**FA23: DATA-225 Sec 11 - Db Systems for Analytics**

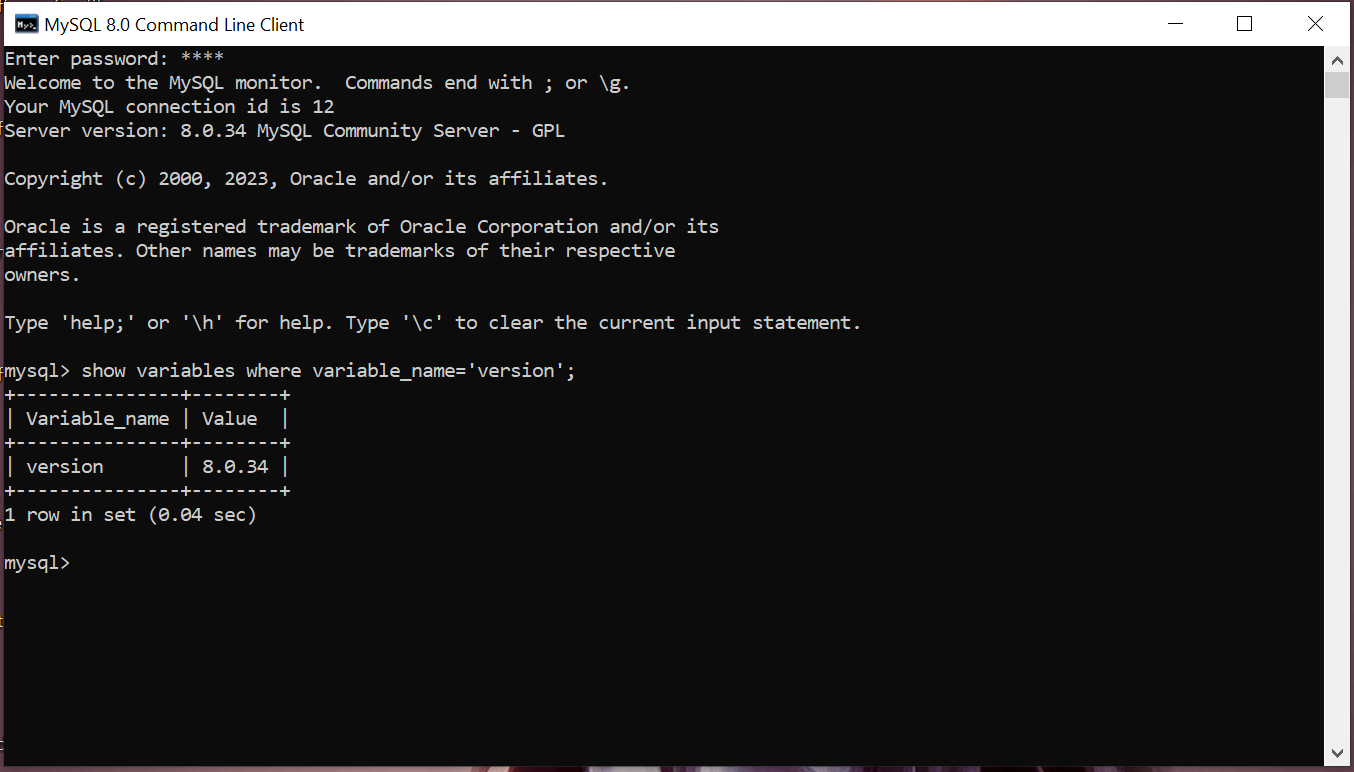
**Homework - - 1**

**Name :- Prayag Nikul Purani**

**SJSU Id :- 017416737**

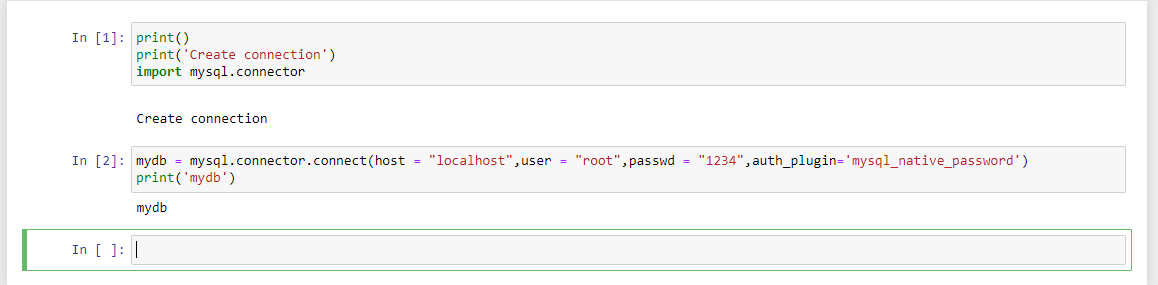
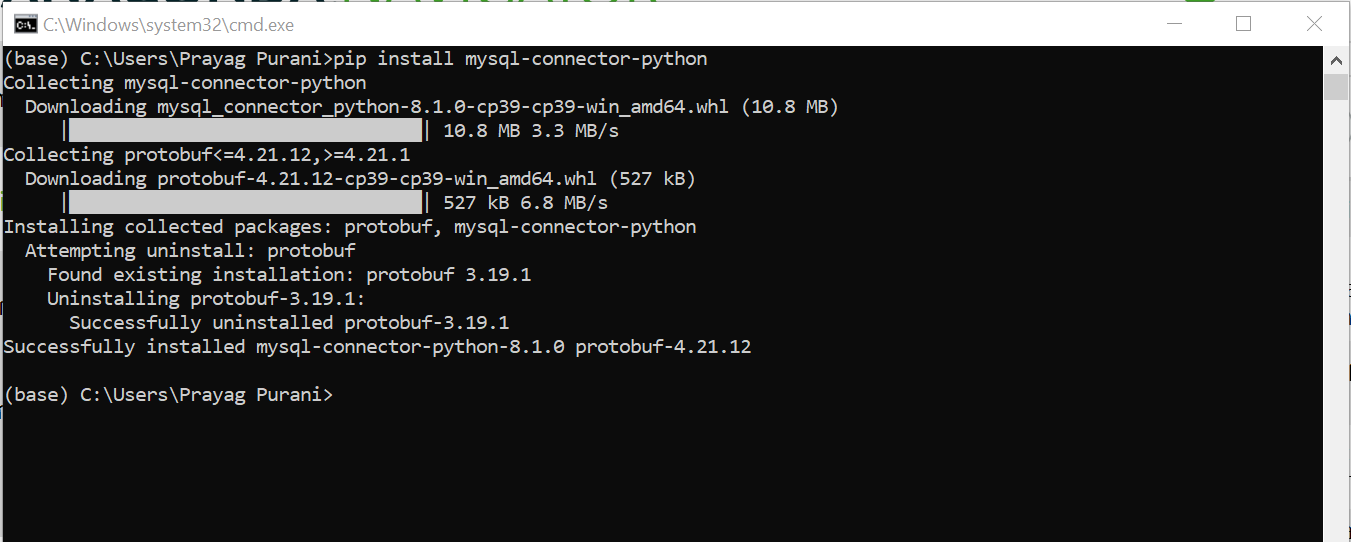
**Question 1a:-**

MySQL Server is already installed on my system.



Now we will be installing MySQL Connector/Python

Pip install mysql-connector-python



So, the mysql connector is working perfectly in our system.

