Jonathan Prindle

CSPB 3287

Git_link: https://github.com/Jprindle264/CSPB3287_Project

School Management System Project

Original Motivation

Overall, I am going to create a school management database system. I have taught in four different states, at various grade levels, and in various content areas, and the one thing they all have in common is poor digital infrastructure. Three out of the four years I taught in public ed schedules were created last minute, rooms were assigned days before school started, and adding new students was a complete nightmare. My main motivation is to create a more functional school database management system. In addition, I would like to explore the unique issues that come along with creating a school management system from scratch.

Database Multiple Relations/Sample Table Ideas

This has been revised from the original proposal phase. Given the amount of implementation and design going into this project, I decided to reduce some of the tables from the original proposal phase. Additionally, I have decided to make this management system model more of an Elementary school format as opposed to Secondary school. For the following T represents Table and A represents the attributes of that table.

1) T: Teachers A: Teacher_ID, Teacher_First, Teacher_Last, Teacher_Salary, Room_Assignment
2) T: Students A: Student_ID, Student_First, Student_Last, Grade_Level, Room_Assignment
3) T: Staff A: Staff_ID, Staff_First, Staff_Last, Position, Staff_Pay_Rate
4) T: Room Assignment A: Room_Number, Grade Level, Grade Section
5) T: Student Fees A: Student_ID, Fee Name, Current_balance

These tables and relations will allow provide a large overview of the internal mechanisms of a basic school management database system. Dummy data was inputted to give a base for the systems. The rest of the data is up to user input (i.e add/delete/update students/teachers/staff/rooms).

Original Learning Outcomes

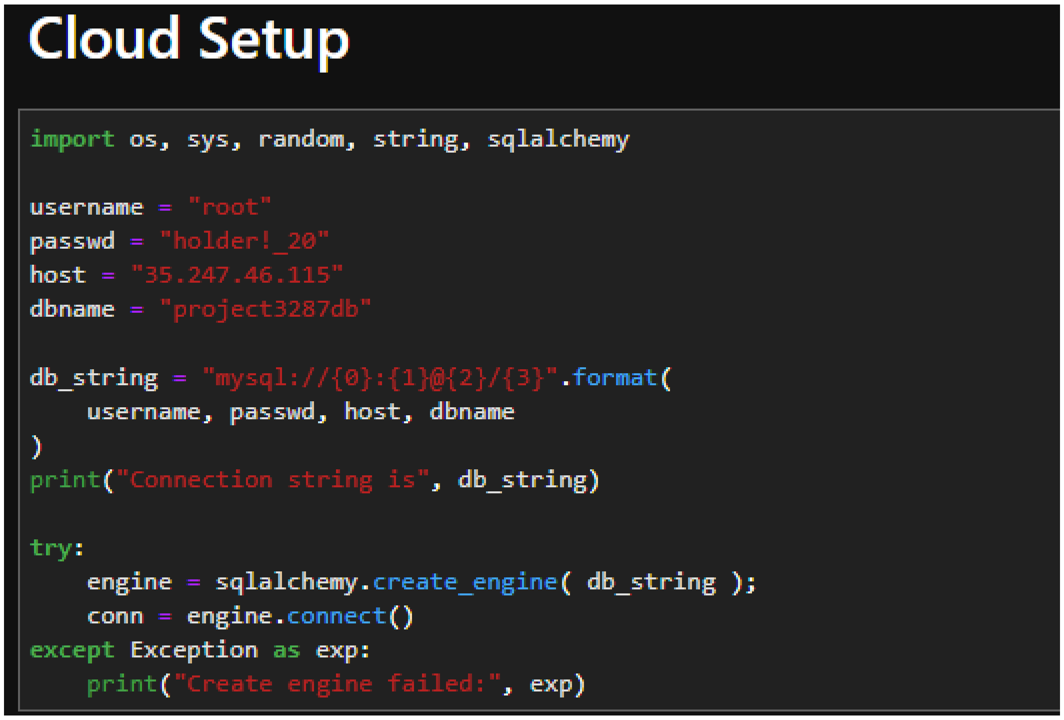
The current implementation and design allow me to meet all of my original learning outcomes, which were…
1) Getting more experience with relational database design and implementation.
2) Create a system from scratch from beginning to end.
3) Gain a better understanding of school management systems.

