```sql
create Database StudentDB;

use StudentDB;


create table Students(StudentID int, Name varchar(20), Age int, DepartmentID int);

create table Departments (DepartmentID int , DepartmentName varchar(20));

create table Marks (MarkID int, StudentID int, Subject varchar(10), Marks int);


insert into Students(StudentID, Name, Age, DepartmentID) values

(1, 'Amit', 20, 1),

(2, 'Riya', 22, 2),

(3, 'Arjun', 20, 2)

(4, 'Karan', 21, 1),

(5, 'Neha', 23, 3);



insert into Departments(DepartmentID, DepartmentName ) values (1, 'Computer Science'), (2, 'Mathematic'), (3,
'Physics');


insert into Marks (MarkID, StudentID, Subject, Marks) values
```

```
(1, 1, 'DBMS', 85),
(2, 1, 'AI', 90),
(3, 2, 'DBMS', 75),
(4, 2, 'AI', 80),
(5, 3, 'DBMS', 88),
(6, 4, 'AI', 92),
(7, 5, 'DBMS', 70);



-- select * from Students;



-- 1. Retrieve all students along with their department names.
SELECT s.Name, d.DepartmentName
FROM Students s
JOIN Departments d ON d.DepartmentID = s.DepartmentID;


-- 2. Find the average marks obtained by each student.
```

```sql
SELECT s.Name, AVG(m.Marks) as AverageMarks
FROM Students s
JOIN Marks m ON s.StudentID = m.StudentID
GROUP BY s.StudentID, s.Name;


-- 3. Find the maximum marks in each subject.
SELECT Subject, MAX(Marks) as MaximumMarks
FROM Marks
GROUP BY Subject;


-- 4. List students who scored more than 80 in all subjects.
SELECT s.Name
FROM Students s
WHERE s.StudentID IN (
    SELECT StudentID
    FROM Marks
    GROUP BY StudentID
    HAVING MIN(Marks) > 80
);
```

```sql
-- 5. Retrieve all students ordered by their average marks in descending order.
SELECT s.StudentID, s.Name, AVG(m.Marks) as AverageMarks
FROM Students s
LEFT JOIN Marks m ON s.StudentID = m.StudentID
GROUP BY s.StudentID, s.Name
ORDER BY AverageMarks DESC;

-- 6. Find the total marks scored by students in each department.
SELECT d.DepartmentName, SUM(m.Marks) as TotalMarks
FROM Departments d
JOIN Students s ON d.DepartmentID = s.DepartmentID
JOIN Marks m ON s.StudentID = m.StudentID
GROUP BY d.DepartmentID, d.DepartmentName;

-- 7. Display students who have not received any marks yet.
SELECT s.StudentID, s.Name
FROM Students s
LEFT JOIN Marks m ON s.StudentID = m.StudentID
```

```sql
WHERE m.MarkID IS NULL;


-- 8. Retrieve department names along with the number of students enrolled in each department.
SELECT d.DepartmentName, COUNT(s.StudentID) as StudentCount
FROM Departments d
LEFT JOIN Students s ON d.DepartmentID = s.DepartmentID
GROUP BY d.DepartmentID, d.DepartmentName;


-- 9. Find the student with the highest marks in AI subject.
SELECT s.Name, m.Marks
FROM Students s
JOIN Marks m ON s.StudentID = m.StudentID
WHERE m.Subject = 'AI'
ORDER BY m.Marks DESC
LIMIT 1;


-- 10. Display all students along with their average marks (if any), showing 0 where marks are not available.
SELECT s.StudentID, s.Name,
       COALESCE(AVG(m.Marks), 0) as AverageMarks
```

```sql
FROM Students s

LEFT JOIN Marks m ON s.StudentID = m.StudentID

GROUP BY s.StudentID, s.Name;
```