le cnam

Guide pratique d'installation d'un serveur MongoDB

Guide Pratique

CNAM Paris
nicolas.travers (at) cnam.fr

Chapitre 1

Installation de MongoDB

Le logiciel MongoDB [1] peut être installé et utilisé dans un environnement local ou distribué. Il est disponible sur claroline, et également en ligne : `http://www.mongodb.org/downloads/`

Le Travaux Pratique associé à ce guide est dédié à une utilisation en mode Console pour bien intégrer les possibilités de l'API mongoDB et de MapReduce.

Sur Claroline est également disponible 2 documents PDF aidant à l'utilisation de MongoDB :

— MongoDB In Action

— MongoDB in Pratice

Téléchargez et installer le logiciel MongoDB [2] ;

1.1 Windows

— Ouvrir un PREMIER shell **cmd**

— pour connaître l'architecture de son système d'exploitation (32 ou 64 bits)
`> wmic os get osarchitecture`

— Extraire l'archive Mongo

— Le répertoire obtenu sera dénommé ci-dessous par **$MONGO** (si vous écrivez tel quel $MONGO, rien ne se passera ou une erreur sera levée)

(1) **Création du répertoire de données**

```
> cd C:\
> md data                          (crée un répertoire data)
> md data\db                       (crée un répertoire db dans data)
```

(2) **Lancement du Serveur** : (ne pas fermer la fenêtre sinon, le serveur serait éteint)

— Soit : Double-click sur l'exécutable Mongod

— Soit : `> $MONGO\bin\mongod.exe`

— Soit : `> $MONGO\bin\mongod.exe --dbpath "dossier data"`
(en spécifiant le dossier si c'est différent de c :
data
db)

(3) **Importer un fichier JSon :**

— Ouvrir une console système - cmd (pas la console mongo!!!)

— Télécharger le fichier de données dblp.json.zip

— Décompresser le fichier dans le répertoire de votre choix, dénommé ci-dessous par `$DBLP_FOLDER`

— Importation du fichier de données :
`$MONGO\bin\mongoimport.exe --db dblp --collection publis $DBLP_FOLDER\dblp.json`

— Attendre quelques secondes que les documents soient importées

(4) **Connexion à MongoDB**

— Soit : Lancer un SECOND shell : (cmd), puis `> $MONGO\bin\mongo.exe`

— Soit : Double-click sur l'exécutable `Mongo.exe`

— Soit : Télécharger *robomongo* [3], un logiciel pour manipuler MongoDB

— Sélectionner la base de données : `use dblp`

— Vérifier que les données ont été insérées : `db.publis.count();`

1. `http://www.mongodb.org/`
2. Documentation : `http://docs.mongodb.org/manual/contents/`
3. `http://robomongo.org/`

1.2 Linux et MacOS

— Ouvrir une PREMIERE console : (terminal ou xterm)

— Décompresser Mongo : `> tar zxvf mongodb-xxx.tgz` (xxx :version téléchargée)

— Déplacer Mongo dans un répertoire de votre choix que nous appelerons **$MONGO**Si vous souhaitez que ce chemin soit connu par le système et ne pas avoir à écrire le chemin $MONGO à chaque instruction, il faut ajouter la commande suivante dans le fichier /.bash_profile :
`export PATH=$PATH:$MONGO/bin` (remplacer $MONGO par le chemin complet vers Mongo) (si vous écrivez tel quel $MONGO, rien ne se passera ou une erreur sera levée)
`> mv mongodb-xxx $MONGO`

— Création du répertoire de données /data/db (par défaut, mongodb utilise ce dossier, il faut donc le créer)
```
> sudo mkdir /data
> sudo mkdir /data/db
```
'sudo' permet de donner les droits administrateurs pour l'instruction. Le mot de passe demandé est le votre

(2) **Lancement du Serveur** : (ne pas fermer la fenêtre, le serveur serait éteint)

— Soit : `> sudo $MONGO/bin/mongod`

— Soit : `> sudo $MONGO/bin/mongod --dbpath /votreDossier` (Dossier perso)

(3) **Importer un fichier JSon :**

— Ouvrir une console système - `cmd` (pas la console mongo !!!)

