

Configuration d'une liaison entre une Base de donnée NoSQL DynamoBb Dynanmo

1. [Spark](#)

- [Installation de Spark](#)
- [Configuration du framework](#)

2. [AWS](#)

- [Installation de l'invité de commande AWS](#)
- [Configuration de l'environnement AWS](#)

3. [Developpement](#)

Avant de se lancer dans la connexion des différentes entités nous devons installer les outils dont nous aurons besoin. Dans ce TP nous ne nous attarderons pas sur le framework Hadoop car nous ne l'utiliserons pas dans ce module mais pour découvrir l'installation de la version compatible avec Spark référez vous au cours NB: La version Hadoop doit être ≥ 3

1 - Spark

Spark est le framework le plus populaire dans le calcul distribué il crée des clusters afin d'accélérer le traitement et faciliter l'exécution.

1.1 - Installation du framework Spark

Pour ce projet j'utiliserai **wget** et **git** afin de télécharger des paquets. Si vous ne l'avez pas installez saisissez la commande suivante pour l'obtenir :

```
sudo apt install wget git
```

Pour installer le framework nous devons installer le **jdk** (Java Development Kit) pour ce faire ouvrez le terminal et exécutez la commande suivante :

```
# Installe le jdk par défaut en ce moment
sudo apt install default-jdk
```

Une fois l'installation terminée vous pouvez vérifier qu'il est correctement installé :

```
# version 11.0.16
java -version
```

Une fois ces éléments sur notre machine nous pouvons procéder à l'installation du framework Spark. Pour exécuter une application Spark, il suffit de télécharger une version nous travaillerons avec la version 3.3.1 [telecharger](#)

```
mkdir code
cd code
wget https://www.apache.org/dyn/closer.lua/spark/spark-3.3.1/spark-3.3.1-
bin-hadoop3.tgz
tar xvf spark-3.3.1-bin-hadoop3.tgz
mv spark-3.3.1-bin-hadoop3 spark
```

ensuite pour vérifier que tout fonctionne nous allons réaliser le **wordcount** (Hello world du calcul distribué) qui utilise l'opération **MapReduce**. Créer le fichier `wordcount.py` avec le contenu suivant :

```
import sys
from pyspark import SparkContext

sc = SparkContext()
lines = sc.textFile(sys.argv[1])
word_counts = lines.flatMap(lambda line: line.split(' ')) \
                    .map(lambda word: (word, 1)) \
                    .reduceByKey(lambda count1, count2: count1 + count2) \
                    .collect()

for (word, count) in word_counts:
    print(word, count)
```

Créons également un fichier avec ces quelques lignes

```
echo "Sur mes cahiers d'écolier Sur mon pupitre et les arbres Sur le sable
de neige J'écris ton nom" > text.txt
echo "Sur les pages lues Sur toutes les pages blanches Pierre sang papier
ou cendre J'écris ton nom" >> text.txt
echo "Sur les images dorées Sur les armes des guerriers Sur la couronne des
rois J'écris ton nom" >> text.txt
```

Vous pouvez alors compter le nombre d'occurrences de chaque mot dans le fichier `text.txt`

