



Rapport Framework Logiciel pour Big Data

*Analyse exhaustive sur les vols aux états-unis sur un jeu
de données de 2015*

Auteur
GHOUIBI Ghassen

Encadrée par
Mme Rakia JAZIRI

Table des matières

1 Introduction	2
I Architecture distribuée	2
II Amazon Web Services	3
II.1. Qu'est-ce qu'Amazon EMR ?	3
II.2. Qu'est-ce qu'Amazon EC2	4
II.3. Qu'est-ce qu'Amazon S3	4
III Déroulement du rapport	4
2 Initialisation	5
I Création du cluster	5
II Mettre les données dans S3	6
III Chargement de données dans le cluster	7
IV Initialisation des tables	8
3 Manipulation hive	11
I Questions à traiter dans le projet	11
II Décomposer les tables	11
II.1. La liste des compagnies aériennes	11
II.2. Division des vols	12
II.3. Opérations sur une compagnie	12
III Classement d'aéroport (aéroport de départ)	13
IV Affichage des données avec Python	16
4 Conclusion	18

Chapitre 1

Introduction

Depuis quelques années, nous constatons une révolution dans le monde grâce à l'automatisation de plusieurs domaines qui peuvent être dans le monde de travail ou un exemple de réseaux social en revanche dans les deux cas ces derniers génère énormément de données chaque seconde surtout qu'on trouve plus ou stocker ces informations et on ne sait pas quoi faire avec la peur de les détruire si jamais une information importante qui est caché fait qu'il faut les exploiter.

Cette évolution à jouer un facteur non négligeables pour la croissance des données dans le monde avec les outils de l'époque on ne peut pas traiter un certain volume de données et avoir une visualisation plus propre qui permet de manipuler ces données.

Le sujet de ce document c'est se familiariser avec une architecture distribuée sur AWS en utilisant un jeu de données mon choix se porte un jeu de données des vols des compagnies aériennes en 2015.

“Information is the oil of the 21st century, and analytics is the combustion engine ”

I Architecture distribuée

L'architecture distribuée ou l'informatique distribuée désigne un système d'information ou un réseau pour lequel l'ensemble des ressources disponibles ne se trouvent pas au même endroit ou sur la même machine. Ce concept, dont une version peut être une combinaison de transmissions du type client-serveur, s'oppose à celui d'informatique centralisée.

Internet est un exemple de réseau distribué puisqu'il ne possède aucun nœud central. Les architectures distribuées reposent sur la possibilité d'utiliser des objets qui s'exécutent sur des machines réparties sur le réseau et communiquent par messages au travers du réseau.

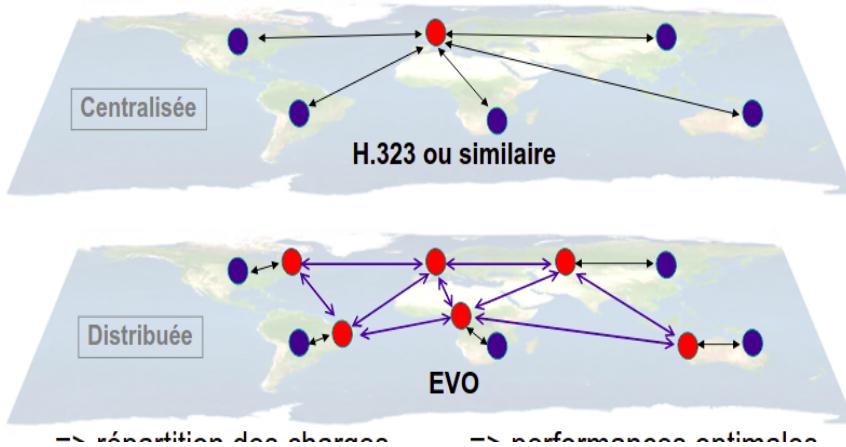


FIGURE 1.1 – La différence entre un architecture distribuée et centralisée

II Amazon Web Services

Amazon Web Services (AWS) est une division du groupe américain de commerce électronique Amazon, spécialisée dans les services de cloud computing à la demande pour les entreprises et particuliers.

La technologie AWS est mise en place grâce à des fermes de serveurs distribuées à travers le monde et maintenues par AWS.