4) Gain a better understanding of various tools involved in DBMS Overall, I hope to understand databases and their management better.

Overview of Project

The first major component of this project was setting up an instance via google cloud and hosting the database of that instance. After that, it is simply a matter of creating and connecting to an interface.

The picture below showcases the basic steps taken to connect to the database for the school management system. First, we connect via main user root, enter our password, host, and database, then using sqlalchemy we create an engine that we can use to interact with the database. Now we're connected.

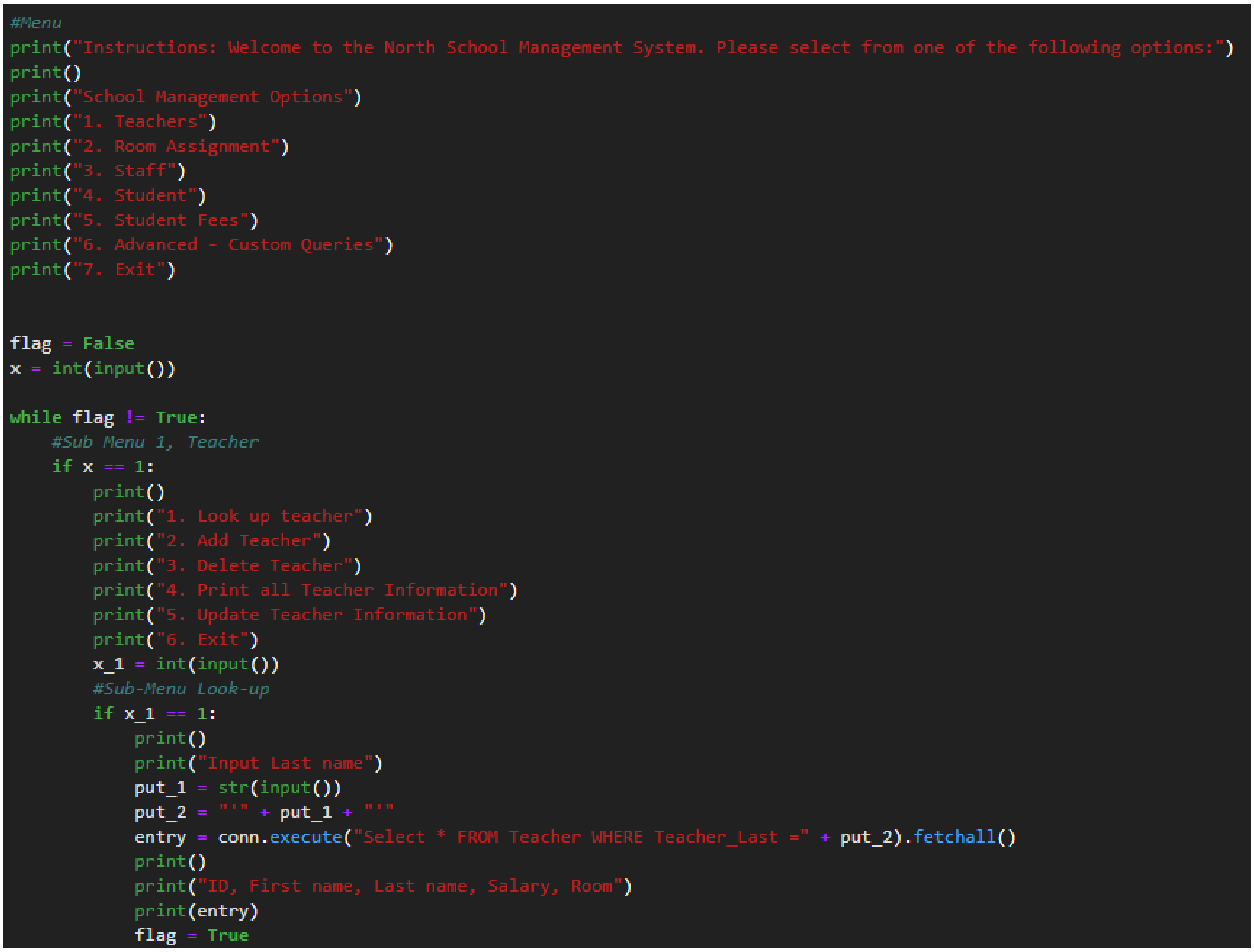


Cloud Setup

```
import os, sys, random, string, sqlalchemy

username = "root"
passwd = "holder!_20"
host = "35.247.46.115"
dbname = "project3287db"

db_string = "mysql://{0}:{1}@{2}/{3}".format(
    username, passwd, host, dbname
)
print("Connection string is", db_string)

try:
    engine = sqlalchemy.create_engine( db_string );
    conn = engine.connect()
except Exception as exp:
    print("Create engine failed:", exp)
```

After that, some dummy data is fed into the system so we can interact with. Typically, the data for a school management system would be entered in by a secretary or registrar. Depending on the school/district you may have someone else input this data.

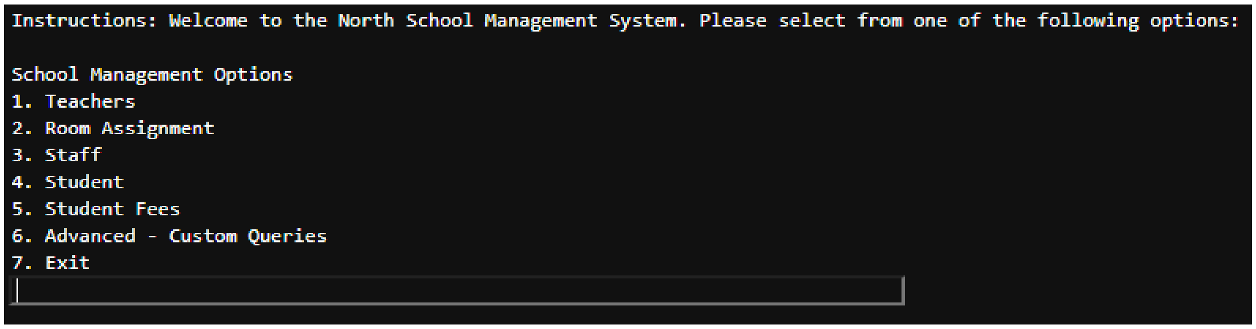
