**Indexing**

Why to use index ?

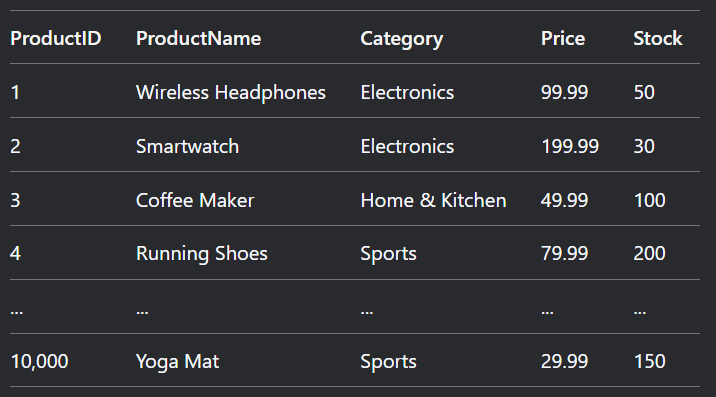
To locate the data quickly…….. (To speed up data-retrieval operation)

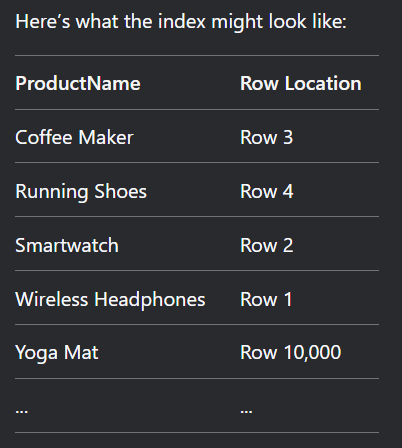
What is index ?

Index is a data-structure that stores, data of selected column, in SORTED manner, with its reference which is pointing towards the actual row in the table.

Note - Indexing is used only when our database is large.

Original Table





Because in original table it has to search each row(10,000) …….but in index, data is stored in sorted manner, so database will perform a binary search to find data this will reduce the time complexity from O(n) to O(log n). The moment we find data in index, that data will have its row number (pointing towards the actual row in the original table). Database will go to that row and fetch all the information related to that data.

For example – In above table, if we have to find Yoga Mat Price, if we search in original table we have to search linearly means we have to search each row means O(n) time complexity….. but if we search in index, products are stored in sorted manner alphabetically, so we can perform binary search and find the data(YogaMat) in O(log n) time complexity… And YogaMat will have its row number, pointing towards the actual row in original table, database will go directly to that row number in original table and find all the data related to Yoga Mat.

How to make Index?

Syntax

CREATE INDEX nameOfIndex  
 ON TableName (columnName1, columnName2);

Note – if we do not mention Clustered or Non-Clustered, by default non-clustered index will be created.

**Types Of Indexing**

* + 1. **Clustered Index (primary index)**

Column selected for Clustered Index will decide the sequential order of table(because that column will be sorted)….

Only one clustered index is allowed for one table(because in one table only one column can be sorted)

No separate index table is created in clustered indexing. Because our original table is sorted.

* Usually Clustering index is created on Primary Key Column because primary key is generally sorted and decides the sequential order of table but we can use any column of the table.
* If we use any other column to create clustered index then our original table sequence will change and original table will be sorted according to the column which is selected for creating clustered index and not on the bases of primary key.

Note- In some databases clustered index is automatically created while defining primary key.

If we use a key attribute to make clustered index?

* No changes in original table because we are using primary key and our table is already sorted according to primary key.

For example

How the Searching works?

Binary search will be used in original table to find any value because all values are unique which led to decrease in time complexity from O(n) to log(n).

If we use a non-key attribute to make clustered index, how index table look like?

* So firstly sequence of our original table changes, our original table will be sorted according to the column which we used for clustered index and not primary key.
* Clustered Index means that column will be sorted but may be that column has repeating values as it is Non-key attribute.

For example

📌 **How Searching Works?**  
 First Binary search to find first occurrence then linear search on that group to find particular data.

Syntax

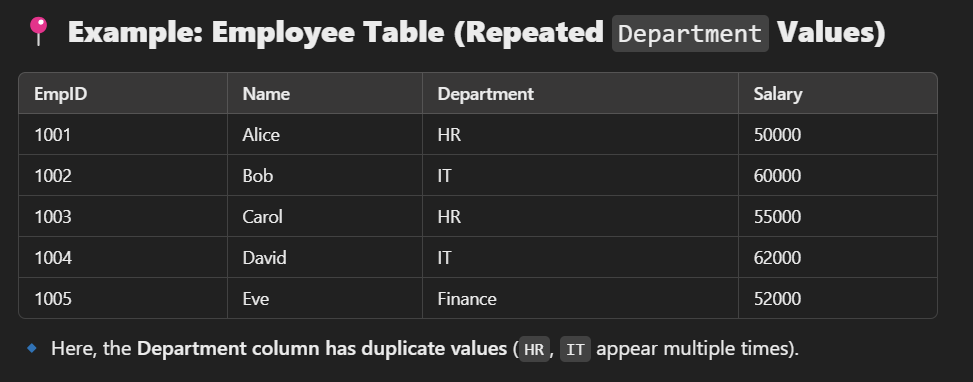
CREATE CLUSTERED INDEX index\_name ON table\_Name(column\_name ASC/DESC);

