BUT Sciences des Donnes 3A - RS 01

Bases de données non relationnelles - 2023/2024

TP – MongoDB

Organisation du TP

Organisation prévisionnelle des séances :

Vous pouvez passer la partie "installation" (pages 2 et 3) si vous utiliser les machines des salles TP.

Séance 1 (mardi)
- Partie 1 : Prise en main de mongodb – page 4
- Partie 2 : recherche dans une collection – la méthode find() - page 6

Séance 2 (vendredi)
- Partie 3 : Le pipeline d'agrégation - page 8
- Rendre 3 exemples de requêtes, voir ci-dessous.

Et suivant le temps :
- Partie 4 : Gestion d'une deuxième collection – page 10
- Partie 5 : Pour aller plus loin : les recherches textuelles – page 10

A rendre (a faire plutôt en 2eme partie de séance vendredi) :

Importer le fichier books.json dans une nouvelle collection 'books'

Proposez 3 énoncés de requêtes basés sur cette collection 'books', faisant intervenir au moins (pas forcément dans la même requête)
- une aggrégation de données ($group)
- une projection

Déposez sur chamillo un fichier au format pdf avec pour chaque requête :
- votre énoncé
- La requête mongodb utilisée pour répondre à la question (avec les methodes find(), ou agregate())
- Le résultat qu'elle produit.

Ressources

- Fichier **MongoDB-recap.pdf :** quelques éléments très résumé de mongodb et des exemples

- Documentation officielle : **https://www.mongodb.com/docs/manual/** Elle est complète, mais principalement en anglais. Il faut parfois choisir 'MongoDB shell' à droite dans 'select your langage', et choisir la bonne version, ici la 6.0.

- Validation de documents json : https://jsonlint.com/

Installation et Démarrage de MongoDB

MongoDB est disponible sur Windows, Mac OS X et Linux
Nous sommes ici dans un contexte d'apprentissage, nous n'allons pas gérer les droits utilisateurs.

MongoDB tourne par défaut sur le port 27017

1.1 Sur les machines des salles de TP

MongoDB est installé sur les machines des salles de TP. Chaque installation contient un serveur plus des outils client d'accès à ce serveur :
- ***mongosh*** : Shell mongo pour l'accès en ligne de commande
- ***MongoDBCompass*** : interface graphique (UI) d'accès à des bases mongo

Le serveur local à la machine s'accède avec la terminologie "localhost"

1.2 Installation sur sa machine

(Passer directement au point 2 si vous utilisez les machines de la salle)

Installer MongoDB Community Edition sur son système.
Suivre le guide d'installation sur: **https://docs.mongodb.com/manual/installation**

windows : Aller sur https://www.mongodb.com/try/download/community
et télécharger le paquet msi

mac : si vous avez brew, installer mongodb avec brew comme recommandé sur le guide d'installation.
Sinon, téléchargez le targz sur le site de téléchargement ci-dessus, décompressez l'archive et déplacez le répertoire dans vos applications "/Applications

Linux
Exemple debian :
```
sudo apt-get install mongodb
```
Au préalable, il faudra potentiellement déclarer le dépôt mongoDB ; pour cela suivre le guide d'installation.

Démarrage du serveur mongoDB

Créer un espace de stockage en lecture écriture pour la base de données, par ex mongodata dans votre home :
```
mkdir ~/mongodata
```
lancer le serveur :
```
mongod --dbpath ~/mongodata
```
Si on veut que le process tourne en permanence en background, on peut utiliser :
```
mongod --dbpath ~/mongodata --fork
```

Test de l'installation :

- Lancer le shell mongo (commande *mongosh*)

- Une fois connecté exécuter le code suivant pour vérifier le bon fonctionnement de la base :

```
db.test.insert({ "test":"helloworld" })
db.test.find({}, {_id:0})
```
Note : le retour chariot déclencha l'exécution de l'instruction, pas besoin de point virgule.

Le résultat attendu est le suivant, la clé générée étant bien entendu différente :

```
{ "test" : "helloworld" }
```

Installation des mongotools (fonction d'import et d'export) :

Voir la doc d'install : https://www.mongodb.com/docs/database-tools/installation/installation/

Crédit

Partie 1 – Prise en main de MongoDB

Premiers pas – Créer des bases et des collections

2 .1 Première prise en main - Insérer et lister des documents

Nous allons voir avec un exemple très simple (la collection 'test') plusieurs méthodes pour exécuter des commandes mongo : les taper dans le client (shell) mongo, ou exécuter un script.

