

MySQL Aggregate Functions

MySQL's aggregate function is used to **perform calculations on multiple values and return the result in a single value like the average of all values**, the sum of all values, and maximum & minimum value among certain groups of values. We mostly use the aggregate functions with **SELECT statements**

in the data query languages.

Syntax:

The following are the syntax to use aggregate functions in MySQL:

```
function_name (DISTINCT | ALL expression)
```

In the above syntax, we had used the following parameters:

- First, we need to specify the name of the aggregate function.
- Second, we use the **DISTINCT** modifier when we want to calculate the result based on distinct values or **ALL** modifiers when we calculate all values, including duplicates. The default is ALL.
- Third, we need to specify the expression that involves columns and arithmetic operators.

There are various aggregate functions available in **MySQL**

. Some of the most commonly used aggregate functions are summarised in the below table:

Aggregate Function	Descriptions
count()	It returns the number of rows, including rows with NULL values in a group.
sum()	It returns the total summed values (Non-NULL) in a set.
average()	It returns the average value of an expression.
min()	It returns the minimum (lowest) value in a set.
max()	It returns the maximum (highest) value in a set.

Why we use aggregate functions?

We mainly use the aggregate functions in databases, spreadsheets and many other data manipulation software packages. In the context of business, different organization levels need different information such as top levels managers interested in knowing whole figures and not the individual details. These functions produce the summarised data from our database. Thus they are extensively used in economics and finance to represent the economic health or stock and sector performance.

Let us take an example of myflix (video streaming website which has huge collections of the movie) database, where management may require the following details:

- Most rented movies.
- Least rented movies.
- Average number that each movie is rented out in a month.

We can easily produce these details with the help of aggregate functions.

Let us discuss the most commonly used aggregate functions in detail. First, we will create a new table for the demonstration of all aggregate functions.

Execute the below statement to create an **employee** table:

```
CREATE TABLE employee(  
  name varchar(45) NOT NULL,  
  occupation varchar(35) NOT NULL,  
  working_date date,  
  working_hours varchar(10)  
);
```

Execute the below statement to **insert the records** into the employee table:

```
INSERT INTO employee VALUES  
( 'Robin', 'Scientist', '2020-10-04', 12),  
( 'Warner', 'Engineer', '2020-10-04', 10),  
( 'Peter', 'Actor', '2020-10-04', 13),  
( 'Marco', 'Doctor', '2020-10-04', 14),  
( 'Brayden', 'Teacher', '2020-10-04', 12),  
( 'Antonio', 'Business', '2020-10-04', 11);
```

Now, execute the **SELECT statement**

to show the record:

```
MySQL 8.0 Command Line Client
mysql> SELECT * FROM employee;
+-----+-----+-----+-----+
| name   | occupation | working_date | working_hours |
+-----+-----+-----+-----+
| Robin  | Scientist  | 2020-10-04   | 12             |
| Warner | Engineer   | 2020-10-04   | 10             |
| Peter  | Actor      | 2020-10-04   | 13             |
| Marco  | Doctor     | 2020-10-04   | 14             |
| Brayden| Teacher    | 2020-10-04   | 12             |
| Antonio| Business   | 2020-10-04   | 11             |
+-----+-----+-----+-----+
6 rows in set (0.00 sec)
```

Count() Function

MySQL count() function **returns the total number of values** in the expression. This function produces all rows or only some rows of the table based on a specified condition, and its return type is **BIGINT**. It returns zero if it does not find any matching rows. It can work with both numeric and non-numeric data types.

Example

Suppose we want to get the total number of employees in the employee table, we need to use the count() function as shown in the following query:

```
mysql> SELECT COUNT(name) FROM employee;
```

Output:

After execution, we can see that this table has six employees.

```
MySQL 8.0 Command Line Client
mysql> SELECT COUNT(name) FROM employee;
+-----+
| COUNT(name) |
+-----+
|          6  |
+-----+
1 row in set (0.15 sec)
```

To read more information, [click here](#)

