

# SHELL

SYSTÈME DE FICHIERS

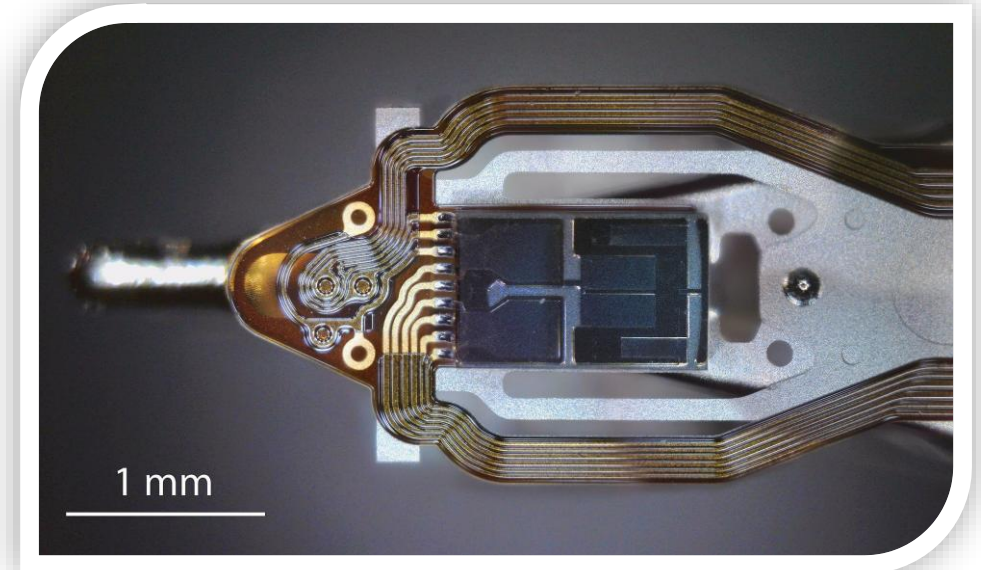
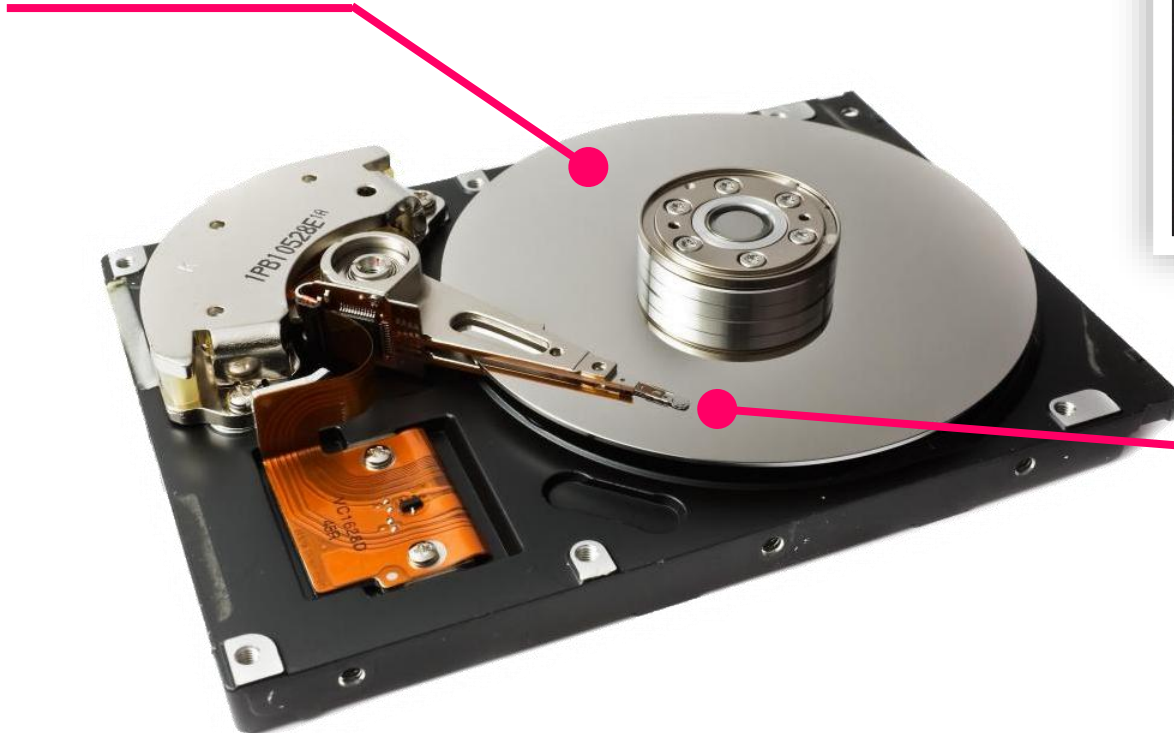




# STOCKAGE DE MASSE

# COMMENT FONCTIONNE UN DISQUE DUR ?

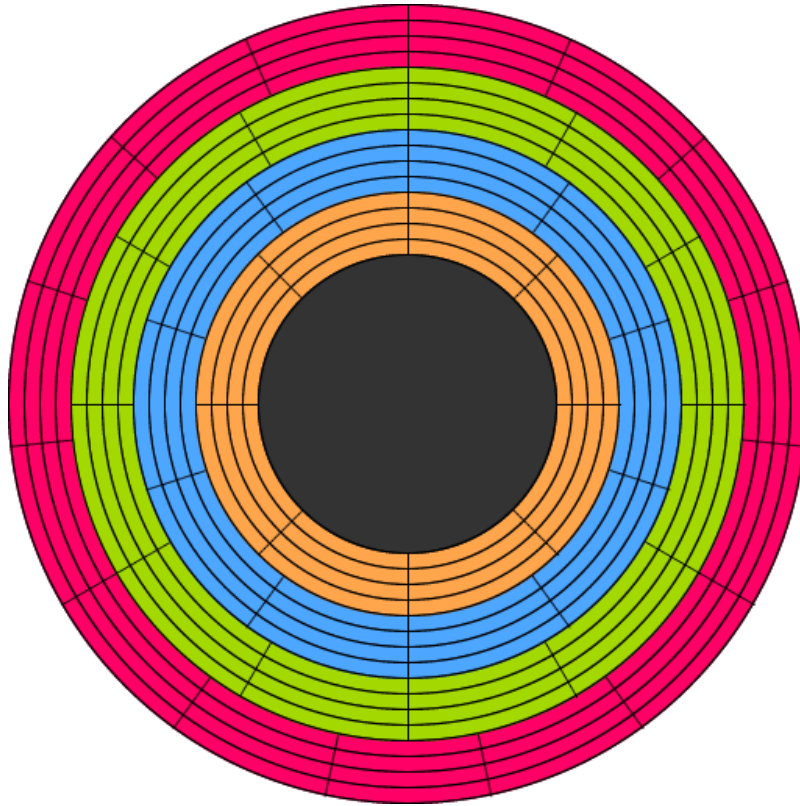
**PLATEAU**



**TÊTE  
DE LECTURE / ÉCRITURE**

# PLATEAU DU DISQUE DUR

Le plateau d'un disque dur est divisé en **PISTES** concentriques elles-mêmes divisées en **SECTEURS**.



