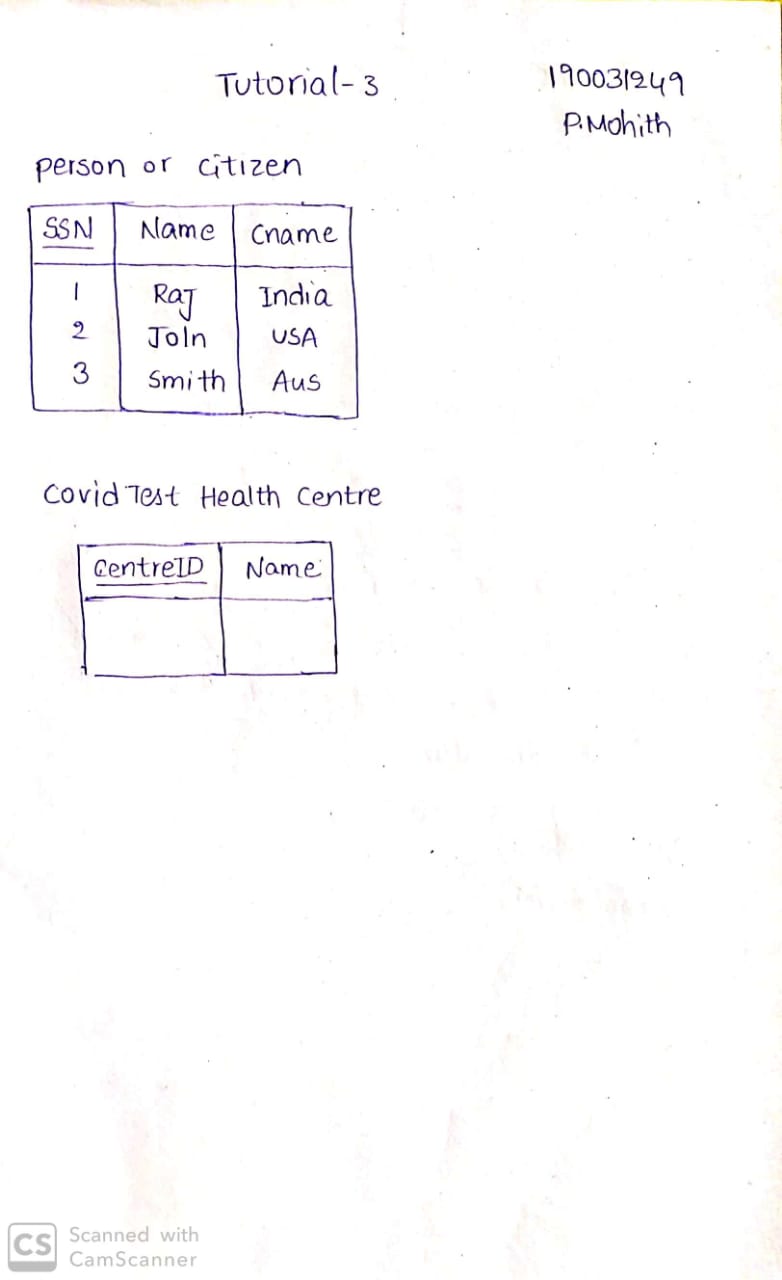
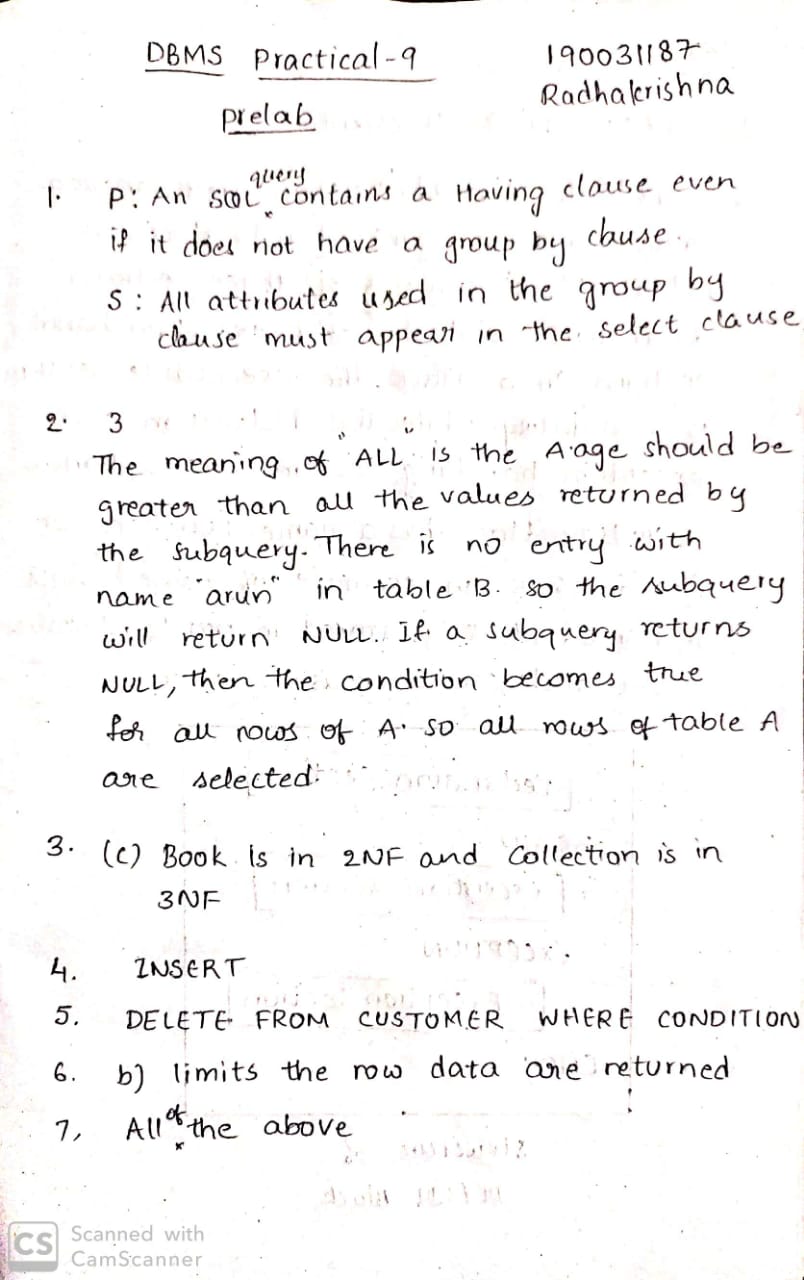
**EXPERIMENT 9**

**PRE-LAB**

1)Which of the following statements are TRUE about an SQL query?  
P : An SQL query can contain a HAVING clause even if it does not have a GROUP BY clause  
Q : An SQL query can contain a HAVING clause only if it has a GROUP BY clause  
R : All attributes used in the GROUP BY clause must appear in the SELECT clause  
S : Not all attributes used in the GROUP BY clause need to appear in the SELECT clause

(A) P and R  
(B) P and S  
(C) Q and R  
(D) Q and S

**ANS) P and S**

****

2)Table A

Id Name Age

--------------------

12 Arun 60

15 Shreya 24

99 Rohit 11

Table B

Id Name Age

--------------------

15 Shreya 24

25 Hari 40

98 Rohit 20

99 Rohit 11

Table C

Id Phone Area

--------------------

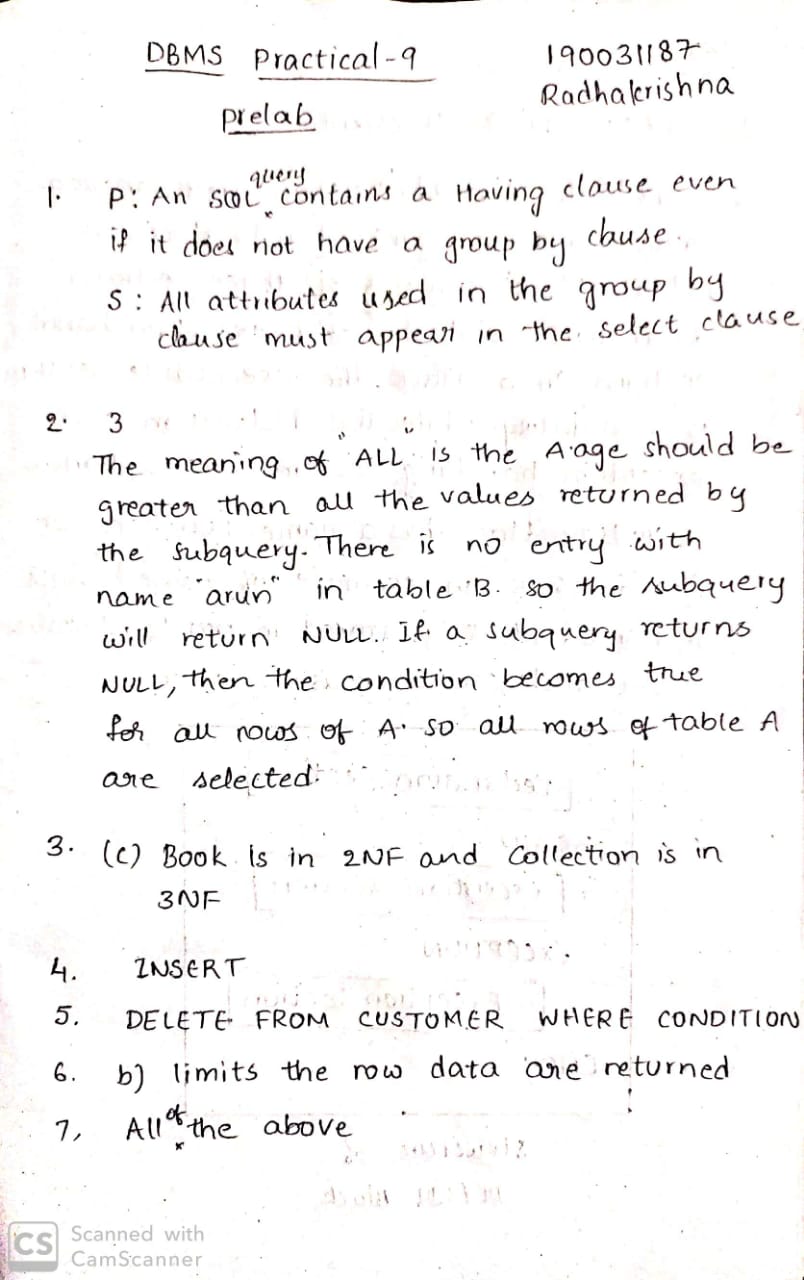
10 2200 02

99 2100 01

Consider the above tables A, B and C. How many tuples does the result of the following SQL query contains?

