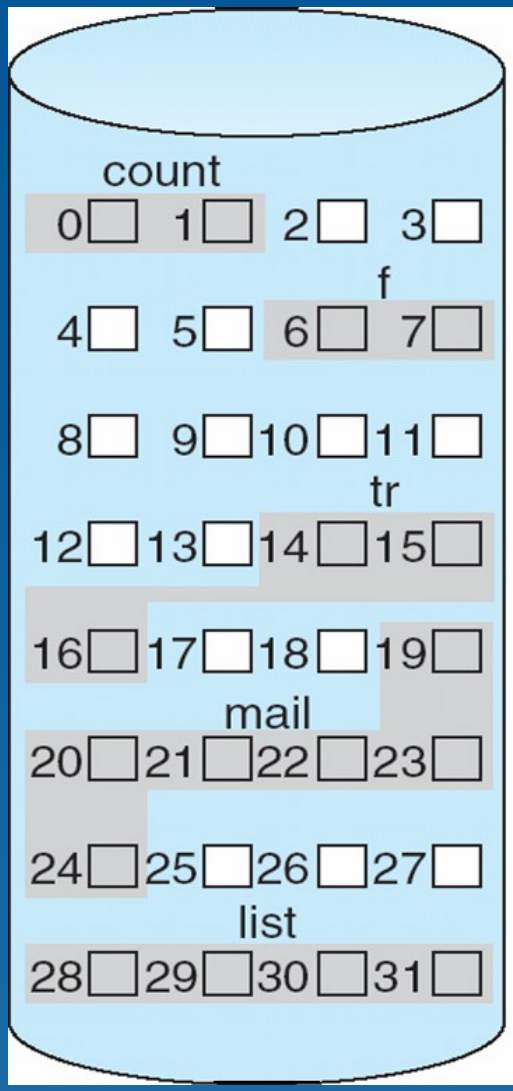
外存分配方法

磁盘空间分配给文件

- 这里讨论分配方法，是指如何将磁盘数据块分配给文件

- 连续分配， Contiguous allocation
- 链接分配， Linked allocation
- 索引分配， Indexed allocation

