó khăn cho việc truy xuất dữ liệu, khiến người dùng cần áp dụng nhiều biện pháp để tăng tốc độ chia sẻ dữ liệu như chia tải database ra nhiều server, hoặc tạo cache MySQL.
- So với Microsoft SQL Server hay Oracle, độ bảo mật của MySQL chưa cao bằng. Và quá trình Restore cũng có phần chậm hơn

6.1.4 MySQL trong bài tập lớn

- MySQL là sự lựa chọn thông dụng, tích hợp đầy đủ các tiện ích, dễ sử dụng. Công cụ này dễ sử dụng, hoạt động được ở nhiều hệ điều hành. MySQL được sử dụng với mục đích nhằm hỗ trợ NodeJs, PHP, Perl, và nhiều ngôn ngữ khác. Và cuối cùng, công cụ này có phiên bản được sử dụng hoàn toàn miễn phí.
- MySQL có các ưu điểm và nhược điểm đã nêu trên. Khi xét về phạm vi của môn học thì MySQL là hệ cơ sở dữ liệu hoàn toàn với những yêu cầu đã đề ra. Chính vì thế, Nhóm em quyết định sử dụng MySQL để xây dựng hệ cơ sở dữ liệu.

6.2 Thiết lập bảng bằng MySQL

- Faculty:

```
1  create database university;
   use university;

   create table if not exists faculty(
       FacID int not null,
6      Fac_Name varchar(255) not null,
       primary key(FacID)
   );
```

- Lecturer:

```
   create table if not exists lecturer(
2      Lecturer_ID int not null,
       Full_Name varchar(255) not null,
       FacID int not null,
       primary key(Lecturer_ID),
       foreign key(FacID)
7      references faculty (FacID)
       on delete restrict
       on update cascade
   );
```

- Student:

```
   create table if not exists student(
       Student_ID int not null,
       Full_Name varchar(255) not null,
       Date_of_birth date,
5      Gender varchar(1) default null,
       current_status varchar(10) default null,
       FacID int not null,
```

```
10      primary key(Student_ID),  
        foreign key(FacID)  
        references faculty(FacID)  
        on delete restrict  
        on update cascade  
    );
```

- Subject:

```
2      create table if not exists subject(  
        SubID int not null,  
        Subject_Name varchar(255) not null,  
        Num_Credit int not null check(Num_Credit > 0 and Num_Credit <= 4),  
        Semester int not null,  
        FacID int not null,  
7      primary key(SubID, Semester),  
        foreign key(FacID)  
        references faculty(FacID)  
        on delete restrict  
        on update cascade  
12    );
```

- Class:

```
3      create table if not exists class(  
        Class_Name varchar(255) not null,  
        SubID int not null,  
        Semester int not null,  
        primary key(Class_Name, SubID, Semester),  
        foreign key(SubID, Semester)  
        references subject(SubID, Semester)  
8    );
```

- Publisher:

```
2      create table if not exists publisher(  
        Publisher_ID INT NOT NULL,  
        Publisher_Name VARCHAR(255) NOT NULL,  
        Domestic_Flag TINYINT NULL,  
        primary key(Publisher_ID)  
7    );
```

- TextBook:

```
4      create table if not exists textbook(  
        TextBook_ID INT NOT NULL,  
        TextBook_Name VARCHAR(255) NOT NULL,  
        specialization VARCHAR(255) NOT NULL,  
        PRIMARY KEY (TextBook_ID)  
    );
```

- Register:

```
3      create table if not exists register(  
      SubID CHAR(6) NOT NULL,  
      Student_ID INT NOT NULL,  
      Semester INT NOT NULL,  
      FOREIGN KEY (`SubID`)  
      REFERENCES Subject (SubID)  
      on delete restrict  
8      on update cascade,  
      FOREIGN KEY (`Student_ID`)  
      REFERENCES `University`.`Student` (`Student_ID`)  
      on delete restrict  
      on update cascade,  
13     FOREIGN KEY (`Semester`)  
      REFERENCES `University`.`Subject` (`Semester`)  
      on delete restrict  
      on update cascade  
18     );
```

