Webapp URL - http://flip2.engr.oregonstate.edu:4745/

Project Title - Rate My Course

Team Members - Nicolas Cortes, Alexander Molotkov.

Fixes based on Feedback from TA:

Step 3:

We were not including feedback from all previous steps so we updated the document to include all previous steps.

Step 2:

We were not meeting the requirement below in the project guide. In order to meet the requirement we removed the not NULL constraint from the authorID FK in the Reviews table. This way Reviews would have an optional participation with Authors. We updated the ERD to reflect this change. We also changed the relationship table name to Majors_Courses and Colleges_Majors.

"In a one-to-many relationship (like bsg_people to bsg_planets), you should be able to set the foreign key value to NULL (such as on a person in bsg_people), that removes the relationship. In case none of the one-to-many relationships in your database has partial participation, you would need to change that to make sure they can have NULL values."

Step 1:

It looked like the grading TA missed the M:M relationship between colleges and majors, dinging us points because they only saw the M:M relationship between majors and courses. This is probably because it wasn't very clearly written in the Outline. We made that relationship more clear. To clarify, the two M:M relationships are courses to majors and colleges to majors.

The grading TA indicated that we should also include the reciprocal relationships under the 'relationships' section of the tables. If X is connected to Y we also wrote down that Y is connected to X to make each individual outline more clear.

Fixes based on Feedback from Peer Reviewers:

Step 4:

- Increase character count to 255 for collegeName.
- Add NOT NULL constraint for Courses.collegeID.
- Update ERD to show Colleges_Majors.
- Add UI functionality and query to DELETE from Majors_Courses.

Step 3:

- Added UI to be able to search reviews by author.
- Dropdown input that will be dynamically populated was added to the Search for a Course page.

- Added basic validation to search and signup forms.
- Added delete buttons to delete reviews, profiles, and courses.
- Added form to create a new college.
- Added option to post as anonymous to Create Review page so the author field can be NULL.
- Display state next to the college name in the Course Reviews page.
- Changed profile page to specify it only shows reviews.
- On outline, changed title to entityTitle on three entities so attributes are clear.

Step 2:

For the most part, feedback was positive and most people agreed with the way that we set up our entities. One comment recommended that we describe the additional tables needed to form our M:M relationships in our 'database outline' section. We were grateful for the feedback but we wanted to keep the section focused on traditional entities and attributes with a physical representation. Furthermore, that data is already contained in the schema.

Upgrades to the Draft version:

Step 4:

- Add a dynamically populated College dropdown in the search page.
- Implement Create and Read functionalities.
- Remove Author field from search page.
- Change search results page to show search parameters and course display.
- Implement login functionality.
- Update routes.

Step 3:

We did not make any changes based on changed design decisions.

Step 2:

First, we made our relationships more clearly formatted in our outline and noted the reciprocal versions of each of our relationships. We also implemented Primary Keys for each entity. Finally, we removed the College Name attribute from the Courses and Majors entities as that information was already contained by their relationships to the Colleges entity. We also removed the username attribute from the Reviews entity in lieu of the authorID foreign key.

