GC-28 Modern Web Scraping

Abstract

The Capstone group was tasked to scrape data of key professionals of more than 2,000 organizations from the open990.org website. The website presented several challenges that were well beyond the standard methods for scraping HTML and Javascript websites. In addition to the challenge of scraping the data, there was not a standard data structure for each organization, so the second challenge was to wrangle data of many different structures into a single, structured format. Finally, the group had to store the data for accessibility and analysis. Our pipeline to scrape and process the data is an example of how to scrape data from websites in a modern, secure environment.

Introduction

The aim of project is to scrape all names of key professionals from the open990.org website and insert them into a structured database for query and review. The Key Professionals dataset seeks to provide global coverage of key investor and analyst professionals engaged in investment decisions, beginning with companies based in the U.S. The eVestment network includes coverage for US investors and consultants. The goal of this initiative is to establish a one-stop center for institutional asset management delivery intelligence. The key professional database work involves creating a web crawler to extract information from the open990 website, wrangling the data into the desired structure, and inserting it into a database for data review.

Project Objective(s)

As the project progressed, the initial objectives were modified. It was discovered that the website is delivered through Cloudflare servers, which we believe blocked some of our attempts to scrape the data. The project objectives were modified to build the webcrawler using Python, which would scrape the JSON data from the website's backend API. The target data crawled were the key professional names and organization name for each EIN. Then the data was organized into the key professional database. Basic analysis and observation was done with database queries and data visualization using MongoDB charts.

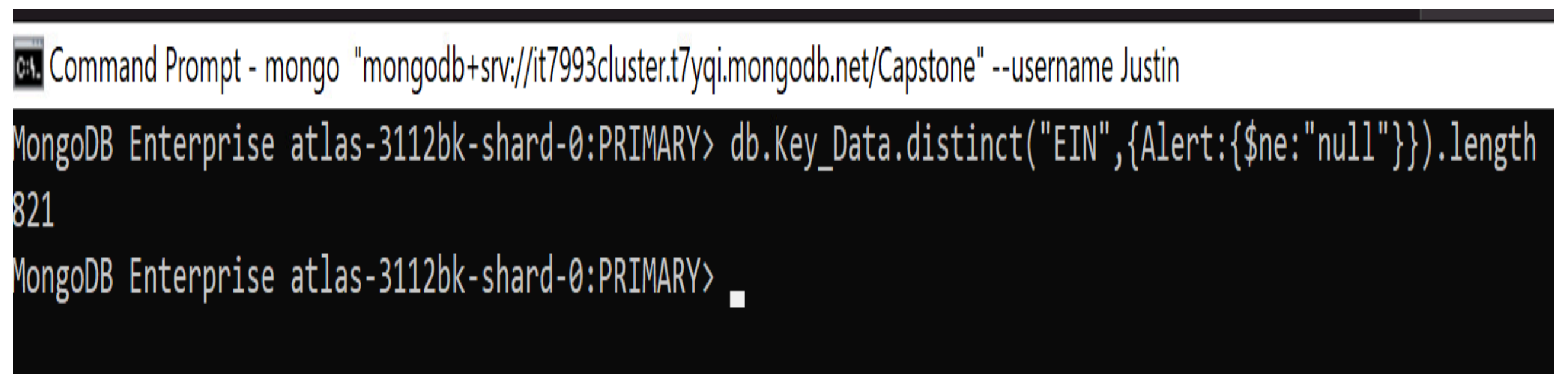
Challenges

The Team tried multiple applications to parse the data from extracted JSON files. The first application that the Team attempted was Splash using Scrapy. The data did not parse well for most of the 2,667 organizations provided. The second application attempted to use for parsing was Selenium where we took the position of the data in scrapy through Visual Studio Code. Using Selenium, we were faced with the different formats of the targets provided by the sponsor. The Team decided to use API Web Scraping where we were able to receive the data in formatted JSON files, parsed through the Alteryx application, and uploaded into the group MongoDB Database.