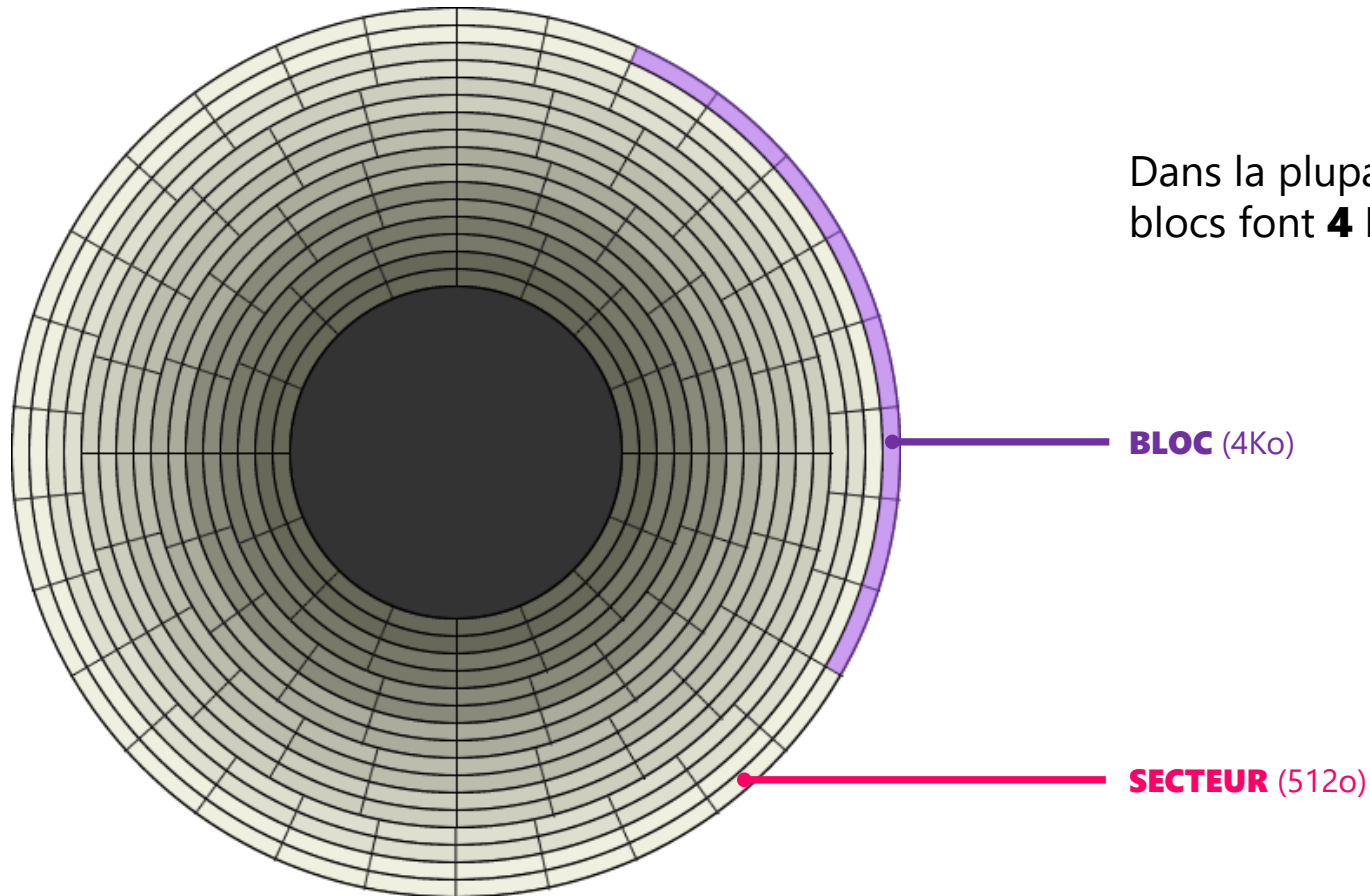
La densité des pistes augmente en s'écartant du centre du disque.

Un secteur a généralement une capacité de 512 octets (ancien) ou de 4096 octets (moderne) en fonction de la norme suivie.

Si le plateau est biface ou que le disque possède plusieurs plateaux, l'ensemble des pistes situées à une même position forment un **CYLINDRE**

# SECTEURS ET UNITÉS D'ALLOCATION

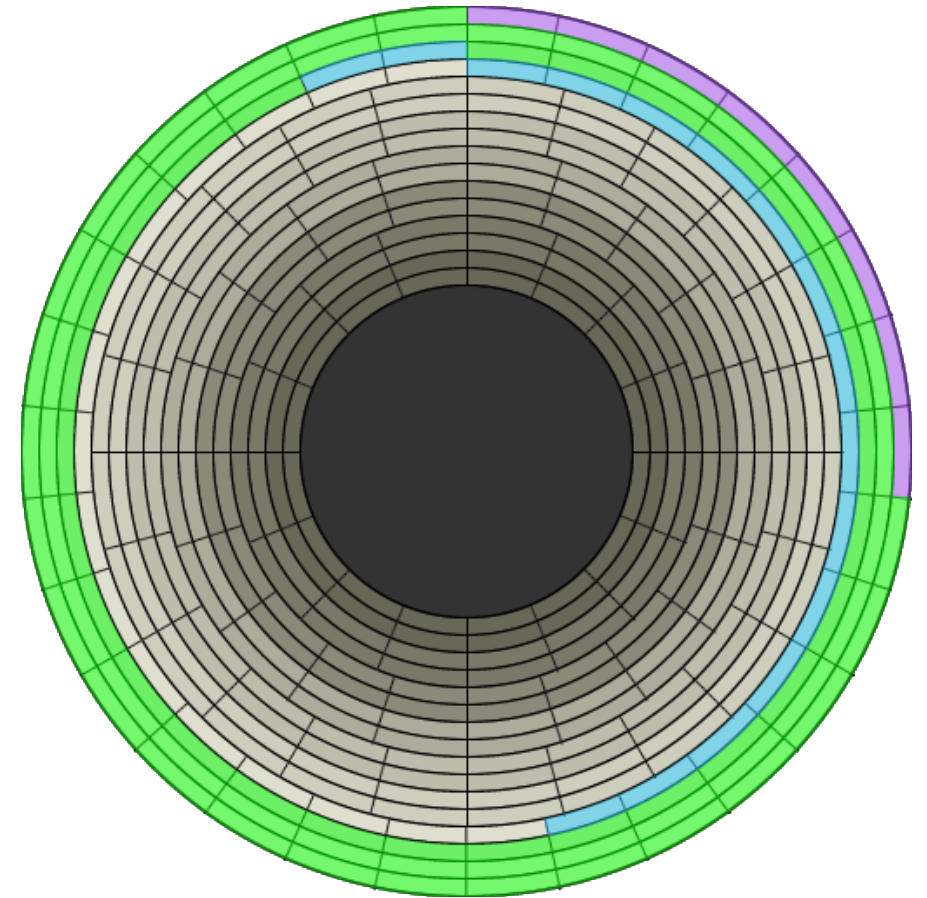
Le système de fichiers travaille avec des unités d'allocations ou blocs (clusters) qui regroupent plusieurs secteurs.



