

---

UNIVERSITÉ PARIS 8 - VINCENNES À SAINT-DENIS

M1 MIASHS : Big Data et fouille de données

## Création d'une architecture distribuée et analyse de données

PANCHALINGAMOORTHY GAJENTHRAN

Organisme d'accueil : Université Paris 8  
Cours : Cadre logiciel pour le Big Data

# Table des matières

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Datasets</b>	<b>2</b>
<b>3</b>	<b>AWS</b>	<b>3</b>
3.1	Service S3 . . . . .	4
3.2	Service EMR . . . . .	5
<b>4</b>	<b>Conclusion</b>	<b>9</b>

# Chapitre 1

## Introduction

Avec l'émergence des entreprises de streaming, les utilisateurs consomment de plus en plus de films, de séries ou encore d'oeuvres cinématographiques. La satisfaction des utilisateurs représente donc un enjeu important pour les entreprises telles que Netflix, Amazon Prime ou encore Disney+. Cela passe donc par un système de recommandation de films afin d'avoir un gain de temps conséquent et d'améliorer l'expérience utilisateur.

A l'aide de l'architecture distribuée **Amazon Web Services** (AWS), nous nous pencherons sur ce sujet et analyserons les données à l'aide d'**Hive**. Apache Hive est une infrastructure d'entrepôt de données intégrée sur Hadoop permettant l'analyse, le requêtage via un langage proche syntaxiquement de SQL ainsi que la synthèse de données. Bien que initialement développée par Facebook, Apache Hive est maintenant utilisée et développée par d'autres sociétés comme Netflix. Amazon maintient un fork d'Apache Hive qui inclut Amazon Elastic MapReduce dans Amazon Web Services (source : Wikipédia).

Dans ce papier, nous allons d'abord décrire les datasets utilisées pour la phase d'analyse, puis nous créerons l'architecture distribuée AWS, puis nous détaillerons les différents requêtes **Hive**, ensuite nous comparerons l'efficacité de ce dernier avec un autre langage et enfin nous discuterons à propos des avantages proposés par l'architecture distribuée.

# Chapitre 2

## Datasets

Le système de recommandation se reposera quasiment que sur les films. Afin d'obtenir des résultats fiables et de profiter pleinement de l'architecture distribuée, une base de données d'environ 1Go a été récupérée, en format *.csv*. Elle a été reprise de *Kaggle* qui elle-même vient des films listés par *MovieLens*, et contient des films venant de 2018 ou avant. Celle-ci est découpée en plusieurs parties :

- *movies.csv* : regroupe l'identifiant des films (`movieId`), l'identifiant des films sur le site IMDB (`imdb_id`) et le titre des films (`title`).
- *ratings.csv* : regroupe l'identifiant des utilisateurs (`userId`), l'identifiant des films (`movieId`), et la note des utilisateurs (`rating`) sur 5 et la date à laquelle ils ont noté (`timestamp`).
- *votes.csv* : regroupe l'identifiant des utilisateurs (`userId`), l'identifiant des films (`movieId`), l'identifiant des films sur le site IMDB (`imdb_id`) et le vote moyen des films (`vote_average`) et le nombre de votant pour chaque film (`vote_count`).

Ainsi, nous nous appuyons sur une base de données de plus de 45K films et notamment 26 millions de votes dont plus de 250K votants.

## Chapitre 3

# Création d'une architecture distribuée (AWS)

Suite à la limite de crédit dépassé sur le compte, le projet a donc dû être reporté sur un autre compte mais cela n'aura une influence ni sur le fonctionnement de l'architecture, ni sur les explications apportées ci-dessous.

Afin de bénéficier des services proposés par AWS, il fallait tout d'abord se connecter à celui-ci. Pour cela, nous sommes passé par l'interface **RosettaHUB** (fig.3.1) afin de créer de notre compte (déjà réalisé lors des séances "Cadre logiciel pour le Big Data"). Une fois le compte créé, nous devons alimenter notre compte en crédit pour pouvoir utiliser les services AWS. Les détails seront épargnés vu qu'il s'agit ici de manoeuvres déjà exécutées en cours.