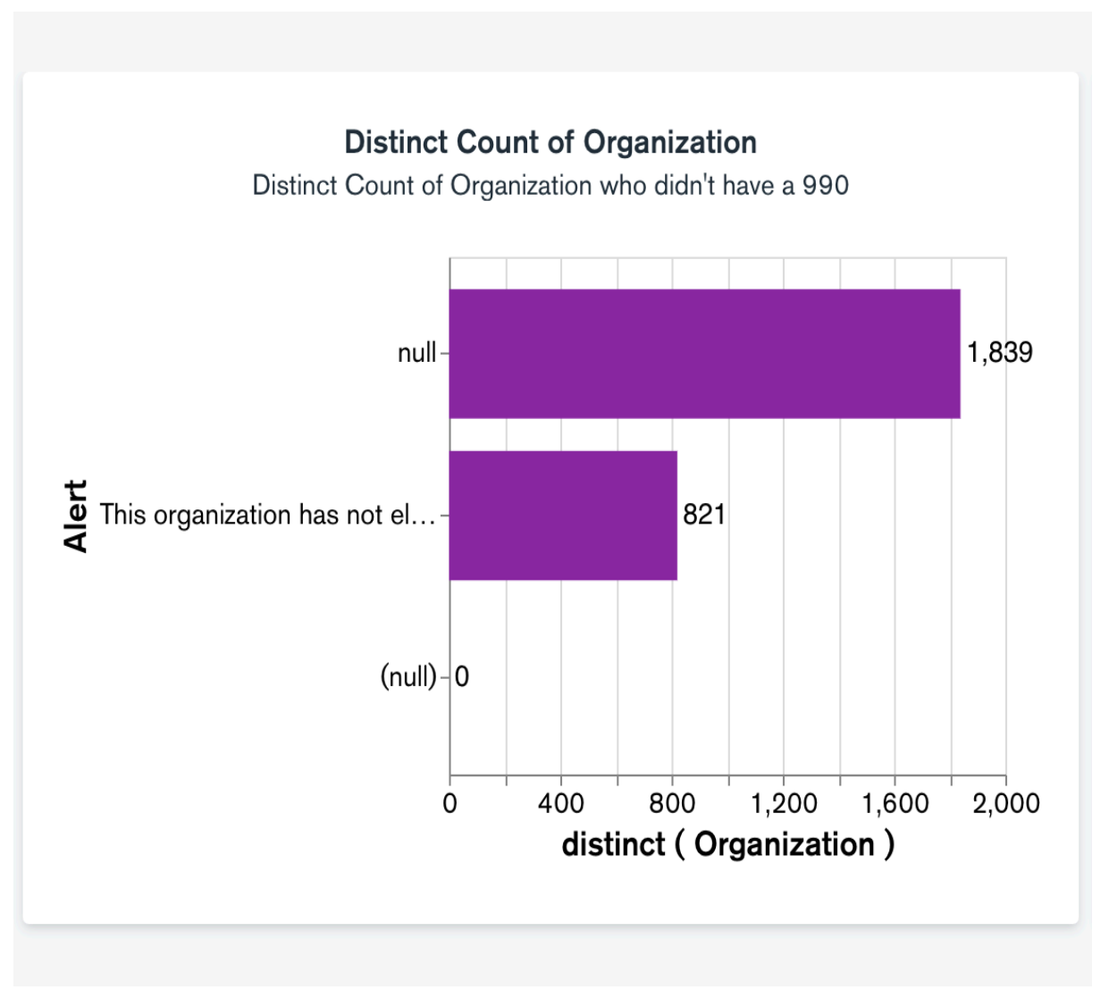
Outcomes

As a result of using Alteryx for our JSON files we uploaded all our data into the group MongoDB Database. Below are some queries and visualization of our data.

