

# 2025第16次课堂小测试



# 操作系统      *Operating System*

## 第七章 文件系统 文件的实现

原仓周

yuancz@buaa.edu.cn

# 内容提要

- 文件系统基本概念
- 文件系统实现方法
- 文件系统实例分析

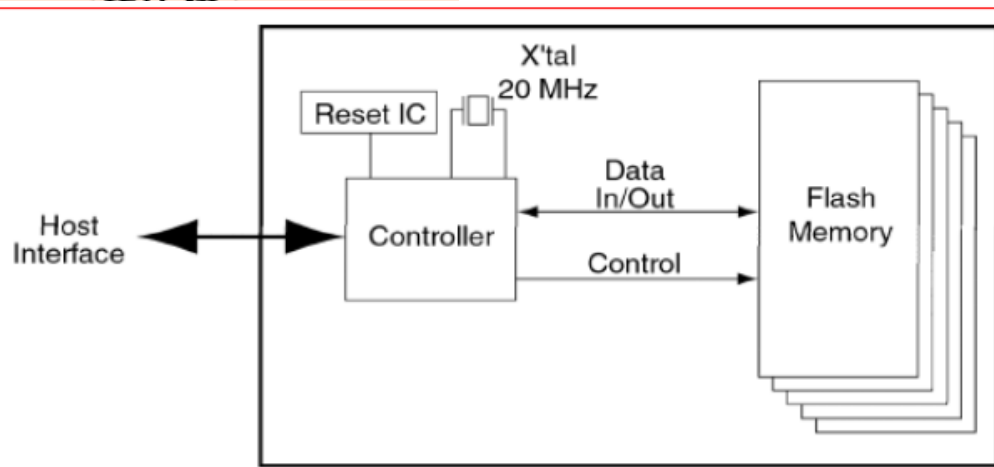
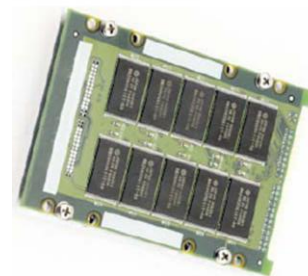
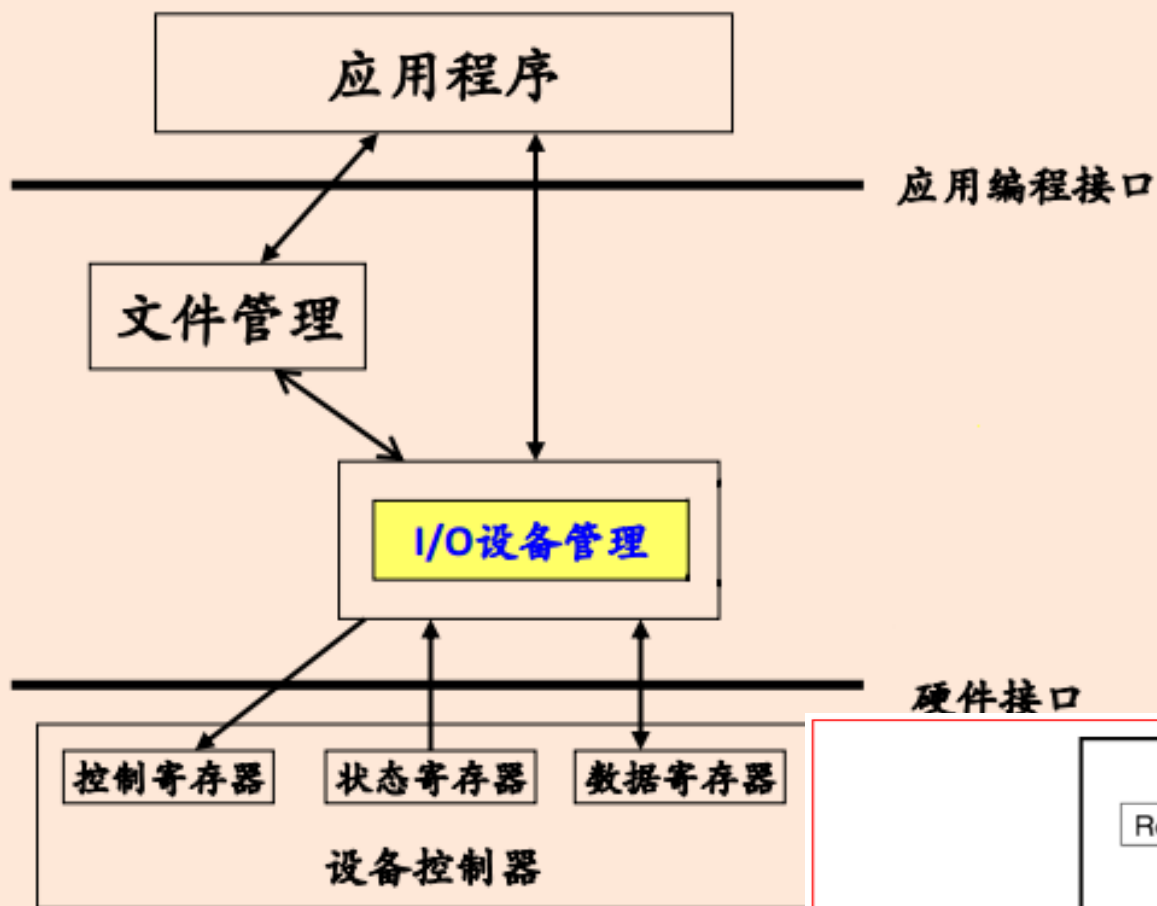
# 内容提要

- 文件系统基本概念
  - 文件系统的概念
  - 文件系统的模型
  - 文件的概念
  - 目录的概念
- 文件系统实现方法
- 文件系统实例分析

# 内容提要

- 文件系统基本概念
  - 文件系统的概念
  - 文件系统的模型
  - 文件的概念
  - 目录的概念
- 文件系统实现方法
- 文件系统实例分析

# 文件系统与计算机系统的资源



# 数据持久化需要满足的基本要求

- 三个基本要求
  - 能够存储大量的数据（突破地址空间限制）
  - 长期保存：进程终止以后，数据依然存在
  - 可以共享数据：有些数据可能会被多个不同的进程所访问；
- 为解决这些问题，人们提出“文件”的概念，把数据组织成文件的形式，用文件作为数据的存储和访问单位。

# 文件系统的定义和目的

定义：

- 操作系统中与文件管理有关的那部分软件和被管理的文件以及实施管理所需要的数据结构的总体

目的：

- 为系统管理者和用户提供了对文件的透明存取（按名存取）
  - 不必了解文件存放的物理机制和查找方法，只需给定一个代表某段程序或数据的文件名称
  - 文件系统就会自动地完成对给定文件名称相对应的文件的有关操作



# 文件系统的任务

## ■ 方便的文件访问

- 以符号名称作为文件标识，便于用户使用

## ■ 并发文件访问和控制

- 在多道程序系统中支持对文件的并发访问和控制

## ■ 统一的用户接口

- 为不同设备提供统一接口，方便用户操作和编程

## ■ 多种文件访问权限

- 在多用户系统中的不同用户对同一文件会有不同的访问权限

## ■ 执行效率和差错恢复

- 存储效率、检索性能、读写性能
- 能够验证文件的正确性，并具有一定的差错恢复能力

# 文件系统的功能

- 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，并方便用户使用
- 文件系统要完成的任务
  1. 统一管理磁盘空间，实施磁盘空间的分配与回收
  2. 实现文件的按名存取：名字空间--映射-->磁盘空间
  3. 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
  4. 提供与IO系统的统一接口
  5. 实现文件信息的共享，并提供文件的保护、保密手段
  6. 提高文件系统的性能

# 内容提要

- 文件系统基本概念
  - 文件系统的概念
  - 文件系统的模型
  - 文件的概念
  - 目录的概念
- 文件系统实现方法
- 文件系统实例分析

# 文件系统模型的三个层次 (1/3)

## 一、文件系统的接口(最接近用户的)

- 为了方便用户使用操作系统，文件系统通过向用户提供两种类型的接口
  - 命令行接口
    - 这是指的是作为用户何文件系统的交互的接口
    - 用户可以通过键盘终端键入命令，取得文件系统的服务
  - 程序接口
    - 这是作为用户程序何文件系统的接口
    - 用户程序可以通过系统调用的形式来取得文件系统的服务

# 文件基本操作

- 创建文件
  - 删除文件
  - 打开文件
  - 关闭文件
  - 读文件
  - 写文件
  - 修改文件名
  - 设置文件的读写位置
1. Create
  2. Delete
  3. Open
  4. Close
  5. Read
  6. Write
  7. Append
  8. Seek
  9. Get attributes
  10. Set Attributes
  11. Rename

# 文件系统模型的三个层次 (2/3)

## 二、对象操作管理的软件集合

- 这是文件管理系统的核心部分。文件系统的功能大多数是在这一层实现的，其中包括：
  - 对文件存储空间的管理
  - 对文件目录的管理
  - 用户将文件的逻辑地址转换伟物理地址的机制
  - 对文件读和写的管理
  - 以及对文件的共享和保护的功能

# 文件系统模型的三个层次 (3/3)

## 三、文件系统管理的对象及其属性

### (1) 文件

它作为文件管理的直接对象

### (2) 目录

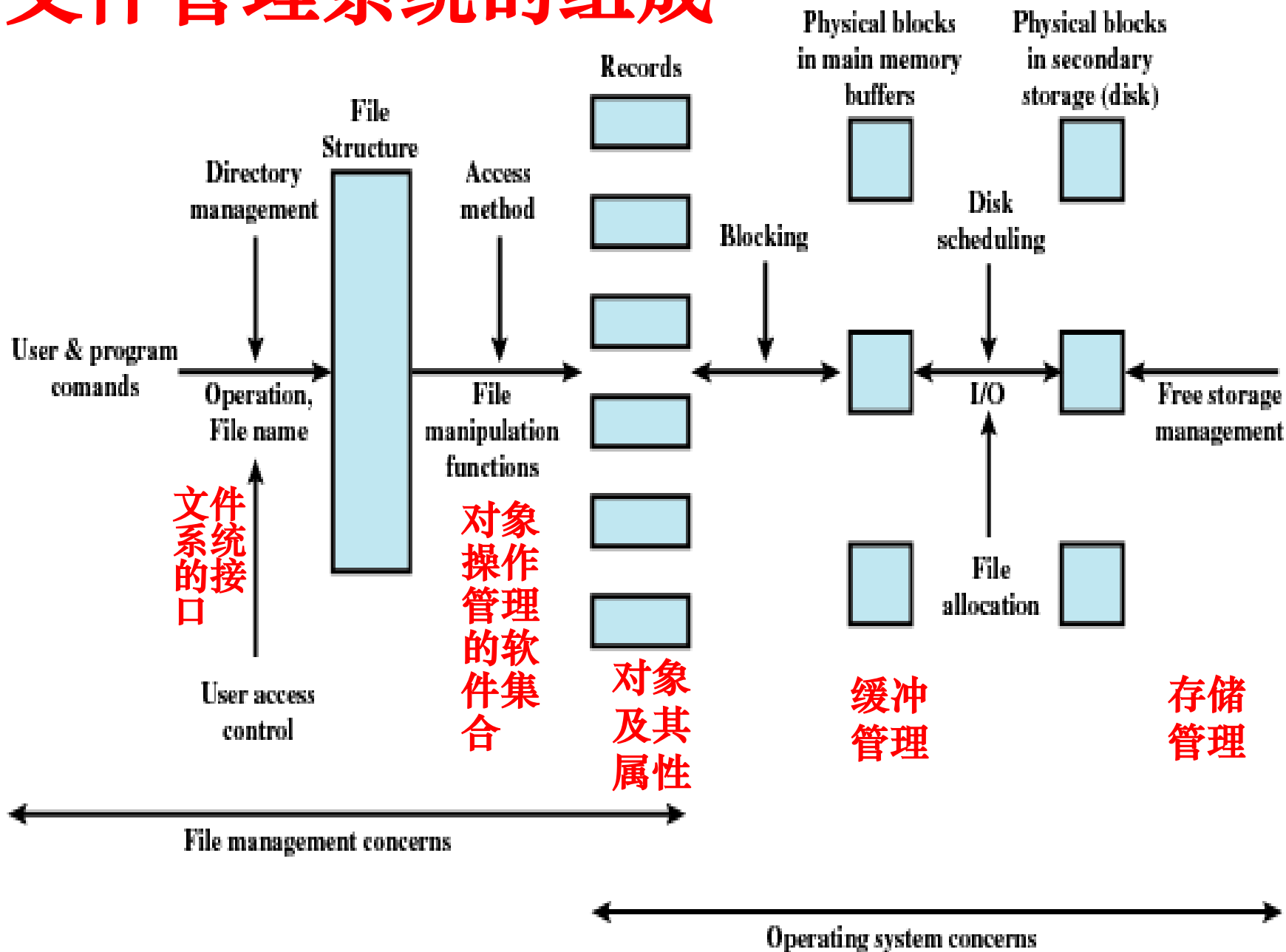
为了方便用户对文件的存取何检索，在文件系统中必须配置目录，每个目录项中，必须包含文件名及文件所在的物理地址（或指针）

