# Mise en place d’un cluster Hadoop

Un **cluster Hadoop** est un type spécial de **cluster** de calcul (grappe de serveurs) conçu spécifiquement pour stocker et analyser de grandes quantités de données non structurées au sein d'un environnement de calcul distribué. Le travail d'analyse de données est réparti entre les différents nœuds du **cluster** (serveurs).

# Installation de Hadoop Spark

Nous allons mettre en place un cluster hadoop pas à pas.

Le seul prérequis à ce tutoriel est d’avoir à disposition quatre machines accessibles via SSH.
Certains fournisseurs permettent la duplication de machines virtuelles. Si cela est possible il suffira alors de réaliser cette configuration sur une machine et de la dupliquer trois fois.
Dans le cas contraire, il faudra reproduire les étapes qui suivent sur chaque machine.

# Configuration des machines

1 - Installation de Java 8

```
sudo apt install openjdk-8-jdk
```

2 - Téléchargement de Hadoop

```
sudo wget -P ~ https://mirrors.sonic.net/apache/hadoop/common/hadoop-3.2.1/hadoop-3.2.1.tar.gz
```

3 - Décompression du fichier

```
tar xzf hadoop-3.2.1.tar.gz
```

4 - Modification du nom du répertoire (pour alléger les commandes suivantes) :

```
mv hadoop-3.2.1 hadoop
```

5 - Ouvrir et modifier le fichier environnement de hadoop :
a - Ouverture du fichier:

```
nano ~/hadoop/etc/hadoop/hadoop-env.sh
```

b - Trouver la ligne :

```
# export JAVA_HOME=
```

c - La remplacer par la ligne suivante :

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/
```

6 - Déplacer le dossier hadoop:

```
sudo mv hadoop /usr/local/hadoop
```

7 - Ouvrir et modifier le fichier environment :
a - Ouvrir le fichier :

```
sudo nano /etc/environment
```

b - Ajouter la configuration suivante :

```
PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:
/usr/local/games:/usr/local/hadoop/bin:/usr/local/hadoop/sbin"
JAVA_HOME="/usr/lib/jvm/java-8-openjdk-amd64/jre"
```

8 - Créer un utilisateur “hadoopuser” :

```
sudo adduser hadoopuser
```

Appuyer sur “Entrée” jusqu’à revenir à l’invite de commande. Puis entrer les commandes suivantes :

```
sudo usermod -aG hadoopuser hadoopuser
sudo chown hadoopuser:root -R /usr/local/hadoop/
sudo chmod g+rwx -R /usr/local/hadoop/
sudo adduser hadoopuser sudo
```

9 - Modification du fichier hosts en accord avec la topologie du cluster :
Une des machines (la première par exemple) sera la machine “maître” les autres seront les “esclaves”. Entrez la commande suivante :

```
sudo nano /etc/hosts
```

Puis ajoutez les informations relatives aux machines du cluster sous la forme suivante :

```
<adresse_ip_machine0> hadoop-master
<adresse_ip_machine1> hadoop-slave1
<adresse_ip_machine2> hadoop-slave2
<adresse_ip_machine3> hadoop-slave3
```

10 - Cloner la machine ou reproduire l’installation **à l’identique** sur les autres machines.

11 - Modifier le nom de chacune des machine :
Pour chaque machine effectuer la manipulation avec le nom correspondant, par exemple pour la machine master :

```
sudo echo “hadoop-master” > /etc/hostname
```

# Configuration spécifique de la machine “maître”

Après avoir configuré **toutes les machines** (y compris la machine “maître”) en suivant les étapes suivantes, effectuer les manipulations suivantes sur la machine maître.

1 - Passer sous l’identité hadoopuser :

```
su - hadoopuser
```

2 - Générer une clef ssh

```
ssh-keygen -t rsa
```

3 - Copier la clef ssh à yous les utilisateurs

```
ssh-copy-id hadoopuser@hadoop-master
ssh-copy-id hadoopuser@hadoop-slave1
ssh-copy-id hadoopuser@hadoop-slave2
ssh-copy-id hadoopuser@hadoop-slave3
```

4 - Modification du fichier core-site.xml

a - Ouvrir le fichier “core-site.xml :

```
sudo nano /usr/local/hadoop/etc/hadoop/core-site.xml
```

b - Ajouter les lignes suivantes :

```
<configuration>
<property>
<name>fs.defaultFS</name>
<value>hdfs://hadoop-master:9000</value>
</property>
</configuration>
```

5 - Modification du fichier hdfs-site.xml

a - Ouvrir le fichier

