



Files & File System

BY:

ROSHAN R. ROHIT

LECTURER,

IT DEPT,

GPG,SURAT

File



A file is a named collection of related information, which is stored in a secondary storage device.



A file is used to store information permanently and is suitable for storing very large information.



A file is used to store various kind of information, such as text, image, audio, video, database tables, machine codes, and so on.

File

A file is an independent entity from processes, users and machines.

Information stored in files can be shared among different processes, users as well as machines.

A file is a static/passive entity.
Information stored in files remains as it is until user modifies it.

A file has a longer life span as once a program is created and stored on disk, it remains there till it is not deleted.



File System

- A part of an operating system, which deals with files is known as **file system** or **file manager**.
- It is the responsibility of a file system to manage all the files and provide efficient access to information stored in files.
- It provides facilities to create & delete files, read & write contents of files, etc.



File System

- It also provides **directories** to organize files.
- It is also the responsibility of a file system to manage memory space on secondary storage devices when files are created, modified or deleted.
- It also provides a protection mechanism to allow users to administer how other users can access the information in files.

Why we need files?

There are three essential requirements for long term information storage:



It must be possible to store a very **large** amount of information.

Information should not be **lost** when process terminates.

Sharing of information and concurrent access by two or more processes should be possible.

User-view v/s System-view

User-View	System-View
From a user point of view, the most important aspect of a file system is how files appear to users.	From a system point of view, it is important to see how files are managed.
It means how files are constituted, how files are named, how files are protected, what operations are allowed on files, and so on.	It means how files are accessed, how files are allocated memory space and so on.
Here, primary goal is user convenience.	Here, primary goal is efficiency.

File Naming

When a file is created, it is given a name. Later, it is referred by this name.

This is to provide convenience to human users.

Information is stored on physical devices such as disks, but OS hides physical device, and provides logical entity (files) to store information which is the reason sometimes files are referred as logical devices.

File Naming

The exact rules for naming files vary from system to system.

All current OS allows strings of 1 to 8 letters, including digits and some special characters, as legal file names.

Some file systems support case sensitive file names, while others do not. UNIX comes under first category while MS-DOS comes in second category.

File Naming

Many OS support two-part file names, with the two parts separated by a period ('.').

The part following the period is called the file extension which usually indicates something about a file.

In MS-DOS, file names are 1 to 8 characters, plus an optional extension of 1 to 3 characters.

In UNIX, the size of the extension depends upon the user and a file may contain two or more extensions.

File Attributes

Name: A string of alpha-numeric characters and some special characters like underscore ('_'). It is used to refer files in a convenient way,

Identifier: It is a unique tag, usually a number which identifies the file within the file system. It is used by OS to refer files.

Type: It is used to identify type of a file and is generally expressed in form of a file extension.

Location: This is a pointer to the device and location on that device of the file.

File Attributes

Size: It specifies the current size of the file as well as the maximum allowed size of the file.

Protection: This information determines who can read file, write file, execute file and so on.

Usage Count: This value indicates the number of processes that are currently using this file.

Time, Date, & User Identification: It specifies information regarding file creation, update and last access.

File Operations

Create:

- A new file can be created by a system call embedded in a program or by an OS command issued by an interactive user.
- While creating a file, two steps are followed: First, necessary disk storage space is allocated to the file and Second, an entry for the new created file is made in the directory.

Delete:

- When a file is no longer needed, it must be deleted to free up disk space.
- While deleting a file, two steps are followed: First, all the file space on the disk is released and Second, directory entry is erased.

File Operations

Open:

- Before using a file, it must be opened.
- File attributes and data contents are fetched in main memory for rapid access on later calls.
- Memory space in main memory is allocated to store fetched information.

Close:

- When use of a file is finished, it should be closed to free up main memory space.
- File attributes and data contents are stored back on disk. This may contain modified information if file is updated.

File Operations

Read:

- Data are read from the file.
- The system maintains a read pointer which specifies the location in a file from where to read data contents.
- The read pointer is updated automatically after each read operation.

Write:

- Data are written to the file.
- The system maintains a write pointer which specifies the location in a file from where to write data.
- Initially write pointer points to the starting location in a file.
- If data are written at the end of a file, file size increases.
- The write pointer is updated automatically after each write operation.

File Operations

Append:

- This is restricted form of a write operation.
- Here, data are only added to the end of file.

Seek:

- For random access files, a location is needed to specify from where to start read/write operation.
- This operation repositions the file pointer to a specific location in a file.

Get Attributes:

- This operation is used to retrieve file attributes.

Set Attributes:

- This operation is used to write file attributes.

Rename:

- This operation is used to change the name of an existing file.

File Structures



File structure is the way of storing data contents (information) in a file.



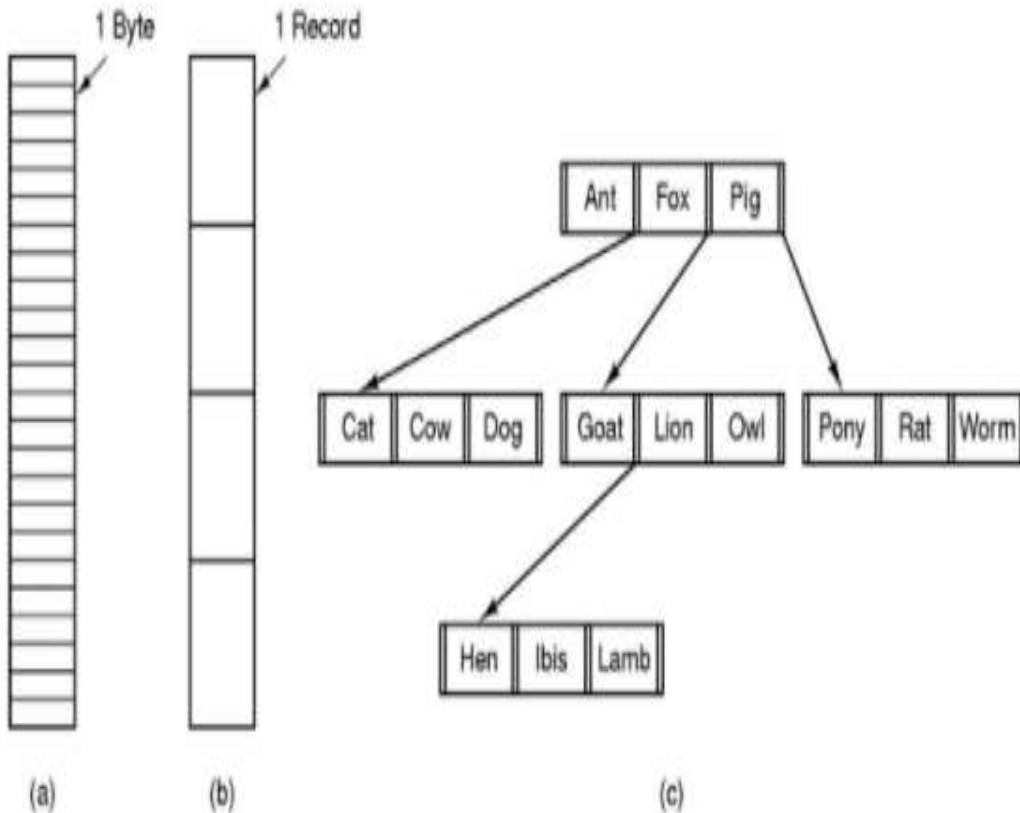
It represents how data are stored in a file.



There are 3 common file structures : Byte sequence, Record sequence and Tree.

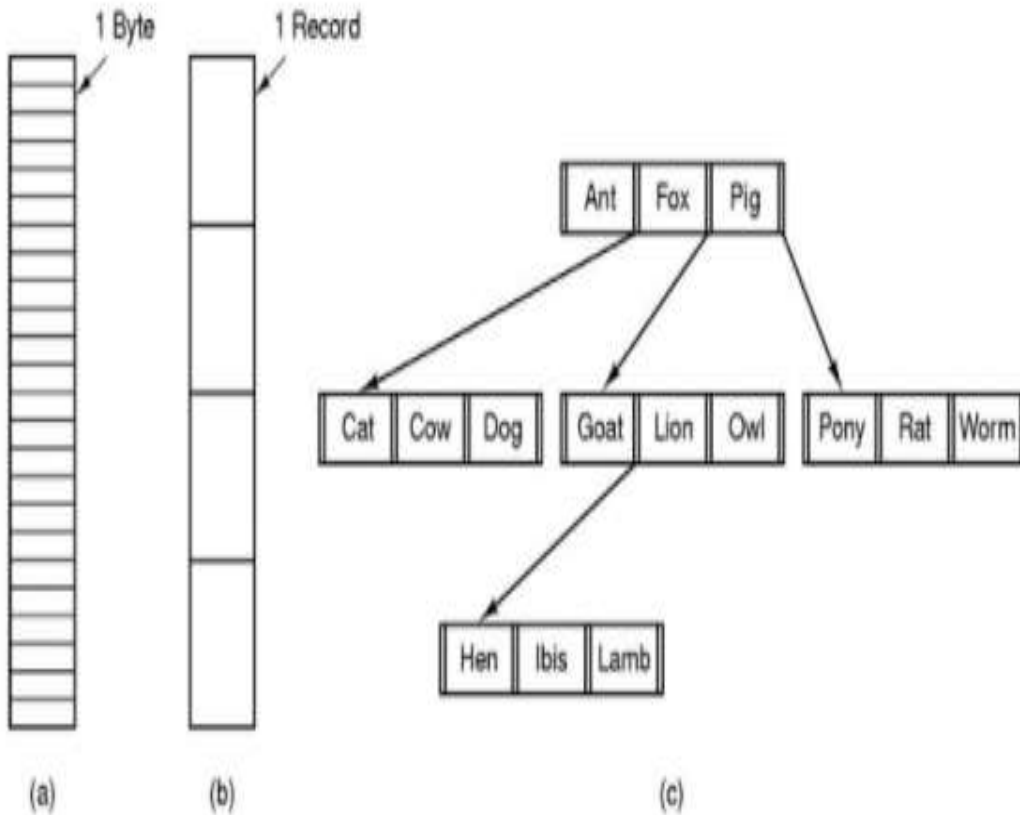
Byte Sequence

- Data are stored as an unstructured sequence of bytes.
- OS does not know or care what is in the file. Entire file is treated as an array of bytes.
- Generally , read operation returns one byte, while write operation overwrites or appends one byte.
- Both UNIX and Windows use this approach.



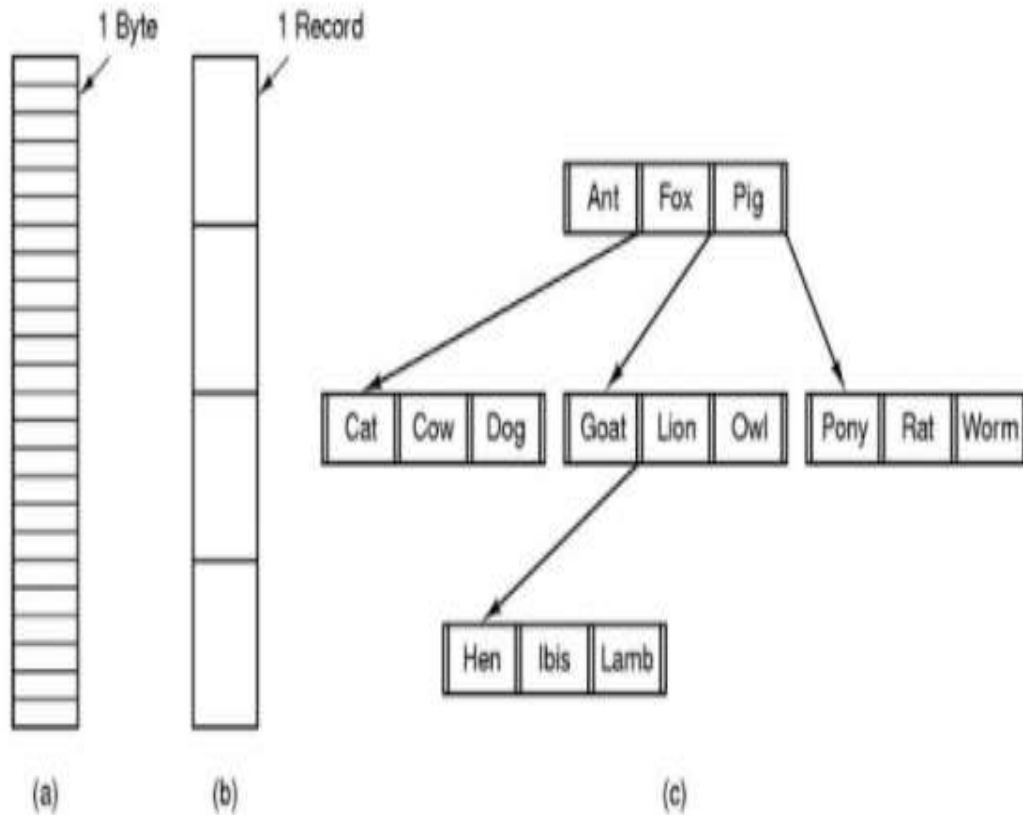
Record Sequence

- Data are stored as a sequence of fixed length records where each record have its own structure.
- Read operation returns one record, while write operation overwrites or appends one record.
- It is suitable for special applications such as database related applications.