对目录的组织和管理是方便用户何提高对文件存取速度的关键

### (3) 磁盘存储空间

文件和目录必定占用存储空间，对这部分空间的高效管理，不仅能够提高外存的利用率，而且能够提高对文件的检索速度

# 文件管理系统的组成





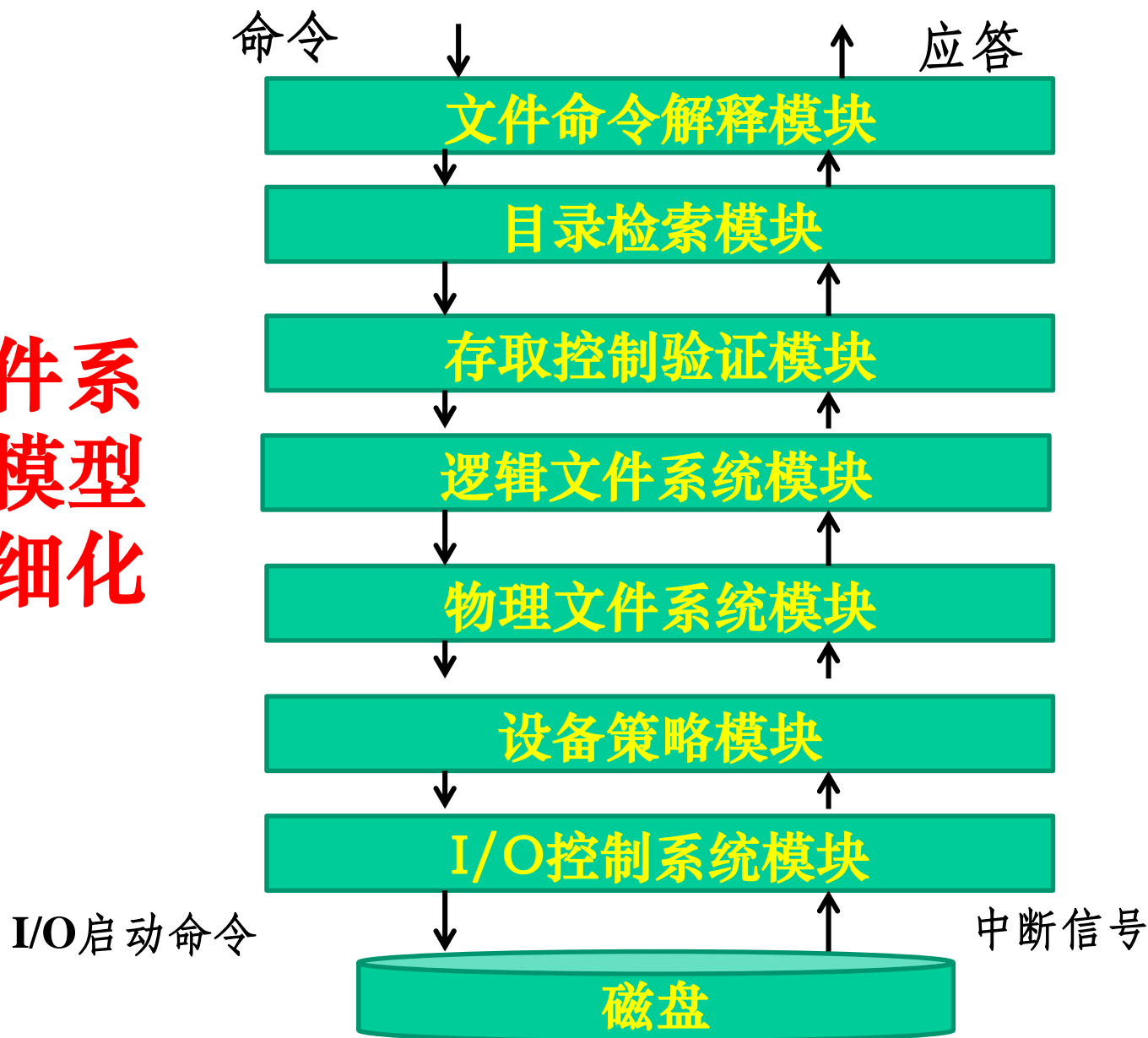
# 文件系统必须解决的几个主要问题

- 提供与I/O的统一接口
- 命名的冲突和文件的共享
- 提供合适的存取方法
- 如何有效的分配磁盘的存储空间
- 文件系统的执行效率
  - 文件系统在操作系统接口中占的比例最大
  - 用户使用操作系统的感觉在很大程度上取决于对文件系统的使用效果

# 理想文件系统应具有的特性

- 具有对用户来说尽可能透明的机制
- 有效的实现各种文件操作的命令
- 尽可能达到对文件存储装置的独立性
- 文件结构和存取的灵活性和多样性
- 有效的分配文件存储器的存储空间
- 存储在文件中的信息的安全性
- 能方便的共享公用的文件

# 文件系统模型的细化



# 内容提要

- 文件系统基本概念
  - 文件系统的概念
  - 文件系统的模型
  - 文件的概念
  - 目录的概念
- 文件系统实现方法
- 文件系统实例分析

# 文件是什么？ ---连续的逻辑存储空间

- 文件是一种抽象机制
  - 它提供了一种把信息保存在磁盘等存储设备上，并且便于以后访问的方法
  - 抽象性体现在用户不必关心具体的实现细节
- 可以视为一个单独的连续的逻辑地址空间
  - 其大小即为文件的大小
  - 与进程的地址空间无关

# 文件是什么？ ---文件名表示的一组字节序列

- 所谓文件是指一组带标识（标识即为文件名）的、在逻辑上有完整意义的**数据项的序列**
  - 数据项：构成文件内容的基本单位（单个字节，或多个字节）
  - 各数据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
- 文件包括两部分
  - 文件体：文件本身的内容
  - 文件说明：文件存储和管理的相关信息
    - 如：文件名、文件内部标识、文件存储地址、访问权限、访问时间等

# 文件是什么？ ---抽象存储空间和设备

- 文件的本质是一组字节序列
  - 源于用户程序对所输入处理的原始数据和输出结果的“长期”（持久化）保存的需求，并按一定“格式”（规范）呈现
    - 所有“需要长期保存”的文件都按某种组织形式存放（映射）到磁盘中
    - 所有的IO设备（字符设备，块设备以及网络设备）都可以看作为字节序列的载体。因此，所有的IO设备也可以抽象为文件进行表达
  - 这就是现代OS中“一切皆文件”的含义
    - 因此所有的I/O设备，包括磁盘、键盘、鼠标、显示器都可以看成是文件
    - 所有的I/O设备都是文件这一抽象概念的具体表现

# 文件管理的需求

## ■ 用户视角（使用逻辑文件）

- 用户关心文件中要使用的数据
  - 不关心具体的存放形式（和位置）
- 关心的是文件系统所提供的对外的用户接口
  - 包括文件如何命名、如何保护、如何访问（创建、打开、关闭、读、写等）；

## ■ 操作系统视角（组织和管理物理文件）

- 文件的描述和分类，关心的是如何实现与文件有关的各个功能模块
  - 包括如何来管理存储空间、文件系统的布局、文件的存储位置、磁盘实际运作方式（与设备管理的接口）等



# 文件名

- 当一个文件被创建时，必须给它指定一个名字
  - 用户通过文件名来访问文件
- 命名规则：文件名是一个有限长度的字符串
  - 一个文件名一般由两部分组成，中间用句点隔开
  - 文件名.扩展名
    - 文件名的长度一般不超过8个字符，但很多系统支持长文件名，如255个字符。
    - 文件名可以由字母、数字和特殊字符组成，有的系统区分字母的大小写（如Unix），有的系统则不区分（如Windows）

# 文件类型

- 程序所处理的数据的种类繁多
  - 如数值数据、字符数据、二进制数据，以及以此为基础的具有各种组织形式的数据
- 文件的分类
  - 按性质和用途：系统文件、用户文件、库文件
  - 按数据形式：源文件、目标文件、可执行文件
  - 按对文件实施的保护级别分：只读文件、读写文件、执行文件、不保护文件
  - 按逻辑结构分：有结构文件、无结构文件
  - 按文件的物理结构分：顺序文件、链接文件、索引文件

# 例：UNIX的文件类型

UNIX按性质和用途将文件分为：

- 普通文件：即用户自己建立的文件，包含了用户的信息，一般为ASCII或二进制文件
- 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，用户模仿串行IO设备，例如终端、打印机、网卡等
  - 块设备文件：磁盘、磁带等
- 管道文件
- 套接字

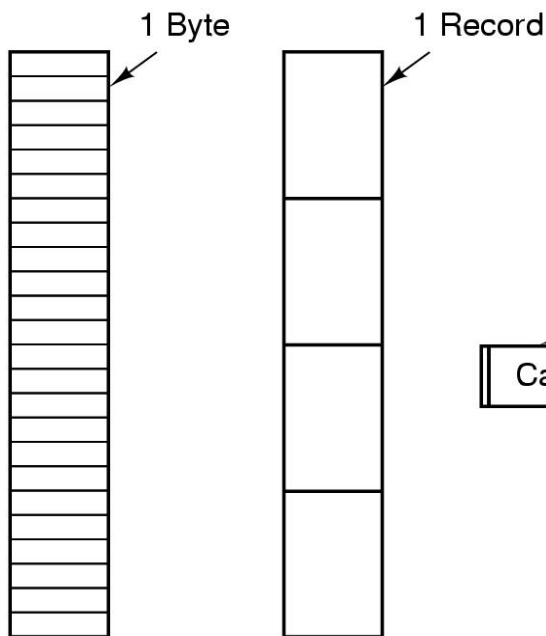
# 常见的文件类型

file type	usual extension	function
executable	exe, com, bin or none	read 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, rrf, doc	various word-processor formats
library	lib, a, so, dll, mpeg, mov, rm	libraries of routines for programmers
print or view	arc, zip, tar	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	binary file containing audio or A/V information

