NITTE (Deemed to be University) | NMAM INSTITUTE OF TECHNOLOGY

Nitte(DU) established under Section 3 of UGC Act 1956 | Accredited with 'A+' Grade by NAAC

Department of Artificial Intelligence & Machine Learning Engineering

LAB MANUAL

DATABASE SYSTEMS LABORATORY 21AM403

Academic Year
2022-2023

LIBRARY DATABASE

Consider the following schema for a Library Database:
BOOK(Book_id, Title, Publisher_Name, Pub_Year)
BOOK_AUTHORS(Book_id, Author_Name)
PUBLISHER(Name, Address, Phone)
BOOK_COPIES(Book_id, Branch_id, No-of_Copies)
BOOK_LENDING(Book_id, Branch_id, Card_No, Date_Out, Due_Date)
LIBRARY_BRANCH(Branch_id, Branch_Name, Address)
Write SQL queries to

1. Retrieve details of all books in the library – id, title, name of publisher, authors, number of copies in each branch, etc.
2. Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017.
3. Delete a book in BOOK table. Update the contents of other tables to reflect this data manipulation operation.
4. Create a view of all books and its number of copies that are currently available in the Library.
5. Retrieve the details of publisher who published more than 3 books.
6. Retrieve the details of publisher who has not published any books.
7. Get the particulars of book with more than 3 authors.
8. Get the particulars of Library branch which has zero copies of book with id 112.

```
create table publisher
(
name varchar(10),
address varchar(10),
phone bigint,
primary key(name)
);

create table book
(
book_id varchar(5),
title varchar(20),
publisher_name varchar(10),
publisher_year int,
primary key(book_id),
foreign key(publisher_name) references publisher(name) on delete cascade
);

create table book_authors
(
book_id varchar(5),
author_name varchar(15),
primary key(book_id,author_name),
foreign key(book_id) references book(book_id) on delete cascade
);

create table library_branch
(
branch_id varchar(5),
branch_name varchar(10),
address varchar(15),
primary key(branch_id),
);

create table book_copies
(
book_id varchar(5),
branch_id varchar(5),
no_of_copies int,
primary key(book_id,branch_id),
foreign key(book_id) references book(book_id) on delete cascade,
foreign key(branch_id) references library_branch(branch_id) on delete cascade
);

create table book_lending
(
book_id varchar(5),
branch_id varchar(5),
card_no varchar(5),
date_out date,
due_date date,
primary key(book_id,branch_id,card_no),
foreign key(book_id) references book(book_id),
foreign key(branch_id) references library_branch(branch_id) on delete cascade;
);

select * from publisher;
```

NAME	ADDRESS	PHONE
mcgraw	noida	9085467001
phi	pune	9945467800
pearson	nagpur	7875622333

```
select * from library_branch;
```

BRANCH_ID	BRANCH_NAME	ADDRESS
11	vc campus	puttur
12	pvs	mangalore
13	mg road	bangalore

```
select * from book;
```

BOOK_ID	TITLE	PUBLISHER_NAME	PUBLISHER_YEAR
111	management	mcgraw	2010
112	computer networks	pearson	2006
113	database concepts	pearson	2014
115	entrepreneurship	pearson	2010
114	formal languages	mcgraw	2006
116	embedded systems	mcgraw	2014
117	programming in java	phi	2010

select * from book_authors;

BOOK_ID	AUTHOR_NAME
111	tripathy, reddy
112	larry peterson
113	ramez,navathe
114	john e hopcroft
115	vasant desai
116	rajkamal
117	herbert schildt

select * from book_copies;

BOOK_ID	BRANCH_ID	NO_OF_COPIES
111	11	5
111	13	10
112	12	5
113	11	12
113	12	20
114	13	15
115	11	9
115	13	25
116	13	5
117	12	5

select * from book_lending;