磁盘空间的连续分配

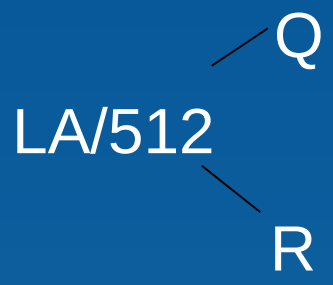


directory

file	start	length
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

连续分配

- 从逻辑地址至物理地址的映射



目标数据块 ＝Q＋ 文件起始物理地址
数据块内部的偏移 ＝R

连续分配

- 每个文件占有磁盘中一串连续的数据块
- 实现简单 – 只需要给出起始位置（块号）和文件长度（数据块总数）
- 支持随机访问

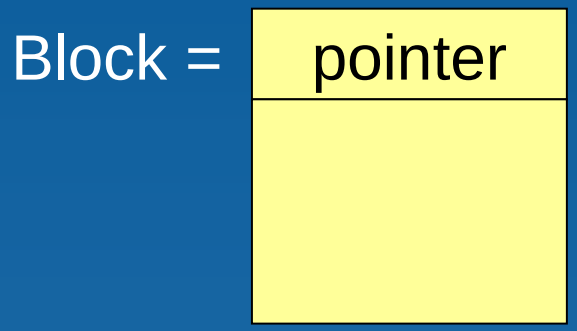
- 碎片浪费（动态存储空间分配问题）
- 不方便文件长度扩展

基于 Extent 的连续分配

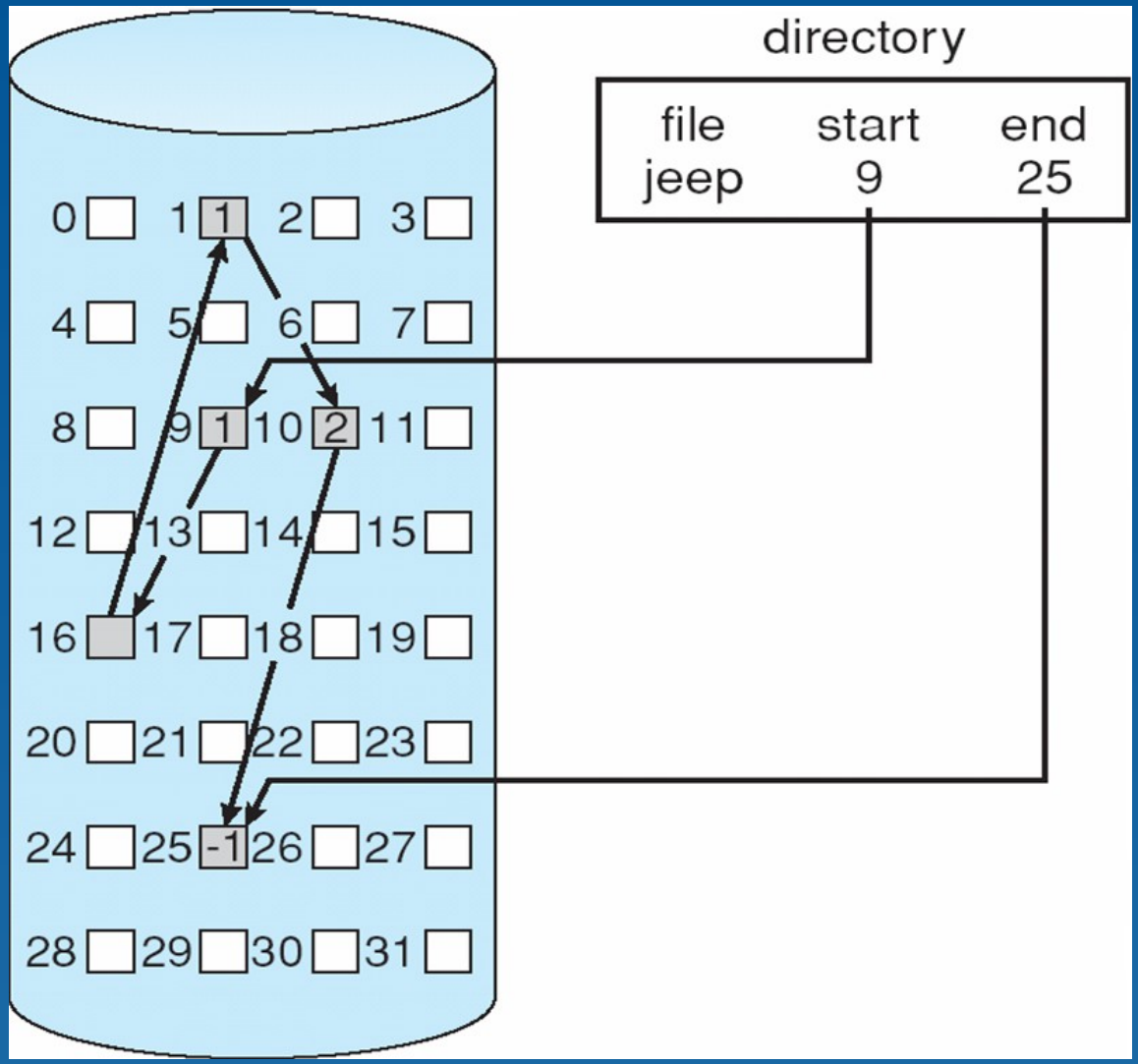
- 改进型的连续分配策略。应用于许多文件系统中
- 为文件分配磁盘空间，以 Extent 为单位
- 一个 extent 就是磁盘中一串连续数据块。一个文件由若干 extents 组成