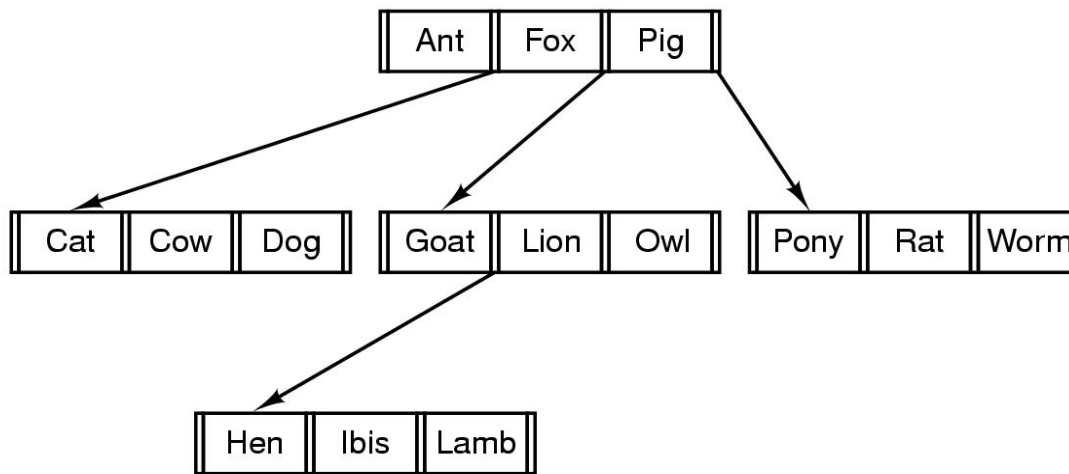
# 文件的逻辑结构

这是从用户角度看文件，由用户的访问方式确定的  
这里给出了三种逻辑结构，还可以组织成堆、顺序、索引、散列等结构

- 第一种是以字节为单位的流式结构
- 第二种是一种记录式文件结构
- 最后一种是树形结构。



(a) 字节序列 (b) 记录序列



(c) 树

# 典型的文件结构、文件存取方式及存储介质

- **流式**文件：构成文件的基本单位是字符。文件是有逻辑意义、无结构的一串字符的集合；
- **记录式**文件：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
- 文件存取方式：
  - **顺序存取**（访问）
  - **随机存取**：提供读写位置（当前位置）
- 文件的存储介质
  - 典型的**存储介质**：磁盘(包括SSD)、磁带、光盘等
  - **物理块**（块block、簇cluster）
    - 数据存储、传输和分配的单位，存储设备通常划分为大小相等的物理块，统一编号

# 内容提要

- 文件系统基本概念
  - 文件系统的概念
  - 文件系统的模型
  - 文件的概念
  - 目录的概念
- 文件系统实现方法
- 文件系统实例分析

# “目录”的提出

- 文件太多了怎么办？

- 不同的应用程序有不同类型的文件，不同的用户有不同的文件，如何对它们进行组织、分类？

- 如何对文件进行管理？

- 当用户需要访问某个文件时，如何根据这个文件名迅速地定位到相应的文件，从而对文件的属性和内容进行各种操作？

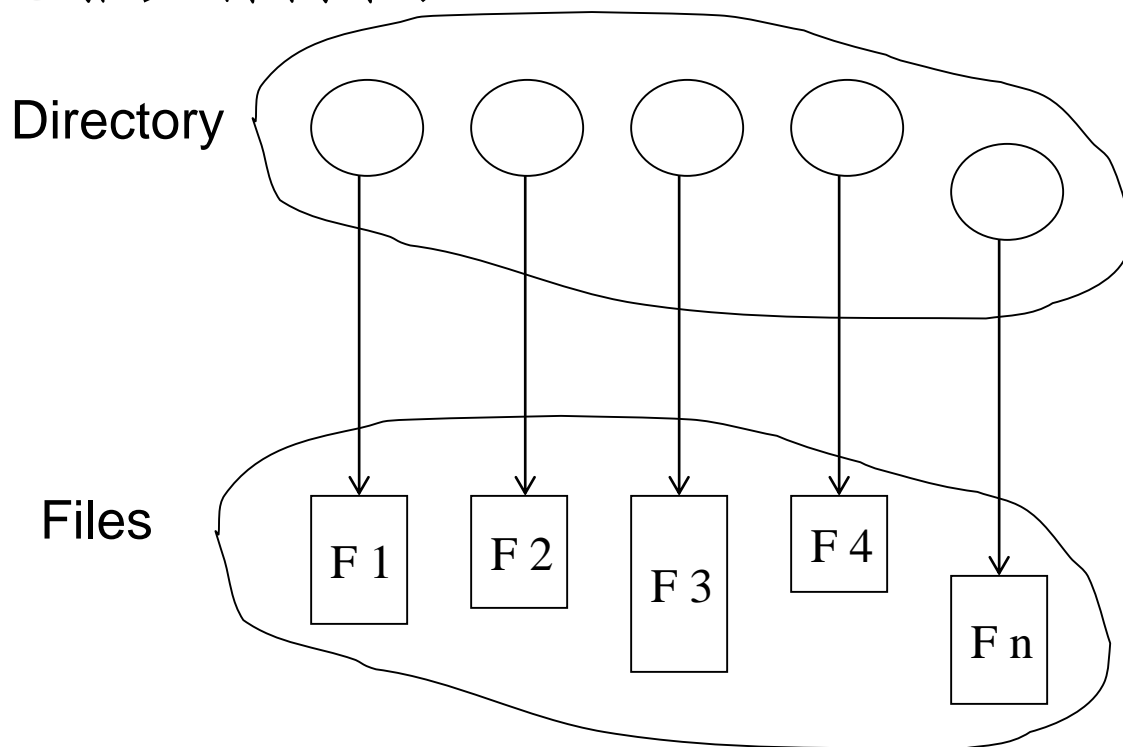


# “目录”的提出

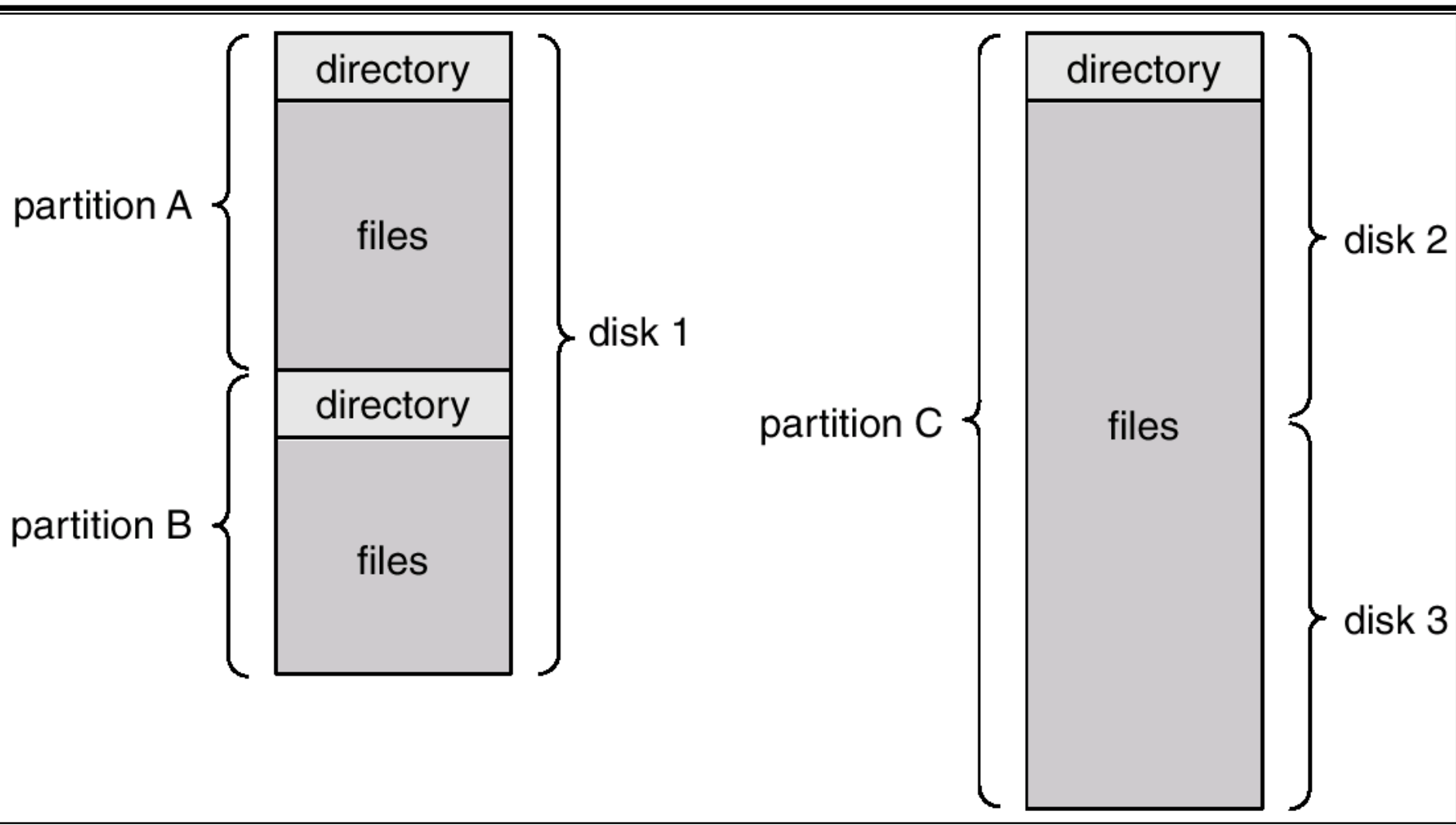
- **效率**：要能迅速定位一个文件
- **命名**：要方便用户
  - 重名：对不同的用户文件，可使用相同名字
  - 别名：对同一个文件，可有多多个不同的名字
- **分组功能**：
  - 能够把不同的文件按照某种属性进行分组
    - 如某门课程的所有文件可分为讲义、作业、参考资料等不同的组

# 目录管理

- 目录是由文件说明索引组成的用于文件检索的特殊文件
  - 文件目录的内容主要是文件访问和控制的信息（不包括文件内容）



# 典型的文件系统的组织



# 目录内容

- 目录的内容是文件属性信息
  - 其中的一部分是用户可获取的
- 基本信息
  - 文件名：字符串
    - 在不同系统中允许不同的最大长度，可以修改
  - 别名的数目
    - 有些系统允许同一个文件有多个别名(alias)
  - 文件类型：可有多种不同的划分方法，如：
    - 有/无结构（记录文件，流式文件）
    - 内容（二进制，文本）
    - 用途（源代码，目标代码，可执行文件，数据）
    - 属性 attribute（如系统，隐含等）
    - 文件组织（如顺序，索引等）

# 目录内容（续）

- 地址信息
  - 存放位置：哪个设备或文件卷volume，及各存储块位置
  - 文件长度（当前和上限）：以字节、字或存储块为单位
- 访问控制信息
  - 文件所有者（属主）：通常是创建文件的用户，也可改变；
  - 访问权限（用户访问方式）：读、写、执行等
- 使用信息
  - 创建时间
  - 最后一次读访问的时间和用户
  - 最后一次写访问的时间和用户

