INTERNATIONAL UNIVERSITY
VIETNAM NATIONAL UNIVERSITY, HCM CITY
📖

School of Computer Science & Engineering

FINAL REPORT
Topic: College Data Management

Advisor: Assoc. Prof. Dr. Nguyễn Thị Thúy Loan
Course: Principle of Database Management

Members	Student ID	Contribution
Lê Huỳnh Nhã Nguyên	ITDSIU21058	20%
Đặng Quốc Anh Duy	ITDSIU20015	20%
Phan Bảo Trân	ITDSIU21125	20%
Bùi Gia Phú	ITDSIU21107	20%
Nguyễn Đức Nguyên Phúc	ITDSIU21108	20%

Table of Contents

Table of Figures

CHAPTER 1: INTRODUCTION

1.1   Topic Overview

In the actual world of big data applications, the college management system is an intriguing topic that sparks lots of interest. College management data refers to the collection, analysis, and interpretation of data related to the operation and administration of a college or university. Generally, this database has a complex organizational structure with multiple departments and courses in which students and instructors play crucial roles. Some of the data that the college management system covers are student enrollment and performance, faculty and staff demographics, and subject registration management. The management system is expected to be optimized manner, allowing even non-technical students to use the app without any problems.

An effective college management system is the ideal solution to improve the institution's operations and enhance the student experience. It can also help colleges stay competitive in a rapidly changing higher education landscape.

1.2   Project Background

The project background of a college management data system typically involves the need to automate and streamline various academic and administrative processes of a college or university. In the past, colleges and universities used to rely on manual processes to manage tasks such as admissions, registration, attendance tracking, examination management, and student record-keeping. However, the complexity of these data makes it challenging to collect, organize, and analyze efficiently. As a result, these manual processes were often time-consuming, error-prone, and inefficient.

To overcome these challenges, colleges and universities started implementing computerized systems to manage various tasks more effectively. In this brand-new system, they allow users to store, and manage the data easily in the same location (DBMS). An example for this system can be derived from "eduSoft", which is a web-based service to communicate between students and staff within the university with user-friendly interface.

1.3   Project Deliverables and Objectives

This project focuses on developing a database management system (DBMS) that can efficiently manage college information, including its departments, majors, courses, sections, instructors, and student enrollments. The goal of this project is to create a reliable and user-friendly system that can save time, resources, and improve the overall management of a college. Moreover, the system can be a useful tool to enhance academic quality, improve communication between staff and students, and promote insights for the Academic Affair to improve the teaching methods.

Overall, the primary goal of a college management system is to provide an integrated and efficient platform for managing the various academic and administrative processes of an institution, with the ultimate aim of improving institutional performance and enhancing the educational experience for students.

CHAPTER 2: PROJECT APPROACH

2.1   Features

- The features of the college management data are included in:
- The system provides students' information on their enrollment course like course name, date of the section, and learning room in detail so that students can search for their learning path easily
- The college management application allows instructors to manage their student by keeping track of students' performance with a user-friendly platform that are suitable for all kinds of users
- This app offers students to view their self-information, and update their latest information
- The details of courses are provided in advance for students to choose and arrange their schedule appropriately.
- The system allows for the management of staff records, including personal information and salary details.
- The department can manage the course information including course schedules, syllabus, assignments, and grading criteria.
- The system is also a useful tool to generate reports on different kinds of aspects like financial management, staff performance, analysis of student grading.

2.2   System's Workflow

- Students apply for admission by providing personal details, academic records, and other necessary information.

- IT department provides academic learning accounts (Blackboard, EduSoft) and the ID number for each student after finishing the application process

- Student entered the learning system and log in to get access to the account

- Students could view the syllabus, planning subjects to plan for their learning schedule

- Students register the course by adding courses into their account and do the authentication

- After registering for courses, the IT department summarizes the enrollment information by generating reports and analytics on student assessments, and other aspects.

- School has responsibility for the staff management, communication with students and staff through announcements, notifications, and messaging.

2.3   Development Tools

- These tools and resources will be used for developing the college management system:
  - Relational database management system (RDBMS) and Structured Query Language (SQL).
  - Implementation app by Java Language. (NetBeans)
  - Generating data by Python in Google Colab
  - Draw the entity diagram with draw.io
- Solution for the project: The college management system will be developed by using Java app that is linked to the database in SQL Server to query all the information related to the college, instructors, students, departments, courses, enrollment. The product of this project is a query form in Java.

2.4   Project Timeline

Time	Tasks
10/2 – 24/2	- Do research on the chosen topic - Identify goals and tools for the project - List out all of requirements - Write the proposal - Join meeting
25/2 – 3/3	- Define necessary tables and figure out all of attributes for each table - Choose the appropriate data types - Generating datasets with Python - Join meeting
4/3 – 10/3	- Define keys (Primary/ Foreign, etc.) for each table - Discuss on the connection of these tables - Do research on entity relationship diagram (ERD) - Join meeting
11/3 – 17/3	- Do the first draft of ERD - Gather all information of finished task - Prepare mid-term report

18/3 – 24/3	• Mid-term
25/3 – 31/3	• Complete the raw ERD • Learn how to link the database with Java application system • Join meeting
1/4 – 7/4	• Convert ERD to the Relational Model • Test the app (ver1: members) and fix bugs if necessary • Optimize the app
8/4 – 14/4	• Edit the ERD and Relational Model • Test the app (ver1: members) and fix bugs if necessary • Generate questions scenarios for the report • Update the database • Write the final report
15/4 – 23/4	• Keep on optimizing the app • Test the app (ver2 : outsiders) : receive feedback from real user and fix bugs if necessary • Final check all documents and applications • Prepare slide for presentation • Submit the final report

CHAPTER 3: ENTITY - RELATIONSHIP DIAGRAM (ERD)

3.1   Cardinality

- One instructor can teach one or more sections, while one section can only be taught by only one instructor. (**Instructor ♦ Section**)

- One instructor can only work in one department whereas one department can contain one or more instructors (**Instructor ♦ Department**)

- One department can only be managed by one instructor (head of Department), while one or no instructor can manage the department. (**Instructor ♦ Department**)

- One or no instructor can manage the department, while be managed by only one instructor(head of Department). (**Instructor ♦ Department**)

- One major can belong to only one Department, while one department can contain one or more majors. (**Major ♦ Department**)

- One major can have one or more students, while the student can beyond only one major. (**Major ♦ Student**)

- One student can enroll in one or more courses, while one course can be enrolled by one or more students (**Student ♦ Course**)

- It is necessary to register for prerequisite subjects before applying for the following subjects.

- One course can have one or many prerequisite subjects, while one prerequisite subject can have many following courses. (**Course ♦ Course**)

- One course can have one or more sections, while one section is beyond only one course.

  (**Course ♦ Section**)

- One student has one or more dependents, while one dependent can only have one student

  (**Student ♦ Dependent**)

3.2   Database Design with explanatory

3.2.1   Student

In the college management system, students are one of the key entities since they are involved in the education process to study as learners. This entity contains all information about the student's identity including name, date of birth, gender, and their email. The system allows students to view and modify their information in real-time if necessary, so students can get access to their information and do the query. The below table illustrates the details of all attributes in the student entity:

Table 1: Student

Index	Field	Description	Data Type	Size
1	sId (PK)	Student ID number	char	11
2	Fname	Student first name	varchar	20
3	Lname	Student last name	varchar	20
4	Dob	Student's date of birth	date	
5	Gender	Student gender	char	1
6	Email	Student email address	varchar	30

3.2.2 Instructor

Instructor is an entity that represents a teacher or professor who is responsible for delivering lectures and conducting classes for students. Instructors are an essential part of the college management system as they play a crucial role in the academic success of the students. In overall, the instructor entity is an integral part of the college management system, and the accurate and efficient management of this entity is critical to the success of the institution. This entity also provides the users with the ability to view and update their information over time. The details of all attributes are shown below:

Table 2: Instructor

Index	Field	Description	Data Type	Size
1	Ins_ID	Instructor ID number	varchar	10
2	Ins_name	Instructor Full Name	varchar	20
3	Email	Instructor Email address	varchar	30
4	Dob	Instructor's date of birth	date	
5	Salary	Instructor salary number	int	

3.2.3 Department

The department entity represents an academic unit responsible for organizing and delivering a specific field of study or discipline. Each department may offer several majors or specializations within the field of study. In this entity, department consists of department ID as the abbreviated name, the full name of department, and the year of establishment.

