

A dark blue vertical bar is on the left. A blue arrow points right from it, containing the date.

23/12/2019

# Projet MapReduce & HDFS

## Module : Hadoop & MapReduce

Intervenant :Taoufik MEGDOULI

Several thin, curved lines in shades of blue and grey originate from the bottom left and sweep upwards and to the right.

### Etudiants EBAM UTT 2019-2020

- AMAIRA Zahira
- SINDIMWO Sophie

## Table des matières

<b>1. MapReduce .....</b>	<b>2</b>
<b>1.1 Code Map .....</b>	<b>2</b>
<b>1.2 Code Reducer .....</b>	<b>3</b>
<b>1.3 Exécution du maper et du reducer en local .....</b>	<b>4</b>
<b>1.4 Exécution de map Reduce sur Hadoop .....</b>	<b>5</b>
<b>1.5 Résultat .....</b>	<b>7</b>
<b>2. Utilisation de HDFS3 .....</b>	<b>9</b>
<b>2.1 Programme python sans suppression des caractères spéciaux (word_count1.py) .....</b>	<b>10</b>
<b>2.2 Programme python avec suppression des caractères spéciaux (word_count2.py) .....</b>	<b>11</b>
<b>2.3 Exécution du programme de comptage sans suppression des caractères spéciaux .....</b>	<b>12</b>
<b>2.4 Exécution du programme de comptage avec suppression des caractères spéciaux .....</b>	<b>13</b>

# 1. MapReduce

**Objectif :** calculer le nombre de vols ayant quitté Sydney chaque mois, les données sont disponibles dans un fichier CSV (International\_airline\_activity\_Australia.csv)

Les étapes pour faire ce calcul :

- Création d'un code Map qui permet de faire un filtre sur les vols de la ville de Sydney, le mois, l'information sur les arrêts, le nombre de passager.
- Création d'un code Reduce qui permet de compter le nombre de vol par mois
- Tester le code en local
- Exécuter le code sur Hadoop

## 1.1 Code Map

```
#!/home/cloudera/anaconda3/bin/python3.7

import sys
wordList = dict()
# input venant de STDIN (standard input)
for line in sys.stdin:
    # Suppression d'espaces vides à la fin de chaque ligne
    line = line.strip()
    # découpage en mots, on découpe avec le symbole ";", car le input est un fichier csv
    words = line.split(';')

    if words[2].strip()=="Sydney" and words[1].strip()=="0" and words[10].strip()=="0":
        # Pour chaque ligne, afficher certaines des valeurs dont on a besoin
        # dans le mapper sur la console STDIN
        # Dans le cadre de ce travail, on récupère
        # informations sur l'année et le nombre de passagers uniquement
        # pour les lignes les Vols entrant et sortant == "0",
        # la ville == "Sydney",
        # l'information sur l'arrêts (transit ou final) == "0"
        print(words[0].strip()+'|'+words[12].strip())
```

## 1.2 Code Reducer

```
#!/home/cloudera/anaconda3/bin/python3.7

from operator import itemgetter
import sys
PeriodeVols = dict()
# Lecture des données renvoyées par le mapper
for line in sys.stdin:
    # suppression d'éventuels espaces vides
    line = line.strip()

    # on récupère les input recus de mapper.py
    mois_Annee, Nbre_Passagers = line.split('|')

    try:
        Nbre_Passagers = int(Nbre_Passagers)
    except ValueError:
        # test sur le contenu de Nbre_Passagers
        #pour s'assurer que c'est un int, sinon, on l'ignore
        continue

    if mois_Annee in PeriodeVols :
        # Pour chaque valeur mois+annee
        PeriodeVols[mois_Annee]+=Nbre_Passagers
        #Sommer le nombre de passagers par mois+annee
    else:
        PeriodeVols[mois_Annee]=Nbre_Passagers
        # première accès à l'occurrence de la clé mois+annee,
        # on affecte stocke la la valeur du nombre de passagers
        #dans le dictionnaire

#On affiche le contenu de la liste : la cle "mois+anne"
#et la valeur agrège y relatif: nombre de passagers
for key in PeriodeVols:
    print(key+' :'+str(PeriodeVols[key]))
```