Dans la plupart des systèmes actuels, les blocs font **4 Ko** soient 8 secteurs de **512o**

Dans cet exemple **1 secteur = 512 o** et **1 bloc = 8 secteurs = 4 Ko**

Fichier	Taille réelle	Secteurs occupés	Blocs occupés	Taille sur le disque
index.html (🍆)	320 o	1	1	4 Ko
git.exe (🍏)	38,2 Ko	77	10	40 Ko
img.jpg (💧)	4,2 Ko	9	2	8 Ko

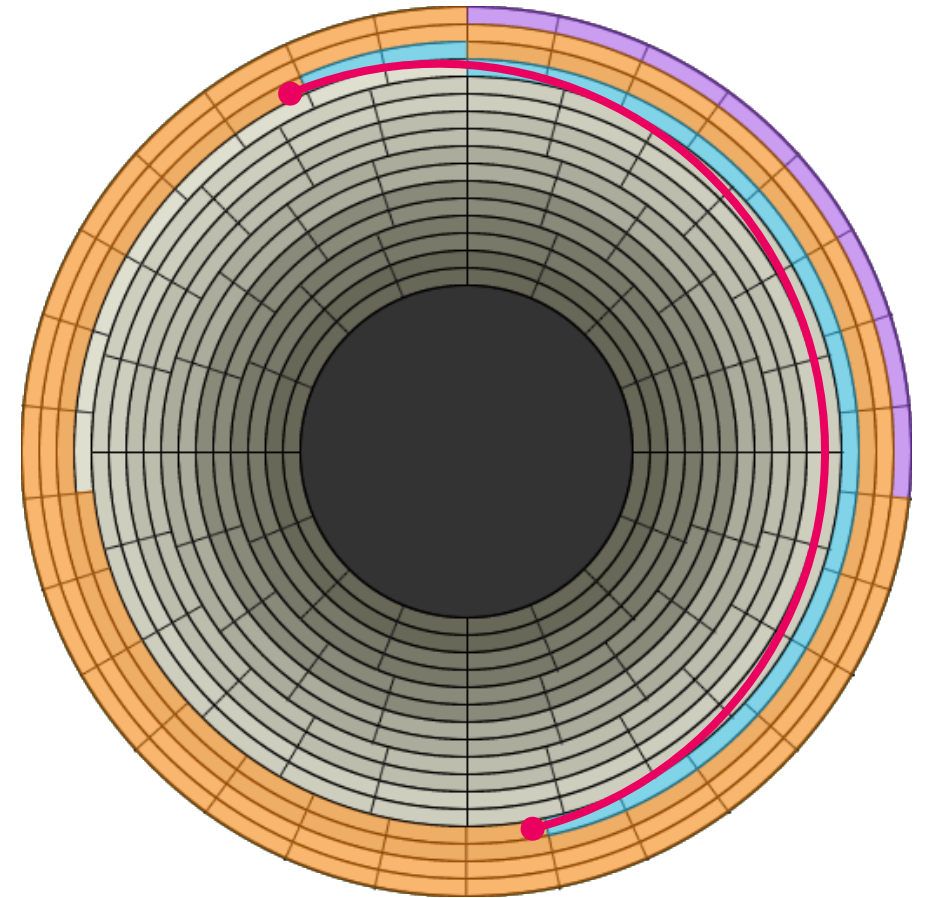


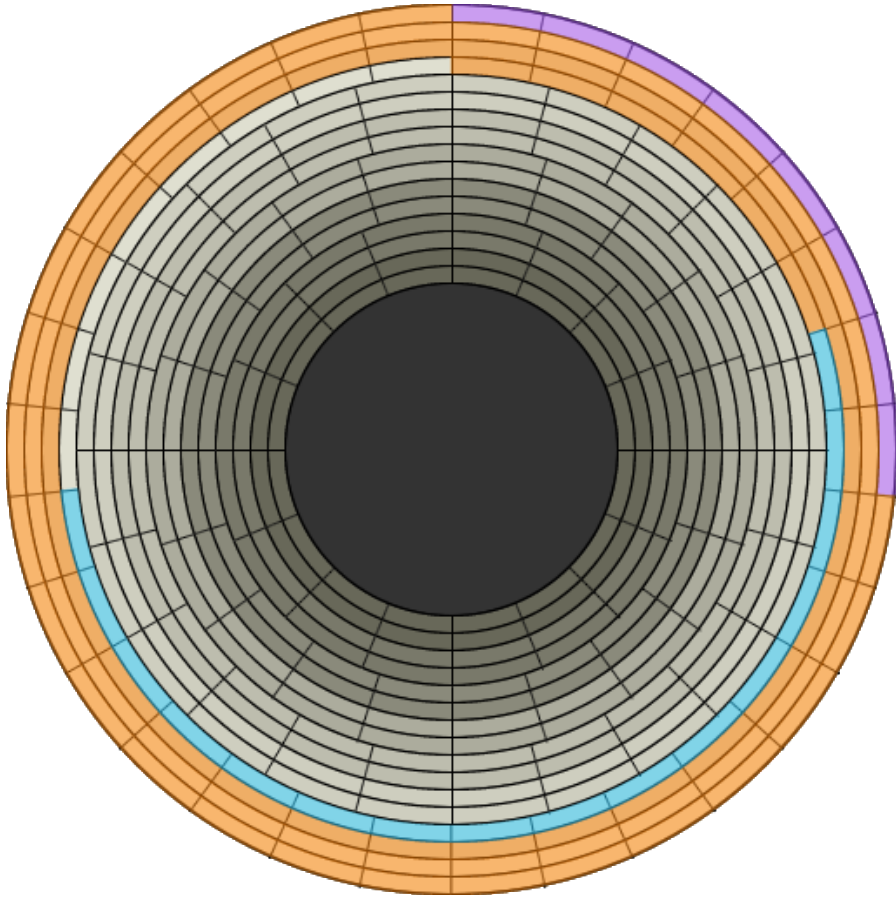


Dans cet exemple **1 secteur = 512 o** et **1 bloc = 8 secteurs = 4 Ko**

Fichier	Taille réelle	Secteurs occupés	Blocs occupés	Taille sur le disque
index.html (🟪)	320 o	1	1	4 Ko
<del>git.exe (🟩)</del>	<del>38,2 Ko</del>	<del>77</del>	<del>10</del>	<del>40 Ko</del>
img.jpg (💧)	4,2 Ko	9	2	8 Ko
svn.exe (🔴)	41,3 Ko	83	11	44 Ko

