# **MySQL Aggregate Functions**

**Summary**: in this tutorial, you will learn about MySQL aggregate functions including AVG, COUNT, SUM, MAX and MIN.

## **Introduction to MySQL aggregate functions**

An aggregate function performs a calculation on multiple values and returns a single value.

For example, you can use the AVG() aggregate function that takes multiple numbers and returns the average value of the numbers.

The following illustrates the syntax of an aggregate function:

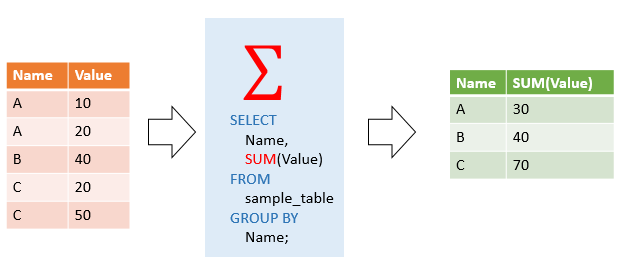


In this syntax:

* First, specify the name of the aggregate function e.g., AVG(). See the list of aggregate functions in the following section.
* Second, use DISTINCT if you want to calculate based on distinct values or ALL in case you want to calculate all values including duplicates. The default is ALL.
* Third, specify an expression that can be a column or an expression that involves column and arithmetic operators.

The aggregate functions are often used with the [GROUP BY](https://www.mysqltutorial.org/mysql-basics/mysql-group-by/) clause to calculate an aggregate value for each group e.g., the average value by the group or the sum of values in each group.

The following picture illustrates the SUM() aggregate function is used in conjunction with a GROUP BY clause:

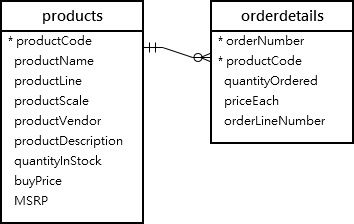


MySQL supports the following aggregate functions:

| **Aggregate function** | **Description** |
| --- | --- |
| [AVG()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-avg/) | Return the summation of all non-NULL values in a set. |
| [BIT\_AND()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-bit_and/) | Perform a bitwise AND of values in a column of a table. |
| [BIT\_OR()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-bit_or/) | Perform a bitwise OR of values in a column of a table. |
| [BIT\_XOR()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-bit_xor/) | Perform a bitwise XOR of values in a column of a table. |
| [COUNT()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-count/) | Return the number of rows in a group, including rows with NULL values. |
| [COUNT(DISTINCT)](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-count-distinct/) | Count the number of unique values of a column in a table. |
| [COUNT(IF)](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-count-if/) | Count the number of values that meet a specified condition. |
| [GROUP\_CONCAT()](https://www.mysqltutorial.org/mysql-group_concat/) | Return a concatenated string. |
| [JSON\_ARRAYAGG()](https://www.mysqltutorial.org/mysql-json/mysql-json_arrayagg/) | Return result set as a single JSON array. |
| [JSON\_OBJECTAGG()](https://www.mysqltutorial.org/mysql-json/mysql-json_objectagg/) | Return result set as a single JSON object. |
| [MAX()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-max-function/) | Return the highest value (maximum) in a set of non-NULL values. |
| [MIN()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-min/) | Return the lowest value (minimum) in a set of non-NULL values. |
| [STDEV()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-standard-deviation/) | Return the summation of all non-NULL values in a set. |
| [STDDEV\_POP()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-standard-deviation/) | Return the population standard deviation. |
| [STDDEV\_SAMP()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-standard-deviation/) | Return the sample standard deviation. |
| [SUM()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-sum/) | Return the summation of all non-NULL values a set. |
| [SUM(IF)](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-sum-if/) | Perform conditional summation using the SUM and IF functions. |
| [VAR\_POP()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-variance/) | Return the summation of all non-NULL values in a set. |
| [VAR\_SAMP()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-var_samp/) | Return the sample variance of values in a column of a table. |
| [VARIANCE()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-variance/) | Return the population standard variance of all non-NULL values in a set. |

## **MySQL aggregate function examples**

We will use the products and orderdetails tables from the [sample database](https://www.mysqltutorial.org/getting-started-with-mysql/mysql-sample-database/) for demonstration:

