

Master Data Science & Big Data



Mémoire du Projet final *Big Data*

Réalisé par :

LGHAOUCH El Mehdi

Encadré par :

GHAZOUANI Mohammed

FSBM : 2021/2022

Table des matières

Introduction	4
Chapitre 1 : Big Data.....	5
1. Définition.....	5
2. Utilisation	5
3. Ecosystème du Big Data.....	5
3.1 Système de fichiers.....	5
3.2 Base NoSQL.....	6
3.3 Base NewSQL	6
3.4 Intégration de données.....	7
3.5 Programmation distribuée	7
3.6 Visualisation.....	8
4. Caractéristiques du Big Data	8
5. Conclusion.....	8
Chapitre 2 : Hadoop	9
1. Définition.....	9
2. L'écosystème utilisé dans le projet.....	9
2.1 HDFS.....	9
2.2 MapReduce.....	10
2.3 Hive	10
2.4 HBase	11
2.5 Nifi.....	11
2.6 Sqoop.....	11
2.7 Tableau	11
3. Conclusion.....	11
Chapitre 3 : Réalisation	12
1. Introduction :	12
2. Partie 1 : Utilisation du Nifi	12
3. Partie 2 : Hive.....	16
5.1 Création de la base de données 'ProjetF' dans Hive	17
5.2 Création de la table 'use_acc' dans 'ProjetF'	17
5.3 Chargement des données dans 'use_acc' à partir du HDFS	19
4. Partie 3 : Hbase.....	19
5. Partie 4 : MySQL et Sqoop	22
5.1 MySQL :.....	22
5.2 Sqoop.....	24
6. Conclusion.....	25

Chapitre 4 : Visualisation des données.....	26
1. Introduction :	26
2. La visualisation :	26
3. Conclusion :.....	30
Conclusion Générale	31

Introduction

Les accidents de la route sont devenus très fréquents de nos jours. Près de 1,25 million de personnes meurent chaque année dans des accidents de la route, soit en moyenne 3 287 décès par jour. De plus, 20 à 50 millions de personnes sont blessées ou handicapées chaque année. Les accidents de la route se classent au 9e rang des causes de décès et représentent 2,2 % de tous les décès dans le monde. Les accidents de la route coûtent 518 milliards de dollars dans le monde, coûtant à chaque pays 1 à 2 % de son PIB annuel.

Aux États-Unis, plus de 37 000 personnes meurent chaque année dans des accidents de la route et 2,35 millions sont blessées ou handicapées. Les accidents de la route coûtent aux États-Unis 230,6 milliards de dollars par an, soit une moyenne de 820 dollars par personne. Les accidents de la route sont la principale cause annuelle de décès de citoyens américains en bonne santé voyageant à l'étranger.

Le jeu de données est tiré de Kaggle. Il s'agit d'un ensemble de données sur les accidents de la circulation à l'échelle nationale, qui couvre 49 États des États-Unis. Les données sont collectées en continu de février 2016 à mars 2019, à l'aide de plusieurs fournisseurs de données, dont deux API qui fournissent des données d'événements de trafic en continu. Ces API diffusent les événements de trafic capturés par diverses entités, telles que les départements des transports des États-Unis et des États, les forces de l'ordre, les caméras de circulation et les capteurs de trafic au sein des réseaux routiers.

L'ensemble de données contient 2 243 939 (2,24 millions) de lignes et 49 colonnes (un ensemble de données assez volumineux). Un point à noter est que même si l'ensemble de données contient des données pour seulement trois ans, il y a déjà 2,24 millions d'accidents.

Dans ce projet, on utilise 1000000 (1 million) de lignes et 47 colonnes.

Chapitre 1 : Big Data

1. Définition

Le terme du 'Big Data' signifie les données de très grandes tailles ou données massives. Il désigne des ensembles très volumineux de données qu'aucun outil classique de la gestion de base de données ou gestion de l'information ne peut vraiment manipuler.

2. Utilisation

- L'utilisation du Big Data à la rescousse des ressources humaines et du recrutement.
- Une aide à la maintenance prédictive dans le milieu aéronautique.
- L'exploitation de données, une technique de fidélisation pour les banques et les assurances.
- Utiliser le Big Data pour personnaliser votre expérience sur les sites de e-commerce.

3. Ecosystème du Big Data

3.1 Système de fichiers

➤ HDFS



➤ GlusterFS



➤ CephFilesystem



3.2 Base NoSQL

➤ MongoDB



➤ Apache Hbase



➤ Cassandra



3.3 Base NewSQL

➤ Hive



➤ Apache DRILL



➤ Impala



3.4 Intégration de données

➤ **Sqoop**



➤ **Nifi**



➤ **Flume**



3.5 Programmation distribuée

➤ **MapReduce**



➤ **Apache Spark**



➤ **Pig**



3.6 Visualisation

➤ Tableau



➤ Spotfire



➤ ZoomData



4. Caractéristiques du Big Data

Les 5V du big data font référence à cinq éléments clés à prendre en compte et à optimiser dans le cadre d'une démarche d'optimisation de la gestion du big data. Ces 5V sont :

- **Volume** : la grande quantité d'information contenue dans ces bases de données.
- **Vélocité** : la vitesse de leur création, collecte, transmission et analyse.
- **Variété** : les différences de natures, formats et structures.
- **Valeur** : la capacité de ces données à générer du profit.
- **Véracité** : leur validité, i.e. qualité et précision ainsi que leur fiabilité

5. Conclusion

Dans ce chapitre, j'ai présenté le Big Data et j'ai parlé à l'utilisation de cet important outil. Et enfin j'ai cité quelques composants de son écosystème.

Chapitre 2 : Hadoop

1. Définition

Hadoop est un framework libre et open source écrit en Java destiné à faciliter la création d'applications distribués (au niveau du stockage des données et de leur traitement) et échelonnables (scalables) permettant aux applications de travailler avec milliers de nœuds et pétaoctets des données.

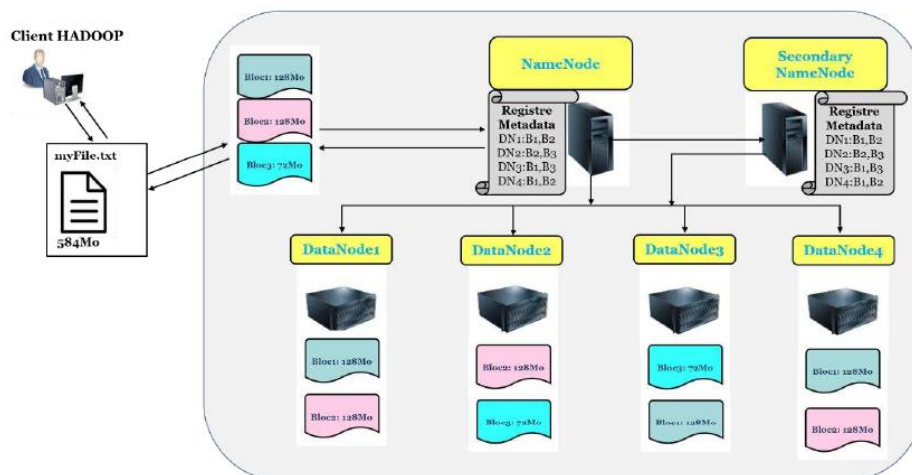
2. L'écosystème utilisé dans le projet



2.1 HDFS

HDFS est un système de fichiers distribué qui donne un accès haute performante aux données réparties dans les clusters Hadoop. Quand HDFS recueille une donnée, le système segmente l'information en plusieurs briques et les distribue sur plusieurs nœuds du cluster, ce qui permet alors le traitement en parallèle.

Architecture du cluster HDFS

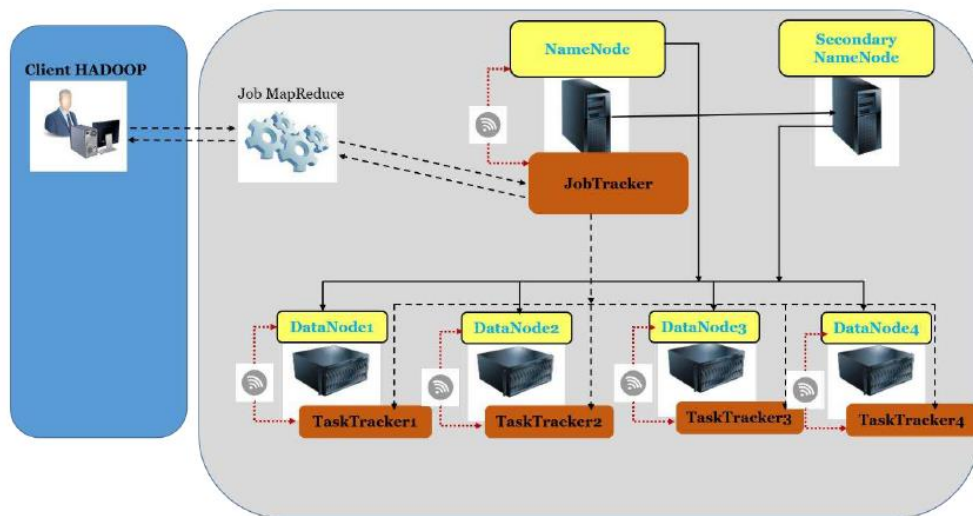


2.2 MapReduce

MapReduce est un moteur de traitement et d'analyse de données permettant les calculs parallèles. On distingue 4 étapes distinctes dans un traitement MapReduce :

- Découper (Split)
- Mapper
- Grouper (Shuffle)
- Réduire (Reduce)

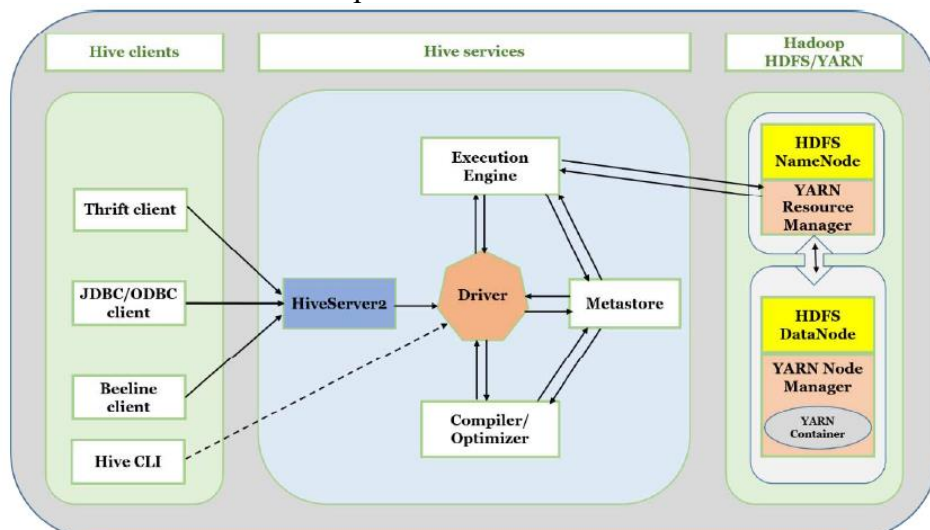
Architecture du cluster MapReduce / HDFS



2.3 Hive

Hive est un système d'entrepôt de données open source. Il permet d'interroger et d'analyser des ensembles de données volumineux (Big Data) stockés dans des fichiers Hadoop. En termes de langage, Hive propose HiveQL, un langage déclaratif, similaire à SQL. Il s'agit d'un système qui maintient des métadonnées décrivant les données stockées dans un HDFS.