Le fichier est inscrit sur des secteurs qui ne sont pas contigus





1. Libérer les secteurs empêchant la continuité du fichier
2. Placer les informations fragmentées du fichiers de manière contiguë
3. Replacer les fichiers déplacés à la fin de l'espace disque occupé



# SOLID STATE DRIVE (SSD)



## Mémoire flash

Les données sont stockées dans des cellules mémoires

## Stockage électronique

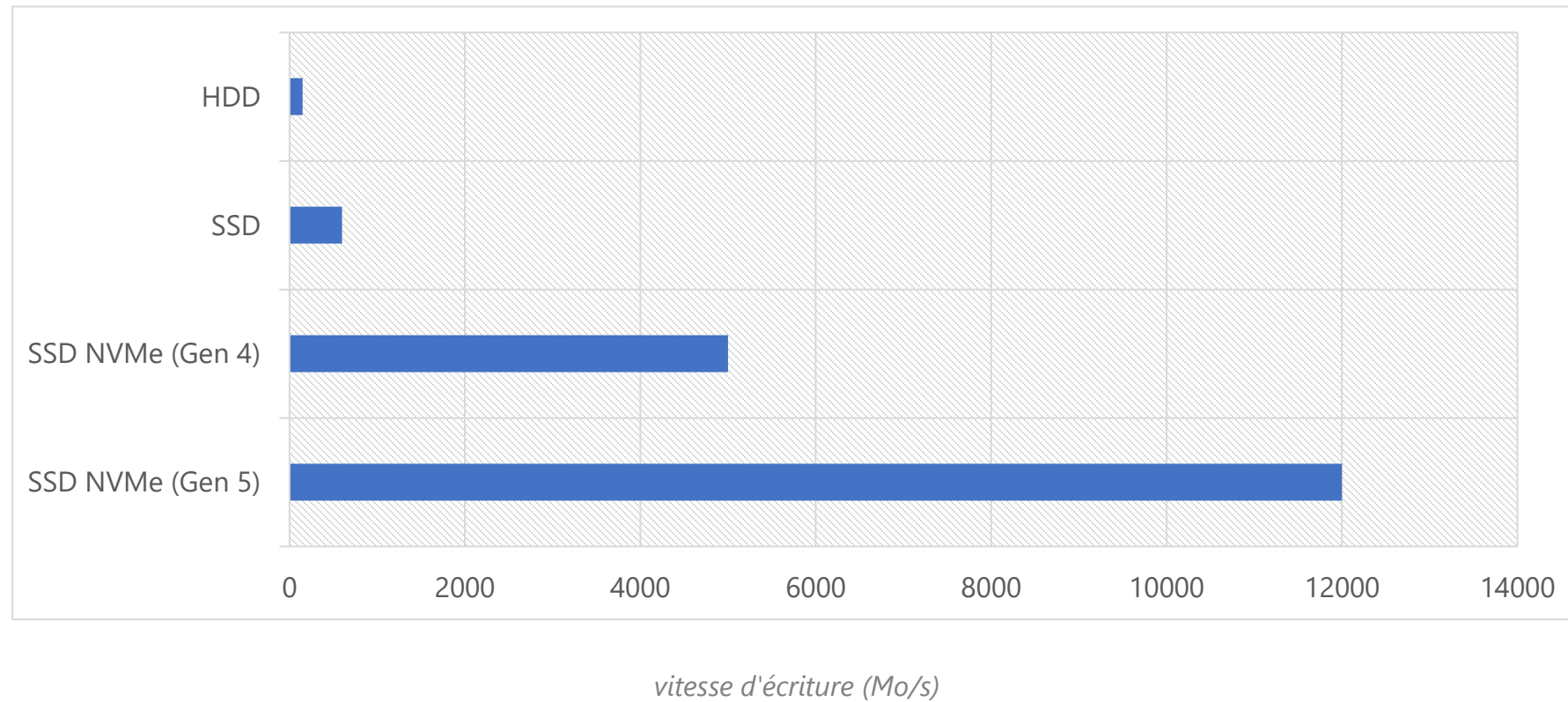
Pas de mécanique ni d'influence de la position physique de la donnée dans la mémoire

## Durée de vie limitée

Les cellules mémoires supportent un nombre limité de cycles d'écriture / effacement

## Wear Leveling

Pour éviter l'usure prématurée des cellules, le contrôleur SSD répartit les opérations d'écriture/effacement sur toutes les cellules mémoires.



## D'AUTRES SUPPORTS DE STOCKAGE ?

### BLU-RAY

50 Go de stockage

~25 € les 10



### BANDE MAGNÉTIQUE

Utilisée pour l'archivage de gros volume de données.