链接 (Linked) 分配

- 文件存储在一个磁盘数据块的链表中。
- 数据块可以分散各处。只要有空闲块，都可以利用



链接分配


directory
file
start
end
jeep
9
25
0
1
1
2
3
4
5
6
7
8
9
1
10
2
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
-1
26
27
28
29
30
31

链接分配（续）

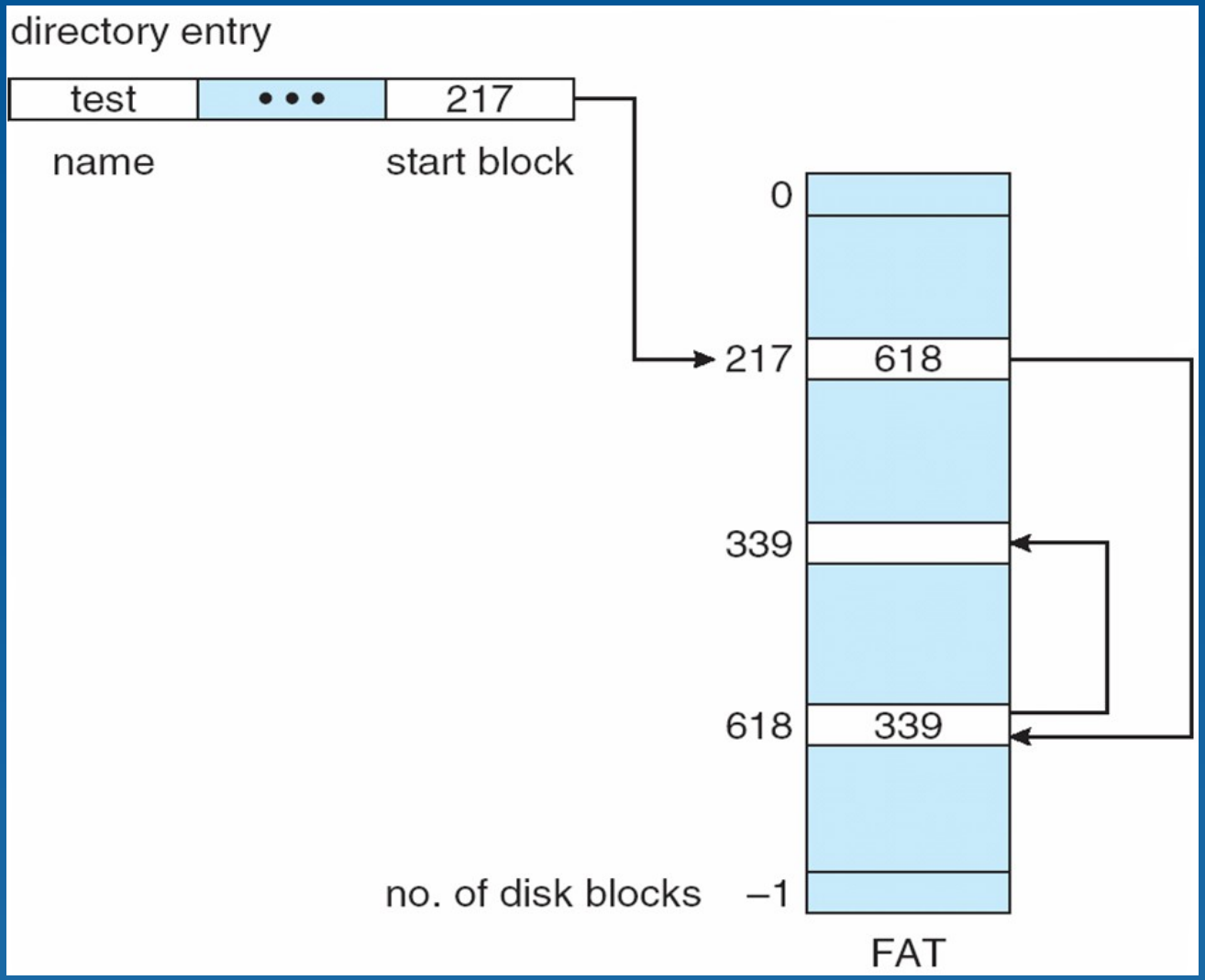
- 文件逻辑地址表示的访问位置，处于链表的第 Q 个数据块
- 数据块内部偏移 = R + 1

