CIS4560 Term Project Tutorial

Authors: Millen Van; Nathan Kwok; Jeffrey Tam; Valerie Phanvongkham; Melody Roman; Joshue Sanchez

Instructor: Jongwook Woo

Date: 12/16/2023

Lab Tutorial

Millen Van (mvan10@calstatela.edu); Nathan Kwok (nkwok2@calstatela.edu); Jeffrey Tam (jtam2@calstatela.edu); Valerie Phanvongkham (vphanvo@calstatela.edu); Melody Roman (mroman@calstatela.edu); Joshue Sanchez (jsanch369@calstatela.edu)

12/16/2023

NYPD Ambulance Cases and COVID-19 by Borough Data Analysis Using Hadoop

---

Objectives

In this hands-on lab, you will learn how to:

- Download and upload multiple CSV files to Linux File System.
- Create and upload datasets to HDFS directories.
- Create Hive tables and perform HiveQL commands to organize and format data and perform the analysis and advanced queries.
- Download the resulting files (including merged files) to the Local File System.
- Perform Visualization using Microsoft Excel 3D Map and create graphs.

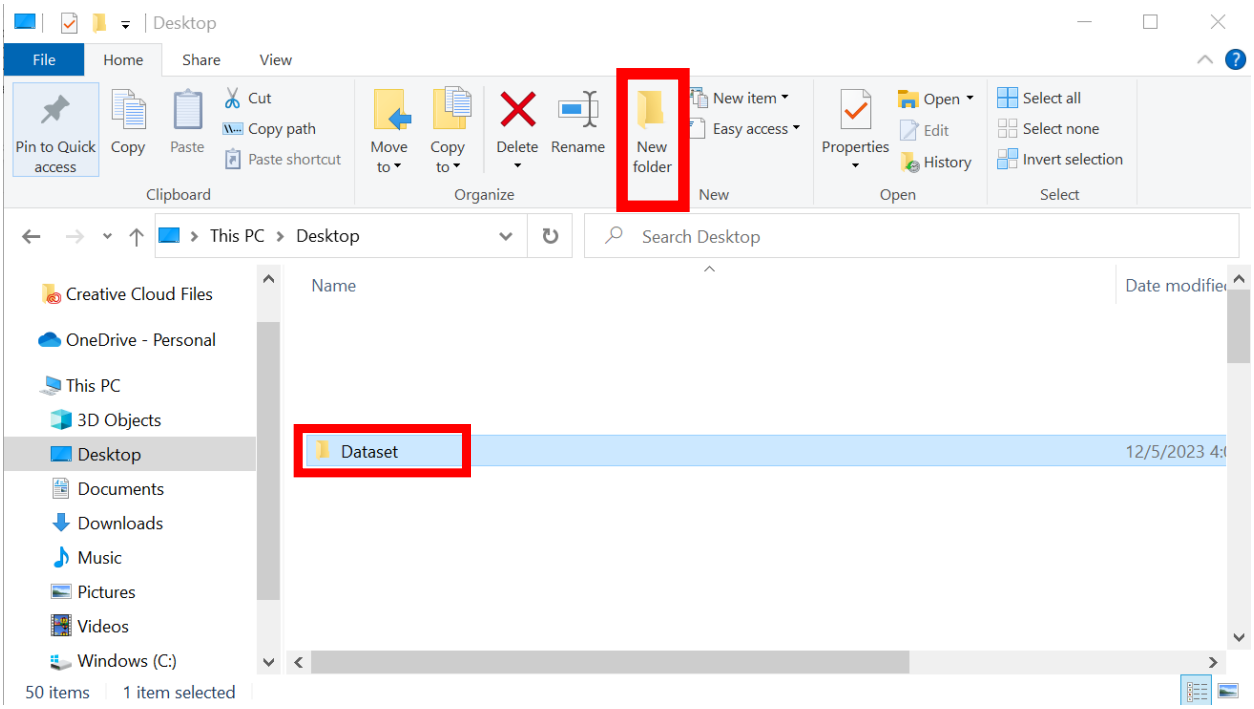
Platform Spec

- IBM Bluemix BigInsights
- CPU Speed: **1995.312 MHz**
- # of CPU cores: **8**
- # of nodes: **3**
- Total Memory Size: **58 GB**

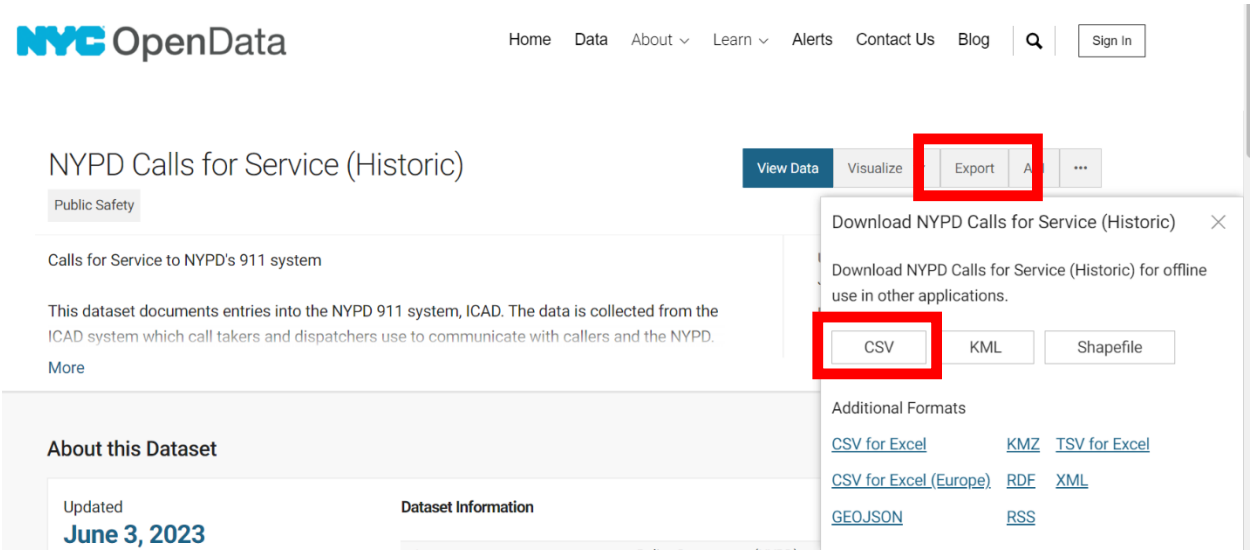
Step 1: Upload Datasets to Linux File System

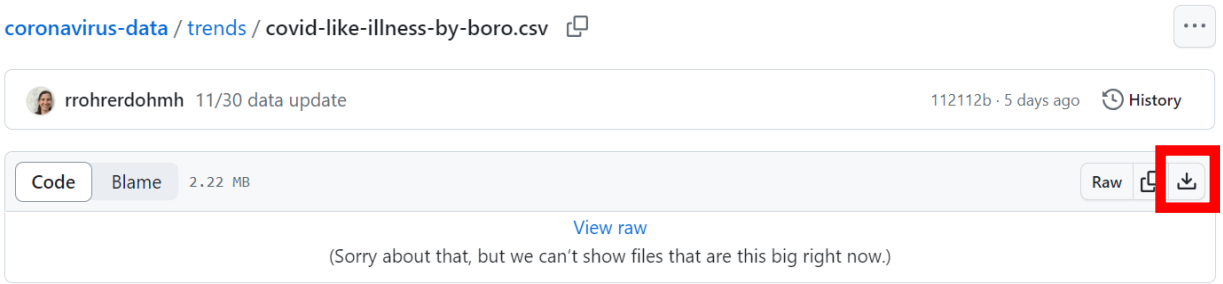
This step is to download the dataset(s) to the local device, upload the files to the Linux File System, and check if the files have been successfully uploaded.

1. Create a folder on the desktop named "Dataset":

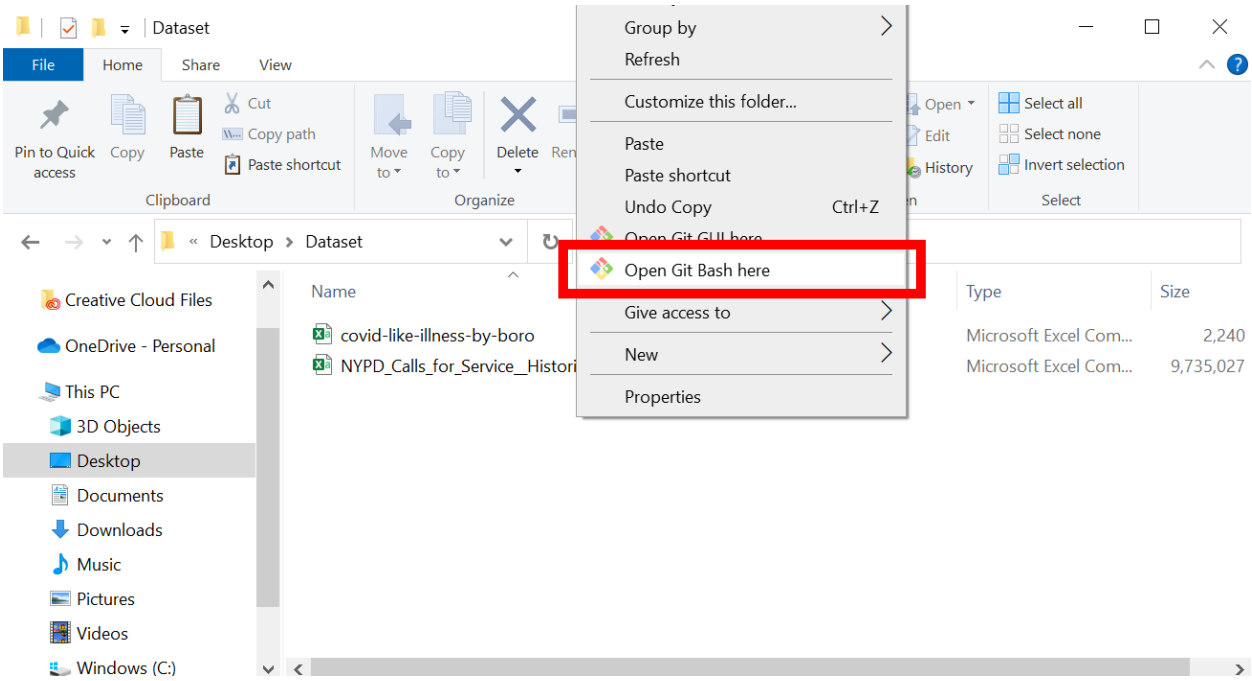


2. Download the datasets from [NYC Open Data]() and [GitHub]() using the links in the references and save them to the folder titled "Dataset" on the desktop:





3. Open the folder titled "Dataset," and right click and click "Open Git Bash here."



4. A new Git Bash terminal will open and the current directory will be noted:

```
sien.tam@LAUSD-PF3DE62E MINGW64 ~/Desktop/Dataset
$ |
```

5. Now that your directory is set to the location of the two datasets, enter the following two commands to upload the files to the Linux File System. ***This process could take a while given that one of the datasets is 9+ GB; also, make sure to replace "jtam2" with your username, and ensure that the IP address is correct.***

```
$ scp covid-like-illness-by-boro.csv jtam2@129.146.90.117:
$ scp NYPD_Calls_for_Service__Historic_.csv jtam2@129.146.90.117:
```

The result should look like the following:

```
sien.tam@LAUSD-PF3DE62E MINGW64 ~/Desktop/Dataset
$ scp covid-like-illness-by-boro.csv jtam2@129.146.90.117:
jtam2@129.146.90.117's password:
covid-like-illness-by-boro.csv
100% 2240KB   3.0MB/s   00:00

sien.tam@LAUSD-PF3DE62E MINGW64 ~/Desktop/Dataset
$ scp NYPD_Calls_for_Service__Historic_.csv jtam2@129.146.90.117:
jtam2@129.146.90.117's password:
NYPD_Calls_for_Service__Historic_.csv
100% 9507MB   8.3MB/s   19:00
```

6. Now, SSH to remotely access the Hadoop cluster. Enter your password as requested.

```
$ ssh jtam2@129.146.90.117
```

7. Input the following command to see if the datasets have been successfully uploaded.

```
$ ls -al
```

Then, you will see the following:

```
total 9737304
drwx------   6 jtam2 jtam2       4096 Dec  6 00:54 .
drwxr-xr-x. 60 root  root        4096 Nov  1 02:35 ..
-rw-rw-r--   1 jtam2 jtam2          0 Nov 14 02:44 ]
-rw-------   1 jtam2 jtam2      22271 Dec  5 23:50 .bash_history
drwxr-xr-x   2 root  root          40 Nov  1 02:35 .beeline
drwxrwxr-x   3 jtam2 jtam2         18 Nov  1 03:44 .cache
drwxrwxr-x   3 jtam2 jtam2         18 Nov  1 03:44 .config
```

```
-rw-r--r--   1 jtam2 jtam2    2293622 Dec  6 00:54 covid-like-illness-by-boro.csv
-rw-rw-r--   1 jtam2 jtam2         10 Nov 10 22:46 .hivehistory
-rw-r--r--   1 jtam2 jtam2 9968666633 Dec  6 00:53
NYPD_Calls_for_Service__Historic_.csv
drwx------   2 jtam2 jtam2         25 Nov 14 02:44 .ssh
```

Step 2: Make and Upload Datasets to HDFS Directories

This step is to create Hadoop File System (HDFS) directories for the respective datasets and put the datasets in those directories.

1. First, create a directory called "Project."

```
$ hdfs dfs -mkdir Project
```

2. Then create two new directories inside of **Project/** named **covid** and **nypd**.

```
$ hdfs dfs -mkdir Project/covid
$ hdfs dfs -mkdir Project/nypd
```

3. Check that the directories exist.

```
$ hdfs dfs -ls Project
```

Then, you will see the following:

```
Found 2 items
drwxr-xr-x   - jtam2 hdfs          0 2023-12-06 01:18 Project/covid
drwxr-xr-x   - jtam2 hdfs          0 2023-12-06 01:18 Project/nypd
```

4. Next, upload the datasets to the respective HDFS directory and check to make sure they are uploaded successfully as follows:

```
$ hdfs dfs -put covid-like-illness-by-boro.csv Project/covid/
$ hdfs dfs -put NYPD_Calls_for_Service__Historic_.csv Project/nypd/
$ hdfs dfs -ls
$ hdfs dfs -ls Project/covid
$ hdfs dfs -ls Project/nypd
```

Then, you will see the following:

```
-bash-4.2$ hdfs dfs -put covid-like-illness-by-boro.csv Project/covid/
-bash-4.2$ hdfs dfs -put NYPD_Calls_for_Service__Historic_.csv Project/nypd/
-bash-4.2$ hdfs dfs -ls
Found 3 items
drwx------   - jtam2 hdfs          0 2023-12-06 00:00 .Trash
drwxr-xr-x   - jtam2 hdfs          0 2023-11-01 03:53 .hiveJars
drwxr-xr-x   - jtam2 hdfs          0 2023-12-06 01:18 Project
-bash-4.2$ hdfs dfs -ls Project/covid
Found 1 items
-rw-r--r--   3 jtam2 hdfs    2293622 2023-12-06 01:24 Project/covid/covid-like-
illness-by-boro.csv
-bash-4.2$ hdfs dfs -ls Project/nypd
Found 1 items
-rw-r--r--   3 jtam2 hdfs 9968666633 2023-12-06 01:25
Project/nypd/NYPD_Calls_for_Service__Historic_.csv
-bash-4.2$
```

Step 3: Create Hive Tables to Query Data & Reformat Data

This step is to open Hive CLI, create a database, create Hive tables to query data, and adjust timestamp format. The following Hive statements creates external tables that allows for data to be queried. The use of external tables is to prevent data loss in case a table is deleted.

1. Open another Git Bash terminal, SSH, and then open the Hive shell environment using **beeline**.

```
$ ssh jtam2@129.146.90.117

$ beeline
```

2. Then, if you do not already have a database, in the Hive shell, create your database using the following command and your account name. ***Make sure to change "jtam2" to your username.***

```
0: jdbc:hive2://bigdaiun0.sub03291929060.trai> CREATE DATABASE IF
NOT EXISTS jtam2;
```

Then, you will see the following:

```
INFO  : Compiling command(queryId=hive_20231206051758_292ec13a-9bc0-4691-b4da-
cb16c8560572): CREATE DATABASE IF NOT EXISTS jtam2
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Returning Hive schema: Schema(fieldSchemas:null, properties:null)
INFO  : Completed compiling command(queryId=hive_20231206051758_292ec13a-9bc0-4691-
b4da-cb16c8560572); Time taken: 0.02 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hive_20231206051758_292ec13a-9bc0-4691-b4da-
cb16c8560572): CREATE DATABASE IF NOT EXISTS jtam2
INFO  : Starting task [Stage-0:DDL] in serial mode
INFO  : Completed executing command(queryId=hive_20231206051758_292ec13a-9bc0-4691-
b4da-cb16c8560572); Time taken: 0.009 seconds
INFO  : OK
INFO  : Concurrency mode is disabled, not creating a lock manager
No rows affected (0.081 seconds)
```

```
0: jdbc:hive2://bigdaiun0.sub03291929060.trai>
```

3. Then, to see if the database was successfully created, enter the following command.

```
0: jdbc:hive2://bigdaiun0.sub03291929060.trai> show DATABASES;
```

Then, you will see the following:

```
INFO  : Compiling command(queryId=hive_20231206052052_73b3aec3-2117-474c-b0df-
a0f1e0933243): show DATABASES
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Returning Hive schema: Schema(fieldSchemas:[FieldSchema(name:database_name,
type:string, comment:from deserializer)], properties:null)
INFO  : Completed compiling command(queryId=hive_20231206052052_73b3aec3-2117-474c-
b0df-a0f1e0933243); Time taken: 0.022 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hive_20231206052052_73b3aec3-2117-474c-b0df-
a0f1e0933243): show DATABASES
INFO  : Starting task [Stage-0:DDL] in serial mode
INFO  : Completed executing command(queryId=hive_20231206052052_73b3aec3-2117-474c-
b0df-a0f1e0933243); Time taken: 0.011 seconds
INFO  : OK
INFO  : Concurrency mode is disabled, not creating a lock manager

+------------------------------------+
|           database_name            |
+------------------------------------+
| jtam2                              |
+------------------------------------+
1 row selected (0.11 seconds)
0: jdbc:hive2://bigdaiun0.sub03291929060.trai>
```

4. Then, make sure to use your database using this command. ***Make sure to change "jtam2" to your username.***

```
0: jdbc:hive2://bigdaiun0.sub03291929060.trai> use jtam2;
```

You will see the following:

```
INFO  : Compiling command(queryId=hive_20231206052406_f2ccd335-04ef-4834-a439-
96cc62e022ed): use jtam2
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Returning Hive schema: Schema(fieldSchemas:null, properties:null)
INFO  : Completed compiling command(queryId=hive_20231206052406_f2ccd335-04ef-4834-
a439-96cc62e022ed); Time taken: 0.028 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hive_20231206052406_f2ccd335-04ef-4834-a439-
96cc62e022ed): use jtam2
INFO  : Starting task [Stage-0:DDL] in serial mode
INFO  : Completed executing command(queryId=hive_20231206052406_f2ccd335-04ef-4834-
a439-96cc62e022ed); Time taken: 0.205 seconds
INFO  : OK
INFO  : Concurrency mode is disabled, not creating a lock manager
No rows affected (0.245 seconds)
0: jdbc:hive2://bigdaiun0.sub03291929060.trai>
```

5. In the Hive shell CLI, paste the following HiveQL code to create an external table called "all_borough_covid," which will contain all the data from the "Covid-like-Illness-by-Boro" Dataset. We start by applying a "DROP" command to remove any existing table that may come into conflict.

```
DROP TABLE IF EXISTS all_borough_covid;


CREATE EXTERNAL TABLE IF NOT EXISTS all_borough_covid(days DATE,
borough STRING, Admit_0_4 DOUBLE, Admit_5_12 DOUBLE, Admit_13_17
DOUBLE, Admit_18_24 DOUBLE, Admit_25_34 DOUBLE, Admit_35_44
DOUBLE,Admit_45_54 DOUBLE, Admit_55_64 DOUBLE, Admit_65_74 DOUBLE,
Admit_75 DOUBLE, Admit_All_ages DOUBLE, Visit_0_4 DOUBLE,
Visit_5_12 DOUBLE, Visit_13_17 DOUBLE, Visit_18_24 DOUBLE,
Visit_25_34 DOUBLE, Visit_35_44 DOUBLE, Visit_45_54 DOUBLE,
Visit_55_64 DOUBLE, Visit_65_74 DOUBLE, Visit_75 DOUBLE,
Visit_All_ages DOUBLE) ROW FORMAT DELIMITED FIELDS TERMINATED BY
',' STORED AS TEXTFILE LOCATION '/user/jtam2/Project/covid'
TBLPROPERTIES ('skip.header.line.count'='1');
```

Then, in the Hive shell, make sure that the table "all_borough_covid" was created and is shown:

```
show tables;
```

You will see the following:

```
+--------------------+
|      tab_name      |
+--------------------+
| all_borough_covid  |
+--------------------+
1 row selected (0.252 seconds)
```

To view the structure of the all_borough_covid table, or any table, input the **describe** command:

```
describe all_borough_covid;
```

You will see the following:

```
+-----------------+------------+----------+
|    col_name     | data_type  | comment  |
+-----------------+------------+----------+
| days            | date       |          |
| borough         | string     |          |
| admit_0_4       | double     |          |
| admit_5_12      | double     |          |
| admit_13_17     | double     |          |
| admit_18_24     | double     |          |
| admit_25_34     | double     |          |
| admit_35_44     | double     |          |
| admit_45_54     | double     |          |
| admit_55_64     | double     |          |
| admit_65_74     | double     |          |
| admit_75        | double     |          |
| admit_all_ages  | double     |          |
| visit_0_4       | double     |          |
| visit_5_12      | double     |          |
```

```
| visit_13_17     | double     |          |
| visit_18_24     | double     |          |
| visit_25_34     | double     |          |
| visit_35_44     | double     |          |
| visit_45_54     | double     |          |
| visit_55_64     | double     |          |
| visit_65_74     | double     |          |
| visit_75        | double     |          |
| visit_all_ages  | double     |          |
+-----------------+------------+----------+
24 rows selected (0.159 seconds)
```

Now, to query the content of the all_borough_covid table, enter the following command.

```
select * from all_borough_covid LIMIT 3;
```

You will see a result like the following:

```
+-------------------------+----------------------------+-------------------------------
+-------------------------------+-------------------------------+-------------------------
------------+-----------------------------+-----------------------------+-----------------------+-------
-------------------------+-------------------------------+---------------------------
-----+-----------------------------+-----------------------------------+--------------
----------------+-------------------------------+-------------------------------+----
----------------------------+-------------------------------+-----------------------
--------+-------------------------------+-------------------------------+-----------
---------------------+-----------------------------+-----------------------------------
-+
| all_borough_covid.days  | all_borough_covid.borough  | all_borough_covid.admit_0_4
| all_borough_covid.admit_5_12  | all_borough_covid.admit_13_17  |
all_borough_covid.admit_18_24  | all_borough_covid.admit_25_34  |
all_borough_covid.admit_35_44  | all_borough_covid.admit_45_54  |
all_borough_covid.admit_55_64  | all_borough_covid.admit_65_74  |
all_borough_covid.admit_75  | all_borough_covid.admit_all_ages  |
all_borough_covid.visit_0_4  | all_borough_covid.visit_5_12  |
all_borough_covid.visit_13_17  | all_borough_covid.visit_18_24  |
all_borough_covid.visit_25_34  | all_borough_covid.visit_35_44  |
all_borough_covid.visit_45_54  | all_borough_covid.visit_55_64  |
all_borough_covid.visit_65_74  | all_borough_covid.visit_75  |
all_borough_covid.visit_all_ages  |
+-------------------------+----------------------------+-------------------------------
+-------------------------------+-------------------------------+-------------------------
------------+-----------------------------+-----------------------------+-----------------------+-------
-------------------------+-------------------------------+---------------------------
-----+-----------------------------+-----------------------------------+--------------
----------------+-------------------------------+-------------------------------+----
----------------------------+-------------------------------+-----------------------
--------+-------------------------------+-------------------------------+-----------
---------------------+-----------------------------+-----------------------------------
-+
| 2020-02-01              | "Bronx"                    | 2.92857212585051
| 1.91360875603431              | 0.0                           | 3.46477054753579
| 2.17409264243568              | 1.65399522546712              | 2.78809372455864
| 4.35085276714236              | 6.86180328190248              | 6.14545052297784
| 3.0025165278061                 | 64.4285867687111            | 33.8070880232729
| 9.60979516501414              | 18.0168068471861              | 12.6097373261269
| 10.4753030946251              | 10.0371374084111              | 9.32325592959077
| 6.86180328190248              | 8.60363073216898            | 17.3866654749702
|
| 2020-02-01              | "Brooklyn"                 | 4.27259132663961
| 0.394225489497258             | 0.0                           | 0.465384983966236
| 0.210607455082695             | 0.833738622941708             | 2.63853112972008
| 2.43270106274284              | 2.93557872488245              | 4.5202993729699
| 1.62612328337521                | 30.4422132023072            | 27.2015587753108
| 4.2188395503451               | 11.6346245991559              | 8.21369074822511
| 5.00243173765025              | 7.58577699794524              | 6.95057446497953
| 5.38189432895117              | 9.0405987459398             | 10.9182563312336
|
```

```
| 2020-02-01              | "Manhattan"                | 1.29973095569217
| 1.00488201383002              | 0.0                              | 0.0
| 0.550163948856759              | 1.7142000042855                | 0.501499483455532
| 1.6264746703678                | 3.41667748614537               | 6.52736188510211
| 1.53496559528115                 | 24.6948881581513             | 23.1122863180904
| 1.78344998933169               | 8.18823490688703               | 5.22655751413921
| 7.71390001928475               | 2.50749741727766               | 7.59021512838307
| 8.20002596674889               | 11.4228832989287            | 8.41161146214069
|
+------------------------+---------------------------+-----------------------------
+------------------------------+------------------------------+--------------------
------------+------------------------------+------------------------------+-------
-------------------------+------------------------------+-------------------------
-----+-----------------------------+-----------------------------+---------------
----------------+-----------------------------+-----------------------------+----
----------------------------+-----------------------------+-----------------------
--------+-----------------------------+-----------------------------+-----------
---------------------+----------------------------+-----------------------------
-+
3 rows selected (0.402 seconds)
```

If you do not see the result above, you need to check if the previous steps worked.

