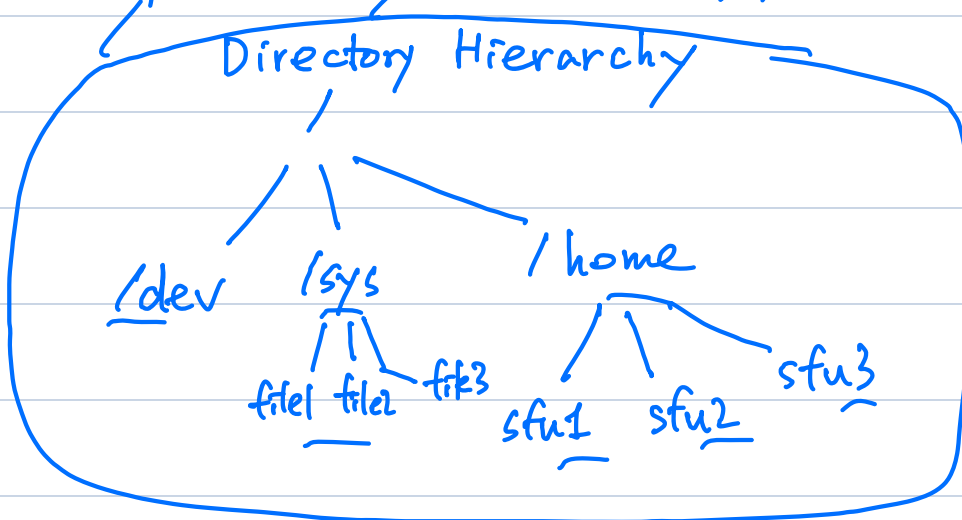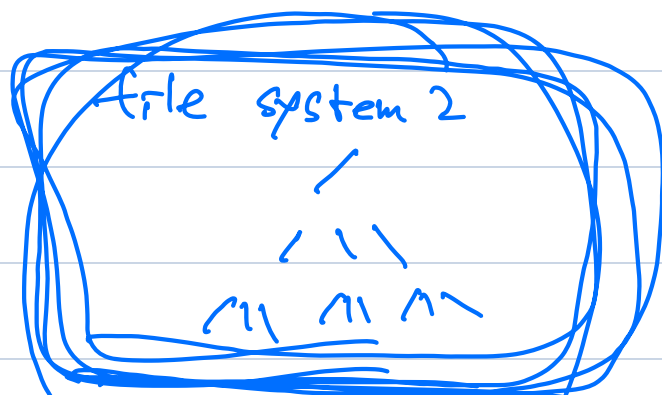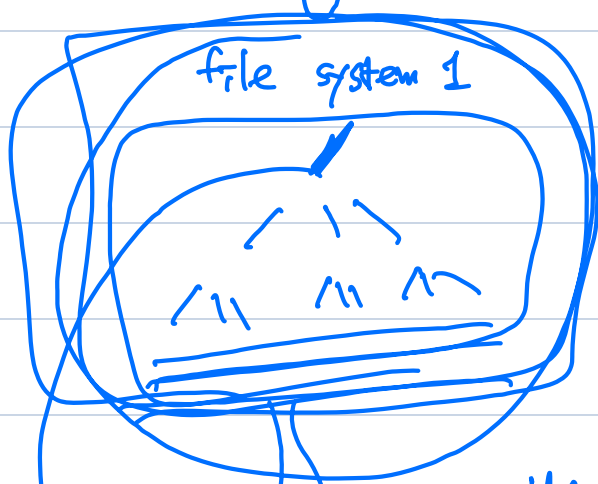Virtual File System.

syscall boundary
file abstraction.
syscalls. (open. close.)
read. write

← file systems
implement the
interface.

ext 4   ext 3   reiferfs.

Directory Hierarchy

/dev   /sys   /home

file1 file2 fik3   sfu1 sfu2   sfu3

mounting / unmounting

file system 1

file system 2

User-facing directory hierarchy

mounting done

file sycem 1

filesystem 2

file system 1
mount.

file system 2.

