**Prior Instructions**

* **Please do read all the questions before performing any operations in the database**
* **Once you have fully gone through the questions then likewise decide the contents and table columns and follow the below instructions**

1. Create Student Database



1. Create the following table under the Student Database:
   1. StudentBasicInformation
      1. Columns
         1. StudentName
         2. StudentSurname
         3. StudentRollNo
         4. StudentAddress
         5. Add more three basic columns of the name of your own

* 1. StudentAdmissionPaymentDetails
     1. Columns
        1. StudentRollNo
        2. AmountPaid
        3. AmountBalance
        4. Add more four basic columns of the name of your own
  2. StudentSubjectInformation
     1. Columns
        1. SubjectOpted
        2. StudentRollNo
        3. SubjectTotalMarks
        4. SubjectObtainedMarks
        5. StudentMarksPercentage
        6. Add more one columns of the name of your own
  3. SubjectScholarshipInformation
     1. Columns
        1. StudentRollNo
        2. ScholarshipName
        3. ScholarshipDescription
        4. ScholarshipAmount
        5. ScholarshipCategory
        6. Add more two columns of the name of your own





1. Insert more than 10 records in each and every table create

* **Adding into studentbasicinformation**
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarship) values ('jatin','kumar',1,'dwarka,delhi',21,'science',9898965345,'yes');
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarship) values ('vikash','prajpati',2,'agra,uttar pradesh',20,'science',7956841289,'no');
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarship) values ('aditya','kumar',3,'uttam nagar,delhi',23,'commerce',8945760214,'yes');
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarship) values ('gaurav','chahuhan',4,'kanpur,uttar pradesh',22,'commerce',9965781230,'no');
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarship) values ('kartik','kumar',5,'dwarka,delhi',21,'arts',9874526378,'yes');
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarship) values ('priyanka','madhwal',6,'rajouri garden,delhi',19,'commerce',8856947812,'no');
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarship) values ('ankita','mishra',7,'lajapat nagar,delhi',19,'science',9801457832,'yes');
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarship) values ('pradeep','kumar',8,'ludhiana,punjab',22,'science',9978441122,'no');
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarship) values ('harshit','kesar',9,'tagore garden,delhi',24,'arts',9899658910,'yes');
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarship) values ('nitin','kumar',10,'dwarka,delhi',20,'science',9999784520,'no');
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarshhip) values ('udit','kumar',11,'dwarka,delhi',24,'commerce',7458963210,'yes');
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarshhip) values ('rahul','kumar',12,'dwarka,delhi',23,'commerce',9856320147,'yes');
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarshhip) values ('vansh','chahun',13,'najafgarh,delhi',20,'commerce',9756841230,'yes');
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarshhip) values ('stuti','bisht',14,'lajpat nagar,delhi',21,'science',9896574231,'yes');
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarshhip) values ('ajay','kumar',15,'moti nagar,delhi',22,'science',9652364050,'yes');
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarshhip) values ('shubham','singh',16,'daryaganj,delhi',22,'science',9456782301,'yes');
* insert into studentbasicinformation (studentname,studentsurname,studentrollno,studentaddress,studentage,studentbranch,studentphone,optedScholarshhip) values ('gaurav','kumar',17,'moti nagar,delhi',20,'commerce',9652364152,'yes');
* **Adding into studentadmissionpaymentdetails**
  + insert into studentAdmissionPaymentDetails(studentrollno,amountpaid,amountbalance,lastpaymentamount,lastpaymentdate,latefeecharges) values(1,8000,12000,4000,'2020-12-24',100);
  + insert into studentAdmissionPaymentDetails(studentrollno,amountpaid,amountbalance,lastpaymentamount,lastpaymentdate,latefeecharges) values(2,18000,2000,8000,'2020-11-30',70);
  + insert into studentAdmissionPaymentDetails(studentrollno,amountpaid,amountbalance,lastpaymentamount,lastpaymentdate,latefeecharges) values(3,20000,0,20000,'2020-11-12',0);
  + insert into studentAdmissionPaymentDetails(studentrollno,amountpaid,amountbalance,lastpaymentamount,lastpaymentdate,latefeecharges) values(4,2000,18000,2000,'2021-01-02',250);
  + insert into studentAdmissionPaymentDetails(studentrollno,amountpaid,amountbalance,lastpaymentamount,lastpaymentdate,latefeecharges) values(5,10000,10000,10000,'2021-01-01',300);
  + insert into studentAdmissionPaymentDetails(studentrollno,amountpaid,amountbalance,lastpaymentamount,lastpaymentdate,latefeecharges) values(6,8500,11500,4500,'2020-12-12',120);
  + insert into studentAdmissionPaymentDetails(studentrollno,amountpaid,amountbalance,lastpaymentamount,lastpaymentdate,latefeecharges) values(7,6000,14000,6000,'2020-12-26',150);
  + insert into studentAdmissionPaymentDetails(studentrollno,amountpaid,amountbalance,lastpaymentamount,lastpaymentdate,latefeecharges) values(8,5000,15000,5000,'2020-12-03',200);
  + insert into studentAdmissionPaymentDetails(studentrollno,amountpaid,amountbalance,lastpaymentamount,lastpaymentdate,latefeecharges) values(9,12000,8000,6000,'2020-11-29',100);
  + insert into studentAdmissionPaymentDetails(studentrollno,amountpaid,amountbalance,lastpaymentamount,lastpaymentdate,latefeecharges) values(10,17000,3000,4000,'2020-12-20',90);
  + insert into studentAdmissionPaymentDetails(studentrollno,amountpaid,amountbalance,lastpaymentamount,lastpaymentdate,latefeecharges) values(11,15000,5000,4000,'2020-12-20',90);
  + insert into studentAdmissionPaymentDetails(studentrollno,amountpaid,amountbalance,lastpaymentamount,lastpaymentdate,latefeecharges) values(12,10000,10000,6000,'2020-11-20',150);
  + insert into studentAdmissionPaymentDetails(studentrollno,amountpaid,amountbalance,lastpaymentamount,lastpaymentdate,latefeecharges) values(13,10500,9500,10500,'2020-11-25',60);
  + insert into studentAdmissionPaymentDetails(studentrollno,amountpaid,amountbalance,lastpaymentamount,lastpaymentdate,latefeecharges) values(14,14000,6000,10000,'2020-12-19',160);
  + insert into studentAdmissionPaymentDetails(studentrollno,amountpaid,amountbalance,lastpaymentamount,lastpaymentdate,latefeecharges) values(15,18000,2000,8000,'2020-12-18',70);
* **Adding into studentsubjectinformation**
  + insert into studentsubjectinformation (subjectopted,studentrollno,subjecttotalmarks,subjectobtainedmarks,subjectcode) values ('algebra',1,100,80,301);
  + call calcPerc(1,301,80,100);
  + insert into studentsubjectinformation (subjectopted,studentrollno,subjecttotalmarks,subjectobtainedmarks,subjectcode)

values ('ds and algo',4,70,60,302);

call calcPerc(4,302,60,70);

* + insert into studentsubjectinformation (subjectopted,studentrollno,subjecttotalmarks,subjectobtainedmarks,subjectcode)

values ('ds and algo',3,70,45,302);

* + call calcPerc(3,302,45,70);
  + insert into studentsubjectinformation (subjectopted,studentrollno,subjecttotalmarks,subjectobtainedmarks,subjectcode)

values ('ds and algo',6,70,65,302);

* + call calcPerc(6,302,65,70);
  + insert into studentsubjectinformation (subjectopted,studentrollno,subjecttotalmarks,subjectobtainedmarks,subjectcode)

values ('chemical engineering',5,100,65,303);

* + call calcPerc(5,303,65,100);
  + insert into studentsubjectinformation (subjectopted,studentrollno,subjecttotalmarks,subjectobtainedmarks,subjectcode)

values ('dbms',11,70,61,304);

* + call calcPerc(11,304,61,70);
  + insert into studentsubjectinformation (subjectopted,studentrollno,subjecttotalmarks,subjectobtainedmarks,subjectcode)

values ('dbms',8,70,33,304);

* + call calcPerc(8,304,33,70);
  + insert into studentsubjectinformation (subjectopted,studentrollno,subjecttotalmarks,subjectobtainedmarks,subjectcode)

values ('dbms',9,70,44,304);

* + call calcPerc(9,304,44,70);
  + insert into studentsubjectinformation (subjectopted,studentrollno,subjecttotalmarks,subjectobtainedmarks,subjectcode)

values ('data mining',7,100,65,305);

* + call calcPerc(7,305,65,100);
  + insert into studentsubjectinformation (subjectopted,studentrollno,subjecttotalmarks,subjectobtainedmarks,subjectcode)

values ('data mining',10,100,95,305);

* + call calcPerc(10,305,95,100);
  + insert into studentsubjectinformation (subjectopted,studentrollno,subjecttotalmarks,subjectobtainedmarks,subjectcode)

values ('accounting',12,100,90,201);

* + call calcPerc(12,201,90,100);
  + insert into studentsubjectinformation (subjectopted,studentrollno,subjecttotalmarks,subjectobtainedmarks,subjectcode)

values ('economics',13,70,69,202);

* + call calcPerc(13,202,69,70);
  + insert into studentsubjectinformation (subjectopted,studentrollno,subjecttotalmarks,subjectobtainedmarks,subjectcode)

values ('algebra',14,100,90,301);

* + call calcPerc(14,301,90,100);
  + insert into studentsubjectinformation (subjectopted,studentrollno,subjecttotalmarks,subjectobtainedmarks,subjectcode)

values ('chemistry',15,100,67,305);

* + call calcPerc(15,305,67,100);
* **Adding into subjectscholarshipinformation**
  + insert into subjectscholarshipinformation(studentrollno,scholarshipname,scholarshipdescription,scholarshipamount,scholarshipyear,incometype)

values (1,'scholarship A','Academic scholarship',10000,2020,'middle-income');

* + call scholarshipCategory(0,1);
  + insert into subjectscholarshipinformation(studentrollno,scholarshipname,scholarshipdescription,scholarshipamount,scholarshipyear,incometype)

values (3,'scholarship B','Athletic scholarship',3000,2020,'middle-income');

* + call scholarshipCategory(0,3);
  + insert into subjectscholarshipinformation(studentrollno,scholarshipname,scholarshipdescription,scholarshipamount,scholarshipyear,incometype)

values (5,'scholarship A','Academic scholarship',4000,2019,'low-income');

* + call scholarshipCategory(0,5);
  + insert into subjectscholarshipinformation(studentrollno,scholarshipname,scholarshipdescription,scholarshipamount,scholarshipyear,incometype)

values (7,'scholarship B','Athletic scholarship',6000,2018,'low-income');