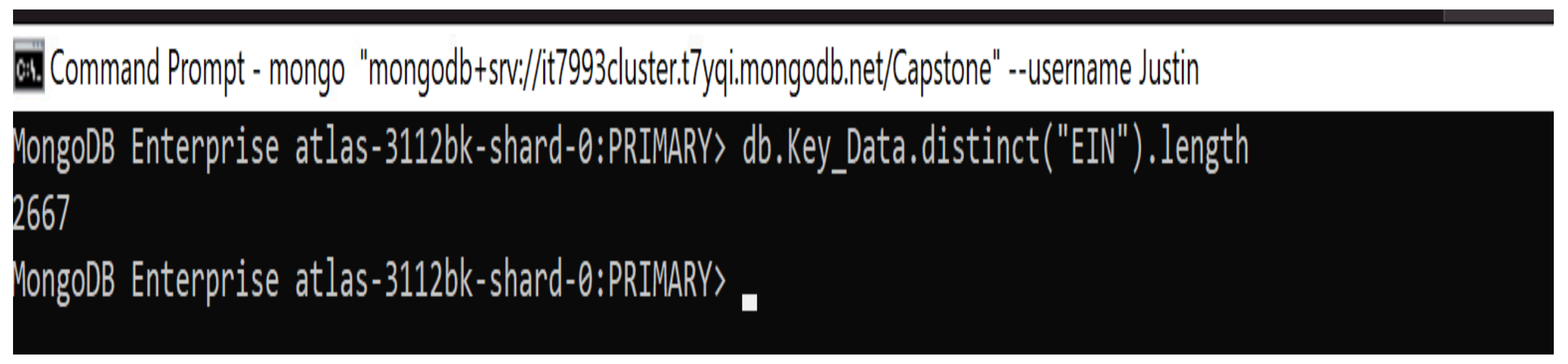
- Count of EIN numbers whose Alert = This organization hasn't filed
- db.Key_Data.distinct("EIN").length



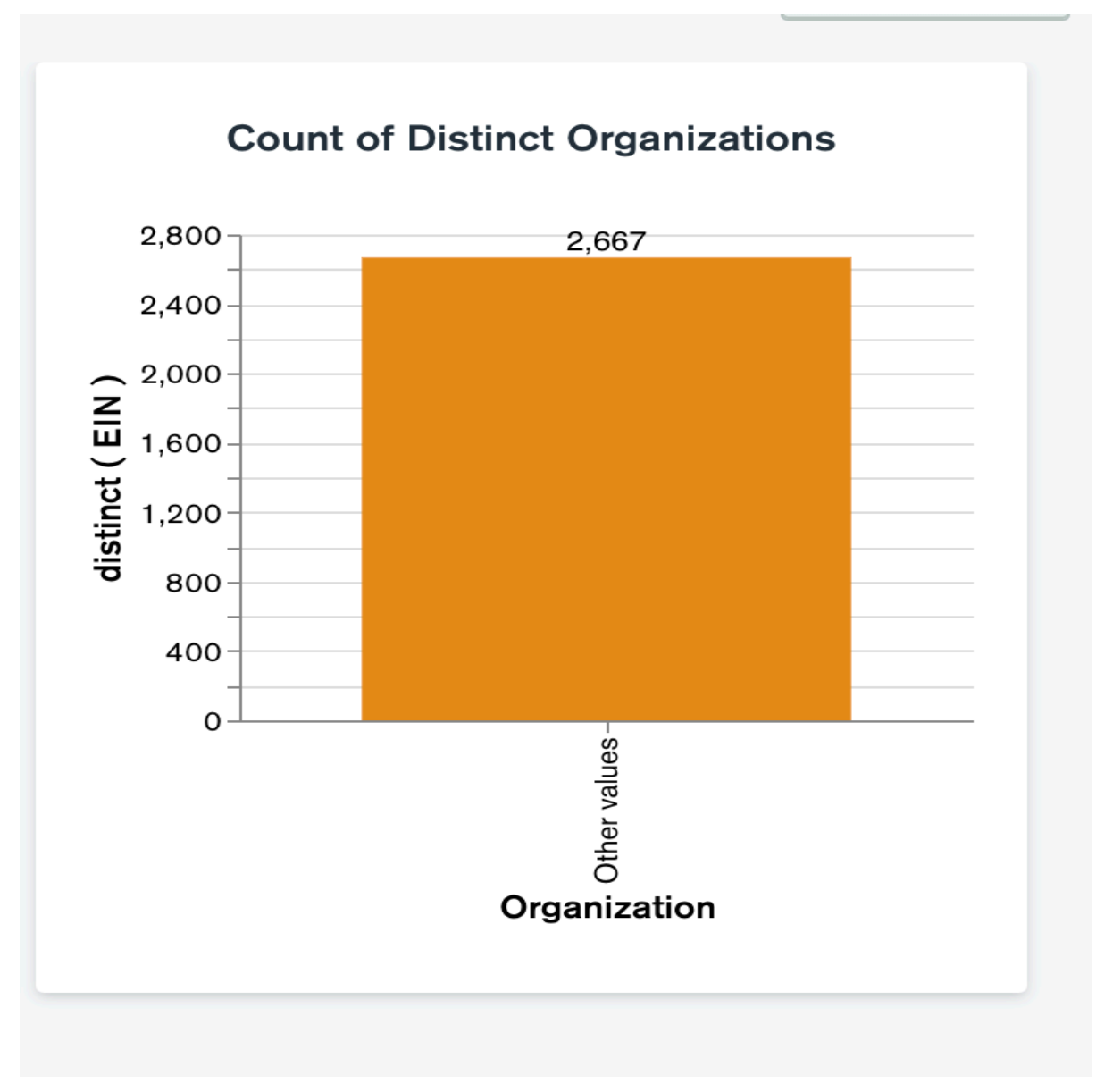
```
Command Prompt - mongo "mongodb+srv://it7993cluster.t7yqi.mongodb.net/Capstone" --username Justin
MongoDB Enterprise atlas-3112bk-shard-0:PRIMARY> db.Key_Data.distinct("EIN",{Alert:{$ne:"null"}}).length
821
MongoDB Enterprise atlas-3112bk-shard-0:PRIMARY>
```