Table 3: Department

Index	Field	Description	Data Type	Size
1	Dept_ID (PK)	Department ID abbreviation	char	2
2	Dept_name	Department full name	varchar	50
3	Year	Year of Established	int	4

3.2.4 Major

A major is an academic program of study that represents a specific field or discipline. Each major has a set of required courses that students must complete to graduate with a degree in that field. The major entity plays a crucial role in the college management system as it serves as the basis for academic advising, course scheduling, and degree program management. The efficient management of the major entity is essential for ensuring the quality and integrity of the academic programs offered by the institution, and for helping students achieve their academic and career goals. Following are the details of attributes in major table:

Table 4: Major

Index	Field	Description	Data Type	Size
1	Major_ID	Major ID abbreviation	char	2
2	Major_name	Major full name	varchar	50

3.2.5   Course

Course entity represents a specific academic course offered by the institution. Each course has a unique course code, a course name, and a description of the course content and objectives. The course entity contains course ID, course name, and the number of credits for that course. This information is used by the college management system to schedule classes, assign courses to faculty members, and manage course registration. The course entity plays a critical role in the college management system as it serves as the foundation for academic programs and degree requirements. The table below shows the details of attributes in course entity:

Table 5: Course

Index	Field	Description	Data Type	Size
1	Course_ID	Course ID string	char	7
2	Course_Name	Name of the course	varchar	50
3	Credits	Number of credits	int	1

3.2.6   Section

A section entity represents a detail of classes that students enrolled in with related information including type of the section and the room number for learning. This entity was created to manage the course since there are two types of sections within the course (lab/theory). Furthermore, this relation can be considered as the schedule for students, so students can keep track of their study without complications.

Table 6: Section

Index	Field	Description	Data Type	Size
1	SectionID	Section ID number	char	7
2	Section_Types	Types of section (lab/theory)	varchar	50
3	Room	Room number	varchar	10

CHAPTER 4: CONVERT FROM ERD TO RELATIONAL MODEL

4.1 Original Diagram

For the college management system, the entity relational diagram is one of the most important parts of building the system. The below figure is the diagram that describes the whole process of operating the college. In the previous part, the ERD has been fully explained with the cardinality of all entities, and the details attributes of each relation. After modifying the ERD several times, some features that our system should be focused on to convert into the relational model are the multivalued of email attribute in the instructor table, and the composite data in students' name.
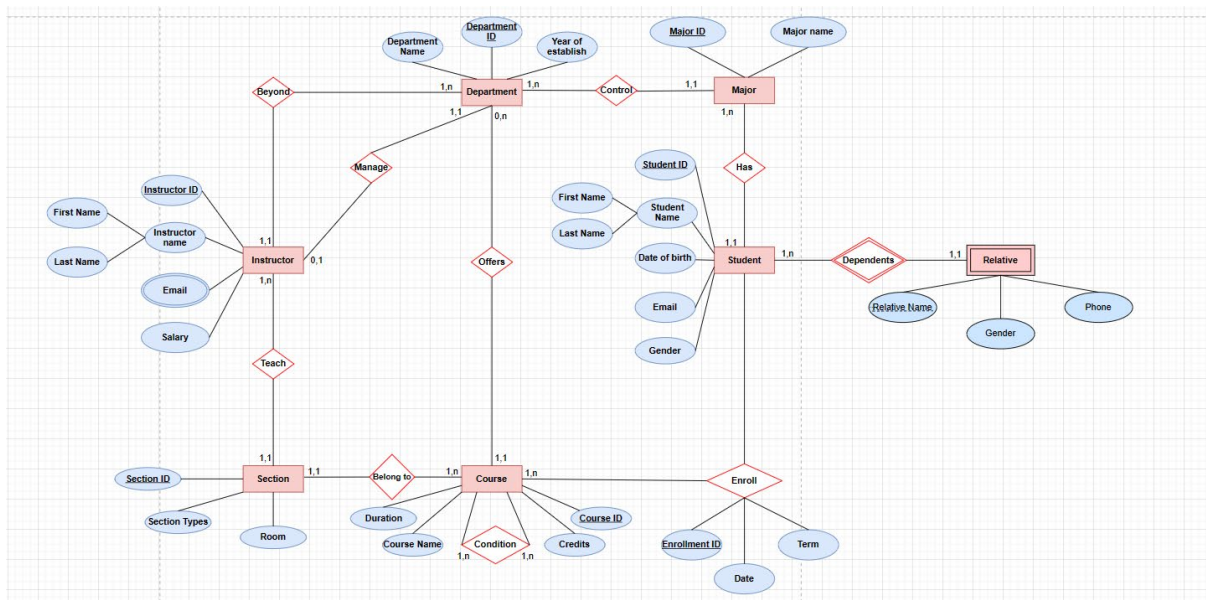

Figure 1: Entity Relationship Diagram for the College Management

4.2   Translational Process

Step 1: Identify normal entities

There are 6 strong entities in total, including Department, Major, Student, Instructor, Course, and Section.

Step 2: Figure out weak entities

One Dependent has only one student, while one student has one or more dependents. So, the dependent is a weak entity type. As a result, adding the primary key of the student table into the relative relation as the foreign key.

→ Dependent(*student _id*, _dependent_id, dp_name, phone)
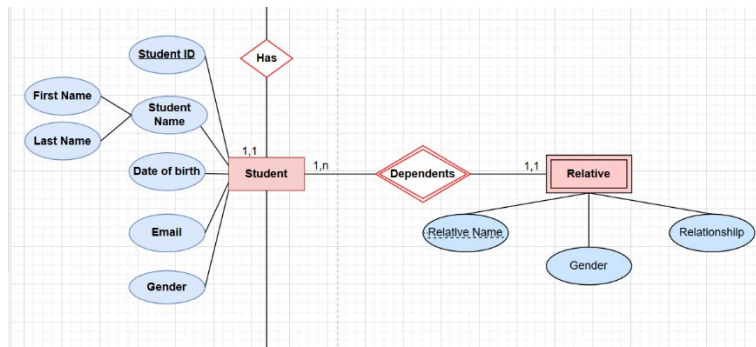
Step 3: Mapping binary 1 – to – 1
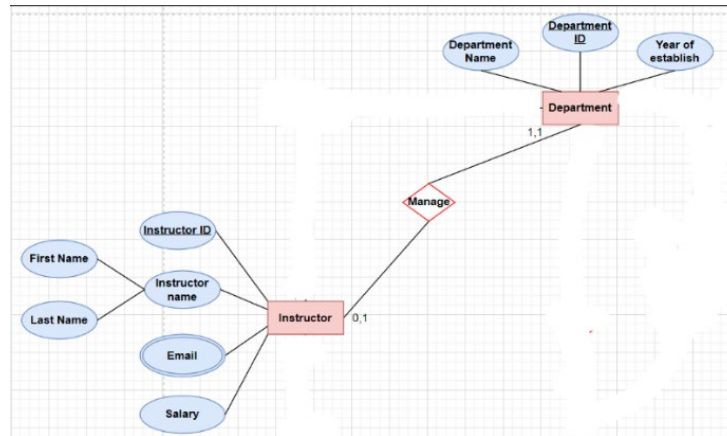
- One or no instructor only manages one department, while the department is only managed by one instructor. So, the instructor and the department are mapping binary 1 to 1.
- ➔ Adding the primary key of the instructor into the department as the foreign key.

Department (Dept_ID, Dept_name, Established_Year, *Ins_Manager_ID*)

- One instructor only belongs to one department (1,1)

➔ Add Dept_ID as a foreign key from the Department table.

