Program 1:-

show databases;

create database Insurance;

use Insurance;

create table PERSON(driver\_id varchar(30) primary key, name varchar(30), address varchar(30));

create table CAR(Regno varchar(30) primary key, model varchar(30), year int);

create table ACCIDENT(report\_number int primary key, adate date, location varchar(30));

create table OWNS(driver\_id varchar(30), Regno varchar(30), primary key(driver\_id,Regno), foreign key(driver\_id) references PERSON(driver\_id), foreign key(Regno) references CAR(Regno));

create table PARTICIPATED(driver\_id varchar(30), Regno varchar(30), report\_number int, damage\_amount int, primary key(driver\_id, Regno), foreign key(driver\_id, Regno) references OWNS(driver\_id, Regno));

show tables;

insert into PERSON values('08K','james','bangalore');

insert into CAR values('6F','santro','2010');

insert into ACCIDENT values('13','2007/1/21','bangalore');

insert into OWNS values('08K','6F');

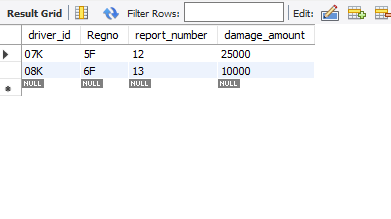
insert into PARTICIPATED values('08K','6F','13','10000');

Query 1:- update damage amount to 25000 for report number 12 and specific regno

update PARTICIPATED set damage\_amount='25000' where Regno='5F' AND report\_number='12';

select \* from PARTICIPATED;

Output:-

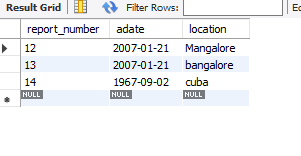


Query 2:- Add a new accident to the database.

insert into ACCIDENT values('14','1967/09/02','cuba');

select \* from ACCIDENT;

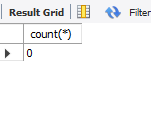
Output:-



Query 3:- Find the total number of people who owned cars that involved in accidents in 2008.

select count(\*) from ACCIDENT where adate>'2007/12/31' AND adate<'2009/01/01';

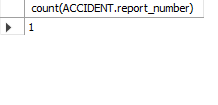
Output:-



Query4:- Find the number of accidents in which cars belonging to a specific model were involved.

select count(ACCIDENT.report\_number) from ACCIDENT,PARTICIPATED,CAR where ACCIDENT.report\_number=PARTICIPATED.report\_number AND PARTICIPATED.Regno=CAR.Regno AND CAR.model='santro';

Output:-



Program 3:-

create database Supplier;

use Supplier;

create table SUPPLIERS(sid int(5) primary key, sname varchar(20), city varchar(20));

create table PARTS(pid int(5) primary key, pname varchar(20), color varchar(10));

create table CATALOG(sid int(5), pid int(5), foreign key(sid) references SUPPLIERS(sid), foreign key(pid) references PARTS(pid), cost float(6), primary key(sid, pid));

show tables;

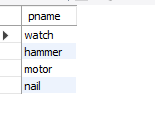
Query 1:- Find the pnames of parts for which there is some supplier.

SELECT DISTINCT P.pname

FROM PARTS P, CATALOG C

WHERE P.pid = C.pid;

Output:-



Query 7:- Find the sids of suppliers who supply only red parts.

SELECT s.sid, S.sname

FROM Parts P, SUPPLIERS S, CATALOG C

WHERE C.pid = P.pid

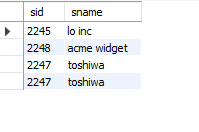
AND C.sid = S.sid

AND C.cost = (SELECT MAX(C1.cost)

FROM CATALOG C1

WHERE C1.pid = P.pid);

Output:-



Query 4:- Find the pnames of parts supplied by ABIBAS Suppliers and by no one else

SELECT P.pname

FROM PARTS P, CATALOG C, SUPPLIERS S

WHERE P.pid = C.pid AND C.sid = S.sid

AND S.sname = 'acme widget'

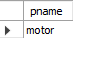
AND NOT EXISTS ( SELECT \*

FROM CATALOG C1, SUPPLIERS S1

WHERE P.pid = C1.pid AND C1.sid = S1.sid AND

S1.sname <> 'acme widget' );

Output:-



Query 5:- Find the sids of suppliers who charge more for some part than the average cost of that part (averaged over all the suppliers who supply that part).

