MongoDB datu-baseen atzipena

Aurkibidea

1. Iturriak:

- MongoDB manuala
- MongoDB Java Driver Documentation
- APIa:
  - Package: com.mongodb.client
  - Package: org.bson
- Maxime Beugnet's Page
- Maxime Beugnet's SlideShow

2. Introduction to MongoDB:

We can read in Wikipedia that: A NoSQL (originally referring to "non-SQL" or "non-relational") database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases. Such databases have existed since the late 1960s, but the name "NoSQL" was only coined in the early 21st century, triggered by the needs of Web 2.0 companies. NoSQL databases are increasingly used in big data and real-time web applications.

There are various ways to classify NoSQL databases. This is a basic classification by data model:

- Wide column: Azure Cosmos DB, Accumulo, Cassandra, HBase.
- Document: Azure Cosmos DB, Apache CouchDB, ArangoDB, BaseX, Clusterpoint, Couchbase, eXist-db, IBM Domino, MarkLogic, MongoDB, OrientDB, Qizx, RethinkDB
- Key–value: Azure Cosmos DB, Aerospike, Apache Ignite, ArangoDB, Berkeley DB, Couchbase, Dynamo, FoundationDB, InfinityDB, MemcacheDB, MUMPS, Oracle NoSQL Database, OrientDB, Redis, Riak, SciDB, SDBM/Flat File dbm, ZooKeeper
- Graph: Azure Cosmos DB, AllegroGraph, ArangoDB, InfiniteGraph, Apache Giraph, MarkLogic, Neo4J, OrientDB, Virtuoso

MongoDB is an open-source *document database* that provides high performance, high availability, and automatic scaling.
A record in MongoDB is a document, which is a data structure composed of field and value pairs. MongoDB documents are similar to JSON objects. The values of fields may include other documents, arrays, and arrays of documents.

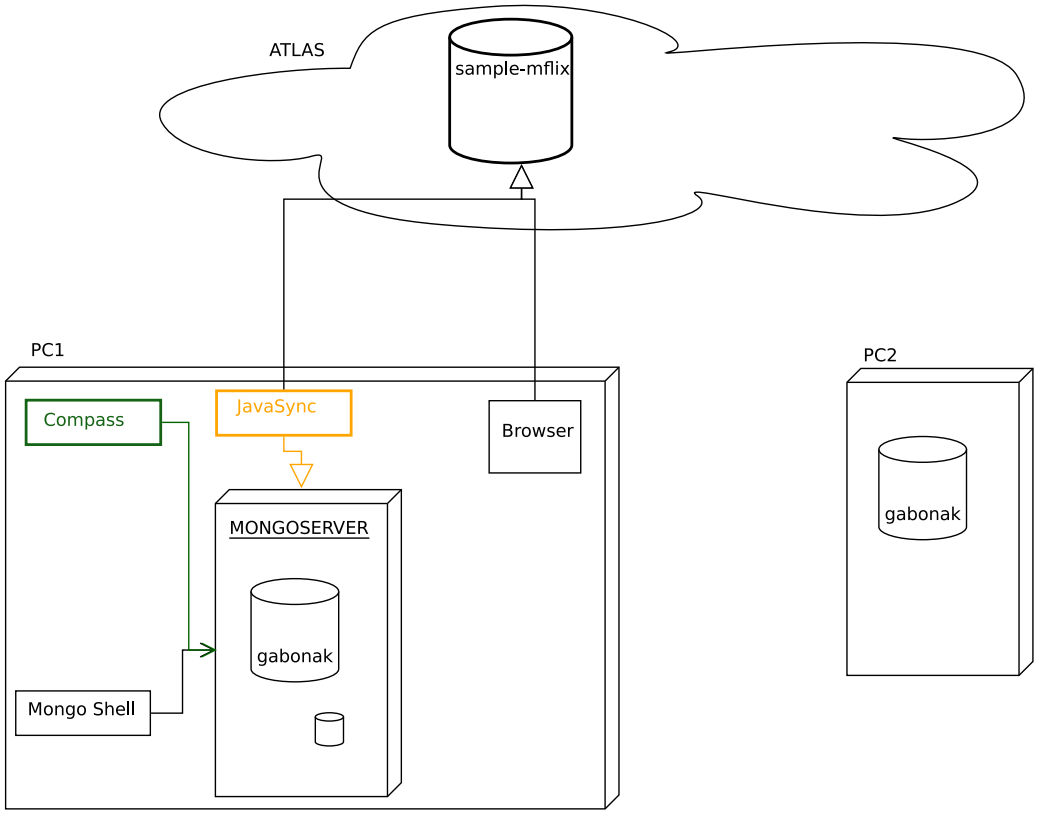
```
{
  name: "sue",                    ←— field: value
  age: 26,                        ←— field: value
  status: "A",                    ←— field: value
  groups: [ "news", "sports" ]    ←— field: value
}
```

