Laurea Magistrale in
DATASCIENCE

How TuBe Popular - HANDBOOK

Handbook del progetto per il corso di *Data Management and Visualization*

Studenti:
Leonardo Alchieri - 860624
Davide Badalotti - 861354
Lucia Ravazzi - 852646
Pietro Bonardi - 859505

GitHub Repository:
https://github.com/LeonardoAlchieri/Top-Of-Youtube

February 11, 2021

Introduction

Small guide for the project in Data Management & Visualization. If one follows the instructions in this handbook, he or she should execute all of the steps as we did.

If not dove already, check out our GitHub repo at:
https://github.com/LeonardoAlchieri/How-Tube-Popular.

1 Pre-requisites

- `MongoDB`
- `Python`, at least version 3.5 or above.
- `Python` libraries. All of the necessary libraries can be install via

```
pip install -r requirements.txt
```

- To connect Mongo to Tableau, just use the following online guide https://docs.mongodb.com/bi-connector/master/installation/.
- Follow this guide to install the Tableau driver for Mongo: https://help.tableau.com/current/pro/desktop/en-us/examples_mongodb.htm.

2 Connect to MongoDB

Steps to follow to start VMs, start the MongoDB sharded cluster and connect to it.

1. Start the Azure Virtual Machines, `tars1`, `CASE` and `HAL`.

   This can be done through the Azure portal platform.[1]

2. Log in each VM, through `ssh` instance, and start the Mongo instance. More specifically:
   - For `tars1`:

     ```
     ssh davidebadalotti@tars1.bounceme.net
     ```

     ```
     ./mongodb_start.sh
     ```

     This will start one of the **mongod config replica** *leopardi* and the **shard** *pascoli* in it hosted.
   - For `HAL`:

     ```
     ssh cooper@168.61.100.92
     ```

     ```
     ./mongodb_start.sh
     ```

     This will start one of the **mongod config replica** *leopardi* and the **shard** *manzoni* in it hosted.
   - For `CASE`:

[1] https://portal.azure.com. For access to our VMs, please contact us.

```
ssh cooper@CASE.bounceme.net

./mongodb_start.sh
```

This will start one of the **mongod config replica** *leopardi*, the **shard** *manzoni* in it hosted and the only **router**, *dante*.

3. Connect to the router with:

```
mongo --host=case.bounceme.net --port=80
```

3 Load data

Steps to follow to load data, as we did, on the Sharded DBMS.

1. Load Kaggle data from SQL database onto Mongo.

```
mpirun -n 3 python3 dataLoading/kaggleMongo.py
```

To change hostname and database for mongo, modify `dataLoading/configMongo.yml`

2. Get data from the API and load on Mongo.

```
python3 dataLoading/APIMongo.py
```

To change hostname and database for mongo, modify `dataLoading/configMongo.yml`.

To change API search settings, modify `dataLoading/configAPI.yml`

3. Scrape data from both API and Kaggle `ids`.

```
mpirun -n <num_cores> python3 dataLoading/scrapeMongo.py <num_source>
```

where `<num_cores>` represents the number of threads the code shall be run on and `<num_source>` is the indetifier for the source of the scraping, i.e. 1 for Kaggle and 2 for API.

4 Enrich & Integrate collections

1. Enrich Kaggle collection with scraping collection.

```
mpirun -n <num_cores> python3 dataIntegration/enrichKaggleWithScraper.py
```

where <num_cores> represents the number of threads the code shall be run on.

2. Enrich scraping collection with API collection.

```
mpirun -n <num_cores> python3 dataIntegration/enrichScrapingWithAPI.py
```

where <num_cores> represents the number of threads the code shall be run on.

3. Enrich scraping collection with Kaggle collection.

```
mpirun -n <num_cores> python3 dataIntegration/enrichScrapingWithKaggle.py
```

where <num_cores> represents the number of threads the code shall be run on.

4. Integrate scraping and Kaggle collections into a single one.

```
mpirun -n <num_cores> python3 dataIntegration/integrateAllTogether.py
```

where <num_cores> represents the number of threads the code shall be run on.

5. Enrich final collection with thumbnails.

```
mpirun -n <num_cores> python3 dataIntegration/enrichThumbnail.py
```

where <num_cores> represents the number of threads the code shall be run on.

6. Enrich final collection with other calculated fields.

```
mpirun -n <num_cores> python3 dataIntegration/enrichFinal.py
```

5 Connect Tableau to Mongo

- Connect `mongosql` to the router.

```
mongosqld --mongo-uri="mongodb://<ip\_address>:<port>/?connect=direct" --sampleNamespaces=<database>.<collection>
```

where <ip_address> specifies the ip address of the mongos, <port> its port, <database> the database to which connect and <collection> the collection to use.

- Open Tableau. Under the `To a server` instance, click `more...` and find `MongoDB BI Connector`. When prompted by a login window, select as `server` the ip address of the `mongosqld` process and its port. One must specify as well the database loaded using `mongosqld`.

  If no authentification is needed (as our case), leave the fields empty.