PRINCIPLES OF OPERATING SYSTEMS

LECTURE 20
OPERATING SYSTEMS

File System Implementation

FILE ALLOCATION METHODS

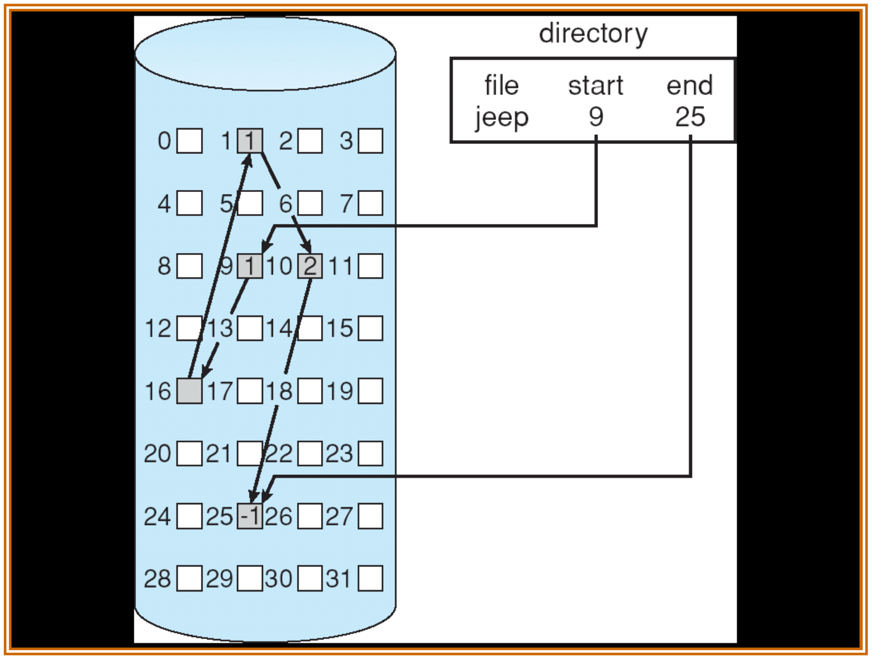
ALLOCATION METHODS

An allocation method refers to how disk blocks are allocated for files:

Contiguous allocation

Linked allocation

Indexed allocation



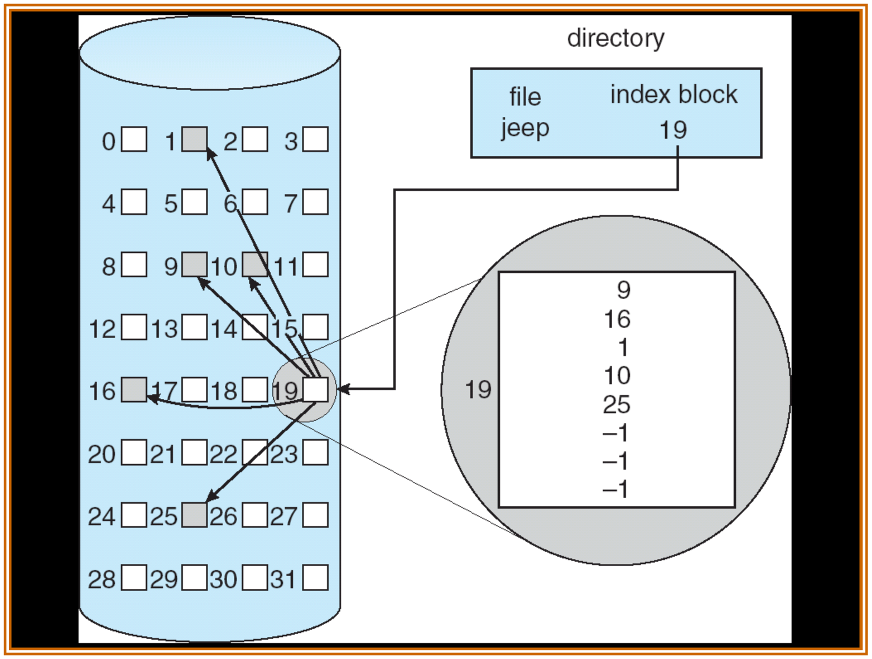
ALLOCATION METHODS

An allocation method refers to how disk blocks are allocated for files:

