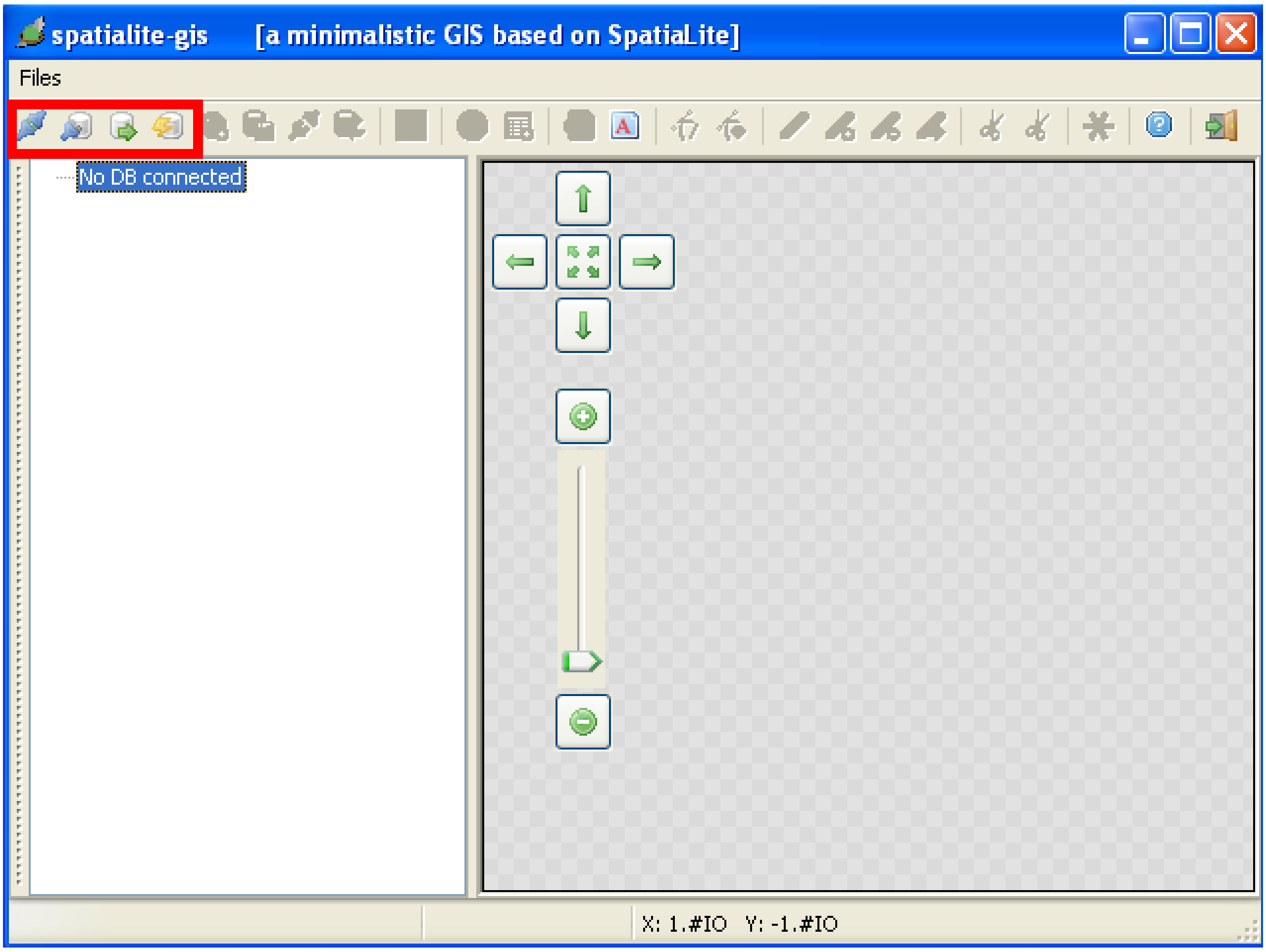
not at all a manual
simply a quick how-to-do guide

As a general rule, the GUI implemented by **spatialite-gis** is closely related to the one implemented by the companion app **spatialite-gui**

So, if you are already accustomed to use **spatialite-gui**, then you'll become quickly familiar with **spatialite-gis** as well

Table of contents:

