

TUTORIAL 7 | JUMP INTO MAPBOX 2

Zoom: **Thursday 06.09, 6:30-7:30pm**
<https://virginiatech.zoom.us/j/2981092726>

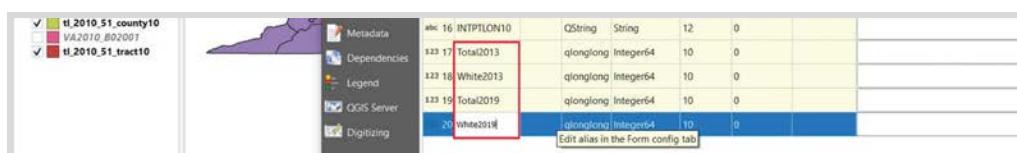
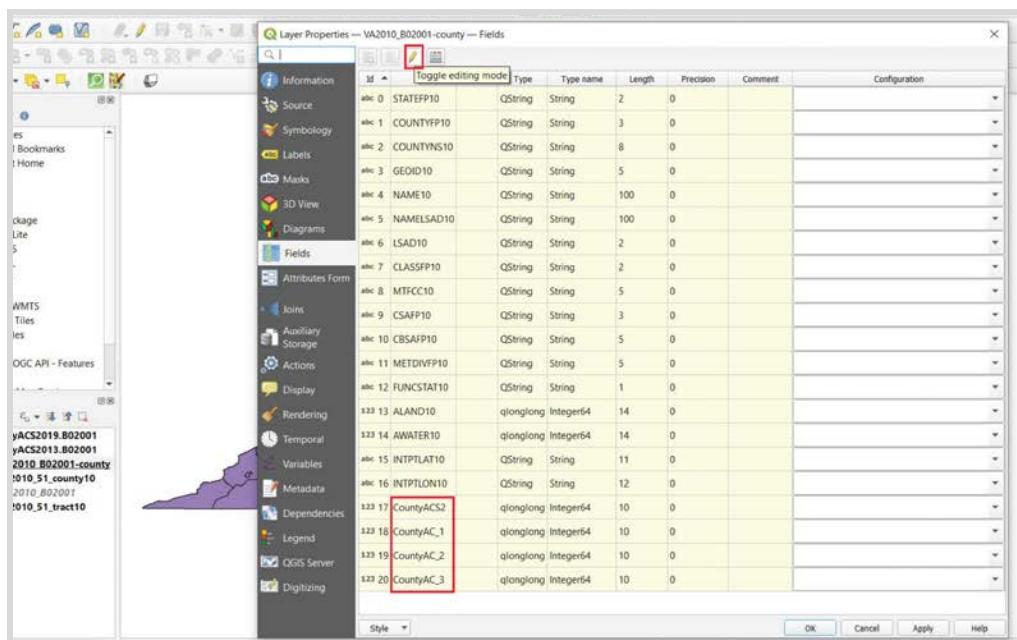
Goals

- Add a new field to QGIS Attribute Table.
- Replace QGIS layer in Mapbox.
- Style Mapbox data by QGIS layer data.
- Code interactive hover feature to display OSM feature attribute data.

Step 1: In QGIS, create a new attribute column for percentage change in nonwhite population between 2013 and 2019.

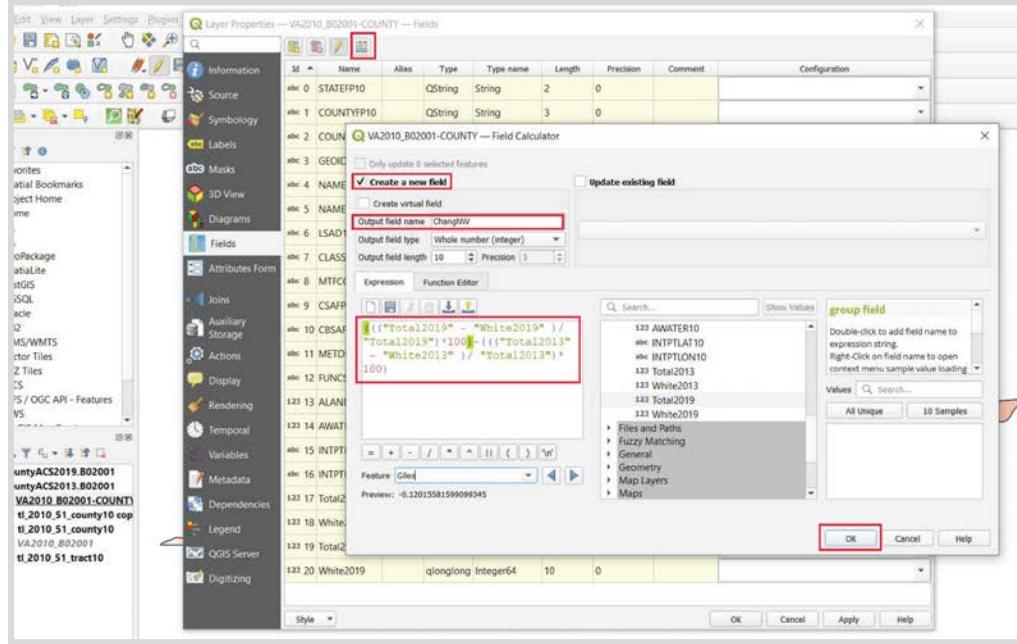
1a Open the **Attribute Table** of your Mapbox census layer, with both ACS tables joined to it. You'll notice that your 4 new columns (total and white 2019, total and white 2013) have been renamed to ACS1, AC_2, AC_3, etc. This is NOT very helpful for knowing which column contains what data.

