Indexing

A database index is a data structure that improves the speed of data retrieval operations on a database table at the cost of additional writes and storage space to maintain the index data structure.
It is a data structure that is added to a file to provide faster access to the data.
It reduces the number of blocks that the DBMS has to check.

How do DBMS access data?

The operations read, modify, update, and delete are used to access data from the database. DBMS must first transfer the data temporarily to a buffer in main memory. Data is then transferred between disk and main memory into units called blocks.

Two Types Of Indices

Ordered index (Primary index or clustering index) – which is used to access data sorted by order of values.

Hash index (secondary index or non-clustering index) - used to access data that is distributed uniformly across a range of buckets.

Ordered Index

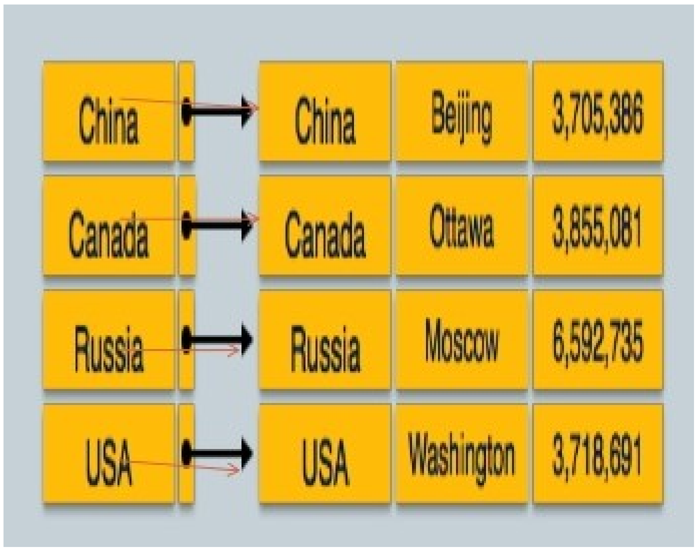
In an ordered index, index entries are stored sorted on the search key value. Example, the author catalog in the library. The indices are usually sorted to make searching faster.

Example: Suppose we have a student table with thousands of records and each of which is 10 bytes long. If their IDs start with 1, 2, 3....and so on and we have to search students with ID-555.
In the case of a database with no index, we have to search the disk block from starting till it reaches 543. The DBMS will read the record after reading 555*10=5550 bytes.

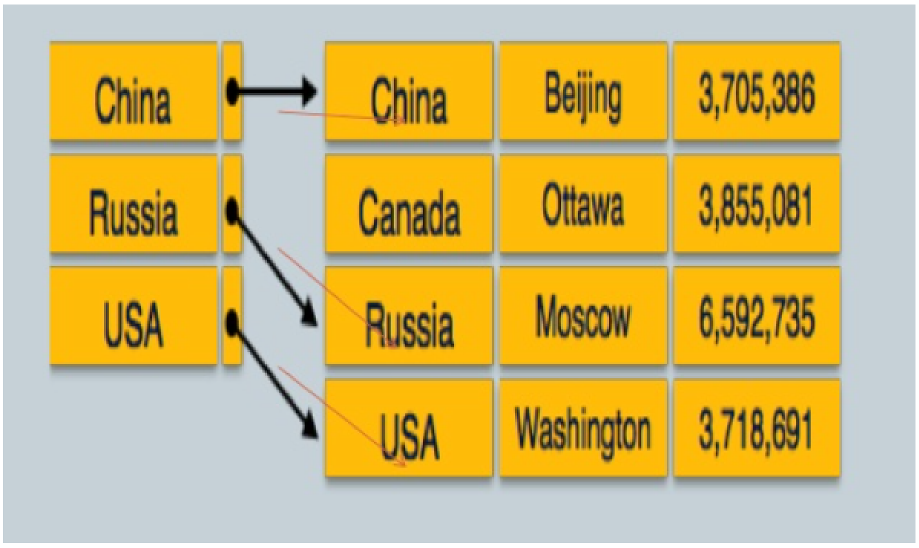
Primary index:
In a sequentially ordered file, the index whose search key specifies the sequential order of the file. Also called clustering index The search key of a primary index is usually but not necessarily the primary key. The primary index can be classified into two types: Dense index and Sparse index.

