The Desktop GIS OpenJUMP: A hands-on introduction

for OpenJUMP 1.6.x

Stefan Steiniger and Michaël Michaud

this is an updated version of the
OGRS 2009 Workshop tutorial for OpenJUMP 1.3

Version: 30 May 2013

Table of Contents

Acknowledgments - The tutorial is based on original workshop material by **E. Bocher** and **A. Lepetit** prepared for the FOSS4G 2006 conference in Lausanne.

Introduction

Summary

The aim of this document is to introduce the free GIS software OpenJUMP and to show how OpenJUMP can be used to visualise, create, edit and analyse geographic data. With those capabilities OpenJUMP is a full functional Desktop GIS that can be used in several places of a Spatial Data Infrastructure (SDI). Figure 1 illustrates the use of desktop GIS in a SDI context.

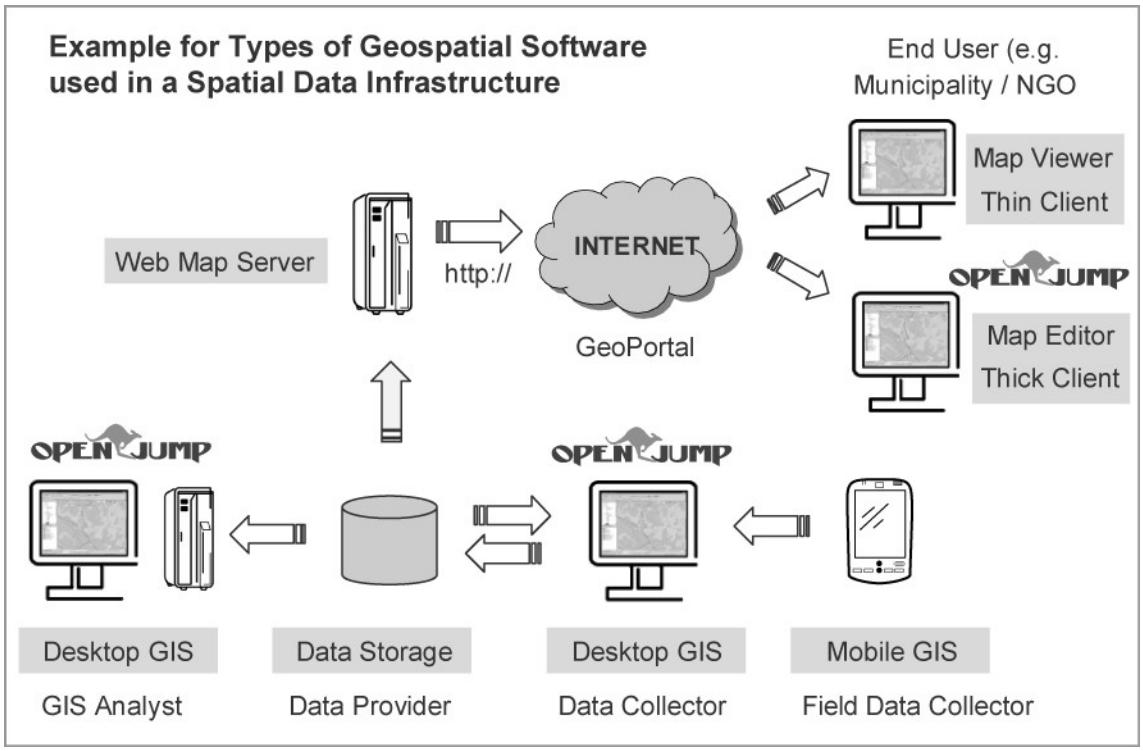


Figure 1 - Using OpenJUMP in a Spatial Data Infrastructure (SDI)

This document is organised in two parts and allows to get to know OpenJUMP step-by-step.

1. Using OpenJUMP

The first part guides you through some basic OpenJUMP functions. You will learn to:

- load and display data,
- work with alphanumeric data,
- create thematic maps,
- export views,
- print,
- query and analyse data, and
- create and edit geometry data.

2. Using OpenJUMP with PostGIS

This part will demonstrate how it's possible to write, access, and query data stored in the PostGIS database. You will learn how to:

- store data to PostGIS,
- read data from PostGIS,
- query the data with spatial and numeric predicates, and
- build expressions with more than one criteria.

Requirements

You should know the basic GIS concepts. For instance you should know the difference between raster and vector data. You should know basic GIS analysis functions, such as buffering and overlay (intersection) operations. You should know the principles of cartographic representation and classification.

Working fully through the tutorial will require about 2 hours. Trained users will need 70 mins for Part 1 and 10 min for Part 2 .

Materials

For going through the tutorial you need :

- OpenJUMP 1.6.x - available from www.openjump.org. We **recommend** downloading the **OpenJUMP PLUS** edition that includes all the functionalities used in this tutorial.
- the following OpenJUMP plugin (extension) that is contained by default in the OpenJUMP PLUS edition:
    - CadPlan's JumpPrinter extension (www.cadplan.com.au)

  If you don't have it installed, then you can download it and put the libraries (*.jar) in the OpenJUMP subfolder "/lib/ext/"
- the PostgreSQL 9.x database (pgAdmin should be included in the PostgreSQL distribution and its PostGIS extension can be installed during or after the installation process) available from www.postgresql.org and http://postgis.refractions.net
- OpenOffice 3.x or newer, available from www.openoffice.org, and
- the spatial data delivered with this tutorial (see the table below).

Data

Data	Geometry	Description
frenchprovinces	polygon	Regional country
departement22	polygon	French sub country after province
municipalities22	polygon	Municipalities for Côtes-d'Armor departement
watershed	polygon	Watershed name's Sterenn
subwaterhed	polygon	Sub watershed in Sterenn
wsmunicipalities	polygon	Municipalities crossing Sterenn watershed
waternetwork	line	Ditches and rivers in Sterenn watershed
landcover2000	polygon	Land cover classification in 2000
hedgerow	line	Hedges network in Sterenn watershed
~~F018_024.TIF~~[1]	Raster images	Topographic map (tif + world file)
~~F018_025.TIF~~[1]		Topographic map (tif + world file)
~~photo2.ecw~~[1]		Aerial photography (ecw)
~~photo3.ecw~~[1]		
pop22-5499.ods	---	Population by municipality 1954 to 1999 in Côtes-d'Armor (Bretagne), Department 22

Note - for MacOSX users the right-click mouse menu is *now* accessed via the "Ctrl" key.

1 These datasets can't be made available for copyright reasons.

I - Using OpenJUMP

1 - Loading and displaying data

1.1 - Loading and organising data

When OpenJUMP starts, you see a main window named "project" (Figure 2). A "project" is not only a window which allows data visualisation and organisation but it's a file too, which contains the path to the datasources, styles to be applied to your data, and parameters describing your OpenJUMP working environment. The file will have the extension *.jmp .

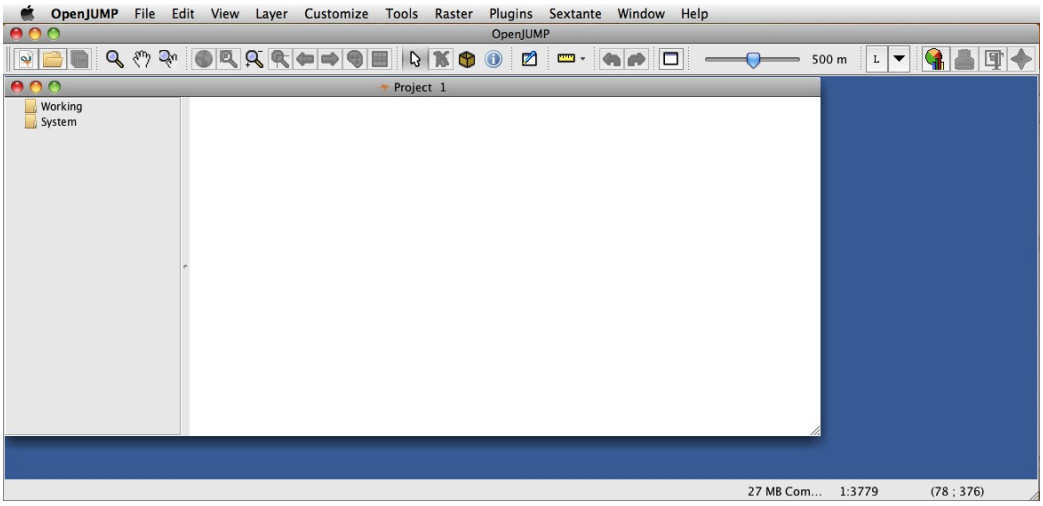


Figure 2 - OpenJUMP 1.6.2 PLUS workbench on MacOSX.

In a first step, we will load some vector data layers. To do so, go to the Menu **[File>Open...]** and make sure that the "File" icon in the left sidebar is selected. Then load:

- frenchprovinces.shp,
- departement22.shp,
- municipalities22.shp.

Another option to load vector data is to "drag & drop" files.

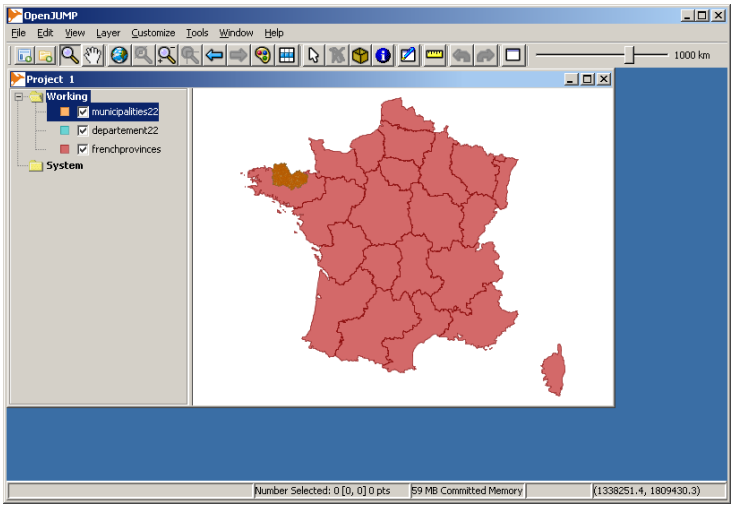


Figure 3 - OpenJUMP with the data loaded (Win).

The layers that have been loaded are listed in the *Layer List View*, to the left of the visualisation window (figure 3). We will call the map visualisation window the *Map View*.

There are several ways to move around in the *Map View* and to zoom in or out of the map area shown. Try out the navigation tools that are provided (i) in the toolbar, (ii) in the **[View]** menu; or choose a function from the mouse context menus accessible via a right-mouse-click: (iii) in the Map View (choose the zoom submenu), (iii) on a category name (e.g. "Working") or (vi) on a layer name (e.g. municipalities22).

zoom in/out,	 or use the mouse wheel
pan map view	
zoom and pan	
zoom to full extent	 or [View>Zoom To full Extend]
zoom to previous/next view	 or [View>Zoom Previous/Next]
zoom to fence	 or [View>Zoom to Fence]
zoom to selected items	 or [View>Zoom To Selected Items]
zoom to scale	[View>Zoom to Scale]
zoom to layer or layers	 in the layer context menu

1.2 - Modifying layer properties with styles

To modify the display of a layer you can use the Styles dialog (figure 4).

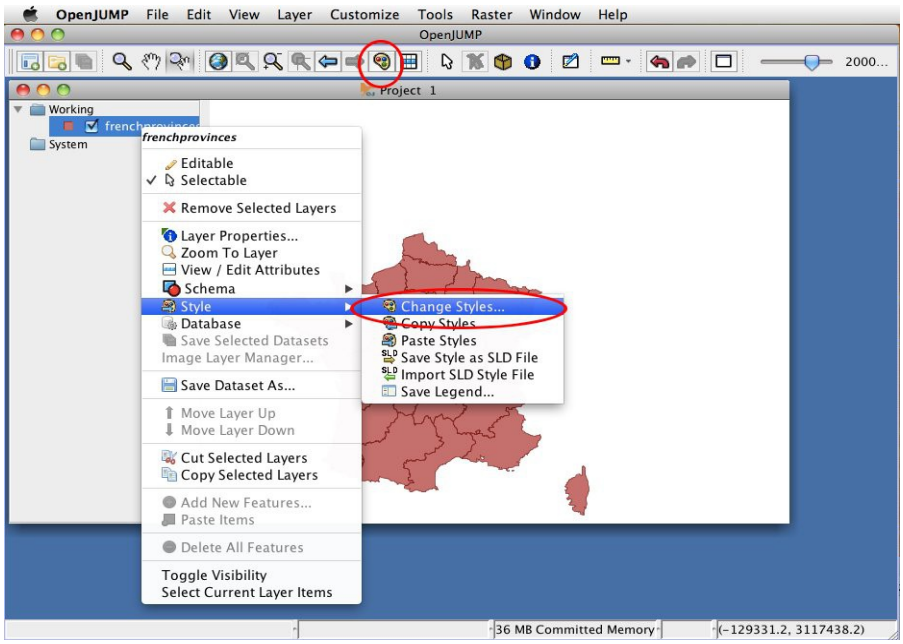


Figure 4 - Accessing the style dialog via the toolbar or the context menu.

- The **Rendering** Tab (figure 5) enables you to change general properties (colour, transparency, line width and pattern, vertices visibility, etc.).
- The **Scale** Tab (figure 6) allows you to customise the display of a layer with respect to the zoom scale. So you can define that a layer is shown only between a smallest and a largest scale. The current scale is displayed in the window left to the coordinate display

of the bottom bar.

- The **Labels** Tab (figure 7) enables you to define text labelling options (e.g., font type, colour, height, and placement options). For the settings here you need to provide the attribute/field of the dataset that contains the text to be displayed..

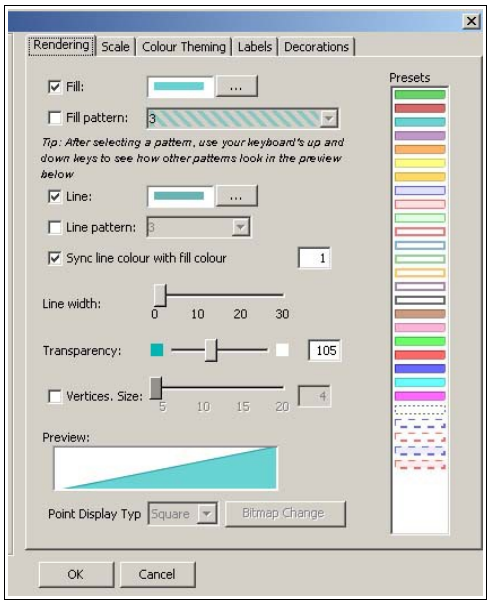


Figure 5 - Rendering dialog (Win).

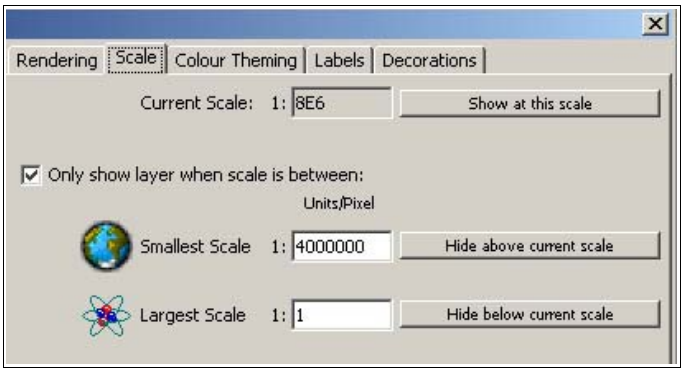


Figure 8 - Map visibility dialog.

