**Question-1**

**Q 1. Consider the following three tables.**

**EMPLOYEE( empno, name, deptno,job,hiredate, sal , commission, dob, city, phone)**

**DEPARTMENT( deptno, dname, manager,loc)**

**SALARY( eno, basic,HR,DA,tax).**

**Write equivalent SQL for the following query. (Use foreign key to join the tables.)**

**1) Get the name and city of the employee working for the accounting department?**

**2) Get the name, department name of all the employees whose pay is greater than 10000.**

**3) Get the name of the employee in ascending and descending order.**

**4) Update the city of the employee no.2 from Mumbai to Delhi.**

**5) Get the sum of the basic salary of the employees belongs to Delhi city.**

**6) Get the details of the highest income tax payee.**

**7) Which employee is the senior most?**

**8) Give the details of second highest salary employee (without use of ?<? operator).**

**9) Give the details of second highest salary employee (without use of max and limit operator).**

**10) Give the details of second highest salary employee (with the use of MINUS operator).**

**11) Give the details of all employees of 5th highest salary ( or nth highest salary).**

**12) How many clerks are there in the company?**

**13) Which department has exactly one employee as clerk?**

**14) Which department has the highest number of clerks? Show the deptno and count.**

**15) How many employees are there in each department?**

**16) List the lowest salary for different jobs used in a company and list them in descending order.**

**17) Which department average salary is the lowest among all? Show the deptno,average salary.**

**18) List the minimum, maximum and average salary for each job.**

**19) Compute the difference between maximum and minimum salary.**

**20) List the names of the employees whose name contains LA.**

**21) List the names of the employees whose joining date is between 2nd April,1981 and 8th Sept,1981.**

**22) How many different job titles exist in the employee table?**

**23) Compute the sum of all salaries of employee working under deptno=30.**

**24) For each salesman in the emp table retrieve the deptno and department name.**

**25) List the names of all the employees with their name of the manager.**

**26) List all employees who are working in department located at CHICAGO.**

**27) List all the employees who are working in same department as their managers.**

**28) Retrieve all the employees who are working in deptno=10 and who earn salary atleast as much as any employee working in deptno=30.**

**29) List all the department who have no employees**

**30) Delete the EC department.**

**Create table**

**DEPARMENT:**

**create table department99(**

**deptno int primary key auto\_increment,**

**dname varchar(100),**

**manager varchar(100),**

**loc varchar(100)**

**);**

**EMPLOYEE:**

**create table employee99(**

**empno int primary key auto\_increment,**

**name varchar(100),**

**deptno int,**

**job varchar(100),**

**hiredate date,**

**sal int,**

**commission int,**

**dob date,**

**city varchar(100),**

**phone varchar(100),**

**foreign key(deptno) references department99 (deptno) ON DELETE CASCADE ON UPDATE CASCADE**

**) ;**

**SALARY:**

**create table salary99(**

**empno int,**

**basic int,**

**hr int,**

**da int,**

**tax int,**

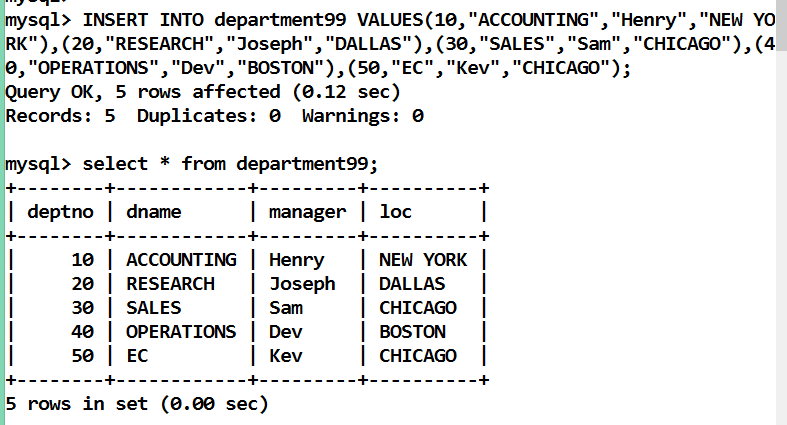
**foreign key(empno) references employee99(empno) ON DELETE CASCADE ON UPDATE CASCADE**

**);**

**Insert into table**

**DEPARTMENT99:**

**INSERT INTO department99 VALUES(10,"ACCOUNTING","Henry","NEW YORK"),(20,"RESEARCH","Joseph","DALLAS"),(30,"SALES","Sam","CHICAGO"),(40,"OPERATIONS","Dev","BOSTON"),(50,"EC","Kev","CHICAGO");**

****

**EMPLOYEE99:**

**insert into employee99 values (NULL, 'Rakesh' , 10,'CLERK' ,'1981/03/15', 12500,100, '1988/05/14', 'Mumbai', '9822121221')**

