VALUES is a DML statement introduced in MySQL 8.0.19 which returns a set of one or more rows as a table. In other words, it is a table value constructor which also functions as a standalone SQL statement.

```
VALUES row_constructor_list [ORDER BY column_designator] [LIMIT BY number]

row_constructor_list:
    ROW(value_list)[, ROW(value_list)][, ...]

value_list:
    value[, value][, ...]

column_designator:
    column_index
```

The VALUES statement consists of the VALUES keyword followed by a list of one or more row constructors, separated by commas. A row constructor consists of the ROW() row constructor clause with a value list of one or more scalar values enclosed in the parentheses. A value can be a literal of any MySQL data type or an expression that resolves to a scalar value.

ROW() cannot be empty (but each of the supplied scalar values can be NULL). Each ROW() in the same VALUES statement must have the same number of values in its value list.

The DEFAULT keyword is not supported by VALUES and causes a syntax error, except when it is used to supply values in an INSERT statement.

The output of VALUES is a table:

```
mysql> VALUES ROW(1,-2,3), ROW(5,7,9), ROW(4,6,8);
+----------+----------+----------+
| column_0 | column_1 | column_2 |
+----------+----------+----------+
|        1 |       -2 |        3 |
|        5 |        7 |        9 |
|        4 |        6 |        8 |
+----------+----------+----------+
3 rows in set (0.00 sec)
```

The columns of the table output from VALUES have the implicitly named columns column_0, column_1, column_2, and so on, always beginning with 0. This fact can be used to order the rows by column using an optional ORDER BY clause in the same way that this clause works with a SELECT statement, as shown here:

```
mysql> VALUES ROW(1,-2,3), ROW(5,7,9), ROW(4,6,8) ORDER BY column_1;
+----------+----------+----------+
| column_0 | column_1 | column_2 |
+----------+----------+----------+
|        1 |       -2 |        3 |
|        4 |        6 |        8 |
|        5 |        7 |        9 |
+----------+----------+----------+
3 rows in set (0.00 sec)
```

The VALUES statement also supports a LIMIT clause for limiting the number of rows in the output.

The VALUES statement is permissive regarding data types of column values; you can mix types within the same column, as shown here:

```
mysql> VALUES ROW("q", 42, '2019-12-18'),
    ->     ROW(23, "abc", 98.6),
    ->     ROW(27.0002, "Mary Smith", '{"a": 10, "b": 25}');
+----------+------------+--------------------+
| column_0 | column_1   | column_2           |
```