CS 411 SU19 Group 7 Final Report
SEQUEL HELP

1. Our project helps students practice SQL problems by breaking down their queries into smaller steps, and displaying the output tables from each step. This can guide students by helping them visualize what each step of their query does so they can adjust accordingly if their final result does not match the expected result. The number of steps is determined by their method of querying, which is described later in the advanced function. From the instructor side, we have a question bank manager, where they can modify, add or delete questions for students to practice and a student manager where they can filter and search the students that are signed up to the website.

2. Discuss the usefulness of your project, i.e. what real problem you solved.
Due to the natural abstraction of SQL queries, beginners may have a hard time visualizing their queries, and not able to understand their final output. Our project serves to break complicated queries into smaller steps, whose result will be displayed accordingly. After brainstorming through various ideas for our advanced functionality, we drew inspiration from the idea of the step by step solutions to math problems from Symbolab and Wolfram Alpha. This closely resonated with our struggles with homework 1 on PrairieLearn, where we struggled to break down questions into smaller steps.

For instance:
SELECT Name
FROM Employees
WHERE Salary > 1000;
This simple query can indeed be broken down into multiple steps:

- SELECT *
  FROM EMPLOYEES;

- SELECT *
  FROM EMPLOYEES
  WHERE Salary > 1000;

And Finally:

- SELECT Name
  FROM EMPLOYEES

```
    WHERE Salary > 1000
```

3. Discuss the data in your database
We have two databases, one to store the user details and the other one to store all the tables which will be used to solve the question. For the user database(sequelhelp_users), we created 2 tables, one named user which stores the details of every signed up user for the application with their basic information such as Name, email, password, year and Major. This information will be helpful for the instructor to search through the students(users in this case) using the application and get their basic info. The second table in the user database is the current_question, which stores the user_id and the question_id. This keeps track of what question the user/student is working on.
For the other database (sequelhelp_mock), we have 8 tables that store the data extracted from the first homework assignment. These are Students, Enrollments, Courses, Customers, Purchases, Products and Brands. For some of these tables we added some extra mock entries to increase the size of the data. Other than these 7 tables, we also created a Questions table which stores the QuestionId, Question (as given in our Prarie learn assignment) and the AnswerQuery(the correct answer for the corresponding question as posted on the course wiki page).

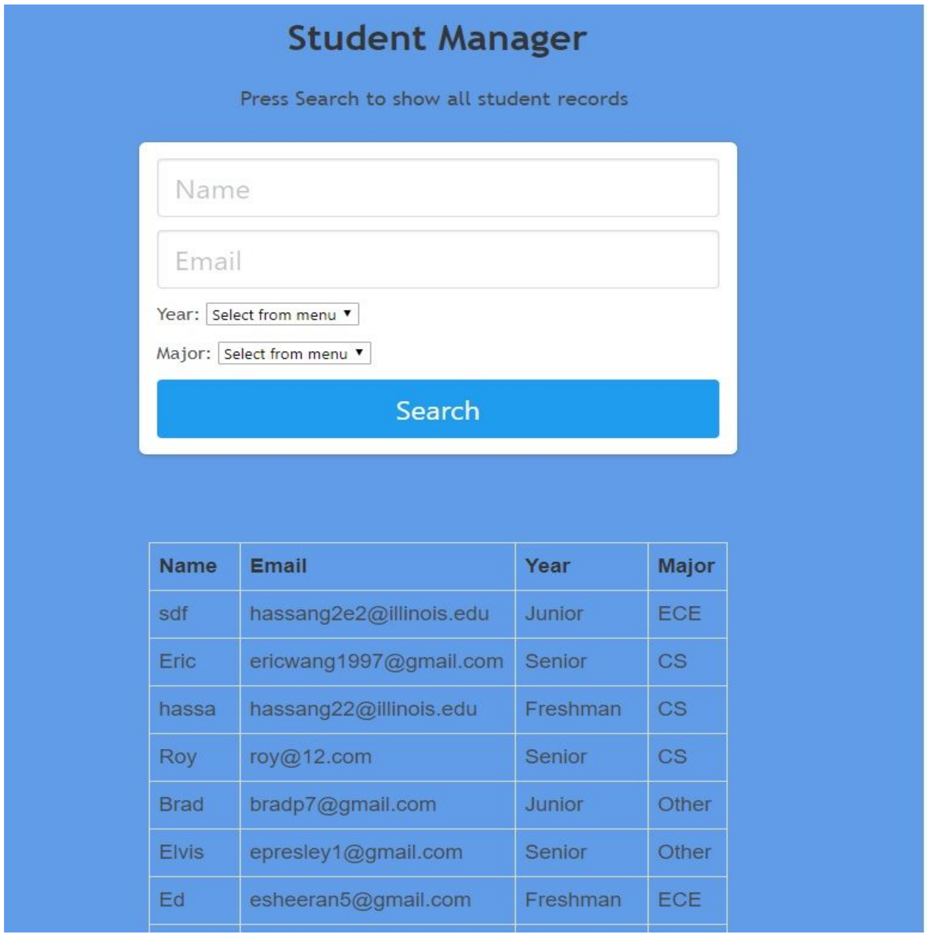
5. Briefly discuss from where you collected data and how you did it
Since our primary goal was to create a functionality which takes in an SQL query and breaks it down into smaller logical steps that make up the query, we needed data that we were familiar with, so that we could predict the accuracy of our basic and advanced functions. Hence, as a group we decided that the best source of data for our project would be our first homework assignment on SQL. In order to collect this data, we ran queries on the prairelearn environment to display all the data for each schema. This data was then collected and edited in excel to convert it to a proper table and eventually to a CSV file. For some of these tables, we added extra mock entries to make the size optimal. Once these CSV files were created, we imported them to phpmyadmin and created tables through them.

