Nicolas Travers
Conservatoire National des Arts et Métiers

Introduction

- Hu*mongo*us (monstre / énorme)
- NoSQL : Orienté Documents
  - JSon
  - Documents sérialisés : BSon objects
  - Implémenté en C++
  - Indexation d'attributs (BTree + 2DSphere)
  - Réplication (*Replica Set*) + Distribution (*Sharding*)
  - Stockage : *GridFS*
- Utilisé par :
  - *Doodle, SAP, sourceforge, NY times, bit.ly, github, foursquare, EA games, grooveshark*
  - Licence AGPL (Apache)

Interrogation

- Création d'une base
  - Use *maBD;*
- Création d'une collection
  - db.createCollection('*users*');
- Insérer un document
  - db.users.save ( ); //pas de quotes
- Requêtes orientées documents

```
{
"_id": ObjectId("4efa8d2b7d284dad101e4bc7"),
"name": "James Bond",
"login": "james",
"age": 50,
"address": {
        "street" : "42 Class Street",
        "city": "London"
}
}
```

```
> db.users.find( { login : "james" } );
> db.users.find( { address.city : "London" }, {name : 1});
> db.users.find( { age : { $gt : 40 } } );
        //$gt, $gte, $lt, $lte, $ne, $in, $nin, $or, $and, $exists, $type…
> db.users.find( { name : { $regex : "james", $options : "i" } } );
> db.users.count();
```


Interrogation – API MongoDB

- Valeurs distinctes
  - db.users.distinct("address.city")
- Aggrégats
  - db.users.aggregate([{$sort : {age : 1}}]);
  - db.users.aggregate([{$project : {login : 1, name:1}}]);
  - db.users.aggregate([
    {$match : {address.city : "London"}},
    {$sort : {age : 1}]);
  - db.users.aggregate([
    {$group : {_id : "$address.city", number : {$sum : 1}},
    {$sort : {number : 1}]);
- Séparer les instances d'un tableau
  - db.users.aggregate([{$unwind : "$array"}]);

Exercices

1. Films sorties en 2015
2. Titre des films de J.J. Abrams avec un rang inférieur à 100
3. Année des films d'aventures où Harrison Ford a joué
4. Nombre de films de genre « Adventure »
5. Trier le nombre de films par genre


```
{
    "title" : "Star Wars: Episode VII",
    "directors" : ["J.J. Abrams"],
    "release_date" : "2015-12-16",
    "genres" : ["Action","Adventure","Fantasy","Sci-Fi"],
    "plot" : "A continuation of the saga created by George Lucas.",
    "rank" : 168,
    "year" : 2015,
    "actors" : ["Mark Hamill","Harrison Ford","Carrie Fisher"]
}
```


Interrogation – Map Reduce

```
// Map : fonction appliquée à chaque document. Filtre et produit les
    éléments (clé/valeur) en sortie
var mapFunction = function () {emit(this.address.city, this.login);}
// Reduce : regroupe tous les documents sur la clé. Applique une fonction
    d'agrégat sur chaque ensemble.
var reduceFunction = function (key, values) {return Array.count(values);}
//queryParam : Paramètres d'exécution
var queryParam = {query : {}, out : "result_set"}
    //query : Permet de filtrer les documents AVANT le map, utilise l'API
    MongoDB. Utile pour l'utilisation des indexes
    //out : collection de stockage du résultat
db.publis.mapReduce(mapFunction, reduceFunction, queryParam);
//Consulter le résutlat
db.result_set.find();
```

Exemple de MapReduce

```
var mapFunction = function () {
        if(this.genres.contains("Action"))          emit(this.directors[0], this.rank);
        if(this.actors.contains("Harrison Ford")) emit(this.directors[0], 1);
}
var reduceFunction = function (key, values) {return Array.sum(values);}
```

```
{
    "title" : "Star Wars: Episode VII",
    "directors" : ["J.J. Abrams"],
    "release_date" : "2015-12-16",
    "genres" : ["Action","Adventure","Fantasy","Sci-Fi"],
    "plot" : "A continuation of the saga created by George Lucas.",
    "rank" : 168,
    "year" : 2015,
    "actors" : ["Mark Hamill","Harrison Ford","Carrie Fisher"]
}
```


Mises à jour

- Pas de transactions
- Modifications atomiques de documents
  - $set – Modifie une valeur
  - $unset – Supprime un attribut
  - $inc – Incrément
  - $push – Ajout dans un tableau
  - $pushAll – Plusieurs valeurs dans un tableau
  - $pull – Supprimer une valeur de tableau
  - $pullAll – Supprimer plusieurs valeurs

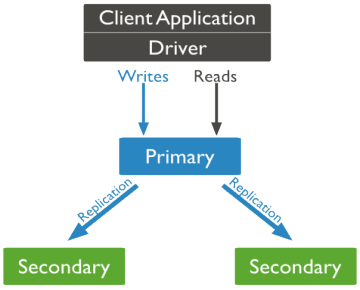
```
> db.users.update( { "_id" :
ObjectId("4efa8d2b7d284dad101e4bc7") } , { "$inc" : { "age" : 1 } }
```

