CISC 7512X Spring 2022 Final Exam

For the below questions, use the following schema definition.

```
student(sid, fname, lname, dob)
faculty(fid, fname, lname)
department(depid, name)
class(cid, description, credits, depid)
term(tid, season, year)
section(scid, cid, tid, fid, stim, etim, day, room)
register(rid, tim, sid, scid)
grade(rid, tim, grade)
```

It is a schema for a school, with students, faculty, departments, classes, etc. A term reprsents a semester, e.g. Spring 2022. A section is an instance of a particular class, e.g. Database 2 for Spring 2022, meeting from 6-8PM (stim/etim) in Room 234. Register object links a student (sid) to a section. A grade is assigned to a particular registration object (with timestamp often after the term ends).

1. (5 points) Find student id of John Doe.

   (a) `select lname,fname from student where fname=’John’ and lname=’Doe’`

   (b) `select sid from student where fname=’John’ and lname=’Doe’`

   (c) `select * from faculty where fname=’John’ and lname=’Doe’`

   (d) `select fid from faculty where fname=’John’ and lname=’Doe’`

   (e) Other:

2. (5 points) What’s the average age of a student?

   (a) `select avg(age) from student`

   (b) `select avg(dob) from student`

   (c) `select avg(extract(year from dob)) from student`

   (d) `select avg(age(dob)) from student`

   (e) Other:

3. (5 points) Find number of classes in each department.

   (a) `select count(*) from department`

   (b) `select depid, count(*) from department group by depid`

   (c) `select depid, count(*) from class group by depid`

   (d) `select depid, count(*) from section group by depid`

   (e) Other:

4. (5 points) Find names of all classes ever taken by ’John Doe’.

   (a) `select d.description from student a inner join register b using(sid) inner join section c using(scid) inner join class d using(cid) where fname=’John’ and lname=’Doe’`

(b) `select c.name from student a inner join register b using(sid) inner join class c using(cid) where fname=’John’ and lname=’Doe’`

(c) `select d.name from student a inner join register b using(sid) inner join section c using(scid) inner join department d using(depid) where a.fname=’John’ and a.lname=’Doe’`

(d) `select c.name from student a inner join section b using(sid, scid) inner join class c using(cid) where fname=’John’ and lname=’Doe’`

(e) Other:

5. (5 points) Find all students who have more than 100 credits. Assume grades other than ’F’ grant credit.

(a) `select sid from register a inner join grade b on a.rid=b.rid and b.grade!=’F’ inner join section c using(scid) inner join class d using(cid) where d.credits > 100`

(b) `select * from register a inner join grade b on a.rid=b.rid and b.grade!=’F’ inner join section c using(scid) inner join class d using(cid) where sum(d.credits) over (group by sid) > 100`

(c) `with pass as ( select sid, scid from register a inner join grade b on a.rid=b.rid and b.grade!=’F’ ), creditcnt as ( select sid, scid, sum(credits) tot from pass a inner join section b using(scid) inner join class c using (cid) ) select * from creditcnt where tot>100`

(d) `select sid from register a inner join grade b on a.rid=b.rid and b.grade!=’F’ inner join section c using(scid) inner join class d using(cid) group by sid having sum(d.credits)>100`

(e) Other:

6. (5 points) Find faculty who have never taught anything.

(a) `select fid, count(*) from section group by fid having count(fid) = 0`

(b) `select a.fid from faculty a inner join section b on a.fid=b.fid group by a.fid having count(b.fid) = 0`

(c) `select a.fid from faculty a left outer join section b on a.fid=b.fid group by a.fid having count(b.fid) = 0`

(d) `with faccnt as ( select fid, count(*) over (partition by fid) cnt from faculty ) select fid from faccnt where cnt = 0`

(e) Other:

7. (5 points) Find instances where a grade was assigned before the student registered (data errors).

(a) `select * from register a natural inner join grade b on a.rid=b.rid and a.tim > b.tim`

(b) `select * from register a left outer join grade b on a.rid=b.rid and a.tim > b.tim`

(c) `select * from register a inner join grade b on a.rid=b.rid and a.tim > b.tim`

(d) `select * from register a cross join grade b where a.rid=b.rid and a.tim > b.tim`

(e) Other:

8. (5 points) Find sections with more than 8 students.

(a) `select scid from section group by scid having count(scid) > 8`

(b) `select scid from register group by scid having count(sid) > 8`

(c) `select scid from section natural left outer join register group by scid having count(sid) > 8`

(d) `select scid from section natural inner join register group by scid having count(sid) > 8`

(e) Other:

9. (5 points) Find sections with *less* than 8 students.

(a) `select scid from section group by scid having count(scid) < 8`

(b) `select scid from register group by scid having count(sid) < 8`

(c) `select scid from section natural left outer join register group by scid having count(sid) < 8`

(d) `select scid from section natural inner join register group by scid having count(sid) < 8`

(e) Other:

10. (5 points) Find sections with above average number of students.

(a) `with stats as ( select scid, count(*) cnt, avg( count(*) ) over () acnt from section group by scid ) select * from stats where cnt > acnt`

(b) `select scid from section group by scid having count(*) > avg( count(*) ) over ()`

(c) `select scid from section where count(*) > avg( count(*) ) over ()`

(d) `with stats as ( select scid, count(*) cnt from section ) select scid from stats where cnt > avg(cnt) group by scid`

(e) Other:

11. (5 points) Find instances of students double-booked for the same time slot (students who registered for two overlapping classes). e.g. Taking databases 6-8PM, and taking geology from 6-9PM on same day.

