# **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:

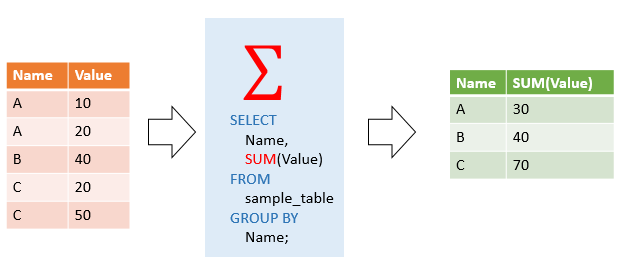


In this syntax:

* 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:

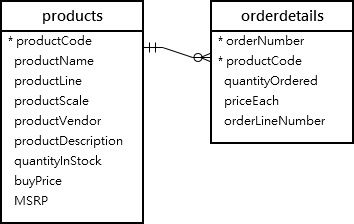


MySQL supports the following aggregate functions:

| **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:

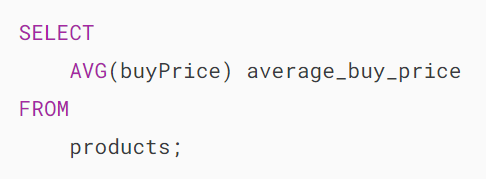


### **The AVG() function examples**

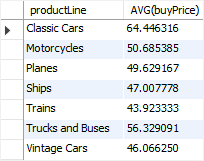
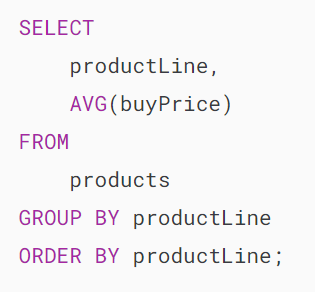
The [AVG()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-avg/) function calculates the average value of a set of values. It ignores NULL in the calculation.



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

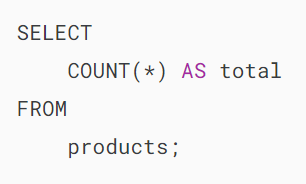
The following example uses the AVG() function to calculate the average buy price for each product line:



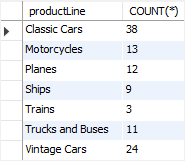
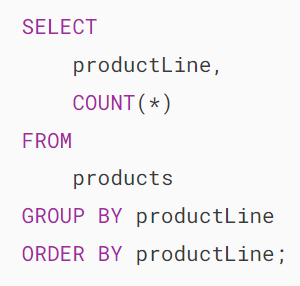
### **The COUNT() function examples**

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

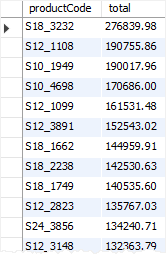
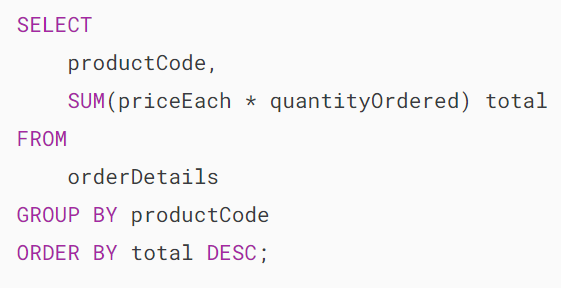
The following statement uses the COUNT() function with the GROUP BY clause to get the number of products for each product line:



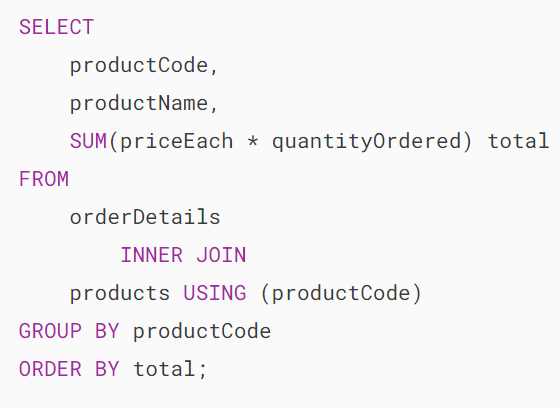
### **The SUM() function examples**

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:

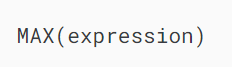


To see the result in more detail, you can [join](https://www.mysqltutorial.org/mysql-basics/mysql-join/) the orderdetails table to the products table as shown in the following query:

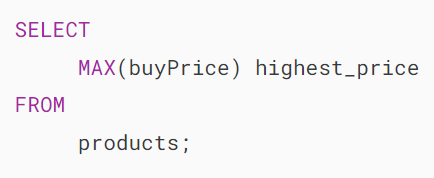


### **The MAX() function examples**

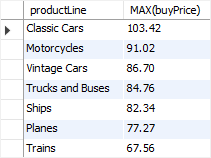
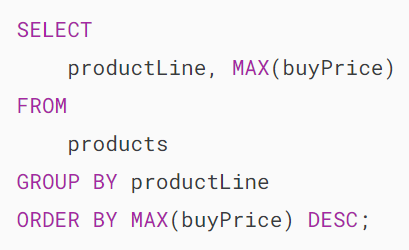
The [MAX()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-max-function/) function returns the maximum value in a set.



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:

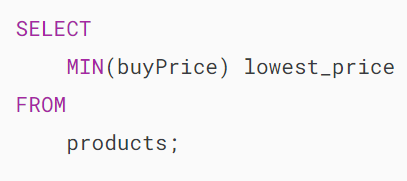


### **The MIN() function examples**

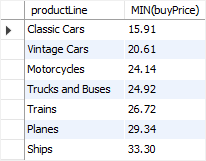
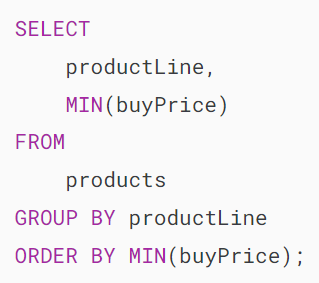
The [MIN()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-min/) function returns the minimum value in a set of values.



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

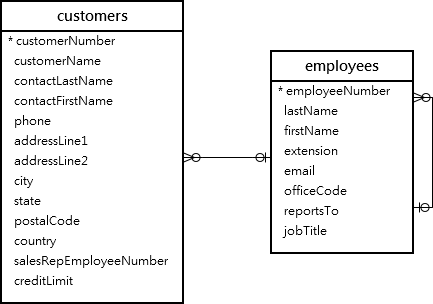
MySQL Aggregate Function - MIN example

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

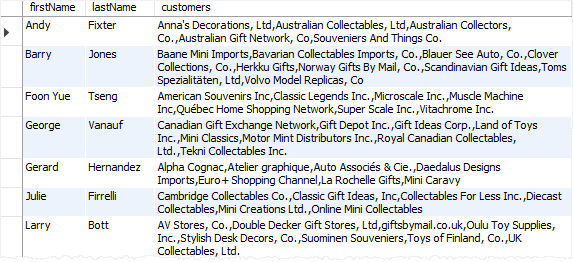
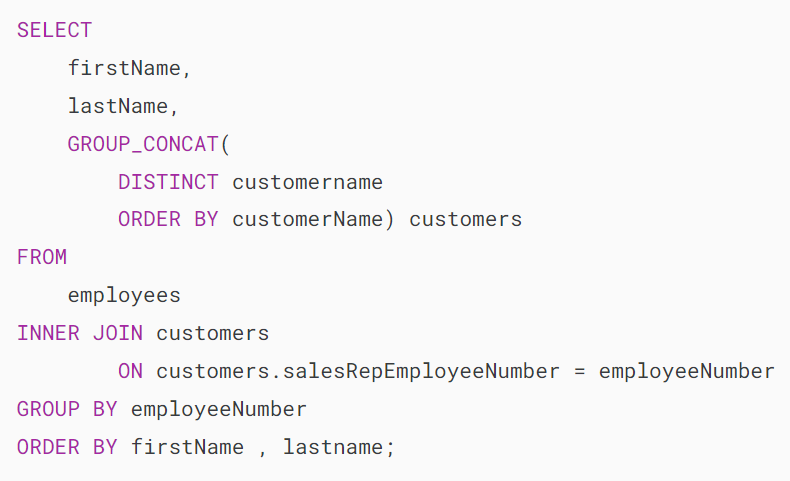


### **The GROUP\_CONCAT() function example**

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.

