**MAWLANA BHASHANI SCIENCE AND TECHNOLOGY UNIVERSITY**

SANTOSH, TANGAIL-1902



DEPARTMENT OF INFORMATION AND COMMUNICATION TECHNOLOGY

**Lab Report**

**Experiment no :** 04

**Experiment name :** Basic queries of MySQL.

**Course Title :** Database Management Systems Lab

**Course Code :** ICT-2108

|  |  |
| --- | --- |
| Submitted by | Submitted to |
| Name : Kuldip Saha Mugdha  ID : IT22018  2nd Year, 1st Semester  Session: 2021-2022  Dept. of ICT, MBSTU | Dr. Md. Abir Hossain  Associate Professor ,  DEPARTMENT OF INFORMATION AND COMMUNICATION TECHNOLOGY  MAWLANA BHASHANI SCIENCE AND TECHNOLOGY UNIVERSITY |

**Date of performance : 19/3/2024**

**Date of submission : 25/3/2024**

**Experiment No:** 04

**Experiment name:** Basic query of MySQL.

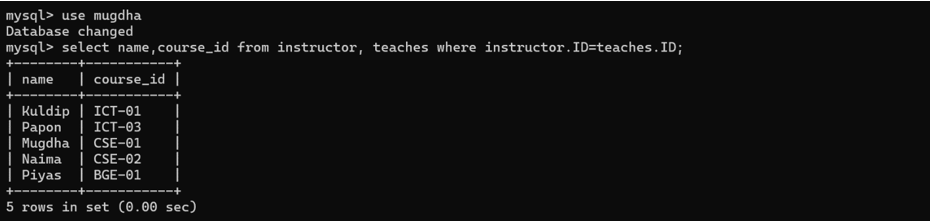
**Objectives:** The main objective of this experiment is to use some basis query of MySQL. We will learn how to show different attributes from database and also learn how to use query with different condition to find desire values.

**Required Instruments**:

1. MySQL 8.3 Command line Client.
2. Computer (At least 4GB RAM).

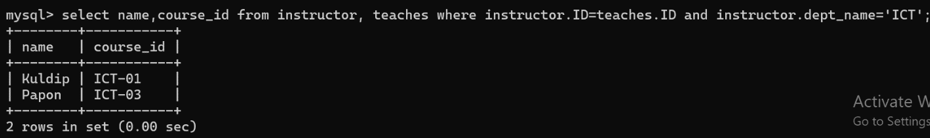
Showing the Cartesian product of the instructor relation with the teaches relation.

**SQL expression:** select name, course\_id from instructor, teaches where instructor.ID=teaches.ID;

****

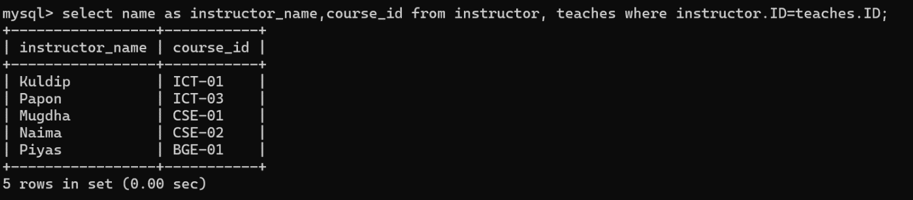
Finding only instructor names and course identifiers for instructors in the ICT department.

**SQL expression:** select name, course id from instructor, teaches where instructor.ID= teaches.ID and instructor.dept\_name = 'ICT';



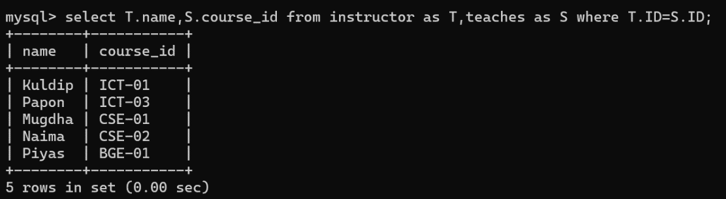
If we want the attribute name ‘*name*’ to be replaced with the name ‘instructor\_name’, we can rewrite the preceding query as:

