GeoAnalytics Enablement

Exercises

September, 2018

Renato Vieira
rvr@qlik.com

Table of Contents

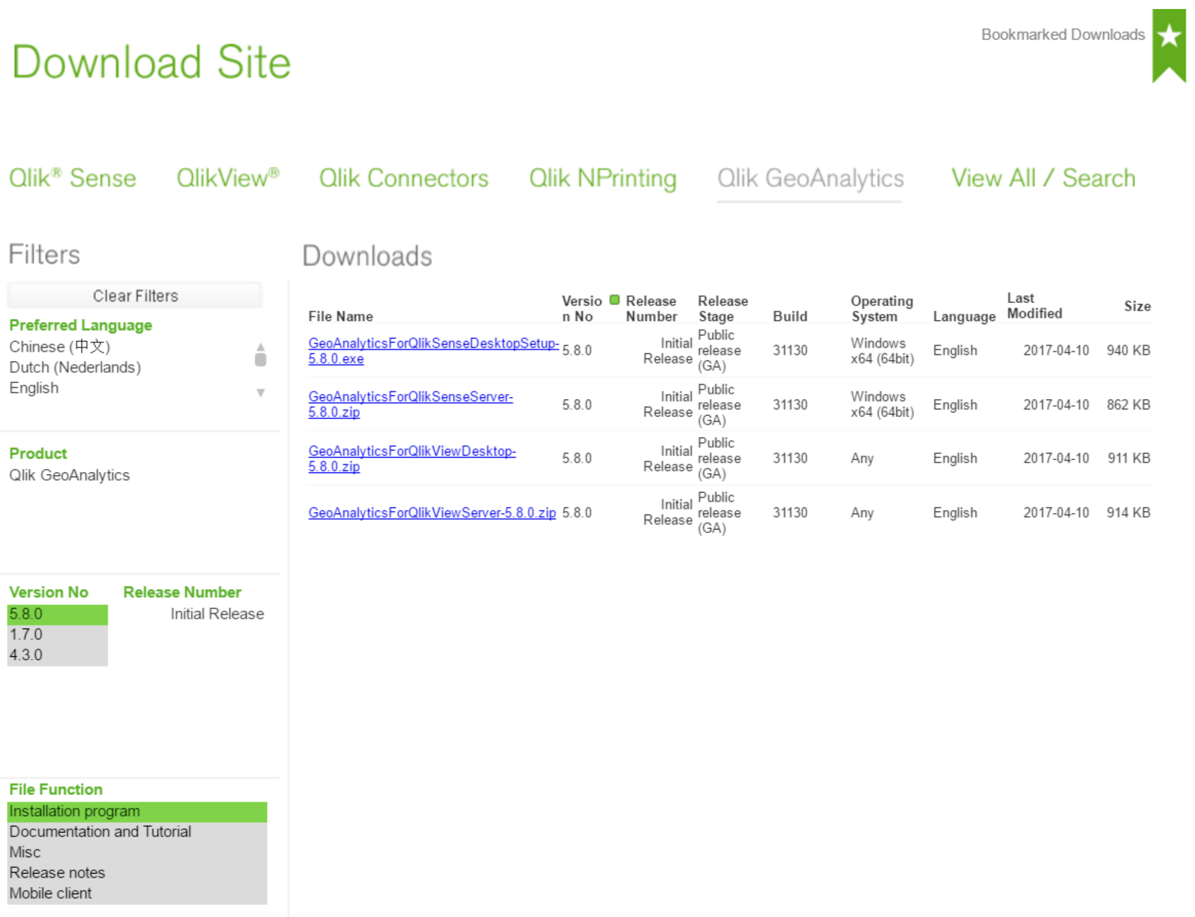
Version History

<table>
<tr><th>Date</th><th>Contents</th><th>Author</th></tr>
<tr><td>13/04/2017</td><td>Initial Version:
Exercises Part 1, Exercises Part 2 and Additional Guidelines</td><td>Renato Vieira</td></tr>
<tr><td>25/09/2018</td><td>Updated
Exercises Part 1 and 2 – added notes regarding official documentation
Additional Guidelines: Enablement Materials
Added:
Exercises Part 3
Mobile “Near me” Location</td><td>Renato Vieira</td></tr>
</table>

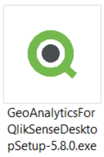
Installing GeoAnalytics

In this section, we will quickly cover how to install GeoAnalytics on your machine with Qlik Sense. My recommendation is to use it on a server version, but Qlik Sense Desktop can work as well.

Download the Qlik GeoAnalytics software from our download site (http://eu-a.demo.qlik.com/download/) and select the most recent version. At the time of the redaction of this document it's version 5.8.0.

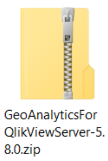


Execute the installer and follow the guided steps.



For Sense Deskop:

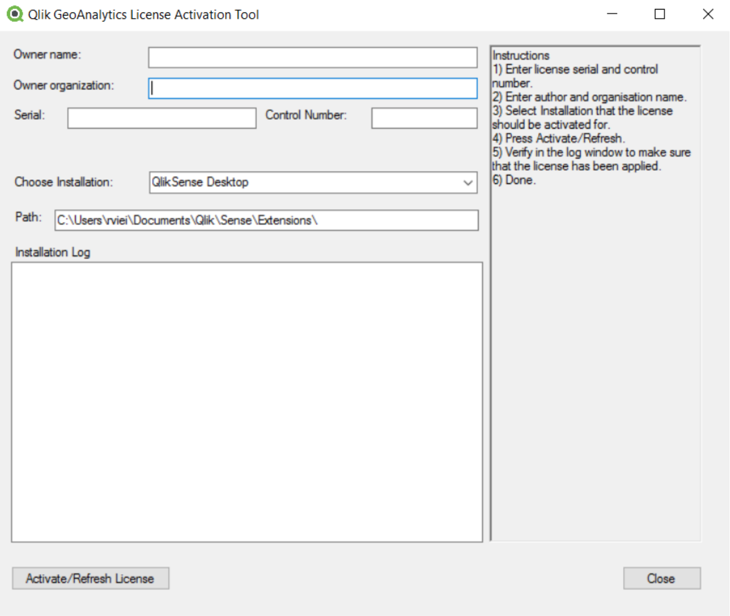
- accept the default path suggestions for both the GeoAnalytics Maps (Extensions) as well as the GeoAnalytics Connector.
- Accept the default URL suggestions for both the GeoAnalytics Maps and GeoAnalytics Connector servers



For Sense Server:

- Unzip the file and import the extensions via QMC
- Execute the GeoAnalyticsSetup_5.8.0.exe file to install GeoAnalytics Connector
- Accept the default URL suggestion for the GeoAnalytics Connector servers

Apply your internal license when the popup appears. You may get your license here at our Internal Licenses Page:
https://q.qliktech.com/Intranet/Departments/Products/RnD/QlikView%20Licences.aspx#4QlikGeoAnalytics



Important note: **select the correct product** in the *Choose Installation* option to avoid malfunction such as *Invalid Key* error when adding a map to your application.

Exercises Part 1 – Using Extensions and Location Services

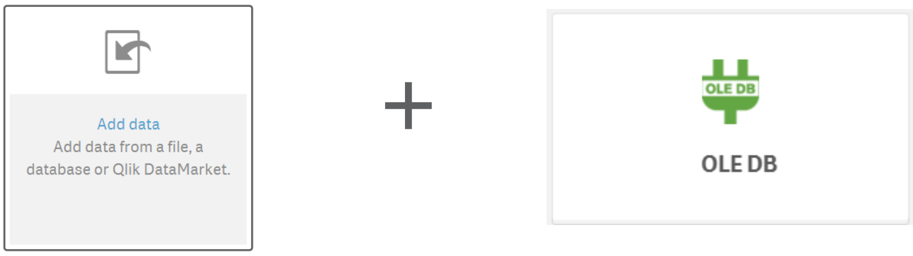
In this section, we will cover a few exercises to let you know the extensions a little bit. No GeoAnalytics connector will be used in here. Only direct requests to our hosted server using the location services intelligence already developed within the extensions.

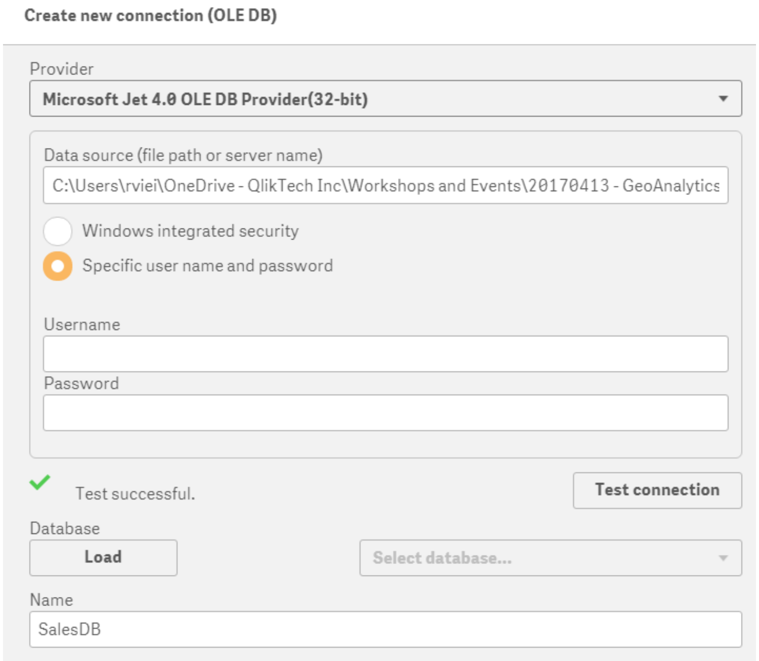
> Note: always check the GeoAnalytics documentation page for full details of the product's capabilities: https://bi.idevio.com/products/idevio-maps-for-qlik-sense/documentation
>
> The *Exercises Part 1* uses contents available with more detail at the following sections:
>
> - User Guide: https://bi.idevio.com/wp-content/qlik/qliksense/releases/IdevioMapsForQlikSense-5.13.0/user_guide-September_2018.html
> - Properties: https://bi.idevio.com/wp-content/qlik/qliksense/releases/IdevioMapsForQlikSense-5.13.0/imap_reference-qs-September_2018.html

Exercise 1: Preparing application data

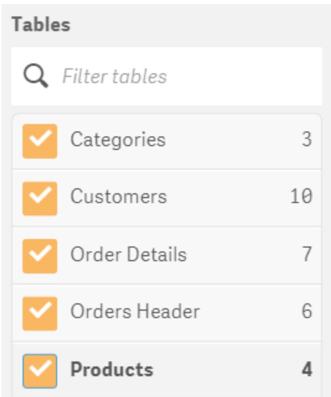
We will create an application from scratch, so you can get familiar with the data. This is a standard application used for workshops and activities on the likes. The data represents an example of a clothing stores franchise that is multi-region. Its database, in the Sales.mdb file, contains all their sales history.

Start by **creating a new application and connecting to the Sales.mdb** file via OLE DB Jet 4.0. This database is open, so there is no need to specify administrators nor passwords. We will use the Bubbles Interface to speed up the datamodel creation process.

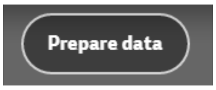




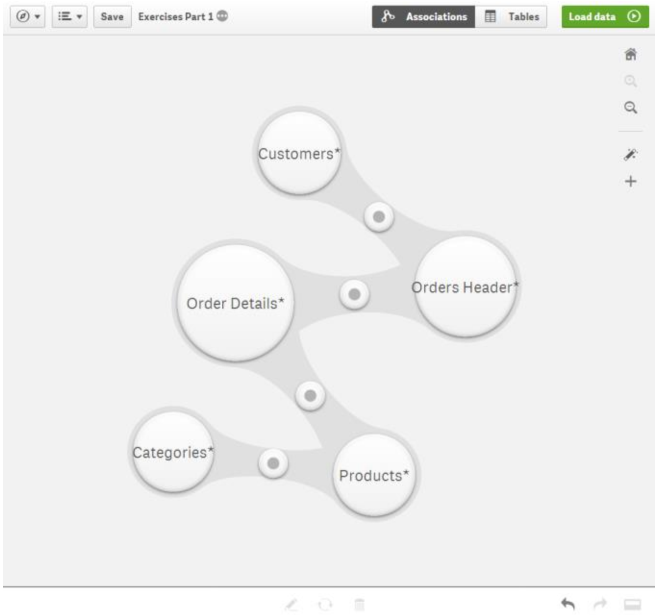
Select the following **tables to load**:



You may load the tables with all the fields. Once the tables are selected choose the **Prepare Data** option.



Click on the *Magic Wand* so the data preparation functionality suggests the datamodel. You should get the same result as indicated in the image below:



Load the data to your application.

Once the data is loaded we will quickly **create a sheet** that has some fields and KPIs. This way we will be able to see the associative model work and how it interacts with the map functionalities we will fiddle with.

Add **4 filter panes**:

- Country
- City
- NomeCat
- ProductName

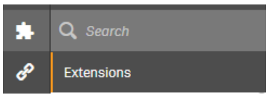
Add **2 KPIs** (recommended via Measures Master Item)

- # of customers - *count(Customer)*
- Sales - *sum(Sales)*

Add **1 Line chart** with the sales trend.

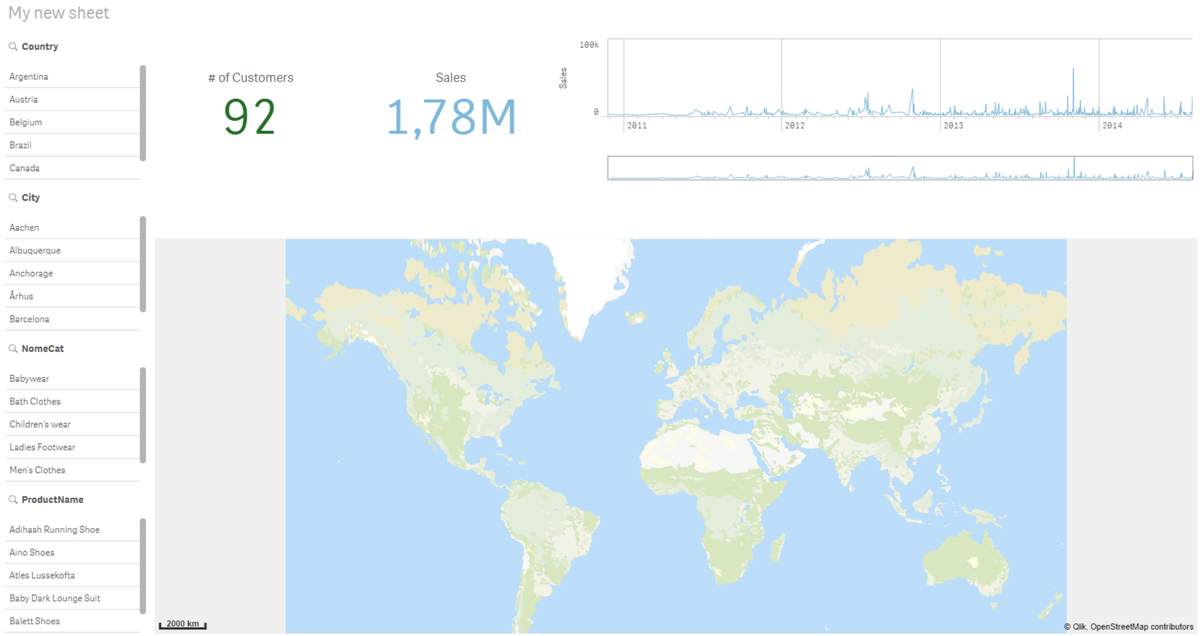
- Dimension: *OrderDate*
- Measure: *sum(Sales)*

Add the **GeoAnalytics Map extension** to your sheet.



+

Here is an **example** of what your sheet could look like:

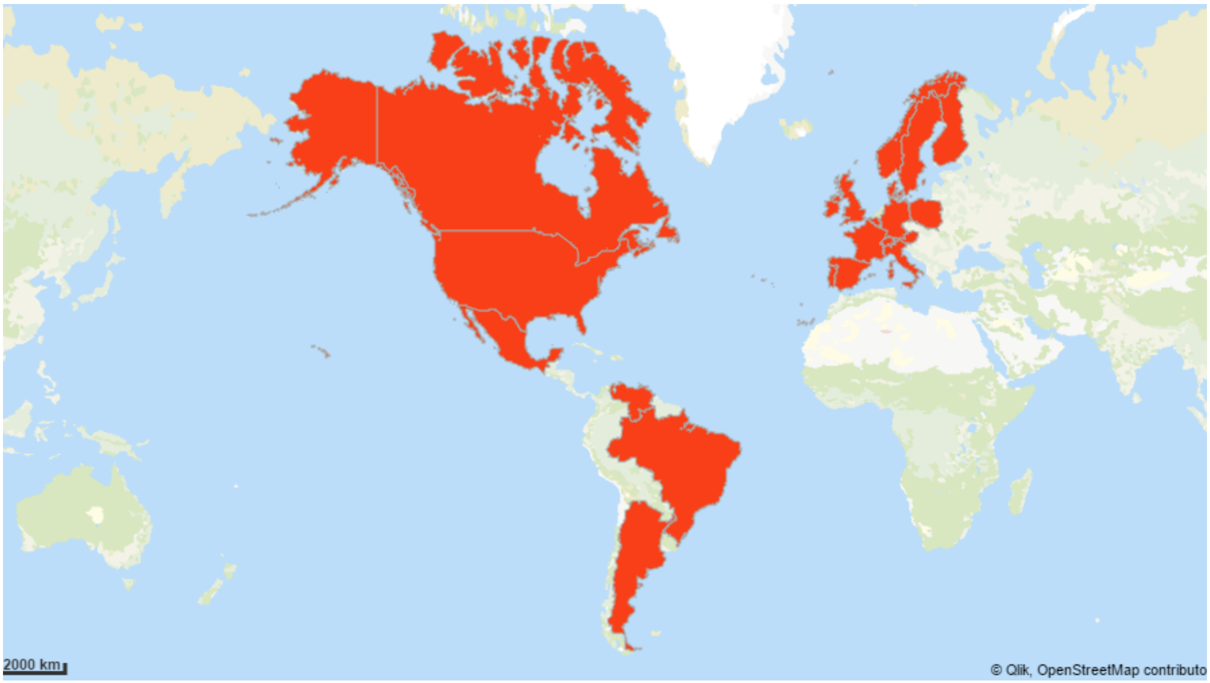


Exercise 2: Area Layer to display Countries

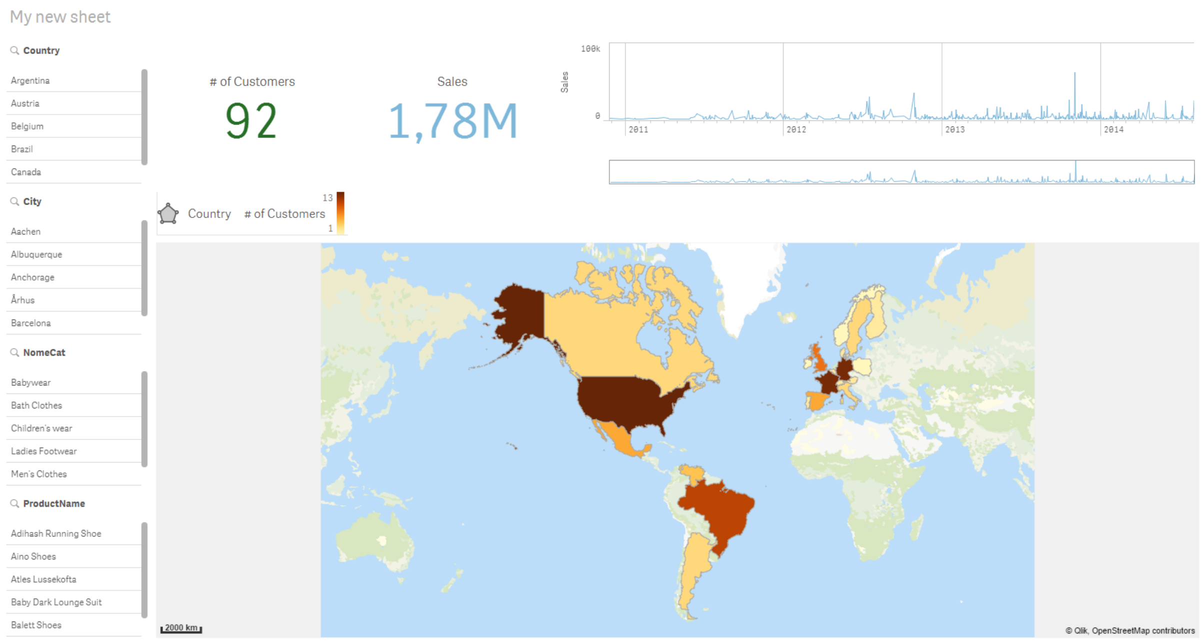
This exercise will allow you to add a layer to your map: The Area Layer. The objective will be to draw all the countries where customers exist and color the area by the number of customers.

Add the **GeoAnalytics Area Layer** to your sheet and drag the **Country** field to it.

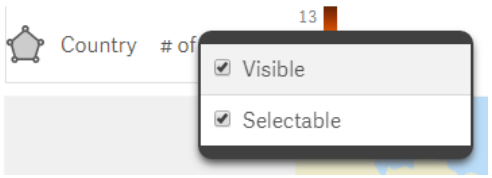
You will immediately see on the map the areas of all the Countries that exist in the Country field.



Now we can **color** all these countries based on a measure. Go to the GeoAnalytics Area Layer extension properties, select **Appearance**, expand **colors** and **deactivate** the ***Auto*** option. **Select** *By* *measure*. If you have created a *# of customers* Master Item select it, otherwise add the *count(Customer)* expression. Here is how your final result should look like:



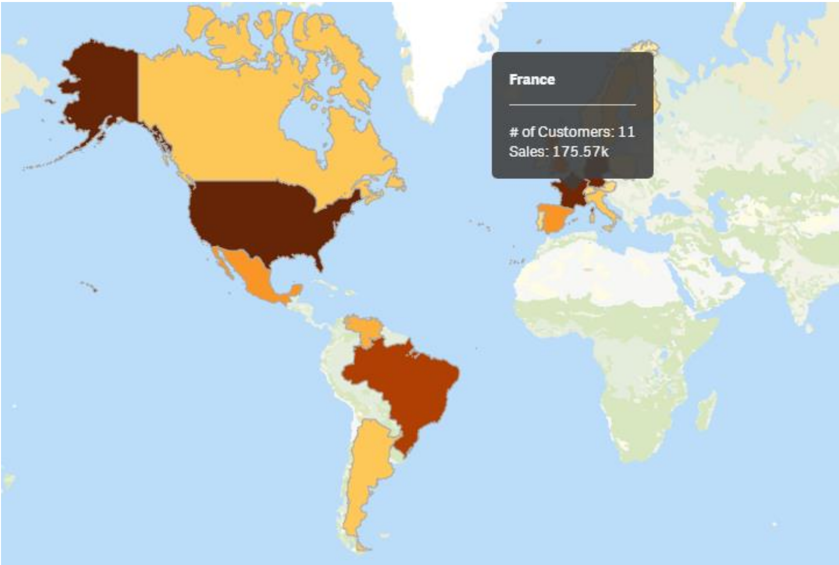
Notice that **you can make this layer visible or not** at will by clicking on the extension and checking/unchecking the *visible* checkbox.



There are also many options to customize the appearance and behavior of the extension. One example could be to **customize the Info Bubble** that appears when a user hovers the area. Go to the GeoAnalytics Area Layer extension properties, select **Appearance,** expand **Info Bubble** and **deactivate** the ***Auto*** option. Note: If you get the *Add measure to use expression* warning message go to the **Location ID** pane, add **Country** as a measure (only the field, no aggregations) and return to the previous Info Bubble configuration area. Click on the ***fx*** option to open the full expression editor and **add a custom expression** at your choosing. **The Info Bubble is affected by HTML tags** to customize its layout and you can even add external images. Add an expression like:

'<h1>'&Country&'</h1><hr /># of Customers: '&count(Customer)&'
Sales: '&sum(Sales)

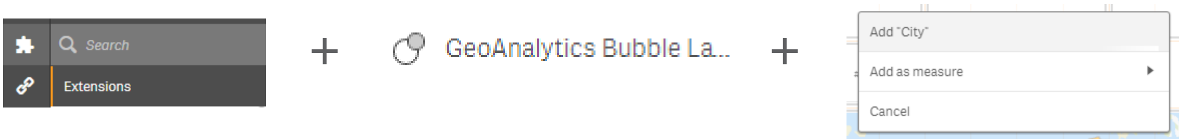
And it is expected you get an output like the image bellow when you **hover a country**:



Exercise 3: Bubble Layer to display sales per customer

This exercise will allow you to add a layer to your map: The Bubble Layer. The objective will be to draw all the customers' cities on the map and both color as well as affect the bubble size based on the overall sales made to that city.

Add the **GeoAnalytics Bubble Layer** to your sheet and drag the **City** field to it.



You will immediately see on the map the bubbles of all the Cities that exist in the City field.

At a first glance, this representation may look good. But there is always a chance that a City name can be the same in more than one Country. To make sure that we are addressing to the representation of the correct City we can add some control to what is sent to the Location Service.

To **improve our representation accuracy,** we will create a measure that sends a <City, Country> representation instead of only sending a City name.

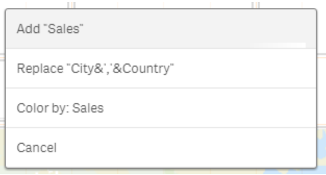
Go to the GeoAnalytics Bubble Layer extension properties, select **Location,Size** and click on **Add measure.** Add the following expression to concatenate the City to Country separated by a comma:

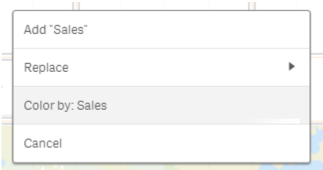
City&','&Country

And compare the result:

For more details about the Location Options specifications you can refer to https://bi.idevio.com/wp-content/qlik/qliksense/releases/IdevioMapsForQlikSense-5.13.0/imap_reference-qs-September_2018.html#Bubble_20Layer

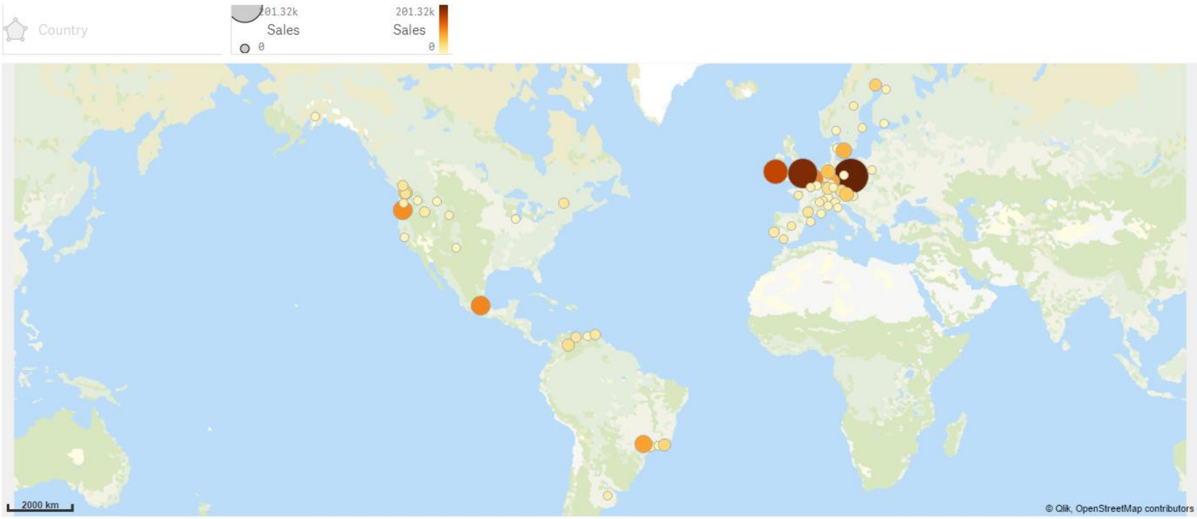
Now that we are representing the cities correctly, we can **add color and size** to the bubbles, based on the overall sales for each city. To do so, an easy way to do it is to drag your *Sales* Master Item to the GeoAnalytics Bubble Layer extension. If you select *Add Sales* it will affect the **size**. If you select *Color by: Sales* it will affect the **colors.** Drag both times to have the desired effect.



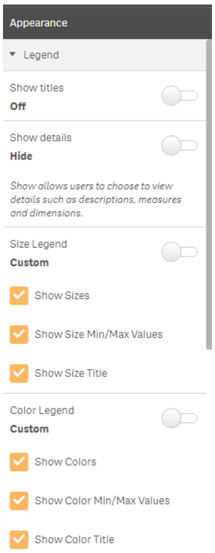


+

Notice that the extension is autoamtically adding some legend to help understand both the bubble size and color meanings.

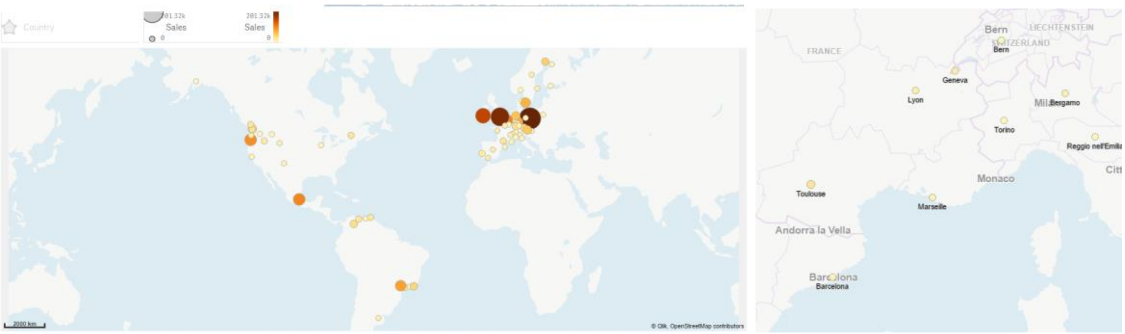


You may customize this at will on **Appearance**, **Legend** properties of the extension.



You can also add a **label next to the bubble**, so it can display the corresponding City. Select the **Appearance**, **Label** properties pane of the extension and **type in an expressi**on of your choosing, such as *City* to represent the value of the field. You may also affect the behavior so this label only appears upon a specific level of **zoom.** Try to select something in the range of 3000 and scroll over your map.

If you feel that the map is getting too confusing due to its underneath layering representing too much information such as City names, roads, etc. you can change the **level of detail of representation of the map background.** Click on the map to get its **properties pane** available, go to **Map Settings** and change the **Base Map** representation to **Plain Map**.

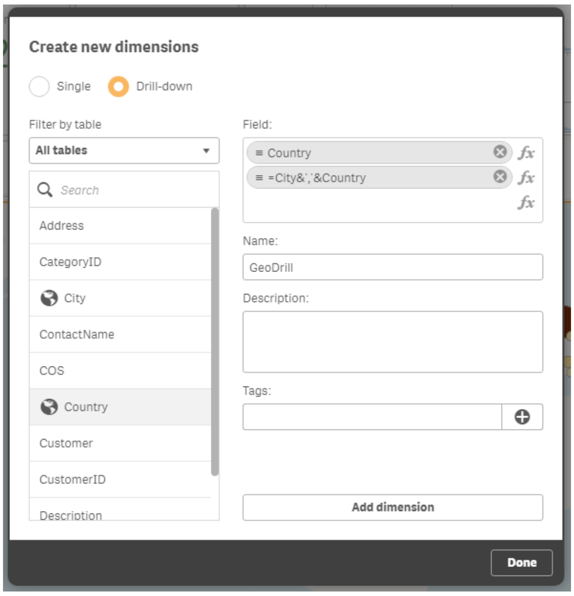


Exercise 4: Drill from one layer to another based on user interactions

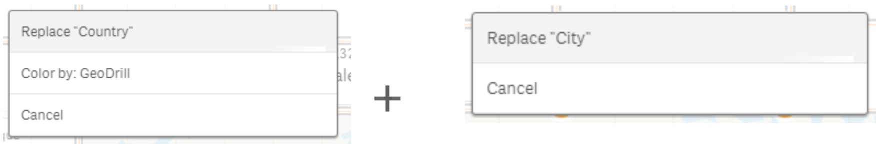
Now that we both have the Country areas representing the number of customers and the City bubbles representing the overall sales we can define some behavior such as: display by default the Countries areas and once a selection is made, zoom in and display all the City bubbles for that selection.

