CS5200 Assignment-1

Baian Chen

Step1 - List nouns that are candidate classes or attributes

Faculty
Student: undergrad, grad
Course
Learning module
Lesson
Topic
Calendar schedule
Widget
Youtube video
Slide
Text document
Raw HTML
Evaluation
Essay assignment
Submission assignment
Exam
Essay question
Multiple choice question
Fill in the blank question
Registrar's office
Section
Semester: fall, spring, full summer, summer 1, summer 2
Academic year
Enrollment
Seat capacity
Final grade
Letter grade
Student feedback
Personal profile info: username, password, first name, last name, email, phone, address

- for faculty: benefit, tenure status, parking, bank account info
- for student: financial aid info, work-study, scholarship

GPA
GPA threshold
Point
Office hour
Teaching assistant

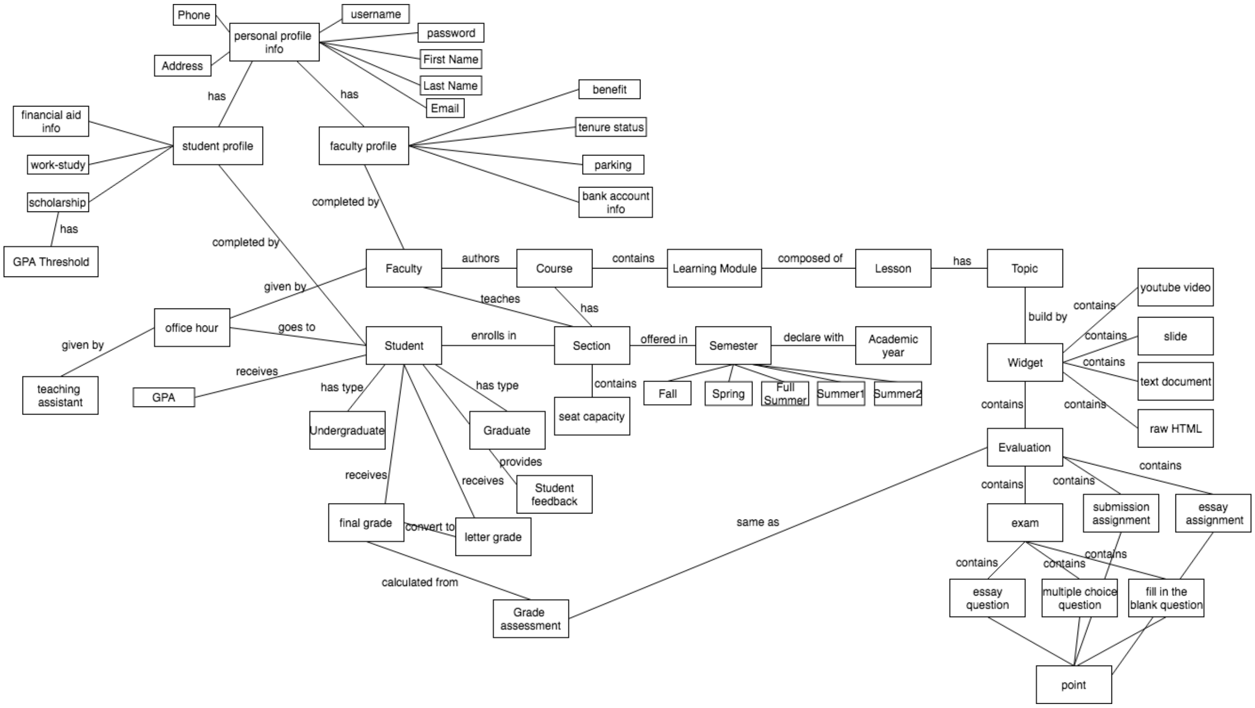
Step2 - List verbs as candidate relations between classes

