UNIT 12 FILE STRUCTURES

Structure		Page Nos.
12.0	Introduction	31
12.1	Objectives	31
12.2	Terminology	32
12.3	File Organisation	32
12.4	Sequential Files	33
	12.4.1 Structure	
	12.4.2 Operations	
	12.4.3 Disadvantages	
	12.4.4 Areas of use	
12.5	Direct File Organisation	35
12.6	Indexed Sequential File Organisation	35
12.7	Summary	37
12.8	Solutions/Answers	37
12.9	Further readings	37

12.0 INTRODUCTION

The structures of files change from operating system to operating system. In this unit, we shall discuss the fundamentals of file structures along with the generic file organisations.

A file may be defined as a collection of records. Though, a text file doesn't conform to this definition at the first glance, it is also a collection of records and records are words in the file.

Consider a file consisting of information about students. We may name such a file as *Student* file. The typical records of such file are shown in *Figure 12.1*.

Enum	Name	Address	State	Country	Programme
012786345	John	D-51, Nebsarai, Maidan Garhi	Delhi	India	BCA
98387123	Suresh	E-345, Banjara Hills	Hyderabad	India	MCA

Figure 12.1: Typical records of a *Student* file

A file should always be stored in such a way that the basic operations on it can be performed easily. In other words, queries should be able to be executed with out much hassle. We focus, in this unit, on the ways of storing the files on external storage devices. Selection of a particular way of storing the file on the device depends on factors such as retrieval records, the way the queries can be put on the file, the total number of keys in each record etc.

12.1 OBJECTIVES

After going through this unit, you should be able to

- learn the terminology of file structures;
- learn the underlying concepts of Sequential files, and
- know the Indexed sequential file organisation.

12.2 TERMINOLOGY

The following are the definitions of some important terms:

1) **Field:** It is an elementary data item characterised by its size, length and type. For example,

   Name : a character type of size 10
   Age : a numeric type

2) **Record:** It is a collection of related fields that can be treated as a unit from an application point of view.

   For example:

A university could use a student record with the fields, namely, University enrolment no. and names of courses.

3) **File:** Data is organised for storage in files. A file is a collection of similar, related records. It has an identifying name.

For example, "STUDENT" could be a file consisting of student records for all the students in a university.

4) **Index:** An index file corresponds to a data file. It's records contain a key field and a pointer to that record of the data file which has the same value of the key field.

The data stored in files is accessed by software which can be divided into the following two categories:

i) **User Programs:** These are usually written by a programmer to manipulate retrieved data in the manner required by the application.

ii) **File Operations:** These deal with the physical movement of data, in and out of files. User programs effectively use file operations through appropriate programming language syntax The File Management System manages the independent files and acts as the software interface between the user programs and the file operations.

   File operations can be categorised as

- CREATION of the file
- INSERTION of records into the file
- UPDATION of previously inserted records
- RETRIEVAL of previously inserted records
- DELETION of records
- DELETION of the file.

12.3 FILE ORGANISATION

File organisation may be defined as a method of storing records in file. Also, the subsequent implications on the way these records can be accessed. The following are the factors involved in selecting a particular file organisation:

- Ease of retrieval
- Convenience of updates
- Economy of storage
- Reliability
- Security
- Integrity.

Different file organisations accord different weightages to the above factors. The choice must be made depending upon the individual needs of the particular application in question.

We now introduce in brief, the various commonly encountered file organisations.

- **Sequential Files**

  Data records are stored in some specific sequence e.g., order of arrival value of key field etc. Records of a sequential file cannot be accessed at random i.e., to access the n^{th} record, one must traverse the preceding (n–1) records. Sequential files will be dealt with at length in the next section.

- **Relative Files**

  Each data record has a fixed place in a relative file. Each record must have associated with it in integer key value that will help identify this slot. This key, therefore, will be used for insertion and retrieval of the record. Random as well as sequential access is possible. Relative files can exist only on random access devices like disks.

- **Direct Files**

  These are similar to relative files, except that the key value need not be an integer. The user can specify keys which make sense to his application.

- **Indexed Sequential Files**

  An index is added to the sequential file to provide random access. An overflow area needs to be maintained to permit insertion in sequence.

- **Indexed Flies**

  In this file organisation, no sequence is imposed on the storage of records in the data file, therefore, no overflow area is needed. The index, however, is maintained in strict sequence. Multiple indexes are allowed on a file to improve access.