**,(NULL, 'Rohit' , 10,'CLERK' ,'1981/02/15', 1200,100, '1988/05/14', 'Delhi', '9822121221')**

**,(NULL, 'Milande' , 20,'SALESMAN' ,'1981/99/15', 5000,100, '1988/05/14', 'Chennai', '9822121221')**

**,(NULL, 'Joseph' , 20,'CLERK' ,'1981/06/15', 10000,100, '1988/05/14', 'Haryana', '9822121221')**

**,(NULL, 'Root' , 30,'MANAGER' ,'1981/07/15', 10002,100, '1988/05/14', 'Chandigarh', '9822121221')**

**,(NULL, 'Drake' , 30,'SALESMAN' ,'1981/08/15', 8211,100, '1988/05/14', 'CHICAGO', '9822121221'),**

**(NULL, 'Jane' , 30 ,'CLERK','1981/09/15', 9800,100, '1988/05/14', 'CHICAGO', '9822121221'),**

**(NULL, 'Niki' , 40 ,'ASS.MANAGER','1981/10/15', 4021,100, '1988/05/14', 'NEW YORK', '9822121221')**

**,(NULL, 'James' , 40,'PRO-MANAGER' ,'1981/11/15', 98002,100, '1988/05/14', 'NEW YORK', '9822121221'),**

**(NULL, 'James' , 40,'CLERK' ,'1981/12/20', 9122,100, '1988/05/14', 'DALLAS', '9822121221'),**

**(NULL, 'Anish' , 40,'SALEMAN' ,'1991/12/20', 8123,100, '1988/05/14', 'DALLAS', '9822121221')**

**,(NULL, 'Saleh' , 50,'ACC.HOLDER' ,'1991/02/20', 2034,100, '1988/05/14', 'BOSTON', '9822121221'),**

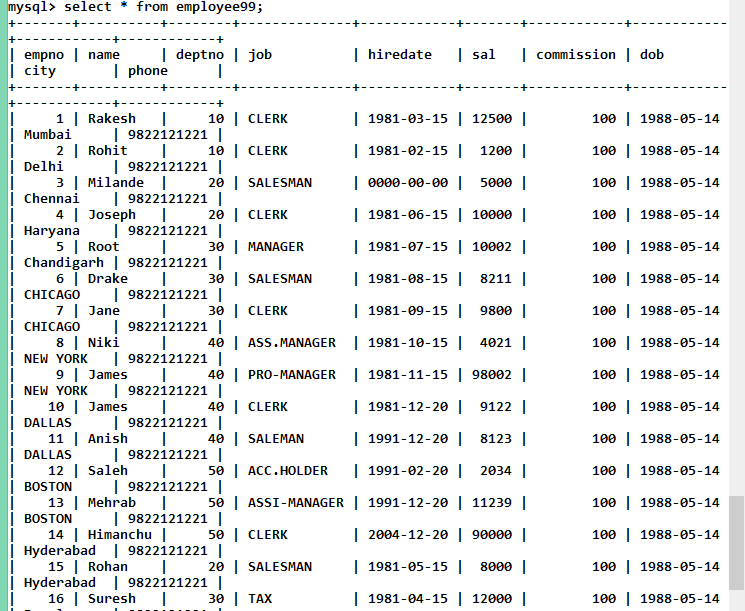
**(NULL, 'Mehrab' , 50 ,'ASSI-MANAGER','1991/12/20', 11239,100, '1988/05/14', 'BOSTON', '9822121221')**

**,(NULL, 'Himanchu' , 50,'CLERK' ,'2004/12/20', 90000,100, '1988/05/14', 'Hyderabad', '9822121221')**

**,(NULL, 'Rohan' , 20,'SALESMAN' ,'1981/05/15', 8000,100, '1988/05/14', 'Hyderabad', '9822121221')**

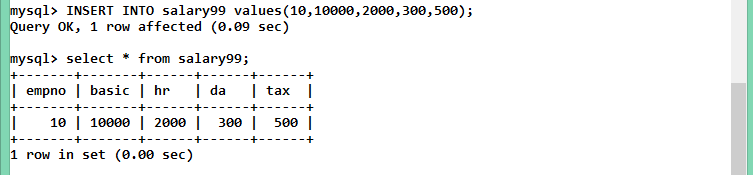
**,(NULL, 'Suresh' , 30,'TAX' ,'1981/04/15', 12000,100, '1988/05/14', 'Banglore', '9822121221') ,**

**(NULL, 'Raja' , 10,'MANAGER' ,'1981/04/15', 19500,100, '1988/08/23', 'Delhi', '9822121781');**

****

**SALARY99:**

**INSERT INTO salary99 values(90,10000,2000,300,500);**

****

**Queries:**

**1) Get the name and city of the employee working for the accounting department?**

* SELECT e.name,e.city FROM ( SELECT deptno FROM department99 WHERE dname="ACCOUNTING" ) d, employee99 e

WHERE d.deptno=e.deptno;

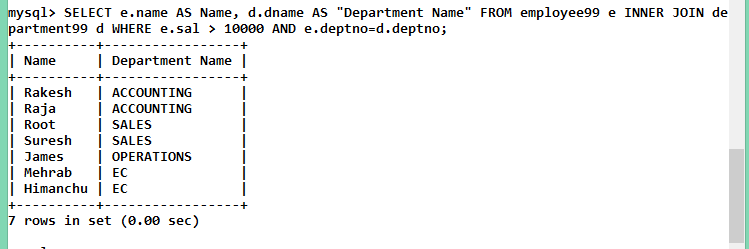


**2) Get the name, department name of all the employees whose pay is greater than 10000.**

* SELECT e.name AS Name, d.dname AS "Department Name"

FROM employee99 e INNER JOIN department99 d

WHERE e.sal > 10000 AND e.deptno=d.deptno;



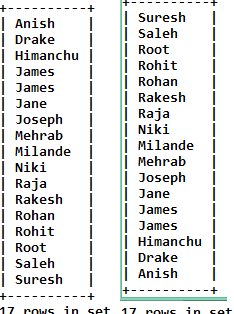
**3) Get the name of the employee in ascending and descending order.**

* (i) SELECT name FROM employee99 ORDER BY name;

(ii) SELECT name

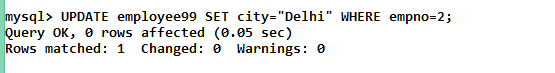
FROM employee99

ORDER BY name DESC;



**4) Update the city of the employee no.2 from Mumbai to Delhi.**

* UPDATE employee99 SET city="Delhi" WHERE empno=2;

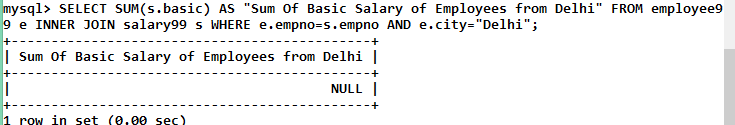


**5) Get the sum of the basic salary of the employees belongs to Delhi city.**

* SELECT SUM(s.basic) AS "Sum Of Basic Salary of Employees from Delhi"

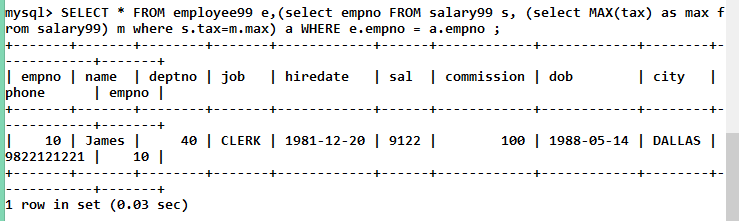
FROM employee99 e INNER JOIN salary99 s

WHERE e.empno=s.empno AND e.city="Delhi";



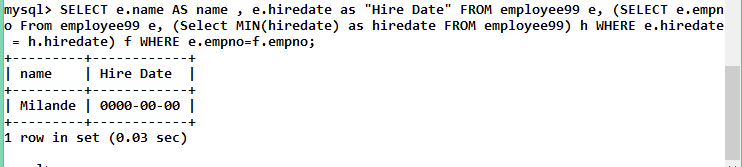
**6) Get the details of the highest income tax payee.**

* SELECT \* FROM employee99 e,(select empno FROM salary99 s, (select MAX(tax) as max from salary99) m where s.tax=m.max) a WHERE e.empno = a.empno ;



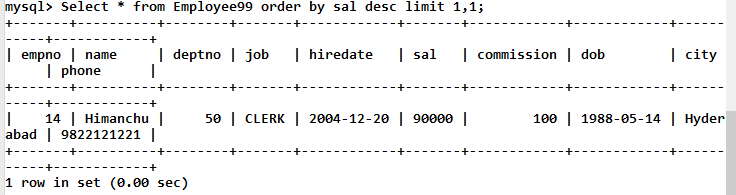
**7) Which employee is the senior most?**

* SELECT e.name AS name , e.hiredate as "Hire Date" FROM employee99 e, (SELECT e.empno From employee99 e, (Select MIN(hiredate) as hiredate FROM employee99) h WHERE e.hiredate = h.hiredate) f WHERE e.empno=f.empno;