This can easily be done by defining a **Drill Dimension** and setting up a behavior in each extension to **only display a specific Drill Level.**

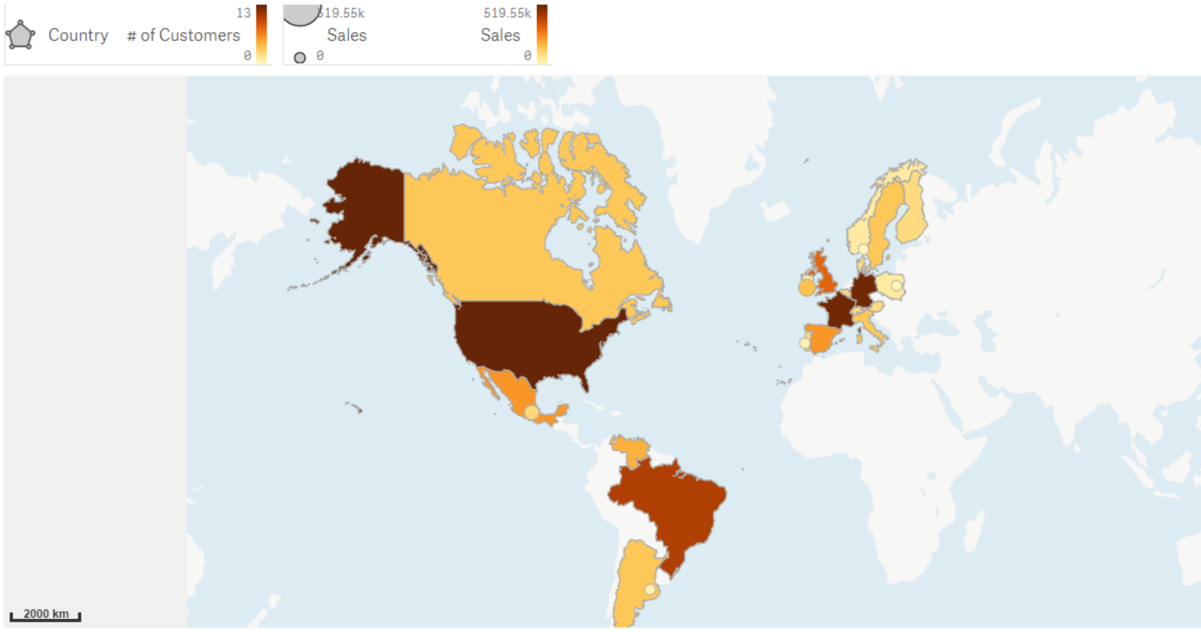
Let's start by creating our Drill Dimension on the Master Items with Country and *<City,Country>.*



And now we **replace the dimensions** of our GeoAnalytics Area Layer and GeoAnalytics Bubble Layer with our newly created Drill Dimension.



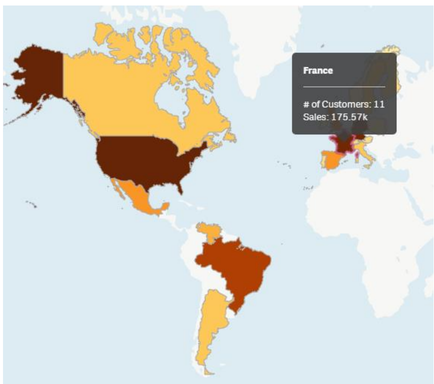
Make sure both layers are now with the *Visible* mode activated and you should have the following temporary representation.



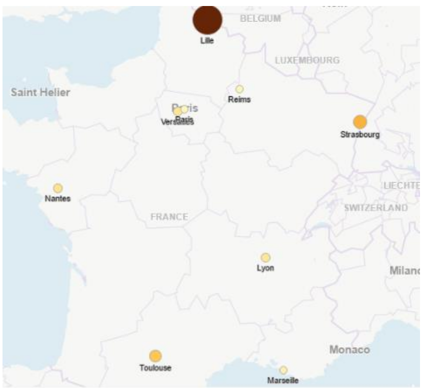
We will now setup which drill level should each extension react to:

- Go to the GeoAnalytics Area Layer **properties**, select **Layer Options** and set **Restrict Drill Down** to 0
- Go to the GeoAnalytics Bubble Layer **properties**, select **Layer Options** and set **Restrict Drill Down** to 1

You will now see that depending on the drill level, one of the extensions will be grayed out to let you know it's inactive. Select a Country, for example, and you will see that the bubbles of that Country will now be displayed.



>

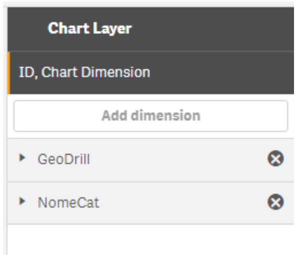


Exercise 5: Chart Layer to display sales by product category on the map

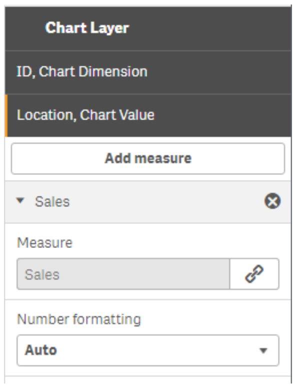
As an alternative to the bubbles representation, we can add mini-charts right on top of the map, that can bring an extra level of information. In this specific case, we will display the amount of sales per product category, affected by the drill level the user has selected at the time.

Add the **GeoAnalytics Chart Layer** to your sheet and drag the **GeoDrill** Master Item Dimension to it.

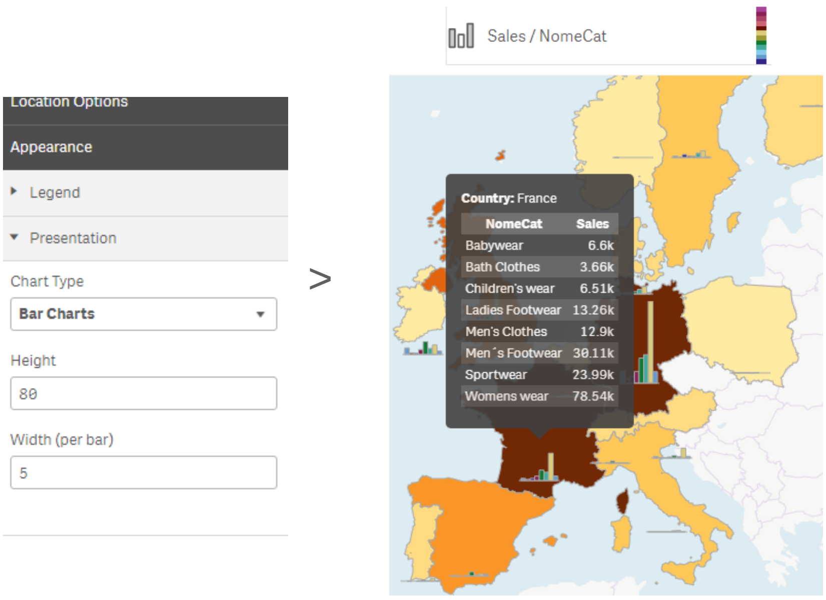
Go to the extension properties and select the **ID, Chart Dimension** pane. The GeoDrill will take care of positioning the charts, but we still need to indicate what is the dimension of the charts we want to display. Add **a second dimension** with the NomeCat field.



As a second configuration step, we now need to indicate the measure for our charts. Just drag the Sales Master Item Measure or manually add the *sum(Sales)* expression on the **Location, Chart Value** pane.



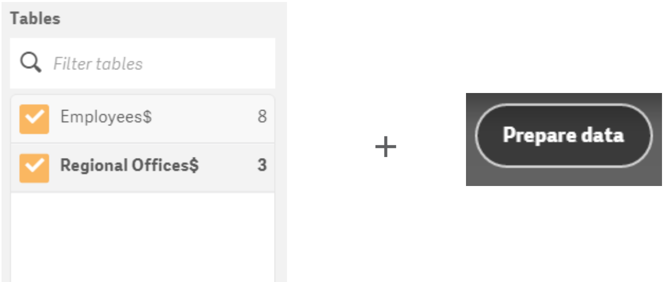
Finally, you may setup the type of chart to display: **Pie or Bar**. Go to the **Appearance** pane and select **Presentation**. Fine tune your charts at your liking.



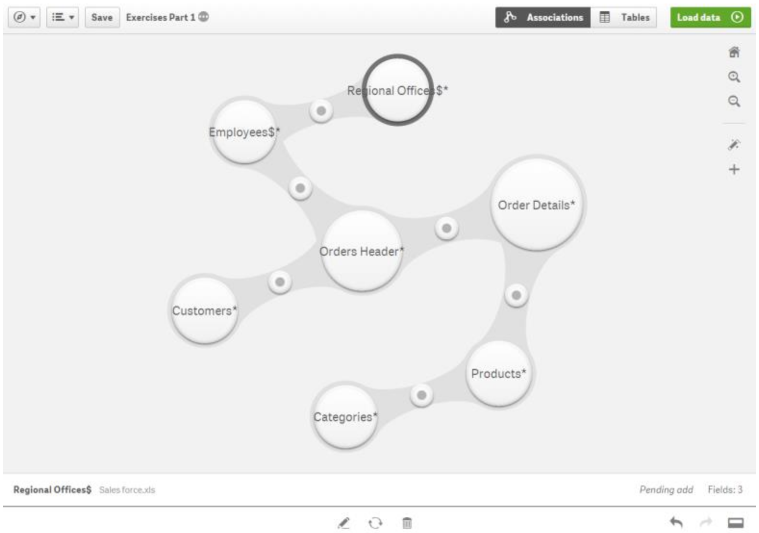
Exercise 6: Line Layer to shipping from warehouses to customers

As the last exercise for this first part we will import the location of our warehouses and look into where their shipping is made to and display the lines width affected by the sales representation of each trajectory. For practical reasons of this exercise, we will assume that the sales offices and the warehouses are the same.

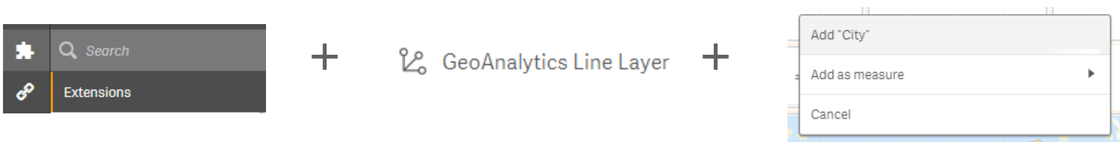
Let's start by **importing an additional excel file**, named Sales force.xls, to our datamodel by dragging it to our application. Import both of the available tables: Employees$ and Regional Offices$. Use the Prepare Data functionality to speed up the process as there are some field names that don't match with our existing datamodel.



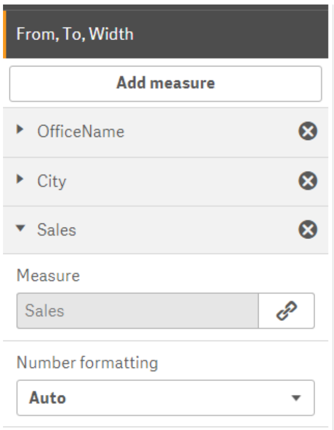
Click on the *Magic Wand* so the data preparation functionality suggests the datamodel. You should get the same result as indicated in the image below:



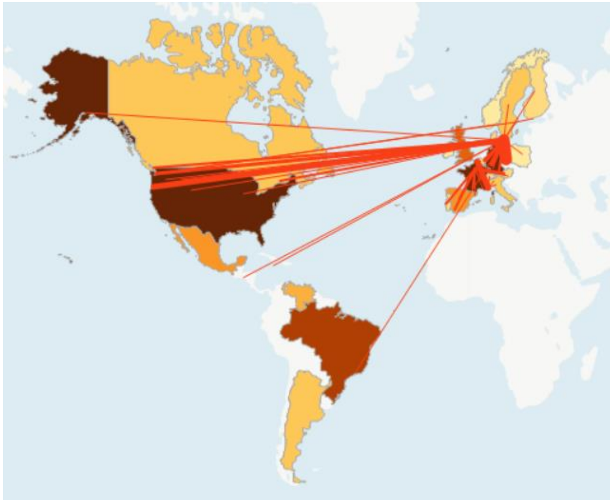
Now we will go back to our sheet and add the **GeoAnalytics Line Layer** extension and drag City as our relationship field to be used as ID/Dimension.



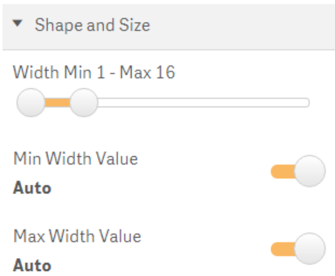
This object needs us to identify the origin and the destination of the line to display. For that we will go to the **From, To, Width** pane to configure that information as well as affect the line's width with the overall of sales. Add *OfficeName* as Origin, *City* as Destination and the *Sales Master Item* (or sum(Sales)) as Width.



>



At last, but not least, we can fine tune our lines representations. For that let's go to the **Appearance** pane and select the **Shape and Size** pane. Add some **max width**, i.e. 16.

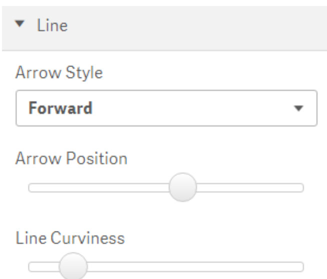


Next, we will affect the lines **colors**. Our customer would like to understand flows between US and Europe.

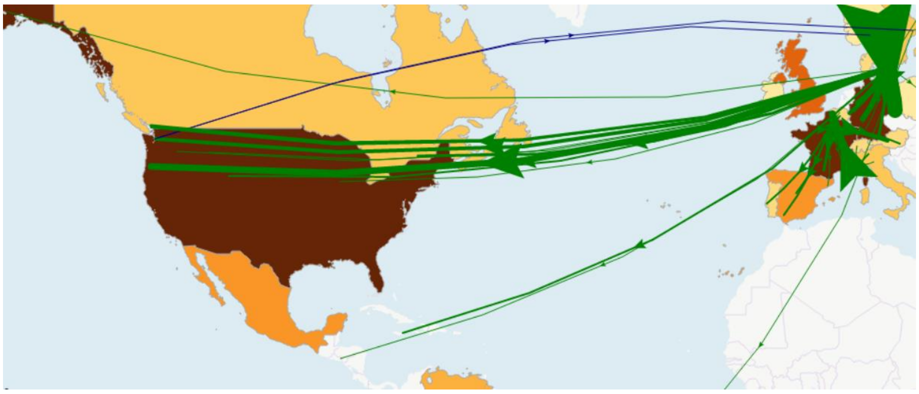
Go to the **Colors** pane, deactivate the *Auto* option and select **By expression** in the dropdown and add the following expression

if(Region='US',blue(),green())

For a last touch, we will add some **line curviness** and **arrows** to show the direction of the flow. Go to the **Line** pane and select **Forward** for the **Arrow Style** and add some **Line Curviness** as well as define the **Arrow Position** of your liking.



Here is an example of the final result. As we can quickly see, there are a lot of shipments from Europe to US and a few from US to Europe, which certainly can be looked at.



Exercises Part 2 – Using the GeoAnalytics Connector

In this section, we will cover a few exercises to let you know the GeoAnalytics Connector a little bit. We will connect to external sources that have geographic information and we will build an application that takes advantage of a few functions available from the GeoAnalytics Connector, such as TravelArea, Binning and Within.

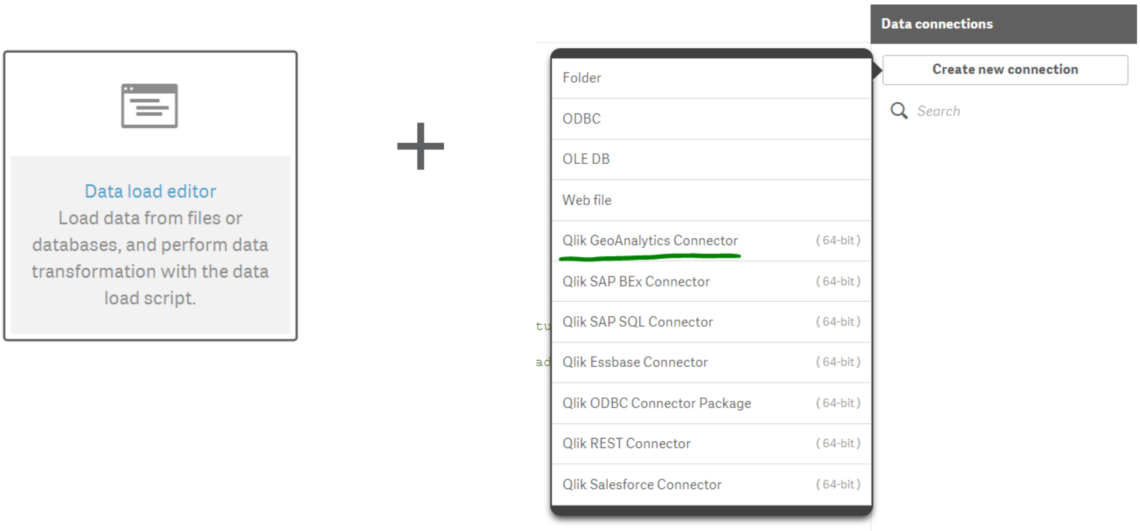
Exercise 1: Load geographical data from an external source (GeoJSON)

We will create an application from scratch, so you can get familiar with the data. This time we will load data that is in the public domain in a GeoJSON format. The source is Lisboa Aberta and it this belongs to the Open Knowledge International initiative.

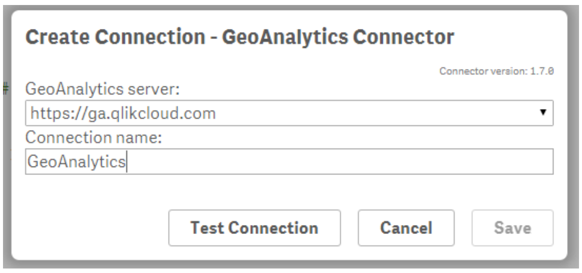
We will use the Data Load Editor combined with a GeoAnalytics Connector connection.

> Note: always check the GeoAnalytics reference page for full details using the Load Operation: https://bi.idevio.com/wp-content/qlik/geoanalytics/releases/IdevioGeoAnalyticsConnector-5.13.0/doc/geoanalytics_reference-September_2018.html#Operation.Load

Start by entering the **Data Load Editor** and **Create a new connection** to your **Qlik GeoAnalytics Connector**.



Accept the **default URL** for the GeoAnalytics server and add a **Connection Name**. You may also click on **Test Connection** to check if your connection to our hosted server is working.



We will now load a GeoJSON file available at Lisboa Aberta's site that contains all the water spots of the city (drinkeable, fountains, decorative, etc.). The direct access to the GeoJSON is via the following URL:

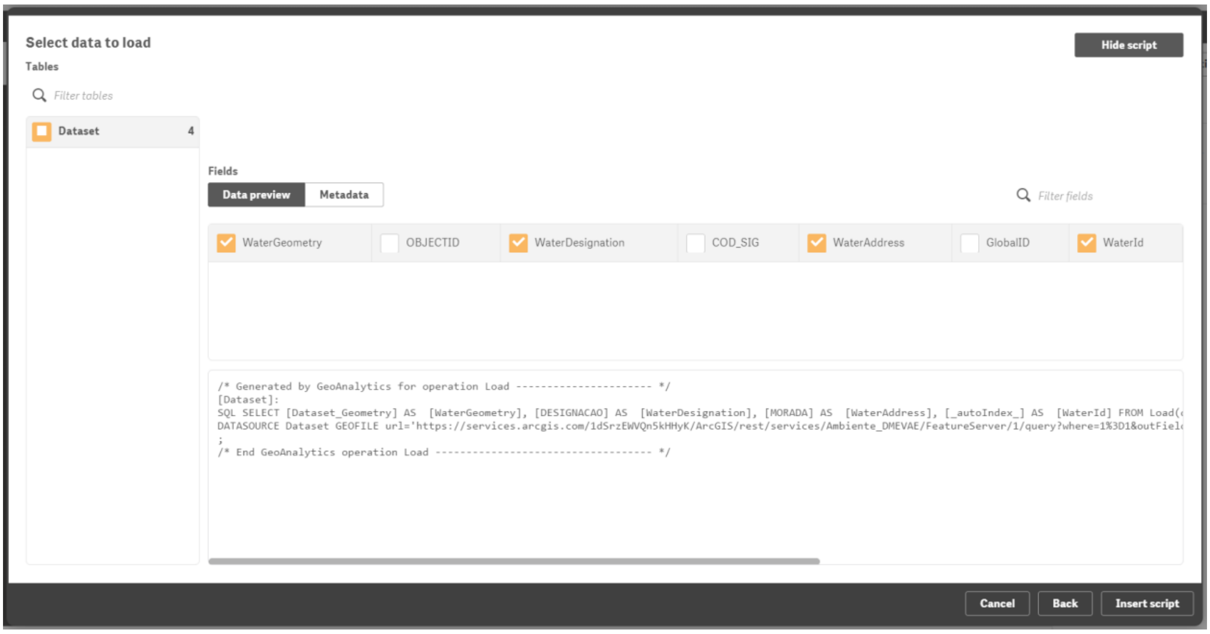
https://services.arcgis.com/1dSrzEWVQn5kHHyK/ArcGIS/rest/services/Ambiente_DMEVAE/FeatureServer/1/query?where=1%3D1&outFields=*&f=pgeojson

On your Qlik Sense **Data Load Editor** click on the **Select data** icon ( ) of your GeoAnalytics Connection and select the **Operation: Load**. Give *WaterRaw* as **Name** for your *Dataset*, select *File based geo dataset* for **Type** and paste the link in **URL.** You may leave all the other options as suggested by default and click on **Next**.

The Load operation of the GeoAnalytics connector is a function that simply loads datasets, where you can easily specify the source and quickly add geographic data to your application. As you can see there is a question mark next to some areas of the wizard. These question marks allow you to jump directly to the help page so you can understand what is expected to do for each operation configuration possibilities.

Once you click on **Next** the connector will check if the parameters are ok and give you access to the **Select data to load** window you are familiar with. Depending on the operation used, more tables can be presented and by clicking on **Insert Script** the required script will automatically be generated for you.

For our application, we don't need all the fields available from the GeoJSON. You may **rename and select** *Dataset_Geometry* to *WaterGeometry,* **rename and select** *DESIGNACAO* to *WaterDesignation*, **rename and select** *MORADA* to *WaterAddress* and **rename and select** *_autoIndex_* to *WaterId*. Once you've selected and renamed the fields you may click on **Insert Script**.



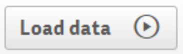
The following script is expected to have been generated for you:

LIB CONNECT TO 'GeoAnalytics';

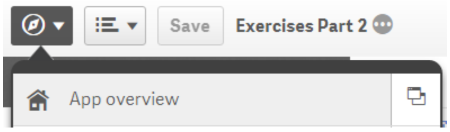
/* Generated by GeoAnalytics for operation Load --------------------- */
[WaterRaw]:
SQL SELECT [Dataset_Geometry] AS [WaterGeometry], [DESIGNACAO] AS [WaterDesignation], [MORADA] AS [WaterAddress], [_autoIndex_] AS [WaterId] FROM Load(dataset='Dataset')
DATASOURCE Dataset GEOFILE url='https://services.arcgis.com/1dSrzEWVQn5kHHyK/ArcGIS/rest/services/Ambiente_DMEVAE/FeatureServer/1/query?where=1%3D1&outFields=*&f=pgeojson', keyField='', type='auto', expectedGeomType='auto', encoding='', crs='Auto'

/* End GeoAnalytics operation Load ----------------------------------- */

Load your data and test it out by creating a **new sheet**.

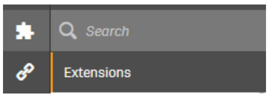


+



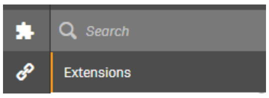
+

Add the **GeoAnalytics Map extension** to your sheet.



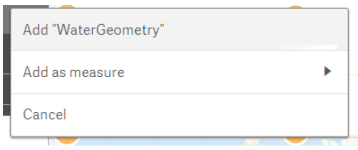
+

Add the **GeoAnalytics Bubble Layer** to your sheet and drag the **WaterGeometry** field to it.

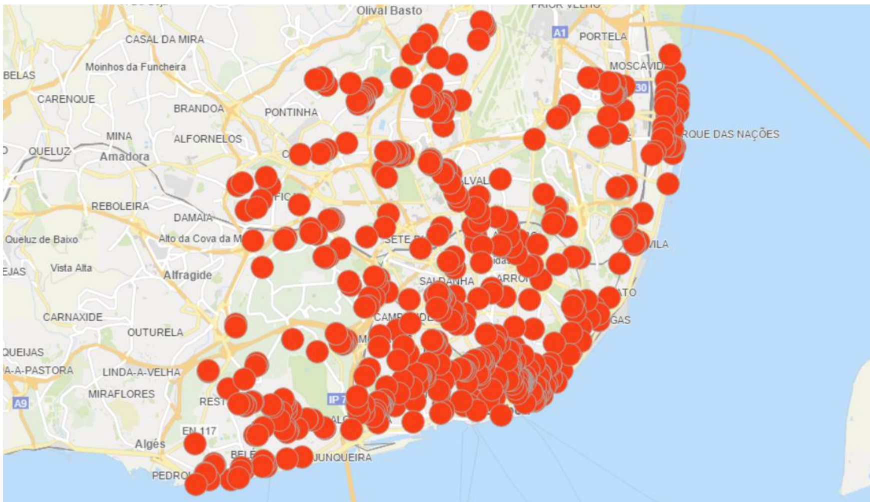


+

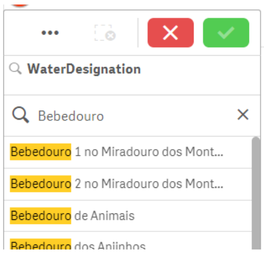
+



You are now able to see all the water spots available in Lisbon, Portugal.



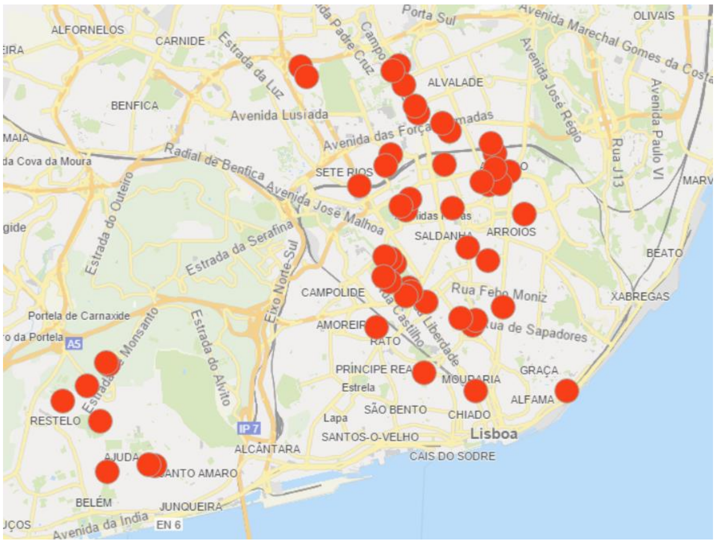
We need to reduce this number of spots to the information that is really relevant to us: the drinking water spots. Those points are easily identifiable from the *WaterDesignation* field. They are all the descriptions that start with *Bebedouro.*



Let's go back to our **Data load editor** and fine tune our script to restrict the data we want in our data model by creating a new table based on our **resident** and restrict to *WaterDesignation* **like** *"Bebedouro*".* You may copy the following script snippet to speed up your script creation.

```
DrinkingWater:
NoConcatenate
Load WaterId,
    WaterAddress,
    WaterDesignation,
    WaterGeometry
resident WaterRaw
where WaterDesignation like 'Bebedouro*';

drop table WaterRaw;
```

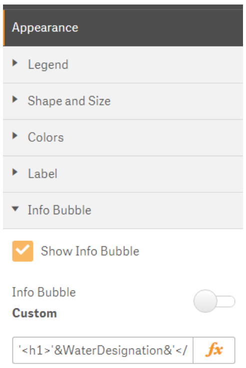
You should now have loaded only 52 lines of data and your map will look like this:



Now let’s make these spots more user friendly by giving them a **Symbol** and adding some information in their **Info Bubble.**

Starting with the **Info Bubble** configuration, go to the GeoAnalytics Bubble Layer extension properties, select **Appearance,** expand **Info Bubble** and **deactivate** the ***Auto*** option. Note: If you get the *Add measure to use expression* warning message go to the **Location ID** pane, add **WaterGeometry** as a measure (only the field, no aggregations) and return to the previous Info Bubble configuration area. Click on the ***fx*** option to open the full expression editor and **add a custom expression** at your choosing to display at least the *WaterDesignation*. **The Info Bubble is affected by HTML tags** to customize its layout and you can even add external images. Add an expression like:

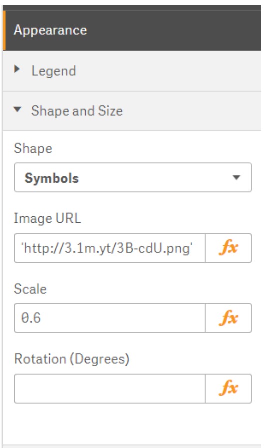
'<h1>'&WaterDesignation&'</h1><hr />'&WaterAddress



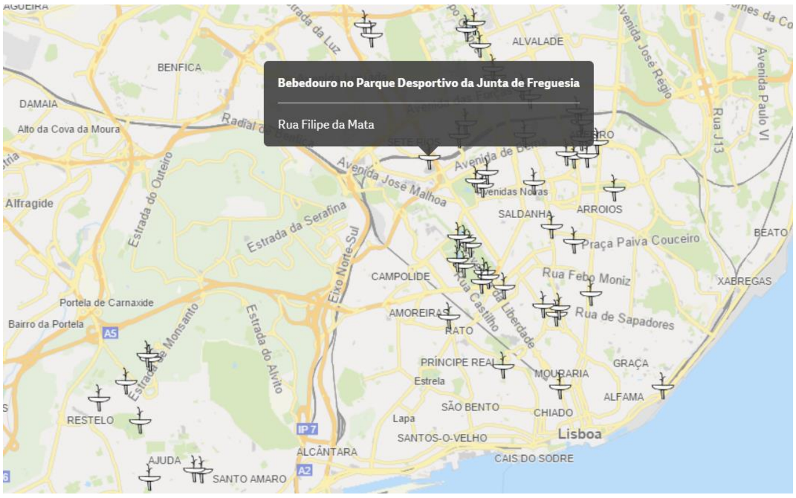
Finishing with the **Symbol** configuration, go to the GeoAnalytics Bubble Layer extension properties, select **Appearance,** expand **Shape and Size**, select **Symbols** in the **Shape** dropdown and add the following **URL**:

'http://3.1m.yt/3B-cdU.png'

Resize it by setting the **Scale** property to 0.6.



Here is an example of how your work may look like:



Exercise 2: Using TravelAreas

Now that we know where we could stop for a little of water, let's give some more information to our map. An interesting way to look at this information could be to find out what would be the potential travel area available for each of these water spots for a maximum distance of five minutes walking and five minutes on bicycle.

To do so we will use another function available on the GeoAnalytics Connector named TravelAreas.

> Note: always check the GeoAnalytics reference page for full details using the TravelAreas Operation: https://bi.idevio.com/wp-content/qlik/geoanalytics/releases/IdevioGeoAnalyticsConnector-5.13.0/doc/geoanalytics_reference-September_2018.html#Operation.TravelAreas

Let's start by going to the **Data load editor,** make sure your cursor is positioned at the end of the script and click on the **Select data** icon ( ) of your GeoAnalytics Connection.