Sum() Function

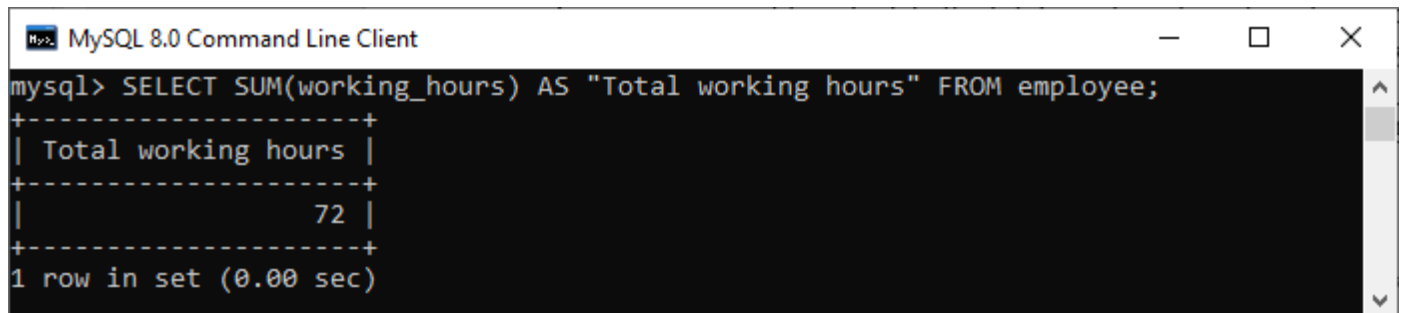
The MySQL sum() function **returns the total summed (non-NULL) value** of an expression. It returns NULL if the result set does not have any rows. It works with numeric data type only.

Suppose we want to calculate the total number of working hours of all employees in the table, we need to use the sum() function as shown in the following query:

```
mysql> SELECT SUM(working_hours) AS "Total working hours" FROM employee;
```

Output:

After execution, we can see the total working hours of all employees in the table.



```
mysql> SELECT SUM(working_hours) AS "Total working hours" FROM employee;
+-----+
| Total working hours |
+-----+
|                72 |
+-----+
1 row in set (0.00 sec)
```

To read more information, [click here](#)

AVG() Function

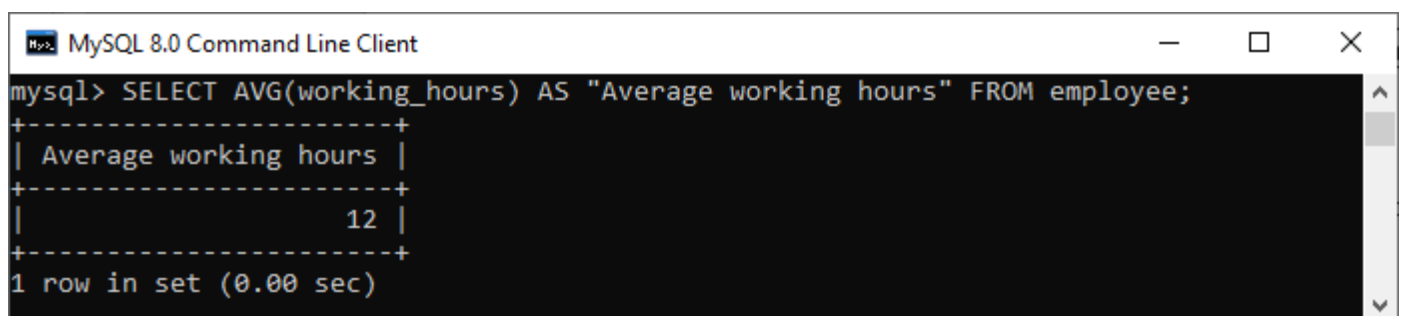
MySQL AVG() function **calculates the average of the values** specified in the column. Similar to the SUM() function, it also works with numeric data type only.

