Nom & Prénom: **GUIZANI Ahmed**
Class: **5A SWE Paris**

LAB2 Big DATA

-Le code source du projet sont disponibles en annexes.
-Un seul programme et algorithme de Map-Reduce ont été développés.
-Le traitement est réalisée grâce à des conditions “if”.

Voici comment exécuter le programme, avec les arguments à utiliser pour spécifier quel type d’analyse effectuer. Si <task_name> n’a pas été fourni, cette aide s’affiche.

```
Usage: salesanalysis.jar <input_file> <output_directory> <task_name> <task_arg>
        - <input_file>: absolute/relative path of the CSV file in hdfs ;
        - <output_directory>: path of the output directory in hdfs, in which Hadoop will generate the output files and logs ;
        - <task_name>: the type of analysis task to perform on the CSV file. Possible values and associated argument:
                - TOTAL_PROFIT_REGION <region_name>: obtain the total profit for the given world region ;
                - TOTAL_PROFIT_COUNTRY <country_name>: obtain the total profit for the given country ;
                - TOTAL_PROFIT_ITEM_TYPE <item_type>: obtain the total profit for the given item type ;
                - SALES_PER_ITEM_TYPE_AND_SALES_CHANNEL: obtain how many sales were performed per item type and sales channel (online/offline)
                - TOTAL_PROFIT_PER_ITEM_TYPE_AND_SALES_CHANNEL: obtain the total profit per item type and sales channel (online/offline) ;
        - <task_arg>: argument associated with the type of task to perform (<region_name>, <country_name> or <item_type>).

Example
        salesanalysis.jar /home/training/LAB2/input/sales.csv /home/training/LAB2/output/1 TOTAL_PROFIT_REGION
```

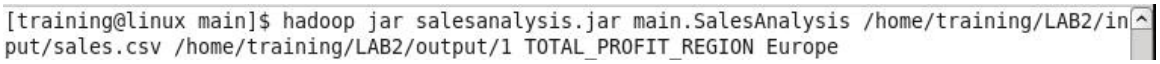
Toutes les paires de clé-valeur retournées possèdent le format <Text, DoubleWritable>, et les résultats obtenus lors des tests ont été vérifiés sous Excel.

Question 1: Obtain the total profit for any given world region.

Voici ci dessous un exemple de commande pour utiliser cette fonctionnalité. Cet exemple calcule le profit total pour l’Europe.

```
hadoop jar salesanalysis.jar main.SalesAnalysis /home/training/LAB2/input/sales.csv /home/training/LAB2/output/1 TOTAL_PROFIT_REGION Europe
```

La capture d’écran ci dessous présente l’exécution de cet exemple avec Hadoop,



```
WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments.
ld implement Tool for the same.
INFO input.FileInputFormat: Total input paths to process : 1
WARN snappy.LoadSnappy: Snappy native library is available
INFO snappy.LoadSnappy: Snappy native library loaded
INFO mapred.JobClient: Running job: job_202109290852_0019
INFO mapred.JobClient:  map 0% reduce 0%
INFO mapred.JobClient:  map 100% reduce 0%
INFO mapred.JobClient:  map 100% reduce 100%
INFO mapred.JobClient: Job complete: job_202109290852_0019
```

Et voici ci dessous le résultat retourné dans le fichier de sortie “part-r-00000”.

```
[training@linux main]$ hdfs dfs -cat /home/training/LAB2/output/1/part-r-00000
Europe  1.0269996127999989E9
[training@linux main]$
```

Avec Excel, on obtient 1026999613, ce qui correspond au résultat obtenu deja .

Question 2: Obtain the total profit for any given country.

