Project Phase 3

link to video: <https://youtu.be/ZdACN027F9Y>

My program is essentially a Hospital Database. It contains most of the entities that are involved in the workings of a doctor/patient relationship. The entities are: Hospital, Doctor, Patient, Lab, and Medicine. I chose to omit the intricacies of Insurance within my database because I felt that it would make it too complex. The main idea behind it was to capture most of the most important relationships that exist within the hospital complex and keep track of them. I chose to do my program in python in PyCharm using SQLite.

The main menu of the program consists of 10 options. Options 1-9 are tables within my database and option 10 is the option to quit the program. If you choose an option 1-9 it will take you to a sub-menu for that specific table. These sub-menus essentially provide the queries that you can chose to do within each table.

Here are all the queries, where they are located in the menu and what topics they cover as well as their functionality. It should be noted that after all inserts, deletes and updates, the entire table is printed to the console so you can see the changes have occurred.

**Hospital sub\_menu:**

1. Insert a hospital to the database: Covers how to insert into a table

INSERT INTO Hospital ( Hname, Haddress)

VALUES("Grey-Sloan", "124 hospitalDr");

1. Delete hospital with less than 5/10 rating or any rating supplied.covers how to delete

DELETE FROM Hospital

WHERE rating < 5

1. Delete hospital based on name: //Also Covers how to delete

DELETE FROM Hospital

WHERE Hname=’{a}’.Format(a=input1)

1. Returns the Hospital name and count of the hospital that has billed the most patients. This query covers how to join tables and also has a group by clause

SELECT Hname, count(\*)

FROM Hospital h, Patient p

WHERE p.Billed\_by = h.billing\_id

GROUP BY h.billing\_id

HAVING count(\*) = (SELECT max(mycount)

FROM (SELECT count(\*) mycount FROM Patient GROUP BY Billed\_by) a);

**Doctor submenu**

1. Inserting a new doctor into the Database //covers insert

INSERT INTO Doctor (Lname, Years\_experiance, EmployeeID, EmployedBy)

VALUES("Smith", 18, 12345678, "Grey-Sloan")

1. Allows user to input a preferred hospital name and returns a list of Doctors who have at least 5 years of experience who work at that hospital along with their specialty. (In this example the hospital name is “Grey-Sloan”). This useful because for many operation you need a doctor with certain years of experience and you need to know if there is one available at a hospital near you. That’s why can choose the name of the hospital you need the doctor to be at as well as their minimum years of experience

SELECT Lname, Employed\_by, Specialty

FROM Doctor

WHERE YEARS\_EXPERIANCE >5 AND EmployedBy = “Grey-Sloan”

1. Get the Average Salary for each Specialty within a specific hospital. It utilizes the group by clause

SELECT Specialty, AVG(Salary)

FROM Doctor

Group by Specialty

Having EmployedBy = “Grey-Sloan”

1. Delete doctors who have less than 2 years’ experience // deleting specific items

DELETE FROM Doctor WHERE Years\_experiance > 2;

**Patient menu**

1. Ouputs the name of patients along with how many medications they take. This query joins two tables to generate the report as well as has a group by clause. It’s useful because it can be important for a doctor to know how many medications a patient is already taking before they can prescribe them any new ones.

SELECT P\_lastName, count(\*)

From Patient p, Takes t

WHERE p.SSN = t.P\_SSN

GROUP BY P\_lastName

1. Add patient //inserting into table

INSERT into Patient(P\_lastName, SSN, Age, Billed\_by)

VALUES ("al", "56111111", 23, 13)

1. Get number of patient who are less than 50 years old at each hospital. Returns the hospital name and number of patients less than 50 years old. This joins two tables as well as has a group by clause. It’s important mostly for data collection, to see the demographics that each hospital is treating in regards to age.

SELECT Hname ,count(\*)

FROM Patient p, Hospital h

WHERE p.Billed\_by = h.billing\_id and p.age<50

GROUP by h.Hname

**Medicine menu**

1. Insert medicine //inserts

INSERT into Medicine(Med\_ID, Price, Mname)

VALUES ("907", 900.90, "mud")

1. Change price of all medicine above a certain price. This is updating values based on a entered price. This query lets you lower the price of medicines which are above a certain price limit, and it lets you lower them by however much you want (There is a check in place to make sure the price stays positive).

