第七章 文件管理

7.1 文件和文件系统

7.2 文件的逻辑结构

7.3 文件目录

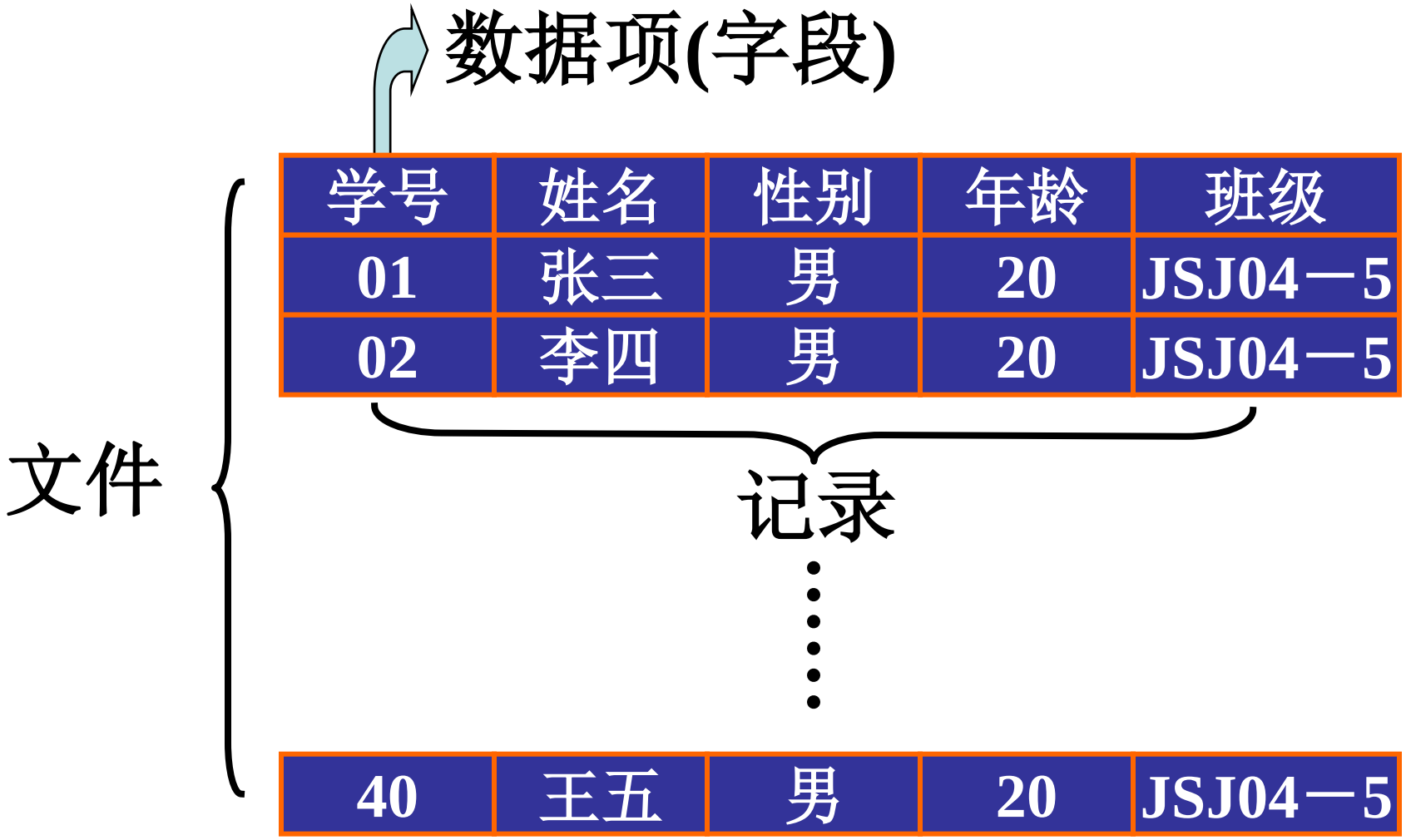
7.4 文件共享

7.5 文件保护

7.1 文件和文件系统

- 现代OS中，通过文件系统组织和管理计算机中存储的大量程序和数据
- 基于文件系统的概念，可以把数据组成分为三级：
  - 数据项
  - 记录
  - 文件

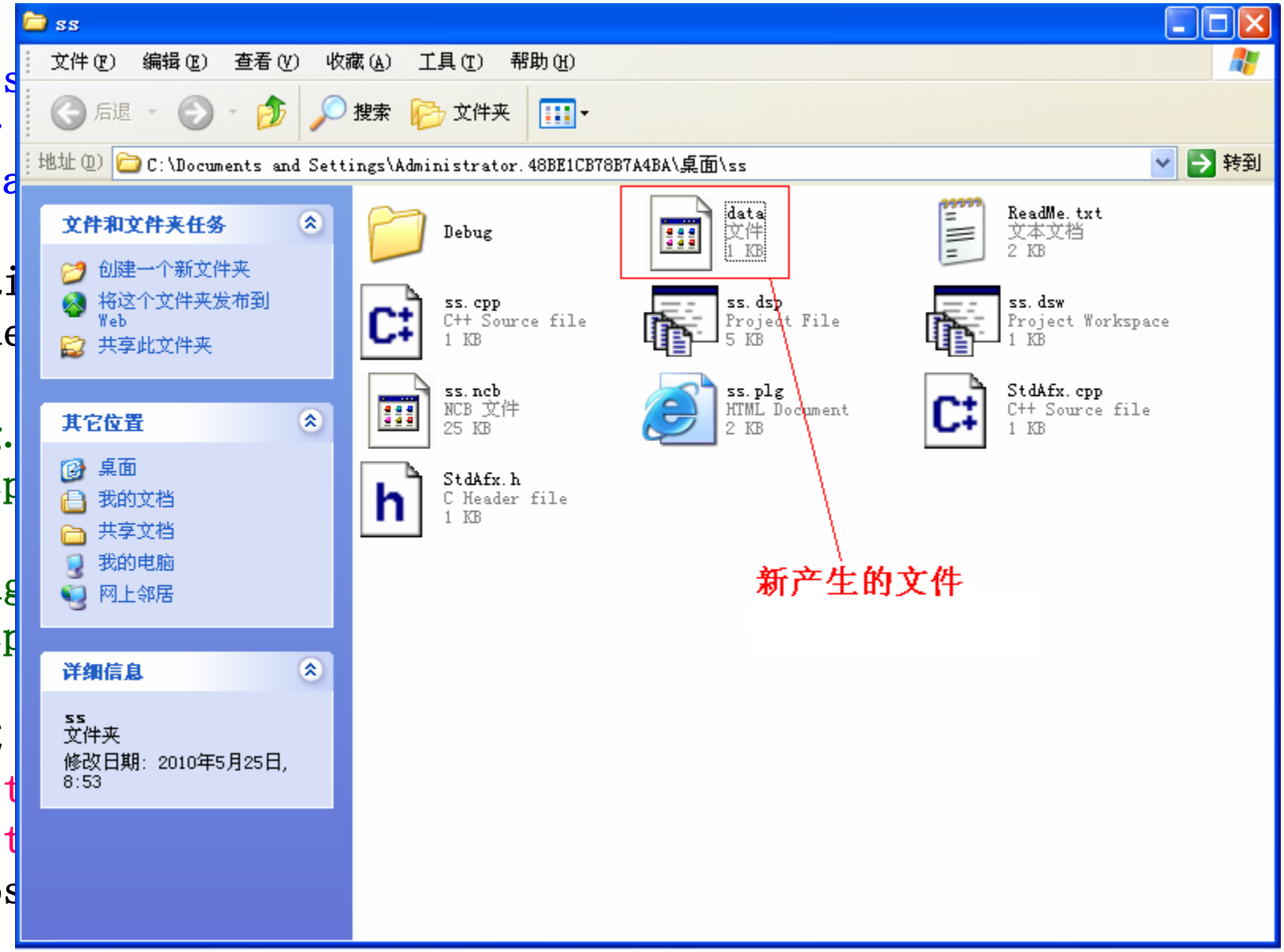
- 数据项
  - 最低级的数据组织形式，分为两类
    - 基本数据项（字段）：学号、姓名、年龄
    - 组合数据项：工资（基本工资、奖励工资等）
- 记录
  - 一组相关数据项的集合，用于描述一个对象在某方面的属性。
  - 一个记录由多个数据项组成。