Contiguous allocation

Linked allocation

Indexed allocation



CONTIGUOUS ALLOCATION

Each file occupies a set of contiguous blocks on the disk

Simple – only starting location (block #) and length (number of blocks) are required

Random and sequential access are possible

CONTIGUOUS ALLOCATION

Wasteful of space (dynamic storage-allocation problem) external fragmentation. (first fit, best fit and worst fit)

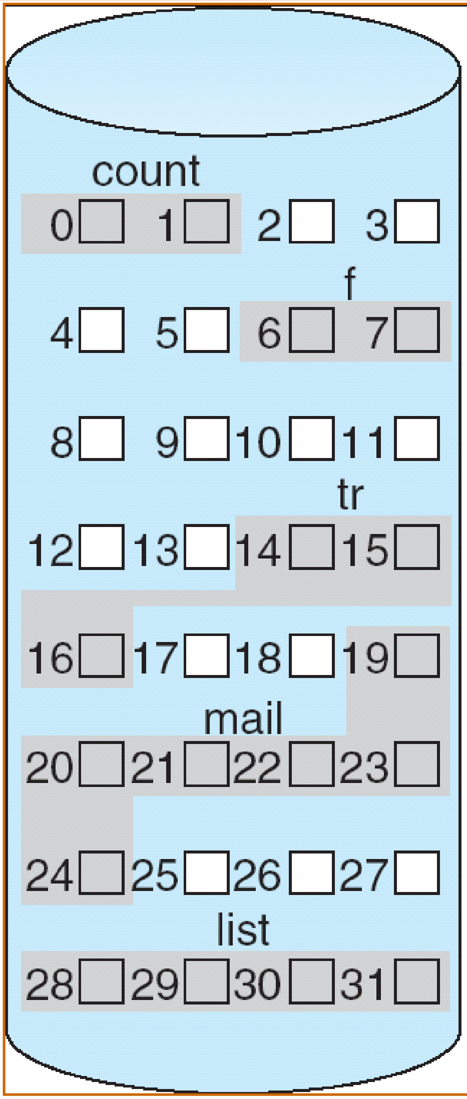
First fit – first hole big enough for the file

Best fit – smallest hole big enough for the file

Worst fit – biggest hole for the file

Solution: Compaction → time consuming.

CONTIGUOUS ALLOCATION OF DISK SPACE



directory

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

LINKED ALLOCATION

Each file is a linked list of disk blocks: blocks may be scattered anywhere on the disk. Directory contains a pointer to the first and the last block of the file.

LINKED ALLOCATION

File header points to 1st block on disk

Each block points to next

Example:

- FAT (MS-DOS)