6. In the Hive shell CLI, paste the following HiveQL code to create an external table called "raw_calls," which will contain all the data from the "NYPD-Calls-for-Service-Historic" Dataset. We start by applying a "DROP" command to remove any existing table that may come into conflict.

```
DROP TABLE IF EXISTS raw_calls;

CREATE EXTERNAL TABLE IF NOT EXISTS raw_calls(objectid INT,
cad_event_id INT, create_date STRING, incident_date STRING,
incident_time STRING, nypd_pct_cd INT, boro_nm STRING,
patrl_boro_nm STRING, geo_cd_x INT, geo_cd_y INT, radio_code
STRING, typ_desc STRING, cip_jobs STRING, add_ts STRING, disp_ts
STRING, arrivd_ts STRING, closng_ts STRING, latitude DOUBLE,
longitude DOUBLE, point STRING) ROW FORMAT DELIMITED FIELDS
TERMINATED BY ',' STORED AS TEXTFILE LOCATION
'/user/jtam2/Project/nypd' TBLPROPERTIES
('skip.header.line.count'='1');
```

Now, to query the content of the raw_calls table, enter the following command.

```
select * from raw_calls LIMIT 3;
```

You will see a result like the following:

```
+---------------------+-------------------------+-------------------------+------------
--------------+-------------------------+-----------------------+------------------
-+-------------------------+---------------------+---------------------+-------------
----------+-----------------------------------------------+---------------------+-
-------------------------+-----------------------+-----------------------+---------
----------------+---------------------+---------------------+-----------------------
------------------------+
| raw_calls.objectid  | raw_calls.cad_event_id  | raw_calls.create_date  |
raw_calls.incident_date  | raw_calls.incident_time  | raw_calls.nypd_pct_cd  |
raw_calls.boro_nm  | raw_calls.patrl_boro_nm  | raw_calls.geo_cd_x  |
raw_calls.geo_cd_y  | raw_calls.radio_code  |               raw_calls.typ_desc
| raw_calls.cip_jobs  |    raw_calls.add_ts     |    raw_calls.disp_ts    |
```

```
raw_calls.arrivd_ts   |   raw_calls.closng_ts   | raw_calls.latitude  |
raw_calls.longitude  |                raw_calls.point                 |
+---------------------+-------------------------+-----------------------+------------
--------------+-------------------------+-----------------------+-------------------
-+-------------------------+---------------------+---------------------+-------------
----------+-------------------------------------------------+---------------------+-
------------------------+-------------------------+-------------------------+---------
----------------+---------------------+---------------------+-----------------------
------------------------+
| NULL                | 45229009                | 02/19/2018            | 02/19/2018
| 14:41:35                | 49                    | BRONX                 | PATROL BORO
BRONX        | 1021869             | 254631              | 10V2                  |
INVESTIGATE/POSSIBLE CRIME: SUSP VEHICLE/OUTSIDE  | Non CIP             | 02/19/2018
02:41:35 PM  | 02/19/2018 02:41:35 PM  | 02/19/2018 02:41:36 PM  | 02/19/2018 02:50:46
PM  | 40.86550598700006   | -73.86399260399998   | POINT (-73.86399260399998
40.86550598700006)   |
| NULL                | 45083401                | 02/13/2018            | 02/13/2018
| 21:57:20                | 81                    | BROOKLYN              | PATROL BORO
BKLYN NORTH  | 1004444             | 190136              | 75D                   |
VISIBILITY PATROL: DIRECTED                       | Non CIP             | 02/13/2018
09:57:20 PM  | 02/13/2018 09:57:21 PM  | 02/13/2018 09:57:27 PM  | 02/13/2018 10:26:55
PM  | 40.68854134700007   | -73.92718524599996   | POINT (-73.92718524599996
40.68854134700007)   |
| NULL                | 46117805                | 03/24/2018            | 03/24/2018
| 19:10:39                | 70                    | BROOKLYN              | PATROL BORO
BKLYN SOUTH  | 994702              | 170702              | 54C1                  |
AMBULANCE CASE: CARDIAC/INSIDE                    | Non CIP             | 03/24/2018
07:10:39 PM  | 03/24/2018 07:12:04 PM  | 03/24/2018 07:17:17 PM  | 03/24/2018 07:29:13
PM  | 40.635216170000035  | -73.96234338999994   | POINT (-73.96234338999994
40.635216170000035)  |
+---------------------+-------------------------+-----------------------+------------
--------------+-------------------------+-----------------------+-------------------
-+-------------------------+---------------------+---------------------+-------------
----------+-------------------------------------------------+---------------------+-
------------------------+-------------------------+-------------------------+---------
----------------+---------------------+---------------------+-----------------------
------------------------+
3 rows selected (0.43 seconds)
```

7. In the Hive shell CLI, paste the following HiveQL code to create a table called “borough_data,” which will contain only a portion of the data from the “all_borough_covid” table. We start by applying a “DROP” command to remove any existing table that may come into conflict.

```
DROP TABLE IF EXISTS borough_data;

CREATE TABLE borough_data ROW FORMAT DELIMITED FIELDS TERMINATED BY
',' STORED AS TEXTFILE LOCATION '/user/jtam2/Project/boro_data' AS
SELECT days, borough, Admit_0_4, Admit_5_12, Admit_13_17,
Admit_18_24, Admit_25_34, Admit_35_44, Admit_45_54, Admit_55_64,
Admit_65_74, Admit_75 FROM all_borough_covid;
```

Now, to query the content of the borough_data table, enter the following command.

```
select * from borough_data LIMIT 3;
```

You will see a result like the following:

```
+--------------------+----------------------+-------------------------+-------------
------------+--------------------------+--------------------------+----------------
```

```
----------+-------------------------+-------------------------+------------------
--------+-------------------------+-----------------------+
| borough_data.days  | borough_data.borough  | borough_data.admit_0_4  |
borough_data.admit_5_12  | borough_data.admit_13_17  | borough_data.admit_18_24  |
borough_data.admit_25_34  | borough_data.admit_35_44  | borough_data.admit_45_54  |
borough_data.admit_55_64  | borough_data.admit_65_74  | borough_data.admit_75  |
+--------------------+-----------------------+-------------------------+--------------
------------+-------------------------+-------------------------+-----------------
----------+-------------------------+-------------------------+------------------
--------+-------------------------+-----------------------+
| 2020-02-01         | "Bronx"               | 2.92857212585051        |
1.91360875603431         | 0.0                     | 3.46477054753579          |
2.17409264243568          | 1.65399522546712          | 2.78809372455864          |
4.35085276714236          | 6.86180328190248          | 6.14545052297784       |
| 2020-02-01         | "Brooklyn"            | 4.27259132663961        |
0.394225489497258        | 0.0                     | 0.465384983966236         |
0.210607455082695         | 0.833738622941708         | 2.63853112972008          |
2.43270106274284          | 2.93557872488245          | 4.5202993729699        |
| 2020-02-01         | "Manhattan"           | 1.29973095569217        |
1.00488201383002         | 0.0                     | 0.0                       |
0.550163948856759         | 1.7142000042855           | 0.501499483455532         |
1.6264746703678           | 3.41667748614537          | 6.52736188510211       |
+--------------------+-----------------------+-------------------------+--------------
------------+-------------------------+-------------------------+-----------------
----------+-------------------------+-------------------------+------------------
--------+-------------------------+-----------------------+
3 rows selected (0.517 seconds)
```

8. In the Hive shell CLI, paste the following HiveQL code to create an empty table called "nypd_calls," which will contain only a portion of the data from the "raw_calls" table once the data has been formatted. We start by applying a "DROP" command to remove any existing table that may come into conflict.

```
DROP TABLE IF EXISTS nypd_calls;

CREATE TABLE IF NOT EXISTS nypd_calls(create_date DATE, boro_nm
STRING, radio_code STRING, typ_desc STRING, cip_jobs STRING, add_ts
TIMESTAMP, latitude DOUBLE, longitude DOUBLE) ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION
'/user/jtam2/Project/export';
```

9. In the Hive shell CLI, paste the following HiveQL code that will reformat the data type of **create_date** from the raw_calls table to DATE as 'MM/dd/yyyy', and reformat the data type of **add_ts** from the raw_calls table to TIMESTAMP as 'MM/dd/yyyy hh:mm:ss aa'. This is simply to convert regular time to unix time and then unix time to unix timestamp. After reformatting the data types, it will insert the updated version of **create_date** and **add_ts** along with boro_m, radio_code, typ_desc, cip_jobs, latitude, and longitude to the nypd_calls table. ***This could take a***

few minutes due to the size of the dataset. (The progress bar may not move, but it is processing; approx: 10 mins).

```
FROM raw_calls INSERT OVERWRITE TABLE nypd_calls SELECT
cast(from_unixtime(unix_timestamp(create_date,'MM/dd/yyyy')) AS
DATE), boro_nm, radio_code, typ_desc, cip_jobs,
cast(from_unixtime(unix_timestamp(add_ts,'MM/dd/yyyy hh:mm:ss aa'))
AS TIMESTAMP), latitude, longitude;
```

Now, to query the content of the nypd_calls table, enter the following command.

```
select * from nypd_calls LIMIT 3;
```

You will see a result like the following:

```
+-------------------------+---------------------+-------------------------+------------
---------------------------------------+----------------------+-----------------------
-+----------------------+-----------------------+
| nypd_calls.create_date  | nypd_calls.boro_nm  | nypd_calls.radio_code  |
nypd_calls.typ_desc                | nypd_calls.cip_jobs  |   nypd_calls.add_ts    |
nypd_calls.latitude  | nypd_calls.longitude  |
+-------------------------+---------------------+-------------------------+------------
---------------------------------------+----------------------+-----------------------
-+----------------------+-----------------------+
| 2018-02-19              | BRONX               | 10V2                    |
INVESTIGATE/POSSIBLE CRIME: SUSP VEHICLE/OUTSIDE  | Non CIP              | 2018-02-19
14:41:35.0  | 40.86550598700006    | -73.86399260399998    |
| 2018-02-13              | BROOKLYN            | 75D                     | VISIBILITY
PATROL: DIRECTED                      | Non CIP              | 2018-02-13 21:57:20.0
| 40.68854134700007    | -73.92718524599996    |
| 2018-03-24              | BROOKLYN            | 54C1                    | AMBULANCE
CASE: CARDIAC/INSIDE                  | Non CIP              | 2018-03-24 19:10:39.0
| 40.635216170000035   | -73.96234338999994    |
+-------------------------+---------------------+-------------------------+------------
---------------------------------------+----------------------+-----------------------
-+----------------------+-----------------------+
3 rows selected (0.413 seconds)
```

10. Now, insert the following command to show tables to ensure you have all tables created so far:

    all_borough_covid, borough_data, nypd_calls, and raw_calls.

```
show tables;
```

You will see a result like the following:

```
+--------------------+
|      tab_name      |
+--------------------+
| all_borough_covid  |
| borough_data       |
| nypd_calls         |
| raw_calls          |
+--------------------+
4 rows selected (0.27 seconds)
```

11. In the Hive shell CLI, paste the following HiveQL code that will create a table called "reduced_nypd_calls" that will contain the results of a query for all data that has a create_date greater than or equal to **2020-03-01**, and a create_date less than or equal to **2021-03-31**, as well as cip_jobs that are "Non CIP" and typ_desc that contains the text "AMBULANCE". We start by applying a "DROP" command to remove any existing table that may come into conflict.

```
DROP TABLE IF EXISTS reduced_nypd_calls;

CREATE TABLE reduced_nypd_calls ROW FORMAT DELIMITED FIELDS
TERMINATED BY ',' STORED AS TEXTFILE LOCATION
'/user/jtam2/Project/export_v2' AS SELECT * FROM nypd_calls WHERE
create_date >= '2020-03-01' AND create_date <= '2021-03-31' AND
cip_jobs == 'Non CIP' AND typ_desc LIKE '%AMBULANCE%';
```

Now, to query the content of the reduced_nypd_calls table, enter the following command.

```
select * from reduced_nypd_calls limit 3;
```

You will see a result like the following:

```
+--------------------------------+----------------------------+---------------------
-----------+-----------------------------------+----------------------------+----
-----------------------+----------------------------+-----------------------------
-+
| reduced_nypd_calls.create_date  | reduced_nypd_calls.boro_nm  |
reduced_nypd_calls.radio_code  |     reduced_nypd_calls.typ_desc      |
reduced_nypd_calls.cip_jobs  | reduced_nypd_calls.add_ts  |
reduced_nypd_calls.latitude  | reduced_nypd_calls.longitude  |
+--------------------------------+----------------------------+---------------------
-----------+-----------------------------------+----------------------------+----
-----------------------+----------------------------+-----------------------------
-+
| 2021-01-01                     | QUEENS                     | 54S9
| AMBULANCE CASE: SERIOUS/TRANSIT      | Non CIP                    | 2021-01-01
02:39:41.0      | 40.604079                  | -73.754726                  |
| 2021-01-01                     | QUEENS                     | 54E1
| AMBULANCE CASE: EDP/INSIDE           | Non CIP                    | 2021-01-01
03:48:41.0      | 40.709903                  | -73.758189                  |
| 2021-01-01                     | BRONX                      | 54U2
| AMBULANCE CASE: UNCONSCIOUS/OUTSIDE  | Non CIP                    | 2021-01-01
03:42:38.0      | 40.826282                  | -73.920708                  |
+--------------------------------+----------------------------+---------------------
-----------+-----------------------------------+----------------------------+----
-----------------------+----------------------------+-----------------------------
-+
3 rows selected (0.542 seconds)
```

Step 4: Download the Data into your PC

This step is to download the data into your PC after Hive tables have been created.

1. Return to the first Git Bash terminal and insert the following commands to display the contents of the directory **/user/jtam2/Project/boro_data/** and **/user/jtam2/Project/export_v2/**

```
$ hdfs dfs -ls /user/jtam2/Project/boro_data/

$ hdfs dfs -ls /user/jtam2/Project/export_v2/
```

You will see a result like the following:

```
-bash-4.2$ hdfs dfs -ls /user/jtam2/Project/boro_data/
Found 1 items
-rw-r--r--   3 jtam2 hdfs     993569 2023-12-07 07:55
/user/jtam2/Project/boro_data/000000_0
-bash-4.2$ hdfs dfs -ls /user/jtam2/Project/export_v2/
Found 17 items
-rw-r--r--   3 jtam2 hdfs        104 2023-12-07 09:18
/user/jtam2/Project/export_v2/000006_0
-rw-r--r--   3 jtam2 hdfs     439452 2023-12-07 09:18
/user/jtam2/Project/export_v2/000022_0
-rw-r--r--   3 jtam2 hdfs    6441872 2023-12-07 09:18
/user/jtam2/Project/export_v2/000023_0
-rw-r--r--   3 jtam2 hdfs    6030035 2023-12-07 09:18
/user/jtam2/Project/export_v2/000024_0
-rw-r--r--   3 jtam2 hdfs    2821780 2023-12-07 09:19
/user/jtam2/Project/export_v2/000025_0
-rw-r--r--   3 jtam2 hdfs    1958312 2023-12-07 09:19
/user/jtam2/Project/export_v2/000047_0
-rw-r--r--   3 jtam2 hdfs    3167775 2023-12-07 09:18
/user/jtam2/Project/export_v2/000048_0
-rw-r--r--   3 jtam2 hdfs    7079181 2023-12-07 09:19
/user/jtam2/Project/export_v2/000049_0
-rw-r--r--   3 jtam2 hdfs    6629853 2023-12-07 09:18
/user/jtam2/Project/export_v2/000050_0
-rw-r--r--   3 jtam2 hdfs    3355211 2023-12-07 09:19
/user/jtam2/Project/export_v2/000051_0
-rw-r--r--   3 jtam2 hdfs    4231333 2023-12-07 09:18
/user/jtam2/Project/export_v2/000052_0
-rw-r--r--   3 jtam2 hdfs    8057234 2023-12-07 09:19
/user/jtam2/Project/export_v2/000053_0
-rw-r--r--   3 jtam2 hdfs    8748534 2023-12-07 09:18
/user/jtam2/Project/export_v2/000054_0
-rw-r--r--   3 jtam2 hdfs    5983965 2023-12-07 09:19
/user/jtam2/Project/export_v2/000055_0
-rw-r--r--   3 jtam2 hdfs    6377127 2023-12-07 09:18
/user/jtam2/Project/export_v2/000056_0
-rw-r--r--   3 jtam2 hdfs    7259314 2023-12-07 09:19
/user/jtam2/Project/export_v2/000057_0
-rw-r--r--   3 jtam2 hdfs      52856 2023-12-07 09:18
/user/jtam2/Project/export_v2/000058_0
-bash-4.2$
```

2. Then, download the output files to the local file systems. Rename the file 000000_0 from the directory boro_data to **borough_data.txt**.

Note: The **export_v2** directory code is formatted differently because the resulting data files were saved into several smaller files. In this final command, we combine them all into one and rename it **combine.txt.**

```
$ hdfs dfs -get /user/jtam2/Project/boro_data/000000_0
borough_data.txt

$ hdfs dfs -getmerge /user/jtam2/Project/export_v2/0000*
combine.txt
```

3. Open a third Git Bash terminal and insert the following code to download the data files to your local computer. **You do not need to SSH. Insert your password when prompted. Also, the download could take a while depending on the data size.**

```
$ scp jtam2@129.146.90.117:/home/jtam2/combine.txt .

$ scp jtam2@129.146.90.117:/home/jtam2/borough_data.txt .
```

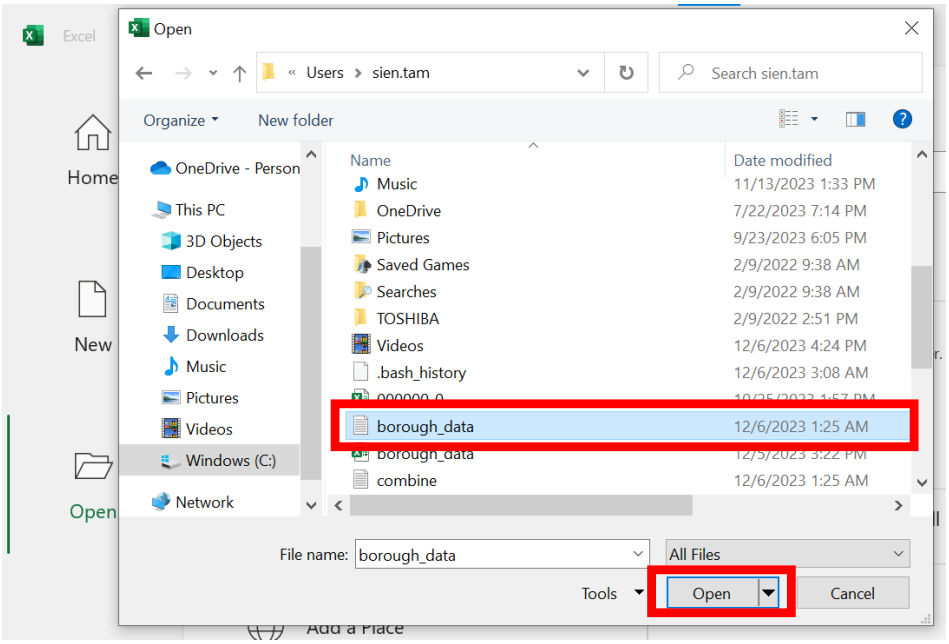
You will see a result like the following:

```
$ scp jtam2@129.146.90.117:/home/jtam2/combine.txt .
jtam2@129.146.90.117's password:
combine.txt
100%   75MB  36.3MB/s   00:02

$ scp jtam2@129.146.90.117:/home/jtam2/borough_data.txt .
jtam2@129.146.90.117's password:
borough_data.txt
100%  970KB   5.1MB/s   00:00
```

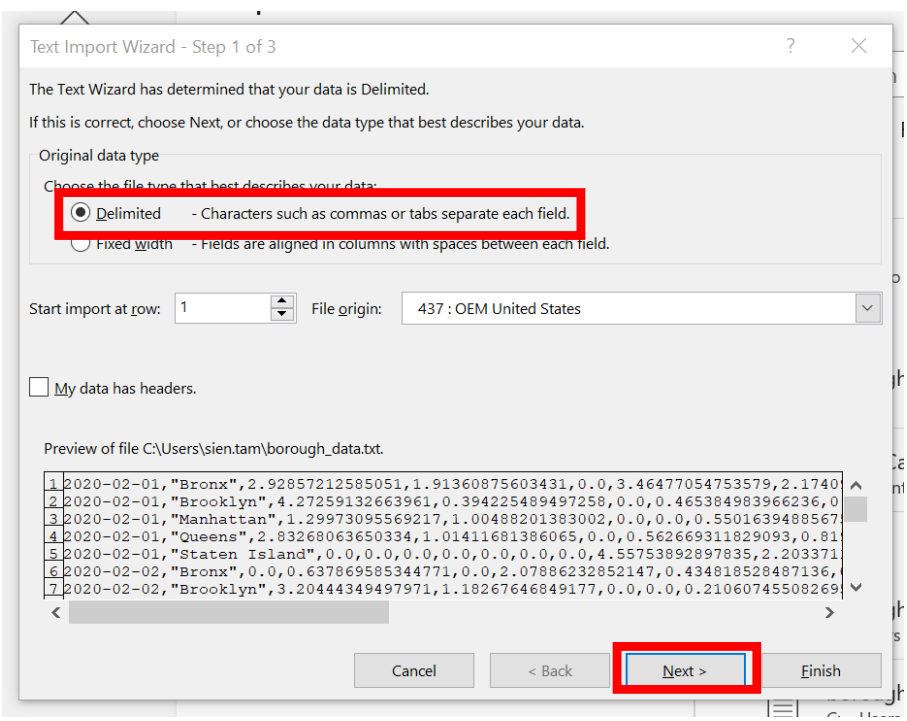
Step 5: Visualization in Excel 3D Map with Borough_Data.txt

This step is to use Excel's 3D Map feature to conduct a tempo-spatial analysis of the Borough_Data.txt file.

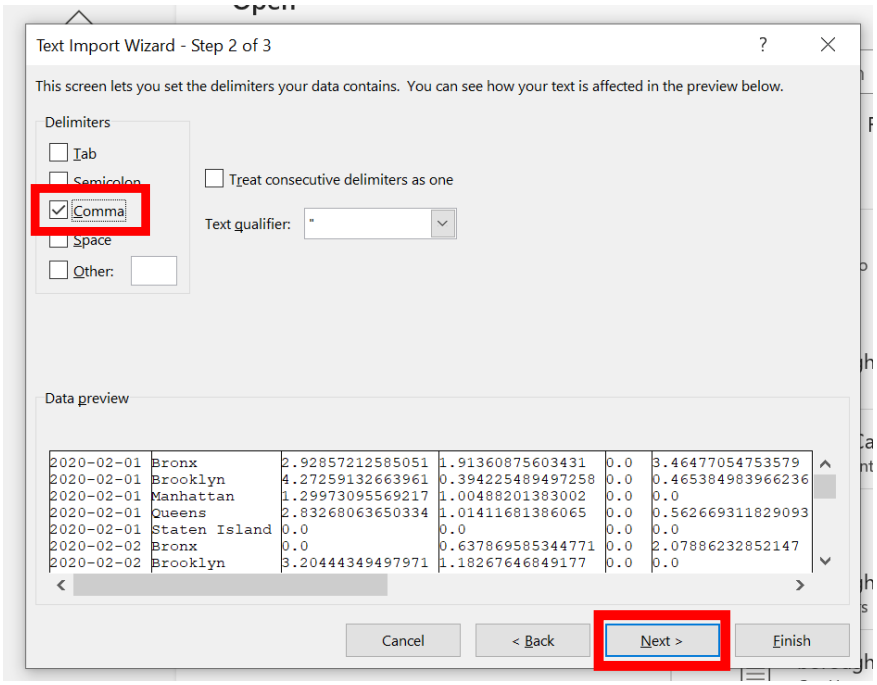
1. Open the file, borough_data.txt, in Excel. You have to open Excel first.



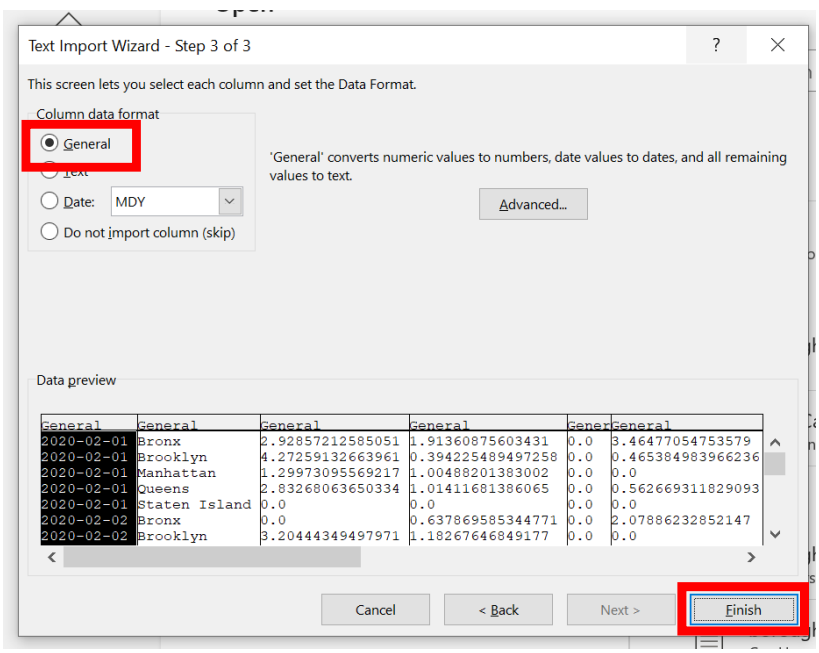
2. Then, in the first page of the Text Import Wizard, make sure “Delimited” is selected. Then press “Next.”



3. Then, in the second page of the Text Import Wizard, make sure “Comma” is selected instead of “tab.” Then press “Next.”