- Manage_Point:

```
3      create table if not exists manage_point(  
      `Class_Name` VARCHAR(255) NOT NULL,  
      `Semester` INT NOT NULL,  
      `SubID` INT NOT NULL,  
      `Student_ID` INT NOT NULL,  
      `Lecturer_ID` INT NOT NULL,  
      `Summary_Point` FLOAT,  
8      FOREIGN KEY (`Class_Name`)  
      REFERENCES `University`.`Class` (`Class_Name`)  
      on delete restrict  
      on update cascade,  
      FOREIGN KEY (`Semester`)  
13     REFERENCES `University`.`Class` (`Semester`)  
      on delete restrict  
      on update cascade,  
      FOREIGN KEY (`SubID`)  
      REFERENCES `University`.`Subject` (`Semester`)  
18     on delete restrict  
      on update cascade,  
      FOREIGN KEY (`Student_ID`)  
      REFERENCES `University`.`Student` (`Student_ID`)  
      on delete restrict  
23     on update cascade  
      );
```

- Manage_Subject:

```
1      create table if not exists manage_subject(  
      `TextBook_ID` INT NOT NULL,  
      `Semester` INT NOT NULL,  
      `SubID` CHAR(6) NOT NULL,  
      `Lecturer_ID` INT NOT NULL,  
6      FOREIGN KEY (`TextBook_ID`)
```

```
REFERENCES `University`.`Textbook` (`TextBook_ID`)
on delete restrict
on update cascade,
FOREIGN KEY (`SubID`)
REFERENCES `University`.`Subject` (`SubID`)
on delete restrict
on update cascade,
FOREIGN KEY (`Semester`)
REFERENCES `University`.`Subject` (`Semester`)
on delete restrict
on update cascade
);
```

- Compile:

```
create table if not exists compile(
`TextBook_ID` INT NOT NULL,
`Author_ID` INT NOT NULL,
FOREIGN KEY (`TextBook_ID`)
REFERENCES `University`.`Textbook` (`TextBook_ID`)
on delete restrict
on update cascade,
FOREIGN KEY (`Author_ID`)
REFERENCES `University`.`Author` (`Author_ID`)
on delete restrict
on update cascade
);
```

- Publish:

```
create table if not exists publish(
`Publisher_ID` INT NOT NULL,
`TextBook_ID` INT NOT NULL,
FOREIGN KEY (`Publisher_ID`)
REFERENCES `University`.`Publisher` (`Publisher_ID`)
on delete restrict
on update cascade,
FOREIGN KEY (`TextBook_ID`)
REFERENCES `University`.`Textbook` (`TextBook_ID`)
on delete restrict
on update cascade
);
```

- Teach:

