Assignment = School Management System

1. **Display all students from Ahmedabad.**

```
SELECT * FROM students WHERE city='ahmedabad';
```

2. **Find all courses taught by "Mr. Sharma".**

```
SELECT courses.course_name,teachers.teacher_name

FROM courses

INNER JOIN teachers

ON courses.course_id=teachers.teacher_id

WHERE teachers.teacher_name='Mr. Sharma';
```

3. **Get details of students older than 18.**

```
SELECT * FROM students WHERE age>18;
```

4. **Show all teachers in "Computer Science" department.**

```
SELECT * FROM teachers WHERE department='computer science';
```

5. **List all enrollments where fees > 20000.**

```
SELECT * FROM enrollments WHERE fees>20000;
```

6. **Display students ordered by name ASC.**

```
SELECT * FROM students ORDER BY name ASC;
```

7. **Display teachers ordered by salary DESC.**

```
SELECT * FROM teachers ORDER BY salary DESC;
```

8. **List courses ordered by course_name ASC.**

```
SELECT * FROM courses ORDER BY course_name ASC;
```

9. **Find students aged between 15 and 20.**

```
SELECT * FROM students WHERE age BETWEEN 15 and 20 ;
```

10. **Show teachers with salary between 30000 and 50000.**

```
SELECT * FROM teachers WHERE salary BETWEEN 30000 and 50000;
```

11.List enrollments with fees between 10000 and 25000.

```
SELECT * FROM enrollments WHERE fees BETWEEN 10000 and 25000;
```

12. Show students from cities ('Ahmedabad','Surat','Baroda').

```
SELECT * FROM students WHERE city IN('ahmedabad','surat','baroda');
```

13. Find students NOT from 'Delhi'.

```
SELECT * FROM students WHERE NOT city='delhi';
```

14. List teachers from departments in ('Maths','Science').

```
SELECT * FROM teachers WHERE department IN('Mathematics','science');
```

15. Count total number of students.

```
SELECT COUNT(*) AS student_id FROM students;
```

16. Find average age of students.

```
SELECT AVG(age) AS avg_age from students;
```

17.Find total fees collected from enrollments.

```
SELECT SUM(fees) AS total_fees FROM enrollments;
```

18. Count how many courses are offered.

```
SELECT COUNT(*) AS course_name FROM courses;
```

19. Find average salary of teachers.

```
SELECT AVG(salary) AS avg_salary FROM teachers;
```

20.Find maximum fees paid by a student.

```
SELECT MAX(fees) AS max_fees FROM enrollments;
```

21. Count number of students in each city.

```
SELECT city, COUNT(*) AS total_name FROM students  GROUP BY city;
```

22.Find total fees collected per course.

```
SELECT SUM(enrollments.fees) AS sum_fees,courses.course_name

FROM enrollments
```

INNER JOIN courses

ON enrollments.enroll_id=courses.course_id

GROUP by courses.course_name;

23. Find average salary department-wise.

SELECT AVG(salary) AS avg_Salary, department FROM teachers GROUP BY department;

24. Count students per class, only show classes having more than 10 students.

Select name,count(*) AS class FROM students GROUP BY class HAVING count(*)>2;

25. Find teachers per department, only show departments with more than 5 teachers.

SELECT teacher_name,COUNT(*) AS department FROM teachers GROUP BY department HAVING COUNT(*)>3;

26. Find courses with average fees > 15000.

SELECT AVG(enrollments.fees)AS avg_fees,courses.course_name

FROM enrollments

INNER JOIN courses ON enrollments.course_id=courses.course_id

GROUP BY courses.course_name

HAVING AVG(enrollments.fees)>15000;

27. Find cities having more than 3 students enrolled.

SELECT students.city, COUNT(enrollments.student_id) AS total_student

FROM enrollments

INNER JOIN students ON enrollments.student_id=students.student_id

GROUP BY students.city HAVING COUNT(enrollments.student_id)>3;

28. Display student name with their enrolled course name.

SELECT students.name,courses.course_name,enrollments.enroll_id

FROM enrollments

```
INNER JOIN students ON enrollments.student_id=students.student_id

INNER JOIN courses ON enrollments.course_id=courses.course_id;
```

29. Show course name with teacher name.

```
SELECT courses.course_name,teachers.teacher_name

FROM teachers

INNER JOIN courses ON courses.teacher_id=teachers.teacher_id;
```

30. List student name, course name, and fees.

```
SELECT students.name,courses.course_name,enrollments.fees

FROM enrollments

INNER JOIN students ON enrollments.student_id=students.student_id

INNER JOIN courses ON enrollments.course_id=courses.course_id;
```

31. Find all students with their grades in each course.

```
SELECT students.name,courses.course_name, enrollments.grade

FROM enrollments

RIGHT JOIN students ON enrollments.student_id =students.student_id

LEFT JOIN courses ON enrollments.course_id=courses.course_id;
```

32.Display teacher name with course name they teach.

```
SELECT teachers.teacher_name,courses.course_name

FROM courses

LEFT JOIN teachers ON courses.teacher_id=teachers.teacher_id;
```