Voici ci dessous un exemple de commande pour utiliser cette fonctionnalité. Cet exemple calcule le profit total pour la France.

```
hadoop jar salesanalysis.jar main.SalesAnalysis /home/training/LAB2/input/sales.csv
/home/training/LAB2/output/1 TOTAL_PROFIT_COUNTRY France
```

La capture d’écran ci dessous présente l’exécution de cet exemple avec Hadoop,

```
[training@linux main]$ hadoop jar salesanalysis.jar main.SalesAnalysis /home/training/LAB2/in
put/sales.csv /home/training/LAB2/output/1 TOTAL_PROFIT_COUNTRY France
```

```
WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments.
ld implement Tool for the same.
INFO input.FileInputFormat: Total input paths to process : 1
WARN snappy.LoadSnappy: Snappy native library is available
INFO snappy.LoadSnappy: Snappy native library loaded
INFO mapred.JobClient: Running job: job_202109290852_0021
INFO mapred.JobClient:  map 0% reduce 0%
INFO mapred.JobClient:  map 100% reduce 0%
INFO mapred.JobClient:  map 100% reduce 100%
INFO mapred.JobClient: Job complete: job_202109290852_0021
```

Et voici ci dessous le résultat retourné dans le fichier de sortie “part-r-00000”.

```
[training@linux main]$ hdfs dfs -cat /home/training/LAB2/output/1/part-r-00000
France  1.9199104959999997E7
[training@linux main]$
```

Question 3: Obtain the total profit for any given item type.

Voici ci dessous un exemple de commande pour utiliser cette fonctionnalité. Cet exemple calcule le profit total pour la catégorie “vêtements”.

```
hadoop jar salesanalysis.jar main.SalesAnalysis /home/training/LAB2/input/sales.csv
/home/training/LAB2/output/1 TOTAL_PROFIT_ITEM_TYPE Clothes
```

La capture d'écran ci dessous présente l'exécution du programme avec Hadoop.

```
[training@linux main]$ hadoop jar salesanalysis.jar main.SalesAnalysis /home/training/LAB2/in
put/sales.csv /home/training/LAB2/output/1 TOTAL_PROFIT_ITEM_TYPE Clothes
```

```
 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments.
ld implement Tool for the same.
 INFO input.FileInputFormat: Total input paths to process : 1
 WARN snappy.LoadSnappy: Snappy native library is available
 INFO snappy.LoadSnappy: Snappy native library loaded
 INFO mapred.JobClient: Running job: job_202109290852_0022
 INFO mapred.JobClient:  map 0% reduce 0%
 INFO mapred.JobClient:  map 100% reduce 0%
 INFO mapred.JobClient:  map 100% reduce 100%
 INFO mapred.JobClient: Job complete: job_202109290852_0022
```

Et voici ci dessous le résultat retourné dans le fichier de sortie "part-r-00000".

```
[training@linux main]$ hdfs dfs -cat /home/training/LAB2/output/1/part-r-00000
Clothes 3.1963658399999976E8
[training@linux main]$
```

Question 4: For each item type, provide:

- How many sales were performed online.
- How many sales were performed offline.

Pour information, nous avons interprété "how many sales" comme faisant référence à une quantité d'unités vendues (colonne "Units Sold").

Voici un exemple de commande pour utiliser cette fonctionnalité. Cet exemple calcule le nombre total d'unités vendues par type de produit et par canal de vente (offline/online).

```
hadoop jar salesanalysis.jar main.SalesAnalysis /home/training/LAB2/input/sales.csv
/home/training/LAB2/output/1 SALES_PER_ITEM_TYPE_AND_SALES_CHANNEL
```

La capture d'écran ci dessous présente l'exécution du programme avec Hadoop.

```
[training@linux main]$ hadoop jar salesanalysis.jar main.SalesAnalysis /home/training/LAB2/in
put/sales.csv /home/training/LAB2/output/1 SALES_PER_ITEM_TYPE_AND_SALES_CHANNEL
```

```
 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments.
ld implement Tool for the same.
 INFO input.FileInputFormat: Total input paths to process : 1
 WARN snappy.LoadSnappy: Snappy native library is available
 INFO snappy.LoadSnappy: Snappy native library loaded
 INFO mapred.JobClient: Running job: job_202109290852_0024
 INFO mapred.JobClient:  map 0% reduce 0%
 INFO mapred.JobClient:  map 100% reduce 0%
 INFO mapred.JobClient:  map 100% reduce 100%
 INFO mapred.JobClient: Job complete: job_202109290852_0024
```