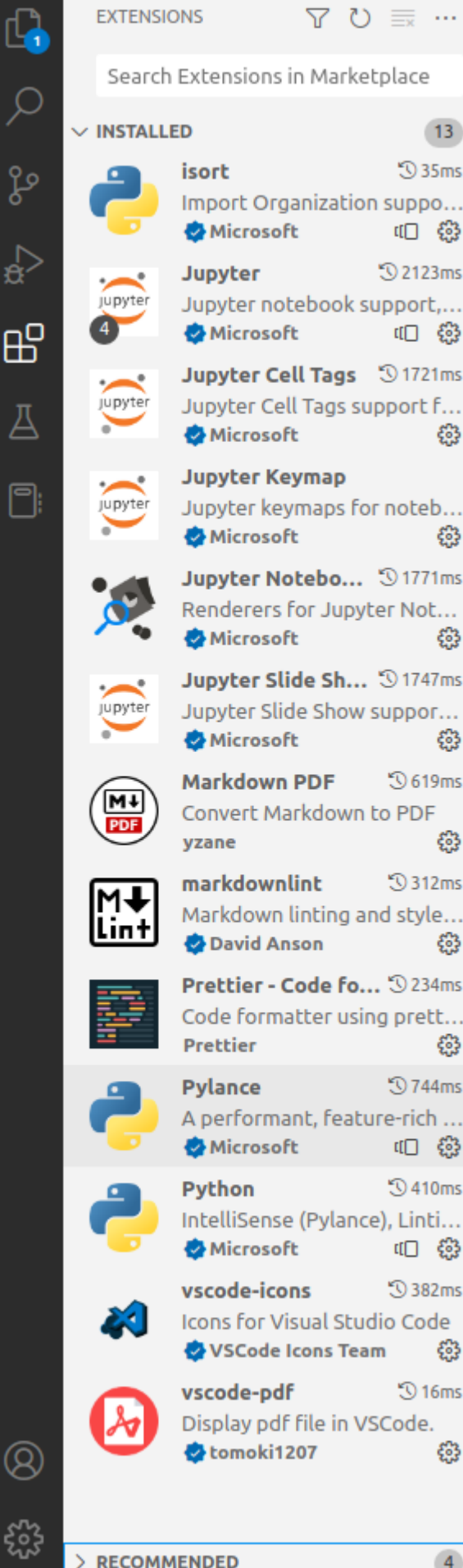
```
./spark/bin/spark-submit ./wordcount.py ./text.txt
```

Configuration de l'environnement de developpement

Une fois l'installation terminée nous devons configurer l'environnement dans lequel le projet sera développé. Commençons par récupérer le lien du jdk et de python (c'est d'autant plus simple lorsque vous avez anaconda installé sur votre machine)

```
readlink -f $(which java)
# pour recuperer le path python
which python
```

Installer L'IDE Vscode developper par Microsoft et disponible dans le **Ubuntu Software** avec les extentions



Lors de l'exécution vous serez invité à télécharger jupyter afin de lancer le serveur de développement via **pip** mais l'éditeur s'occupe de tous, vous devez simplement accepter le popup qui apparaît à l'écran

Ensuite modifions les variables d'environnement qui devrait ressembler à ceci

```
#remplacer la chaine par le chemin java
export JAVA_HOME="chemin java"
export PATH=$PATH:$JAVA_HOME/bin

#remplacer la chaine par le chemin spark
export SPARK_HOME="chemin spark"
export PATH=$PATH:$SPARK_HOME/bin

#remplacer la chaine par le chemin python
export PYSPARK_PYTHON="chemin python"
export PYSPARK_DRIVER_PYTHON=code
export PYSPARK_DRIVER_PYTHON_OPTS=notebook
```

Pour finir validez les modifications apportées et lancer le serveur Spark qui ouvrira l'éditeur Vscode

```
pyspark
```

ensuite lancer le serveur jupyter si l'éditeur ne l'a pas fait

```
jupyter-notebook --no-browser
```