# 目录操作

1. Create
2. Delete
3. Opendir
4. Closedir

1. Readdir
2. Rename
3. Link
4. Unlink

# 文件目录分类

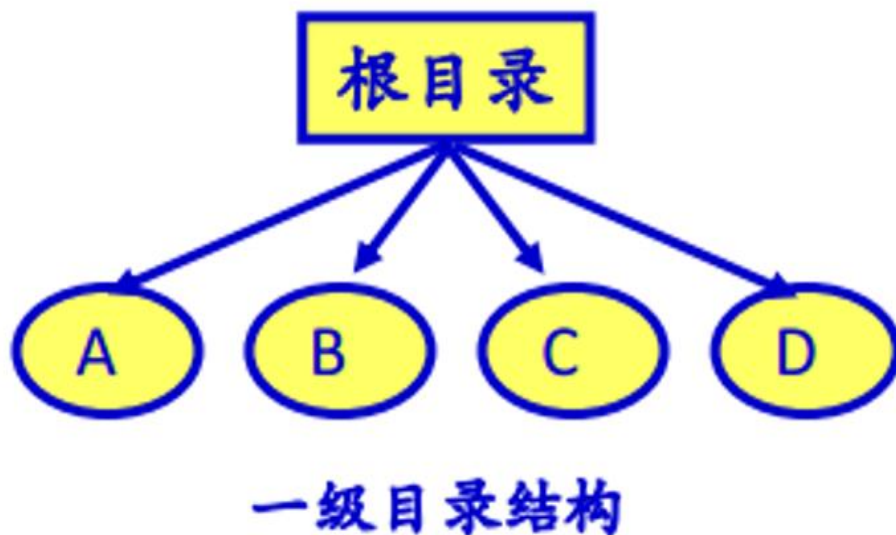
## ■ 目录分类

- 单级文件目录
- 二级文件目录
- 多级文件目录

# 单级文件目录

文件目录的每个表目应包含：

1. 文件的符号名
2. 文件所在物理地址
3. 文件结构信息
4. 存取控制信息
5. 管理信息



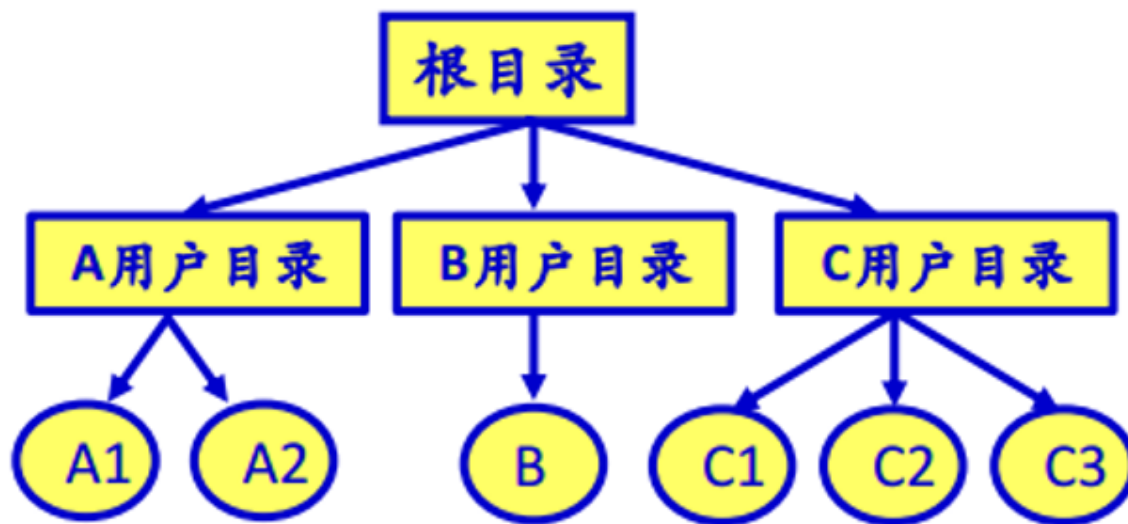


# 特点

- 结构简单（只在早期的个人计算机上使用过）
- 文件多时，目录检索时间长
- 有命名冲突：
  - 如多个文件有相同的文件名（不同用户的相同作用的文件）
  - 或一个文件有多个不同的文件名（不同用户对同一文件的命名）
- 不便于实现共享

# 两级目录

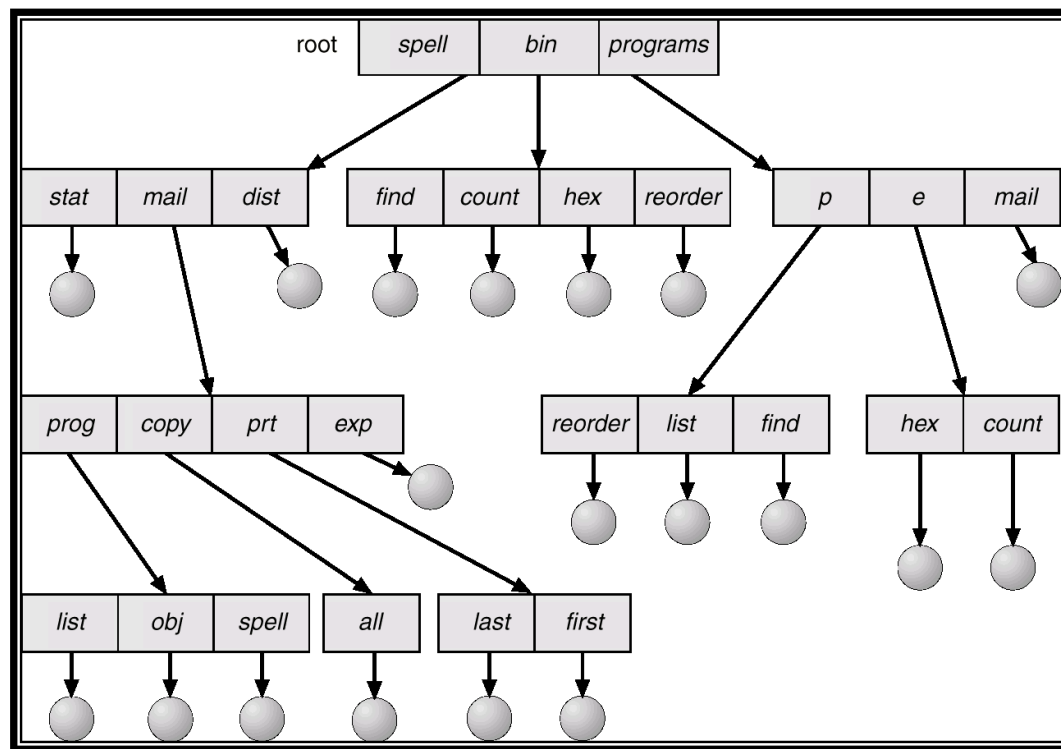
- 在根目录（第一级目录）下，每个用户对应一个目录（第二级目录），在用户目录下是该用户的文件，而不再有下级目录。适用于多用户系统，各用户可有自己的专用目录。



二级目录结构

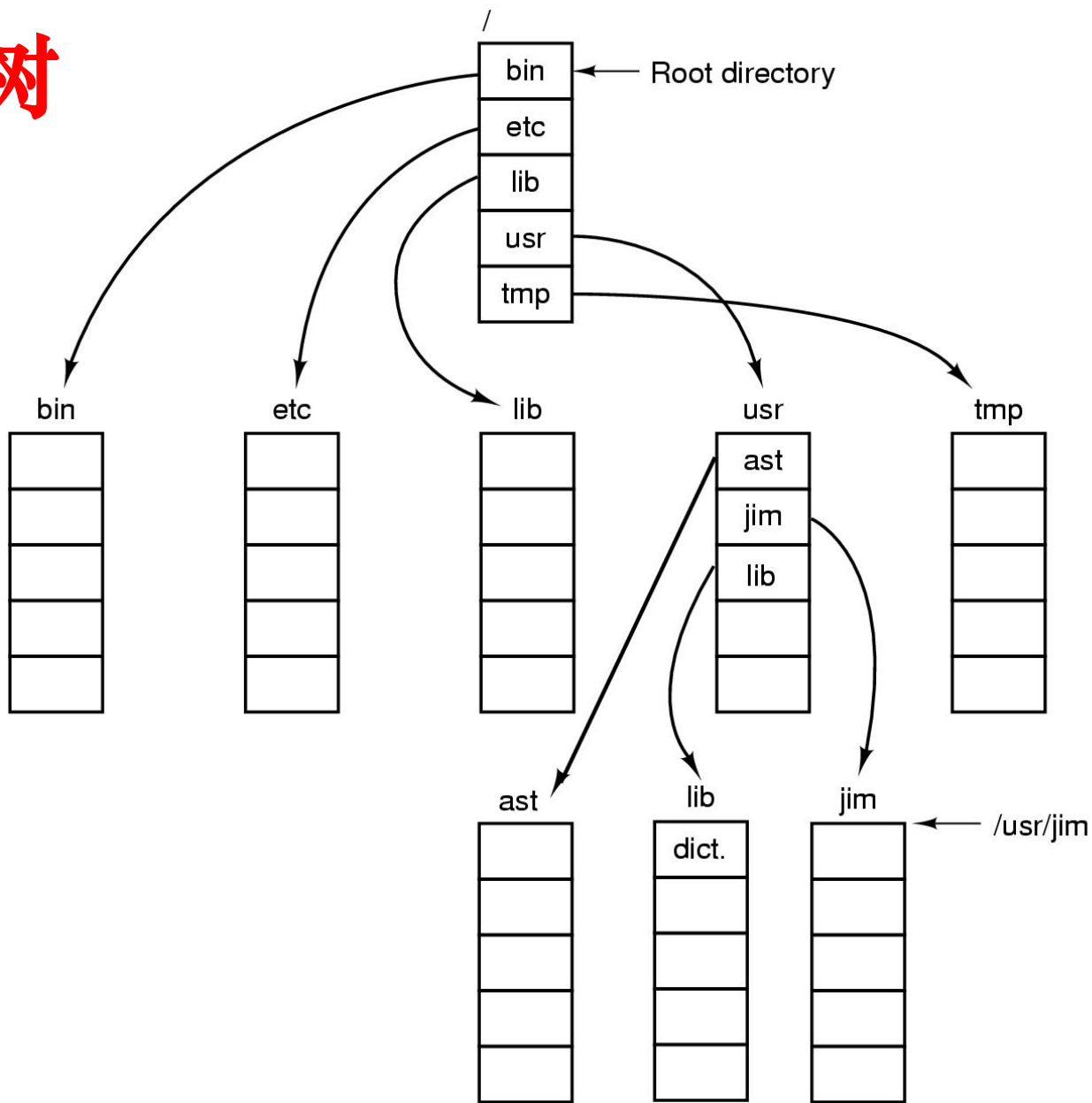
# 多级目录（层次目录）

- 在较高的目录级，其目录表目为下一级目录名以及一个指向其目录的指针。
- 在最后一级目录，这个指针指向文件的物理地址。
- 几乎所有现代文件系统都采用这种方案。



- 在多级目录结构中，如何来指定需要访问的文件？
  - **绝对路径名**：对于每一个文件或目录，可以用从根目录开始依次经由的各级目录名，再加上最终的文件名或目录名来表示（之间用分隔符隔开）**一个文件或目录的绝对路径名是唯一的**。例如：\spell\mail\copy\all
  - **相对路径**：结合当前路径进行使用，比如上个例子中，如果当前目录为 /usr 那么 ast/mailbox 指的就是根目录下的 usr 文件夹中的 ast 文件中的 mailbox 文件
  - **当前目录**：也叫工作目录，用户可以指定一个目录作为当前的工作目录；
  - **上一级目录**：使用 “..” 表示上一级目录，读作 dotdot, 在根目录下 “..” 表示它本身，还是根目录。

# Unix 目录树



# 多级目录特点

## ■ 层次清楚

- 不同性质、不同用户的文件可以构成不同子树，便于管理
- 不同用户的文件可以被赋予不同的存取权限，有利于文件的保护
- 数目较多时，便于系统和用户将文件分散管理，使得文件和目录的层次结构较为清晰。适用于较大的文件系统管理