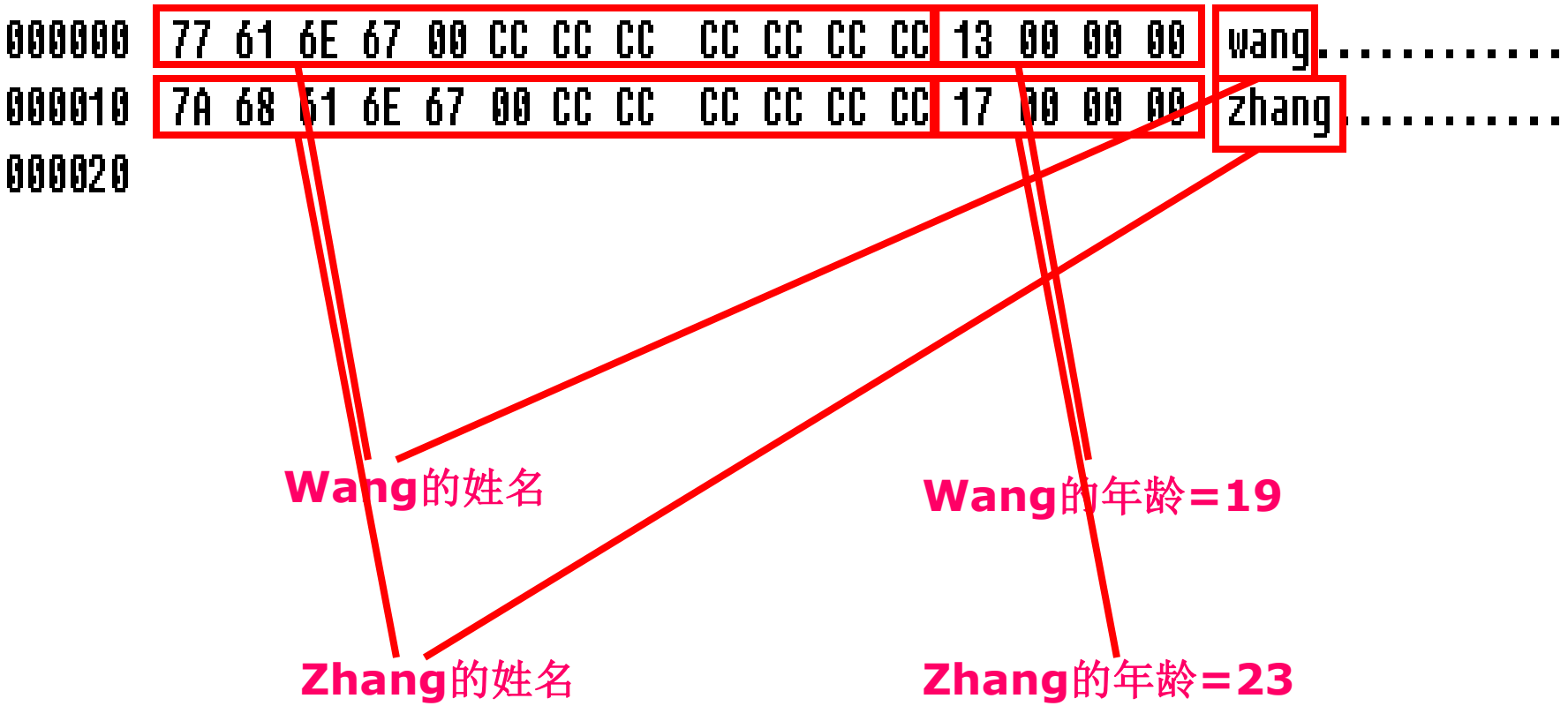

数据项(字段)
学号
姓名
性别
年龄
班级
01
张三
男
20
JSJ04—5
02
李四
男
20
JSJ04—5
记录
文件
40
王五
男
20
JSJ04—5

- 文件
  - 由创建者所定义的，具有文件名的一组相关元素的集合。
- 分类
  - 有结构的文件
    - 文件由若干个相关记录组成
    - 大量的数据结构和数据库采用这种形式
  - 无结构文件
    - 被看成是一个字符流
    - 比如，源程序、可执行文件、库函数等，即流式文件

- 有结构的文件举例

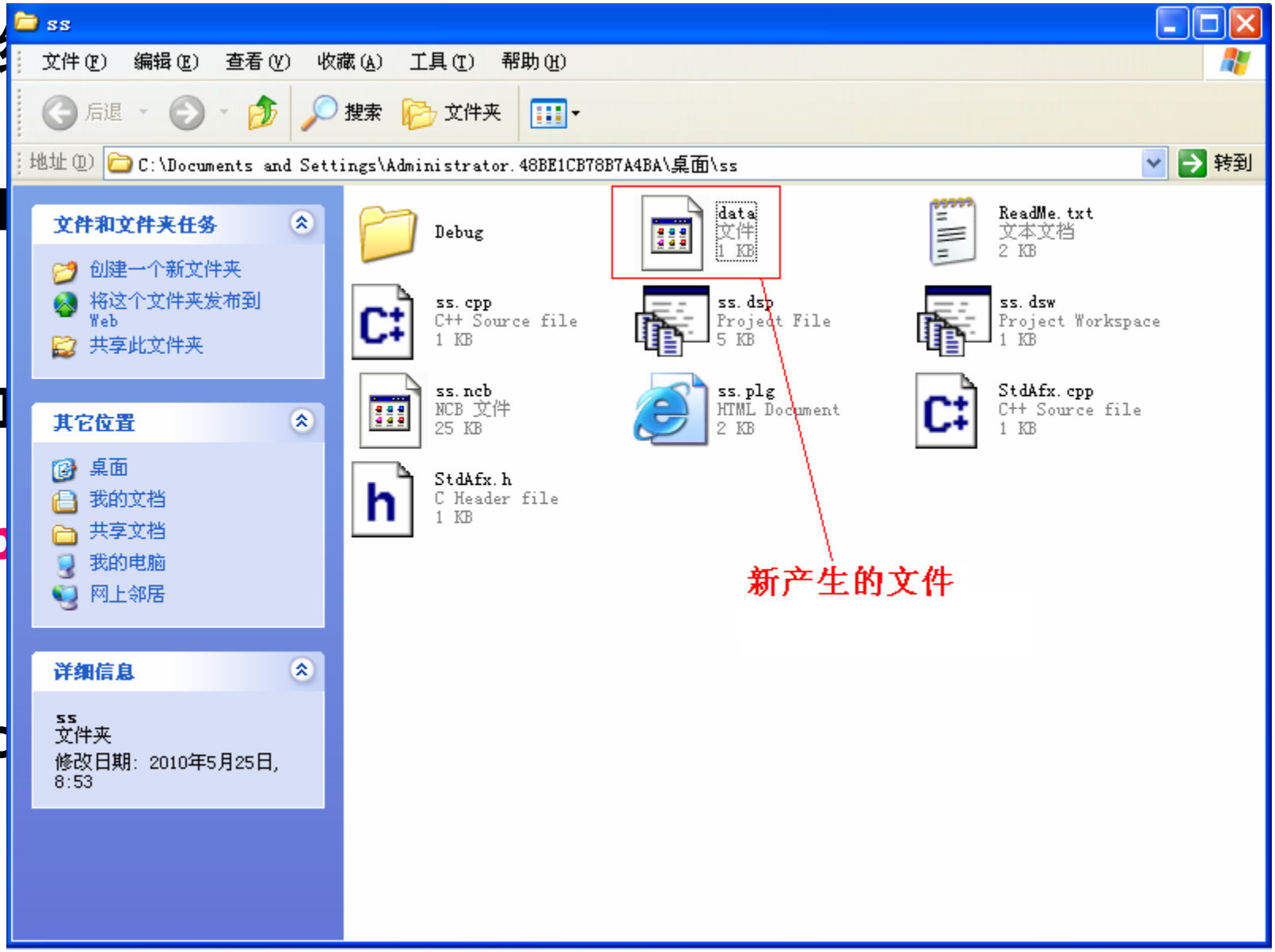
```
struct s
{  char
   int a
};
void mai
{  stude

   wang.
   strcp

   zhang
   strcp

   FILE
   fwrit
   fwrit
   fclos
}
```

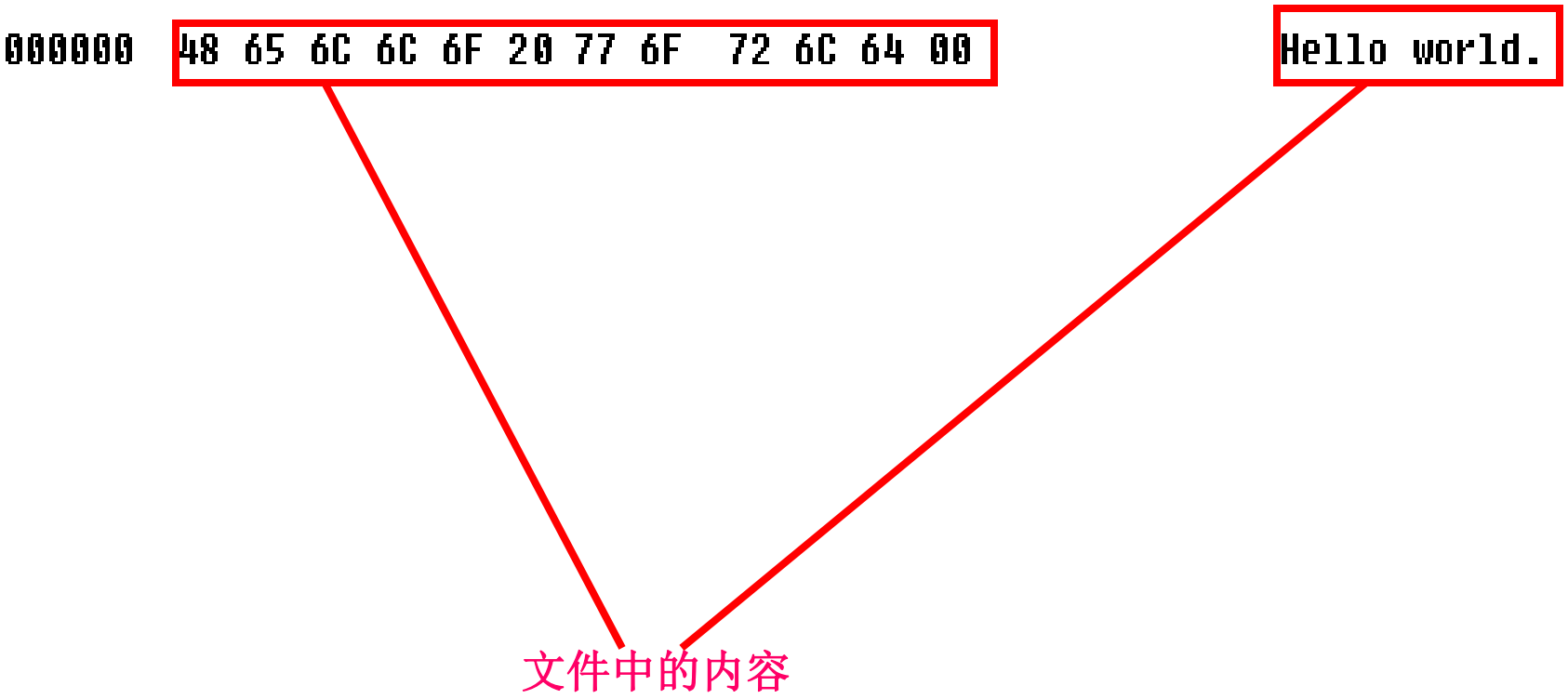



000000 77 61 6E 67 00 CC CC CC CC CC CC CC 13 00 00 00 wang.....
000010 7A 68 61 6E 67 00 CC CC CC CC CC CC 17 00 00 00 zhang.....
000020
Wang的姓名
Wang的年龄=19
Zhang的姓名
Zhang的年龄=23

