MySQLTUTORIAL

MySQL NOT NULL Constraint

Summary: in this tutorial, you will learn how to define a `NOT NULL` constraint for a column, add a `NOT NULL` constraint to an existing column, and remove a `NOT NULL` constraint from a column.

Introduction to the MySQL NOT NULL constraint

The `NOT NULL` constraint is a column constraint that ensures values stored in a column are not `NULL` (https://www.mysqltutorial.org/mysql-null/) .

The syntax of defining a `NOT NULL` constraint is as follows:

```
column_name data_type NOT NULL;
```

A column may contain only one `NOT NULL` constraint which specifies a rule that the column must not contain any `NULL` value. In other words, if you update (https://www.mysqltutorial.org/mysql-update-data.aspx) or insert (https://www.mysqltutorial.org/mysql-insert-statement.aspx) `NULL` into a `NOT NULL` column, MySQL will issue an error.

The following `CREATE TABLE` (https://www.mysqltutorial.org/mysql-create-table/) statement creates the `tasks` table:

```
CREATE TABLE tasks (
    id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    start_date DATE NOT NULL,
    end_date DATE
);
```

In the `tasks` table, we explicitly define the `title` and `start_date` columns with `NOT NULL` constraints. The `id` column has the `PRIMARY KEY` (https://www.mysqltutorial.org/mysql-primary-key/) constraint, therefore, it implicitly includes a `NOT NULL` constraint.

The `end_date` column can have `NULL` values, assuming that when you create a new task, you may not know when the task can be completed.

It's a good practice to have the `NOT NULL` constraint in every column of a table unless you have a good reason not to do so.

Generally, the `NULL` value makes your queries more complicated because you have to use functions such as `ISNULL()` (https://www.mysqltutorial.org/mysql-isnull-function/), `IFNULL()` (https://www.mysqltutorial.org/mysql-ifnull/), and `NULLIF()` (https://www.mysqltutorial.org/mysql-nullif/) for handling `NULL`.

Add a NOT NULL constraint to an existing column

Typically, you add `NOT NULL` constraints to columns when you create the table. Sometimes, you want to add a `NOT NULL` constraint to a NULL-able column of an existing table. In this case, you use the following steps:

1. Check the current values of the column if there is any `NULL`.
2. Update the `NULL` to non-`NULL` if `NULLs` exist.
3. Modify the column with a `NOT NULL` constraint.

Consider the following example.

The following statement inserts some rows (https://www.mysqltutorial.org/mysql-insert-multiple-rows/) into the `tasks` table for the demonstration.

```
INSERT INTO tasks(title ,start_date, end_date)
VALUES('Learn MySQL NOT NULL constraint', '2017-02-01','2017-02-02'),
      ('Check and update NOT NULL constraint to your database', '2017-02-01',NULL);
```

Suppose that you want to force users to give an estimated end date when creating a new task. To implement this rule, you add a `NOT NULL` constraint to the `end_date` column of the `tasks` table.

First, use the `IS NULL` (https://www.mysqltutorial.org/mysql-is-null/) operator to find rows with `NULLs` in the column `end_date`:

```
SELECT *
FROM tasks
WHERE end_date IS NULL;
```

The query (https://www.mysqltutorial.org/mysql-select-statement-query-data.aspx) returned one row with `NULL` in the column `end_date`.

Second, update (https://www.mysqltutorial.org/mysql-update-data.aspx) the `NULL` values to non-null values. In this case, you can make up a rule that if the `end_date` is `NULL`, the end date is one week after the start date.

```
UPDATE tasks
SET
    end_date = start_date + 7
WHERE
    end_date IS NULL;
```

This query verifies the update:

```
SELECT * FROM tasks;
```

Third, add a `NOT NULL` constraint to the `end_date` column using the following `ALTER TABLE` (https://www.mysqltutorial.org/mysql-alter-table.aspx) statement:

```
ALTER TABLE table_name
CHANGE
   old_column_name
   new_column_name column_definition;
```

In this case, the name of the old and new column names are the same except that the column must have a `NOT NULL` constraint:

```
ALTER TABLE tasks
CHANGE
    end_date
    end_date DATE NOT NULL;
```

Let's verify the change by using the `DESCRIBE` (https://www.mysqltutorial.org/mysql-show-columns/) statement:

```
DESCRIBE tasks;
```

As you see, the `NOT NULL` constraint was added to the `end_date` column successfully.

Drop a NOT NULL constraint

To drop a `NOT NULL` constraint for a column, you use the `ALTER TABLE..MODIFY` statement:

```
ALTER TABLE table_name
MODIFY column_name column_definition;
```

Note that the column definition (column_definition) must restate the original column definition without the `NOT NULL` constraint.

For example, the following statement removes the `NOT NULL` constraint from the `end_date` column in the `tasks` table:

```
ALTER TABLE tasks
MODIFY
```

```
    end_date
    end_date DATE NOT NULL;
```

To ensure that the statement actually removed the `NOT NULL` constraint, you can use the `SHOW CREATE TABLE` command to view the full column definition:

Note that the `DESCRIBE` statement also does the trick:

```
DESCRIBE tasks;
```

In this tutorial, you have learned how to define a `NOT NULL` constraint for a column, add a `NOT NULL` constraint to a column, and remove a `NOT NULL` constraint from a column.