DEPARTMENT OF MATHEMATICAL AND COMPUTATIONAL SCIENCES

NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA, SURATHKAL

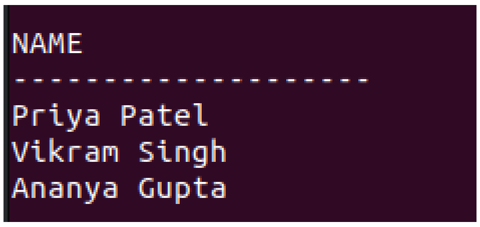
MA611 – 2nd Semester MCA 2024-2025
DATABASE SYSTEMS LAB

Assignment-2

NAME : Sunil Kumar Dalei
ROLL NO. : 244CA056

1. Retrieve the names of all instructors who teach at least one course in the 'Computer Science' department.

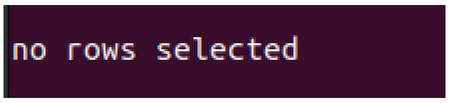
SELECT DISTINCT i.name FROM instructor i JOIN teaches t ON i.ID = t.ID JOIN course c ON t.course_id = c.course_id WHERE c.dept_name = 'Computer Science';



```
NAME
--------------------
Priya Patel
Vikram Singh
Ananya Gupta
```

2. List all students who have taken at least one course in 'Fall' and 'Spring' semesters in the same year.

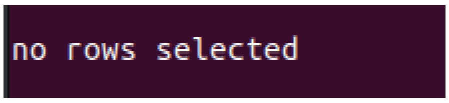
SELECT DISTINCT t1.ID, s.name FROM takes t1 JOIN takes t2 ON t1.ID = t2.ID AND t1.year = t2.year JOIN student s ON t1.ID = s.ID WHERE t1.semester = 'Fall' AND t2.semester = 'Spring';



```
no rows selected
```

3. Find all classrooms that have a capacity of more than 100 but are not assigned to any section.

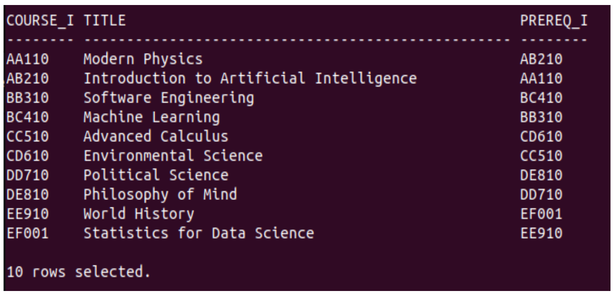
SELECT c.building, c.room_number, c.capacity FROM classroom c LEFT JOIN section s ON c.building = s.building AND c.room_number = s.room_number WHERE c.capacity > 100 AND s.building IS NULL;



```
no rows selected
```

4 Display all courses along with their prerequisites, including courses that have no prerequisites.

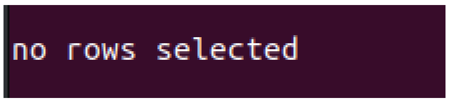
SELECT c.course_id, c.title, p.prereq_id FROM course c LEFT JOIN prereq p ON c.course_id = p.course_id ORDER BY c.course_id;



```
COURSE_I TITLE                                            PREREQ_I
-------- ------------------------------------------------ --------
AA110    Modern Physics                                   AB210
AB210    Introduction to Artificial Intelligence          AA110
BB310    Software Engineering                             BC410
BC410    Machine Learning                                 BB310
CC510    Advanced Calculus                                CD610
CD610    Environmental Science                            CC510
DD710    Political Science                                DE810
DE810    Philosophy of Mind                               DD710
EE910    World History                                    EF001
EF001    Statistics for Data Science                      EE910

10 rows selected.
```

5. Retrieve the list of instructors who have never taught a course.

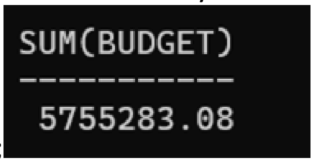
SELECT i.ID, i.name FROM instructor i LEFT JOIN teaches t ON i.ID = t.ID WHERE t.ID IS NULL;



```
no rows selected
```

