

Connexion de Mongo-DB dans AWS	1
Connexion SPARK - MongoDB	7
Création MongoDB backups	8
Sauvegarde sous AWS S3 du fichier de backup	8
Restauration MongoDB backups	9
Redémarrage de processus	9

0. Créer sur EMR (comme en cours) un cluster qui contient 3 Machines, ou cloner le cluster utilisé en cours:

Pour cloner, aller sur le cluster crée et cliquer sur cloner. Ensuite la page suivante devrait apparaître.

You can use the AWS Glue Data Catalog as your external Hive metastore for Apache Spark [S](#), Apache Hive [H](#), and Presto [P](#) workloads on Amazon EMR release 5.10.0 and later. To get started, simply select the AWS Glue Data Catalog for table metadata when creating your cluster.

Create cluster
View details
Clone
Terminate

Filter: All clusters ▾ Filter clusters ... 8 clusters (all loaded) ↻

Name	ID	Status	Creation time (UTC+1)	Elapsed time	Normalized instance hours
GdeltCluster	j-2886HXVA3WX7Y	Starting	2019-01-10 06:22 (UTC+1)	1 minute	0

Summary
Master public DNS: --
Termination protection: Off Change
Tags: -- View All / Edit

Steps

Name	Status	Start time (UTC+1)	Elapsed time
Setup hadoop debugging	Pending	--	--

Bootstrap actions

Name
No bootstrap actions available

Hardware
Master: Provisioning 1 m1.large
Core: Provisioning 2 m1.large
Task: --

View cluster details
View monitoring details

Create Cluster - Advanced Options [Go to quick options](#)

[Step 1: Software and Steps](#)

[Step 2: Hardware](#)

[Step 3: General Cluster Settings](#)

Step 4: Security

Security Options

EC2 key pair ⓘ

☒ Cluster visible to all IAM users in account ⓘ

Permissions ⓘ

☐ Default ☒ Custom

Select custom roles to tailor permissions for your cluster.

EMR role ⓘ

EC2 instance profile ⓘ

Auto Scaling role ⓘ

Authentication and encryption

EC2 security groups

An EC2 security group acts as a virtual firewall for your cluster nodes to control inbound and outbound traffic. There are two types of security groups you can configure, [EMR managed security groups](#) and [additional security groups](#). EMR will [automatically update](#) the rules in the EMR managed security groups in order to launch a cluster. [Learn more](#)

Type	EMR managed security groups <small>EMR will automatically update the selected group</small>	Additional security groups <small>EMR will not modify the selected groups</small>
Master	<input type="text" value="Default: sg-0055e98f7ae88e8d1 (ElasticMapR"/>	No security groups selected ✎
Core & Task	<input type="text" value="Default: sg-00a251cdbfe2cea18 (ElasticMapR"/>	No security groups selected ✎

[Create a security group](#)

[Cancel](#)

[Previous](#)

[Create cluster](#)

1. **Se diriger sur la plateforme EC2 et nommer les machines**

The screenshot displays the AWS Management Console interface. On the left, a sidebar lists navigation options like EC2 Dashboard, Events, Tags, Reports, Limits, INSTANCES, Launch Templates, Spot Requests, Reserved Instances, Dedicated Hosts, Scheduled Instances, Capacity Reservations, IMAGES, AMIs, Bundle Tasks, ELASTIC BLOCK STORE, Volumes, Snapshots, Lifecycle Manager, NETWORK & SECURITY, Security Groups, Elastic IPs, Placement Groups, Key Pairs, Platform, Network Interfaces, LOAD BALANCING, Load Balancers, Target Groups, and AUTO SCALING. The main area shows the 'Launch Instance' page with a table of instances:

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP	IPv6 IPs	Key Name	Monitoring	Launch Time
Master	i-05dfeaa6e78a2c4f1	m1.large	us-east-1c	running	Initializing	None	ec2-3-88-181-92.comput...	3.88.181.92	-	gdelKeyPair	disabled	January 10
Slave1	i-06a3b381a58362d25	m1.large	us-east-1c	running	Initializing	None	ec2-3-86-228-9.comput...	3.86.228.9	-	gdelKeyPair	disabled	January 10
Slave2	i-0b28283a47961c9c4	m1.large	us-east-1c	running	Initializing	None	ec2-3-84-195-246.comp...	3.84.195.246	-	gdelKeyPair	disabled	January 10

Below the table, the details for instance 'i-0b28283a47961c9c4 (Slave2)' are shown, including its Public DNS, IPv4 Public IP, and various configuration details like VPC ID, Subnet ID, and Network interfaces.

On the left, a terminal window shows a script for setting up an EC2 instance:

```
#!/bin/bash
set -e

# Set the instance profile
INSTANCE_PROFILE="arn:aws:iam::123456789012:instance-profile/EC2_DefaultRole"

# Set the key pair
KEY_PAIR="gdelKeyPair"

# Set the security groups
SECURITY_GROUPS="sg-12345678,sg-87654321"

# Set the user data
USER_DATA="
#!/bin/bash
set -e

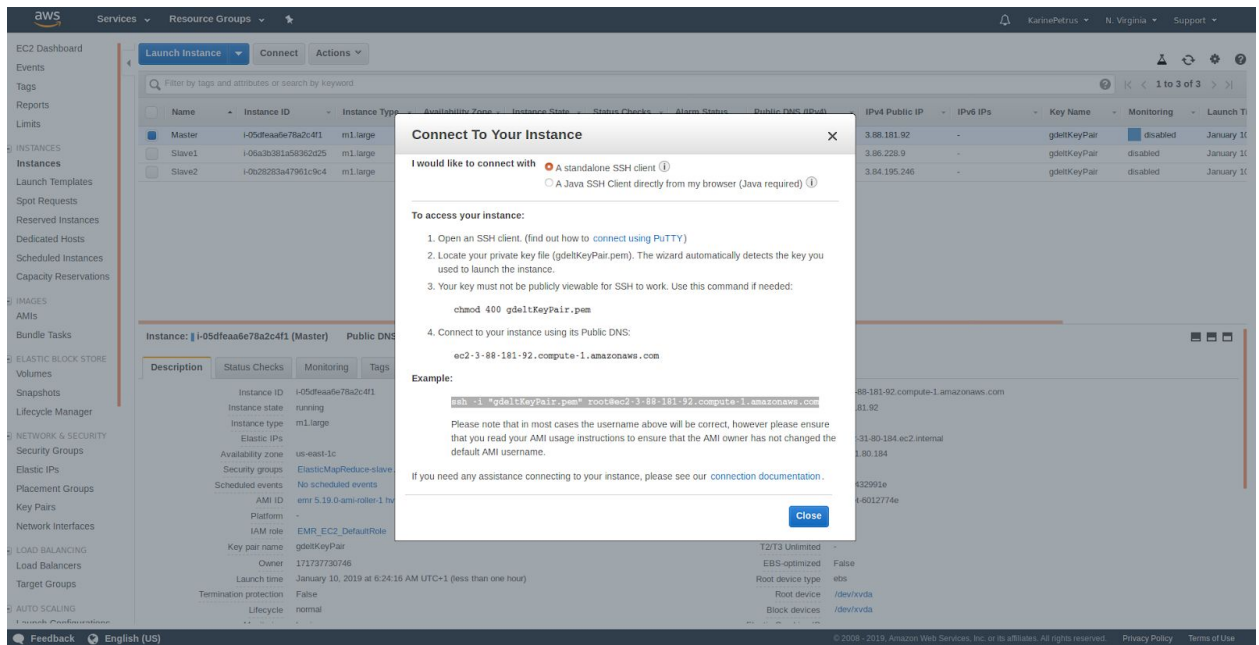
# Install the software
yum install -y java-1.8.0-openjdk

# Set the environment variables
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk
export PATH=$PATH:$JAVA_HOME/bin

# Start the service
systemctl start java
systemctl enable java

# Check the status
systemctl status java
"
```