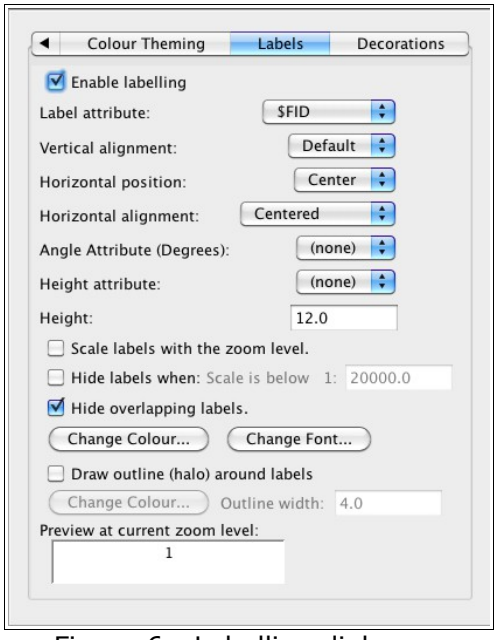


Figure 6 - Labelling dialog on MacOSX.

Exercise 1 (figure 8)

Symbolise *frenchprovinces*
- uncheck fill
- change line colour to grey with width=7

Symbolise *departement22*
- uncheck fill
- make lines black with width=2

Symbolise *municipalities22*
- use from the preset styles the yellow one
- apply labels using the attribute "name_mun";
check the box "Draw outline (halo) around labels"

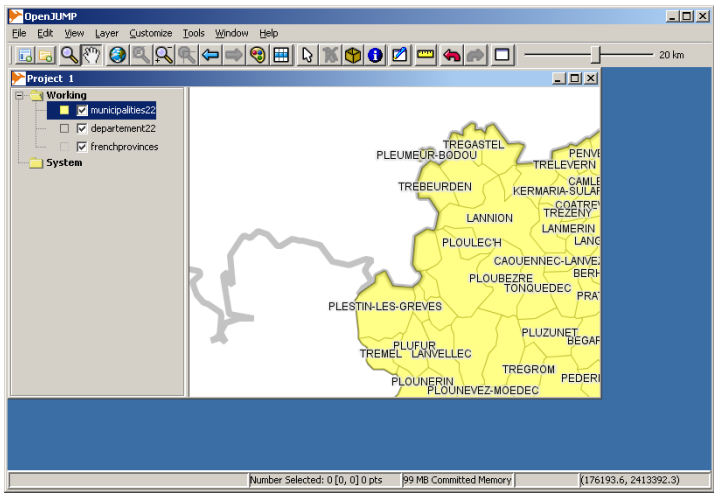


Figure 7 - Customized display of our dataset.

In the next step you should load some raster files into OpenJUMP using **[File>Open File...]** after creating a new category *Fond* in the *Layer List View*. Creating a new category allows for easier management later on. You can create (using **[File>New>Category]**), rename and move categories in the layer list view and order them as you want (figures 9, 10 and 11). Now load the following image datasets:

- photo2.ecw (note: *.ecw and jpg2000 files can only be loaded with the PLUS edition)
- photo3.ecw
- F018_024.TIF (note: images are not delivered due to copyrights)
- F018_025.TIF (note: images are not delivered due to copyrights)

If the *.TIF images are not displayed, try to load them with **[File>Open...]** and choosing the option "*Sextante Raster Image*" on the left. To load images with this option is necessary if they should be analysed with the Sextante Toolbox.

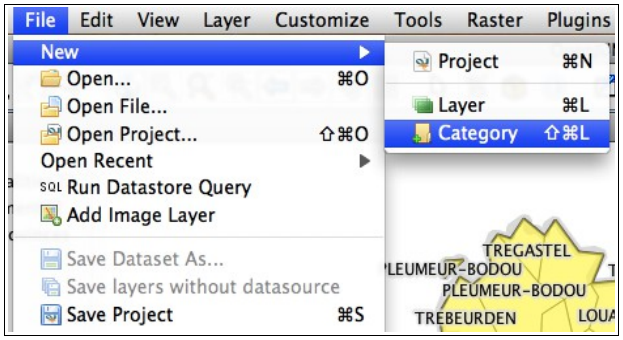


Figure 9 - Creating a new category.

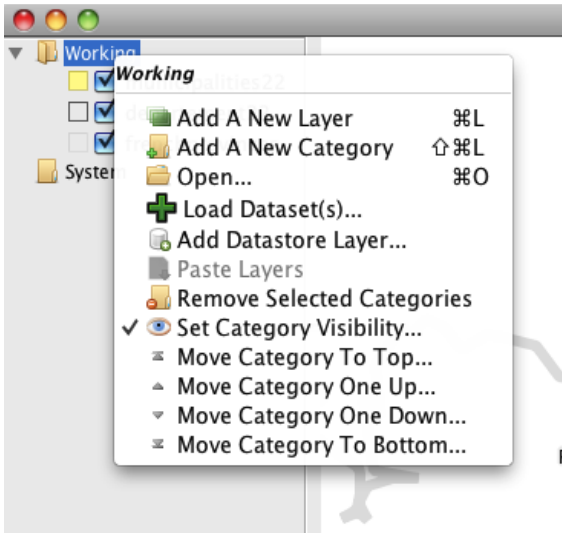


Figure 10 - Category context menu.

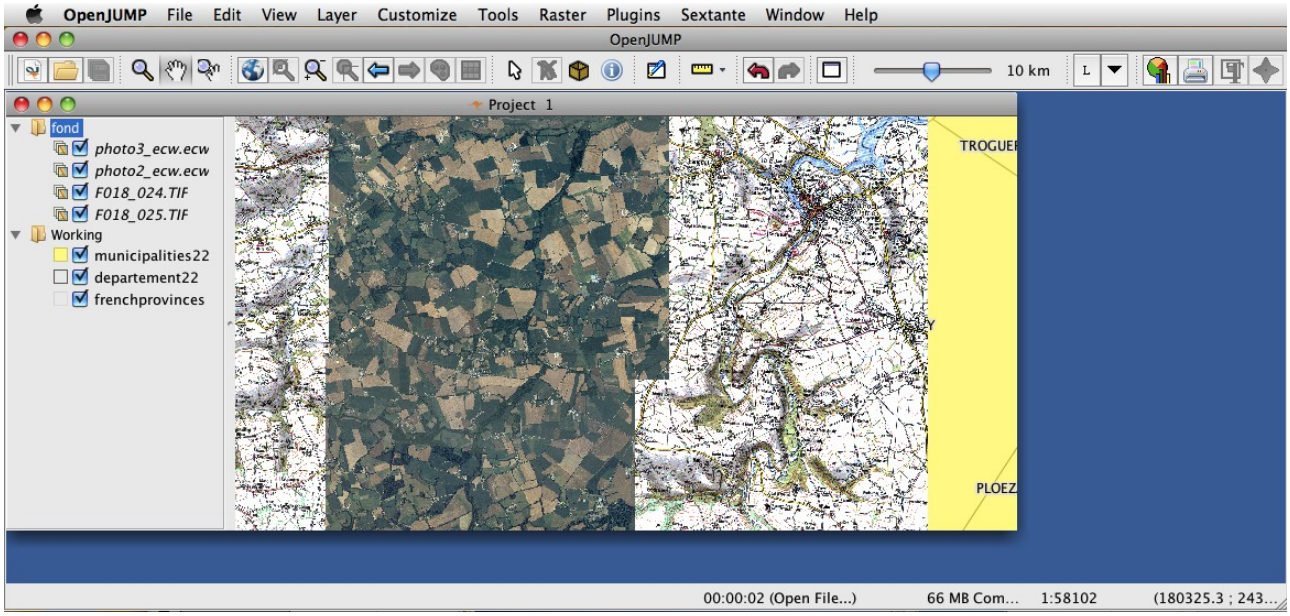


Figure 11 - Displaying vector and raster image layers combined.

Exercise 2: customise the display of the raster image for certain zoom levels

Use the Style dialogs that are accessible via the layer context mouse menu and try to:

- set for F018_024.TIF and F018_025.TIF the smallest scale = 15'000
- load photo2.ecw and photo3.ecw - set the smallest display scale to 12'000

To see the current display scale check the scale window on the bottom (left from the mouse coordinate display) or use **[View>Map Decorations>Scale Display]**.

2 - Working with alphanumeric data

2.1 - Viewing attributes

In a layer, each feature can have alphanumeric attributes. If you want to inspect the attributes of a layer, then you can use the table icon from the toolbar or choose **[View/Edit Attributes]** from the layer context menu (see figure 12).

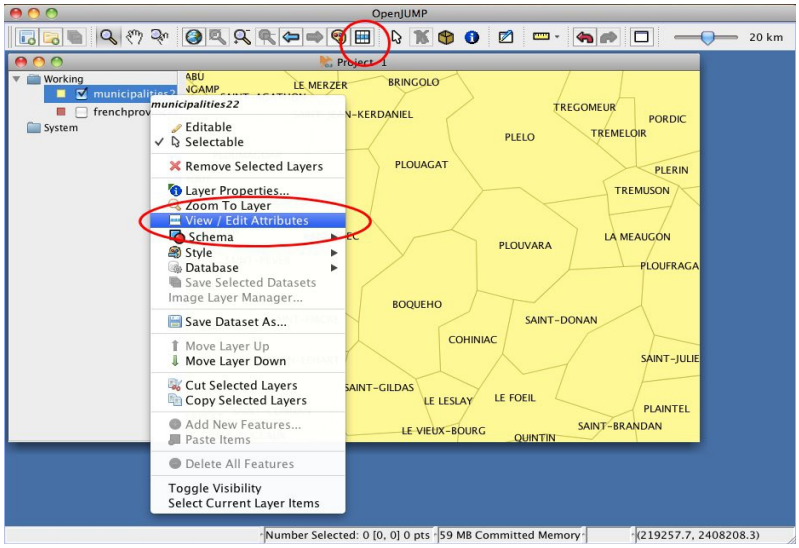
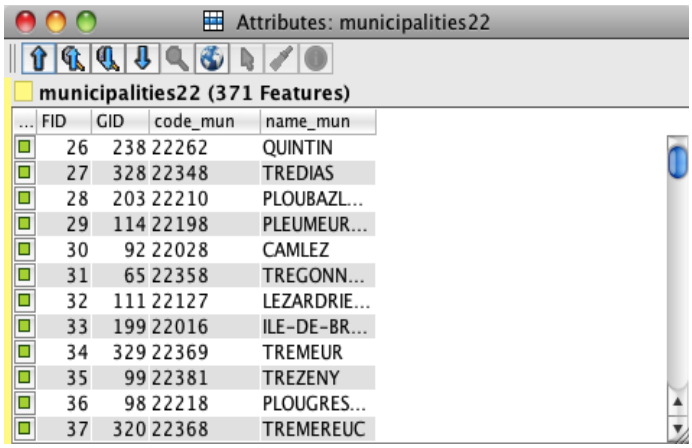


Figure 12 - Accessing the attribute view / table view.

For the *municipalities22* layer you will see a new window named "Attributes" as in figure 13.



...	FID	GID	code_mun	name_mun
	26	238	22262	QUINTIN
	27	328	22348	TREDIAS
	28	203	22210	PLOUBAZL...
	29	114	22198	PLEUMEUR...
	30	92	22028	CAMLEZ
	31	65	22358	TREGONN...
	32	111	22127	LEZARDRIE...
	33	199	22016	ILE-DE-BR...
	34	329	22369	TREMEUR
	35	99	22381	TREZENY
	36	98	22218	PLOUGRES...
	37	320	22368	TREMEREUC

Figure 13 - The attribute window.

Another way to view attributes for one or several particular objects is to use the *Feature Info Tool* from the toolbar: 

2.2 - Editing attributes of a layer

With OpenJUMP you can easily add or remove fields from a table when the layer is editable. Let us calculate the area for each municipality. Right-click on the layer *municipalities22* and check "Editable" (figure 14). If the layer is editable the layer name should be displayed in red (yellow if selected) in the Layer List View. You will also notice that a toolbox will appear, which allows to edit geometries in that layer.

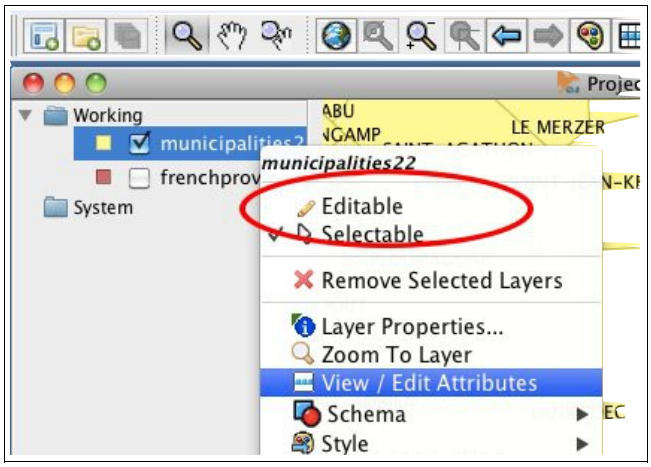


Figure 14 - Making a layer editable.

Now select from *municipalities22* layer context menu **[Schema > View/Edit Schema...]**. A new window appears that lists all attributes with their names and the data type (e.g.: String, numeric types, or Geometry – see figure 15).

Click on the lasted row to add a new field "area". Click on the corresponding data type field and chose from the dropdown list the data type "Double". Click "Apply Changes" (figure 15).

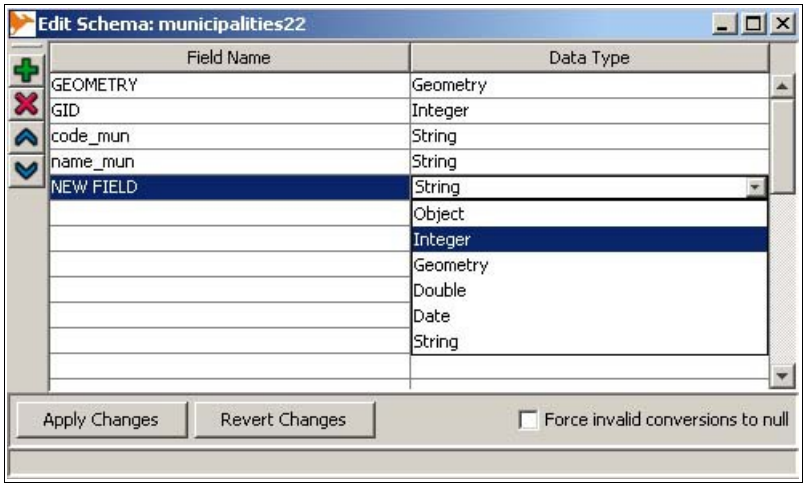


Figure 15 - Adding a new attribute to a dataset.

After this is done we can populate the attribute "area" with the values for each geometry.

To calculate the area of each municipality and write that value into the field, chose from the menu **[Tools>Edit Attributes>Calculate Areas and Lengths...]**. Select what kind of operation you want to do and specify the correct attribute (figure 16)[2].

2 This exercice has been kept to show you how to create a new attribute. However, from OpenJUMP 1.6, there is a simpler way to compute areas and length with the function [Tools > Edit Attributes > Add Geometry Attributes...]

