Chapter 8: File Organization and Indexing

Physical Data Storage:
A computer system has a well-defined hierarchy of memory. A CPU has direct access to its main memory as well as its inbuilt registers. The access time of the main memory is obviously less than the CPU speed.

To minimize this speed mismatch, cache memory is introduced. Cache memory provides the fastest access time and it contains data that is most frequently accessed by the CPU.

The memory with the fastest access is the costliest one. Larger storage devices offer slow speed and they are less expensive, however they can store huge volumes of data as compared to CPU registers or cache memory.

Memory Hierarchy for Computer System:

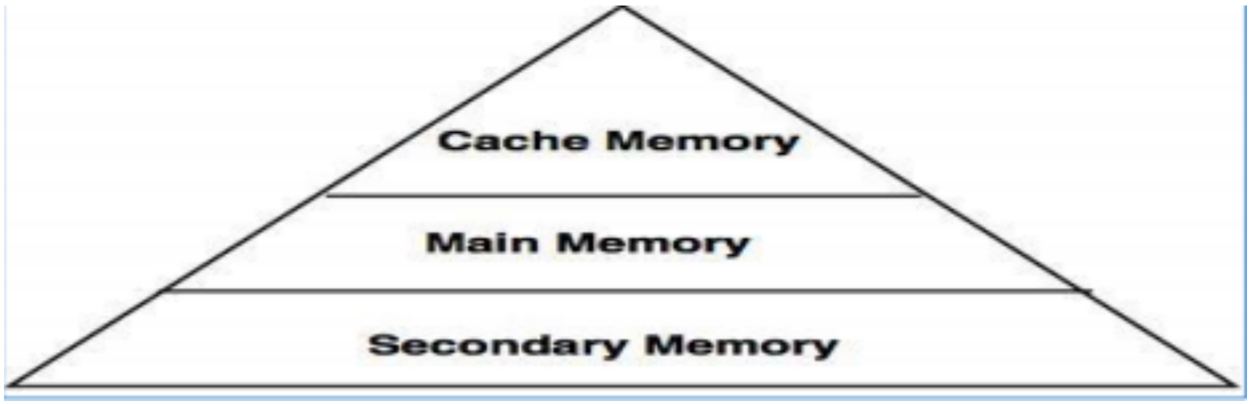


Fig: Memory Hierarchy

1. Cache Memory and Main memory:

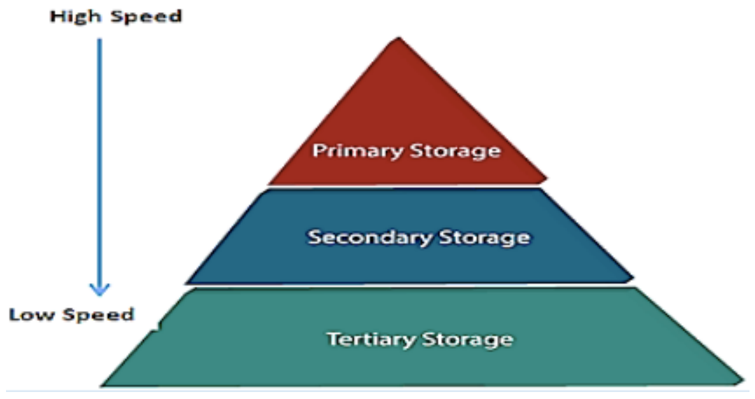
Cache memory and main memory are at the top level in the memory hierarchy which are **responsible for fast execution**. Example: RAM (Random Access Memory), ROM (Read-only memory) etc.

2. **Secondary Memory:**

Secondary memory or storage is used to store data in computer system. The secondary storage is relatively slower than cache or main memory. Example: **Magnetic tape, hard disk, CD, DVD etc.**

Data Storage Methods:

Databases are stored in file formats, which contain records. At physical level, the actual data is stored in electromagnetic format on some device. These storage devices can be broadly categorized into three types.



1. Primary Storage:

It is the primary area that offers quick access to the stored data. We also know the primary storage as volatile storage. It is because this type of memory does not permanently store the data.

As soon as the system leads to a power cut or a crash, the data also get lost. Main memory and cache are the types of primary storage.

a**. Main Memory**

It is the one that is responsible for operating the data that is available by the storage medium. The main memory handles each instruction of a computer machine.
This type of memory can store gigabytes of data on a system but is small enough to carry the entire database. At last, the main memory loses the whole content if the system shuts down because of power failure or other reasons.

b. Cache:

It is one of the costly storage media but it is the fastest one. A cache is a tiny storage media which is maintained by the computer hardware usually.

While designing the algorithms and query processors for the data structures, the designers keep concern on the cache effects.

2. Secondary Storage:
Secondary storage is also called as online storage. It is the storage area that allows the user to save and store data permanently. This type of memory does not lose the data due to any power failure or system crash. That's why we also call it non-volatile storage.

There are some commonly described secondary storage media which are available in almost every type of computer system:

a. Flash Memory:
A flash memory stores data in USB(Universal Serial Bus) keys which are further plugged into the USB slots of a computer system. These USB keys help transfer data into a computer system, but varies in size limits.

Unlike the main memory, it is possible to get back the stored data which may be lost due to a power cut or other reasons. This type of memory storage is most commonly used in the server system of caching the frequently used data.

This leads the system towards high performance and is capable of storing large amounts of databases than the main memory.

b. Magnetic Disk Storage:
This type of storage media is also known as online storage media. A magnetic disk is used for storing the data for a long time. It is capable of storing an entire database.

It is the responsibility of the computer system to make availability of the data from a disk to the main memory for further accessing. Also, if the system performs any operation over the data, the modified data should be written back to the disk.

The tremendous capability of a magnetic disk is that it does not affect the data due to a system crash or failure, but a disk failure can easily ruin as well as destroy the stored data.

3. Tertiary Storage:
It is the storage type that is external from the computer system. It has the slowest speed. But it is capable of storing a large amount of data. It is also known as Offline storage. Tertiary storage is generally used for data backup. There are following tertiary storage devices available:

a. Optical Storage:
An optical storage can store megabytes or gigabytes of data. A compact Disk (CD) can store 700 megabytes of data with a playtime of around 80 minutes. On the other hand, a Digital Video Disk or a DVD can store 4.7 or 8.5 gigabytes of data on each side of the disk.

b. Tape Storage:
It is the cheapest storage medium than disks. Generally, tapes are used for archiving or backing up the data. It provides slow access to data as it accesses data sequentially from the start.