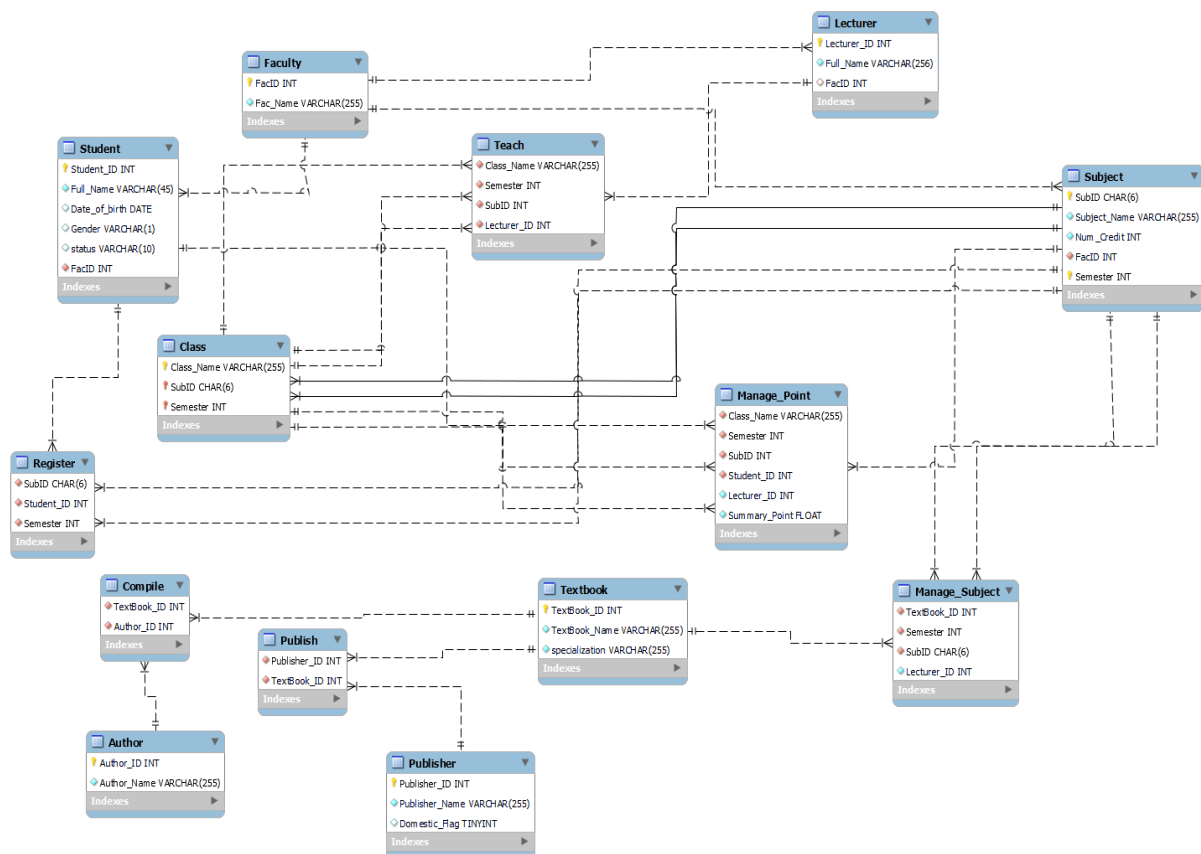
```
create table if not exists teach(
`Class_Name` VARCHAR(255) NOT NULL,
`Semester` INT NOT NULL,
`SubID` INT NOT NULL,
`Lecturer_ID` INT NOT NULL,
FOREIGN KEY (`Lecturer_ID`)
REFERENCES `University`.`Lecturer` (`Lecturer_ID`)
on delete restrict
```

```

13      on update cascade,
      FOREIGN KEY (`SubID`)
      REFERENCES `University`.`Class` (`Semester`)
      on delete restrict
      on update cascade,
      FOREIGN KEY (`Class_Name`)
      REFERENCES `University`.`Class` (`Class_Name`)
      on delete restrict
      on update cascade,
18      FOREIGN KEY (`Semester`)
      REFERENCES `University`.`Class` (`Semester`)
      on delete restrict
      on update cascade
23  );

```

6.3 Table Relationships



Hình 4: Mối quan hệ giữa các bảng

6.4 Thêm dữ liệu - Insert Data

6.4.1 Tập lệnh

```

/*Insert data into faculty*/
insert into faculty(FacID, Fac_Name)
values(106, 'Khoa hoc va ky thuat may tinh'),
(114, 'Ky thuat hoa hoc'),
(108, 'Ky thuat dien'),
(100, 'Ly luan chinh tri'),
(200, 'Toan Ung Dung'),
(201, 'Vat ly');

/*Insert data into student*/
insert into student(Student_ID,Full_Name,Date_of_birth,Gender,current_status,FacID)
values(2013537, 'Dinh Tien Khoi', '2002-09-30', 'M', 'active', 106),
(2014999, 'Nguyen Quang Hai', '2002-06-01', 'M', 'active', 114),
(2014347, 'Huynh Thanh Sang', '2002-10-16', 'M', 'active', 108);

/*Insert data into lecturer*/
insert into lecturer(lecturer_id, full_name, facid)
values(1000, "Tran Thanh Binh", 106),
(1001, "Vu Trong Thien", 106),
(1002, "Truong Quynh Chi", 106),
(1003, "Do Thanh Thai", 106),
(1004, "Dang Kieu Diem", 100),
(1005, "Cao Hong Quan", 100)

