

OS chapter10: File system

尹志涛 22307150055

2024 年 12 月 3 日

1 File Concept 一些文件的基本知识

1.1 File Type 文件种类

文件种类取决于两种:file name and file extension(拓展)

一般文件的全名是: name.extension, 例如 txt 就是一种拓展名

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

1.2 file attribute 文件属性

1. name
2. identifier: 标识符, 每个文件都有一个独特的标识符, 一般是数字和字母的无意义的组合。
在 user 看来没什么意义

3. type, 类型, 一般和标识符差不多
4. location, 位置, 同一个位置(文件夹)中同名文件只能有一个。这里的同名指 name+extension
5. size: 文件大小
6. protection: 可读否, 可写否等等
7. data, time, user identification: 文件里的数据, 创建时间和上次修改的时间, 创建文件的用户, 或者文件所有者信息。

1.3 file data organization

1.3.1 流式文件 flow file

流式文件是指文件中的数据就是一堆字母, 二进制数字或者字符流等等。

1.3.2 记录式文件 record structure

例如 excel 表, 记录式文件又称有结构文件, excel 就把数据分成一格一格, 一列一行的类似这种。

record: 指一组相关数据项的集合, 例如 excel 的一行: 分别表示一个学生的姓名学号专业, 那么这就是一个 record。

数据项: 例如一个学生的姓名: 张三, 这是记录的最小单位, 这就是数据项。

1.4 file operation

1. create: 新建 (W)
2. read: 查看文件
3. write: 写文件
4. delete: 删除 (D)
5. open: 打开文件, 这个打开文件不是把可视化界面打开, 而是打开文件流, 允许上面几个系统调用操作文件
6. close: 关闭文件流
7. truncate: 截断, 暂无定义, 我也不懂
8. reposition within file 重定位, 暂无定义, 我也不懂

2 file structure 文件结构

2.1 无结构文件

无结构文件上面已经提到过了, 就是 flow file。这里不做解释了

2.2 有结构文件

上面也提到过了, 就是记录式文件, 但是记录式文件有很多可讲的部分, 此处展开。

2.2.1 simple record structure 简单的记录式文件

1. lines, 比如 excel, 直接按行记录一个学生的信息
2. fixed length, 定长记录, 规定每一个记录的长度为固定 byte, 其实 excel 的例子就是定长记录, 如果你确保每个学生都只记录姓名学号等等 (并且每个名字要一样长噢)
3. variable length, 变长记录: 允许姓名和学号不一样长的 excel 就是变长 record。

2.3 文件的逻辑结构

我们上面讨论的结构是: 一个 record 占多少内存空间, 并且一个 record 占空间的形式是怎样的。但是并没有将, record-s 之间应该怎么存放。

我举个例子, 我们说按定长记录, record1-record2, 这两个是定长记录, 但是怎么存放到文件里面呢, 是连续的放在一起, 还是中间搁一行, 还是随机的摆放? 哪里有空就放哪里? 这就是文件的逻辑结构所要讲的东西。

2.3.1 Sequential file 顺序存储文件

记录按顺序摆放, 我们不要求记录一定是定长/变长, 都可以。只是在物理结构上, 这些记录是连续摆放或者链式摆放的。也就是说, 内存上不一定连续!

但是这个顺序摆放, 并不是先到先得的顺序摆放, 而是有两种:

1. 先到先得, 无所谓什么顺序, 只要连续就行。这样叫做顺序结构。
2. 按关键字的顺序, 这种摆放方式称为串结构。

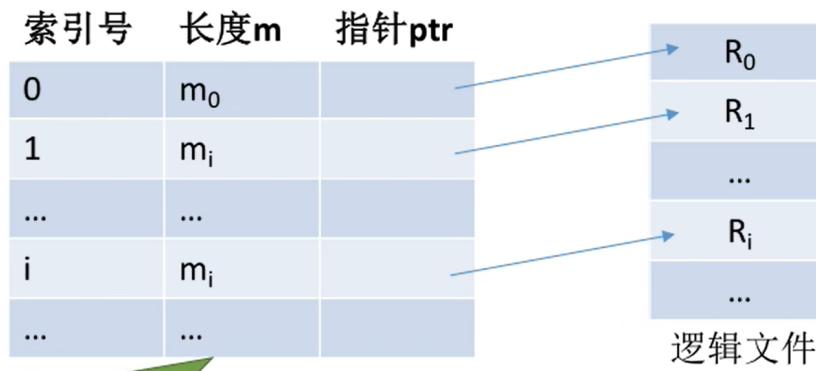
因此总结一下, 一共有这几种叫做 Sequential file

1. 链表 + 定长
2. 链表 + 变长
3. 顺序 + 定长: 可以快速索引, 根据 $\text{start}+\text{idx}*\text{length}$, 而且如果是顺序结构, 你可以用二分法 lgn 时间内查找。如果是串结构, 说实话你也不知道你的关键字重要不重要, 得看情况看看能不能快速查找。
4. 顺序 + 变长

