Chipotle Location Data Management

Nick Videtti

IST-769 Advanced Big Data Management

Final Project

Fall 2022

Data

The data used for this project were the data found on Kaggle for Chipotle locations in the United States. The key to this is having the user download the data file to a file on their computer. The data file can be found attached to this assignment or at https://www.kaggle.com/datasets/jeffreybraun/chipotle-locations.

Two files are included in these data. One of the files is a CSV file with Chipotle store locations, and the other is a JSON file containing locations of state borders for states with Chipotle locations. Both will be used in this project.

Project Goals

This project will cover the following topics. Each of these steps will be discussed and brief descriptions/reflections will be included. Full Python code can be found in the Jupyter Notebook attached to this assignment.

1. Configure PySpark session for both MongoDB and Elasticsearch
2. Use PySpark to read in raw data from downloaded files
3. Write raw data to MongoDB, then read it back into PySpark
4. Use PySpark to clean the raw data
5. Write cleaned data to MongoDB, then read it back into PySpark
6. Use PySpark to combine the two cleaned data sets
7. Create PySpark data frames for only coordinate data from the combined cleaned data, the cleaned location data, and the cleaned state borders data
8. Write the data frames created in steps 6 and 7 to MongoDB, then read them back into PySpark
9. Write all 8 data frames to Elasticsearch, then read them back into PySpark
10. Use Drill to query the cleaned location data using a few queries
11. Create a Kibana Map visualization using the coordinate-only location data

1 - Configure PySpark session for both MongoDB and Elasticsearch

Configuring a PySpark session for both MongoDB and Elasticsearch involved the following configuration options:

- MongoDB
    - *.config("spark.mongodb.input.uri", "mongodb://admin:mongopw@mongo:27017/admin?authSource=admin")* \
    - *.config("spark.mongodb.output.uri", "mongodb://admin:mongopw@mongo:27017/admin?authSource=admin")* \
    - *.config("spark.jars.packages","org.mongodb.spark:mongo-spark-connector_2.12:3.0.1")*\
- Elasticsearch
    - *.config("spark.es.nodes", "elasticsearch")* \
    - *.config("spark.es.port","9200")* \

This was different from other assignments during the course due to needing to configure for multiple databases. I first attempted separate PySpark sessions for each of the two databases, then attempted to consolidate to one PySpark session, and ultimately took that latter approach.

(See attached Jupyter Notebook for full code)

2 - Use PySpark to read in raw data from downloaded files

Both of the files were read into PySpark via local system files downloaded from Kaggle. The CSV file needed the “header” option set to “True”. After the data were read in, the first 10 rows of each resulting data frame were displayed to prove that the data were successfully read into PySpark. The JSON file with state borders was named “RAW_CHIPOTLE_STATE_BORDERS” and the CSV file with locations was named “RAW_CHIPOTLE_LOCATIONS”.

First 10 rows of RAW_CHIPOTLE_STATE_BORDERS read in from downloaded file

```
+--------------------+--------------------+----+-------------+-------+
|     _corrupt_record|            geometry|  id|   properties|   type|
+--------------------+--------------------+----+-------------+-------+
|{"type":"FeatureC...|                null|null|         null|   null|
|                null|{[[[-87.359296, 3...|  AL|    {Alabama}|Feature|
|                null|{[[[[-131.602021,...|  AK|     {Alaska}|Feature|
|                null|{[[[-109.042503, ...|  AZ|    {Arizona}|Feature|
|                null|{[[[-94.473842, 3...|  AR|   {Arkansas}|Feature|
|                null|{[[[-123.233256, ...|  CA| {California}|Feature|
|                null|{[[[-107.919731, ...|  CO|   {Colorado}|Feature|
|                null|{[[[-73.053528, 4...|  CT|{Connecticut}|Feature|
|                null|{[[[-75.414089, 3...|  DE|   {Delaware}|Feature|
|                null|{[[[-85.497137, 3...|  FL|    {Florida}|Feature|
+--------------------+--------------------+----+-------------+-------+
only showing top 10 rows
```

First 10 rows of RAW_CHIPOTLE_LOCATIONS read in from downloaded file

```
+-------+----------+--------------------+------------------+------------------+
|  state|  location|             address|          latitude|         longitude|
+-------+----------+--------------------+------------------+------------------+
|Alabama|    Auburn|346 W Magnolia Av...|32.606812966051244|-85.48732833164195|
|Alabama|Birmingham|300 20th St S Bir...|33.509721495414745|-86.80275567068401|
|Alabama|Birmingham|3220 Morrow Rd Bi...| 33.59558141391436|-86.64743684970283|
|Alabama|Birmingham|4719 Highway 280 ...| 33.42258214624579| -86.6982794650297|
|Alabama|   Cullman|1821 Cherokee Ave...| 34.15413376734492|-86.84122007667406|
|Alabama|    Hoover|1759 Montgomery H...|33.378958029568594|-86.80380210088629|
|Alabama|Huntsville|5900 University D...|34.742319254429496| -86.6657204641674|
|Alabama|    Mobile|3871 Airport Blvd...|30.675337809949887|  -88.143753929995|
|Alabama|    Mobile|7765 Airport Blvd...| 30.68273057569605|-88.22499815689844|
|Alabama|Montgomery|2560 Berryhill Rd...| 32.35917687650774|-86.16225285227608|
+-------+----------+--------------------+------------------+------------------+
only showing top 10 rows
```

(See attached Jupyter Notebook for full code)

3 - Write raw data to MongoDB, then read it back into PySpark

Both RAW_CHIPOTLE_STATE_BORDERS and RAW_CHIPOTLE_LOCATIONS were written to MongoDB into a database named “project” under the collections “raw_chipotle_state_borders” and “raw_chipotle_locations”, respectively. Then, the data were read back into PySpark from MongoDB and the first 10 rows of each data frame were displayed to prove that the data were successfully read into PySpark.

