File-System Interface

File System

A file is a named collection of related information that is recorded on secondary storage such as magnetic disks, magnetic tapes and optical disks. In general, a file is a sequence of bits, bytes, lines or records whose meaning is defined by the files creator and user.

File Structure

A File Structure should be according to a required format that the operating system can understand.

- A file has a certain defined structure according to its type.
- A text file is a sequence of characters organized into lines.
- A source file is a sequence of procedures and functions.
- An object file is a sequence of bytes organized into blocks that are understandable by the machine.

Attributes of a File

- **Name** . It is the only information which is in human-readable form.
- **Identifier**. The file is identified by a unique tag(number) within file system.
- **Type**. It is needed for systems that support different types of files.
- **Location**. Pointer to file location on device.
- **Size**. The current size of the file.
- **Protection**. This controls and assigns the power of reading, writing, executing.
- **Time, date, and user identification**. This is the data for protection, security, and usage monitoring.

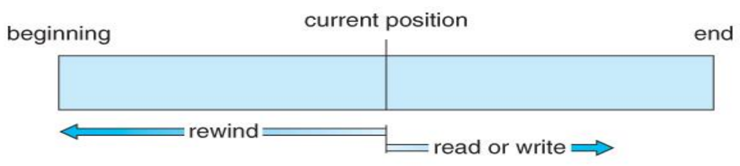
File Access Mechanisms

File access mechanism refers to the manner in which the records of a file may be accessed. There are several ways to access files

1. **Sequential access**
2. **Direct/Random access**
3. **Indexed sequential access**

Sequential access

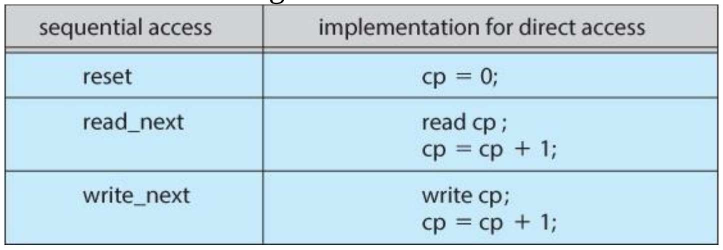
- A sequential access file emulates magnetic tape operation, and generally supports a few operations:
- **read next** - read a record and advance the tape to the next position.
- **write next** - write a record and advance the tape to the next position.



Example: Compilers usually access files in this fashion.

Direct/Random/relative access

- Random access file organization provides, accessing the records directly.
- Each record has its own address on the file with by the help of which it can be directly accessed for reading or writing.
- The records need not be in any sequence within the file and they need not be in adjacent locations on the storage medium.

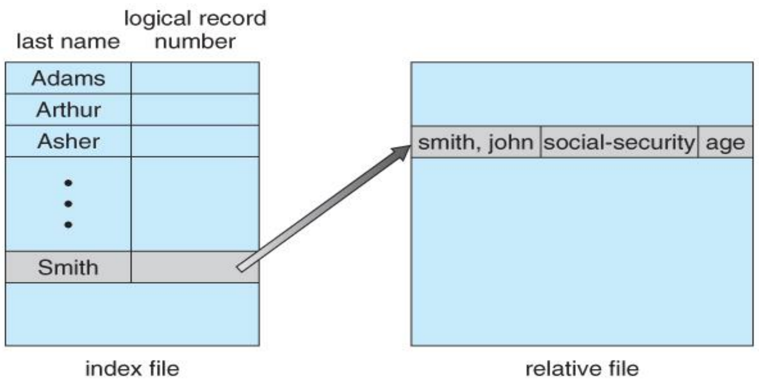


sequential access	implementation for direct access
reset	cp = 0;
read_next	read cp ; cp = cp + 1;

write_next	write cp;
	cp = cp + 1;

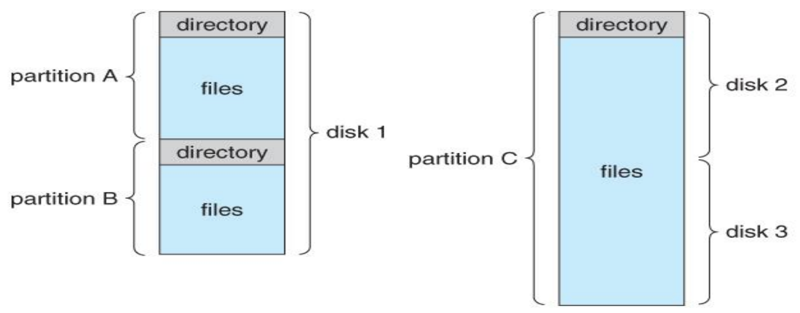
Indexed sequential access

- An index is created for each file which contains pointers to various blocks.
- Index is searched sequentially and its pointer is used to access the file directly.



