Introduction to Files

Prof. Darrell Long

CSE 13S

Long-term Information Storage

- We want to store large amounts of data.
  - The definition of large has changed by 6 orders of magnitude over the years.
- Information stored must survive the termination of the process using it.
  - Memory (DRAM) does not survive turning the computer off.
- Files are accessed using *names*, memory is accessed using *addresses*.

File Names

"In the once-upon-a-time days of the First Age of Magic, the prudent sorcerer regarded his own true name as his most valued possession but also the greatest threat to his continued good health, for—the stories go—once an enemy, even a weak unskilled enemy, learned the sorcerer's true name, then routine and widely known spells could destroy or enslave even the most powerful."

—Vernor Vinge, *True Names*

- Your computer has millions of files.
  - A data center has *billions* of files.
  - The Internet has *trillions* of files.
- How do you uniquely specify which file you want?
- How do you find that file among the millions?

File Naming

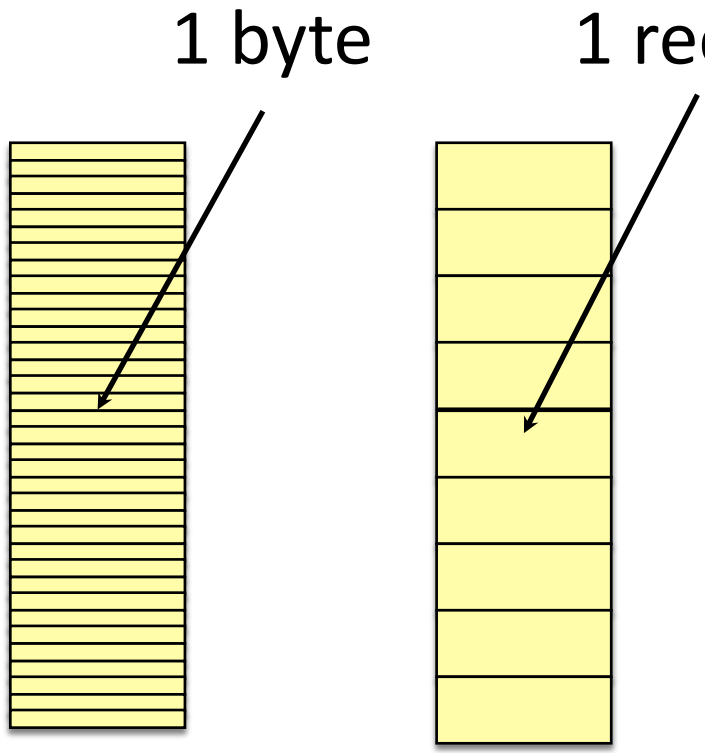
Files have base names and extensions.

In some operating systems, the extension is a separate entity.

In UNIX, the extension is only used by *convention*.

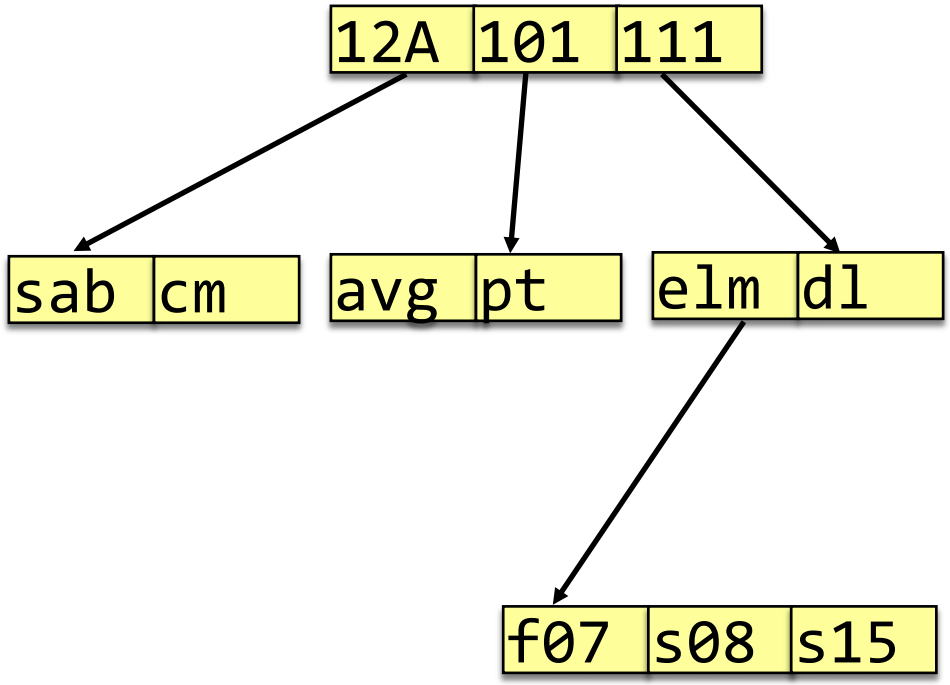
Extension	Meaning
file.bak	Backup file
file.c	C source program
file.gif	Compuserve Graphical Interchange Format image
file.hlp	Help file
file.html	World Wide Web HyperText Markup Language document
file.jpg	Still picture encoded with the JPEG standard
file.mp3	Music encoded in MPEG layer 3 audio format
file.mpg	Movie encoded with the MPEG standard
file.o	Object file (compiler output, not yet linked)
file.pdf	Portable Document Format file
file.ps	PostScript file
file.tex	Input for the TEX formatting program
file.txt	General text file
file.zip	Compressed archive