Thus, tape storage is also known as sequential-access storage. Disk storage is known as direct-access storage as we can directly access the data from any location on disk.

Organization of Records into Blocks:
There are different ways of storing data in the database. Storing data in files is one of them. A user can store the data in files in an organized manner. These files are organized logically as a sequence of records and reside permanently on disks.

Each file is divided into fixed-length storage units known as **Blocks.** These blocks are the units of storage allocation as well as data transfer. Although the default block size in the database is 4 to 8 kilobytes, many databases allow specifying the size at the time of creating the database instance.

Usually, the record size is smaller than the block size. But, for large data items such as images, the size can vary. For accessing the data quickly, it is required that one complete record should reside in one block only.

It should not be partially divided between one or two blocks. In RDBMS, the size of tuples varies in different relations. Thus, we need to structure our files in multiple lengths for implementing the records.

In file organization, there are two possible ways of representing the records.

1. Fixed Length Records
2. Variable Length Records

Fixed Length Records

Fixed-length records means setting a length and storing the records into the file. If the record size exceeds the fixed size, it gets divided into more than one block. Due to the fixed size there occurs following two problems:

a. Partially storing subparts of the record in more than one block requires access to all the blocks containing the subparts to read or write in it.
b. It is difficult to delete a record in such a file organization. It is because if the size of the existing record is smaller than the block size, then another record or a part fills up the block.

However, including a certain number of bytes is the solution to the above problems. It is known as File Header. The allocated file header carries a variety of information about the file, such as the address of the first record. The address of the second record gets stored in the first record and so on. This process is similar to pointers.

The method of insertion and deletion is easy in fixed-length records because the space left or freed by the deleted record is exactly similar to the space required to insert the new records. But this process fails for storing the records of variable lengths.

Variable Length Records:
Variable-length records are the records that vary in size. It requires the creation of multiple blocks of multiple sizes to store them. These variable-length records are kept in the following ways in the database system:

a. Storage of multiple record types in a file.
b. It is kept as Record types that enable repeating fields like multi-sets or arrays.
c. It is kept as Record types that enable variable lengths either for one field or more.

In variable-length records, there exist the following two problems:

a. Defining the way of representing a single record so as to extract the individual attributes easily.
b. Defining the way of storing variable-length records within a block so as to extract that record in a block easily.

Thus, the representation of a variable-length record can be divided into two parts:

a. An initial part of the record with fixed-length attributes such as numeric values, dates, fixed-length character attributes for storing their value.
b. The data for variable-length attributes such as varchar type is represented in the initial part of the record by (offset, length) pair. The offset refers to the place where that record begins, and length refers to the length of the variable-size attribute. Thus, the initial part stores fixed-size information about each attribute, i.e, whether it is the fixed-length or variable-length attribute.

File Organization:

The **File** is a collection of records. Using the primary key, we can access the records. The type and frequency of access can be determined by the type of file organization which was used for a given set of records.

File organization is a logical relationship among various records. This method defines how file records are mapped onto disk blocks. File organization is used to describe the way in which the records are stored in terms of blocks, and the blocks are placed on the storage medium.

The first approach to map the database to the file is to use the several files and store only one fixed length record in any given file. An alternative approach is to structure our files so that we can contain multiple lengths for records.

Files of fixed length records are easier to implement then the files of variable length records.

Objective of File Organization:
1. It contains an optimal selection of records, i.e, records can be selected as fast as possible.
2. To perform insert, delete or update transaction on the records should be quick and easy.
3. The duplicate records cannot be induced as a result of insert, update or delete.
4. For the minimal cost of storage, records should be store efficiently.

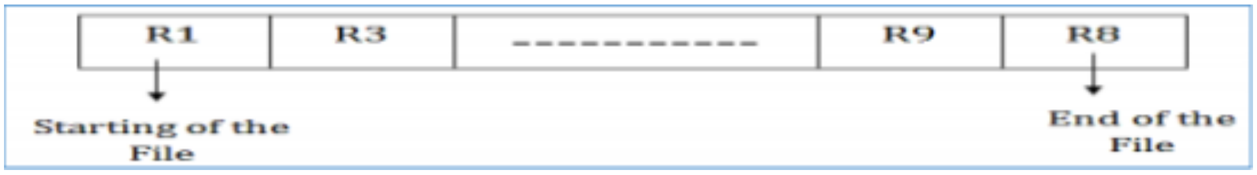
Types of File Organization:
File organization contains various methods. These particular methods have pros and cons on the basis of access or selection. In the file organization, the programmer decides the best-suited file organization method according to his requirement.

1. Sequential File Organization:
This method is the easiest method for file organization. In this method, files are stored sequentially. This method can be implemented in two ways:

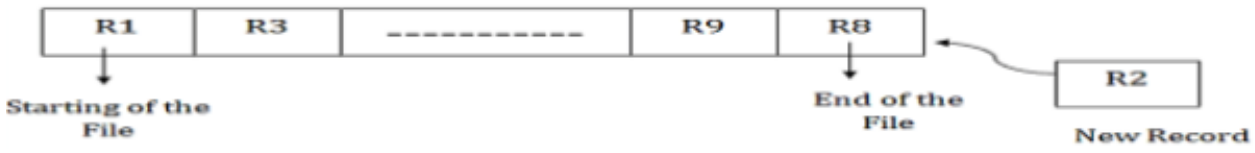
a. Pile File Method:
b. Sorted File Method:

a. Pile File Method:
It is a quiet simple method. In this method, we store the record in a sequence, i.e, one after another. Here, the record will be inserted in the order in which they are inserted into tables.

In case of updating or deleting of any record, the record will be searched in the memory blocks. When it is found, then it will be marked for deleting, and the new record is inserted.

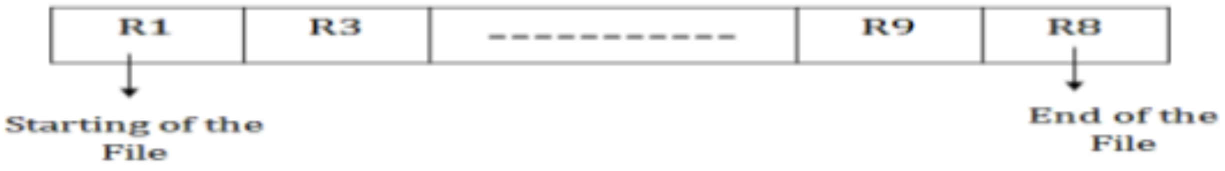


Insertion of the New Record:
Suppose we have four records R1, R3 and so on upto R9 and R8 in a sequence. Hence, records are nothing but a row in the table. Suppose we want to insert a new record R2 in the sequence, then it will be placed at the end of the file. Here, records are nothing but a row in any table.