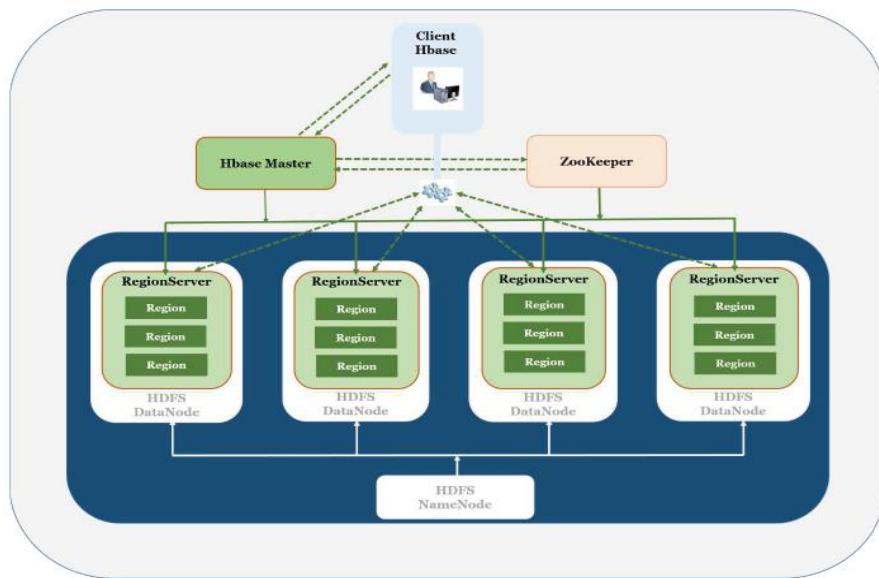
Les composants d'architecture Hive



2.4 HBase

HBase est un système de gestion de base de données non relationnelles distribué, écrit en Java, disposant d'un stockage structuré pour les grandes tables. La base de données HBase s'installe généralement sur le système de fichiers HDFS d'Hadoop pour faciliter la distribution, même si ce n'est pas obligatoire.

Architecture du cluster HBase



2.5 Nifi

Nifi est un logiciel libre de gestion de flux de données. Il permet de gérer et d'automatiser des flux de données entre plusieurs systèmes informatiques, à partir d'une interface web et dans un environnement distribué.

2.6 Sqoop

Sqoop est une interface en ligne de commande de l'application pour transférer des données entre des bases de données relationnelles et Hadoop.

2.7 Tableau

Tableau Software est une société de logiciel américaine dont le siège se trouve à Seattle. Elle conçoit une famille de produits orientés visualisation de données.

3. Conclusion

Dans ce chapitre, on a défini le frameworks et les composants de l'écosystème Hadoop utilisés au sein du projet.

Chapitre 3 : Réalisation

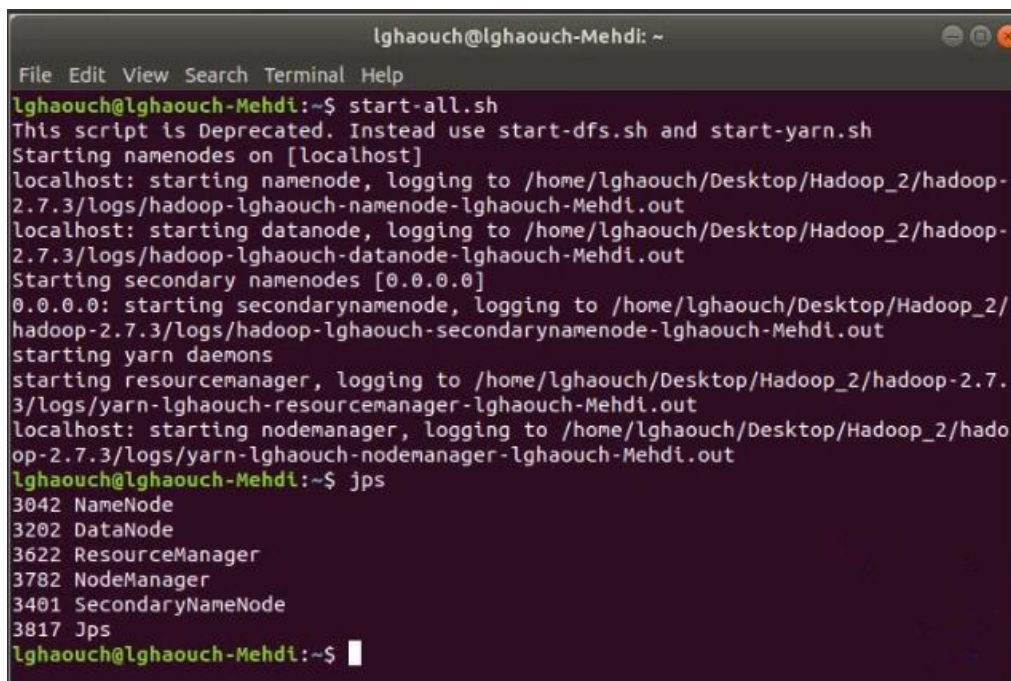
1. Introduction :

Dans ce chapitre on va présenter les étapes du projet par des captures avec les parties distinctes, la première partie est l'utilisation du Nifi, la deuxième est Hive et la troisième est HBase.

Le démarrage du Hadoop :

Pour démarrer Hadoop on utilise les commandes '*start-dfs.sh*' et '*start-yarn.sh*' ou la commande '*start-all.sh*'

Pour confirmer le démarrage on utilise la commande '*jps*' pour afficher les processus en cours d'exécution.

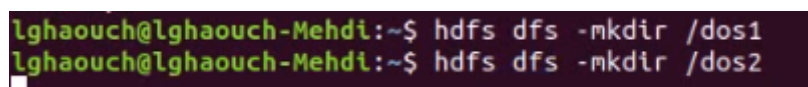
A terminal window titled 'lghaouch@lghaouch-Mehdi: ~' showing the execution of 'start-all.sh' and 'jps'. The script starts namenodes and datanodes on localhost, then starts secondary namenodes on 0.0.0.0, and finally starts yarn daemons (resourcemanager and nodemanager). The 'jps' command lists the running processes: NameNode, DataNode, ResourceManager, NodeManager, SecondaryNameNode, and Jps.

```
lghaouch@lghaouch-Mehdi:~$ start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
Starting namenodes on [localhost]
localhost: starting namenode, logging to /home/lghaouch/Desktop/Hadoop_2/hadoop-2.7.3/logs/hadoop-lghaouch-namenode-lghaouch-Mehdi.out
localhost: starting datanode, logging to /home/lghaouch/Desktop/Hadoop_2/hadoop-2.7.3/logs/hadoop-lghaouch-datanode-lghaouch-Mehdi.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /home/lghaouch/Desktop/Hadoop_2/hadoop-2.7.3/logs/hadoop-lghaouch-secondarynamenode-lghaouch-Mehdi.out
starting yarn daemons
starting resourcemanager, logging to /home/lghaouch/Desktop/Hadoop_2/hadoop-2.7.3/logs/yarn-lghaouch-resourcemanager-lghaouch-Mehdi.out
localhost: starting nodemanager, logging to /home/lghaouch/Desktop/Hadoop_2/hadoop-2.7.3/logs/yarn-lghaouch-nodemanager-lghaouch-Mehdi.out
lghaouch@lghaouch-Mehdi:~$ jps
3042 NameNode
3202 DataNode
3622 ResourceManager
3782 NodeManager
3401 SecondaryNameNode
3817 Jps
lghaouch@lghaouch-Mehdi:~$
```

2. Partie 1 : Utilisation du Nifi

Avant d'utiliser Nifi on utilise la commande '*copyFromLocal*' qui permet de copier les fichiers du Local au HDFS.

