Chapter 4

Database query using sql–order by, group by,having

New syllabus 2020-21

Informatics Practices

Class XII (As per CBSE Board)

Visit : python.mykvs.in for regular updates

MySQL Order By

MySQL Order By clause is used to sort the table data in either Ascending order or Descending order. By default, data is not inserted into Tables in any order unless we have an index.

So, If we want to retrieve the data in any particular order, we have to sort it by using MySQL Order By statement.

Syntax:-SELECT Column_Names

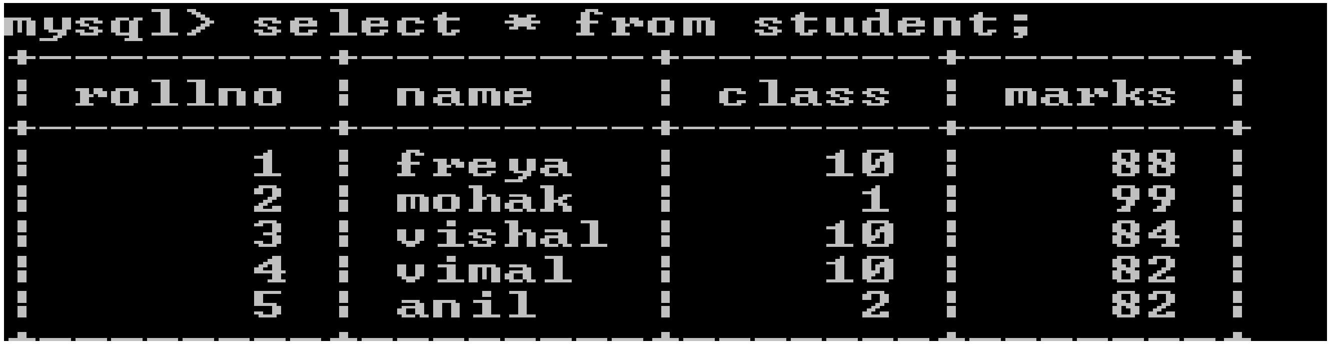
FROM Table_Name

ORDER BY {Column1}[ASC | DESC] {Column2}[ASC | DESC]

MySQL Order By

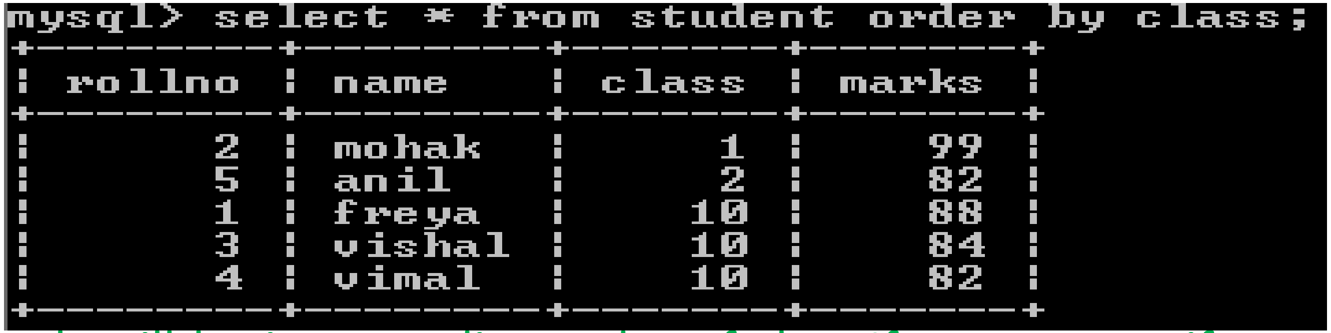
MySQL Order by– e.g.

Suppose we are having student table with following data.



```
mysql> select * from student;
+----------+----------+---------+---------+
| rollno   | name     | class   | marks   |
+----------+----------+---------+---------+
|        1 | freya    |      10 |      88 |
|        2 | mohak    |       1 |      99 |
|        3 | vishal   |      10 |      84 |
|        4 | vimal    |      10 |      82 |
|        5 | anil     |       2 |      82 |
+----------+----------+---------+---------+
```

Now we write the query – select * from student order by class;