First 10 rows of RAW_CHIPOTLE_STATE_BORDERS read in from MongoDB

```
JSON file from MongoDB
+--------------------+--------------------+--------------------+----+-------------+-------+
|     _corrupt_record|                 _id|            geometry|  id|   properties|   type|
+--------------------+--------------------+--------------------+----+-------------+-------+
|{"type":"FeatureC...|{63a10577b40f1b1d...|                null|null|         null|   null|
|                null|{63a10577b40f1b1d...|{[[[-87.359296, 3...|  AL|    {Alabama}|Feature|
|                null|{63a10577b40f1b1d...|{[[[[-131.602021,...|  AK|     {Alaska}|Feature|
|                null|{63a10577b40f1b1d...|{[[[-109.042503, ...|  AZ|    {Arizona}|Feature|
|                null|{63a10577b40f1b1d...|{[[[-94.473842, 3...|  AR|   {Arkansas}|Feature|
|                null|{63a10577b40f1b1d...|{[[[-123.233256, ...|  CA| {California}|Feature|
|                null|{63a10577b40f1b1d...|{[[[-107.919731, ...|  CO|   {Colorado}|Feature|
|                null|{63a10577b40f1b1d...|{[[[-73.053528, 4...|  CT|{Connecticut}|Feature|
|                null|{63a10577b40f1b1d...|{[[[-75.414089, 3...|  DE|   {Delaware}|Feature|
|                null|{63a10577b40f1b1d...|{[[[-85.497137, 3...|  FL|    {Florida}|Feature|
+--------------------+--------------------+--------------------+----+-------------+-------+
only showing top 10 rows
```

(See attached Jupyter Notebook for full code)

4 - Use PySpark to clean the raw data

Both data frames were duplicated as temporary views and then Spark SQL was used to generate cleaned datasets, named CHIPOTLE_STATE_BORDERS and CHIPOTLE_LOCATIONS. The following transformations were made.

- RAW_CHPOTLE_STATE_BORDERS -> CHIPOTLE_STATE_BORDERS
    - Select only “geometry” and “properties” columns
    - Explode columns until no rows are nested
    - Rename fields “COORDINATES” and “STATE”

- RAW_CHIPOTLE_LOCATIONS -> CHIPOTLE_LOCATIONS
    - Select “state”, “location”, “location” and “, “ and “state” concatenated, “address”, “longitude” and “ - “ and “latitude concatenated, “longitude”, and “latitude”
    - Rename columns listed above to “STATE”, “CITY”, “CITY_STATE”, “ADDRESS”, “LON”, “LAT”, “LON_LAT”, respectively

Schema of RAW_CHIPOTLE_STATE_BORDERS

```
root
 |-- _corrupt_record: string (nullable = true)
 |-- geometry: struct (nullable = true)
 |    |-- coordinates: array (nullable = true)
 |    |    |-- element: array (containsNull = true)
 |    |    |    |-- element: array (containsNull = true)
 |    |    |    |    |-- element: string (containsNull = true)
 |    |-- type: string (nullable = true)
 |-- id: string (nullable = true)
 |-- properties: struct (nullable = true)
 |    |-- name: string (nullable = true)
 |-- type: string (nullable = true)
```

First 10 rows of CHIPOTLE_STATE_BORDERS

```
+--------------------+----------+
|         COORDINATES|     STATE|
+--------------------+----------+
|[-134.499322,57.0...|    Alaska|
|[-151.116359,59.7...|    Alaska|
|[-162.14144,66.07...|    Alaska|
|[-160.837928,70.3...|    Alaska|
|[-155.881297,19.0...|    Hawaii|
|[-75.972737,39.55...|  Maryland|
|[-77.018833,38.44...|  Maryland|
|[-88.416346,47.37...|  Michigan|
|[-117.498899, 37....|California|
|[-80.381674, 27.7...|   Florida|
+--------------------+----------+
only showing top 10 rows
```

First 10 rows of CHIPOTLE_LOCATIONS

```
+-------+----------+------------------+--------------------+--------------------+------------------+------------------+
|  STATE|      CITY|        CITY_STATE|             ADDRESS|             LON_LAT|               LON|               LAT|
+-------+----------+------------------+--------------------+--------------------+------------------+------------------+
|Alabama|    Auburn|    Auburn, Alabama|346 W Magnolia Av...|-85.4873283316419...|-85.48732833164195|32.606812966051244|
|Alabama|Birmingham|Birmingham, Alabama|300 20th St S Bir...|-86.8027556706840...|-86.80275567068401|33.509721495414745|
|Alabama|Birmingham|Birmingham, Alabama|3220 Morrow Rd Bi...|-86.6474368497028...|-86.64743684970283| 33.59558141391436|
|Alabama|Birmingham|Birmingham, Alabama|4719 Highway 280 ...|-86.6982794650297...| -86.6982794650297| 33.42258214624579|
|Alabama|   Cullman|   Cullman, Alabama|1821 Cherokee Ave...|-86.8412200766740...|-86.84122007667406| 34.15413376734492|
|Alabama|    Hoover|    Hoover, Alabama|1759 Montgomery H...|-86.8038021008862...|-86.80380210088629|33.378958029568594|
|Alabama|Huntsville|Huntsville, Alabama|5900 University D...|-86.6657204641674...| -86.6657204641674|34.742319254429496|
|Alabama|    Mobile|    Mobile, Alabama|3871 Airport Blvd...|-88.143753929995,...|  -88.143753929995|30.675337809949887|
|Alabama|    Mobile|    Mobile, Alabama|7765 Airport Blvd...|-88.2249981568984...|-88.22499815689844| 30.68273057569605|
|Alabama|Montgomery|Montgomery, Alabama|2560 Berryhill Rd...|-86.1622528522760...|-86.16225285227608| 32.35917687650774|
+-------+----------+------------------+--------------------+--------------------+------------------+------------------+
only showing top 10 rows
```

(See attached Jupyter Notebook for full code)

5 - Write cleaned data to MongoDB, then read it back into PySpark

Both CHIPOTLE_STATE_BORDERS and CHIPOTLE_LOCATIONS were written to MongoDB into a database named “project” under the collections “chipotle_state_borders” and “chipotle_locations”, respectively. Then, the data were read back into PySpark from MongoDB and the first 10 rows of each data frame were displayed to prove that the data were successfully read into PySpark.