Faculty authors courses
Student has type undergraduate
Student has type graduate
Course contains learning modules
Learning module composed of lessons
Learning module has calendar schedule
Lesson has calendar schedule
Lesson has topics
Topic build by widgets
Widget contains youtube videos
Widget contains slides
Widget contains text documents
Widget contains raw HTML
Widget contains evaluations
Evaluation contains essay assignments
Evaluation contains submission assignments
Evaluation contains exams
Exam contains essay questions
Exam contains multiple choice questions
Exam contains fill in the blank questions
Course has sections
Section offered in semester
Semester declared with academic year
Semester has type fall
Semester has type spring
Semester has type full summer
Semester has type summer 1
Semester has type summer 2
Student enrolls in sections
Section contains seat capacity
Section assigned with faculty
Student receives final grade
Student receives letter grade
Student provides student feedback
Student has personal profile info
Faculty has personal profile info
Personal profile info contains username
Personal profile info contains password
Personal profile info contains first name
Personal profile info contains last name

Personal profile info contains emails
Personal profile info contains phones
Personal profile info contains addresses
Personal profile info contains benefits (faculty only)
Personal profile info contains tenure status (faculty only)
Personal profile info contains parking (faculty only)
Personal profile info contains bank account info (faculty only)
Personal profile info contains financial aid info (student only)
Personal profile info contains work-study (student only)
Personal profile info contains scholarship (student only)
Scholarship has GPA threshold
Student receives GPA
Final grades calculated from grade assessments
Grade assessment contains assignments
Grade assessment contains exams
Assignment grades with points
Exam graded with points
Student goes to office hours
Office hour given by faculty or teaching assistant.

Step3 – Preliminary Class Diagram

From the verb list in step2, we can then get the preliminary class diagram for this LMS system.

