This week's assignment is largely many smaller questions all centered around one dataset, though there is a more theory based initial question. All questions have templates provided for them within the repository. **I'm going to ask that you only use techniques that we have talked about in class on this week's homework. We will discuss `GROUP BY` later which makes some of this easier, but working with more limited options is good practice when learning those specific skills.**

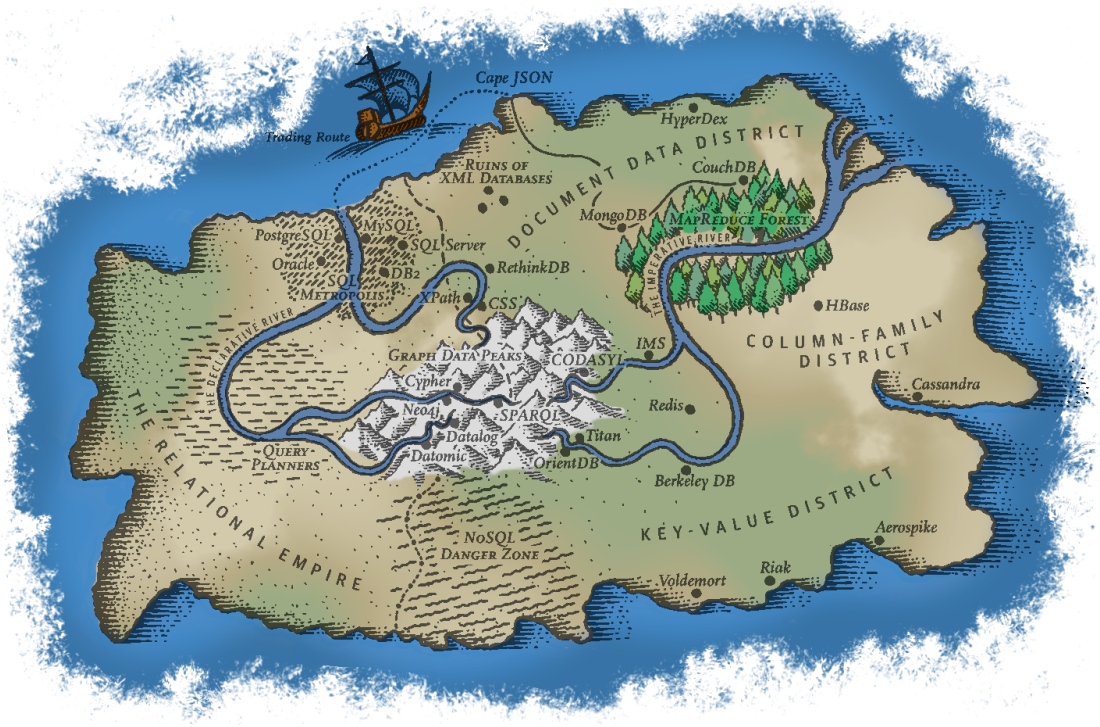
In order to accept the assignment and get access to the repository, you should follow the link here:

Assignment link: https://classroom.github.com/a/DBSX0sDC

The data CSV for problem 2 and 3 this week is too large to store in a GitHub repository, so I've stored it elsewhere, and I am linking you to it below. It is a large CSV (about 250 MB) so make sure you have a decent internet connection when you go to download it. I am also linking you to the official data dictionary for this table.

Data link: here
Data Dictionary: here

1. There were numerous database systems depicted as cities on the map shown in class (and also shown below). Choose *two* non-SQL based "cities" on that map, and do a bit of research on them. Write up 2-3 sentences for each explaining what makes that system unique or special and why it is located where it is on the overall map.



2. The data for this problem is based on all the yellow taxi rides that took place within New York City over the month of January 2022. Once you've downloaded the data, you'll need to begin by creating a new table named `taxi_rides` and populating it with the necessary columns and corresponding data types. The provided (and official!) data dictionary should help here with getting column data types correct, though you will probably still want to look at the CSV itself. In particular, note that the columns in the CSV are not entirely in the same order as provided in the data dictionary. Additionally, the first column in the CSV is not mentioned in the data dictionary, and is just a unique integer assigned to each ride to be able to uniquely identify them.