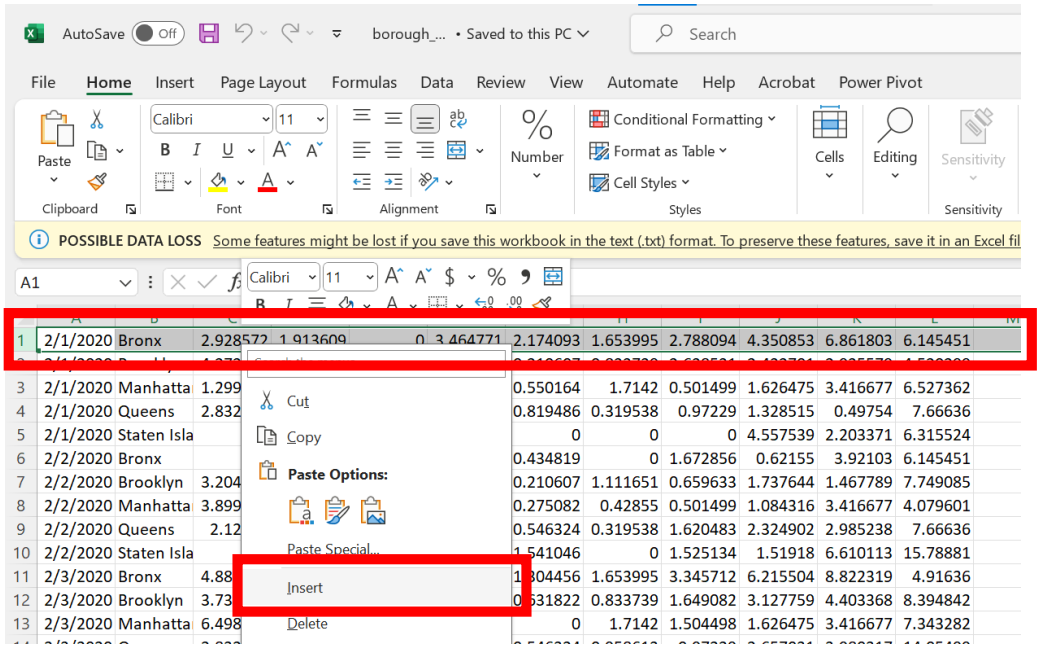


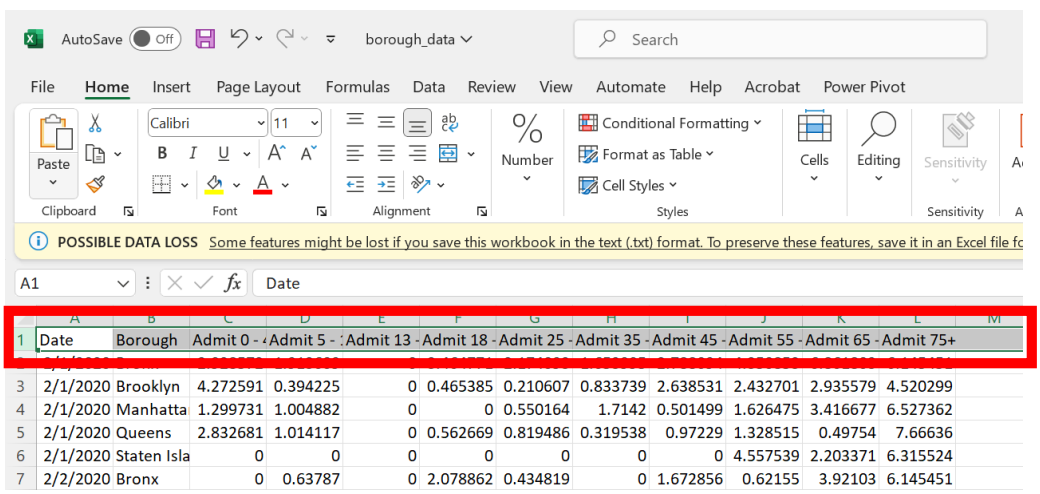
4. Then, in the third page of the Text Import Wizard, make sure "General" is selected for all columns. Then press "Finish."



5. Upon reaching the Excel spreadsheet view, insert a new row above the current row 1. In the first row, insert the header of each column in this order: **Date, Borough, Admit 0 – 4, Admit 5 – 12,**

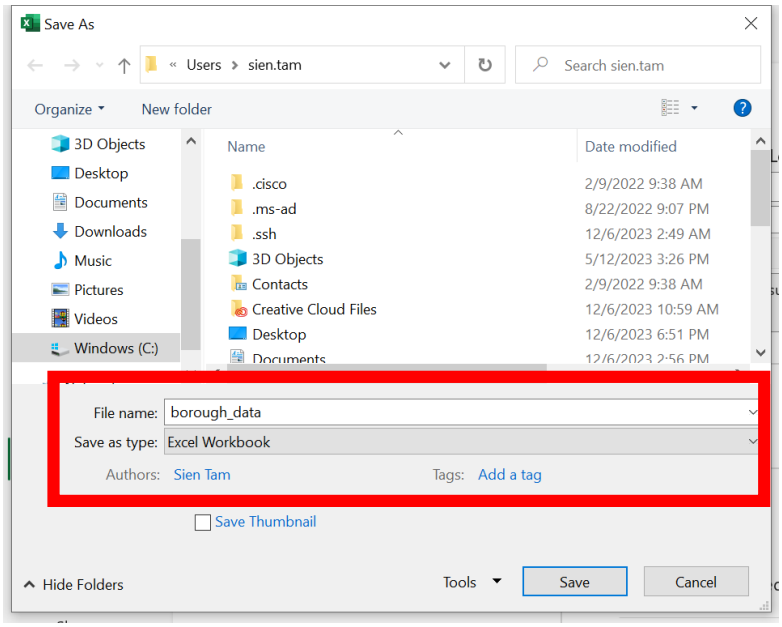
Admit 13 – 17, Admit 18 – 24, Admit 25 – 34, Admit 35 – 44, Admit 45 – 54, Admit 55 – 64, Admit 65 – 74, and Admit 75+.



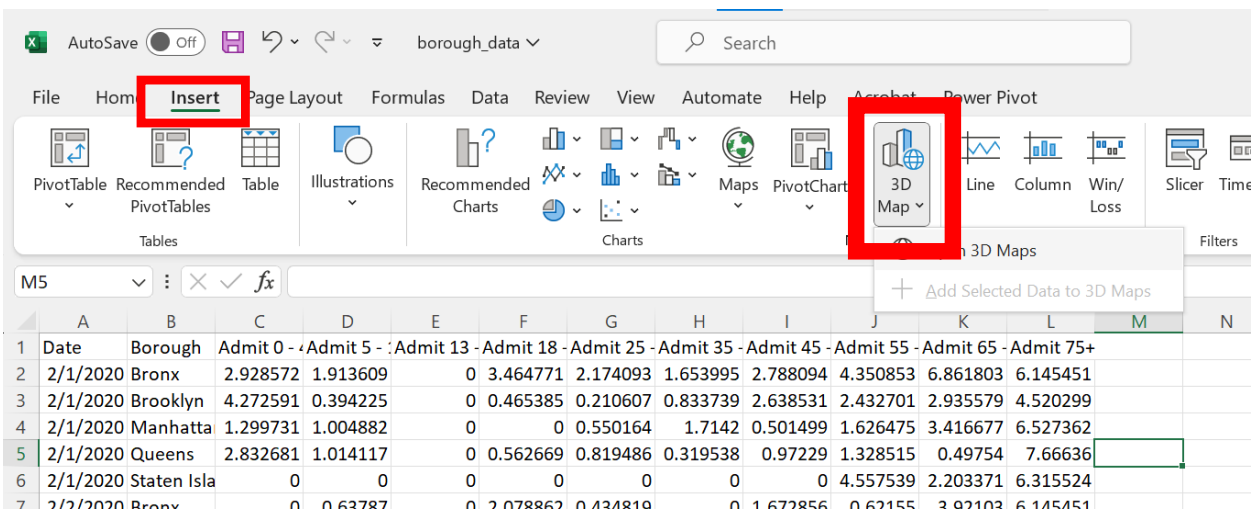


	A	B	C	D	E	F	G	H	I	J	K	L
1	Date	Borough	Admit 0 -	Admit 5 -	Admit 13 -	Admit 18 -	Admit 25 -	Admit 35 -	Admit 45 -	Admit 55 -	Admit 65 -	Admit 75+
3	2/1/2020	Brooklyn	4.272591	0.394225	0	0.465385	0.210607	0.833739	2.638531	2.432701	2.935579	4.520299
4	2/1/2020	Manhatta	1.299731	1.004882	0	0	0.550164	1.7142	0.501499	1.626475	3.416677	6.527362
5	2/1/2020	Queens	2.832681	1.014117	0	0.562669	0.819486	0.319538	0.97229	1.328515	0.49754	7.66636
6	2/1/2020	Staten Isla	0	0	0	0	0	0	0	4.557539	2.203371	6.315524
7	2/2/2020	Bronx	0	0.63787	0	2.078862	0.434819	0	1.672856	0.62155	3.92103	6.145451

6. To visualize the results on Excel 3D Map, convert csv file to “Excel Workbook” (.xlsx) and keep the name “borough_data.”

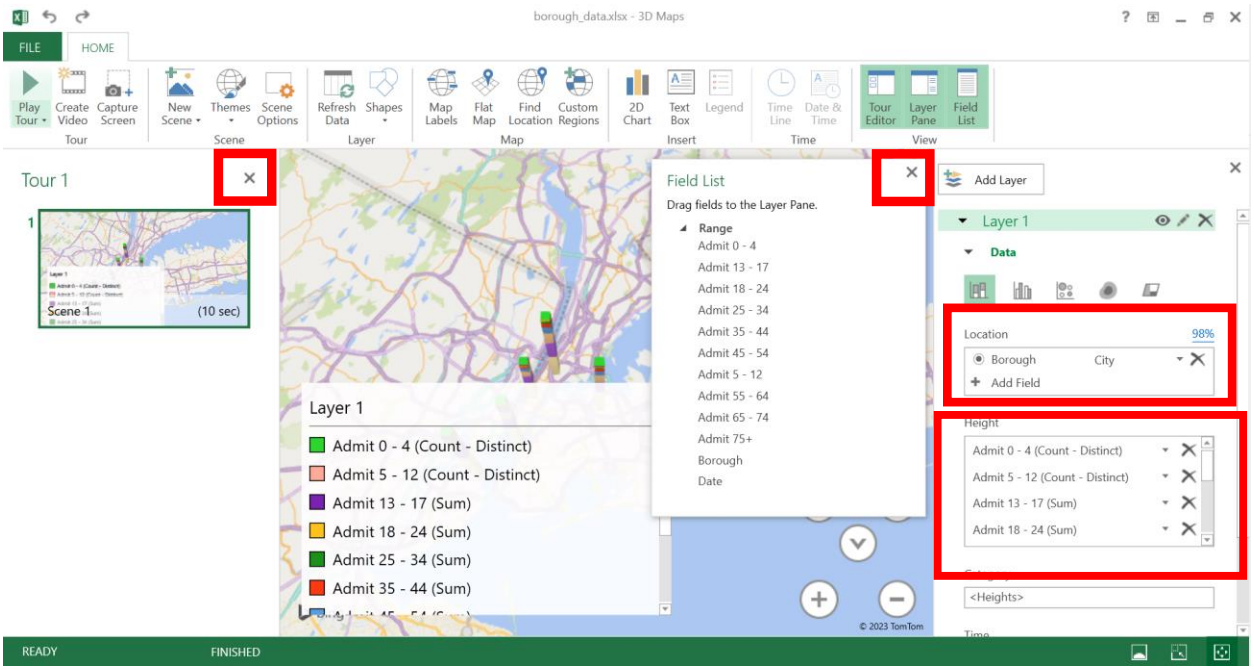


7. Then, go to the "Insert" tab at the top and click "3D Map."

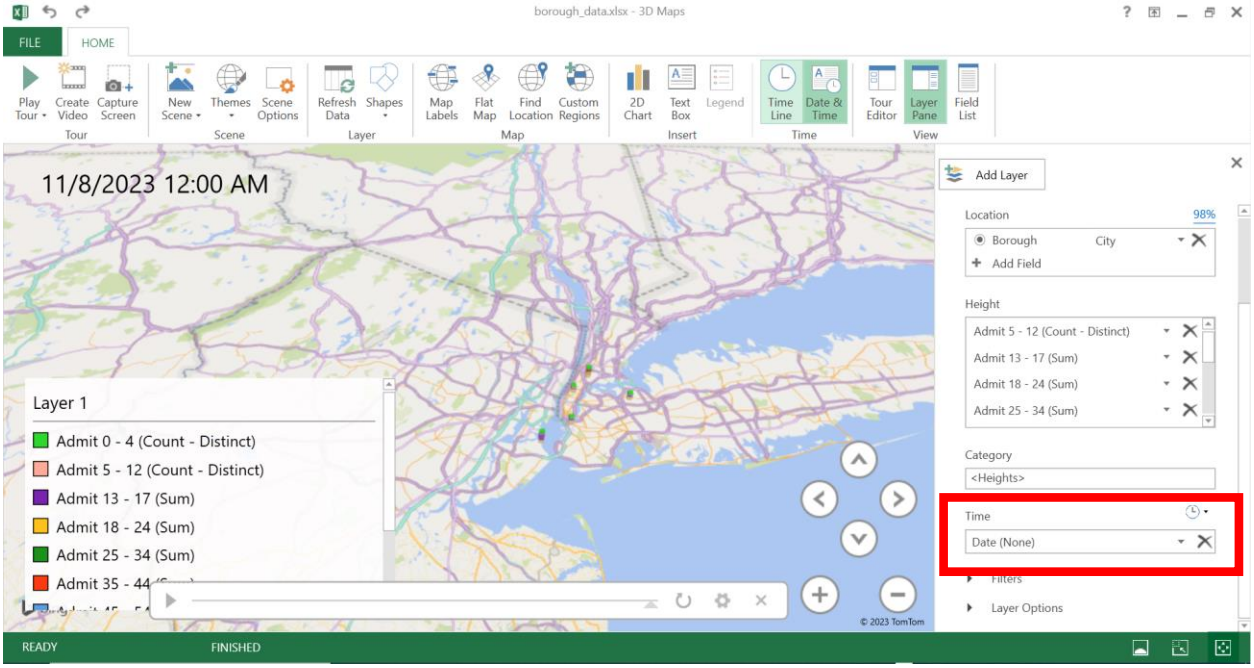


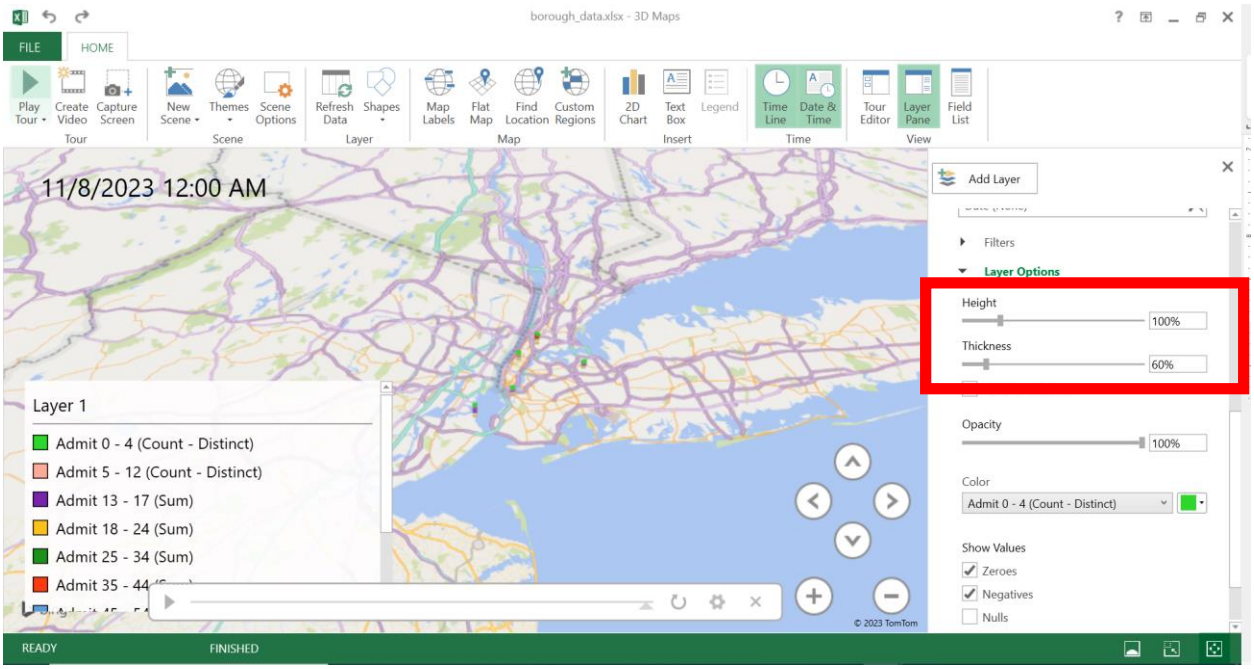
NOTE: If you do not see the layer frame on the right side of the 3D Map, you need to select all data manually before opening the 3D map.

8. Make "Borough" and "City" the location. This should be set by default. Then, add all the Admit columns to the Height. Make **Admit 0 – 4** and **Admit 5 – 12** as "(Count – Distinct)," and leave the others as "(Sum)." Also, you may close the Tour 1 window and the Field List window.

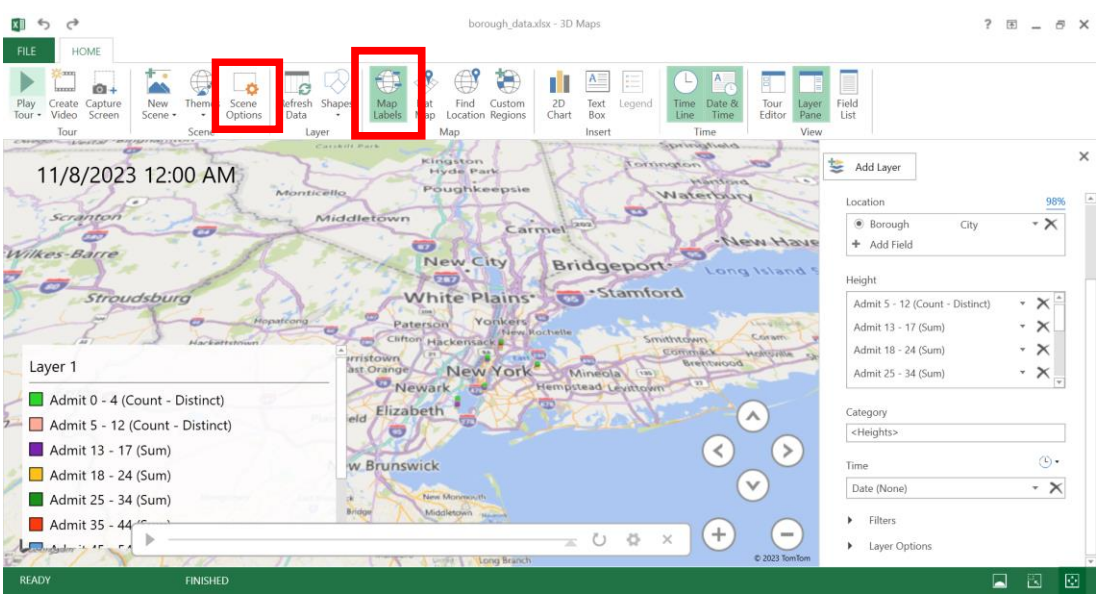


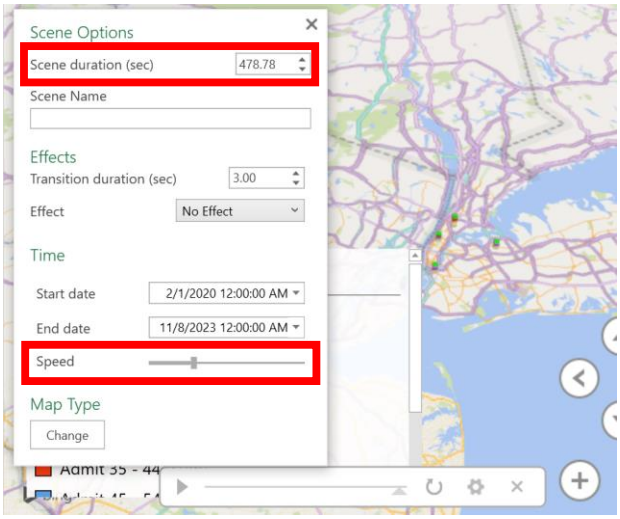
9. Then, in the Time field, use "Date (None)." In the Layer Options drop down, set the Height to **100%** and the Thickness to **60%**.



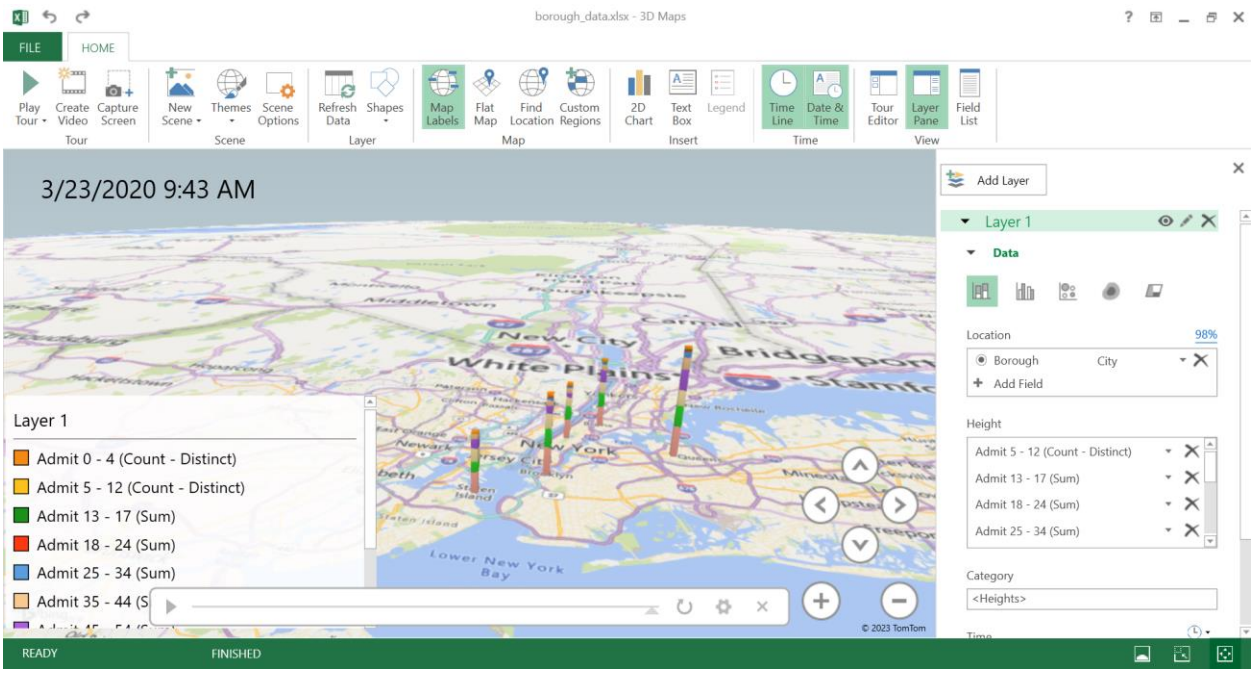


10. Select Map Labels to indicate the names of the locations. In the Scene Options Menu, slow down the Speed to the point where Scene Duration (sec) lasts about **478.78 seconds**. For the purposes of our experiment, that is the speed we chose, but feel free to adjust it to your liking. You can close the pop-out menus when complete and press play to see the bars move, indicating periods when there were high and low amounts of COVID-19.





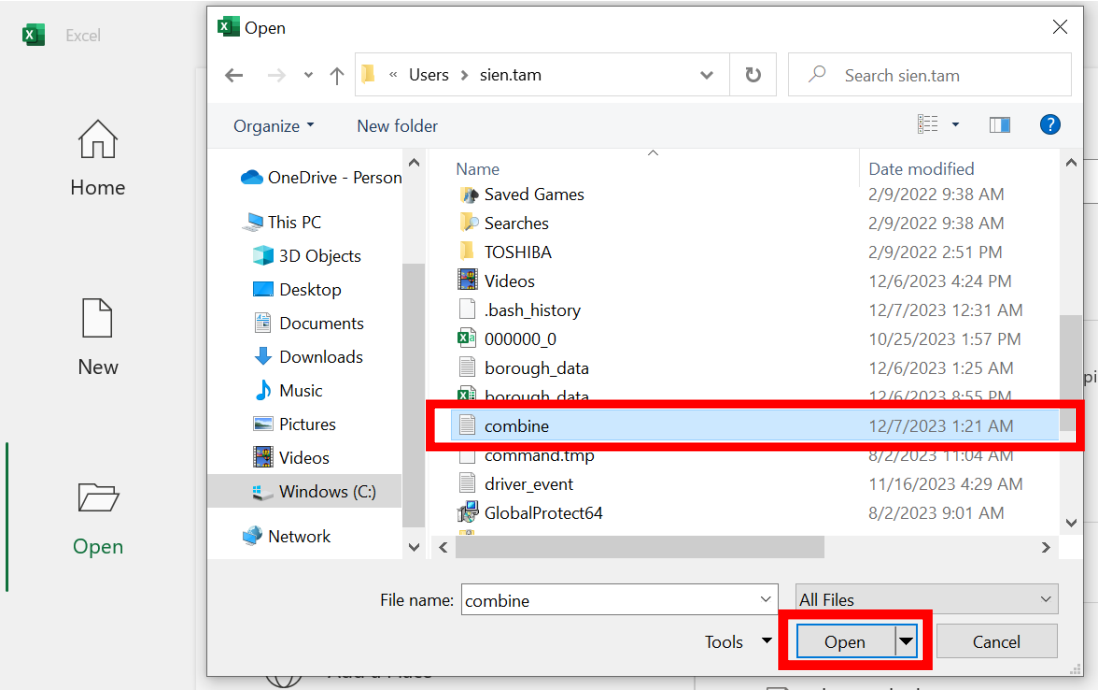
You should get the following result around **March 23, 2020, at 9:43 AM:**



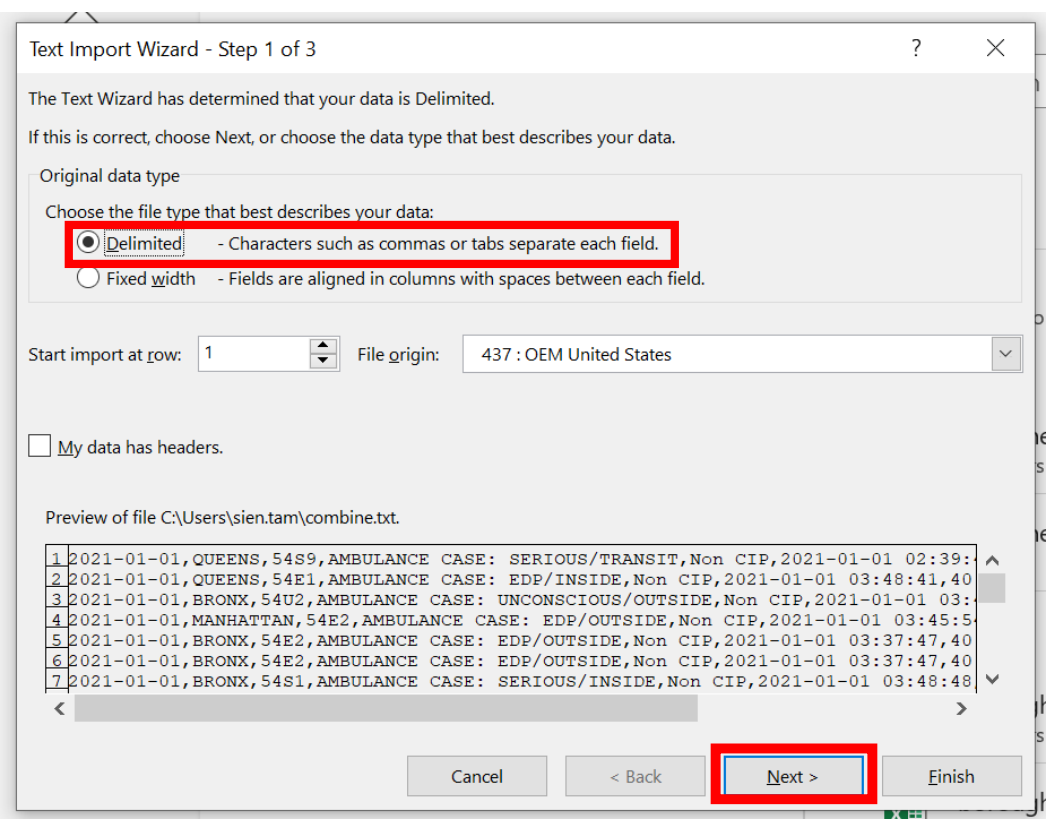
Step 6: Visualization in Excel 3D Map with Combine.txt

This step is to use Excel's 3D Map feature to conduct a tempo-spatial analysis of the Combine.txt file.