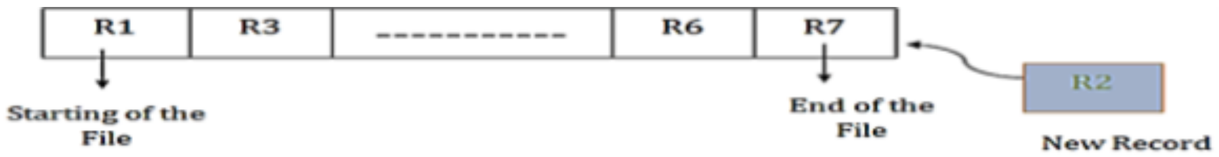


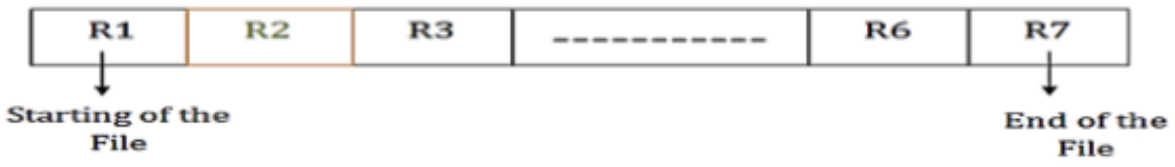
b. Sorted File Method:
In this method, the new record is always inserted at the file's end, and then it will sort the sequence in ascending or descending order. Sorting of records is based on any primary key or any other key.

In the case of modification of any record, it will update the record and then sort the file, and lastly, the updated record is placed in the right place.



Insertion of the New Record:
Suppose there is a preexisting sorted sequence of four records R1, R3 and so on up to R6 and R7. Suppose a new record R2 has to be inserted in the sequence, then it will be inserted at the end of the file, and then it will sort the sequence.





Pros of Sequential File Organization:

a. It contains a fast and efficient method for the huge amount of data.
b. In this method, feels can be easily stored in cheaper storage mechanism like magnetic tapes.
c. It is simple in design. It requires no much effort to store the data.
d. This method is used when most of the records have to be accessed like grade calculation of a student, generating the salary slip, etc.
e. This method is used for report generation or statistical calculations.

Cons of Sequential File Organization:

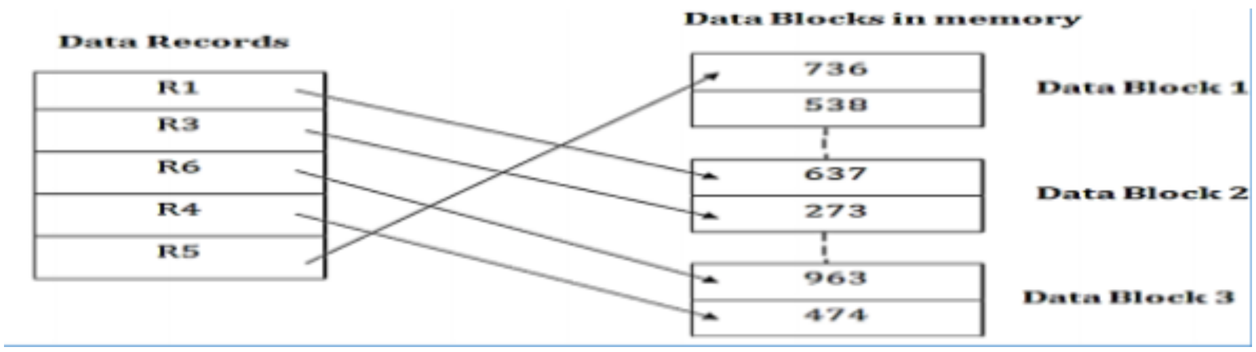
a. It will waste time as we cannot jump on a particular record that s required but we have to move sequentially which takes our time.
b. Sorted file method takes more time and space for sorting the records.

2. Heap File Organization:

It is the simplest and most basic type of organization. It works with data blocks. In heap file organization, the records are inserted at the file's end. When the records are inserted, it doesn't require the sorting and ordering of records.

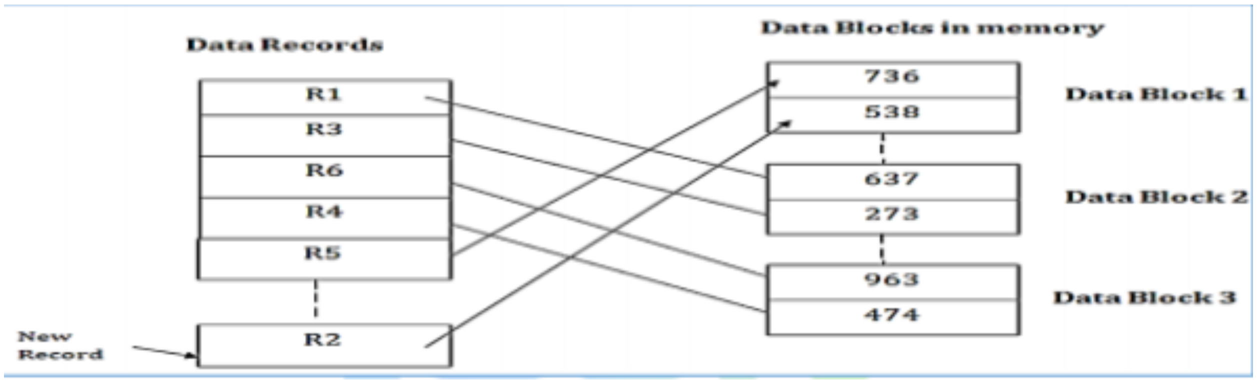
When the data block is full, the new record is stored in some other block. This new data block need not to be the very next data block, but it can select any data block in the memory to store new records. The heap files is also known as an unordered file.

In the file, every record has a unique id, and every page in a file is of the same size. It is the DBMS responsibility to store and manage the new records.



Insertion of a New Record:

Suppose we have five records R1, R3, R6, R4 and R5 in a heap and suppose we want to insert a new record R2 in a heap. If the data block 3 is full then it will be inserted in any of the database selected by the DBMS, let's say data block 1.



If we want to search, update or delete the data in heap file organization, then we need to traverse the data from starting of the file till we get the requested record.