6. Find the total budget allocated to all departments that have at least one instructor earning more than ₹100,00.

SELECT SUM(d.budget) AS total_budget FROM department d WHERE d.dept_name IN
(
    SELECT DISTINCT i.dept_name
    FROM instructor i
    WHERE i.salary > 100000

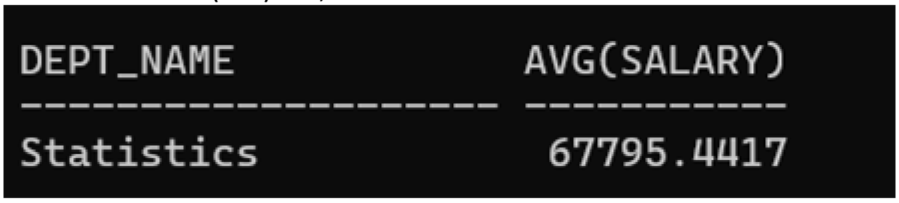


```
SUM(BUDGET)
-----------
 5755283.08
```

);

7. Find the average salary of instructors grouped by department but only include departments with more than 5 instructors.

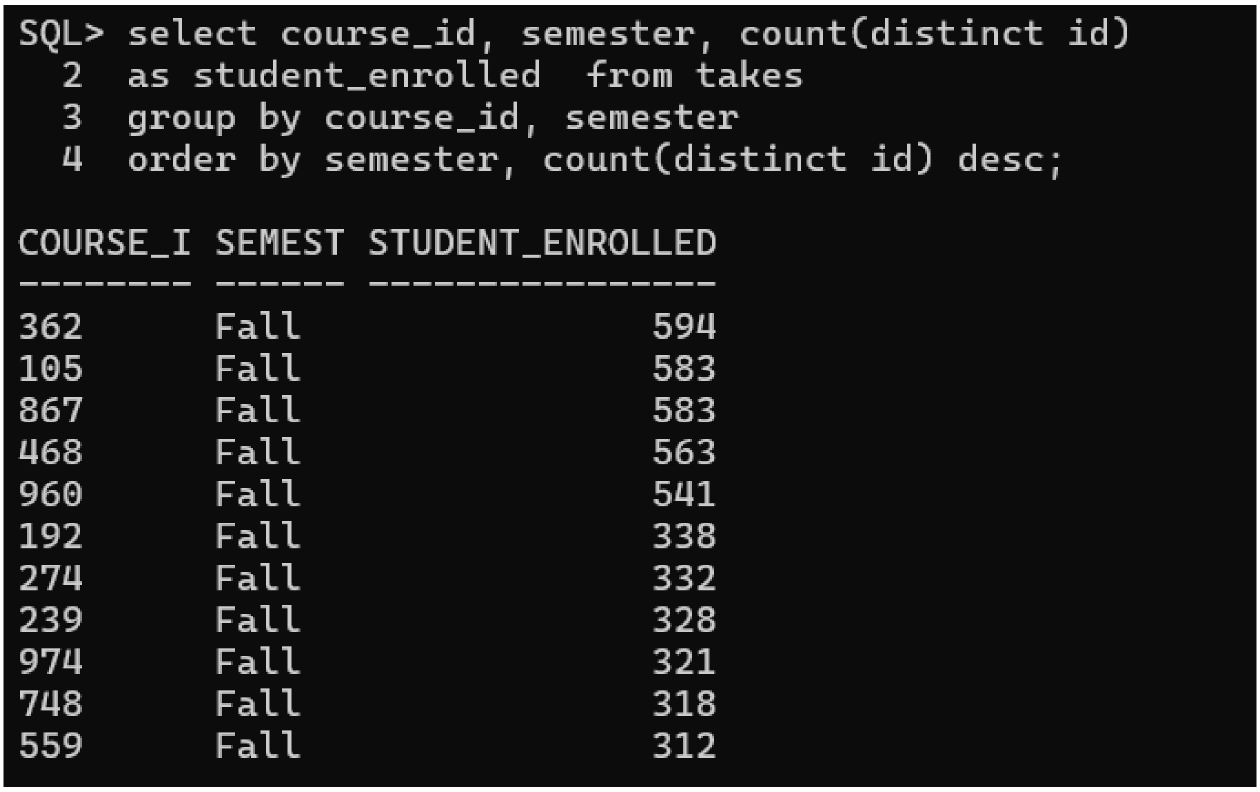
SELECT i.dept_name, AVG(i.salary) AS average_salary
FROM instructor i
GROUP BY i.dept_name
HAVING COUNT(i.ID) > 5;



```
DEPT_NAME            AVG(SALARY)
-------------------- -----------
Statistics            67795.4417
```