- 无

```
void
{
    Fl
    fp
    fc
}
```




000000 48 65 6C 6C 6F 20 77 6F 72 6C 64 00
Hello world.
文件中的内容

- 文件的属性

① 文件类型

可以从不同角度规定文件类型，比如源文件、目标文件、执行文件等

② 文件长度

③ 文件的物理位置

④ 文件的建立时间

……

- 文件名和扩展名
  - 文件必须要有文件名，名字的长度因系统不同而异。
  - 扩展名是为方便系统和用户了解文件类型而设置

- 文件类型

① 按用途分类

- 系统文件、用户文件、库文件

② 按文件中数据的形式分类

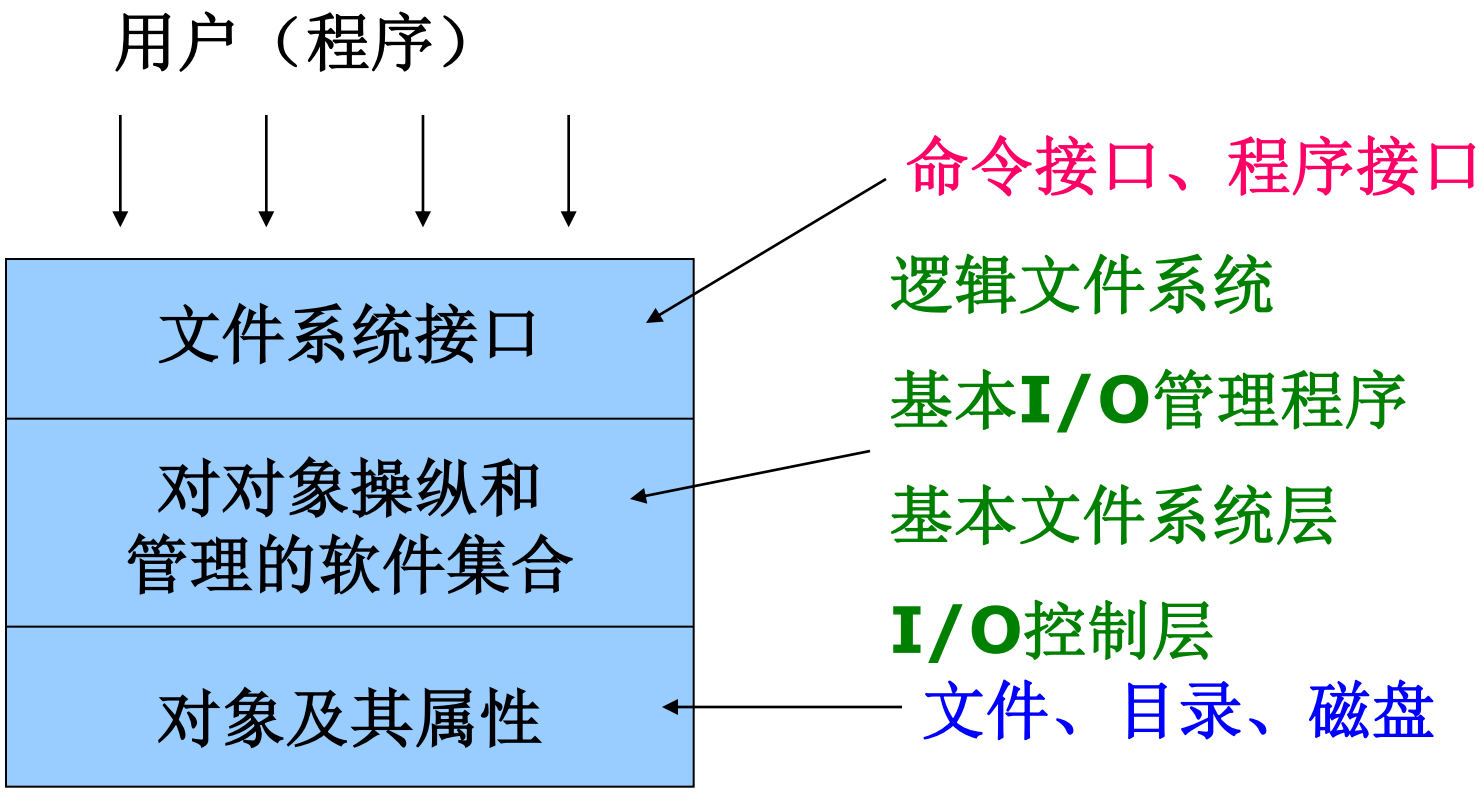
- 源文件、目标文件、可执行文件

③ 按存取控制属性分类

- 只执行文件、只读文件、读写文件

- 某些系统，如UNIX等，把I/O设备设为特殊文件

- 文件系统的层次结构


用户（程序）
文件系统接口
对对象操纵和
管理的软件集合
对象及其属性
命令接口、程序接口
逻辑文件系统
基本I/O管理程序
基本文件系统层
I/O控制层
文件、目录、磁盘

- 最基本的文件操作

① 创建文件
② 删除文件
③ 读文件
④ 写文件
⑤ 截断文件
⑥ 设置文件的读/写位置
⑦ 文件的“打开”和“关闭”操作

- 文件的“打开”和“关闭”操作
  - 用户要对一个文件实施多次访问时，每次都要从检索目录开始，为避免重复检索目录，在大多数OS中都引入“打开”、“关闭” 系统调用

C语言里的打开、关闭：

```
#include <stdio.h>

char buf[300];
FILE *fp;
fp=fopen("C:\\document\\1.txt","r");
if (NULL == fp)
{ printf("open file error\n"); return;}
if (NULL == fgets(fp,299,buf))
{printf("fgets error\n"); return;}
fclose(fp);
```

– 打开文件的步骤：

- 用户请求操作某文件时，先利用**open**系统调用打开文件
- **OS** 检索目录，找到指定文件，将其属性（包括文件在外存的物理位置）从外存拷贝到内存的打开文件表中，并产生一个表项，将表项编号返回给用户
- 此后，用户可利用表项编号在打开文件表中找到文件信息，这样便可直接访问文件，而无需检索目录

– 关闭文件的步骤：

- 用户不需要对该文件操作时，可利用close系统调用关闭文件
- OS 将文件的表项从打开文件表中删除
- 此后，用户若想再操作该文件，需重新利用open打开文件

7.2 文件的逻辑结构

- 文件系统设计的关键
  ① 记录在文件中的组织方法 (以什么方法组织文件的内容)
  ② 文件在外存上的储存方法
- 因此，对于任何一个文件，都存在以下两种形式的结构
  ① 文件的逻辑结构。
  ② 文件的物理结构 (文件的存储结构)。
- 无论哪种存储方式，都会影响文件的操作速度。

- 文件逻辑结构设计的基本要求

① 提高检索速度，能从大量记录中快速找到指定的记录。

② 便于修改，便于增加、修改、删除记录。

③ 减少文件占用的存储空间。

7.2.1 文件逻辑结构的类型

1. 有结构文件
   - 按记录长度分类
     - ① 定长记录：所有记录长度相同。
     - ② 变长记录：记录长度可以不同。
   - 按记录组织方式分类
     - ① 顺序文件：记录按某种顺序排列
     - ② 索引文件：为变长记录文件建一个索引，为每个记录建一个表项，以加快检索速度
     - ③ 索引顺序文件：上述两种方式的结合。建立索引表，为每组记录中的第一个记录建表项

