

2.- Show me ids, names, surnames, and marks of the students with mark > 3.6.

```
select id, name, surname, mark  
from STUDENTS  
where mark > 3.6
```

3.- Show me student names, surnames, and majors that they have applied to. Sort the results by name and surname.

```
select name, surname, major  
from STUDENTS S, APPLIES A  
where S.id = A.sid  
order by surname, name;
```

4. Show me names, surnames, marks and application decision of students with size_high_school < 1000 applying to CS at Stanford.

```
select name, surname, mark, decision  
from STUDENTS S, APPLIES A  
where S.id = A.sid and  
    size_high_school < 1000 and  
    major = 'CS' and  
    college = 'Stanford';
```

5. Show me all large campuses (enrollment>20000) with CS applicants.

```
select distinct C.name
```

```
from COLLEGES C, APPLIES A
where C.name = A.college and
      enrollment > 20000 and
      major = 'CS';
```

6.- Show me applicants to bio majors.

```
select A.sid, S.name, S.surname, A.college, A.major
from APPLIES A, STUDENTS S
where A.sid = S.id and
A.major like '%bio%';
```

7. Show the weighted scores on 10 points (now weighted on 5 points).

```
select id, name, surname, mark, (mark*10)/5 Wmark
from STUDENTS
```

or simply

```
select id, name, surname, mark, mark*2 Wmark
from STUDENTS
```

**8. Scale the marks considering the size of the high school
(mark*(high_school_size/1000)). Show the weighted scores on 5 and 10 points.**

```
select id, name, surname, mark, mark*(size_high_school/1000) as ScaledOver5,
((mark*10)/5)*(size_high_school/1000) as ScaledOver10
```

from STUDENTS

9. Show me the average mark of all students applying to CS (you must think that the same student could apply for CS in many Universities). Clue: wrong average 3.714285714285714, right average 3.6800000000000006.

Wrong:

```
select avg(mark) as AverageMark
from STUDENTS S, APPLIES A
where S.id = A.sid and
A.major = 'CS';
```

Fix incorrect counting of marks:

```
select avg(mark) as AverageMark
from STUDENTS
where id in (select distinct sid from APPLIES where major = 'CS')
```

10. Show me the lowest mark of students applying to CS.

```
select min(mark) as MinMark
from STUDENTS S, APPLIES A
where S.id = A.sid and
major = 'CS';
```

11. Show me the number of colleges bigger than 15,000.

```
select count(*) as NumberOfColleges  
from COLLEGES  
where enrollment > 15000;
```

12. Show me the number of students applying to Cornell (you must think that students can apply to many majors in the same University). Clue: The right number should be 3.

Wrong (6):

```
select count(*)  
from APPLIES  
where college = 'Cornell';
```

Right (3):

```
select count(distinct sid)  
from APPLIES  
where college = 'Cornell'
```

13. Show me the number of applications to each college (sorted by college name).

```
select college, count(*)  
from APPLIES  
group by college  
order by college;
```

15. Show me the number of college enrollments by state.

```
select state, sum(enrollment)
from COLLEGES
group by state;
```

16. Show me the minimum and maximum marks of all the students.

```
select min(S.mark) myMark
from STUDENTS S

UNION

select max(S.mark) myMark
from STUDENTS S;
```

17. Show me the difference between the maximum and the minimum marks of all the students.

```
SELECT (max(mark) - min(mark))
from STUDENTS
```

OR

```
select B.myMark - A.myMark
from (select min(S.mark) myMark
      from STUDENTS S) A,
     (select max(S.mark) myMark
      from STUDENTS S) B;
```

18. Show me the number of colleges applied to by each student. Sort the results by student surname, student name and student id.

```
select S.id, S.surname, S.name, count(distinct A.college)
from STUDENTS S, APPLIES A
where S.id = A.sid
group by S.id, S.surname, S.name
order by S.surname, S.name, S.id
```

19. Show me the number of colleges applied to by each student, including 0 for those who applied nowhere. CLUE: Do the union between to queries.

```
select S.surname, S.name, S.id, count(distinct A.college)
from STUDENTS S, APPLIES A
where S.id = A.sid
group by S.id
union
select surname, name, id, 0
from STUDENTS
where id not in (select sid from APPLIES)
```

