RÉPERTOIRE

Sortie de la commande ls -l :

-rw-r--r--	1	oracle	dba	466	Feb 8 2001	input.log
1	2	3	4	5	6	7

1. Le premier caractère représente le type de fichier, les autres, par blocs de trois, les droits pour l'utilisateur, le groupe et tous (expliqué plus loin).
2. Compteur de liens (expliqué plus loin)
3. Propriétaire du fichier.
4. Groupe auquel appartient :le fichier
5. Taille du fichier en octets
6. Date de dernière modification (parfois avec l'heure)
7. Nom du fichier

Répertoire

Deux autres commandes utiles :

- **cat:** concaténation de fichiers, le résultat étant affiché par défaut sur la sortie standard(écran).

- **touch:** permet de créer un fichier s'il n'existe pas et s'il existe de modifier sa date d'accès et sa date de modification
- **touch toto:** crée le fichier toto s'il n'existe pas.

Gestion des fichiers et répertoires

Création de répertoires

- La commande **mkdir** **(make directory)** permet de créer un ou plusieurs répertoires, ou une arborescence complète.

mkdir rep1 [rep2] … [repn]

$ mkdir documents

$ mkdir documents/texte documents/calcul documents/images

- La commande mkdir accepte un paramètre « **-p** » permettant de créer une arborescence.

Gestion des fichiers et répertoires

- Dans l'exemple précédent, si je veux créer documents/texte et que documents n'existe pas, alors :
- **$ mkdir -p documents/texte**: va créer à la fois documents et texte.
- C'est valable pour tous les répertoires de niveau supérieur :
- **$ mkdir -p documents/texte/perso**: va créer les répertoires documents, texte et perso s'ils n'existent pas. S'il existent ils ne sont pas modifiés.

Gestion des fichiers et répertoires

Suppression de répertoires

- La commande **rmdir** **(remove directory)** supprime un ou plusieurs répertoires.
- Elle ne supprime pas une arborescence.
- Si des fichiers sont encore présents dans le répertoire, la commande retourne une erreur.
- Le répertoire ne doit donc contenir ni fichiers ni répertoires.

rmdir rep1 [rep2] ... [repn]

```
$ cd documents
$ rmdir texte/perso
```

Gestion des fichiers et répertoires

Copie de fichiers

- La commande **cp** **(copy)** copie un ou plusieurs fichiers vers un autre fichier ou vers un répertoire.

  cp fic1 fic2

  cp fic1 [fic2 ... ficn] rep1

- Dans le premier cas, fic1 est recopié en fic2.
- Si fic2 existe, il est écrasé sans avertissement (sauf droit particulier).
- Dans le second cas, fic1, fic2 et ainsi de suite sont recopiés dans le répertoire rep1.
- Les chemins peuvent êtres absolus ou relatifs.

Gestion des fichiers et répertoires

- La commande peut prendre les options suivantes :
  - **-i** : demande une confirmation pour chaque fichier avant d'écraser un fichier existant.
  - **-p** : préservation des permissions, dates d'accès de modification
  - **-r** : Récursif. Si la source est un répertoire copie de ce répertoire et de toute son arborescence.

  - Les liens symboliques ne sont pas recopiés tels quels, mais seulement les fichiers pointés (avec le nom du lien cependant).

Gestion des fichiers et répertoires

- **-R** : comme -r mais la recopie est identique à l'original (les liens symboliques sont copiés tels quels)

Gestion des fichiers et répertoires

Déplacer et renommer un fichier

- La commande **mv** (**move**) permet de déplacer et/ou de renommer un fichier.
- Elle a la même syntaxe que la commande **cp**.
- On peut à la fois déplacer et changer de nom.

```
$ cd
$ mv cv.txt cv_toto.txt
$ mv image.jpg documents/images/photo_toto_cv.jpg
```

Gestion des fichiers et répertoires

Supprimer un fichier ou une arborescence

- La commande **rm** **(remove)** supprime un ou plusieurs fichiers, et éventuellement une arborescence complète, suivant les options.
- La suppression est définitive (à moins d'avoir un utilitaire système propre au filesystem).
    - **rm [Options] fic1 [fic2...]**

Options :

- **-i** : la commande demandera une confirmation pour chacun des fichiers à supprimer.