* + call scholarshipCategory(0,7);
  + insert into subjectscholarshipinformation(studentrollno,scholarshipname,scholarshipdescription,scholarshipamount,scholarshipyear,incometype)

values (9,'scholarship B','Athletic scholarship',8000,2016,'middle-income');

* + call scholarshipCategory(0,9);
  + insert into subjectscholarshipinformation(studentrollno,scholarshipname,scholarshipdescription,scholarshipamount,scholarshipyear,incometype)

values (11,'scholarship B','Athletic scholarship',9000,2019,'low-income');

* + call scholarshipCategory(0,11);
  + insert into subjectscholarshipinformation(studentrollno,scholarshipname,scholarshipdescription,scholarshipamount,scholarshipyear,incometype)

values (12,'scholarship B','Athletic scholarship',2500,2020,'middle-income');

* + call scholarshipCategory(0,12);
  + insert into subjectscholarshipinformation(studentrollno,scholarshipname,scholarshipdescription,scholarshipamount,scholarshipyear,incometype)

values (13,'scholarship A','Academic scholarship',4500,2020,'low-income');

* + call scholarshipCategory(0,13);
  + insert into subjectscholarshipinformation(studentrollno,scholarshipname,scholarshipdescription,scholarshipamount,scholarshipyear,incometype)

values (14,'scholarship A','Academic scholarship',5000,2018,'middle-income');

* + call scholarshipCategory(0,14);
  + insert into subjectscholarshipinformation(studentrollno,scholarshipname,scholarshipdescription,scholarshipamount,scholarshipyear,incometype)

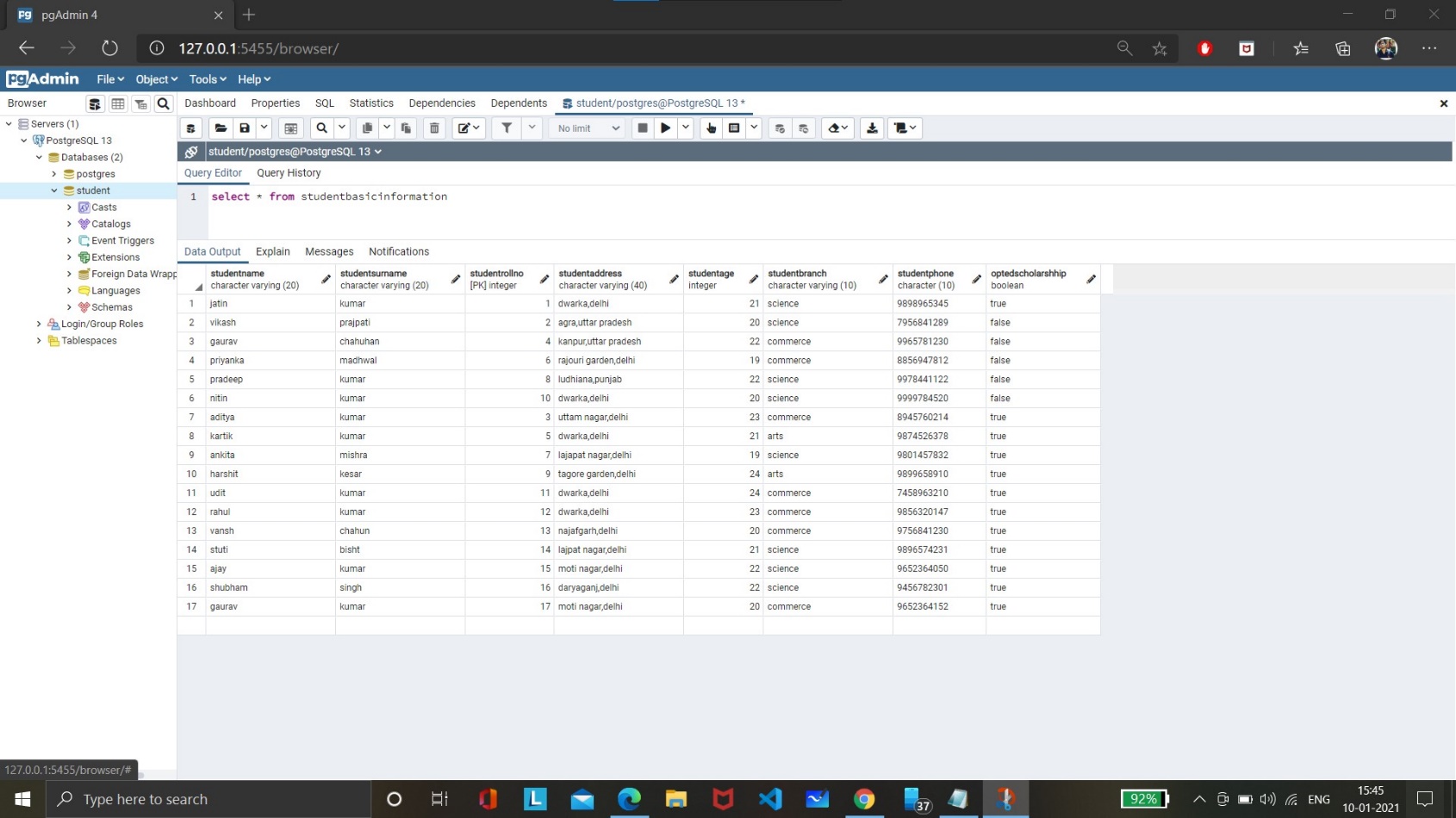
values (15,'scholarship A','Academic scholarship',3000,2017,'low-income');

* + call scholarshipCategory(0,15);

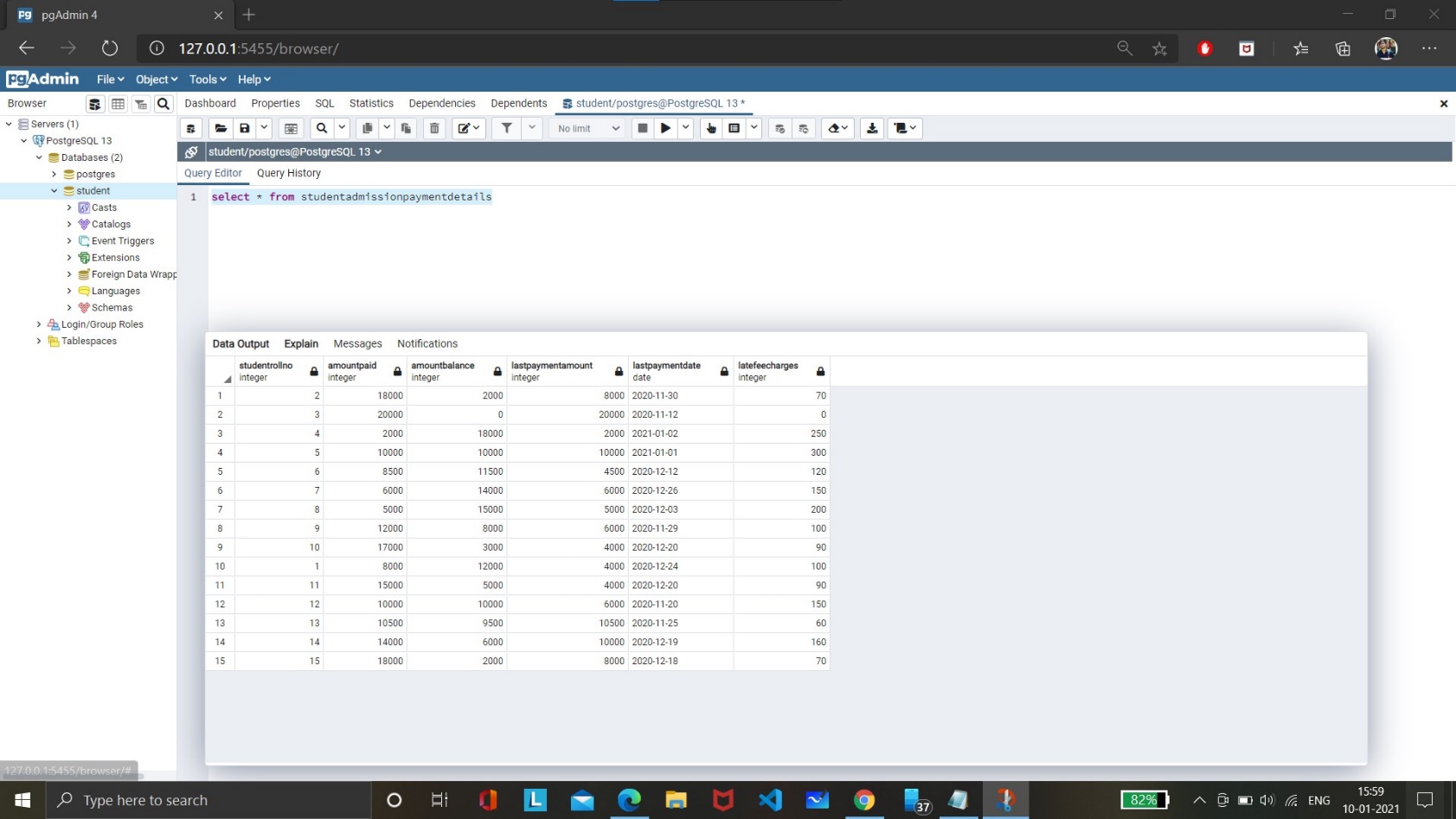
1. Snap of the all the tables once the insertion is completed



