CPSC 408

12/14/2018

Yixing Zheng

Final Project Report

Grading System For School Districts

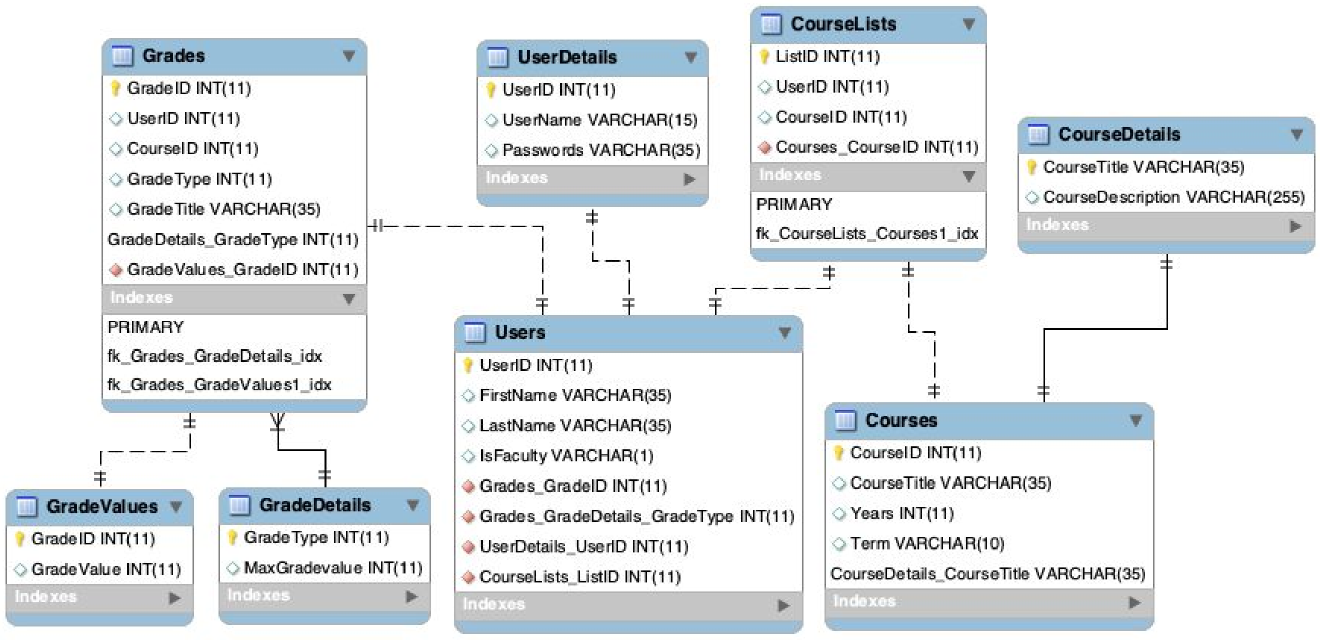
I. Introduction

The problem I was trying to solve was that schools need a system to stores all the grades for their students and teachers. It is clear that in this day and age, nobody wants to take 20-30 pieces of 1-2 pages quizzes to home and then the next morning bringing them back to work for every quiz there are going to have. And it would be extremely difficult if a teacher wanted to find a grade for a student; the teacher would need to dive into a mountain of papers just to find the one, so the student's question regarding the grade can be answered. In this manually searing process, there is nothing to rely on but checking each and every quiz there is. Moreover, if there is only a hard copy for both the teacher and the student, then it would be difficult for the student to study on it since the teacher would also need to archive the quiz for the record.

So, with a database system that stores everything in it and is easy for everyone with authorization to check on, all the problems discovered above are old news. There are some well-established businesses in the industry and they have been working on this kind of systems for quite a time. The most famous one, at the moment, is BlackBoard, which provides more than just storing the grades for schools, but also pushing notifications and online assignment system that allows students to submit their works through the Internet, which saves everybody's time and effort of trying to submitting and receiving a piece of paper by hands. There is another approach to it, that is creating rubrics for storing the grades and even performing evaluations based on the data stored. In a way, this approach seems more complicated since each rubric is an individual module and contains various types of data, as supposed to what we have seen in the Blackboard system, which is grades for each student organized by the courses that student is taking. Personally, if there were more time, I would want to study more on the rubric system. It interests me and I feel it would require more techniques on the database management.

I. My Solution

In a nutshell, I organized the grades of each course that each student is taking, so every grade is traceable to that student. The following is my schema:



As it states, the Users are at the center; a user could be either a teacher or a student. If there is user A, who is a teacher, then A would only see the menu for teachers, which contains more features, such as create, add, update, and delete for both courses and grades. There is also a print result feature that allows the teacher to check all the results that the database stores, just in case this teacher A needs to gain an insight of how other courses are going and how the students are doing in those courses. One interesting subject to study could be how the teachers grade their students based on the standard of grading. For example, there is the one standard for all teachers to grade certain subjects, say essays, one teacher might be more strict while another is more generous on giving points. This difference can be studied and therefore promotes an improvement on the document writing for the grading standards. And all of these would need to be based on an adequate amount of information that is provided to the teacher, or the management.