- ➔ Création d'un répertoire (input) dans HDFS et chargement du fichier csv dans ce répertoire

```
##Création fichiers dans HDFS

hadoop fs -mkdir /input

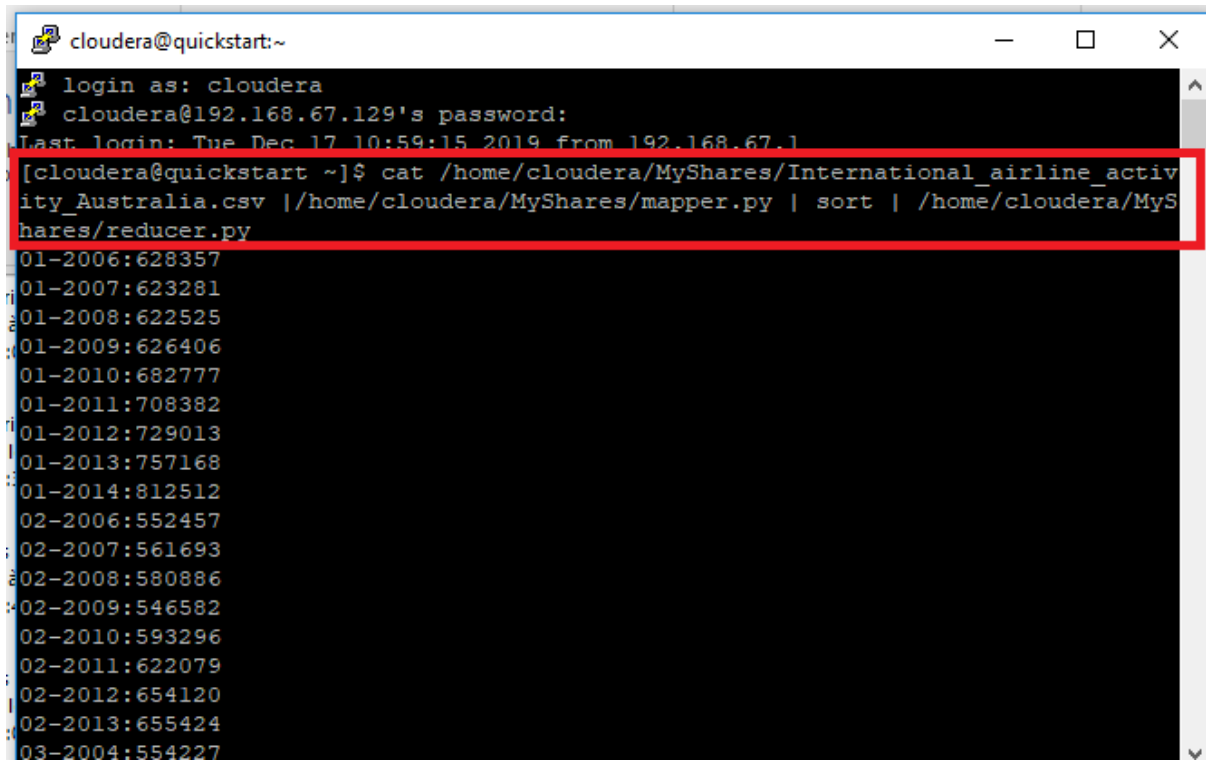
#import du fichier dans HDFS
hadoop fs -put /home/cloudera/MyShares/International_airline_activity_Australia.csv /input
```

### 1.3 Exécution du mapper et du reducer en local

```
##Lancement en local

cat /home/cloudera/MyShares/International_airline_activity_Australia.csv
|/home/cloudera/MyShares/mapper.py | sort |
/home/cloudera/MyShares/reducer.py
```

➔ Aperçu du résultat



```
cloudera@quickstart:~
login as: cloudera
cloudera@192.168.67.129's password:
Last login: Tue Dec 17 10:59:15 2019 from 192.168.67.1
[cloudera@quickstart ~]$ cat /home/cloudera/MyShares/International_airline_activ
ity_Australia.csv | /home/cloudera/MyShares/mapper.py | sort | /home/cloudera/MyS
hares/reducer.py
01-2006:628357
01-2007:623281
01-2008:622525
01-2009:626406
01-2010:682777
01-2011:708382
01-2012:729013
01-2013:757168
01-2014:812512
02-2006:552457
02-2007:561693
02-2008:580886
02-2009:546582
02-2010:593296
02-2011:622079
02-2012:654120
02-2013:655424
03-2004:554227
```