2.4 索引文件 Indexed file

索引文件是为了处理变长 record 文件而产生的。因为变长, 所以如果你要访问一个 record, 你只能在文件中一个一个 record 的读取, 这太慢了。

存在一个索引表, 索引表的结构如:



建立一张索引表以加快文件检索速度。每条记录对应一个索引项。

文件中的这些记录在物理上可以离散地存放。

索引文件看似用处不大，因为他既然要单独创建一个跟定长的顺序文件一样的索引表的话，为什么我不一开始就直接顺序定长存储呢？

但是你要考虑到，如果一个记录，特别大，相比于其他的记录大数倍，那么这种情况用定长是非常浪费的。其次，索引表并没有那么浪费空间，因为索引表中索引号可以是关键字，且不说关键字可以做很多文章，例如用数字做关键字，这样 random access(随意访问，也就是根据索引访问) 会很快。而且你看索引表项，只有索引号 + 长度 + 指针，最多 2byte 一个项，比 record 肯定小很多。

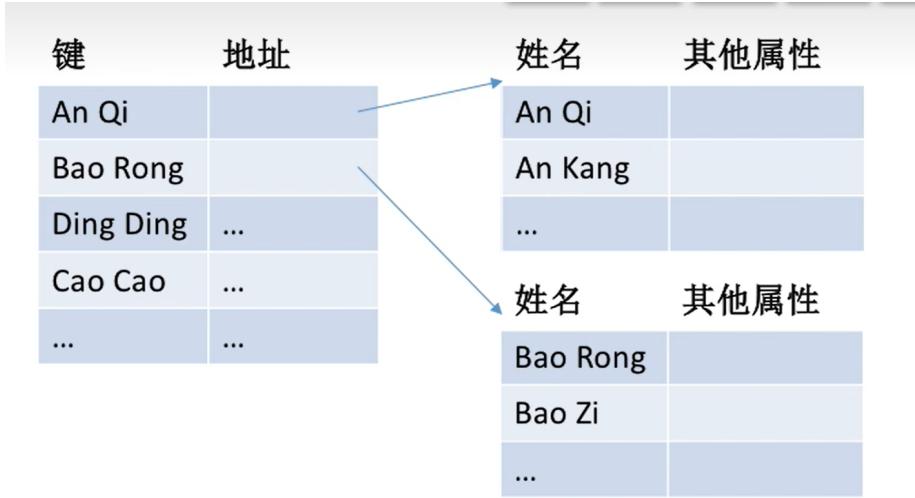
我必须提到的一点是，书上对 indexed file 的解释相当模糊，书上将索引表单独列为 indexed file，把另一部分存放记录的部分称为 relative file，这很愚蠢，他把 indexed file 分成了两个文件，这虽然很好调控，但是不利于理解。

2.5 indexed Sequential file 索引顺序文件

看这名字你就很容易知道这个文件是什么了。前面我说了，索引表相对于变长记录要小不少，但是还能更小，因此出现了索引顺序文件。

这个索引顺序文件的规则是这样的：

1. 顺序文件必须按串结构摆放，可以不定长，但你必须按串结构。也就是说，按关键字顺序存放。我举个例子，例如英文名字 Amy ,Annie,Bob,Catty,Bull, 那么按串结构，A 开头的名字放一起，B 开头的放一起。
2. 索引表是泛用索引，就是把 A 的一整块做个索引，比较像 hash，比如你查找名字 Amy，他在索引表里只有 A 块对应的开始地址，那么你就从这个开始地址里面，去顺序文件里面找。



2.5.1 overflow file

然后我现在讲，如何在顺序索引文件中加入元素。

加入元素很麻烦，因为你是按串结构定义的顺序文件，所以如果你再加一个 Amlia 进去，你必须让所有的 B,C,D... 开头的名字全都后退一格，再放你。这太麻烦，也很不好管理，因为你的索引表全乱了！

