

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA KHOA HỌC & KỸ THUẬT MÁY TÍNH



DATABASE SYSTEM

Báo Cáo Assignment 2 (tổng hợp 1&2)

Đề Tài 1: TEACHING & LEARNING

GVHD: Phạm Nguyễn Hoàng Nam

SVTH: Bùi Xuân Thông - 1915350

Hoàng Văn Hiếu - 1913328

Nguyễn Minh Phú - 1914659

Tp. Hồ Chí Minh, Tháng 12/2021

Mục lục

1	1.Overview	4
2	2. Requirements Analysis	4
	2.1. Requirement.....	4
	2.2. Constraint.....	5
3	3. Conceptual Design	5
	3.1. Entity Relationship Diagram.....	5
	3.2. Entity Table.....	5
	3.3. Relationship Table.....	6
	3.4. Attribute table.....	8
4	4. Logic Design	9
	4.1.Relational Schema	9
	4.2.Sử dụng công cụ để thiết kế Relational schema.....	9
	4.2.1.Relational schema.....	10
	4.2.2.Code sql được tạo ra từ công cụ ERDplus.....	10
5	5.Normalization	11
	5.1.Dạng chuẩn 1NF	11
	5.2.Dạng chuẩn 2NF	11
	5.3.Dạng chuẩn 3NF	11
	5.4.Dạng chuẩn BCNF (Boyce Codd Normal Form).....	12
6	6. Physical Design	12
	6.1. Introduce Database Management System (DBMS)	12
	6.1.1. Giới thiệu tổng quan về MYSQL.....	12
	6.1.2. Ưu điểm và nhược điểm của MYSQL.....	12

	6.1.3. Sử dụng MYSQL vào bài tập lớn	13
	6.2. Tạo bảng bằng công cụ MySQL.....	14
	6.3. Table Relationships	19
	6.4. Insert Data.....	20
	5.4.1. Code	20
	6.4.2. Result	20
7	8.Kiểm soát quyền truy cập của người dùng dựa trên vai trò	21
	8.1. Data Definition Language-DDL	21
	8.2. Data Query Language-DQL	21
	8.3. Data Manipulation Language-DML	22
	8.4. Data Control Language-DCL	22
	8.5.Áp dụng vào hệ thống.....	22
8	9.Query 24	
9	10.Trigger, stored procedures 25	
	10.1.Trigger	25
	10.2.Stored procedures.....	25
10	11.Indexing 25	
11	12.Phát triển app 27	
12	13.Source code: 31	

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

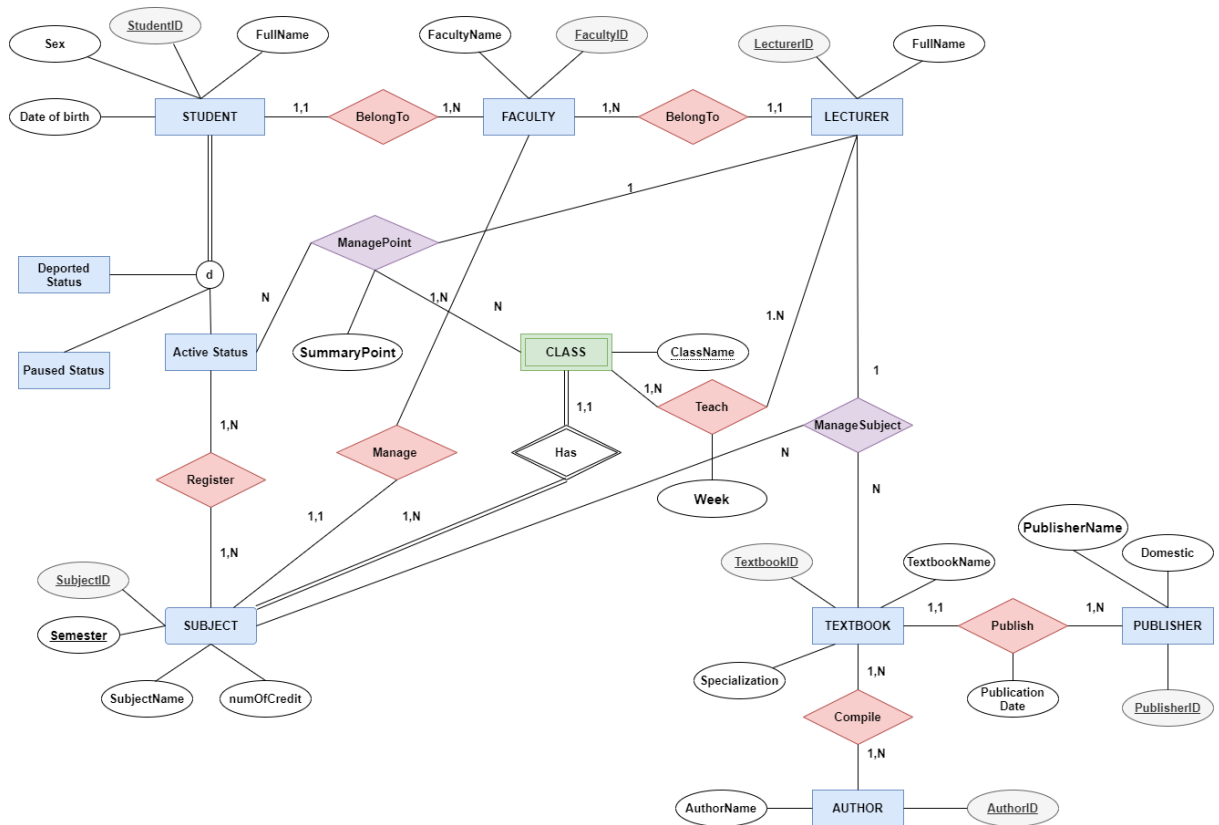
- Sinh viên (STUDENT) có trạng thái học tập bình thường (ACTIVE STATUS) có thể đăng ký một hoặc nhiều môn học, những sinh viên có trạng thái tạm dừng (PAUSED STATUS) hoặc buộc thôi học (DEPORTED STATUS) sẽ không thể đăng ký bất kỳ môn học nào ở học kỳ đó.
- Một môn học (SUBJECT) được mở và triển khai 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



Entity-Relationship Diagram (ERD)

3.2. Entity Table

Kiểu thực thể	Mô tả
---------------	-------

STUDENT	Lưu trữ thông tin của tất cả sinh viên, được chia làm 3 loại dựa theo 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. Relationship Table

Quan hệ	Mô tả
BelongTo	<p>Thể hiện quan hệ giữa :</p> <ul style="list-style-type: none"> - Khoa và sinh viên (FACULTY – STUDENT) - Khoa và giảng viên (FACULTY- LECTURER) <p>Thông qua quan hệ này có thể xác định sinh viên và giảng viên thuộc về khoa nào hay khoa đó có những sinh viên hay giảng viên nào.</p>
Register	<p>Thể hiện quan hệ giữa sinh viên có trạng thái học tập bình thường và môn học (ACTIVE STATUS – SUBJECT).</p>

	Thông qua quan hệ này có thể xác định sinh viên đăng ký những môn học nào.
Manage	Thể hiện quan hệ giữa khoa và môn học (FACULTY – SUBJECT) Thông qua đó, các khoa có thể quản lý các môn học thuộc chương trình đào tạo của mình.
Teach	Thể hiện quan hệ giữa giảng viên và lớp (LECTURER – CLASS) Thông qua đó, có thể xác định được các giảng viên dạy cho từng lớp.
Compile	Thể hiện quan hệ giữa tác giả và giáo trình (AUTHOR – TEXTBOOK) Thông qua đó, có thể xác định được người đã biên soạn cho giáo trình mà chúng ta sử dụng.
Publish	Thể hiện quan hệ giữa nhà xuất bản và giáo trình (PUBLISHER – TEXTBOOK) Thông qua đó, có thể xác định được nhà xuất bản của các giáo trình.
ManagePoint	Thể hiện quan hệ giữa ba kiểu thực thể: giảng viên chính, sinh viên và lớp học (LECTURER – STUDENT – CLASS). Thông qua đó, giảng viên chính của các lớp có thể ghi điểm cho sinh viên của mình vào cuối học kỳ.
ManageSubject	Thể hiện quan hệ giữa ba kiểu thực thể : giảng viên, môn học và giáo trình (LECTURER – SUBJECT – TEXTBOOK) Thông qua đó, giảng viên chỉ định những giáo trình nào được sử dụng cho môn học.
Has	Thể hiện quan hệ phụ thuộc của kiểu thực thể yếu lớp học (CLASS) phụ thuộc vào kiểu thực thể mạnh là môn học (SUBJECTS).