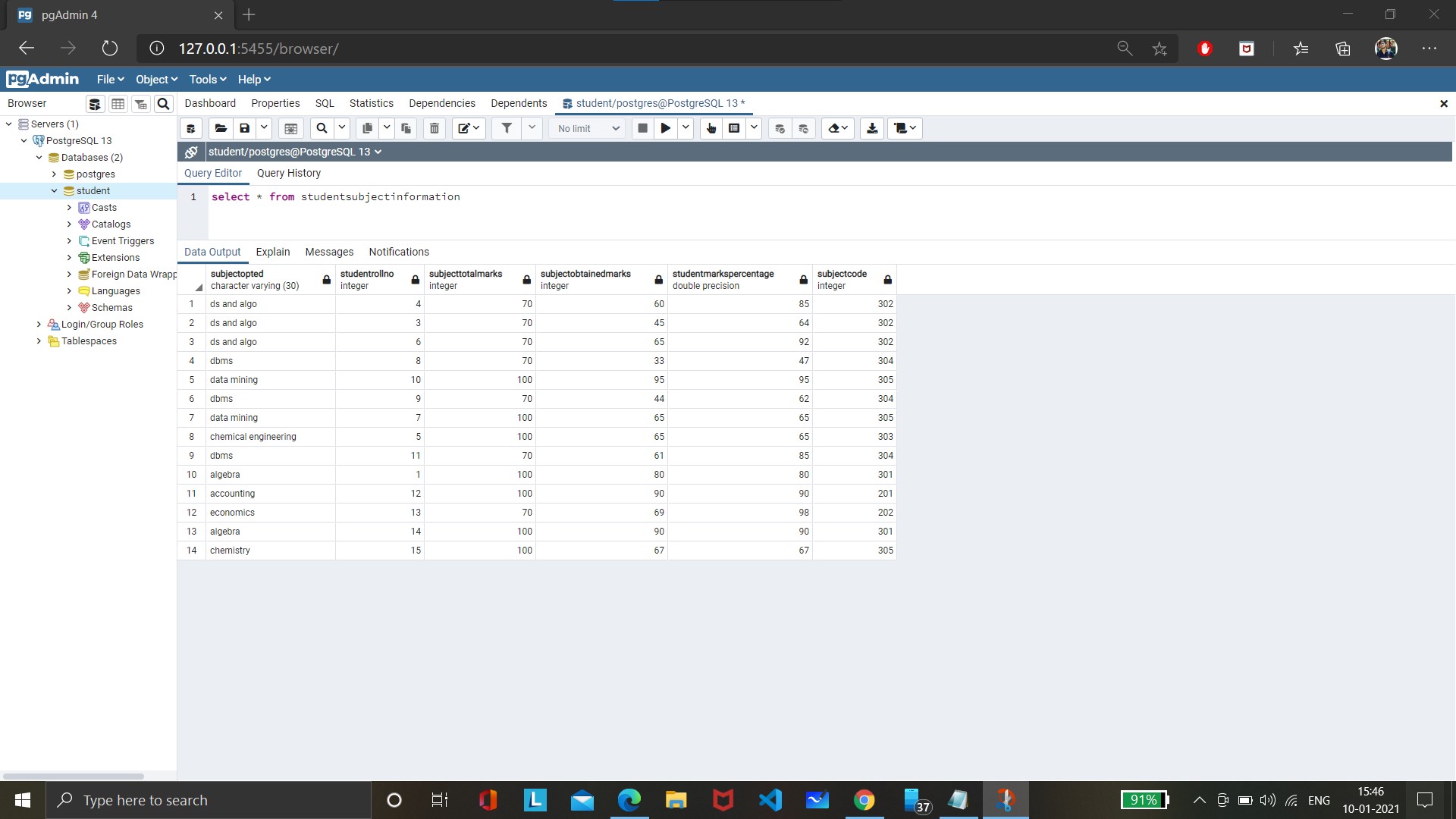
**Subjectbasicinformation**



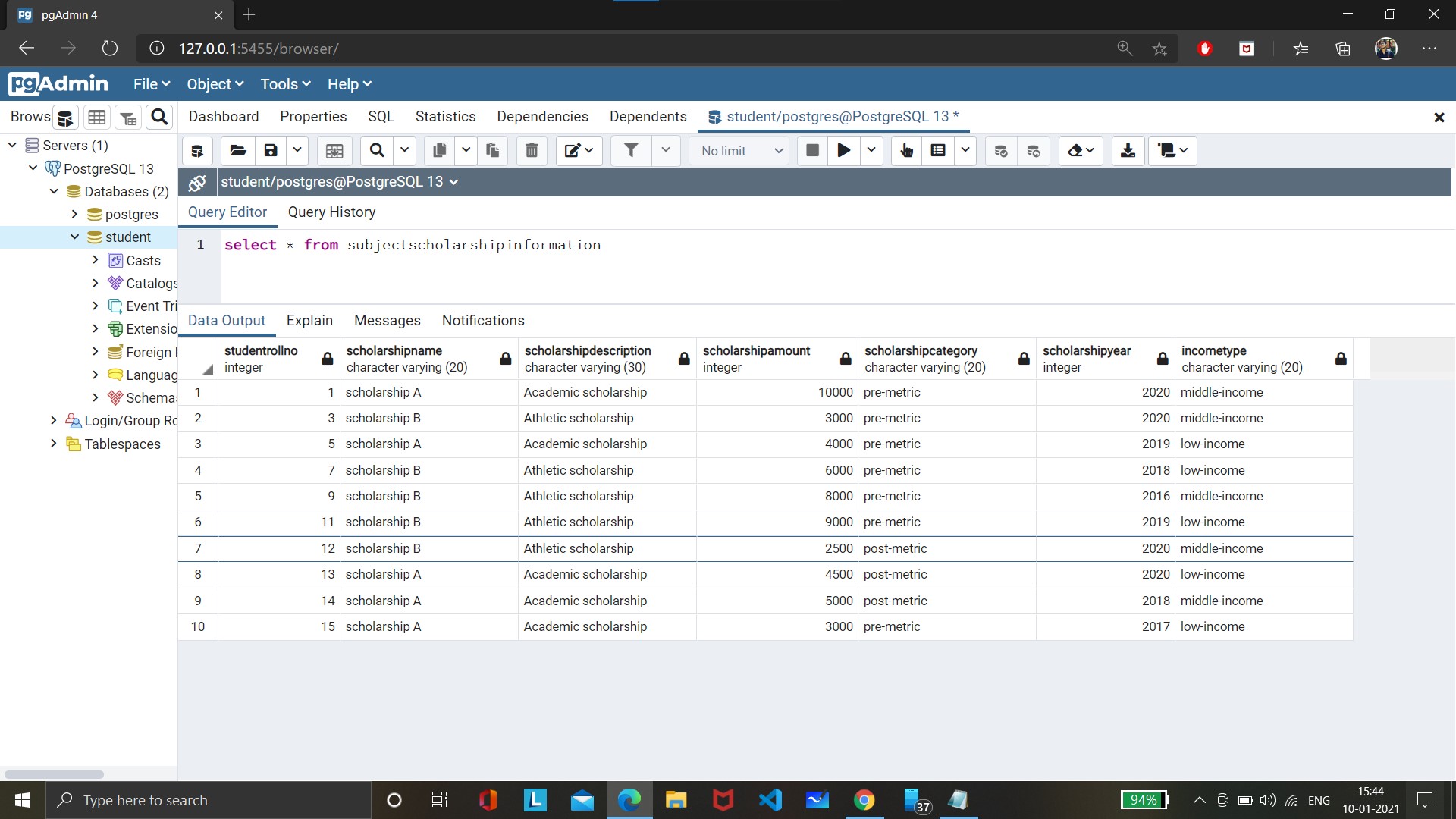
**Studentadmissionpaymentdetails**



**Studentsubjectbasicinformation**



**Subjectscholarshipinformation**



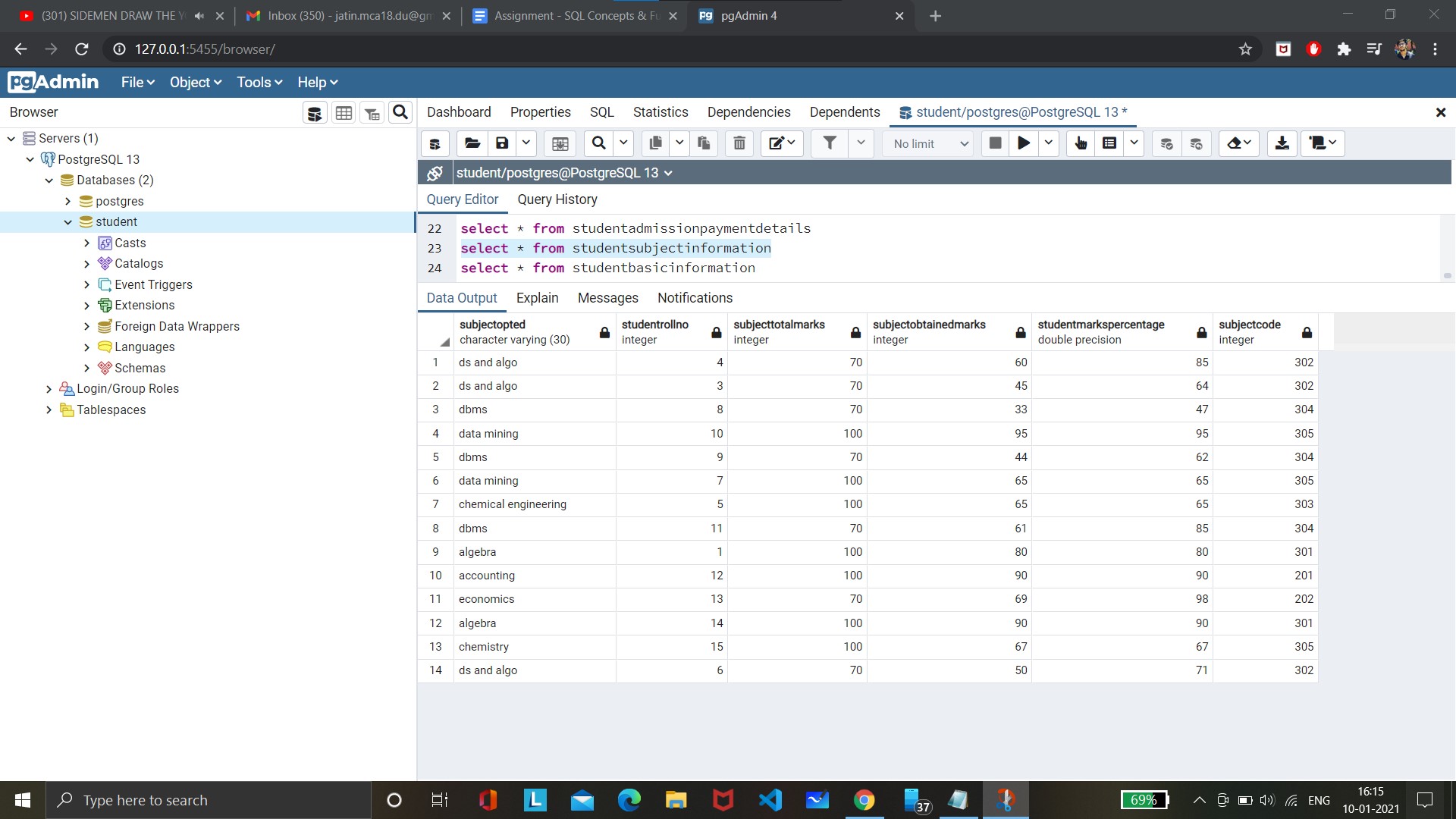
1. Update any 5 records of your choice in any table like update the StudentAddress with some other address content and likewise so on with any records of any table of your choice