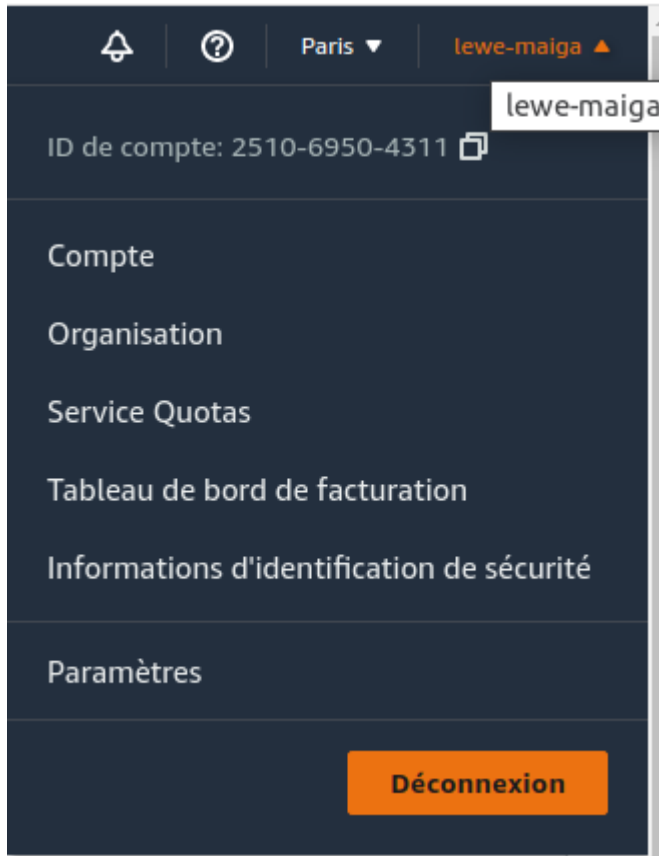
AWS

Une fois l'environnement de developpement installé et fin prêt nous nous attaquons à present à la base de donnée DynamoDB, malgré le fait qu'il soit possible de l'installer localement ([consulter la documentation](#)) le moyen le plus simple serait de creer un compte AWS pour profiter du service gratuitement du moment qu'on ne dépasse pas les limites imposées, sur ce procedons sans plus tarder à l'ouverture du compte

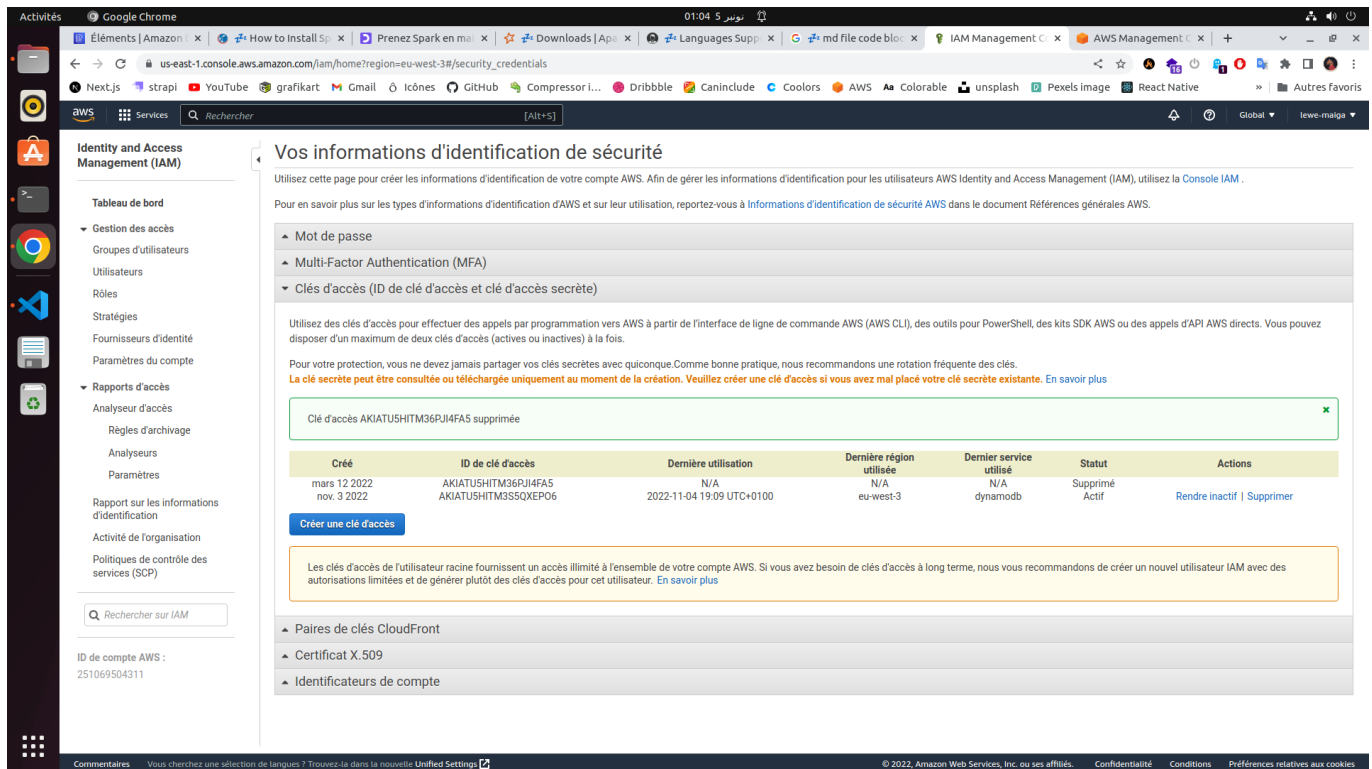
Rendez vous sur le [site officiel](#) et commencer l'inscription je suis sûr que vous pourrez vous débrouiller tout seul