```

6.4.2 Kết Quả Thêm Dữ Liệu

	Student_ID	Full_Name	Date_of_birth	Gender	current_status	FacID
▶	2013537	Dinh Tien Khoi	2002-09-30	M	active	106
	2014999	Nguyen Quang Hai	2002-06-01	M	active	114
	2014347	Huynh Thanh Sang	2002-10-16	M	active	108
✱	NULL	NULL	NULL	NULL	NULL	NULL

Hình 5: Student table

	FacID	Fac_Name
▶	106	Khoa hoc va ky thuat may tinh
	114	Ky thuat hoa hoc
	108	Ky thuat dien
	200	Toan ung dung
	201	Vat ly
	100	Ly luan chinh tri
✱	NULL	NULL

Hình 6: Student table

7 Phân Quyền Và Kiểm Soát Truy Cập

Ngôn ngữ SQL là ngôn ngữ cơ sở dữ liệu có thể thực hiện các tác vụ nhất định được định nghĩa trên DBMS. Do vậy, để tăng tính bảo mật, mỗi DBMS cần sử dụng một trong những phương pháp hướng đến phân quyền cũng như quản lý dữ liệu được truy cập.

7.1 Data Definition Language-DDL

DDL hay Data Definition Language bao gồm các câu lệnh được sử dụng để định nghĩa lược đồ cơ sở dữ liệu. DDL xây dựng những định nghĩa cơ bản về cơ sở dữ liệu và sử dụng chúng để tương tác với các đối tượng trong cơ sở dữ liệu. Để đạt được mục đích ấy, DDL sở hữu một tập các câu lệnh liên quan như CREATE, ALTER và DROP databases - không tương tác với dữ liệu. Do vậy, cần phân quyền cho các user nhất định có thể thực hiện các câu lệnh trên - đặc biệt là DROP. Danh sách các câu lệnh:

- CREATE: Câu lệnh được sử dụng để tạo cơ sở dữ liệu hoặc các đối tượng trong nó như bảng - table, trigger, store procedure, view, etc.
- DROP: Câu lệnh được sử dụng để xóa một đối tượng trong cơ sở dữ liệu.
- ALTER: Câu lệnh được dùng để chỉnh sửa cấu trúc, đặc tính của database.
- TRUNCATE: Câu lệnh được sử dụng để xóa tất cả các hàng dữ liệu trong một bảng, bao gồm tất cả các vùng dữ liệu được cấp phát.
- COMMENT: Câu lệnh được sử dụng để thêm comment vào kho dữ liệu
- RENAME: Câu lệnh được sử dụng để thay đổi tên gọi của một đối tượng trong database.

7.1.1 Data Query Language-DQL

Câu lệnh DQL được sử dụng để hiện thực các câu truy vấn dữ liệu trong các đối tượng thuộc cơ sở dữ liệu. Mục đích chính của câu lệnh là truy vấn dữ liệu và gom chúng thành một relation phục vụ cho một mục đích nhất định. Để thực hiện được các yêu cầu trên, ta sử dụng câu lệnh SELECT. Sau khi câu truy vấn được thực thi, kết quả sẽ được biên dịch thành một bảng tạm thời và gửi nó đến nơi truy vấn.

7.2 Data Manipulation Language-DML

Các câu lệnh DML được sử dụng để thực hiện các thao tác với các dữ liệu đối với các bảng trong cơ sở dữ liệu bao gồm thêm, xóa, cập nhật và lock.

Danh sách các câu lệnh:

- INSERT: Câu lệnh được sử dụng để thêm dữ liệu vào bảng.
- UPDATE: Câu lệnh được sử dụng để cập nhật, chỉnh sửa dữ liệu đã tồn tại trong bảng.
- DELETE: Câu lệnh được sử dụng để xóa một data row tồn tại trong bảng.
- LOCK: Câu lệnh sử dụng khi tồn tại các mối truy cập song song cần được điều khiển thứ tự truy cập.