SELECT A.id FROM A WHERE A.age > ALL (SELECT B.age FROM B WHERE B. name = "arun")

**ANS) 3**



3) Consider the following relational schemes for a library database:

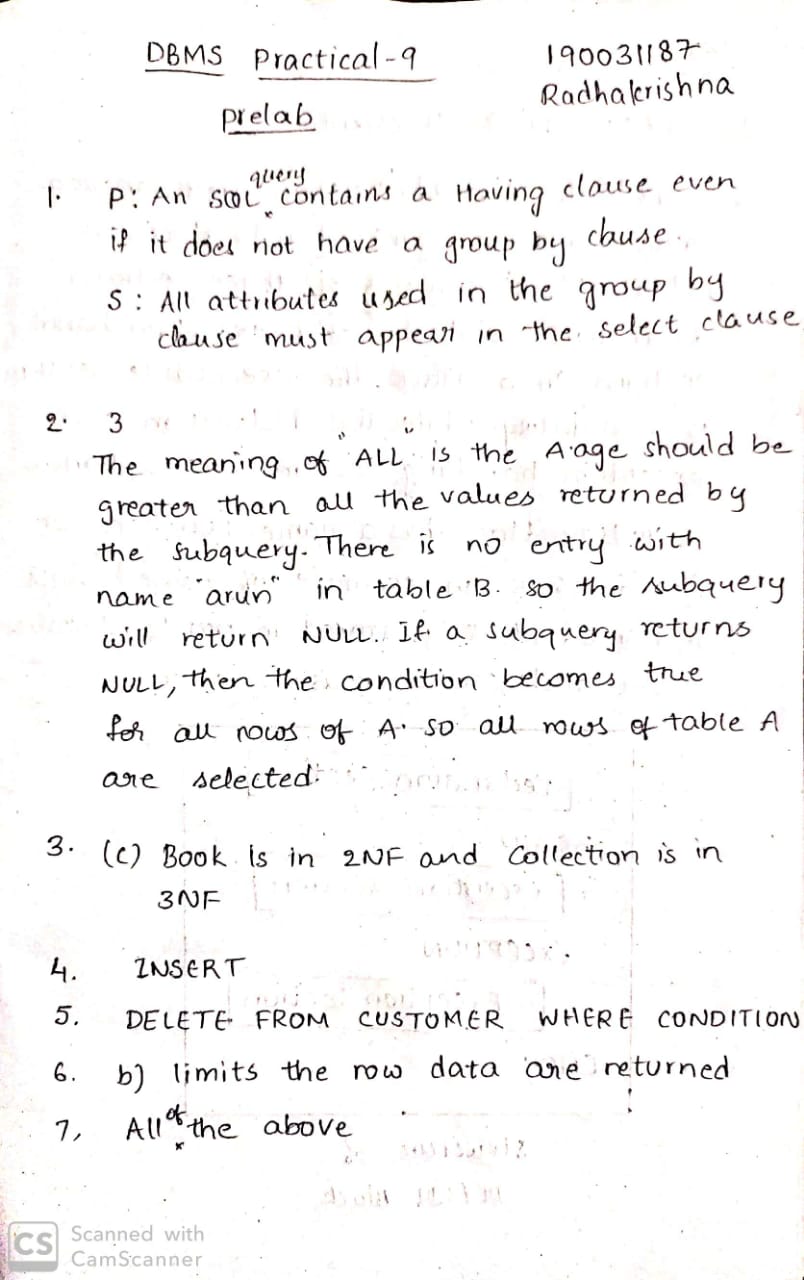
Book (Title, Author, Catalog\_no, Publisher, Year, Price)  
Collection (Title, Author, Catalog\_no)

within the following functional dependencies:

1. Title Author --> Catalog\_no
2. Catalog\_no --> Title Author Publisher Year
3. Publisher Title Year --> Price

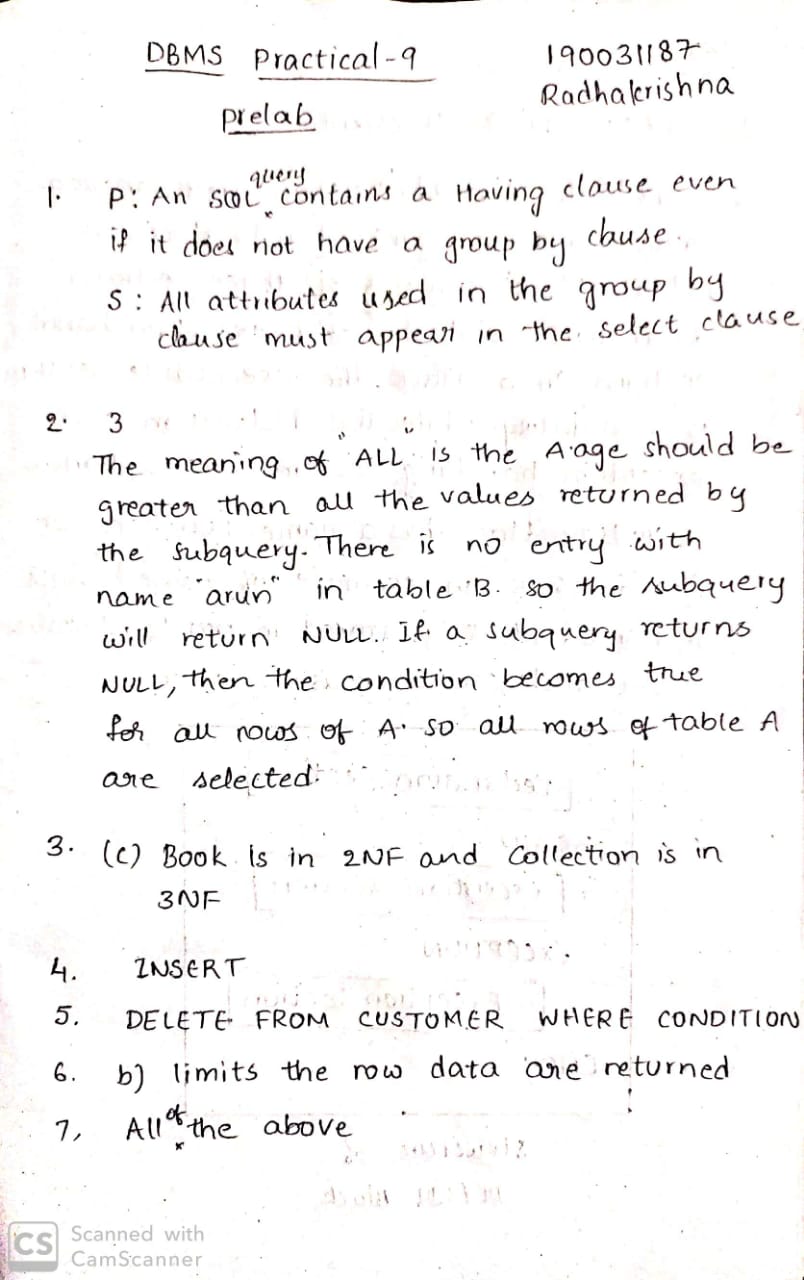
Assume {Author, Title} is the key for both schemes. Which of the following statements is true?  
(A) Both Book and Collection are in BCNF  
(B) Both Book and Collection are in 3NF only  
(C) Book is in 2NF and Collection is in 3NF  
(D) Both Book and Collection are in 2NF only

**ANS) (C) Book is in 2NF and Collection is in 3NF**



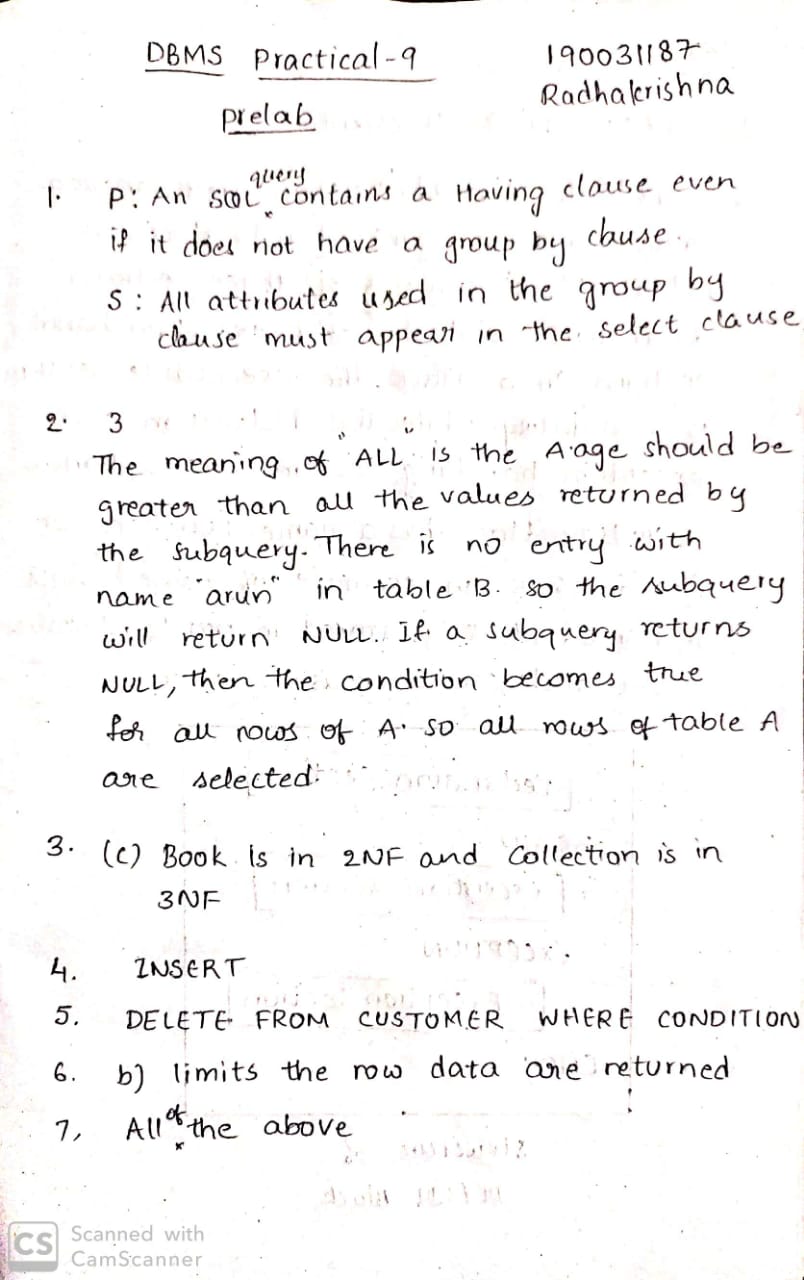
4) You can add a row using SQL in a database by using which statement