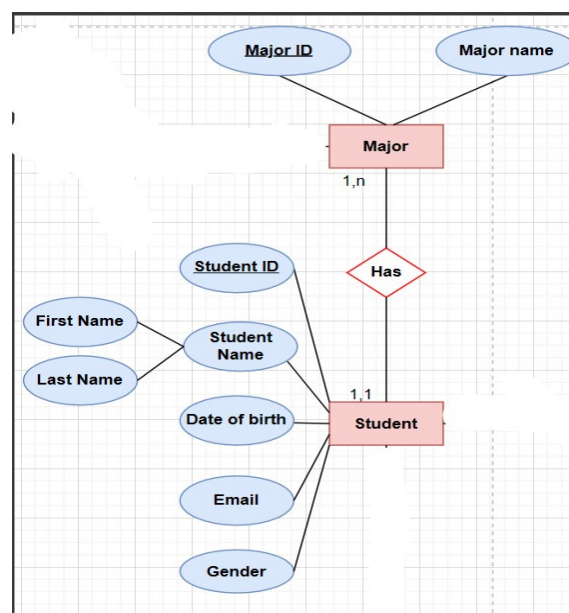Instructor(Ins_ID, Ins_name, Dob, **Dept_ID**)

Step 4: Mapping binary 1 – to – N

- One major has one or more students, while one student belongs to a major. So, the major and the student are mapping binary 1 to many.

➔ Adding the primary key of the major into the student becomes the foreign key.

Student (Student_ID, First_name, Last_name, Dob, Gender, Email, *Major_ID*)

- One department can control one or more majors (1,n), but one major is only controlled by one department. So, the department and the major are mapping 1 to many.

    o **Major**(Major_ID, Major_name, *Dept_ID*)
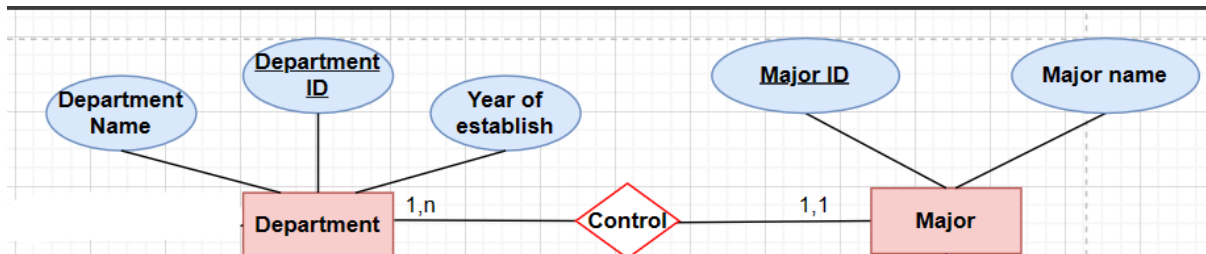
- One Instructor teaches one or more sections (1,n), while one section can only be taught by one Instructor. So, the instructor and section relationship are mapping binary 1 to many.

➔ Adding the primary key of the department into the instructor becomes the foreign key.

    o **Instructor**(Ins_ID, Ins_name, Dob, *Dept_ID)*
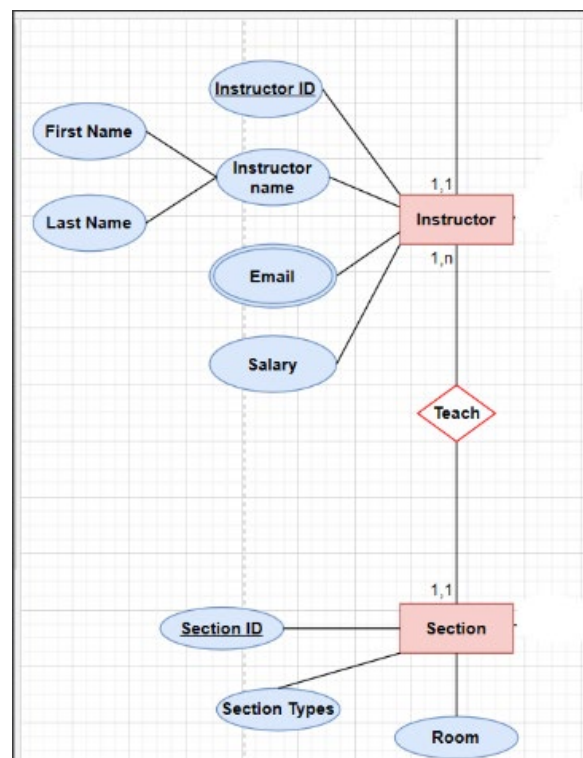
- One section beyond only one course, one course has one or more sections. So, the course and section are mapping binary 1 to many.

➔ Adding the primary key of the course into the section becomes the foreign key.

- o **Section** (SectionID, Section Types, Room, *Course ID, Instructor ID)*
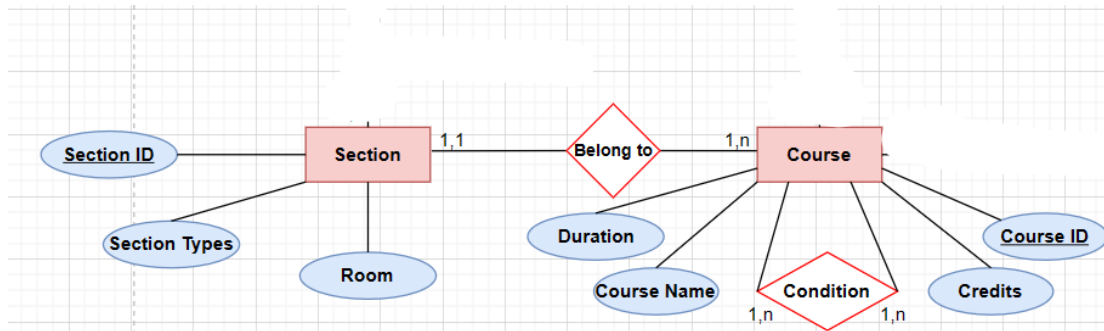
Step 5: Mapping binary M – to – N

- One course has one or more prerequisite courses, while one prerequisite course has one or more courses.

➔ Create the Condition table with the primary key as the primary key of Course table.

- o Condition (Prerequisite Course ID, Course ID)
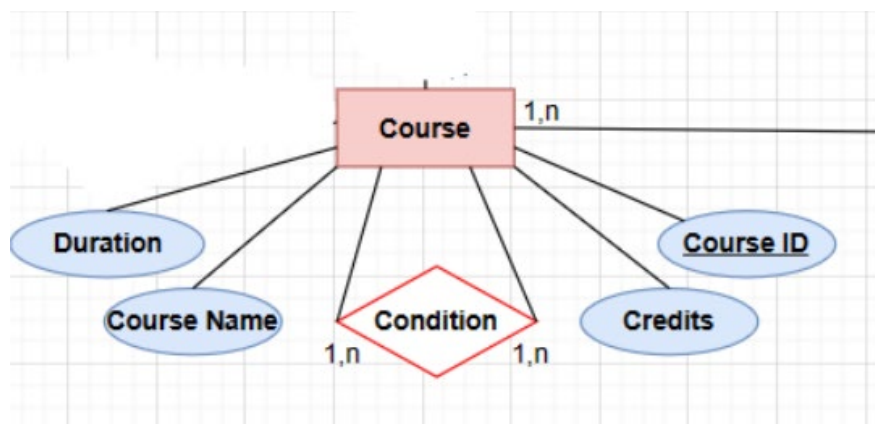
- One or more students can enroll in more courses. One course is enrolled by Students. So, the student and the course are mapping binary many to many

➔ Create the Enrollment table with the primary key as the primary key of the Student and Course table.

