MySQL VARCHAR Data Type

Summary: this tutorial introduces you to the MySQL `VARCHAR` data type and discusses some important features of `VARCHAR`.

Introduction to MySQL VARCHAR data type

MySQL `VARCHAR` is the variable-length string whose length can be up to 65,535. MySQL stores a `VARCHAR` value as a 1-byte or 2-byte length prefix plus actual data.

The length prefix specifies the number of bytes in the value. If a column requires less than 255 bytes, the length prefix is 1 byte. In case the column requires more than 255 bytes, the length prefix is two length bytes.

The maximum length, however, is subject to the maximum row size (65,535 bytes) and the character set used. It means that the total length of all columns should be less than 65,535 bytes.

Let's take a look at an example.

We will create a new table that has two columns `s1` and `s2` with the length of 32765(+2 for length prefix) and 32766 (+2). Note that 32765+2+32766+2=65535, which is the maximum row size.

```
CREATE TABLE IF NOT EXISTS varchar_test (
```

```
    s1 VARCHAR(32765) NOT NULL,
    s2 VARCHAR(32766) NOT NULL
)  CHARACTER SET 'latin1' COLLATE LATIN1_DANISH_CI;
```

The statement created the table successfully. However, if we increase the length of the `s1` column by 1.

```
CREATE TABLE IF NOT EXISTS varchar_test_2 (
    s1 VARCHAR(32766) NOT NULL, -- error
    s2 VARCHAR(32766) NOT NULL
)  CHARACTER SET 'latin1' COLLATE LATIN1_DANISH_CI;
```

MySQL will issue the error message:

```
Error Code: 1118. Row size too large. The maximum row size for the used table type, not cc
```

The output indicates that the row size is too large and the statement fails.

If you insert a string whose length is greater than the length of a `VARCHAR` column, MySQL will issue an error and skip inserting data. For example:

```
CREATE TABLE items (
    id INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(3)
);

INSERT INTO items(title)
VALUES('ABCD');
```

In this example, MySQL will issue the following error message:

```
Error Code: 1406. Data too long for column 'title' at row 1 0.000 sec
```

MySQL VARCHAR and spaces

MySQL does not implicitly pad space when storing the `VARCHAR` values. Additionally, MySQL

retains the trailing spaces when inserting or retrieving `VARCHAR` values. For example:

```
INSERT INTO items(title)
VALUES('AB ');
```

```
SELECT
    id, title, length(title)
FROM
    items;
```

Output:

```
+----+-------+---------------+
| id | title | length(title) |
+----+-------+---------------+
|  1 | AB    |             3 |
+----+-------+---------------+
1 row in set (0.00 sec)
```

However, MySQL will truncate the trailing spaces when inserting a `VARCHAR` value that contains trailing spaces which cause the column length exceeded. In addition, MySQL issues a warning. Let's see the following example:

```
INSERT INTO items(title)
VALUES('ABC ');
```

Output:

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

This statement inserts a string whose length is 4 into the `title` column. MySQL inserts the string but truncates the trailing space before inserting the value.

To show the warning message, you can use the `SHOW WARNINGS` statement:

```
SHOW WARNINGS
```

Output:

```
1 row(s) affected, 1 warning(s): 1265 Data truncated for column 'title' at row 1
```

The following query verifies the action:

```
SELECT
    title, LENGTH(title)
FROM
    items;
```

Output:

```
+-------+---------------+
| title | LENGTH(title) |
+-------+---------------+
| AB    |             3 |
| ABC   |             3 |
+-------+---------------+
2 rows in set (0.00 sec)
```

In this tutorial, you have learned how to use MySQL `VARCHAR` data type to store variable strings in the database.

Data analysis tools

MySQL hosting service

Cloud database solutions

PREVIOUSLY
MySQL CHAR Data Type

UP NEXT
MySQL TEXT Data Type

Search ...

GETTING STARTED

QUERYING DATA

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