Simran Kumari
70362019031
A018

PAGE No. 1
DATE / /

25/03/2020

# ASSIGNMENT - 2

Q1.) Explain indexed sequential access scheme for file organization.

A-1.] What is File? File can be defined as a collection of records related to each other. The file size is limited by the size of memory and storage medium.

- THERE ARE 2 IMPORTANT FEATURES OF FILE :-

1) File Activity :- specifies percent of actual records which proceed in a single run.
2) File volatility :- addresses the properties of record changes. It helps to increase the efficiency of disk design than tape.

- FILE ORGANIZATION :-

File organization ensures that records are available for processing. It is used to determine an efficient file organization for each base relation.

- TYPES OF FILE ORGANIZATION :-

There are three types of organizing the file :-

1.) Sequential access file organization
2.) Direct access file organization
3.) Indexed sequential access file organization.

## 3.) INDEXED SEQUENTIAL ACCESS FILE ORGANIZATION

- Indexed sequential access file combines both sequential file (storing and sorting in contiguous block within files on tape or disc) and direct access file organization (all records are stored in direct access storage device (DASD), such as hard disk. The records are randomly placed throughout the file).

- In indexed sequential access file, records are stored randomly on a direct access device such as magnetic disk by a primary key.
- This file have multiple keys. These keys can be alphanumeric in which the records are ordered is called primary key
- The data can be access either sequentially or randomly using the index. The index is stored in a file and read into memory when the file is opened.
- Indexed sequential file organization stores data for fast retrieval. The records in an indexed sequential file are of fixed length and every record is uniquely identified by a key field. We maintain a table known as the index table which stores the record number and the address of all the records. That is for every file, we have an index table. This type of file organization is called as indexed sequential file organization because physically the records may be stored anywhere, but the index table stores the address of those records.
- The ith entry in the index table points to the ith record of the file. Initially, when the file is created, each entry in the index table contains NULL. When the ith record of the file is written, free space is obtained from the free space manager and its address is stored in the ith location of the index table.
- Now, if one has to read the 4th record, then there is no need to access the first three records. Address of the 4th record can be obtained from the index table and the record can be straightaway read from the ~~index~~ specified address (742, in our example). Conceptually, the index sequential file organization can be visualized as shown in the figure.

Simran Kumari
A018



## - INDEXED SEQUENTIAL FILE ORGANISATION -

| Record Number | Address of the Record |
|---|---|
| 1 | 765 → Record |
| 2 | 27 → Record |
| 3 | 876 → Record |
| 4 | 742 → Record |
| 5 | NULL |
| 6 | NULL |
| 7 | NULL |
| 8 | NULL |
| 9 | NULL |

- As, An indexed sequential file uses the concept of both sequential as well as relative files. While the index table is read sequentially to find the address of the desired record, a direct access is made to the address of the specified record in order to access it randomly.
- Indexed sequential files perform well in situations where sequential access as well as random access is made to the data. Indexed sequential files can be stored only on devices that support random access, for example, magnetic disks. For example, take an example of a college where the details of students are stored in an indexed sequential file. This file can be accessed in two ways:
  - → Sequentially - to print the aggregate marks obtained by each student in a particular course or,
  - → Randomly - to modify the name of a particular student.

- **FEATURES OF SE INDEXED SEQUENTIAL FILE ORGANIZATION**
  - → Provides fast data retrieval
  - → Records are of fixed length

→ Index table stores the address of the records in the file.
→ The ith entry in the index table points to the ith record of the file.
→ While the index table is read sequentially to find the address of the desired record, a direct access is made to the address of the specified record in order to access it randomly.
→ Indexed sequential files perform well is situations where sequential access as well as random access is made to the data.

• ADVANTAGES:-
→ The key improvement is that the indices are small and can be searched quickly, allowing the database to access only the records it needs.
→ supports applications that require both batch and interactive processing.
→ Records can be accessed sequentially as well as randomly
→ Updates the records in the same file.