In **Layer Properties > Fields**, toggle on “edit” and rename these to **Total2013**, **White2013**, **Total2019**, and **White2019** respectively. **Save** your changes and **toggle off edit**.



1b Next, add a new column ("field) using the **Field Calculator** (upper left abacus button in your Attribute Table). Check "Add New Field" and name it something short and simple like "**NWChange**".

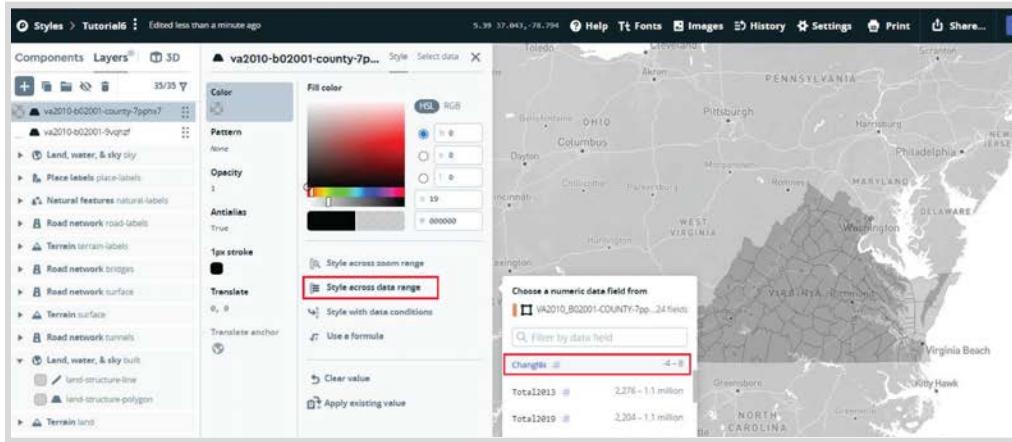
Subtract the NonWhite percentage for 2013 from 2019 to get the change between the two years:
 $((("Total2019" - "White2019") / "Total2019") - ((("Total2013" - "White2013") / "Total2013")) * 100$
 Multiply by 100 at the end to get a percentage.