8. Find the total number of students enrolled in each course for every semester, and sort by semester and number of students (descending).

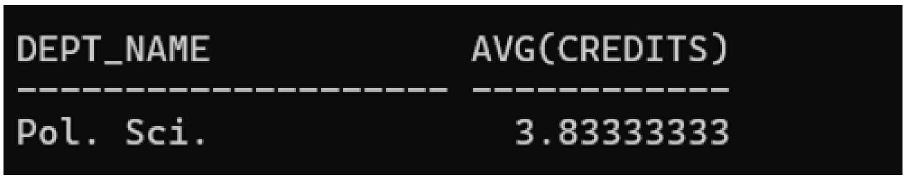
SELECT t.course_id, t.semester, t.year, COUNT(t.ID) AS num_students FROM takes t GROUP BY t.course_id, t.semester, t.year ORDER BY t.semester, num_students DESC;



```
SQL> select course_id, semester, count(distinct id)
  2  as student_enrolled  from takes
  3  group by course_id, semester
  4  order by semester, count(distinct id) desc;

COURSE_I SEMEST STUDENT_ENROLLED
-------- ------ ----------------
362      Fall                594
105      Fall                583
867      Fall                583
468      Fall                563
960      Fall                541
192      Fall                338
274      Fall                332
239      Fall                328
974      Fall                321
748      Fall                318
559      Fall                312
```

9. Determine which department has the highest average course credit.

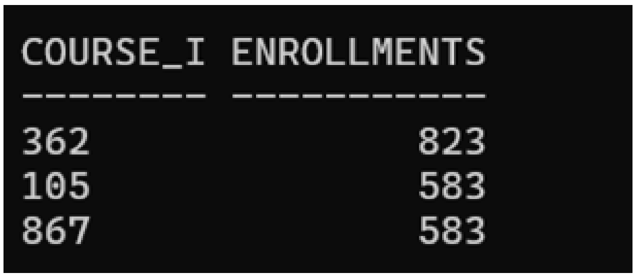
SELECT dept_name, avg(credits) from course group by dept_name having avg(credits) >= all (select avg(credits) from course group by dept_name);



```
DEPT_NAME            AVG(CREDITS)
-------------------- ------------
Pol. Sci.              3.83333333
```

10. Find the top 3 courses with the most students enrolled across all semesters.

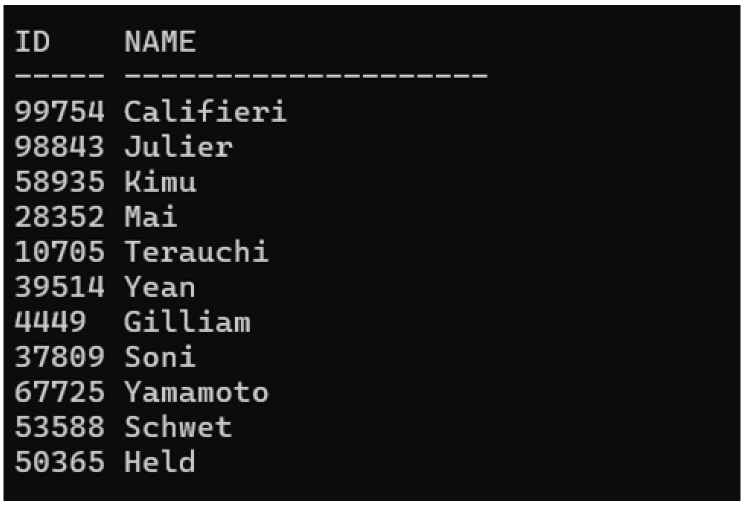
SELECT course_id, enrollments from (select course_id, count(distinct id) as enrollments from takes group by course_id order by count(distinct id) desc) where rownum <= 3;



```
COURSE_I ENROLLMENTS
-------- -----------
362              823
105              583
867              583
```