Pour appliquer cette commande et le processus du Nifi on va créer deux dossiers dans HDFS pour contenir les fichiers des données, pour cela on tape la commande '*hdfs dfs -mkdir*'

A terminal window showing two consecutive 'hdfs dfs -mkdir' commands being executed to create directories 'dos1' and 'dos2' in HDFS.

```
lghaouch@lghaouch-Mehdi:~$ hdfs dfs -mkdir /dos1
lghaouch@lghaouch-Mehdi:~$ hdfs dfs -mkdir /dos2
```


On confirme la création des dossiers par la commande *'hdfs dfs -ls'*

```
lghaouch@lghaouch-Mehdi:~$ hdfs dfs -mkdir /dos2
lghaouch@lghaouch-Mehdi:~$ hdfs dfs -ls /
Found 14 items
drwxr-xr-x - lghaouch supergroup          0 2021-12-20 13:21 /apps
drwxr-xr-x - lghaouch supergroup          0 2022-01-10 10:52 /dos1
drwxr-xr-x - lghaouch supergroup          0 2022-01-10 10:52 /dos2
drwxr-xr-x - lghaouch supergroup          0 2021-12-29 15:29 /hbase
-rw-r--r-- 1 lghaouch supergroup        349 2021-12-17 21:35 /lghaouch
drwxr-xr-x - lghaouch supergroup          0 2022-01-01 15:16 /med
drwxr-xr-x - lghaouch supergroup          0 2022-01-01 13:52 /myimport
drwxr-xr-x - lghaouch supergroup          0 2022-01-01 16:50 /nifi_recev
-rw-r--r-- 1 lghaouch supergroup       1669 2021-11-17 23:40 /poeme.txt
drwxr-xr-x - lghaouch supergroup          0 2021-11-17 23:41 /results
drwxr-xr-x - lghaouch supergroup          0 2021-12-07 19:21 /system
drwxr-xr-x - lghaouch supergroup          0 2021-12-15 17:08 /tablesExternes
drwx----- lghaouch supergroup          0 2021-12-15 15:59 /tmp
drwxr-xr-x - lghaouch supergroup          0 2021-12-15 16:35 /user
lghaouch@lghaouch-Mehdi:~$
```

Après on copie le fichier de données csv dans HDFS premièrement par la commande au-dessus :

```
lghaouch@lghaouch-Mehdi:~$ hdfs dfs -copyFromLocal /home/lghaouch/Desktop/Projet
Final/US_Accidents_Dec20_updated.csv /dos1
lghaouch@lghaouch-Mehdi:~$ hdfs dfs -ls /dos1
Found 1 items
-rw-r--r-- 1 lghaouch supergroup 419971286 2022-01-10 11:03 /dos1/US_Accident
s_Dec20_updated.csv
lghaouch@lghaouch-Mehdi:~$
```

Deuxièmement, on utilise Nifi pour la copie.

- On démarre Nifi par la commande *'./nifi.sh start'* puis on affiche les processus en cours d'exécution par la commande *'jps'*

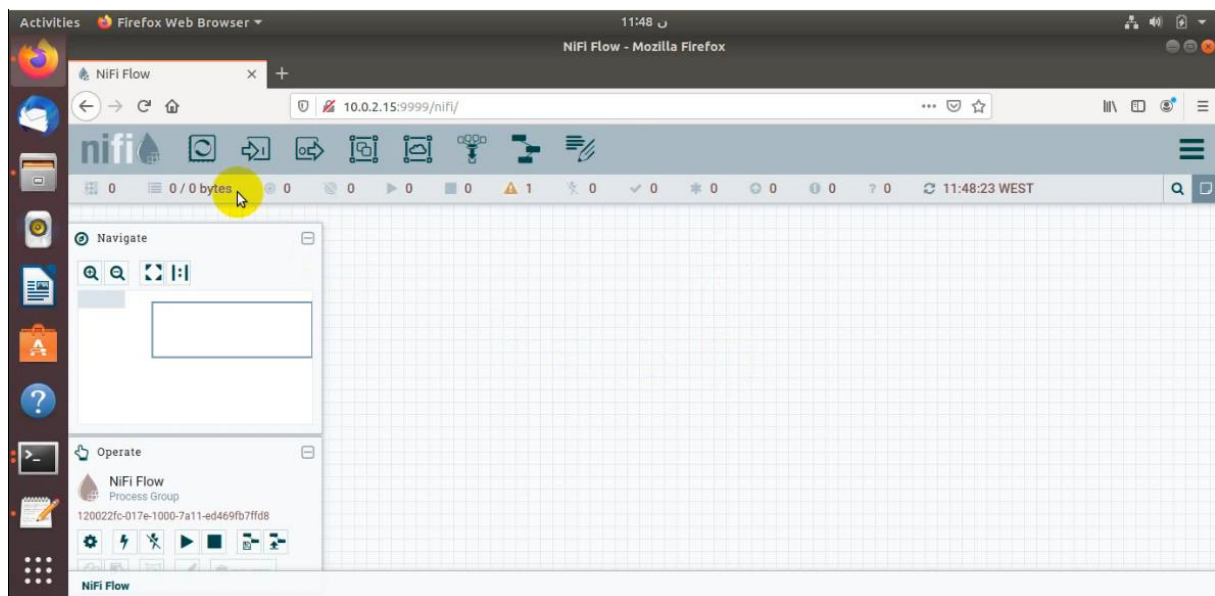
```
lghaouch@lghaouch-Mehdi:/usr/local/nifi/nifi-1.9.0/bin$ ./nifi.sh start

Java home: /usr/lib/jvm/java-8-openjdk-amd64
NiFi home: /usr/local/nifi/nifi-1.9.0

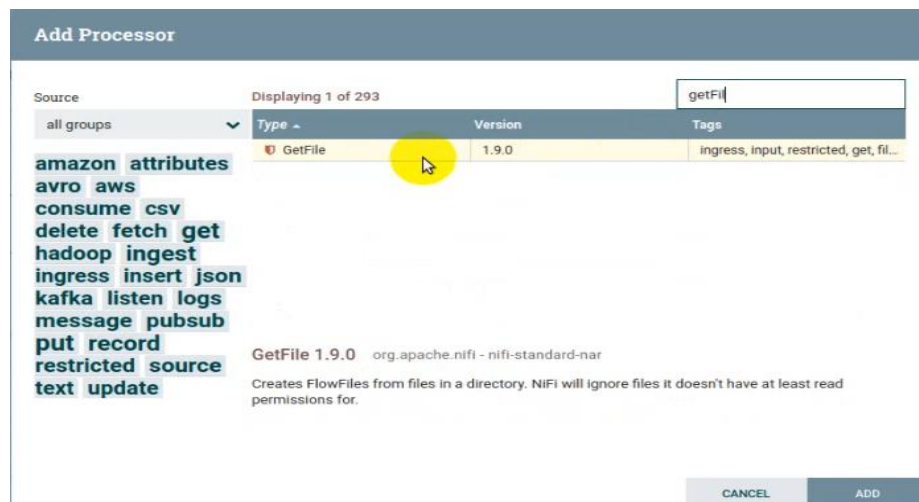
Bootstrap Config File: /usr/local/nifi/nifi-1.9.0/conf/bootstrap.conf

lghaouch@lghaouch-Mehdi:/usr/local/nifi/nifi-1.9.0/bin$ jps
3042 NameNode
3202 DataNode
3622 ResourceManager
3782 NodeManager
3401 SecondaryNameNode
4891 RunNiFi
4907 Jps
lghaouch@lghaouch-Mehdi:/usr/local/nifi/nifi-1.9.0/bin$
```

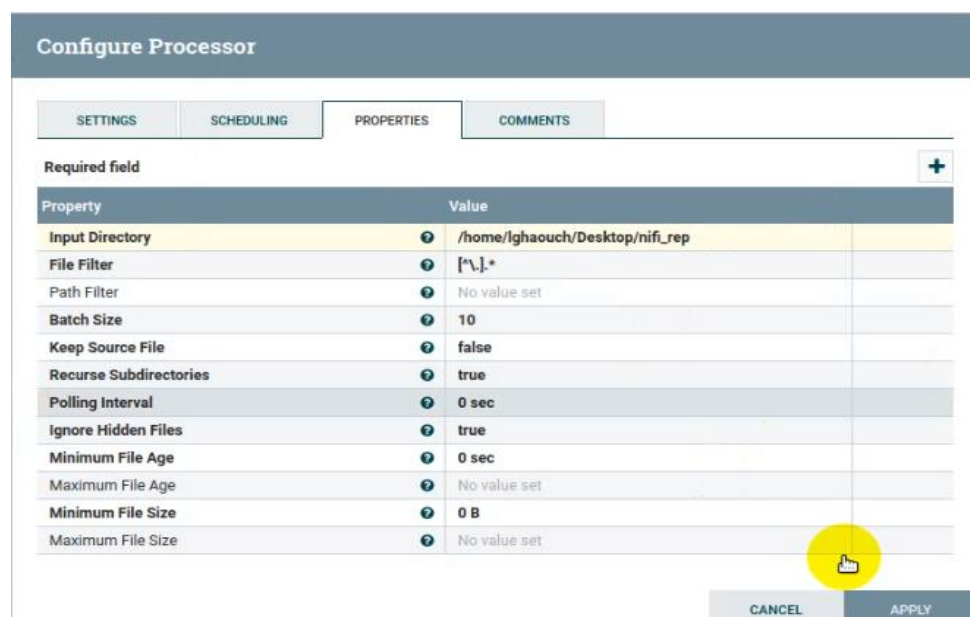
- On va sur le web de Nifi : 10.0.2.15:9999



- L'ajout du process '*GetFile*'

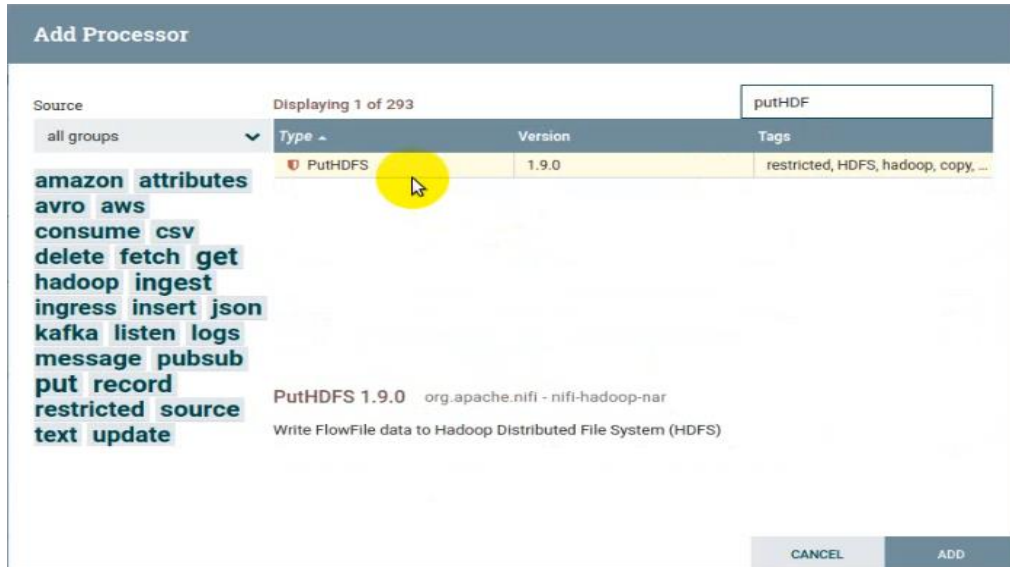


- La configuration du process '*GetFile*'