因此引入： **overflow file**

当你添加新元素的时候，首先添加到 overflow file 里面，overflow file 中对 A,B,C, D 等都分别配置了一个起始地址，只要没填满你就接着填。然后当你读取 main file 也就是顺序文件的时候，如果你找 A，找到最后都没找到，这时 main file 中 A 的最后一行会存在一个指针让你去找 overflow file 的 A 的起始地址然后继续找。

当 overflow 满了的时候，update main file。全部加进去，然后对 Idx file 和 main file 全部做一次大更新。

2.6 Direct,or Hashed file 直接文件/哈希文件

这两一个意思。

然后哈希文件是对 idxseqfile 的抄袭。在 record 中找一个是数字的关键字，作为关键字。

然后搞个哈希表，哈希表里面存 val 和 Ptr。对应 val 的 ptr 直接在 file 里面去找就完了。

还有个好点的办法，就是，哈希结合 chain 链表。简单来说就是搞个 BUCKET。hash 之后顺着这个 BUCKET 慢慢找。

如果你不知道什么是 BUCKET，我解释一下。如果 50 做 hash 到 0，然后你在哈希表里 val=50 找到了一个 ptr，然后你访问 ptr 对应的地址发现，这不是你要找的。这时 ptr 对应的地址是这样的：content+nxtptr。除了 content，这个 record 项还记录了下一个 ptr 地址，你就继续去找。这就是一个链表，这个链表，称之为 BUCKET。

3 Directory 目录

目录很有用，显然。目录本身就是一种有结构文件。由一条条记录组成，每条记录对应一个放在该目录下的文件。目录的结构你可以按一个 table 去理解：

“照片”目录对应的目录文件

文件名	类型	存取权限	物理位置
2015-08	目录	只读	...	外存25号块
2015-09	目录	读/写	...	外存278号块
.....			...	
2016-02	目录	读/写	...	外存152号块
.....				
微信截图 _20180826102629	PNG	只读	...	外存995号块

3.1 File control block 文件控制块

目录文件中的一条记录就是 FCB，文件控制块。

FCB 的有序集合称之为“文件目录”，一个 FCB 就是文件目录项。

3.2 Operations performed on Directory 目录操作

我们一般会对目录进行哪些操作？

1. Search for a file
2. Create a file
3. delete a file
4. list a directory 显示目录
5. rename a file
6. traverse the file system 遍历文件系统

3.3 Directory structure 目录结构

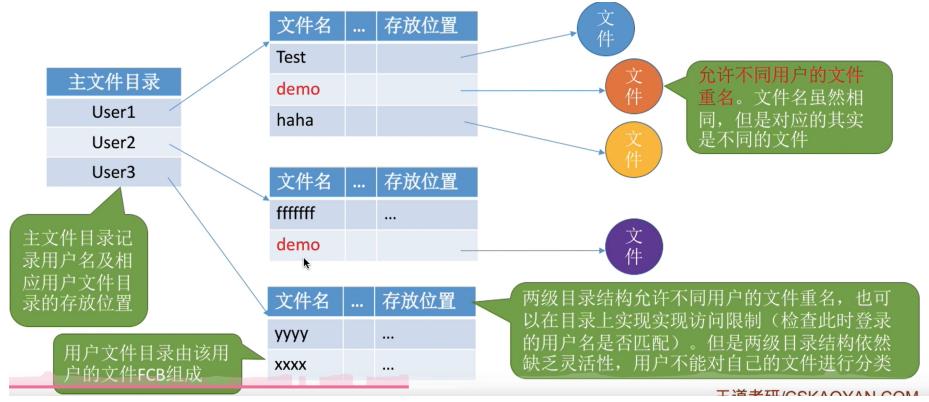
3.3.1 单级目录 single-level Directory

单级目录很好理解，就是只给你一个目录，所以你整个电脑不允许任何文件重名 (name and extension)。就相当于你只有一个 C 盘，C 盘里面不让你创建文件夹。