**Question 1b:**

<https://www.kaggle.com/code/shariful07/student-mental-health-data-analysis/input>

According to me the Timestamp is not required so we can remove it.

The data was not in #NF as it had partial and transitive as there were no unique identifier so we if we go with the primary as (gender,course and study year) so it had to be spliced in to two tables and they are as shown. And is have added primary key as the id so to make it more simple and easy computation.

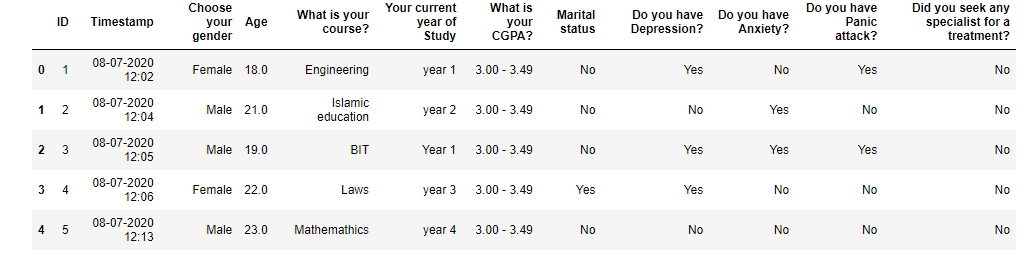


Table 1 =>

Create table Student(

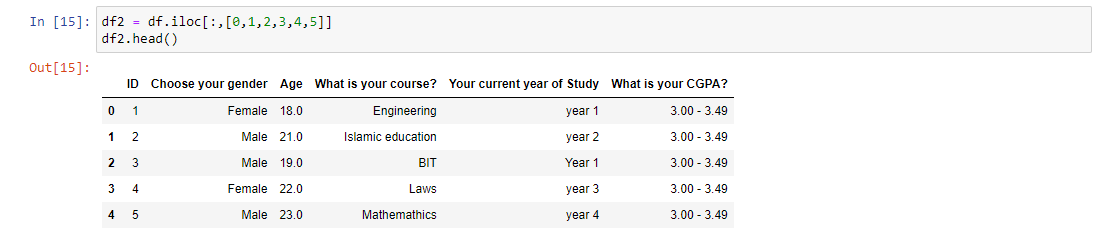
ID int Not Null,

Gender varchar(6) Not Null,

Course varchar(20) Not Null,

CGPA varchar(10) Not Null,

Constraint stud\_pk primary key(ID))



Create table Stud\_info (

ID int Not Null,

Marital\_Status varchar(3),

Depression varchar(3),

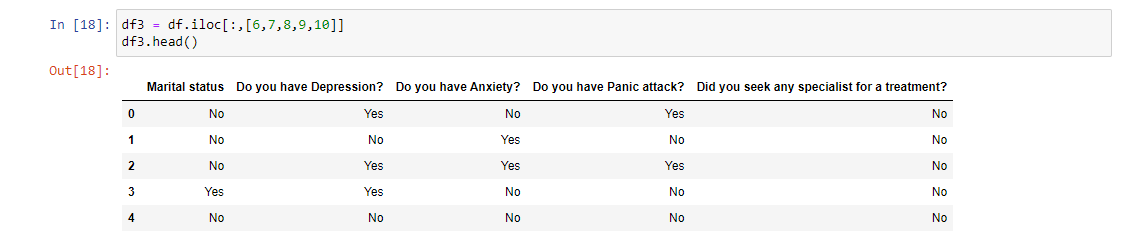
Anxiety varchar(3),

Panic varchar(3),

Treatment varchar(3),

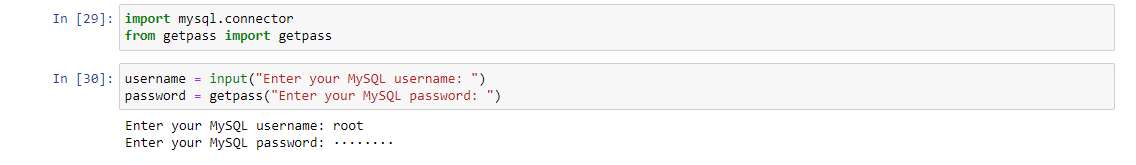
Constraint info\_pk primary key (ID),

Constraint foreign key (ID) references Student(ID))



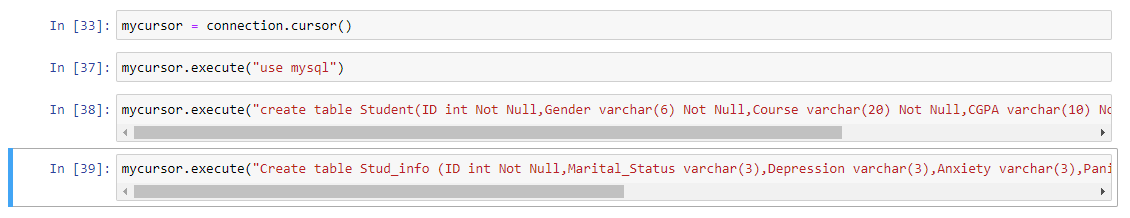
**Question 1c:**

First creating connection:

