Rakib SHEIKH
Asano - ESGI 3 IABD - Machine Learning for Information Access
rsheikh1@myges.fr | noobzik@pm.me

TD/TP 1 : Introduction

Pour rappel, le Github est disponible à cette adresse : https://github.com/Noobzik/TP_NoSQL

Je teste actuellement une nouvelle méthode de cours en provence des université qui est la pédagogie inversé. Cela consiste à vous faire pratiquer, puis en fin de semaine, vous faire un rappel de tout ce que vous avez pu voir.

Ce TD/TP sert d'introduction pour la mise en place d'un serveur MongoDB sur vos PC via docker. Elle permet aussi d'introduire Mongo Atlas.

- Commencer par une introduction à la plateforme de Mongo Atlas.
- Par la suite, mettre à jour le répertoire github que vous avez du téléchargé pour le TP de Cassandra.
- Ensuite, nous allons devoir lancer une instance docker à l'aide de la commande `docker compose up`.

1. Prise en main de Mongo Atlas

1. 1. Mongo Atlas

MongoDB Atlas est un service d'inforgérence qui permet de manager un cluster de mongodb dans le cloud. Elle permet d'éviter de gérer la partie infrastructure de ce dernier. Il est possible de choisir dans quel infrastructure de cloud l'on peut déployer entre Google Cloud Platform, Amazon Web Service et Microsoft Azure.

Nous avons le choix entre 3 instance à savoir :

- M10 et M0 qui sont des instances de serveurs fournis. Le M10 a une capacité de 10Gb et 2Gb de Ram avec 2vCPU, elle est facturée $0,08/h. Le M0 est dédié aux environnement d'apprentissage et est gratuit. Elle possède cependant 512mb de stockage. Un cluster est plusieurs serveurs MongoDB travaillant ensemble. Cependant vous avez le choix entre un cluster dédié ou un cluster partagé. Dans l'idée, les clusters partagés sont plus petit comme le M0 et est gratuit pour toujours. Tandis que le cluster dédié à une fonctionnalité qui permet d'avoir une scalabilité horizontale basé sur la charge de travail qui lui est affecté.
- Serverless qui est payant en fonction des ressources utilisées. Elle est fortement utile pour les charges de travail très variable ou éparses. C'est un bon choix lorsque vous ne savez pas quel sont les ressources utilisés.

Atlas dispose des informations opérationnel, un système de point de sauvegarde / d'archivage en ligne, et d'un CLI et une API pour l'administration à distance.

Elle comprend également un ensemble de service tel que Atlas Search, qui est un moteur de recherche basé sur des clé valeur, Atlas Stream Processing qui est une façon native de MongoDB de traiter des données en continue, (soit en temps réel) pour en citer quelque exemple.

1. 1. 1. Question : Parmi les éléments suivants, quel est celui que vous pouvez faire avec MongoDB Atlas ? (Sélectionnez tous ceux qui s'appliquent.)

1. Stocker vos données avec le service mondial multi-cloud de MongoDB.
2. Ajouter une fonctionnalité de recherche à votre application, comme une barre de recherche.
3. Écrire et héberger une application complète dans un environnement cloud géré.
4. Effectuer des requêtes sur plusieurs clusters Atlas pour obtenir une vue globale de vos données.
5. Aucune de ces réponses.

1. 1. 2. Question : Lesquelles des affirmations suivantes sont vraies à propos de MongoDB Atlas Clusters ? (Sélectionnez toutes les réponses qui s'appliquent.)

1. Un cluster d'Atlas MongoDB est un groupe de serveurs connectés via un réseau qui contient des copies de vos données.
2. Une fois que vous avez installé et configuré votre cluster MongoDB Atlas, vous ne pouvez plus modifier le fournisseur de cloud, la région ou le niveau de cluster.
3. Les clusters MongoDB Atlas peuvent être déployés à l'échelle mondiale dans une seule région géographique ou dans plusieurs régions géographiques, en fonction du niveau de cluster.
4. Les clusters dédiés permettent d'accéder à toutes les fonctionnalités d'Atlas.
5. Aucune de ces réponses.

1. 1. 3. Activité pratique

1. Connectez-vous sur Mongo Atlas à l'adresse suivante : https://cloud.mongodb.com/
2. Cliquez sur le bouton Create
3. Sélectionnez le cluster M0 ayant comme nom ESGI
4. Laissez les paramètres par défaut mais gardez les informations de connexion ! Elle ne seront qu'affiché qu'une fois ! Cliquez sur Create Database User
5. Dans la méthode de connexion, nous allons choisir Atlas SQL et cliquez sur Done.
6. Chargez les données par défaut avec Load Sample Data.
7. Maintenant, on va pouvoir supprimer notre cluster, cliquez sur Databse ensuite sur les trois petits points situé à droite du bouton Browse Collections, et enfin sur Terminate.