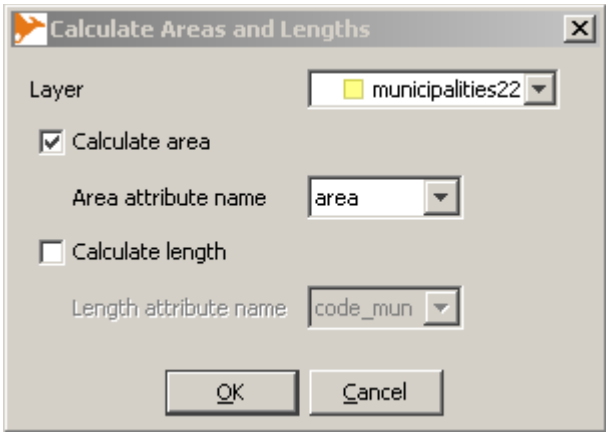


Figure 16

The results are expressed in m². For our purposes this unit is not very comfortable, so we will convert it into km². The *Attribute Calculator* lets you perform simple calculations on all records for one attribute.

- Open the calculator: **[Tools>Edit Attributes>Attribute Calculator]**[3].
- Set in "name of new attribute" field: "AREA_KM2"
- Chose type: DOUBLE
- Set the expression as in figure 17 (use the buttons "area" and ''/" and "copy value to formula" to avoid errors)
- Click "OK"

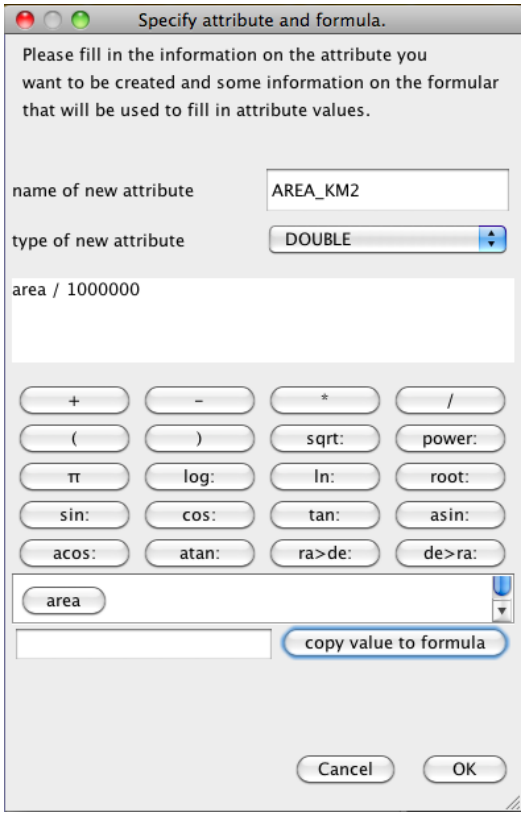
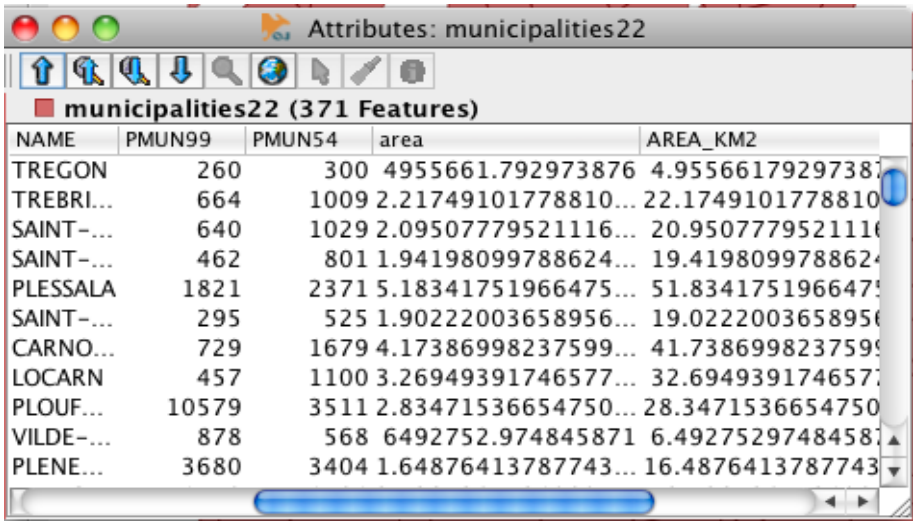


Figure 17 - the attribute calculator

This tool will create automatically a new field (AREA_KM2) in the attribute table (figure 18).

3 For more complex expressions, you can also use [Tools > Edit Attributes > Beanshell Attribute Calculators...]. Try to compute area_km2 and density with Beanshell Attribute Calculator instead of Attribute Calculator.



NAME	PMUN99	PMUN54	area	AREA_KM2
TREGON	260	300	4955661.792973876	4.95566179297387
TREBRI...	664	1009	2.21749101778810...	22.1749101778810
SAINT-...	640	1029	2.09507779521116...	20.9507779521116
SAINT-...	462	801	1.94198099788624...	19.4198099788624
PLESSALA	1821	2371	5.18341751966475...	51.8341751966475
SAINT-...	295	525	1.90222003658956...	19.0222003658956
CARNO...	729	1679	4.17386998237599...	41.7386998237599
LOCARN	457	1100	3.26949391746577...	32.6949391746577
PLOUF...	10579	3511	2.83471536654750...	28.3471536654750
VILDE-...	878	568	6492752.974845871	6.49275297484587
PLENE...	3680	3404	1.64876413787743...	16.4876413787743

Figure 18 - The new field AREA_KM2 created with the calculator.

You can save the changes to the datasets in two ways: either by using "Save Selected Datasets" from the layer context menu (figure 19), or by using "Save Dataset as ..." from the layer context menu. The latter option allows you to chose a new file name or new file type.

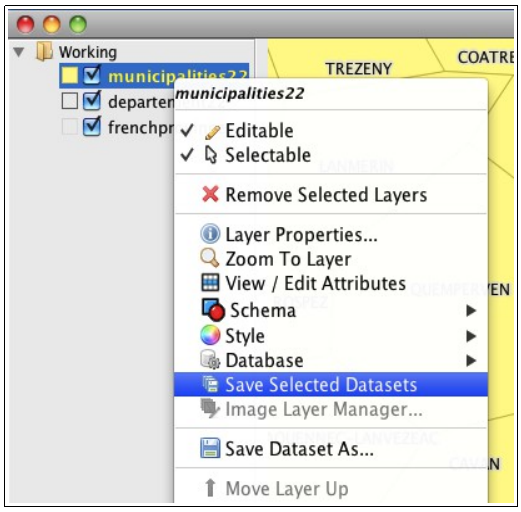


Figure 19 - Saving edited datasets.

2.3 - Importing and joining alphanumeric data

In the next step we want to join demographic data that contain the number of inhabitants for the year 1954 and 1999, stored in a table (from Excel or OpenOffice), with the polygons that describe the municipal areas from our dataset "municipalities22".

First, let's open the file with the demographic data, *pop22-5499.ods*, with OpenOffice. Go to **[File>Save As...]**. Choose the format *Text CSV (.csv)* from the drop down list, click "save". In the next dialog select "Keep Current Format", and then set in another dialog the values as given in figure 20 (Text delimiter field is empty). Save with click on "ok".

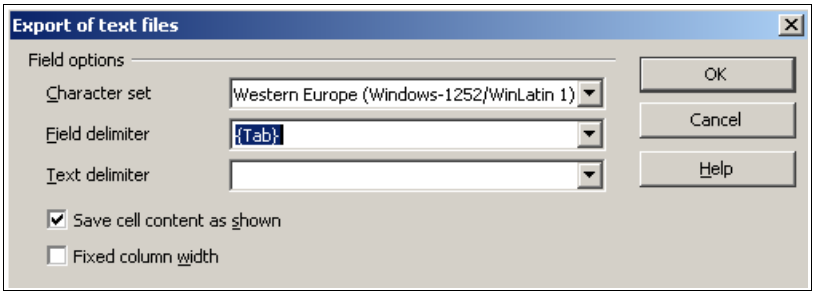


Figure 20 - Exporting table data to CSV format in OpenOffice. You may also use UTF-8 as character set.

Now we go back to OpenJUMP and select **[File>Open File...]**. Choose from the drop down list of file types the option "Csv (set options)(*.txt, *.xyz, *.csv)" (figure 21).

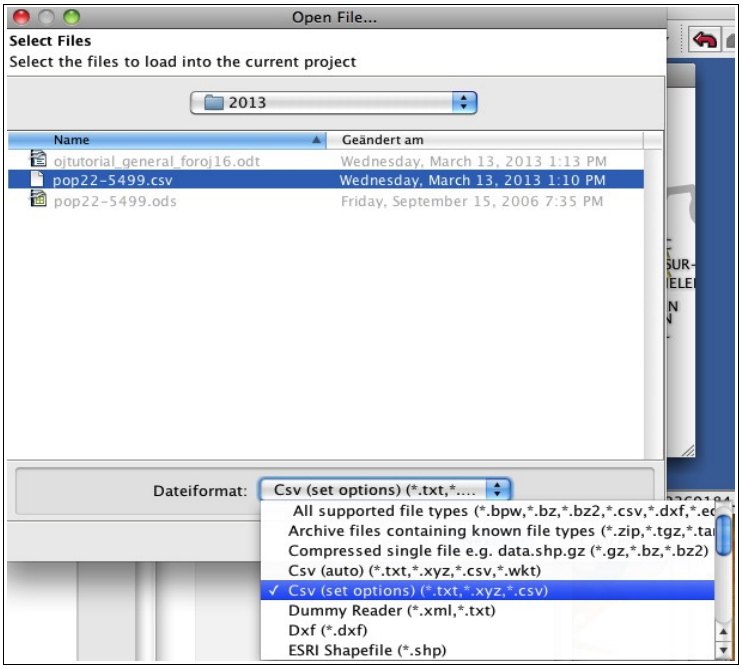


Figure 21 - Loading of *.csv files.

After selecting the file type and the file, click on the [Next] button to do the file specific settings (figure 22). For the encoding type, select "windows-1252" (or UTF-8, if you chose that character set before). As the file does not have comments, you can leave the setting as is. Next chose as Field Separator "{tab}". Leave the box for existing "Header (column names)" checked. Leave unchecked the box for "Data types". Finally, as this dataset has no geometry information, set for all "Column containing … value" to "-" (found at the very end of the list). If you are done with that click the [Finish] button, so that the data are loaded as a new layer without geometries (indeed, empty GeometryCollections are associated to each feature).

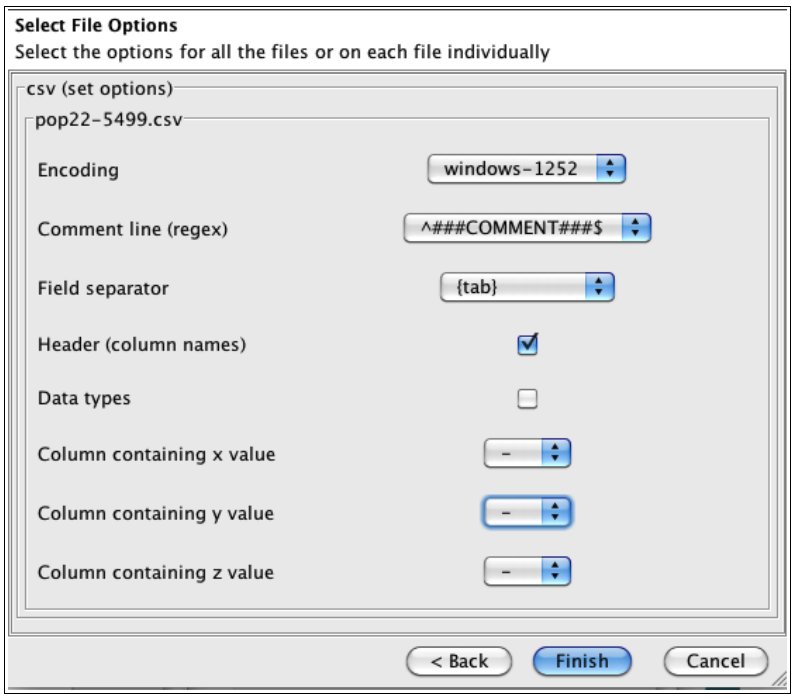


Figure 22 - Settings for reading the file *pop22-5499.csv*.

You will have now a new layer named "*pop22-5499*" in your layer list. Next we need to connect the municipalities with the demographics data using a unique key – a muncipality code in this case. A first problem to solve is that all attributes are read as type text, i.e. String. Therefore

we cannot use the attribute DEPCOM, yet, to link both datasets. How to change text attributes into numerical values works similar to what we descibed in the previous section "Editing Attributes of a Layer", by selecting the attribute type.

First, make both layers *editable* to change the atttribute types. Next, by using [**Schema** > **View / Edit Schema...**] from the layer context menu, change the type of the attribute "code_num" of the layer *municipalities22* from "String" to "Integer" (don't forget to hit the "Apply changes" button). Then, change the attribute type of attribute DEPCOM of the layer *pop22-5499* to the type "Integer". Change also the attribute types for the two attributes PMUN99 and PMUN54, but to type "Double". Do not change the attribute type for the attribute NAME as this is a text attribute.

Now, having the correct attribute type, we can "join" our two datasets using "DEPCOM" and "code_num" as keys/ID's. To do that, select the function [**Tools>Edit Attributes>Join Table...**]. In the dialog that appears choose as base-layer *municipalities22* and the attribute "code_mun" as the one containing the unique feature IDs. Select the layer *pop22-5499* as the layer with attributes to join and use the attribute "DEPCOM" (figure 23).

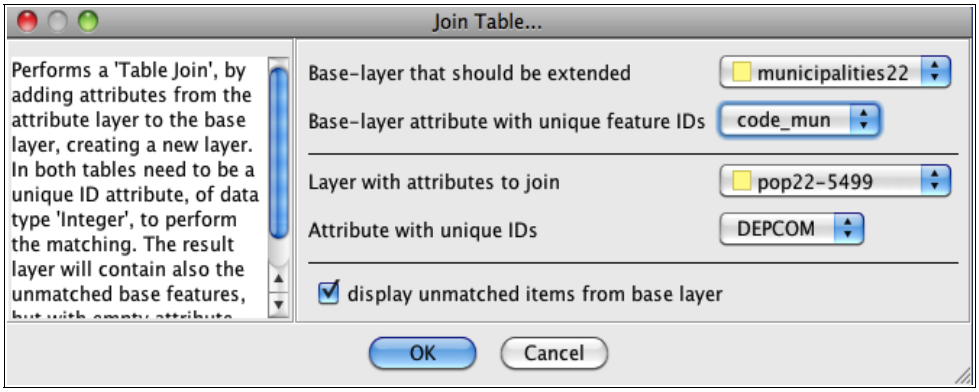
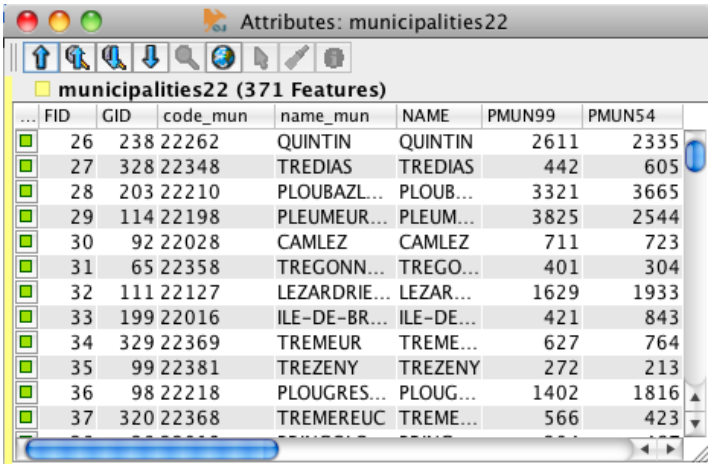


Figure 23 - Setting the unique keys.