— Télécharger le fichier de données dblp.json.zip

— Décompresser le fichier dans le répertoire de votre choix, dénommé ci-dessous par `$DBLP_FOLDER`

— Importation du fichier de données :
```
$MONGO/bin/mongoimport --db dblp --collection publis $DBLP_FOLDER/dblp.json
```

— Attendre quelques secondes que les documents soient importées

(4) **Connexion à MongoDB**

— Soit : Lancer un SECOND shell : (terminal), puis `> $MONGO/bin/mongo`

— Soit : Télécharger *robomongo*[3], un logiciel pour manipuler MongoDB

— Sélectionner la base de données : `use dblp`

— Vérifier que les données ont été insérées : `db.publis.count();`

1.3 Après l'installation

Une fois que les étapes précédentes ont été réalisées, vous n'aurez pas à refaire l'ensemble à chaque fois. A chaque lancement du TP, il faut lancer :

(2) Lancer le serveur mongod

(4) Lancer la console mongo

Chapitre 2

Interrogation de MongoDB

Dans la console client (voir section précédente - pas *robomongo*), le langage de manipulation de données de MongoDB est inspiré de JavaScript avec des objets. Pour faciliter l'utilisation, il vous est possible de :

— Commandes précédente : *flèche vers le haut*

— Compléter automatiquement une instruction : *tabulation*

— Utiliser des variables

2.1 Gestion de collections

— **Création de la base :**
Une base de données MongoDB est créée dès sa première utilisation avec la commande `use <nom de la base>`
Une base contient un ensemble de collections de documents.

— **Création d'une collection :**
Les documents que l'on stocke, sont placés dans une collection que l'on créer à l'aide de l'instruction : `db.createCollection('<nom de la collection>')` ; Dans ce guide, nous utiliserons la collection 'publis' : `db.createCollection('publis');`
Toute opération sur cette collection devra alors commencer par : `db.publis.xxx`

— **Création d'un document :**
Un document est un *JSon document* qui est encapsulé dans des accolades. Chaque attribut à une clé et une valeur. Une valeur peut être un chaine de texte, un nombre, une liste de valeur ou un document JSon. Il est possible d'intégrer des listes de valeurs à l'aide d'un tableau avec des crochets. Voici un exemple de document : {"type" : "Book","title" : "Modern Database Systems : The Object Model, Interoperability, and Beyond.","year" : 1995,"publisher" : "ACM Press and Addison-Wesley", "authors" : ["Won Kim"], "source" : "DBLP"}
L'insertion du document se fait à l'aide de l'instruction : `db.publis.save(<Document JSon>);`
Il est possible également de sauvegarder un document dans une variable, et ensuite de l'utiliser :

```
w = {"type": "Book","title": "Modern Database Systems: The Object Model, Interoperability, and
Beyond.","year": 1995,"publisher": "ACM Press and Addison-Wesley", "authors": ["Won Kim"], "source":
"DBLP"}
db.publis.save(w);
```

— **Consulter le contenu d'une collection :**
`db.publis.find();`
Seule les 20 premiers éléments sont affichés. Utilisez la commande `lt` pour afficher les résultats suivants.

2.2 API MongoDB

Pour des recherches simples, il faut créer des "documents requêtes". Exemple de recherche de tous les documents ayant un attribut 'type' ayant pour valeur 'Book' : `db.publis.find({"type":"Book"});`

Il est également possible de d'enlever les doublons d'un attribut donné : `db.publis.distinct("title");`

Afin de faire des traitements sur résultats (projection, tri, groupement), la fonction 'aggregate' est nécessaire. Une liste de paramètres doivent être traité, chaque paramètre est précédé d'un '$'. Exemple :

```
db.publis.aggregate([
    {$match : {'type' : 'Book'}},
    {$group : { _id : "$publisher", number : { $sum : 1 } }},
    {$projet : {number : 1}},
    {$sort : {number : -1}}
] );
```

Opérations disponibles :

— $match : recherche simple de l'API MongoDB
— $project : projection des attributs demandés pour le résultat
— $group : attribut à grouper sur valeur, l'opération d'agrégat peut préciser le nom de l'attribut et la fonction ($sum).
— $sort : tri les documents sur les valeurs de l'attribut spécifié
— $unwind : désimbrique une liste de valeurs

2.3 Map/Reduce

Pour faire une requête MapReduce, il faut :

— Définir le map : `var mapFunction = function () {if(this.type == "Book") emit(this.title, 1);};`
— Définir le reduce : `var reduceFunction = function (key, values) {return Array.sum(values);};`
— Définir les paramètres d'interrogation : `var queryParam = {query : {}, out : "result_set"}`
— Lancer la requête : `db.publis.mapReduce(mapFunction, reduceFunction, queryParam);`
— Consulter le résultat : `db.result_set.find();`