MySQLTUTORIAL

How to Add Columns to a Table Using MySQL ADD COLUMN Statement

Summary: in this tutorial, we will show you how to add a column to a table using MySQL ADD COLUMN statement.

Introduction to MySQL ADD COLUMN statement

To add a new column to an existing table, you use the ALTER TABLE (https://www.mysqltutorial.org/mysql-alter-table.aspx) `ADD COLUMN` statement as follows:

```
ALTER TABLE table
ADD [COLUMN] column_name column_definition [FIRST|AFTER existing_column];
```

Let's examine the statement in more detail.

- First, you specify the table name after the `ALTER TABLE` clause.
- Second, you put the new column and its definition after the `ADD COLUMN` clause. Note that `COLUMN` keyword is optional so you can omit it.
- Third, MySQL allows you to add the new column as the first column of the table by specifying the `FIRST` keyword. It also allows you to add the new column after an existing column using the `AFTER existing_column` clause. If you don't explicitly specify the position of the new column, MySQL will add it as the last column.

To add two or more columns to a table at the same time, you use the following syntax:

```
ALTER TABLE table
ADD [COLUMN] column_name_1 column_1_definition [FIRST|AFTER existing_column],
ADD [COLUMN] column_name_2 column_2_definition [FIRST|AFTER existing_column],
...;
```

Let's take a look some examples of adding a new column to an existing table.

MySQL ADD COLUMN examples

First, we create a table (https://www.mysqltutorial.org/mysql-create-table/) named `vendors` for the demonstration purpose using the following statement:

```
CREATE TABLE IF NOT EXISTS vendors (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255)
);
```

Second, we add a new column named `phone` to the `vendors` table. Because we specify the position of the `phone` column explicitly after the `name` column, MySQL will obey this.

```
ALTER TABLE vendors
ADD COLUMN phone VARCHAR(15) AFTER name;
```

Third, we add a new column named `vendor_group` to the `vendors` table. At this time, we don't specify the new column's position so MySQL adds the `vendor_group` column as the last column of the `vendors` table.

```
ALTER TABLE vendors
ADD COLUMN vendor_group INT NOT NULL;
```

Let's insert some rows (https://www.mysqltutorial.org/mysql-insert-statement.aspx) into the `vendors` table.

```
INSERT INTO vendors(name,phone,vendor_group)
VALUES('IBM','(408)-298-2987',1);

INSERT INTO vendors(name,phone,vendor_group)
VALUES('Microsoft','(408)-298-2988',1);
```

We can query the data (https://www.mysqltutorial.org/mysql-select-statement-query-data.aspx) of the `vendors` table to see the changes.

```
SELECT
    id, name, phone,vendor_group
FROM
    vendors;
```

Fourth, add two more columns `email` and `hourly_rate` to the `vendors` table at the same time.

```
ALTER TABLE vendors
ADD COLUMN email VARCHAR(100) NOT NULL,
ADD COLUMN hourly_rate decimal(10,2) NOT NULL;
```

Note that both `email` and `hourly_rate` columns are assigned to `NOT NULL values` However, the `vendors` table already has data. In such cases, MySQL will use default values for those new columns.

Let's check the data in the `vendors` table.

```
SELECT
    id, name, phone, vendor_group, email, hourly_rate
FROM
    vendors;
```

The email column is populated with blank values, not the `NULL` values. And the hourly_rate column is populated with `0.00` values.

If you accidentally add a column that already exists in the table, MySQL will issue an error. For example, if you execute the following statement:

```
ALTER TABLE vendors
```

```
ADD COLUMN vendor_group INT NOT NULL;
```

MySQL issued an error message:

```
Error Code: 1060. Duplicate column name 'vendor_group'
```

For the table with a few columns, it is easy to see which columns are already there. However, with a big table with hundred of columns, it is more difficult.

In some situations, you want to check whether a column already exists in a table before adding it. However, there is no statement like `ADD COLUMN IF NOT EXISTS` available. Fortunately, you can get this information from the `columns` table of the `information_schema` database as the following query:

```
SELECT
    IF(count(*) = 1, 'Exist','Not Exist') AS result
FROM
    information_schema.columns
WHERE
    table_schema = 'classicmodels'
        AND table_name = 'vendors'
        AND column_name = 'phone';
```

In the WHERE clause (https://www.mysqltutorial.org/mysql-where/) , we passed three arguments: table schema or database, table name, and column name. We used IF function (https://www.mysqltutorial.org/mysql-if-function.aspx) to return whether the column exists or not.

In this tutorial, you have learned how to add one or more columns to a table using MySQL ADD COLUMN statement.