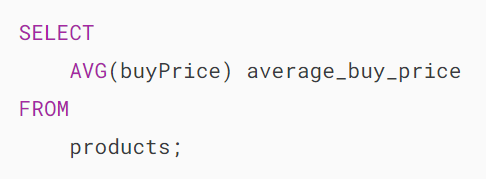


### **The AVG() function examples**

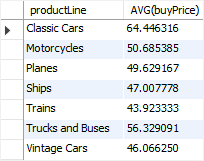
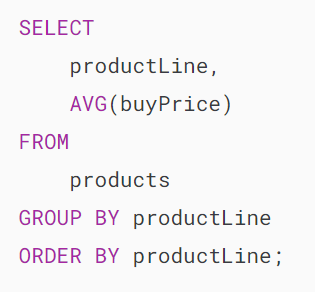
The [AVG()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-avg/) function calculates the average value of a set of values. It ignores NULL in the calculation.



For example, you can use the AVG function to calculate the average buy price of all products in the products table by using the following query:

MySQL Aggregate Function - AVG example

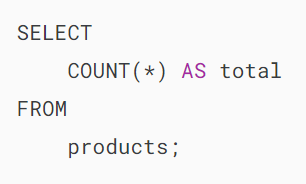
The following example uses the AVG() function to calculate the average buy price for each product line:



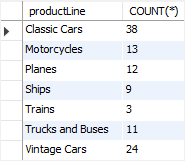
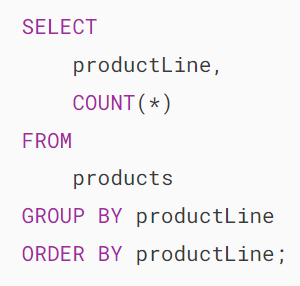
### **The COUNT() function examples**

The [COUNT()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-count/) function returns the number of the values in a set.

For example, you can use the COUNT() function to get the number of products in the products table as shown in the following query:

MySQL Aggregate Function - COUNT example

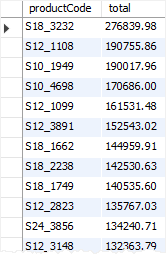
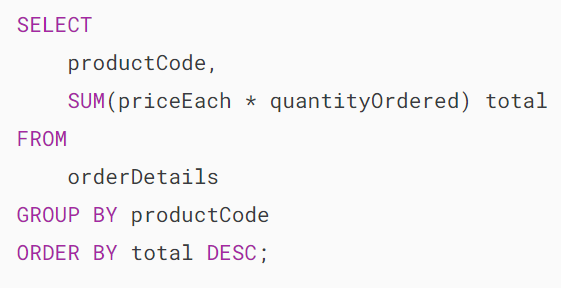
The following statement uses the COUNT() function with the GROUP BY clause to get the number of products for each product line:



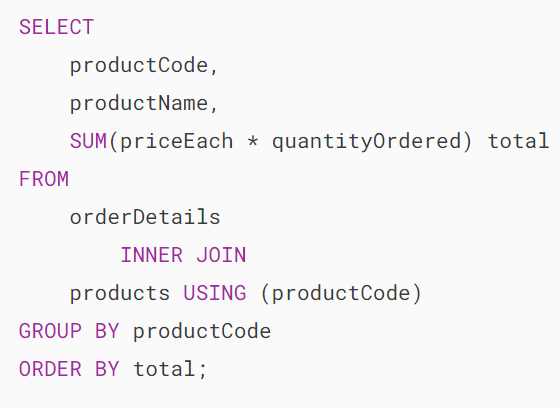
### **The SUM() function examples**

The [SUM()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-sum/) function returns the sum of values in a set. The SUM() function ignores NULL. If no matching row is found, the SUM() function returns NULL.

To get the total order value of each product, you can use the SUM() function in conjunction with the GROUP BY clause as follows:

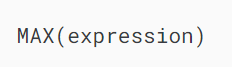


To see the result in more detail, you can [join](https://www.mysqltutorial.org/mysql-basics/mysql-join/) the orderdetails table to the products table as shown in the following query:

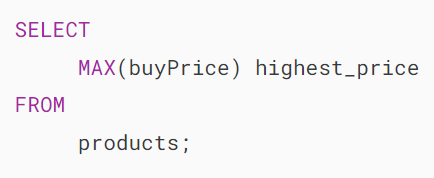


### **The MAX() function examples**

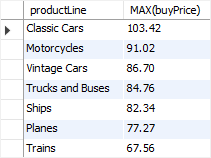
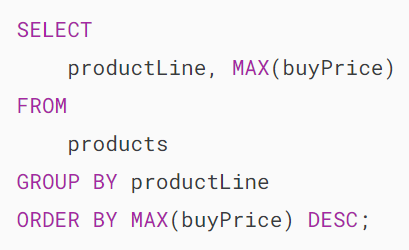
The [MAX()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-max-function/) function returns the maximum value in a set.