If the database is very large then searching, updating or deleting of record will be time-consuming because there is no sorting or ordering of records. In the heap file organization, we need to check all the data until we get the requested record.

Pros of Heap File Organization:

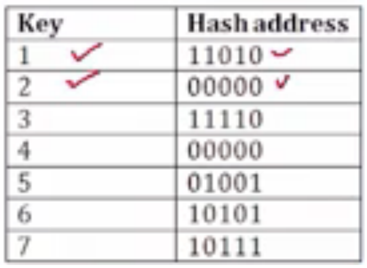
a. It is a very good method of file organization for bulk insertion. If there is a large number of data which needs to load into the database at a time, then this method is best suited.
b. In case of a small database, fetching and retrieving of records is faster than the sequential record.

Cons of Heap File Organization:

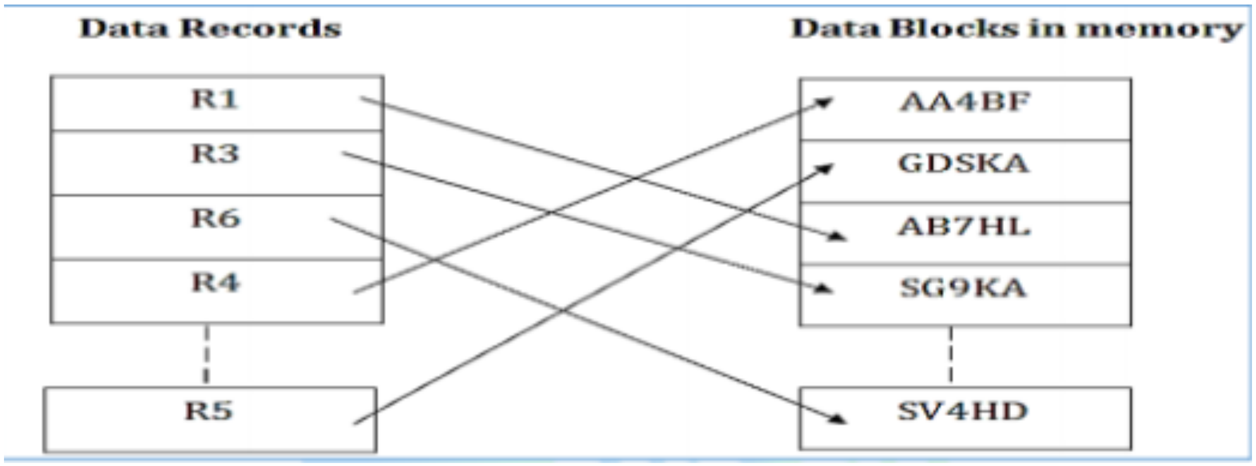
a. This method is inefficient for the large database because it takes time to search or modify the record.
b. This method is inefficient for large databases.

3. Hash File Organization

Hash File organization uses the computation of hash function on some fields of the records. The hash function's output determines the location of disk block where the records are to be placed.

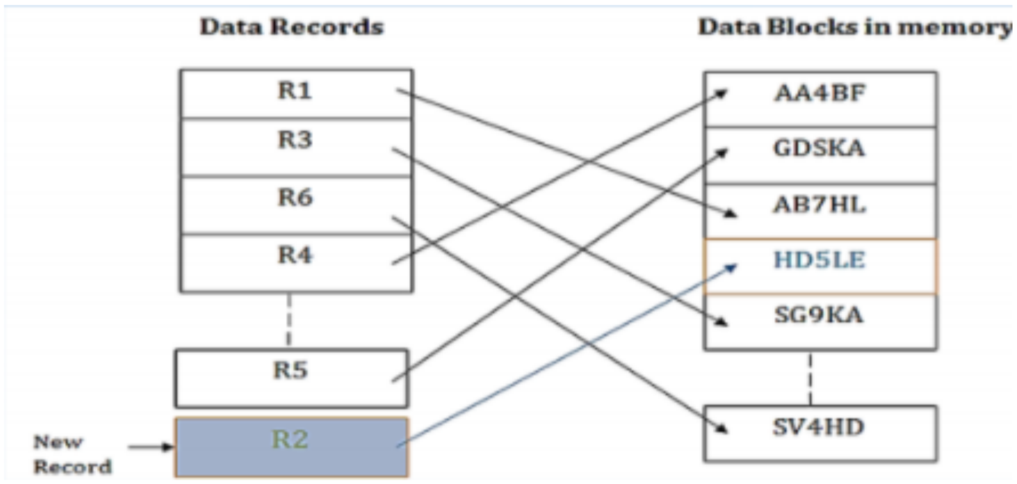


Key	Hash address
1 ✓	11010 ✓
2 ✓	00000 ✓
3	11110
4	00000
5	01001
6	10101
7	10111



When a record has to be received using the hash key columns, then the address is generated, and the whole record is retrieved using that address. In the same way, when a new record has to be inserted, then the address is generated using the hash key and record is directly inserted. The same process is applied in the case of delete and update.

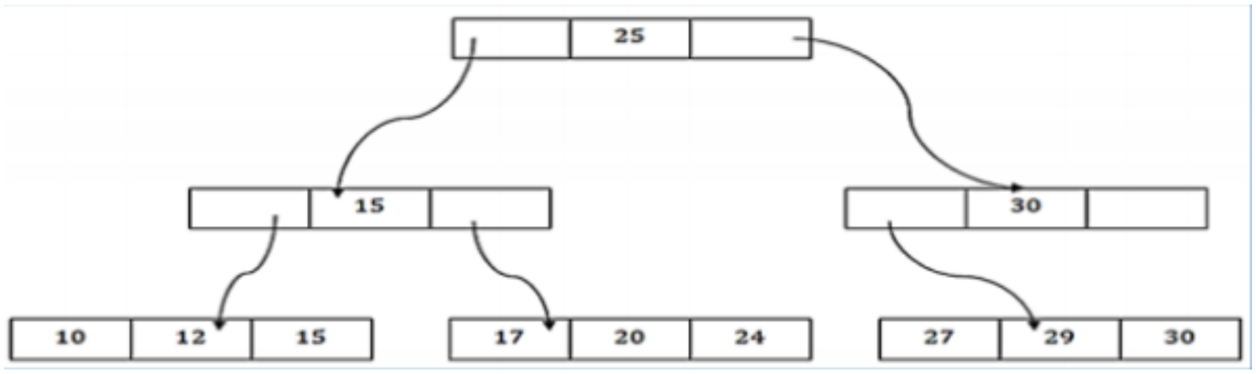
In this method, there is no effort for searching and sorting the entire file. In this method, each record will be stored randomly in the memory.



