Margin notes version!!

Stairwalker Manual

We could write something here

Dennis Muller
Jochem Elsinga

October 13, 2014
Universiteit Twente $\boxed{\mathcal{UT}}$

Contents

1 Introduction **3**

2 Set Up **4**

2.1 Database Setup 4

2.1.1 PostgreSQL Database Manager Setup 4

2.1.2 PostgreSQL Configuration 4

2.1.3 PostGIS Configuration in PostgreSQL 5

2.1.4 Serverside Stairwalker Extension in PostgreSQL 6

2.2 Pre-Aggregate Database Table 6

2.2.1 Description of Process 6

2.2.2 PreAggregate Tool 7

2.2.3 Current Status Support of Datatypes 7

2.2.4 Current Status Support of Aggregation Types 8

2.3 Tomcat Server Setup 8

2.4 Geoserver Deployment on Tomcat 9

2.4.1 Obtaining GeoServer and Aggregate Extension 9

2.4.2 Installing Extension 9

2.4.3 Deploying GeoServer 10

3 Deployment **11**

3.1 Add Source 11

3.2 Setup Style 11

3.3 Adding Layer 12

3.4 On Client (html page) Link Geoserver + Layer 13

4 Development **14**

4.1 Code Development 14

4.2 Geoserver Visualization 14

4.3 Client Side Development 14

5 Conclusion **15**

1 Introduction

This manual will explain how the Stairwalker Program of the Database section of the University of Twente works. This manual will start by explaining on how to instal all required components in order to have the basic running. This includes on how to instal all the required extensions for your database but also for other applications that are required to run this system. The second section will explain how to add all databases and different kind of datatypes to Geoserver, which is required to show in order to make the Stairwalker program run. Not only is this section going to explain how to show the data but it will give the option to customise this view and have it display data on all possibly ways. And the third section will explain how to make adjustments to the existing program and what future possibilities the system has. For example on how to add a new dimension to the aggregation tool or how to use a different aggregation type (median for example). This manual is made for a Postgres database other database programs are also possible, however this manual will not cover how to instal certain extensions on those programs.

2 Set Up

In this section a detailed description will be given about setting up all the required peripheral programs to use stairwalker.

2.1 Database Setup

At the moment the stairwalker program only works with the PostgreSQL[1] and MonetDB[2] database management systems. This manual will only concern itself with PostgreSQL (shortened to Postgres). Any further reference to a database implies a Postgres database.

In this section a walk through explaining the steps needed to setup PostgreSQL along with PostGIS[3] and a custom made database extension for stairwalker on an Ubuntu Linux kernel.

2.1.1 PostgreSQL Database Manager Setup

Installing Postgres on Linux should be straight forward. In the terminal search for postgresql and then choose which version of postgresql should be installed. Below the commands used to search and install version 9.1 of Postgres are shown.

1. `aptitude search postgresql`
2. `apt-get install postgresql-9.1`

For further information or if there are difficulties with the installation more details can be found on the installation page of the PostgreSQL website.

2.1.2 PostgreSQL Configuration

Once Postgres is installed on Linux two alterations will need to be made to the configuration files so that the database can be accessed from outside and by other users.

First off in the `postgresql.conf` file the listen_addresses need to be changed from localhost to all. This can be done as follows (assuming version 9.1 of postgres).

1. `cd /etc/postgresql/9.1/main`
2. `vi postgresql.conf`
3. *change* listen_addresses = 'localhost' $\rightarrow$ listen_addresses = '*'
4. *save and close*

[1]PostgreSQL: `http://www.postgresql.org`
[2]MonteDB: `http://www.monetdb.org`
[3]PostGIS: `http://www.postgis.net`

Secondly in the file `pg_hba.conf` a line needs to be added to allow other users in Linux to access postgres. This can be done as follows:

5. `vi pg_hba.conf`
6. *Add* host all all 0.0.0.0/0 password
7. *save and close*
8. `/etc/init.d/postgresql restart`

It should now be possible to create a database user and database in postgres. More information about how this is done can be found on the Postgres manuals webpage.

2.1.3 PostGIS Configuration in PostgreSQL

The next step is to extend Postgres with PostGIS database expansion. This again should be straight forward, first search for PostGIS and then choice the version that goes with Postgres to install. Note in order to do the install root privileges are required. Below the commands to search and install PostGIS for Postgres-9.1 are shown.