For example, you can use the MAX() function to get the highest buy price from the products table as shown in the following query:

MySQL Aggregate Function - MAX example

The following statement uses the MAX() function with the GROUP BY clause to get the highest price per product line:

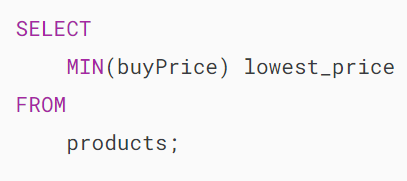


### **The MIN() function examples**

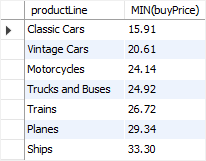
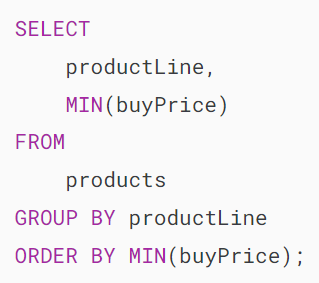
The [MIN()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-min/) function returns the minimum value in a set of values.



For example, the following query uses the MIN() function to find the lowest price from the products table:

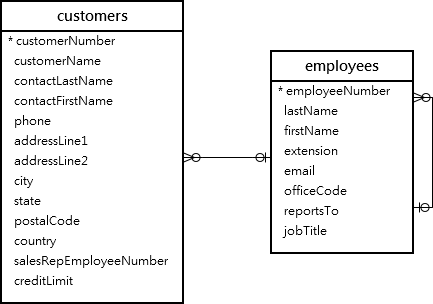
MySQL Aggregate Function - MIN example

The following example uses the MIN() function with the GROUP BY clause to get the lowest price per product line:

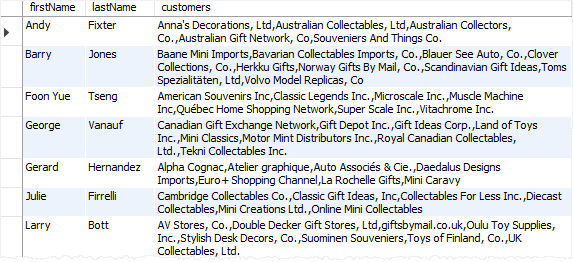
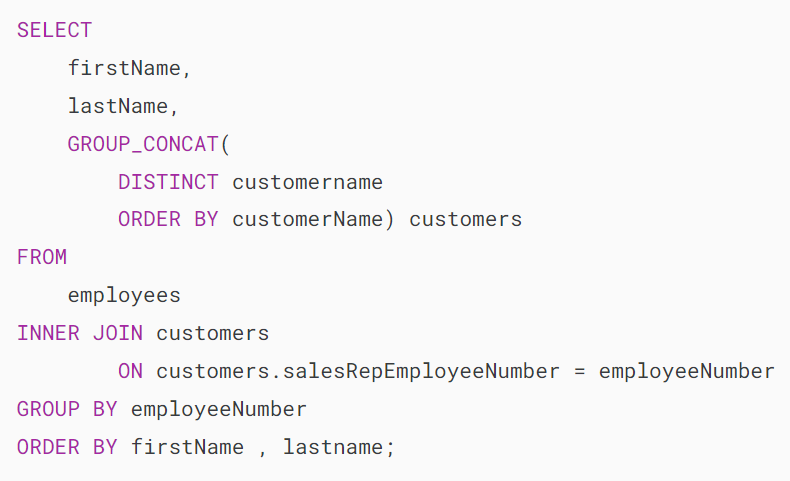


### **The GROUP\_CONCAT() function example**

The [GROUP\_CONCAT()](https://www.mysqltutorial.org/mysql-group_concat/) concatenates a set of strings and returns the concatenated string. See the following employees and customers tables:



The following statement uses the GROUP\_CONCAT() function to return the sales staff and list of customers that each sales staff is in charge of:



In this tutorial, you have learned how to use the most commonly used MySQL aggregate functions.

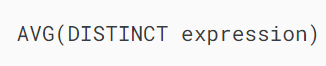
# **MySQL AVG() Function**

**Summary**: in this tutorial, you will learn how to use MySQL AVG() function to calculate the average value of a set of values.

