CPSC 5071 Project Milestone 1 (Group 5)

Mission Statement

The purpose of this project is to design, implement, and populate a database for storing information relevant to academic administration and performance at a university. The database will:

- support the day-to-day needs of students, professors, and administrators
- enable analysis of student academic outcomes by tracking relevant data such as section/room capacity, location, or number of TA's.

For us, a successful database is one that is both functional and easy to use and maintain.

Our ultimate goal is to develop a model to help us predict and understand student performance. By tracking detailed information about courses, enrollment, and outcomes, we aim to identify patterns and important predictors that will facilitate data-driven decisions to best improve student learning and achievement.

Description

Universities both *require* and *generate* a lot of data, from course offerings and student enrollments to room capacities and technological resources. As universities grow, managing this data with manual processes or ad-hoc systems becomes unsustainable; information inconsistencies, data redundancies, and coordination challenges emerge when different people maintain their own records without a unifying system. So, as universities grow, so too does the need for a systematic approach to data management. In this project, we develop a database for perhaps the most important part of the university data ecosystem: the academics.

One of the key challenges of developing a database of any kind is in clarifying who your users are and figuring out what they actually need. This involves iterative improvement via collaborative communication, modeling, and feedback. The users of *our* database are students, professors, and administrators, and each type of user has their own view, use case(s), and needs of the database. For example, administrators are probably a lot more concerned with room capacity and scheduling conflicts than professors.

The diversity of views and use cases naturally leads to increased database complexity–there are a lot of entities with a lot of relationships and a lot of constraints. The remedy to complexity is a systematic approach. For us, this means systematic designs and documentation (across varying levels of abstraction). For example, we construct diagrams for our users' view of the data (external models) and then merge these into a single, unified view (the conceptual model). But even organized database diagrams cannot save you from requirements that cannot (feasibly) be implemented in a DBMS, such as exceedingly complex constraints. Cases like this motivate the need for *project-wide* documentation so that such constraints aren't forgotten about and implemented at the application level.

One technical challenge in this project is in properly accounting for intermediate states/different modes of operation. For example, you'd usually think that each section of a course needs to have an assigned professor, but this isn't necessarily true *during registration* when things are still being decided. Handling these "different modes" requires (1) learning about them from the users and then (2) designing the database in a way that can accommodate all of them. In the previous example, we'd want to make the professor-section relationship *optional* so that sections can be created without an associated professor.

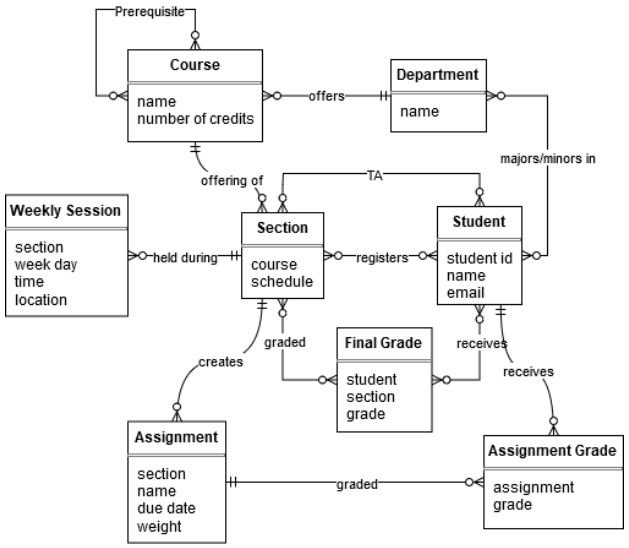
Universities both generate and store a lot of data, and they don't do it for fun but because it is useful and necessary. Whether it be in maintaining student private records or budget allocations, data is *important*. Earlier we talked about developing a database to support the use cases and needs of students, professors, and administrators, but *we* also have a use case for the database–data analysis–and it brings its own challenges, namely *performance*. We want to be able to analyze the database for patterns in and predictors of academic outcomes. Such analysis can be computationally expensive depending on the size and design of the database and queries. We address this in our choice of indexes and whether to deformalize certain tables to optimize the performance of common queries.