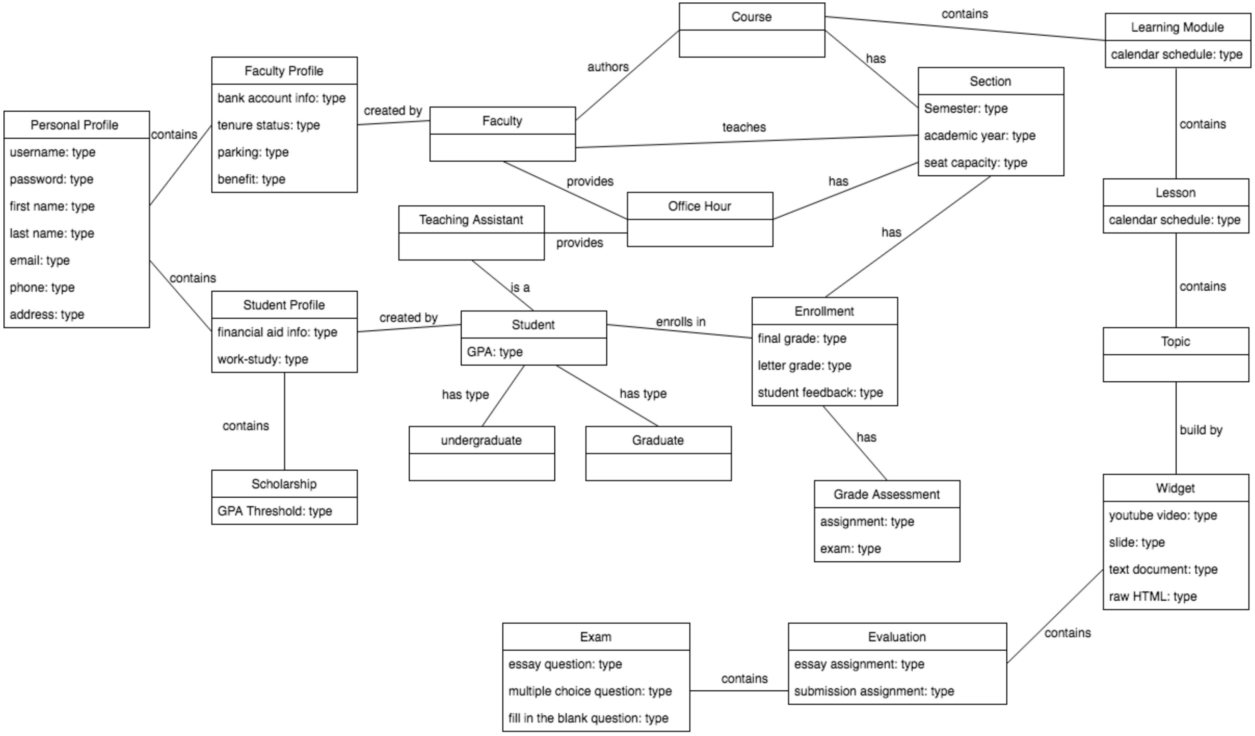


Step4 – Reification

Reification applied here to convert verb *"enroll"* into a class *Enrollment*, which connects between class *Student* and class *Section*, also it should contain *final grade, letter grade* and *student feedback,* which would make sense since a student should receive grade from a particular section, also provides feedback to this section.

Step5 – Class vs. Attribute Analysis

From the preliminary class diagram, we could set each noun properly to be a class or an associated attribute. The result shown below for a very initial class diagram after class/attribute analysis.



Step6 - Inadequacy or Redundancy

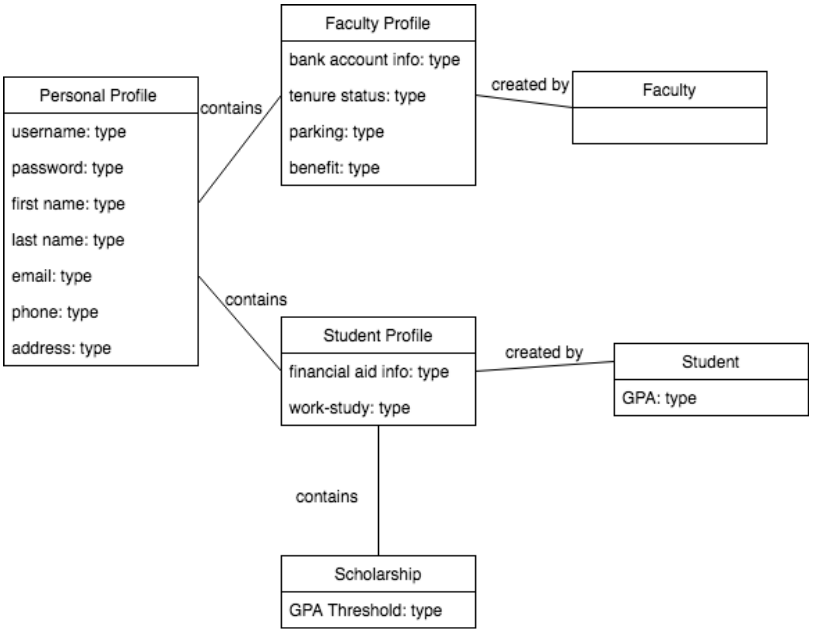
A.

By looking carefully at two paths start from *Personal profile*:

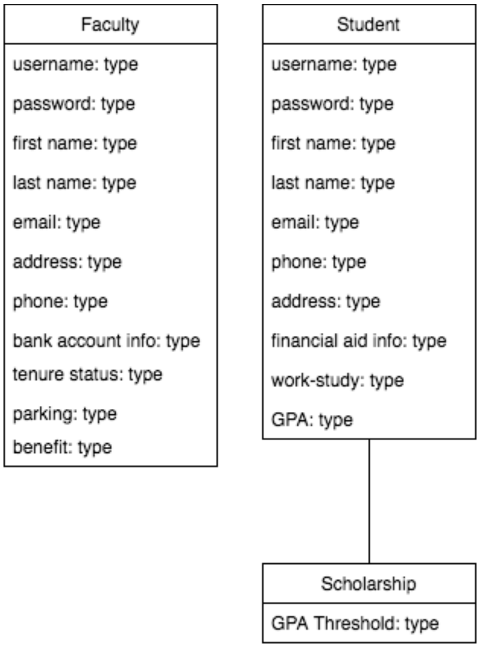
- personal profile → faculty profile →faculty
- personal profile →student profile→student

Those two paths are redundant since those attributes could be added to *Faculty* class and *Student* class directly.