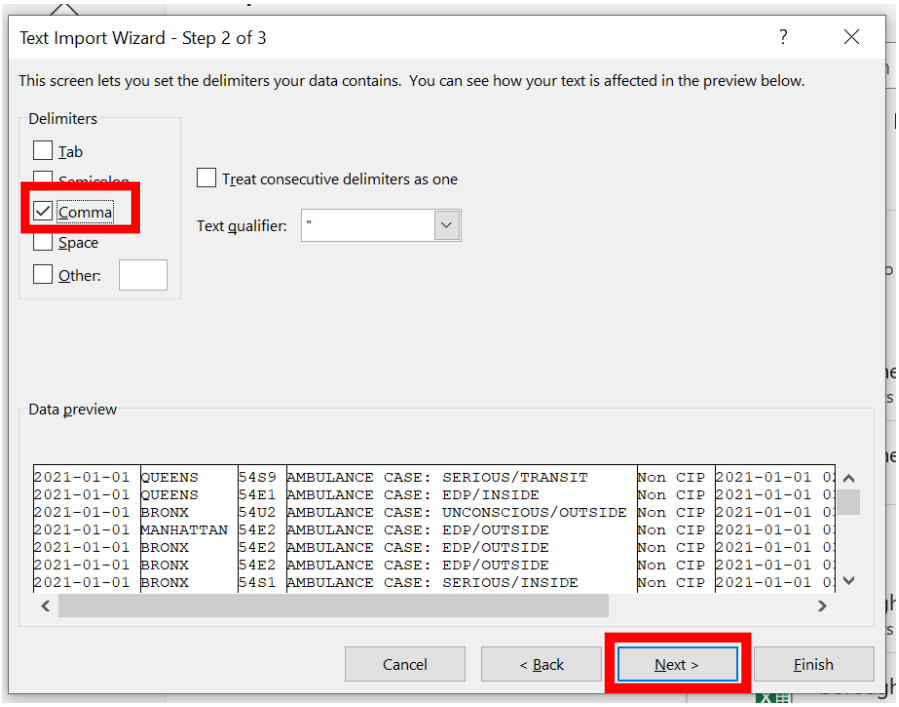
1. Open the file, combine.txt, in Excel. You have to open Excel first. **This is a larger file, so it can take some time throughout the following steps to load and process.**



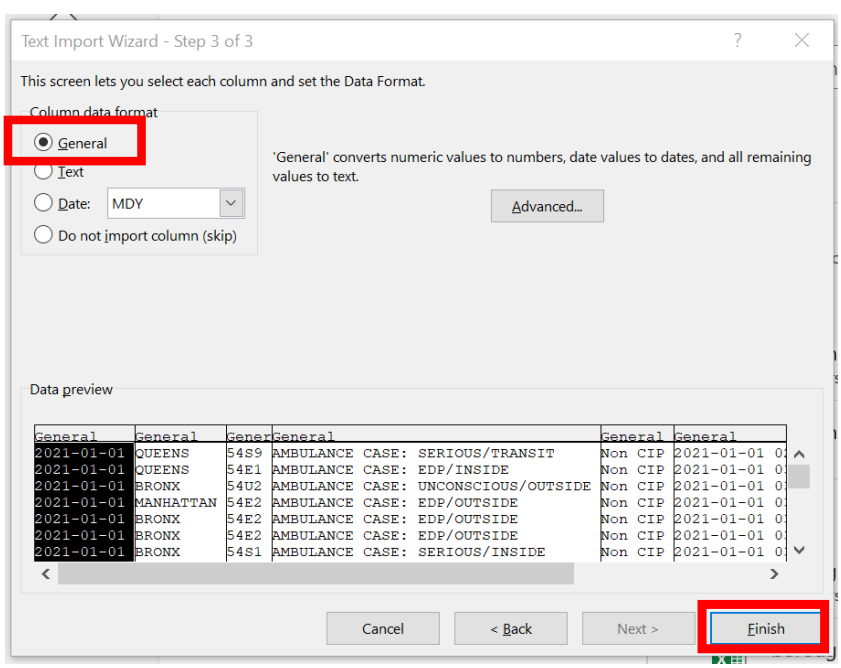
2. Then, in the first page of the Text Import Wizard, make sure “Delimited” is selected. Then press “Next.”



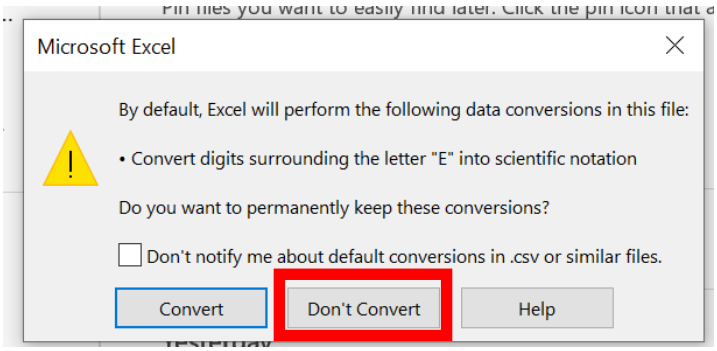
3. Then, in the second page of the Text Import Wizard, make sure “Comma” is selected instead of “tab.” Then press “Next.”



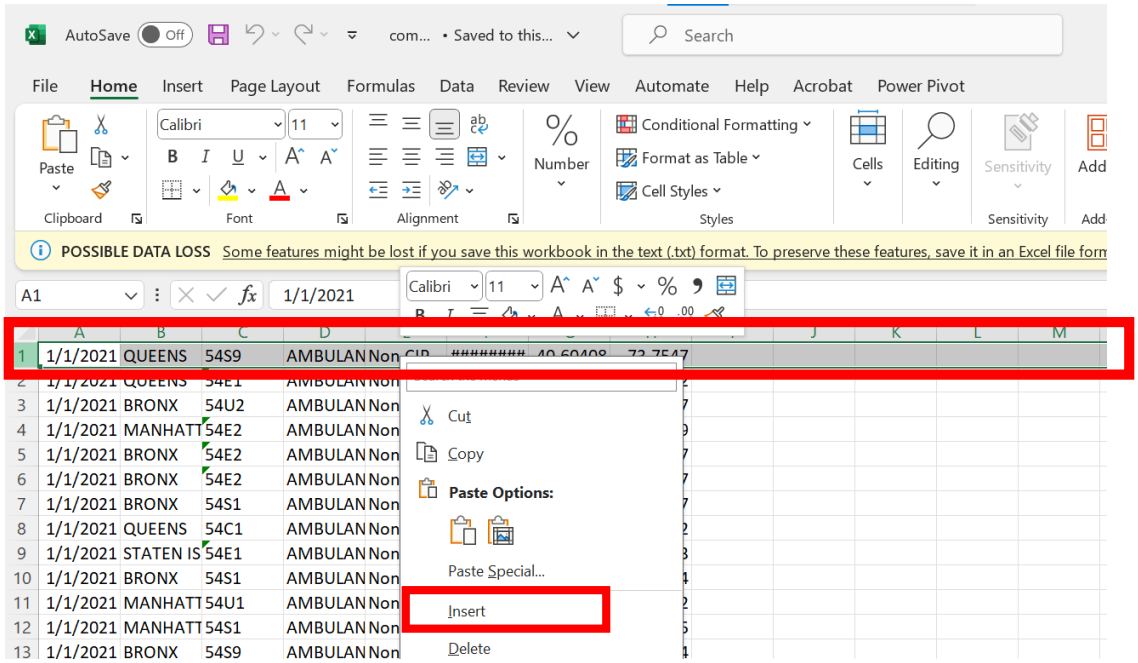
4. Then, in the third page of the Text Import Wizard, make sure “General” is selected for all columns. Then press “Finish.”

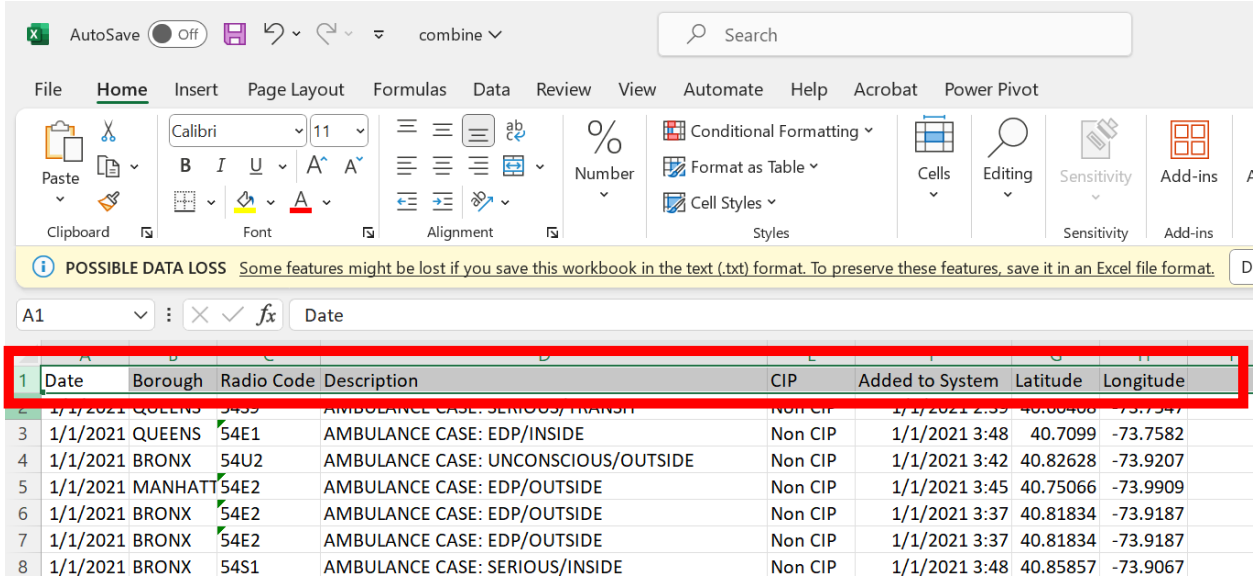


Note: If a pop-up appears asking you to process data conversions, press **Don't Convert**.



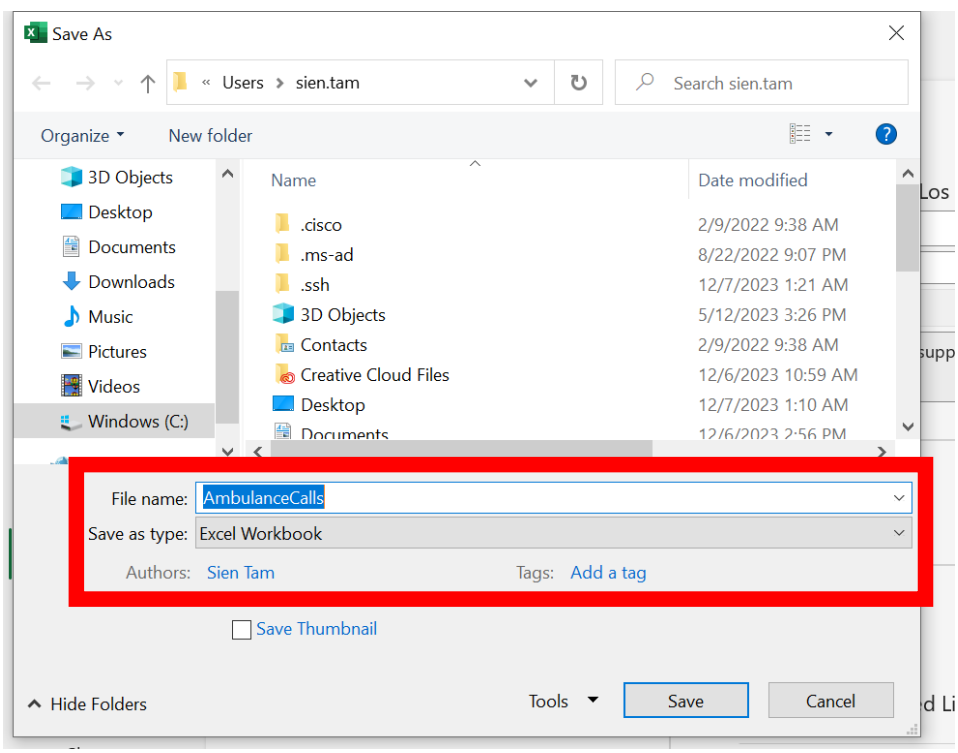
5. Upon reaching the Excel spreadsheet view, insert a new row above the current row 1. In the first row, insert the header of each column in this order: **Date, Borough, Radio Code, Description, CIP, Added to System, Latitude, and Longitude. Note:** You may need to adjust the sizes of the columns to see the data points.



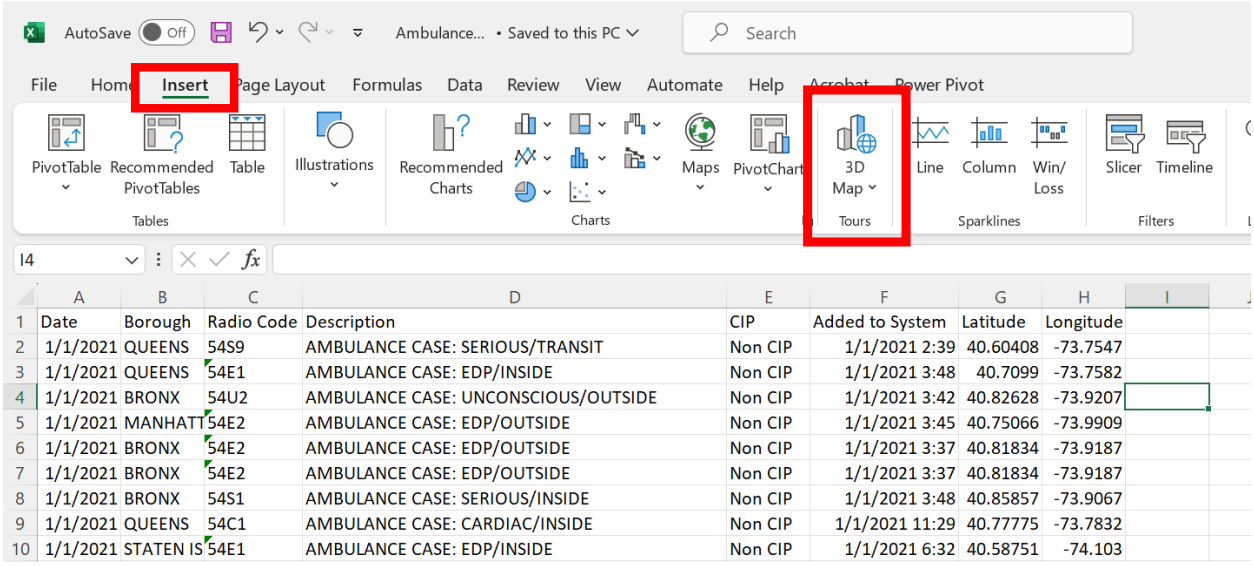


	Date	Borough	Radio Code	Description	CIP	Added to System	Latitude	Longitude
3	1/1/2021	QUEENS	54E1	AMBULANCE CASE: EDP/INSIDE	Non CIP	1/1/2021 3:48	40.7099	-73.7582
4	1/1/2021	BRONX	54U2	AMBULANCE CASE: UNCONSCIOUS/OUTSIDE	Non CIP	1/1/2021 3:42	40.82628	-73.9207
5	1/1/2021	MANHATT	54E2	AMBULANCE CASE: EDP/OUTSIDE	Non CIP	1/1/2021 3:45	40.75066	-73.9909
6	1/1/2021	BRONX	54E2	AMBULANCE CASE: EDP/OUTSIDE	Non CIP	1/1/2021 3:37	40.81834	-73.9187
7	1/1/2021	BRONX	54E2	AMBULANCE CASE: EDP/OUTSIDE	Non CIP	1/1/2021 3:37	40.81834	-73.9187
8	1/1/2021	BRONX	54S1	AMBULANCE CASE: SERIOUS/INSIDE	Non CIP	1/1/2021 3:48	40.85857	-73.9067

6. To visualize the results on Excel 3D Map, convert csv file to “Excel Workbook” (.xlsx) and rename the file as “AmbulanceCalls.”



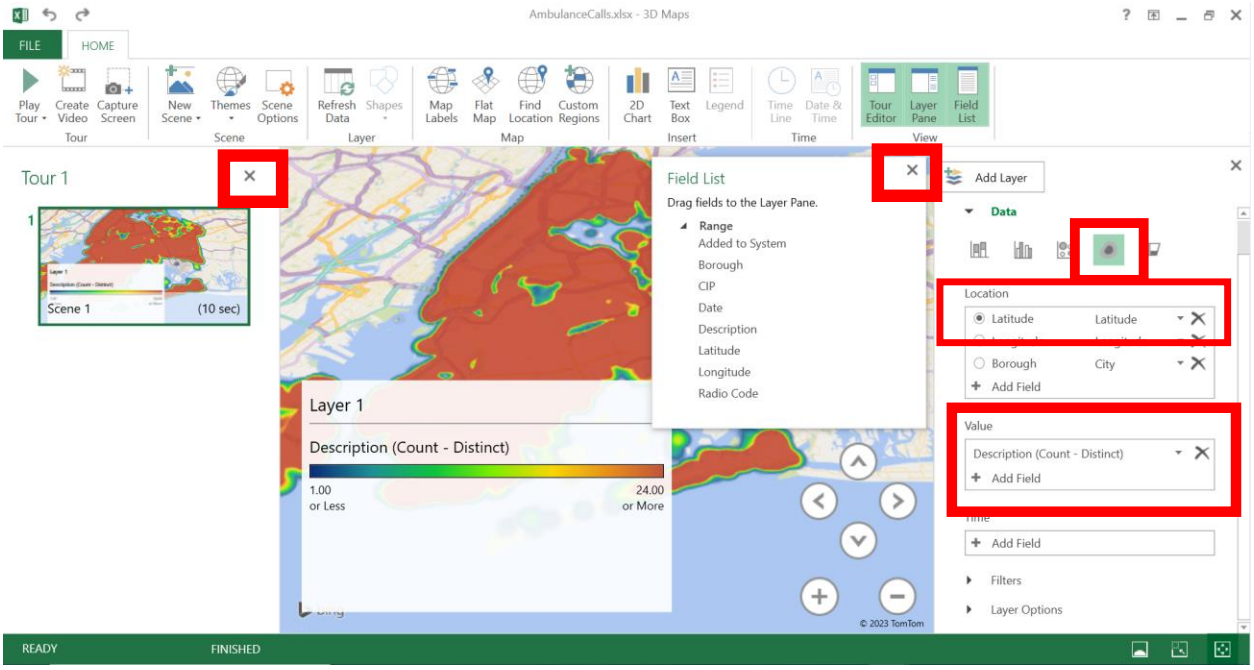
7. Then, go to the “Insert” tab at the top and click “3D Map.”



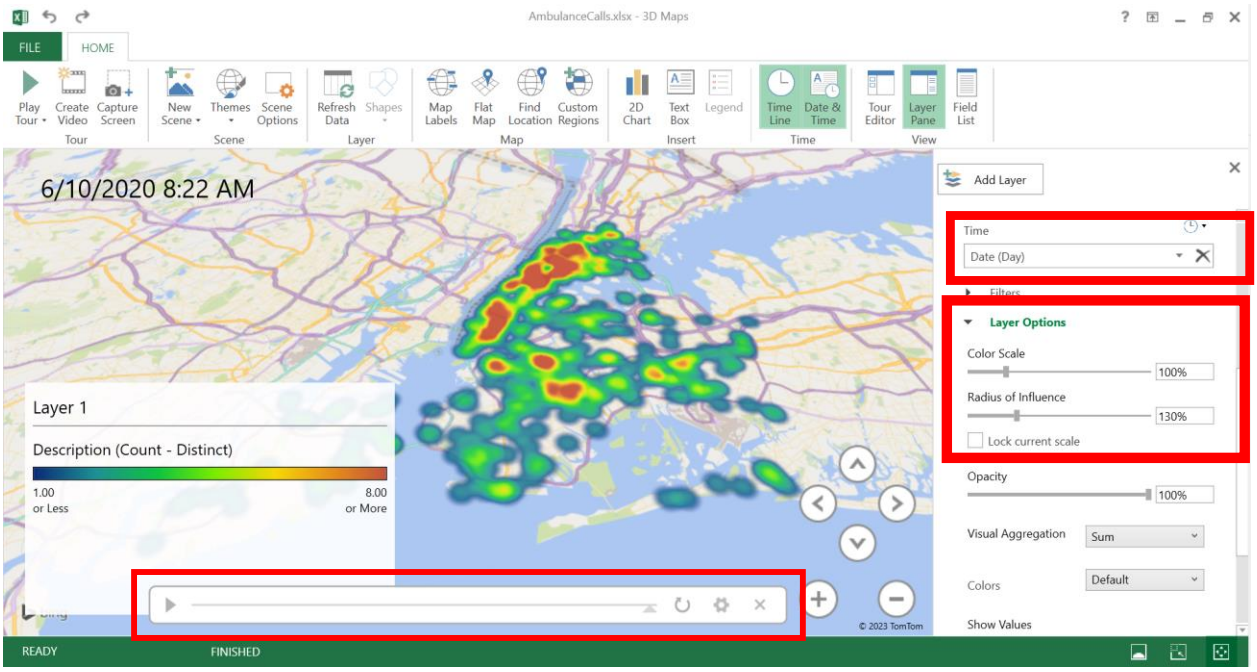
	A	B	C	D	E	F	G	H	I
1	Date	Borough	Radio Code	Description	CIP	Added to System	Latitude	Longitude	
2	1/1/2021	QUEENS	54S9	AMBULANCE CASE: SERIOUS/TRANSIT	Non CIP	1/1/2021 2:39	40.60408	-73.7547	
3	1/1/2021	QUEENS	54E1	AMBULANCE CASE: EDP/INSIDE	Non CIP	1/1/2021 3:48	40.7099	-73.7582	
4	1/1/2021	BRONX	54U2	AMBULANCE CASE: UNCONSCIOUS/OUTSIDE	Non CIP	1/1/2021 3:42	40.82628	-73.9207	
5	1/1/2021	MANHATT	54E2	AMBULANCE CASE: EDP/OUTSIDE	Non CIP	1/1/2021 3:45	40.75066	-73.9909	
6	1/1/2021	BRONX	54E2	AMBULANCE CASE: EDP/OUTSIDE	Non CIP	1/1/2021 3:37	40.81834	-73.9187	
7	1/1/2021	BRONX	54E2	AMBULANCE CASE: EDP/OUTSIDE	Non CIP	1/1/2021 3:37	40.81834	-73.9187	
8	1/1/2021	BRONX	54S1	AMBULANCE CASE: SERIOUS/INSIDE	Non CIP	1/1/2021 3:48	40.85857	-73.9067	
9	1/1/2021	QUEENS	54C1	AMBULANCE CASE: CARDIAC/INSIDE	Non CIP	1/1/2021 11:29	40.77775	-73.7832	
10	1/1/2021	STATEN IS	54E1	AMBULANCE CASE: EDP/INSIDE	Non CIP	1/1/2021 6:32	40.58751	-74.103	

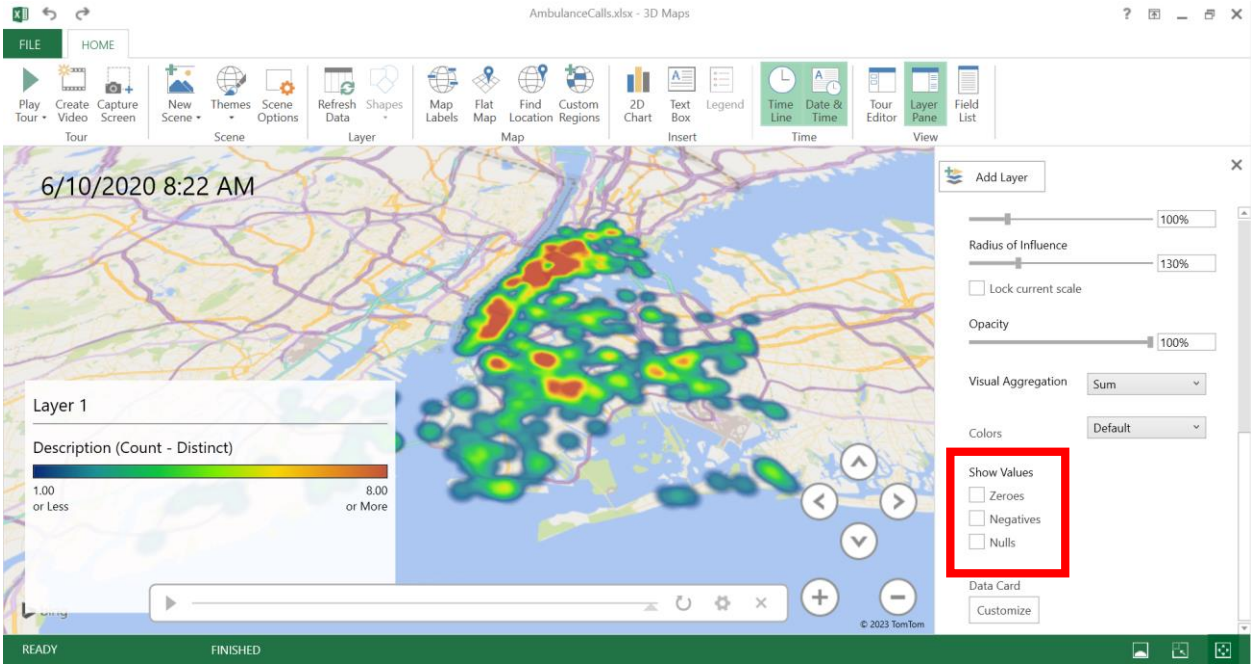
NOTE: If you do not see the layer frame on the right side of the 3D Map, you need to select all data manually before opening the 3D map.