I created a user interface using a series of flags, inputs, and conditional statements via python. This is a basic interface but serves its purposes for the scope of this project. The menu allows users to essentially alter the database depending on their needs. A user can add/delete/update/print out information for various sections. Additionally, if a user has the skills, they are able to generate custom queries into the database. Below is a snippet of the code used to construct the menu.



```python
#Menu
print("Instructions: Welcome to the North School Management System. Please select from one of the following options:")
print()
print("School Management Options")
print("1. Teachers")
print("2. Room Assignment")
print("3. Staff")
print("4. Student")
print("5. Student Fees")
print("6. Advanced - Custom Queries")
print("7. Exit")


flag = False
x = int(input())

while flag != True:
    #Sub Menu 1, Teacher
    if x == 1:
        print()
        print("1. Look up teacher")
        print("2. Add Teacher")
        print("3. Delete Teacher")
        print("4. Print all Teacher Information")
        print("5. Update Teacher Information")
        print("6. Exit")
        x_1 = int(input())
        #Sub-Menu Look-up
        if x_1 == 1:
            print()
            print("Input Last name")
            put_1 = str(input())
            put_2 = "'" + put_1 + "'"
            entry = conn.execute("Select * FROM Teacher WHERE Teacher_Last =" + put_2).fetchall()
            print()
            print("ID, First name, Last name, Salary, Room")
            print(entry)
            flag = True
```