7.3 Data Control Language-DCL

DCL được các DBMS sử dụng một cách phổ biến như Mysql hay Sql Server để điều khiển các quyền truy cập vào cơ sở dữ liệu bằng cách tạo các user và cấp quyền cho chúng, cũng có thể là thu hồi các quyền đã cấp.

Danh Sách Các Câu lệnh:

- GRANT: Câu lệnh được sử dụng để cấp quyền truy cập, điều khiển cho một user nhất định.
- REVOKE: Câu lệnh được sử dụng để thu hồi lại quyền truy cập của user.

7.4 Định Nghĩa Trong Database

MySQL sử dụng cơ chế Discretionary Access Control (DAC) giúp cho người lập trình có thể cấp quyền cho các đối tượng nhất định nhằm bảo vệ cơ sở dữ liệu tránh khỏi các cuộc tấn công có chủ đích cũng như tránh rò rỉ thông tin ở mức tối đa.

Trong cơ sở dữ liệu, tồn tại 3 đối tượng cần được phân quyền và đó là 3 đối tượng chính sử dụng ứng dụng bao gồm Lecturer, Student và Faculty. Khi xét về mức độ quyền lực, có thể thấy rằng Faculty sẽ có quyền truy vấn, truy cập ở nhiều bảng nhất, sau đó đến Lecturer và cuối cùng là Student.

Table	Faculty	Lecturer	Student
Student	DQL	DQL	DQL
Lecturer	DQL	DQL	DQL
Faculty	DQL	DQL	DQL
Class	DQL, DML	DQL	DQL
Subject	DQL, DML	DQL	DQL
Textbook	DQL	DQL, DML	DQL
Register	DQL	DQL	DQL, DML
Author	DQL	DQL	DQL
Teach	DQL, DML	DQL	DQL
Publisher	DQL	DQL	DQL
Manage_point	DQL	DQL, DML	DQL
Manage_subject	DQL, DML	DQL, DML	DQL, DML
Compile	DQL	DQL	DQL

Cùng với sự phân quyền tối các user đối với các các lệnh trên, ta tiến hành phân quyền các store procedure, view .v.v tới các đối tượng phục vụ cho việc truy vấn.

```
CREATE USER 'student'@'localhost' IDENTIFIED BY 'student_password';
CREATE USER 'lecturer'@'localhost' IDENTIFIED BY 'lecturer_password';
CREATE USER 'faculty'@'localhost' IDENTIFIED BY 'faculty_password';
grant INSERT, UPDATE, DELETE on university.manage_point to 'student'@'localhost' WITH
GRANT OPTION;
grant select on university.* to 'student'@'localhost' WITH GRANT OPTION;
grant select on university.* to 'lecturer'@'localhost' WITH GRANT OPTION;
grant select on university.* to 'faculty'@'localhost' WITH GRANT OPTION;
grant execute on university.* to 'faculty'@'localhost';
grant execute on university.* to 'student'@'localhost';
grant execute on university.* to 'lecturer'@'localhost';
FLUSH PRIVILEGES;
```

8 Trigger, Store Procedure

8.1 Trigger

Trigger là tập các câu lệnh bên trong hệ thống. Nó là một kiểu store procedure đặc biệt bởi lẽ nó sẽ tự động thực thi dựa trên một sự kiện nhất định mà người dùng định nghĩa. Mỗi trigger được định nghĩa trên một bảng nhất định và được kích hoạt khi bằng thực thi các câu lệnh DML như INSERT, UPDATE, DELETE.

Một vài đặc điểm của trigger:

- Trigger có thể kiểm tra tính đúng đắn của dữ liệu trước khi được thêm vào bảng hoặc kiểm tra tính toàn vẹn của dữ liệu khi xóa hoặc cập nhật dữ liệu trên bảng.
- Trigger sử dụng tham số NEW và OLD trong chương trình tương ứng với dữ liệu được thêm vào hay dữ liệu đã được chỉnh sửa.