NB: Vous devez renseigner vos informations bancaires mais ne vous inquiétez pas vous n'en aurez pas besoin

Une fois l'inscription terminée, cliquez sur le nom d'utilisateur situé sur le coin en haut à droite de la plateforme



Ensuite accédez aux **informations d'identification de sécurité** dans la partie clés d'accès pour la générer. Vous devez télécharger cette clé car vous en aurez besoin, cliquez simplement sur le message pour le faire



Vos informations d'identification de sécurité

Utilisez cette page pour créer les informations d'identification de votre compte AWS. Afin de gérer les informations d'identification pour les utilisateurs AWS Identity and Access Management (IAM), utilisez la [Console IAM](#).

Pour en savoir plus sur les types d'informations d'identification d'AWS et sur leur utilisation, reportez-vous à [Informations d'identification de sécurité AWS](#) dans le document Références générales AWS.

▲ Mot de passe

▲ Multi-Factor Authentication (MFA)

▼ Clés d'accès (ID de clé d'accès et clé d'accès secrète)

Utilisez des clés d'accès pour effectuer des appels par programmation vers AWS à partir de l'interface de ligne de commande (AWS CLI), des outils pour PowerShell, des kits SDK AWS ou des appels d'API AWS directs. Vous pouvez disposer d'un maximum de deux clés d'accès (actives ou inactives) à la fois.

Pour votre protection, vous ne devez jamais partager vos clés secrètes avec quiconque. Comme bonne pratique, nous recommandons une rotation fréquente des clés.

La clé secrète peut être consultée ou téléchargée uniquement au moment de la création. Veuillez créer une clé d'accès si vous avez mal placé votre clé secrète existante. [En savoir plus](#)

Clé d'accès AKIATUSHITM36PJ4FA5 supprimée

Créé	ID de clé d'accès	Dernière utilisation	Dernière région utilisée	Dernier service utilisé	Statut	Actions
mars 12 2022 nov. 3 2022	AKIATUSHITM36PJ4FA5 AKIATUSHITM3SSQXEP06	N/A 2022-11-04 19:09 UTC+0100	N/A eu-west-3	N/A dynamodb	Supprimé Actif	Rendre inactif Supprimer

[Créer une clé d'accès](#)

Les clés d'accès de l'utilisateur racine fournissent un accès illimité à l'ensemble de votre compte AWS. Si vous avez besoin de clés d'accès à long terme, nous vous recommandons de créer un nouvel utilisateur IAM avec des autorisations limitées et de générer plutôt des clés d'accès pour cet utilisateur. [En savoir plus](#)

▲ Paires de clés CloudFront

▲ Certificat X.509

▲ Identificateurs de compte

Installation de l'invité de commande AWS

Une fois la clé en votre possession nous allons procéder à l'installation de l'invité de commande d'Amazon afin d'accéder aux services depuis le terminal.

```
# Valider les informations du profil
source ~/.profile

#Installer le cli à l'aide de pip
pip3 install awscli --upgrade --user

# Emplacement du cli
which aws

# Verifier la version du cli
aws --version
```

Configuration de l'environnement AWS

L'installation de l'invité de commande terminée, configurons tout ça. Commençons par afficher le contenu de la clé

```
cat rootkey.csv
```

Nous sommes fin prêt pour debuter la configuration d' `awscli`

```
aws configure
```

Selon votre localisation géographique, choisissez `eu-west-1` (Europe ou Afrique), `us-east-1`, `us-west-1` ou `ap-southeast-1` comme nom de région. Utilisez `json` comme "Default output format", ce qui indique que vous souhaitez obtenir des réponses de l'API au format JSON.

Vous pouvez vérifier que `awscli` est bien configuré en exécutant `aws s3 ls` qui devrait répondre... rien du tout ! (pas de message d'erreur en tout cas)

Installer aussi `boto3` qui est le sdk d'AWS pour python

```
pip install boto3
```

Ok, vous êtes équipés pour interagir avec les services sur AWS. C'est parti !

```
pyspark && jupyter-notebook --no-browser
```

L'environnement de travail est prêt, à present clonez le projet et executer les notebooks

```
git clone https://github.com/lewe-maiga/spark-and-dynamodb.git
```

