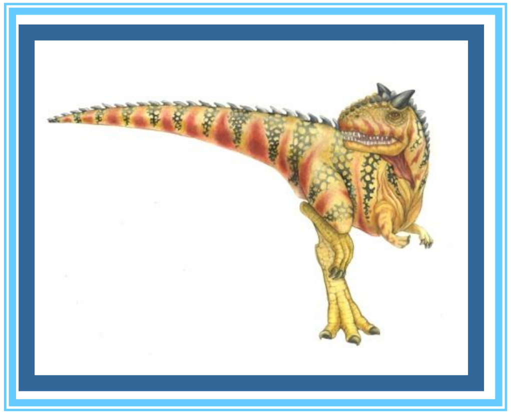
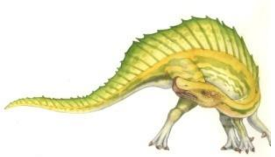
Chapter 10: File-System Interface



Chapter 10: File-System Interface

- File Concept
- Access Methods
- Directory Structure
- File-System Mounting
- File Sharing
- Protection

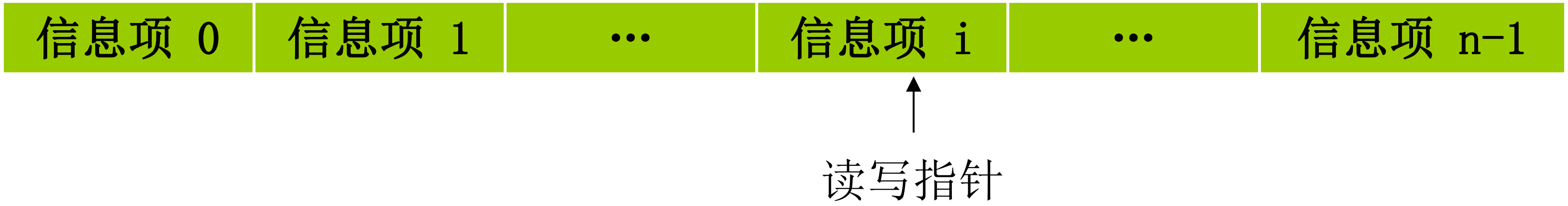


Objectives

- To explain the function of file systems
- To describe the interfaces to file systems
- To discuss file-system design tradeoffs, including access methods, file sharing, file locking, and directory structures
- To explore file-system protection

文件是什么？

- 文件 是 对磁盘的 抽象
- 所谓 文件 是指 一组带标识（标识即文件名）的、在逻辑上有完整意义的信息项的序列
- 信息项：构成文件内容的基本单位（单个字节或多个字节），各信息项之间具有顺序关系
- 文件内容的意义：由文件建立者和使用者解释



文件系统的需求分析

操作系统角度

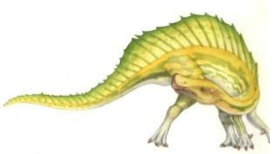
怎么组织、管理文件?

- 文件的描述、分类
- 文件目录的实现
- 存储空间的管理
- 文件的物理地址
- 文件系统效率与性能

用户角度

文件系统如何呈现在用户面前:

- 一个文件的组织
- 如何命名?
- 如何保护文件?
- 可以实施的操作?
- …



What is File System？

- 文件 是 信息项的序列
- 文件系统是操作系统中统一管理信息资源的一种软件，管理文件的存储、检索、更新，提供安全可靠的共享和保护手段，并且方便用户使用。

● 统一管理磁盘空间，实施磁盘空间的分配与回收

● 实现文件的按名存取

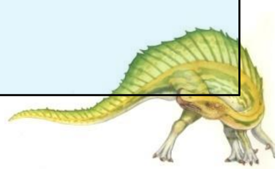
名字空间 ——→映射 磁盘空间

●实现文件信息的共享，并提供文件的保护、保密手段

●向用户提供一个方便使用、易于维护的接口，并向用户提供有关统计信息

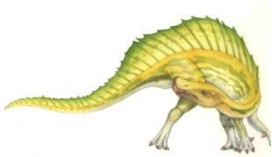
●提高文件系统性能

●提供关于I/O系统的统一接口



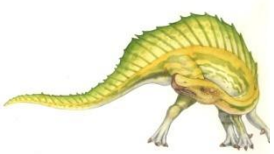
File Concept

- A file is a named collection of related information that is recorded on secondary storage.
  - A file is the smallest unit of allocation to the secondary storage that can be seen by a user.
  - Information in a file is defined by its creator.
  - Information about files are often maintained in the directory structure.
- A file system is a collection of files in an organized way, with proper storage and directory structure.
- Types:
  - Data
    - numeric
    - character
    - binary
  - Program



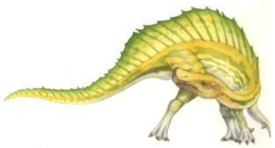
File Types

- 按文件性质和用途分类 （UNIX）
  - 普通文件
    - 用户建立和使用的文件，用户委托文件系统保存的文件。如源程序，目标程序，原始数据等。
  - 目录文件
    - 管理文件系统的系统文件
  - 特殊文件（设备文件）
    - 字符设备文件：和输入输出有关，用于模仿串行I/O设备，例如终端，打印机，网卡等
    - 块设备文件：磁盘



File Types

- 按信息保存期限分类：
  - 临时文件，永久文件，档案文件
- 按文件的保护方式分类：
  - 只读文件；读写文件；只执行文件
- 按信息流向分类：
  - 输入文件，输出文件，输入/输出文件（存储设备）
- 按文件中的数据分类：
  - 源文件；目标文件；可执行文件

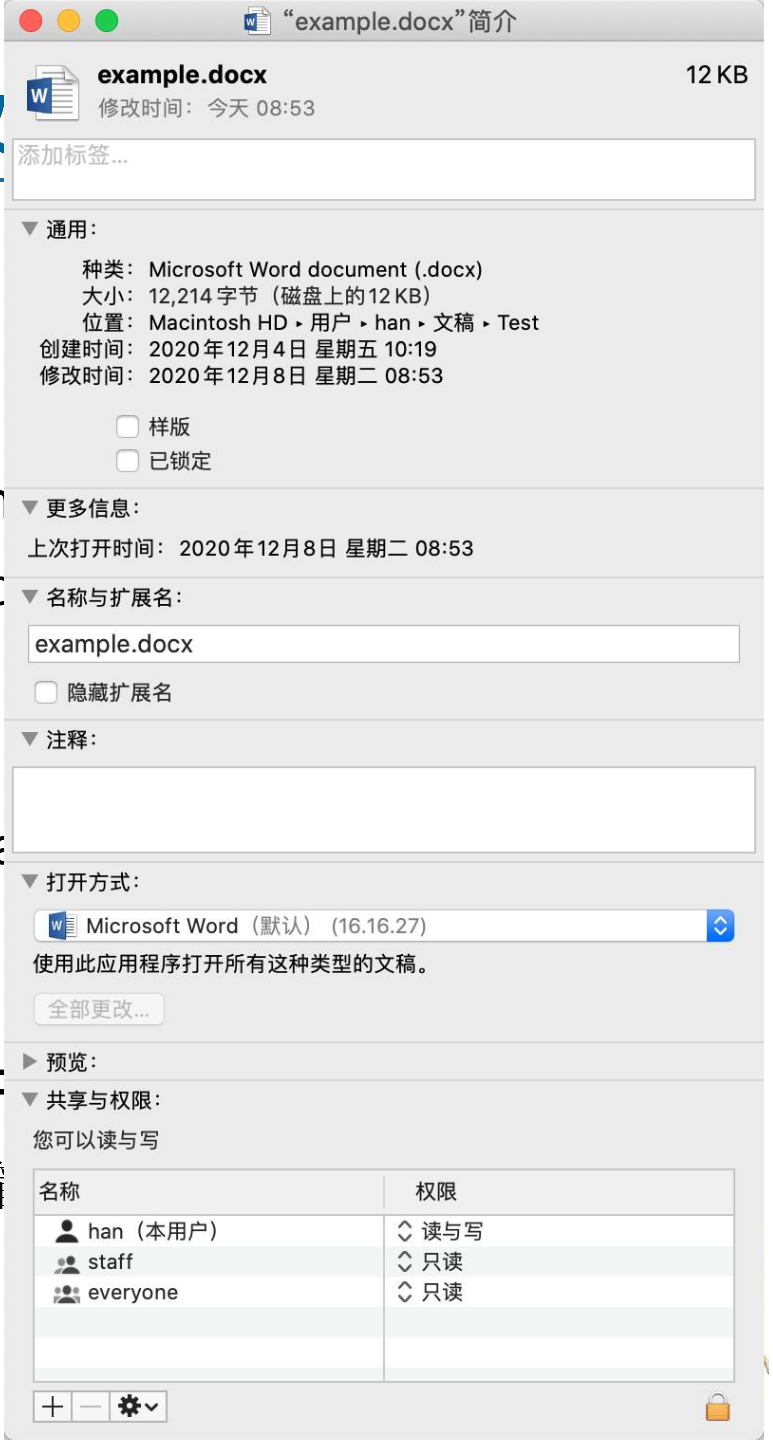


File Types

- 按文件的逻辑结构分类
  - 从用户角度看文件，由用户的访问方式确定
  - 流式文件；记录式文件
- 按文件的物理结构分类：
  - 顺序（连续）文件；链接文件；索引文件

File Attributes（文

- Information about files are kept in which is maintained on the disk.
  - **Name** – only information kept in hum
  - **Identifier** – unique tag (number) iden
  - **Type** – needed for systems that supp
  - **Location** – pointer to file location on
  - **Size** – current file size
  - **Protection** – controls who can do rea
  - **Time, date, and user identification** and usage monitoring
- File Control Block (文件控制块， F
  - 为管理文件而设置的数据结构，保存 文件属性或元数据）



“example.docx”简介

example.docx 12 KB

修改时间：今天 08:53

添加标签...

▼ 通用：

种类：Microsoft Word document (.docx)
大小：12,214 字节（磁盘上的12 KB）
位置：Macintosh HD ▸ 用户 ▸ han ▸ 文稿 ▸ Test
创建时间：2020年12月4日 星期五 10:19
修改时间：2020年12月8日 星期二 08:53

□ 样版
□ 已锁定

▼ 更多信息：

上次打开时间：2020年12月8日 星期二 08:53

▼ 名称与扩展名：

example.docx

□ 隐藏扩展名

▼ 注释：

▼ 打开方式：

Microsoft Word（默认）(16.16.27)

使用此应用程序打开所有这种类型的文稿。

全部更改...

▶ 预览：

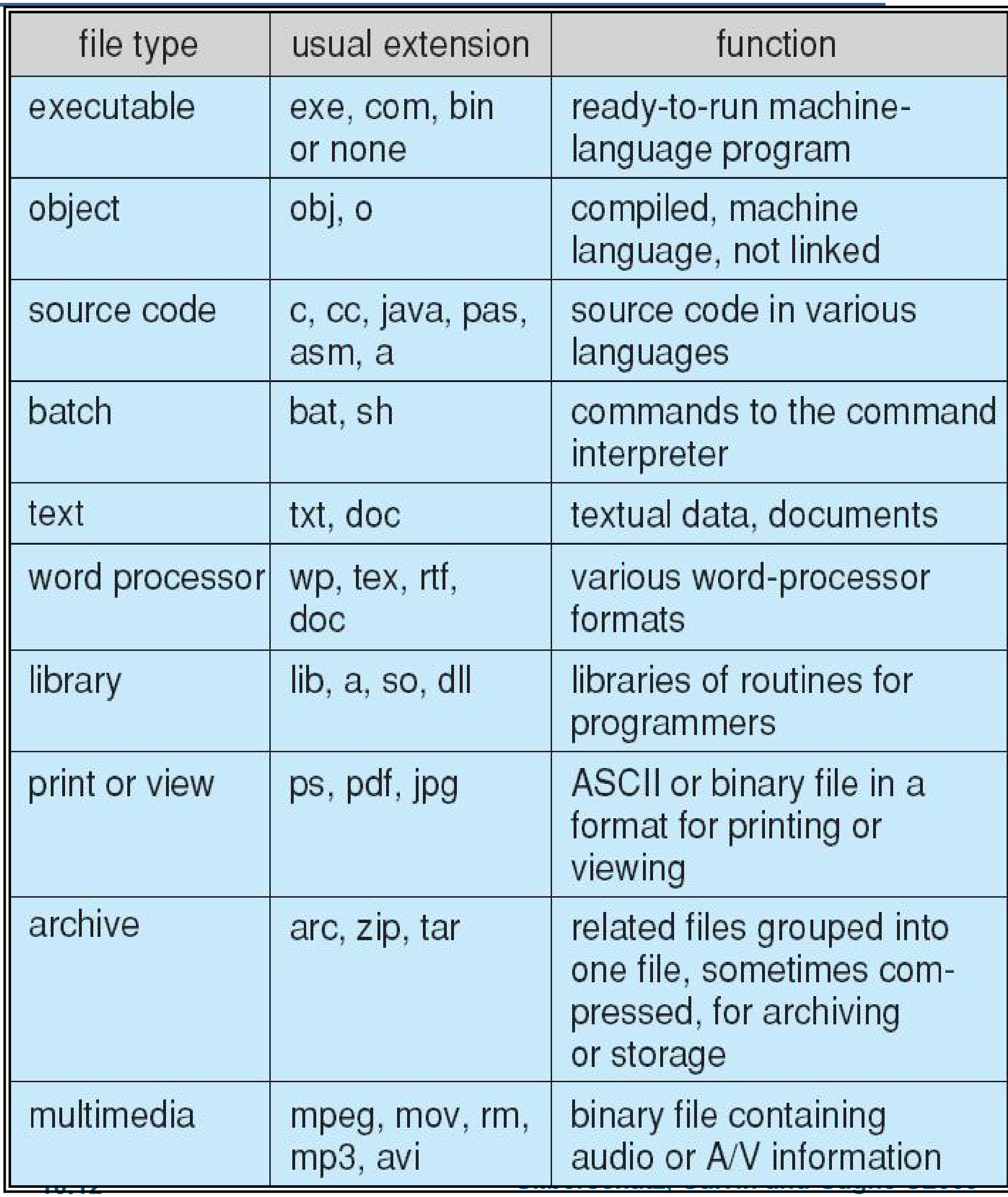
▼ 共享与权限：

您可以读与写

名称	权限
han（本用户）	读与写
staff	只读
everyone	只读

File Types – Name, Extension

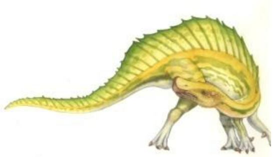
- Two major types of files:
  - Program files
    - Source code
    - Object code
    - Executable program
  - Data files
    - Character or ASCII file
    - Binary file
- There are different subtypes of files.
  - They are often indicated by the file extension.