Les frais sont calculés à partir de l'usage, des caractéristiques matérielles, logicielles, réseau et du système d'exploitation choisis par le client, ainsi que des options de disponibilité, de redondance, de sécurité et de service.

Pour notre projet il faut choisir les services les mieux adapté pour notre analyse de jeu de données.

II.1 Qu'est-ce qu'Amazon EMR ?

Amazon EMR est une plate-forme de cluster gérée qui simplifie l'exécution des infrastructures de données massives, telles qu'Apache Hadoop et Apache Spark, sur AWS pour traiter et analyser de grandes quantités de données. Grâce à ces infrastructures et des projets open source connexes, tels qu'Apache Hive et Apache Pig, vous pouvez traiter des données à des fins d'analyse et pour des charges de travail d'aide à la décision.

En outre, vous pouvez utiliser Amazon EMR pour transformer et transférer de grandes quantités de données dans et hors d'autres bases de données et magasins de données AWS, tels que Amazon Simple Storage Service (Amazon S3) et Amazon DynamoDB.

II.2 Qu'est-ce qu'Amazon EC2

Amazon Elastic Compute Cloud ou EC2 est un service proposé par Amazon permettant à des tiers de louer des serveurs sur lesquels exécuter leurs propres applications web. EC2 permet un déploiement extensible des applications en fournissant une interface web par laquelle un client peut créer des machines virtuelles, c'est-à-dire des instances du serveur, sur lesquelles le client peut charger n'importe quel logiciel de son choix. Un client peut créer, lancer, et arrêter des instances de serveurs en fonction de ses besoins, et paye en fonction du temps d'usage des serveurs, d'où le terme d'« Élastique » (Elastic en anglais).

II.3 Qu'est-ce qu'Amazon S3

Amazon Simple Storage Service (Amazon S3) est un service de stockage d'objet offrant une scalabilité, une disponibilité des données, une sécurité et des performances de pointe.

Les clients de toutes tailles et de tous secteurs peuvent ainsi utiliser ce service afin de stocker et protéger n'importe quelle quantité de données pour un large éventail de cas d'utilisation comme des sites web, des applications mobiles, la sauvegarde et la restauration, l'archivage, des applications d'entreprise, des appareils IoT et des analyses du Big Data.

Amazon S3 fournit des fonctions de gestion faciles à utiliser pour vous permettre d'organiser vos données et de configurer des contrôles d'accès affinés pour vos exigences métier, d'organisation et de conformité spécifiques.

III Déroulement du rapport

Dans ce rapport on va suivre par étape de la création de notre architecture jusqu'au résultat sachant qu'une présentation sera accompagné avec ce rapport et le code utilisé.

Pour éviter la redondance et ne pas avoir un long rapport on va éviter les étapes simples.

Les étapes seront accompagnées d'une capture d'écran si besoin, finalement le choix des technologies suivantes : EC2, EMR, S3.

Dans le chapitre suivant on va s'affronter à la création de notre cluster ce chapitre est intitulé Phase d'initialisation.

Chapitre 2

Initialisation

Dans ce chapitre on va présenter les étapes à suivre pour la création d'une architecture distribuée.

I Crédation du cluster

Pour créer un cluster il faut aller dans la rubrique EMR sur AWS après cliquer sur le bouton créer un cluster vous aurez une page comme ci-dessous. Dans la configuration générale il faudrait préciser le nom de cluster dans notre cas c'est Flights-cluster, ensuite choisir la journalisation qui sera très importante pour suivre les logs de votre cluster.

L'étape de configuration des logiciels choisir la version de emr ensuite l'application dans notre cas on opte pour Core Hadoop: Hadoop 2.8.5 with Ganglia 3.7.2, Hive 2.3.6, Hue 4.4.0, Mahout 0.13.0, Pig 0.17.0, and Tez 0.9.2.

La configuration du matériel on choisit m3.xlarge avec 3 instances dont 1 nœud maître et 2 nœuds principaux.

Finalement on met notre clé sinon il faudra créer une clé, la figure ci-dessous montre les étapes remplir maintenant que tout est prêt on clique sur créer une cluster.