Step 2 Peer Feedback

Criteria	Gavin Slusher	Jordan Colbeth	Chase Naples	Adam Pham
Does the overview describe what problem is to be solved by a website with DB back end? Does the overview list specific facts?	Yes, especially with good, specific facts on the scope of universities, number of classes and degrees. The problem statement might be slightly more overt, saying something like how course quality may be somewhat opaque from those outside of universities	Yes, the overview lists specific facts and it's made clear that this website & DB would help students with selecting courses for university.	Yes, the overview specifically describes a scenario involving students attending universities. The website with DB back end will allow students to use reviews to select which classes they want to take at their university. Yes, specific facts such as the number of students attending universities and how many classes they take on average each term are listed as a few example facts.	Yes - Overview describes the need for a DB to implement user reviews for classes. Overview lists specific facts of the requirements of the DB
Are at least four entities described and does each one represent a single idea to be stored a s a list?	Yes, Colleges, Courses, Majors, Authors, and Reviews all represent a single idea.	Yes, there are five entities and each one represents a single idea.	Yes, there are five entites which include Majors, Courses, Colleges, Reviews, and Authors.	Yes - College, Courses, Majors, Authors, and Reviews
Does the outline of entity details describe the purpose of each, list attribute datatypes and constraints and describe relationships between entities? Does the outline clearly indicate which entities (tables) will be implemented and which team member is primarily assigned to the associated page(s)?	Yes. I especially like the contraints on the number of chars for attributes that are strings, and the not NULL designations.	Yes, each entity has a thorough description, a list of attributes, and those attributes contain datatypes and constraints. One thing that could be a little confusing is that all of the entities are underlined - this threw me for a second because with ERD's and DB design we've learned that underlined attributes signify primary keys. This is clear in the doc though because they explicity say which attributes are PK's.	Yes, each entity is described and has a specific purpose, datatypes and constraints, and the specifc relationships between them are listed. The outline also clearly indicates which entities are designated to each team member by listing them in parenthesis. I had to infer this but am assuming that is how the implementation will be done.	Yes - Each entity is described with its purpose and attributes and is assigned to a team member
Are 1:M relationships correctly formulated? Is there at least one M:M relationship?	They are, and there is; though, the two many to many relationships haven't been broken down into 3 entities with many to one relationships yet. Presumably, the composite entities could be named College_Majors and College_Courses.	Yes, 1:M relationships are formulated (Courses/Reviews, Colleges/Courses, etc.) and there is at least one M:M relationship (Colleges/Majors, Majors/Courses).	Yes, the 1:M relationships are correctly formulated (College/Courses, Courses/Reviews for example). There is at least one M:M relationship (College/Majors)	Yes - The 1:M relationships (Colleges/Courses and Authors/Reviews) are correctly formulated. There is are two M:M relationships (Colleges/Majors and Majors/Courses). I would recommend describing the additional M:M relationship tables
Is there consistency in a) naming between overview and entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?	Yes. It looks like entities are capitalized and plural and attributes are camelCase.	Yes, there is consistency in the naming of the entities/attributes and they are properly pluralized and capitalized.	Yes, there is consistency in the naming between the overview and entities and their attirbutes. There is also consistency between both plural entities and singular attributes. Lastly, the use of capitalization for naming is also consistent.	Yes - naming between entities and attributes are consistent