	Thông qua đó có thể xác định được lớp học được mở cho môn học nào.
--	--

3.4. Attribute table

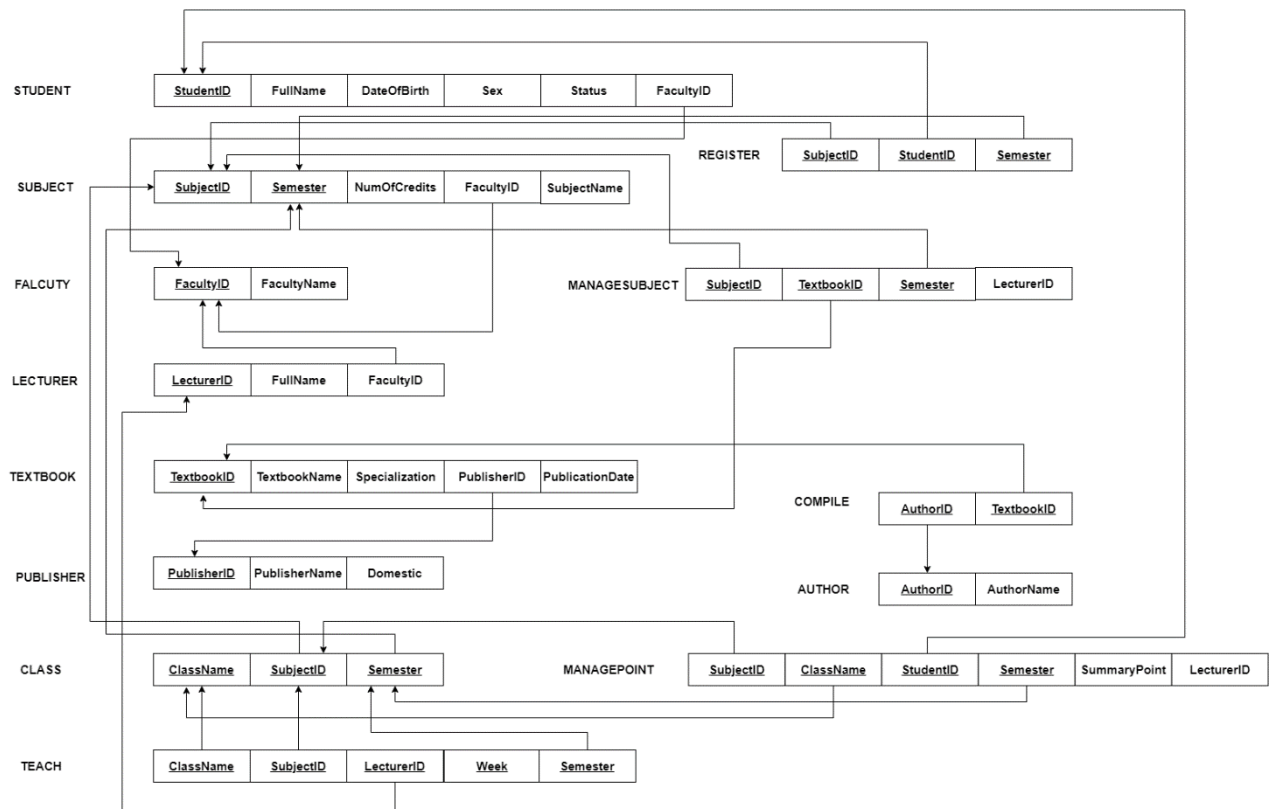
Thực thể (Entity)	Thuộc tính (Attribute)	Khóa chính (Primary Key)
STUDENT	<ul style="list-style-type: none"> - StudentID: mã số sinh viên - FullName: họ và tên - DateOfBirth: ngày sinh - Sex: giới tính 	<u>StudentID</u>
LECTURER	<ul style="list-style-type: none"> - LecturerID: mã số giảng viên - FullName: họ và tên 	<u>LecturerID</u>
FACULTY	<ul style="list-style-type: none"> - FacultyID: mã khoa - FacultyName: tên của khoa 	<u>FacultyID</u>
SUBJECT	<ul style="list-style-type: none"> - SubjectID: mã môn học - SubjectName: tên môn học - NumberOfCredit: số tín chỉ 	<u>SubjectID</u>
AUTHOR	<ul style="list-style-type: none"> - AuthorID: mã số tác giả - AuthorName: tên tác giả 	<u>AuthorID</u>
TEXTBOOK	<ul style="list-style-type: none"> - TextbookID: mã giáo trình - TextbookName: tên giáo trình - Specialization: lĩnh vực, chuyên ngành 	<u>TextbookID</u>
PUBLISHER	<ul style="list-style-type: none"> - PublisherID: mã nhà xuất bản - PublisherName: tên nhà xuất bản 	<u>PublisherID</u>

	- Domestic: trong nước	
CLASS (WEAK ENTITY)	- ClassName: tên lớp (khóa riêng phần phụ thuộc vào kiểu thực thể mạnh Subject)	<u>ClassName; SubjectID</u>

4. Logic Design

4.1.Relational Schema

Sau khi ánh xạ:

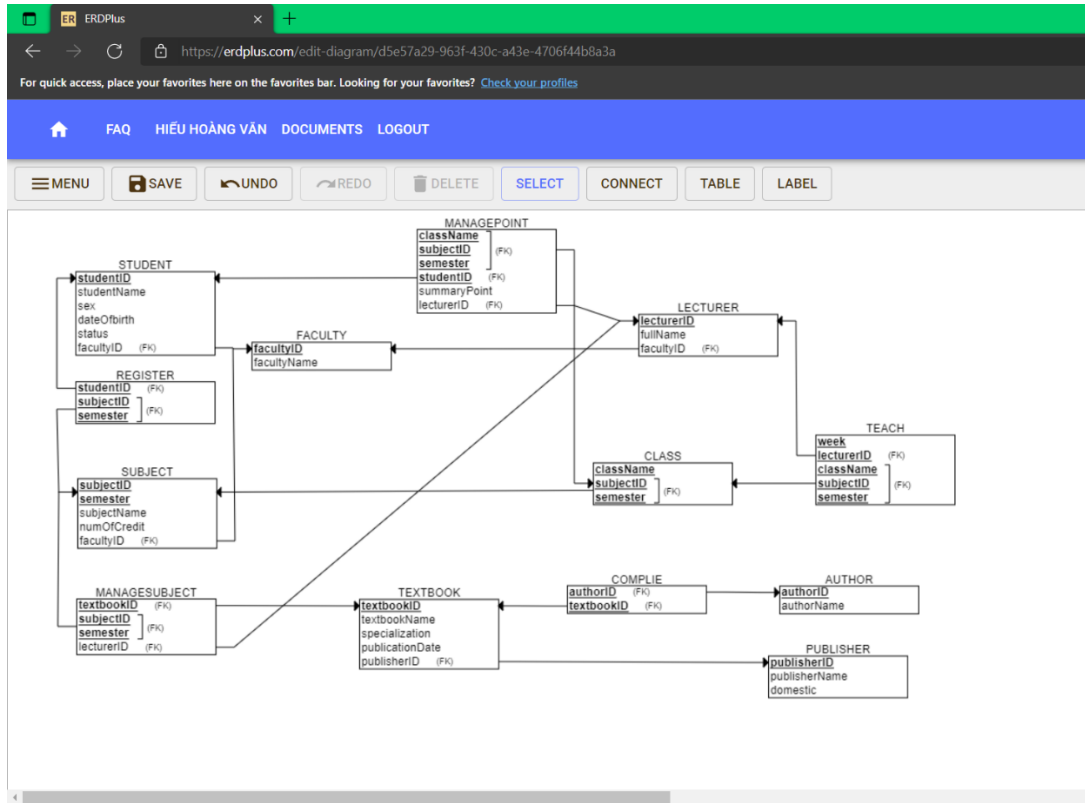


Relational Schema

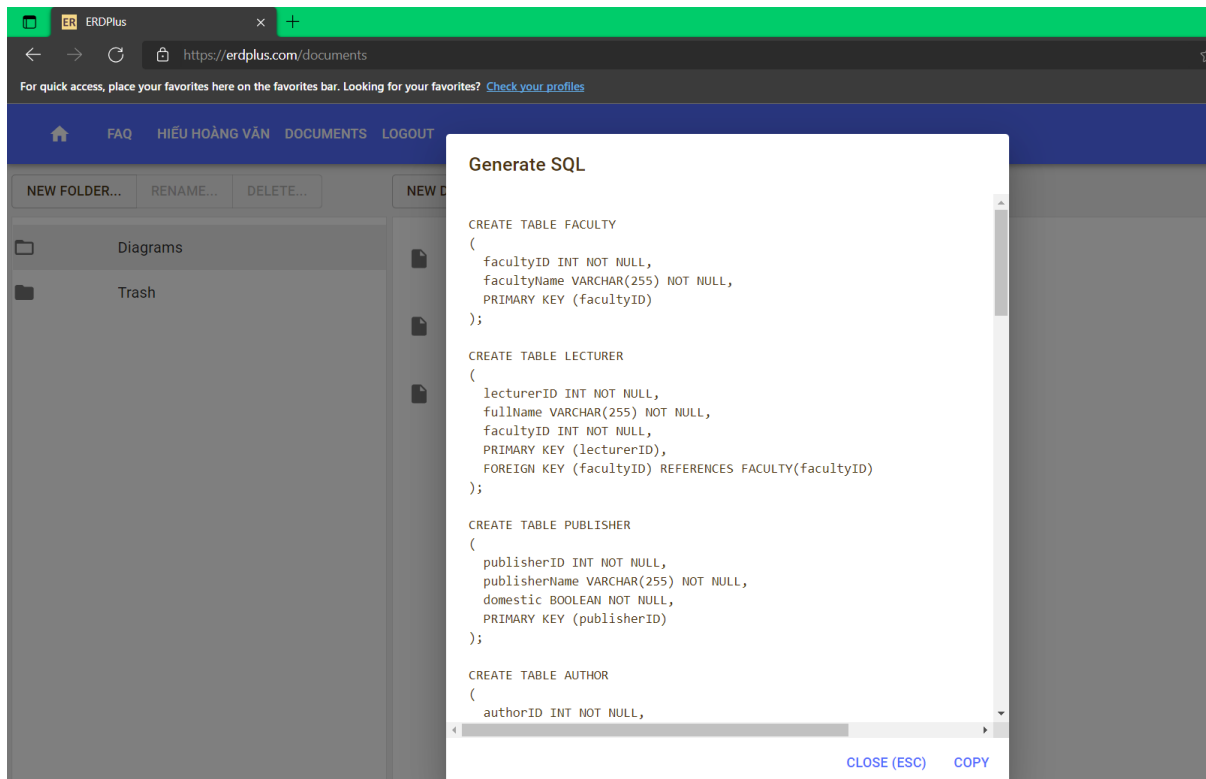
4.2.Sử dụng công cụ để thiết kế Relational schema

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... nhóm quyết định chọn công cụ ERDplus để thiết kế.

4.2.1. Relational schema



4.2.2. Code sql được tạo ra từ công cụ ERDplus



```

    CREATE TABLE FACULTY
    (
        facultyID INT NOT NULL,
        facultyName VARCHAR(255) NOT NULL,
        PRIMARY KEY (facultyID)
    );

    CREATE TABLE LECTURER
    (
        lecturerID INT NOT NULL,
        fullName VARCHAR(255) NOT NULL,
        facultyID INT NOT NULL,
        PRIMARY KEY (lecturerID),
        FOREIGN KEY (facultyID) REFERENCES FACULTY(facultyID)
    );

    CREATE TABLE PUBLISHER
    (
        publisherID INT NOT NULL,
        publisherName VARCHAR(255) NOT NULL,
        domestic BOOLEAN NOT NULL,
        PRIMARY KEY (publisherID)
    );

    CREATE TABLE AUTHOR
    (
        authorID INT NOT NULL,
  
```

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 khóa mà phụ thuộc hàm vào thuộc tính không khóa.

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)

Trong phạm vi môn học, sử dụng công cụ MySQL để tạo bảng 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, có khả năng thay đổi mô hình sử dụng phù hợp với điều kiện công việc khả chuyển.

Với tốc độ và tính bảo mật cao, MySQL thích hợp với các ứng dụng có truy cập cơ sở dữ liệu trên internet.

6.1.2. Ưu điểm và nhược điểm của MySQL

Ưu điểm:

- Sử dụng dễ dàng : công cụ dễ sử dụng và hoạt động trên nhiều hệ điều hành
- Tính bảo mật cao: phù hợp với các ứng dụng có truy cập cơ sở dữ liệu trên internet.
- Đa tính năng: có thể hỗ trợ hàng loạt các chức năng SQL từ hệ quản trị cơ sở dữ liệu quan hệ trực tiếp và cả gián tiếp.
- Khả năng mở rộng và mạnh mẽ : có khả năng xử lý khối dữ liệu lớn và có thể mở rộng
- Tương thích trên nhiều hệ điều hành: cung cấp phương tiện mà các máy khách có thể chạy trên cùng một máy tính với máy chủ hoặc trên một máy tính khác
- Cho phép khôi phục: cho phép các transaction được khôi phục, cam kết và phục hồi sự cố.

Nhược điểm:

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

6.1.3. Sử dụng MYSQL vào 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, lại còn 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 cũng có các nhược điểm, tuy nhiên trong 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, chúng em quyết định sử dụng MySQL để lập trình hệ cơ sở dữ liệu.

6.2. Tạo bảng bằng công cụ MySQL

6.2.1. FACULTY

```
create schema school;  
use school;  
  
CREATE TABLE IF NOT EXISTS FACULTY(  
    facultyID INT NOT NULL,  
    facultyName VARCHAR (255) NOT NULL,  
  
    PRIMARY KEY (facultyID)  
);
```

6.2.2. LECTURER

```
CREATE TABLE IF NOT EXISTS LECTURER(  
    lecturerID INT NOT NULL,  
    fullName VARCHAR (255) NOT NULL,  
    facultyID INT NOT NULL,  
  
    PRIMARY KEY (lecturerID),  
    FOREIGN KEY (facultyID)  
        REFERENCES FACULTY (facultyID)  
        ON DELETE RESTRICT  
        ON UPDATE CASCADE  
);
```

6.2.3. STUDENT

```
CREATE TABLE IF NOT EXISTS STUDENT (  
    studentID INT NOT NULL ,  
    fullName VARCHAR (255) NOT NULL ,  
    dateOfBirth date,  
    sex VARCHAR(10) DEFAULT NULL,  
    status VARCHAR(10) DEFAULT NULL,  
    facultyID INT NOT NULL,  
  
    PRIMARY KEY (studentID),  
    FOREIGN KEY (facultyID)  
        REFERENCES FACULTY (facultyID)  
        ON DELETE RESTRICT  
        ON UPDATE CASCADE  
);
```

6.2.4. SUBJECT

```
CREATE TABLE IF NOT EXISTS SUBJECT(  
    subjectID CHAR (6) NOT NULL,  
    subjectName VARCHAR (255) NOT NULL,  
    facultyID INT NOT NULL ,  
    numOfCredits INT NOT NULL CHECK (numOfCredits > 0 AND numOfCredits <  
4),  
    semester INT NOT NULL,  
  
    PRIMARY KEY (subjectID, semester),  
    FOREIGN KEY (facultyID)  
        REFERENCES FACULTY (facultyID)  
        ON DELETE RESTRICT  
        ON UPDATE CASCADE  
);
```

6.2.5. CLASS

```
CREATE TABLE IF NOT EXISTS CLASS(  
    className VARCHAR (255) NOT NULL,  
    subjectID CHAR (6) NOT NULL,  
    semester INT NOT NULL,  
  
    PRIMARY KEY (className, subjectID, semester),  
    FOREIGN KEY (subjectID, semester)  
        REFERENCES SUBJECT (subjectID, semester)  
        ON DELETE RESTRICT  
        ON UPDATE CASCADE  
);
```

6.2.6. PUBLISHER

```
CREATE TABLE IF NOT EXISTS PUBLISHER(  
    publisherID INT NOT NULL,  
    publisherName VARCHAR (255) NOT NULL,  
    domestic BOOLEAN DEFAULT FALSE,  
  
    PRIMARY KEY (publisherID)  
);
```

6.2.7. TEXTBOOK

```
CREATE TABLE IF NOT EXISTS TEXTBOOK(  
    textbookID INT NOT NULL,  
    textbookName VARCHAR (255) NOT NULL,  
    specialization VARCHAR (255) NOT NULL,
```

```
publisherID INT NOT NULL,  
publishcationDate DATE,  
  
PRIMARY KEY (textbookID),  
FOREIGN KEY (publisherID)  
REFERENCES PUBLISHER (publisherID)  
ON DELETE RESTRICT  
ON UPDATE CASCADE  
);
```

6.2.8. AUTHOR

```
CREATE TABLE IF NOT EXISTS AUTHOR(  
authorID INT NOT NULL,  
authorName VARCHAR (255) NOT NULL,  
PRIMARY KEY (authorID)  
);
```