file type	usual extension	function
executable	exe, com, bin or none	ready-to-run machine-language program
object	obj, o	compiled, machine language, not linked
source code	c, cc, java, pas, asm, a	source code in various languages
batch	bat, sh	commands to the command interpreter
text	txt, doc	textual data, documents
word processor	wp, tex, rtf, doc	various word-processor formats
library	lib, a, so, dll	libraries of routines for programmers
print or view	ps, pdf, jpg	ASCII or binary file in a format for printing or viewing
archive	arc, zip, tar	related files grouped into one file, sometimes compressed, for archiving or storage
multimedia	mpeg, mov, rm, mp3, avi	binary file containing audio or A/V information

File Operations

- File is an abstract data type
- Basic file operations
  - Create
  - Write
  - Read
  - Reposition within file (Seek)
  - Delete
  - Truncate(截断)
  - Get Attributes
  - Set Attributes
  - Rename
  - Copy
  - …



File Operations

- **Create**
  - 建立系统与文件的联系，实质是建立文件的FCB
  - Makes an entry in the file directory
  - Allocates disk space

Create(文件名，访问权限)

①检查参数的合法性

例如：文件名是是否符合命名规则；

有无重名文件；

合②->　　　　；　否则->报错、返回

②申请空闲FCB，并填写相关内容

③为文件申请磁盘块

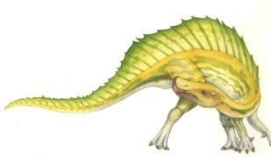
④返回

File Operations

- **Delete**
  - Searches the directory
  - Releases all file space
  - Erases the directory entry for the file

File Operations

- **Read**
  - requires specifying file name and the memory location where the next block of the file to be put
  - Searches the directory to find the file’s location
  - Reads the block of the file into memory
  - Updates the read pointer



File Operations

- **Read（文件描述符，读指针，要读的长度，内存目的地址）**

① 根据打开文件时得到的文件描述符，找到相应的FCB

确定读操作的合法性

读操作合法②->　　　；　否则->出错处理

② 将文件的逻辑块号转换为物理块号

根据参数中的读指针、长度与文件控制块中的信息，确定块号、块数、块内位移

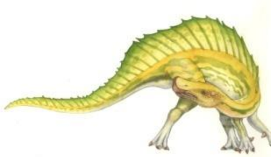
③ 申请缓冲区

④ 启动磁盘I/O操作，把磁盘块中的信息读入缓冲区，再传送到指定的内存区（多次读盘）

⑤ 反复执行3、4直至读出所需数量的数据

File Operations

- **Write**
  - requires specifying file name and the information to be written to the file
  - Searches the directory to find the file's location
  - Writes to the file according to the write pointer to the location in the file
  - Updates the writer pointer



File Operations

- **Repositioning within a file**
  - Known as a file seek
  - Searches the directory to find the file’s entry
  - Updates the current-file-position pointer to a given value
- **Turncate 截断**
  - Erases the contents of a file but keeps its attributes
    - Only the file length be reset to 0
  - Releases its file space

用基本文件操作构造copy操作

```
 /*File copy program. Error checking and reporting is minimal.*/
int main(int argc, chaar *argv[])
{
 /*Open the input file and create the output file */
 in_fd = open(argv[1], O_RDONLY);  /*open the source file */
 if(in_fd <0) exit(2);  /*if it cannot be opened, exit */
 out_fd = create(argv[2], OUTPUT_MODE) ;  /*create the destination file*/
 if(in_ out <0) exit(3);  /*if it cannot be opened, exit */
```

用基本文件操作构造copy操作(续)

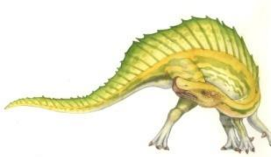
```
 /*Copy loop*/
while(TRUE){
    rd_count = read(in_fd, buffer, BUF_SIZE); /*read a block of data */
if(rd_count <=0) break; /*if end of file or error, exit loop*/
    wt_count = write(out_fd, buffer, rf_count) /*write data*/
    if (wt_count <=0) exit(4);   /* wt_count <=0 is an error*/
}
/*Close the files*/
close(in_fd);
close(out_fd);
if (rf_cout ==0) exit(0); );   /* no error on last read*/
Else
exit(5);
}
```

File Operations

- Besides the basic operations, two more common operations are often provided in the implementation of most file systems.
- *Open(F_i)* :make a file ready for reading/writing.
  - searches the directory structure on disk for entry F_i
  - moves the content of entry to memory
  - 根据文件名在文件目录中检索，并将该文件的目录项读入内存，建立相应的数据结构（系统打开文件表、用户打开文件表），为后续的文件操作做好准备。返回文件描述符/文件句柄
- *Close (F_i)* : mark the completion of operation on file.
  - moves the content of entry F_i in memory to directory structure on disk

Open Files

- Most of the file operations involve searching the directory for the entry of the named file.
- To avoid this constant searching, many OS require that an open() system call be made before a file first used actively.
- The OS keeps a small table, called the open-file table, containing information about all open files.
- The open() takes a file name and searches the directory, copying the directory entry into the open-file table.
- The open() system call typically returns a pointer to the entry in the open-file table.

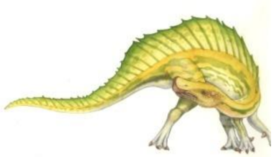


Open Files

- There may be several processes opening the same file at the same time.
- OS uses two levels of internal tables:
  - Process open-file table
    - Tracks all files opened by a process
    - Stores information about the use of the file, e.g. current file pointer, access rights and accounting information.
    - Each entry in process open-file table in turn points to a system-wide open-file table.
  - System open-file table
    - Contains process-independent information, such as the disk location of the file, access date, file size and open count.

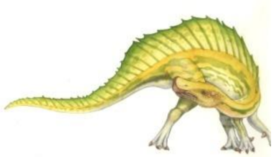
Open Files

- Several pieces of data are needed to manage open files:
  - File pointer: pointer to last read/write location, per process that has the file open
  - File-open count: counter of number of times a file is open – to allow removal of data from open-file table when last processes closes it
  - Disk location of the file: the information to locate the file on the disk
  - Access rights: per-process access mode information



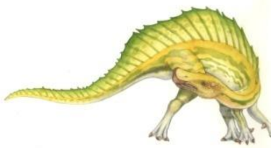
Open File Locking

- Provided by some operating systems and file systems
- File locks allow one process to lock a file and prevent other processes from gaining access to it.
- File locks are useful for files that are shared by several processes—for example, a system log file that can be accessed and modified by many processes in the system.
- File locks provide functionality similar to reader-writer locks.
  - Shared lock共享锁: like reader lock, allowing reading currently
  - Exclusive lock独占锁: like writer lock, only one process can acquire at a time.



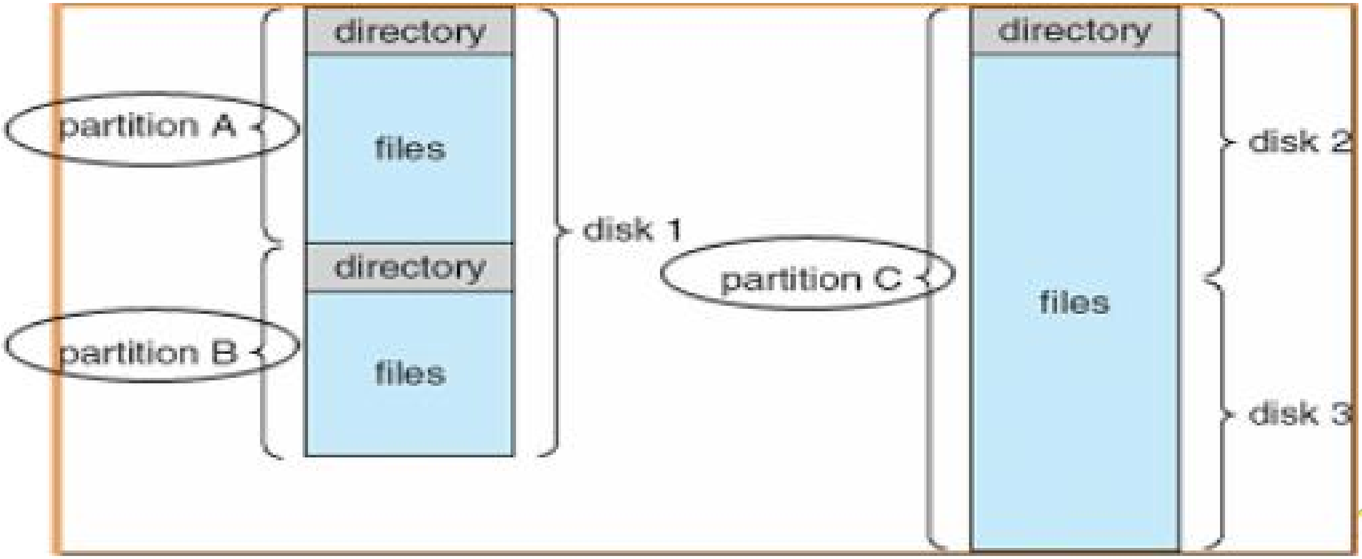
Open File Locking

- Further more, OS may provide either mandatory or advisory file-locking mechanisms.
  - **Mandatory 强制**
    - Once a process acquires an exclusive lock, the OS will prevent any other process from accessing the locked file, Even if other process will not modify the file.
    - OS ensures locking integrity 完整性.
    - Windows
  - **Advisory 建议**
    - Processes can find status of locks and decide what to do
    - Software developers ensure that locks are appropriately acquired and released.
    - Unix



Disk Structure

- A disk may be divided into many parts.
  - A part may hold an individual file system.
  - This is called a partition (分区).
- Sometimes, several disks are combined together to hold one large file system.
  - This collection of disks is also called a partition

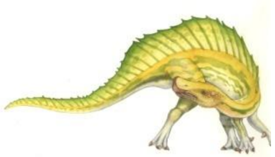


Disk Structure

- Disk or partition can be used raw – without a file system, or without formatted with a file system
- Partitions also known as minidisks, slices
- Partition containing file system known as a volume (卷)
- Each volume containing file system also tracks that file system’s info in device directory(设备目录) or volume table of contents(卷目录表)
- There may be several file systems, frequently all within the same operating system or computer

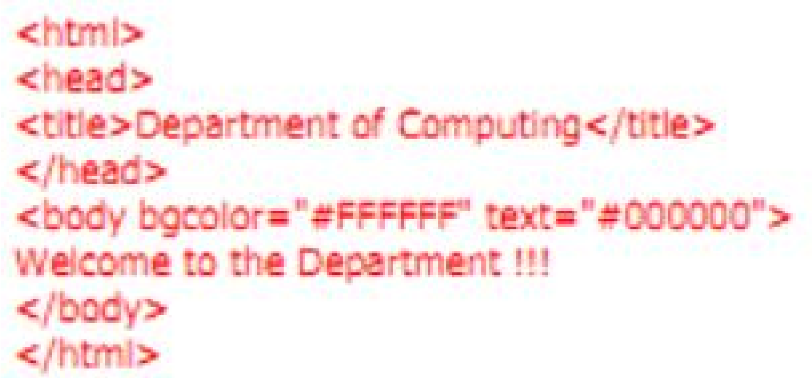
文件结构与存储设备

- 用户和文件系统往往从不同角度对待同一个文件：
  - 用户:从使用的角度，按信息的使用和处理方式组织文件。
  - 文件系统：从文件的存储和检索的角度，根据用户对文件的存取方式和存储介质的特性组织文件，决定用户文件存放在存储介质上的方式。
- 文件有两种形式的结构：逻辑结构和物理结构
  - 逻辑结构：从用户的角度看文件，由用户的访问方式确定。
  - 物理结构：文件在外存储器上的存储结构
  - 物理结构直接影响存储空间的使用和检索文件信息的速度
  - 逻辑文件保存到存储介质上的工作由文件系统来做，这样可以减轻用户的负担。根据用户对文件的存取方式和存储介质的特性，文件在存储介质上可以有多种组织形式。



