VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
HO CHI MINH UNIVERSITY OF TECHNOLOGY

COMPUTER SCIENCE & ENGINEERING DEPARTMENT

ASSIGMENT 1 REPORT

Subject: Database System

ACADEMIC MANAGEMENT SYSTEM

Supervisor: PhD. Nguyen Thi Ai Thao

Students:

Nguyen Phuc Gia Khiem	2211573
Nguyen Truong Thai Bao	2210251

Ho Chi Minh City, November 13, 2024

CONTENTS

LIST OF FIGURES

Chapter 1. DESCRIPTION

1.1 Business context

OSU University is a new university established in the Republic of Rhythm. Therefore, leaders of the organization require an application used for course registration and scheduling activities as a part of their admission plan.

According to the demands, the system has to store the information of students including a unique student ID, first name, last name, date of birth, the current number of credits, program, and the ID of passed courses. Teachers are also the actors interacting with the system, their information that is stored includes ID, first name, last name, date of birth, and their current academic qualification. Each of the teachers and students has their accounts and the IDs of students are different from those of teachers, the student ID is a sequence of 7 digits with the first two digits referring to the start year while the teacher ID has only 5 digits with the first two digits refer to their department.

The university consists of many majors, and each of them manages its students, teachers, and courses. Any major has its ID and name, a student in a specific major must take some mandatory courses as they progress. In each of the majors, there is a teacher who is the leader.

There are many courses provided at this university, and each of them is characterized by the course ID, the prerequisite courses, the number of credits, its' program, the number of classes that it has, and the number of students attending, a course has at least one class and can have more, while a class just belongs to a specific course. A student can register for a class associated with a particular course, each class has its ID, the ID of the course it belongs to, the number of days in a week the class is held, the number of periods in a day, and the majority of classes are held at a specific room whereas some of them are online classes. A teacher can also choose the course in which he\she wants to participate but it must be met with their qualifications. Some courses require conditions for students to attend such as prerequisite courses or total credits of students.

There are three phases in the registration process.

1. **The first phase** just takes the information about students' expectations related to which courses they want to learn, there aren't any limits for students to take part in the courses. Teachers also choose their courses in this phase.

2. **The second phase** provides the particular classes for courses and schedules for students and teachers for the first time. The system must log the reasons for failed registration with some courses if they exist. In this phase, students can change the class that they belong to if and only if these changes do not overlap the timetables of other classes. The same things can be done by teachers. Users (including teachers and students) can still join a class belonging to a course that was not registered in the **first phase**.

3. **The final phase** has the same function as the second one but the student can not remove the classes that they already have been in.

After each of the steps in the process above, students and teachers can view their schedules as well as logs created by the system via their accounts. Besides the standard classes, last year's students must assign a project for graduation, each project can be handled by a group of students but there is only one teacher to supervise. A student only is able to study in a range of 11 to 25 credits per semester.

1.2 Data description

1.2.1 Entity types and attributes

User

- **ID (number, not NULL, unique**: Unique identifier for each user.
- **Name (string)**: Name of the user.
- **DateOfBirth (date)**: Date of birth of the user.
- **HashKey (string)**: The hashed crypton for authentication.

Student

- **Current_credits (number)**: The number of credits the student has earned can be inferred.
- **Program (string)**: Program the student is enrolled in (Gifted: **TN**, OISP: **CC**, Standard: **L**).

Teacher

- **Academic_qualification (string)**: Current academic qualification of the teacher. There are 3 values available for this attribute consisting of `bachelor`, `master`, and `PhD`.

Major

- **MajorID (number, not NULL, unique)**: Unique identifier for each major.
- **MajorName (string)**: Name of the major.

Course

- **CourseID (string, unique, not NULL)**: Unique identifier for each course.
- **Credits (number)**: Number of credits the course offers.
- **CourseName (string)**: Name of the course.
- **Program (string)**: Program to which the course belongs. There are 3 programs: Gifted: **TN**, OISP: **CC**, Standard: **L**.
- **No.Classes (number)**: Number of classes offered for this course.
- **No.Students (number)**: Number of students enrolled.

Class

Class is a weak entity type depending on the course.