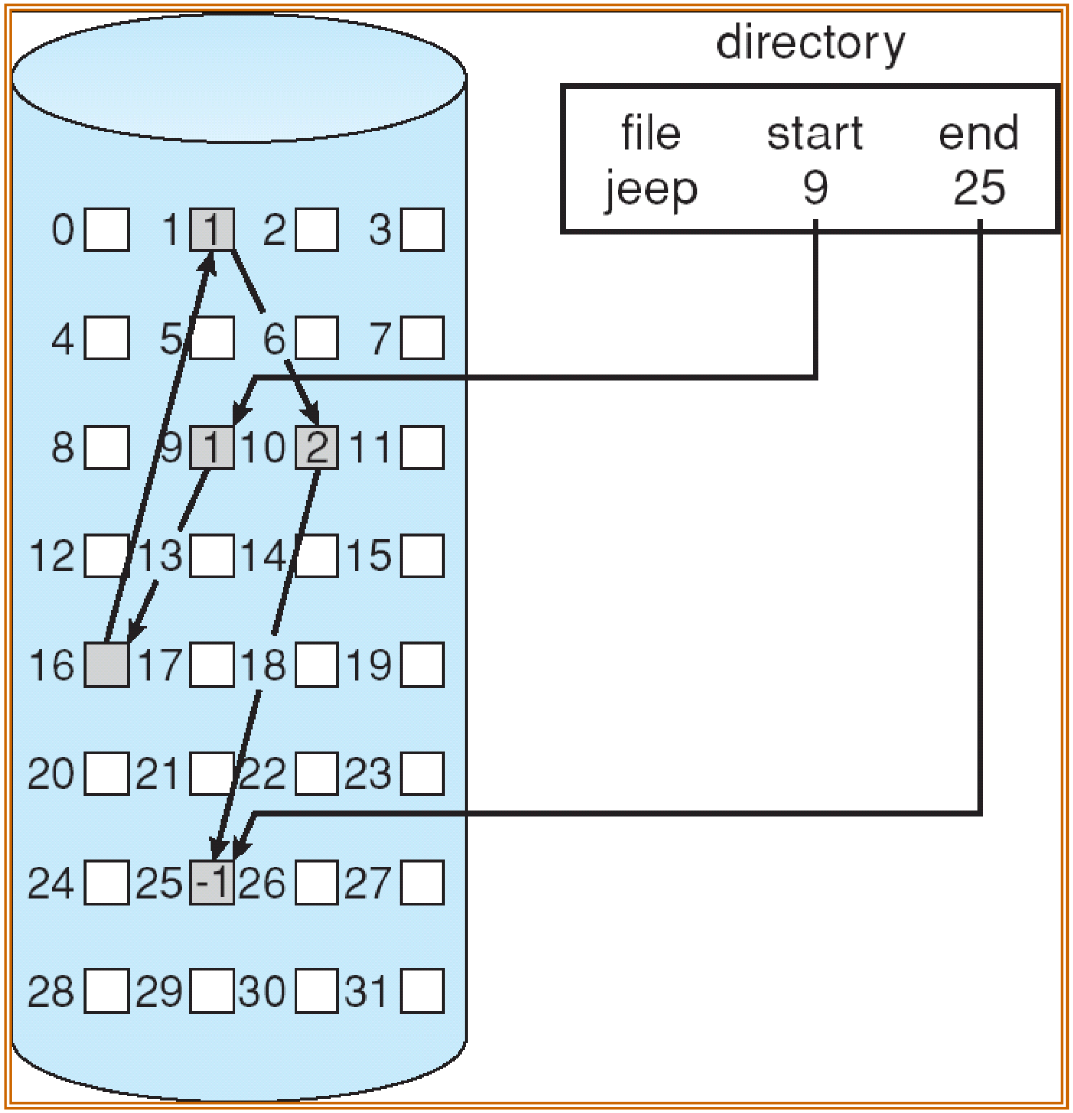
Pros

- Can grow files dynamically
- Space efficient, No external fragmentation but little internal fragmentation

Cons

- Random/direct access: horrible
- unreliable: if pointer was damaged or lost, losing a block means losing the rest
- Need some bytes to store pointers