File Structure

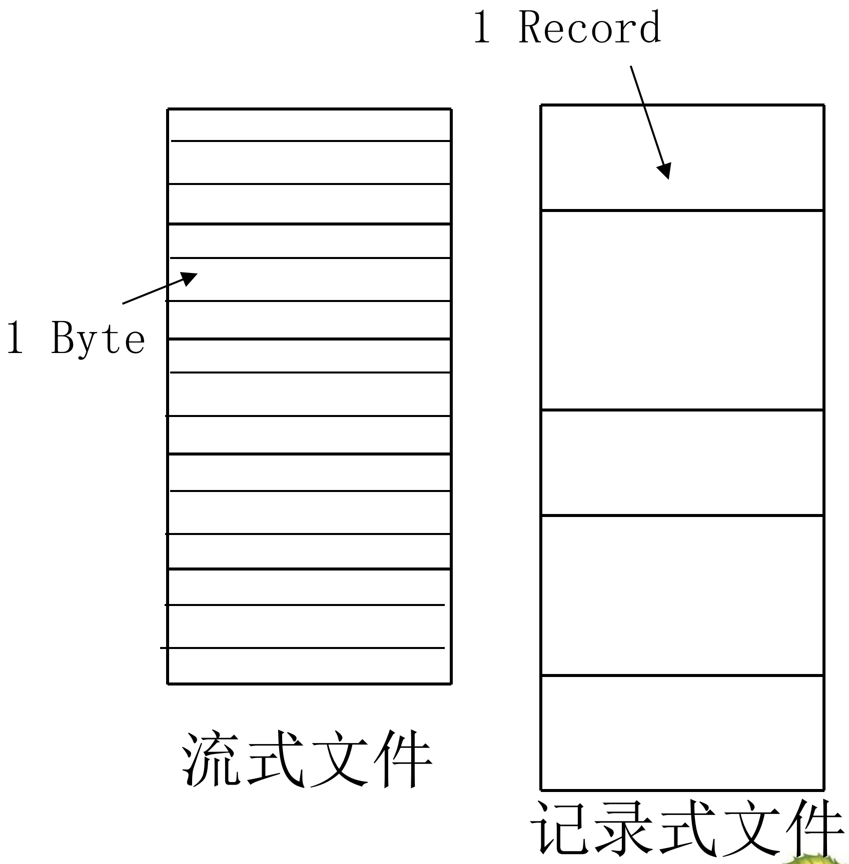
- A file may be structured or unstructured.
- Unstructured
  - The file is just a sequence of words or bytes.
- Simple record structure
  - Each record is stored in a line or a fixed number of lines.
  - Fixed length records.
  - Variable length records.
- Complex structure
  - Formatted document.
  - Relocatable load file.
- Unix only supports a simple unstructured file of consecutive bytes.
  - Application programs must interpret the file content by themselves.



```
<html>
<head>
<title>Department of Computing</title>
</head>
<body bgcolor="#FFFFFF" text="#000000">
Welcome to the Department !!!
</body>
</html>
```

文件的逻辑结构

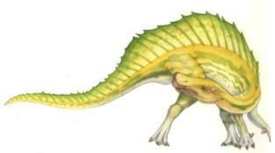
- 流式文件（无结构文件）
  - 构成文件的基本单位：字符
  - 文件是有逻辑意义、无结构的一串字符的集合
- 记录式文件（有结构文件）
  - 文件由若干记录组成，可以按记录进行读、写、查找等操作
  - 每条记录有其内部结构



文件的逻辑结构–顺序文件

（Sequential File）

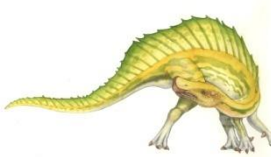
- 顺序文件的所有记录按键值的约定次序组织；
- 记录可以是定长的，也可以是变长的；
- 顺序文件常用于批量记录读取，对于访问某个记录的请求则处理性能不佳；
- 对于定长记录文件，若要查找第i个记录，可根据下式得到相对于第一个记录首址的地址：$A_i=i*d$（d为记录的长度）；
- 对于非定长的记录文件，若要查找第i个记录，则需要有每个记录的长度：$A_i=\Sigma d_i$（d为记录的长度）。



文件的逻辑结构–索引文件

（Indexed File）

- 为解决这类问题，往往建立一张索引表，记录下每个记录的长度及指向该记录的指针，从而方便了直接存取；
- 索引文件对主文件中的记录按需要的数据项（一个或几个）建索引表；
- 为每个记录设置一个表项；

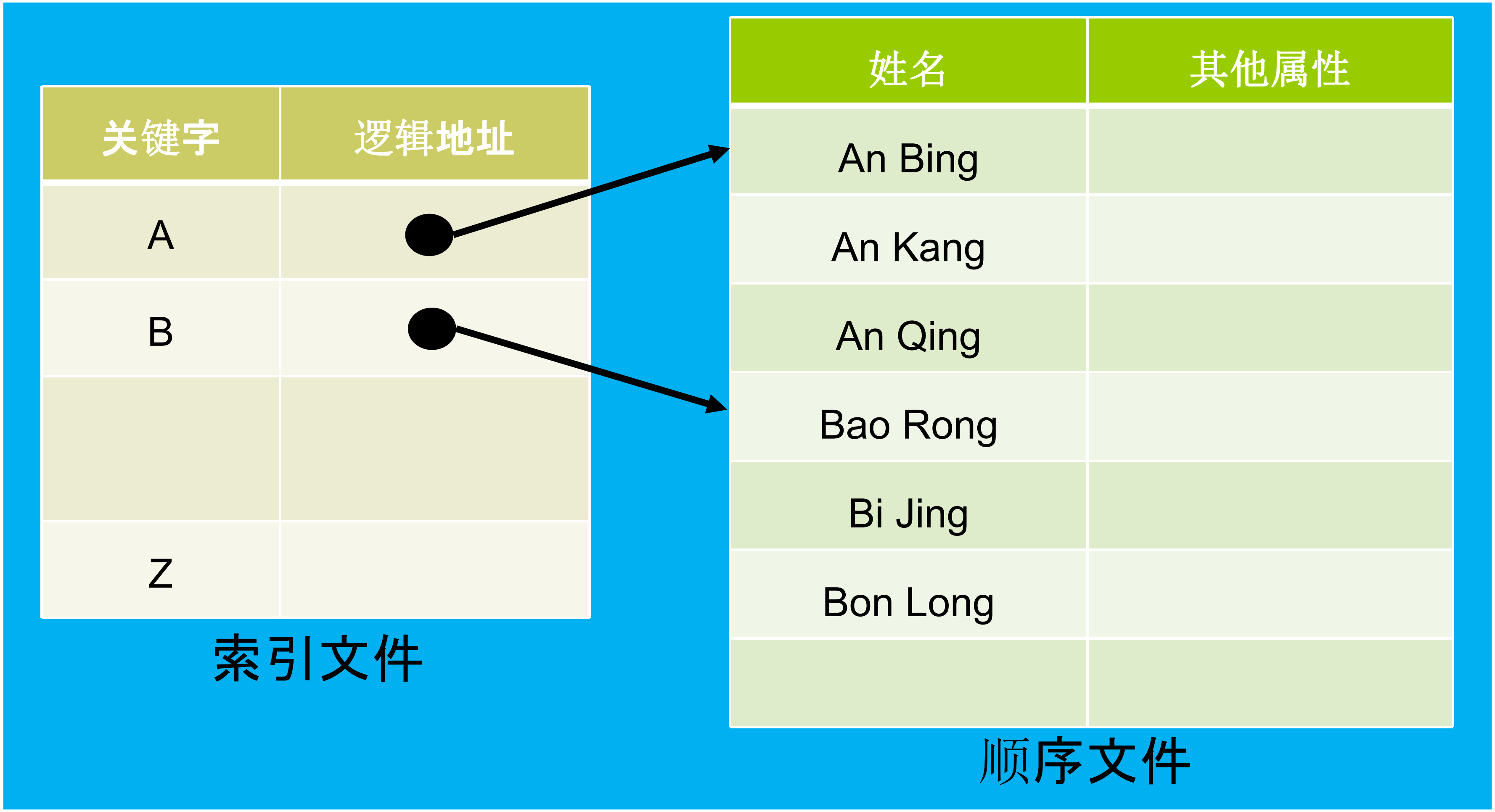


文件的逻辑结构–索引顺序文件

（Indexed Sequential File）

- 索引顺序文件是基于键的约定次序组织的。将顺序文件中的所有记录分为若干个组；
- 再为顺序文件建立一张索引表，表中记录每个组的第一个记录，该索引项包含记录的键值和指向该记录的指针。它是顺序文件和索引文件的结合。
- 检索时，先根据关键字去检索索引表，找到该记录所在组的第一个记录的位置，然后再利用顺序查找法查找主文件，找到所需记录。

索引顺序文件



关键字	逻辑地址
A	●
B	●
Z	

索引文件

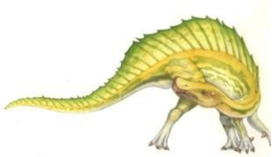
姓名	其他属性
An Bing	
An Kang	
An Qing	
Bao Rong	
Bi Jing	
Bon Long	

顺序文件

还可以组织成堆、散列等结构

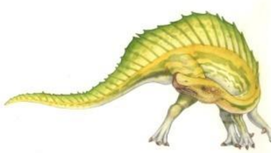
File Access Methods

- The information in the file can be accessed in several ways.
- Some systems only support some of them.
  - Sequential Access: simplest
  - Direct Access
  - Indexed access

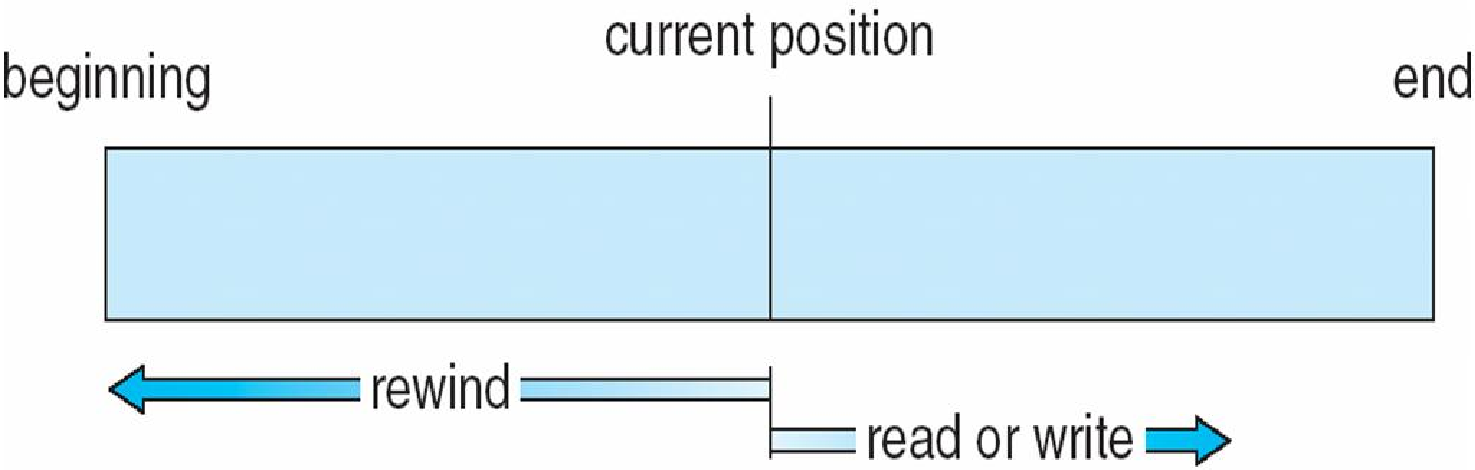


Access Methods

- **Sequential Access: simplest**
  - **Information in the file is processed in order, one record after the other.**
  - read_next(): read the next portion of the file and automatically advances a file pointer
  - write_next(): appends to the end of the file and advances to the new end of the file
  - Reset: moves the file pointer to the beginning
  - Skip forward/backward: move the file pointer forward without reading or move it backward.
    - is only supported in some systems.
  - Most systems support sequential access.