(a) `select * from register a inner join section b on a.scid=b.scid inner join register c on a.sid=b.scid inner join section d on c.scid=b.scid and b.scid < d.scid and b.tid = d.tid and b.day = d.day where b.stim >= d.stim and b.stim < d.etim`

(b) select * from register a inner join section b on a.scid=b.scid inner join register c on a.sid=b.scid inner join section d on c.scid=b.scid and b.scid < d.scid and b.tid = d.tid and b.day = d.day where (b.stim >= d.stim and b.stim < d.etim) or (d.stim >= b.stim and d.stim < b.etim)

(c) select * from register a inner join section b on a.scid=b.scid inner join register c on a.sid=b.scid inner join section d on c.scid=b.scid and b.scid < d.scid and b.tid = d.tid and b.day = d.day and a.sid != c.sid where (b.stim >= d.stim and b.stim < d.etim) or (d.stim >= b.stim and d.stim < b.etim)

(d) with reg as (select a.sid, b.tid, b.day, b.stim, b.etim from register a inner join section b on a.scid=b.scid) select * from reg a inner join reg b on a.sid=b.sid and a.tid=b.tid and a.day=b.day where a.stim >= b.stim and a.stim < b.etim

(e) Other:

12. (5 points) For each term, find the fraction of students who pass Databases2 (assume any grade other than ’F’ is passing).

(a) select b.tid, sum(case when d.grade != ’F’ then 1.0 else 0.0 end)/sum(1.0) f from class a inner join section b on a.cid=b.cid inner join register c on b.scid=c.scid inner join grade d on c.rid=d.rid and d.grade != ’F’ where a.description = ’Databases2’ group by b.tid

(b) select b.tid, sum(case when d.grade != ’F’ then 1.0 else 0.0 end)/sum(1.0) f from class a inner join section b on a.cid=b.cid inner join register c on b.scid=c.scid inner join grade d on c.rid=d.rid where a.description = ’Databases2’ group by b.tid

(c) select b.tid, count(c.rid)/count(*) f from class a inner join section b on a.cid=b.cid inner join register c on b.scid=c.scid inner join grade d on c.rid=d.rid where a.description = ’Databases2’ group by b.tid having g.grade != ’F’

(d) select b.tid, sum(case when grade != ’F’ then 1.0 else 0.0 end)/sum(1.0) f from class a inner join section b on a.cid=b.cid inner join register c on b.scid=c.scid where a.description = ’Databases2’ group by b.tid

(e) Other:

13. (5 points) Which department has the most classes?

(a) select depid from class group by depid having count(*) >= ALL(select count(*) from class group by depid)

(b) with stats as (select depid, row_number() over (order by count(*) desc) rn from class group by depid) select * from stats where rn=1

(c) with stats as (select depid, dense_rank() over (order by count(*)) rnk from class group by depid) select * from stats where rnk=1

(d) with stats as (select depid,count(*) cnt from class group by depid), mx as (select max(cnt) mx from stats) select * from stats inner join mx where cnt = mx

(e) Other:

14. (5 points) The most appropriate index type for student.lname column?

    (a) Bitmap index
    (b) Clustered index
    (c) B-tree index
    (d) B-list index
    (e) Other:

15. (5 points) The most appropriate index type for student.sid column?

    (a) Bitmap index
    (b) Clustered index
    (c) B-tree index
    (d) B-list index
    (e) Other:

16. (5 points) The below code (tip: write out the first few output numbers):

```
with recursive n(n) as (
    select 2 n union all
    select n+1 from n where n<1000
)
select a.n
from n a left join n b on b.n <= sqrt(a.n)
group by a.n
having a.n<=3 or min(a.n % b.n) > 0
```

    (a) Is invalid
    (b) Will generate a list of numbers 1 to 1000
    (c) Will create a table with all dates between 19000101 and 21000101
    (d) Will output list of all prime numbers between 1 and 1000
    (e) Other:

17. (5 points) 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)
```
select a.*,b.val from T1 a
inner join T2 b on a.key=b.key and a.val!=b.val
```

(b)
```
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
```

(c)
```
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
```

(d) All of the above queries are identical.

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

18. (5 points) Both table A and table B have about the same number of records N. Assume `key` has no skew. What's the expected peformance of `select * from A inner join B on a.key=b.key`?

    (a) $O(N)$

    (b) $O(\log N)$

    (c) $O(N \log N)$

    (d) $O(N^2)$

    (e) Other:

19. (5 points) Both table A and table B have about the same number of records N. Assume `key` has no skew. What's the expected peformance of `select * from A inner join B on a.key != b.key`?

    (a) $O(N)$

    (b) $O(\log N)$

    (c) $O(N \log N)$

    (d) $O(N^2)$

    (e) Other:

20. (5 points) In general, on limited memory system, no indexes, and huge tables, what join type would perform best?

    (a) hash join.

    (b) indexed lookup join.

    (c) merge join.

    (d) inner loop join.

    (e) Other: