File system Management

What is File ?

- File in which information is recorded on secondary storage.

What is file system ?

- A file system is **a method an operating system uses to store, organize, and manage files and directories on a storage device**.

Some common types of file systems include:

- **FAT (File Allocation Table):** An older file system used by older versions of Windows and other operating systems.
- **NTFS (New Technology File System):** A modern file system used by Windows. It supports features such as file and folder permissions, compression, and encryption.
- **ext (Extended File System):** A file system commonly used on Linux and Unix-based operating systems.
- **HFS (Hierarchical File System):** A file system used by macOS.
- **APFS (Apple File System):** A new file system introduced by Apple for their Macs and iOS devices.

File attributes

- **Name**

  This denotes the symbolic name of the file. The file name is the only attribute that is readable by humans easily.
- **Identifier**

  This denotes the file name for the system. It is usually a number and uniquely identifies a file in the file system.
- **Type**

  If there are different types of files in the system, then the type attribute denotes the type of file.
- **Location**

  This points to the device that a particular file is stored on and also the location of the file on the device.
- **Size**

  This attribute defines the size of the file in bytes,words or blocks. It may also specify the maximum allowed file size.
- **Protection**

  The protection attribute contains protection information for the file such as who can read or write on the file.

File operations

- **Creating a file**

  To create a file, there should be space in the file system. Then the entry for the new file must be made in the directory. This entry should contain information about the file such as its name, its location etc.

- **Reading a file**

  To read from a file, the system call should specify the name and location of the file. There should be a read pointer at the location where the read should take place. After the read process is done, the read pointer should be updated.

- **Writing a file**

  To write into a file, the system call should specify the name of the file and the contents that need to be written. There should be a write pointer at the location where the write should take place. After the write process is done, the write pointer should be updated.

- **Deleting a file**

  The file should be found in the directory to delete it. After that all the file space is deleted so it can be reused by other files.

- **Repositioning in a file**

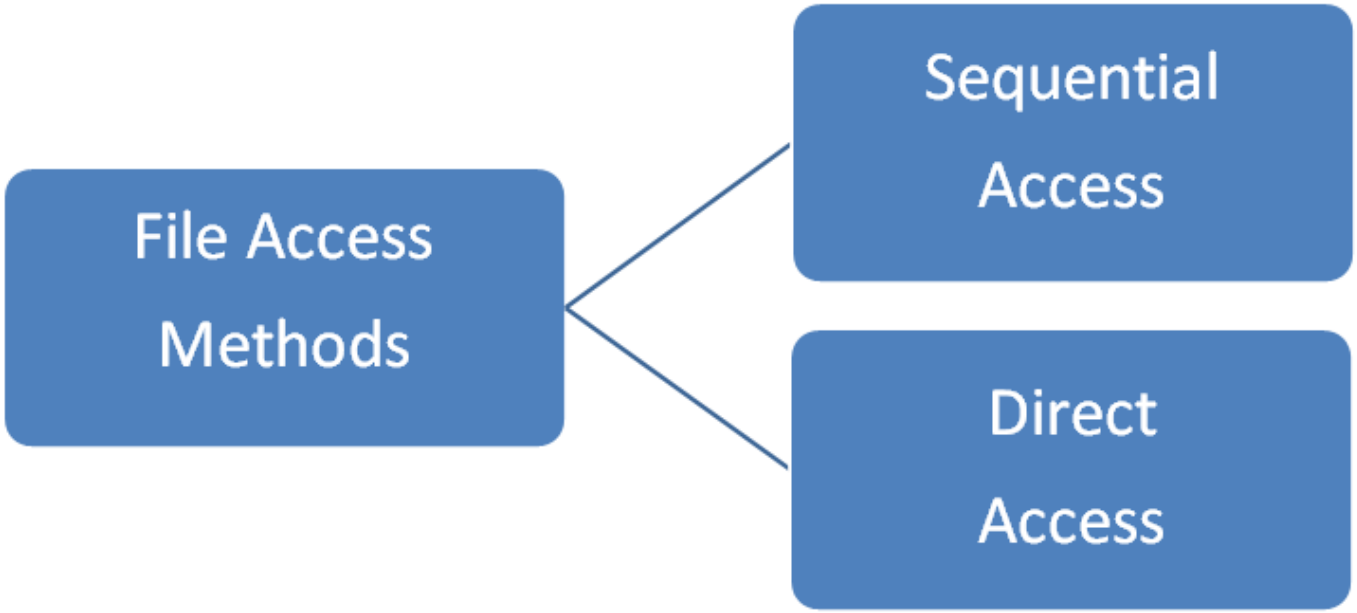
  This is also known as file seek. To reposition a file, the current file value is set to the appropriate entry. This does not require any actual I/O operations.

