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

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

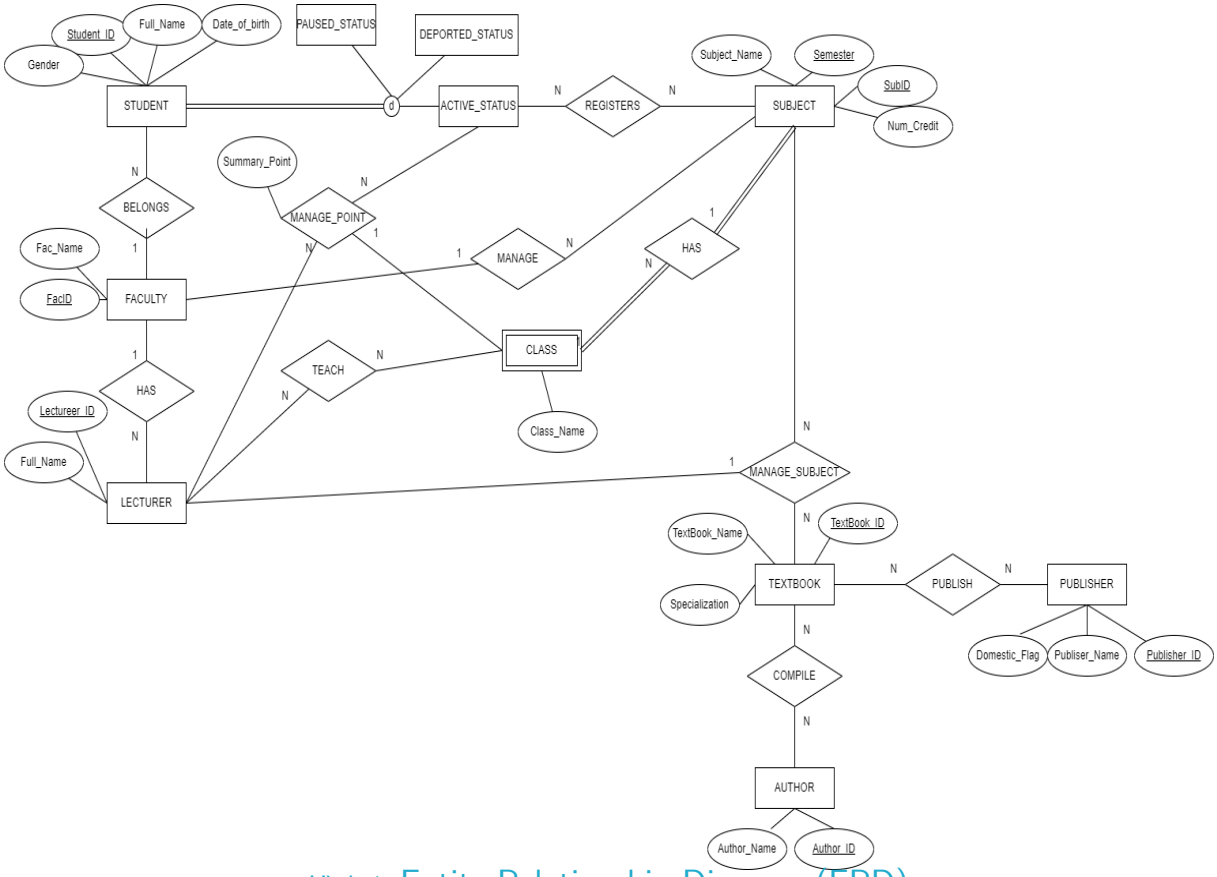
- Sinh viên (STUDENT) có trạng thái học tập bình thường (ACTIVE STATUS) có thể đăng ký một hoặc nhiều môn học, những sinh viên có trạng thái tạm dừng (PAUSED STATUS) hoặc buộc thôi học (DEPORTED STATUS) sẽ không thể đăng ký bất kỳ môn học nào ở học kỳ đó.
- Một môn học (SUBJECT) được mở và triển khai 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 mỗi 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 quan mối liên hệ có thể biết giảng viên thuộc khoa nào và khoa có những giảng viên nào. + SUBJECT - CLASS: Thông qua mỗi 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 quan 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	Biệu 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.

