

TP-3 Part-2: MongoDB  
Stéphane Vialle & Gianluca Quercini

## Exercise 1: Getting started with MongoDB.

We'll use a database containing information about movies. Data is stored in two files.

- ***moviesEmbedded.json***: This file contains a document for each movie with the following attributes:
  - *\_id*, movie identifier.
  - *title*, movie title.
  - *year*, movie release year.
  - *genre*, movie genre.
  - *summary*, movie summary.
  - *country*, code of the movie country (FR for France, IT for Italy...).
  - *director*, embedded document describing a director with the following attributes:
    - *\_id*, director identifier.
    - *first\_name*, director first name.
    - *last\_name*, director last name.
    - *birth\_date*, director birth date.
  - *actors*, collection of documents. Each document describes an actor with the same attributes as the directors and the following attribute:
    - *role*, the actor role in the movie.
- ***moviesBoxOffice.json***: This file contains the number of tickets sold for each movie, with the following attributes:
  - *\_id*, movie identifier.
  - *box\_office*, number of tickets sold for the movie.

### Question 1.1: Creating a storage folder and launching the MongoDB server.

- Open a first terminal to create a storage folder for your MongoDB database. Name your folder with your login.

```
mkdir /data/mdb/ecm2/ecm2_i      (replace i with your account number)
```

- Load 2 *modules* to access MongoDB:

```
module load mongodb/6.2/gcc-12.3.0
module load mongo-tools/4.2/gcc-12.3.0
```

- Use this terminal to launch the server « mongod » and specify the storage folder:

```
mongod --dbpath /data/mdb/ecm2/ecm2_i --port 10000
```

Note: if port 10000 is busy, use another one (10001, 10002, ...)

- **Don't close this terminal** (minimize it).

### Question 1.2: Loading data into MongoDB.

**Open a second terminal on the same Kyle node** (while the first is still running **mongod**) and load the 2 *modules* required to use MongoDB:

```
module load mongodb/6.2/gcc-12.3.0
module load mongo-tools/4.2/gcc-12.3.0
```

We create in MongoDB a database named *cinema* and two collections called *movies* and *movies\_boffice* respectively. We'll load data into these collections.

- a. Copy the two JSON files containing the data to your home folder:

```
cp ~/vialle/CDL-BigData/movies* ~/
```

- b. Then create the collection *movies* in the database *cinema*. The collection will contain the data from file *movies.json*:

```
mongoimport --port 10000 --db cinema --collection movies
            --jsonArray --file moviesEmbedded.json
```

You should get a message that reads: *imported 88 documents*.

- c. The create a collection *movies\_boffice* that contains the documents from file *moviesBoxOffice.json*:

```
mongoimport --port 10000 ...
```

**Question 1.3: Launching the MongoDB *shell* to query the database.**

- a. Use the following command to launch the MongoDB *shell* that you'll use to interact with the server MongoDB:

```
mongosh --port 10000      (or another port if you did not run mongod on port 10000)
```

- b. Use the following command to tell the MongoDB server that you want to use the database *cinema*.

```
use cinema
```

- c. Verify that the documents are in the collection *movies* with the following query:

```
db.movies.findOne()
```

- d. Verify that the documents are in the collection *movies\_boffice* with the following query:

```
db.movies_boffice.findOne()
```

## Exercise 2: Queries, aggregations, joins, and mapReduce.

### Question 2.1: MongoDB queries

Using the MongoDB shell, write the following queries in MongoDB to get:

- a. The release year of the movie “Le parrain III”.
- b. The title of the movies released between 1980 and 1990.
- c. Same query in b. with the titles sorted in alphabetical order.
- d. The title of the French movies.
- e. The title of the movies with genre “crime” or “drama”.
- f. The names and birth dates of the directors of French movies.
- g. The title of the movies of which Sofia Coppola is one of the actors.
- h. The title and the genres of the movies of which the director is Hitchcock.

### Question 2.2: Aggregation in MongoDB

Using the aggregation framework (*with `aggregate([...])`*), write the following queries in MongoDB to get:

- a. The number of movies per country. Show the result in descending order.
- b. The name of the actor with the role “Mary Corleone” in the movie “Le parrain III”.
- c. The number of actors by film, sorted in descending order.
- d. The average number of actors per film.

### Question 2.3: Join in MongoDB

Find the number of tickets sold for “Le parrain III” by using the operator `$lookup` (that implements a “join”) on the collections `movies` and `movies_boffice`.

The result should be: 5061739.

### Question 2.4: Join and mapReduce

- a. Generate the collection “movies2” with the result of `$lookup` for all films in the database.
- b. Write a mapReduce in MongoDB to compute the number of tickets sold to which each actor contributed.
  - Describe your algorithm.
  - Define a function `map1` and a function `reduce1` in JavaScript in the MongoDB *shell*.
  - Implement a mapReduce calling the functions `map1` and `reduce1`, and saving the result to a collection “actorsbo”.
  - Print the number of tickets sold for each actor, in decreasing order.

The result should be :

Bruce Willis : 34090927

Morgan Freeman : 26362352

....

Rutger Hauer : 2123163

Christian Slater : 2118341