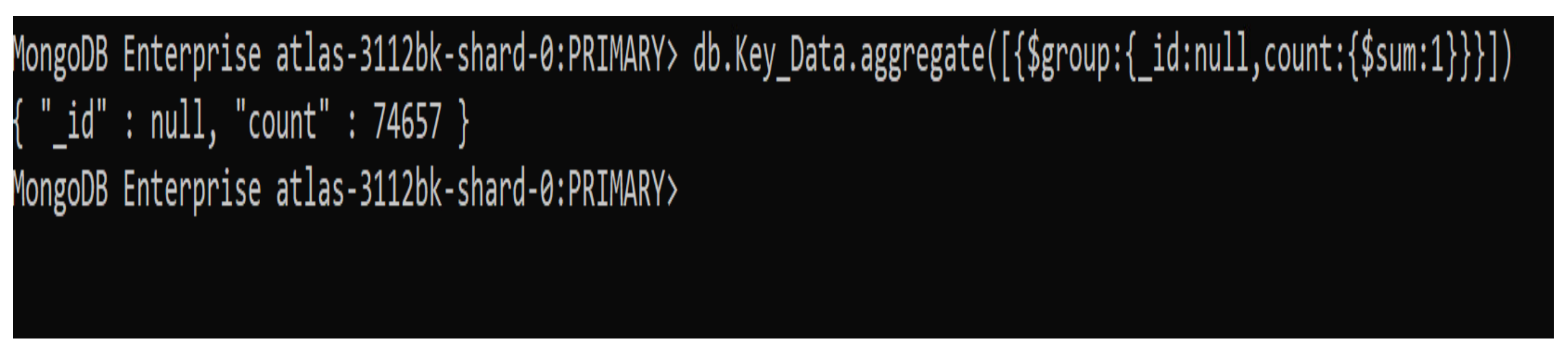
- Count of EIN number db.Key_Data.distinct("EIN").length



