CS5200—Database Management System

Assignment #1

Name(First, Last) : Yifan, Bao

NUID : 001058681

1. **Nouns:** faculty, student, course, module, lesson, widget, YouTube videos, slides, text documents, raw HTML, evaluations, essay assignment, submission assignment, exam, essay questions, multiple choice questions, blank questions, popularity, register's office, section, semester,seat capacity, undergraduate, graduate, enrolment, final grade, letter grade, student feedback, username, password, first name, last name, emails, phones, addresses, users, benefits, tenure status, parking, bank account info, financial aid info, work-study, scholarship, GPA, threshold, office hour, teaching assistant

2. **Verbs:** author, contain, broke up, build, come, create, keep track of, enroll in, register, assign, teach, review, receive, use, keep an eye on, have, is, offer

After classifying the nouns and verbs, I create a naive diagram based on description in problem statement, as showing below:

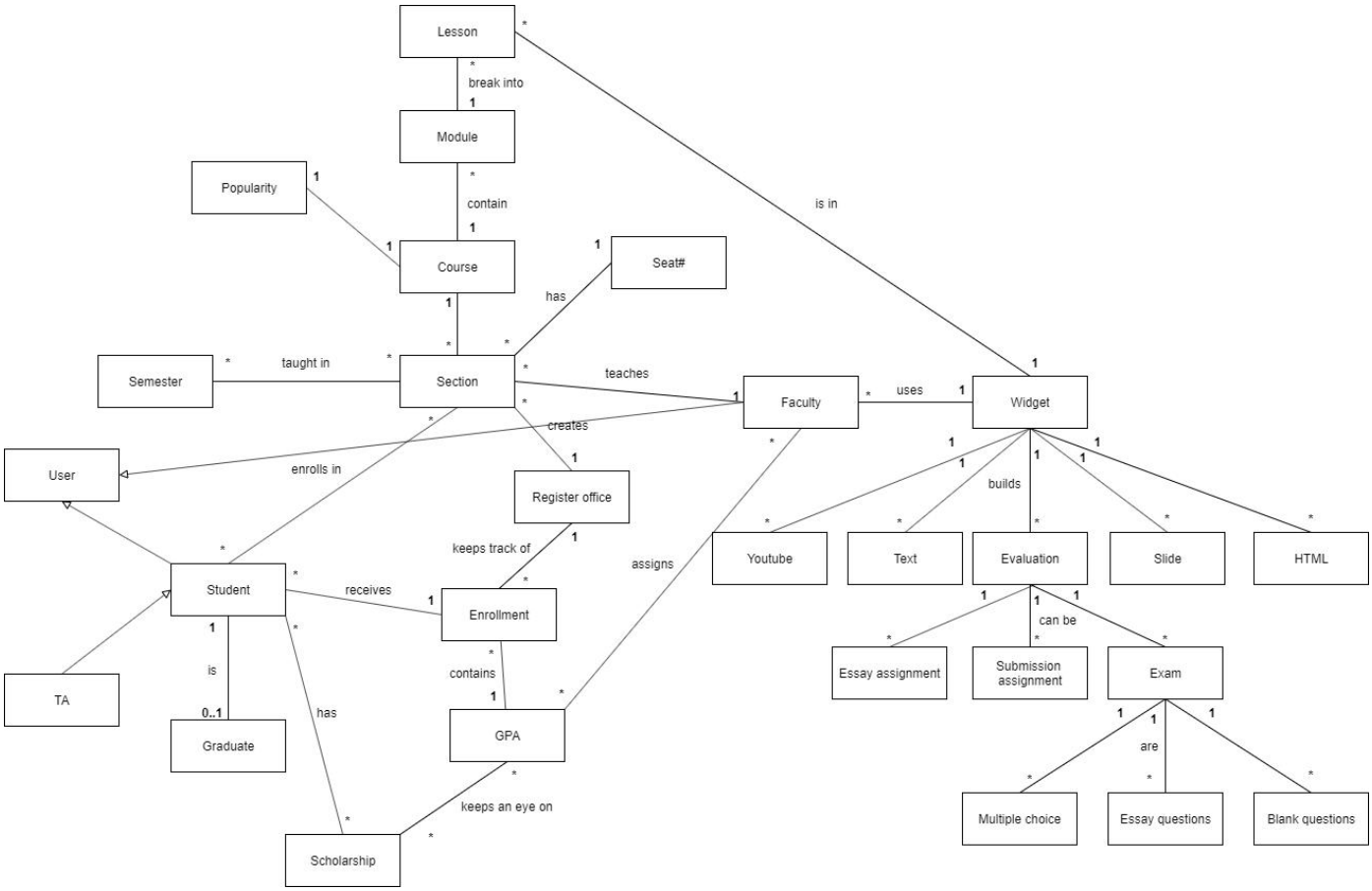


Figure 1: Naive class diagram

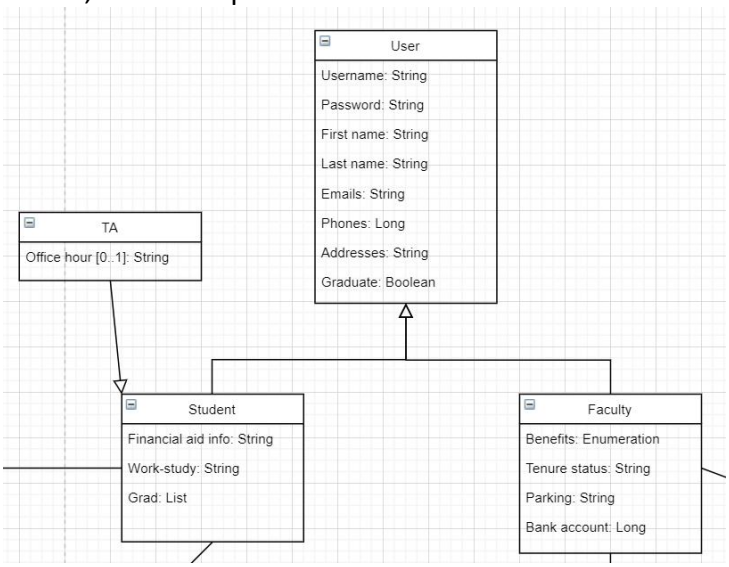
3. **Generalization/specialization**

Class of Student, Faculty and TA use the inheritance:

a) Student and Faculty are two subclasses of User respectively, they have all attributes of User. What's more, they have their own unique attributes. For

Student, it has scholarship and financial aid info. For Faculty, it has benefits and tenure status.

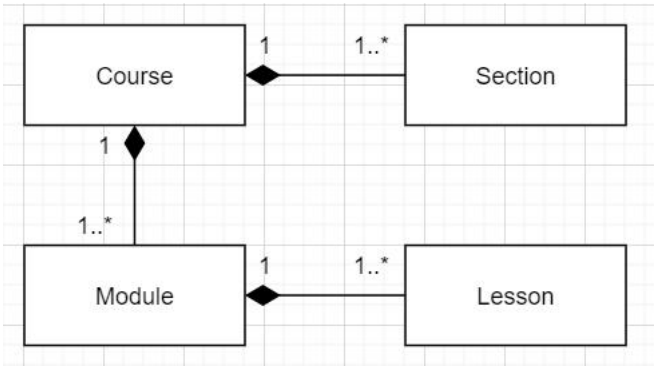
b) TA is a subclass of Student, it has the unique attribute, office hour, besides name, address, scholarship and so on.



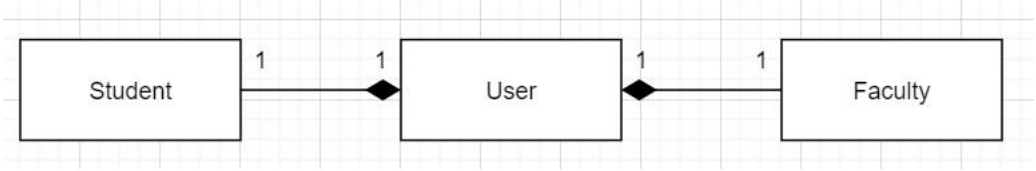
4. Associations, aggregation and/or composition

4.1 Composition

A) Courses contain Modules. Without Courses, Modules can not exist. Modules are broken into Lessons. Without Modules, Lessons can't exist. Courses have some Sections. Without Courses, Sections can't exist.