First 10 rows of CHIPOTLE_STATE_BORDERS read in from MongoDB

```
Cleaned JSON file from MongoDB
+--------------------+----------+--------------------+
|         COORDINATES|     STATE|                 _id|
+--------------------+----------+--------------------+
|[-134.499322,57.0...|    Alaska|{63a105e1b40f1b1d...|
|[-151.116359,59.7...|    Alaska|{63a105e1b40f1b1d...|
|[-162.14144,66.07...|    Alaska|{63a105e1b40f1b1d...|
|[-160.837928,70.3...|    Alaska|{63a105e1b40f1b1d...|
|[-155.881297,19.0...|    Hawaii|{63a105e1b40f1b1d...|
|[-75.972737,39.55...|  Maryland|{63a105e1b40f1b1d...|
|[-77.018833,38.44...|  Maryland|{63a105e1b40f1b1d...|
|[-88.416346,47.37...|  Michigan|{63a105e1b40f1b1d...|
|[-117.498899, 37....|California|{63a105e1b40f1b1d...|
|[-80.381674, 27.7...|   Florida|{63a105e1b40f1b1d...|
+--------------------+----------+--------------------+
only showing top 10 rows
```

First 10 rows of CHIPOTLE_LOCATIONS read in from MongoDB

```
Cleaned CSV file from MongoDB
+--------------------+----------+------------------+------------------+------------------+--------------------+-------+--------------------+
|             ADDRESS|      CITY|        CITY_STATE|               LAT|               LON|             LON_LAT|  STATE|                 _id|
+--------------------+----------+------------------+------------------+------------------+--------------------+-------+--------------------+
|346 W Magnolia Av...|    Auburn|    Auburn, Alabama|32.606812966051244|-85.48732833164195|-85.4873283316419...|Alabama|{63a105e6b40f1b1d...|
|300 20th St S Bir...|Birmingham|Birmingham, Alabama|33.509721495414745|-86.80275567068401|-86.8027556706840...|Alabama|{63a105e6b40f1b1d...|
|3220 Morrow Rd Bi...|Birmingham|Birmingham, Alabama| 33.59558141391436|-86.64743684970283|-86.6474368497028...|Alabama|{63a105e6b40f1b1d...|
|4719 Highway 280 ...|Birmingham|Birmingham, Alabama| 33.42258214624579| -86.6982794650297|-86.6982794650297...|Alabama|{63a105e6b40f1b1d...|
|1821 Cherokee Ave...|   Cullman|   Cullman, Alabama| 34.15413376734492|-86.84122007667406|-86.8412200766740...|Alabama|{63a105e6b40f1b1d...|
|1759 Montgomery H...|    Hoover|    Hoover, Alabama|33.378958029568594|-86.80380210088629|-86.8038021008862...|Alabama|{63a105e6b40f1b1d...|
|5900 University D...|Huntsville|Huntsville, Alabama|34.742319254429496| -86.6657204641674|-86.6657204641674...|Alabama|{63a105e6b40f1b1d...|
|3871 Airport Blvd...|    Mobile|    Mobile, Alabama|30.675337809949887|  -88.143753929995|-88.143753929995,...|Alabama|{63a105e6b40f1b1d...|
|7765 Airport Blvd...|    Mobile|    Mobile, Alabama| 30.68273057569605|-88.22499815689844|-88.2249981568984...|Alabama|{63a105e6b40f1b1d...|
|2560 Berryhill Rd...|Montgomery|Montgomery, Alabama| 32.35917687650774|-86.16225285227608|-86.1622528522760...|Alabama|{63a105e6b40f1b1d...|
+--------------------+----------+------------------+------------------+------------------+--------------------+-------+--------------------+
only showing top 10 rows
```

(See attached Jupyter Notebook for full code)

6 - Use PySpark to combine the two cleaned data sets

Both CHIPOTLE_STATE_BORDERS and CHIPOTLE_LOCATIONS were duplicated as temporary views. Then, Spark SQL was used to UNION both data frames. Columns that did not exist in both data frames needed to be added as NULL columns in the data where they did not exist, and some columns needed to be renamed and/or transformed in one data frame to match the other. The resulting data frame was named CHIPOTLE_COMBINED_DATA. This resulting data frame was validated by looking at the three columns that should not be null in any records, “LON”, “LAT”, and “LON_LAT” and returning the number of rows with null values in any of those 3 columns. The result was indeed 0, as was expected.

