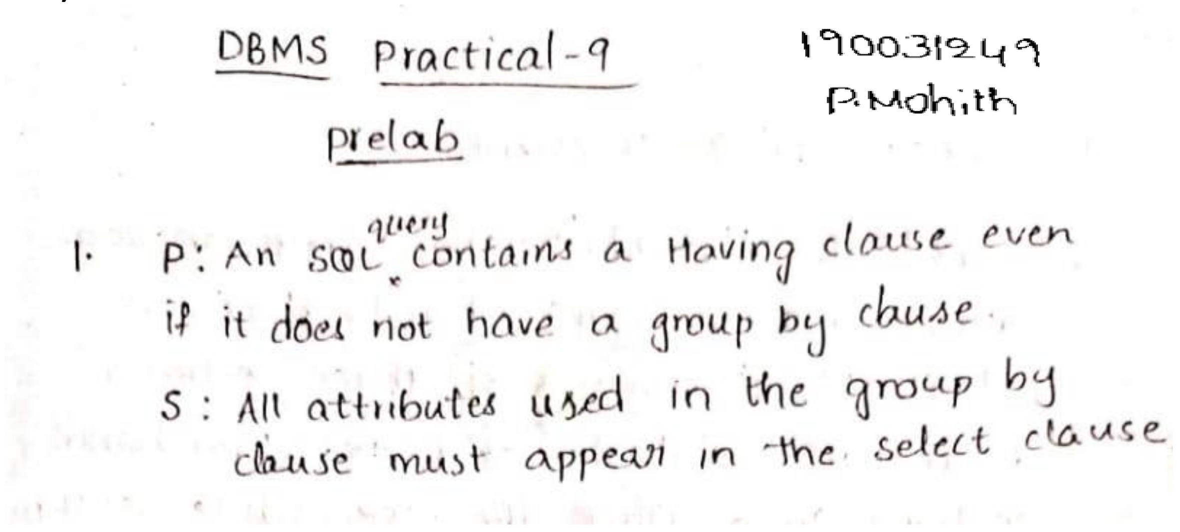
190031249 **P.MOHITH**

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



DBMS Practical-9 190031249
P.Mohith
prelab

1. P: An SQL query contains a Having clause even if it does not have a group by clause.
S : All attributes used in the group by clause must appear in the select clause

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

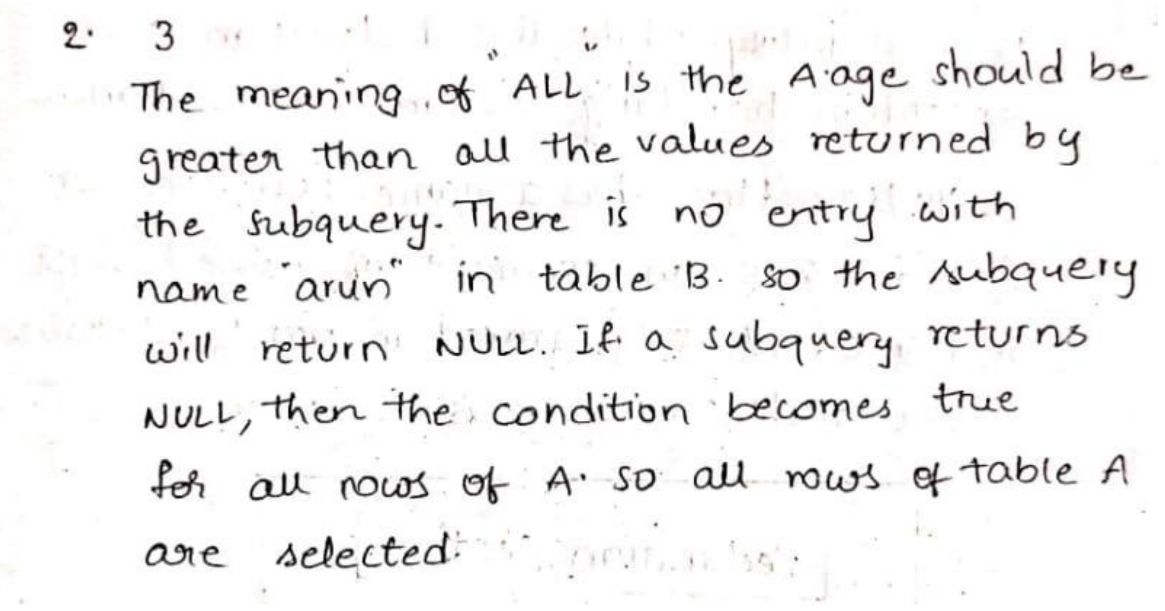
Table C

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

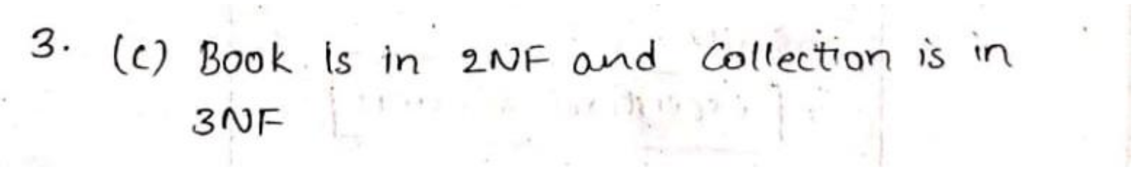

2. 3
The meaning of "ALL" is the A.age should be greater than all the values returned by the subquery. There is no entry with name "arun" in table B. so the subquery will return NULL. If a subquery returns NULL, then the condition becomes true for all rows of A. so all rows of table A are selected.