Before:



After



B.

There is a loop between *Faculty, course* and *Section* which imply a redundancy. A faculty teaches a particular section, which belongs to a particular course. Therefore there is no need for the connection between *faculty* and *course* since we could know exactly a faculty teaches which course by searching through *faculty* → *section*→ *course*.

C.

The class *office hour* is kind of redundant here that we could make it to be an attribute of class *section*.

D.

The abstract class *widget* is kind of redundant here that we could move its attributes to class *topic,* which could simplify the structure.

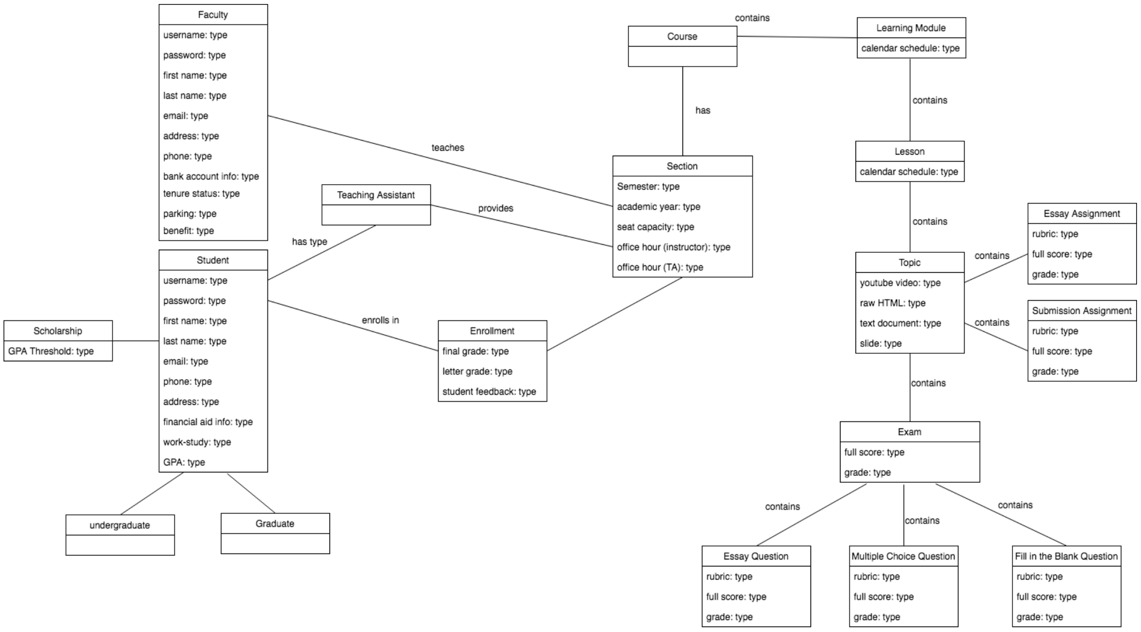
E.

The class *evaluation* could be removed by making its attributes to be classes. This is better since we miss an important part --- points in the initial diagram. From the requirement, LMS should have the ability to provide grade for each course, which is calculated by each part of evaluation, a student should receive a score for each assignment/exam/question, which is based on points. By changing assignments / exam's questions to be class, we could assign additional attributes to illustrate it.

F.

The class *Grade Assessment* could be removed since it is actually the same to the assignment and exam classes. So there is no need to leave both of them.

