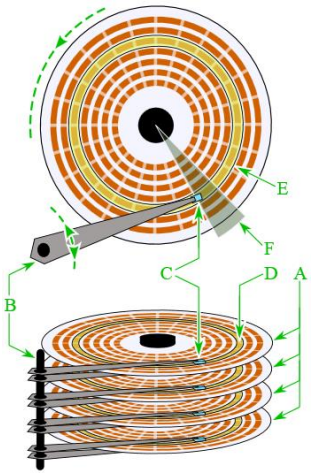
1. Les disques et les partitions.

1.1. Vision physique d'un disque, cylindres et secteurs

Un disque dur **mécanique** à supports magnétiques à rotation rapide - par opposition au disque **électronique** SSD (solid-state drive) permettant le stockage de données sur de la mémoire flash - est physiquement composé de **plateaux** ayant chacun deux faces. Sur chaque face, il y a un certain nombre de **pistes** concentriques. Chaque piste est composée d'un certain nombre de **secteurs**. Un **cylindre** correspond à l'ensemble des pistes que le bras de lecture/écriture peut lire sans se déplacer. Un secteur a habituellement une taille de 512 octets.



A : plateaux

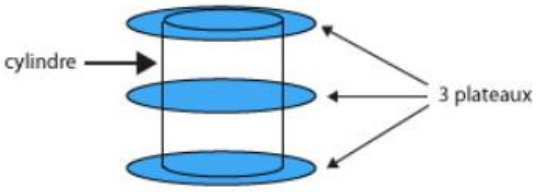
B : Bras

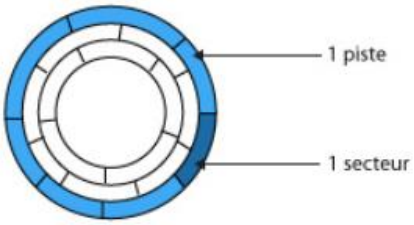
C : Tête

D : cylindre

E : piste

F : secteur





1.2. Le nom des disques

Les disques durs disposent de différentes interfaces permettant de les connecter à la carte mère :

- SATA (Serial Advanced Technology Attachment) ;
- IDE (Integrated Drive Electronics) ;
- SCSI (Small Computer System Interface) ;
- SAS (Serial Attached SCSI).

- Les disques reliés à des **contrôleurs IDE** sont nommés de la manière suivante :
  /dev/hda : le disque maître géré par le 1er contrôleur ;
  /dev/hdb : le disque esclave géré par le 1er contrôleur ;
  /dev/hdc : le disque maître géré par le 2ème contrôleur ;
  /dev/hdd : le disque esclave géré par le 2ème contrôleur ;
  Etc.
- Les disques **SCSI**, **SATA** et **USB** sont nommés de la manière suivante :
  /dev/sda : le 1er disque ;
  /dev/sdb : le 2ème disque ;
  Etc.

1.3. Partitionnement, table des partitions et le MBR

- Le partitionnement consiste en un **découpage logique** du disque : le disque physique, réel, est fractionné en plusieurs **disques virtuels**, logiques, les partitions.
- **Chaque partition** est vue comme un disque indépendant et contient son propre **système de fichiers** (**File system**).
- Les informations sur les partitions figurent sur le disque dans des zones nommées **tables de partitions**. La table des partitions **principales** est contenue dans le **premier secteur** du disque ou secteur d'amorçage (Master Boot Record ou **MBR**). D'une taille de **512 octets**, il contient, dans ses 446 premiers octets, un **code d'amorçage** (le bootstrap code) destinée à démarrer le **chargeur de démarrage** (le bootloader appelé **GRUB**). Les 64 octets suivants contiennent la **table des partitions principales**.

  Chaque ligne d'une table de partition contient l'adresse de début de la partition et sa taille.
- Dans la table des partitions **principales**, on peut créer **au plus quatre partitions** :
  - soit quatre partitions **primaires** ;
  - soit 3 partitions **principales** puis une partition **étendue**.

  Une **partition étendue** n'est ni plus ni moins qu'une **partition primaire spéciale** qui peut contenir des **partitions secondaires** appelées **partitions logiques**. Le nombre de partitions logiques est limité à 12 pour un disque SCSI et à 60 pour un disque IDE. Lorsque l'on veut plus de quatre partitions, il faut donc créer une partition étendue. Une partition étendue contient à son tour une table de partition ayant la même structure que la table principale.

1.4. Le nom des partitions

- Pour un disque, les partitions sont nommées de la manière suivante (exemple du disque hda) :

  /dev/hda1 : la 1ère partition primaire ;

  /dev/hda2 : la 2ème partition primaire ;

  /dev/hda3 : la 3ème partition primaire ;