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:

I. Title Author --> Catalog_no
II. Catalog_no --> Title Author Publisher Year
III. 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


3. (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

190031249 **P.MOHITH**

4. INSERT

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

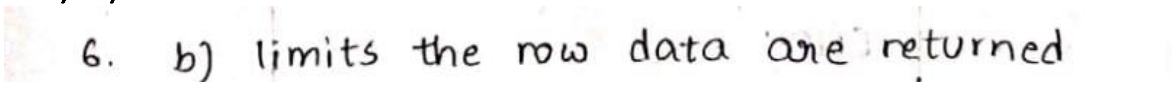
ANS) DELETE FROM CUSTOMER WHERE {CONDITION}

5. 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.


6. 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.

7, All of the above

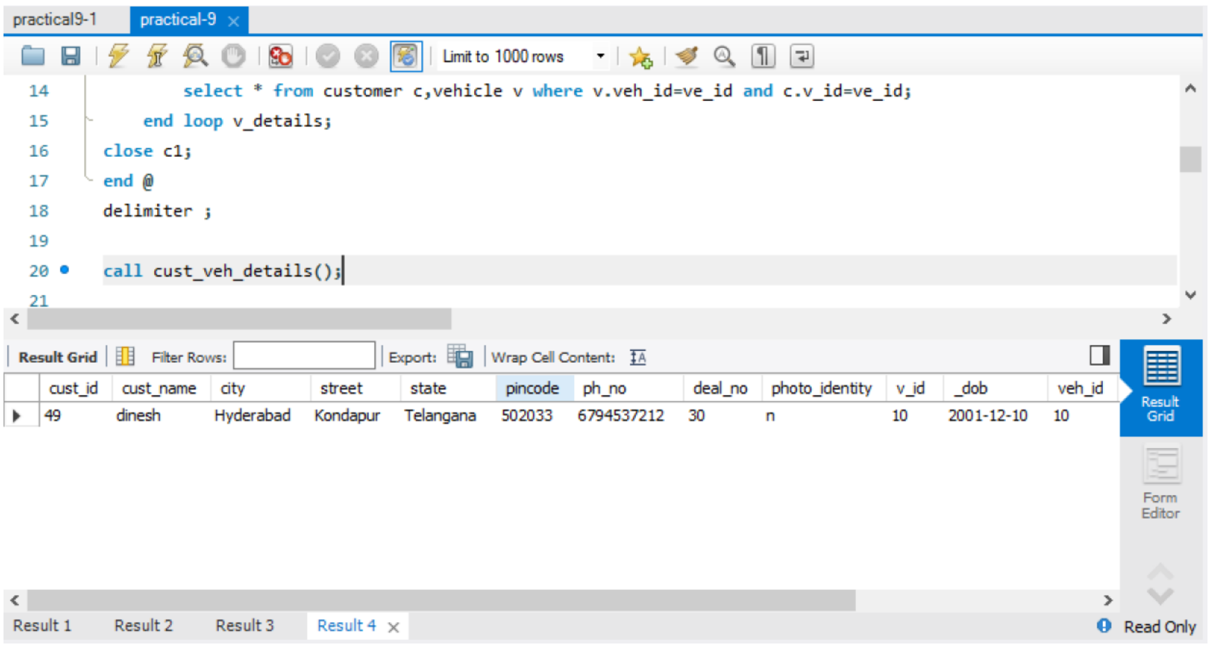
IN-LAB

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();
```



```
        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();
```

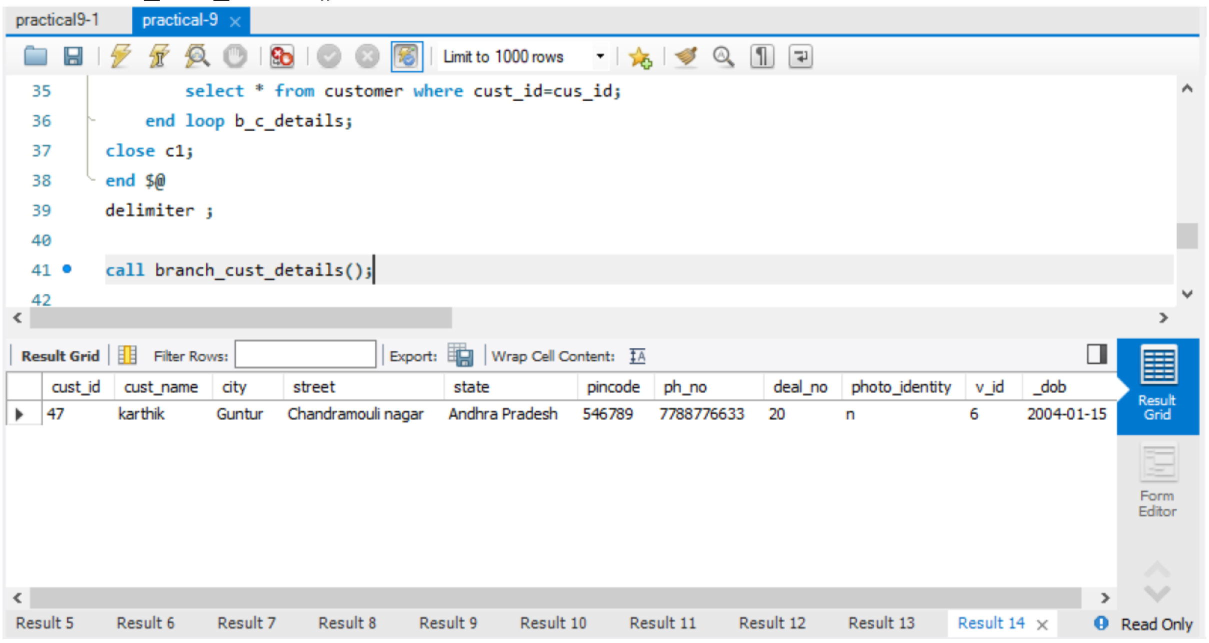
cust_id	cust_name	city	street	state	pincode	ph_no	deal_no	photo_identity	v_id	_dob	veh_id
49	dinesh	Hyderabad	Kondapur	Telangana	502033	6794537212	30	n	10	2001-12-10	10

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();
```



```
            select * from customer where cust_id=cus_id;
        end loop b_c_details;
    close c1;
    end $@
    delimiter ;

    call branch_cust_details();
```