**ANS) INSERT**



5) The command to remove rows from a table 'CUSTOMER' is \_\_\_\_

**ANS) DELETE FROM CUSTOMER WHERE {CONDITION}**

`

6) The SQL WHERE clause:

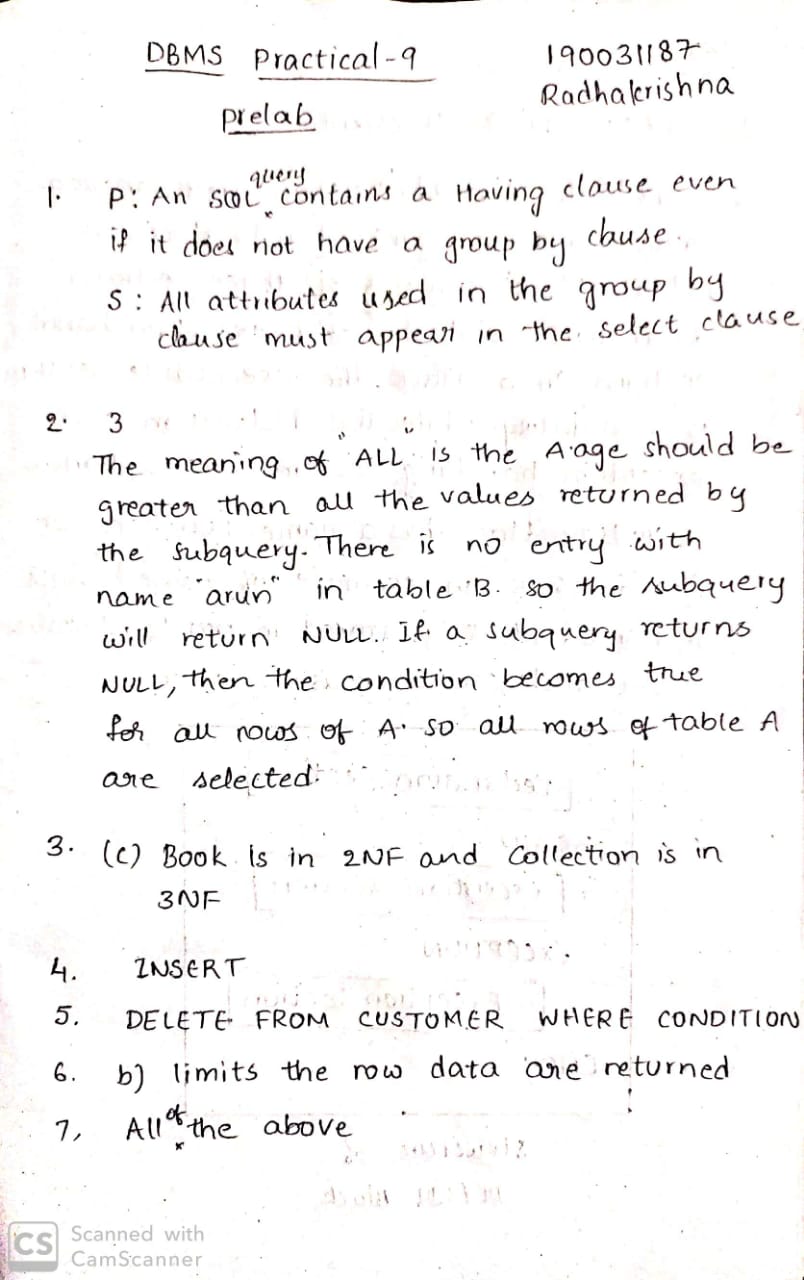
a) limits the column data that are returned

b) limits the row data are returned.

c) Both A and B are correct.

d) Neither A nor B are correct

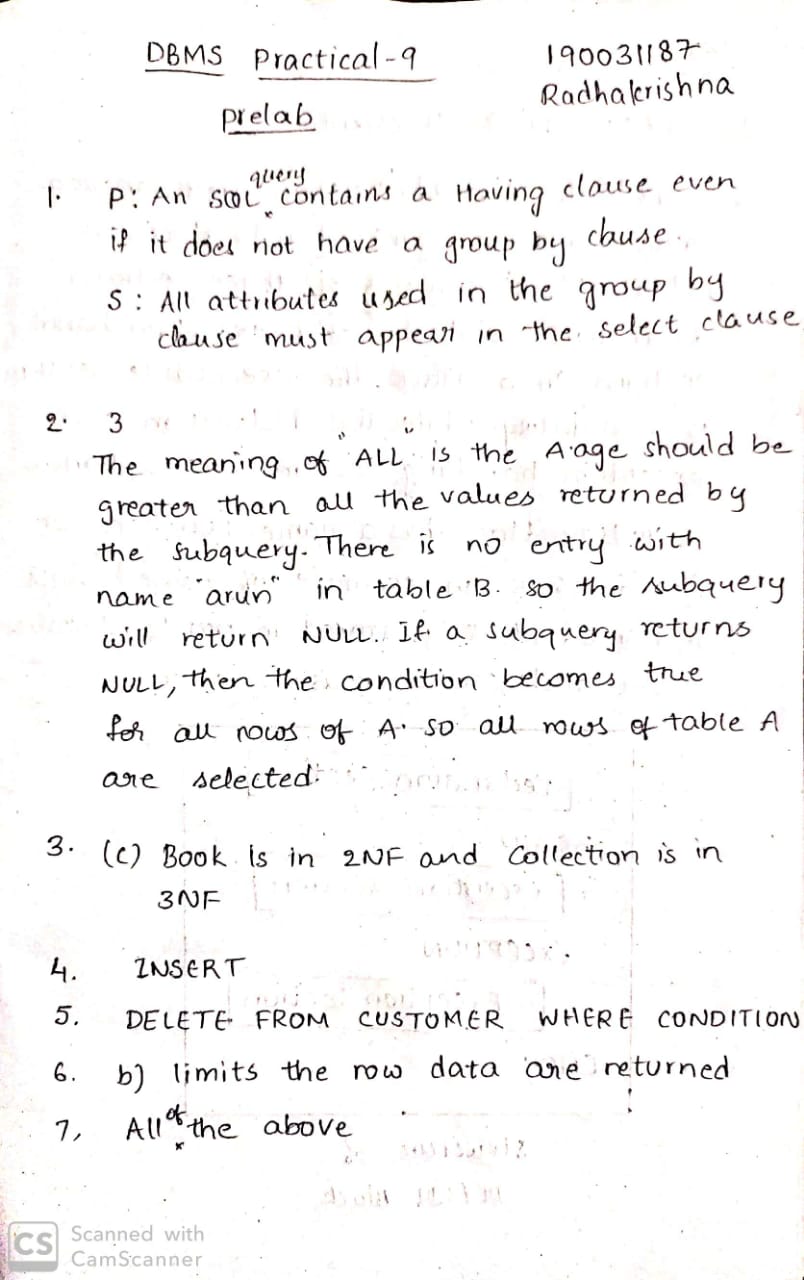
**ANS) b) limits the row data are returned.**



7) An action assertion must include which of the following?

a) Anchor object b) Action c) Corresponding object d) All of the above.

**ANS) d) All of the above.**



**IN-LAB**

**Case Study 1: TRANSPORT DEPARTMENT**

1) Create a cursor to display all the customer details of a particular branch

delimiter @

create procedure cust\_veh\_details()

begin

declare ve\_id int;

declare v\_finished int default 0;

declare c1 cursor for select veh\_id from contract\_permission;

declare continue handler for not found set v\_finished=1;

open c1;

v\_details:loop

fetch c1 into ve\_id;

if v\_finished=1 then

leave v\_details;

end if;

select \* from customer c,vehicle v where v.veh\_id=ve\_id and c.v\_id=ve\_id;