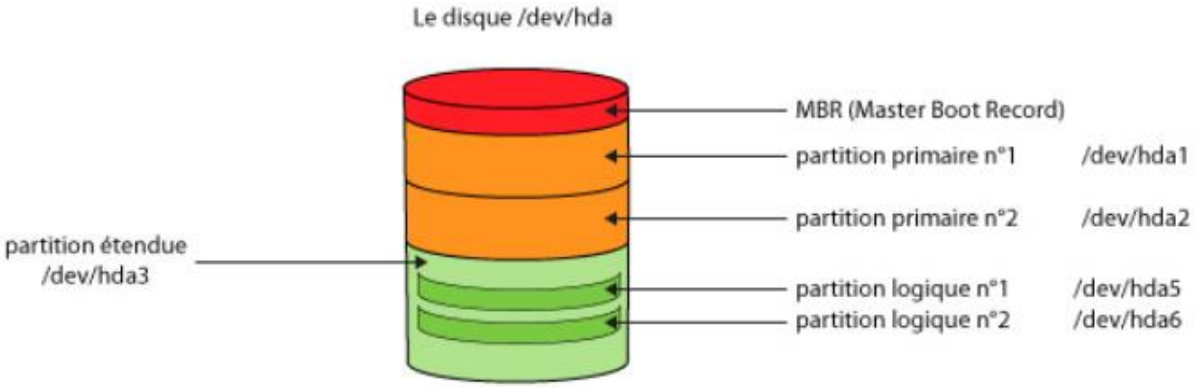
  /dev/hda4 : la 4ème partition primaire ;

  /dev/hda5 : la 1ère partition logique (à l'intérieur de la partition étendue).

  La numérotation des partitions logiques commence donc obligatoirement à 5.

  Toutes les partitions primaires ne sont pas forcément présentes.

  Une et une seule partition primaire peut jouer le rôle de partition étendue.



1.5. Les commandes

- La commande **sfdisk -s** : permet de lister les différents disques.

```
root@US1:~# sfdisk -s
/dev/sda:   8388608
/dev/sdb:   8388608
total: 16777216 blocks
root@US1:~#
```

- La commande **fdisk** : l'outil standard de partitionnement des disques.
  - La commande **fdisk -l** permet de listez l'ensemble des partitions :

```
sio1@UD2: ~
sio1@UD2:~$ sudo fdisk -l
[sudo] password for sio1:

Disque /dev/sda : 8589 Mo, 8589934592 octets
255 têtes, 63 secteurs/piste, 1044 cylindres, total 16777216 secteurs
Unités = secteurs de 1 * 512 = 512 octets
Taille de secteur (logique / physique) : 512 octets / 512 octets
taille d'E/S (minimale / optimale) : 512 octets / 512 octets
Identifiant de disque : 0x0007f8e1

Périphérique Amorce  Début        Fin      Blocs     Id  Système
/dev/sda1   *        2048    14680063     7339008   83  Linux
/dev/sda2        14682110    16775167     1046529    5  Étendue
/dev/sda5        14682112    16775167     1046528   82  partition d'échange Linu
x / Solaris
sio1@UD2:~$
```

Dans l'exemple ci-dessus, le plan de partition retenu pour le disque **/dev/sda** fait état de deux partitions : une partition **racine** pour le système et les données (**sda1**) d'environ 7,5 Go ainsi qu'une partition **d'échange** « **swap** » (**sda5**) d'environ 1 Go. La partition racine a été créée en tant que partition **primaire**. La partition swap est de type **logique**. La partition **étendue sda2** a été automatiquement créée à la création de la partition logique **sda5**.

- La commande **fdisk en mode interactif** permet de lister les partitions, de créer ou de supprimer une partition. Les manipulations effectuées avec fdisk **sans argument** ne sont prises en compte qu'à la fin du programme et non au fur et à mesure. Vous pouvez quitter sans sauvegarder en appuyant sur **q** ou bien, à n'importe quel moment, à l'aide des touches **Ctrl+c**.

```
sio1@UD2: ~
sio1@UD2:~$ sudo fdisk /dev/sda
[sudo] password for sio1:

Commande (m pour l'aide): m
Commande d'action
   a   bascule le drapeau d'amorce
   b   éditer l'étiquette BSD du disque
   c   basculer le drapeau de compatibilité DOS
   d   supprimer la partition
   l   lister les types de partitions connues
   m   afficher ce menu
   n   ajouter une nouvelle partition
   o   créer une nouvelle table vide de partitions DOS
   p   afficher la table de partitions
   q   quitter sans enregistrer les changements
   s   créer une nouvelle étiquette vide pour disque de type Sun
   t   modifier l'identifiant de système de fichiers d'une partition
   u   modifier les unités d'affichage/saisie
   v   vérifier la table de partitions
   w   écrire la table sur le disque et quitter
   x   fonctions avancées (pour experts seulement)

Commande (m pour l'aide):
```

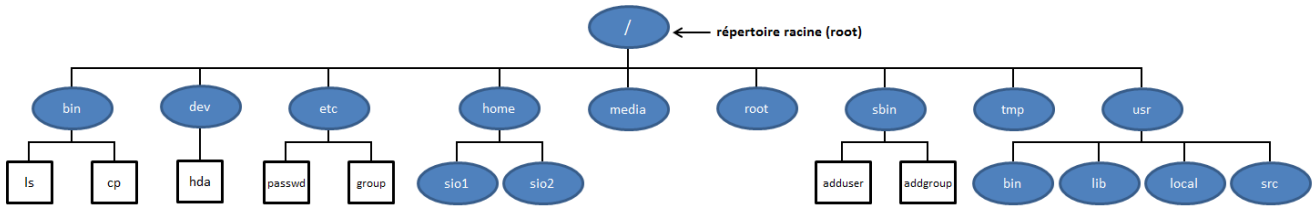
2. Les systèmes de fichier Linux.