```
sudo nano /usr/local/hadoop/etc/hadoop/hdfs-site.xml
```

b - Ajouter les configurations suivantes

```
<configuration>
<property>
<name>dfs.namenode.name.dir</name><value>/usr/local/hadoop/data/nameNode</valu
e>
</property>
<property>
<name>dfs.datanode.data.dir</name><value>/usr/local/hadoop/data/dataNode</valu
e>
</property>
<property>
```

```
<name>dfs.replication</name>
<value>2</value>
</property>
</configuration>
```

6 - Ouvrir et modifier le fichier worker
a - Ouvrir le fichier

```
sudo nano /usr/local/hadoop/etc/hadoop/workers
```

b - Ajouter les lignes suivantes

```
hadoop-slave1
hadoop-slave2
hadoop-slave3
```

7 - Copier les configurations du “maître”sur les “esclaves” (depuis la machine “maître”) :

```
scp /usr/local/hadoop/etc/hadoop/*
hadoopuser@hadoop-slave1:/usr/local/hadoop/etc/hadoop/
scp /usr/local/hadoop/etc/hadoop/*
hadoopuser@hadoop-slave2:/usr/local/hadoop/etc/hadoop/
scp /usr/local/hadoop/etc/hadoop/*
hadoopuser@hadoop-slave3:/usr/local/hadoop/etc/hadoop/
```

8 - Formatage du système de fichier HDFS

```
source /etc/environment
hdfs namenode -format
```

9 - Démarrage du fichier de fichier HDFS

```
start-dfs.sh
```

# Configuration de yarn

1 - **Sur la machine maître**

```
export HADOOP_HOME="/usr/local/hadoop"
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
export HADOOP_HDFS_HOME=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_YARN_HOME=$HADOOP_HOME
```

2 - **Sur les machines esclaves**
a - Ouvrir le fichier yarn-site.xml

```
sudo nano /usr/local/hadoop/etc/hadoop/yarn-site.xml
```

b - Ajouter les lignes suivantes

```
<property>
<name>yarn.resourcemanager.hostname</name>
<value>hadoop-master</value>
</property>
```

3 - **Sur la machine maître** démarrer yarn

```
start-yarn.sh
```

# Finalisation et démarrage du master

Formatez le serveur HDFS :

```
hdfs namenode -format -force
```

Démarrez tous les services Hadoop (HDFS, Yarn…)

