My work so far has been in a few areas.

The first thing I wanted to do is to try to make sense of the data. I have spent a fair amount of time doing this.

One of the first things that I wanted to do was load in all of the data. There are 13 files to load. I used Queue.js to do this. I found that one file in particular gives problems on load, proximity.csv. I am not sure what the issue is. I have spent considerable time trying to analyze it. The file is over 2 million rows, too big to load into Excel.

I loaded the file into TextWrangler and tried to Zap Gremlins. This left the md5 signature unchanged. I then loaded the file into Google Refine, and simply exported back out as a CSV. This resulted in a changed md5 signature. The file still gave me problems.

I then split the file into three files. Two 1M row files and the remainder, which was much smaller. The smaller sized file loaded fine, neither of the 1M row files loaded properly.

Another file being used WLAN2.csv is over 1M rows with no trouble. I continue to try to troubleshoot this issue, as we wish to use the proximity data. However in the meantime, I have simply commented out the loading of Proximity.csv until I can figure out what is going on with it.

**File Loader**

The goal of my File Loader is to load all files, in parallel, using Queue.js. The routine will then perform any transformations on the data. For example many of the files have dates in them, and many of the dates use entirely different formats. So we will convert all the dates to actual JavaScript dates in this routine. Another thing that may be useful is certain field name transformations, for example changing “techno / lounge / electronic” to just “techno”.

A goal of File Loader is to give the user visual feedback. I am currently doing this using an event handler that is attached to the XHR object that gets passed into Queue.js

My File Loader code is here:

<https://github.com/andrew-reece/cs171-final-project/tree/master/loadData>

**Visual File Loader**

An improvement to the File Loader is to make the loading of files and associated progress “visual”. My thought is to use a circle which when completed shows 100% progress. Each file will show an even percentage of the progress. The progress will be updated when the .get() callback fires after the XHR object is loaded.

I do not have this working yet, but I have made good progress in understanding what it is I need to do.

In creating the Visual File Loader, I also started a new Branch, so I spent some time becoming acquainted with branching and working in git with a group.

My Visual File Loader is here:

<https://github.com/andrew-reece/cs171-final-project/tree/visualLoader/loadData>

**Animating a Time Series**

“Social Evolution” data is organized mostly in Time Series. The data is individual observations and so there can be multiple challenges with visualizing it. One thought is to make a series of aggregates, and display them over time. So this thought is that the individual may not be of interest, perhaps the aggregate is. Of course there are hybrid visualizations that will show both individuals and aggregates and let you focus on what you want.

In order to experiment with visualizing data changing over time, I used the MusicGenreImmersion data. This data is a good example of data that may not be so useful as individual observations, but may be useful as an aggregate. I load the data, and materialize the aggregate. I then have outlined a process like so:

1. Take the extent of the date range
2. Create a linear scale with a domain of 0,50 and a range of the date range
3. Create a slider with values 0,50 and attach an event handler to it which passes the value of the slider into the linear scale to produce a filterDate
4. On “change” of the slider call updateVis() which will filter the data by filterDate and transition the graph to reflect the filtered changes

This is a general framework, which I plan to use for multiple graphs. So a single “clock” will “tock” and as it does it will change the visualization(s) . The user activating a slider could manually induce this change or it could be automatic by a clicking a “Play” button which will animate the visualization.

I have not yet implemented a working version of the Play button, but I am very close. I have the concept sketched and I see no challenges.

1. Attach an event handler to the Play button
2. On “click” the event handler will call a play() function
3. play() will iterate 0,50 and inside the loop it will update filterDate and call updateVis

I will likely swap my minimal HTML slider for a more modern d3 slider. Another idea is to not just use a slider to control the filterDate, which essentially sets the maximum data, but actually use a slider that allows you to control the entire extent, adjusting a lower bound and upper bound for date. Right now these are just ideas, and are things that could put into our visualization if there is enough time.

My Animating a Time Series code can be found here:

<https://github.com/andrew-reece/cs171-final-project/tree/master/musicGenreImmersion>

Each of these things are ideas which we did not do in class.

In summary my time has been spent:

1. Figuring out what data may be interesting to look at
2. Loading and transforming data (data cleansing), which I still have challenges with
3. Figuring out how to visualize the loading of