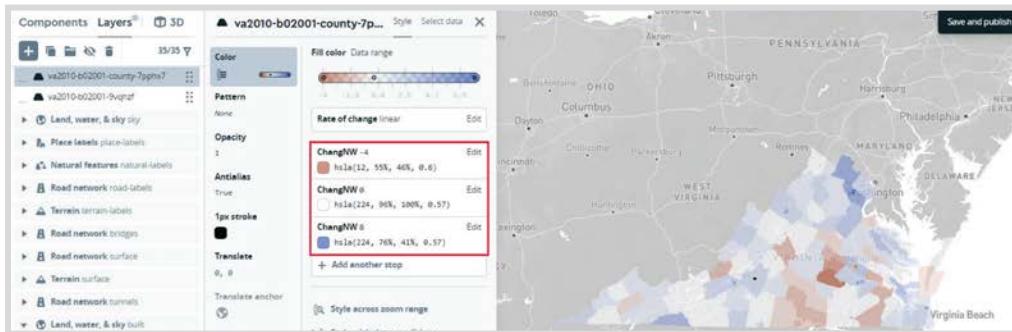
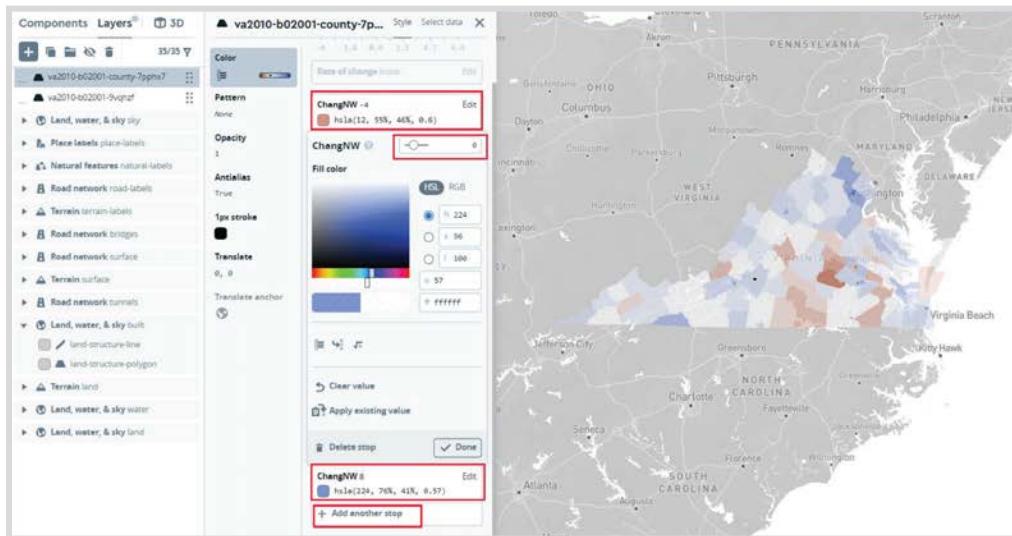
1c Re-export your layer as an Esri shapefile with Pseudo-Mercator projection. **Rezip** and **reupload** your shapefile to Mapbox.

Step 2: In Mapbox, style your newly updated layer by data.

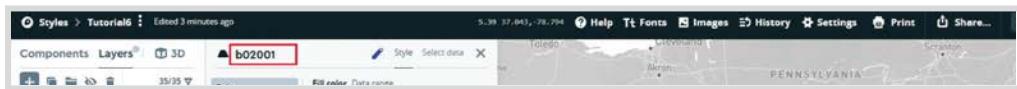
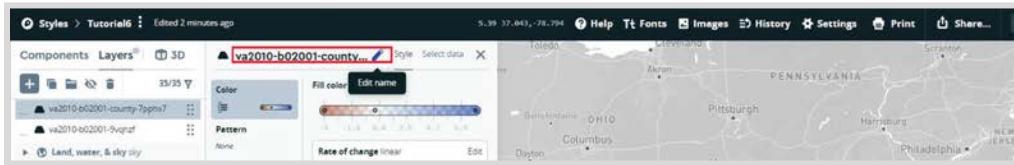
2a In the layer style, select “Style Across Data Range.” Select your new “NWChange” field from the list (scroll down)



2b Choose some endpoint colors (I chose red and blue), and then “Add Another Stop” (below your two colors). Make this one white or some other neutral color, and set the slider at the top to “0” – this color will represent no change between the years (eg 0 percent change).



2c As a last step, **rename your layer** to something short and logical (eg “b02001”).



2d If all looks correct, **Publish** your map.

Step 3: Add an interactive hover feature to display OSM feature attribute data.

One very cool thing about the QGIS- Mapbox-HTML workflow is that you can call and display data from your Attribute Tables on your interactive map. For this next step, you'll add a **popup box** to display the **percent change in non-white population** that appears automatically **on hover** and disappears when you mouse off the feature.

Note: you may not see the Tutorial 7 green comments in your template, but this will not change how the code works. Insert the new code in the place indicated.

3a First, add some styles for the new popup and its text. Copy the following into the **<style>** section:

```
.mapboxgl-popup {max-width: 400px; font: 12px/20px 'Helvetica Neue', Arial, Helvetica, sans-serif;}  
h1 {font: 12px sans-serif; font-weight: normal;}  
h2 {font: 12px sans-serif; font-weight: bold;}
```

```
6   <meta name="viewport" content="initial-scale=1,maximum-scale=1,user-scalable=no" />  
7   <script src="https://api.mapbox.com/mapbox-gl-js/v2.1.0/mapbox-gl.js"></script>  
8   <link href="https://api.mapbox.com/mapbox-gl-js/v2.1.0/mapbox-gl.css" rel="stylesheet" />  
9   <style>  
10    body { margin: 0; padding: 0; }  
11    #map { position: absolute; top: 0; bottom: 0; width: 100%; }  
12    #map canvas {cursor: move;}  
13  
14   /*add popup and legend info, Tutorial 7*/  
15  
16   ADD POPUP STYLE INFO HERE  
17  
18   </style>  
19   </head>
```