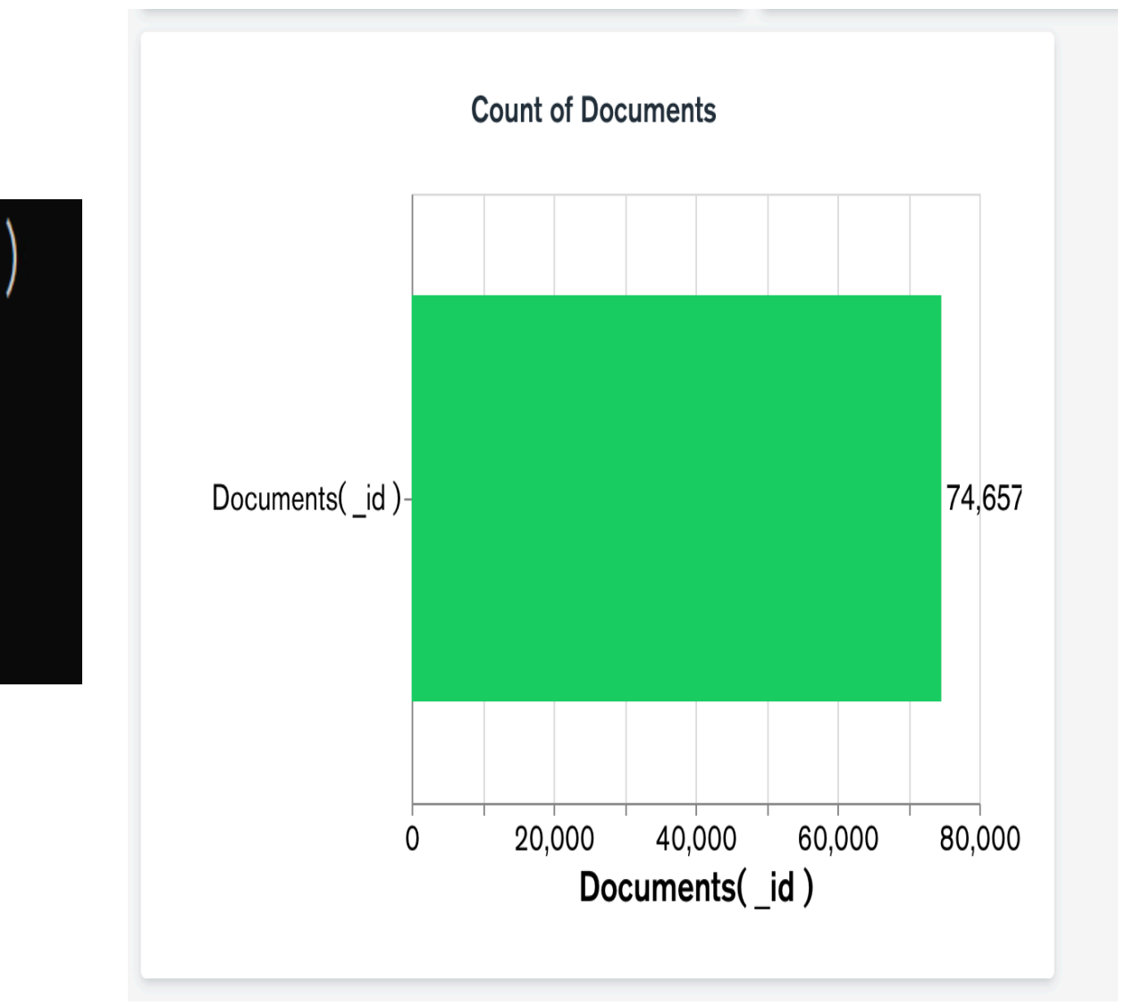
```
Command Prompt - mongo "mongodb+srv://it7993cluster.t7yqi.mongodb.net/Capstone" --username Justin
MongoDB Enterprise atlas-3112bk-shard-0:PRIMARY> db.Key_Data.distinct("EIN").length
2667
MongoDB Enterprise atlas-3112bk-shard-0:PRIMARY>
```



- Count of All Documents db.Key_data.aggregate([{$group:{_id:null,count:{$sum:1}}}])



```
MongoDB Enterprise atlas-3112bk-shard-0:PRIMARY> db.Key_Data.aggregate([{$group:{_id:null,count:{$sum:1}}}])
{ "_id" : null, "count" : 74657 }
MongoDB Enterprise atlas-3112bk-shard-0:PRIMARY>
```



Conclusions

The problem of scraping data from the website turned out to be complex due to the structure of the website. The website was coded entirely in JavaScript, the data was delivered from a backend API, and the website was delivered from Cloudflare servers with security services. While not impossible to scrape, we had to use a bespoke approach, and this kind of design causes problems for generic web scrapers to crawl the web in general retrieving as much data as possible. In most projects, finding the right applications are necessary to meeting the needs of the tasks. We arrived at our solution by attempting multiple applications in the development phase, working in an iterative process of development, testing, and review with each application. The solution in the end was to scrape the API and parse the data through Alteryx to extract the target information. The database selection of MongoDB provided us with a storage solution with a terminal shell for queries and data analysis to extract data from 2,667 organizations within the open990 website.