Les fichiers **sont organisés sur une partition d'un disque**, une clé ou sur tout autre support de mémoire secondaire **selon ce que l'on appelle un système de fichiers** appelé communément **File System** ou **FS**. Un système de fichiers définit donc **comment sont gérés et organisés** les fichiers par le système d'exploitation sur un support de stockage.

- Les systèmes de fichiers **Microsoft** sont **FAT32** et **NTFS**. Ce dernier est apparu avec Windows NT. C'est le FS des OS Microsoft actuels. Il fragmente moins les fichiers que FAT32 et permet de définir, outre des autorisations de partage, des autorisations de sécurité sur les dossiers et fichiers. Contrairement à FAT32, c'est un système de fichiers **journalisé** qui permet en cas de crash de récupérer les données sans trop de problème.
- Le système de fichiers **Linux** le plus récent est **ext4**. C'est un système **journalisé** tout comme son prédécesseur ext3. Il y a très peu de fragmentation avec un FS Linux contrairement à un FS Microsoft : il n'y a pas besoin de défragmenter sous Linux.

2.1. L'arborescence des fichiers

- Le système de fichiers de Linux est **hiérarchique**. Il décrit une arborescence de répertoires et de sous-répertoires, en partant d'un élément de base appelé la **racine** ou **root directory** :



L'arborescence de fichiers

Sous Windows, il y a **plusieurs racines**. **C:** est la racine d'un disque ou d'une partition. **D:** peut être la racine d'un second disque, d'une seconde partition ou du lecteur CD/DVD.
Sous Linux, il n'y a qu'**une seule racine** notée « / » soit un dossier de base qui contient tous les autres dossiers et fichiers. **Il n'y a pas en conséquence de lettre de lecteur**.

- **Les dossiers de la racine :**
  - **bin** : contient les fichiers binaires exécutables susceptibles d'être utilisés par tous les utilisateurs de la machine (principales commandes utilisées dans la console).
  - **boot** : fichiers permettant le démarrage de Linux.
  - **dev** : ce dossier contient des fichiers spéciaux qui représentent chacun un périphérique.
  - **etc** : fichiers de configuration du système et de diverses applications.
  - **home** : répertoires personnels des utilisateurs.
  - **lib** : dossier contenant les bibliothèques partagées utilisées par les programmes. C'est en fait là que l'on trouve l'équivalent des .dll de Windows.
  - **media** : répertoire de montage des périphériques temporaires (clé USB, disque externe, etc.). Lorsqu'un périphérique amovible est inséré dans l'ordinateur, Linux permet d'y accéder à partir d'un sous-dossier de média.
  - **mnt** : autre répertoire permettant des montages temporaires de FS.
  - **root** : c'est le dossier personnel de l'utilisateur « root ». Normalement, les dossiers personnels sont placés dans /home, mais celui de « root » fait exception.
  - **sbin** : contient les fichiers binaires exécutables dédiés à l'administrateur.
  - **tmp** : dossier temporaire utilisé par les programmes pour stocker des fichiers.

- **usr** : répertoire dans lequel vont s'installer la plupart des applications des utilisateurs.
- **var** : ce dossier contient des fichiers et dossiers à contenus variables comme des logs (traces écrites de ce qui s'est passé récemment sur l'ordinateur), les spools d'impression, répertoire www, fichiers de zone DNS...

- **Les chemins**

Un chemin indique l'emplacement d'un fichier dans l'arborescence. Il peut être :

- **absolu** : un chemin absolu part de la racine.

  Exemple : **/home/sio1/si1/chapitre1.txt**

- **relatif** : un chemin relatif indique l'emplacement d'un fichier à partir du répertoire courant. Dans ce type de chemin, « **.** » et « **..** » désignent respectivement le **répertoire courant** et le **répertoire père** (le répertoire hiérarchiquement supérieur).

  Exemple :

  ./chapitre1.txt si le répertoire courant est /home/sio1/si1/

  si1/chapitre1.txt si l'on est dans le répertoire /home/sio1/

  ../chapitre1.txt si l'on est dans le répertoire /home/sio1/si1/tp/

  ../si1/chapitre1.txt si l'on est dans le répertoire /home/sio1/si2

Le shell interprète le **caractère tilde ~** comme un alias du répertoire personnel. Les chemins peuvent être relatifs au tilde. Ainsi, si vous êtes connecté en tant que sio1, le chemin peut s'écrire **~/si1/chapitre1.txt**

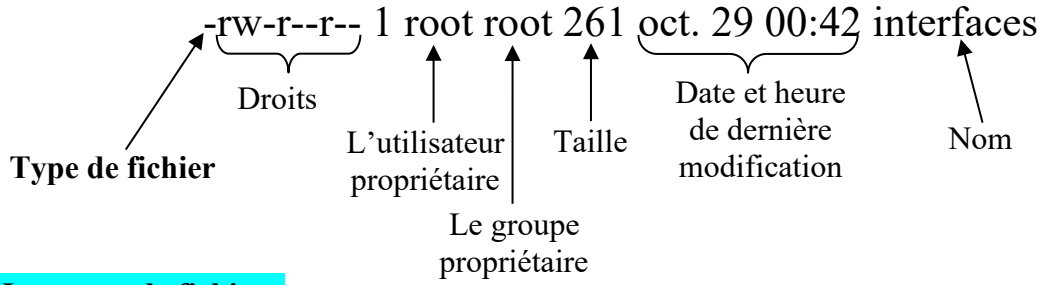
2.2. « Tout est un fichier »

Linux est un système d'exploitation entièrement orienté **fichier**. Tout est représenté par un fichier, tant les **données** que les **périphériques**.