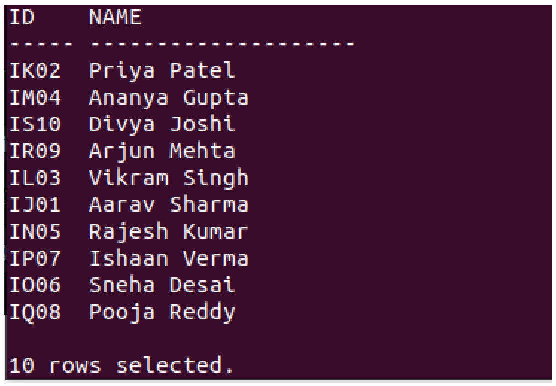
11. Find all students who have taken every course taught by the instructor 'John Doe'.

SELECT id, name from takes natural join student where course_id in (select course_id from teaches natural join instructor where name='John Doe') group by id, name having count(distinct course_id) = (select count(distinct course_id) from teaches natural join instructor where name = 'John Doe');


```
ID    NAME
----- --------------------
99754 Califieri
98843 Julier
58935 Kimu
28352 Mai
10705 Terauchi
39514 Yean
4449  Gilliam
37809 Soni
67725 Yamamoto
53588 Schwet
50365 Held
```

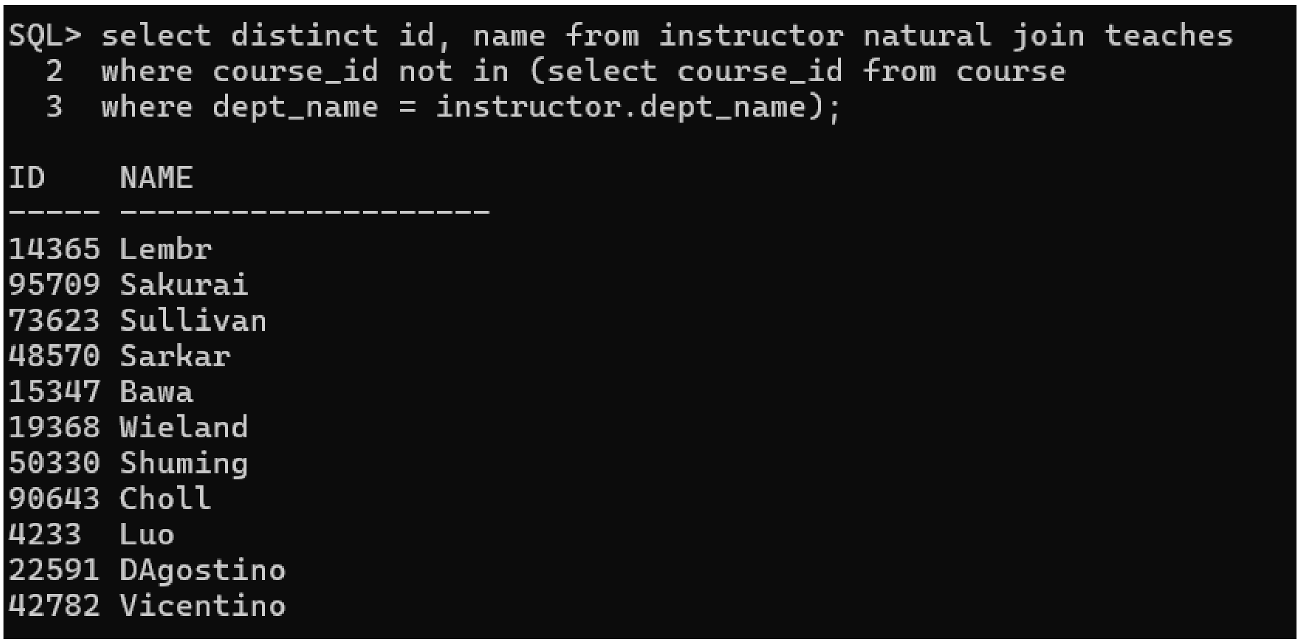
12. Retrieve the names of students who have the same name as their advisor.