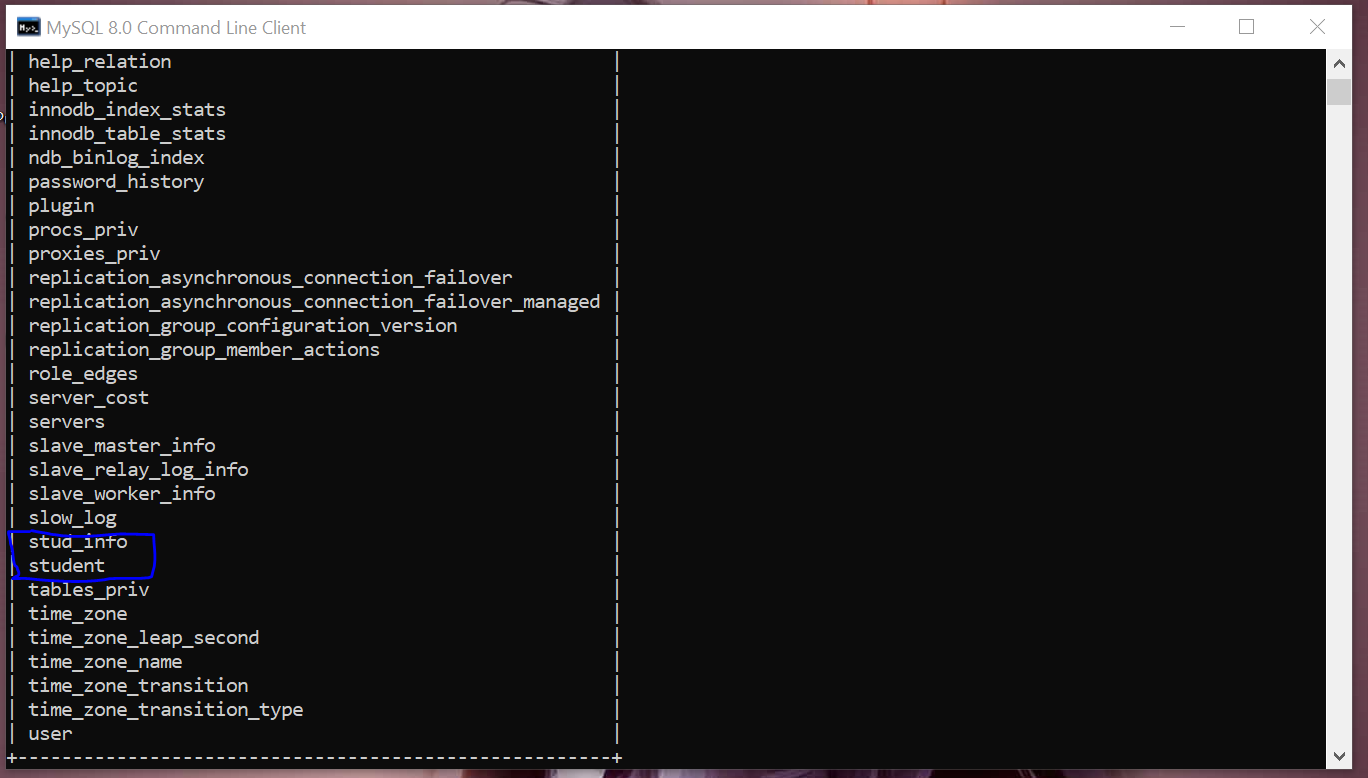




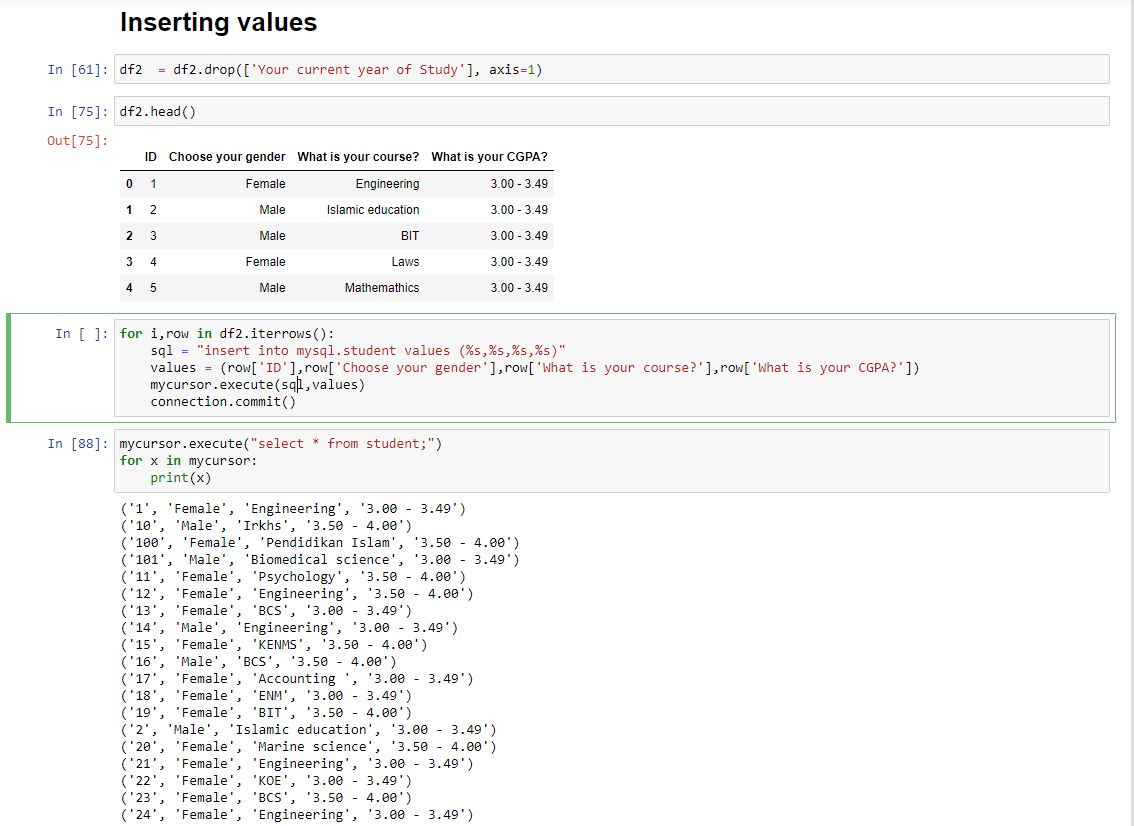
After creating the connection, we need to created cursor and the creation of tables.

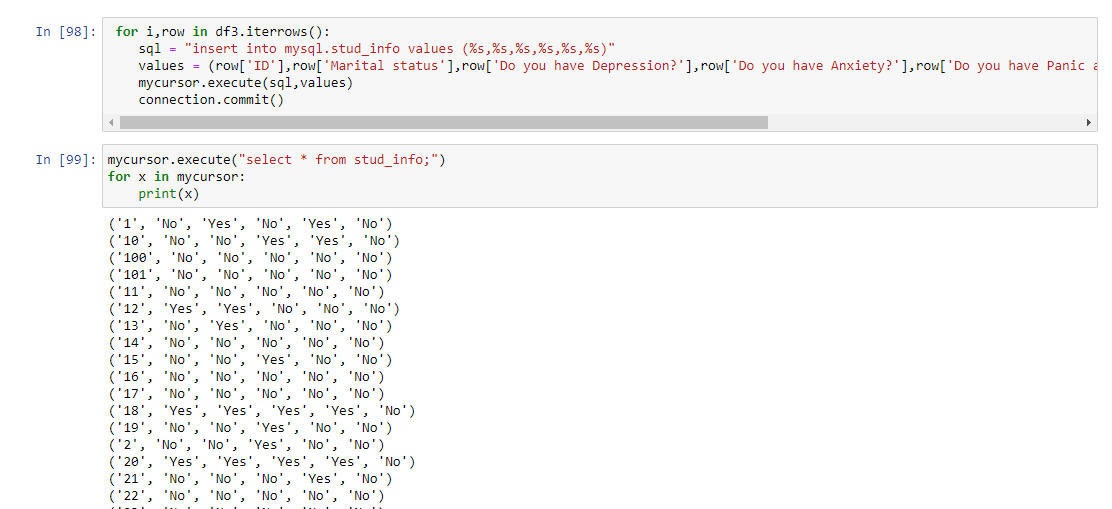


Successfully created the tables and we can check in the MySQL too.

