Databses 2, Projects 2023

Igor Wojnicki

March 10, 2023

Contents

1 General terms

There will be 2-person groups.

1. Working environment. Each topic will get a separate GitLab project, which should be used to manage the project (board/issues), document it and store the code and other data. All relevant files should be in the master branch.

2. Choice of technology. Unless stated otherwise, the chosen technologies and architecture should be as light as possible, e.g. require minimum setup, be easy to launch and evaluate. Projects must run under Linux. Architectural choices have to be approved by the instructor.

3. Self-sufficiency. Projects should be self-sufficient, e.g. include code needed to set up the dependencies, create database structures and populate them with data. Source data should be collected from external sources by a script or included in the repo.

4. Documentation. Projects should include documentation in `README.md` or `README.org` file in the root directory of branch main in the form of Markdown or org-mode files respectively. It should contain instructions how to run the software, define the prerequisites and document the project design and implementation process. It has to state the roles of all the students in the project and description of who did what.

5. Results. The results should be easy to evaluate, and reproducible. A a step-by-step manual how to reproduce the results have to be provided.

6. Self-evaluation. Self-evaluation of the projects is mandatory. Efficiency should be discussed, along with proposed strategies for future mitigation of identified shortcomings.

2 Grading policy

The projects will be graded based on the items 2-6 of the General Terms and technical merit.

When your project is ready to be graded send me an email indicating it. You must do it at least one week before the semester ends to pass the project class on time.

You can make up the project twice. A make up request must be sent by email at least one week before the make up exam session ends. All make ups are graded and the grades remain on your record.

3 Topic

There is a comma separated value (CSV) file `taxonomy_iw.csv.gz` given which forms a directed graph. It represents Wikipedia main topic classification categories. Each line is a single record with two fields indicating category-subcategory relationship, e.g. :

```
"1880s_films","1889_films"
```

indicates that there is a `"1880s_films"` category which has `"1889_films"` subcategory.

4 Goals

The main goal is to make a command line utility that:

1. finds all children of a given node,
2. counts all children of a given node,
3. finds all grand children of a given node,
4. finds all parents of a given node,
5. counts all parents of a given node,
6. finds all grand parents of a given node,
7. counts how many uniquely named nodes there are,
8. finds a root node, one which is not a subcategory of any other node,
9. finds nodes with the most children, there could be more the one,
10. finds nodes with the least children, there could be more the one,
11. renames a given node.

5 Steps

1. Knowing the data format and that it represents a graph, choose a database that can accommodate it. Consult and confirm your choice.
2. Design a database representation of the provided CSV data. Consult the schema or representation.
3. Import the CSV into the database of your choosing. Consult the tooling and how the import is performed.
4. Implement the utility to meet the goals.
5. Make the utility as fast as possible, use indexing if appropriate. Provide timings how fast it takes to run the command for each of the goals.