SELECT s.name FROM student s JOIN advisor a ON s.ID = a.s_ID JOIN instructor i ON a.i_ID = i.ID WHERE s.name = i.name;

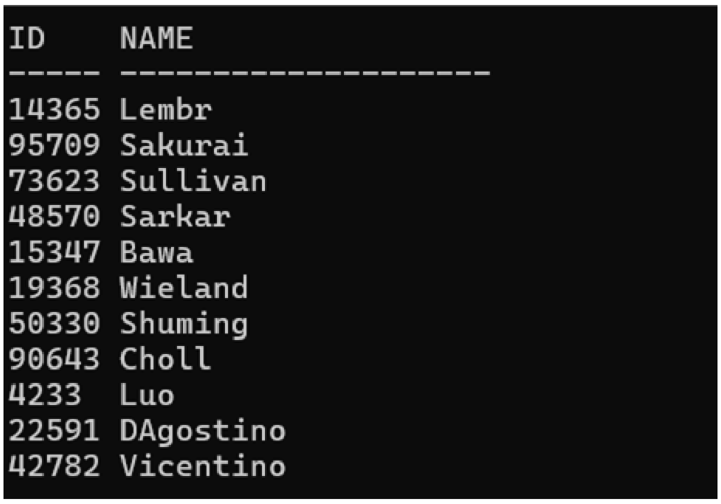

```
ID    NAME
----- --------------------
IK02  Priya Patel
IM04  Ananya Gupta
IS10  Divya Joshi
IR09  Arjun Mehta
IL03  Vikram Singh
IJ01  Aarav Sharma
IN05  Rajesh Kumar
IP07  Ishaan Verma
IO06  Sneha Desai
IQ08  Pooja Reddy

10 rows selected.
```

13. Find all instructors who have taught at least one course that they did not belong to the department of.

SELECT DISTINCT i.ID, i.name FROM instructor i JOIN teaches t ON i.ID = t.ID JOIN course c ON t.course_id = c.course_id WHERE i.dept_name != c.dept_name;



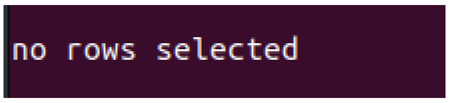
```
SQL> select distinct id, name from instructor natural join teaches
  2  where course_id not in (select course_id from course
  3  where dept_name = instructor.dept_name);

ID    NAME
----- --------------------
14365 Lembr
95709 Sakurai
73623 Sullivan
48570 Sarkar
15347 Bawa
19368 Wieland
50330 Shuming
90643 Choll
4233  Luo
22591 DAgostino
42782 Vicentino
```



```
ID    NAME
----- --------------------
14365 Lembr
95709 Sakurai
73623 Sullivan
48570 Sarkar
15347 Bawa
19368 Wieland
50330 Shuming
90643 Choll
4233  Luo
22591 DAgostino
42782 Vicentino
```

14. List all students who have taken a course in a classroom that has a capacity less than the number of students enrolled.