Tree

- Data are stored as a tree of records where each record has its own structure.
- Records may be of varying length.
- Each record has a key in a fixed position in the record.
- Records are sorted by key for easy search & retrieval.
- Used with large mainframe systems.



File Types

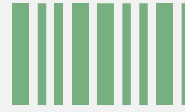
Regular Files: They contain user information.

Directories: They are containers for files and other sub-directories.

Character special files: They are related to I/O, used to model serial I/O devices such as terminals, printers, etc.

Block special files: They are also related to I/O, used to model block I/O devices such as disks.

Categories of Regular files



Encode type into **file extension**: This is common technique for implementing file types, same file extensions are used for all the files of same type.



Encode type into **file attribute**: Each file contains an attribute called file type. File type can be encoded in this attribute.



Encode type into **file data**: File type can also be encoded in file data.

ANY
QUESTIONS?



File Security & Protection Mechanism



BY:

ROSHAN R. ROHIT

LECTURER,

IT DEPT,

GPG, SURAT

Introduction

Files are used to store information.

No one wants to lose information stored in files in any way.
Everyone wants information stored in the files to be safe.

Here, safety comes in two ways: Reliability & Protection.

Reliability

Reliability means safety from physical damage.

File systems can be damaged by hardware problems, fire, dirt, power failures, head crashes and so on.

Reliability is generally provided by keeping more than one copies of files, means duplicating files.

Files are duplicated mostly on some different locations, or in different devices.

If original files are damaged, then they can be retrieved back from the backups.

Protection

Protection means safety from improper access.

It is required where files can be shared among various users.

Some files need to be shared among all users, while some need to be shared among limited users.

Protection can be given by limiting the type of file access to perform some operation on that file.

Password

A password is associated with each file. Users can access a file, only if he/she knows the password.

This scheme looks good, but it suffers from several disadvantages.

There may be large number of files in a computer system. So, users need to remember lots of passwords.

To avoid this problem, if a single password is used for all files, then once it is discovered, all files are accessible.

Access Control

A list, called an access control list (ACL), is associated with each file which specifies the username and type of access allowed for that user.

This list is generally kept in directory entry along with file name and other information.

First problem with this method is constructing an ACL which is tedious task as all users of the system are not known in advance.

Second problem with this method is that ACL is kept in directory entry. Now, directory entry should be of varying length instead of fixed length leading to more complicated space management.

Access Control

Solution to the problem is to divide all users in various categories and then to allow access on such categories instead of individual users.

UNIX uses such type of scheme.

It divides users in three categories: i) Owner, ii) Group and iii) Others.

Access is given on such category. All the users of that category will be allowed that access.





File Access Methods

*BY:
ROSHAN R. ROHIT
LECTURER,
IT DEPT,
GPG, SURAT*

Introduction



When a file is used, information is read and accessed into computer memory and there are several ways to access this information of the file.



Some systems provide only one access method for files.



Other systems, such as those of IBM, support many access methods, and choosing the right one for a particular application is a major design problem.



There are three ways to access a file into a computer system: Sequential-Access, Direct Access, Index sequential Method.

Sequential Access



Data is accessed one record right after another record in an order.



When we use read command, it move ahead pointer by one



When we use write command, it will allocate memory and move the pointer to the end of the file



Such a method is reasonable for tape.

Direct Access



Direct access method also known as relative access method.



A file-length logical record that allows the program to read and write record rapidly in no order.



The direct access is based on the disk model of a file since disk allows random access to any file block.



For direct access, the file is viewed as a numbered sequence of block or record. Thus, we may read block 14 then block 59, and then we can write block 17. There is no restriction on the order of reading and writing for a direct access file.

Index Sequential method



It is the other method of accessing a file that is built on the top of the sequential access method.



These methods construct an index for the file. The index, like an index in the back of a book, contains the pointer to the various blocks.



To find a record in the file, we first search the index, and then by the help of pointer we access the file directly.



Disk Organization

BY:

ROSHAN R. ROHIT

LECTURER,

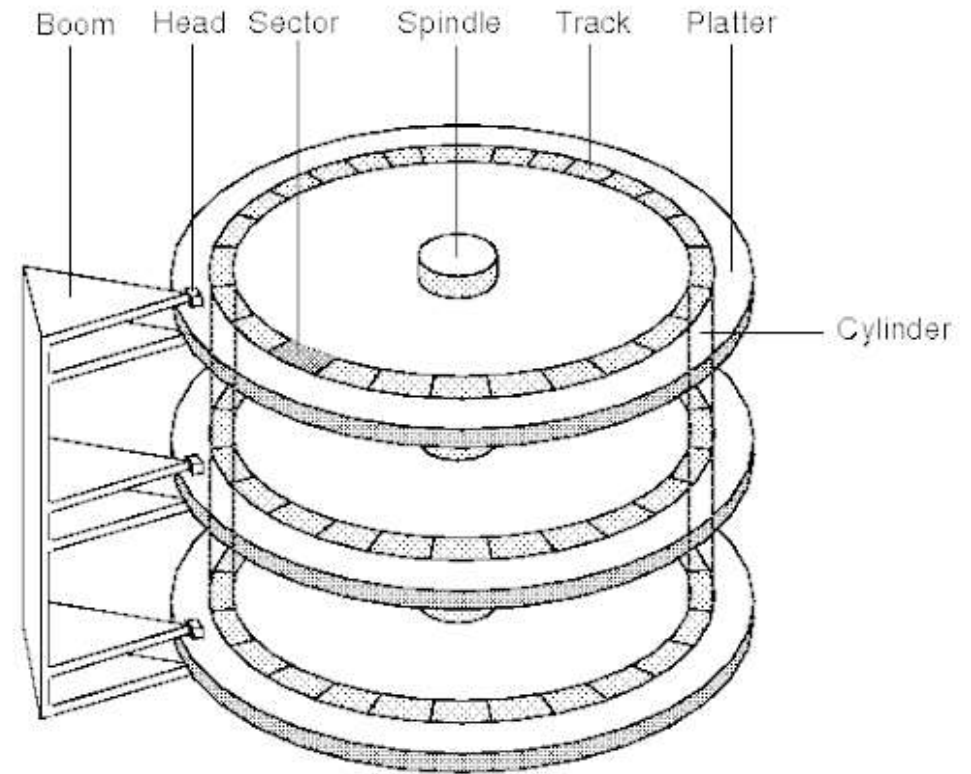
IT DEPT,

GPG,SURAT



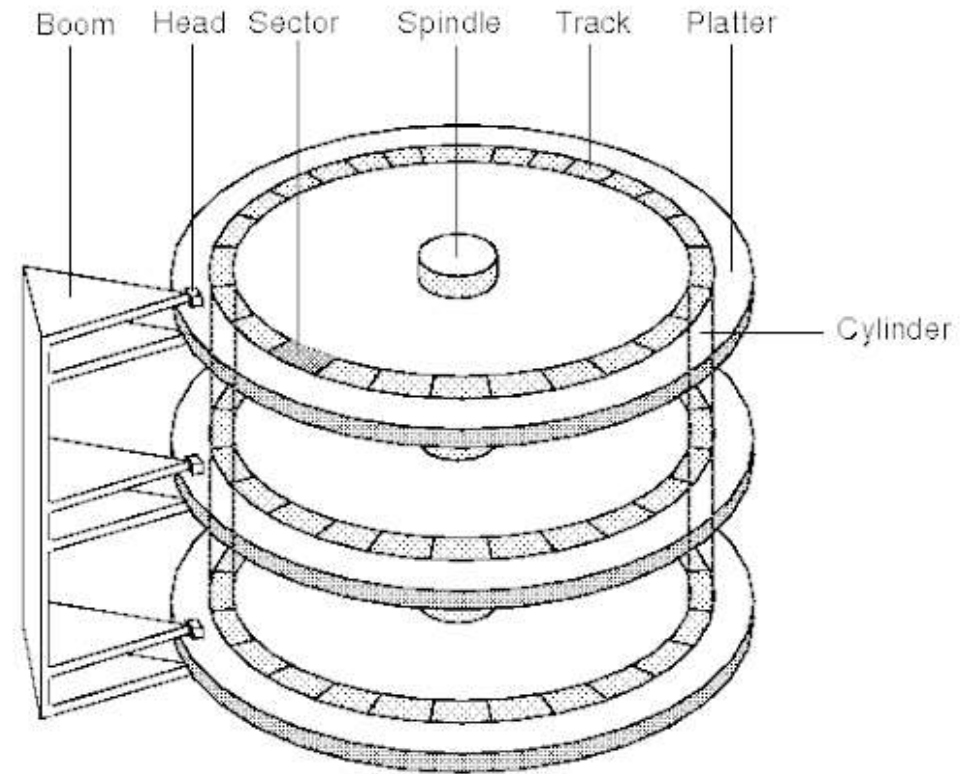
Physical Structure of a Disk

- + A hard disk is a sealed unit which contains one or more circular platters.
- + Each platter contains magnetic coating on both of its surfaces.
- + The platters are stacked one on top of another and they rotate together around a central spindle.



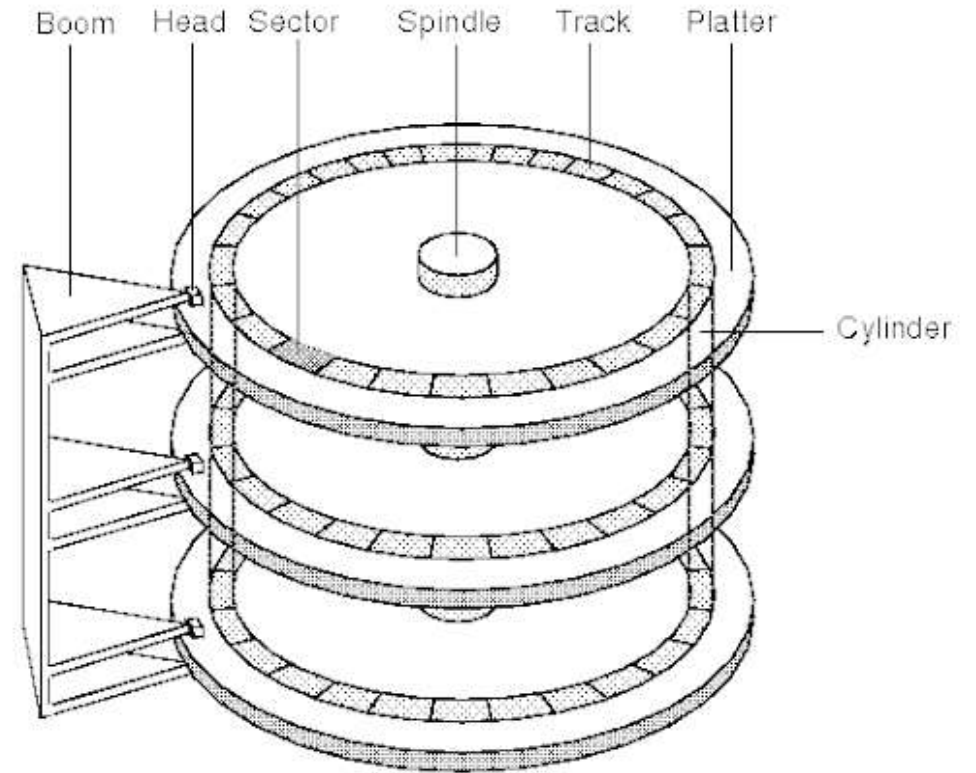
Physical Structure of a Disk

- + Data is read from and written to these platters using several read/write **heads**.
- + A platter can be thought of as a collection of circular, concentric, flat rings called **tracks**.