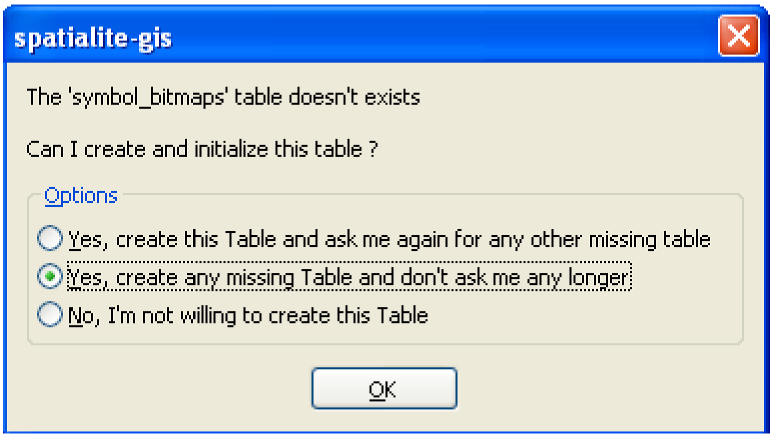
1- Database connection



There is no DB currently connected … **spatialite-gis** absolutely requires a DB connection in order to be fully operative: so you can now:

1. establish a connection to some already existent DB [*may well be, one you've previously prepared using the companion GUI-app* **spatialite-gui**]
2. create an empty [*brand new*] DB, and then connect it
3. load an already existent DB as a volatile, non-persistent, IN-MEMORY DB [*arguably, a reasonably sized one*]
4. create an empty [*brand new*] volatile, non-persistent, IN-MEMORY DB

Now we'll use the **world.sqlite** sample DB in order to go on
So you've now to establish a connection to this sample database



spatialite-gis

The 'symbol_bitmaps' table doesn't exists

Can I create and initialize this table ?

Options

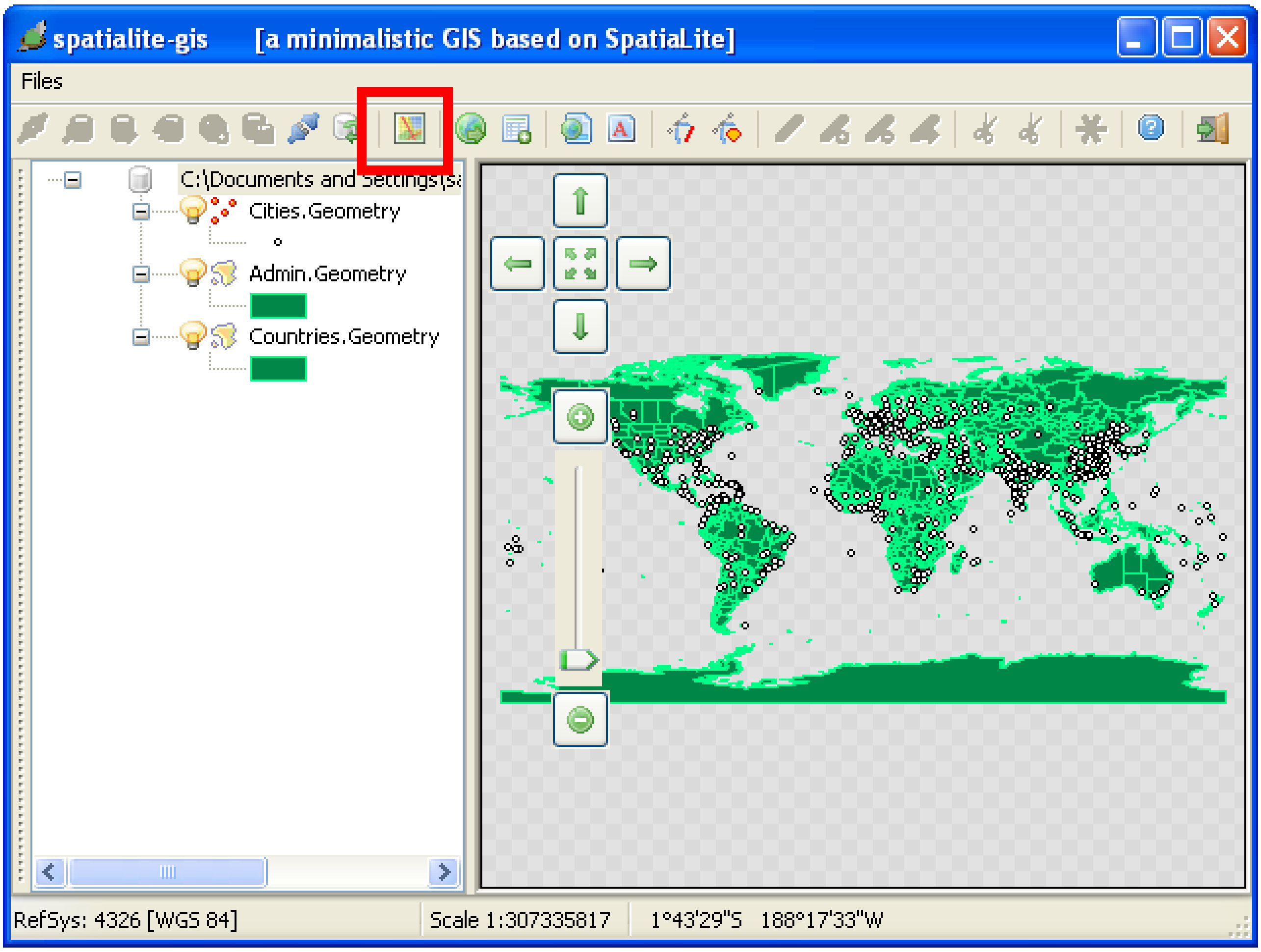
○ Yes, create this Table and ask me again for any other missing table