- On se connecte sur le cluster Master et slaves (machines) via:
`ssh -i "gdelKeyPair.pem" ec2-user@ec2-3-88-181-92.compute-1.amazonaws.com`
 Cette commande est écrite lorsque l'on clique droit sur chaque machine
 (attention au lieu de root, on écrira ec2-user)



3. Sur la machine sur laquelle on s'est connecté:

On crée le fichier :

/etc/yum.repos.d/mongodb-org-4.0.repo

En utilisant la commande :

sudo vim /etc/yum.repos.d/mongodb-org-4.0.repo

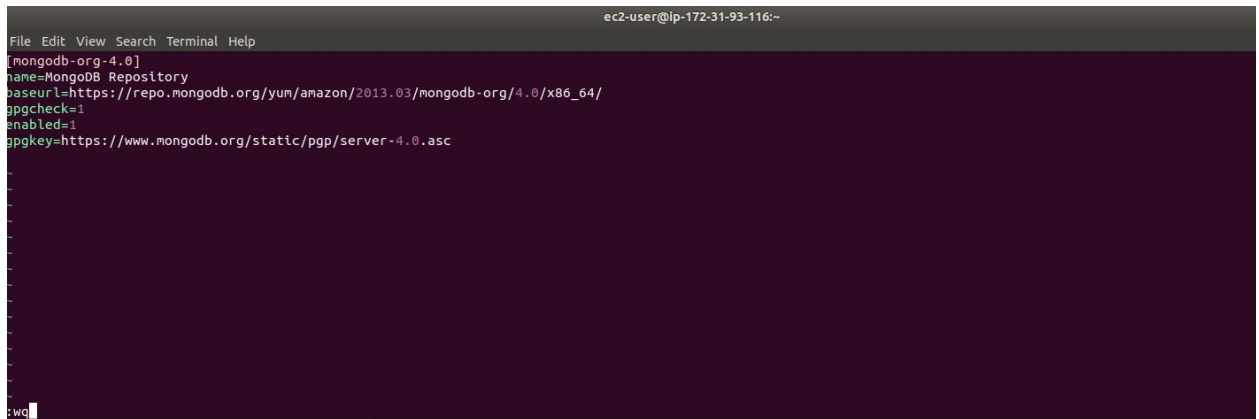
```
ec2-user@ip-172-31-93-116:~$ https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/
25 package(s) needed for security, out of 31 available
Run "sudo yum update" to apply all updates.

EEEEEEEEEEEEEEEEEEEE MMMMMMMM MMMMMMMM RRRRRRRRRRRRRRR
E:::EEEEEEEEEEEEEEEE M:::M M:::M R:::RRRRRRRRRRRR
E:::EEEEEEEEEEEEEEEE M:::M M:::M R:::RRRRRRRRRRRR
E:::E EEEEE M:::M M:::M R:::RRRRRRRRRRRR
E:::E EEEEE M:::M M:::M R:::RRRRRRRRRRRR
E:::EEEEEEEEEEEEEEEE M:::M M:::M R:::RRRRRRRRRRRR
E:::EEEEEEEEEEEEEEEE M:::M M:::M R:::RRRRRRRRRRRR
E:::E EEEEE M:::M M:::M R:::RRRRRRRRRRRR
E:::E EEEEE M:::M M:::M R:::RRRRRRRRRRRR
E:::EEEEEEEEEEEEEEEE M:::M M:::M R:::RRRRRRRRRRRR
E:::EEEEEEEEEEEEEEEE M:::M M:::M R:::RRRRRRRRRRRR
EEEEEEEEEEEEEEEEEEEE MMMMMMMM MMMMMMMM RRRRRRRRRRRRR

ec2-user@ip-172-31-93-116 ~$ vim /etc/yum.repos.d/mongodb-org-4.0.repo
ec2-user@ip-172-31-93-116 ~$ vim /etc/yum.repos.d/mongodb-org-4.0.repo
ec2-user@ip-172-31-93-116 ~$ vim /etc/yum.repos.d/mongodb-org-4.0.repo
ec2-user@ip-172-31-93-116 ~$ sudo vim /etc/yum.repos.d/mongodb-org-4.0.repo
ec2-user@ip-172-31-93-116 ~$
```