## **Introduction to MySQL AVG() function**

The MySQL AVG() function is an [aggregate function](https://www.mysqltutorial.org/mysql-aggregate-functions/) that allows you to calculate the average value of a set of values.

Here’s the syntax of the AVG() function:

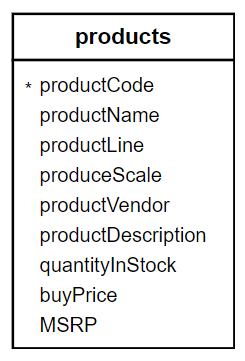


You use the [DISTINCT](https://www.mysqltutorial.org/mysql-basics/mysql-distinct/) operator in the AVG function to calculate the average value of the distinct values.

For example, if you have a set of values 1,1,2,3, the AVG function with DISTINCT operator will return 2 i.e., (1 + 2 + 3) / 3.

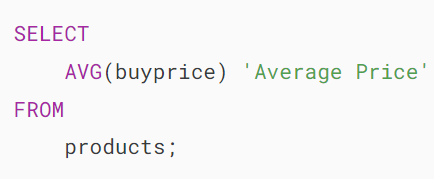
## **MySQL AVG() function examples**

We will use the products table in the [sample database](https://www.mysqltutorial.org/getting-started-with-mysql/mysql-sample-database/) for the demonstration:



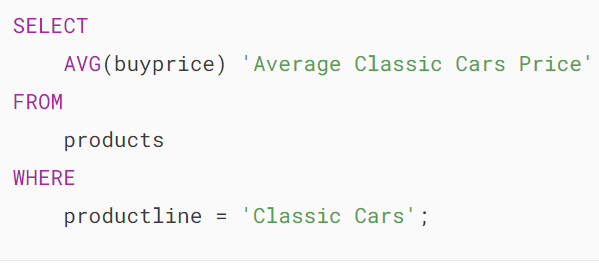
### **1) Using MySQL AVG() function to calculate an average of all values in a column example**

This example uses the AVG() function to calculate the average buy price of all products from the products table:

MySQL AVG function - average product price

### **2) Using MySQL AVG() function with a WHERE clause example**

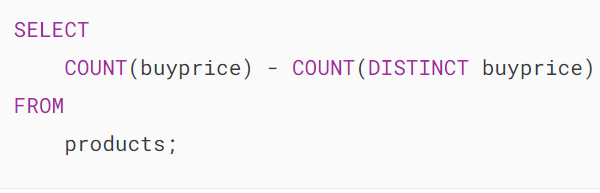
The following  example uses the AVG() function to calculate the average buy price of products in the product line Classic Cars:

**MySQL AVG function - Average Classic Cars Price**

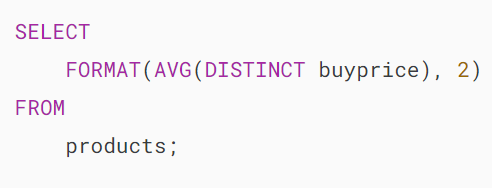
In this example, the WHERE clause has a condition that includes only the Classic Cars product line. Therefore, the AVG() function calculates the average value for the buy prices of products in Classic Cars only.

### **3) Using MySQL AVG with DISTINCT option example**

This query checks if there are any products which have the same prices:

MySQL AVG function - COUNT function

This query uses the AVG() function with the [DISTINCT](https://www.mysqltutorial.org/mysql-basics/mysql-distinct/) option to calculate the average of distinct buy prices:

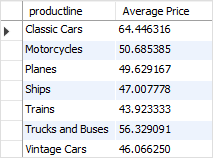
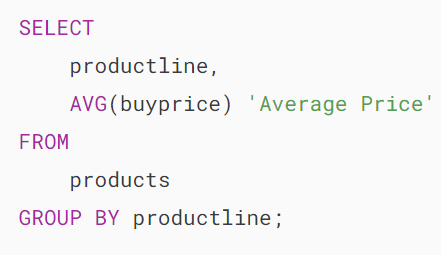
MySQL AVG function - average distinct product prices

Notice that the result is different from the average buy price without using the DISTINCT operator.

### **4) MySQL AVG with GROUP BY clause example**

The AVG() function is often used in conjunction with the [GROUP BY](https://www.mysqltutorial.org/mysql-basics/mysql-group-by/) clause to calculate the average value for each group of rows in a table.