External Model

The Student View

Out of the three different types of users, students have the most use cases of the database including:

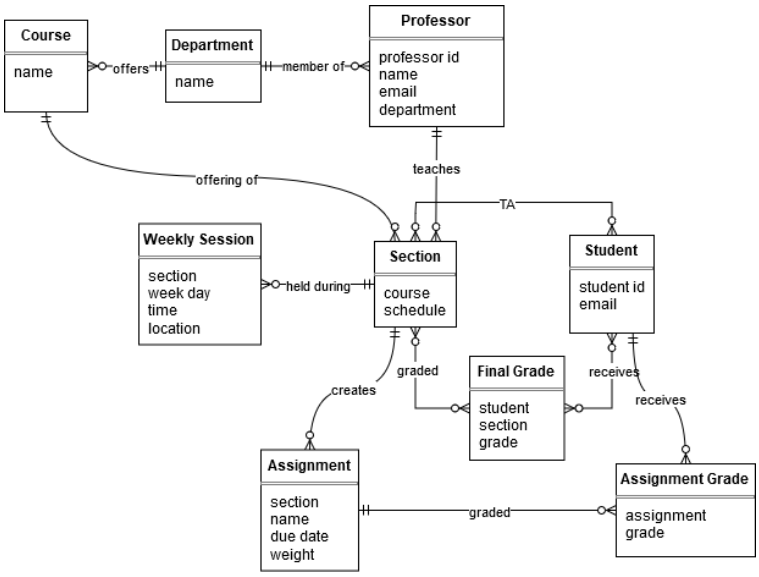
- Registration: students want to find *sections* of *courses* with *weekly sessions* that fit their schedule. They will need to have already taken the course *prerequisites*.
- Major/minor status: students want to see their major(s) and minor(s) positions in *departments*.
- TA position(s): students want to see which *sections* they have a *TA* position for.
- Assignments and grades: students want to see the *assignments* of the *sections* they're in and the *grades* they receive for those assignments. At the end of the quarter, they want to see their *final grades*.



The Professor View

The use cases that a professor might have for the database include:

- Teaching schedule: professors want to see which *sections* they're teaching and the *weekly sessions* of those sections.
- Teaching assistants: professors want to see who their *TA*'s are.
- Creating and grading assignments: professors want to create *assignments* and enter *assignment grades* (and *final grades*).
- Department position: professors want to see their position (membership) within a *department.*

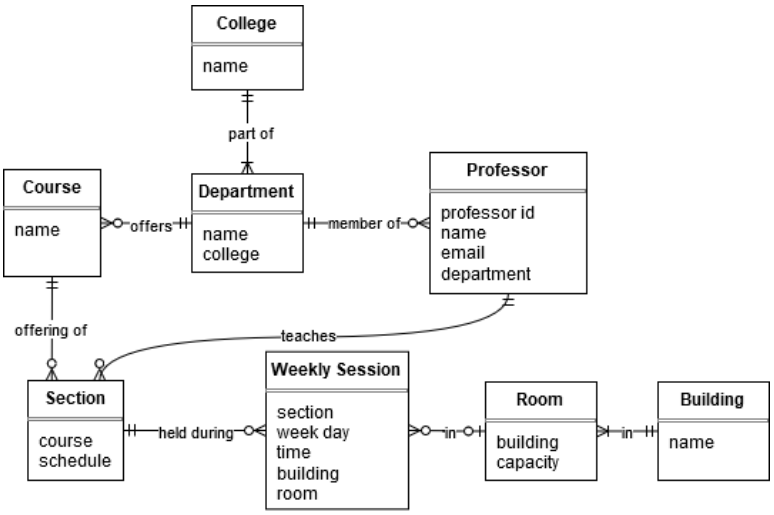


The Administrator View

Administrators (or the programs that execute with their authority) are tasked with managing the operations of a university:

- Course scheduling: administrators want to see all the available *buildings* and their *rooms* so that they can schedule the *weekly sessions* of *sections.*
- Managing professors: administrators want to see *professors'* positions within *departments*
- University organization: administrators want to see how *departments* are organized into *colleges*

For the purposes of this database, this mainly boils down to course scheduling (e.g., making sure that high-capacity sections are assigned high-capacity rooms), but it can also include managing professor positions within departments as well as the hierarchical organization of the university.



Logical Model