

**Rapport**- 1er Avril 2021 -

**Big Data and Hadoop :**

**Practical Project**

**NADAUD Edouard**

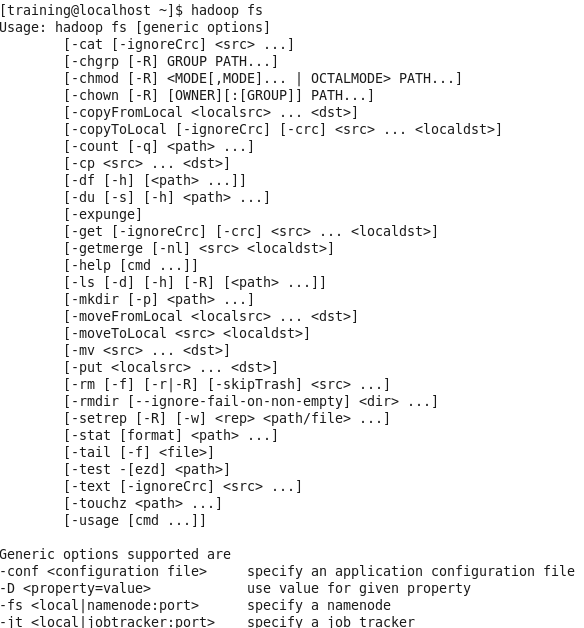
**NESPOULOUS Guilhem**

**NOBLANC Aymeric**

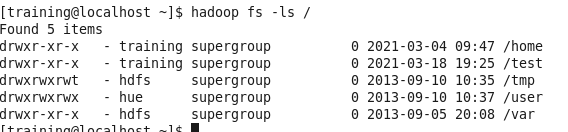
**2 Hadoop Basics :**

**2.1 Exploring HDFS**

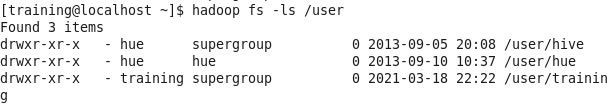
1)

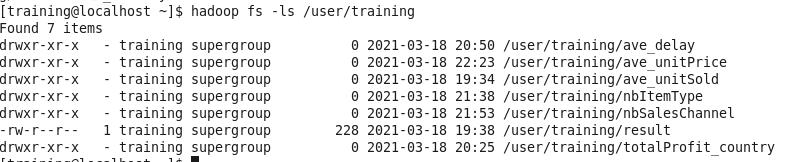


La commande ‘hadoop fs’ affiche toutes les commandes étant utilisable avec le FsShell. On reconnait toutes les commandes de base d’un système linux.

2)

3)



4)

(Nous allons présenter des commandes dans le reste de ce rapport. Surtout celles qui nous ont vraiment servis.)

*Résultat après avoir fait le projet pour avoir des données affichées*

**2.2 MapReduce :**

Pseudo-code pour l’exercice de l’anagramme :

**- map**

Input: key = LongWritable, value = Text

Output: key = Text, value = Text

\* split up the value text into words

\* split up each word into characters

\* for each word, sort all characters by alphabetical order (wordSort)

\* for each word, add an output key/value pair of <wordSort, word>

**- reduce**

Input: key = Text, value = Text (or Iterator<Text>, either way of writing this is fine for me)

Output: key = IntWritable, value = Text

\* calculate the sum for each of the values associated with the key

\* add an output key/value pair of <sum, allAnagramWords>, that is how many times the anagram occurred and all the anagram word

(A retrouver au format .txt dans le dossier envoyé)

**3 Sales analysis :**

Voici le fonctionnement que nous avons retenu pour mettre en place l’utilisation de MapReduce en python sur la VM fournie.

Pour commencer, nous créons un dossier (nommé test dans l’exemple ci-dessous) dans le Hadoop Distributed File System (HDFS). Il servira à contenir tous les documents, les programmes et les résultats de nos algorithmes.



Ensuite nous uploadons le fichier des ventes dans le dossier test. Cela permet de pouvoir y accéder lors de nos expérimentations avec hadoop.



Pour finir, nous créons 2 fichiers (Mapper.py et Reducer.py) dans le répertoire courant. Ils contiendront les programmes permettant de traiter les données. Pour pouvoir les utiliser faut leurs ajouter des droits d’exécution comme montré ci-dessous.





Après avoir créé un environnement de travail stable, nous nous attelons à trouver les algorithmes MapReduce que nous pouvons faire avec les données mises à notre disposition. Nous commençons à l’écrit sur un papier puis nous développons cela en python car c’est le langage principal que nous utilisons actuellement.

Tous les pseudo-code ainsi que leurs implémentations en python et leurs résultats sont à retrouver dans le dossier envoyé. Chaque statistique est regroupée dans un dossier spécifique.

Après leurs créations, nous commençons par tester nos programmes en local (à comprendre sans hadoop) grâce à la commande ci-dessous. Cela nous est très utile pour comprendre les problèmes et les résoudre plus facilement.



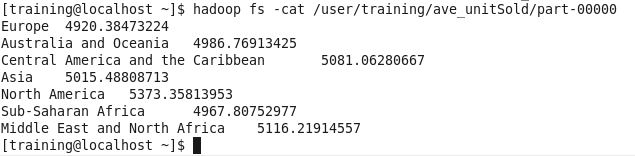
Lorsque nos algorithmes sont implémentés, nous lançons le traitement des données directement avec hadoop grâce à la commande ci-dessous. Ses arguments sont :

* mapper : comment appeler la fonction Map.
* reducer : comment appeler la fonction Reduce.
* Input : où est le fichier à traiter
* output : où mettre le fichier résultat (nom d’un dossier non existant)
* file : où trouver les fichiers utilisé dans les arguments mapper et reducer)



Ensuite nous affichons les résultats grâce à cette commande : (un exemple de résultat est disponible juste en dessous)





Pour récupérer ce fichier résultat nous utilisons cette commande. Elle copie le document part-00000 contenant le résultat dans le répertoire courant avec le nom result.txt. (C’est comme cela que l’on fournit les résultats de chaque statistique dans son dossier attitré)



Pour finir, nous transférons tous les fichiers intéressants de la VM à notre ordinateur avec la commande de copie de document via ssh ‘scp’.