- **Truncating a file**

  This deletes the data from the file without destroying all its attributes. Only the file length is reset to zero and the file contents are erased. The rest of the attributes remain the same.

Access Methods

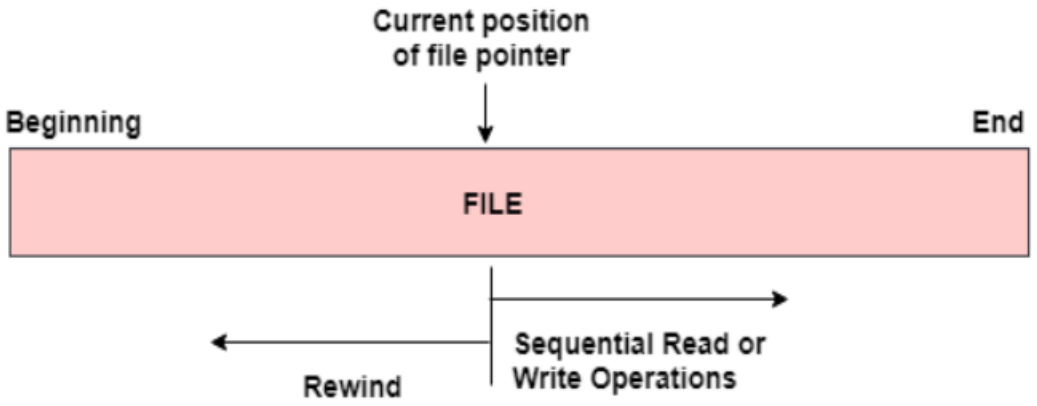
- The information in a file can be accessed in various ways.
- The most common among them are using sequential access or direct access.



Sequential Access

- The information in a file is processed in order using sequential access.
- The files records are accessed on after another.
- Most of the file systems such as editors, compilers etc. use sequential access.
- It is based on the tape model of a file and so can be used with sequential access devices as well as random access devices.

As seen in the image, the read and write operations in the file can only be done in a sequential manner. However, the file can be reset to the beginning or rewinded as required.

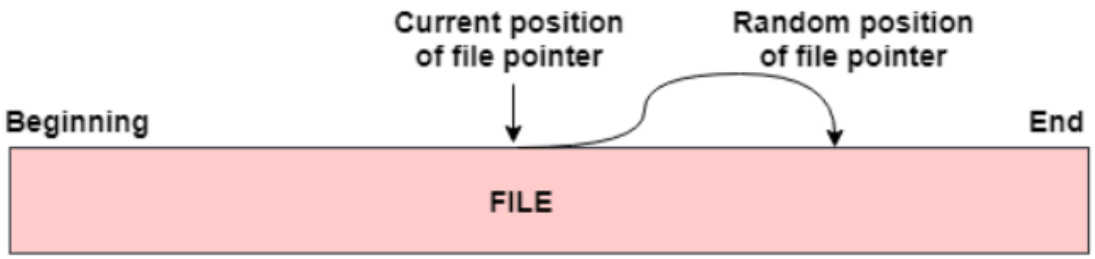


SEQUENTIAL ACCESS IN FILES

Direct Access

- In direct access or relative access files can be accessed in random for read and write operations.
- The direct access model is based on the disk model of a file, since it allows random accesses.
- In this method, the file is divided into numbered blocks. Any of these arbitrary blocks can be read or written.
- For example, we may read block 8, then write into block 10 and then read block 15. **Direct access system is quite useful and mostly databases are of this type.**

As seen in the above image, the file pointer can be positioned randomly as required for read and write operations. This can be done without any particular order in positioning.



DIRECT ACCESS IN FILES

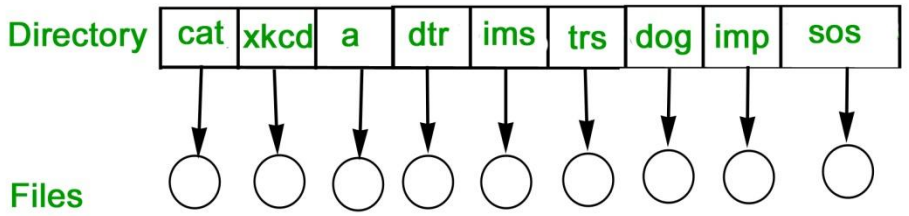
FILE DIRECTORIES:

Collection of files is a file directory.