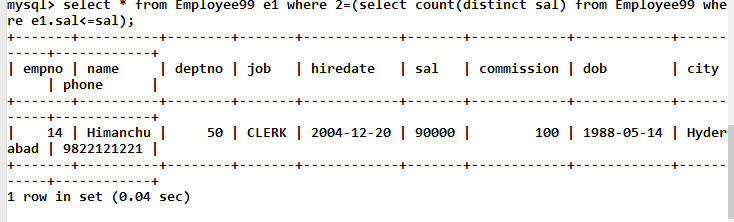
**8) Give the details of second highest salary employee (without use of ?<? operator).**

* Select \* from Employee99 order by sal desc limit 1,1;



**9) Give the details of second highest salary employee (without use of max and limit operator).**

* select \* from Employee99 e1 where 2=(select count(distinct sal) from Employee99 where e1.sal<=sal);

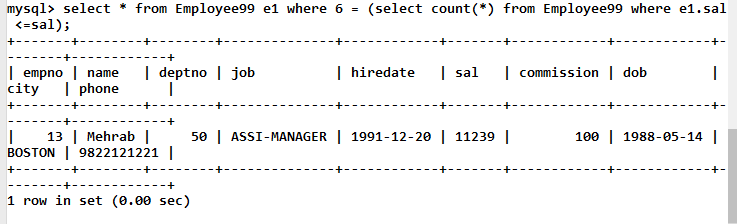


**10) Give the details of second highest salary employee (with the use of MINUS operator).**

* MINUS not support in sql

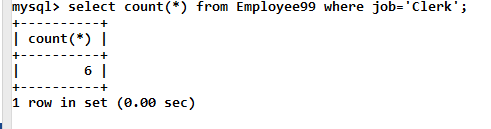
**11) Give the details of all employees of 5th highest salary ( or nth highest salary).**

* select \* from Employee99 e1 where 6 = (select count(\*) from Employee99 where e1.sal <=sal);



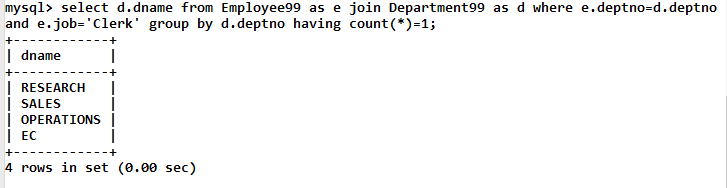
**12) How many clerks are there in the company?**

* select count(\*) from Employee99 where job='Clerk';



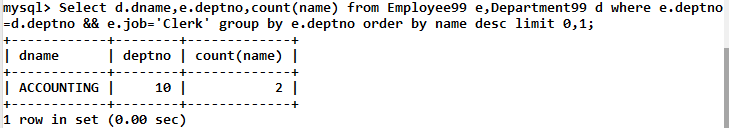
**13) Which department has exactly one employee as clerk?**

* select d.dname from Employee99 as e join Department99 as d where e.deptno=d.deptno and e.job='Clerk' group by d.deptno having count(\*)=1;



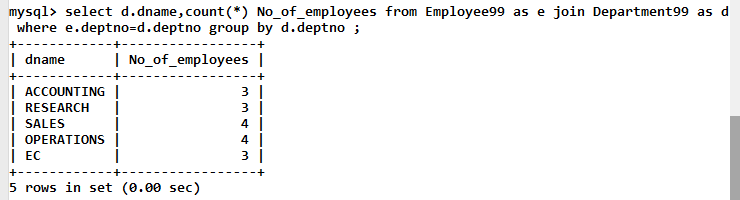
**14) Which department has the highest number of clerks? Show the deptno and count.**

* Select d.dname,e.deptno,count(name) from Employee99 e,Department99 d where e.deptno=d.deptno && e.job='Clerk' group by e.deptno order by name desc limit 0,1;



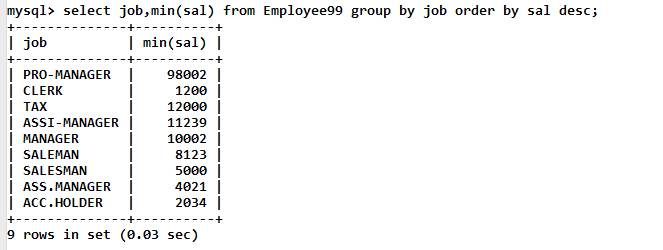
**15) How many employees are there in each department?**

* select d.dname,count(\*) No\_of\_employees from Employee99 as e join Department99 as d where e.deptno=d.deptno group by d.deptno ;



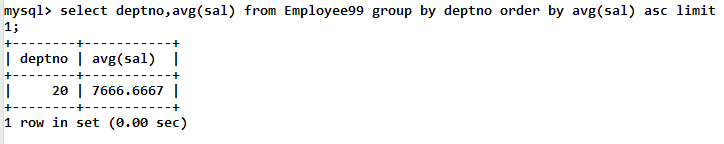
**16) List the lowest salary for different jobs used in a company and list them in descending order.**

* select job,min(sal) from Employee99 group by job order by sal desc;



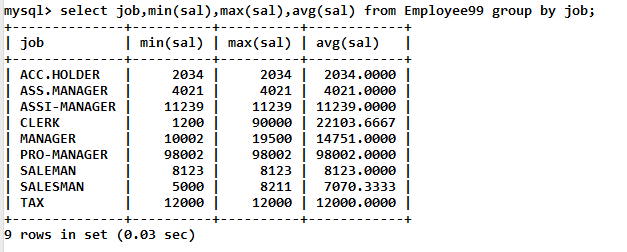
**17) Which department average salary is the lowest among all? Show the deptno,average salary.**

* select deptno,avg(sal) from Employee99 group by deptno order by avg(sal) asc limit 1;



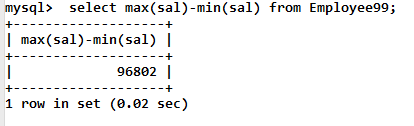
**18) List the minimum, maximum and average salary for each job.**

* select job,min(sal),max(sal),avg(sal) from Employee99 group by job;



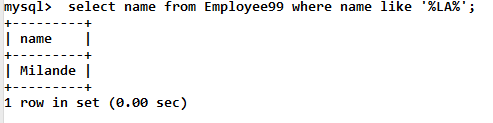
**19) Compute the difference between maximum and minimum salary.**

* select max(sal)-min(sal) from Employee99;



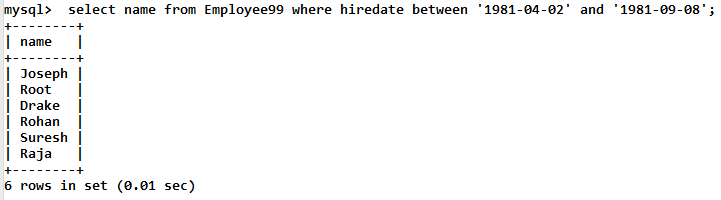
**20) List the names of the employees whose name contains LA.**

* select name from Employee99 where name like '%LA%';



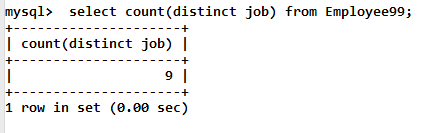
**21) List the names of the employees whose joining date is between 2nd April,1981 and 8th Sept,1981.**