Trigger của chương trình được lưu trữ tại file **trigger.sql**

8.2 Store Procedure

Store Procedure là tập các câu lệnh được khai báo bằng câu lệnh CREATE PROCEDURE xây dựng với một mục đích nhất định. Khác với **Function**, **Store Procedure** không có tham số trả về nhưng có thể thực hiện truy vấn trả về một relation hoặc một thông báo nhất định.

Ở chương trình sử dụng rất nhiều Store Procedure với nhiều mục đích khác nhau như thêm, xóa, sửa mục đích để đơn giản hóa các câu truy vấn ở Repository đồng thời giảm băng thông truy cập vào database. Các Store Procedure và Transaction mà chương trình sử dụng được lưu trữ tại **Store_Procedure.sql**

```
CREATE DEFINER=`khoidinh309`@`localhost` PROCEDURE `get_unreleased_subject`(faculty_id
    int)
BEGIN
    select s.subID, s.subject_name, s.num_credit, s.semester
    from subject as s
    where s.facID = faculty_id and not exists (select * from class as c where c.subID
    = s.subID);
END
```

9 Indexing

Indexing là cấu trúc rất mạnh mẽ trong Mysql nhằm cải thiện tốc độ truy vấn của các câu lệnh truy vấn thường dùng. Để đạt được mục đích ấy, MySql tiến hành khởi tạo các bảng nhỏ hơn được gọi là index từ từng cột - Nondense Index hay một tập các cột Dense Index.

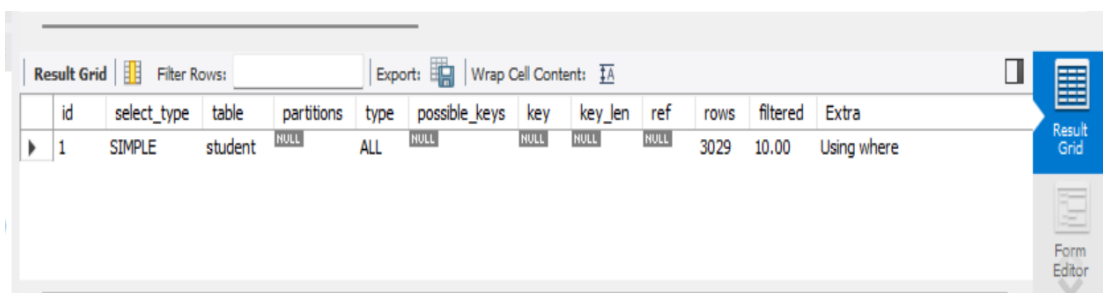
Áp dụng vào hệ thống, ta tiến hành tạo index cho full_name và ở các cột thường xuyên đưa ra câu lệnh truy vấn trong bảng:

```
create index student_index on university.student(full_name);
```

Để đánh giá được độ hiệu quả của việc thêm index cho bảng student, ta tiến hành thực hiện câu truy vấn sau:

```
explain select * from student where full_name="Nguyen Quang Hai";
```

- Trước khi thêm index:

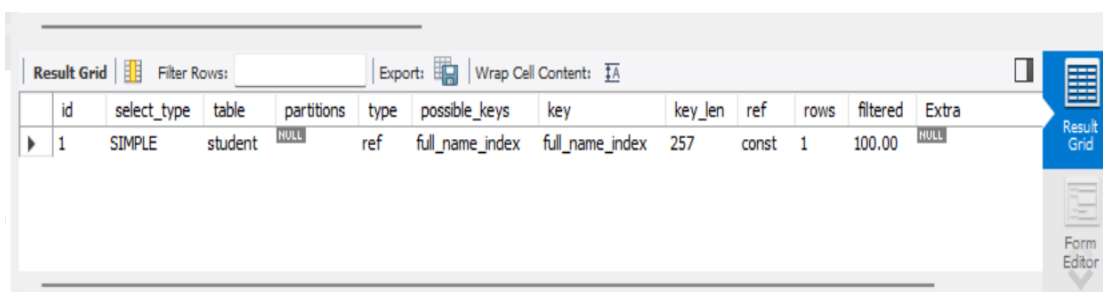


	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
▶	1	SIMPLE	student	HULL	ALL	HULL	HULL	HULL	HULL	3029	10.00	Using where

Hình 7: Before Indexing

*Nhận xét: Có thể thấy số lượng dòng mà câu lệnh dự đoán cần duyệt qua là 3029 dòng - tổng số dòng trong bảng.