- The directory contains information about the files, including attributes, location and ownership.
- Much of this information, especially that is concerned with storage, is managed by the operating system.
- The directory is itself a file, accessible by various file management routines.
- **Operation performed on directory are:**
- Search for a file
- Create a file
- Delete a file
- List a directory
- Rename a file
- Traverse the file system

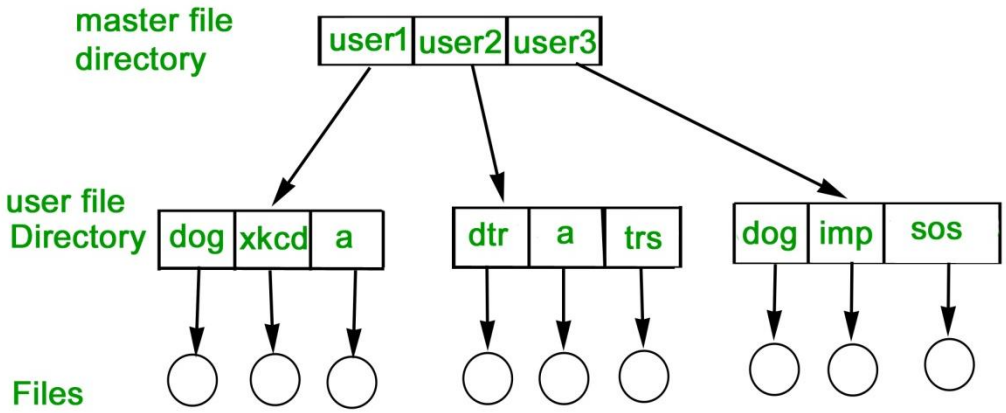
SINGLE-LEVEL DIRECTORY

- In this a single directory is maintained for all the users.
- **Naming problem:** Users cannot have same name for two files.
- **Grouping problem:** Users cannot group files according to their need.



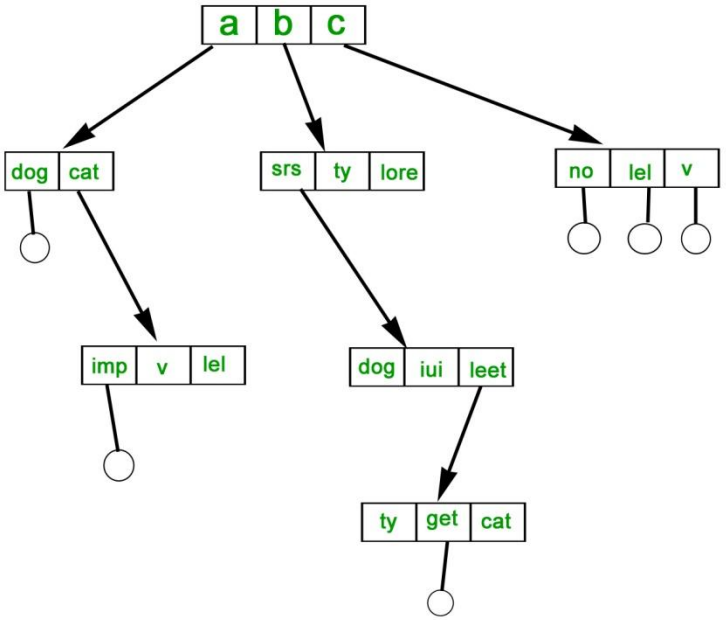
TWO-LEVEL DIRECTORY

- In this separate directories for each user is maintained.
  Path name:Due to two levels there is a path name for every file to locate that file.
- Now,we can have same file name for different user.
- Searching is efficient in this method.



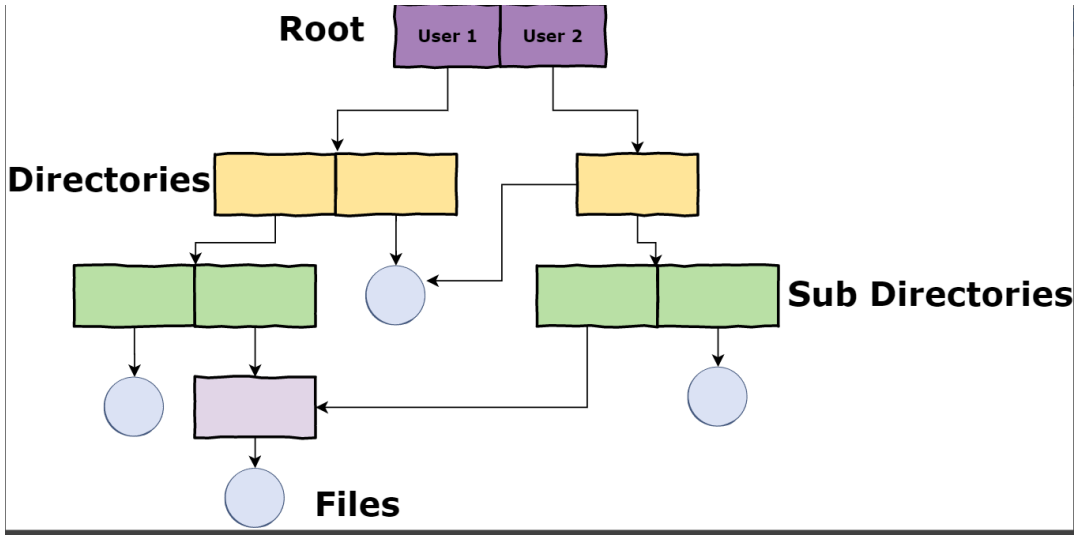
TREE-STRUCTURED DIRECTORY :