  o Enrollment (*Student_ID, Course_ID*, Enrollment_ID, Date, Term)
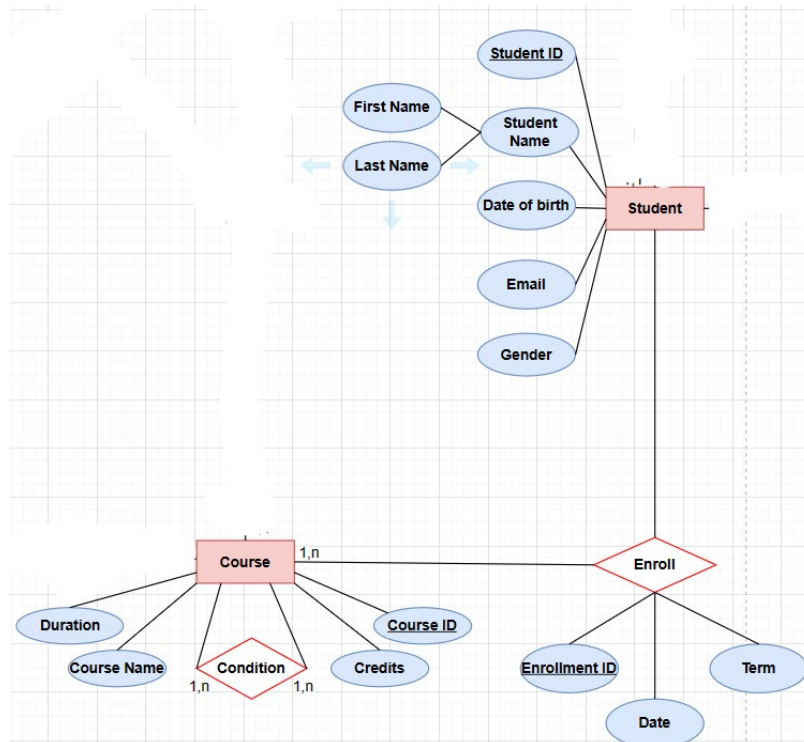
Step 6: Multivalued attributes

- One instructor has more emails

➔ Create the instructor_email with the primary key is Ins_ID, and Email.

  o Instructor_Email(Ins_ID,Email)
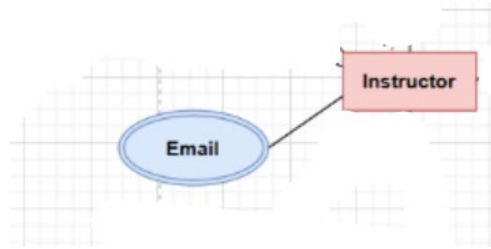
Step 7: Specialization / Generalization (skip)

4.3   Relational Model

1. **Department** (Dept_ID, Dept_name, Year, Ins_Manager_ID)

2. **Major** (Major_ID, Major_name, Dept_ID)

3. **Student** (Student_ID, First_name, Last_name, Date_of_birth, Gender, Email, Major_ID)

4. **Instructor** (Ins_ID, Ins_name, Date_of_birth, Dept_ID)

5. **Instructor_Email**

6. **Enrollment** (Student_ID, Course_ID, Enrollment_ID, Date, Term)

7. **Course** (Course ID, Course Name, Credits, Room, Department ID)

8. **Section** (SectionID, Section Types, Room, Course ID, Instructor ID)

9. **Condition** (Pre_Course_ID, Course_ID)

10. **Relative (**Student_ID, Relative_name, Relative_phone, Gender)

	Dept_ID	Dept_Name	Established_Year	Ins_Manager_ID
1	BA	School of Business	2003	INS21
2	BT	School of Biotechnology	2017	INS6
3	CE	School of Civil Engineering and Management	2021	INS5
4	EE	School of Electrical Engineering	2004	INS26
5	EN	School of Languages	2017	INS7
6	IE	School of Industrial Engineering and Management	2009	INS52
7	IT	School of Computer Science and Engineering	2004	INS1
8	MA	Department of Mathematics	2013	INS4
9	PH	Department of Physics	2016	INS17

Table 7: Department Data

	Major_ID	Major_name	Dept_ID
1	AC	Control Engineering and Automation	EE
2	AE	Civil Engineering	CE
3	AR	Aquatic Resource Management	BT
4	BA	Business Management	BA
5	BT	BioTechnology	BT
6	CE	Computer Engineering	IT
7	CM	Construction Management	CE
8	CS	Computer Science	IT
9	DS	Data Science	IT
10	EE	Electronic and Telecommunication	EE
11	EN	Arts in English Linguistics and Literature	EN
12	FT	Food Technology	BT
13	HM	Hospitality Management	BA
14	IB	International Business Management	BA
15	IE	Industrial System and Engineering	IE
16	LS	Logistics and Supply Chain Management	IE
17	MA	Applied Mathematics	MA

Table 8: Major Data

	Course_ID	Course_Name	Credits	Room	Dept_ID
1	BA003IU	Principles of Marketing	3	A1.201	BA
2	BA005IU	Financial Accounting	3	A2.203	BA
3	BA006IU	Business Communication	2	A2.202	BA
4	BA016IU	Fundamental of Financial Management	2	A1.205	BA
5	BA018IU	Quality Management	1	A2.207	BA
6	BA020IU	Business Ethics	3	A1.307	BA
7	BT155IU	General Biology	4	LA2.103	BT
8	BT164IU	Microbiology	2	A1.308	BT
9	BT207IU	Human Pharmacology	3	A2.107	BT
10	BT210IU	Human physiology	3	A2.206	BT
11	CH009IU	Organic chemistry	3	A2.307	BT
12	CH101IU	General Chemistry	4	A1.608	BT
13	EE010IU	Electromagnetic Theory	1	A2.606	EE
14	EE049IU	Introduction to EE	3	A1.202	EE
15	EE051IU	Principles of EE	2	A1.402	EE
16	EE097IU	Thesis	1	A1.507	EE
17	EL001IU	Reading 1 B2-C1	3	A2.210	EN

Table 9: Course Data

	Ins_ID	Ins_name	Dob	Dept_ID
1	INS1	Jacob Daniels	1966-09-25	IT
2	INS10	Dale Torres	1981-12-11	EN
3	INS100	Travis Rivera	1991-04-21	EE
4	INS11	Alexis Valencia	1973-06-23	EN
5	INS12	Tonya Williamson	1973-08-09	EN
6	INS13	Samantha Stein	1987-04-20	EN
7	INS14	Devin Ellis	1986-01-27	EN
8	INS15	Brandy Brooks	1993-07-26	EN
9	INS16	Linda Nelson	1974-08-01	EE
10	INS17	Melanie Kaufman	1987-10-13	PH

Table 10: Instructor Data

	Ins_ID	Email
1	INS1	patrickgarcia@example.com
2	INS10	james39@example.net
3	INS17	qhendricks@example.org
4	INS21	michael66@example.org
5	INS22	williamsonbrandi@example.com
6	INS24	mli@example.com
7	INS25	deleonthomas@example.net
8	INS26	jgarcia@example.net
9	INS28	traciewalters@example.org
10	INS29	williamsaustin@example.org

Table 11: Instructor Email

	Student_ID	Course_ID	Enrollment_ID	Term	Registration_Date
1	BABAIU20412	IT069IU	39	1	2021-09-03
2	BABAIU20412	MA023IU	116	1	2021-09-04
3	BABAIU20412	MA023IU	148	2	2022-01-04
4	BAHMIU22192	BA003IU	31	2	2022-01-03
5	BAHMIU22192	BA006IU	34	2	2022-01-02
6	BAHMIU22192	IT069IU	41	2	2022-01-01
7	BAHMIU22192	MA023IU	104	1	2021-09-02
8	BAIBIU19142	BA005IU	32	2	2022-01-05
9	BAIBIU19142	BA006IU	29	2	2022-01-05
10	BAIBIU19142	IT069IU	43	1	2021-09-03

Table 12: Enrollment Data

	Student_ID	Relative_name	Relative_phone	Gender
1	EEACIU19114	Amber	1859737253754	M
2	BTARIU22745	Barbara	122945110104193	F
3	CECEIU22135	Carol	13949338621089	F
4	CECMIU21108	Christina	11355754636106	F
5	ITNEIU21162	Darren	11355754636106	F
6	BTARIU21124	Gregory	14172871063246	M
7	ITCSIU19513	Jerry	1236416148826	M
8	EEEEIU22135	Katelyn	14453710256294	M
9	ENENIU21138	Martin	17766109974239	M
10	EEEEIU22305	May	1931021106231088	M

