For the past few weeks, I have decided to play around with an online service that brands itself as “*the Easiest Way to Map and Analyze Your Location Data*“. I am of course talking about CartoDB.

Why CartoDB?

CartoDB is a cloud service that let’s you create stunning maps using their open-source platform without writing one single line of code. The beauty of the service is that it is build upon great open source software and tools such as PostGIS and CartoCSS. The service is widely used in the news industry, with big American newspapers such as the LA Times and The Wall Street Journal but also some international newspapers such as the Frankfurter Rundschau, LeMonde and even The Guardian. It is needless to say that CartoDB is used worldwide and is an important tool to quickly and simply display geographic data.

All of that is great, but what about us cartographers? Should we learn CartoDB and use it to create maps?

I believe that CartoDB is an important mapping tool and should be thought in the UW-Madison curriculum as soon as possible. Here are the reasons as to why I think it should be added to the curriculum:

1. We are already learning about tools, libraries, and services such as Mapbox, D3 and Leaflet. Why not add the CartoDB platform in there?
2. CartoDB is freemium, meaning that it is free until one reaches a certain threshold, and open source!

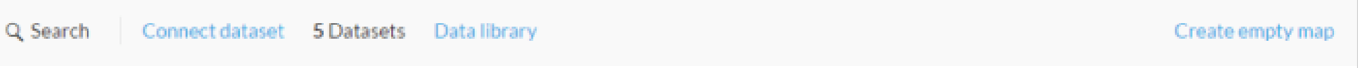
To support what I just wrote, let me show you an example on how to use CartoDB and what I liked about the service.

A map of fires

As an example of what is possible to do with the CartoDB platform, I decided to use a dataset about worldwide active fires that I found on the NASA website. The dataset is updated everyday using data gathered from the MODIS sensor of all the active fires around the world.

I will assume that you have already created an account on CartoDB.com and logged into your account.

Creating a map with a dataset is really easy. Click on “Create Map”. You are now asked to add data.



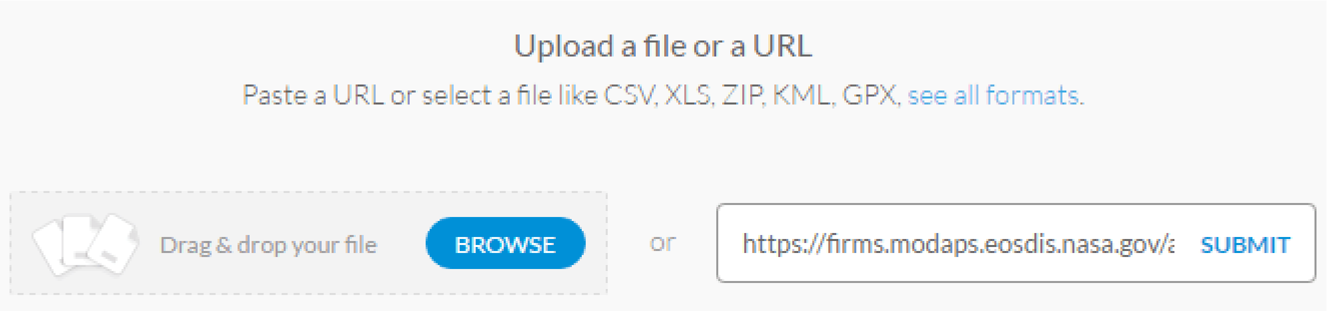
The available data options

You have the possibility to:

1. Connect to a dataset in the form of a CSV, shapefile, and more by uploading a file from your computer or entering a link containing a supported file format.
    1. You have the possibility to connect CartoDB to different cloud storage services such as Google Drive or Dropbox
    2. You can also connect directly to different APIs such as Twitter to access their respective data.
2. Use a dataset that you have uploaded previously

3. Go through their data library that contains data from various sources such as Natural Earth Data, some NASA datasets

In our case, we are going to insert a link. On the NASA website, right click and “Copy link address”. You can select any of the datasets as long as it is a supported format. Paste that link into the CartoDB platform and click “submit”.

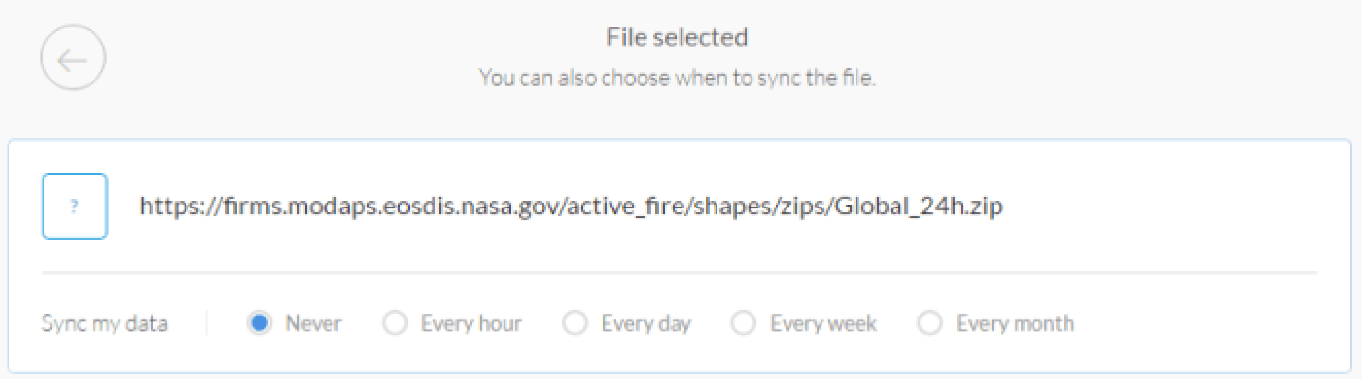


Upload a file or a URL

Paste a URL or select a file like CSV, XLS, ZIP, KML, GPX, see all formats.

https://firms.modaps.eosdis.nasa.gov/a

Since you have a free education account, you are able to ask CartoDB to sync the dataset that you just submitted at regular intervals. Depending on the dataset that you selected, you might want to sync your data every day or every week.