3. Mongo Infrastructure

We can learn MongoDB using:

a) The Cloud service Atlas

b) A MongoDB server installed in your machine



4. Queries

First of all we'll learn to make simple queries using Atlas. I created a cluster where you all have read-only permisions.

Owner: imadariaga@uni.eus
Organization: Uni
Project: DAM2
Cluster: 0

In this project you will find some sample databases: sample_supplies, sample_airbnb, sample_restaurants... an sample_mflix

You will find information about MongoDB Tutorial => CRUD Operations=>Queries in order to query the sample_mflix database for the following information:

- Movies released in 1920

- Movies directed by Quentin Tarantino

- Movies from outside USA

- Movies from Spain released after 2014

- List of all movies sorted by year

- Theaters near a specified one (geospatial query)

Aggregation => Aggregation Pipelines

- Zuzendari ekipo bakoitzak zuzendutako pelikula kopurua

- Izenburu errepikatua daukaten pelikulak

- Altuen baloratutako pelikulen kolekzioa sortu

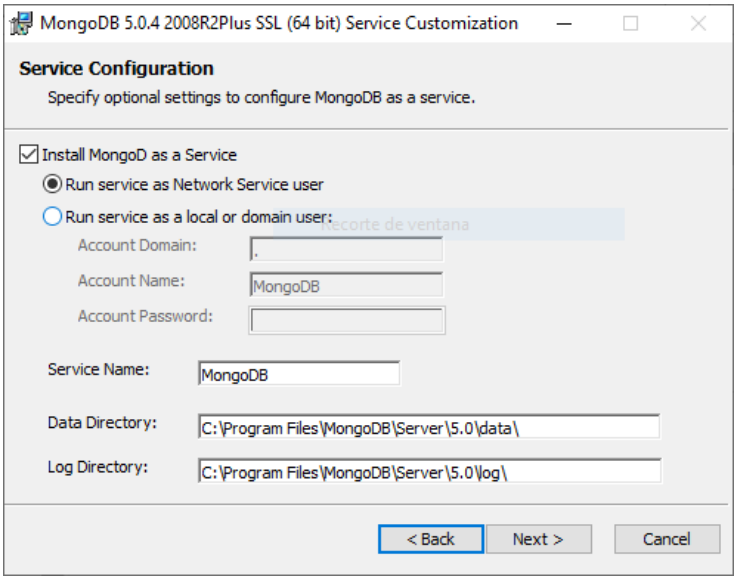
...

(aggregation/aggregation pipelines, geospatial queries, textsearch...)

5. Modification operations: create, update, delete

We will perform modification operations on a local server.

So you must install the MongoDB Server. Here the instructions.



To use the server you can use mongosh shell or Compass, the official GUI client. We'll install Compass (that includes mongosh) using the script InstallCompass in "C:\Program Files\MongoDB\Server\5.0\bin"

(Dokumentu honetan DBEragiketenLaburpena SQL lengoaiaren Mongo sententzia baliokideak aurkituko dituzu.)

5.1 Gabonak

Sortu ezagu "gabonak" izeneko datu-base bat.
Bertan, kolekzio bi izango ditugu "umeak" eta "eskatzaileak".

Umeen kolekzioan dokumentu nahiko sinpleak gordeko ditugu. Ume bakoitzagatik, izena eta eskatu dituen oparien zerrenda. Eskatzaileen kolekzioa antzekoa izango da baina, opari bakoitzagatik, zer opari dan, zein lehentasun daukan eta nori (izena eta adina) eskatzen zaion gordeko dugu.