Physical Structure of a Disk

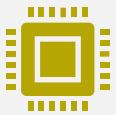
- + Each track is divided into a fixed size blocks, called **sectors**.
- + A group of tracks with the same radius are called **cylinders**.



Logical Structure of a Disk



Disk is considered as a large one-dimensional array of fixed size logical **blocks**.

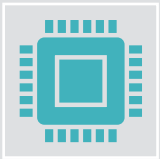


Size of blocks can be varied from system to system which is normally 512 bytes.

Logical Structure of a Disk




Whenever any file requires storage space, it is allocated one or more blocks either contiguously or non-contiguously as per the allocation method.



Whenever there is a need to access any information from disk, operating system specifies a logical block number which is mapped to a physical cylinder number, head number and sector number.

Addressing

In order to store data on the disks or to read it back, the OS needs a way to specify where to write the data, or from where to read it on disk.



The OS tells the drive the required position and the drive then moves the head to the right position and either reads or writes data.



There are two methods to provide an address of such location: i) CHS and ii) LBA.

CHS (Cylinder-Head-Sector)



This method identifies sectors by simply specifying the cylinder (radius), head (platter side), and sector (angular position) numbers.



Cylinders are numbered from 0 to some maximum from outer cylinder to inner cylinder.



Sectors on each track are numbered from 0 to some maximum. Also, read/write heads are given numbers.

LBA (Logical Block Addressing)



This method identifies sectors by simply specifying the sector number.



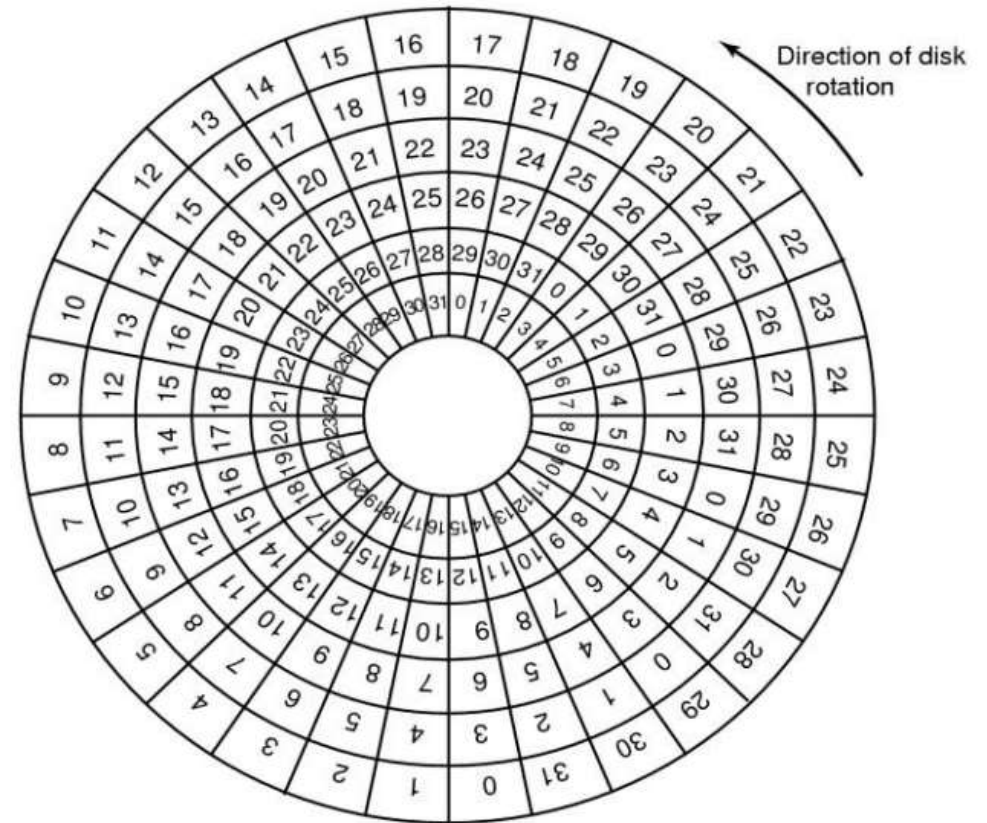
Each sector of the disk is assigned a sequential number starting from 0.



Disk is considered as large 1-D array of fixed size logical blocks which are mapped into the sectors of the disk sequentially.

LBA (Logical Block Addressing)

- + Sector 0 is the first sector of the first track on the outermost cylinder.
- + First, all the sectors on that track are numbered and then other tracks in that cylinder are covered.
- + The rest of the cylinders are covered in order from outermost to innermost.



Disk I/O



It refers to the read and write operations on data stored in disk.



Data is read from disk to memory and written from memory to disk.



Hard disk controller is an interface between operating system and disk which converts various commands provided by an OS into proper signals so that disk hardware can understand it.



Addressing system to identify locations on hard disk can be either LBA or CHS.

Disk I/O

Once the location is identified, the following steps are followed to perform read/write operation:

1. **Move** the disk arm to position read/write head at specified cylinder. Time required for this operation is called **seek time**.
2. **Wait** till specified sector is directly comes above/below the read/write head. Time required for this operation is called **rotational latency**.
3. **Read/write** the data.





Disk Space Allocation Methods

BY
ROSHAN R. ROHIT
LECTURER,
IT DEPT,
GPG, SURAT

Introduction

When file is created, storage space is allocated to it.

When new data is added to existing file, file size grows, and it needs extra storage space.

Similarly, storage space is released when file is deleted or data from file is deleted.

There are two main goals which should be fulfilled while allocating space on disk to files which are described as follows.

1. Disk space should be utilized effectively

2. Files should be accessed quickly

Disk allocation methods

Whenever there is a need to allocate storage space to a file, one or more free disk blocks are found and allocated to a file.

In a similar way, whenever a file is deleted, allocated disk blocks are freed. Such disk blocks can be re-used for other files.

There are mainly three different allocation methods:

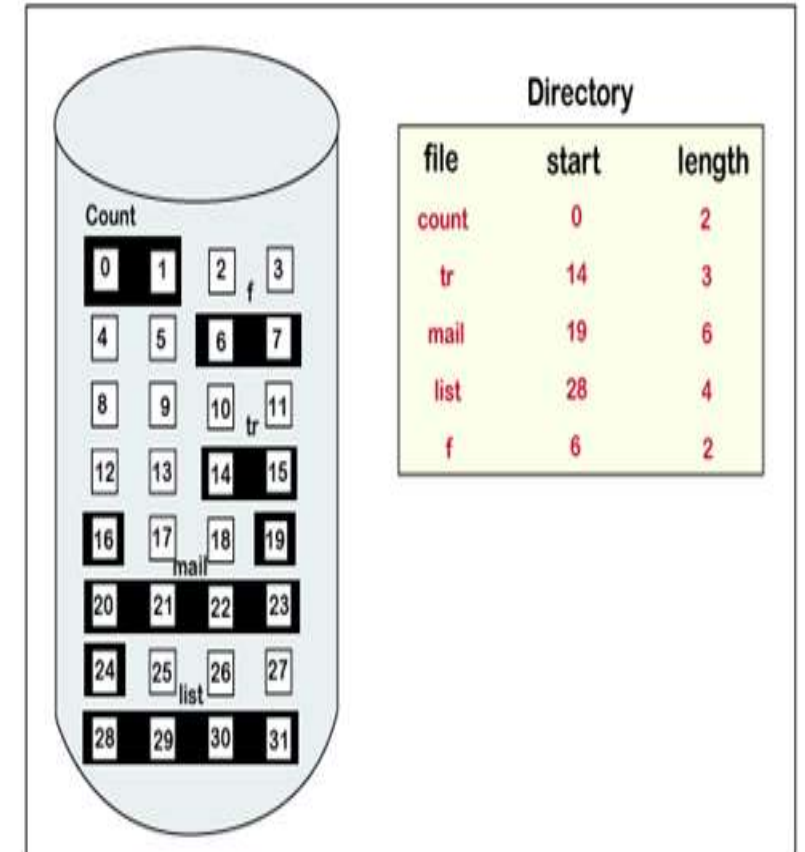
1. Contiguous allocation

2. Linked allocation

3. Indexed Allocation

Contiguous Allocation

- Each file occupies a set of contiguous blocks on disk.
- On a disk with blocks of 1-KB, a file of size 50 KB would contain 50 consecutive blocks, with 2-KB blocks, it would have 25 consecutive blocks.
- When a file is created, a disk is searched to find out a chunk of free memory having enough size to store a file. Once such chunk is found, required memory is allocated.
- This method is widely used on CD-ROMs.



Contiguous Allocation

Advantages:

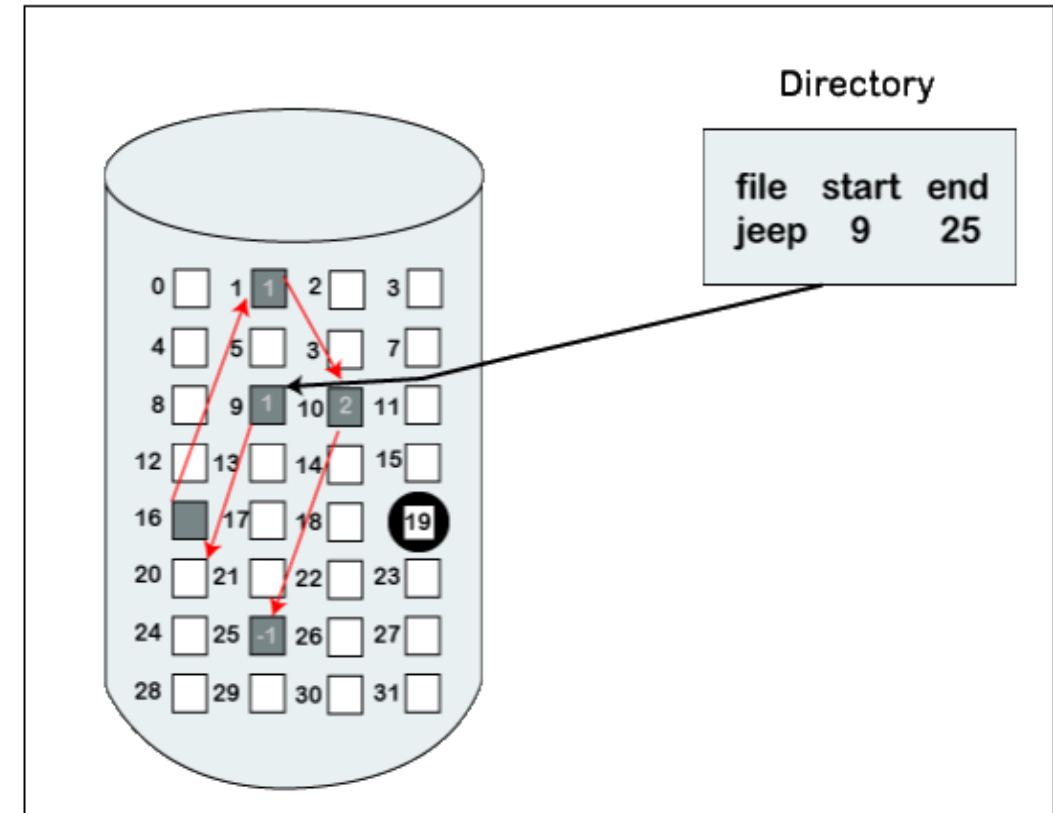
- Simple to implement.
- File access is quick.

