P-DB-I165

Une image contenant Police, logo, Graphique, conception

Description générée automatiquement

Tiago Rodrigues Sousa – MID2

A02

32 périodes

Mathieu Meylan

Table des matières

[Spécifications 3](#_Toc160782297)

[Titre 3](#_Toc160782298)

[Description 3](#_Toc160782299)

[Matériel et logiciels à disposition 3](#_Toc160782300)

[Prérequis 3](#_Toc160782301)

[Réalisation 3](#_Toc160782302)

[Restaurer une archive 3](#_Toc160782303)

[Faire un backup de la base de données 4](#_Toc160782304)

[Restaurer le backup de la base de données 5](#_Toc160782305)

[Index 6](#_Toc160782306)

[Conclusion 6](#_Toc160782307)

[Bilan des fonctionnalités demandées 6](#_Toc160782308)

[Bilan personnel 7](#_Toc160782309)

# Spécifications

## Titre

Projet I165 MongoDB

## Description

Dans le cadre du projet I165, il a été demandé de faire plusieurs choses :

* Importation d’un db
* Gestion des rôles et des utilisateurs
* Des requêtes de sélection
* Des requêtes d’agrégation
* Des requêtes textuelles
* Exportation de la db

Pour ce faire nous avons suivi le module I165.

## Matériel et logiciels à disposition

* Un PC ETML
* Environnement docker avec container MongoDB (serveur et mongosh)
* Interface de gestion MongoDB compass + VS Code avec l’extension MongoDB
* Accès à Internet

## Prérequis

Module I165

# Réalisation

## Restaurer une archive

Une archive d’un dump a été fourni et elle s’appelle db\_mflix.gz. Il a été demandé de restaurer la base de données. Voici la commande :



Pour rappel, notre base de données MongoDB est stockée sur un container docker. Ce qui signifie que pour importer la base de données, il a fallu créer un endroit sur l’ordinateur que le container puisse voir.

***docker*** : notre base de données est stocké sur un container docker, ce bout de code indique que nous allons exécuter une commande sur docker

***exec***: cela indique que nous allons exécuter une commande sur un container

***-i mongo***: permet d’indiquer sur quel container va être appliquer la commande (ici, se sera sur le container mongo)

***mongorestore***: indique que nous allons utiliser le programme « *mongorestore »* qui permet de restaurer la base de données qui a été dump (avec la commande mongodump)

***--uri***: indique dans quel serveur nous voulons restaurer la base de données

***mongodb://root:admin@localhost:27017***:

* mongodb = le protocole
* root = nom d’utilisateur
* admin = mot de passe
* localhost (127.0.0.1) = adresse IP
* 27017 = port (attention si vous utiliser docker a bien vérifié quel port est attacher au container)

***authenticationDatabase=admin*** : cette ligne de commande indique dans quel base de données il faut aller chercher les informations de connexion (ici, c’est admin)

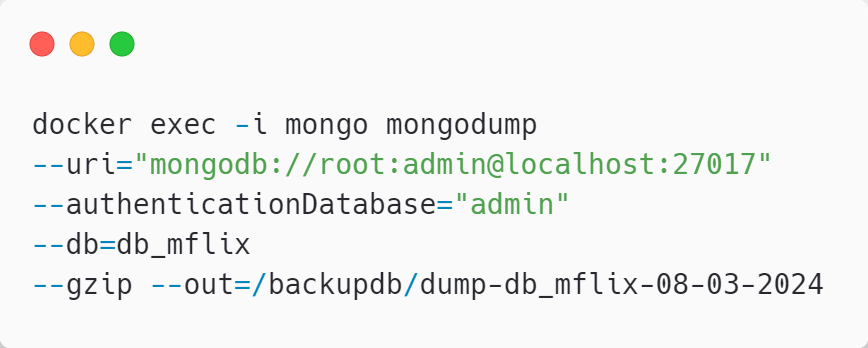
***--gzip***: cette option indique que la base de données que nous lui donnons est compressée (on peut le voir grâce au format .gz)

***--archive***: cette option indique que la base de données que nous lui donnons a été archivé lors du dump, cette à dire que cela a donné un seul fichier (comme lorsqu’on compresse une arborescence de fichier)

***/backupdb/db\_mflix.gz***: ici nous lui indiquons le fichier (attention, il faut que le container puisse voir le fichier donc il faut au préalable partager un endroit entre le container et la machine host)

PS : il faut faire attention au nom des élément dans la commande, si quelque chose comporte des espaces ou des guillemets, à faire attention à entourer le mot de guillemets simple ou double.

## Faire un backup de la base de données



***docker*** : notre base de données est stocké sur un container docker, ce bout de code indique que nous allons exécuter une commande sur docker

***exec***: cela indique que nous allons exécuter une commande sur un container

***-i mongo***: permet d’indiquer sur quel container va être appliquer la commande (ici, se sera sur le container mongo)

***mongodump***: indique que nous allons utiliser le programme « *mongodump »* qui permet de faire une sauvegarde complète de la base de données (en BSON)

***--uri***: indique dans quel serveur nous voulons restaurer la base de données

***mongodb://root:admin@localhost:27017***:

* mongodb = le protocole
* root = nom d’utilisateur
* admin = mot de passe
* localhost (127.0.0.1) = adresse IP
* 27017 = port (attention si vous utiliser docker a bien vérifié quel port est attacher au container)

