Name Himanshu Shekhar Mumru

Roll No 244ca019

DBMS Assignment 2

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

SQL: 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 = 'Comp. Sci.';

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

SQL: SELECT DISTINCT S.name

FROM student S

JOIN takes T1 ON S.ID = T1.ID

JOIN takes T2 ON S.ID = T2.ID

WHERE T1.semester = 'Fall'

AND T2.semester = 'Spring'

AND T1.year = T2.year;

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

SQL: SELECT C.room\_number, C.building

FROM classroom C

LEFT JOIN section S ON C.room\_number = S.room\_number AND C.building = S.building

WHERE C.capacity > 100 AND S.room\_number IS NULL;

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

SQL: SELECT C.course\_id, C.title, P.prereq\_id

FROM course C

LEFT JOIN prereq P ON C.course\_id = P.course\_id;

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

SQL: SELECT I.name

FROM instructor I

LEFT JOIN teaches T ON I.ID = T.ID

WHERE T.course\_id IS NULL;

6. Find the total budget allocated to all departments that have at least one instructor earning more than ₹100,000.

SQL: 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

);

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

SQL: SELECT I.dept\_name, AVG(I.salary) AS avg\_salary

FROM instructor I

GROUP BY I.dept\_name

HAVING COUNT(I.ID) > 5;

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

SQL: SELECT T.course\_id, T.semester, T.year, COUNT(T.ID) AS total\_students

FROM takes T

GROUP BY T.course\_id, T.semester, T.year

ORDER BY T.semester, total\_students DESC;

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

SQL: SELECT dept\_name, AVG(credits) AS avg\_credits

FROM course

GROUP BY dept\_name

HAVING AVG(credits) = (

SELECT MAX(avg\_credits)

FROM (

SELECT dept\_name, AVG(credits) AS avg\_credits

FROM course

GROUP BY dept\_name

) AS dept\_avg

);

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

SQL: SELECT T.course\_id, COUNT(T.ID) AS total\_students

FROM takes T

GROUP BY T.course\_id

ORDER BY total\_students DESC

LIMIT 3;

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

SQL: SELECT S.name

FROM student S

WHERE NOT EXISTS (

SELECT C.course\_id

FROM teaches T

JOIN instructor I ON T.ID = I.ID

JOIN course C ON T.course\_id = C.course\_id

WHERE I.name = 'John Doe'

AND C.course\_id NOT IN (

SELECT T2.course\_id

FROM takes T2

WHERE T2.ID = S.ID

)

);

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

SQL: SELECT DISTINCT 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;

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

SQL: 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 I.dept\_name <> C.dept\_name;

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

SQL: SELECT S.name

FROM student S

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

JOIN classroom C ON Sec.room\_number = C.room\_number AND Sec.building = C.building

GROUP BY S.ID, S.name

HAVING COUNT(T.ID) > MAX(C.capacity);

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

SQL: SELECT 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 T.course\_id

FROM takes T

WHERE T.ID = S.ID

AND T.course\_id = C.course\_id

)

);

16. Identify the department(s) with the largest number of distinct courses offered.

SQL: SELECT dept\_name

FROM course

GROUP BY dept\_name

HAVING COUNT(DISTINCT course\_id) = (

SELECT MAX(course\_count)

FROM (

SELECT dept\_name, COUNT(DISTINCT course\_id) AS course\_count

FROM course

GROUP BY dept\_name

) AS dept\_course\_counts

);

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

SQL: SELECT DISTINCT S.name

FROM student S

JOIN takes T ON S.ID = T.ID

WHERE T.grade IS NULL;

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

SQL: SELECT I1.\*

FROM instructor I1

JOIN instructor I2

ON I1.salary = I2.salary

AND I1.ID <> I2.ID;

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

SQL: SELECT DISTINCT P.course\_id

FROM prereq P

WHERE P.prereq\_id NOT IN (SELECT course\_id FROM prereq);

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

SQL: SELECT S.name

FROM student S

JOIN takes T ON S.ID = T.ID

GROUP BY S.ID, S.name

HAVING COUNT(DISTINCT T.semester) = 4;

21. Find all prerequisite chains for a given course (i.e., if A is a prerequisite for B, and B is a prerequisite for C, return A → B → C).