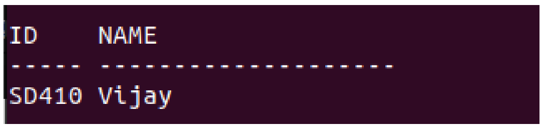
SELECT DISTINCT s.ID, s.name FROM student JOIN takes t ON s.ID = t.ID JOIN section sec ON t.course_id = sec.course_id AND t.sec_id = sec.sec_id AND t.semester = sec.semester AND t.year = sec.year
JOIN classroom c ON sec.building = c.building AND sec.room_number = c.room_number
WHERE c.capacity < (SELECT COUNT(*) FROM takes WHERE course_id = t.course_id AND sec_id = t.sec_id AND semester = t.semester AND year = t.year);



```
no rows selected
```

15. Find students who have taken every course offered by their department.

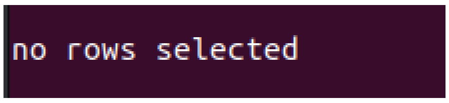
```
SELECT s.ID, s.name FROM student s
WHERE NOT EXISTS (
   SELECT c.course_id
   FROM course c
   WHERE c.dept_name = s.dept_name AND NOT EXISTS (
        SELECT 1
        FROM takes t
        WHERE t.ID = s.ID AND t.course_id = c.course_id
    )
);
```



```
ID    NAME
----- --------------------
SD410 Vijay
```

16. Find students who have taken a course but have not received a grade.

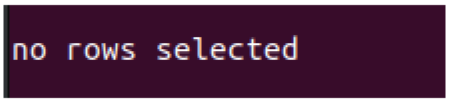
```
SELECT s.ID, s.name
FROM student s
JOIN takes t ON s.ID = t.ID
WHERE t.grade IS NULL OR t.grade = '';
```



```
no rows selected
```

18. Retrieve the details of instructors who have the exact same salary as another instructor.

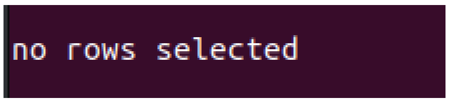
```
SELECT i1.ID, i1.name, i1.dept_name, i1.salary
FROM instructor i1
JOIN instructor i2 ON i1.salary = i2.salary AND i1.ID != i2.ID;
```



```
no rows selected
```

19. Identify all courses that have prerequisites, but the prerequisite itself has no prerequisite.

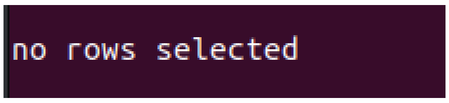
```
SELECT c.course_id, c.title
FROM course c
JOIN prereq p ON c.course_id = p.course_id
LEFT JOIN prereq p2 ON p.prereq_id = p2.course_id
WHERE p2.course_id IS NULL;
```



```
no rows selected
```

20. Find all students who have taken courses in every semester (Fall, Winter, Spring, Summer) at least once.

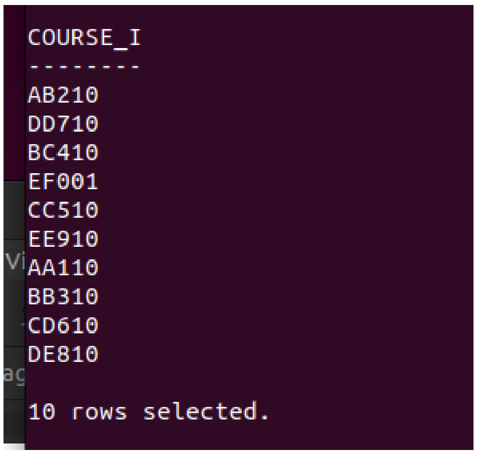
```
SELECT s.ID FROM student s

JOIN takes t ON s.ID = t.ID

GROUP BY s.ID

HAVING COUNT(DISTINCT t.semester) = 4;
```



```
no rows selected
```