First 10 rows of CHIPOTLE_COMBNED_DATA

STATE	CITY	CITY_STATE	ADDRESS	LON_LAT	LON	LAT
California	City Of Industry	City Of Industry,...	15495 Valley Blvd...	-117.959593500000...	-117.95959350000001	34.0219656
California	Los Angeles	Los Angeles, Cali...	7660 W Sunset Blv...	-118.356905212680...	-118.35690521268083	34.0978471129027
Florida	Coral Springs	Coral Springs, Fl...	1775 N University...	-80.2523191679101...	-80.25231916791013	26.252865595198628
Iowa	Ankeny	Ankeny, Iowa	2125 SE Delaware ...	-93.5807689589128...	-93.5807689589128	41.709325487080214
New York	East Meadow	East Meadow, New ...	2312 Hempstead Tp...	-73.5505444434707...	-73.55054444347071	40.72452263660088
North Carolina	Fayetteville	Fayetteville, Nor...	4715 Ramsey St Fa...	-78.879958, 35.12...	-78.879958	35.125971
Texas	Austin	Austin, Texas	610 E Stassney Ln...	-97.7684819999999...	-97.76848199999999	30.201279
Vermont	Burlington	Burlington, Vermont	580 Shelburne Rd ...	-73.208072, 44.44...	-73.208072	44.448782
Virginia	Fairfax	Fairfax, Virginia	9506 Main St No 2...	-77.2721522, 38.8...	-77.2721522	38.842982
Wisconsin	null	null	null	-87.189511, 44.96...	-87.189511	44.969211

only showing top 10 rows

Number of rows in CHIPOTLE_COMBINED_DATA with nulls for “LON”, “LAT”, or “LON_LAT”

count(1)
0

(See attached Jupyter Notebook for full code)

7 - Create PySpark data frames for only coordinate data from the combined cleaned data, the cleaned location data, and the cleaned state borders data

Three PySpark data frames were created by selecting only the “LON_LAT” column (which was renamed from “COORDINATES” to “LON_LAT” for CHIPOTLE_STATE_BORDERS) from CHIPOTLE_STATE_BORDERS, CHIPOTLE_LOCATIONS, and CHIPOTLE_COMBINED_DATA. These were named CHIPOTLE_STATE_COORDINATES, CHIPOTLE_LOCATION_COORDINATES, and CHIPOTLE_ALL_COORDINATES, respectively.

(See attached Jupyter Notebook for full code)

8 - Write the data frames created in steps 6 and 7 to MongoDB, then read them back into PySpark

CHIPOTLE_COMBINED_DATA, CHIPOTLE_ALL_COORDINATES, CHIPOTLE_STATE_COORDINATES and CHIPOTLE_LOCATION_COORDINATES were written to MongoDB into a database named “project” under the collections “chipotle_combined_data”, “chipotle_all_coordinates”, “chipotle_state_coordinates” and “chipotle_location_coordinates”, respectively. Then, the data were read back into PySpark from MongoDB and the first 10 rows of each data frame were displayed to prove that the data were successfully read into PySpark.

First 10 rows of CHIPOTLE_COMBINED_DATA read in from MongoDB

```
Combined Data from MongoDB
+--------------------+----------------+--------------------+------------------+--------------------+--------------------+--------------+--------------------+
|             ADDRESS|            CITY|          CITY_STATE|               LAT|                LON|             LON_LAT|         STATE|                 _id|
+--------------------+----------------+--------------------+------------------+--------------------+--------------------+--------------+--------------------+
|15495 Valley Blvd...|City Of Industry|City Of Industry,...|        34.0219656|-117.95959350000001|-117.959593500000...|    California|{63a1060fb40f1b1d...|
|7660 W Sunset Blv...|     Los Angeles|Los Angeles, Cali...|  34.0978471129027|-118.35690521268083|-118.356905212680...|    California|{63a1060fb40f1b1d...|
|1775 N University...|   Coral Springs|Coral Springs, Fl...|26.252865595198628| -80.25231916791013|-80.2523191679101...|       Florida|{63a1060fb40f1b1d...|
|2125 SE Delaware ...|          Ankeny|        Ankeny, Iowa|41.709325487080214|  -93.5807689589128|-93.5807689589128...|          Iowa|{63a1060fb40f1b1d...|
|2312 Hempstead Tp...|     East Meadow|East Meadow, New ...| 40.72452263660088| -73.55054444347071|-73.5505444434707...|      New York|{63a1060fb40f1b1d...|
|4715 Ramsey St Fa...|    Fayetteville|Fayetteville, Nor...|         35.125971|         -78.879958|-78.879958, 35.12...|North Carolina|{63a1060fb40f1b1d...|
|610 E Stassney Ln...|          Austin|       Austin, Texas|         30.201279| -97.76848199999999|-97.7684819999999...|         Texas|{63a1060fb40f1b1d...|
|580 Shelburne Rd ...|      Burlington| Burlington, Vermont|         44.448782|         -73.208072|-73.208072, 44.44...|       Vermont|{63a1060fb40f1b1d...|
|9506 Main St No 2...|         Fairfax|   Fairfax, Virginia|         38.842982|        -77.2721522|-77.2721522, 38.8...|      Virginia|{63a1060fb40f1b1d...|
|                null|            null|                null|         44.969211|         -87.189511|-87.189511, 44.96...|     Wisconsin|{63a1060fb40f1b1d...|
+--------------------+----------------+--------------------+------------------+--------------------+--------------------+--------------+--------------------+
only showing top 10 rows
```

First 10 rows of CHIPOTLE_ALL_COORDINATES read in from MongoDB

```
All Coordinates from MongoDB
+--------------------+--------------------+
|             LON_LAT|                 _id|
+--------------------+--------------------+
|-117.959593500000...|{63a10618b40f1b1d...|
|-118.356905212680...|{63a10618b40f1b1d...|
|-80.2523191679101...|{63a10618b40f1b1d...|
|-93.5807689589128...|{63a10618b40f1b1d...|
|-73.5505444434707...|{63a10618b40f1b1d...|
|-78.879958, 35.12...|{63a10618b40f1b1d...|
|-97.7684819999999...|{63a10618b40f1b1d...|
|-73.208072, 44.44...|{63a10618b40f1b1d...|
|-77.2721522, 38.8...|{63a10618b40f1b1d...|
|-87.189511, 44.96...|{63a10618b40f1b1d...|
+--------------------+--------------------+
only showing top 10 rows
```