File Structure



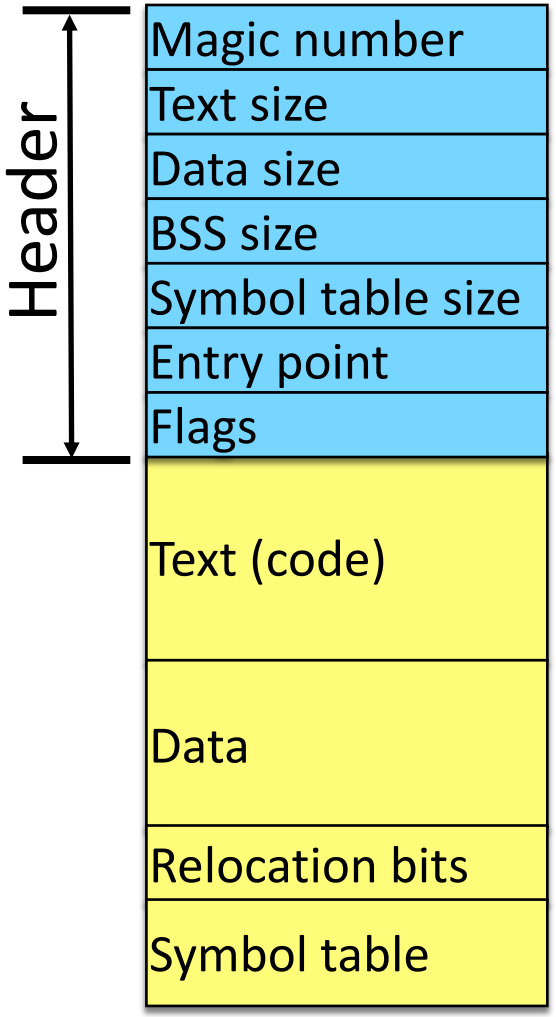
Sequence of bytes

Sequence of records

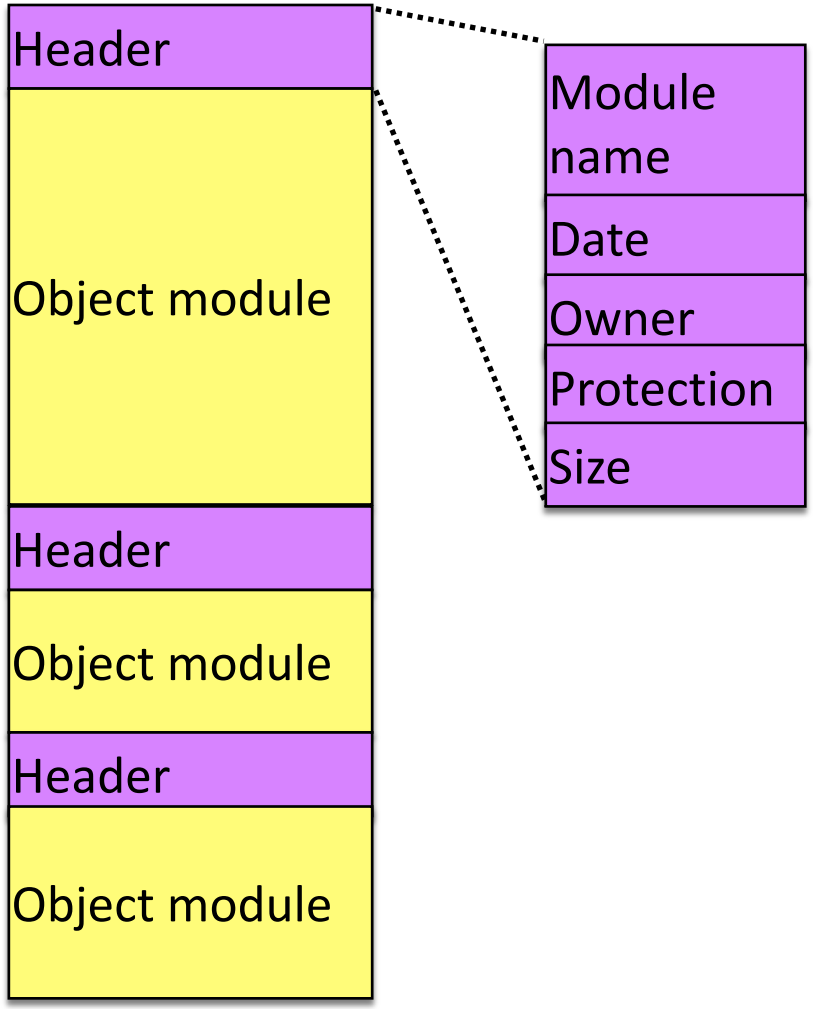


Tree

Files may have internal structure



Executable file



Archive file

File Access

Sequential access

- Read all bytes/records from the beginning
- Cannot jump around, may rewind or back up
- Convenient when medium was magnetic tape

Random access

- Bytes/records read in any order
- Essential for data base systems
- Read can be:
  - Move file marker (seek), then read or …
  - Read and then move file marker