Step 3 Peer Feedback

Criteria	Pamela Yin	Dominic Lupo	Matt Garber	Kaithlyn Morris
Does the UI utilize a SELECT for every table in the schema?	Yes. All tables are utilized in the UI. Search function uses majors, courses, colleges, and create review uses courses, majors, colleges, reviews, authors.	I think author is missing. Maybe add author to searched reviews? Profile and results both show data from major, college, course and reviews.	I believe so.	Yes, it appears to!
Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?	Yes. Search course filters the course ratings by major, college, and course.	I do not see any but dropdowns could be added to the course search form easily.	Assuming reviews when navigating to the profile page are filtered to your user then yes.	Yes, the use of the search function is clear on the homepage. The user may search/filter by course title, major, and college name.
Does the UI implement an INSERT for every table in the schema?	Yes. By creating a new review, input fields in every table and author table enables user to login to input all fields.	Yes authors use a traditional sign up form and then remaining 4 tables use the review form.	I believe so.	Yes, between the user registration form and the review form, everything is covered.
Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship?	Yes. Courses: Majors relationship is M:M and inserting rows in majors or courses will enable majors_courses to reference courseID and majorID as foreign keys.	Yes a logged in user creates a review which add ones fk and the remaining three are added to the form by the user.	I believe inserting a review will insert your user id.	Yes, an example of this is Courses:Majors
Is there at least one DELETE and does at least one DELETE remove things from a M:M relationship?	No. There is no delete shown from UI. May need to add it to the profile page next to edit button.	I do not see a delete button.	I don't see a way to delete.	Unfortunately, I do not see an instance of DELETE.
Is there at least one UPDATE for any one entity?	Yes, for every review written by the user, there is edit button. By editing each review, it will update attributes in the reviews entity.	Yes reviews can be edited.	Yes there is an edit button.	Yes, a user may edit data on their profile (change password) or edit existing reviews
Is at least one relationship NULLable?	From the outline, it looks like review comments are NULLable since it is the only entity that does not have NOT NULL. It is hard to tell from the UI is only front end without validation yet.	I think from the ER diagram the authroID fk in review should be nullable but I do not see it implemented. Maybe the team will implement this within the course edit form?	Potentially if reviews could have null in the comments attribute	I don't see how this is possible, unless a review can be posted anonymously without an author? Or maybe a course may not have a major?
Do you have any other suggestions for the team to help with their HTML UI?	It looks like the state attribute of colleges entity is not used in the UI. Probably need to take it out if not going to be used, or incorporate into UI when searching / creating review for college. I'd suggest to put in for practicality, since it is possible for two colleges in different states to have same name. I think overall your UI is very simple and easy to use. This is personal preference, but I would use modal box for sign up and actual web page for login (swap the signup and login). The name profile is somewhat misleading that I expected personal information, not list of ratings I (user) posted. I would rename it to something like My Reviews. As mentioned above, probably need a delete function/button to your UI. If your search function is requiring every single input, may be good idea to put validation into it so it would not go to next step if input is null. Same as write a review page. This is for your outline, but you have used an attribute named title for majors, courses, and reviews. It is confusing to keep track of and is not distinguishable between different entities. I would suggest using something like course_title, major_name, review_title.	Maybe add a directory page which has links and descriptions for every page. I think an efficient way to add a select query for author, a delete button and the nullable implementation would be the following: 1. Show author usernames on reviews. 2. Allow users to delete his or her profile. I think this would fit well with the theme of the website. If author A and author B reviewed 3 classes the same and A is wondering what class to take next, A would probaby trust B reviews to decide. If a user deleletes a profile the reviews are still valid, the user probably just wanted his or her name disassociated with them so just show nulls as anoymous.	I liked the styling and the ui a lot. Could add a search field for the filter requirement.	It's really clean and really easy to navigate! It looks really great!