BOOK_ID	BRANCH_ID	CARD_NO	DATE_OUT	DUE_DATE
111	11	1111	10-JUL-17	20-JUL-17
111	13	1112	13-JUL-17	23-JUL-17
114	13	1113	05-JUL-17	15-JUL-17
115	13	1113	10-JUN-17	20-JUN-17
116	13	1113	15-JUL-17	25-JUL-17
111	13	1113	23-MAR-17	02-APR-17
111	13	1114	20-MAR-17	30-MAR-17
113	11	1111	02-APR-17	12-APR-17
113	12	1111	05-MAY-17	15-MAY-17
115	11	1111	02-FEB-17	12-FEB-17

1. select b.book_id, b.title, b.publisher_name, ba.author_name, bc.branch_id, bc.no_of_copies from book b,book_authors ba, book_copies bc where b.book_id = bc.book_id and b.book_id = ba.book_id;

BOOK_ID	TITLE	PUBLISHER_NAME	AUTHOR_NAME	BRANCH_ID	NO_OF_COPIES
111	management	mcgraw	tripathy, reddy	11	5
111	management	mcgraw	tripathy, reddy	13	10
112	computer networks	pearson	larry peterson	12	5
113	database concepts	pearson	ramez,navathe	11	12
113	database concepts	pearson	ramez,navathe	12	20

114	formal languages	mcgraw	john e hopcroft	13	15
115	entrepreneurship	pearson	vasant desai	11	9
115	entrepreneurship	pearson	vasant desai	13	25
116	embedded systems	mcgraw	rajkamal	13	5
117	programming in java	phi	herbert schildt	12	5

2.
```
select distinct card_no from book_lending b where (date_out between '01-jan-2017' and '30-jul-2017') group by card_no having count(*)>3;
```

CARD_NO
1113
1111

3.
```
delete from book where book_id='112';
```

4.
```
create view available as
(
select book_id,sum(no_of_copies) - (select count(card_no) from book_lending
where b.book_id = book_id) as avail_copies
from book_copies b group by book_id
);

select * from available;
```

BOOK_ID	AVAIL_COPIES
113	30
115	32
117	5
112	5
116	4
111	11
114	14

OR

```
select book_id, branch_id, sum(no_of_copies) - (select count(card_no) from book_lending
where b.book_id = book_id and b.branch_id = branch_id) as avail_copies
from book_copies b group by (book_id,branch_id);
```

BOOK_ID	BRANCH_ID	AVAIL_COPIES
115	11	8
111	13	7
111	11	4
116	13	4
112	12	5
113	11	11
114	13	14
117	12	5
113	12	19

115	13	24

MOVIE DATABASE

Consider the schema for Movie Database:
ACTOR(Act_id, Act_Name, Act_Gender)
DIRECTOR(Dir_id, Dir_Name, Dir_Phone)
MOVIES(Mov_id, Mov_Title, Mov_Year, Mov_Lang, Dir_id)
MOVIE_CAST(Act_id, Mov_id, Role)
RATING(Rat_id,Mov_id, Rev_Stars)

Write SQL queries to

1. List the titles of all movies directed by ‘Hitchcock’.
2. Find the movie names where one or more actors acted in two or more movies.
3. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).
4. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.
5. Update the rating of all movies directed by ‘Steven Spielberg’ to 5.
6. Find the count of movies released in each year in each language.
7. Find the total number of movies directed by each director.

```
create table actor
(
act_id varchar(5),
act_name varchar(15),
act_gender varchar(6),
primary key(act_id)
);

create table director
(
dir_id varchar(5),
dir_name varchar(15),
dir_phone int,
primary key(dir_id)
);

create table movies
(
mov_id varchar(5),
mov_title varchar(15),
mov_year int,
mov_lang varchar(10),
dir_id varchar(5),
primary key(mov_id),
foreign key(dir_id)  references director(dir_id) on delete cascade
);

create table movie_cast
(
act_id varchar(5),
mov_id varchar(5),
role varchar(10),
primary key(act_id,mov_id),
foreign key(act_id) references actor(act_id) on delete cascade,
foreign key(mov_id) references movies(mov_id) on delete cascade
);

create table rating
(
rat_id varchar(5),
mov_id varchar(5),
rev_stars int,
primary key(rat_id),
foreign key(mov_id) references movies(mov_id) on delete cascade
);

select * from actor;
```