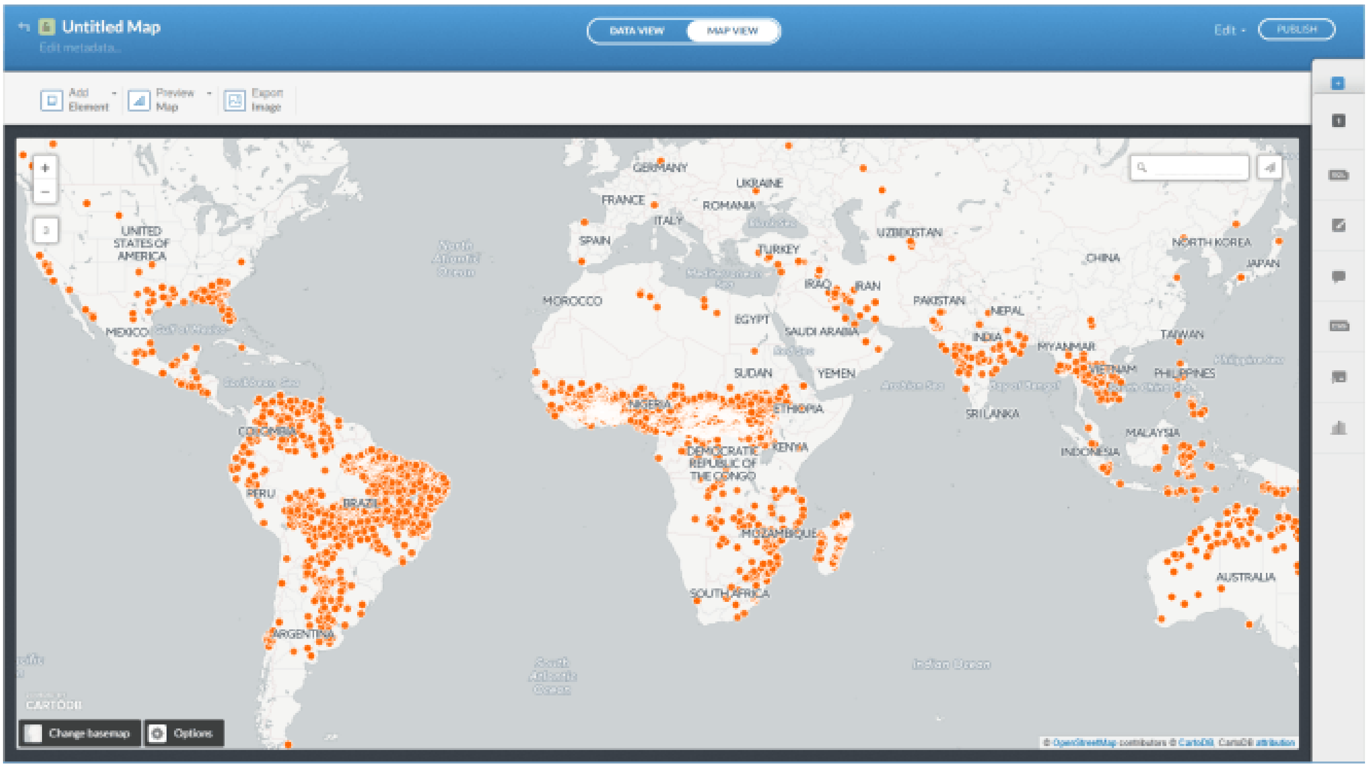
File selected

You can also choose when to sync the file.

https://firms.modaps.eosdis.nasa.gov/active_fire/shapes/zips/Global_24h.zip

Sync my data ● Never ○ Every hour ○ Every day ○ Every week ○ Every month

AWESOME! You have now created your very first CartoDB map. Easy right? You should have a map that looks something similar to the one below.



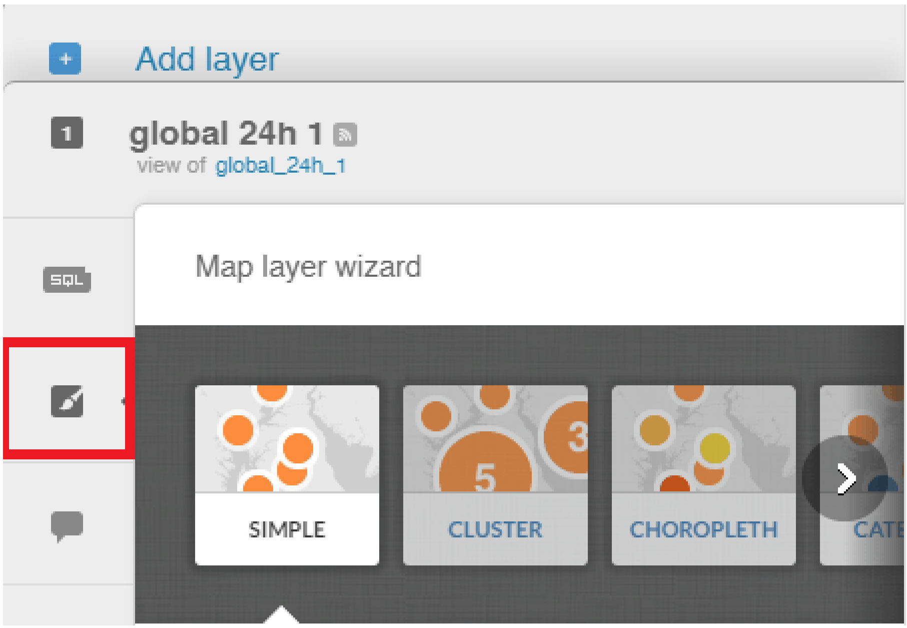
The interface

One of the best reasons to use CartoDB is how user friendly their interface is.