Dès que les prépartifis sont terminés, nous pouvons nous attaquer à l'architecture. Tout d'abord, nous devons créer une instance en se dirigeant vers le service **EC2**. Un tutoriel a été mis en ligne sur la plateforme Moodle afin d'expliquer en détails les différentes étapes à suivre pour lancer une instance. Avant de nous occuper des clusters, nous allons dans un premier temps, récolter les données depuis notre ordinateur et les placer dans le service **S3**.

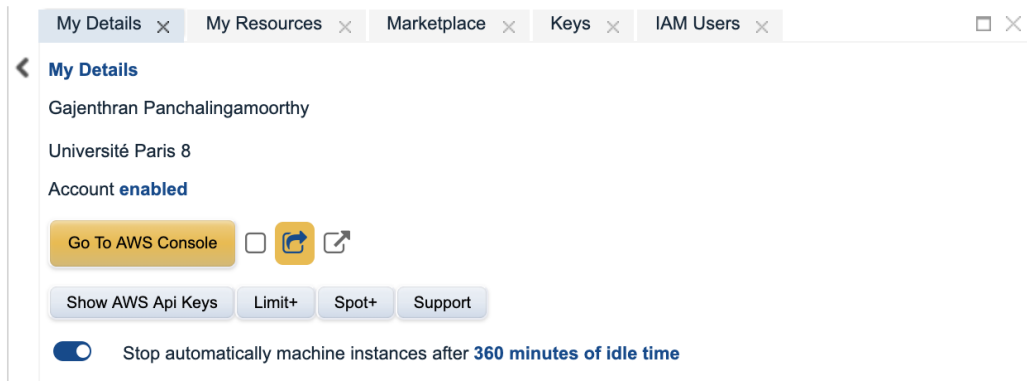


FIGURE 3.1 – Interface RosettaHUB

### 3.1 Service S3

Le service S3 permet de stocker les données en ligne dans le cloud afin qu'on puisse les réutiliser avec d'autres services d'AWS dans la section suivante. Il offre la possibilité de stocker n'importe quelle quantité de données et également d'assurer leurs protections avec une sécurité très fiable. Pour profiter de ce service, dirigeons nous vers la console AWS, et cherchons le service S3.

Le service S3 range les données par compartiments (fig.3.2). Pour ajouter de nouvelles données, il suffit de créer un compartiment en cliquant sur le bouton *Créer un compartiment*. Cela ouvrira une nouvelle fenêtre dans laquelle nous préciserons le nom du compartiment pour pouvoir l'identifier et le récupérer lors de notre connexion (fig.3.3). Une fois le compartiment créé, nous pouvons importer des fichiers à travers le bouton *Charger* et de charger les fichiers que nous souhaitons. Les fichiers sont plutôt lourds donc le temps des transactions risque de durer assez longtemps. A noter que ne nous comptons pas utiliser toutes les données de la base de données, donc une répartition des données de façon judicieuse s'impose<sup>1</sup>. De plus, afin d'avoir une base de données plus organisées, nous pouvons également les ranger dans des dossier avec *Créer un dossier*.

---

1. Toutes les colonnes des fichiers *.csv* ne vont pas être exploitées, ainsi la décision de supprimer les colonnes inutiles a été prise afin d'alléger la taille de la base.

Compartiments S3 [Découvrir la console](#)

Rechercher compartiments Tous les types d'accès

+ Créer un compartiment Modifier les paramètres d'accès public Vider Supprimer 4 Compartiments 1 Régions

<input type="checkbox"/>	Nom du compartiment	Accès	Région	Date de création
<input type="checkbox"/>	aws-logs-265677857824-eu-west-1	Les objets peuvent être publics	UE (Irlande)	janv. 18, 2020 8:53:26 PM GMT+0100
<input type="checkbox"/>	com-rosettahub-default-u-2532857f-f816-47f7-bc74	Les objets peuvent être publics	UE (Irlande)	sept. 23, 2019 2:03:04 PM GMT+0200
<input type="checkbox"/>	gajenmovies	Compartiments et objets non publics	UE (Irlande)	janv. 18, 2020 9:04:21 PM GMT+0100
<input type="checkbox"/>	gajmovies	Compartiments et objets non publics	UE (Irlande)	janv. 19, 2020 2:38:03 PM GMT+0100