**SQL expression:** select name as instructor\_name, course id from instructor, teaches where instructor.ID= teaches.ID;



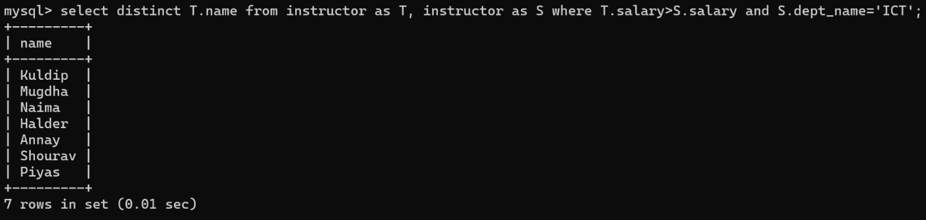
For all instructors in the university who have taught some course, finding their names and the course ID of all courses they taught.

**SQL expression:** select T.name, S.course\_id from instructor as T, teaches as S where T.ID= S.ID;



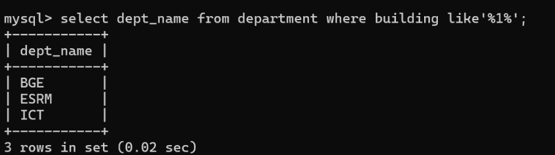
Finding the names of all instructors whose salary is greater than at least one instructor in the Physics department.

**SQL expression:** select distinct T.name from instructor as T, instructor as S where T.salary > S.salary and S.dept\_name = 'ICT';



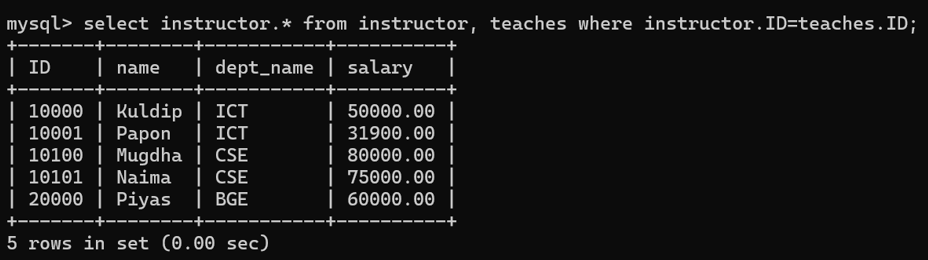
Finding the names of all departments whose building name includes the substring '1’.

**SQL expression:** select dept\_name from department where building like '%1%';



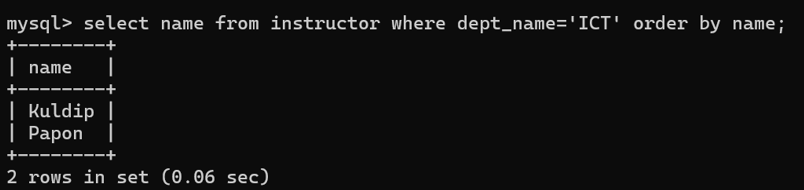
The asterisk symbol ‘\*’ can be used in the select clause to denote all attributes. Herer is and example:

**SQL expression:** select instructor.\* from instructor, teaches where instructor.ID= teaches.ID;



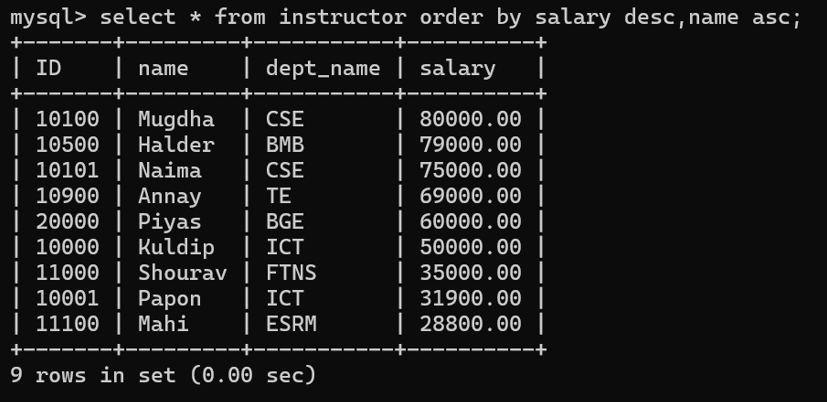
The order by clause causes the tuples in the result of a query to appear in sorted order. Here is an example:

**SQL expression:** select name from instructor where dept\_name = 'ICT' order by name;



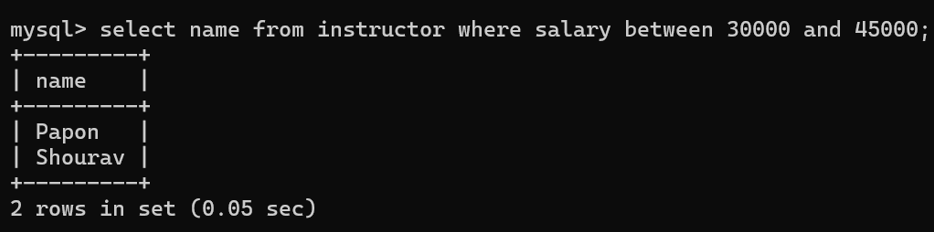
We wish to list the entire instructor relation in descending order of salary. If several instructors have the same salary, we order them in ascending order by name.

**SQL expression:** select \* from instructor order by salary desc, name asc;



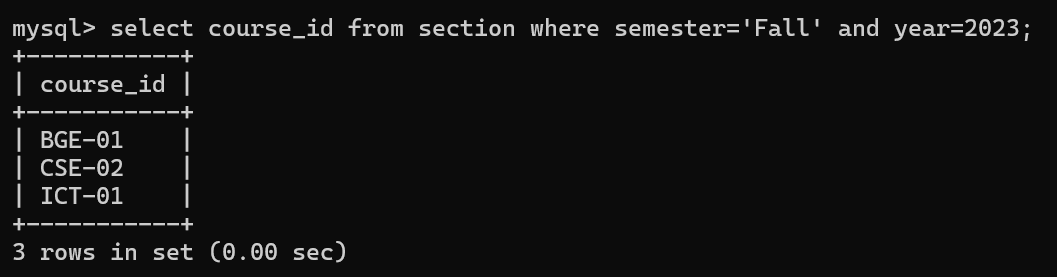
If we wish to find the names of instructors with salary amounts between 30,000 and 45,000. The query will be like this:

**SQL expression:** select name from instructor where salary between 30000 and 45000;



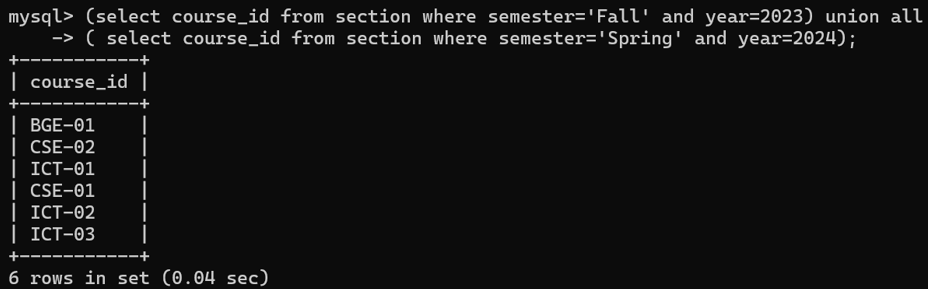
Finding the set of all courses taught in the Fall 2019 semester.

**SQL expression:** select course id from section where semester = ‘Fall' and year= 2023;



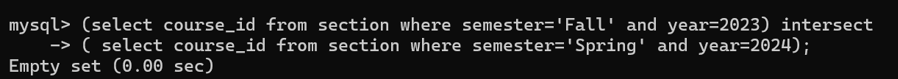
Finding the set of all courses taught either in Fall 2023 or in Spring 2024, or both.