Remarque Le temps de la mise en oeuvre du projet peut prendre un certain temps.

2. Prise en main de mongosh

2. 1. La partie Docker

1. Commencez tout d'abord par mettre à jour le répertoire Github du cours de NoSQL.
2. Lancez le docker-compose qui se trouve dans le dossier MongoDB.
3. Connectez-vous depuis le terminal à votre instance Docker à l'aide de la commande suivante : `docker run -it mongodb-mongodb1 mongosh`
4. Saisissez la commande `db.help()` et prennez connaissance des commandes qui y existe.

2. 2. Début de manipulation de commandes

1. Quel commande permet de retourner le nom de la base de donnée ?
2. Quel commande permet de retourner le nom de tout les collections contenus dans la base de donnée actuelle ?
3. Que se passe-t-il si l'on créer un utilisateur en doublon ?

2. 2. 1. Création d'un utilisateur

Il est important de noter la différence entre `root` et `admin`, Pour cela, le role `userAdminAnyDatabase` permet de créer des utilisateurs et de leur assigner des roles. En revanche, à part pour la création des utilisateurs, elle ne peut pas réaliser d'autre action. Le role supérieur à celle-ci est `root`.

Nous allons créer un utilisateur dans notre base de donnée mongo. Mais pour cela, nous devons switcher sur une base de donnée `admin`. Pour cela :

1. Utilisez la commande `use admin` pour switcher vers la base de donnée `admin`
2. Testez la commande suivante `db.adminCommand({listDatabase: 1})` : Que remarquez vous ?
3. Utilisez alors la commande `db.auth()` qui prend deux argments, user et `pwd`.
4. Créez un utilisateur mongo-admin . Elle aura pour `roles` `[{role: "userAdminAnyDatabase", db:"admin"}]`

En exemple de syntaxe :

```
{
    user: "mongo-admin",
    pwd: "passw0rd",
    roles: [ { role: "userAdminAnyDatabase", db: "admin" } ]
  }
```

3. Créez un utilisateur mongo-admin . Elle aura pour role `root`.
4. Listez l'ensemble des bases de données qui sont disponible dans votre cluster local.

```
{
  databases: [
    { name: 'admin', sizeOnDisk: Long('102400'), empty: false },
    { name: 'config', sizeOnDisk: Long('12288'), empty: false },
    { name: 'local', sizeOnDisk: Long('73728'), empty: false }
  ],
  totalSize: Long('188416'),
  totalSizeMb: Long('0'),
  ok: 1
}
```

5. Maintenant, ce que nous allons faire, c'est de créer un utilisateur sur une nouvelle base de donnée qui va avoir l'autorisation de lire et d'écrire dans les base de donnée concernées.

   Mais avant, créez deux base de données nommés db1 et db2 et injectez les données que vous souhaitez disponible à l'adresse suivante : https://github.com/neelabalan/mongodb-sample-dataset ,le tout à l'aide de Mongo Compass. Ensuite vous pouvez créer un compte my-user qui aura accès à ces deux bases de données.

   - Téléchargez et installez Mongo Compass dans le lien suivant : https://www.mongodb.com/products/tools/compass
   - Lancez Mongo Compass et surtout, n'allez pas vite sur la page de connexion. En effet, nous avons fixé dans le docker compose, les identifiants de connexion pour accéder à la base de donnée. Par défaut, c'est `root/example`.
   - Pour importer une base de donnée à partir d'un fichier, nous pouvons cliquer sur le signe + sur la droite de `Database` et de suivre simplement les instructions demandé. Respectez cependant la manière dont c'est présenté dans le Github, c'est à dire qu'une collection est représenté par un dossier, et que l'ensemble des documents est représenté par un fichier json ou bson. Autrement dit :
   - Remplacez `db1` et `db2` par les noms de dossier que vous avez choisi d'importer et précisez dans le rapport à quoi correspond ces deux `db`.

Nous pouvons maintenant procéder à la création du compte utilisateur.

Pour cela :
- user: "my-user"
- pwd: Libre
- roles : [{roles: "readWrite", db: "db1"}, {role: "read", db:"db2"}]

6. Vérifiez bien que cet utilisateur peut bien lire et écrire dans db1 et uniquement lire dans db2. Autrement dit, à vous de chercher comment insérer des données à partir de Mongo Compass.

7. Pour supprimer un utilisateur, il suffit d'appeler la fonction `dropUser("l'utilisateur")`. Supprimez l'utilisateur "my-user"

Nous sommes arrivé à la fin de TP introductive, vous avez vu
- Le fonctionnement de Atlas.
- Le fonctionnement très brève de Mongo Compass qui sera rappelé dans le prochain TP.

- Certaines commandes liée à l'administration.
- Le fonctionnement de mongosh