2. 无结构文件

- 无结构的文件，即流式文件
- 对流式文件的访问，采用读写指针来指出下一个要访问的字符。
- 可以把流式文件看作是记录式文件的一个特例（一条记录只有一个字符）

顺序文件

- 特点
  - 记录可以任意顺序排列

1. 逻辑记录的排序

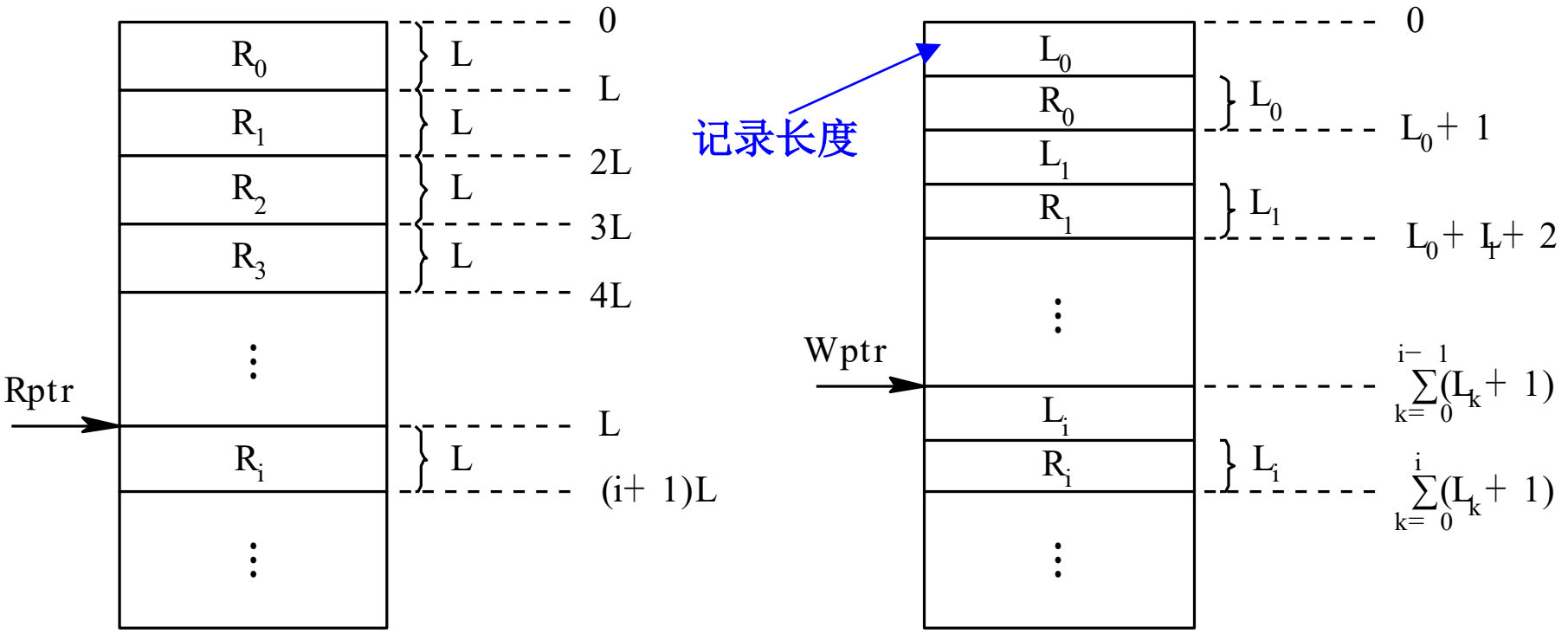
① 串结构

- 通常依据时间，最先存入的记录作为第一个记录，其次存入的为第二个记录， 依此类推

② 顺序结构

- 记录按关键字排列。
- 可以按关键字大小排序 或 按其英文字母顺序排序
- 关键字必须唯一

2. 对顺序文件的读/写操作



(*a*) 定长记录文件　　(*b*) 变长记录文件

定长和变长记录文件

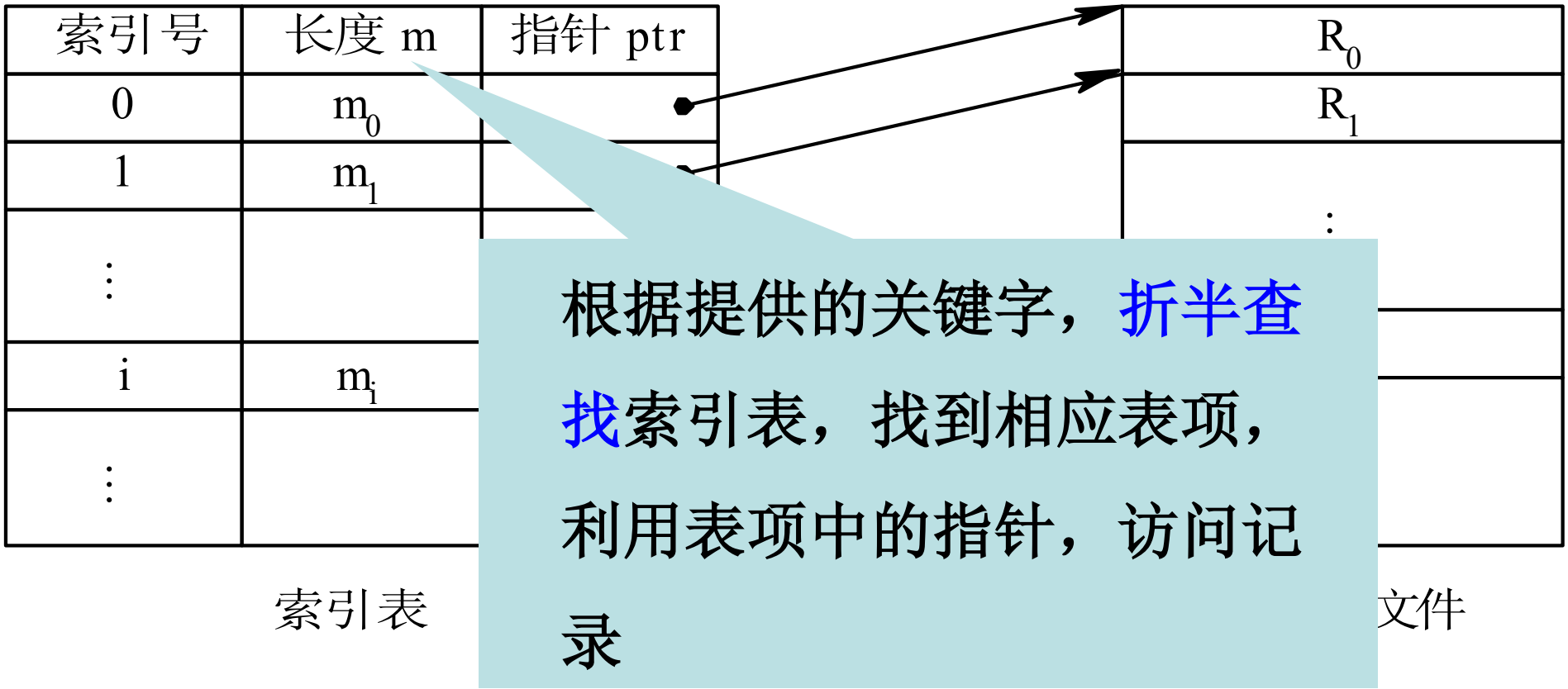
3. 顺序文件的优缺点

- 最佳应用场合
  - 每次要读或写一大批记录，存取效率最高
  - 特定设备，特别是磁带：只能顺序存储
- 不适宜场合
  - 交互应用，要求查找或修改单个记录，OS要逐个查找记录。性能可能很差，尤其是当文件较大时。
- 另一个缺点
  - 增加、删除单个记录困难。
- 解决办法
  - 为顺序文件配置一个运行记录文件(事务文件)，把要增加、删除或修改的信息记录其中，定时将该文件与原主文件合并，产生一个按关键字排序的新文件。

索引文件

- 对于变长记录文件，要查找某个记录，必须从第一条记录查起，即只能进行顺序查找
- 所以，对于可变长度记录文件，查找起来比较费时
- 如何实现对变长记录文件的直接存取呢？

- 解决方法
  - 给变长记录文件配备一张索引表
- 索引文件的目的
  - 使变长记录文件可以直接存取
- 索引表的结构
  - 按记录的关键字排序
  - 表中每个索引项长度相等（索引表是一个定长记录的顺序文件）

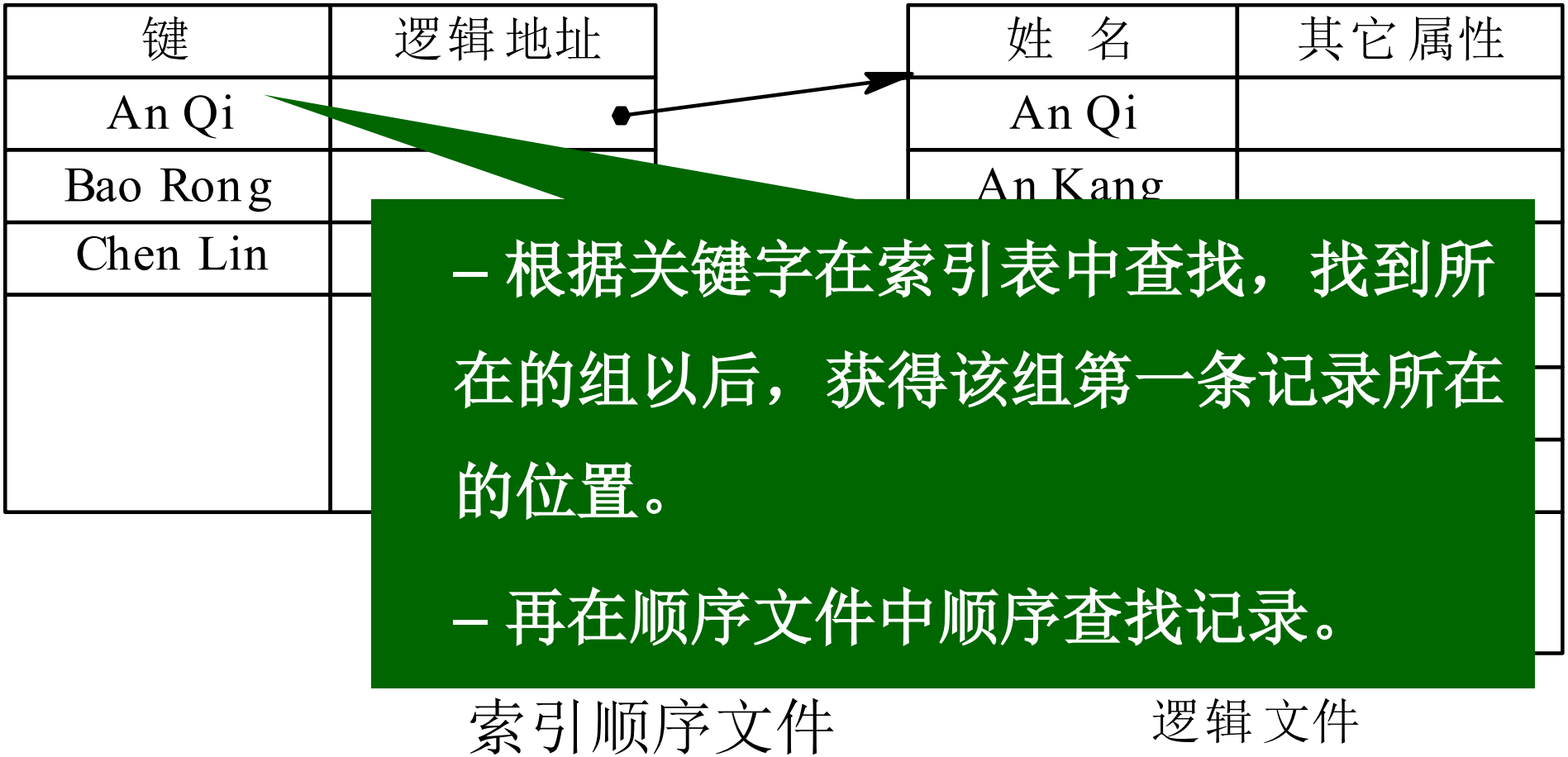


具有单个索引表的索引文件

- 注意
  - 增加记录时，须同时对索引表进行修改
- 存在的问题
  - 存储量比较大 (每个记录都要建索引)

索引顺序文件

- 它克服了变长记录不能直接存取、索引文件存储量较大的问题。
- 基本思想
  - 将顺序文件中的所有记录分组，为每组建一个索引项，指向每组的第一个记录。（一级索引）
- 索引项包含
  - 关键字、指向每组第一个记录的指针


键
逻辑地址
姓 名
其它属性
An Qi
An Qi
Bao Rong
An Kang
Chen Lin
– 根据关键字在索引表中查找，找到所在的组以后，获得该组第一条记录所在的位置。
– 再在顺序文件中顺序查找记录。
索引顺序文件
逻辑文件