ACT_ID	ACT_NAME	ACT_GENDER
1	suriya	male
3	dicaprio	male
5	bipasha	female
2	ajay	male
4	cameron diaz	female

```
select * from director;
```

DIR_ID	DIR_NAME	DIR_PHONE
11	rohith shetty	9087600123
12	murugadoss	7789563456
13	martin	8954623150

```
select * from movies;
```

MOV_ID	MOV_TITLE	MOV_YEAR	MOV_LANG	DIR_ID
hm1	singham	2011	hindi	11
hm2	zameen	2003	hindi	11
hm3	golmaal	2010	hindi	11
hm4	all the best	2009	hindi	11
tm1	ghajini	2005	tamil	12
em1	gangs of ny	2002	english	13
em2	the departed	2006	english	13
em3	wallstreet	2013	english	13

```
select * from movie_cast;
```

ACT_ID	MOV_ID	ROLE
1	tm1	hero
3	em1	lead
3	em2	lead
3	em3	lead
2	hm1	hero
2	hm2	colonel
2	hm3	lead

2	hm4	lead
4	em1	supporting
5	hm3	supporting
5	hm4	supporting

```
select * from rating;
```

RAT_ID	MOV_ID	REV_STARS
1	tm1	5
2	em1	5
3	em1	3
4	em2	2
5	hm1	5
6	hm1	3
7	hm3	3
8	hm3	2

```
1. select mov_title
from movies m, director d
where m.dir_id = d.dir_id and d.dir_name ='martin';
```

MOV_TITLE
gangs of ny
the departed
wallstreet

```
2. select distinct mov_title from movies m, movie_cast mc
where m.mov_id = mc.mov_id and
                    (select count(mov_id)
                    from movie_cast
                    where act_id =mc.act_id)>=2;
```

MOV_TITLE
singham
golmaal
wallstreet
the departed
gangs of ny
zameen
all the best

```
3. select act_name
from actor a join movie_cast mc on a.act_id = mc.act_id join movies m
on mc.mov_id = m.mov_id
where m.mov_year<2004
intersect
select act_name
from actor a join movie_cast mc on a.act_id = mc.act_id join movies m
```

```
on mc.mov_id = m.mov_id
where m.mov_year>2010;
```

ACT_NAME
ajay
dicaprio

```
4.select mov_title,max(rev_stars)
from movies m, rating r
where m.mov_id = r.mov_id group by (m.mov_title,m.mov_id) order by m.mov_title;
```

MOV_TITLE	MAX(REV_STARS)
gangs of ny	5
ghajini	5
golmaal	3
singham	5
the departed	2

```
5.update rating set rev_stars=5
where mov_id in
                    (select m.mov_id
                    from movies m,director d
                    where m.dir_id = d.dir_id and d.dir_name='martin');
```

Before

RAT_ID	MOV_ID	REV_STARS
1	tm1	5
2	em1	5
3	em1	3
4	em2	2
5	hm1	5
6	hm1	3
7	hm3	3
8	hm3	2

After

RAT_ID	MOV_ID	REV_STARS
1	tm1	5
2	em1	5
3	em1	5
4	em2	5
5	hm1	5
6	hm1	3
7	hm3	3
8	hm3	2

ORDER DATABASE

Consider the following schema for Order Database:
SALESMAN(Salesman_id, Name, City, Commission)
CUSTOMER(Customer_id, Cust_Name, City, Grade, Salesman_id)
ORDERS(Ord_No, Purchase_Amt, Ord_Date, Customer_id, Salesman_id)