* select name from Employee99 where hiredate between '1981-04-02' and '1981-09-08';



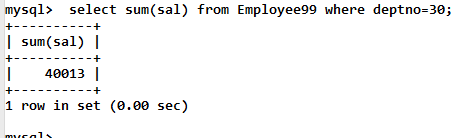
**22) How many different job titles exist in the employee table?**

* select count(distinct job) from Employee99;



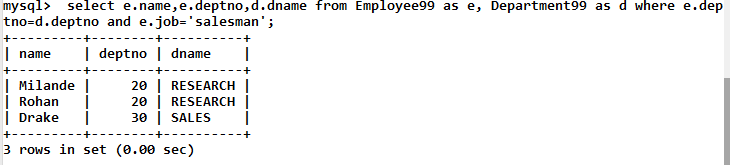
**23) Compute the sum of all salaries of employee working under deptno=30.**

* select sum(sal) from Employee99 where deptno=30;



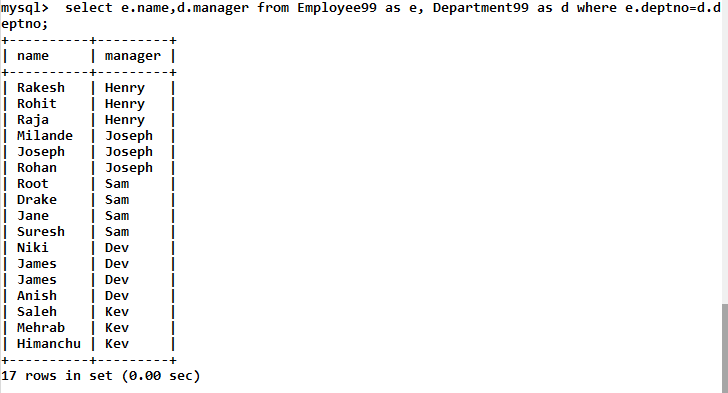
**24) For each salesman in the emp table retrieve the deptno and department name.**

* select e.name,e.deptno,d.dname from Employee99 as e, Department99 as d where e.deptno=d.deptno and e.job='salesman';



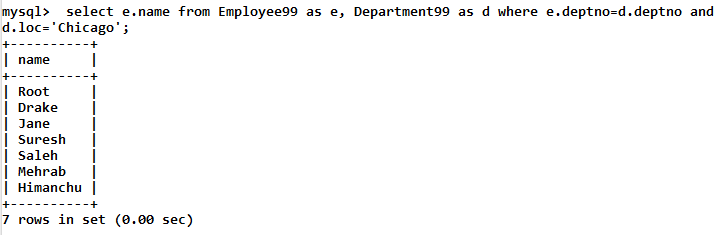
**25) List the names of all the employees with their name of the manager.**

* select e.name,d.manager from Employee99 as e, Department99 as d where e.deptno=d.deptno;



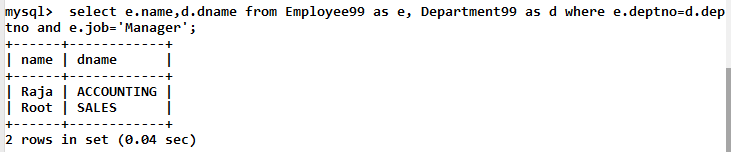
**26) List all employees who are working in department located at CHICAGO.**

* select e.name from Employee99 as e, Department99 as d where e.deptno=d.deptno and d.loc='Chicago';



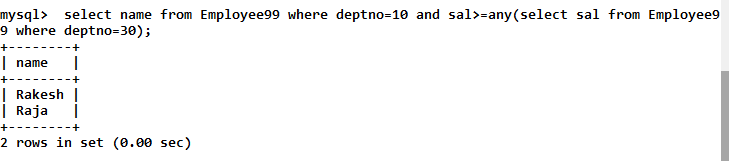
**27) List all the employees who are working in same department as their managers.**

* select e.name,d.dname from Employee99 as e, Department99 as d where e.deptno=d.deptno and e.job='Manager';



**28) Retrieve all the employees who are working in deptno=10 and who earn salary atleast as much as any employee working in deptno=30.**

* select name from Employee99 where deptno=10 and sal>=any(select sal from Employee99 where deptno=30);



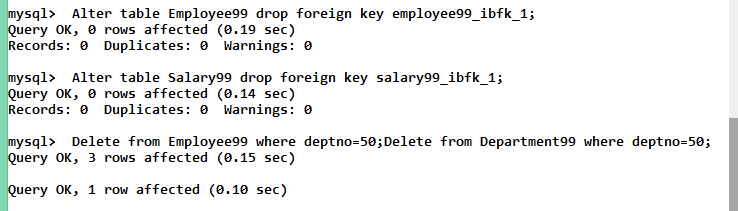
**29) List all the department who have no employees.**

* select deptno, dname from Department99 where deptno not in(select deptno from Employee99);



**30) Delete the EC department.**

* Alter table Employee99 drop foreign key employee99\_ibfk\_1;
* Alter table Salary99 drop foreign key salary99\_ibfk\_1;
* Delete from Employee99 where deptno=50;Delete from Department99 where deptno=50;



**question-2**

**Q 1. Write a function and a stored procedure to print Hello ! How are you?.**

**Store-Procedure:**

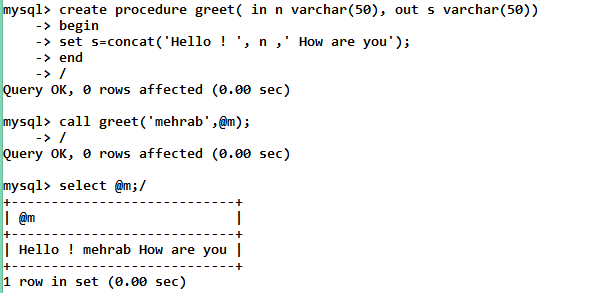
create procedure greet( in n varchar(50), out s varchar(50))

begin

set s=concat('Hello ! ', n ,' How are you');

end

/



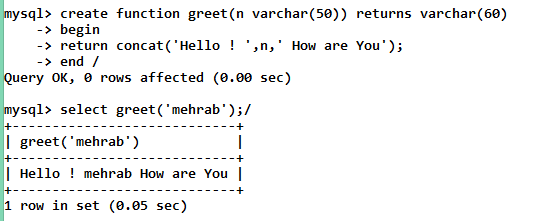
**Function:**

create function greet(n varchar(50)) returns varchar(60)

begin

return concat('Hello ! ',n,' How are You');

end /



**Q 2. Write a function and a stored procedure to count the number of employees in the table employee.**

**Store-Procedure:**

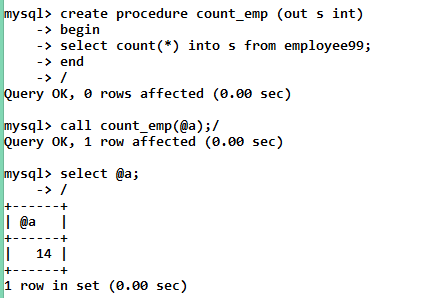
create procedure count\_emp (out s int)

begin

select count(\*) into s from employee99;

end

/



**Function:**

create function count\_emp ()

returns int

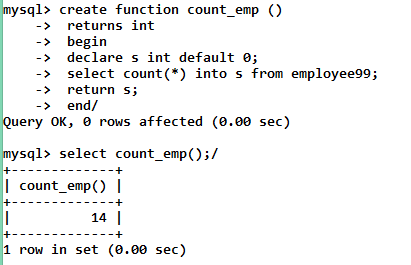
begin

declare s int default 0;

select count(\*) into s from employee99;

return s;

end/



**Q 3. Write a function and a stored procedure to calculate the factorial of the given number.**

**Store-Procedure:**

create procedure fac(in N int)

begin

declare f int default 1;

myloop:loop

if N=0 then

leave myloop;

else

set f=f\*N;