6.2.9. TEACH

```
CREATE TABLE IF NOT EXISTS TEACH(  
className VARCHAR(255) NOT NULL,  
subjectID CHAR(6) NOT NULL,  
lecturerID INT NOT NULL,  
week INT NOT NULL,  
semester INT NOT NULL,  
  
PRIMARY KEY (className, subjectID, week, lecturerID, semester),  
FOREIGN KEY (className)  
REFERENCES CLASS (className)  
ON DELETE RESTRICT  
ON UPDATE CASCADE,  
FOREIGN KEY (subjectID, semester)  
REFERENCES CLASS (subjectID, semester)  
ON DELETE RESTRICT  
ON UPDATE CASCADE,  
FOREIGN KEY (lecturerID)  
REFERENCES LECTURER (lecturerID)  
ON DELETE RESTRICT  
ON UPDATE CASCADE  
);
```

6.2.10. REGISTER

```
CREATE TABLE IF NOT EXISTS REGISTER (
```



```
subjectID CHAR(6) NOT NULL,  
studentID INT NOT NULL,  
semester INT NOT NULL,  
  
PRIMARY KEY (subjectID , studentID, semester),  
FOREIGN KEY (subjectID, semester)  
REFERENCES SUBJECT (subjectID, semester)  
ON DELETE RESTRICT  
ON UPDATE CASCADE,  
FOREIGN KEY (studentID)  
REFERENCES STUDENT (studentID)  
ON DELETE RESTRICT  
ON UPDATE CASCADE  
);
```

6.2.11. MANAGESUBJECT

```
CREATE TABLE IF NOT EXISTS MANAGESUBJECT(  
subjectID CHAR(6) NOT NULL,  
textbookID INT NOT NULL,  
lecturerID INT NOT NULL,  
semester INT NOT NULL,  
  
PRIMARY KEY (subjectID, textbookID, semester),  
FOREIGN KEY (subjectID, semester)  
REFERENCES SUBJECT (subjectID, semester)  
ON DELETE RESTRICT  
ON UPDATE CASCADE ,  
FOREIGN KEY (textbookID)  
REFERENCES TEXTBOOK (textbookID)  
ON DELETE RESTRICT  
ON UPDATE CASCADE  
);
```

6.2.12. COMPILE

```
CREATE TABLE IF NOT EXISTS COMPILE(  
authorID INT NOT NULL,  
textBookID INT NOT NULL,  
  
PRIMARY KEY (authorID, textBookID),  
FOREIGN KEY (authorID)  
REFERENCES AUTHOR (authorID)  
ON DELETE RESTRICT  
ON UPDATE CASCADE ,  
FOREIGN KEY (textBookID)  
REFERENCES TEXTBOOK (textBookID)
```

```
ON DELETE RESTRICT  
ON UPDATE CASCADE  
);
```

6.2.13. MANAGEPOINT

```
CREATE TABLE IF NOT EXISTS MANAGEPOINT (  
  subjectID CHAR(6) NOT NULL,  
  className VARCHAR(255) NOT NULL,  
  studentID INT NOT NULL,  
  lecturerID INT NOT NULL,  
  summaryPoINT FLOAT(1) CHECK ((summaryPoINT >= 0 AND  
summaryPoINT <= 10) OR summaryPoINT IS NULL ),  
  semester INT NOT NULL,  
  
  PRIMARY KEY (subjectID, className, studentID, semester),  
  FOREIGN KEY (subjectID, semester)  
    REFERENCES CLASS (subjectID, semester)  
    ON DELETE RESTRICT  
    ON UPDATE CASCADE ,  
  FOREIGN KEY (className)  
    REFERENCES CLASS (className)  
    ON DELETE RESTRICT  
    ON UPDATE CASCADE,  
  FOREIGN KEY (studentID)  
    REFERENCES STUDENT (studentID)  
    ON DELETE RESTRICT  
    ON UPDATE CASCADE  
);
```

6.3. Table Relationships

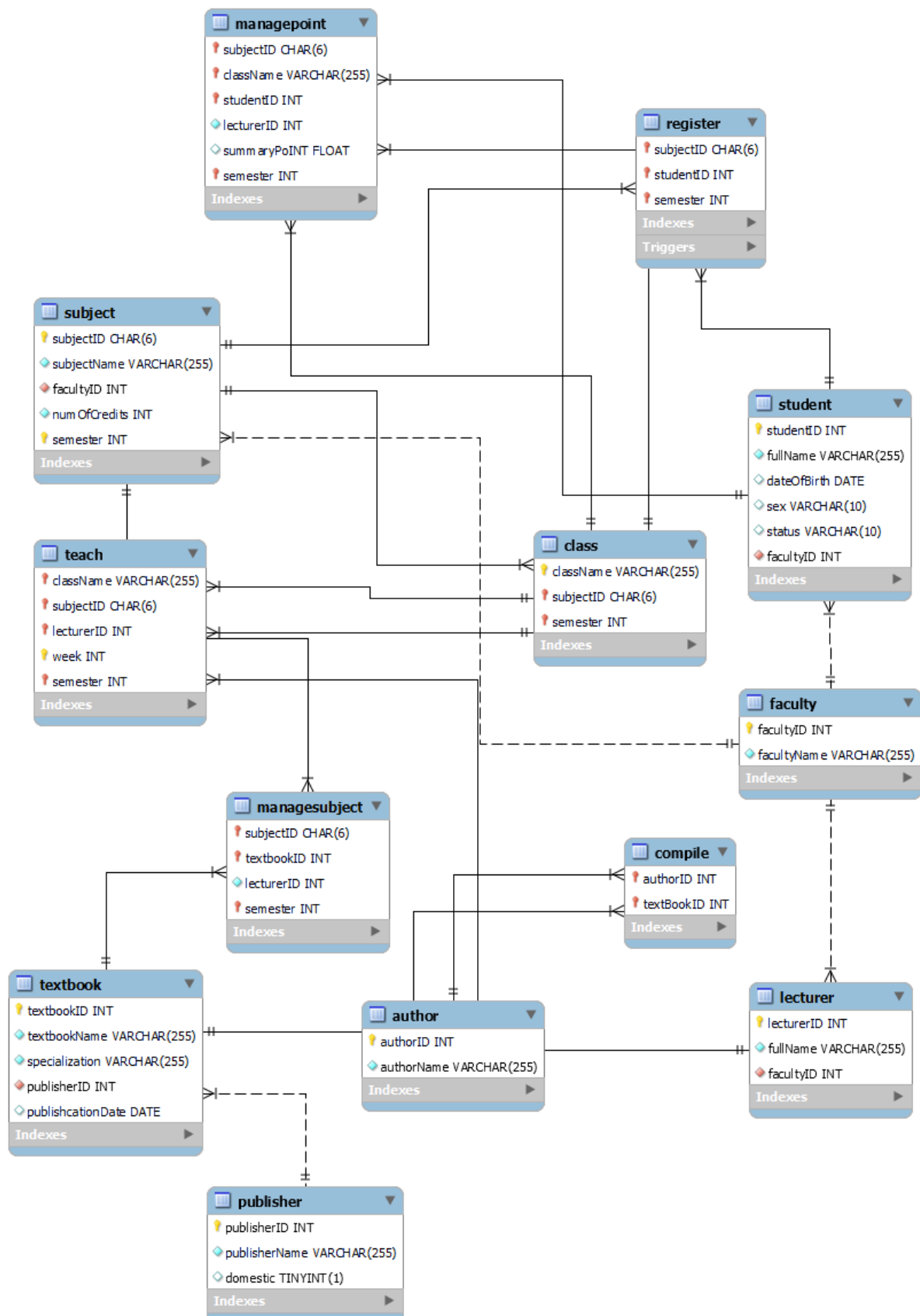


Table Relationship in MySQL

6.4. Insert Data

5.4.1. Code

*/*Dump data into Table Student and Faculty*/*

```
INSERT INTO `FACULTY` (`facultyID`, `facultyName`)
VALUES (106, 'Khoa học và kĩ thuật máy tính'),
      (113, 'Kĩ thuật hóa học'),
      (115, 'Quản lí công nghiệp');
```

```
INSERT INTO `Student` (`studentID`, `fullName`, `dateOfBirth`, `sex`, `status`, `facultyID`)
VALUES (1914262, 'Nguyen Ngoc Trinh', '2000-10-04', 'Female', 'Active', 106),
      (1921345, 'Luong Huu Phuoc', '2001-10-03', 'Male', 'Active', 113),
      (1922415, 'Nhan Phuc Vinh', '2001-03-03', 'Male', 'Active', 115);
```

6.4.2. Result

facultyID	facultyName
106	Khoa học và kĩ thuật máy tính
113	Kĩ thuật hóa học
115	Quản lí công nghiệp
NULL	NULL

FACULTY Table

studentID	fullName	dateOfBirth	sex	status	facultyID
1914262	Nguyen Ngoc Trinh	2000-10-04	Female	Active	106
1914562	Nguyen Ngoc Trinh	2000-10-04	Female	Active	106
1921345	Luong Huu Phuoc	2001-10-03	Male	Active	113
1922415	Nhan Phuc Vinh	2001-03-03	Male	Active	115
NULL	NULL	NULL	NULL	NULL	NULL

STUDENT Table

8. Kiểm soát quyền truy cập của người dùng dựa trên vai trò

Ngôn ngữ truy vấn có cấu trúc (SQL) là ngôn ngữ cơ sở dữ liệu có thể thực hiện các hoạt động nhất định trên cơ sở dữ liệu hiện có và cũng có thể sử dụng ngôn ngữ này để tạo cơ sở dữ liệu. SQL sử dụng các lệnh nhất định như Tạo, Thả, Chèn, ... để thực hiện các tác vụ cần thiết.