Disadvantages:

- Finding free space for a new file is time consuming.
- If size of an existing file increases, it may not be possible to accommodate such extension.
- External fragmentation is possible.

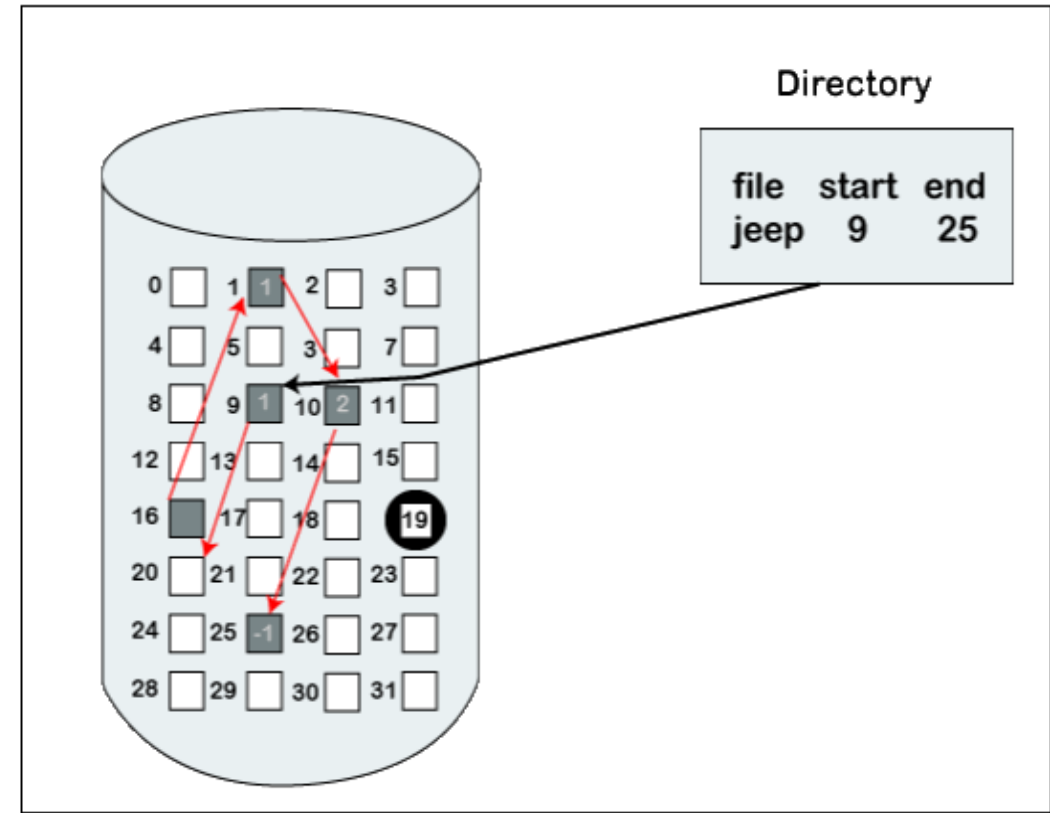
Linked Allocation

- Each file is a linked list of disk blocks. Each linked block contains pointer to the next block in a list.
- These disk blocks may be scattered anywhere on the disk.
- Directory entry contains the start and last block numbers in a linked list.
- When a new file is created, a new directory entry is created. Initially it contains 'null' as both the block numbers.



Linked Allocation

- A write to the file causes free data blocks to be added to the file; such blocks are added at the end of a linked list.
- To read a file, all blocks are read by following the pointers from block to block.
- An important variation on the linked allocation method, called File Allocation Table (FAT), is used by the MS-DOS and other Windows operating systems.



Linked Allocation

Advantages:

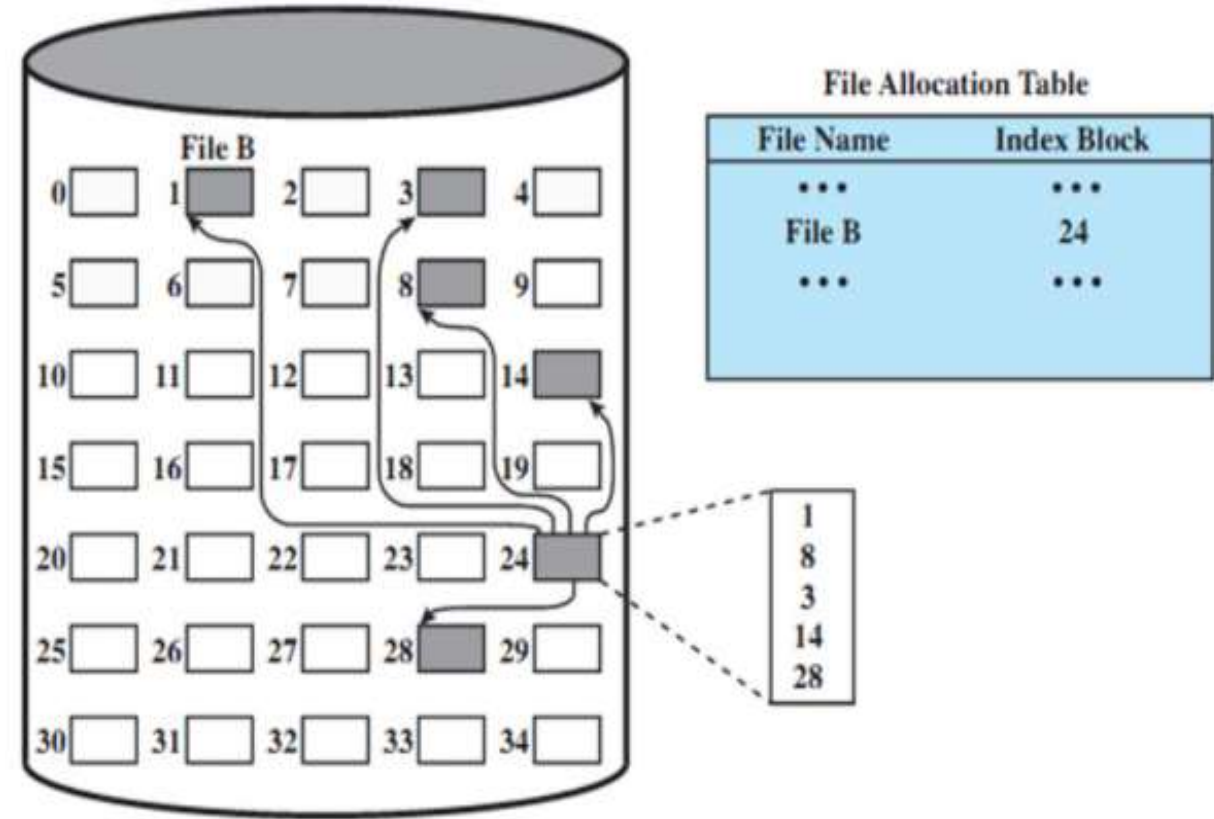
- It does not suffer from external fragmentation.
- Any free disk block can be allocated to a file which does not need to be consecutive, hence disk space can be utilized effectively.

Disadvantages:

- File access is time consuming.
- Random access is not possible directly.
- Extra space is required for pointers in each data block.

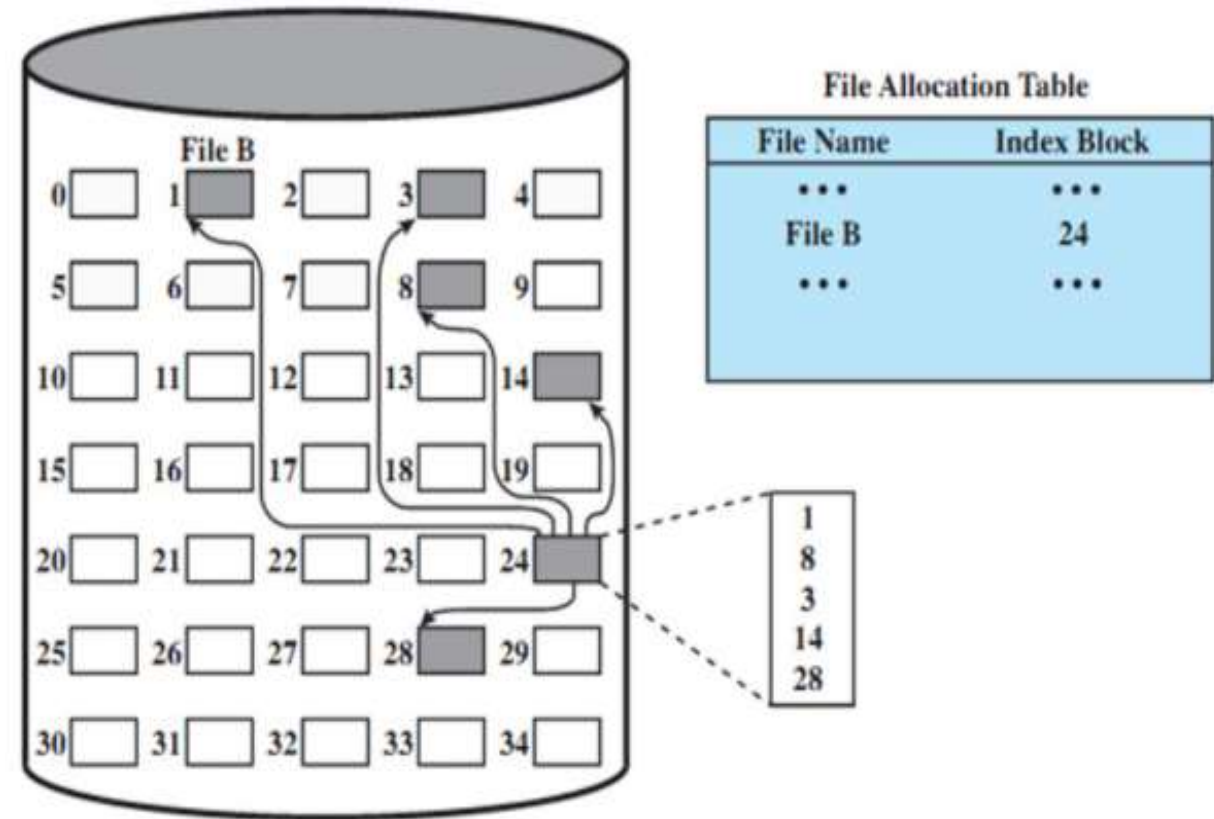
Indexed Allocation

- In a linked allocation, pointers to various disk blocks are scattered on disk among various disk blocks.
- Due to this problem, linked allocation cannot support efficient direct access.
- Indexed allocation solves this problem by bringing all the pointers together into one location : the **Index Block**.
- Each file contains its own index block which is an array of 'disk block addresses'.



Indexed Allocation

- The 'i'th entry in the index block points to the 'i'th block of the file.
- Directory entry contains file name and index block number.
- When new file is created, a new directory entry is created. Initially, all pointers in index block are set 'null'.
- When the 'i'th block is first written, a free disk block is allocated, and its address is put in the 'i'th entry in index block.



Indexed Allocation

Advantages:

- It does not suffer from external fragmentation.
- Direct access is efficient.

Disadvantages:

- It suffers from wasted space. Index block may be partially filled which wastes remaining memory space of an index block.
- Maximum allowable file size depends on size of an index block.