Step 4 Peer Feedback

Criteria	Lauren Work	Dennie Devito	Daniel Brezavar	Josh Jones
Are the queries syntactically correct? Disregard the part where input will be substituted as shown in the sample_data_manipulation_queries.sql	The queries seem to be syntactically correct, and are very easy to follow what their purpose is.	Yes, they appear to be correct. I think you guys did a good job on this one.	Yes, all queries appeared to be syntactically correct. The queries in dmq.sql were very well formatted.	The queries appear to be syntactically correct
Are there queries providing all functionalities as required by the CS340 Project Guide ? What query is missing ? What needs to be fixed?	There doesn't appear to be insert options for Courses or Majors. If I am interpreting the guideline correctly, this one requires that the project must have inserts for these: "You need to include one DELETE and one UPDATE function in your website, for any one of the entities. In addition, it should be possible to add and remove things from at least one many-to-many relationship and it should be possible to add things to all relationships. This means you need INSERT functionality for all relationships as well as entities. And DELETE for at least one many-to-many relationship."	In my opinion, your group might not following the requirement fully yet. I said this because obviously I can see some queries inside your website and schema which fulfil some requirement of the CS340 Project Guide. However, you still have some "requirenment" that you still have to implemented inside . For example, I think you are missing a "DELETE query for one many-to-many relationship" requirement which is fine for now. Just remember to read the requirement again and fulfill everything in it!	There are insert queries for each table. There are queries that select from every table at least once. There are delete and update functionality. Authors and Reviews could be deleted so a one to many relationship could be set to null (also in erd). Courses were able to be deleted which satisfies the MM delete row criteria	Yes the queries seem to fulfill all the required functions in the project guide. That said there is a query that joins all the tables in the database, I know this has been advised against in multiple videos throughout the canvas materials so this functionality might wan to be split between different views.
Do the queries cover the relationships as required by the CS340 Project Guide?	See the answer to the previous question.	The same as above, while you guys do have implemented some queries which is right, I think you also missed some of the requirements. However, the ones that are provided in here are good already in my opinion.	Yes, queries set up at least 4 relationships and 1 MM relationship.	Yes all required relationships are demonstrated in the queries.
Is the SQL file syntactically correct? This can be easily verified by importing/copy pasting it in phpmyadmin.	The syntax appears correct and the file uploads to PHPMyAdmin and appears to produce the expected working database.	Yes, I tried to import it and it worked perfectly fine.	The SQL file did appear to be syntactically correct. I was able to import with myadmin successfully.	Yes it was correct I was able to import it without issues
Are the data types appropriate considering the description of the attribute in the database outline?	It may be worth increasing the character count option for College name, as many colleges are very long when not abbreviated, especially with state schools that start with University of *** and then also have in their name the city after. For example, Cal Berkeley is actually University of California Berkeley, which is 33 characters.	Yes, data types look appropriate	The data types appear to match the pdf they provided in their submission	Yes the data types seem to match the outline
Are the foreign keys correctly defined when compared to the Schema?	The foreign keys appear to match the schema, and have instructions for the database for ON UPDATE and ON DELETE that appear appropriate for each category.	Yes, I can clearly see that the foreign keys are correctly defined in the Schema, as well as the implementation of it.	Foreign keys appear to be handled appropriately based on the pdf they provided in their submission	Yes the foreign key definitions look to match the ERD in the outline
Are relationship tables present when compared to the ERD/Schema?	Yes, the two relationship tables for Colleges_Majors and Majors_Courses are both present. Just as a note, the given ER diagram does not have the M:M relationship between colleges and majors that the outline, schema, and sql files show.	Yes, the relationship tables are present.	Relationships in the ddq.sql file appear to be handled correctly and match the ERD/Schema	Yes relationship tables appear to be handled correctly.

Overview

About 19.9 million students attended colleges and universities in fall 2019 (source). A typical full-time student will take 4-6 classes per quarter. A database driven website will collect reviews for classes from students that have taken the class. It will also allow students to search for student reviews of classes. This website will include classes from all the colleges and universities in the US (approximately 5,300 colleges and universities), all classes for all degree programs offered by the school (range between approximately 20 to 1200+). The reviews will include a rating for class difficulty, a rating for the amount of work required by the class, a rating about how interesting the class was, and comments from the author about the class. This website will allow students to research classes before registering to better budget their time for the upcoming school period.
Scope: All Universities in the US, all classes in university, all degree programs.

Database Outline (updated)

- **Colleges (Alex):** Records the details of a given university
    - collegeID: int, auto_increment, unique, not NULL, PK
    - state: string, < 31 chars, not NULL
    - collegeName: string, < 255 chars, not NULL
    - **Relationships:**
        - A 1:M relationship between Colleges and Courses is implemented with collegeID as an FK inside of Courses.
        - A M:M relationship between Colleges and Majors is implemented as an additional relationship table