12.4 SEQUENTIAL FILES

In this section, we shall discuss about Sequential file organisation. Sequential files have data records stored in a specific sequence. A sequentially organised file may be stored on either a serial-access or a direct-access storage medium.

12.4.1 Structure

To provide the ‘sequence’ required, a ‘key’ must be defined for the data records. Usually a field whose values can uniquely identify data records is selected as the key. If a single field cannot fulfil this criterion, then a combination of fields can serve as the key.

12.4.2 Operations

1. **Insertion:** Records must be inserted at the place dictated by the sequence of the keys. As is obvious, direct insertions into the main data file would lead to

frequent rebuilding of the file. This problem could be mitigated by reserving' overflow areas' in the file for insertions. But, this leads to wastage of space.

The common method is to use transaction logging. This works as follows:

- It collects records for insertion in a transaction file in their order of their arrival.
- When population of the transactions file has ceased, sort the transaction file in the order of the key of the primary data file
- Merge the two files on the basis of the key to get a new copy of the primary sequential file.

Such insertions are usually done in a batch mode when the activity/program which populates the transaction file have ceased. The structure of the transaction files records will be identical to that of the primary file.

2. **Deletion:** Deletion is the reverse process of insertion. The space occupied by the record should be freed for use. Usually deletion (like-insertion) is not done immediately. The concerned record is written to a transaction file. At the time of merging, the corresponding data record will be dropped from the primary data file.
3. **Updation:** Updation is a combination of insertion and deletions. The record with the new values is inserted and the earlier version deleted. This is also done using transaction files.
4. **Retrieval:** User programs will often retrieve data for viewing prior to making decisions. Therefore, it is vital that this data reflects the latest state of the data, if the merging activity has not yet taken place.

Retrieval is usually done for a particular value of the key field. Before return in to the user, the data record should be merged with the transaction record (if any) for that key value.

The other two operations 'creation' and 'deletion' of files are achieved by simple programming language statements.

12.4.3 Disadvantages

Following are some of the disadvantages of sequential file organisation:

- Updates are not easily accommodated.
- By definition, random access is not possible.
- All records must be structurally identical. If a new field has to be added, then every record must be rewritten to provide space for the new field.

12.4.4 Areas of Use

Sequential files are most frequently used in commercial batch oriented data processing where there is the concept of a master file to which details are added periodically. Example is payroll applications.

Check Your Progress

1) Describe the record structure to be used for the lending section of a library.

……

……

……

.....

12.5 DIRECT FILE ORGANISATION

It offers an effective way to organise data when there is a need to access individual records directly.

To access a record directly (or random access) a relationship is used to translate the key value into a physical address. This is called the mapping function R.
R(key value) = Address

Direct files are stored on DASD (Direct Access Storage Device).

A calculation is performed on the key value to get an address. This address calculation technique is often termed as hashing. The calculation applied is called a hash function.

Here, we discus a very commonly used hash function called Division - Remainder.

Division-Remainder Hashing

According to this method, key value is divided by an appropriate number, generally a prime number, and the division of remainder is used as the address for the record.

The choice of appropriate divisor may not be so simple. If it is known that the file is to contain n records, then we must, assuming that only one record can be stored at a given address, have divisor n.

Also we may have a very large key space as compared to the address space. Key space refers to all the possible key values. The address space possibly may not match actually for the key values in the file. Hence, calculated address may not be unique. It is called Collision, i.e.,

R(K1) = R(K2), where K1 = K2.

Two unequal keys have been calculated to have the same address. The keys are called synonyms.

There are various approaches to handle the problem of collisions. One of these is to hash to buckets. A bucket is a space that can accommodate multiple records. The student is advised to read some text on bucket Addressing and related topics.

12.6 INDEXED SEQUENTIAL FILE ORGANISATION

When there is need to access records sequentially by some key value and also to access records directly by the same key value, the collection of records may be organised in an effective manner called Indexed Sequential Organisation.

You must be familiar with search process for a word in a language dictionary. The data in the dictionary is stored in sequential manner. However an index is provided in terms of thumb tabs. To search for a word we do not search sequentially. We access the index. That is, locate an approximate location for the word and then proceed to find the word sequentially.

To implement the concept of indexed sequential file organisations, we consider an approach in which the index part and data part reside on a separate file. The index file has a tree structure and data file has a sequential structure. Since the data file is sequenced, it is not necessary for the index to have an entry for each record. Consider the sequential file with a two-level index.