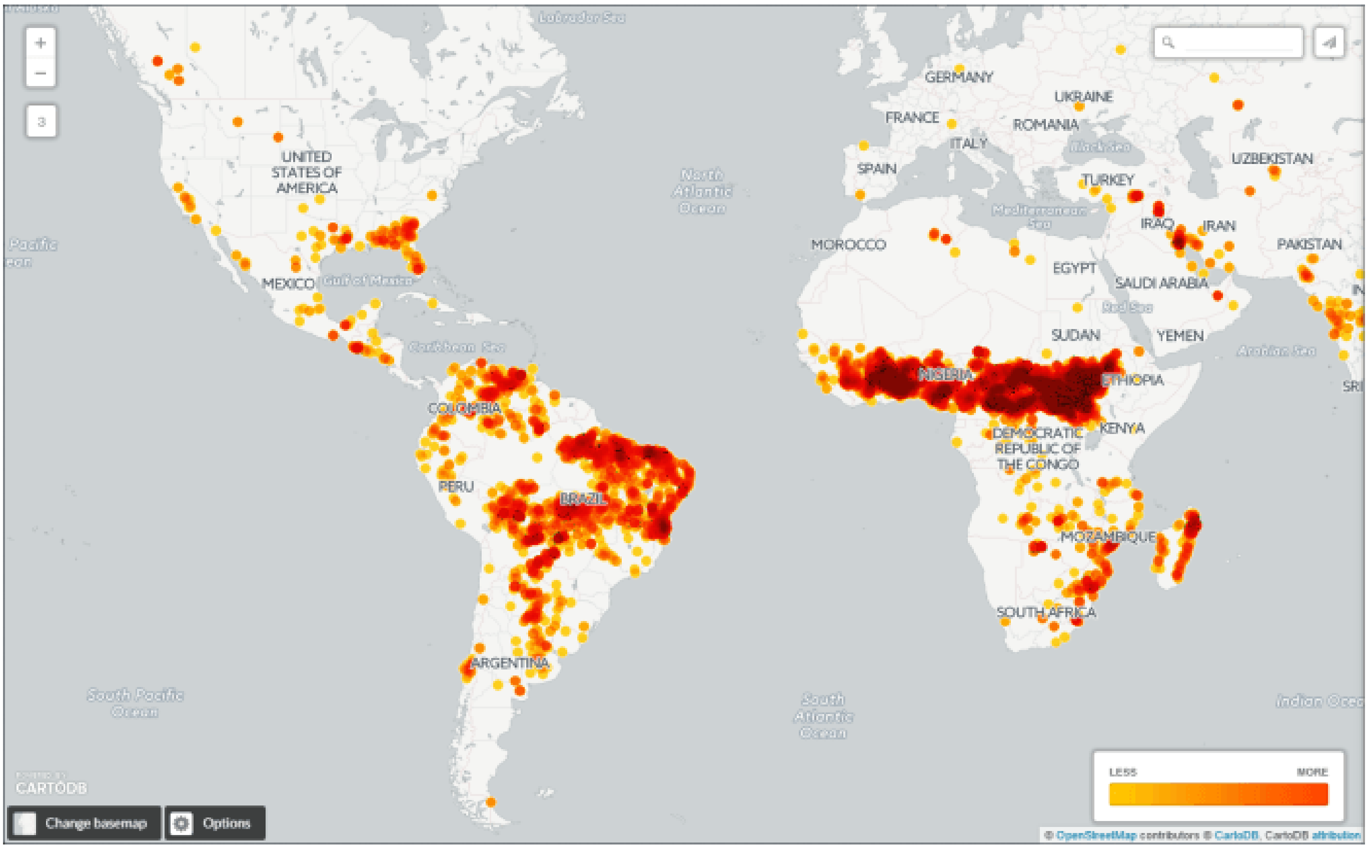
Styling our layer

The data that we is displayed on our map is now a **layer**. Let’s start by changing the way our Active Fire layer is displayed on the map. To do that, you will be using the sidebar that contains all the tools necessary to perform operations on your layer.

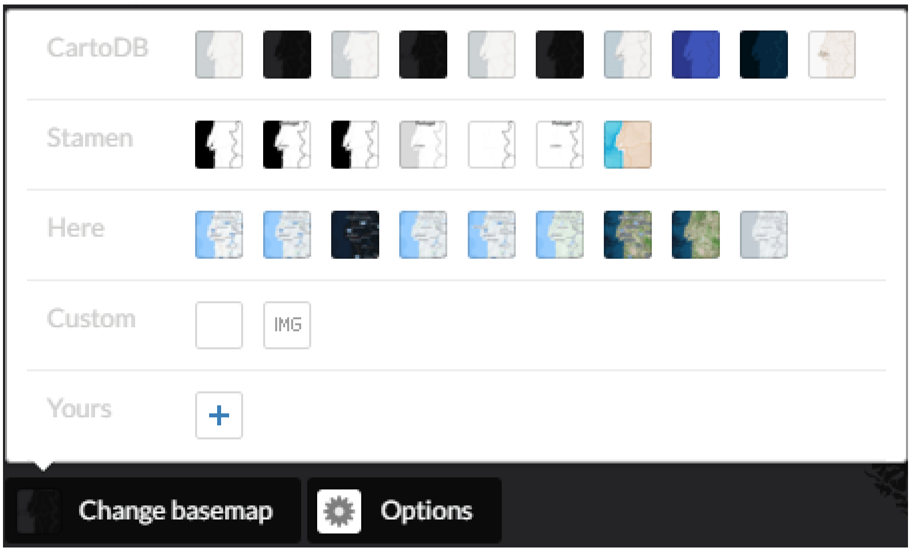
Click on the wizard icon (see image below) to open up the *“Map layer Wizard*“. The Map Layer Wizard page is a GUI (graphical user interface) for CartoCSS.



Now choose how you would prefer to display your data on the map. I decided to go with “Intensity”, which displays clustered points in a darker color than points that are “outliers”. You should get something similar to the map below.



This is great! My eyes are now drawn naturally to the darker cluster of points where more Active Fires have been detected in the last 7 days. However, the basemap could be better if it was darker. To change your basemap, just click on "*Change basemap*" on the bottom left of the map and select an appropriate basemap for your data. I decided to go with the "*Dark matter (labels below)*" basemap as I think it fits quite well to the data that we have.

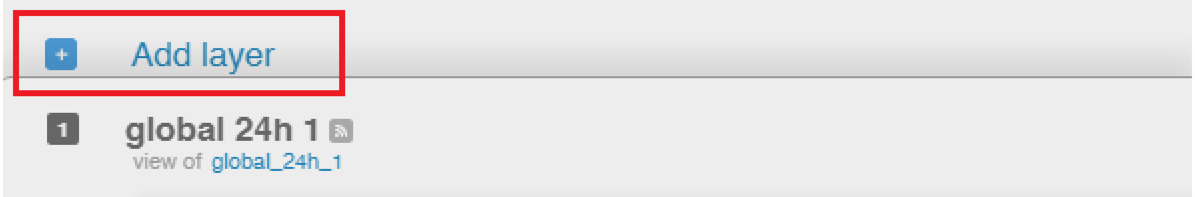


If you decided to, you could publish this map right away but let's work on it a bit more to see what CartoDB is really capable of.

Making a choropleth map

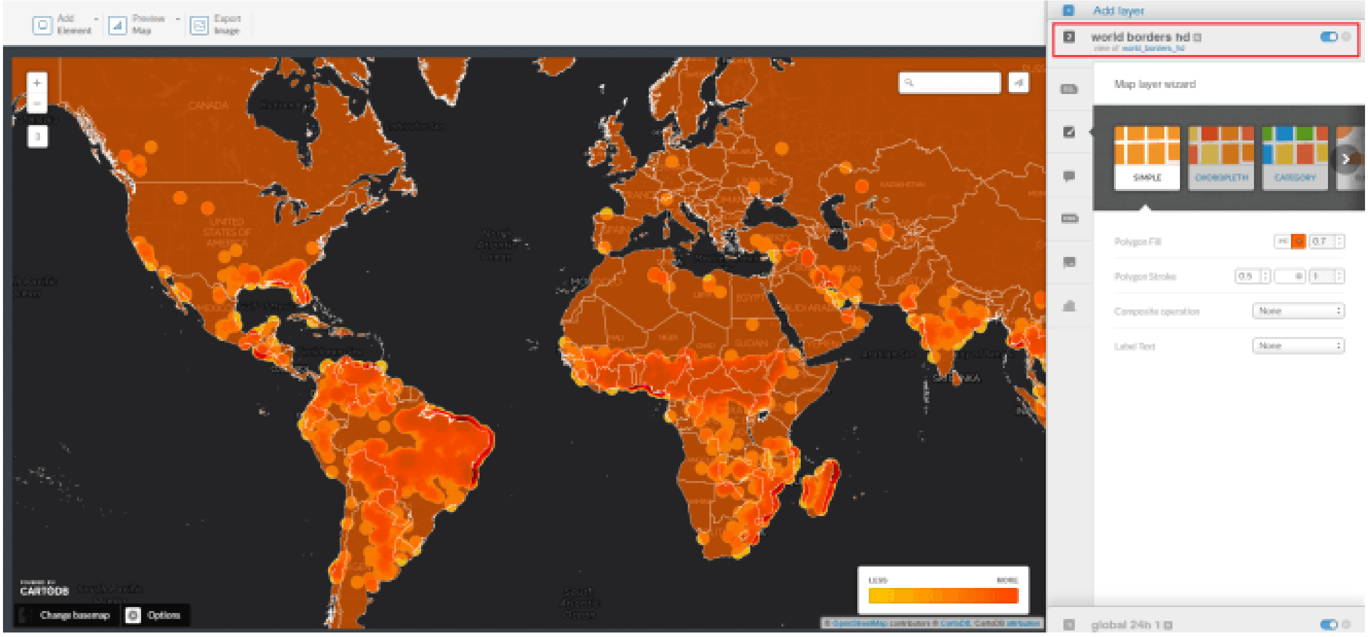
Let's say that you are unhappy about the map and that you would rather have a choropleth map of active fires by country, you do not have to go through any GIS software. Remember that CartoDB is based on PostGIS, a database extension for PostgreSQL that let's you do GIS operations on your data by writing SQL queries.