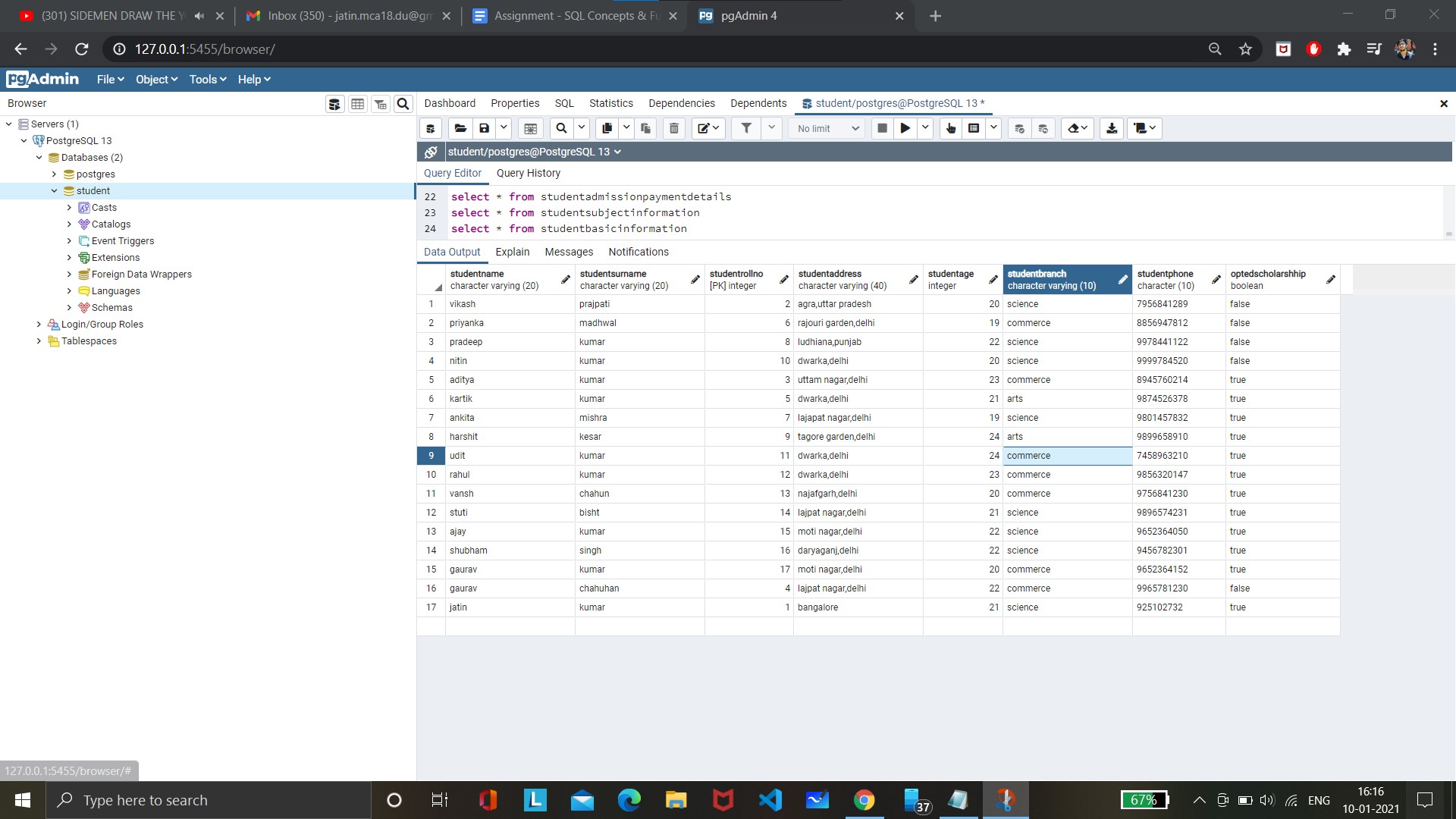


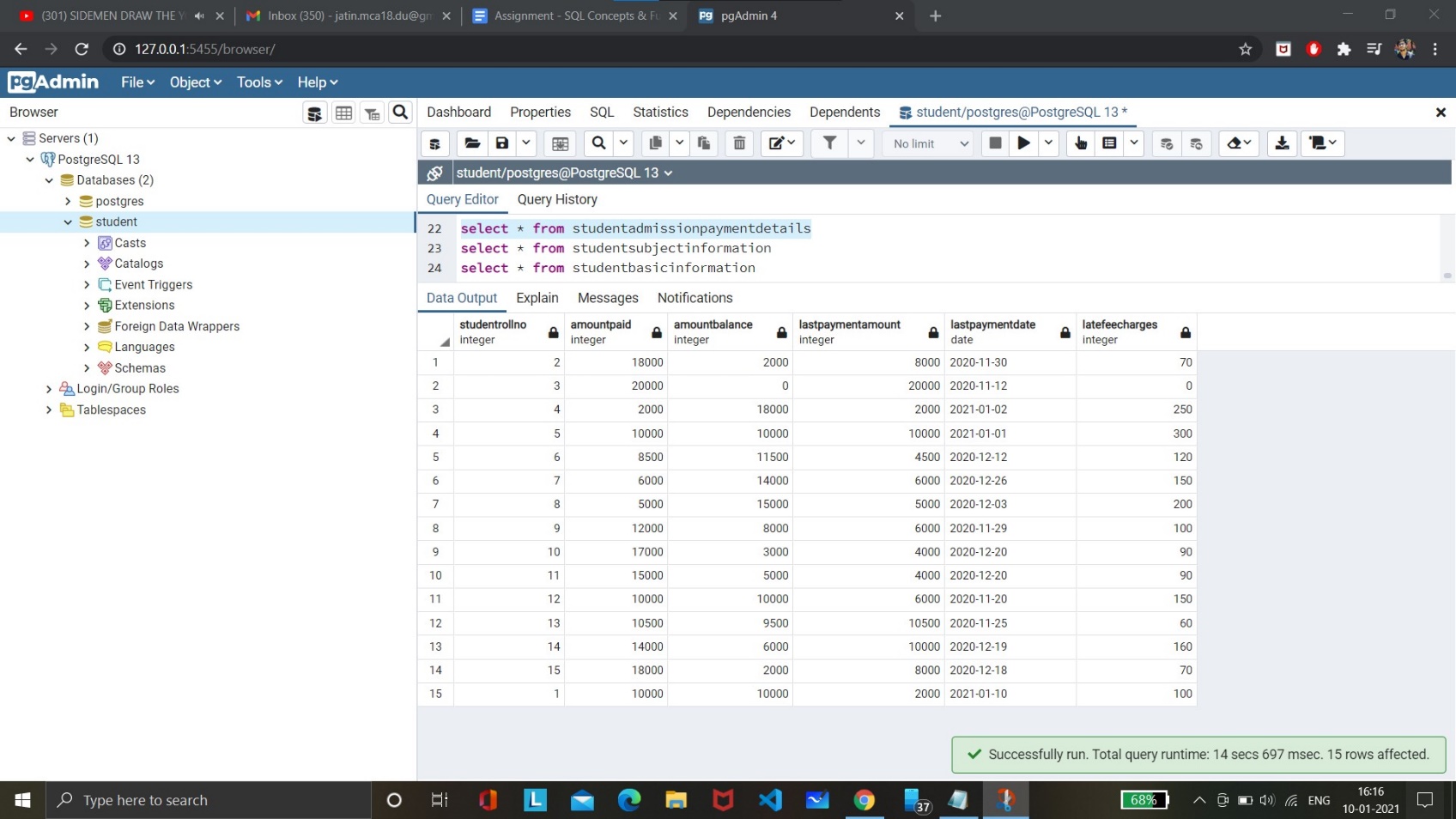


1. Snap of the all the tables post updation





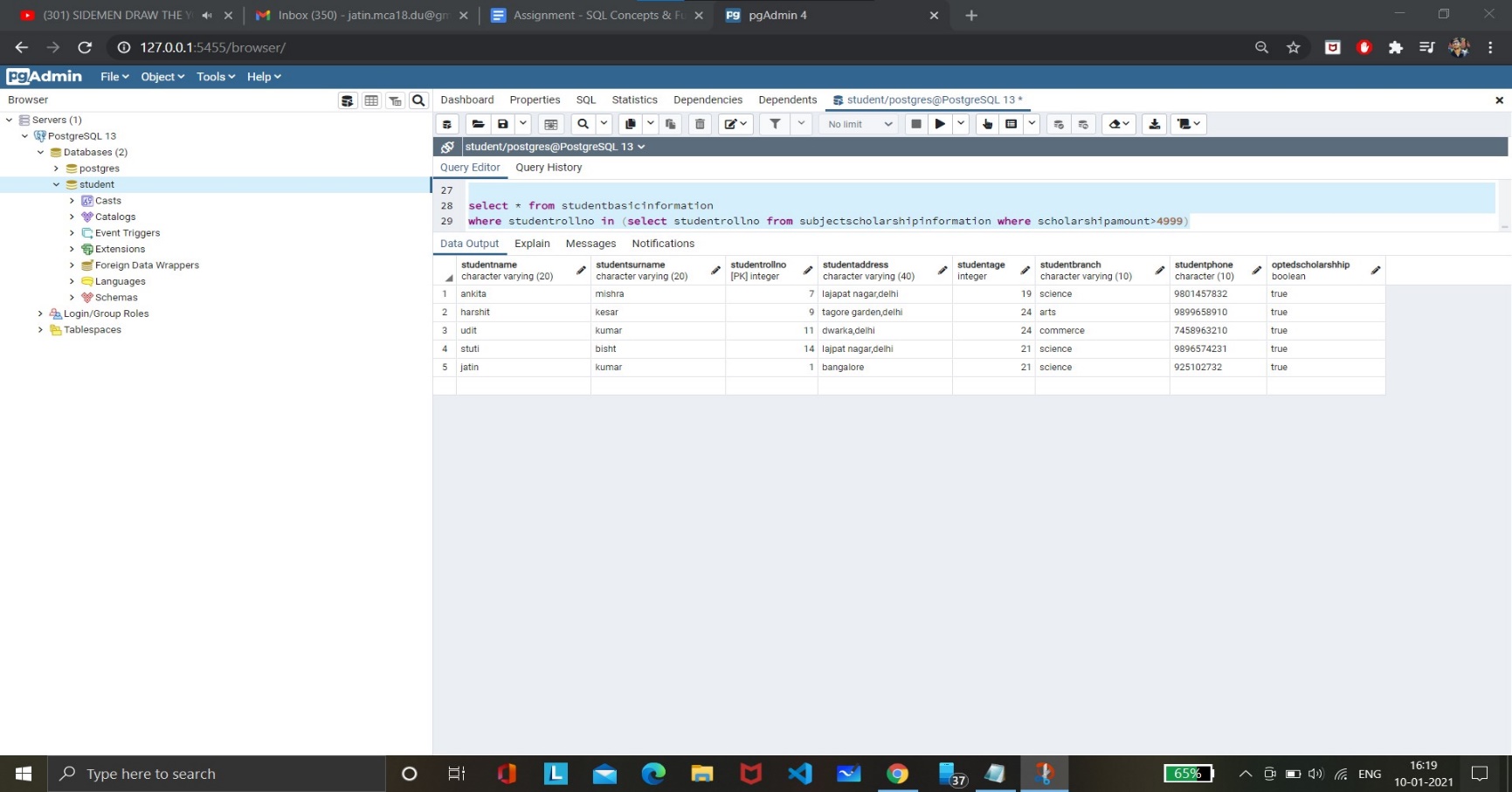




1. Select the student details records who has received the scholarship more than 5000Rs/-



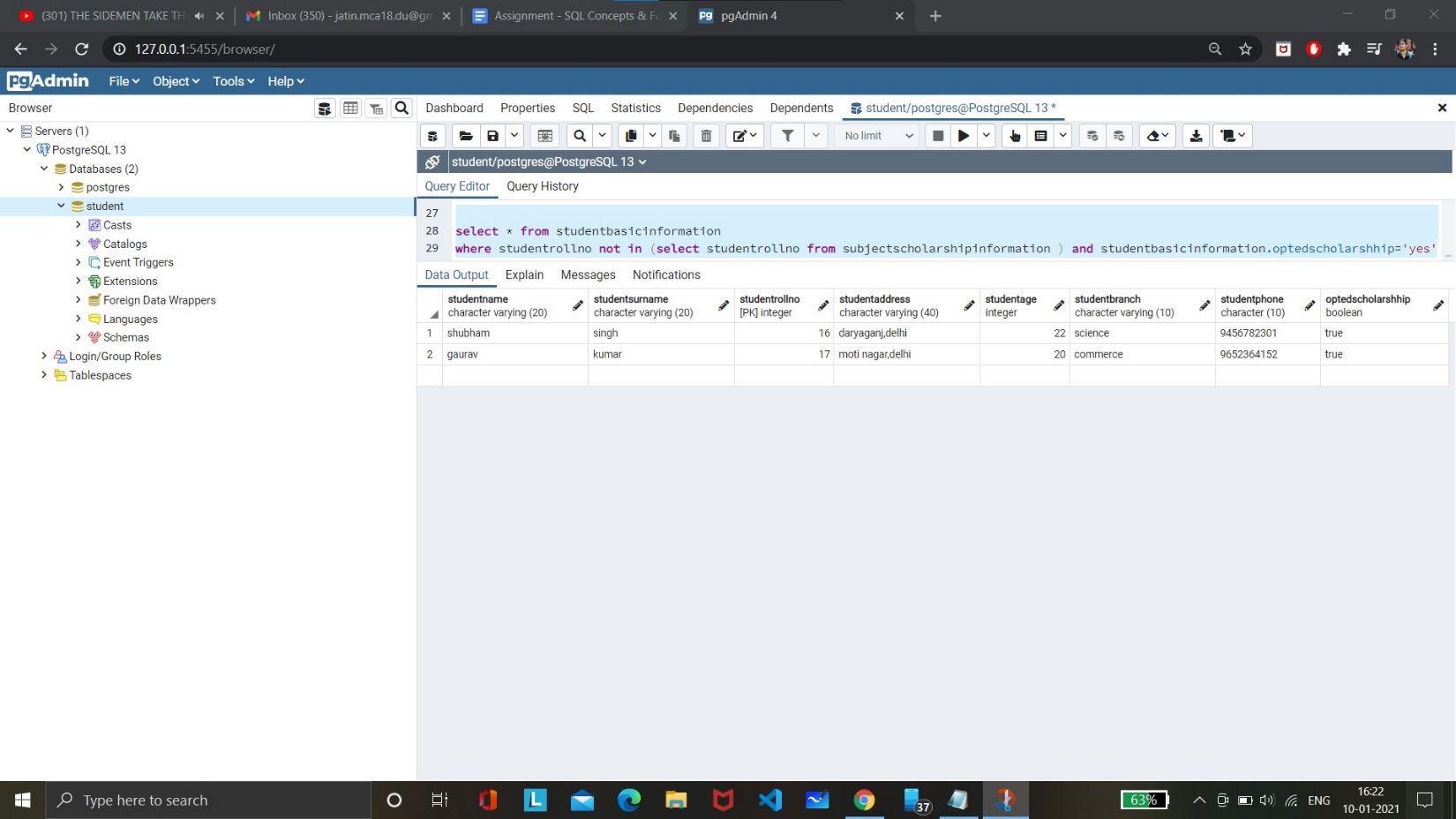




1. Select the students who opted for scholarship but has not got the scholarship







1. Fill in data for the percentage column i.e. StudentMarksPercentage in the table StudentSubjectInformation by creating and using the stored procedure created





***IMPROVEMENTS ON PROCEDURES***

**Update percentages**

create or replace procedure updatePerc(

)

language plpgsql

as $$

begin

update studentsubjectinformation set

studentmarkspercentage= subjectobtainedmarks \* 100 / subjecttotalmarks ;

commit;

end;$$

**note:now we only have to call the procedure once rather calling it again and again.**

1. Decide the category of the scholarship depending upon the marks/percentage obtained by the student and likewise update the ScholarshipCategory column, create a stored procedure in order to handle this operation