链接分配（续）

- 实现简单 – 只需要给出起始位置（块号）
- 磁盘数据块充分利用

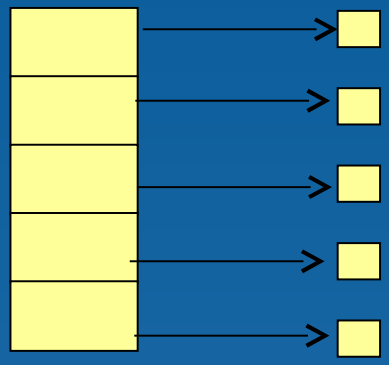
- 不支持随机访问
- 太依靠数据块中的指针，要求磁盘设备可靠性好

DOS 的 FAT - File-Allocation Table



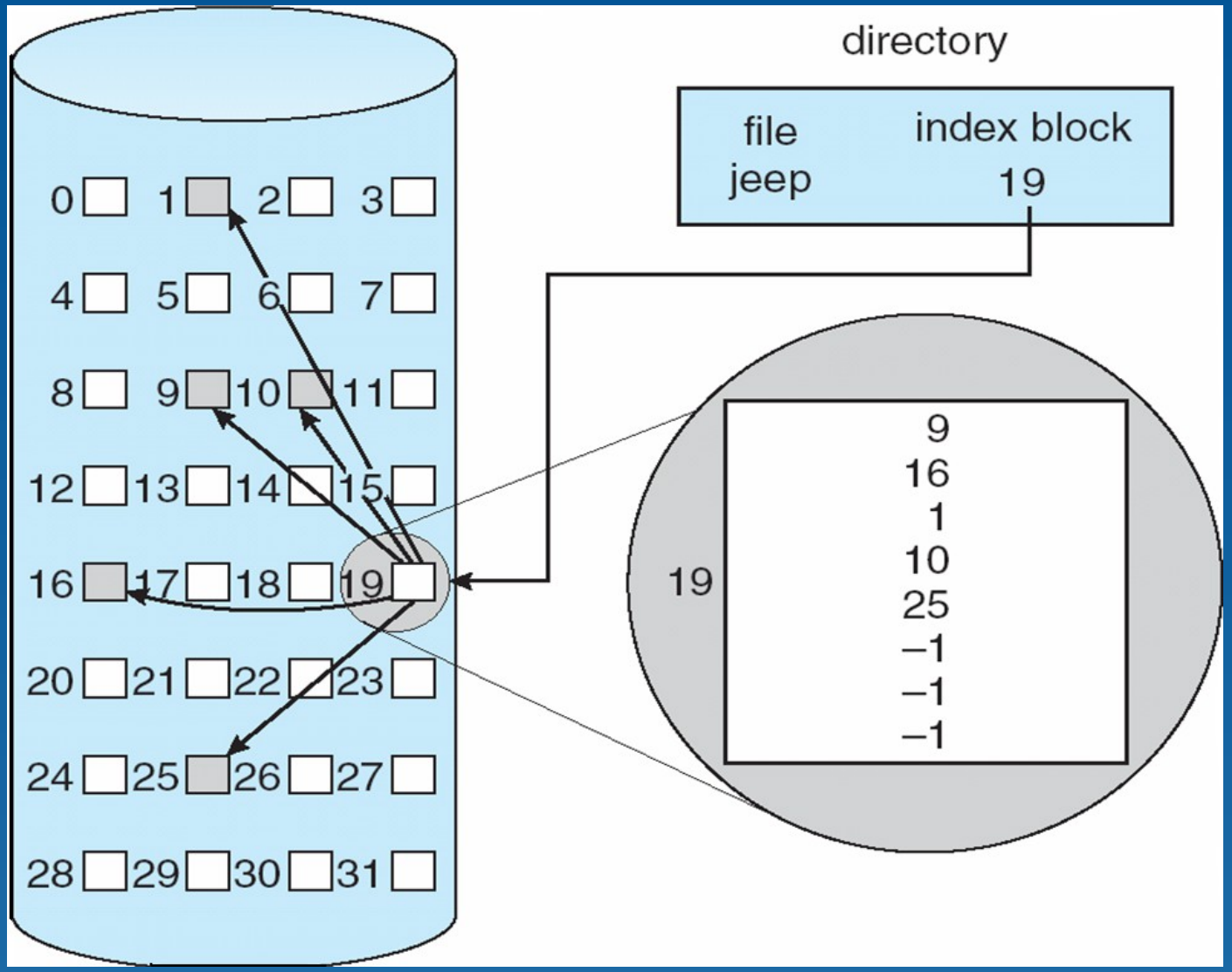
索引 (Indexed) 分配方法

- 将链表分配方法的数据块指针，全部归置到一个数据块，称作索引块 (index block)
- 逻辑视图如下



索引表

索引分配方法


directory
file
jeep
index block
19
0 1 2 3
4 5 6 7
8 9 10 11
12 13 14 15
16 17 18 19
20 21 22 23
24 25 26 27
28 29 30 31
19
9
16
1
10
25
−1
−1
−1

索引分配方法（续）

- Q = 数据块在索引表的偏移
- R = 目标位置在数据块内部的偏移

索引分配方法（续）

- 支持随机访问
- 支持动态伸缩文件长度，没有外部碎片

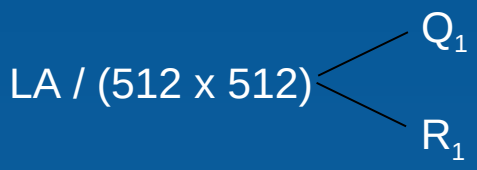
- 需要一个索引数据块
- 访问索引数据块，增减开销

索引分配方法 – 映射

- 设文件长度不超过 256K words ，数据块长度 512 words 。文件的索引表只占用 1 个数据块。
- 那么，如果文件长度再扩充呢？

索引分配方法 – 映射(续)

