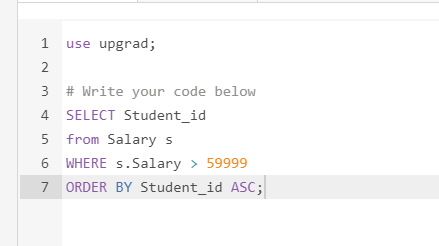
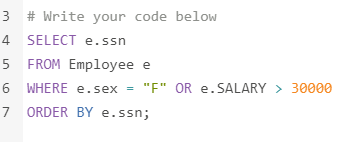
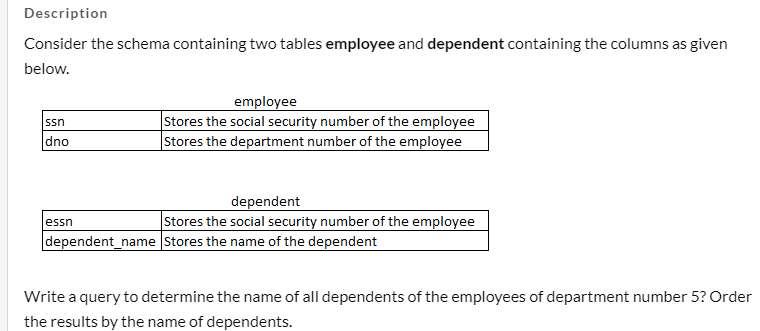
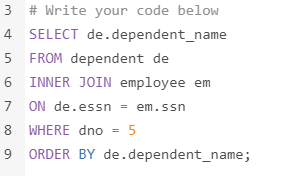
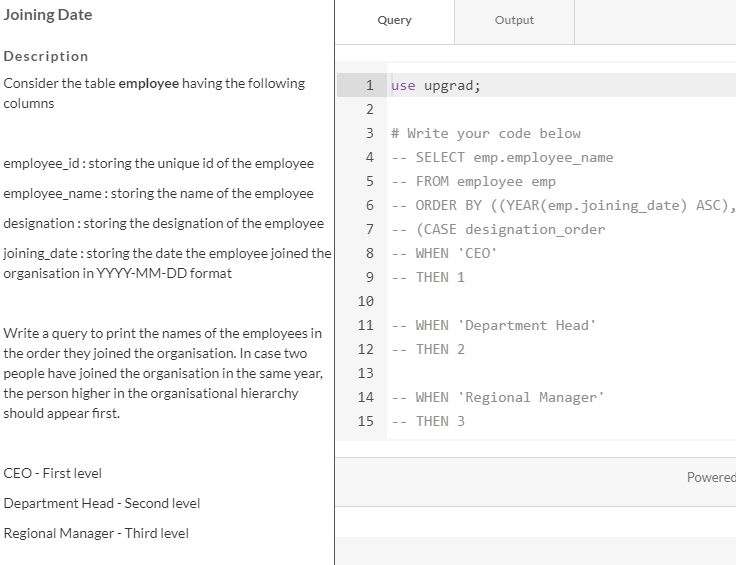
Given the table **salary**from a university database, write a query to find the student\_ids of students with salary greater than 59,999. Order the students in ascending order of student\_id. 

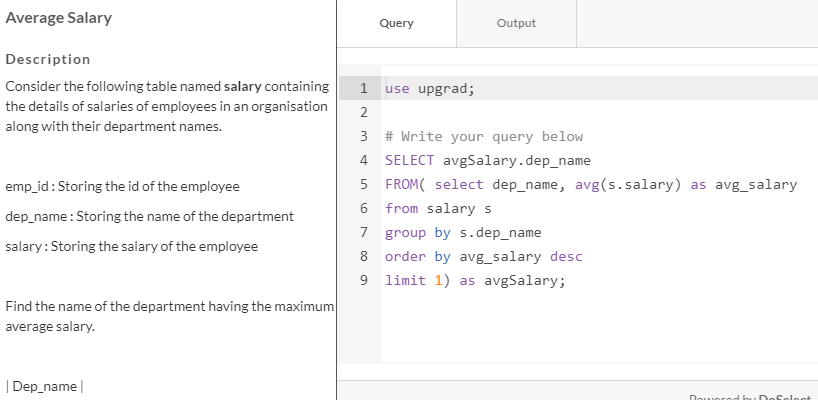
Write a query to find the social security numbers of all employees who are either female or have salary greater than 30000. Order the results on the basis of social security number in ascending order. Please note that the gender is denoted by either F or M.