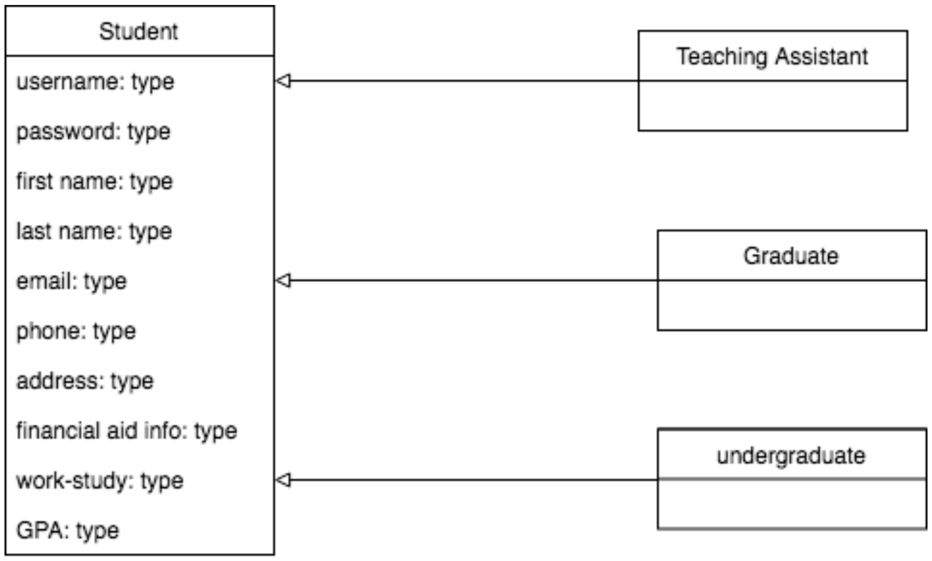
Diagram after step 6



step7 – Generalization / Specialization

There are three parts applied inheritance (however all those parts would be edited in further step)

- class *undergraduate* inherits *student*
- class *graduate* inherits *student*
- class *teaching assistant* inherits *student*



Step8 - Associations, aggregation and/or composition

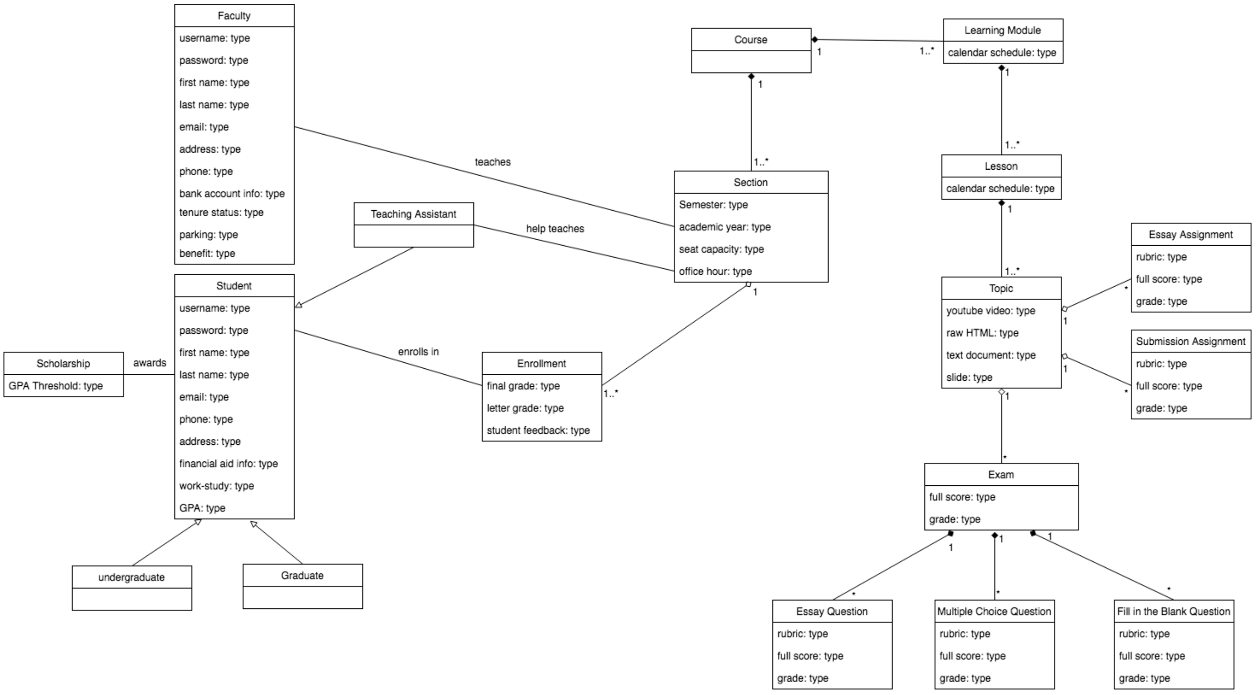
Let's look carefully at connections between each classes and edit them to be a proper type.

