Séance pratique MongoDB

TD 3IF, Mars 2019, Előd Egyed-Zsigmond

## Prérequis :

* MongoDB server Community, v3.x installé téléchargeable depuis : (https://www.mongodb.com/download-center?jmp=docs#community).
* Un traitement de texte pour écrire du code non auto correcteur (type Notepad++, SublimeText, emacs, Notepad, Eclipse, …) mais pas MS Word, Libre Office, Google Doc qui modifient le texte que vous écrivez sans vous le dire)
* Le client MongoDB : Studio3T (https://studio3t.com/download/)

## Préparation

Nous allons créer une base contenant des informations sur les stations Vélo’V.

Vérifiez que le serveur Mongodb tourne (sinon lancez-le : lancer un invité de commande ou shell (sous Windows : touche Windows + R, puis écrire : cmd.exe, se placer dans le répertoire bin de votre installation mongo et lancez l’exécutable mongod).

Se connecter à la base MongoDB : et lancez le client mongodb dans un autre invité de commande, avec l’instruction : mongo

Créer la base tpMongo avec l’instruction

use tpMongo

Créer la collection velov\_geo : db.createCollection('velov\_geo');

Créer le document suivant : {

"type" : "Feature",

"properties" : {

"lat" : 45.7873840849558,

"available\_bike\_stands" : NumberInt(7),

"gid" : NumberInt(1065),

"last\_update" : "2017/02/02 14:49:23",

"number" : NumberInt(9013),

"address2" : "",

"nmarrond" : NumberInt(13),

"pole" : "pôle d'activité (Cegid, Infogrammes)",

"bike\_stands" : NumberInt(20),

"banking" : NumberInt(0),

"address" : "Quai Paul Sédallon",

"lng" : 4.81437377940006,

"availabilitycode" : NumberInt(1),

"availability" : "Vert",

"name" : "Sédaillan / Naviguation",

"commune" : "Lyon 9 ème",

"bonus" : "Non",

"status" : "OPEN",

"available\_bikes" : NumberInt(13),

"last\_update\_fme" : "2017/02/02 14:54:00"

},

"geometry" : {

"type" : "Point",

"coordinates" : [

4.814373779400059,

45.787384084955846

]

}

}

Insérer le document dans la collection velov\_geo  avec

db.velov\_geo.save(...);

Créer les fiches json de deux autres stations Vélo’V (allez sur la page : https://velov.grandlyon.com/mapping ) et insérez les dans la base.

Charger plus de données dans la base. Pour cela téléchargez directement le fichier : [velov\_geo.json](http://liris.cnrs.fr/eegyedzs/Cours/CentraleNoSQL/velov_geo.json) inclus dans l’archive de ce sujet.

(Ces données étaient disponibles sur le site https://data.grandlyon.com/ plus précisément : https://download.data.grandlyon.com/ws/ldata/velov.stations/all.json jusqu’à l’an dernier, aujourd’hui l’accès est plus riche pour les utilisateurs mais l’information est moins simple à collecter)

## Charger le fichier json dans la base

La manière la plus simple est de copier velov\_geo.json dans le répertoire bin de votre installation mongodb.

Lancez une fenêtre de ligne de commande/shell et placez-vous dans le répertoire bin de votre installation mongodb. Dans ce répertoire se trouve un fichier exécutable mongoimport (mongoimport.exe sous Windows).

Lancez l’instruction (pas depuis le client mongo!! Mais bien depuis l’invité de commande du système d’exploitation) :

mongoimport --db tpMongo --collection velov\_geo --file velov\_geo.json

Cet instruction crée la base tpMongo si elle n’existe pas, la collection velov\_geo et y insère les documents contenus dans le fichier velov\_geo.json.

Vous pouvez exécuter mongoimport depuis n’importe où à condition que le chemin vers l’exécutable mongoimport soit enregistré dans la variable d’environnement PATH

## Interrogation simple

Exprimez des requêtes simples pour les recherches suivantes :

1. Liste de toutes les stations dont la commune est VILLEURBANNE
2. Lister toutes les stations qui ne sont pas à VILLEURBANNE
3. Liste des stations qui n’ont pas de seconde adresse (adress2 vide)
4. Nombre de stations dont la seconde adresse a une valeur
5. Lister les stations avec plus de 2 vélos disponibles (available\_bikes)
6. Lister toutes les communes distinctes
7. Lister toutes les communes distinctes, trier les résultats par ordre alphabétique
8. Lister les stations du 9ème arrondissement (Lyon 9 ème) par ordre croissant de vélos disponibles
9. Projeter les résultats sur le nom, l’adresse et le nombre de vélos disponibles
10. Afficher les communes et le nombre de stations Vélo’V par commune
11. Triez par commune
12. Triez par nb de stations en ordre décroissante
13. Nombre de stations Vélo’V de Villeurbanne groupés par nombre de vélos disponibles triés selon ce nombre
14. Rajouter au résultat précédent, les noms des stations pour chaque nombre de vélos disponible
15. Calculer le nombre moyen de vélos disponibles par commune

## Requêtes géographiques

Préciser que les coordonnées sont des données géographiques en créant un index 2dsphere :

use tpMongo;

db.velov\_geo.createIndex( { "geometry.coordinates" : "2dsphere" } )

Vérifiez que l’index a bien été créé avec :

db.velov\_geo.getIndexes() ;

1. Chercher les stations Vélo’V qui sont à moins de 500m du point : $geometry**:** **{** type**:** "Point"**,** coordinates**:** **[** 4.863132722360224**,** 45.77022676914935 **]** **}** (utilisez : $near)
2. Listez les 5 stations les plus proches des coordonnées type **:** "Point"**,** coordinates**:** **[** 4.863132722360224**,** 45.77022676914935 **] }** (utilisez : geoNear)

## Requêtes textuelles

Préciser que le champ pole est de type texte et créez un index sur ce champ

db.velov\_geo.createIndex**(** **{** "properties.pole"**:** "text" **}** **,** **{** default\_language**:** "french" **})**

1. Cherchez les stations dont le pole répond à la requête « quartiers »