Table 13: Relative Data

	Prerequisite_Course_ID	Course_ID
1	BA003IU	BA005IU
2	BA003IU	BA006IU
3	BA003IU	BA016IU
4	BA003IU	BA020IU
5	BA018IU	BA005IU
6	BT210IU	BT164IU
7	BT210IU	BT207IU
8	BT210IU	CH009IU
9	EE010IU	EE049IU
10	EE010IU	EE051IU

Table 14: Condition Data

	Section_ID	Section_Types	Room	Course_ID	Ins_ID
1	1000	Lab	LA1.101	BA003IU	INS46
2	1001	Theory	A1.101	BA005IU	INS46
3	1002	Lab	LA1.102	BA006IU	INS48
4	1003	Theory	A1.102	BA006IU	INS48
5	1004	Lab	A1.103	BA016IU	INS21
6	1005	Theory	A1.104	BA003IU	INS21
7	1006	Lab	LA1.201	BT155IU	INS20
8	1007	Theory	A2.201	BT155IU	INS20
9	1008	Lab	LA1.202	BT164IU	INS20
10	1009	Theory	A2.202	BT164IU	INS25

Table 15: Section Data
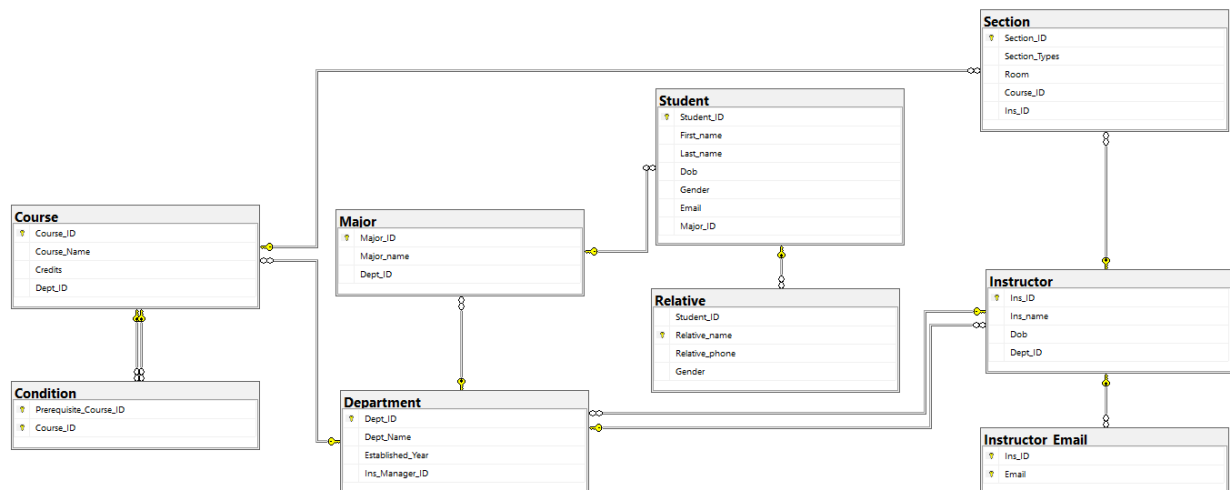
Figure 2: Relational Model (SQL Server)

Instructor_Email
Email
Instructor ID   (FK)

Department
Department ID
Department   Name
Year of Establish

Major
Major ID
Major Name
Department ID   (FK)

Instructor
Instructor ID
First Name
Last Name
Position
Salary
Department ID
Department ID   (FK)

Student
Student ID
First Name
Last Name
Date of Birth
Email
Gender
Major_ID
Major ID   (FK)

Section
Section ID
Section Type
Room
Course ID
Instructor ID
Course ID   (FK)
Instructor ID   (FK)

Course
Course ID
Credits
Course Name
Duration
Department ID
Department ID   (FK)

Enrollment
Student ID
Course ID
Enrollment ID
Term
New Column
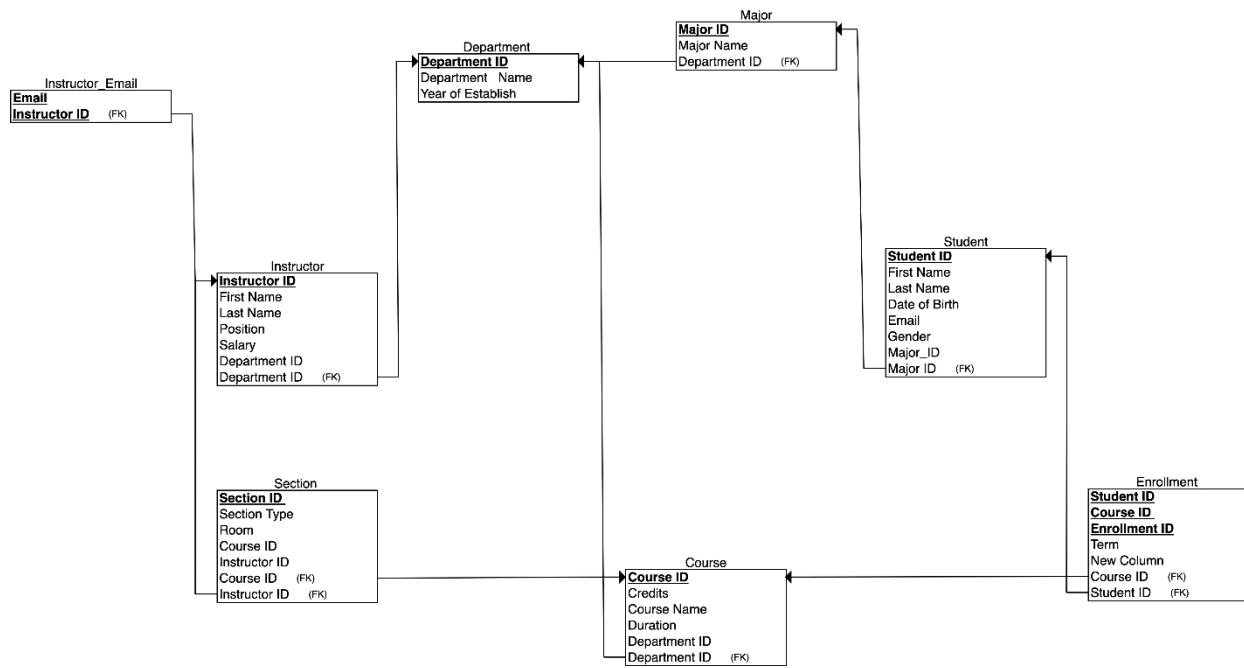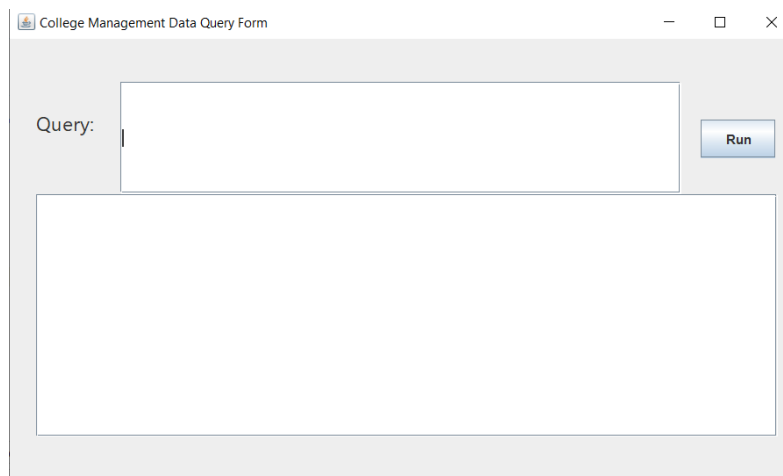Course ID   (FK)
Student ID   (FK)

Figure 3: Relational Model (ERD+)