Le fichier doit contenir les infos suivantes :

```
[mongodb-org-4.0]
name=MongoDB Repository
baseurl=https://repo.mongodb.org/yum/amazon/2013.03/mongodb-org/4.0/x86_64/
gpgcheck=1
enabled=1
gpgkey=https://www.mongodb.org/static/pgp/server-4.0.asc
```



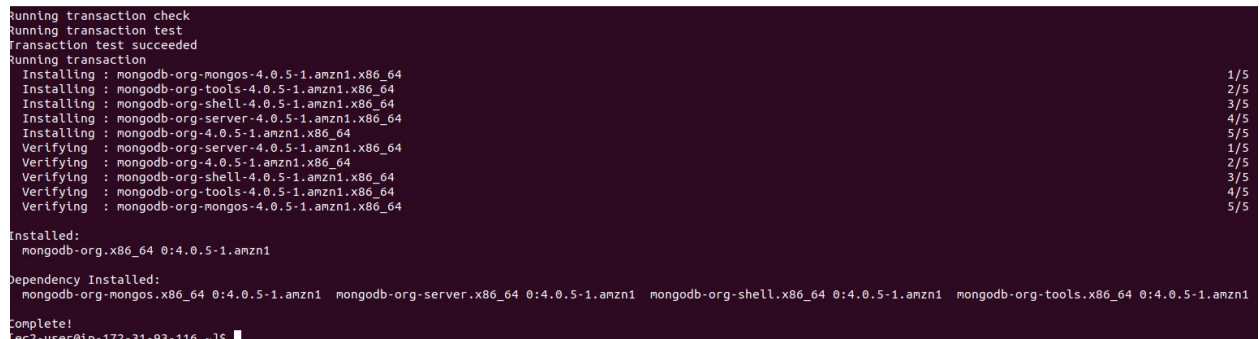
```
ec2-user@ip-172-31-93-116:~
File Edit View Search Terminal Help
[mongodb-org-4.0]
name=MongoDB Repository
baseurl=https://repo.mongodb.org/yum/amazon/2013.03/mongodb-org/4.0/x86_64/
gpgcheck=1
enabled=1
gpgkey=https://www.mongodb.org/static/pgp/server-4.0.asc
```

4. Installer Mongo

Via la commande line :

```
sudo yum install -y mongodb-org
```

L'écran final devrait apparaître de cette manière :



```
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : mongodb-org-mongos-4.0.5-1.amzn1.x86_64 1/5
  Installing : mongodb-org-tools-4.0.5-1.amzn1.x86_64 2/5
  Installing : mongodb-org-shell-4.0.5-1.amzn1.x86_64 3/5
  Installing : mongodb-org-server-4.0.5-1.amzn1.x86_64 4/5
  Installing : mongodb-org-4.0.5-1.amzn1.x86_64 5/5
  Verifying  : mongodb-org-server-4.0.5-1.amzn1.x86_64 1/5
  Verifying  : mongodb-org-4.0.5-1.amzn1.x86_64 2/5
  Verifying  : mongodb-org-shell-4.0.5-1.amzn1.x86_64 3/5
  Verifying  : mongodb-org-tools-4.0.5-1.amzn1.x86_64 4/5
  Verifying  : mongodb-org-mongos-4.0.5-1.amzn1.x86_64 5/5

Installed:
  mongodb-org.x86_64 0:4.0.5-1.amzn1

Dependency Installed:
  mongodb-org-mongos.x86_64 0:4.0.5-1.amzn1  mongodb-org-server.x86_64 0:4.0.5-1.amzn1  mongodb-org-shell.x86_64 0:4.0.5-1.amzn1  mongodb-org-tools.x86_64 0:4.0.5-1.amzn1

Complete!
ec2-user@ip-172-31-93-116:~$
```

