SQL Indexing

*In simple terms, an index in SQL is a tool used to quickly identify rows with specific column values. If there were no indexes, the SQL server would have to start with the first row and then go through the entire table until it discovers the relevant rows. This method is known as a full-table scan and can be highly inefficient for large tables.*

***Why are Indexes in SQL Used?***

* *Improved Query Performance: The primary reason for using indexes is to accelerate query processing. Indexes can drastically reduce the amount of data the server needs to examine.*
* *Efficient Data Access: Indexes provide a quick way to access row data for SELECT statements. This is particularly beneficial for tables with a large number of rows.*
* *Sorting and Grouping Speed: Indexes improve the speed of data retrieval operations by providing a sorted version of the data, which is faster to process for ORDER BY and GROUP BY operations.*
* *Unique Constraints: Indexes can be used to enforce uniqueness for columns to ensure that no two rows of a table have duplicate values in a particular column or a combination of columns.*
* *Optimized Join Operations: In databases with multiple tables, indexes improve the speed of join operations by quickly locating the joining rows in each table.*

***Types of Indexes –***

* *Primary - A primary key index is an automatically generated index associated with the primary key column.*
* *Unique Key Index - A unique index in a relational database is a data structure that enforces the uniqueness constraint on one or more columns within a table.*
* *Clustered Index - A clustered index sorts and stores the rows of a table based on the values in one or more specified columns.*

*A clustered index is created only when both the following conditions are satisfied:*

* + - * *The data or file, that you are moving into secondary memory should be in sequential or sorted order.*
      * *There should be a key value, meaning it cannot have repeated values.*

*Whenever you apply clustered indexing in a table, it will perform sorting in that table only. You can create only one clustered index in a table like a primary key.*

*A clustered index is as same as a dictionary where the data is arranged in alphabetical order.*

* *Non-Clustered Index - The non-Clustered Index is like the index of a book. The data is stored in one place, and the index is stored in another place. Since the data and non-clustered index is stored separately, then you can have multiple non-clustered indexes in a table.*

*In a non-clustered index, the index contains the pointer to data.*

| **CLUSTERED INDEX** | **NON-CLUSTERED INDEX** |
| --- | --- |
| A clustered index is faster. | A non-clustered index is slower. |
| The clustered index requires less memory for operations. | A non-Clustered index requires more memory for operations. |
| In a clustered index, the clustered index is the main data. | In the Non-Clustered index, the index is the copy of data. |
| A table can have only one clustered index. | A table can have multiple non-clustered indexes. |
| The clustered index has the inherent ability to store data on the disk. | A non-Clustered index does not have the inherent ability to store data on the disk. |
| Clustered index store pointers to block not data. | The non-clustered index stores both the value and a pointer to the actual row that holds the data |
| In Clustered index leaf nodes are actual data itself. | In Non-Clustered index leaf nodes are not the actual data itself rather they only contain included columns. |
| In a Clustered index, Clustered key defines the order of data within a table. | In a Non-Clustered index, the index key defines the order of data within the index. |
| A Clustered index is a type of index in which table records are physically reordered to match the index. | A Non-Clustered index is a special type of index in which the logical order of the index does not match the physical stored order of the rows on the disk. |
| The size of The primary clustered index is large. | The size of the non-clustered index is compared relativelyThe composite is smaller. |
| Primary Keys of the table by default are clustered indexes. | The [composite key](https://www.geeksforgeeks.org/composite-key-in-sql/) when used with unique constraints of the table act as the non-clustered index. |

***Disadvantages of Index*** *–*

*Apart from these advantages of Indexes in SQL, they have some limitations too, like:*

* *Overuse of Indexes: While indexes speed up data retrieval, they can slow down data input, through INSERT, UPDATE, and DELETE statements. Each index needs to be updated when data is modified.*
* *Storage Space: Indexes consume additional disk space.*
* *Maintenance Overhead: Indexes need to be maintained and rebuilt over time, which can add overhead to database maintenance routines.*

***When Should We Avoid Using Indexes?***

*The purpose of indexes is to enhance a database’s performance, there are multiple scenarios where indexes should not be used.*

* *Indexes should not be used on tables containing few records.*
* *Tables that have frequent, large batch updates or insert operations.*
* *Indexes should not be used on columns that contain a high number of NULL values.*
* *Indexes should not be used on the columns that are frequently manipulated.*

***Indexes will be useful if****:*

* *The column or columns have a high degree of uniqueness.*
* *You frequently need to look for a certain value or range of values for the column.*

***They will not be useful if****:*

* *You are selecting a large % (>10-20%) of the rows in the table.*
* *The additional space usage is an issue.*
* *You want to maximize insert performance. Every index on a table reduces insert and update performance because they must be updated each time the data changes.*

*Primary key columns are typically great for indexing because they are unique and are often used to lookup rows.*