Et voici ci dessous le résultat retourné dans le fichier de sortie “part-r-00000”.

```
[training@linux main]$ hdfs dfs -cat /home/training/LAB2/output/1/part-r-00000
Baby Food (Offline)     1997291.0
Baby Food (Online)      2199715.0
Beverages (Offline)     1944340.0
Beverages (Online)      1966096.0
Cereal (Offline)        2041649.0
Cereal (Online) 2161745.0
Clothes (Offline)       2213549.0
Clothes (Online)        2138801.0
Cosmetics (Offline)     2066494.0
Cosmetics (Online)      2036796.0
Fruits (Offline)        1869056.0
Fruits (Online) 2186865.0
Household (Offline)     2266555.0
Household (Online)      2070248.0
Meat (Offline)  1927007.0
Meat (Online)   2058383.0
Office Supplies (Offline)       2041378.0
Office Supplies (Online)        2078885.0
Personal Care (Offline) 2122789.0
Personal Care (Online)  2280038.0
Snacks (Offline)        2112292.0
Snacks (Online) 2010489.0
Vegetables (Offline)    2193905.0
Vegetables (Online)     2044193.0
[training@linux main]$
```

Question 5: and for each of those quantities, how much the combined total profit for those sales was.

Voici ci dessous un exemple de commande pour utiliser cette fonctionnalité. Cet exemple calcule le profit total combiné.

```
hadoop jar salesanalysis.jar main.SalesAnalysis /home/training/LAB2/input/sales.csv
/home/training/LAB2/output/1 TOTAL_PROFIT_PER_ITEM_TYPE_AND_SALES_CHANNEL
```

La capture d’écran ci dessous présente l’exécution du programme avec Hadoop.

```
[training@linux main]$ hadoop jar salesanalysis.jar main.SalesAnalysis /home/training/LAB2/in
put/sales.csv /home/training/LAB2/output/1 TOTAL_PROFIT_PER_ITEM_TYPE_AND_SALES_CHANNEL
```

```
 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments.
ld implement Tool for the same.
 INFO input.FileInputFormat: Total input paths to process : 1
 WARN snappy.LoadSnappy: Snappy native library is available
 INFO snappy.LoadSnappy: Snappy native library loaded
 INFO mapred.JobClient: Running job: job_202109290852_0023
 INFO mapred.JobClient:  map 0% reduce 0%
 INFO mapred.JobClient:  map 100% reduce 0%
 INFO mapred.JobClient:  map 100% reduce 100%
 INFO mapred.JobClient: Job complete: job_202109290852_0023
```

Et voici ci dessous le résultat retourné dans le fichier de sortie “part-r-00000”.

```
[training@linux main]$ hdfs dfs -cat /home/training/LAB2/output/1/part-r-00000
Baby Food (Offline)     1.9146031525999987E8
Baby Food (Online)      2.108646799E8
Beverages (Offline)     3.0448364400000006E7
Beverages (Online)      3.0789063359999996E7
Cereal (Offline)        1.8086968490999985E8
Cereal (Online) 1.915089895499999E8
Clothes (Offline)       1.6256303855999982E8
Clothes (Online)        1.5707354544000015E8
Cosmetics (Offline)     3.5930131178000027E8
Cosmetics (Online)      3.5413772052000016E8
Fruits (Offline)        4504424.960000003
Fruits (Online) 5270344.65
Household (Offline)     3.7563616015000004E8
Household (Online)      3.4310220103999996E8
Meat (Offline)  1.1022480039999998E8
Meat (Online)   1.1773950760000001E8
Office Supplies (Offline)       2.577239725E8
Office Supplies (Online)        2.6245923125E8
Personal Care (Offline) 5.3197092339999974E7
Personal Care (Online)  5.713775227999998E7
Snacks (Offline)        1.1647178088000001E8
Snacks (Online) 1.1085836346000011E8
Vegetables (Offline)    1.3850122265000004E8
Vegetables (Online)     1.2904990409000003E8
[training@linux main]$
```