Acknowledgments

Ms. Jing Wang, eVestments

Dr. Meng Han, Kennesaw State University

Dr. Lei Li, Kennesaw State University

Department of Information Technology, Kennesaw State University

Contact Information

Sandya Bantu - https://www.linkedin.com/in/sbantu/
Justin Bridges -.linkedin.com/in/justin-bridges-14124a10b/
Joselyn Giron Rivera - https://www.linkedin.com/in/joselyn-giron-964ab3104/
Kenny Randolph – linkedin.com/in/kenny-randolph/
Denise Tucker - linkedin.com/in/denise-tucker-7496617

References

Hong, L. (2020, February 02). How to build a simple web crawler. Retrieved April 08, 2021, from https://towardsdatascience.com/how-to-build-a-simple-web-crawler-66082fc82470

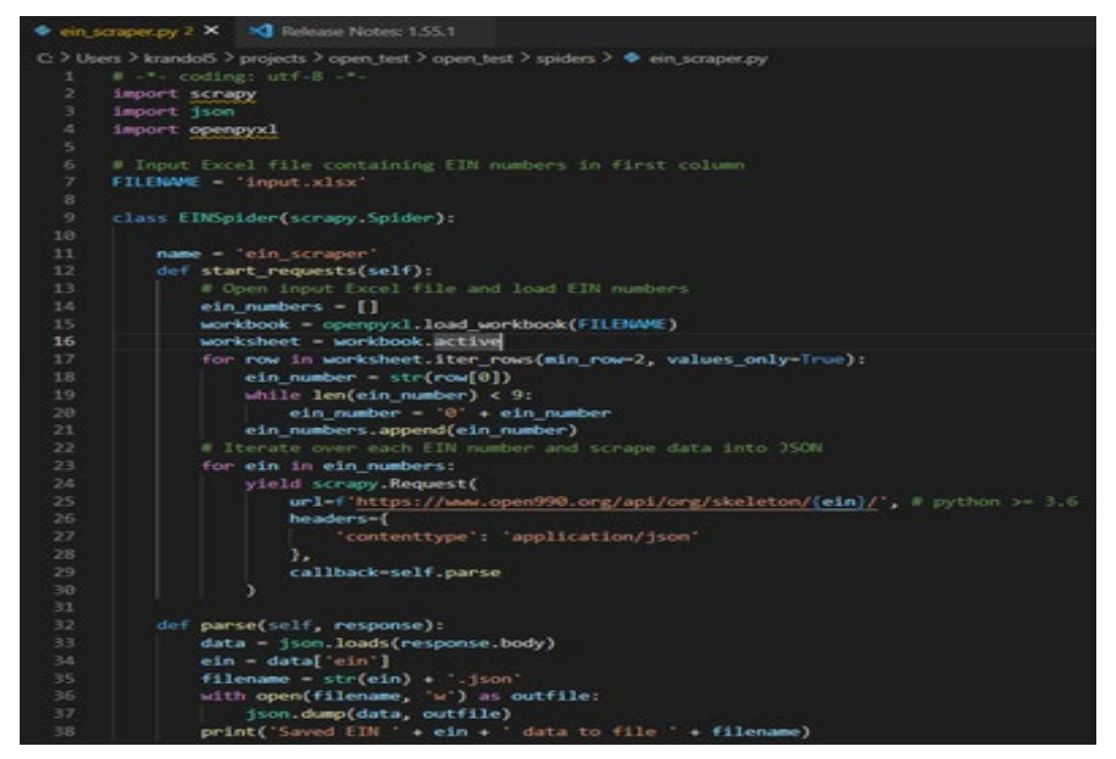
Here at Cloudflare, we make the internet work the way it Should. OFFERING CDN, DNS, DDoS protection and Security, find out how we can help your site. (n.d.). Retrieved April 09, 2021, from https://www.cloudflare.com/

Jabeen, H .(2019). Making web crawlers using scrapy for python. Retrieved from April 8, 2021 https://www.datacamp.com/community/tutorials/making-web-crawlers-scrapy-python

MongoDB. Aggregation. Retrieved April 8.2021 from https://docs.mongodb.com/manual/aggregation/.