- **La commande ls :**

La commande **ls** permet de lister le contenu d'un répertoire. Le paramètre **-l** permet d'afficher les principaux **attributs** des fichiers :

```
sio1@UD2: /etc/network
sio1@UD2:~$ cd /etc/network
sio1@UD2:/etc/network$ ls -l
total 20
drwxr-xr-x 2 root root 4096 oct.  23 11:01 if-down.d
drwxr-xr-x 2 root root 4096 févr. 13  2013 if-post-down.d
drwxr-xr-x 2 root root 4096 févr. 13  2013 if-pre-up.d
drwxr-xr-x 2 root root 4096 oct.  23 11:01 if-up.d
-rw-r--r-- 1 root root  261 oct.  29 00:42 interfaces
lrwxrwxrwx 1 root root   12 sept.  8 12:34 run -> /run/network
sio1@UD2:/etc/network$
```



- **Les types de fichier :**

- Les fichiers **ordinaires (-)** sont des fichiers tout à fait classiques qui contiennent des données. Par « données », comprenez n'importe quel contenu, y compris du binaire (un exécutable par exemple).

  Par défaut, rien ne permet de différencier les uns des autres, sauf à utiliser la commande **file** :

```
root@US1:~# which fdisk
/sbin/fdisk
root@US1:~# file /sbin/fdisk
/sbin/fdisk: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically li
nked (uses shared libs), for GNU/Linux 2.6.24, BuildID[sha1]=0xbc9106905e003a0de
109075195ddb04561cefcd8, stripped
root@US1:~# file /etc/network/interfaces
/etc/network/interfaces: ASCII English text
root@US1:~#
```

  La **notion d'extension de fichier** comme composante interne de la structure du système de fichiers **est inconnue de Linux**. Une extension n'a aucun rôle au niveau du système de fichiers. Elle sert simplement à distinguer visuellement et rapidement l'éventuel contenu d'un fichier par rapport à un autre.

- Les fichiers **catalogues** sont les répertoires (**d** pour **directory**). On a vu au paragraphe 2.1 que les répertoires permettent d'organiser le système de fichiers en créant une hiérarchie. **Un répertoire ou dossier n'est rien d'autre qu'un fichier particulier** contenant la liste des fichiers eux-mêmes présents dans ce répertoire.

- Le troisième type de fichiers est le fichier **spécial** (**c**, **b**). Les fichiers de ce type se trouvent principalement dans **/dev** et représentent des périphériques.

  Par exemple, un disque dur **sda** est un fichier pour Linux.

  Commande **ls -l | less** :

```
brw-rw---- 1 root disk      8,   0 nov.  10 10:29 sda
brw-rw---- 1 root disk      8,   1 nov.  10 10:29 sda1
brw-rw---- 1 root disk      8,   2 nov.  10 10:29 sda2
brw-rw---- 1 root disk      8,   5 nov.  10 10:29 sda5
brw-rw---- 1 root disk      8,   6 nov.  10 10:29 sda6
brw-rw---- 1 root disk      8,   7 nov.  10 10:29 sda7
brw-rw---- 1 root disk      8,  16 nov.  10 10:29 sdb
brw-rw---- 1 root disk      8,  17 nov.  10 10:29 sdb1
crw-rw---- 1 root disk     21,   0 nov.  10 10:29 sg0
crw-rw---- 1 root disk     21,   1 nov.  10 10:29 sg1
crw-rw---- 1 root cdrom    21,   2 nov.  10 10:29 sg2
lrwxrwxrwx 1 root root          8 nov.  10 10:29 shm -> /run/shm
crw------- 1 root root     10, 231 nov.  10 10:29 snapshot
drwxr-xr-x 3 root root         180 nov.  10 10:29 snd
brw-rw---- 1 root cdrom    11,   0 nov.  10 10:29 sr0
```

- Un autre type de fichiers est le **lien symbolique** (**l** pour **link**). Un fichier lien symbolique permet d'accéder à un autre fichier. Le nom du fichier lié apparaît derrière une flèche dans la commande **ls**.

2.3. FS : tables systèmes et inodes

Physiquement, un système de fichiers est composé de différentes **tables système** dont la **table des inodes** qui contient la table de **description et d'allocation mémoire des fichiers** ainsi que la **table des répertoires** qui est une table de correspondance entre les noms des fichiers et les numéros d'inodes.