***IMPROVEMENTS ON PROCEDURES***

**Update on scholarshipsCategory**

create or replace procedure updatescholarshipcategory(

)

language plpgsql

as $$

declare marks int;

i record ;

begin

for i in ( select studentrollno from subjectscholarshipinformation )

loop

select studentmarkspercentage into marks from studentsubjectinformation

where studentsubjectinformation.studentrollno = i.studentrollno ;

if( marks > 85 ) then

update subjectscholarshipinformation

set scholarshipcategory = 'post-metric'

where studentrollno =i.studentrollno;

else

update subjectscholarshipinformation

set scholarshipcategory = 'pre-metric'

where studentrollno =i.studentrollno;

end if;

end loop;

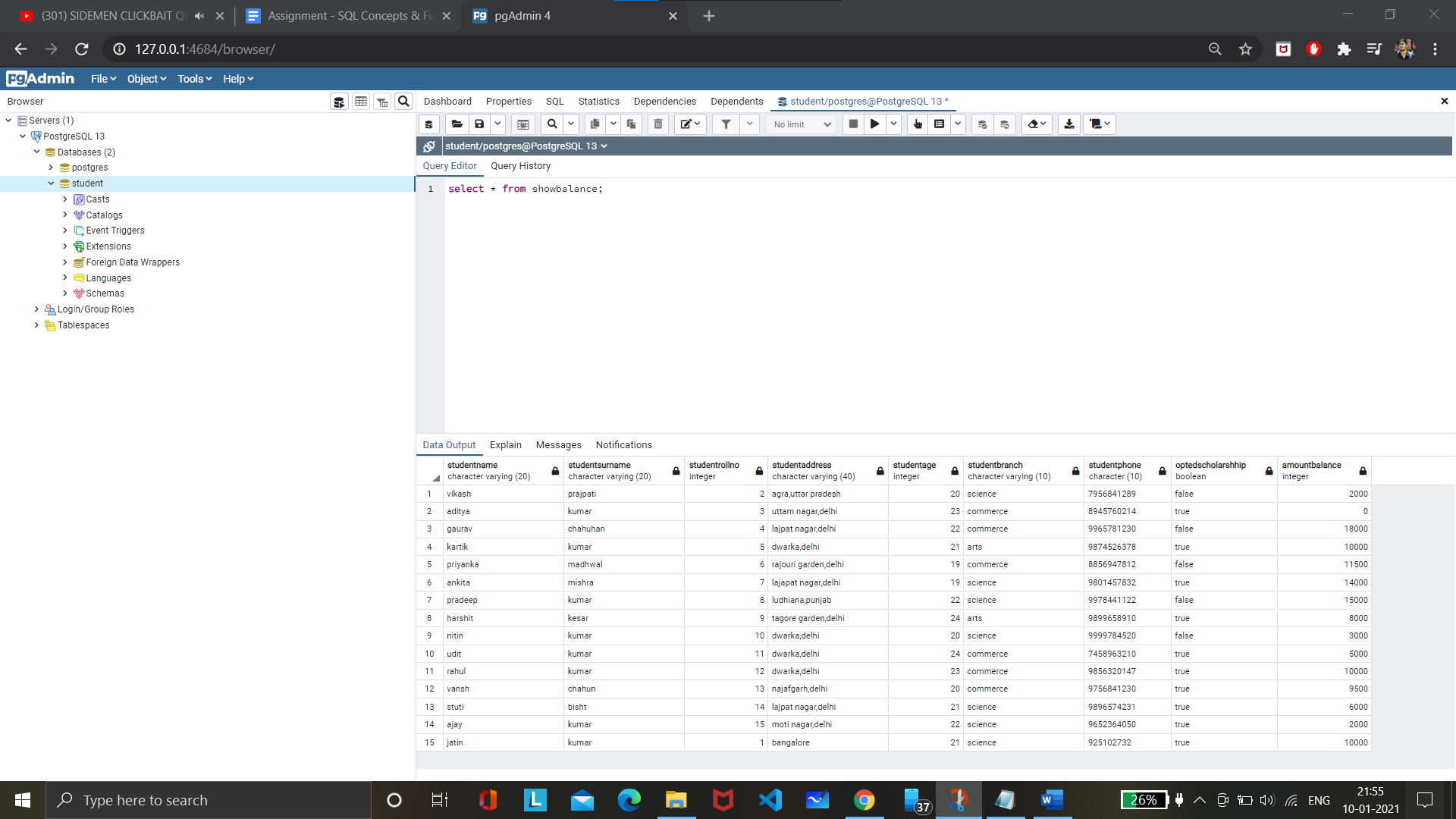
end;$$

**note:now we only have to call the procedure once rather calling it again and again.**

1. Create the View which shows balance amount to be paid by the student along with the student detailed information (use join)