6. Clearly list the functionality of your application (feature specs)
Sequel helper is an application that we created to help out students creating sql queries to break down their queries into smaller logical steps which are easier to comprehend than the whole query at once. This can be comparable to the step by step solution that can be found on wolfram alpha for arithmetic computation. We have created a website where there are pages to view profile, login, signUp, Logout, Parser, Students, Question Bank and Student Manager.
Using signup, a student can create their account application, which will provide them access to the parser. The Parser is our advanced functionality, which has the question and the schema on

the top, a text box after that for user to input their query. Once the student inputs the query and submits it, the parser goes through the query and displays a step by step implementation of the query. The student can also move to the next question using the next button. For basic functionality, the question bank page provides a table with all the questions in the database with their corresponding answers. This is made such that the instructor can delete or update each entry using the buttons on the side. There is also an option to add a new entry to the query in the form of a new question and answer. The student manager page lets the instructor search through the user database and display the user information using filters such as name, email, year and major.

7. One of the basic functions we implemented was the “search database and print results” in the Student Manager page for the instructor. The purpose of this functionality is to help the instructor find all the students in the database that matched the desired filter attributes. These attributes include name, email, year, and major from which the instructor can choose what to filter by. If the filters are empty, it will search all student records by default. The entries consists of fill-in text boxes or drop down menus which the information from each attribute is sent to a python script to be concatenated in the WHERE clause in the sql query. The sql query will pull data from the student records database, sends the information back to the interface to be

displayed in a table as shown below.



Student Manager

Press Search to show all student records

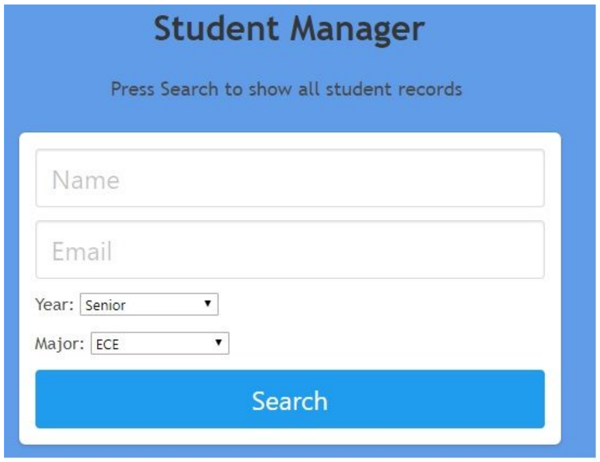
Name	Email	Year	Major
sdf	hassang2e2@illinois.edu	Junior	ECE
Eric	ericwang1997@gmail.com	Senior	CS
hassa	hassang22@illinois.edu	Freshman	CS
Roy	roy@12.com	Senior	CS
Brad	bradp7@gmail.com	Junior	Other
Elvis	epresley1@gmail.com	Senior	Other
Ed	esheeran5@gmail.com	Freshman	ECE

8. Code snippet of the search function:

```
 def get_search(student_name, student_email, student_year,
student_major):
    queryresult = 'SELECT name, email, year, major FROM user '
    selection = ''
    if student_name:
        selection += "AND name = '" + student_name + "' "
    if student_email:
        selection += "AND email = '" + student_email + "' "
    if student_year != 'any':
        selection += ("AND year = '" + student_year + "' ")
    if student_major != 'any1':
        selection += ("AND major = '" + student_major + "' ")
    if len(selection) > 0:
        queryresult += 'WHERE ' + selection[4: -1]
    results = run_query(queryresult, 'sequelhelp_users')
    ret = []
    for r in results:
        ret.append({"name": r[0], "email": r[1], "year": r[2],
"major": r[3]})

    return ret
```

For example, if the instructor wants to search for all students that are Seniors and ECE they would fill in the drop-down menu respectively:



Year: Senior
Major: ECE

When sent to the python script, the sql query would be:

```
SELECT name, email, year, major FROM user WHERE year = “Senior”
AND major = “ECE”
```

Which is sent to the database to search for the appropriate results to be displayed on the table.

9. As mentioned previously in 7 and 8, the dataflow is as followed:

1. User (student or instructor) inputs information into the forms/textbox from the browser.
2. The information is sent to the backend where the python script processes the information via flask and interacts with the database if necessary.
3. Once the information is processed, the result is sent back to the browser to be displayed.

10. Eric pls explain

11. Eric pls explain

12. Originally we were going to build our website using WordPress

13. Overall, our group was successful in collaborating with each other to create this project. We frequently met up as a whole team for long periods of time to assist with each other's responsibilities. Mrinroy collected all the data and created the databases necessary for our project to connect to the front-end interface. Steven was in charge of implementing the Student Manager functionality and providing recommendations and assistance towards other functionalities and their relevance to the databases we created. Eric was in charge of developing the advanced function for the parser with the assistance from everyone. Hassan was in charge of creating the Question Bank with add/delete/update functionalities as well as user registration/login. Since we were not familiar with web development, we all had to help each other to create the web interface.