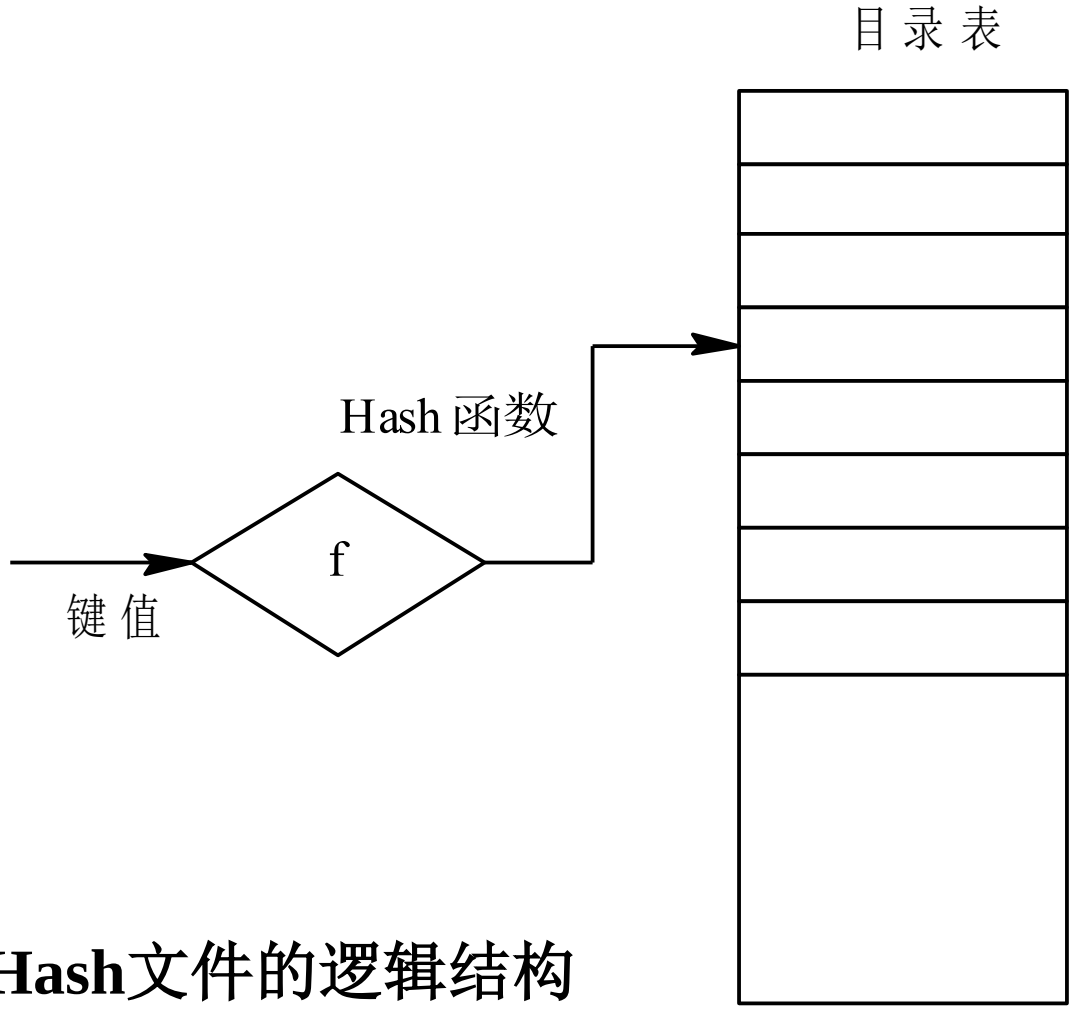
直接文件和哈希文件

1. 直接文件

- 根据给定的记录关键字，可直接获得记录保存的位置。
  - 即，关键字本身决定了记录的物理地址。
- 设计直接文件的关键
  - 用什么方法从关键字到物理地址的转换。

2. 哈希(Hash)文件

- 应用最广泛的一种直接文件



Hash文件的逻辑结构

7.3 文件目录

- 文件目录是一种数据结构，用于标识系统中的文件及其物理地址，供检索文件时使用。
- 对目录管理的要求如下：

① 实现“按名存取”

– 只提供文件名，便可快速找到文件在外存上的位置

② 提高对目录的检索速度

③ 文件共享

④ 允许文件重名

– 允许不同用户对不同文件使用相同的文件名

文件控制块和索引结点

- 为实现对文件的正确存取，须为每个文件设置一个文件控制块(FCB)，用以描述和控制文件
- 文件目录：文件控制块的有序集合
- 文件目录项：也即文件控制块
- 一个文件目录也被看做是一个文件，称为目录文件

1. FCB中包含的信息

① 基本信息类

- 文件名 、文件物理位置、文件逻辑结构、文件的物理结构

② 存取控制信息类

- 文件存取权限等

③ 使用信息类

- 文件建立时间、修改时间等

文件名	扩展名	属性	备用	时间	日期	第一块号	盘块数

MS-DOS的文件控制块

2. 索引结点

① 索引结点的引入

- 文件目录会占多个盘块，每次查找文件时，先调入第一个盘块的信息到内存，查找文件名，没找到，再查第二个盘块，依此类推

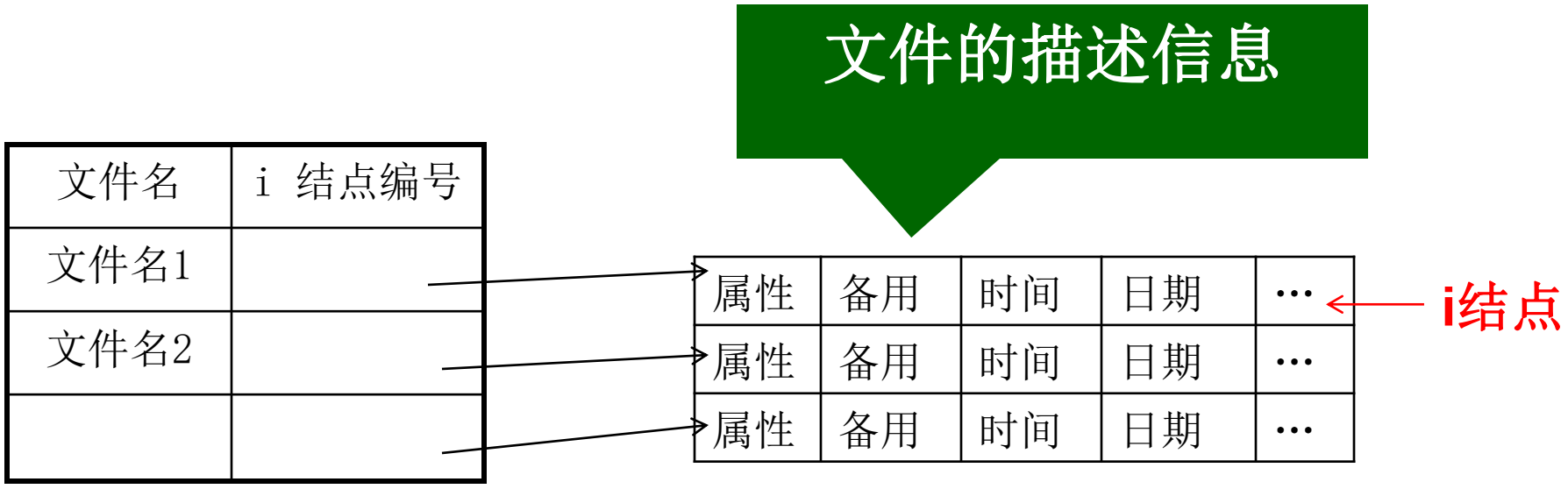
文件名1	扩展名	属性	备用	时间	日期	…
文件名2	扩展名	属性	备用	时间	日期	…
文件名3	扩展名	属性	备用	时间	日期	…

- 这种方法的缺点
  - 调入内存的信息太多。
- 解决办法
  - 只调入文件名，由此引入索引结点的概念

- 在UNIX的文件目录中，把文件名和文件的描述信息分开保存
- 文件描述信息单独形成一个数据结构，称为“i结点”(索引结点)
- 目录项仅由两部分组成

  ① 文件名

  ② 指向相应的i结点的指针


文件的描述信息
文件名
i 结点编号
文件名1
文件名2
属性
备用
时间
日期
…
属性
备用
时间
日期
…
属性
备用
时间
日期
…
i结点

带i结点的文件目录

② 磁盘索引结点

- 磁盘索引结点保存在磁盘上，每个文件有唯一的一个磁盘索引结点

③ 内存索引结点

- 当文件打开时，要将磁盘索引结点加载到内存的索引结点中，便于以后使用

简单的文件目录

- 目录结构的组织，关系到文件系统的存取速度，也关系到文件的共享性和安全性。
- 因此，组织好文件的目录，是设计好文件系统的重要环节
- 目前，最简单的文件目录形式是

