Marissa D'Alonzo
Database Management Systems
23 September 2018

Homework Assignment 1

Question 1: List nouns that are candidate classes or attributes.

Figure 1: Nouns as classes and attributes

Nouns (Classes)	**Nouns (Attributes)**
Faculty	benefits, tenure status, parking, bank account
Courses	student feedback
Learning Modules	
Lessons	
Calendar Schedule	
Widgets	Youtube videos, slides, text documents, raw HTML, evaluations
Evaluations	essay, submission, exam
Exam	essay, fill in the blank, multiple choice
Registrar	
Section	seat capacity, faculty, office hours
Semesters	fall, spring, full summer, summer 1, summer 2
Grades	final grade, letter grade, gpa overall, assignments, exams
Students	financial aid, work-study, scholarship
Everyone	username, password, first name, last name, emails, phones, addresses,

I created this chart by reading through the problem statement and highlighted all the nouns. Then, I naively determined which were classes and which were attributes using both my knowledge of Blackboard and which nouns were used as subjects in sentences.

Question 2: List verbs as candidate relations between classes

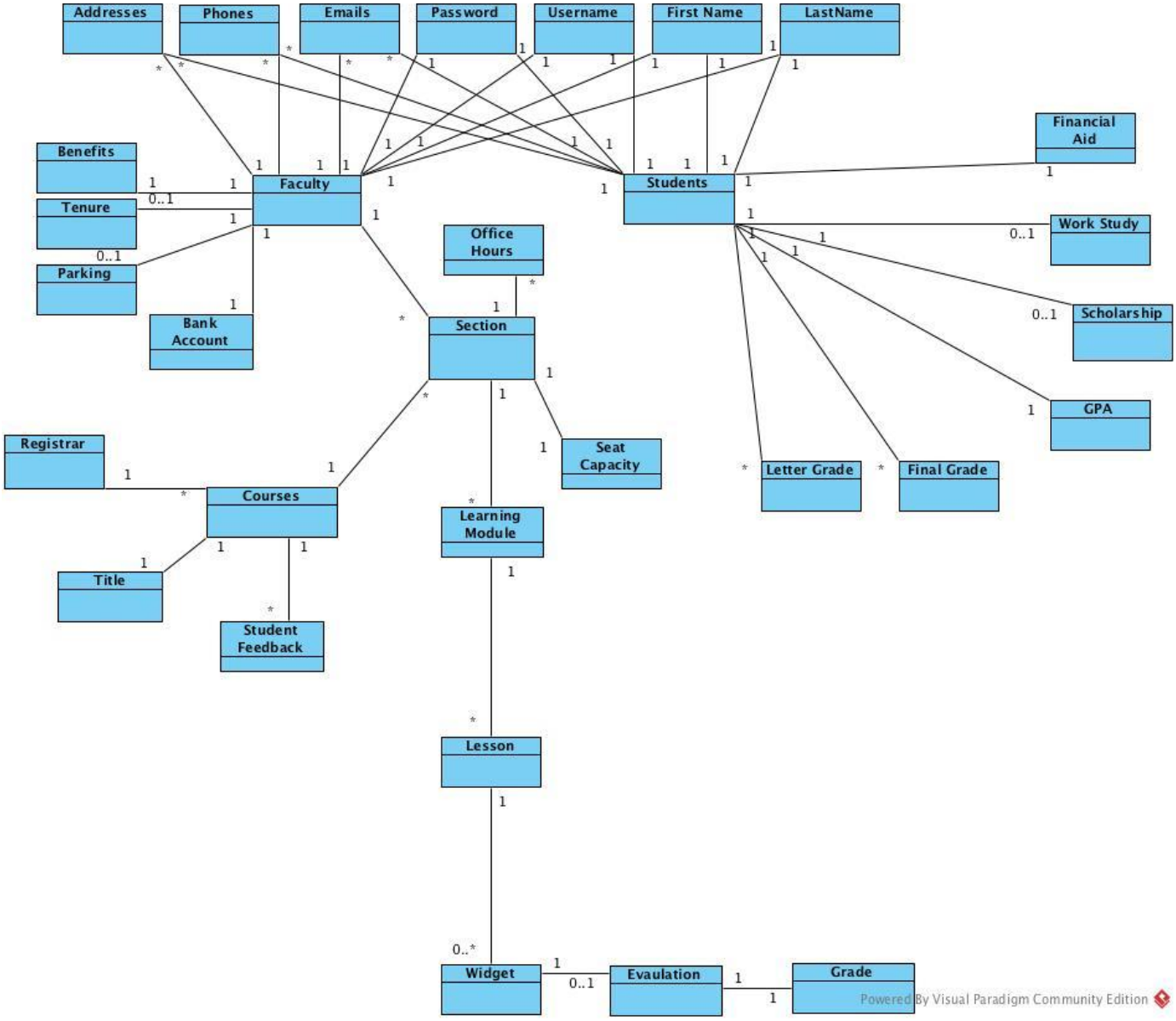
Figure 2: Verbs as candidate relations

Verbs
Faculty **authors** courses
Courses **contain** learning modules
Learning modules **are broken up** into lessons
Modules and lessons are **rearranged** depending on calendar
Exams **evaulate** students
Registrar **creates** courses/sections
Students **enroll** in classes
People **verify** information
Students **see** seat capacity and professor
Lessons **contain** widgets
Registrar **tracks** student progress
Grades **broken up** into exams and assignments