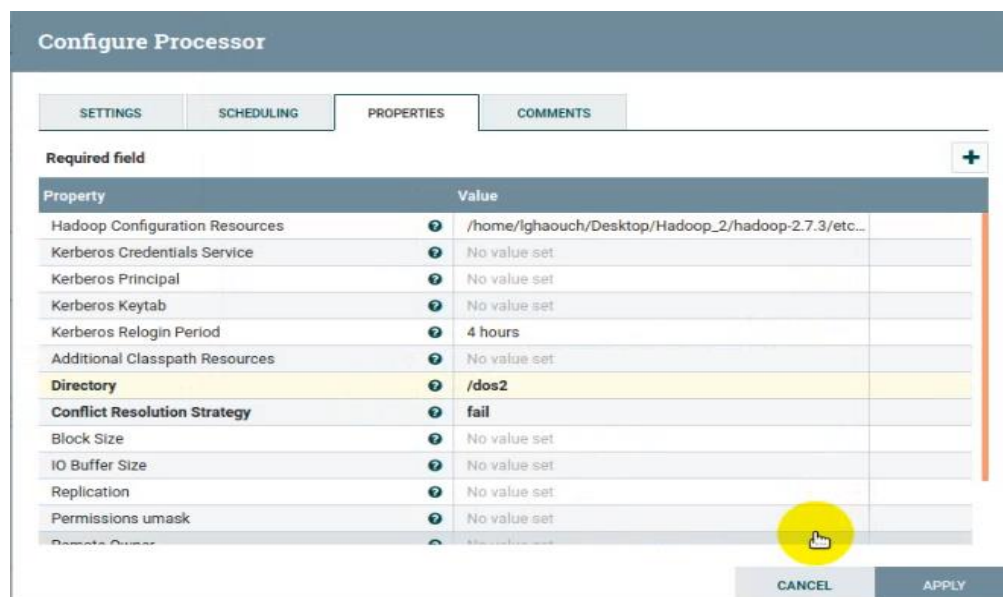


⇒ Dans **‘input directory’** on ajoute le chemin du répertoire qui contient le fichier avant le transfert : **‘/home/lghaouch/Desktop/nifi_rep’**

- L’ajout du process **‘PutHDFS’**



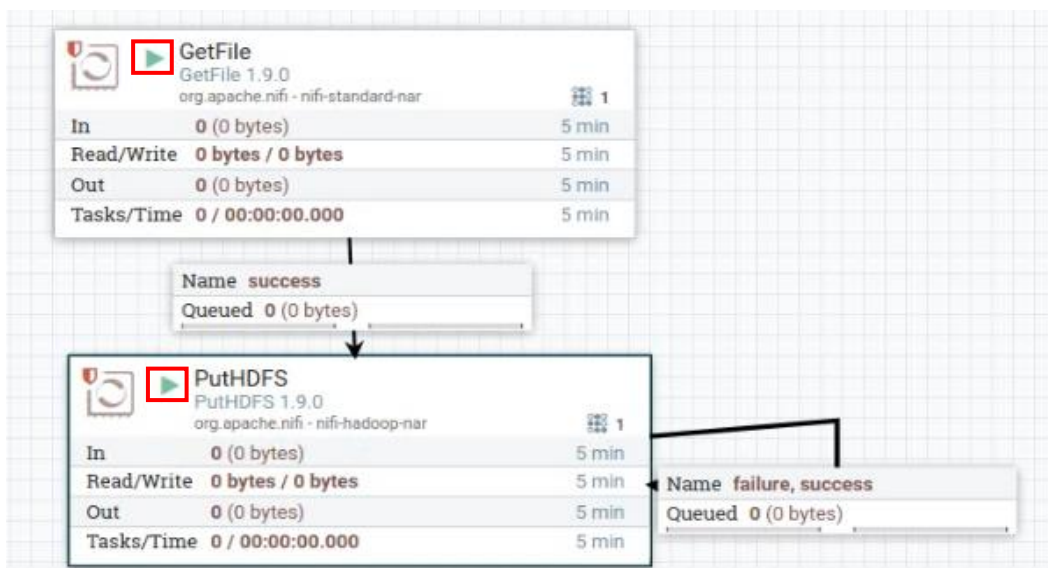
- La configuration du process **‘PutHDFS’**



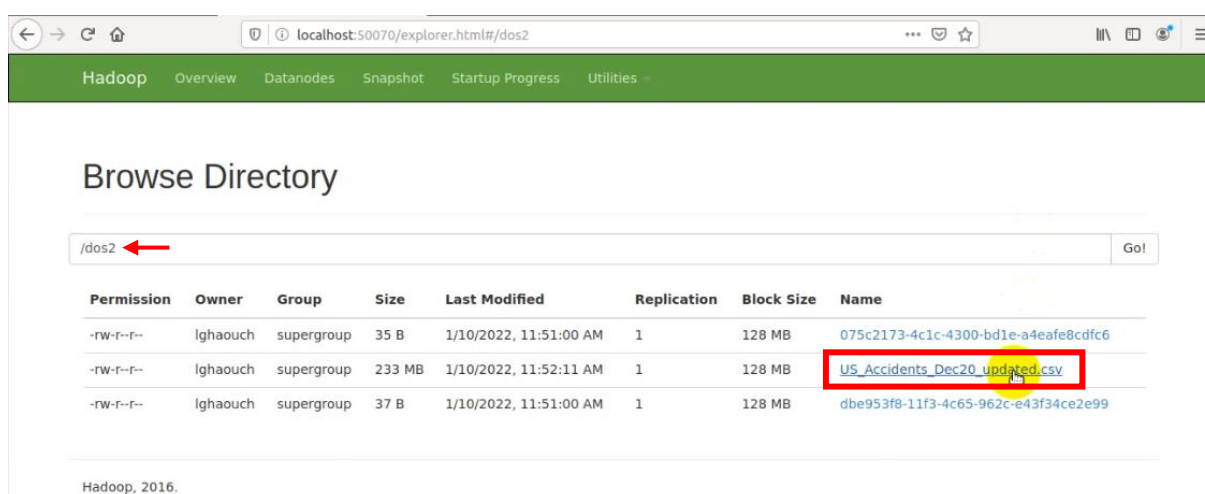
⇒ Dans **‘Hadoop Configuration Resources’** on ajoute le chemin de fichiers suivants **‘hdfs-site.xml’** et **‘core-site.xml’**

⇒ Dans **Directory** on ajoute le chemin du dossier dans HDFS qui reçoit le fichier : **‘/dos2’**

- Le démarrage du **‘GetFile’** et **‘PutHDFS’**



- Le fichier csv existe dans HDFS :



3. Partie 2 : Hive

Le démarrage du Hive :

On démarre Hive par la commande '*hive*'

```
lgbaouch@lgbaouch-Mehdi: ~
File Edit View Search Terminal Help
lgbaouch@lgbaouch-Mehdi:~$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/lgbaouch/hive/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/lgbaouch/Desktop/Hadoop_2/hadoop-2.7.3/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]

Logging initialized using configuration in jar:file:/home/lgbaouch/hive/lib/hive-common-2.3.9.jar!/hive-log4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
hive>
```


5.1 Création de la base de données 'ProjetF' dans Hive

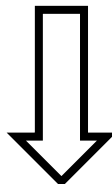
On crée la base de données '**ProjetF**' par la commande au-dessous, on affiche les bases de données existantes et on utilise la base de données '**ProjetF**'.

```
hive> create database if not exists projetF;
OK
Time taken: 24.708 seconds
hive> show databases;
OK
company
default
masterbigdata
projetf
Time taken: 0.637 seconds, Fetched: 4 row(s)
hive> use projetF;
OK
Time taken: 0.109 seconds
hive>
```

5.2 Création de la table 'use_acc' dans 'ProjetF'

- On crée la table '**use_acc**' par la commande au-dessous.

```
hive> create table use_acc (
  > id varchar(100),
  > severity int,
  > start_time timestamp,
  > end_time timestamp,
  > start_lat float,
  > start_lng float,
  > end_lat float,
  > end_lng float,
  > distance float,
  > description varchar(200),
  > number double,
  > street varchar(100),
  > side varchar(100),
  > city varchar(100),
  > county varchar(100),
  > state varchar(100),
  > zipcode varchar(100),
  > country varchar(10),
  > time_zone varchar(100),
  > airport_code varchar(100),
  > weather_timestamp timestamp,
  > temperature float,
  > wind_chill float,
  > humidity float,
  > pressure float,
  > visibility float,
  > wind_direction varchar(50),
  > wind_speed float,
  > precipitation float,
  > weather_condition varchar(100),
  > ...
);
```



```

> weather_condition varchar(100),
> amenity boolean,
> bump boolean,
> crossing boolean,
> give_way boolean,
> junction boolean,
> no_exit boolean,
> railway boolean,
> roundabout boolean,
> station boolean,
> stop boolean,
> traffic_calming boolean,
> traffic_signal boolean,
> turning_loop boolean,
> sunrise_sunset varchar(50),
> civil_twilight varchar(50),
> nautical_twilight varchar(50),
> astronomical_twilight varchar(50))
> row format delimited fields terminated by ','
> lines terminated by '\n'
> stored as textfile;
OK
Time taken: 4.034 seconds
hive>

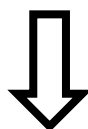
```