Pour se connecter au serveur mongodb sur le port standard, sur le localhost (machine locale), utiliser la commande :
mongosh
(sur les vieilles versions : *mongo*)

pour sortir du client, taper
exit

1) Dans le **shell mongo** taper directement les commandes :

`use my_db` // **Remplacez my_db par le nom de votre base**. Ex : votre login, tpmongo, ...

`db.test.drop()` // **Efface une collection.** Au cas où on a déjà fait des tests

`db.test.insertOne({"a":1})` // **Insertion d'un document** dans la collection 'test'

`db.test.insertOne({a:2})` // les guillemets sont parfois optionnels dans mongo

`db.test.insertMany([{a:3},{a:4}])`// **Insertion de plusieurs documents**

`db.test.find()` // **affiche le contenu de la collection test**

2) **Le fichier script-javascript.js** contient des instructions équivalentes.

Quand on exécute directement un script js, toutes les fonctionnalités de mongosh ne sont pas disponibles. Il y a quelques différences à apporter.

- Ouvrir le fichier, regarder les différences avec les instructions précédentes, mettre à jour le nom de la base si besoin.

- Exécutez le script depuis la ligne de commande : `mongosh script-javascript.js`

- Exécutez le script depuis le shell mongo : `load("script-javascript.js")`

(Il faut avoir lancé mongosh depuis le répertoire contenant le script – ou donner le chemin absolu du fichier)

3) Pour lancer un script avec exactement les même commandes que mongosh, on peut utiliser le '**<' :**

Mettez à jour `script-mongosh.js` avec le bon nom de votre base de donnée, et testez :
`mongosh < "script-mongosh.js"`

2.2 Utiliser une interface graphique

Il faut savoir se servir de la ligne de commande (shell mongo), mais l'utilisation d'une interface graphique pour manipuler les bases est souvent plus confortable.

La GUI officielle fournie avec mongo est **Compass**. Une autre GUI recommandée est Robo 3T (→ Studio 3T)

Test

- Ouvrir mongoDB compass. Pour la lancer sur les machines de TP, faire une recherche sur mongoDBCompass, s'il n'y a pas d'icone.

- Se connecter à la base par défaut sur le localhost : mongodb://localhost:27017

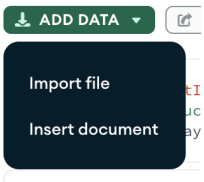
- Retrouver la collection 'test' , tester les 3 modes d'affichage d'une collection (objet, json, tableau)

3) **Import de données depuis un fichier json**

On peut importer des données dans une base mongo, depuis un fichier json ou csv. Nous allons tester ici l'import via l'interface graphique compass puis avec l'utilitaire en ligne de commande mongoimport.

1) Créer un fichier test.json contenant les données suivantes : [{"a":1},{"a":2},{"a":3},{"a":4}]

2) Ouvrir l'interface graphique mongoDBCompass. Depuis la collection 'test', testez la commande 'Insert document' pour insérer les données du fichier précédemment créé.



3) Testez la commande mongoimport depuis la ligne de commande

```
mongoimport --db my-db --collection test --jsonArray --drop test.json
```

Vérifiez via la console ou l'UI que les données ont bien été insérées.

Remarques, pratique :

- Ne pas hésiter à valider avant l'insertion les documents json (https://jsonlint.com/, ou autre)
- Annuler une instruction en ligne de cde : ctrl C
- sur compass, parfois les données ne se rafraichissent pas immédiatement . Penser à utiliser les options de refresh

Partie 2 - Recherche dans une collection avec la méthode find()

Nous allons maintenant utiliser la **méthode `find()`** pour effectuer diverses recherches dans une collection ‘cinema’.

Voir le document mongodb-recap pour un résumé de la méthode find(), ou la documentation mongodb

https://www.mongodb.com/docs/manual/reference/operator/query/

Exercices

1) Informations générales sur une collection

Avec la méthode de votre choix, insérer les données du fichier json *cinema.json* dans une collection nommée ‘**cinema**’.

On cherche tout d’abord à avoir une idée générales du contenu de cette collection :

- La taille (= le nombre de documents)
- La structure des documents

a) Via la ligne de commande *mongosh* :

Visualiser un seul document, en chainant les méthodes *`find()`* et *`limit()`*

Trouver le nombre de documents (plusieurs solutions possibles, par ex count() ou `find()` + `size()`)

b) Avec mongo compass : tester dans l’onglet schema la fonction ‘Analyze Schema"

Y’a t’il des champs qui ont la valeur ‘null’ dans la collection cinema ?

Y’a t’il des champs qui ne sont pas présents dans tous les documents de la collection cinema ?

Les requêtes suivantes sont à exécuter en priorité avec mongosh, mais vous pouvez aussi les tester avec compass.