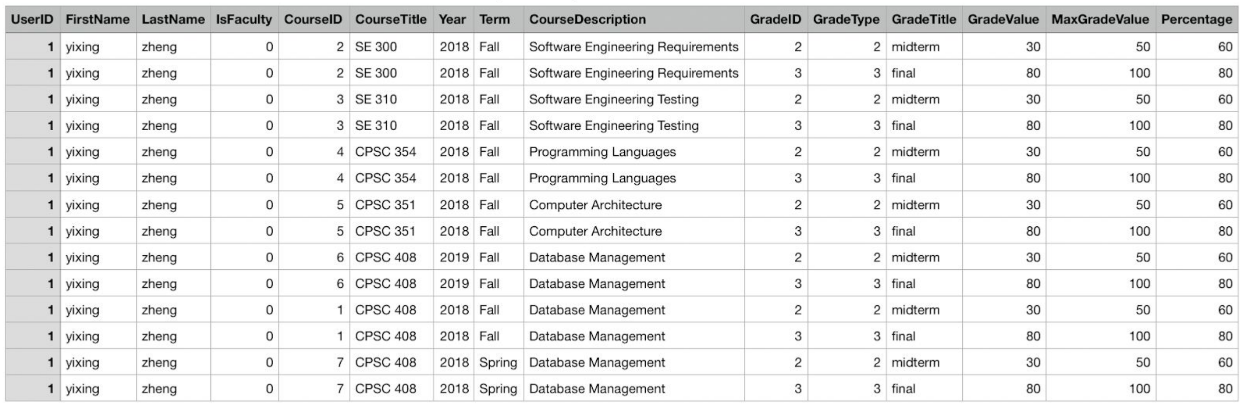
On the other hand, as a student user in the system, one can only add, delete, and check grades for the courses that one is taking. This is very much alike to the Blackboard's system since that is all it needs.

On the schema, the user is connected to the UserDetails table for the confidential, such as username and password. The user is also connected to the grades directly, so there is a unique grade id that aligns with the user's id and ensures that each grade is unique. The Grades table has two relations with the GradeValues table and GradeDetails table that specify the actual values of grades, types of grades and their maximum values, respectively. On the other hand, the user is connected to the courses through the table CourseLists, which matches the user and the courses they are taking. So, we can find all the information about a course, id, title, year, term, and description from the Courses table and CourseDetails table and match this information up with what we have in the Users table and CourseLists table; then, through the relation to the Grades table, we have all the information we need for a student's grade for a certain course.

As for the frameworks and algorithms, I did not really include any. I was planning to export the data into R and do some analysis there, then import the results back for displaying in command lines, or even better, a light web application. However, I never had a chance to implement it. I also tried to find a package for exporting to CSV, then found out it could be done by simply using the FileWriter. The only fancy bit of the code was the password validation, which contains some regular expressions to filter the weak inputs and thus enhance security. I knew that as a wheel, it must have been there for quite a time, so I just found it and put it on without trying to reinvent the wheel myself; and that was one of the most important things I have learned recently, to find and use the wheel.

Therefore, technically, my project was basically writing queries in Java and having it communicate with the database. Sending a query, receiving results, checking conditions, sending another query for inserting to table or selecting and displaying results were most of the project. I did find it interesting since I had to think hard to figure out the relationships between tables and sometimes join three to four tables to display the result. And everything in the result just comes down the following picture:

presumably all the data in the world



UserID	FirstName	LastName	IsFaculty	CourseID	CourseTitle	Year	Term	CourseDescription	GradeID	GradeType	GradeTitle	GradeValue	MaxGradeValue	Percentage
1	yixing	zheng	0	2	SE 300	2018	Fall	Software Engineering Requirements	2	2	midterm	30	50	60
1	yixing	zheng	0	2	SE 300	2018	Fall	Software Engineering Requirements	3	3	final	80	100	80
1	yixing	zheng	0	3	SE 310	2018	Fall	Software Engineering Testing	2	2	midterm	30	50	60
1	yixing	zheng	0	3	SE 310	2018	Fall	Software Engineering Testing	3	3	final	80	100	80
1	yixing	zheng	0	4	CPSC 354	2018	Fall	Programming Languages	2	2	midterm	30	50	60
1	yixing	zheng	0	4	CPSC 354	2018	Fall	Programming Languages	3	3	final	80	100	80
1	yixing	zheng	0	5	CPSC 351	2018	Fall	Computer Architecture	2	2	midterm	30	50	60
1	yixing	zheng	0	5	CPSC 351	2018	Fall	Computer Architecture	3	3	final	80	100	80
1	yixing	zheng	0	6	CPSC 408	2019	Fall	Database Management	2	2	midterm	30	50	60
1	yixing	zheng	0	6	CPSC 408	2019	Fall	Database Management	3	3	final	80	100	80
1	yixing	zheng	0	1	CPSC 408	2018	Fall	Database Management	2	2	midterm	30	50	60
1	yixing	zheng	0	1	CPSC 408	2018	Fall	Database Management	3	3	final	80	100	80
1	yixing	zheng	0	7	CPSC 408	2018	Spring	Database Management	2	2	midterm	30	50	60
1	yixing	zheng	0	7	CPSC 408	2018	Spring	Database Management	3	3	final	80	100	80

III. Conclusion

I genuinely enjoy doing this project since it is just so much fun. And I am sure it is obvious that there is a lot of room for improvement. The log was one of the things that could have been done, but thankfully the transaction still works well with commits and rollbacks, so the issue with invalid inputs is resolved. Another tricky thing was to only display the course that a user is not taking; I have tried a few ways, but it did not work out. If there can only be one final thought, it would be that I wish I started earlier and spent more time on it. But, to be fair, for what it took, it turned out to be better than I thought.

References:

CSV export,
https://www.daniweb.com/programming/software-development/threads/493294/export-data-from-mysql-db-to-csv-file-using-java#

Password validation,
https://stackoverflow.com/questions/36097097/password-validate-8-digits-contains-upper-lowercase-and-a-special-character