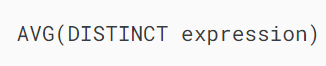
# **MySQL AVG() Function**

**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:

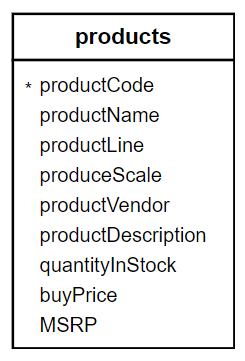


You use the [DISTINCT](https://www.mysqltutorial.org/mysql-basics/mysql-distinct/) operator in the AVG function to calculate the average value of the distinct values.

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.

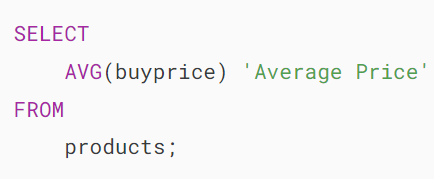
## **MySQL AVG() function examples**

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



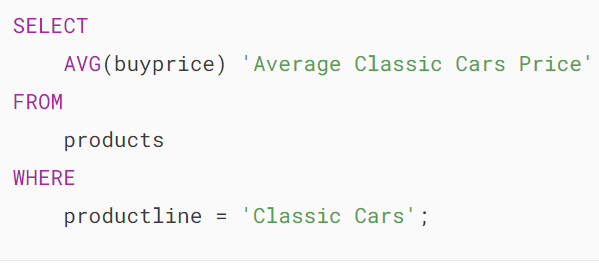
### **1) Using MySQL AVG() function to calculate an average of all values in a column example**

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**

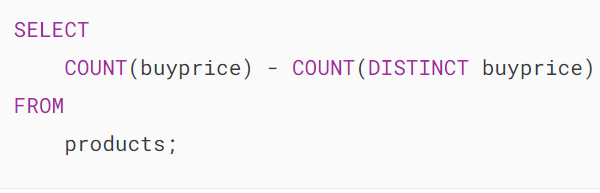
The following  example uses the AVG() function to calculate the average buy price of products in the product line Classic Cars:

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

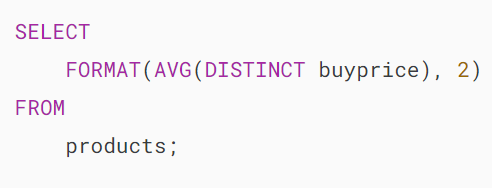
In this example, the WHERE clause has a condition that includes only the Classic Cars product line. Therefore, the AVG() function calculates the average value for the buy prices of products in Classic Cars only.

### **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:

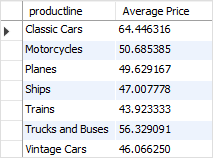
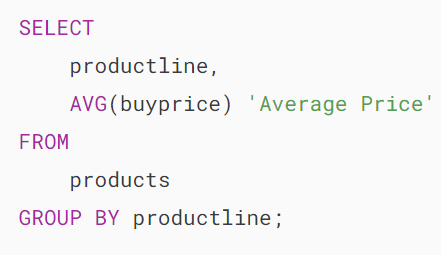
MySQL AVG function - average distinct product prices

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.

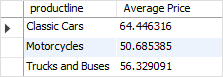
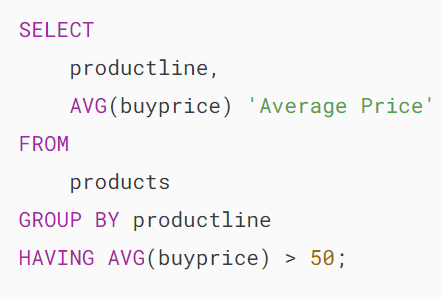
For example, to calculate the average buy price of products for each product line, you use the AVG() function with the GROUP BY clause as the following query:



### **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.

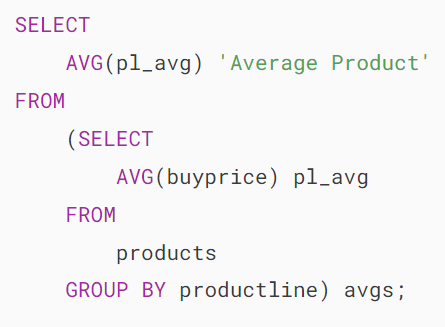
For example, if you want to select only product lines that have the product’s average buy prices greater than 50, you can use the following query:



### **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

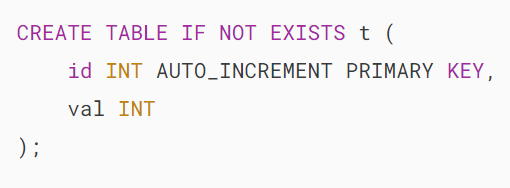
How it works.

* 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.

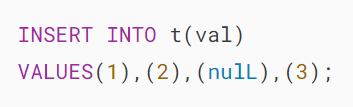
### **7) Using MySQL AVG() function with NULL example**

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

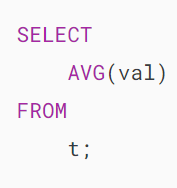
First, [create a new table](https://www.mysqltutorial.org/mysql-basics/mysql-create-table/) named t with two columns id and val. The val column can contain NULL values.



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:

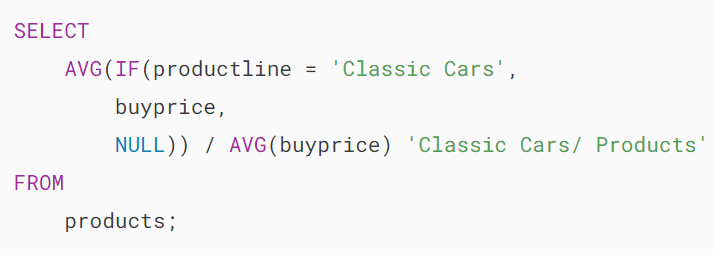
MySQL AVG function - NULL example

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)**

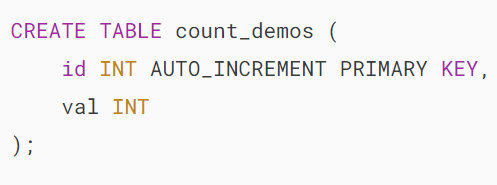
The COUNT(DISTINCT expression) returns the number of distinct rows that do not contain NULL values as the result of the expression.

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.