UPDATE Medicine SET PRICE = PRICE – {b} WHERE PRICE >{a}.format(a=input1, b=input2)

1. Delete a Certain medicine //Deleting

Delete From medicine

Where MedID = 907

**Lab Table**

1. Insert Lab // Inserting

INSERT INTO Lab (Lab\_ID, Lab\_name)

VALUES (420420, "eww")

1. Delete Lab //Deleting

Delete from Lab

Where lab\_id =420420

1. Update lab name //Updating

UPDATE Lab

SET Lab\_name = “coolio”

Where Lab\_Id = 420420

**Prescribes menu**

1. Add prescription //Inserting

INSERT INTO Prescribes(Prescibed\_By, Medicine\_ID)

VALUES(11111111,907)

1. Find the most prescribed medicine. Returns the medicine name and the total number of times it has been prescribed. This query has multiple things. It joins tables, it has nested queries and has a group by clause as well.

SELECT Mname, count(\*)

FROM Medicine m, Prescribes p

WHERE p.Medicine\_ID=m.MedID

GROUP BY m.MedID

HAVING count(\*) = (SELECT max(mycount)

FROM (SELECT count(\*) mycount FROM Prescribes GROUP BY Medicine\_ID) a);

1. Returns names of all the medicines along with how many Doctors have prescribed each of them. Joins two tables and has a group by clause. This query is useful just for statistics to find out how many times each medicine is being prescribed. Because for example if several medicines aren’t being prescribed at all then the hospital should probably just stop carrying them.

SELECT Mname, count(\*)

from Medicine m, Prescribes p

WHERE p.Medicine\_ID=m.MedID

GROUP BY m.MedID

**Manufactures menu**

1. Add manufacturing // uses insert

INSERT INTO Manufactures (Lab\_Number, Med\_ID)

VALUES (420, 907)

1. Get the number of different medicines each lab manufactures. Returns lab name and number of medicines that they make. This query joins two tables as well as has a group by clause. It’s useful because a hospital usually needs to know who is suppling what medications.

SELECT Lab\_name ,count(\*)

from Lab l, Manufactures m

WHERE l.Lab\_ID=m.Lab\_Number

GROUP BY m.Lab\_Number

**` 3.** Returns the name and phone number and address of all the labs who make a certain medicine (ex: all labs that make the medicine named “mud”). This uses a nested query as well as joins two tables and has a Group by clause. It’s relevant because it lets a hospital search for a specific medicine name and returns which labs make it so if they need to place an order they have all the information they need.

SELECT Lab\_name ,Lab\_Phone, Address

from Lab l, Manufactures m

WHERE l.Lab\_ID=m.Lab\_Number AND m.Med\_ID = (SELECT MedID

FROM Medicine

WHERE Mname ="mud")

GROUP BY m.Lab\_Number

**Examines Table:**

1. Insert //insert

INSERT into Examines(Doctor\_ID, Patient\_SSN)

VALUES(12345699,123456789)

1. Tells you how many patients each doctor has and where they work. Essentially shows how much business each doctor is bringing into the hospital. This query uses a group by clause and joins two tables.

SELECT Lname, EmployedBy, count(\*)

FROM Doctor d, Examines e

WHERE e.Doctor\_ID = d.EmployeeID

GROUP BY d.EmployeeID

1. Outputs the name of the doctors who have more than the average number of patients, how many patients they have and where they work. Shows the most popular doctors so hospitals know which doctors are bringing in the most business. This can be useful in deciding things like salary raises. This query uses two tables as well as has a GROUP BY clause. It also has a nested query and has a “having” clause.

SELECT Lname, count(\*), EmployedBy

From Doctor d, Examines e

WHERE e.Doctor\_ID=d.EmployeeID

Group by d.EmployeeID HAVING count(\*)> (SELECT avg(mycount)FROM (SELECT count(\*) mycount FROM Examines GROUP by Doctor\_ID));

**TAKES menu**

1. Insert //insert

INSERT INTO Takes (P\_SSN, M\_ID)

VALUES(541111111,67)

1. Gives you the name, social security number and phone number of patients who are allergic to a certain medicine. This uses a nested query and joins two tables. It’s useful because you can enter the name of a specific medicine and get all the information about the patients who are allergic to it. This prevents doctors from prescribing a medication to a patient who has already shown negative symptoms to it.

SELECT P\_lastName, SSN, Phone\_nuimber

FROM Patient p, Takes t

WHERE p.SSN = t.P\_SSN AND t.Symptoms="Y" AND t.M\_ID =(SELECT MedID FROM Medicine WHERE Mname = "muud")

1. Delete //deletes

DELETE From Takes

WHERE P\_SSN =123456789