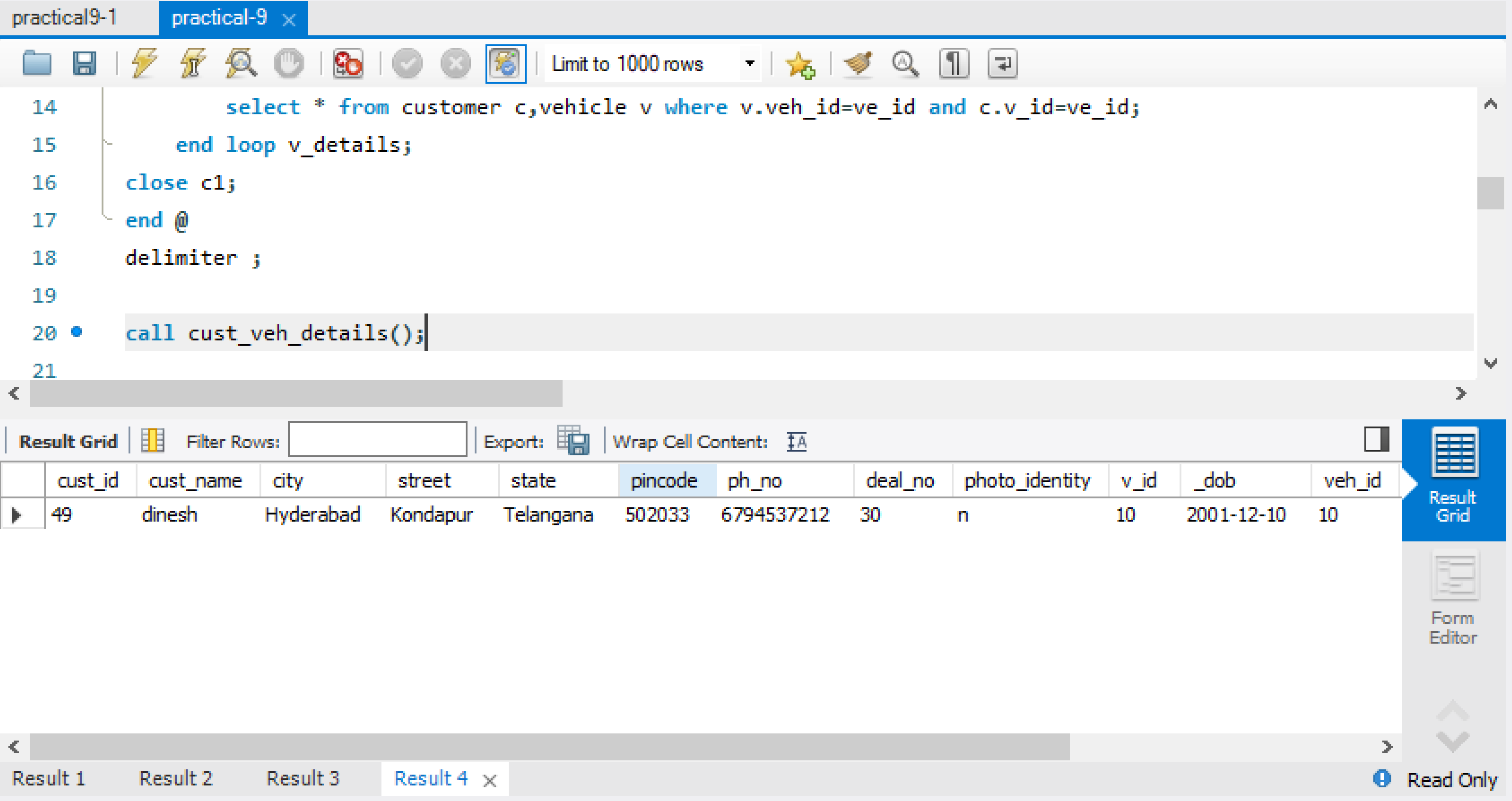
end loop v\_details;

close c1;

end @

delimiter ;

call cust\_veh\_details();



2) Create a cursor to display the customer details along with his vehicle details which are given contract permission.

delimiter $@

create procedure branch\_cust\_details()

begin

declare cus\_id int;

declare b\_finished int default 0;

declare c1 cursor for select c\_id from branch;

declare continue handler for not found set b\_finished=1;

open c1;

b\_c\_details:loop

fetch c1 into cus\_id;

if b\_finished=1 then

leave b\_c\_details;

end if;

select \* from customer where cust\_id=cus\_id;

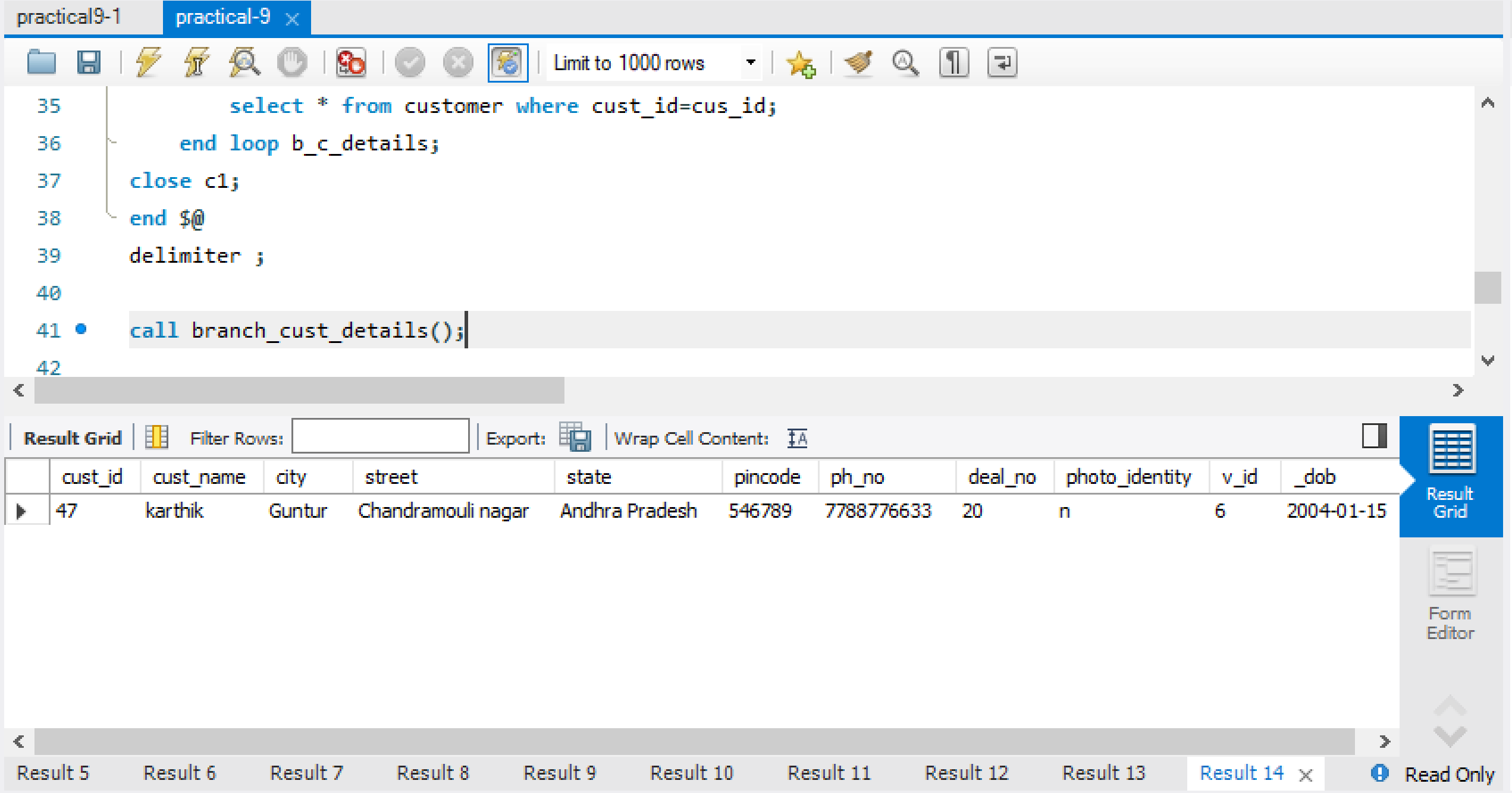
end loop b\_c\_details;

close c1;

end $@

delimiter ;

call branch\_cust\_details();



3) Create a cursor to display the customer details who are under a particular dealer.

delimiter $$

create procedure cust\_details\_under\_dealer()

begin

declare cust\_id int;

declare cust\_name varchar(10);

declare cust\_dob varchar(10);

declare cust\_city varchar(25);

declare cust\_street varchar(100);

declare cust\_state varchar(25);

declare pincode numeric;

declare cust\_phno bigint;

declare deal\_no int;

declare photo\_identity varchar(1);

declare v\_id int;

declare deal\_id int;

declare deal\_name varchar(10);

declare c\_finished integer default 0;

declare c1 cursor for select c.\*,d.deal\_id,d.deal\_name from Customer c inner join registration r on r.cust\_id = c.cust\_id inner join Dealer d on r.deal\_id = d.deal\_id;

declare continue handler for NOT FOUND set c\_finished = 1;

open c1;

get\_customer: LOOP

fetch c1 into cust\_id,cust\_name,cust\_city,cust\_street,cust\_state,pincode,cust\_phno,deal\_no,photo\_identity,v\_id,cust\_dob,deal\_id,deal\_name;

if c\_finished = 1 then

leave get\_customer;

end if;

select concat(cust\_id,',',cust\_name,',',cust\_dob,',',cust\_street,',',cust\_state,',',pincode,',',cust\_phno,',',photo\_identity,',',v\_id,',',deal\_id,',',deal\_name);

