**How to Create Index in MySQL**

An index is a data structure that allows us to add indexes in the existing table. It enables you to improve the faster retrieval of records on a database table. It creates an entry for each value of the indexed columns. We use it to quickly find the record without searching each row in a database table whenever the table is accessed. We can create an index by using one or more columns of the table for efficient access to the records.

When a table is created with a primary key or unique key, it automatically creates a special index named PRIMARY. We called this index as a clustered index. All indexes other than PRIMARY indexes are known as a non-clustered index or secondary index.

**Need for Indexing in MySQL**

Suppose we have a contact book that contains names and mobile numbers of the user. In this contact book, we want to find the mobile number of Martin Williamson. If the contact book is an unordered format means the name of the contact book is not sorted alphabetically, we need to go over all pages and read every name until we will not find the desired name that we are looking for. This type of searching name is known as sequential searching.

**To find the name and contact of the user from table contactbooks, generally, we used to execute the following query:**

mysql> SELECT mobile\_number FROM contactbooks WHERE first\_name = 'Martin' AND last\_name = 'Taybu';

This query is very simple and easy. Although it finds the phone number and name of the user fast, the database searches entire rows of the table until it will not find the rows that you want. Assume, the contactbooks table contains millions of rows, then, without an index, the data retrieval takes a lot of time to find the result. In that case, the database indexing plays an important role in returning the desired result and improves the overall performance of the query.

**MySQL CREATE INDEX Statement**

Generally, we create an index at the time of table creation in the database. The following statement creates a table with an index that contains two columns col2 and col3.

mysql> CREATE TABLE t\_index(

col1 INT PRIMARY KEY,

col2 INT NOT NULL,

col3 INT NOT NULL,

col4 VARCHAR(20),

INDEX (col2,col3)

);

**If we want to add index in table, we will use the CREATE INDEX statement as follows:**

mysql> CREATE INDEX [index\_name] ON [table\_name] (column names)

In this statement, index\_name is the name of the index, table\_name is the name of the table to which the index belongs, and the column\_names is the list of columns.

Let us add the new index for the column col4, we use the following statement:

mysql> CREATE INDEX ind\_1 ON t\_index(col4);

**SN Storage Engine Index Type**

1. InnoDB BTREE

2. Memory/Heap HASH, BTREE

3. MYISAM BTREE

**Example**

In this example, we are going to create a table student and perform the CREATE INDEX statement on that table.

Table Name: student



Now, execute the following statement to return the result of the student whose class is CS branch:

mysql> SELECT studentid, firstname, lastname FROM student WHERE class = 'CS';

This statement will give the following output:



In the above table, we can see the four rows that are indicating the students whose class is the CS branch.

If you want to see how MySQL performs this query internally, execute the following statement:

mysql> EXPLAIN SELECT studentid, firstname, lastname FROM student WHERE class = 'CS';

You will get the output below. Here, MySQL scans the whole table that contains seven rows to find the student whose class is the CS branch.



Now, let us create an index for a class column using the following statement.

mysql> CREATE INDEX class ON student (class);

After executing the above statement, the index is created successfully. Now, run the below statement to see how MySQL internally performs this query.

mysql> EXPLAIN SELECT studentid, firstname, lastname FROM student WHERE class = 'CS';

The above statement gives output, as shown below:



In this output, MySQL finds four rows from the class index without scanning the whole table. Hence, it increases the speed of retrieval of records on a database table.

If you want to show the indexes of a table, execute the following statement:

mysql> SHOW INDEXES FROM student;



**MySQL Drop Index**

MySQL allows a DROP INDEX statement to remove the existing index from the table. To delete an index from a table, we can use the following query:

mysql>DROP INDEX index\_name ON table\_name [algorithm\_option | lock\_option];

**DROP INDEX class ON student;**

Example

First, execute the following command to show the indexes available in the table.

**mysql> SHOW INDEXES FROM student;**

**It will give the following output.**