- Sau khi thêm index:



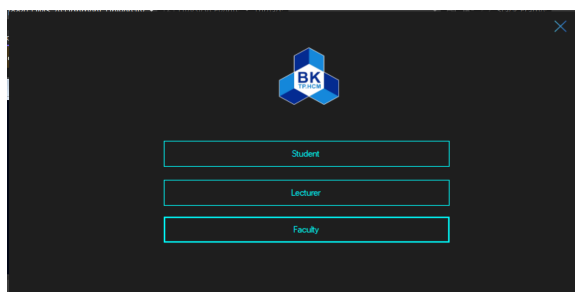
	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
▶	1	SIMPLE	student	HULL	ref	full_name_index	full_name_index	257	const	1	100.00	HULL

Hình 8: After Indexing

*Nhận xét: Số lượng dòng mà hệ thống dự đoán duyệt qua chỉ có 1 - có thể là do cái tên truy vấn là unique trong bảng. Nhưng nhìn chung quá trình truy vấn đã ngắn hơn và có thể thấy độ hiệu quả của việc thêm index một cách rõ ràng, đặc biệt ở các cơ sở dữ liệu

10 Phát Triển Ứng Dụng

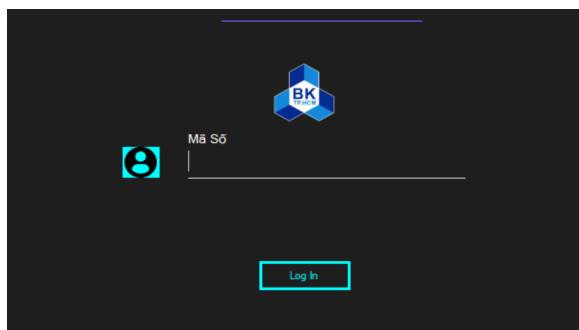
Dựa trên cấu trúc dữ liệu đã được xây dựng, nhóm em tiến hành phát triển ứng dụng dựa trên .Net framework với thư viện MySQLClient. Nhóm hiện thực app dựa trên mô hình MVP và tiến hành phân chia các thành phần trong app. Hệ thống được xây dựng dựa trên 3 đối tượng chính là Faculty, Lecturer và Student.



Hình 9: Log In Option

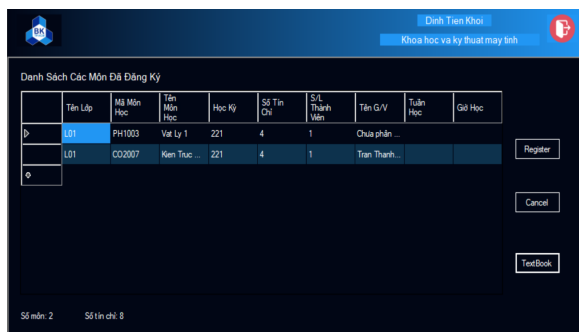
Sau khi chọn đối tượng sử dụng, người dùng tiến hành đăng nhập vào hệ thống bằng mã định danh được cấp.

- Tiến hành đăng nhập với tư cách là student, ta có giao diện chính như sau



Form đăng nhập (Login) của hệ thống. Giao diện tối màu với logo BK ở trên. Có một ô nhập "Mã Số" và một nút "Log In" màu xanh.

Hình 10: Log In Form



Giao diện chính của sinh viên. Ở trên có thanh tiêu đề với logo BK và thông tin người dùng "Đinh Tiến Khôi" và "Khoa học và kỹ thuật máy tính". Phần chính là bảng "Danh Sách Các Môn Đã Đăng Ký".