Suppose we want to get the average working hours of all employees in the table, we need to use the AVG() function as shown in the following query:

```
mysql> SELECT AVG(working_hours) AS "Average working hours" FROM employee;
```

Output:

After execution, we can see that the average working hours of all employees in the organization:



```
mysql> SELECT AVG(working_hours) AS "Average working hours" FROM employee;
+-----+
| Average working hours |
+-----+
|                12 |
+-----+
1 row in set (0.00 sec)
```

To read more information, [click here](#)

MIN() Function

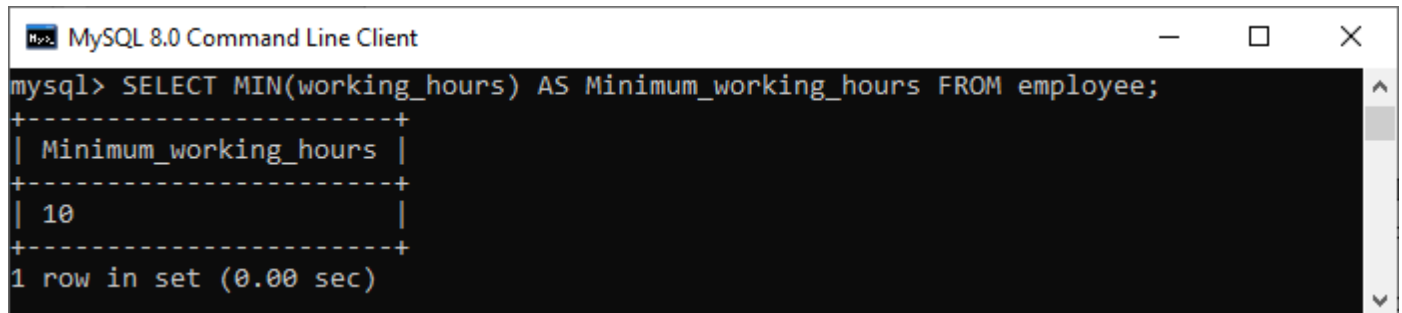
MySQL MIN() function **returns the minimum (lowest) value** of the specified column. It also works with numeric data type only.

Suppose we want to get minimum working hours of an employee available in the table, we need to use the MIN() function as shown in the following query:

```
mysql> SELECT MIN(working_hours) AS Minimum_working_hours FROM employee;
```

Output:

After execution, we can see that the minimum working hours of an employee available in the table:



```
mysql> SELECT MIN(working_hours) AS Minimum_working_hours FROM employee;
+-----+
| Minimum_working_hours |
+-----+
| 10                     |
+-----+
1 row in set (0.00 sec)
```

To read more information, [click here](#)

MAX() Function

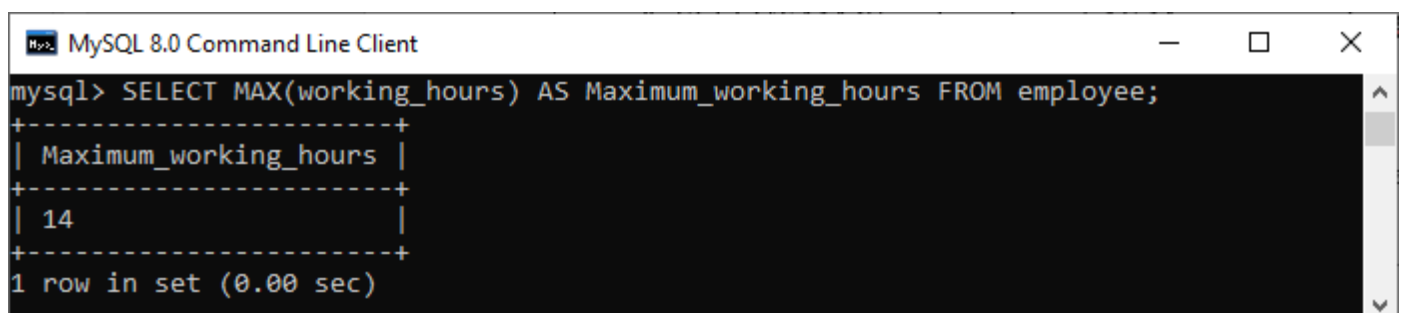
MySQL MAX() function **returns the maximum (highest) value** of the specified column. It also works with numeric data type only.

Suppose we want to get maximum working hours of an employee available in the table, we need to use the MAX() function as shown in the following query:

```
mysql> SELECT MAX(working_hours) AS Maximum_working_hours FROM employee;
```

Output:

After execution, we can see that the maximum working hours of an employee available in the table:



```
mysql> SELECT MAX(working_hours) AS Maximum_working_hours FROM employee;
+-----+
| Maximum_working_hours |
+-----+
| 14                     |
+-----+
1 row in set (0.00 sec)
```