"mount" command

/dev

partition1   partition2   partition3

partition for a disk.

fs 1   fs 2   fs 3

"volumes.

fs

linux

windows
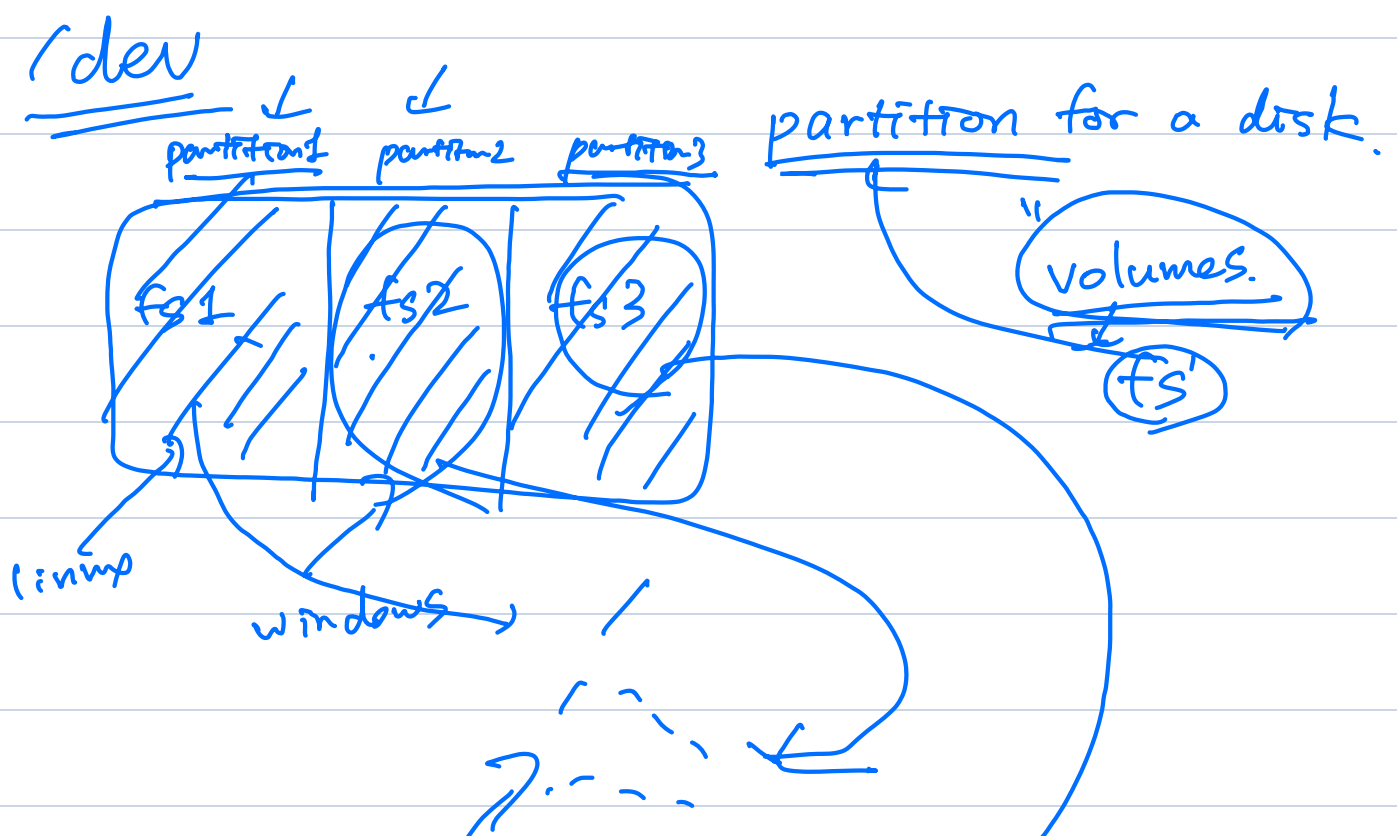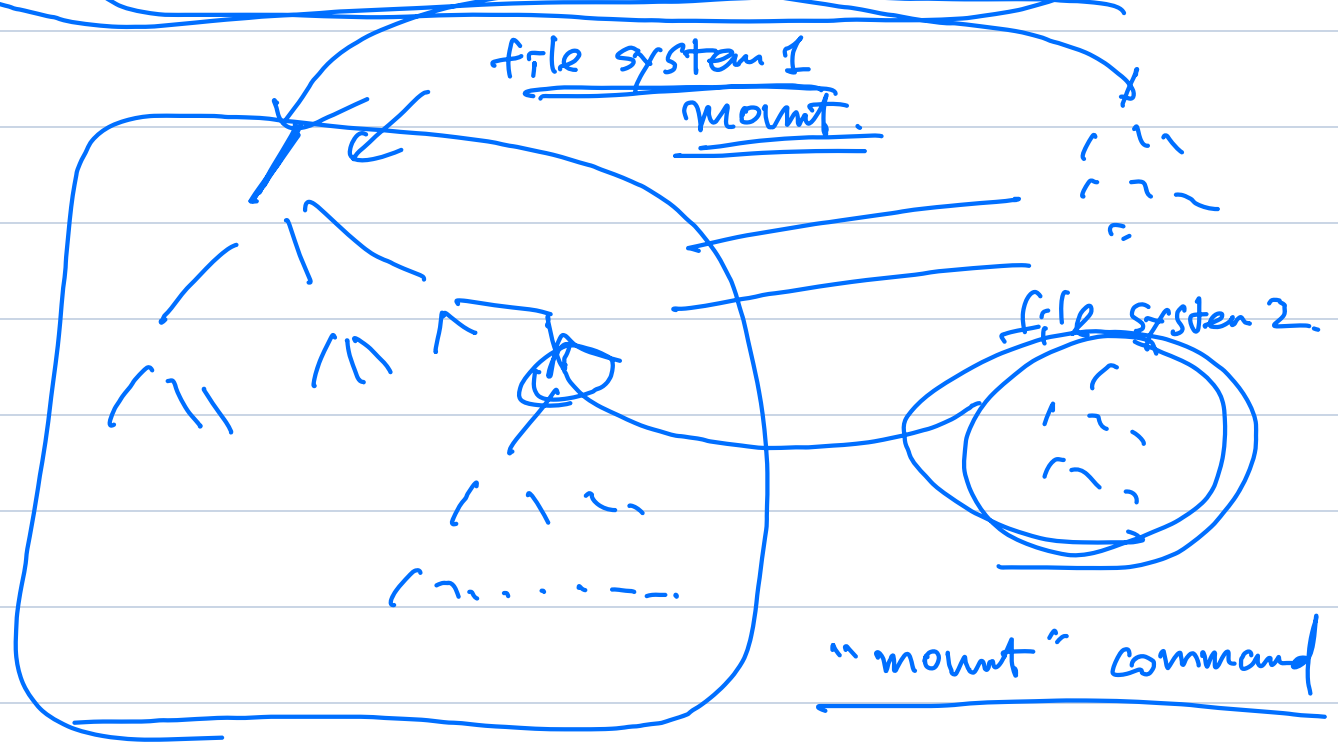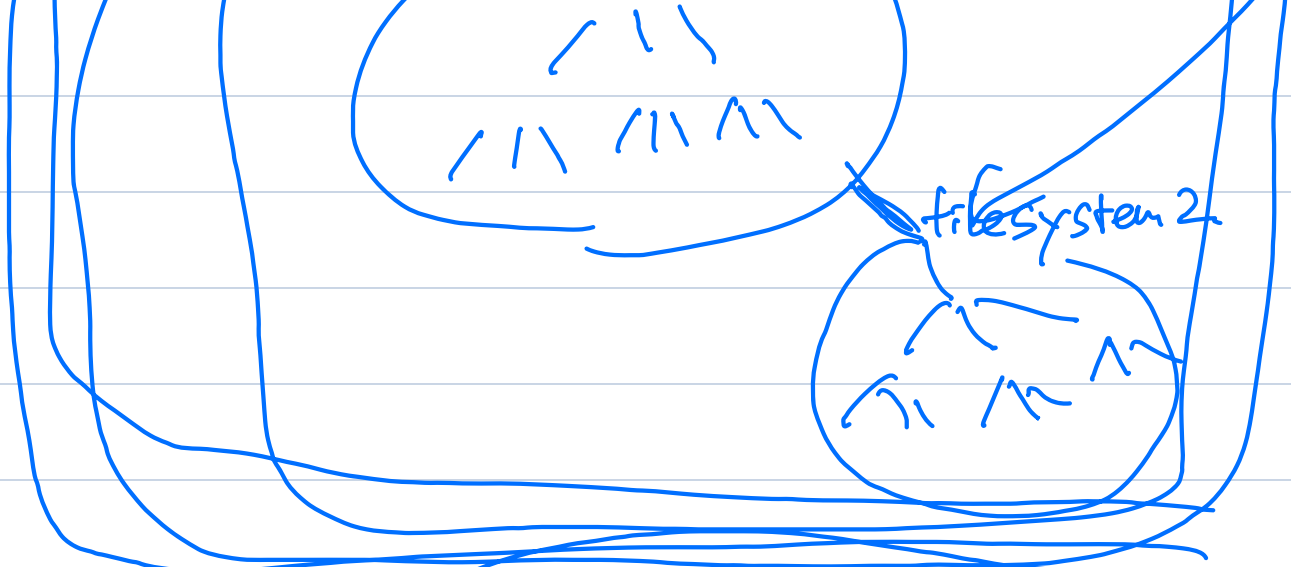
MBR