- **La table des inodes**

Un fichier est décrit par des **propriétés** appelées les **métadonnées**. Sous Linux, il s'agit de l'**inode** qui définit les **caractéristiques** (sauf le nom du fichier) ainsi que **l'emplacement d'un fichier**. Le **contenu**, c'est-à-dire les **données**, est placé dans d'autres blocs du support de stockage. Dans un inode, on y trouve les **informations** suivantes :

- Le numéro d'inode ;
- les droits (autorisations d'accès) ;
- les dernières dates d'accès et de modification ;
- l'utilisateur propriétaire (UID) et le groupe propriétaire (GID) ;
- la taille ;
- le nombre de blocs utilisés ;
- le type de fichiers (fichier standard, répertoire, lien symbolique…) ;
- Les adresses des blocs de données.

La commande **ls -li** fournit une partie de ces informations :

```
sio1@UD2: ~
sio1@UD2:~$ ls -li
total 104
197094 drwxr-xr-x 2 sio1 sio1  4096 oct.  30 09:48 Bureau
197220 drwxr-xr-x 2 sio1 sio1  4096 oct.  30 08:57 Documents
143120 -rw-r--r-- 1 sio1 sio1  8445 oct.  29 10:15 examples.desktop
136067 -rw-rw-r-- 1 sio1 sio1    27 nov.  17 10:39 fichier1.txt
```

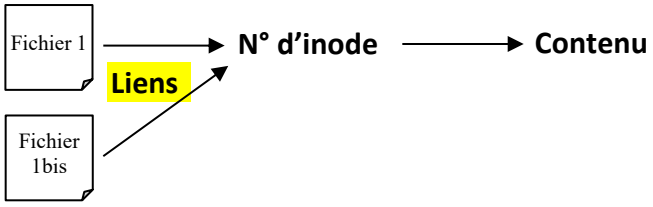
Le paramètre **-i** permet d'afficher le numéro d'inode qui est associé au fichier.

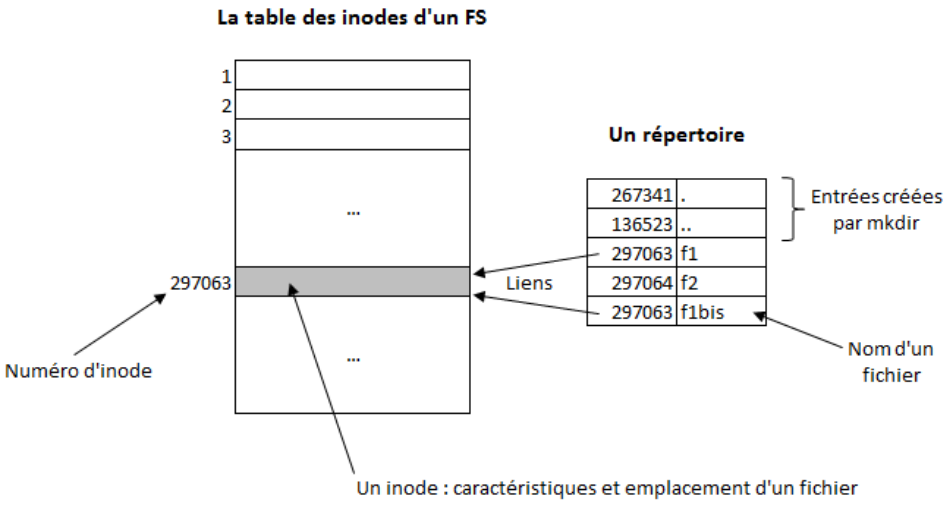
Vous pouvez obtenir également quelques informations sur un inode avec la commande **stat** :

```
sio1@UD2: ~
sio1@UD2:~$ stat index.html
  Fichier : «index.html»
   Taille : 22          Blocs : 8          ES blocs : 4096   fichier
Device : 801h/2049d     Inode : 197711      Liens : 1
Accès : (0664/-rw-rw-r--)  UID : ( 1002/    sio1)   GID : ( 1003/    sio1)
Accès : 2013-10-30 21:01:08.830041732 +0100
Modi. : 2013-10-30 20:59:43.738038417 +0100
Chgt  : 2013-10-30 20:59:43.738038417 +0100
Créé  : -
```

• La table des répertoires

Les noms des fichiers ne sont pas placés dans les métadonnées mais dans une table de correspondance {**nom de fichier - numéro d'inode**} car seul le numéro d'inode est associé au contenu d'un fichier. C'est la raison pour laquelle il est possible de donner plusieurs noms à un même fichier avec les **liens « physiques »**.





En résumé, chaque fichier est décomposé en trois parties stockées à des endroits différents :

- son ou ses noms (table des répertoires) ;
- ses propriétés (table des inodes d'un FS) ;
- son contenu.

2.4. La commande ln : créer des liens entre fichiers

- **Les liens physiques :**

Ils permettent d'avoir plusieurs noms de fichiers qui partagent exactement le même contenu et qui ont par conséquent le même numéro d'inode.
Ce type de lien correspond à l'**exemple présenté ci-dessus**. Que l'on modifie **f1** ou **f1bis**, on modifie le même contenu.