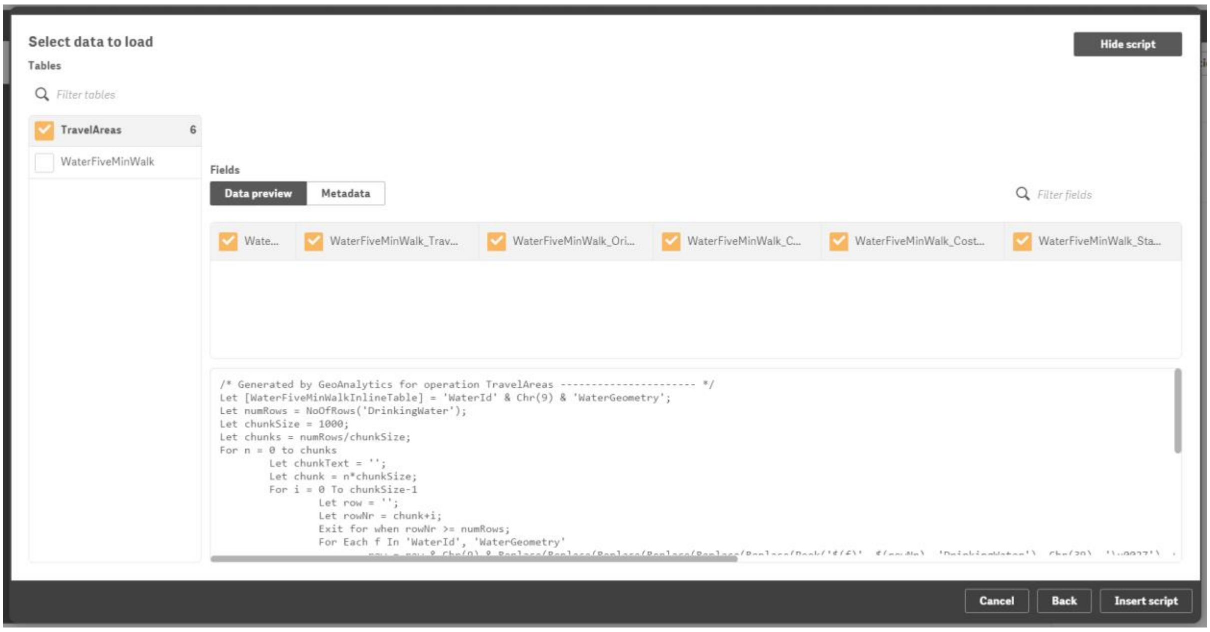
Select the **Operation: TravelAreas** and we will now set it up so it gets the necessary information from our *DrinkingWater* table. Remember the importance of the little question marks on the wizard to help you in these configuration windows.

Regarding the **Operation parameters** input *5* as **Cost Value**, change **Cost Unit** to *Minutes* and select *Pedestrian* as **Transportation**.

In the **Origins** parameters setup the **Name** to *WaterFiveMinWalk* (quick note: this is the name of the temporary dataset sent via the connector and not the final table name), select *Loaded table* in the **Type** dropdown, input *DrinkingWater* as **Table Name** (this will let the connector know what is the name of our resident table) and input *WaterId,WaterGeometry* in **Table Fields (key,geometry,…)**. You may let the other fields all with the default values. A quick note: it is important to define the **Table Fields (key,geometry…)** correctly since this is the parameter that will match what the connector needs to look at as well as how the resulting table will be associated with your data model via the "key" field.

Once all the parameters are introduced click on **Next**. Here is a screenshot to let you double check how the parameters should be introduced:

Once you get the **Select data to load** window, mark the **TravelAreas** table checkbox to load all the fields and click on **Insert script**.

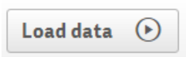


Finally (and optionally), to help us making a clear and readable datamodel, **rename** the [TravelAreas] table name to [WaterFiveMinWalk].

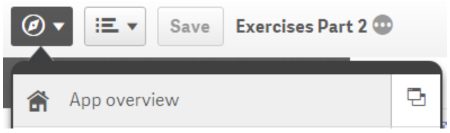
Your generated script should look like the following:

```
LIB CONNECT TO 'GeoAnalytics';

/* Generated by GeoAnalytics for operation TravelAreas ---------------------- */
Let [WaterFiveMinWalkInlineTable] = 'WaterId' & Chr(9) & 'WaterGeometry';
Let numRows = NoOfRows('DrinkingWater');
Let chunkSize = 1000;
Let chunks = numRows/chunkSize;
For n = 0 to chunks
		Let chunkText = '';
		Let chunk = n*chunkSize;
		For i = 0 To chunkSize-1
				Let row = '';
				Let rowNr = chunk+i;
				Exit for when rowNr >= numRows;
				For Each f In 'WaterId', 'WaterGeometry'
						row = row & Chr(9) & Replace(Replace(Replace(Replace(Replace(Replace(Peek('$(f)', $(rowNr),
'DrinkingWater'), Chr(39), '\u0027'), Chr(34), '\u0022'), Chr(91), '\u005b'), Chr(47), '\u002f'), Chr(42), '\u002a'), Chr(59), '\u003b');
				Next
				chunkText = chunkText & Chr(10) & Mid('$(row)', 2);
		Next
		[WaterFiveMinWalkInlineTable] = [WaterFiveMinWalkInlineTable] & chunkText;
Next
chunkText=''


[WaterFiveMinWalk]:
SQL SELECT [WaterId], [WaterFiveMinWalk_TravelArea], [WaterFiveMinWalk_Origin], [WaterFiveMinWalk_Cost],
[WaterFiveMinWalk_CostUnit], [WaterFiveMinWalk_Status] FROM TravelAreas(costValue='5', costField='', costUnit='Minutes',
transportation='pedestrian', dataset='WaterFiveMinWalk')
DATASOURCE WaterFiveMinWalk INLINE tableName='DrinkingWater', tableFields='WaterId,WaterGeometry', geometryType='POINT',
loadDistinct='NO', suffix='', crs='Auto' {$(WaterFiveMinWalkInlineTable)}
;

[WaterFiveMinWalkInlineTable] = '';

/* End GeoAnalytics operation TravelAreas ----------------------------------- */
```

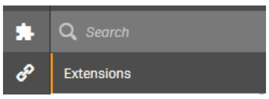
Load your data and test it out returning to your previously created sheet.



\+



Add the **GeoAnalytics Area Layer** to your sheet and drag the **WaterFiveMinWalk_TravelArea** field to it.

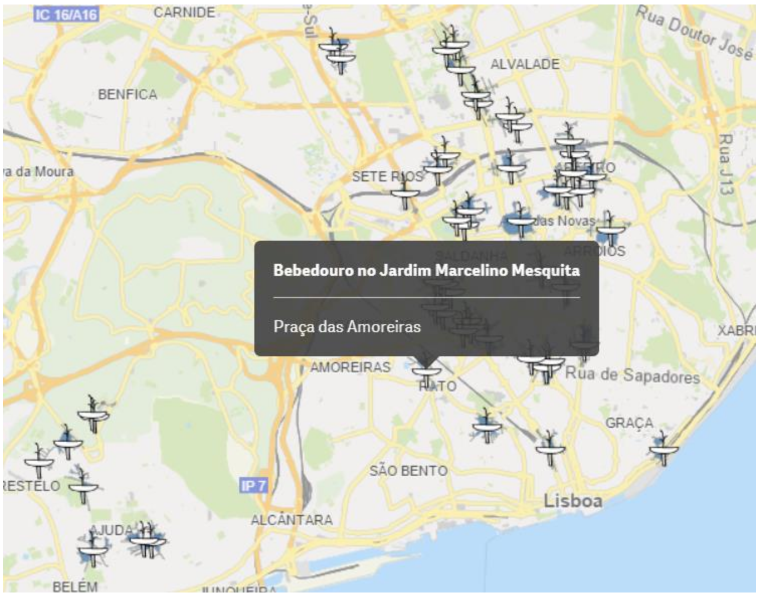


\+

\+

You will now see on the map the corresponding areas of reach for each of the water spots in a 5 minutes walking distance.

Optionally fine tune the look of the areas by changing their color and transparency as you see fit going to the GeoAnalytics Area Layer extension properties, select **Appearance**, open the **Color** pane and change its properties.



Now we will repeat the process of adding a TravelArea table, this time specifying it for bicycle as the transportation.

Go back to the **Data load editor,** make sure your cursor is positioned at the end of the script and click on the **Select data** icon ( ) of your GeoAnalytics Connection.

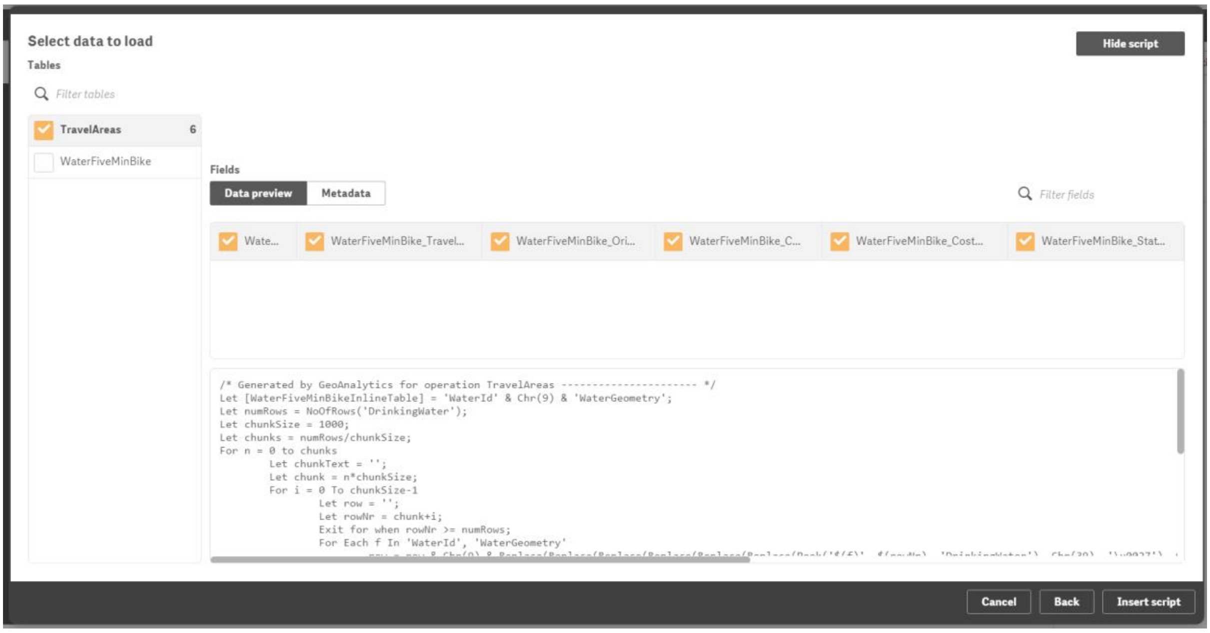
Select the **Operation: TravelAreas** set it to load the necessary information from our *DrinkingWater* table.

In the **Operation parameters** section input *5* as **Cost Value**, change **Cost Unit** to *Minutes* and select *Bike* as **Transportation**.

In the **Origins** parameters section setup the **Name** to *WaterFiveMinBike,* select *Loaded table* in the **Type** dropdown, input *DrinkingWater* as **Table Name** (this will let the connector know what is the name of our resident table) and input *WaterId,WaterGeometry* in **Table Fields (key,geometry,…)**. You may let the other fields all with the default values.

Once all the parameters are introduced click on **Next**. Here is a screenshot to let you double check how the parameters should be introduced:

Once you get the **Select data to load** window, mark the **TravelAreas** table checkbox to load all the fields and click on **Insert script**.



Once again (and optionally), to help us making a clear and readable datamodel, **rename** the [TravelAreas] table name to [WaterFiveMinBike].

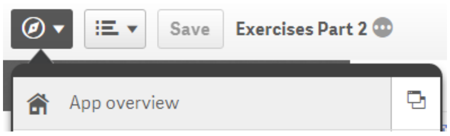
Your generated script should look like the following:

```
LIB CONNECT TO 'GeoAnalytics';

/* Generated by GeoAnalytics for operation TravelAreas ---------------------- */
Let [WaterFiveMinBikeInlineTable] = 'WaterId' & Chr(9) & 'WaterGeometry';
Let numRows = NoOfRows('DrinkingWater');
Let chunkSize = 1000;
Let chunks = numRows/chunkSize;
For n = 0 to chunks
        Let chunkText = '';
        Let chunk = n*chunkSize;
```

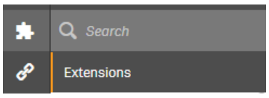
```
		For i = 0 To chunkSize-1
			Let row = '';
			Let rowNr = chunk+i;
			Exit for when rowNr >= numRows;
			For Each f In 'WaterId', 'WaterGeometry'
				row = row & Chr(9) & Replace(Replace(Replace(Replace(Replace(Replace(Peek('$(f)', $(rowNr), 'DrinkingWater'), Chr(39), '\u0027'), Chr(34), '\u0022'), Chr(91), '\u005b'), Chr(47), '\u002f'), Chr(42), '\u002a'), Chr(59), '\u003b');
			Next
			chunkText = chunkText & Chr(10) & Mid('$(row)', 2);
		Next
		[WaterFiveMinBikeInlineTable] = [WaterFiveMinBikeInlineTable] & chunkText;
Next
chunkText=''


[WaterFiveMinBike]:
SQL SELECT [WaterId], [WaterFiveMinBike_TravelArea], [WaterFiveMinBike_Origin], [WaterFiveMinBike_Cost], [WaterFiveMinBike_CostUnit], [WaterFiveMinBike_Status] FROM TravelAreas(costValue='5', costField='', costUnit='Minutes', transportation='bike', dataset='WaterFiveMinBike')
DATASOURCE WaterFiveMinBike INLINE tableName='DrinkingWater', tableFields='WaterId,WaterGeometry', geometryType='POINT', loadDistinct='NO', suffix='', crs='Auto' {$(WaterFiveMinBikeInlineTable)}
;

[WaterFiveMinBikeInlineTable] = '';

/* End GeoAnalytics operation TravelAreas ---------------------------------- */
```

Load your data and test it out returning to your previously created sheet.

+

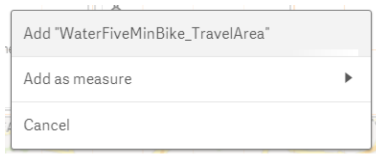


Add the **GeoAnalytics Area Layer** to your sheet and drag the **WaterFiveMinBike_TravelArea** field to it.



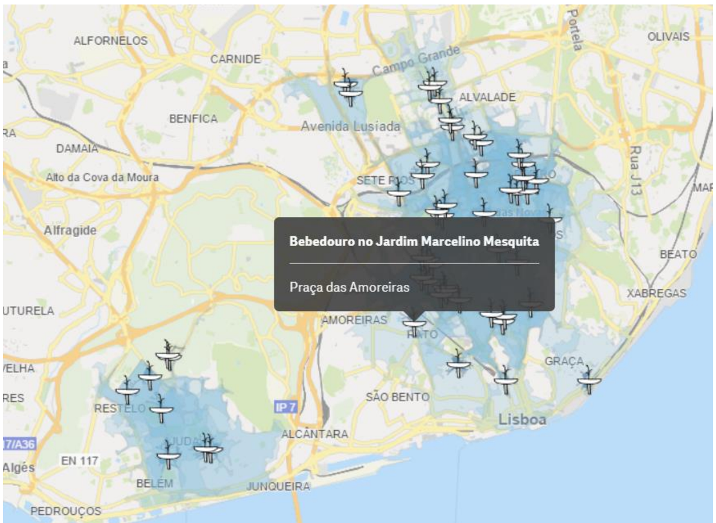
+

+

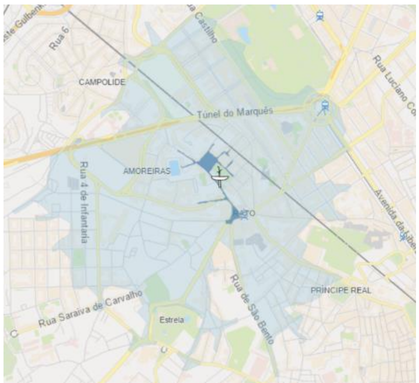


You will now see on the map the corresponding areas of reach for each of the water spots in a 5 minutes cycling distance.

Optionally fine tune the look of the areas by changing their color and transparency as you see fit going to the GeoAnalytics Area Layer extension properties, select **Appearance**, open the **Color** pane and change its properties.



>



As our multi-layer work evolves, we might need to take ownership about the drawing order of the layers. In this case we can ensure that the bicycle travel area is in the bottom, the walking travel area is in the middle and the water spot icon is on top of both the other layers.

This can be achieved by setting up the **Draw Order** in each of the extensions' properties. This is located within the **Layer Options** pane. Deactivate the *Auto* option for **Draw Order** and manually edit the layering level. Increasing this value makes the layer draw later, i.e. be more visible in the category. Allowed values are in the range -10 to 10. If you want the layer to be on top of other layers in the same category increase this value, if you want it to be overdrawn by other layers in the category decrease it.

Recommended settings for this exercise are:

- Water spot Bubble Layer: *Bubble Layer* for **Draw Order Category** and 0 for **Draw Order Adjustment**
- Five Minutes Walk Area Layer: *Area Layer* for **Draw Order Category** and 1 for **Draw Order Adjustment**
- Five Minutes Bicycle Area Layer: *Area Layer* for **Draw Order Category** and 0 for **Draw Order Adjustment**

Exercise 3: Using Binning

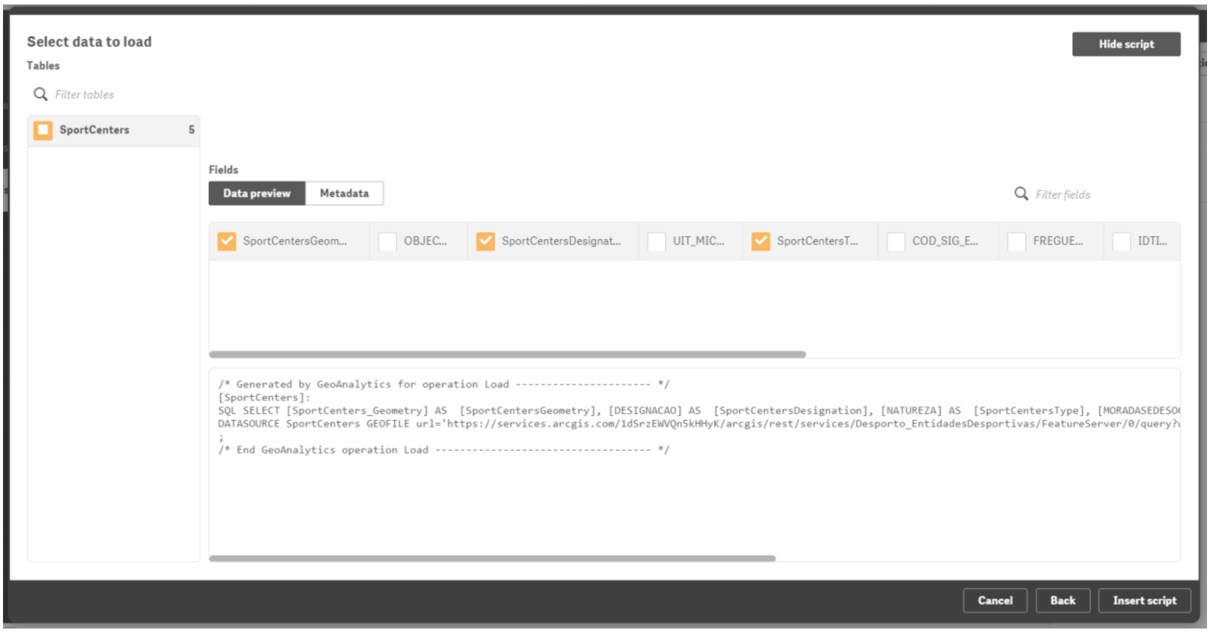
To bring up our active lifestyle up a notch we will go back to our data source Lisboa Aberta and this time load all the public and private sport centers, gyms, etc. available in the city and check if they have water spots nearby in case we go out for a run as well.

Let's start by going to the **Data load editor,** make sure your cursor is positioned at the end of the script and click on the **Select data** icon ( ) of your GeoAnalytics Connection. Select the **Operation: Load**. Give *SportCenters* as **Name** for your dataset, select *File based geo dataset* for **Type** and paste the link bellow in **URL.**

https://services.arcgis.com/1dSrzEWVQn5kHHyK/arcgis/rest/services/Desporto_EntidadesDesportivas/FeatureServer/0/query?where=1%3D1&outFields=*&f=pgeojson

You may leave all the other options as suggested by default and click on **Next**.

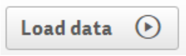
For our application, we don't need all the fields available from the GeoJSON. You may **rename and select** *SportCenters_Geometry* to *SportCentersGeometry*, **rename and select** *DESIGNACAO* to *SportCentersDesignation*, **rename and select** *NATUREZA* to *SportCentersType*, rename and select *MORADASEDESOCIAL* to *SportCentersAddress* and **rename and select** *_autoIndex_* to *SportCentersId*. Once you've selected and renamed the fields you may click on **Insert Script**.



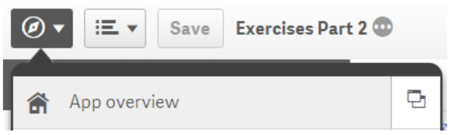
The following script is expected to have been generated for you:

```
LIB CONNECT TO 'GeoAnalytics';

/* Generated by GeoAnalytics for operation Load ---------------------- */
[SportCenters]:
SQL SELECT [SportCenters_Geometry] AS [SportCentersGeometry], [DESIGNACAO] AS [SportCentersDesignation], [NATUREZA] AS [SportCentersType], [MORADASEDESOCIAL] AS [SportCentersAddress], [_autoIndex_] AS [SportCentersId] FROM Load(dataset='SportCenters')
DATASOURCE SportCenters GEOFILE url='https://services.arcgis.com/1dSrzEWVQn5kHHyK/arcgis/rest/services/Desporto_EntidadesDesportivas/FeatureServer/0/query?where=1%3D1&outFields=*&f=pgeojson', keyField='', type='auto', expectedGeomType='auto', encoding='', crs='Auto'
;
/* End GeoAnalytics operation Load ----------------------------------- */
```

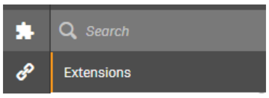
Load your data and test it out returning to your previously created sheet.



+

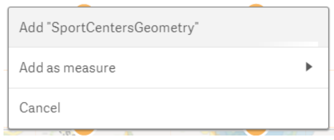


Add the **GeoAnalytics Bubble Layer** to your sheet and drag the **SportCentersGeometry** field to it.

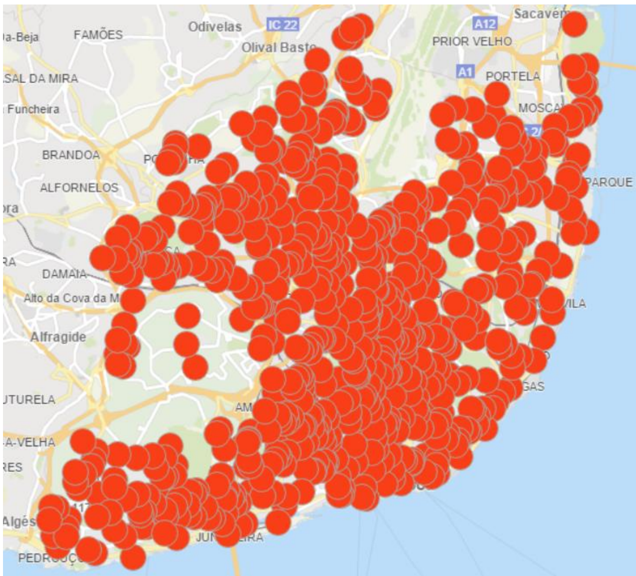


+

+



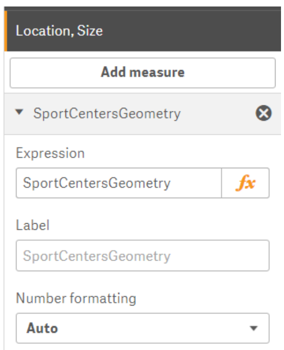
You are now able to see all the sport centers available in Lisbon, Portugal



Before going into simplifying the representation of this information by creating bins where the corresponding points can be contained, we will only add a quick color coding to the bubbles so we can quickly identify which of these sport centers are public or private.

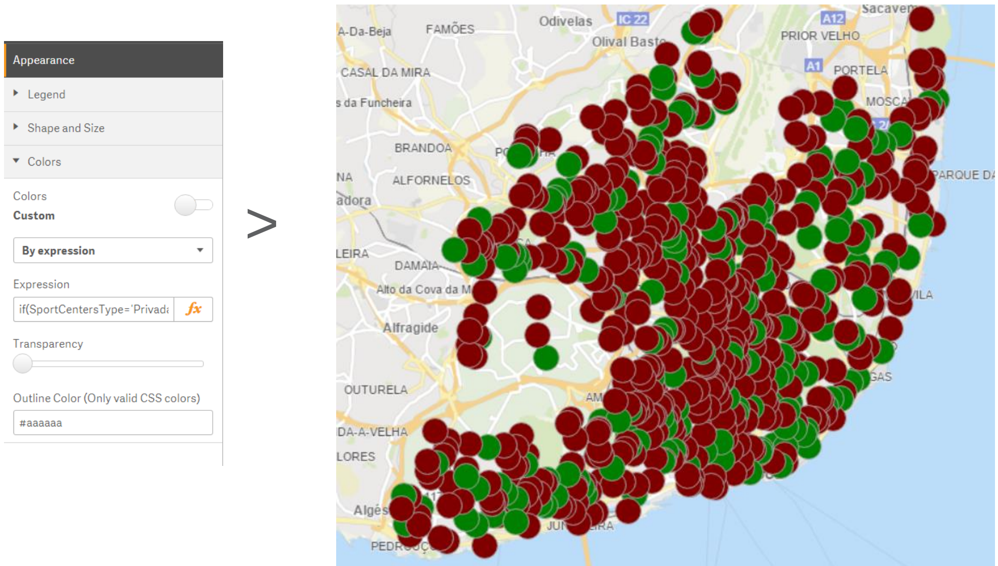
Go to the GeoAnalytics Bubble Layer extension properties, select **Appearance** and expand the **Colors** pane. Deactivate the *Auto* option and search in the dropdown for the **By expression** option. You'll notice it is not available. To make it available you must have the location specified via **Location, Size** pane instead of the default **ID** Pane.

Go to the **Location, Size** pane and **add** the **measure** *SportCentersGeometry* without any kind of aggregation.



Go back to the **Appearance** pane and check if the **By Expression** option is now available within the **Colors** pane. Click on **fx** to add an expression with a simple if just like this one:

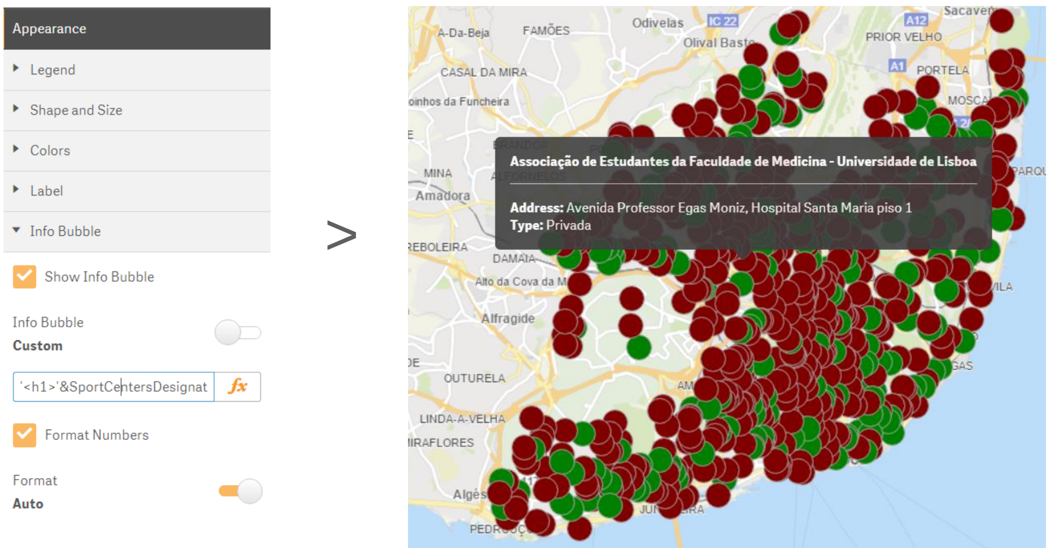
if(SportCentersType='Privada', red(),green())



Optionally, you can also add some contextual information in the **Info Bubble** such as the Sport Center name, type and address.

Go to the **Info Bubble** pane and deactivate the *Auto* option to access the **fx** expression window and add the information you would like to display, such as:

'<h1>'&SportCentersDesignation&'</h1><hr>Address: '&SportCentersAddress&'
Type: '&SportCentersType



Now we will clean up the map a little bit using the **Binning** function. This function will allow us to create bins/blocks that cover the corresponding points with their area. This way we can display a few blocks and once a user selects the block we can drill into detail and show the actual points.

Note: always check the GeoAnalytics reference page for full details using the Binning Operation: https://bi.idevio.com/wp-content/qlik/geoanalytics/releases/IdevioGeoAnalyticsConnector-5.13.0/doc/geoanalytics_reference-September_2018.html#Operation.Binning

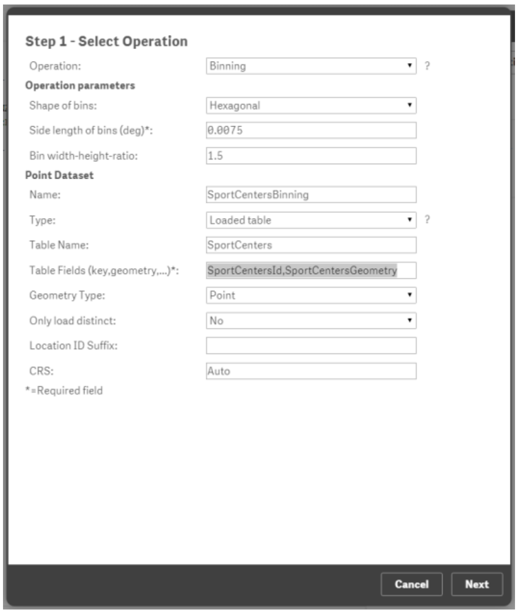
Let's start by going back to the **Data load editor,** make sure your cursor is positioned at the end of the script and click on the **Select data** icon ( ) of your GeoAnalytics Connection.

Select the **Operation: Binning** and we will now set it up so it gets the necessary information from our *SportCenters* table. Remember the importance of the little question marks on the wizard to help you in these configuration windows.

Regarding the **Operation parameters** select *Hexagonal* for the Shape of bins, input *0.0075* for the **Side length of bins (deg)** – 1 corresponds to approximately 10 000m, don't forget to refer the help – and set *1.5* to **Bin width-height-ratio**.

In the **Point Dataset** parameters input *SportCentersBinning* as **Name**, select *Loaded Table* for **Type**, input *SportCenters* for **Table Name** and input *SportCentersId,SportCentersGeometry* for **Table Fields (key,geometry,…)**.

Once all the parameters are introduced click on **Next**. Here is a screenshot to let you double check how the parameters should be introduced:



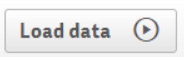
Once you get the **Select data to load** window, notice that there won't be any selectable table available. Don't worry as this is an expected behavior since it is something that is actually created and calculated via script and not via the receiving of an external table from a source. Click on **Insert Script** to continue.