disk block : 4KB. (matches the page size)

▢ ▢ ▢▢ ▢▢ ▢ - - - - - - -

* Files. : data content
* Directories : considered files.
            content is the files that they contain.

(metadata : i-nodes

  data   : ⊕ file contents

(data blocks

(metadata blocks (i-nodes)

* Example
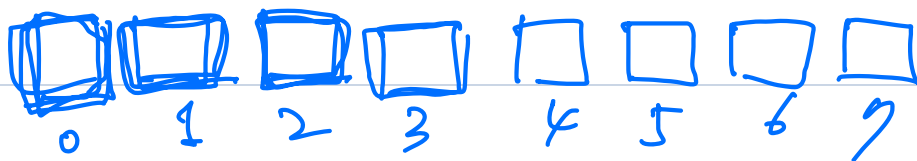
▢▢ ▢▢ ▢ ▢ ▢ ▢   : meta data.
0 1 2 3   4 5 6 7   * i-node block
                                 * i-bitmap. d-bitmap
                                 * super block
▢. ▢. - - - - -   : data blocks
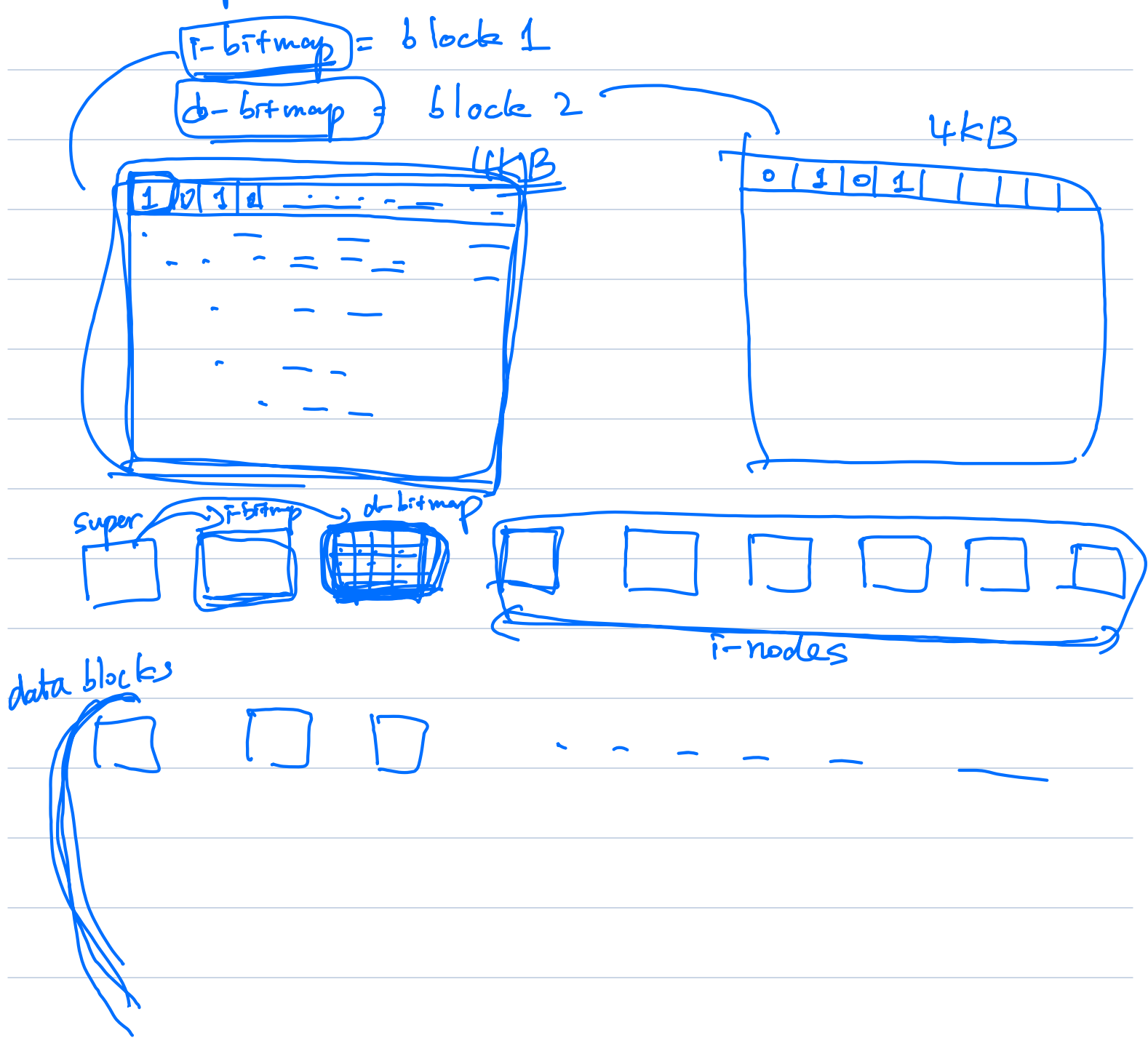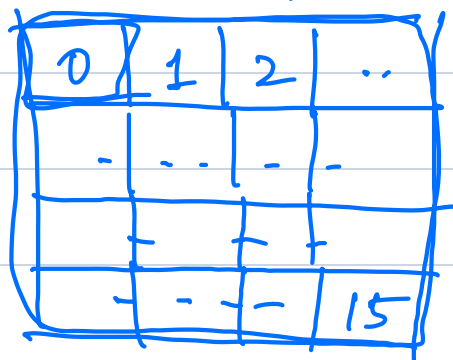8   9 - - - - -

* meta data blocks
    * (i-node blocks) ⟵
    * i-bitmap. d-bitmap ⟵
    * Superblock (first block)
              block 0

* Example.