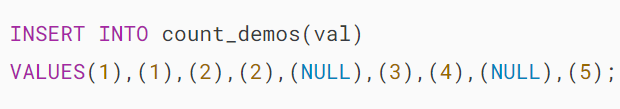
## **MySQL COUNT() function illustration**

### **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:

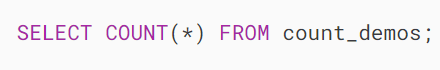


Third, [query data](https://www.mysqltutorial.org/mysql-basics/mysql-select-from/) from the count\_demos table:

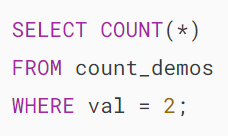
### 

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

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

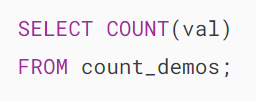
MySQL COUNT star example

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**

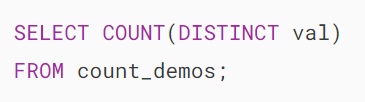
If you specify the val column in the COUNT() function, the COUNT() function will count only rows with non-NULL values in the val column:



Notice that two NULL values are not counted.

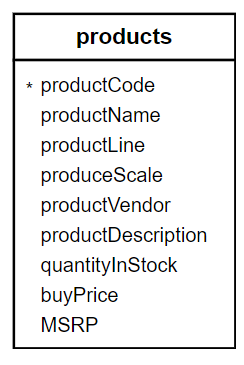
### **MySQL COUNT(DISTINCT expression) example**

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**

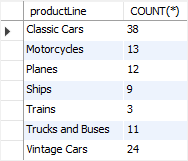
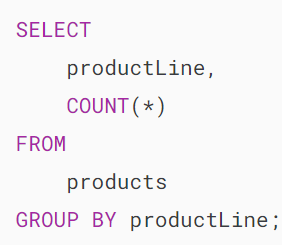
We’ll use the products table from the [sample database](https://www.mysqltutorial.org/getting-started-with-mysql/mysql-sample-database/) for the next examples:



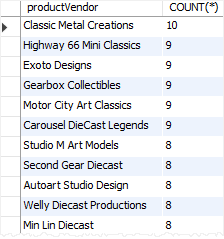
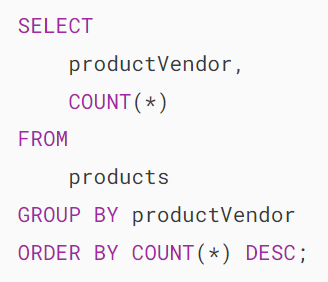
### **1) Using MySQL COUNT(\*) function with a GROUP BY example**

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:

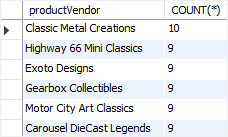
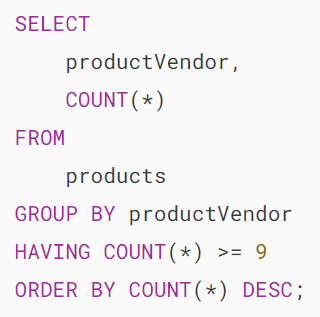


Similarly, this example uses the COUNT(\*) function to find the number of products supplied by each vendor:



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

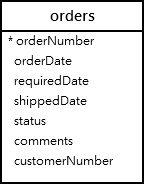
To find vendors who supply at least 9 products, you use the COUNT(\*) function in the [HAVING](https://www.mysqltutorial.org/mysql-basics/mysql-having/) clause as shown in the following query:



### **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

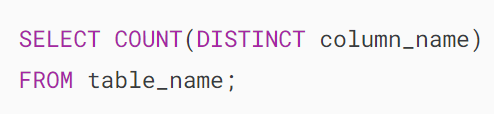
# **MySQL COUNT DISTINCT**

**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:



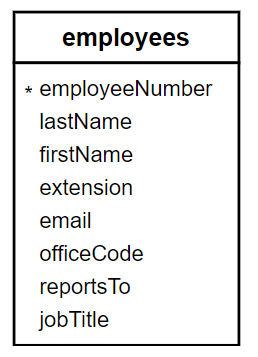
In this syntax:

* 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.

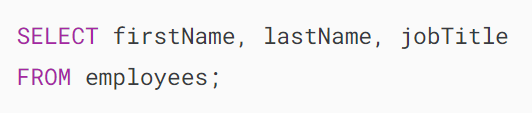
In practice, you use the COUNT DISTINCT when you want to find out how many unique values a present in a column.

## **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:



Output:



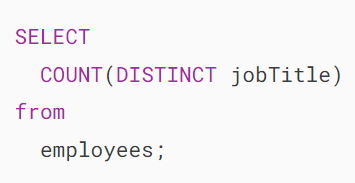
..



The output indicates that the employees table has 23 rows.

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

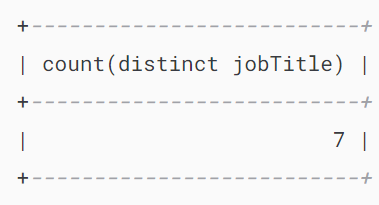
The following example uses the COUNT DISTINCT function to get the unique number of job titles from the jobTitle column of the employees table:



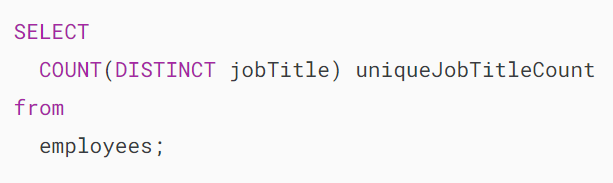
In this query:

* 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:



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:

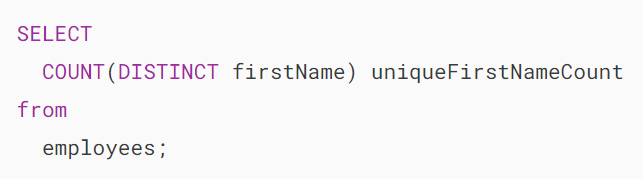


Output:

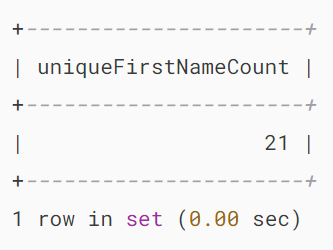
### 

### **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:



Output:

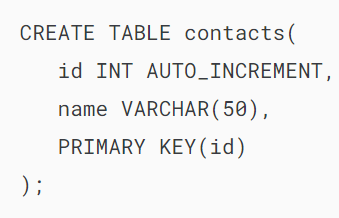


## **COUNT DISTINCT and NULL**

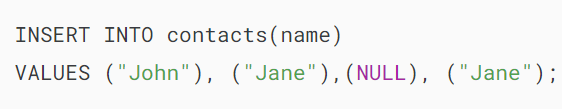
In MySQL, the COUNT DISTINCT function does not count NULL values. It only counts unique, non-null values in the specified column. In other words, if a column has null values, the COUNT DISTINCT function will ignore NULL values from the count.

Let’s take a look at the following example.