CHAPTER 5: DEMO APPLICATION IN JAVA
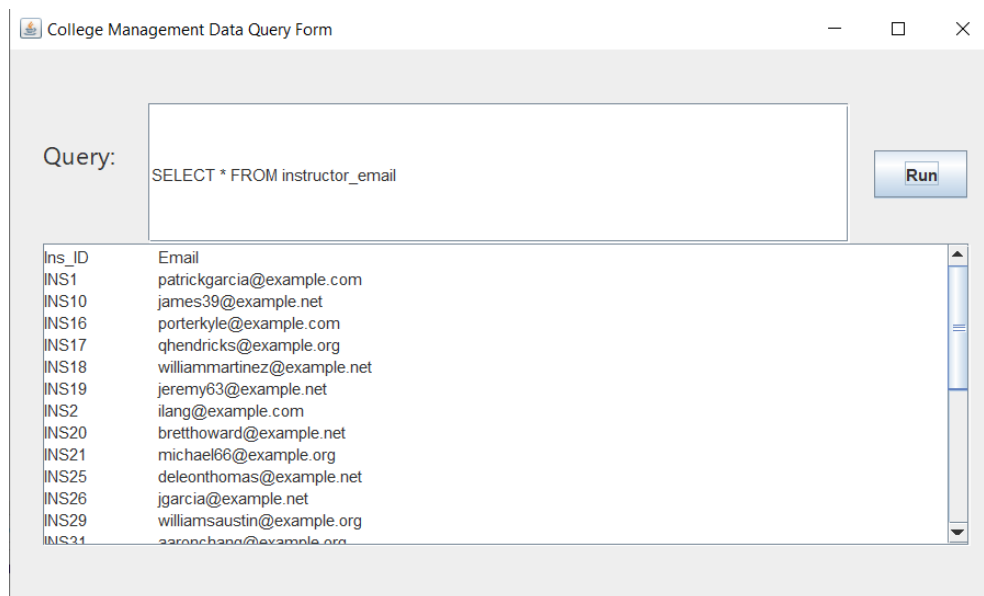
5.1   Classes and its responsibilities

No.	Class Name	Responsibility
1	CollegeManagement	Call frmMain class
2	frmMain	Connect the program to the SQL Server

5.2   Query form

- **Example of querying form:**

CHAPTER 6: REAL WORLD SCENARIOS

6.1 Question 1

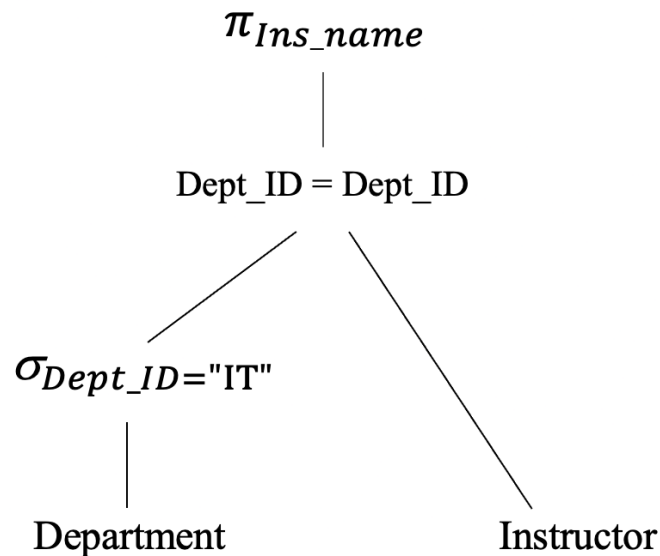- Find name of the instructor who is the manager of the department with Dept_ID = 'IT'.

  ### 1. Relational Algebra

  $$Q_1 \leftarrow \text{Department} \bowtie_{(Department.Dept\_ID=Instructor.Dept\_ID)} \text{Instructor}$$

  $$Q_2 \leftarrow \sigma_{(Dept\_ID="IT")}(Q_1)$$

  $$\text{Result} \leftarrow \pi_{Ins\_name}(Q_2)$$

  ### 2. Tree

  $$\pi_{Ins\_name}$$

  $$\text{Dept\_ID} = \text{Dept\_ID}$$

  $$\sigma_{Dept\_ID="IT"}$$

  Department   Instructor

  ### 3. SQL Query

```sql
SELECT Ins_name
FROM Instructor, Department
WHERE Instructor.Ins_ID = Department.Ins_Manager_ID AND Department.Dept_ID = 'IT'
```

6.2 Question 2

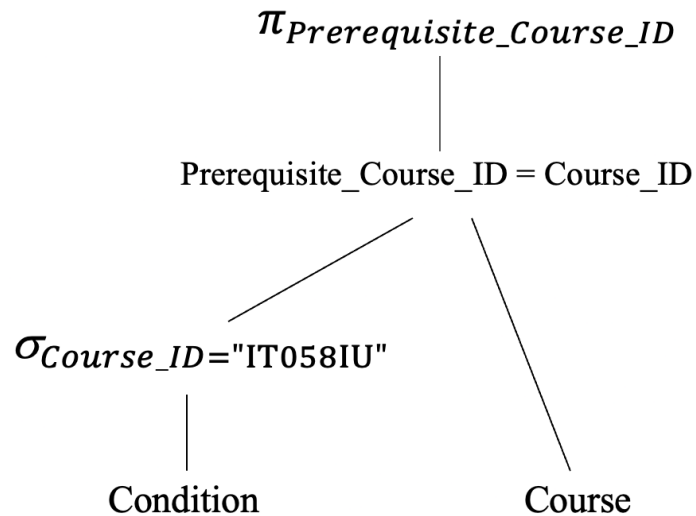- Find the prerequisite courses for the course with Course ID = 'IT058IU'.

  1. **Relational Algebra**

$$Q_1 \leftarrow \text{Condition} \bowtie_{(\text{Condition}.Prerequisite\_Course\_ID=\text{Course}.Course\_ID)} \text{Course}$$

$$Q_2 \leftarrow \sigma_{(Q1.Course\_ID="IT058IU")}(Q_1)$$

$$\text{Result} \leftarrow \pi_{Prerequisite\_Course\_ID}(Q_2)$$

  2. **Tree**

$$\pi_{Prerequisite\_Course\_ID}$$

Prerequisite_Course_ID = Course_ID

$$\sigma_{Course\_ID="IT058IU"}$$

Condition

Course

  3. **SQL Query**

```
--- Question 2:
SELECT Condition_.Prerequisite_Course_ID, Course.Course_Name 'Prerequisite_Course_Name'
FROM Course
    JOIN (SELECT *
            FROM Condition
            WHERE Condition.Course_ID= 'IT058IU') AS Condition_
    ON Course.Course_ID = Condition_.Prerequisite_Course_ID
```

6.3   Question 3

- Find First_name, Last_name of male students who registered for the course's name "Principles of Marketing".
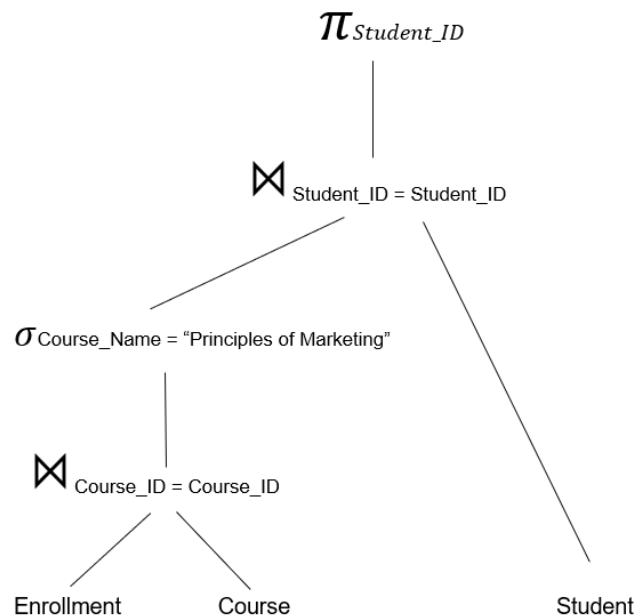
   1. **Relational Algebra**

$$Q_1 \leftarrow \text{Enrollment} \bowtie_{(\text{Enrollment}.Course\_ID=\text{Course}.Course\_ID)} \text{Course}$$

$$Q_2 \leftarrow \sigma_{(Q1.Course\_Name="Principles of Marketing")} (Q_1)$$

$$Q_3 \leftarrow \text{Student} \bowtie_{(\text{Student}.Student\_ID=Q2.Student\_ID)} Q_2$$

$$\text{Result} \leftarrow \pi_{First\_name,Last\_name}(Q_3)$$

   2. **Tree**

   3. **SQL Query**

```
--- Question 3:
select Student.First_name, Student.Last_name
from Course
    Join Enrollment on Course.Course_ID = Enrollment.Course_ID
    Join Student on Student.Student_ID = Enrollment.Enrollment_ID
where Course.Course_Name = 'Principles of Marketing' and Student.Gender= 'm'
```