## 1.4 Exécution de map Reduce sur Hadoop

```
##Lancement sur HADOOP

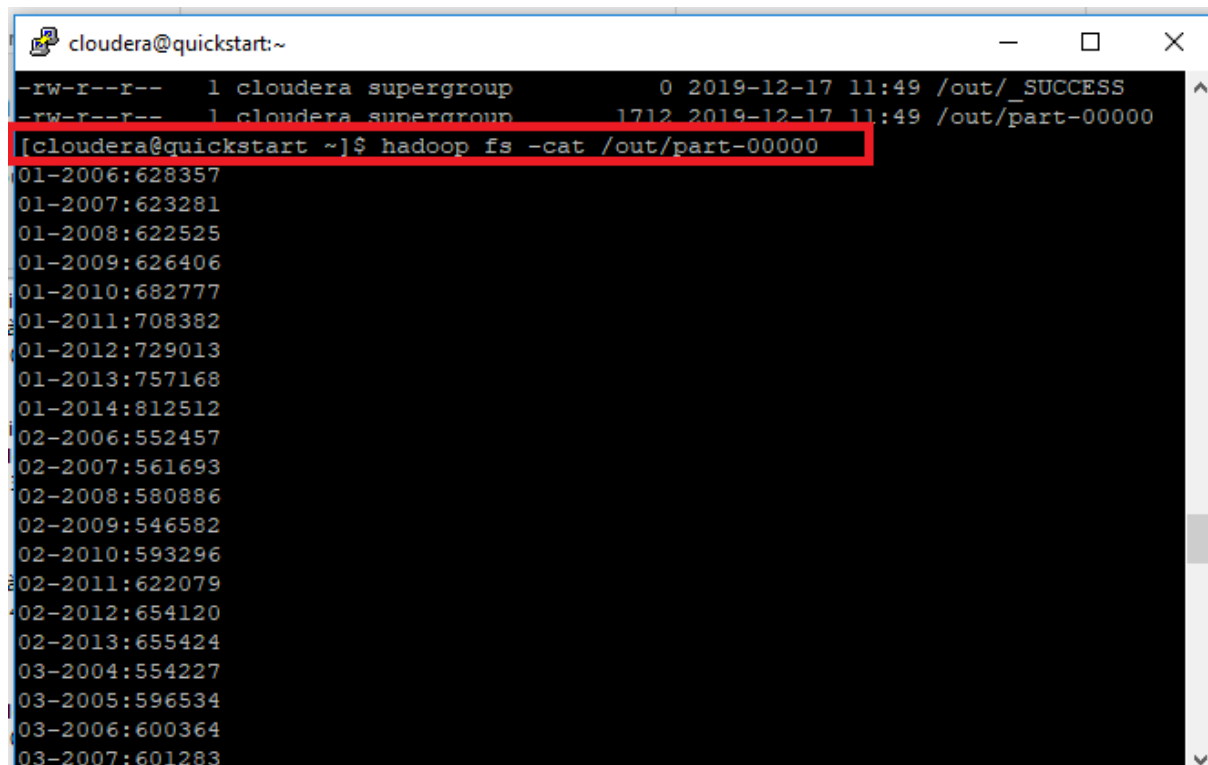
hadoop jar /usr/lib/hadoop-0.20-mapreduce/contrib/streaming/hadoop-streaming-2.6.0-mr1-cdh5.13.0.jar
-mapper "/home/cloudera/MyShares/mapper.py" -reducer "/home/cloudera/MyShares/reducer.py"
-input /user/input/International_airline_activity_Australia.csv -output /out
```

### ➔ Aperçu du résultat

```
cloudera@quickstart:~
12-2005:618713
12-2006:619897
12-2007:612876
12-2008:618514
12-2009:667769
12-2010:689319
12-2011:711800
12-2012:750584
12-2013:797045
[cloudera@quickstart ~]$ hadoop fs -rm -r /out
Deleted /out
[cloudera@quickstart ~]$ hadoop jar /usr/lib/hadoop-0.20-mapreduce/contrib/streaming/hadoop-streaming-2.6.0-mr1-cdh5.13.0.jar -mapper "/home/cloudera/MyShares/mapper.py" -reducer "/home/cloudera/MyShares/reducer.py" -input /user/input/International_airline_activity_Australia.csv -output /out
packageJobJar: [] [/usr/lib/hadoop-mapreduce/hadoop-streaming-2.6.0-cdh5.13.0.jar] /tmp/streamjob5729542731726069764.jar tmpDir=null
19/12/17 11:47:26 INFO client.RMPProxy: Connecting to ResourceManager at /0.0.0.0:8032
19/12/17 11:47:26 INFO client.RMPProxy: Connecting to ResourceManager at /0.0.0.0:8032
19/12/17 11:47:28 INFO mapred.FileInputFormat: Total input paths to process : 1
19/12/17 11:47:28 INFO mapreduce.JobSubmitter: number of splits:2
19/12/17 11:47:28 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_15...
```

```
cloudera@quickstart:~
Virtual memory (bytes) snapshot=4524679168
Total committed heap usage (bytes)=392372224
Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
File Input Format Counters
Bytes Read=6130129
File Output Format Counters
Bytes Written=1712
19/12/17 11:49:05 INFO streaming.StreamJob: Output directory: /out
[cloudera@quickstart ~]$ hadoop fs -ls /out
Found 2 items
-rw-r--r-- 1 cloudera supergroup 0 2019-12-17 11:49 /out/_SUCCESS
-rw-r--r-- 1 cloudera supergroup 1712 2019-12-17 11:49 /out/part-000000
[cloudera@quickstart ~]$ hadoop fs -cat /out/part-000000
01-2006:628357
01-2007:623281
01-2008:622525
01-2009:626406
01-2010:682777
```