**ANY
QUESTIONS?**





Directory Systems

BY

ROSHAN R. ROHIT

LECTURER,

IT DEPT,

GPG, SURAT

Basics

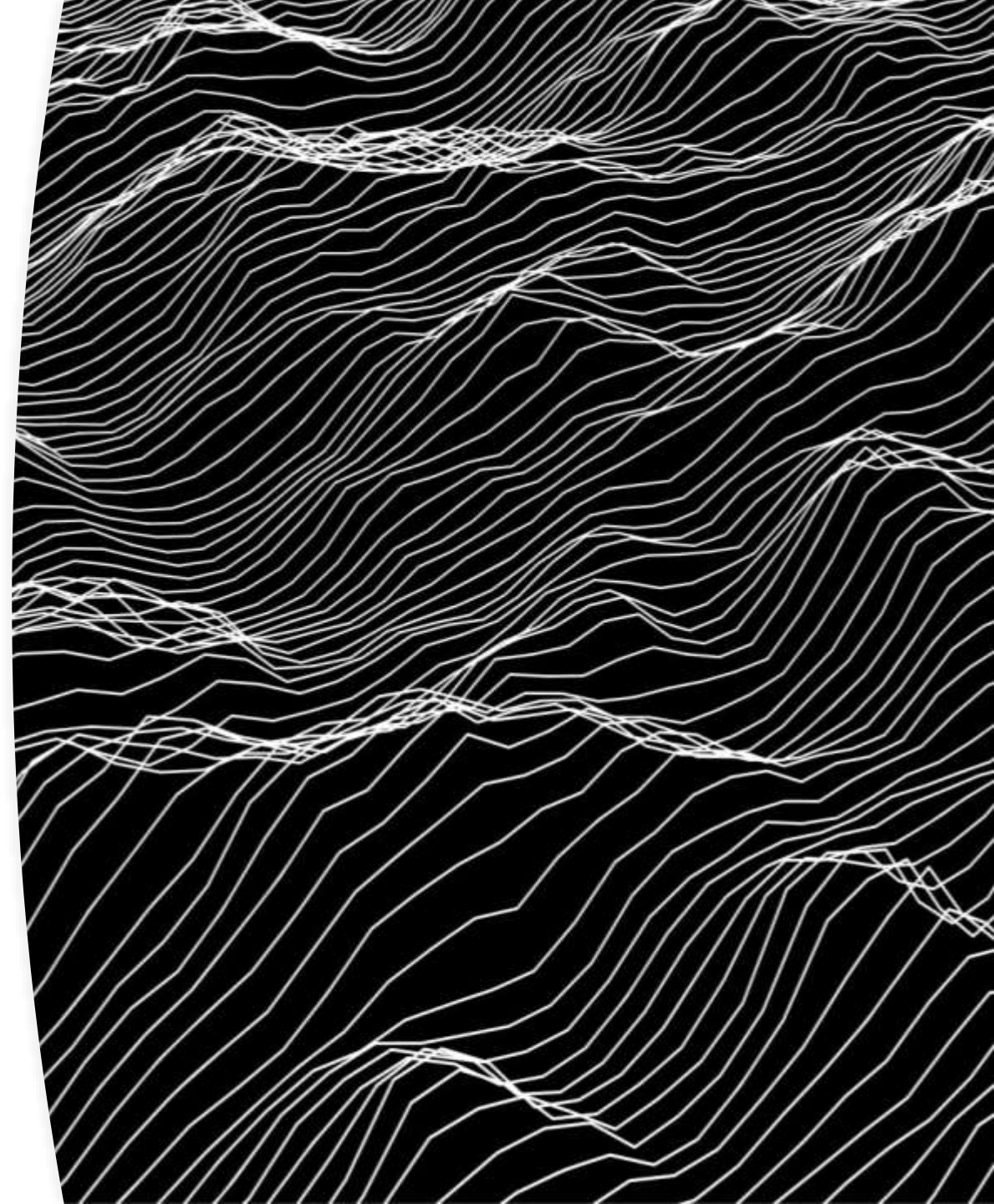
- A **directory** is a container for other files and sub-directories.
- In GUI based system, directories are generally represented as **folders**.
- They are used to group and organize files & other directories, called sub-directories.
- Directories provide a hierarchical file structure.
- Generally, directory entry contains a file name and a file identifier.
- An operating system uses directory entries to map file names to file identifiers.





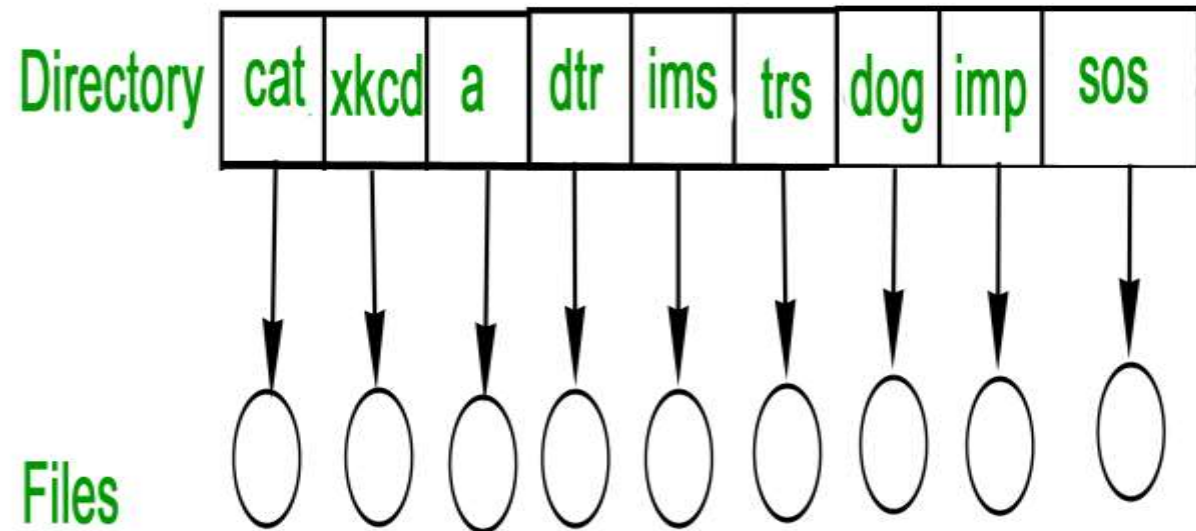
Directory Structures

- Directory Structures refer to the way how directories and files are organized.
- There are five different types of directory structures available and all of them have their own advantages and disadvantages.



Single Level Directory

- This is the simplest directory structure.
- All files are contained in the same directory, sometimes referred to as root directory.
- Directory can contain only files.
- This type of structure was used in early computers.



Single level Directory

Advantages:

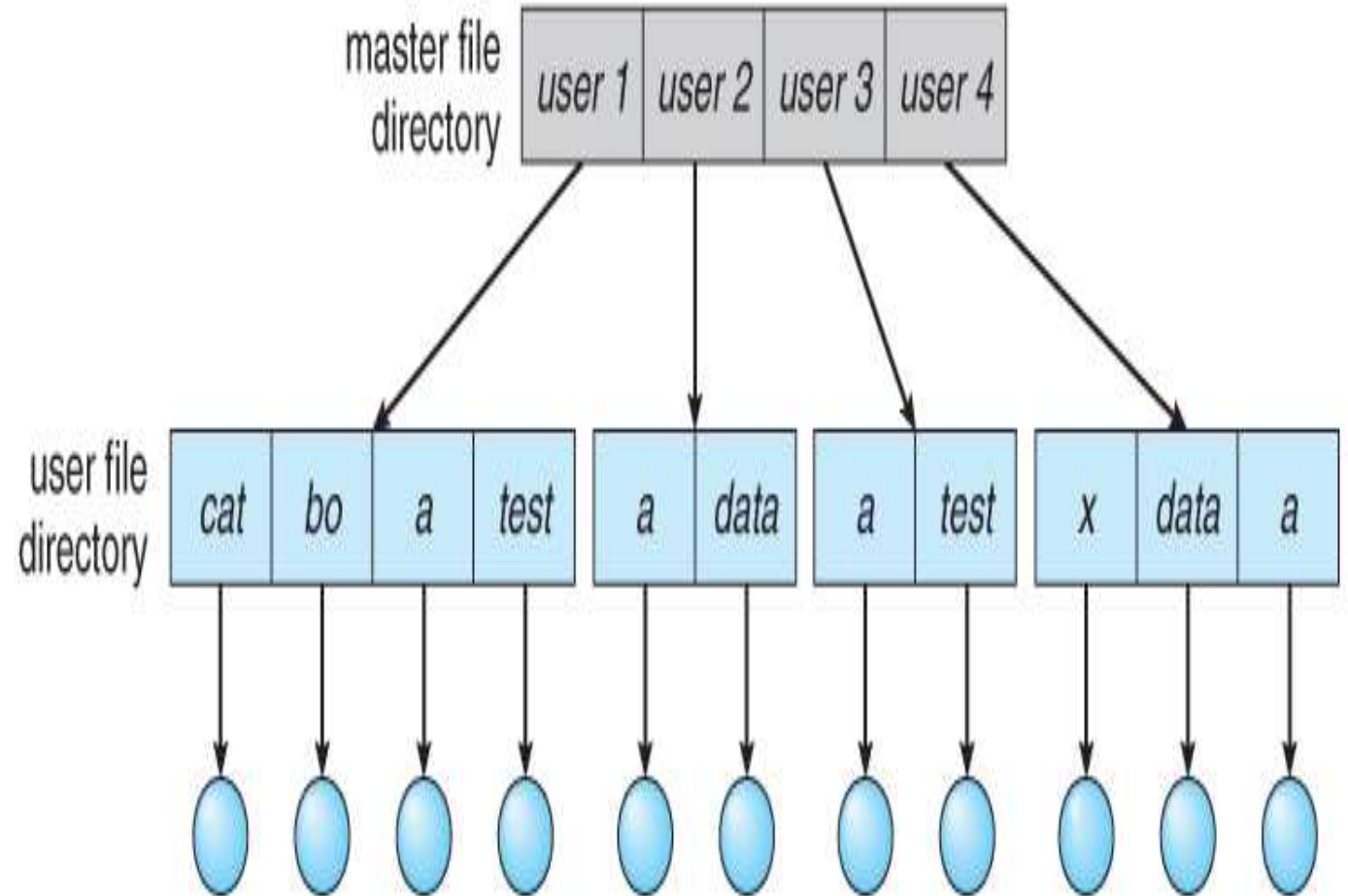
- This type of structure is very simple.
- As there is only one directory, searching a file is very quick.

Disadvantages:

- It is not suitable for multi-user systems.
- It is even not suitable for single user system when number of files becomes too large, users cannot remember the names of all files.
- Different files cannot be grouped.

Two-Level Directory

- It gives each user a private directory.
- This is to avoid file name collisions among different users.
- First level of directory is a root directory or master file directory(MFD); and second are user file directories (UFD) or simply directories.



Two-level Directory

Advantages:

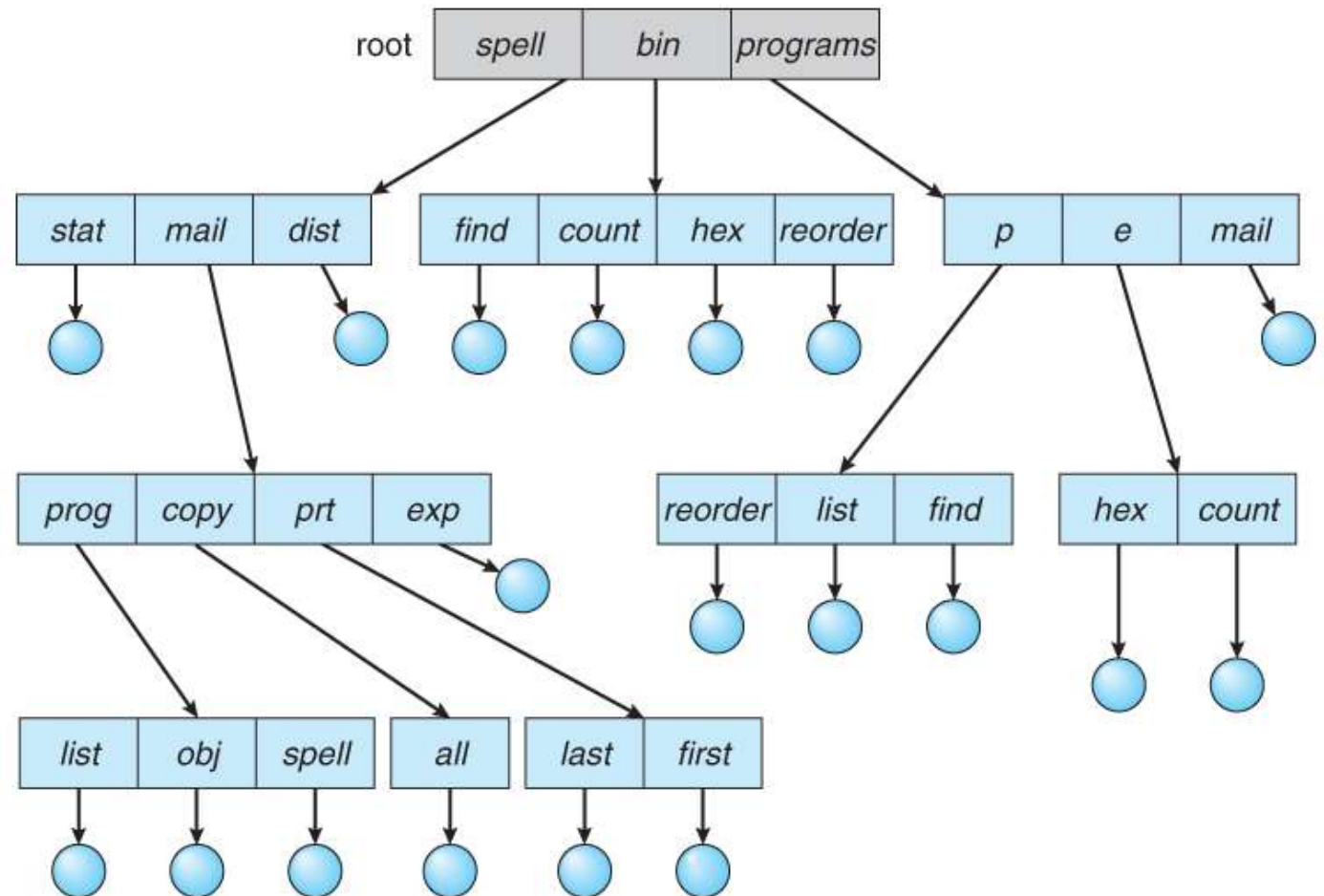
- It avoids file name collisions.
- It can be used in multi-user systems.
- Searching is efficient here; as a user have to search in its own single directory.

Disadvantages:

- It is not suitable for users having large number of files.
- One extra system directory is required to store system files.
- Different files cannot be grouped.

Tree Structured Directories

- It permits users to create own sub-directories.
- This is also called a hierarchical file system as more than two levels of directories are possible.
- Directories can contain files as well as sub-directories also.
- Each file contains a unique file path.



Tree Structured Directories

Advantages:

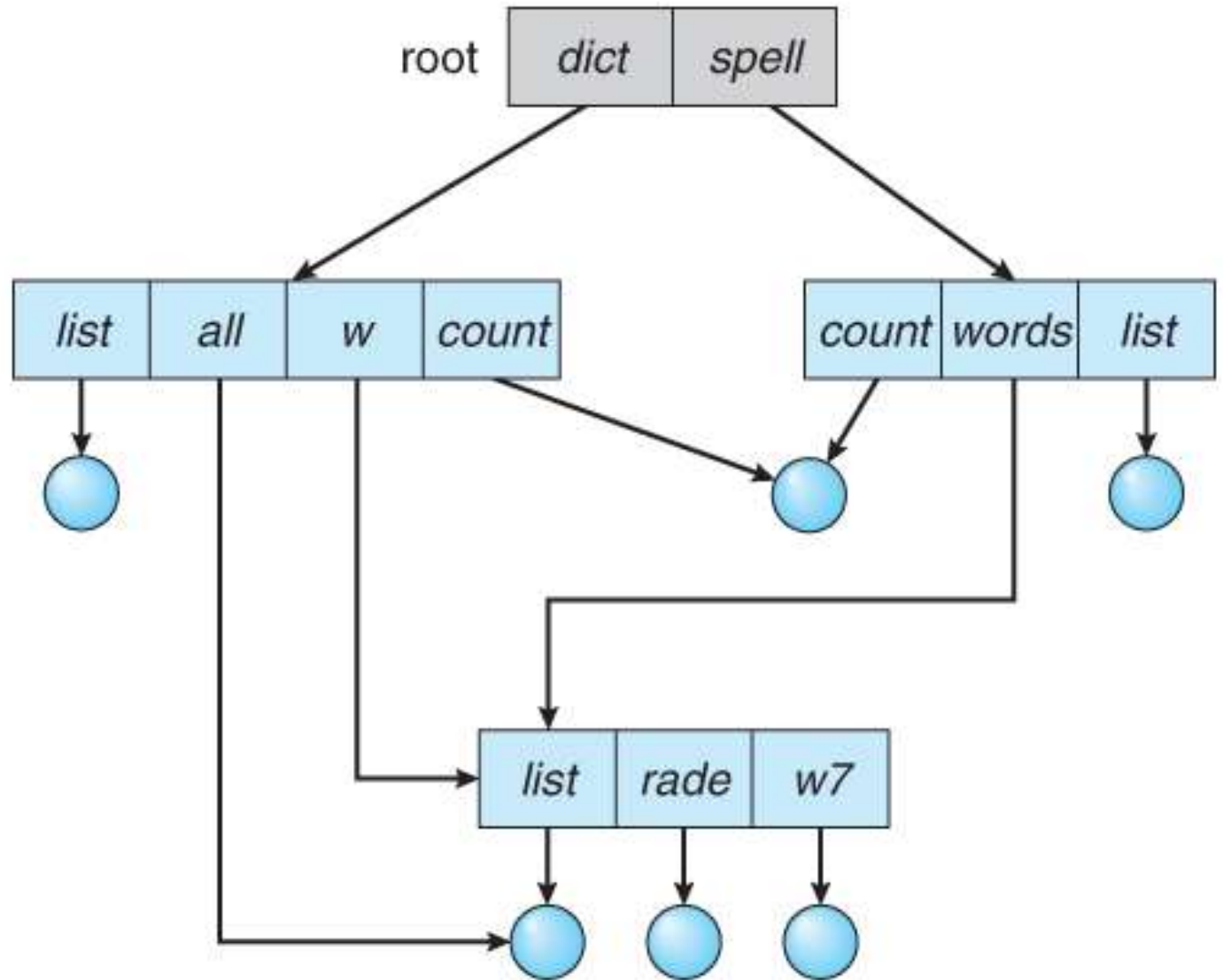
- It avoids file name collisions.
- It can be used in multi-user systems.
- Different users can be grouped here.

Disadvantages:

- Sharing of files and directories is not possible here.

Acyclic Graph Directory

- It permits users to create shared files and directories.
- Shared files and directories can be created using 'link' operation which allows a file or a directory to appear in more than one directory.
- A file may contain more than one file path.
- This directory structures does not contain cycles.



Acyclic Graph Directory

Advantages:

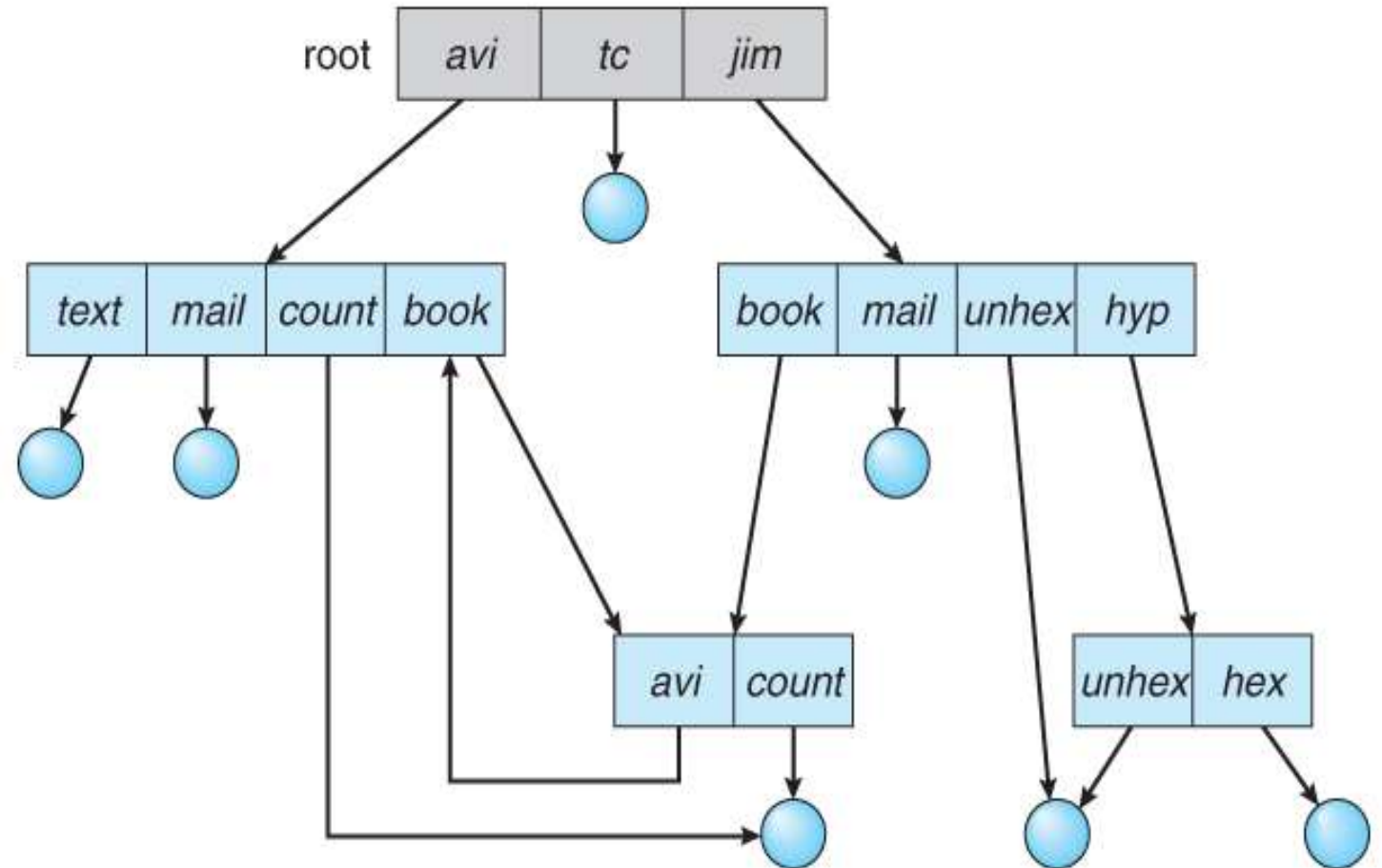
- It allows sharing of files and directories.

Disadvantages:

- Deletion of shared file or directory is complex.
- A file may have multiple file paths which may create problem in some operations.
- It is difficult to ensure that there are no any cycles in a directory structure.

General Graph Directory

- It is same as acyclic graph directory, but it allows cycles in a directory structure.



General Graph Directory

Advantages:

- It allows sharing of files and directories.

Disadvantages:

- Searching should be done carefully else it may get into infinite loops.

File Paths



A file path is a character string divided into separate components by special characters such as slash ('/').



It is used to uniquely specify a location of a file or a directory in a directory structure.



Different files may contain same file names, but file paths will be unique per file.



Special characters which are used to separate different components depend on the operating system.

Absolute File-paths

It starts with the root directory of a file system.

File paths are given with respect to the root directory.

Examples:

C:\Progs\CPP\test.cpp (Windows)

/user/home/test.c (UNIX)

Relative File-paths

It starts with the current directory designated by a user.

File paths are given with respect to such current directories.

Dot ('.') and dotdot ('..') can be used as a current directory.

Example: Consider that two files test.cpp and test.c contains absolute file path as given above.

If 'C:\Progs' is a current directory, then path for file test.cpp can be given as CPP\test.cpp

ANY
QUESTIONS?