3b Second, add some code to the **<script>** section. Below your map controls, add a popup **variable**:

```
var popup = new mapboxgl.Popup({  
  closeButton: false,  
  closeOnClick: false  
});
```

```
34   map.addControl(new mapboxgl.NavigationControl());  
35   map.dragRotate.disable();  
36   map.touchZoomRotate.disableRotation();  
37   map.scrollZoom.disable();  
38   map.addControl(new mapboxgl.FullscreenControl());  
39  
40  
41  
42  
43 //add popup variable (Tutorial 7)  
44 ADD POPUP VAR HERE  
45  
46  
47  
48 //add map.on mouse enter function (Tutorial 7)  
49 ADD MOUSEENTER, MOUSELEAVE, AND ONCLICK FUNCTIONS HERE  
50  
51  
52  
53  
54 ABOVE THE </SCRIPT>
```

Below this, add three new functions: one for **mouseenter** (when you mouse over a county), for **mouseleave** (when your mouse leaves a county), and for **onclick** (when you click a feature). The **mouseenter** will

generate a new popup, `mouseleave` will remove it, and `onclick` will act as a backup to address some glitchy features with the mouseover function.

Note: make sure that you **replace EACH ‘b02001’** with your own Shapefile **layer name** (as you changed it in Step 2c). Likewise, **replace EACH ‘ChangeNW’** with the name of your **Change in Non-White attribute field**, exported from QGIS in Step 1b.

```
map.on('mouseenter', 'b02001', (evt) => {
  map.getCanvas().style.cursor = 'pointer';

  var coordinates = evt.lngLat;
  var change = evt.features[0].properties.ChangeNW;

  while (Math.abs(evt.lngLat.lng - coordinates[0]) > 180) {
    coordinates[0] += evt.lngLat.lng > coordinates[0] ? 360 : -360;
  }

  popup
    .setLngLat(coordinates)
    .setHTML('<h1>' + 'Non-white population change: ' + '<h2>' + change + '%' + '<h1>' + 'between 2013
      and 2019.' //change text here as you like
    .addTo(map);
  });

map.on('click', 'b02001', (evt) => {
  var change = evt.features[0].properties.ChangeNW;
  popup
    .setLngLat(evt.lngLat)
    .setHTML('<h1>' + 'Non-white population change: ' + '<h2>' + change + '%' + '<h1>' + 'between 2013
      and 2019.')
    .addTo(map);
  });

map.on('mouseleave', 'b02001', () => {
  map.getCanvas().style.cursor = 'move';
  popup.remove();
});
```

3c Add a few `console.log` messages to check code works as intended. This is a helpful trick to make sure that your variables are all carrying the right information.

Check that the code is correctly finding the Long/Lat and percent change information that the popup will use (to locate itself and display county information, respectively). To do this, add the following lines of code after their respective variables are defined, as shown:

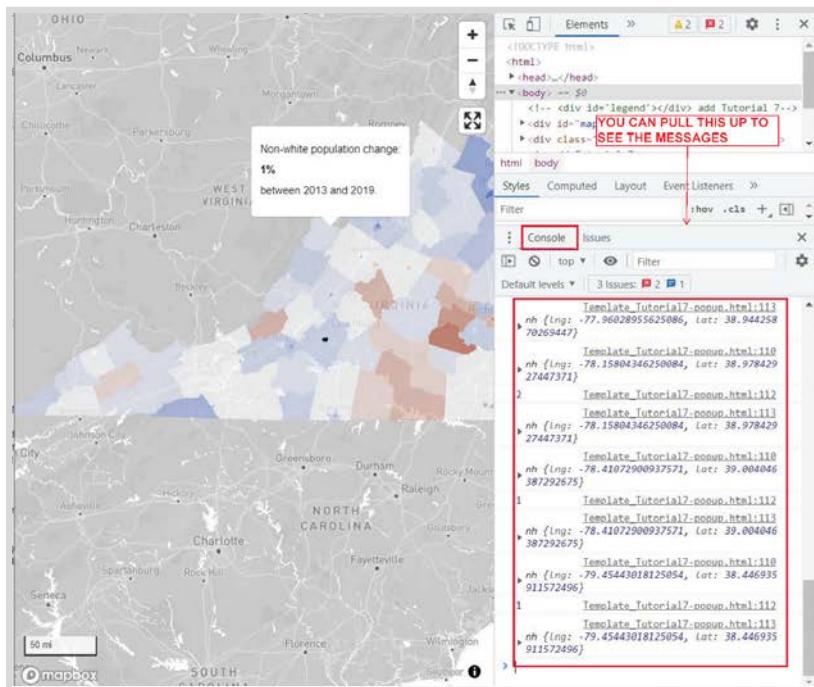
```
console.log(coordinates);

console.log(change);
```

```
49 //add map.on mouse enter function (Tutorial 7)
50 map.on('mouseenter', 'b02001', (evt) => {
51   map.getCanvas().style.cursor = 'pointer';
52
53   var coordinates = evt.lngLat;
54   console.log(coordinates);
55   var change = evt.features[0].properties.ChangNW;
56   console.log(change); //check
57 }
```