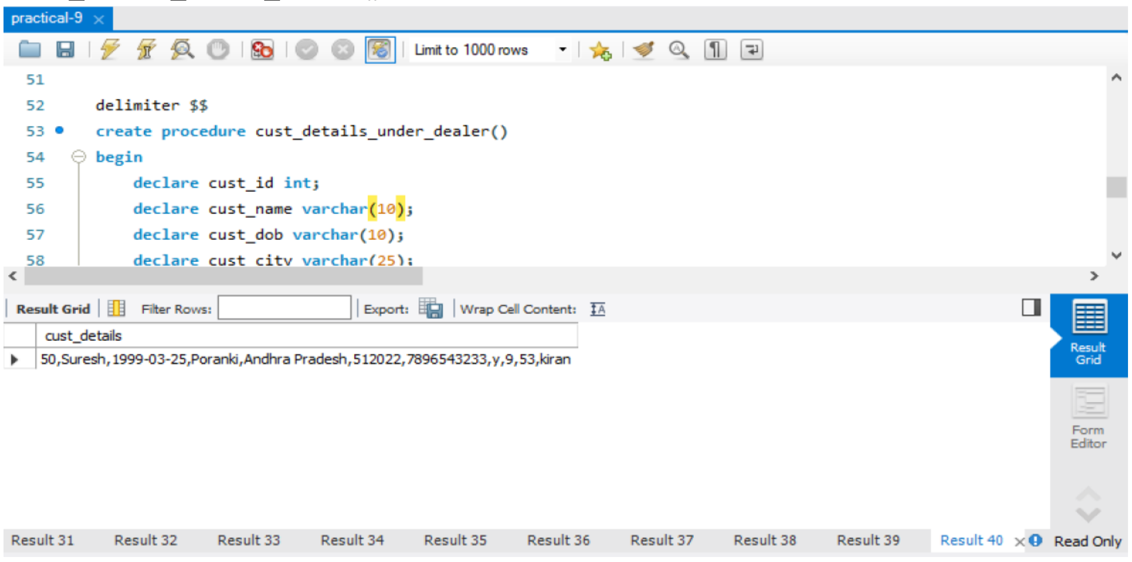
cust_id	cust_name	city	street	state	pincode	ph_no	deal_no	photo_identity	v_id	_dob
47	karthik	Guntur	Chandramouli nagar	Andhra Pradesh	546789	7788776633	20	n	6	2004-01-15

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_iden
tity,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_phn
o,',',photo_identity,',',v_id,',',deal_id,',',deal_name);
        end loop get_customer;
   close c1;
end $$
delimiter ;

call cust_details_under_dealer();
```



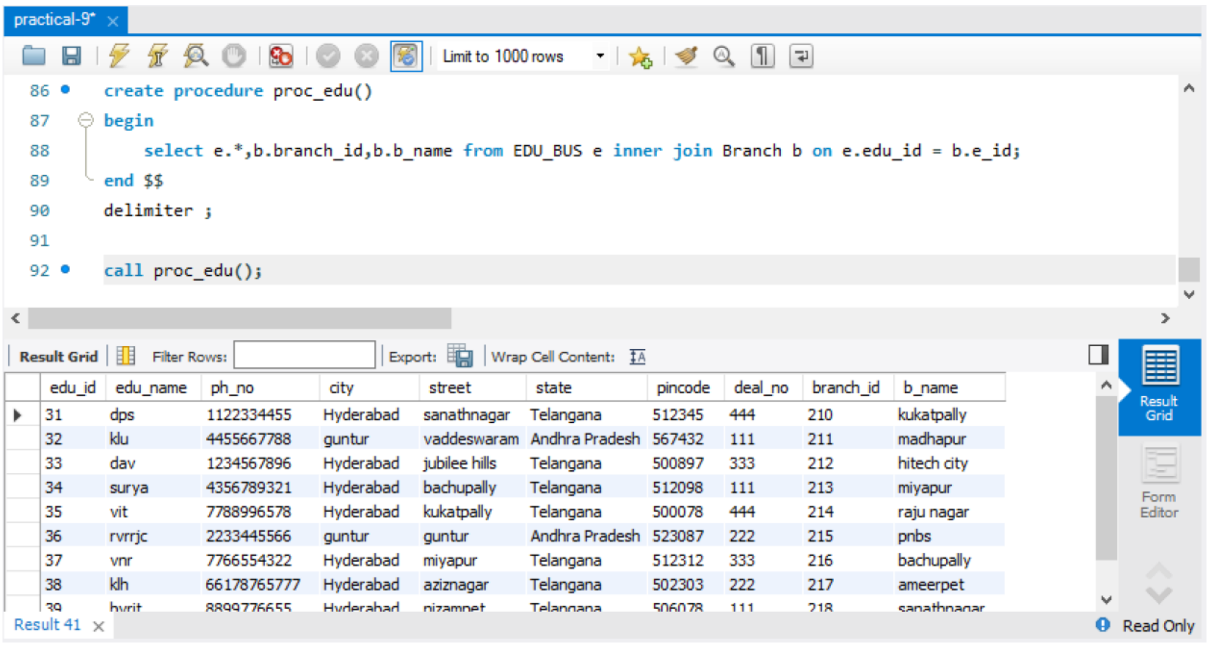
```
delimiter $$
create procedure cust_details_under_dealer()
begin
    declare cust_id int;
    declare cust_name varchar(10);
    declare cust_dob varchar(10);
    declare cust_city varchar(25);
```