LINUX FILE SYSTEM STRUCTURE

BY:

ROSHAN R. ROHIT

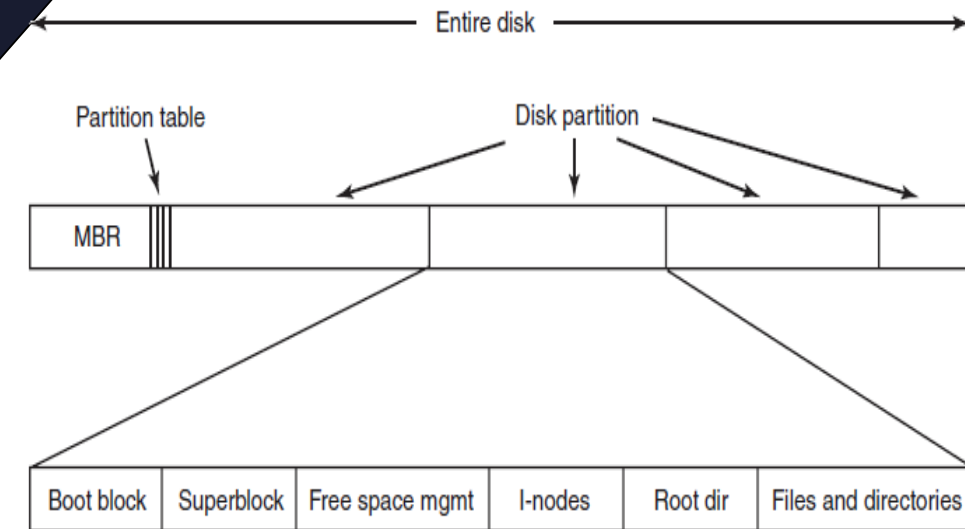
LECTURER,

IT DEPT,

GPG,SURAT

File System Layout

- Files are stored on secondary storage devices like hard disks, CD-ROMs, DVDs, magnetic tapes, pen drives and so on.
- To manage such large storage space, disks are divided into one or more partitions, known as drives. Every partition can have its own independent file system.
- A possible file system layout is shown in the figure.



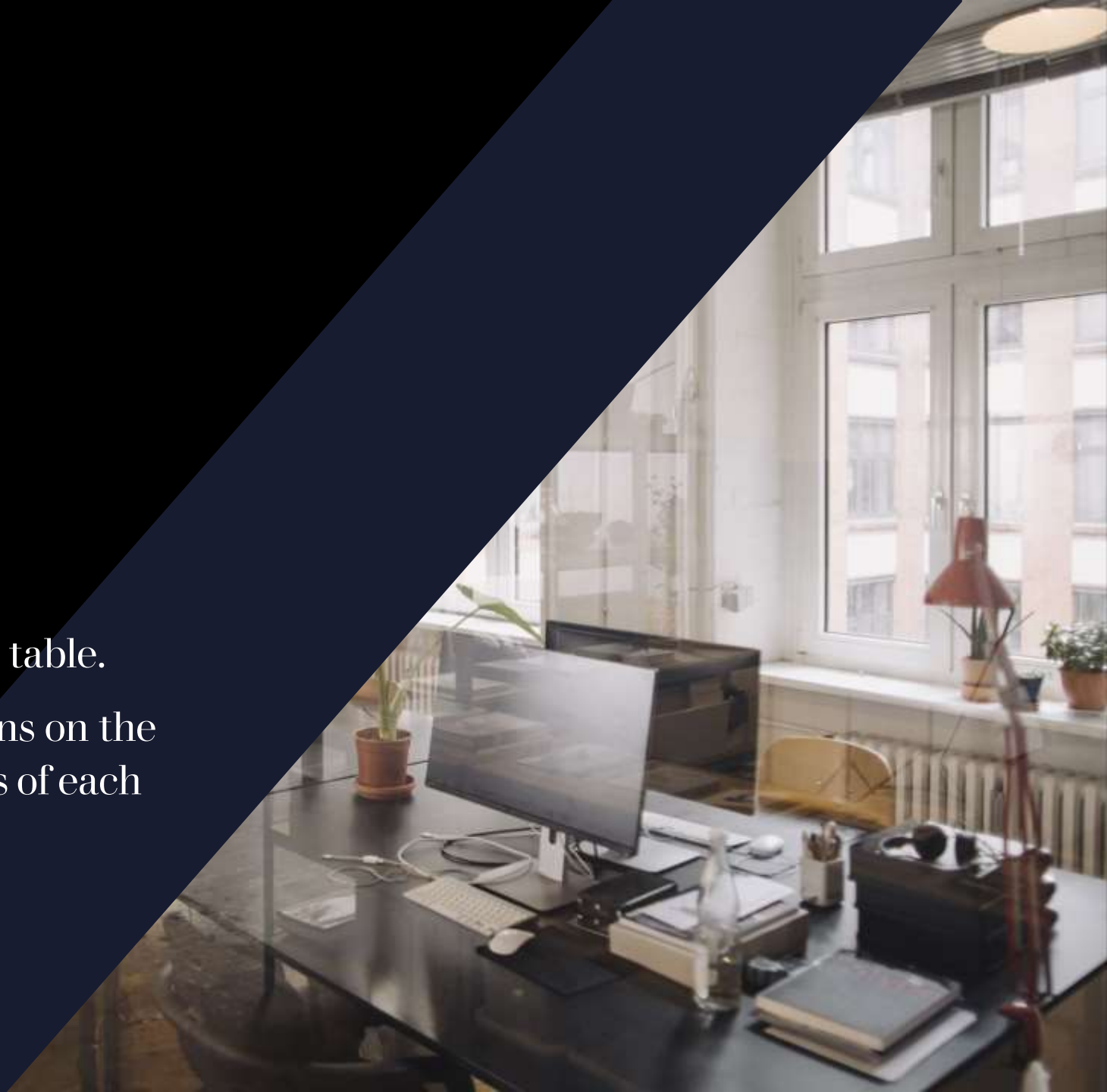
Parts of file system

MBR (Master Boot Record)

- It is the starting sector of the disk.
- It is used to boot the computer.

Partition Table

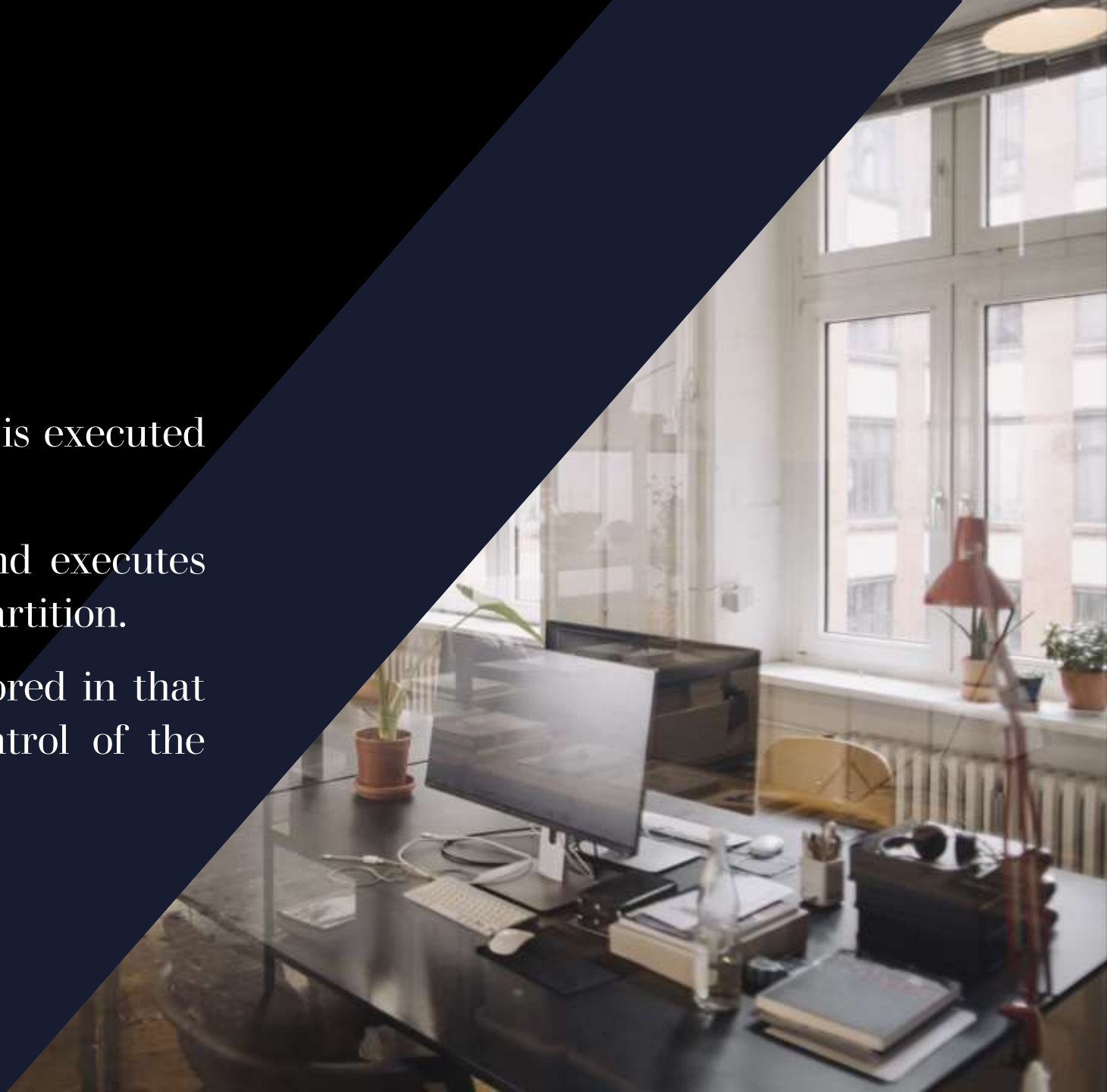
- The end of the MBR contains the partition table.
- This table is used to locate various partitions on the disk. It gives starting and ending addresses of each partition.



Parts of file system

Boot Block

- It is the first block of each partition.
- When the computer is booted, the MBR is executed by the BIOS.
- The MBR locates the active partition and executes the code stored in the boot block of that partition.
- This code loads the operating system stored in that partition after which the OS takes control of the computer system.



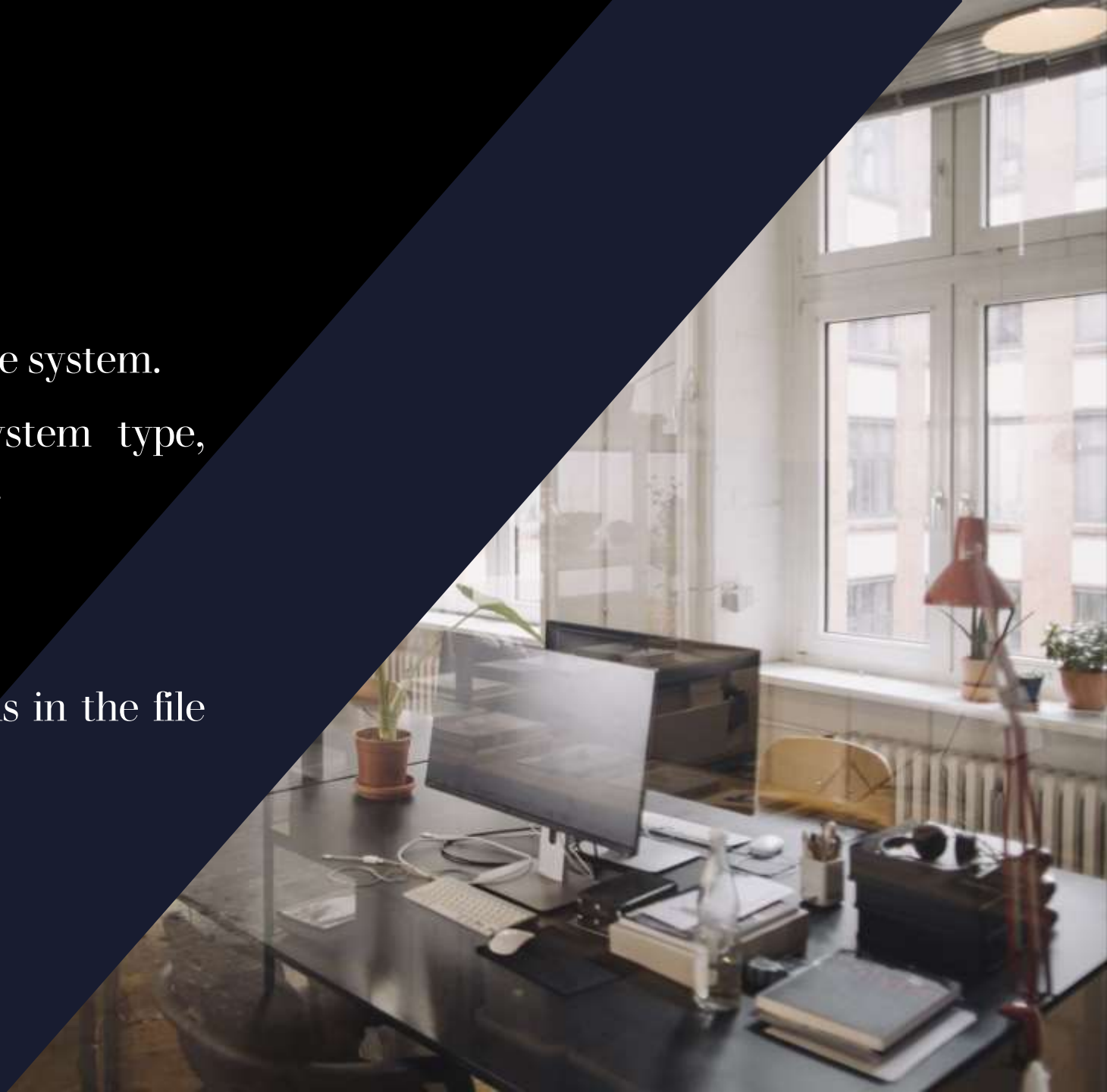
Parts of file system

Super Block

- It contains all the key parameters of the file system.
- It contains information such as file system type, number of blocks or size of file system, etc.