II Mettre les données dans S3

Il y a pas de plus simple que charger les données dans le S3 on créer un dossier de input et un autre dossier de output pour les données de sorties. Comme dans la figure ci-dessous.

A screenshot of the Amazon S3 console. The left sidebar shows 'Amazon S3 > flights2015'. The main area displays a list of objects in the 'flights2015' bucket. The objects are:

Nom	Dernière modification	Taille	Classe de stockage
input	--	--	--
jCQDOVACOY9GCPGS	--	--	--
j2CVNWEXTJNU	--	--	--
j392BCYBB2K09	--	--	--
jBURXJPSUPCVH	--	--	--
output	--	--	--

The 'Actions' dropdown menu is open at the bottom, showing options: 'Charger', '+ Créer un dossier', 'Télécharger', and 'Actions'.

Voici le jeu de données vous pouvez télécharger à partir de ce lien.

III Chargement de données dans le cluster

Tout d'abord on retourne dans notre EMR on affiche les données présentent dans le S3. On va exécuter quelque commandes comme ci dessous.

```
[hadoop@ip-172-31-18-115 ~]$ aws s3 ls
2019-12-02 15:28:18 aws-logs-023429030018-eu-west-1
2019-09-23 12:22:52 com-rosettahub-default-u-4246485c-1f94-4717-9292
2020-01-16 23:15:51 flights2015
2020-01-15 15:35:45 ler06
[hadoop@ip-172-31-18-115 ~]$ hadoop fs -ls s3://flights2015
/l: 's3://flights2015': No such file or directory
[hadoop@ip-172-31-18-115 ~]$ /
-bash: /: Is a directory
[hadoop@ip-172-31-18-115 ~]$ hadoop fs -ls s3://flights2015/input/
Found 3 items
-rw-rw-rw- 1 hadoop hadoop      359 2020-01-16 23:17 s3://flights2015/input/airlines.csv
-rw-rw-rw- 1 hadoop hadoop    23867 2020-01-16 23:17 s3://flights2015/input/airports.csv
-rw-rw-rw- 1 hadoop hadoop  592406591 2020-01-16 23:17 s3://flights2015/input/flights.csv
[hadoop@ip-172-31-18-115 ~]$
```

Maintenant tant qu'on a le chemin des données il suffit de les copiers dans le hdfs on s'utilisant cette commande on arrive a ce résultat.

Sachant qu'il y a plus qu'une méthode mais pour mon cas c'est la seule qui a fonctionner.

```
[hadoop@ip-172-31-18-115 ~]$ hadoop dfs -copyToLocal s3://flights2015/input/airlines.csv
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

20/01/18 15:17:07 INFO s3n.S3NativeFileSystem: Opening 's3://flights2015/input/airlines.csv' for reading
[hadoop@ip-172-31-18-115 ~]$ ls
airlines.csv
[hadoop@ip-172-31-18-115 ~]$ hadoop dfs -copyFromLocal airlines.csv /user/hadoop/
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

[hadoop@ip-172-31-18-115 ~]$
```

Il faut répéter l'opération pour les autres fichiers de la même manière.

```
hadoop dfs -copyToLocal s3://flights2015/input/airports.csv
&& hadoop dfs -copyFromLocal airports.csv /user/hadoop/
hadoop dfs -copyToLocal s3://flights2015/input/flights.csv
&& hadoop dfs -copyFromLocal flights.csv /user/hadoop/
```

Dans la prochaine section on utiliser hive pour mettre ces données dans des tables.

IV Initialisation des tables

Dans cette étape il suffit de lancer hive ensuite créer une base de données, créer une table et importer nos données sachant que le plus gros fichiers c'est flights qui comporte plus que 6 millions de ligne on fera un exemple sur un fichier plus petit ensuite c'est la même logique, les autres requêtes seront fourni aussi.