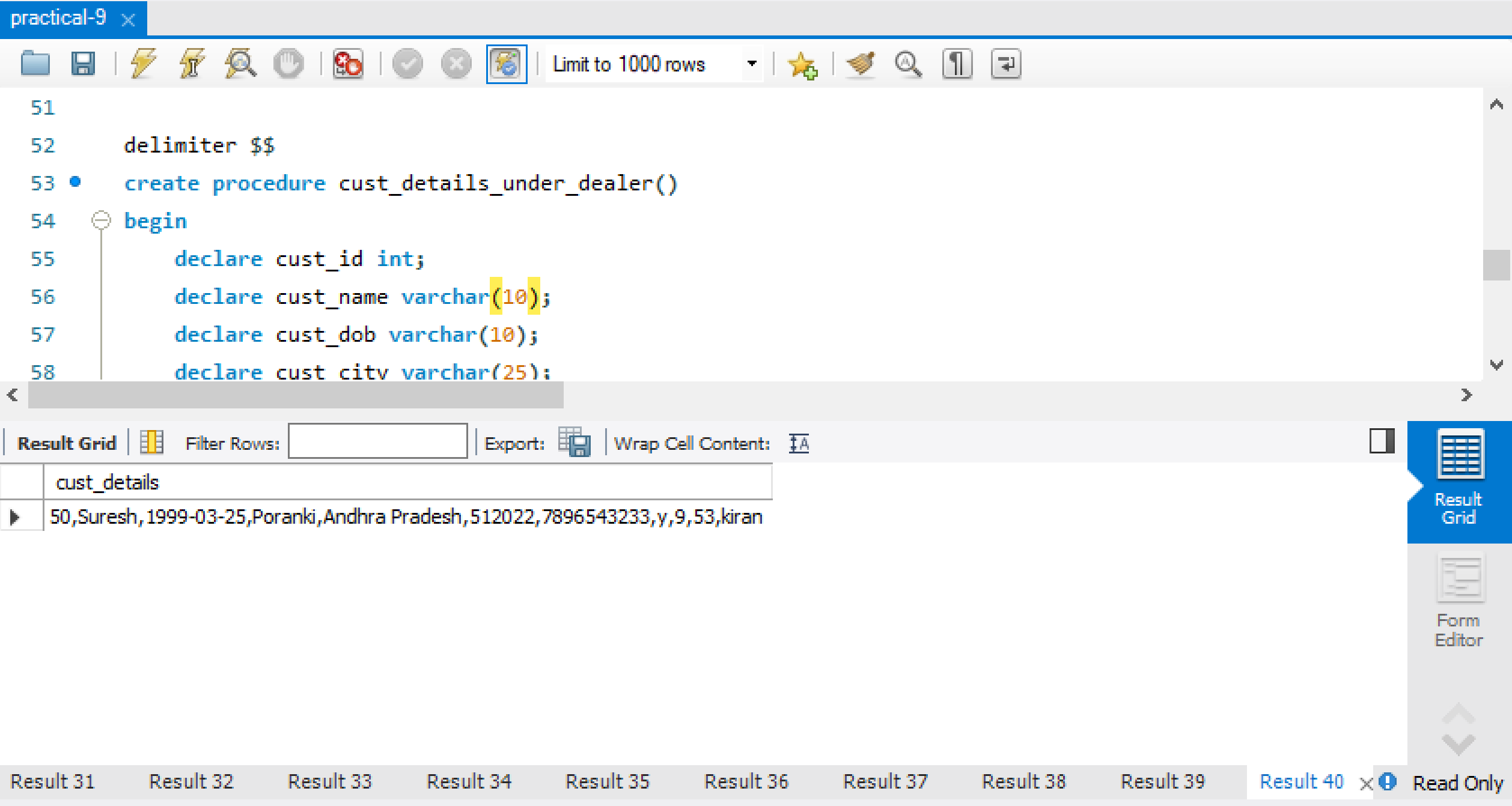
end loop get\_customer;

close c1;

end $$

delimiter ;

call cust\_details\_under\_dealer();

****

4) Create a procedure to display the educational vehicles applied for permit in a particular branch

ANS)delimiter $$

create procedure proc\_edu()

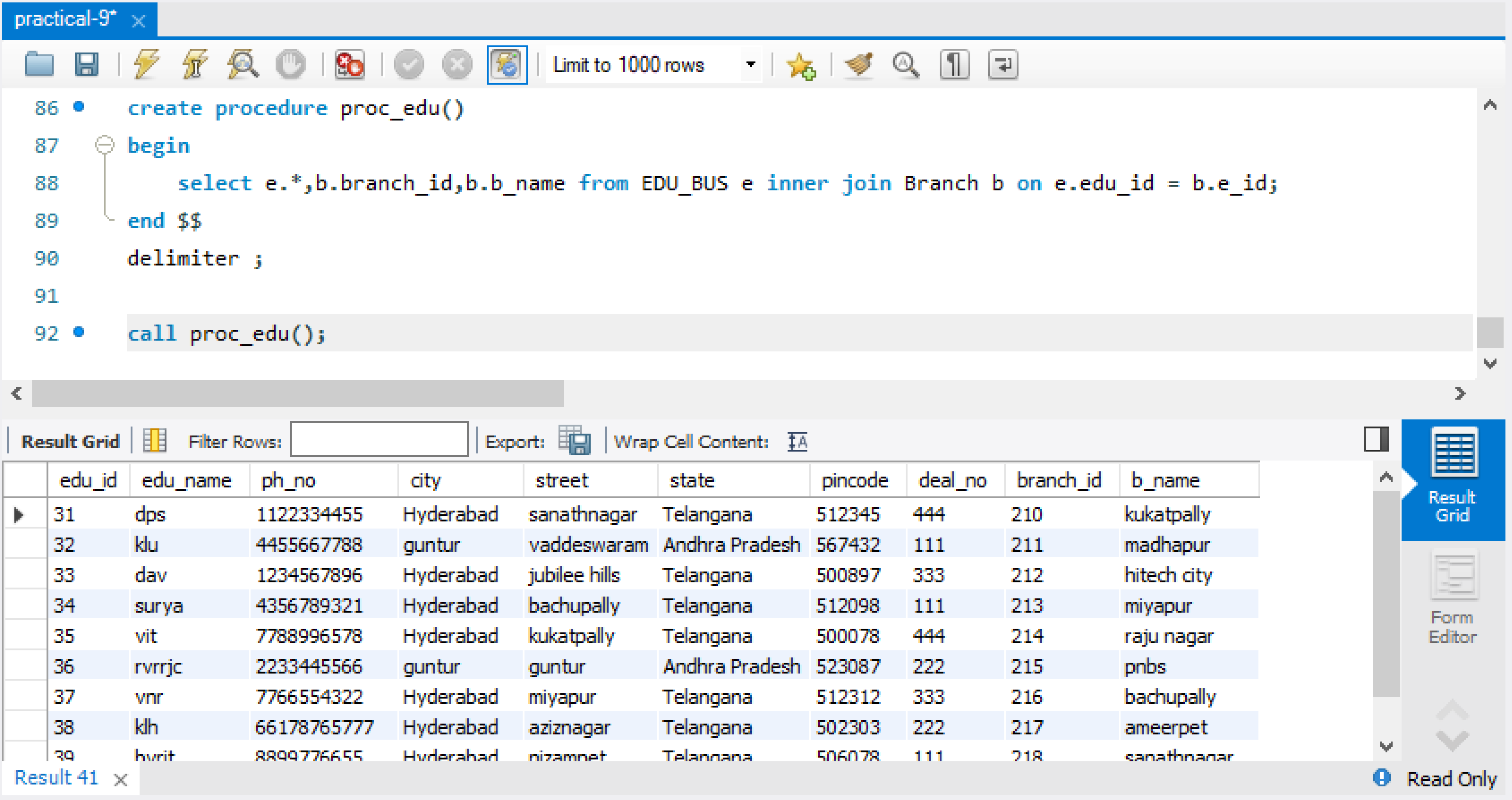
begin

select e.\*,b.branch\_id,b.b\_name from EDU\_BUS e inner join Branch b on e.edu\_id = b.e\_id;

end $$

delimiter ;

call proc\_edu();



5) Create a procedure to display the details of the branches in a particular state when state is given as input when executing the procedure

delimiter $$

create procedure branch\_details(in st varchar(20))

begin

select \* from branch where state=st;

end $$

delimiter ;

call branch\_details('Andhra pradesh');

**Case Study 4: KL UNIVERSITY ERP**

1. Create a cursor to display students details who register for a particular course

delimiter $$

drop procedure if exists proc\_stu;

create procedure proc\_stu()

begin

declare regno, year, semester, fid int;

declare name varchar(10);

declare Mobileno bigint;

declare email varchar(20);

declare address varchar(10);

declare branch varchar(10);

declare course\_code varchar(10);

declare s\_finished integer default 0;

declare c1 cursor for select s.REGNO,s.NAME,s.Mobileno,s.EMAIL,s.Address,s.Branch,st.YEAR,st.SEMESTER,st.COURSECODE,st.FID from Student s inner join Stu\_Reg\_Courses st on st.REGNO = s.REGNO;

declare continue handler for NOT FOUND set s\_finished = 1;

open c1;

get\_stu: LOOP

fetch c1 into regno,name,Mobileno,email,address,branch,year,semester,course\_code,fid;

if s\_finished = 1 then

leave get\_stu;

end if;

select concat(regno,',',name,',',Mobileno,',',email,',',address,',',branch,',',year,',',semester,',',course\_code,',',fid) as student\_course\_details;

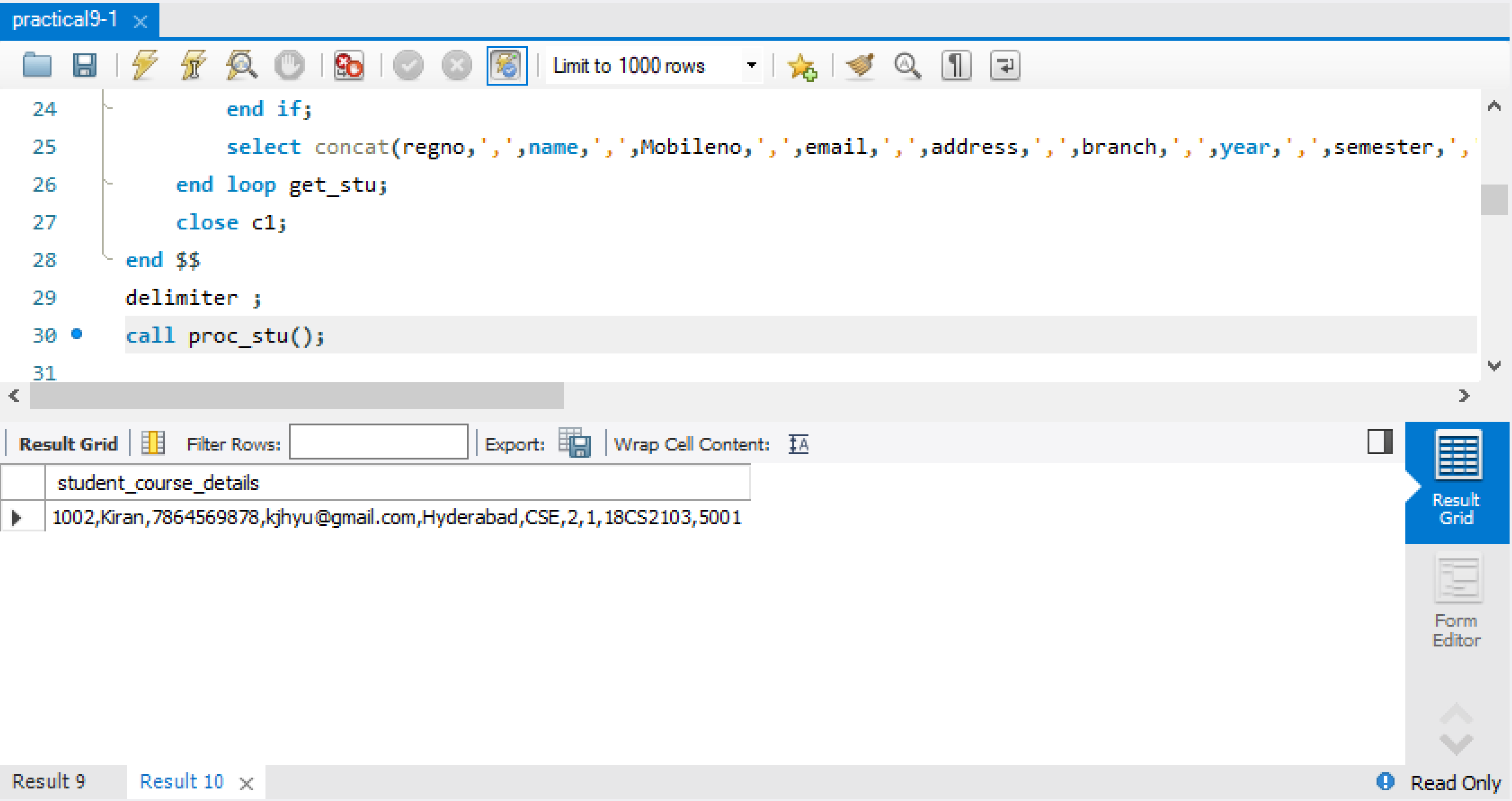
end loop get\_stu;