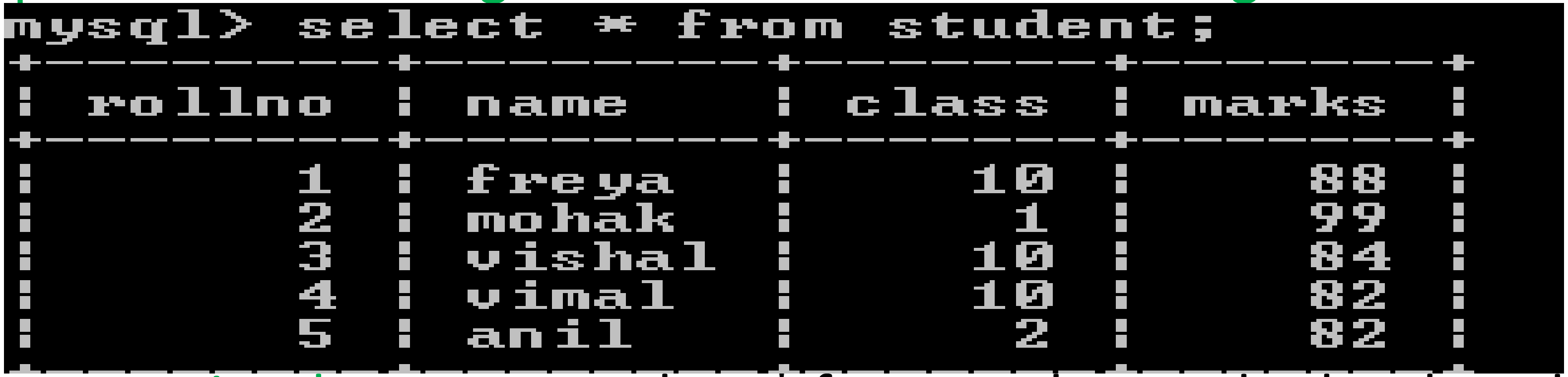
```
mysql> select * from student order by class;
+--------+--------+-------+-------+
| rollno | name   | class | marks |
+--------+--------+-------+-------+
|      2 | mohak  |     1 |    99 |
|      5 | anil   |     2 |    82 |
|      1 | freya  |    10 |    88 |
|      3 | vishal |    10 |    84 |
|      4 | vimal  |    10 |    82 |
+--------+--------+-------+-------+
```

Query result will be in ascending order of class.If we not specify asc/desc in query then ascending clause is applied by default

MySQL Order By

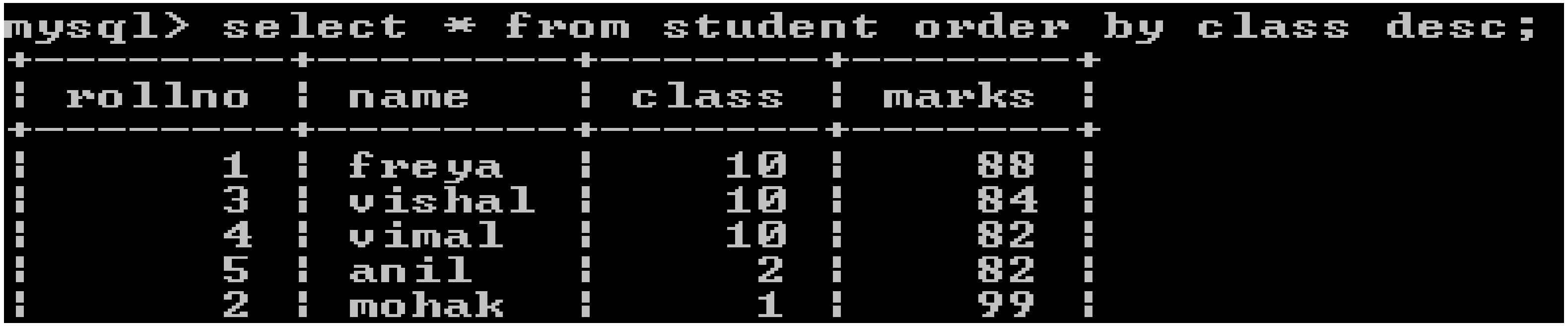
MySQL Order by– e.g.

Suppose we are having student table with following data.



```
mysql> select * from student;
+----------+----------+----------+----------+
| rollno   | name     | class    | marks    |
+----------+----------+----------+----------+
|        1 | freya    |       10 |       88 |
|        2 | mohak    |        1 |       99 |
|        3 | vishal   |       10 |       84 |
|        4 | vimal    |       10 |       82 |
|        5 | anil     |        2 |       82 |
```

Now we write the query – select * from student order by class desc;



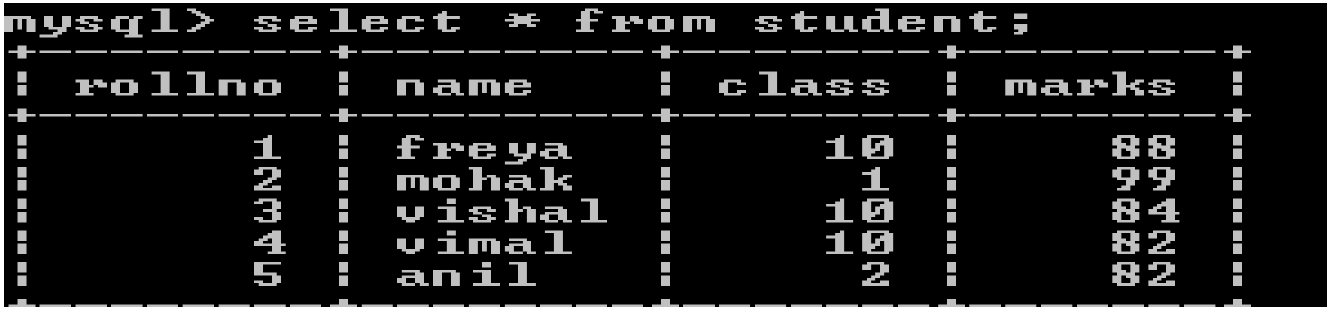
```
mysql> select * from student order by class desc;
+--------+--------+-------+-------+
| rollno | name   | class | marks |
+--------+--------+-------+-------+
|      1 | freya  |    10 |    88 |
|      3 | vishal |    10 |    84 |
|      4 | vimal  |    10 |    82 |
|      5 | anil   |     2 |    82 |
|      2 | mohak  |     1 |    99 |
```

