**MySQL CREATE INDEX Statement**

**The CREATE INDEX statement is used to create indexes in tables.**

Indexes are used to retrieve data from the database more quickly than otherwise. The users cannot see the indexes, they are just used to speed up searches/queries.

**Note:** Updating a table with indexes takes more time than updating a table without (because the indexes also need an update). So, only create indexes on columns that will be frequently searched against.

**CREATE INDEX Syntax**

**Creates an index on a table. Duplicate values are allowed:**

**CREATE INDEX *index\_name*  
ON *table\_name* (*column1*, *column2*, ...);**

**CREATE UNIQUE INDEX Syntax**

Creates a unique index on a table. Duplicate values are not allowed:

**CREATE UNIQUE INDEX *index\_name*  
ON *table\_name* (*column1*, *column2*, ...);**

MySQL CREATE INDEX Example

The SQL statement below creates an index named "idx\_lastname" on the "LastName" column in the "Persons" table:

CREATE INDEX idx\_lastname  
ON Persons (LastName);

If you want to create an index on a combination of columns, you can list the column names within the parentheses, separated by commas:

CREATE INDEX idx\_pname  
ON Persons (LastName, FirstName);

DROP INDEX Statement

The DROP INDEX statement is used to delete an index in a table.

ALTER TABLE *table\_name*DROP INDEX *index\_name*;

**OR**

**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:

1. 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.

1. mysql> **CREATE** **TABLE** t\_index(
2. col1 **INT** **PRIMARY** **KEY**,
3. col2 **INT** NOT NULL,
4. col3 **INT** NOT NULL,
5. col4 **VARCHAR**(20),
6. **INDEX** (col2,col3)
7. );

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

1. 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:

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

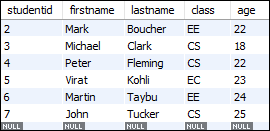
By default, [MySQL](https://www.javatpoint.com/mysql-tutorial) allowed index type **BTREE** if we have not specified the type of index. The following table shows the different types of an index based on the storage engine of the table.

|  |  |  |
| --- | --- | --- |
| **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.

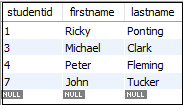
**Table Name: student**



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

1. 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:

1. 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.

How to Create Index in MySQL

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

1. 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.

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

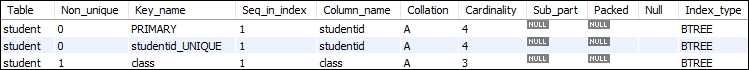
The above statement gives output, as shown below:

How to Create Index in MySQL

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:

1. mysql> SHOW INDEXES **FROM** student;



It will give the following output.