Sequential-access File



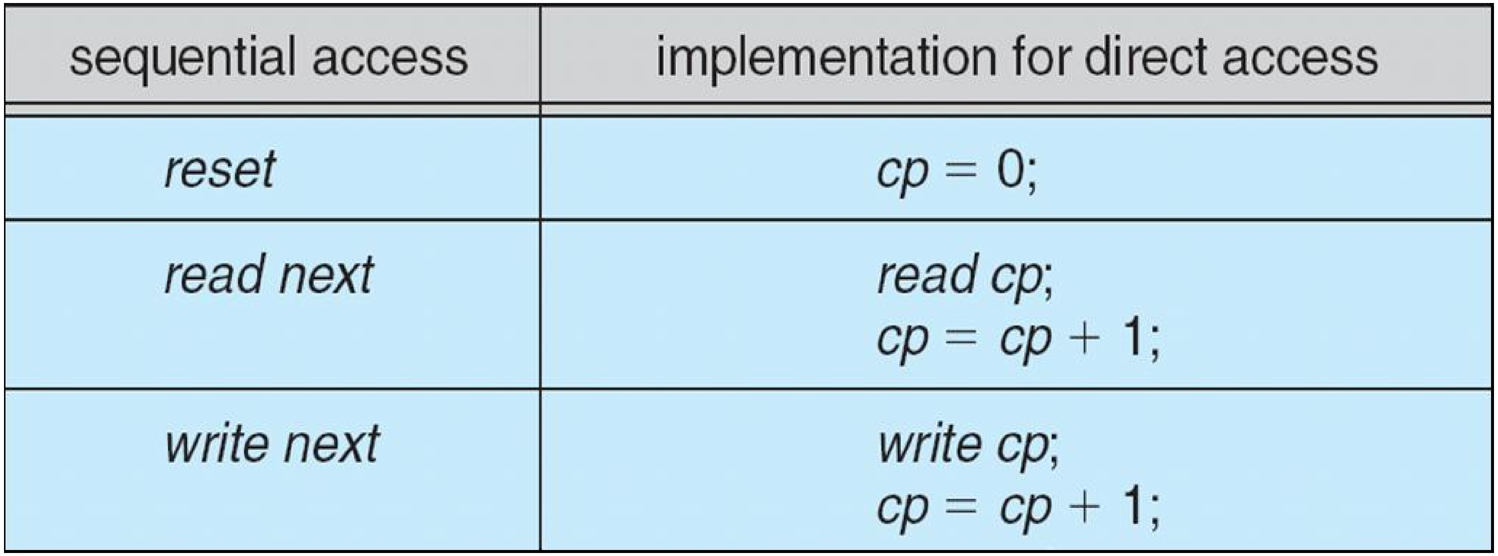
Access Methods

- **Direct Access: or relative access**
  - Based on a disk model of a file
  - The file is viewed as a numbered sequence of blocks or records.
  - There are no restrictions on the order of reading or writing for a direct-access file.
  - Are useful for immediate access to large amounts of information, for example, database query
  - For the direct-access method, the file operations must be modified to include the block number as a parameter.
    - read *n*: *return the n^{th} data item or block.*
    - write *n*: *update the n^{th} data item or block*
    - A system that provides sequential access operations Read next and Write next can support the two direct access operations above through a new operation *Position n*.
      - Position n: move the file pointer to the nth data item or block.
      - Read n and Write n could be implemented using Position n and then Read next and Write next.

Simulation of Sequential Access on Direct-access File

- Sequential access can also be provided easily by a system that only supports direct access.

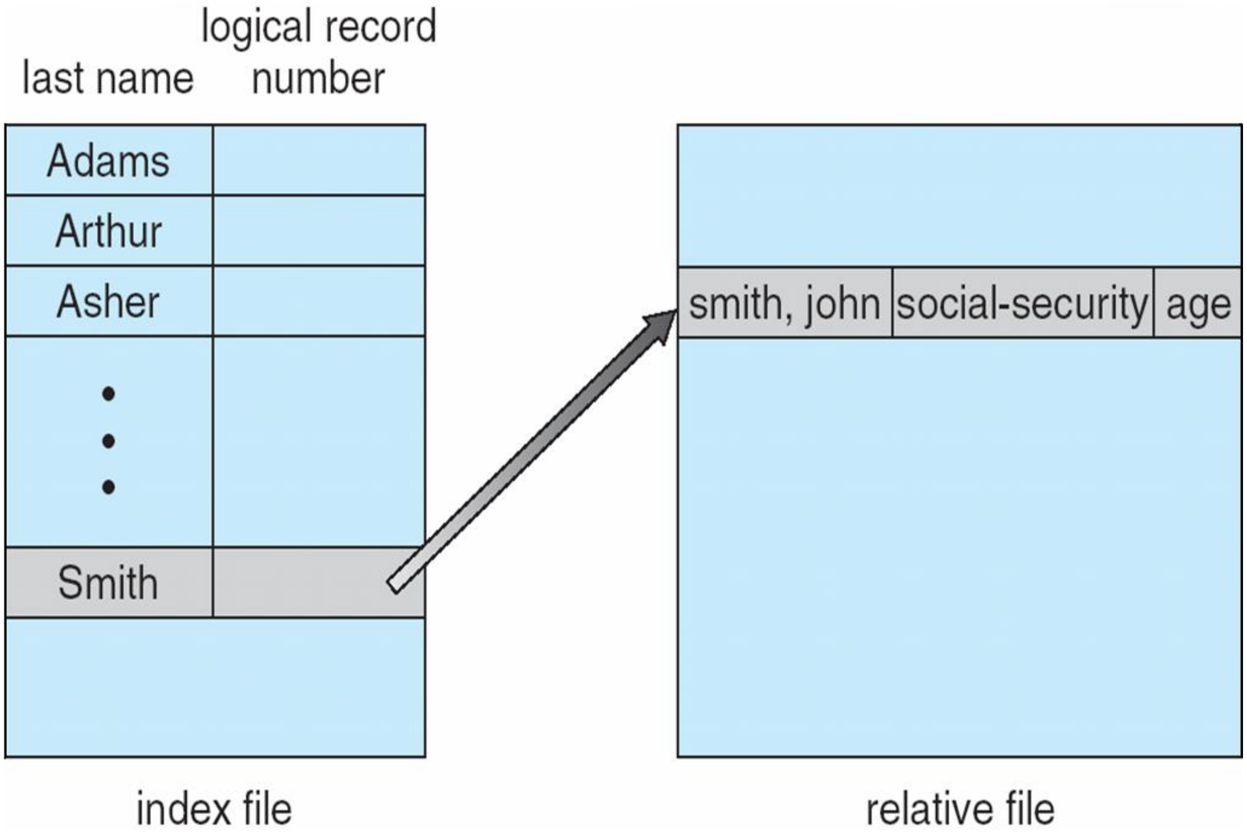
Keep a variable ***cp*** that defines the current position



sequential access	implementation for direct access
reset	*cp* = 0;
read next	*read cp*; *cp* = *cp* + 1;
write next	*write cp*; *cp* = *cp* + 1;

Other Access Methods

- **Other access methods can be built on top of a direct-access method.**
- These methods generally involve the construction of an index for the file.



Directory Structure

- Some systems store millions of file on disk.
- To manage all these data, we need to organize them.
- This organization involves the use of directory.
- A directory is a collection of nodes containing information about all files.

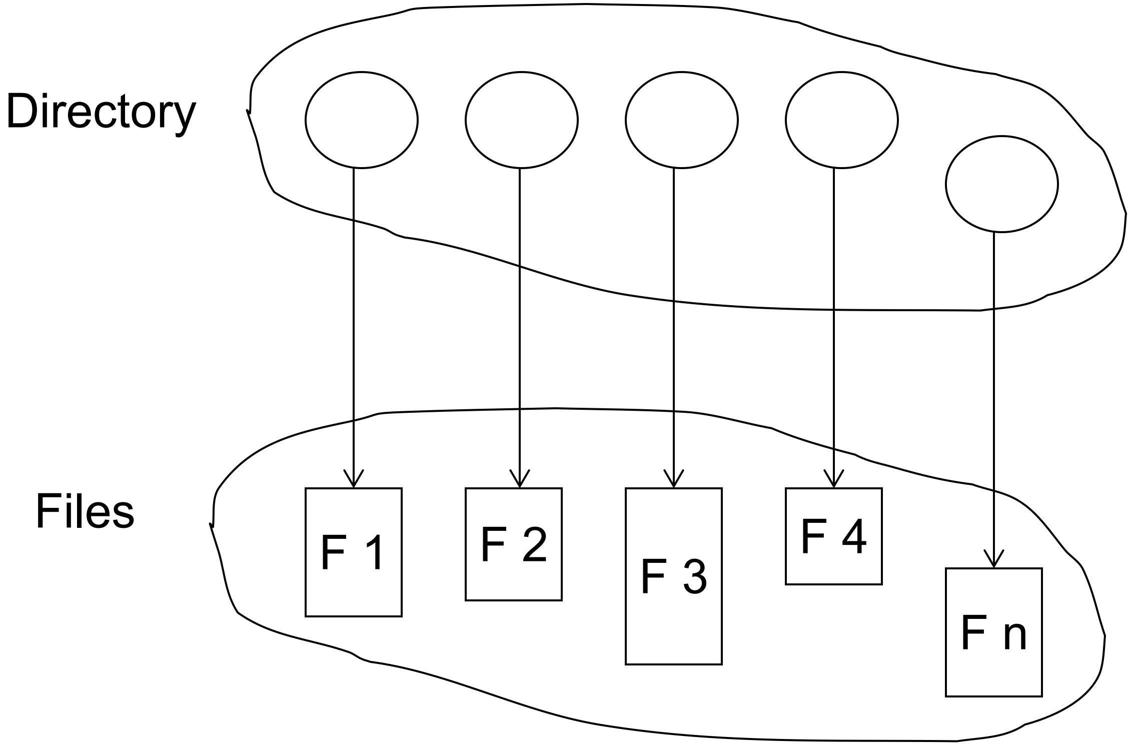
文件目录、目录项与目录文件

- 文件目录
  - 统一管理每个文件的元数据，以支持文件名到文件物理地址的转换
  - 将所有文件的管理信息组织到一起，即构成文件目录
  - 文件目录是用于检索文件的，它是文件系统实现按名存取的重要手段，它的组织和管理应便于检索和防止冲突；
  - 目录项：构成文件目录的基本单位（目录项是FCB，文件目录是文件控制块的有序集合）。
- 目录文件：为了实现对文件目录的管理，将文件目录以文件的形式保存在外存，这个文件就叫目录文件
- 目录结构的组织关系到：
  - 文件的存取速度
  - 文件共享性和安全性

file permissions
file dates (create, access, write)
file owner, group, ACL
file size
file data blocks or pointers to file data blocks

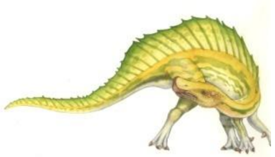
Directory Structure

- Both the directory structure and the files reside on disk
- Backups of these two structures are kept on tapes



Operations Performed on Directory

- The directory can be viewed as a symbol table that translates file names into their directory entries.
- Operations performed on a directory:
  - Search for a file
  - Create a file
  - Delete a file
  - List a directory 遍历目录
  - Rename a file
  - Traverse the file system 遍历文件系统

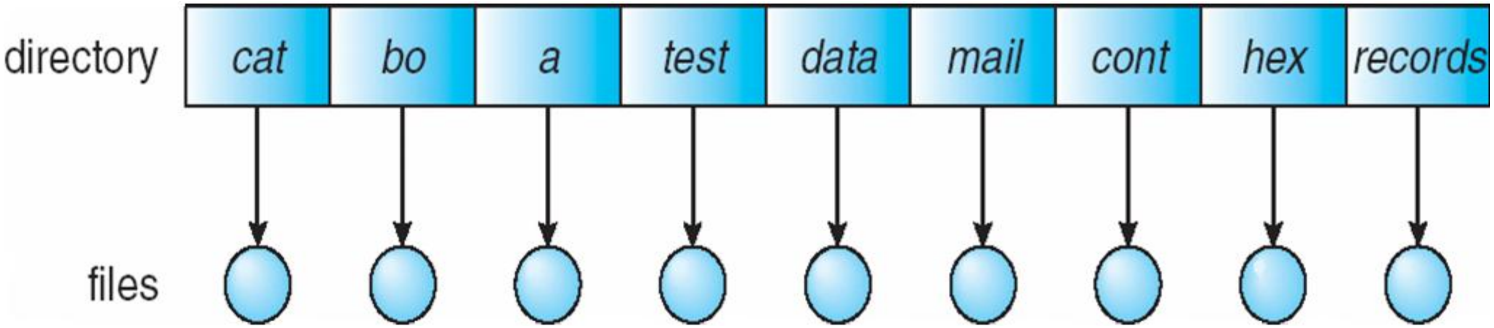


Organize the Directory (Logically) to Obtain

- Efficiency – locating a file quickly
- Convenient to users
  - Naming
    - Two users can have same name for different files
    - The same file can have several different names
  - Grouping – logical grouping of files by properties, (e.g., all Java programs, all games, …)