6.4 Question 4

- Find Ins_ID, Ins_name who belong to IT department and teach Lab Section and their last name contains N as the second letter

1. Relational Algebra

$Q_1 \leftarrow$ Department $\bowtie_{(Department.Dept\_ID=Instructor.Dept\_ID)}$ Instructor

$Q_2 \leftarrow \sigma_{(Q1.Dept\_ID="IT")} (Q_1)$

$Q_3 \leftarrow$ Section $\bowtie_{(Section.Instructor\_ID=Q2.Instructor\_ID)} Q_2$

$Q_4 \leftarrow \sigma_{(Q3.Section\_types="Theory")} (Q_3)$

$Q_5 \leftarrow \sigma_{(Q4.Ins\_name="\_a\%")} (Q_4)$

Result $\leftarrow \pi_{Ins\_ID,Ins\_name}(Q_5)$

2. Tree

π Ins_ID, Ins_name

|

$\sigma_{Ins\_name="\_N\%"}$

|

$\sigma_{Section\_types="Lab"}$

|

$\bowtie$ Instructor_ID = Instructor_ID

$\sigma_{Dept\_ID="IT"}$

|

$\bowtie$ Dept_ID = Dept_ID

Department   Instructor   Section
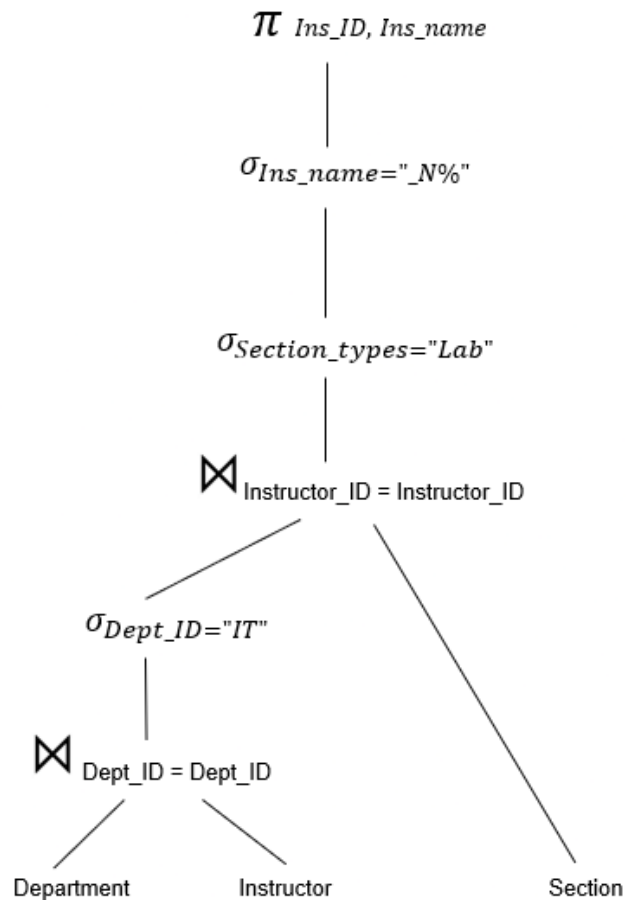
3. SQL Query

```sql
--- Question 4:
select Instructor.Ins_ID, Instructor.Ins_name
from Department
    Join Instructor on Department.Dept_ID = Instructor.Dept_ID
    Join Section on Section.Ins_ID = Instructor.Ins_ID
where Section.Section_Types = 'Lab' and Instructor.Ins_ID like '_N%'
```

6.5 Question 5

- Find Course_ID, Course_name of the Courses registered by all students

1. Relational Algebra

$Q_1 \leftarrow$ Enrollment $\bowtie_{(Enrollment.Course\_ID = Course.Course\_ID)}$ Course

$Q_2 \leftarrow \pi_{Course.Course\_ID, Course.Course\_name, Enrollment.Student\_ID}(Q_2)$

$Q_3 \ (Course\_ID, Course\_name, the\_number\_of\_student\_registered) \leftarrow$
$\qquad Course.Course\_ID, Course.Course\_name \Upsilon_{(count(Enrollment.Student\_ID))}(Q_2)$

$Q_4 \ (the\_number\_of\_student) \leftarrow \Upsilon_{(count(Student.Student\_ID))} Student$

$Q_5 \leftarrow \sigma_{(Q3.the\_number\_of\_student\_registered \, = \, Q4.the\_number\_of\_student)}(Q_4)$

Results $\leftarrow \pi_{Q3.Course\_ID, Q3.Course\_name}(Q_5)$

2. SQL Query

```sql
--- Question 5:
select Enrollment.Course_ID
from Enrollment
Group by Course_ID
Having COUNT(distinct(Student_ID)) = (select COUNT(distinct(Student_ID))
                                      from Student)
```

3. Tree

5

$$\pi_{Course\_ID,Course\_name}$$

$$\sigma_{\text{the number of student registered} = \text{the number of student}}$$

$$\rho_{(Course\_ID,Course\_name,the\ number\ of\ student\ registered)}$$

$$\rho_{(the\ number\ of\ student)}$$

$$_{Course\_ID,Course\_name,the\_number\_of\_student\_registered}\gamma_{Count(student\_ID)}$$

$$\gamma_{Count(student\_ID)}$$

$$\pi_{Course\_ID,Course\_name,Student\_ID}$$

$$\pi_{Student\_ID}$$

$$\bowtie_{Course\_ID=Course\_ID}$$

Enrollment   Course   Student
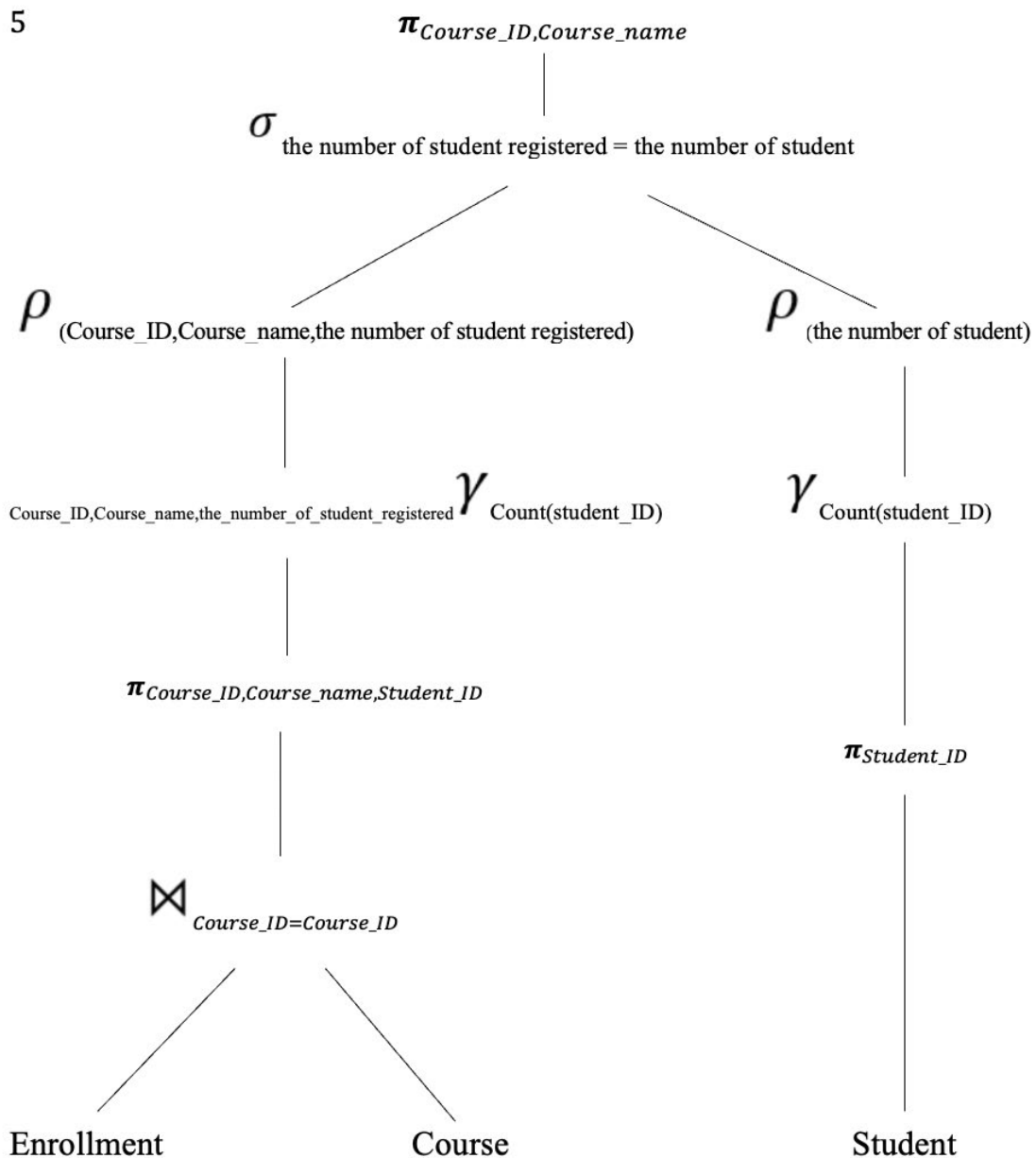
6.6   Create View