```
cust_details
50,Suresh,1999-03-25,Poranki,Andhra Pradesh,512022,7896543233,y,9,53,kiran
```

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();



```
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();
```

edu_id	edu_name	ph_no	city	street	state	pincode	deal_no	branch_id	b_name
31	dps	1122334455	Hyderabad	sanathnagar	Telangana	512345	444	210	kukatpally
32	klu	4455667788	guntur	vaddeswaram	Andhra Pradesh	567432	111	211	madhapur
33	dav	1234567896	Hyderabad	jubilee hills	Telangana	500897	333	212	hitech city
34	surya	4356789321	Hyderabad	bachupally	Telangana	512098	111	213	miyapur
35	vit	7788996578	Hyderabad	kukatpally	Telangana	500078	444	214	raju nagar
36	rvrrjc	2233445566	guntur	guntur	Andhra Pradesh	523087	222	215	pnbs
37	vnr	7766554322	Hyderabad	miyapur	Telangana	512312	333	216	bachupally
38	klh	66178765777	Hyderabad	aziznagar	Telangana	502303	222	217	ameerpet
39	hyrit	8899776655	Hyderabad	nizampet	Telangana	506078	111	218	sanathnagar

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');

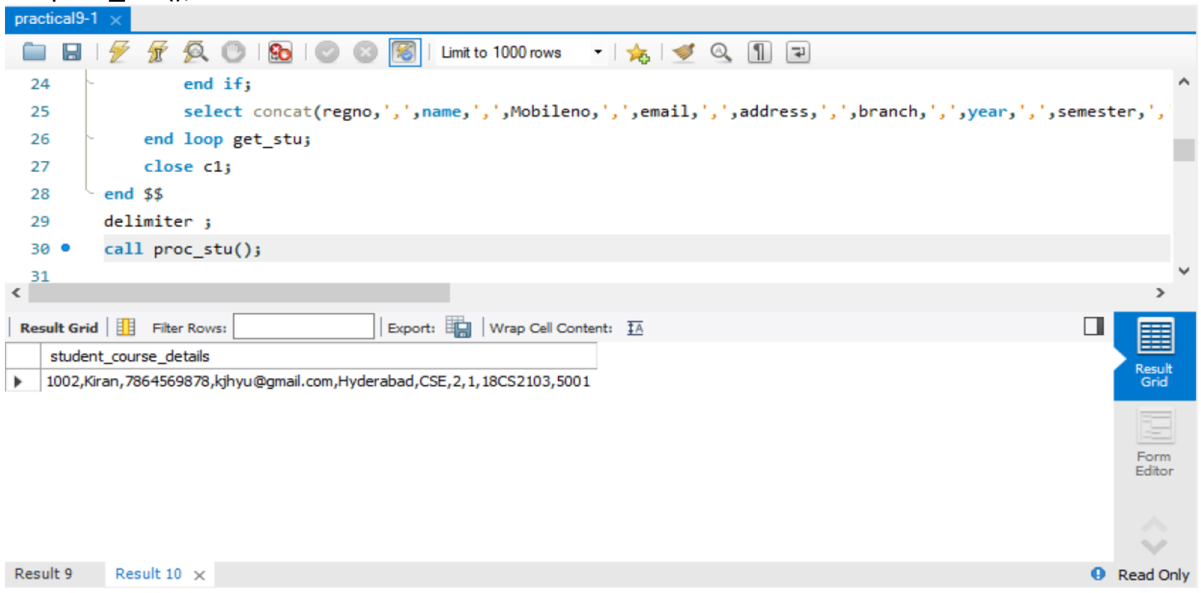
Case Study 4: KL UNIVERSITY ERP

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.COURSEC
ODE,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();
```



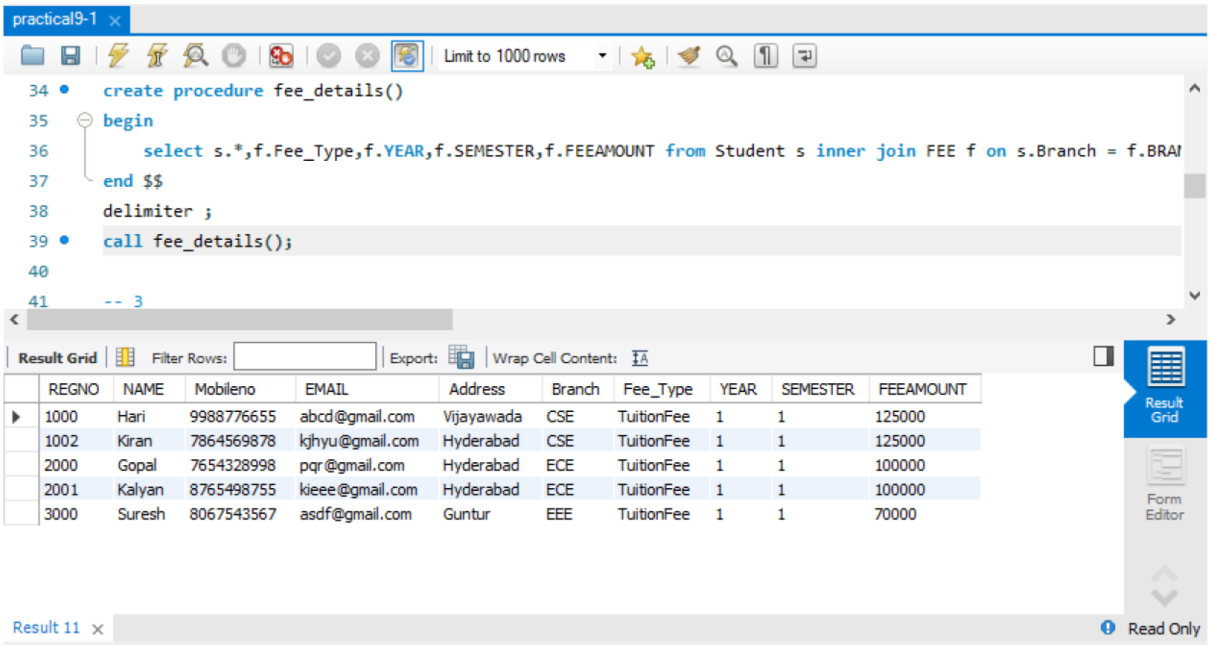
```
            end if;
            select concat(regno,',',name,',',Mobileno,',',email,',',address,',',branch,',',year,',',semester,',
        end loop get_stu;
        close c1;
    end $$
    delimiter ;
    call proc_stu();
```

```
student_course_details
1002,Kiran,7864569878,kjhyu@gmail.com,Hyderabad,CSE,2,1,18CS2103,5001
```

