Practical -6
Group BY and Having Cluse MySQL

MySQL GROUP BY

- The GROUP BY clause groups a set of rows into a set of summary rows by values of columns or expressions.
- The GROUP BY clause returns one row for each group.
- it reduces the number of rows in the result set.
- GROUP BY clause with aggregate functions such as SUM, AVG, MAX, MIN, and COUNT.
- The aggregate function that appears in the SELECT clause provides the information about each group.
- **Aggregate functions**

- AVG() - Returns the average value
- COUNT() - Returns the number of rows
- FIRST() - Returns the first value
- LAST() - Returns the last value
- MAX() - Returns the largest value
- MIN() - Returns the smallest value
- SUM() - Returns the sum

```
Syntax :   SELECT
              c1, c2,..., cn, aggregate_function(ci)
                  FROM     table
                  [WHERE     where_conditions ]
              GROUP BY c1 , c2,...,cn ;
```

C1,c2..cn : column names
The GROUP BY clause must appear after the FROM and WHERE clauses and before the HAVING , ORDER BY and LIMIT clauses:

e.g select average age of worker from employee table. [without group by]

```
mysql> select avg(age),designation from employee where designation='worker';
+----------+-------------+
| avg(age) | designation |
+----------+-------------+
|  25.0000 | worker      |
+----------+-------------+
```

Count a number of person living in each cities available in employee table.
e.g mysql> select count(name), city from employee group by city;

```
count(name) | city   |
+-------------+--------+
|           2 | NULL   |
|           2 | mumbai |
|           3 | baroda |
|           2 | pune   |
|           2 | surat  |
+-------------+--------+
```

e.g Display total number of order , average of order quantity done by each order.

select avg(qtyorder) , count(productno) , orderno from sales_order_detail group by orderno;

```
avg(qtyorder) | count(productno) | orderno |
+---------------+------------------+---------+
|        2.6667 |                3 | o19001  |
|       10.0000 |                1 | o19002  |
|        1.5000 |                2 | o19003  |
|        7.5000 |                2 | o19008  |
|        5.3333 |                3 | o46865  |
|        1.0000 |                2 | o46866  |
+---------------+------------------+---------+
```

e.g display the total of order amount(qtyorder*productrate) for each product ordered.
select sum(productrate*qtyorder) as totalamount,rom productno fsales_order_detail group by productno;

```
totalamount | productno |
+-------------+-----------+
|    17850.00 | p00001    |
|     1050.00 | p0345     |
|      525.00 | p06345    |
|     1575.00 | p07868    |
|     2625.00 | p07885    |
|     1575.00 | p07965    |
|     3150.00 | p07975
```

MySQL Having Clause

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. If the GROUP BY clause is omitted, the HAVING clause behaves like the WHERE clause.

Notice that the HAVING clause applies a filter condition **to each group of rows**, while the WHERE clause applies the filter condition to each individual row.

Syntax :

```
    SELECT
  select_list
FROM
  table_name
WHERE
  search_condition
GROUP BY
  group_by_expression
HAVING
  group_condition;
```

e.g select only those orderdetails whose total order amount is greater than 3000.

```
mysql> select  orderno, sum(qtyorder*productrate)  as total from
sales_order_detail group by orderno having total>3000;
+---------+---------+
| orderno | total   |
+---------+---------+
| o19001  | 4200.00 |
| o19002  | 5250.00 |
| o19008  | 7875.00 |
| o46865  | 8400.00 |
+---------+---------+
```

e.g display product details which product is ordered more than equal to 2 times.

Steps 1 : productno and order no is available in sales_order_detail.
Count the productno where group by on productno is required.

1. **select count(productno) as countproduct from sales_order_detail group by productno;**

```
| countproduct |
+--------------+
|            4 |
|            1 |
|            1 |
|            1 |
|            2 |
|            2 |
|            2 |
```

Step 2 : count should be more than 2. Condition required on count() aggregate functions so Having clause is used.

```
Mysql> select productno  from sales_order_detail group by productno having count(productno) >= 2;
   productno |
+-----------+
| p00001    |
| p07885    |
| p07965    |
| p07975
```

Step3 : for outer query for display the product details. Which is available in product table.

```
Mysql > select *from product where productno in (select productno  from sales_order_detail group by productno having count(productno) >= 2);

| p00001    | t-shirts     |        5.00 | piece   |      200 |        50 |    350.00 |    250.00 |
| p07885    | pull overs   |        2.50 | piece   |       80 |        30 |    700.00 |    450.00 |
| p07965    | denim shirts |        4.00 | piece   |      100 |         4 |    350.00 |    250.00 |
| p07975    | lycra tops   |        5.00 | piece   |       70 |        30 |    300.00 |    175.00
```

Exercise

create table for following
student (**sid**,name) sid primary key
subject (subid,sname) subid is primary key
stud_sub (sid,subid,teachername,marks)   primary key(sid,subid).
Sid is foreign key from references student(sid)
Subid is foreign key from references subject(subid).

Student

1	simon
2	alvin
3	vidya
4	rohit
5	kaushik
6	reema

Subject

1	CONSM
2	DBMS
3	physics
4	Maths
5	Biology

Stud_sub table data

sid	subid	teachernam e	marks
1	1	Reshma	62

1	2	Vihar	50
1	3	Bhavik	55
2	1	Jigar	64
2	2	kamlesh	68
2	3	suhana	72
2	4	Reshma	59
2	5	Vihar	71
3	1	Jigar	65
3	2	Bhavik	66
3	3	suhana	54
4	1	Vihar	81
4	4	suhana	64
4	5	Jigar	64
5	2	kamlesh	70
5	3	Reshma	56
6	1	Bhavik	76
6	4	Jigar	68

Perfom following query based on given table data

1. count number of students who has join the subject physics.
   Hint :: not required group by . use count() aggregate function.
2. Find the maximum mark of student id =1 .
3. Find the maximum marks for each subject. (hint : use group by)
   Display maximum mark and subid both.
4. Find maximum marks of each students.
   Display maximum marks and subid, studentid. (hint : group by sid)

5 .Find the total marks for each subject.

6. Find number of students enrolled in each subject .
7. display subjectid and total marks whose total marks is grather than 300.(hint : groupby and having)
8. display subject name whose total marks is grater than 300.(hint : group by ,having and subquery)
9. display total number of students and teacername each teacherwise. (hint group by)
10. display teacher name who is teaching more than 3 students (hint : group by , having)
11. display teachers name who is teaching subject DBMS (hint subquery)
12. display each subject , display the student name who got maximum mark.
    (display subjected and studentname)
13. display number of subject teach by each teacher.
14. . display only those teacher who is teach more than 2 subject. (hint group by , having).
15. display those subject name which is teach by more than 2 teacher. (hint : group by , having,subquery)
16. Find out the total marks of student vidya.
17. Display the teacher name who has given total marks is more than 180.(hint groupby,having)