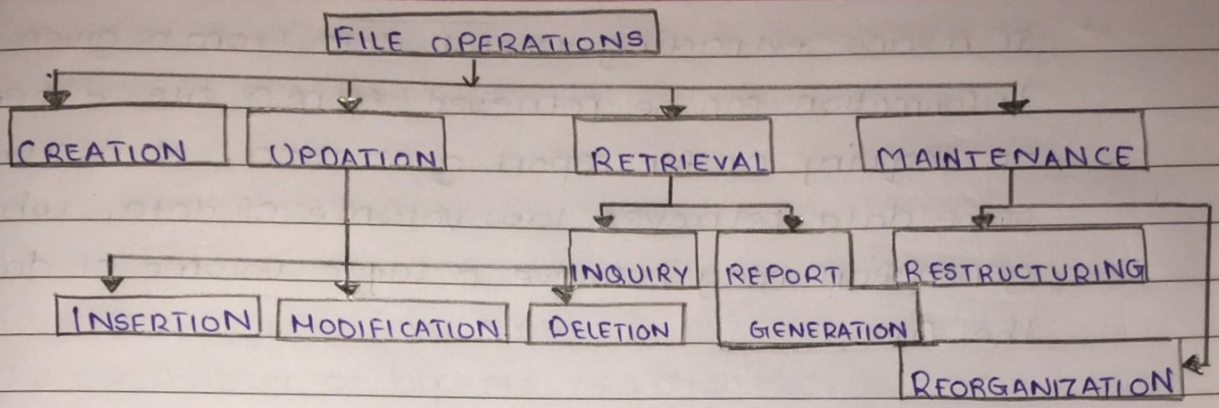
• DISADVATAGES:-
→ Indexed sequential files can be stored only on disks.
→ Needs extra space and overhead to store indices
→ Handling these files is more complicated than handling sequential files.
→ Supports only fixed length records.

Q-2.) What are the basic file operations.

A-2.) The basic operations that can be performed on a file are given in the below mentioned figure.

Simran Kumari
A018






- **CREATING A FILE :-**

A file is created by specifying its name and mode. Then the file is opened for writing records that are read from an input device. Once all the records have been written into the file, the file is closed. The file is now available for future read / write operations by any program that has been designed to use it in some way or the other.

- **UPDATING A FILE :-**

Updating a file means changing the contents of the file to reflect a current picture of reality. A file can be updated in the following ways:-

→ Inserting a new record in the file. For eg, if a student joins the course, we need to add his record to the STUDENT file.

→ Deleting an existing record. For eg, if a student quits a course in the middle of the session, his record has to be deleted from the STUDENT file.

→ Modifying an existing record. For eg, if the name of a student was spelt incorrectly, then correcting the name will be a modification of the existing record.

- **RETRIEVING FROM A FILE :-**

It means extracting useful data from a given file. Information can be retrieved from a file either for an inquiry or for report generation. A inquiry for some data retrieves low volume of data, while report generation may retrieve a large volume of data from the file.

- **MAINTAINING A FILE:-**

It involves restructuring or re-organizing the file to improve the performance of the programs that access this file. Restructuring a file keeps the file organization unchanged and changes only the structural aspects of the file (For eg, changing the field width or adding/deleting fields.) On the other hand, file reorganization may involve changing the entire organisation of the file. ~~We~~ ~~to~~.

Q-3.) Explain Hashed Indices:-

A-3.) We have studied that hashing is used to compute the address of a record by using a hash function on the search key value. If at any point of time, the hashed values map to the same address, then collision occurs and schemes to resolve these collisions are applied to generate a new address.
Choosing a good hash function is critical to the success of this technique. By a good hash function, we mean two things. First, a good hash function, irrespective of the number of search keys, gives an average-case lookup that is a small constant. Second, the function distributes records uniformly and randomly among the buckets, where a bucket is defined as a unit of one or more records

(typically a disk block). Correspondingly, the worst hash function is one that maps all the keys to the same bucket.

However, the drawback of using hashed indices includes:

1.) Though the number of buckets is fixed, the number of files may grow with time.

2.) If the number of buckets is too large, storage space is wasted.

3.) If the number of buckets is too small, there may be too many collisions.

It is recommended to set the number of buckets to twice the number of the search key values in the file This gives a good space-performance trade off. A hashed file organization uses hashed indices. Hashing is used to calculate the address of disk block where the desired record is stored. If K is the set of all search key values and B is the set of all bucket addresses, then a hash function H maps K to B. We can perform the following operations in a hashed file organization

→ Insertion :-

To insert a record that has ki as its search ~~file~~ value, use the hash function h(ki) to compute the address of the bucket for that record. If the bucket is free, store the record else use chaining to store the record.

→ Search :-

To search a record having the key value ki, use h(ki) to compute the address of the bucket where the record is stored. The bucket may contain one or several records, so check for every record in the bucket (by comparing ki with the key of every record) to finally retrieve the desired record with the given key value.

Simran Kumari
A018

PAGE No. 8
DATE / /

→ **Deletion :-**

To delete a record with key value ki, use h(Ki) to compute the address of the bucket where the record is stored. The bucket may contain one or several records so check for every record in the bucket (by comparing ki with the key of every record). Then delete the record as we delete a node from a linear linked list.

Note that in a hashed file organization, the secondary indices need to be organized using hashing.