After the join is performed a new layer "municipalities22 – join result" is created under the category "Result". This layer will contain the attributes of both layers as seen in the attribute table of figure 24. To keep the result, you need to save the layer.



...	FID	GID	code_mun	name_mun	NAME	PMUN99	PMUN54
	26	238	22262	QUINTIN	QUINTIN	2611	2335
	27	328	22348	TREDIAS	TREDIAS	442	605
	28	203	22210	PLOUBAZL...	PLOUB...	3321	3665
	29	114	22198	PLEUMEUR...	PLEUM...	3825	2544
	30	92	22028	CAMLEZ	CAMLEZ	711	723
	31	65	22358	TREGONN...	TREGO...	401	304
	32	111	22127	LEZARDRIE...	LEZAR...	1629	1933
	33	199	22016	ILE-DE-BR...	ILE-DE...	421	843
	34	329	22369	TREMEUR	TREME...	627	764
	35	99	22381	TREZENY	TREZENY	272	213
	36	98	22218	PLOUGRES...	PLOUG...	1402	1816
	37	320	22368	TREMEREUC	TREME...	566	423

Figure 24 - Added table data from the CSV file.

Exercise 3 (result in figure 25)

Calculate population densities for the years 1999 and 1954 for the layer *municipalities22* using the Attribute Calculator, and save the results.

The fields with the population counts are *PMUN99* and *PMUM54*, respectively.

The population density is given as density_year = (population_count_year / km²).

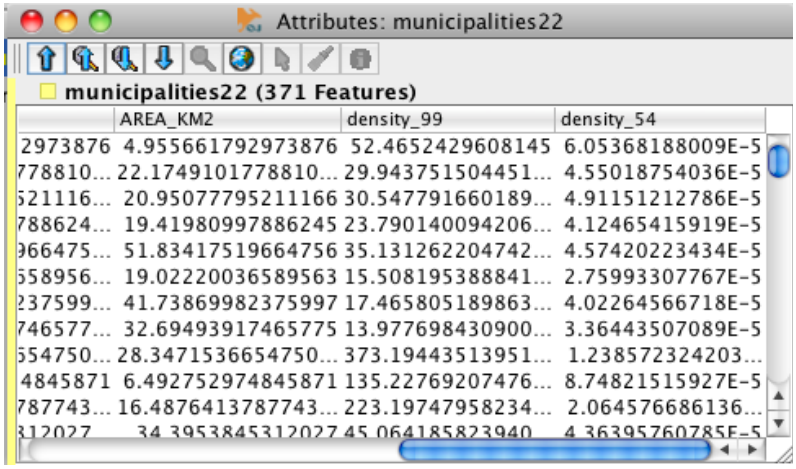


Figure 25 - Result of Exercise 3, the calculated population densities.

3 - Thematic maps

Next we want to create some thematic maps to display and compare the population densities that we just calculated in the previous section's exercise (figure 25). For that we will utilize OpenJUMP functions to perform a classification of the density data first, and then use the viewing tools to visually compare between the population densities for the years 1954 and 1999.

- First, let's create a new project **[File>New>New Project]**.
- Select the layer *municipalities22* in the original project and chose [**Copy Selected Layers]** from the layer context menu.
- Then go to the newly create project and chose **[Paste Layers]** from the category context menu for the category named "working".
- Select **[Window>Mosaic]**.
- Select **[Window>Synchronization>Synchronize pan and zoom]** Now click in one *Map View* and use the zoom and pan functions. The views should be synchronised (figure 26).

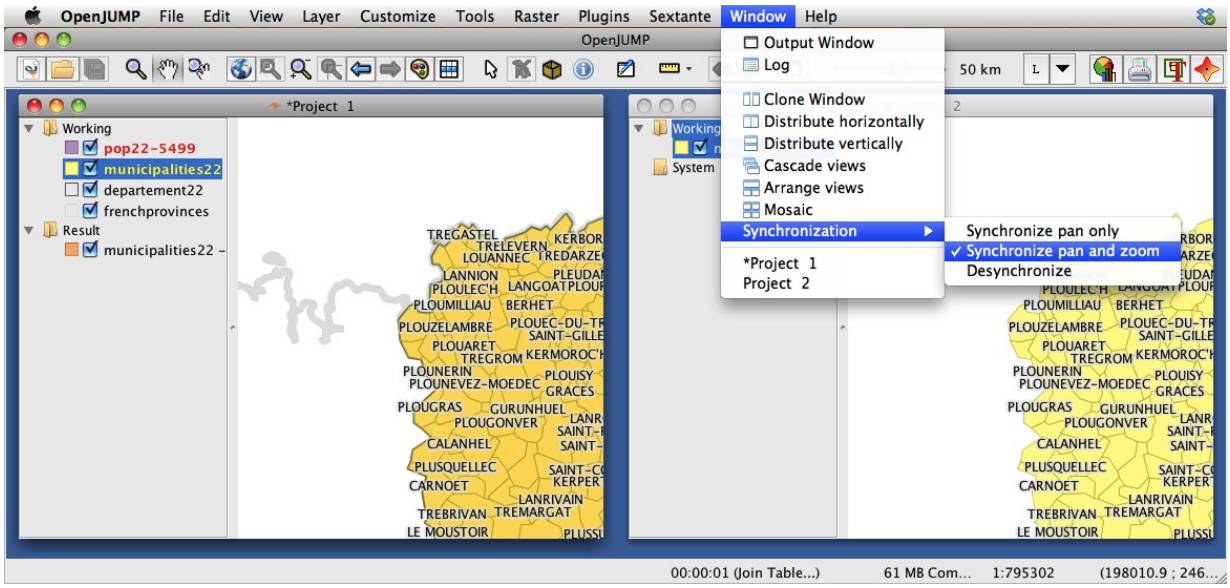


Figure 26 - Synchronising Map Views of different projects.

Now we use OpenJUMPs colour theming functions from the Styling Dialog to display the population density in different colour scales (see figures 27 and 28). Open the "change styles" dialog (with the palette button). The Colour theming tab enables you to choose (i) the attribute you want to classify, (ii) the classification method you want to apply to this attribute and the number of desired classes (numeric attributes only), and (iii) the colour scheme you want to use. Apply the following colour theming: Quantile classification, 7 classes, green colour schema:

- project 1 : population density for 1954
- project 2 : population density for 1999

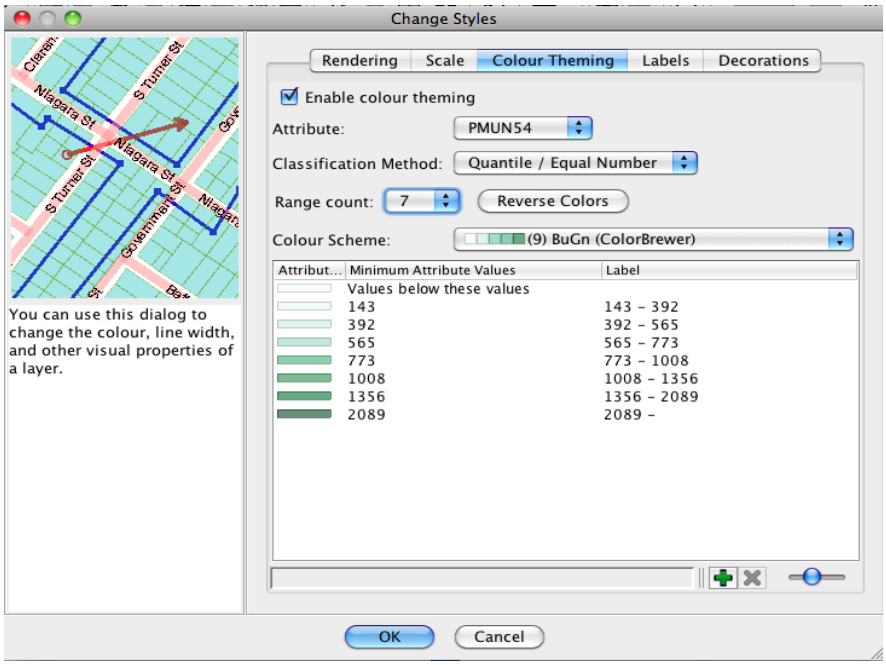


Figure 27 apply colour theming to the dataset (result in Figure 26)

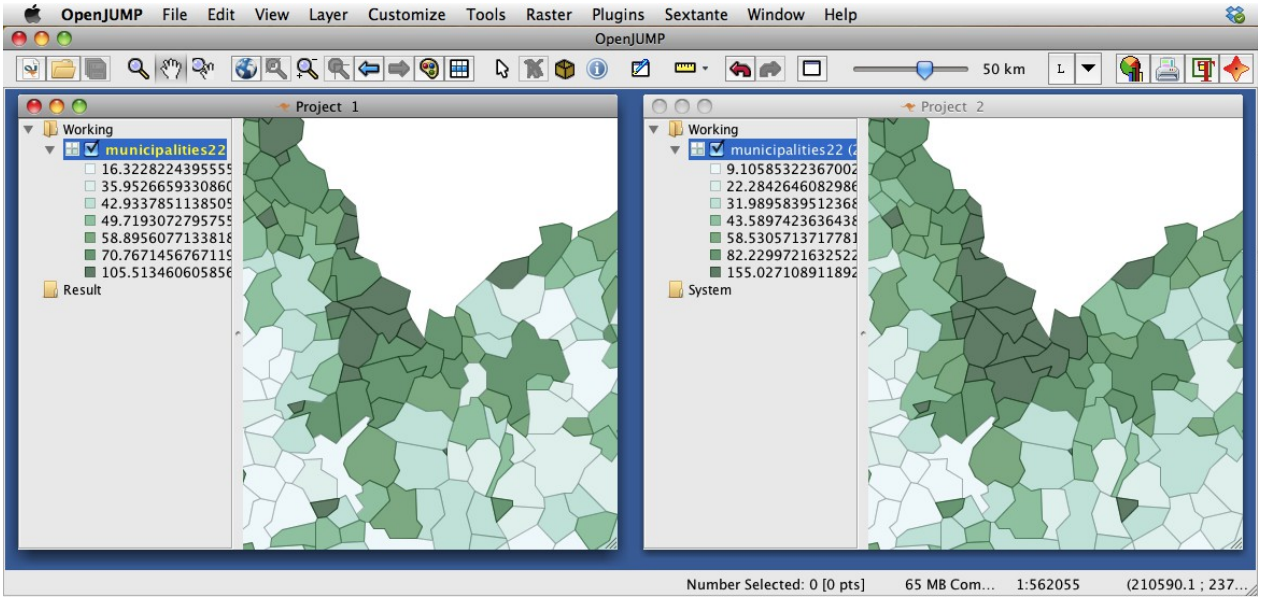


Figure 28 - Result of the colour theming based on the population density. Note that the ranges of the classes are different since the ranges are based on different data (i.e. densities for different years). So the views are not really comparable.

Besides the comparison for different years you may also compare the different visualisations for the different classification methods, e.g. Quantiles vs. Jenks Optimal Method. If you are finished you should de-synchronize the views and close Project 2, so that the original project is left.

4 - Exporting views

Currently OpenJUMP provides two build-in options for the export of Map Views. One can store an image in *.png or *.jpg format, and in the PLUS edition one can export the map view as Scalable Vector Graphics (*.svg). With svg you can later do cartographic editing in a drawing and layout software such as Inkscape or Adobe Illustrator. However, with Cadplan's JumpPrinter Extension that comes with the PLUS edition you can also store in *.pdf format (see next Chapter).

Before we export the image, it is useful to add a scale bar to the view via **[View>Map Decorations>Scale Bar]**. It is *not advisable* to add the map scale as a number (using **[View>Map Decorations>Scale Display]**), since during printing and image export the final map scale (on paper) will likely be different. But the scale bar can be helpful to determine in what map scale the prints will be. Hence, you can test the existence of a difference between (i) map scale determined by OpenJUMP for the screen display, and (ii) the real map scale in the print-out, by measuring the length of a unit of the scale bar with a ruler.

The screen scale can be adjusted with **[View>Zoom to Scale]**.

To export the view chose **[File>Save View As>....]** (figure 29). If the map view is stored as raster graphics, then a geographic reference file (*.jgw or *.pgw) is written as well, if the option "Write world file" is checked (figure 30). The field labelled *Width (#pixels)* can be used to define the number of pixels for the *width* of the output image.

Thematic legends can only be saved as *.png images. To do so, click on the layer with the legend you want to save and chose from the layer context menu **[Style>Save Legend...]**.

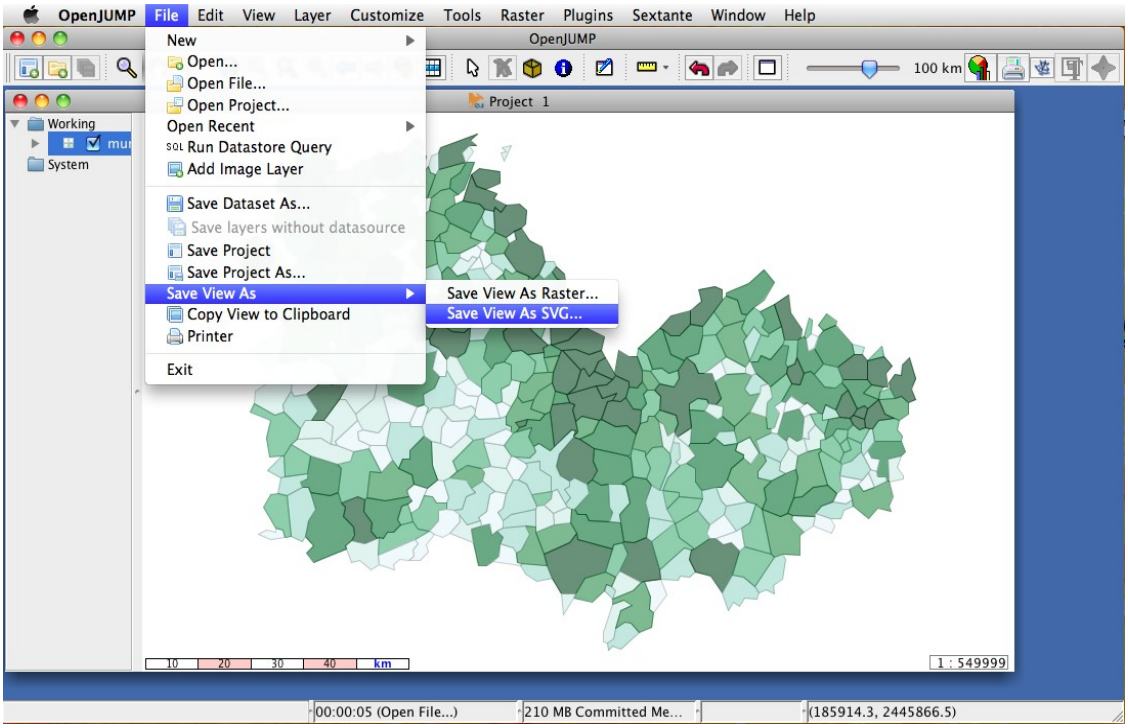


Figure 29 - Saving a Map View as raster or vector graphics.