Suivant la version d'Unix, le message change et la réponse aussi : y, Y, O, o, N, n, parfois toutes.

Gestion des fichiers et répertoires

- **-r** : le paramètre suivant attendu est un répertoire. Dans ce cas, la suppression est récursive : tous les niveaux inférieurs sont supprimés, les répertoires comme les fichiers.
- **-f** : force la suppression. Si vous n'êtes pas le propriétaire du fichier à supprimer**, rm** demande une confirmation, mais pas avec l'option **-f**. Aucun message n'apparaîtra si la suppression n'a pu avoir lieu.

```
$ cd
$ rm -rf documents
```

Gestion des fichiers et répertoires

Les liens : plusieurs noms pour un fichier

- Un lien permet de donner plusieurs noms à un même fichier, ou de faire pointer un fichier sur un autre.
- Plutôt que de faire plusieurs copies d'un même fichier pour plusieurs utilisateurs, on peut par exemple permettre à ceux-ci d'accéder à une copie unique, mais depuis des endroits et des noms différents.
- Un lien est un couple (**nom_de_fichier, numéro d'inode**).
- On utilise la commande **ln.**

ln [options] fic1 fic2

ln [options] fic1 rep1

ln [options] rep1 fic2

Gestion des fichiers et répertoires

- Il existe deux types de liens : les liens en dur **«hard links»** et les liens symboliques **«symbolic links».**

- Liens Physiques (hard link)
- Création d'un nouveau nom pour le même inode.
- Impossible de faire un lien physique vers un répertoire: le système de fichiers doit être arborescent.(seules les feuilles peuvent être partagés)
- Impossible de faire un lien physique vers un autre volume (partition)

Gestion des fichiers et répertoires

- Lorsqu'un fichier possède deux liens physiques, la suppression de l'un ou l'autre de ces liens n'entraine pas la suppression du fichier.
- Plus exactement, tant qu'il subsiste au minimum un lien physique, le fichier n'est pas effacé.
- En contrepartie lorsque l'ensemble des liens physiques d'un même fichier est supprimé le fichier l'est aussi.
- Il faut noter toutefois qu'il n'est possible de créer des liens physiques qu'au sein d'un seul et même système de fichiers.

ln nom-du-fichier-reel nom-du-lien-physique

Gestion des fichiers et répertoires

- Liens Symboliques (soft link)
- C'est des fichiers spécial contenant le chemin d'accès à un autre fichier (correspond aux « raccourcis » de Windows).
- Création d'un nouvel inode de type lien symbolique (possède sa propre inode)
- Le bloc de données contient une chaine de caractères représentant le chemin (relatif ou absolu) vers un fichier.
- Possibilité de lien symbolique vers un répertoire.
- Possibilité de lien symbolique vers un autre volume (partition).

ln -s nom-du-fichier-reel nom-du-lien-symbolique

Les Inodes

- Sous un système UNIX, un fichier quel que soit son type est identifié par un numéro appelé numéro d'inode, qu'on pourrait traduire en français par "i-noeud".
- Ainsi derrière la façade du shell, un répertoire n'est qu'un fichier, identifié aussi par un inode, contenant une liste d'inode représentant chacun un fichier.
- La différence entre un lien hard et symbolique se trouve au niveau de l'inode.
- Un lien hard n'a pas d'inode propre, il a l'inode du fichier vers lequel il pointe.
- Par contre un lien symbolique possède sa propre inode.
- A noter que vous ne pouvez pas créer de liens hards entre deux partitions de disque différente, vous n'avez pas cette contrainte avec les liens symboliques.

