## Configuration of Aggregation Index

Author: Andreas Wombacher

This document describes the configuration of the geoserver for the demo in cooperation with Arcadis.

### Requirements

The system requirements are listed below:

* PostGIS or MonetDB (with GIS compatibility)
* GeoServer
* Aggregation Index data store for geoserver  
  <https://github.com/utwente-db/neogeo/tree/master/geoserver-ext>

In this document I refer a dataset we have collected containing approx 7 million geo-referenced tweets of the UK in table *uk\_neogeo*. Further, there is a table for the aggregation index named *uk\_neogeo\_pa*. Both tables are in the *public* schema and the *twitter* database. Furthermore both tables are accessible for PostGIS user *geoserver*.

To test the setup of the geoserver, I provide a little demo application which is contained in

<https://github.com/utwente-db/neogeo/tree/master/pre-aggregate-demo>

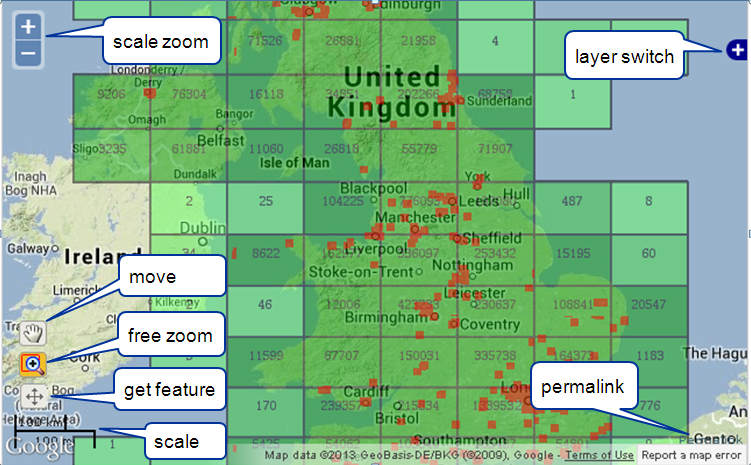
This demo has to be configured and then deployed as a war file in the same server as the geoserver is installed. Otherwise the JavaScript is not able to retrieve the GetFeatureRequests. The web application is then available at

<http://localhost:8080/aggregate-demo/>

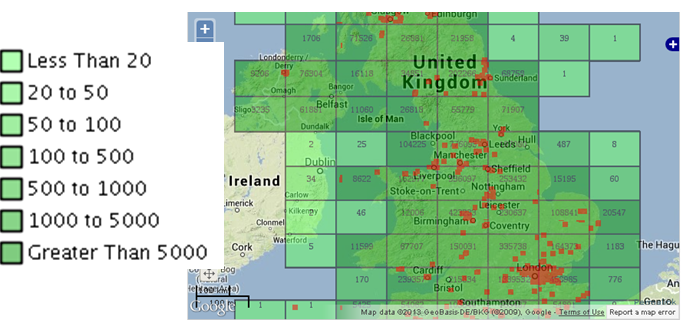
### How does it look like?

The basic idea is that often if you want to visualize a lot of information in a GIS the layers get very slow and overly crowded. We suggest to have in addition to the layer providing the features for each individual object another layer where we present only aggregates. Thus, we put a grid over the map with a configurable amount of cells and we display e.g. the number of objects in a particular cell instead of all objects of that cell. The idea is that the user can first get an idea about the distribution of the data in the current view of the map and zoom in on the parts which are most interesting. Once the number of objects is sufficiently small you switch the detailed layer on again to investigate the individual objects.

The figure below shows a screen shot of the demo application explaining the different control elements in the application. The numbers in the grid cells represents the number of tweets available in the grid cell. These values are translated into a color code to visualize the count value in the map. The red dots represent a second layer showing a subset of all tweets – using the WMS maxFeature option. This way the application has a response time which allows actually browsing the data without getting frustrated.



The scale of the color coding is displayed in the figure below. We think that the color coding gives you a faster impression where are hot spots in the map, although there is room for improvement on the color scheme.