Comme illustre la figure ci-dessous.

```
[hadoop@ip-172-31-18-115 ~]$ hive
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive> create database flights2015;
OK
Time taken: 0.741 seconds
hive> use flights2015;
OK
Time taken: 0.033 seconds
hive> create external table if not exists flights2015.airlines
    row format delimited
    fields terminated by ','
    >     AIRLINE string)
    >     ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';
OK
Time taken: 0.544 seconds
hive> load data inpath '/user/hadoop/airlines.csv' into table flights2015.airlines;
Loading data to table flights2015.airlines
OK
Time taken: 0.712 seconds
hive> select * from airlines;
OK
FATA_CODE      AIRLINE
UA              United Airlines Inc.
AA              American Airlines Inc.
US              US Airways Inc.
FD              FedEx Express Inc.
B6              JetBlue Airways
OO              Southwest Airlines Inc.
AS              Alaska Airlines Inc.
NK              Sun Country Airlines
WN              Delta Air Lines Inc.
DL              Atlantic Southeast Airlines
EV              Hawaiian Airlines Inc.
MQ              American Eagle Airlines Inc.
VX              Virgin America
Time taken: 1.705 seconds, Fetched: 15 row(s)
hive>
```

Voici les requêtes suivantes concernant les autres tables.

```
create external table if not exists flights2015.airlines
(IATA_CODE string,
AIRLINE string)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';;
```

```
load data inpath '/user/hadoop/airlines.csv' into table
flights2015.airlines;
```

```
create external table if not exists flights2015.airports
(IATA_CODE string,
AIRPORT string,
CITY string,
STATE string,
COUNTRY string,
LATITUDE string,
LONGITUDE string)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';;
```

```
load data inpath '/user/hadoop/airports.csv' into table
flights2015.airports;
```

```
create external table if not exists flights2015.flights
(YEAR integer,
MONTH integer,
DAY integer,
DAY_OF_WEEK integer,
AIRLINE string,
FLIGHT_NUMBER integer,
TAIL_NUMBER string,
ORIGIN_AIRPORT string,
DESTINATION_AIRPORT string,
SCHEDULED_DEPARTURE integer,
DEPARTURE_TIME integer,
DEPARTURE_DELAY integer,
TAXI_OUT integer,
WHEELS_OFF integer,
SCHEDULED_TIME integer,
ELAPSED_TIME integer,
AIR_TIME integer,
DISTANCE integer,
WHEELS_ON integer,
TAXI_IN integer,
SCHEDULED_ARRIVAL integer,
```

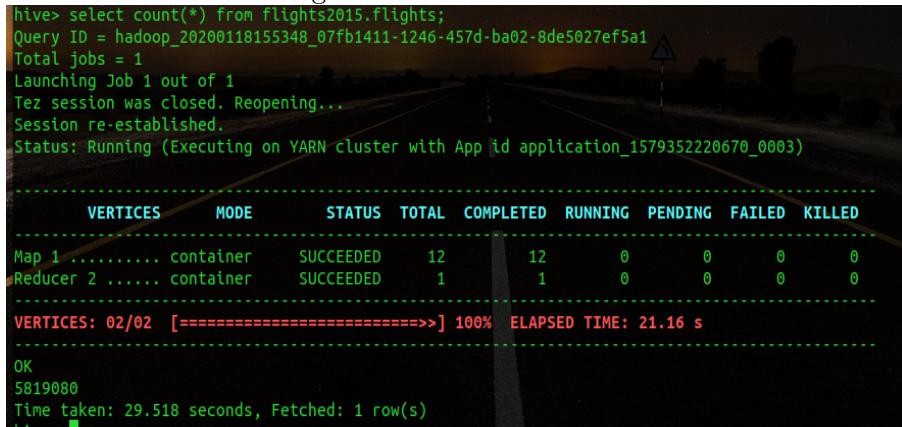
```

ARRIVAL_TIME integer ,
ARRIVALDELAY integer ,
DIVERTED integer ,
CANCELLED integer ,
CANCELLATION_REASON string ,
AIR_SYSTEM_DELAY integer ,
SECURITY_DELAY integer ,
AIRLINE_DELAY integer ,
LATE_AIRCRAFT_DELAY integer ,
WEATHER_DELAY integer )
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

```

```
load data inpath '/user/hadoop/flights.csv' into table flights2015.flights;
```

