Index

Romain Tavenard

1 Création d'une base fictive

Ouvrez un client MongoDB et exécutez la ligne suivante, qui va créer une base appelée `large_db` et sa collection `users` contenant `n` (ici 500000) utilisateurs :

```
> use large_db
> n = 500000
> for (i=0; i<n; i++) {
    db.users.insert({"i": i, "name": "user"+i,
                     "age": Math.floor(Math.random()*120),
                     "created" : new Date()})
    }
```

L'exécution de cette commande peut prendre plusieurs minutes.

1. Surveillez sa progression en ouvrant, dans un autre terminal, un second client MongoDB (connecté au même serveur) et lui demandant d'afficher le nombre d'éléments de la collection `users` de la base `large_db`.

```
> use large_db
> db.users.count()
```

2 Sans index

MongoDB permet d'obtenir un certain nombre d'informations sur le déroulement d'une requête à l'aide de la fonction `explain()` :

```
> db.nomDeLaCollection.find({...}).explain("executionStats")
```

2. Utilisez cette méthode pour obtenir (i) le temps d'exécution, (ii) le nombre de documents parcourus et (iii) le nombre de documents retournés par une simple requête demandant d'afficher les informations des utilisateurs dont la clé `"name"` vaut `"user101"`.

```
> db.users.find({name: "user101"}).explain("executionStats")
```

3. Même chose en ne demandant de retourner qu'un utilisateur. Que remarquez-vous ?

```
> db.users.find({name: "user101"}).limit(1).explain("executionStats")
```

Le nombre de documents parcourus a chuté : une fois un premier document trouvé, la requête s'arrête.

4. Répétez encore cette opération en cherchant maintenant l'utilisateur `"user499999"`. Que remarquez-vous ?

Dans ce cas, il faut parcourir tous les documents avant de finir par trouver le bon.

3 Avec un index

5. Mettez maintenant en place un index sur la clé `"name"`.

```
> db.users.createIndex({"name" : 1})
```

6. Répétez les opérations précédentes (questions 2 à 4). Que remarquez-vous ?

Le nombre de documents parcourus pour effectuer la requête est maintenant égal à 1 pour chacune des trois requêtes : il a suffi de chercher les valeurs de `"name"` dans l'index pour résoudre la requête.

4 Index composés

7. Effectuez les requêtes suivantes et examinez leur exécution :

— Afficher les utilisateurs âgés de 20 ans ;
— Afficher les utilisateurs âgés de 20 ans par ordre croissant de nom d'utilisateur ;
— Afficher les utilisateurs âgés de 20 ans par ordre croissant de date de création ;
— Afficher les utilisateurs âgés de 20 ans dont le nom d'utilisateur est compris entre `"user100000"` et `"user500000"`.

```
> db.users.find({"age": 20}).explain("executionStats")
> db.users.find({"age": 20}).sort({"name":1}).explain("executionStats")
> db.users.find({"age": 20}).sort({"created":1}).explain(
    "executionStats")
> db.users.find({"age": 20, "name":
    {$gt: "user100000", $lt: "user500000"}}).explain("executionStats")
```

8. Mettez en place un index composé sur les clés `"age"` et `"name"` (dans cet ordre).

```
> db.users.createIndex({"age":1, "name":1})
```

9. Répétez les requêtes de la question 7. Quelles sont les requêtes qui ont été accélérées ?

Elles ont toutes été accélérées grâce au préfixe `"age"` du nouvel index. Pour la requête triant sur la date de création, le temps d'exécution reste un peu plus important que pour les autres car il faut trier un nombre important de résultats.

5 Exercice de synthèse

Liens utiles :

— http://docs.mongodb.org/master/core/2dsphere/
— http://docs.mongodb.org/master/core/index-text/
— http://docs.mongodb.org/master/core/index-single/

10. Passez à la base `food`. Créez sur la collection `NYfood` de cette base quatre index :

— un index géospatial sur la localisation des restaurants ;
— un index textuel couvrant tous les champs textuels de la collection ;
— un index classique sur le quartier et un autre sur le type de cuisine.

Vérifiez l'efficacité de ces index à l'aide de quelques requêtes simples (éventuellement issues des TD précédents).

```
> db.NYfood.createIndex({"address.loc" : "2dsphere"})
> db.NYfood.createIndex({"$**" : "text"})
> db.NYfood.createIndex({"borough" :1})
> db.NYfood.createIndex({"cuisine" :1})
```