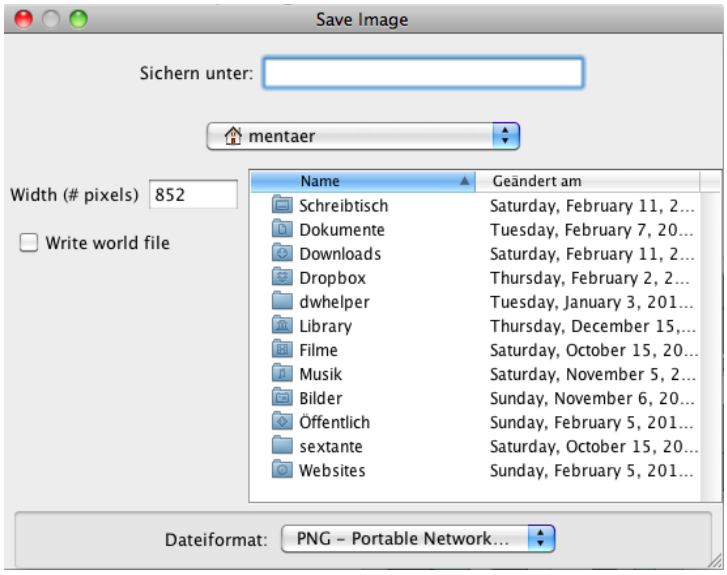


Figure 30 - Saving a Map View as raster image with options to write a world file and to define the image size.

5 - Printing

OpenJUMP does not have build-in printing functionality. However, two plug-ins/extensions exist that allow to generate maps: i) CadPlan's JumpPrinter Extension (available on www.cadplan.com.au[4]), and ii) Intevation's PrintLayout Plugin (available from http://sourceforge.net/projects/jump-pilot/ under files, OpenJUMP plugins). The CadPlan plugin is included in OpenJUMPs PLUS edition. Hence, for the following task we'll use the CadPlan extension.

First we do some preparation to print more than the areas of the municipalities:

- except for the *municipalities22* layer make all other layers invisible by selecting them in the *Layer List View* and chose then in the layer context menu **[Toggle Visibility]**
- open the *Attribute Window* of municipalities22 and go to the field that contains the population density for the year 1999

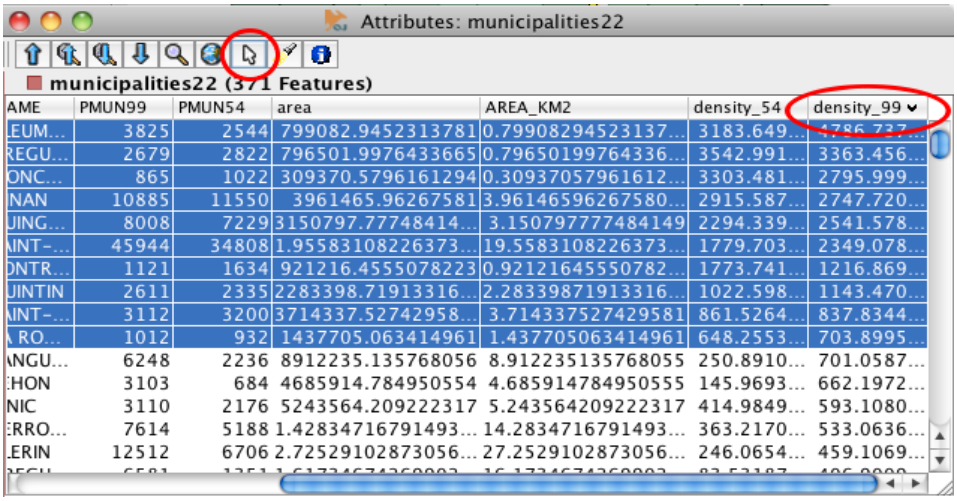


Figure 31 - Sorting rows for population density.

- sort the field with the highest to lowest population density by clicking on the column header that contains the name (figure 31)

4 Note, the JumpPrinter Extension requires the Vertex Symbols Extension.

- now select the top ten entries (highest density for 1999) and click on the *select button* on top (figure 30) to select the municipalities in the *Map View*. Close the attribute window.

We copy those areas with the highest population density into a new layer to print only the name of those:

- while the 10 municipalities are still selected chose **[Edit>Replicate Selected Features]** and press the [ok] button in the upcoming dialog. A new Layer, called "new" will be created with those 10 areas. You can now deselect the areas using the "Clear Selection" button of the toolbar.
- now open the styles dialog for the layer "new" and in the Rendering Tab uncheck the box for *Fill*, select a red line colour, and set the line width to 3.
- Go to the Labels tab and enable labelling, choose "name_mun" as attribute to label, and check the box for "draw outline around labels" (figure 32)
- rename the layer "new" to "Top Ten" by double clicking on the name

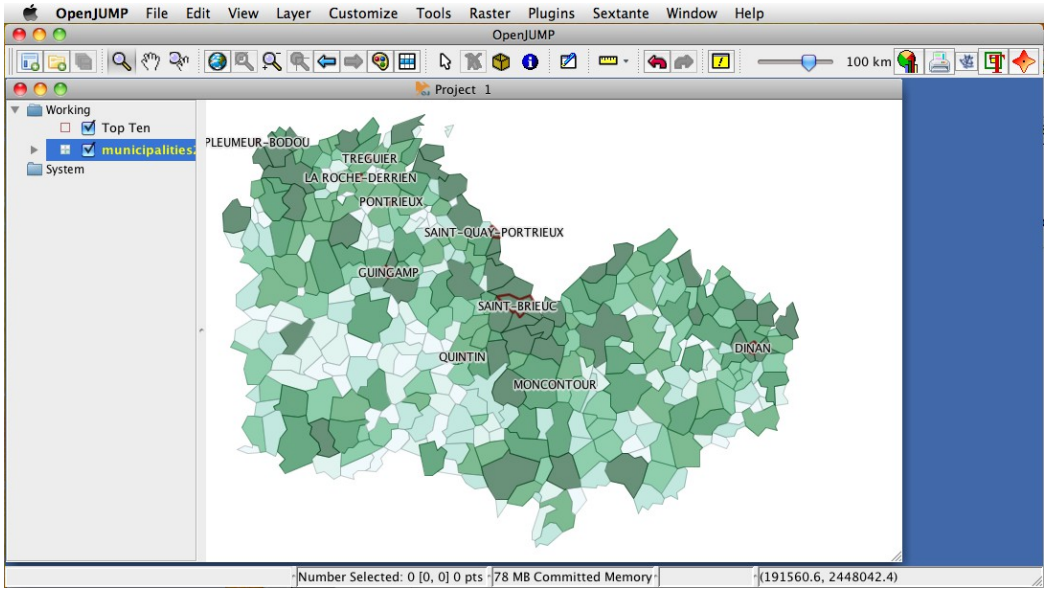


Figure 32 - Municipalities with highest population density.

To print this map shown in figure 32 we click on the "Printer" button  that is in the top toolbar, if CadPlan's JumpPrinter Extension is installed. A new window will appear (figure 33).

- click on the button [Page Setup] to set the page to *landscape* format and use the button [-] on top to zoom out for checking the printer borders.
- in the scale box, set the scale to the value 850000 – or check the box for "Fit to Page"
- click on the button [furniture] to add some map items of your choice, such as a scale bar, a north arrow, a legend, and so on (figure 34).

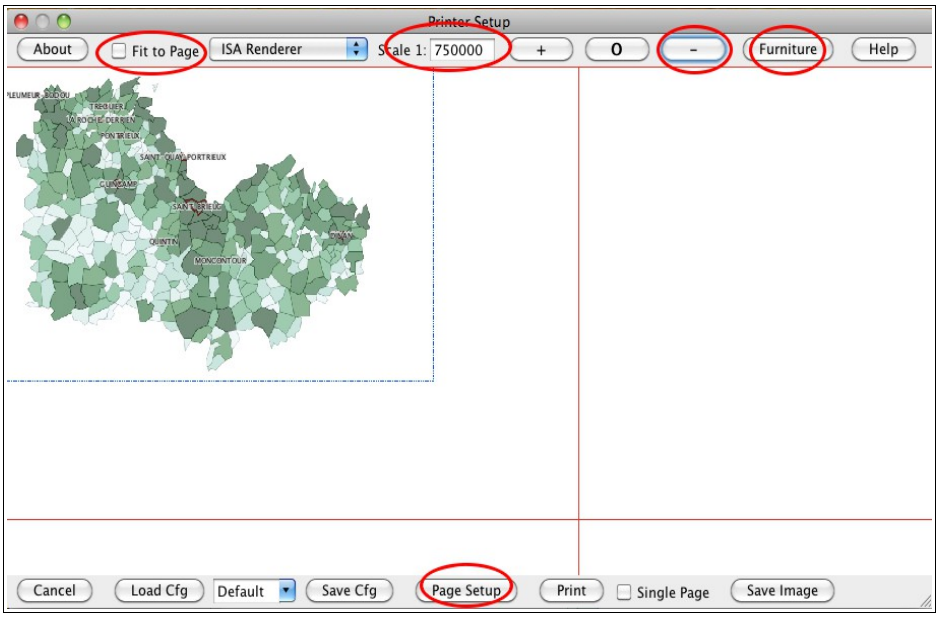


Figure 33 - The print window of CadPlan's JumpPrinter extension.

Besides printing the result, the CadPlan JumpPrinter Extension allows as well to save the map as image in *.jpg, *.png, *.svg or *.pdf format by using the [Save Image] button.

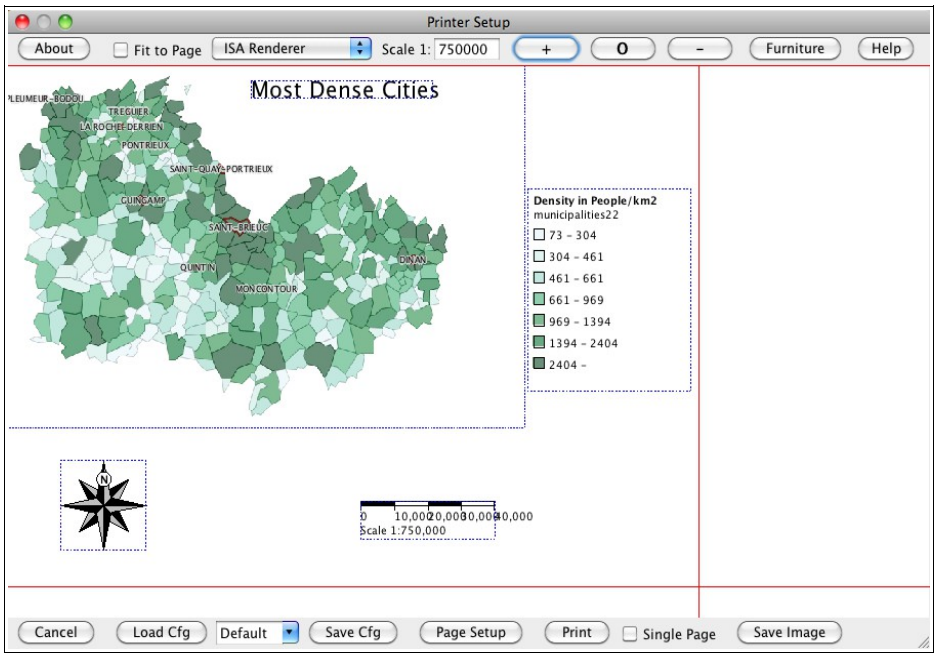


Figure 34 - Map/Furniture Items added to the map.

6 - Querying data

In this section we will introduce you to the different data query tools in OpenJUMP. In general we will distinguish between Spatial Query tools and Attribute Query tools below. All query tools can be found in **[Tools>Queries>...]**.

6.1 - Simple attribute queries

Load the following datasets (figure 35) :

- landcover2000.shp,
- waternetwork.shp
- hedgerow.shp,

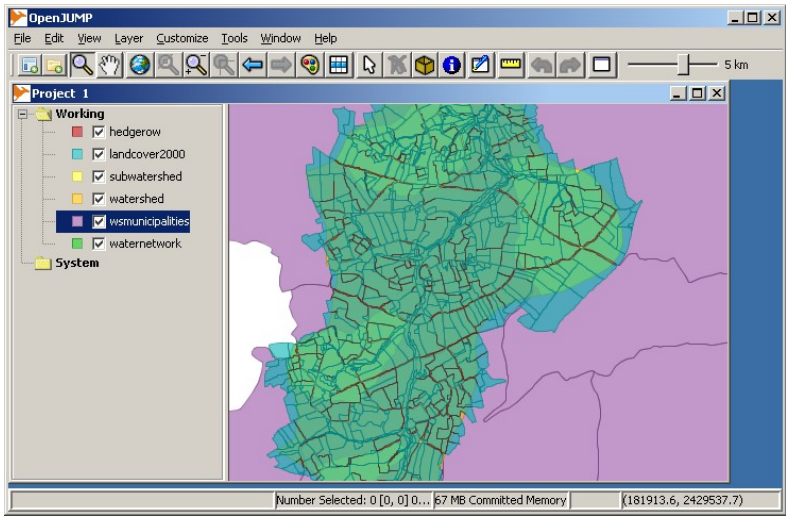
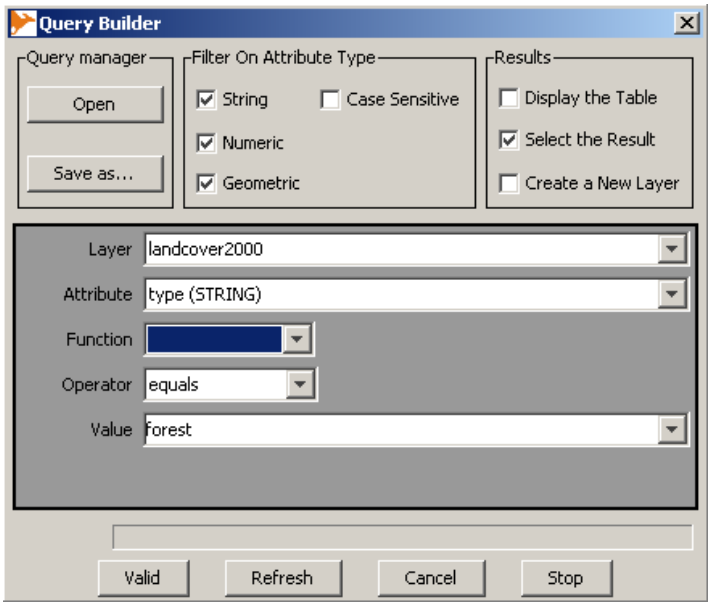


Figure 35 - Data for the query exercise.

Example 1: We want to select forest parcels (from *landcover2000*) with areas exceeding one hectare (i.e. 10'000 m^2) and set for these parcels the *type* attribute value to "big forest". This query needs to be subdivided into two queries. With the first query we will obtain all forest parcels and with the second we retrieve all parcels that exceed 1ha.

- Go to **[Tools>Queries>Simple Query...]**. This function will open a new dialog named "Query Builder" (figure 36). The Query Builder enables to save your queries and to choose what kind of result you want (features selection, table display, or new layer).
- Select the forest parcels according to the settings in Figure 36. Click on the [valid] button to accomplish the selection.

Figure 36 - The query builder of Simple Query.



- Change the value in the field "Layer" from "lancover2000" to "Selection" using the dropdown list. Then set the other values in such way that we retain areas larger than 1ha (figure 37). After using [valid] 26 forest areas should be selected now (Figure 37). How many objects are selected, can be seen in the bottom bar.