```
cd /usr/local/hadoop/sbin
./start-all.sh
```

Vous pouvez vérifier que tous les services sont bien démarrés :

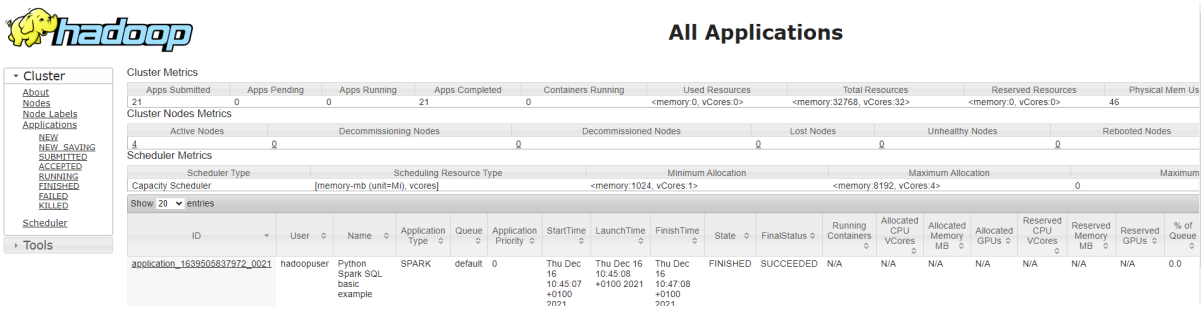
```
jps
10384 DataNode
11009 NodeManager
4113 ResourceManager
11143 Jps
10218 NameNode
10620 SecondaryNameNode
```

Vous pouvez lister le contenu du serveur HDFS :

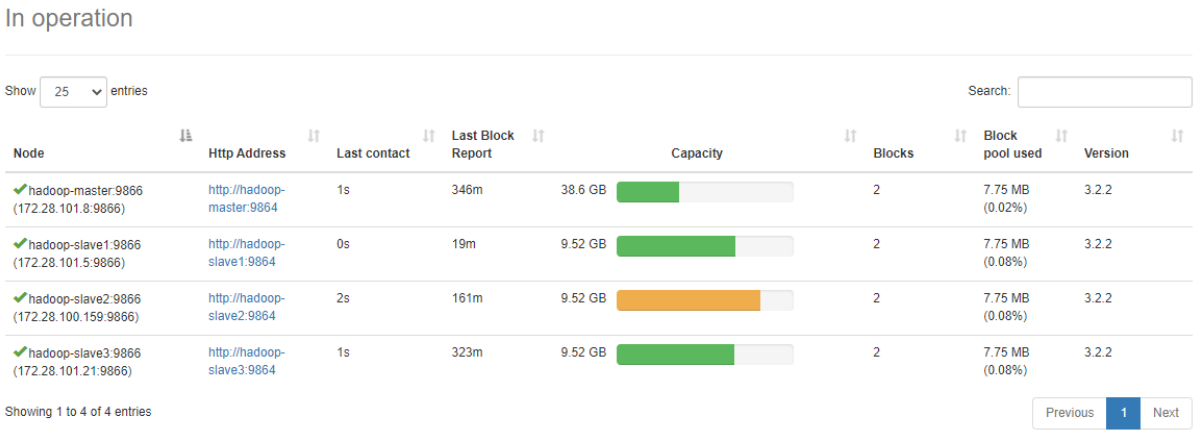
```
hdfs dfs -ls /
Found 0 item
```

## Superviser

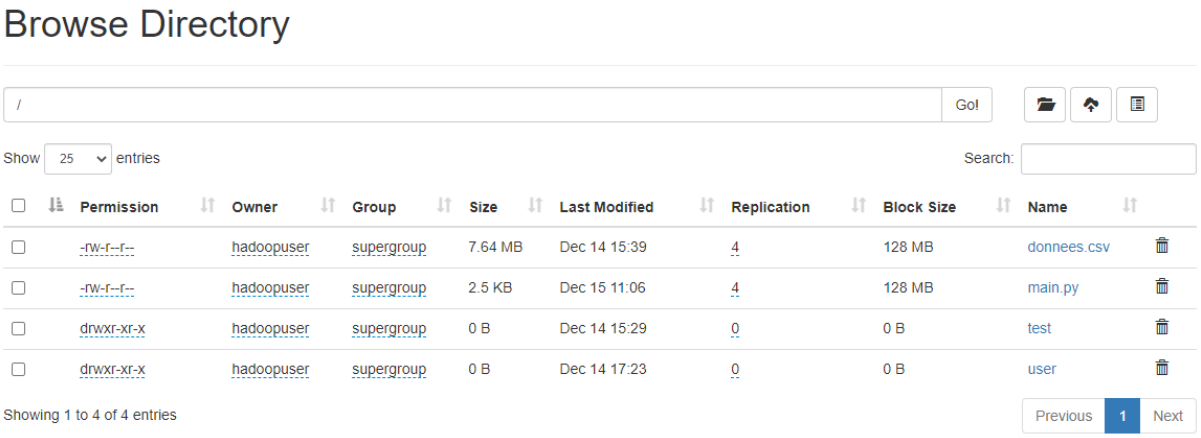
Vous pouvez visualiser les tâches données à Hadoop en allant sur votre navigateur, sur l'adresse IP de votre master en ":8088/cluster"




Vous avez aussi accès en ":9870" au tableau de bord de Hadoop. Dans l'onglet Datanodes vous pourrez visualiser les machines connectées au système Hadoop.




Et dans l'onglet "Utilities > Browse File System" vous aurez accès au fichier importés sur le serveur HDFS.




# Installation de Spark

Téléchargez le binaire de spark et décompressez le :

```
wget https://d3kbcqa49mib13.cloudfront.net/spark-2.2.0-bin-hadoop2.7.tgz
tar -xvf spark-2.2.0-bin-hadoop2.7.tgz
mv spark-2.2.0-bin-hadoop2.7 spark
```

Modifiez le profil utilisateur :