4. B+ File Organization

B+ tree file organization is the advanced method of an indexed sequential access method. It uses a tree-like structure to store records in File. It uses the same concept of key-index where the primary key is used to sort the records. For each primary key, the value of the index is generated and mapped with the record.

The B+ tree is similar to a binary search tree (BST), but it can have more than two children. In this method, all the records are stored only at the leaf node. Intermediate nodes act as a pointer to the leaf nodes. They do not contain any records.



The Above B+ Tree Shows That:

a. There is one root node of the tree, i.e 25
b. There is an intermediary layer with nodes. They do not store the actual record. They have only pointers to the leaf node.

c. The nodes to the left of the root node contain the prior value of the root and nodes to the right contain next value of the root, i.e, 15 and 30 respectively.
d. There is only one leaf node which has only values, i.e., 10, 12, 17, 20, 24, 27 and 29.
e. Searching for any record is easier as all the leaf nodes are balanced.
f. In this method, searching any record can be traversed through the single path and accessed easily.

Pros of B+ Tree File Organization:

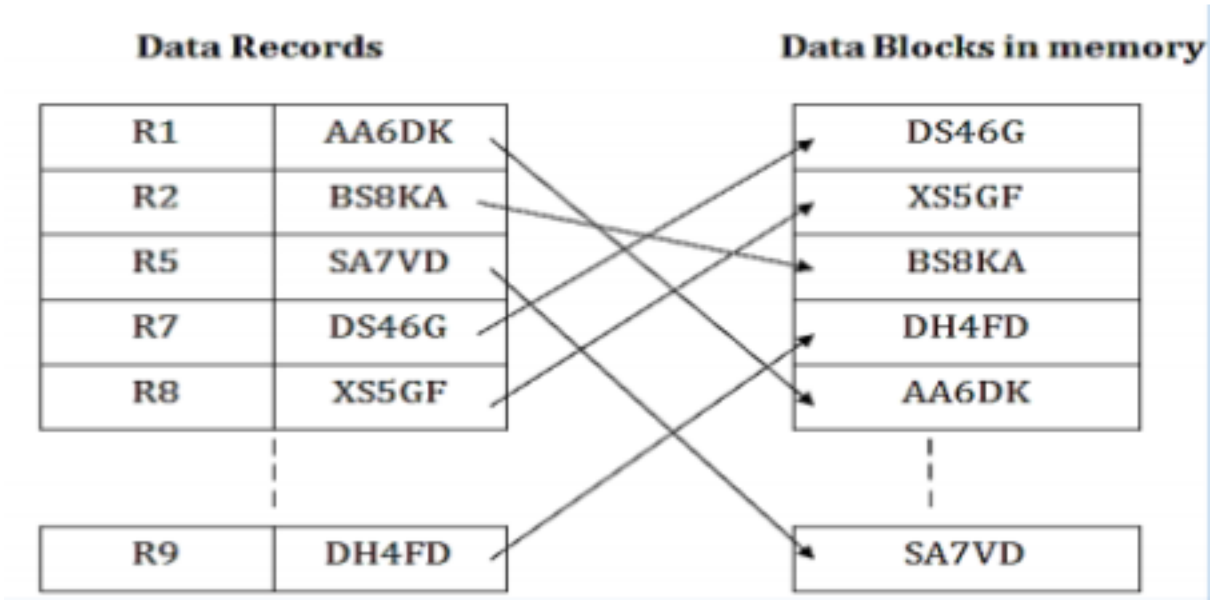
a. In this method, searching become very easy as all the records are stored only in the leaf nodes and sorted the sequential linked list.
b. Traversing through the tree structure is easier and faster.
c. The size of the B+ tree has no restrictions, so the number of records can increase or decrease and the B+ tree structure can also grow or shrink
d. It is a balanced tree structure, and any insert/update/delete does not affect the performance of tree.

Cons of B+ Tree File Organization:

a. This method is inefficient for the static method.

5. Indexed Sequential Access Method (ISAM)

ISAM method is an advanced sequential file organization. In this method, records are stored in the file using the primary key. An index value is generated for each primary key and mapped with the record. This index contains the address of the record in the file.



If any record has to retrieved based on its index value, then the address of the data block is fetched and the record is retrieved from the memory.

Pros of ISAM:

a. In this method, each record has the address of its data block, searching a record in a huge database is quick and easy
b. This method supports range retrieval and partial retrieval of records. Since the index is based on the primary key values, we can retrieve the data for the given range of value. In the same way, the partial value can also be easily searched, i.e, the student name starting with 'JA' can be easily searched.

Cons of ISAM:

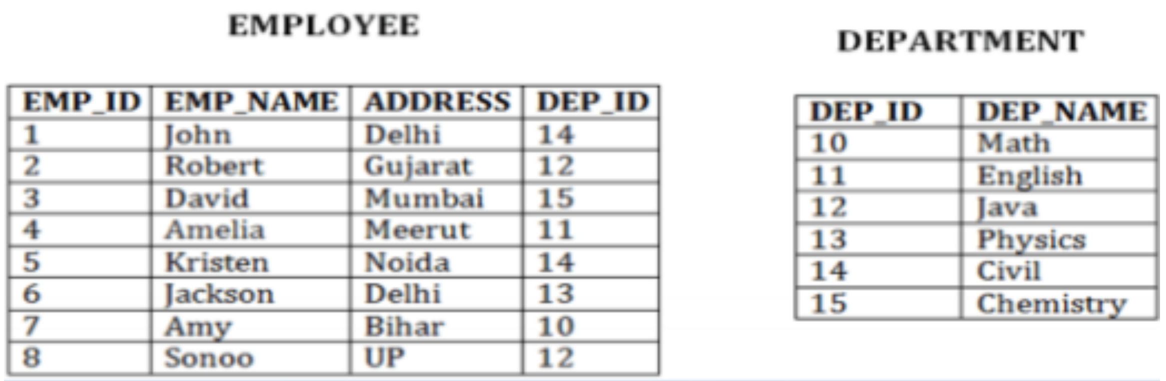
a. This method requires extra space in the disk to store the index value.
b. When the new records are inserted, then these files have to be reconstructed to maintain the sequence.
c. When the record is deleted, then the space used by it needs to be released. Otherwise, the performance of the database will slow down.