Faible coût

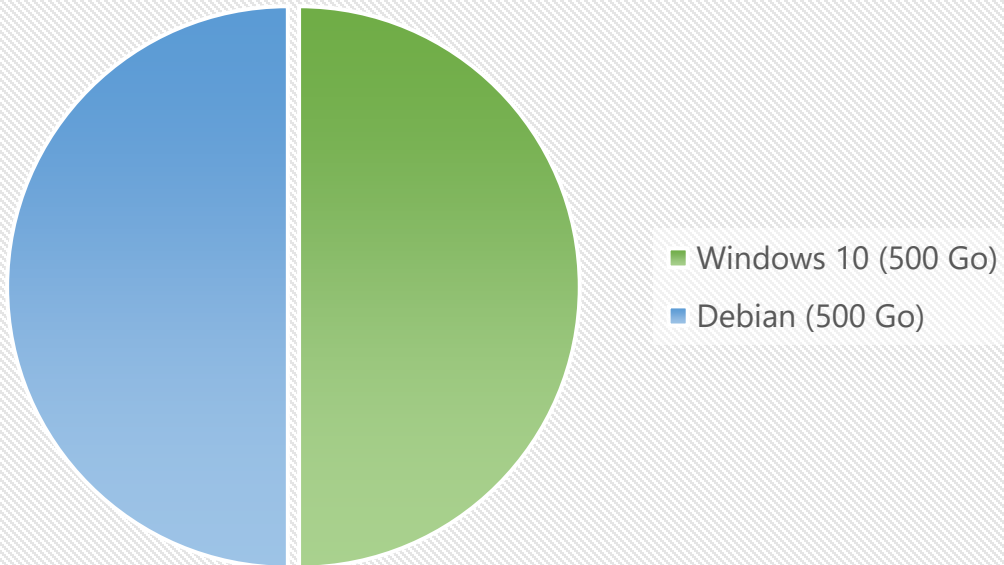
HDD 22 To = ~700 €

BM 30 To = ~80 €

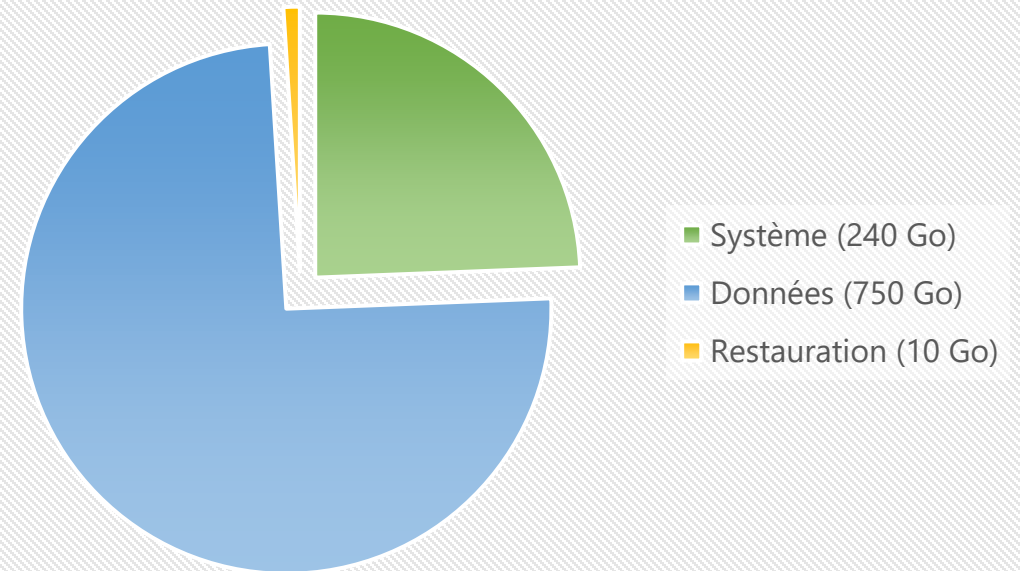
Accès séquentiel : il faut tout lire depuis le début pour trouver une information.

Les partitions regroupent les secteurs du disque dur au sein d'espaces virtuellement séparés.

### Cohabitation de plusieurs systèmes d'exploitation

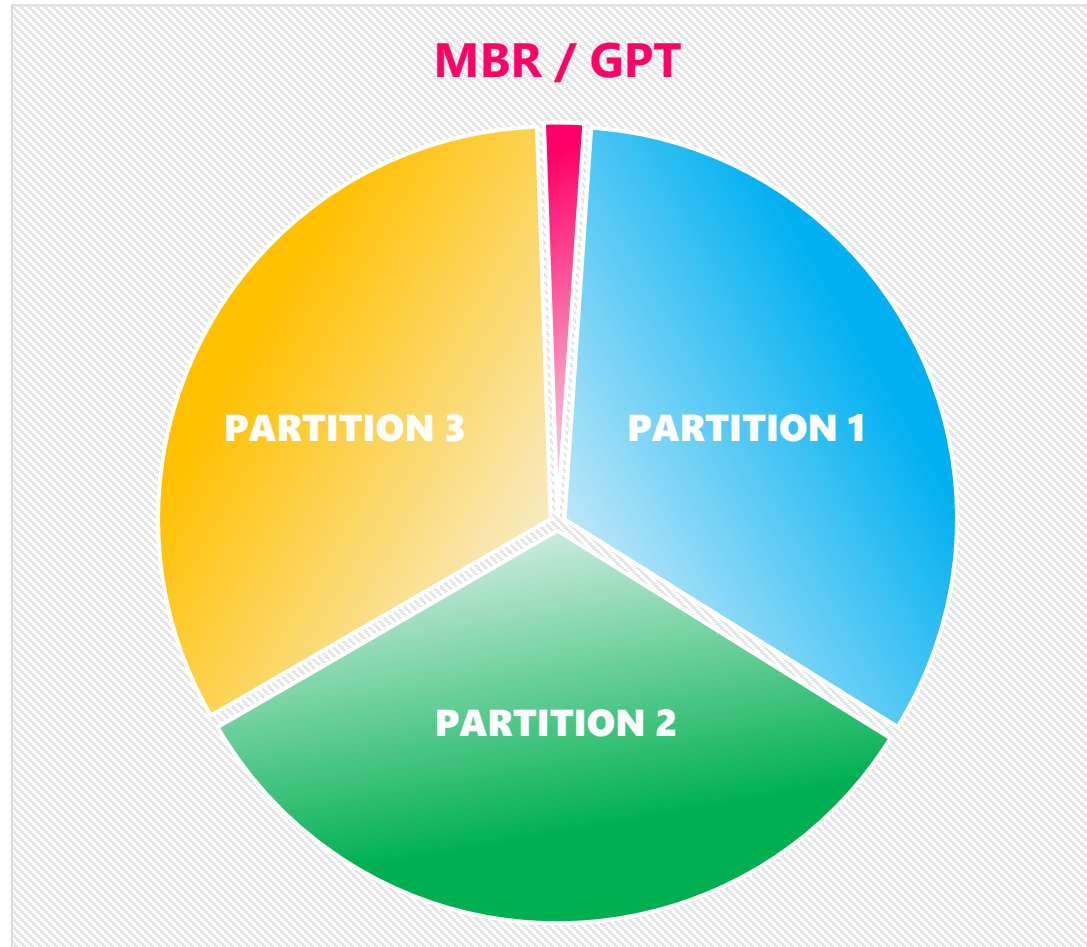


### Séparer les données du système d'exploitation



# ÉTAT DU DISQUE À CE STADE

Une fois tous les secteurs répartis dans différentes partitions, le disque dur se trouve dans l'état suivant :



Les premiers secteurs du disque dur sont utilisés pour stocker les informations sur le partitionnement du disque ainsi que le code d'amorçage (bootloader)

## MASTER BOOT RECORD

- 4 partitions max
- 2 To max / partition

## GUID PARTITION TABLE

- 128 partitions max
- 256 To max / partition



# SYSTÈME DE FICHIERS



L'opération de formatage consiste en l'installation d'un système de fichiers pour la gestion d'une partition.

A horizontal pink line with three solid pink circles at regular intervals, serving as a timeline for the steps.

## CHOIX

Chaque système d'exploitation peut prendre en charge plusieurs systèmes de fichiers. Il faudra choisir celui adapté à vos besoins.

En général, le système d'exploitation propose un système de fichiers par défaut (NTFS pour Windows, EXT4 pour Linux).

## IDENTIFICATION

Le système de fichiers identifie les secteurs de la partition qu'il va pouvoir utiliser pour stocker des données.

## CRÉATION

Le système de fichiers crée ensuite la structure qui lui permettra de manipuler les secteurs de la partition.

Regroupement des secteurs en blocs (clusters) pour augmenter les performances.