① 单级文件目录

② 两级文件目录

1. 单级目录结构

- 系统中仅一张目录表，每个文件一个目录项

文件名	扩展名	文件长度	物理位置
文件名1			
文件名2			
⋮			

单级文件目录

- 单级文件目录的优点
  - 简单，能实现目录管理的基本功能—按名存取
- 单级文件目录的缺点

① 查找速度慢：特别是大文件系统

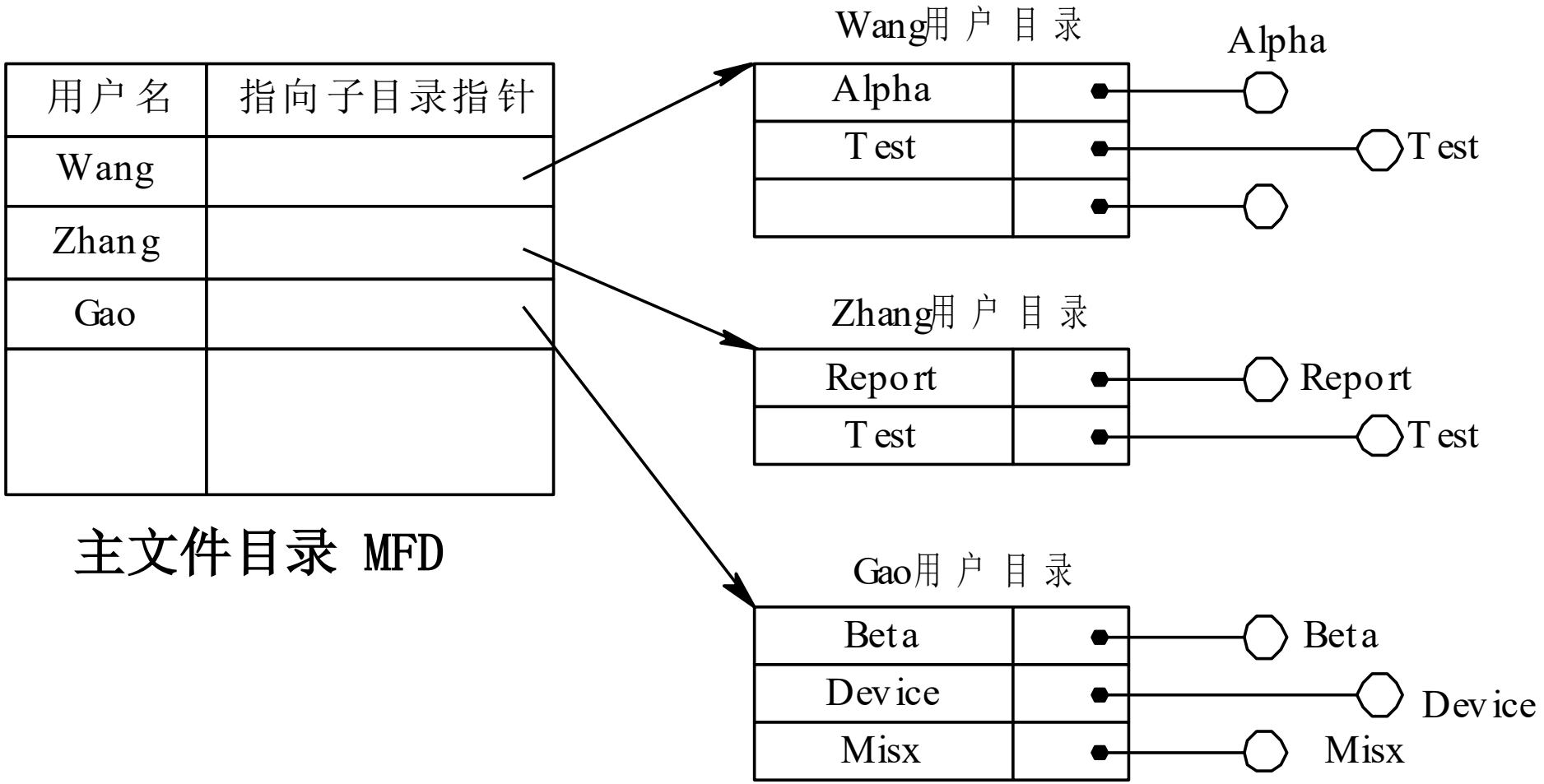
② 不允许重名

③ 不便于实现文件共享

不同用户不能用不同名字访问同一个文件。不灵活！

2. 两级文件目录

– 为每个用户再建立一个单独的用户文件目录 (UFD)



两级目录结构

- 两级文件目录的优点

① 提高了检索目录的速度

② 不同用户目录中，可以使用相同的文件名

③ 不同用户还可使用不同的文件名来访问系统中的同一个共享文件

树形结构目录（多级目录）

- 现代OS中，树形结构目录是最通用且实用的文件目录，可明显提高目录的检索速度和文件系统的性能。
  - 根结点：根目录（唯一性）
  - 树的中间结点：子目录
  - 树叶：数据文件

- 路径名
  - 从根目录到任何文件，都只有一条惟一的通路
- 当前目录
  - 因为使用全路径名访问文件太麻烦，又因为进程运行时访问的文件，大多仅局限于某个范围。所以，可为每个进程设置一个“当前目录”(工作目录)
  - 进程对各文件的访问都相对于“当前目录” 进行
  - 相对路径名：从当前目录开始直到数据文件为止所构成的路径名
  - 绝对路径名：从树根开始的路径名

目录查询技术

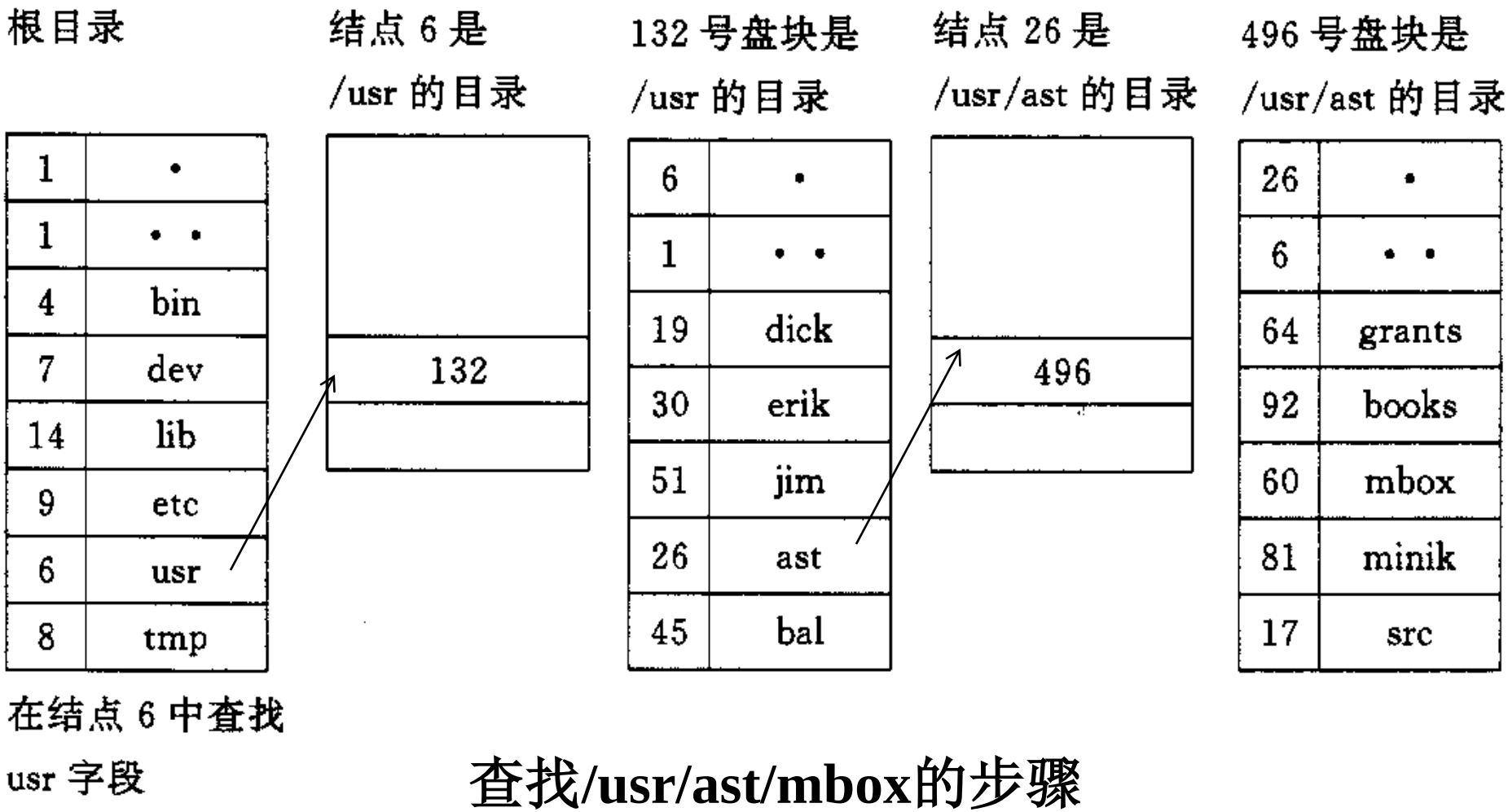
- 目录查询技术所要解决的问题是
  - 如何高效地从目录中查找文件
- 基本思路
  - 当用户要访问一个已存在的文件时，
  - OS首先根据文件名对目录进行查询，找出文件的FCB或对应的i结点
  - 根据FCB或i结点中记录的文件物理地址，换算出文件在磁盘上的物理位置
  - 再通过磁盘驱动程序，将所需文件读入内存

- 目前，目录查询主要有两种方式

  ① 线性检索法（顺序检索法）

  ② **Hash法**

① 线性检索法



查找/usr/ast/mbox的步骤

② Hash方法

- 基本思路
  - 将文件名变换为文件目录的索引值，再利用该查找目录
- 注：如果使用通配符进行查找，不能用此方法，仍然要用线性检索法。
- 通配符：

***：a*.exe (所有以a开头、扩展名是exe的文件)**

？：a?.exe (所有以a开头、两字符文件名、扩展名为exe的文件)