1. `aptitude search postgis`
2. `apt-get install postgresql-9.1-postgis`

More information about installing PostGIS can be found on the PostGIS website.

Once PostGIS is installed some extra configuration is required. This needs to be done as postgres user. The commands are as follows.

Why do we do this again? What does it do?

1. `createlang plpgsql geonames`
2. `psql -f ‘find/usr/share/postgresql/ -name postgis.sql -print’ -d geonames`
3. `psql -f ‘find/usr/share/postgresql/ -name spatial_ref_sys.sql -print’ -d geonames`
4. `psql -f ‘find/usr/share/postgresql/ -name postgis_comments.sql -print’ -d geonames`

2.1.4 Serverside Stairwalker Extension in PostgreSQL

Finally there is C extension which allows processing of the SQL queries for the pre-aggregated database to happen on the server side instead of in the GeoServer application. This extension has to be installed on the Postgres database. The extension can be found in the directory `neogeo/pre-aggregate/src/db-extensions/postgres/pa_grid`.

The extension can be installed on Linux using the following commands.

1. *go to the pa_grid directory in the terminal*
2. `make`
   this creates the dynamic library
3. `make install`
   this installs the library in the postgres installation
4. `make sql`
   declare the module in the database wanted

Note the extension has to be installed specifically for the database which will be used for pre-aggregation. In the makefile (also in the `pa_grid` folder) there is a `DATABASE` macro which should be set to the desired database.

Installing the extension also requires root access. Why was this again?

2.2 Pre-Aggregate Database Table

2.2.1 Description of Process

Stairwalker makes use of the Pre-Aggregate functions. These functions create a granularity in the dataset and calculate blocks at the lowest granularity for certain dimensions. Then subsequent blocks of higher granularity are built up from lower blocks. This creates a new dataset which as summarized the original dataset in terms of blocks which represent some significant data for every level of granularity and dimension.

Concretely take the dataset of tweets sent within the netherlands. A pre-aggregation of this data could be in the following form. As significant data we consider the number of tweets in the x- and y-coordinate dimensions. We then set a highest granularity (zoom). The pre-aggregate algorithm then creates blocks bounded by the x- and y-coordinates at the highest granularity and for all those blocks calculates the number of tweets within those coordinates. Then each subsequent layer is built up from the previous layer. The final result is a new dataset which contains all blocks with the number of tweets in a coordinate block at each level of granularity.

2.2.2 PreAggregate Tool

To do the Pre-Aggregation step a tool has been develped which is explained here.
The tool can be found in the directory `neogeo/pre-aggregate-tools/`. To generate the binary tools from source the Appassembler plugin of Maven is used. Run the following command to generate the tool.

```
mvn package appassembler:assemble
```

After successful completion of this command a new directory `appassembler` will have been created in the `target` directory containing a `repo` and a `bin` directory. The `bin` directory contains the actual binaries of the tool (in both Linux/Unix and Windows version) and the `repo` directory contains the tool dependencies. The tool should now be ready for use.

The tool is used to create a PreAggregate index for a table with n dimensions and a measure/aggregate column. It uses with the following commands:

```
usage: create-pa-index
 -axistosplit <axis index>    index of axis to split
 -chunksize <size>            maximum chunk size after split
 -config <file>               PreAggregate XML config file
 -d,--database <dbname>       name of database
 -dbtype <postgresql|monetdb> type of database
 -h,--host <host>             database host name or ip address
 -help                        prints this help message
 -p,--port <port>             port number of the database
 -password <password>         database password
 -s,--schema <schema>         schema name in the database
 -u,--user <user>             database username
 -v,--verbose                 Enable verbose output logging
```

The tool is dependent on the `PreAggregate.XML` config file which is used to define the PreAggregate index by specifying the column to aggregate, the type of aggregate that should done and the dimensions to include. In the `neogeo/pre-aggregate-tools/` directory a sample configuration file is included.

How does this tool work nominal axis? More preprocessing may be required when first parsing words? Example london_words or gender_words

see Section 4.1 for more detail about preaggregation not using the special help tool.

Apart from creating a new PreAggregate table `<original-table-name>_pa` pre-aggregation also creates/updates two other tables, these are `pre_aggregate` and `pre_aggregate_axis`. These two tables are support tables for the aggregation. They contain information about which tables have been aggregated and which axis have been used in pre-aggregation.