## ■ 可解决文件重名问题

- 文件在系统中的搜索路径时从根开始到文件名为止的各文件名组成，只要在同一目录下的文件名不发生重复就不会由文件重名而引发混乱

## ■ 查找速度快

- 可为每类文件建立一个子目录，由于对多级目录的查找每次只查找目录的一个子集，所以搜索速度快于一级和二级目录

## ■ 目录级别太多时，会增加路径检索时间

# 内容提要

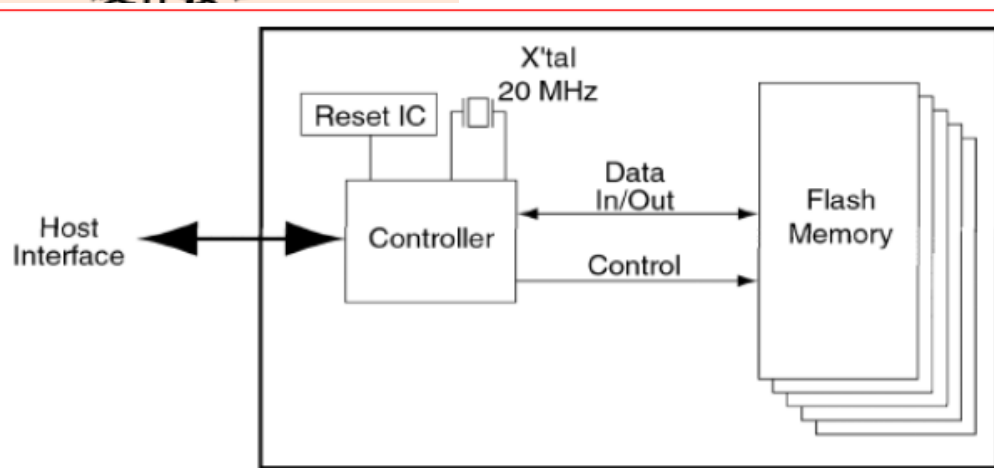
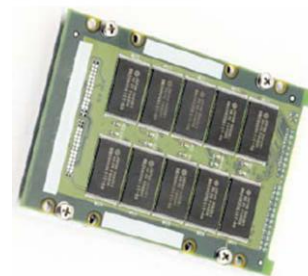
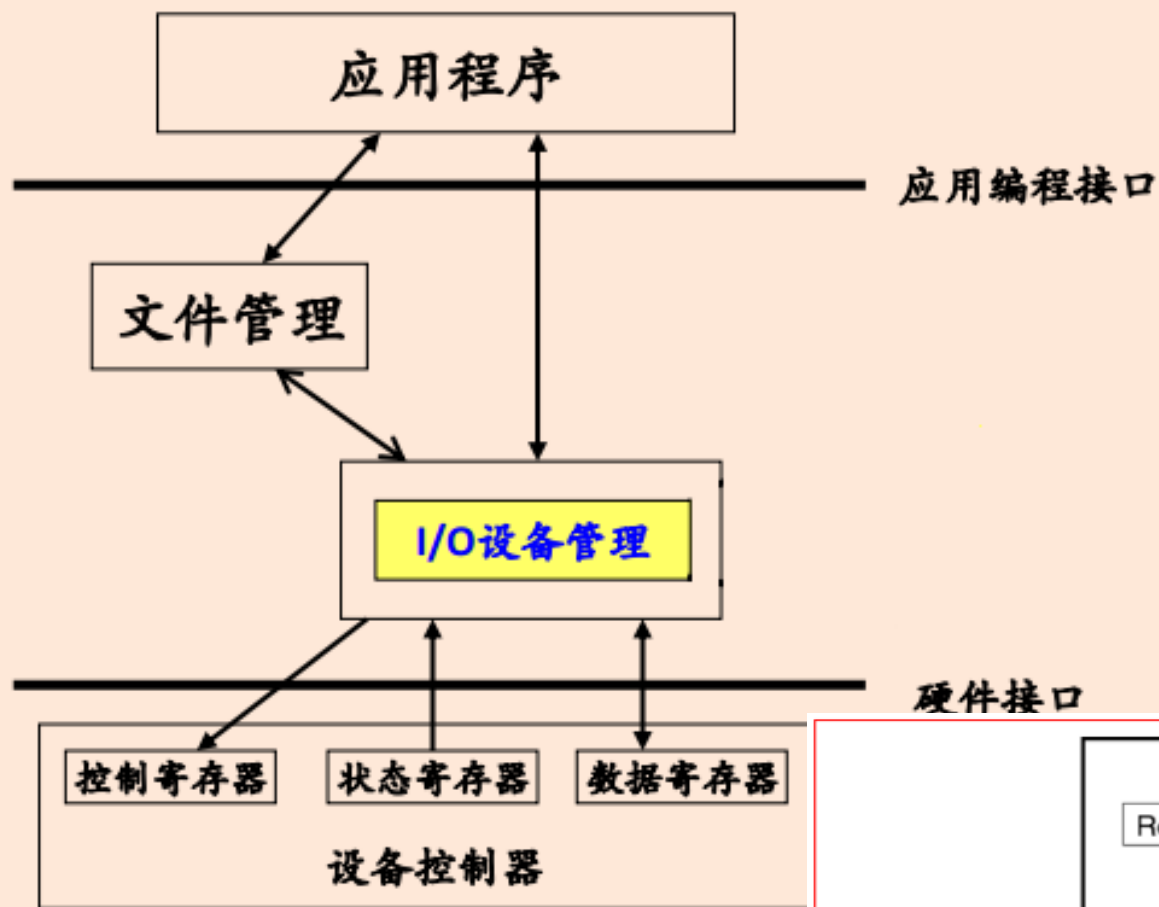
- 文件系统基本概念
- 文件系统实现方法
- 文件系统实例分析

# 内容提要

- 文件系统基本概念
- 文件系统实现方法
  - 文件的实现
  - 目录的实现
  - 硬链接和软链接
  - 内存中的文件系统
  - 文件的保护
  - 文件系统的性能
- 文件系统实例分析

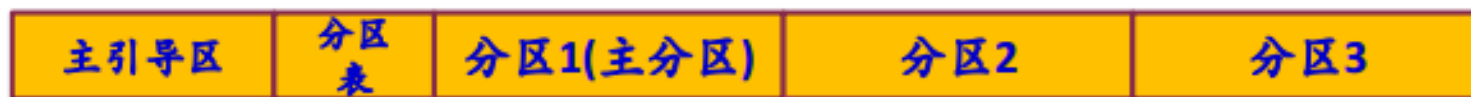


# 文件系统与I/O管理示意



# 磁盘上文件系统的布局

整块磁盘



UNIX文件系统布局



文件卷(逻辑分区)



Windows的FAT文件系统布局

# 内容提要

- 文件系统基本概念
- 文件系统实现方法
  - 文件的实现
  - 目录的实现
  - 硬链接和软链接
  - 内存中的文件系统
  - 文件的保护
  - 文件系统的性能
- 文件系统实例分析

# 文件的实现需解决以下两个问题

- 如何来描述一个文件（FCB）？
  - 如何来记录文件的各种管理信息？
- 如何来存放文件（分配磁盘块）？
  - 即如何把文件的各个连续的逻辑块存放在磁盘上的空闲物理块当中？
  - 如何来记录逻辑块与物理块之间的映射关系？

# 一个典型的文件控制块FCB 示意

文件名	
文件所在的物理地址	
记录长度	记录个数
文件占有者的存取权限	
其它用户的存取权限	
.....	
.....	
文件建立时间	
上次存取时间	
暂存文件/永久文件	

# FCB与文件属性

## 文件控制块(FCB, File Control Block)

- 为管理文件而设置的数据结构，保存管理文件所需的所有有关信息（文件属性或元数据）

## 文件常用属性

- 文件名，文件号，文件大小，文件地址，创建时间，最后修改时间，最后访问时间，保护，口令，创建者，当前拥有者，文件类型，共享计数
- 各种标志（只读、隐藏、系统、归档、ASCII/二进制、顺序/随机访问、临时文件、锁）

# 文件控制块(FCB, File Control Block)

## ■ 基本信息

- **文件名**：字符串，通常在不同系统中允许不同的最大长度。可以修改。
- **物理位置**；
- **文件逻辑结构**：有/无结构（记录文件，流式文件）
- **文件物理结构**：（如顺序，索引等）