- L'affichage des colonnes avec la commande 'desc' :

```

hive> show tables;
OK
use_acc
Time taken: 1.123 seconds, Fetched: 1 row(s)
hive> desc use_acc;
OK
id                varchar(100)
severity          int
start_time        timestamp
end_time          timestamp
start_lat         float
start_lng         float
end_lat           float
end_lng           float
distance          float
description        varchar(200)
number            double
street            varchar(100)
side              varchar(100)
city              varchar(100)
county            varchar(100)
state             varchar(100)
zipcode           varchar(100)
country           varchar(10)
time_zone         varchar(100)
airport_code      varchar(100)
weather_timestamp timestamp
temperature       float
wind_chill        float
humidity          float
pressure          float
visibility        float
wind_direction    varchar(50)
wind_speed        float
precipitation     float
weather_condition varchar(100)
amenity           boolean
bump              boolean
crossing          boolean
give_way          boolean
junction          boolean
no_exit           boolean
railway           boolean

```



```

railway          boolean
roundabout      boolean
station         boolean
stop            boolean
traffic_calming  boolean
traffic_signal   boolean
turning_loop     boolean
sunrise_sunset  varchar(50)
civil_twilight   varchar(50)
nautical_twilight varchar(50)
astronomical_twilight varchar(50)
Time taken: 1.706 seconds, Fetched: 47 row(s)
hive>

```

5.3 Chargement des données dans 'use_acc' à partir du HDFS

- Le chargement des données dans hive par la commande au-dessous.

```

hive> load data inpath '/dos1/US_Accidents_Dec20_updated.csv' into table use_acc;
Loading data to table projetf.use_acc
OK
Time taken: 8.112 seconds

```

- L'affichage des données par la commande de sélection :

```

hive> select * from use_acc;
OK

```

- Le résultat de la commande de sélection :

```

;;;;;;;;;;;;;
A-3723323      2      2020-02-12 16:11:00      2020-02-12 17:24:06      44.76812      -
93.9667 44.76812      -93.9667      0.0      At CR-31 - Accident.      17279.0      Hig
hway 212      R      Norwood Young America      Carver      MN      55368-9673      US U
S/Central      KGYL      2020-02-12 15:59:00      19.0      5.0      54.0      28.88      10.0
N      16.0      0.0      Cloudy      false      false      false      false      false      fals
e      false      false      false      false      false      false      Day      Day      Day      Day;
;;;;;;;;;;;;;
A-3723324      2      2020-02-12 16:17:00      2020-02-12 17:31:05      44.67953      -
93.57172      44.67953      -93.57172      0.0      At MN-21/Broadway St - Vehic
le spun around. 2370.0      Country Trl W R      Jordan      Scott      MN      55352-9343 U
S      US/Central      KFCM      2020-02-12 15:53:00      24.0      9.0      65.0      28.9
2      10.0      NNW      21.0      0.0      Cloudy / Windy      false      false      false      fals
e      false      false      false      false      false      false      DayD
ay      Day      Day;;;;;;;;;;;;;
A-3723325      2      2020-02-12 16:18:00      2020-02-12 17:31:05      45.81258      -
94.09394      45.81258      -94.09394      0.0      At CR-40/CR-22/165th St NE -
Accident.      16607.0      Highway 25 NE L      Rice      Benton      MN      56367-9760 U
S      US/Central      KLXL      2020-02-12 16:15:00      9.0      -6.0      40.0      28.8
10.0      N      10.0      0.0      Fair      false      false      false      false      false      fals
e      false      false      false      false      false      false      Day      Day      DayD
ay;;;;;;;;;;;;;
Time taken: 7.487 seconds, Fetched: 1000000 row(s)
hive>

```

4. Partie 3 : Hbase

Le démarrage du HBase :

Pour démarrer HBase on utilise la commande '*start-hbase.sh*' et on affiche les processus existants par '*jps*'


```

lghaouch@lghaouch-Mehdi:~$ start-hbase.sh
localhost: starting zookeeper, logging to /usr/local/hbase/bin/./logs/hbase-lghaouch-zookeeper-lghaouch-Mehdi.out
starting master, logging to /usr/local/hbase/logs/hbase-lghaouch-master-lghaouch-Mehdi.out
starting regionserver, logging to /usr/local/hbase/logs/hbase-lghaouch-1-regionserver-lghaouch-Mehdi.out
lghaouch@lghaouch-Mehdi:~$ jps
2818 ResourceManager
2979 NodeManager
3605 HMaster
2405 DataNode
3701 HRegionServer
2613 SecondaryNameNode
3542 HQuorumPeer
2249 NameNode
3820 Jps
lghaouch@lghaouch-Mehdi:~$

```

⇒ Création de la table **Hbase** 'hbase_table_usacc' à partir d'un script **HiveQL** avec le nom 'us_acc_hbase' dans Hbase, avec la famille de colonnes 'accident' et les mêmes noms des colonnes du table **Hive** 'use_acc'.

```

hive> CREATE TABLE hbase_table_usacc (
  > id varchar(100),
  > severity int,
  > start_time timestamp,
  > end_time timestamp,
  > start_lat float,

```



```

  > sunrise_sunset varchar(50),
  > civil_twilight varchar(50),
  > nautical_twilight varchar(50),
  > astronomical_twilight varchar(50))
  > STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
  > WITH SERDEPROPERTIES ("hbase.columns.mapping" = ":key,accident:severity,accident:start_time ,accident:end_time ,accident:start_lat,
    > accident:start_lng,accident:end_lat,accident:end_lng,accident:distance,accident:description,accident:number,accident:street,accident:side,
    > accident:city,accident:county,accident:state,accident:zipcode,accident:country,accident:time_zone,accident:airport_code,
    > accident:weather_timestamp,accident:temperature,accident:wind_chill,accident:humidity,accident:pressure,accident:visibility,
    > accident:wind_direction,accident:wind_speed,accident:precipitation,accident:weather_condition,accident:amenity, accident:bump,accident:crossing,accident:give_way,accident:junction,accident:no_exit,accident:railway,accident:roundabout,accident:station, accident:stop,accident:traffic_calming,accident:traffic_signal,accident:turning_loop,accident:sunrise_sunset ,
    > accident:civil_twilight ,accident:nautical_twilight ,accident:astronomical_twilight")
  > TBLPROPERTIES ("hbase.table.name" = "us_acc_hbase");
OK
Time taken: 13.077 seconds
hive>

```

⇒ Insérer des données dans le tableau Hbase : INSERT INTO TABLE
hbase_table_usacc SELECT * FROM accident

```
hive> INSERT INTO TABLE hbase_table_usacc SELECT * FROM use_acc;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = lghaouch_20220117133519_ee102776-f1c2-47bf-83b4-46671d9e29c0
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1642420752056_0001, Tracking URL = http://lghaouch-Mehdi:8088/proxy/application_1642420752056_0001/
Kill Command = /home/lghaouch/Desktop/Hadoop_2/hadoop-2.7.3/bin/hadoop job -kill job_1642420752056_0001
Hadoop job information for Stage-3: number of mappers: 2; number of reducers: 0
2022-01-17 13:36:35,845 Stage-3 map = 0%, reduce = 0%
2022-01-17 13:37:36,684 Stage-3 map = 0%, reduce = 0%, Cumulative CPU 17.52 sec
```



```
2022-01-17 15:09:03,776 Stage-3 map = 75%, reduce = 0%, Cumulative CPU 2750.26 sec
2022-01-17 15:10:04,144 Stage-3 map = 75%, reduce = 0%, Cumulative CPU 2769.4 sec
2022-01-17 15:11:05,435 Stage-3 map = 75%, reduce = 0%, Cumulative CPU 2787.71 sec
2022-01-17 15:12:06,179 Stage-3 map = 75%, reduce = 0%, Cumulative CPU 2805.79 sec
2022-01-17 15:13:06,405 Stage-3 map = 75%, reduce = 0%, Cumulative CPU 2821.82 sec
2022-01-17 15:14:03,262 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 2839.24 sec
MapReduce Total cumulative CPU time: 47 minutes 19 seconds 340 msec
Ended Job = job_1642420752056_0001
MapReduce Jobs Launched:
Stage-Stage-3: Map: 2 Cumulative CPU: 2839.34 sec HDFS Read: 420026320 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 47 minutes 19 seconds 340 msec
OK
Time taken: 5931.984 seconds
hive>
```

⇒ Lancer le Shell du Hbase :

```
lghaouch@lghaouch-Mehdi:~$ hbase shell
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hbase/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/lghaouch/Desktop/Hadoop_2/hadoop-2.7.3/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.5, rd7b05f79dee10e0ada614765bb354b93d615a157, Wed Mar 1 00:34:48 CST 2017

hbase(main):001:0>
```


⇒ Afficher les données dans la table Hbase : **scan 'us_acc_hbase'**

```
hbase(main):001:0> scan 'us_acc_hbase'
ROW                                COLUMN+CELL
"A-2983732"                        column=accident:airport_code, timestamp=1642425688181, value=KF70
"A-2983732"                        column=accident:amenity, timestamp=1642425688181, value=false
"A-2983732"                        column=accident:astronomical_twilight, timestamp=1642425688181, value=Night";;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
"A-2983732"                        column=accident:bump, timestamp=1642425688181, value=false
"A-2983732"                        column=accident:city, timestamp=1642425688181, value=Murrieta
```

5. Partie 4 : MySQL et Sqoop

Le démarrage du MySQL :

5.1 MySQL :

Pour démarrer MySQL on utilise la commande **'mysql -u root -p'** puis on tape le mot de pass.

```
lghaouch@lghaouch-Mehdi:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.7.36-0ubuntu0.18.04.1 (Ubuntu)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

⇒ La création de la base de données **'mysql_us_db'** dans MySQL et l'utilisation de cette base de données.

```
mysql> create database mysql_us_db;
Query OK, 1 row affected (0.00 sec)

mysql> use mysql_us_db;
Database changed
```

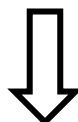
⇒ La création de la table **'mysql_table_usacc'** dans la base de données **'mysql_us_db'**.