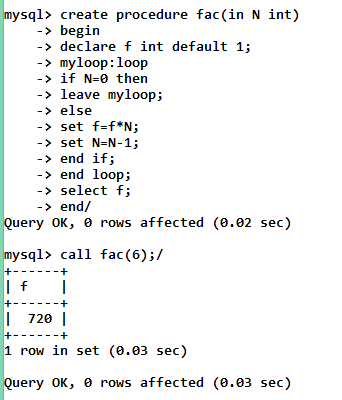
set N=N-1;

end if;

end loop;

select f;

end/



**Function:**

create function fac(N int)

returns int

begin

declare f int default 1;

myloop:loop

if N=0 then

leave myloop;

else

set f=f\*N;

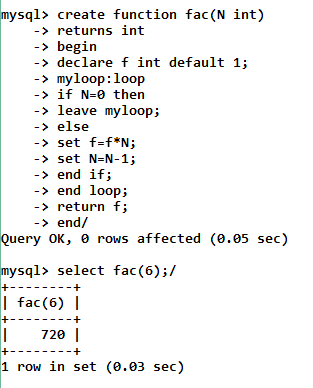
set N=N-1;

end if;

end loop;

return f;

end/



**Q 4. Write a function and a stored procedure to calculate the average of**

**three numbers.**

**Store-Procedure:**

create procedure avg(in a real,in b real,in c real)

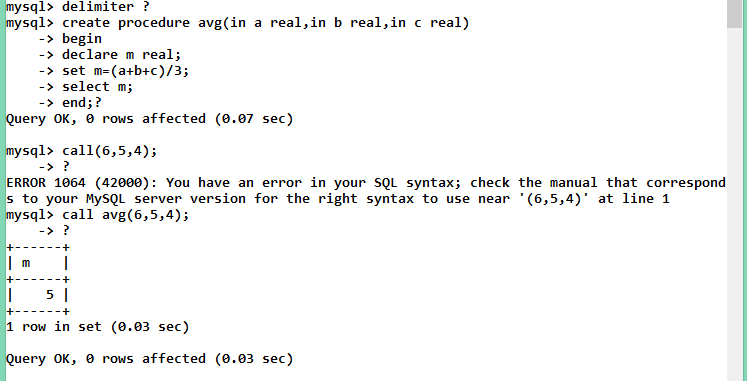
begin

declare m real;

set m=(a+b+c)/3;

select m;

end;



**Function:**

create function ave(a real,b real,c real)

returns real

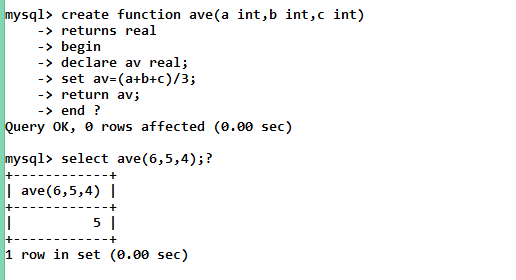
begin

declare av real;

set av=(a+b+c)/3;

return av;

end ?



**Q5. Write a function and stored procedure to find fibonacci series and its sum.**

**Store-procedure:**

create procedure fib(in n int , out m int)

begin

declare fib varchar(100) default ' ';

declare a int default 1;

declare b int default 1;

declare c int default 0;

set m =0;

myloop : loop

if(n <= 1)then

leave myloop;

else

set fib=concat(fib,' ',a,' ');

set m = m+a+b;

set c=b;

set b=b+a;

set a=c;

set n=n-1;

iterate myloop;

end if;

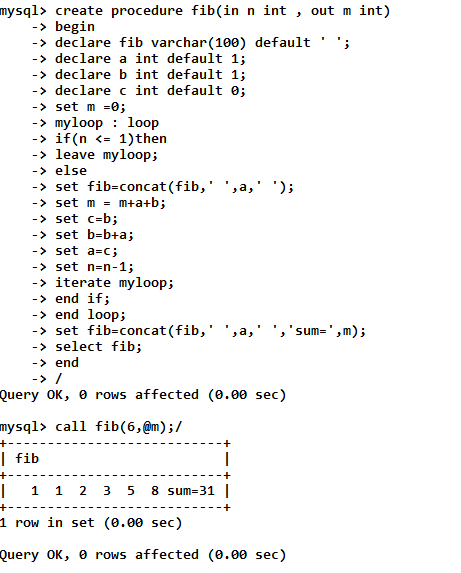
end loop;

set fib=concat(fib,' ',a,' ','sum=',m);

select fib;

end

/



**Function:**

create function fib (n int)

returns varchar(100)

begin

declare fib varchar(100) default ' ';

declare a int default 1;

declare b int default 1;

declare c int default 0;

declare sm int default 0;

myloop : loop

if(n <= 1)then

leave myloop;

else

set fib=concat(fib,' ',a,' ');

set sm = sm+a+b;

set c=b;

set b=b+a;

set a=c;

set n=n-1;

iterate myloop;

end if;

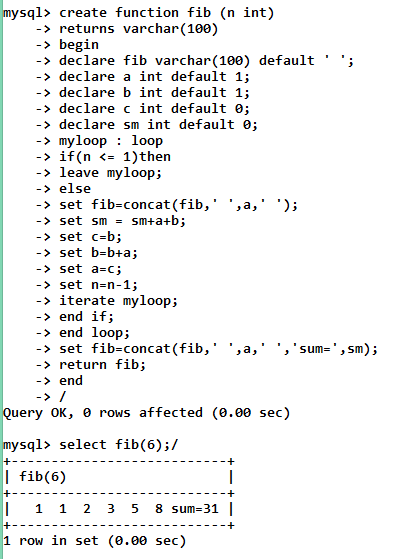
end loop;

set fib=concat(fib,' ',a,' ','sum=',sm);

return fib;

end

/



question-3

Consider the following relations

Student (snum : integer ,sname:string,major :string,level : string,age:integer).,

Class (name: string, meets\_at: time, room: string, fid: integer).

Enrolled (snum: integer, cname:string).Faculty (fid: intger, fname: string, deptid: integer);

Enrolled has on record per student-class pair such that the student is enrolled in the class.