i-bitmap = block 1

d-bitmap = block 2

4kB

0	1	0	1				

4kB

1	0	1	0				

Super   i-bitmap   d-bitmap

i-nodes

data blocks

+ i-node block

4K

0	1	2	..
-	-	-	-
-	-	-	15

256/4

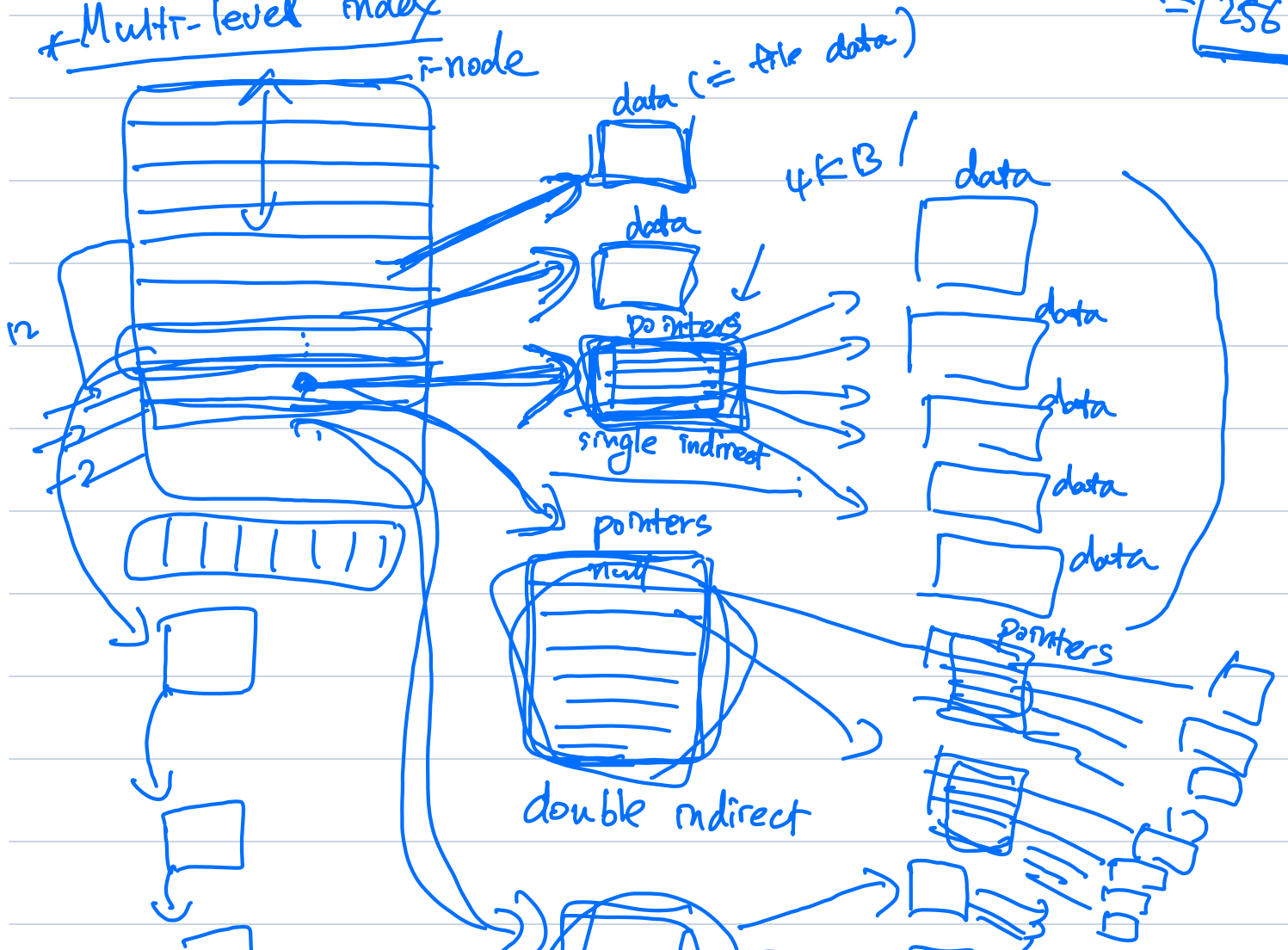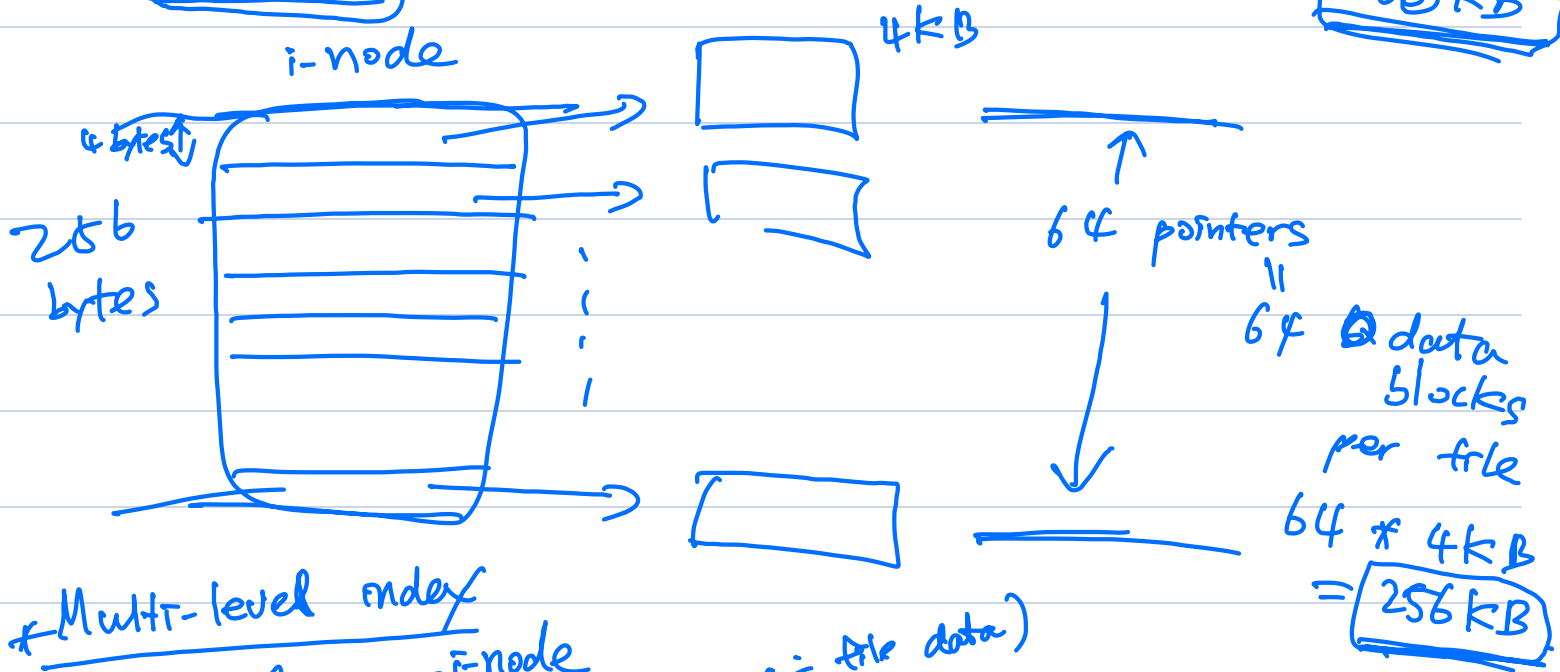i-node   256 bytes