8. Select the **Heat Map**. Then make “Latitude” and “Latitude” the location. This should be set by default. Then, add “Description (Count – Distinct)” column to the value section. Also, you may close the Tour 1 window and the Field List window. **Please be sure to wait for your settings to be processed throughout the following steps.**

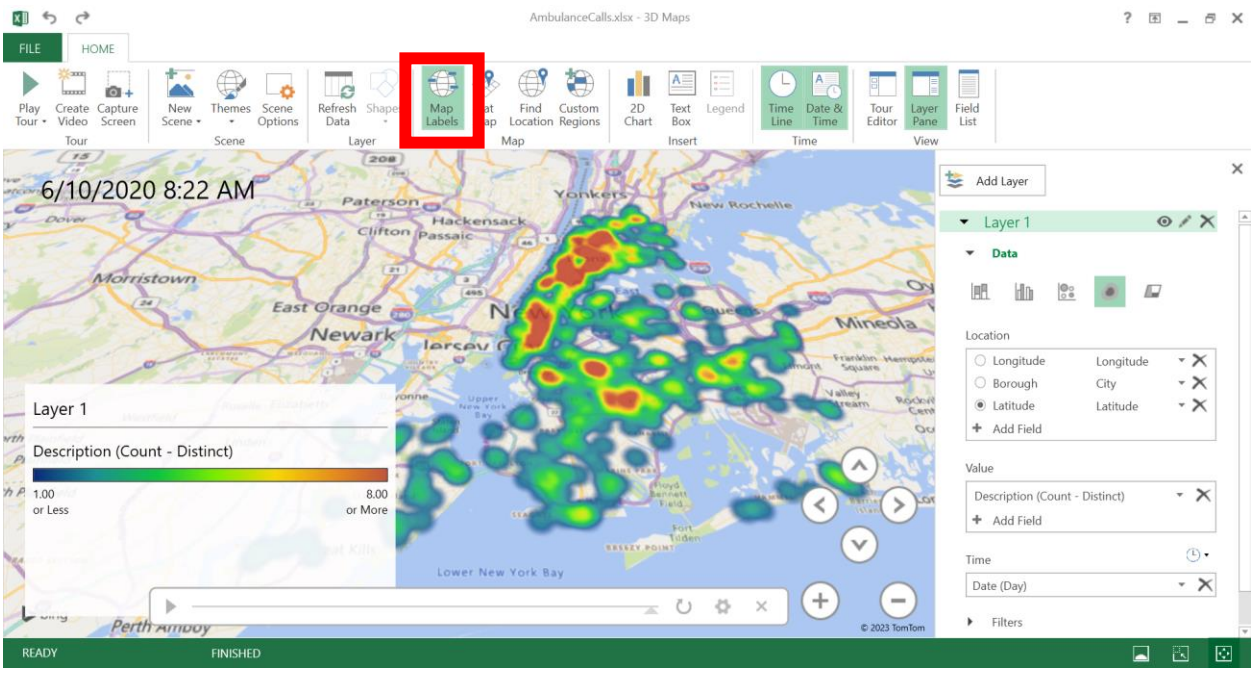


9. Then, in the Time field, use "Date (Day)." In the Layer Options drop down, set the Color Scale to **100%** and the Radius of Influence on **130%**. Also, uncheck the "Show Values" section, specifically Zeroes and Negatives. ***Please wait for the timeline bar to show up at the bottom of the screen and for processing to be completed before proceeding.***

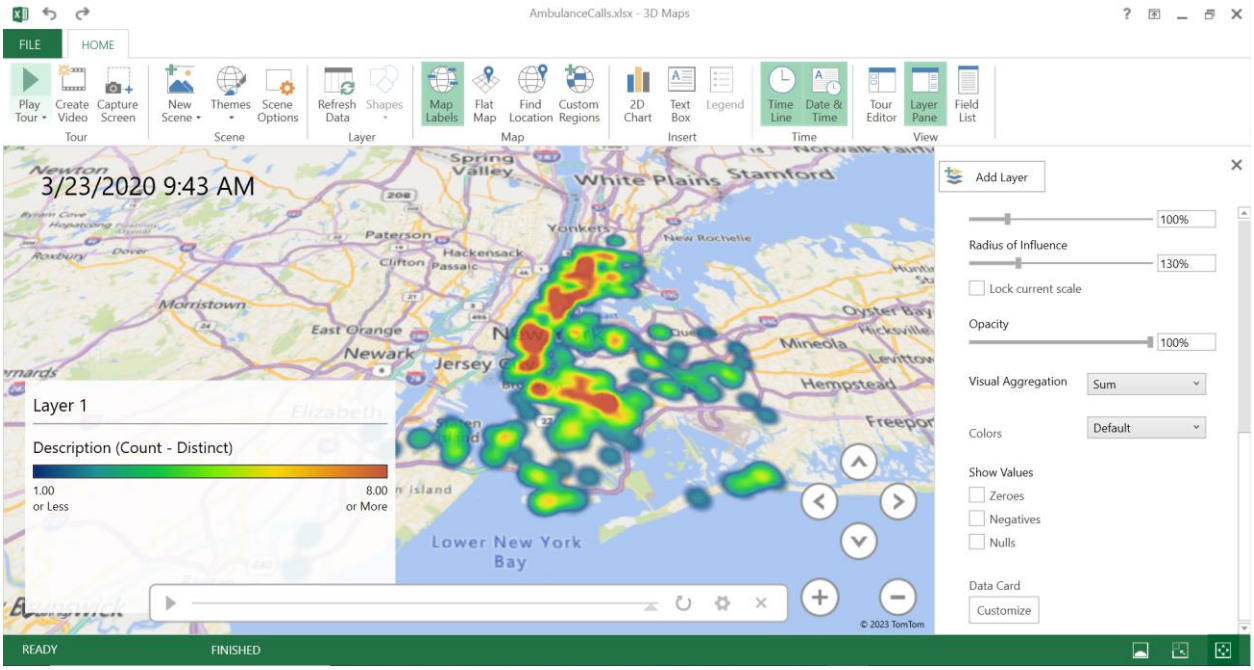




10. Enable the Map Labels to know the names of the locations.



You should get the following result around **March 23, 2020, at 9:43 AM:**



Step 7: Query and Download Monthly Total Averages into your PC

This step is to use HiveQL subqueries to generate monthly total averages of ambulance calls and COVID-19 cases from the year 2020 to 2021 and download the data into your PC.

1. Return to the second Git Bash terminal (the one containing HiveQL) and enter the following

   code to create a text file containing the monthly total averages of ambulance calls in 2020.

```
INSERT OVERWRITE DIRECTORY '/user/jtam2/Project/export_2020' ROW
FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE Select
month, avg(count) as monthly_avg from(SELECT MONTH(create_date) AS
month, DAY(create_date) AS day, count(typ_desc) AS count FROM
Nypd_calls Where year(create_date) = '2020' and typ_desc like
'%AMBULANCE%' GROUP BY MONTH(create_date), DAY(create_date) Order
by month) temp group by month order by month;
```

To view the content of the query, remove the portion of the code that saves the file:

```
Select month, avg(count) as monthly_avg from(SELECT
MONTH(create_date) AS month, DAY(create_date) AS day,
count(typ_desc) AS count FROM Nypd_calls Where year(create_date) =
'2020' and typ_desc like '%AMBULANCE%' GROUP BY MONTH(create_date),
DAY(create_date) Order by month) temp group by month order by
month;
```

You will see a result like the following:

```
+--------+---------------------+
| month  |     monthly_avg     |
+--------+---------------------+
| 1      | 2338.0967741935483  |
| 2      | 2276.3793103448274  |
| 3      | 1983.774193548387   |
| 4      | 1630.9333333333334  |
| 5      | 1739.9354838709678  |
| 6      | 1936.4666666666667  |
| 7      | 1989.6129032258063  |
| 8      | 1950.6774193548388  |
| 9      | 1914.7              |
| 10     | 1860.774193548387   |
| 11     | 1872.4              |
| 12     | 1892.6129032258063  |
+--------+---------------------+
12 rows selected (75.374 seconds)
```

2. Then, enter the following code to create a text file containing the monthly total averages of ambulance calls in 2021.

```
INSERT OVERWRITE DIRECTORY '/user/jtam2/Project/export_2021' ROW
FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE Select
month, avg(count) as monthly_avg from(SELECT MONTH(create_date) AS
month, DAY(create_date) AS day, count(typ_desc) AS count FROM
Nypd_calls Where year(create_date) = '2021' and typ_desc like
'%AMBULANCE%' GROUP BY MONTH(create_date), DAY(create_date) Order
by month) temp group by month order by month;
```

To view the content of the query, remove the portion of the code that saves the file:

```
Select month, avg(count) as monthly_avg from(SELECT
MONTH(create_date) AS month, DAY(create_date) AS day,
count(typ_desc) AS count FROM Nypd_calls Where year(create_date) =
'2021' and typ_desc like '%AMBULANCE%' GROUP BY MONTH(create_date),
DAY(create_date) Order by month) temp group by month order by
month;
```

You will see a result like the following:

```
+--------+---------------------+
| month  |     monthly_avg     |
+--------+---------------------+
| 1      | 1733.258064516129   |
| 2      | 1661.357142857143   |
| 3      | 1684.0              |
| 4      | 1697.0              |
| 5      | 1745.0              |
| 6      | 1815.9666666666667  |
| 7      | 1776.9354838709678  |
| 8      | 1791.516129032258   |
| 9      | 1775.8              |
| 10     | 1794.225806451613   |
| 11     | 1669.7              |
| 12     | 1706.9354838709678  |
+--------+---------------------+
12 rows selected (65.91 seconds)
```

3. Then, enter the following code to create a text file containing the monthly total averages of COVID-19 cases in 2020.

```
INSERT OVERWRITE DIRECTORY '/user/jtam2/Project/export_2020_covid'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
Select month, avg(sum) as monthly_avg from( SELECT MONTH(days) AS
month, DAY(days) AS day, sum(admit_all_ages) AS sum FROM
All_borough_covid Where year(days) = '2020' GROUP BY days Order by
month,days) temp group by month order by month;
```

To view the content of the query, remove the portion of the code that saves the file:

```
Select month, avg(sum) as monthly_avg from( SELECT MONTH(days) AS
month, DAY(days) AS day, sum(admit_all_ages) AS sum FROM
All_borough_covid Where year(days) = '2020' GROUP BY days Order by
month,days) temp group by month order by month;
```

You will see a result like the following:

```
+--------+---------------------+
| month  |     monthly_avg     |
+--------+---------------------+
| 2      | 8.362397985529777   |
| 3      | 26.69612539900416   |
| 4      | 31.98072770027502   |
| 5      | 5.463712035013657   |
| 6      | 3.0982977370337377  |
| 7      | 3.5851699435018456  |
| 8      | 3.9075026543093965  |
| 9      | 4.097234857513678   |
| 10     | 4.67468647502288    |
| 11     | 7.554886815943046   |
| 12     | 12.761091793493778  |
+--------+---------------------+
11 rows selected (21.326 seconds)
```

4. Then, enter the following code to create a text file containing the monthly total averages of COVID-19 cases in 2021.

```
INSERT OVERWRITE DIRECTORY '/user/jtam2/Project/export_2021_covid'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
Select month, avg(sum) as monthly_avg from(SELECT MONTH(days) AS
month, DAY(days) AS day, sum(admit_all_ages) AS sum FROM
All_borough_covid Where year(days) = '2021' GROUP BY days Order by
month,days) temp group by month order by month;
```

To view the content of the query, remove the portion of the code that saves the file:

```
Select month, avg(sum) as monthly_avg from(SELECT MONTH(days) AS
month, DAY(days) AS day, sum(admit_all_ages) AS sum FROM
All_borough_covid Where year(days) = '2021' GROUP BY days Order by
month,days) temp group by month order by month;
```

You will see a result like the following:

```
+--------+---------------------+
| month  |     monthly_avg     |
+--------+---------------------+
| 1      | 16.755909470007776  |
| 2      | 16.589602027714648  |
| 3      | 12.970537064355003  |
| 4      | 9.375418152443192   |
| 5      | 5.07680626189958    |
| 6      | 4.314765384243634   |
| 7      | 5.002093479244949   |
| 8      | 7.933090967496386   |
| 9      | 7.33540079004887    |
| 10     | 8.250365391718901   |
| 11     | 5.971116124481338   |
| 12     | 12.454811237632061  |
+--------+---------------------+
12 rows selected (19.23 seconds)
```

5. Return to the first Git Bash terminal which is the Linux File System. Insert the following commands to display the contents of the directory **/user/jtam2/Project/export_2020/** and **/user/jtam2/Project/export_2021/**.

```
$ hdfs dfs -ls /user/jtam2/Project/export_2020/
$ hdfs dfs -ls /user/jtam2/Project/export_2021/
```

You will see a result like the following:

```
-bash-4.2$ hdfs dfs -ls /user/jtam2/Project/export_2020/
Found 1 items
-rw-r--r--   3 jtam2 hdfs        229 2023-12-15 07:56
/user/jtam2/Project/export_2020/000000_0
-bash-4.2$ hdfs dfs -ls /user/jtam2/Project/export_2021/
Found 1 items
-rw-r--r--   3 jtam2 hdfs        191 2023-12-15 07:44
/user/jtam2/Project/export_2021/000000_0
-bash-4.2$
```

6. Insert the following commands to display the contents of the directory **/user/jtam2/Project/export_2020_covid/** and **/user/jtam2/Project/export_2021_covid/**.

```
$ hdfs dfs -ls /user/jtam2/Project/export_2020_covid/
$ hdfs dfs -ls /user/jtam2/Project/export_2021_covid/
```

You will see a result like the following:

```
-bash-4.2$ hdfs dfs -ls /user/jtam2/Project/export_2020_covid/
Found 1 items
-rw-r--r--   3 jtam2 hdfs        226 2023-12-15 08:23
/user/jtam2/Project/export_2020_covid/000000_0
-bash-4.2$ hdfs dfs -ls /user/jtam2/Project/export_2021_covid/
Found 1 items
-rw-r--r--   3 jtam2 hdfs        245 2023-12-15 08:33
/user/jtam2/Project/export_2021_covid/000000_0
```

7. Then, enter the following commands to download the output files to the local file systems.

   Rename the file in the **export_2020** directory to **monthly_Avg_2020** and the file in the

   export_2021 directory to **monthly_Avg_2021**.

```
$ hdfs dfs -get /user/jtam2/Project/export_2020/000000_0
monthly_Avg_2020.txt

$ hdfs dfs -get /user/jtam2/Project/export_2021/000000_0
monthly_Avg_2021.txt
```

8. Then, enter the following commands to download the output files to the local file systems.

   Rename the file in the **export_2020_covid** directory to **monthly_Avg_Covid_2020** and the file in

   the **export_2021_covid** directory to **monthly_Avg_Covid_2021**.

```
$ hdfs dfs -get /user/jtam2/Project/export_2020_covid/000000_0
monthly_Avg_Covid_2020.txt

$ hdfs dfs -get /user/jtam2/Project/export_2021_covid/000000_0
monthly_Avg_Covid_2021.txt
```

9. Return to the third Git Bash terminal and insert the following code to download the data files to

   your local computer. **You do not need to SSH. Insert your password when prompted. Also, the**

   download could take a while depending on the data size.

```
$ scp jtam2@129.146.90.117:/home/jtam2/monthly_Avg_2020.txt .
$ scp jtam2@129.146.90.117:/home/jtam2/monthly_Avg_2021.txt .
$ scp jtam2@129.146.90.117:/home/jtam2/monthly_Avg_Covid_2020.txt .
$ scp jtam2@129.146.90.117:/home/jtam2/monthly_Avg_Covid_2021.txt .
```

You will see a result like the following:

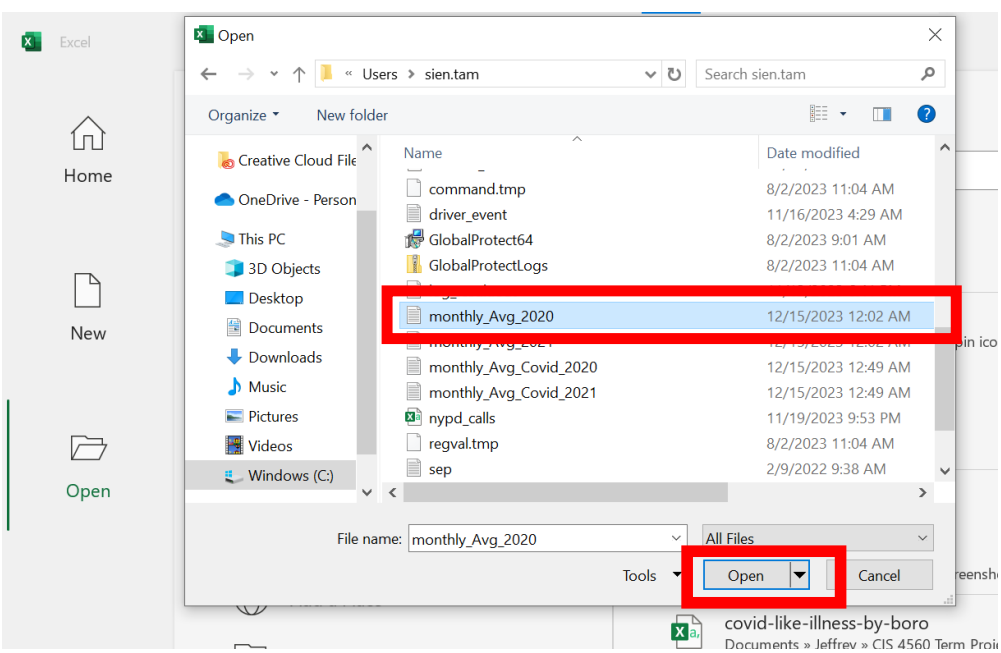
```
$ scp jtam2@129.146.90.117:/home/jtam2/monthly_Avg_2020.txt .
```

```
jtam2@129.146.90.117's password:
monthly_Avg_2020.txt
100%  229     7.3KB/s   00:00

$ scp jtam2@129.146.90.117:/home/jtam2/monthly_Avg_2021.txt .
jtam2@129.146.90.117's password:
monthly_Avg_2021.txt
100%  191     1.3KB/s   00:00

$ scp jtam2@129.146.90.117:/home/jtam2/monthly_Avg_Covid_2020.txt .
jtam2@129.146.90.117's password:
monthly_Avg_Covid_2020.txt
100%  226     6.8KB/s   00:00

$ scp jtam2@129.146.90.117:/home/jtam2/monthly_Avg_Covid_2021.txt .
jtam2@129.146.90.117's password:
monthly_Avg_Covid_2021.txt
100%  245     7.4KB/s   00:00
```

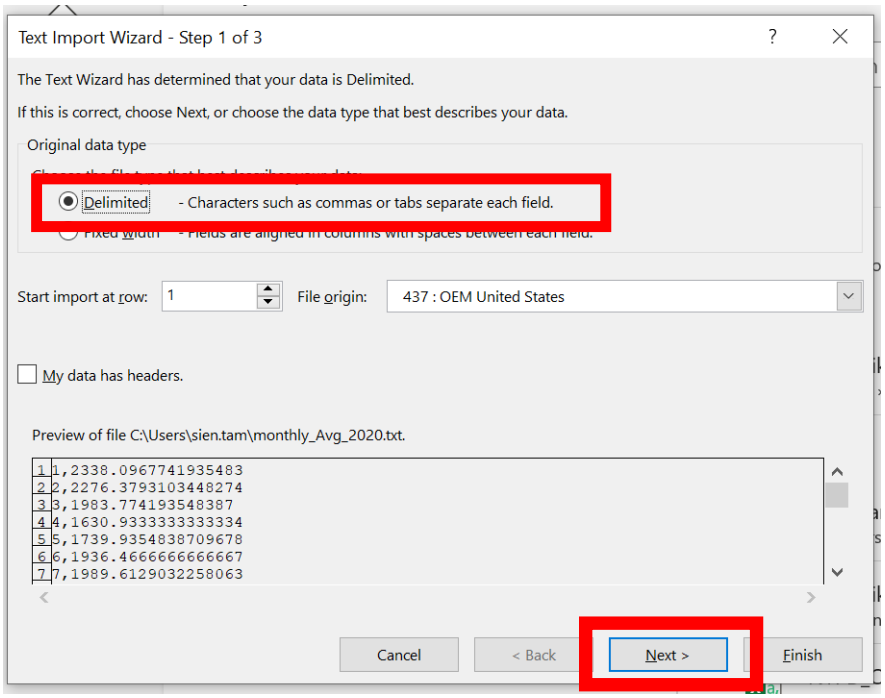
Step 8: Create Graphs to Compare the Monthly Total Averages in 2020 Using Excel

This step is to use Excel to generate graphs of the monthly total averages of 2020 and compare them. The data, which was exported separately, will all need to be copied and pasted into one Excel file (monthly_Avg_Covid_2020 → monthly_Avg_2020, etc.) and saved as an Excel workbook in order to have the data together for the creation of the graphs. Follow the directions carefully.

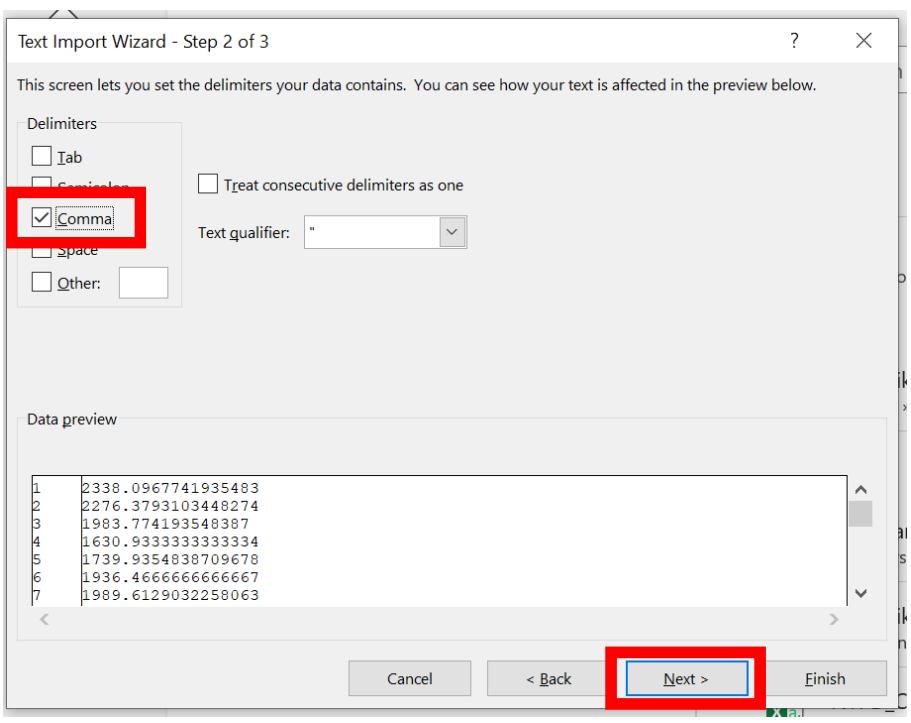
1. Open the file, **monthly_Avg_2020.txt**, in Excel. You have to open Excel first.