Write the SQL queries. No duplicates should be printed.(use foreign key )

1. Find the names of all Juniors (level = JR) who are enrolled in a class taught by I. Teach.

2. Find the age of the oldest student who is either a History major or enrolled in a course taught by I. Teach.

3. Find the names of all classes that either meet in room BA1080 or have 2 or more students enrolled.

4. Find the names of all students who are enrolled in two classes that meet at the same time.

5. Find the names of faculty members who teach in every room in which some class is taught.

6. Find the names of faculty members for whom the combined enrollment of the courses that they teach is less than five.

7. For each level, print the level and the average age of students for that level.

8. For all levels except JR, print the level and the average age of students for that level.

9. For each faculty member that has taught classes only in room R128, print the faculty members name and the total number of classes she or he has taught.

10. Find the names of students enrolled in the maximum number of classes.

**TABLE-CREATE:**

**1)STUDENT99:** create table student99(

snum int primary key,

sname varchar(30),

major varchar(30),

level varchar(30),

age int

);

**2)CLASS99:** create table class99(

name varchar(20) primary key,

meets\_at time,

room varchar(20),

fid int,

foreign key(fid) references faculty99(fid)

ON UPDATE CASCADE ON DELETE CASCADE

);

**3)ENROLLED99:** create table enrolled99(

snum int,

cname varchar(20),

foreign key(snum) references student99(snum)

ON UPDATE CASCADE ON DELETE CASCADE,

foreign key(cname) references class99(name)

ON UPDATE CASCADE ON DELETE CASCADE

);

**4)FACULTY99:** create table faculty99(

fid int primary key,

fname varchar(30),

deptid int

);

**INSERTING VALUES:**

**1)STUDENT99:**

INSERT INTO STUDENT99 VALUES(199,"anjum","CS","JR",20);

INSERT INTO STUDENT99 VALUES(102,"farhan khan","CS","JR",21);

INSERT INTO STUDENT99 VALUES(103,"Aly","ELEC","JR",22);

INSERT INTO STUDENT99 VALUES(104,"Mehrab","Mech","SR",25);

INSERT INTO STUDENT99 VALUES(105,"ahsan","civ","SR",40);

**2)CLASS99:**

insert into class99 values ("DBMS",'09:00:00', '324',1);

insert into class99 values ("Automata",'09:50:00', '324',2);

insert into class99 values ("signal",'09:50:00', '324',1);

insert into class99 values ("Basic Electronics",'09:00:00', '325',3);

insert into class99 values ("STRESS",'09:50:00', '307',2);

insert into class99 values ("FLUIDS",'09:00:00', '324',2);

insert into class99 values ("DATAstructure",'09:00:00', 'BA1080',1);

insert into class99 values ("Microprocessor",'09:00:00', 'R128',3);

**3)ENROLLED99:**

INSERT INTO enrolled99 VALUES(199,"DBMS");

INSERT INTO enrolled99 VALUES(102,"Automata");

INSERT INTO enrolled99 VALUES(103,"signal");

INSERT INTO enrolled99 VALUES(104,"Basic Electronics");

INSERT INTO enrolled99 VALUES(104,"STRESS");

INSERT INTO enrolled99 VALUES(199,"FLUIDS");

**4)FACULTY99:**

insert into faculty99 values(1,"computer",123);

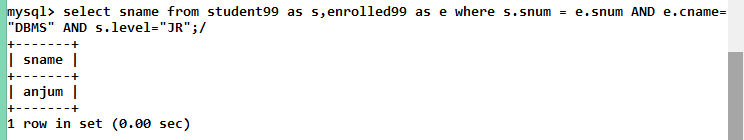
insert into faculty99 values(2,"civil",191);

insert into faculty99 values(3,"electronics",423);

QUERIES

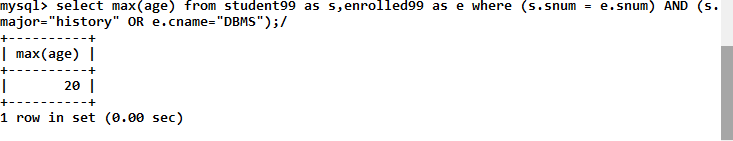
**1.Find the names of all Juniors (level = JR) who are enrolled in a class taught by I. Teach.**

select sname from student99 as s,enrolled99 as e where s.snum = e.snum AND e.cname="DBMS" AND s.level="JR";



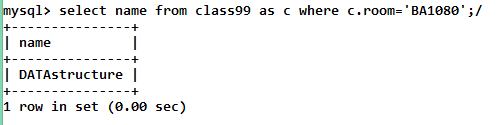
**2.Find the age of the oldest student who is either a History major or enrolled in a course taught by I. Teach.**

select max(age) from student99 as s,enrolled99 as e where (s.snum = e.snum) AND (s.major="history" OR e.cname="DBMS");



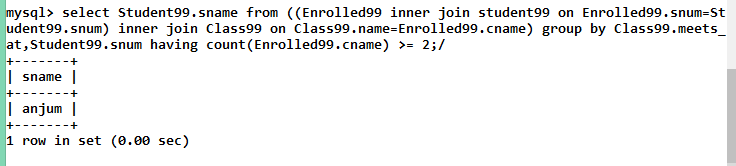
**3.Find the names of all classes that either meet in room BA1080 or have 2 or more students enrolled.**

select name from class99 as c where c.room='BA1080';



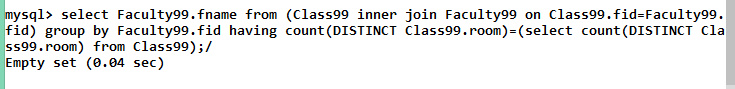
**4.Find the names of all students who are enrolled in two classes that meet at the same time.**

select Student99.sname from ((Enrolled99 inner join student99 on Enrolled99.snum=Student99.snum) inner join Class99 on Class99.name=Enrolled99.cname) group by Class99.meets\_at,Student99.snum having count(Enrolled99.cname) >= 2;



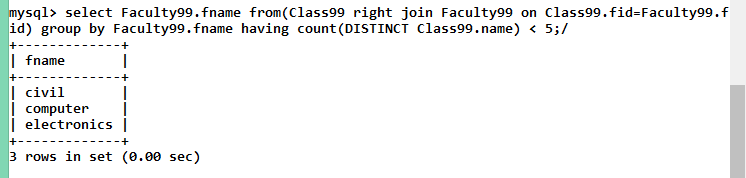
**5.Find the names of faculty members who teach in every room in which some class is taught.**

select Faculty99.fname from (Class99 inner join Faculty99 on Class99.fid=Faculty99.fid) group by Faculty99.fid having count(DISTINCT Class99.room)=(select count(DISTINCT Class99.room) from Class99);