**SQL expression:** (select course\_id from section where semester = 'Fall' and year= 2023) union all (select course\_id from section where semester = 'Spring' and year= 2024);



Finding the set of all courses taught in both the Fall 2017 and Summer 2018.

**SQL expression:** (select course\_id from section where semester = 'Fall' and year= 2023) intersect (select course\_id from section where semester = ‘Spring' and year= 2024);



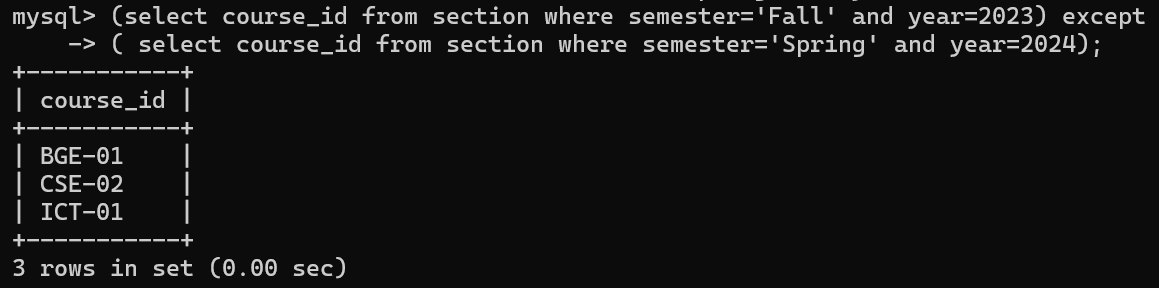
Finding all instructors who appear in the instructor relation with null values for salary.

**SQL expression:** select name from instructor where salary is null;



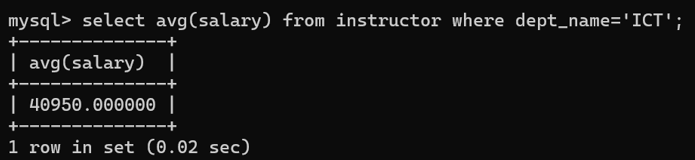
Finding all courses taught in the Fall 2023 semester but not in the Spring 2024 semester.

**SQL expression:** (select course\_id from section where semester = 'Fall' and year= 2023) except (select course\_id from section where semester = 'Spring' and year= 2024);



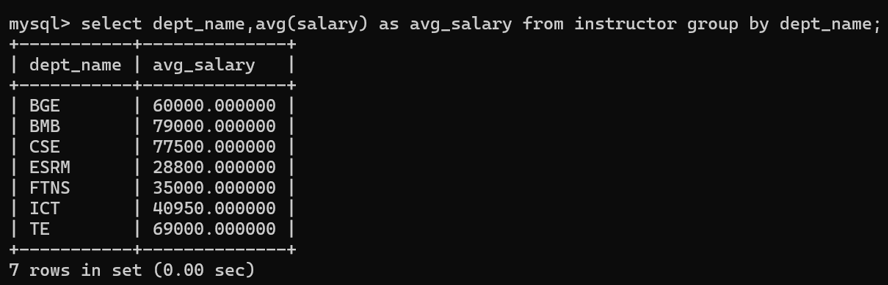
Finding the average salary of instructors in the ICT department.

**SQL expression:** select avg(salary) from instructor where dept\_name = 'ICT';



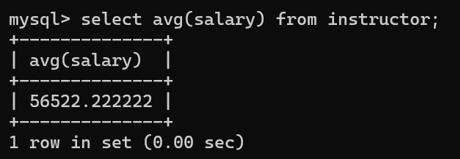
Finding the average salary in each department.

**SQL expression:** select dept\_name, avg(salary) as avg\_salary from instructor group by dept\_name;



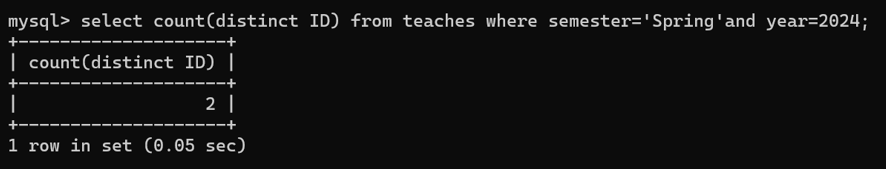
Find the average salary of all instructors.