FIGURE 3.2 – Amazon S3 et son rangement par compartiment

Créer un compartiment

1 Nom et région 2 Configurer des options 3 Définir des autorisations 4 Vérification

Nom et région

Nom du compartiment

Région

Copier les paramètres d'un compartiment existant

FIGURE 3.3 – Création d'un nouveau compartiment dans Amazon S3

3.2 Service EMR

Désormais, nous pouvons lancer les clusters. Pour cela, nous allons nous rediriger vers la console AWS et chercher le service **EMR**. Le service **EMR** (fig.3.4) est la plateforme de mégadonnées native cloud leader qui traite de grandes quantités de données rapidement et à moindre coût. Utilisant des outils open source tels que **Apache Spark**, **Apache Hive**, **Apache HBase**, **Apache Flink**, **Apache Hudi** (Incubating) et **Presto**, associés à la scalabi-

lit  dynamique d'Amazon EC2 et au stockage  volutif d'Amazon S3 (source : AWS.Amazon.com).

i ** conomisez jusqu'  90 % sur le calcul**  
 R duisez les co ts et r duisez le d lai de visibilit  en incluant des instances Spot dans vos clusters EMR. [En savoir plus.](#) x

**Cr er un cluster**
Afficher les d tails
Cloner
R silier

Filtre : Tous les clusters
Filtrer les clusters...
2 clusters (tous charg s)
C

	Nom	ID	Statut	Heure de cr�ation (UTC+1)	Temps �coul�	Heures d'instances normalis�es
<input type="checkbox"/>	<span style="color: green;">▶</span> clustermovies	j-1124UINM9VFBD	D�marrage en cours	19-01-2020 16:43 (UTC+1)	1 minute	0

FIGURE 3.4 – Amazon EMR et la liste des clusters

En cliquant sur *Cr er un cluster*, nous allons pouvoir cr er un nouveau cluster (fig.3.5). Il y a plusieurs param tres   remplir comme le nom du cluster, les logiciels pouvant  tre utilis s (en l'occurrence, pour ce projet, Coore Hadoop sera utilis ).

### Configuration g n rale

**Nom du cluster** clustermovies

☒ Journalisation i

**Dossier S3** s3://aws-logs-265677857824-eu-west-1/elasticmapreduce/ 📁

**Mode de lancement** ☒ Cluster i ☐ Ex cution d' tape i

### Configuration des logiciels

**Lib rer** emr-5.29.0 ⬆ ⬇ i

**Applications** ☒ 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

☐ HBase: HBase 1.4.10 with Ganglia 3.7.2, Hadoop 2.8.5, Hive 2.3.6, Hue 4.4.0, Phoenix 4.14.3, and ZooKeeper 3.4.14

☐ Presto: Presto 0.227 with Hadoop 2.8.5 HDFS and Hive 2.3.6 Metastore

☐ Spark: Spark 2.4.4 on Hadoop 2.8.5 YARN with Ganglia 3.7.2 and Zeppelin 0.8.2

☐ Utiliser AWS Glue Data Catalog pour les m tadonn es de table i

FIGURE 3.5 – Cr ation de clusters dans Amazon EMR

De plus, nous pr ciserons le nom de la cl  que nous avons instanci , avec



comme nombre équivalent d'instance à 3 (à savoir 1 noeud maître et 2 noeuds principaux) avec le type d'instance *m3.xlarge* afin d'éviter la surconsommation (fig.3.6)

## Configuration du matériel

Type d'instance

m3.xlarge

Nombre d'instances

3

(1 nœud maître et 2 nœuds principaux)

## Sécurité et accès

Paire de clés EC2

gajaws

[Apprenez à créer une paire de clés EC2](#)

Autorisations

☒ Par défaut

☐ Personnalisé

Utilisez les rôles IAM par défaut. Si des rôles sont absents, ils seront créés automatiquement pour vous avec des stratégies gérées pour les mises à jour automatiques de stratégies.