Write SQL queries to

1. Count the customers with grades above Bangalore’s average.
2. Find the name and numbers of all salesman who had more than one customer.
3. List all the salesman and indicate those who have and don’t have customers in their cities (Use UNION operation.)
4. Create a view that finds the salesman who has the customer with the highest order of a day.
5. Demonstrate the delete operation by removing salesman with id 1000. All his orders must also be deleted.

```
create table salesman
(
        salesman_id varchar(5),
        name varchar(15),
        city varchar(15),
        commission int,
        primary key(salesman_id)
);
```

```
create table customer
(
        customer_id varchar(5),
        cust_name varchar(15),
        city varchar(15),
        grade int,
        salesman_id varchar(5),
        primary key(customer_id),
        foreign key(salesman_id) references salesman(salesman_id) on delete cascade
);
```

```
create table orders
(
        ord_no varchar(5),
        purchase_amt int,
        ord_date date,
        customer_id varchar(5),
        salesman_id varchar(5),
        primary key(ord_no),
        foreign key(customer_id) references customer(customer_id) on delete cascade,
        foreign key(salesman_id) references salesman(salesman_id) on delete cascade
);
```

```
select * from salesman;
```

SALESMAN_ID	NAME	CITY	COMMISSION
1	guru	mangalore	5
2	ravi	puttur	3

3	girish	udupi	3

select * from customer;

CUSTOMER_ID	CUST_NAME	CITY	GRADE	SALESMAN_ID
11	srikanth	puttur	4	2
12	sandeep	mangalore	2	1
13	uday	puttur	3	2
14	mahesh	sullia	2	3
15	shivaram	puttur	2	2
16	shyam	mangalore	5	1

select * from orders;

ORD_NO	PURCHASE_AMT	ORD_DATE	CUSTOMER_ID	SALESMAN_ID
111	2500	11-JUL-17	11	2
112	1999	09-JUL-17	12	1
113	999	12-JUL-17	13	2
114	9999	12-JUL-17	14	3
115	7999	11-JUL-17	15	2
116	1099	09-JUL-17	16	2

1. Count the customers with grades above Bangalore’s average.

```
select count(*) as count
        from customer where grade >
                (select avg(grade)
                from customer
                where city ='puttur');
```

COUNT
2

2. Find the name and numbers of all salesman who had more than one customer.

```
select s.salesman_id,s.name,count(customer_id)
        from salesman s, customer c
        where s.salesman_id = c.salesman_id
        group by (s.salesman_id,s.name) having count(customer_id)>1;
```

SALESMAN_ID	NAME	COUNT(CUSTOMER_ID)
1	guru	2
2	ravi	3

3. List all the salesman and indicate those who have and don’t have customers in their cities (Use UNION operation.)

```
        select name,'exists' as same_city
        from salesman s
        where city in
                (select city
                from customer
                where s.salesman_id = salesman_id)
        union
        select name,'not exists' as same_city
        from salesman s where
        city not in
                (select city
                from customer
                where s.salesman_id = salesman_id);
```

NAME	SAME_CITY
girish	not exists
guru	exists
ravi	exists

4. Create a view that finds the salesman who has the customer with the highest order of a day.

```
create view highest_order as
        select s.salesman_id,s.name,o.purchase_amt,o.ord_date
        from salesman s,orders o
        where s.salesman_id = o.salesman_id;

        select name,ord_date
        from highest_order h
        where purchase_amt =
                (select max(purchase_amt)
                from highest_order
                where h.ord_date = ord_date);
```

NAME	ORD_DATE
guru	09-JUL-17
girish	12-JUL-17
ravi	11-JUL-17

5. Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also be deleted.

```
delete from salesman where salesman_id =3;
```

SALESMAN_ID	NAME	CITY	COMMISSION
1	guru	mangalore	5
2	ravi	puttur	3

ORD_NO	PURCHASE_AMT	ORD_DATE	CUSTOMER_ID	SALESMAN_ID
111	2500	11-JUL-17	11	2
112	1999	09-JUL-17	12	1
113	999	12-JUL-17	13	2
115	7999	11-JUL-17	15	2

116	1099	09-JUL-17	16	2

CUSTOMER_ID	CUST_NAME	CITY	GRADE	SALESMAN_ID
11	srikanth	puttur	4	2
12	sandeep	mangalore	2	1
13	uday	puttur	3	2
15	shivaram	puttur	2	2
16	shyam	mangalore	5	1

