```
+------+---------------+--------------+
```

In MySQL 8.0.17 and later, the implicit copmparison of the extracted value with JSON integer 0 leads to a different result:

```
mysql> SELECT id, col, col->"$.val" FROM test WHERE col->"$.val" IS TRUE;
+------+----------------+--------------+
| id   | col            | col->"$.val" |
+------+----------------+--------------+
|    1 | {"val": true}  | true         |
|    2 | {"val": false} | false        |
+------+----------------+--------------+
```

Beginning with MySQL 8.0.21, you can use `JSON_VALUE()` on the extracted value to perform type conversion prior to performing the test, as shown here:

```
mysql> SELECT id, col, col->"$.val" FROM test
    ->     WHERE JSON_VALUE(col, "$.val" RETURNING UNSIGNED) IS TRUE;
+------+---------------+--------------+
| id   | col           | col->"$.val" |
+------+---------------+--------------+
|    1 | {"val": true} | true         |
+------+---------------+--------------+
```

Also beginning with MySQL 8.0.21, the server provides the warning `Evaluating a JSON value in SQL boolean context does an implicit comparison against JSON integer 0; if this is not what you want, consider converting JSON to a SQL numeric type with JSON_VALUE RETURNING` when comparing extracted values in an SQL boolean context in this manner.

- In MySQL 8.0.17 and later a `WHERE` condition having `NOT IN (`*`subquery`*`)` or `NOT EXISTS (`*`subquery`*`)` is transformed internally into an antijoin. (An antijoin returns all rows from the table for which there is no row in the table to which it is joined matching the join condition.) This removes the subquery which can result in faster query execution since the subquery's tables are now handled on the top level.

  This is similar to, and reuses, the existing `IS NULL` (`Not exists`) optimization for outer joins; see EXPLAIN Extra Information.