CIS5200 Term Project Tutorial

Authors: Serena Tang, Sophathriya Sen, Alicia Martinez, Dima Kim
Instructor: Jongwook Woo
Date: 12/14/2024

Lab Tutorial

Serena Tang (xtang13@calstatela.edu), Sophathriya Sen (ssen5@calstatela.edu), Alicia Martinez (amarti761@calstatela.edu), Dima Kim (dkim171@calstatela.edu)

Crime Analysis - Chicago & Los Angeles

Objectives

- Get data from government open source
- Clean data using Python
- Create Hadoop cluster
- Visualization using Tableau and ArcGIS

Platform Specification

Cluster Version	Oracle Big Data Service
Number of Nodes	5
CPU Speed	2.5 GHz
Memory Size	30 GB
HDFS Capacity	1 TB
Hadoop Version	3.1.2
Hive Version	3.1.2

Step 1: Download and prepare Los Angeles and Chicago map data for visualization in ARCGIS

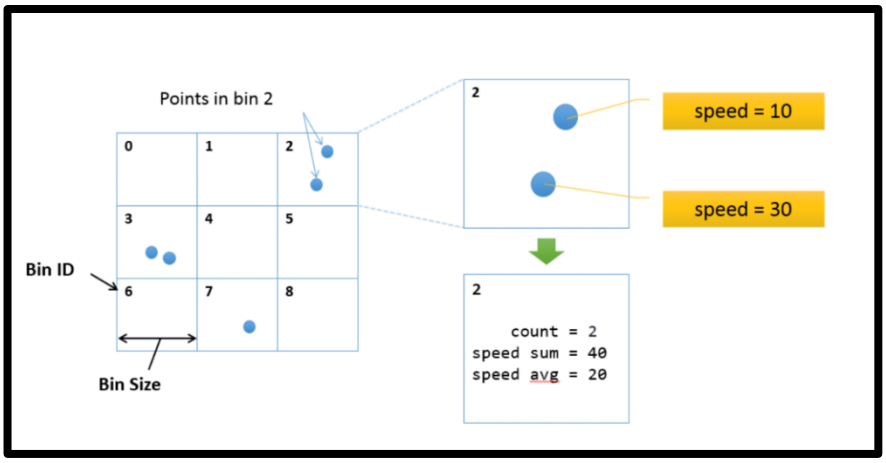
Download Los Angeles crime data set from 2020 to present from:

https://catalog.data.gov/dataset/crime-data-from-2020-to-present

Download Chicago crime data set from 2001 to present from:

https://catalog.data.gov/dataset/crimes-2001-to-present

Prepare Los Angeles and Chicago Crime data for spatial aggregation (sometimes called spatial binning). Spatial aggregation is extremely useful in summarizing big data to gain a meaningful snapshot of patterns in your data. Spatial aggregation works by creating square bins of a user specified size, like this:



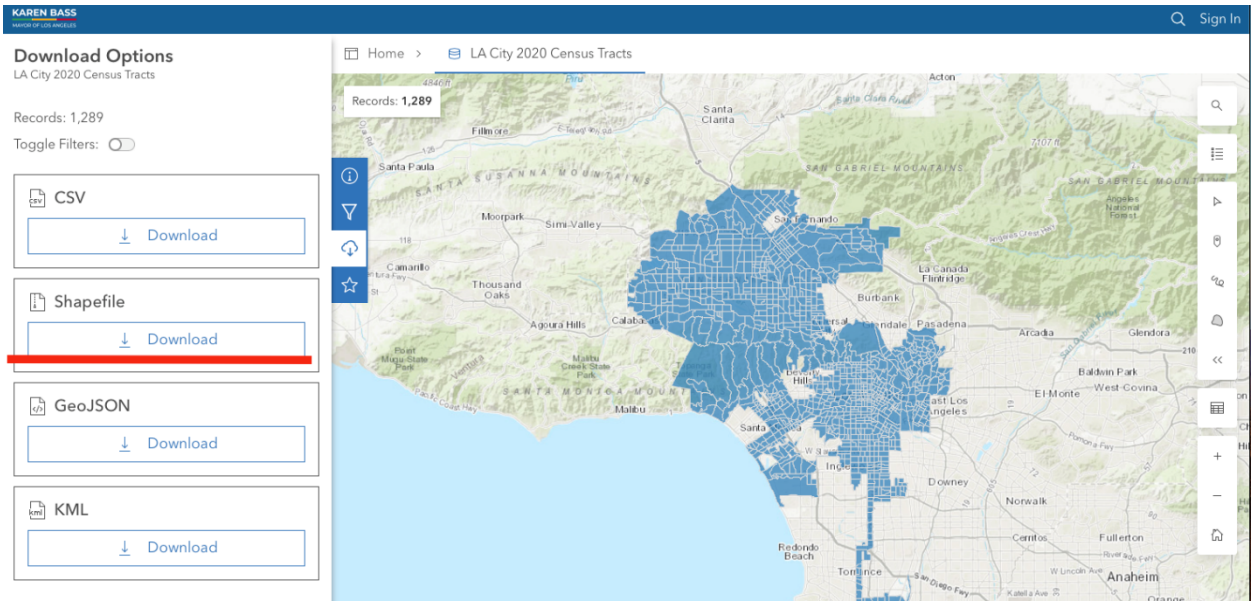
For the aggregation we need to download shapefiles that contain square bins for Los Angeles and Chicago.

Chicago:

https://data.cityofchicago.org/api/geospatial/5jrd-6zik?method=export&format=Shapefile

Los Angeles:

https://geohub.lacity.org/datasets/la-city-2020-census-tracts-/explore?location=34.018934%2C-118.412043%2C10.11



Next we are going to use Python to convert latitude and longitude from the original crime data files to square bins and create .csv files with four columns:

- geoid
- primary type
- Count
- Year

Chicago:

Open PyCharm IDE, then open folder where we saved all data and create the corresponding directories:

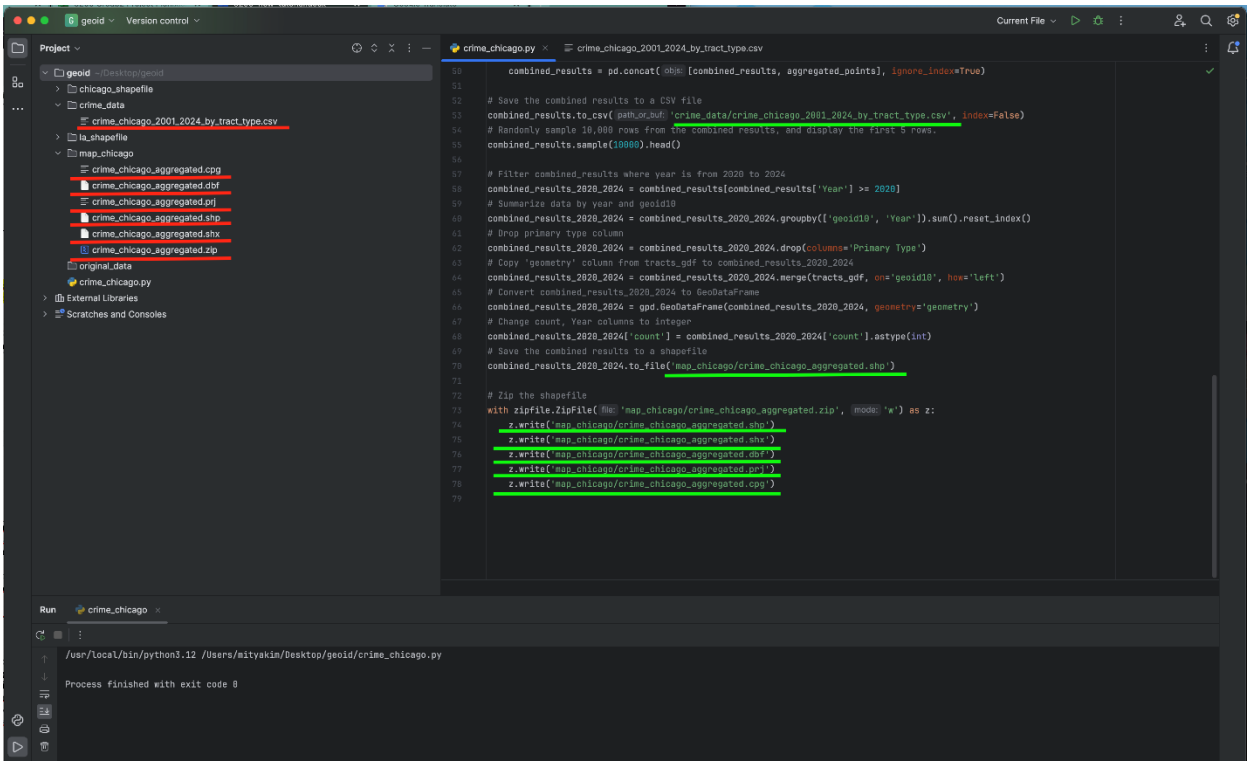
- Create a new Python file crime_chicago.py , paste the code below and execute it:

```
import pandas as pd
import geopandas as gpd
import zipfile

census_tract_shp = 'chicago_shapefile/geo_export_5c337cc7-e5d1-4343-a2ec-2aae0af46f7e.shp'
crime_csv = 'original_data/crime_chicago_2001_2024.csv'
# Generate list of years from 2020 to 2024
```

```
years = list(range(2020, 2025))
# Read in the census tract shapefile
tracts_gdf = gpd.read_file(census_tract_shp)
tracts_gdf = tracts_gdf.to_crs(epsg=4326)
tracts_gdf = tracts_gdf[['geometry', 'geoid10']]

# Read in the crime points data, this is very large dataset
crime_df = pd.read_csv(crime_csv)

# Show first 5 rows of tracts_gdf, you notice the geoid10 is 11 digits
tracts_gdf.head()

# Define function to aggregate crime points counts by census tract, by type of crime
def aggregate_points(points_gdf, geometry_gdf):
    # Spatial join the crime points with the census tracts
    points_gdf = gpd.sjoin(points_gdf, geometry_gdf, how='inner', predicate='within')
    # Group by census tract and type of crime
    points_gdf = points_gdf.groupby(['geoid10', 'Primary Type']).size().reset_index(name='count')
    return points_gdf

# Create an empty dataframe to store the combined results
combined_results = pd.DataFrame()

for year in years:
    # Filter the crime data for the current year
    crime_df_year = crime_df[crime_df['Year'] == year]

    # Convert the filtered crime data to a GeoDataFrame
    crime_gdf_year = gpd.GeoDataFrame(crime_df_year,
                                      geometry=gpd.points_from_xy(crime_df_year.Longitude, crime_df_year.Latitude))
    crime_gdf_year = crime_gdf_year.set_crs(epsg=4326)

    # Aggregate the crime points by census tract
    aggregated_points = aggregate_points(crime_gdf_year, tracts_gdf)

    # Add the year to the aggregated points dataframe
    aggregated_points['Year'] = year

    # Append the results to the combined dataframe
    combined_results = pd.concat([combined_results, aggregated_points], ignore_index=True)

# Save the combined results to a CSV file
combined_results.to_csv('crime_data/crime_chicago_2001_2024_by_tract_type.csv', index=False)
# Randomly sample 10,000 rows from the combined results, and display the first 5 rows.
combined_results.sample(10000).head()

# Filter combined_results where year is from 2020 to 2024
combined_results_2020_2024 = combined_results[combined_results['Year'] >= 2020]
# Summarize data by year and geoid10
combined_results_2020_2024 = combined_results_2020_2024.groupby(['geoid10', 'Year']).sum().reset_index()
# Drop primary type column
combined_results_2020_2024 = combined_results_2020_2024.drop(columns='Primary Type')
# Copy 'geometry' column from tracts_gdf to combined_results_2020_2024
```

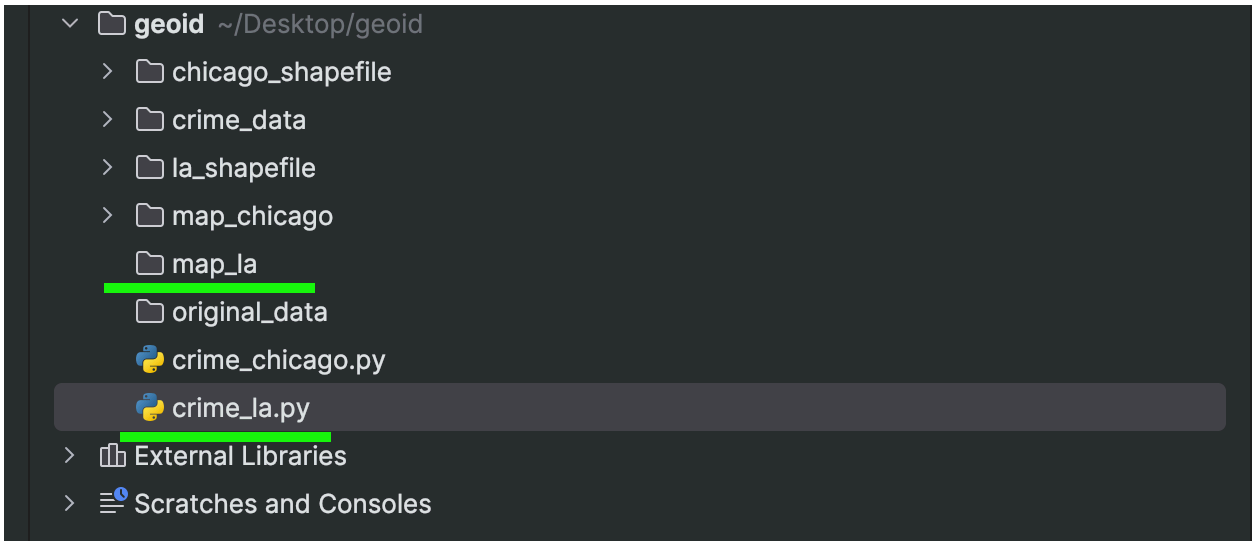
```
combined_results_2020_2024 = combined_results_2020_2024.merge(tracts_gdf, on='geoid10', how='left')
# Convert combined_results_2020_2024 to GeoDataFrame
combined_results_2020_2024 = gpd.GeoDataFrame(combined_results_2020_2024, geometry='geometry')
# Change count, Year columns to integer
combined_results_2020_2024['count'] = combined_results_2020_2024['count'].astype(int)
# Save the combined results to a shapefile
combined_results_2020_2024.to_file('map_chicago/crime_chicago_aggregated.shp')

# Zip the shapefile
with zipfile.ZipFile('map_chicago/crime_chicago_aggregated.zip', 'w') as z:
    z.write('map_chicago/crime_chicago_aggregated.shp')
    z.write('map_chicago/crime_chicago_aggregated.shx')
    z.write('map_chicago/crime_chicago_aggregated.dbf')
    z.write('map_chicago/crime_chicago_aggregated.prj')
    z.write('map_chicago/crime_chicago_aggregated.cpg')
```

- Then check the new created files in the corresponding folders:



Los Angeles:

- Create new folder map_la and then create a new Python file crime_la.py.

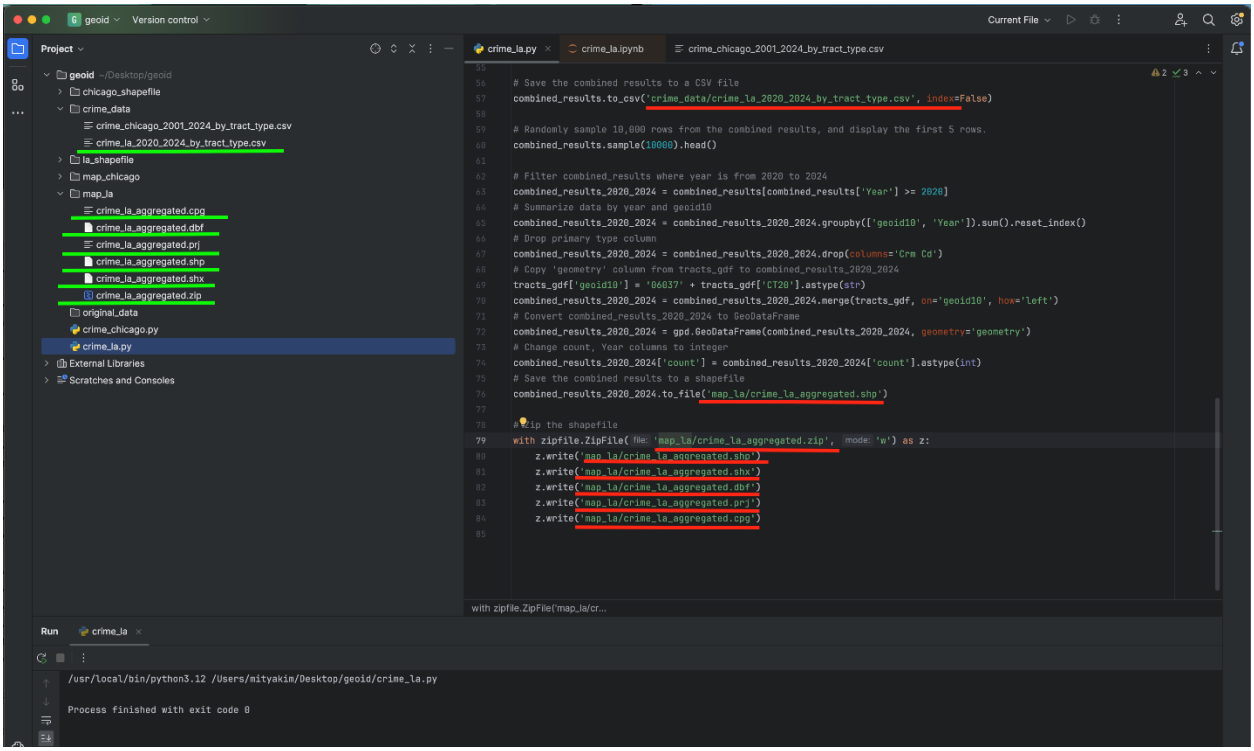


- Paste the code below and execute it:

```
import pandas as pd
import geopandas as gpd
import zipfile

census_tract_shp = 'la_shapefile/LA_City_2020_Census_Tracts_.shp'
crime_csv = 'original_data/crime_la_2020_2024.csv'

# Generate list of years from 2001 to 2024
years = list(range(2020, 2024))
# Read in the census tract shapefile
tracts_gdf = gpd.read_file(census_tract_shp)
tracts_gdf = tracts_gdf.to_crs(epsg=4326)
tracts_gdf = tracts_gdf[['geometry', 'CT20']]

# Read in the crime points data, this is very large dataset
crime_df = pd.read_csv(crime_csv)

# Convert the date to a datetime object
crime_df['Date Rptd'] = pd.to_datetime(crime_df['Date Rptd'], format='%m/%d/%Y %I:%M:%S %p')
crime_df['Year'] = crime_df['Date Rptd'].dt.year

# Define function to aggregate crime points counts by census tract, by type of crime
def aggregate_points(points_gdf, geometry_gdf):
    # Spatial join the points to the census tracts
    points_gdf = gpd.sjoin(points_gdf, geometry_gdf, how='inner', predicate='within')
    # Group by the census tract and
    points_gdf = points_gdf.groupby(['CT20', 'Crm Cd']).size().reset_index(name='count')
    return points_gdf



# Create an empty dataframe to store the combined results
combined_results = pd.DataFrame()

# Loop through each year
```

```
for year in years:
    # Filter the crime data for the current year
    crime_df_year = crime_df[crime_df['Year'] == year]

    # Convert the filtered crime data to a GeoDataFrame
    crime_gdf_year = gpd.GeoDataFrame(crime_df_year, geometry=gpd.points_from_xy(crime_df_year.LON,
crime_df_year.LAT))
    crime_gdf_year = crime_gdf_year.set_crs(epsg=4326)

    # Aggregate the crime points by census tract
    aggregated_points = aggregate_points(crime_gdf_year, tracts_gdf)

    # Add the year to the aggregated points dataframe
    aggregated_points['Year'] = year

    # Append the results to the combined dataframe
    combined_results = pd.concat([combined_results, aggregated_points], ignore_index=True)

# Create a 'geoid10' columns. goeid10 = '06037' + CT20
combined_results['geoid10'] = '06037' + combined_results['CT20'].astype(str)
combined_results = combined_results[['geoid10', 'Crm Cd', 'count', 'Year']]

# Save the combined results to a CSV file
combined_results.to_csv('crime_data/crime_la_2020_2024_by_tract_type.csv', index=False)

# Randomly sample 10,000 rows from the combined results, and display the first 5 rows.
combined_results.sample(10000).head()

# Filter combined_results where year is from 2020 to 2024
combined_results_2020_2024 = combined_results[combined_results['Year'] >= 2020]
# Summarize data by year and geoid10
combined_results_2020_2024 = combined_results_2020_2024.groupby(['geoid10', 'Year']).sum().reset_index()
# Drop primary type column
combined_results_2020_2024 = combined_results_2020_2024.drop(columns='Crm Cd')
# Copy 'geometry' column from tracts_gdf to combined_results_2020_2024
tracts_gdf['geoid10'] = '06037' + tracts_gdf['CT20'].astype(str)
combined_results_2020_2024 = combined_results_2020_2024.merge(tracts_gdf, on='geoid10', how='left')
# Convert combined_results_2020_2024 to GeoDataFrame
combined_results_2020_2024 = gpd.GeoDataFrame(combined_results_2020_2024, geometry='geometry')
# Change count, Year columns to integer
combined_results_2020_2024['count'] = combined_results_2020_2024['count'].astype(int)
# Save the combined results to a shapefile
combined_results_2020_2024.to_file('map_la/crime_la_aggregated.shp')

# Zip the shapefile
with zipfile.ZipFile('map_la/crime_la_aggregated.zip', 'w') as z:
    z.write('map_la/crime_la_aggregated.shp')
    z.write('map_la/crime_la_aggregated.shx')
    z.write('map_la/crime_la_aggregated.dbf')
    z.write('map_la/crime_la_aggregated.prj')
    z.write('map_la/crime_la_aggregated.cpg')
```

- Then check the new created files in the corresponding folders:



Step 2: Crime Data Loaded into Hadoop Clusters

Then follow the steps:

- create a new folder on your PC
- put all files in a one Crime.zip
  - crime_chicago_2001_2024_by_tract_type.csv (result file from the step 1)
  - crime_la_2020_2024_by_tract_type.csv (result file from the step 1)
  - crime_la_2020_2024.csv (original data set LA)
  - crime_chicago_2001_2024.csv (original data set Chicago)
- open terminal and using your path and username enter command below and then enter your password:

scp "/Users/mityakim/Desktop/Crime.zip" dkim171@129.146.230.230:/home/dkim171

- enter to the Linux server:

ssh dkim171@129.146.230.230

- enter command:

unzip Crime.zip

- You may check out all directories exist, and its files are created:

```
-bash-4.2$ ls
```

- Create crime folders and sub folders

```
-bash-4.2$ hdfs dfs -mkdir Project/crime
-bash-4.2$ hdfs dfs -mkdir Project/crime/chicago
-bash-4.2$ hdfs dfs -mkdir Project/crime/la
-bash-4.2$ hdfs dfs -mkdir Project/crime/chicago_tractid
-bash-4.2$ hdfs dfs -mkdir Project/crime/la_tractid
```