Le résultat montre que la requête hadoop map reduce a été exécutée avec succès.

A terminal window titled 'cloudera@quickstart:~' with standard window controls. It displays the output of a Hadoop MapReduce job. The first two lines show file permissions and details for '/out/\_SUCCESS' and '/out/part-00000'. The third line shows the command '[cloudera@quickstart ~]\$ hadoop fs -cat /out/part-00000' highlighted with a red box. Below this, a list of 21 date-year pairs is displayed, representing the output of the map function.

```
cloudera@quickstart:~  
-rw-r--r--  1 cloudera supergroup      0 2019-12-17 11:49 /out/_SUCCESS  
-rw-r--r--  1 cloudera supergroup 1712 2019-12-17 11:49 /out/part-00000  
[cloudera@quickstart ~]$ hadoop fs -cat /out/part-00000  
01-2006:628357  
01-2007:623281  
01-2008:622525  
01-2009:626406  
01-2010:682777  
01-2011:708382  
01-2012:729013  
01-2013:757168  
01-2014:812512  
02-2006:552457  
02-2007:561693  
02-2008:580886  
02-2009:546582  
02-2010:593296  
02-2011:622079  
02-2012:654120  
02-2013:655424  
03-2004:554227  
03-2005:596534  
03-2006:600364  
03-2007:601283
```

## 1.5 Résultat

Copié à partir du fichier log

```
-rw-r--r-- 1 cloudera supergroup      0 2019-12-17 11:49 /out/_SUCCESS
-rw-r--r-- 1 cloudera supergroup 1712 2019-12-17 11:49 /out/part-00000
]0;cloudera@quickstart:~[cloudera@quickstart ~]$ hadoop fs -cat /out/part-00000
01-2006:628357
01-2007:623281
01-2008:622525
01-2009:626406
01-2010:682777
01-2011:708382
01-2012:729013
01-2013:757168
01-2014:812512
02-2006:552457
02-2007:561693
02-2008:580886
02-2009:546582
02-2010:593296
02-2011:622079
02-2012:654120
02-2013:655424
03-2004:554227
03-2005:596534
03-2006:600364
03-2007:601283
03-2008:610098
03-2009:596356
03-2010:634380
03-2011:676351
03-2012:675644
03-2013:716551
04-2006:581205
04-2007:563075
04-2008:579828
04-2009:582837
04-2010:608030
04-2011:663109
04-2012:674681
04-2013:708042
05-2006:576843
05-2007:562521
05-2008:593481
05-2009:586048
05-2010:603295
05-2011:650324
05-2012:663790
05-2013:692525
06-2004:539183
```



06-2005:569057  
06-2006:567687  
06-2007:553939  
06-2008:571365  
06-2009:564161  
06-2010:594696  
06-2011:614870  
06-2012:669329  
06-2013:690990  
07-2006:600386  
07-2007:583057  
07-2008:614763  
07-2009:612112  
07-2010:642945  
07-2011:689314  
07-2012:720682  
07-2013:750102  
08-2006:588643  
08-2007:575999  
08-2008:596635  
08-2009:604721  
08-2010:638477  
08-2011:677257  
08-2012:703989  
08-2013:726781  
09-2003:487489  
09-2004:550141  
09-2005:579150  
09-2006:571883  
09-2007:564396  
09-2008:562295  
09-2009:584070  
09-2010:616638  
09-2011:661884  
09-2012:690536  
09-2013:721742  
10-2006:593800  
10-2007:585538  
10-2008:585773  
10-2009:619346  
10-2010:643887  
10-2011:696907  
10-2012:712018  
10-2013:749828  
11-2006:573123  
11-2007:576085  
11-2008:585932  
11-2009:599650  
11-2010:630428  
11-2011:662271  
11-2012:694402  
11-2013:729524

```
12-2003:557005
12-2004:606719
12-2005:618713
12-2006:619897
12-2007:612876
12-2008:618514
12-2009:667769
12-2010:689319
12-2011:711800
12-2012:750584
12-2013:797045
```