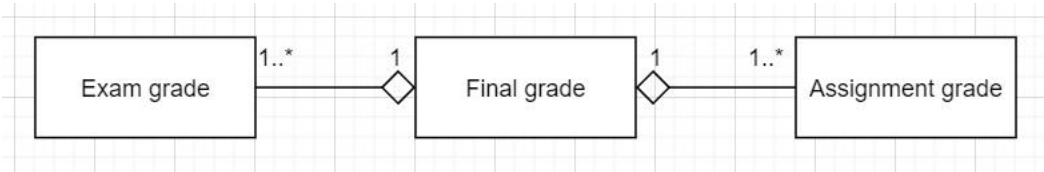


B) Students extend Users. Without User, Students can't exist, so do Faculty.



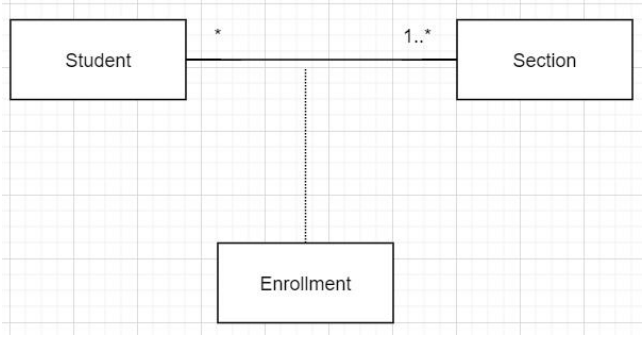
4.2 Aggregation

A) Final grades contain grades of each exams and assignments. But without the final grade, exams grades and assignments grades can still exist.

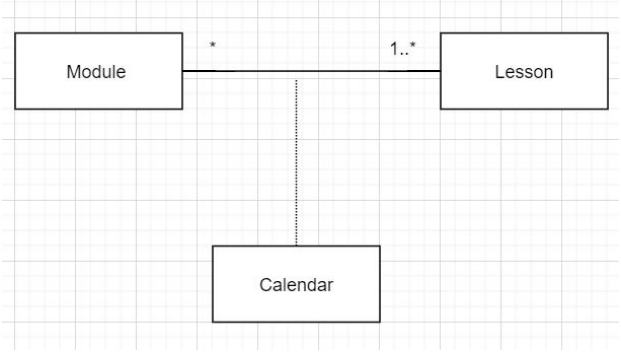


4.3 Association

A) Students enroll in different sections. Enrollment as an association class establish a relationship between Students and Sections.



B）Modules and Lessons can be ordered in different time. Calendar class can get info which Lesson of which Module should taken in what time.



5. Classes vs. attributes analysis

After creating the naive class diagram, I divide whole system into several parts.

A) User-Student/User-Faculty part: Based on problem statement, I find that Student and Faculty have some common attributes, like username, password, name. So I create a new class named User to contain these common attributes, and treat Student and Faculty as class with their unique attributes.

For Student class, I set two extra attributes, Scholarship and Graduate, to satisfy the requirement in problem statement. Because the number of courses in one semester that graduate student can register is less than undergraduate can register, I set Graduate type as Boolean to distinguish between the two types of students. What's more, not all TA has scholarship, so I set Scholarship type as Boolean to avoid all TA has scholarship attributes.

B) Widget part: I extract some types of documents as a new Document class which contains HTML, text, slide and YouTube video Hyperlink to simplify this part. At the same time, I create Evaluation class and Exam class. In Evaluation class, scores of Essay assignment and submission assignment is attributes which can be directly got. And for Exam grade, there are 3 attributes in Exam class to match the requirement that displaying scores for each section of the exam.

C) Section-Course part: Section class has Seat capacity, Prof name, Semester and Popularity, the first two attributes show info to Student, the last two attributes confirm in which semester section is scheduled. For Course class, I create Calendar class between Module class and Lesson class, which satisfy the requirement in problem statement.

D) GPA part: I extract GPA as a class, because there is an important attribute, threshold, in GPA. It determines whether a student can get a scholarship, and register's office will keep an eye on it.

Connect all the parts into one whole.

6. Correct data types

There are some data types which confused me at the outset.