- Création d'un répertoire et de deux fichiers :

```
sio1@UD2: ~/liens
sio1@UD2:~$ mkdir liens
sio1@UD2:~$ cd liens
sio1@UD2:~/liens$ cal > f1
sio1@UD2:~/liens$ date > f2
sio1@UD2:~/liens$ ls -l
total 8
-rw-rw-r-- 1 sio1 sio1 188 nov.  17 14:18 f1
-rw-rw-r-- 1 sio1 sio1  47 nov.  17 14:18 f2
sio1@UD2:~/liens$
```

- Création d'un lien sur le fichier **f1** appelé **f1bis** :

```
sio1@UD2: ~/liens
sio1@UD2:~/liens$ ln f1 f1bis
sio1@UD2:~/liens$ ls -l
total 12
-rw-rw-r-- 2 sio1 sio1 188 nov.  17 14:18 f1
-rw-rw-r-- 2 sio1 sio1 188 nov.  17 14:18 f1bis
-rw-rw-r-- 1 sio1 sio1  47 nov.  17 14:18 f2
sio1@UD2:~/liens$
```

Les noms **f1** et **f1bis** se réfèrent au même contenu. Le nombre de liens apparaît dans la seconde colonne de la liste après les droits : en l'espèce, deux fichiers partagent le même inode. En l'absence de lien, pour la plupart des fichiers, la seconde colonne indique « **1** ». Outre l'affichage des numéros d'inode (cf. ci-après), c'est le seul indice qui permet de savoir qu'un **lien physique** a été créé sans savoir d'ailleurs lequel.

- Affichage des numéros d'inode de l'ensemble des fichiers du répertoire avec la commande **ls -lia** :

```
sio1@UD2: ~/liens
sio1@UD2:~/liens$ ls -lia
total 20
267341 drwxrwxr-x 2 sio1 sio1 4096 nov.  17 14:22 .
136523 drwxr-xr-x 27 sio1 sio1 4096 nov.  17 14:18 ..
297063 -rw-rw-r-- 2 sio1 sio1  188 nov.  17 14:18 f1
297063 -rw-rw-r-- 2 sio1 sio1  188 nov.  17 14:18 f1bis
297064 -rw-rw-r--  1 sio1 sio1   47 nov.  17 14:18 f2
sio1@UD2:~/liens$
```

Le paramètre **-i** permet d'afficher les numéros d'inode. On constate que les fichiers **f1** et **f1bis** ont bien le même numéro d'inode. Le paramètre **-a** permet d'afficher les fichiers et dossiers cachés. Un répertoire a au moins 2 liens : son père le référence par son nom (répertoire « liens » et il s'autoréférence sous le nom « **.** »).

- Si on supprime l'un des deux fichiers, l'autre fichier reste en place et le contenu est toujours présent sur le disque. L'inode ne sera supprimé que quand elle n'aura plus de lien c'est-à-dire quand plus aucun nom de fichier ne pointera dessus.

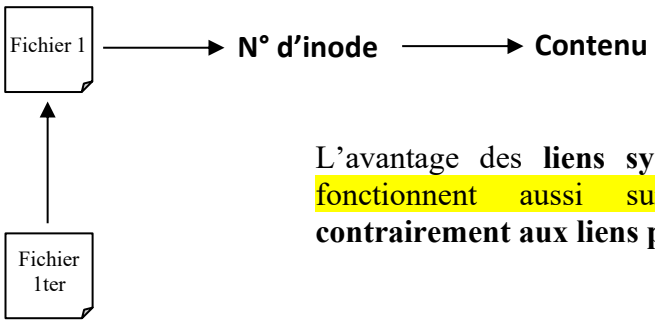
```
sio1@UD2: ~/liens
sio1@UD2:~/liens$ rm f1
sio1@UD2:~/liens$ cat f1bis
   Novembre 2013
di lu ma me je ve sa
                1  2
 3  4  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30

sio1@UD2:~/liens$ ls -li
total 8
297063 -rw-rw-r-- 1 sio1 sio1 188 nov.  17 14:18 f1bis
297064 -rw-rw-r-- 1 sio1 sio1  47 nov.  17 14:18 f2
sio1@UD2:~/liens$
```

- On renomme le lien **f1bis** en **f1** :

```
sio1@UD2: ~/liens
sio1@UD2:~/liens$ mv f1bis f1
sio1@UD2:~/liens$ ls -li
total 8
297063 -rw-rw-r-- 1 sio1 sio1 188 nov.  17 14:18 f1
297064 -rw-rw-r-- 1 sio1 sio1  47 nov.  17 14:18 f2
sio1@UD2:~/liens$
```

- **Les liens symboliques :**

Les liens symboliques sont créés sous Linux pour faire un **raccourci**. Contrairement au lien physique, un lien symbolique pointe vers un autre nom de fichier et non vers l'inode directement.



L'avantage des **liens symboliques** est qu'ils fonctionnent aussi sur des **répertoires contrairement aux liens physiques**.