- Directory is maintained in the form of a tree. Searching is efficient and also there is grouping capability. We have absolute or relative path name for a file.



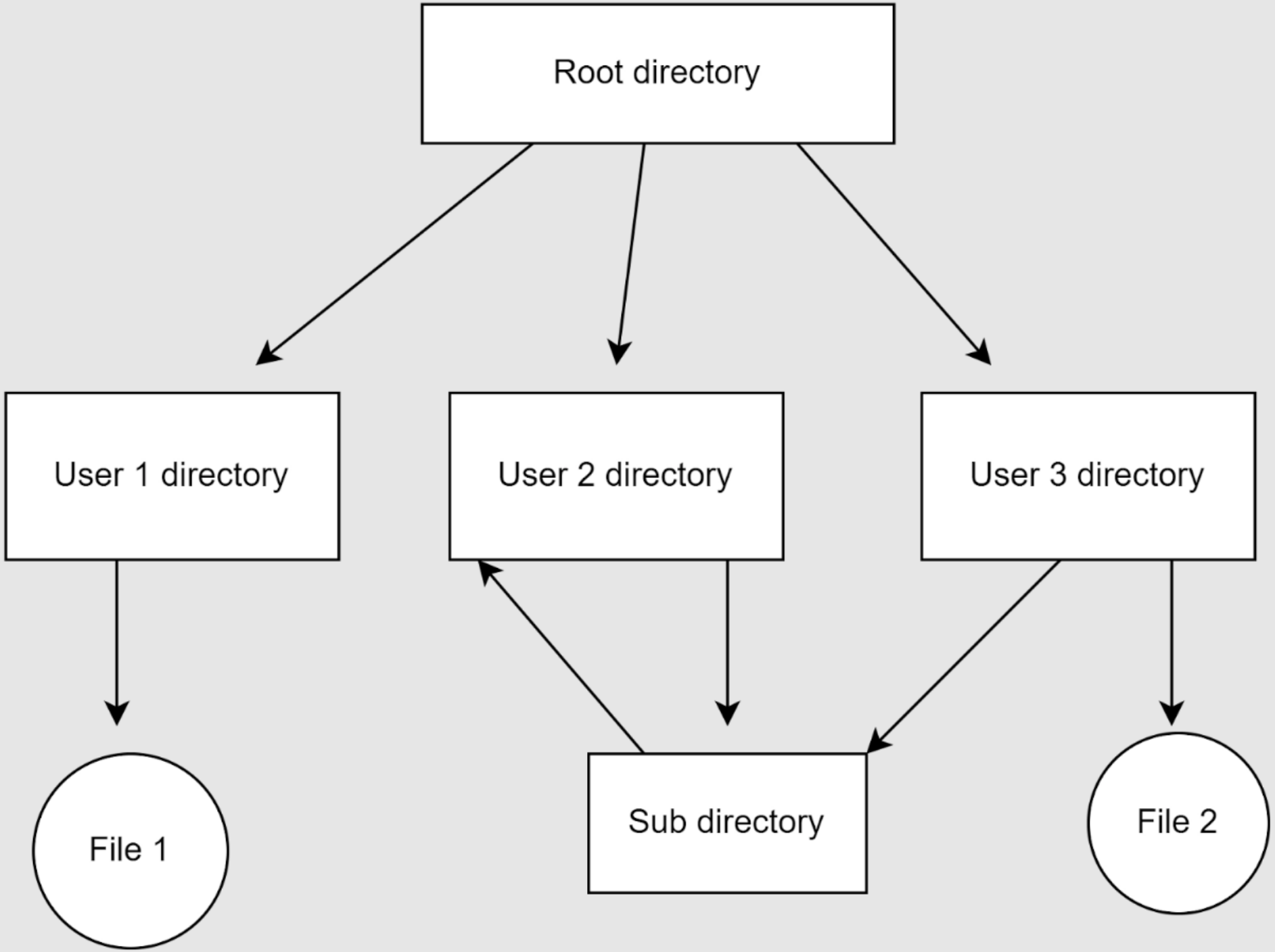
Acyclic directories

- **Acyclic directories** are a generalization of the **Tree directory** structure.
- Acyclic directories allow files to have multiple parent directories; this means that multiple users can access the same file from different paths.
- However, it is important to note that the *shared* file is *not* a copy of the same file in two different directories.