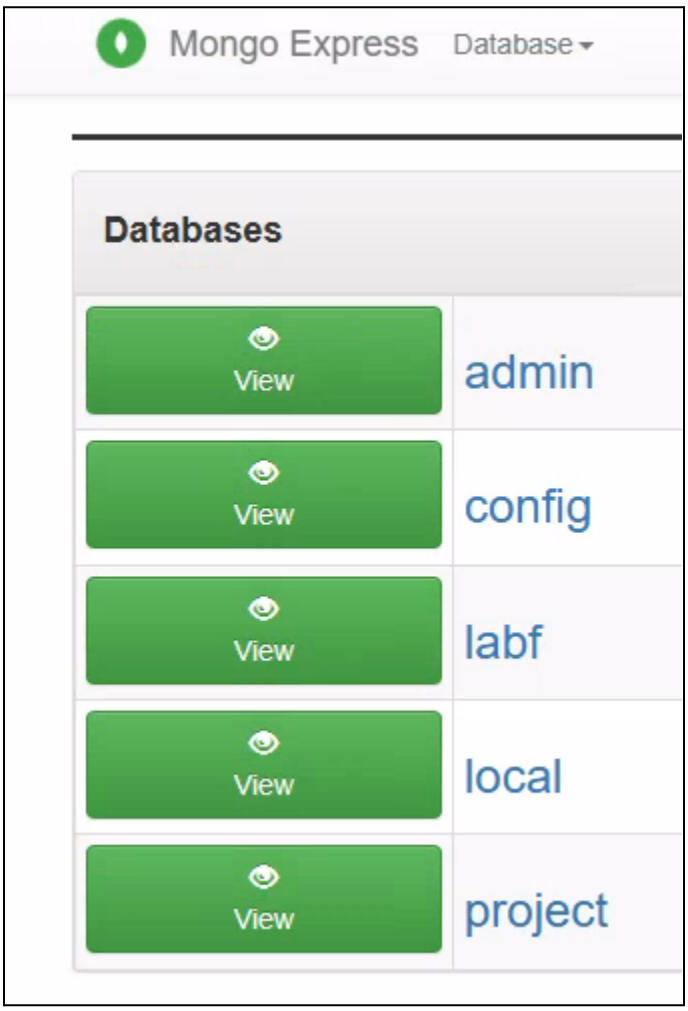
First 10 rows of CHIPOTLE_STATE_COORDINATES read in from MongoDB

```
State Coordinates from MongoDB
+--------------------+--------------------+
|             LON_LAT|                 _id|
+--------------------+--------------------+
|[-134.499322,57.0...|{63a1061cb40f1b1d...|
|[-151.116359,59.7...|{63a1061cb40f1b1d...|
|[-162.14144,66.07...|{63a1061cb40f1b1d...|
|[-160.837928,70.3...|{63a1061cb40f1b1d...|
|[-155.881297,19.0...|{63a1061cb40f1b1d...|
|[-75.972737,39.55...|{63a1061cb40f1b1d...|
|[-77.018833,38.44...|{63a1061cb40f1b1d...|
|[-88.416346,47.37...|{63a1061cb40f1b1d...|
|[-117.498899, 37....|{63a1061cb40f1b1d...|
|[-80.381674, 27.7...|{63a1061cb40f1b1d...|
+--------------------+--------------------+
only showing top 10 rows
```