● Yes, create any missing Table and don't ask me any longer

○ No, I'm not willing to create this Table

When connecting to an already existing DB, this message may be shown; this means one or more internal tables [*absolutely required by* **spatialite-gis**] are missing.
The simplest thing to do is allow **spatialite-gis** creating such tables: so you've to select the ***"Yes, create any missing table"*** option.

2 – Project's configuration



C:\Documents and Settings\s

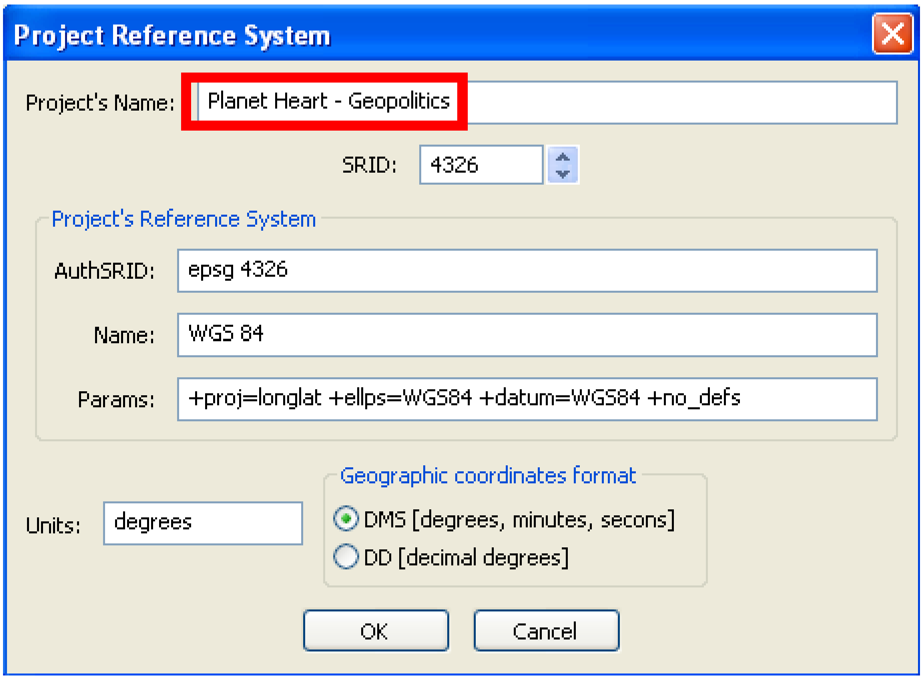
Cities.Geometry

Admin.Geometry

Countries.Geometry

RefSys: 4326 [WGS 84]   Scale 1:307335817   1°43'29"S 188°17'33"W

Now **spatialite-gis** is in a fully operative state.
First we can check the currently used **project's configuration**.