Directory

Information about files is maintained by Directories. A directory can contain multiple files. It can even have directories inside of them. In Windows we also call these directories as folders.



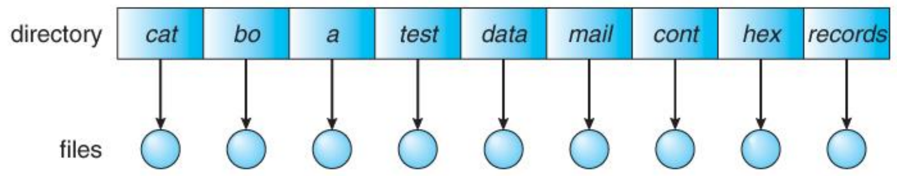
Following is the information maintained in a directory:

- **Name** : The name visible to user.
- **Type** : Type of the directory.
- **Location** : Device and location on the device where the file header is located.
- **Size** : Number of bytes/words/blocks in the file.
- **Position** : Current next-read/next-write pointers.
- **Protection** : Access control on read/write/execute/delete.
- **Usage** : Time of creation, access, modification etc.
- **Mounting** : When the root of one file system is "grafted" into the existing tree of another file system it's called Mounting.

Types of directory structures

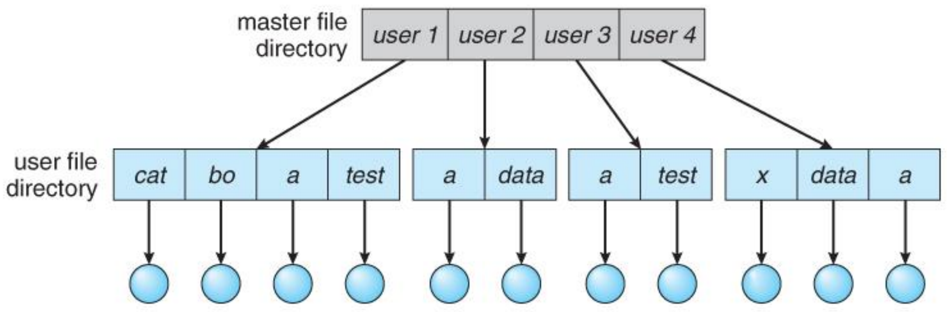
Single-Level Directory

- Simple to implement, but each file must have a unique name.



