MySQL CHAR Data Type

Summary: in this tutorial, you will learn about MySQL `CHAR` data type and how to apply it in your database table design.

Introduction to MySQL CHAR data type

The `CHAR` data type is a fixed-length character type in MySQL. You often declare the `CHAR` type with a length that specifies the maximum number of characters that you want to store. For example, `CHAR(20)` can hold up to 20 characters.

If the data that you want to store is a fixed size, then you should use the `CHAR` data type. You'll get a better performance in comparison with `VARCHAR` this case.

The length of the `CHAR` data type can be any value from 0 to 255. When you store a `CHAR` value, MySQL pads its value with spaces to the length that you declared.

When you query the `CHAR` value, MySQL removes the trailing spaces.

> Note that MySQL will not remove the trailing spaces if you enable the PAD_CHAR_TO_FULL_LENGTH SQL mode.

Consider the following example.

First, create a table with a `CHAR` column.

```
CREATE TABLE mysql_char_test (
    status CHAR(3)
);
```

The data type of the `status` column is `CHAR` . It can hold up to 3 characters.

Second, insert two rows into the `mysql_char_test` table.

```
INSERT INTO mysql_char_test(status)
VALUES('Yes'),('No');
```

Third, use the length function to get the length of each `CHAR` value.

```
SELECT
    status,
    LENGTH(status)
FROM
    mysql_char_test;
```

	status	LENGTH(status)
▶	Yes	3
	No	2

Fourth, insert a `CHAR` value with the leading and trailing spaces.

```
INSERT INTO mysql_char_test(status)
VALUES(' Y ');
```

Finally, query the inserted values, and you will see that MySQL removes the trailing spaces.

```
SELECT
    status,
    LENGTH(status)
FROM
```

```
    mysql_char_test;
```

Comparing MySQL CHAR values

When storing or comparing the `CHAR` values, MySQL uses the character set collation assigned to the column.

MySQL does not consider trailing spaces when comparing `CHAR` values using the comparison operator such as =, <>, >, <, etc.

Notice that the `LIKE` operator does consider the trailing spaces when you do pattern matching with `CHAR` values.

In the previous example, we stored the value `Y` with both leading and trailing spaces. However, when we execute the following query:

```
SELECT *
FROM mysql_char_test
WHERE status = 'Y';
```

MySQL returns no row because it does not consider the trailing space. To match with the ' Y ', we need to remove the trailing space as follows:

```
SELECT *
FROM mysql_char_test
WHERE status = ' Y';
```

MySQL CHAR and UNIQUE index

If the `CHAR` column has a `UNIQUE` index and you insert a value that is different from an existing value in a number of trailing spaces, MySQL will reject the changes because of duplicate-key

error.

See the following example.

First, create a unique index for the `status` column of the `mysql_char_test` table.

```
CREATE UNIQUE INDEX uidx_status
ON mysql_char_test(status);
```

Second, insert a new row into the `mysql_char_test` table.

```
INSERT INTO mysql_char_test(status)
VALUES('N');
```

Third, insert the following value will cause a duplicate key error.

```
INSERT INTO mysql_char_test(status)
VALUES('N ');
```

```
Error Code: 1062. Duplicate entry 'N' for key 'uidx_status'
```

Summary

- MySQL `CHAR` data type is a fixed-length character type.
- Use the `CHAR` data type to store fixed-length character data.

PREVIOUSLY
MySQL TIME Data Type

UP NEXT
MySQL VARCHAR Data Type

Search ...

GETTING STARTED

QUERYING DATA

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

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

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,

LATEST TUTORIALS

MySQL Port

MySQL Commands

SITE LINKS

Donation ❤️

Contact Us

and with enjoyment. Our tutorials make learning MySQL a breeze.

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

More About Us

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

About

Privacy Policy

OTHERS

MySQL Cheat Sheet

MySQL Resources

MySQL Books