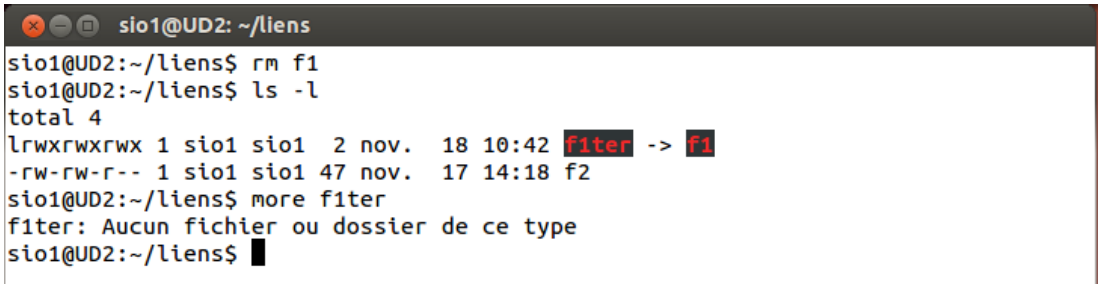
- Création d'un lien symbolique sur le fichier **f1** appelé **f1ter** : on utilise là encore la commande **ln** mais avec le paramètre **-s** (s pour symbolique) :

```
sio1@UD2: ~/liens
sio1@UD2:~/liens$ ln -s f1 f1ter
sio1@UD2:~/liens$ ls -li
total 8
297063 -rw-rw-r-- 1 sio1 sio1 188 nov.  17 14:18 f1
295641 lrwxrwxrwx 1 sio1 sio1   2 nov.  18 10:42 f1ter -> f1
297064 -rw-rw-r-- 1 sio1 sio1  47 nov.  17 14:18 f2
sio1@UD2:~/liens$
```

On constate 3 choses :

Un lien symbolique utilise un inode à part entière de type lien.
Le premier caractère de la seconde colonne (juste avant les droits) est un **l** (**link**).
A la dernière colonne, une flèche montre clairement que **f1ter** pointe vers **f1**.

- Si on supprime **f1ter**, il ne se passe rien de problématique. Par contre, si on détruit le fichier lié, à savoir **f1**, le lien **f1ter** existe toujours mais il pointe vers un fichier qui n'existe plus. Le lien symbolique est alors considéré comme un « **lien mort** » :



2.5. Les différents types de FS Linux

Les systèmes de fichiers de Linux sont de la famille des **ext** (Extended Filesystem*)*. Certains systèmes de fichiers ont un journal, d'autres pas. Les systèmes de fichiers tenant à jour un journal y enregistrent tous les changements intervenus avant de les effectuer réellement ce qui permet en cas d'arrêt brutal de récupérer les données.

- **ext2fs**

  Le « second extended filesystem », ext2, est considéré comme le système de fichiers historique de Linux. Ext2 n'est pas journalisé.
  Bien que disposant d'un successeur (ext3), il est toujours utilisé, voire conseillé dans certains cas. Il est rapide et nécessite moins d'écritures que les autres, donc il occasionne moins d'usure des supports de stockage, notamment les disques SSD, les clés USB ou les cartes mémoire. Ces supports peuvent parfois ne supporter qu'un nombre restreint de cycles de lecture/écriture...

- **ext3fs**

  Le « *third extended filesystem* », ext3, est le successeur d'ext2 depuis 1999. Il est journalisé. Il est entièrement compatible avec ext2. Il est possible d'utiliser un système de fichiers ext3 comme étant ext2, avec les mêmes commandes et les mêmes manipulations. Il est possible de transformer en quelques secondes un système ext2 en ext3, et vice versa. C'est l'un des systèmes de fichiers de choix pour Linux, et il a été le plus utilisé pendant plus de dix ans pour sa souplesse.

- **ext4fs**

  Le « *fourth extended filesystem* », ext4, est le successeur d'ext3. Il est stable depuis fin 2008 et il est le système de fichiers par défaut de nombreuses distributions depuis 2010. Il est compatible avec ext3 tant que certaines options ne sont pas activées. Il est journalisé. Il apporte de nouvelles fonctionnalités, notamment la notion d'allocation par extent qui effectue une pré-allocation de blocs dans des zones contiguës afin de limiter la fragmentation des fichiers et du disque, et donc d'améliorer les performances.

2.6. Le montage des systèmes de fichiers

L'arborescence globale des fichiers est composée d'un ou de **plusieurs systèmes de fichiers**. Typiquement, un FS est stocké dans une **partition** d'un disque dur mais également dans un **cdrom** ou une **clef USB**.

Les **fichiers d'un FS ne sont accessibles** que si le FS est **activé** c'est-à-dire « **monté** ». Le montage d'un FS consiste à **associer la racine du FS à un répertoire dit répertoire de montage**. Le **FS root** est monté automatiquement par le noyau au démarrage. Les **autres FS** peuvent être montés ultérieurement de manière **automatique** ou **manuelle**.