COLLEGE DATABASE
Consider the schema for College Database:
STUDENT(USN, SName, Address, Phone, Gender)
SEMSEC(SSID, Sem, Sec)
CLASS(USN, SSID)
SUBJECT(Subcode, Title, Sem, Credits)
IAMARKS(USN, Subcode, SSID, Test1, Test2, Test3, FinalIA)

Write SQL queries to
1. List all the student details studying in fourth semester ‘C’ section.
2. Compute the total number of male and female students in each semester and in each section.
3. Create a view of Test1 marks of student USN ‘1BI15CS101’ in all subjects.
4. Calculate the FinalIA (average of best two test marks) and update the corresponding table for all students.
5. Categorize students based on the following criterion:
If FinalIA = 17 to 20 then CAT = ‘Outstanding’
If FinalIA = 12 to 16 then CAT = ‘Average’
If FinalIA< 12 then CAT = ‘Weak’
GIVE THESE DETAILS ONLY FOR 8 TH SEMESTER A, B, AND C SECTION STUDENTS.

```
create table student
(
        usn varchar(10),
        sname varchar(15),
        address varchar(15),
        phone int,
        gender varchar(6),
        primary key(usn)
);


create table semsec
(
        ssid varchar(5),
        sem int,
        sec varchar(1),
        primary key(ssid)
);

create table class
(
        usn varchar(10          ),
        ssid varchar(5),
        primary key(usn),
        foreign key(usn) references student(usn) on delete cascade,
        foreign key(ssid) references semsec(ssid) on delete cascade
);

create table subject
(
        subcode varchar(7),
        title varchar(15),
        sem int,
        credits int,
        primary key(subcode)
);

create table iamarks
(
        usn varchar(7),
        subcode varchar(7),
        ssid varchar(5),
        test1 int,
        test2 int,
        test3 int,
        final_ia int,
        primary key(usn,subcode,ssid),
        foreign key(usn) references student(usn) on delete cascade,
        foreign key(subcode) references subject(subcode),
        foreign key(ssid) references semsec(ssid)
);

select * from student;
```

USN	SNAME	ADDRESS	PHONE	GENDER
4vp14cs001	namitha	udupi	7860054110	female
4vp14cs021	mithun	virajpet	8762514991	male
4vp14cs053	kshama	puttur	9000876123	female
4vp14is010	raghavendra	karwar	8700967408	male
4vp14is025	sooraj	bangalore	7773334422	male
4vp13cs011	karthik	puttur	7789086125	male

```
select * from semsec;
```

SSID	SEM	SEC
a235	4	a
a234	4	b
a218	6	a

```
select * from class;
```

USN	SSID
4vp14cs001	a235
4vp14cs021	a235
4vp14cs053	a234
4vp14is010	a234
4vp14is025	a234
4vp13cs011	a218

```
select * from subject;
```

SUBCODE	TITLE	SEM	CREDITS
15cs41	algorithms	4	4
15cs462	graph theory	4	3
15cs44	processors	4	4
15cs45	oop with c++	4	4
15cs64	networks	6	4

```
select * from iamarks;
```

USN	SUBCODE	SSID	TEST1	TEST2	TEST3
4vp14cs001	15cs41	a235	19	18	20
4vp14cs001	15cs44	a235	15	18	19
4vp14cs021	15cs41	a235	15	17	14
4vp14cs021	15cs44	a235	17	16	12
4vp14cs053	15cs462	a234	13	17	15
4vp14cs053	15cs45	a234	13	17	19
4vp14is010	15cs462	a234	12	19	13
4vp14is010	15cs45	a234	19	17	16
4vp14is025	15cs462	a234	19	17	18
4vp14is025	15cs45	a234	17	15	16
4vp13cs011	15cs64	a218	17	18	19
4vp13cs011	15cs45	a218	10	11	9

1. List all the student details studying in fourth semester ‘C’ section.

```
        select s.usn, sname, gender, address
        from student s, semsec sc ,class c
        where s.usn=c.usn and c.ssid= sc.ssid and sc.sem = 4 and sc.sec ='a';
```