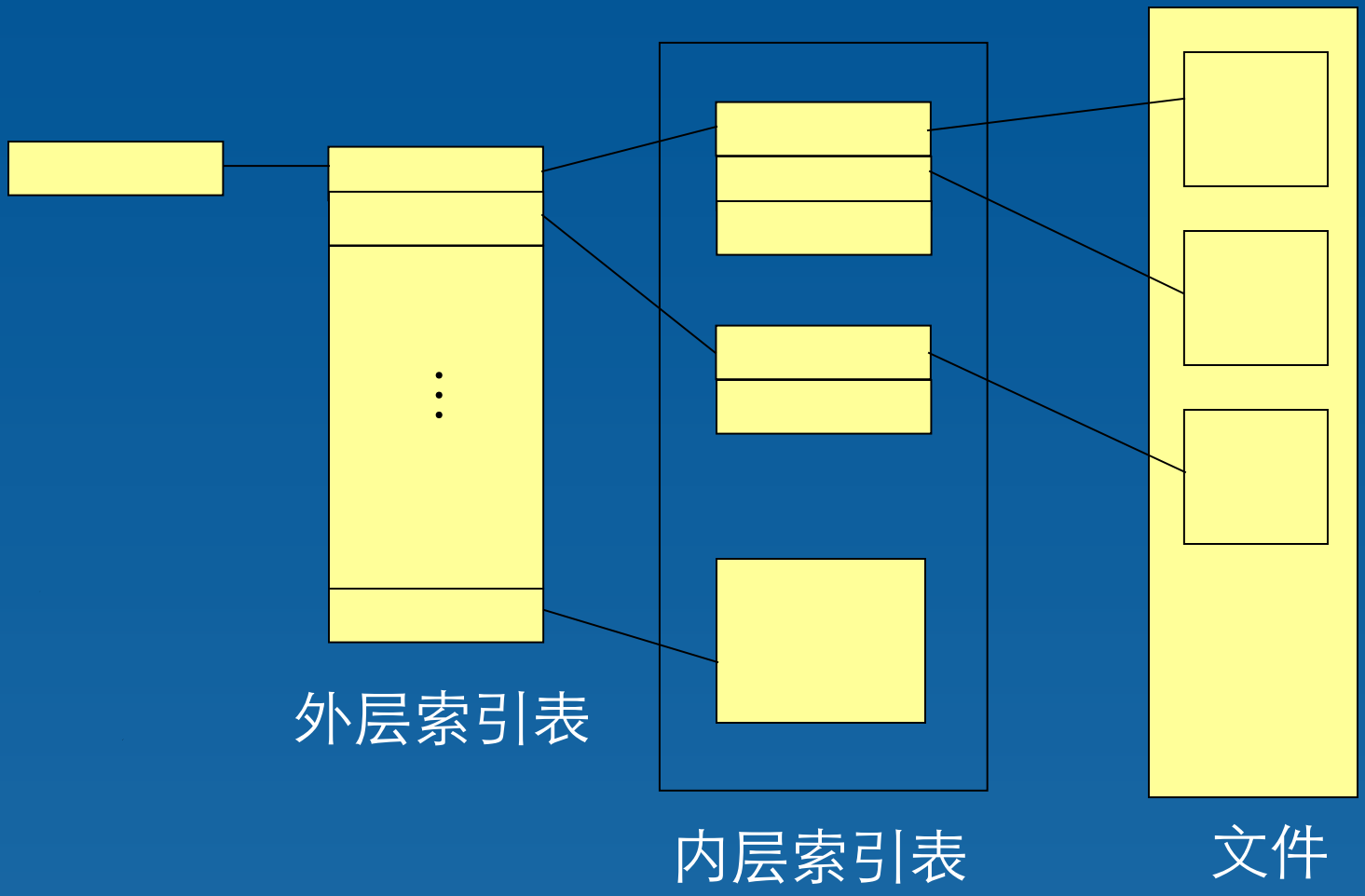
- 二层索引(最大文件长度可达 512^3)