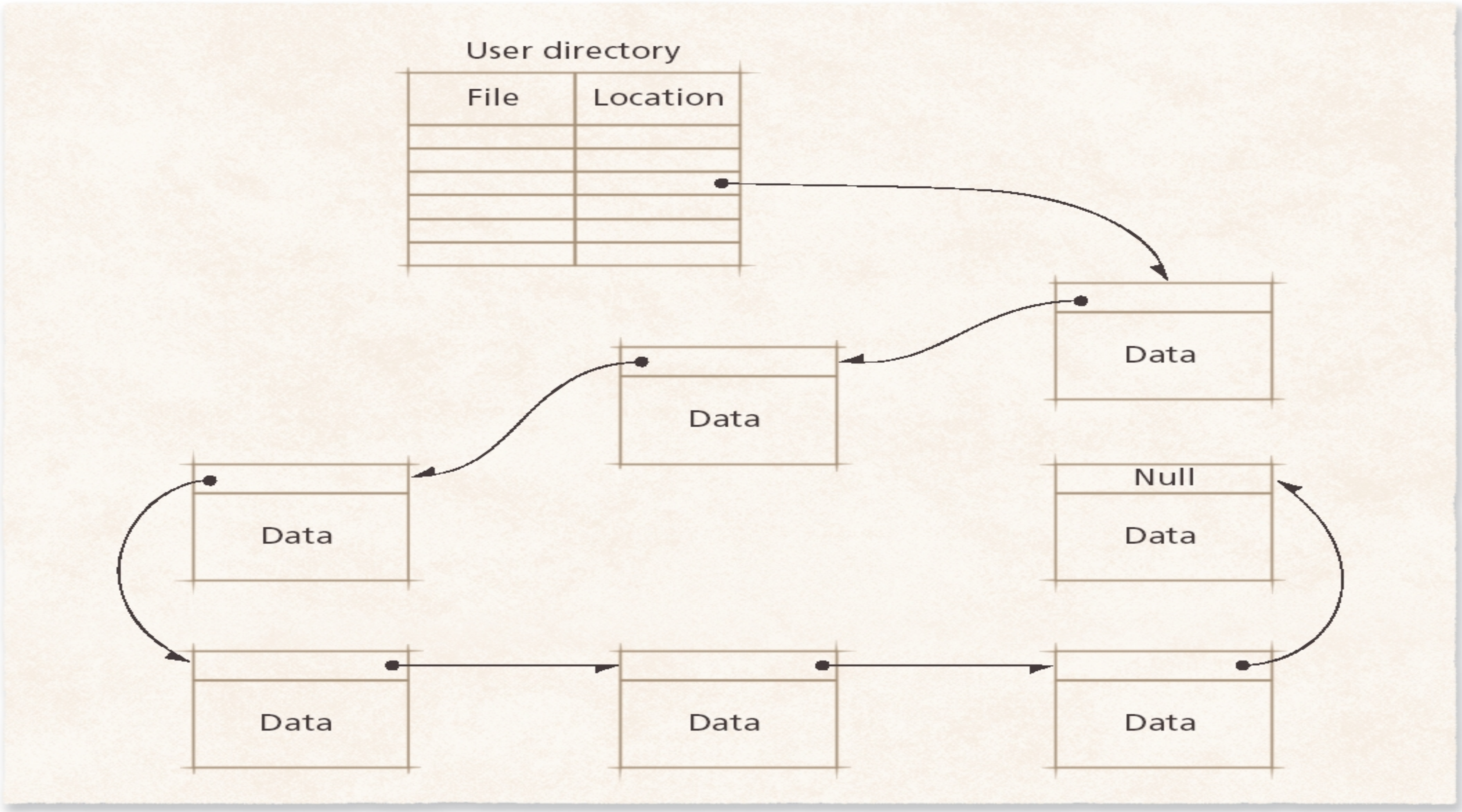
LINKED ALLOCATION



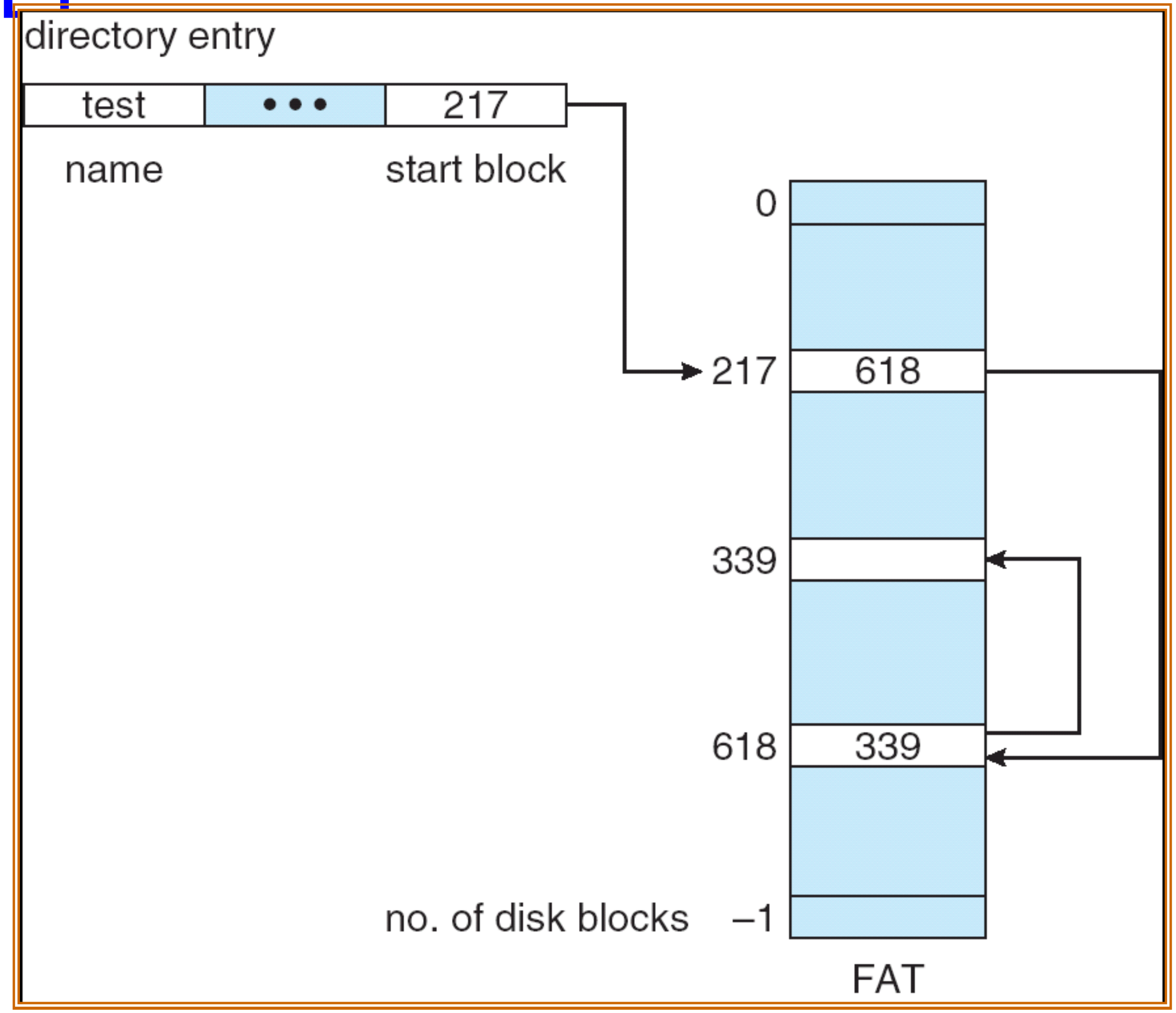
directory

file	start	end
jeep	9	25

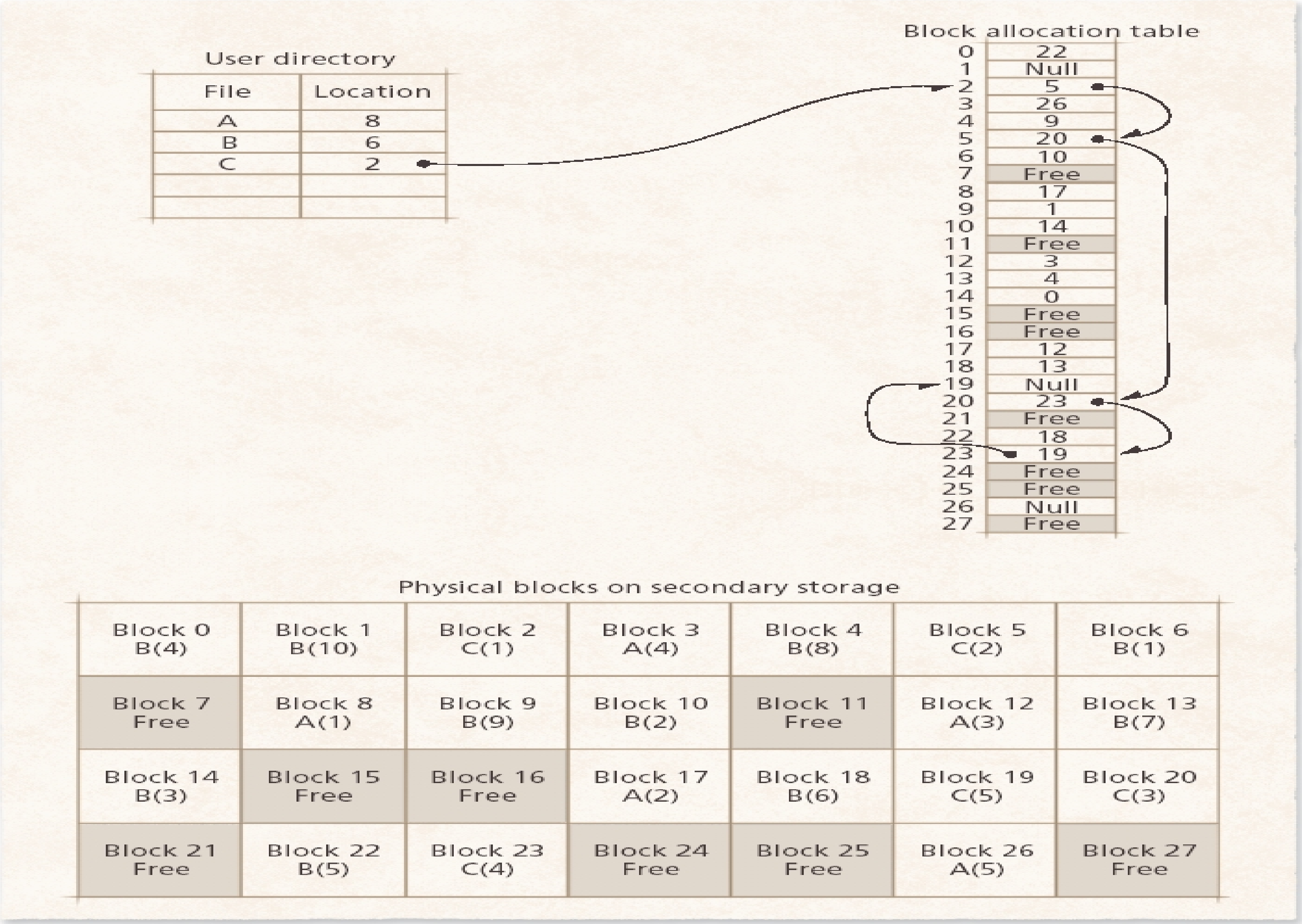
LINKED ALLOCATION



FILE-ALLOCATION TABLE



FILE-ALLOCATION TABLE



User directory

File	Location
A	8
B	6
C	2

Block allocation table

0	22
1	Null
2	5
3	26
4	9
5	20
6	10
7	Free
8	17
9	1
10	14
11	Free
12	3
13	4
14	0
15	Free
16	Free
17	12
18	13
19	Null
20	23
21	Free
22	18
23	19
24	Free
25	Free
26	Null
27	Free

Physical blocks on secondary storage

Block 0 B(4)	Block 1 B(10)	Block 2 C(1)	Block 3 A(4)	Block 4 B(8)	Block 5 C(2)	Block 6 B(1)
Block 7 Free	Block 8 A(1)	Block 9 B(9)	Block 10 B(2)	Block 11 Free	Block 12 A(3)	Block 13 B(7)
Block 14 B(3)	Block 15 Free	Block 16 Free	Block 17 A(2)	Block 18 B(6)	Block 19 C(5)	Block 20 C(3)
Block 21 Free	Block 22 B(5)	Block 23 C(4)	Block 24 Free	Block 25 Free	Block 26 A(5)	Block 27 Free