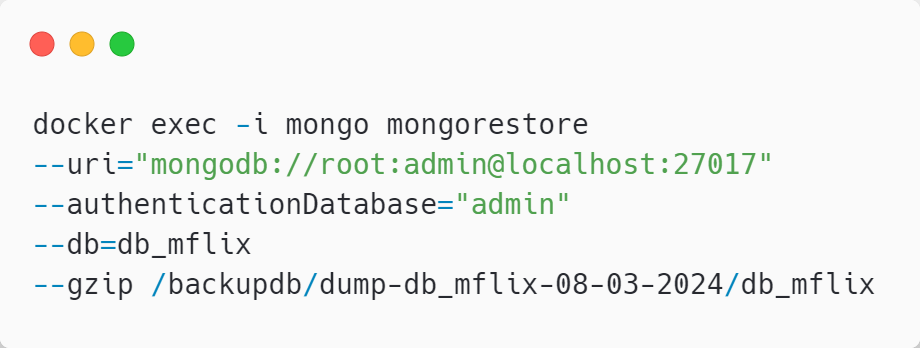
***authenticationDatabase=admin*** : cette ligne de commande indique dans quel base de données il faut aller chercher les informations de connexion (ici, c’est admin)

***--gzip***: cette option indique que nous allons compresser le dump en .gz

***--out/backupdb/mongo-dump-db\_mflix-08-03-2024***: cette option indique l’emplacement d’où va être le backup. Elle sera composée de plusieurs fichiers qui représente chaque une collection. Attention, notre base de données est stockée sur un container donc le fichier va se retrouver la où est le volume partagé

Dans cette commande nous utilisons la méthode de compression gzip pour réduire un maximum la taille du fichier. Pour la restaurer il suffit de suivre les étapes de « Restaurer le backup de la base de données » ci-dessous

## Restaurer le backup de la base de données



Pour rappel, notre base de données MongoDB est stockée sur un container docker. Ce qui signifie que pour importer la base de données, il a fallu créer un endroit sur l’ordinateur que le container puisse voir.

***docker*** : notre base de données est stocké sur un container docker, ce bout de code indique que nous allons exécuter une commande sur docker

***exec***: cela indique que nous allons exécuter une commande sur un container

***-i mongo***: permet d’indiquer sur quel container va être appliquer la commande (ici, se sera sur le container mongo)

***mongorestore***: indique que nous allons utiliser le programme « *mongorestore »* qui permet de restaurer la base de données qui a été dump (avec la commande mongodump)

***--uri***: indique dans quel serveur nous voulons restaurer la base de données

***mongodb://root:admin@localhost:27017***:

* mongodb = le protocole
* root = nom d’utilisateur
* admin = mot de passe
* localhost (127.0.0.1) = adresse IP
* 27017 = port (attention si vous utiliser docker a bien vérifié quel port est attacher au container)

***authenticationDatabase=admin*** : cette ligne de commande indique dans quel base de données il faut aller chercher les informations de connexion (ici, c’est admin)

***--gzip***: cette option indique que la base de données que nous lui donnons est compressée (on peut le voir grâce au format .gz)

***/backupdb/dump-db\_mflix-08-03-2024/db\_mflix*** : ici nous lui indiquons le dossier (attention, il faut que le container puisse voir le fichier donc il faut au préalable partager un endroit entre le container et la machine host) où sont stockée les fichiers compressés

## Index

Un index de type texte composé du champ sur « *title* » avec « *full plot*», un index de type unique sur l’adresse mail de l’utilisateur et un index sur le nom d’utilisateur dans un commentaire peuvent être intéressant.

Généralement, un utilisateur va chercher dans une barre de recherche, des mot clés ou directement le titre dans la barre de recherche. En mettant un index de type texte, on optimise le temps de recherche et le fait d’être composé permet de rechercher des films avec la même syntaxe.

Un mail est unique et il peut être intéressant de mettre un index de type unique car l’email va souvent être utilisé pour se connecter. Cela améliorait le temps de connexion d’un utilisateur. Même s’il y a peu de documents et que donc cela ne fait pas une grande différence, il faut prévoir dans le cas où il y a un grand nombre d’utilisateur.

Un index sur le nom d’utilisateur dans un commentaire peut être intéressant pour améliorer la rapidité à voir un profil lorsque les gens veulent en savoir plus sur la personne qui commente.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Collection** | **Indexe(s)** | **Requêtes** | **Avant (en ms)** | **Après (en ms)** |
| movies | Texte composé  title, full plot | use("db\_mflix");  db.movies.find({ $text: { $search: "\"time travel\"" } }, { \_id: 0, title: 1, fullplot: 1 }); | 34 | 14 |
| users | Unique  email | use("db\_mflix");  db.users.find({email: "mark\_addy@gameofthron.es"}); | 0 | 0 |
| comments | Normal  name | use("db\_mflix");  db.comments.find({name: "John Bishop"}) ; | 10 | 1 |

# Conclusion

## Bilan des fonctionnalités demandées

Il s’agit de reprendre point par point les fonctionnalités décrites dans les spécifications de départ et de définir si elles sont atteintes ou pas, et pourquoi.

Si ce n’est pas le cas, estimer en « % » ou en « temps supplémentaire » le travail qu’il reste à accomplir pour terminer le tout.

## Bilan personnel

Si c’était à refaire:

Qu’est-ce qu’il faudrait garder ? Les plus et les moins ?

Qu’est-ce qu’il faudrait gérer, réaliser ou traiter différemment ?

Qu’est que ce projet m’a appris ?

Suite à donner, améliorations souhaitables, …

Remerciements, signature, etc.