**CHALLENGES**

* We originally wanted to include hurricane data as an additional disaster, but we were not able to find the data in a usable format, so we had to eliminate hurricane as an option.
* General Observations: The historical data for the various sources didn’t always cover the same timeframe. We decided to just pull it all in. But as a next step we need queries to limit comparison or we should limit the timeframe that us consistent among all data. Based on keeping all data, then … so to do an in-depth analysis (apple to apple comparison) will require limiting the date range for some of the queries/questions.
* We had to make sure that certain columns in the tables that were created, did not have constraints when importing the data via Python. We had to make sure that the syntax for Date and Time for the tables were imported correctly and that the data for some of the fields/columns were correctly formatted
* FEMA: It was difficult to find the data for FEMA and Zip Code.  Once it was found we also had to find location data, such as County Zip. As it related to the primary key, Zip codes could not be a unique value because a zip code could have multiple cities in a county.  In order to rectify this, we added an addition column to count the rows in the .csv files by using the function = row() to add the count of numbers per row and then set that column as the primary field. Ultimately, we didn’t use the zip code data.
* Earthquake CSV: Many earthquakes that would have affected the mainland United States were centered in bodies of water, wilderness, or in Canada and Mexico.  This made it near impossible to retrieve USA zip codes or state data based on latitude and longitude coordinates.  Instead, the earthquake table was joined to other tables in our database, based on ID numbers and dates.  Earthquakes in or around the mainland USA were identified by determining a latitude and longitude area over the country’s territory, rather than calling APIs to link latitude and longitude to zip code and state via reverse geocoding.
* Tornado CSV: The tornado csv had ‘om’ as the primary key, but we were not able to use this as when we ran the query it displayed all 1s.  We therefore had to create a unique ‘id’ as the primary key.
* Wildfire CSV:  The Wildfire file was over 1.8 million records.  At the beginning we did not realize that the size would have been an issue until we were attempting to push them into GITHUB.  We tried to reduce the file size by deleting some of the columns, but it took a very long time. GITHUB alerted us that we needed the files to be less than 100mb, so we split the files into 100mbs (using split command in bash) and tried to re-upload to GITHUB, but then it told us we needed it to be 50mb.  We eventually had to split the wildfire csv into 94 files and rename each file. For Wildfire csv we kept the empty columns because it was taking too long to remove them.
* SCHEMA: We had SCHEMA challenges when trying to use the ERD tool to get the diagram to show the relationship of all source table feeding the natural\_disaster table. It required creating primary key/foreign key relationship between natural disaster table and other table. Since source tables were populated first, we simply commented out the foreign key constraints. Also, we had to tweak schemas to remove quotes so that table/column names were lowercase. We also had to make a few additional adjustments to columns as we tried loading the data; sometimes removing constraints or changing a field from not null, to nullable.
* Github - Coordinating code check-in of multiple team members sometimes caused data to be out of sync and sometimes code was overwritten with older version.