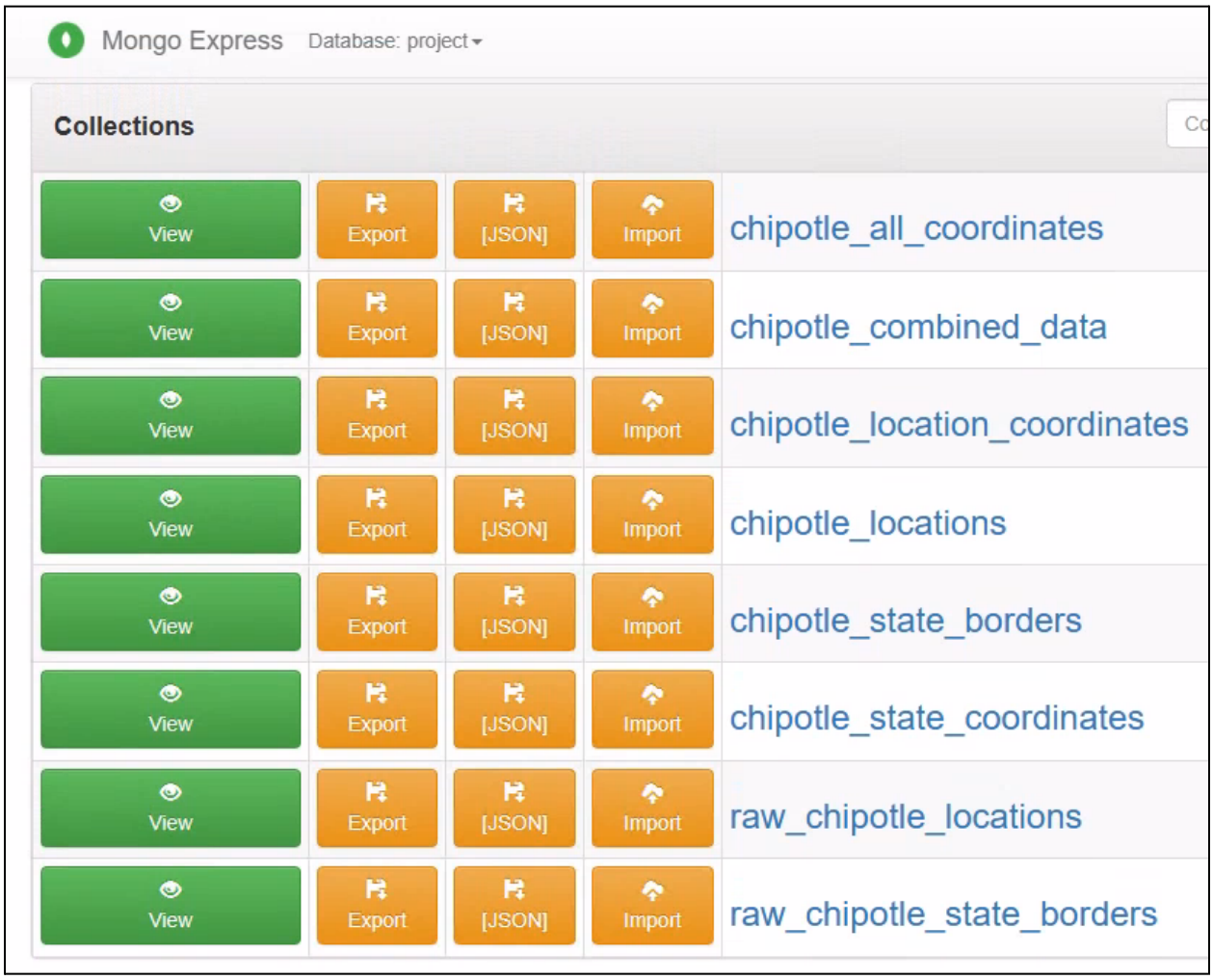
First 10 rows of CHIPOTLE_LOCATION_COORDINATES read in from MongoDB

```
Location Coordinates from MongoDB
+--------------------+--------------------+
|             LON_LAT|                 _id|
+--------------------+--------------------+
|-85.4873283316419...|{63a1061eb40f1b1d...|
|-86.8027556706840...|{63a1061eb40f1b1d...|
|-86.6474368497028...|{63a1061eb40f1b1d...|
|-86.6982794650297...|{63a1061eb40f1b1d...|
|-86.8412200766740...|{63a1061eb40f1b1d...|
|-86.8038021008862...|{63a1061eb40f1b1d...|
|-86.6657204641674...|{63a1061eb40f1b1d...|
|-88.143753929995,...|{63a1061eb40f1b1d...|
|-88.2249981568984...|{63a1061eb40f1b1d...|
|-86.1622528522760...|{63a1061eb40f1b1d...|
+--------------------+--------------------+
only showing top 10 rows
```

MongoDB Databases



MongoDB Collections in project Database



(See attached Jupyter Notebook for full code)

9 - Write all 8 data frames to Elasticsearch, then read them back into PySpark

RAW_CHIPOTLE_STATE_BORDERS, RAW_CHIPOTLE_LOCATIONS, CHIPOTLE_STATE_BORDERS, CHIPOTLE_LOCATIONS, CHIPOTLE_COMBINED_DATA, CHIPOTLE_ALL_COORDINATES, CHIPOTLE_STATE_COORDINATES and CHIPOTLE_LOCATION_COORDINATES were written to Elasticsearch under the indices “raw_chipotle_state_borders”, “raw_chipotle_locations”, “chipolte_state_borders”, “chipotle_locations”, “chipotle_combined_data”, “chipotle_all_coordinates”, “chipotle_state_coordinates” and “chipotle_location_coordinates”, respectively. Then, the data were read back into PySpark from Elasticsearch and the first row of each data frame was displayed to prove that the data were successfully read into PySpark.

First row of RAW_CHIPOTLE_STATE_BORDERS read in from Elasticsearch

```
raw_chipotle_state_borders
UNABLE TO READ "raw_chipotle_state_borders" FROM ELASTICSEARCH DUE TO FORMATTING/SCHEMA ISSUES
```

First row of RAW_CHIPOTLE_LOCATIONS read in from Elasticsearch

```
raw_chipotle_locations
+--------------------+------------------+--------+------------------+-------+
|             address|          latitude|location|         longitude|  state|
+--------------------+------------------+--------+------------------+-------+
|346 W Magnolia Av...|32.606812966051244|  Auburn|-85.48732833164195|Alabama|
+--------------------+------------------+--------+------------------+-------+
only showing top 1 row
```

First row of CHIPOTLE_STATE_BORDERS read in from Elasticsearch

```
chipotle_state_borders
+--------------------+------+
|         COORDINATES| STATE|
+--------------------+------+
|[-131.426759,55.2...|Alaska|
+--------------------+------+
only showing top 1 row
```

First row of CHIPOTLE_LOCATIONS read in from Elasticsearch

```
chipotle_locations
+--------------------+------+---------------+------------------+------------------+--------------------+-------+
|             ADDRESS|  CITY|     CITY_STATE|               LAT|               LON|             LON_LAT|  STATE|
+--------------------+------+---------------+------------------+------------------+--------------------+-------+
|346 W Magnolia Av...|Auburn|Auburn, Alabama|32.606812966051244|-85.48732833164195|-85.4873283316419...|Alabama|
+--------------------+------+---------------+------------------+------------------+--------------------+-------+
only showing top 1 row
```

First row of CHIPOTLE_COMBINED_DATA read in from Elasticsearch

```
chipotle_combined_data
+--------------------+------+---------------+----------+------------+--------------------+-------+
|            ADDRESS|  CITY|     CITY_STATE|       LAT|         LON|             LON_LAT|  STATE|
+--------------------+------+---------------+----------+------------+--------------------+-------+
|1202 W Irvington ...|Tucson|Tucson, Arizona|32.1639328|-110.9897341|-110.9897341, 32....|Arizona|
+--------------------+------+---------------+----------+------------+--------------------+-------+
only showing top 1 row
```

