Manipulations et exécution de programmes map-reduce sous hadoop

 # Version Hadoop :
hadoop version

 # Affichage des fichiers sous un répertoire
Hadoop fs -ls TP1/
hadoop fs -ls -R TP1/

 # Déplacer un répertoire
hadoop fs -mv <src> <dest>

 # Copier un fichier de HDFS dans HDFS
hadoop fs -cp <src> <dest>
Exemple : <src> TP1/data/purchases.txt
<dest> TP1/data/

 # Télécharger un fichier
wget ….

Afficher le dernier kilo-octet du fichier sur stdout.
hadoop fs -tail hadoop/purchases.txt

Pour afficher le contenu de votre fichier texte purchase.txt
qui est présent dans votre répertoire hadoop.
hadoop fs -cat hadoop/purchases.txt

Attribuer la permission à un fichier
chmod +x code/reducer.py

Application 1 : Wordcount

Créer les deux codes python suivants :
Programme Mapper

```
#!/usr/bin/env python2
import sys
for line in sys.stdin:
```

```
    line = line.strip()
    words = line.split()
    for word in words:
        print '%s\t%s' % (word, 1)
```

Programme Reducer

```
#!/usr/bin/env python2
from operator import itemgetter
import sys
current_word = None
current_count = 0
word = None
for line in sys.stdin:
    line = line.strip()
    word, count = line.split('\t', 1)
    try:
        count = int(count)
    except ValueError:
        continue
    if current_word == word:
        current_count += count
    else:
        if current_word:
            print '%s\t%s' % (current_word, current_count)
        current_count = count
        current_word = word
if current_word == word:
    print '%s\t%s' % (current_word, current_count)
```

 # Tester votre programme mapper

```
echo "le groupe CD de troisième année de cette année est formidable !!!" |
code/mapper2.py
```

 # Tester vos codes en local :

```
head -50 Downloads/purchases.txt |code/mapper2.py |sort |code/reducer.py
```

 # Exécuter map-reduce sous hadoop :

hadoop jar /usr/lib/hadoop-0.20-mapreduce/contrib/streaming/hadoop-streaming-2.6.0-mr1-cdh5.13.0.jar -mapper code/mapper2.py -reducer code/reducer.py -file code/mapper2.py -file code/reducer.py -input myinput/purchases.txt -output joboutput

Remarque : nombre de map = nombre de block (en fonction du volume du fichier)
Généralement le nombre de reducer est au voisinage de 10% du nombre de mapper

 # Afficher le fichier de sortie :
hadoop fs -tail output/part-r-00000

 # consulter la liste des tâches map-reduce
mapred job -list all

 # consulter le statut du job
mapred job -status job_1670364086924_0001

#😉 Forcer l'arrêt une tâche

mapred job -status job_1670364086924_0002

Remarque : vous pouvez consulter ces informations via la page web.
--- all applications

Pour aller loin :
Utiliser un cluster AWS-EMR (Elastic Map Reduce) de Amazon pour exécuter un Job Map Reduce de votre choix sur un vrai cluster distribué. [https://aws.amazon.com/fr/emr/]