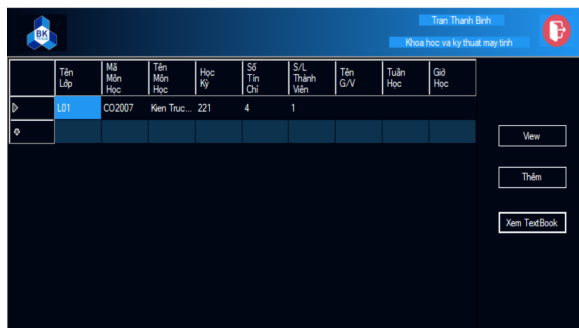
	Tên Lớp	Mã Môn Học	Tên Môn Học	Học Kỳ	Số Tín Chỉ	S/L Thành Viên	Tên GV/V	Tuần Học	Giáo Học
▶	LB1	PH1003	Vật lý 1	221	4	1	Chưa phân...		
	LB1	CO2007	Kiến Trúc...	221	4	1	Trần Thanh...		
⊕									

Ở bên phải bảng có các nút: Register, Cancel, TextBook. Ở dưới cùng có thông tin "Số môn: 2" và "Số tín chỉ: 8".

Hình 11: Student Main View

+ Với giao diện này, Student có thể đăng ký những môn mà khoa phát hành, hủy môn - đổi với đăng ký đợt 1 2 và xem các giáo trình các môn mà mình đã đăng ký - Chi tiết tại buổi Presentation.

- Tiến hành đăng nhập với tư cách là lecturer, ta có giao diện chính như sau



Giao diện chính của giảng viên. Thanh tiêu đề tương tự sinh viên nhưng người dùng là "Trần Thanh Bình". Bảng "Danh Sách Các Môn Đã Đăng Ký" chỉ có một dòng.

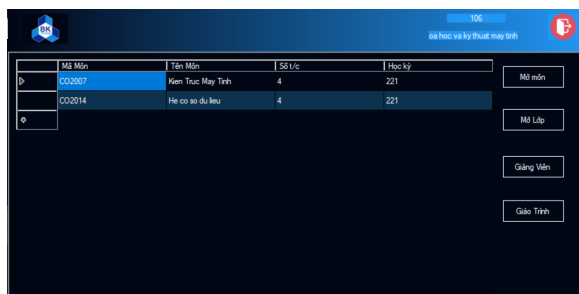
	Tên Lớp	Mã Môn Học	Tên Môn Học	Học Kỳ	Số Tín Chỉ	S/L Thành Viên	Tên GV/V	Tuần Học	Giáo Học
▶	LB1	CO2007	Kiến Trúc...	221	4	1			
⊕									

Ở bên phải bảng có các nút: View, Thêm, Xem TextBook.

Hình 12: Lecturer Main View

+ Với giao diện trên, giảng viên có thể xem danh sách thành viên của từng lớp, cho điểm, thêm textbook và xem các textbook của một lớp mà mình phụ trách. Chi tiết tại buổi Presentation.

- Tiến hành đăng nhập với tư cách là faculty, ta có giao diện chính như sau:



Giao diện chính của giáo viên. Thanh tiêu đề tương tự sinh viên nhưng người dùng là "Bùi Văn Hoàng". Bảng "Danh Sách Các Môn Đã Đăng Ký" có hai dòng.

	Mã Môn	Tên Môn	Số Lớp	Học Kỳ
▶	CO2007	Kiến Trúc Máy Tính	4	221
	CO2014	Hệ cơ sở dữ liệu	4	221
⊕				

Ở bên phải bảng có các nút: Mã môn, Mã Lớp, Giảng Viên, Giáo Trình.

Hình 13: Faculty Main View



- + Với giao diện trên, phòng đào tạo có thể mở môn, mở lớp và phân công công việc cho giảng viên và có thể xem giáo trình của một môn nhất định. Chi tiết tại buổi Presentation.
- Kho lưu trữ source code: https://github.com/khoidinh309/DMS_Assignment_University



TÀI LIỆU THAM KHẢO

- [1]: Ramez Elmasri and Shamkant Navathe, Fundamentals of Database Systems, Seventh Edition [7th Ed], 2015.
- [2]: Windows Forms documentation - .Net Framework by Microsoft
- [3]: MySQL Documentation - Oracle