File Attributes

Possible file attributes

Attribute	Meaning
Protection	Who can access the file and in what way
Password	Password needed to access the file
Creator	ID of the person who created the file
Owner	Current owner
Read-only flag	0 for read/write; 1 for read only
Hidden flag	0 for normal; 1 for do not display in listings
System flag	0 for normal files; 1 for system file
Archive flag	0 for has been backed up; 1 for needs to be backed up
ASCII/binary flag	0 for ASCII file; 1 for binary file
Random access flag	0 for sequential access only; 1 for random access
Temporary flag	0 for normal; 1 for delete file on process exit
Lock flags	0 for unlocked; nonzero for locked
Record length	Number of bytes in a record
Key position	Offset of the key within each record
Key length	Number of bytes in the key field
Creation time	Date and time the file was created
Time of last access	Date and time the file was last accessed
Time of last change	Date and time the file has last changed
Current size	Number of bytes in the file
Maximum size	Number of bytes the file may grow to

File Operations

- Create
- Delete
- Open
- Close
- Read
- Write

- Append
- Seek
- Get attributes
- Set Attributes
- Rename

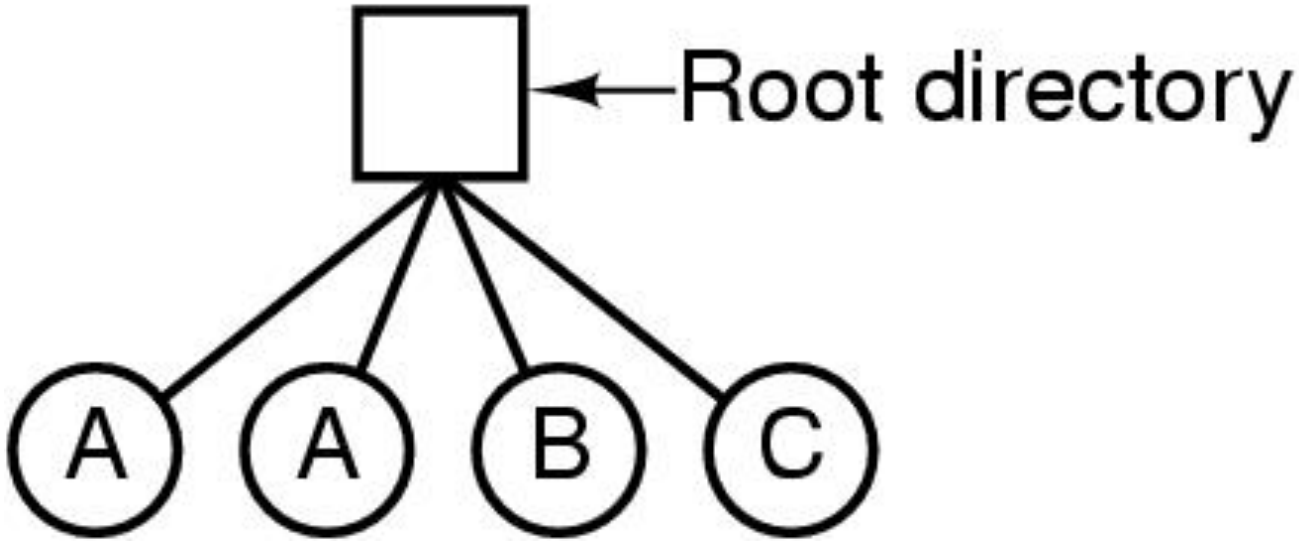
Directories

- Naming is better than using numbers, but still limited.
- Humans like to group things together for convenience.
  - We use hierarchy to manage complexity.
- File systems allow this to be done with directories
  - You may call them folders.
- Grouping makes it easier to:
  - Find files in the first place: remember the enclosing directories for the file,
  - Locate related files,
  - Determine which files are related.

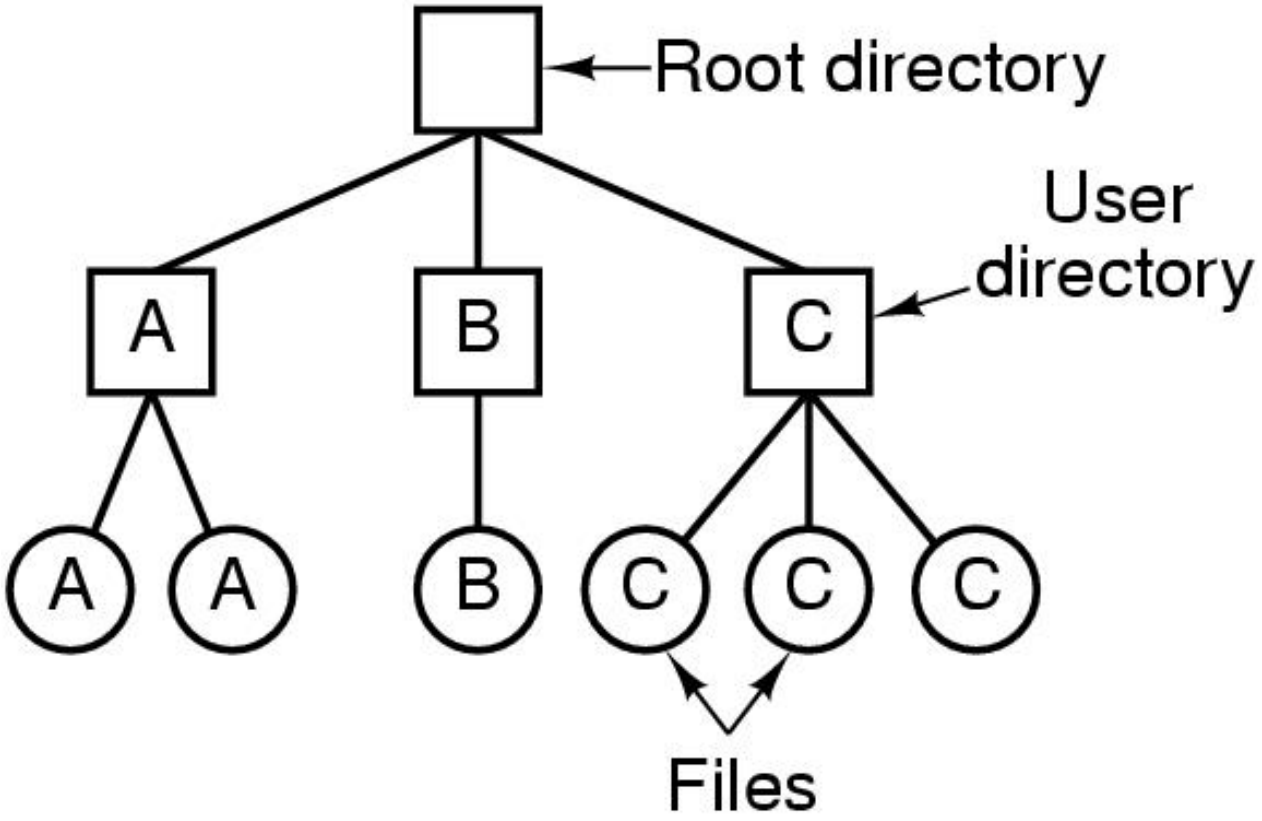
Directories
Single-Level Directory Systems