Các lệnh SQL này được chia thành 4 loại.

8.1. Data Definition Language-DDL

DDL hay Ngôn ngữ Định nghĩa Dữ liệu bao gồm các lệnh SQL có thể được sử dụng để xác định lược đồ cơ sở dữ liệu. Nó chỉ đơn giản giải quyết các mô tả của lược đồ cơ sở dữ liệu và được sử dụng để tạo và sửa đổi cấu trúc của các đối tượng trong cơ sở dữ liệu.

DDL là một tập hợp các lệnh SQL được sử dụng để tạo, sửa đổi và xóa cấu trúc cơ sở dữ liệu nhưng không phải dữ liệu. Những lệnh này thường không được sử dụng bởi một người dùng thông thường, những người sẽ truy cập cơ sở dữ liệu thông qua một ứng dụng.

Danh sách các lệnh DDL:

- CREATE : tạo cơ sở dữ liệu hoặc các đối tượng của nó (như bảng, chỉ mục, hàm, dạng xem, thủ tục lưu trữ và trình kích hoạt).
- DROP : xóa các đối tượng khỏi cơ sở dữ liệu.
- ALTER : thay đổi cấu trúc của cơ sở dữ liệu.
- TRUNCATE : loại bỏ tất cả các bản ghi khỏi một bảng, bao gồm tất cả các khoảng COMMENT : thêm nhận xét vào từ điển dữ liệu.
- RENAME : đổi tên một đối tượng hiện có trong cơ sở dữ liệu.

8.2. Data Query Language-DQL

DQL hay ngôn ngữ truy vấn dữ liệu được sử dụng để thực hiện các truy vấn về dữ liệu trong các đối tượng lược đồ. Mục đích của các lệnh DQL là lấy một số quan hệ lược đồ dựa trên truy vấn được chuyển đến nó. Chúng ta có thể định nghĩa DQL như sau, nó là

một thành phần của câu lệnh SQL cho phép lấy dữ liệu từ cơ sở dữ liệu và áp đặt thứ tự cho nó.

Danh sách DQL:

SELECT : Nó được sử dụng để lấy dữ liệu từ cơ sở dữ liệu.

8.3. Data Manipulation Language-DML

Các lệnh SQL xử lý thao tác dữ liệu có trong cơ sở dữ liệu thuộc về DML hay ngôn ngữ thao tác dữ liệu và điều này bao gồm hầu hết các câu lệnh SQL. Nó là thành phần của câu lệnh SQL kiểm soát quyền truy cập vào dữ liệu và cơ sở dữ liệu. Về cơ bản, các câu lệnh DCL được nhóm với các câu lệnh DML.

Danh sách các lệnh DML:

- INSERT : chèn (thêm) dữ liệu vào bảng.
- UPDATE : cập nhật dữ liệu hiện có trong bảng.
- DELETE : xóa các bản ghi khỏi một bảng cơ sở dữ liệu.
- LOCK: Đồng thời điều khiển bảng.
- CALL: Gọi một chương trình con PL / SQL hoặc JAVA.
- EXPLAIN PLAN: mô tả đường dẫn truy cập đến dữ liệu

8.4. Data Control Language-DCL

DCL hay ngôn ngữ điều khiển dữ liệu bao gồm các lệnh như GRANT và REVOKE chủ yếu xử lý các quyền, quyền hạn và các điều khiển khác của hệ thống cơ sở dữ liệu.

Danh sách các lệnh DCL:

- GRANT: cung cấp cho người dùng đặc quyền truy cập vào cơ sở dữ liệu.
- REVOKE: thu hồi các đặc quyền truy cập của người dùng được cấp bằng cách sử dụng lệnh GRANT.

8.5. Áp dụng vào hệ thống

Trong một hệ thống cơ sở dữ liệu, có nhiều kiểu người dùng, mỗi người dùng có những đặc quyền cụ thể để truy cập và thao tác với cơ sở dữ liệu. Tuy nhiên, việc để tất cả người dùng đăng nhập vào cơ sở dữ liệu với vai trò là một “Root”- nghĩa là họ có tất

cả các quyền hạn đối với cơ sở dữ liệu, điều này dẫn đến nhiều rủi ro khi một ai đó có ý định phá hoại cơ sở dữ liệu như chỉnh sửa bảng hay thậm chí là xóa bảng.

Do đó, để khắc phục vấn đề này và tăng khả năng an toàn cho dữ liệu, hệ thống được thiết kế để cung cấp cho người dùng với 4 vai trò khác nhau nhằm hạn chế khả năng truy cập và chỉnh sửa dữ liệu: STUDENT (sinh viên), LECTURER (giảng viên), FACULTY (khoa quản lý) và OFFICE (phòng đào tạo).

Permission Table

Table	OFFICE	FACULTY	LECTURER	STUDENT
Student	DQL,DML	DQL	DQL	DQL
Lecturer	DQL,DML	DQL	DQL	DQL
Faculty	DQL,DML	DQL	DQL	DQL
Class	DQL	DQL,DML	DQL	DQL
Subject	DQL	DQL,DML	DQL	DQL
Textbook	DQL	DQL	DQL,DML	DQL
Register	DQL,DML	DQL	DQL	DQL,DML
Author	DQL	DQL	DQL	DQL
Teach	DQL	DQL,DML	DQL	DQL
Publisher	DQL	DQL	DQL	DQL
Managepoint	DQL	DQL	DQL,DML	DQL
Managesubject	DQL	DQL,DML	DQL	DQL
Compile	DQL	DQL	DQL	DQL

Với việc phân chia người dùng theo các vai trò, mỗi người dùng chỉ có một số quyền hạn nhất định để tương tác với cơ sở dữ liệu.

Ví dụ, từ bảng trên có thể thấy, tất cả người dùng đều có 1 quyền hạn chung chính ra DQL, hay người dùng có thể INSERT để lấy dữ liệu từ các bảng, tuy nhiên

không phải người dùng nào cũng có thể tùy ý chỉnh sửa dữ liệu, cụ thể đối với bảng “REGISTER” chứa thông tin đăng ký môn học của sinh viên, sinh viên có thể thực hiện các hành động như thêm sửa xóa (DML) hay xem lại những cái mình đã đăng ký, nhưng giảng viên “LECTURER” chỉ có thể xem nhưng không thể thực hiện hiện các thao tác dữ liệu khác.

Truy cập theo vai trò giúp cơ sở dữ liệu trở nên an toàn và bảo mật hơn, giảm thiểu các nguy cơ về rủi ro dữ liệu....

Phân quyền cho 2 vai trò là sinh viên (student) và giảng viên (lecturer):

```
-- Creating Roles on
-- student
DROP ROLE IF EXISTS 'student';
CREATE ROLE 'student ';
-- lecturer
DROP ROLE IF EXISTS 'lecturer';
CREATE ROLE 'lecturer ';

-- Granting Privileges
-- Grant student
GRANT SELECT on school.student to 'student';
GRANT SELECT on school.lecturer to 'student';
GRANT SELECT on school.faculty to 'student';
GRANT SELECT , INSERT , UPDATE on school.register to 'student';
GRANT SELECT on school.subject to 'student';
-- Grant lecturer
GRANT INSERT , UPDATE, SELECT ON school.textbook to 'lecturer ';
GRANT SELECT , INSERT , UPDATE , DELETE ON school.managepoint to 'lecturer
';
GRANT SELECT ON school.register to 'lecturer ';
GRANT SELECT ON school.class to 'lecturer ';
GRANT SELECT ON school.faculty to 'lecturer ';
```

9.Query

Các **query** về yêu cầu về dữ liệu của mỗi nhóm người dùng được lưu trữ tại file *all_query.sql*

10. Trigger, stored procedures

10.1. Trigger

Trigger là một đoạn procedure code, chỉ được vận hành khi có một sự kiện xảy ra. Có nhiều loại sự kiện khác nhau để kích hoạt trigger trong SQL. Có thể kể đến như việc chèn các hàng trong bảng, thay đổi cấu trúc bảng hoặc thậm chí người dùng đăng nhập vào một phiên trong SQL.