2.2.3 Current Status Support of Datatypes

1. Aggregate Axis

2. Metric Axis
3. Nominal Axis

2.2.4 Current Status Support of Aggregation Types

At the moment you can aggregate your data in a total of 4 options and all 4 options have a different outcome.

The options that you can use at the moment are the following:

1. ALL: ???
2. Count: returns the total amount of data in an aggregate box. Returns a number with the total amount of that tile.
3. Sum: returns the total value of all data added up to eachother. For instance when you do a sum on the tweet length you'll get the total amount of characters tweeted in a tile.
4. Min: returns the lowest value of all data that is aggregated. With the example of tweet length, it will returns the value of the lowest length tweet in that tile.
5. Max: returns the highest value of all data that is aggregated. With the example of tweet length, it will returns the value of the highest length tweet in that tile.

It is important to chose the right type in order to get a good representation of your data. If you want to show the total amount in a tile, you should chose count as this will returns the total number of tweets in a tile. Whereas if you want only the highest value of your data (for example the highest building in an area) then it is important to use max. It is also possible to add other datatypes than the onces mentioned here. To do so check the section about development, here you will find more information about how to add a different aggregation type.

2.3 Tomcat Server Setup

To visually display the aggregated dataset a third party application is used. This application is called GeoServer, GeoServer is an open source server which can be used to display geospatial data. In order to run GeoServer a web server is needed.

During development the Apache Tomcat was used for this purpose. The use of the Tomcat is not required for stairwalker alternative options may also be used as long as it is possible to deploy WAR files. Because Tomcat was used during development a description of how to obtain Tomcat is given here. Furthermore whenever web hosting is mentioned in this manual it will be assumed Tomcat is being used.

Information about setting up a Tomcat server can be found on the Apache Tomcat website.

2.4 Geoserver Deployment on Tomcat

2.4.1 Obtaining GeoServer and Aggregate Extension

The GeoServer[4] is used to visually display the aggregated data. To use the aggregated dataset an extension has be written for GeoServer which needs to be included in the web application. This is done by including the JAR files of the extension in the WEB-INF of the GeoServer WAR file. A step by step guild is given below in Section 2.4.2. However before following these instructions the necessary JAR and WAR files will need to be obtained.

The GeoServer WAR file can be downloaded from the GeoServer website. The custom Java extension files should be built using the source code. The extension consists of the `pre-aggregate` and `geoserver-ext` projects in the `neogeo` project. Note that first the JAR file of `pre-aggregate` should be created as it is a dependency of `geoserver-ext`. The JARs can be built using the command line or in a IDE such as Eclipse or Netbeans, using the `clean and build` command on a project in Netbeans should build `<project-name>-0.0.1-SNAPSHOT.jar` which can be found in the `target` directory of the project.

2.4.2 Installing Extension

These next instructions assume the geoserver extention files are located in the directory `/data/upload/` and that the `geoserver.war` file is located in the directory `/data/tmp/tmp_war`. If this is not the case the file paths in the instructions below should be changed accordingly.

Unpack the WAR file

1. `jar -xvf geoserver.war`

Copy the JAR files of the extension into `WEB-INF/lib` directory of the GeoServer unpacked WAR file

2. `cp /data/upload/pre-aggregate-0.0.1-SNAPSHOT.jar /data/tmp/tmp_war /WEB-INF/lib`

3. `cp /data/upload/geoserver-ext-0.0.1-SNAPSHOT.jar /data/tmp/tmp_war /WEB-INF/lib`

Recreate the GeoServer WAR file

4. `jar -cvf geoserver.war META-INF/ WEB-INF/ index.html data/`

The GeoServer WAR with the stairwalker extension should now be ready for deployment.

[4] GeoServer: `http://www.geoserver.org`

2.4.3 Deploying GeoServer

After the GeoServer WAR file has been repacked with the aggregate extension included it is ready to be deployed in a web server. Section 2.3 describes how to set up such a Tomcat web server. Once the server is running it can be accessed locally with `http://localhost:8080/` assuming default installation configuration were used, otherwise the port number might be different. In the case that the Tomcat server is installed on a difference machine then the web server can be accessed by replacing `localhost` with the IP address of that machine.

From the Tomcat homepage it should be possible to access the Tomcat manager webapp. With the default setup it should be possible to login with the following credentials:

username: manager
password: tcmanager

In the Tomcat manager webapp under the `deploy` section it is possible to upload a WAR file to be deployed. Select the repacked WAR file from Section 2.4.2 and deploy the application. Once the application is deployed it will be displayed in the `application` section of the Tomcat manager webapp. From there it is possible to follow the path given for the GeoServer application or, if the default configuration was used to go to `http://<Tomcat-IPaddress>:8080/geoserver/web/`.

