REEU 2021 – Homework 2

Justin A. Gould
gould29@purdue.edu

June 21, 2021

Instructions
All files and data you need to complete this assignment are:

- `./tl_2015_26_prisecroads`: A folder containing a shapefile (and supporting files) of Michigan's limited access highways.
- `./tl_2016_26_cousub`: A folder containing a shapefile (and supporting files) of Michigan's counties.
- `./ch11`: A folder containing the data used in weeks 5-6 lectures, from chapter 11 of our textbook. *If you followed along our lecture notebook, loading the data into your Postgres database was already done.*

Problem 1

Split a POLYGON

Please create a geometry collection consisting of 2 halves of a right-hand-winding polygon, based on the following:

Input: Below are the shapes you will use in your SQL query. Split the `POLYGON` with the `LINESTRING`

- `POLYGON`:

```
ST_Buffer(
    ST_ForceRHR(
        ST_Boundary(
            ST_GeomFromText('POLYGON ((50 50, 50 150, 150 150, 150 50, 50 50))')
            )
            ),
            20,'side=right')
```

- `LINESTRING`: `ST_MakeLine(ST_MakePoint(10, 10),ST_MakePoint(190, 190))`

Desired Outputs:

1. Correct SQL query text shown.
2. Showing the `.head()` of your `GeoDataFrame`. Columns to include:
   (a) Geometry
   (b) Geometry type
3. Showing the `.plot()` of your `GeoDataFrame`.

Problem 2

Dividing San Francisco into Hexagons

Using the `ch11.stclines_streets` dataset from our lecture, split the road network of the San Francisco into hexagons of $500 ft^2$. Your geometry's CRS should be WGS 84.

Input: The `ch11.stclines_streets` dataset.

Desired Outputs:

1. Correct SQL query text shown.
2. Showing the `.head()` of your `GeoDataFrame`. Columns to include:
   (a) Geometry
   (b) The row number along the grid, i
   (c) The column number along the grid, j
   (d) The number of streets contained within each hexagon
   (e) Order the `GeoDataFrame` by the number of streets within each hexagon descending
3. Showing the `.plot()` of your `GeoDataFrame`.

Problem 3

Create a LINESTRING from a Series of POINTs
Using the `ch11.aussie_track_points` dataset from our lecture, return a `LINESTRING` containing all chronologically-ordered points for each race (`track_fid`). Please transform your coordinates into WGS 84 / Pseudo-Mercator.

Input: The `ch11.aussie_track_points` dataset.

Desired Outputs:

1. Correct SQL query text shown.
2. Showing the `.head()` of your `GeoDataFrame`. Columns to include:
   (a) Race ID (`track_fid`)
   (b) Geometry
   (c) The race's start time
   (d) The race's end time
   (e) The length of each race in meters
   (f) Order the `GeoDataFrame` by the `track_fid` ascending
3. Showing the `.plot()` of your `GeoDataFrame`.

Problem 4

Create a MULTILINESTRING of all Races from Problem 3
Using the result of problem 3, please combine every race's `LINESTRING` into a single `MULTILINESTRING`.

Input: The `GeoDataFrame` from problem 3.

Desired Outputs:

1. Correct SQL query text shown.
2. Showing the `.head()` of your `GeoDataFrame`. Columns to include:
   (a) Geometry
3. Showing the `.plot()` of your `GeoDataFrame`.

Problem 5

Determining which County in Michigan Has the Highest Mileage of Limited-Access Highways

Michigan–particularly the Detroit area–is known for having many highways. Please determine if this is, indeed, true. Given the shapefiles for both the counties (`POLYGON`) and limited-access highways (`LINESTRING`) in the state of Michigan, please determine the total mileage of limited-access highways belonging to each county in Michigan.

Input: The shapefiles contained within the `tl_2015_26_prisecroads` and `tl_2016_26_cousub` folders provided in this homework directory.

Desired Outputs: You will be required to provide a `GeoDataFrame` and a `pandas.DataFrame`. Please see the below requirements:

1. Correct SQL query text to generate the `GeoDataFrame` shown.
2. Showing the `.head()` of your `GeoDataFrame`. Columns to include:
   (a) County name
   (b) Highway name
   (c) Highway ID (`LINEARID`)
   (d) Geometry of the portion of a given highway within a county's `POLYGON`
   (e) Length of the geometry of the portion of a given highway within a county's `POLYGON`, in miles
3. Showing the `.plot()` of your `GeoDataFrame`.
4. Showing a `pandas.DataFrame` with the following columns:
   (a) County name
   (b) Sum of the length of highway portions within the county's `POLYGON`, in miles. **Convert the GeoDataFrame into a Pandas DataFrame to perform aggregation.**
   (c) Order the `pandas.DataFrame` by the summed highway length descending

TIP: The SQL query to generate the `GeoDataFrame` may take a couple minutes to run. To test your logic, I suggest running on a subset of counties (via SQL's `LIMIT` function, before applying to all data once your query behaves as desired.)

HINT: In order to return lengths in miles, you will need to change both the county and highway CRS. This can be done by changing to a CRS providing lengths and distances in either meters or feet, and calculating a simple conversion.