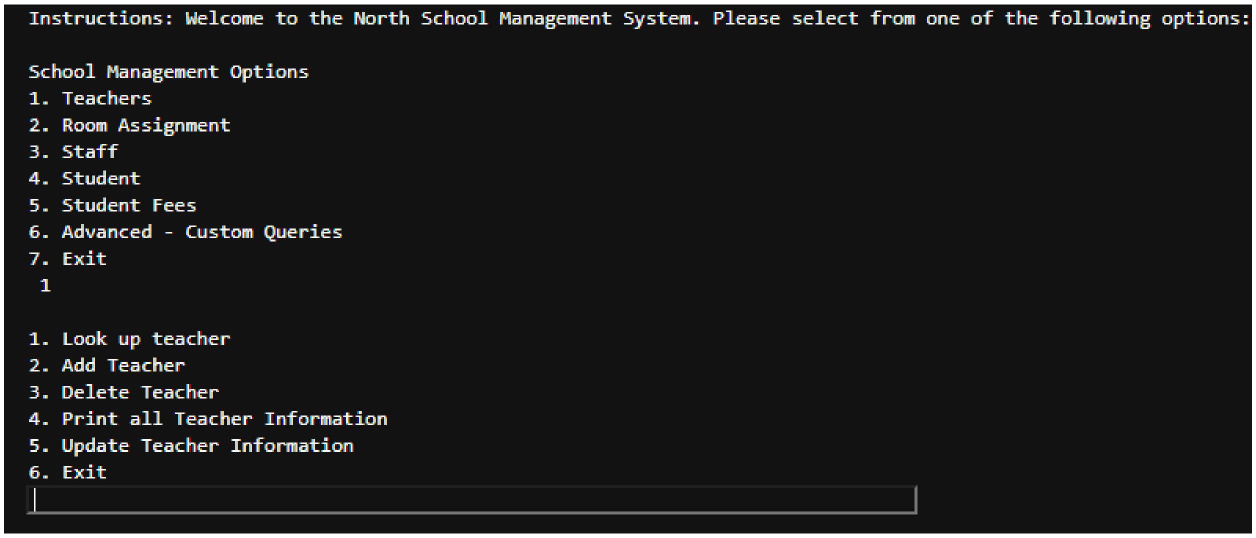
The overall construction of the menu is relatively simple. Using a while loop with a flag, you provide a series of options through conditional statements. After a statement is executed, the flag is changed from false to true thus exiting the while loop. This is a relatively simple implementation, but gets the job done.

This implementation leads to the following interface.



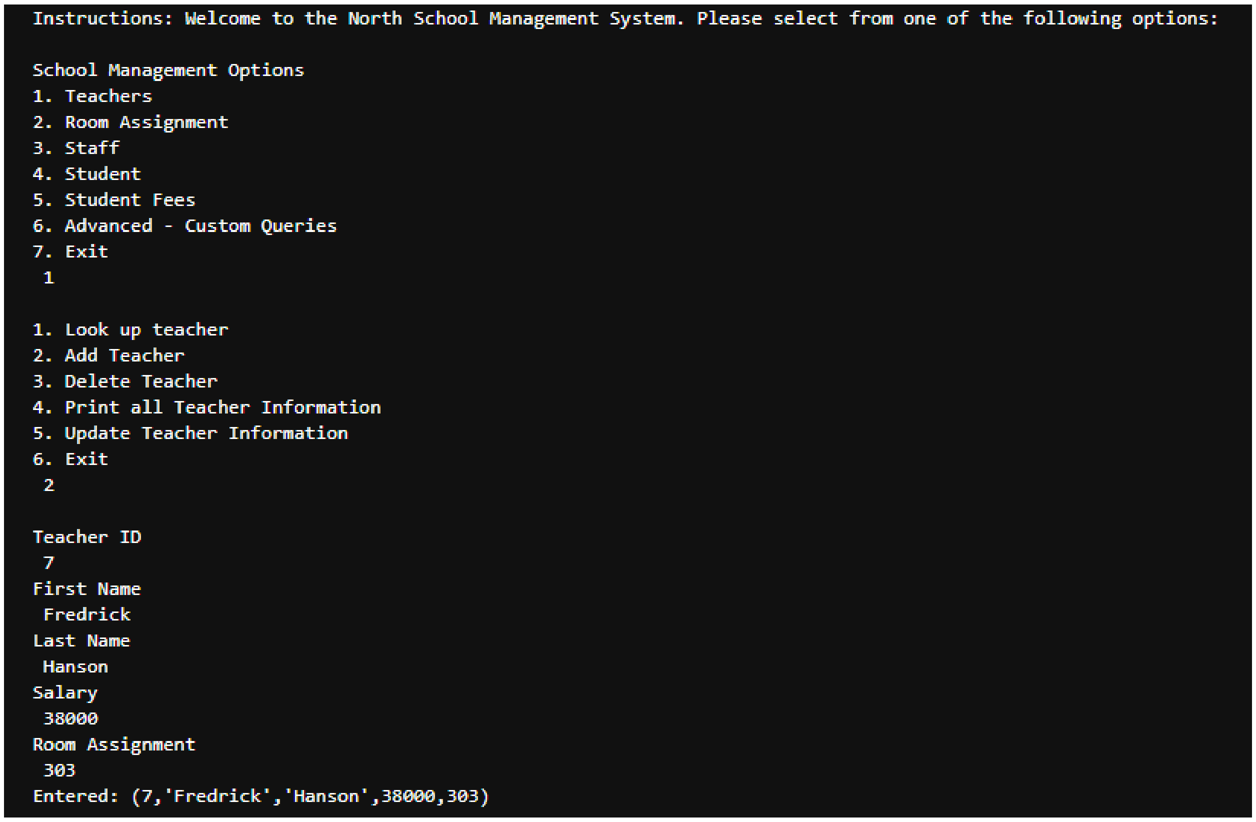
```
Instructions: Welcome to the North School Management System. Please select from one of the following options:

School Management Options
1. Teachers
2. Room Assignment
3. Staff
4. Student
5. Student Fees
6. Advanced - Custom Queries
7. Exit
```