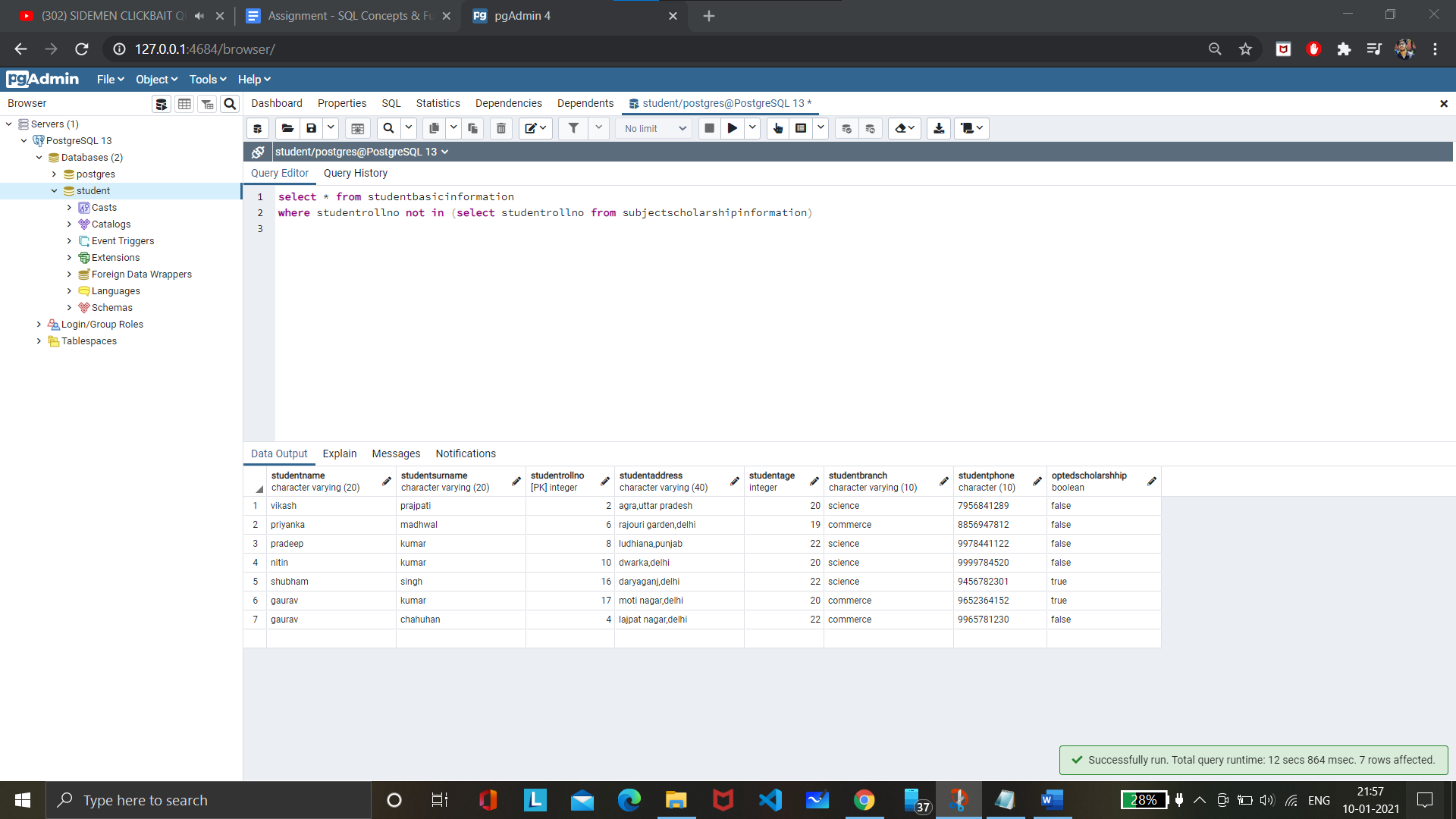




1. Get the details of the students who haven’t got any scholarship (use joins/subqueries)



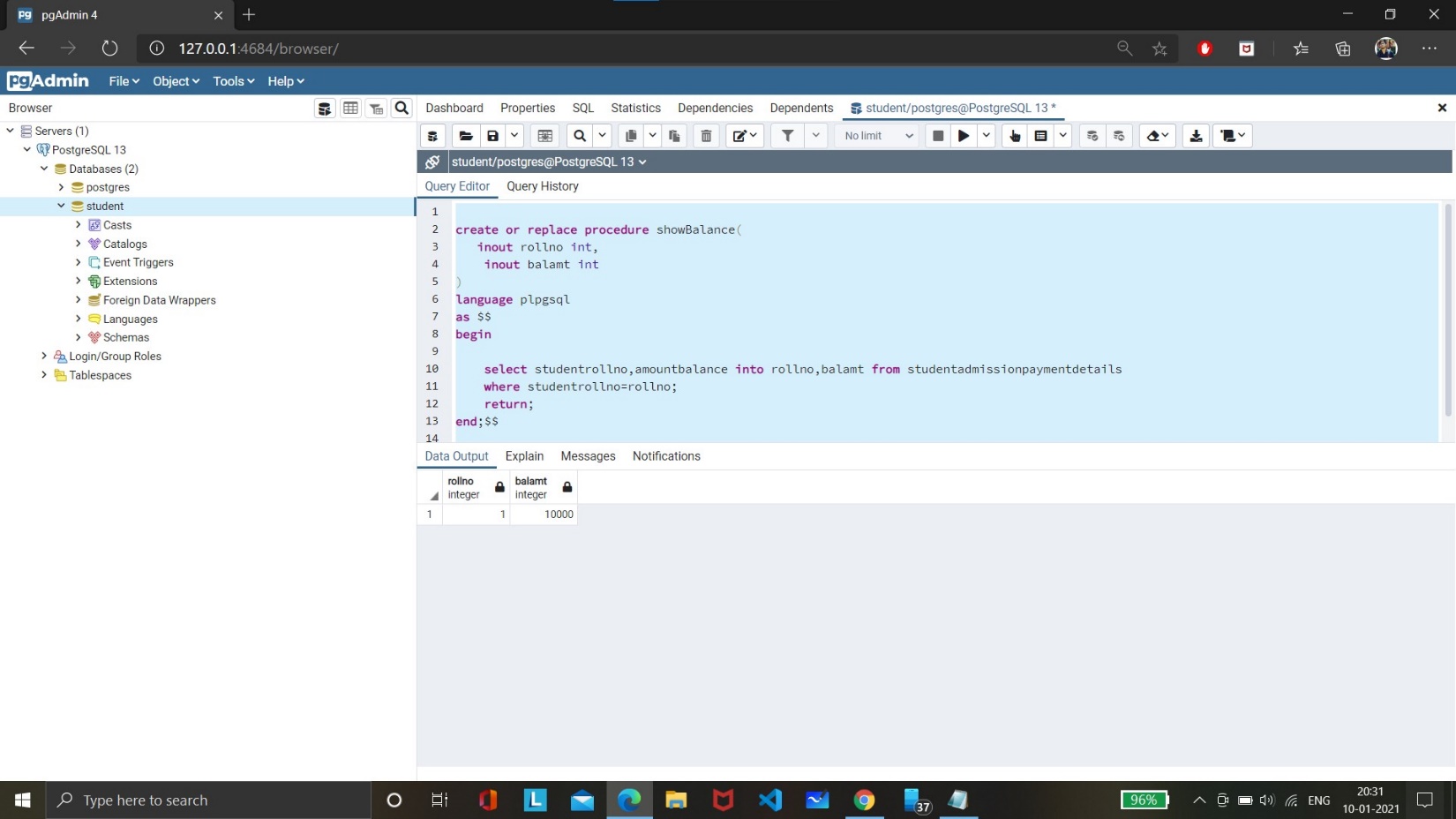




1. Create Stored Procedure which will be return the amount balance to be paid by the student as per the student roll number passed through the stored procedure as the input