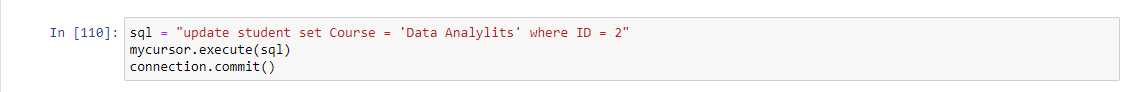


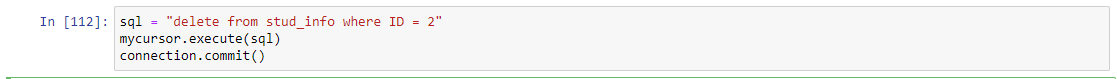
Inserting & reading from the tables:



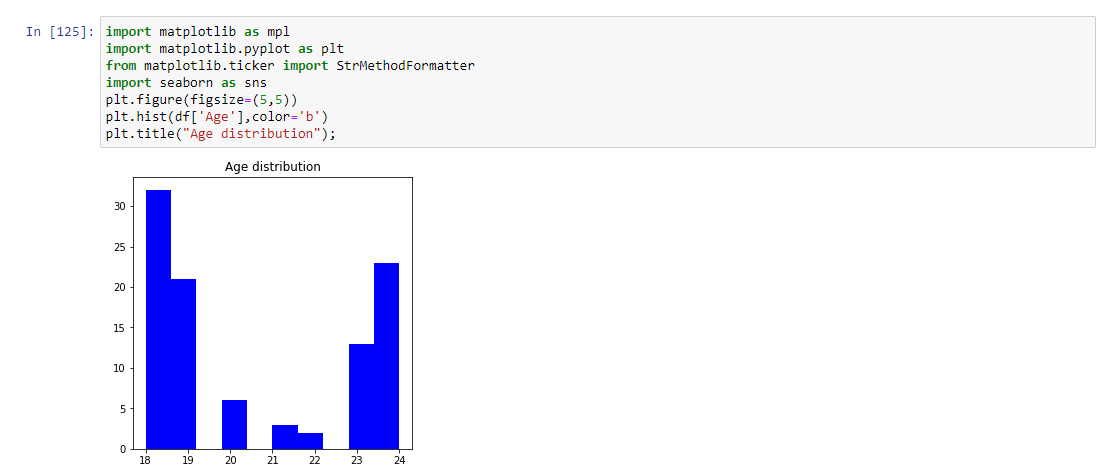


Updating and deleting:

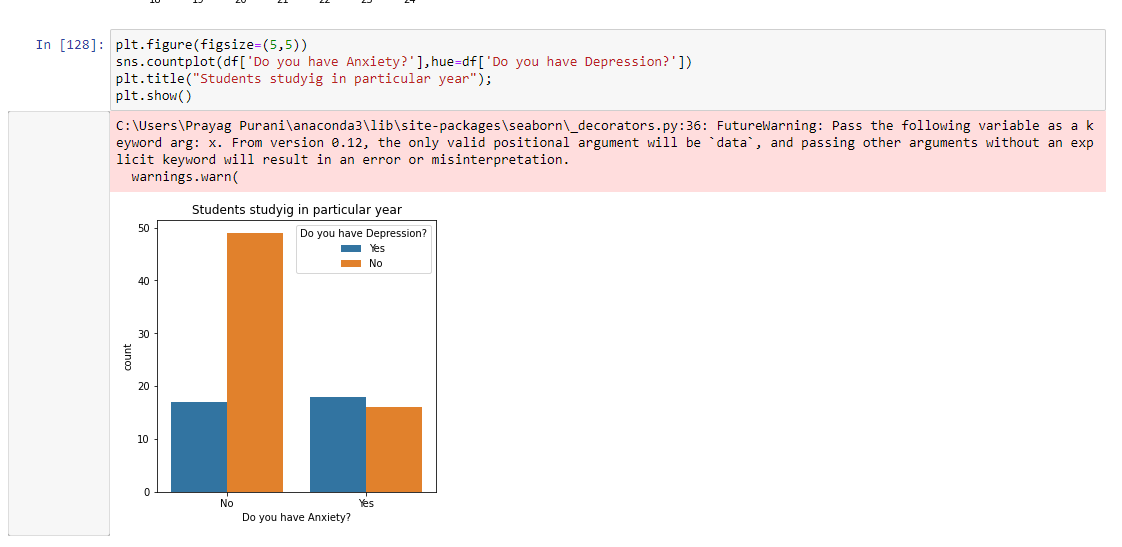




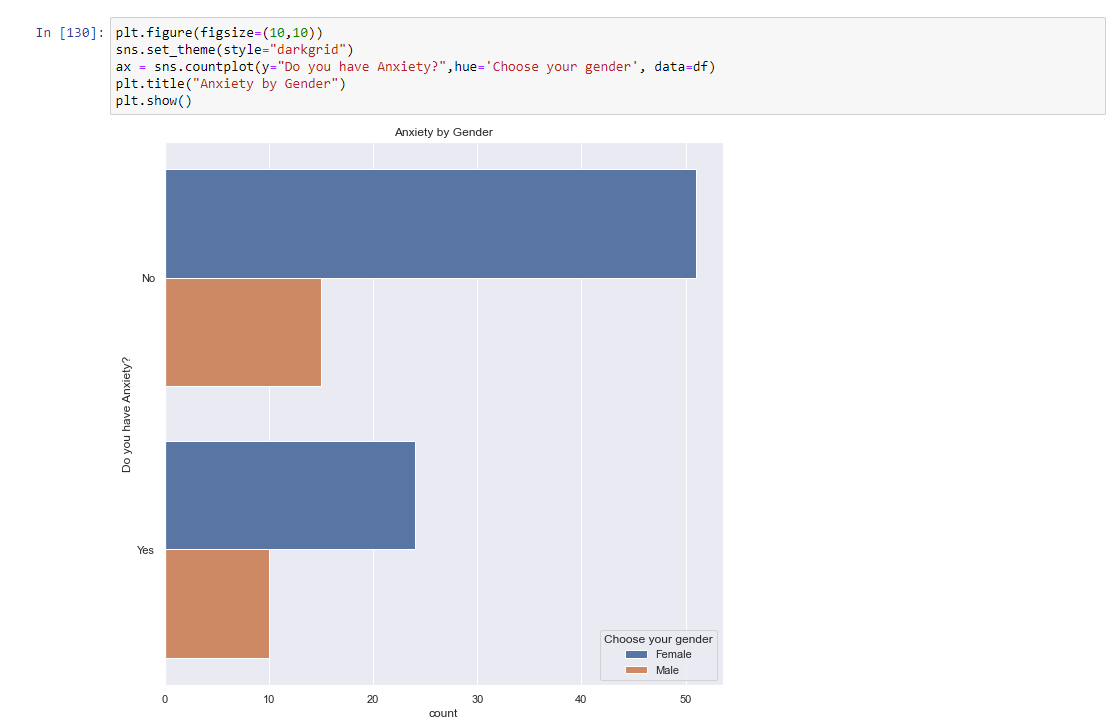
**Question 1d:**

****

1. **There are many people between 18-19**
2. **After that its between 23-24**

****

1. **This tells us about if there are people with anxiety can have or not depression**
2. **But if they don’t have anxiety then there is a high change of having depression**

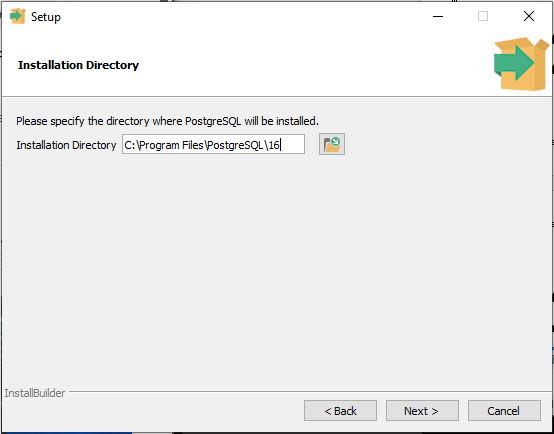
****

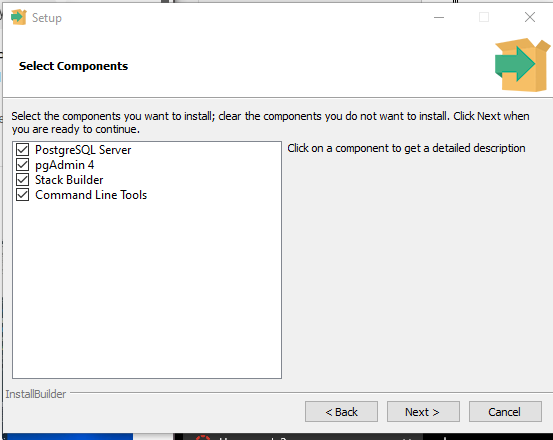
1. **The are more number of females with anxiety then males.**