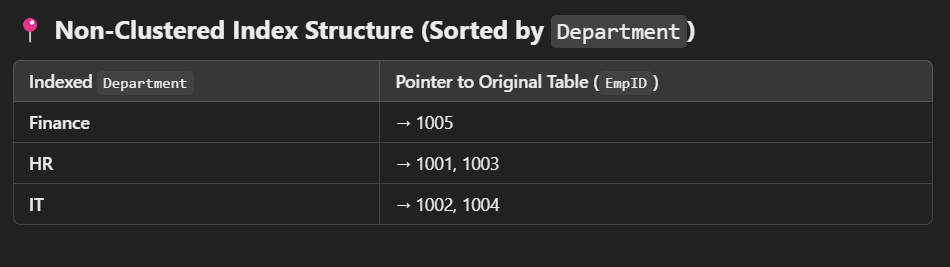
* + 1. **Non-clustered Index (secondary index)**
* Applied on those columns, which are unsorted ….. We can only sort one column in one table rest of the column becomes unsorted.
* A table can have multiple non-clustered index.
* No change in original table

How index table looks like ?

* In index table we will store unique values in sorted manner with all the reference, of that value present in table, in linked list like structure. This can be called dense indexing.

For example





Syntax

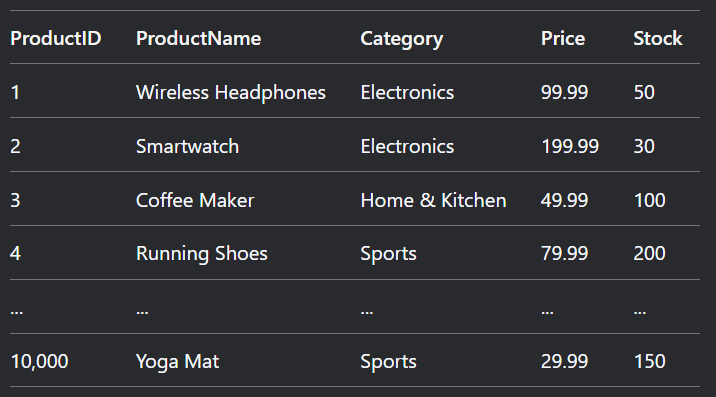
CREATE NONCLUSTERED INDEX index\_name ON table\_name(column\_name ASC/DESC) ;

**Dense Index**

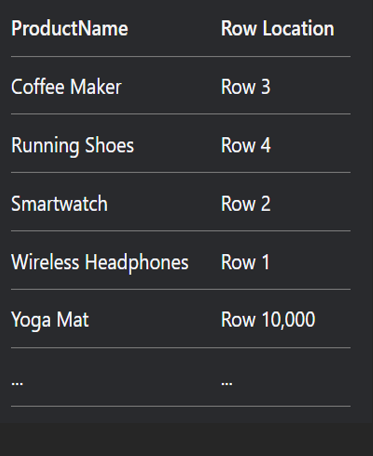
* Every unique value of original table will be stored in Index.

No of values in index table = No of unique values in original table

Example of dense index Original table



Dense Index

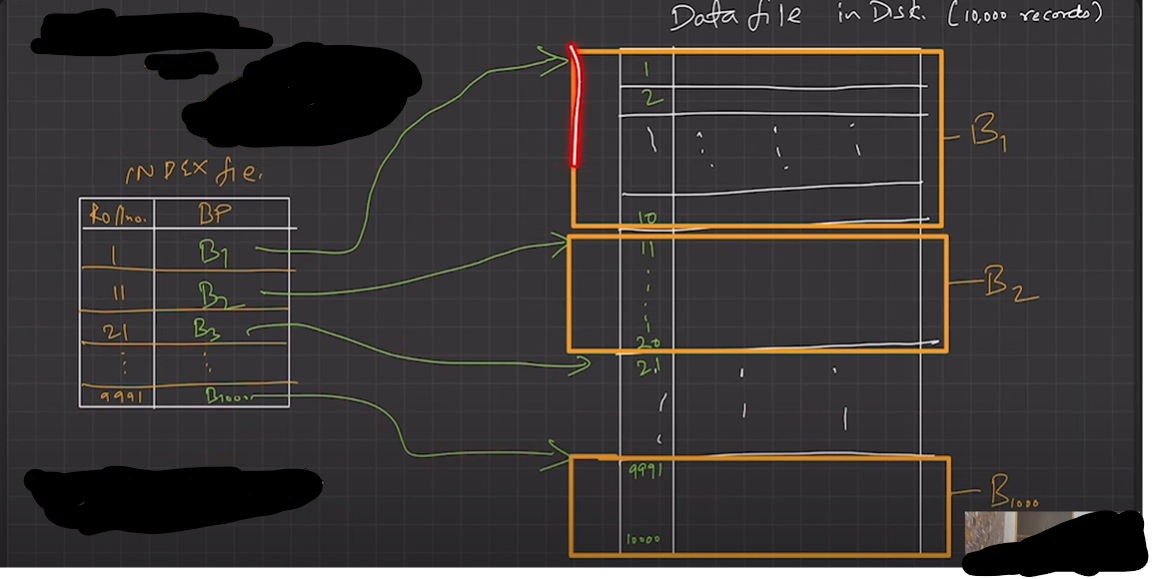


**Sparse Index**

* In this we create block and only first record of each block will be stored in index table with its block address

No of values in index table = no of blocks in original table

Example of sparse Index



In this example each block has 10 records, total 1000 blocks and only first record of each block will be stored in index table with its Block number.



**Multi-level Indexing**

What ?

When we create multiple level of indexing is called multi-level indexing.

Why ?

If single index become so large that binary search itself will take time then we add one more layer of indexing.