We first need to add a new layer with all the countries. Using the sidebar, click on the "*Add Layer*".

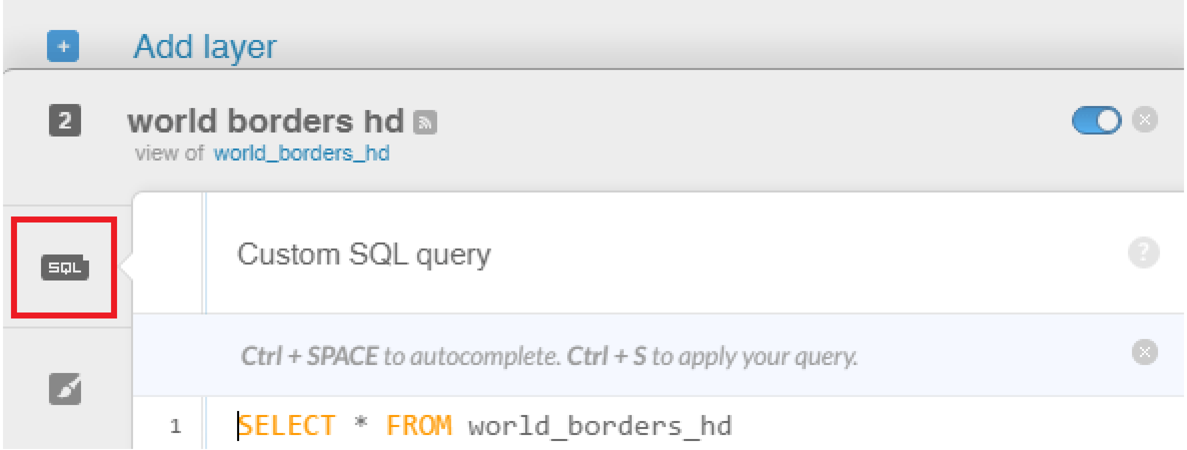


You will now have to select a dataset that has all the countries in the world. Thankfully, CartoDB has the dataset already available to use. Select "*Data library*" and on the left search for "*World Borders (High Definition)*". Once you have selected the layer, click on

“*Add layer*” and wait for about 30 seconds for the layer to be added to your map. Once it is added, you should have a new layer tab called “*world borders hd*” that was created in the sidebar. You can edit anything about that layer by selecting it.



Great! Now that you have added the border layer to your map, let’s use PostGIS to calculate the number of points that are within every country. To do this, go into the SQL tab for the borders layer.



Custom SQL query

Ctrl + SPACE to autocomplete. Ctrl + S to apply your query.

```
SELECT * FROM world_borders_hd
```

To do this, the first step is to add a column to our layer that contains world borders to store the number of points that a country contains.

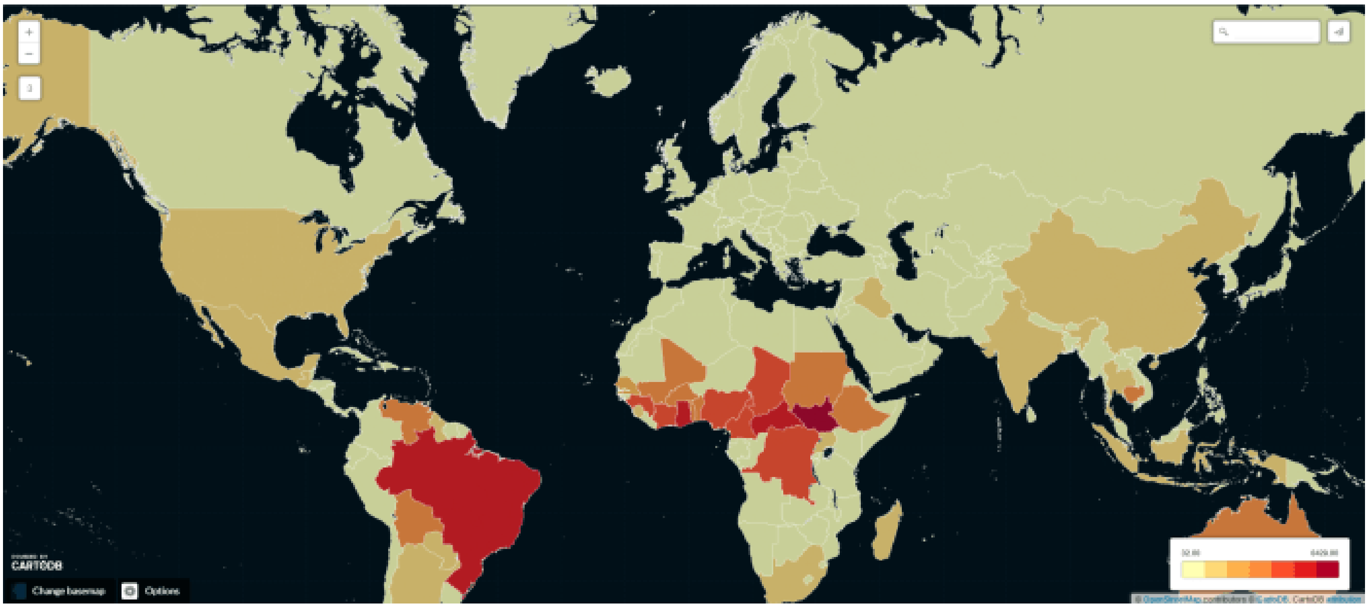
> ALTER TABLE world_borders_hd ADD COLUMN fireCount smallint DEFAULT 0;

Now that we have a new column, let’s use PostGIS to figure out how many points are within each and every country.

```
UPDATE world_borders_hd
SET fireCount = addToColumnss.fireCounts
FROM
(
SELECT world_borders_hd.name, count(fires.the_geom_webmercator) as fireCounts
FROM world_borders_hd LEFT JOIN global_24h_1 as fires ON
st_contains(world_borders_hd.the_geom_webmercator, fires.the_geom_webmercator)
GROUP BY world_borders_hd.name
HAVING count(fires.the_geom_webmercator) > 0
) addToColumnss
WHERE world_borders_hd.name = addToColumnss.name
```

Not so hard is it? All I am doing here is updating the column that we just created with the results that I got from the subquery and matching them with the country name so that they are placed in the right spot within the table.