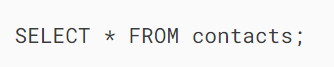
First, [create a new table](https://www.mysqltutorial.org/mysql-basics/mysql-create-table/) contacts that has two columns id and name:



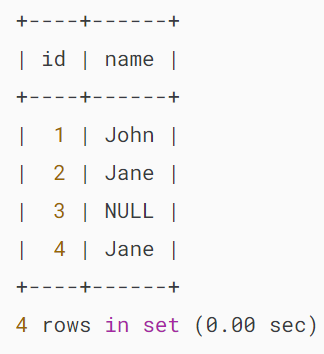
Second, insert four rows into the contacts table:



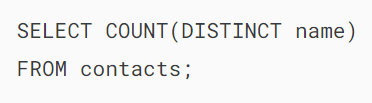
Third, select data from the contacts table:



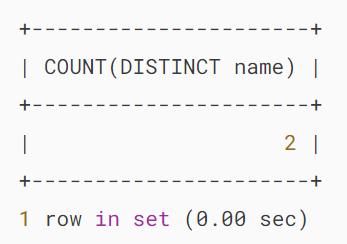
Output:



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



Output:



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.

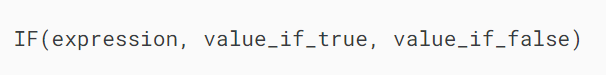
# **MySQL COUNT IF**

**Summary**: in this tutorial, you will learn how to use the MySQL COUNT IF to count values in a set based on a condition.

## **Introduction to MySQL COUNT IF function**

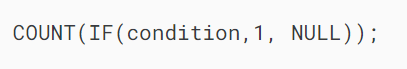
The [IF function](https://www.mysqltutorial.org/mysql-control-flow-functions/mysql-if-function/) evaluates an expression and returns a value depending on whether the result of the expression is true or false.

Here’s the syntax of the IF function:



In this syntax, the IF function returns value\_if\_true if the expression is true or value\_if\_false if the expression is false.

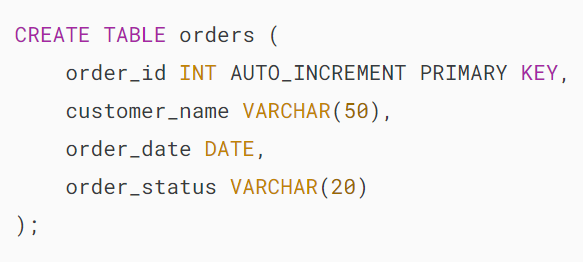
The [COUNT()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-count/) function is an [aggregate function](https://www.mysqltutorial.org/mysql-aggregate-functions/) that returns the number of non-null values in a set. Because the COUNT() function accepts an expression, you can use the IF to form an expression like this:



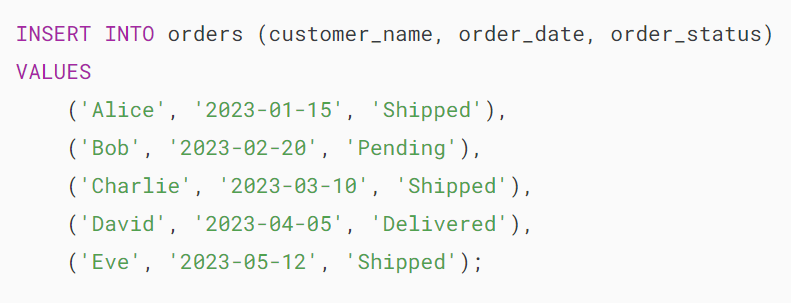
In this syntax, the COUNT() will return the number of values that make the condition true. Note that you can use another value other than the number 1.

## **MySQL COUNT IF example**

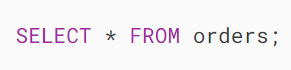
First, [create a new table](https://www.mysqltutorial.org/mysql-basics/mysql-create-table/) called orders with four columns order\_id, customer\_name, order\_date, and order\_status:



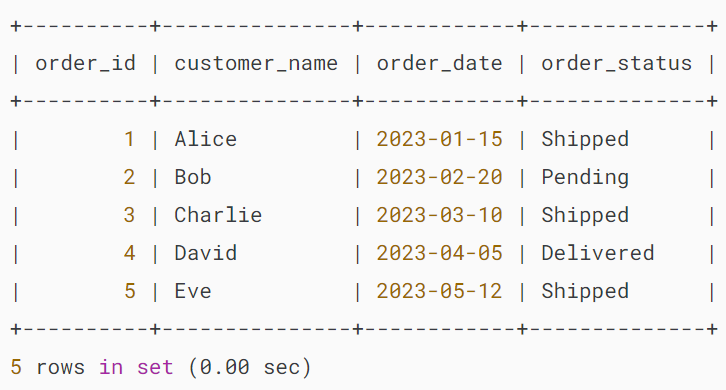
Second, [insert some rows](https://www.mysqltutorial.org/mysql-basics/mysql-insert/) into the orders table with different order statuses:



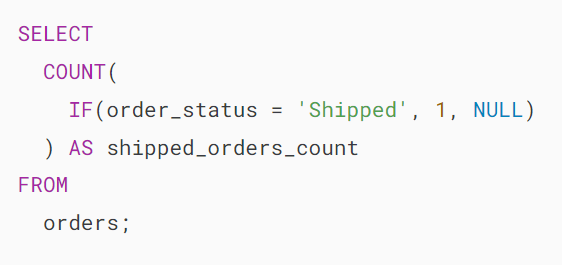
Third, query data from the orders table:



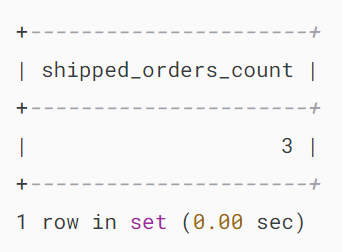
Output:



Finally, count the orders by status using the COUNT IF to count the number of shipped orders:



Output:

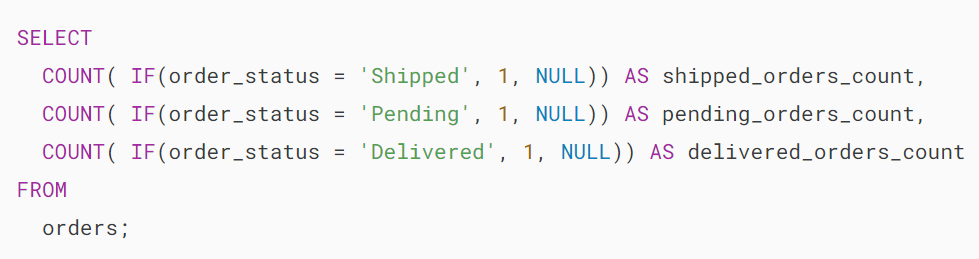
’

There are three shipped orders.

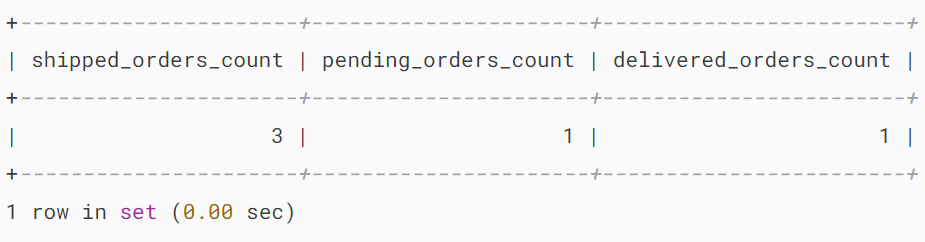
In this example:

* The IF(order\_status = 'Shipped', 1, NULL) returns 1 if the order\_status is shipped or NULL otherwise.
* The COUNT will count only 1 and ignore the NULL value. Therefore, it returns the number of shipped orders.

Similarly, you can use the COUNT IF to counter the number of orders by status:



Output:



## **Summary**

* Use MySQL COUNT IF to count a number of values based on a specific condition.