A single level directory system:

- Contains 4 files
- Owned by 3 different people, A, B, and C

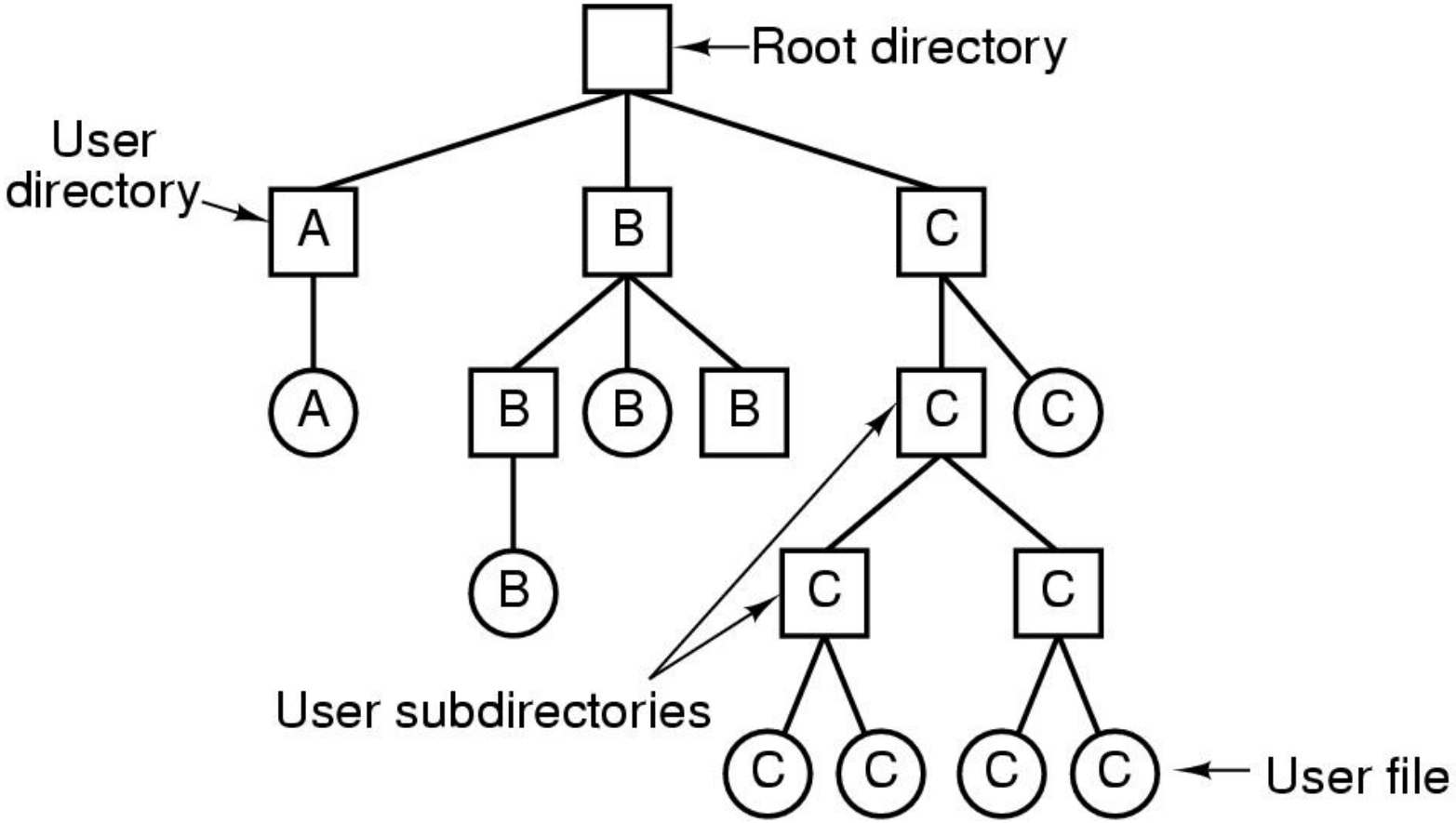


Two-level Directory Systems



Letters indicate *owners* of the directories and files

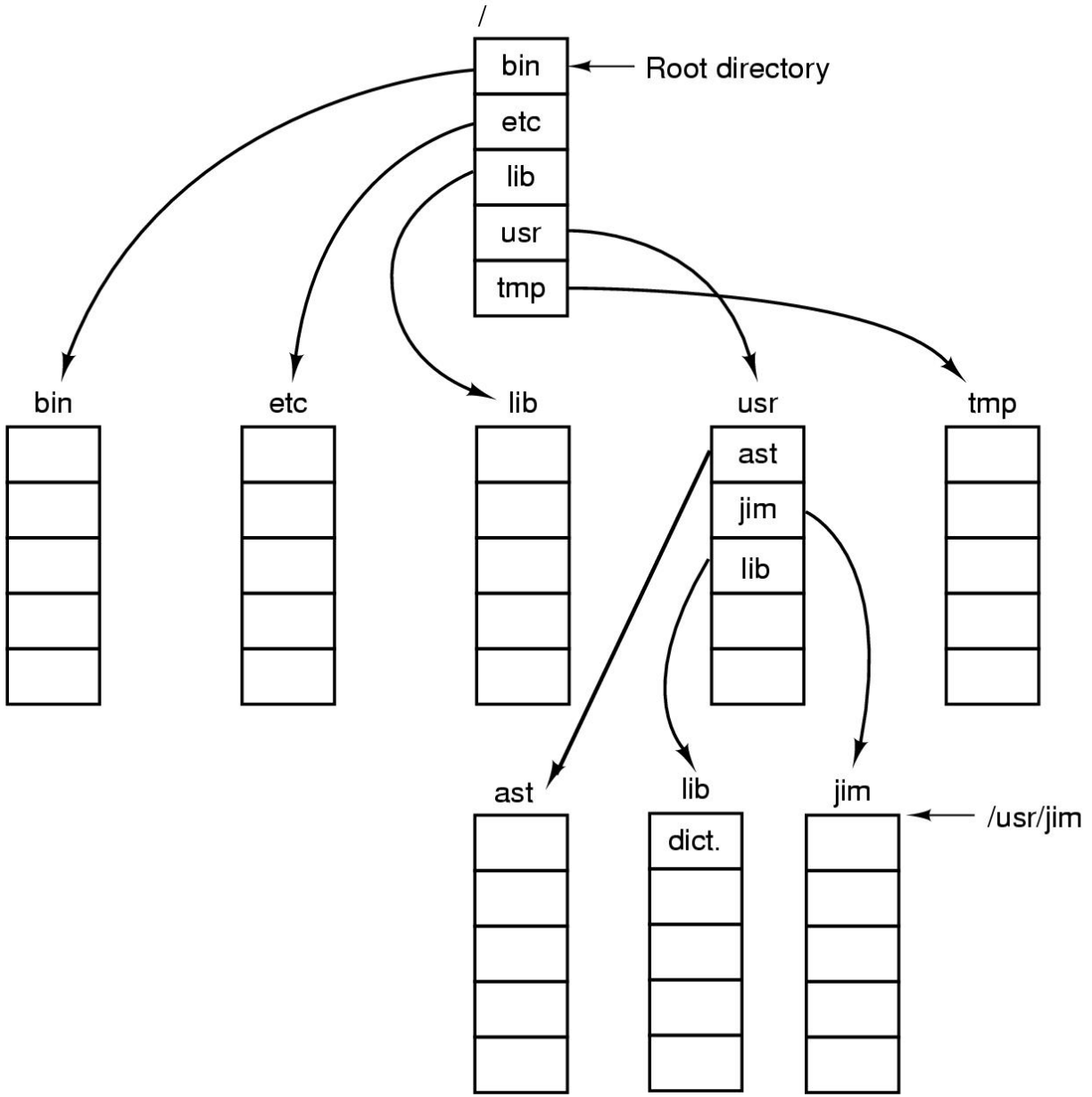
Hierarchical Directory Systems



A hierarchical directory system

Path Names