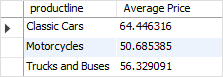
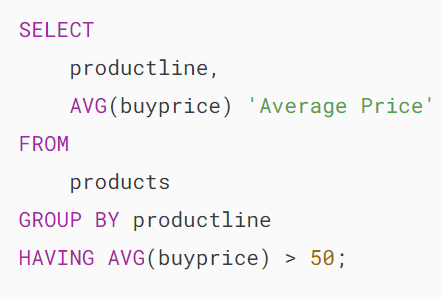
For example, to calculate the average buy price of products for each product line, you use the AVG() function with the GROUP BY clause as the following query:



### **5) Using MySQL AVG() function with a HAVING clause example**

You can use the AVG() function in the [HAVING](https://www.mysqltutorial.org/mysql-basics/mysql-having/) clause to set conditions for the average values of groups.

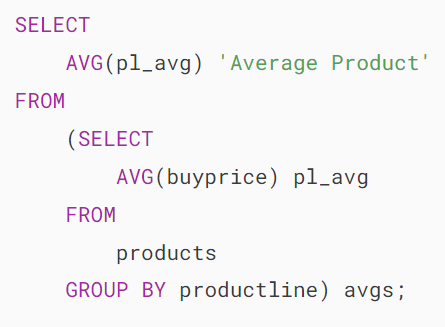
For example, if you want to select only product lines that have the product’s average buy prices greater than 50, you can use the following query:



### **6) Using MySQL AVG() function with a subquery example**

You can use the AVG() function in an SQL statement multiple times to calculate the average value of a set of average values.

This query uses the AVG() function to calculate the average buy price of the average buy prices of product lines:

MySQL AVG function - with subquery example

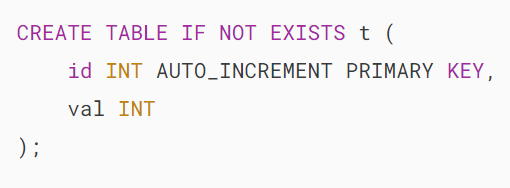
How it works.

* The [subquery](https://www.mysqltutorial.org/mysql-basics/mysql-subquery/)calculates the average buy price by product lines.
* The outer query calculates the average buy price of the average buy prices of product lines returned from the subquery.

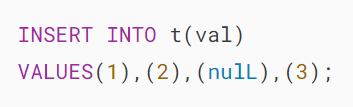
### **7) Using MySQL AVG() function with NULL example**

The AVG() function ignores NULL values in the calculation. See the following example:

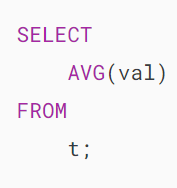
First, [create a new table](https://www.mysqltutorial.org/mysql-basics/mysql-create-table/) named t with two columns id and val. The val column can contain NULL values.



Second, [insert](https://www.mysqltutorial.org/mysql-basics/mysql-insert/)some rows into the t table, including NULL value.



Third, calculate the average value of the values in the val column by using the AVG function:

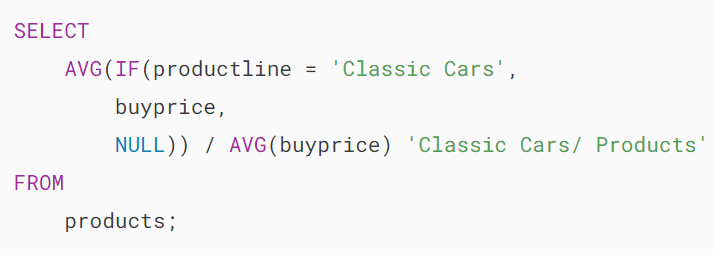
MySQL AVG function - NULL example

The statement returns 2 as expected because the NULL value is not included in the calculation of the AVG function.

### **8) Using MySQL AVG() function with control flow functions**

To calculate the average value of a column and calculate the average value of the same column conditionally in a single statement, you use AVG() function with [control flow functions](https://www.mysqltutorial.org/mysql-control-flow-functions/) e.g., [IF](https://www.mysqltutorial.org/mysql-control-flow-functions/mysql-if-function/), [CASE](https://www.mysqltutorial.org/mysql-control-flow-functions/mysql-case-function/), [IFNULL](https://www.mysqltutorial.org/mysql-control-flow-functions/mysql-ifnull/), and [NULLIF](https://www.mysqltutorial.org/mysql-control-flow-functions/mysql-nullif/).

