Introduction to SQL Server HAVING clause

The HAVING clause is often used with the [GROUP BY](https://www.sqlservertutorial.net/sql-server-basics/sql-server-group-by/) clause to filter groups based on a specified list of conditions. The following illustrates the HAVING clause syntax:

The SQL Server (Transact-SQL) HAVING clause is used in combination with the GROUP BY **clause to restrict the groups of returned rows to only those whose the condition is TRUE**.

The HAVING clause was added to SQL because the WHERE **keyword cannot be used with aggregate functions**.

SELECT

select\_list

FROM

table\_name

GROUP BY

group\_list

HAVING

conditions;

Code language: SQL (Structured Query Language) (sql)

HAVING is like WHERE but operates on grouped records.

HAVING requires that a GROUP BY clause is present.

Groups that meet the HAVING criteria will be returned.

HAVING is used with aggregrates: [COUNT](https://www.dofactory.com/sql/count), [MAX](https://www.dofactory.com/sql/max), [SUM](https://www.dofactory.com/sql/sum), etc.

## Syntax

## [#](https://www.dofactory.com/sql/having#syntax)

HAVING syntax.

1. **SELECT column-names**
2. **FROM table-name**
3. **WHERE condition**
4. **GROUP BY column-names**
5. **HAVING condition**

HAVING syntax with ORDER BY.

1. **SELECT column-names**
2. **FROM table-name**
3. **WHERE condition**
4. **GROUP BY column-names**
5. **HAVING condition**
6. **ORDER BY column-names**

In this syntax, the [GROUP BY](https://www.sqlservertutorial.net/sql-server-basics/sql-server-group-by/) clause summarizes the rows into groups and the HAVING clause applies one or more conditions to these groups. Only groups that make the conditions evaluate to TRUE are included in the result. In other words, the groups for which the condition evaluates to  FALSE or UNKNOWN are filtered out.

Because SQL Server processes the HAVING clause after the GROUP BY clause, you cannot refer to the [aggregate function](https://www.sqlservertutorial.net/sql-server-aggregate-functions/) specified in the select list by using the [column alias](https://www.sqlservertutorial.net/sql-server-basics/sql-server-alias/). The following query will fail:

SELECT

column\_name1,

column\_name2,

aggregate\_function (column\_name3) column\_alias

FROM

table\_name

GROUP BY

column\_name1,

column\_name2

HAVING

column\_alias > value;

Code language: SQL (Structured Query Language) (sql)

Instead, you must use the aggregate function expression in the HAVING clause explicitly as follows:

SELECT

column\_name1,

column\_name2,

aggregate\_function (column\_name3) alias

FROM

table\_name

GROUP BY

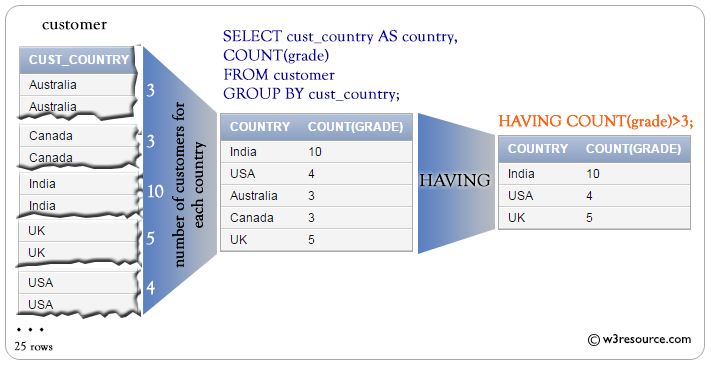
column\_name1,

column\_name2

HAVING

aggregate\_function (column\_name3) > value;

Code language: SQL (Structured Query Language) (sql)

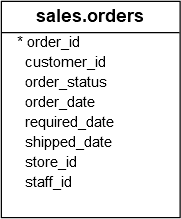


SQL Server HAVING examples

Let’s take some examples to understand how the HAVING clause works.