Single-Level Directory

- A single directory for all users



Naming problem

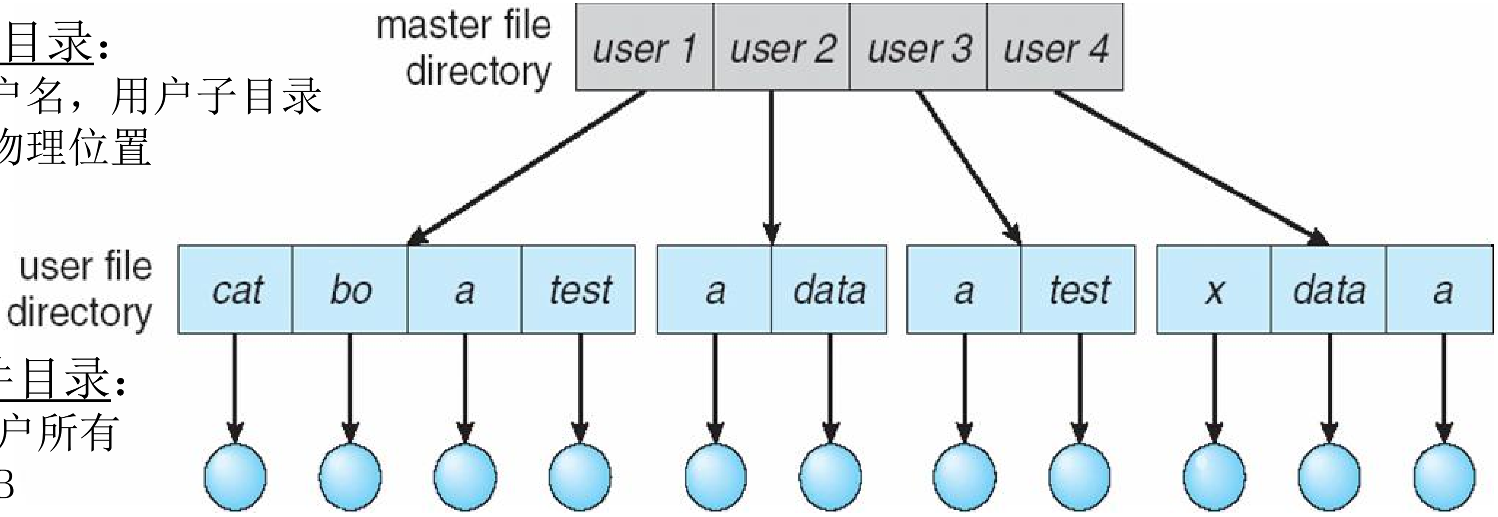
Grouping problem

Searching problem: time consuming

Two-Level Directory

- Separate directory for each user

<u>主文件目录</u>：
给出用户名，用户子目录
所在的物理位置

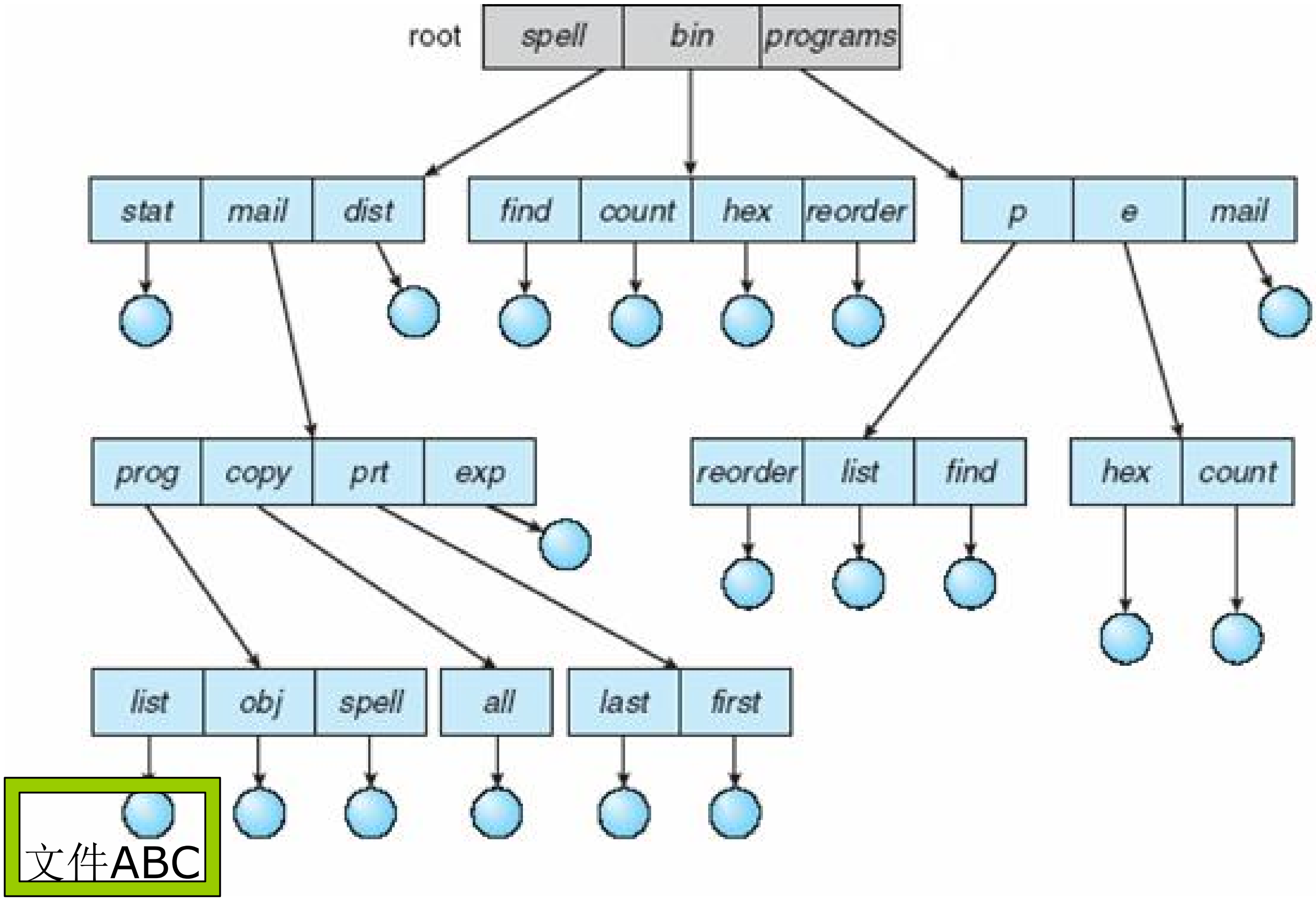


<u>用户文件目录</u>：
给出该用户所有
文件的FCB

- Path name
- Can have the same file name for different user
- Efficient searching
- No grouping capability

Tree-Structured Directories

Extend two-level directory into multiple levels like a tree



路径名：/spell/mail/prog/list/ABC

Tree-Structured Directories (Cont)

- Efficient searching
- Grouping Capability
- A current working directory is maintained to define the default searching place for files.
  - Path names should be given to locate for a file.
  - Can use absolute or relative path names (from the current working directory).
- Adopted in MS-DOS.

Tree-Structured Directories (Cont)

- **Absolute** or **relative** path name
- Creating a new file is done in current directory
- Delete a file

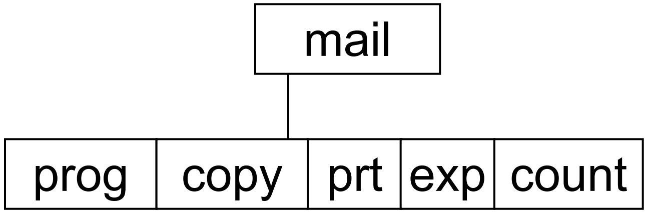
  rm <file-name>

- Creating a new subdirectory is done in current directory

  mkdir <dir-name>

  Example: if in current directory /mail

  mkdir count



Deleting "mail" ⇒ deleting the entire subtree rooted by "mail"

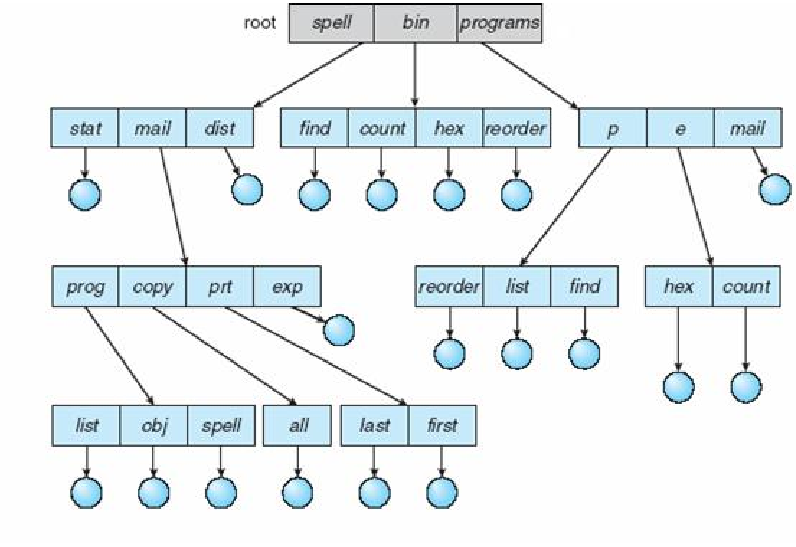
Tree-Structured Directories (Cont)

优点

- 层次结构清晰
- 便于管理与保护
- 有利于文件分类
- 解决重名问题
- 提高文件检索速度

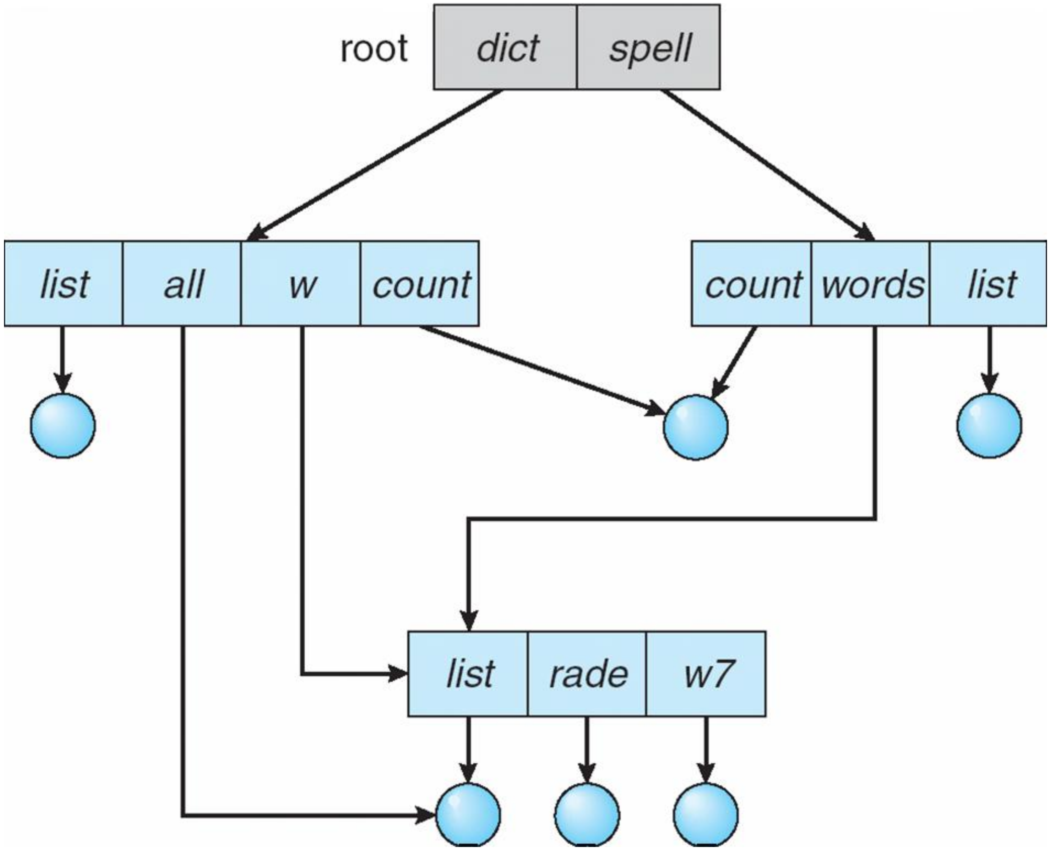
缺点

- 查找一个文件按路径名逐层检查，由于每个文件都放在外存，多次访盘影响速度
- 不支持共享



Acyclic-Graph Directories 无环图目录

- Have shared subdirectories and files

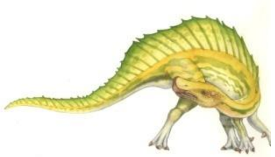


Acyclic-Graph Directories (Cont.)

- An acyclic-graph directory structure is more flexible than a simple tree structure, but it is also more complex.
- A common way is to create a new directory entry called a link.
  - A **link** is effectively a **pointer** to another file or subdirectory.
- Several problems:
  - A file may have multiple absolute path names.
  - File deletion: when can the space allocated to a shared file be deallocated and reused?

Acyclic-Graph Directories (Cont.)

