Teza Technologies Structured Data Test

**Objective**

We are interested in understanding how you might approach a potentially complex data problem**.** We **strongly encourage you to spend at most an hour on this test, and would be perfectly happy to receive pseudo-code solutions**, since our main focus from your response would be on attention to detail and critical thinking. Please include any code/terminal history used to help analyze or process data along the way.

**Background**

Teza just started trial data access for a (hypothetical) vendor named Social Media Signal (SMS), who monitors various social media outlets such as Twitter and uses natural language processing to determine whether the social media messages is positive or not positive for various stock symbols. The company aggregates their feeds and provides a daily file at ~8pm EDT with the file name format below, where YYYYMMDD is the date of file generation:

social\_media\_signal.YYYYMMDD.csv

The files may include updates to historical values up to 15 days ago, and have the following columns:

|  |  |  |
| --- | --- | --- |
| Column | Data Type | Description |
| date | date | The effective date of the social media signal |
| ticker | text | The stock ticker of the company as of the corresponding date |
| positive | float | The fraction of total number of social media signals that were determined to be positive, this column should be between 0 and 1 |
| tweets | int | The total number of social media signals |

**Task**

1. Analyze 2017 trial data from Social Media Signal and identify any issues. Please keep a history of commands/code/tools used to review vendor data.

2. Design (and implement if time allows) an API and its necessary data workflow that will allow researchers to specify a teza identifier (teza\_id), a date range, and a file generation date. The API should return the most up-to-date revisions of SMS data, for the specified teza\_id valid within the specified date range, from files that are generated up to and including the specified file generation date.

Here’s an example python interface, but feel free to write in other languages:

1. **class** SocialMediaClient(object):
2. \_\_metaclass\_\_ = ABCMeta
4. @abstractmethod
5. **def** get\_data\_by\_range(self, teza\_id, start\_date, end\_date,
6. file\_generation\_date=None):
7. """
8. get the Social Media Signal vendor data for the specified teza\_id
9. between start\_date and end\_date inclusive, from files that are
10. generated up to but not after the specified file\_generation\_date.
12. :param teza id: unique teza instrument identifier
13. :type teza\_id: int
14. :param start\_date: inclusive start date of vendor data
15. :type start\_date: date
16. :param end\_date: inclusive end date of vendor data
17. :type end\_date: date
18. :param file\_generation\_date: the maximum file generation date to
19. get vendor data from, defaults to latest
20. :type file\_generation\_date: date
21. :return: list of dictionaries in the format of
22. {'date': <date>,
23. 'ticker': <ticker>,
24. 'positive': <positive>,
25. 'tweets': <tweets>}
26. :rtype: list
27. """

(Please assume the JSON content stored in the file teza\_ticker\_mapping\_ts.json is the result of an existing API that maps Teza’s teza\_ids to SMS’s tickers, and specifies the inclusive valid date range of each mapping where the dates are in YYYYMMDD format)

3. How does your design change for question 2 if the files were very large or updated much more frequently? How does it change if arbitrary historical data could be revised?