- Place the files in Hadoop:

```
-bash-4.2$ hdfs dfs -put crime_chicago_2001_2024.csv Project/crime/chicago
-bash-4.2$ hdfs dfs -put crime_la_2020_2024.csv Project/crime/la
-bash-4.2$ hdfs dfs -put crime_chicago_2001_2024_by_tract_type.csv
Project/crime/chicago_tractid
-bash-4.2$ hdfs dfs -put crime_la_2020_2024_by_tract_type.csv
Project/crime/la_tractid
```

- to check files, enter:

```
-bash-4.2$ hdfs dfs -ls Project/crime/chicago_tractid
-bash-4.2$ hdfs dfs -ls Project/crime/la_tractid
-bash-4.2$ hdfs dfs -ls Project/crime/chicago
-bash-4.2$ hdfs dfs -ls Project/crime/la
```

Step 3: Creating Hive tables to Query Crime Data

Open Hive CLI (Command Line Shell Interface):

```
-bash-4.2$ beeline
```

Make sure that you are using your database:

use <username>

```
0: jdbc:hive2://bigdaiun0.sub03291929060.trai> use <username>
```

Los Angeles Crime Tractid Table

```
DROP TABLE IF EXISTS original_la_crime;

CREATE EXTERNAL TABLE IF NOT EXISTS original_la_crime(
  DR_NO STRING,
  Date_Rpt STRING,
  DATE_OCC STRING,
  TIME_OCC STRING,
  AREA STRING,
  AREA_NAME STRING,
  RptDist_No STRING,
  Part_1_2 STRING,
  Crm_Cd STRING,
  Crm_Cd_Desc STRING,
  Mocodes STRING,
  Vict_Age STRING,
  Vict_Sex STRING,
  Vict_Descent STRING,
  Premis_Cd STRING,
  Premis_Desc STRING,
  Weapon_Used_Cd STRING,
  Weapon_Desc STRING,
  Status STRING,
  Status_Desc STRING,
  Crm_Cd_1 STRING,
  Crm_Cd_2 STRING,
  Crm_Cd_3 STRING,
  Crm_Cd_4 STRING,
  LOCATION STRING,
  Cross_Street STRING,
  LAT STRING,
  LON STRING
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
  "separatorChar" = ",",
  "quoteChar" = "\"",
  "escapeChar" = "\\"
)
LOCATION '/user/dkim171/Project/crime/la '
```

```
TBLPROPERTIES ('skip.header.line.count'='1');
```

- Check if the table was successfully created:

```
0: jdbc:hive2://bigdaiun0.sub03291929060.trai> show tables;
+-----------------------------+
|            tab_name         |
+-----------------------------+
| la_crime                    |
+-----------------------------+

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from original_la_crime
LIMIT 10;
```

Chicago Crime Table

```
DROP TABLE IF EXISTS original_chicago_crime;
CREATE EXTERNAL TABLE IF NOT EXISTS original_chicago_crime(
   ID STRING,
   Case_Number STRING,
   `Date` STRING,
   Block STRING,
   IUCR STRING,
   Primary_Type STRING,
   Description STRING,
   Location_Description STRING,
   Arrest STRING,
   Domestic STRING,
   Beat STRING,
   District STRING,
   Ward STRING,
   Community_Area STRING,
   FBI_Code STRING,
   X_Coordinate STRING,
   Y_Coordinate STRING,
   Year STRING,
   Updated_On STRING,
   Latitude STRING,
   Longitude STRING,
   Location STRING
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
  "separatorChar" = ",",
  "quoteChar" = "\"",
  "escapeChar" = "\\"
```

```
)
LOCATION '/user/dkim171/Project/crime/chicago'
TBLPROPERTIES ('skip.header.line.count'='1');
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from original_la_crime
LIMIT 10;
```

Los Angeles Crime Tractid Table

```
DROP TABLE IF EXISTS la_tractid;

CREATE EXTERNAL TABLE IF NOT EXISTS la_tractid(
  tract_id STRING,
  primary_type STRING,
  count STRING,
  year STRING
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
  "separatorChar" = ",",
  "quoteChar" = "\"",
  "escapeChar" = "\\"
)
LOCATION '/user/dkim171/Project/crime/la_tractid'
TBLPROPERTIES ('skip.header.line.count'='1');
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from la_tractid LIMIT
10;
```

Chicago Crime Tractid Table

```
DROP TABLE IF EXISTS chicago_tractid;

CREATE EXTERNAL TABLE IF NOT EXISTS chicago_tractid(
  tract_id STRING,
  primary_type STRING,
  count STRING,
  year STRING
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
```

```
  "separatorChar" = ",",
  "quoteChar" = "\"",
  "escapeChar" = "\\"
)
LOCATION '/user/dkim171/Project/crime/chicago_tractid'
TBLPROPERTIES ('skip.header.line.count'='1');
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from chicago_tractid
 LIMIT 10;
```

Step 4: Download and clean topic data

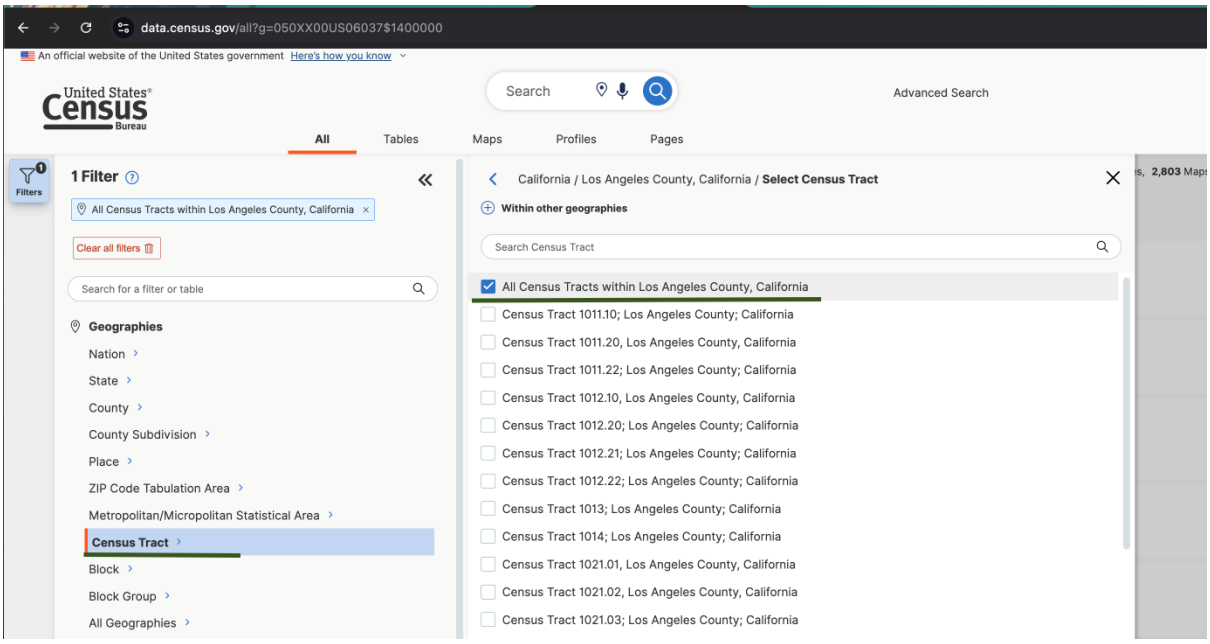
Education Data

- Go to the website:

https://data.census.gov/all

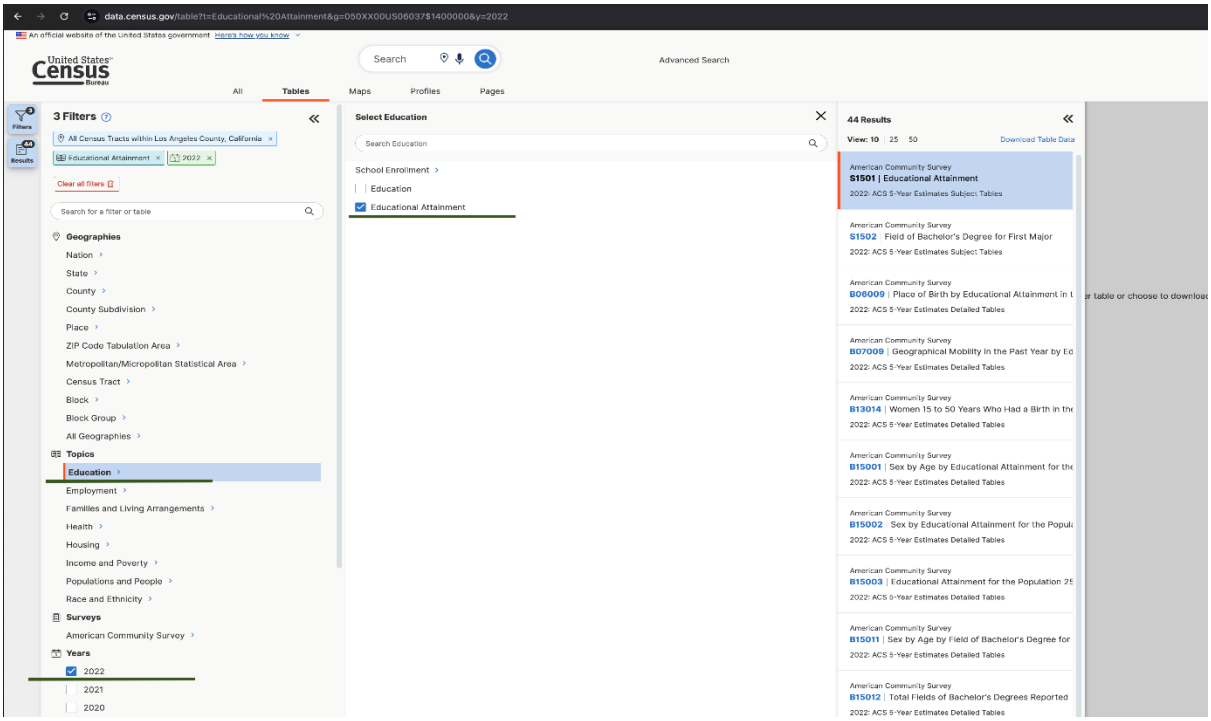
For the analysis purposes we will use:

1. Geographies → Census Tract → California → Los Angeles County → All Census Tracts

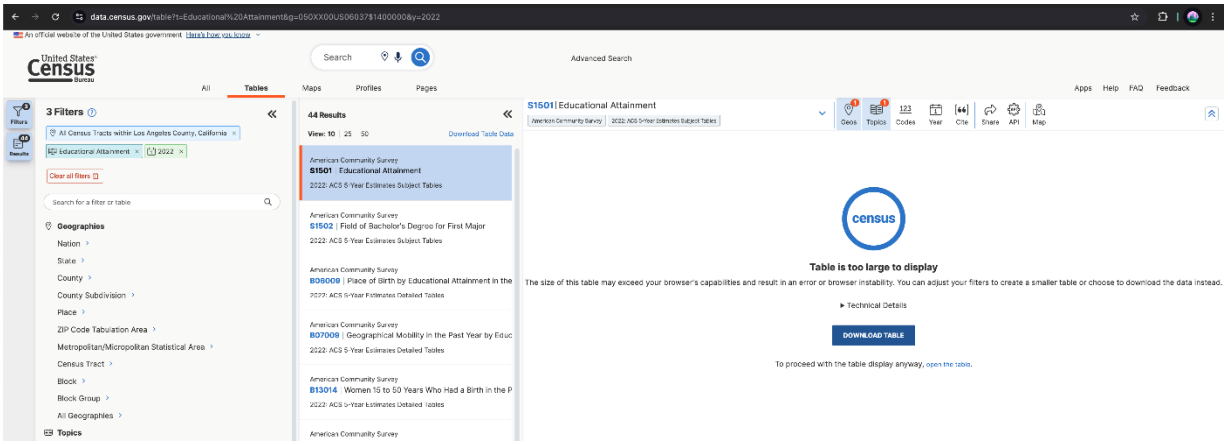


- Topics → Education → Educational Attainment

- Years → 2022



Then we need to open Tables and click Download and repeat for 2020, 2021 years.

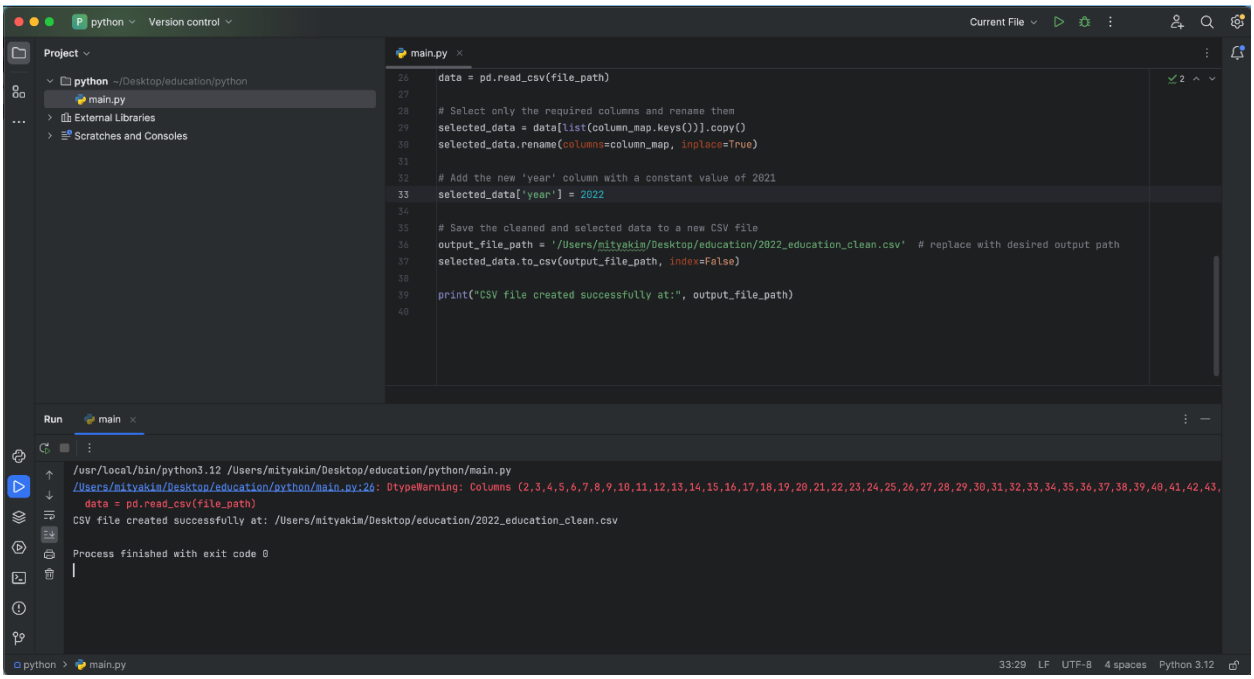


2. After we need to clean our data and get columns that we will use for the future analysis, and we are going to use Python and PyCharm or Visual Studio Code.

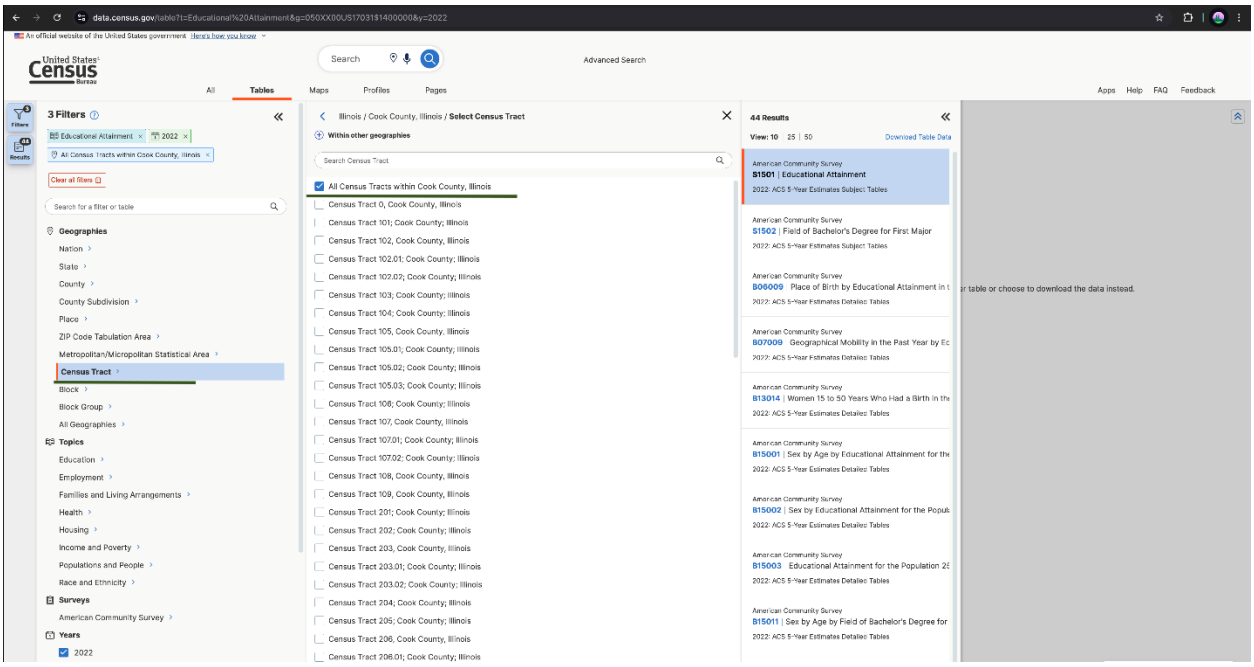
```
import pandas as pd

# Define the mapping of original columns to new column names
```

```
column_map = {
    "GEO_ID": "geo_id",
    "NAME": "area_name",

    "S1501_C01_001E": "population_age_18_24",
    "S1501_C01_003E": "high_school_grad_or_higher_18_24",

    "S1501_C01_016E": "population_25_34_years",
    "S1501_C01_017E": "high_school_grad_or_higher_25_34",

    "S1501_C01_019E": "population_35_44_years",
    "S1501_C01_020E": "high_school_grad_or_higher_35_44",

    "S1501_C01_022E": "population_45_64_years",
    "S1501_C01_023E": "high_school_grad_or_higher_45_64",

    "S1501_C01_025E": "population_age_65_and_over",
    "S1501_C01_026E": "high_school_grad_or_higher_65_and_over"
}

# Load the data from the provided file
file_path = '/Users/mityakim/Desktop/education/2020_education.csv'  # replace with the actual file path
data = pd.read_csv(file_path)

# Select only the required columns and rename them
selected_data = data[list(column_map.keys())].copy()
selected_data.rename(columns=column_map, inplace=True)

# Add the new 'year' column with a constant value of 2020
selected_data['year'] = 2020

# Save the cleaned and selected data to a new CSV file
output_file_path = '/Users/mityakim/Desktop/education/2020_education_clean.csv'  # replace with desired output path
selected_data.to_csv(output_file_path, index=False)

print("CSV file created successfully at:", output_file_path)
```

3. Repeat operation for all 3 files (2020, 2021, 2022).



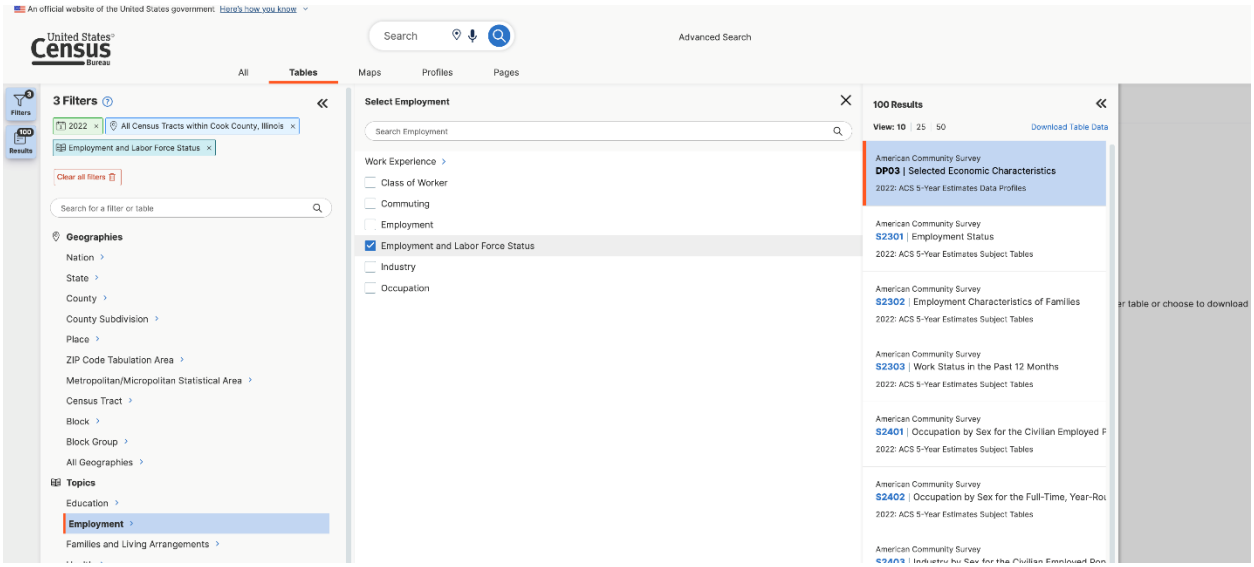
- Then change filter to Chicago



Download 2020, 2021, 2022 years and repeat steps for 2 and 3.

Employment Data

- Topics → Employment → Employment and Labor Force Status

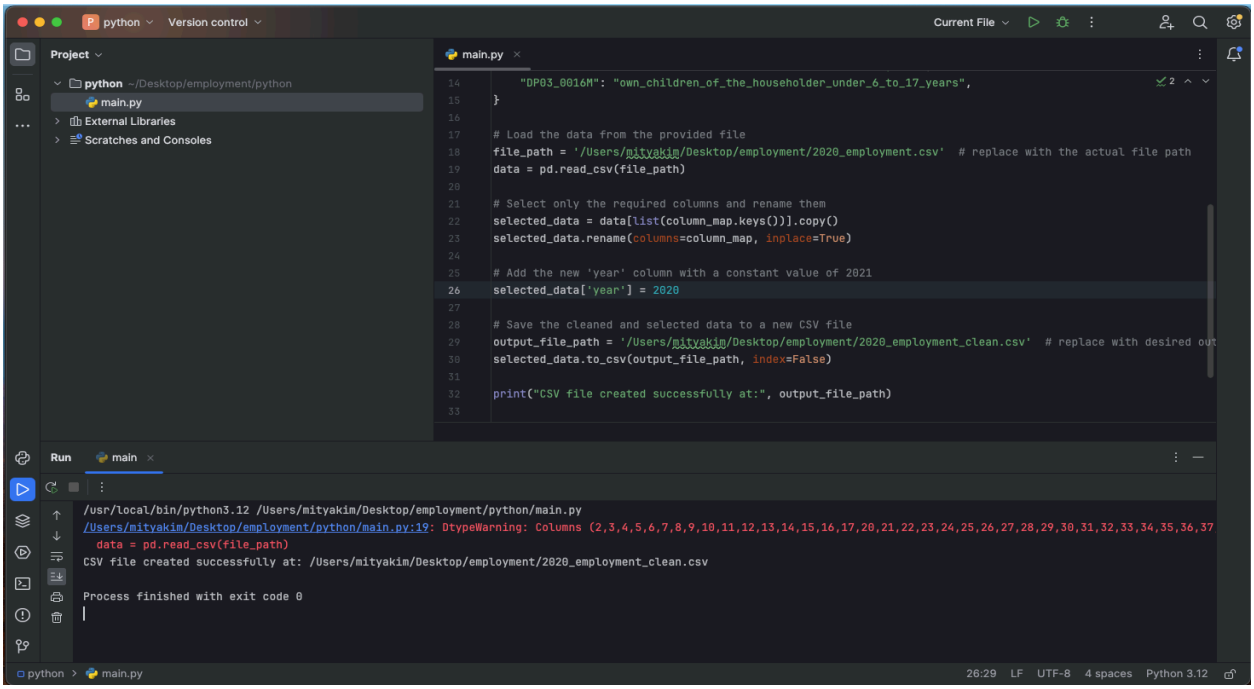


The same filters for the Year and Geographic. Download data for LA and Chicago for 2020, 2021 and 2022.