Query result will be in descending order of class

MySQL Order By

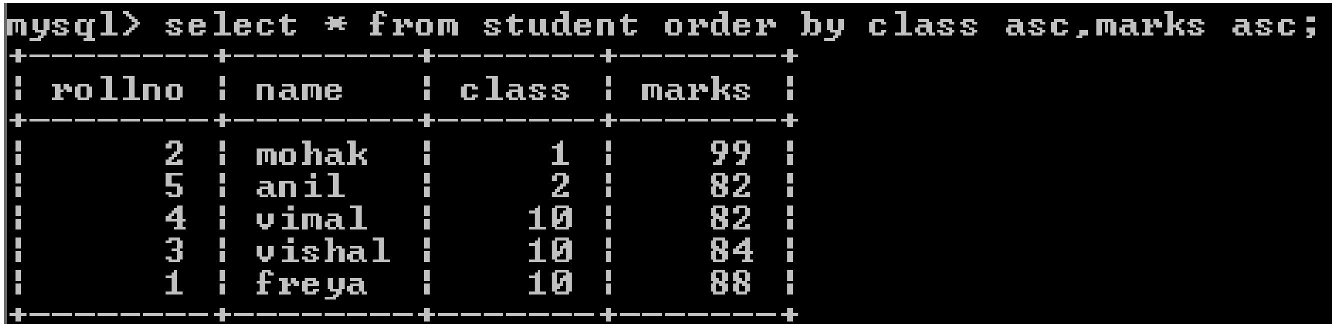
MySQL Order by – e.g.

Suppose we are having student table with following data.



```
mysql> select * from student;
+----------+-----------+----------+----------+
| rollno   | name      | class    | marks    |
+----------+-----------+----------+----------+
|        1 | freya     |       10 |       88 |
|        2 | mohak     |        1 |       99 |
|        3 | vishal    |       10 |       84 |
|        4 | vimal     |       10 |       82 |
|        5 | anil      |        2 |       82 |
+----------+-----------+----------+----------+
```

Now we write query–select * from student order by class asc, marks asc;



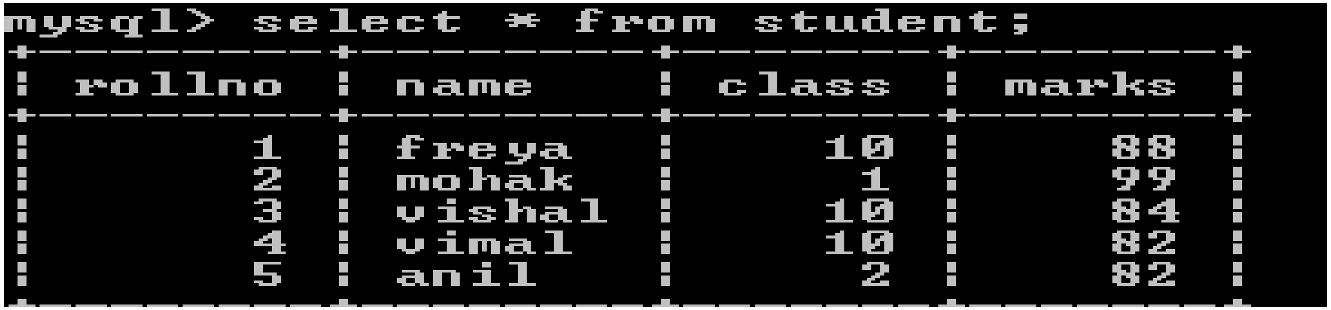
```
mysql> select * from student order by class asc,marks asc;
+--------+----------+--------+--------+
| rollno | name     | class  | marks  |
+--------+----------+--------+--------+
|      2 | mohak    |      1 |     99 |
|      5 | anil     |      2 |     82 |
|      4 | vimal    |     10 |     82 |
|      3 | vishal   |     10 |     84 |
|      1 | freya    |     10 |     88 |
+--------+----------+--------+--------+
```

Query result will be ascending order of class and if same class exists then ordering will done on marks column(ascending order)