The first one is Scholarship of class Student. If I set Scholarship type as Int, TA who extends Student will have scholarship, but not all TA has scholarship in reality. So, I set Scholarship type as Boolean, and create a new Scholarship class with an attribute. If Student have scholarship, value is true, and we can set the value of scholarship in Scholarship class.

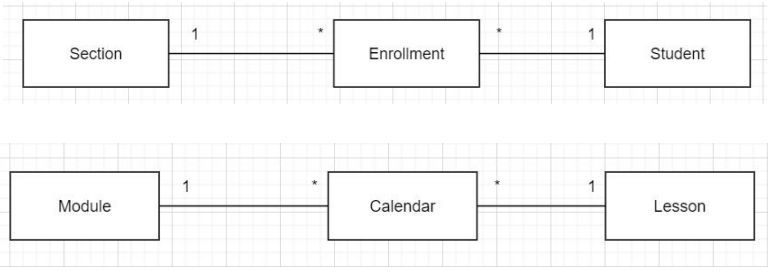
The second one is Graduate of class Student. I set Graduate type as Boolean. If the value is true, this student can choose less course in a semester. If the value is false, this student is undergraduate who can choose more course in a semester.

The next one is Semester of class Section. There are 5 different values the semester can be, so I set Semester type as Enumeration.

The rest of data types are shown in the final class diagram.

7. Cardinality

There are two association showing below:



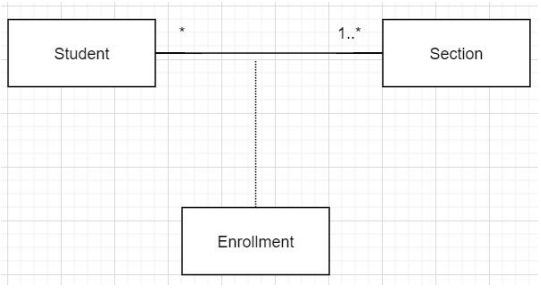
8. Remove any inadequate or redundant relationships, entities or attributes

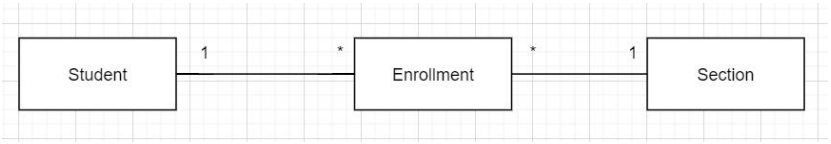
There are lots of inadequate and redundant relationships in my first naive class diagram. For instance, Faculty authors Course is inadequate. What's more, Widget types, like HTML, texts, slides, are redundant. And for Question types, multiple choice questions, blank questions and essay questions are also redundant.

In order to remove these relationships, I treat HTML, texts, slides as attributes of Widget, I use String type to describe the hyperlink of documents to simplify the system. And for Question types, I take the same way to get scores of different types respectively. As shown in final class diagram.

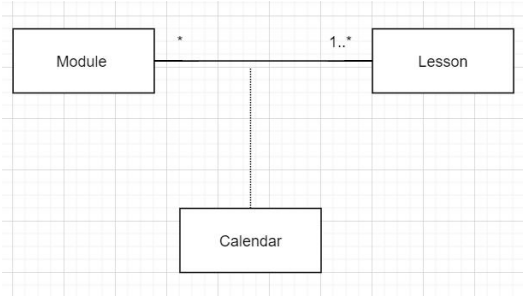
9. Reify

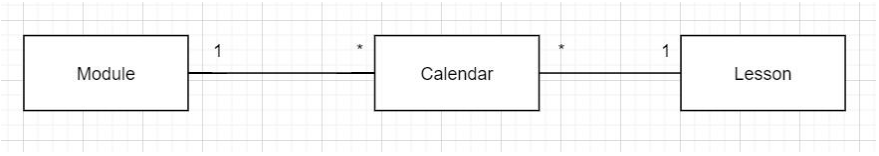
Students enroll in sections, they are many-many relationships. So we need a Enrollment class to reify this many-many relationship.





In the same way, we need a Calendar class to confirm which lesson of which module should be taken in what time.