6. Cluster File Organization:

When the two or more records are stored in the same file, it is known as clusters. These files will have two or more tables in the same data block, and key attributes which are used to map these tables together are stored only once.

This method reduces the cost of searching for various records in different files. The cluster file organization is used when there is a frequent need for joining the tables with the same condition

These joins will give only a few records from both tables. In the given example, we are retrieving the record for only particular departments. This method can't be used to retrieve the record for the entire department.

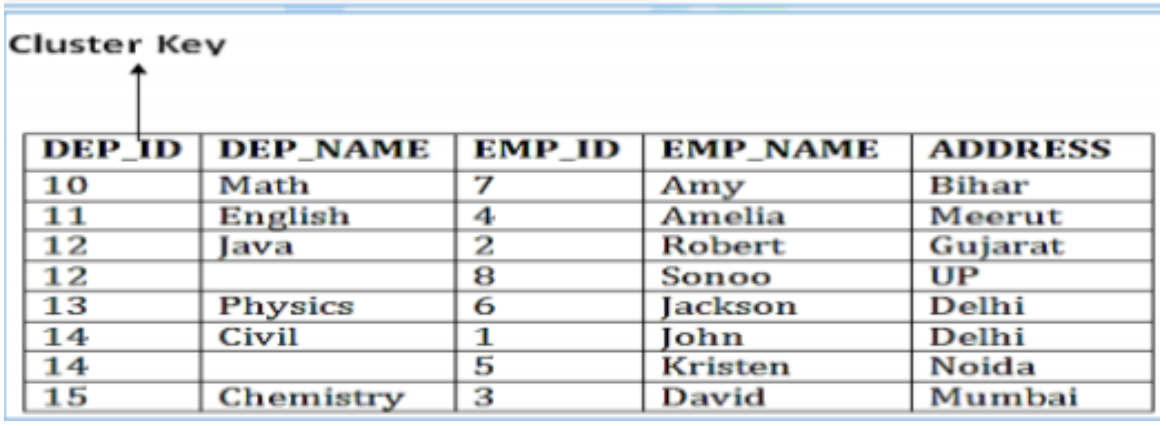


EMPLOYEE

EMP_ID	EMP_NAME	ADDRESS	DEP_ID
1	John	Delhi	14
2	Robert	Gujarat	12
3	David	Mumbai	15
4	Amelia	Meerut	11
5	Kristen	Noida	14
6	Jackson	Delhi	13
7	Amy	Bihar	10
8	Sonoo	UP	12

DEPARTMENT

DEP_ID	DEP_NAME
10	Math
11	English
12	Java
13	Physics
14	Civil
15	Chemistry



Cluster Key

DEP_ID	DEP_NAME	EMP_ID	EMP_NAME	ADDRESS
10	Math	7	Amy	Bihar
11	English	4	Amelia	Meerut
12	Java	2	Robert	Gujarat
12		8	Sonoo	UP
13	Physics	6	Jackson	Delhi
14	Civil	1	John	Delhi
14		5	Kristen	Noida
15	Chemistry	3	David	Mumbai

In this method, we can directly insert, update or delete any record. Data is sorted based on the key with which searching is done. Cluster key is a type of key with which joining of the table is performed.

Types of Cluster File Organization:

1. Indexed Clusters:

In indexed cluster, records are grouped based on ht cluster key and stored together. The above EMPLOYEE and DEPARTMENT relationship is an example of an indexed cluster. Here, all the records are grouped based on the cluster key- DEP_ID and all the records are grouped.

2. Hash Clusters:

It is similar to the indexed cluster. In hash cluster, instead of storing the records based on the cluster key, we generate the value of the hash key for the cluster key and store the records with the same hash key value.

Pros of Cluster File Organization:

a. The cluster file organization is used when there is a frequent request for joining the tables with same joining condition.

b. It provides the efficient result when there is a 1:M mapping between the tables.

Cons of Cluster File Organization:

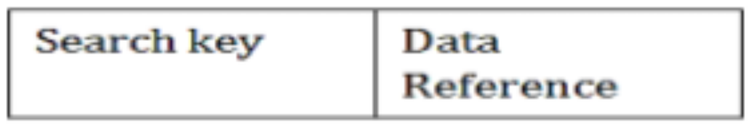
a. This method has the low performance for the very large database.
b. If there is any change in joining condition, then this method cannot use. If we change the condition of joining then traversing the file takes a lot of time.
c. This method is not suitable for a table with a 1:1 condition

Indexing in DBMS:

Indexing is using to optimize the performance of a database by minimizing the number of disk accesses required when a query is processed. The index is a type of data structure. It is used to locate and access the data in a database table quickly.

Index Structure:

Indexes can be creating using some database columns.



Search key	Data Reference

Fig: Structure of Index

The first column of the database is the search key that contains a copy of the primary key or candidate key of the table. The values of the primary key are stored in sorted order so that the corresponding data can be accessed easily.

The second column of the database is the data reference. It contains a set of pointers holding the address of the disk block where the value of the particular key can be found.

Indexing Methods:

1. Ordered Indices

2. **Primary Index**
    1. **Dense Index**
    2. **Sparse Index**
3. **Clustering Index**
4. **Secondary Index**

1. Ordered Indices:
The indices are usually sorted to make searching faster. The indices which are sorted are known as ordered indices.

Example:
Suppose we have an employee table with thousands of record and each of which is 10 bytes long. If their IDs start with 1,2,3…and so on and we have to search student with ID-543

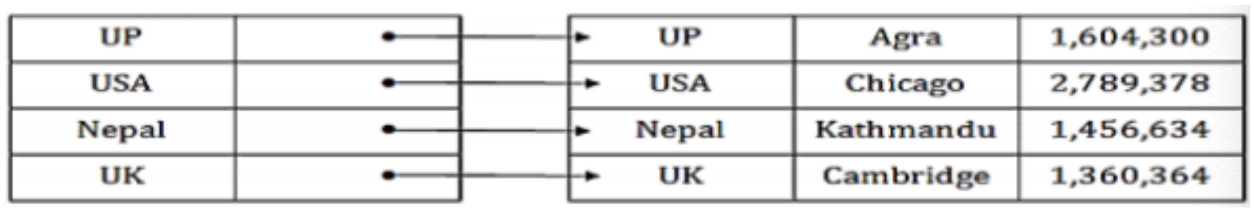
1. In the case of a database with no index, we have to search the disk block from starting till it reaches 543. The DBMS will read the record after reading 543*10=5430 bytes.
2. In the case of an index, we will search using indexes and the DBMS will read the record after reading 542*2=1084 bytes which are very less compared to the previous case.