# **MySQL MAX() Function**

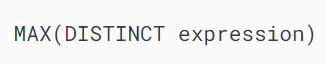
**Summary**: in this tutorial, you will learn how to use the MySQL MAX() function to get the maximum value in a set of values.

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

In MySQL, the MAX() function returns the maximum value in a set of values.

The MAX() function comes in handy in many cases such as finding the highest number, the most expensive product, or the largest payment from customers.

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



In this syntax:

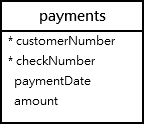
* expression: is a column or an expression that you want to find the maximum value.

If you use the DISTINCT operator, the MAX() function returns the maximum value of distinct values, which is the same as the maximum value of all values. It means that DISTINCT does not have any effects on the MAX() function.

Notice that DISTINCT has effects on other [aggregate functions](https://www.mysqltutorial.org/mysql-aggregate-functions/) such as [COUNT()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-count/), [SUM()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-sum/), and [AVG()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-avg/).

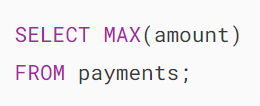
## **MySQL MAX() function examples**

We’ll use the payments table in the[sample database](https://www.mysqltutorial.org/getting-started-with-mysql/how-to-load-sample-database-into-mysql-database-server/) to demonstrate the MAX() function.



### **1) Using MySQL MAX() function to find the maximum value in a column example**

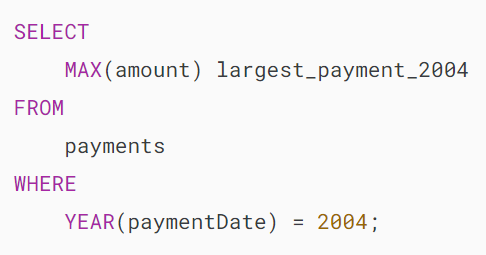
This example uses the MAX() function to return the largest amount of all payments:

MySQL MAX function example

In this example, the MAX() function checks all values in the amount column of the payments table to find the largest amount.

### **2) Using MySQL MAX() function with WHERE clause**

The following statement uses the MAX() function to find the largest payment in 2004:



In this example:

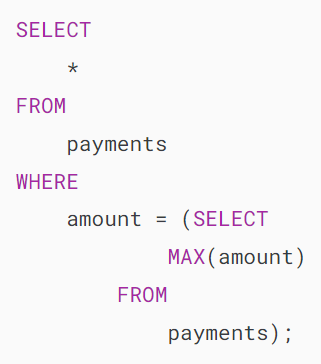
* First, use a condition in the [WHERE](https://www.mysqltutorial.org/mysql-basics/mysql-where/) clause to get only payments whose year is 2004. We used the [YEAR()](https://www.mysqltutorial.org/mysql-year/) function to extract the year from the payment date.
* Then, use the MAX() function in the SELECT clause to find the largest amount of payments in 2004.

This picture shows the output:

mysql max with where clause

### **3) Using MAX() function in subquery example**

To obtain not only the largest payment amount but also additional payment information, such as customer number, check number, and payment date, you can utilize the MAX() function within a subquery, as demonstrated in the following query:

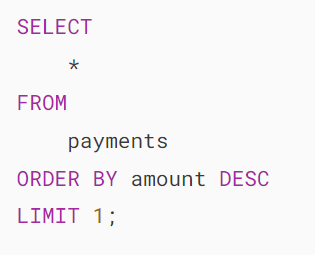




How it works.

* The subquery returns the largest amount of all payments.
* The outer query gets the payment whose amount is equal to the largest amount returned from the subquery and also other payment information.

Another way to do this without using the MAX() function is to sort the result set in descending order using the [ORDER BY](https://www.mysqltutorial.org/mysql-basics/mysql-order-by/) clause and get the first row in the result set using the [LIMIT](https://www.mysqltutorial.org/mysql-basics/mysql-limit/) clause as follows:



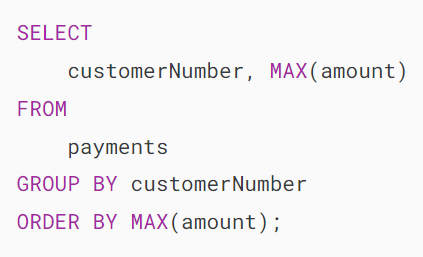
If you don’t have an [index](https://www.mysqltutorial.org/mysql-index/) on the amount column, the second query with the LIMIT clause is faster because it examines all rows in the payments table, while the first query examines all the rows in the payments table twice, first once in the subquery and another in the outer query.

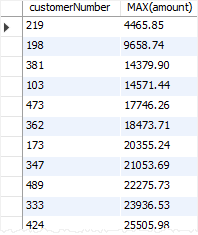
However, if the amount column is indexed, the first query executes faster.

### **4) Using MySQL MAX() with GROUP BY clause example**

To find the maximum value for every group, you use the MAX function with the [GROUP BY](https://www.mysqltutorial.org/mysql-basics/mysql-group-by/) clause.

This statement uses the MAX() to get the largest payment from each customer:





In this example:

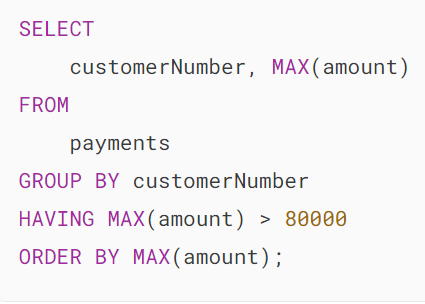
* First, the GROUP BY clause group payments into groups by customer number.
* Second, the MAX() function returns the largest payment in each group.

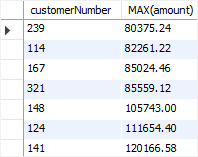
### **5) Using MySQL MAX() with HAVING clause**

When you use the MAX() function with the GROUP BY clause, you can find the maximum value for each group.

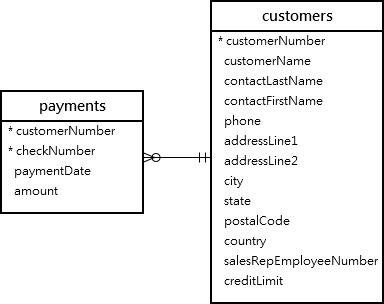
If you want to filter groups based on a condition, you can use the MAX() function in a HAVING clause.

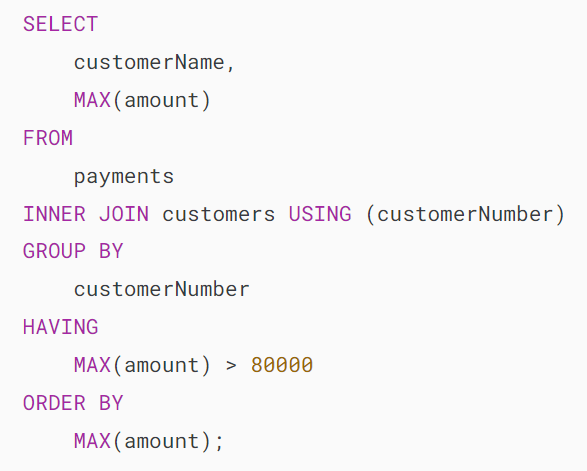
The following query finds the largest payment of each customer; and based on the returned payments, gets only payments whose amounts are greater than 80,000 .