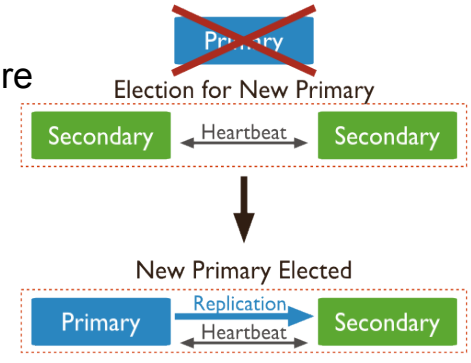
MongoDB et le passage à l'échelle

1. Gestion de la tolérance aux pannes
   - **Replica Set**
   - Ensemble de serveurs gérant la réplication des données
   - ➢ Forte disponibilité
2. Gestion de la distribution
   - **Sharding**
   - Chaque « shard » correspond à un segment des données
   - ➢ Distribution des ressources et calculs

Replica Set

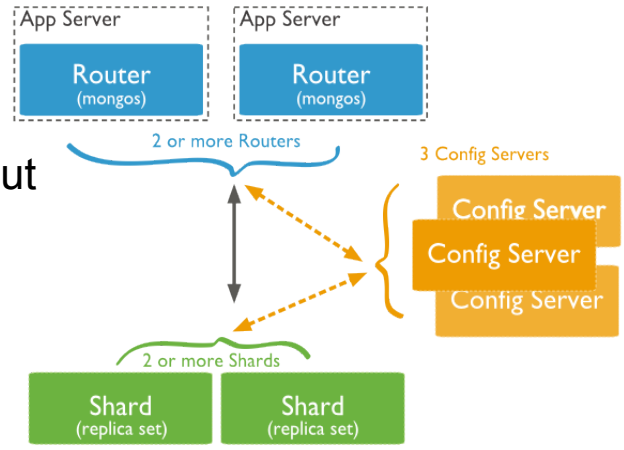
- Réplication d'un serveur
  - Asynchrone
    - Primary server : écritures
    - Secondary servers : lectures
      - Mise à jour via oPlog (fichier de log)
  - Distribue la charge de lecture (requêtes)
  - Tolérance aux pannes
    - Élection d'un nouveau serveur primaire
    - Besoin d'un serveur arbitre



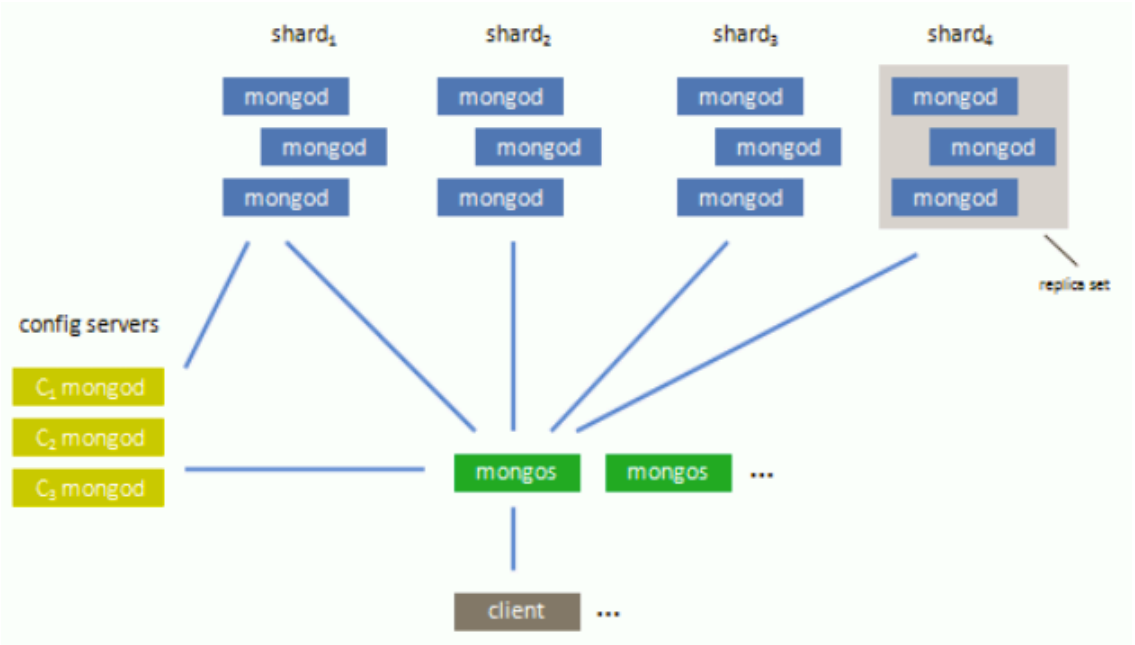


Sharding

- Distribution sur un cluster de machines
  - Equilibrage de charge des données
  - Combinaison avec *Replica Set*
  - Nécessite :
    - *Config Servers*
    - *Mongos* (routeur)
  - Partitionnement sur attribut
    - Ranged-based
    - Hash-based



Sharding & Replica Sets



Outils MongoDB

- Répertoire bin/
  - **mongod** : Lancement du serveur (daemon)
  - **mongo** : shell pour exécuter les commandes
  - **Mongoimport** *:* importation d'un fichier de données
  - **mongostat** : Status du process en cours (inserts, updates, queries, commands, memory…)
  - **mongotop** : Temps réparti en lecture vs écriture
  - **mongos** : Service de routage (sharding)

A savoir

- Utiliser la version 64 bits (32 bits limité à 2Go)
- Pas d'authentification par défaut
- 1 document = 16Mo maximum
- Port par défaut : 27017