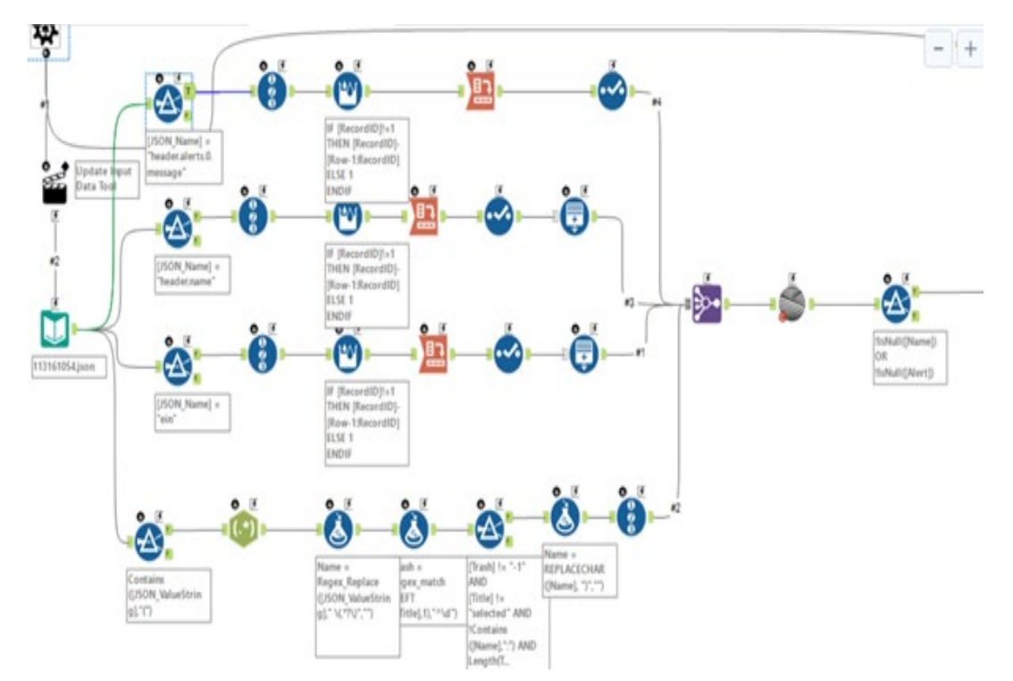
Materials and Methods – The Data Pipeline

