---------- seti 25/11 ----------

Persistenza:
tipicamente utilizziamo la parola "dischi" per riferirci a dispositivi a blocchi di memoria secondaria, perché storicamente erano dischi (anche se ora sono chip)
dispositivi a blocchi -> quando interagisco con un blocco, non posso solo leggere o scrivere un singolo byte ma devo leggere o scrivere l'intero blocco
tipicamente sono divisi in 512 byte l'uno (anche a 4K)

ogni indirizzo di memoria ram contiene dei byte -> sui dischi la questione è diversa

- per indirizzare una memoria molto grossa servono molti bit -> se raggruppo in blocchi i dati, bastano meno bit per indirizzare i vari blocchi

La grossa differenza tra un disco rispetto alla ram è la **persistenza** -> quando stacco la corrente non perdo il contenuto del disco, mentre perdo quello della ram
La memoria secondaria è quindi persistente, molto più economica ma molto più lenta

Nei sistemi unix c'è una cache, chiamata **buffer cache**, che contiene i blocchi del disco che abbiamo letto o che vogliamo scrivere

- quando scriviamo un settore, il settore finisce nella buffer cache e non viene scritto immediatamente su disco, così se viene modificato viene modificato solo in ram e finisce poi su disco -> problema: se saltasse la corrente e i dati non sono ancora stati salvati su disco, si perdono

un volume è una sequenza di blocchi indirizzabili (anche non contigui o appartenenti a dischi diversi) -> ad esempio un insieme di dischi economici

se vogliamo utilizzare un disco mettendoci ad esempio due sistemi operativi diversi, per farlo partizioniamo il disco.
per partizionare i pc, esistono due standard:

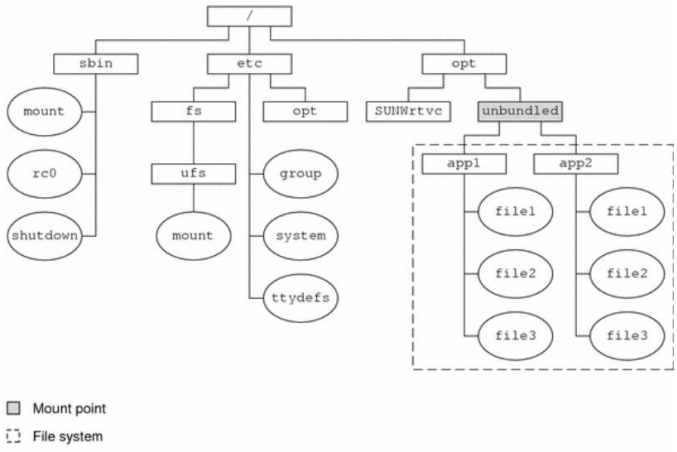
- MBR -> master boot record (4 partizioni primarie + partizioni estese)
- GPT -> GIUD partition table (128 partizioni identificate da un UUID -> universal unique identifier)

Nei sistemi unix-like esistono dei file speciali che fanno riferimento a dei dispositivi, che possono essere a caratteri (stampante, tastiera, ecc.) o a blocco

Questi file speciali che corrispondono a dispositivi possono essere creati con la system call mknod da root e sono identificati da due numeri:

- Il major number identifica il tipo di dispositivo (classe)
- Il minor number identifica uno dei vari dispositivi di quel tipo (istanze della classe)

Da utente, il file system viene visto come **un albero che ha una radice che è lo slash /, delle cartelle con all'interno dei file e altre cartelle ecc.**
Su sistema unix quando si aggiunge un disco, questo deve essere montato in una cartella del file system corrente
Su windows viene invece creata una lettera di unità (esempio c quella principale, attacco una chiavetta che diventa d, ecc.)



Questi file che corrispondono ai dispositivi storicamente vengono inseriti sotto la cartella /dev -> nei sistemi moderni il file che corrisponde al dispositivo viene creato on demand quando viene creato il dispositivo e viene fatto sparire quando lo stacco -> il kernel solleva degli eventi quando i dispositivi vengono collegati e scollegati

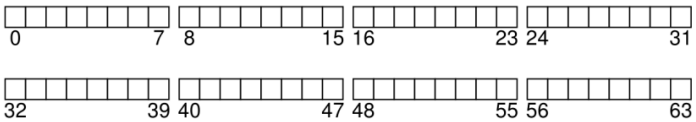
Implementazione di un file-system

Un file system è una struttura dati, che risiede su volume

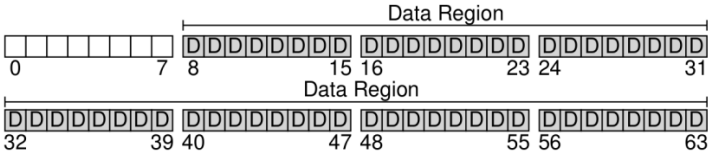
- Gestisce dati e metadati
- Quando formatto un volume sto andando a formalizzare questa struttura dati

Noi consideriamo una versione semplificata, vsfs (very simple file-system)
Esempio: assumiamo di avere un volume di 64 blocchi da 4K

- Spesso i settori si raggruppano in blocchi logici, chiamati **cluster**



non potremo utilizzare tutti i blocchi per i dati perché dobbiamo memorizzare anche altre cose, come i nomi dei file (quindi i metadati)



per ogni file c'è una struttura dati che si chiama inode, che contiene quasi tutti i metadati (tutti tranne i nomi dei file), e gli inode sono contenuti in una tabella chiamata tabella degli inode
quindi inode vado ad allocare quando formatto un disco? a seconda dell'utilizzo del disco può aver senso allocare più o meno inode

dobbiamo definire anche delle strutture dati:

- la bitmat degli inode
- la bitmat dei blocchi di dati

la bitmat è un settore di booleani -> per ogni inode ci sarà un bit nella bitmat che mi dirà se l'inode è occupato o libero, mentre nella bitmat dei dati si saprà se un blocco dati è utilizzato o libero

- superblocco -> intestazione del file-system che identifica le sue caratteristiche