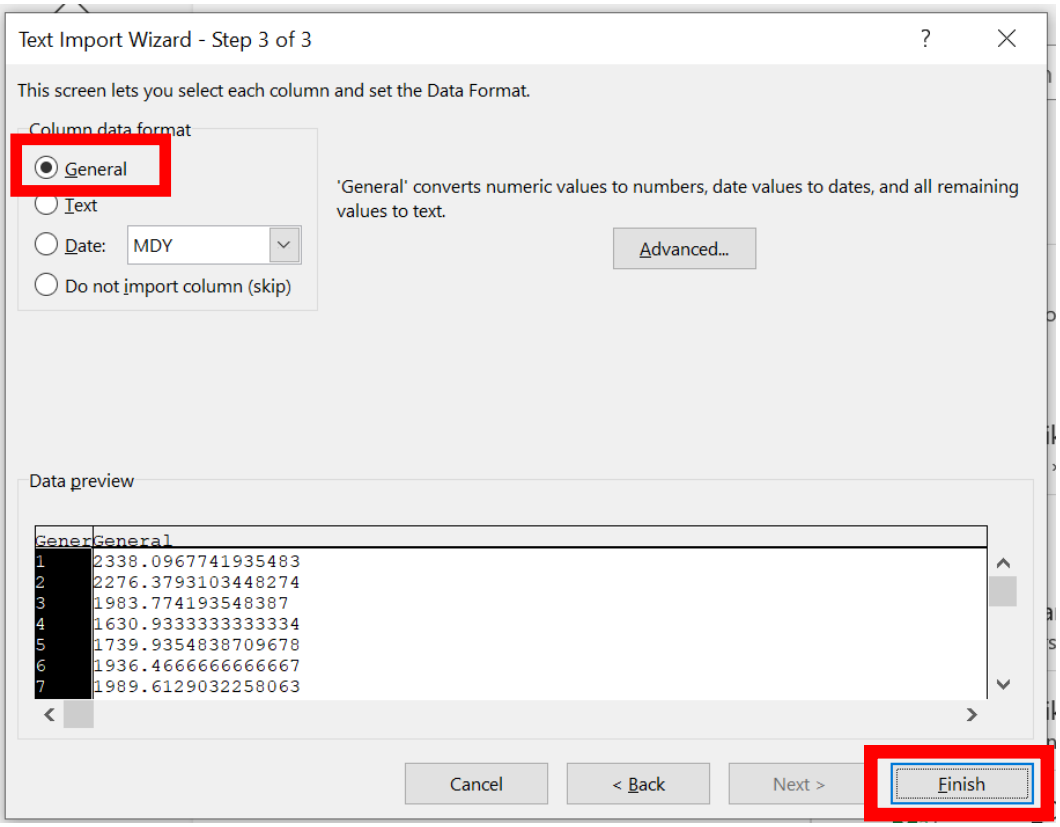
2. Then, in the first page of the Text Import Wizard, make sure "Delimited" is selected. Then press "Next."



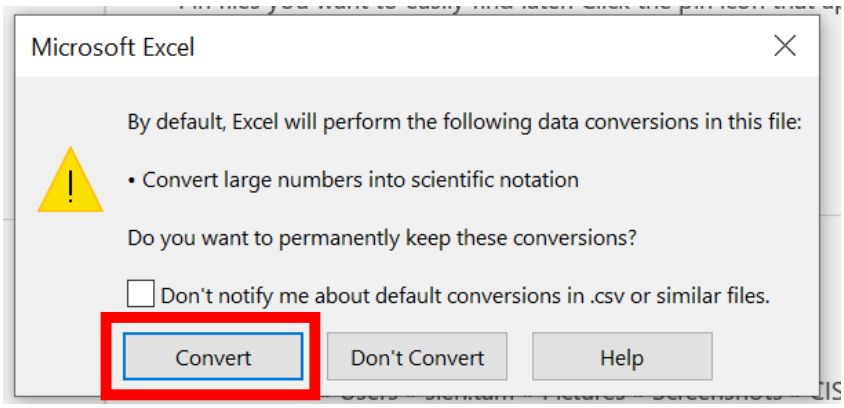
3. Then, in the second page of the Text Import Wizard, make sure “Comma” is selected instead of “tab.” Then press “Next.”



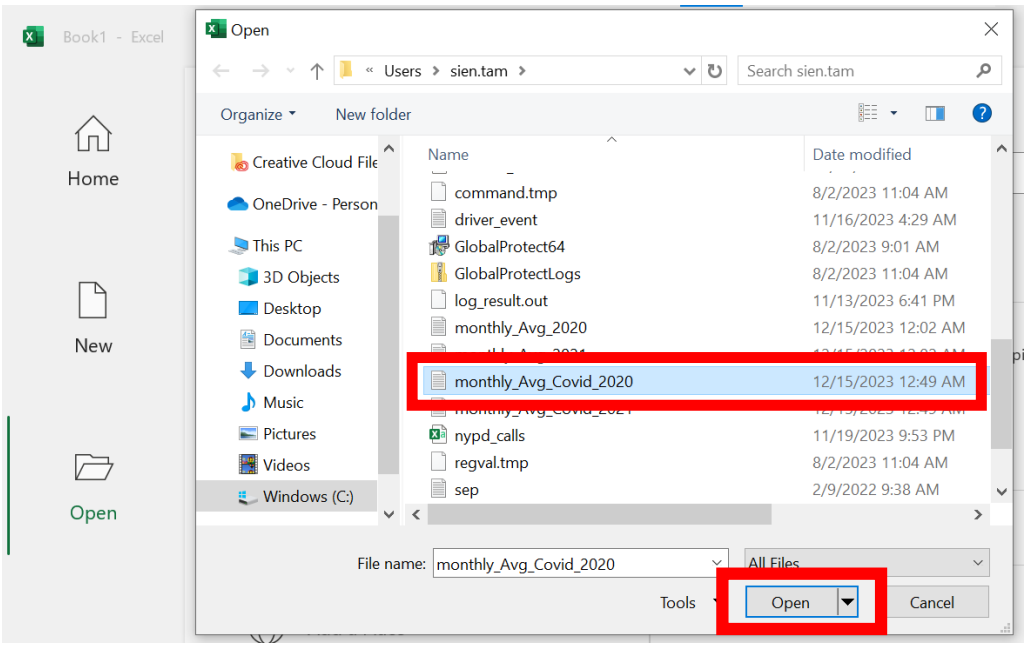
4. Then, in the third page of the Text Import Wizard, make sure “General” is selected for all columns. Then press “Finish.”



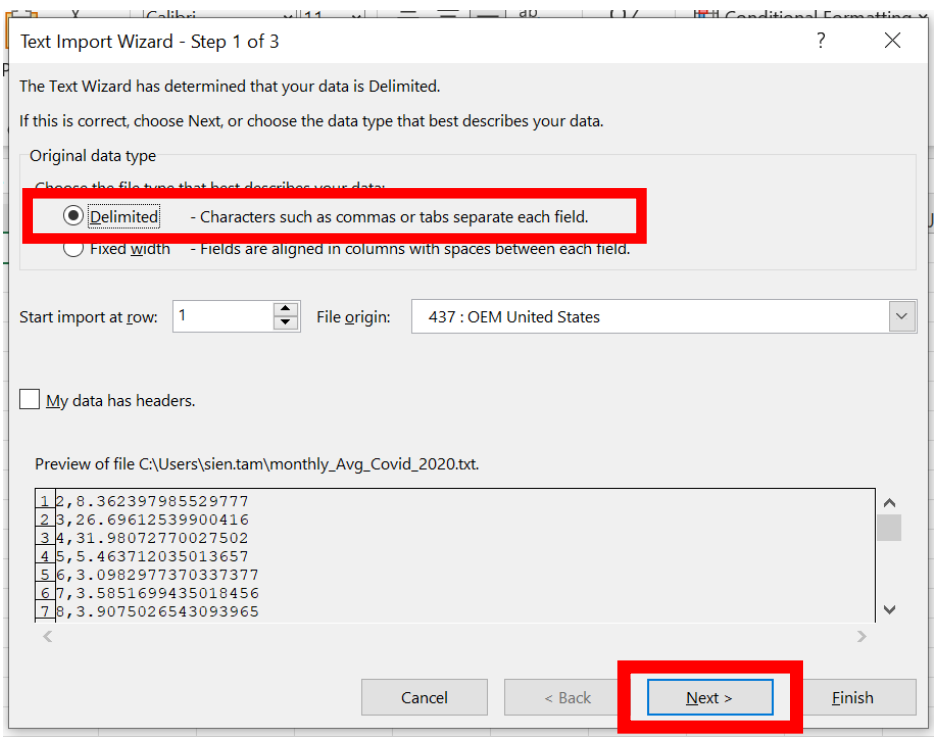
Note: If a pop-up appears asking you to process data conversions, press **Convert**. This conversion will make sure numbers stay numbers and deal with the decimals.



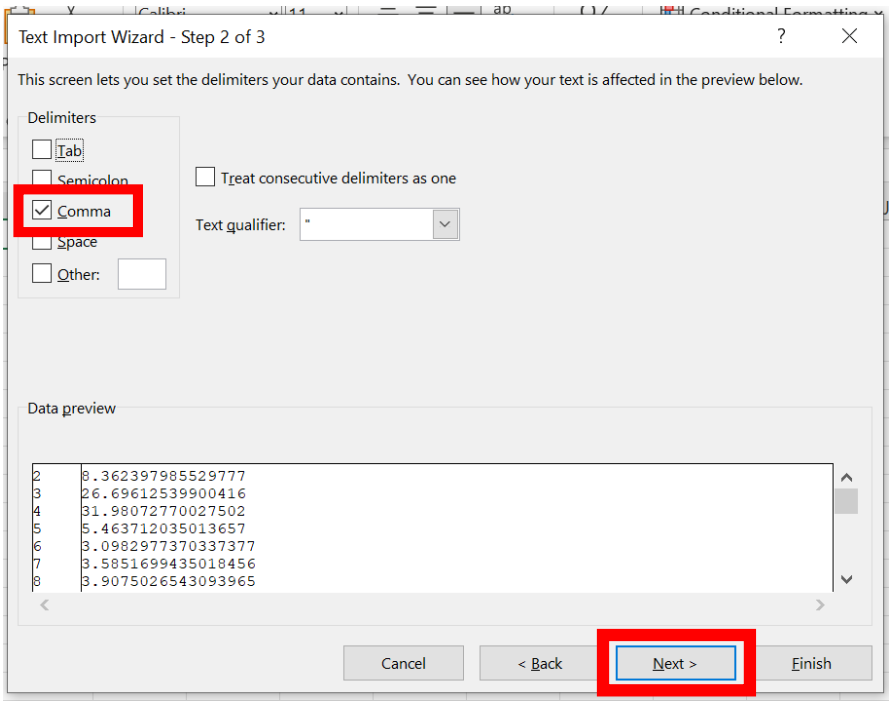
5. Open the file, **monthly_Ave_Covid_2020.txt**, in Excel. You have to open Excel first.



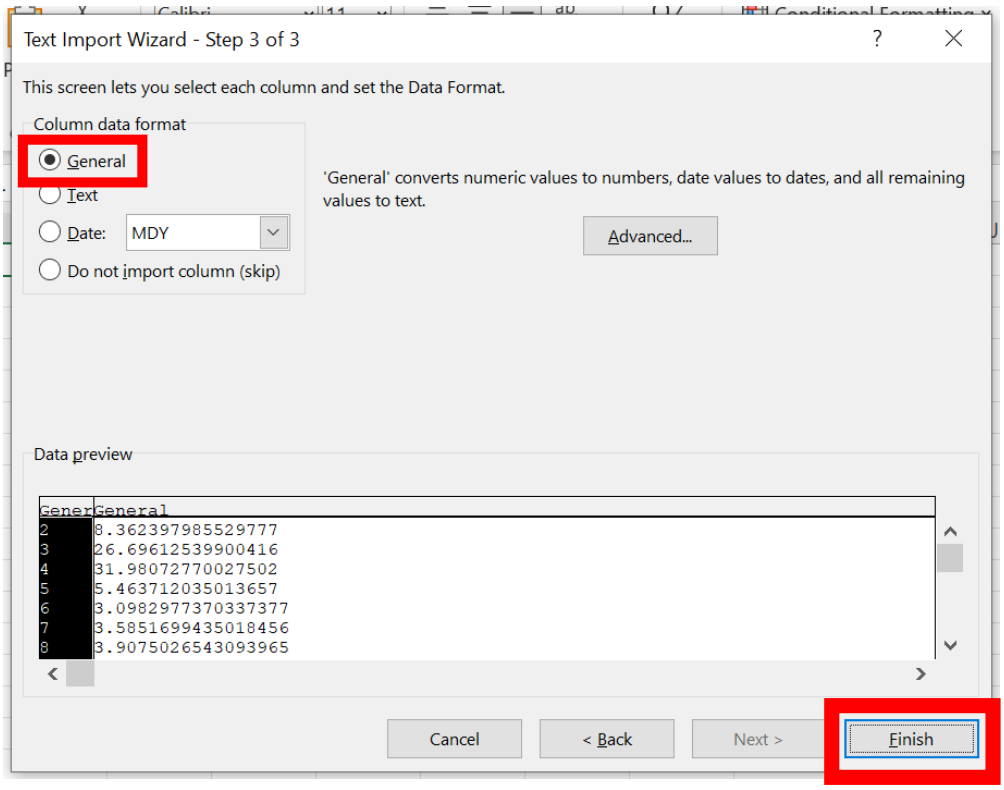
6. Then, in the first page of the Text Import Wizard, make sure “Delimited” is selected. Then press “Next.”



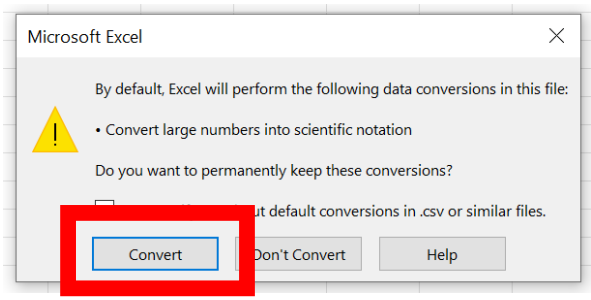
7. Then, in the second page of the Text Import Wizard, make sure “Comma” is selected instead of “tab.” Then press “Next.”



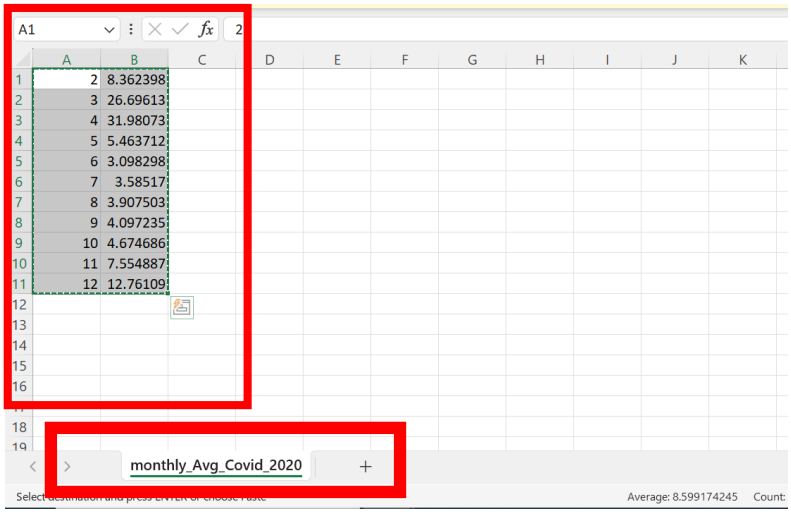
8. Then, in the third page of the Text Import Wizard, make sure "General" is selected for all columns. Then press "Finish."

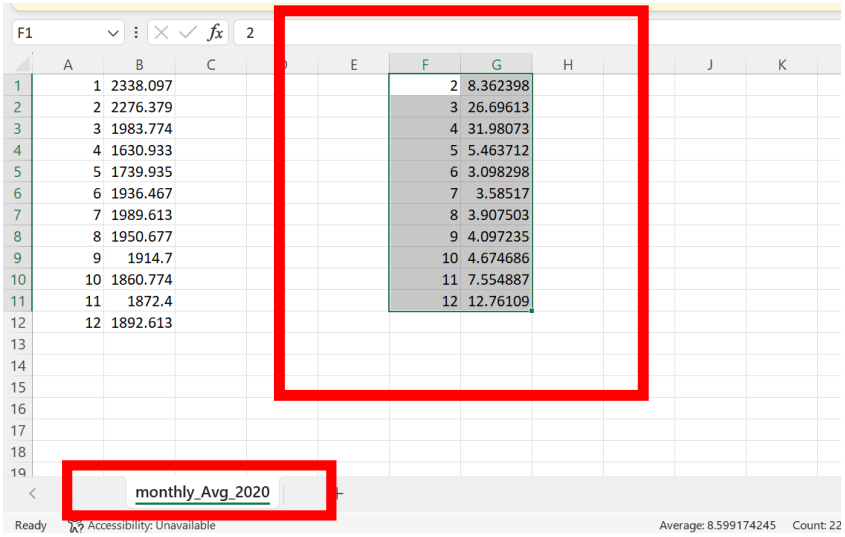


Note: If a pop-up appears asking you to process data conversions, press **Convert**. This conversion will make sure numbers stay numbers and deal with the decimals.

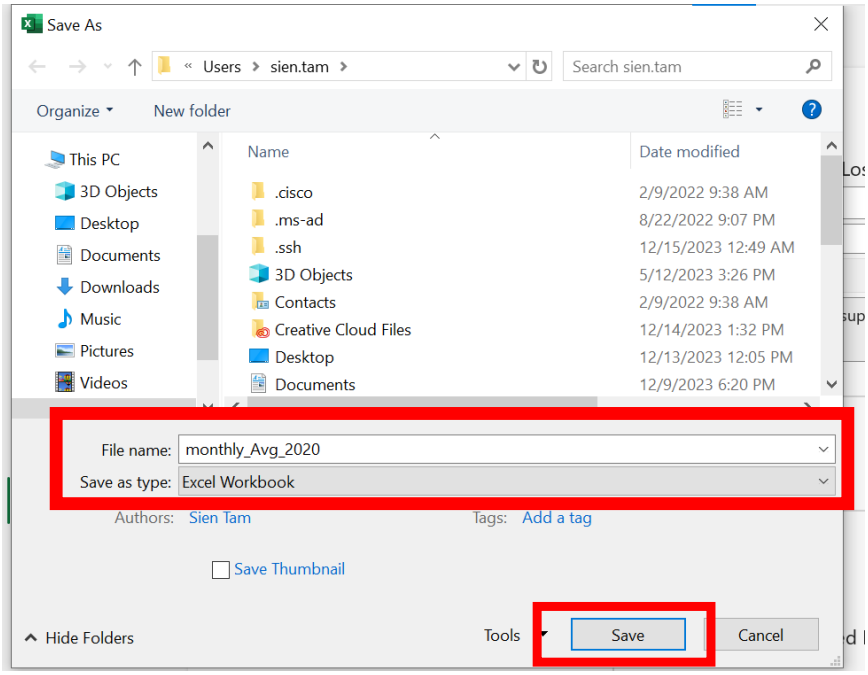


9. Then, select the data in the **monthly_Avg_Covid_2020** file, copy it (Ctrl + C), and paste it in the **monthly_Avg_2020** file in cell F1.

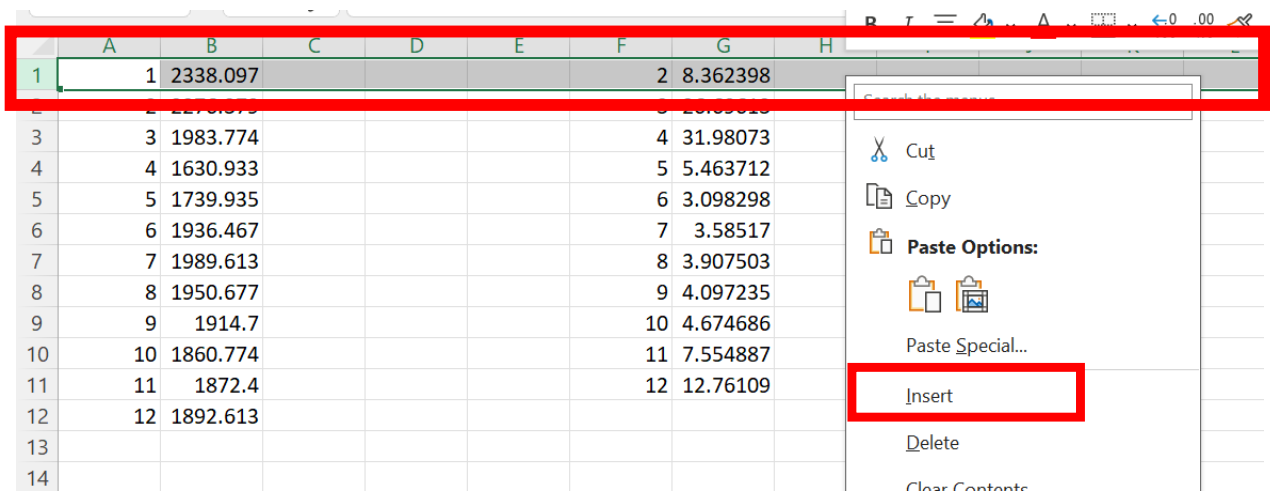




10. You may close the **monthly_Avg_Covid_2020** file. Save the current file “monthly_Avg_2020” as the same name as an **Excel Workbook (.xlsx)**:



11. Then, insert **two** rows above the current row 1.

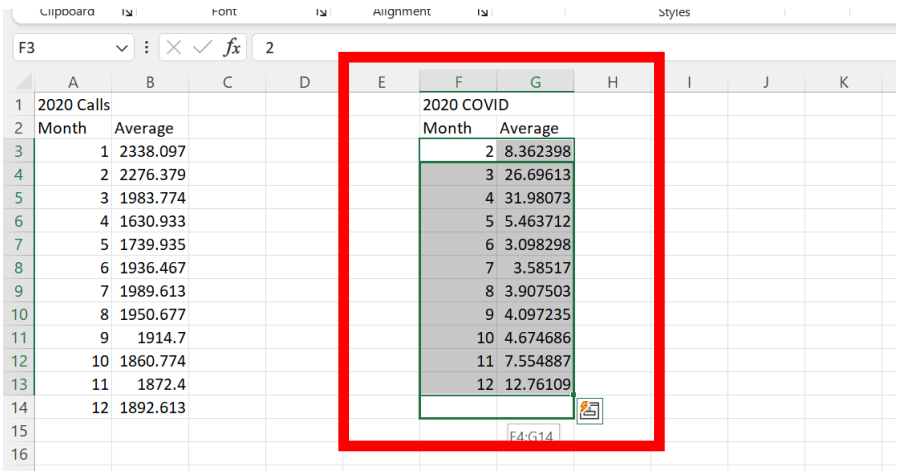


	A	B	C	D	E	F	G	H
1	1	2338.097				2	8.362398	
3	3	1983.774				4	31.98073	
4	4	1630.933				5	5.463712	
5	5	1739.935				6	3.098298	
6	6	1936.467				7	3.58517	
7	7	1989.613				8	3.907503	
8	8	1950.677				9	4.097235	
9	9	1914.7				10	4.674686	
10	10	1860.774				11	7.554887	
11	11	1872.4				12	12.76109	
12	12	1892.613						
13								
14								

12. In the new rows, insert into cell A1 “2020 Calls”, cell A2 “Month”, cell B2 “Average”, cell F1 “2020 COVID”, cell F2 “Month”, and cell G2 “Average”.

	A	B	C	D	E	F	G	H	I	J	K
1	2020 Calls					2020 COVID					
2	Month	Average				Month	Average				
3	1	2338.097				2	8.362398				
4	2	2276.379				3	26.69613				
5	3	1983.774				4	31.98073				
6	4	1630.933				5	5.463712				
7	5	1739.935				6	3.098298				
8	6	1936.467				7	3.58517				
9	7	1989.613				8	3.907503				
10	8	1950.677				9	4.097235				
11	9	1914.7				10	4.674686				
12	10	1860.774				11	7.554887				
13	11	1872.4				12	12.76109				
14	12	1892.613									
15											
16											

13. Then, select cells F3:H13. Hover over the edge of the selected cells until you see the "move" icon. Drag the cells down 1 row to cell F4.

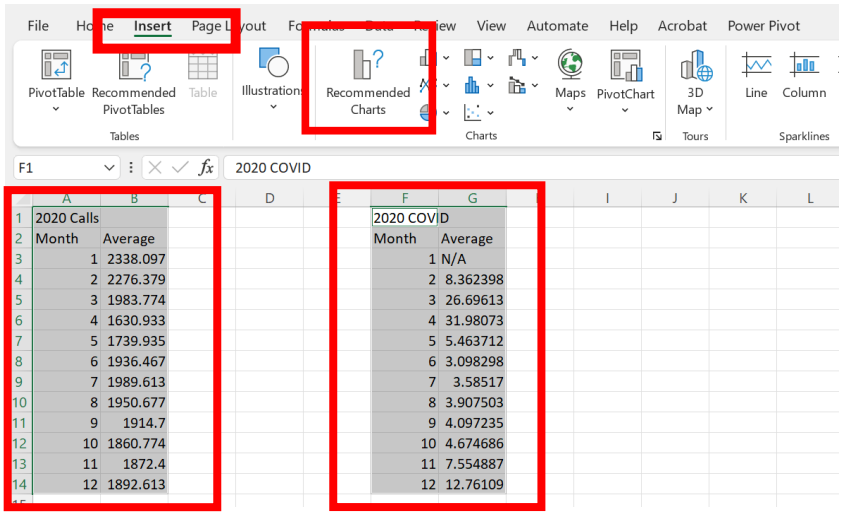


	A	B	C	D	E	F	G	H
1	2020 Calls					2020 COVID		
2	Month	Average				Month	Average	
3	1	2338.097						
4	2	2276.379				2	8.362398	
5	3	1983.774				3	26.69613	
6	4	1630.933				4	31.98073	
7	5	1739.935				5	5.463712	
8	6	1936.467				6	3.098298	
9	7	1989.613				7	3.58517	
10	8	1950.677				8	3.907503	
11	9	1914.7				9	4.097235	
12	10	1860.774				10	4.674686	
13	11	1872.4				11	7.554887	
14	12	1892.613				12	12.76109	

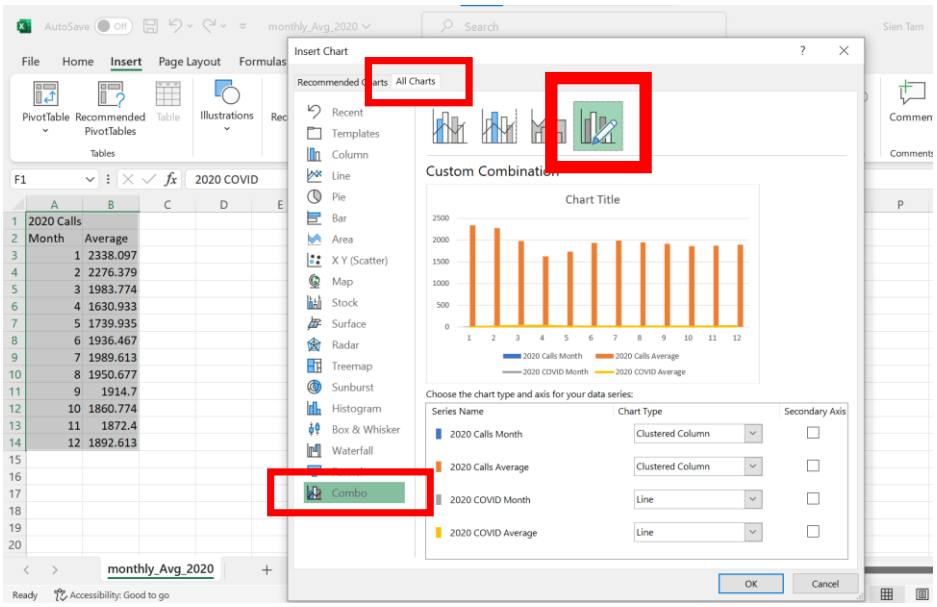
14. In cell F3 enter "1" and in cell G3 enter "N/A". We do this because the COVID-19 dataset does not cover the month of January 2020.