3.3.2 两级目录

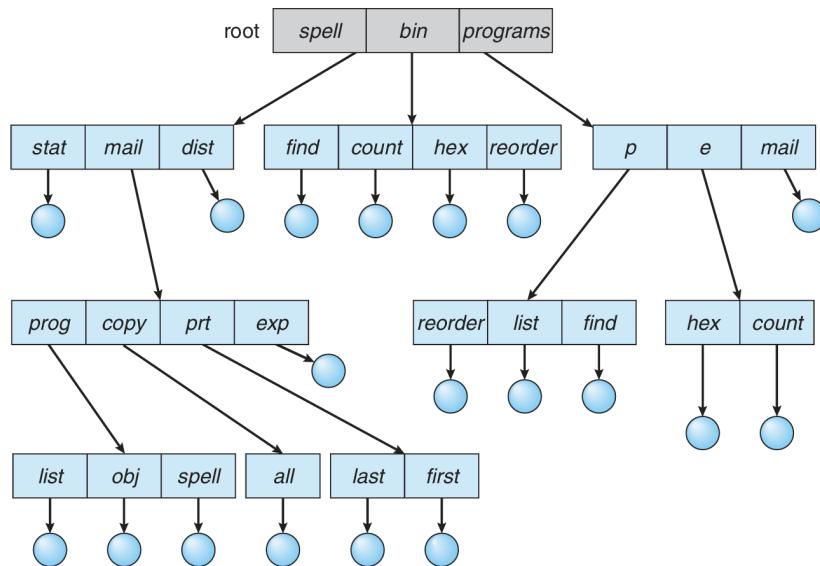
第一级目录的目录项为第二级目录。

第二级目录的目录项为文件。



3.3.3 多级目录 Tree-structure Directory

我们的电脑就用的这种方式。你可以一直创建文件夹。文件夹里面再创建文件夹。嵌套



这里就涉及了绝对路径和相对路径的概念。

绝对路径就是从根目录出发，精准的详细的告诉你需要访问的文件的地址。

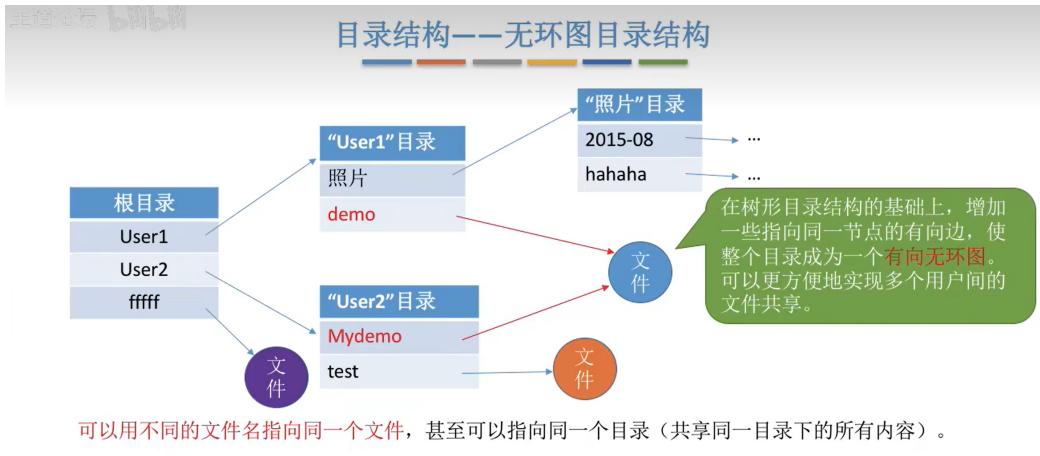
相对路径就是从当前目录出发。

3.3.4 Acyclic-Graph Directory 有向无环图目录

这个和树目录的区别就是，可以共享一个文件，也就是一个文件可能有两个绝对路径！但这样意味着，你可能不能创建同名文件了。

因为你创建或者删除之前你得考虑：1. 你的同一层是不是已经有你这个名字了？注意，是同一层，而且同一层是没那么容易检测的。

我举个例子，先看结构图。



比如说，我创建了一个 common.txt，同时给 user1 和 user2 用，好了，现在我要删除 common.txt，我至少需要知道哪些文件指向了 txt，假设 user1, 2 之外 user2/temp 也指向这个 txt，那你在目录中目录项至少要添加一个 <parent> 部分，而且这个 parent 还不是定长...

3.4 access list and Groups

访问的模式有：

1. read(R)
2. write(W)
3. execute(X)

对于用户来说分为三级，如果是文件 owner 的访问，那么 RWX 为 111 为 7；如果是 group access，那么不具备执行权限，为 110；如果是公共访问，那么不具备修改以及读取权限，为 001；