- Next clean data and get columns using Python:

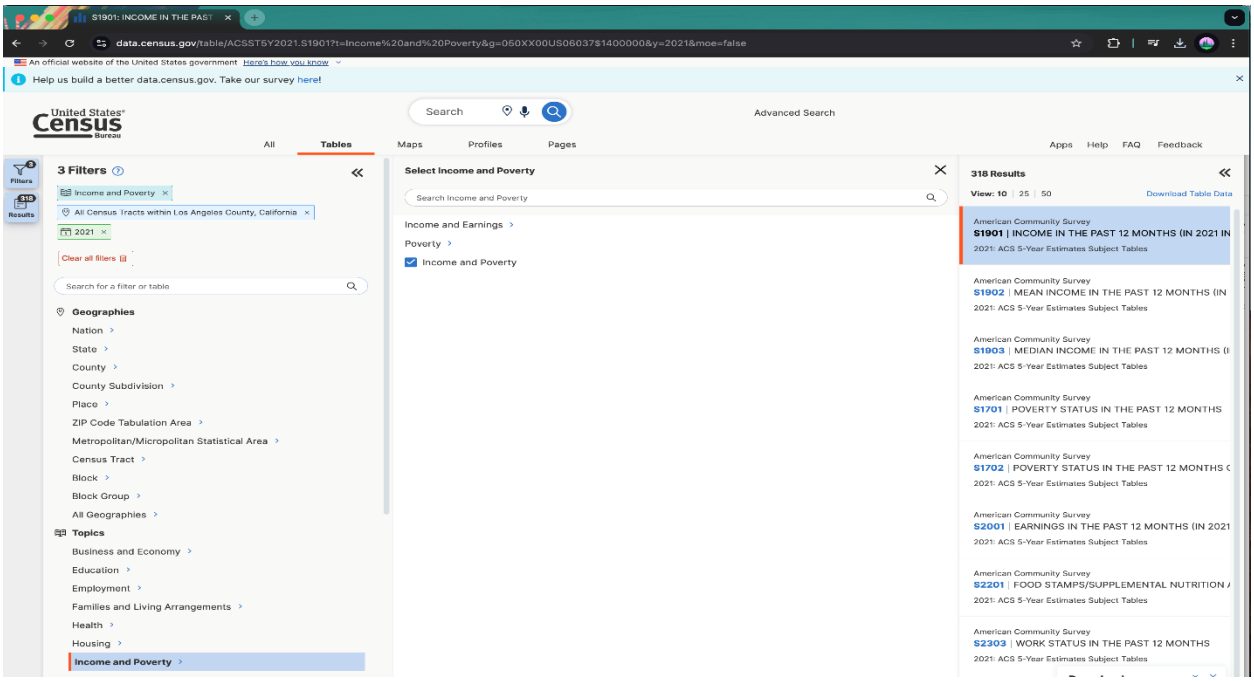
```
import pandas as pd

# Define the mapping of original columns to new column names
column_map = {
    "GEO_ID": "geo_id",
    "NAME": "area_name",

    "DP03_0001E": "population_age_16_and_over",
    "DP03_0002E": "population_age_16_and_over_in_labor_force",
    "DP03_0004E": "population_age_16_and_over_in_civ_labor_force_employed",
    "DP03_0005E": "population_age_16_and_over_in_civ_labor_force_unemployed",
    "DP03_0007E": "population_age_16_and_over_not_in_labor_force",
    "DP03_0014E": "own_children_of_the_householder_under_6_years",
    "DP03_0016E": "own_children_of_the_householder_under_6_to_17_years",
}
# Load the data from the provided file
file_path = '/Users/mityakim/Desktop/employment/2020_employment.csv'  # replace with the actual file path
data = pd.read_csv(file_path)

# Select only the required columns and rename them
selected_data = data[list(column_map.keys())].copy()
selected_data.rename(columns=column_map, inplace=True)

# Add the new 'year' column with a constant value of 2020
```

```
selected_data['year'] = 2020

# Save the cleaned and selected data to a new CSV file
output_file_path = '/Users/mityakim/Desktop/employment/2020_employment_clean.csv'  # replace with desired output path
selected_data.to_csv(output_file_path, index=False)

print("CSV file created successfully at:", output_file_path)
```



Income Data

- Topics → Income and Poverty → Income and Poverty

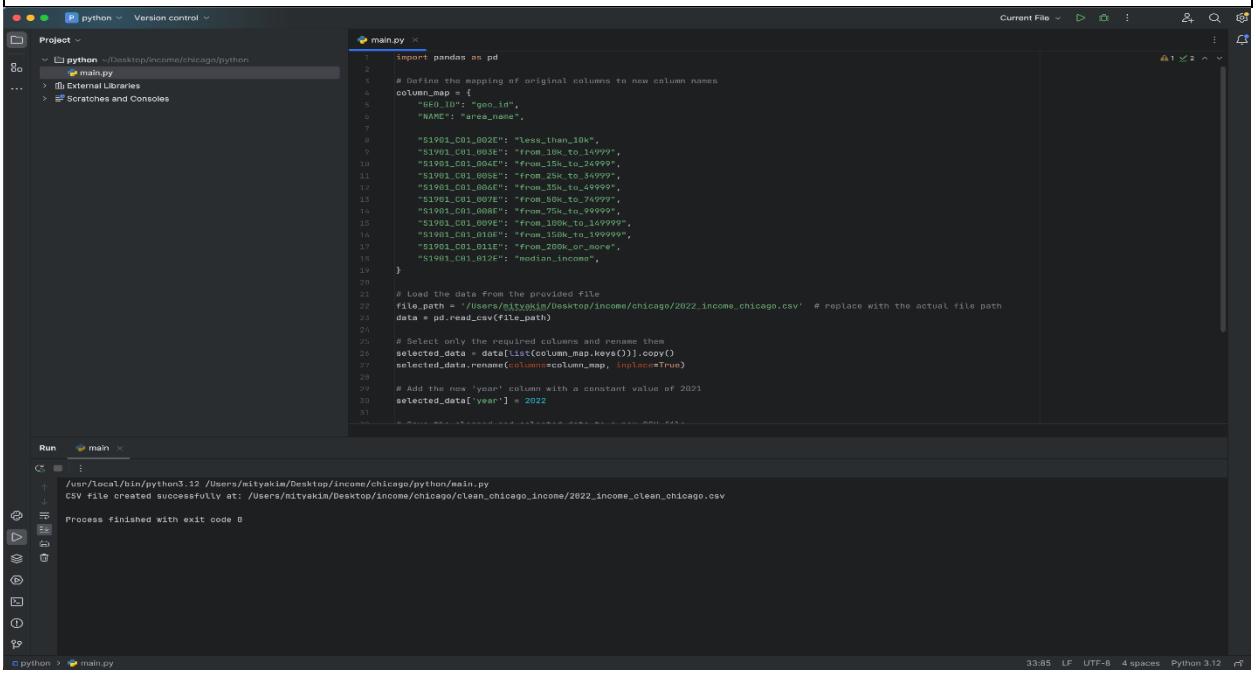


The same filters for the Year and Geographic. Download data for LA and Chicago for 2020, 2021 and 2022.

Next clean data and get columns using Python:

```
import pandas as pd

# Define the mapping of original columns to new column names
column_map = {
    "GEO_ID": "geo_id",
    "NAME": "area_name",

    "S1901_C01_002E": "less_than_10k",
    "S1901_C01_003E": "from_10k_to_14999",
    "S1901_C01_004E": "from_15k_to_24999",
    "S1901_C01_005E": "from_25k_to_34999",
    "S1901_C01_006E": "from_35k_to_49999",
    "S1901_C01_007E": "from_50k_to_74999",
    "S1901_C01_008E": "from_75k_to_99999",
    "S1901_C01_009E": "from_100k_to_149999",
    "S1901_C01_010E": "from_150k_to_199999",
    "S1901_C01_011E": "from_200k_or_more",
    "S1901_C01_012E": "median_income",
}

# Load the data from the provided file
file_path = '/Users/mityakim/Desktop/income/chicago/2022_income_chicago.csv'  # replace with the actual file path
```

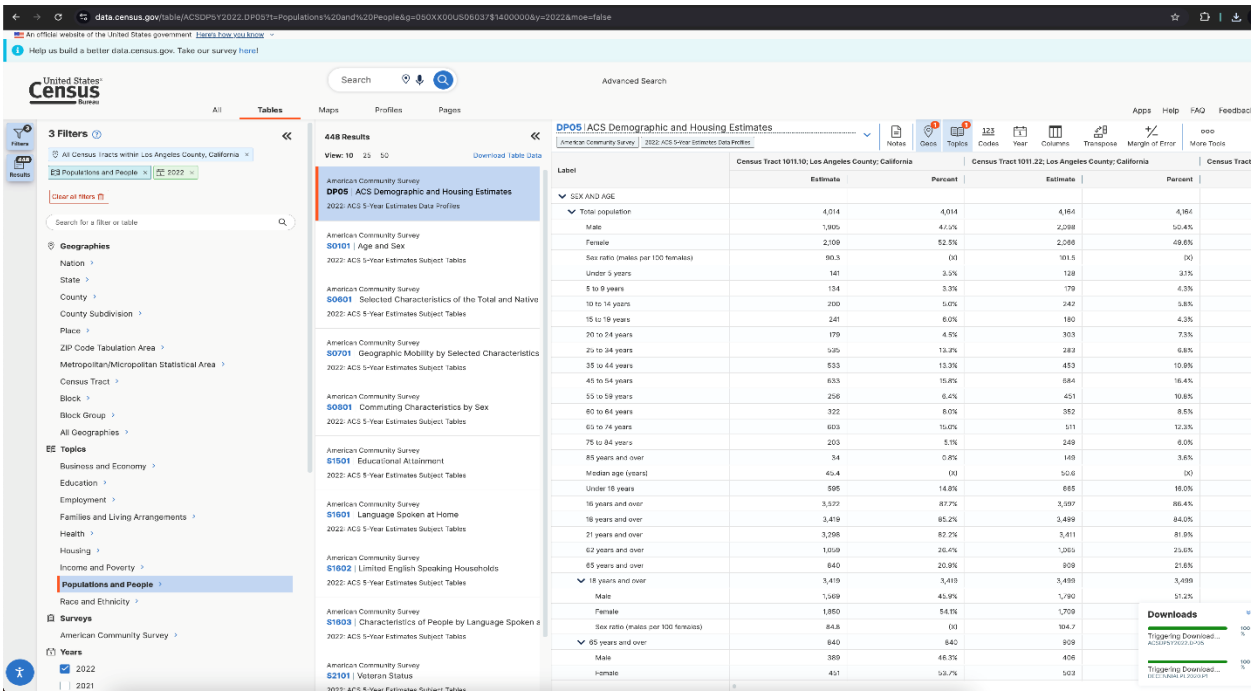
```
data = pd.read_csv(file_path)

# Select only the required columns and rename them
selected_data = data[list(column_map.keys())].copy()
selected_data.rename(columns=column_map, inplace=True)

# Add the new 'year' column with a constant value of 2021
selected_data['year'] = 2022

# Save the cleaned and selected data to a new CSV file
output_file_path =
'/Users/mityakim/Desktop/income/chicago/clean_chicago_income/2022_income_clean_chicago.csv'  # replace with desired output path
selected_data.to_csv(output_file_path, index=False)

print("CSV file created successfully at:", output_file_path)
```



Population Data

Topics → Population and People

- DP05 ACS Demographic and Housing Estimates

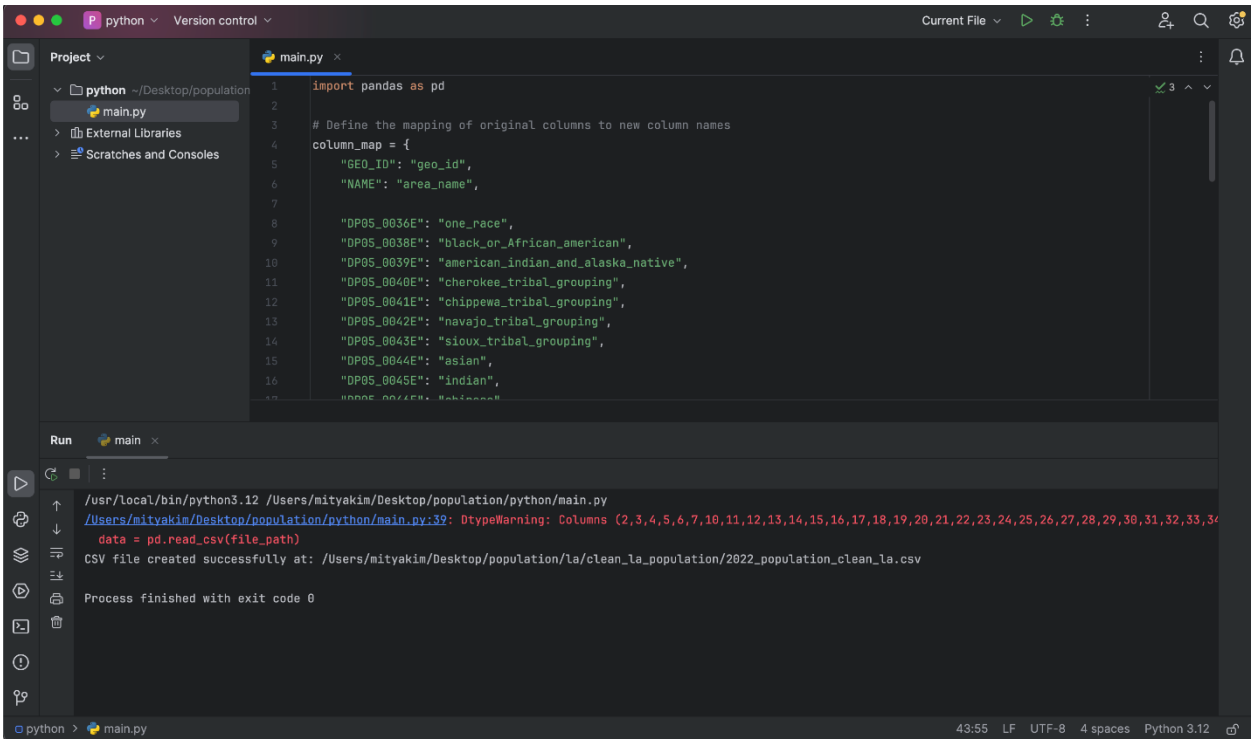


The same filters for the Year and Geographic. Download data for LA and Chicago for 2020, 2021 and 2022.

Next clean data and get columns using Python:

```
import pandas as pd

# Define the mapping of original columns to new column names
column_map = {
    "GEO_ID": "geo_id",
    "NAME": "area_name",

    "DP05_0036E": "one_race",
    "DP05_0038E": "black_or_African_american",
    "DP05_0039E": "american_indian_and_alaska_native",
    "DP05_0040E": "cherokee_tribal_grouping",
    "DP05_0041E": "chippewa_tribal_grouping",
    "DP05_0042E": "navajo_tribal_grouping",
    "DP05_0043E": "sioux_tribal_grouping",
    "DP05_0044E": "asian",
    "DP05_0045E": "indian",
    "DP05_0046E": "chinese",
    "DP05_0047E": "filipino",
    "DP05_0048E": "japanese",
    "DP05_0049E": "korean",
    "DP05_0050E": "vietnamese",
    "DP05_0051E": "other_asian",
```

```
    "DP05_0052E": "native_hawaiian_and_other_pacific_islander",
    "DP05_0053E": "chamorro",
    "DP05_0054E": "native_hawaiian",
    "DP05_0055E": "samoan",
    "DP05_0056E": "other_native_hawaiian_and_other_pacific_pslander",
    "DP05_0065E":
    "race_alone_or_in_combination_with_one_or_more_other_races_total_population",
    "DP05_0072E": "hispanic_or_latino_total_population",
    "DP05_0073E": "hispanic_or_latino_total_population_of_any_race",
    "DP05_0074E": "mexican",
    "DP05_0075E": "puerto_rican",
    "DP05_0076E": "cuban",
    "DP05_0076M": "other_hispanic_or_latino"
}

# Load the data from the provided file
file_path = '/Users/mityakim/Desktop/population/la/2020_population_la.csv'
data = pd.read_csv(file_path)

# Select only the required columns and rename them
selected_data = data[list(column_map.keys())].copy()
selected_data.rename(columns=column_map, inplace=True)

# Add the new 'year' column with a constant value of 2020
selected_data['year'] = 2020

# Save the cleaned and selected data to a new CSV file
output_file_path =
'/Users/mityakim/Desktop/population/la/clean_la_population/2020_population_clean_la.csv'
selected_data.to_csv(output_file_path, index=False)

print("CSV file created successfully at:", output_file_path)
```



Step 5: Education, Employment, Income, Population Data Loaded into Hadoop Clusters

- Put all clean files in data.zip and then open the terminal. Do not forget to use your path and username:
scp "/Users/mityakim/Desktop/data.zip" dkim171@129.146.230.230:/home/dkim171

- Check files on Linux server:

ssh dkim171@129.146.230.230

- Enter command:

```
-bash-4.2$ ls
```

```
[-bash-4.2$ ls
000000_0                              CrimesChicago.zip  log_result.out              SensorFiles
909f2b.log                            Crime.zip          __MACOSX                    senti_out.csv
Austin                                Dallas             minion_tweets.tar.gz        time_zone_map.tsv
Austin_Crime.csv                      data.zip           ratings_2012.txt            Tweets
baseball_salaries_2003.txt            dictionary.tsv     ratings_2013.txt
Chicago                               LA                 reddit_opinion_PSE_ISR.csv
Crimes_-_2001_to_Present_20241019.csv LOAD               reddit.zip
-bash-4.2$
```

- Then unzip data:

```
-bash-4.2$ unzip data
```

- Create project folder and sub folders

```
-bash-4.2$ hdfs dfs -mkdir Project

-bash-4.2$ hdfs dfs -mkdir Project/education
-bash-4.2$ hdfs dfs -mkdir Project/education/la
-bash-4.2$ hdfs dfs -mkdir Project/education/chicago

-bash-4.2$ hdfs dfs -mkdir Project/employment
-bash-4.2$ hdfs dfs -mkdir Project/employment/la
-bash-4.2$ hdfs dfs -mkdir Project/employment/chicago
-bash-4.2$ hdfs dfs -mkdir Project/income
-bash-4.2$ hdfs dfs -mkdir Project/income/la
-bash-4.2$ hdfs dfs -mkdir Project/income/chicago

-bash-4.2$ hdfs dfs -mkdir Project/population
-bash-4.2$ hdfs dfs -mkdir Project/population/la
-bash-4.2$ hdfs dfs -mkdir Project/population/chicago
```

- Move all files to the HDFS directories respectively:

Chicago education

```
-bash-4.2$ hdfs dfs -put 2020_education_clean_chicago.csv Project/education/chicago
-bash-4.2$ hdfs dfs -put 2021_education_clean_chicago.csv Project/education/chicago
-bash-4.2$ hdfs dfs -put 2022_education_clean_chicago.csv Project/education/chicago
```

- to check files, enter:

```
-bash-4.2$ hdfs dfs -ls Project/education/chicago
```

LA education

```
-bash-4.2$ hdfs dfs -put 2020_education_clean.csv Project/education/la
-bash-4.2$ hdfs dfs -put 2021_education_clean.csv Project/education/la
-bash-4.2$ hdfs dfs -put 2022_education_clean.csv Project/education/la
```

- to check files, enter:

```
-bash-4.2$ hdfs dfs -ls Project/education/la
```

Chicago Employment

```
-bash-4.2$ hdfs dfs -put 2020_employment_clean_chicago.csv Project/employment/chicago
-bash-4.2$ hdfs dfs -put 2021_employment_clean_chicago.csv Project/employment/chicago
-bash-4.2$ hdfs dfs -put 2022_employment_clean_chicago.csv Project/employment/chicago
```

- to check files, enter:

```
-bash-4.2$ hdfs dfs -ls Project/employment/chicago
```

LA employment

```
-bash-4.2$ hdfs dfs -put 2020_employment_clean.csv Project/employment/la
-bash-4.2$ hdfs dfs -put 2021_employment_clean.csv Project/employment/la
-bash-4.2$ hdfs dfs -put 2022_employment_clean.csv Project/employment/la
```

- to check files, enter:

```
-bash-4.2$ hdfs dfs -ls Project/employment/la
```

Chicago income

```
-bash-4.2$ hdfs dfs -put 2020_income_clean_chicago.csv Project/income/chicago
-bash-4.2$ hdfs dfs -put 2021_income_clean_chicago.csv Project/income/chicago
-bash-4.2$ hdfs dfs -put 2022_income_clean_chicago.csv Project/income/chicago
```

- to check files, enter:

```
-bash-4.2$ hdfs dfs -ls Project/income/chicago
```

LA income

```
-bash-4.2$ hdfs dfs -put 2020_income_clean_la.csv Project/income/la
-bash-4.2$ hdfs dfs -put 2021_income_clean_la.csv Project/income/la
-bash-4.2$ hdfs dfs -put 2022_income_clean_la.csv Project/income/la
```

- to check files, enter:

```
-bash-4.2$ hdfs dfs -ls Project/income/la
```

Chicago population

```
-bash-4.2$ hdfs dfs -put 2020_population_clean_chicago.csv Project/population/chicago
-bash-4.2$ hdfs dfs -put 2021_population_clean_chicago.csv Project/population/chicago
-bash-4.2$ hdfs dfs -put 2022_population_clean_chicago.csv Project/population/chicago
```

- to check files, enter:

```
-bash-4.2$ hdfs dfs -ls Project/population/chicago
```

LA population

```
-bash-4.2$ hdfs dfs -put 2020_population_clean_la.csv Project/population/la
-bash-4.2$ hdfs dfs -put 2021_population_clean_la.csv Project/population/la
-bash-4.2$ hdfs dfs -put 2022_population_clean_la.csv Project/population/la
```

- to check files, enter:

```
-bash-4.2$ hdfs dfs -ls Project/population/la
```

Step 6: Creating Hive tables to Query Education, Employment, Income and Population Data

- Enter Hadoop:

```
-bash-4.2$ beeline
```

- Use database:

```
0: jdbc:hive2://bigdaiun0.sub03291929060.trai> use 5200group2
```

Create LA education table for 2020, 2021, 2022 years

```
DROP TABLE IF EXISTS original_la_education;

CREATE EXTERNAL TABLE IF NOT EXISTS original_la_education(
    geo_id STRING,
   area_name STRING,
   population_age_18_24 STRING,
   high_school_grad_or_higher_18_24 STRING,
   population_25_34_years STRING,
   high_school_grad_or_higher_25_34 STRING,
   population_35_44_years STRING,
   high_school_grad_or_higher_35_44 STRING,
   population_45_64_years STRING,
   high_school_grad_or_higher_45_64 STRING,
   population_age_65_and_over STRING,
   high_school_grad_or_higher_65_and_over STRING,
   `year` STRING
   )
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
  "separatorChar" = ",",
  "quoteChar" = "\"",
  "escapeChar" = "\\"
)
LOCATION '/user/dkim171/Project/education/la/'
```

```
TBLPROPERTIES ('skip.header.line.count'='2');
```