- 文件名转换成索引值时，可能存在不同文件名转换为相同索引值的情况，即出现“冲突”
- 一种处理“冲突”的有效方法

① 查找目录时，若目录表中相应的目录项是空的，表示无指定文件

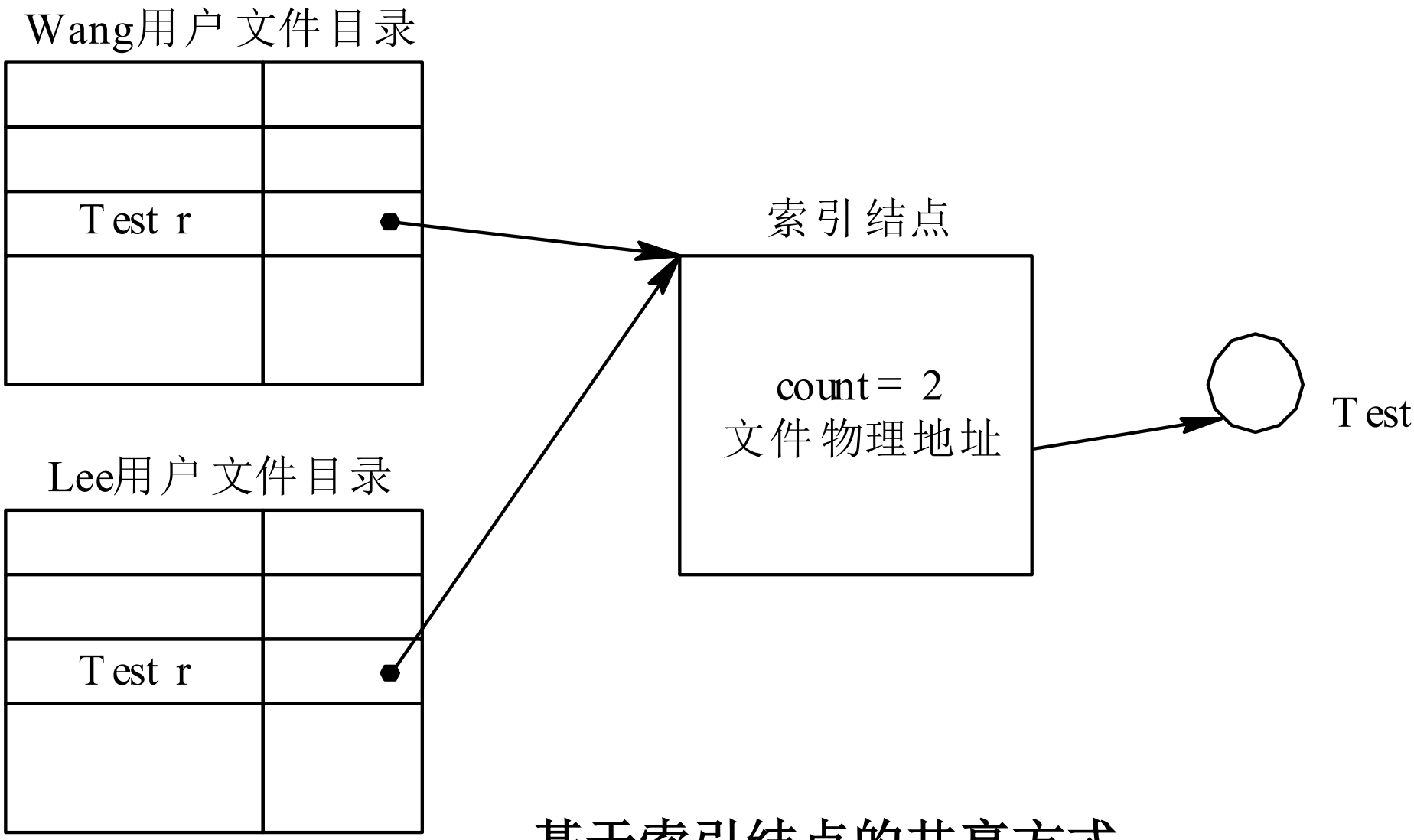
② 若目录项中文件名与指定文件名匹配，表示找到指定文件

③ 若不匹配，表示发生“冲突”。将Hash值再加上一个常数(与目录的长度互素)，形成新索引值，再到①重新查找

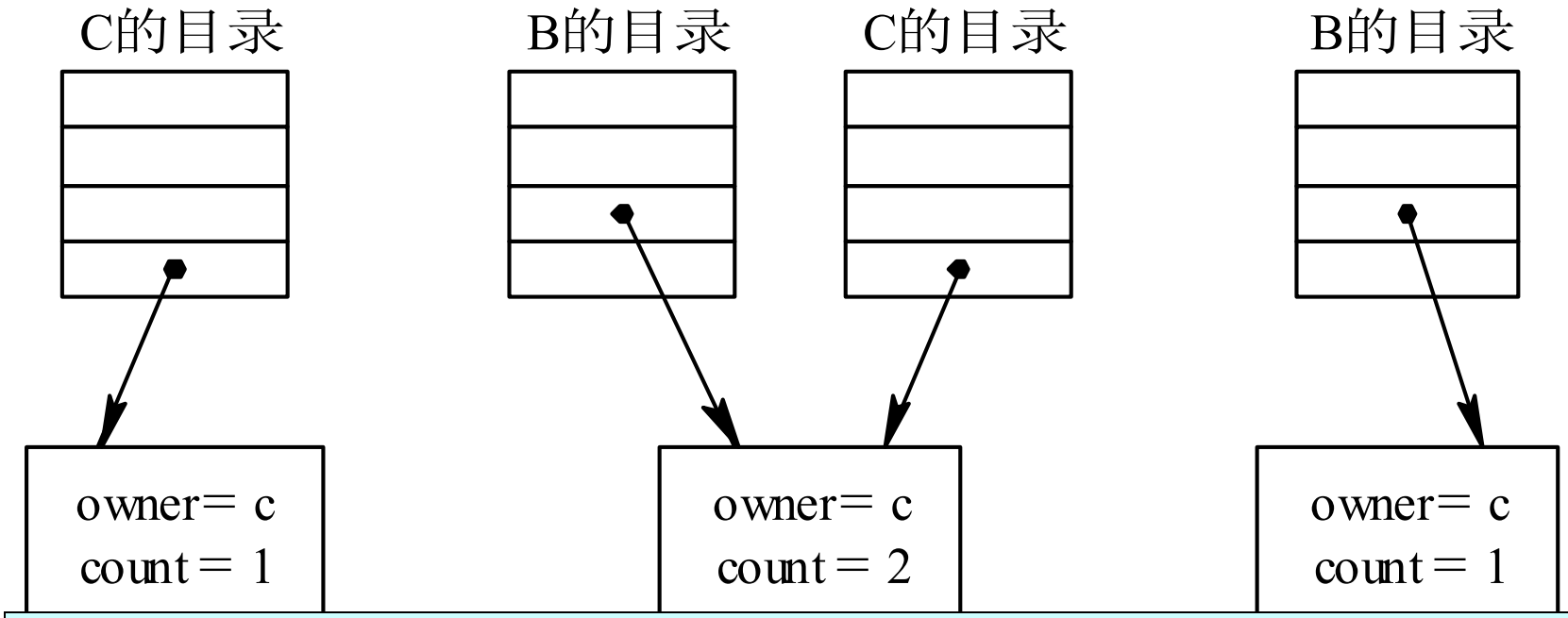
7.4 文件共享

- 文件共享可以使多个用户（进程）共享同一份文件，这样系统中只需保留该共享文件的一份副本，节省外存空间。

基于索引结点的共享方式

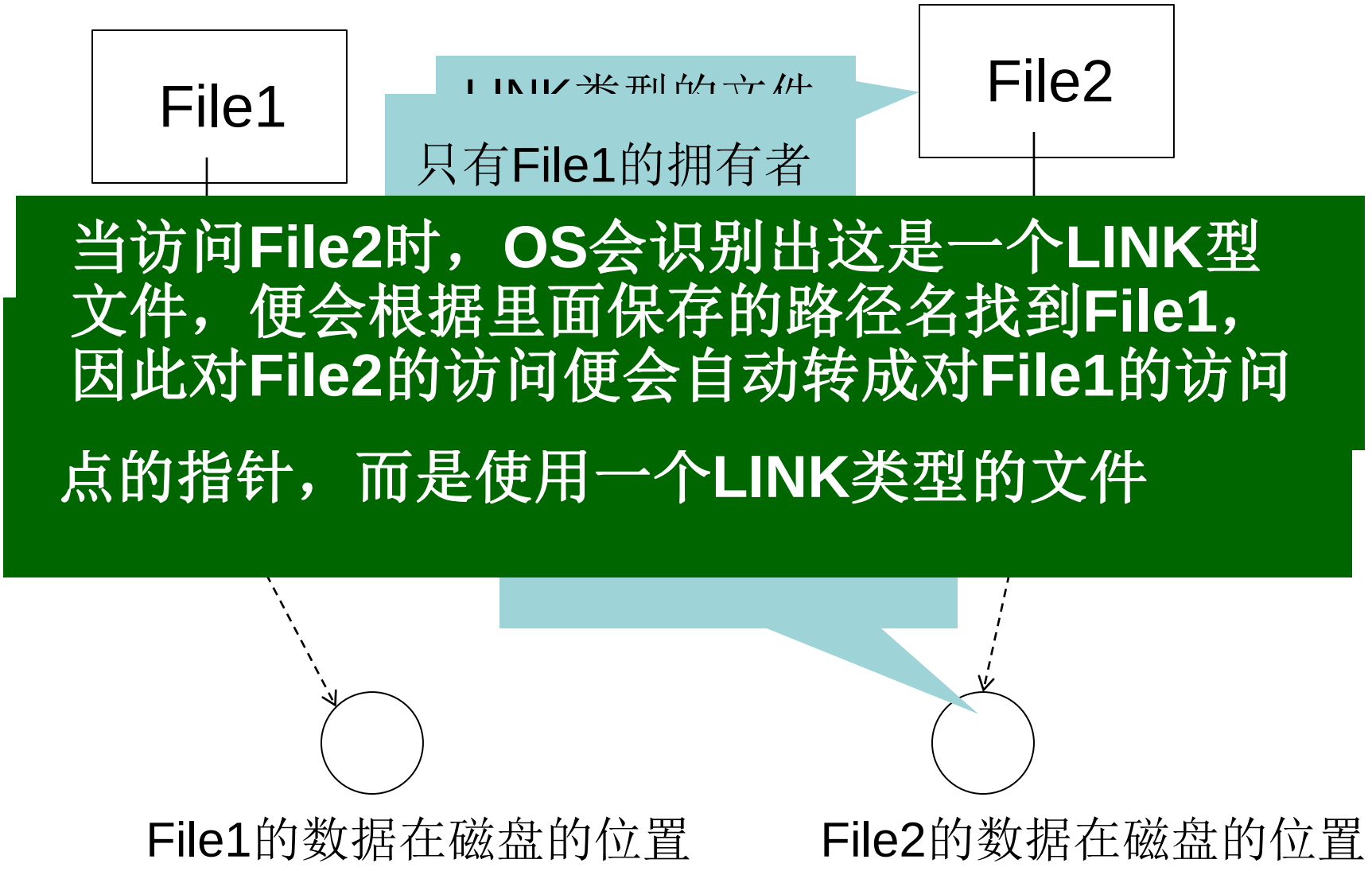


基于索引结点的共享方式



- 缺点：
  - C删除文件时，不能直接删除文件对应的索引结点
  - 因为，这样会使用户B的文件指针悬空，造成错误
  - 但如果不删除索引结点，因为C是文件拥有者，如果对文件的使用是付费的，那么C必须为B使用该文件而承担费用

利用符号链实现文件共享



- 这样，就不会发生文件拥有者删除文件后，留下一悬空指针了
- 文件拥有者把共享文件删除后，其他用户试图通过符号链去访问该文件时，会因OS找不到该文件而使访问失败，于是再将符号链删除，此时不会产生任何影响。

符号链的主要优缺点

- 优点
  - 不会留下悬空指针
  - 可以链接到世界上任何机器的文件（类似HTML）
- 缺点
  - 使用符号链访问文件时，OS是根据路径名逐个查找目录，因此要多次读盘，导致开销大
  - 符号链本身也是一个文件，占用磁盘空间