Avant de finir on va regarder la taille de nos données présent dans le hive comme illustre la figure ci-dessous.



```

hive> select count(*) from flights2015.flights;
Query ID = hadoop_20200118155348_07fb1411-1246-457d-ba02-8de5027ef5a1
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1579352220670_0003)

-----  

 VERTICES      MODE      STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED  

-----  

 Map 1 ..... container SUCCEEDED    12      12      0      0      0      0  

 Reducer 2 ..... container SUCCEEDED     1       1      0      0      0      0  

-----  

 VERTICES: 02/02 [=====>>>] 100% ELAPSED TIME: 21.16 s  

-----  

OK  

5819080  

Time taken: 29.518 seconds, Fetched: 1 row(s)

```

Plus 5.8 millions de ligne présentent dans la table `flights`, que c'était impossible à ouvrir avec un ordinateur normal en csv est là c'est la magie de hive ,Maintenant on peut commencer à executer d'autres requêtes pour manipuler notre table c'est ce qu'on va voir dans le prochain chapitre.

Chapitre 3

Manipulation hive

I Questions à traiter dans le projet

1. Classement des aéroports avec plus influences
2. Classement des aéroports avec moins influences
3. Classement des meilleures compagnies aériennes par rapport nombre de vol
4. estimation du retard par rapport à l'aéroports

II Décomposer les tables

II.1 La liste des compagnies aériennes

Sachant que cette table comporte 14 lignes il sera facile d'identifier les codes de compagnies aériennes après extraire leurs informations.

```
hive> select * from flights2015.airlines;
OK
IATA_CODE      AIRLINE
UA      United Air Lines Inc.
AA      American Airlines Inc.
US      US Airways Inc.
F9      Frontier Airlines Inc.
B6      JetBlue Airways
OO      Skywest Airlines Inc.
AS      Alaska Airlines Inc.
NK      Spirit Air Lines
WN      Southwest Airlines Co.
DL      Delta Air Lines Inc.
EV      Atlantic Southeast Airlines
HA      Hawaiian Airlines Inc.
MQ      American Eagle Airlines Inc.
VX      Virgin America
Time taken: 2.334 seconds, Fetched: 15 row(s)
hive>
```

II.2 Division des vols

Pour faire un traitement de données sur chaque compagnies on va créer des tables pour chaque compagnies avec leur propres vol grâce à la requête ci-dessous.

```
create table flights2015.virgin_airlines AS
(select * from flights2015.flights where airline='VX');
```

Voici la liste des tables :

1. united_airlines
2. american_airlines
3. us_airways
4. frontier_airlines
5. jetblue_airlines
6. skywest_airlines
7. alaska_airlines
8. spirit_airlines
9. southwest_airlines
10. delta_airlines
11. atlantic_airlines
12. hawaiian_airlines
13. eagle_airlines
14. virgin_airlines

II.3 Opérations sur une compagnie

Dans le but de décomposer les compagnies aériennes chacun tout seul qui va nous faciliter la tâche pour réaliser nos statistiques.

Après la création d'une nouvelle table à partir des informations de la table vol qui comporte plus que 5.8 millions de lignes à fin de faciliter le traitement on va se baser sur cette dernière pour trouver le nombre de vol effectuer par aéroport. Voici la requête correspondante :

```
create table flights2015.united_airlines_counter AS (select
ORIGIN_AIRPORT, count(ORIGIN_AIRPORT) AS counter
from flights2015.united_airlines GROUP BY ORIGIN_AIRPORT);
```

Si on regarde bien notre nouvelle table elle comporte quelques intrus qui sont des nombres au lieu des noms des aéroports pour régler ce problème mineur on va exécuter une nouvelle requête.

```
create table flights2015.virgin_airlines_counter_clean
AS (select origin_airport,counter from
flights2015.virgin_airlines_counter JOIN flights2015.airports ON (origin_
```

Voici maintenant notre table plus propre :

```
hive> select * from virgin_airlines_counter_clean;
OK
BOS      1593
DCA      1234
LAS      3766
LAX      11801
LGA      1197
MCO      334
ORD      1380
PSP      211
SAN      1742
SEA      2128
TUL      1
PDX      342
AUS      1399
DAL      5050
HNL      60
IAD      1488
SFO      15940
FLL      1240
EWR      1832
JFK      3672
OGG      29
Time taken: 0.082 seconds, Fetched: 21 row(s)
hive>
```

III Classement d'aéroport (aéroport de départ)

