# 10 Essential MariaDB / MySQL DELETE Command Examples

When you are working on MySQL database, on several situations you may want to delete existing records from one or more tables.

In this tutorial, we'll explain how to use MySQL Delete with some useful examples.

The following are covered in this tutorial:

- Delete a specific row
- Delete multiple rows (using number column matching)
- Delete multiple rows (using string column matching)
- Delete records with LIMIT option
- Delete Vs Truncate (for deleting all rows)
- Delete records with IGNORE option
- When to use order by with Delete operation?
- Delete records with QUICK option
- Delete records with LOW_PRIORITY option
- Delete records from multiple tables using INNER join

To delete records from a table, you should have DELETE privilege on the particular table from where you are trying to delete the rows.

For this tutorial, we'll be using the following employee table as an example.

```
MariaDB [tgs]> SELECT * FROM employee;
+-----+--------+------------+--------+
| id  | name   | dept       | salary |
+-----+--------+------------+--------+
| 100 | Thomas | Sales      |   5000 |
| 200 | Jason  | Technology |   5500 |
| 300 | Mayla  | Technology |   7000 |
| 400 | Nisha  | Marketing  |   9500 |
| 500 | Randy  | Technology |   6000 |
+-----+--------+------------+--------+
```

If you are new to MySQL, you should probably first understand MySQL basics including how to create MySQL database.

# 1. Delete a Specific Row

To delete a specific row, you should specify the WHERE condition using one of the primary column value, or a unique column value.

The following example will delete one record from employee table which has the id of 100.

```
MariaDB [tgs]> DELETE FROM employee WHERE id = 100;
Query OK, 1 row affected (0.01 sec)
```

In the above output:

- The output of the delete command will show you how many records it has deleted. As you see from the above output, it says “1 row affected”, which means that it has deleted one row.
- The output will also say “Query OK” if the query was executed. If there is a syntax error, it will display it here. Even when it didn’t delete any record, this line will still say “Query OK” as long as there were no syntax error and the statement was clean. Finally, this will also show how long it took for MySQL to execute the query (for example: 0.01 seconds).

The first record with id of 100 is now deleted from the employee table.

```
MariaDB [tgs]> SELECT * FROM employee;
+-----+-------+------------+--------+
| id  | name  | dept       | salary |
+-----+-------+------------+--------+
| 200 | Jason | Technology |   5500 |
| 300 | Mayla | Technology |   7000 |
| 400 | Nisha | Marketing  |   9500 |
| 500 | Randy | Technology |   6000 |
+-----+-------+------------+--------+
```

# 2. Delete Multiple Rows Using String Column Matching

You can delete records from a table by matching a string column with certain criteria.

For string, you can use the keyword “like” which will do a partial matching. For partial matching use % in the value column.

The following example will delete records from employee table where the value in the dept column begins with “Tech”. So, this will delete all the records where the department is “Technology”.

```
MariaDB [tgs]> DELETE FROM employee WHERE dept LIKE 'Tech%';
Query OK, 3 rows affected (0.00 sec)
```

The above output indicates that it has deleted 3 records. As we see from the following output, we don’t see records with dept “Technology” anymore.

```
MariaDB [tgs]> SELECT * FROM employee;
+-----+--------+-----------+--------+
| id  | name   | dept      | salary |
+-----+--------+-----------+--------+
| 100 | Thomas | Sales     |   5000 |
| 400 | Nisha  | Marketing |   9500 |
+-----+--------+-----------+--------+
```

We discussed a lot about the various practical WHERE conditions in our MySQL select command tutorial. It is very helpful to understand how to use the WHERE clause effectively during DELETE statement: 25 Essential MySQL Select Command Examples

## 3. Delete Multiple Rows Using Number Column Matching

Similar to the above number column example, you can also delete records from a table by matching a string column with certain criteria.