7.5 文件保护

- 由于人们有意或无意的行为，会导致文件系统中的数据遭到破坏或丢失
- 为防范这种问题的发生，现代OS中通常采用存取控制机制进行保护。

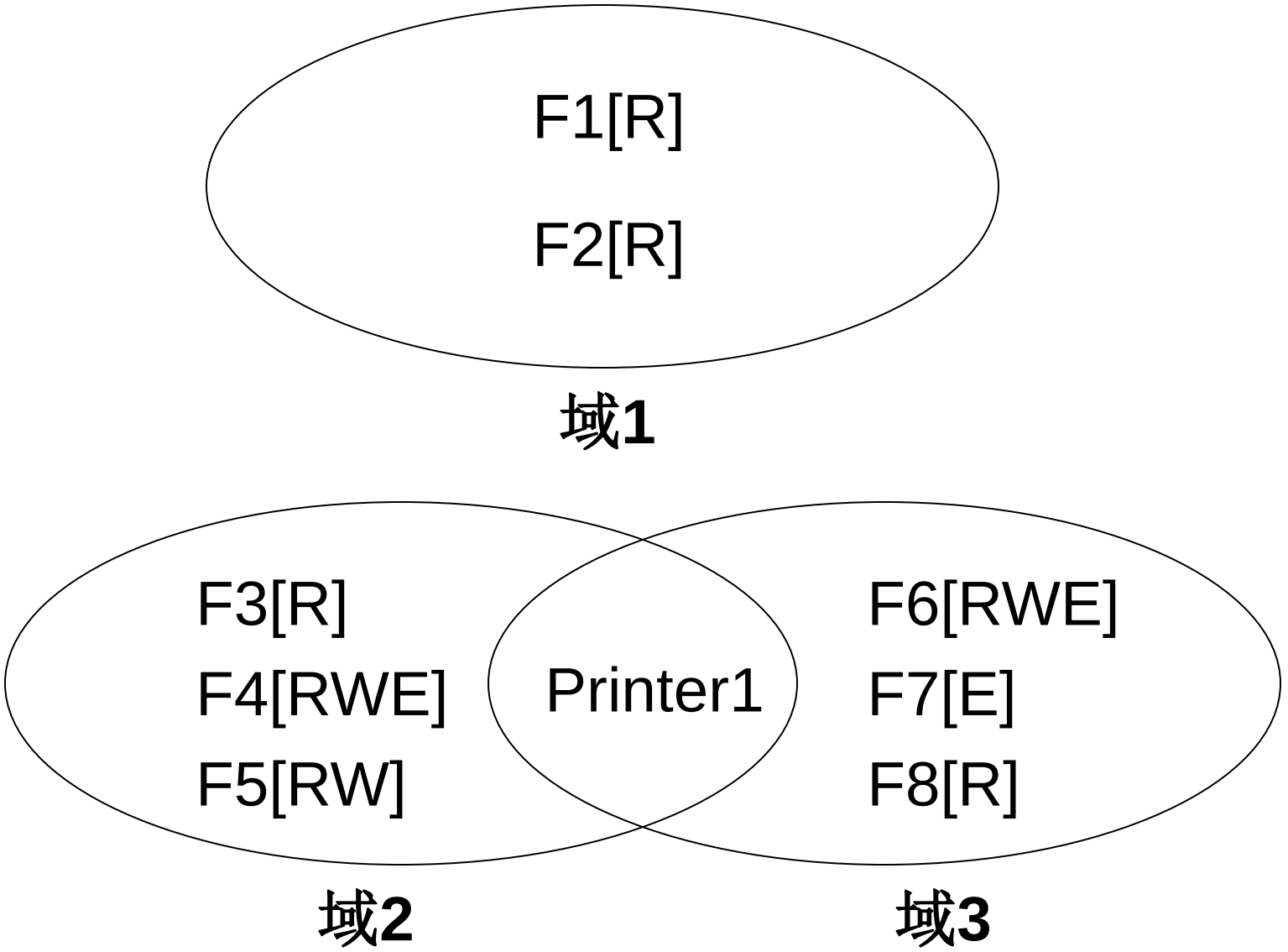
7.5.1 保护域

- 对象
  - 系统中的硬件（磁盘驱动器、打印机等）、软件（文件、程序等）
  - 他们是存取控制机制所要保护的对象
- 进程对不同对象所施加的操作有所不同
  - 文件：读/写/执行

- 访问权
  - 进程能对某对象执行操作的权力
- 每个访问权可以用一个有序对（对象名，权集）来表示
  - 某进程有对文件**F1**执行读/写的权力，则可将该进程的访问权表示为（**F1**，**{R/W}**）

- 保护域（域）
  - 进程能对哪些对象执行哪些操作

    （进程对一组对象访问权的集合）
  - 进程只能在“域”指定的权限内执行操作



保护域的例子

- 进程与域的联系

① 静态域：进程运行期间，只允许访问一个域所规定的对象，及对对象的操作。因此，进程所能使用的资源及相应的操作是固定的。

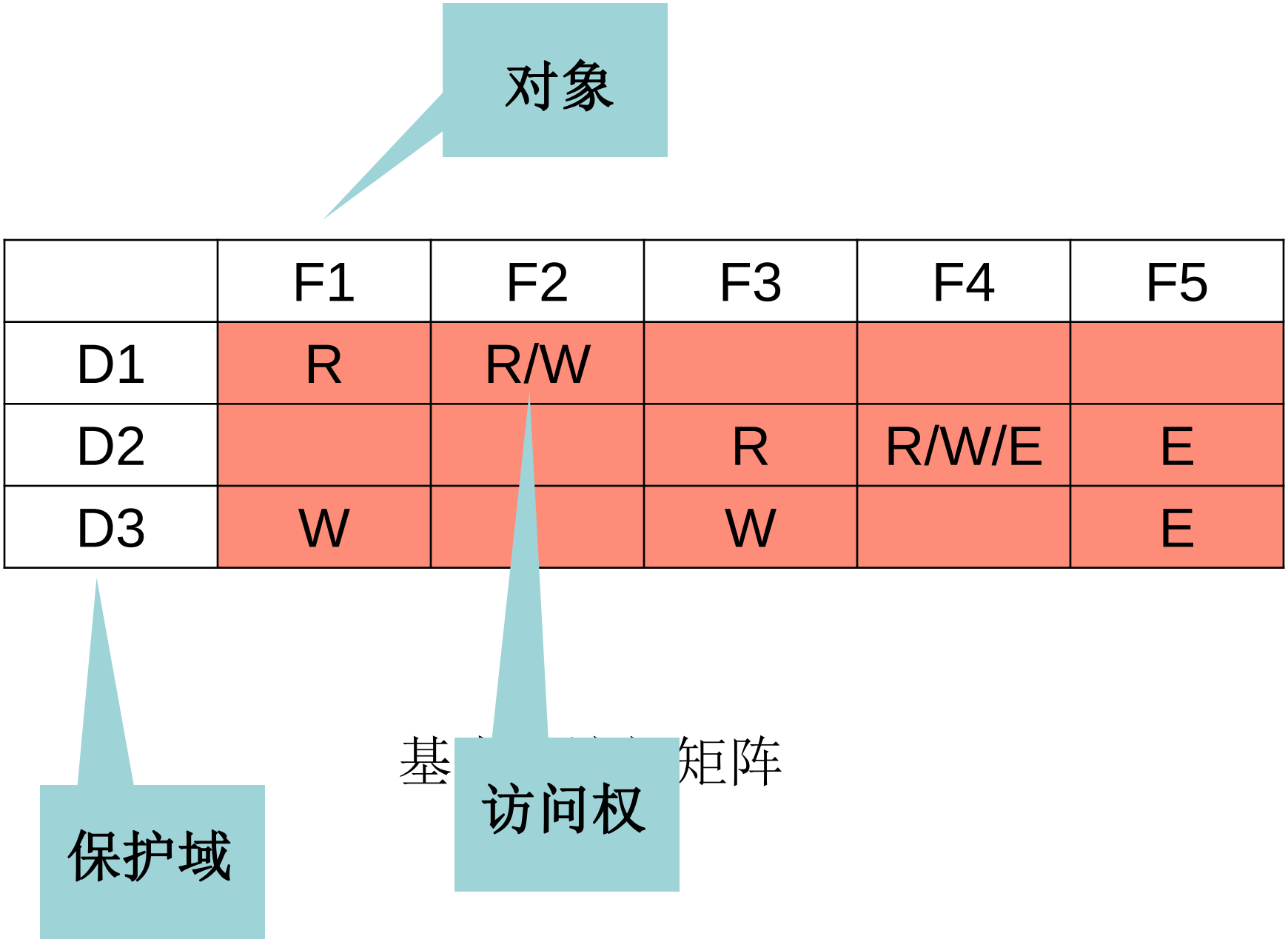
- 缺点：不能反映进程运行时的实际需要

② 动态域：进程运行期间，可按照不同域的规定进行访问。

- 优点：可根据进程运行反映其实际需要

7.5.2 访问矩阵

- 保护域是一个二维概念，每个保护域都规定了与其联系的进程可以对哪些对象执行哪些操作
- 将不同的保护域组合起来，便可形成一个二维矩阵，可以描述系统的访问控制，这个矩阵就叫访问矩阵。



基 矩阵

– 为实现进程与域的动态联系，使之能从一个域切换到另一个域

– 为能对[illegible]一种权力

与D1联系的进程可以切换到D2

	F1	F2	D1	D2	D3
D1	R	R/W		S	
D2	W	W			S
D3	W				

具有域切换权的访问矩阵

练习题

1. 文件系统为用户提供了__A__功能，使得用户能透明地存储访问文件

A. 按名存取　　B. 密码存取

C. 路径存取　　D. 命令调用

2. 文件系统是指__D__

A. 文件的集合

B. 文件的目录

C. 实现文件管理的一组软件

D. 文件、管理文件的软件及数据结构的总体

3. 从用户角度看，引入文件系统的主要目的是

D

A. 实现虚拟存储　　B. 保存系统文档

C. 保存用户和系统文档　　D. 实现对文件的按名存取

4. 文件的逻辑组织将文件分为记录式文件和

B

A. 索引文件　　B. 流式文件

C. 字符文件　　D. 读写文件

5. 为了解决不同用户文件的“命名冲突”问题，通常在文件系统中采用__B__

A. 约定的方法　　B. 两级目录

C. 路径　　D. 索引

6. 一个文件的绝对路径名是从__B__开始，逐步沿着每一级子目录向下，最后到指定文件的整个通路上所有结点组成的字符串

A. 当前目录　　B. 根目录

C. 多级目录　　D. 两级目录

7. UNIX系统中，把输入、输出设备看作是____B____

A. 普通文件　　B. 特殊文件

C. 目录文件　　D. 索引文件

8. 磁盘上的文件以____A____单位读写

A. 块　　B. 记录　　C. 柱面　　D. 磁道

9. 磁带上的文件一般只能____A____

A. 顺序存取　　B. 随机存取

C. 以字节为单位存取　　D. 直接存取

10. 在下列文件的物理结构中，__A__不利于文件长度动态增长

A. 顺序结构　　B. 链接结构

C. 索引结构　　D. Hash结构

11. 文件系统采用两级目录结构，这样可以__D__

A. 缩短访问文件存储器时间

B. 实现文件共享

C. 节省主存空间

D. 解决不同用户之间的文件名冲突问题

12. 利用索引结点实现文件共享时，对文件主删除了共享文件后会造成__A__问题，可以利用符号链问题来解决

A. 指针悬空　　B. 文件仍然存在

C. 浪费存储空间　　D. 以上答案都不对

13. 使用文件前必须先__C__文件

A. 命名　　B. 关闭　　C. 打开　　D. 备份