2) 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();
```



```
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.BRAI
end $$
delimiter ;
call fee_details();

-- 3
```

REGNO	NAME	Mobileno	EMAIL	Address	Branch	Fee_Type	YEAR	SEMESTER	FEEAMOUNT
1000	Hari	9988776655	abcd@gmail.com	Vijayawada	CSE	TuitionFee	1	1	125000
1002	Kiran	7864569878	kjhyu@gmail.com	Hyderabad	CSE	TuitionFee	1	1	125000
2000	Gopal	7654328998	pqr@gmail.com	Hyderabad	ECE	TuitionFee	1	1	100000
2001	Kalyan	8765498755	kieee@gmail.com	Hyderabad	ECE	TuitionFee	1	1	100000
3000	Suresh	8067543567	asdf@gmail.com	Guntur	EEE	TuitionFee	1	1	70000

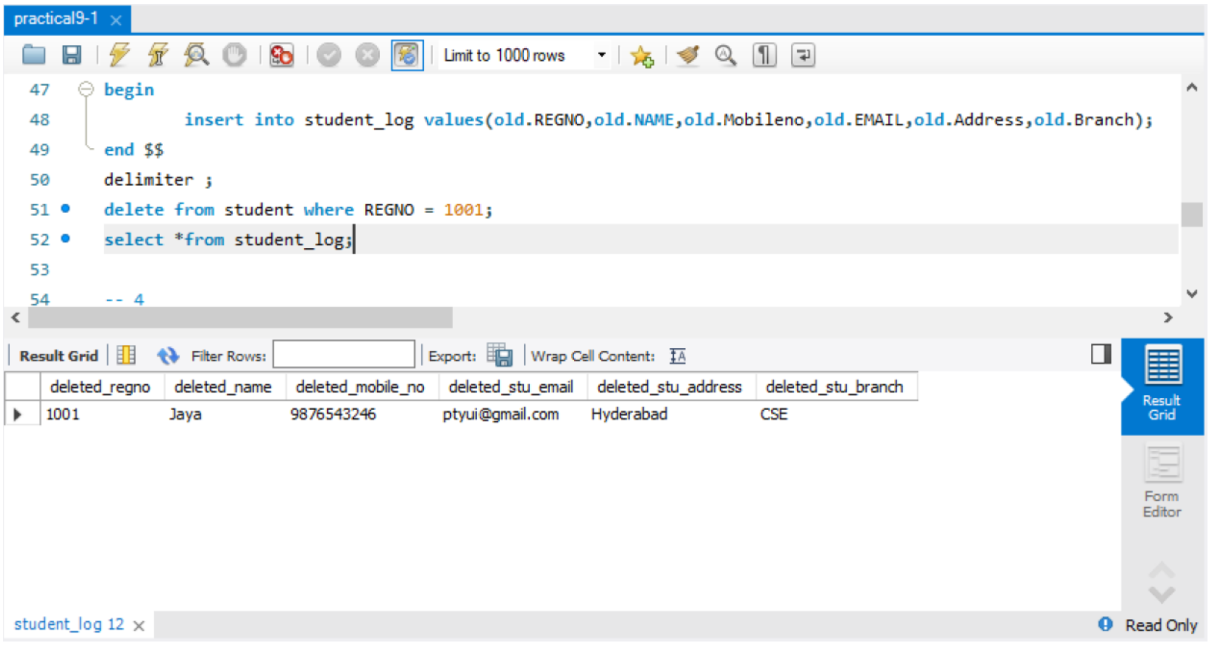
3) Create a trigger that will store the deleted student records in a log file

```
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;
```



```
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;

