CISC 7510X Final Exam

For the below questions, use the following schema definition.

```
contact(cid, fname, lname, street1, stree2, city, state, zip, dob)
phone(pid, cid, phone, typ)
email(emailid, cid, email, typ)
event(eid, title, starttim, endtim, location, phone)
invitelst(iid, eid, cid, typ)
```

It is a schema for contacts and calendar. The `contact` table has contact details, the `phone` and `email` tables have the phones and emails associated with the contact, with `typ` indicating the type, such as home, work, mobile, etc. The `events` table has the calendar events such as "Review for 7510 Final Exam" starting at "2025-05-16 18:05:00" and ending at "2025-05-16 20:10:00" at "Brooklyn College, 232IA", with phone of "+1 424-327-5528", etc. The `invitelst` has the invited contacts to the calendar events, with `typ` indicating required or optional.

Each question is worth 5 points. You get 1 point for leaving an answer blank. You get no points for a wrong answer.

1. Find John Doe's zip code.
   (a) select zip from calendar where fname='John' and lname='Doe'
   (b) select zipcode from contact where fname='John' and lname='Doe'
   (c) select zip from contact where fname='John' and lname='Doe'
   (d) select * from contact where fname='John' and lname='Doe'
   (e) Other:
2. Find all of John Doe's phone numbers.
   (a) select * from calendar inner join phone where fname='John' and lname='Doe'
   (b) select phone from contact inner join phone using(cid) where fname='John' and lname='Doe'
   (c) select phone from contact natural inner join event where fname='John' and lname='Doe'
   (d) select phone from contact inner join event using(cid) where fname='John' and lname='Doe'
   (e) Other:
3. Find all events (event ids) that have more than 5 invited contacts.
   (a) select eid from invitelst group by eid having count(*) > 5
   (b) select eid from invitelst where count(*) over (partition by eid) > 5
   (c) with cnts as (select eid, count(*) from invitelst group by eid)
   select * from cnts where cnt > 5
   (d) with cnts as (select eid, count() over (partition by eid) cnt
   from invitelst) select eid from cnts where cnt > 5
   (e) Other:
4. Find all events (event ids) that have less than 5 invited contacts.
   (a) select a.eid from event a inner join invitelst b on a.eid=b.eid group by a.eid having count(*) < 5

(b) select a.eid from event a left outer join invitelst b on a.eid=b.eid group by a.eid having count(*) < 5

(c) select eid from invitelst group by eid having count(*) < 5

(d) with cnts as (select eid, count(*) cnt from invitelst group by eid)
select * from cnts where cnt < 5

(e) Other:

5. Which month has the most birthdays?

(a) select dob, count(*) from contact group by dob order by 2 desc limit 1

(b) with cnts as (select to_char(dob,'MM') m,row_number() over (order by count(*) desc) rn from contact group by 1) select m from cnts where rn=1

(c) with cnts as (select extract(month from dob) m, rank() over (partition by count(*)) rnk from contact group by 1) select m from cnts where rnk=1

(d) select extract(month from dob) from contact group by 1 having count(*) >= ALL(select extract(month from dob) from contact group by 1)

(e) Other:

6. How many different types of phones are there? (e.g. home phone, work phone, etc.)

(a) select count(*) from phones

(b) select count(*) from phones group by typ

(c) select typ, count(*) from phones

(d) select count(distinct typ) from phones

(e) Other:

7. Count contacts by age group, where age 0-30 is "A", 31-50 is "B", 51-70 is "C", and "D" for older.

(a) select extract(years from age(dob)) grp,count(*)
from contact group by extract(years from age(dob))

(b) with age as (select extract(years from age(dob)) a from contact),
agegrp as (
select case when a<=30 then 'A' when a<=50 then 'B'
when a<=70 then 'C' else 'D' end g from age)
select g,count(*) from agegrp group by g

(c) with agegrp as (
select case when extract(years from age(dob))<=30 then 'A'
when extract(years from age(dob))<=50 then 'B'
when extract(years from age(dob))<=70 then 'C'
else 'D' end g
from age)
select g,count(*) from agegrp