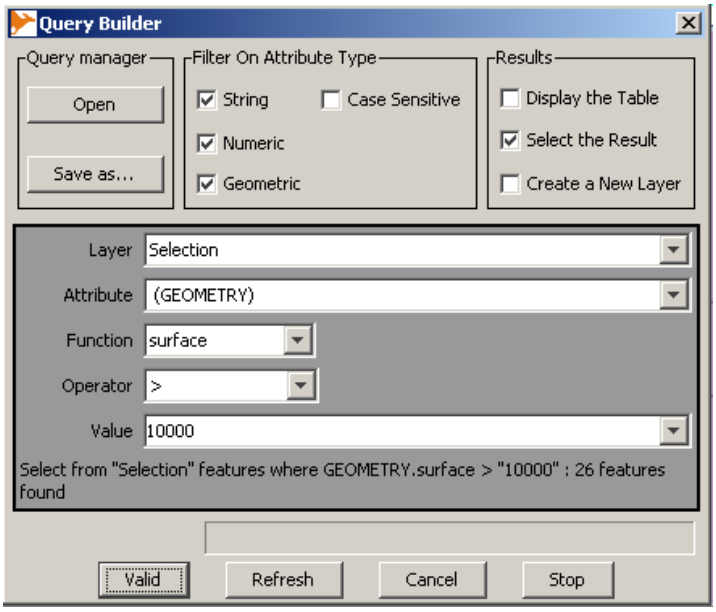


Figure 37 - Selecting only larger parcels.

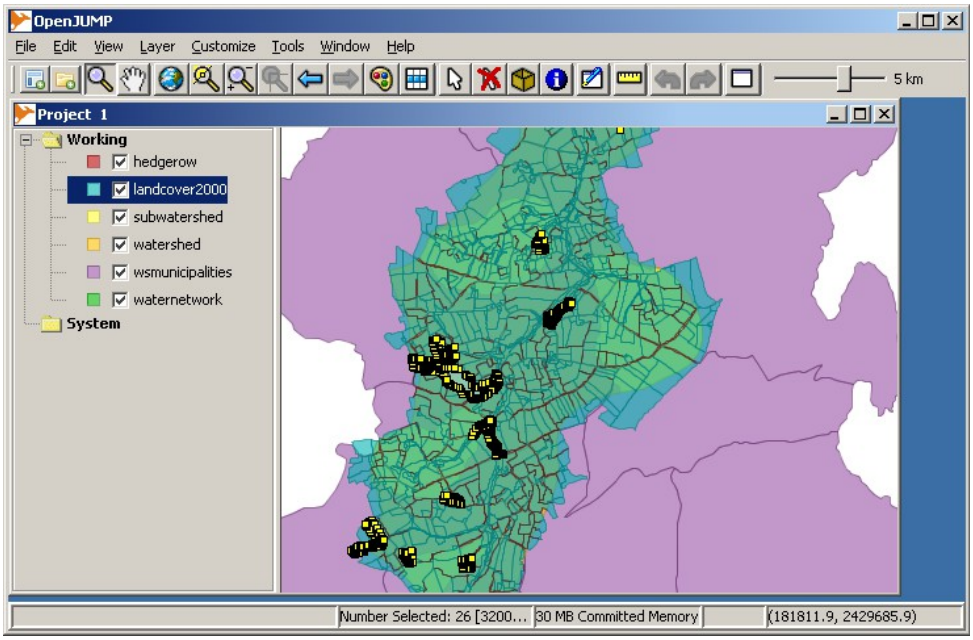


Figure 38 - Selected larger forest parcels from landcover2000.

To change *type* value for the selection parcels we do the following:

- Make the *landcover2000* layer editable.
- Go to [**Tools>Edit Attributes>Auto Assign Attribute...**], which opens a new dialog (figure 39).
- Set the correct layer (*landcover2000*) and destination attribute (*type*), make sure that "selected features only" is checked. Then check "Assign a fixed value", and set as value: "big forest" (figure 39) and click "ok".
- Check with the Feature Info Tool from the toolbar or within the attribute window if the values were changed. 

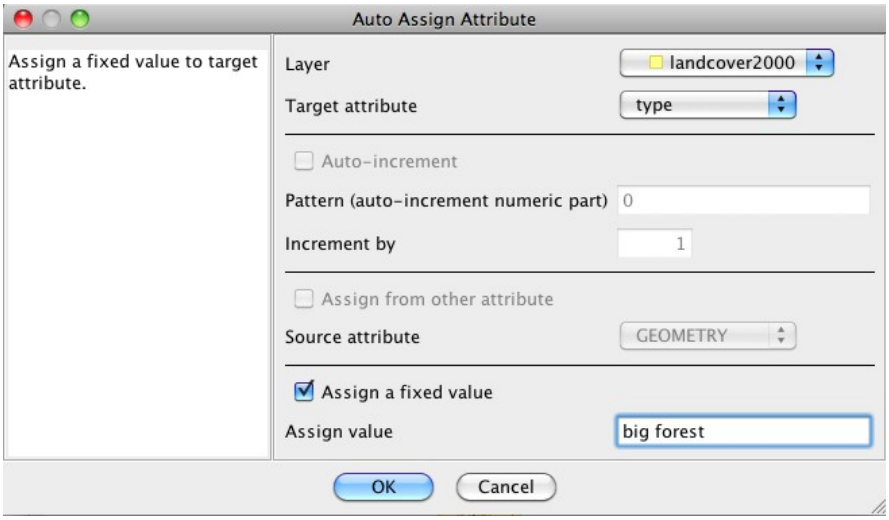


Figure 39 - the "Auto Assign Attribute" dialog

6.2 Spatial queries

Spatial queries have a spatial component and are used if we ask questions such as: *What objects are within a certain distance of object A?* Or: *What objects are within object A?* Hence, spatial predicates such as "contains", "within", "touches", "overlaps", etc. are used in such types of queries.

As an example for a spatial query we want to know what *landcover2000* parcels are within 100 meters of the river network. Such information is useful if, for instance, the river was contaminated by some chemicals. To answer the question we will not use "Simple Query" as earlier, which could be used as well to perform the task, but we will use "Spatial Query".

However, first we need to extract the rivers.

- Open **[Tools>Queries>Attribute Query...]** (figure 40, left)
- set the dialog fields as follows: (i) source layer: *waternetwork*, (ii) attribute: *type_axe*, (iii) relation: "=", (iv) and in the *Value* field type "*river*", (v) keep "create a new layer for results" checked – and press "ok".

Now a new layer called "*waternetwork_type_axe_=_river*" is created that contains rivers only. Rename this layer to "*river*". Next we perform the spatial query:

- Open **[Tools>Queries>Spatial Query...]** (figure 40, right)
- set the dialog fields as follows: i) Source layer: *landcover2000*, ii) relation: *is within distance*, iii) Mask layer: *river*, iv) parameter: 100.0, v) keep "create a new layer for results" checked – and press "OK".

The result of this query is returned in the new layer named "*landcover2000_is within distance_100.0_river*" (figure 41), which contains 646 parcels. This value can be derived from the dialog that opens if **[Layer Properties...]** from the layer context menu is chosen, or simply by flying over the layer name with the mouse pointer.

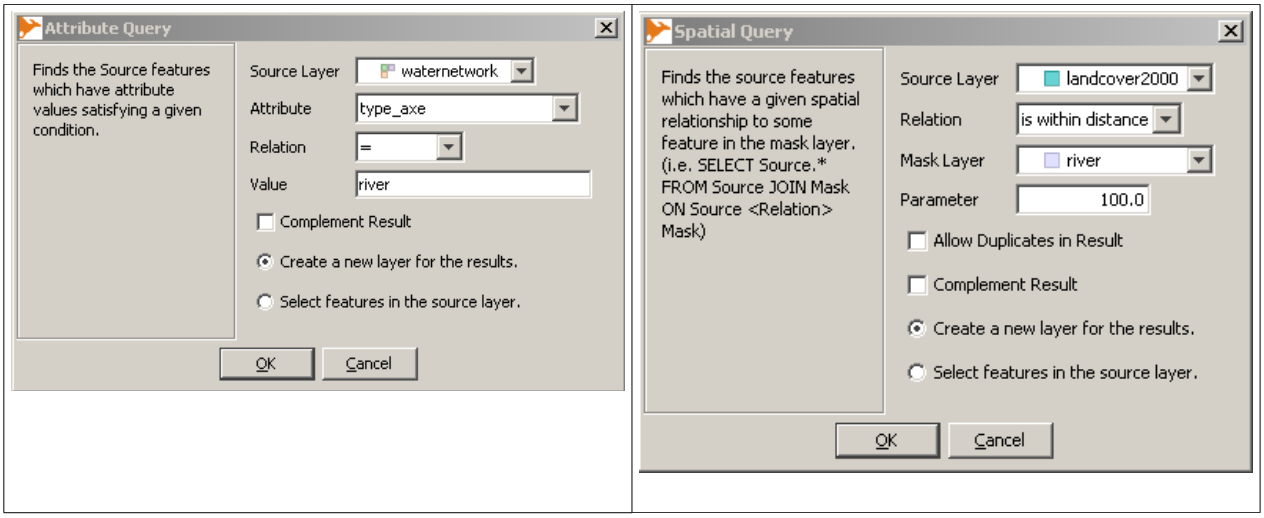


Figure 40 - left: *Attribute Query* dialog, right: *Spatial Query* dialog

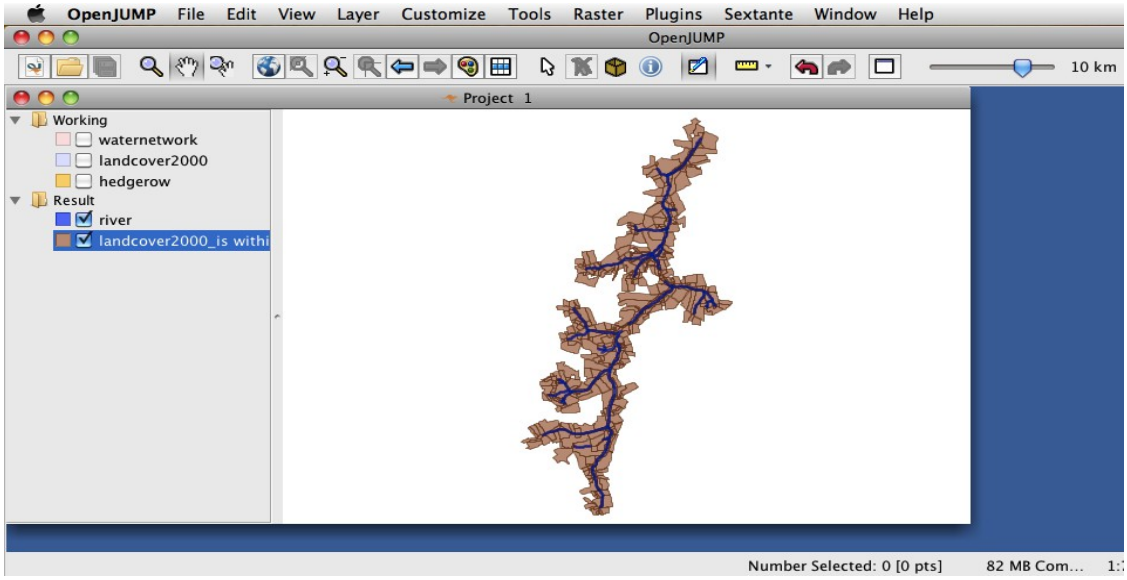


Figure 41 - Land parcels within 100m distance to rivers.

Exercise 4

Exercise 4a

What landcover2000 parcels of the type "big forest" are within 100 meter of the rivers?

Note: You should find 20 parcels.

Exercise 4b

Which are the hedges (from layer *hedgerows*) that "*touch*" grassland parcels (from layer *landcover2000*)? Write down their total length and display the result in a new layer.

Note: To obtain the total length, i.e. the sum, you can use the function **[Tools> Statistics>Layer Attribute Statistics...]** that calculates statistics for any attribute value. As an alternative you can use the function **[Tools>Statistics>Layer Statistics...]** that calculates geometric statistics only.

The number of hedges should be 523, and the calculated total length should be 42'177.79m.

7. Creating and editing geometry data

Before we can start with data creation and editing we need to load our basedata:

- Open the rasters files F018_024.TIF and F018_025.TIF.
- Modify the styling of the *watershed* dataset so that the topographic maps can be seen.
- Extract into a new layer those features of the *waternetwork* dataset those attribute value of *type_axe* is equal to "*river*". Name this layer *rivers* (as in the previous section on Attribute Queries).
- Symbolize the new layer *rivers* using a blue line with line width 4 (figure 42).

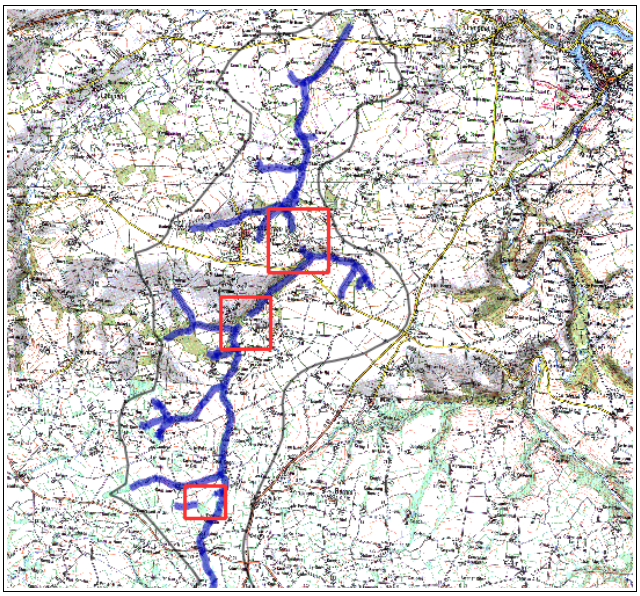


Figure 42 - The topographic maps and the *rivers* of the *waternetwork* dataset.

You will recognize that the rivers in the dataset are not continuous, i.e. that there are three gaps in the network (within the red boxes in Figure 42). The objective is to complete the river network by creating new features based on the topographic map.

- Make all layers except *rivers* "un-selectable" (remove the "check" in layer mouse menu)
- Zoom near the red box in the north
- Make *rivers* editable
- Use the draw line tool to create a new line  (finish the line with a double click)

  If necessary reshape the line

  - by adding, 
  - moving, or 
  - deleting vertices. 

OpenJUMPs keyboard shortcut functions may be useful if one needs to zoom or pan while a drawing tool is activated. See in **[Help>Shortcut Keys...]** what functions are available.

If you have drawn the line, we need to ensure that the (newly drawn) segments really connect to the existing river segments (i.e. no under-shoots are created). Zoom in and use the 
Snap Vertices Tool by drawing a box around the end points of the segments (Figure 43). For that, vertices can be made visible via the styles dialog.

Before snapping *After snapping*

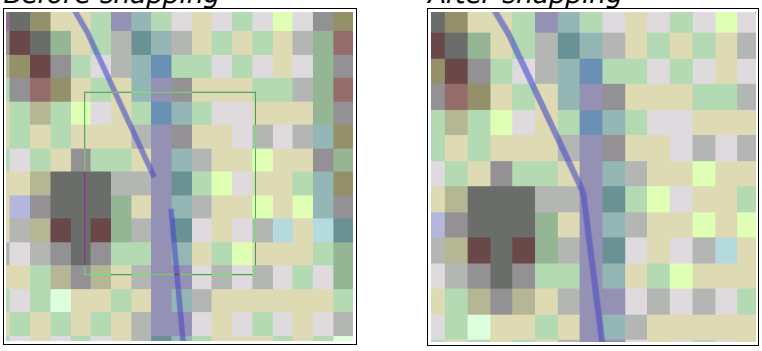


Figure 43 - Using *Snap to Vertices* tool to close gaps in line work.