```
mysql> create table mysql_table_usacc (
-> id varchar(100),
-> severity int,
-> start_time timestamp,
-> end_time timestamp,
-> start_lat float,
-> start_lng float,
-> end_lat float,
-> end_lng float,
-> distance float,
-> description varchar(200),
-> number double,
-> street varchar(100),
-> side varchar(100),
-> city varchar(100),
-> county varchar(100),
-> state varchar(100),
-> zipcode varchar(100),
-> country varchar(10),
-> time_zone varchar(100),
-> airport_code varchar(100),
-> weather_timestamp timestamp,
-> temperature float,
-> wind_chill float,
-> humidity float,
-> pressure float,
-> visibility float,
-> wind_direction varchar(50),
-> wind_speed float,
-> precipitation float,
-> weather_condition varchar(100),
-> amenity boolean,
-> bump boolean,
-> crossing boolean,
-> give_way boolean,
-> junction boolean,
-> no_exit boolean,
-> railway boolean,
-> roundabout boolean,
-> station boolean,
-> stop boolean,
-> traffic_calming boolean,
-> traffic_signal boolean,
-> turning_loop boolean,
-> sunrise_sunset varchar(50),
-> civil_twilight varchar(50),
-> nautical_twilight varchar(50),
-> astronomical_twilight varchar(50));
Query OK, 0 rows affected (0.08 sec)

mysql>
```

⇒ Afficher les colonnes de la table ‘mysql_table_usacc’ avec la commande ‘describe’

```
mysql> describe mysql_table_usacc;
```



station	tinyint(1)	YES		NULL
stop	tinyint(1)	YES		NULL
traffic_calming	tinyint(1)	YES		NULL
traffic_signal	tinyint(1)	YES		NULL
turning_loop	tinyint(1)	YES		NULL
sunrise_sunset	varchar(50)	YES		NULL
civil_twilight	varchar(50)	YES		NULL
nautical_twilight	varchar(50)	YES		NULL
astronomical_twilight	varchar(50)	YES		NULL

```

+-----+-----+-----+-----+
47 rows in set (0.00 sec)

mysql>

```

⇒ Le chargement des données du Local dans la table mysql

```

mysql> load data local infile '/home/lghaouch/Desktop/ProjetFinal/US_Accidents_D
ec20 updated.csv' into table mysql_table_usacc fields terminated by ',';
Query OK, 1000000 rows affected, 65535 warnings (5 min 45.26 sec)
Records: 1000000 Deleted: 0 Skipped: 0 Warnings: 14343225

mysql>

```

⇒ Afficher le nombre des lignes dans la table mysql

```

mysql> select count(*) from mysql_table_usacc;
+-----+
| count(*) |
+-----+
| 1000000 |
+-----+
1 row in set (24.77 sec)

mysql>

```

5.2 Sqoop

Avant l'utilisation de la commande sqoop on va au chemin '**usr/lib/sqoop/bin**'

- Importer la table '**mysql_table_usacc**' de la base de données mysql '**mysql_us_db**' vers hdfs :

```

lghaouch@lghaouch-Mehdi:~$ cd /usr/lib/sqoop/bin
lghaouch@lghaouch-Mehdi:/usr/lib/sqoop/bin$ sqoop import --connect "jdbc:mysql:/
localhost:3306/mysql_us_db?autoReconnect=true&useSSL=false" --username=root --p
assword= --table mysql_table_usacc --target-dir '/mysql_us_import' -m 1
Warning: /usr/lib/sqoop/./hcatalog does not exist! HCatalog jobs will fail.
Please set $HCAT_HOME to the root of your HCatalog installation.
Warning: /usr/lib/sqoop/./accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
Warning: /usr/lib/sqoop/./zookeeper does not exist! Accumulo imports will fail.
Please set $ZOOKEEPER_HOME to the root of your Zookeeper installation.

```

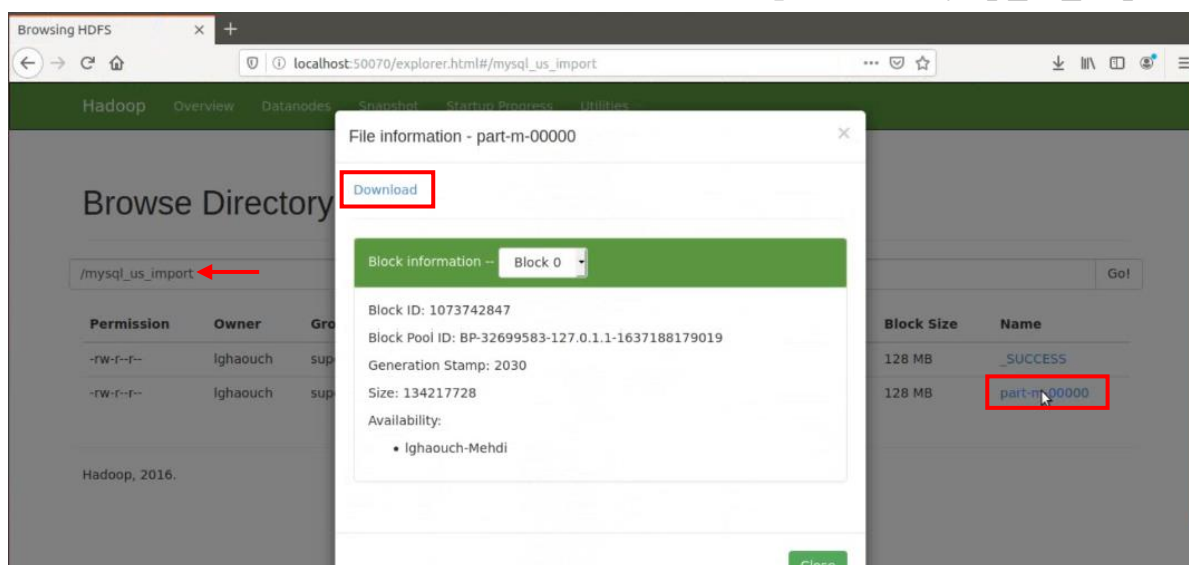