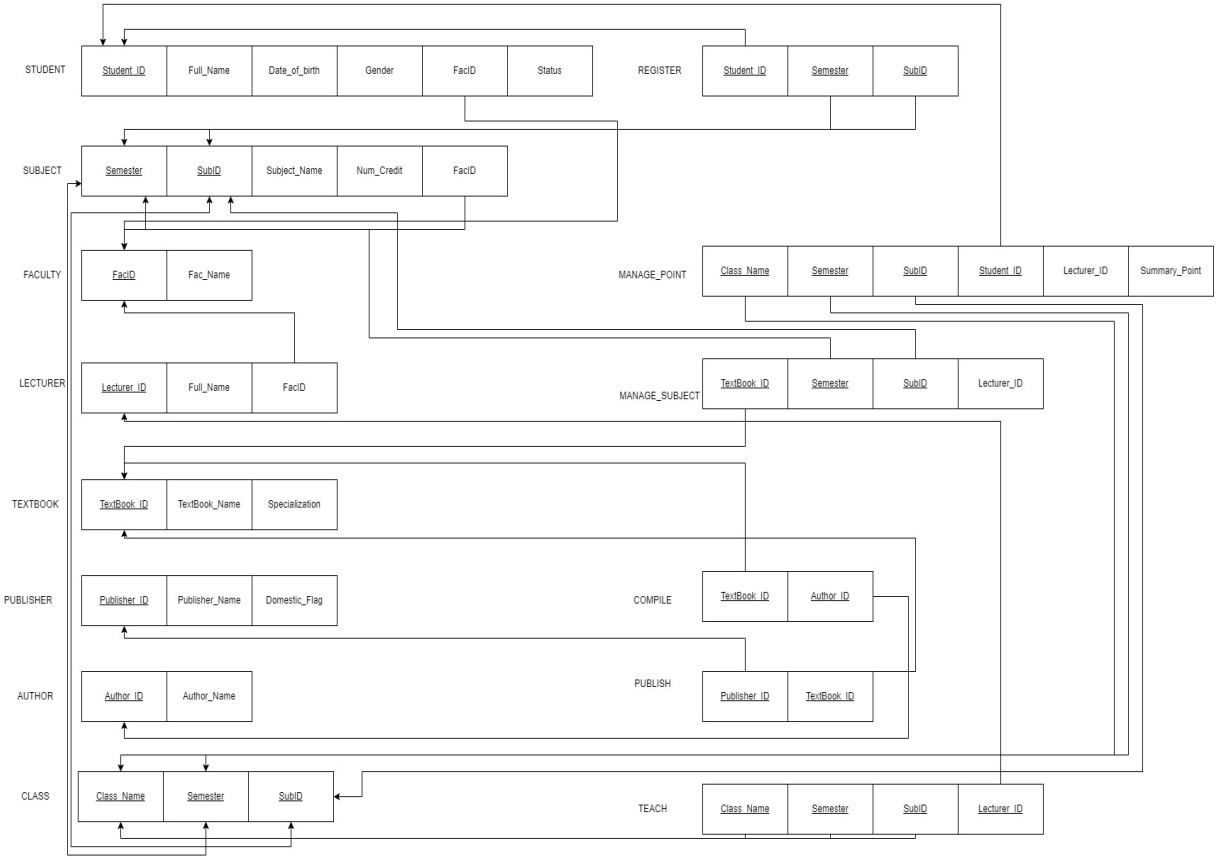
3.4 Mô Tả Thuộc Tính Của Thực Thể

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:

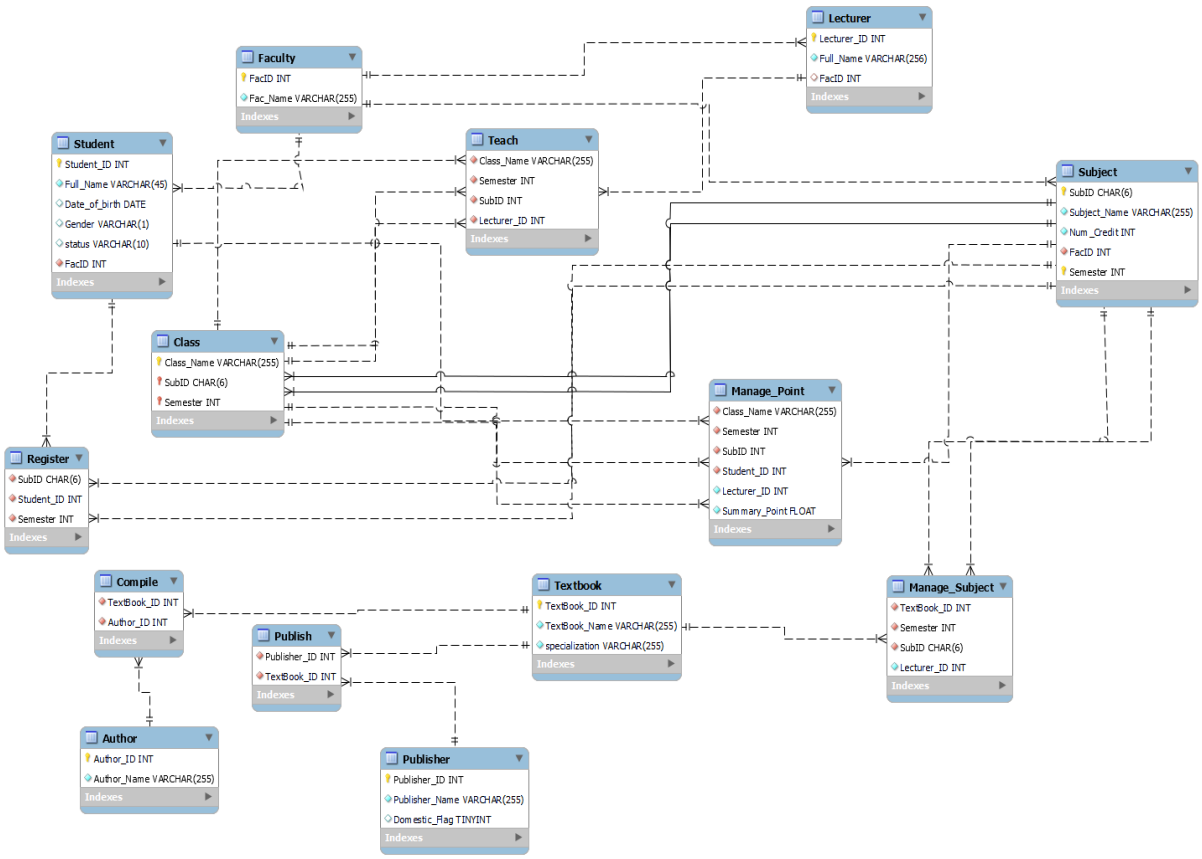


Hình 2: Relational Schema

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 Ánh Xạ

4.2.2 Generated Code

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 thoả 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 khóa.
- 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 các 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 xây dựng cơ sở dữ liệu khổng lồ, 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 bổ 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:

```
create database university;
use university;

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

- Lecturer:

```
create table if not exists lecturer(
  Lecturer_ID int not null,
    Full_Name varchar(255) not null,
    FacID int not null,
    primary key(Lecturer_ID),
    foreign key(FacID)
    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,
    Gender varchar(1) default null,
    current_status varchar(10) default null,
    FacID int not null,
```

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

- Subject:

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

- Class:

```
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)
);
```

- Publisher:

```
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)
);
```

- TextBook:

```
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:

```
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
    on update cascade,
    FOREIGN KEY (`Student_ID`)
    REFERENCES `University`.`Student` (`Student_ID`)
    on delete restrict
    on update cascade,
    FOREIGN KEY (`Semester`)
    REFERENCES `University`.`Subject` (`Semester`)
    on delete restrict
    on update cascade
);

