Business Intelligence per i Big Data

Practice #7 – Exploiting NoSQL on MongoDB

Goal

The objective of the practice is to connect to a MongoDB instance, create and successfully populate a collection of documents. Then, visually explore the newly created collection and query the database exploiting different MongoDB-powered NoSQL functionalities and patterns.

Initial set-up

This section provides the instructions to execute the practice in the Lab at Polito or at home.

In the Lab, boot Windows and follow these instructions:

1) Create a local folder (e.g.: C:\Users\Public\Desktop*mongo_database*) and save its path, from now on called: *my_database_path*.
2) Open a Command Shell, navigate to C:\ProgramFiles\Mongo*\bin
3) Run the following command:
   `mongod --dbpath my_database_path`
4) Open another Command Shell, navigate to the same folder as did before.
5) Run the following command:
   `mongo`

You are now logged into the Mongo Shell.

At home, before creating the database collection, we need a running MongoDB instance. To do that, we will user the Docker Platform to create a service running MongoDB and Mongo-Express.

Linux requirements: Docker, Docker Compose
Windows requirements: Docker Toolbox for Windows 10 or older.
(for Windows 10 Pro or Enterprise download and run Docker Desktop)

Place the Docker Compose file (***docker-compose.yml***, provided with practice) in a folder (e.g., $HOME/MongoDB), open a terminal and run the following commands (if the Docker Containers do not start, try renaming the file ***docker-compose.yml*** into ***docker-compose.yaml***):

```
docker-compose up -d
docker-compose ps
```

Copy the name of the docker running MongoDB (from now on called: `name_docker_mongo`).

```
docker exec -it name_docker_mongo bash
```

Login into the Mongo Shell:

```
mongo admin -u root -p example
```

You are now logged into the Mongo Shell.

If you want to use a graphical user interface, you can connect to the MongoDB Express contained in the provided docker image:
Linux: connect to *localhost:8081*
Windows: in the Docker Interactive Shell, run the command: `docker-machine ip default`, then connect to the retrieved IP address at port 8081. Alternatively, access *Kitematic* (installed along with Docker Toolbox), go to the mongodb_mongo_express container and look-up the Docker IP address and port.

Creating the database collection

1) Create a new database for the practice.
2) Populate the newly created database by inserting the documents from the **restaurants_collection.txt** file into a collection called *restaurants*.
3) In order to check the success of the insert, run the command:
   ```
   db.restaurants.find().pretty()
   ```
4) Connect to MongoDB Express to visually explore the dataset (only for the Home set-up).

Running queries of interest

Run the following queries of interest:

1) **Find all restaurants whose cost is *medium***
2) **Find all restaurants whose review is bigger than 4 and cost is *medium* or *low***
3) **Find all restaurants that can contain more than 5 people and:**
   a. **whose tag contains "italian" or "japanese" and cost is *medium* or *high***
      OR
   b. **whose tag does not contain neither "italian" nor "japanese", and whose review is higher than 4.5**
4) **Calculate the average review of all restaurants using the aggregation framework**
5) **Count the number of restaurants whose review is higher than 4.5 and can contain more than 5 people using the aggregation framework**
6) **Run query n. 4 using the Map-Reduce paradigm**
7) **Find the restaurant in the collection which is nearest to the point [45.0644, 7.6598]**
   Hint: remember to create the geospatial index.
8) **Find how many restaurants in the collection are within 500 meters from the point [45.0623, 7.6627]**