A. Aggregation

- *Section* and *enrollment*: A section should have at least one enrollment (1 to 1..*), which means at least one student registers for this section. Also the enrollment record for student should remain valid even if the section has been removed.
- *Topic* and *Essay Assignment, Submission Assignment, Exam*: A topic should have zero or more assignments and exams (1 to *). Also those evaluation widgets should be remained after topic's deletion since they are part of students' grade, which should always exist as long as the student is in record.

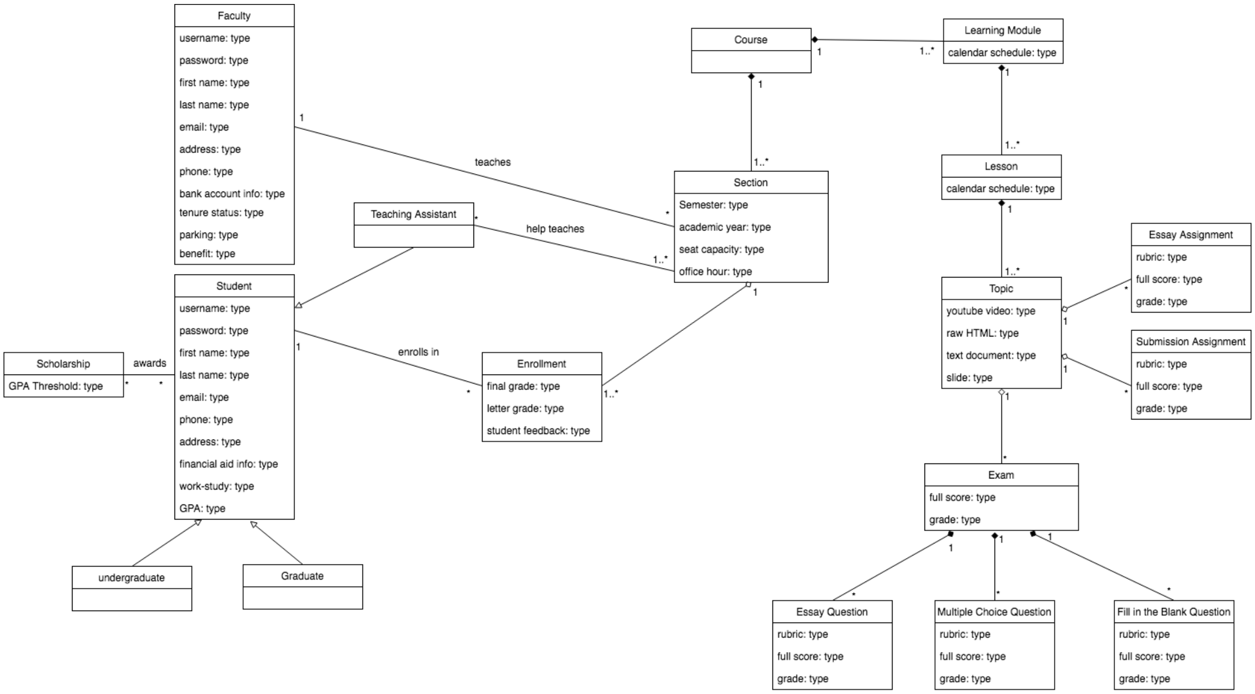
B. Composition

- *Course* and *Section*: A course should have at least one section (1 to 1..*). When delete course, along with its section.
- *Course* and *Learning Module:* A course should have at least one learning module (1 to 1..*). When delete course, along with its learning module.
- *Learning Module* and *lesson*: A learning module should have at least one lesson (1 to 1..*). When delete learning module, along with its lesson.
- *Lesson* and *topic*: A lesson should have at least one topic (1 to 1..*). When delete lesson, along with its topic.
- *Exam* and *its questions*: A exam should have zero or more questions (1 to *). When delete exam, along with its questions.



