# Indexes

An index is a schema object. It is used by the server to speed up the retrieval of rows by using a pointer. It can reduce disk I/O(input/output) by using a rapid path access method to locate data quickly.

An index helps to speed up select queries and where clauses, but it slows down data input, with the update and the insert statements. Indexes can be created or dropped with no effect on the data. In this article, we will see how to create, delete, and use the INDEX in the database.

## Why SQL Indexing is Important?

Indexing is an important topic when considering advanced MySQL, although most people know about its definition and usage they don’t understand when and where to use it to change the efficiency of our queries or stored procedures by a huge margin.

Here are some scenarios along with their explanation related to Indexing:

- **When executing a query on a table with huge data (> 100,000 rows)** − MySQL performs a full table scan, which takes much time and may lead to server timeouts. To avoid this, always use the EXPLAIN option for the query within MySQL. This shows the state of execution, including which columns are being used and potential performance issues with large data sets. Check the order of columns in conditions and their indexing.
- **The order of the index is crucial** − By creating an index with the same order as the columns used in conditions, you can maximize query speed. For instance, if a query joins tables based on customer_id and order_date, creating a single index on (customer_id, order_date) can be beneficial. This single index can be used for multiple queries and saves storage.
- **Avoid creating an index for each query** − Creating too many indexes can consume storage and create performance issues with large data sets. It’s important to index columns that are frequently used in queries and to avoid indexing rarely used columns. Periodically review and remove unnecessary indexes.
- **Indexes improve performance with sorting and grouping** − Creating an index on a column frequently used for sorting or grouping can significantly improve performance. The index allows MySQL to quickly access, sort, or group data, avoiding full table scans.
- **MySQL may not always use an index** − Sometimes, the query optimizer may choose a full table scan over using an index if it determines that a full scan is faster.

## Conclusion

- **Data retrieval is speeded up by the usage of indexes** − Indexes act as a table of contents for database rows, enhancing query speed.
- **They function as a database row’s table of contents** − Indexes provide a quick way to locate data without scanning the entire table.
- **Enhance query speed, however, inserts and updates might take longer** − While indexes improve query performance, they can slow down insert and update operations due to the overhead of maintaining the indexes.
- **Bitmap, Hash, and B-tree indexes are examples of typical types** − These are common types of indexes used to optimize different query patterns and data structures.
- **Indexes must be carefully selected and kept up-to-date** − Proper selection and maintenance of indexes are crucial for smooth database operations and optimal

performance.