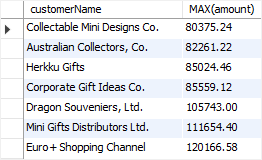


If you want to see the names of customers instead of numbers, you can [join](https://www.mysqltutorial.org/mysql-basics/mysql-join/) the payments table with the customers table:





Here is the output:



## **Summary**

* Use the MAX() function to find the maximum value in a set of values.

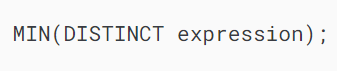
# **MySQL MIN() Function**

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

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

The MIN() function returns the minimum value in a set of values. The MIN() function is very useful in some scenarios such as finding the smallest number, selecting the least expensive product, or getting the lowest credit limit.

Here’s the basic syntax of the MIN() function:

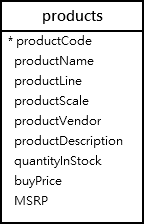


In this syntax, the MIN() function accepts an expression that can be a column or a valid expression that involves columns.

The DISTINCT does not have any effect on the MIN() function like other [aggregate functions](https://www.mysqltutorial.org/mysql-aggregate-functions/) such as [SUM()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-sum/), [AVG()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-avg/)and [COUNT()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-count/).

## **MySQL MIN function examples**

We’ll 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 MIN() function to find the minimum value in a column**

This query uses the MIN() function to get the lowest price of all products from the products table:

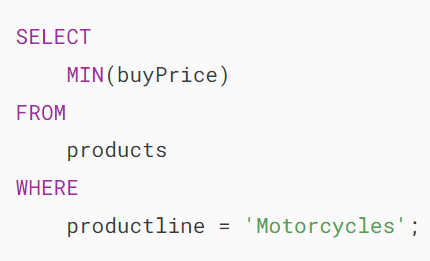


MySQL MIN Function Example

In this example, the query checks all values in the column buyPrice of the products table and returns the lowest value.

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

This example uses the [MIN()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-min/) function to find the lowest buy price of all motorcycles:



In this example:

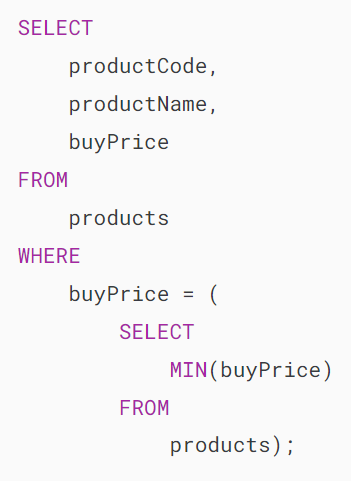
* First, specify a condition in the WHERE clause that gets only products whose product line is Motorcycles.
* Second, use the MIN() function to get the lowest value of the buy price of all motorcycles.

Here is the output:

MySQL MIN with a WHERE clause

### **3) Using MySQL MIN() with a subquery example**

To find not only the price but also other product information such as product code and product name, you use the MIN() function in a [subquery](https://www.mysqltutorial.org/mysql-basics/mysql-subquery/)as shown in the following query:



MySQL MIN with Subquery

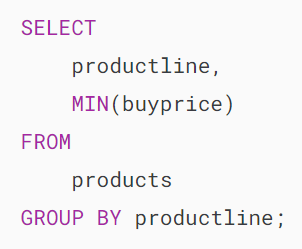
How it works.

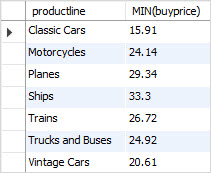
* The [subquery](https://www.mysqltutorial.org/mysql-basics/mysql-subquery/) returns the lowest buy-price product in the products table.
* The outer query selects the product whose buy price is equal to the lowest price returned from the subquery.

### **4) Using MySQL MIN() function with a GROUP BY example**

Like other aggregate functions, the MIN() function is often used with the [GROUP BY](https://www.mysqltutorial.org/mysql-basics/mysql-group-by/) clause to find the minimum value for every group.

This example uses the MIN() function with a GROUP BY clause to get the lowest buy price product for each product line:



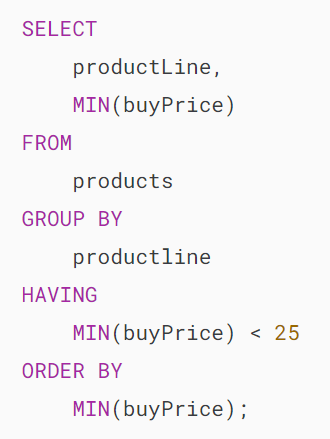


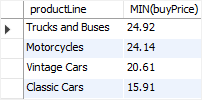
In this example:

* First, the GROUP BY clause groups products by product line.
* Second, the MIN() function returns the lowest buy-price product in each product line.

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

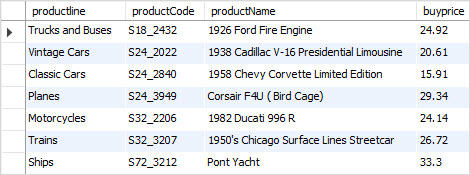
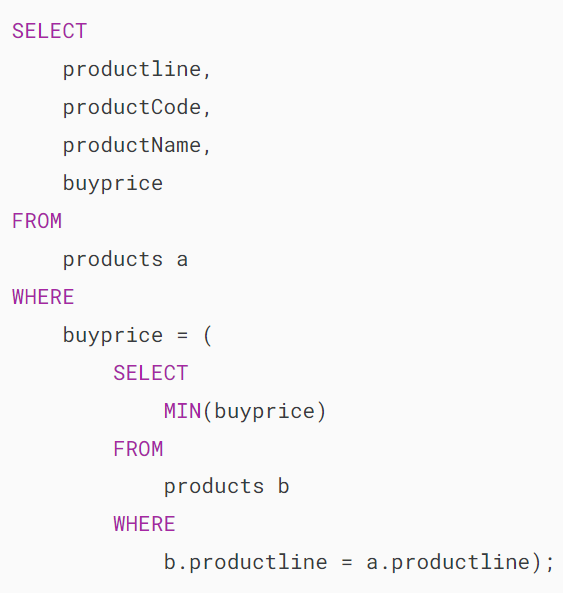
This query uses the MIN() function with the GROUP BY and HAVING clauses to find the product lines that have the lowest buy price of less than 21:





### **6) Using MIN() with a correlated subquery**

The following query selects the lowest-priced product in every product line by combining the MIN() function with a [correlated subquery](https://www.mysqltutorial.org/mysql-basics/mysql-subquery/):



In this example, for each product line from the outer query, the correlated subquery selects the lowest-priced product in the product line and returns the lowest price.

The returned lowest price is then used as input for the outer query to find the related product data including product line, product code, product name, and buy price.

If you want to achieve the same result without using the MIN() function and a [subquery](https://www.mysqltutorial.org/mysql-basics/mysql-subquery/), you can use a [self join](https://www.mysqltutorial.org/mysql-basics/mysql-self-join/" \o "MySQL Self Join) as follows:

## 

## **Summary**

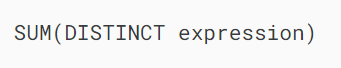
* Use the MIN() function to find the minimum value in a set of values.