We're only using this for reference, and so you can see the web page Inspector console, but keep in mind that it's a helpful trick for troubleshooting your code later on. You can add a `console.log` to check behavior before breakpoints and errors that you don't understand.

3d Save your code, and reload your Chrome preview page. To check the console messages, right click on the Chrome webpage that loads your map and “**Inspect**”. In the bottom window, select the “**Console**” tab. Here, you should see log messages appear when you trigger the popup functions by hovering over counties.



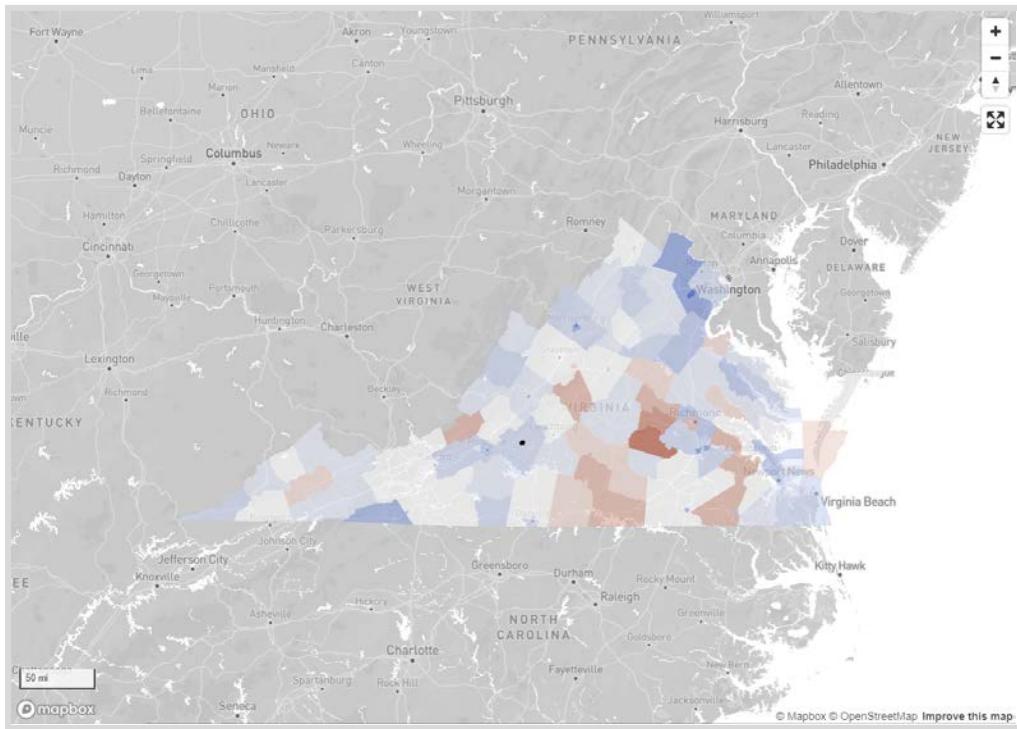
Step 4: Add legend (via mapbox layer) and scale bar

4a After the map controls (before the *var popup*), add:

```
map.addControl(new mapboxgl.ScaleControl({position: 'bottom-right', unit: 'imperial'}));
```

```
34     map.addControl(new mapboxgl.NavigationControl());
35     map.dragRotate.disable();
36     map.touchZoomRotate.disableRotation();
37     map.scrollZoom.disable();
38     map.addControl(new mapboxgl.FullscreenControl());
39
40 //add scale bar (Tutorial 7)
ADD SCALE BAR HERE
42
43 //add popup variable (Tutorial 7)
```

4b Reload and check out the new scale at the bottom left of your map.



- Bonus -

Step 5: Layers of those years can be toggled on and off (by default map only shows percentage change):

<https://docs.mapbox.com/mapbox-gl-js/example/toggle-layers/>