**SQL expression:** select avg(salary) from instructor;



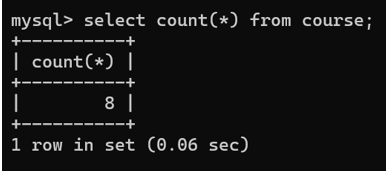
Finding how many teachers taught in Spring 2024 semester.

**SQL expression:** select count (distinct ID) from teaches where semester = 'Spring' and year = 2024;

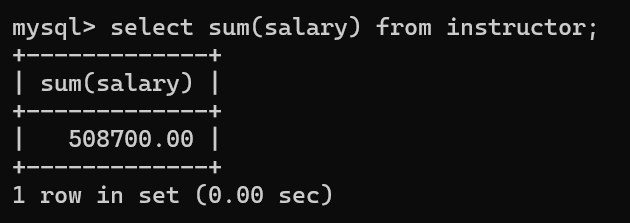


Finding the number of tuples in the course relation.

**SQL expression:** select count(\*) from course;

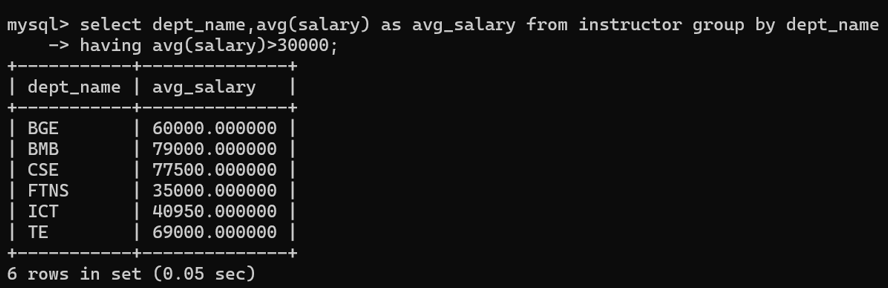


Finding the total salary.

**SQL expression:** select sum(salary) from instructor;

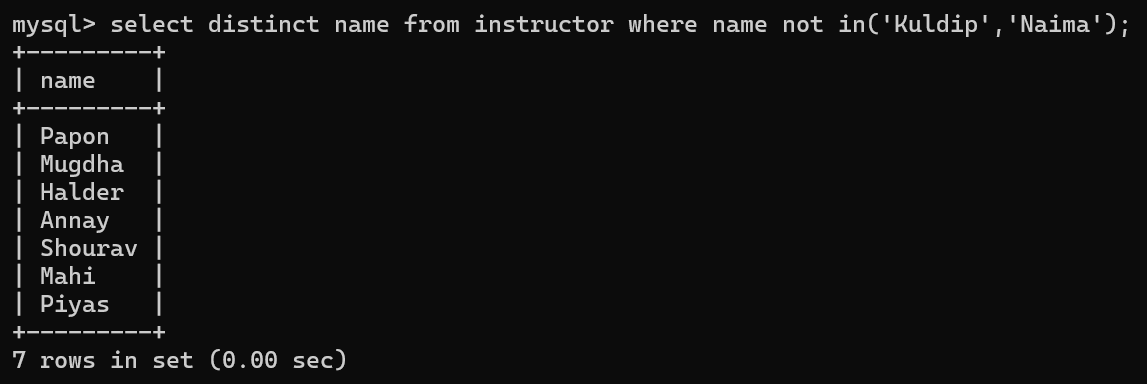
Finding the department name and average salary where the average salary of the instructors is more than 30,000.

**SQL expression:** select dept\_name,avg(salary) as avg\_salary from instructor group by dept\_name having avg(salary)>30000;



We wish to know the names of instructors whose names are neither “Kuldip” nor “Naima”.

**SQL expression:** select distinct name from instructor where name not in (‘Kuldip’, ‘Naima’);



**Discussion:** From the above query we can easily find our desired attributes. We can command different types of queries to find any type of data set that we are looking for.