2. Primary Index:
If the index is created on the basis of the primary key of the table, then it is known as primary indexing. These primary keys are unique to each record and contain 1:1 relation between the records.

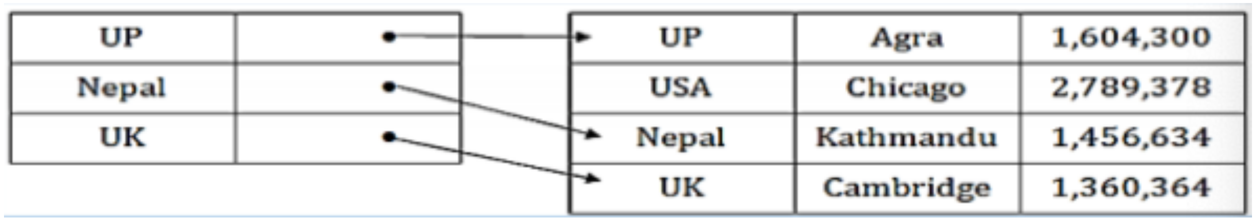
As primary keys are stored in sorted order, the performance of the searching operation is quite efficient. The primary index can be classified into two types: **Dense index** and **Sparse index.**

2.1 Dense Index
The dense index contains an index record for every search key value in the data file. It makes searching faster. In this, the number of records in the index table is same as the number of records in the main table.

It needs more space to store index record itself. The index records have the search key and a pointer to the actual record on the disk.



2.2 Sparse Index
In the data file, index record appears only for a few items. Each item points to a block. In this, instead of pointing to each record in the main table, the index points to the records in the main table in a gap.

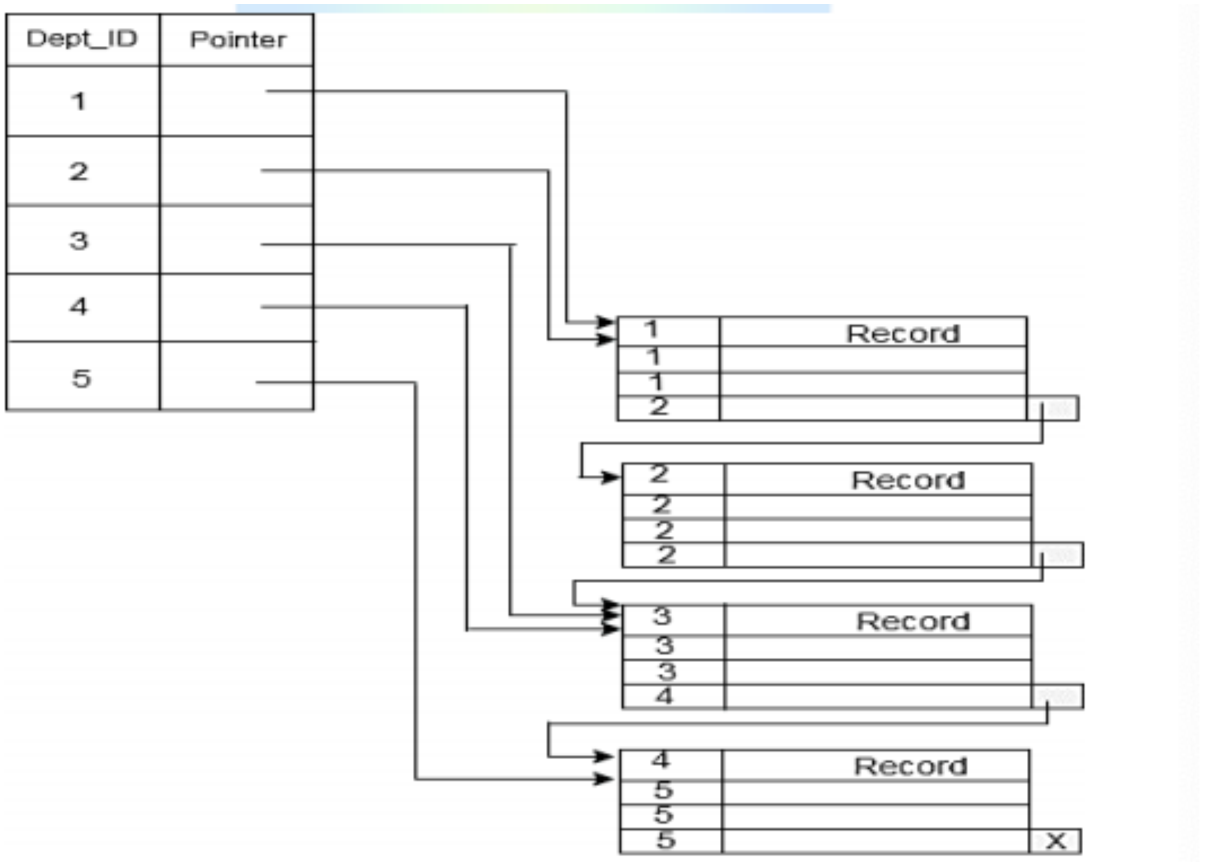


3. Clustering Index
A clustering index can be defined as an ordered data file. Sometimes the index is created on non-primary key columns which may not be unique for each record.