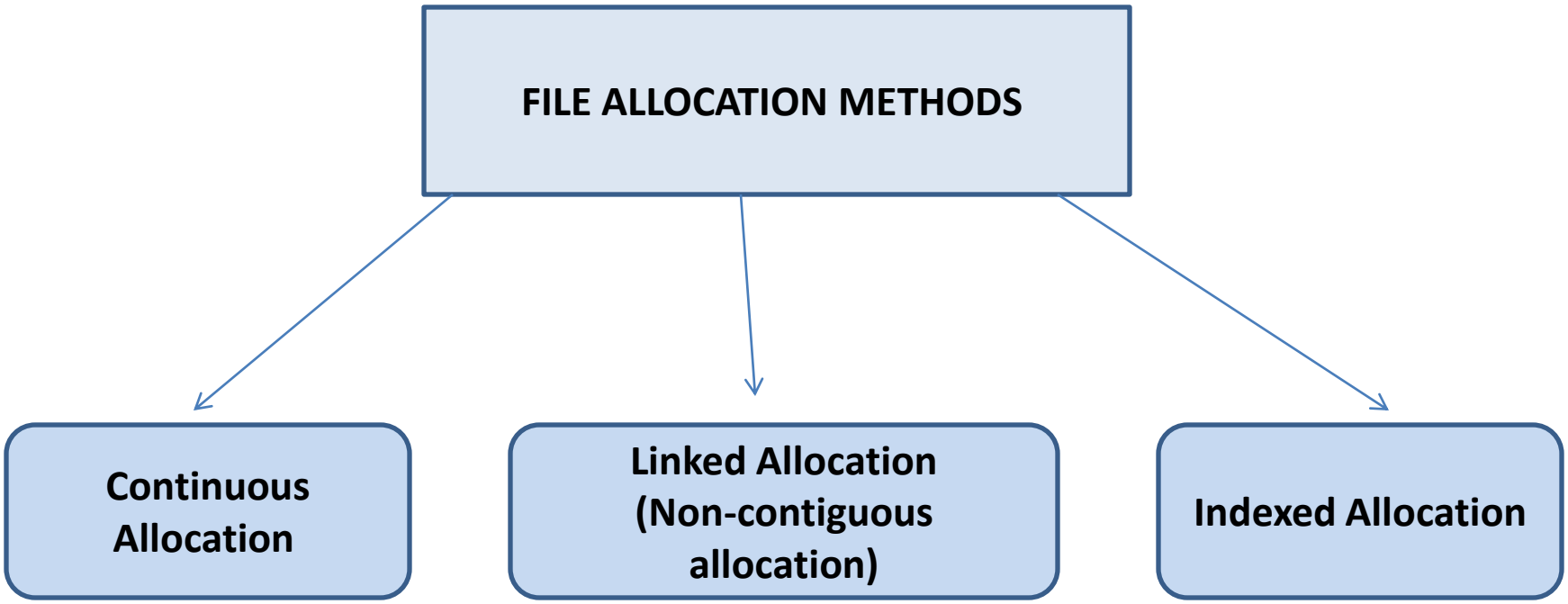


General Graph Directories

- Acyclic graphs are usually right, but it's expensive to check for cycles
- Instead, many systems allow any kind of graph
  - A is a subdirectory of B, and B is a subdirectory of A
- Now, applications that traverse the tree have to be careful
- Also, reference counting for deletion may not work