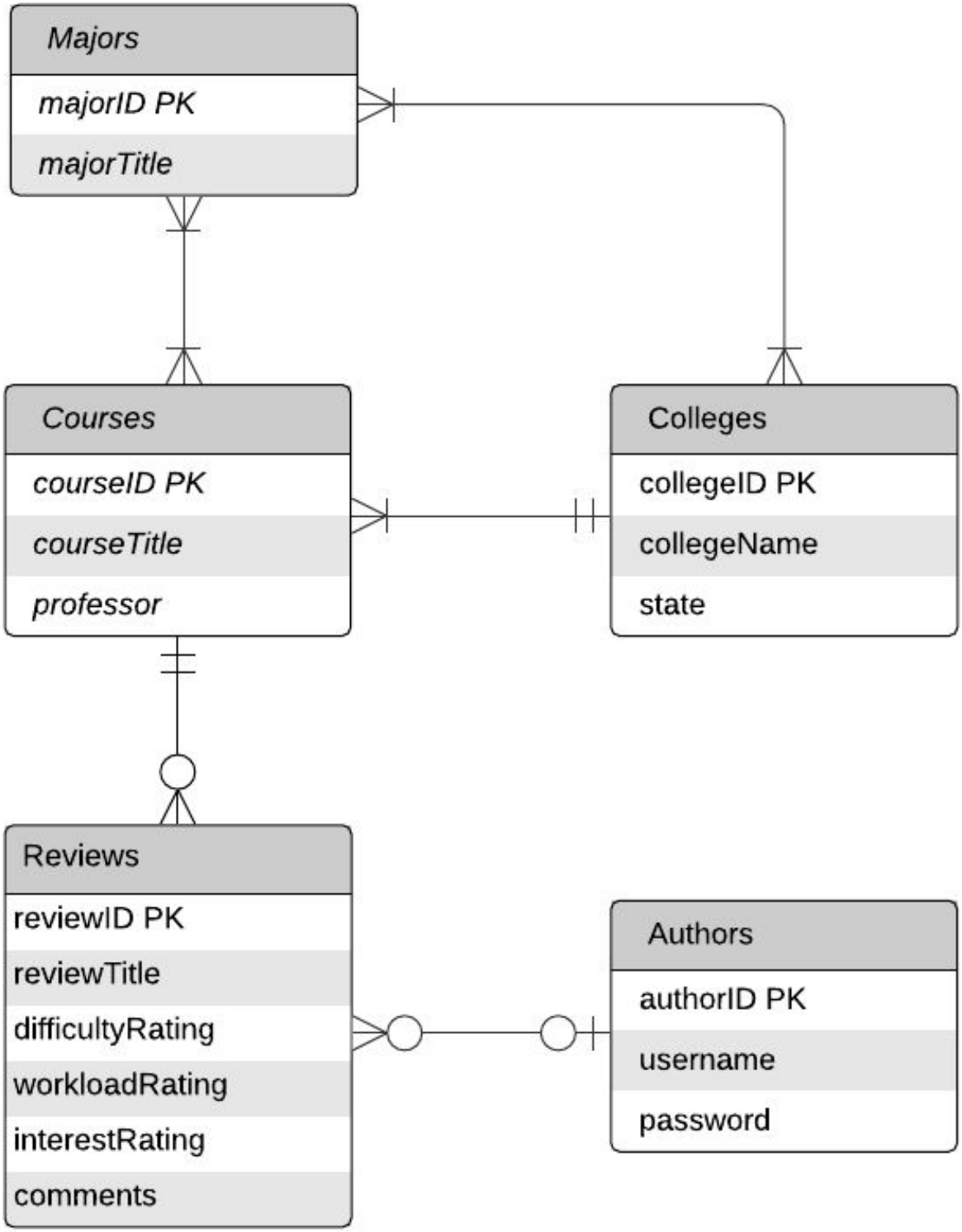
- **Courses (Alex):** Records the details of a university course
    - courseID: int, auto_increment, unique, not NULL, PK
    - collegeID: int, auto_increment, unique, not NULL, FK
    - courseTitle: string, < 51 chars, not NULL
    - professor: string, < 31 chars, not NULL
    - **Relationship:**
        - A 1:M relationship between Courses and Reviews is implemented with courseID as an FK inside of Reviews.
        - A 1:M relationship between Colleges and Courses is implemented with collegeID as an FK inside of Courses.
        - A M:M relationship between Majors and Courses is implemented as an additional relationship table.

- **Majors (Nick) :** A course of study with a number of required classes towards a degree
    - majorID: int, auto_increment, unique, not NULL, PK
    - majorTitle: string, < 51 chars, not NULL
    - collegeID: int, unique, not NULL, FK
    - **Relationships:**
        - A M:M relationship between Majors and Courses is implemented as an additional relationship table.
        - A M:M relationship between Colleges and Majors is implemented as an additional relationship table