USN	SNAME	GENDER	ADDRESS
4vp14cs001	namitha	female	udupi
4vp14cs021	mithun	male	virajpet

2. Compute the total number of male and female students in each semester and in each section.

```
        select sem, sec, gender, count(*) as count
        from student s,semsec sc, class c
        where s.usn = c.usn and sc.ssid = c.ssid
        group by (sem,sec,gender);
```

SEM	SEC	GENDER	COUNT
4	b	female	1
4	a	female	1
6	a	male	1
4	b	male	2
4	a	male	1

3. Create a view of Test1 marks of student USN ‘1BI15CS101’ in all subjects.

```
create view test1 as
(
        select usn, marks1, subcode
        from iamarks
        where usn='4vp14is025'
);
select * from test1;
```

USN	TEST1	SUBCODE
4vp14is025	19	15cs462
4vp14is025	17	15cs45

4. Calculate the FinalIA (average of best two test marks) and update the corresponding table for all students.

```
create view average_finder as
(
        select usn,subcode,greatest(test1,test2,test3) as highest, case
        when test1<greatest(test1,test2,test3) and test1>least(test1,test2,test3) then test1
        when test2<greatest(test1,test2,test3) and test2>least(test1,test2,test3) then test2
        else test3
        end as second_highest from iamarks
);

update iamarks a set final_ia =
        (select (highest+second_highest)/2 from average_finder
        where a.usn =usn and a.subcode= subcode);
```

USN	SUBCODE	SSID	TEST1	TEST2	TEST3	FINAL_IA
4vp14cs001	15cs41	a235	19	18	20	20
4vp14cs001	15cs44	a235	15	18	19	19
4vp14cs021	15cs41	a235	15	17	14	16
4vp14cs021	15cs44	a235	17	16	12	17
4vp14cs053	15cs462	a234	13	17	15	16
4vp14cs053	15cs45	a234	13	17	19	18
4vp14is010	15cs462	a234	12	19	13	16
4vp14is010	15cs45	a234	19	17	16	18
4vp14is025	15cs462	a234	19	17	18	19
4vp14is025	15cs45	a234	17	15	16	17
4vp13cs011	15cs64	a218	17	18	19	19
4vp13cs011	15cs45	a218	10	11	9	11

5. Categorize students based on the following criterion:
If FinalIA = 17 to 20 then CAT = ‘Outstanding’
If FinalIA = 12 to 16 then CAT = ‘Average’
If FinalIA< 12 then CAT = ‘Weak’
Give these details only for 8 th semester A, B, and C section students.

```
select usn, subcode, case
when final_ia>=17 and final_ia<=20 then 'outstanding'
when final_ia>=12 and final_ia<=16 then 'average'
when final_ia<12 then 'weak'
end as category
from iamarks
where usn in
        (SELECT USN FROM SEMSEC SC,CLASS C WHERE SC.SSID=C.SSID AND SEM=4);
```

USN	SUBCODE	CATEGORY
4vp14cs001	15cs41	outstanding
4vp14cs001	15cs44	outstanding
4vp14cs021	15cs41	average
4vp14cs021	15cs44	outstanding
4vp14cs053	15cs462	average
4vp14cs053	15cs45	outstanding
4vp14is010	15cs462	average
4vp14is010	15cs45	outstanding
4vp14is025	15cs462	outstanding
4vp14is025	15cs45	outstanding

EMPLOYEE DATABASE

Consider the schema for Company Database:
EMPLOYEE(SSN, Name, Address, Sex, Salary, SuperSSN, DNo)
DEPARTMENT(DNo, DName, MgrSSN, MgrStartDate)
DLOCATION(DNo,DLoc)
PROJECT(PNo, PName, PLocation, DNo)
WORKS_ON(SSN, PNo, Hours)