- Problems about deleting sharing files
- If remove the file whenever anyone deletes it, there may be dangling pointers(悬挂指针) to the nonexistent file.
- Solutions:
  - Sharing by symbolic links
    - Deletion of a link: do not affect the original file
    - Deletion the shared file: the space for the file is deallocated, leaving the links dangling, we can leave the links until an attempt is to made to use them—treat as an illegal file access.
      - Unix and Windows do like this.
  - Sharing by hard link(nonsymbolic links)
    - Entry-hold-count solution
    - Unix iNode



文件别名实现

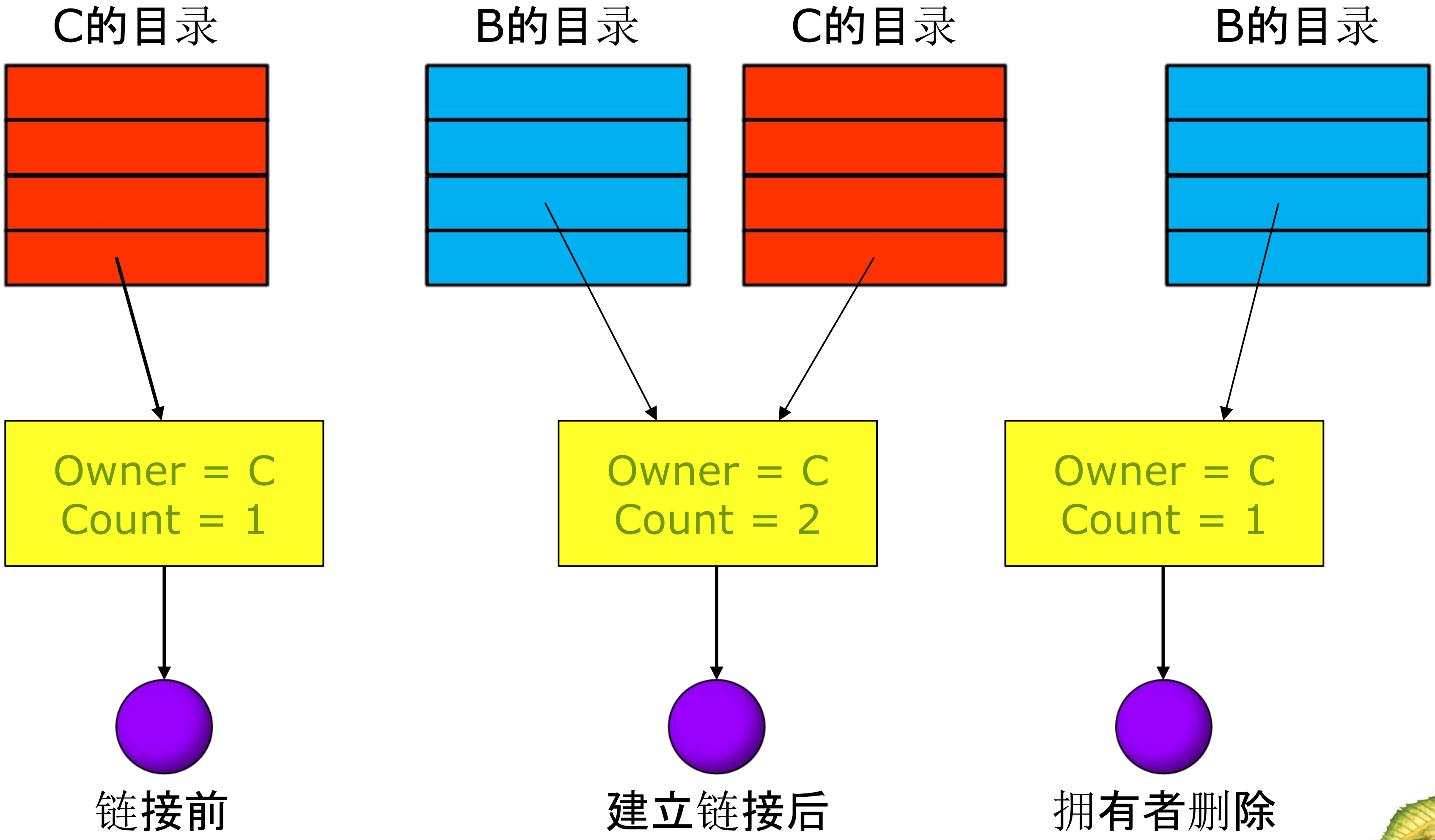
（1）基于符号链接（symbolic link, shortcut）:特殊类型的文件，其内容是到另一个目录或文件路径的链接。建立符号链接文件，并不影响原文件，实际上它们各是一个文件。

（2）基于索引结点（index node）：也称为硬链接（hard link），基于改进的多级目录结构，将目录内容分为两部分——文件名和索引结点。前者包括文件名和索引结点编号，后者包括文件的其他内容（包括属主和访问权限）。

– 通过多个文件名链接（link）到同一个索引结点，可建立同一个文件的多个彼此平等的别名。

– 别名的数目记录在索引结点的链接计数中，若其减至0，则文件被删除。

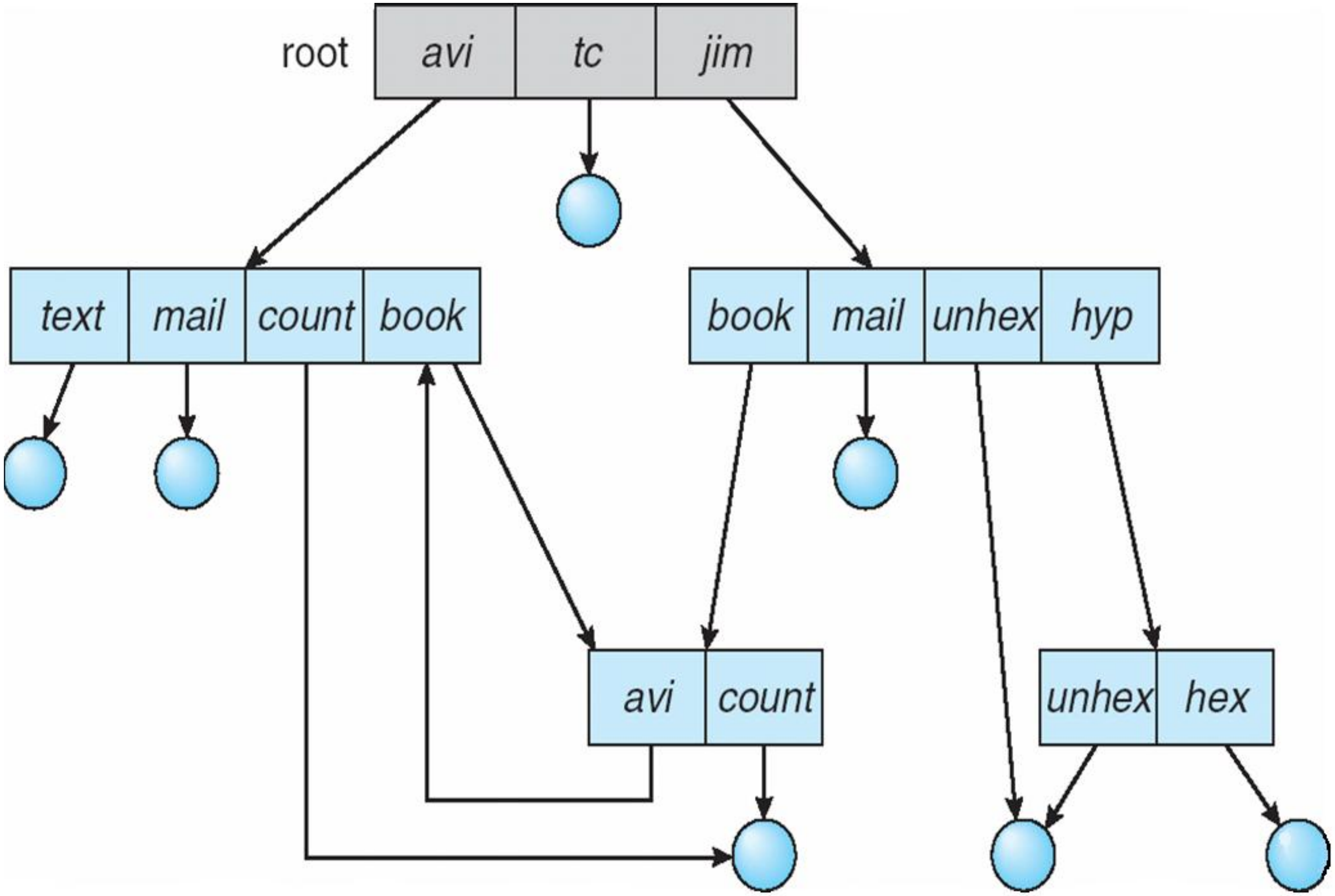
基于索引节点的共享方式



进程B链接前后的情况

General Graph Directory 通用图目录

Allow for cycles in directories to exist to form general graph.



General Graph Directory (Cont.)

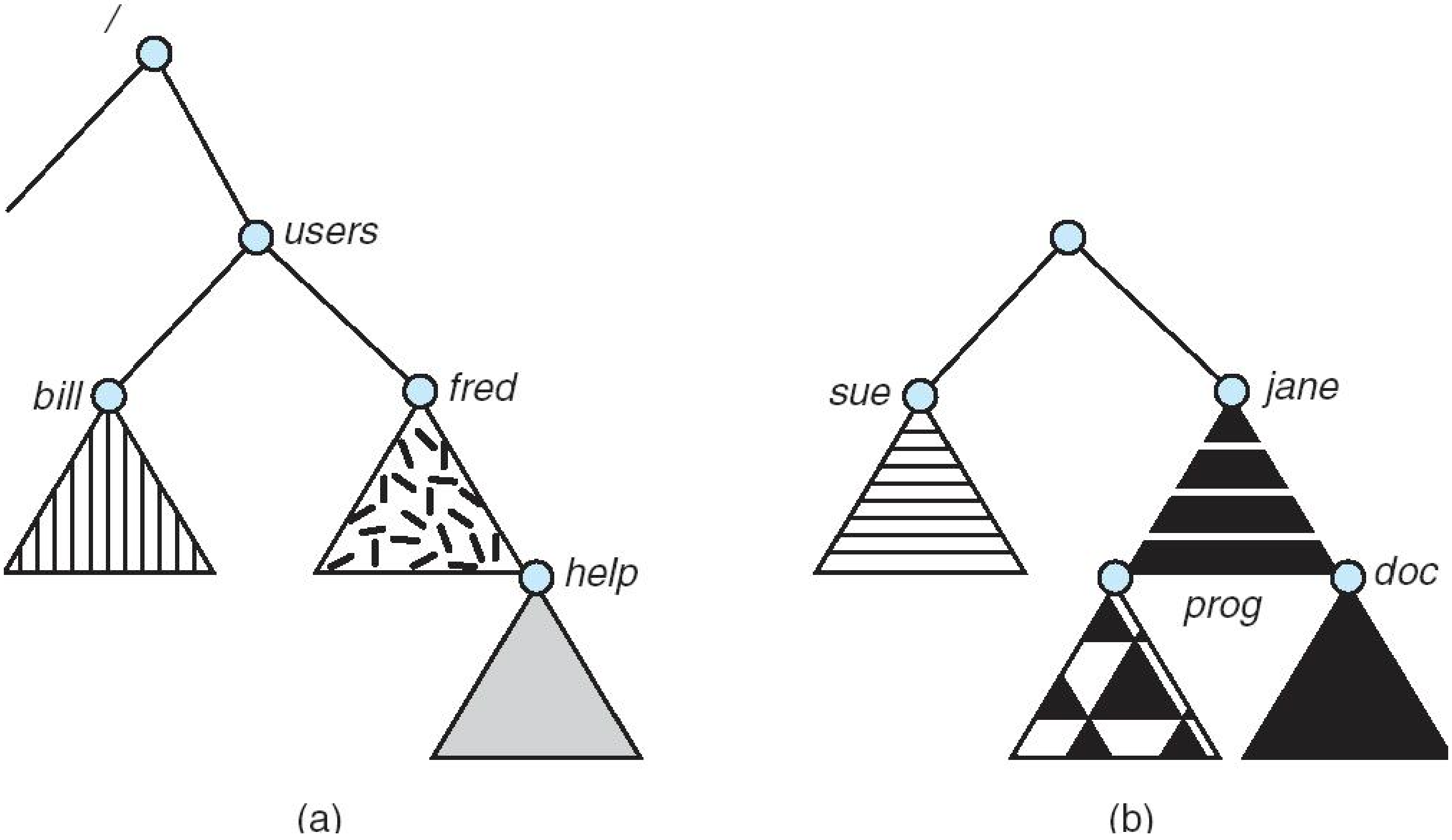
- The general graph with a cycle could cause some problems.
  - How to count the total number of files?
    - A poorly design algorithms will fall into an infinite loop.
  - If we need a recursive backup of all the files down the directories, how can that be done? 限制搜索时访问目录的数量
  - How can we delete a file using reference count?
    - Without cycle, a deleted file has a reference count of zero.
    - With cycle, deleted files may have reference count greater than zero.
    - **Garbage collection** techniques are needed, but garbage collection algorithms are expensive to run.
- We could prevent the problems from happening.
  - Ensure that there is no cycle in the graph.
    - Allow only links to file, not subdirectories.
    - Every time a new link is added, use a cycle detection algorithm to determine whether there is a cycle created.

File System Mounting

- A file system must be **mounted** before it can be accessed 安装
- How to mount?
  - The OS is given the name of the device and the mount point—the location within the directory structure where the file system is to be attached.
    - Typically, a mount point is an empty directory.
  - Then, the OS verifies that the device contains a valid file system
  - Finally, the OS notes in its directory structure that a file system is mounted at the specified mount point.

