Technical test

The goal of this test is to evaluate your ability to parse and process a large amount of data and to use the appropriate data structures to solve a challenge.

The goal is to extract the popular (most frequent) queries done during a specific time range from a given log dataset.

The provided sample file is a TSV file. Each line contains a timestamp and a query separated by a tab.

Instructions

Part 1: Code

Build a small application exposing the following endpoint through a REST API:

- `GET /1/queries/count/<DATE_PREFIX>` : returns a JSON object specifying the number of distinct queries that have been done during a specific time range

Examples

- Distinct queries done in 2015: `GET /1/queries/count/2015` : `{"count": 573697}`
- Distinct queries done in Aug: `GET /1/queries/count/2015-08` : `{"count": 573697}`
- Distinct queries done on Aug 3rd: `GET /1/queries/count/2015-08-03` : `{"count": 198117}`
- Distinct queries done on Aug 1st between 00:04:00 and 00:04:59: `GET /1/queries/count/2015-08-01 00:04` : `{"count": 617}`

Guidelines

- You can use third-party libraries if you know how to re-implement the features they're providing
- Your application cannot depend on any database or external software (one of the goals being to evaluate your ability to choose the right data structures)
- Provide a solution in one of the following languages: Go, Python
- Don't overthink the assignment. Provide a solution that you would be happy to push to production
- If you have several implementations in mind, pick one and discuss the alternatives in the README of your project

Evaluation Criteria

- Please push your code to a GitHub repository or send us an archive
- Include a Readme helping us run your service
- Include a section about your thought process explaining your choices and share other alternative designs you considered
- Add any information you deem interesting for us to better understand your assignment

We'll evaluate:

- The complexity & scalability of your algorithm
- The readability of your code (including the readability of your tests)
- The correctness of the outputs of the API
- You're ability to share your design choices and clearly weight pros & cons of alternative solutions
- The overall quality of your written technical communication

Whenever in doubt, feel free to implement subtle things on your own.

Good luck!