close c1;

end $$

delimiter ;

call proc\_stu();



1. Create a procedure to display the fee details of the student

delimiter $$

create procedure fee\_details()

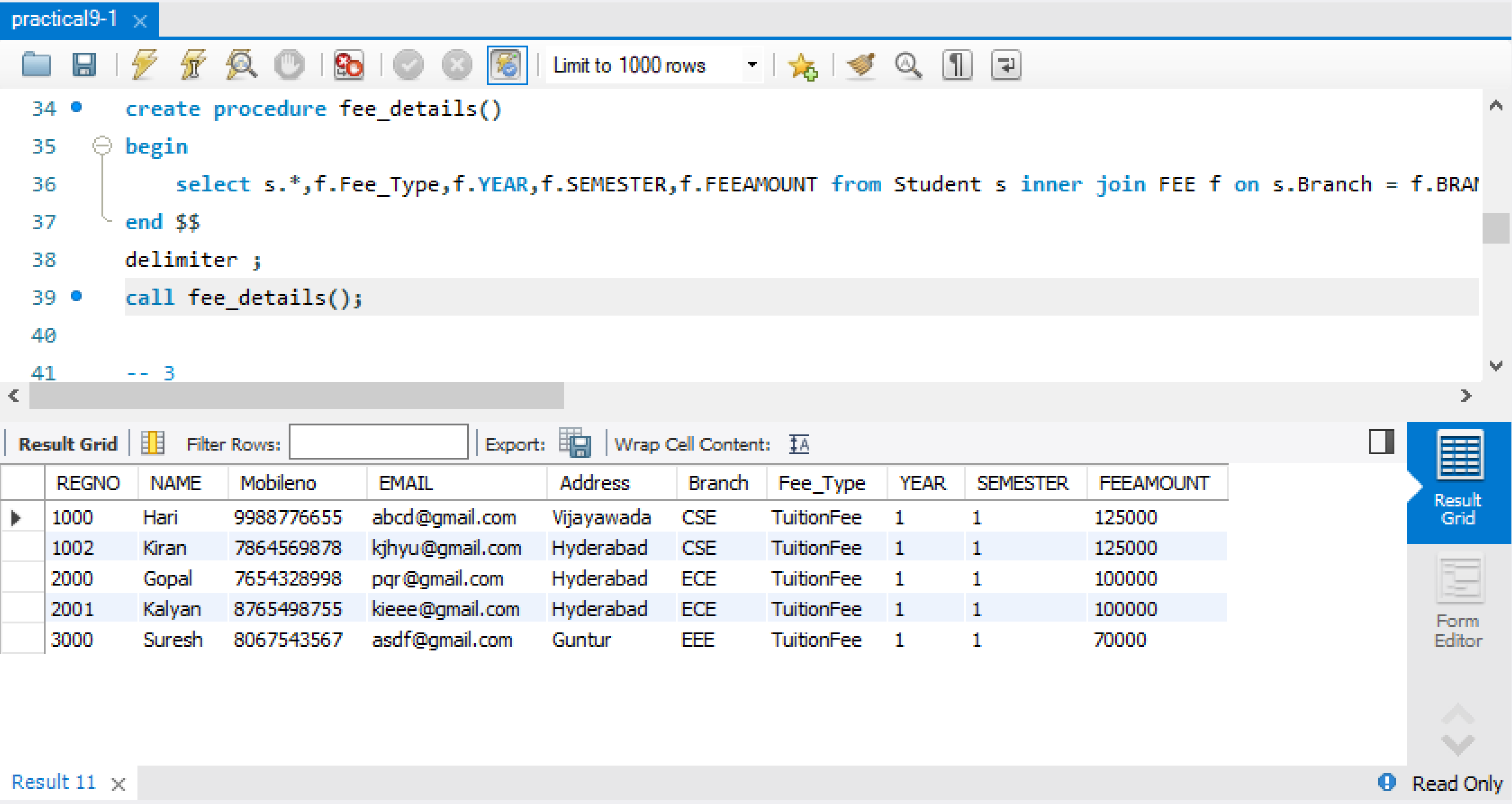
begin

select s.\*,f.Fee\_Type,f.YEAR,f.SEMESTER,f.FEEAMOUNT from Student s inner join FEE f on s.Branch = f.BRANCH;

end $$

delimiter ;

call fee\_details();



1. Create a trigger that will store the deleted student records in a log file

create table student\_log(deleted\_regno int,deleted\_name varchar(10),deleted\_mobile\_no bigint,deleted\_stu\_email varchar(20),deleted\_stu\_address varchar(20),deleted\_stu\_branch varchar(10));

delimiter $$

create trigger trig\_stu\_delete after delete on Student

for each row

begin

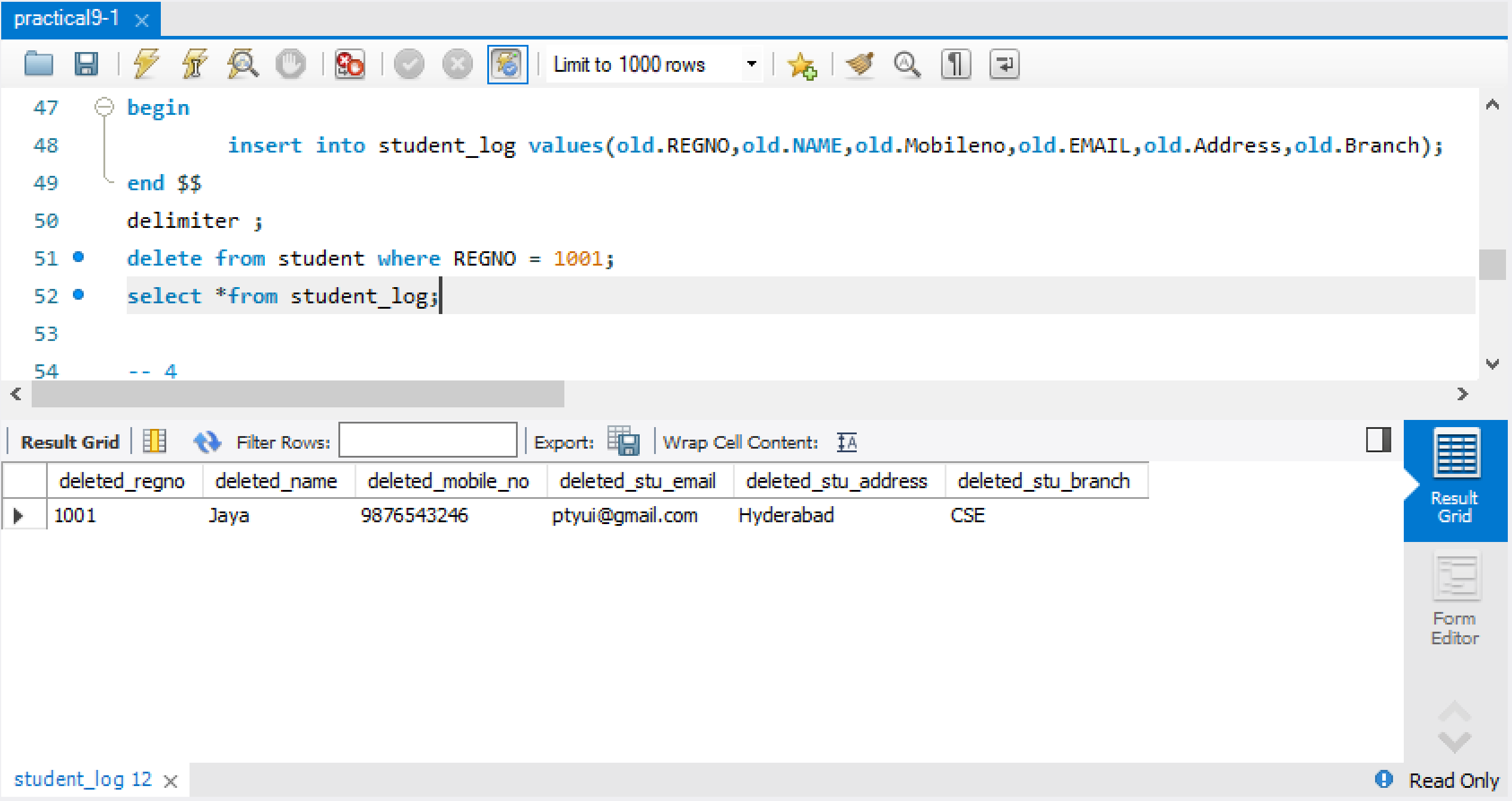
insert into student\_log values(old.REGNO,old.NAME,old.Mobileno,old.EMAIL,old.Address,old.Branch);

end $$

delimiter ;

delete from student where REGNO = 1001;

select \*from student\_log;