The following will delete records from employee table where the salary is less than or equal to 700.

```
MariaDB [tgs]> DELETE FROM employee WHERE salary <=7000;
Query OK, 4 rows affected (0.00 sec)
```

The above command has deleted 4 records from our employee table. We currently have only one record left.

```
MariaDB [tgs]> SELECT * FROM employee;
+-----+-------+-----------+--------+
| id  | name  | dept      | salary |
+-----+-------+-----------+--------+
| 400 | Nisha | Marketing |   9500 |
+-----+-------+-----------+--------+
```

## 4. Delete Rows with LIMIT option

Even when a where clause is matching many records, you can still restrict how many records should be deleted using the LIMIT option.

This is very helpful when delete command takes a very long time, and you want to break-down the delete operation into multiple chunks. For example, if you are deleting over 100,000 records, you can use LIMIT 10000 to delete only 10,000 records at a time.

For example, The following delete command without any LIMIT option will delete all matching records. i.e This has deleted 5 records.

```
MariaDB [tgs]> DELETE FROM employee WHERE id > 1 ;
Query OK, 5 rows affected (0.00 sec)
```

But when you use LIMIT option, the following will delete only 2 records even when it matches 5 records.

```
MariaDB [tgs]> DELETE FROM employee WHERE id > 1 LIMIT 2;
Query OK, 2 rows affected (0.01 sec)
```

Execute the same delete command again, which will delete 2 more records.

```
MariaDB [tgs]> DELETE FROM employee WHERE id > 1 LIMIT 2;
Query OK, 2 rows affected (0.00 sec)
```

When you execute it again, this time it has deleted only one record, as this is the last matching records.

```
MariaDB [tgs]> DELETE FROM employee WHERE id > 1 LIMIT 2;
Query OK, 1 row affected (0.00 sec)
```

Now we don't have any more matching records to delete.

```
MariaDB [tgs]> DELETE FROM employee WHERE id > 1 LIMIT 2;
Query OK, 0 rows affected (0.00 sec)
```

# 5. Delete Vs Truncate (for deleting all rows)

When you want to delete all the rows from a table, you can use either DELETE or TRUNCATE command as shown below. Both will do the same thing. But, there is a difference.

```
MariaDB [tgs]> DELETE FROM employee;
Query OK, 5 rows affected (0.00 sec)

(or)

MariaDB [tgs]> TRUNCATE TABLE employee;
Query OK, 0 rows affected (0.00 sec)
```

The following are few things that you should know about DELETE vs TRUNCATE:

- Truncate is lot faster than delete. If you just want a quick way to empty a table, use truncate.
- Delete is a DML statement. Truncate is a DDL statement.
- When you execute Truncate command, it really does drop the table and re-create the empty table. This is the reason truncate is very fast, as it doesn't delete the records one-by-one like DELETE command.
- When there is a lock on a particular table, you cannot use truncate command.
- As you see from the above truncate command output, it doesn't show you how many rows are deleted. It will just say "0 rows affected" even when it truncates a table that had several rows.
- So, if you need to know how many records were deleted during the wipe-out of the table, you should use delete.
- If you have an on-delete trigger on the table, when you execute truncate command, those triggers won't be executed.

- Finally, an important difference is this: When you execute truncate command, you cannot rollback. But when you execute delete command, you can rollback. Truncate does an implicit commit.

# 6. Delete Rows with IGNORE Option

You can use IGNORE keyword along with delete command (i.e DELETE IGNORE) when you want to ignore the real valid error message that you know your particular DELETE command will throw.

So, when you use DELETE IGNORE, if there are any valid error message, it will be treated only as warning. You have to be very careful when you are using DELETE IGNORE, especially on a table where you have foreign key, as delete ignore will simply delete the records without worrying about any external foreign key dependencies.

```
MariaDB [tgs]> DELETE IGNORE FROM contractorbenefits WHERE id > 200;
Query OK, 3 rows affected, 1 warning (0.01 sec)
```

As you see from the above output, the delete ignore deleted the records from the table even though there was a foreign key dependency. It just treated those real error message as warnings.