1. Retrieve the top five student details as per the StudentMarksPercentage values (use subqueries)

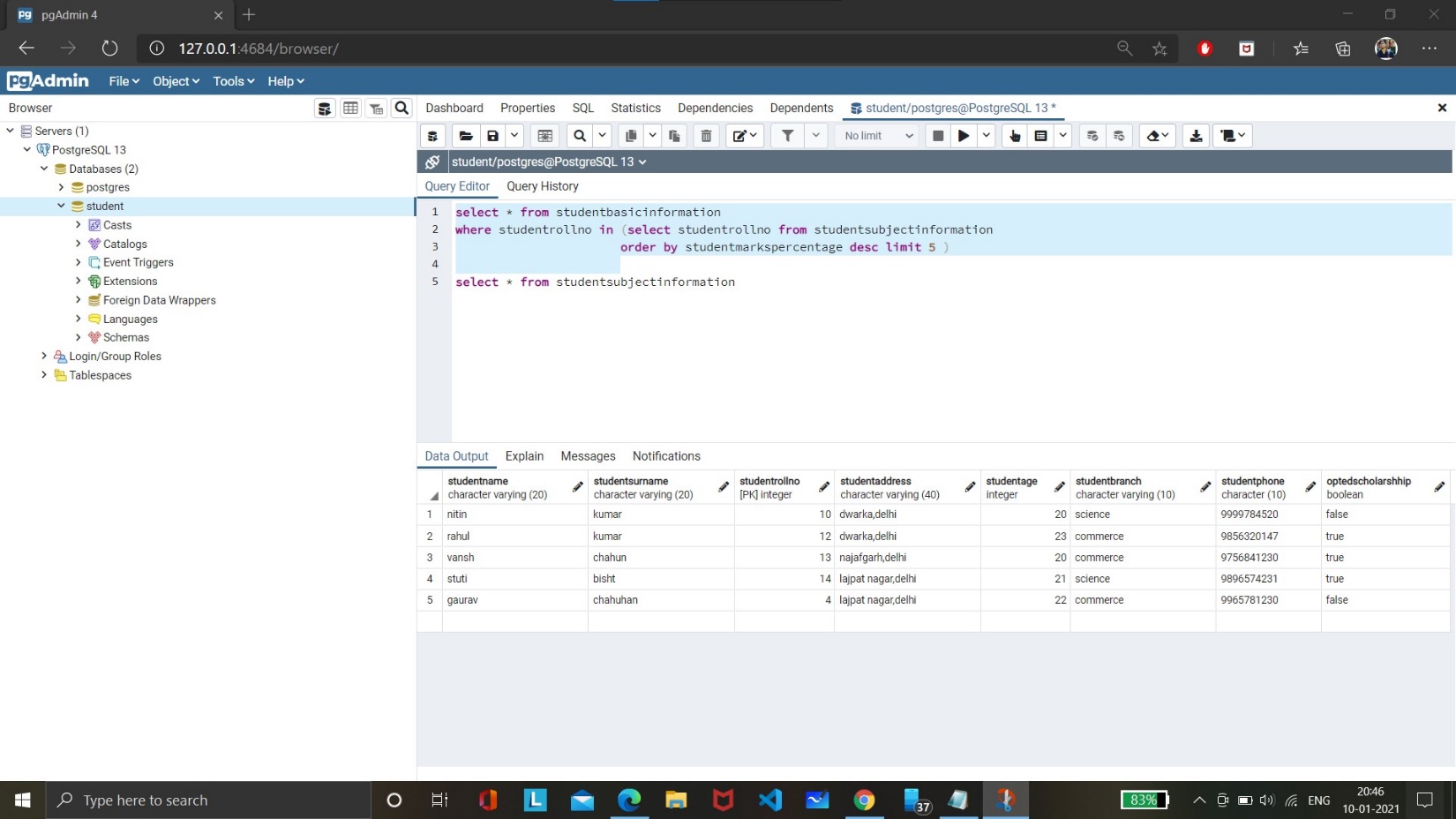




**It doesn’t maintain the order returned from subquery**

**Alternative query which does maintain the order**

****



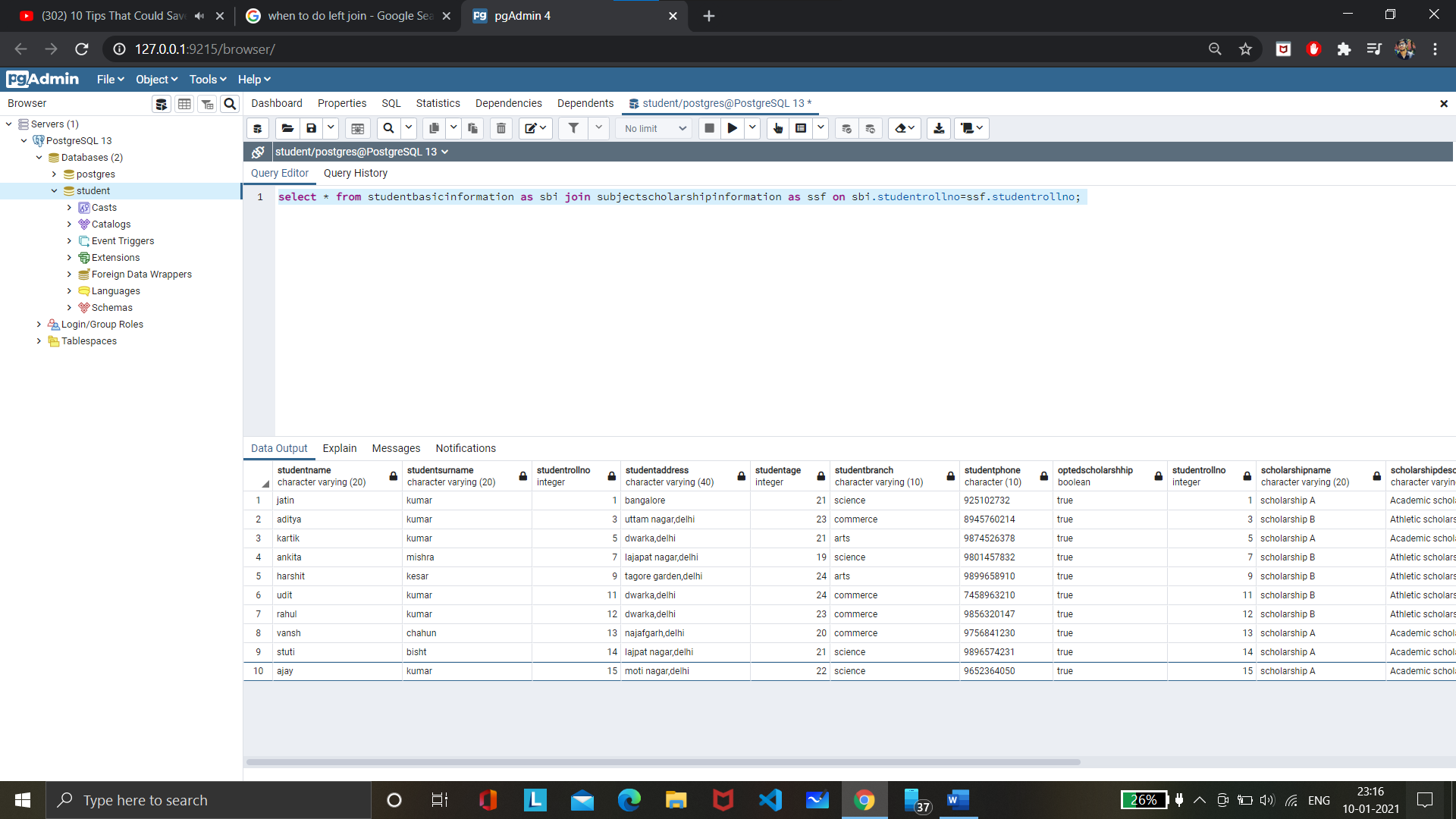
1. Try to use all the three types of join learned today in a relevant way, and explain the same why you thought of using that particular join for your selected scenarios (try to cover relevant and real time scenarios for all the three studied joins)



**INNER JOIN:**

Perform join on *student information* and *subject scholarship information* to get the data of all students who got scholarship.





**LEFT JOIN:**

Getting all subject information along with the student basic information.

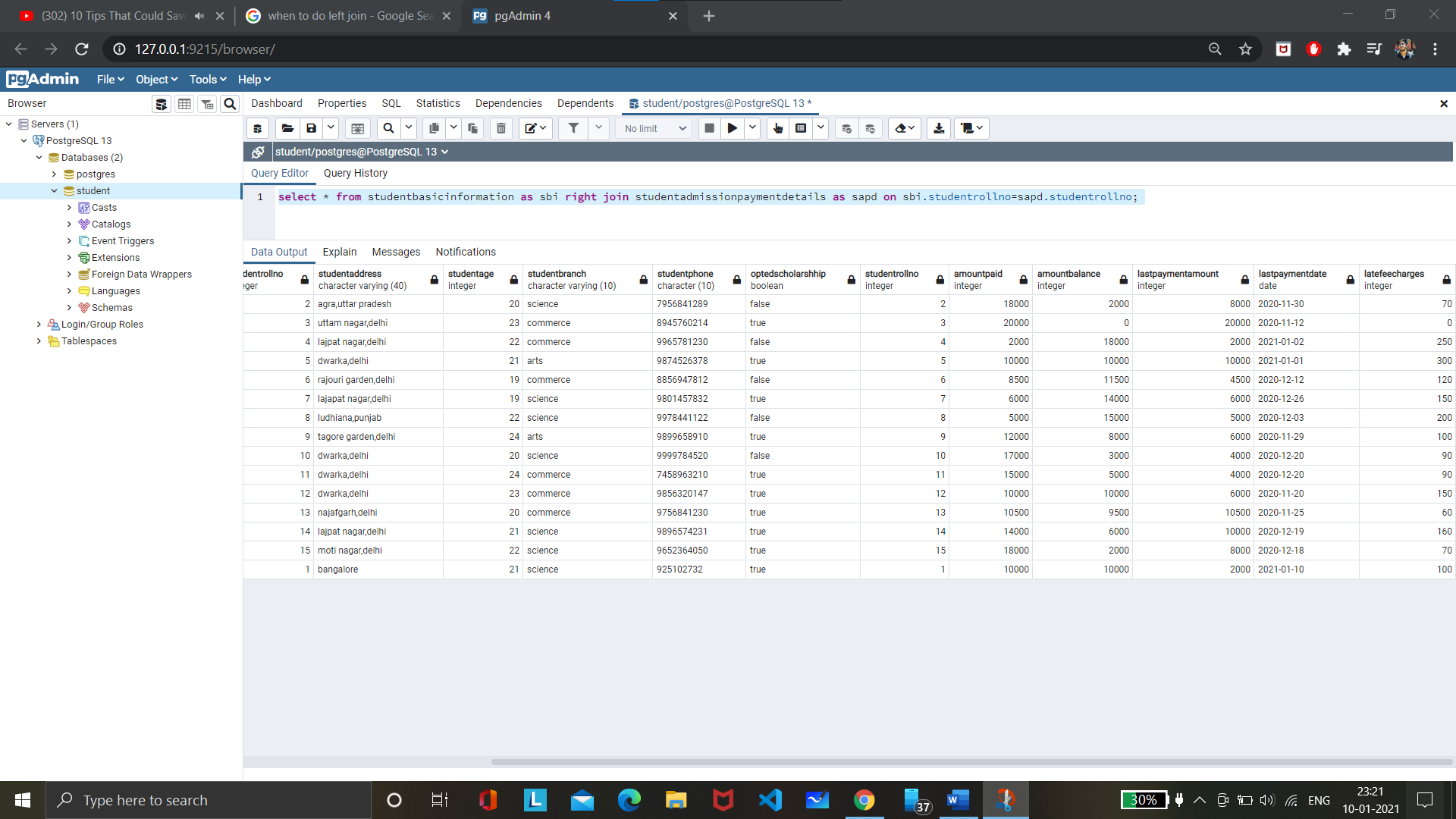




**RIGHT JOIN:**

Getting student basic information with all the payment details of all students.

****

****

1. Mention the differences between the delete, drop and truncate commands

* **DROP :** drop deletes all the data from the table and the table and all the trigger,contraints are removed.

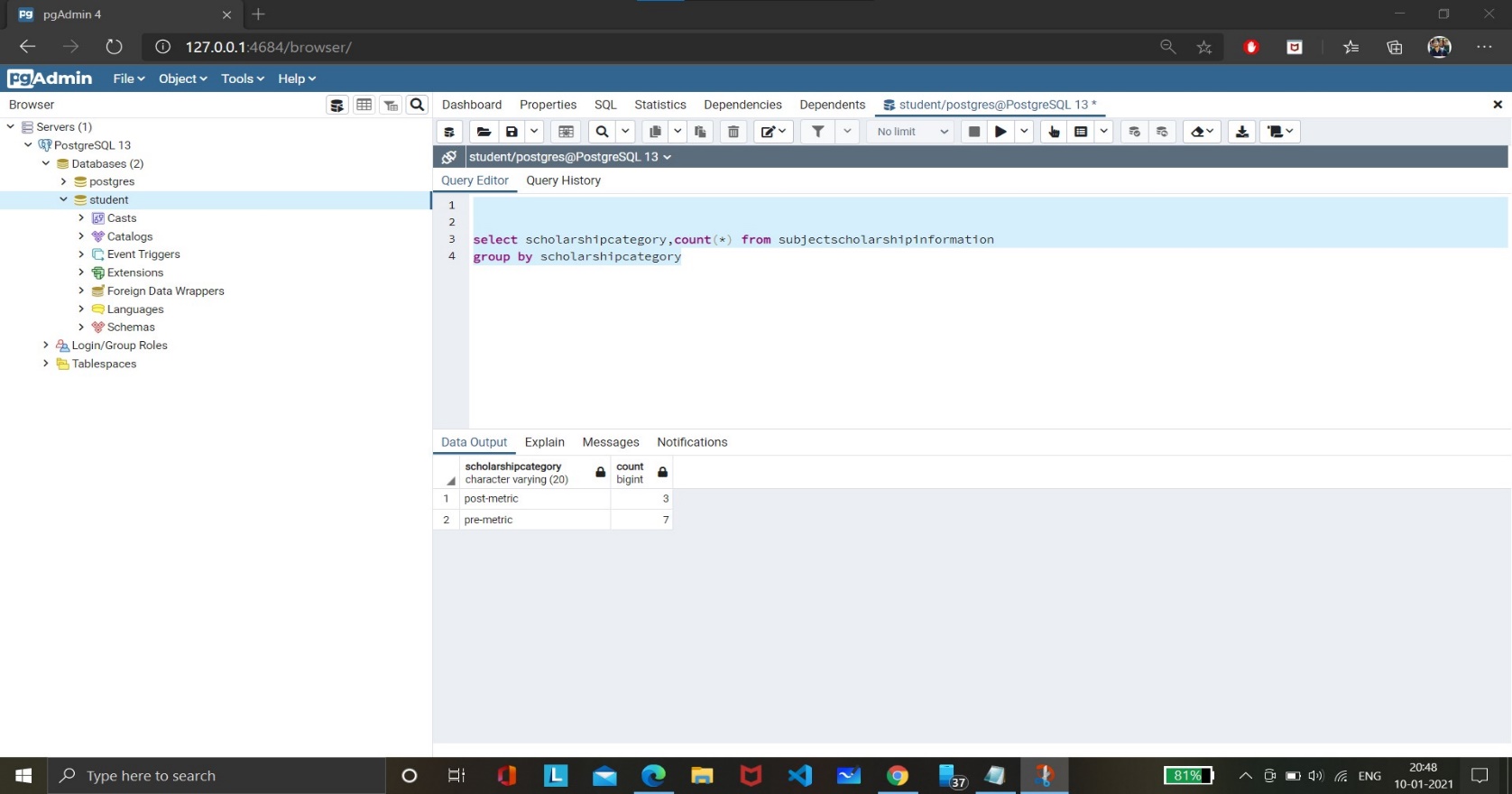
**DELETE :** delete removes the data from the table but the space is not removed. We can use rollback to restore the table.