The interface is simple to interact with. Just enter in the proper number for the option you wish to select. This then leads to a sub-menu with various options for that section.



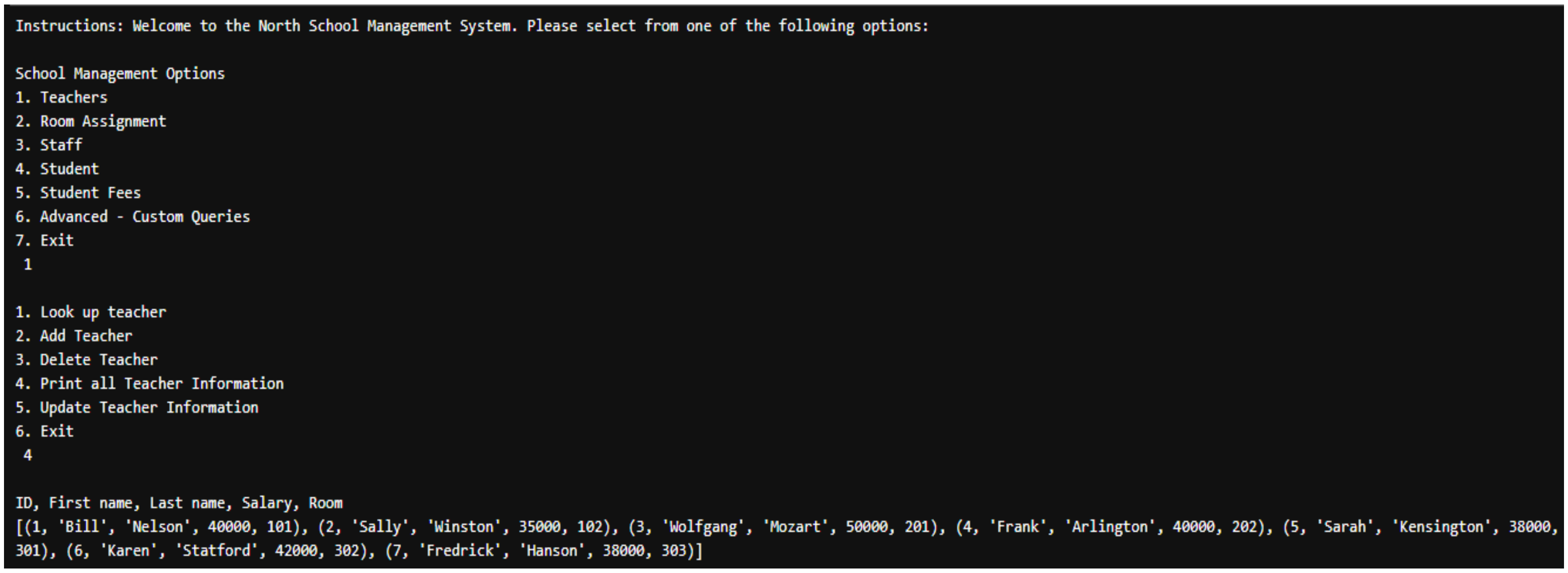
```
Instructions: Welcome to the North School Management System. Please select from one of the following options:

School Management Options
1. Teachers
2. Room Assignment
3. Staff
4. Student
5. Student Fees
6. Advanced - Custom Queries
7. Exit
 1

1. Look up teacher
2. Add Teacher
3. Delete Teacher
4. Print all Teacher Information
5. Update Teacher Information
6. Exit
```