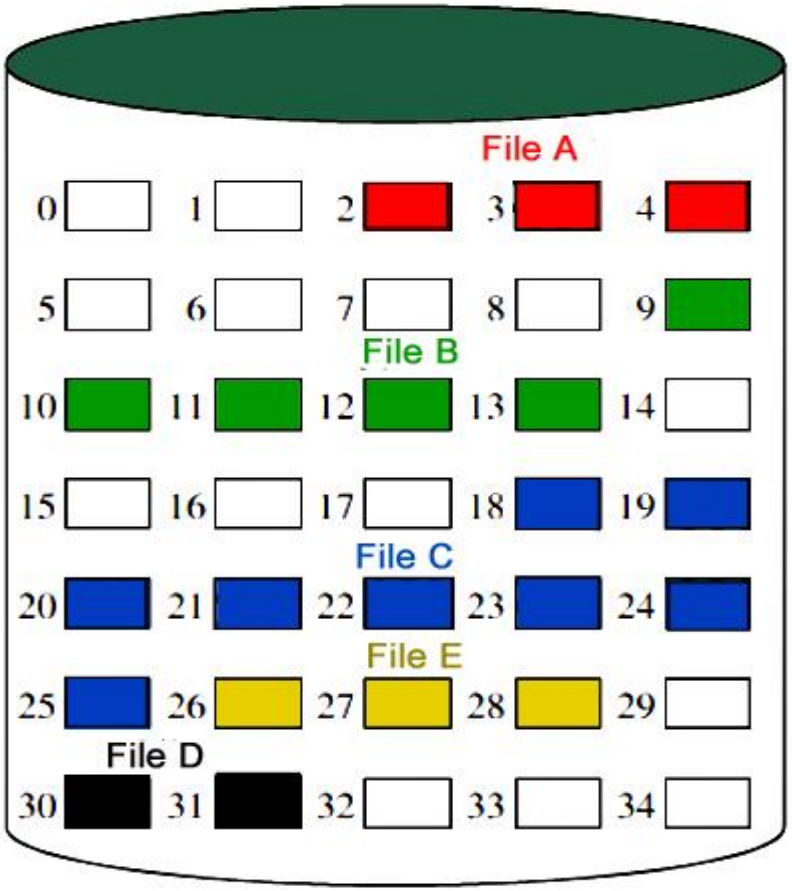

Root directory
User 1 directory
User 2 directory
User 3 directory
File 1
Sub directory
File 2

FILE ALLOCATION METHODS



Continuous Allocation

- A single continuous set of blocks is allocated to a file at the time of file creation.
- Thus, this is a pre-allocation strategy, using variable size portions.
- The file allocation table needs just a single entry for each file, showing the starting block and the length of the file.
- This method is best from the point of view of the individual sequential file.
- Multiple blocks can be read in at a time to improve I/O performance for sequential processing. It is also easy to retrieve a single block.
- For example, if a file starts at block b, and the ith block of the file is wanted, its location on secondary storage is simply b+i-1.



File allocation table

File name	Start block	Length
File A	2	3
File B	9	5
File C	18	8
File D	30	2
File E	26	3

Disadvantage –

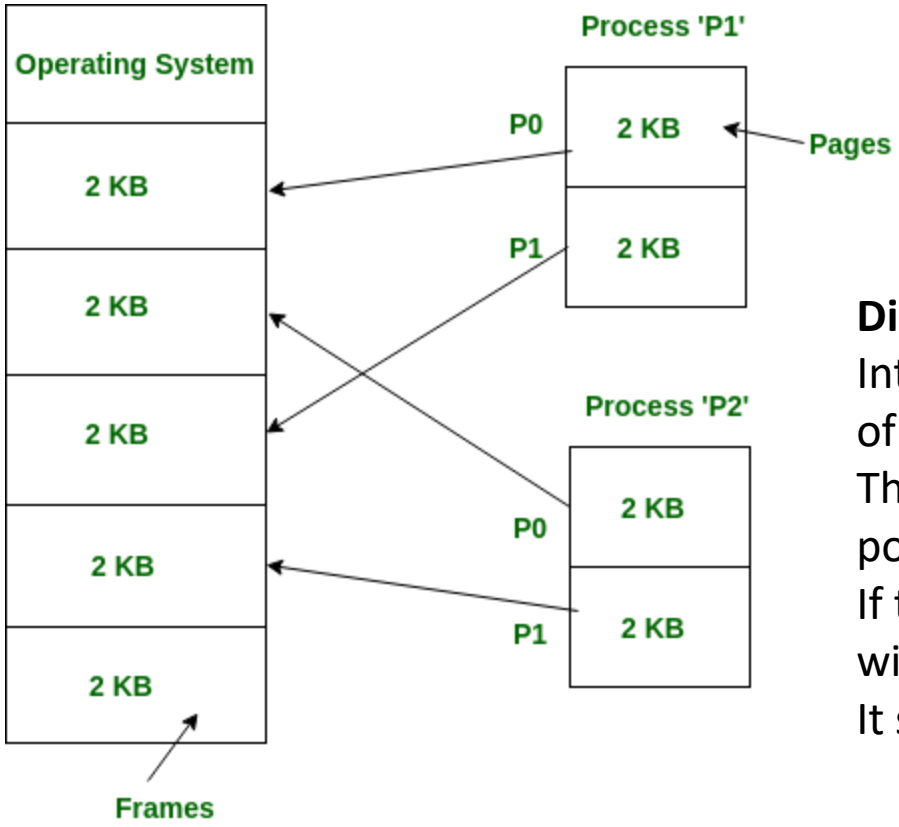
External fragmentation will occur, making it difficult to find contiguous blocks of space of sufficient length.