Level 1 of the index holds an entry for each three-record section of the main file. The Level 2 indexes Level 1 in the same way.

When the new records are inserted in the data file, the sequence of records need to be preserved and also the index is accordingly updated.

Two approaches used to implement indexes are static indexes and dynamic indexes.

As the main data file changes due to insertions and deletions, the contents of the static index may change, but the structure does not change. In case of dynamic indexing approach, insertions and deletions in the main data file may lead to changes in the index structure. Recall the change in height of B-Tree as records are inserted and deleted.

Both dynamic and static indexing techniques are useful depending on the type of application.

A directory is a component of file. Consider a file which doesn't have any keys for its records. When a query is executed on such a file, the time consumed to execute the query is more when compared to another file which is having keys, because, there may be arising necessity where in the file has to be sorted on the field(s) on which the query is based. So, for each query, the file has to be sorted on the field on which the query is based which is cumbersome. In the case of files which have keys, different versions of the files which result due to sorting on the keys are stored in the directory of that file. Such files are called index files and the number of index files vary from file to file. For example, consider the file of *Figure 12.1*. If we designate Enrolment Number (Enum) and Name as keys, then we may have two index files based on the each key. Of course, we can have more than two index files, to deal with queries which use both the keys. Different software store index files in a different manner so that the operations on the records can be performed as soon as possible after the query is submitted.

One of the prominent indexing techniques is Cylinder-Surface indexing.

Since, there exists a primary key for each of the files, there will be an index file based on the primary key. Cylinder-Surface Indexing is useful for such index file. In this type of indexing, the records of the file are stored one after another in such a way that the primary keys of the records are in increasing order. The index file will have two fields. They are cylinder index and corresponding surface indexes. There will be multiple cylinders and there are multiple surfaces corresponding to each cylinder. Suppose that the file needs m cylinders, then cylinder index will have m entries. Each cylinder will be having one entry which corresponds to the largest key value in that cylinder. Assume that the disk has n surfaces which can be used. Then, each surface index has n entries. The k-th entry in surface index for cylinder l^{th} cylinder if the value of the largest key on the l^{th} track of the k^{th} surface. Hence, m.n indicates the total number of surface index entries.

Suppose that the need arises to search for a record whose key value is B. Then, the first step is to load the cylinder index of the file into memory. Usually, each cylinder index occupies only one track as the number of cylinders are only few. The cylinder which holds the desired record is found by searching the cylinder index. Usually, the search takes O(log m) time. After the search of cylinder index, the corresponding cylinder is determined. Based on the cylinder, the corresponding surface index is retrieved to look the record for which the search has started. Whenever a search is initiated for the surface index, usually sequential search is used. Of course, it depends on the number of surfaces. But, usually, the number of surfaces are less. After finding the cylinder to be accessed and after finding the surface to be accessed, the corresponding track is loaded into memory and that track is searched for the needed record.

12.7 SUMMARY

This unit dealt with the methods of physically storing data in the files. The terms fields, records and files were defined. The organisation types were introduced.

The various file organisation techniques were discussed. Sequential File Organisation finds use in application areas where batch processing is more common. Sequential files are simple to use and can be stored on inexpensive media. They are suitable for applications that require direct access to only particular records of the collection. They do not provide adequate support for interactive applications.

In Direct file organisation, there exists a predictable relationship between the key used and to identify a particular record on secondary storage. A direct file must be stored on a direct access device. Direct files are used extensively in application areas where interactive processing is used.

An Indexed Sequential file supports both sequential access by key value and direct access to a particular record, given its key value. It is implemented by building an index on top of a sequential data file that resides on a direct access storage device.

12.8 SOLUTIONS/ANSWERS

Check Your Progress

1) The following record structure could take care of the general requirements of a lending library.

   Member No., Member Name, Book Classification, i.e., Book Name, Author, Issue Date, Due Date.

12.9 FURTHER READINGS

Reference Books

1. *Fundamentals of Data Structures in C++* by E. Horowitz, Sahani and D. Mehta, Galgotia Publications.

2. *Data Structures using C and C ++* by Yedidyah Hangsam, Moshe J. Augenstein and Aaron M. Tanenbaum, PHI Publications.

3. *Fundamentals of Data Structures in C* by R.B. Patel, PHI Publications.

Reference Websites

http:// www.cs.umbc.edu

http://www.fredosaurus.com