Your generated script should look like the following:

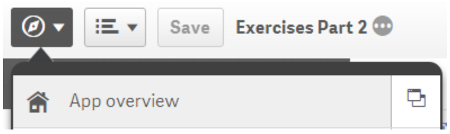
```
LIB CONNECT TO 'GeoAnalytics';

/* Generated by GeoAnalytics for operation Binning ---------------------- */
[HexTemp1]:
Load
  SportCentersId,
  sqrt(3)/3 * subfield(TextBetween(SportCentersGeometry, '[', ']'), ',' ,1)/0.006495 - 1/3 * subfield(TextBetween(SportCentersGeometry, '[',
']'), ',' ,2)/0.004330 as qq,
  2/3 * subfield(TextBetween(SportCentersGeometry, '[', ']'), ',' ,2)/0.004330 as rr
Resident 'SportCenters'
;

[HexRoundTable1]:
Load
  SportCentersId,
  qq,
  rr,
  -qq - rr as ss
Resident [HexTemp1]
;
drop table [HexTemp1];

[HexRoundTable2]:
Load
  SportCentersId,
  qq,
  rr,
  ss,
  fabs(Round(qq) - qq) as qDiff,
  fabs(Round(rr) - rr) as rDiff,
  fabs(Round(ss) - ss) as sDiff
Resident [HexRoundTable1]
;
drop table [HexRoundTable1];

[HexRoundTable3]:
Load
  SportCentersId,
  If(qDiff > rDiff and qDiff > sDiff, -Round(rr) - Round(ss), Round(qq)) as rq,
  If(rDiff > sDiff and (qDiff <= rDiff or qDiff <= sDiff), -Round(qq) - Round(ss), Round(rr)) as rr,
  If((qDiff <= rDiff or qDiff <= sDiff) and (rDiff <= sDiff), -Round(qq) - Round(rr), Round(ss)) as rs
Resident [HexRoundTable2]
;
```

```
drop table [HexRoundTable2];

[HexToCoordTable]:
Load
 SportCentersId,
 (sqrt(3) * rq + sqrt(3)/2 * rr) * 0.006495 as lon,
 (3/2 * rr) * 0.004330 as lat
Resident [HexRoundTable3]
;
drop table [HexRoundTable3];

[HH0_0075W0_01125GridTable]:
Load
 SportCentersId,
 '[[[['
   & Num(lon + 0.005625,'########','.') & ',' & Num(lat + 0.002165,'########','.') & '],['
   & Num(lon + 0.000000,'########','.') & ',' & Num(lat + 0.004330,'########','.') & '],['
   & Num(lon + -0.005625,'########','.') & ',' & Num(lat + 0.002165,'########','.') & '],['
   & Num(lon + -0.005625,'########','.') & ',' & Num(lat + -0.002165,'########','.') & '],['
   & Num(lon + -0.000000,'########','.') & ',' & Num(lat + -0.004330,'########','.') & '],['
   & Num(lon + 0.005625,'########','.') & ',' & Num(lat + -0.002165,'########','.') & '],['
   & Num(lon + 0.005625,'########','.') & ',' & Num(lat + 0.002165,'########','.')
   & ']]]]' as HH0_0075W0_01125,
   '[' & Num(lon,'########','.') & ',' & Num(lat,'########','.') & ']' as CenterPoint
Resident [HexToCoordTable]
;
drop table [HexToCoordTable];

/* End GeoAnalytics operation Binning ---------------------------------- */
```

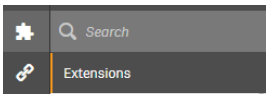
Load your data and test it out returning to your previously created sheet.



+

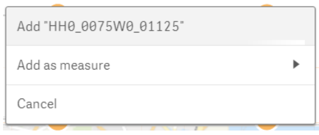


Add the **GeoAnalytics Area Layer** to your sheet and drag the **HH0_0075W0_01125** field to it.

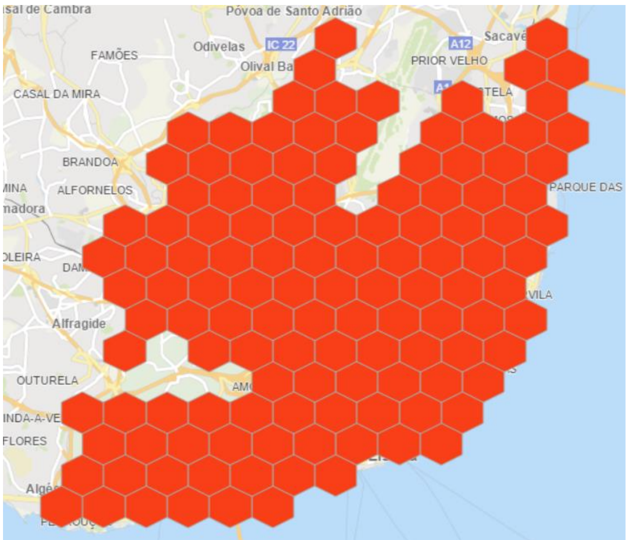


+

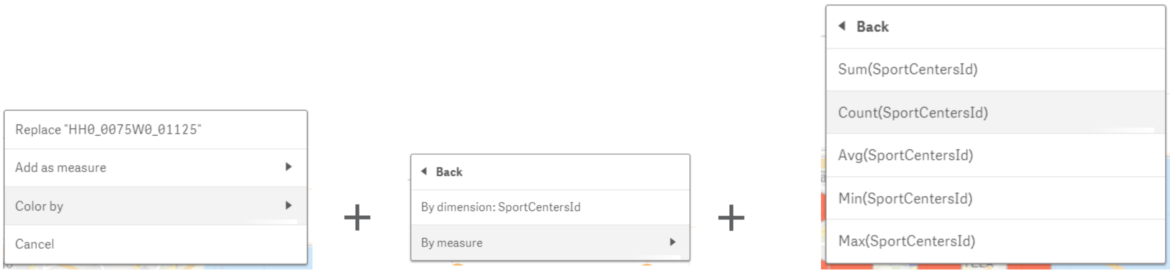
+



As a suggestion, make all the other layers invisible while we work in the look of our bins.

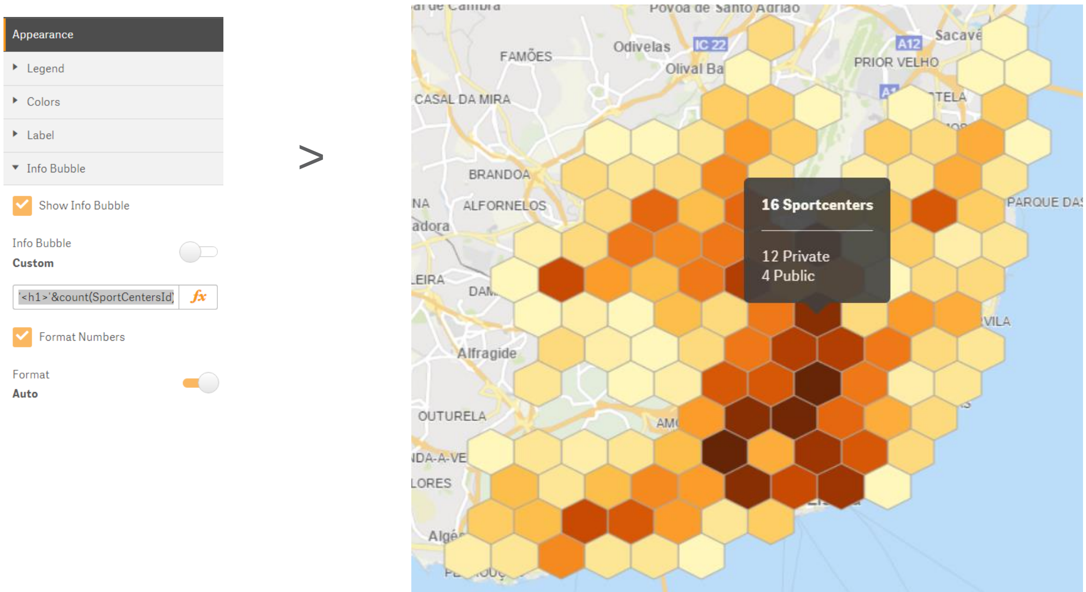


To bring some additional information we can color the bins based on the number of Sport Centers they have. Drag the field *SportCentersID* over your **GeoAnalytics Area Layer** extension, select **Color by,** select **By measure** and choose **Count(SportCentersId)**.



Represent also the overall resume of each bin by adding a custom **Info Bubble** with an expression such as

'<h1>'&count(SportCentersId) &' Sportcenters</h1><hr />'&count({<SportCentersType={"Privada"}>}SportCentersId)&' Private
'&count({<SportCentersType={"Pública"}>}SportCentersId)&' Public
'



Finally, we will add a drill down interaction between our Area Layer with the bins and our Bubble layer that contains the sport centers locations. To achieve this we will generate an activation/deactivation of the layer based on the map zoom. Make sure both layers are visible before starting the configuration.

Go to the **GeoAnalytics Area Layer** extensions properties for the Binning areas and select the **Layer Options** pane. Set the **Zoom Limits In** to range starting in 7 and **Out** ending in 160 000.

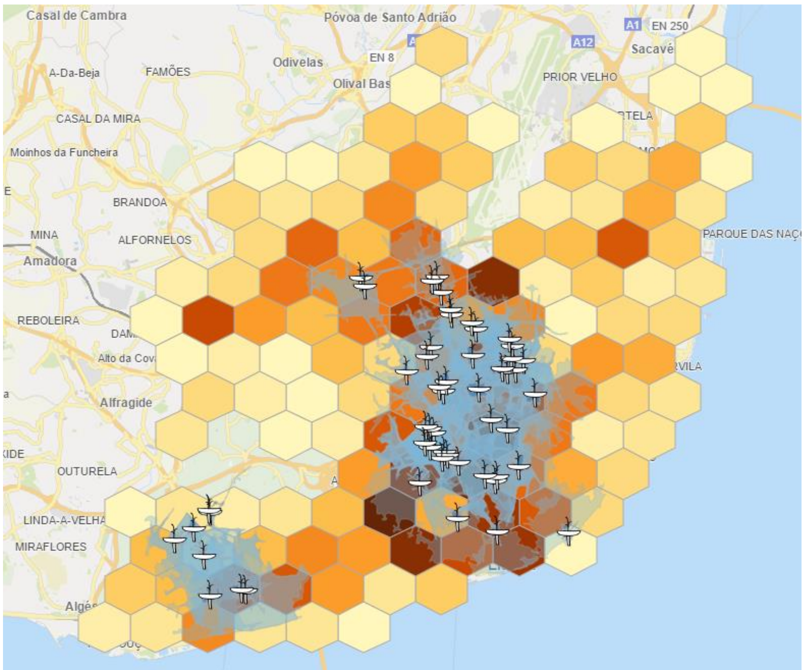
Next, go to the **GeoAnalytics Bubble Layer** extensions properties for the sport centers locations and select the **Layer Options** pane. Set the **Zoom Limits In** to range starting in 0 and **Out** ending in 7.

Now if you zoom in the map or select a bin you should be able to see only the correct information based on your zoom level.

Exercise 4: Using Within

We have now information from water spots as well as sport centers. If we try and make visible all the layers, we can have 2 issues.

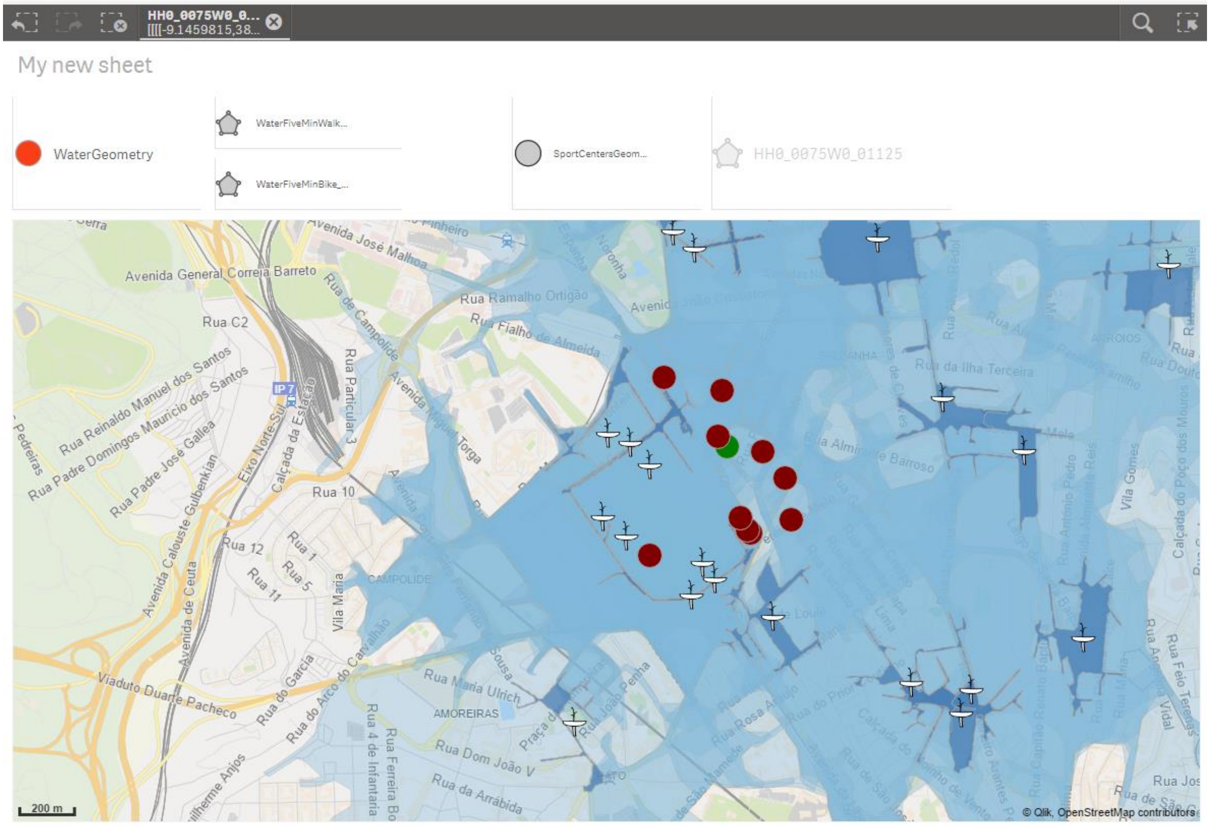
The first issue, and simpler one to solve, is the layering order so we can make sure representations are correct. In my example below the travel areas from exercise 2 are over the binning areas.



Here is a suggestion of how each extension **Layer Options, Draw Order** could be setup:

Extension	Draw Order Category	Draw Order Adjustment
GeoAnalytics Area Layer: Binning areas	Area Layer	5
GeoAnalytics Bubble Layer: Sport Centers	Area Layer	4
GeoAnalytics Bubble Layer: Water Spots	Area Layer	3
GeoAnalytics Area Layer: Five Minues Walking	Area Layer	2
GeoAnalytics Area Layer Five Minutes Cycling	Area Layer	1

The second issue is the one you get when you select a binning area and this has no association with the water spots.



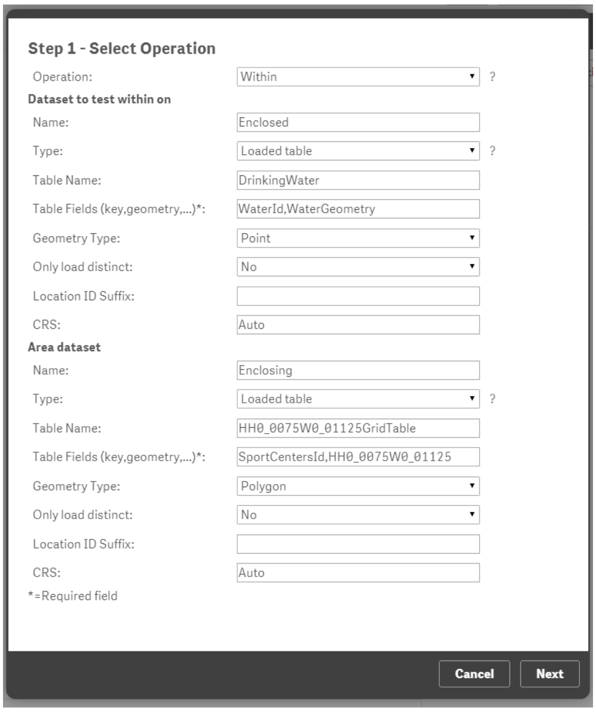
To solve this issue you can use another of the functions available in the Qlik GeoAnalytics Connector: **Within**. The **Within** function allows you to indicate two datasets and find out which points are within/correlated between them.

> Note: always check the GeoAnalytics reference page for full details using the Within Operation: https://bi.idevio.com/wp-content/qlik/geoanalytics/releases/IdevioGeoAnalyticsConnector-5.13.0/doc/geoanalytics_reference-September_2018.html#Operation.Within

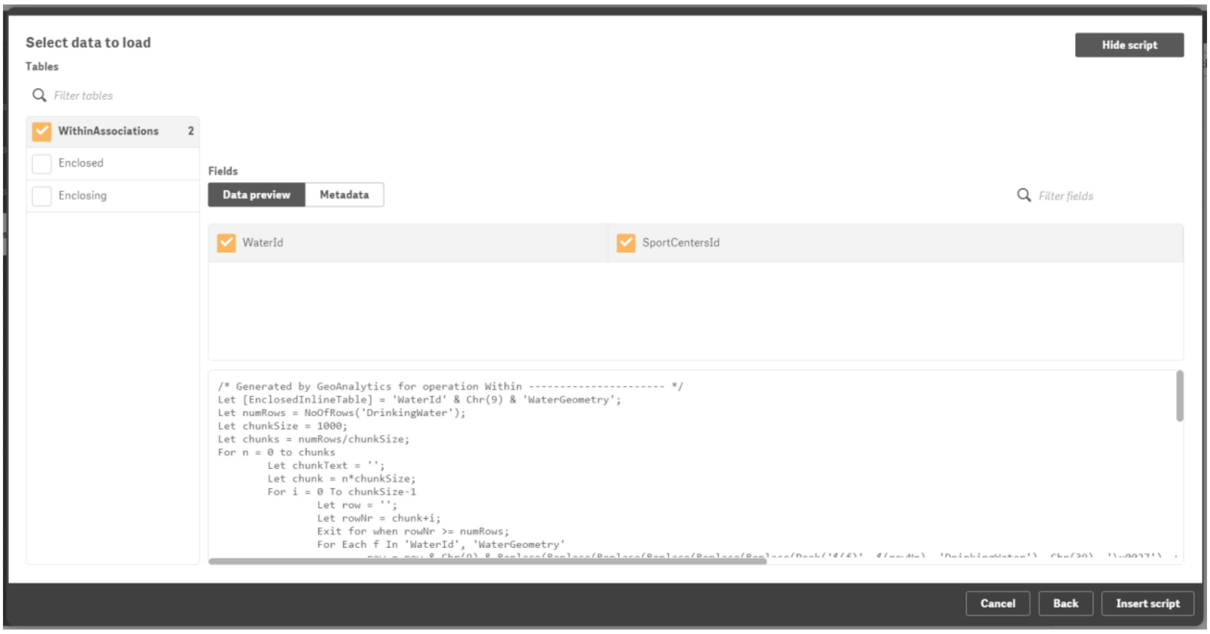
In our case, we have some areas, the binning areas, that will certainly have water spots within them. Let's start by going to the **Data load editor,** make sure your cursor is positioned at the end of the script and click on the **Select data** icon ( ) of your GeoAnalytics Connection and select the **Operation: Within.**

Let's start by indicating which is the **Dataset to test within on**: the water spots. Select *Loaded Table* for **Type**, insert *DrinkingWater* as **Table Name** and insert *WaterId,WaterGeo*metry as **Table Fields (key,geometry,…).**

Now we will indicate which is the **Area dataset:** the sport centers. Select *Loaded Table* for **Type**, insert *HH0_0075W0_01125GridTable* as **Table Name**, insert *SportCentersId,HH0_0075W0_01125* as **Table Fields (key,geometry,…),** select *Polygon* as **Geometry Type** and click on **Next**.



The resulting **Select data to load** window will give you 3 resulting tables. Of our interest is only the table that has the relations between both our WaterId and SportCentersId fields. Check the **WithinAssociations** table and click on **Insert script**.

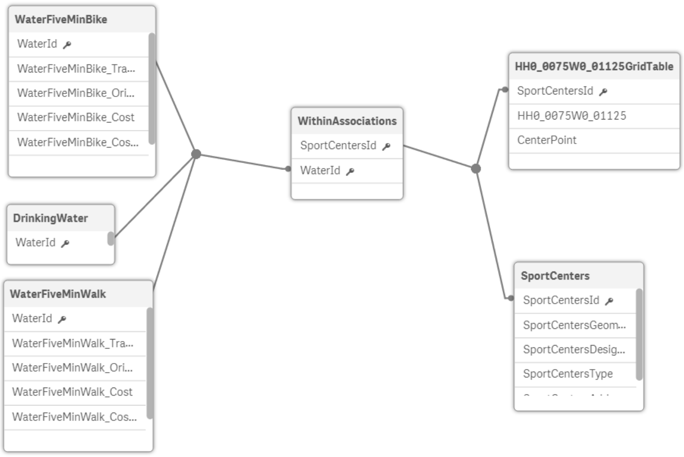


The following script is expected to have been generated for you:

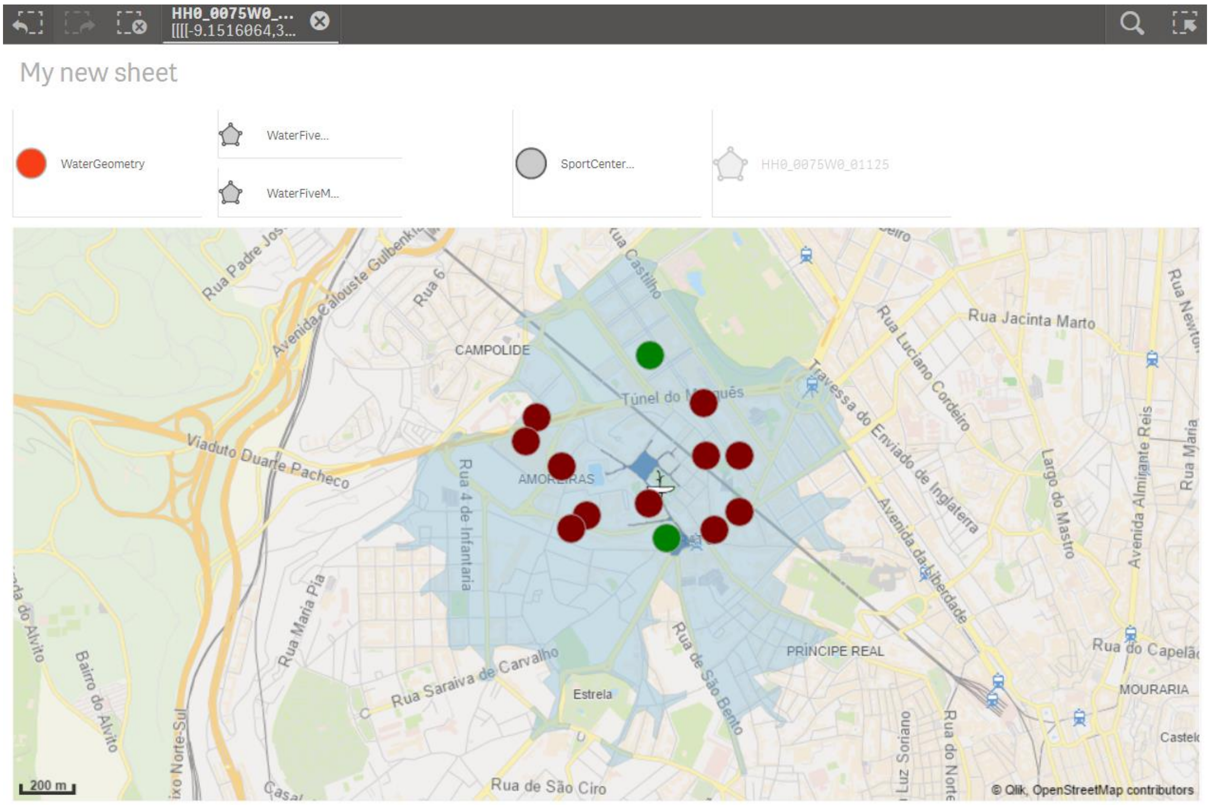
```
LIB CONNECT TO 'GeoAnalytics';

/* Generated by GeoAnalytics for operation Within ---------------------- */
Let [EnclosedInlineTable] = 'WaterId' & Chr(9) & 'WaterGeometry';
Let numRows = NoOfRows('DrinkingWater');
Let chunkSize = 1000;
Let chunks = numRows/chunkSize;
For n = 0 to chunks
        Let chunkText = '';
        Let chunk = n*chunkSize;
```

```
	For i = 0 To chunkSize-1
		Let row = '';
		Let rowNr = chunk+i;
		Exit for when rowNr >= numRows;
		For Each f In 'WaterId', 'WaterGeometry'
			row = row & Chr(9) & Replace(Replace(Replace(Replace(Replace(Replace(Peek('$(f)', $(rowNr), 'DrinkingWater'), Chr(39), '\u0027'), Chr(34), '\u0022'), Chr(91), '\u005b'), Chr(47), '\u002f'), Chr(42), '\u002a'),  Chr(59), '\u003b');
		Next
		chunkText = chunkText & Chr(10) & Mid('$(row)', 2);
	Next
	[EnclosedInlineTable] = [EnclosedInlineTable] & chunkText;
Next
chunkText=''
Let [EnclosingInlineTable] = 'SportCentersId' & Chr(9) & 'HH0_0075W0_01125';
Let numRows = NoOfRows('HH0_0075W0_01125GridTable');
Let chunkSize = 1000;
Let chunks = numRows/chunkSize;
For n = 0 to chunks
	Let chunkText = '';
	Let chunk = n*chunkSize;
	For i = 0 To chunkSize-1
		Let row = '';
		Let rowNr = chunk+i;
		Exit for when rowNr >= numRows;
		For Each f In 'SportCentersId', 'HH0_0075W0_01125'
			row = row & Chr(9) & Replace(Replace(Replace(Replace(Replace(Replace(Peek('$(f)', $(rowNr), 'HH0_0075W0_01125GridTable'), Chr(39), '\u0027'), Chr(34), '\u0022'), Chr(91), '\u005b'), Chr(47), '\u002f'), Chr(42), '\u002a'), Chr(59), '\u003b');
		Next
		chunkText = chunkText & Chr(10) & Mid('$(row)', 2);
	Next
	[EnclosingInlineTable] = [EnclosingInlineTable] & chunkText;
Next
chunkText=''


[WithinAssociations]:
SQL SELECT [WaterId], [SportCentersId] FROM Within(enclosed='Enclosed', enclosing='Enclosing')
DATASOURCE Enclosed INLINE tableName='DrinkingWater', tableFields='WaterId,WaterGeometry', geometryType='POINT', loadDistinct='NO', suffix='', crs='Auto' {$(EnclosedInlineTable)}
DATASOURCE Enclosing INLINE tableName='HH0_0075W0_01125GridTable', tableFields='SportCentersId,HH0_0075W0_01125', geometryType='POLYGON', loadDistinct='NO', suffix='', crs='Auto' {$(EnclosingInlineTable)}
;

[EnclosedInlineTable] = '';
[EnclosingInlineTable] = '';

/* End GeoAnalytics operation Within ----------------------------------- */
```

You may now **load** your data and check for the results. As you can see, all the tables of our data model are associated.



And if you go your application sheet and select a binning area you will be able to see only the corresponding water spots of your selection.



Exercises Part 3 – Additional GeoAnalytics Connector exercises

Exercise 1: Cluster

Let's start by loading some Pharmacies location information and cluster them by proximity. In our use case, we will cluster them by (a max of) 2000m.

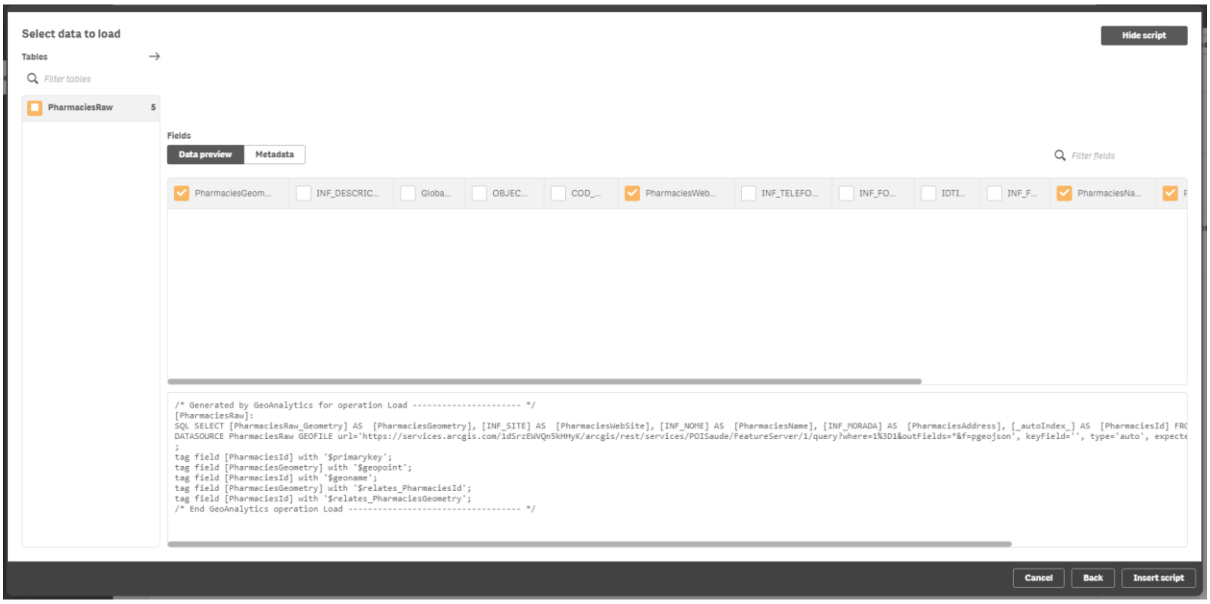
Similar to the previous exercises, load the information from:

https://services.arcgis.com/1dSrzEWVQn5kHHyK/arcgis/rest/services/POISaude/FeatureServer/1/query?where=1%3D1&outFields=*&f=pgeojson

Note: always check the GeoAnalytics reference page for full details using the Load Operation: https://bi.idevio.com/wp-content/qlik/geoanalytics/releases/IdevioGeoAnalyticsConnector-5.13.0/doc/geoanalytics_reference-September_2018.html#Operation.Load

On your Qlik Sense **Data Load Editor** click on the **Select data** icon ( ) of your GeoAnalytics Connection and select the **Operation: Load**. Give *PharmaciesRaw* as **Name** for your *Dataset*, select *File based geo dataset* for **Type** and paste the link in **URL.** You may leave all the other options as suggested by default and click on **Next.**

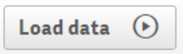
We will not need all the fields available. You may **select and rename** *PharmaciesRaw_Geometry* to *PharmaciesGeometry,* **select and rename** *INF_SITE* to *PharmaciesWebSite,* **select and rename** *INF_NOME* to *PharmaciesName,* **select and rename** *INF_MORADA* to *PharmaciesAddress* and **select and rename** *_autoIndex_ to PharmaciesId.* Once you've selected and renamed the fields you may click on **Insert Script.**



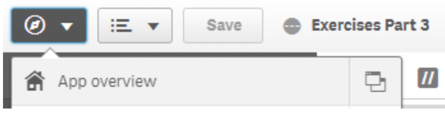
The following script is expected to have been generated for you:

```
LIB CONNECT TO 'GeoAnalytics';

/* Generated by GeoAnalytics for operation Load --------------------- */
[PharmaciesRaw]:
SQL SELECT [PharmaciesRaw_Geometry] AS [PharmaciesGeometry], [INF_SITE] AS [PharmaciesWebSite], [INF_NOME] AS [PharmaciesName], [INF_MORADA] AS [PharmaciesAddress], [_autoIndex_] AS [PharmaciesId] FROM Load(dataset='PharmaciesRaw')
DATASOURCE PharmaciesRaw GEOFILE url='https://services.arcgis.com/1dSrzEWVQn5kHHyK/arcgis/rest/services/POISaude/FeatureServer/1/query?where=1%3D1&outFields=*&f=pgeojson', keyField='', type='auto', expectedGeomType='auto', encoding='', crs='Auto'
;
tag field [PharmaciesId] with '$primarykey';
tag field [PharmaciesGeometry] with '$geopoint';
tag field [PharmaciesId] with '$geoname';
tag field [PharmaciesGeometry] with '$relates_PharmaciesId';
tag field [PharmaciesId] with '$relates_PharmaciesGeometry';
/* End GeoAnalytics operation Load ----------------------------------- */
```

Load your data and test it out by creating a **new sheet**.

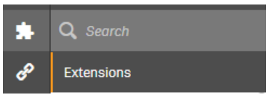


+



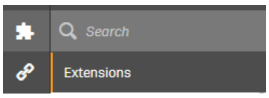
+

Add the **GeoAnalytics Map extension** to your sheet.



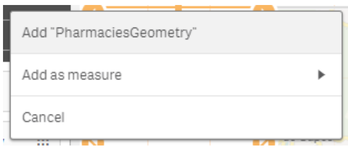
+

Add the **GeoAnalytics Bubble Layer** to your sheet and drag the **PharmaciesGeometry** field to it.