Python Webscraper

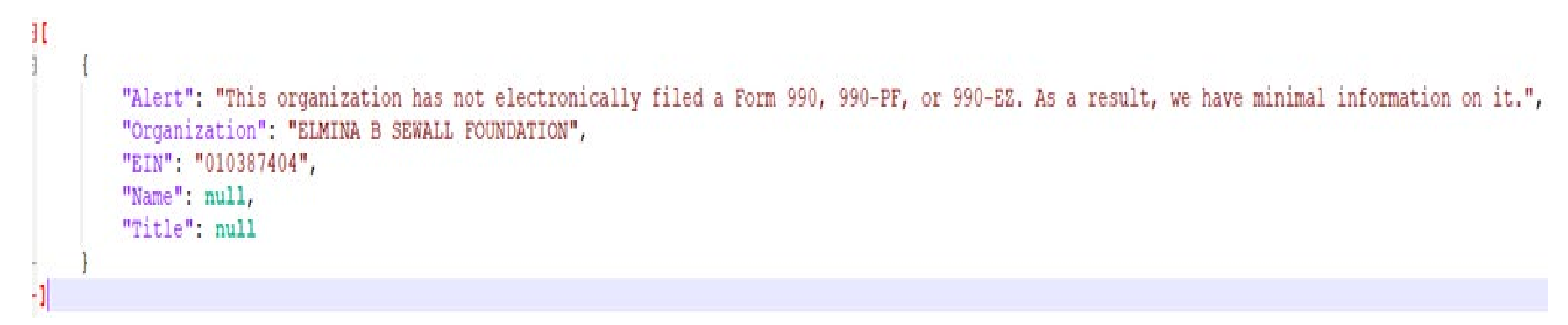


Raw JSON Files

Data Processing through Alteryx

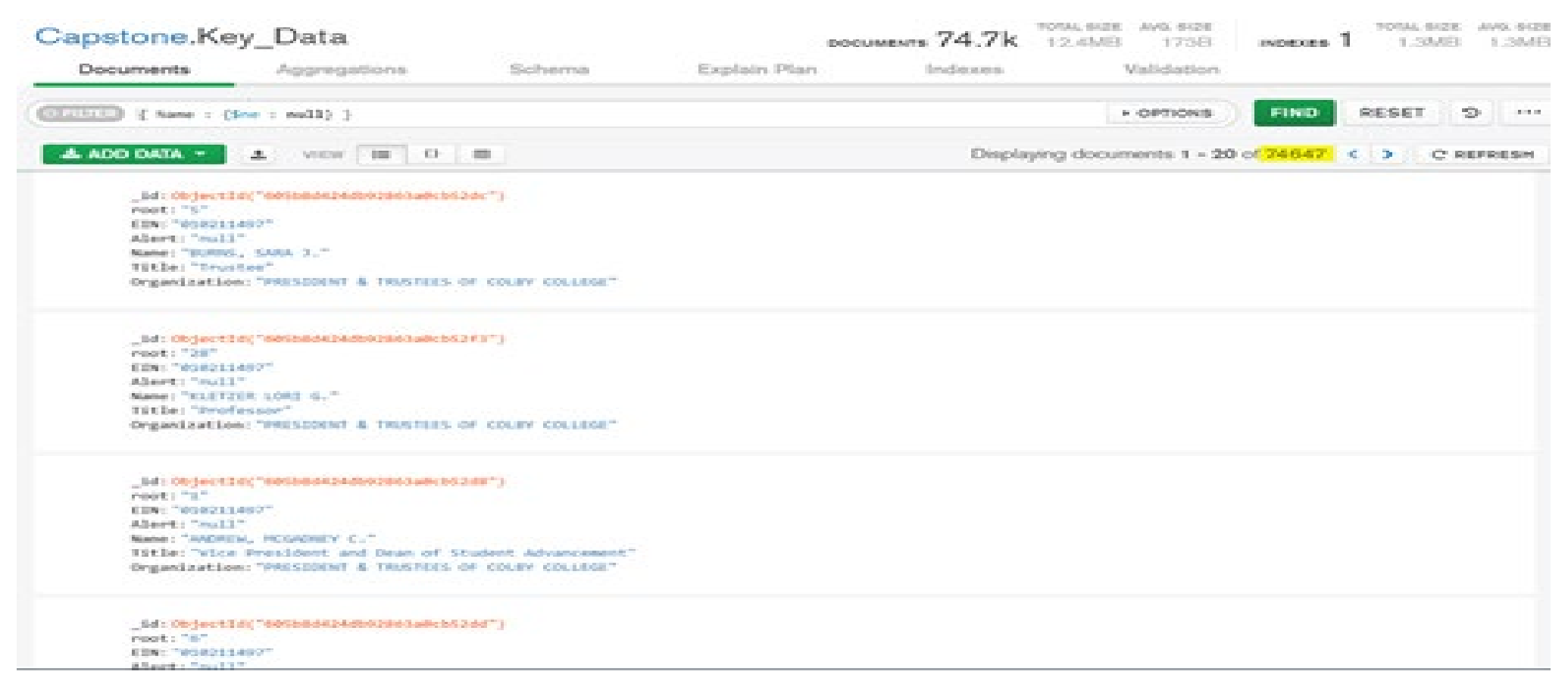


Processed JSON Alteryx Output



```
[
    {
        "Alert": "This organization has not electronically filed a Form 990, 990-PF, or 990-EZ. As a result, we have minimal information on it.",
        "Organization": "ELMINA B SEWALL FOUNDATION",
        "EIN": "010387404",
        "Name": null,
        "Title": null
    }
]
```

Key Professional Database in MongoDB



KENNESAW STATE UNIVERSITY
COLLEGE OF COMPUTING AND SOFTWARE ENGINEERING

Author(s): Sandhya Bantu, Justin Bridges, Joselyn Giron Rivera, Kenny Randolph, Denise Tucker
Advisors(s): Jing Wang, Dr Meng Han