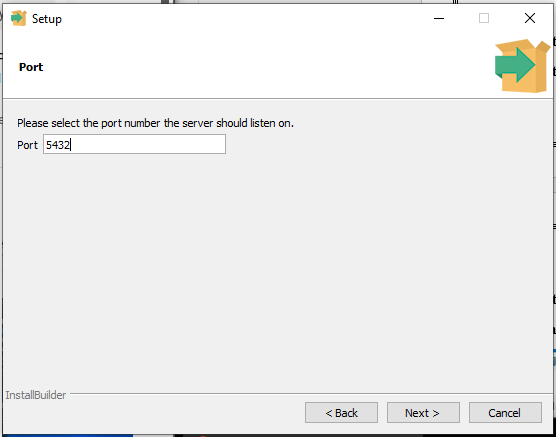
****

**Question 2:-**

**🡪 Installing Postgre SQL and admin**

****

****

****

* **Creating tables.**

create table Student(

ID varchar(20) Not Null,

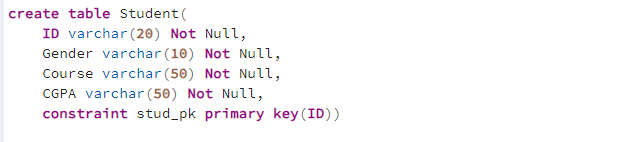
Gender varchar(10) Not Null,

Course varchar(50) Not Null,

CGPA varchar(50) Not Null,

constraint stud\_pk primary key(ID))

select \* from Student



create table Stud\_info(

ID varchar(20) Not Null,

Marital\_Status varchar(10),

Depression varchar(10),

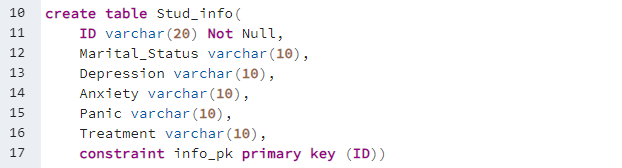
Anxiety varchar(10),

Panic varchar(10),

Treatment varchar(10),

constraint info\_pk primary key (ID))

select \* from Stud\_info



alter table Stud\_info

add constraint fk

foreign key (ID)

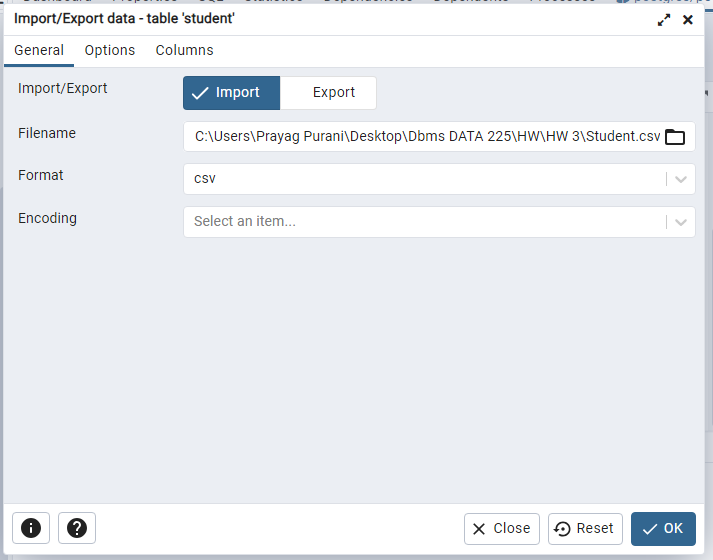
references Student(ID)

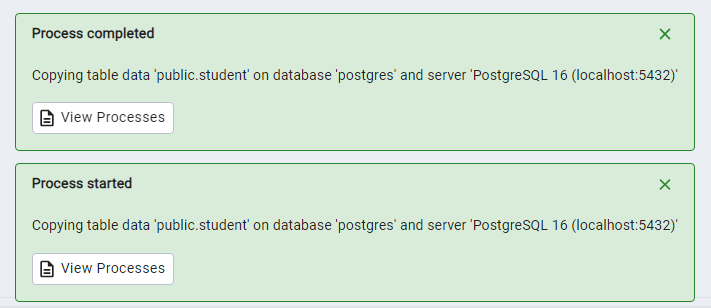


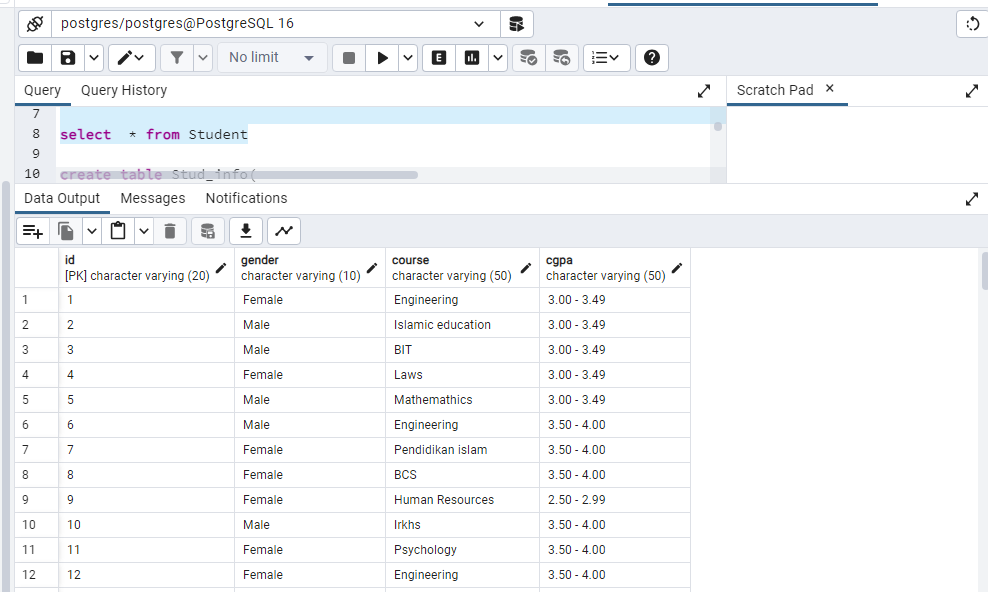
* **Coping the data from the csv file to the database.**