### Configuration aggregation index extension

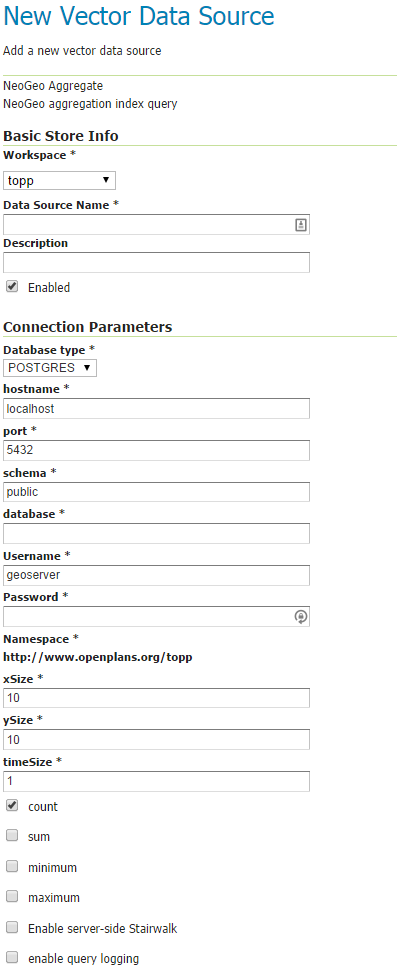
After the extension is installed in geoserver – see installation guide for details – the aggregation index can be configured via the geoserver admin interface at

<http://localhost:8080/geoserver/web/>

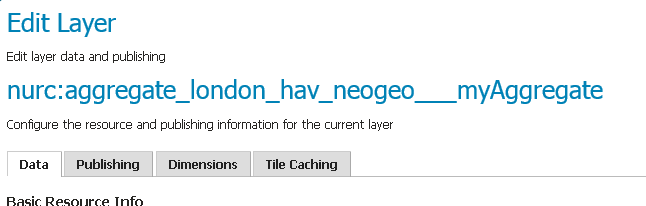
After logging in as admin, you can create a new data store choosing the option “add stores”.

You will see something like the screen shot on the right hand side. Choose the option “NeoGeo Aggregate”. If you do not see the option then the aggregation index geoserver extension has not been installed properly.

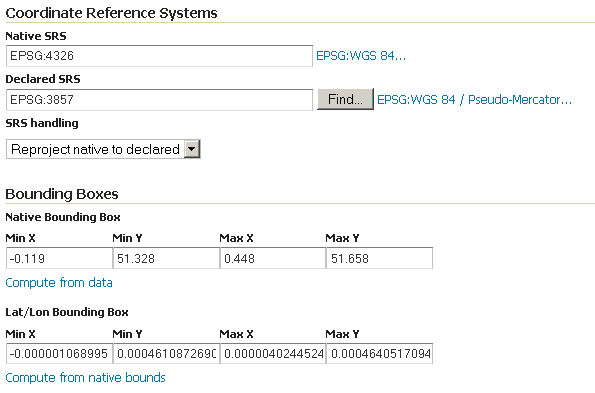
Next you will see a screen similar to the screen shot on the right hand side. Please specify the mandatory fields.

* Data Source Name: just a name you assign to the store
* Database type: select the type of underlying database, either PostgreSQL or MonetDB
* Hostname: hostname of the database server where the aggregation index is maintained. Please realize that the creation of the aggregation index is outside the scope of this document.
* Port: database port
* Schema: name of the schema where the aggregation index is maintained
* database: name of the database where the aggregation index is maintained
* username and password of the used database server
* xSize, ySize, timeSize: specifies the dimensions of the grid which is created for every view of the map to calculate the aggregates per cell. The higher the number of cells the more detailed the information – be careful not to have too much information. However, the performance decreases in case of high resolution.
* Count, sum, minimum, maximum: tick the boxes for the aggregates you are going to use in your application. Please note that from these basic aggregates much more aggregates like e.g. mean can be derived, by dividing the sum by the count.
* Enable server-side Stairwalk: by ticking this box the data source will rely on the use of the database plugins to use the PreAggregate index. For performance reasons it is highly recommended to use this option and install the database plugins.
* Enable query logging: ticking this box will turn on the logging of all PreAggregate queries into a separate table called pre\_aggregate\_logging. This table will be automatically created.

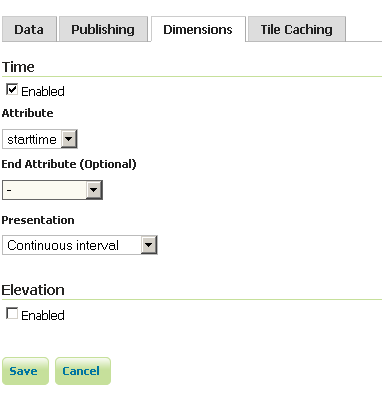
After creating the store, a layer is created automatically. Looking similar to the screen shot below.



In the Data tab of the Edit Layer screen several actions have to be performed as indicated by the red boxes in the figure below. Specify the coordinate system of the application as EPSG:3857 since this is what is usually used for tile based map representations. Further state that the coordinate system translation is done in geoserver. Finally hit the “Compute from native bounds” to determine the bounding boxes.



Next hit the Dimensions tab on the page to do some further adjustments assuming that you aggregation index has a time dimension.

Select the attribute handling the time, which is “starttime”. Please do not specify the end attribute. Select as Presentation the Continuous interval.  
Then save the data.