MySQL Order By

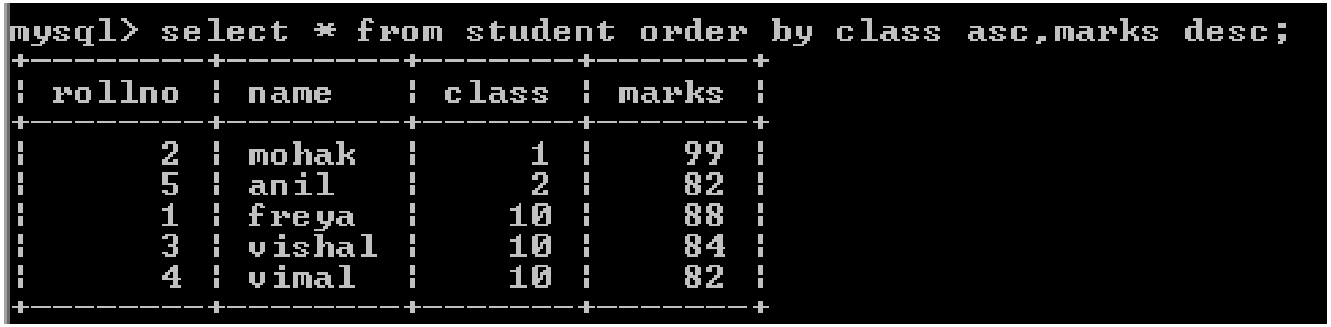
MySQL Order by– e.g.

Suppose we are having student table with following data.



```
mysql> select * from student;
+----------+----------+---------+---------+
| rollno   | name     | class   | marks   |
+----------+----------+---------+---------+
|        1 | freya    |      10 |      88 |
|        2 | mohak    |       1 |      99 |
|        3 | vishal   |      10 |      84 |
|        4 | vimal    |      10 |      82 |
|        5 | anil     |       2 |      82 |
+----------+----------+---------+---------+
```

Now we write query–select * from student order by class asc, marks desc;



```
mysql> select * from student order by class asc,marks desc;
+--------+--------+-------+-------+
| rollno | name   | class | marks |
+--------+--------+-------+-------+
|      2 | mohak  |     1 |    99 |
|      5 | anil   |     2 |    82 |
|      1 | freya  |    10 |    88 |
|      3 | vishal |    10 |    84 |
|      4 | vimal  |    10 |    82 |
+--------+--------+-------+-------+
```

Query result will be ascending order of class and if same class exists then ordering will done on marks column(descending order)

MySQL Group By

The GROUP BY clause groups a set of rows/records into a set of summary rows/records by values of columns or expressions. It returns one row for each group.

We often use the GROUP BY clause with aggregate functions such as SUM, AVG, MAX, MIN, and COUNT. The aggregate function that appears in the SELECT clause provides information about each group.

The GROUP BY clause is an optional clause of the SELECT statement.

Syntax –

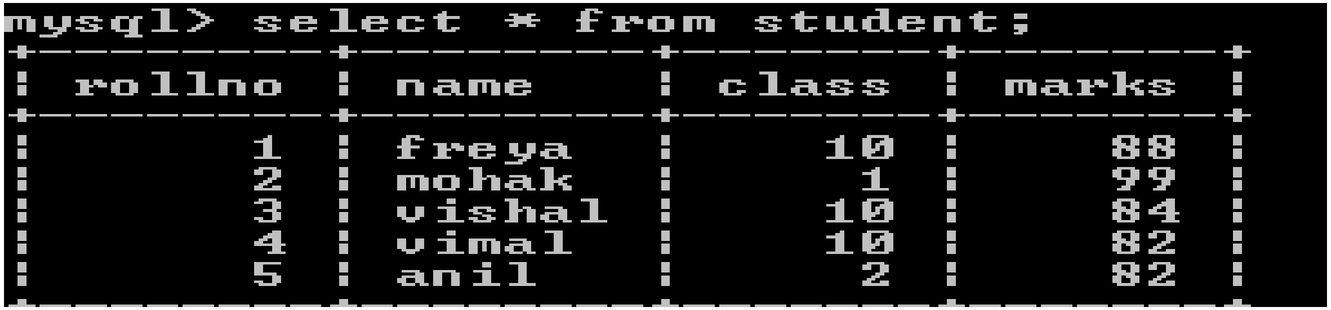
```
SELECT   1, c2,..., cn, aggregate_function(ci)
FROM  table WHERE where_conditions GROUP BY c1 , c2,...,cn;
```

Here c1,c2,ci,cn are column name

MySQL Group By

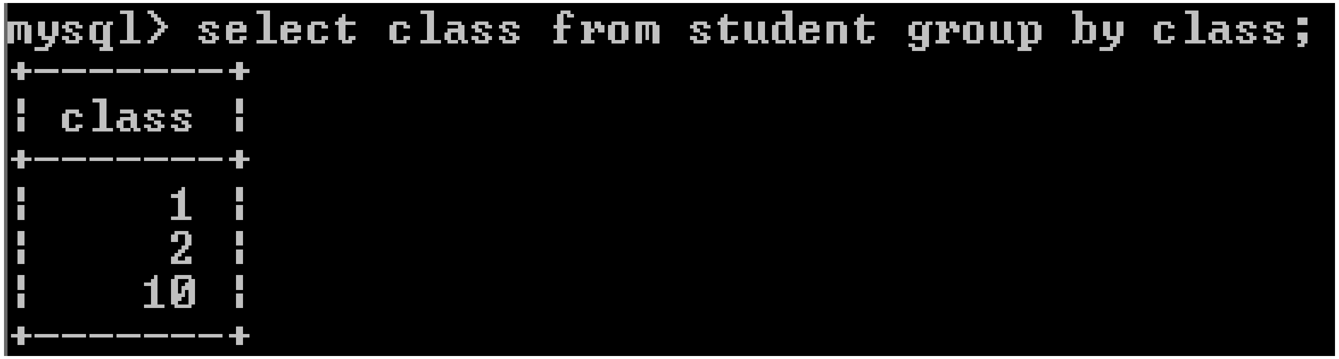
MySQL group by – e.g.