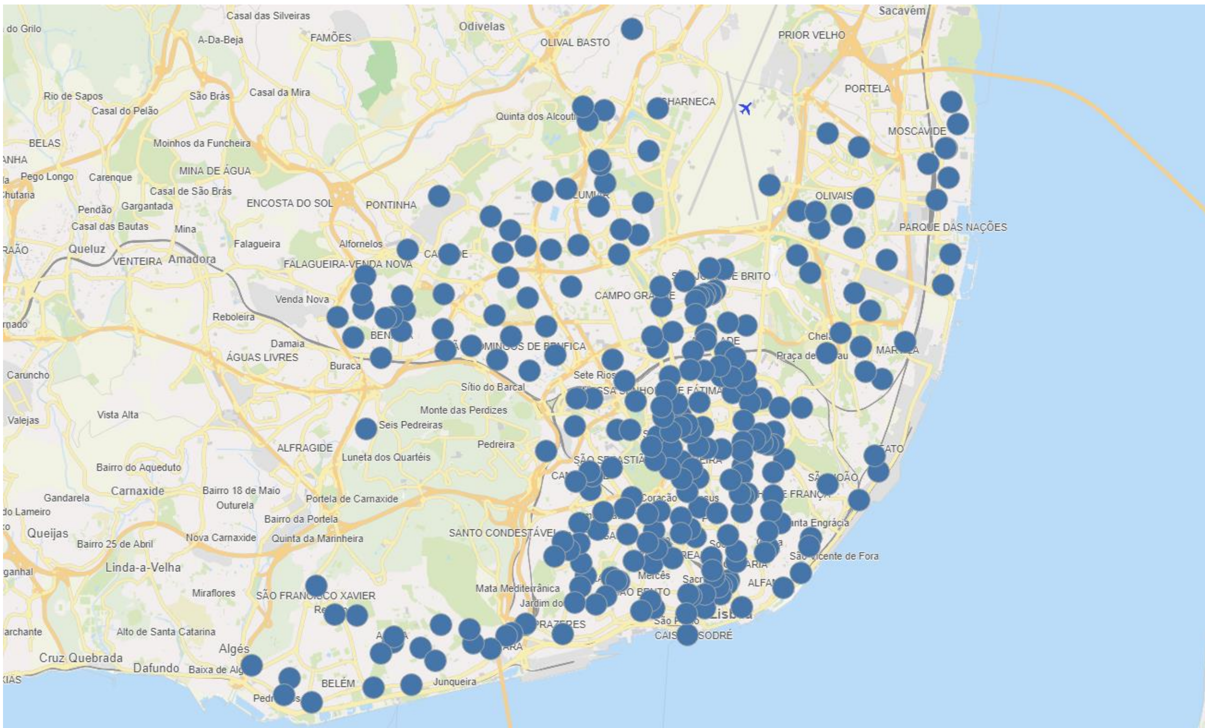


+

+



You are now able to see all the pharmacies available in Lisbon, Portugal.



We can see that in fact there are a lot of Pharmacies available within the city. To simplify the reading, one way to aggregate all these points can be using clusters. As mentioned previously, we will do the clustering aggregation based on a maximum distance 2000m from the center point of the cluster.

Return to the **Data load editor,** make sure your cursor is positioned at the end of the script and click on the **Select data** icon ( ) of your GeoAnalytics Connection.

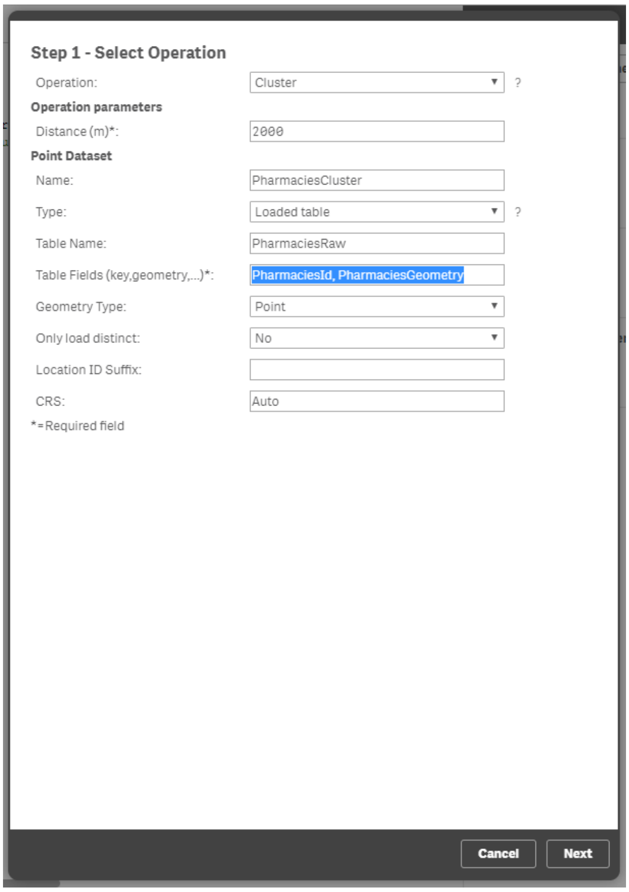
Select the **Operation: Cluster** and we will now set it up so it gets the necessary information from our *PharmaciesRaw* table. Remember the importance of the little question marks on the wizard to help you in these configuration windows.

> Note: always check the GeoAnalytics reference page for full details using the Cluster Operation: https://bi.idevio.com/wp-content/qlik/geoanalytics/releases/IdevioGeoAnalyticsConnector-5.13.0/doc/geoanalytics_reference-September_2018.html#Operation.Cluster

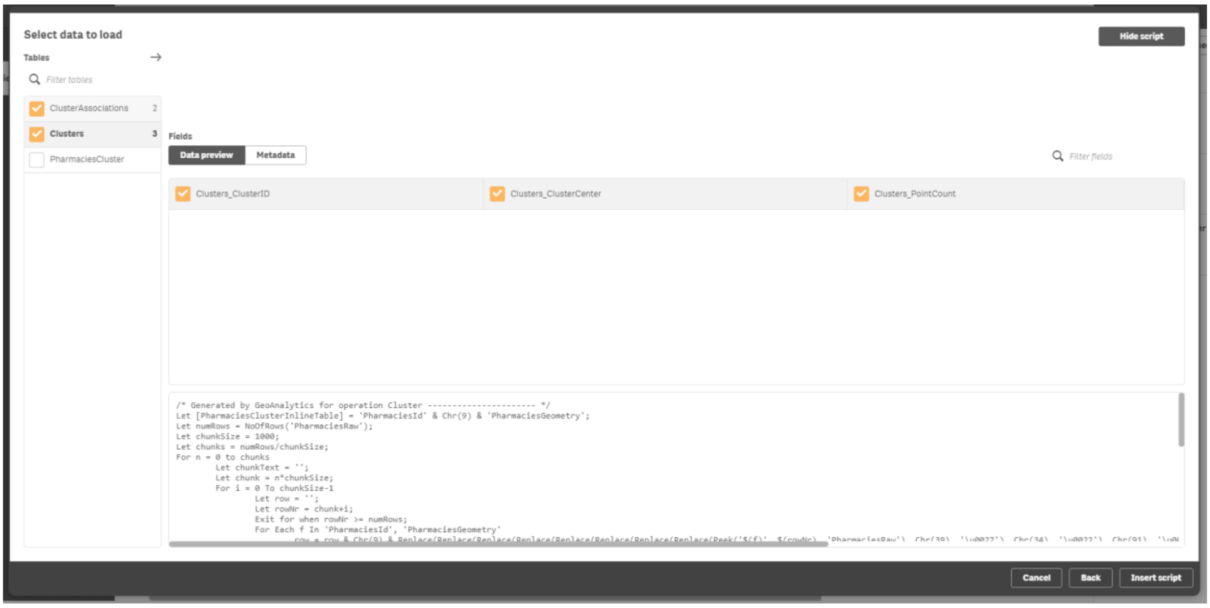
Regarding the **Operation parameters** input 2000 as **Distance (m)**.

In the **Point Dataset** parameters input *PharmaciesCluster* as **Name**, select *Loaded Table* for **Type**, input *PharmaciesRaw* for **Table Name** and input *PharmaciesId, PharmaciesGeometry* for **Table Fields (key,geometry,…)**.

Once all the parameters are introduced click on **Next**. Here is a screenshot to let you double check how the parameters should be introduced:



Once you get the **Select data to load** window, mark the **ClusterAssociations** and **Clusters** tables checkboxes to load all the fields of both tables and click on **Insert script**.

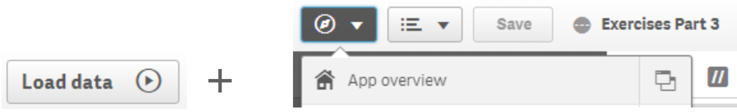


The following script is expected to have been generated for you:

```
LIB CONNECT TO 'GeoAnalytics';

/* Generated by GeoAnalytics for operation Cluster ---------------------- */
Let [PharmaciesClusterInlineTable] = 'PharmaciesId' & Chr(9) & 'PharmaciesGeometry';
Let numRows = NoOfRows('PharmaciesRaw');
Let chunkSize = 1000;
```

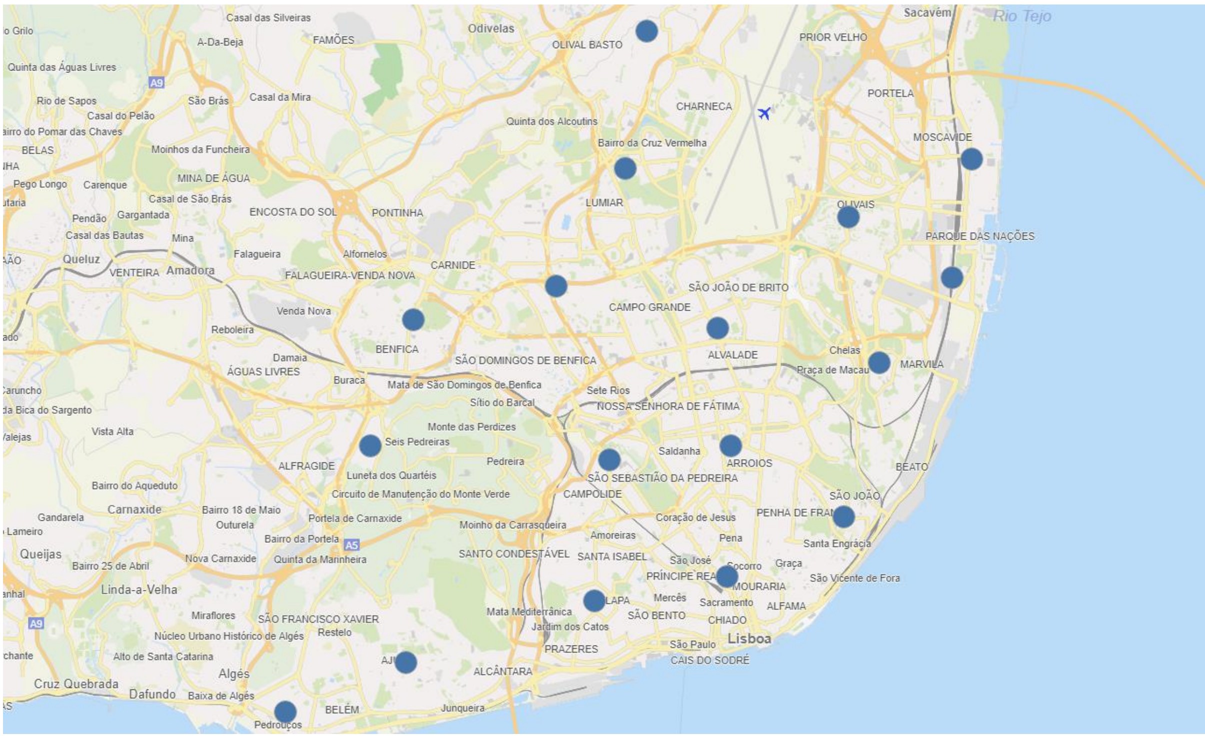
```
Let chunks = numRows/chunkSize;
For n = 0 to chunks
	Let chunkText = '';
	Let chunk = n*chunkSize;
	For i = 0 To chunkSize-1
		Let row = '';
		Let rowNr = chunk+i;
		Exit for when rowNr >= numRows;
		For Each f In 'PharmaciesId', 'PharmaciesGeometry'
			row = row & Chr(9) & Replace(Replace(Replace(Replace(Replace(Replace(Replace(Replace(Peek('$(f)', $(rowNr), 'PharmaciesRaw'), Chr(39), '\u0027'), Chr(34), '\u0022'), Chr(91), '\u005b'), Chr(47), '\u002f'), Chr(42), '\u002a'), Chr(59), '\u003b'), Chr(125), '\u007d'), Chr(123), '\u007b');
		Next
		chunkText = chunkText & Chr(10) & Mid('$(row)', 2);
	Next
	[PharmaciesClusterInlineTable] = [PharmaciesClusterInlineTable] & chunkText;
Next
chunkText=''


[ClusterAssociations]:
SQL SELECT [PharmaciesId], [Clusters_ClusterID] FROM Cluster(distance='2000', points='PharmaciesCluster')
DATASOURCE PharmaciesCluster INLINE tableName='PharmaciesRaw', tableFields='PharmaciesId,PharmaciesGeometry', geometryType='POINT', loadDistinct='NO', suffix='', crs='Auto' {$(PharmaciesClusterInlineTable)}
SELECT [Clusters_ClusterID], [Clusters_ClusterCenter], [Clusters_PointCount] FROM Clusters;
[Clusters]:
SQL LOAD * FROM Clusters;
tag field [PharmaciesId] with '$primarykey';
tag field [Clusters_ClusterID] with '$primarykey';
tag field [Clusters_ClusterCenter] with '$geopoint';
tag field [Clusters_ClusterID] with '$geoname';
tag field [Clusters_ClusterCenter] with '$relates_Clusters_ClusterID';
tag field [Clusters_ClusterID] with '$relates_Clusters_ClusterCenter';

[PharmaciesClusterInlineTable] = '';

/* End GeoAnalytics operation Cluster ----------------------------------- */
```

Load your data and return to your previously created sheet to test your results.



Add a new **GeoAnalytics Bubble Layer** to your sheet and drag the **ClustersCenter** field to it.

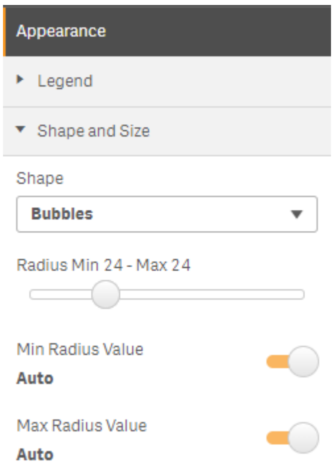
Here is an example of the expected result. For clarity's sake at this point the Pharmacies Bubble Layer is deactivated.



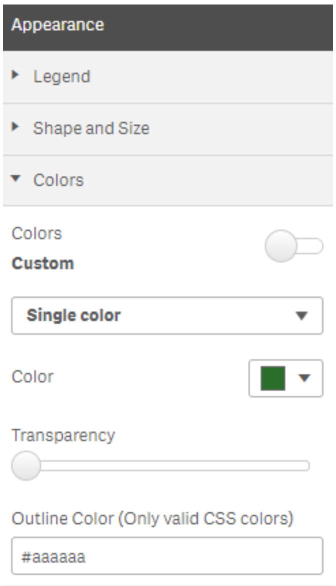
To bring some clarity into our map with our new developments, let's give some layout configurations to both the Bubble Layers representing the Clusters as well as the Pharmacies.

As a simple starting point, we will give different coloring and sizing to both Bubble Layers.

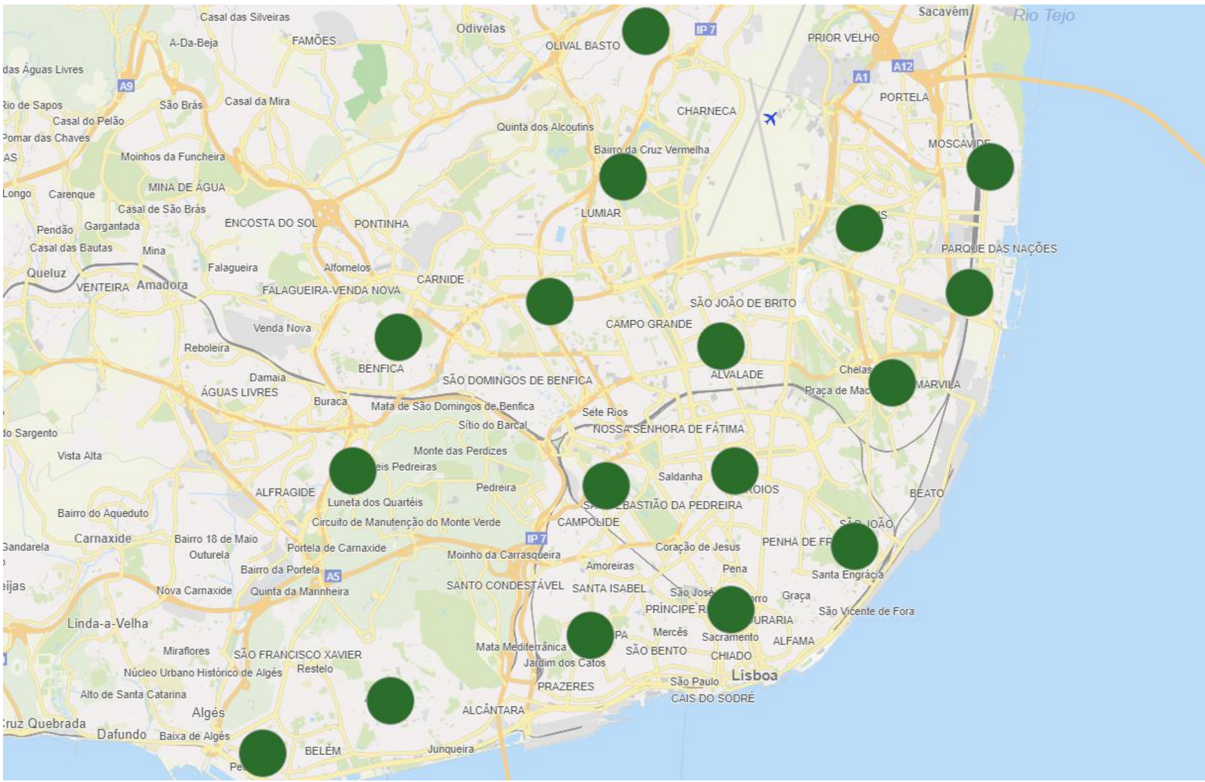
Start with the Clusters GeoAnalytics Bubble Layer properties, select **Appearance** and expand the **Shape and Size** pane. Select a larger radius, for example minimum of 24 (we can also vary the size based on an expression, which is covered further in this exercise).



Now in the same **Appearance** properties pane expand **Colors.** Turn off the **Auto** selector to enter in **Custom** mode. Select **Single Color** from the dropdown and a color of your choosing (I will select green).

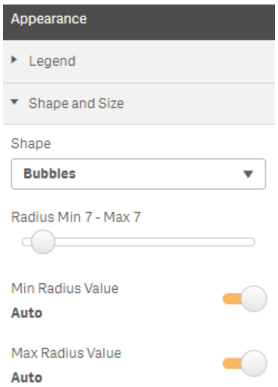


Here is a quick example of the outcome:

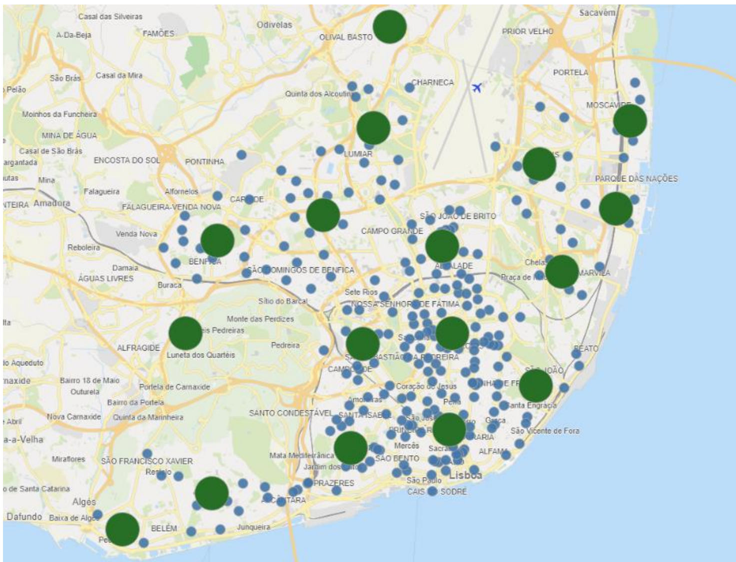


Now configure the Pharmacies Bubble Layer, but making the bubbles smaller. Don't forget to make this layer visible again if you've hidden it before.

Select the Pharmacies GeoAnalytics Bubble Layer properties, select **Appearance** and expand the **Shape and Size** pane. Select a smaller radius, for example minimum of 7.

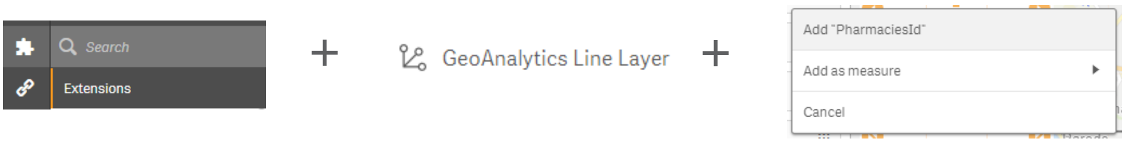


>

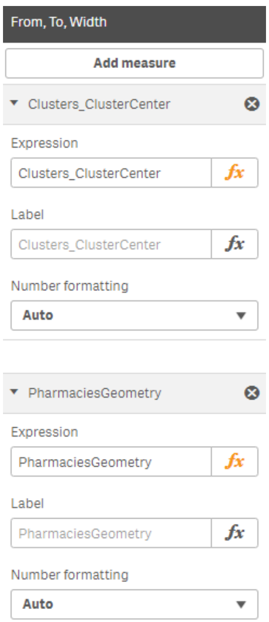


Now that we can see both clusters and pharmacies in the same geographical representation we can take it a step further. We can show visually how actually each pharmacy relates with it's cluster. For that we can use the **GeoAnalytics Line Layer** to draw connection line between the Clusters and the Pharmacies. Finally, we could also add a quick hover context at the Cluster bubble to identify how many Pharmacies belong to that Cluster.

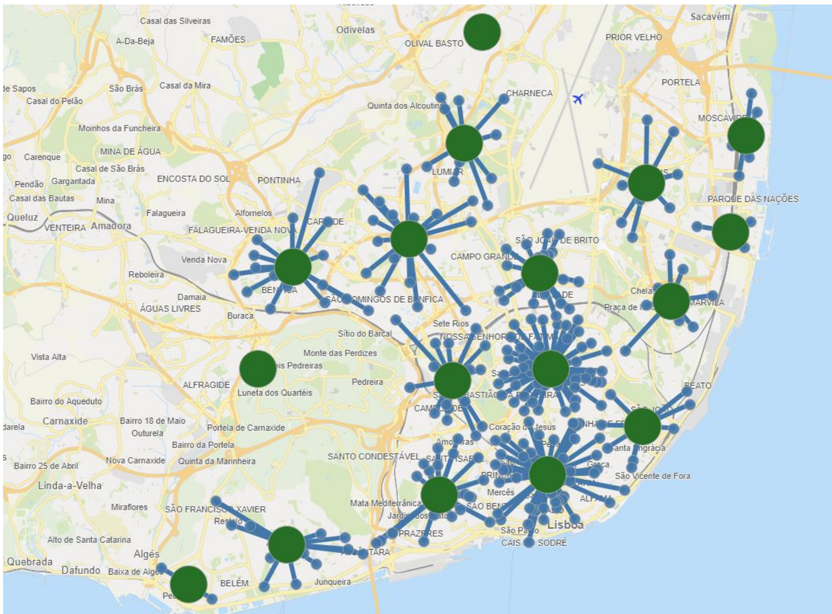
Starting with the lines, add a **GeoAnalytics Line Layer** to your sheet and drag *PharmaciesId* as our relationship field to be used as ID/Dimension.



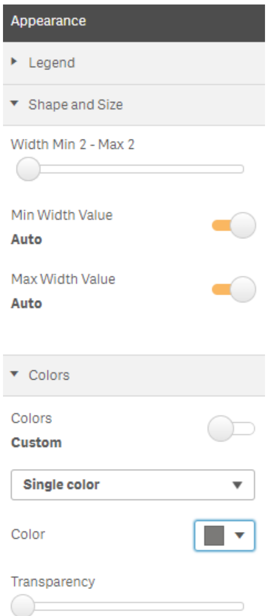
Go to the **From, To, Width** pane to configure that information needed to draw the line. Add *Clusters_ClusterCenter* as Origin and *PharmaciesGeometry* as Destination.



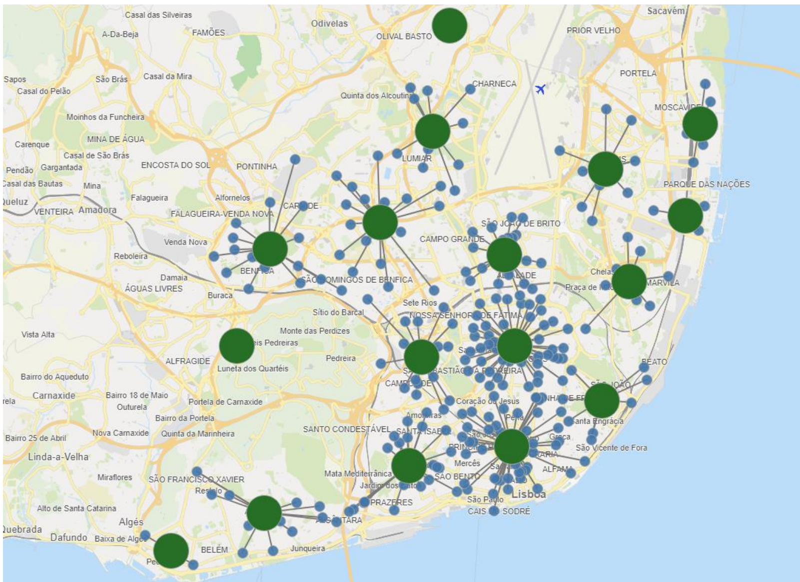
>



You can also reduce the line width and give it a different color to improve readability. Select **Appearance**, go to the **Shape and Size** pane and set a smaller width, such as 2. For the color, go to the **Colors** pane and select grey (#7b7a78).

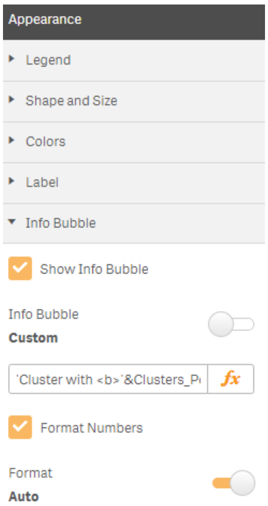


>

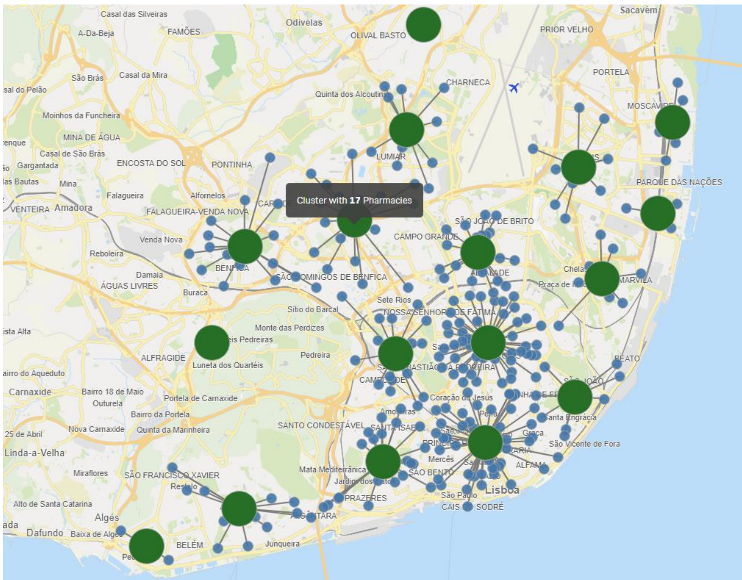


As last steps, we can now configure the hover behavior for the Cluster related GeoAnalytics Bubble Layer. Jump to the Clusters' **GeoAnalytics Bubble Layer** properties and select the **Appearance** pane. Under the **Info Bubble** pane deactivate the **Auto** feature and enter the following HTML in the expression's text area:

'Cluster with '&Clusters_PointCount&' Pharmacies'

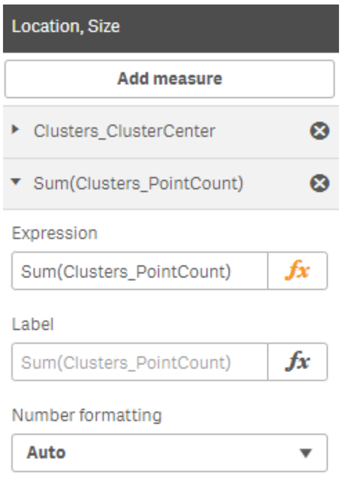


>

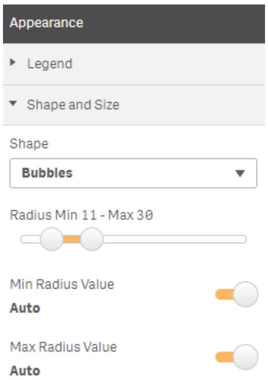


And we can also make the cluster size vary based on the number of points (pharmacies) associated with the cluster. Once more, go to the Clusters' **GeoAnalytics Bubble Layer** properties and select the **Location, Size** pane and add a new expression:

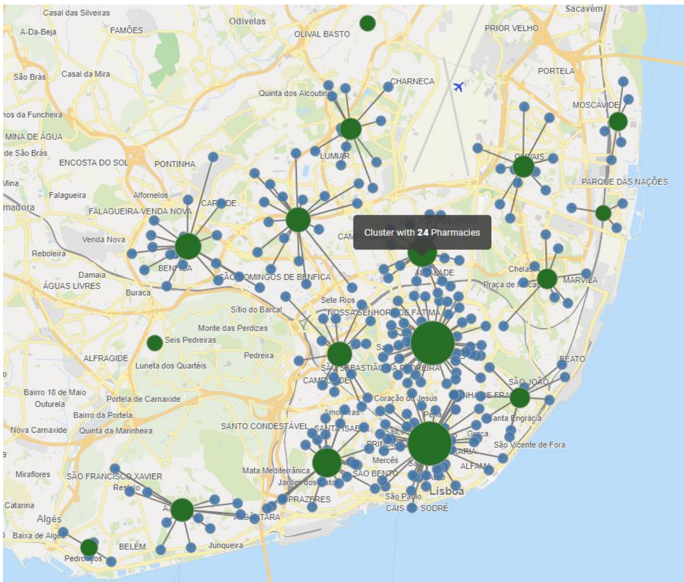
Sum(Clusters_PointCount)



Now go to the **Appearance** pane and (re)configure the **Shape and** Size, giving a minimum value of 11 and a maximum value of 30.



>



Exercise 2: Routes

Another representation that we could leverage based on our Clusters' calculated data is the route. For example, which would be the fastest path by car from a Cluster to a Pharmacy.

Go to the **Data load editor,** make sure your cursor is positioned at the end of the script. We will need to create a new table that has the required information so the function can calculate correctly. This will be the route's origin and destiny as well as the corresponding Pharmacies and Clusters IDs. We must make sure it is a correct "one to many" identification instead of a "many to many".