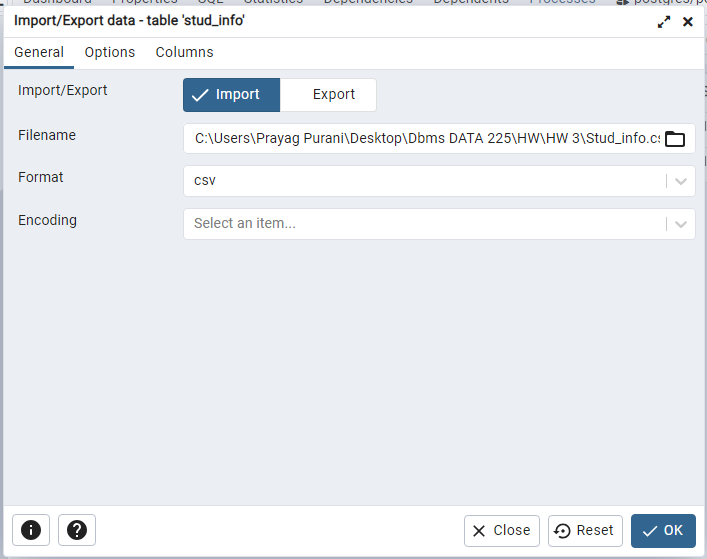
copy public.student (id, gender, course, cgpa) FROM 'C:/Users/Prayag Purani/Desktop/Dbms DATA 225/HW/HW 3/Student.csv' DELIMITER ',' CSV

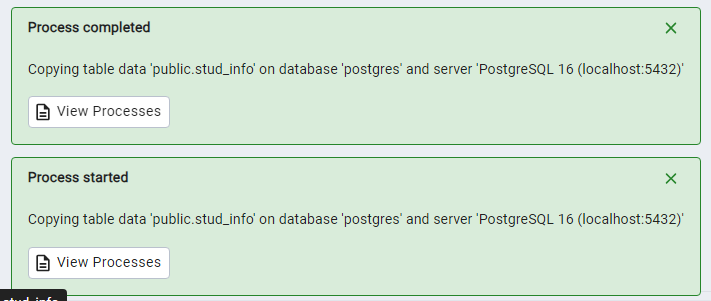
copy public.stud\_info 'C:/Users/Prayag Purani/Desktop/Dbms DATA 225/HW/HW 3/Stud\_info.csv' DELIMITER ',' CSV

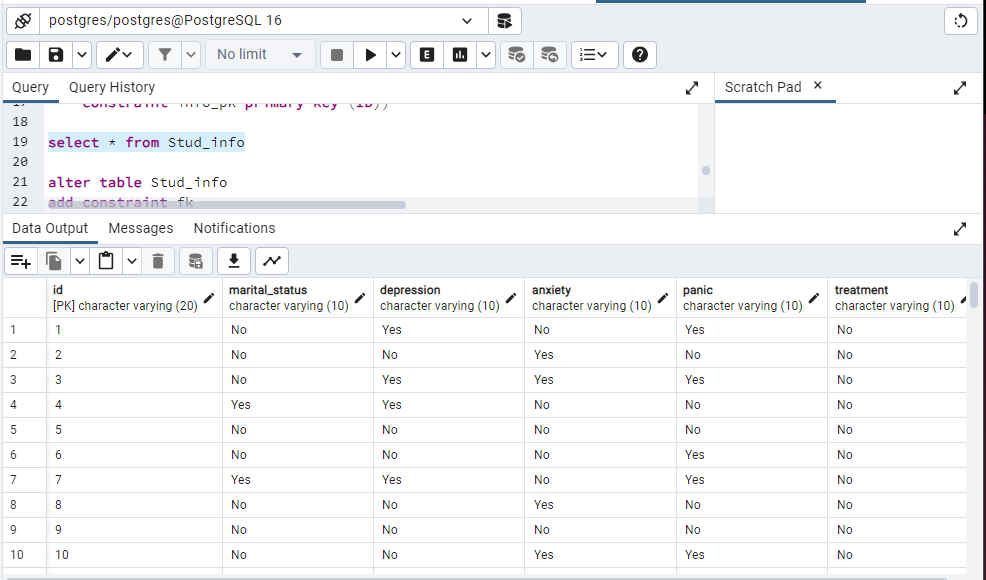






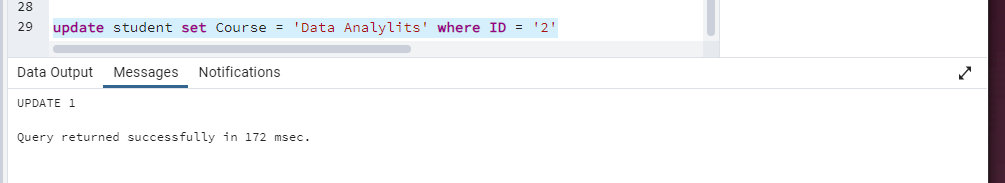






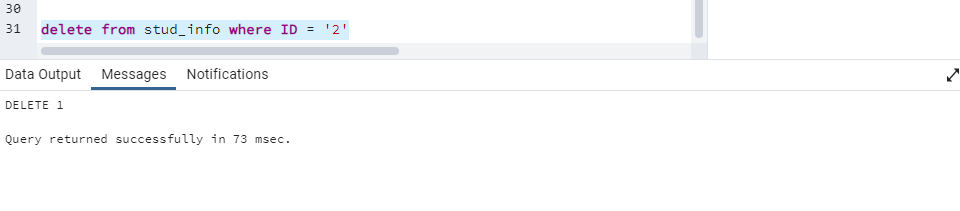
* **Updating the table**

update student set Course = 'Data Analylits' where ID = '2'



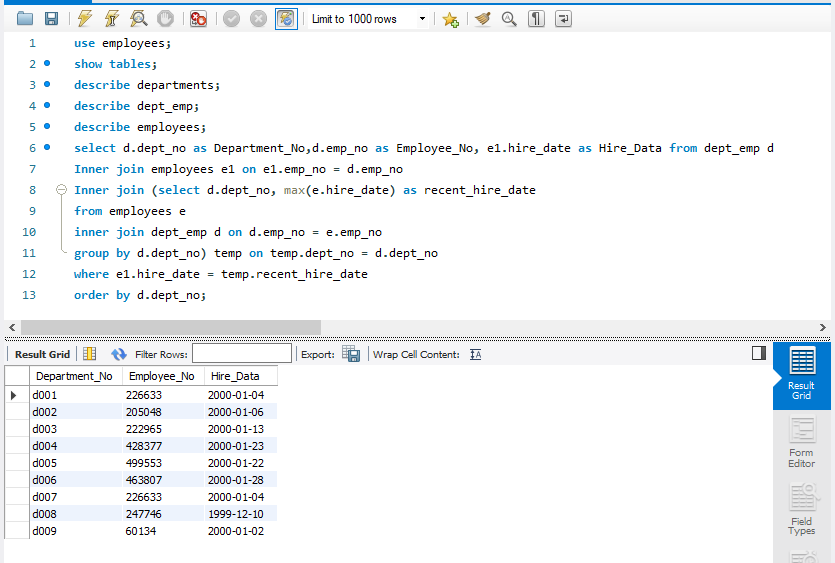
* **Deleting the record**

delete from stud\_info where ID = '2'