SQL Server HAVING with the COUNT function example

See the following orders table from the [sample database](https://www.sqlservertutorial.net/sql-server-sample-database/):



The to find the customers who placed at least two orders per year:

SELECT

customer\_id,

YEAR (order\_date),

COUNT (order\_id) order\_count

FROM

sales.orders

GROUP BY

customer\_id,

YEAR (order\_date)

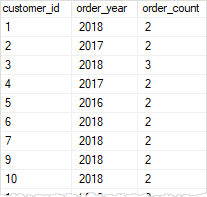
HAVING

COUNT (order\_id) >= 2

ORDER BY

customer\_id;

Code language: SQL (Structured Query Language) (sql)

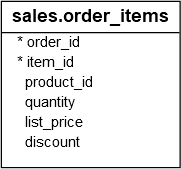


In this example:

* First, the GROUP BY clause groups the sales order by customer and order year. The [COUNT()](https://www.sqlservertutorial.net/sql-server-aggregate-functions/sql-server-count/) function returns the number of orders each customer placed in each year.
* Second, the HAVING clause filtered out all the customers whose number of orders is less than two.

SQL Server HAVING clause with the SUM() function example

Consider the following order\_items table:



The following statement finds the sales orders whose net values are greater than 20,000:

SELECT

order\_id,

SUM (

quantity \* list\_price \* (1 - discount)

) net\_value

FROM

sales.order\_items

GROUP BY

order\_id

HAVING

SUM (

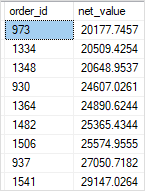
quantity \* list\_price \* (1 - discount)

) > 20000

ORDER BY

net\_value;

Code language: SQL (Structured Query Language) (sql)

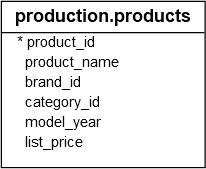


In this example:

* First, the [SUM()](https://www.sqlservertutorial.net/sql-server-aggregate-functions/sql-server-sum/) function returns the net values of sales orders.
* Second, the HAVING clause filters the sales orders whose net values are less than or equal to 20,000.

SQL Server HAVING clause with MAX and MIN functions example

See the following products table:



The following statement first finds the maximum and minimum list prices in each product category. Then, it filters out the category which has the maximum list price greater than 4,000 or the minimum list price less than 500:

SELECT

category\_id,

MAX (list\_price) max\_list\_price,

MIN (list\_price) min\_list\_price

FROM

production.products

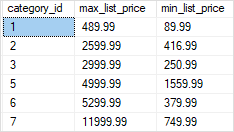
GROUP BY

category\_id

HAVING

MAX (list\_price) > 4000 OR MIN (list\_price) < 500;

Code language: SQL (Structured Query Language) (sql)



SQL Server HAVING clause with AVG() function example

The following statement finds product categories whose [average](https://www.sqlservertutorial.net/sql-server-aggregate-functions/sql-server-avg/) list prices are between 500 and 1,000:

SELECT

category\_id,

AVG (list\_price) avg\_list\_price

FROM

production.products

GROUP BY

category\_id

HAVING

AVG (list\_price) BETWEEN 500 AND 1000;

Code language: SQL (Structured Query Language) (sql)

https://www.sqlservertutorial.net/wp-content/uploads/SQL-Server-HAVING-AVG-example.png

Exercises

Problem: List the number of customers in each country. Only include countries with more than 10 customers.

|  |
| --- |
| **CUSTOMER** |
| Id |
| FirstName |
| LastName |
| City |
| Country |
| Phone |

**SELECT Country, COUNT(Id) AS Customers**

**FROM Customer**

**GROUP BY Country**

**HAVING COUNT(Id) > 10**

Problem: List the number of customers in each country, except the USA, sorted high to low. Only include countries with 9 or more customers.

|  |
| --- |
| **CUSTOMER** |
| Id |
| FirstName |
| LastName |
| City |
| Country |
| Phone |

**SELECT Country, COUNT(Id) AS Customers**

**FROM Customer**

**WHERE Country <> 'USA'**

**GROUP BY Country**

**HAVING COUNT(Id) >= 9**

**ORDER BY COUNT(Id) DESC**

HAVING with AVG BETWEEN

|  |
| --- |
| **ORDER** |
| Id |
| OrderDate |
| OrderNumber |
| CustomerId |
| TotalAmount |

|  |
| --- |
| **CUSTOMER** |
| Id |
| FirstName |
| LastName |
| City |
| Country |
| Phone |

Problem: List customer with average orders between $1000 and $1200.