First row of CHIPOTLE_ALL_COORIDNATES read in from Elasticsearch

```
chipotle_all_coordinates
+--------------------+
|             LON_LAT|
+--------------------+
|-111.683468532820...|
+--------------------+
only showing top 1 row
```

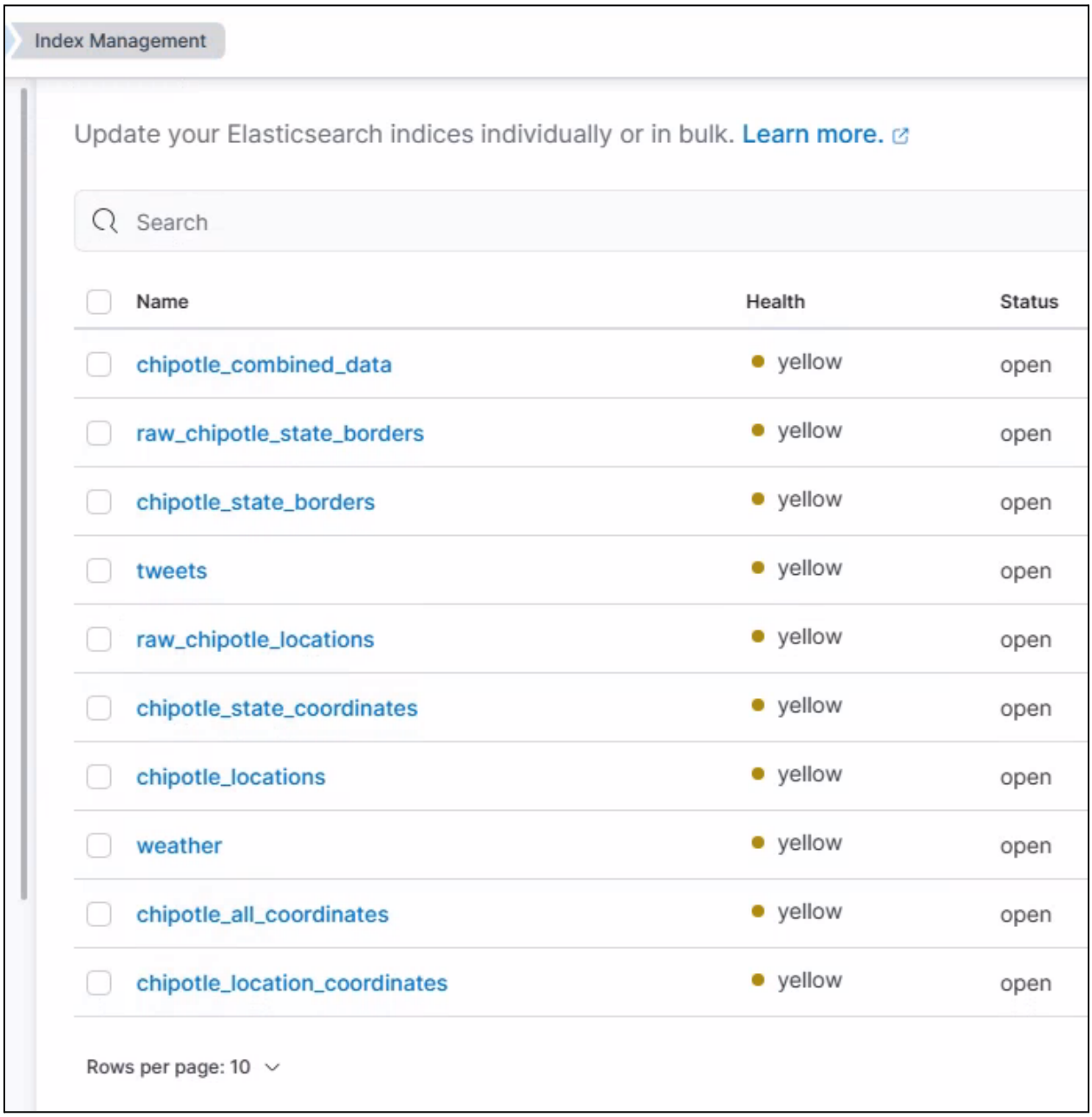
First row of CHIPOTLE_STATE_COORDINATES read in from Elasticsearch

```
chipotle_state_coordinates
+-----------+--------------------+
|COORDINATES|             LON_LAT|
+-----------+--------------------+
|       null|[-134.953908,58.4...|
+-----------+--------------------+
only showing top 1 row
```

First row of CHIPOTLE_LOCATION_COORIDNATES read in from Elasticsearch

```
chipotle_location_coordinates
+--------------------+
|             LON_LAT|
+--------------------+
|-85.4873283316419...|
+--------------------+
only showing top 1 row
```

Elasticsearch Indices



(See attached Jupyter Notebook for full code)

10 - Use Drill to query the cleaned location data using a few queries

Drill queries were run on the chipotle_locations collection from MongoDB to answer the following 3 data questions.

- What are the addresses of the Chipotle locations in Rochester, NY?
- How many Chipotle locations are there in each state, for states with at least 75 locations?
- What are the top 10 cities in the United States for most Chipotle locations, and how many Chipotle locations do those cities have?