I created this chart by reading through the problem statement and highlighting all the verbs.

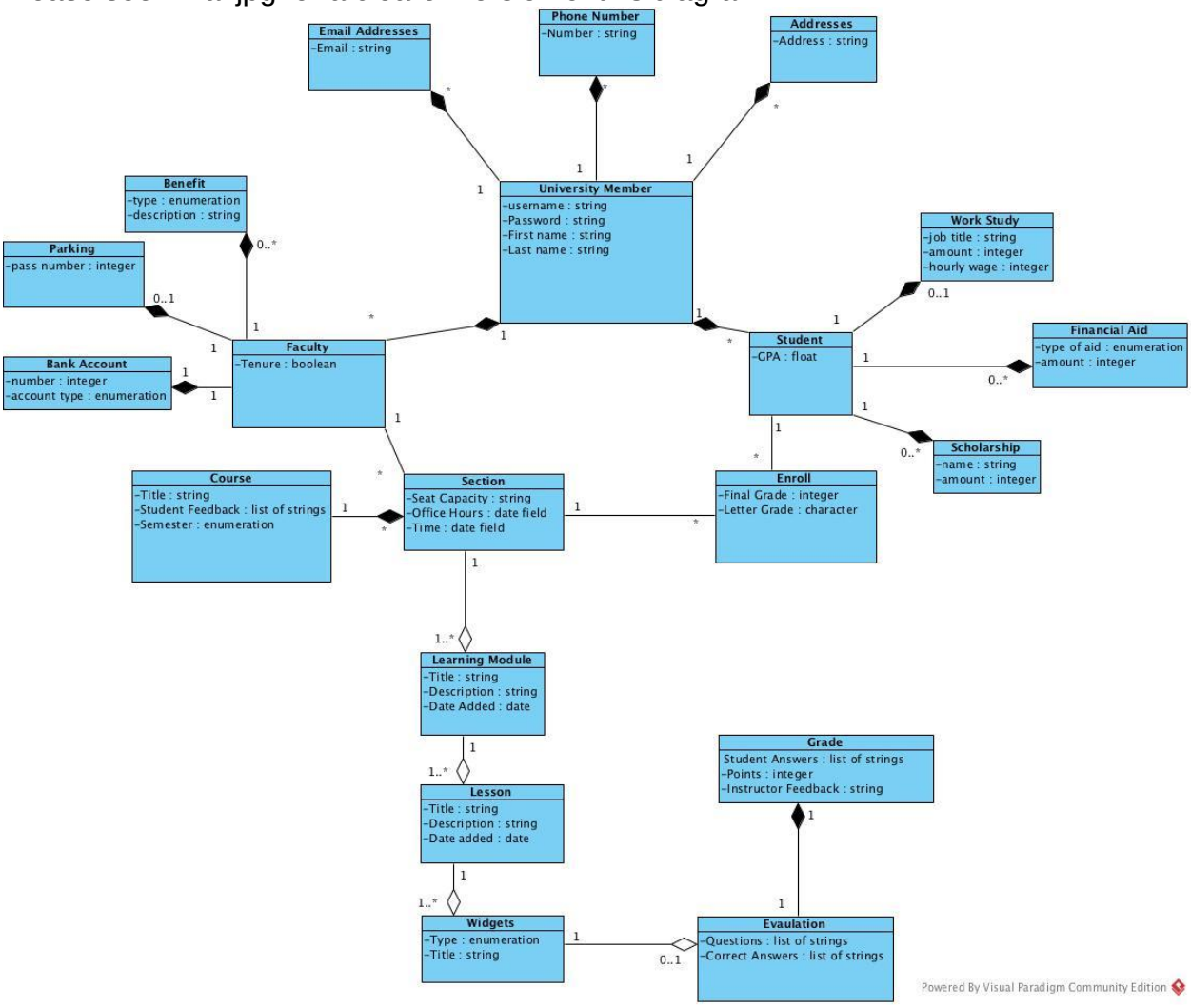
Question 5: Naïve class diagram

After making a list of all the nouns and verbs, I created this naïve class diagram by simply writing out all the nouns and connecting them with lines in the same way they were connected with verbs in the problem statement. Please see “Naïve.jpg” for a higher quality photo.