+ size
* last accessed time
* time created
+ permission
  ⋮

+ file name is not stored.
(stored in the directory
   file that contains
   this file)

+ where the data blocks are

data block 0

data block 1

$64 \times 4kB$
$\Rightarrow$ 256 kB

i-node

4 bytes
256
bytes

4kB

64 pointers
=
64 data
blocks
per file

$64 \times 4kB$
= 256 KB

* Multi-level index

i-node

data (= file data)

4KB

data

data
pointers

single indirect

12

2

pointers

null

double indirect

data

data

data

data

data
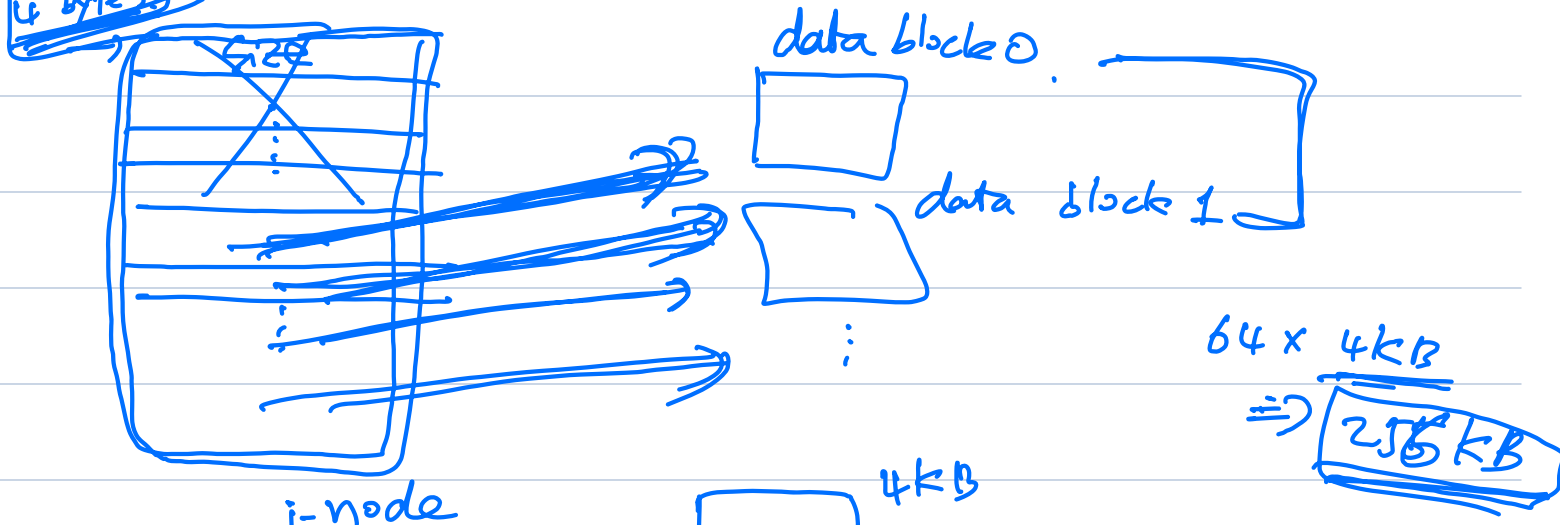
pointers

pointers

* Q: Assume block size: 4KB
pointer size : 4 bytes

→ 12 direct entries (pointers)
→ 1 single indirect entry (pointer)
→ 1 double indirect entry (pointer)
→ 1 triple indirect entry

what's the maximum file size?

A: 12 * 4KB   4KB block / 4 = 1K.
                                        1024 pointers.
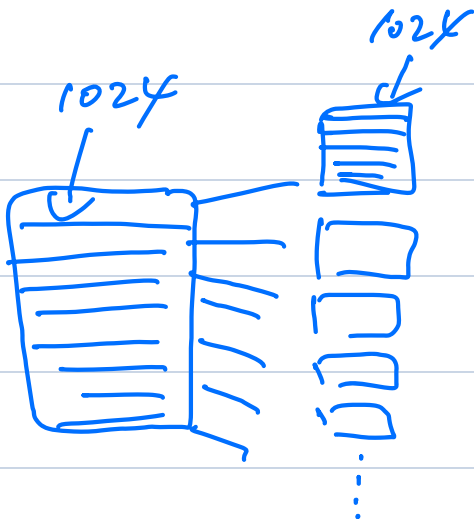   1024 * 4KB   4KB
     1024 * 1024 * 4KB   4bytes   4KB/4B
     1024 * 1024 * 1024 * 4KB   = 1K
                                                = 1024
                   "
                  4TB

     1024
          1024

          =   1024 * 1024

open:
*   / foo / bar   i-node #2

     read   root   i-node   fileo → i-node
     read   root   data block →   file1 → i-node
                                     foo → i-node

read foo r-node
read foo data block ⟶   file2 ⟶ i-node
                                      file3 ⟶ r-node
                                        bar ⟶ i-node

read bar i-node

(fd) = open ("/foo/bar");

        ⟶ bar r-node
    read (fd, buf: -)
    write (fd, buf. ---)

* read (fd).
    read   bar   i-node
    read   bar   data
    write   bar   i-node

* write (fd)
    read   bar   i-node
    read   data bitmap
    write   data bitmap
    write   data block
    write   bar   r-node

* create "/foo/bar"

    ● read root r-node
     read root data   foo ⟶ i-node
     read foo r-node
     read foo data ⟶   ~~bar~~

— read r-node bitmap
— write r-node bitmap

— write foo data   (bar → r-node

— read bar r-node
— write bar r-node   (initialization)
— write foo r-node

* caching. (write-back. w.
         (write-through