The SQL\_query.php file runs when the Interactive Map website gets loaded, and connects to the server to query the database and encode the results as a JSON file.

First, you put in the $username, $password, and $database that you are connecting to. You will need to replace the current values that we used for the Georgia Tech server access with the ones you need for your database.

Then, I built the queries for the SQL databases. I’m using 3 queries - one for Potential Inspections, one for Fires, and one for Current inspections. This is so I can control the order in which they get drawn.

The first query is for the Potential Inspections (so they will be drawn underneath of the other two layers on the map).

SELECT `name`, `address`, `in\_fsaf\_yn`, `occup\_type`, `new\_prop\_type`, `liq\_id`, `inspection\_date`, `risk\_rating`, `risk\_category`, `newx`, `newy`, `NPU`, `BATTALION`, `DISTRICT` FROM `Current\_And\_Potential\_Inspections` WHERE `ATLANTA` != 0 AND `in\_fsaf\_yn` = 0

If you are familiar with SQL queries, this will look familiar to you. I am **selecting** all of the columns listed (name through DISTRICT), **from**, the Current\_And\_Potential\_Inspections table in my SQL database, **where** Atlanta doesn’t equal 0 (ie: they **are** inside the city limits), **and** in\_fsaf\_yn = 0 (ie: they are **not** currently being inspected. Thus, potentials).

This is repeated for the other two queries for Current Inspections and Fire Incidents.

\*\*\* If you want to change the query to SELECT some other set of variables, this is what you would change. \*\*\* Think about how this will affect the variables the map (index.html) is expecting to see, though.

Then, we actually query the database with the mysql\_query function, and if it fails, it returns an error.

Then, we build the JSON file that we will be using to draw the points. This is an array, which has certain parameters that are expected (type, features), and those features are themselves an array, of each of the details for each of the points. (feature has type (Feature), properties (name, address…. DISTRICt), and geometry (type = Point, coordinates).

Because the latitude and longitude points came in as strings instead of float (decimal) values, we need to first cast them as floats in each of the loops as we build the JSON.

Finally, after all 3 while loops have created the geojson file using the parameters we queried for, we close the mysql server connection, and return a json\_encoded file called “geojson”).