Suppose we are having student table with following data.



```
mysql> select * from student;
+----------+----------+---------+---------+
| rollno   | name     | class   | marks   |
+----------+----------+---------+---------+
|        1 | freya    |      10 |      88 |
|        2 | mohak    |       1 |      99 |
|        3 | vishal   |      10 |      84 |
|        4 | vimal    |      10 |      82 |
|        5 | anil     |       2 |      82 |
```

Now we write query–select class from student group by class;



```
mysql> select class from student group by class;
+-------+
| class |
+-------+
|     1 |
|     2 |
|    10 |
+-------+
```

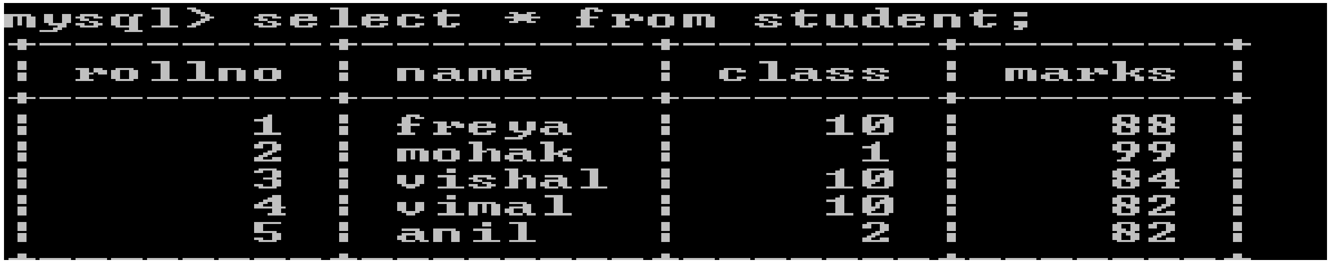
Query result will be unique occurrences of class values,just similar to use distinct clause like(select distinct class from student).

MySQL Group By

MySQL GROUP BY with aggregate functions

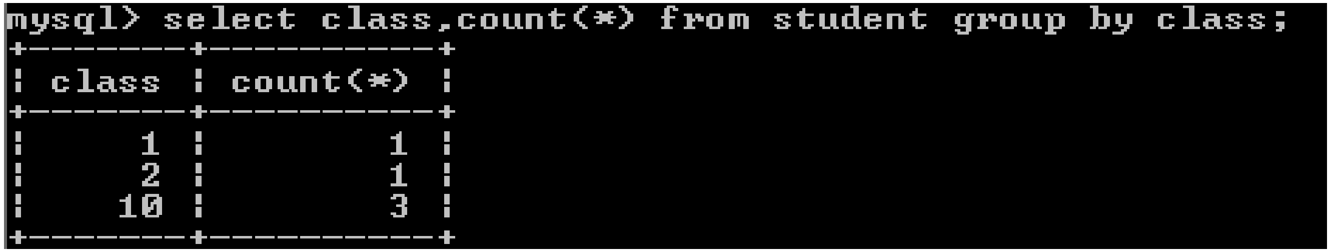
The aggregate functions allow us to perform the calculation of a set of rows and return a single value. The GROUP BY clause is often used with an aggregate function to perform calculation and return a single value for each subgroup.

For example, if we want to know the number of student in each class, you can use the COUNT function with the GROUP BY clause as follows:Suppose we are having student table with following data.



```
mysql> select * from student;
+----------+----------+---------+---------+
| rollno   | name     | class   | marks   |
+----------+----------+---------+---------+
|        1 | freya    |      10 |      88 |
|        2 | mohak    |       1 |      99 |
|        3 | vishal   |      10 |      84 |
|        4 | vimal    |      10 |      82 |
|        5 | anil     |       2 |      82 |
```

Now we write query–select class,count(*) from student group by class;



