een moderne NoSQL database

Oscar Buse
12 November 2024
LUGN

Eigenschappen MongoDB

- NoSQL
- ***documents*** (“(json) records”) worden opgeslagen in ***collections*** (“tables”) in een ***database***
- data type: BSON (Binair JSON)
  - *lightweight*, sneller te “bewandelen”
- Goed horizontaal schaalbaar (mbv *sharding*)
- JSON: geen uniforme data

Niet uniforme data

```
{
 id_: ObjectID("..."),
 Naam: "Bob",
 Woonplaats: "Nijmegen",
 Thuis telefoon: "024 - 12345678",
 Mobiel telefoon: "0612345678"
}
```

```
{
  id_: ObjectID("..."),
  Naam: "Alice",
  Woonplaats: "Arnhem",
  Mobiel telefoon: "0687654321"
}
```

Install

Meerdere servers: gebruik ansible!

Rollen op internet te vinden, bv:

https://medium.com/cloud-native-daily/ansible-playbooks-for-nginx-and-mongodb-configuration-2-f34828d4c937

Pas aan aan jouw situatie.

mongo shell (mongosh)

```
$ mongosh -u user
```

`test> use mydb` # database wordt gemaakt indien deze nog niet bestaat

Switcht naar de database mydb

CRUD operaties:

```
mydb>db.mycollection.insertOne({ x: 1, y: 2 })
```

```
mydb>db.mycollecion.find()
```

```
[ { _id: ObjectId('67327954022eaf7c8cc1c18c'), x: 1, y: 2 } ]
```

`_id` = reserved primary key: kun je wel zelf kiezen (maar moet uniek zijn)

SELECT

```
db.mycol.find()                          # SELECT *
db.mycol.find({"name": "Dirty Harry"}) # "WHERE"
db.mycol.find({'name': /^D/})            # "like"
db.mycol.find({ "age": { $gt: 30}})

db.books.find()
db.books.find({item: 'ABC123'})
db.books.find({item: 'ABC123'},{info:1})
```

Update

```
db.books.update(
   { _id: 1 },           # "WHERE"
   {                     # "UPDATE"
     $inc: { stock: 5 },
     $set: {
       item: "ABC123",
       "info.publisher": "2222",
       tags: [ "software" ],
       "ratings.1": { by: "xyz", rating: 3 }
     }
   }
)
```

Gebruik van python

```
uri = f'mongodb://{username}:{password}@{remote_host}:{port}/{database_name}'
```

Voor replicaset:

```
    replica_set_hosts = [ 'ubu1:27017',    'ubu2:27017',    'ubu3:27017']
    user = “bob”
    pw = “bobtest”
    rep_name = “rs0”
    uri = f'mongodb://{user}:{pw}@{",".join(replica_set_hosts)}/?replicaSet={rep_name}'
client = MongoClient(uri)
db = client[database_name]
collection = db[collection_name]
insert_result = collection.insert_one(document) # document = json-data
```

Users en rollen

Admin user:

```
user: 'oscar',
    db: 'admin',
    roles: [ { role: 'root', db: 'admin' }, { role: 'userAdmin', db: 'admin' },
                { role: 'readWriteAnyDatabase', db: 'admin' },
                { role: 'dbAdmin', db: 'admin' } ]
```

```
rs0 [direct: primary] admin> db.createUser({ user: "bob", pwd: "bobtest", \
                                                roles: [{role: "readWrite", db: "mydb"}] })
```

Replicatie 1/2

- Je maakt een ***replica set***: een groep van mongod processen met dezelfde data set.
- Zeer gebruikelijk in een produktie omgeving met meerdere nodes (servers)

Veel voorkomende setup:

- 1 *primary* node (*write* en *read* operaties)
- *secondary* nodes (alleen *read* operaties)
- arbiter node (optioneel)

Voordelen:

- hogere beschikbaarheid
- snellere *read* operaties (gaan naar meerdere servers)

Nadelen:

- grens aan de groei
- complexer (hoewel tegenwoordig vaak “ingebakken”)

Replicatie 2/2

Primary: operations logged in “oplog”

Secondaries: *replay* “oplog”

Als primary wegvalt: 1 secondary maakt zichzelf primary

Sharding 1/2

Verdeel je data met een “verdeelsleutel” (*shard key*).

Aan de hand van deze key wordt de data verdeeld over de servers (databases).

Kies een juiste ***shard key***. Een shard key geldt altijd voor collections (tables).

- Bv. id_ zou een shard key kunnen zijn (id_ is de primary key van een document).
- Maar een kolom (of een combinatie van kolommen kan ook). Bv. op regio of combinatie van Naam en Woonplaats
- Hashed sharding kan ook: bv. met 5 shard-servers (md5(“row”) mod 5).

Sharding 2/2

Voordelen:

- horizontaal schaalbaar
- beschikbaarheid

Nadelen:

- complexer (hoewel tegenwoordig vaak “ingebakken”)
  - juiste shard key (voor het opdelen (sharden) van de data).
  - extra *routing logic.* MongoDB: bv. een config-server die bijhoudt welke data op welke shard-server staat.
  - geen uniforme distributie (*resharding*?).
    - MongoDB 5.0: je kunt de shard-key wijzigen
    - MongoDB 7.2: reshard met behoud van dezelfde shard-key
  - bulk queries

Bier?