22. List all courses that have at least two levels of prerequisites (i.e., a prerequisite has another prerequisite).

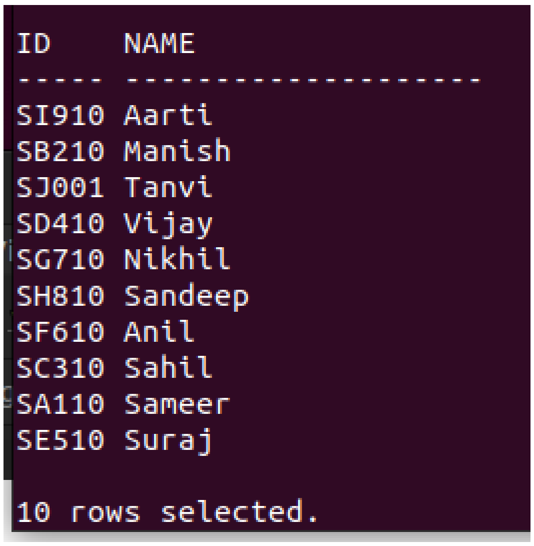
```
SELECT DISTINCT p1.course_id FROM prereq p1 JOIN prereq p2 ON p1.prereq_id = p2.course_id;
```



```
COURSE_I
--------
AB210
DD710
BC410
EF001
CC510
EE910
AA110
BB310
CD610
DE810

10 rows selected.
```

24. Identify students who have taken a course whose prerequisite they have never taken.

SELECT DISTINCT s.ID, s.name FROM student s JOIN takes t ON s.ID = t.ID JOIN prereq p ON t.course_id = p.course_id LEFT JOIN takes t2 ON s.ID = t2.ID AND t2.course_id = p.prereq_id WHERE t2.course_id IS NULL;

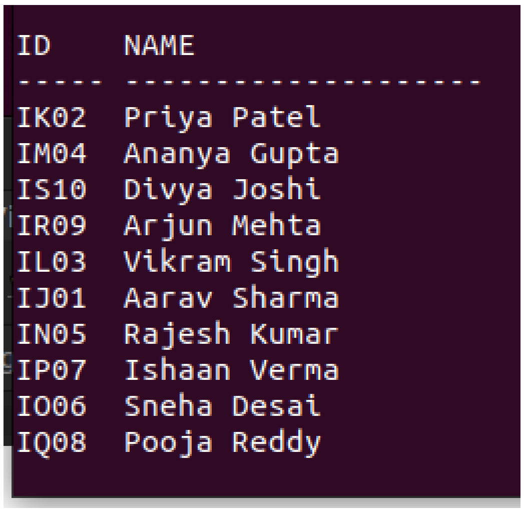


```
ID    NAME
----- --------------------
SI910 Aarti
SB210 Manish
SJ001 Tanvi
SD410 Vijay
SG710 Nikhil
SH810 Sandeep
SF610 Anil
SC310 Sahil
SA110 Sameer
SE510 Suraj

10 rows selected.
```

25. Find all instructors who have taught a course that has another course as a prerequisite.

SELECT DISTINCT i.ID, i.name
FROM instructor i
JOIN teaches t ON i.ID = t.ID

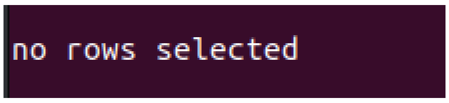
JOIN prereq p ON t.course_id = p.course_id;



```
ID    NAME
----- --------------------
IK02  Priya Patel
IM04  Ananya Gupta
IS10  Divya Joshi
IR09  Arjun Mehta
IL03  Vikram Singh
IJ01  Aarav Sharma
IN05  Rajesh Kumar
IP07  Ishaan Verma
IO06  Sneha Desai
IQ08  Pooja Reddy
```

26. Find all students who have the same total credits as another student in a different department.

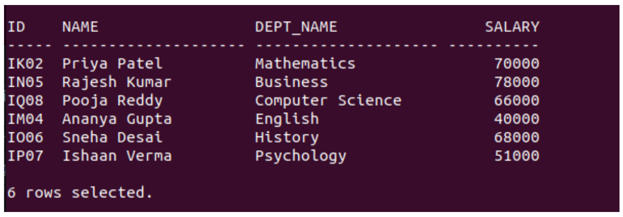
SELECT s1.ID, s1.name
FROM student s1
JOIN student s2 ON s1.tot_cred = s2.tot_cred AND s1.dept_name != s2.dept_name
WHERE s1.ID != s2.ID;



```
no rows selected
```

27. Identify instructors who have the highest salary in their department but still earn less than the highest salary in another department.