# **MySQL SUM() function**

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

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

The SUM() function is an [aggregate function](https://www.mysqltutorial.org/mysql-aggregate-functions/) that allows you to calculate the sum of values in a set. The syntax of the SUM() function is as follows:

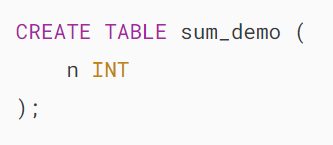


Here is how the SUM() function works:

* If you use the SUM() function in a [SELECT](https://www.mysqltutorial.org/mysql-basics/mysql-select-from/) statement that returns no row, the SUM() function returns NULL, not zero.
* The DISTINCT option directs the SUM() function to calculate the sum of unique values in a set.
* The SUM() function ignores the NULL values in the calculation.

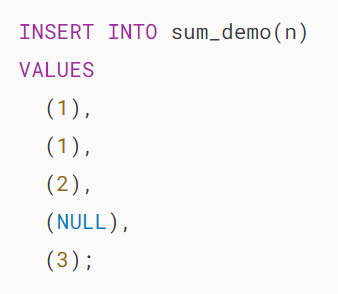
## **MySQL SUM() function illustration**

First, [create a new table](https://www.mysqltutorial.org/perl-mysql/perl-mysql-create-table/) called sum\_demo:

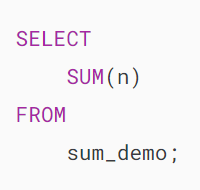


The sum\_demo table includes one column called n with the type INT.

Then, [insert some rows](https://www.mysqltutorial.org/mysql-basics/mysql-insert-multiple-rows/) into the sum\_demo table:



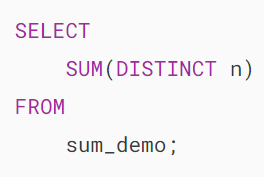
Third, calculate the total values in the n column using the SUM() function:





As you can see, the SUM() function calculates the total of 1, 1, 2, and 3. And it ignores NULL.

Finally, calculate the total unique values in the n column using the SUM() function with the DISTINCT option:

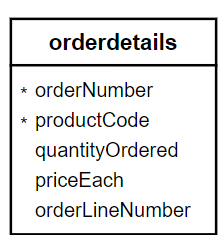




In this case, the SUM() with the DISTINCT option calculates the sum of unique values which are 1, 2, and 3.

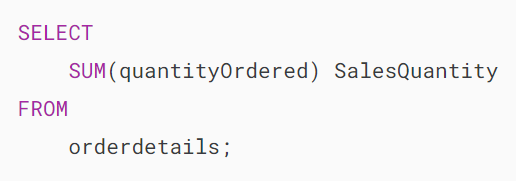
## **MySQL SUM() function examples**

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



### **1) Simple MySQL SUM() function example**

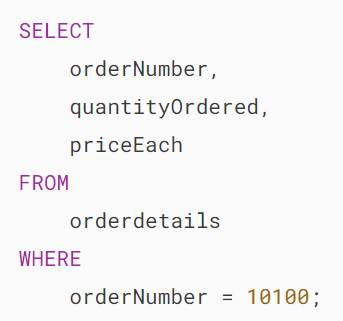
This example uses the SUM() function to get the total number of items of the order details:

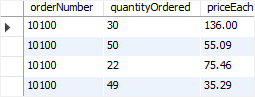




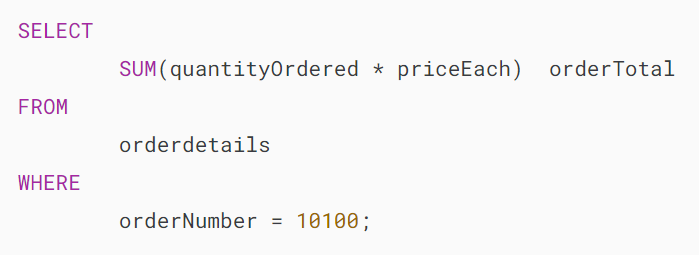
### **2) MySQL SUM() function with expression example**

The following shows the order line items of the order number 10110:





To calculate the total for the order number 10110, you use the SUM() function as follows:





In this example, the SUM() function calculates the total of the following expression of all order line items of order number 10110:

### 

### **3) MySQL SUM() with the GROUP BY clause example**

The SUM() function is often used with the [GROUP BY](https://www.mysqltutorial.org/mysql-basics/mysql-group-by/) clause to calculate the sum for each group.

For example, you can calculate the total amount of each order by using the SUM() function with the GROUP BY clause as shown in the following query:

### 

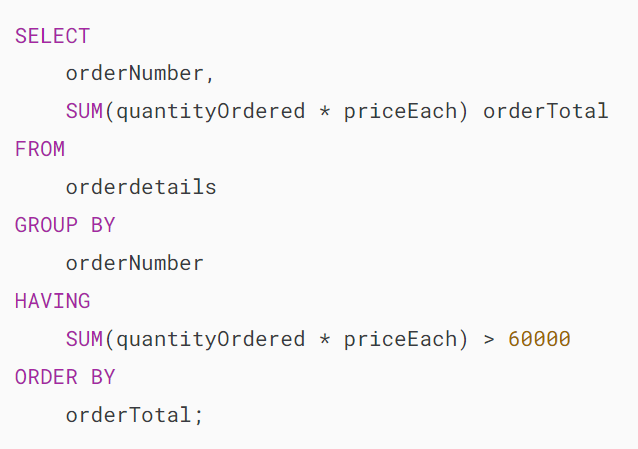
### 

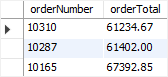
In this example:

* The GROUP BY clause divides order details into groups grouped by the order number.
* The SUM() function calculates the total of each amount in each order.

### **4) MySQL SUM() with HAVING clause example**

You can use the SUM() function in the [HAVING](https://www.mysqltutorial.org/mysql-basics/mysql-having/) clause to filter the group. This example illustrates how to select orders whose order amounts are greater than 60,000.



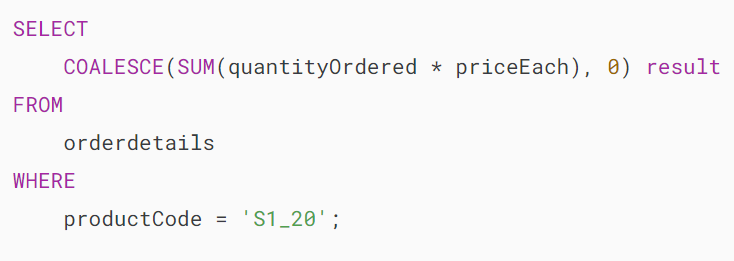


### **5) MySQL SUM() with NULL example**

The SUM() function returns NULL if the result set is empty. Sometimes, you may want the SUM() function to return zero instead of NULL.

In this case, you can use the [COALESCE()](https://www.mysqltutorial.org/mysql-comparison-functions/mysql-coalesce/) function. The COALESCE function accepts two arguments and returns the second argument if the first argument is NULL; otherwise, it returns the first argument.

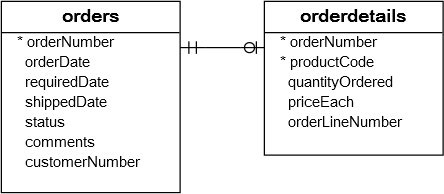
See the following query:





### **6) MySQL SUM() with join example**

See the following orders and orderdetails tables:



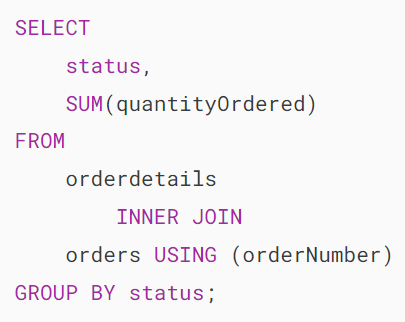
You can use the SUM() function in a SELECT with [JOIN](https://www.mysqltutorial.org/mysql-basics/mysql-join/) clause to calculate the sum of values in a table based on a condition specified by the values in another table.