For example, to calculate the ratio of the average buy price of Classic Cars product line to average buy price of all products, you use the following statement:

MySQL AVG function - with control flow function

The IF(productline='Classic Cars',buyprice,NULL) expression returns the buy price if the product line is Classic Cars, otherwise NULL.

Because the AVG() function ignores the NULL values in the calculation so the AVG(IF(productline='Classic Cars',buyprice,NULL)) expression returns the average buy price for only products whose product line is Classic Cars.

## **Summary**

* Use the AVG() function to calculate the average value of a set of values.
* Use the AVG() function with the GROUP BY clause to calculate the average value for each group.

# **MySQL COUNT() Function**

**Summary**: in this tutorial, you will learn how to use the MySQL COUNT() function to return the number of rows in a table.

## **Introduction to the MySQL COUNT() function**

The COUNT() function is an [aggregate function](https://www.mysqltutorial.org/mysql-aggregate-functions/) that returns the number of rows in a table. The COUNT() function allows you to count all rows or only rows that match a specified condition.

The COUNT() function has three forms:

* COUNT(\*)
* COUNT(expression)
* COUNT(DISTINCT expression)

### **COUNT(\*) function**

The COUNT(\*) function returns the number of rows in a result set returned by a [SELECT](https://www.mysqltutorial.org/mysql-basics/mysql-select-from/) statement. The COUNT(\*) returns the number of rows including duplicate, non-NULL and NULL rows.

### **COUNT(expression)**

The COUNT(expression) returns the number of rows that do not contain NULL values as the result of the expression.

### **COUNT(DISTINCT expression)**

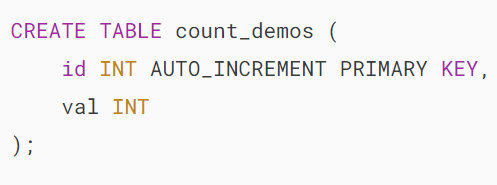
The COUNT(DISTINCT expression) returns the number of distinct rows that do not contain NULL values as the result of the expression.

The return type of the COUNT() function is [BIGINT](https://www.mysqltutorial.org/mysql-basics/mysql-int/). The COUNT()  function returns 0 if there is no matching row found.

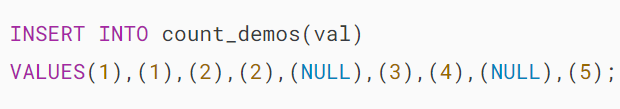
## **MySQL COUNT() function illustration**

### **Setting up a sample table**

First, [create a table](https://www.mysqltutorial.org/mysql-basics/mysql-create-table/) called count\_demos:



Second, [insert some rows](https://www.mysqltutorial.org/mysql-basics/mysql-insert-multiple-rows/) into the count\_demos table:

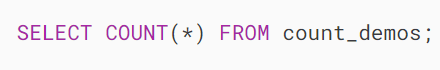


Third, [query data](https://www.mysqltutorial.org/mysql-basics/mysql-select-from/) from the count\_demos table:

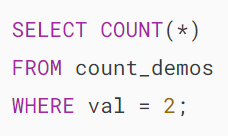
### 

### **MySQL COUNT(\*) example**

The following statement uses the COUNT(\*) function to return all rows from the count\_demos table:

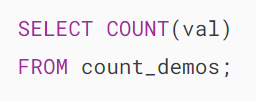
MySQL COUNT star example

This example uses the COUNT(\*) function with a [WHERE](https://www.mysqltutorial.org/mysql-basics/mysql-where/) clause to specify a condition to count only rows whose value in the column val is 2:

MySQL COUNT star with a WHERE clause

### **MySQL COUNT(expression) example**

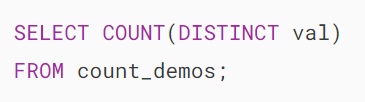
If you specify the val column in the COUNT() function, the COUNT() function will count only rows with non-NULL values in the val column:



Notice that two NULL values are not counted.