	A	B	C	D	E	F	G	H	I
1	2020 Calls					2020 COVID			
2	Month	Average				Month	Average		
3	1	2338.097				1	N/A		
4	2	2276.379				2	8.362398		
5	3	1983.774				3	26.69613		
6	4	1630.933				4	31.98073		
7	5	1739.935				5	5.463712		
8	6	1936.467				6	3.098298		
9	7	1989.613				7	3.58517		
10	8	1950.677				8	3.907503		
11	9	1914.7				9	4.097235		
12	10	1860.774				10	4.674686		
13	11	1872.4				11	7.554887		
14	12	1892.613				12	12.76109		
15									

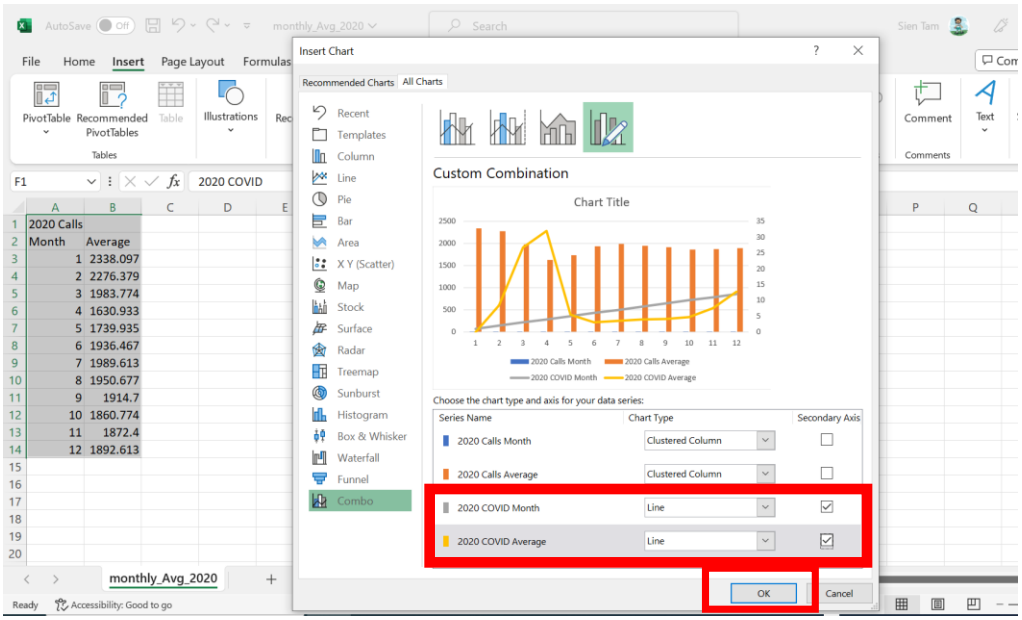
15. Next, select both datasets. In the ribbon, select the Insert tab, then Recommended Charts.



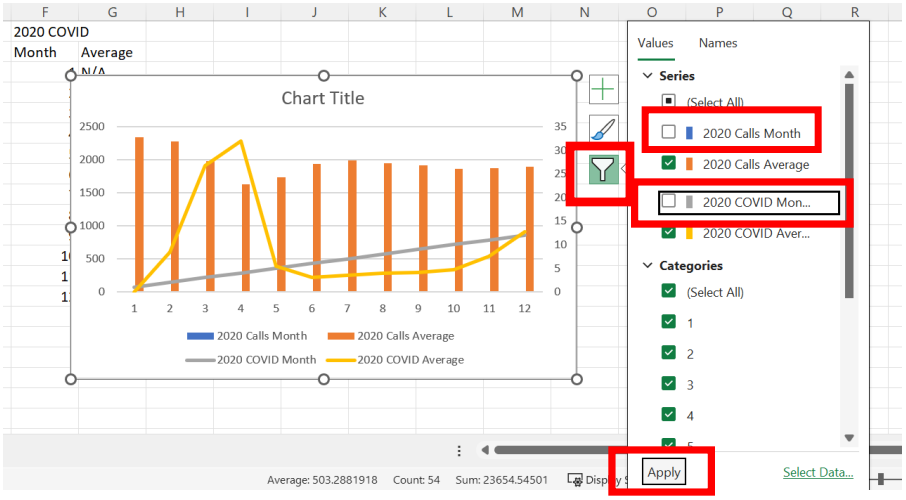
16. Then, in the Recommended Charts pop-up, select the All Charts tab, then Combo, and then Custom Combo.



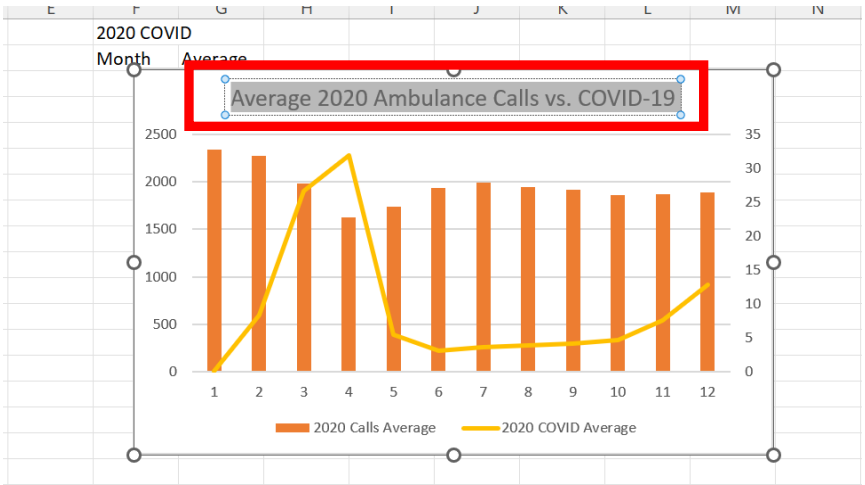
17. Next, toggle Secondary Axis for the Series Name “2020 COVID Month” and “2020 COVID Average” with the Chart Type “Line”. Press OK.



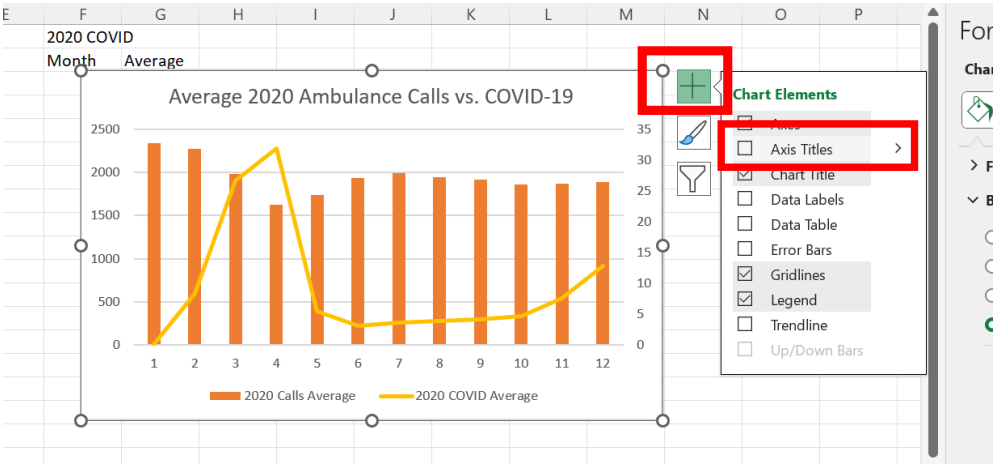
18. Next, click on the Chart Filters icon. Then, deselect “2020 Calls Month” and “2020 COVID Month” from the **Series** category. Then press Apply.

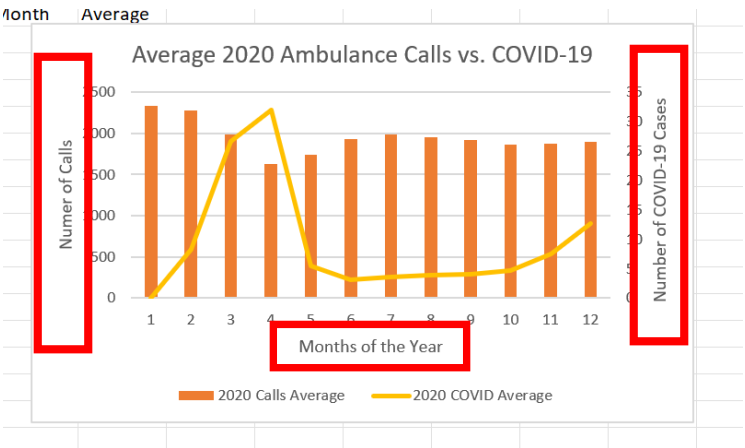


19. Then, change the Chart Title to “Average 2020 Ambulance Calls vs. COVID-19”.

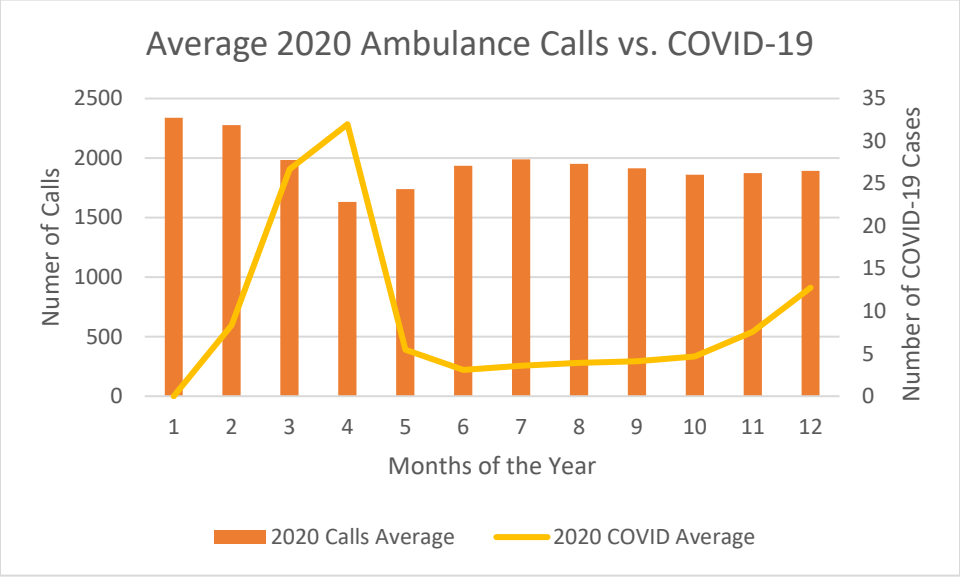


20. Click on the Chart Elements icon and select Axis Titles. On the left y-axis, name it “Number of Calls.” On the right y-axis, name it “Number of COVID-19 Cases.” On the x-axis, name it “Months of the Year.”





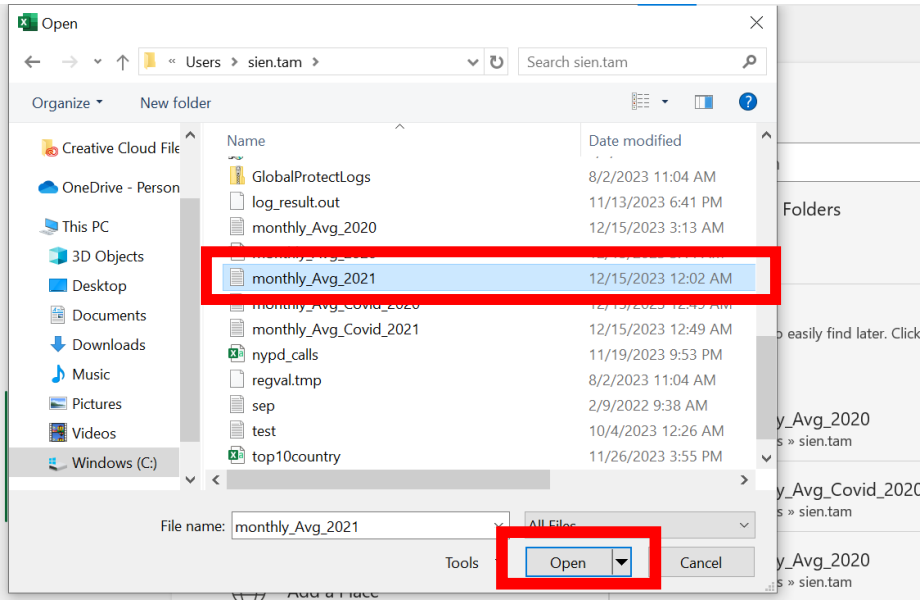
You should get the following result:



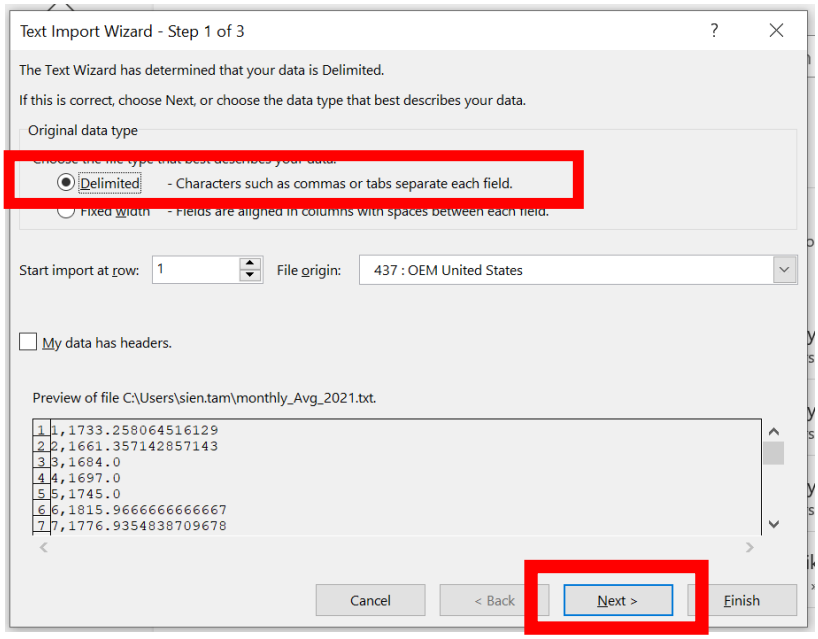
Step 10: Create Graphs to Compare the Monthly Total Averages in 2021 Using Excel

This step is to use Excel to generate graphs of the monthly total averages of 2021 and compare them. The data, which was exported separately, will all need to be copied and pasted into one Excel file (monthly_Avg_Covid_2021 → monthly_Avg_2021, etc.) and saved as an Excel workbook in order to have the data together for the creation of the graphs. Follow the directions carefully.

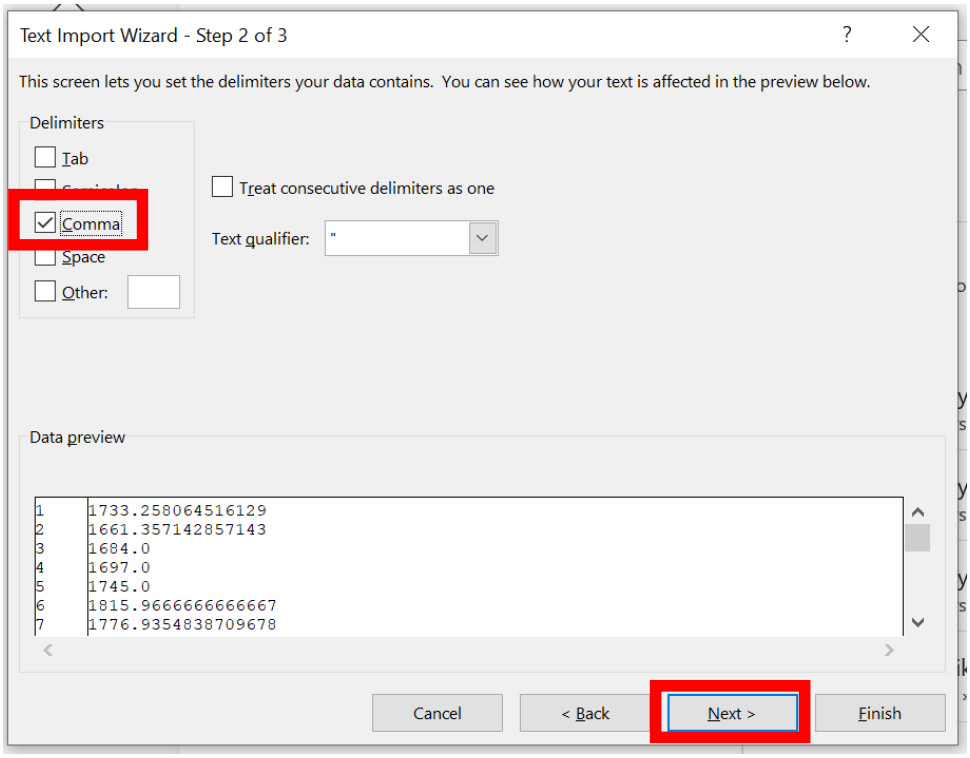
1. Open the file, **monthly_Avg_2021.txt**, in Excel. You have to open Excel first.



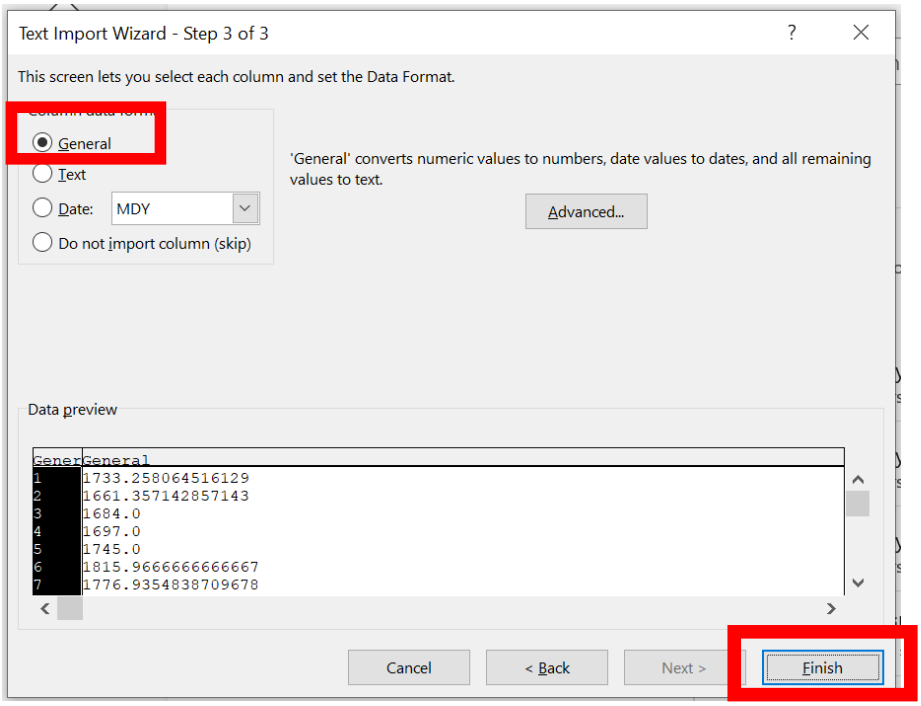
2. Then, in the first page of the Text Import Wizard, make sure “Delimited” is selected. Then press “Next.”



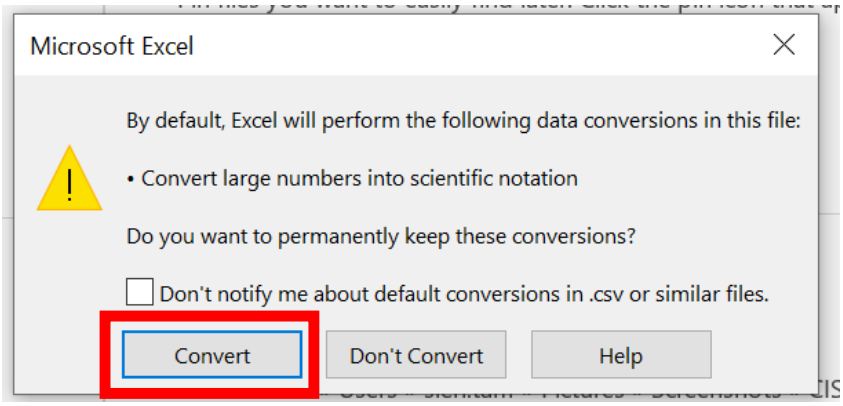
3. Then, in the second page of the Text Import Wizard, make sure “Comma” is selected instead of “tab.” Then press “Next.”



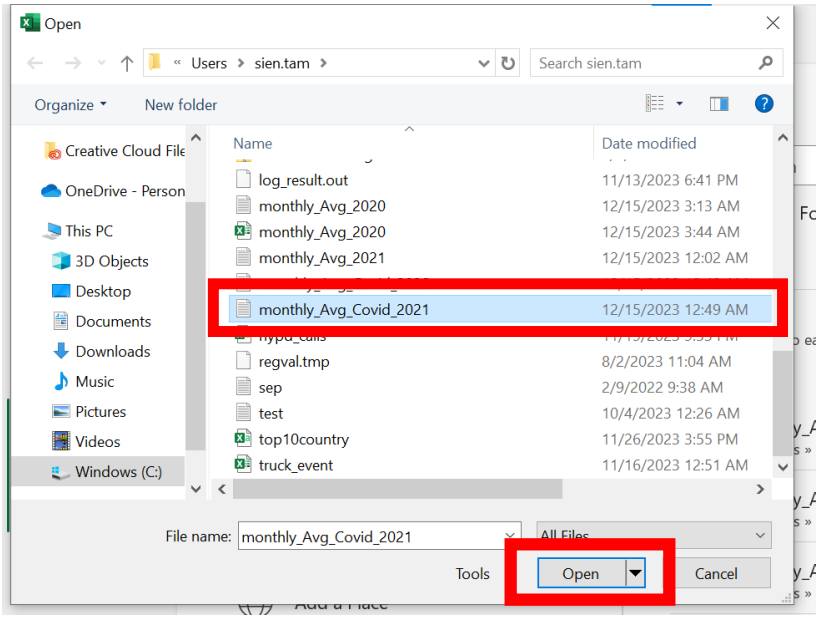
4. Then, in the third page of the Text Import Wizard, make sure “General” is selected for all columns. Then press “Finish.”



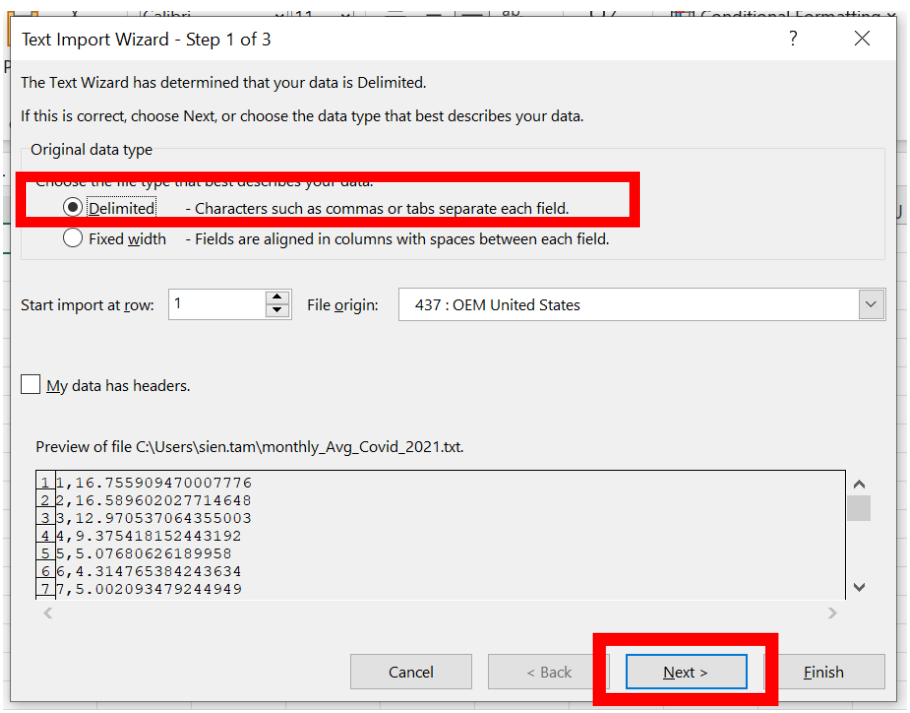
Note: If a pop-up appears asking you to process data conversions, press **Convert**. This conversion will make sure numbers stay numbers and deal with the decimals.



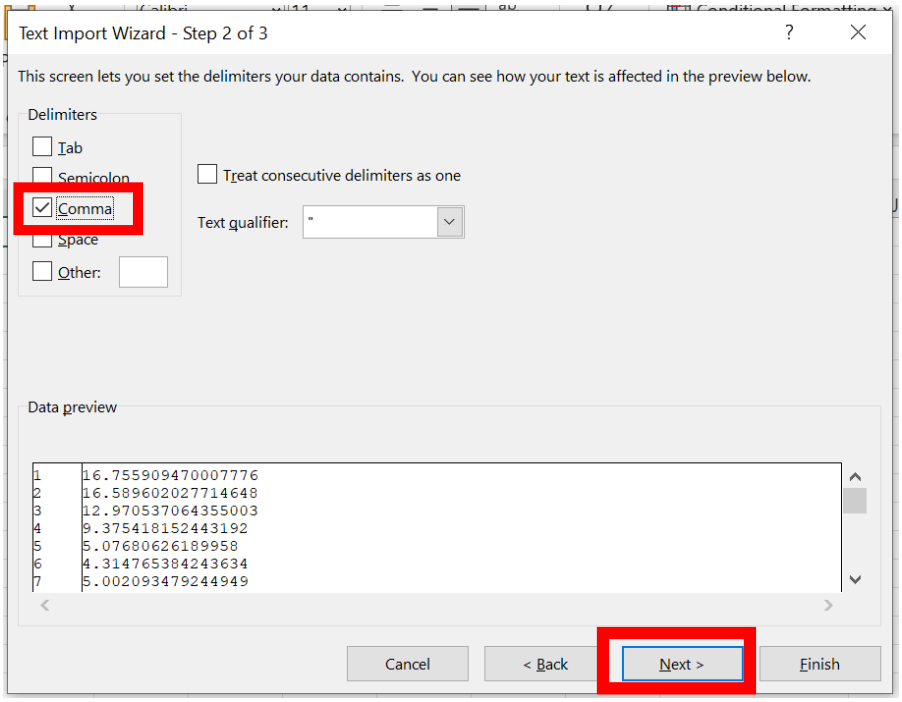
5. Open the file, **monthly_Avg_Covid_2021.txt**, in Excel. You have to open Excel first.