Dense index:

- An index record appears for **every** search key value in file.
- This record contains a search key value and a pointer to the actual record.
- It needs more space to store the index record itself. The index records have the search key and a pointer to the actual record on the disk.



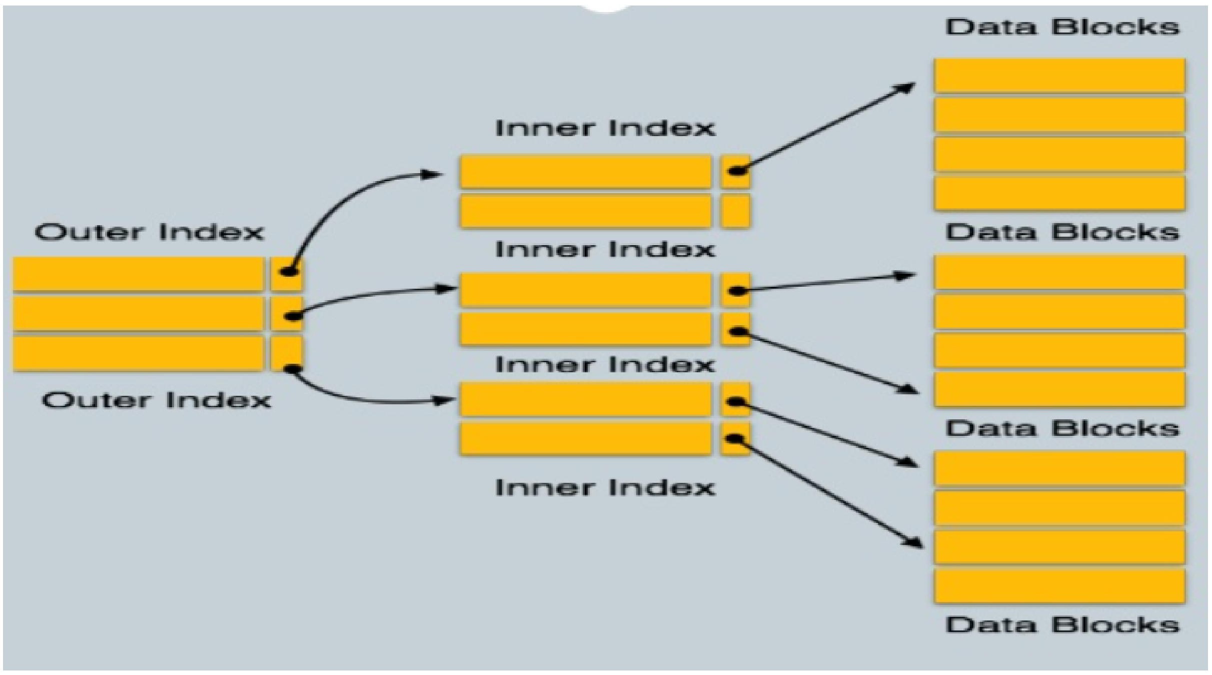
Sparse Index:
It contains index records for only some search-key values.
Applicable when records are sequentially ordered on search-key.
To locate a record with search-key value K we: Find index record with largest search-key value < K Search file sequentially starting at the record to which the index record points.



Compared to dense indices:
Less space and less maintenance overhead for insertions and deletions.
Generally slower than dense index for locating records.
Good tradeoff: sparse index with an index entry for every block in file, corresponding to least search-key value in the block.

Multilevel Indexing

In this method, we can see that index mapping growth is reduced to a considerable amount. But this method can also have the same problem as the table size increases. In order to overcome this, we can introduce multiple levels between primary memory and secondary memory. This method is also known as multilevel indexing. In this method, the number of secondary level indexes is two or more.