# 7. Why use Order by with Delete?

You can use Order by clause with Delete command.

This can be very helpful when you want your rows to be deleted in a particular order.

In delete command, when you combine “order by” and “limit” option, you can do some clever tricks.

For example, when you execute the following delete command without the “order by” clause, this will delete the employee record with id 100 (i.e the oldest employee in the “Technology” department).

```
MariaDB [tgs]> DELETE FROM employee WHERE dept = 'Technology' LIMIT 1;
Query OK, 1 row affected (0.00 sec)
```

The above command has deleted employee id 200 (oldest technology employee).

```
MariaDB [tgs]> SELECT * FROM employee;
+-----+--------+------------+--------+
| id  | name   | dept       | salary |
+-----+--------+------------+--------+
| 100 | Thomas | Sales      |   5000 |
| 300 | Mayla  | Technology |   7000 |
| 400 | Nisha  | Marketing  |   9500 |
| 500 | Randy  | Technology |   6000 |
+-----+--------+------------+--------+
```

But, you can also delete the newest employee in the “Technology” department, if you combine ORDER by and LIMIT as shown below.

```
MariaDB [tgs]> DELETE FROM employee WHERE dept = 'Technology' ORDER BY id DESC
LIMIT 1;
```

```
Query OK, 1 row affected (0.00 sec)
```

The above command has deleted employee id 500 (newest technology employee).

```
MariaDB [tgs]> SELECT * FROM employee;
+-----+--------+------------+--------+
| id  | name   | dept       | salary |
+-----+--------+------------+--------+
| 100 | Thomas | Sales      |   5000 |
| 200 | Jason  | Technology |   5500 |
| 300 | Mayla  | Technology |   7000 |
| 400 | Nisha  | Marketing  |   9500 |
+-----+--------+------------+--------+
```

Also, keep in mind that you can use ORDER BY clause when you want to delete records in a particular sequence to avoid any referential integrity constraints.

# 8. Delete Rows with QUICK option

For a huge table that has some indexes, you can use the QUICK option to speed-up the delete operation as shown below.

```
MariaDB [tgs]> DELETE QUICK FROM employee WHERE id > 200;
Query OK, 3 rows affected (0.00 sec)
```

In Delete, when you use QUICK option, it doesn't merge index leaves, which may speed-up the delete operation depending on the kind of indexes you have on the table.

# 9. Delete Rows with LOW_PRIORITY option

The low priority keyword is very helpful when you are trying to delete records from a table that is getting used heavily.

The LOW_PRIORITY keyword will be effective only on these storage engines: MyISAM, MEMORY and MERGE. i.e You can use this only on the database storage engines that implement the table-level locking.

```
MariaDB [tgs]> DELETE LOW_PRIORITY FROM EMPLOYEE WHERE dept IN ('Sales',
'Marketing');
Query OK, 2 rows affected (0.01 sec)
```

When you use LOW_PRIORITY keyword, the delete operation will happen only when nobody else is reading from the table.

# 10. Delete from Multiple Tables using INNER Join

In all the previous examples, we used delete command to only delete records from a single table.

There are two ways you can specify deleting records from multiple tables.

In the first method, you can specify the tables from which the records to be deleted before the “FROM” keyword as shown below. The following will delete the records from both employee and benefits tables for the records that match the criteria specified in the where condition.

```
DELETE employee, benefits FROM employee
 INNER JOIN benefits INNER JOIN contractor
 WHERE employee.id=benefits.id AND benefits.id=contractor.id;
```

In the second method, you can specify the tables from which the records to be deleted after the “FROM” keyword as shown below. The following will delete the records from both employee and benefits for the records that match the criteria specified in the where condition.

```
DELETE FROM employee, benefits USING employee
 INNER JOIN benefits INNER JOIN contractor
 WHERE employee.id=benefits.id AND benefits.id=contractor.id;
```