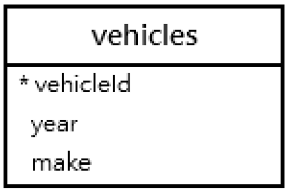
MySQL ALTER TABLE

Summary: in this tutorial, you will learn how to use the MySQL `ALTER TABLE` statement to add a column, alter a column, rename a column, drop a column and rename a table.

Setting up a sample table

Let's create a table (https://www.mysqltutorial.org/mysql-create-table/) named `vehicles` for the demonstration:

```
CREATE TABLE vehicles (
    vehicleId INT,
    year INT NOT NULL,
    make VARCHAR(100) NOT NULL,
    PRIMARY KEY(vehicleId)
);
```



MySQL ALTER TABLE – Add columns to a table

The `ALTER TABLE ADD` statement allows you to add one or more columns to a table.

1) Add a column to a table

To add a column (https://www.mysqltutorial.org/mysql-add-column/) to a table, you use the `ALTER TABLE ADD` syntax:

```
ALTER TABLE table_name
ADD
```

```
    new_column_name column_definition
    [FIRST | AFTER column_name]
```

In this syntax:

- `table_name` – specify the name of the table that you want to add a new column or columns after the `ALTER TABLE` keywords.
- `new_column_name` – specify the name of the new column.
- `column_definition` – specify the datatype, maximum size, and column constraint of the new column
- `FIRST | AFTER column_name` specify the position of the new column in the table. You can add a column after an existing column (`ATER column_name`) or as the first column (`FIRST`). If you omit this clause, the column is appended at the end of the column list of the table.

The following example uses the `ALTER TABLE ADD` statement to add a column at the end of the `vehicles` table:

```
ALTER TABLE vehicles
ADD model VARCHAR(100) NOT NULL;
```

This statement shows the column list of the `vehicles` table:

```
DESCRIBE vehicles;
```

As shown clearly from the output, the column `model` has been added to the `vehicles` table.

2) Add multiple columns to a table

To add multiple columns (https://www.mysqltutorial.org/mysql-add-column/) to a table, you use the following form of the `ALTER TALE ADD` statement:

```
ALTER TABLE table_name
    ADD new_column_name column_definition
    [FIRST | AFTER column_name],
    ADD new_column_name column_definition
    [FIRST | AFTER column_name],
    ...;
```

For example, this statement adds two columns `color` and `note` to the `vehicles` table:

```
ALTER TABLE vehicles
ADD color VARCHAR(50),
ADD note VARCHAR(255);
```

This statement shows the new structure of the `vehicles` table:

```
DESCRIBE vehicles;
```

MySQL ALTER TABLE – Modify columns

1) Modify a column

Here is the basic syntax for modifying a column in a table:

```
ALTER TABLE table_name
MODIFY column_name column_definition
[ FIRST | AFTER column_name];
```

It’s a good practice to view the attributes of a column before modifying it.

Suppose that you want to change the `note` column a `NOT NULL (https://www.mysqltutorial.org/mysql-not-null-constraint/)` column with a maximum of 100 characters.

First, show the column list of the `vehicles` table:

```
DESCRIBE vehicles;
```

Then, modify the `note` column:

```
ALTER TABLE vehicles
MODIFY note VARCHAR(100) NOT NULL;
```

Finally, show the column list of the `vehicles` table to verify the change:

```
DESCRIBE vehicles;
```

2) Modify multiple columns

The following statement allows you to modify multiple columns:

```
ALTER TABLE table_name
    MODIFY column_name column_definition
    [ FIRST | AFTER column_name],
    MODIFY column_name column_definition
    [ FIRST | AFTER column_name],
    ...;
```

First, show the current columns of the `vehicles` table:

Second, use the `ALTER TABLE MODIFY` statement to modify multiple columns:

```
ALTER TABLE vehicles
MODIFY year SMALLINT NOT NULL,
MODIFY color VARCHAR(20) NULL AFTER make;
```

In this example:

- First, modify the data type of the `year` column from `INT` to `SMALLINT`
- Second, modify the `color` column by setting the `maximum` length to `20`, removing the `NOT NULL` constraint, and changing its position to appear after the `make` column.

Third, show the new column list of the `vehicles` table to verify the modifications:

MySQL ALTER TABLE – Rename a column in a table

To rename a column, you use the following statement:

```
ALTER TABLE table_name
    CHANGE COLUMN original_name new_name column_definition
    [FIRST | AFTER column_name];
```

In this syntax:

- First, specify the name of the table to which the column belongs.
- Second, specify the column name and the new name followed by column definition after the `CHANGE COLUMN` keywords.
- Third, use the `FIRST` or `AFTER column_name` option to determine the new position of the column.

The following example uses the `ALTER TABLE CHANGE COLUMN` statement to rename the column `note` to `vehicleCondition` :

```
ALTER TABLE vehicles
CHANGE COLUMN note vehicleCondition VARCHAR(100) NOT NULL;
```

Let’s review the column list of the `vehicles` table:

```
DESCRIBE vehicles;
```

MySQL ALTER TABLE – Drop a column

To drop a column (https://www.mysqltutorial.org/mysql-drop-column/) in a table, you use the `ALTER TABLE DROP COLUMN` statement:

```
ALTER TABLE table_name
DROP COLUMN column_name;
```

In this syntax:

- First, specify the name of the table that you want to drop a column after the `ALTER TABLE` keywords.
- Second, specify the name of the column that you want to drop after the `DROP COLUMN` keywords.

This example shows how to remove the `vehicleCondition` column from the `vehicles` table:

```
ALTER TABLE vehicles
DROP COLUMN vehicleCondition;
```

MySQL ALTER TABLE – Rename table

To rename a table (https://www.mysqltutorial.org/mysql-rename-table/) , you use the `ALTER TABLE RENAME TO` statement:

```
ALTER TABLE table_name
RENAME TO new_table_name;
```

In this syntax:

- First, specify the name of the table that you want to rename after the `ALTER TABLE` keywords.
- Second, specify the new name for the table after the `RENAME TO` keywords.

This example renames the `vehicles` table to `cars` :

```
ALTER TABLE vehicles
RENAME TO cars;
```

In this tutorial, you have learned how to use the MySQL `ALTER TABLE` statement to add a column, modify a column, rename a column, drop a column and rename a table.