# LES PLUS COURANTS



**HFS+** (Hierarchical File System Plus)

**APFS** (APple File System)



**EXT4** (Fourth Extended File System)



**NTFS** (New Technology File System)



**FAT32** (File Allocation Table)

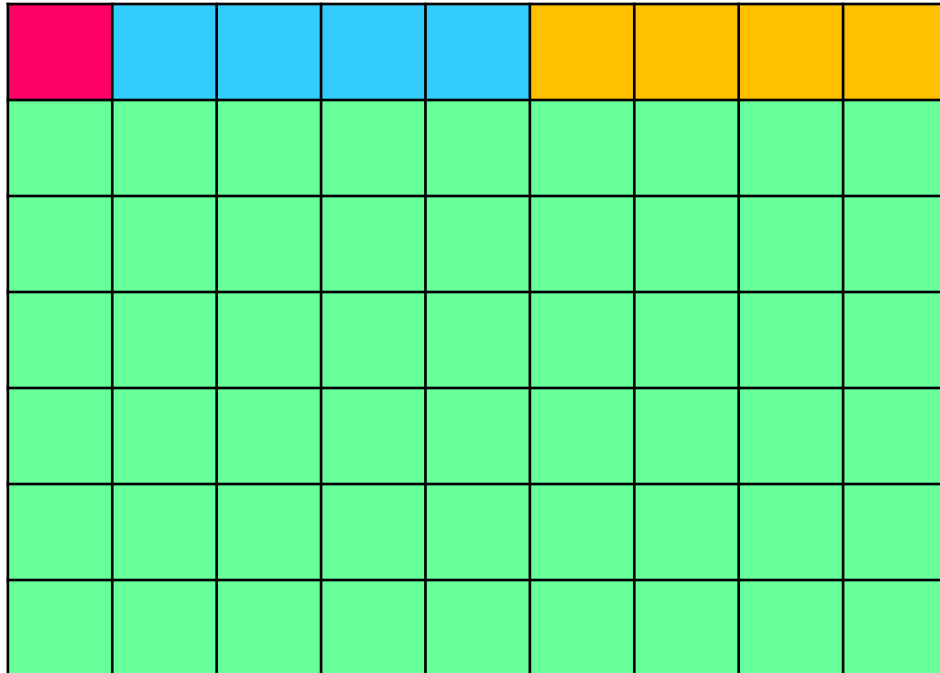
**EXFAT** (Extended File Allocation Table)



# **SYSTÈME DE FICHIERS**

## FONCTIONNEMENT DE FAT32

# FORMATAGE DE LA PARTITION



*Blocs de la partition formatée avec FAT32*

## BIOS PARAMETER BLOCK

Contient les informations sur la géométrie du disque (taille des secteurs, nombre de secteurs par cluster, nombre total de secteurs) ainsi que le programme de chargement du système d'exploitation (si un système est installé sur la partition).

## TABLE D'ALLOCATION DES FICHIERS

Secteurs contenant une entrée de 4 octets pour chaque bloc de la partition dédié au stockage des fichiers.

## COPIE DE LA FAT

Secteurs contenant une copie de la FAT en cas de corruption de cette dernière.

## ZONE DE DONNÉES

Blocs utilisés pour le stockage des fichiers.

# TABLE D'ALLOCATION DES FICHIERS (FAT)

Chaque entrée de 4 octets de la table d'allocation des fichiers décrit l'état d'un bloc :

<b>VIDE</b> 0x00000000	<b>DÉFECTUEUX</b> 0x0FFFFFFF7	<b>OCCUPÉ</b> 0x00000001 à 0x0FFFFFFF6 ou 0x0FFFFFFF
---------------------------	----------------------------------	--

0x0FFFFFFF indique que la fin du fichier se trouve sur ce bloc

Toute autre valeur représente l'adresse du bloc sur lequel trouver la suite du fichier

**FAT**

1	0x00000000
2	0x0FFFFFFF
3	0x00000004
4	0x0FFFFFFF
5	0x00000006
6	0x00000008
7	0x0FFFFFFF7
8	0x0FFFFFFF

**BLOCS DE STOCKAGE DE LA PARTITION**

0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08
<b>VIDE</b>	<b>Fichier</b> fin de fichier	<b>Fichier</b> se poursuit sur le bloc 0x04	<b>Fichier</b> fin de fichier	<b>Fichier</b> se poursuit sur le bloc 0x06	<b>Fichier</b> se poursuit sur le bloc 0x08	<b>DÉFECTUEUX</b>	<b>Fichier</b> fin de fichier

La table d'allocation ne contient pas les noms des fichiers



Les dossiers sont considérés comme des fichiers spéciaux contenant des entrées de 32 octets représentant un fichier ou un sous-dossier.



L'adresse du premier bloc est découpée en 2 x 2 octets. La première paires d'octets représente les octets de poids fort, la seconde les octets de poids faible.

Le dossier racine de la partition se trouve sur le premier bloc qui suit la copie de la table d'allocation.



CLE (E:)				
Nom	Date	Type	Taille	Mots clés
 code.js	05/05/2023 22:00	JSFile	6 Ko	
 img_12345.jpg	12/05/2020 09:44	Fichier JPG	3 935 Ko	

## Contenu de la clé USB

00046000	F8	FF	FF	0F	FF	FF	FF	FF	FF	FF	FF	0F	FF	FF	FF	0F	øYY.YYYYYYY.YYY.
00046010	FF	FF	FF	0F	FF	FF	FF	0F	07	00	00	00	FF	FF	FF	0F	YY.YYY....YY.
00046020	09	00	00	00	0A	00	00	00	0B	00	00	00	0C	00	00	00	.....
00046030	0D	00	00	00	0E	00	00	00	0F	00	00	00	10	00	00	00	.....
00046040	11	00	00	00	12	00	00	00	13	00	00	00	14	00	00	00	.....
00046050	15	00	00	00	16	00	00	00	17	00	00	00	18	00	00	00	.....
00046060	19	00	00	00	1A	00	00	00	1B	00	00	00	1C	00	00	00	.....
00046070	1D	00	00	00	1E	00	00	00	1F	00	00	00	20	00	00	00	.....
00046080	21	00	00	00	22	00	00	00	23	00	00	00	24	00	00	00	!..."....#...\$...
00046090	25	00	00	00	26	00	00	00	27	00	00	00	28	00	00	00	%...&...'...(...
000460A0	29	00	00	00	2A	00	00	00	2B	00	00	00	2C	00	00	00	)...*...+...,...
000460B0	2D	00	00	00	2E	00	00	00	2F	00	00	00	30	00	00	00	-...../...0...
000460C0	31	00	00	00	32	00	00	00	33	00	00	00	34	00	00	00	1...2...3...4...
000460D0	35	00	00	00	36	00	00	00	37	00	00	00	38	00	00	00	5...6...7...8...
000460E0	39	00	00	00	3A	00	00	00	3B	00	00	00	3C	00	00	00	9...:...;...<...
000460F0	3D	00	00	00	3E	00	00	00	3F	00	00	00	40	00	00	00	=...>...?...@...