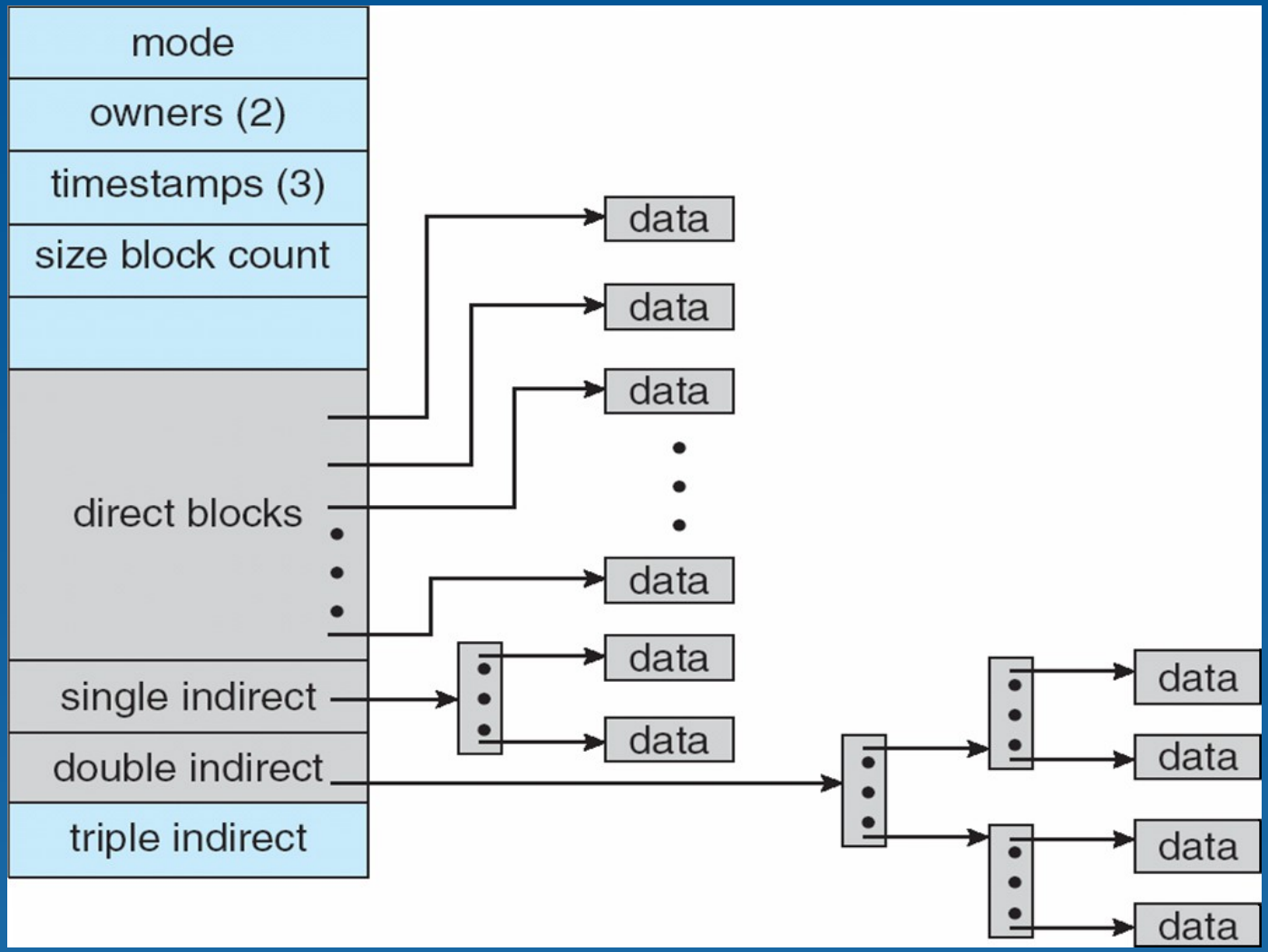
Q_1 = 外层索引表的偏移
R_1 继续展开

Q_2 = 内层索引表的偏移
R_2 = 目标位置在数据块内部的偏移

索引分配方法 – 映射(续)

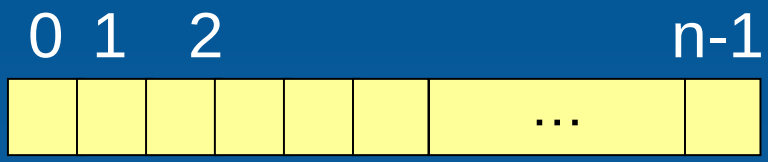


示例： UNIX UFS (数据块长度 4K bytes)



空闲空间管理 – 位图

◆位图 （管理 *n* 数据块）



$$bit[i] = \begin{cases} 0 \Rightarrow \text{数据块 } [i] \text{ 空闲} \\ 1 \Rightarrow \text{数据块 } [i] \text{ 被占用} \end{cases}$$

- 被占用数据块的地址（块号）

= （每个 word 含 bit 总数）*（内容是全 0 的 word 总数）
+ 第一个含 “1”bit 的 word 中 “1” 的偏移位置

空闲空间管理 – 位图(续)

- 保存位图需要额外的空间
  - 举例:

数据块长度 = 2^{12} bytes

磁盘空间容量 = 2^{30} bytes (1 GB)