Có ba đặc điểm chính làm cho trigger trong SQL khác với các **stored procedures**:

- Người dùng không thể thực hiện thủ công các trigger.
- Trigger không thể nhận tham số, nó chỉ sử dụng NEW và OLD để lấy các giá trị đang bị ảnh hưởng

Trigger được lưu trữ tại file [*all_trigger.sql*](#)

10.2. Stored procedures

Stored procedure là tập hợp một hoặc nhiều câu lệnh T-SQL thành một nhóm đơn vị xử lý logic và được lưu trữ trên Database Server. Khi một câu lệnh gọi chạy stored procedure lần đầu tiên thì SQL Server sẽ chạy nó và lưu trữ vào bộ nhớ đệm, gọi là plan cache, những lần tiếp theo SQL Server sẽ sử dụng lại plan cache nên sẽ cho tốc độ xử lý tối ưu.

Stored procedure rất tiện lợi cho người quản trị database (DBA), nó giúp DBA tạo ra những nhóm câu lệnh và gửi đến một bộ phận khác mà họ sẽ không cần quan tâm đến nội dung bên trong stored procedure có gì, họ chỉ quan tâm đến tham số đầu vào và đầu ra.

Stored procedures được lưu trữ tại file [*all_procedure.sql*](#)

11. Indexing

Các chỉ mục hay index trong SQL là các bảng tra cứu đặc biệt mà công cụ tìm kiếm cơ sở dữ liệu có thể sử dụng để tăng tốc độ truy xuất dữ liệu. Đơn giản chỉ cần thiết lập một chỉ số là một con trỏ đến dữ liệu trong một bảng. Một chỉ mục trong một cơ sở dữ liệu rất giống với một chỉ mục ở mặt sau của một cuốn sách.

Ví dụ: nếu bạn muốn tham chiếu tất cả các trang trong một cuốn sách thảo luận về một chủ đề nhất định. Trước tiên bạn hãy tham chiếu chỉ mục, liệt kê tất cả các chủ đề theo thứ tự bảng chữ cái và sau đó được gọi đến một hoặc nhiều số trang cụ thể.

Một chỉ mục giúp tăng tốc các truy vấn SELECT và các mệnh đề WHERE , nhưng nó làm chậm dữ liệu nhập vào, với các câu lệnh UPDATE và INSERT . Các chỉ mục có thể được tạo ra hoặc bỏ đi mà không ảnh hưởng đến dữ liệu.

Áp dụng vào bài tập lớn:

- Tạo index cho facultyName ở bảng faculty:

```
CREATE INDEX index_for_name
ON faculty (facultyName);
```

- Từ đó việc truy vấn facultyName ở bảng faculty sẽ được thực hiện nhanh hơn:

```
SELECT * FROM faculty WHERE facultyName = 'faltest250';
```

- Đánh giá
 - Trước khi thêm index vào faculty

Your SQL query has been executed successfully.

```
EXPLAIN SELECT * FROM `faculty` WHERE facultyName = 'faltest250';
```

[Edit inline] [Edit] [Skip Explain SQL] [Analyze Explain at mariadb.org] [Create PHP code]

+ Options

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	faculty	ALL	NULL	NULL	NULL	NULL	1830	Using where

Cột “rows” cho biết số lượng hàng mà MySQL tin rằng nó phải kiểm tra từ bảng faculty, để thực hiện truy vấn. Như ta thấy, câu truy vấn dự đoán sẽ duyệt qua tổng cộng là 1830 rows để trả về kết quả.

- Sau khi thêm index vào faculty

Your SQL query has been executed successfully.

```
EXPLAIN SELECT * FROM faculty WHERE facultyName = 'faltest250';
```

[Edit inline] [Edit] [Skip Explain SQL] [Analyze Explain at mariadb.org] [Create PHP code]

+ Options

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	faculty	ref	index_for_name	index_for_name	1022	const	1	Using where; Using index

Lúc này, Mysql hiểu ra rằng chỉ có 1 hàng đúng so với câu tìm kiếm và thể hiện qua cột rows = 1, kiểu key = index_for_name được sử dụng và chiều dài key_len là 1022. Việc tìm 1 hàng tất nhiên sẽ tốt và nhanh hơn nhiều so với tìm 1830 hàng.

12. Phát triển app

Đầu tiên, người dùng sẽ thấy được trang Homepage của webapp. Hệ thống chia làm bốn loại user : Sinh viên, Giảng viên, Khoa Quản Lí và Phòng đào tạo.

Người dùng sẽ tiến hành nhấn chọn quyền mà mình sở hữu.

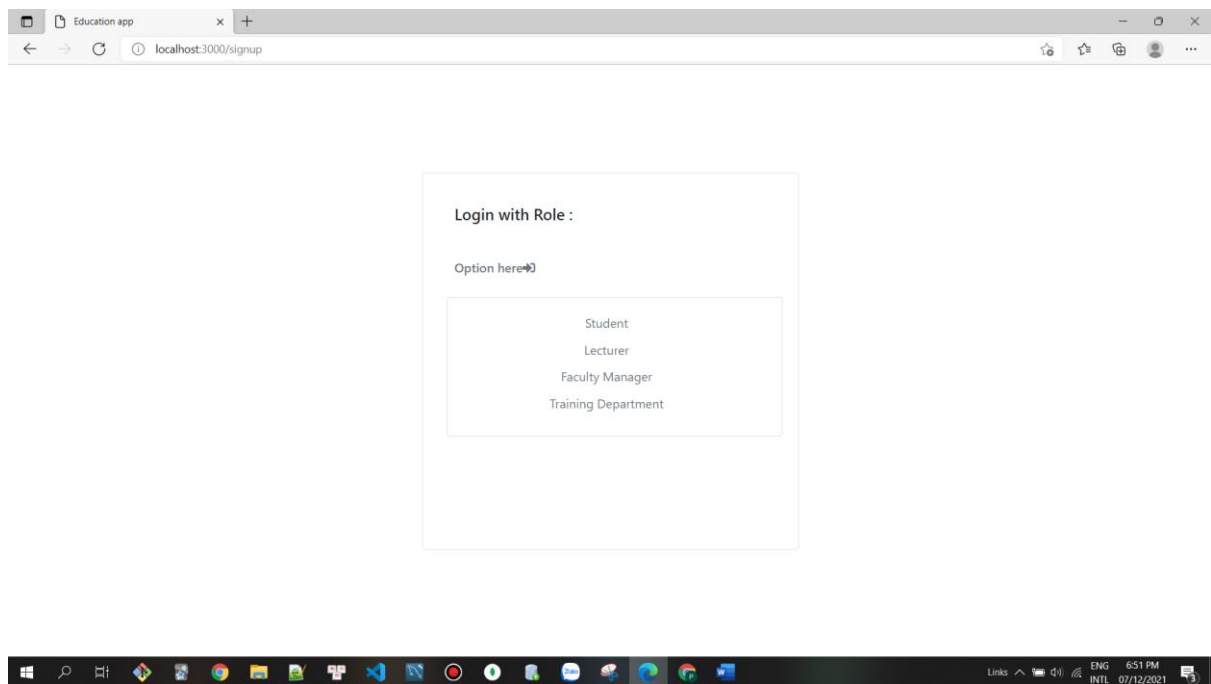


Figure 1: Phân quyền hệ thống

Sau khi chọn quyền, trang web sẽ hiển thị ra trang đăng nhập. Người dùng tiến hành nhập ID của mình để được xác thực vào hệ thống.