SELECT DISTINCT C.sid

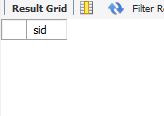
FROM CATALOG C

WHERE C.cost > ( SELECT AVG (C1.cost)

FROM CATALOG C1

WHERE C1.pid = C.pid );

Output:-



Program 4:-

create database airline;

use airline;

create table flights(flno integer not null,

coming varchar(20) not null,

going varchar(20) not null,

distance int not null,

departs time not null,

arrives time not null,

price int not null,

primary key(flno));

create table aircraft( aid int not null,

aname varchar(20) not null,

cruisingrange int not null,

primary key(aid));

create table employee( eid int not null,

ename varchar(20) not null,

salary int not null,

primary key(eid));

create table certified(

eid int not null,

aid int not null,

foreign key(eid) REFERENCES employee(eid) on delete cascade on update cascade,

foreign key(aid) references aircraft(aid) on delete cascade on update cascade);

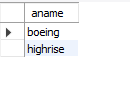
show tables;

Query 1:- Find the names of aircraft such that all pilots certified to operate them have salaries more than Rs.80,000.

select distinct aname from aircraft where aid in (select aid from certified where eid in (select eid from employee

where salary > 80000));

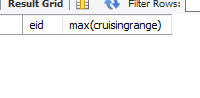
Output:-



Query 2:- For each pilot who is certified for more than three aircrafts, find the eid and the maximum cruising range of the aircraft for which she or he is certified.

select c.eid, max(cruisingrange) from certified c, aircraft a where c.aid = a.aid group by c.eid having count(\*) > 3;

Output:-

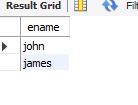


Query 5:- Find the names of pilots certified for some Boeing aircraft.

select e.ename from employee e, certified c, aircraft a where a.aname like '%Boeing%' and a.aid = c.aid

and c.eid = e.eid

Output:-

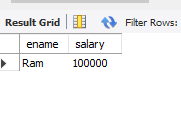


Query 7:- Print the name and salary of every non-pilot whose salary is more than the average salary for pilots.

select e1.ename, e1.salary from employee e1 where e1.salary > (select avg(e.salary) from employee e where e.eid in

(select eid from certified)) and not exists(select \* from certified c where c.eid = e1.eid)

Output:-



Program 5:- create database Student\_Faculty;

use Student\_Faculty;

CREATE TABLE STUDENT(snum INT, sname VARCHAR(10), major VARCHAR(2), lvl VARCHAR(2), age INT, primary key(snum));

CREATE TABLE FACULTY(fid INT,fname VARCHAR(20), deptid INT, PRIMARY KEY(fid));

CREATE TABLE CLASS(cname VARCHAR(20), meets\_at TIMESTAMP, room VARCHAR(10), fid INT, PRIMARY KEY(cname), FOREIGN KEY(fid) REFERENCES faculty(fid));

CREATE TABLE ENROLLED(snum INT, cname VARCHAR(20), PRIMARY KEY(snum,cname), FOREIGN KEY(snum) REFERENCES student(snum), FOREIGN KEY(cname) REFERENCES class(cname));

SHOW TABLES;

Query 1:- Find the names of all Juniors (level(lvl) = Jr) who are enrolled in a class taught by james.

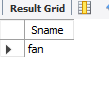
SELECT DISTINCT S.Sname

FROM Student S, Class C, Enrolled E, Faculty F

WHERE S.snum = E.snum AND E.cname = C.cname AND C.fid = F.fid AND

F.fname = 'James' AND S.level = 'Jr';

Output:-



Query 2:- Find the names of all classes that either meet in room R128 or have five or more Students enrolled.

SELECT C.cname

FROM Class C

WHERE C.room = 'R128'

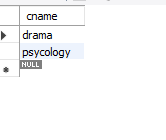
OR C.cname IN (SELECT E.cname

FROM Enrolled E

GROUP BY E.cname

HAVING COUNT(\*) >= 5);

Output:-



Query 3:- Find the names of all students who are enrolled in two classes that meet at the same time.

SELECT DISTINCT S.sname

FROM STUDENT S

WHERE S.snum IN (SELECT E1.snum

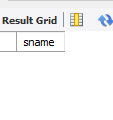
FROM ENROLLED E1, ENROLLED E2, CLASS C1, CLASS C2

WHERE E1.snum = E2.snum AND E1.cname <> E2.cname

AND E1.cname = C1.cname

AND E2.cname = C2.cname AND C1.meets\_at = C2.meets\_at);

Output:-



Query 6:- Find the names of students who are not enrolled in any class.

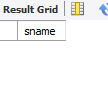
SELECT DISTINCT S.sname

FROM Student S

WHERE S.snum NOT IN (SELECT E.snum

FROM Enrolled E);

Output:-



Query 7:- For each age value that appears in Students, find the level value that appears most often. For example, if there are more FR level students aged 18 than SR, JR, or SO students aged 18, you should print the pair (18, FR).

SELECT S.age, S.level

FROM Student S

GROUP BY S.age, S.level

HAVING S.level IN (SELECT S1.level FROM Student S1

WHERE S1.age = S.age

GROUP BY S1.level, S1.age

HAVING COUNT(\*) >= ALL (SELECT COUNT(\*)

FROM Student S2

WHERE s1.age = S2.age

GROUP BY S2.level, S2.age));

Output:-