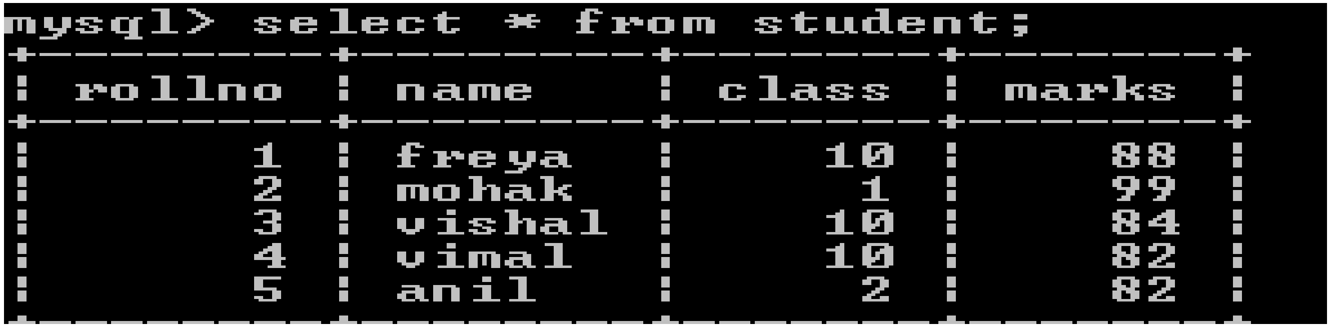
```
mysql> select class,count(*) from student group by class;
+-------+----------+
| class | count(*) |
+-------+----------+
|     1 |        1 |
|     2 |        1 |
|    10 |        3 |
+-------+----------+
```

Query result will be unique occurrences of class values along with counting of students(records) of each class(sub group).

MySQL Group By

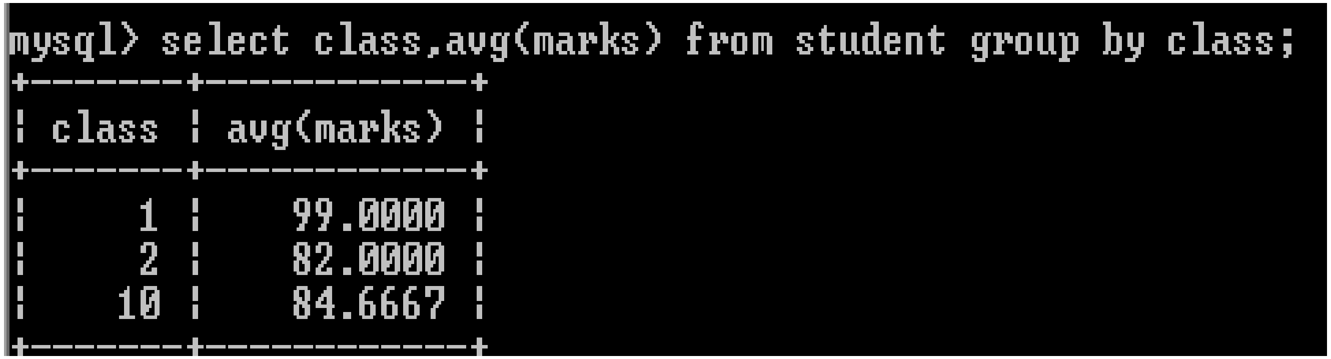
MySQL GROUP BY with aggregate functions

we are having student table with following data.



```
mysql> select * from student;
+----------+----------+---------+---------+
| rollno   | name     | class   | marks   |
+----------+----------+---------+---------+
|        1 | freya    |      10 |      88 |
|        2 | mohak    |       1 |      99 |
|        3 | vishal   |      10 |      84 |
|        4 | vimal    |      10 |      82 |
|        5 | anil     |       2 |      82 |
```

Now we write query–select class,avg(marks) from student group by class;



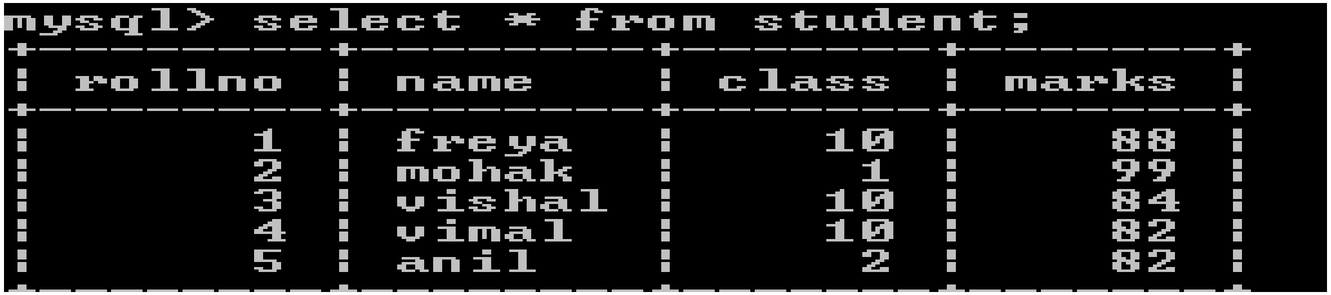
```
mysql> select class,avg(marks) from student group by class;
+-------+------------+
| class | avg(marks) |
+-------+------------+
|     1 |    99.0000 |
|     2 |    82.0000 |
|    10 |    84.6667 |
+-------+------------+
```

Query result will be unique occurrences of class values along with average marks of each class(sub group).

