Université de Lille

TP 5.1
MongoDB - CRUD

R5.A.10 - NoSQL
BUT3 - Info
Philippe Mathieu
2024–2025

Objectifs

Savoir manipuler les opérations CRUD de MongoDB

Exercice 1 : Installation du serveur MongoDB

Il existe différentes manières d'installer Mongo. Via votre gestionnaire de paquetage (`apt-get install`) , via le tarball fourni sur le site officiel https://www.mongodb.com ou via un service docker (voir Annexe). Pour des raisons de simplicité nous utiliserons docker.

Q1. Créez une fois pour toutes le service docker avec la dernière version de Mongo

```
docker run --name monmongo -v mes_data_mongo:/data/db -p 27017:27017 -d mongo:latest
```

Q2. Connectez-vous à ce service via le client standard `mongosh`

```
docker exec -it monmongo mongosh
ou
docker exec -it monmongo bash
```

Exercice 2 : Bien démarrer

Q1. Le client `mongosh` est avant tout une console javascript (tout comme `node.js`). Testez d'abord un peu de code Javascript !

```
2+3
a=5
2*a

function f(x){return 2*x+5;}
f(3)

function fact(n) { if (n == 0) { return 1; } else { return n * fact(n-1); } }
for(i=1;i<20;i++){print(i+"\t"+ fact(i));}
```

Q2. MongoDB est initialisé avec 3 bases : `admin`, `config`, `local`
Listez ces bases .....

Q3. Au lancement du client, si rien n'est précisé, on se trouve dans une base nommée `test` (Tant que rien n'est dedans, une base n'apparait pas avec `show dbs`). C'est la base par défaut. Affichez la base courante

Q4. Le changement de base se fait avec la commande `use`. Passez de la base `test` à la base `but3` , vérifiez , et revenez à `test` .....

Q5. Quittez le client `mongosh` .....

Exercice 3 : premiers pas

Q1. Re-connectez vous à la base `test`

Q2. Créez une nouvelle collection nommée `etudiant`.

Q3. Ajoutez 4 documents dedans, une fois avec les 3 clés : `nom`, `prenom`, `ville`, et les autres fois en enlevant tour à tour une de ces 3 clés.

nom	prenom	ville
DURAND	philippe	ici
LEFEBVRE	jacques	
HENRY		ici
	lucie	la bas

Q4. Avec Mongo il est possible d'utiliser des objets javascript pour effectuer l'insertion. Créez un objet javascript avec les 3 propriétés (`DURAND,Pierre,la bas`) et insérez directement cet objet.

Q5. Affichez le nombre de documents de la collection `etudiant` (a priori 5)

Q6. Listez l'ensemble de la collection `etudiant` .....

Vous remarquerez que les 2 premiers enregistrements (qui ont une ville) n'ont pas le même schéma que les suivant (qui ont un login). Un SGBD comme Mongo se moque complètement des schémas et donc on peut mélanger dans la même collection des données complètement hétérogènes..

Q7. L'ordre `insertMany` permet d'insérer toute une liste de documents d'un coup. Créez une liste de plusieurs documents et insérez la liste en une fois .....

```
let  liste =
[
{nom:"ALLART", prenom:"Kevin" , annee:2023 , login:"kevinallartetu"},
{nom:"COLLIN", prenom:"Eliott" , annee:2023 , login:"eliottcolinetu"},
{nom:"DECORTE", prenom:"Anthony" , annee:2023 , login:"anthonydecorteetu"},
{nom:"FLORENT", prenom:"Cyril" , annee:2023 , login:"cyrilflorentetu"},
{nom:"FOSSART", prenom:"Thomas" , annee:2023 , login:"thomasfossartetu"},
{nom:"GRAFTEAUX", prenom:"Edouard" , annee:2023 , login:"edouardgrafteauxetu"},
{nom:"INFELTA", prenom:"Eliott" , annee:2023 , login:"eliottinfeltaetu"},
{nom:"LHOTE", prenom:"Basil" , annee:2023 , login:"basillhoteetu"},
{nom:"LABRE", prenom:"Alexandre" , annee:2023 ,  login:"alexandrelabreetu"},
{nom:"LEDUC", prenom:"Alexandre" , annee:2023 , login:"alexandreleducetu"},
{nom:"MALDEREZ", prenom:"Nathanael" , annee:2023 , login:"nathanaelmalderezetu"},
{nom:"MILLEVERT", prenom:"Matthieu" , annee:2023 , login:"matthieumillevertetu"},
{nom:"PAREE", prenom:"Hugo" , annee:2023 , login:"hugopareeetu"},
{nom:"PARENT", prenom:"Pierre" , annee:2023 , login:"pierreparentetu"},
{nom:"ROUSERE", prenom:"Alexis" , annee:2023 , login:"alexisrousereetu"},
{nom:"TACCOEN", prenom:"Theo" , annee:2023 , login:"theotaccoenetu"},
{nom:"THOMAS", prenom:"Paco" , annee:2023 , login:"pacothomasetu"},
{nom:"VANOORENBERGHE", prenom:"Amaury" , annee:2023 , login:"amauryvanoorengergheetu"},
{nom:"VERRIEST", prenom:"Jordan" , annee:2023 , login:"jordanverriestetu"}
]
```

Q8. Affichez l'ensemble des collections créées .....

Q9. Affichez l'un des documents (le premier) de la collection .....