33.Show students who have not enrolled in any course (LEFT JOIN).

```
SELECT students.name,courses.course_name,enrollments.enroll_id

FROM enrollments

LEFT JOIN students ON enrollments.student_id=students.student_id

LEFT JOIN courses ON enrollments.course_id=courses.course_id;
```

34. Show courses with no students enrolled (RIGHT JOIN).

```
SELECT courses.course_name,students.name,enrollments.enroll_id
FROM enrollments
RIGHT JOIN courses ON enrollments.course_id=courses.course_id
RIGHT JOIN students ON enrollments.student_id=students.student_id;
```

35. Display student-city wise courses.

```
SELECT students.name,courses.course_name,students.city
FROM enrollments
INNER JOIN courses ON enrollments.course_id=courses.course_id
INNER JOIN students ON enrollments.student_id=students.student_id
GROUP BY students.name,courses.course_name;
```

36.Show course wise student count using join.

```
SELECT courses.course_name, COUNT(enrollments.student_id) AS stud_count
FROM enrollments
INNER JOIN courses
ON enrollments.course_id=courses.course_id
GROUP BY courses.course_name;
```

37. List teachers and students connected through courses.

```
SELECT teachers.teacher_name,students.name,courses.course_name
FROM teachers
INNER JOIN courses ON courses.teacher_id=teachers.teacher_id
INNER JOIN enrollments ON enrollments.course_id=courses.course_id
INNER JOIN students ON enrollments.student_id=students.student_id
GROUP BY teachers.teacher_name,students.name,courses.course_name;
```

38. Find students who enrolled in course 'Mathematics'.

```
SELECT students.name,courses.course_name
FROM enrollments
INNER JOIN courses ON courses.course_id=enrollments.course_id
INNER JOIN students ON students.student_id=enrollments.student_id
WHERE courses.course_name='Mathematics';
```

39.List teachers who teach at least one course.

```
SELECT DISTINCT teachers.teacher_name, courses.course_name
FROM courses
INNER JOIN teachers ON teachers.teacher_id = courses.teacher_id;
```

40. Find students who paid fees more than average fees.

```
SELECT students.name,enrollments.fees
FROM enrollments
INNER JOIN students ON students.student_id=enrollments.student_id
WHERE enrollments.fees>(SELECT AVG(fees) FROM enrollments);
```

41. Show students who enrolled in more than 2 courses.

```
SELECT students.name,COUNT(enrollments.course_id) AS total_course
FROM enrollments
INNER JOIN students ON students.student_id=enrollments.student_id
GROUP BY students.name,students.student_id
HAVING COUNT(enrollments.course_id)>2;
```

42. Find teachers whose salary is greater than average salary of all teachers.

```
SELECT teacher_name,salary FROM teachers WHERE salary>(SELECT AVG(salary) FROM
teachers);
```

43.Create a view StudentFeesView showing student name, course, and fees.

```
CREATE VIEW StudentFeesView AS
```

```
SELECT students.name AS stud_name,courses.course_name AS course_name
,enrollments.fees AS stud_fess  FROM enrollments

INNER JOIN courses ON enrollments.course_id=courses.course_id

INNER JOIN students ON students.student_id=enrollments.student_id
```

44. Create a view TeacherSalaryView showing teacher_name and salary.

```
CREATE VIEW TeacherSalaryView AS SELECT teacher_name,salary FROM teachers;
```

45. Create a view CourseStatsView with course, total students enrolled.

```
CREATE VIEW CourseStatsView AS

SELECT courses.course_name, COUNT(enrollments.student_id) AS total_stud

FROM enrollments

INNER JOIN courses ON courses.course_id=enrollments.course_id

GROUP BY courses.course_name;
```

46. Create a view CityWiseStudents showing city and count of students.

```
CREATE VIEW CityWiseStudents AS SELECT COUNT(name)AS total_stud, city FROM
students GROUP BY city;
```

47.Create a view HighFeesStudents for students paying fees > 20000.

```
CREATE VIEW HighFeesStudents AS SELECT students.name,enrollments.fees

FROM enrollments

INNER JOIN students ON students.student_id=enrollments.student_id

GROUP BY students.name,enrollments.fees HAVING enrollments.fees>20000;
```

48. Find course-wise average fees, only show courses having avg fees > 20000.

```
SELECT courses.course_name,AVG(enrollments.fees) AS avg_fees

FROM enrollments

INNER JOIN courses ON courses.course_id=enrollments.course_id

GROUP BY courses.course_name HAVING AVG(enrollments.fees)>20000;
```

49. Display teacher name and total number of students under them.

```
SELECT teachers.teacher_name, COUNT(enrollments.student_id)AS total_stud
FROM teachers
INNER JOIN courses ON courses.teacher_id=teachers.teacher_id
INNER JOIN enrollments ON courses.course_id=enrollments.course_id
GROUP BY teachers.teacher_name;
```

50.Find city-wise total fees collected, order by highest to lowest.

```
SELECT students.city,SUM(enrollments.fees) as total_fees
FROM enrollments
INNER JOIN students ON students.student_id=enrollments.student_id
GROUP BY students.city
ORDER BY total_fees DESC;
```