Pratique : pour les requêtes un peu longue, on peut les écrire dans un fichier .js (coloration syntaxique, correspondance des {} et des []) puis les copier coller dans mongosh ou les executer avec mongosh <my_file.js

2) Affichage – utilisation de la projection avec la méthode find :

Afficher la liste des titres et des années de parution des films, triés par ordre décroissant d’année

Afficher la liste des titres des films, des années de parution des films, et des réalisateurs

Bonus : afficher la liste des titres des film et le premier acteur de la liste des acteurs

3) Recherche. En utilisant la méthode find, répondre aux questions suivantes :

R1 Quels sont les films sortis en 1990

R2 Quels sont les films sortis avant 2000

R3 Quels sont les films sortis soit en 1982 soit en 1990

R4 Quels sont les films pour lesquels on ne connaît pas l'année de sortie

R5 Quels sont les films qui ne sont pas sortis en 1990. Que donnerait une requête SQL équivalente ? Quels sont les films dont on est absolument certain qu'ils ne sont pas sortis en 1990 ?

R6 Quels sont les films réalisés par Martin Scorsese

R7 Quels sont les films réalisés par quelqu'un dont le nom est Eastwood

R8 Quels sont les films réalisé par quelqu'un prénommé Clint et réalisés avant 2000 ?

R9 Quels sont les films dans lesquels joue Clint Eastwood

R10 Donnez le détail des films réalisés par Clint Eastwood, dans lesquels il ne joue pas

R11 Donner le détail des films pour lesquels le premier acteur cité est prénommé Sean

R12 Trouvez les films qui ont plus de 2 acteurs

4. Javascript

Affichage avec javascript

Ecrivez un programme javascript qui affiche les films ainsi:

```
1972  -  Francis Ford   Coppola  -  The Godfather
xxxx  -  Clint   Eastwood  -  Million Dollar Baby
1982  -  Clint   Eastwood  -  Honkytonk Man
xxxx  -  Clint   Eastwood  -  Unforgiven
2008  -  Clint   Eastwood  -  Gran Torino
xxxx  -  Clint   Eastwood  -  Mystic River
1990  -  Martin   Scorsese  -  Goodfellas
```

indices :
- Vous pouvez adapter le fichier script-javascript.js
- Pour instancier un document d'une collection dans un curseur : `document=recordset.next();`
- Option pour tester si une variable `myvar` est de type '`unknown type`' : `if (!(myvar)){…}`

Pour aller plus loin : insertion de dates avec Javascript.
Créer un fichier cinema2.json en copiant le fichier cinema.json, et ajouter un champs "date_sortie" aux objects de ce fichier . Remplir les données de ce champ, avec des dates de type 'yyyy-mm-dd'
Utiliser un script javascript pour insérer les données de ce fichier dans une collection cinema2, en construisant un type date dans la collection pour le champ date_sortie
(Non corrigé)

Partie 3 - Le pipeline d'agrégation

Voir mongodb-recap.pdf, partie 'pipeline d'agregation' pour des exemples.

Pour cette partie, si cela n'a pas été fait dans les parties précédentes, utiliser un fichier annexe pour écrire les requêtes (ex : query.js) avant de les exécuter dans mongosh

On va travailler sur une collection 'films'.

Charger le contenu du fichier films.json dans une collection '*films*' avec la méthode de votre choix.

Note : Utiliser la commande suivante si besoin pour afficher plus de 20 lignes à la fois :
`config.set("displayBatchSize",100) // 100 ou autre valeur`
(Parfois elle n'est pas bien prise en compte – fermer et ré-ouvrir mongosh si c'est le cas)

1) Utilisation de l'étape $project

On utilise la requête ci-dessous avec la méthode find() pour lister les films
français de 2019, en affichant les nom et prénom du réalisateur concaténés.
Mais avec la méthode find() on ne peut pas les trier avec le champ calculé ' director_name'
Ecrivez une requête équivalente avec aggregate(), en ajoutant à la fin un tri par director_name.

```
db.films.find(
        {country:"FR", year:2019},
        { title:1,
          "_id": 0,
          "director_name":{
              $concat:["$director.last_name"," ","$director.first_name"]
          }
        })
```

Bonus : ajouter 2 champ à la racine des documents : l'année de naissance du realisateur (director_birth_date) et son age (director_age). Trier les résultats par age du réalisateur.