1. Create a cursor to update faculty salary with 1500 and display the updated details of faculty

delimiter $$

create procedure update\_Faculty\_salary()

begin

declare F\_ID int;

declare f\_finished integer default 0;

declare c1 cursor for select FID from Faculty;

declare continue handler for NOT FOUND set f\_finished = 1;

open c1;

get\_Faculty: LOOP

fetch c1 into F\_ID;

if f\_finished = 1 then

leave get\_Faculty;

end if;

update Faculty set Salary = Salary + 1500 where FID = F\_ID;

end loop get\_Faculty;

close c1;

end $$

delimiter ;

-- before update

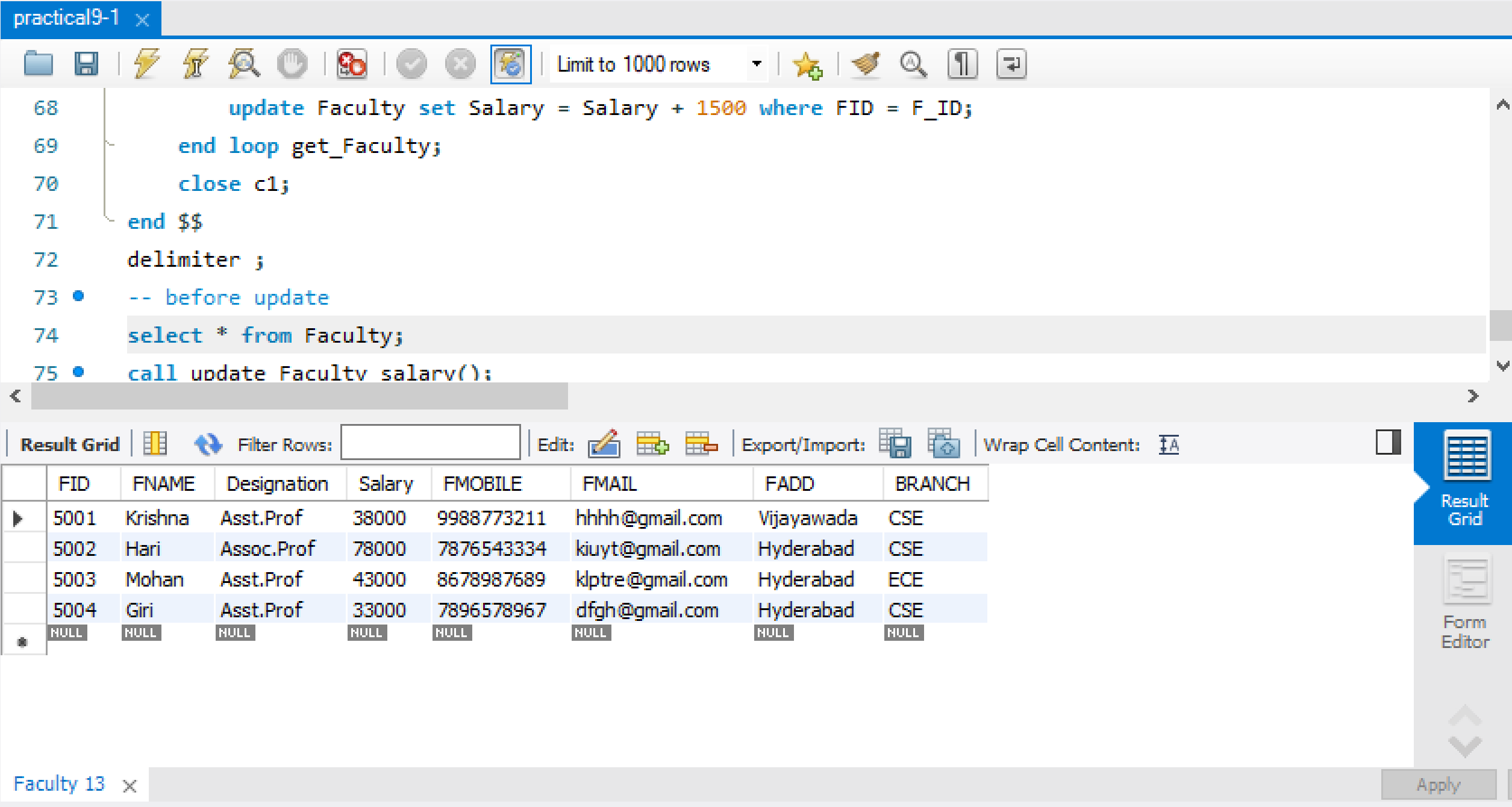
select \* from Faculty;

call update\_Faculty\_salary();

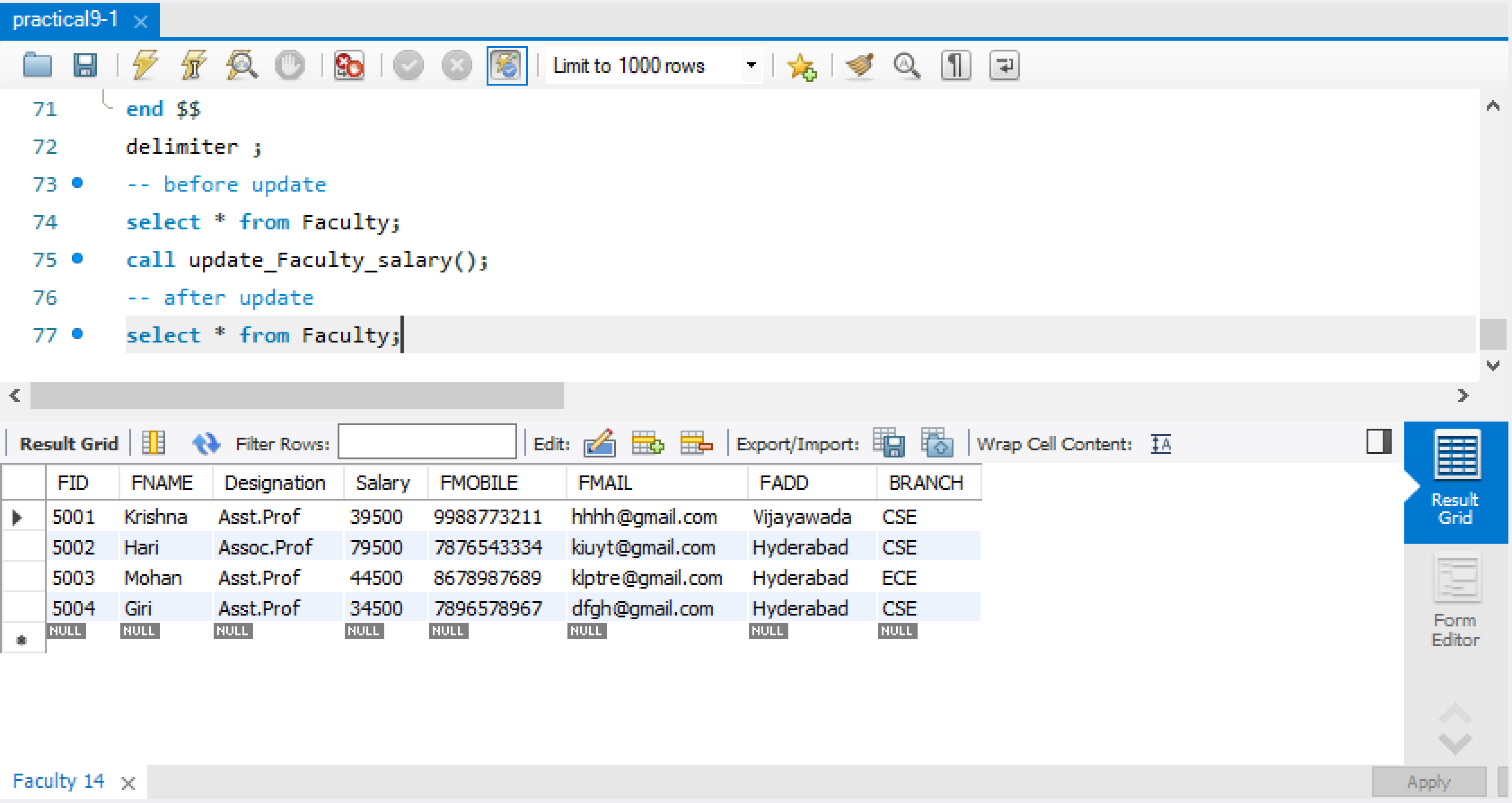
-- after update

select \* from Faculty;