A UNIX directory tree

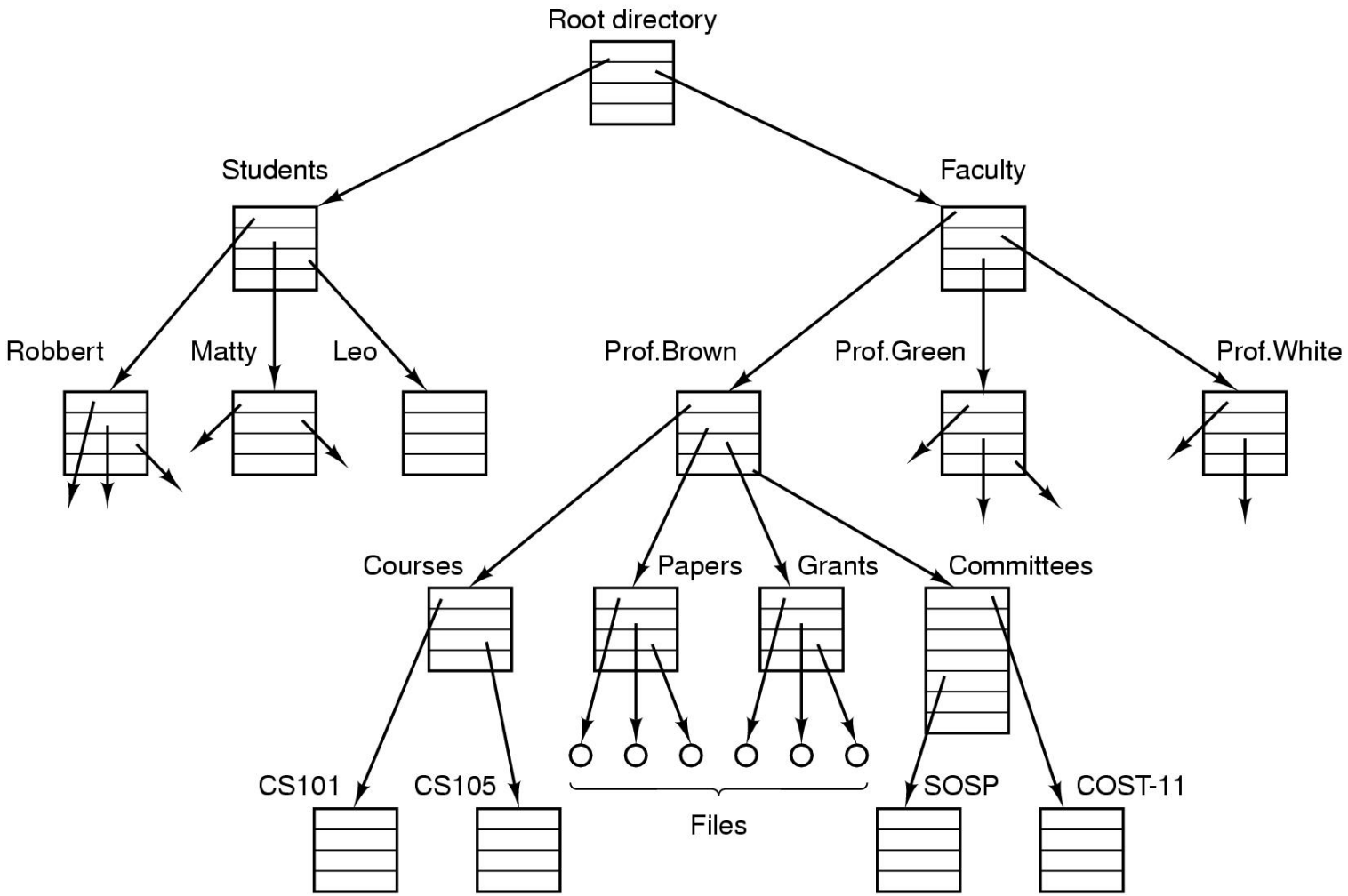


Directory Operations

- Create
- Delete
- Opendir
- Closedir

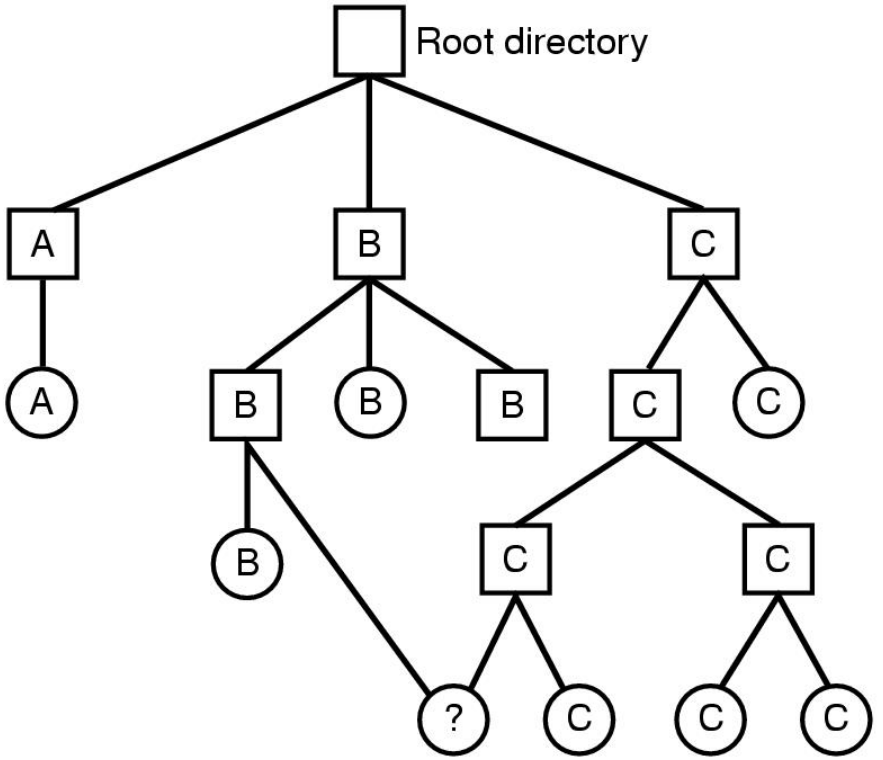
- Readdir
- Rename
- Link
- Unlink

Hierarchical File System



Shared Files

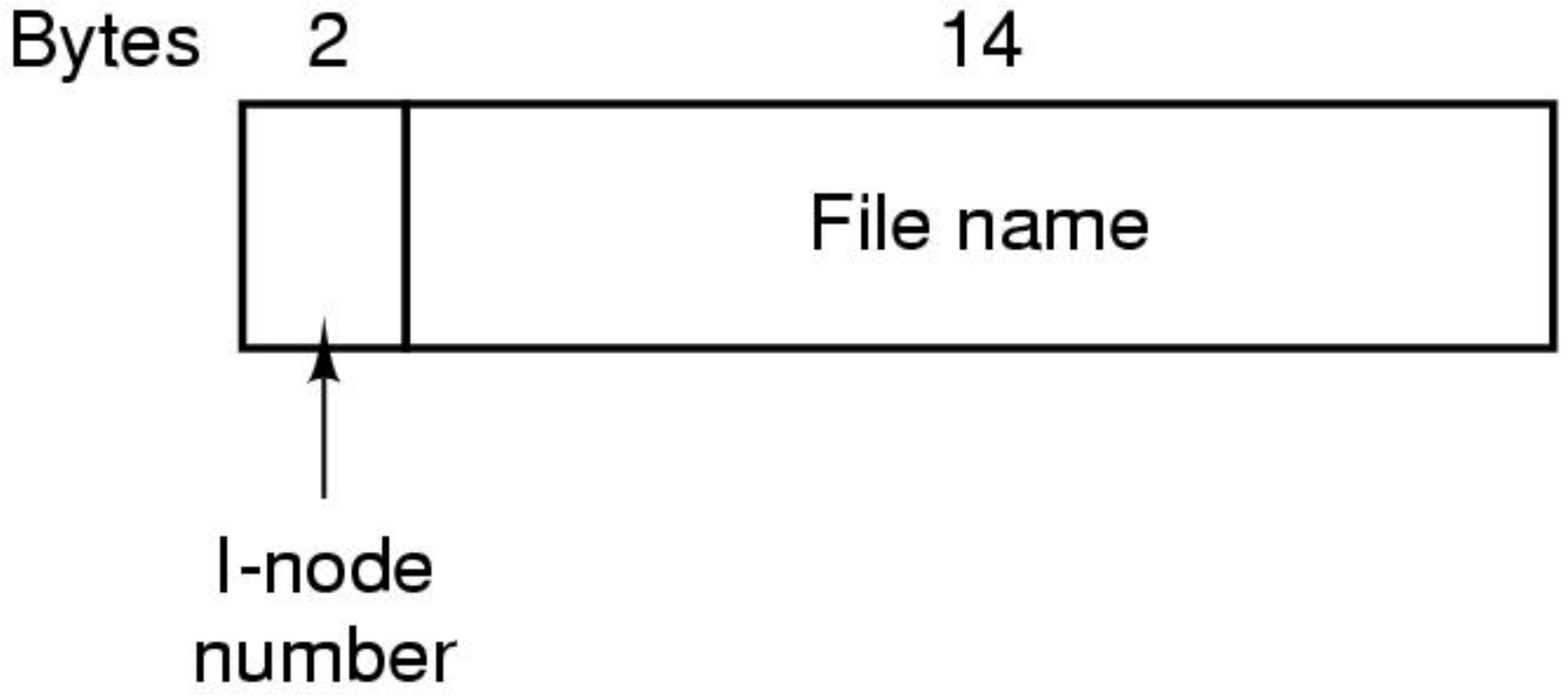
File system containing a shared file



Shared file