(a) Existing. (b) Unmounted Partition

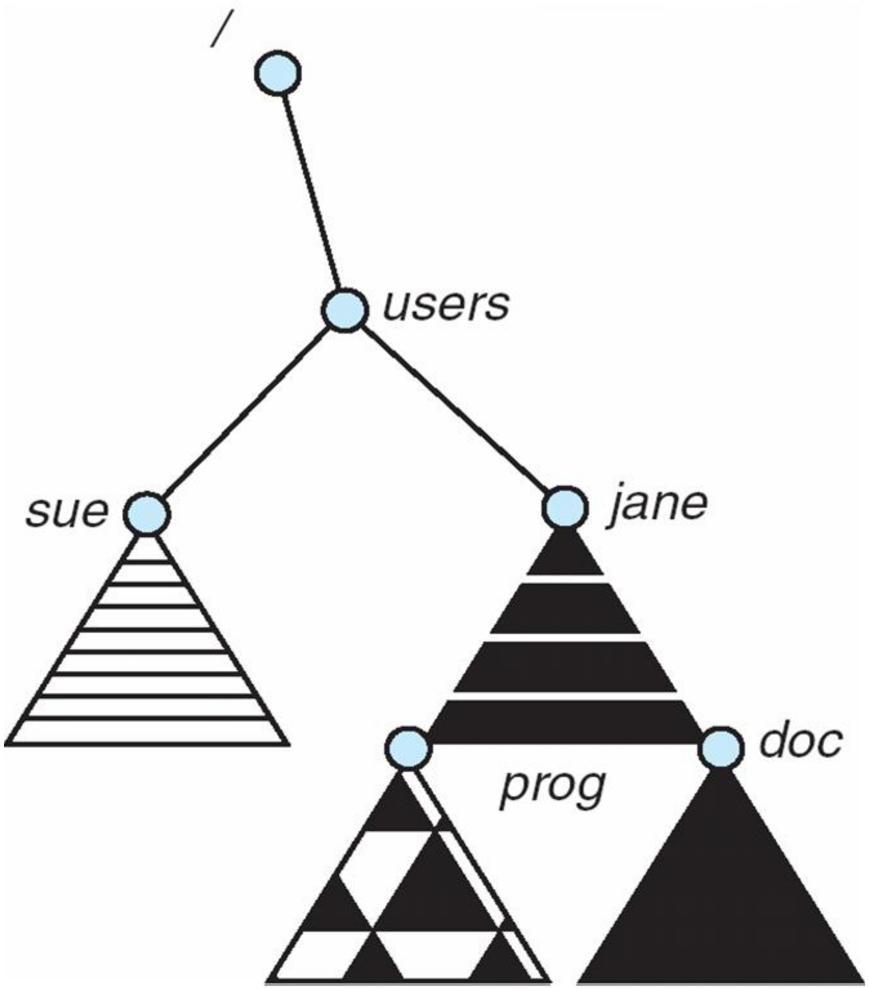
- A unmounted file system (i.e. Fig. (b)) is mounted at a **mount point**



Existing file system

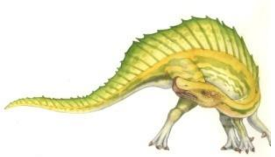
Unmounted volume

Mount Point



File Sharing – Multiple Users

- For multiple users, the issues of file sharing, file naming, and file protection become preeminent.
- To implement sharing and protection, the system must maintain more file and directory attributes than are needed on a single-user system.
- File owner user, group user, other user
  - **User IDs** identify users, allowing permissions and protections to be per-user
  - **Group IDs** allow users to be in groups, permitting group access rights



File Sharing– Remote File Systems

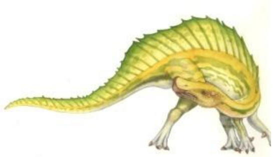
- Uses networking to allow file system access between systems
  - Manually via programs like FTP
  - Automatically, seamlessly using **distributed file systems (remote directories are visible from a local machine.)**
  - Semi automatically via the **world wide web**

File Sharing– Remote File Systems

- Client-server model allows clients to mount remote file systems from servers.
  - Server: machine containing the files
  - Client: machine seeking access to the files
  - A server can serve multiple clients.
  - Standard operating system file calls are translated into remote calls to be executed at the remote machine.

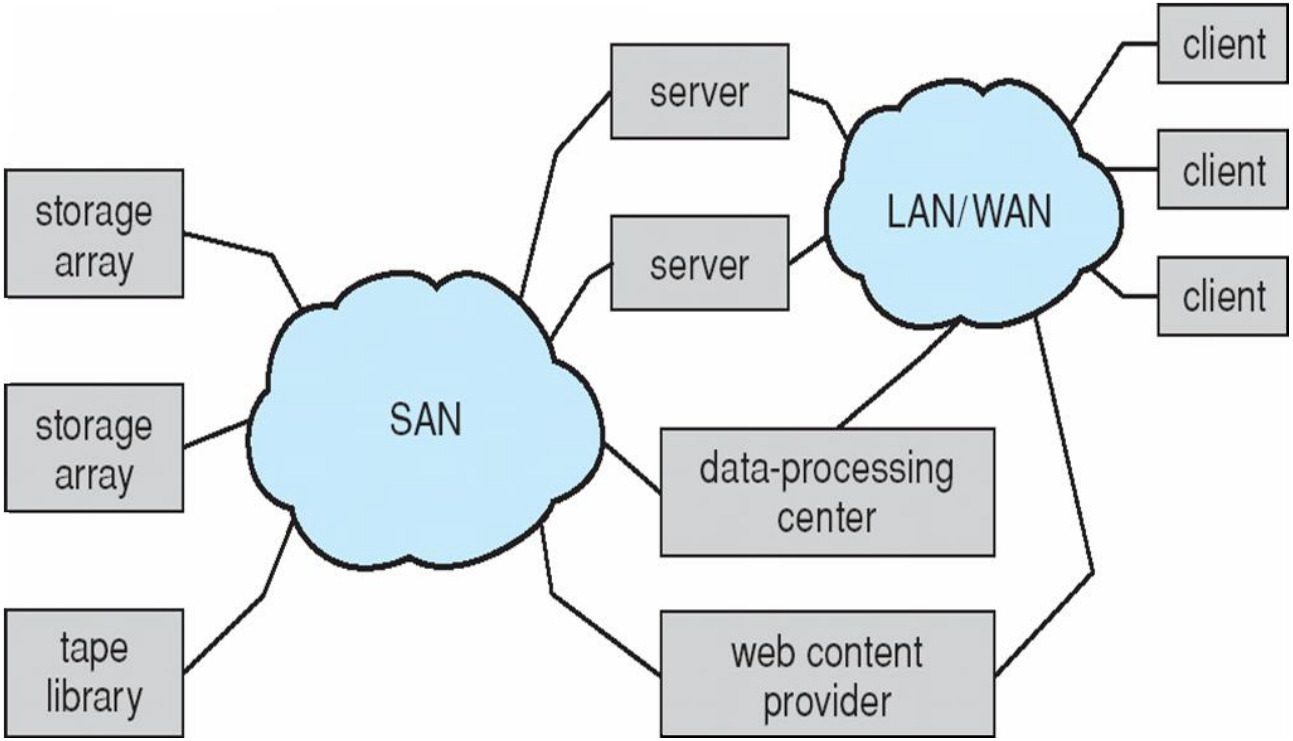
File Sharing

- Network File System (NFS) developed by Sun is a common distributed file-sharing method.
  - It is the standard Unix client-server file sharing approach.
- Storage Area Network (SAN 存储区域网络) is a current technology to provide large storage capacities to a large user population.
  - It is a high-speed network dedicated to the task of transporting data for storage and retrieval.
  - It connects together computers and storage devices to allow sharing of the pool of storage devices.
  - It is adopted by large institutions.
- Cloud storage is the newest technology to host data in a collection of nodes over the cloud.
  - Service is usually provided by third party, usually data center.
  - Common storage: iCloud, dropbox, Google drive.



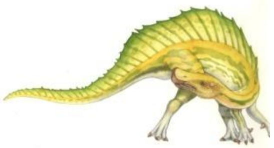
Storage Area Network

- Common in large storage environments (and becoming more common)
- Multiple hosts attached to multiple storage arrays - flexible



Protection

- File owner/creator should be able to control:
  - what can be done
  - by whom
- Types of access
  - **Read**
  - **Write**
  - **Execute**
  - **Append**：在文件末尾写入新的信息
  - **Delete**
  - **List: list the name and attributes of the file**
  - 实现：
- 用户身份验证
  - 口令、密码、回答问题、指纹、虹膜…
- 访问控制



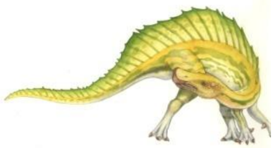
Access control

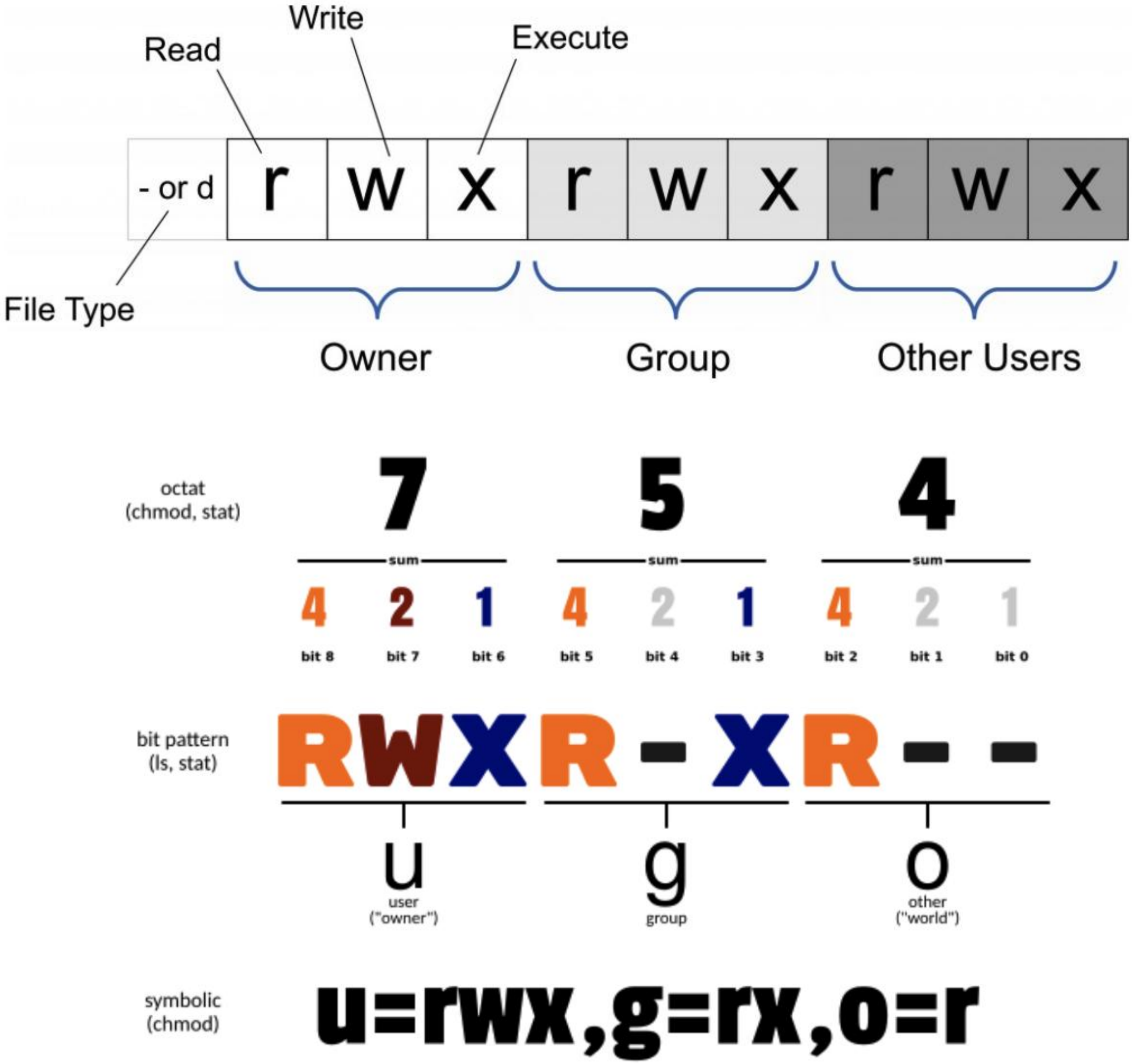
- The most general scheme is to associate with each file and directory an access-control-list (ACL).
  - showing who can do what.
- Consider Unix/Linux，采用文件的二级存取控制
  - 审查用户身份、审查操作的合法性
  - 第一级：对访问者的识别
    - Three classes of users: owner, group, other (public or universe).
  - 第二级： 对操作权限的识别
    - Three modes of accesses: read, write, execute.
  - For example, the command chmod 751 game will define the access control list on game to be something like

    <owner: read, write, execute>

    <group: read, execute>

    <other: execute>





Read

Write

Execute

- or d

r w x r w x r w x

File Type

Owner

Group

Other Users

octat
(chmod, stat)

7 5 4

sum sum sum

4 2 1 4 2 1 4 2 1

bit 8 bit 7 bit 6 bit 5 bit 4 bit 3 bit 2 bit 1 bit 0

bit pattern
(ls, stat)

