|  |
| --- |
|  |
| Problem 1: Data wrangling Edgar data from text files |
|  |
| Praising Data |

**Northeastern University**

October 6, 2017

Authored by: Madhumathi Prakash and Sonali Chaudhari

Problem 1: Data wrangling Edgar data from text files

Praising Data

# The Data Set

In this problem we are using the EDGAR (Electronic Data Gathering, Analysis, and Retrieval) system. It contains company’s performs automated collection, validation, indexing, acceptance, and forwarding of submissions.

# The Problem

We are accessing data from Edgar site. Given a CIK number and the accession number, we are creating a url to get the data but replacing the X, Y, and Z in the following url: [http://www.sec.gov/Archives/edgar/data/XXX/ZZZ/YYY- index.html](http://www.sec.gov/Archives/edgar/data/XXX/ZZZ/YYY-%20index.html) . From this url we are locating the 10Q file html link and then extracting all the tables. Lastly, we are loading all the tables into a CSV file. When looking for the tables in the 10Q files, we should also deal with logging all activities. For instance, we should deal with error if there is an invalid CIK or accession number or if the amazon keys aren’t valid.

We are using Docker for this problem and building a docker image which will automate the task. Put all zip file on Amazon S3.

# Our Approach

1. We first took a specific use case of the IBM url and worked with it: ([http://www.sec.gov/Archives/edgar/data/51143/000005114313000007/0000051143-13-000007- index.html](http://www.sec.gov/Archives/edgar/data/51143/000005114313000007/0000051143-13-000007-%20index.html)) and parsed the url and got the specific 10-Q file.
2. To make the url general: [http://www.sec.gov/Archives/edgar/data/CIK/ZZZ/access number- index.html](http://www.sec.gov/Archives/edgar/data/CIK/ZZZ/access%20number-%20index.html)
   1. We added CIK number and accession number to a text file.
   2. Then we read each line in the text file and assigned it to the CIK and access number accordingly.
   3. To deal with “ZZZ” we realized that it was the access number without the “-“ so we removed it.
   4. We then concatenated URL with these items.
3. In terms of error logging, we are dealing with the use case of if user does not provide CIK or access number, then we will provide an logging.info message in the error log saying that, because CIK or access number was not given, we will be assigning a default CIK and access number of IBM.
   1. We also deal with the cease if there is invalid url or 10-Q url and these cases will show Warning message in log file and exit.
   2. We also clear the log file every time the code is run.
4. After looking at the specific 10-Q file and generating url, we generalized by using the cik and access number, similar to the main url.
   1. We have also dealt with the case where there might be multiple 10-Q files and created a loop. This way we will be reading each 10-Q file.
5. We used BeautifulSoup in dealing with the 10-Q file to get the tables and format it.
   1. We first grabbed all tables in the 10-Q file by finding the div table tags in the html.
   2. We then looked at the html page and noticed all the tables which have financial data (numerical data) had tables with % and $ signs. When looking at the htm markup, we noticed that there are tables that are really just text and those are ones we do not want.
   3. So we looped through the table and for each table we are looking at the row and certain cell to see if the table has $ or % and if and only if a table had either, we added the table to table.append.
   4. Next step was to clean up and retrieve text from the cells and make a table from it. Then put the table in a CSV. So each table had its own CSV.

# Steps to Run the code