To do so, add the following snippet to your script:

```
RoutesRaw:
load PharmaciesId, PharmaciesGeometry as Origin Resident [Pharmacies];
join (RoutesRaw) load PharmaciesId, Clusters_ClusterID as DestinationId Resident [ClusterAssociations];
join (RoutesRaw) load Clusters_ClusterID as DestinationId, Clusters_ClusterCenter as Destination Resident Clusters;
```

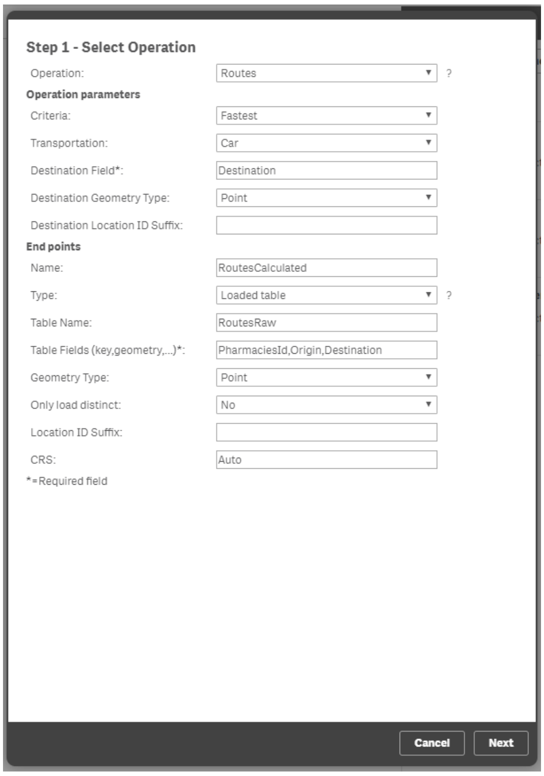
Once again, make sure your cursor is positioned at the end of the script and click on the **Select data** icon ( ) of your GeoAnalytics Connection. Select the **Operation: Routes**.

> Note: always check the GeoAnalytics reference page for full details using the Routes Operation: https://bi.idevio.com/wp-content/qlik/geoanalytics/releases/IdevioGeoAnalyticsConnector-5.13.0/doc/geoanalytics_reference-September_2018.html#Operation.Routes

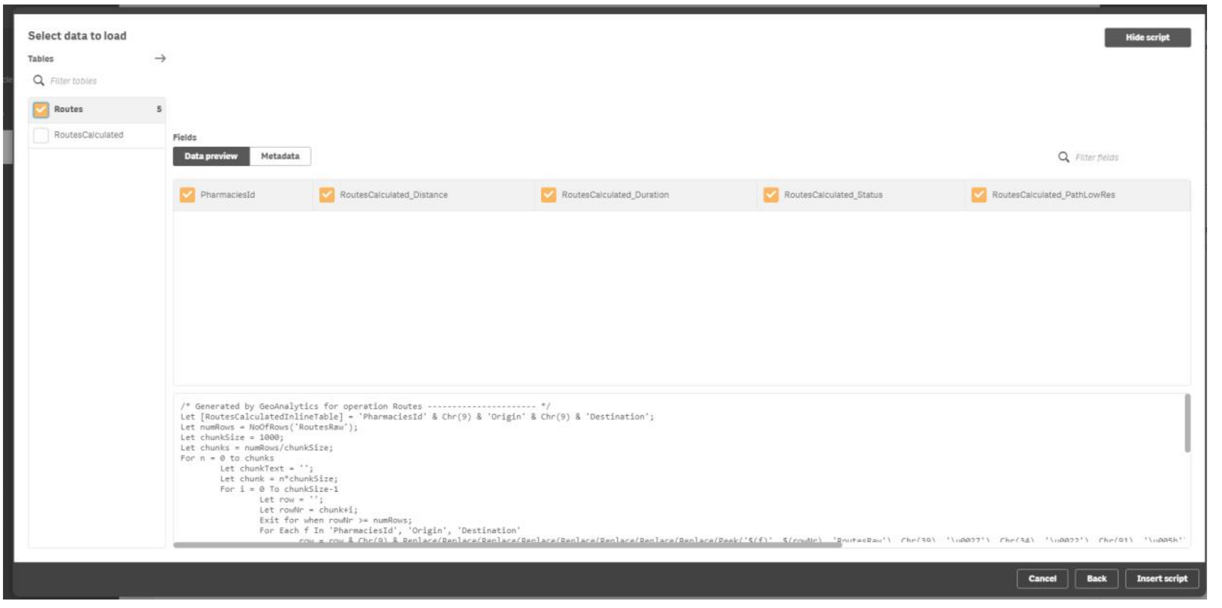
In the **Operation parameters** section for the **Criteria** choose *Fastest,* for **Transportation** select *Car*, as the **Destination Field** type *Destination* and for the **Destination Geometry Type** leave selected *Point.*

In the **End points** section input the **Name** as *RoutesCalculated,* select the **Type** as *Loaded table,* **Table Name** as *RoutesRaw,* **Table Fields (key,geometry,…)** with *PharmaciesId,Origin,Destination.* Select the **Geometry Type** as *Point* and leave the rest of the fields as default.

When all the options are filed, click on **Next.** Here is an example of the configurations above:



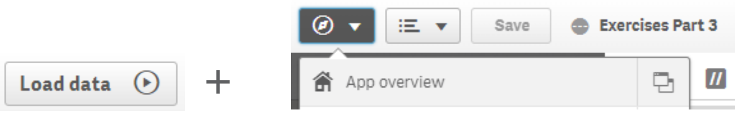
As a result, the connector will show the tables selection wizard. Check the **Routes** table, make sure all the fields are selected and click in **Insert Script**.



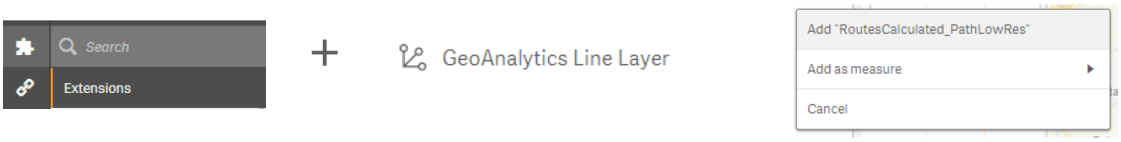
Here is an example of the expected script generated by the connector:

```
LIB CONNECT TO 'GeoAnalytics';

/* Generated by GeoAnalytics for operation Routes ---------------------- */
Let [RoutesCalculatedInlineTable] = 'PharmaciesId' & Chr(9) & 'Origin' & Chr(9) & 'Destination';
Let numRows = NoOfRows('RoutesRaw');
Let chunkSize = 1000;
Let chunks = numRows/chunkSize;
For n = 0 to chunks
	Let chunkText = '';
	Let chunk = n*chunkSize;
	For i = 0 To chunkSize-1
		Let row = '';
		Let rowNr = chunk+i;
		Exit for when rowNr >= numRows;
		For Each f In 'PharmaciesId', 'Origin', 'Destination'
			row = row & Chr(9) & Replace(Replace(Replace(Replace(Replace(Replace(Replace(Replace(Peek('$(f)', $(rowNr), 'RoutesRaw'), Chr(39), '\u0027'), Chr(34), '\u0022'), Chr(91), '\u005b'), Chr(47), '\u002f'), Chr(42), '\u002a'), Chr(59), '\u003b'), Chr(125), '\u007d'), Chr(123), '\u007b');
		Next
		chunkText = chunkText & Chr(10) & Mid('$(row)', 2);
	Next
	[RoutesCalculatedInlineTable] = [RoutesCalculatedInlineTable] & chunkText;
Next
chunkText=''


[Routes]:
SQL SELECT [PharmaciesId], [RoutesCalculated_Distance], [RoutesCalculated_Duration], [RoutesCalculated_Status], [RoutesCalculated_PathLowRes] FROM Routes(criteria='fastest', transportation='car', dest='Destination', destType='Point', suffix='', dataset='RoutesCalculated')
DATASOURCE RoutesCalculated INLINE tableName='RoutesRaw', tableFields='PharmaciesId,Origin,Destination', geometryType='POINT', loadDistinct='NO', suffix='', crs='Auto' {$(RoutesCalculatedInlineTable)}
;
tag field [PharmaciesId] with '$primarykey';
tag field [RoutesCalculated_PathLowRes] with '$geoline';
tag field [PharmaciesId] with '$geoname';
tag field [RoutesCalculated_PathLowRes] with '$relates_PharmaciesId';
tag field [PharmaciesId] with '$relates_RoutesCalculated_PathLowRes';

[RoutesCalculatedInlineTable] = '';

/* End GeoAnalytics operation Routes ----------------------------------- */
```

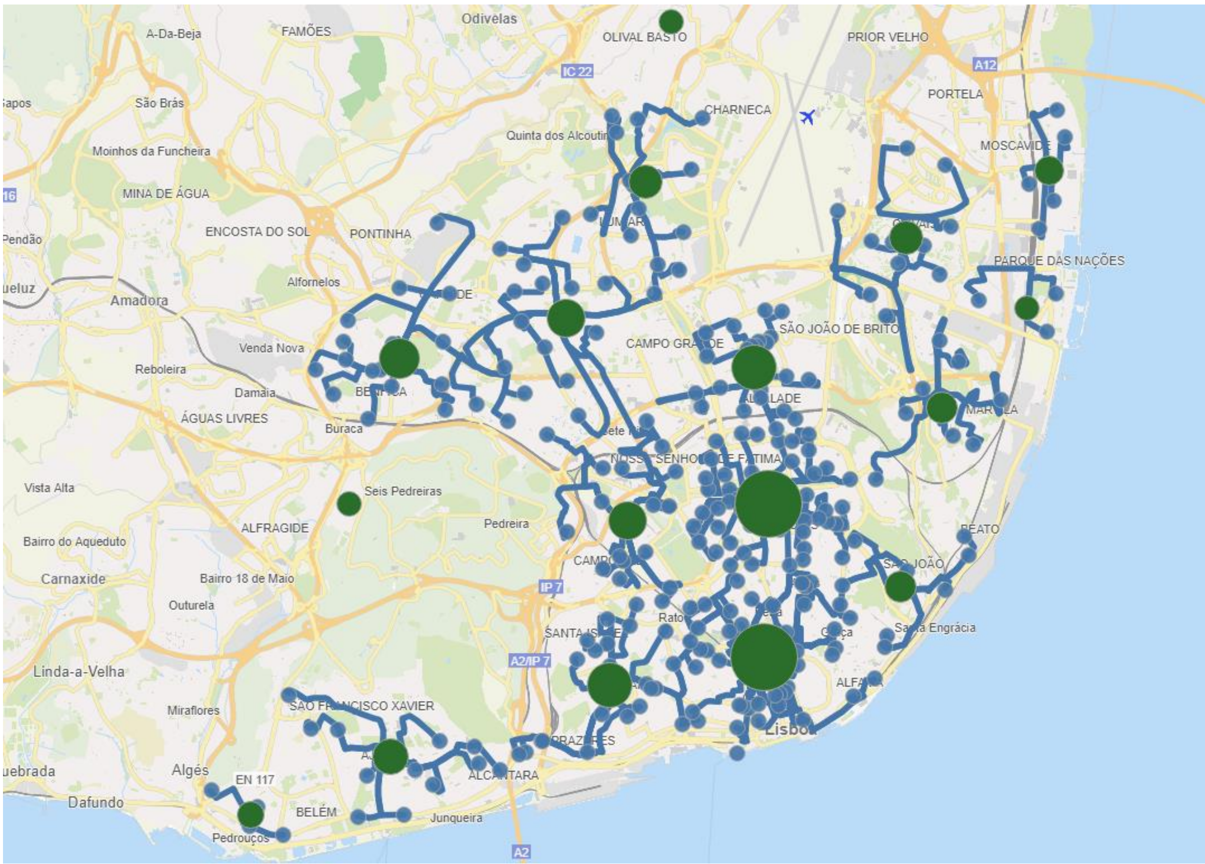
Load your data and return to your previously created sheet to test your results.



Add a new **GeoAnalytics Line Layer** to your sheet and drag the **RoutesCalculated_PathLowRes** field to it..



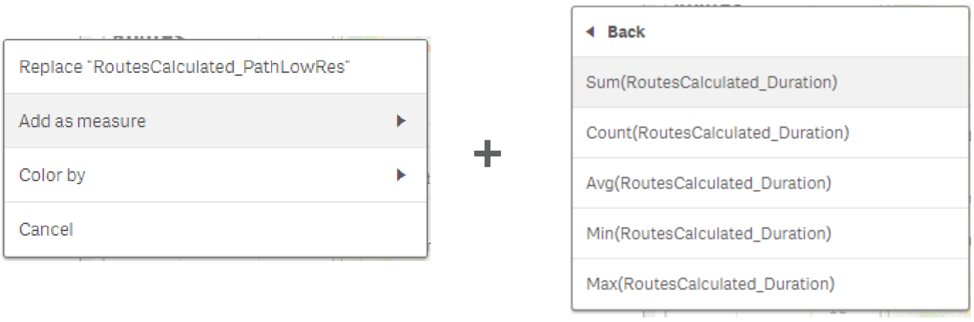
You can now see represented on your map the Routes for the fastest path by car between the Pharmacies and the center points of the Clusters they are allocated to.



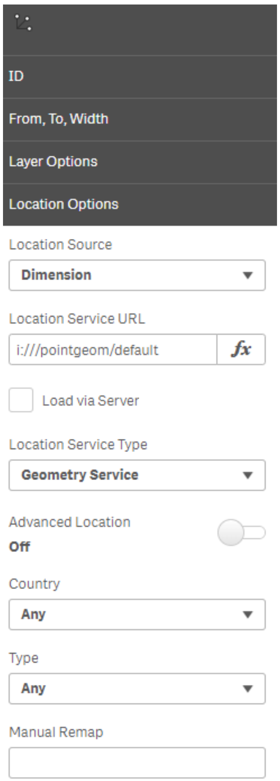
You can add some more visual information by fine tuning some appearance properties, such as reduce the path width and color it based on the *expected* travel time. We will force a color coding of our own instead of using the default **Colors** *by Measure* available in the **Appearance** properties pane.

If you go now to the **GeoAnalytics Line Layer**'s properties, under the **Appearance** pane and select **Colors**, when indicating you want to use custom colors, you should only be able to select either *Single Color, by dimension* or *by measure* as color source options. To enable the *by expression* option you must add a measure to the extension to "boot up" the extensions' hypercube.

Drag a field, such as *RoutesCalculated_Duration* to the extension and add it as an expression, such as *sum(RoutesCalculated_Duration)*. This will make sure the extension has an hypercube.



You will now see that under the **Appearance** pane **Colors,** the option *By expression* is available. The problem is that now the extension is also trying to use the expression we've specified as a location. To make sure it keeps using the routes provided before go to the **Location Options** and under **Location Source** select *Dimension.*



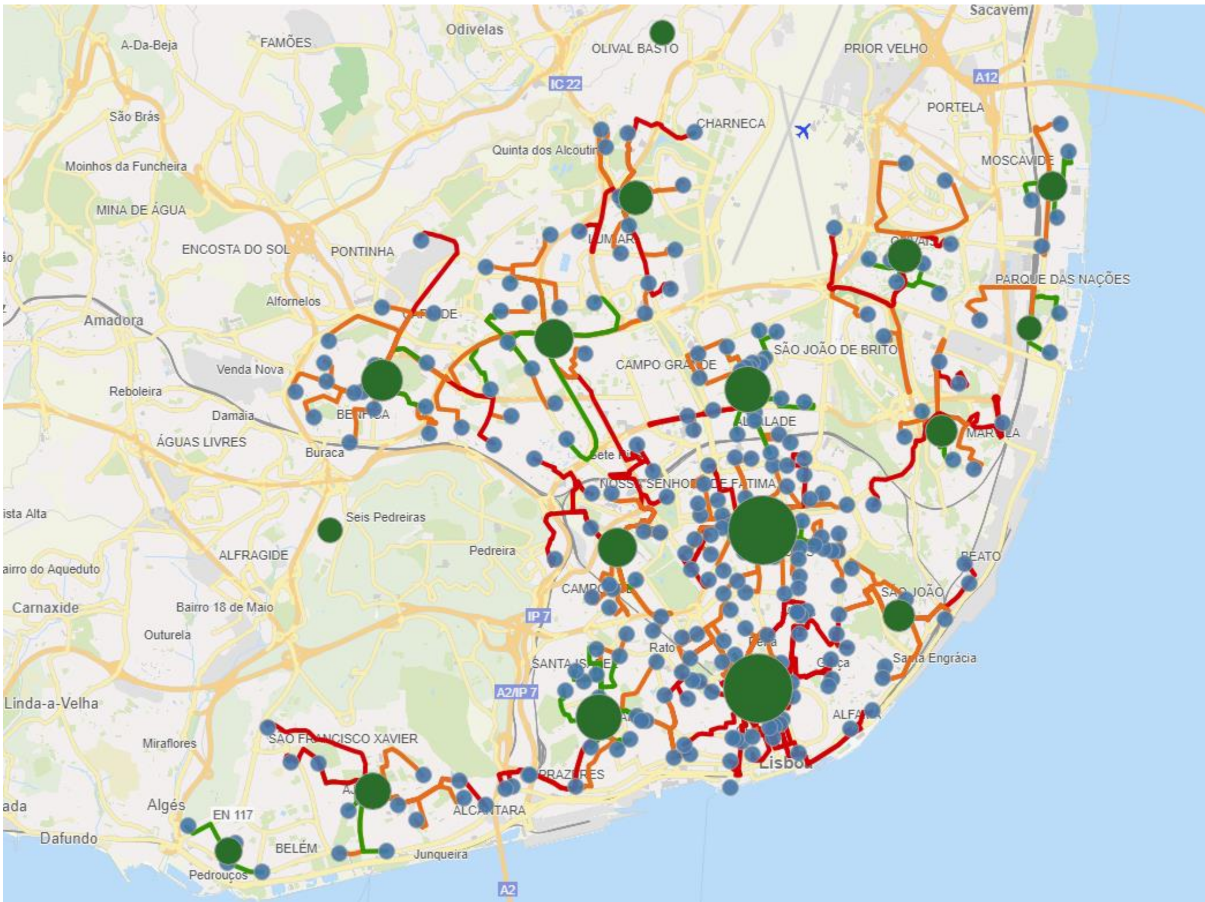
As a final step, we can now specify our color expression for the routes. Go once more to the **Appearance** pane, select **Colors** and the option *By expression.* Add the following expression:

=if(sum(RoutesCalculated_Duration)<180,rgb(48,150,0), if(sum(RoutesCalculated_Duration)>=180 and sum(RoutesCalculated_Duration)<300,rgb(229,110,20),rgb(210,0,1)))

The expression translates the sum of seconds to the following criteria:

Time	Color
< 3 minutes	Green
>= 3 minutes and < 5 minutes	Orange
>= 5 minutes	Red

Here is an example of your final result:



Exercise 3: Closest

We were able to calculate all the routes between the Clusters and the Pharmacies related with them. Another use case can be to only show the X closest. For example, which are the 10 closest Pharmacies to each Cluster center point at a maximum of 5 minutes of travel time?

To answer this use case, the GeoAnalytics Connector has a function, named Closest, that does this calculation.

Go to the **Data load editor,** make sure your cursor is positioned at the end of the script. **Select data** icon ( ) of your GeoAnalytics Connection and select the **Operation: Closest**.

> Note: always check the GeoAnalytics reference page for full details using the Closest Operation: https://bi.idevio.com/wp-content/qlik/geoanalytics/releases/IdevioGeoAnalyticsConnector-5.13.0/doc/geoanalytics_reference-September_2018.html#Operation.Closest

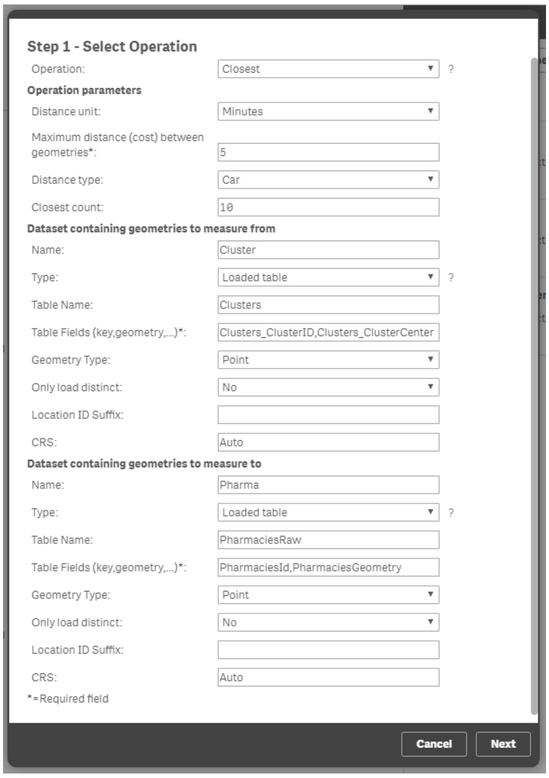
In the **Operation parameters** section for the **Distance unit** choose *Minutes.* In the **Maximum distance (cost) between geometries** field type 5. For the **Distance type** select *Car* and **Closest count** type in *10.*

For the section **Dataset containing geometries to measure from,** under **Name** type *Cluster.* The **Type** is *Loaded table* and the **Table name** is *Clusters.* The **Table fields (key,geometry,…)** are

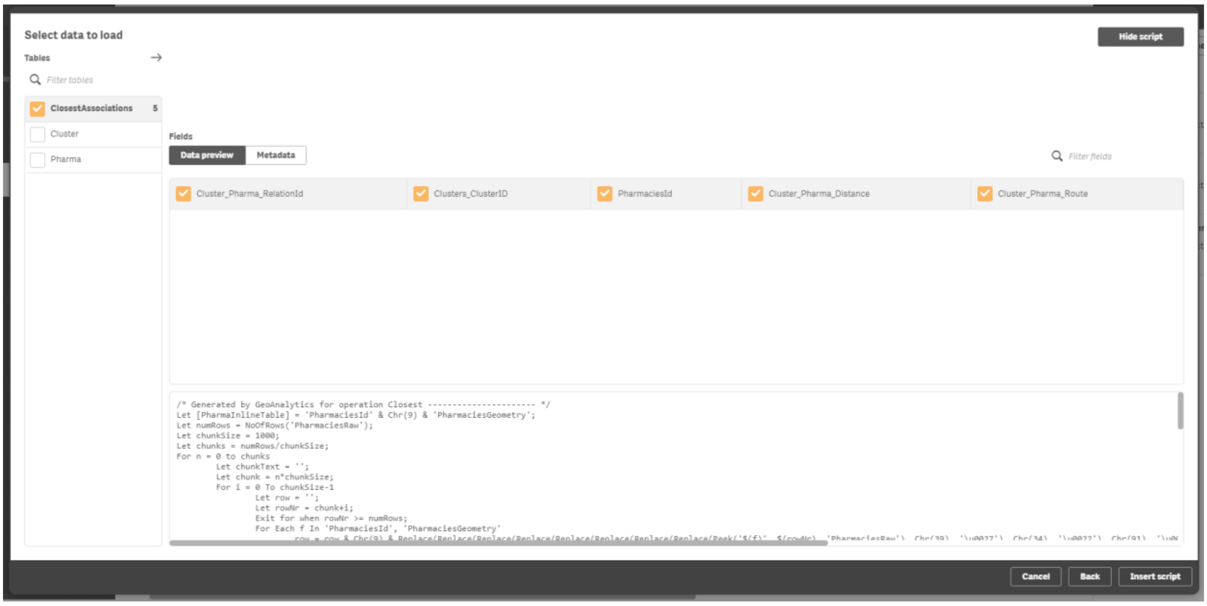
Clusters_ClusterID,Clusters_ClusterCenter. The **Geometry type** is *Point* and you can leave the rest of the fields of this section as default.

For the section **Dataset containing geometries to measure to** under **Name** type *Pharma.* The **Type** is *Loaded table* and the **Table name** is *PharmaciesRaw.* The **Table fields (key,geometry,…)** are *PharmaciesId,PharmaciesGeometry.* The **Geometry type** is *Point* and you can leave the rest of the fields of this section as default.

When all the options are filed, click on **Next.** Here is an example of the configurations above:



As a result, the connector will show the tables selection wizard. Check the **ClosestAssociations** table, make sure all the fields are selected and click in **Insert Script**.



Important note, the resulting table is a new associations table between the Pharmacies and the Clusters. When reloading you will have a synthetic key. Don't forget to optimize your datamodel accordingly, especially when your application is going to production.

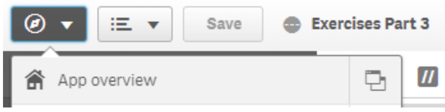
Here is an example of the expected script generated by the connector:

```
LIB CONNECT TO 'GeoAnalytics';

/* Generated by GeoAnalytics for operation Closest ---------------------- */
Let [PharmaInlineTable] = 'PharmaciesId' & Chr(9) & 'PharmaciesGeometry';
Let numRows = NoOfRows('PharmaciesRaw');
Let chunkSize = 1000;
Let chunks = numRows/chunkSize;
For n = 0 to chunks
	Let chunkText = '';
	Let chunk = n*chunkSize;
	For i = 0 To chunkSize-1
		Let row = '';
		Let rowNr = chunk+i;
		Exit for when rowNr >= numRows;
		For Each f In 'PharmaciesId', 'PharmaciesGeometry'
			row = row & Chr(9) & Replace(Replace(Replace(Replace(Replace(Replace(Replace(Replace(Peek('$(f)', $(rowNr), 'PharmaciesRaw'), Chr(39), '\u0027'), Chr(34), '\u0022'), Chr(91), '\u005b'), Chr(47), '\u002f'), Chr(42), '\u002a'), Chr(59), '\u003b'), Chr(125), '\u007d'), Chr(123), '\u007b');
		Next
		chunkText = chunkText & Chr(10) & Mid('$(row)', 2);
	Next
	[PharmaInlineTable] = [PharmaInlineTable] & chunkText;
Next
chunkText=''
Let [ClusterInlineTable] = 'Clusters_ClusterID' & Chr(9) & 'Clusters_ClusterCenter';
Let numRows = NoOfRows('Clusters');
Let chunkSize = 1000;
Let chunks = numRows/chunkSize;
For n = 0 to chunks
	Let chunkText = '';
	Let chunk = n*chunkSize;
	For i = 0 To chunkSize-1
		Let row = '';
		Let rowNr = chunk+i;
		Exit for when rowNr >= numRows;
		For Each f In 'Clusters_ClusterID', 'Clusters_ClusterCenter'
			row = row & Chr(9) & Replace(Replace(Replace(Replace(Replace(Replace(Replace(Replace(Peek('$(f)', $(rowNr), 'Clusters'), Chr(39), '\u0027'), Chr(34), '\u0022'), Chr(91), '\u005b'), Chr(47), '\u002f'), Chr(42), '\u002a'), Chr(59), '\u003b'), Chr(125), '\u007d'), Chr(123), '\u007b');
		Next
		chunkText = chunkText & Chr(10) & Mid('$(row)', 2);
	Next
	[ClusterInlineTable] = [ClusterInlineTable] & chunkText;
Next
chunkText=''


[ClosestAssociations]:
SQL SELECT [Cluster_Pharma_RelationId], [Clusters_ClusterID], [PharmaciesId], [Cluster_Pharma_Distance], [Cluster_Pharma_Route] FROM Closest(costUnit='Minutes', distance='5', distanceType='car', closestCount='10', dataset1='Cluster', dataset2='Pharma')
DATASOURCE Cluster INLINE tableName='Clusters', tableFields='Clusters_ClusterID,Clusters_ClusterCenter', geometryType='POINT', loadDistinct='NO', suffix='', crs='Auto' {$(ClusterInlineTable)}
DATASOURCE Pharma INLINE tableName='PharmaciesRaw', tableFields='PharmaciesId,PharmaciesGeometry', geometryType='POINT', loadDistinct='NO', suffix='', crs='Auto' {$(PharmaInlineTable)}
;
tag field [Cluster_Pharma_RelationId] with '$primarykey';
tag field [Cluster_Pharma_Route] with '$geoline';
tag field [Cluster_Pharma_RelationId] with '$geoname';
tag field [Cluster_Pharma_Route] with '$relates_Cluster_Pharma_RelationId';
tag field [Cluster_Pharma_RelationId] with '$relates_Cluster_Pharma_Route';

[ClusterInlineTable] = '';
[PharmaInlineTable] = '';
```

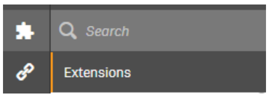
/* End GeoAnalytics operation Closest ---------------------------------- */

Load your data and return to your previously created sheet to test your results.

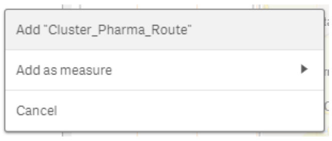
+



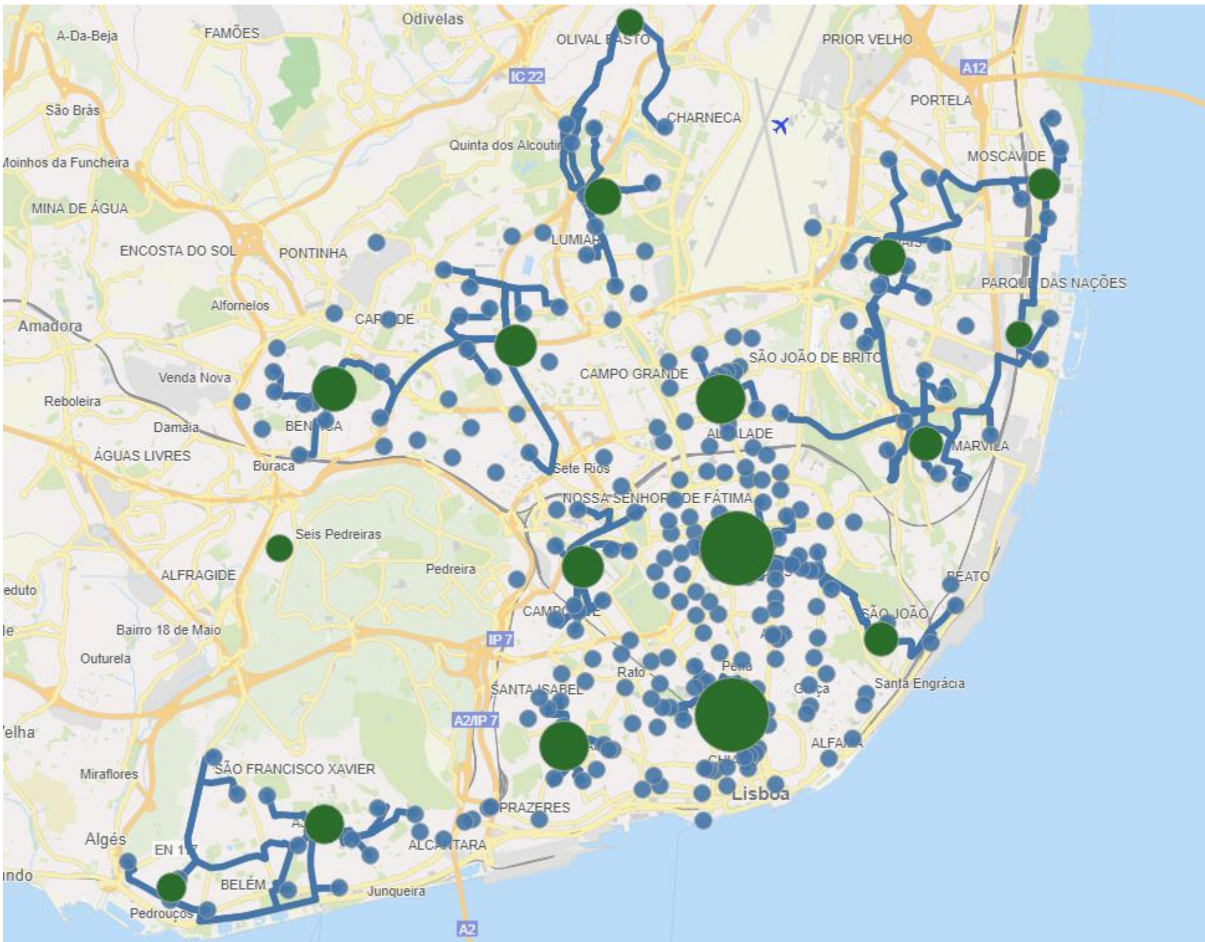
Add a new **GeoAnalytics Line Layer** to your sheet and drag the **Cluster_Pharma_Route** field.



+



You can now see represented on your map the Routes for the 10 closest Pharmacies to each Cluster. Notice that albeit Pharmacies can be perceived close to a Cluster on map it doesn't mean they have the fastest travel time, being in fact associated to a further Cluster that has faster access roads.



Exercise 4: Dissolve

The next exercise will show you how to dissolve multiple polygons into a single larger polygon. To achieve this, we will use the GeoAnalytics Connector function Dissolve.