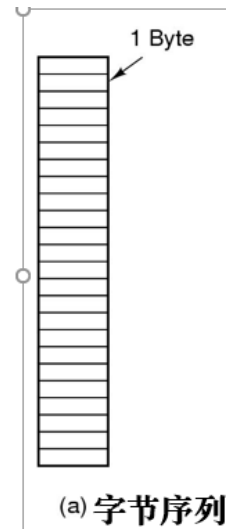
## ■ 访问控制信息

- **文件所有者（属主）**：通常是创建文件的用户，或者改变已有文件的属主；
- **访问权限**（控制各用户可使用的访问方式）：如读、写、执行、删除等；

## ■ 使用信息

- **创建时间**，**上一次修改时间**，当前使用信息等。

# 文件逻辑结构和物理结构



## ■ 文件逻辑结构（文件组织）

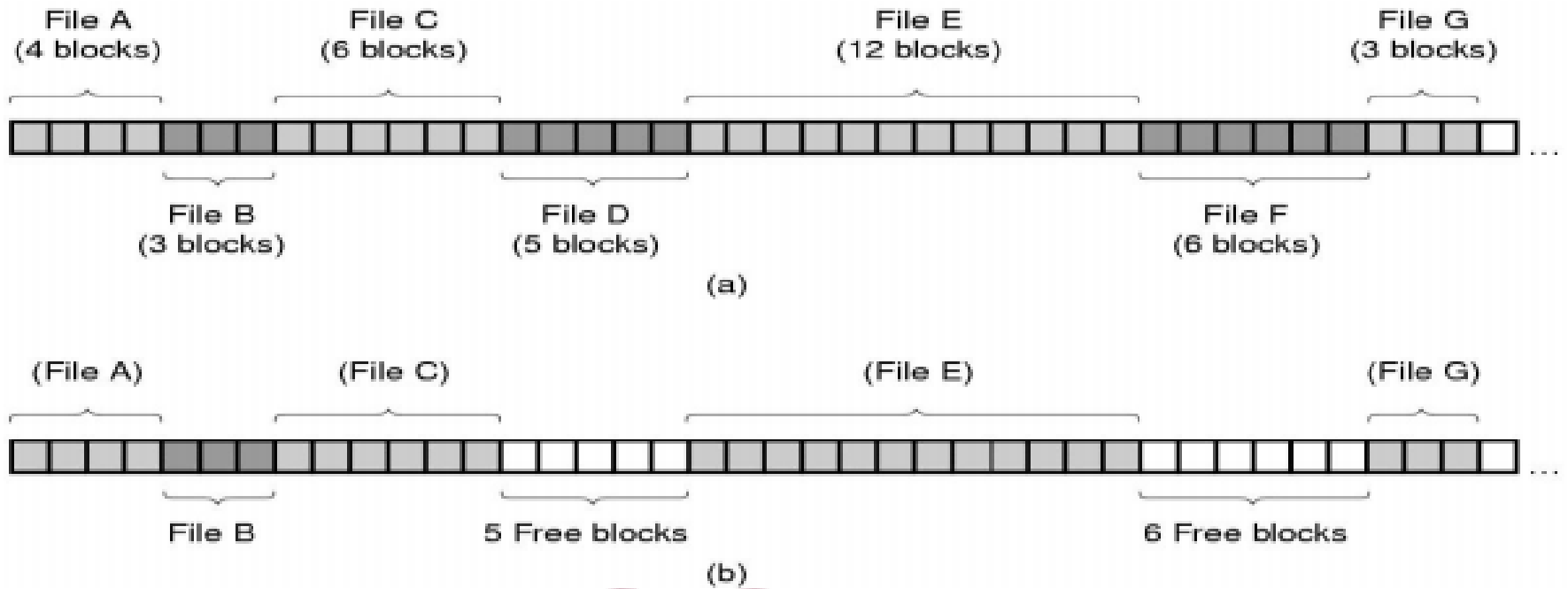
- 提高检索效率；便于修改；降低文件存储费用

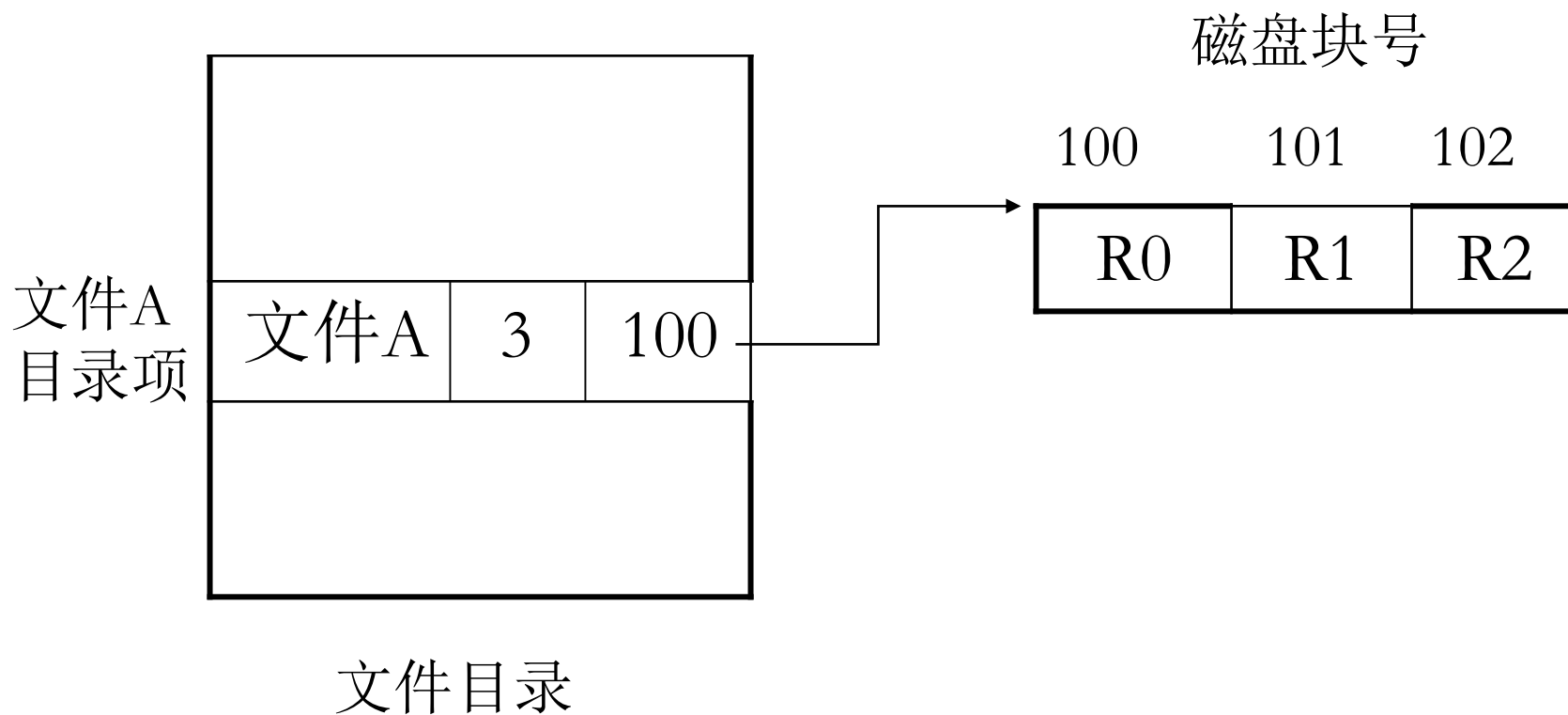
## ■ 文件物理结构

- 文件在存储介质上的存放方式，表示了一个文件在文件存储介质上的位置、链接和编目的方法。
- 主要解决两个问题：
  - 假设一个文件被划分成N块，这N块在磁盘上是怎么存放的？
  - 其地址（块号或簇号）在FCB中是怎样记录的？
- 主要结构：连续结构、索引结构、串联结构



# 连续（顺序）结构





在连续文件结构下，当要存取 $R_i$ 记录时，应如何操作？

# 连续结构的特点

## ■ 优点：

- 结构简单，实现容易，不需要额外的空间开销
- 支持顺序存取和随机存取，顺序存取速度快
- 连续存取时速度较快

## ■ 缺点：

- 文件长度一经固定便不易改变
- 不利于文件的动态增加和修改

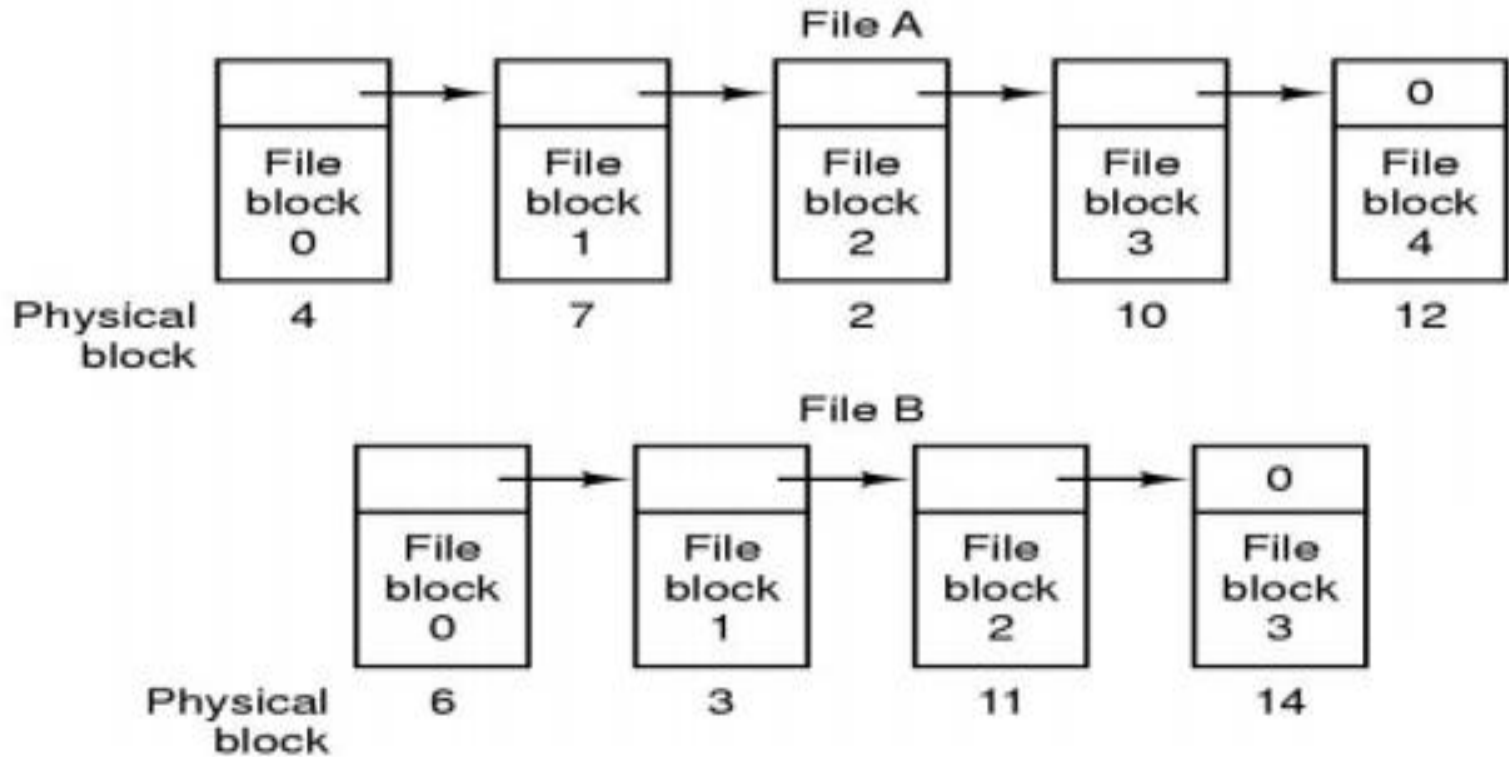
**适用于变化不大的顺序访问的文件**

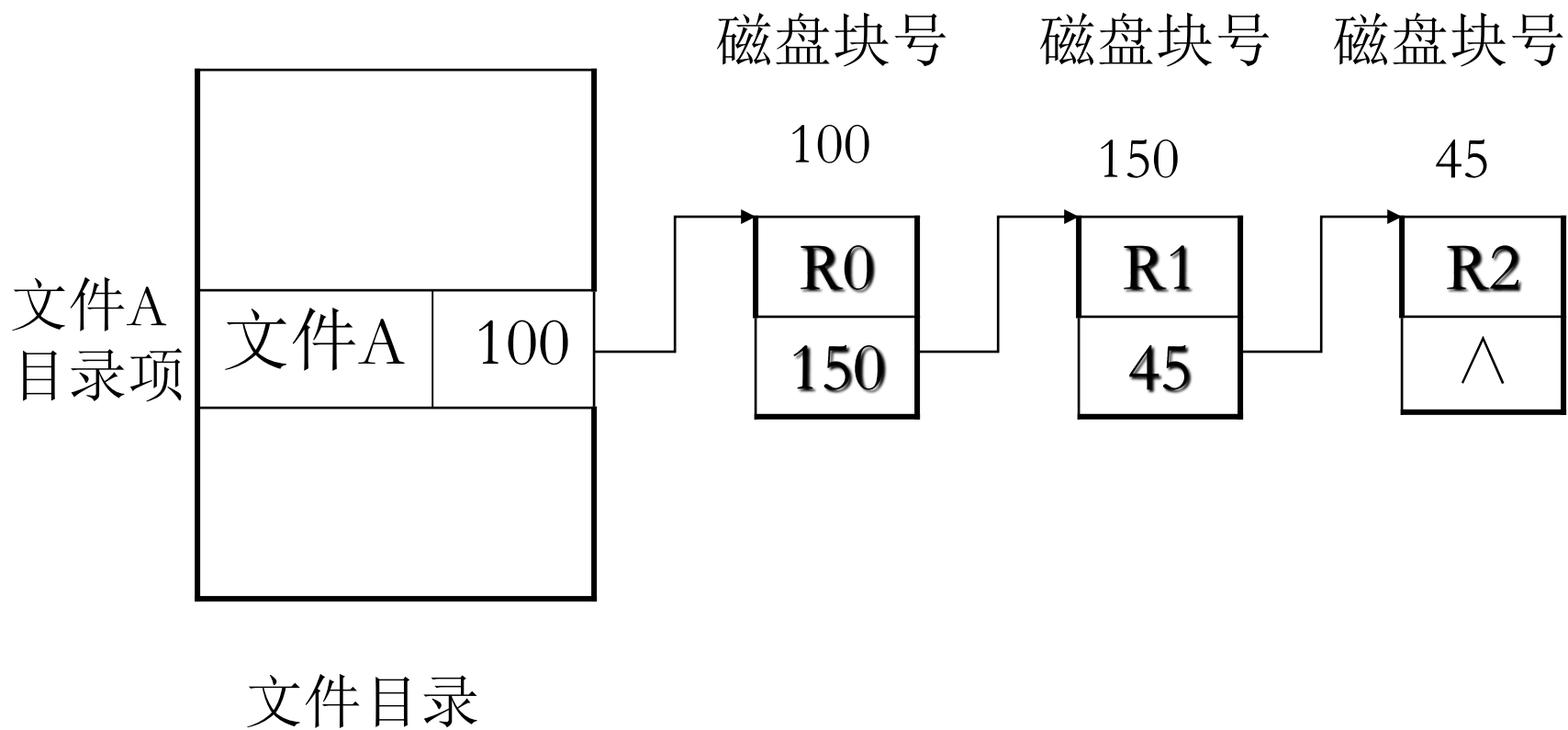
# 串联/链接文件结构

## 什么是串联结构

- 串联文件结构是按顺序由串联的块组成的，即文件的信息按存储介质的物理特性存于若干块中。
- 每个物理块的最末一个字(或第一个字)作为链接字，它指出后继块的物理地址。链首指针存放在该文件目录中。文件的结尾块的指针为“^”，表示文件至本块结束。
- 对于记录式文件一块中可包含一个逻辑记录或多个逻辑记录，也可以若干物理块包含一个逻辑记录。

