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.

```
library("mongolite")

mdb = mongo(collection="NYfood", db="food",
            url="mongodb+srv://etudiant:ur2@clusterm1.0rm7t.mongodb.net/",
            verbose=TRUE)
```

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

```
print(mdb$index())
```

2 Requêtes de lecture

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

```
q = '{"borough": "Manhattan", "name": {"$regex": "^A", "$options" : "i"}}'
print(mdb$find(query = q))
```

5. Combien de résultats comporte cette liste ?

```
print(mdb$count(query= q))
```

6. Affichez le résultat de la fonction `explain()` pour cette requête (ceci n'est pas possible en R avec `mongolite`).
7. Reprenez la requête précédente et n'affichez que les 5 premiers résultats.

```
print(mdb$find(query = q, limit = 5))
```

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

```
print(mdb$find(query = q, limit = 5, sort = '{"name": -1}'))
```

9. Affichez la liste des notes attribuées à des restaurants de `Manhattan`. En R, réalisez un test statistique pour vérifier l'hypothèse selon laquelle les notes des restaurants suivent la même distribution dans les quartiers de Manhattan et Brooklyn.

```
q = '[{"$match": {"borough": "Manhattan"}},
      {"$project": {"notes": "$grades.grade"}},
      {"$unwind": "$notes"}]'
grades_Manhattan = mdb$aggregate(pipeline = q)$notes

q = '[{"$match": {"borough": "Brooklyn"}},
      {"$project": {"notes": "$grades.grade"}},
      {"$unwind": "$notes"}]'
grades_Brooklyn = mdb$aggregate(pipeline = q)$notes

test = chisq.test(table(grades_Brooklyn), table(grades_Manhattan))
print(test)
```

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

```
print(mdb$distinct(key = "grades.grade"))
```

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

```
q = '{"grades.date": {"$gte": {"$date": "2015-01-20T00:00:00Z"}}}'
print(mdb$find(query=q))
```