Jump to the **Data load editor,** make sure your cursor is positioned at the end of the script. We will first load some polygons. These can be found here:

https://services.arcgis.com/1dSrzEWVQn5kHHyK/ArcGIS/rest/services/Limites_Cartografia/FeatureServer/0/query?where=1%3D1&outFields=*&f=pgeojson

Select the data icon ( ) of your GeoAnalytics Connection and select the **Operation: Load**.

> Note: always check the GeoAnalytics reference page for full details using the Load Operation: https://bi.idevio.com/wp-content/qlik/geoanalytics/releases/IdevioGeoAnalyticsConnector-5.13.0/doc/geoanalytics_reference-September_2018.html#Operation.Load

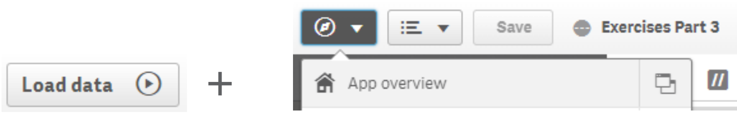
Give *Parish* as **Name** for your *Dataset*, select *File based geo dataset* for **Type** and paste the link in **URL.** You may leave all the other options as suggested by default and click on **Next.**

We will not need all the fields available. You may **select and rename** *Parish_Geometry* to *ParishGeometry,* **select and rename** *Shape__Length* to *ParishLength,* **select and rename** *Shape_Area* to *ParishArea,* **select and rename** *AREA_M2* to *ParishAreaM2,* **select and rename** *NOME* to *ParishName* and **select and rename** *_autoIndex_ to ParishId.* Once you've selected and renamed the fields you may click on **Insert Script.**

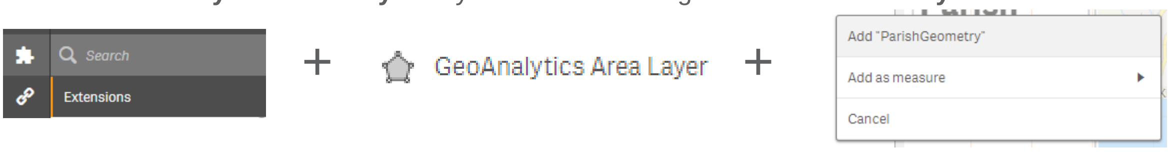
The following script is expected to have been generated for you:

```
LIB CONNECT TO 'GeoAnalytics';

/* Generated by GeoAnalytics for operation Load --------------------- */
[Parish]:
SQL SELECT [Parish_Geometry] AS  [ParishGeometry], [Shape__Length] AS  [ParishLength], [Shape__Area] AS  [ParishArea], [AREA_M2] AS [ParishAreaM2], [NOME] AS  [ParishName], [_autoIndex_] AS  [ParishId] FROM Load(dataset='Parish')
DATASOURCE Parish GEOFILE url='https://services.arcgis.com/1dSrzEWVQn5kHHyK/ArcGIS/rest/services/Limites_Cartografia/FeatureServer/1/query?where=1%3D1&outFields=*&f=pgeojson', keyField='', type='auto', expectedGeomType='auto', encoding='', crs='Auto'
;
tag field [ParishId] with '$primarykey';
tag field [ParishGeometry] with '$geopolygon';
tag field [ParishId] with '$geoname';
tag field [ParishGeometry] with '$relates_ParishId';
tag field [ParishId] with '$relates_ParishGeometry';
/* End GeoAnalytics operation Load ----------------------------------- */
```

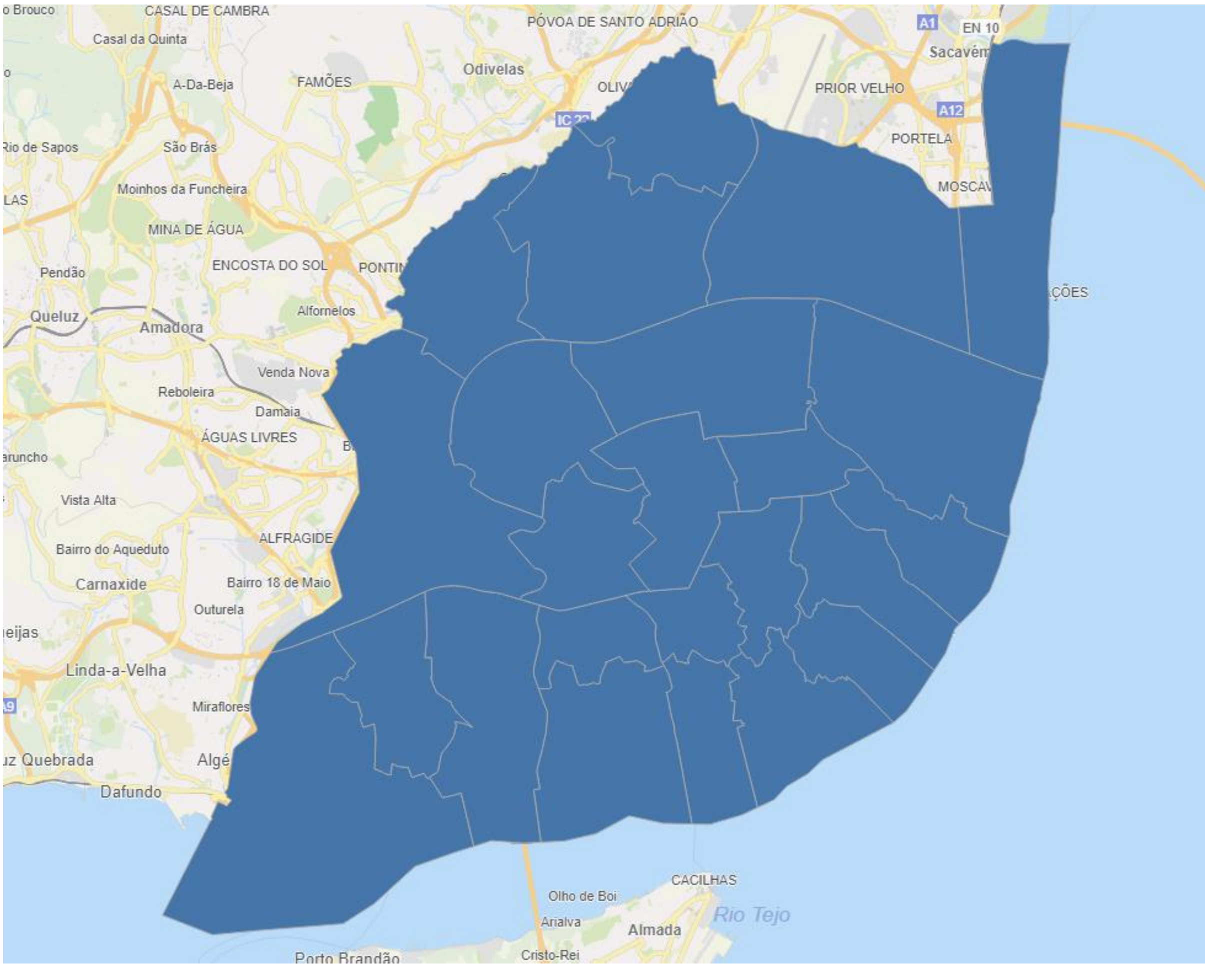
Load your data and return to your previously created sheet to test your results.



Add the **GeoAnalytics Area Layer** to your sheet and drag the **ParishGeometry** field to it.



You should now to be able to see all the Parishes of Lisbon city.



Now, to dissolve these polygons into a ~~blob~~ single polygon we will use the GeoAnalytics Connector Dissolve function.

Return to the **Data load editor,** make sure your cursor is positioned at the end of the script. As we did for the Routes exercise, we need to do a little pre-work to ensure we give the connector the right information about the relation between the polygons to dissolve together.

Add the following script:

```
ParishDissolvedRaw:
Load 'Lisboa' as DissolveField,
              ParishId,
   [ParishGeometry] as ParishGeo
Resident Parish;
```

This will create a resulting table associating all the *ParishId* values to the *Lisboa* value of the newly created field *DissolveField.* This way we will have a common factor that will result creating a single giant polygon by merging all the smaller ones when we use the GeoAnalytics Connector Dissolve function.

Once added the script **select the data** icon () of your GeoAnalytics Connection and select the **Operation: Dissolve**.

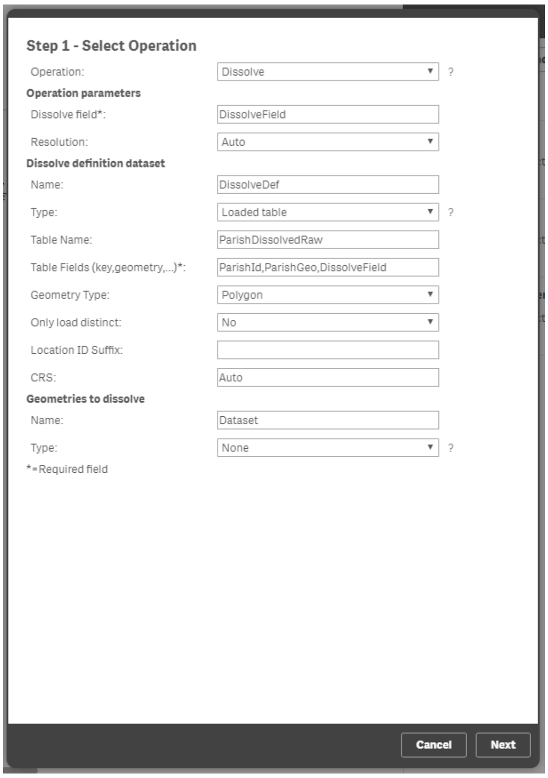
Note: always check the GeoAnalytics reference page for full details using the Dissolve Operation: https://bi.idevio.com/wp-content/qlik/geoanalytics/releases/IdevioGeoAnalyticsConnector-5.13.0/doc/geoanalytics_reference-September_2018.html#Operation.Dissolve

In the **Operation parameters** section for the **Dissolve field** type *DissolveField* and for **Resolution** leave it in *Auto.*

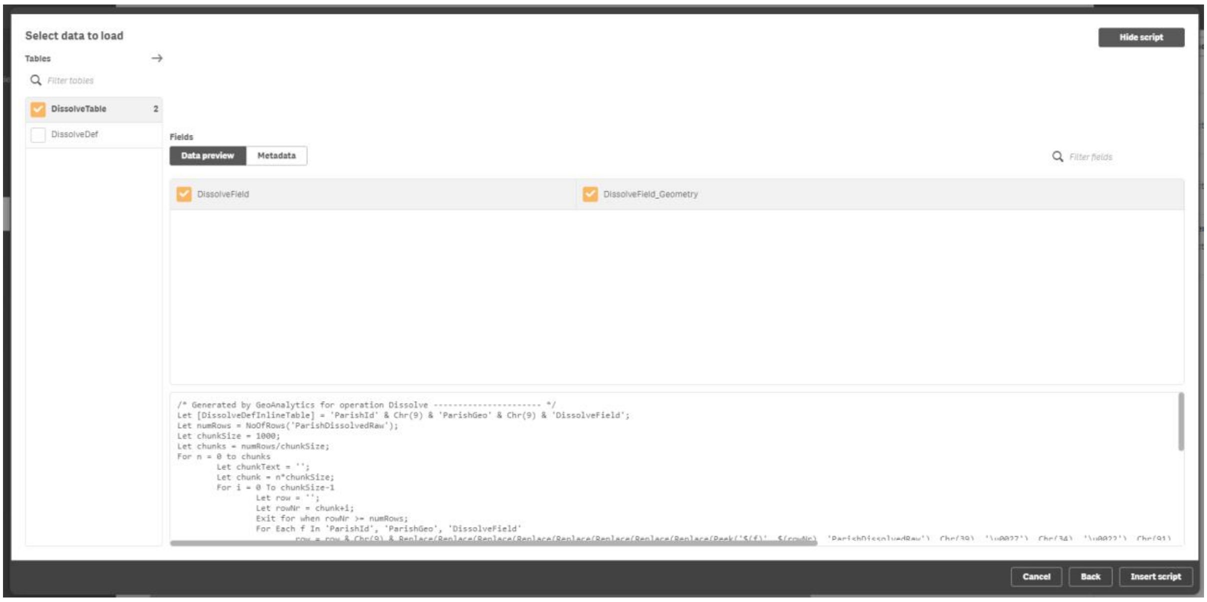
For the section **Dissolve definition dataset,** under **Name** type *DissolveDef.* The **Type** is *Loaded table* and the **Table name** is *ParishDissolvedRaw.* The **Table fields (key,geometry,…)** are *ParishId,ParishGeo,DissolveField.* The **Geometry type** is *Polygon* and you can leave the rest of the fields of this section as default.

For the section **Geometries to dissolve** select **Type** as *None.*

When all the options are filed, click on **Next.** Here is an example of the configurations above:



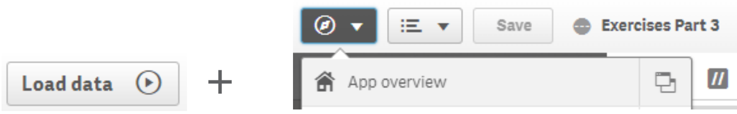
As a result, the connector will show the tables selection wizard. Check the **DissolveTable** table, make sure all the fields are selected and click in **Insert Script**.



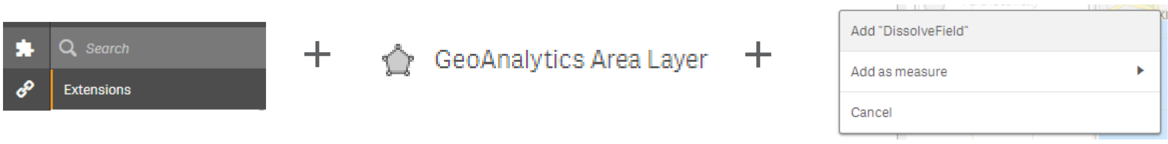
Here is an example of the expected script generated by the connector:

```
LIB CONNECT TO 'GeoAnalytics';

/* Generated by GeoAnalytics for operation Dissolve ---------------------- */
Let [DissolveDefInlineTable] = 'ParishId' & Chr(9) & 'ParishGeo' & Chr(9) & 'DissolveField';
Let numRows = NoOfRows('ParishDissolvedRaw');
Let chunkSize = 1000;
Let chunks = numRows/chunkSize;
For n = 0 to chunks
	Let chunkText = '';
	Let chunk = n*chunkSize;
	For i = 0 To chunkSize-1
		Let row = '';
		Let rowNr = chunk+i;
		Exit for when rowNr >= numRows;
		For Each f In 'ParishId', 'ParishGeo', 'DissolveField'
			row = row & Chr(9) & Replace(Replace(Replace(Replace(Replace(Replace(Replace(Peek('$(f)', $(rowNr), 'ParishDissolvedRaw'), Chr(39), '\u0027'), Chr(34), '\u0022'), Chr(91), '\u005b'), Chr(47), '\u002f'), Chr(42), '\u002a'), Chr(59), '\u003b'), Chr(125), '\u007d'), Chr(123), '\u007b');
		Next
		chunkText = chunkText & Chr(10) & Mid('$(row)', 2);
	Next
	[DissolveDefInlineTable] = [DissolveDefInlineTable] & chunkText;
Next
chunkText=''


[DissolveTable]:
SQL SELECT [DissolveField], [DissolveField_Geometry] FROM Dissolve(dissolveField='DissolveField', resolution='auto', dissolveDef='DissolveDef')
DATASOURCE DissolveDef INLINE tableName='ParishDissolvedRaw', tableFields='ParishId,ParishGeo,DissolveField', geometryType='POLYGON', loadDistinct='NO', suffix='', crs='Auto' {$(DissolveDefInlineTable)}
;
tag field [DissolveField] with '$primarykey';
tag field [DissolveField_Geometry] with '$geopolygon';
tag field [DissolveField] with '$geoname';
tag field [DissolveField_Geometry] with '$relates_DissolveField';
tag field [DissolveField] with '$relates_DissolveField_Geometry';

[DissolveDefInlineTable] = '';

/* End GeoAnalytics operation Dissolve ----------------------------------- */
```

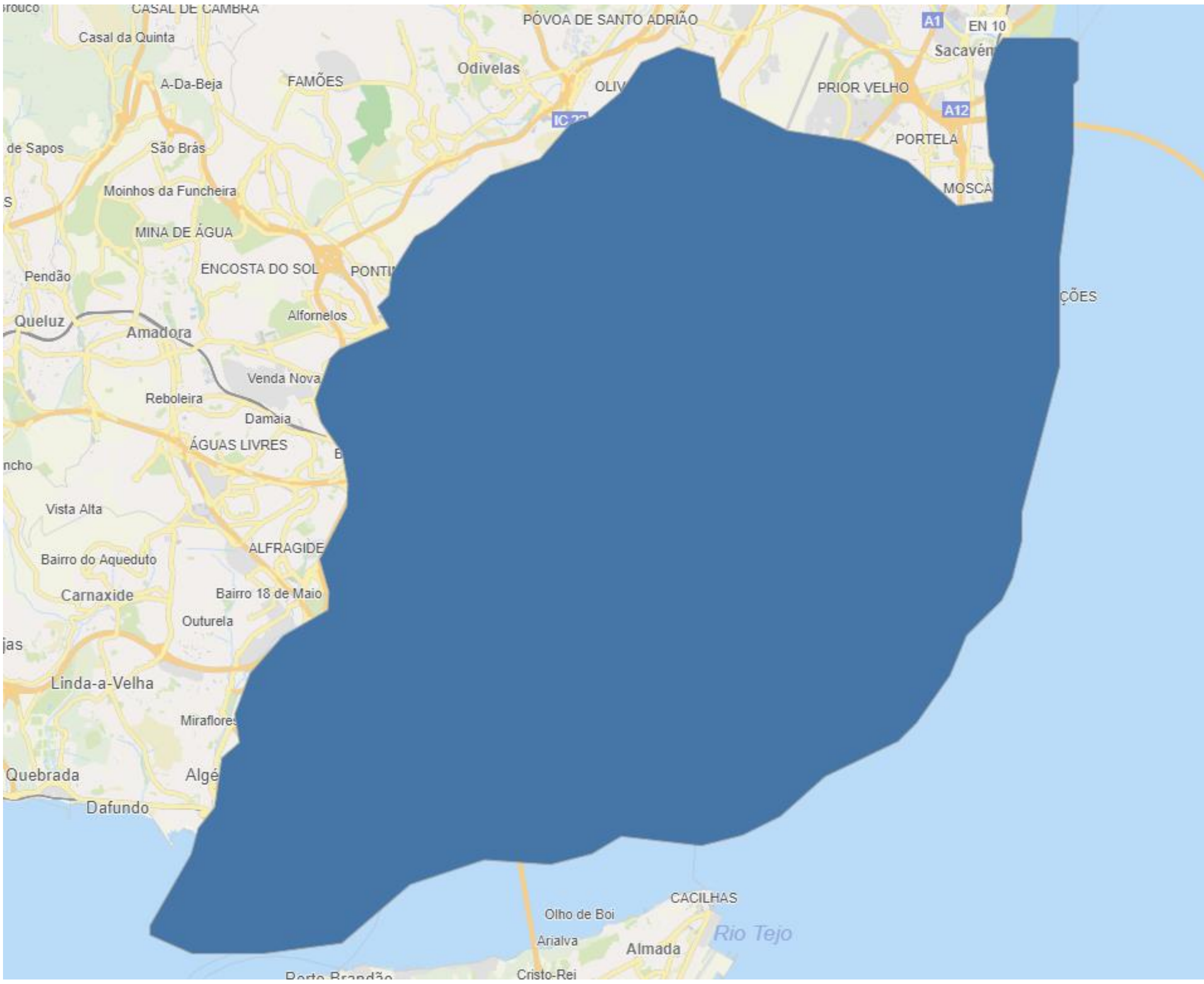
Load your data and return to your previously created sheet to test your results.



Add the **GeoAnalytics Area Layer** to your sheet and drag the **DissolveField** field to it.



You should now to be able to see a single polygon with the merge of all Parishes of Lisbon city.



Exercise 5: Intersect

As an alternative to the Within function, you can also calculate the intersection between components using the GeoAnalytics Connector. In this exercise we will calculate the intersection between the Pharmacies and the Parishes.

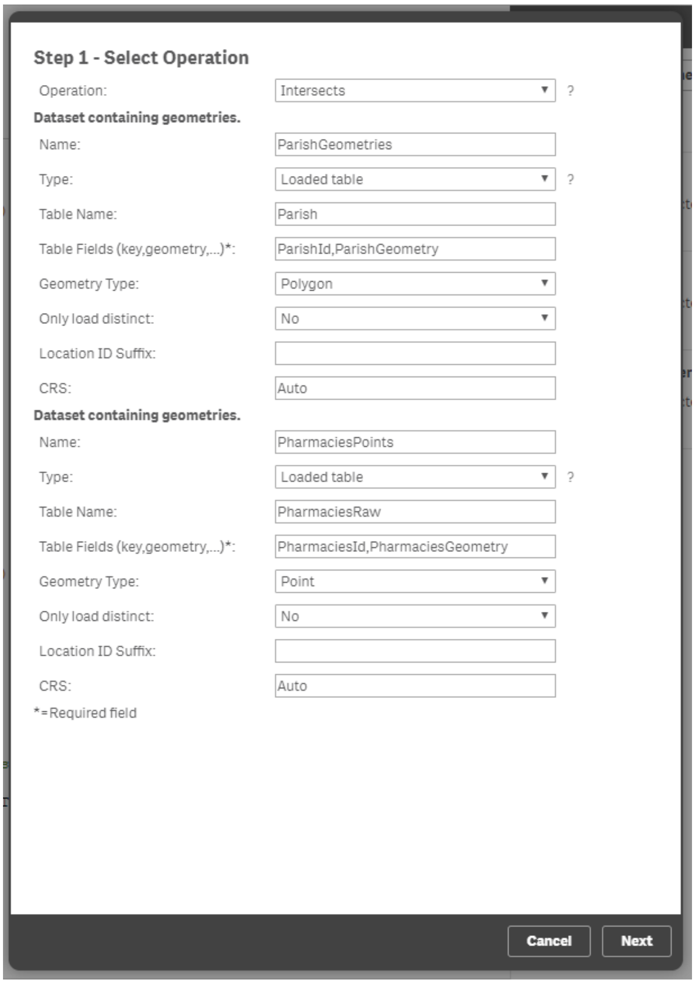
Go to the **Data load editor,** make sure your cursor is positioned at the end of the script. **Select data** icon () of your GeoAnalytics Connection and select the **Operation: Intersects**.

> Note: always check the GeoAnalytics reference page for full details using the Intersects Operation: https://bi.idevio.com/wp-content/qlik/geoanalytics/releases/IdevioGeoAnalyticsConnector-5.13.0/doc/geoanalytics_reference-September_2018.html#Operation.Intersects

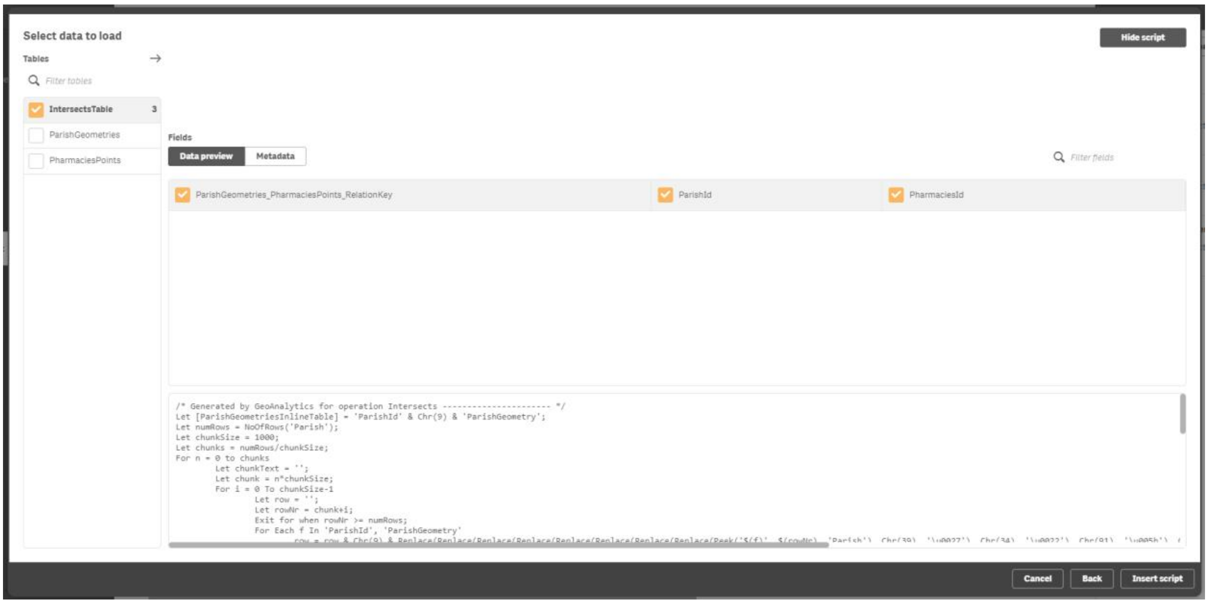
In the **Dataset containing geometries** section for the **Name** type *ParishGeometries* and select **Type** as *Loaded table.* For the **Table Name** type *Parish*, in **Table Fields (key,geometry,…)** type *ParishId,ParishGeometry* and set the **Geometry Type** as *Polygon.*

In the second **Dataset containing geometries** section for the **Name** type *PharmaciesPoints* and select **Type** as *Loaded table.* For the **Table Name** type *PharmaciesRaw*, in **Table Fields (key,geometry,…)** type *PharmaciesId,PharmaciesGeometry* and set the **Geometry Type** as *Point.*

You may leave all the other options as suggested by default and click on **Next**.



As a result, the connector will show the tables selection wizard. Check the **IntersectsTable** table, make sure all the fields are selected and click in **Insert Script**.

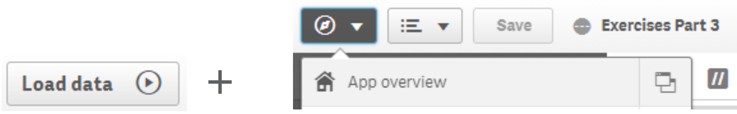


Here is an example of the expected script generated by the connector:

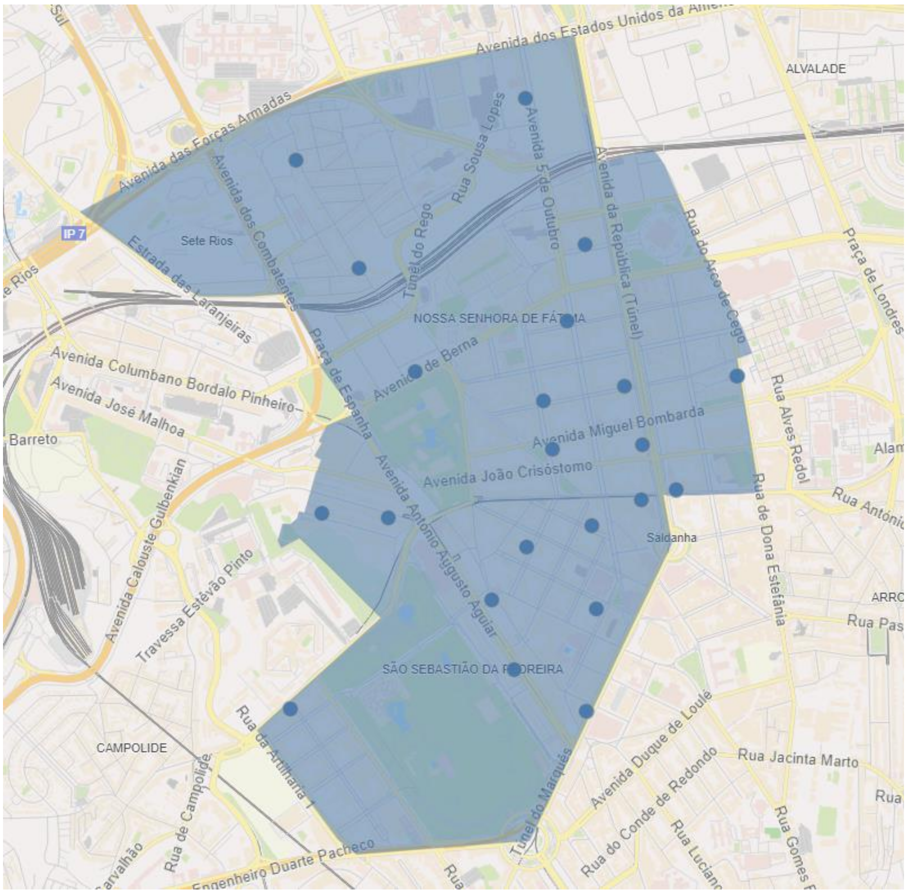
```
LIB CONNECT TO 'GeoAnalytics';

/* Generated by GeoAnalytics for operation Intersects ---------------------- */
Let [ParishGeometriesInlineTable] = 'ParishId' & Chr(9) & 'ParishGeometry';
Let numRows = NoOfRows('Parish');
Let chunkSize = 1000;
Let chunks = numRows/chunkSize;
For n = 0 to chunks
        Let chunkText = '';
        Let chunk = n*chunkSize;
        For i = 0 To chunkSize-1
                Let row = '';
                Let rowNr = chunk+i;
                Exit for when rowNr >= numRows;
                For Each f In 'ParishId', 'ParishGeometry'
                        row = row & Chr(9) & Replace(Replace(Replace(Replace(Replace(Replace(Replace(Replace(Peek('$(f)', $(rowNr), 'Parish'), Chr(39), '\u0027'), Chr(34), '\u0022'), Chr(91), '\u005b'), Chr(47), '\u002f'), Chr(42), '\u002a'), Chr(59), '\u003b'), Chr(125), '\u007d'), Chr(123), '\u007b');
                Next
                chunkText = chunkText & Chr(10) & Mid('$(row)', 2);
        Next
        [ParishGeometriesInlineTable] = [ParishGeometriesInlineTable] & chunkText;
Next
chunkText=''
Let [PharmaciesPointsInlineTable] = 'PharmaciesId' & Chr(9) & 'PharmaciesGeometry';
Let numRows = NoOfRows('PharmaciesRaw');
Let chunkSize = 1000;
Let chunks = numRows/chunkSize;
For n = 0 to chunks
        Let chunkText = '';
        Let chunk = n*chunkSize;
        For i = 0 To chunkSize-1
                Let row = '';
                Let rowNr = chunk+i;
                Exit for when rowNr >= numRows;
                For Each f In 'PharmaciesId', 'PharmaciesGeometry'
                        row = row & Chr(9) & Replace(Replace(Replace(Replace(Replace(Replace(Replace(Replace(Peek('$(f)', $(rowNr), 'PharmaciesRaw'), Chr(39), '\u0027'), Chr(34), '\u0022'), Chr(91), '\u005b'), Chr(47), '\u002f'), Chr(42), '\u002a'), Chr(59), '\u003b'), Chr(125), '\u007d'), Chr(123), '\u007b');
                Next
                chunkText = chunkText & Chr(10) & Mid('$(row)', 2);
        Next
        [PharmaciesPointsInlineTable] = [PharmaciesPointsInlineTable] & chunkText;
Next
```

```
chunkText=''


[IntersectsTable]:
SQL SELECT [ParishGeometries_PharmaciesPoints_RelationKey], [ParishId], [PharmaciesId] FROM Intersects(dataset1='ParishGeometries', dataset2='PharmaciesPoints')
DATASOURCE ParishGeometries INLINE tableName='Parish', tableFields='ParishId,ParishGeometry', geometryType='POLYGON', loadDistinct='NO', suffix='', crs='Auto' {$(ParishGeometriesInlineTable)}
DATASOURCE PharmaciesPoints INLINE tableName='PharmaciesRaw', tableFields='PharmaciesId,PharmaciesGeometry', geometryType='POINT', loadDistinct='NO', suffix='', crs='Auto' {$(PharmaciesPointsInlineTable)}
;
tag field [ParishId] with '$primarykey';

[ParishGeometriesInlineTable] = '';
[PharmaciesPointsInlineTable] = '';

/* End GeoAnalytics operation Intersects ----------------------------------- */
```

Load your data and return to your previously created sheet to test your results.



Try to make a selection on a Parish and check if the corresponding Pharmacies are displayed on your map.