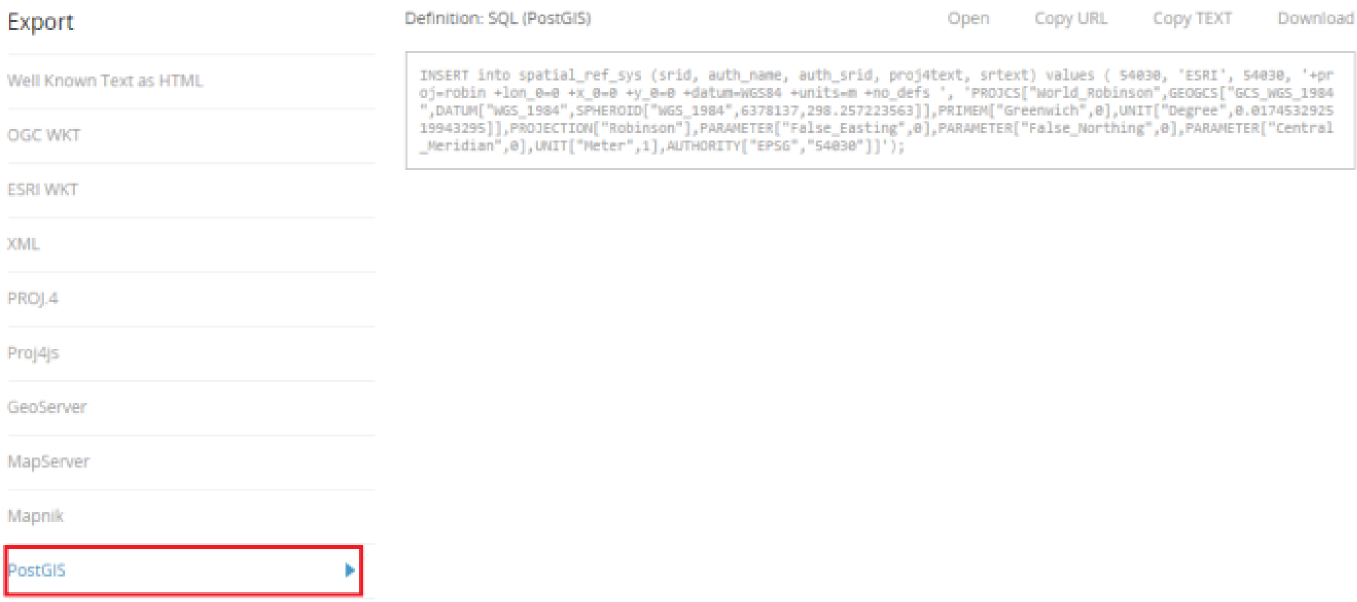
Now all we have to do is go into our Layer Wizard, select “Choropleth” and making sure that the correct column is selected. You might want to deactivate the Active Fire layer. Here is the choropleth map from the data that we just computed:



Projection

Now that we have a choropleth map, let’s project our map so that it uses something else than Web Mercator.

You will first need to go on EPSG.io and get the definition for the projection that you would like to apply to your map. Not all projections are supported (yet?). Copy the PostGIS definition and paste it directly in the SQL tab of any of your layer (don’t forget to “**Apply query**“). This will add the definition for that projection to the list of projections from CartoDB.

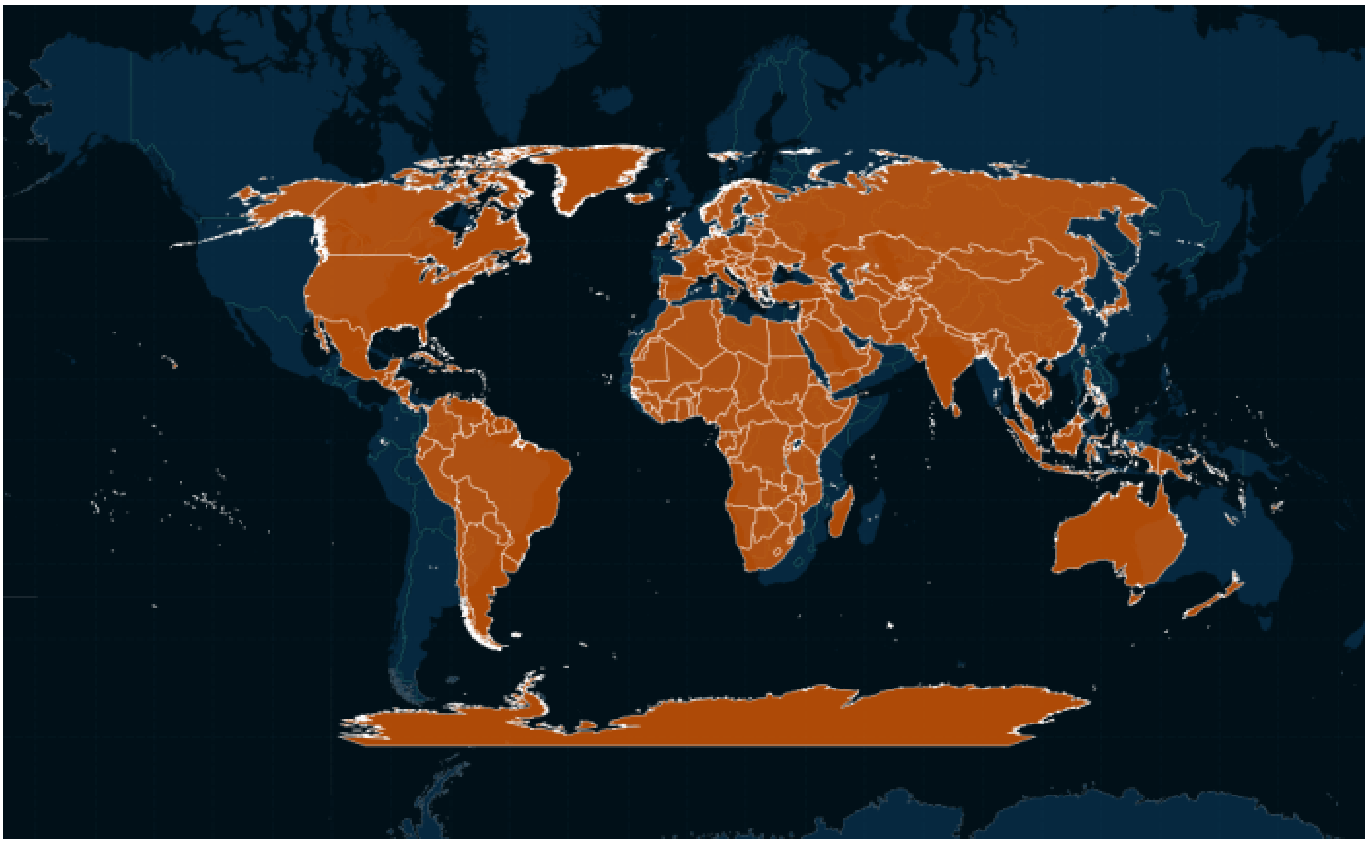


```
INSERT into spatial_ref_sys (srid, auth_name, auth_srid, proj4text, srtext) values ( 54030, 'ESRI', 54030, '+pr
oj=robin +lon_0=0 +x_0=0 +y_0=0 +datum=WGS84 +units=m +no_defs ', 'PROJCS["World_Robinson",GEOGCS["GCS_WGS_1984
",DATUM["WGS_1984",SPHEROID["WGS_1984",6378137,298.257223563]],PRIMEM["Greenwich",0],UNIT["Degree",0.0174532925
19943295]],PROJECTION["Robinson"],PARAMETER["False_Easting",0],PARAMETER["False_Northing",0],PARAMETER["Central
_Meridian",0],UNIT["Meter",1],AUTHORITY["EPSG","54030"]]');
```