   Importing in the CSV data may take a little time, as it is a *large* CSV file. Once everything is imported, you can proceed to answering the following questions.

   (a) How many records of taxi rides are included in the data set? How many records are included which both started and ended outside the supposed January 2022 time span? (Some of these are pretty wild if you look at their actual times!)

   (b) How many *total passengers* rode in taxis that traveled over toll roads?

   (c) What is the most common number of passengers on a ride, and how many of these types of rides occurred within the dataset?

   (d) What percentage of the total trips had a disputed charge? (*Consult the data guide for how to identify a disputed charge!*)

   (e) For those riders that pay with a credit card, what is the average credit tip that is left for the driver?

   (f) What was the median amount charged (total) *per passenger* across all rides?

   (g) What is the most common pickup location? Dropoff location? What about the most common route (pair of pickup + dropoff locations)? *Hint: You can get the mode of a pair of values together if you include them both in a set of parenthesis in the ordering statement, e.g.* `(ORDER BY (col1, col2))`. Want to know what these correspond to? There is a separate officially provided location lookup table here!

   (h) Here is a fun question! How many taxi rides seemingly took their passengers back in time? What was the average duration that they traveled back in time?

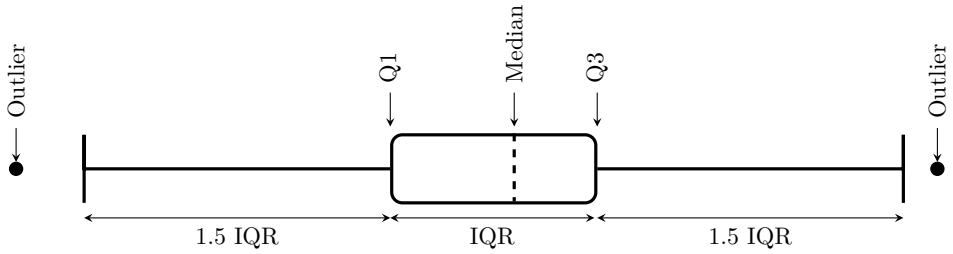
3. This question is still using the taxi rides data set, but is a bit more involved, so I wanted to give it more room for an explanation. Suppose you wanted information on the statistical speeds of taxi drivers across the city. You have access to both distances and times, so calculating a speed should be straightforward. However, there are a few things to consider:

   - As seen above, not all the time durations seem like real physical trips. So you should only consider rides that were at least 30 seconds long.
   - Also, some trip distances are somehow reported as 0. Those should be ignored.
   - Finally, by default, when you subtract two timestamps you will get an interval. Unfortunately, the way that Postgres stores intervals is not conducive to the sorts of arithmetic you need to do here, as you need to be able to divide by a time in a known unit. So to get an interval in hours that you can actually use to calculate a speed, you can do the following:

     ```
     (EXTRACT(EPOCH FROM (dropoff_time - pickup_time))/3600)
     ```

     where `dropoff_time` and `pickup_time` are whatever you named those columns in your table. We'll talk more about how these functions involving times and intervals work later in the semester, but for now you can use the above to get the trip duration in hours, which will allow you to compute a speed in miles per hour.

   Your primary objective here is to identify all the trips with speeds that meet the above criteria but which are *outliers* of the main speed distribution. Here we are going to define outliers as points that are below the lower whisker or above the upper whisker on a classic boxplot, for which I am including a diagram below. In this case, the whiskers are located 1.5 IQR below and above the 1st and 3rd quartiles, respectively.



   You need to identify all the valid rides that fall into these ranges (and are thus an outlier), and write them out to a CSV file entitled `bad_taxi_mphs.csv`. The file should be ordered by increasing speed and should include just the taxi ride ID and the speed of that ride. Include a header at the top and make sure to upload your CSV back to GitHub (it shouldn't be that large).

   Do *not* feel like you need to do this all in a single query! Break it up as you need! And you are free to do simple calculations like arithmetic either with your own calculator or using Postgres as a calculator.