Project Reference System

Project's Name: Planet Heart - Geopolitics

SRID: 4326

Project's Reference System

AuthSRID: epsg 4326

Name: WGS 84

Params: +proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs

Units: degrees

Geographic coordinates format

● DMS [degrees, minutes, secons]

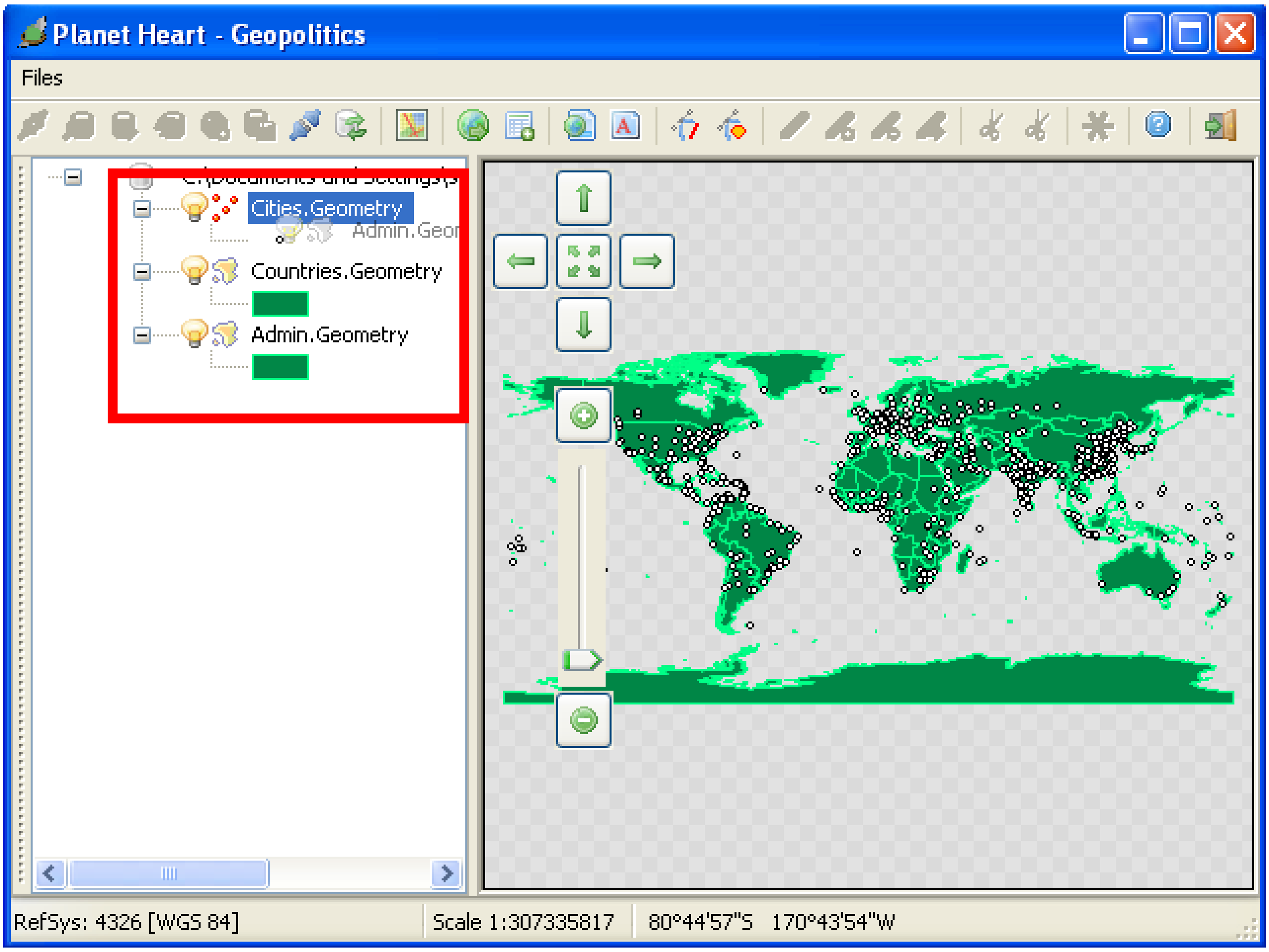
○ DD [decimal degrees]

As you can notice, **spatialite-gis** already applies a **default** project's configuration.