RWXR-XR--

u
user
("owner")

g
group

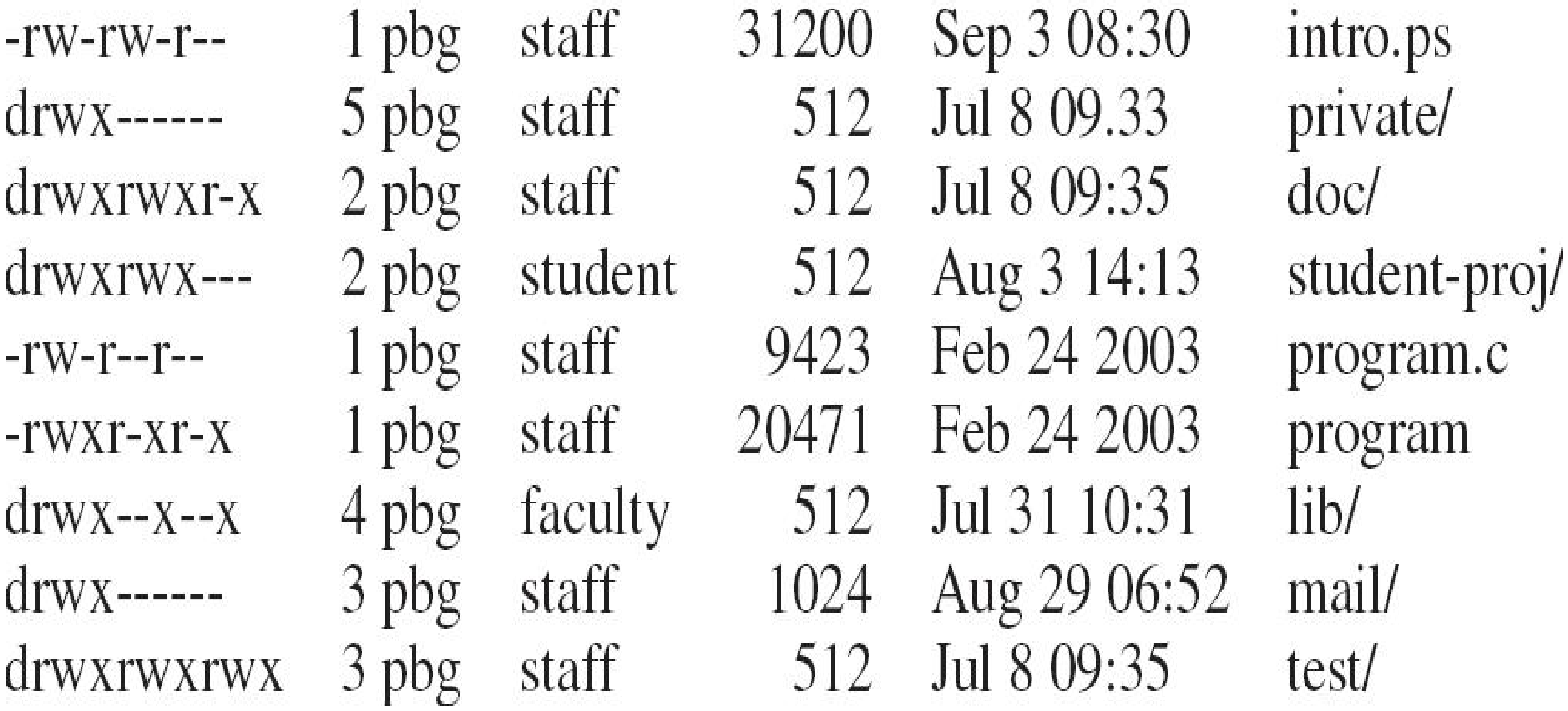
o
other
("world")

symbolic
(chmod)

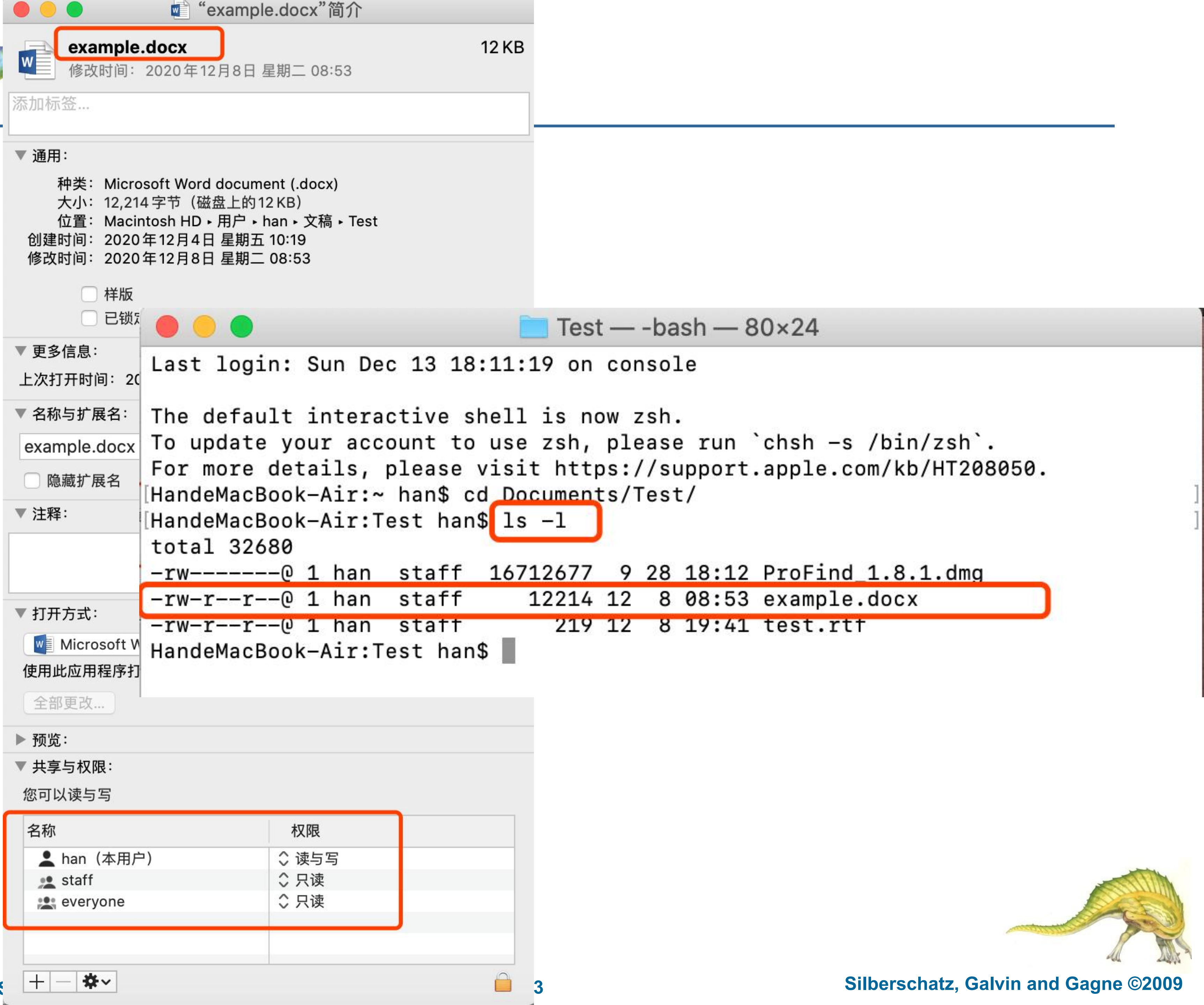
u=rwx,g=rx,o=r

A Sample UNIX Directory Listing

ls –l   who can do what



```
-rw-rw-r--    1 pbg   staff     31200  Sep 3 08:30    intro.ps
drwx------    5 pbg   staff       512  Jul 8 09.33    private/
drwxrwxr-x    2 pbg   staff       512  Jul 8 09:35    doc/
drwxrwx---    2 pbg   student     512  Aug 3 14:13    student-proj/
-rw-r--r--    1 pbg   staff      9423  Feb 24 2003    program.c
-rwxr-xr-x    1 pbg   staff     20471  Feb 24 2003    program
drwx--x--x    4 pbg   faculty     512  Jul 31 10:31   lib/
drwx------    3 pbg   staff      1024  Aug 29 06:52   mail/
drwxrwxrwx    3 pbg   staff       512  Jul 8 09:35    test/
```



“example.docx”简介

example.docx 12 KB

修改时间： 2020年12月8日 星期二 08:53

▼ 通用：

种类： Microsoft Word document (.docx)

大小： 12,214字节（磁盘上的12 KB）

位置： Macintosh HD ▸ 用户 ▸ han ▸ 文稿 ▸ Test

创建时间： 2020年12月4日 星期五 10:19

修改时间： 2020年12月8日 星期二 08:53

□ 样版

□ 已锁

▼ 更多信息：

上次打开时间： 20

▼ 名称与扩展名：

example.docx

□ 隐藏扩展名

▼ 注释：

▼ 打开方式：

Microsoft W

使用此应用程序打

▶ 预览：

▼ 共享与权限：

您可以读与写

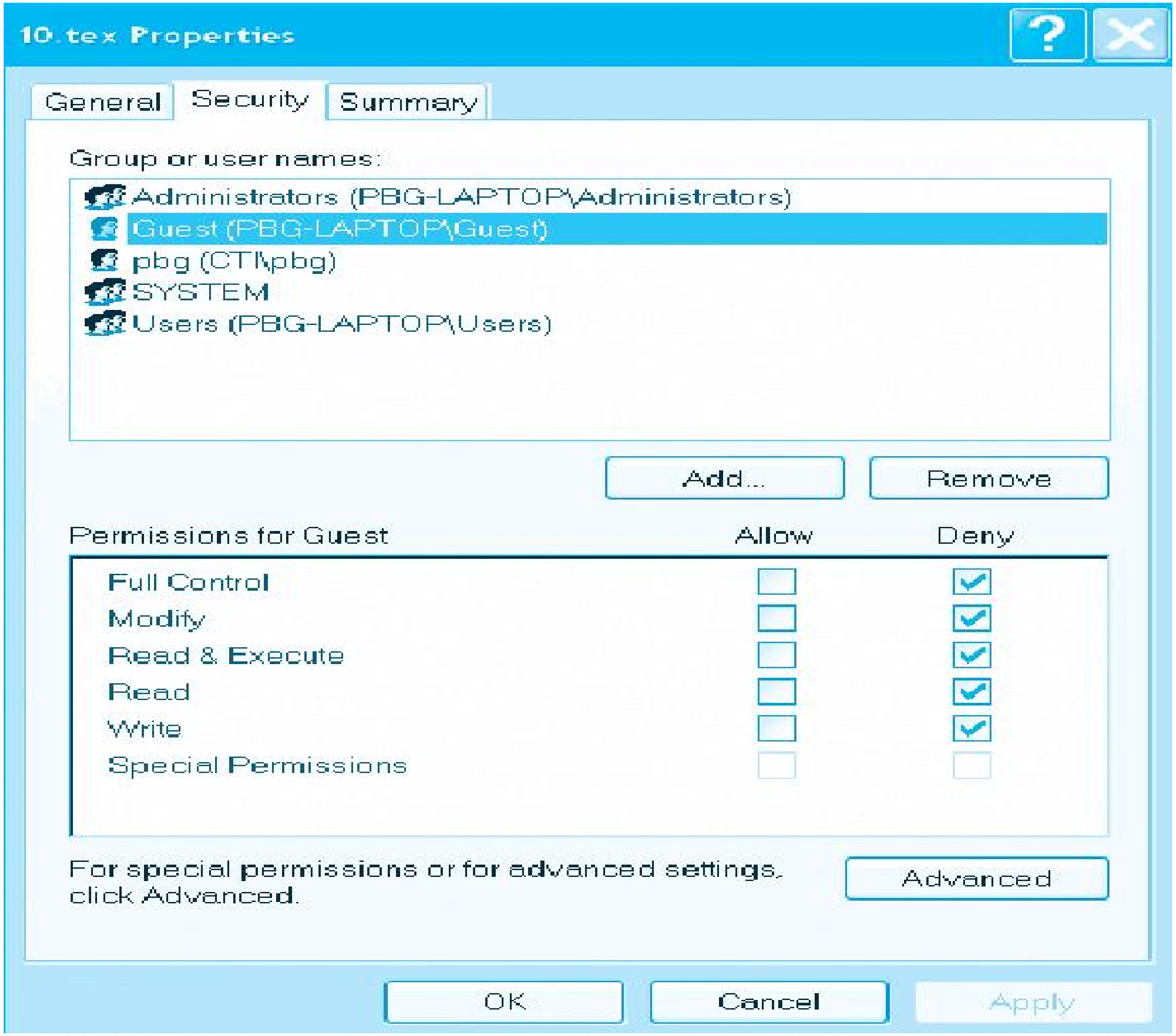
名称	权限
han（本用户）	读与写
staff	只读
everyone	只读

Test — -bash — 80×24

```
Last login: Sun Dec 13 18:11:19 on console

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
HandeMacBook-Air:~ han$ cd Documents/Test/
HandeMacBook-Air:Test han$ ls -l
total 32680
-rw-------@ 1 han  staff  16712677  9 28 18:12 ProFind_1.8.1.dmg
-rw-r--r--@ 1 han  staff     12214 12  8 08:53 example.docx
-rw-r--r--@ 1 han  staff       219 12  8 19:41 test.rtf
HandeMacBook-Air:Test han$ 
```

Windows XP Access-control List Management



Group or user names:

- Administrators (PBG-LAPTOP\Administrators)
- Guest (PBG-LAPTOP\Guest)
- pbg (CTI\pbg)
- SYSTEM
- Users (PBG-LAPTOP\Users)

Permissions for Guest	Allow	Deny
Full Control	□	☑
Modify	□	☑
Read & Execute	□	☑
Read	□	☑
Write	□	☑
Special Permissions	□	□

For special permissions or for advanced settings, click Advanced.

End of Chapter 10