Critères de recherche sur noms de fichier

- Lors de l'utilisation de commandes en rapport avec le système de fichier, il peut devenir intéressant de filtrer la sortie de noms de fichiers à l'aide de certains critères, par exemple avec la commande ls.
- Au lieu d'afficher toute la liste des fichiers, on peut filtrer l'affichage à l'aide de divers critères et caractères spéciaux.

Critères de recherche sur noms de fichier

Caractère spécial	*Rôle*
*	Remplace une chaîne de longueur variable, même vide
?	Remplace un caractère unique quelconque
[]	Une série ou une plage de caractères
[!...]	Inversion de la recherche

Critères de recherche sur noms de fichier

- **ls a*** :

tous les fichiers commençant par a

- **ls a??** :

tous les fichiers de trois caractères commençant par a

- **ls a??*** :

tous les fichiers d'au moins trois caractères et commençant par a

- **ls [aA]*** :

tous les fichiers començant par a ou A

- **ls [a-m]?*txt** :

tous les fichiers commençant par les lettres de a à m, possédant au moins un second caractère avant la terminaison txt.

Critères de recherche sur noms de fichier

- C'est le shell qui est chargé d'effectuer la substitution de ces caractères avant le passage des paramètres à une commande.
- Ainsi lors d'un **cp * documents** cp ne reçoit pas le caractère * mais la liste de tous les fichiers et répertoires du répertoire actif.

Verrouillage de caractères

- Certains caractères spéciaux doivent être verrouillés, par exemple en cas de caractères peu courants dans un nom de fichier.
- Le backslash \ permet de verrouiller un caractère unique.

  ls paie\ *.xls va lister tous les fichiers contenant un espace après paie.
- Les guillemets "..." les guillemets permettent l'interprétation des caractères spéciaux, variables, au sein d'une chaîne.
- Les apostrophes '...'verrouillent tous les caractères spéciaux dans une chaîne ou un fichier.

Utilisateur, groupes

- On distingue sous Unix les utilisateurs et les groupes.
- Un groupe définit un ensemble d'utilisateurs,
- un utilisateur fait obligatoirement partie d'au moins un groupe, ou de plusieurs.
- Le groupe par défaut d'un utilisateur est « **users** ».

Les droits d'accès

- Nous avons indiqué que le rôle d'un système d'exploitation est aussi d'assurer la sécurité et l'accès aux données, ce qui se fait grâce au mécanisme des droits.
- Chaque fichier se voit attribué des droits qui lui sont propres, des autorisations d'accès individuelles.
- Lors d'un accès le système vérifie si celui-ci est permis.

- A sa création par l'administrateur, un utilisateur se voit affecté un **UID** **(«User Identifier»)** unique.
- Les utilisateurs sont définis dans le fichier **/etc/passwd**.
- De même chaque utilisateur est rattaché à au moins un groupe (groupe principal).

Les droits d'accès

- Chaque groupe possédant un identifiant unique, le **GID («Group Identifier»).**
- Les groupes sont définis dans **/etc/group**.
- La commande id permet d'obtenir ces informations.
- En interne, le système travaille uniquement avec les UID et GID, et pas avec les noms par eux-mêmes.
- A chaque fichier sont associés un UID et un GID définissant son propriétaire et son groupe d'appartenance.
- On peut affecter des droits pour le propriétaire, pour le groupe d'appartenance et pour le reste du monde.

Les droits d'accès

- On distingue trois cas:

  - **1) UID de l'utilisateur identique à l'UID défini pour le fichier :** cet utilisateur est propriétaire du fichier.
  - **2) Les UID sont différents :** le système vérifie si le GID de l'utilisateur est identique au GID du fichier : si oui l'utilisateur appartient au groupe associé au fichier.
  - **3) Dans les autres cas (aucune correspondance) :** il s'agit du reste du monde (others), ni les propriétaires, ni appartenant au groupe.