Question 5: Final class diagram

Please see Final.jpg for a clearer version of this diagram.



In the final class diagram, I have created a new class not specifically mentioned in the problem statement, the University Member. There are several attributes, such as username and password, that are shared between Faculty and Students. It would be wasteful to list them twice, so I created University Member as a parent class to hold them both. Because multiple email addresses, phone number, and home addresses can be listed, I made each of them their own class to keep first normal form. The Faculty and Student classes each hold or link to the attributes unique to them. Originally, I had work-study, financial aid, and scholarship as attributes of student, but when I was attempting to determine the data type of each, I realized that no useful information can be gleaned by making them a single line and expanded them to their own classes. The same logic was followed with the benefit, parking, and bank account information by the faculty class.

There is an Enroll association class, explained in more detail in Question 9, to connect the Section and Student classes together. This holds the final grades for that

class, instead of just holding it in the Student class. This eliminates issues with storing multiple grades in the student's section, such as identifying which grade goes with each class.

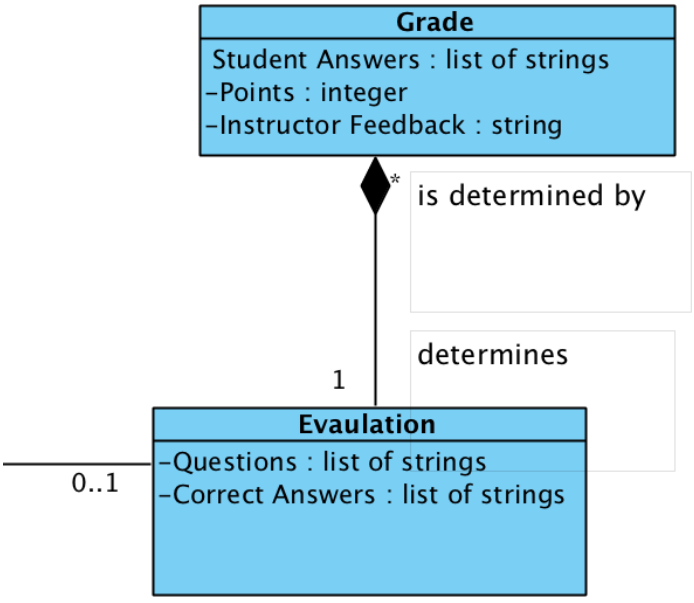
In the Section class, I added the "Time" attribute because, although not mentioned in the problem statement, is something students need to know. Title is not mentioned here, because it is stored in the Course class along with student feedback. Learning Module and Lesson contain title, description, and date added. This allows them to be rearranged based off the calendar, as specified in the problem statement. The Widgets class contains a field for the type of widget, such as HTML, video, etc, and a title. This field must be an enumeration, in order to prevent invalid file types from being uploaded.

If the widget contains an evaluation, there is an optional evaluation class. This class holds the questions asked in the evaluations, as well as the correct answers. The correct answer field must be hidden to students. There is another association class, grade, which stores the student's answers, the points awarded, and any instructor feedback, such as suggestions to improve essays or simply an encouraging message. Originally, I considered combining the Evaluation and Grade classes, but that was not feasible because for each evaluation, there are many different students receiving grades, and it could become difficult to organize the grades as an attribute of a class.