- Check if the table was successfully created:

```
0: jdbc:hive2://bigdaiun0.sub03291929060.trai> show tables;
+--------------------------------+
|            tab_name            |
+--------------------------------+
| original_la_education          |
+--------------------------------+

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select* from
original_la_education LIMIT 10;
```

Create Chicago education table for 2020, 2021, 2022 years

```
DROP TABLE IF EXISTS original_chicago_education;

CREATE EXTERNAL TABLE IF NOT EXISTS original_chicago_education(
    geo_id STRING,
   area_name STRING,
   population_age_18_24 STRING,
   high_school_grad_or_higher_18_24 STRING,
   population_25_34_years STRING,
   high_school_grad_or_higher_25_34 STRING,
   population_35_44_years STRING,
   high_school_grad_or_higher_35_44 STRING,
   population_45_64_years STRING,
   high_school_grad_or_higher_45_64 STRING,
   population_age_65_and_over STRING,
   high_school_grad_or_higher_65_and_over STRING,
   `year` STRING
   )
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
  "separatorChar" = ",",
  "quoteChar" = "\"",
  "escapeChar" = "\\"
)
LOCATION '/user/dkim171/Project/education/chicago/'
TBLPROPERTIES ('skip.header.line.count'='2');
```

- Check if the table was successfully created:

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

```
0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select* from
original_chicago_education LIMIT 10;
```

Create LA employment table for 2020, 2021, 2022 years

```
DROP TABLE IF EXISTS original_la_employment;

CREATE EXTERNAL TABLE IF NOT EXISTS original_la_employment(
    geo_id STRING,
   area_name STRING,
   population_age_16_and_over STRING,
   population_age_16_and_over_in_labor_force STRING,
   population_age_16_and_over_in_civ_labor_force_employed STRING,
   population_age_16_and_over_in_civ_labor_force_unemployed STRING,
   population_age_16_and_over_not_in_labor_force STRING,
   own_children_of_the_householder_under_6_years STRING,
   own_children_of_the_householder_under_6_to_17_years STRING,
   `year` STRING
   )
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
  "separatorChar" = ",",
  "quoteChar" = "\"",
  "escapeChar" = "\\"
)
LOCATION '/user/dkim171/Project/employment/la/'
TBLPROPERTIES ('skip.header.line.count'='2');
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select* from
original_la_employment LIMIT 10;
```

Create Chicago employment table for 2020, 2021, 2022 years

```
DROP TABLE IF EXISTS original_chicago_employment;

CREATE EXTERNAL TABLE IF NOT EXISTS original_chicago_employment(
    geo_id STRING,
   area_name STRING,
   population_age_16_and_over STRING,
   population_age_16_and_over_in_labor_force STRING,
   population_age_16_and_over_in_civ_labor_force_employed STRING,
```

```
    population_age_16_and_over_in_civ_labor_force_unemployed STRING,
    population_age_16_and_over_not_in_labor_force STRING,
    own_children_of_the_householder_under_6_years STRING,
    own_children_of_the_householder_under_6_to_17_years STRING,
    `year` STRING
    )
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
   "separatorChar" = ",",
   "quoteChar" = "\"",
   "escapeChar" = "\\"
)
LOCATION '/user/dkim171/Project/employment/chicago/'
TBLPROPERTIES ('skip.header.line.count'='2');
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select* from
original_chicago_employment LIMIT 10;
```

Create LA income table for 2020, 2021, 2022 years

```
DROP TABLE IF EXISTS original_la_income;

CREATE EXTERNAL TABLE IF NOT EXISTS original_la_income(
    geo_id STRING,
    area_name STRING,

    less_than_10k STRING,
    from_10k_to_14999 STRING,
    from_15k_to_24999 STRING,
    from_25k_to_34999 STRING,
    from_35k_to_49999 STRING,
    from_50k_to_74999 STRING,
    from_75k_to_99999 STRING,
    from_100k_to_149999 STRING,
    from_150k_to_199999 STRING,
    from_200k_or_more STRING,
    median_income STRING,

    `year` STRING
    )
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
   "separatorChar" = ",",
   "quoteChar" = "\"",
```

```
  "escapeChar" = "\\"
)
LOCATION '/user/dkim171/Project/income/la/'
TBLPROPERTIES ('skip.header.line.count'='2');
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select* from original_la_income
LIMIT 10;
```

Create Chicago income table for 2020, 2021, 2022 years

```
DROP TABLE IF EXISTS original_chicago_income;

CREATE EXTERNAL TABLE IF NOT EXISTS original_chicago_income(
    geo_id STRING,
   area_name STRING,

   less_than_10k STRING,
   from_10k_to_14999 STRING,
   from_15k_to_24999 STRING,
   from_25k_to_34999 STRING,
   from_35k_to_49999 STRING,
   from_50k_to_74999 STRING,
   from_75k_to_99999 STRING,
   from_100k_to_149999 STRING,
   from_150k_to_199999 STRING,
   from_200k_or_more STRING,
   median_income STRING,

   `year` STRING
   )
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
  "separatorChar" = ",",
  "quoteChar" = "\"",
  "escapeChar" = "\\"
)
LOCATION '/user/dkim171/Project/income/chicago/'
TBLPROPERTIES ('skip.header.line.count'='2');
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select* from
original_chicago_income LIMIT 10;
```

Create LA Population table for 2020, 2022 years

```
DROP TABLE IF EXISTS original_la_population;

CREATE EXTERNAL TABLE IF NOT EXISTS original_la_population(
    geo_id STRING,
   area_name STRING,

   one_race STRING,
   black_or_African_american STRING,
   merican_indian_and_alaska_native STRING,
   herokee_tribal_grouping STRING,
   chippewa_tribal_grouping STRING,
   navajo_tribal_grouping STRING,
   sioux_tribal_grouping STRING,
   asian STRING,
   indian STRING,
   chinese STRING,
   ilipino STRING,
   japanese STRING,
   korean STRING,
   vietnamese STRING,
   ther_asian STRING,
   native_hawaiian_and_other_pacific_islander STRING,
   chamorro STRING,
   native_hawaiian STRING,
   samoan STRING,
   ther_native_hawaiian_and_other_pacific_pslander STRING,
   race_alone_or_in_combination_with_one_or_more_other_races_total_population STRING,
   hispanic_or_latino_total_population STRING,
   ispanic_or_latino_total_population_of_any_race STRING,
   mexican STRING,
   puerto_rican STRING,
   cuban STRING,
   other_hispanic_or_latin STRING,

   `year` STRING
   )
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
  "separatorChar" = ",",
  "quoteChar" = "\"",
  "escapeChar" = "\\"
)
```

```
LOCATION '/user/dkim171/Project/population/la/'
TBLPROPERTIES ('skip.header.line.count'='2');
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select* from
original_la_population LIMIT 10;
```

Create Chicago Population table for 2020, 2022 years

```
DROP TABLE IF EXISTS original_chicago_population;

CREATE EXTERNAL TABLE IF NOT EXISTS original_chicago_population(
    geo_id STRING,
   area_name STRING,

   one_race STRING,
   black_or_African_american STRING,
   merican_indian_and_alaska_native STRING,
   herokee_tribal_grouping STRING,
   chippewa_tribal_grouping STRING,
   navajo_tribal_grouping STRING,
   sioux_tribal_grouping STRING,
   asian STRING,
   indian STRING,
   chinese STRING,
   ilipino STRING,
   japanese STRING,
   korean STRING,
   vietnamese STRING,
   ther_asian STRING,
   native_hawaiian_and_other_pacific_islander STRING,
   chamorro STRING,
   native_hawaiian STRING,
   samoan STRING,
   ther_native_hawaiian_and_other_pacific_pslander STRING,
   race_alone_or_in_combination_with_one_or_more_other_races_total_population STRING,
   hispanic_or_latino_total_population STRING,
   ispanic_or_latino_total_population_of_any_race STRING,
   mexican STRING,
   puerto_rican STRING,
   cuban STRING,
   other_hispanic_or_latin STRING,
```

```
    `year` STRING
    )
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
   "separatorChar" = ",",
   "quoteChar" = "\"",
   "escapeChar" = "\\"
)
LOCATION '/user/dkim171/Project/population/chicago/'
TBLPROPERTIES ('skip.header.line.count'='2');
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select* from
original_chicago_population LIMIT 10;
```

Step 7: Join Hive tables Education, Employment, Income and Population with Crime Data and download for the future analysis

Create Los Angeles crime description table

```
DROP TABLE IF EXISTS crime_description_la;

CREATE TABLE crime_description_la AS
SELECT DISTINCT
   CAST(crm_cd AS INT) AS crm_cd,
   crm_cd_desc
FROM original_la_crime;
```

Join Los Angeles education and crime tables and download final file

```
DROP TABLE IF EXISTS la_education_and_crime;
CREATE TABLE la_education_and_crime AS
SELECT
   SUBSTRING(c.geo_id, LOCATE('US', c.geo_id) + 2) AS tract_id,
   c.year,
   CAST(p. primary_type AS INT) AS crm_cd,
   p.count,
   c.population_age_18_24,
   c.high_school_grad_or_higher_18_24,
```

```
    c.population_25_34_years,
    c.high_school_grad_or_higher_25_34,
    c.population_35_44_years,
    c.high_school_grad_or_higher_35_44,
    c.population_45_64_years,
    c.high_school_grad_or_higher_45_64,
    c.population_age_65_and_over,
    c.high_school_grad_or_higher_65_and_over,
    'Los Angeles County' AS area_name
FROM
  original_la_education c
LEFT JOIN
   la_tractid p
ON
   c.year = p.year
   AND SUBSTRING(c.geo_id, LOCATE('US', c.geo_id) + 2) = p.tract_id
ORDER BY
   c.year;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
la_education_and_crime LIMIT 10;
```

Add new column with crime description:

```
DROP TABLE IF EXISTS la_education_with_crime_description;

CREATE TABLE la_education_with_crime_description AS
SELECT t1.*,
REGEXP_REPLACE(t2.crm_cd_desc, ',', '') AS description
FROM
   la_education_and_crime t1
LEFT JOIN
   crime_description_la t2
ON
   t1.crm_cd = t2.crm_cd
ORDER BY t1.year;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
la_education_with_crime_description LIMIT 10;
```

Since the education table has some tractid values and the crime dataset does not, we need to **clean up our output table.**

```
DROP TABLE IF EXISTS final_la_education_and_crime;

CREATE TABLE final_la_education_and_crime
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/dkim171/Project/education/final/la'
AS
SELECT *
FROM la_education_with_crime_description
WHERE
   crm_cd IS NOT NULL
   AND count IS NOT NULL;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
final_la_education_and_crime LIMIT 10;
```

- Open Linux terminal and check file:

hdfs dfs -ls /user/dkim171/Project/education/final/la

- Get file from the HDFS to the Linux server:

hdfs dfs -get /user/dkim171/Project/education/final/la/000000_0

- Exit from the Linux server and enter command in terminal to download file:

scp dkim171@129.146.230.230:/home/dkim171/000000_0 ~/Desktop/

Join Chicago education and crime tables and download final file

```
DROP TABLE IF EXISTS chicago_education_and_crime;

CREATE TABLE chicago_education_and_crime AS
SELECT
   SUBSTRING(c.geo_id, LOCATE('US', c.geo_id) + 2) AS tract_id,
   c.year,
   p. primary_type,
   p.count,
```

```
    c.population_age_18_24,
    c.high_school_grad_or_higher_18_24,
    c.population_25_34_years,
    c.high_school_grad_or_higher_25_34,
    c.population_35_44_years,
    c.high_school_grad_or_higher_35_44,
    c.population_45_64_years STRING,
    c.high_school_grad_or_higher_45_64,
    c.population_age_65_and_over,
    c.high_school_grad_or_higher_65_and_over,
    'Cook County' AS area_name
FROM
  original_chicago_education c
LEFT JOIN
   chicago_tractid p
ON
   c.year = p.year
   AND SUBSTRING(c.geo_id, LOCATE('US', c.geo_id) + 2) = p.tract_id
ORDER BY
   c.year;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
final_chicago_education_and_crime LIMIT 10;
```

Clean up our output table.

```
DROP TABLE IF EXISTS final_chicago_education_and_crime;

CREATE TABLE final_chicago_education_and_crime
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/dkim171/Project/education/final/chicago'

AS
SELECT *
FROM chicago_education_and_crime
WHERE
   primary_type IS NOT NULL
   AND count IS NOT NULL;
```

- Check if the table was successfully created:

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

```
0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
final_chicago_education_and_crime LIMIT 10;
```

- Open Linux terminal and check file:

hdfs dfs -ls /user/dkim171/Project/education/final/chicago

- Remove file from the linux server

```
-bash-4.2$ rm 000000_0
```

- Get new file from the HDFS to the Linux server:

hdfs dfs -get /user/dkim171/Project/education/final/000000_0

- Exit from the Linux server and enter command in terminal to download file:

scp dkim171@129.146.230.230:/home/dkim171/000000_0 ~/Desktop/

Join LA employment and crime tables and download final file

```
DROP TABLE IF EXISTS la_employment_and_crime;

CREATE TABLE la_employment_and_crime AS
SELECT
   SUBSTRING(c.geo_id, LOCATE('US', c.geo_id) + 2) AS tract_id,
   c.year,
   CAST(p. primary_type AS INT) AS crm_cd,
   p.count,
   c.population_age_16_and_over,
   c.population_age_16_and_over_in_labor_force,
   c.population_age_16_and_over_in_civ_labor_force_employed,
   c.population_age_16_and_over_in_civ_labor_force_unemployed,
   c.population_age_16_and_over_not_in_labor_force,
   c.own_children_of_the_householder_under_6_years,
   c.own_children_of_the_householder_under_6_to_17_years,
   'Los Angeles County' AS area_name
FROM
  original_la_employment c
LEFT JOIN
   la_tractid p
ON
   c.year = p.year
```

```
  AND SUBSTRING(c.geo_id, LOCATE('US', c.geo_id) + 2) = p.tract_id
ORDER BY
  c.year;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
la_employment_and_crime LIMIT 10;
```

Add new column with crime description:

```
DROP TABLE IF EXISTS la_employment_with_crime_description;

CREATE TABLE la_employment_with_crime_description AS
SELECT t1.*,
REGEXP_REPLACE(t2.crm_cd_desc, ',', '') AS description
FROM
   la_employment_and_crime t1
LEFT JOIN
  crime_description_la t2
ON
  t1.crm_cd = t2.crm_cd
ORDER BY t1.year;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
la_employment_with_crime_description LIMIT 10;
```

- Clean up our output table.

```
DROP TABLE IF EXISTS final_la_employment_and_crime;

CREATE TABLE final_la_employment_and_crime
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/dkim171/Project/employment/final/la'

AS
SELECT *
FROM la_employment_with_crime_description
WHERE
    crm_cd IS NOT NULL
   AND count IS NOT NULL;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
final_la_employment_and_crime LIMIT 10;
```

- Open Linux terminal and check file:

hdfs dfs -ls /user/dkim171/Project/employment/final/la

- Remove file from the Linux server

```
-bash-4.2$ rm 000000_0
```

- Get new file from the HDFS to the Linux server:

hdfs dfs -get /user/dkim171/Project/employment/final/la/000000_0

- Exit from the Linux server and enter command in terminal to download file:

scp dkim171@129.146.230.230:/home/dkim171/000000_0 ~/Desktop/

Join Chicago employment and crime tables and download final file

```
DROP TABLE IF EXISTS chicago_employment_and_crime;

CREATE TABLE chicago_employment_and_crime AS
SELECT
   SUBSTRING(c.geo_id, LOCATE('US', c.geo_id) + 2) AS tract_id,
   c.year,
   p. primary_type,
   p.count,

   c.population_age_16_and_over,
   c.population_age_16_and_over_in_labor_force,
   c.population_age_16_and_over_in_civ_labor_force_employed,
   c.population_age_16_and_over_in_civ_labor_force_unemployed,
   c.population_age_16_and_over_not_in_labor_force,
   c.own_children_of_the_householder_under_6_years,
   c.own_children_of_the_householder_under_6_to_17_years,
   'Cook County' AS area_name
FROM
  original_chicago_employment c
LEFT JOIN
   chicago_tractid p
ON
   c.year = p.year
   AND SUBSTRING(c.geo_id, LOCATE('US', c.geo_id) + 2) = p.tract_id
```

```
ORDER BY
  c.year;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
chicago_employment_and_crime LIMIT 10;
```

- Clean up our output table.

```
DROP TABLE IF EXISTS final_chicago_employment_and_crime;

CREATE TABLE final_chicago_employment_and_crime
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/dkim171/Project/employment/final/chicago'

AS
SELECT *
FROM chicago_employment_and_crime
WHERE
   primary_type IS NOT NULL
   AND count IS NOT NULL;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai>select * from
final_chicago_employment_and_crime LIMIT 10;
```

- Open Linux terminal and check file:

hdfs dfs -ls /user/dkim171/Project/employment/final/chicago

- Remove file from the Linux server

```
-bash-4.2$ rm 000000_0
```

- Get new file from the HDFS to the Linux server:

hdfs dfs -get /user/dkim171/Project/employment/final/chicago/000000_0

- Exit from the Linux server and enter command in terminal to download file:

scp dkim171@129.146.230.230:/home/dkim171/000000_0 ~/Desktop/

Join LA Income and Crime tables and download final file

```
DROP TABLE IF EXISTS la_income_and_crime;

CREATE TABLE la_income_and_crime AS
SELECT
   SUBSTRING(c.geo_id, LOCATE('US', c.geo_id) + 2) AS tract_id,
   c.year,
   CAST(p. primary_type AS INT) AS crm_cd,
   p.count,
   c.less_than_10k,
   c.from_10k_to_14999,
   c.from_15k_to_24999,
   c.from_25k_to_34999,
   c.from_35k_to_49999,
   c.from_50k_to_74999,
   c.from_75k_to_99999,
   c.from_100k_to_149999,
   c.from_150k_to_199999,
   c.from_200k_or_more,
   REGEXP_REPLACE(c.median_income, '250,000\\+', '250000') AS median_income,
   'Los Angeles County' AS area_name
FROM
  original_la_income c
LEFT JOIN
   la_tractid p
ON
   c.year = p.year
   AND SUBSTRING(c.geo_id, LOCATE('US', c.geo_id) + 2) = p.tract_id
ORDER BY
   c.year;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
la_income_and_crime LIMIT 10;
```

- Add new column with crime description:

```
DROP TABLE IF EXISTS la_income_with_crime_description;

CREATE TABLE la_income_with_crime_description AS
```

```
SELECT t1.*,
REGEXP_REPLACE(t2.crm_cd_desc, ',', '') AS description
FROM
   la_income_and_crime t1
LEFT JOIN
  crime_description_la t2
ON
  t1.crm_cd = t2.crm_cd
ORDER BY t1.year;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
la_income_with_crime_description LIMIT 10;
```

- Clean up our output table.

```
DROP TABLE IF EXISTS final_la_income_and_crime;

CREATE TABLE final_la_income_and_crime
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/dkim171/Project/income/final/la'

AS
SELECT *
FROM la_income_with_crime_description
WHERE
   crm_cd IS NOT NULL
   AND count IS NOT NULL;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
final_la_income_and_crime LIMIT 10;
```

- Open Linux terminal and check file:

```
hdfs dfs -ls /user/dkim171/Project/income/final/la
```

- Remove previous file from the Linux server

```
-bash-4.2$ rm 000000_0
```

- Get new file from the HDFS to the Linux server:

hdfs dfs -get /user/dkim171/Project/income/final/la/000000_0

- Exit from the Linux server and enter command in terminal to download file:

scp dkim171@129.146.230.230:/home/dkim171/000000_0 ~/Desktop/

Join Chicago Income and Crime tables and download final file

```
DROP TABLE IF EXISTS chicago_income_and_crime;

CREATE TABLE chicago_income_and_crime AS
SELECT
   SUBSTRING(c.geo_id, LOCATE('US', c.geo_id) + 2) AS tract_id,
   c.year,
   p. primary_type,
   p.count,

   c.less_than_10k,
   c.from_10k_to_14999,
   c.from_15k_to_24999,
   c.from_25k_to_34999,
   c.from_35k_to_49999,
   c.from_50k_to_74999,
   c.from_75k_to_99999,
   c.from_100k_to_149999,
   c.from_150k_to_199999,
   c.from_200k_or_more,
   c.median_income,

   'Cook County' AS area_name
FROM
  original_chicago_income c
LEFT JOIN
   chicago_tractid p
ON
   c.year = p.year
   AND SUBSTRING(c.geo_id, LOCATE('US', c.geo_id) + 2) = p.tract_id
ORDER BY
   c.year;
```

- Check if the table was successfully created:

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

```
0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
chicago_income_and_crime LIMIT 10;
```

- Clean up our output table.

```
DROP TABLE IF EXISTS final_chicago_income_and_crime;

CREATE TABLE final_chicago_income_and_crime
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/dkim171/Project/income/final/chicago'

AS
SELECT *
FROM chicago_income_and_crime
WHERE
   primary_type IS NOT NULL
   AND count IS NOT NULL;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
final_chicago_income_and_crime LIMIT 10;
```

- Open Linux terminal and check file:

hdfs dfs -ls /user/dkim171/Project/income/final/chicago

- Remove previous file from the Linux server

```
-bash-4.2$ rm 000000_0
```

- Get new file from the HDFS to the Linux server:

hdfs dfs -get /user/dkim171/Project/income/final/chicago/000000_0

- Exit from the Linux server and enter command in terminal to download file:

scp dkim171@129.146.230.230:/home/dkim171/000000_0 ~/Desktop/

Join LA Population and Crime tables and download final file

```
DROP TABLE IF EXISTS la_population_and_crime;

CREATE TABLE la_population_and_crime AS
SELECT
   SUBSTRING(c.geo_id, LOCATE('US', c.geo_id) + 2) AS tract_id,
   c.year,
   CAST(p. primary_type AS INT) AS crm_cd,

   p.count,

   c.one_race,
   c.black_or_African_american,

   c.merican_indian_and_alaska_native,
   c.herokee_tribal_grouping,
   c.chippewa_tribal_grouping,
   c.navajo_tribal_grouping,
   c.sioux_tribal_grouping,
   c.asian,
   c.indian,
   c.chinese,
   c.ilipino,
   c.japanese,
   c.korean,
   c.vietnamese,
   c.ther_asian,
   c.native_hawaiian_and_other_pacific_islander,
   c.chamorro,
   c.native_hawaiian,
   c.samoan,
   c.ther_native_hawaiian_and_other_pacific_pslander,
   c.race_alone_or_in_combination_with_one_or_more_other_races_total_population,
   c.hispanic_or_latino_total_population,
   c.ispanic_or_latino_total_population_of_any_race,
   c.mexican,
   c.puerto_rican,
   c.cuban,
   c.other_hispanic_or_latin,

   'Los Angeles County' AS area_name
FROM
  original_la_population c
LEFT JOIN
  la_tractid p
ON
  c.year = p.year
  AND SUBSTRING(c.geo_id, LOCATE('US', c.geo_id) + 2) = p.tract_id
```

```
ORDER BY
  c.year;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
la_population_and_crime LIMIT 10;
```