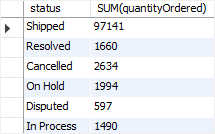
This statement uses the SUM() function to calculate the total amounts of the canceled orders:

### 

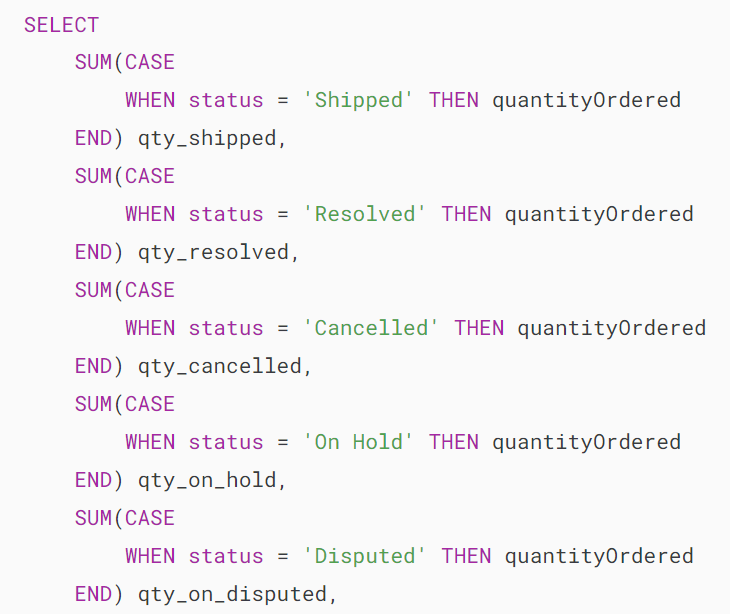
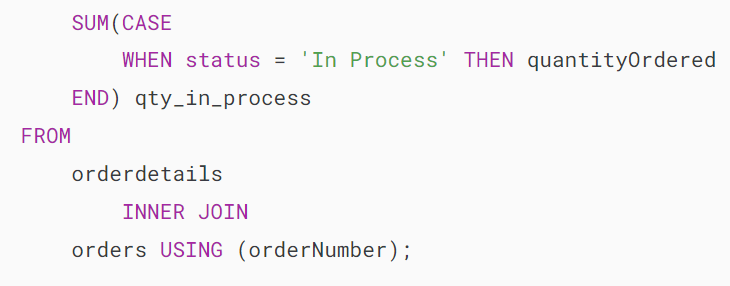
### **7) MySQL SUM IF example**

The following statement uses the SUM() function to calculate the number of items sold for each order status:





If you want to rotate rows to columns, you can use the SUM() function with [CASE](https://www.mysqltutorial.org/mysql-control-flow-functions/mysql-case-function/) expression. It is kind of SUMIF logic:

## **Summary**

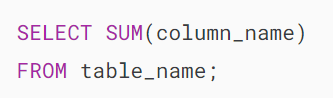
* Use the MySQL SUM() function to calculate the sum of values in a set.

# **MySQL SUM IF**

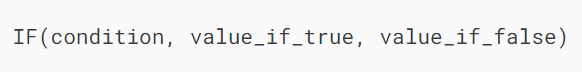
**Summary**: in this tutorial, you will learn about the MySQL SUM IF function to perform conditional summation.

## **Introduction to MySQL SUM IF functions**

The [SUM()](https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-sum/) function is an [aggregate function](https://www.mysqltutorial.org/mysql-aggregate-functions/) that returns the total of values in a column of a table.:

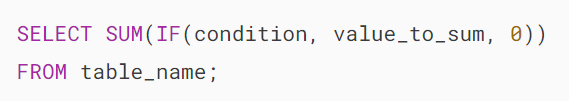


The [IF](https://www.mysqltutorial.org/mysql-control-flow-functions/mysql-if-function/) function is a [flow control function](https://www.mysqltutorial.org/mysql-control-flow-functions/) that returns a value if a condition is true and another value if the expression is false:



When you combine the SUM function with the IF function, you can perform conditional summation, making it a powerful tool for data analysis.

Here’s the basic syntax of the SUM IF:



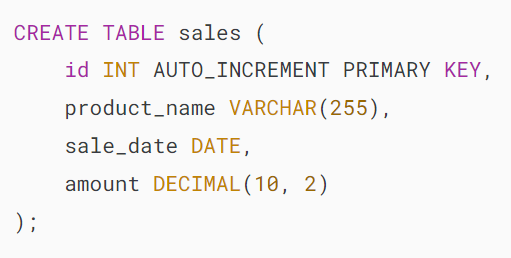
In this syntax:

* condition: The condition that you want to apply.
* value\_to\_sum: The value that you want to sum if the condition is true.
* 0: The value to sum if the condition is false. You can change it to any default value you want.

## **MySQL SUM IF example**

Let’s take an example of using the SUM IF.

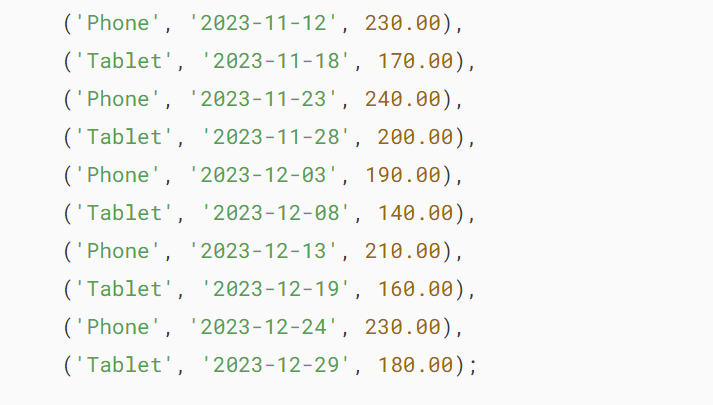
First, [create a new table](https://www.mysqltutorial.org/mysql-basics/mysql-create-table/) named sales with the following structure:



The sales table has four columns:

* id for a unique identifier.
* product\_name to store the product name.
* sale\_date to store the date of the sale.
* amount to store the sale amount.

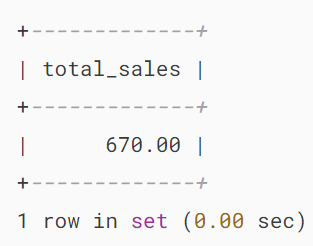
Second, [insert some rows](https://www.mysqltutorial.org/mysql-basics/mysql-insert/) into the sales table:

Third, calculate the total sales amount for “Phone” in October 2023



Output:



Here’s how the query works.

First, the SUM function calculates the sum of values.

Second, the IF function checks three conditions:

* product\_name = 'Phone': This ensures only rows with Phone are considered.
* MONTH(sale\_date) = 10: It checks if the sale date is in October (10).
* YEAR(sale\_date) = 2023: It checks if the year is 2023.

The query returns the total sales amount for Phone in October 2023.

## **Summary**

* Use the MySQL SUM IF to perform a conditional summation.

# **MySQL Standard Deviation Functions**

https://www.mysqltutorial.org/mysql-aggregate-functions/mysql-standard-deviation/