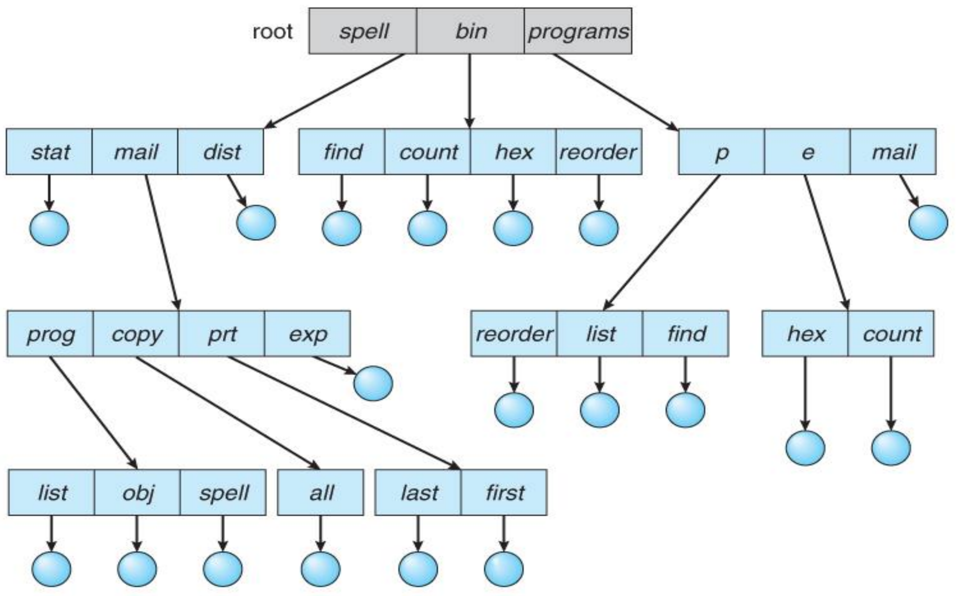
Two-Level Directory

- Each user gets their own directory space.
- File names only need to be unique within a given user's directory.
- A master file directory is used to keep track of each users directory, and must be maintained when users are added to or removed from the system.
- A separate directory is generally needed for system (executable) files.
- Systems may or may not allow users to access other directories besides their own
- If access to other directories is allowed, then provision must be made to specify the directory being accessed.
- If access is denied, then special consideration must be made for users to run programs located in system directories. A *search path* is the list of directories in which to search for executable programs, and can be set uniquely for each user.



Tree-Structured Directories

- An obvious extension to the two-tiered directory structure, and the one with which we are all most familiar.
- Each user / process has the concept of a *current directory* from which all (relative) searches take place.
- Files may be accessed using either absolute pathnames (relative to the root of the tree) or relative pathnames (relative to the current directory.)
- Directories are stored the same as any other file in the system, except there is a bit that identifies them as directories, and they have some special structure that the OS understands.
- One question for consideration is whether or not to allow the removal of directories that are not empty - Windows requires that directories be emptied first, and UNIX provides an option for deleting entire sub-trees.



Path names:

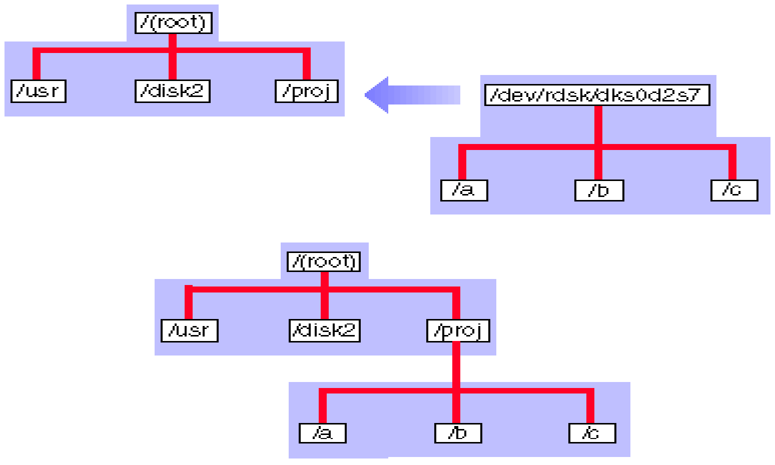
An absolute or full path points to the same location in a file system, regardless of the current working directory. To do that, it must include the root directory.

By contrast, **A relative path** starts from some given working directory, avoiding the need to provide the full absolute path.

Mounting File Systems

File systems must be ***mounted*** to be used.
When a file system is mounted, the name of the device file for the file system (/dev/rdsk/dks0d2s7) and the name of a directory (/proj) are given.
The directory, /proj,is called a ***mount point*** and forms the connection between the file system containing the mount point and the file system to be mounted. Mounting a file system tells the kernel that the mount point is to be considered equivalent to the top level directory of the file system when pathnames are resolved.

In the below figure the files a, b, and c in the /dev/rdsk/dks0d2s7 file system become /proj/a, /proj/b, and /proj/c as shown in the bottom of the figure.



File Sharing

File sharing, also known as **file-swapping** is the accessing or sharing of files by one or more users. It is performed on computer networks as a quick way to transmit data. Generally, a file-sharing system usually has more than one administrator, where the users may have the same or different access privileges. It also implies having an allocated amount of personal files in the common storage.

File sharing has been used in mainframe and multi-user computer systems for many years, and now with widespread access to the internet, a file transfer system known as the **File-Transfer Protocol** or FTP is widely used.