Maintenant que nos table de compteurs sont prêtes on va enchaîner pour trouver l'aéroport avec le plus d'influence concernant le départ des avions. On créer une table qui va stocker ces données sur toutes les compagnies.

```
create external table if not exists flights2015.departure_airports_class
(IATA_CODE string ,
NUMBER_OF_FLIGHTS_YEARLY integer ,
MOST_LANDED_AEROPORT string ,
NB_MOST_LANDED_AEROPORT integer ,
LESS_LANDED_AEROPORT string ,
NB_LESS_LANDED_AEROPORT integer );
```

On commence par faire la somme des vols de cette compagnies aériennes, ensuite on cherche le maximum et le minimum de vol et les noms d'aéroports correspondant.

Voici une vue sur les requêtes :

```
hive> select SUM(counter) from virgin_airlines_counter_clean;
Query ID = hadoop_20200119193441_fb8a4053-4699-43dd-ac23-583a9cc8a6a4
Total Jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1579447531230_0010)
User: hadooperf@30-57-111.eu-west-1.compute.amazonaws.com Permission denied (public key)
    hope
--> VERTICES DOCUMENT MODE STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED
Map 1 ..... container SUCCEEDED 1 1 0 0 0 0
Reducer 2 ..... container SUCCEEDED 1 1 0 0 0 0
----- VERTICES: 02/02 [=====>>>] 100% ELAPSED TIME: 6.33 s
OK
56439
Time taken: 13.885 seconds, Fetched: 1 row(s)
hive> select MAX(counter) from virgin_airlines_counter_clean;
Query ID = hadoop_20200119193542_9c86205e-5262-4495-869d-8dcbaa0c4681
Total Jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1579447531230_0010)
--> VERTICES DOCUMENT MODE STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED
Map 1 ..... container SUCCEEDED 1 1 0 0 0 0
Reducer 2 ..... container SUCCEEDED 1 1 0 0 0 0
----- VERTICES: 02/02 [=====>>>] 100% ELAPSED TIME: 5.96 s
OK
15940
Time taken: 6.446 seconds, Fetched: 1 row(s)
hive> select * from virgin_airlines_counter_clean where counter=15940;
OK
SFO 15940
Time taken: 0.113 seconds, Fetched: 1 row(s)
```

Après l'extraction de données on va insérer celà dans notre nouvelle table comme ceci.

```
hive> insert into table departure_airports_class VALUES('VX',56439,'SFO',15940,'TUL',1);
Query ID = hadoop_20200119194452_c21f240c-2ffb-4270-bef1-936f2a40f61c
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1579447531230_0011)

----- VERTICES DOCUMENT MODE STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED
Map 1 ..... container SUCCEEDED 1 1 0 0 0 0
----- VERTICES: 01/01 [=====>>>] 100% ELAPSED TIME: 6.07 s
----- Loading data to table flights2015.departure_airports_class
OK
Time taken: 13.843 seconds
```

Maintenant il faudra répéter l'opération pour le reste des compagnies aériennes, finalement on obtient une table comme ceci.

```
TIME taken: 0.778 seconds
hive> select * from class_departure_airports;
OK
MQ      272650  ORD     63130   PHL     9
HA      70030   HNL     30206   PPG    107
US      198715  CLT     44373   DSM     1
AA      648694  DFW     134270  LIT    10
UA      469829  ORD     59538   AGS     1
UX      56439   SFO     15940   TUL     1
EV      526249  IAH     58330   DAB     2
DL      800329  ATL     221705  LAN     4
WN      1157339 MDW     76350   PNS    962
NK      107171  FLL     11511   CRW     64
AS      158054  SEA     50004   CHS     26
OO      539545  DEN     44633   BWI     1
B6      245135  JFK     40455   ALB     44
F9      82735   DEN     21175   FAT     2
Time taken: 0.067 seconds, Fetched: 14 row(s)
hive> □
```

A partir de cette table on peut trouver :

1. Classement des aéroports avec plus influences
2. Classement des aéroports avec moins influences
3. Classement des meilleures compagnies aériennes par rapport nombre de vol