- **Add new column with crime description:**

```
DROP TABLE IF EXISTS la_population_with_crime_description;

CREATE TABLE la_population_with_crime_description AS
SELECT t1.*,
REGEXP_REPLACE(t2.crm_cd_desc, ',', '') AS description
FROM
   la_population_and_crime t1
LEFT JOIN
  crime_description_la t2
ON
  t1.crm_cd = t2.crm_cd
ORDER BY t1.year;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
la_population_with_crime_description LIMIT 10;
```

- Clean up our output table.

```
DROP TABLE IF EXISTS final_la_population_and_crime;

CREATE TABLE final_la_population_and_crime
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/dkim171/Project/population/final/la'

AS
SELECT *
FROM la_population_with_crime_description
WHERE
   crm_cd IS NOT NULL
   AND count IS NOT NULL;
```

- Check if the table was successfully created:

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

```
0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
final_la_population_and_crime LIMIT 10;
```

- Open Linux terminal and check file:

hdfs dfs -ls /user/dkim171/Project/population/final/la

- Remove previous file from the Linux server

```
-bash-4.2$ rm 000000_0
```

- Get new file from the HDFS to the Linux server:

hdfs dfs -get /user/dkim171/Project/population/final/la/000000_0

- Exit from the Linux server and enter command in terminal to download file:

scp dkim171@129.146.230.230:/home/dkim171/000000_0 ~/Desktop/

Join Chicago Population and Crime tables and download final file

```
DROP TABLE IF EXISTS chicago_population_and_crime;
CREATE TABLE chicago_population_and_crime AS
SELECT
   SUBSTRING(c.geo_id, LOCATE('US', c.geo_id) + 2) AS tract_id,
   c.year,
   p. primary_type,
   p.count,
   c.one_race,
   c.black_or_African_american,
   c.merican_indian_and_alaska_native,
   c.herokee_tribal_grouping,
   c.chippewa_tribal_grouping,
   c.navajo_tribal_grouping,
   c.sioux_tribal_grouping,
   c.asian,
   c.indian,
   c.chinese,
   c.ilipino,
   c.japanese,
   c.korean,
```

```
    c.vietnamese,
    c.ther_asian,
    c.native_hawaiian_and_other_pacific_islander,
    c.chamorro,
    c.native_hawaiian,
    c.samoan,
    c.ther_native_hawaiian_and_other_pacific_pslander,
    c.race_alone_or_in_combination_with_one_or_more_other_races_total_population,
    c.hispanic_or_latino_total_population,
    c.ispanic_or_latino_total_population_of_any_race,
    c.mexican,
    c.puerto_rican,
    c.cuban,
    c.other_hispanic_or_latin,
    'Cook County' AS area_name
FROM
  original_chicago_population c
LEFT JOIN
   chicago_tractid p
ON
   c.year = p.year
   AND SUBSTRING(c.geo_id, LOCATE('US', c.geo_id) + 2) = p.tract_id
ORDER BY
   c.year;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
chicago_population_and_crime LIMIT 10;
```

- Clean up our output table.

```
DROP TABLE IF EXISTS final_chicago_population_and_crime;

CREATE TABLE final_chicago_population_and_crime
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/dkim171/Project/population/final/chicago'

AS
SELECT *
FROM chicago_population_and_crime
WHERE
   primary_type IS NOT NULL
   AND count IS NOT NULL;
```

- Check if the table was successfully created:

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

0: jdbc:hive2://bigdaiun0.sub03291929060.trai> select * from
final_chicago_population_and_crime LIMIT 10;
```

- Open Linux terminal and check file:

hdfs dfs -ls /user/dkim171/Project/population/final/chicago

- Remove previous file from the Linux server