```

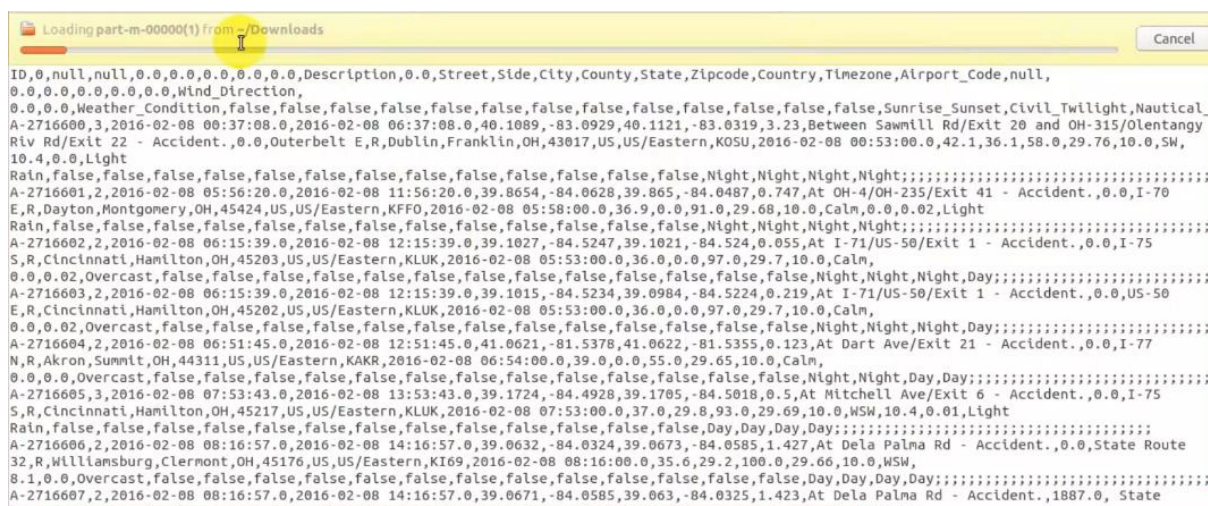
Total time spent by all reduces in occupied slots (ms)=0
Total time spent by all map tasks (ms)=249770
Total vcore-milliseconds taken by all map tasks=249770
Total megabyte-milliseconds taken by all map tasks=255764480
Map-Reduce Framework
  Map input records=1000000
  Map output records=1000000
  Input split bytes=87
  Spilled Records=0
  Failed Shuffles=0
  Merged Map outputs=0
  GC time elapsed (ms)=5508
  CPU time spent (ms)=74920
  Physical memory (bytes) snapshot=152711168
  Virtual memory (bytes) snapshot=1908293632
  Total committed heap usage (bytes)=62980096
File Input Format Counters
  Bytes Read=0
File Output Format Counters
  Bytes Written=422412770
22/01/25 12:22:00 INFO mapreduce.ImportJobBase: Transferred 402.8442 MB in 351.3
12 seconds (1.1467 MB/sec)
22/01/25 12:22:00 INFO mapreduce.ImportJobBase: Retrieved 1000000 records.
lghaouch@lghaouch-Mehdi: /usr/lib/sqoop/bin$

```

- Consulter le fichier dans HDFS au sein du répertoire ‘mysql_us_import’



- Télécharger le fichier pour confirmer le travail.



6. Conclusion

Dans ce chapitre on a vu la réalisation du projet en utilisant l'écosystème du Hadoop avec des captures d'écran qui montrent le travail.

Chapitre 4 : Visualisation des données

1. Introduction :

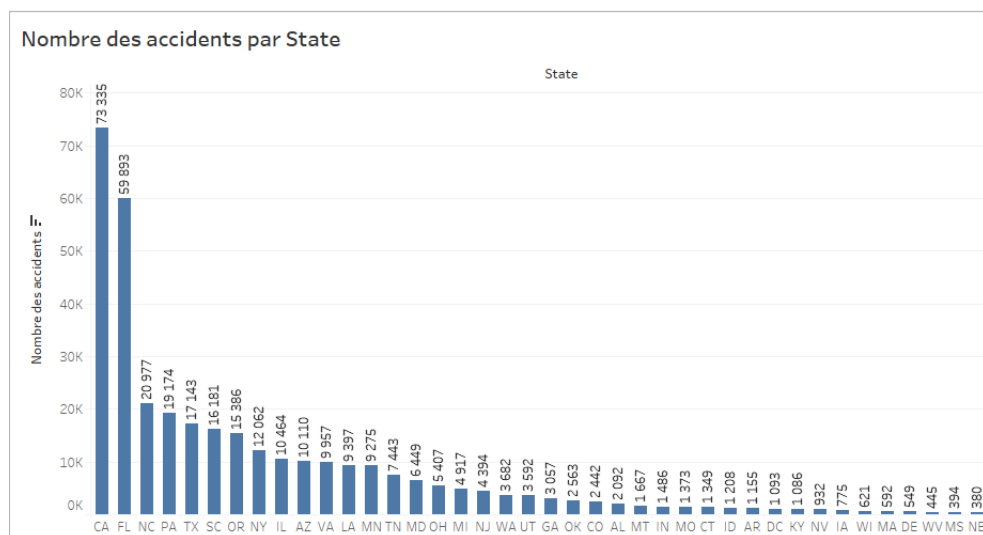
La visualisation de données désigne la représentation graphique d'informations et de données. À l'aide d'éléments visuels comme les graphiques et les cartes, une visualisation de données permet de voir et de comprendre des tendances ou des valeurs inhabituelles dans les données, de manière très accessible.

Dans ce chapitre on va analyser et visualiser les données par le logiciel 'Tableau'.

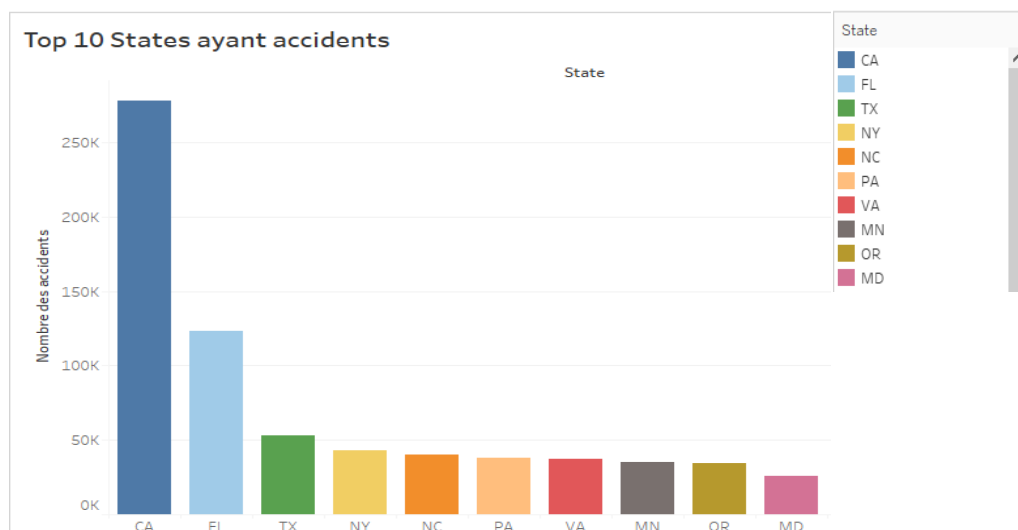
2. La visualisation :

2.1 Les accidents dans chaque Etat :

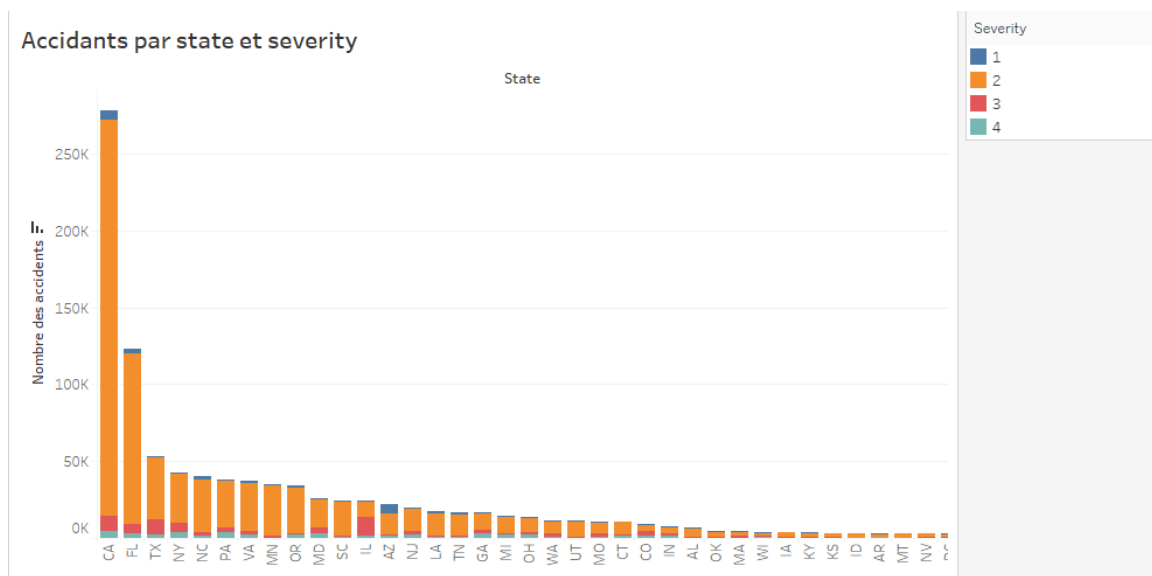
- Le nombre des accidents par état (State) :



- Top 10 des états (States) ayant des accidents :

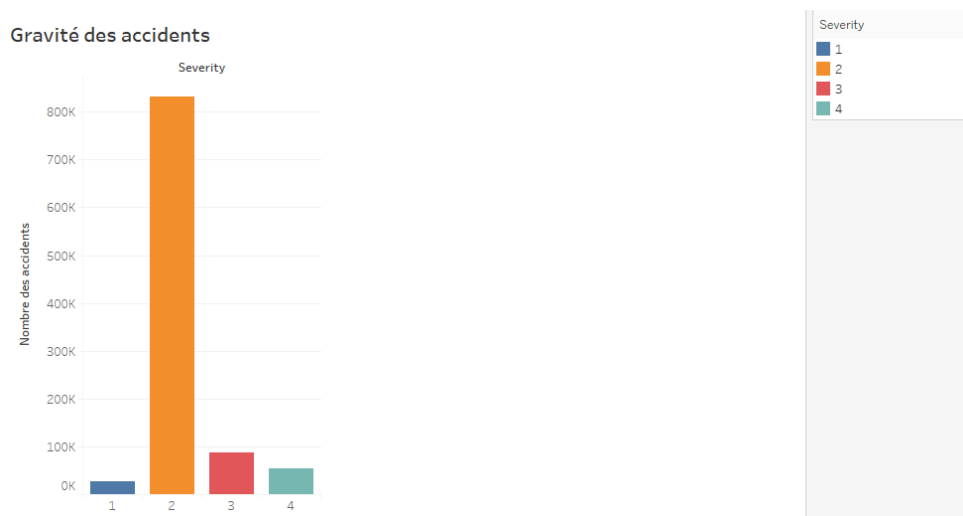


- Le nombre des accidents par état (State) et gravité (Severity) :

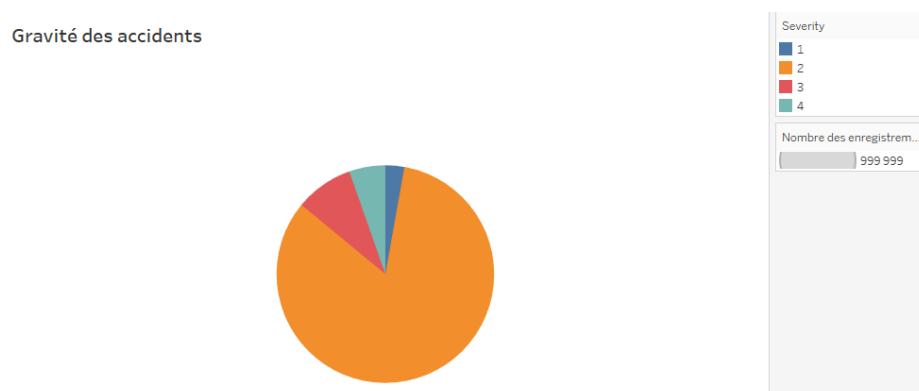


2.2 Gravité (Severity) des accidents :

- Gravité des accidents – diagramme en bâtons :

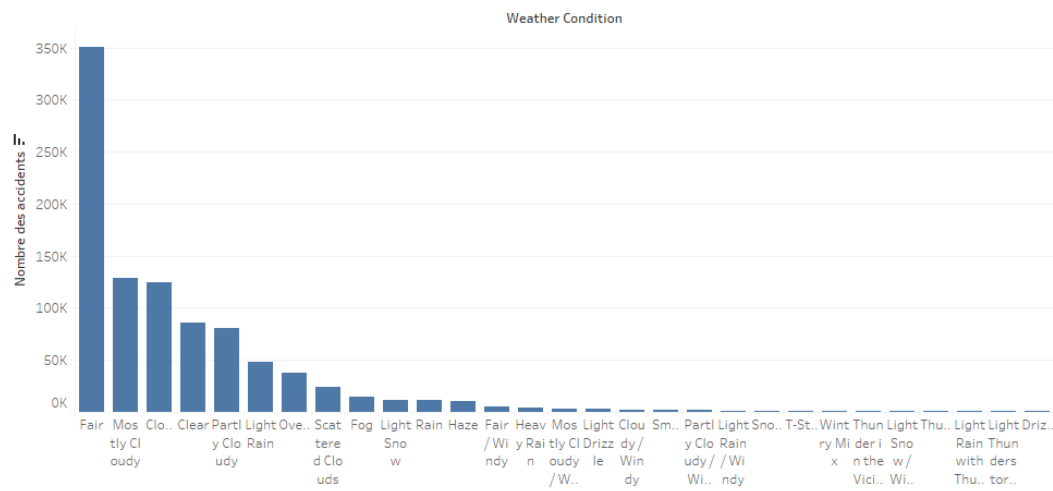


- Gravité des accidents – diagramme circulaire :



2.3 Les accidents par conditions météorologiques

