Concepts : Bases de données - collections de documents
Outils : MongoDB

Ce TP illustre l'utilisation d'une BD dite *NoSQL*. MongoDB se présente comme un support de persistance pour des donnés organisés sous forme de collections de documents. La structure des données reste flexible. Les documents sont décrits en *BSON* (Binary *JSON*) une extension de *JSON* (JavaScript Object Notation).

Exercice 1 : Démarrer MongoDB.

- Créer un répertoire de travail pour Mongo (par exemple mkdir dbmongo).
- Lancer un serveur MongoDB (mongod --shardsvr --dbpath dbmongo --port 27021)
- Dans une autre fenêtre de commande lancer un client mongo (mongo --host localhost:27021)

Exercice 2 : Créer une collection et insertion de documents

Utiliser le fichier datasetRestaurants.json pour importer les documents dans la BD dans une collection nommé restaurants.
Dans une autre fenêtre de commande exécuter la commande mongoimport :

```
mongoimport --host localhost:27021 --db test --collection restaurants --drop --file ~/datasetRestaurants.json
```

Les documents de la collection « restaurants » ont comme minimum les attributs suivants :

```
{
   "address" : {
      "building" :  "----",
      "coord" :  [ -- , --],
      "street" :  "----",
      "zipcode" : "----"
   },
   "cuisine" : "----",
   "grades" : [
      {
         "date" : ISODate("----"),
         "type" : "----",
         "score" : --
      }
 …
   ],
   "name" : "----",
   "restaurant id" : "----"
}
```

*** "id restaurant" référence un document dans la collection "restaurants".

Exercice 3 : Interroger une collection

1. Lister tous les restaurants en les triant par ordre alphabétique.
2. Trouver les restaurant avec de la cuisine italienne (« Italian ») et donner pour chacun d'eux son nom, le code postal et ses coordonnées.
3. Trouver les restaurants ayant pour code postal supérieur à 10075 qui servent de la cuisine italienne et dont le numéro de téléphone est renseigné.
4. Trouver les restaurants avec de la cuisine italienne ou un code postal de 10075.

Exercice 4 : Mises à jour

1. Changer le type de cuisine ”Other” en ”Cuisine to be determined” pour les restaurants dont le code postal est ”10016”.
2. Modifier le restaurant dont l'identifiant (restaurant_id) est ”41154403” avec les informations suivantes :
   "name" : "Vella 2",
   "address":{ "city":"1480","street":"2Avenue","zipcode":"10075"}

Exercice 5 : Requêtes avec agrégation

1. Combien de restaurants pour chaque type de cuisine ?
2. Combien de restaurant avec de la cuisine italienne pour chaque code postal

Exercice 6 : Manipuler plusieurs collections

1. Créer une nouvelle collection comments.
2. Insérer 6 documents conforme à l'structuration ci-dessous (le type peut être positive ou négative). Pour ces 6 documents, utiliser le champ restaurant_id présent dans la collection contenant les documents relatifs aux restaurants. Faire en sorte que plusieurs commentaires portent sur le même restaurant et que plusieurs commentaires proviennent d'un même client, avec au moins un client commentant plusieurs fois le même restaurant.

```
{
        "_id" :"...",
        "restaurant id" :"...",
        "name client" :"...",
        "comment" : "...",
        "type" : "...",
}
```

3. Trouver le nom des restaurants ayant des commentaires provenant d'un client particulier.

Exercice 7 : Index

1. Utiliser la méthode .explain(*{*"executionStats":1*}*) sur la requête permettant de trouver tous les restaurants ayant "cuisine"="Italian". Observer les différentes statistiques.
2. Créer un index sur les champs cuisine de la collection des restaurants.
3. Etudier le nouveau plan d'exécution pour la requête précédente et comparer les performances des plans d'exécutions.
4. Créer d'autres index pour optimiser des requêtes plus complexes.