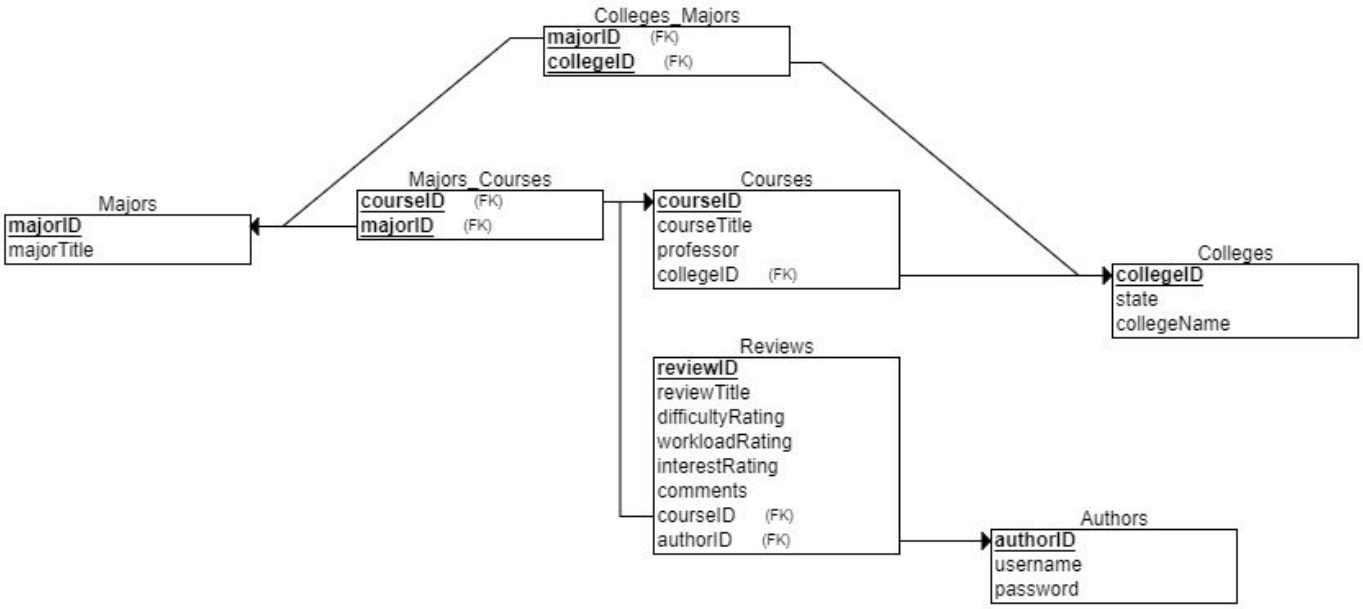
- **Authors (Nick) :** A user account that may leave reviews
    - authorID: int, auto_increment, unique, not NULL, PK
    - username: string, < 16 chars, unique, not NULL
    - password: string, < 21 chars, not NULL
    - **Relationships:**
        - A 1:M relationship between Authors and Reviews is implemented with authorID as an FK inside of Reviews

- **Reviews (Alex / Nick) :** A user-created review for a class
    - reviewID: int, auto_increment, unique, not NULL, PK
    - courseID: int, auto_increment, unique, not NULL, FK
    - authorID: int, auto_increment, unique, FK
    - reviewTitle: string, < 51 chars, not NULL
    - difficultyRating: int, 1-10, not NULL
    - workloadRating: int, 1-10, not NULL
    - interestRating: int, 1-10, not NULL
    - comments: string, < 501 chars
    - **Relationships:**
        - A 1:M relationship between Courses and Reviews is implemented with courseID as an FK inside of Reviews.
        - A 1:M relationship between Authors and Reviews is implemented with authorID as an FK inside of Reviews

Entity-Relationship Diagram



Schema


Colleges_Majors
majorID (FK)
collegeID (FK)
Majors
majorID
majorTitle
Majors_Courses
courseID (FK)
majorID (FK)
Courses
courseID
courseTitle
professor
collegeID (FK)
Colleges
collegeID
state
collegeName
Reviews
reviewID
reviewTitle
difficultyRating
workloadRating
interestRating
comments
courseID (FK)
authorID (FK)
Authors
authorID
username
password