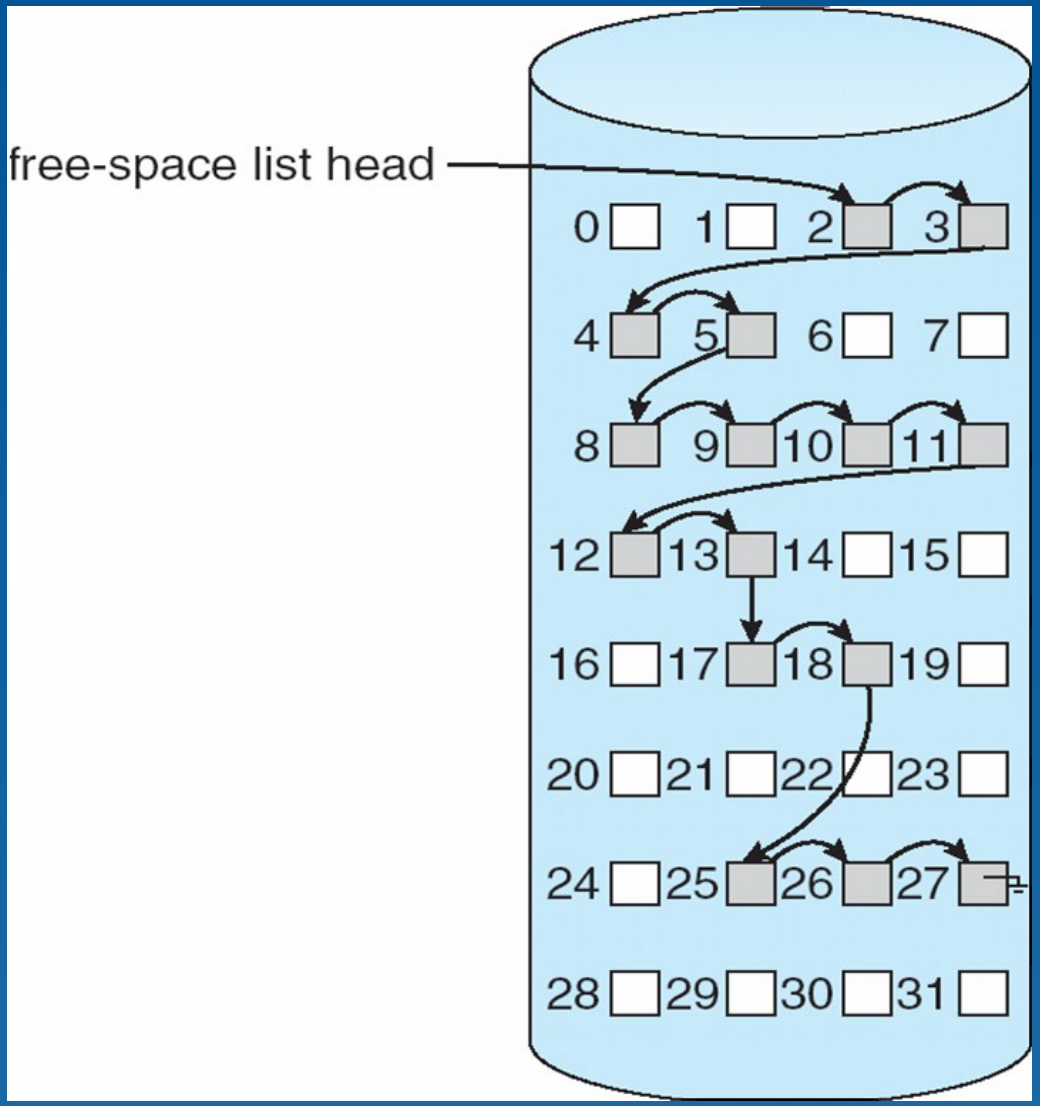
数据块总数 n = $2^{30}/2^{12} = 2^{18}$

位图占用 2^{18} bits，或 32K bytes)

空闲空间管理 – 链表

- 空闲数据块链表
  - 与链接分配方法配合使用
  - 不浪费空间
  - 不易整合出连续空闲空间

示例：空闲数据块链表


free-space list head
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31


End