--with an outer join

```
select S.surname, S.name, S.id, count(distinct A.college)
from STUDENTS S, APPLIES A
where S.id = A.sid
group by S.id
union
```

```
select S.surname, S.name, S.id, count(distinct A.sid)
from STUDENTS S LEFT OUTER JOIN APPLIES A
ON S.id = A.sid
WHERE
S.id not in (select sid from APPLIES)
group by S.surname, S.name, S.id;
```

20. Do the last query sorting by student surname, student name and student id.

```
SELECT *
FROM (select S.surname, S.name, S.id, count(distinct A.college)
from STUDENTS S, APPLIES A
where S.id = A.sid
group by S.id
union
select surname, name, id, 0
from STUDENTS
where id not in (select sid from APPLIES)) S
ORDER BY S.surname, S.name, S.id
```

21. Show me the colleges with fewer than 5 applications. Show only the college name and sort the results by name. Do the exercise with a simple query.

In this result duplicate applications are considered:

In this result duplicate applications are not considered:

Do both queries!

Considering duplicates:

```
select college
```

```
from APPLIES
```

```
group by college
```

```
having count(*) < 5
```

```
order by college
```

Not considering duplicates:

```
select college
```

```
from APPLIES
```

```
group by college
```

```
having count(distinct sid) < 5
```

```
order by college
```

22. Show me colleges with fewer than 5 applications. Show only the college name and sort the results by name. Do the exercise using a subquery.

In this result duplicate applications are considered:

In this result duplicate applications are not considered:

Do both queries!

Duplicated applications: A student that applies to the same college but to different majors.

Considering duplicates:

```
select distinct college
```



```
from APPLIES A1  
  
where 5 > (select count(*) from APPLIES A2 where A2.college = A1.college)  
  
order by college;
```

--or

```
select distinct college  
  
from APPLIES  
  
where college in (select college  
  
from APPLIES  
  
group by college  
  
having count(sid)<5);
```

Not considering duplicates:

```
select distinct college  
  
from APPLIES A1  
  
where 5 > (select count(distinct sid) from APPLIES A2 where A2.college = A1.college)  
  
order by college;
```

--or

```
select distinct college  
  
from APPLIES  
  
where college in (select college  
  
from APPLIES  
  
group by college  
  
having count(distinct sid)<5);
```

23. Show me the majors whose applicant's maximum mark is below the average.

```
select A.major  
from STUDENTS S, APPLIES A  
where S.id = A.sid  
group by major  
having max(mark) < (select avg(mark) from STUDENTS);
```

24. Show me IDs, names, and surnames of students applying to CS. Sort the results by student surname, student name and student id.

```
select id, name, surname  
from STUDENTS  
where id in (select sid from APPLIES where major = 'CS')  
order by surname, name, id
```

or

```
select distinct S.id, S.name, S.surname  
from STUDENTS S, APPLIES A  
where S.id = A.sid and major = 'CS'  
order by S.surname, S.name, S.id;
```

25. Show me the students who applied to CS but not EE. Sort the results by student surname, student name and student id.

```
select id, name, surname
from STUDENTS
where id in (select sid from APPLIES where major = 'CS')
and id not in (select sid from APPLIES where major = 'EE')
order by surname, name, id;
```

--or

```
select distinct S.id,S.name,S.surname
from STUDENTS S,APPLIES A
where S.id=A.sid
and A.major="CS"
and A.sid not in(select sid from APPLIES where major="EE")
order by surname,name,id;
```

26. Show me the colleges such that some other college is in the same state. Sort the results by state and college. It's mandatory to use EXISTS.

```
select name, state
from COLLEGES C1
where exists (select * from COLLEGES C2
              where C2.state = C1.state and C2.name <> C1.name)
order by state, name;
```

--or without exists

```
select name,state
```

```
from COLLEGES

WHERE state in (select state

                from COLLEGES

                group by state

                having count(name)>1);
```

27. Show me the name of the biggest college.

```
select C1.name

from COLLEGES C1

where not exists (select * from COLLEGES C2

                  where C2.enrollment > C1.enrollment);
```

28. Show me the student (or students) with highest mark (using EXISTS).

```
SELECT S1.name, S1.surname, S1.mark

FROM STUDENTS S1

WHERE NOT

EXISTS (
```

```
SELECT *

FROM STUDENTS S2

WHERE S2.mark > S1.mark

)
```