5. Créer le repository mongo, le port et réplica set

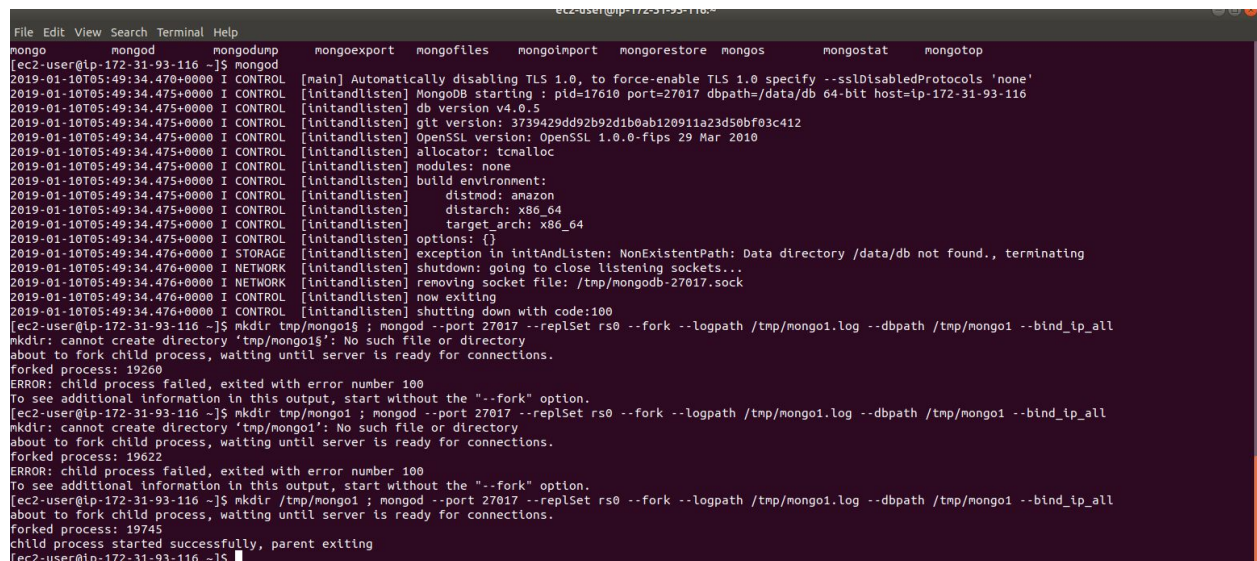
Ecrire la commande sur la machine :

Master:

```
mkdir -p /mnt/mongo1 ; mongod --port 27017 --replSet rs0 --fork --logpath /mnt/mongo1/mongo1.log --dbpath /mnt/mongo1 --bind_ip_all &
```

```
mkdir -p /mnt1/mongo2 ; mongod --port 27018 --replSet rs0 --fork --logpath /mnt1/mongo2/mongo2.log --dbpath /mnt1/mongo2 --bind_ip_all &
```