FILE-ALLOCATION TABLE

Used in MS-DOS and OS/2

A section of disk at the beginning of each partition contains the FAT

FILE-ALLOCATION TABLE

1. **FAT has one entry for each disk block**

2. **Directory entry for the file contains the first block number**

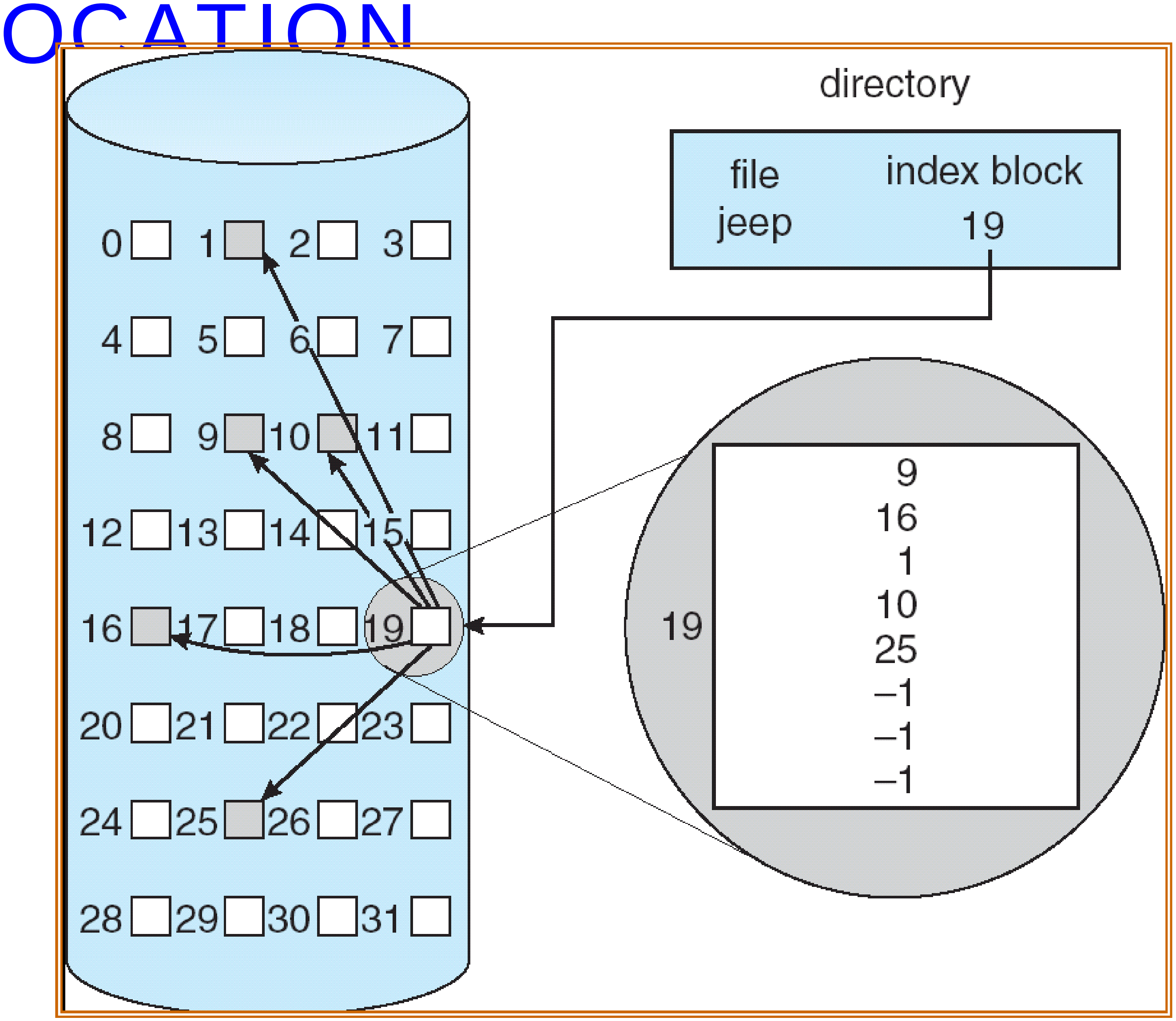
3. **The table entry for the first block number indicates the next block number of the file**

4. **Unused blocks are indicated by 0 table value**

INDEXED ALLOCATION

Brings all pointers together into the *index block.*

EXAMPLE OF INDEXED ALLOCATION



INDEXED ALLOCATION

Solves external fragmentation

Supports sequential and direct access

Access requires at most one access to index block first.

File can be extended by rewriting a few blocks and index block

Requires extra space for index block, possible wasted space.