MySQL - Show Indexes

A MySQL Index is a type of special lookup table that is used to make data retrieval easier in a database. It points to the actual data in the database.

MySQL allows various types of indexes to be created on one or more columns in a table. They are:

- Primary Key Index
- Unique Index
- Simple Index
- Composite Index
- Implicit Index

To check if any of these indexes are defined on a table or not, MySQL provides the SHOW INDEX statement.

The MySQL SHOW INDEX Statement

The SHOW INDEX Statement of MySQL is used to list out the information about table index.

The vertical-format output (specified by \G) in MySQL often is used with this statement, to avoid a long line wraparound.

Syntax

Following is the basic syntax of the SHOW INDEX Statement −

```
SHOW INDEX FROM table_name;
```

Example

In this example, we are create a new table CUSTOMERS and adding a PRIMARY KEY index to one of its columns using the following CREATE TABLE query −

```
CREATE TABLE CUSTOMERS (
   ID INT NOT NULL,
   NAME VARCHAR (20) NOT NULL,
   AGE INT NOT NULL,
   ADDRESS CHAR (25),
   SALARY DECIMAL (18, 2),
   PRIMARY KEY(ID),
   INDEX(NAME)
);
```

Now, we can display the indexes present on the CUSTOMERS table using the following SHOW INDEX query −

```
SHOW INDEX FROM CUSTOMERS\G
```

Output

The vertical-output will be displayed as –

```
*************************** 1. row ************************
       Table: customers
   Non_unique: 0
     Key_name: PRIMARY
 Seq_in_index: 1
  Column_name: ID
    Collation: A
  Cardinality: 0
     Sub_part: NULL
       Packed: NULL
         Null:
   Index_type: BTREE
      Comment:
Index_comment:
      Visible: YES
   Expression: NULL
```

```
*************************** 2. row ***********************
       Table: customers
  Non_unique: 1
    Key_name: NAME
Seq_in_index: 1
 Column_name: NAME
   Collation: A
 Cardinality: 0
    Sub_part: NULL
      Packed: NULL
        Null:
  Index_type: BTREE
     Comment:
Index_comment:
     Visible: YES
  Expression: NULL
2 rows in set (0.01 sec)
```

With IN Clause

In this example, let us first create an index on the AGE column of CUSTOMERS table using the following CREATE INDEX query −

```
CREATE INDEX AGE_INDEX ON CUSTOMERS (AGE);
```

You can also retrieve the information by specifying the database name as −

```
SHOW INDEX IN CUSTOMERS FROM sample\G
```

Output

The output will be the same as above −

```
*************************** 1. row ***************************
        Table: customers
   Non_unique: 0
     Key_name: PRIMARY
 Seq_in_index: 1
  Column_name: ID
    Collation: A
  Cardinality: 0
     Sub_part: NULL
       Packed: NULL
         Null:
   Index_type: BTREE
      Comment:
Index_comment:
      Visible: YES
   Expression: NULL
*************************** 2. row ***************************
        Table: customers
   Non_unique: 1
     Key_name: NAME
 Seq_in_index: 1
  Column_name: NAME
```

```
    Collation: A
  Cardinality: 0
     Sub_part: NULL
       Packed: NULL
         Null:
   Index_type: BTREE
      Comment:
Index_comment:
      Visible: YES
   Expression: NULL
2 rows in set (0.01 sec)
```

With WHERE Clause

As the indexes are displayed in a table format, we can use a WHERE clause with SHOW INDEX statement to retrieve specified indexes matching a given condition.

```
SHOW INDEX IN CUSTOMERS WHERE Column_name = 'NAME'\G
```

Output

The index created on NAME column is displayed −

```
*************************** 1. row *************************
        Table: customers
   Non_unique: 1
     Key_name: NAME
 Seq_in_index: 1
  Column_name: NAME
    Collation: A
  Cardinality: 0
     Sub_part: NULL
       Packed: NULL
         Null:
   Index_type: BTREE
      Comment:
Index_comment:
      Visible: YES
   Expression: NULL
1 row in set (0.00 sec)
```