Compaction algorithm will be necessary to free up additional space on disk.

Also, with pre-allocation, it is necessary to declare the size of the file at the time of creation.

Linked Allocation(Non-contiguous allocation)

- Allocation is on an individual block basis.
- Each block contains a pointer to the next block in the chain.
- Again the file table needs just a single entry for each file, showing the starting block and the length of the file.
- Although pre-allocation is possible, it is more common simply to allocate blocks as needed.
- Any free block can be added to the chain. The blocks need not be continuous.
- Increase in file size is always possible if free disk block is available.
- There is no external fragmentation because only one block at a time is needed but there can be internal fragmentation but it exists only in the last disk block of file.



Disadvantage –

Internal fragmentation exists in last disk block of file.

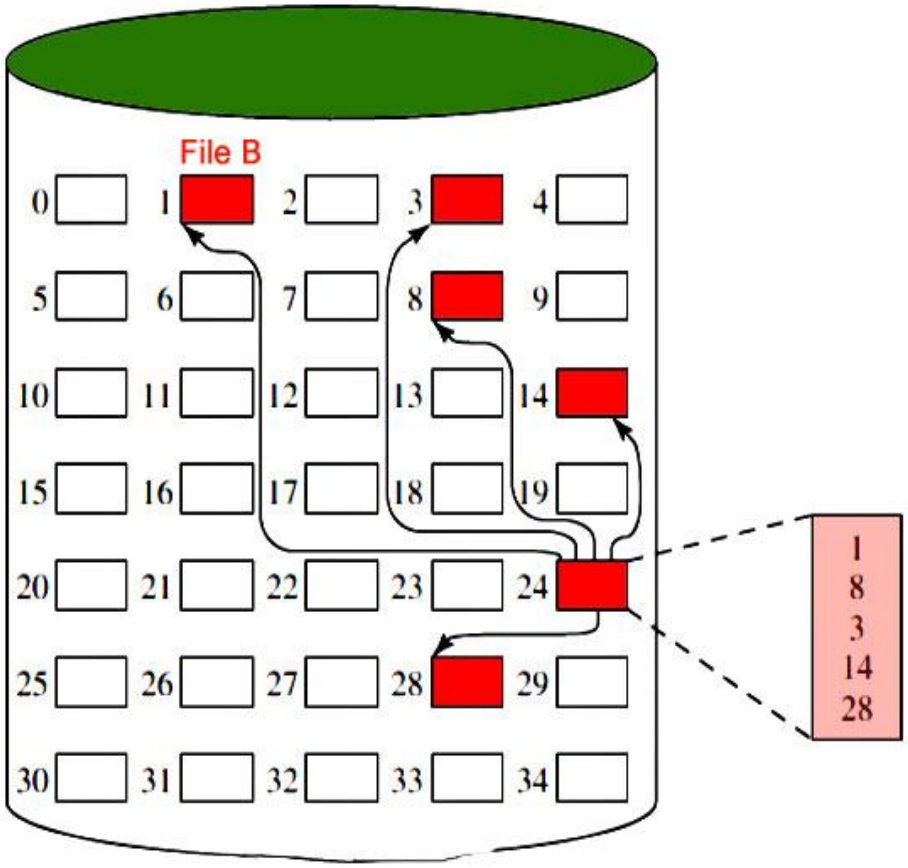
There is an overhead of maintaining the pointer in every disk block.

If the pointer of any disk block is lost, the file will be truncated.

It supports only the sequential access of files.

Indexed Allocation

- It addresses many of the problems of contiguous and chained allocation.
- In this case, the file allocation table contains a separate one-level index for each file: The index has one entry for each block allocated to the file.
- Allocation may be on the basis of fixed-size blocks or variable-sized blocks. Allocation by blocks eliminates external fragmentation, whereas allocation by variable-size blocks improves locality.
- This allocation technique supports both sequential and direct access to the file and thus is the most popular form of file allocation.



