Requêtes MongoDB depuis R et Python

Corrigé de TD pour un cours dispensé à l'université de Rennes 2

Romain Tavenard

Dans ce TD, vous allez effectuer des requêtes classiques à une base MongoDB depuis des scripts. Seront abordés ici deux langages:

- les scripts R vous permettront de récupérer les résultats de vos requêtes sous forme de *dataframe* dans R pour ensuite y appliquer vos traitements statistiques: un utilisera pour cela la librairie `mongolite`;
- les scripts Python vous permettront de récupérer les résultats de vos requêtes sous forme de variable Python: on utilisera pour cela la librairie `pymongo` qui présente l'avantage d'être maintenue par les développeurs de MongoDB (ce qui garantit, a priori, une certaine pérennité et une cohérence avec l'interface MongoDB).

Ainsi, pour chaque manipulation de cet énoncé, il est demandé d'effectuer le travail dans chacun de ces deux langages.

1 Connexion à la base de données

1. Connectez vous à la base `food` hébergée sur le serveur MongoDB Atlas dont l'URL est `clusterm1.0rm7t.mongodb.net`.

Attention: ci-dessous les identifiants et mot de passe sont inscrits en clair pour que vous puissiez tester ces bouts de code chez vous, mais il s'agit d'une très mauvaise pratique : ceux-ci doivent toujours être lus dans un fichier externe de manière à ce que vous puissiez partager votre code sans révéler vos identifiants.

```
from pymongo import MongoClient

db_uri = "mongodb+srv://etudiant:ur2@clusterm1.0rm7t.mongodb.net/"
client = MongoClient(db_uri)
db = client["food"]
```

2. Affichez la liste des collections de la base (ceci n'est pas possible en R avec `mongolite`).

```
print(db.list_collection_names())
```

3. Affichez la liste des index de la collection `NYfood`.

```
coll = db["NYfood"]
print(coll.index_information())
```

2 Requêtes de lecture

4. Affichez la liste des restaurants de Manhattan dont le nom commence par `A`.

```
query = {"borough": "Manhattan", "name": {"$regex": "^A"}}
cursor = coll.find(query)
print(list(cursor)[:10])
```

5. Combien de résultats comporte cette liste ?

```
print(cursor.count())
```

6. Affichez le résultat de la fonction `explain()` pour cette requête (ceci n'est pas possible en R avec `mongolite`).

```
print(coll.find(query).explain())
```

7. Reprenez la requête précédente et n'affichez que les 5 premiers résultats.

```
for doc in coll.find(query).limit(5):
    print(doc)
```

8. Même chose en ayant trié les résultats par ordre alphabétique inverse du nom de restaurant.

```
for doc in coll.find(query).sort("name", -1).limit(5):
    print(doc)
```

9. Affichez la liste des notes attribuées à des restaurants de `Manhattan`. À l'aide d'une boucle, stockez, pour chaque note attribuée, le nombre de restaurants qui ont reçu cette note au moins une fois. Visualisez ces données sous la forme d'un diagramme en bâton.

```
import seaborn as sns
import matplotlib.pyplot as plt

cursor = coll.distinct("grades.grade",
                       {"borough": "Manhattan"})
possible_grades = list(cursor)
counts = []
for grade in possible_grades:
    counts.append(
        coll.count_documents(
            {"grades.grade": grade,
             "borough": "Manhattan"}
```

```
        )
    )
```

```
sns.barplot(x=possible_grades, y=counts)
plt.show()
```

10. Affichez la liste des notes existant dans la base.

```
print(coll.distinct("grades.grade"))
```

11. Affichez la liste des restaurants ayant au moins une note postérieure au 20 janvier 2015.

```
cursor = db.NYfood.find(
    {"grades.date": {"$gte": datetime.datetime.strptime("2015/01/20", "%Y/%m/%d")}}
)
print(list(cursor)[:10])
```