Write SQL queries to

1. Make a list of all project numbers for projects that involve an employee whose last name is ‘Scott’, either as a worker or as a manager of the department that controls the project.
2. Show the resulting salaries if every employee working on the ‘IoT’ project is given a 10 percent raise.
3. Find the sum of the salaries of all employees of the ‘Accounts’ department, as well as the maximum salary, the minimum salary, and the average salary in this department
4. Retrieve the name of each employee who works on all the projects controlledby department number 5 (use NOT EXISTS operator).
5. For each department that has more than five employees, retrieve the department number and the number of its employees who are making more THAN RS. 6,00,000.

```
create table department
(
dno varchar(5),
dname varchar(15),
mgrssn varchar(5),
mgrstartdate date,
primary key(dno),
);

create table employee
(
ssn varchar(5),
name varchar(15),
address varchar(15),
sex varchar(6),
salary int,
superssn varchar(5),
dno varchar(5),
primary key(ssn),
);

alter table employee add constraint fk1 foreign key(dno) references department(dno) on delete cascade;

alter table employee add constraint fk2 foreign key(superssn) references employee(ssn) on delete cascade

alter table department add constraint fk3 foreign key(mgrssn) references employee(ssn) on delete cascade


create table dlocation
```

```
(
dno varchar(5),
dloc varchar(15),
primary key (dno,dloc),
foreign key(dno) references department(dno) on delete cascade
);

create table project
(
pno varchar(5),
pname varchar(10),
plocation varchar(10),
dno varchar(5),
primary key(pno),
foreign key(dno) references department(dno) on delete cascade
);

create table works_on
(
ssn varchar(5),
pno varchar(5),
hours int,
primary key(ssn,pno),
foreign key(ssn) references employee(ssn) on delete cascade,
foreign key(pno) references project(pno) on delete cascade
);

select * from department;
```

DNO	DNAME	MGRSSN	MGRSTARTDATE
1	networks	111	10-JUN-13
2	data mining	333	17-OCT-10

```
select * from employee;
```

SSN	NAME	ADDRESS	SEX	SALARY	SUPERSSN	DNO
555	brian a smith	texas	male	700000	222	2
666	alicia zelaya	colarado	female	500000	333	1
777	julian smith	las vegas	female	700000	333	2
111	john b smith	nevada	male	650000		1
222	ramesh narayan	nebraska	male	800000	111	2
333	ahmad jabbar	san jose	male	750000	111	1
444	joyce a english	los angeles	female	650000	222	1

```
select * from dlocation;
```

DNO	DLOC
1	stanford
2	houston
1	vegas
2	texas

```
select * from project;
```

PNO	PNAME	PLOCATION	DNO
11	iot	texas	1
12	webmining	texas	2
13	sensors	vegas	1
14	routing	stanford	1
15	cluster	houston	2

```
select * from works_on;
```

SSN	PNO	HOURS
555	11	4
666	12	4
666	15	3
666	13	3
777	14	2
222	11	4
444	15	3
777	15	3

```
1. select distinct pno
from project
where pno in
        (select p.pno
        from project p,department d,employee e
        where p.dno = d.dno and d.mgrssn = e.ssn and name like '%smith')
        or
        pno in
        (select pno
        from works_on w,employee e
        where w.ssn = e.ssn and name like '%smith');
```

PNO
13
11
14
15

```
2. select e.name, e.salary*1.1 as new_salary
from  employee e, works_on w
where e.ssn = w.ssn and
w.pno in
        (select pno
        from project
        where pname ='iot');
```

NAME	NEW_SALARY
brian a smith	770000

ramesh narayan	880000

```
3. select sum(salary), max(salary), min(salary), avg(salary)
from (employee e join department d on d.dno = e.dno)
where d.dname = 'data mining';
```

SUM(SALARY)	MAX(SALARY)	MIN(SALARY)	AVG(SALARY)
2200000	800000	700000	733333.333333333

```
4. select e.name from employee e
where not exists ((select pno from project where dno=2) minus
(select pno from works_on where ssn = e.ssn));
```

NAME
alicia zelaya

```
5. select d.dno,count(*) as count
from department d,employee e
where d.dno= e.dno and salary >600000
and d.dno in
        (select dno from employee
        group by dno having count(*)>3)
        group by d.dno;
```

DNO	COUNT
1	3