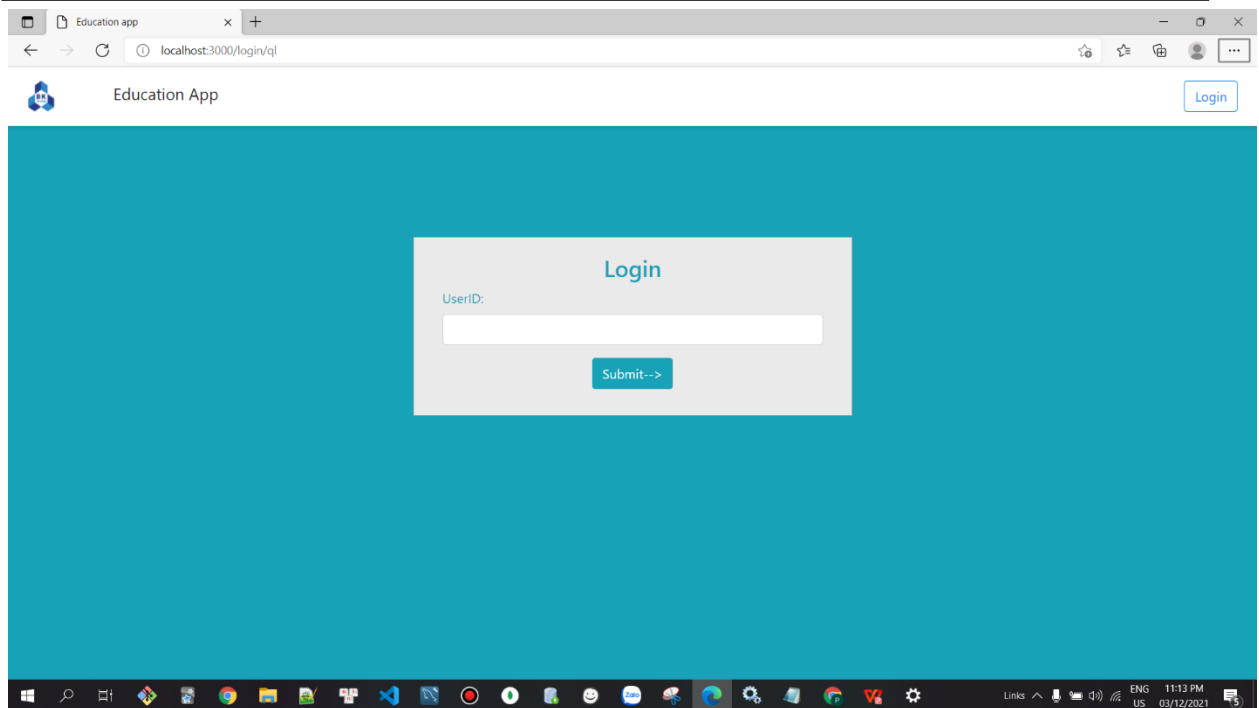


Figure 2: Giao diện Login cho người dùng

Khoa Quản Lý

Sau khi hệ thống xác thực, trang web sẽ hiển thị ra giao diện người dùng bao gồm: tên và các chức năng phù hợp với quyền của người dùng đó.

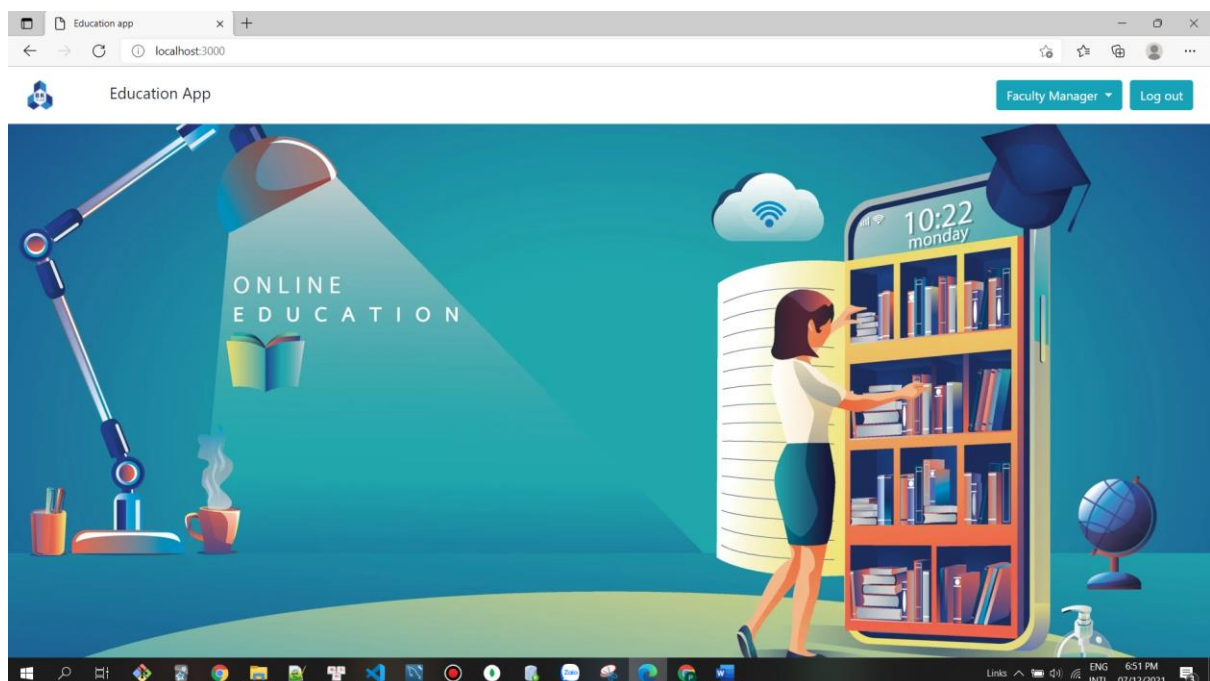


Figure 3 : Giao diện và các chức năng của Khoa quản lý

Người dùng chọn chức năng Quản lý khóa học, trang web sẽ hiển thị ra giao diện gồm có các khóa học hiện có trong mỗi học kì. Người quản lý có thể chọn học kì để xem được danh sách các môn học ở các học kì khác nhau.



Figure 4 : Người quản lý chọn học kì và xem các khóa học trong học kì tương ứng

Ngoài ra người quản lý còn có thể chỉnh sửa hoặc xóa đi khóa học hiện có

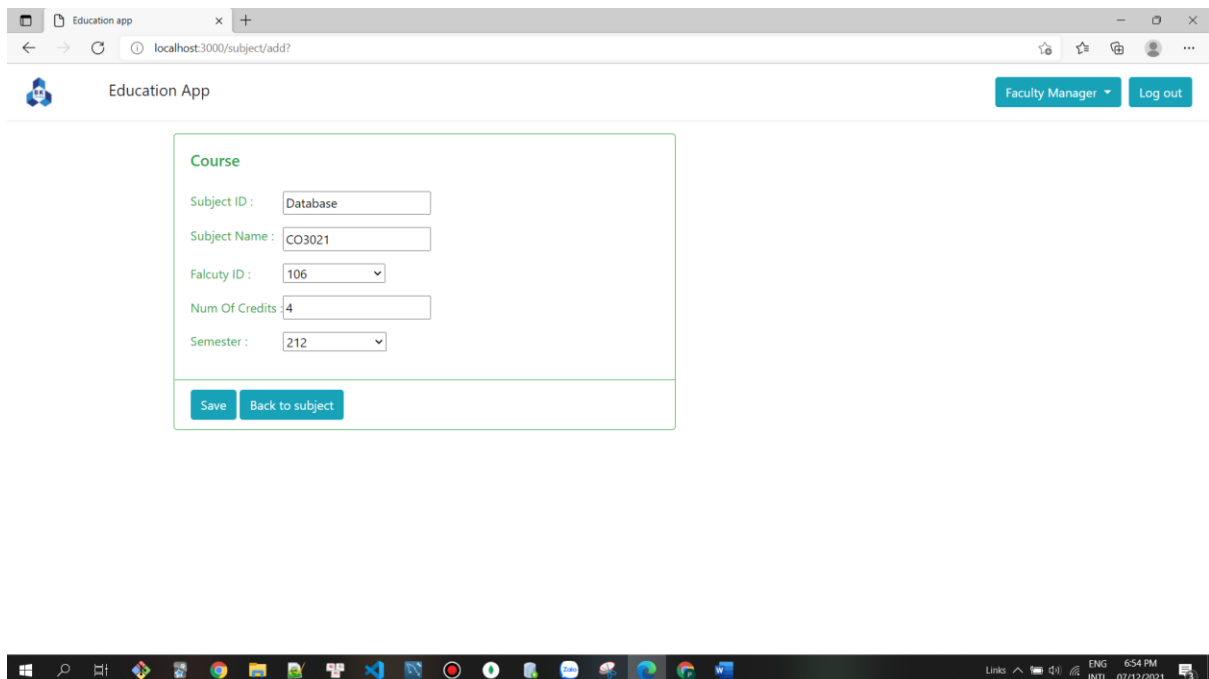


Figure 5 : Người quản lý thực hiện thêm sửa, xóa khóa học

Giao diện Sinh viên:

Sau khi sinh viên được xác thực vào hệ thống, trang web sẽ hiển thị tên và danh sách các chức năng của người dùng