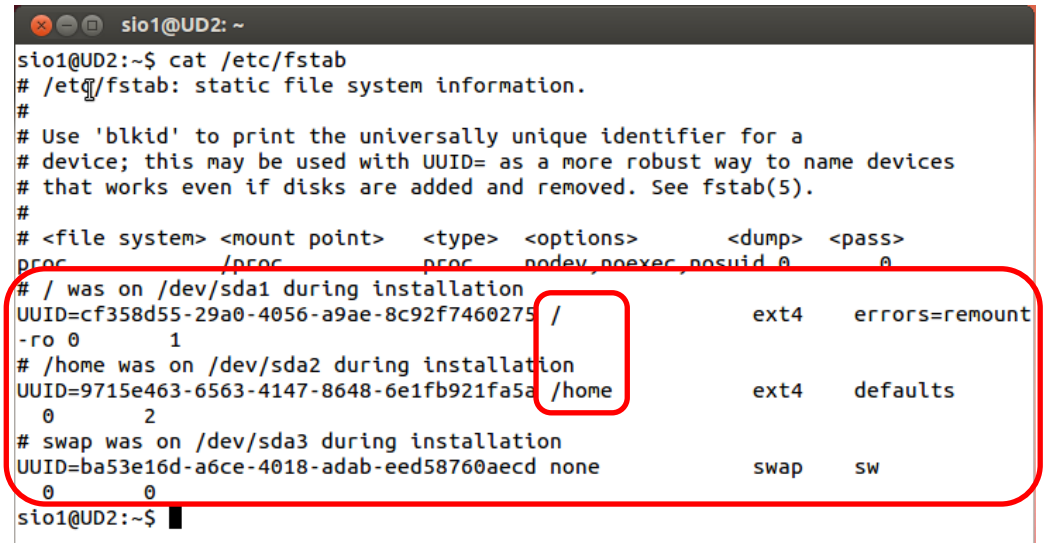
Les commandes de gestion de FS

- **mkfs** : crée un FS. Concrètement, la commande **mkfs** formate une partition c'est-à-dire écrit les tables systèmes.
- **mount** : monte un FS. L'option **-a** (all) monte l'ensemble des FS décrits dans le fichier **/etc/fstab**. Cette commande est activée par les scripts lancés automatiquement au démarrage.
- **umount** : démonte un FS.
- **df** : liste les FS montés et mentionne la place libre par FS.

Le fichier /etc/fstab

Chaque ligne du fichier **fstab** est composée de différents champs :

- la partition qui abrite le FS (son nom, son UUID) ;
- le répertoire de montage ;
- son type ;
- les options de montage : « defaults » indiquant un montage automatique au démarrage en lecture/écriture, « noauto » ou « ro » indiquant un montage manuel en lecture seule ;
- les deux derniers champs n'ont pas de rapport avec le montage.



3. Partitionner les disques.

3.1. Partitions indispensables

Source : http://doc.ubuntu-fr.org

Par défaut, Ubuntu a besoin d'au moins deux partitions : un espace d'échange (**swap**) et une partition **racine** (/). Dans certains cas des partitions de démarrage peuvent aussi se révéler indispensables.

- **Partition racine**
  - *Point de montage :* /
  - *Utilité :* la partition racine est la base de l'arborescence du système Ubuntu. Par défaut, c'est dans celle-ci que tous les fichiers vont être placés : fichiers de configuration, programmes, documents personnels, etc.
  - *Taille :* minimum 8 Go, cependant, au moins 15 Go conseillé. Attention, si cette partition est pleine, Ubuntu ne pourra plus démarrer.
- **Partition swap**
  - *Point de montage :* **aucun**

- *Utilité :* l'espace d'échange est une extension de la mémoire vive (RAM) sur le disque dur. Afin d'éviter un blocage de l'ordinateur lorsque sa RAM est pleine, Linux se sert de cette partition pour décharger temporairement la RAM. Son utilisation à cet effet est plutôt rare dans les ordinateurs modernes, disposant d'au moins 1 Go de RAM. Cependant, elle sert aussi de décharge de la RAM lors de la mise en hibernation, c'est pour cette raison que la taille de la partition swap doit être d'au moins la taille de la RAM.
- *Taille :* jusqu'à 1 Go de RAM, taille de la partition swap de 2× la RAM ; au-delà, swap de 1 à 1,5 × la RAM.

- **Partition boot (parfois nécessaire)**
  - *Point de montage :* **/boot**
  - *Utilité :* Certains ordinateurs n'arrivent pas à lire les fichiers de démarrage s'ils sont situés trop loin (>100Go) du début du disque. Dans ce cas, il est généralement nécessaire de créer une partition /boot en début de disque. Mais cette partition /boot séparée n'est généralement pas utile sur un ordinateur récent et elle peut même générer des problèmes.
  - *Taille :* 250Mo-1Go

3.2. Partitions optionnelles

- **Partition de données personnelles**
  - *Point de montage :* **/home**
  - *Utilité :* une partition home permet d'isoler les paramètres personnels et les dossiers personnels des utilisateurs du reste du système. Il est effectivement conseillé de séparer les données des traitements (système et applications). En cas de réinstallation du système, les données des utilisateurs sont conservées.
  - *Taille :* Selon le besoin.

- **Partition pour les fichiers temporaires**
  - *Point de montage :* **/tmp**
  - *Utilité :* ce dossier contient les fichiers qui ne sont nécessaires que temporairement. Ce dossier concerne certains programmes comme les installateurs qui ont parfois besoin de beaucoup de place pour stocker des fichiers temporairement. Il peut comprendre également des pages web au moment de leur visionnement, des documents en cours de modifications, des fichiers en cours de téléchargement.

- **Partition var**
  - *Point de montage :* **/var**
  - *Utilité :* le dossier /var contient toutes les données variables (les journaux systèmes, les nombreuses données de différents services, les messages électroniques, les sites web, le cache du système des paquets).