

MySQL BOOLEAN Data Type

Discover more

 Databases

 API development platforms

 Server monitoring tools

 Advanced SQL techniques

 Data analysis tools

 Sample MySQL databases

 Data migration services

 database

 Backup and recovery software

Summary: in this tutorial, you will learn about MySQL `BOOLEAN` data type and how to use it to store Boolean values in the databases.

Introduction to MySQL BOOLEAN data type

MySQL does not have a dedicated Boolean data type. Instead, MySQL uses `TINYINT(1)` to represent the `BOOLEAN` data type.

To make it more convenient when defining `BOOLEAN` column, MySQL offers `BOOLEAN` or `BOOL` as the synonym for `TINYINT(1)`.

So instead of defining a `BOOLEAN` column like this:

```
column_name TINYINT(1)
```

You can use the `BOOL` or `BOOLEAN` keyword as follows:

```
column_name BOOL
```

In MySQL, the convention is that zero is considered false, while a non-zero value is considered

true.

When working with Boolean literals, you can use the constants `true` and `false` case-insensitively, which is equivalent to 1 and 0 respectively. For example:

```
SELECT true, false, TRUE, FALSE, True, False;
```

Output:

```
1 0 1 0 1 0
```

MySQL BOOLEAN example

We'll take an example of using the MySQL BOOLEAN data type.

First, [create a new table](#) called `tasks` :

```
CREATE TABLE tasks (
    id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    completed BOOLEAN
);
```

The `tasks` table has three columns `id`, `title`, and `completed`.

The `completed` is a `BOOLEAN` column. Since the `BOOLEAN` is a synonym for `TINYINT(1)`, when you [describe the table structure](#), MySQL shows the `TINYINT(1)` instead:

```
DESCRIBE tasks;
```

Output:

Field	Type	Null	Key	Default	Extra
<code>id</code>	<code>int</code>	<code>NO</code>	<code>PRI</code>	<code>NULL</code>	<code>auto_increment</code>
<code>title</code>	<code>varchar(255)</code>	<code>NO</code>		<code>NULL</code>	

```
| completed | tinyint(1) | YES |      | NULL   |  
+-----+-----+-----+-----+-----+  
3 rows in set (0.00 sec)
```

Second, insert two rows into the `tasks` table:

```
INSERT INTO tasks(title, completed)  
VALUES  
( 'Master MySQL Boolean type', true),  
( 'Design database table', false);
```

Before saving data into the Boolean column, MySQL converts it into 1 or 0.

Third, retrieve data from `tasks` table:

```
SELECT  
    id,  
    title,  
    completed  
FROM  
    tasks;
```

Output:

```
+-----+-----+  
| id | title           | completed |  
+-----+-----+  
| 1  | Master MySQL Boolean type | 1 |  
| 2  | Design database table     | 0 |  
+-----+-----+  
2 rows in set (0.00 sec)
```

The output indicates that MySQL converted the `true` and `false` to 1 and 0 respectively.

Fourth, because `BOOLEAN` is `TINYINT(1)`, you can insert values other than 1 and 0 into the `BOOLEAN` column. For example:

```
INSERT INTO tasks(title, completed)
```

VALUES

```
('Test Boolean with a number', 2);
```

Output:

```
Query OK, 1 row affected (0.01 sec)
```

Fifth, query data from the `tasks` table:

```
SELECT * FROM tasks;
```

Output:

```
+-----+-----+
| id | title           | completed |
+-----+-----+
| 1  | Master MySQL Boolean type | 1 |
| 2  | Design database table   | 0 |
| 3  | Test Boolean with a number | 2 |
+-----+-----+
3 rows in set (0.00 sec)
```

If you want to output the result as `true` and `false`, you can use the `IF` function as follows:

```
SELECT
    id,
    title,
    IF(completed, 'true', 'false') completed
FROM
    tasks;
```

Output:

```
+-----+-----+
| id | title           | completed |
+-----+-----+
| 1  | Master MySQL Boolean type | true |
+-----+-----+
```

```
| 2 | Design database table | false |
| 3 | Test Boolean with a number | true |
+---+-----+-----+
3 rows in set (0.00 sec)
```

Sixth, insert NULL into the completed column:

```
INSERT INTO tasks(title, completed)
VALUES
('Test Boolean with NULL', NULL);
```

Finally, retrieve data from the tasks table:

```
SELECT * FROM tasks;
```

Output:

```
+-----+-----+
| id | title | completed |
+-----+-----+
| 1 | Master MySQL Boolean type | 1 |
| 2 | Design database table | 0 |
| 3 | Test Boolean with a number | 2 |
| 4 | Test Boolean with NULL | NULL |
+-----+-----+
4 rows in set (0.00 sec)
```

MySQL BOOLEAN operators

To retrieve all completed tasks from the `tasks` table, you might come up with the following query:

```
SELECT
    id, title, completed
FROM
    tasks
WHERE
```

```
completed = TRUE;
```

Output:

```
+----+-----+-----+
| id | title           | completed |
+----+-----+-----+
| 1  | Master MySQL Boolean type |      1 |
+----+-----+-----+
1 row in set (0.00 sec)
```

The query returned the task with `completed` value 1. It does not show the task with the completed value 2 because `TRUE` is 1, not 2.