```
nano .profil
```

```
PATH=/home/hadoopuser/spark/bin:$PATH
export HADOOP_CONF_DIR=/home/hadoopuser/hadoop/etc/hadoop
export SPARK_HOME=/home/hadoopuser/spark
export PYSPARK_PYTHON=/usr/bin/python3
export PYSPARK_DRIVER_PYTHON=/usr/bin/python3
```

# Lancer une tâche Hadoop PySpark

Ici nous utilisons Python avec la librairie Spark pour établir des requêtes SQL. Le jeu de données est ici les hospitalisations du Covid19, nous faisons une requête pour récupérer la différence entre les hommes et les femmes dans les entrées en hospitalisation et en réanimation.
Nous nous servons de la librairie Plotly basée sur GraphX de Spark pour générer le graphique, cette librairie à la possibilité de distribuer ses opérations au contraire de matplotlib.

Téléchargement du jeu de données :

```
wget
https://www.data.gouv.fr/fr/datasets/r/63352e38-d353-4b54-bfd1-f1b3ee1cabd7
```

Notre fichier main.py :

```
from chart_studio import plotly as py
import plotly.graph_objs as go
import pandas as pd
from pyspark.sql import SparkSession
from pyspark.sql.types import StructType,StructField, StringType, IntegerType
from pyspark.sql.types import ArrayType, DoubleType, BooleanType
from pyspark.sql.functions import col,array_contains

spark = SparkSession \
    .builder \
    .appName("Python Spark SQL basic example") \
    .config("spark.some.config.option", "some-value") \
    .getOrCreate()

schema = StructType() \
      .add("dep",StringType(),True) \
      .add("sexe",StringType(),True) \
      .add("jour",StringType(),True) \
      .add("hosp",IntegerType(),True) \
      .add("rea",IntegerType(),True) \
      .add("HospConv",StringType(),True) \
      .add("SSR_USLD",StringType(),True) \
      .add("autres",StringType(),True) \
      .add("rad",StringType(),True) \
      .add("dc",StringType(),True)

df = spark.read.format("csv") \
      .option("header", True) \
```

```
      .option("delimiter", ";") \
      .schema(schema) \
      .load("hdfs://hadoop-master:9000/donnees.csv")

df.createOrReplaceTempView("data")
df2 = spark.sql("SELECT jour, SUM(hosp) as h , SUM(rea) as r FROM data WHERE
sexe='0' GROUP BY jour ORDER BY jour")
df3 = spark.sql("SELECT jour, SUM(hosp) as h , SUM(rea) as r FROM data WHERE
sexe='1' GROUP BY jour ORDER BY jour")
df4 = spark.sql("SELECT jour, SUM(hosp) as h , SUM(rea) as r FROM data WHERE
sexe='2' GROUP BY jour ORDER BY jour")

data = go.Figure()
data.add_trace(go.Scatter(
    x=df2.toPandas()['jour'],
    y=df3.toPandas()['h'],
    name='Hospitalisations H',
    line=dict(color="#77B5FE")
))
data.add_trace(go.Scatter(
    x=df2.toPandas()['jour'],
    y=df4.toPandas()['h'],
    name='Hospitalisations F',
    line=dict(color="#d0312d")
))
data.add_trace(go.Scatter(
    x=df2.toPandas()['jour'],
    y=df3.toPandas()['r'],
    name='Réanimation H',
    line=dict(color="#03224c")
))
data.add_trace(go.Scatter(
    x=df2.toPandas()['jour'],
    y=df4.toPandas()['r'],
    name='Réanimation F',
    line=dict(color="#fd6c9e")
))
data.write_image("covid.png")
```

Afin de pouvoir démarrer une tâche distribuée sur Hadoop il faut premièrement téléverser le fichier de données et le script python.

