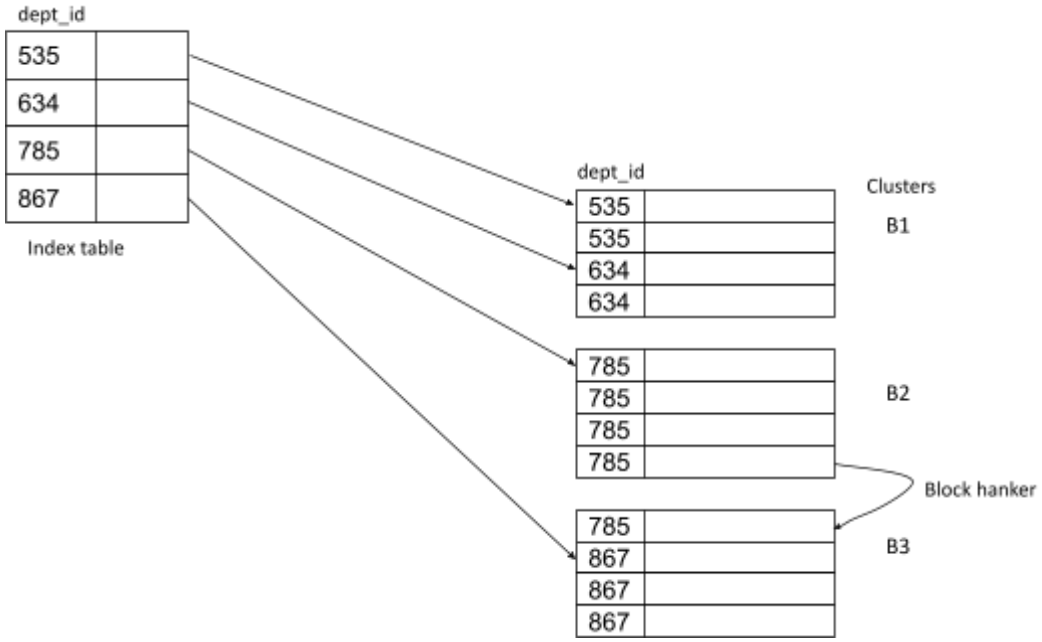
Types of Indexing:

- **Clustering Index:-** It is the index that is created and ordered on the basis of the non-primary key attributes of the table which are not known to be unique for each record. In Clustering index, to fetch a record we group two or more attributes together to get the unique values and create an index out of them**.**

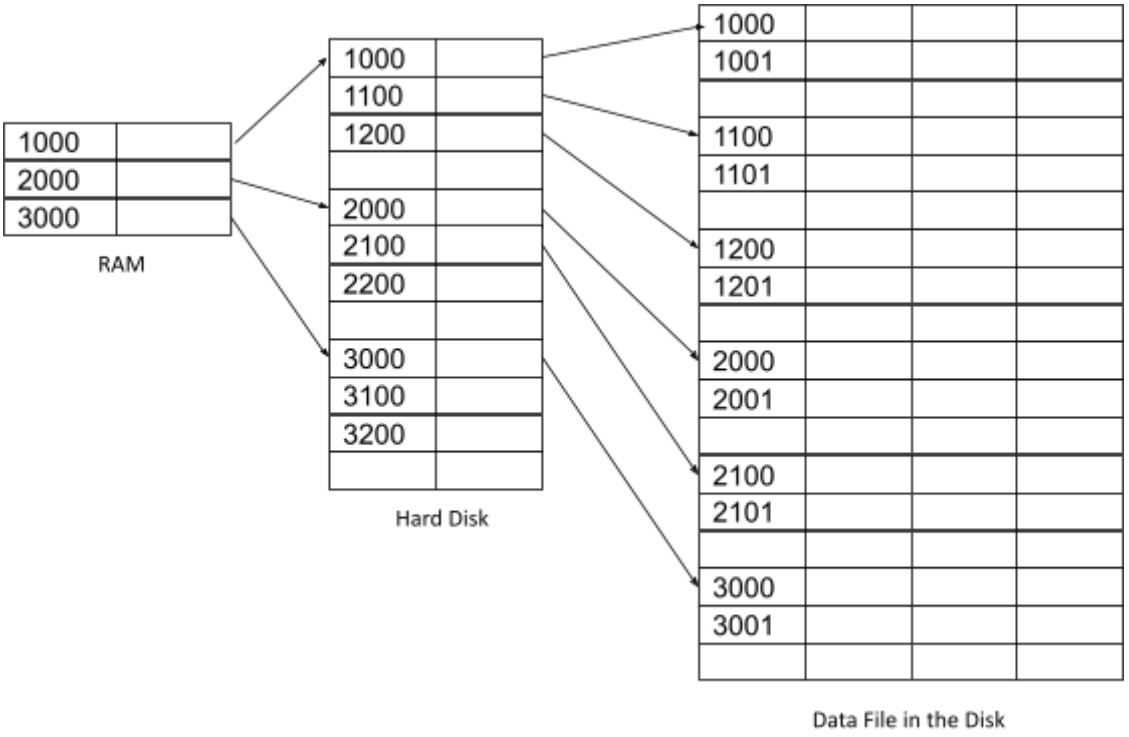
  Example: Suppose a Dental college contains several students posted in a particular department.
  Suppose we use a clustering index, where all Students which are posted in the same department (i.e. same dept_id) are considered within a single cluster, and index pointers point to the cluster as a whole.
  Also, here dept_id is a non-unique key.



  Here, we have 3 memory blocks as clusters. We can see two id's present in a single block and also a case when all the records corresponding to one key cannot be contained in one block it gets stored in another blck and is connected to the previous block with another pointer named Block Hanker.
  We can see, The pointer from the index table points to the first record of the main table corresponding to that particular key id.
  These cluster blocks are present in the main memory.

➢ **Secondary Index:-** It is introduced to tackle the problem when the size of datafile increases i.e. size of table increases, sparse indexing starts to slow down. Now to overcome this we introduce another level of indexing.
In this, we select a huge chunk of columns initially and put it up at the first level of indexing that is Primary Level index, it is stored in primary memory.
Now each chunk is divided into smaller ranges in the second level of indexing, this is stored in the secondary memory along with the actual datafile.

Example:



In this, say if we are looking for a record with search key value S. We have to find an index record with the largest search key value less than or equal to S.
Remember, this searching of the record in the file happens sequentially.
For example we need to find the record corresponding to the key value 2101.
We will start looking for the key, in the first index table present in the RAM, whose value will be either less than or equal to 2101.
So, upon searching we will fetch key 2000 from the first index table.

Now we will perform the same step of searching for the second index table stored at Disk as we did above for the first index table.
Similarly, upon searching we will fetch key 2100 from the second index table.
We will move to the third table in the data file and again repeat the very first step for searching.
Now, upon searching we will find out the desired key, i.e. 2101.
Remember the above procedure will keep continuing until the key is found. (if it exists in the table.)

Note: This procedure is also the same for DML operations like Insert, Update and Delete.

Hence, we need Indexing because,

- We can access and retrieve data faster.
- Indexing helps us with sorting the data, as index is a sorted data structure.
- Indexing reduces the number of I/O operations needed to be performed for retrieving data.

Drawbacks of Indexing:

- They take up additional space to store the index table.
- Indexing slows down INSERT,UPDATE and DELETE operations.