Défi Big Data

Travailler avec de vraies données

Xavier Serpaggi

mercredi 5 Avril 2017

Travail à rendre

Vous écrirez les réponses aux questions ci-dessous dans un fichier texte. Aucun autre format n'est attendu. Ce fichier texte sera déposé sur Campus, même partiel, à la fin de la séance.

Arrivée

Démarrez votre machine virtuelle Linux en prenant soin de vérifier que le mode réseau dans VirtualBox est bien positionné sur NAT.

Remarques :

— Dans ce document, les caractères `$>` représentent l'invite de commande. Il ne faut pas les retaper.

— Le caractère `\` à la fin d'une ligne signifie que la commande continue sur la ligne suivante.

Trucs & astuces

Quelques trucs et astuces supplémentaires pour l'utilisation du shell :

— Il est possible de connaître la liste des processus en cours de fonctionnement avec la commande `'ps'`, ou avec la commande `'top'` pour avoir un affichage dynamique.

— On peut envoyer des signaux aux processus en cours de fonctionnement avec la commande `'kill'`. Le signal envoyé par défaut (signal 15, ou SIGTERM) est une (courtoise) demande d'arrêt du processus.

— Votre ordinateur peut faire plusieurs choses en même temps. Dans un terminal vous éditez un fichier, dans un autre vous lisez une page de manuel et dans un troisième vous avez un programme qui réalise une tâche longue, ...

Étape 1 : interlude

■ *Choisir le bon outil au bon moment, c'est important. Pour cela, il est nécessaire d'avoir une panoplie d'outils à sa disposition (une trousse à outils ?). N'oubliez pas que si tout ce dont on dispose est un marteau, on a tendance à tout voir comme un clou !*

La commande `'comm'` permet de trouver les lignes en commun ou différentes entre deux fichiers [1]. Encore faut-il que ces fichiers soient triés !

Exercices

Pour trouver des mots en commun à plusieurs fichiers, triés ou pas, il est possible d'utiliser des filtres de Bloom.

Vous allez essayer de répondre à ce problème en utilisant les outils de la ligne de commande.

Vous trouverez les fichiers nécessaires à la réalisation de cet exercice sur Campus.

1. Trouvez les mots communs aux fichiers `texte-Shakespeare.txt` et `corncob_lowercase.txt` [2].
2. Refaites cet exercice en utilisant le fichier `large_Shakespeare.txt` à la place de `texte-Shakespeare.txt` (il faudra sans doute un peu travailler ce fichier au préalable).

1. Il existe également la commande `'join'` qui permet de répondre à ce problème.
2. Vu qu'ils ne sont ni triés ni correctement formatés, vous indiquerez et vous expliquerez les traitements préalables que vous avez fait subir à vos fichiers.

3. L'opération demandée (la recherche d'éléments communs entre deux ensembles) est une opération commune en l'algèbre relationnelle, c'est une jointure (une jointure interne plus précisément). Alors pourquoi ne pas utiliser une base de données pour réaliser cette opération ? Nous allons essayer ! Mais nous n'avons pas de serveur de base de données à notre disposition. Qu'à cela ne tienne, nous allons utiliser une base de données *légère* : `'sqlite3'`.
Cette partie prend beaucoup de temps, surtout sur les ordinateurs de l'école, vous disposez donc du fichier de base de donnée déjà créé : `communs.db`.
Voici néanmoins les étapes qui ont été réalisées pour créer ce fichier :
   (a) Création d'une base de données sqlite :
   ```
   $> sqlite3 communs.db \
   'create table shakespeare ( mot char(28) );
   create table corncob ( mot char(28) );'
   ```
   (b) Insertion des mots du fichier `texte_Shakespeare.txt` dans la table `shakespeare` et des mots du fichier `corncob_lowercase.txt` dans la table `corncob` (avec mesure du temps nécessaire) :
   ```
   $> time sed -e 's/\(.*\)/insert into shakespeare values ("\1");/' \
   texte_Shakespeare.txt | sqlite3 communs.db &&

   real    2m36,927s
   user    0m1,332s
   sys     0m7,324s
   $> time sed -e 's/\(.*\)/insert into corncob values ("\1");/' \
   corncob_lowercase.txt | sqlite3 communs.db

   real    6m41,917s
   user    0m2,940s
   sys     0m18,516s
   $>
   ```
   À vous de faire la requête pour trouver les mots communs depuis la base de données ! Vous mesurerez le temps nécessaire à son exécution avec la commande `'time'` et vous comparerez au temps nécessaire pour le faire avec la commande `'comm'`. Quelles conclusions en tirez-vous ?

Étape 2 : des données brutes vers quelque chose de plus exploitable

■ *Les BigData sont souvent des données issues de capteurs répartis dans l'environnement et qui remontent leurs informations vers un serveur de stockage. Ces données deviennent rapidement volumineuses et il faut faire le tri. Souvenons-nous que l'ordinateur est très fort pour faire le sale boulot, pour peu qu'on l'ait correctement programmé !*

Exercices

Vous avez découvert dans les séances précédentes les principaux outils nécessaires à la réalisation de travaux sur des données variées. Il se peut que vous imaginiez utiliser des outils supplémentaires (outils du shell bien entendu) ; n'hésitez pas, cette fois-ci ce sont les fonctionnalités attendues qui vous sont données. Vous devrez trouver un moyen de les réaliser.

Il n'y a pas de réponse unique à ce travail, cependant, une solution valide peut être mal présentée et/ou peu efficace. Décrivez ce que vous faites dans des commentaires, pensez « réutilisabilité ». Enfin, pensez à rester le plus concis possible : au moins il y a de commandes enchaînées, au plus elles seront efficaces et faciles à débugger.

Le fichier de données que vous allez utiliser est un *dump* d'une table de base de données dont le schéma est donné sur la figure 1.

1. Téléchargez le fichier de données (voir sur Campus) et vérifiez rapidement que sa structure correspond à celle de la table.
2. à partir de ce script obtenez les informations suivantes :
   (a) Lister les capteurs ayant émis des données ;
   (b) Séparer le fichier en autant de fichiers 'YYYYMMDD.dat' qu'il y a de jours d'émission de données ;
   (c) Extraire les informations de type `TYPE`, pour le capteur ayant la référence `ref_capteur` depuis le fichier de données vers le fichier `ref_capteur-TYPE.dat`. TYPE pourra prendre les valeurs `TEMP`, `HUM` ou `LUM`.

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
ref	varchar(100)	YES		NULL	
ts	datetime	YES		NULL	
server_ts	datetime	NO		NULL	
name	varchar(100)	NO		NULL	
value	float	YES		NULL	
firm_vers	varchar(6)	YES		NULL	

FIGURE 1 – schéma de la table relationnelle dans laquelle sont stockées les données issues des capteurs.

(d) Extraire tous les types d'informations pour le capteur `ref_capteur`. Un fichier `ref_capteur-TYPE.dat` sera créé pour chaque type d'information (c'est la même chose que la question précédente, mais avec une approche systématique et automatique).

(e) Extraire tous les types d'informations pour tous les capteurs. Un fichier `ref_capteur-TYPE.dat` sera créé pour chaque type d'information et pour chaque capteur (encore une fois, c'est une généralisation de l'étape précédente).

3. Extrayez les données pour la journée du 23 Mars 2017.