(d) with age as (select age(dob) a from contact),
agegrp as (select case when a<=30 then 'A' when a<=50 then 'B'
when a<=70 then 'C' else 'D' end g from age)
select g,count(*) from agegrp group by g

(e) Other:

8. What percentage of contacts live in NY tri-state area (NY,NJ,CT)?

   (a) select 100*sum(
   case when state in ('NY','NJ','CT') then 1.0 else 0.0 end)/sum(1.0)
   from contact

   (b) select 100*sum(
   case when state in ('NY','NJ','CT') then 1.0 else 0.0 end)/sum(1.0)
   from contact
   where state in ('NY','NJ','CT')
   group by state
   having state in ('NY','NJ','CT')

   (c) select 100*sum(
   case when state in ('NY','NJ','CT') then 1.0 else 0.0 end)/sum(1.0)
   from contact
   group by state

   (d) select 100*sum(
   case when state in ('NY','NJ','CT') then 1.0 else 0.0 end)/sum(1.0)
   from contact
   where (case when state in ('NY','NJ','CT') then 'NYTRI' else 'NOT' end)='NYTRI'
   group by case when state in ('NY','NJ','CT') then 'NYTRI' else 'NOT' end
   having count(*)>0

   (e) Other:

9. Find potential dups, first name, last name, and dob the same for different contacts. List the first name, last name, dob, and number of dups.

   (a) select a.fname, a.lname, a.dob, 'DUP' as lbl from contact a inner join contact b on a.fname=b.fname and a.lname=b.lname and a.dob=b.dob where a.cid < b.cid

   (b) with cnts as (select fname,lname,dob,count(*) from contact group by 1,2,3) select * from cnts where cnt>1

   (c) with cnts as (select fname,lname,dob, count() over (partition by fname,lname,dob) cnt from contact) select * from cnts where cnt>1

   (d) select fname,lname,dob, count(*) from contact group by fname,lname,dob having count(*) > 1

   (e) Other:

10. Find all meetings (event ids) that have John Doe on the invite list.

    (a) select b.eid from contact a inner join invitelst b on a.cid=b.cid where a.fname='John' and a.lname='Doe'

    (b) select a.eid from event a inner join invitelst b using(eid) inner join contact c using(cid) where c.fname='John' and c.lname='Doe'

    (c) select a.eid from event a full outer join invitelst b using(eid) full outer join contact c using(cid) where c.fname='John' and c.lname='Doe'

    (d) select a.eid from event a left outer join invitelst b using(eid) left outer join contact c using(cid) where c.fname='John' and c.lname='Doe'

(e) Other:

11. Find all meetings that *do not* have John Doe on the invite list.

    (a) select b.eid from contact a left outer join invitelst b on a.cid=b.cid where a.fname='John' and a.lname='Doe' and b.cid is null

    (b) select a.eid from event a left outer join invitelst b on a.eid=b.eid left outer join contact c on b.cid=c.cid group by a.eid having sum(case when c.fname='John' and c.lname='Doe' then 1 else 0 end)=0

    (c) select a.eid from event a left outer join invitelst b using(eid) inner join contact c using(cid) where c.fname='John' and c.lname='Doe' and a.eid is null

    (d) select a.eid from event a inner join invitelst b on a.eid=b.eid inner join contact c on b.cid=c.cid group by a.eid having sum(case when c.fname='John' and c.lname='Doe' then 1 else 0 end)=0

    (e) Other:

12. Find all meetings that have both John Doe and Jane Doe.

    (a) select a.eid from invitelst a inner join contact b using(cid) where b.lname='Doe' group by a.eid having max(case when fname in('John','Jane') then 1 else 0 end)=1

    (b) select a.eid from event a inner join invitelst b on a.eid=b.eid inner join contact c on b.cid=c.cid group by a.eid having sum(case when c.fname in ('John','Jane') and c.lname='Doe' then 1 else 0 end)>1

    (c) select a.eid from invitelst a inner join contact b on a.cid=b.cid where b.fname in ('John','Jane') and b.lname='Doe'