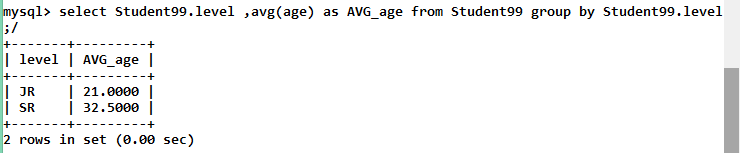
**6.Find the names of faculty members for whom the combined enrollment of the courses that they teach is less than five.**

select Faculty99.fname from(Class99 right join Faculty99 on Class99.fid=Faculty99.fid) group by Faculty99.fname having count(DISTINCT Class99.name) < 5;



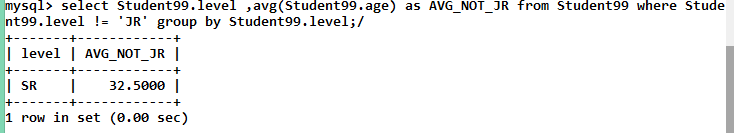
**7.For each level, print the level and the average age of students for that level.**

select Student99.level ,avg(age) as AVG\_age from Student99 group by Student99.level;



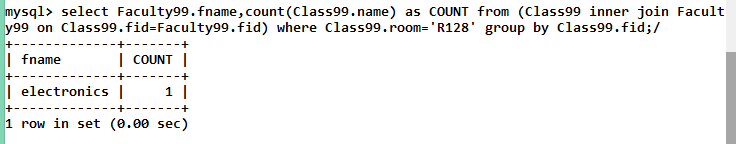
**8.For all levels except JR, print the level and the average age of students for that level.**

select Student99.level ,avg(Student99.age) as AVG\_NOT\_JR from Student99 where Student99.level != 'JR' group by Student99.level;



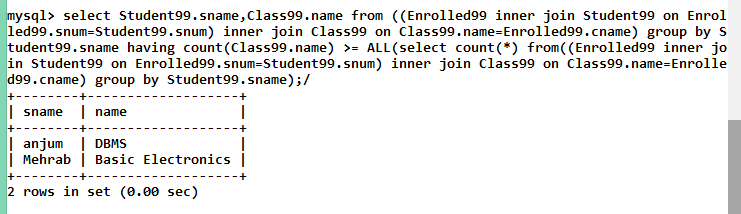
**9.For each faculty member that has taught classes only in room R128, print the faculty members name and the total number of classes she or he has taught.**

select Faculty99.fname,count(Class99.name) as COUNT from (Class99 inner join Faculty99 on Class99.fid=Faculty99.fid) where Class99.room='R128' group by Class99.fid;



**10.Find the names of students enrolled in the maximum number of classes.**

select Student99.sname,Class99.name from ((Enrolled99 inner join Student99 on Enrolled99.snum=Student99.snum) inner join Class99 on Class99.name=Enrolled99.cname) group by Student99.sname having count(Class99.name) >= ALL(select count(\*) from((Enrolled99 inner join Student99 on Enrolled99.snum=Student99.snum) inner join Class99 on Class99.name=Enrolled99.cname) group by Student99.sname);



QUESTION-4

Write equivalent SQL for the following query.

1. Get the title,author name,publisher name for author whose city contain total no of a=2?

2. Give the details of the book which is written by at least two authors.

3. Write a stored procedure (SP Name : insertIntoAuth) to insert the Author information.

4. Write a stored procedure (SP Name : insertBookInfo) to insert the book information such as bookid, title, no. of pages, copyright, authorId, Publisher Name. (Use two stored procedure and call it from one stored procedure i.e nested SP). (SP Name : insertBook, insertWBy).

5. Write a stored procedure to delete the Author information using its AuthID. (Notr: If Author book(in Book Table) exists for AuthID, then it should display message as You cant delete Author because total no book exist in BookTable. First delete all the books written by him).

6. Write a stored procedure to delete the Book using AuthID.

NOTE: Book information should be deleted from both Book and Book\_writtenBy\_Author table.

Book ID Already exist when we insert Book with same id.

Author of given ID does not exist when we enter wrong AuthID in insertBookInfo, insertBook stored procedure.

Author ID already exist when we insert duplicate AuthID in insertIntoAuth stored procedure.

Age Should be greater than 18 and less than 60′ if age is invalid (Age data type should be DATE ). Use function to validate the age in stored procedure. Function Name : AgeValidate .

**create table author99(**

**authid int(10) primary key,**

**authfirst varchar(30) NOT NULL,**

**authlast varchar(30) NOT NULL,**

**authmid varchar(30),**

**authcity varchar(30),**

**age int**

**);**

**create table book99(**

**bookid int(10) primary key NOT NULL,**

**copyright varchar(30),**

**title varchar(30),**

**pages int**

**);**

**create table written\_by99(**

**bookid int(10),**

**authid int(10),**

**foreign key(bookid) references book99(bookid)**

**ON UPDATE CASCADE ON DELETE CASCADE,**

**foreign key(authid) references author99(authid)**

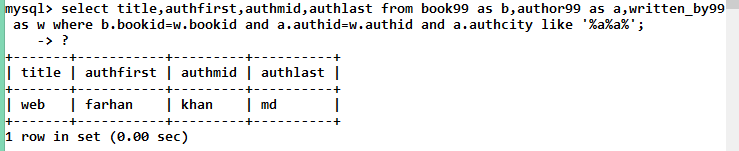
**ON UPDATE CASCADE ON DELETE CASCADE**

**);**

**Queries:**

1. **Get the title,author name,publisher name for author whose city contain total no of a=2?**

select title,authfirst,authmid,authlast from book99 as b,author99 as a,written\_by99 as w where b.bookid=w.bookid and a.authid=w.authid and a.authcity like '%a%a%';



**2.Give the details of the book which is written by at least two authors.**

select \* from book99 where bookid in(select bookid from written\_by99

as w,author99 as a where a.authid=w.authid group by w.bookid having count(\*)>=2

);

**3.Write a stored procedure (SP Name : insertIntoAuth) to insert the Author information.**

create procedure insertIntoAuth(in authid varchar(25),in authfirst varchar(25),in authmiddle varchar(25),in authlast varchar(25),in age date,in authcity varchar(25))

begin

declare x int;

declare m varchar(45);

select AgeVAlidate(age) into x;

set m=concat("age not valid");

if x=1

then

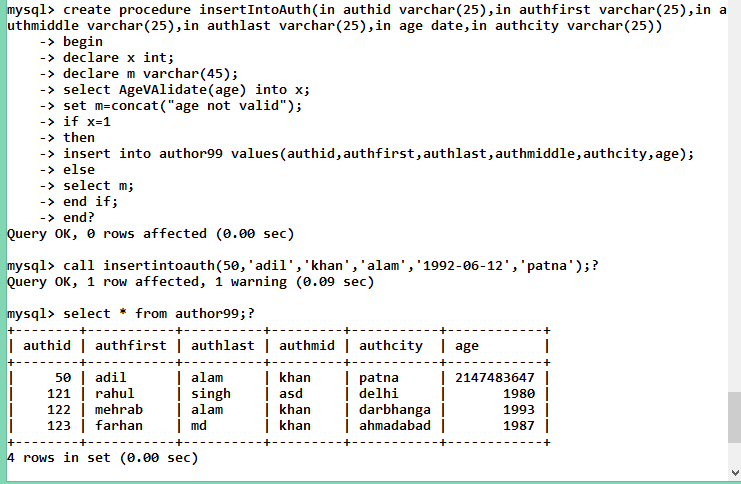
insert into author99 values(authid,authfirst,authlast,authmiddle,authcity,age);

else

select m;

end if;

end?