BEFORE UPDATE

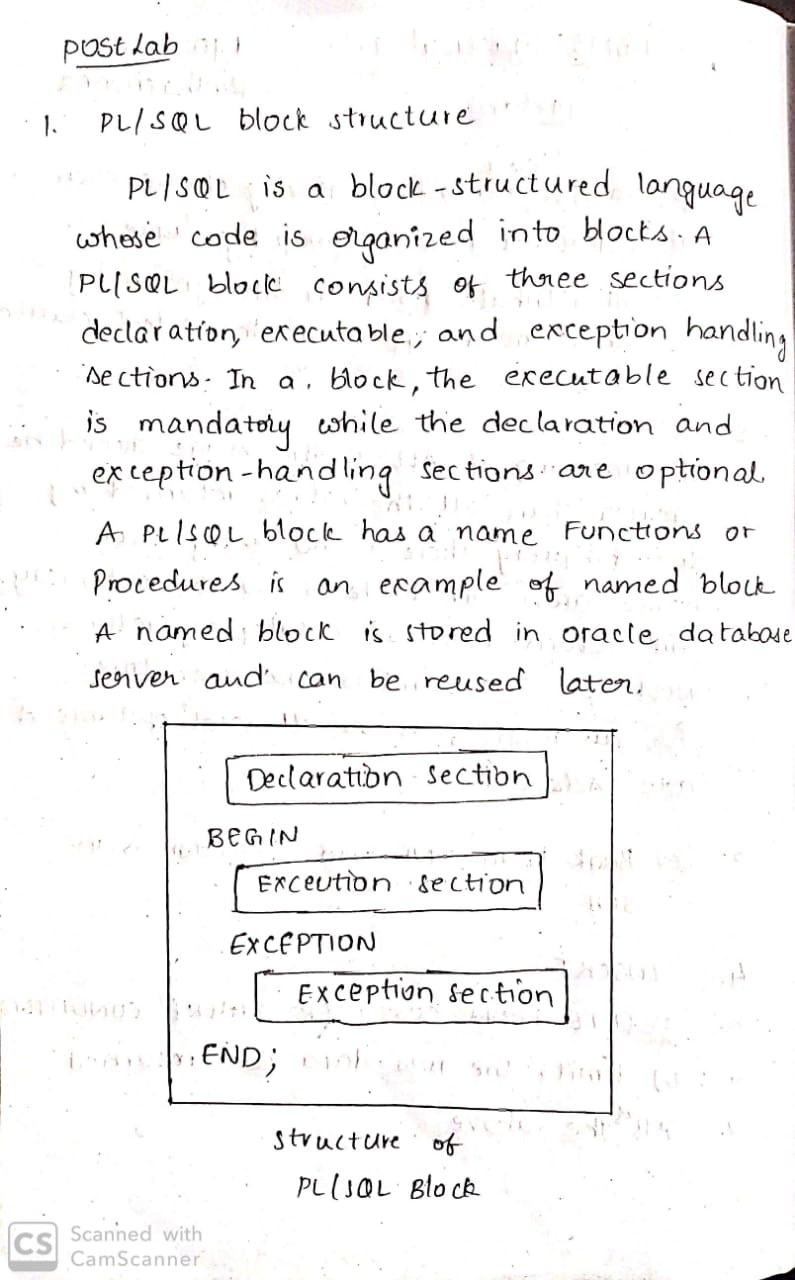


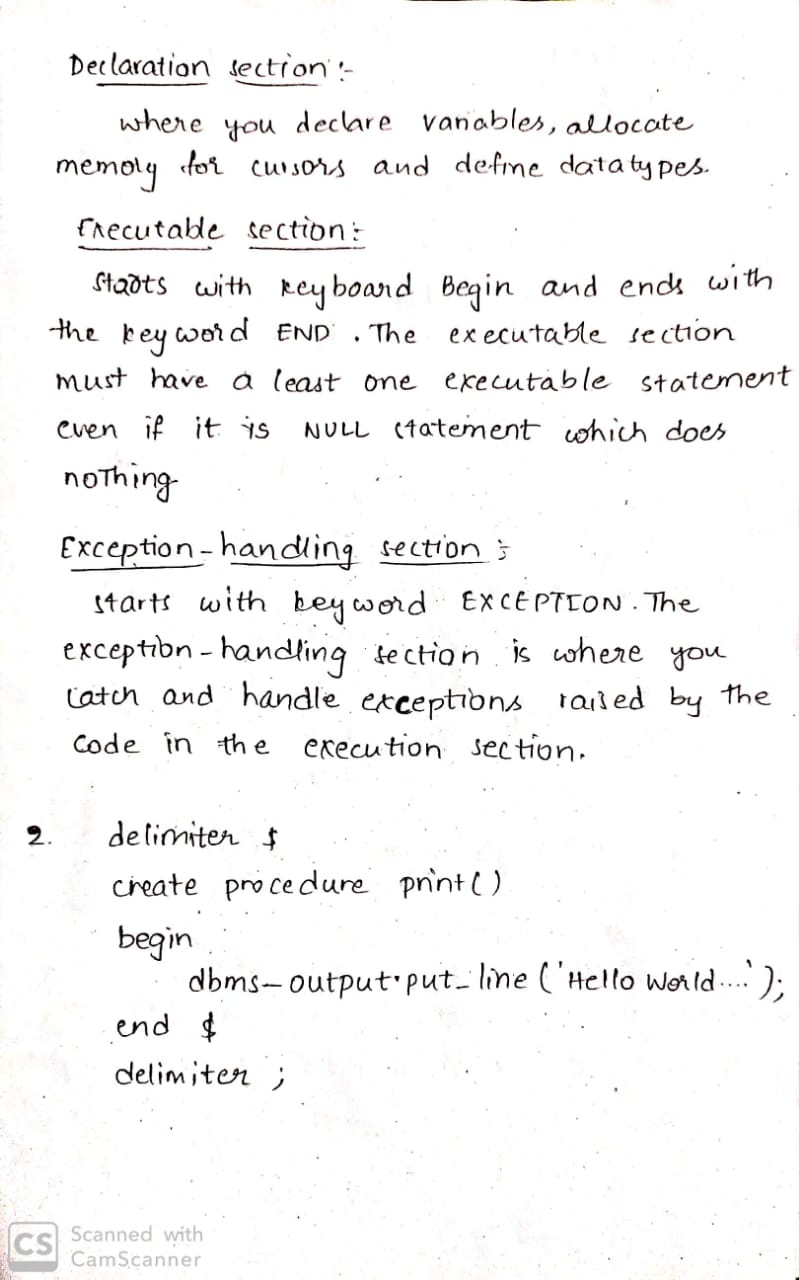
AFTER UPDATE



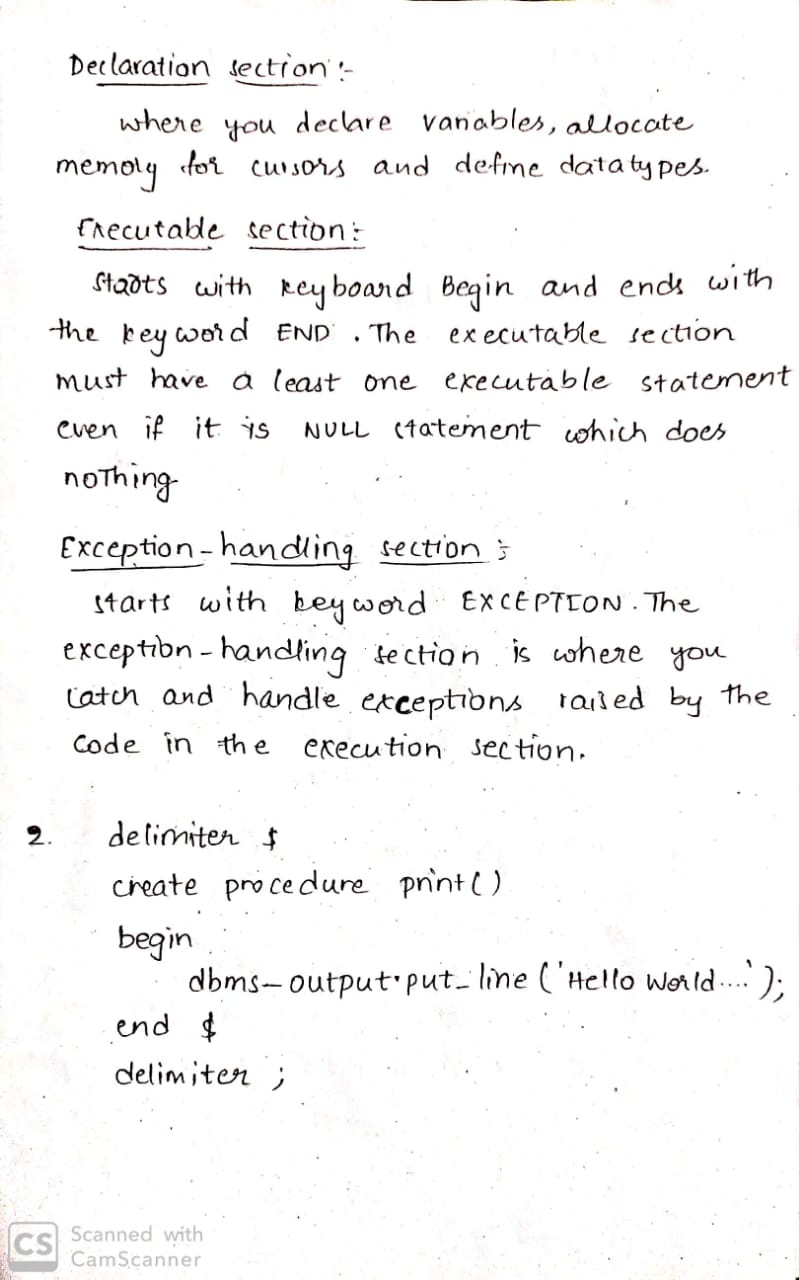
**POSTLAB**

1. Determine basic structure of a PL/SQL block. And define it briefly





2. write a small query to print hello world by using PL/SQL Block.



3. prepare query to understand loops by using for loop to print

i is: 1 and j is: 1

i is: 1 and j is: 2

i is: 1 and j is: 3

i is: 2 and j is: 1

i is: 2 and j is: 2

i is: 2 and j is: 3

i is: 3 and j is: 1

i is: 3 and j is: 2

i is: 3 and j is: 3

**ANS)**

**delimiter $**

**create procedure loops()**

**begin**

**declare i,j int;**

**set i=1;**

**l1:loop**

**set j=1;**

**l2:loop**

**select concat('i is:',i,' j is:',j);**

**set j=j+1;**

**if j>3 then**

**leave l2;**

**end if;**

**end loop l2;**

**set i=i+1;**

**if i>3 then**

**leave l1;**

**end if;**

**end loop l1;**

**end $**

**delimiter ;**

**call loops();**

4. A Query on PL/SQL to find LEAST number. This function accepts some parameters like exp1, exp2, … exp\_n. These each expression may be numbers or alphabets

**ANS)**

**delimiter @@**

**create function least\_(i float,j float,k float) returns float**

**begin**

**return least(i,j,k);**

**end @@**

**delimiter ;**

**select least\_(1,2,3);**

5.Query to find floor value. This function accepts a parameter number which is the input number on which FLOOR function

**ANS)**

**delimiter $@**

**create function floor\_(f float) returns int**

**begin**

**declare r int;**

**set r=floor(f);**

**return r;**

**end $@**

**delimiter ;**

**select floor\_(2.6);**