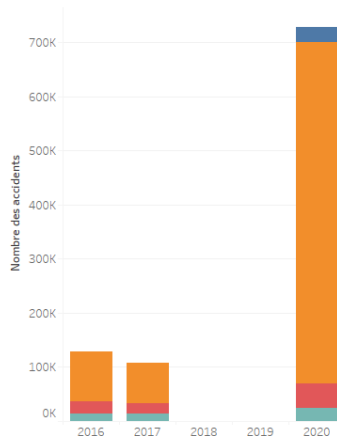
Accidents par Weather Condition



2.4 Les accidents par temps :

- Les accidents par année :

Accidents par année

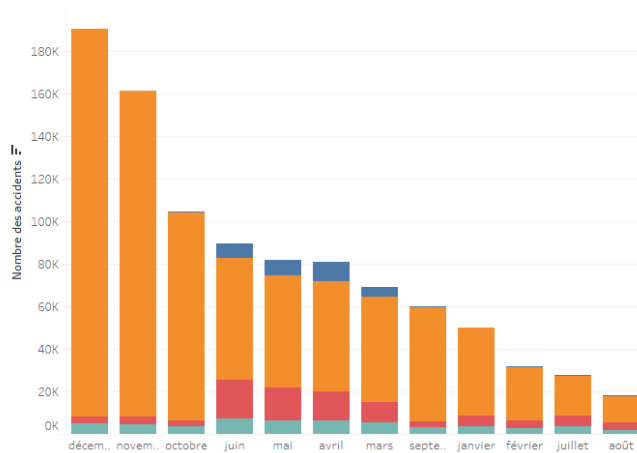


Severity



- Les accidents par mois :

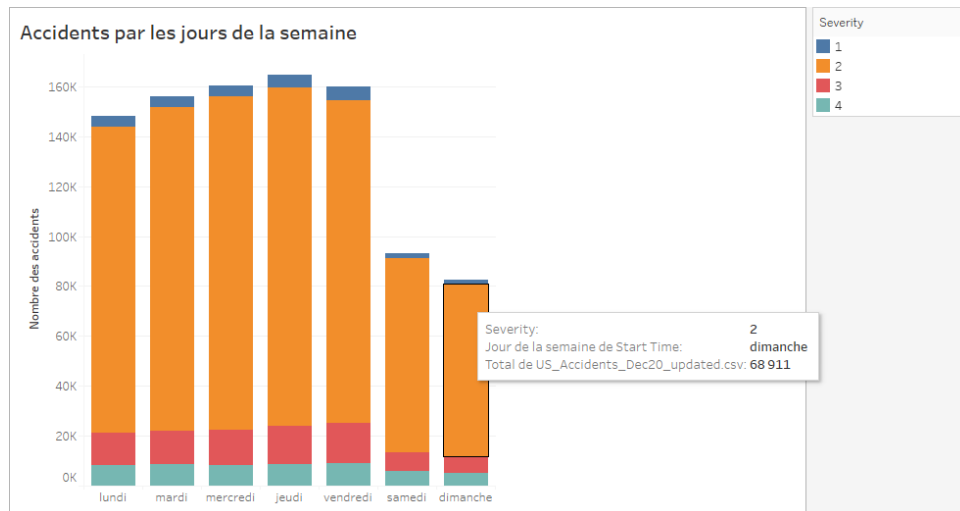
Accidents par mois



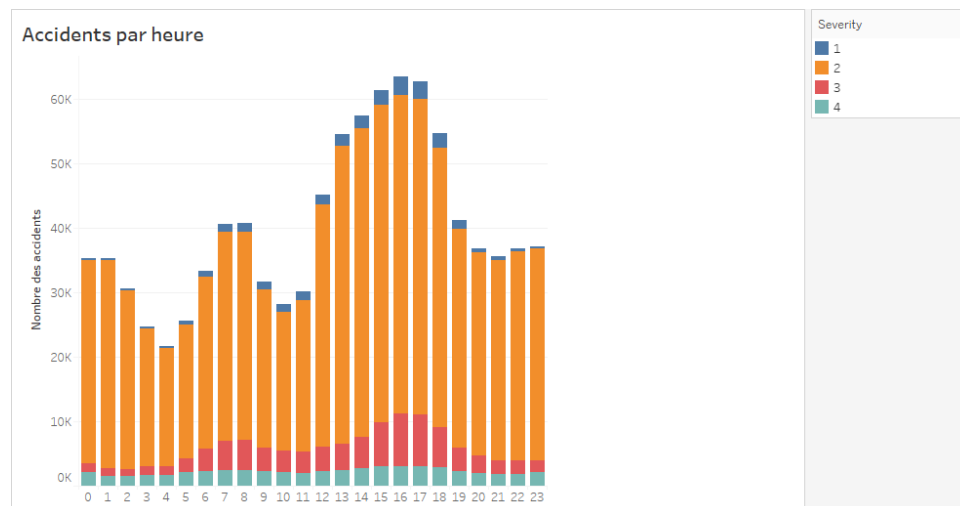
Severity



- Les accidents par jour :

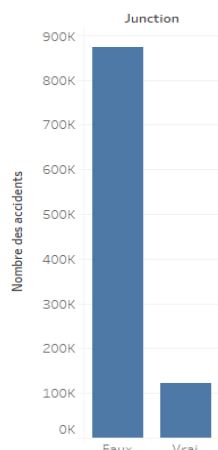


- Les accidents par heure :

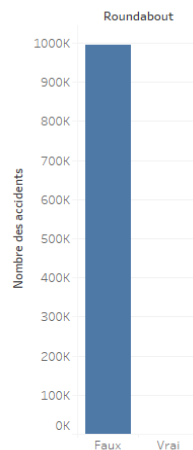


2.5 Les accidents par 'Junction', 'Bump', 'Roundabout', 'Crossing', 'Stop', 'Traffic Signal'

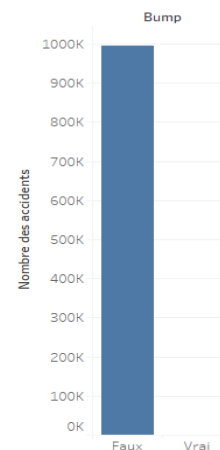
Accidents par Junction

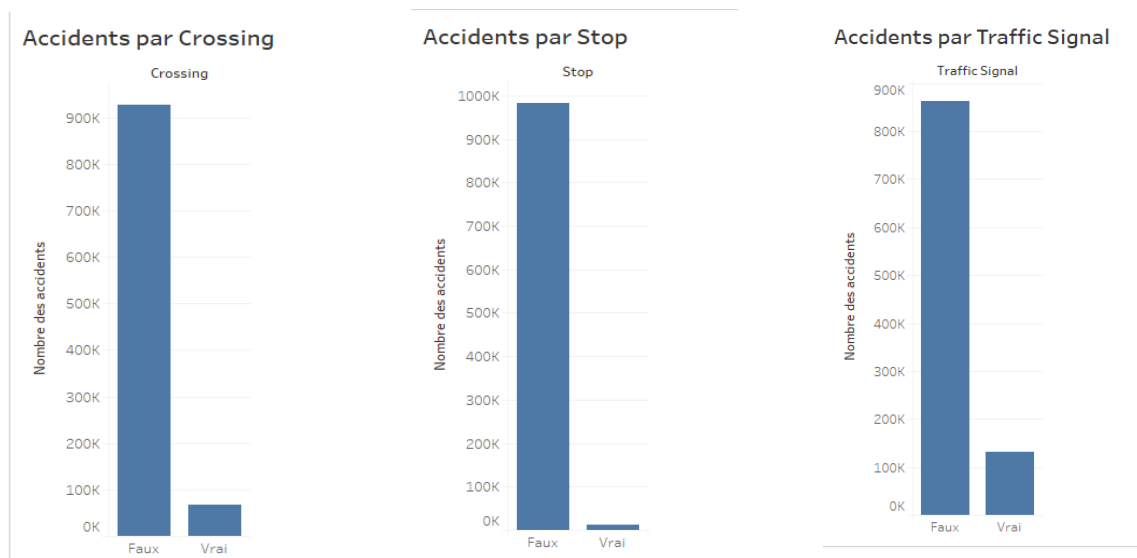


Accidents par Roundabout



Accidents par Bump





3. Conclusion :

Les captures d'écran affichent les résultats de la visualisation des données par le logiciel '**Tableau**' sous forme des statistiques en plusieurs types des diagrammes.

Conclusion Générale

Dans le cadre de ce projet, j'ai analysé un fichier de données sur les accidents dans les états unis américain. Pour ce faire j'ai travaillé avec le framework Hadoop et son écosystème ainsi le logiciel tableau pour la visualisation.

J'ai essayé tout au long de ce projet de construire les étapes incrément par incrément.

Sur le plan des acquis fonctionnels, ce projet m'a permis de renforcer mes compétences relationnelles, et de mesurer les différences, notables, entre le monde Universitaire et le monde professionnel.

Sur le plan technique, ce projet m'a donné l'occasion de mettre en pratiques mes connaissances théoriques au niveau du Big Data et les relations entre les bases de données relationnelles et non relationnelles.

Il est bien évident qu'une telle expérience ne pourra être qu'une bonne expérience. Dans laquelle j'ai pu renforcer mes connaissances, de développer ma confiance en soi, et de gérer les difficultés rencontrées.

Au cours de la réalisation du projet, j'ai astreint par quelques limites notamment, la contrainte du temps qui était relativement un obstacle devant l'ajout de certaines autres fonctionnalités. Cependant les tâches qui j'ai été assignées ont été accomplies dans l'ensemble.

Je suis par ailleurs convaincu que le travail élaboré ni qu'une étape primaire aussi bien pour une carrière professionnelle que pour des études plus approfondies.