```
-bash-4.2$ rm 000000_0
```

- Get new file from the HDFS to the Linux server:

hdfs dfs -get /user/dkim171/Project/population/final/chicago/000000_0

- Exit from the Linux server and enter command in terminal to download file:

scp dkim171@129.146.230.230:/home/dkim171/000000_0 ~/Desktop/

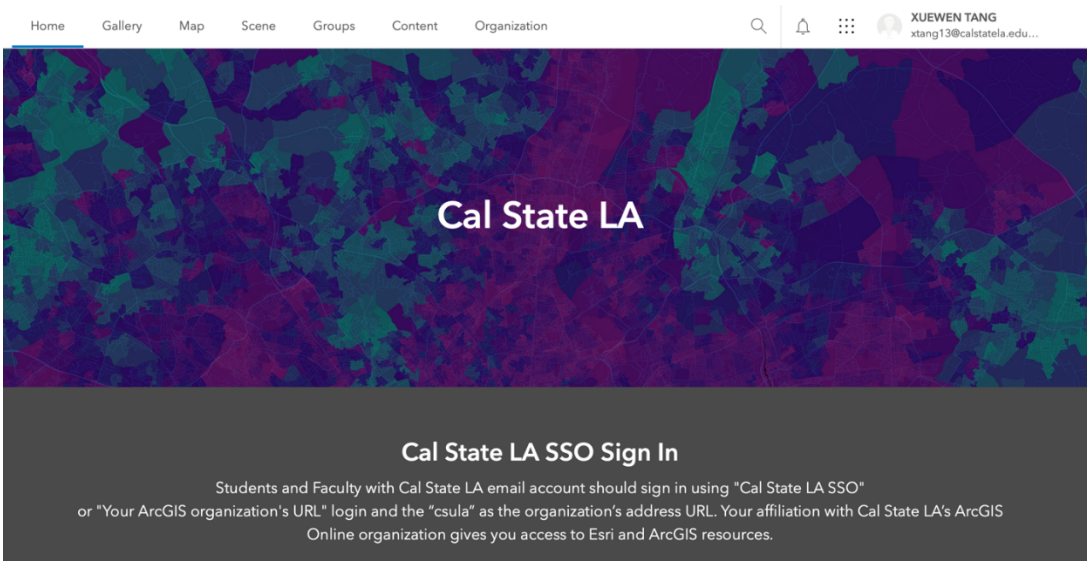
STEP 8: ArcGIS Online Tutorial

Access Your Account

1. Login with Your Organization Account:

- Navigate to ArcGIS Online: https://csula.maps.arcgis.com/home/index.html
- Select **"Cal State LA SSO"** or **"Your ArcGIS organization's URL"**.
- Enter **"csula"** as the organization's address URL.

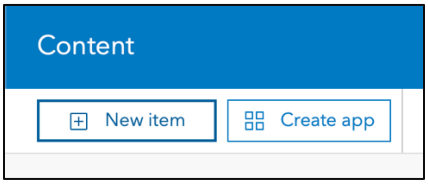
Example username: xtang13@calstatela.edu. You need to change the user domain name to your own.



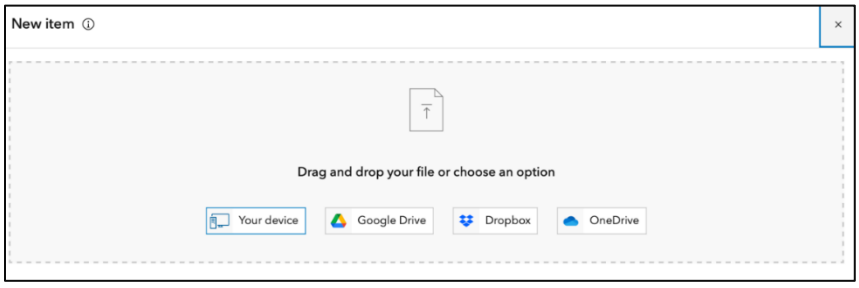
Create feature layer for Chicago and LA:

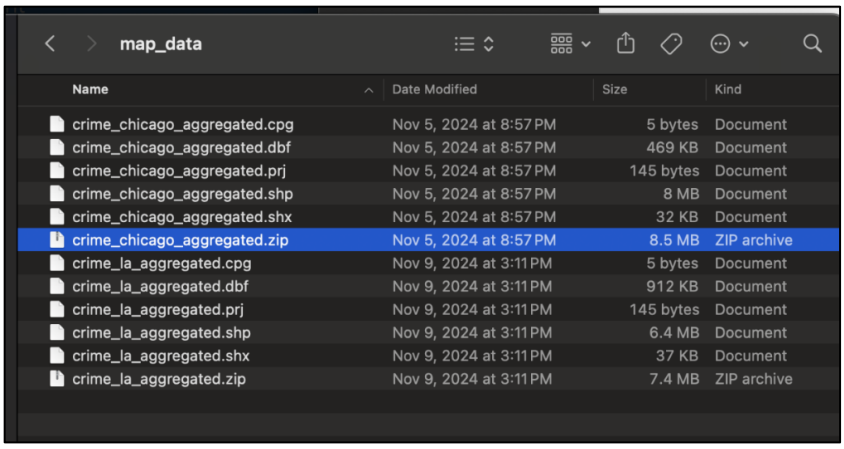
1. Open the **"Content"** section from the navigation bar.

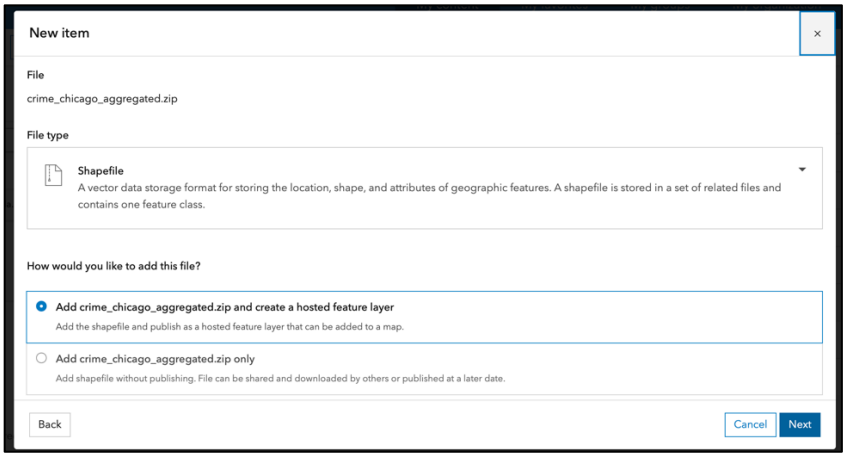
2. Click "New Item" Button.

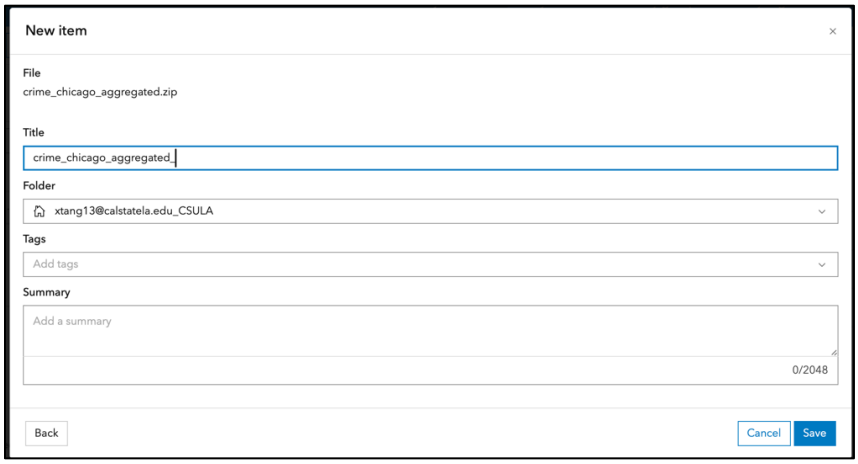


3. Upload the aggregate file "crime_chicago_aggregated.zip" in the map data folder. Then click "Next", Title keeps the same as the file name. Tags: crime, Chicago and click "Save".

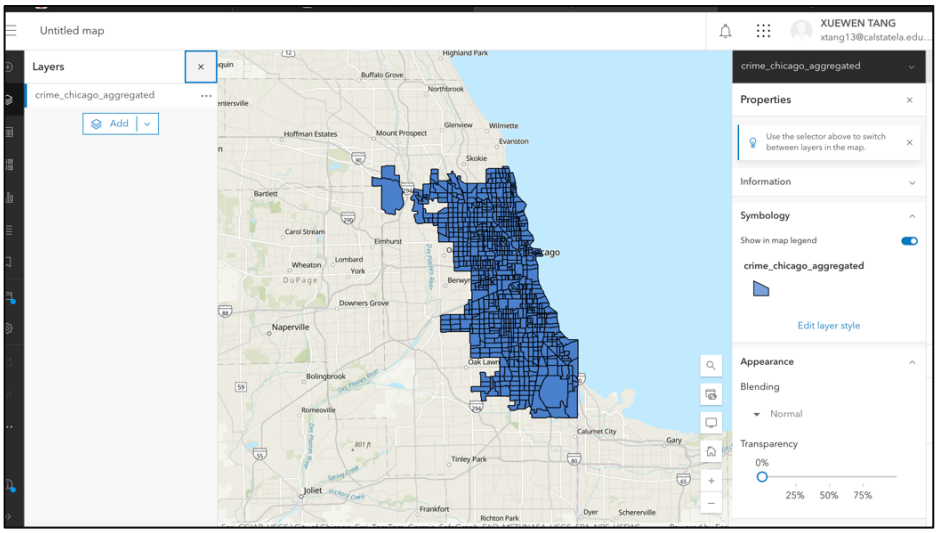






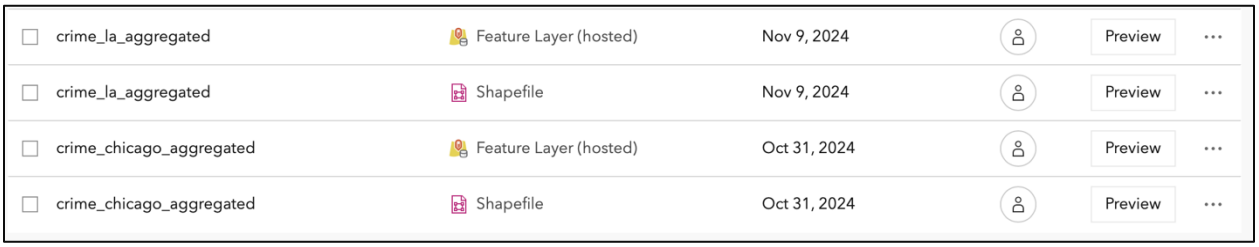


Then wait for the page processing the file, you will get the map as below:



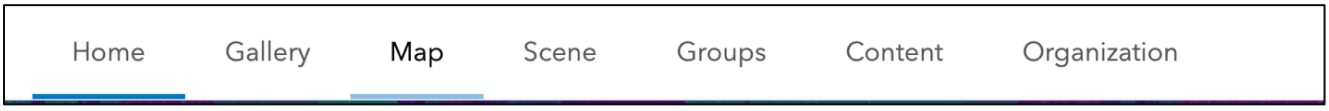
4. Repeat above steps to create the feature layer for LA.

Then you can get two feature layers ready for Chicago and LA in my content.



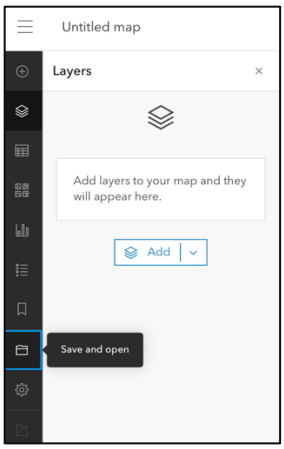
Create a Web Map

1. Open the **"Map"** section from the navigation bar.



2. Save the map:

- Click **"Save As"**.



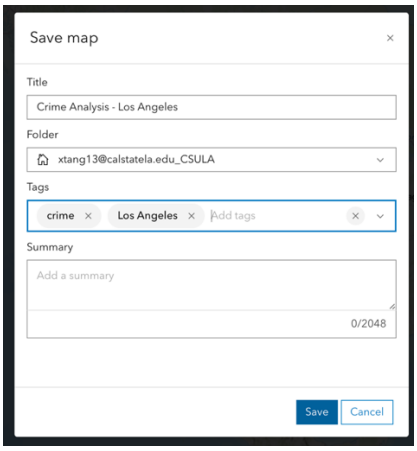
- Complete the details:

Title: Crime Analysis – Los Angeles

Folder: Default (no changes needed)

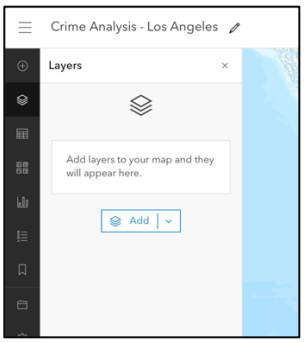
Tags: Crime, Los Angeles

Click **“Save”**.

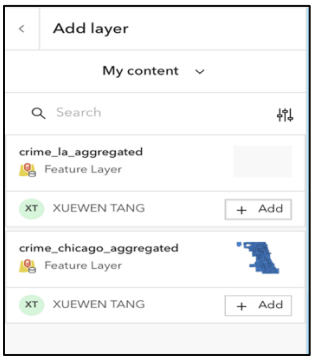


3. Add and Configure Layers

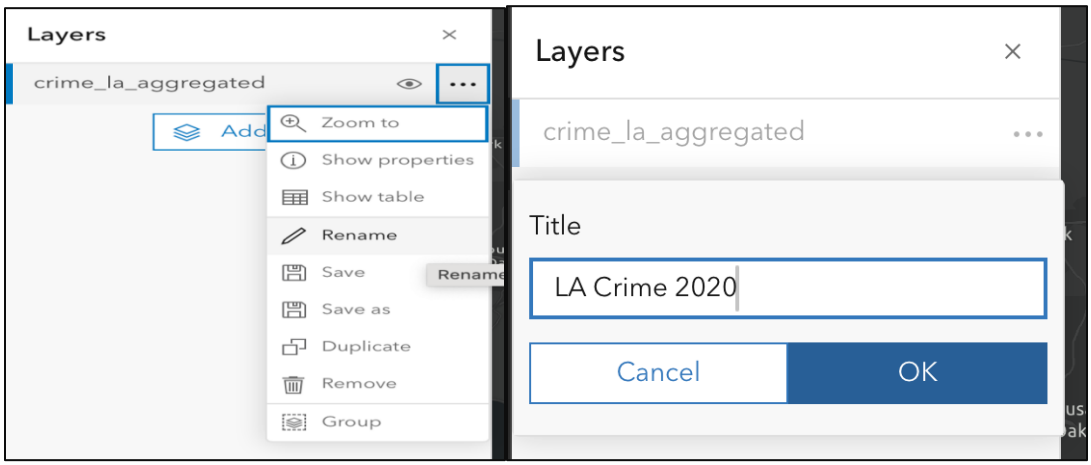
- **Add a Layer**:
  -Click **“Add”**.



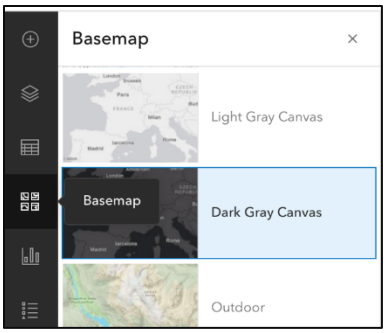
-From **My Content**, select crime_la_aggregated and click **"+ Add"**.



-Rename the layer to **"LA Crime 2020"** for clarity.

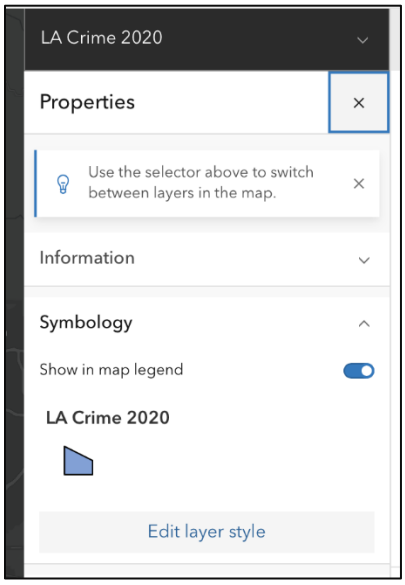


- **Set Basemap**:
  -Click **"Basemap"** and select **"Dark Grey Canvas"**.



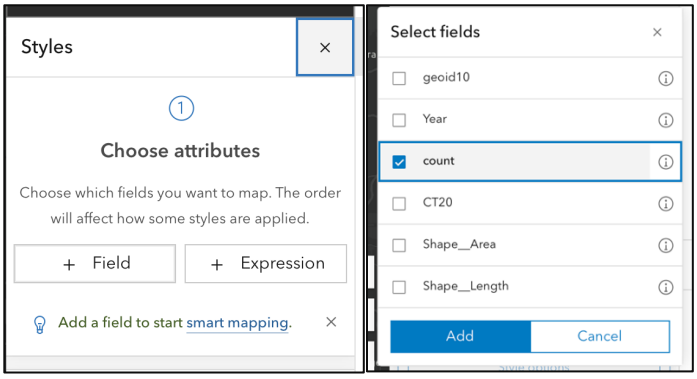
- **Edit Layer Style**:

- Click Properties and click **“Edit Layer Style”**.



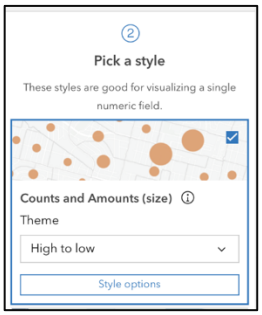
- Choose attributes:

Click **”+ Field”** and select count for the legend.

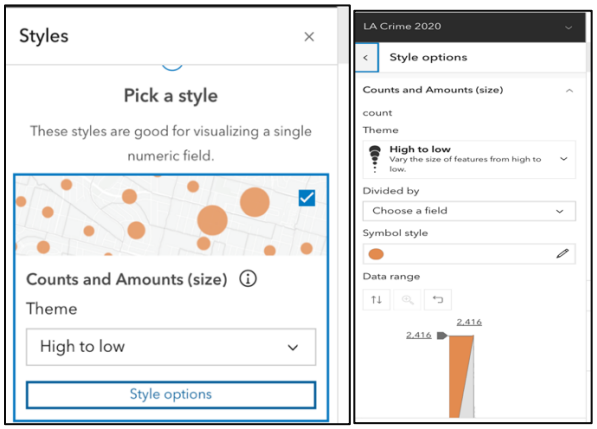


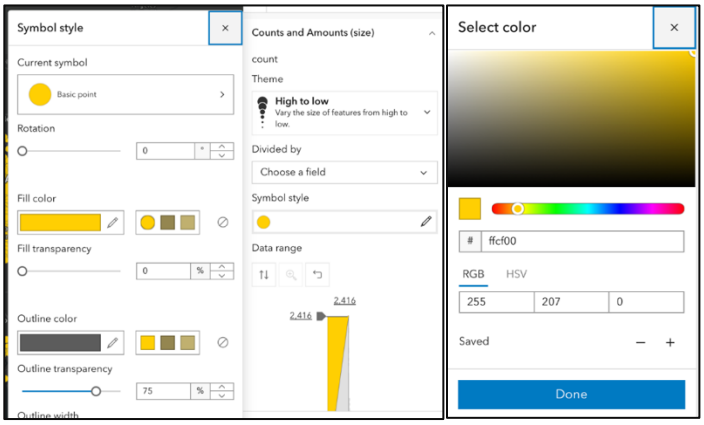
- Pick a style:

Use **“Counts and Amounts (size)”**.



- Adjust options: Click “**Style options**”--“**Symbol style**” edit.





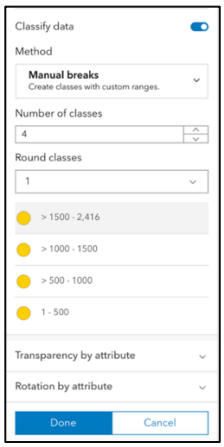
Bubble size: Set from 4 to 40.

Bubble color: **Fill**: #ffcf00. **Outline**: #2b2406.

- Modify data range: Open Classify data

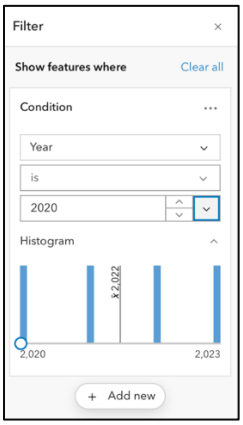
Use **Manual breaks** method with 4 classes and round classes to 1.

Set ranges as: 1–500, 500–1000, 1000–1500, 1500–maximum.



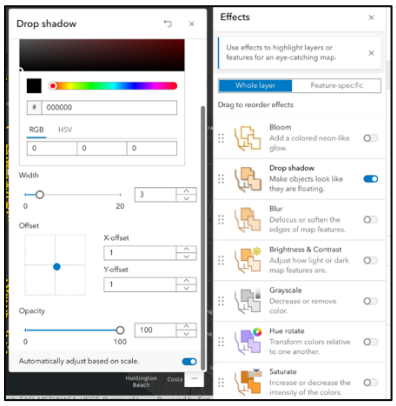
4. **Apply Filters**:

- Click **"Filter"** and set **Year = 2020**.



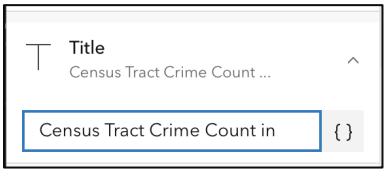
5. **Enable Effects**:

- Click **"Effects"** and enable **Drop shadow**:
- X-offset = 1, Y-offset = 1.

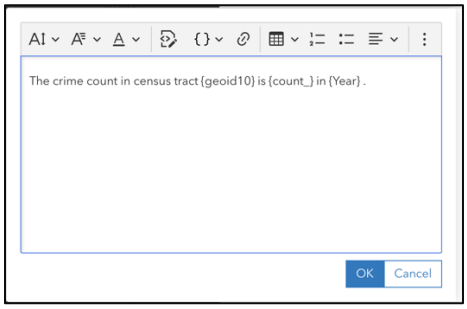


6. **Customize Pop-Ups**:

- Set the title: Census Tract Crime in {Year}. set “Year” as variable



- Delete default content:
- Add text: The crime count in census tract {geoid10} is {count_} in {Year}.

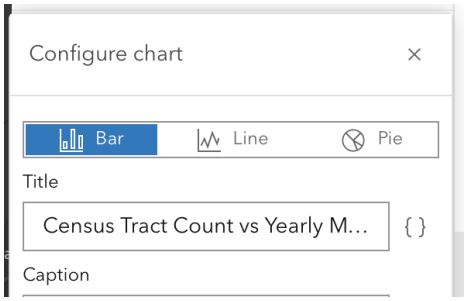


- Add a bar chart media: select “Bar”

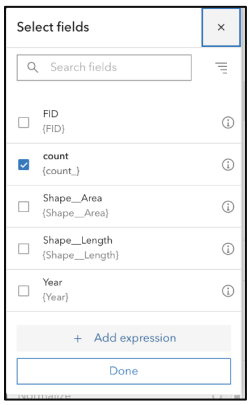
  Title: Comparison Analysis.

  - Edit media content:

    Title: Census Tract Count vs Yearly Maximum

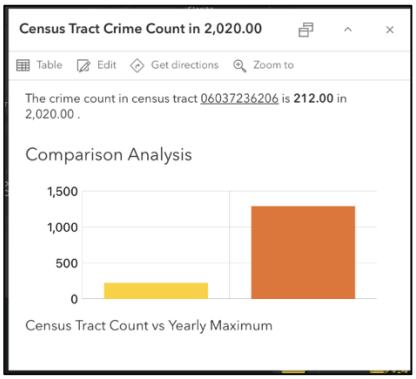


- Select fields: Count



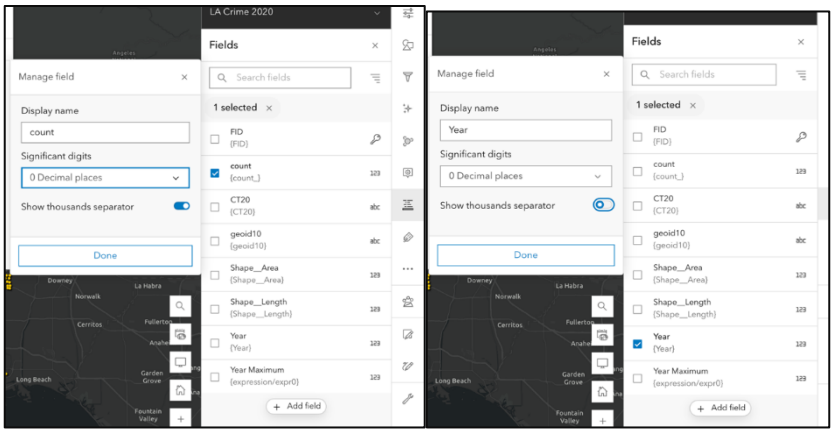
- Add Expression: **Name**: Year Maximum. Formula: Max($layer, "count_").

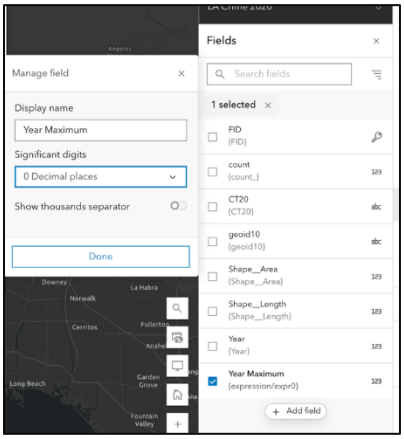
Then you can get the pop-up as below and you can find the display problem in count and year, there is decimal in the number.



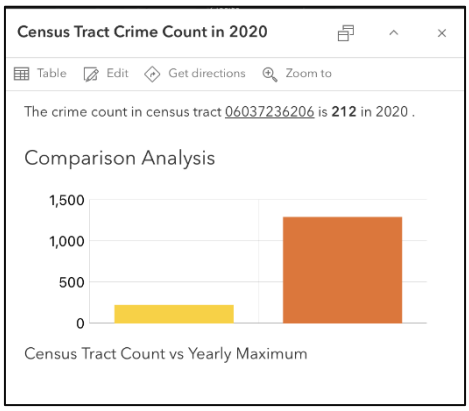
7. **Format Fields**:

- Click “Fields”, select “count”, change decimal to 0 and show thousands separator open. Select “Year” and “Year Maximum”, change decimal to 0 and close thousands separator.





Then you can see the pop-up changed.

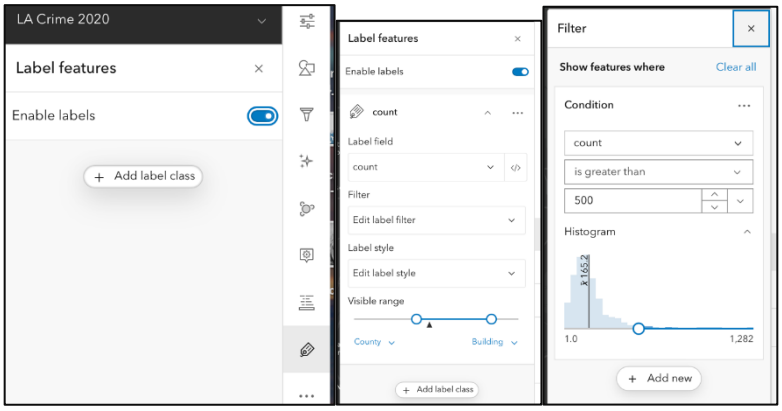


8. **Label Layers**:

Cilck “Labels” to add label class. Click **“+Add label class”,** change label field as “count” and make visible range from County to Building.

- Add labels: **Field**: count. **Visible Range**: From County to Building.

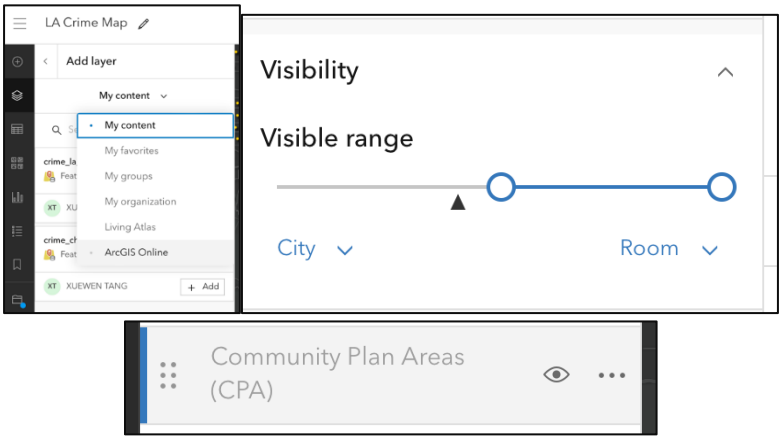
- Filter labels: Edit label filter: click “Add new”, set condition “count” “is greater than””500”



Add Supporting Layers

1. **Community Plan Areas**:

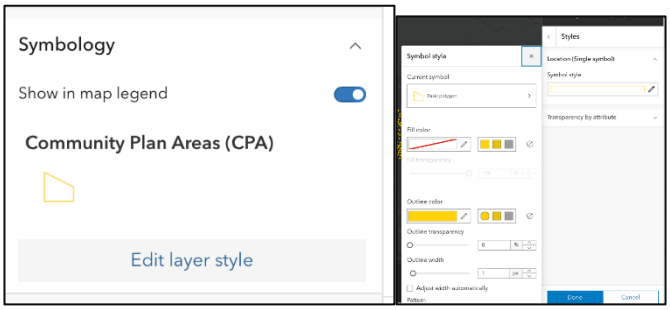
- Search **“Community Plan Areas (CPA)”** in **ArcGIS Online**.
- Add the layer and move it to the bottom.
- Open Properties, Set visibility range: City to Room.



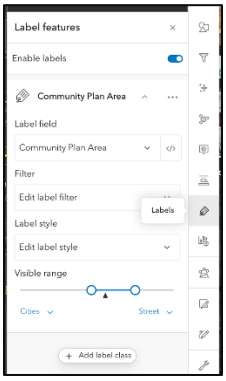
- Style:

Edit Layer style: pick the Location (single symbol) style.

Open style options: edit the symbol style, set the Fill color to “no color” and set the outline color to yellow.



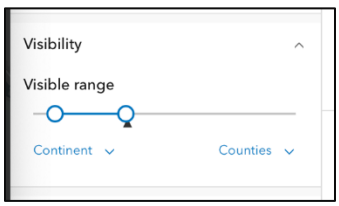
- Label Features: Click “Labels”. Select Label field as “Community Plan Area”, set the visible range from Cities to Street.



2. **LA City Council Districts**:

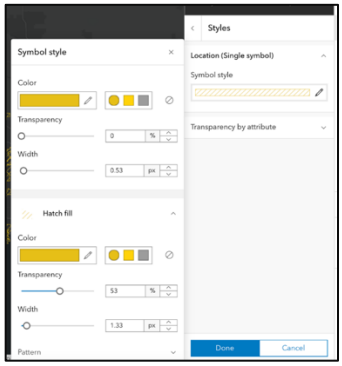
- Search **“LA City Council Districts (Adopted 2021)”** in **ArcGIS Online**.
- Add the layer and move it to the bottom.

- Open Properties , set visibility range: Continent to Counties.

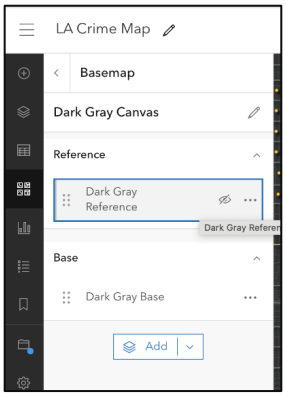


- Style: Edit Layer style: pick the Location (single symbol) style.

Open style options: edit the symbol style, set the **Solid stroke** to “yellow” and select the **Hatch fill** and set the color to “yellow”. **Transparency** set to 53%.



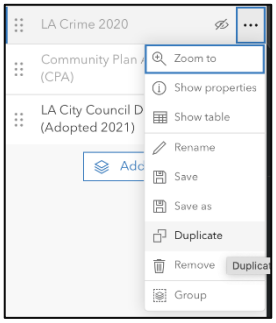
Close Basemap Reference



Duplicate for Additional Years

1. Duplicate **LA Crime 2020**:

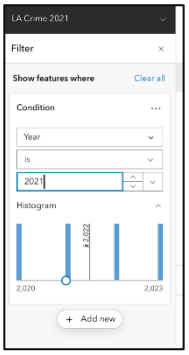
- Click **”…”** next to the layer and select **“Duplicate”**.



2. Rename and Filter:

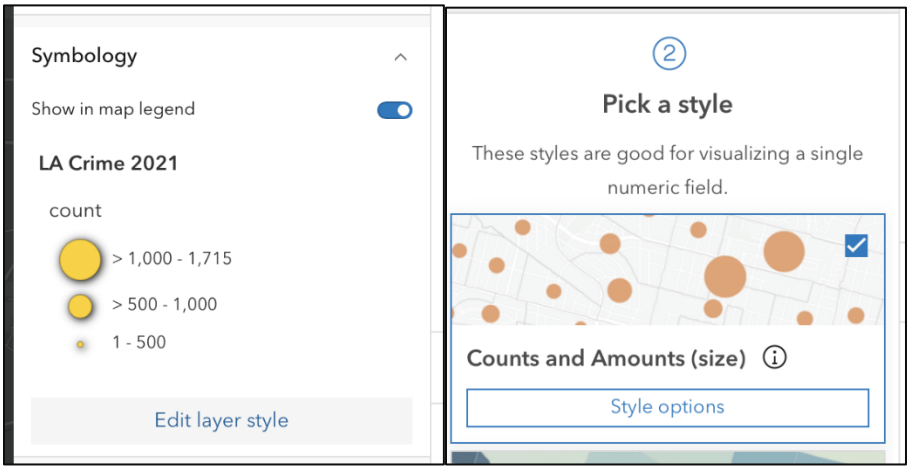
- Rename the layer to **LA Crime [Year]** (e.g., 2021).

- Update the filter to **Year = [Year]**.



3. Adjust Style Options: Click “Properties”-“Edit Layer style”-“Style Options”.

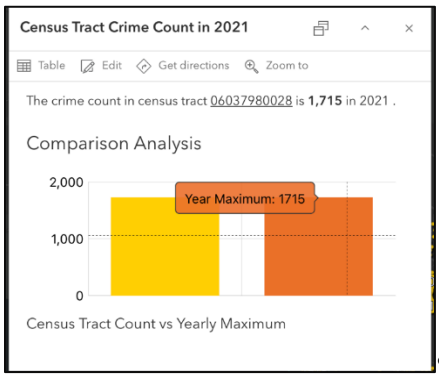
- Update the maximum data range based on the year’s max value.



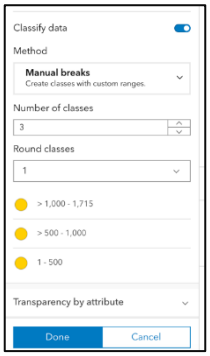
- For 2021, as example :Select any bubble to see the pop-up, you can find the maximum count number in this year, for year 2021 the maximum number is 1715.

- Max: 1715.

- Classes: 3.

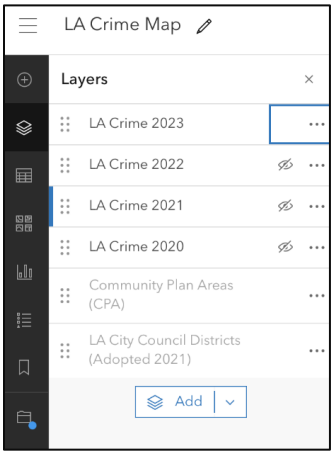


Ranges: 1–500, 500–1000, 1000–1715.



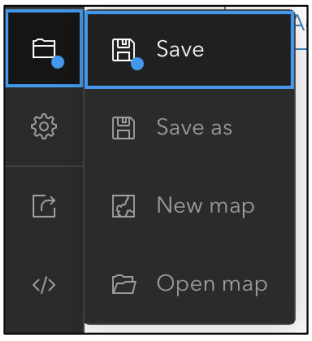
Repeat for 2022 and 2023.

Then you will finish the LA Crime Map as below:

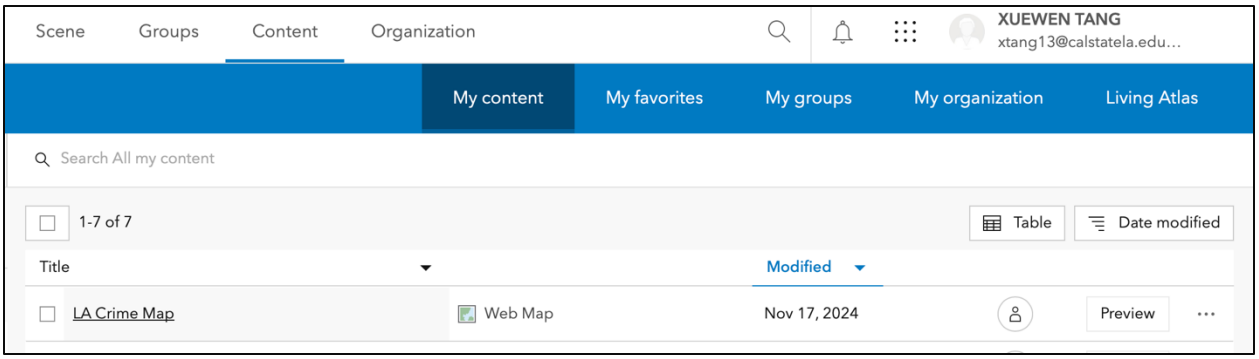


Save and Access Your Map

1. Click **"Save"** to preserve changes.



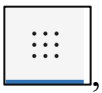