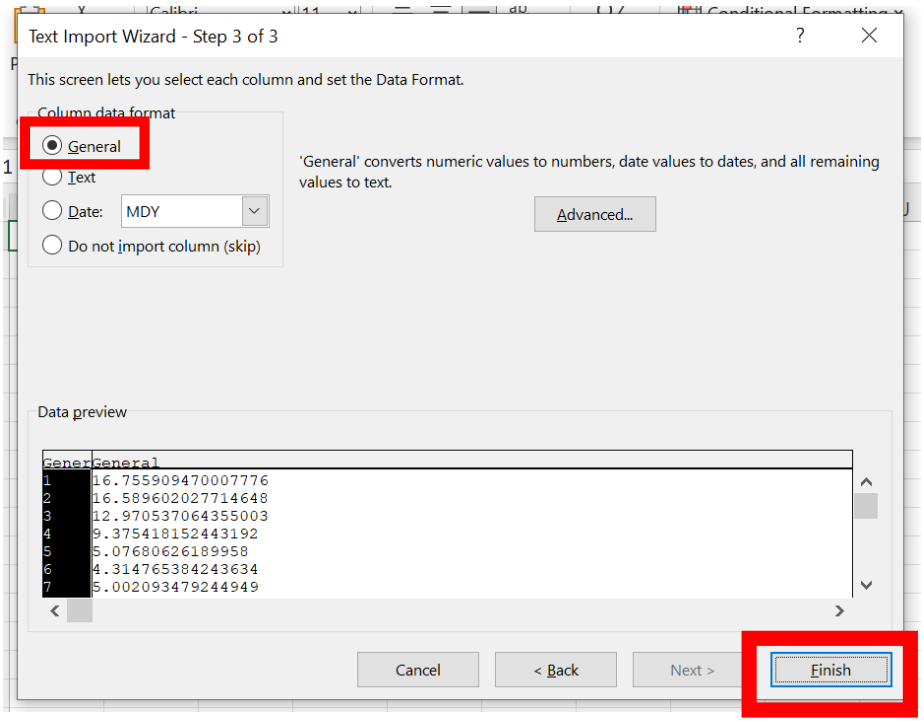
6. Then, in the first page of the Text Import Wizard, make sure “Delimited” is selected. Then press “Next.”



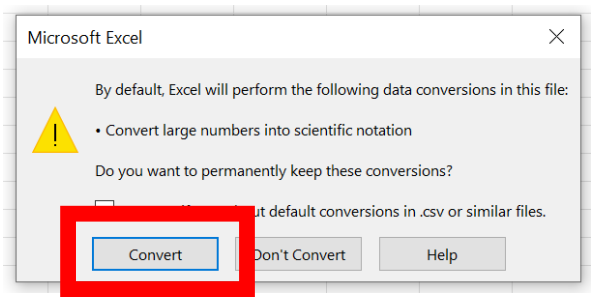
7. Then, in the second page of the Text Import Wizard, make sure “Comma” is selected instead of “tab.” Then press “Next.”



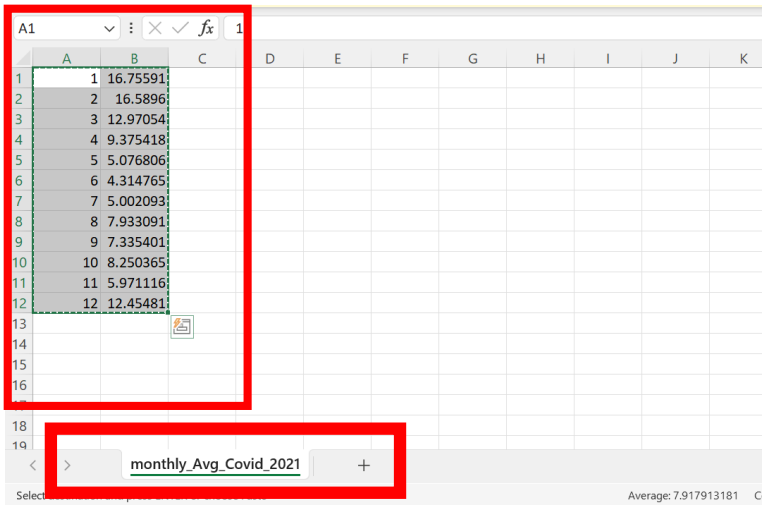
8. Then, in the third page of the Text Import Wizard, make sure "General" is selected for all columns. Then press "Finish."

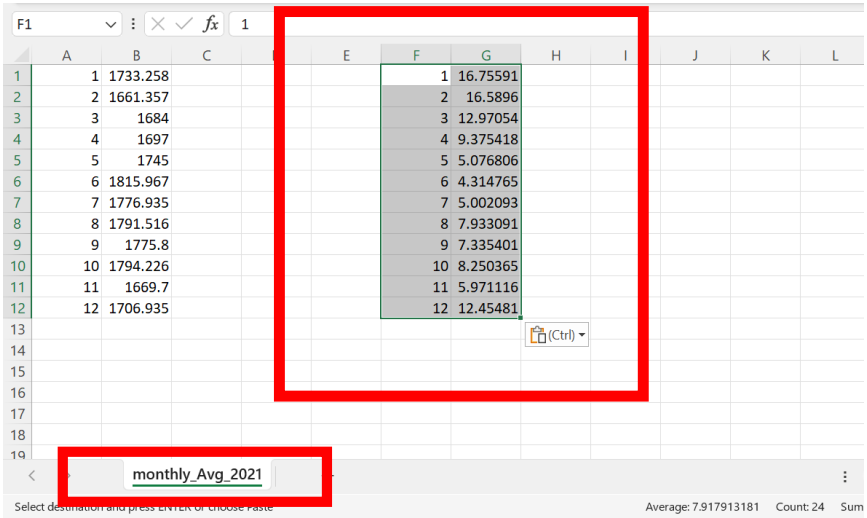


Note: If a pop-up appears asking you to process data conversions, press **Convert**. This conversion will make sure numbers stay numbers and deal with the decimals.

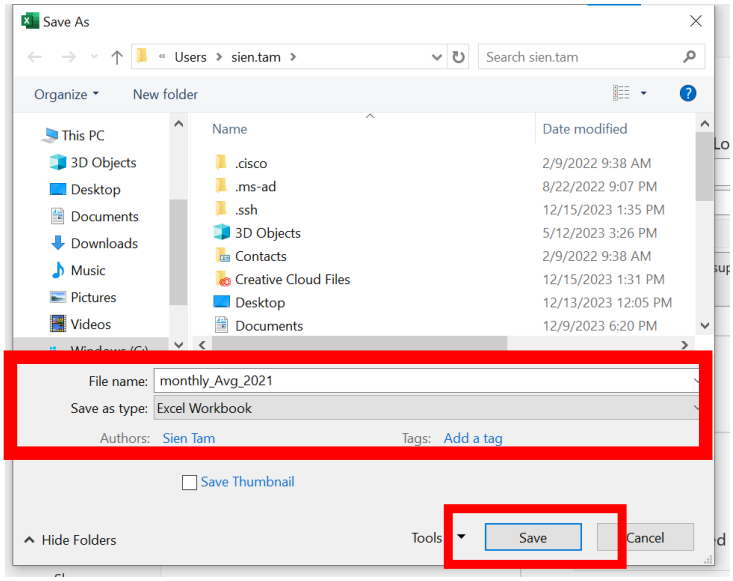


9. Then, select the data in the **monthly_Avg_Covid_2021** file, copy it (Ctrl + C), and paste it in the **monthly_Avg_2021** file in cell F1.

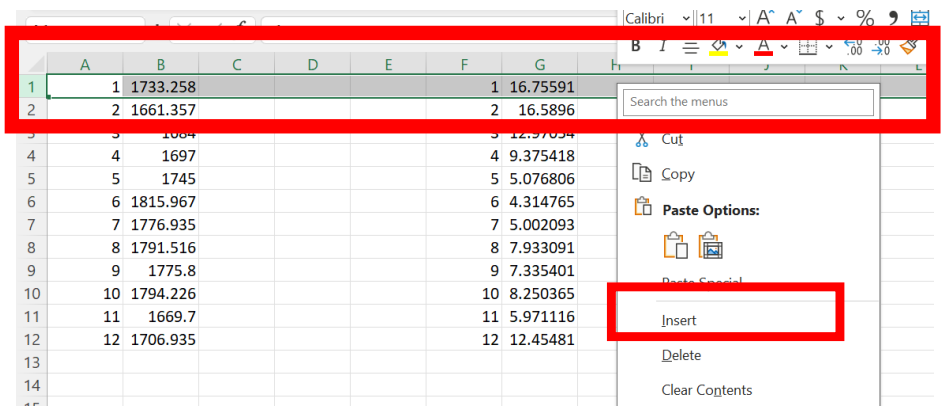




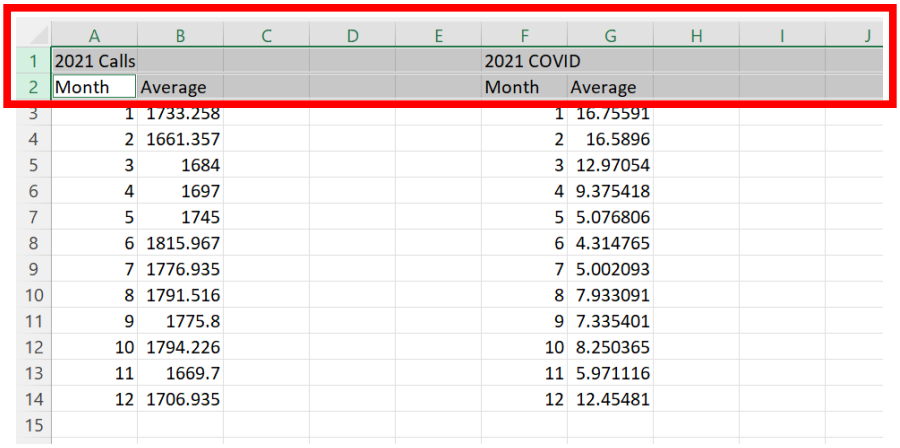
10. You may close the **monthly_Avg_Covid_2021** file. Save the current file "monthly_Avg_2021" as the same name as an **Excel Workbook (.xlsx)**:



11. Then, insert **two** rows above the current row 1.

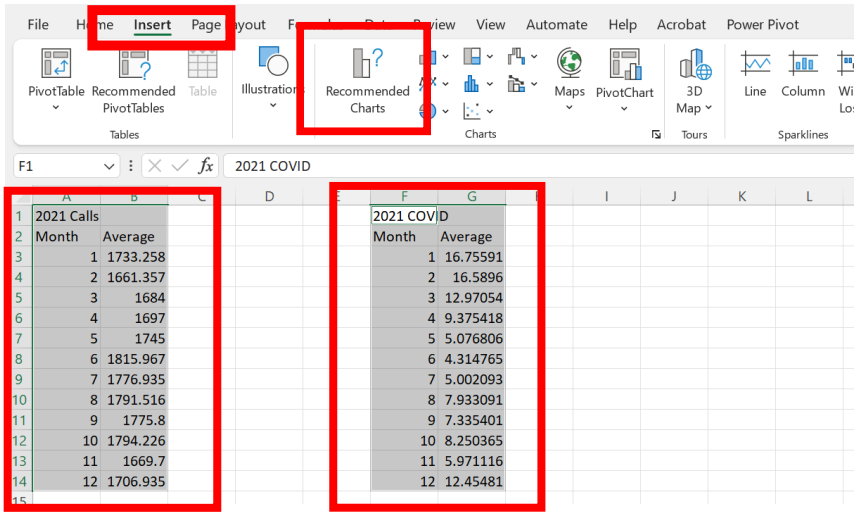


12. In the new rows, insert into cell A1 “2021 Calls”, cell A2 “Month”, cell B2 “Average”, cell F1 “2021 COVID”, cell F2 “Month”, and cell G2 “Average”.

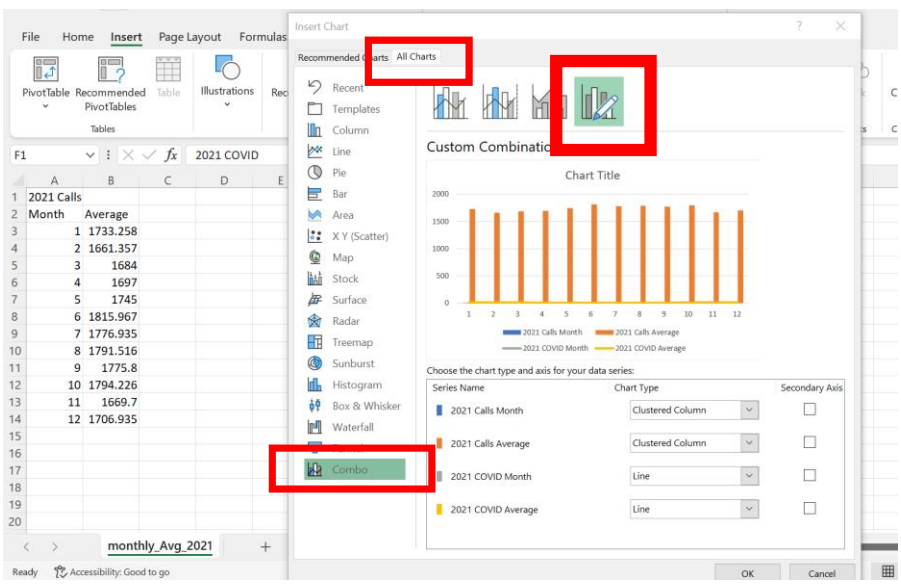


	A	B	C	D	E	F	G	H	I	J
1	2021 Calls					2021 COVID				
2	Month	Average				Month	Average			
3	1	1733.258				1	16.75591			
4	2	1661.357				2	16.5896			
5	3	1684				3	12.97054			
6	4	1697				4	9.375418			
7	5	1745				5	5.076806			
8	6	1815.967				6	4.314765			
9	7	1776.935				7	5.002093			
10	8	1791.516				8	7.933091			
11	9	1775.8				9	7.335401			
12	10	1794.226				10	8.250365			
13	11	1669.7				11	5.971116			
14	12	1706.935				12	12.45481			
15										

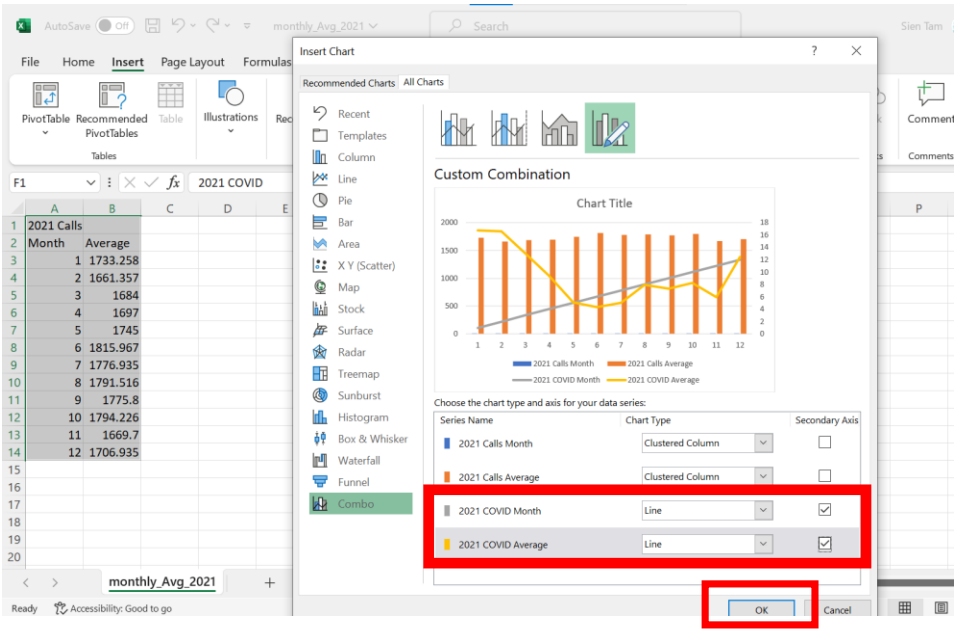
13. Next, select both datasets. In the ribbon, select the Insert tab, then Recommended Charts.



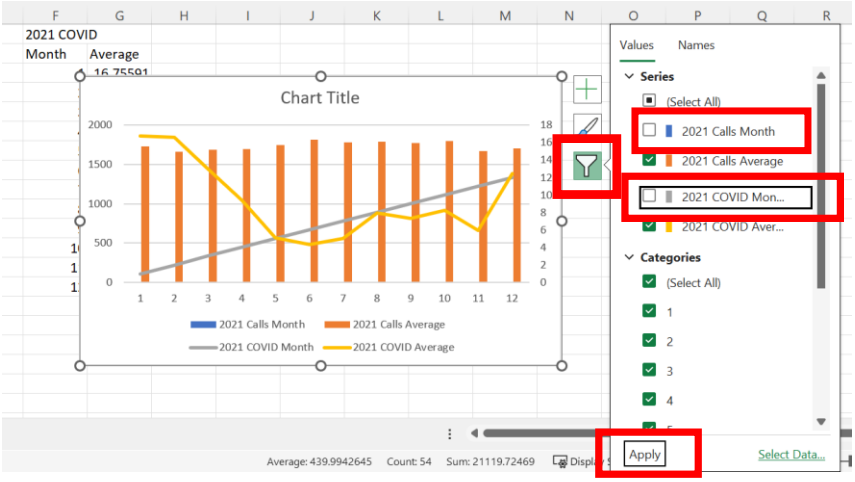
14. Then, in the Recommended Charts pop-up, select the All Charts tab, then Combo, and then Custom Combo.



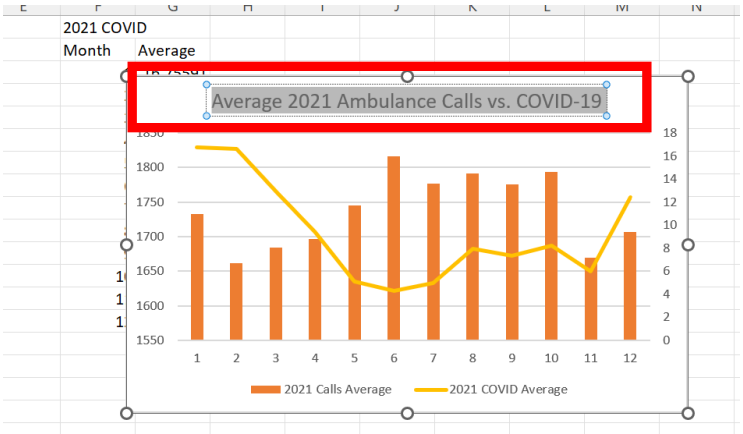
15. Next, toggle Secondary Axis for the Series Name “2021 COVID Month” and “2021 COVID Average” with the Chart Type “Line”. Press OK.



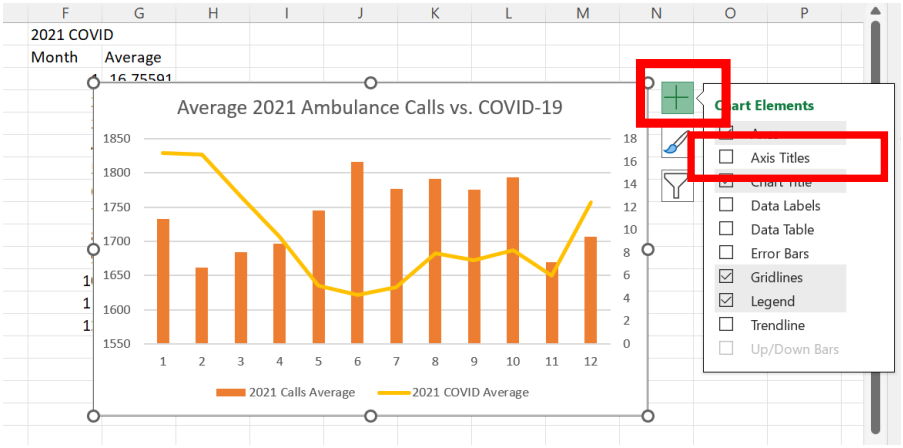
16. Next, click on the Chart Filters icon. Then, deselect “2021 Calls Month” and “2021 COVID Month” from the **Series** category. Then press Apply.

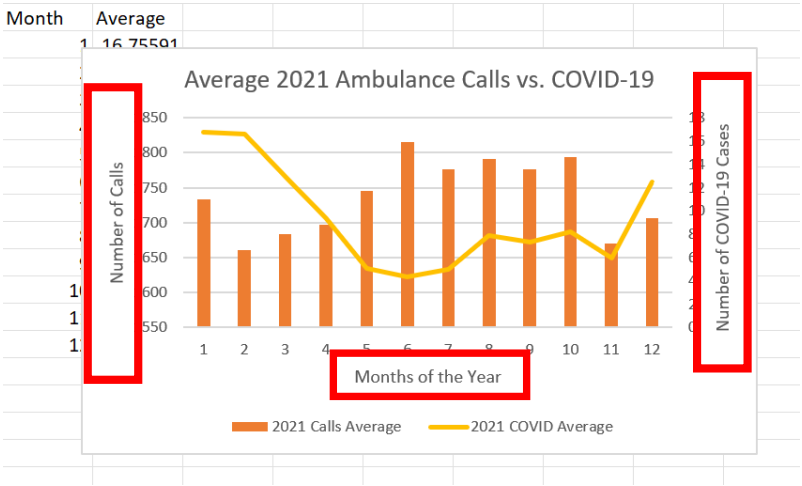


17. Then, change the Chart Title to “Average 2021 Ambulance Calls vs. COVID-19”.

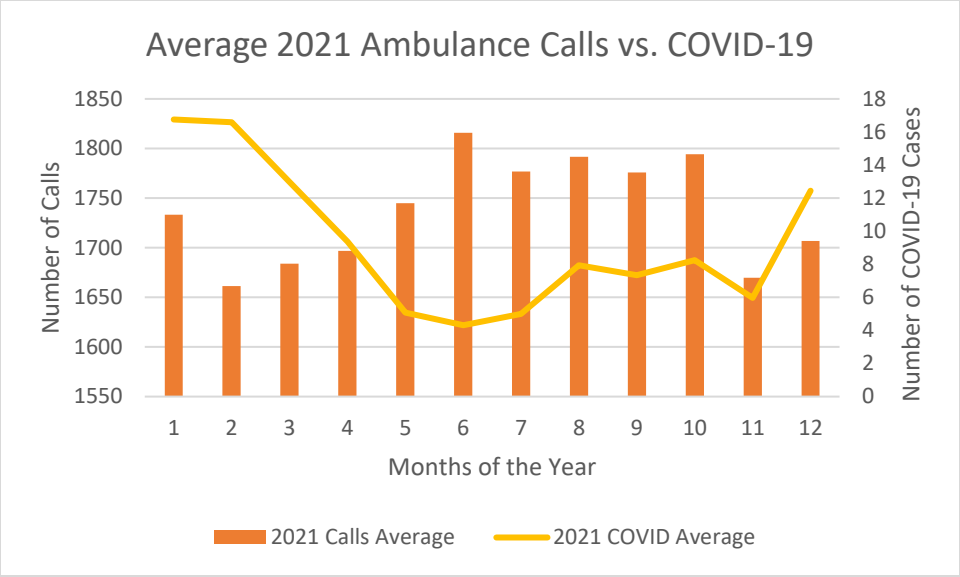


18. Click on the Chart Elements icon and select Axis Titles. On the left y-axis, name it “Number of Calls.” On the right y-axis, name it “Number of COVID-19 Cases.” On the x-axis, name it “Months of the Year.”





You should get the following result:



References

1. **NYPD-Calls-for-Service-Historic Dataset:** https://data.cityofnewyork.us/Public-Safety/NYPD-Calls-for-Service-Historic-/d6zx-ckhd
2. **Covid-like-Illness-by-Boro Dataset:** https://github.com/nychealth/coronavirus-data/blob/master/trends/covid-like-illness-by-boro.csv
3. **GitHub Source Code:** https://github.com/mvan10/4880FinalProject/blob/main/Project_Code.txt
4. **Shell Command List:** https://www.tutorialspoint.com/unix_commands/index.htm
5. **How to Use Excel 3D Map:** https://www.youtube.com/watch?v=P--qP4mfxEg
6. **How to Open Text File in Excel and Get Through Text Import Wizard:** https://youtu.be/t_UIh2081Tk?si=jXbl5crwj2z80a_x&t=519
7. **Hive Timestamp Conversion with "AM/PM":** https://cloudera.ericlin.me/2020/01/hive-impala-timestamp-conversion-with-am-pm/
8. **Hive Date and Timestamp Functions:** https://sparkbyexamples.com/apache-hive/hive-date-and-timestamp-functions-examples/

9. **Merge Several Files in Hadoop Using "-getmerge" Command:** https://www.geeksforgeeks.org/hadoop-getmerge-command/
10. **Use "scp" Command in Linux to Copy a File from Local Machine to Remote Machine/Copy a File from a Remote Machine to Local Machine:** https://www.geeksforgeeks.org/scp-command-in-linux-with-examples/
11. **Hadoop Commands:** http://www.sqlnosql.com/hadoop-important-commands-on-ubuntu-18-04/
12. **Hive Commands:** https://medium.com/nerd-for-tech/hive-commands-b5b6957d8820
13. **Move or Copy Cells:** https://support.microsoft.com/en-us/office/move-or-copy-cells-and-cell-contents-803d65eb-6a3e-4534-8c6f-ff12d1c4139e#:~:text=You%20can%20move%20cells%20in,the%20Cut%20and%20Paste%20commands.&text=Select%20the%20cells%20or%20range,of%20cells%20to%20another%20location.