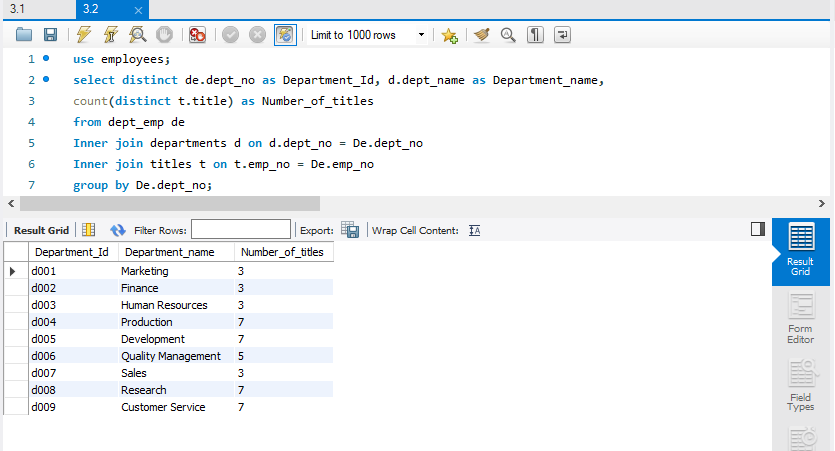


**Question 3:-**

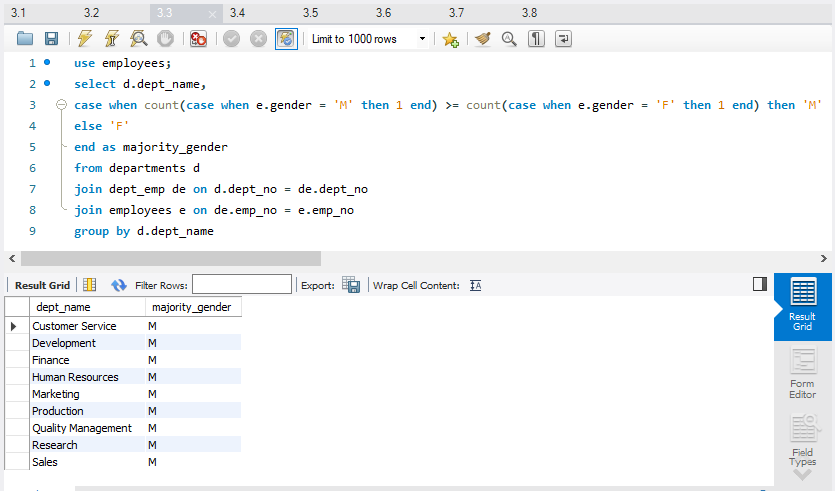
**1.  The most recently hired employee in each of the departments of the organization**

****

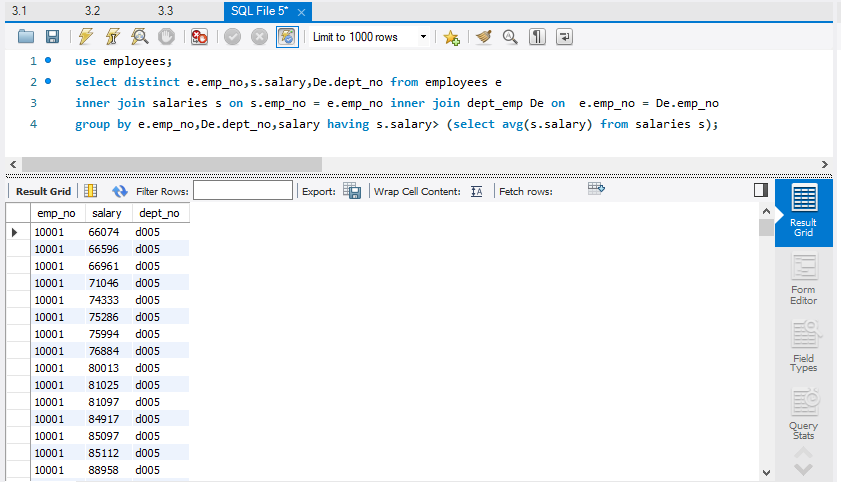
**2. The number of unique titles in each department**

****

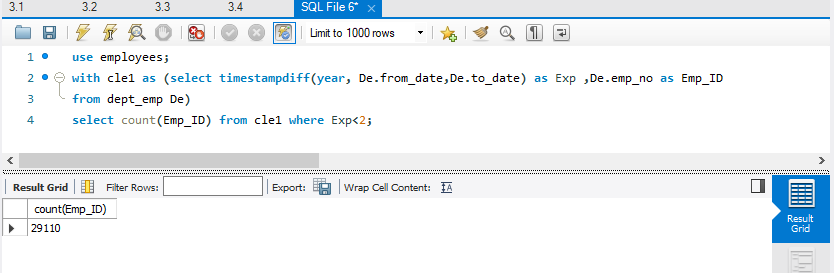
**3.** **The majority gender in each department**

****

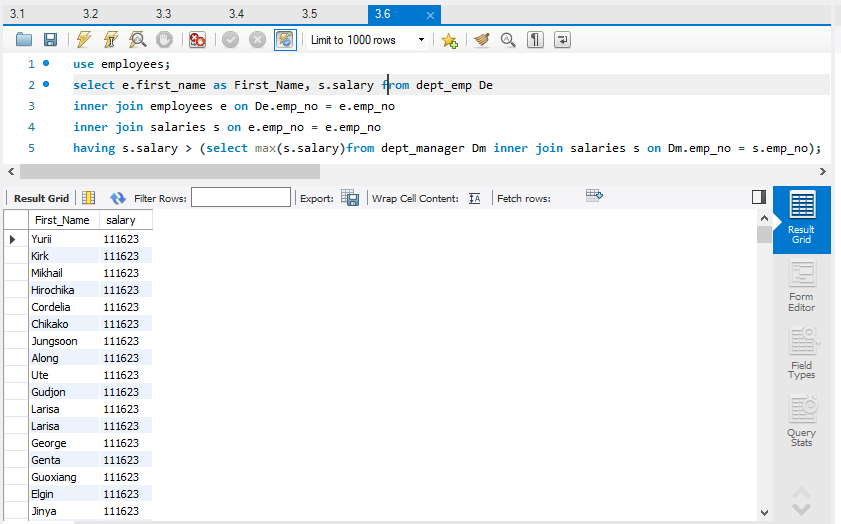
**4. Employees who are making more than the average salary in their department**

****

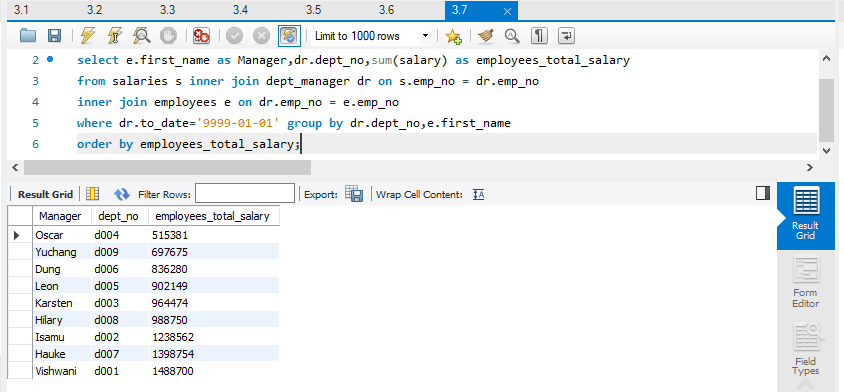
**5. The number of employees who have been working for less than two years in each department.**