We can now use the projection that we just added to the list of projection by applying an SQL query on the layer that needs to be projected.

> SELECT name, firecount, ST_Transform(the_geom, **54030**)
> AS the_geom_webmercator
> FROM world_borders_hd

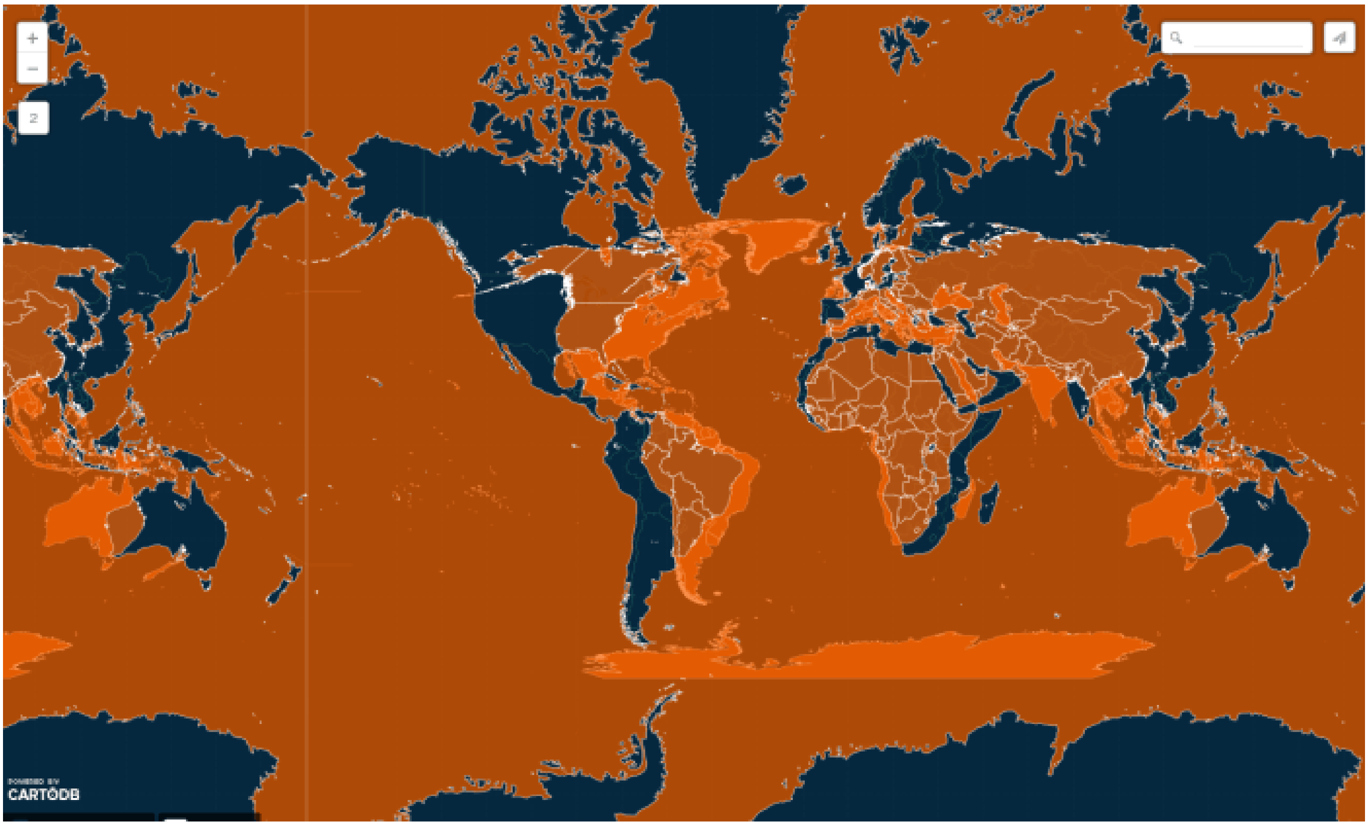
Makes to select the name of the country and the Active Fire count that we calculated earlier. If you do not select those columns, you will lose those and will not be able to create a projected choropleth map. You should also make sure to change 54030 to the srid of your projection!



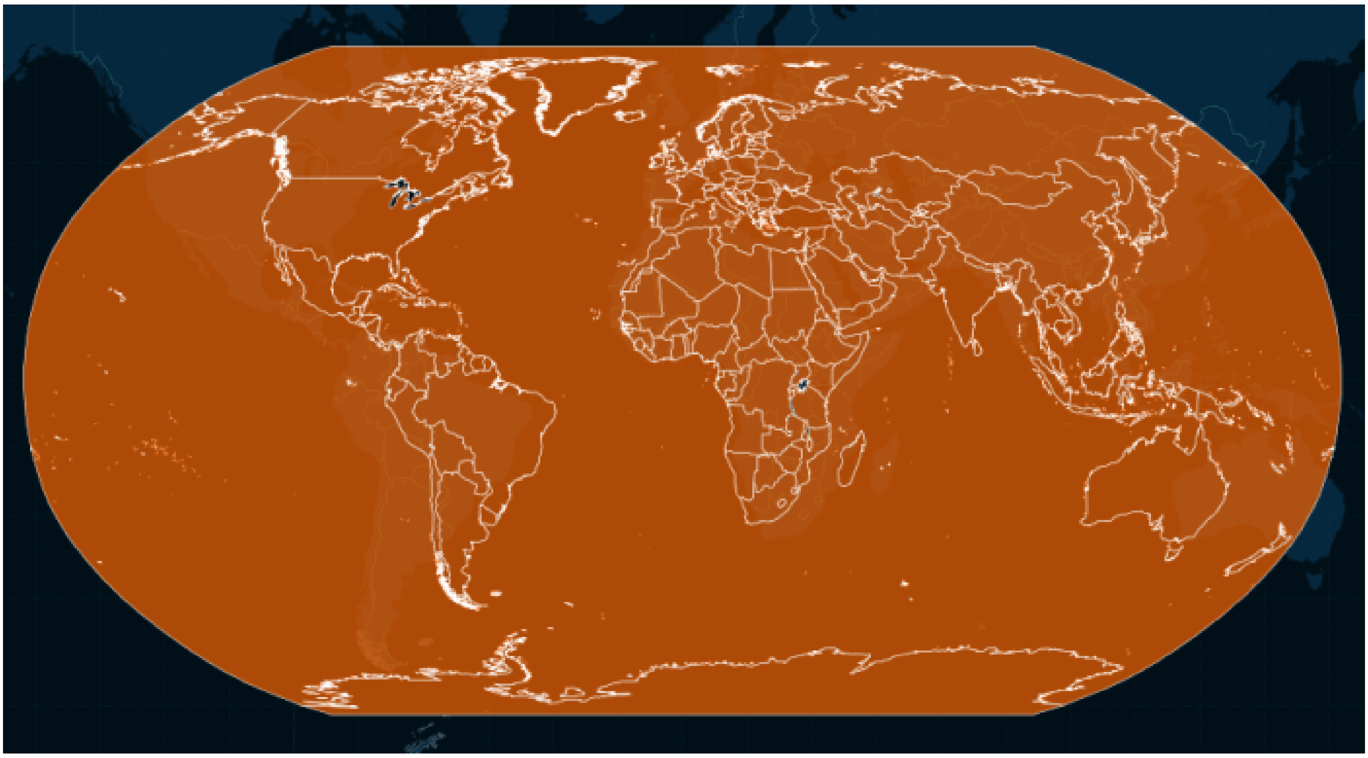
Cool, we now have a projected map. But there are a few problems:

1. It’s not displaying our data anymore
2. Countries are displayed twice

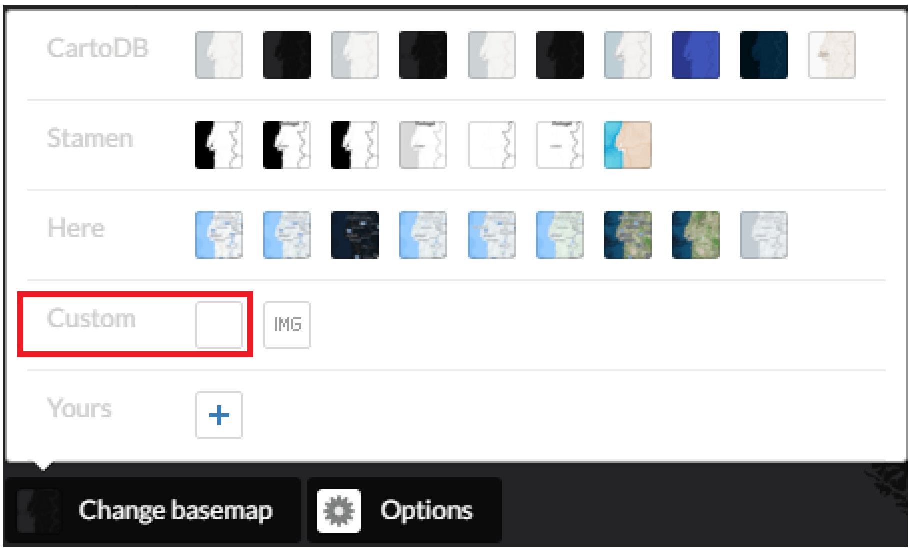
Let’s hide the countries from the original tiles. To do this, we just need to add a ocean layer to our map. You should know how to do this by now, and will not go in details on how to add a layer to your map.



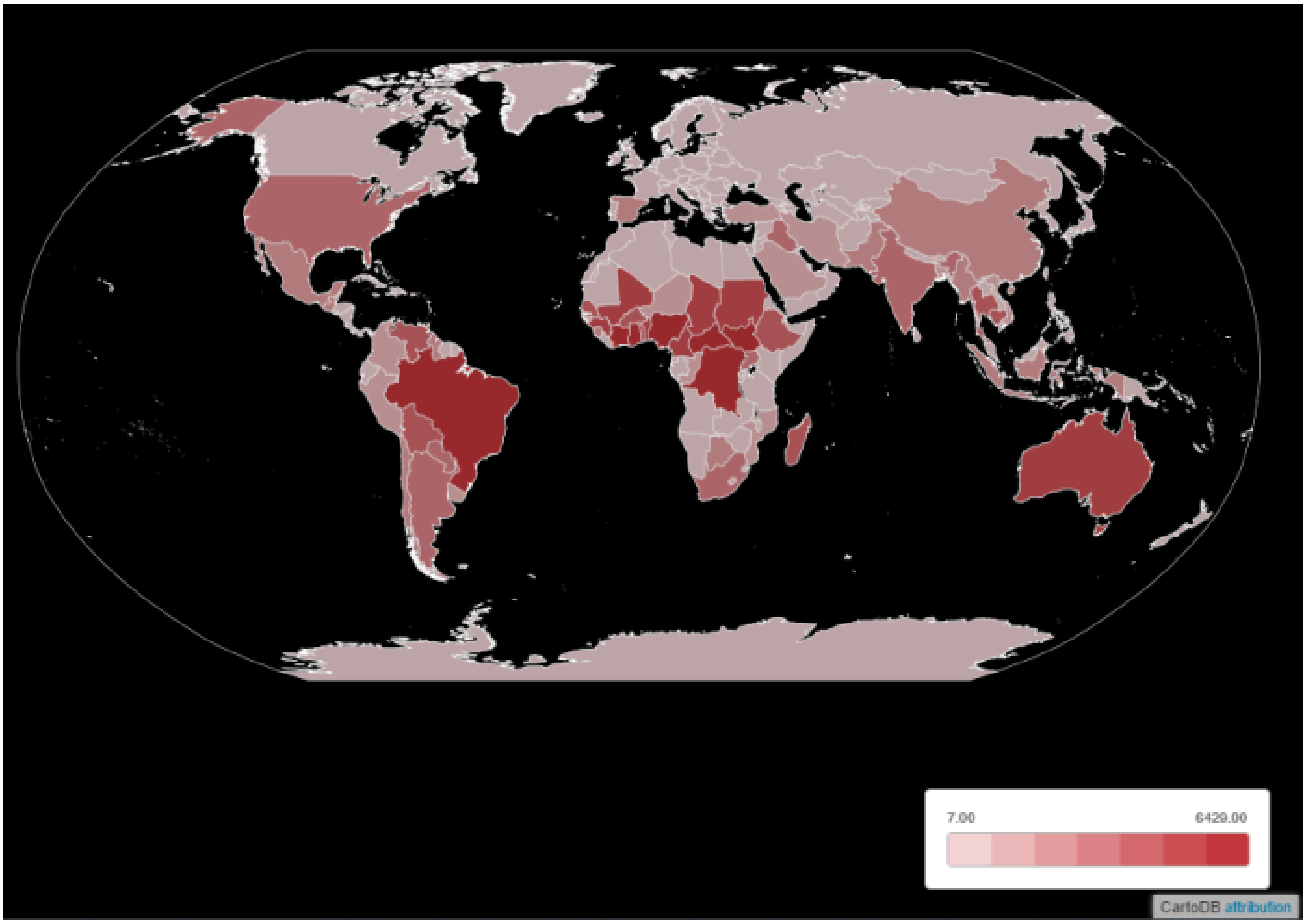
We now have an ocean layer, but it’s not projected! Let’s project it by following the same exact steps from above, but instead of projecting the border layer, we will now project the water layer. You do not need to add the definition back into the set of projection since it is still in there.