Début de la table d'allocation

- Les deux premières entrées de la FAT ne sont pas utilisées pour l'allocation
- Première entrée correspondant à une allocation sur le cluster n°2 :  
Le **dossier racine**
- Début du fichier "code.js" sur le cluster n°6 qui se poursuit sur le cluster n°7
- Fin du fichier "code.js" sur le cluster n°7
- Extrait de la chaîne de clusters du fichier "img\_12345.jpg" qui commence au secteur n°8.

# ETUDE D'UNE ENTRÉE DE DOSSIER

Deux fichiers sur une clé USB formatée avec FAT32

```

00400080 43 4F 44 45 20 20 20 20 4A 53 20 20 18 27 2F B8 CODE JS .'/,
00400090 41 57 41 57 00 00 19 B0 A5 56 06 00 56 16 00 00 AWAW...°¥V..V...
004000A0 49 4D 41 47 45 20 20 20 4A 50 47 20 18 29 2F B8 IMAGE JPG .)/,
004000B0 41 57 41 57 00 00 86 4D AC 50 08 00 44 7A 3D 00 AWAW...†M→P..Dz=.
  
```

*Les deux entrées correspondant aux fichiers dans le dossier racine de la clé*

	ENTRÉE 1	ENTRÉE 2
<b>NOM + EXTENSION</b>	CODE + JS	IMAGE + JPG
<b>ATTRIBUTS</b>	0x20 = Fichier	0x20 = Fichier
<b>CLUSTER DE DÉBUT (POIDS FORT)</b>	0x0000	0x0000
<b>CLUSTER DE DÉBUT (POIDS FAIBLE)</b>	0x0006	0x0008
Cluster de début (1 cluster 4 Ko = 8 secteurs 512 o)	0x00000006 = cluster n°6	0x00000008 = cluster n°8
<b>TAILLE DU FICHIER</b>	0x00001656 = 5718 o	0x003D7A44 = 3935 Ko

# LES LIMITES DE FAT32

**TAILLE DES  
PARTITIONS**

limitée à 2 To

**TAILLE DES  
FICHIERS**

limitée à 4 Go

**GESTION DES  
DROITS**

aucune



# EXTENDED FILE SYSTEM

PRÉSENTATION RAPIDE

Système de fichiers par défaut de Linux

## **SUPER BLOCK**

Information sur le  
système de fichiers

## **BLOCK**

Regroupement de  
secteurs

## **BLOCKS GROUP**

Regroupement de blocs

## **INODES**

Information sur les  
fichiers et les dossiers

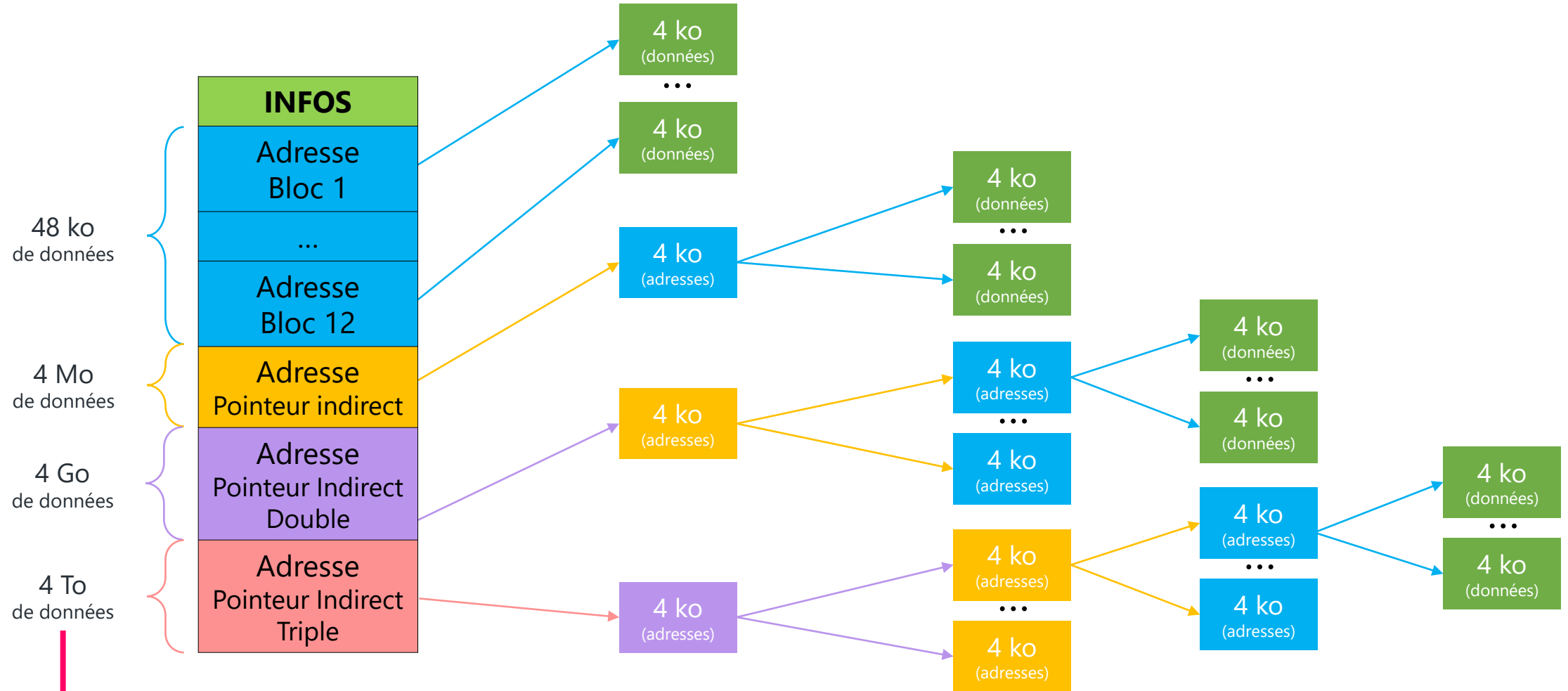
Objectifs :

- Réduire la fragmentation
- Optimiser les temps d'accès
- Augmenter la fiabilité

- Informations du le fichier (type, propriétaire, droits d'accès, date de création, ...)
- 12 pointeurs directs vers les blocs de données
- 1 pointeur indirect
- 1 pointeur indirect double
- 1 pointeur indirect triple