```

- Manage_Point:

```
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,
    FOREIGN KEY (`Class_Name`)
    REFERENCES `University`.`Class` (`Class_Name`)
    on delete restrict
    on update cascade,
    FOREIGN KEY (`Semester`)
    REFERENCES `University`.`Class` (`Semester`)
    on delete restrict
    on update cascade,
    FOREIGN KEY (`SubID`)
    REFERENCES `University`.`Subject` (`Semester`)
    on delete restrict
    on update cascade,
    FOREIGN KEY (`Student_ID`)
    REFERENCES `University`.`Student` (`Student_ID`)
    on delete restrict
    on update cascade
);

```

- Manage_Subject:

```
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,
    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
    );

```

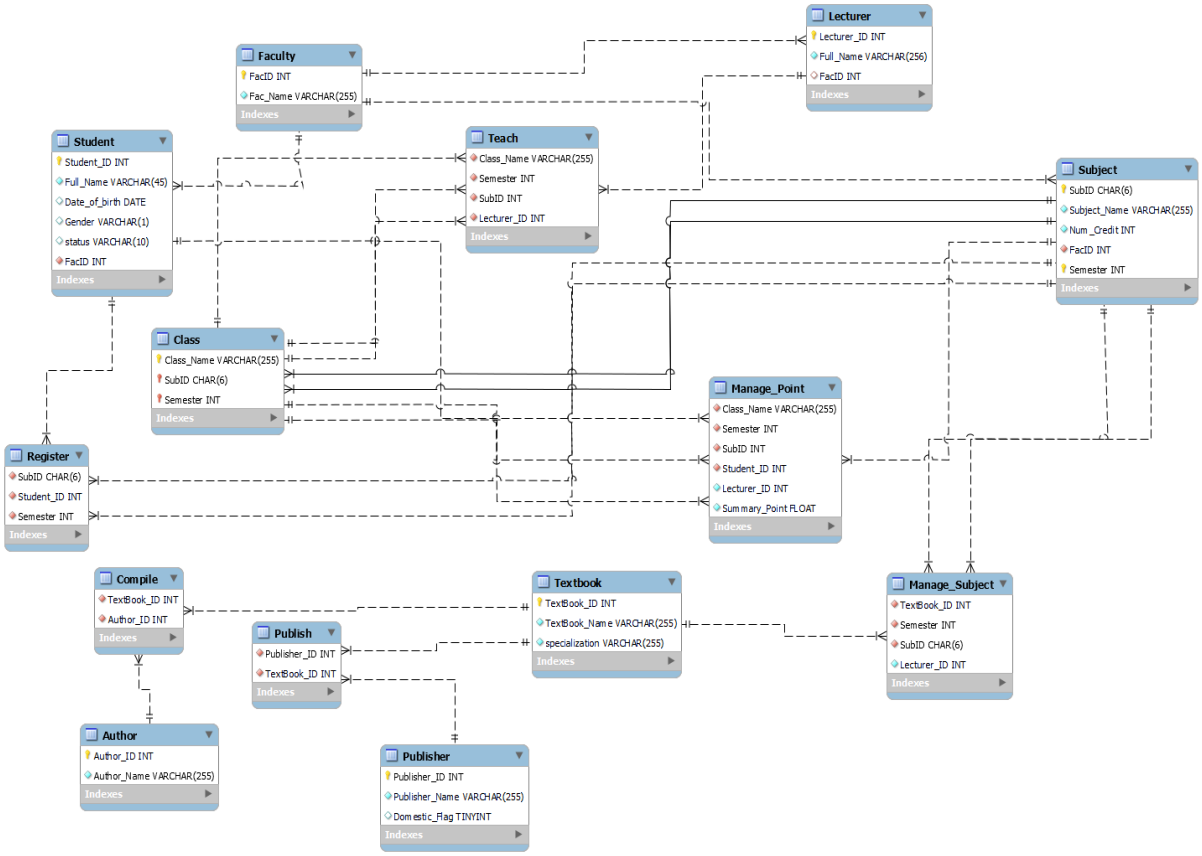
- Teach:

```
    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
```

```
        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,
         FOREIGN KEY (`Semester`)
        REFERENCES `University`.`Class` (`Semester`)
        on delete restrict
        on update cascade
    );

```

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