Step9 – Cardinality

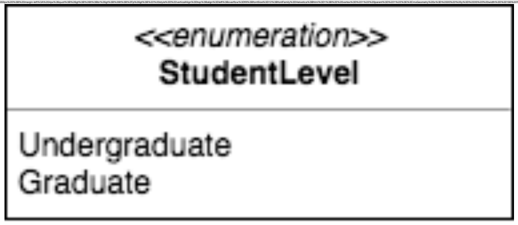
- *Faculty* and *Section*: A section should be instructed by one faculty; A faculty could teach more than one class.
- *Teaching Assistant* and *Section*: A section could have zero or more TAs; A TA could assist more than one sections but at least one section.
- *Student* and *Enrollment*: A student could have zero or more enrollment; A enrollment must corresponds to a student.
- *Student* and *Scholarship*: A student could have zero or more scholarship; A scholarship could have zero or more student.



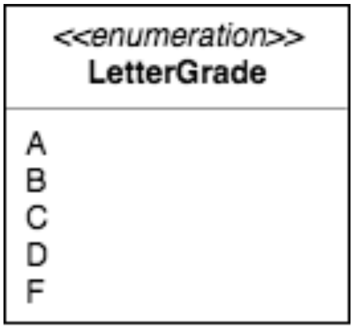
Step10 – Correct Data Type & Add Additional Attribute

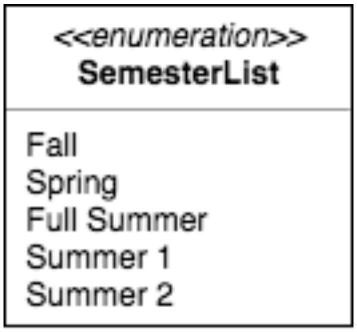
A.

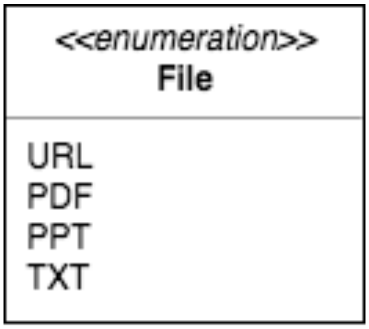
There are two classes *undergraduate / graduate* inherit from *student* but do not provide any additional useful info, what they relate to is a constraint that undergraduate should enroll in more course than graduate. However, this constraint seems like not easy to solve at database level. So we could roll those two classes to be attribute of student and solve this constraint in application level. We could add an attribute “level” and make its type to be an enumeration with “undergraduate” and “graduate”



B. Other customized type applied

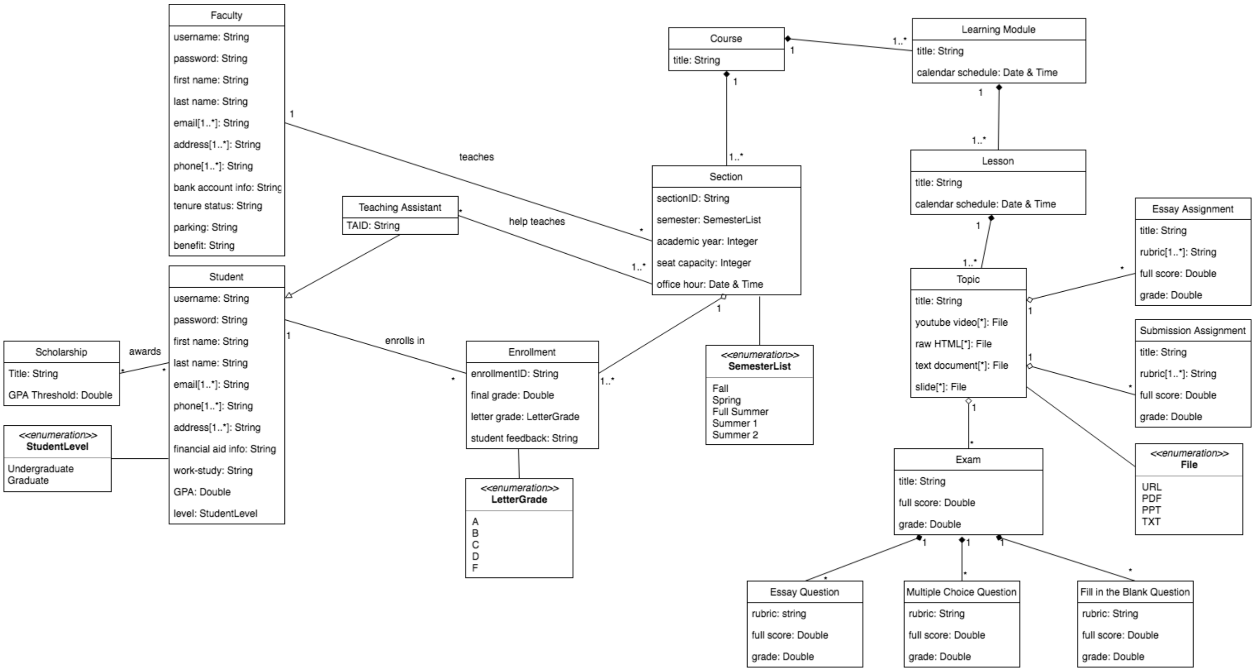






For “youtube video”, “text document”, “raw HTML”, “slide”, I defined a new type “File” since there is not a clear requirement for them. This “file” type could be a URL link that direct to those content widgets, or it could be specific file types, like PDF, PPT, TXT, etc.
Those attributes might be adjusted to separate classes if any detailed action required for them.

Final Class Diagram


Faculty
username: String
password: String
first name: String
last name: String
email[1..*]: String
address[1..*]: String
phone[1..*]: String
bank account info: String
tenure status: String
parking: String
benefit: String
1
teaches
*
Course
title: String
1
1
1..*
Learning Module
title: String
calendar schedule: Date & Time
1
1..*
Lesson
title: String
calendar schedule: Date & Time
1
1..*
Section
sectionID: String
semester: SemesterList
academic year: Integer
seat capacity: Integer
office hour: Date & Time
1..*
1
Teaching Assistant
TAID: String
*
help teaches
1..*
Student
username: String
password: String
first name: String
last name: String
email[1..*]: String
phone[1..*]: String
address[1..*]: String
financial aid info: String
work-study: String
GPA: Double
level: StudentLevel
Scholarship
Title: String
GPA Threshold: Double
awards
*
*
<<enumeration>>
StudentLevel
Undergraduate
Graduate
1
enrolls in
*
Enrollment
enrollmentID: String
final grade: Double
letter grade: LetterGrade
student feedback: String
1..*
<<enumeration>>
LetterGrade
A
B
C
D
F
<<enumeration>>
SemesterList
Fall
Spring
Full Summer
Summer 1
Summer 2
Topic
title: String
youtube video[*]: File
raw HTML[*]: File
text document[*]: File
slide[*]: File
1
*
Essay Assignment
title: String
rubric[1..*]: String
full score: Double
grade: Double
1
*
Submission Assignment
title: String
rubric[1..*]: String
full score: Double
grade: Double
<<enumeration>>
File
URL
PDF
PPT
TXT
1
*
Exam
title: String
full score: Double
grade: Double
1
1
1
*
*
*
Essay Question
rubric: string
full score: Double
grade: Double
Multiple Choice Question
rubric: String
full score: Double
grade: Double
Fill in the Blank Question
rubric: String
full score: Double
grade: Double