Note: To enable snapping while drawing and editing geometries, OpenJUMP offers to set snapping options all the time. You'll find these options in [**Customize>Options...>Snap / Grid**].

Next we need to set the attributes of the newly drawn segments.

Use the *Feature Info Tool* from the toolbar to display the attributes of each new segment.

Put the value "river" into the *type_axe* field for each new river segment (figure 44).

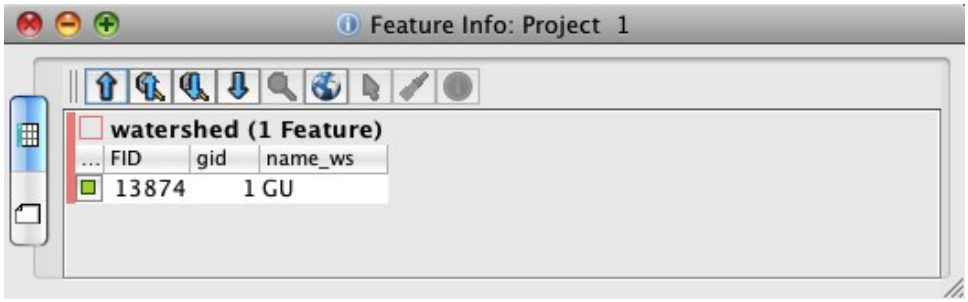


Figure 44 - Using the Feature Info Tool for editing attributes.

If you are done with editing you should save the dataset. You can store the data in a file or in a database, such as PostGIS. To store the data in a file chose from the mouse layer context menu **[Save Dataset As...]**.

How to store data in PostGIS will be described below in Part II of this tutorial. However, if data are to be stored in a database, then it is necessary to have a unique access key for each feature. If no "gid" (i.e., "geometry id") attribute exists in your dataset, which could hold as a unique access key, then you need to create one and assign unique values (see Part II - Section 1.3.1). Otherwise, if the "gid" attribute exists already, then you should assign unique values to the newly created river segments only. Sorting the "gid" values in the table may help you to find out what gid values have been used already.

II - Using OpenJUMP with PostGIS

1 - Installing PostgreSQL / PostGIS

Below we will shortly explain how to start with Postgres and PostGIS. However, if you only want to play around with PostGIS and OpenJUMP then you can also use the LiveDVD/Virtual Machine from http://live.osgeo.org, which has both installed. To get up to speed with PostGIS from the LiveDVD, use the user "user" with password "user" and then walk through the first part of the PostGIS "Quickstart" of the LiveDVD documentation.

1.1 - Creating a database in PostgreSQL/PostGIS

If you don't have access to a **PostgreSQL** database, then install it on your computer together with the **pgAdmin** software. The easiest way to install PostgreSQL and PostGIS on Windows is with the One Click installer package maintained by EnterpriseDB. To install PostgreSQL on different OS, check the postgresql download page :

http://www.postgresql.org/download/

If you have access to a PostgreSQL server, start the pgAdmin tool that is used to manage the PostgreSQL database.

Go to **[File> Add Server...]** (Figure 45).

Apply the following settings:

Name : localhost

Host: localhost (if PostgreSQL is installed on your computer)

Mainteance DB : postgres

Username : postgres

Password : leave empty (or: postgres)

DB restrictions: leave empty

Service : leave empty

You should obtain the database view in Figure 46.

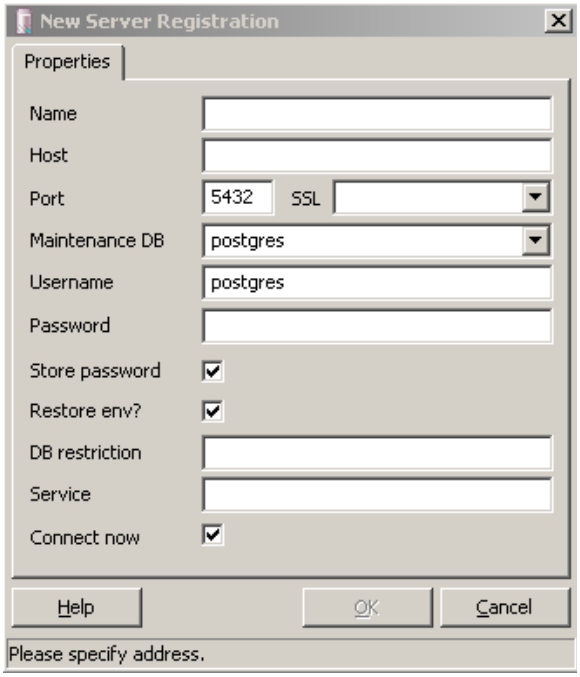


Figure 45 - connecting to a database server

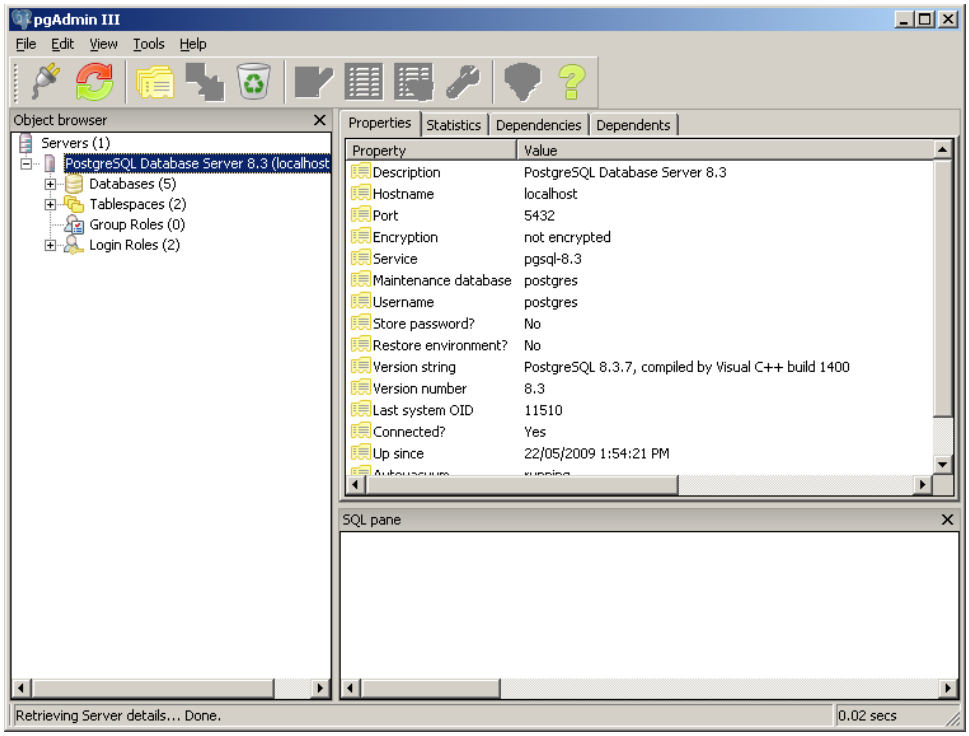


Figure 46 - the pgAdmin interface

Next, we create a new database on the server using **[New Database...]** from the mouse menu (Figure 47) with the following settings:

- Name: gisdb
- Owner: postgres (or the owner of your database installation)
- Encoding: UTF8
- Template: template_postgis (for the workshop use: *gis_template*)

If you can not select *template_postgis* then it may be necessary to carry out the steps in Section 1.2. If an error occurs ensure that the database with the template name shows a red cross, indicating that it is not accessed yet. If *gisdb* was created, continue with Section 1.3

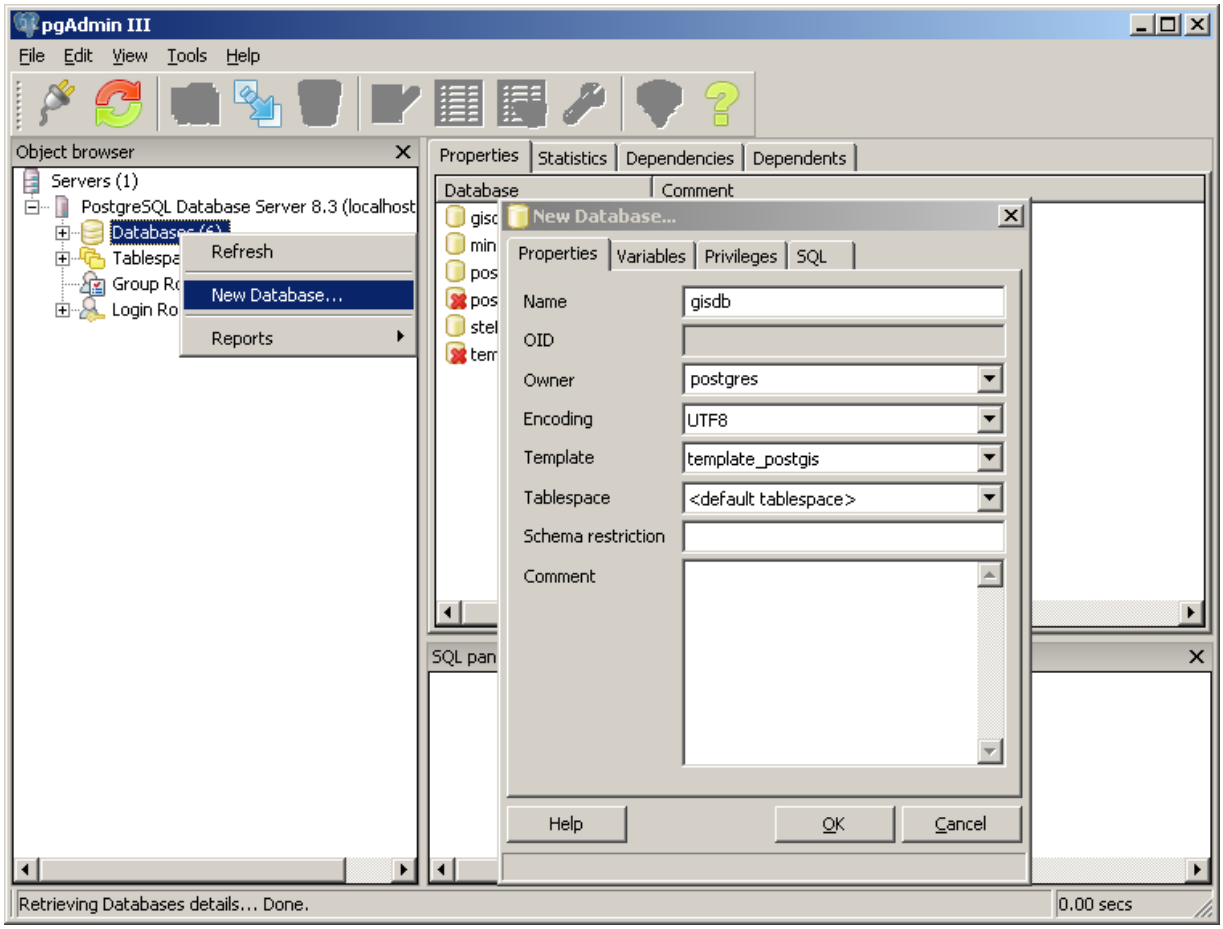


Figure 47 - creating a new database

1.2 - Installing the PostGIS extension

If PostGIS is not installed and you installed PostgreSQL with the "*EnterpriseDB One-click installer*" (recommended for windows users), you have also installed an application called StackBuilder. StackBuilder is an extension manager for PostgreSQL (figure 49).

Start StackBuilder, connect to your PostgreSQL server, then choose the appropriate PostGIS extension as shown in figure 50.

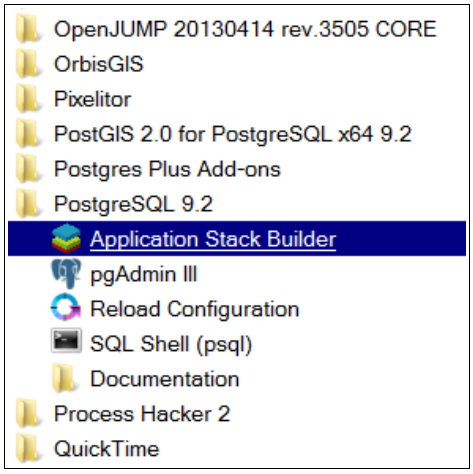


Figure 48

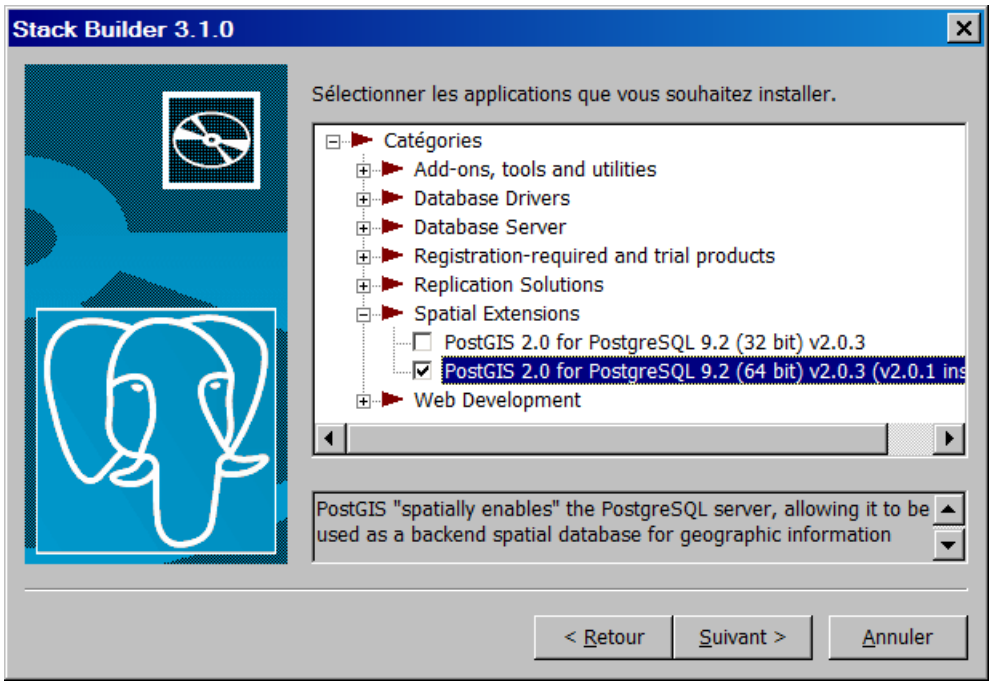


Figure 49: Install PostGIS from StackBuilder

If you are using MacOSX or a Linux distribution, visit this link to get more information about how to install PostGIS :

http://postgis.net/install/

Now we are ready to import and work with spatial data and features in PostgreSQL.

2 - Using PostGIS in OpenJUMP

2.1 - Populating the database

Preparing the data

Create a new project in OpenJUMP and load the following files:

"waternetwork.shp", "watershed.shp".

For each layer:

- In PostGIS tutorials the geometry attribute is often called: "the_geom". In our query examples below we will use "GEOMETRY" instead of "the_geom". However, if you wish to use "the_geom", then do the following: Open the layer schema in OpenJUMP and change the name of the geometry attribute to "the_geom". (figure 51)
- For storage of the data we need a unique access key for every object. Such key is in our case an attribute that contains each value only once. If such attribute is not already existing[5], we can add an attribute "gid" of type integer to the dataset/ layer. Populate it with function **[Tools>Edit Attributes>Auto Assign Attribute...]** using the option "Auto-Increment". Now, every object should have a unique key value.
- Change the SRID[6] to the (French) projection of our dataset: EPSG:27582 - NTF(Paris)/Lambert II (étendu) by using **[Layer>Change SRID...]** (figure 52). Set

5 Note, the FID attribute can not be used for this, since it is a dynamic attribute that is newly generated each time.

6 SRID: Spatial Reference ID

value “27582”. If the SRID number is not known, it can be also set to “-1” for PostGIS version 1.x, or to “0” for PostGIS 2.x.