-- 4
```

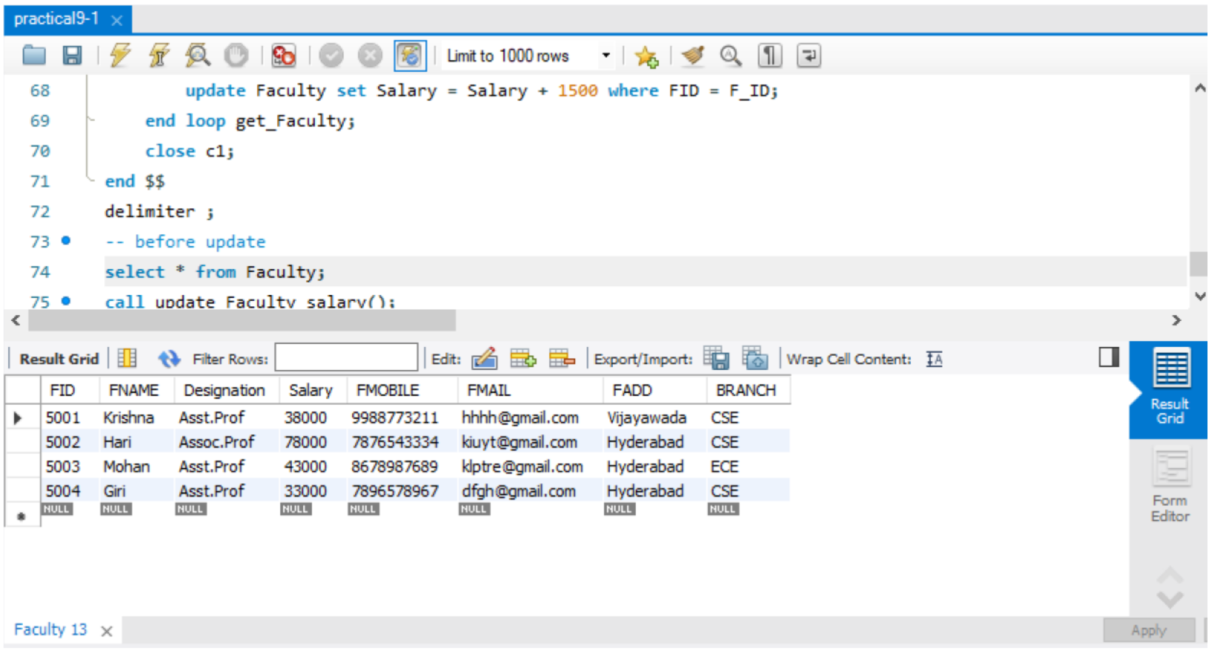
```
deleted_regno  deleted_name  deleted_mobile_no  deleted_stu_email  deleted_stu_address  deleted_stu_branch
1001           Jaya          9876543246         ptyui@gmail.com    Hyderabad            CSE
```

4) 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

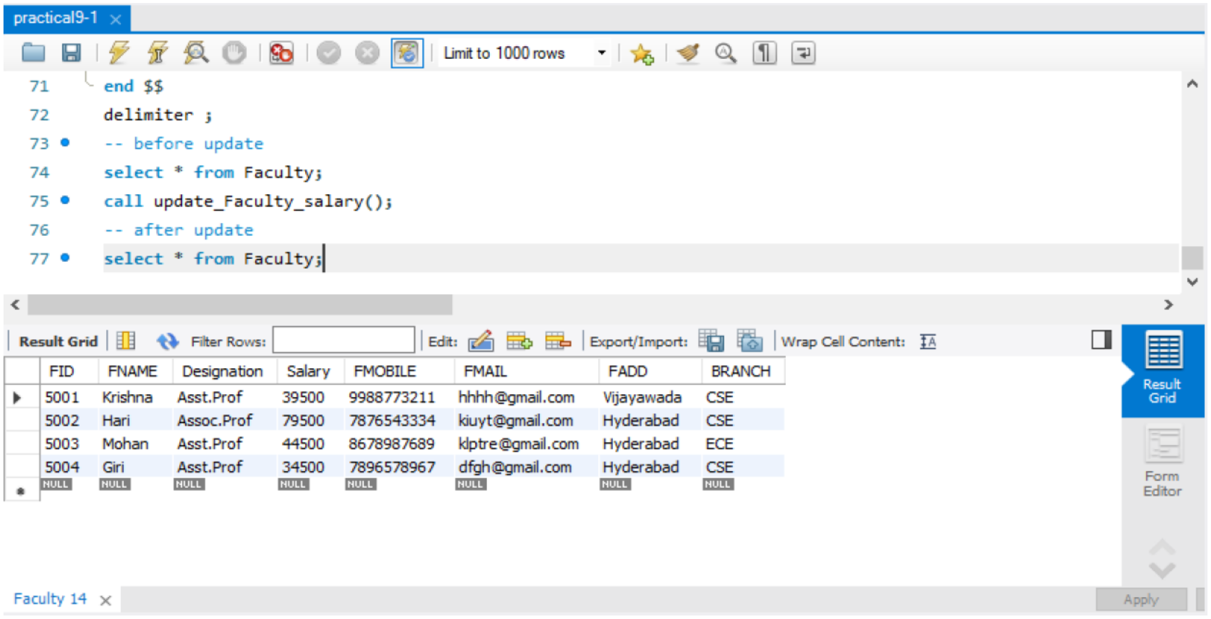


```
        update Faculty set Salary = Salary + 1500 where FID = F_ID;
    end loop get_Faculty;
    close c1;
end $$
delimiter ;
-- before update
select * from Faculty;
call update_Faculty_salary();
```

FID	FNAME	Designation	Salary	FMOBILE	FMAIL	FADD	BRANCH
5001	Krishna	Asst.Prof	38000	9988773211	hhhh@gmail.com	Vijayawada	CSE
5002	Hari	Assoc.Prof	78000	7876543334	kiuyt@gmail.com	Hyderabad	CSE
5003	Mohan	Asst.Prof	43000	8678987689	klptre@gmail.com	Hyderabad	ECE
5004	Giri	Asst.Prof	33000	7896578967	dfgh@gmail.com	Hyderabad	CSE
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Faculty 13

AFTER UPDATE