Mobile “Near me” Location

Currently, Qlik Sense is not leveraging a web browser’s location capability. This can be extremely handy for users on the field requiring “near my” geo location information. You can offer this feature leveraging an open source extension available at Qlik Branch: **My Location Finder for Qlik Sense**.

https://developer.qlik.com/garden/56fc07ae509a5ad18831bef8

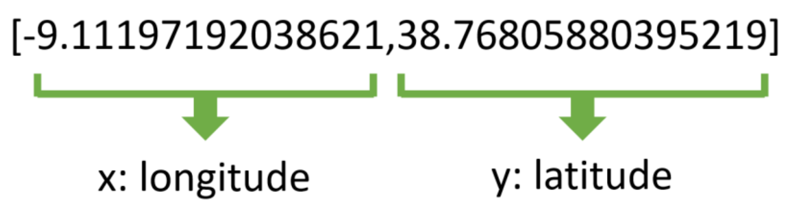
> Note: please refer to Qlik’s help site for documentation about how to install extensions
>
> - Qlik Sense Server: https://help.qlik.com/en-US/sense/September2018/Subsystems/ManagementConsole/Content/Sense_QMC/import-extensions.htm

To make the extension work you need to have within your datamodel two fields corresponding to Latitude and Longitude. It will ask to the user’s web browser location and perform an associative search on the fields based on a radius interval you also define in the extension’s properties.

On our examples, we could try to find the nearest Pharmacy. You are probably not located in Lisbon, but the idea of this exercise is that you can have a train of thought to use as base for your own developments.

If we look into the values in the **PharmaciesGeometry** field from exercise 1 in part 3, we can see they have a Point format ready to use, such as [-9.11197192038621,38.76805880395219].

We need to break this information to create two separate fields: one for latitude and another for longitude. And… of course for some reason, breaking this up right away will not get us the correct values. Here is why: the format actually represents [x,y] and on a map, Latitude is y whilst Longitude is x.



So, in our script all we need to do is to add a preceding load to our *PharmaciesRaw* and ensure we load the Latitudes and Longitudes correctly. Here is an example of the script you should add:

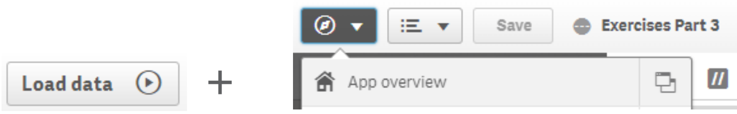
```
LOAD *,
     subfield(replace(PharmaciesGeometry,'[',''),',',1) as PharmaLongitude,
     subfield(replace(PharmaciesGeometry,']',''),',',2) as PharmaLatitude;
```

And for your convenience, if you prefer, you can copy the fully updated load script for the *PharmaciesRaw* table:

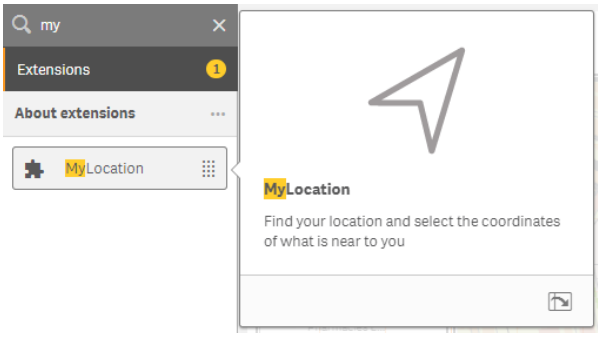
```
LIB CONNECT TO 'GeoAnalytics';

/* Generated by GeoAnalytics for operation Load --------------------- */
[PharmaciesRaw]:
LOAD *,
     subfield(replace(PharmaciesGeometry,'[',''),',',1) as PharmaLongitude,
     subfield(replace(PharmaciesGeometry,']',''),',',2) as PharmaLatitude;
SQL SELECT [PharmaciesRaw_Geometry] AS [PharmaciesGeometry], [INF_SITE] AS [PharmaciesWebSite], [INF_NOME] AS [PharmaciesName], [INF_MORADA] AS [PharmaciesAddress], [_autoIndex_] AS [PharmaciesId] FROM Load(dataset='PharmaciesRaw')
```

```
DATASOURCE PharmaciesRaw GEOFILE
url='https://services.arcgis.com/1dSrzEWVQn5kHHyK/arcgis/rest/services/POISaude/FeatureServer/1/query?where=1%3D1&outFields=*&f
=pgeojson', keyField='', type='auto', expectedGeomType='auto', encoding='', crs='Auto'
;
tag field [PharmaciesId] with '$primarykey';
tag field [PharmaciesGeometry] with '$geopoint';
tag field [PharmaciesId] with '$geoname';
tag field [PharmaciesGeometry] with '$relates_PharmaciesId';
tag field [PharmaciesId] with '$relates_PharmaciesGeometry';
/* End GeoAnalytics operation Load ---------------------------------- */
```

Load your data and return to your previously created sheet to test your results.

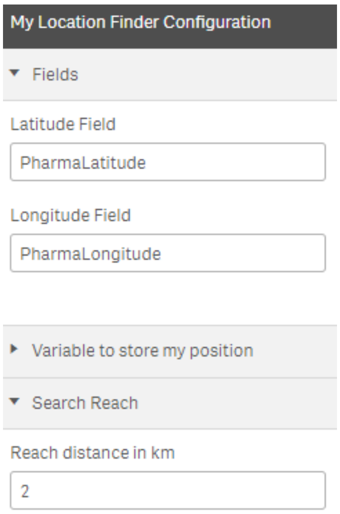


Drag the Extension to your sheet.

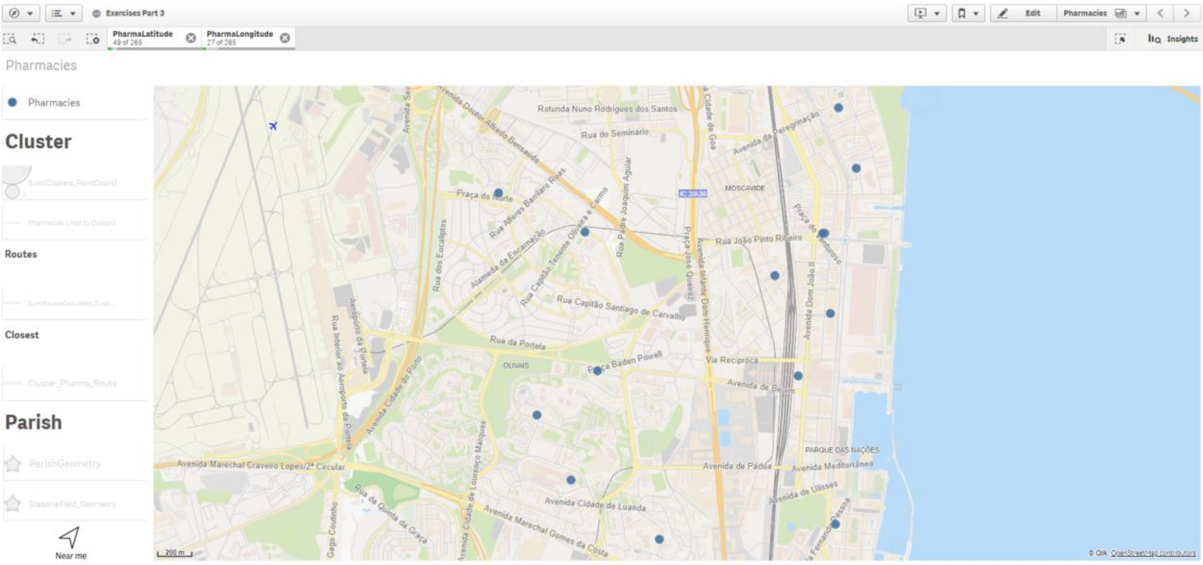


On the extension's properties, under the **Fields** pane, for the **Latitude Field** type *PharmaLatitude* and for the **Longitude Field** type *PharmaLongitude.*

Configure the coordinates reach radius the extension should search by going to the **Search Reach** pane and for the **Reach distance in km** type 2.



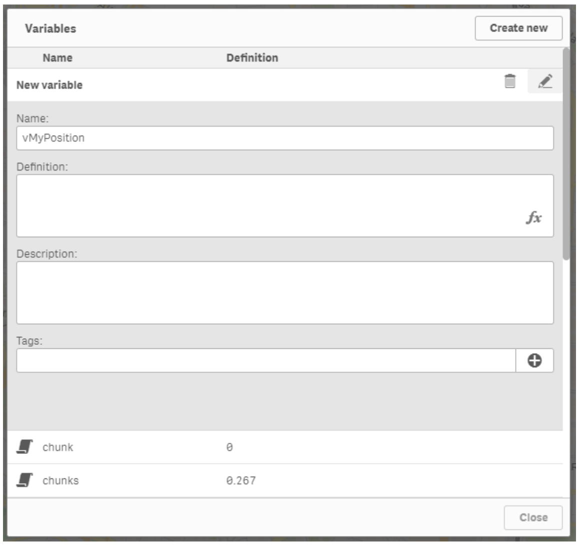
The extension is now configured. If you exit from the sheet edit mode and hit the icon, the extension will perform a search in the specified fields and select them so the map zooms to show which Pharmacies are nearby.



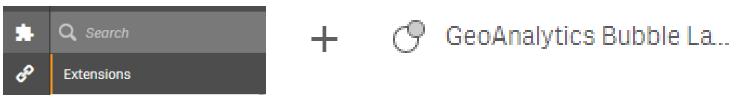
As a final extra, you can also add a new **GeoAnalytics Bubble Layers** to simulate the user's position in the map.

The extension can also update a variable that you specify. This way you can use this information to render a bubble using the **GeoAnalytics Bubble Layer.**

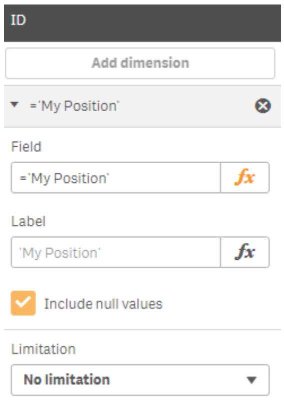
Go to your application's **Variable Editor**  and create a new variable *vMyPosition.*



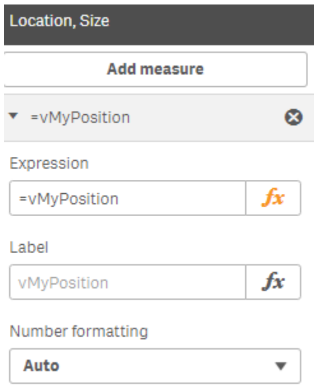
Add a first **GeoAnalytics Bubble Layer** to your sheet to display the user's received position.



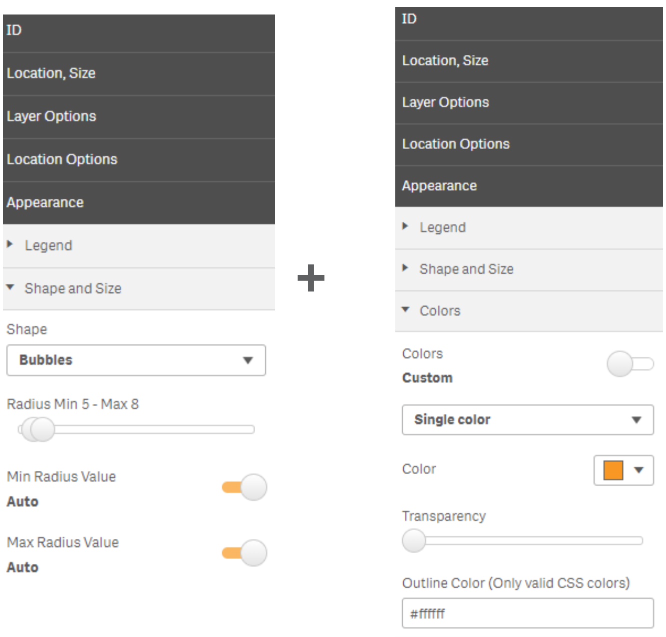
Jump to the **Properties** pane and under **ID** add text of your choosing in the **Field** identification, such as *='My Position'.*



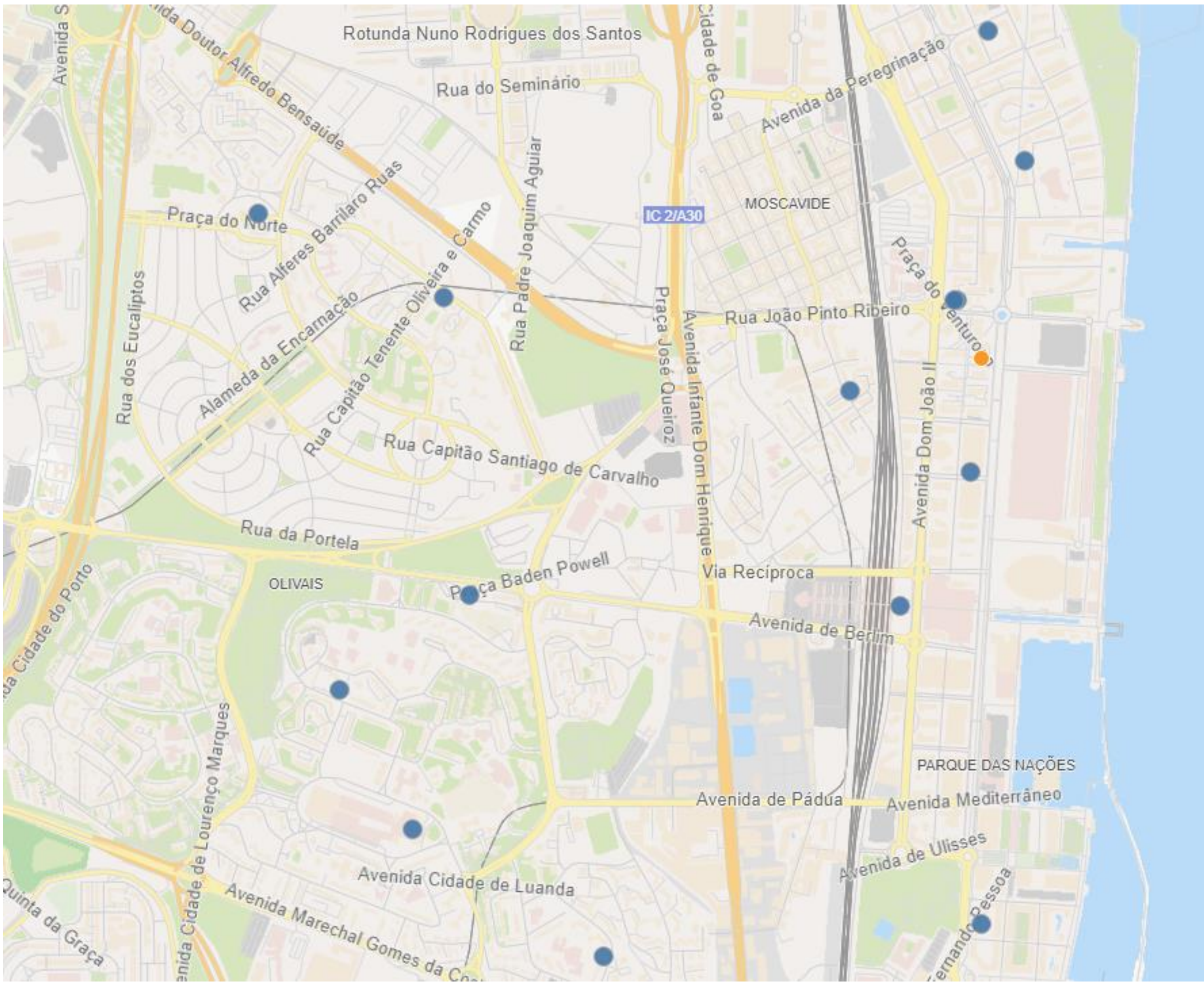
Now go to the **Location, Size** pane and this is where we configure the **GeoAnalytics Bubble Layer** to use our previously created variable. Under **Expression** type *=vMyPosition.*



Go to the **Appearance** pane, and under **Shape and Size** fine tune the bubble size to your liking. This should be the smallest center bubble representing the user's location. A value around 5 to 8 should be ok. Under the **Color** pane you can also choose a color if the default doesn't suit you. In my example I am using orange and outlining with white (#ffffff).

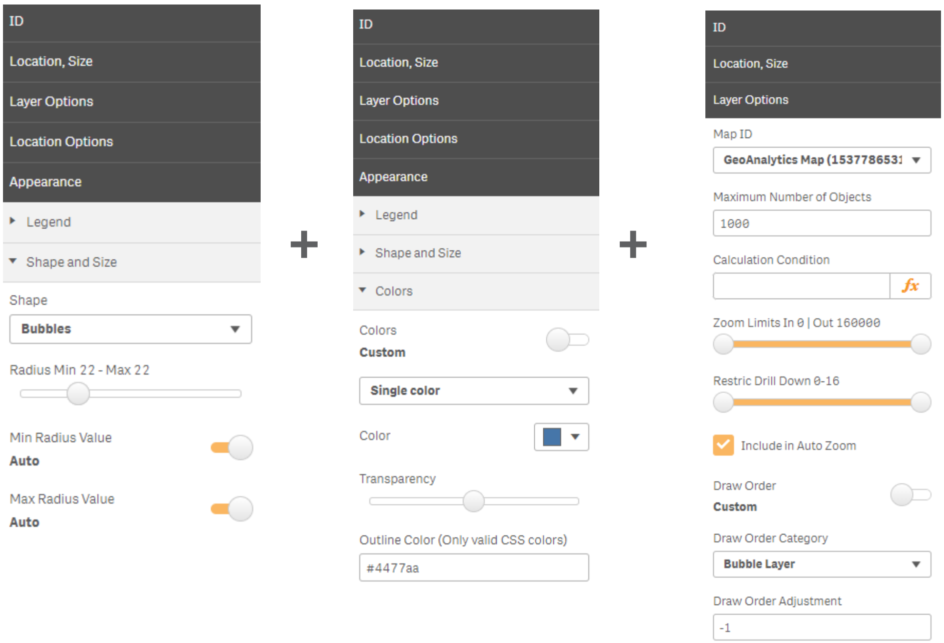


An example of my output so far:

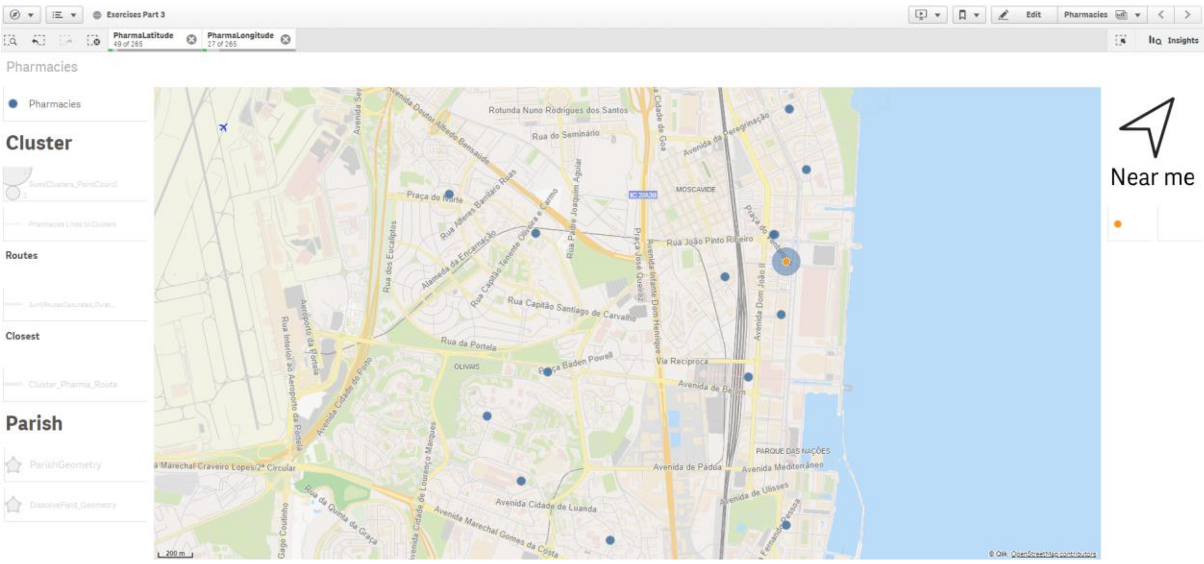


To finish the user's location representation, we can add a second **GeoAnalytics Bubble Layer** that can have a larger radius and some transparency. Copy and paste the **GeoAnalytics Bubble Layer** you've just created and go to its properties.

Under the **Appearance** pane, go to **Shape and Size** and select a larger radius value, i.e. 22. Under **Colors** (update the color if you prefer and) give some T*ransparency* to your bubble, around 50%. You can also update the outline color to match the bubble color. Also, if the bubble is over the previously created, make sure the drawing order is the expected. Go to the **Layer Options** pane and edit the **Draw Order**, for example attributing *-1* to this bubble as **Draw Order Adjustment.**



You've done it! Your user's can now search for nearby points within your Qlik Sense application. Here is an example of a possible final result:



Additional Guidelines

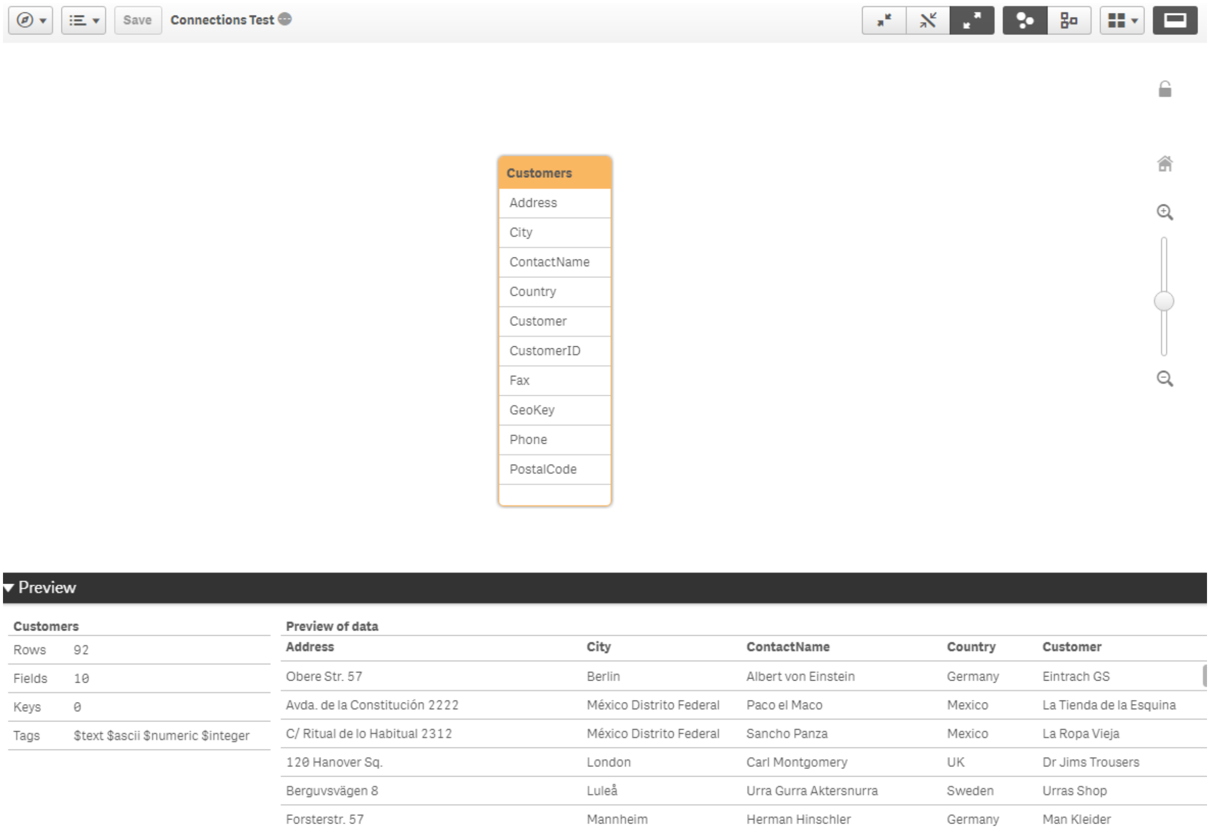
How to perform GeoCoding

This section only explains a general configuration example to perform GeoCoding: the process to retrieve Latitude and Longitude coordinates from an address.

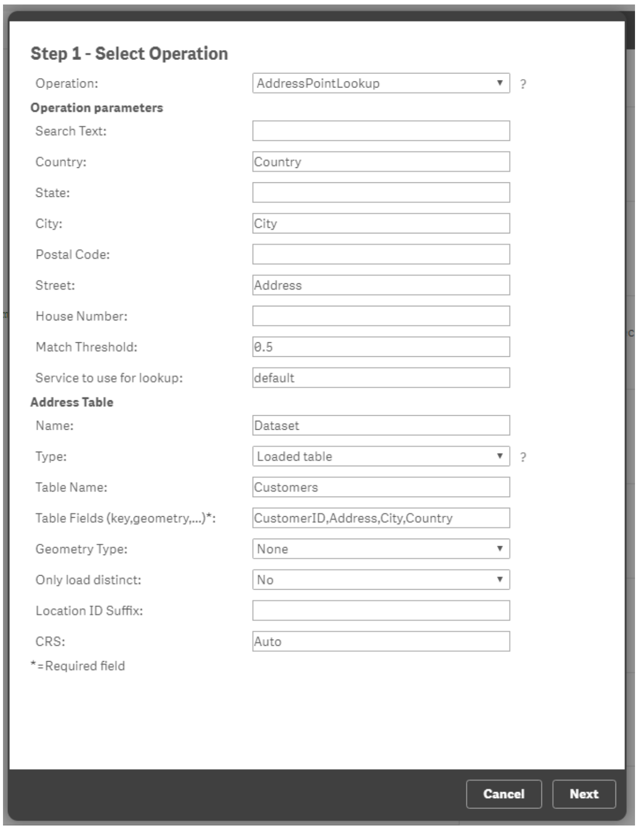
This is not an exercise covered in the session and the Qlik internal license does not include this functionality. You must explicitly request it as this is a service provided by a third party to Qlik.

The configuration is straight forward as you only have to match your data to the parameters required for the **AddressPointLookup** operation of the GeoAnalytics Connector.

In a table such as the one representing the Customers and used for the Exercises Part 1



The configuration would be:



How to perform Reverse GeoCoding

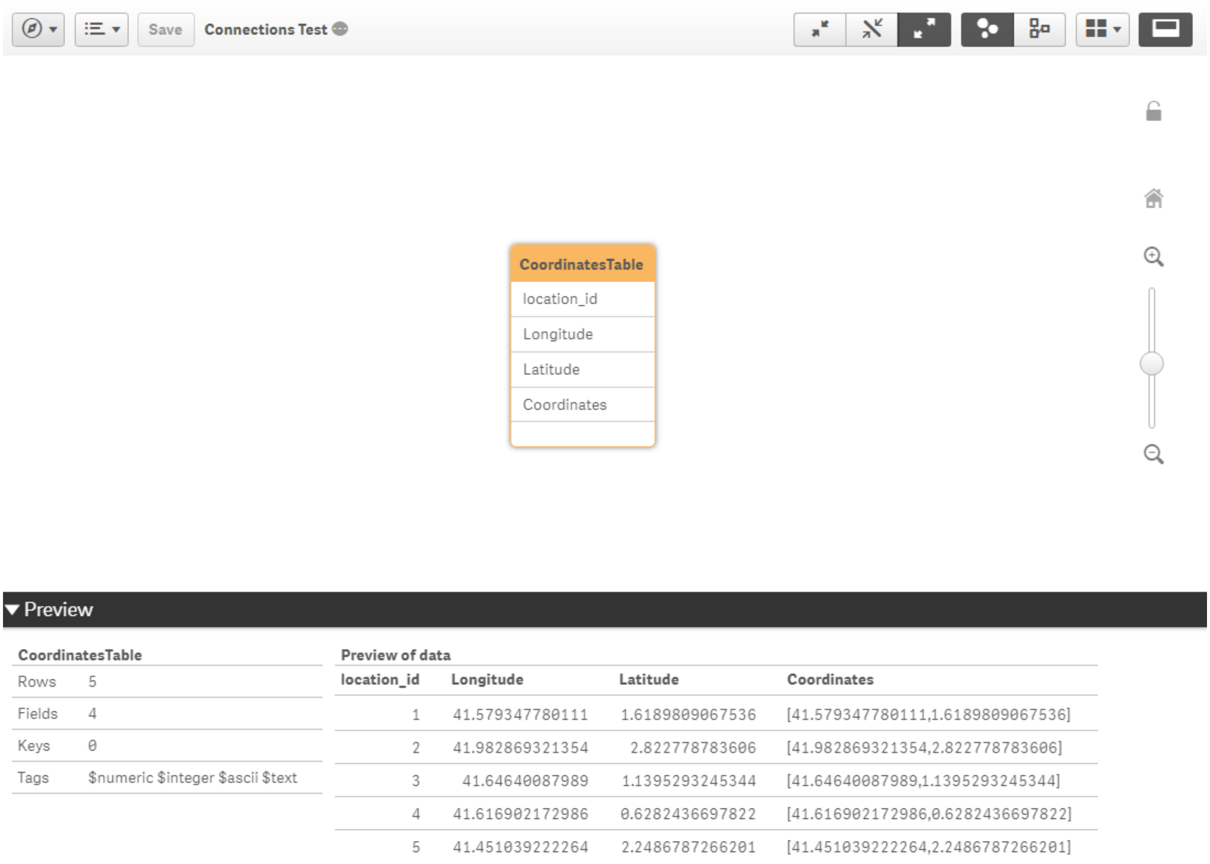
This section only explains a general configuration example to perform Reverse GeoCoding: the process to retrieve Addresses from Latitude and Longitude coordinates.

This is not an exercise covered in the session and the Qlik internal license does not include this functionality. You must explicitly request it as this is a service provided by a third party to Qlik.

The configuration is straight forward as you only have to match your data to the parameters required for the **PointToAddressPointLookup** operation of the GeoAnalytics Connector.

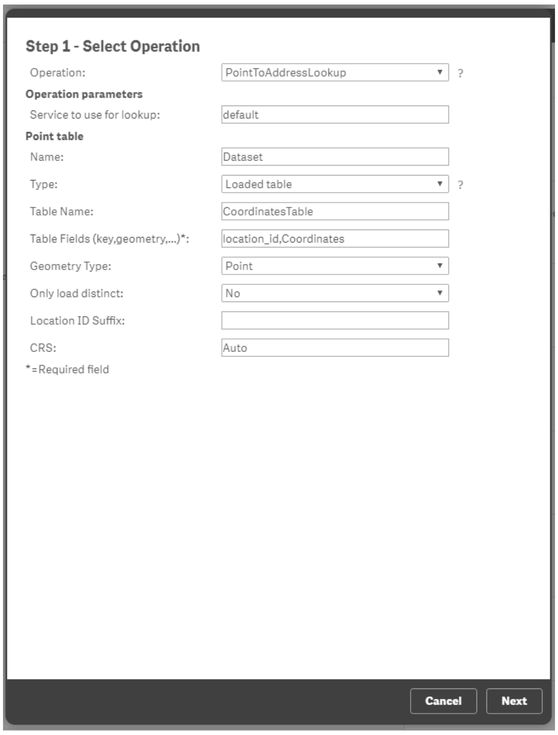
In this example I am using a table that has the following structure:

```
CoordinatesTable:
LOAD
   RowNo() as location_id,
   CoordX as Longitude,
   CoordY as Latitude,
   '['&CoordX&','&CoordY&']' as Coordinates
FROM [lib://QVDs/CoordsXY.qvd] (qvd);
```



Quick note: if you look at a map in a X,Y perspective, X is the Longitude and Y is the Latitude. Check https://en.wikipedia.org/wiki/Longitude and https://en.wikipedia.org/wiki/Latitude for more information.

The configuration for **PointToAddressPointLookup** would be:



Enablement Materials

You can access to Qlik GeoAnalytics full documentation on Qlik's help site:

- https://help.qlik.com/en-US/geoanalytics/Content/Home.htm

For more Qlik Sense examples check the Qlik Sense Guides & Examples micro site:

- https://bi.idevio.com/products/idevio-maps-for-qlik-sense/qlik-sense-2

For QlikView examples check the IM5QV Guides and Examples micro site:

- https://bi.idevio.com/products/idevio-maps-5-for-qlikview/guides-and-examples