```
hdfs dfs -put donnes.csv /donnes.csv
hdfs dfs -put main.py /main.py
```

```
hdfs dfs -ls /
Found 2 items
-rw-r--r--   4 hadoopuser supergroup    8014150 2021-12-14 15:39 /donnees.csv
-rw-r--r--   4 hadoopuser supergroup       2557 2021-12-15 11:06 /main.py
```

Nous pouvons ensuite lancer la tâche Spark :

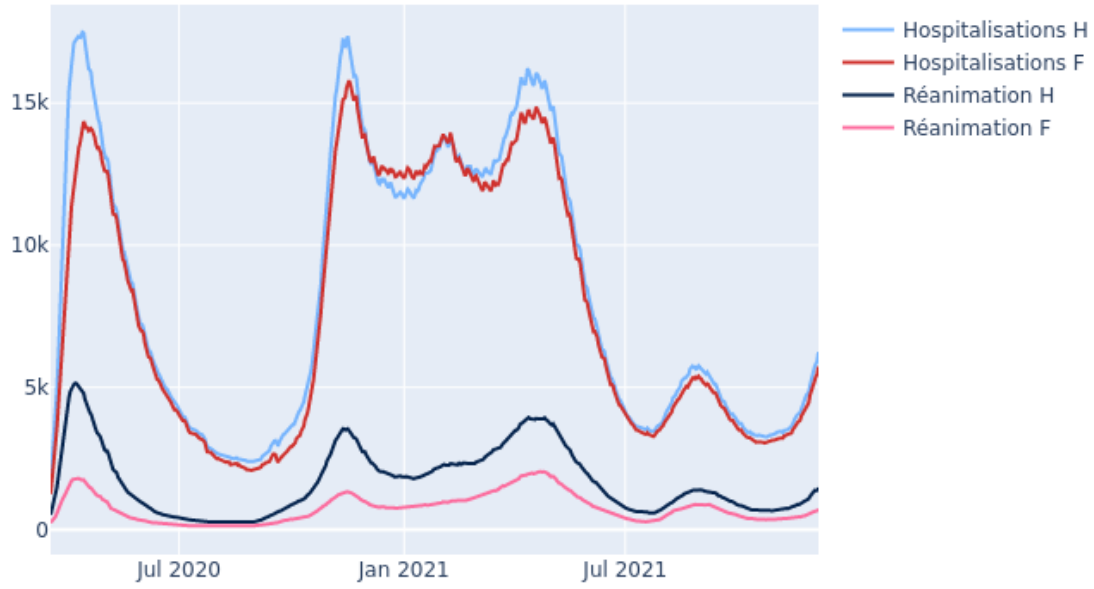
```
spark-submit --master yarn --deploy-mode cluster --py-files
hdfs://hadoop-master:9000/main.py hdfs://hadoop-master:9000/main.py
```

Nous voyons sur l'écran de supervision la tâche qui se déroule sur le cluster :

| ID | User | Name | Application Type | Queue | Application Priority | StartTime | LaunchTime | FinishTime | State | FinalStatus | Running Containers | Allocated CPU VCores | Allocated Memory MB | Allocated GPUs | Reserved CPU VCores | Reserved Memory MB | Reserved GPUs | % of Queue |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| application_1639505837972_0023 | hadoopuser | Python Spark SQL basic example | SPARK | default | 0 | Thu Dec 16 12:12:00 +0100 2021 | Thu Dec 16 12:12:00 +0100 2021 | N/A | RUNNING | UNDEFINED | 3 | 3 | 5120 | -1 | 0 | 0 | -1 | 15.6 |

# Conclusion

La tâche Spark précédente a généré le fichier image suivant, présentant le nombre d'hommes et de femmes atteint de covid-19 hospitalisés en France, ainsi que parmi ces hospitalisations le nombre de patients admis en service de réanimation :




Analyse du graphe :

Les courbes ci-dessus semblent indiquer qu'il y a légèrement plus d'hommes que de femmes hospitalisés pour cause de covid-19, notamment lors des pics épidémiques. Cette différence est d'autant plus remarquable en ce qui concerne les patients en réanimation atteint de la covid-19.
Les questions suivantes se posent alors :

- Y a t-il plus d'hommes que de femmes en France ?
- Les hommes ont-ils plus de comportements à risque impliquant une exposition à la covid-19 plus importante que les femmes en France ?
- Les hommes sont-ils plus sujets au développement de formes graves de covid-19 nécessitant d'être placés dans un service de réanimation ?
- Y a t-il une différence de traitement entre les hommes et les femmes en ce qui concerne le passage d'un service d'hospitalisation générale à un service de réanimation en France ?
- Ce constat est-il le même dans d'autres pays ?

Sans recherches plus approfondies sur ces questions, il n'est pas possible d'apporter de conclusion quant à la prédisposition des hommes à développer des formes plus graves de covid-19 que les femmes nécessitant la prise en charge dans un service de réanimation.

Comparaison des versions locale et distribuée du programme :

Nous n'observons pas de gain de temps en lançant le programme de manière distribuée sur le cluster hadoop plutôt qu'en le lançant localement avec les données stockées localement. Nous avons même un temps d'exécution globalement plus long avec la version distribuée. Nous pensons que cela est dû principalement à des temps de communication (réseau) entre les machines et peut-être un peu aussi à un coût d'orchestration de la distribution. Nous pensons aussi que le fichier de données n'est pas assez volumineux pour compenser ces coûts et apporter un réel intérêt à la distribution.