Drill Query for Locations in Rochester, NY

Query

```
SELECT
    ADDRESS
FROM
    MONGO.PROJECT.chipotle_locations
WHERE
    CITY_STATE = 'Rochester, New York'
```

ADDRESS
1360 Mount Hope Ave Rochester, NY 14620 US
1495 E Ridge Rd Rochester, NY 14621 US
1847 W Ridge Rd Rochester, NY 14615 US
3349 Monroe Ave Rochester, NY 14618 US
640 Jefferson Rd Rochester, NY 14623 US

Showing 1 to 5 of 5 entries

Drill Query for Number of Locations per STATE for states with 75 or more locations

Query

```sql
SELECT
    STATE,
    COUNT(*) LOCATIONS
FROM
    MONGO.PROJECT.chipotle_locations
GROUP BY
    STATE
HAVING
    COUNT(*) >= 75
```

STATE	LOCATIONS
Arizona	85
California	421
Colorado	79
Florida	177
Illinois	144
Maryland	94
New York	160
Ohio	193
Pennsylvania	96
Texas	226
Virginia	107

Showing 1 to 11 of 11 entries

Drill Query for Top 10 CITY_STATE by Number of locations

Query

```sql
SELECT
    *
FROM
    (SELECT
        RANK() OVER(ORDER BY LOCATIONS DESC) RANK,
        CITY_STATE,
        LOCATIONS
    FROM
        (SELECT
            CITY_STATE,
            COUNT(*) LOCATIONS
        FROM
            MONGO.PROJECT.chipotle_locations
        GROUP BY
            CITY_STATE))
WHERE
    RANK <= 10
```

RANK	CITY_STATE	LOCATIONS
1	New York, New York	52
2	Chicago, Illinois	36
3	Houston, Texas	31
4	Washington DC, Washington DC	21
5	Los Angeles, California	20
6	Phoenix, Arizona	19
6	Columbus, Ohio	19
6	Dallas, Texas	19
6	Las Vegas, Nevada	19
10	Cincinnati, Ohio	17

Showing 1 to 10 of 10 entries

RANK	CITY_STATE
1	New York, New York
2	Chicago, Illinois
3	Houston, Texas
4	Washington DC, Washington DC
5	Los Angeles, California
6	Phoenix, Arizona
6	Columbus, Ohio
6	Dallas, Texas
6	Las Vegas, Nevada
10	Cincinnati, Ohio

Showing 1 to 10 of 10 entries

CITY_STATE	LOCATIONS
New York, New York	52
Chicago, Illinois	36
Houston, Texas	31
Washington DC, Washington DC	21
Los Angeles, California	20
Phoenix, Arizona	19
Columbus, Ohio	19
Dallas, Texas	19
Las Vegas, Nevada	19
Cincinnati, Ohio	17

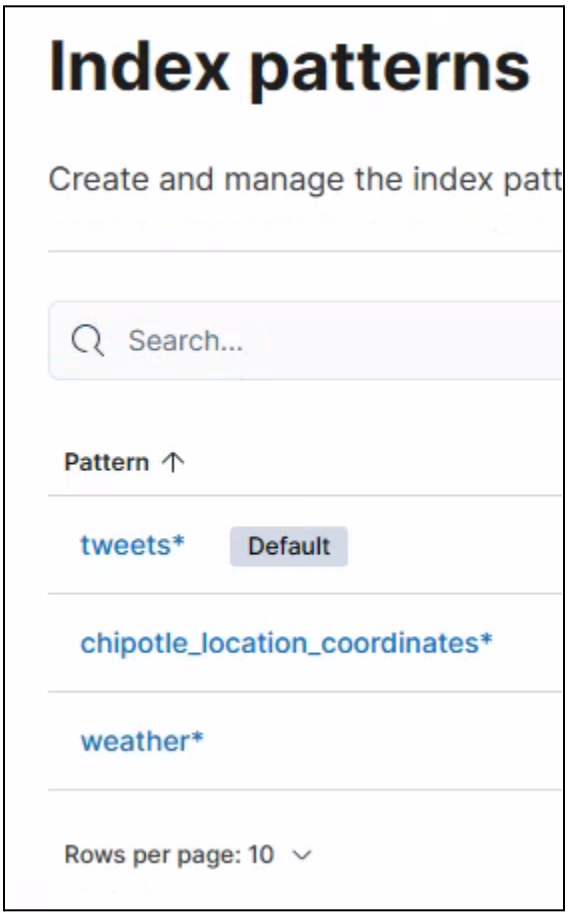
11 - Create a Kibana Map visualization using the coordinate-only location data

Using the chipotle_location_coordinates index, an index pattern was created in order to give a data source for a Kibana Map visualization. A geographic field was needed for the location coordinates, but “LON_LAT” was a string field. Kibana requires an array of two doubles, and needs reversed order from the data source since latitude needs to be before longitude in Kibana. Once this was achieved, the map visualization was created.

Calculation script for geo-point field

```
emit(
  Double.parseDouble(
  doc['LON_LAT.keyword']
  .value
  .substring(
    doc['LON_LAT.keyword']
    .value
    .indexOf(',')+1))
  ,
  Double.parseDouble(
  doc['LON_LAT.keyword']
  .value
  .substring(0,
    doc['LON_LAT.keyword']
    .value
    .indexOf','))))
```

Kibana Index Patterns



Kibana geo_point field for Map

Name

GEO_LAT_LON

Type

Geo point

Set custom label

Create a label to display in place of the field name in Discover, Maps and Visualize. Useful for shortening a long field name. Queries and filters use the original field name.

Set value

Set a value for the field instead of retrieving it from the field with the same name in `_source`.

Define script

```
emit(
    Double.parseDouble(
    doc['LON_LAT.keyword']
    .value
    .substring(
        doc['LON_LAT.keyword']
        .value
        .indexOf(',')+1))
    ,
    Double.parseDouble(
    doc['LON_LAT.keyword']
    .value
    .substring(0,
        doc['LON_LAT.keyword']
        .value
        .indexOf(','))))
```

Kibana Map