Il faudra maintenant exporter cette table pour manipuler cette table qui faisait au début 5.8 millions de lignes elle est maintenant réduite à 14 lignes. Tout d'abord on va mettre cette table dans un dossier, voici le résultat :

```
hive> insert overwrite local directory '/home/hadoop/res' row format delimited fields terminated by ',' select * from class_departure_airports;
Query ID = hadoop_20200119213059_f85cebb4-b9dc-477c-97b2-a64d8d2e92b9
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1579447531230_0019)

-----  

VERTICES  MODE    arc STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED  

-----  

Map 1ervers..... container  SUCCEEDED  2      2      0      0      0      0  

-----  

VERTICES: 01/01  [=====>>>] 100% ELAPSED TIME: 7.08 s  

-----  

Moving data to local directory /home/hadoop/res
OK
```

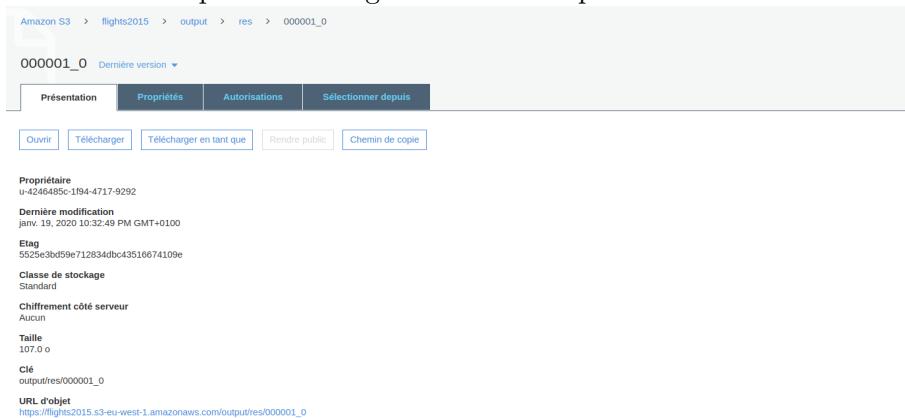
Ensuite on va mettre ça sur le S3 :

```
[hadoop@ip-172-31-14-252 ~]$ hadoop dfs -copyFromLocal /home/hadoop/res s3://flights2015/output/
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

a1430708.

28/01/19 21:32:47 INFO s3n.MultipartUploadOutputStream: close closed:false s3://flights2015/output/res/000000_0_COPYING_
28/01/19 21:32:47 INFO s3n.MultipartUploadOutputStream: close closed:true s3://flights2015/output/res/000000_0_COPYING_
28/01/19 21:32:47 INFO s3n.S3NativeFileSystem: rename s3://flights2015/output/res/000000_0_COPYING_ s3://flights2015/output/res/000000_0
28/01/19 21:32:48 INFO s3n.MultipartUploadOutputStream: close closed:false s3://flights2015/output/res/000001_0_COPYING_
28/01/19 21:32:48 INFO s3n.MultipartUploadOutputStream: close closed:true s3://flights2015/output/res/000001_0_COPYING_
28/01/19 21:32:48 INFO s3n.S3NativeFileSystem: rename s3://flights2015/output/res/000001_0_COPYING_ s3://flights2015/output/res/000001_0
```

Finalement on peut télécharger les fichiers à partir du S3 :