- **Create view to extract the most registered course**

```
---- Create view
----------------------------------------------
Create view The_Most_Registered_Course
AS
(select *
from Course
where Course_ID in  (select E1.Course_ID
                     from Enrollment E1
                     where not exists    (select Student_ID
                                          from Student
                                          except
                                          select Student_ID
                                          from Enrollment E2
                                          where E1.Course_ID = E2.Course_ID)))
```

- **View of courses that have both lab and theoretical sections**

```
----------------------------------------------
Create view The_Course_Have_Both_Of_Lab_and_Theory
as (
select *
from Course
where Course_ID in  (select Course_ID
                     from Section
                     Group By Course_ID
                     Having Count(Section_Types) >= ALL(select Count(Section_Types)
                                                        from Section
                                                        Group By Course_ID)))

select*from The_Course_Have_Both_Of_Lab_and_Theory
```

- **Create view to find the students with smallest age in each enrollment course**

```
Create view All_Students_Have_The_Smallest_Age_In_Each_Course
AS
select Student.Student_ID, Student.First_name + ' ' + Student.Last_name 'Full_Name',
Year(CURRENT_TIMESTAMP) - Year(Student.Dob) 'Age', Course.Course_ID, Course.Course_Name,
Course.Credits, Course.Dept_ID
from Student
        join Enrollment on Student.Student_ID = Enrollment.Student_ID
        join Course on Course.Course_ID = Enrollment.Course_ID
where Student.Student_ID in (
select Student.Student_ID
from   Student
where Year(Student.Dob) >= All
                              (select Year(Student.Dob)
                               from Student)
)
```

CHAPTER 7: CONCLUSION

7.1 Limitations

Our project does not include a user application with a UI designer to engage with clients because of the shortage of project time. Instead, we put an emphasis on building the database system and other related materials like an ER diagram, a relational model, and query technique to extract data that serves the consumer's desire.

Data security is also a limitation, which we have not been able to improve. Our project has not implemented multi-factor authentication to ensure that only authorized users have access to the system. Besides, in this project, we have not implemented data encryption techniques to protect important information from attackers who intend to steal or use it illegally.

One of the further limitations of creating a college management system is the limited scope of research. This can happen when the developers fail to conduct comprehensive research to understand the needs and requirements of the college administration, instructors, and students. For instance, the research team may have only focused on a particular aspect of the college management system, such as the student registration process, without considering other crucial features, such as financial management or academic performance tracking. As a result, the final system may not provide a comprehensive solution to the college's needs.

Developing a comprehensive college management system requires a team of skilled developers with expertise in different programming languages and technologies. Additionally, the development process may require expensive software tools, servers, and other resources. Without the proper technical expertise, the development team may not be able to create a high-quality application. This may result in a system that is unreliable, prone to errors, and difficult to

maintain. Besides, the average time to build a complete application can be suspended for at least four months, but we have just three months to set the plan and develop the application.

7.2   Future Plans

Since our current system does not contain many services for users, we have a decision to improve the system by fulfilling all of the limitations in the previous parts and then list out some features to enhance in the future.

The key problem of the college management system can be derived from the situation that most of the common work needs to be completed at the center of the department. To achieve this goal, we have several plans for this application. Firstly, our application will have a centralized platform that will contain all relevant information for each department. This will ensure that students and faculty members can access the required information from a single location, thereby reducing information from a single location, thereby decreasing the time and effort needed to visit each department individually. Secondly, the application will be designed to provide real-time updates on any changes or updates to information. This will ensure that students and faculty members always have the most accurate and up-to-date information available. Finally, we plan to integrate a chatbot feature into our application, which will enable users to ask questions and receive immediate responses without having to visit each department individually.

Another plan that we want to achieve is the application of new advanced technologies in our system. As we all know, modern technologies such as cloud computing, artificial intelligence (AI), and data analytics can provide greater flexibility, scalability, and efficiency in managing college operations. For example, cloud computing can enable seamless data sharing and

collaboration between different departments, while AI can automate routine administrative tasks, freeing up valuable time for instructors and students. As a result, by continually researching and developing new features, the college management system can remain up to date with the latest trends and practices in education management. This can help the college stay competitive and attract more students. To overcome this challenge, our team tends to put many efforts on the improvement of the current system by doing more research, and apply new technologies into the project.

CHAPTER 8: REFERENCES

List of references:

[1]   *Bhise, H. (2022, January 28). Beginners SQL Project: College Management Database. Medium. Retrieved March 13, 2023, from https://medium.com/@harshbhisemi/beginners-sql-project-college-management-database-4bdc421fd153*

[2]   *David Johnson. (2018). The Ultimate Guide to Implementation Plans. Wrike. Retrieved March 13, 2023, from https://www.wrike.com/blog/implementation-plan-ultimate-guide/*

[3]   *Fergusson, K. (2021, December 23). Entity relationship diagrams with Draw.io. draw.io. Retrieved March 13, 2023, from https://drawio-app.com/blog/entity-relationship-diagrams-with-draw-io/*

[4]   *Raghu Ramakrishnan University of Wisconsin Madison, WI, USA, Johannes Gehrke Cornell University Ithaca, NY, USA, Jeff Derstadt, Scott Selikoff, and Lin Zhu Cornell University Ithaca, NY, USA: https://pages.cs.wisc.edu/~dbbook/openAccess/thirdEdition/solutions/ans3ed-oddonly.pdf*

[5]   *The effect of MySQL Workbench in teaching entity ... - MECS press*. (n.d.). Retrieved May 2, 2023, from https://www.mecs-press.org/ijmecs/ijmecs-v8-n7/IJMECS-V8-N7-1.pdf

[6]   *Li, Y., Jing, C., & Xu, J. (1970, January 1). Immune Mobile agent and its application in Intrusion detection system. SpringerLink. Retrieved May 2, 2023, from https://link.springer.com/chapter/10.1007/978-3-642-29637-6_7*

[7]   *Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer Widom, of Department of Computer Science Stanford University: Database_Systems_The_Complete_Book*

[8]   *Varshney, H., Agarwal, S., & DSouza, D. (2023, February 9). Java connect to Microsoft SQL server: 4 easy steps - learn. Hevo. Retrieved May 2, 2023, from https://hevodata.com/learn/java-connect-to-microsoft-sql-server/*

[9]   *Lluno Christiana, The power of relational algebra in Database Management System. (n.d.). Retrieved May 2, 2023, from https://www.researchgate.net/publication/348606461_The_Power_of_Relational_Algebra_in_Database_Management_System*