**SELECT FirstName, LastName,**

**CONVERT(DECIMAL(10,2), AVG(TotalAmount)) AS 'Average Order'**

**FROM [Order] O**

**JOIN Customer C ON O.CustomerId = C.Id**

**GROUP BY FirstName, LastName**

**HAVING AVG(TotalAmount) BETWEEN 1000 AND 1200**

**Difference between Having clause and Group by clause :**

| S.No. | Having Clause | GroupBy Clause |
| --- | --- | --- |
| 1. | It is used for applying some extra condition to the query. | The groupby clause is used to group the data according to particular column or row. |
| 2. | Having can be used without groupby clause,in aggregate function,in that case it behaves like where clause. | groupby can be used without having clause with the select statement. |
| 3. | The having clause can contain aggregate functions. | It cannot contain aggregate functions. |
| 4. | It restrict the query output by using some conditions | It groups the output on basis of some rows or columns. |

## HAVING vs. WHERE

The [WHERE](https://www.sqltutorial.org/sql-where/) clause applies the condition to individual rows before the rows are summarized into groups by the GROUP BY clause. However, the HAVING clause applies the condition to the groups after the rows are grouped into groups.

Therefore, it is important to note that the HAVING clause is applied after whereas the WHERE clause is applied before the GROUP BY clause.

## SQL HAVING clause examples

We will take the employees and departments tables in the [sample database](https://www.sqltutorial.org/sql-sample-database/) for the demonstration.



To get the managers and their direct reports, you use the GROUP BY clause to group employees by the managers and use the [COUNT function](https://www.sqltutorial.org/sql-aggregate-functions/sql-count/) to count the direct reports.

The following query illustrates the idea:

SELECT

manager\_id,

first\_name,

last\_name,

COUNT(employee\_id) direct\_reports

FROM

employees

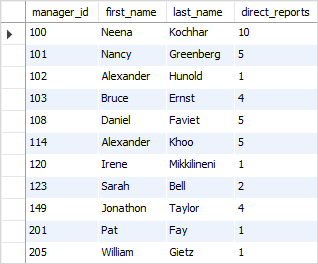
WHERE

manager\_id IS NOT NULL

GROUP BY manager\_id;

Code language: SQL (Structured Query Language) (sql)

[**Try It**](https://www.sqltutorial.org/seeit/query/sql-having/#1)



To find the managers who have at least five direct reports, you add a HAVING clause to the query above as the following:

SELECT

manager\_id,

first\_name,

last\_name,

COUNT(employee\_id) direct\_reports

FROM

employees

WHERE

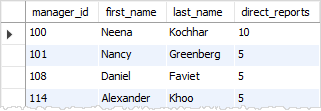
manager\_id IS NOT NULL

GROUP BY manager\_id

HAVING direct\_reports >= 5;

Code language: SQL (Structured Query Language) (sql)

[**Try It**](https://www.sqltutorial.org/seeit/query/sql-having/#2)



### SQL HAVING with SUM function example

The following statement calculates the [sum](https://www.sqltutorial.org/sql-aggregate-functions/sql-sum/) of salary that the company pays for each department and selects only the departments with the sum of salary [between](https://www.sqltutorial.org/sql-between/) 20000 and 30000.