**4.Write a stored procedure (SP Name : insertBookInfo) to insert the book information such as bookid, title, no. of pages, copyright, authorId, Publisher Name. (Use two stored procedure and call it from one stored procedure i.e nested SP). (SP Name : insertBook, insertWBy).**

create procedure insertBook(in bookid varchar(45),in title varchar(45),in no\_pages int,in copyright varchar(25))

begin

insert into Book99 values(bookid,title,no\_pages,copyright);

end?

create procedure insertWBy(in bookid varchar(45),in publisher\_name varchar(45),in authid varchar(25))

begin

insert into writtenBy99 values(bookid,Publisher\_Name,authid);

end ?

create procedure insertBookinfo(in bid varchar(45),in title varchar(45),in no\_pages int,in copyright varchar(25),in publisher\_name varchar(45),in aid varchar(25))

begin

declare i int;

declare j int;

declare m varchar(45);

declare n varchar(45);

set m=concat("book id already exist");

set n=concat("author id must exist");

select count(\*) INTO I from Book99 as b where b.bookid=bid;

select count(\*) from author99 as a where a.authid=aid into j;

if i=0

then

if j>0

then

call insertbook(bid,title,no\_pages,copyright);

call insertWBy(bid,publisher\_name,aid);

end if;

else

if i>0

then

select m;

end if;

if j=0

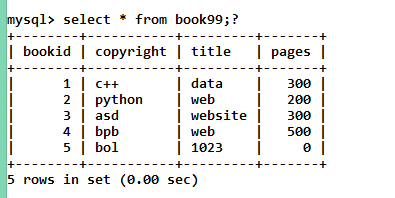
then

select n;

end if;

end if;

end ?



**5.Write a stored procedure to delete the Author information using its AuthID. (Notr: If Author book(in Book Table) exists for AuthID, then it should display message as You cant delete Author because total no book exist in BookTable. First delete all the books written by him).**

delimiter /

create procedure del\_auth(in aid int)

begin

Declare cnt int default 0;

Declare at int default 0;

set at=aid;

select count(\*) into cnt from written\_by99 where authid=aid;

if cnt=0 then

delete from Author99 where authid=aid;

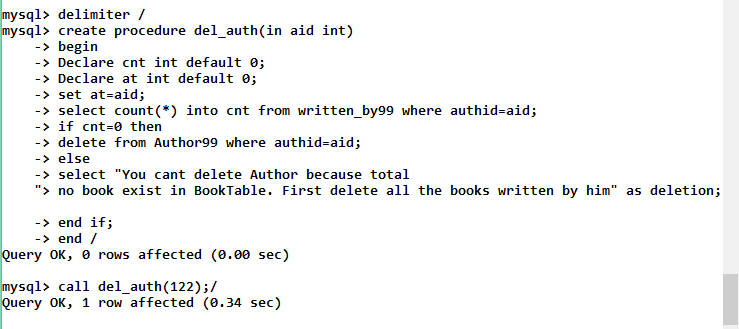
else

select "You cant delete Author because total

no book exist in BookTable. First delete all the books written by him" as deletion;

end if;

end /



QUESTION-5

Create function that validate the age of employee. Function accept the dob of employee and return 1 if age is lies between 18 and 60 else return 0

create function AgeValidate(age date)

returns int

begin

declare set\_age int;

declare message int;

set set\_age=datediff(curdate(),age)/365;

if set\_age>18 and set\_age<60

then

set message=1;

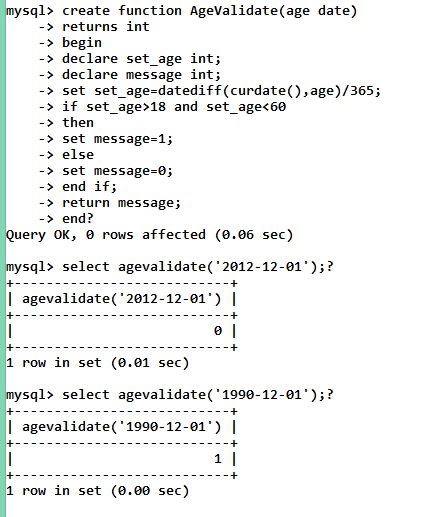
else

set message=0;

end if;

return message;

end?



QUESTION-6

Consider a following table of a database :Book9999(bid, bname,authrname)

Create triggers which create a log of every Insert ,Delete and Update operation on the book table record.

It should also hold the username who was operating at that time and time and type of operation.

NOTE: log table attributes are user, operation, pbid, pauthname, nbid, nbname, nauthname and timeofop

create trigger mytrigger after insert on Book99 for each row

begin

insert into log10 value ('root','insert','','','',new.bid,new.bname,new.authrname,now());

end?

create trigger mytrigger1 after update on Book99 for each row

begin

insert into log99 value ('root','update',old.bid,old.bname,old.authrname,new.bid,new.bname,new.authrname,now());

end ?

create trigger mytrigger3 after delete on Book99 for each row

begin

insert into log99 value ('root','delete',old.bid,old.bname,old.authrname,'','','',now());

end ?

Question-7

Q..) UNDO OPERATION ON LOG99 table By Maintaining BOOK99 table :

Let operation on LOG99 be:

Corresponding BOOK99 table be:

NOW CURSOR:

CREATE PROCEDURE tundo(n int)

BEGIN

DECLARE done INT DEFAULT 1;

DECLARE a VARCHAR(30);

DECLARE b VARCHAR(30);

DECLARE c VARCHAR(30);

DECLARE d VARCHAR(30);

DECLARE e VARCHAR(30);

DECLARE f VARCHAR(30);

DECLARE g VARCHAR(30);

DECLARE h VARCHAR(30);

DECLARE i DATE;

DECLARE cur1 CURSOR FOR SELECT \* FROM log99 ORDER BY timeofop DESC;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 0;

OPEN cur1;

label: LOOP

FETCH cur1 INTO a,b,c,d,e,f,g,h,i;

IF (b="INSERT") THEN

DELETE FROM log99 WHERE nbid=f;

DELETE FROM book99 WHERE bid=f;

DELETE FROM log99 WHERE pbid=f;

END IF;

IF (b="UPDATE") THEN

DELETE FROM log99 WHERE nbid=f;

UPDATE book99 set bid=c, bname=d, authrname=e where nbid=f;

DELETE FROM log99 WHERE pbid=f;

END IF;

IF (b="DELETE") THEN

DELETE FROM log99 WHERE pbid=c;

INSERT INTO book99 VALUES(c,d,e);

DELETE FROM log99 WHERE nbid=c;

END IF;

set n = n - 1;

IF n>0 THEN ITERATE label; END IF;

LEAVE label;

END LOOP label;

CLOSE cur1;

END ?