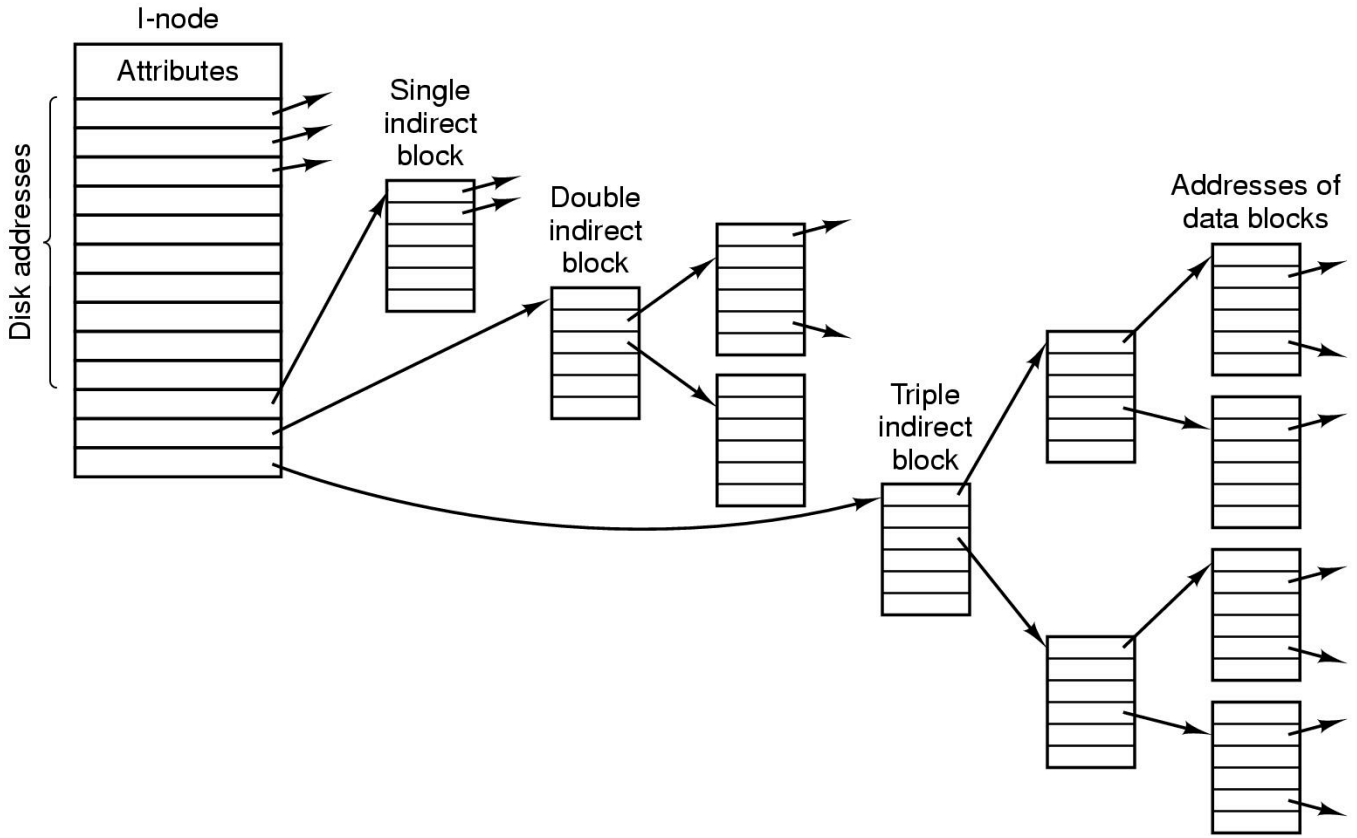
The UNIX V7 File System

A UNIX V7 directory entry



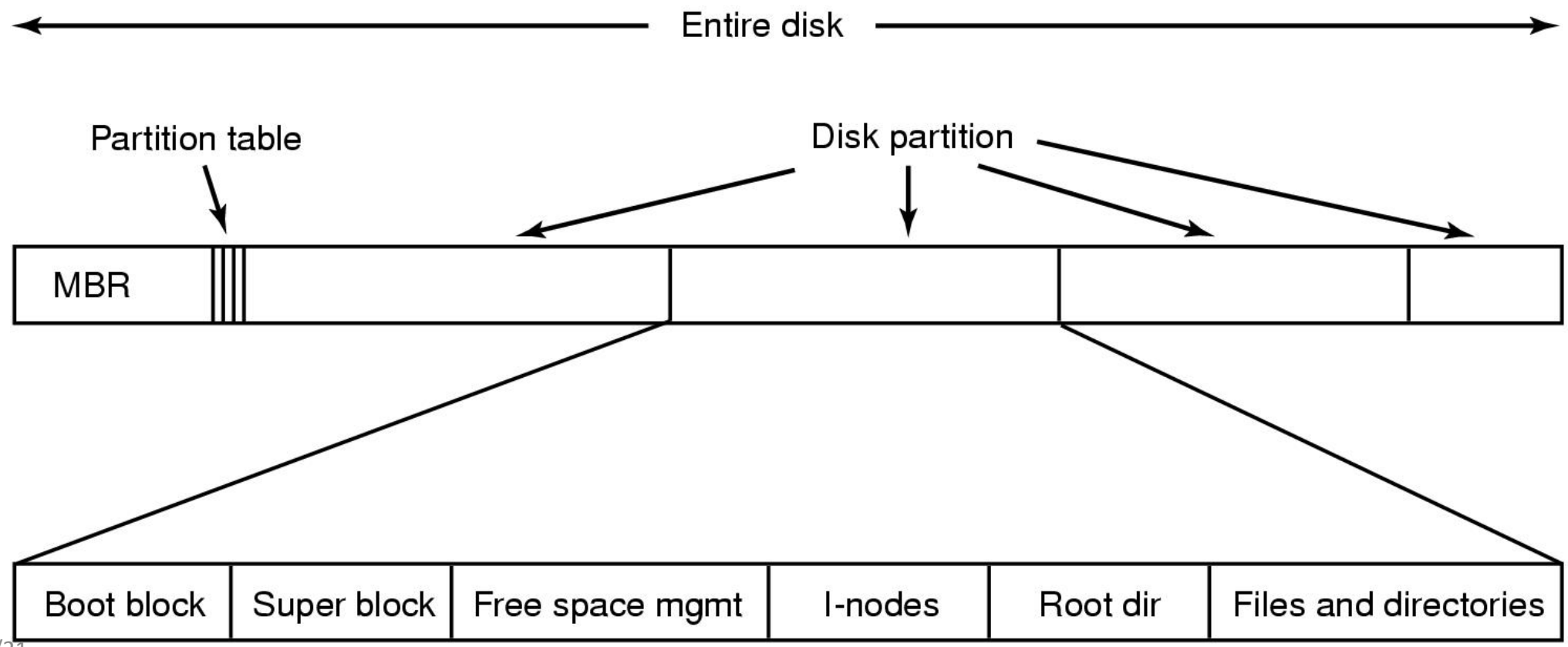
The UNIX V7 File System

A UNIX i-node



File System Implementation

A possible file system layout



cat.c

- Copies standard input to standard output, or …
- Sequentially copies multiple files to standard output.
- We will discuss this in more detail later.

darrell — vi cat.c — 80×32

```c
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/errno.h>
#include <unistd.h>

#define BUFFER_SIZE 1024

int main(int argc, char **argv) {
  int count;
  unsigned char buffer[BUFFER_SIZE];

  if (argc < 2) {
    while ((count = read(0, buffer, BUFFER_SIZE)) > 0)
      write(1, buffer, count);
  } else {
    int i;
    for (i = 1; i < argc; i++) {
      int file;
      file = open(argv[i], O_RDONLY);
      if (file < 0) {
        perror("cat");
        exit(errno);
      }
      while ((count = read(file, buffer, BUFFER_SIZE)) > 0)
        write(1, buffer, count);
      close(file);
    }
  }
  exit(0);
}
```