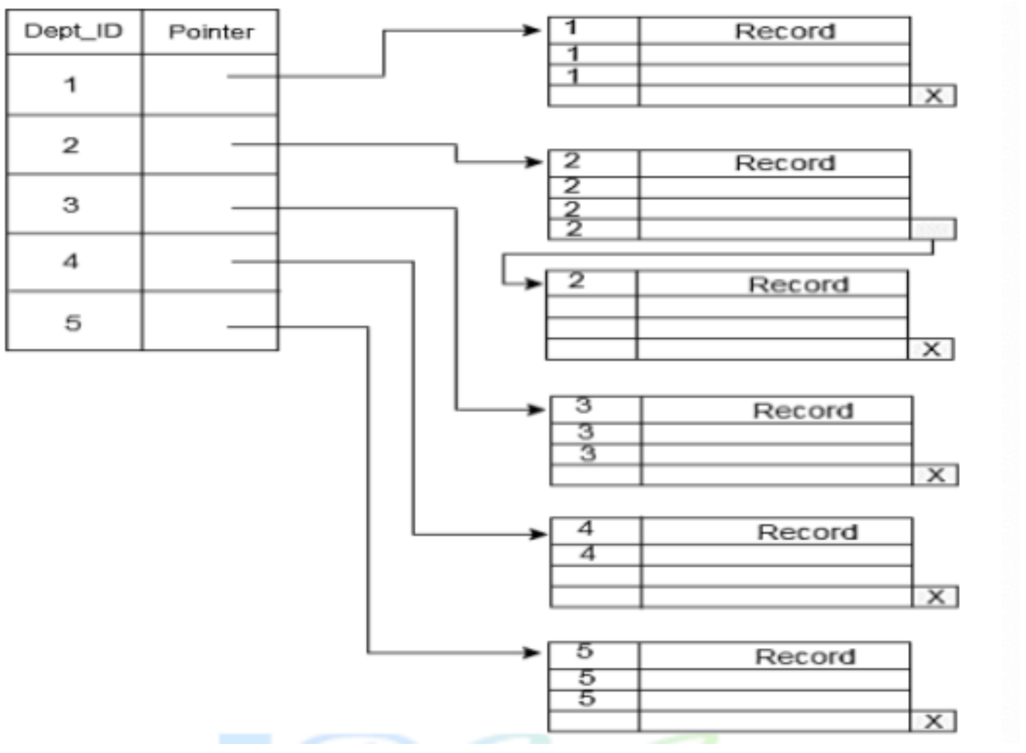
In this case, to identify the record faster, we will group two or more columns to get the unique value and create index out of them. This method is called a clustering index.

The records which have similar characteristics are grouped, and indexes are created for these group.

Example: Suppose a company contains several employees in each department. Suppose we use a clustering index, where all employees which belong to the same Dept_ID are considered within a single cluster, and index pointers point to the cluster as a whole. Here Dept_Id is a non-unique key.



The previous schema is little confusing because one disk block is shared by records which belong to the different cluster. If we use separate disk block for separate clusters, then it is called better technique.



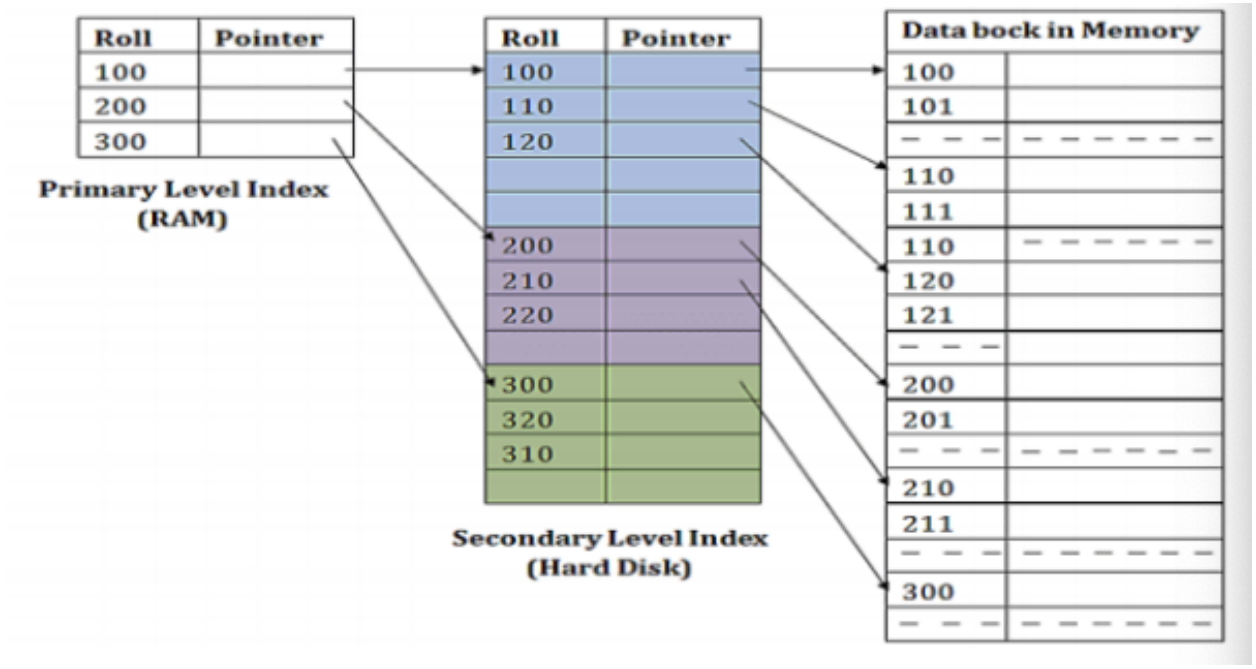
4. Secondary Index:

In the sparse indexing, as the size of the table grows, the size of mapping also grows. These mappings are usually kept in the primary memory so that address fetch should be faster.

Then the secondary memory searches the actual data based on the address got from mapping. If the mapping size grows then fetching the address itself becomes slower. In this case, the sparse index will not be efficient. To overcome this problem, secondary indexing is introduced.

In secondary indexing, to reduce the size of mapping, another level of indexing is introduced. In this method, the huge range of the columns is selected initially so that the mapping size of the first level becomes small.

Then each range is further divided into smaller ranges. The mapping of the first level is stored in the primary memory, so that address fetch is faster. The mapping of the second level and actual data are stored in the secondary memory (hard disk).



For Example:

1. If we 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.

2. 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.
3. This is how a search is performed in this method. Inserting, updating or deleting is also done in the same manner.

Types of Indexes:

1. Function-Based Indexes:

- A function-based index computes the values of expression which are present in one or more column and stored in the table.
- The expression can be an arithmetic expression or SQL function
- A function-based index cannot contain null value.

Example: CREATE INDEX Sample_idx on table_(a+c*(b+d));

2. Bitmap Indexes:

- Bitmap index is used to work with well for low-cardinal (refers to columns few unique values) columns in table. **For example:** Boolean data which has only two values true or false.
- Bitmap indexes are very useful in data ware house applications for joining the large fact tables.

3. Domain Indexes:

Domain index is used to create index type schema object and an application specific index. It is used for indexing data in application specific domain.

4. Clusters:

Clustering in DBMS is design for high availability of data. Clustering is applied on tables which are repeatedly used by the user. **For example:** When there are many employees in the department, we can create index of non-unique key, such that Dept-id. With this, all employees belonging to the same department are considered to be within a same cluster. Clustering can improve the performance of the system.

5. Indexed Sequential Access Method (ISAM):

ISAM was developed by IBM for mainframe computers but the term is used in several concepts.
In DBMS, ISAM is used to access data in sequentially (sequence in which data is entered) or randomly (with an index).