Rôle EMR

[EMR\\_DefaultRole](#)

[?](#)

Profil d'instance EC2

[EMR\\_EC2\\_DefaultRole](#)

[?](#)

FIGURE 3.6 – Choix des instances dans Amazon EMR

Puis, pour établir la connexion ssh et profiter des fonctionnalités proposées par **EMR**, attendez que votre soit totalement prêt (symbolisé par un rond rempli en vert), puis dirigez vous sur le cluster en question afin de trouver la commande utile à la connexion (fig.3.7).

```
MacBook-Pro-de-Gajenthiran:~ gajen$ sudo ssh -i ~/gajaws.pem hadoop@ec2-34-253-8-79.eu-west-1.compute.amazonaws.com
Password:
[The authenticity of host 'ec2-34-253-8-79.eu-west-1.compute.amazonaws.com (34.253.8.79)' can't be established.
ECDSA key fingerprint is SHA256:lsJXy78aj9Z6mrD88aeUjZ6G6XuNd1s3rk0tKDSWPZw.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ec2-34-253-8-79.eu-west-1.compute.amazonaws.com,34.253.8.79' (ECDSA) to the list of known
Last login: Sun Jan 19 16:06:51 2020
```

```
--|  --|  )
_|  (    /  Amazon Linux AMI
---|\---|---
```

```
https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/
3 package(s) needed for security, out of 17 available
Run "sudo yum update" to apply all updates.
```

```

EEEEEEEEEEEEEEEEEEEE MMMMMMMM MMMMMMMM RRRRRRRRRRRRRR
E:.....:.....:.....: M:.....: M:.....: M:.....: R:.....: R
EE:.....:EEEEEEEE:.....: M:.....: M M:.....: M R:.....: RRRRRR:.....: R
E:.....: EEEEE M:.....: M M:.....: M RR:.....: R R:.....: R
E:.....: E M:.....: M: M M:.....: M R:.....: R R:.....: R
E:.....:EEEEEEEE M:.....: M M: M M:.....: M R:.....: RRRRRR:.....: R
E:.....:.....: E M:.....: M M:.....: M M:.....: M R:.....: RRR:.....: RR
E:.....:EEEEEEEE M:.....: M M:.....: M M:.....: M R:.....: RRRRRR:.....: R
E:.....: E M:.....: M M:.....: M M:.....: M R:.....: R R:.....: R
E:.....: E EEEEE M:.....: M MMM M:.....: M R:.....: R R:.....: R
EE:.....:EEEEEEEE:.....: E M:.....: M M:.....: M R:.....: R R:.....: R
E:.....:.....: M:.....: M M:.....: M RR:.....: R R:.....: R
EEEEEEEEEEEEEEEEEEEE MMMMMMMM MMMMMMMM RRRRRRR RRRRRR

```

FIGURE 3.7 – Etablissement de la connexion ssh avec EMR

La connexion étant établie, il nous reste à transférer les fichiers stockés dans S3 (fig.3.8, dans notre EMR à l'aide de l'argument `-copyToLocal` en indiquant le nom de notre compartiment (fig.3.9. Une fois les fichiers `.csv` importés localement, il faudra envoyer ces fichiers dans `/user/hadoop/` via l'argument `-fromLocal` (fig.3.10).

```
[hadoop@ip-172-31-44-42 ~]$ hadoop dfs -ls s3://gajmovies/
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

Found 2 items
-rw-rw-rw-  1 hadoop hadoop      1500061 2020-01-19 13:39 s3://gajmovies/movies.csv
-rw-rw-rw-  1 hadoop hadoop      1041857 2020-01-19 13:39 s3://gajmovies/votes.csv
```

FIGURE 3.8 – Affichage des compartiments stockés sur S3 avec EMR

```
[hadoop@ip-172-31-44-42 ~]$ hadoop dfs -copyToLocal s3://gajmovies/*
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

20/01/19 16:20:42 INFO s3n.S3NativeFileSystem: Opening 's3://gajmovies/movies.csv' for reading
20/01/19 16:20:42 INFO s3n.S3NativeFileSystem: Opening 's3://gajmovies/votes.csv' for reading
[hadoop@ip-172-31-44-42 ~]$ ls
movies.csv  votes.csv
```

FIGURE 3.9 – Copie des fichiers contenus dans S3 avec EMR

```
[hadoop@ip-172-31-44-42 ~]$ hadoop dfs -copyFromLocal *.csv /user/hadoop/
```

FIGURE 3.10 – Transfert des fichiers locaux dans `/user/hadoop/` avec EMR

Désormais, il nous reste plus qu'à exploiter **Hive** afin d'analyser les données que nous venons de récupérer. Pour tester si le logiciel fonctionne correctement, il suffit de taper la commande `hive`, ou `hive -f <fichier.hive>`.

## Chapitre 4

### Analyse des données (Hive)

Chapitre 5

Conclusion