29. Show me the student with highest mark (using "where mark >= all")

```
select S1.name, S1.surname, S1.mark
from STUDENTS S1
where mark >= all (select mark from STUDENTS S2
                  where S2.id <> S1.id);
```

```
--or
```

```
select name,surname,mark
from STUDENTS
where mark>=all(select mark
from STUDENTS);
```

30. Show me the name of the college with the higher enrollment than all other colleges (using 'where enrollment > all').

```
select name
from COLLEGES C1
where enrollment > all (select enrollment from COLLEGES C2
                        where C2.name <> C1.name);
```

31. Show me the name of the college with the higher enrollment than all other colleges (using 'not enrollment <= Any')

[illegible]

32. Show me the students not from the smallest high school.

```
select id, name, surname, size_high_school
from STUDENTS S
where size_high_school > any (select size_high_school from STUDENTS)
order by surname, name, id
```

33. Show me the application information order by student surname and name.

```
select S.id, S.name, S.surname, A.college, A.major, A.decision
from STUDENTS S, COLLEGES C, APPLIES A
where A.sid = S.id and A.college = C.name
order by S.surname, S.name
```

34. Show me the pairs of students with same mark (ordered by its marks descendent and surnames/names ascendent).

First approach:

```
select S1.id, S1.name, S1.surname, S1.mark, S2.id, S2.name, S2.surname, S2.mark
from STUDENTS S1, STUDENTS S2
where S1.mark = S2.mark
order by S1.mark desc, S1.surname, S1.name, S2.surname, S2.name;
```

Get rid of self-pairings:

```
select S1.id, S1.name, S1.surname, S1.mark, S2.id, S2.name, S2.surname, S2.mark
from STUDENTS S1, STUDENTS S2
where S1.mark = S2.mark and
S1.id <> S2.id
order by S1.mark desc, S1.surname, S1.name, S2.surname, S2.name;
```

Get rid of reverse-pairings:

```
select S1.id, S1.name, S1.surname, S1.mark, S2.id, S2.name, S2.surname, S2.mark
from STUDENTS S1, STUDENTS S2
where S1.mark = S2.mark and
S1.id <> S2.id and
S1.id < S2.id
order by S1.mark desc, S1.surname, S1.name, S2.surname, S2.name;
```

35. Show me a list of college names and student names together ordered by name.

```
select * from
(select name as cName from COLLEGES
union
select concat(name, ' ', surname) as cName from STUDENTS) as S
order by S.cName;
```

36. Show me IDs of students who applied to both CS and EE.

```
select distinct sid from APPLIES
where major = 'CS'
```

and sid in

(select sid from APPLIES where major = 'EE');

OR

select distinct A1.sid

from APPLIES A1, APPLIES A2

where A1.sid = A2.sid and A1.major = 'CS' and

A2.major = 'EE';

37. Show me IDs of students who applied to CS but not EE.

select distinct sid from APPLIES

where major = 'CS'

and sid not in

(select sid from APPLIES where major = 'EE');

38. Show me the students that don't belong to the smallest high school (use exists).

select id, name, surname, size_high_school

from STUDENTS S1

where exists (select * from STUDENTS S2

where S2.size_high_school < S1.size_high_school);

39. Show me id of the students who applied to CS but not EE (use 'any' and two subqueries).


```
select id
from STUDENTS
where id = any (select sid from APPLIES where major = 'CS')
and not id = any (select sid from APPLIES where major = 'EE');
```

40. Insert a new college with name 'UIB', in the state 'IB', and with size 11500.

```
insert into COLLEGIES values ('UIB', 'IB', 11500);
```

41. Insert into APPLIES with college 'UIB', major 'IB', AND DECISION NULL all students who didn't apply anywhere. (Clue: insert with subselect).

First see who will be inserted (STUDENTS NOT INSIDE APPLIES):

```
select *
from STUDENTS
where ID not in (select sid from APPLIES);
```

Then insert them:

```
insert into APPLIES
select id, 'UIB', 'IB', null
from STUDENTS
where id not in (select sid from APPLIES);
```

**42. Admit to UIB EE all students who were refused (decision=false) in EE elsewhere.
(Clue: insert with subselect).**

First find who will be inserted:

```
select sid from APPLIES  
        where major = 'EE' and decision = false;
```

Then insert them:

```
insert into APPLIES  
        select id, 'UIB', 'EE', true  
        from STUDENTS  
        where id in (select sid from APPLIES  
                    where major = 'EE' and decision = false);
```