Just a single required problem this week, though I'm including a possible extra credit question as well if you are on the hunt for some last extra homework points. Everything is dealing with PostGIS, so make sure you have a database up and running with that extension installed. Most of the data this week is provided in the form of text files that are *tab* delimited, which is the default for reading in text files. I have also included a `.sql` file to help you generate the necessary tables. The extra credit problem will require importing in a shapefile, so you'll need to handle that as we discussed in class should you want to complete that problem.

As always, ensure that you include both the code from your queries as well as your final answers to the question in your submission. Follow the link here to accept the assignment and get access to the repository:

Assignment link: https://classroom.github.com/a/h0zMH8bB

1. Suppose you managed to get on a non-stop flight directly from Portland, OR to Paris, France, and you are curious as to what cities you will be able to see from the air over the duration of your flight. The distance you can see from an airplane naturally varies with altitude and seeing conditions, but let's take, as a reasonable estimate, a distance of 150 kilometers. The below parts will provide you with some checkpoints and specific questions to answer.

   (a) (5 points) Make sure you have the `cities1000` data read into the `cities` table. You are going to need the location of these cities in a GIS format, so add another column named `geog_loc` which will store a `geography` point object in SRID 4326. Go ahead and populate this column using the latitude and longitude data already in the table, and then add a GIST index to it for faster lookups later.

   (b) (5 points) The flight path itself can be computed as a line between Portland and Paris, being sure that you cast it to a geography type, so that you get the great circle path (that a plane would actually fly around the curved Earth), instead of a straight line path. To get the visible cities, you can use the `ST_DWithin` function, which will work with a line and a point just as well as with two points. Select from the `cities` table only those cities that are within 150 kilometers of the flight path. As a sanity check, you should get 6128. You don't need to include this output with your submission, just the SQL you used.

   (c) (3 points) Building on the query you created above, what are the 5 most populous cities that you will see over the duration of your trip?

   (d) (5 points) Create a breakdown of how many cities you see from each country over the duration of your flight. Your table should include full country names as well as city counts, and should be ordered in descending order by count. Export it as `cities_per_country.csv` and upload it to GitHub.

2. (6 points (bonus)) Included in the repository is a shapefile entitled `us_counties_2010`. Importing this shapefile into your database will get you a table of all the county polygons in the US. Use this data in conjunction with the `cities` table from Problem 1 to return a table of all the counties in Oregon along with their population density (in units of people per square kilometer). Order your table in descending order by population density. Export it as `or_county_densities.csv` and upload it to GitHub.