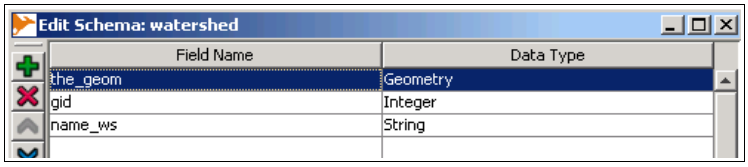


Figure 50 - Change geometry attribute name as needed.

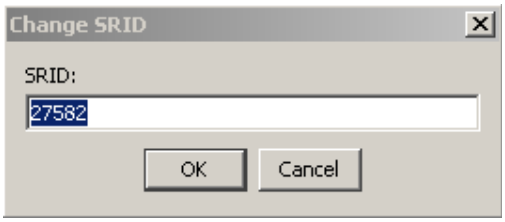


Figure 51 - Set the SRID.

Storing the layers in the PostGIS database “gisdb”

Select the *watershed* layer you want to store in your database and choose **[Save Dataset As...]** from the layer context menu.

In the dropdown list for the storage format choose “PostGIS Table”. (From OpenJUMP 1.6.x onwards, this option is included in OpenJUMP CORE version).

First, you will have to define a connection to a database. Create the connection with the add button  , or select a connection if you have already defined a connection to this database.

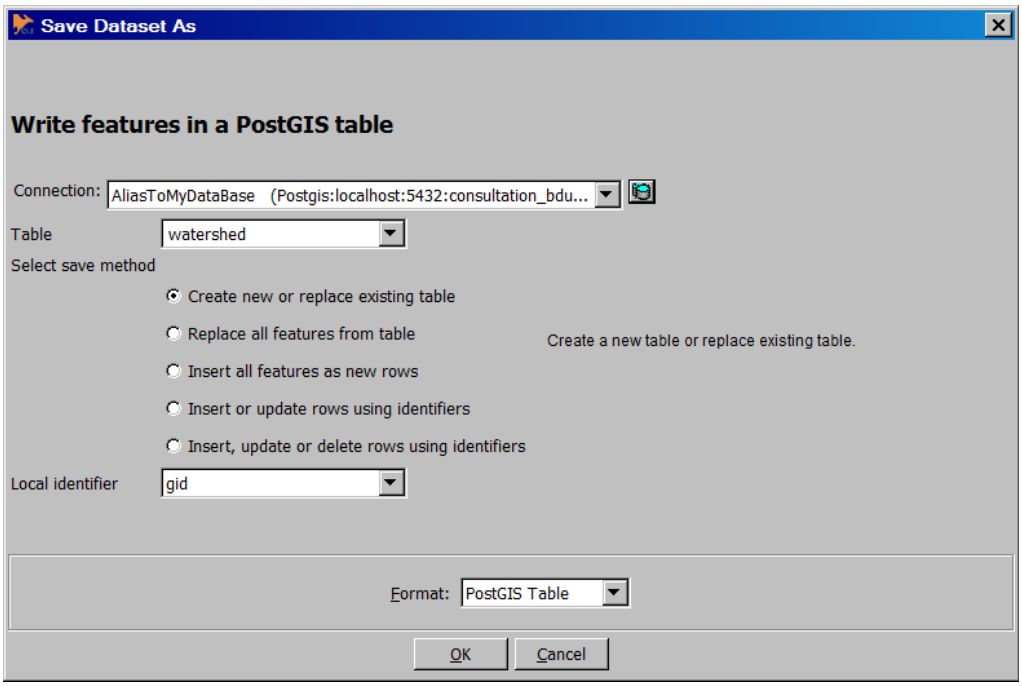


Figure 52 - Committing data to the PostGIS database.

Figure 53 shows you the connection manager to add or delete connections. To create anew connection, fill the form as shown in figure 54.

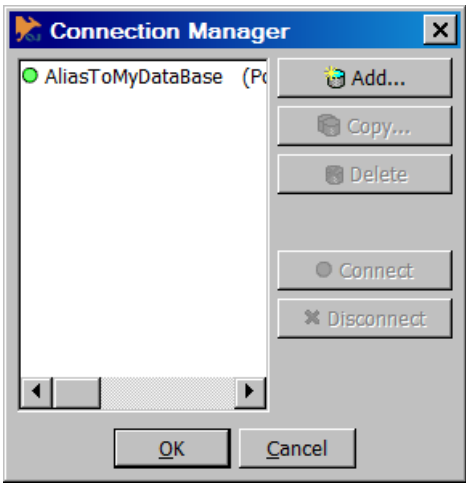


Figure 53: Choose a base to connect to

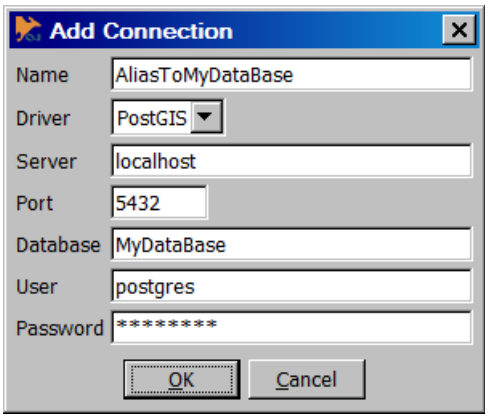


Figure 54: or define a new connection

Warning: The table names and column names are managed by OpenJUMP in a case sensitive way. This means that you'll usually have to use double quotes to query the table from pgAdmin or from other database tools (except if you only use lowercase identifiers).

Once the watershed table is created, add also the *waternetwork* dataset to the database.

2.2 - Display data from PostGIS in OpenJUMP

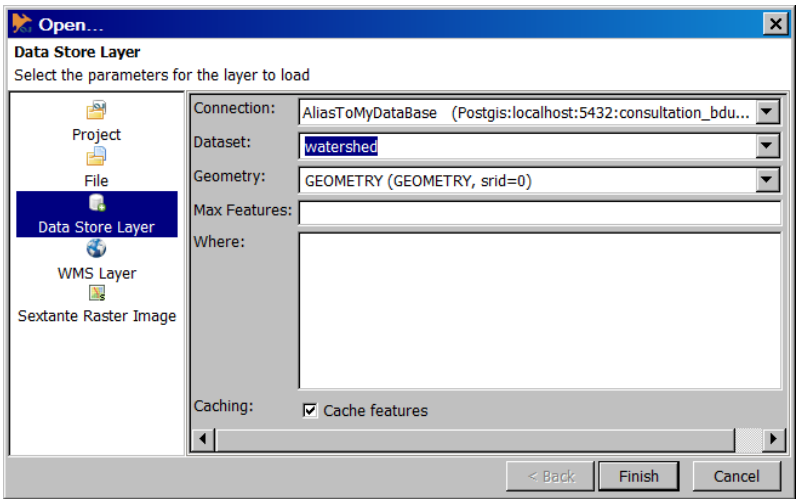
Open **[File>Open...]** and select "Data Store Layer" from the list of options on the left side. (figure 56)

Select the connection (AliasToMyDataBase) and the dataset (watershed) you want to load.

If your dataset has several geometry columns, choose one. The geometry type and the SRID of the column is shown after its name.

You can leave Max Features and Where fields empty.

Clicking on [**Finish**] button will load data intersecting the map view.



Note: If data are loaded with the "Data Store Layer" method then not all data are loaded in OpenJUMP, but only those data that are covered by the current map view (i.e. extent). Hence, if operations/calculations should be performed on the complete dataset, then one needs to zoom to the full dataset (layer mouse menu function: "Zoom to Layer") and eventually duplicate all layer objects into a new layer first (by choosing from the mouse menu: 1. Zoom to Layer, 2. Select current layer items, and 3. [Edit>Replicate Selected Features...]).

That OpenJUMPs is displaying data in a dynamic fashion from data stores is sometimes recognizable by a small clock icon that temporary shows up next to the layer name. This icon indicates when data are requested from the database that are within the current map extents.

2.3 - Performing SQL queries

To filter data in a PostGIS database and display the result in OpenJUMP you can use the Datastore Query function that provides a small GUI to run SQL syntax. Open the dialog by using **[File>Run Datastore Query]** (figure 57).

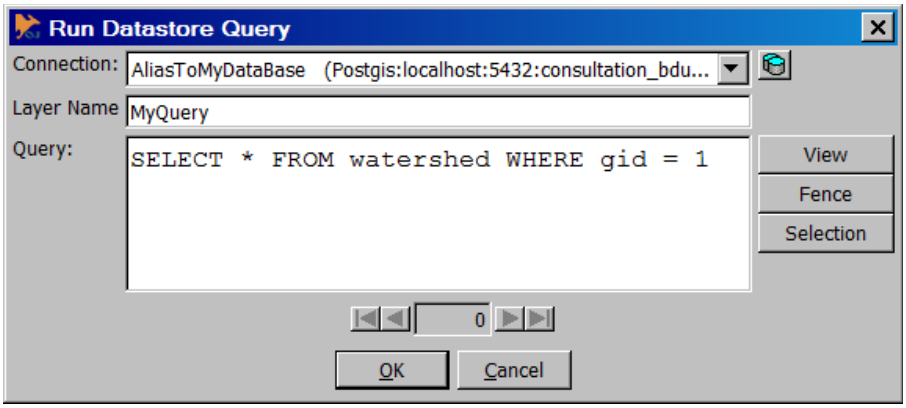


Figure 56 - the Datastore Query Dialog to perform SQL queries in OpenJUMP.

If there are no databases provided in the "connection" drop down list, then you need to add a connection with the button shown on the right of the dialog. How to do that has been explained in the previous section.

Attribute queries can have this syntax:

```
SELECT * FROM <table> WHERE <column name>=<value>;

or:

SELECT <geometry column>, <column list> FROM <table> WHERE <column name>=<value>;
```

The second form should be used if:

(1) you want to extract some columns from your table but not all - because all columns may be a lot(!) of data, and

(2) if your table contains several binary or geometry columns, to make sure OpenJUMP will use the column you want for the geometry.

A query example in which we want to obtain all objects in the *waternetwork* dataset those type is equal to *ditch* is given in Figure 57.

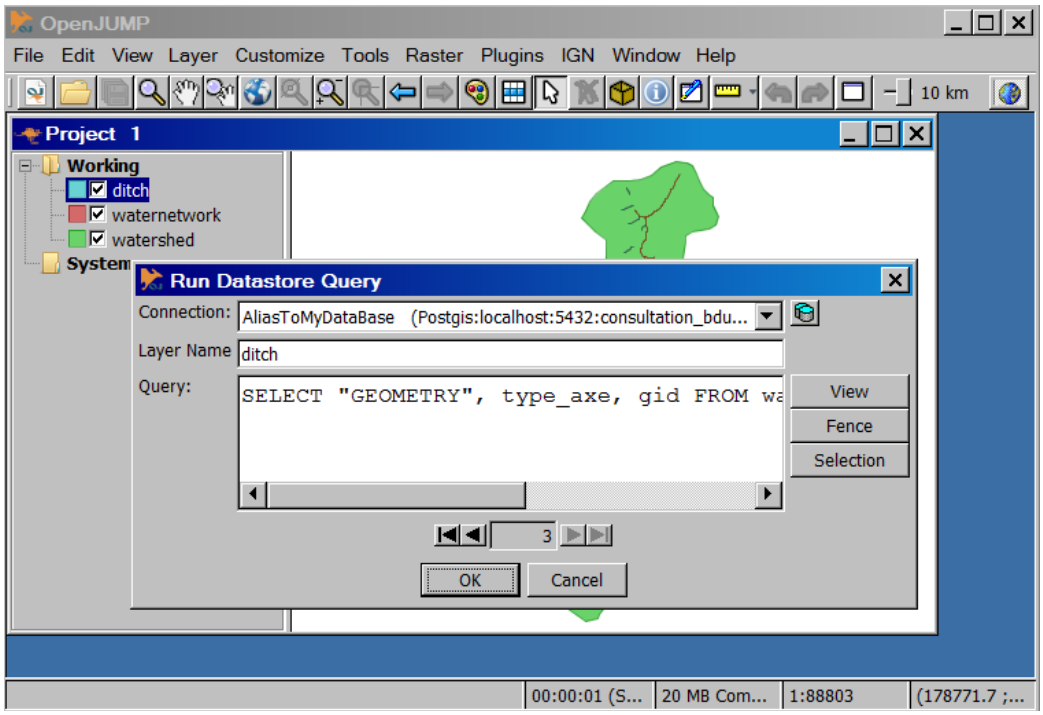


Figure 57 - Selecting ditches from the waternetwork dataset.

The query text is:

```
SELECT "GEOMETRY", type_axe, gid FROM waternetwork WHERE type_axe='ditch';
```

Note that we had to use double quotes as GEOMETRY is upper-case.

In the second example we perform a query that will return those river segments that have a certain length. To get the length for each geometry we have to use ST_Length(<geometry_column_name>).

```
SELECT "GEOMETRY", type_axe, gid FROM waternetwork WHERE ST_Length("GEOMETRY") >
200;
```

Note that the length unit used, is the one used by the dataset itself (meters). The resulting layer should contain 35 *waternetwork* objects of length larger than 200m.

It is also possible to perform queries that include several search statements. This is achieved by using the "AND" statement.

Exercise 5

After saving *landcover2000* in PostGIS find those parcels that
- are *grassland*, and
- have an area greater than 2 hectares

The result must be displayed in OpenJUMP and layer must contains three columns: gid, type and area.

Notes:
- function area is ST_Area(<geometry column name>),
- 10'000 square meters is equal to 1 hectare.
Don't forget that SRID value is 27582 and the unit is metric.

Next, lets see how we can do a spatial query. In the example below we want to select parcels from *landcover2000* dataset that are contained in areas of *wsmunicipalities* that have the municipality name "QUEMPERVEN".

```
SELECT landcover2000."GEOMETRY", landcover2000.type FROM landcover2000,
wsmunicipalities WHERE ST_Contains(wsmunicipalities."GEOMETRY",
landcover2000."GEOMETRY") AND wsmunicipalities.name_mun='QUEMPERVEN';
```

If you have performed the query, then you may zoom in and check out what the predicate "contains" actually covers. That is, in the northern corner of the returned polygons idataset it look like one or two landcover2000 polygons are missing. Can you tell why?

The same query can be performed with *pgAdmin*. The pgAdmin tool also allows to create very complex queries. However, when performing queries in pgAdmin we can not display the query output as a geographic data layer and we need to formulate the query like this:

```
SELECT landcover2000.type FROM landcover2000, wsmunicipalities WHERE
ST_Contains(wsmunicipalities."GEOMETRY", landcover2000."GEOMETRY") AND
wsmunicipalities.name_mun='QUEMPERVEN';
```

3 - More PostGIS Readings & Hands-On

An extensive introduction and more examples on how to perform (spatial) SQL queries with PostGIS can be found on www.postgis.org and in the PostGIS book: **"PostGIS in Action"**, now its 2nd edition, by R.O. Obe and L.S. Hsu (see http://www.manning.com/obe/).

In our download section for OpenJUMP 1.3 there is also a second OpenJUMP - PostGIS tutorial on **mineral targeting** by Ravi Kumar, which presents further examples:
https://sourceforge.net/projects/jump-pilot/files/Documentation/OpenJUMP%201.3%20Docs%20%28English%29/