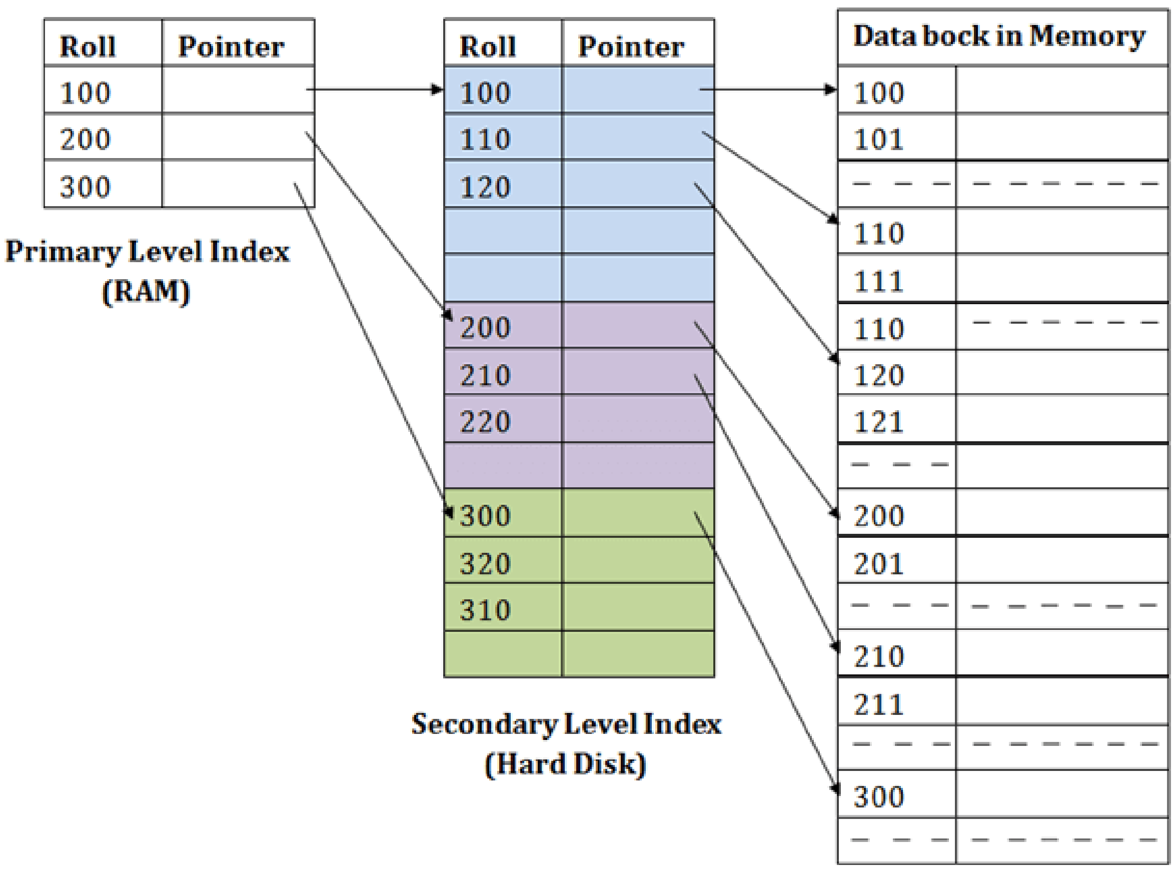
Secondary index/Non-cluster

An index whose search key specifies an order different from the sequential order of the file. Also called non-clustering index.

If the search key of a secondary index is not a candidate key, it is not enough to point to just the first record with each search-key value because the remaining records with the same search-key value could be anywhere in the file. Therefore, a secondary index must contain pointers to all the records.

Secondary indices must be dense, with an index entry for every search-key value, and a pointer to every record in the file.

Secondary indices improve the performance of queries on non-primary keys.



For example:

- If you want to find the record of roll 111 in the diagram, then it will search the highest entry which is smaller than or equal to 111 in the first level index. It will get 100 at this level.
- Then in the second index level, again it does max (111) <= 111 and gets 110. Now using the address 110, it goes to the data block and starts searching each record till it gets 111.
- This is how a search is performed in this method. Inserting, updating or deleting is also done in the same manner.