```
/*
      Assignment 2a: More SQL: Aggregates
      Statement :  Intermediate SQL: Aggregates and grouping and ordering
*/

/*
      Query 1:
      Find the number of instructors who have never taught any course.
      If the result of your query is empty, add appropriate data
      (and include corresponding insert statements) to ensure the
      result is not empty.
*/

select count(instructor.ID) as count_no_course
from instructor left join teaches on instructor.ID = teaches.ID
where teaches.ID is NULL
;

/*
      OUTPUT:-
      +-----------------+
      | count_no_course |
      +-----------------+
      |               3 |
      +-----------------+
      1 row in set (0.00 sec)
*/

/*
      Query 2:
      Find the total capacity of every building in the university.
*/

select x.building, sum(capacity)
from (select distinct department.building
      from department) as x left join classroom on x.building = classroom.building
group by x.building
;

/*
      OUTPUT:-
      +----------+---------------+
      | building | sum(capacity) |
      +----------+---------------+
      | Packard  |           500 |
      | Painter  |            10 |
      | Taylor   |            70 |
      | Watson   |            80 |
      +----------+---------------+
      4 rows in set (0.00 sec)
*/

/*
      Query 3:
      Find the maximum number of teachers for any single course section.
      Your output should be a single number.  For example if CS-101 section 1 in
      Spring 2012 had 3 instructors teaching the course, and no other section had
      more instructors teaching the section, your answer would be 3.
*/


select max(x.ins_count) max_ins_count
from
       (select count(ID) as ins_count
        from section as s left join teaches as t
        on s.course_id = t.course_id and s.sec_id = t.sec_id and s.semester = t.semester and s.year =
t.year
        group by s.course_id, s.sec_id, s.semester, s.year) as x
;
```

```
/*
      OUTPUT:-
      +---------------+
      | max_ins_count |
      +---------------+
      |             2 |
      +---------------+
      1 row in set (0.00 sec)
*/


/*
      Query 4:
      Find all departments that have at least one instructor, and list the
      names of the departments along with the number of instructors;
      order the result in descending order of number of instructors.
*/

select dept_name, count(ID) as ins_count
from instructor group by dept_name
order by ins_count desc
;

/*
      OUTPUT:-
      +------------+-----------+
      | dept_name  | ins_count |
      +------------+-----------+
      | Comp. Sci. |         3 |
      | Finance    |         2 |
      | History    |         2 |
      | Physics    |         2 |
      | Elec. Eng. |         2 |
      | Music      |         1 |
      | Biology    |         1 |
      +------------+-----------+
      7 rows in set (0.00 sec)
*/

/*
      Query 5:
      As in the previous question, but this time you should include departments
      even if they do not have any instructor,with the count as 0.
*/

select department.dept_name, count(ID) as ins_count
from department left join instructor on department.dept_name = instructor.dept_name group by
department.dept_name
order by ins_count desc
;

/*
      OUTPUT:-
      +------------+-----------+
      | dept_name  | ins_count |
      +------------+-----------+
      | Comp. Sci. |         3 |
      | Finance    |         2 |
      | History    |         2 |
      | Physics    |         2 |
      | Elec. Eng. |         2 |
      | Music      |         1 |
      | Biology    |         1 |
      +------------+-----------+
      7 rows in set (0.00 sec)
*/
```

```
/*
      Query 6:
      For each student, compute the total credits they have successfully completed,
      i.e. total credits of courses they have taken, for which they have a non-null
      grade other than 'F'. Do NOT use the tot_creids attribute of student.
