The most basic mongodb

MongoDB

"Mongo DB is scalable, open source, high performance, document oriented database." - 10 gen

Focus on:

- Scalability
- High Availability • Performance

Assessable by different programming languages

Uses JSON as the interchange of data

MongoDB - Setup

- just download and install mongodb from the oficial site

MongoDB - Setup

Execute the mongodb database

<path_to_mongdb>/bin/mongod –dbpath path_for_your_db_files

In another shell, execute the mongodb shell issuing:

<path_to_mongdb>/bin/mongo

1. Your first data

1. **create a new database called db**
2. insert a new document into a new collection called myCollection
3. The document should have attr name with value “go test”
4. Insert a second document with the anme attr “test done”
5. **show all elements** of “myCollection”:

1R. Your first data

1. **create a new database called db**
2. insert a new document into a new collection called myCollection
3. The document should have attr name with value “go test”
4. Insert a second document with the anme attr “test done”
5. **show all elements** of “myCollection”:

- use db
- db.myCollection.insertOne({ name: "go test" }); •
- db.myCollection.find().pretty()
- db.myCollection.insertOne({ name: "go" })
- db.myCollection.find()
- db.myCollection.find({ name: "go" })

Formulário básico

whichdb.whichCollection.find()

db.coll.find({ attr: { $operator: value} })

db.coll.find({ attr: { $operator: value} [, { attr: { $operator: value}] }) •

db.coll.update({attr:value},{$set:{attr:value}})

db.coll.updateMany({ attr: {$op: val} } , { $inc: { attr: val} })

aggreg=db.coll.aggregate([
{ '$group': { '_id': attr, 'total': { ’$op': ”$attr" } } }, { '$sort': { 'total': -1 } }

])

MongoDB- Setup for Exercises

Download from inforestudante the “mongodb_emp.txt”

execute all the lines in the Mongodb shell

Confirm by using : db.emp.find()

mongodb_emp.txt example:

```
db.emp.insertOne( { name: "Joao", sal: 1000.0, age: 18 })
db.emp.insertOne( { name: "Manel", sal: 1100.0, age: 28 })
db.emp.insertOne( { name: "Antonio", sal: 1500.0, age: 30 })
db.emp.insertOne( { name: "Joana", sal: 1500.0, age: 21 })
db.emp.insertOne( { name: "Ines", sal: 1200.0, age: 25  })
db.emp.insertOne( { name: "Ana", sal: 1000.0, age: 18 })
db.emp.insertOne( { name: "Pedro", sal: 1800.0, age: 26 })
db.emp.insertOne( { name: "Toze", sal: 1700.0, age: 27 })
db.emp.insertOne( { name: "Jose", sal: 1200.0, age: 30 } )
```

2. Filtra empregados

- Procura empregados baseado nas condições:

  (op= lt=lower than, gt= greater than)

Devolve todos os empregados com idade menor de 25

Devolve todos os empregados com idade entre 20 e 25

3. Actualiza salario (sal) para valor fixo

- Actualiza salario (sal) e prova que ficou alterado (com find)

Empregados de idade 25 para 1500;

Empregados com idade menor de 25 para 2500;

Empregados com idade entre 20 e 23 para 500

4. Actualiza salários por incremento

- Actualiza em 1000 sal de todos com menos de 25 anos (op=inc incrementa)

5. Agregação

- Dada a seguinte agregação (op=sum) :

```
{ "_id" : 18, "total" : 6000 }
{ "_id" : 30, "total" : 2700 }
{ "_id" : 21, "total" : 2500 }
{ "_id" : 26, "total" : 1800 }
{ "_id" : 27, "total" : 1700 }
{ "_id" : 25, "total" : 1200 }
{ "_id" : 28, "total" : 1100 }
```

- Escreva o correspondente em SQL
- Faça um comando mongodb para devolver este resultado