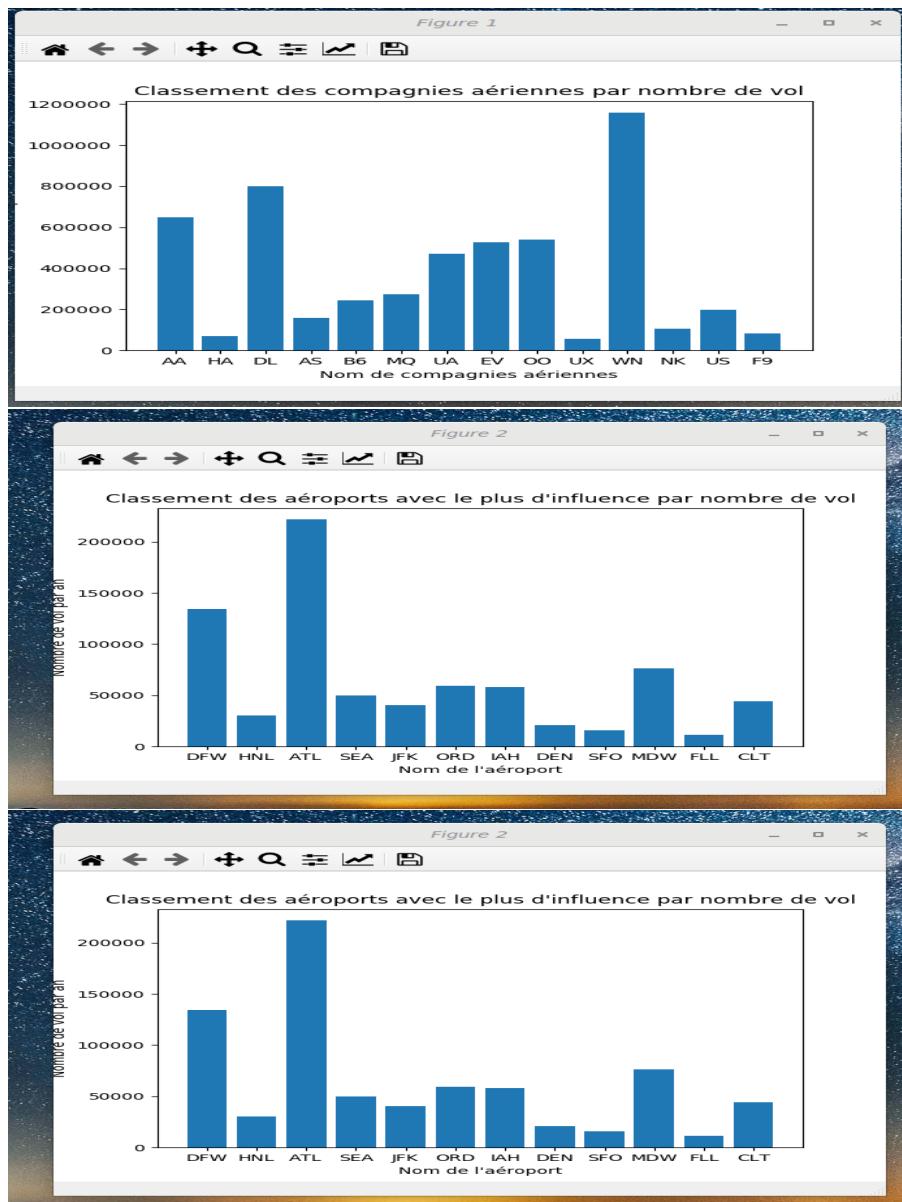
IV Affichage des données avec Python

Dans cette section on va afficher notre table avec une représentation graphique c'est simple on commence par télécharger notre fichier et le convertir en csv.

```
$ hope@hope:~/Documents/BigData/0-Cadre_Logiciel_Big_Data/Sparky$ cat classement_airports.txt | tr
-s [:blank:] ',' > file.csv
$ hope@hope:~/Documents/BigData/0-Cadre_Logiciel_Big_Data/Sparky$
```

Maintenant on va se servir de pyspark et des bibliothèques comme `numpy`, `pandas`, `plt`, on va joindre le code qui va nous permettre de faire cette représentation.

Enfin, voici notre table avec une représentation plus belle on peut maintenant lire facilement la meilleure compagnie aériennes classé par nombre de vol et l'aéroport qui à le plus d'influence en étant aéroport de départ.



On voulais faire la même chose avec les aéroports d'arrivée ça sera très intéressant d'avoir une visualisation sur les deux aéroports et d'où peut être déduire la différence et voir si l'aéroport peut jouer un rôle sur le retard d'un avion.

Chapitre 4

Conclusion

Pour conclure, Les services AWS représentent beaucoup d'avantages par rapport à la conception de l'architecture normal de programmation qui permettent la facilité de l'utilisation et la flexibilité surtout la sécurité sachant que ces derniers sont utilisé par des grands groupes.

Finalement ce projet m'a permis d'apprendre des nouvelles technologies et approndir mes connaissances en Big Data malheureusement je voulais faire un plus de choses sur ce jeu de données intéressant mais il y avait quelques contraintes.