On constate que chaque enregistrement possède un `_id`. Soit l'utilisateur le fournit, soit Mongo le génère, mais dans tous les cas il est unique.

Q10. Affichez les documents contenant un prénom `Paco` .....

Q11. Affichez uniquement les noms et prénoms des étudiants, sans leur id .....

Q12. Supprimez tous les documents contenant un prenom `Alexandre` (il y en a 2)

Q13. Recherchez l'étudiant `PARENT` et placez le document dans une variable javascript `x`. Détruisez maintenant ce document de la collection en utilisant cette variable `x`.

Q14. Supprimez toute la collection.

Q15. Quittez le client .....

La documentation en ligne fournie par Mongo est bien détaillée. Jetez-y un oeil ! : https://docs.mongodb.com/manua

Vous voilà prêts à être lancés dans le grand bain !

Exercice 4 : Le CRUD : partie R

Q1. Récupérez à partir de moodle le fichier `employes.json`

Q2. Importez ce fichier dans votre base `test` : `mongoimport -d <base> -c <coll> <file>`

Q3. Reconnectez vous à la base `test`

Q4. Affichez toutes les collections de la base

Q5. Listez les documents de la collection `employes`. Combien sont affichés par défaut ?

Q6. Comptez le nombre de documents de la collection `employes`

Q7. Affichez le premier document de cette collection

Q8. Quelle différence y-a t-il entre utiliser `findOne` et `find().limit(1)` ?

Q9. Affichez trois documents de cette collection

Q10. Affichez la liste des employés dont le prénom est `David`

Q11. Affichez uniquement les codes postaux des employés (sans le `_id`)

Q12. Affichez les prénoms sans doublon

Q13. Combien de prénoms différents y a-t-il ?

Q14. combien y-a-t-il de prénoms différents commençant par 'D' (les expressions régulières s'écrivent entre `/` et `/`) ?

Q15. Affichez la liste des employés dont le prénom commence par D ou se termine par e

Q16. Affichez les données triées sur le prénom

Q17. Affichez uniquement les nom et prénom de chaque employé ayant une ancienneté > 10

Q18. Affichez les noms prénoms et ancienneté des employés dont soit le prénom est `David`, soit l'ancienneté est > 20

Q19. Réalisez la requête précédente soit avec une projection inclusive (on spécifie ce que l'on veut) soit une projection exclusive (on spécifie ce que l'on ne veut pas)

Q20. Affichez les noms des employés qui ont touché une prime (l'existence d'une clé se fait avec `$exists`)

Q21. Affichez les nom et adresse complète des employés ayant un attribut `rue` dans l'objet `adresse`

Q22. Affichez les trois premières personnes ayant la plus grande valeur d'ancienneté

Q23. Affichez les personnes dont la ville de résidence est Toulouse (afficher nom,prénom et ancienneté uniquement)

Q24. Affichez les statistiques sur la base courante

Exercice 5 : Le CRUD : parties U et D

Q1. Incrémentez de 200 la prime des employés ayant déjà un champ `prime` (il existe un mot clé `$inc` pour incrémenter une valeur.)

Q2. Supprimez le champs `prime` de tous les documents

Q3. Mettez à jour l'adresse de Dominique Mani : nouvelle adresse (numero : 20, ville : 'Marseille',codepostal : '13015'). Attention, il ne doit plus y avoir d'attribut `rue` dans `adresse` (mot clé `$unset`)

Q4. Ajoutez une clé `ordi` avec un tableau vide comme valeur à tous les employés.

Q5. Ajoutez une valeur `macbook` à cette clé `ordi` pour `Jean Dupond`.

Q6. Supprimez cette clé `ordi` de tous les documents

Q7. Créez une copie `employes2` de la collection `employes`

Q8. Créez une nouvelle collection `result1` contenant les 3 clés (nom,prenom,ville) de tous les documents de la collection employés

Q9. Créez une nouvelle collection `result2` constituée des employés dont le prénom est `David`

Q10. Supprimez de la collection `employe` tous les employés dont le nom commence par 'M'

Exercice 6 : Retour à Postgres (s'il vous reste du temps)

Un SGBD-R comme Postgres sait depuis longtemps stocker et manipuler du Json. Si l'objectif est juste de stocker du json, Postgres répond parfaitement à cette tâche.

Q1. Suivez le petit tutoriel fourni ici

Q2. Testez l'utilisation de la fonction `row_to_json` sur l'une de vos tables Postgres (`livre` par exemple).

Q3. La fonction `json_agg` est une fonction d'agrégation (comme `count` ou `avg`) qui regroupe dans un array json toutes les lignes à agréger (pour info il existe aussi `string_agg` et `array_agg`). Testez cette fonction sur l'une de vos tables (`livre` par exemple) avec regroupement sur la clé étrangère (a priori `ano`)

Q4. La fonction `json_build_object` permet de "fabriquer" n'importe quel objet json. Avec la requête précédente, l'agréga est en json, mais pas la totalité du résultat. Utilisez `json_build_object` pour obtenir cette fois une liste de documents entièrement en JSON.

Q5. Exportez vos auteurs avec leurs livres dans une collection json, et importez l'ensemble dans Mongo.

A la fin de ce TP détruisez complètement les instances `docker` que vous avez créés.
Vérifiez que les commandes `docker ps --all` et `docker volume ls` ne retournent plus rien