*/

select student.ID, ifnull(sum(total_credits), 0) as total
from student left join (select ID, sum(credits) as total_credits
                        from takes natural join course
                        where grade is not NULL and grade != 'F'
                        group by ID) as x on student.ID = x.ID
group by student.ID
;

/*
      OUTPUT:-
      +-------+-------+
      | ID    | total |
      +-------+-------+
      | 00128 |     7 |
      | 12345 |    14 |
      | 19991 |     3 |
      | 23121 |     3 |
      | 44553 |     4 |
      | 45678 |     7 |
      | 54321 |     8 |
      | 55739 |     3 |
      | 70557 |     0 |
      | 76543 |     7 |
      | 76653 |     3 |
      | 98765 |     7 |
      | 98988 |     4 |
      +-------+-------+
      13 rows in set (0.00 sec)
*/

/*
      Query 7:
      Find the number of students who have been taught (at any time) by an
      instructor named 'Srinivasan'. Make sure you count a student only once
      even if the student has taken more than one course from Srinivasan.
*/

select count(distinct takes.ID) as total_students
from
      (select course_id, sec_id, semester, year
       from teaches natural join instructor
       where name = 'Srinivasan') as x natural join takes
;

/*
      OUTPUT:-
      +----------------+
      | total_students |
      +----------------+
      |              6 |
      +----------------+
      1 row in set (0.00 sec)
*/
```

```
/*----------------------------OPTIONAL-----------------------------------------------*/

/*
      Query 1:
      Find the name of all instructors who get the highest salary in their department.
*/

select name
from instructor join (select max(salary) as maxi
                      from instructor
                      group by dept_name) as m on instructor.salary = m.maxi
order by dept_name;

/*
      OUTPUT:-
      +-----------+
      | name      |
      +-----------+
      | Crick     |
      | Brandt    |
      | success   |
      | Kim       |
      | Singh     |
      | Wu        |
      | Califieri |
      | Mozart    |
      | Einstein  |
      +-----------+
      9 rows in set (0.00 sec)
*/
/*
      Query 2:
      Find all students who have taken all courses taken by instructor 'Srinivasan'.
      (This is the division operation of relational algebra) You can implement it by
      counting the number of courses taught by Srinivasan, and for each student
      (i.e. group by student), find the   number of courses taken by that student,
      which were taught by Srinivasan. Make sure to count each course ID only once.
*/

select name, ID
from student natural join
(select distinct ID
 from (select ID, course_id
       from (select ID, cnt
             from (select ID, count(distinct course_id) as cnt
                   from takes
                   group by ID) as x
             where cnt >= (select count(distinct course_id)
                           from teaches
                           where ID = (select ID
                                       from instructor
                                       where name = 'Srinivasan')
                           group by ID)) as y natural join takes) as xy natural join (select distinct
course_id
                                                                                      from teaches
                                                                                      where ID =
(select ID
                                                                                                 from
instructor
                                                                                                 where
name = 'Srinivasan')) as z) a;
/*
      OUTPUT:-
      +---------+-------+
      | name    | ID    |
      +---------+-------+
      | Shankar | 12345 |
      +---------+-------+
      1 row in set (0.00 sec)
*/
```

```
/*
      Query 3:
      Find the total money spent by each department for salaries of instructors of that department.
*/

select department.dept_name, ifnull(sum(salary), 0) as total_salary
from department left join instructor on department.dept_name = instructor.dept_name
group by department.dept_name;

/*
      Query 4:
      Find the names of all students whose advisor has taught the maximum number of courses
      (multiple offerings of a course count as only1).
      (Note: this is a complex query, break it into parts by using the with clause.)
*/

select name
from (select s_id
      from advisor natural join (select i_id
                                 from (select i_id, count(distinct course_id) as cnt
                                       from advisor left join teaches on i_id = ID
                                       group by i_id) as a
                                       where cnt =  (select max(cnt)
                                                     from (select ID, count(distinct course_id) as cnt
                                                           from advisor join teaches on i_id = ID
                                                           group by ID) as b)) as w) as q join student
on s_id = ID;

/*
      OUTPUT:-
      +---------+
      | name    |
      +---------+
      | Shankar |
      +---------+
      1 row in set (0.00 sec)
*/
```