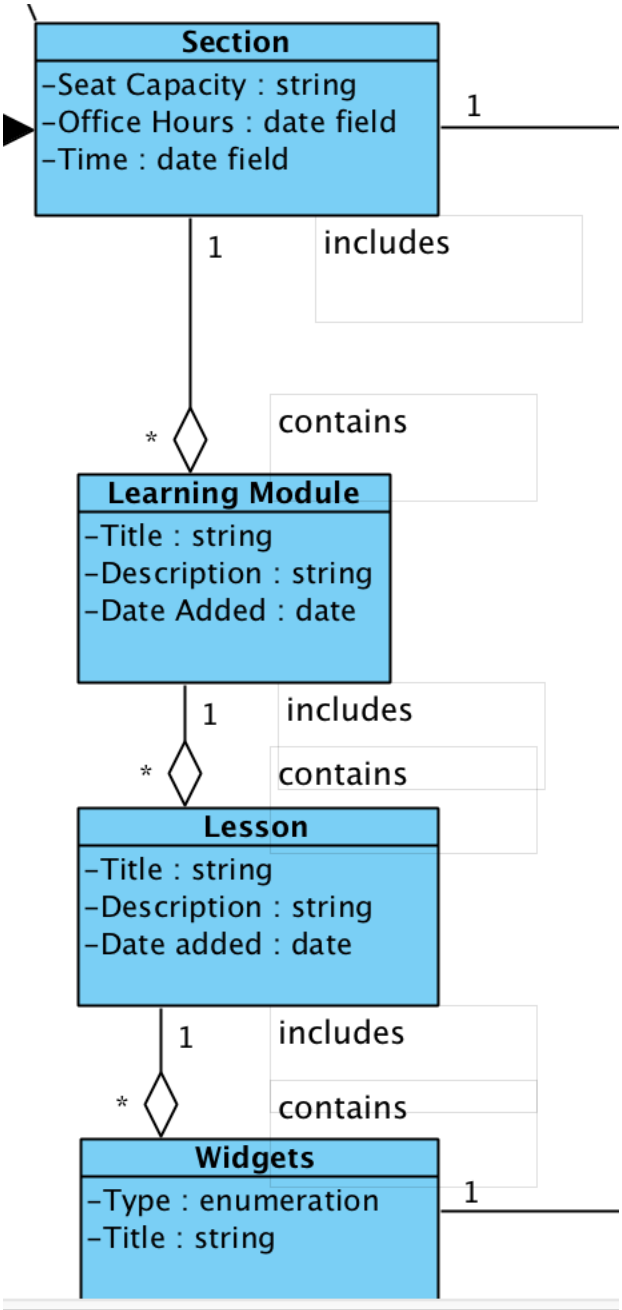
Question 3: Generalization/inheritance

Inheritance is used several times in the final class diagram, most notably with the Faculty and Student classes inheriting all fields from University Member, instead of listing them twice in each class. University Member also inherits fields from Phone Numbers, Email Addresses, and Addresses. Students inherit info from Work Study, Financial Aid, and Scholarship, while Faculty inherit from Benefits, Parking, and Bank Account. In addition, inheritance is also visible in Course->Section->LearningModule->Lesson->Widgets. This is useful for the student and professor to easily organize and navigate to the lessons and widgets.

Question 4: Associations, aggregation, composition



This figure shows the composition of Grade and Evaluation.



This figure shows the aggregations of Section, Learning Module, Lesson, and Widgets

University Person and Faculty/Student classes are an example of composition; a university person cannot exist that is not a faculty or student. In addition, Email Address, Phone Number, and Address are compositions of University Person. Benefits, parking, and bank account rely on Faculty, and are thus an example of composition. The same logic can be applied to Scholarship, Financial Aid, and Work-Study with Student. Section and enroll are an aggregation, there can be a section with no students enrolled, at least temporarily. However, student and enroll are an example of composition. Learning Module, Lesson, and Widgets are all aggregations to section; they do not necessarily need to be tied to a section in order to exist. For example, if a professor

taught a class for multiple semesters or taught multiple sections, they would not want all the lessons deleted if they deleted an old semester section. A Grade class cannot exist without its corresponding Evaluation, so that is an example of composition.

Question 6: Correct Data Types

Please see Final.jpg for the correct data types assigned to each attribute. Most are self-explanatory. I chose the enumeration type for benefits, financial aid, semester, account type, and widget type because they are all attributes that must be of a certain type in order to be valid. For example, if “semester” was just a string, you could make a course in a semester that does not exist, such as Winter, instead of the 5 valid options: fall, spring, summer 1, summer 2, and full summer.

Question 7: Cardinality

Please refer to Final.jpg and Naïve.jpg to see the cardinalities assigned to each relationship. Most are self-explanatory, but I did make some assumptions. A student can only have one work-study job, and a faculty member can only list one bank account and parking pass. Students can have between 0 and infinite scholarships and types of financial aid. Each Learning Module and Lesson must contain at least one Lesson or Widget, respectively.

Question 8: Removing redundancies

In moving from the list of nouns and verbs to the naïve class diagram, I removed all nouns that were just examples of attributes, such as “Youtube videos, slides, text documents, raw HTML, evaluations”, “fall, spring, summer 1, summer 2, full summer”, and “essay, submission, exam”. When moving from the naïve class diagram to the final class diagram, I eliminated the Registrar class because it was not necessary. The registrar determines the courses, but since they serve no other purpose, and who makes the courses does not seem to be relevant in this application, I ignored it. I also deleted the connection between the Enroll and Grade classes, because it was redundant. A student could find their grade for each evaluation by traveling through Section->LearningModule->Lesson->Widget->Evalulation->Grade.

Question 9: Association classes

There was on association class in my class diagram, the enroll class. The enroll class was made to clarify the relationship between the Section and Student classes. It is a many-many relationship, and as a result, storing the grades for each student in the Section class would make it difficult to tell who received it, while storing it in the Student class would make it difficult to tell what section it was for. Making a separate class to store that information eliminates that issue.