## 2. Utilisation de HDFS3

**Objectif :** comptage du nombre de mots présents dans 4 fichiers txt avec l'utilisation de HDFS3

**Les étapes suivies :**

- Création d'un répertoire (data\_in) dans HDFS et chargement des 4 fichiers txt.
- Création d'un code python qui permet de compter le nombre de mot présent dans tous les fichiers et affichage du nombre de mots répétés et les 10 mots les plus répétés.

Dans les 4 fichiers il y a des caractères spéciaux ( , . ) et sans suppression de ces caractères spéciaux, un mot avec un caractère spécial peut être compté autant qu'un autre mot, pour illustration dans l'exemple ci-dessous « ex » et « ex. » sont comptés autant que deux mots différents et non pas un seul.

```
Cras quis egestas ex. Praesent luctus ex arcu, ac pharetra ligula sagittis nec.
```

Deux programmes python ont été créé :

- Un programme qui compte le nombre de mots dans les 4 fichiers txt sans suppression des caractères spéciaux (word\_count1.py)
- Un programme qui compte le nombre de mots dans les 4 fichiers txt avec suppression des caractères spéciaux (word\_count2.py)

➔ Création du répertoire data\_in dans HDFS3

```
##Création répertoire dans HDFS
hadoop fs -mkdir /data_in
```

➔ Chargement des 4 fichiers txt dans le répertoire data\_in dans HDFS

```
#Chargement des 4 fichiers dans HDFS

hadoop fs -put /home/cloudera/SHARES/data1.txt /data_in
hadoop fs -put /home/cloudera/SHARES/data2.txt /data_in
hadoop fs -put /home/cloudera/SHARES/data3.txt /data_in
hadoop fs -put /home/cloudera/SHARES/data4.txt /data_in
```

## 2.1 Programme python sans suppression des caractères spéciaux (word\_count1.py)

```
#!/home/cloudera/anaconda3/bin/python3.7

##Import de la librairie HDFS3
import hdfs3
##import du dictionnaire de la librairie collections
from collections import defaultdict, Counter
from hdfs3 import HDFFileSystem

hdfs=HDFFileSystem(host='localhost',port=8020)
##affiche la liste des fichiers disponible dans le répertoire
filenames = hdfs.glob('/data_in')
print('Liste des fichiers :')
print(filenames)
```

```
##fonction permet de compter le nombre de mot d'un fichier
def count_words(file):
    word_counts = defaultdict(int)
    for line in file:
        #utilisation de la fonction decode pour l'encodage
        line = line.decode('utf8').strip()
        for word in line.split():
            word_counts[word] += 1
    return word_counts

##fonction permet de compter le nombre dans l'ensemble des fichiers disponibles dans le répertoire data_in
all_counts = Counter()
for fn in filenames:
    ##lecture fichiers avec hdfs
    with hdfs.open(fn) as f:
        data=f.read()
        counts = count_words([data])
        all_counts.update(counts)

##affiche le nombre de mot répétés
print('Le nombre de mots qui se repète SANS SUPPRESSION DES CARACTERES SPECIAUX : '+str(len(all_counts)))

##affiche le top 10 des mots les plus repetés
print('Le top 10 des mots les plus repetés SANS SUPPRESSION DES CARACTERES SPECIAUX est : ')

for k,v in sorted(all_counts.items(), key=lambda p:p[1], reverse=True)[:10]:
    print('le mot ' + k, 'est repeté ' + str(v) + ' fois')
    print("\n")
```