Let’s now style this, so that it actually looks like a map. First, let’s hide the tiles in the background by changing the basemap to a color. You would probably want your basemap color to be the same as your ocean layer color.



All you have to do now is to change the style (i.e., color) of your water and make a choropleth map, which we already did earlier in this post.



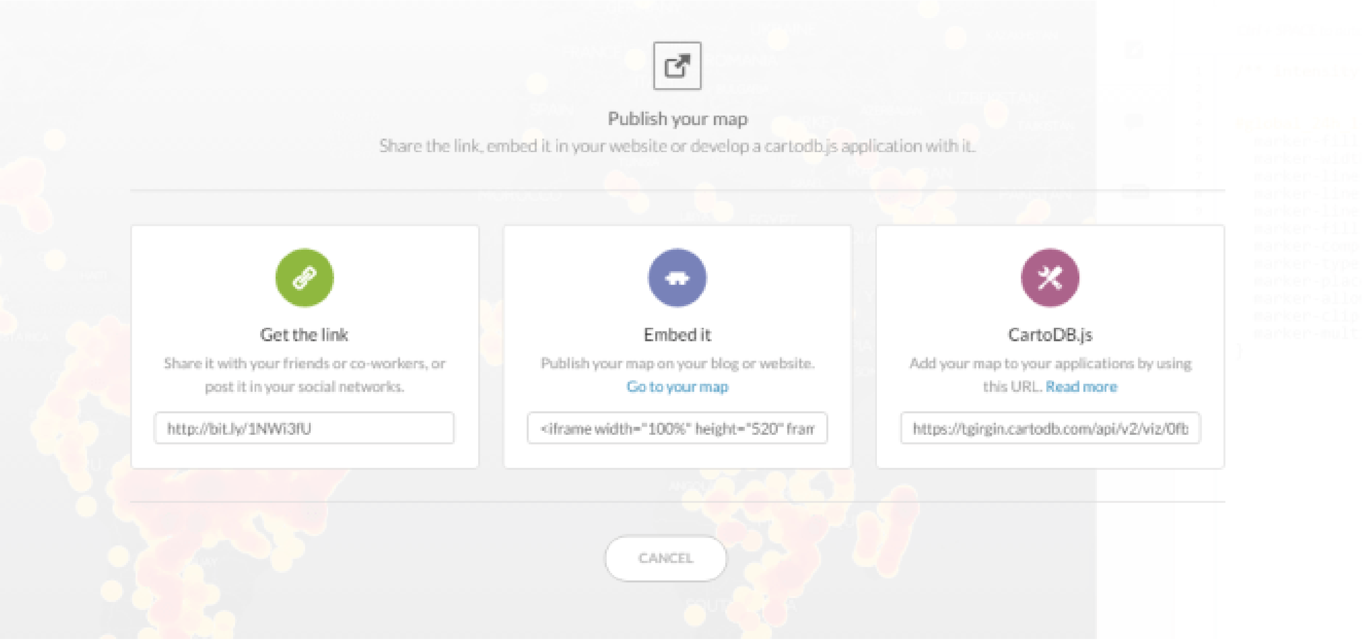
CartoCSS

CartoDB let’s you use the power of CartoCSS to style your map. However, I have found it quite limited compared to how you can use CartoCSS in Mapbox.

For example, I wanted to change the current font used on the map to a local font that I have. However, it seems that CartoDB is only able to render the fonts that they have already installed on their servers. You can read more about it in this great StackOverflow post.

Publish your map

You are now ready to publish your map! Just click on “Publish”, located to the top right of the window.



Publish your map

Share the link, embed it in your website or develop a cartodb.js application with it.

Get the link

Share it with your friends or co-workers, or post it in your social networks.

http://bit.ly/1NWi3fU

Embed it

Publish your map on your blog or website. Go to your map

<iframe width="100%" height="520" fram

CartoDB.js

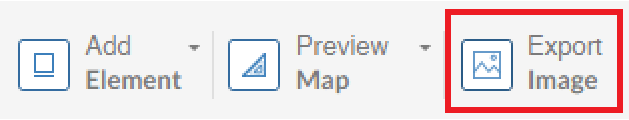
Add your map to your applications by using this URL. Read more

https://tgirgin.cartodb.com/api/v2/viz/0fb

You have three options to publish your "interactive" map:

1. Get the link
    1. This is just a shareable link. You can of course use it in your website, as a direct link to the map.
2. Embed it
    1. If you would rather have the map directly embedded in your website, copy the html code and paste it in the correct element of your website. This will display your website at a fixed size that you can change in the html code.
3. CartoDB.js
    1. CartoDB.js is similar to Leaflet.js. By copying the link given to you and adding it to your JavaScript code, you will be able to interact directly with your map and its data. If you take Geography 575, you will learn more on how to use Leaflet.js which will give you sufficient knowledge to understand how CartoDB.js works.

That's not it though. There is a great option that will let you export your map as an image (static). If you would rather have a static map to have printed or to publish on your website, just select the "Export Image" button.



There you go! You now have a published map 🙂

Where would CartoDB fit in the UW-Madison curriculum?

As you might already now UW-Madison is one of the best schools in the world for Cartography and GIS. The program prepares you very well in becoming not only a cartographer, but also a designer and a programmer. My favorite class was by far Geography 575 (Interactive Cartography & Geovisualization), in which I learned how to use different tools such as Leaflet, D3, jQuery, and JavaScript to create my own interactive map.

I have recently worked with professor Rob Roth to figure out how to implement the CartoDB platform into the current cartography curriculum at UW-Madison.

A note

CartoDB is a great tool that is still in its early stage (this is my opinion of course). There can be a lot added to it to make the user experience better. It would be nice for example to have more tools available directly through the click of a button instead of having to go through SQL.

Imagine being able to project your map into different projection in just a few clicks. At the moment, the only way to project your map in CartoDB is to copy the projection definition (that you need to look for online), and then write some SQL to apply the projection to your map. For someone with some SQL knowledge, that should not be too hard to do. However, most people do not understand what SQL. They should not even know it exist. The way I am thinking about it is that CartoDB should include a PostGIS tab in every layer that would give direct access to some simple PostGIS functions such as “ST_Intersect” or “ST_Within”, giving the user more freedom on it’s data and map. CartoDB should use PostGIS to its full potential, and make it easy to the user to use it while still leaving the freedom for more experimented user to use the SQL Query builder.

Share this:

- Click to share on Twitter (Opens in new window)
- Share on Facebook (Opens in new window)
- Click to share on Google+ (Opens in new window)