

TP n° 3

Exercice 0

Cet exercice consiste à configurer son compte pour pouvoir utiliser Hadoop.

1. Éditez le fichier `~/.ssh/config` de manière à ce qu'il contienne les lignes :

```
AddressFamily inet
Host 0.0.0.0 127.0.0.1 localhost ip6-localhost
    StrictHostKeyChecking no
    UserKnownHostsFile=/dev/null
```

2. Exécutez la commande suivante : `ssh localhost`. Si vous ne pouvez pas vous logger sans mot de passe, effectuez l'opération suivante :

```
ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa
cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

puis vérifier que la commande `ssh localhost` fonctionne sans problème. Cette commande vous connecte, via le réseau à la machine locale (un nouveau shell est démarré dans votre terminal). Quittez ce dernier en faisant `ctrl-D` ou `exit`.

3. Ajouter à la fin de votre fichier `~/.bashrc` la ligne :

```
source /public/kn/setup-kn.sh
```

et relancer le terminal

4. Exécutez la commande suivante : `mkdir -p ~/.hadoop_logs`
5. Vérifier si des fichiers temporaires hadoop existent :

```
ls -d /tmp/hadoop*
```

Si cette commande affiche des répertoires dont le nom contient votre login Unix, les effacer :

```
rm -rf /tmp/hadoop*
```

6. Formater le HDFS

```
hdfs namenode -format
```

7. Démarrer HDFS

```
start-dfs.sh
```

Vérifier que HDFS est démarré en vous rendant à l'URL `http://localhost:9870` et contrôler que la ligne `Configured Capacity` n'indique pas 0 octets.

8. Créer une arborescence de répertoire sur HDFS et y déposer les fichiers textes du TP (on suppose que le répertoire courant contient les fichiers texte). Il faut remplacer `VOTRELOGIN` par votre login Unix.

```
hdfs dfs -mkdir -p /user/VOTRELOGIN/input/
hdfs dfs -put *.txt /user/VOTRELOGIN/input/
```

9. Vérifier que les fichiers sont bien présents :

```
hdfs dfs -ls /user/VOTRELOGIN/input/*
```

Exercice 1

Dans cet exercice, on va implémenter un comptage de mot, c'est à dire utiliser Map/Reduce pour compter de manière distribuée¹ le nombre d'occurrences de chaque mots dans un texte.

1. Importez le projet Eclipse sur la page du cours
2. Lisez attentivement la classe DriverWordCount. La transformation utilise un *map* et un *reduce*. Le *map* prend en argument des paires (*nomdefichier, texte*) (on se fiche du nom du fichier). Le *reduce* prend en argument des chaînes de caractères et des tableaux de *null*
3. Compléter le code des méthodes *map* et *reduce* comme vu en cours pour compter les occurrences de chaque mot. On pourra convertir le Text d'entrée du *map* en une chaîne Java et utiliser la méthode `split` des chaînes pour la découper en tableau de mots.
4. Une fois le code complété, vous devez générer un fichier jar à partir du projet (Export -> Jar file) à l'endroit que vous souhaitez puis tester avec :

```
hdfs dfs -rm -r /user/VOTRELOGIN/output/
```

```
hadoop jar ~/Untitled.jar tdd.tp03.DriverWordCount \  
/user/VOTRELOGIN/output /user/VOTRELOGIN/input/jungle2.txt
```

vérifier le résultat du calcul (s'il n'y a pas eu d'erreur :)

```
hdfs dfs -cat /user/VOTRELOGIN/output/* | less
```

(q pour quitter).

Exercice 2

Copier (sous eclipse) votre classe DriverWordCount en une classe DriverInter. Modifier le code pour qu'il effectue la transformation suivante :

entrée une suite de fichier contenant sur chaque ligne 2 nombres séparés par un espace

sortie l'intersection des 2 colonnes, c'est à dire l'ensemble des nombres qui apparaissent en première position et en deuxième position au moins une fois dans les fichiers.

Exercice 3

Copier (sous eclipse) votre classe DriverWordCount en une classe DriverUnion. Modifier le code pour qu'il effectue la transformation suivante :

entrée une suite de fichier contenant sur chaque ligne 2 nombres séparés par un espace

sortie l'union des 2 colonnes, c'est à dire l'ensemble des nombres qui apparaissent en première position ou en deuxième position au moins une fois dans les fichiers.

Exercice 4

On considère les deux fichiers `exo2_1.txt` et `exo2_2.txt`. Proposer un algorithme (en pseudo-code) pour calculer leur jointure sur la première colonne.

Implémenter cette solution.

1. en pratique, vous n'aurez qu'un seul *worker*, la machine sur laquelle vous vous trouvez, mais il suffit d'ajouter des machines autres que localhost dans le fichier de configuration pour avoir plusieurs nœuds, le reste est identique