BDMM - Final Project

June 4, 2020

1 Big Data Modeling and Management Assigment

1.1 EU Procurements Explorer Dashboard Competition

For the final project we will continue to work with the european public procurement notices database to analyse contracts and money expenditure within the European Union!

This time the goal is to feed data to a dashboard where we can explore the contracts in three diferent ways: per procurement code, per country and per company.

Problem description Explore the code in the zip file shared with this project description. Spin up the dashboard and check there are no errors. Go to the `queries.py` and replace the exercices with actual queries following the example in `ex0_cpv_example`. Check the dashboard charts are now working. Run the `Performance test`in the front page of the dashborad and try to optize the speed of your queries with the materials teached in classe like indexes and data modelling. Confirm you have fast queries and the dashboard is working. Submit the `queries.py` on moodle

Connection details to the MongoDB database Each group should have received by email the credentials to connect to the group's mongo database. The same ones as homework 2.

`Connection example: mongodb://username:password@host:port`

These credentials should be added in the file `DB.py` within the project folder in the backend folder.

Project structure

- apps - All the dash dashboard code
- assets - Web assets for the dashboard
- backend
  - `DB.py` - File with the database connection, this should be changed to your groups database connection
  - `queries.py` - File where the queries will go (the one to be submited)
  - `performance_evaluation.py` - Code used to run the performance evaluation
- `index.py` && `app.py` - Dash basic files. To start the app run `python3 index.py` (or `docker-compose up` if familiar with docker technologies)
- `test_insert_document.json`- example document to insert on the dashboard and measure the time taken

1.1.1 Questions

Procurement codes (CPV)

1. 5 descriptive metrics of the contracts related to the CPV, the average of:
    1. Each CPV's division contracts average spending ('VALUE_EURO')
    2. Each CPV's division contract count
    3. Each CPV's division contracts average number of offers ('NUMBER_OFFERS')
    4. Each CPV's division contracts average spending ('VALUE_EURO') with european funde ('B_EU_FUNDS')
    5. Each CPV's division contracts average spending ('VALUE_EURO') without european funds ('B_EU_FUNDS')
2. The count of contracts for each CPV Division
3. Per CPV Division get the average spending ('VALUE_EURO') and return the highest 5 cpvs
4. Per CPV Division get the average spending ('VALUE_EURO') and return the lowest 5 cpvs
5. Per CPV Division get the average spending ('VALUE_EURO') and return the highest 5 cpvs for contracts which recieved european funds ('B_EU_FUNDS')
6. Per CPV Division and get the average ('VALUE_EURO') return the highest 5 cpvs for contracts which did not recieve european funds ('B_EU_FUNDS')

7. The highest CPV Division on average spending ('VALUE_EURO') per country ('ISO_COUNTRY_CODE')
8. Returns bucketed data with the contract counts of a particular cpv in a given range of values (bucket) according to spending ('VALUE_EURO')
9. The average time and value difference for each CPV, return the highest 5 cpvs

Countries

10. 5 descriptive metrics of the contracts related to the Country, the average of:
    1. Each Country's contracts average spending ('VALUE_EURO')
    2. Each Country's contract count
    3. Each Country's contracts average NUMBER_OFFERS'
    4. Each Country's contracts average VALUE_EURO' with 'B_EU_FUNDS'
    5. Each Country's contracts average 'VALUE_EURO' without 'B_EU_FUNDS'
11. The count of contracts per country ('ISO_COUNTRY_CODE')
12. Returns the average 'VALUE_EURO' for each country, return the highest 5 countries
13. Returns the average 'VALUE_EURO' for each country, return the lowest 5 countries
14. For each country get the sum of the respective contracts 'VALUE_EURO' which recieved european funds ('B_EU_FUNDS')

Companies

15. 5 descriptive metrics of the contracts related to the Company, the average of:
    1. Each Company's contracts average spending ('VALUE_EURO')
    2. Each Company's contract count
    3. Each Company's contracts average NUMBER_OFFERS'
    4. Each Company's contracts average VALUE_EURO' with 'B_EU_FUNDS'
    5. Each Company's contracts average 'VALUE_EURO' without 'B_EU_FUNDS'

16. Returns the average 'VALUE_EURO' for company ('CAE_NAME') return the highest 5 companies
17. Returns the average 'VALUE_EURO' for company ('CAE_NAME') return the lowest 5 companies
18. Returns the count of contracts for each company 'CAE_NAME', for the 15 companies with the most contracts
19. For each country get the highest company ('CAE_NAME') in terms of 'VALUE_EURO' sum contract spending
20. Returns the top 5 most frequent co-occurring companies ('CAE_NAME' and 'WIN_NAME')

All resulting documents should allow to perfom filters by min and max of the field year of contract, as well as for issuer country.(ISO_COUNTRY_CODE) (see example 0 for more details)

Insert query
21. On the `queries.py` there is a working function that inserts documents on the contracts database.
If any precomputed table is generated they should be recomputed with the new data on the this insert method.

1.1.2 Group

This project assumes groups to be the same as the previous project, any copying detected by the professors will lead to a grading of zero on the project and/or other disciplinary actions!

1.1.3 Submission

Submit the `queries.py` file with all the queries (running on the group's own database) on moodle.

Delivery date: Until **23:59 of June 19th** (as there will be no exam the due date got extended)

1.1.4 Evaluation

This will be 30% of the final grade.
1. The queries run and generate the desired visualizations. (60%) 1. The speed of the query. (This will be benchmarked for all groups) (20%) 1. The document insertion speed. (This will be benchmarked for all groups) (10%) 1. The simplicity of the query. (10%)

The queries will be run against each groups database. So any index or extra table created will be used.

All code will go through plagiarism automated checks. Groups with the same code will undergo investigation.

1.1.5 Extra information

Rounding of numbers can be perfomend with any function, they will not be an evalution criteria.
Hint: To speed up the queries two sugestions are indexes and precomputed tables.