Mongo

Van start gaan met Mongo (lokaal geïnstalleerd)

Mongo lokaal installeren op Windows

Voor we lokaal kunnen werken met Mongo moeten we MongoDB, de MongoDB Shell en de MongoDB tools installeren.

We installeren ook de Community versie en niet de Enterprise versie (biedt extra's zoals geavanceerde beveiliging)

Volg deze stappen: https://www.mongodb.com/docs/manual/tutorial/install-mongodb-on-windows/

ALTERNATIEF, maar onderstaande werkt ENKEL indien Windows Subsystem for Linux is geïnstalleerd.

Ik raad zowiezo aan om de bash shell te gebruiken!

We installeren eerst https://github.com/aheckmann/m - de mongodb version manager

Deze maakt gebruik van npm (check aanwezigheid met npm version)

Ziehier de installatie-instructies (inclusief nagaan indien geïnstalleerd):

```
npm install -g m
m latest
mongod --version
```

mongod — version = testen indien geïnstalleerd

! Verzeker je ervan dat volgende folder is toegevoegd aan je pad: /home/<gebruikersnaam>/.local/bin

Om de shell te installeren (in bash - volg je de stappen voor ubuntu): https://www.mongodb.com/docs/mongodb-shell/install/

Check de instructies!

```
wget -qO- https://www.mongodb.org/static/pgp/server-7.0.asc |
echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/u
sudo apt-get update
sudo apt-get install -y mongodb-mongosh
mongosh --version
```

mongosh —version = testen indien geïnstalleerd

Tot slot de mongodb database tools:

```
sudo apt-get install mongodb-database-tools
mongoimport
```

mongoimport = testen indien geïnstalleerd

Mongo lokaal installeren op Mac

We installeren eerst https://github.com/aheckmann/m - de mongodb version manager

Deze maakt gebruik van npm (check aanwezigheid met npm version)

We gebruiken homebrew voor de shell

Ziehier de installatie-commando's in de zsh-terminal:

```
npm install -g m
m latest
mongod --version
brew tap mongodb/brew
brew install mongosh
mongosh --version
brew install mongodb-database-tools
mongoimport
```

Zorg dat volgende folder is toegevoegd aan je pad: /Users/kristofmichiels/.local/bin

vi ~/.zshrc ⇒ export PATH="/Users/kristofmichiels/.local/bin:$PATH"

Een database opzetten

mongod =

- is een daemon proces voor MongoDB
- behandelt data requests van de mongodb shell of drivers
- voorziet in background management operaties
- luistert standaard naar connecties op poort 27017
- standaard data opslag: /data/db of C:\data\db (werkt niet meer op mac - read only folder)
- Slechts één mongod: niet geschikt in productie-omgeving

mongod uitvoeren en db laten aanmaken in welbepaalde folder:

```
mkdir mongod_folder
mongod --dbpath mongod_folder
```

In afzonderlijk terminal tabblad openen we de mongo-shell:

```
mongosh
show databases
db.test.insertOne({"hallo": "wereld"})
```

Werken met mongodb

Het document model

- Werkt van nature met JSON-documenten.
- Je kunt JSON-gegevens opslaan zoals ze zijn.
- MongoDB gebruikt eigenlijk een binair gecodeerde serialisatie van op JSON lijkende documenten, genaamd BSON, voor het opslaan van gegevens, evenals voor hun drivers en hulpmiddelen.
- BSON is ontworpen om bijzonder licht, doorloopbaar en efficiënt te zijn.
- BSON biedt ook ondersteuning voor extra typen zoals afbeeldingen, tijdstempels, longs (lange gegevensvormen) en nog veel meer.

Namespaces, collections, documents

- Je hebt één database-implementatie en binnen je implementatie kun je één of meer databases hebben. Meestal zullen alle gegevens die bij één applicatie horen, zoals een blog, in één database zijn.
- Binnen elke database zijn er collecties. Een collectie is een groepering van MongoDB-documenten. Voor een blog zou er bijvoorbeeld een gebruikerscollectie en een berichtencollectie kunnen zijn.
- Binnen elke collectie bevinden zich documenten. Documenten vormen de basiseenheid (MAX 16MB/document) van gegevens in MongoDB en elk document bevat één individueel record. Voor een auteurscollectie zou bijvoorbeeld elk document informatie over één auteur bevatten.

Met use maak je een db aan (indien nieuw). Indien ze al bestaat schakel je ernaar over. Elk document dat je aanmaakt heeft een uniek id.

```
use blog
show collections
db.auteurs.insertOne({"naam": "Kristof"})
```

MongoDB Query Language (MQL)

- Tool om data te benaderen in MongoDB
- Wordt ook de MongoDB Query API genoemd
- MQL laat toe om CRUD operaties te doen
- JavaScript gebaseerde shell
- Drivers voor zo goed als alle programmeertalen

```
db.auteurs.insertOne({"naam": "Ben"})
db.auteurs.insertMany([{"naam": "Linda"}, {"naam": "Niels"},
db.auteurs.findOne()
db.auteurs.find()
db.auteurs.find({"naam": "Ben"})
db.auteurs.updateOne({"naam": "Ben"}, {$set: {"website": "htt
db.auteurs.updateMany({}, {$set: {"boeken": []}})
db.auteurs.deleteOne({"naam": "Gregor"})
db.auteurs.deleteMany({})
```

Indexes

- Als je geen index hebt, controleert de database elk document.
- Dit wordt een collectiescan genoemd en is zeer inefficiënt.
- Een index is een georganiseerde manier om gegevens op te zoeken door een subset van de gegevens op te slaan met verwijzingen naar de locatie van het volledige record.
- Als je een index hebt, controleert de database de index.
- Als een query beantwoord kan worden met een index, wordt dit een 'covered query' genoemd.

- Efficiëntere queries en updates.
- Creëer deze! wanneer je vaak query's uitvoert op dezelfde velden of wanneer je vaak bereikgebaseerde query's uitvoert op velden.
- Indexen hebben een prijs: ze moeten onderhouden worden: dit voegt ongeveer 10% schrijfbelasting toe ⇒ snellere leesacties versus tragere schrijfacties.
- Dus: creëer op de meest voorkomende zoekpatronen en wanneer je genoeg RAM hebt om de index te passen.

Soorten indexen

- Single field indexes
- Partial indexes (velden die antwoorden op een query)
- Compound indexes (combinatie van velden)
- Multikey indexes (op maximaal één arraywaarde)
- Text indexes (zoeken binnen tekstvelden)
- Wildcard indexes (velden waarvan je de naam niet weet omdat het schema dynamisch verandert)
- Geospatial indexes
- Hashed indexes (verkleinen de grootte als de oorspronkelijke waarden erg groot zijn)

```
db.auteurs.find()
db.auteurs.createIndex({"naam": 1})
```

findOne() en find()

```
db.authors.insertOne({ <document> })
```

findOne() en find()

```
db.films.findOne({"ratings.mndb": 10})
db.films.findOne({"genres.0": "Musical"})
```

Vergelijkingsoperatoren

De eerste zes zijn de gebruikelijke die je zou verwachten, $eq voor gelijk aan, $gt voor groter dan, $gte voor groter dan of gelijk aan, $lt voor kleiner dan, $lte voor kleiner dan of gelijk aan en $ne voor niet gelijk aan.

```
db.inventory.findOne({"variations.quantity": {$gte: 8} })
db.inventory.findOne({"price": {$lt: 2000} })
```

De eerste is $in en de andere is $nin, maw not in.

```
db.inventory.findOne({"variations.variation": {$in: ["Blue",
```

Logische operatoren

Er zijn 4 logische operatoren die je kan gebruiken in MongoDB, $and, $or, $nor en $not.

```
db.inventory.findOne({"variations.variation": {$and: [{"varia
{"variations.quantity": {$exists: true}]} })
```