****

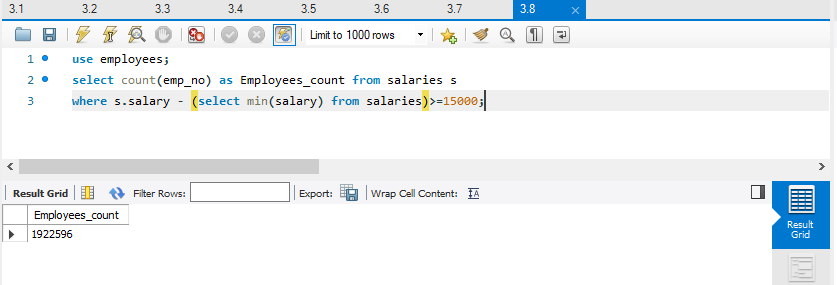
**6. Names of the employees (if any) who are drawing more than their managers.**

****

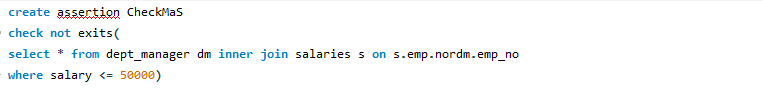
**7. Find the name of the manager of the department which pays the most salaries to its employees (total salary for all employees in a department).**

****

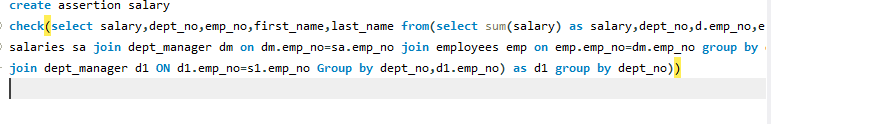
**8.  Find the number of employees whose salary is at least 15,000 more than the least salary in the company.**

****

**9. Using the syntax discussed in class (SQL-92 reference:**[**http://www.contrib.andrew.cmu.edu/~shadow/sql/sql1992.txtLinks to an external site.**](http://www.contrib.andrew.cmu.edu/~shadow/sql/sql1992.txt)**), write an assertion (may not execute) on dept\_manager table which will ensure that all managers have a salary > 50,000**

****

**10. Similarly, write an assertion (may not execute) on salaries table which will ensure that no employee has a salary that is greater than his or her manager’s salary**

****

**Question 4 :-**

1. **SELECT \* FROM t1, t2;**

This query will display the cross production or cartesian product if the table t1 and t2. If the number of rows in t1 is 5 and the number of rows in t2 is 7 the number of rows that will be printed is (5\*7)=35.

1. **SELECT \* FROM t1 a, t1 b;**

This query is same as the above one the only different is the a,b which are used as the alias name for the t1 and t2 respectively.

1. **SELECT \* FROM t1 a, t1 b WHERE a.k = b.m;**

It is also the cartesian product of table 1 and table 2 with alias names a and b with a condition the value of k in table1 should be equal to the value in m column of table 2

1. **SELECT \* FROM t1 a RIGHT JOIN t1 b USING (k);**

This query is performing the right outer join on the two tables t1 and t2 with alias a and b and the join is done using k column.

1. **SELECT \* FROM t1 RIGHT JOIN t2 ON t1.k = t2.k UNION ALL SELECT \* FROM t1 LEFT JOIN t2 ON t1.k = t2.k**

This query contains two join operation and the union is done on the two operations, and we need to take care that the format of both union operations such as column name the types and rest of the things are same.

First is the right operation on both tables using the column k and the second operation is left join on the both the tables using the same column k.

**Question 5a:-**

1. **SELECT E.ename FROM Equipment E, PriceList C, Vendors V**

**WHERE E.eid = C.eid AND C.vid = V.vid AND V.vname = ‘ABC’**

**AND NOT EXISTS ( SELECT \* FROM PriceList C1, Vendors V1**

**WHERE E.eid = C1.eid AND C1.vid = V1.vid AND**

**V1.vname <> ‘ABC’ )**

In order to extract every row from the two databases, the equi join must first be performed on the three tables. The subqueries then provide the data that the vendor name equals ABC. The data from the sub-query will ultimately be negated by the not existing, however I believe the sub-query will be null because there won't be any data with the vendor name ABC.

1. **SELECT V.vname, COUNT(\*) as ToolsCount**

**FROM Vendors V, Equipment E, PriceList C WHERE E.eid = C.eid AND C.vid = V.vid**

**GROUP BY V.vname, V.vid HAVING EVERY (E.category=’Electronics’)**

The vendor's name and the total number of items they have sold will be printed by this query. The output will also be filtered so that just the category "Electronic" is displayed. A group by clause will sort the data in accordance with the vendor's name and ID.

1. **SELECT DISTINCT C.vid FROM PriceList C, Equipment E**

**WHERE C.eid = E.eid AND E.category = ‘Mechanical’**

**UNION**

**SELECT DISTINCT C1.vid FROM PriceList C1, Equipment E1**

**WHERE C1.eid = E1.eid AND E1.category = ‘Electronics’**

Query 1: The "category" in the "Equipment" table is set to "Mechanical," and it chooses unique "vid" values from the "pricelist" table where the "Eid" in the "Equipment" table matches the "Eid" in the "pricelist" table.

Query 2: The "category" in the "Equipment" table is set to "Electronics," and it chooses distinct "vid" values from the "pricelist" table where the "Eid" in the "pricelist" table matches the "Eid" in the "Equipment" table.The outcomes of these two inquiries are combined into a single final result set using the UNION operator.

1. **SELECT E.eid, V.vname FROM Equipment E, Vendors V, PriceList C**

**WHERE C.eid = E.eid AND C.vid = V.vid**

**AND C.price = (SELECT MAX (C1.price)**

**FROM PriceList C1 WHERE C1.eid = E.eid)**

Three tables are queried to provide a list of equipment IDs and vendor names. Based on the "EID" (equipment ID) and "VID" (vendor ID), these tables are connected. After that, a filter is applied by the query to only include the sellers that have sold a certain piece of equipment for the most money.

1. **SELECT DISTINCT C.vid FROM PriceList C**

**WHERE C.price > ( SELECT AVG (C1.price)**

**FROM PriceList C1 WHERE C1.eid = C.eid )**

The equipment's price must be higher than its average price in order for this query to get unique "vid" values from the "PriceList" table.It use a subquery using the AVG function to get the average price, giving the "PriceList" table the alias "C1." The average cost for each particular kind of equipment is calculated by the subquery. Next, each price in the "PriceList" database is compared to the average cost of that particular piece of equipment using the main query.

Top of Form

**References:**

<https://www.youtube.com/watch?v=VcugzjCAjOE>

<https://stackoverflow.com/questions/17669678/how-can-i-check-the-mysql-server-is-installed-or-not-before-installing-the-appli>

<https://realpython.com/python-mysql/>

<https://www.youtube.com/watch?v=z0jVsEuuLLc>

<https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.drop.html>

<https://www.youtube.com/watch?v=91iNR0eG8kE>

<https://hasura.io/docs/latest/schema/postgres/postgres-guides/import-data-from-csv/>

<https://www.youtube.com/watch?v=yDtgk_OLHUc>

<https://stackoverflow.com/questions/54031813/i-am-trying-to-copy-a-file-but-getting-error-message>

<https://www.youtube.com/watch?v=yDtgk_OLHUc>

<https://www.youtube.com/watch?v=6Jf7eTkIaR4>

<https://hasura.io/docs/latest/schema/postgres/postgres-guides/import-data-from-csv/>

<https://hevodata.com/learn/postgresql-import-csv/>

<https://hevodata.com/learn/postgresql-import-csv/>