L'écran suivant devrait s'afficher:



```
File Edit View Search Terminal Help
mongo  mongod  mongodump  mongoexport  mongofiles  mongoimport  mongorestore  mongos  mongostat  mongotop
[ec2-user@ip-172-31-93-116 ~]$ mongod
2019-01-10T05:49:34.470+0000 I CONTROL [main] Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'
2019-01-10T05:49:34.475+0000 I CONTROL [initandlisten] MongoDB starting : pid=17610 port=27017 dbpath=/data/db 64-bit host=ip-172-31-93-116
2019-01-10T05:49:34.475+0000 I CONTROL [initandlisten] db version v4.0.5
2019-01-10T05:49:34.475+0000 I CONTROL [initandlisten] git version: 3739429dd92b92d1b0ab120911a23d50bf03c412
2019-01-10T05:49:34.475+0000 I CONTROL [initandlisten] OpenSSL version: OpenSSL 1.0.0-fips 29 Mar 2010
2019-01-10T05:49:34.475+0000 I CONTROL [initandlisten] allocator: tcmalloc
2019-01-10T05:49:34.475+0000 I CONTROL [initandlisten] modules: none
2019-01-10T05:49:34.475+0000 I CONTROL [initandlisten] build environment:
2019-01-10T05:49:34.475+0000 I CONTROL [initandlisten] distmod: amazon
2019-01-10T05:49:34.475+0000 I CONTROL [initandlisten] distarch: x86_64
2019-01-10T05:49:34.475+0000 I CONTROL [initandlisten] target_arch: x86_64
2019-01-10T05:49:34.475+0000 I CONTROL [initandlisten] options: {}
2019-01-10T05:49:34.476+0000 I STORAGE [initandlisten] exception in initAndListen: NonExistentPath: Data directory /data/db not found., terminating
2019-01-10T05:49:34.476+0000 I NETWORK [initandlisten] shutdown: going to close listening sockets...
2019-01-10T05:49:34.476+0000 I NETWORK [initandlisten] removing socket file: /tmp/mongod-27017.sock
2019-01-10T05:49:34.476+0000 I CONTROL [initandlisten] now exiting
2019-01-10T05:49:34.476+0000 I CONTROL [initandlisten] shutting down with code:100
[ec2-user@ip-172-31-93-116 ~]$ mkdir tmp/mongo1 ; mongod --port 27017 --replSet rs0 --fork --logpath /tmp/mongo1.log --dbpath /tmp/mongo1 --bind_ip_all
mkdir: cannot create directory 'tmp/mongo1': No such file or directory
about to fork child process, waiting until server is ready for connections.
forked process: 19260
ERROR: child process failed, exited with error number 100
To see additional information in this output, start without the "--fork" option.
[ec2-user@ip-172-31-93-116 ~]$ mkdir tmp/mongo1 ; mongod --port 27017 --replSet rs0 --fork --logpath /tmp/mongo1.log --dbpath /tmp/mongo1 --bind_ip_all
mkdir: cannot create directory 'tmp/mongo1': No such file or directory
about to fork child process, waiting until server is ready for connections.
forked process: 19622
ERROR: child process failed, exited with error number 100
To see additional information in this output, start without the "--fork" option.
[ec2-user@ip-172-31-93-116 ~]$ mkdir tmp/mongo1 ; mongod --port 27017 --replSet rs0 --fork --logpath /tmp/mongo1.log --dbpath /tmp/mongo1 --bind_ip_all
about to fork child process, waiting until server is ready for connections.
forked process: 19745
child process started successfully, parent exiting
[ec2-user@ip-172-31-93-116 ~]$
```

Slave1:

```
mkdir -p /mnt/mongo3 ; mongod --port 27019 --replSet rs0 --fork --logpath /mnt/mongo3/mongo3.log --dbpath /mnt/mongo3 --bind_ip_all &
```

```
mkdir -p /mnt1/mongo4 ; mongod --port 27020 --replSet rs0 --fork --logpath /mnt1/mongo4/mongo4.log --dbpath /mnt1/mongo4 --bind_ip_all &
```

Slave2:

```
mkdir -p /mnt/mongo5 ; mongod --port 27021 --replSet rs0 --fork --logpath /mnt/mongo5/mongo5.log --dbpath /mnt/mongo5 --bind_ip_all &
```

```
mkdir -p /mnt1/mongo6 ; mongod --port 27022 --replSet rs0 --fork --logpath /mnt1/mongo6/mongo6.log --dbpath /mnt1/mongo6 --bind_ip_all &
```

Arbitre

```
mkdir -p /mnt1/mongoa/ ; mongod --port 27023 --replSet rs0 --fork --logpath /mnt1/mongoa.log --dbpath /mnt1/mongoa --bind_ip_all &
```