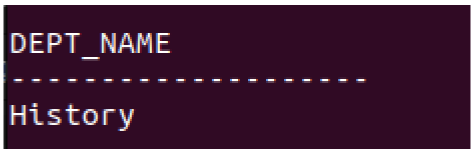
SELECT i1.ID, i1.name, i1.dept_name, i1.salary
FROM instructor i1
JOIN (
   SELECT dept_name, MAX(salary) AS max_salary
   FROM instructor
   GROUP BY dept_name
) i2 ON i1.salary = i2.max_salary
WHERE i1.salary < (SELECT MAX(salary) FROM instructor WHERE dept_name != i1.dept_name);



```
ID    NAME                 DEPT_NAME                SALARY
----- -------------------- -------------------- ----------
IK02  Priya Patel          Mathematics               70000
IN05  Rajesh Kumar         Business                  78000
IQ08  Pooja Reddy          Computer Science          66000
IM04  Ananya Gupta         English                   40000
IO06  Sneha Desai          History                   68000
IP07  Ishaan Verma         Psychology                51000

6 rows selected.
```

28. List all departments where the highest-paid instructor earns more than the total department budget divided by the number of instructors.

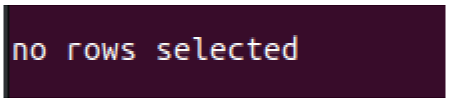
SELECT d.dept_name
FROM department d
JOIN instructor i ON d.dept_name = i.dept_name
GROUP BY d.dept_name
HAVING MAX(i.salary) > (SELECT SUM(budget) / COUNT(*) FROM instructor WHERE dept_name = d.dept_name);



```
DEPT_NAME
--------------------
History
```

29. Find courses that have never been taught in the same semester for consecutive years.

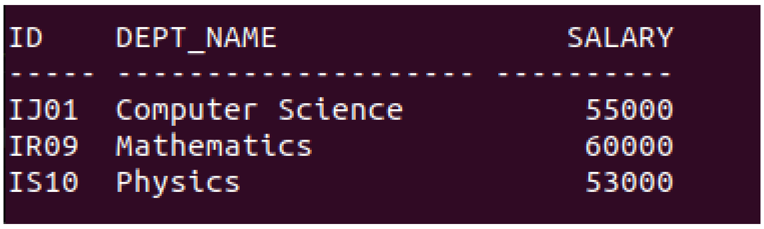
```
SELECT DISTINCT c.course_id
FROM course c
JOIN section s1 ON c.course_id = s1.course_id
JOIN section s2 ON c.course_id = s2.course_id
WHERE s1.semester = s2.semester AND s1.year = s2.year – 1;
```



```
no rows selected
```

30. Rank instructors by salary within their department and return only the second-highest salary in each department.

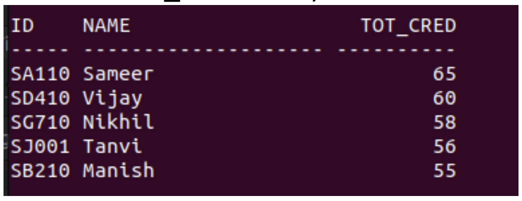
```
WITH RankedSalaries AS (
   SELECT ID, dept_name, salary, RANK() OVER (PARTITION BY dept_name ORDER BY salary DESC)
AS rank
   FROM instructor
)
SELECT ID, dept_name, salary
FROM RankedSalaries
WHERE rank = 2;
```



```
ID    DEPT_NAME                SALARY
----- -------------------- ----------
IJ01  Computer Science          55000
IR09  Mathematics               60000
IS10  Physics                   53000
```

31. Retrieve the top 5 students with the highest total credits, including ties.

```
WITH RankedStudents AS (
   SELECT s.ID, s.name, s.tot_cred, RANK() OVER (ORDER BY s.tot_cred DESC) AS rank
   FROM student s
)
SELECT ID, name, tot_cred
FROM RankedStudents
WHERE rank <= 5;
```



```
ID    NAME                   TOT_CRED
----- -------------------- ----------
SA110 Sameer                       65
SD410 Vijay                        60
SG710 Nikhil                       58
SJ001 Tanvi                        56
SB210 Manish                       55
```