Free Space Management

- It keeps the information about free blocks in the file system.



Parts of file system

I-nodes

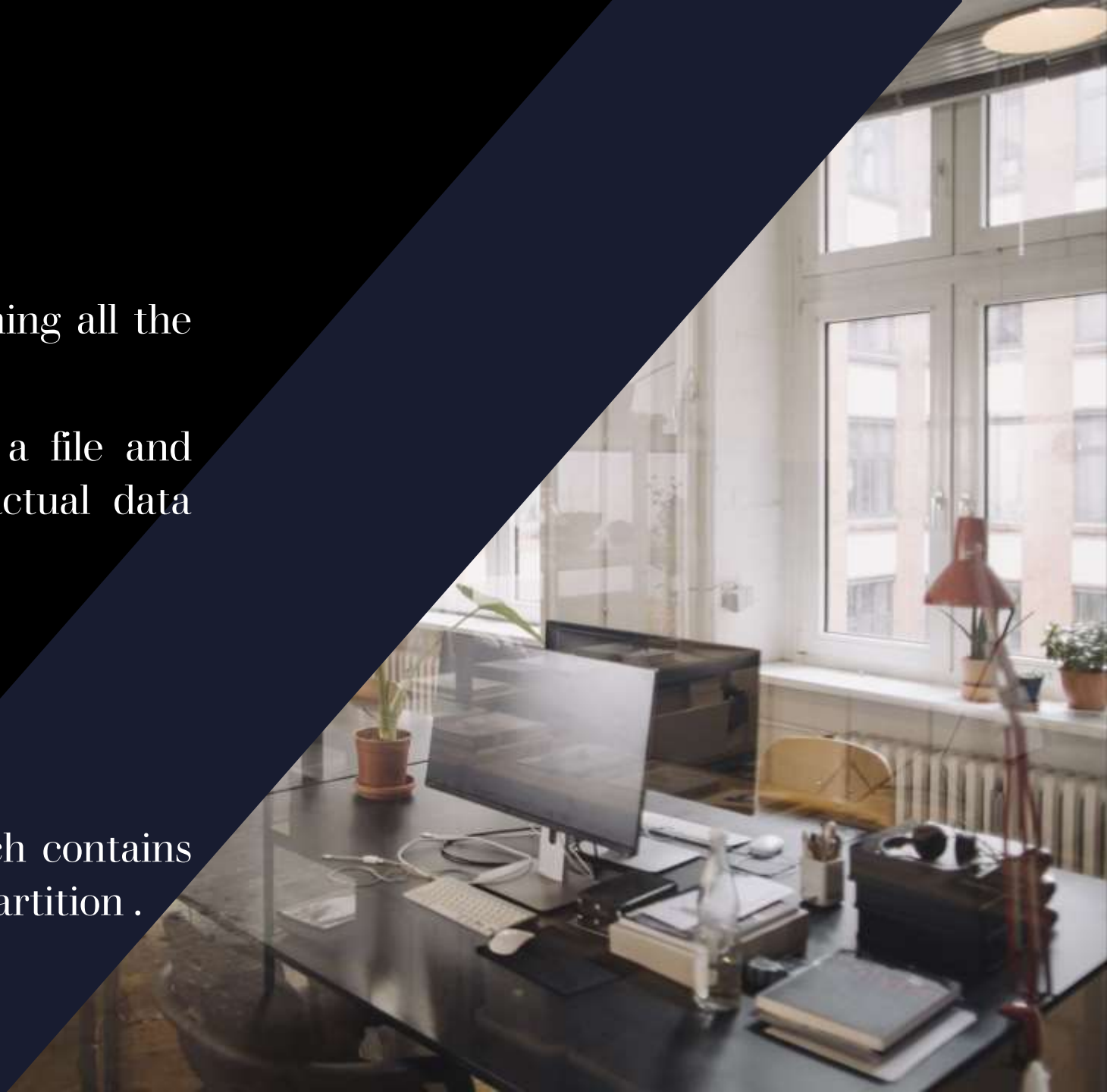
- An index-node is a data structure containing all the information about the file.
- It contains all the attributes related to a file and pointers to blocks, which contain the actual data contents of a file.

Root Directory

- It contains the top of the file system tree.

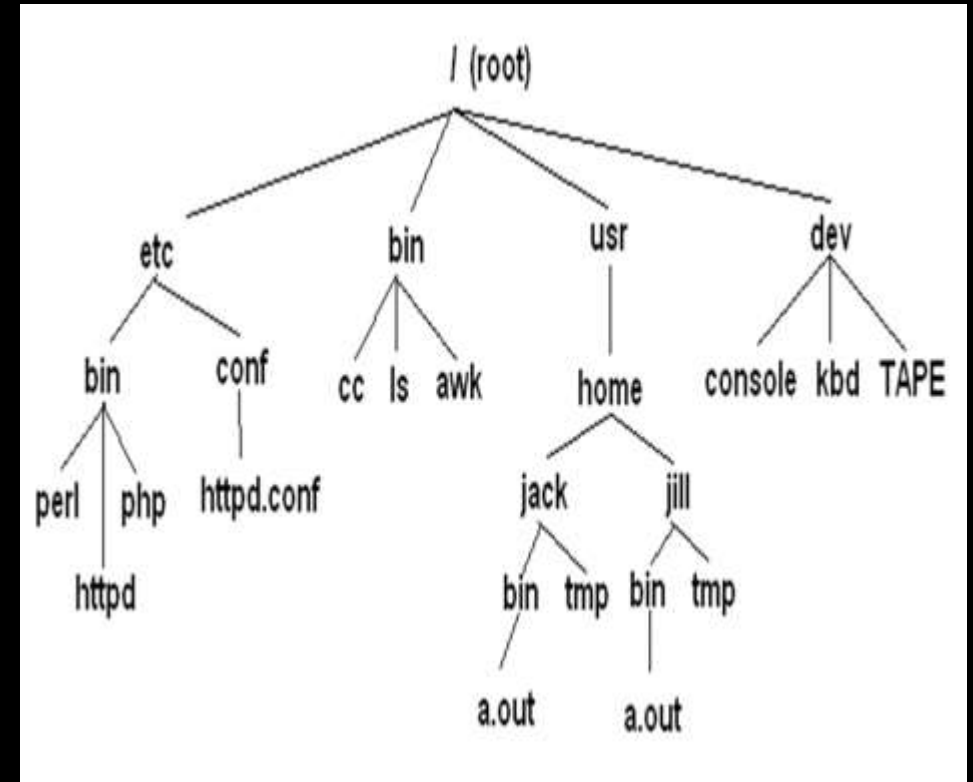
Files and Directories

- This is remaining part of a partition which contains all the directories and files stored on that partition .



Linux File System Features

- Linux uses a hierarchical directory structure like UNIX. Here all files are organized in an inverted tree like structure.
- The top of the hierarchy is called a 'root' directory or simply a 'root', denoted by "/".
- Every non-leaf node in a tree is a directory.
- Every leaf node is either a directory, regular file or a device file.



File Types

Regular (Ordinary) Files

These files are used to store user information.

When a file is referred in general, it belongs to this category and are used to store source programs, texts, pictures, sound, video, executable codes, etc.

They are further categorized in two parts: 1) Text files 2) Binary files

File Types

Directory Files

Directories are used to store files and sub-directories.

Directory files do not contain user information.

They keep some details of the files and sub-directories, such as name and unique identification number called I-node number.

They help in managing files in a well-organized manner.

File Types

Device Files

In Linux, various physical devices such as disks, printers, scanners, CD-ROMs, memory are considered as files also.

Kernel maintains special device files; one such file per device, to represent that device.

All operations related to that device are performed via reading or writing to such special files.

Such special files do not contain user data as do regular files.

File Path

Linux does not use the backslash (\) to separate the components; it uses forward slash (/) as an alternative.

For example, as in Windows, the data may be stored in C:\ My Documents\ Work, whereas, in Linux, it would be stored in /home/ My Document/ Work.

All the absolute path names start from a root ("/").

Partition, Directories, and Drives



Linux does not use drive letters to organize the drive as Windows does.

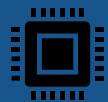


In Linux, we cannot tell whether we are addressing a partition, a network device, or an "ordinary" directory and a Drive.

Case Sensitivity



Linux file system is case sensitive. It distinguishes between lowercase and uppercase file names.



Such as, there is a difference between `test.txt` and `Test.txt` in Linux.



This rule is also applied for directories and Linux commands.

File Extensions



In Linux, a file may have the extension '.txt,' but it is not necessary that a file should have a file extension.

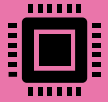


While working with Shell, it creates some problems for the beginners to differentiate between files and directories.



If we use the graphical file manager, it symbolizes the files and folders.

Hidden files



Linux distinguishes between standard files and hidden files, mostly the configuration files are hidden in Linux OS.



Usually, we don't need to access or read the hidden files. The hidden files in Linux are represented by a dot (.) before the file name (e.g., .ignore).



To access the files, we need to change the view in the file manager or need to use a specific command in the shell.

Types of Linux File System



The file system Ext stands for **Extended File System**.



It was primarily developed for **MINIX OS**.



The Ext file system is an older version and is no longer used due to some limitations.

Types of Linux File System



Ext2 is the first Linux file system that allows managing two terabytes of data.



Ext3 is developed through **Ext2**; it is an upgraded version of **Ext2** and contains backward compatibility.



The major drawback of **Ext3** is that it does not support servers because this file system does not support file recovery and disk snapshot.

Types of Linux File System



Ext4 file system is the faster file system among all the Ext file systems.



It is a very compatible option for the SSD (solid-state drive) disks, and it is the default file system in Linux distribution.

Types of Linux File System



JFS stands for **Journaled File System**, and it is developed by **IBM for AIX Unix**.



It is an alternative to the Ext file system.



It can also be used in place of Ext4, where stability is needed with few resources.



It is a handy file system when CPU power is limited.

Types of Linux File System



ReiserFS is an alternative to the Ext3 file system. It has improved performance and advanced features.



In the earlier time, the ReiserFS was used as the default file system in SUSE Linux, but later it has changed some policies, so SUSE returned to Ext3.



This file system dynamically supports the file extension, but it has some drawbacks in performance.

Types of Linux File System



XFS file system was considered as high-speed JFS, which is developed for parallel I/O processing.



NASA still using this file system with its high storage server (300+ Terabyte server).



ANY
QUESTIONS?

