

Projet de BigData

Les bases NoSQL

par Jordan Baudin, Corentin LeGuen et Geoffrey Spaur

11 janvier 2018

Contents

1	Présentation	3
2	La base de donnée CouchBase	4
2.1	Les modalités d'installation	4
2.2	Les méthodes d'insertion de données	4
2.3	Le langage de recherche	4
2.4	L'indexation interne	4
2.5	Le support de la concurrence d'accès	4
2.6	L'architecture du système	4
2.7	Les techniques de distribution	4
3	La base de donnée MongoDB	5
3.1	Les modalités d'installation	5
3.2	Les méthodes d'insertion de données	6
3.3	Le langage de recherche	7
3.4	L'indexation interne	7
3.5	Le support de la concurrence d'accès	9
3.6	L'architecture du système	9
3.7	Les techniques de distribution	9
4	Conclusion	10

1 Présentation

Ce projet a pour but de comparer différentes bases de données NoSQL. Nous allons prendre les bases données CouchBase et MongoDB.

2 La base de donnée CouchBase

2.1 Les modalités d'installation

Recommandations système

CPU	6 coeurs 64-bits à 3GHz	TODO
RAM	16 GB physique	

2.2 Les méthodes d'insertion de données

TODO

2.3 Le langage de recherche

TODO

2.4 L'indexation interne

TODO

2.5 Le support de la concurrence d'accès

TODO

2.6 L'architecture du système

TODO

2.7 Les techniques de distribution

TODO

3 La base de donnée MongoDB

3.1 Les modalités d'installation

Recommandations système Il est recommandé par la documentation d'utiliser des processeurs 64-bits car sinon, la mémoire sera limitée à 2 GB. Les performances du serveur sont fortement dépendantes de la RAM du système.

Téléchargement Pour l'installation de MongoDB, il vous suffira dans un premier temps de télécharger l'archive contenant MongoDB.

Après le téléchargement de l'archive, vous pourrez la décompresser. Avant de lancer le serveur, il vous faudra créer le dossier **/data/db**. Ce dossier contiendra toutes les bases de MongoDB.

Enfin vous pouvez lancer le serveur MongoDB avec la commande:

```
$ ./bin/mongod
```

```
geoffrey@Debian:~/git/s9/BD/mongodb-linux-x86_64-debian81-3.6.1$ ./bin/mongod
2018-01-11T14:28:55.444+0100 I CONTROL [initandlisten] MongoDB starting : pid=2409
port=27017 dbpath=/data/db 64-bit host=Debian
2018-01-11T14:28:55.444+0100 I CONTROL [initandlisten] db version v3.6.1
2018-01-11T14:28:55.444+0100 I CONTROL [initandlisten] git version: 025d4f4fe61efd1fb6f0005be20cb45a004093d1
2018-01-11T14:28:55.444+0100 I CONTROL [initandlisten] OpenSSL version: OpenSSL 1.0.1t 3 May 2016
2018-01-11T14:28:55.444+0100 I CONTROL [initandlisten] allocator: tcmalloc
2018-01-11T14:28:55.444+0100 I CONTROL [initandlisten] modules: none
2018-01-11T14:28:55.444+0100 I CONTROL [initandlisten] build environment:
2018-01-11T14:28:55.444+0100 I CONTROL [initandlisten] distmod: debian81
2018-01-11T14:28:55.444+0100 I CONTROL [initandlisten] distarch: x86_64
2018-01-11T14:28:55.444+0100 I CONTROL [initandlisten] target_arch: x86_64
2018-01-11T14:28:55.444+0100 I CONTROL [initandlisten] options: {}
2018-01-11T14:28:55.508+0100 I - [initandlisten] Detected data files in /data/db created by the 'wiredTiger' storage engine, so setting the active storage engine to 'wiredTiger'.
2018-01-11T14:28:55.508+0100 I STORAGE [initandlisten]
2018-01-11T14:28:55.508+0100 I STORAGE [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine
2018-01-11T14:28:55.508+0100 I STORAGE [initandlisten] ** See http://docs.mongodb.org/core/prodnotes-filesystem
2018-01-11T14:28:55.524+0100 I STORAGE [initandlisten] wiredtiger_open config: create,cache_size=1427M,session_max=20000,eviction=(threads_min=4,threads_max=4),config_base=false,statistics=(fast),log=(enabled=true,archive=true,path=journal,compressor=snappy),file_manager=(close_idle_time=100000),statistics_log=(wait=0),verbose=(recovery_progress),
2018-01-11T14:28:56.039+0100 I STORAGE [initandlisten] WiredTiger message [1515677336:39864][2409:0x7fc29f566a00], txn-recover: Main recovery loop: starting at 2/10752
2018-01-11T14:28:56.152+0100 I STORAGE [initandlisten] WiredTiger message [1515677336:152674][2409:0x7fc29f566a00], txn-recover: Recovering log 2 through 3
2018-01-11T14:28:56.269+0100 I STORAGE [initandlisten] WiredTiger message [1515677336:269196][2409:0x7fc29f566a00], txn-recover: Recovering log 3 through 3
2018-01-11T14:28:57.010+0100 I CONTROL [initandlisten]
2018-01-11T14:28:57.010+0100 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2018-01-11T14:28:57.010+0100 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2018-01-11T14:28:57.010+0100 I CONTROL [initandlisten]
2018-01-11T14:28:57.010+0100 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.
2018-01-11T14:28:57.010+0100 I CONTROL [initandlisten] ** Remote systems
```