File allocation table

File name	Index block
•••	•••
File B	24
•••	•••

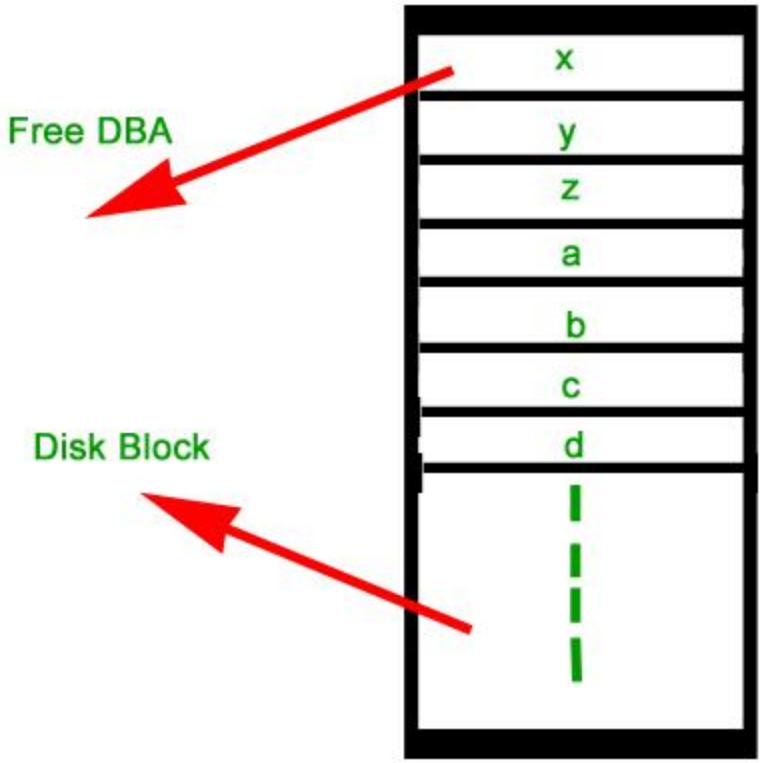
Disk Free Space Management :

- Just as the space that is allocated to files must be managed ,so the space that is not currently allocated to any file must be managed.
- To perform any of the file allocation techniques,it is necessary to know what blocks on the disk are available.
- Thus we need a disk allocation table in addition to a file allocation table.

The following are the approaches used for free space management.

- **Bit Tables** : This method uses a vector containing one bit for each block on the disk.
- Each entry for a 0 corresponds to a free block and each 1 corresponds to a block in use.
  For example: 00011010111100110001 In this vector every bit correspond to a particular block and 0 implies that, that particular block is free and 1 implies that the block is already occupied.
- A bit table has the advantage that it is relatively easy to find one or a contiguous group of free blocks.
- Thus, a bit table works well with any of the file allocation methods. Another advantage is that it is as small as possible.

- **Free Block List** : In this method, each block is assigned a number sequentially and the list of the numbers of all free blocks is maintained in a reserved block of the disk.



- **Grouping –**
  This approach stores the address of the free blocks in the first free block. The first free block stores the address of some, say n free blocks. Out of these n blocks, the first n-1 blocks are actually free and the last block contains the address of next free n blocks.
  An **advantage** of this approach is that the addresses of a group of free disk blocks can be found easily.
- **Counting –**
  This approach stores the address of the first free disk block and a number n of free contiguous disk blocks that follow the first block.
  Every entry in the list would contain:
  - Address of first free disk block
  - A number n

- **1 . What is a File?**
- **2. List the various File Attributes.**
- **3. What are the various File Operations?**
- **5. What are the different Accessing Methods of a File?**
- **6. What is Directory?**
- **7. What are the operations that can be performed on a Directory?**
- **8. What are the most common schemes for defining the Logical Structure of a Directory?**
- **9. What are the Allocation Methods of a Disk Space?**

What are the advantages of Contiguous Allocation?

The advantages are,
Supports direct access
Supports sequential access
Number of disk seeks is minimal

What are the drawbacks of Contiguous Allocation of Disk Space?

The disadvantages are,
Suffers from extrernal fragmentation
Suffers from internal fragmentation
Difficulty in finding space for a new file
File connot be extended
Size of the file is to be declared in advance

What are the advantages of Linked Allocation?

The advantages are,
No external fragmentation
Size of the file does not need to be declared

What are the disadvantages of Linked Allocation?

The disadvantages are,
Used only for sequential access of files.
Direct access is not supported
Memory space required for the pointers.
Reliability is compromised if the pointers are lost or damaged

What are the advantages of Indexed Allocation?

The advantages are,
No external fragmention problem
Solves the size declaration problems
Supports direct access

How free-space is managed using Bit Vector Implementation?

- The free-space list is implemented as a bit map or bit vector. Each block is represented by 1 bit. If the block is free, the bit is 1; if the block is allocated, the bit is 0.