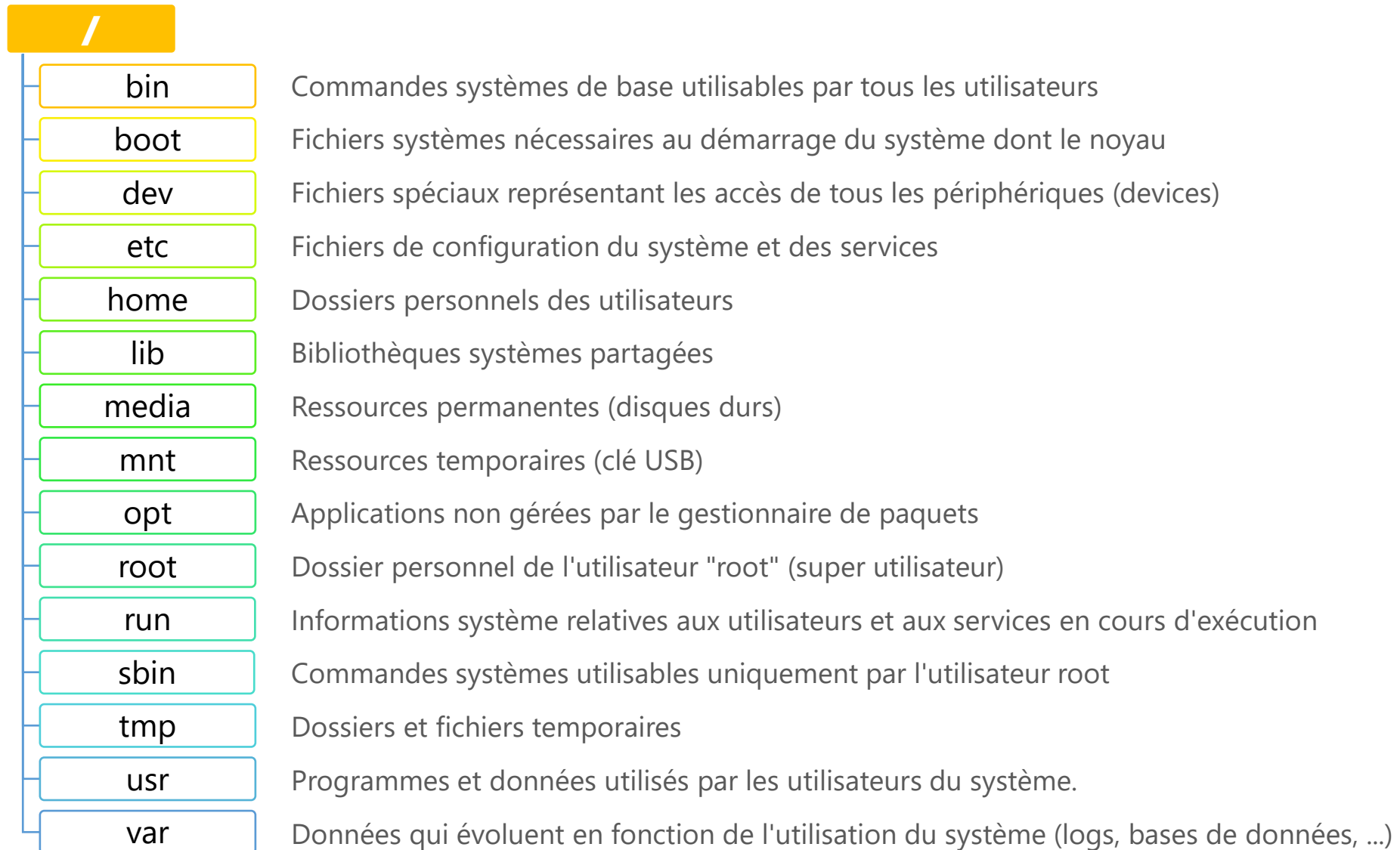
} remplacés par les **EXTENTS** sur EXT4





En réalité des contraintes limites à 2 To max sur ext3

# RAPPEL ARBORESCENCE LINUX





# **DROITS UTILISATEURS**

# TYPE D'UTILISATEURS

## PROPRIÉTAIRE

(u)

L'utilisateur propriétaire  
d'un fichier ou d'un dossier

## GROUPE

(g)

Le groupe d'utilisateurs  
associé à un fichier ou un  
dossier

## LES AUTRES

(o)

Ceux qui ne sont pas  
propriétaire et qui ne font  
pas partie du groupe  
associé au fichier ou au  
dossier

# TYPE DE DROITS

## LECTURE

$r = 4$

Le droit de lire le contenu  
d'un fichier

## ECRITURE

$w = 2$

Le droit de modifier le  
contenu d'un fichier

## EXÉCUTION

$x = 1$

Le droit d'**exécuter un  
fichier** ou de **traverser un  
dossier**

# REPRÉSENTATION OCTALE

u			g			o		
r	w	x	r	w	x	r	w	x
4	2	1	4	2	1	4	2	1

700 :     $[u] = 7 = 4 + 2 + 1 = r + w + x$   
            $[g] = 0$   
            $[o] = 0$

} Le propriétaire a tous les droits,  
 Le groupe et les autres n'en ont aucun

744 :     $[u] = 7 = 4 + 2 + 1 = r + w + x$   
            $[g] = 4 = 4 + 0 + 0 = r$   
            $[o] = 4 = 4 + 0 + 0 = r$

} Le propriétaire a tous les droits,  
 Le groupe et les autres n'ont que le droit  
 d'accès en lecture



Définir les droits du fichier avec une forme octale

```
chmod 744 index.html
```

Ajouter le droit d'exécution au fichier pour les membres du groupe associé

```
chmod g+x index.html
```

Ajouter le droit d'écriture pour les membres du groupe associé et retirer le droit en lecture pour les autres utilisateurs

```
chmod g+w,o-r index.html
```

Il faut être propriétaire du fichier ou du dossier ou administrateur pour pouvoir utiliser la commande chmod

L'option "-R" permet d'appliquer les modifications à tous les éléments contenus dans un dossier.

# MODIFIER LE PROPRIÉTAIRE ET LE GROUPE ASSOCIÉ

Faire de l'utilisateur "eric" le propriétaire du fichier "index.html"

```
chown eric index.html
```

Associer le groupe d'utilisateurs "esirem3a" au fichier "index.html"

```
chgrp esirem3a index.html
```

Modifier le propriétaire et le groupe associé d'un fichier

```
chown adipanda:direction index.html
```

L'option "-R" permet d'appliquer les modifications à tous les éléments contenus dans un dossier.



# SCRIPTS SHELL



## **FICHER TEXTE**

Cela ne veut pas dire qu'il a une extension ".txt" mais que ce n'est pas un binaire.

## **CONTENANT DES COMMANDES SHELL**

Mais aussi des structures de contrôles permettant de traiter des tâches complexes.

## **ET QUI POSSÈDE LE DROIT D'EXÉCUTION POUR AU MOINS UN TYPE D'UTILISATEUR**

En général, le propriétaire.

# CRÉER UN SCRIPT SHELL

## CRÉER UN FICHIER

```
nano mon_script.sh
```

## IDENTIFIER LE SHELL À UTILISER

```
#!/bin/bash
```

## SAISIR LES COMMANDES

```
echo "Début du script"
```

```
ls ~ | wc -l
```

```
echo "Fin du script"
```

## DONNER LES DROITS

```
chmod u+x mon_script.sh
```

Via un chemin relatif

```
./mon_script.sh
```

Via un chemin absolu

```
/home/moi/mon_script.sh
```



# LA PROCHAINE FOIS

Structures de contrôle et Expressions rationnelles