6. Dans le shell mongo (taper mongo dans l'une des machines)
Écrire le fichier de configuration du réplica set :

```
rsconf = {
  _id: "rs0",
  members: [
    {
      _id: 0,
      host: "172.31.33.253:27017"
    }
  ]
}
```

7) Puis on active cette configuration en utilisant la fonction rs.initiate(rsconf).
rs.initiate(rsconf);

8) Vérification du statut
rs.status()

9) Utiliser la fonction rs.add() pour ajouter les autres membres au replica-set. Puis utiliser rs.status() pour afficher l'état du replica-set.

```
rs.add('172.31.33.253:27018')
rs.add('172.31.44.85:27019')
rs.add('172.31.44.85:27020')
rs.add('172.31.47.231:27021')
rs.add('172.31.47.231:27022')
rs.add('172.31.47.231:27023',true)
```

Note: Ci-dessus il faut mettre les bons adresses IPs, dans l'exemple ci-dessous nous avons:

- Master: 172.31.33.253
- Slave1: 172.31.44.85
- Slave2: 172.31.47.231

Liens :

<https://www.youtube.com/watch?v=lr68GVsNWB4>

<https://www.youtube.com/watch?v=NZR1C0u1SML&t=1584s>

https://github.com/ClementTr/AWS_MongoDB_Cluster

Connexion SPARK - MongoDB

Aller dans le répertoire `/etc/alternatives/spark-conf` et éditer le fichier de configuration par défaut de spark comme ci-dessous:

```
sudo vi spark-defaults.conf
```

Ajouter les lignes suivantes à la fin du fichier:

```
spark.mongodb.input.uri=mongodb://172.31.40.176:27017,172.31.40.176:27018,172.31.32.115:27019,172.31.32.115:27020,172.31.36.231:27021,172.31.36.231:27022/
spark.mongodb.input.database=Gdelt
spark.mongodb.input.readPreference.name=primaryPreferred
spark.mongodb.output.uri=mongodb://172.31.40.176:27017,172.31.40.176:27018,172.31.32.115:27019,172.31.32.115:27020,172.31.36.231:27021,172.31.36.231:27022/
spark.mongodb.output.database=Gdelt
```

Ci-dessus il faut remplacer le nom de la base de donnée par le bon, de la même manière on doit modifier les adresses IP.

Gestion des sauvegardes MongoDB

Création MongoDB backups

Sauvegarde de la collection Requete1 dans le répertoire `/mnt1/db_backups/22_01/1`:

```
mongodump --collection Requete1 --db Gdelt --host spark1 --port 27017 --out /mnt1/db_backups/22_01/1
```

Sauvegarde de la collection Fips sur la base Gdelt:

```
mongodump --collection Fips --db Gdelt --host spark1 --port 27017 --out /mnt1/db_backups/22_01/1
```

Sauvegarde de toute la base Gdelt:

```
mongodump --db Gdelt --out /mnt1/db_backups/22_01/1
```

Sauvegarde sous AWS S3 du fichier de backup

```
cd /mnt1/db_backups/22_01/1
```

```
tar cvf Gdelt.22_01.1.tar.gz *
```

```
aws s3 cp Gdelt.22_01.1.tar.gz s3://stephane-mulard-telecom-gdelt2018/
```


Restauration MongoDB backups

Restauration de la collection Requete1 en remplaçant les données existantes:
mongorestore --drop --collection Requete1 --db Gdelt --host spark2 --port 27019
/mnt1/db_backups/22_01/1/Gdelt/Requete1.bson

Restauration de la collection Requete1 sans remplacer les données existantes:
mongorestore --collection Requete1 --db Gdelt --host spark2 --port 27019
/mnt1/db_backups/22_01/1/Gdelt/Requete1.bson

Restauration de la collection Fips avec remplacement des données existantes:
mongorestore --drop --collection Fips --db Gdelt --host spark2 --port 27019
/mnt1/db_backups/22_01/1/Gdelt/Fips.bson

Redémarrage de processus

- **Zeppelin**
 - sudo stop zeppelin
 - sudo start zeppelin
-