- **ClassId (string)**: Unique identifier for each class.
- **Semester (string)**: The semester variable indicates which semester the class held in.
- **No.Periods (number)**: Number of periods in a session.
- **IsOnline (boolean)**: A boolean value specifying whether a class is an online class or held in a specific room.
- **Class session**:
  - **Time (string)**: The time in a day that the class occurs.
  - **Day (string)**: The day in a week that the class occurs. There are 7 values available: `Monday`, `Tuesday`, `Wednesday`, `Thursday`, `Friday`, `Saturnday`, and `Sunday`.
  - **RoomNumber (string)**: The ID of the room that the class is hosted. It includes two first characters referring to the campus ($\{A, B, C, D, E, F\} < number >$ for the first campus and $H < number >$ for the second), the rest of `RoomNumber` is 3 digits.

1.2.2 Relationships

- **Student - Major (IS_MANAGED)**: A student is associated with one major, one major can manage many students.
- **Teacher - Major (BELONGS TO)**: A teacher is associated with one major and one major can manage many teachers.
- **Teacher - Major (MANAGES)**: A major is leaded by only one teacher.
- **Major - Course (CONTAINS)**: A major has many courses, some of which are mandatory courses that students must complete.
- **Course - Class (DIVIDES)**: Each course is divided into one or more classes.
- **User - Course (REGISTERS, CANCELS)**: A user (student or teacher) can register/cancel to study/teach a course in the first phase of the registration process. Each relationship has the attribute `semester` and `timestamp`.
- **User - Class (JOINS P2, JOINS P3, LEAVES)**: When a course is divided into many classes (phases 2 and 3), those who registered for that course can join/leave the classes of the course but they cannot leave in phase 3. Each relationship has the attribute `semester` and `timestamp`.
- **Student - Course (HAS_RESULT)**: A student who studied a course will have the result of that studying, the result contains `Grade` is a number and `isPassed` is a boolean value to specify that if the student passed this course.
- **Course - Course (IS_PREREQUISITES)**: A course can have a list of requisite courses and can also be a requirement course for another.
- **Student - Class (STUDIES)**: After all phases, a student will specify the courses that they will study. The study processes of students are also stored in this relationship with `Result` as the grade and `semester` attributes.
- **Teacher - Course (CAN_TAKE)**: There are some courses that the teachers can take based on their qualifications.

1.2.3 Data and Business Constraints

- A student must belong to exactly one major.
- A teacher must belong to exactly one major.
- A course must have one or more classes whereas a class has to belong to only one course.

- Each course has to be contained by only one major while a major must have at least one course.
- A major has only one teacher as the leader, as well as a teacher, can lead only one major.
- A teacher can teach multiple courses but must meet certain academic qualifications.
- There are some courses that students associated with a particular major have to attend.

1.2.4 Semantic Constraints

- **Student ID and Teacher ID**:
  - **Student ID**: The `ID` starts with 18, 19, 20, 21, 22, 23, or 24 referring to students' start years.
  - **Teacher ID**: The `ID` starts with 00.
- **Credit Requirement**: A student can only register for courses if they meet a minimum number of earned credits. In detail:
  - For courses whose `CourseID` starts with 1: Every student can take it.
  - For courses whose `CourseID` starts with 2: `CurrentCredit` of a student must be greater than or equal to 28.
  - For courses whose `CourseID` starts with 3 and 4: `CurrentCredit` of a student must be greater than or equal to 56.
- **Qualification Requirement**: A teacher can only teach a course that meets with his\her degree, in detail, for the course whose `CourseID` starts with 3 or 4: The requirement qualification is at least Master's degree.
- **Prerequisite Condition**: Students cannot register for courses without completing the prerequisite courses specified by the course. In detail, in relation **RESULT** of a student, he\she must have attribute `Passed` is `True` to confirm that he\she completed the course. Students have to pass all the courses having relation **IS_PREREQUISITES** with the one they want to register for.
- **Class Overlap**: A student or teacher cannot register for classes with overlapping schedules. In detail, students cannot **JOIN** a **CLASS** that has attribute `time` of its lesson overlap the time of another **CLASS** that students already have joined in the same `DaysPerWeek` value.
- **Attendance Requirement**: A student has to study for at least 11 and at most 25 credits each semester, the total credits are calculated after each phase by adding attribute `Credit` of all courses having a class joined by students. The standards for teachers are at least 40 periods a week, the periods are calculated after each phase by adding `No.periods` of all classes that they attended group by week.