## 2.2 Programme python avec suppression des caractères spéciaux (word\_count2.py)

```
#!/home/cloudera/anaconda3/bin/python3.7

##Import de la librairie HDFS3
import hdfs3
##import du dictionnaire de la librairie collections
from collections import defaultdict, Counter
from hdfs3 import HDFFileSystem

hdfs=HDFFileSystem(host='localhost',port=8020)

##affiche la liste des fichiers disponible dans le répertoire
filenames = hdfs.glob('/data_in')
print('liste des fichiers :')
print(filenames)

##fonction permet de supprimer les caractères spéciaux (.,, *)
def cleanse_word(word):
    # find regex for word
    return word.lower().strip(',').strip('.').strip('\').strip(' ').strip('*').strip('?').strip('!').strip(';').strip(':')

##fonction permet de calculer le nombre de mot d'un fichier
def count_words(file):
    word_counts = defaultdict(int)
    for line in file:
        #utilisation de la fonction decode pour l'encodage
        line = line.decode('utf8').strip()
        for word in line.split():
            #appel de la fonction pour suppression des caractères spéciaux
            word = cleanse_word(word)
            word = (word)
            word_counts[word] += 1
    return word_counts

##fonction permet de calculer le nombre de mot pour l'ensemble des fichiers disponible dans le répertoire data_in
all_counts = Counter()
for fn in filenames:
    ##lecture des fichiers
    with hdfs.open(fn) as f:
        data=f.read()
        counts = count_words([data])
        all_counts.update(counts)

## Affichage du nombre des mots répétés
print('Le nombre de mots qui se repète EN SUPPRIMANT LES CARACTERES SPECIAUX: '+str(len(all_counts)))
##Affichage du top 10 des mots les plus répétés en supprimant les caractères spéciaux
print('Le top 10 des mots les plus répétés EN SUPPRIMANT LES CARACTERES SPECIAUX est : ')

for k,v in sorted(all_counts.items(), key=lambda p:p[1], reverse=True)[:10]:
    print('le mot ' + k, 'est repeté ' + str(v) + ' fois')
    print("\n")
```

## 2.3 Exécution du programme de comptage sans suppression des caractères spéciaux

```
#Lancement du programme python sans suppression des caractères spéciaux
[cloudera@quickstart SHARES]$ chmod 777 word_count1.py
[cloudera@quickstart SHARES]$ ./word_count.py
```

### → Résultats sans suppression des caractères spéciaux

```
[cloudera@quickstart SHARES]$ chmod 777 word_count1.py
[cloudera@quickstart SHARES]$ ./word_count1.py
Liste des fichiers :
['/data_in/data1.txt' '/data_in/data2.txt' '/data_in/data3.txt' '/data_in/data4.txt']
Le nombre de mots qui se repète SANS SUPPRESSION DES CARACTERES SPECIAUX : 465
Le top 10 des mots les plus répétés SANS SUPPRESSION DES CARACTERES SPECIAUX est :
le mot et est repeté 544 fois

le mot sit est repeté 514 fois

le mot ac est repeté 504 fois

le mot in est repeté 477 fois

le mot sed est repeté 452 fois

le mot id est repeté 425 fois

le mot eget est repeté 419 fois

le mot ut est repeté 415 fois

le mot quis est repeté 415 fois

le mot vel est repeté 413 fois
```

## 2.4 Exécution du programme de comptage avec suppression des caractères spéciaux

```
#Lancement du programme python avec suppression des caractères spéciaux
[cloudera@quickstart SHARES]$ chmod 777 word_count2.py
[cloudera@quickstart SHARES]$ ./word_count2.py
```

➔ Résultat prenant en compte la suppression des caractères spéciaux

```
[cloudera@quickstart SHARES]$ chmod 777 word_count2.py
[cloudera@quickstart SHARES]$ ./word_count2.py
Liste des fichiers :
['/data_in/data1.txt', '/data_in/data2.txt', '/data_in/data3.txt', '/data_in/data4.txt']
Le nombre de mots qui se repète EN SUPPRIMANT LES CARACTERES SPECIAUX: 186
Le top 10 des mots les plus répétés EN SUPPRIMANT LES CARACTERES SPECIAUX est :
le mot sed est repeté 845 fois

le mot in est repeté 758 fois

le mot ut est repeté 656 fois

le mot et est repeté 619 fois

le mot ac est repeté 580 fois

le mot sit est repeté 514 fois

le mot amet est repeté 514 fois

le mot id est repeté 502 fois

le mot vel est repeté 500 fois

le mot nec est repeté 498 fois

[cloudera@quickstart SHARES]$
```

### Conclusion :

- Le nombre total de mots qui se répètent sans enlever les caractères spéciaux est de **465**, et en supprimant les caractères spéciaux ce nombre est de seulement **186**.
- Le mot « **et** » est répété plus de **544 fois** sans enlever les caractères spéciaux, alors qu'en supprimant les caractères ce mot est répété **619 fois**, cela veut dire que le mot « **et** » ayant des caractères spéciaux (.,) est de **75**