SELECT

department\_id, SUM(salary)

FROM

employees

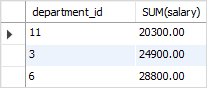
GROUP BY department\_id

HAVING SUM(salary) BETWEEN 20000 AND 30000

ORDER BY SUM(salary);

Code language: SQL (Structured Query Language) (sql)

[**Try It**](https://www.sqltutorial.org/seeit/query/sql-having/#3)



### SQL HAVING with MIN function example

To find the department that has employees with the lowest salary greater than 10000, you use the following query:

SELECT

e.department\_id,

department\_name,

MIN(salary)

FROM

employees e

INNER JOIN departments d ON d.department\_id = e.department\_id

GROUP BY

e.department\_id

HAVING

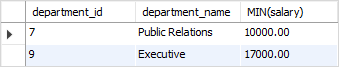
MIN(salary) >= 10000

ORDER BY

MIN(salary);

Code language: SQL (Structured Query Language) (sql)

[**Try It**](https://www.sqltutorial.org/seeit/query/sql-having/#4)



How the query works.

1. First, use the GROUP BY clause to groups employees by department.
2. Second, use the [MIN](https://www.sqltutorial.org/sql-aggregate-functions/sql-min/) function to find the lowest salary per group.
3. Third, apply the condition to the HAVING clause.

### SQL HAVING clause with AVG function example

To find the departments that have the average salaries of employees between 5000 and 7000, you use the [AVG](https://www.sqltutorial.org/sql-aggregate-functions/sql-avg/) function as the following query:

SELECT

e.department\_id,

department\_name,

ROUND(AVG(salary), 2)

FROM

employees e

INNER JOIN departments d ON d.department\_id = e.department\_id

GROUP BY

e.department\_id

HAVING

AVG(salary) BETWEEN 5000

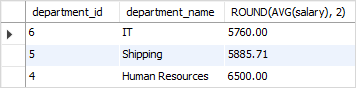
AND 7000

ORDER BY

AVG(salary);

Code language: SQL (Structured Query Language) (sql)

[**Try It**](https://www.sqltutorial.org/seeit/query/sql-having/#5)



## SQL HAVING using where

In the following example, the SQL WHERE clause along with the HAVING clause have used to make a select statement.

**Example:**

**Sample table: customer**

To get list of cust\_city, sum of opening\_amt, average of receive\_amt and maximum payment\_amt from customer table with following conditions-

1. grade of customer table must be 2,

2. average of receive\_amt for each group of cust\_city must be more than 500,

then, the following SQL statement can be used:

**SQL Code:**

SELECT cust\_city,SUM(opening\_amt),

AVG(receive\_amt),MAX(payment\_amt)

FROM customer

WHERE grade=2

GROUP BY cust\_city

HAVING AVG(receive\_amt)>500;

Copy

Sample Output:

CUST\_CITY SUM(OPENING\_AMT) AVG(RECEIVE\_AMT) MAX(PAYMENT\_AMT)

----------------------------------- ---------------- ---------------- ----------------

Bangalore 29000 8250 7000

Brisban 7000 7000 9000

London 10000 7000 7000

Mumbai 7000 11000 9000

New York 3000 5000 2000

Torento 8000 7000 7000

## SQL HAVING with order by

In the following example, the SQL WHERE clause along with the HAVING clause is used to make a query. An ORDER BY clause arranges the final result in the specific order. The default order is ascending.

**Example:**

Sample table : customer

To get list of cust\_city, sum of opening\_amt, average of receive\_amt and maximum payment\_amt from customer table with following conditions-

1. grade of customer table must be 2,

2. average of receive\_amt for each group of cust\_city must be more than 500,

3. the output should be arranged in the ascending order of SUM(opening\_amt),

then, the following SQL statement can be used:

**SQL Code:**

SELECT cust\_city,SUM(opening\_amt),

AVG(receive\_amt),MAX(payment\_amt)

FROM customer

WHERE grade=2

GROUP BY cust\_city

HAVING AVG(receive\_amt)>500

ORDER BY SUM(opening\_amt);

Copy

Sample Output:

CUST\_CITY SUM(OPENING\_AMT) AVG(RECEIVE\_AMT) MAX(PAYMENT\_AMT)

----------------------------------- ---------------- ---------------- ----------------

New York 3000 5000 2000

Brisban 7000 7000 9000

Mumbai 7000 11000 9000

Torento 8000 7000 7000

London 10000 7000 7000

Bangalore 29000 8250 7000