Vous pourrez ensuite accéder à MongoDB en lançant le client grâce à la commande suivante:

```
$ ./bin/mongo
```

3.2 Les méthodes d'insertion de données

Création de la base Avant toute insertion, lancez le client MongoDB avec la commande précédente. Nous allons ajouter une nouvelle base de donnée. Vous pouvez lister les différentes bases de données déjà présentes avec la commande:

```
> show dbs
```

```
> show dbs
admin      0.000GB
bigdata    0.000GB
config     0.000GB
local      0.000GB
```

Puis pour créer et/ou utiliser notre base de donnée, utilisez:

```
> use DATABASE
```

```
> use bigdata  
switched to db bigdata
```

Vous pouvez lister vos collections de document avec la commande suivante:

```
> show collections
```

```
> show collections  
alcoholism  
articles  
medarticles
```

Insertion de documents Vous pouvez maintenant sortir du client MongoDB. Vos documents doivent être en format json sous forme d'objet. Vous trouverez un exemple avec le fichier *prettyAlcoholism.json*. Nous allons utiliser la commande *mongoimport* afin d'ajouter nos document dans notre base:

```
$ ./bin/mongoimport --db bigdata --collection medarticles  
--file ../Projet/prettyAlcoholism.json
```

```
geoffrey@Debian:~/git/s9/BD/mongodb-linux-x86_64-debian81-3.6.1$ ./bin/mongoimport  
--db bigdata --collection medarticles --file ../Projet/prettyAlcoholism.json  
2018-01-11T15:31:59.658+0100 connected to: localhost  
2018-01-11T15:31:59.858+0100 imported 20 documents
```

Vous pouvez cependant ajouter des document directement à partir du client MongoDB avec les commandes *import* ou *importMany*. Ces commandes sont utiles dans le cas où nous avons peu de document ou des document de petites tailles.

3.3 Le langage de recherche

Pour rechercher des documents dans MongoDB, nous utiliserons la fonction *find()* sur nos collections. Cette commande peut prendre jusqu'à deux arguments sous format JSON:

- Le premier permet de déterminer la clause **where**.
- La seconde permet de spécifier les champs à afficher.

3.4 L'indexation interne

Pour chaque document ajouter dans une collection, MongoDB lui ajoute un champs *_id*. C'est ce champs qui sera indexé par défaut. Il est possible de lister tous les indexes d'une collection avec la commande suivant:

```
> db.medarticles.getIndexes()
```

```
> db.medarticles.getIndexes()
[
  {
    "v" : 2,
    "key" : {
      "_id" : 1
    },
    "name" : "_id_",
    "ns" : "bigdata.medarticles"
  }
]
```

Vous pouvez cependant ajouter vos propres indexes avec la commande *ensureIndex()*. L'objet JSON passé en argument doit être une liste de **int**. Le nom doit être le champs à indexer et le **int** doit être égal à 1 ou -1; la valeur 1 correspond à un tri ascendant et la valeur -1 à un tri descendant.

```
> db.medarticles.ensureIndex({"MedlineCitation.PMID.content":1})
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "ok" : 1
}
```

Afin de mesurer l'impact de vos indexes, à chaque recherche vous pouvez ajouter la méthode *explain*:


```
> db.medarticles.find({"MedlineCitation.PMID.content":29054077}).explain()
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "bigdata.medarticles",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "MedlineCitation.PMID.content" : {
        "$eq" : 29054077
      }
    },
    "winningPlan" : {
      "stage" : "FETCH",
      "inputStage" : {
        "stage" : "IXSCAN",
        "keyPattern" : {
          "MedlineCitation.PMID.content" : 1
        },
        "indexName" : "MedlineCitation.PMID.content_1",
        "isMultiKey" : false,
        "multiKeyPaths" : {
          "MedlineCitation.PMID.content" : [ ]
        },
        "isUnique" : false,
        "isSparse" : false,
        "isPartial" : false,
        "indexVersion" : 2,
        "direction" : "forward",
        "indexBounds" : {
          "MedlineCitation.PMID.content" : [
            "[29054077.0, 29054077.0]"
          ]
        }
      },
      "rejectedPlans" : [ ]
    },
    "serverInfo" : {
      "host" : "Debian",
      "port" : 27017,
      "version" : "3.6.1",
      "gitVersion" : "025d4f4fe61efd1fb6f0005be20cb45a004093d1"
    },
    "ok" : 1
  }
}
```

3.5 Le support de la concurrence d'accès

TODO

3.6 L'architecture du système

TODO

3.7 Les techniques de distribution

TODO

4 Conclusion

TODO