Select an option in the sub-menu, then fill in the values for each box to complete the transaction.



```
Instructions: Welcome to the North School Management System. Please select from one of the following options:

School Management Options
1. Teachers
2. Room Assignment
3. Staff
4. Student
5. Student Fees
6. Advanced - Custom Queries
7. Exit
 1

1. Look up teacher
2. Add Teacher
3. Delete Teacher
4. Print all Teacher Information
5. Update Teacher Information
6. Exit
 2

Teacher ID
 7
First Name
 Fredrick
Last Name
 Hanson
Salary
 38000
Room Assignment
 303
Entered: (7,'Fredrick','Hanson',38000,303)
```

The new entry has now been added to the database. At this point you can double check to see if the addition was actually made.

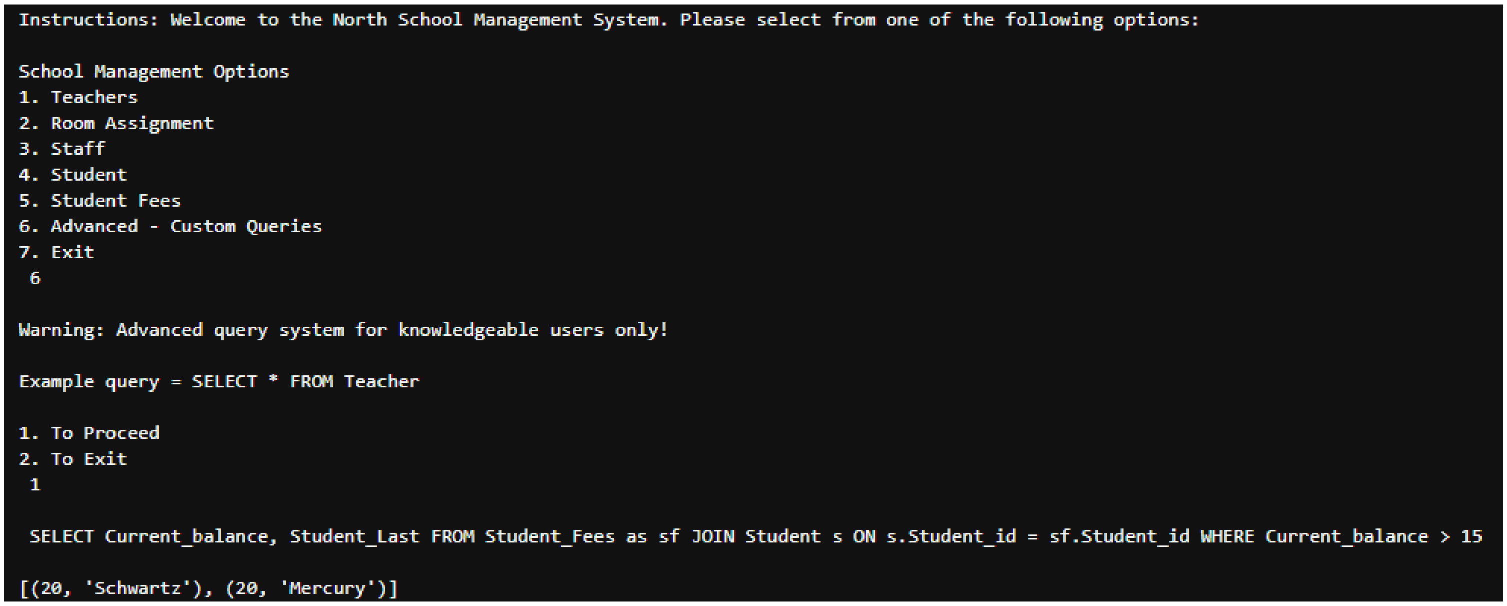


```
Instructions: Welcome to the North School Management System. Please select from one of the following options:

School Management Options
1. Teachers
2. Room Assignment
3. Staff
4. Student
5. Student Fees
6. Advanced - Custom Queries
7. Exit
 1

1. Look up teacher
2. Add Teacher
3. Delete Teacher
4. Print all Teacher Information
5. Update Teacher Information
6. Exit
 4

ID, First name, Last name, Salary, Room
[(1, 'Bill', 'Nelson', 40000, 101), (2, 'Sally', 'Winston', 35000, 102), (3, 'Wolfgang', 'Mozart', 50000, 201), (4, 'Frank', 'Arlington', 40000, 202), (5, 'Sarah', 'Kensington', 38000,
301), (6, 'Karen', 'Statford', 42000, 302), (7, 'Fredrick', 'Hanson', 38000, 303)]
```