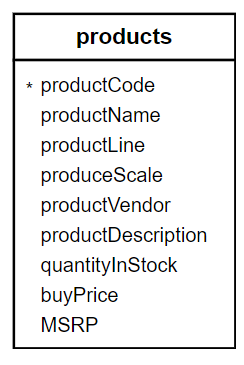
### **MySQL COUNT(DISTINCT expression) example**

This example uses COUNT(DISTINCT expression) to count non-NULL and distinct values in the column val:

MySQL COUNT DISTINCT expr example

## **MySQL COUNT() function practical examples**

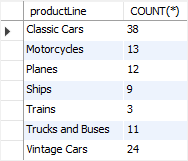
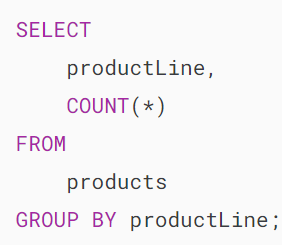
We’ll use the products table from the [sample database](https://www.mysqltutorial.org/getting-started-with-mysql/mysql-sample-database/) for the next examples:



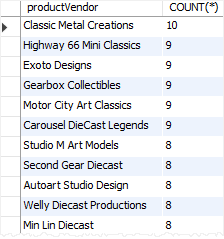
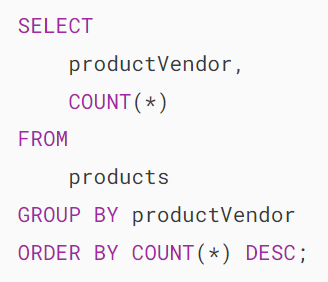
### **1) Using MySQL COUNT(\*) function with a GROUP BY example**

The COUNT(\*) function is often used with a [GROUP BY](https://www.mysqltutorial.org/mysql-basics/mysql-group-by/) clause to return the number of elements in each group.

For example, this statement uses the COUNT() function with the [GROUP BY](https://www.mysqltutorial.org/mysql-basics/mysql-group-by/) clause to return the number of products in each product line:

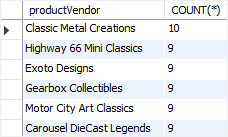
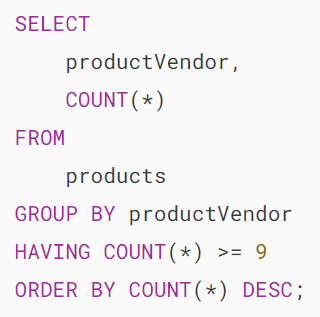


Similarly, this example uses the COUNT(\*) function to find the number of products supplied by each vendor:



### **2) Using MySQL COUNT(\*) with a HAVING clause example**

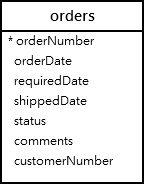
To find vendors who supply at least 9 products, you use the COUNT(\*) function in the [HAVING](https://www.mysqltutorial.org/mysql-basics/mysql-having/) clause as shown in the following query:



### **3) MySQL COUNT IF example**

You can use a control flow expression and functions e.g., [IF](https://www.mysqltutorial.org/mysql-control-flow-functions/mysql-if-function/), [IFNULL](https://www.mysqltutorial.org/mysql-control-flow-functions/mysql-ifnull/), and [CASE](https://www.mysqltutorial.org/mysql-control-flow-functions/mysql-case-function/) in the COUNT() function to count rows whose values match a condition.

See the following orders table from the sample database:



The following query use COUNT() with IF function to find the number of canceled, on hold, and disputed orders from the orders table:



The IF() function returns 1 if the order’s status is canceled, on hold, or disputed, otherwise, it returns NULL.

The COUNT function only counts 1, not NULL values, therefore, the query returns the number of orders based on the corresponding status.

MySQL COUNT IF example

# **MySQL COUNT DISTINCT**

**Summary**: in this tutorial, you will learn how to use the MySQL COUNT DISTINCT function to count the number of unique values in a specific column of a table.

## **Introduction to MySQL COUNT DISTINCT function**

The COUNT DISTINCT allows you to [count](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-count/) the number of unique values in a specific column of a table.

Here’s the basic syntax for using the COUNT DISTINCT function:

SELECT COUNT(DISTINCT column\_name)

FROM table\_name;Code language: SQL (Structured Query Language) (sql)

In this syntax:

* COUNT DISTINCT: The function for counting unique values.
* column\_name: The name of the column for which you want to count distinct values.
* table\_name: The name of the table that contains the column\_name.

In practice, you use the COUNT DISTINCT when you want to find out how many unique values a present in a column.