2) Calcul d'un nombre d'items - count(*) en SQL

On cherche le nombre de films réalisés par Peter Jackson
Effectuer cette requête avec avec find() puis avec aggregate()

3) Distinct et count distinct

a) On veut savoir combien il y a de réalisateurs différents dans la base film
- Trouver le résultat avec la commande `db.collection.distinct()` qui renvoie un tableau.
Il faut ensuite obtenir la taille du tableau. Trouver la propriété javascript pour cela.
- Retrouver le résultat avec la méthode `aggregate()`

b) On veut un affichage plus lisible de la liste des réalisateurs.
Afficher pour chaque réalisateur ses noms et prénoms concaténés et son année de naissance

4) Utilisation du tri dans le pipeline

a) Quel est le dernier film en date réalisé par Spielberg ?
Il est possible de répondre à cette question avec les méthodes `find()` ou `aggregate()`
Tester les deux. Indice, utiliser les fonctions ou opérateurs `sort` et `limit`
Donner l'id du film, le titre, l'année.

b) On veut maintenant savoir quel est le dernier film réalisé pour chaque réalisateur.
Donner le titre, l'année, et le nom-prénom concaténé du réalisateur.
indices : utiliser le tri au bon endroit dans le pipeline d'agrégation,
utiliser l'opérateur $first dans l'étape $group du pipeline
vérifier quelques résultats avec compass

5) Exemple d'équivalent au 'group by' SQL

a) Compter le nombre de films par genre.
Afficher le genre, le nombre de film pour ce genre, et trier les résultats par ordre décroissant de nombre de film.
Quel genre est le plus représenté ?

b) Comptez le nombre de genres de films différents par pays
Afficher le pays, puis le nombre de genre représentés.
(Indice, il faut faire 2 `$group` de suite)
Quels sont les 3 pays qui ont le plus de genres de films représentés ?
Bonus, utiliser `$push` dans la requête pour ajouter la liste des genres pour chaque pays (array)

c) Réalisateurs qui ont réalisé 5 films ou plus
Affiche le nombre de film, et les nom-prenom concaténés du réalisateur . Trier par ordre croissant de nombre de films

d) Trouver la moyenne du nombre de film réalisés par réalisateur (opérateur de groupe `$avg`)

6) Manipulation des tableaux d'objets

On cherche maintenant des informations sur les acteurs. Pour cela, on va 'déplier' le tableau des acteurs avec l'opérateur `$unwind`

a) Afficher la liste des films dans lesquels a joué l'acteur Viggo Mortensen.
Afficher le titre, l'année et le réalisateur.
Utiliser la méthode `aggregate()` avec `$unwind, $projet` et `$match`

b) Afficher les acteurs (prenom nom) et le nombre de films dans lesquels ils ont joué.
On va restreindre l'affichage aux acteurs qui ont joué dans plus de 4 films.

Partie 4 - Utiliser 2 collections

On repart de la collection ‘cinema’.

On va ajouter une collection ‘*utilisateurs*’

On veut à présent ajouter une nouvelle collection permettant de gérer des utilisateurs et leurs préférences. Pour chaque utilisateur on gérera un pseudonyme, et une liste de films préférés avec une note allant de une à trois étoiles.

On propose l’insertion suivante :

```
db.user.drop()
db.user.insert(
{
 "pseudo":"Luke",
 "liked" :
   [
     {"film":ObjectId("590c366d70f50381c920ca71"),"star":3},
     {"film":ObjectId("590c366d70f50381c920ca72"),"star":1}
   ]
}
)
```

1) Critiquer cette insertion utilisant les identifiants des films. Que se passe-il si vous l’utilisez telle quelle ? Pourquoi n’est-ce pas reproductible ?

2) Proposer un autre (ou des autres) modèles de document pour gérer cette nouvelle collection, et ajouter 2 documents à cette collection.

3) Afficher pour chaque utilisateur son pseudo et les titres des films qu’il a notés, et les notes attribuées.

4) Que faut-il faire pour afficher, pour chaque utilisateur, la liste des réalisateurs dont il a aimé au moins un des films ? Pourquoi n’est-ce pas simple ?

5) Bonus : réaliser un fichier javascript qui réalise cet affichage.
[Non corrigé]

6) Bonus : utiliser l’opérateur $lookup du pipeline d’agrégation pour réaliser cet affichage
[Non corrigé]

Supplément - Recherches textuelles

Voir mongodb-recap, partie ‘recherches textuelles’ pour des exemples.

Sur la collection ‘films’ :

- Cherchez les films réalisés par Mr Gonzalez Inarritu, et donner l’orthographe exacte de son nom.

- Cherchez le film dont le titre est "De battre mon coeur s’est arrêté"

- Trouver les films dont le titre contient "les bronzés"

- En utilisant la fonction `aggregate()`, afficher les films français (titre, prenom-nom du réalisateur), triés par ordre alphabétique sur le prénom (ou le prenom-nom concaténé)