**TRUNCATE:** truncate removes the data as well the space with it . but the table still remains.

1. Get the count of the Scholarship category which is highly been availed by the students, i.e. get the count of the total number of students corresponding to the each scholarships category



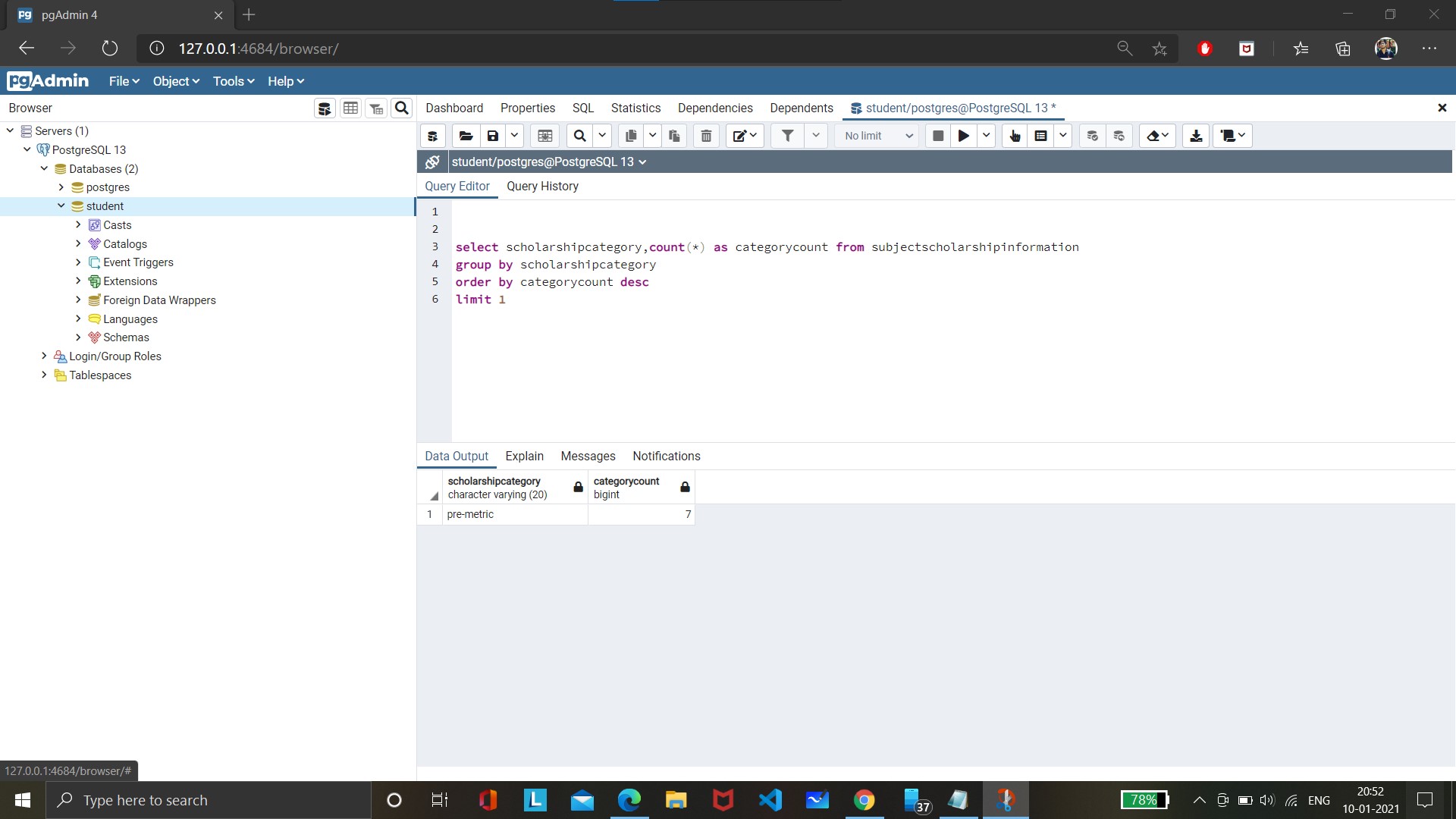




1. Along with the assignment no. 17 try to retrieve the maximum used scholarship category



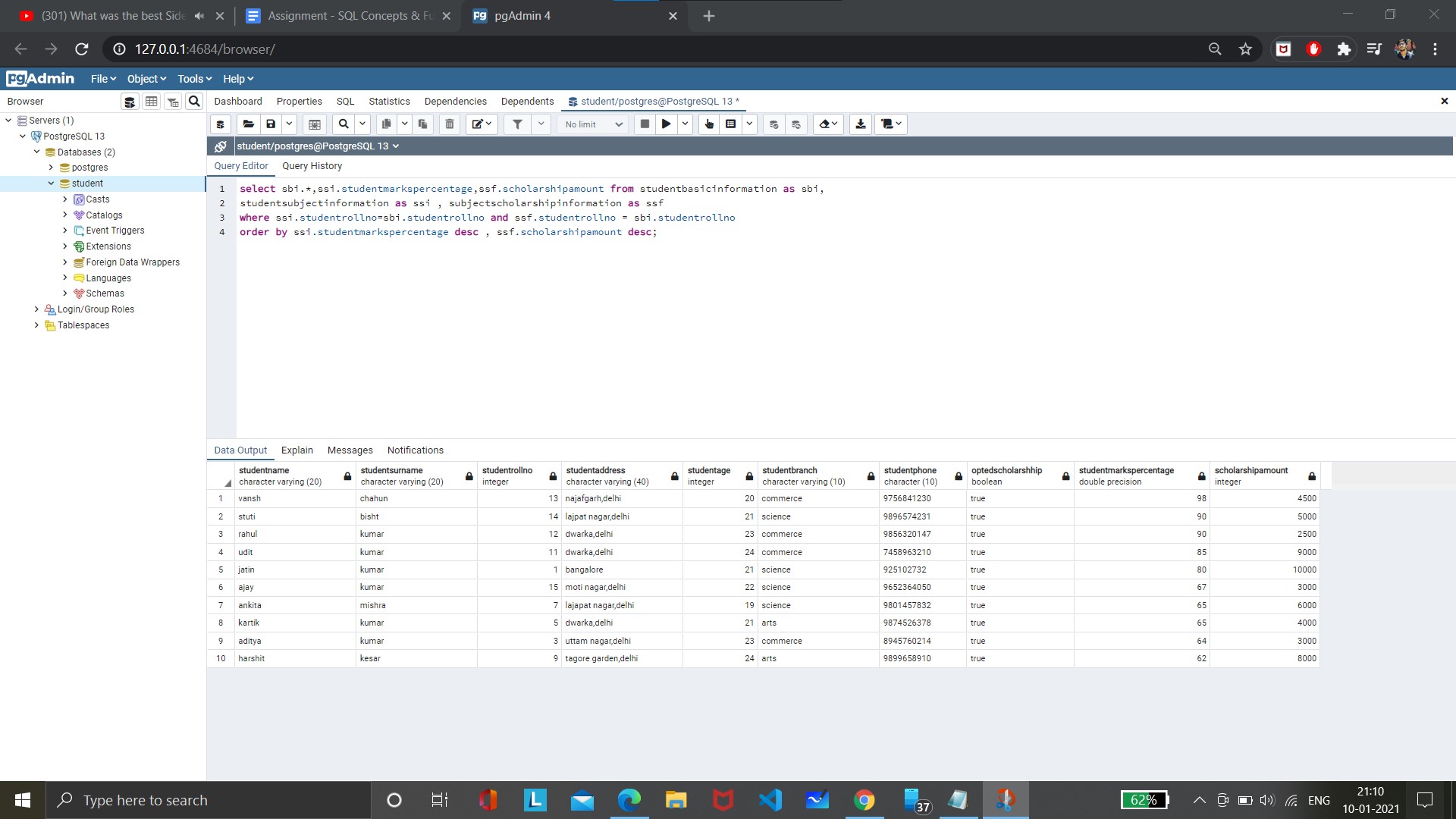




1. Retrieve the percentage of the students along with students detailed information who has scored the highest percentage along with availing the maximum scholarship amount







1. Difference between the Triggers, Stored Procedures, Views and Functions

**VIEW:** A view is a virtual table. It does not physically exist. Rather, it is created by a query joining one or more tables.

**TRIGGERS:** A trigger is a set of SQL statements in the database which automatically execute whenever any special event occurs in the database, like insert, delete, update, etc.

**STORED PROCEDURE:** A stored procedure is a set of pre-compiled Structured Query Languages (SQL), so it can be reused and shared by multiple programs. It can access or modify data in a database.it cannot return values. It only return values as modifying the values passed as parameters.

**FUNCTION:** A function is compiled and executed every time whenever it is called. A function must return a value and cannot modify the data received as parameters.

**Thank you. All The Best. Enjoy The Assignment.**