## **MySQL COUNT DISTINCT examples**

Let’s take some examples of using the COUNT DISTINCT function. We’ll use the employees table from the [sample database](https://www.mysqltutorial.org/getting-started-with-mysql/mysql-sample-database/) for the demonstration:

The following query retrieves first name, last name, and job title from the employees table:

SELECT firstName, lastName, jobTitle

FROM employees;Code language: SQL (Structured Query Language) (sql)

Output:

+*-----------+-----------+----------------------+*

| firstName | lastName | jobTitle |

+*-----------+-----------+----------------------+*

| Diane | Murphy | President |

| Mary | Patterson | VP Sales |

| Jeff | Firrelli | VP Marketing |

| William | Patterson | Sales Manager (APAC) |

| Gerard | Bondur | Sale Manager (EMEA) |

| Anthony | Bow | Sales Manager (NA) |

| Leslie | Jennings | Sales Rep |

| Leslie | Thompson | Sales Rep |

| Julie | Firrelli | Sales Rep |

| Steve | Patterson | Sales Rep |

| Foon Yue | Tseng | Sales Rep |

| George | Vanauf | Sales Rep |

| Loui | Bondur | Sales Rep |

| Gerard | Hernandez | Sales Rep |

| Pamela | Castillo | Sales Rep |

| Larry | Bott | Sales Rep |

| Barry | Jones | Sales Rep |

| Andy | Fixter | Sales Rep |

| Peter | Marsh | Sales Rep |

| Tom | King | Sales Rep |

| Mami | Nishi | Sales Rep |

| Yoshimi | Kato | Sales Rep |

| Martin | Gerard | Sales Rep |

+*-----------+-----------+----------------------+*

23 rows in set (0.00 sec)Code language: SQL (Structured Query Language) (sql)

The output indicates that the employees table has 23 rows.

### **1) Using COUNT DISTINCT to get the number of unique job titles**

The following example uses the COUNT DISTINCT function to get the unique number of job titles from the jobTitle column of the employees table:

SELECT

COUNT(DISTINCT jobTitle)

from

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

In this query:

* COUNT( DISTINCT jobTitle): Counts the number of distinct values in the jobTitle column.
* FROM employees: Specifies the table from which to retrieve the data.

When you execute the query, MySQL will return the following output:

+*--------------------------+*

| count(distinct jobTitle) |

+*--------------------------+*

| 7 |

+*--------------------------+*Code language: SQL (Structured Query Language) (sql)

You can use a [column alias](https://www.mysqltutorial.org/mysql-basics/mysql-alias/) to assign a more meaningful name to the output column as follows:

SELECT

COUNT(DISTINCT jobTitle) uniqueJobTitleCount

from

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

Output:

+*---------------------+*

| uniqueJobTitleCount |

+*---------------------+*

| 7 |

+*---------------------+*

1 row in set (0.00 sec)Code language: SQL (Structured Query Language) (sql)

### **2) Using COUNT DISTINCT to get the number of unique first names**

The following example uses the COUNT DISTINCT to get the number of unique first names of employees:

SELECT

COUNT(DISTINCT firstName) uniqueFirstNameCount

from

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

Output:

+*----------------------+*

| uniqueFirstNameCount |

+*----------------------+*

| 21 |

+*----------------------+*

1 row in set (0.00 sec)Code language: SQL (Structured Query Language) (sql)

## **COUNT DISTINCT and NULL**

In MySQL, the COUNT DISTINCT function does not count NULL values. It only counts unique, non-null values in the specified column. In other words, if a column has null values, the COUNT DISTINCT function will ignore NULL values from the count.

Let’s take a look at the following example.

First, [create a new table](https://www.mysqltutorial.org/mysql-basics/mysql-create-table/) contacts that has two columns id and name:

CREATE TABLE contacts(

id INT AUTO\_INCREMENT,

name VARCHAR(50),

PRIMARY KEY(id)

);

Second, insert four rows into the contacts table:

INSERT INTO contacts(name)

VALUES ("John"), ("Jane"),(NULL), ("Jane");Code language: PHP (php)

Third, select data from the contacts table:

SELECT \* FROM contacts;

Output:

+----+------+

| id | name |

+----+------+

| 1 | John |

| 2 | Jane |

| 3 | NULL |

| 4 | Jane |

+----+------+

4 rows in set (0.00 sec)Code language: JavaScript (javascript)

Finally, count the number of unique names in the name column of the contacts table:

SELECT COUNT(DISTINCT name)

FROM contacts;

Output:

+----------------------+

| COUNT(DISTINCT name) |

+----------------------+

| 2 |

+----------------------+

1 row in set (0.00 sec)Code language: JavaScript (javascript)

In this example, the COUNT DISTINCT function ignores the NULL value and only counts unique non-null values.

## **Summary**

* Use MySQL COUNT DISTINCT function to count the number of unique values in a column of a table.
* The COUNT DISTINCT function ignores NULL values from the count.

https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-count-distinct/