MySQL Group By

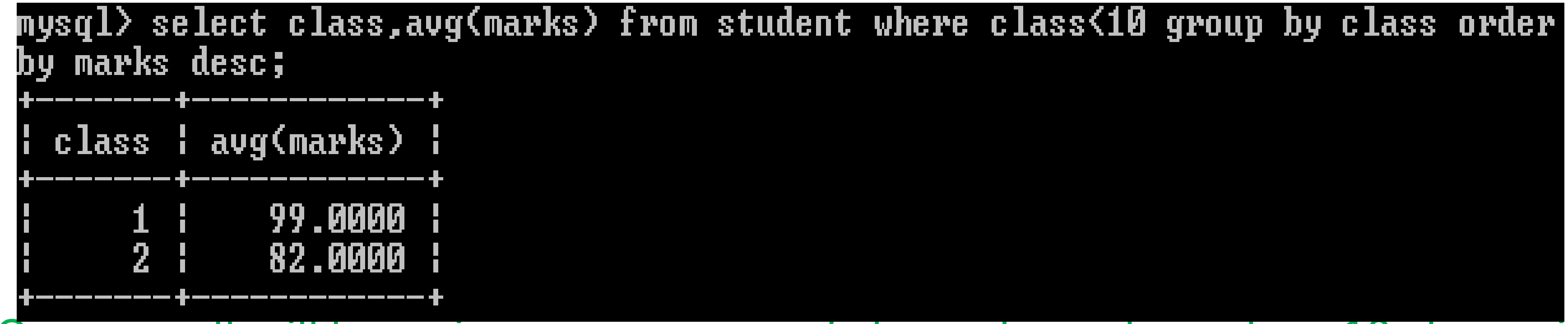
MySQL GROUP BY with aggregate functions (with where and order by clause)

we are having student table with following data.



```
mysql> select * from student;
+----------+----------+---------+---------+
| rollno   | name     | class   | marks   |
+----------+----------+---------+---------+
|        1 | freya    |      10 |      88 |
|        2 | mohak    |       1 |      99 |
|        3 | vishal   |      10 |      84 |
|        4 | vimal    |      10 |      82 |
|        5 | anil     |       2 |      82 |
+----------+----------+---------+---------+
```

Now we write query–select class,avg(marks) from student where class<10 group by class order by marks desc;



```
mysql> select class,avg(marks) from student where class<10 group by class order
by marks desc;
+-------+------------+
| class | avg(marks) |
+-------+------------+
|     1 |    99.0000 |
|     2 |    82.0000 |
+-------+------------+
```

Query result will be unique occurrences of class values where class<10 along with average marks of each class(sub group) and descending ofer of marks.

MySQL Group by with Having

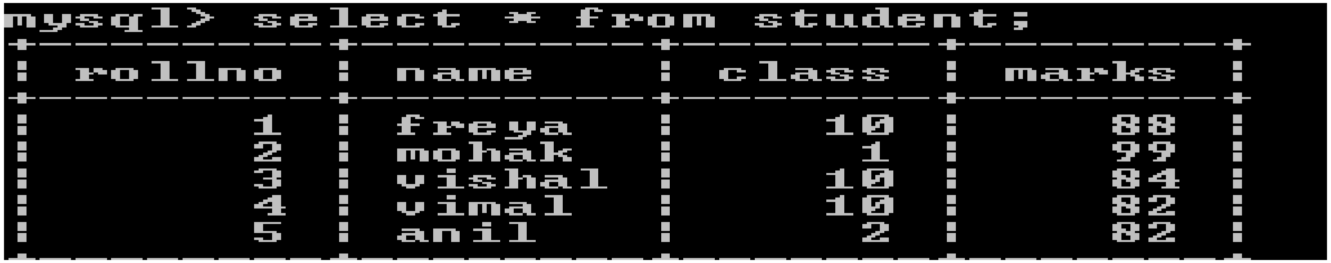
The HAVING clause is used in the SELECT statement to specify filter conditions for a group of rows or aggregates. The HAVING clause is often used with the GROUP BY clause to filter groups based on a specified condition. To filter the groups returned by GROUP BY clause, we use a HAVING clause.

WHERE is applied before GROUP BY, HAVING is applied after (and can filter on aggregates).