- **Daily Periods**: A student or a teacher must not exceed 10 periods each day. After each phase, the daily periods can be calculated using attribute `No.periods` and `DaysPerWeek` of a **CLASS**.
- **Tuition**: The tuition of a student can be calculated depending on his\her program. It is 800.000 VND\credit for Standard and Gifted and 1.200.000 VND\credit for OISP. So that we can find total tuition based on attribute `Program` of entity **STUDENT** and total credits that a student attended.
- **Duplicate course**: A student must not join a class that belongs to a similar course with another existing class without leaving it.
- **JOIN and LEAVE relationship**: With X is the number of joins for a specific class and Y is the number of leaves. We must have $X = Y$ or $X = Y + 1$.
- **Class registration based on the program**: Class ID is divided into three types based on their first letters:
  - **L** type: This ID shows that the class is for Standard students.
  - **CC** type: This ID shows that the class is for OISP students.
  - **TN** type: This ID shows that the class is for Gifted students.

  Students cannot join a class having the wrong type compared to their program.

Chapter 2. CONCEPTUAL DESIGN

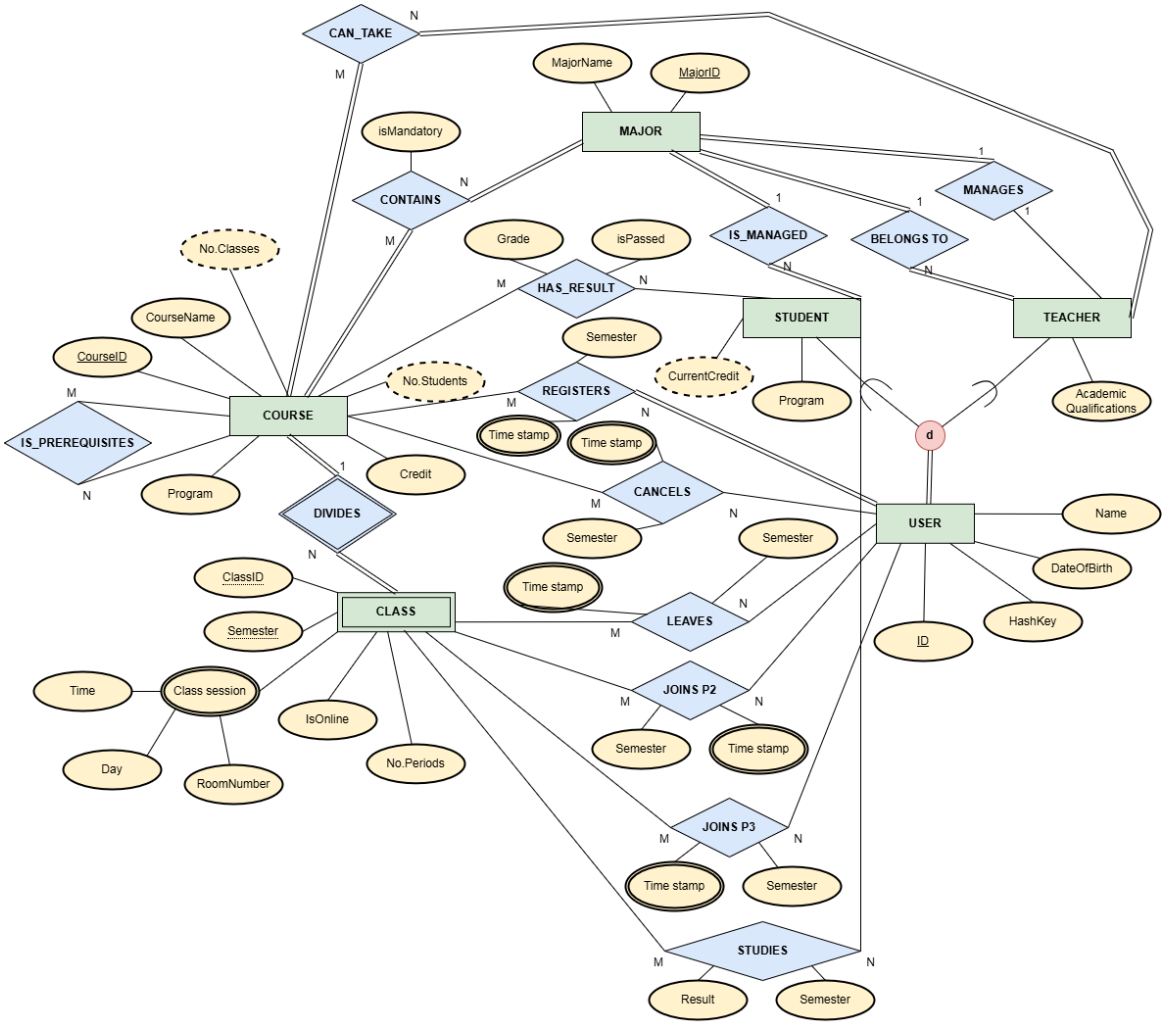

Figure 2.1: EER diagram

Chapter 3. LOGICAL DESIGN

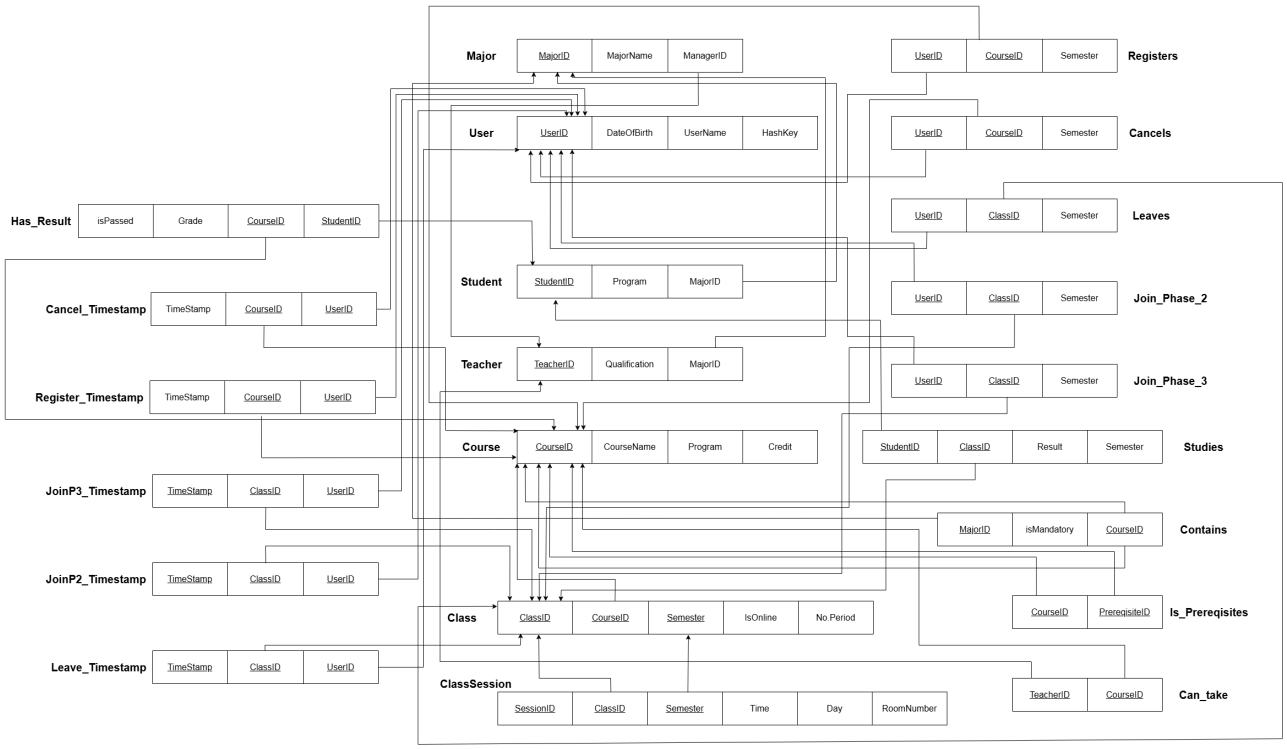

Figure 3.1: Logical design

Chapter 4. QUERY

1. **Specify the schedule of a student having ID 2210251 in semester 241**:

   $R_1 \leftarrow \sigma_{UID='2210251' \ \wedge \ Semester='241'}(JoinP2 \cup JoinP3)$
   $R_2 \leftarrow (Class \bowtie_{Class.ClassID=ClassSession.ClassID \ \wedge \ Class.Semester=ClassSession.Semester} ClassSession)$
   $R_3 \leftarrow (R_2 \bowtie_{R_2.ClassID=R_1.ClassID} R_1)$
   $Schedule \leftarrow \pi_{UserID,Semester,ClassID,CourseName,Credit,RoomNumber,Time,Day,No.Periods} R_3$

2. **Find the student ID and total credits of students who registered for less than 11 credits in semester 241**:

   $R_4 \leftarrow \sigma_{Semester='241'}(JoinP2 \cup JoinP3)$
   $R_5 \leftarrow (R_4 \bowtie_{R_4.ClassID=Class.ClassID} Class)$
   $Total\_Credit\_All \leftarrow_{UserID} T_{SUM(Credit)}(R_5)$
   $Invalid\_Credit \leftarrow \sigma_{Total\_Credit<11} Total\_Credit\_All$

3. **Find all the course codes that a student with code '2210251' is eligible to register for**:

   $R_6 \leftarrow \sigma_{StudentID='2210251' \ \wedge \ isPassed='true'}(Has\_Result)$
   $R_7 \leftarrow R_6 \bowtie_{R_6.CourseID=PrereqisiteID} Is\_Prerequisites$
   $\big(\pi_{CourseID} R_7) \cup (\pi_{CourseID}(\sigma_{PrereqisiteID=Null}(Is\_Prerequisites))\big)$

4. **Find the timetable for classroom number 101, building B1 at campus 1 on Monday at semester 241**:

   $R_8 \leftarrow \sigma_{RoomNumber='B1-101' \ \wedge \ day='Mon' \ \wedge \ Semester='241'} ClassSession$
   $Class\_Timetable \leftarrow$
   $\pi_{ClassID,CourseID,Time,No.Periods}(R8 \bowtie_{R8.ClassID=Class.ClassID \ \wedge \ R_8.Semester=Class.Semester} Class)$

5. **Find all courses of a student with student id '2210251' registered in phase 2**

of semester 241:

$R_{e1} \leftarrow JoinPhase2 \bowtie_{JoinPhase2.UID=Student.StudentID} Student$
$R_{e2} \leftarrow R_{e1} \bowtie_{R_{e1}.ClassID=Class.ClassID} Class$
$R_{e3} \leftarrow R_{e2} \bowtie_{R_{e2}.CourseId=Course.CourseID} Course$
$Course\_Names \leftarrow \pi_{CourseName}(\sigma_{UID='2210251' \wedge Semester='241'} R_{e3})$

6. **Find all compulsory courses of a student with code 2210251**:

$R_{e4} \leftarrow Student \bowtie_{Student.MajorID=Contains.MajorID} Contains$
$R_{e5} \leftarrow R_{e4} \bowtie_{R_{e4}.CourseID=Course.CourseID} Course$
$Must\_Take\_Courses \leftarrow \pi_{CourseID,CourseName}(\sigma_{UID='2210251' \ \wedge \ isMandatory='true'} R_{e5})$

7. **Find all courses of a student with code 2210251 passed which are his\her mandatory courses**:

$R_{e6} \leftarrow Must\_Take\_Courses \bowtie_{Must\_Take\_Courses.CourseID=Has\_Result.CourseID} Has\_Result$
$Passed\_Must\_Take\_Courses \leftarrow \pi_{CourseID,CourseName}(\sigma_{isMandatory='true'} R_{e6})$