If we look at the last entry, we see 7, Fredrick, Hanson, 38000, 303, has correctly been entered.

Finally, in order to offer more functionality to this interface, I have also included the ability to run custom queries. This allows someone who has the technical ability to search the database with more ease and to generate more complex queries.

The query below searches for the last names of students who have fees in excess of $15.



```
Instructions: Welcome to the North School Management System. Please select from one of the following options:

School Management Options
1. Teachers
2. Room Assignment
3. Staff
4. Student
5. Student Fees
6. Advanced - Custom Queries
7. Exit
 6

Warning: Advanced query system for knowledgeable users only!

Example query = SELECT * FROM Teacher

1. To Proceed
2. To Exit
 1

 SELECT Current_balance, Student_Last FROM Student_Fees as sf JOIN Student s ON s.Student_id = sf.Student_id WHERE Current_balance > 15

[(20, 'Schwartz'), (20, 'Mercury')]
```

Overall, the interface offers a variety of options and functionality for a typical user and additional functionality for advanced users.

Were Learning Outcomes Met?

1) Throughout the course of this project I was able to learn more about designing and implementing database systems in general. I found that creating a comprehensive system with an online user interface takes much longer then what I had originally expected. Additionally, there were various features that seemed relevant for future iterations. For a first iteration, however, I feel like this design provides a solid foundation for a school management system.
2) I was able to create the entirety of this system from scratch. In fairness I used the template from Lab four as a starting point but found myself deviating quite a bit from that original implementation. In particular, creating a user-friendly query system for users without any querying knowledge proved to be challenging but informing.
3) Throughout the course of this project I realized how big/how much of a time commitment a full-blown school management system would be. I also started to realize that an independent school management system would probably be integrated into a district wide management system thus making the overall management system something beyond that scope of what this project could create. It was a good learning experience in understanding the need to truly define the overall parameters of a project before beginning. This design suffices for a simple school management system, but there is room to add much more functionality as well.
4) Throughout the course of this project, I had issues with google cloud and MySql Workbench, and a few integration issues with the jupyter notebook. It was a good learning experience mess around and figure out the various obstacles on the road to the final product. I had a better understanding of how google houses the database system and how MySQL actually engineers the tables. I gained a bit more understanding of certain networking principles as well. One of the biggest takeaways was finding a way to integrate queries so that an everyday average user could input data.

Overall, I’m satisfied that I’ve met the goals I set out meet in this project. Your everyday user can easily add/delete/update or query information. Your advanced user can look up more specific information if they need it. The system lets you know who is who and where everyone is at. As mentioned before, I believe this is a solid start to a school management system.