For the operative system, a disk is a collection of data blocks that can be read or written independently. To allow their management, each one is characterised by the **Logical block address** (LBA), which is a unique numerical address.
To simplify the access, blocks are grouped into clusters, which are the smallest unit that an operative system can read or write on a disk.

They can contain file data, the actual content of a file, as well as metadata, which is the information required to support the file system and contain the file name, the owner, the permissions, the list of the blocks that contain the actual data, the size, the type, the creation, modification and last access dates.

So, the disk can contain four kinds of files:

- Fixed-position metadata, to bootstrap the file system;
- Variable-position metadata, to hold the folders structure;
- File data;
- Unused space.

The system can perform three actions on a file

1. Read

The metadata must be accessed to locate the blocks, then the real content is read from them

2. Write

The free space is located by accessing the metadata, then the file is written, but it is possible to access only clusters, so on each writing some space is lost due to the difference between the size of the clusters that must be written and the actual size of the file. This loss is called **internal fragmentation**.
Also, there might not be enough contiguous clusters to store all the file, so it is split and put into the free clusters, that are spread all over the disk causing the so-called **external fragmentation**.

3. Delete

It is the simplest operation as it needs to access only the metadata, the clusters on which the file was stored are added to the free list.

HDD: structure

It reads data in a random-access manner, meaning that data can be stored or retrieved in any order rather than sequentially.
An HDD is made of rigid rotating disks, each one of them has a moving actuator arm onto which are arranged magnetic heads, that read and write data to the surface; they float on a film of air above the platters and can be parked close to the centre or to the outer diameter, as in mobile devices.

Another component is the disk buffer cache, which is an embedded memory with the function of a buffer between the disk and the computer, it is large to avoid needing to stop the reading because the CPU is too slow in processing data.

The read/write atomic unit is called **sector**, sectors are numbered sequentially, are grouped into tracks and have a header and an error correction code.
Each one of them is identified by three numbers: cylinder (C), sector (S) and head (H); a function in the BIOS converts them into the LBA and vice versa.

$$LBA = (C \cdot heads\ per\ cylinder + H) \cdot sector\ per\ track + (S - 1)$$

$$C = \left\lceil \frac{LBA}{heads\ per\ cylinder \cdot sector\ per\ track} \right\rceil$$

$$H = \left\lfloor \frac{LBA}{sector\ per\ track} \right\rfloor \bmod head\ per\ cylinder$$
$$S = (LBA \bmod sector\ per\ track) + 1$$

The response time is the sum of the time spent in the queue waiting and of the service time, which is the time effectively spent to read the data.
We will evaluate only the latter, which is the sum of

- Seek time: head movement time;
- Rotational latency: the time to wait for the sector, which is assumed to be half a round;
- Data transfer time: a function of the rotation speed, the storing density and the cylinder position;
- Controller overhead: the time required to send the data.

Another quantity to consider is the data locality, which considers the fragmentation of the disk by telling the percentage of data that are stored contiguously. The real formula becomes:

$$service\ time = (1 - locality) \cdot (seek\ time + rot.latency) + tran.time + contr.overhead$$

If multiple blocks are considered, there are two ways to calculate the service time.

1. (easiest) consider the effect of the locality in the mean service time and multiply it with the number of I/O operations required to read/write the file;
2. Multiply the number of I/O operations affected by the locality for the service time with the locality and the other for the service time without the locality, then sum the two quantities.

$$total = affected \cdot (1 - locality) \cdot (tr. + contr. + seek + lat.) + unaffected \cdot locality \cdot (tr. + contr.)$$

SDD: structure

The Solid-State disk is a non-volatile storage device, it includes a controller and one or more solid state memory components.
Differently from the HDD, there is not a formula that translates the LBA into the physical page, the conversion is done through special look-up tables (LUT); this allows to dynamically modify the mapping, just changing the pointers into the LUT.

Data are stored in an array of NAND cells, which can be of two types:

- Single layer, that store only one bit;
- Multiple layer, that store more than one (typically two) bit by multilevel voltage, they have a higher capacity but a smaller data reliability.

Reading or writing and erasing cannot be done on the same quantity of data: the first two operations are allowed on pages, while the latter is possible only on blocks, which consists of multiple pages.

A page can be in three states:

- Empty: it is possible to write only on them,
- Dirty: the content has been marked as "can be deleted", but it has not been deleted yet,
- In use: it is meaningful to read them.

Reading a single page consists of transferring it to the data register and then outputting it, while reading more than one occurs in the "cache mode": from the data register, data are transferred to the cache, then there is a simultaneous transfer of the next page into the register and data output of the current page.

To mark a page as dirty, the **Trim** command can be used: it allows the operative system to tell the controller that certain pages are no more in use and can be marked as dirty.

The main drawback of this architecture is that a dirty block it must be erased, by resetting its voltage level to neutral, before writing new data, but the deleting must be done on an entire block. If there is no block that contains just dirty or empty pages, a special procedure is activated to cancel only the dirty pages:

- The whole block is read into the cache,
- The dirty pages are deleted (this operation can be done in the cache),
- Then the data are written on the cache,
- The whole block is copied again into the memory.

The procedure is clearly inefficient and leads to a high overhead when writing on an almost full disk.

Some other negative aspects are:

- Higher cost if compared with HDD
- The memory can be modified only a small number of times:
  - Shorter lifetime,
  - Error-correcting codes,
  - Over-provisioning
- Different read/write speed, as the write performances degrades with the writings
- The controller becomes the real bottleneck
- Data must not be defragmented, as it could damage the disk.

Comparison

Some key definitions related to SSD:

- Retention failure: a data error which occurs when the disk is read after a long period after the last write,
- Endurance failure: failure caused by endurance stressing,
- Endurance or TBW rating: number of terabytes that can be written while performances are still guaranteed,
- Write amplification factor: data written to the non-volatile memory divided by that written by the host to the SSD.

The only important quantity related to HDD is the unrecoverable bit error (RATIO), which is the rate of occurrence of data errors.

When comparing the disks, it can be noticed that the SSD is much better than the HDD at the early stages, but near the TBW is becomes worse than it, this is due to the fixed error rate of the HDD. For this reasin, hybrid solutions are the best:

- Some large storage servers use the SSD as a cache for several HDD,
- Solid State Hybrid disks (SS**H**D) combine a small SSD with a large HDD in a single unit.