2. Find your web map in **"My Content"**.

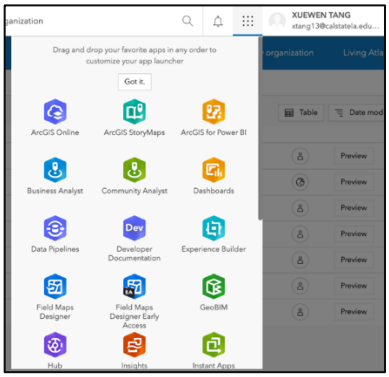


Repeat the create web map step of LA above, create the Chicago Crime Map.

Create a Map Story

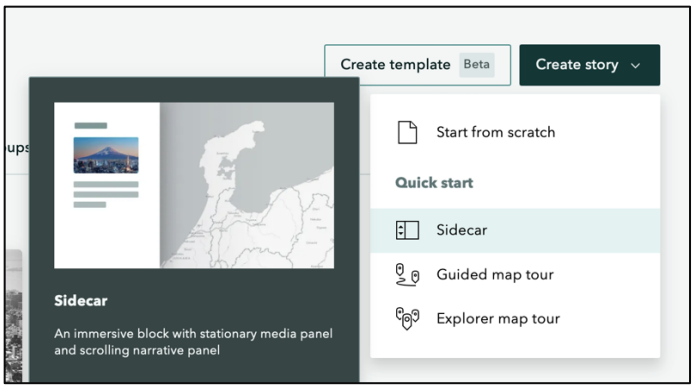
1. Go back to : https://csula.maps.arcgis.com/home/index.html
2. Go to "Content" on the navigation bar.

3. Click the extend  , Select " ArcGIS StoryMaps"

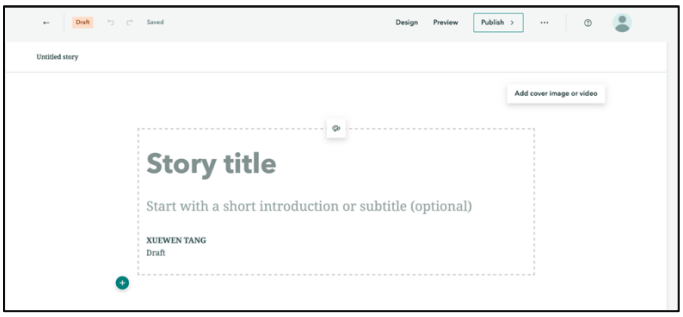


4. Click "**Create Story**"

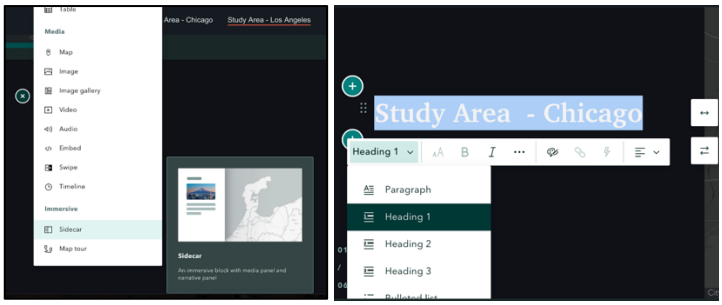
5. You can select the template or a blank scratch.



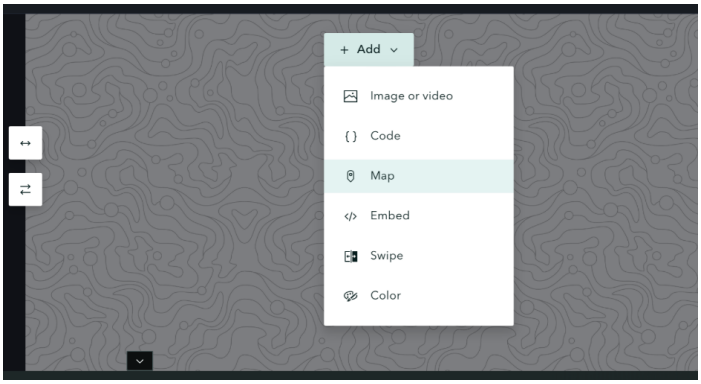
6. Edit your Story title, subtitle. You can upload cover image or video.



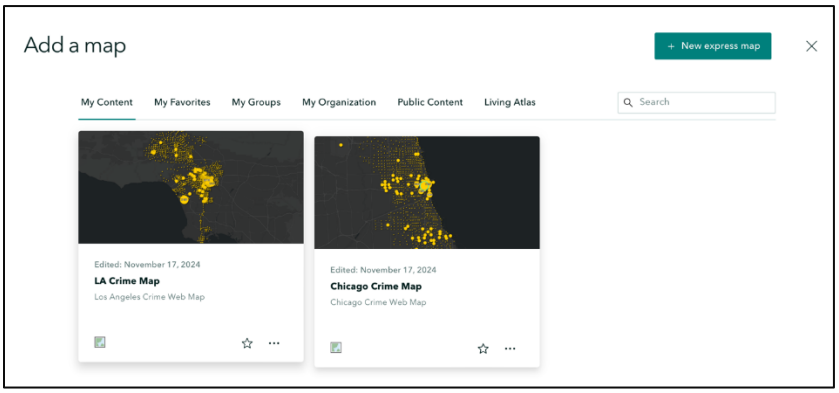
7. Add sidecar
8. Heading: Study Area – Chicago. Double click the text and change paragraph to Heading 1.



9. Click ”+Add” at the right side, select “Map”.



10. You will see the web map we created; it can be select as below: choose Chicago Crime Map

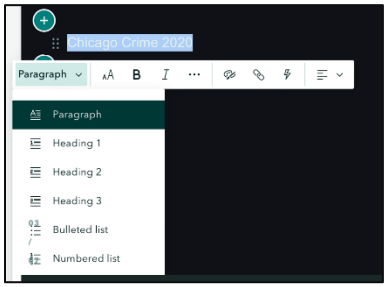


11. Set the Layer: City of Chicago’s Boundaries, as visible, set other layers as invisible.

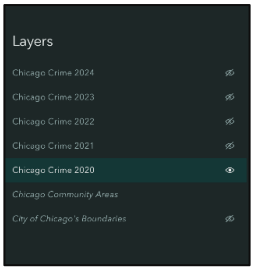
12. Click “New slide” in the bottom Sidecar. Add new slide page.

13. Repeat the same step as the first page:

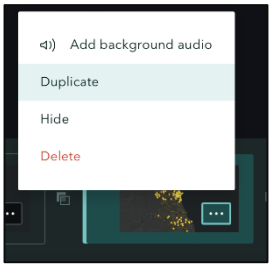
    Title: Chicago Crime 2020; Paragraph



14. Add Map: Set the Layer—Chicago Crime 2020; Chicago Community Areas, as visible, set other layers as invisible.

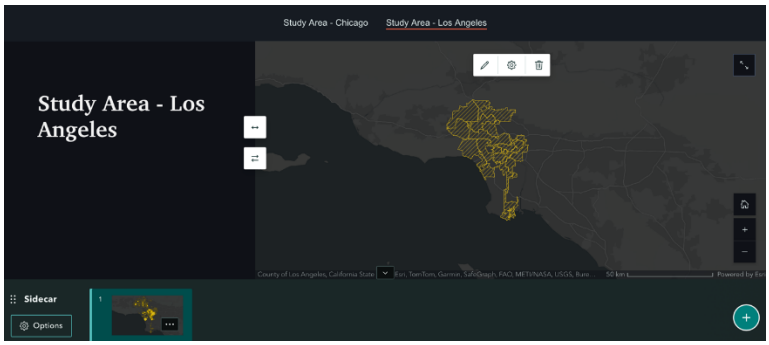


15. You can use Duplicate to add Chicago Crime 2021/2022/2023/2024. Rember to change the year for each slides.



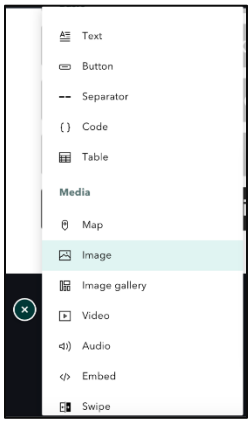
16. **Add sidecar for Los Angeles:**

    Heading: Study Area – Los Angeles. Double click the text and change paragraph to Heading 1.



Add slide and repeat the same steps as Chicago to add slides for year 2020/2021/2022/2023.

17. You can export PPT slides as PNG image, import the PPT slide to the map story.

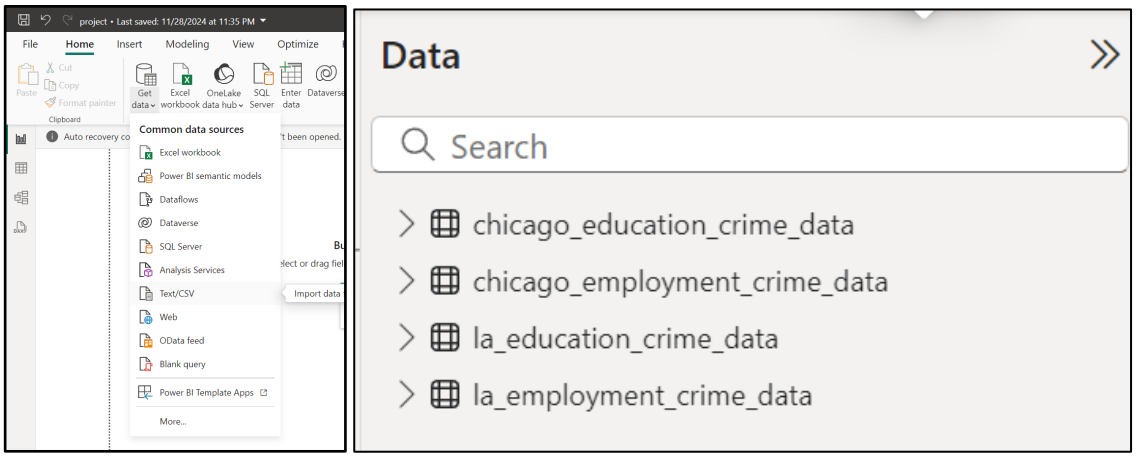


Step 9: Visual Analysis of Los Angeles and Chicago Crime

Topic: Employment and Crime

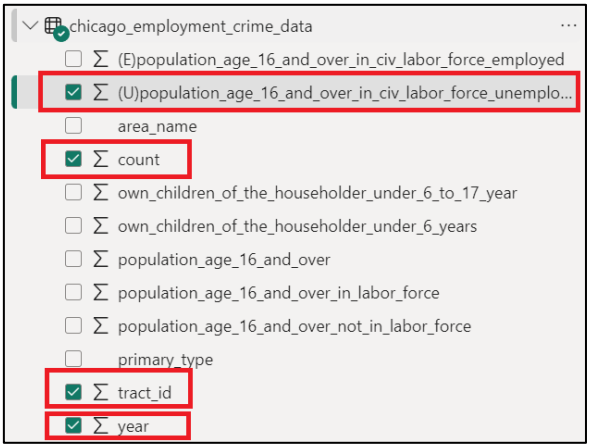
Download csv files from Hadoop, in this case, chicago_employment_and_crime and la_education_and_crime as csv files to local computer.

1. Open PowerBI → Home tab → Get Data → text/CSV → (upload all downloaded tables to PowerBI) shown as below



For Chicago's employment and crime dataset

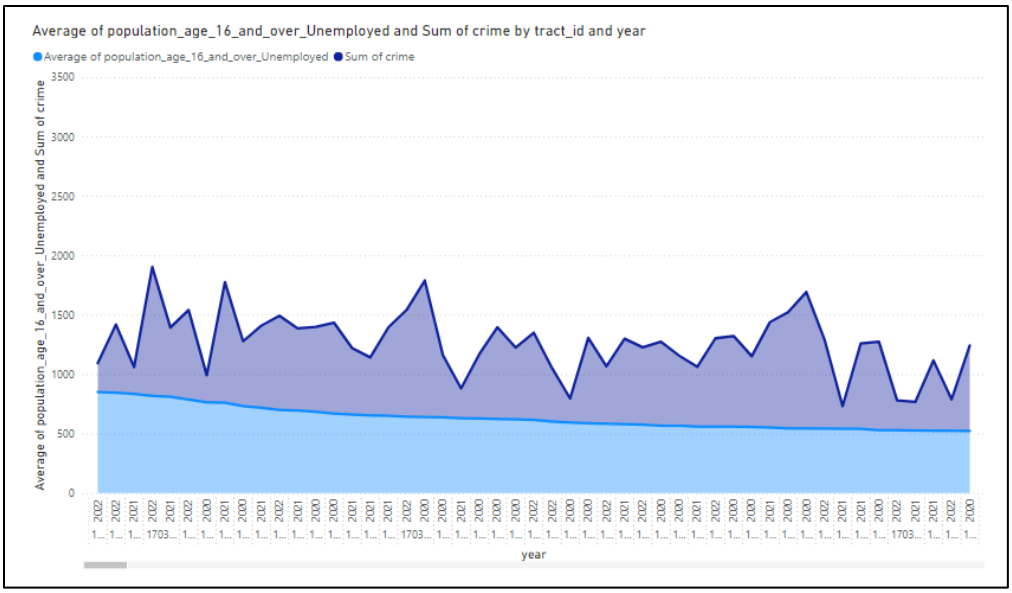
2. Select stacked area chart as visualization.



Select

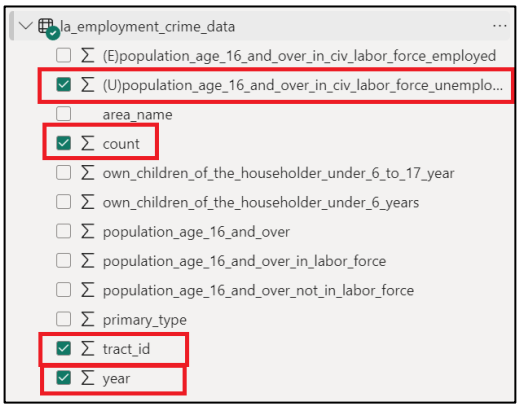
- tract_id and year for X-axis,
- [sum] count and [average] (U)population_age_16_and_over_in_civ_labor_force_unemployed for Y-axis

Results should be similar or equivalent as chart below.



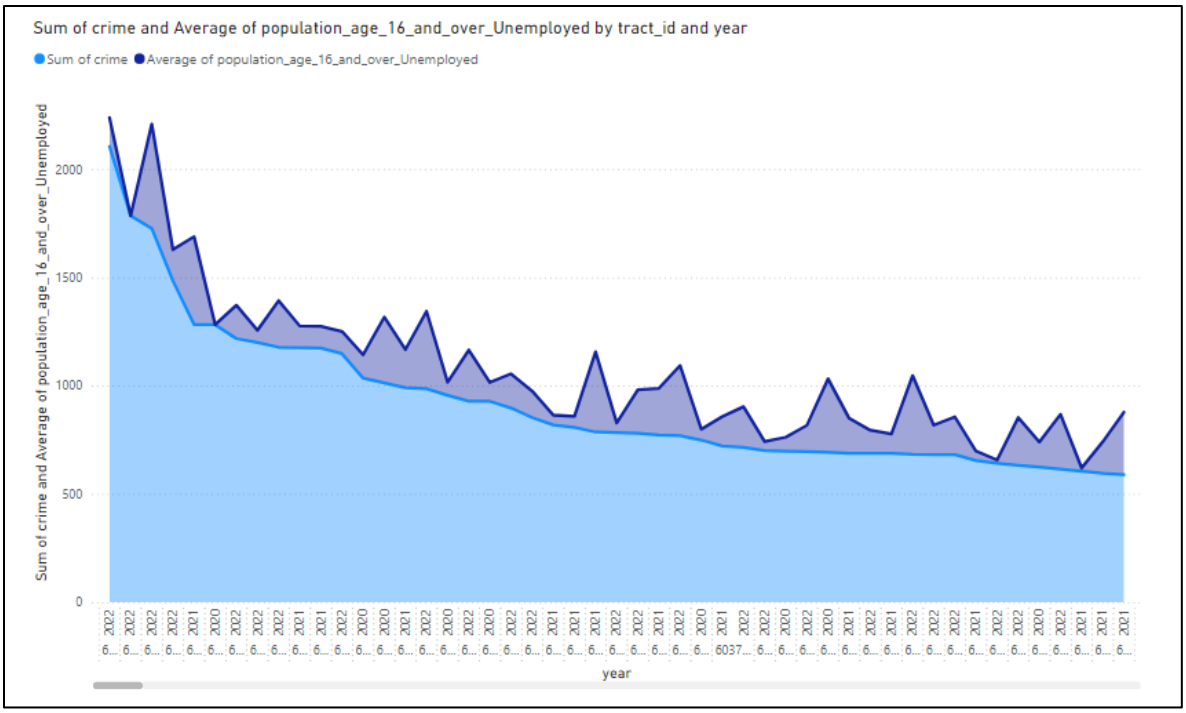
For Los Angeles's employment and crime dataset

3. Select line chart as visualization



Select

- tract_id and year for X-axis,
- [sum] count and [average] (U)population_age_16_and_over_in_civ_labor_force_unemployed for Y-axis

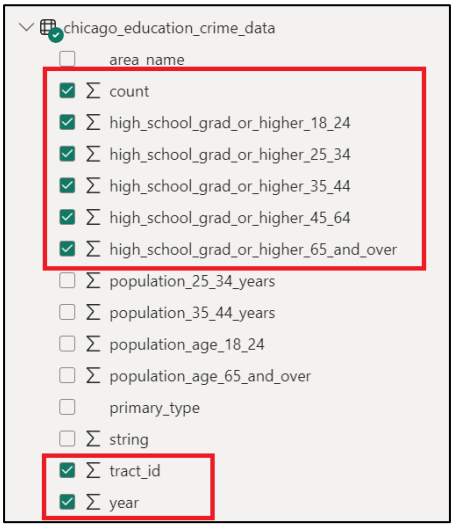


Topic: Education and Crime

<u>For Chicago's education and crime dataset</u>

1. Select line chart as visualization.

   Under employment and crime dataset, select all fields listed in the picture below.

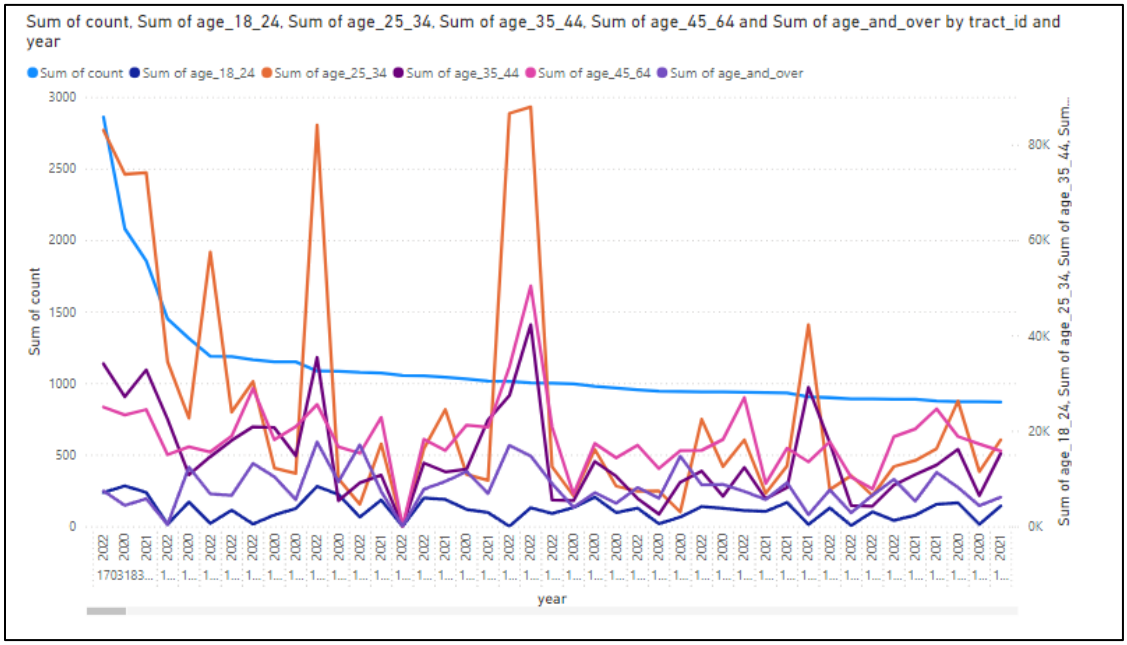


Select

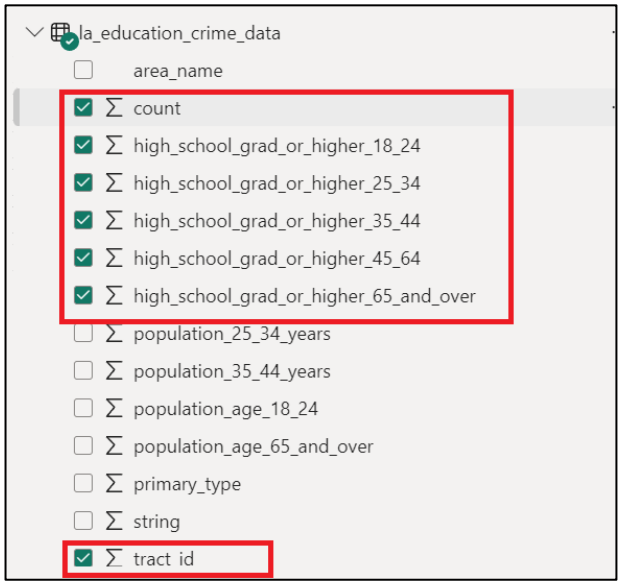
- tract_id and year for X-axis,

- sum of count for Y-axis,
- [sum] high_school_grad_or_higher_18_24 and the rest of the checkboxes for Secondary Y-axis

As a result, the chart should show up in this format shown below. Apply the same methodology to LA's education and crime dataset.

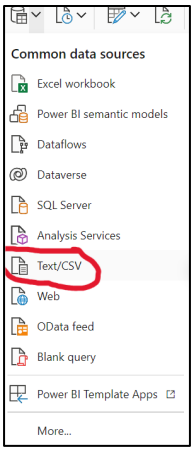


Apply the method for la_education_crime_data

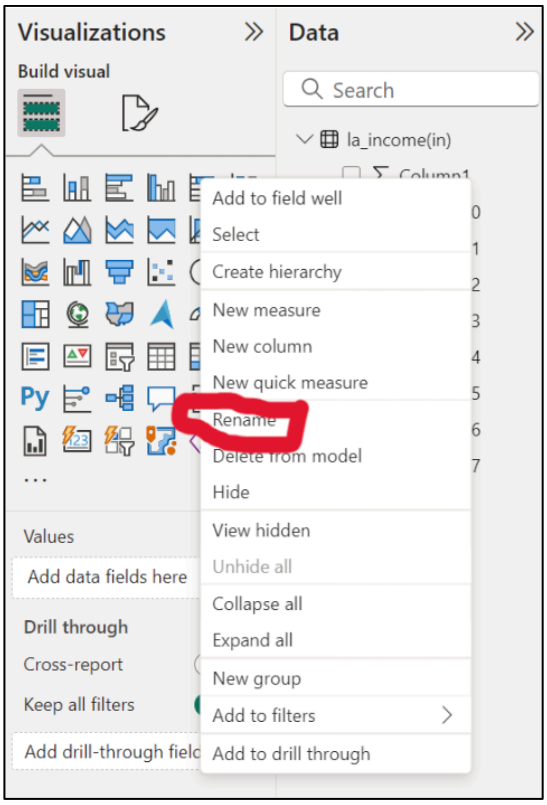


Topic: LA Income and Crime

Step 1: Set Up Power BI Open Power BI Desktop. Click "Get Data" → Select "Text/CSV". Browse to your dataset file and load it into Power BI.

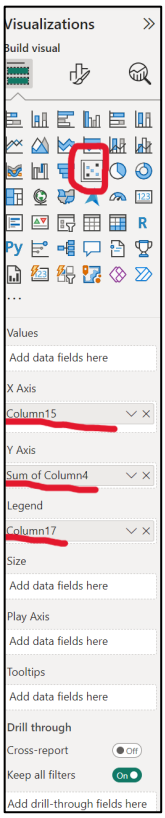


Step 2: Load and Prepare the Data simply click load
step 3: label all columns correspondingly:
COLUMN 1 = tract_id
COLUMN 2 = year
COLUMN 3 = crm_cd
COLUMN 4 = count
COLUMN 5 = less_than_10k
COLUMN 6 = from_10k_to_14999
COLUMN 7 = from_15k_to_24999
COLUMN 8 = from_25k_to_34999
COLUMN 9 = from_35k_to_49999
COULUMN 10 = from_50k_to_74999
COLUMN 11 = from_75k_to_99999
COLUMN 12 = from_100k_to_149999
COLUMN 13 = from_150k_to_199999
COLUMN 14 = from_200k_or_more
COLUMN 15 = median_income
COUMN 16 = area_name
COLUMN 17 = description

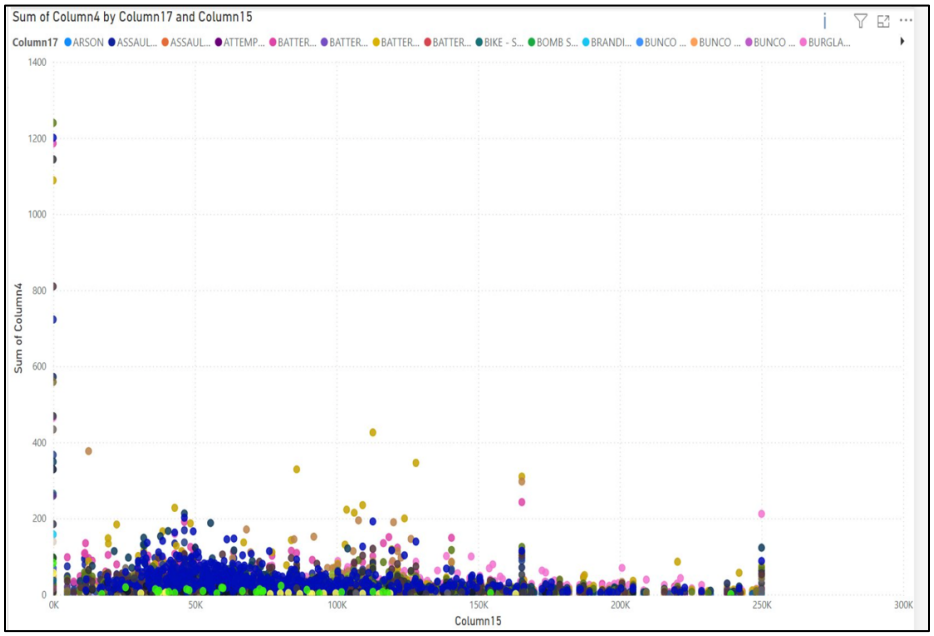


Step 4:

Create the Scatter Chart In the Report View, click on Scatter Chart from the Visualizations pane. Assign the fields to the scatter chart: X-Axis: Drag Column 15 (median_income). Y-Axis: Drag Column 4 (count). Legend: Drag Column 17 (description).

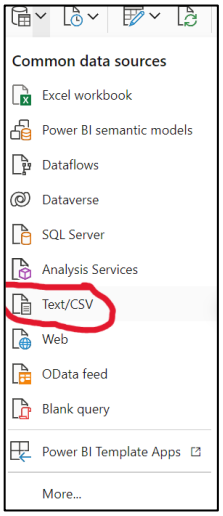


RESULTS:

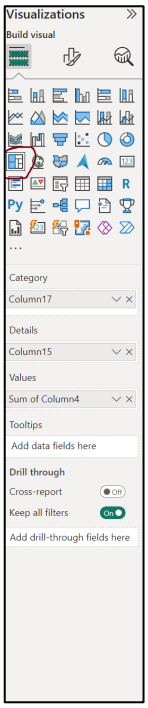


Treemap LA INCOME:

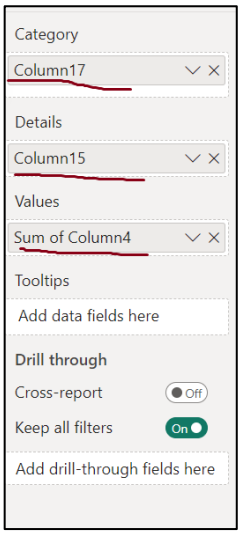
Step 1: Set Up Power BI Open Power BI Desktop. Click "Get Data" → Select "Text/CSV". Locate and import your dataset.



Step 2: Add a Treemap Visualization Navigate to the Report View in Power BI Desktop. From the Visualizations Pane, click on the Treemap icon to add it to your canvas.

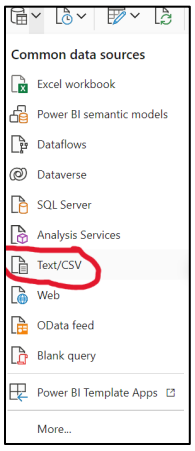


Step 3: Configure the Treemap Assign the following fields to the treemap:
Category: Drag Column 17: (description) into the Category field well.
Details: Column 15:
Values (Details): Drag Column 4 (median_income) into the Values field well.
The treemap will now show categories (from description) sized based on their corresponding median income values.



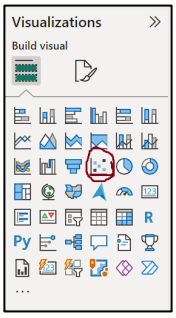
Topic: CHICAGO INCOME and CRIME:

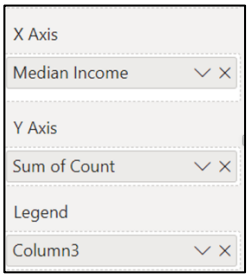
STEP 1: open PowerBI
STEP 2: create a blank document
STEP 3: in the GET DATA module under home
click text/csv
STEP 4: in the data preview window click load

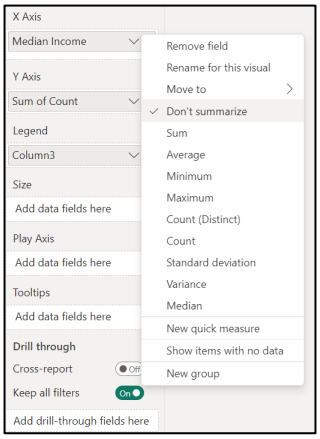


STEP 5:

Under Data Visualization Click the Scatter chart icon
STEP 6: Column 15 in the "Data" Section represents "median income data"
STEP 7: Drag Column 15 to the x-axis
STEP 8: Drag Column 4 into the Y axis portion this will represent the "Count" (frequency of crimes)

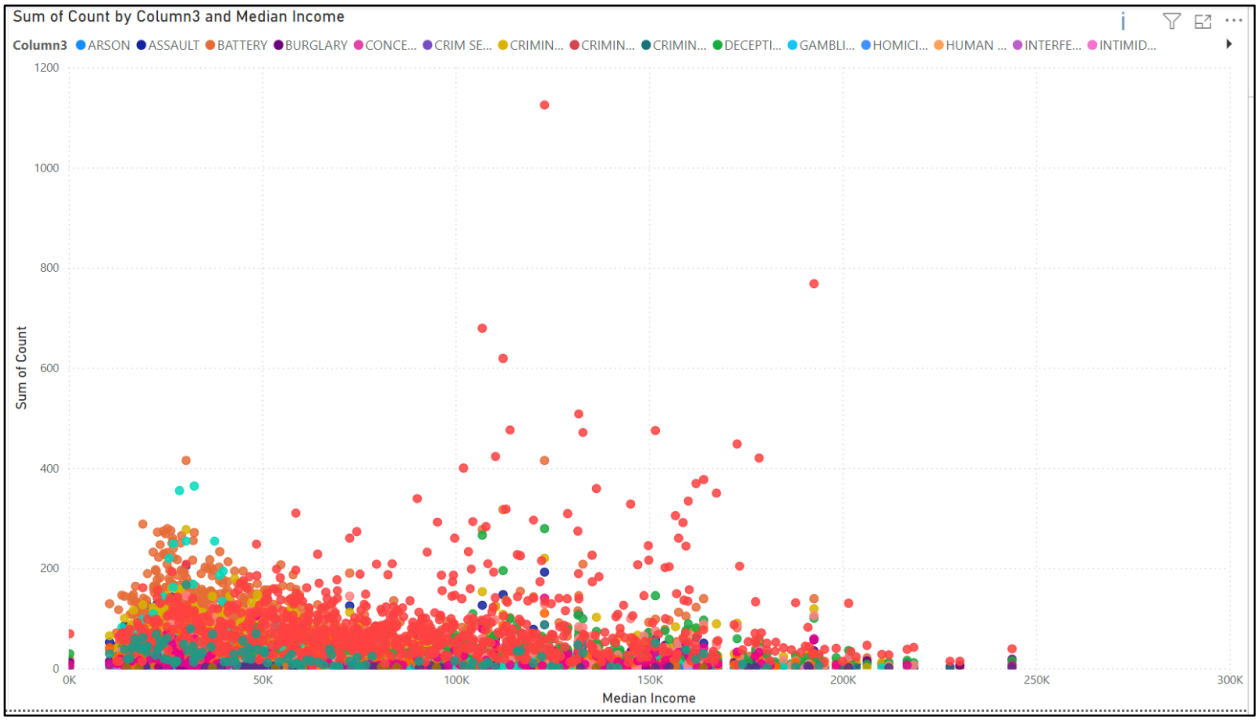






STEP 9: Under the x column click the drop down menu and put "do not Summarize"
This scatter plot show the positive correlation between income and the frequency of crime within Chicago. It signifies that everything from 100k and below has a significantly higher crime rate frequency.

RESULTS:



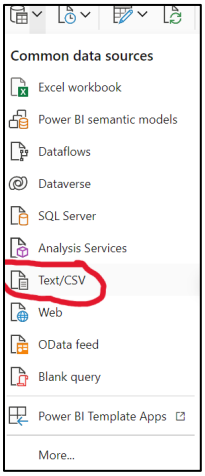
<u>TREEMAP MAP CHICAGO INCOME/CRIME:</u>

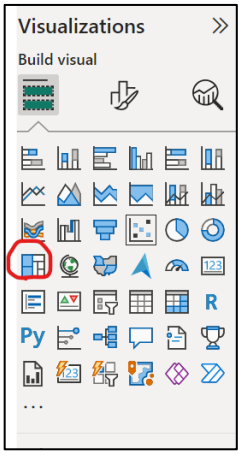
STEP 1: Open a new PowerBI Blank Document

STEP 2: in the GET DATA module under home click text/csv

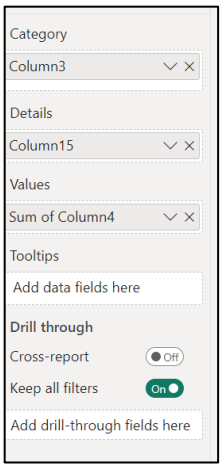
STEP 3: In the data preview window click load

STEP 4: In the Visualizations menu click "TreeMap" .

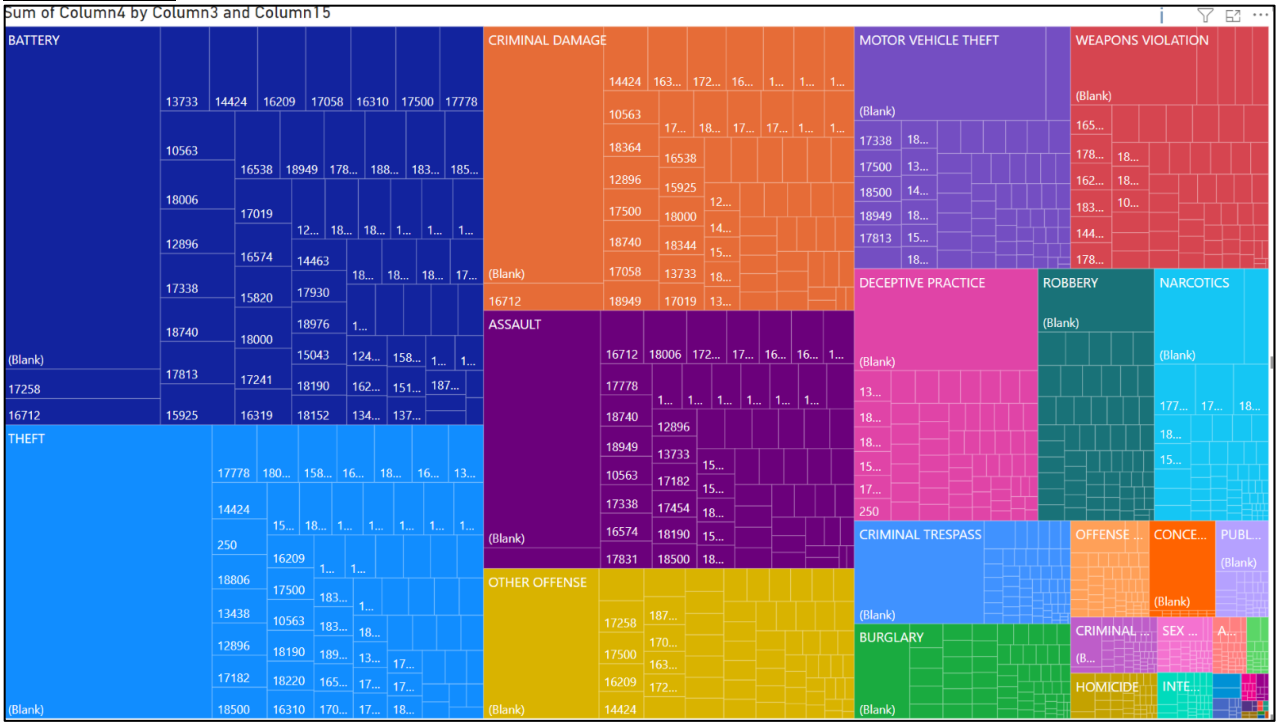




STEP 5:
Category: Drag "primary_type" (column 3) to the Category field. This will create the main groupings or segments in the treemap based on different crime types.
6)Values: Drag "count" (or the metric representing the number of crimes)(column 4) to the Values field. This determines the size of each segment in the treemap based on the crime count.
