MySQL HAVING

mysqltutorial.org/mysql-having.aspx

Summary: in this tutorial, you will learn how to use **MySQL HAVING** clause to specify a filter condition for groups of rows or aggregates.

Introduction to MySQL `HAVING` clause

The `HAVING` clause is used in the `SELECT` statement to specify filter conditions for a group of rows or aggregates.

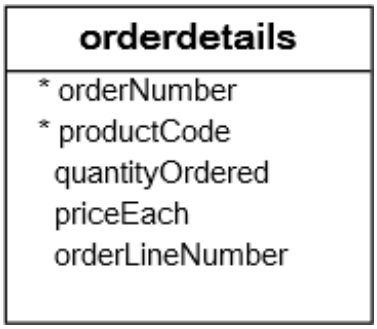
The `HAVING` clause is often used with the `GROUP BY` clause to filter groups based on a specified condition. If the `GROUP BY` clause is omitted, the `HAVING` clause behaves like the `WHERE` clause.

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.

MySQL `HAVING` clause examples

Let's take some examples of using the `HAVING` clause to see how it works. We will use the `orderdetails` table in the sample database for the demonstration.

You can use `GROUP BY` clause to get order numbers, the number of items sold per order, and total sales for each:




```
1 SELECT
2     ordernumber,
3     SUM(quantityOrdered) AS itemsCount,
4     SUM(priceeach*quantityOrdered) AS total
5 FROM
6     orderdetails
7 GROUP BY ordernumber;
```

Try It Out

Now, you can find which order has total sales greater than `1000` by using the `HAVING` clause as follows:

ordernumber	itemsCount	total
10100	151	10223.83
10101	142	10549.01
10102	80	5494.78
10103	541	50218.95
10104	443	40206.20
10105	545	53959.21
10106	675	52151.81
10107	229	22292.62

```
1 SELECT
2     ordernumber,
3     SUM(quantityOrdered) AS itemsCount,
4     SUM(priceeach*quantityOrdered) AS total
5 FROM
6     orderdetails
7 GROUP BY ordernumber
8 HAVING total > 1000;
```

Try It Out

You can construct a complex condition in the HAVING clause using logical operators such as OR and AND. Suppose you want to find which orders have total sales greater than 1000 and contain more than 600 items, you can use the following query:

```
SELECT
    ordernumber,
    SUM(quantityOrdered) AS itemsCount,
    SUM(priceeach*quantityOrdered) AS total
FROM
    orderdetails
GROUP BY ordernumber
HAVING total > 1000 AND itemsCount > 600;
```

ordernumber	itemsCount	total
10100	151	10223.83
10101	142	10549.01
10102	80	5494.78
10103	541	50218.95
10104	443	40206.20
10105	545	53959.21
10106	675	52151.81
10107	229	22292.62

Try It Out

Suppose you want to find all orders that have shipped and total sales greater than 1500, you can join the orderdetails table with the orders table using the INNER JOIN clause and apply a condition on status column and total aggregate as shown in the following query:

```
SELECT
    a.ordernumber, status, SUM(priceeach*quantityOrdered)
total
FROM
    orderdetails a
        INNER JOIN
    orders b ON b.ordernumber = a.ordernumber
GROUP BY ordernumber, status
HAVING status = 'Shipped' AND total > 1500;
```

ordernumber	itemsCount	total
10106	675	52151.81
10126	617	57131.92
10135	607	55601.84
10165	670	67392.85
10168	642	50743.65
10204	619	58793.53
10207	615	59265.14
10212	612	59830.55
10222	717	56822.65

Try It Out

The HAVING clause is only useful when you use it with the GROUP BY clause to generate the output of the high-level reports. For example, you can use the HAVING clause to answer statistical questions like finding the number orders this month, this quarter, or this year that have total sales greater than 10K.

In this tutorial, you have learned how to use the MySQL HAVING clause with the GROUP BY clause to specify filter conditions for groups of rows or aggregates.

ordernumber	status	total
10100	Shipped	10223.83
10101	Shipped	10549.01
10102	Shipped	5494.78
10103	Shipped	50218.95
10104	Shipped	40206.20
10105	Shipped	53959.21
10106	Shipped	52151.81