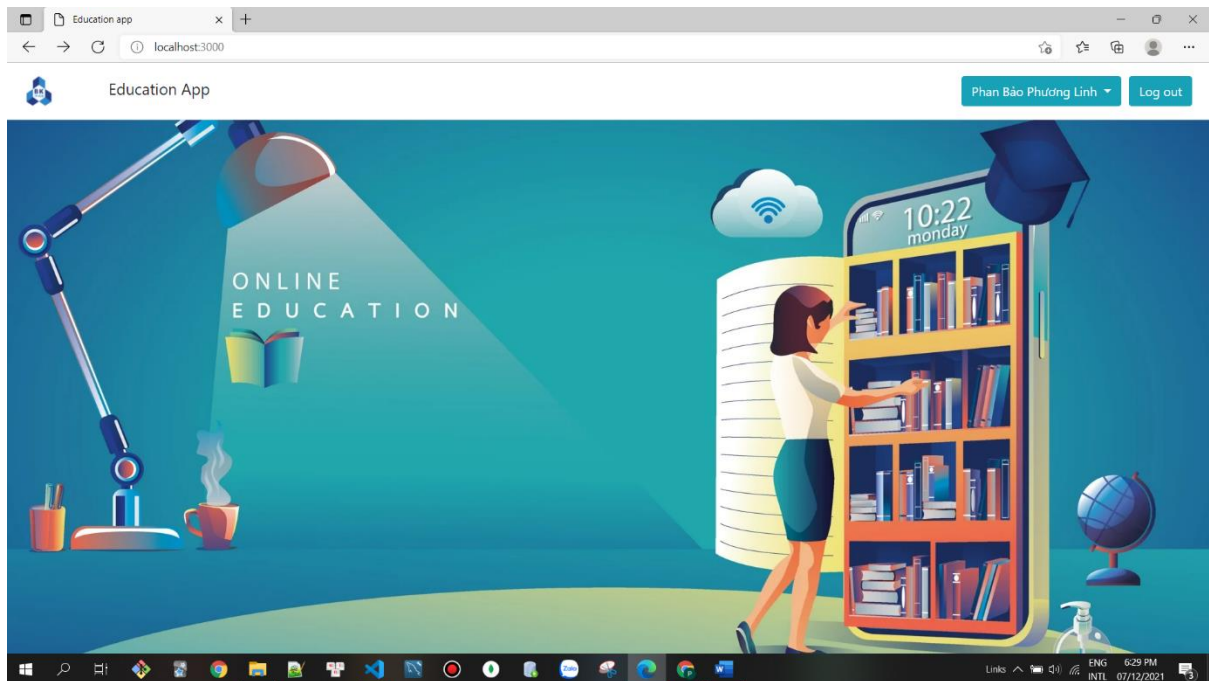


Figure 6 : Giao diện trang chủ của sinh viên

Người dùng tiến hành chọn xem danh sách các khóa học hiện có của mình trong học kì hiện tại

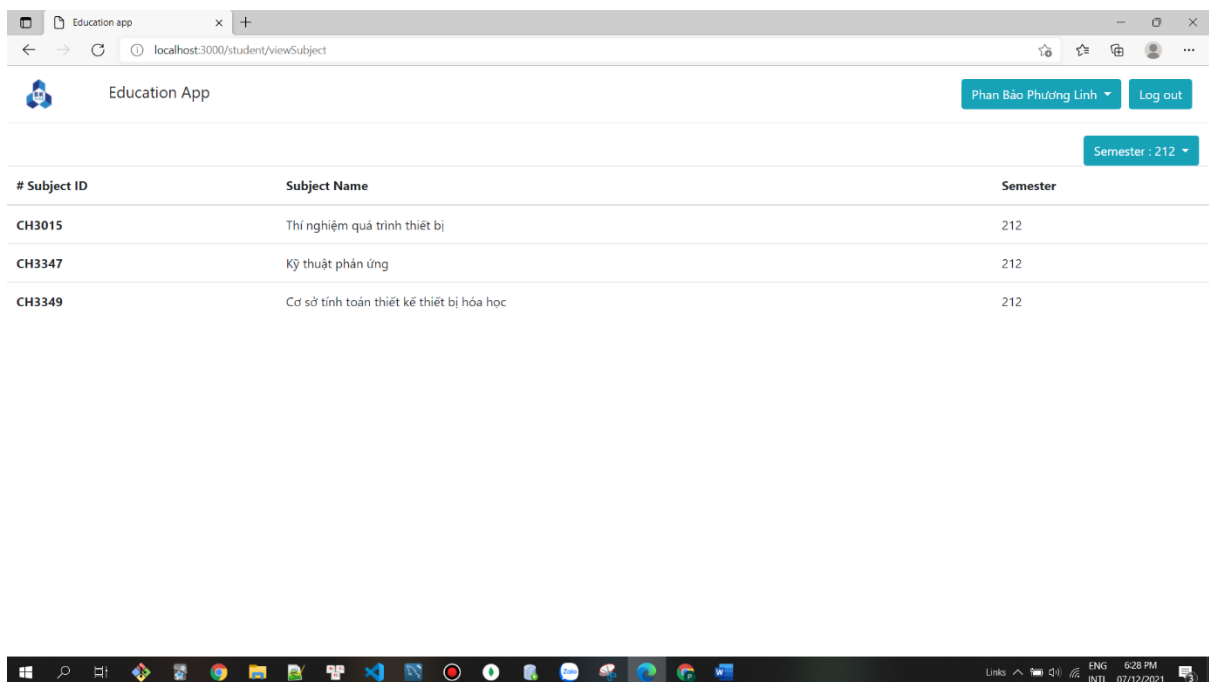


Figure 7 : Sinh viên xem khóa học đã đăng kí của mình

Ngoài ra người dùng còn có thể đăng ký khóa học mới bằng cách chọn option Register New Course, trang web sẽ chuyển đến trang đăng ký. Người dùng chọn khóa học đã mở phù hợp với mình để đăng ký.

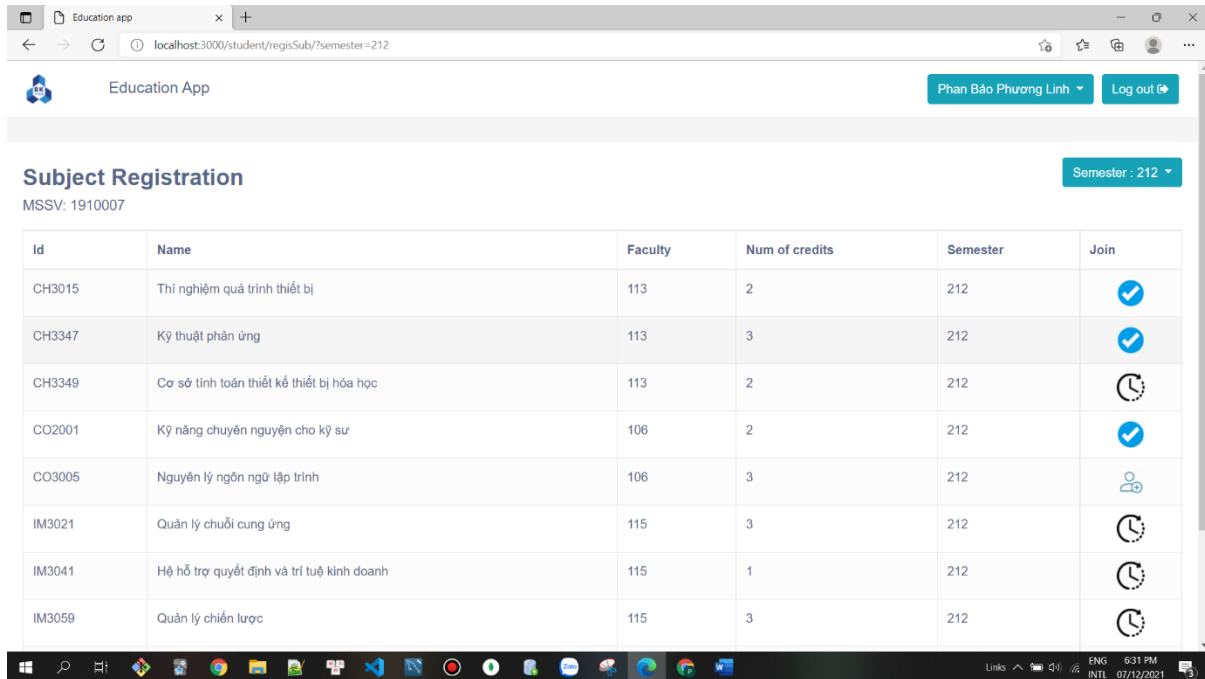


Figure 8 : Sinh viên chọn lớp thích hợp để đăng kí

13.Source code:

https://github.com/PhuMinh08082001/EducationSystem_Database