7)Details : (column 15), you can drag median_income into details section

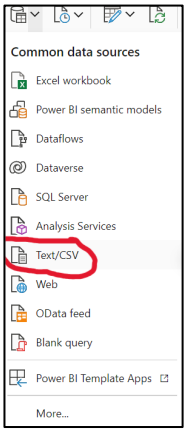


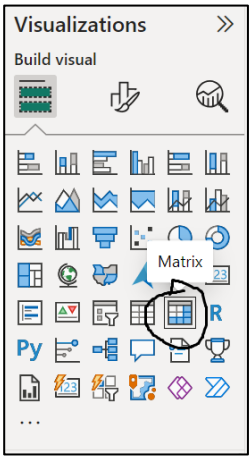
RESULTS:



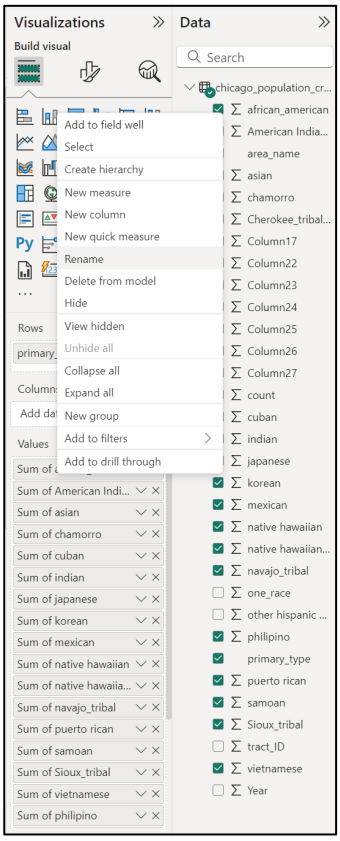
LOS ANGELES POPULATION HEAT MAP:

STEP 1: Open a new PowerBI Blank Document
STEP 2: in the GET DATA module under home click text/csv
STEP 3: In the data preview window click load
STEP 4: in the visualization, click the heat map symbol





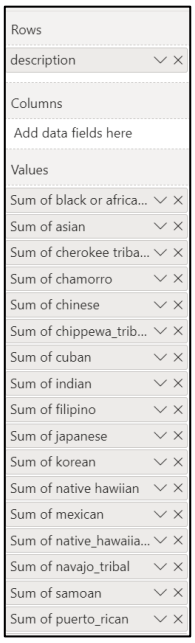
STEP 5: Label columns correspondingly
COLUMN 1 = tract_id
COLUMN 2 = year
COLUMN 3 = crm_cd
COLUMB 4 = count
COLUMN 5 = one_race
COLUMN 6 = black_or_african_american
COLUMN 7 = american_indian_and_alaska_native
COLUMN 8 = cherokee_tribal_grouping
COLUMN 9 = chippewa_tribal_grouping
COLUMN 10 = navajo_tribal_grouping
COLUMN 11 = sioux_tribal_grouping
COLUMN 12 = asian
COLUMN 13 = indian
COLUMN 14 = chinese
COLUMN 15 = filipino
COLUMN 16 = japanese
COLUMN 17 = korean
COLUMN 18 = vietnamese
COLUMN 19 = other_asian
COLUMN 20 = native_hawaiian_and_other_pacific_islander
COLUMN 21 = chamorro
COLUMN 22 = native_hawaiian
COLUMN 23 = samoan COLUMN 24 = other_native_hawaiian_and_other_pacific_islander COLUMN 25 = race_alone_or_in_combination_with_one_or_more_other_races_total_population
COLUMN 26 = hispanic_or_latino_total_population
COLUMN 27 = hispanic_or_latino_total_population_of_any_race
COLUMN 28 = mexican COLUMN 29 = puerto_rican
COLUMN 30 = cuban
COLUMN 31 = other_hispanic_or_latino
COLUMN 32 = area_name COLUMN 33 = description



STEP 6:

In the "row" section put the "description" here

STEP 7: Under the "values" put all the ethnicities column 6 - 31



Result:

description	Sum of black or african american	Sum of asian	Sum of cherokee tribal grouping	Sum of chamorro	Sum of chinese	Sum of chi
ARSON	281707	283988	600	951	49645	
ASSAULT WITH DEADLY WEAPON AGGRAVATED ASSAULT	680458	867343	1689	2565	151139	
ASSAULT WITH DEADLY WEAPON ON POLICE OFFICER	125056	86081	106	293	19734	
ATTEMPTED ROBBERY	431266	435576	818	1431	77369	
BATTERY - SIMPLE ASSAULT	700525	934257	1742	2664	164298	
BATTERY ON A FIREFIGHTER	26928	34030	87	23	7938	
BATTERY POLICE (SIMPLE)	238753	204791	277	1119	40236	
BATTERY WITH SEXUAL CONTACT	365527	437296	836	1256	81021	
BEASTIALITY CRIME AGAINST NATURE SEXUAL ASSLT WITH ANIM	1878	385	0	0	184	
BIGAMY	276	160	0	0	3	
BIKE - ATTEMPTED STOLEN	3481	3569	0	0	1381	
BIKE - STOLEN	300891	536331	916	1175	112158	
BLOCKING DOOR INDUCTION CENTER	630	430	9	0	31	
BOAT - STOLEN	10267	17986	23	13	2290	
BOMB SCARE	43466	68028	86	127	15709	
BRANDISH WEAPON	616639	724194	1385	2268	124432	
BRIBERY	915	790	0	0	85	
BUNCO ATTEMPT	73145	47777	65	303	8659	
BUNCO GRAND THEFT	427641	644046	1238	2049	114254	
BUNCO PETTY THEFT	252144	336545	698	1077	58854	
BURGLARY	690060	924309	1733	2572	163990	
BURGLARY ATTEMPTED	350017	475359	1180	1341	85387	
BURGLARY FROM VEHICLE	690944	922079	1697	2680	163985	
BURGLARY FROM VEHICLE ATTEMPTED	109360	119114	266	606	22392	
CHILD ABANDONMENT	7022	4788	0	0	1451	
CHILD ABUSE (PHYSICAL) - AGGRAVATED ASSAULT	132106	76771	103	435	10612	
CHILD ABUSE (PHYSICAL) - SIMPLE ASSAULT	391879	339549	801	1448	53479	
CHILD ANNOYING (17YRS & UNDER)	138774	148011	267	651	25614	
CHILD NEGLECT (SEE 300 W.I.C.)	173222	162414	417	427	26636	
CHILD PORNOGRAPHY	44954	44403	94	86	9930	
CHILD STEALING	116493	61369	265	244	10836	
CONSPIRACY	2631	3727	0	19	271	
CONTEMPT OF COURT	215852	119107	391	477	15471	
Total	**25048390**	**30043448**	**57980**	**90914**	**5334900**	

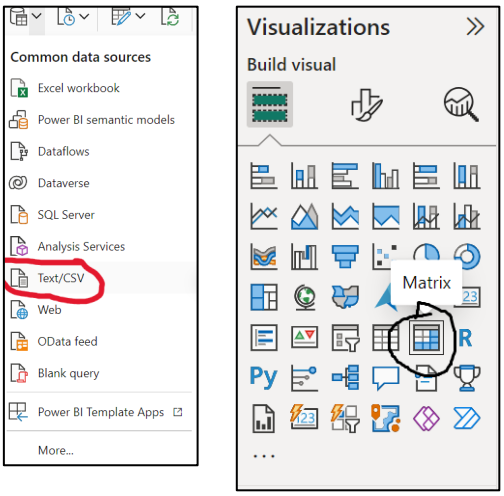
CHICAGO POPULATION HEAT MAP:

STEP 1: Open a new PowerBI Blank Document

STEP 2: in the GET DATA module under home click text/csv

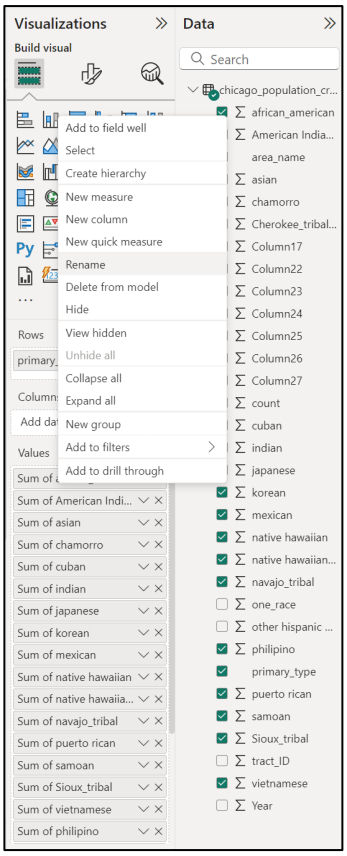
STEP 3: In the data preview window click load

STEP 4: in the visualization, click the heat map symbol



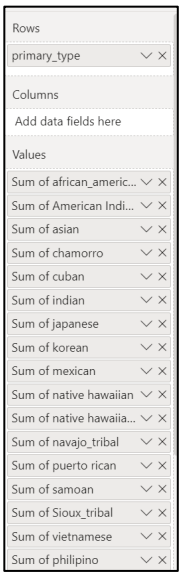
STEP 5: Label columns correspondingly

COLUMN 1 = tract_id
COLUMN 2 = year
COLUMN 3 = primary_type
COLUMN 4 = count
COLUMN 5 = one_race
COLUMN 6 = black_or_african_american
COLUMN 7 = american_indian_and_alaska_native
COLUMN 8 = cherokee_tribal_grouping
COLUMN 9 = chippewa_tribal_grouping
COLUMN 10 = navajo_tribal_grouping
COLUMN 11 = sioux_tribal_grouping
COLUMN 12 = asian
COLUMN 13 = indian
COLUMN 14 = chinese
COLUMN 15 = filipino
COLUMN 16 = japanese
COLUMN 17 = korean
COLUMN 18 = vietnamese
COLUMN 19 = other_asian
COLUMN 20 = native_hawaiian_and_other_pacific_islander
COLUMN 21 = chamorro
COLUMN 22 = native_hawaiian
COLUMN 23 = samoan
COLUMN 24 = other_native_hawaiian_and_other_pacific_islander
COLUMN 25 = race_alone_or_in_combination_with_one_or_more_other_races_total_population
COLUMN 26 = hispanic_or_latino_total_population
COLUMN 27 = hispanic_or_latino_total_population_of_any_race
COLUMN 28 = mexican COLUMN 29 = puerto_rican
COLUMN 30 = cuban
COLUMN 31 = other_hispanic_or_latino
COLUMN 32 = area_name



STEP 6:

In the "ROW" section put the "primary_type" here
STEP 7: Under the "values" put all the ethnicities column 6 - 31



RESULT:

primary_type	m of mexican	Sum of native hawaiian	Sum of native hawaiian and other pacific islander	Sum of navajo_tribal	Sum of puerto rican	Sum of samoan	Sum
ASSAULT	585136	2337	21406	784	189303	507	1
BATTERY	587336	2352	21406	784	189462	522	1
BURGLARY	578996	2315	21305	784	188048	507	1
CRIMINAL DAMAGE	585771	2352	21406	784	189346	522	1
CRIMINAL TRESPASS	553740	2156	20149	783	174648	414	1
DECEPTIVE PRACTICE	584226	2352	21406	784	189780	522	1
MOTOR VEHICLE THEFT	585392	2342	21356	784	188615	522	1
NARCOTICS	471499	1747	16825	733	158422	398	1
OTHER OFFENSE	583914	2352	21393	784	188518	522	1
ROBBERY	572061	2214	20590	782	181991	490	1
SEX OFFENSE	411067	1573	15270	358	130906	429	1
THEFT	587336	2352	21406	784	189972	522	1
WEAPONS VIOLATION	555710	1929	17945	765	163831	465	1
ARSON	232345	733	7770	642	71898	197	1
OFFENSE INVOLVING CHILDREN	477165	1950	14917	679	145627	479	1
PUBLIC PEACE VIOLATION	250304	1192	12005	411	97770	355	
CRIMINAL SEXUAL ASSAULT	421890	1686	16122	542	132242	399	1
HOMICIDE	247689	677	6405	417	64710	126	1
LIQUOR LAW VIOLATION	114331	459	3499	0	37874	188	
STALKING	192825	640	5910	236	56365	89	
INTIMIDATION	144546	415	5092	261	37465	121	1
INTERFERENCE WITH PUBLIC OFFICER	151797	630	4390	305	56783	154	1
OBSCENITY	39982	141	2545	191	11293	0	
KIDNAPPING	81526	245	2541	59	23396	46	
CONCEALED CARRY LICENSE VIOLATION	49060	234	718	159	18618	78	
CRIM SEXUAL ASSAULT	589	147	2003	191	6898	26	
GAMBLING	3948	52	494	0	2148	9	
HUMAN TRAFFICKING	1758	0	411	0	1819	0	
NON-CRIMINAL	522	0	63	0	118	0	
OTHER NARCOTIC VIOLATION	7139	20	525	24	2322	11	
PROSTITUTION	20637	164	239	0	7903	34	
PUBLIC INDECENCY	1320	35	105	0	1130	1	
RITUALISM	0	0	0	0	0	0	
Total	**9681557**	**37793**	**347617**	**13810**	**3099221**	**8655**	**37**

References

1. URL of Data Source: https://data.census.gov/
2. URL of Data Source: https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-Present/ijzp-q8t2/about_data
3. URL of Data Source: https://data.lacity.org/Public-Safety/Crime-Data-from-2020-to-Present/2nrs-mtv8/about_data
4. URL of Data Source: https://www.esri.com/arcgis-blog/products/product/data-management/new-spatial-aggregation-tutorial-for-gis-tools-for-hadoop/?rmedium=redirect&rsource=blogs.esri.com/esri/arcgis/2015/03/25/new-spatial-aggregation-tutorial-for-gis-tools-for-hadoop
5. GitHub: https://github.com/xuewentang/5200group2