To fix it, you can use the `IS` operator:

```
SELECT
    id,
    title,
    completed
FROM
    tasks
WHERE
    completed IS TRUE;
```

Output:

```
+----+-----+-----+
| id | title           | completed |
+----+-----+-----+
| 1  | Master MySQL Boolean type |      1 |
| 3  | Test Boolean with a number |      2 |
+----+-----+-----+
2 rows in set (0.00 sec)
```

In this example, we used the `IS` operator to test a value against the `TRUE` value.

To get all the pending tasks, you can use `IS FALSE` or `IS NOT TRUE` as follows:

```
SELECT
    id,
    title,
    completed
FROM
    tasks
WHERE
    completed IS NOT TRUE;
```

Output:

```
+----+-----+-----+
| id | title           | completed |
+----+-----+-----+
|  2 | Design database table |      0 |
|  4 | Test Boolean with NULL |    NULL |
+----+-----+-----+
2 rows in set (0.00 sec)
```

Summary

- MySQL has no dedicated `BOOLEAN` data type. Instead, it uses `TINYINT(1)` to represent the `BOOLEAN` type.
- Use the `BOOLEAN` keyword to declare a column with the `BOOLEAN` type. The `BOOLEAN` and `TINYINT(1)` are synonyms.
- By convention, zero is false while non-zero is true.

Was this tutorial helpful?



ADVERTISEMENTS





?

PREVIOUSLY

[MySQL INT Data Type](#)

UP NEXT

[MySQL DECIMAL Data Type](#)

ADVERTISEMENTS

Search ...

GETTING STARTED

[What Is MySQL?](#)

[Install MySQL Database Server](#)

[Connect to MySQL Server](#)

[Download MySQL Sample Database](#)

[Load Sample Database](#)

QUERYING DATA

[SELECT FROM](#)

SELECT
ORDER BY
WHERE
SELECT DISTINCT
AND
OR
IN
NOT IN
BETWEEN
LIKE
LIMIT
IS NULL

Table & Column Aliases

Joins

INNER JOIN
LEFT JOIN
RIGHT JOIN
Self Join

CROSS JOIN

GROUP BY

HAVING

HAVING COUNT

ROLLUP

Subquery

Derived Tables

[EXISTS](#)

[EXCEPT](#)

[INTERSECT](#)

ADVERTISEMENTS

MANAGING DATABASES

[Select a Database](#)

[Create Databases](#)

[Drop Databases](#)

MANAGING TABLES

[Create Tables](#)

[AUTO_INCREMENT](#)

[Rename Tables](#)

[Add Columns](#)

[Drop Columns](#)

[Drop Tables](#)

[Temporary Tables](#)

[Generated Columns](#)

MYSQL CONSTRAINTS

[Primary Key](#)

[Foreign Key](#)

[Disable Foreign Key Checks](#)

[UNIQUE Constraint](#)

[NOT NULL Constraint](#)

[DEFAULT Constraint](#)

CHECK Constraint

ADVERTISEMENTS



?

INSERT DATA

[Insert Into](#)

[Insert Multiple Rows](#)

[INSERT INTO SELECT](#)

[Insert On Duplicate Key Update](#)

[INSERT IGNORE](#)

[Insert DateTimes](#)

[Insert Dates](#)

UPDATE DATA

[UPDATE](#)

[UPDATE JOIN](#)

DELETE DATA

[DELETE JOIN](#)

[ON DELETE CASCADE](#)

[TRUNCATE TABLE](#)

MYSQL TRANSACTIONS

[Table Locking](#)

MYSQL DATA TYPES

[BIT](#)

[INT](#)

[BOOLEAN](#)

[DECIMAL](#)

[DATETIME](#)

[TIMESTAMP](#)

[DATE](#)

[TIME](#)

[CHAR](#)

[VARCHAR](#)

[TEXT](#)

[BINARY](#)

[VARBINARY](#)

[ENUM](#)

[BLOB](#)

MYSQL GLOBALIZATION

[MySQL Character Sets](#)

[MySQL Collation](#)

MYSQL IMPORT & EXPORT

[Import a CSV File Into a Table](#)

[Export a Table to a CSV File](#)

ABOUT MYSQL TUTORIAL**WEBSITE**

MySQLTutorial.org helps you master MySQL quickly, easily, and with enjoyment. Our tutorials make learning MySQL a breeze.

All MySQL tutorials are clear, practical and easy-to-follow.

[More About Us](#)

LATEST TUTORIALS

[MySQL Port](#)

[MySQL Commands](#)

[innodb_dedicated_server:](#)

[Configure InnoDB Dedicated Server](#)

[innodb_flush_method:](#)

[Configure InnoDB Flush Method](#)

[innodb_log_buffer_size:](#)

[Configure InnoDB Log Buffer Size](#)

[innodb_buffer_pool_chunk_size:](#)

[Configure Buffer Pool Chunk Size](#)

[innodb_buffer_pool_instances:](#)

[Configuring Multiple Buffer Pool Instances for Improved Concurrency in MySQL](#)

[innodb_buffer_pool_size:](#)

[Configure InnoDB Buffer Pool Size](#)

[MySQL InnoDB Architecture](#)

[How to Kill a Process in MySQL](#)

SITE LINKS

[Donation](#) 

[Contact Us](#)

[About](#)

[Privacy Policy](#)

OTHERS

[MySQL Cheat Sheet](#)

[MySQL Resources](#)

[MySQL Books](#)