Begiratu adibideak eta txertatu ezazu datu-basean zuri dagokizun dokumentua eskatzaileen kolekzioan.

```
==================================================


use gabonak

db.dropDatabase()


db.umeak.insertMany([
```

```
{ izena:"Pepa",opariak:["Snow Boots","Apple Pie"]},

{izena:"George",opariak:["A trip to 'Barranco Perdido'","A ticket to watch 'Jurassic Park'","Triceratops"]}
])



db.eskatzaileak.insertOne(

    {izena:"Idoia",opariak:[{zer:"Bidaia(Urtarrilaren 7tik Ekainaren 6ra)", lehentasuna:1, nori:{izena:"Olentzero",adina:800}},

                            {zer:"Erregetako opil handia", lehentasuna:2, nori:{izena:"Meltxor",adina:2080}}]}
);
```

Urruneko zerbitzari bati konektatzea

Compass irekitzerakoan, hutsik utzi ezkero zerbitzariaren eremua, lokalean daukagun mongo zerbitzarira konektatuko gara 27017 portutik.

Beste zerbitzari batera konektatzeko, sintaxi hau jarraitzen duen URLa zehaztu beharko dugu:

```
mongodb://[username:password@]host1[:port1][,...hostN[:portN]][/[defaultauthdb][?options]]

Compass erabiltzen badugu, eremuz eremu bete ditzakegu balioak, eta gero konekzio stringa jaso. Adibidez:

mongodb://prueba:prueba@192.168.65.6:27017/?authSource=gabonak&readPreference=primary&ssl=false
```

Mongo zerbitzariak urruneko konekzioak onartu ditzan aldaketa hauek egin beharko ditugu:

- mongod.cfg fitxategian

```
net:
    port: 27017
    bindIp: 127.0.0.1, X.X.X.X
```

non X.X.X.X, konekzioa eskatuko duen ordenadorearen IPa den.
0.0.0.0 jarriz gero edozein ordenadoretatik konektatu ahal izango ginateke.
Fitxategi honetan autentikazioa habilitatzen ez badugu, ez dugu erabiltzaile eta pasahitza zehaztu beharko.

- Firewallea: Gainera, 27017 portu ireki beharko genuke firewallean, hau da, eskaerei sartzen utzi beharko genieke.
(https://www.howtogeek.com/394735/how-do-i-open-a-port-on-windows-firewall/)

<

6. How to access Mongo from Java (mongo-javatik proiektua)

Mongok nola funtzionatzen duen gutxi gora-behera ulertu ondoren, oinarrizko eragiketa horiek Java erabiliz nola egin ditzakegun ikasiko dugu.

Mongo Java Driver Sync nola erabili azaltzen duen QuickStart tutoriala jarraituko dugu.

Hemen API dokumentazioa eta klaseen diagrama laburtua.

6.1 bat (sample_mflix movies)

Lehenik eta behin, irakaslearen clusterrean daukagun pelikulen datu-basean egingo dugu tutorialak proposatzen duen bilaketa. Datu-base baimena daukan "user1-user1" erabiltzailea erabili dezakezu.

Konekzioa lortu baduzu, eta "Back to the Future" pelikularen datuak kontsolan bistaratzea lortu, primeran.

Dokumentuak bilatzeari buruzko informazioa:
- Find
- Retrieve Data
- Specify a Query

Bertako adibideak jarraituta, saiatu aurreko astean proposatu zenituen 10 kontsultak Javatik exekutatzen.

Ahalik eta proba gehien egin (... eta igo moodlera)

6.2 bi (gabonak => umeak)

Pakete honetan lokalean daukagun *gabonak* datu-baseko *umeak* kolekzioarekin egingo dugu lan POJOak erabilita, Hau da, Umea klase sortu beharko duzu, hemengo adibidea jarraituta: Document Data Format: POJOs

Gero, programatu itzazu CRUD eragiketak Usage Examples orrian aurkituko dituzun adibideak jarraituz.

6.3 hiru (gabonak => erabiltzaileak)

h3>

Pakete honetan, erabiltzaileak kolekzioan dauzkazun dokumentuak mapeatuko dituzten POJO klasea sortu behar duzu. Zehazki, konposizioz erlazionatutako hiru klase sortu behar dituzu.

Egin programatxo bat kolekzio horren inguruko CRUD eragiketak praktikatzeko.

Azkenik, egizu pakete honetan bertan beste programatxo bat zera egingo duena:
Kolekzio berri bat sortu ("opariakemaileko") emaile bakoitzeko dokumentu bat izango duena.
Dokumentu bakoitzean, emailearen izen eta adinaz gain, eskatu dizkioten opari guztien zerrenda agertuko da.