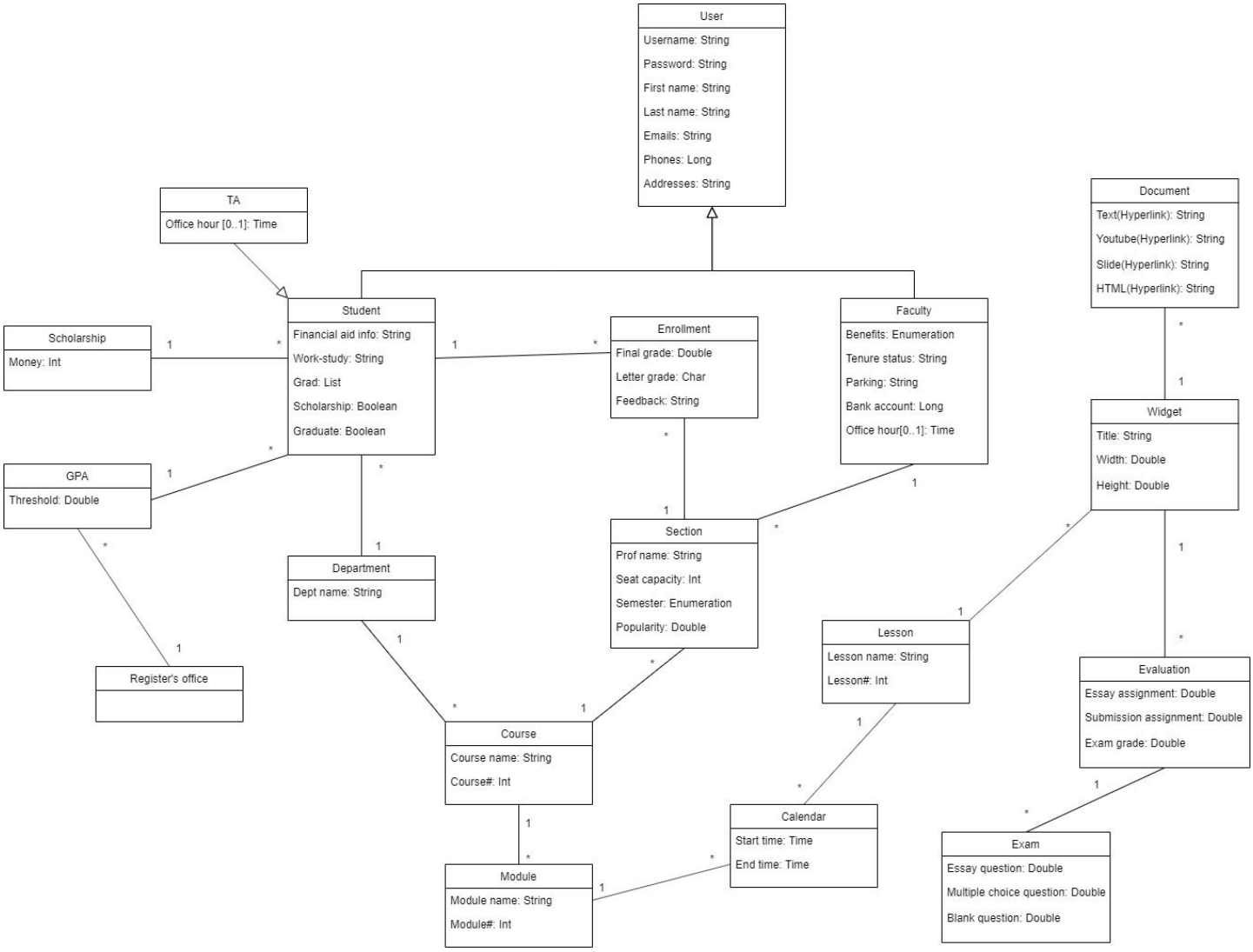


Figure 2: Final class diagram

10. Prose

First of all, I classify nouns to be candidates of Classes and Attributes, and select verbs to represent the relationship between Classes. Then, I create the naive class diagram based on those nouns and verbs.

Second, I reconsidered the relationship between classes in naive class diagram. After correcting data types and removing inadequate and redundant relationships, I simplify and optimize the class diagram version 2.

Finally, I reify many-many relationships left in class diagram to get the final class diagram.