```
end $$
delimiter ;
-- before update
select * from Faculty;
call update_Faculty_salary();
-- after update
select * from Faculty;
```

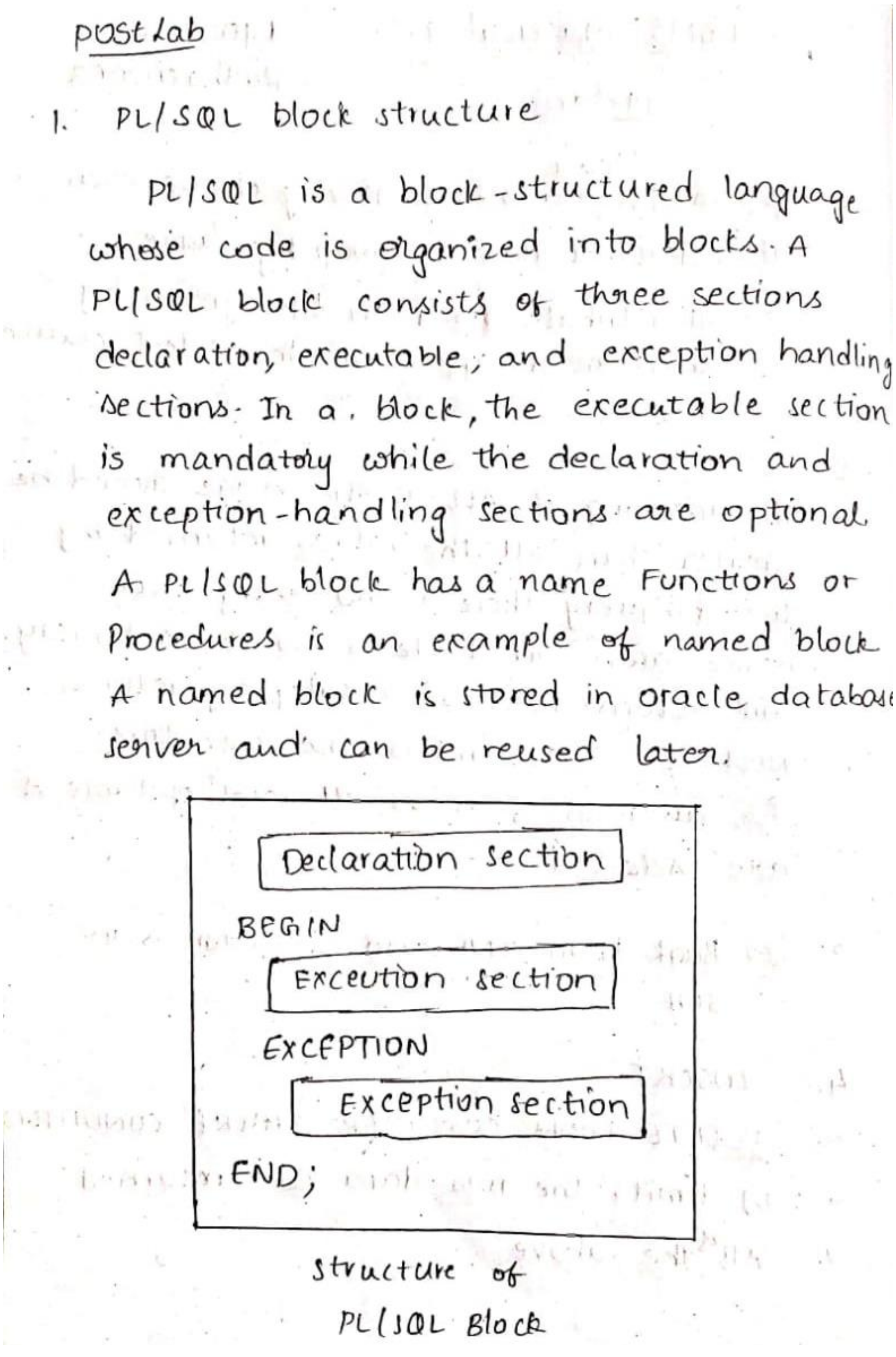
FID	FNAME	Designation	Salary	FMOBILE	FMAIL	FADD	BRANCH
5001	Krishna	Asst.Prof	39500	9988773211	hhhh@gmail.com	Vijayawada	CSE
5002	Hari	Assoc.Prof	79500	7876543334	kiuyt@gmail.com	Hyderabad	CSE
5003	Mohan	Asst.Prof	44500	8678987689	klptre@gmail.com	Hyderabad	ECE
5004	Giri	Asst.Prof	34500	7896578967	dfgh@gmail.com	Hyderabad	CSE
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Faculty 14

190031249 **P.MOHITH**

POSTLAB

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

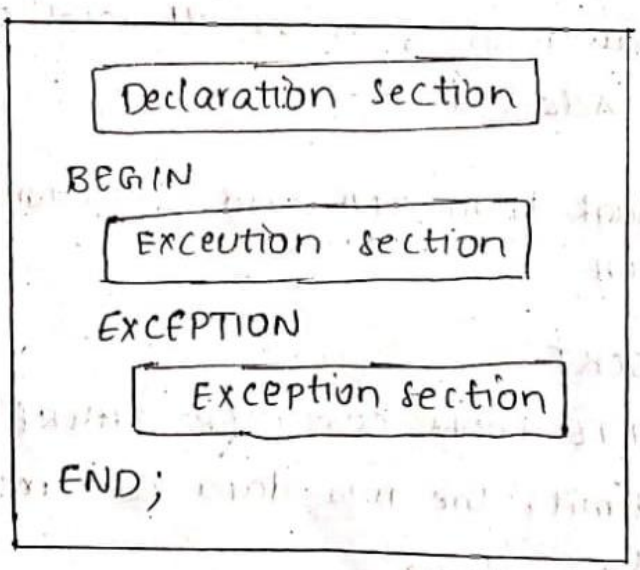


postlab

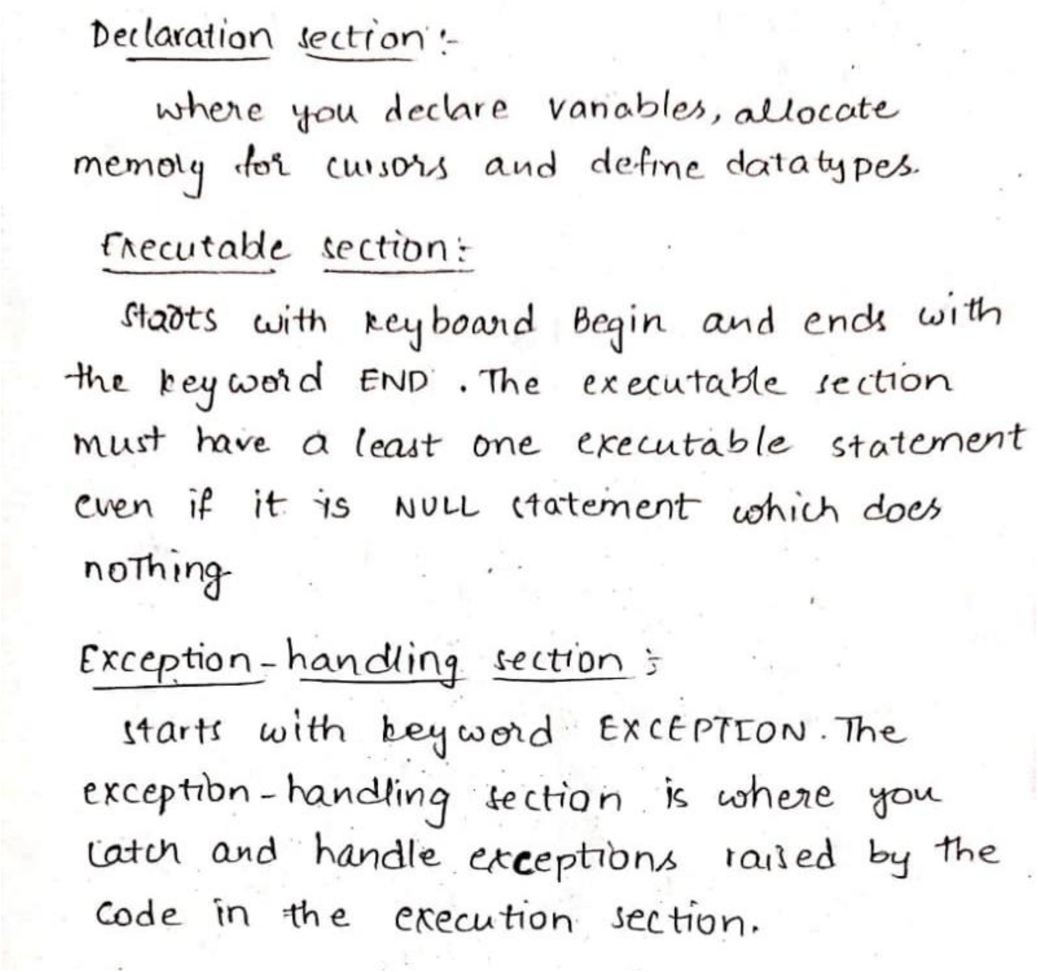
1. PL/SQL block structure

PL/SQL is a block-structured language whose code is organized into blocks. A PL/SQL block consists of three sections declaration, executable, and exception handling sections. In a block, the executable section is mandatory while the declaration and exception-handling sections are optional.