SQL: SELECT P1.course\_id AS course, P1.prereq\_id AS direct\_prerequisite, P2.prereq\_id AS indirect\_prerequisite

FROM prereq P1

LEFT JOIN prereq P2 ON P1.prereq\_id = P2.course\_id

WHERE P1.course\_id = 'CS101' OR P2.course\_id = 'CS101';

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

SQL: SELECT DISTINCT P1.course\_id

FROM prereq P1

JOIN prereq P2 ON P1.prereq\_id = P2.course\_id;

23. Find the shortest prerequisite chain for any course that eventually leads to a specific course, say 'CS101'.

SQL: SELECT P1.prereq\_id AS level\_1,

P2.prereq\_id AS level\_2,

P3.prereq\_id AS level\_3

FROM prereq P1

LEFT JOIN prereq P2 ON P1.prereq\_id = P2.course\_id

LEFT JOIN prereq P3 ON P2.prereq\_id = P3.course\_id

WHERE P1.course\_id = 'CS101'

ORDER BY (CASE

WHEN P3.prereq\_id IS NOT NULL THEN 3

WHEN P2.prereq\_id IS NOT NULL THEN 2

ELSE 1

END)

LIMIT 1;

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

SQL: SELECT DISTINCT T.ID, S.name

FROM takes T

JOIN prereq P ON T.course\_id = P.course\_id

LEFT JOIN takes T2 ON T.ID = T2.ID AND T2.course\_id = P.prereq\_id

JOIN student S ON T.ID = S.ID

WHERE T2.ID IS NULL;

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

SQL: SELECT DISTINCT I.name

FROM instructor I

JOIN teaches T ON I.ID = T.ID

JOIN prereq P ON T.course\_id = P.course\_id;

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

SQL: SELECT S1.name, S1.dept\_name, S2.name AS other\_student, S2.dept\_name AS other\_dept

FROM student S1

JOIN student S2 ON S1.tot\_cred = S2.tot\_cred AND S1.dept\_name <> S2.dept\_name;

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

SQL: WITH DeptMaxSalaries AS (

SELECT dept\_name, MAX(salary) AS max\_salary

FROM instructor

GROUP BY dept\_name

)

SELECT I.name, I.salary, I.dept\_name

FROM instructor I

JOIN DeptMaxSalaries D ON I.dept\_name = D.dept\_name AND I.salary = D.max\_salary

WHERE I.salary < (SELECT MAX(salary) FROM instructor);

28. Retrieve students who have taken more than one course but have failed at least half of them (assuming grade 'F' is failing).

SQL: SELECT S.name

FROM student S

JOIN takes T ON S.ID = T.ID

GROUP BY S.ID, S.name

HAVING COUNT(CASE WHEN T.grade = 'F' THEN 1 END) >= COUNT(T.course\_id) / 2;

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

SQL: SELECT D.dept\_name

FROM department D

JOIN (

SELECT dept\_name, MAX(salary) AS max\_salary, COUNT(\*) AS num\_instructors

FROM instructor

GROUP BY dept\_name

) I ON D.dept\_name = I.dept\_name

WHERE I.max\_salary > D.budget / I.num\_instructors;

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

SQL: WITH CourseSemesters AS (

SELECT course\_id, semester, year, LAG(year) OVER (PARTITION BY course\_id, semester ORDER BY year) AS prev\_year

FROM section

)

SELECT DISTINCT course\_id

FROM CourseSemesters

WHERE prev\_year IS NOT NULL AND year <> prev\_year + 1;