# 串联（链式）结构





在串联文件结构下，当要存取 $R_i$ 记录时，应如何操作？

# 串联文件的特点

## ■ 优点：

- 空间利用率高；能较好的利用辅存空间。
- 文件动态扩充和修改容易。
- 顺序存取效率高

## ■ 缺点：

- 随机存取效率太低，如果访问文件的最后的内容，实际上是要访问整个文件。
- 可靠性问题，如指针出错；
- 链接指针占用一定的空间。

# 索引结构

## ■ 索引结构：

- 一个文件的信息存放在若干个不连续物理块中
- 系统为每个文件建立一个专用数据结构：索引表，并将这些物理块的块号存放在该索引中。
- 索引表就是磁盘块地址数组，其中第 $i$ 个目录指向文件的第 $i$ 块。

## ■ 索引表应该存放在何处？

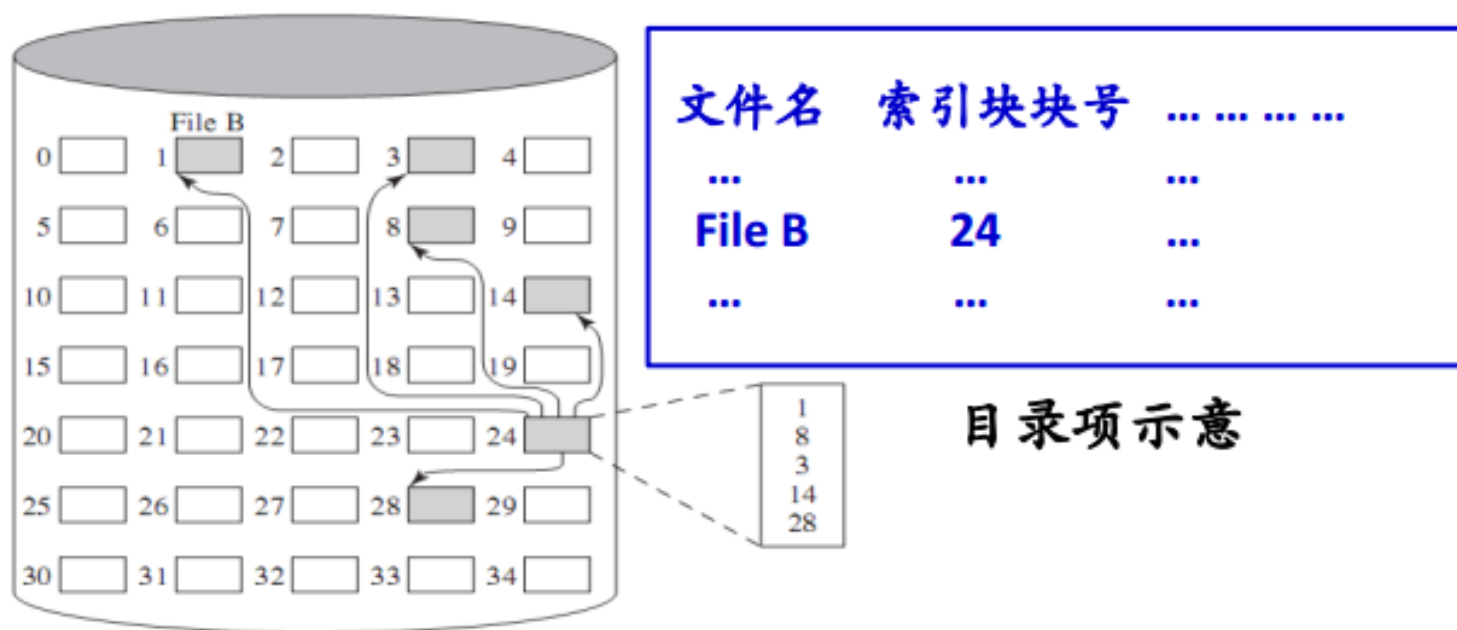
- 这里必须知道每个文件的索引表长度是不一样的，不能存放在FCB中，因此FCB中只记录索引表的地址。



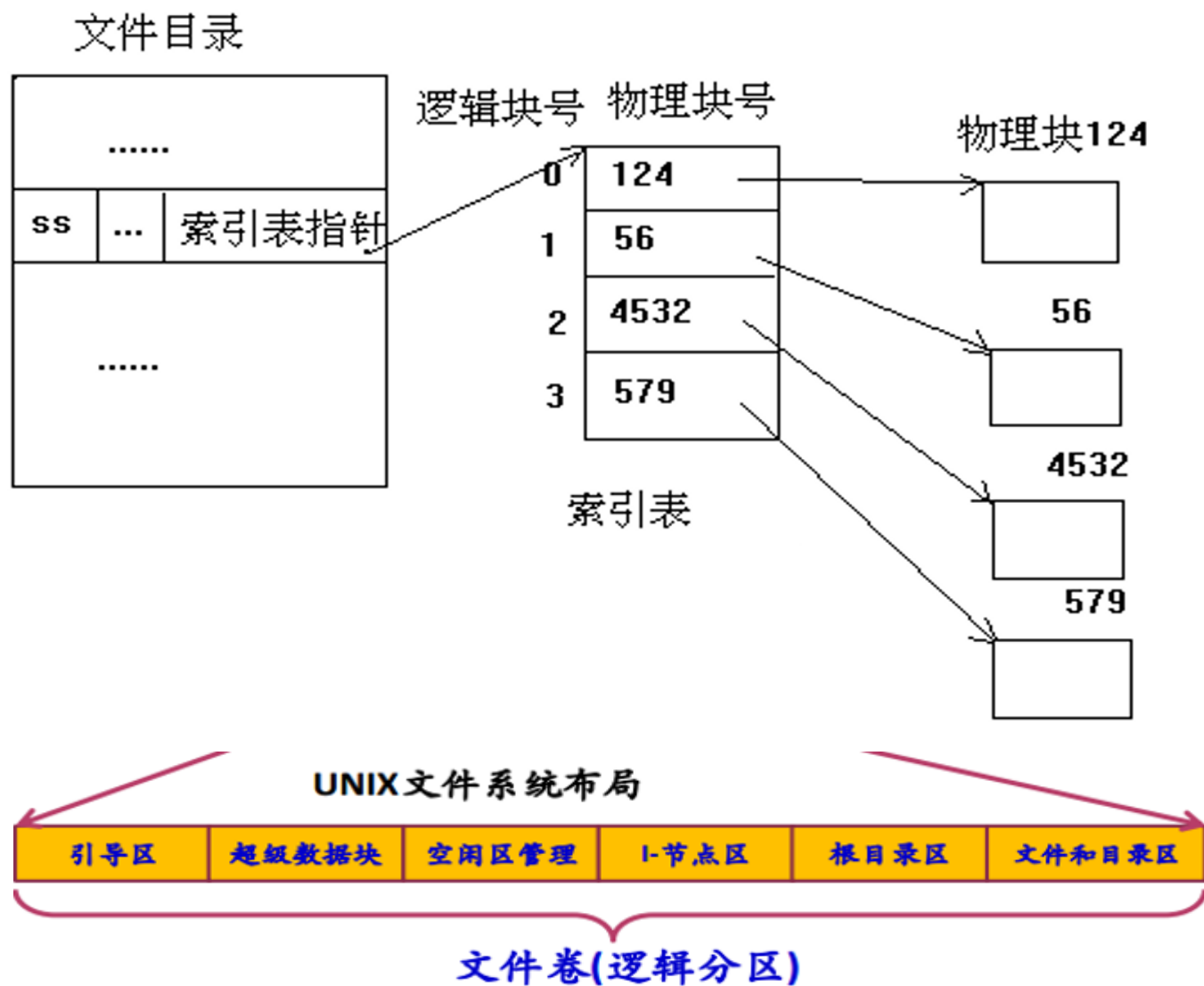
# 索引结构

系统为每个文件建立逻辑块号与物理块号的对照表，称为文件的索引表。文件由数据文件和索引表构成。这种文件称为索引文件。

- **索引表位置**：文件目录中，文件的开头等。
- **索引表大小**：固定大小，非固定大小。



# 分离出索引节点



# 索引文件的操作

- 索引文件在存储区中占两个区
  - 索引区，存放索引表
  - 数据区，存放数据文件本身
- 访问索引式文件需要两步操作
  - 查索引区，由逻辑块号查得物理块号
  - 查数据区，由此物理块号从磁盘上获得所要求的数据

# 索引文件的特点

## ■ 优点：

保持了链接结构的优点，又回避了其缺点：

- 即能顺序存取，又能随机存取
- 满足了文件动态增长、插入删除的要求
- 能充分利用外存空间

## ■ 缺点：

索引表本身带来了系统开销

如：内外存空间，存取时间

# 多个索引表的组织

- 链接模式：
  - 一个盘块一个索引表，多个索引表链接起来
- 多级索引：
  - 将一个大文件的所有索引表（二级索引）的地址放在另一个索引表（一级索引）中
- 综合模式：
  - 直接索引方式与间接索引方式结合

# File1的 i 结点 1235

## 直接索引

	...
	1890
0	124
1	56
2	7981
3	226
4	null
5	null
6	null
7	null
8	null
9	null
10	null
11	null
12	null

文件大小



### 文件目录

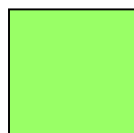
File 1	1235
...	

假设:

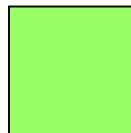
磁盘块大小: 512字节

磁盘块号: 4字节

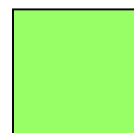
124



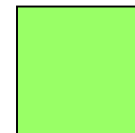
56



7981



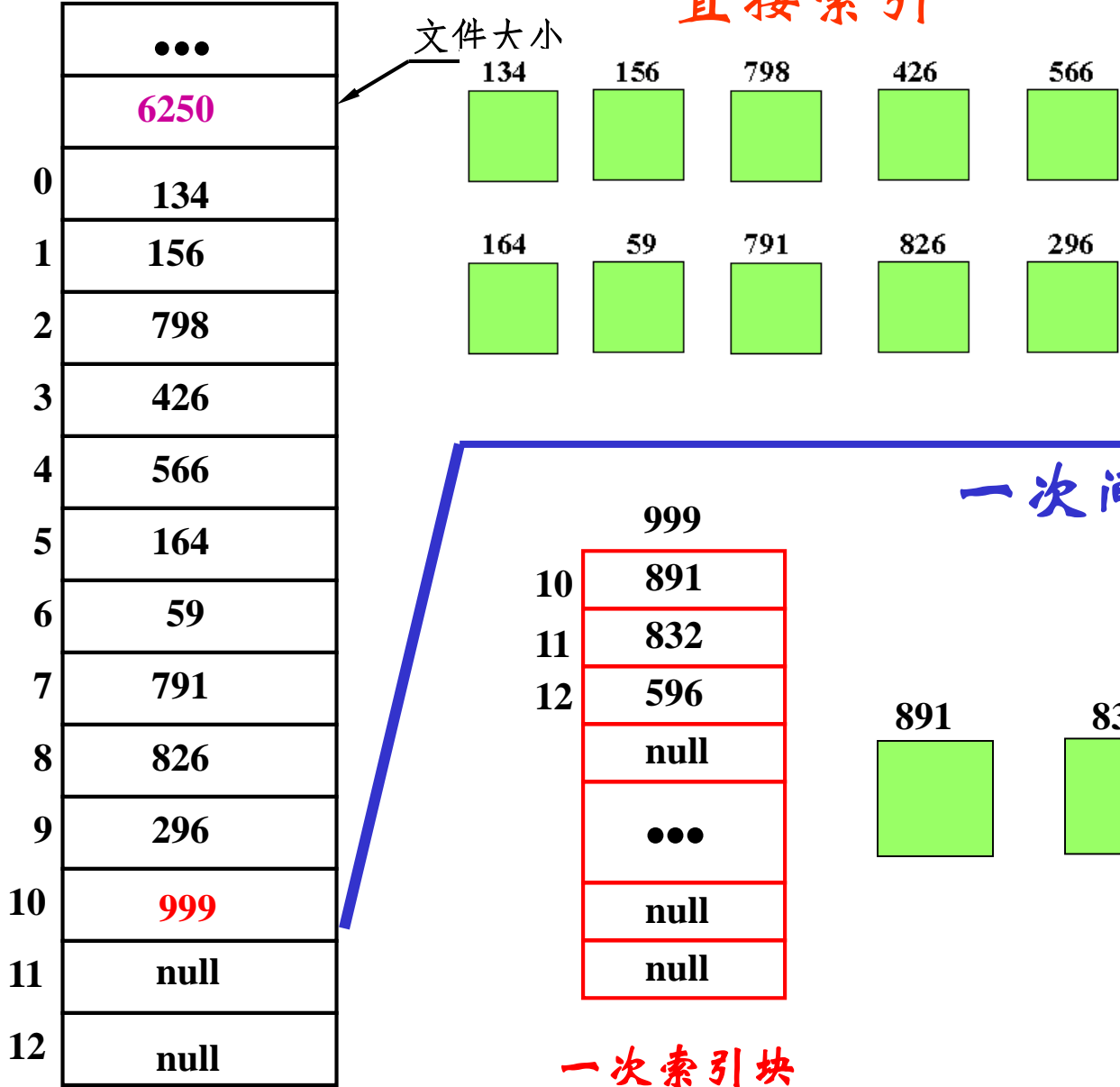
226



直接索引

文件目录

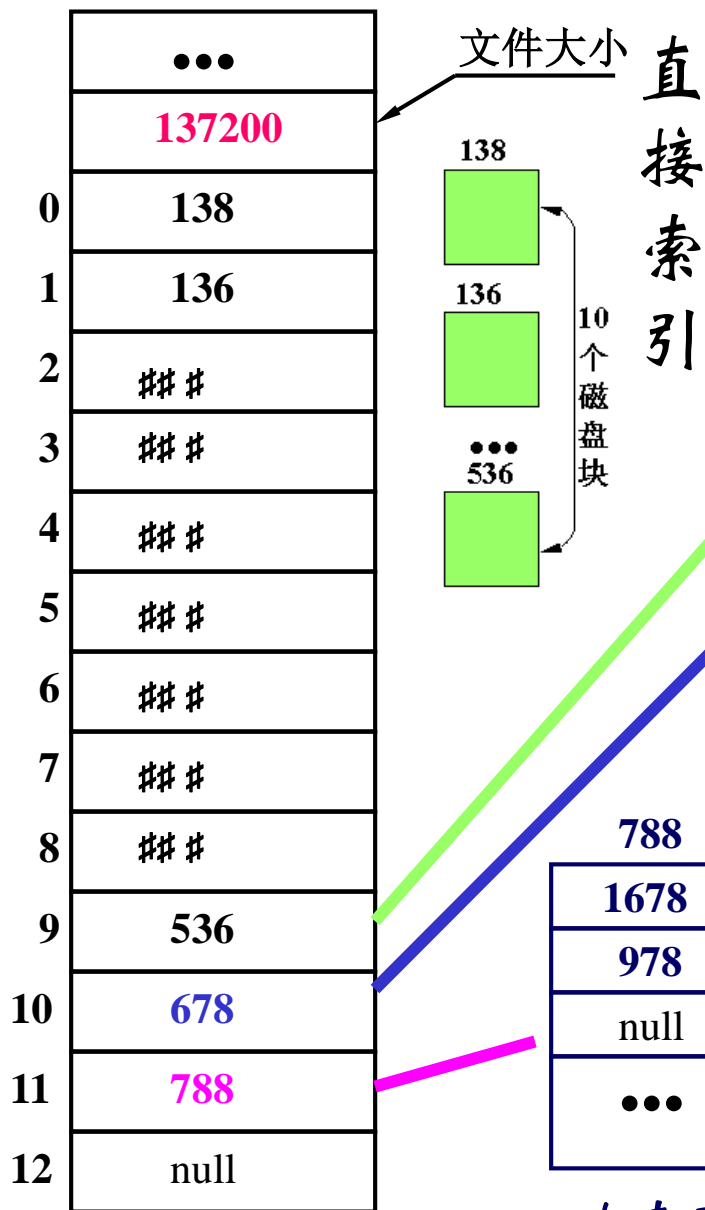
File 1	1235
File 2	896
...	



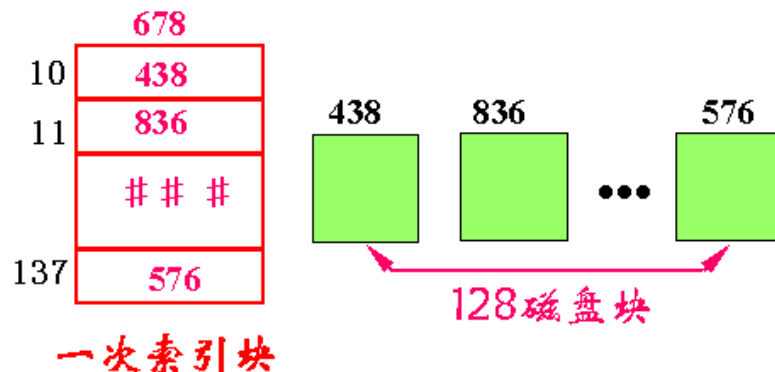
一次间接索引

一次索引块

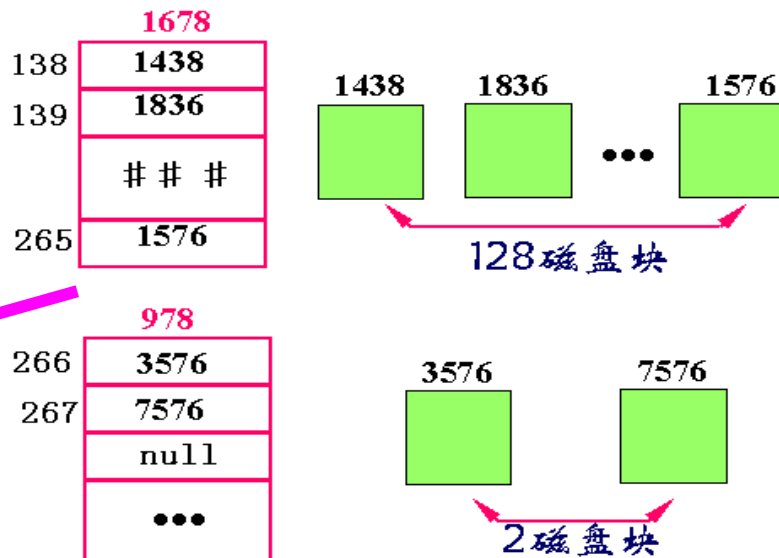
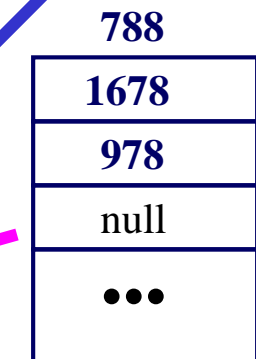
# File3 i 结点 1896



## 一次间接索引



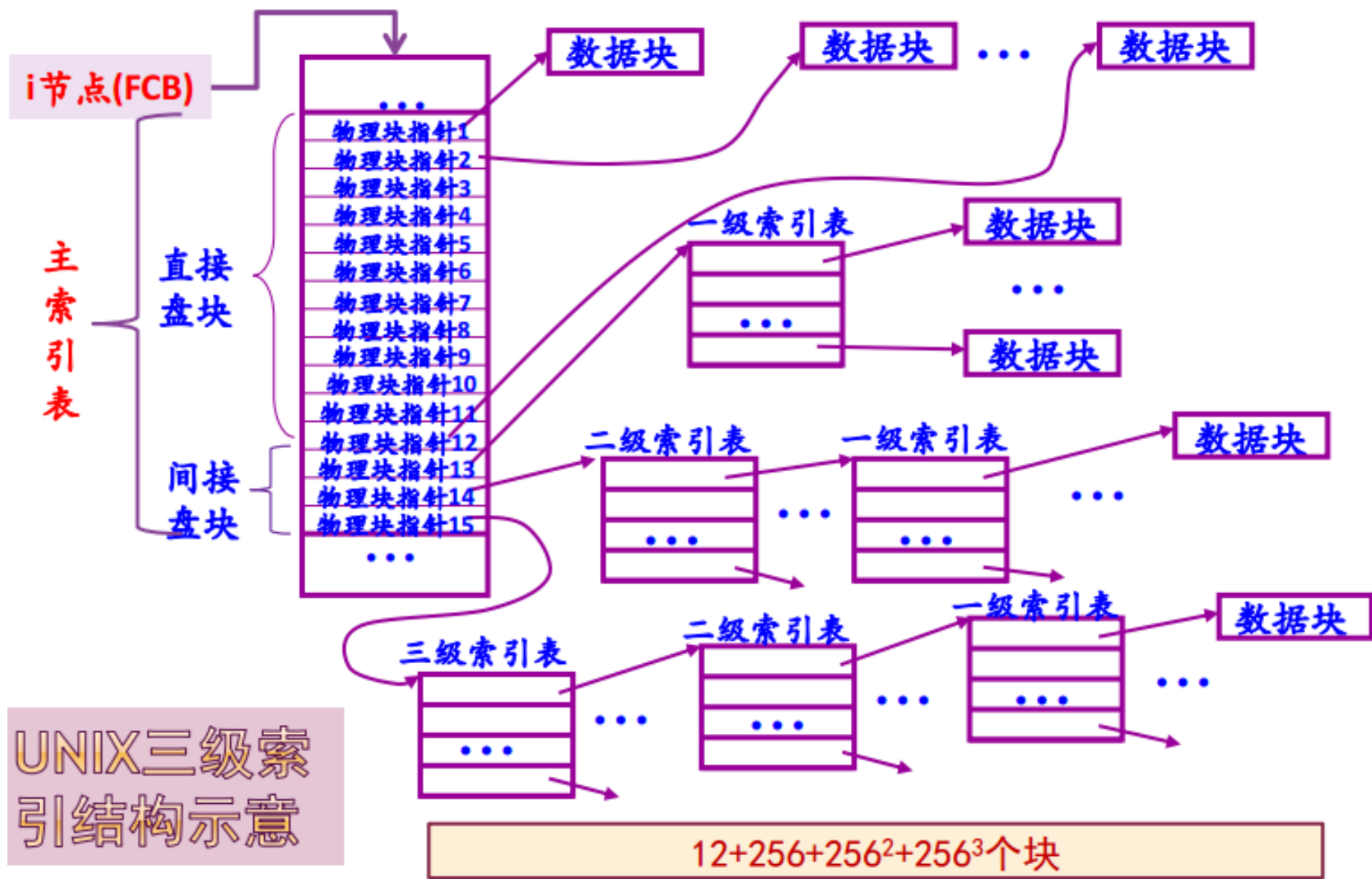
## 二次间接索引



## 文件目录

File 1	1235
File 3	1896





UNIX三级索引结构示意

# 文件物理结构的比较

- **连续结构**：优点是不需要额外的空间开销，只要在文件目录中指出文件的大小和首块的块号即可，对顺序的访问效率很高。适应于顺序存取。缺点是动态地增长和缩小系统开销很大；文件创建时要求用户提供文件的大小；存储空间浪费较大
- **串联结构**：克服了连续文件的不足之处，但文件的随机访问系统开销较大。适应于顺序访问的文件。DOS系统中改造了串联文件的结构，使其克服了串联文件的不足，但增加了系统的危险性
- **索引结构**：既适应于顺序存访问，也适应于随机访问，是一种比较好的文件物理结构，但要有用于索引表的空间开销和文件索引的时间开销。UNIX系统是使用索引结构成功的例子

# 文件存储器存储空间的管理

- 空白文件目录
- 空白物理块链
- 位视图