    (d) select a.eid from invitelst a inner join contact b on a.cid=b.cid where b.lname='Doe' group by a.eid having max(case when fname='John' then 1 else 0 end)=1 and max(case when fname='Jane' then 1 else 0 end)=1

    (e) Other:

13. Find contacts (contact ids) who were never invited to a meeting.

    (a) select cid from invitelst group by cid having count(*)=0

    (b) select a.cid from contact a left outer join invitelst b on a.cid=b.cid where b.cid is null

    (c) select a.cid from contact a inner join invitelst b on a.cid=b.cid group by a.cid having count(*)=0

    (d) select a.cid from contact a left outer join invitelst b on a.cid=b.cid group by a.cid having count(*)=0

    (e) Other:

14. Find calendar conflicts. Instances when one meeting starts during the other.

    (a) select * from event a inner join event b on a.starttim > b.starttim and a.starttim < b.endtim

    (b) select a.endtim - a.starttim from event group by 1 having count(*)>1

    (c) select * from event a inner join event b on a.starttim between b.starttim and b.endtim

(d) select a.starttim from event group by a.startim having count(*)>1

(e) Other:

15. Find instances when time is triple-booked (e.g. has 3 or more meeting scheduled for the same time)

    (a) select * from event a inner join event b on a.starttim > b.starttim and a.starttim < b.endtim
    inner join event c on a.starttim > c.starttim and a.starttim < c.endtim and b.starttim > c.starttim and b.starttim < c.endtim

    (b) with blah as (select starttim t,1 c from event union all select endtim t,-1 c from event), cn as (select t,sum(c) over (order by t) c from blah) select * from cn where c > 2

    (c) select starttim,endtim from event group by starttim,endtim having count(*) > 2

    (d) with blah as (select case when start then starttim else endtime end as t, case when start then 1 else -1 end as c from event),
    cn as (select t,sum(c) over (order by t) c from blah) select * from cn where c > 2

    (e) Other:

16. Write query to identify all the free time (time when there are no events). Use `now()` as starting time, and Jan 1st year 2100 as the end.

17. Write a query to insert into events table all the "free time events" identified in previous query (note that after this, there should be no more free time in the calendar).

18. The below code (tip: run the code):

```
with recursive n(n) as (
    select 2 n union all
    select n+1 from n where n<1000
)
select a.n
from n a inner join n b on b.n < sqrt(a.n)+1
group by a.n
having a.n=2 or min(a.n % b.n) > 0 order by 1
```

(a) Is invalid

(b) Will generate a list of numbers 1 to 1000

(c) Will generate a list of all primes between 1 and 1000

(d) Will generate a list of all odd numbers divisible by 3.

(e) Other:

19. Below query is identical to:

```
select a.*,b.val
from T1 a
left outer join T2 b
on a.key=b.key and a.val!=b.val
```

(a)
```
with TMP as (
select a.*,b.val from T1 a inner join T2 b on a.key=b.key
where a.val!=b.val)
select a.*,b.val from T1 a left outer join TMP b on a.key=b.key
```

(b)
```
with TMP as (
select a.*,b.val from T1 a left outer join T2 b on a.key=b.key
where a.val!=b.val)
select a.* from TMP where a.val!=b.val
```

(c)
```
select a.*,b.val
from T1 a inner join T2 b
on a.key=b.key and a.val!=b.val
```

(d) All of the above queries are identical.

(e) None of the queries are identical to the question.

20. When you write:
`select * from T1 a inner join T2 b on a.tim between b.start and b.end`
what is the expected performance?

(a) Hash join, approximately $O(NlogN)$, where N is the number of records in both T1 and T2.

(b) Inner loop join, approximately $O(N^2)$, where N is the number of records in both tables.

(c) Distributed hash join, approximately $O(N)$ to distribute data, and $O(NlogN)$ after distribution.

(d) Sort merge join, approximately $O(N)$, where N is the number of records in both T1 and T2.

(e) Other:

```
contact(cid, fname, lname, street1, stree2, city, state, zip, dob)
phone(pid, cid, phone, typ)
email(emailid, cid, email, typ)
event(eid, title, starttim, endtim, location, phone)
invitelst(iid, eid, cid, typ)
```