A PL/SQL block has a name Functions or Procedures is an example of named block A named block is stored in oracle database server and can be reused later.



structure of PL/SQL Block



Declaration section:-

where you declare variables, allocate memory for cursors and define datatypes.

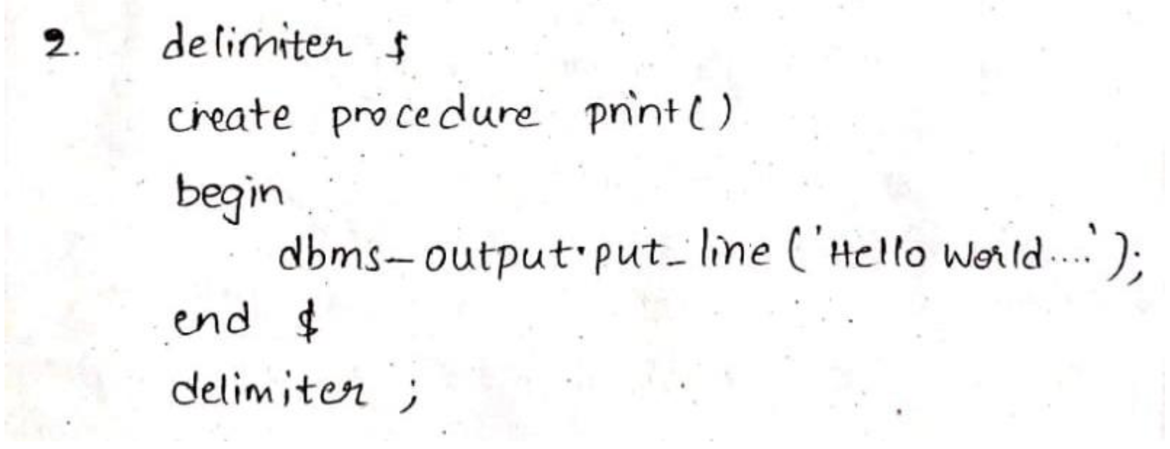
Executable section:

Starts with keyboard Begin and ends with the keyword END. The executable section must have a least one executable statement even if it is NULL statement which does nothing

Exception-handling section:

starts with keyword EXCEPTION. The exception-handling section is where you catch and handle exceptions raised by the code in the execution section.

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



2.
```
delimiter $
create procedure print()
begin
    dbms-output.put_line('Hello World....');
end $
delimiter ;
```

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);
```