3 Deployment

This section will give a detailed description of how to import an aggregated database table into GeoServer to get a visual representation of the dataset. First instructions will be given on how to link the database table to GeoServer. Next creating styles and layers for data representation will be discussed. The final sub section will discuss how to view the data using GeoServer. For this section it is assumed that all the initial preparation discussed in Section 2 has been completed.
newline
GeoServer should already be deployed on a web server (see Section 2.4.3), and can then be accessed with `http://<Tomcat-IPaddress>:8080/geoserver/web/`. It is required to login in order to work on the server side of GeoServer. When using the default setup of GeoServer the login credentials are:
username: admin
password: geoserver

3.1 Add Source

Once the webapplication has logged in, then start by adding the source of the database.

1. On the left side there should be a menu with all kinds of options, but in order to add a source, go to Configuration → Sources.
2. At the top there should be an add source button.
3. Now under vectortypes there should be a list of types. Select what kind of source you want to add. In order to get the Stairwalker program running there should be a NeoGeo Aggregate - NeoGeo aggregation index query. If this is not the case something went wrong with deploying the geoserver and the last steps in this manual should be done again.
4. If it is here, click on it and a new menu will open. A new screen should open where all information for your source has to be filled in. There is also an option on what aggregation type the program should use, this should have been defined before so use what has been used before (count/sum/minimum/maximum).
5. Once everything is filled in all, the source should have been added to the Geoserver. This can be checked by going to sources again and then it should display the source that has just been added.

3.2 Setup Style

The next step is to make a style in which you want your data to be shown. A style has to be made in SLD/XML, there are a lot of options which can be used to make a style. First navigate to the style tab, Configuration -¿ Style here you can add a new style with the button on top. Here you need to add your code (or file) for your style.

http://docs.geoserver.org/2.5.x/en/user/styling/sld-cookbook/index.html contains a lot of information regarding styles, it is a lot of trial and error t oget a nice working style. The next few steps however should give a short setup on how to start making your own style.

1. Decide what data should be shown, in our example were using the count of the aggregated data.
2. Decide what kind of representation should be used, you can either use Polygonsymbolizer, pointsymbolizer or linesymbolizer. All of these have their own advantages and disadvantages, incase it isn't clear what to use, start off with a Polygonsymbolizer.
3. Add all filters, for instance if the data should how a lighter color if the amount is lower. Or make a tile red once the data exceed a certain maximum (for instance if there were too many orders in a timerange).
4. Add colors to the layers representing it better and giving it a nice layout.
5. Make sure the positioning of the label and all filters are working.
6. Finish the layer by adding the last details (Halo behind the text for example).

This is just a basic manual on how to add a style, for more information or more explanation on how to make a style, check out the development chapter in this manual. In that section there are far more details on how to make a style.

3.3 Adding Layer

Now we have a source and have a style to use, it is time to setup a layer to display this style.

1. Go to Configuration → Layers and press the button on top, A dropdown menu should appear in where the database has to be selected. Once a database has been selected there should be a couple of options of which data should be displayed.
2. Select the data you want and press Publish. Fill in all these fields in order to get the layer to work. Important are the projections. In our layers we use Projection of Source, EPSG:4326, and Given Projection, EPSG:3587. And set handling of the projection to use the given projection. Then for Boundary Rectangles let them be calculated based on sourceprojection. And then all fields should be filled in, do not press save yet however.

3. On top select publish. Select the style which should display your data, select the style that has been made in the last step here. If there is no style that can be displayed, something went wrong and the last steps should be repeated.

4. Once this is done press save and the layer should be added.

3.4 On Client (html page) Link Geoserver + Layer

Now in order to see the layer go to view, here it should display the layer that has been made (with the added name) with a dropdown menu. Select PNG/JPEG or something similar to show a basecase of the data. A new window should open and here it should show your data in a weird graph. If this is not the case, its likely a mistake in the style. It is also possible to adjust your view in the images by adding a VIEWPARAM to the url. To do so just add: &VIEWPARAMS=<Type>:<Value> to the url and the image should change.

4 Development

4.1 Code Development

4.2 Geoserver Visualization

4.3 Client Side Development

5 Conclusion