MySQL Group By with having

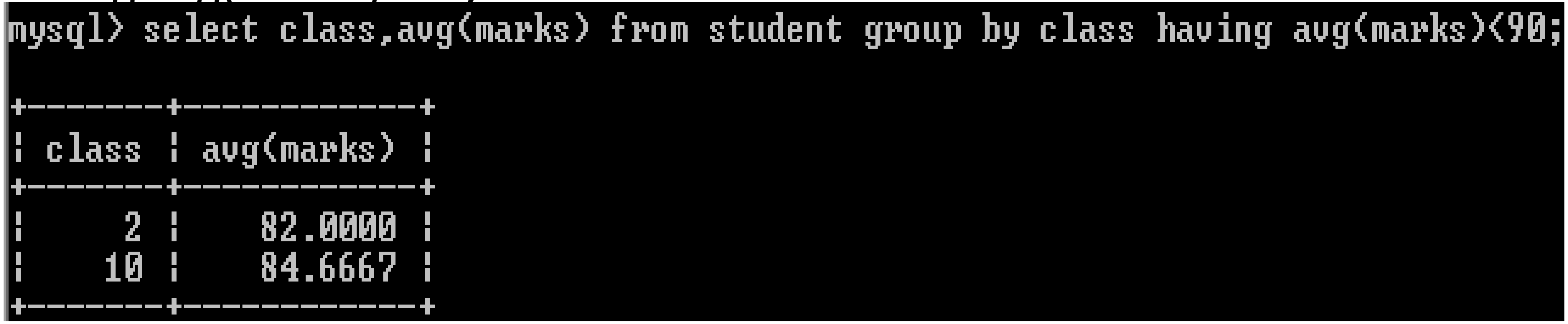
MySQL GROUP BY with aggregate functions

we are having student table with following data.



```
mysql> select * from student;
+----------+-----------+----------+----------+
| rollno   | name      | class    | marks    |
+----------+-----------+----------+----------+
|        1 | freya     |       10 |       88 |
|        2 | mohak     |        1 |       99 |
|        3 | vishal    |       10 |       84 |
|        4 | vimal     |       10 |       82 |
|        5 | anil      |        2 |       82 |
+----------+-----------+----------+----------+
```

Now we write query–select class,avg(marks) from student group by class having avg(marks)<90;



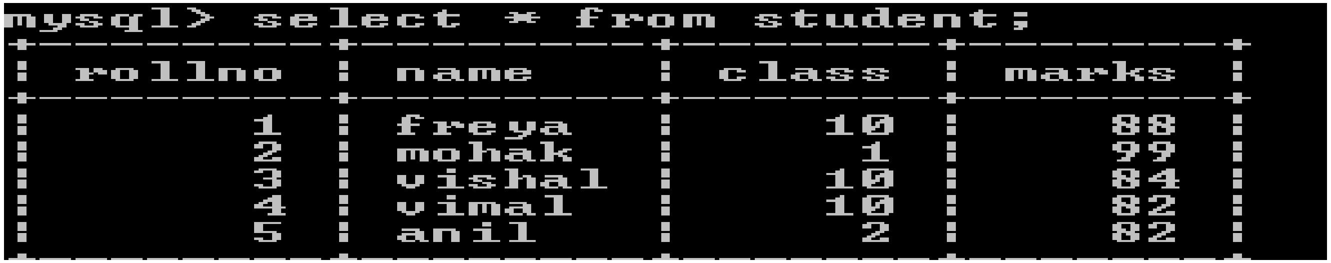
```
mysql> select class,avg(marks) from student group by class having avg(marks)<90;

+-------+------------+
| class | avg(marks) |
+-------+------------+
|     2 |    82.0000 |
|    10 |    84.6667 |
+-------+------------+
```

Query result will be unique occurrences of class values along with average marks of each class(sub group) and each class having average marks<90.

MySQL Group By with having

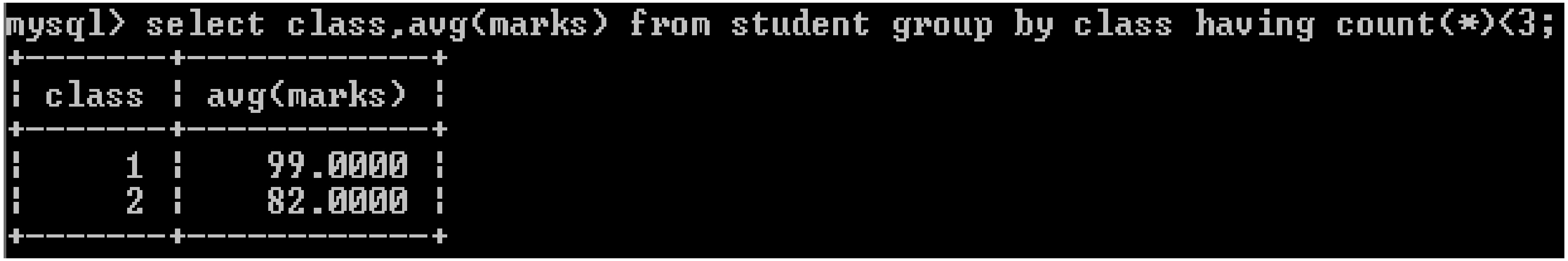
MySQL GROUP BY with aggregate functions

we are having student table with following data.



```
mysql> select * from student;
+----------+----------+----------+----------+
| rollno   | name     | class    | marks    |
+----------+----------+----------+----------+
|        1 | freya    |       10 |       88 |
|        2 | mohak    |        1 |       99 |
|        3 | vishal   |       10 |       84 |
|        4 | vimal    |       10 |       82 |
|        5 | anil     |        2 |       82 |
```

Now we write query–select class,avg(marks) from student group by class having count(*)<3;



```
mysql> select class,avg(marks) from student group by class having count(*)<3;
+-------+------------+
| class | avg(marks) |
+-------+------------+
|     1 |    99.0000 |
|     2 |    82.0000 |
+-------+------------+
```

Query result will be unique occurrences of class values along with average marks of each class(sub group) and each class having less than 3 rows.