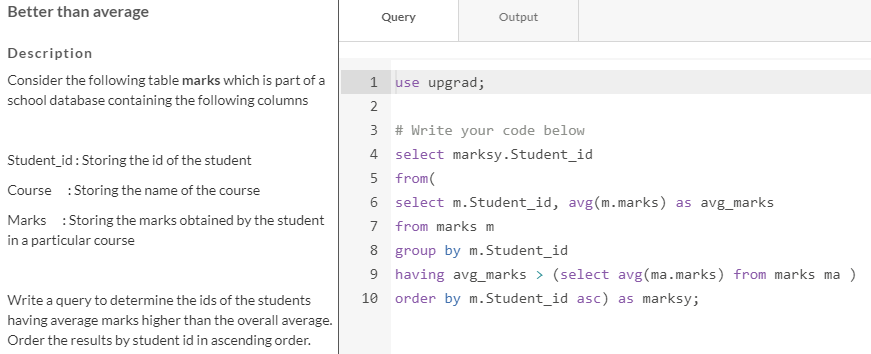


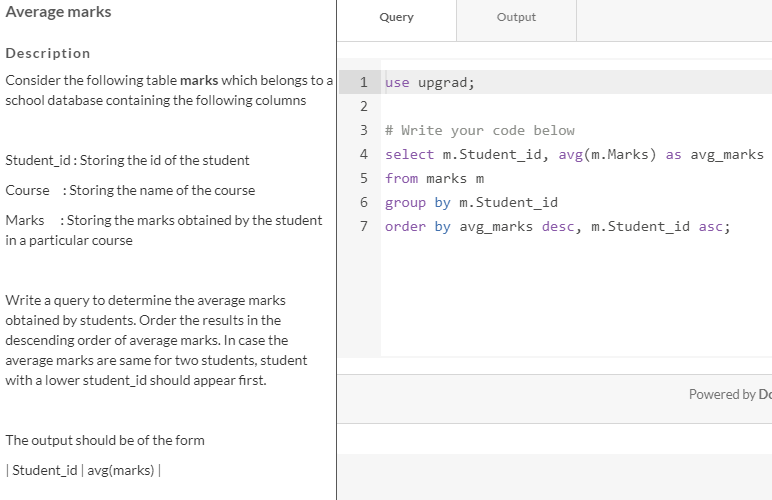












**Spread**

**Description**

Consider the following table **marks**which is part of a school database containing the following columns

Student\_id : Storing the id of the student

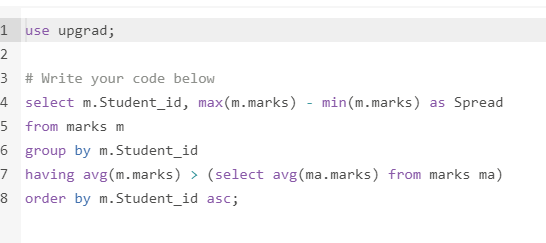
Course     : Storing the name of the course

Marks      : Storing the marks obtained by the student in the particular course

Write a query to determine the spread of the marks of the student having average marks greater than the overall average. Alias the column as **Spread**. Spread is defined as the difference between the highest and lowest marks obtained by the student. Order the output in order of student id.

The output should be as below

| Student\_id | Spread |



**Best teacher**

**Description**

Given the table **marks**containing the details of marks obtained by students containing the following columns

Student\_id : Storing the id of the student

Course      : Storing the name of the course

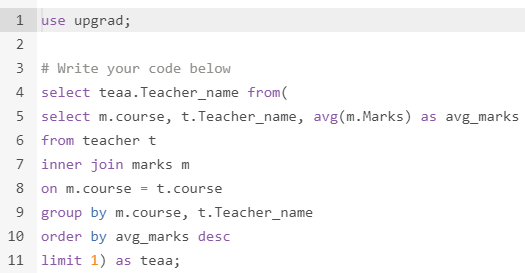
Marks       : Storing the marks obtained by the student in the particular course

and the table **teacher**containing the details of the teacher with the following columns

Teacher\_name : Storing the name of the teacher

Course           : Storing the course taught by the teacher

Write a query to find the name of the teacher who taught the course with the highest average.



**Most assessments**

**Description**

Rahul asked his students to create some mathematics problems. Write a query to print the student\_id, student\_name, and the total number of problems created by each student. Sort your results by the total number of problems in descending order. Alias the number of problems as **Number\_of\_problems**.

If more than one student created the same number of challenges, then sort the result by student\_id. If more than one student created the same number of challenges and the count is less than the maximum number of challenges created, then exclude those students from the result.

Table Student

student\_id      : Unique id for the student

student\_name : Name of the student

Table Problem

student\_id  : id of the student who created the problem

problem\_id : unique id of the problem created by the student

Columns in output

| Student\_id | Student\_name | Number\_of\_problems |

ANSWER:

use upgrad;

# write your code below

select

s.student\_id,

s.student\_name,

count(Pr.problem\_id) as Number\_of\_problems

from

Student as S inner join Problem as Pr

on Pr.student\_id = s.student\_id

group by

s.student\_id, s.student\_name

having

Number\_of\_problems =

(select count(maxprob.problem\_id)

from Problem as maxprob

group by maxprob.student\_id

Order by count(\*) desc

limit 1)

OR

Number\_of\_problems not in

(

select count(uniqueprob.problem\_id)

from problem as uniqueprob

group by uniqueprob.problem\_id

having uniqueprob.problem\_id <> Pr.problem\_id

)

order by Number\_of\_problems desc, s.student\_id;