- The **project's SRID** is the one of the first theme/layer found [*but you can change this setting, if required*]
- When using geographic coordinates, you can choose to use the **DMS** or the **DD** notation.

Anyway, it's wisest to set some useful and meaningful **project's name**

3 – Layers' configuration



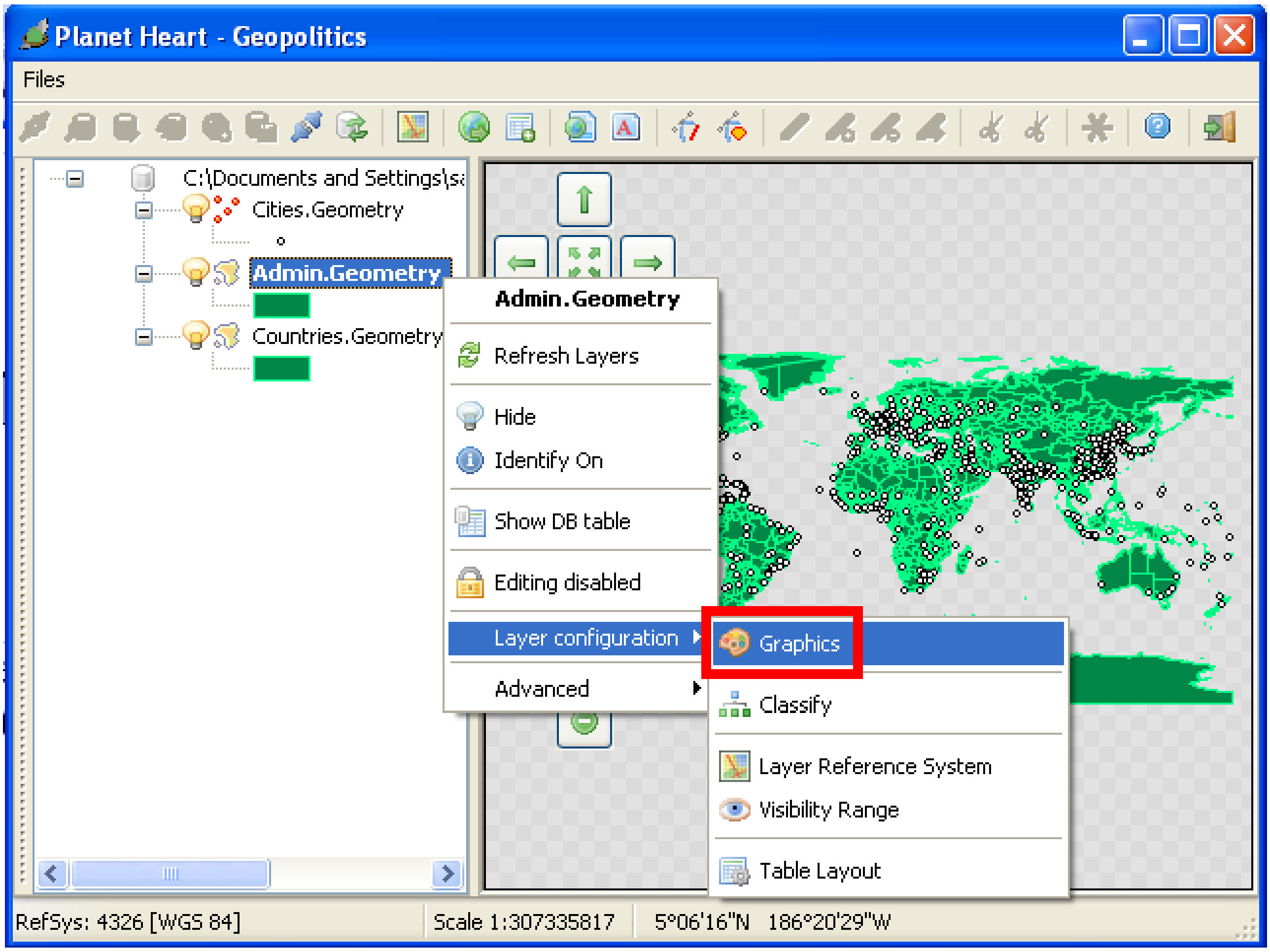
Planet Heart - Geopolitics

C:\Documents and Settings\

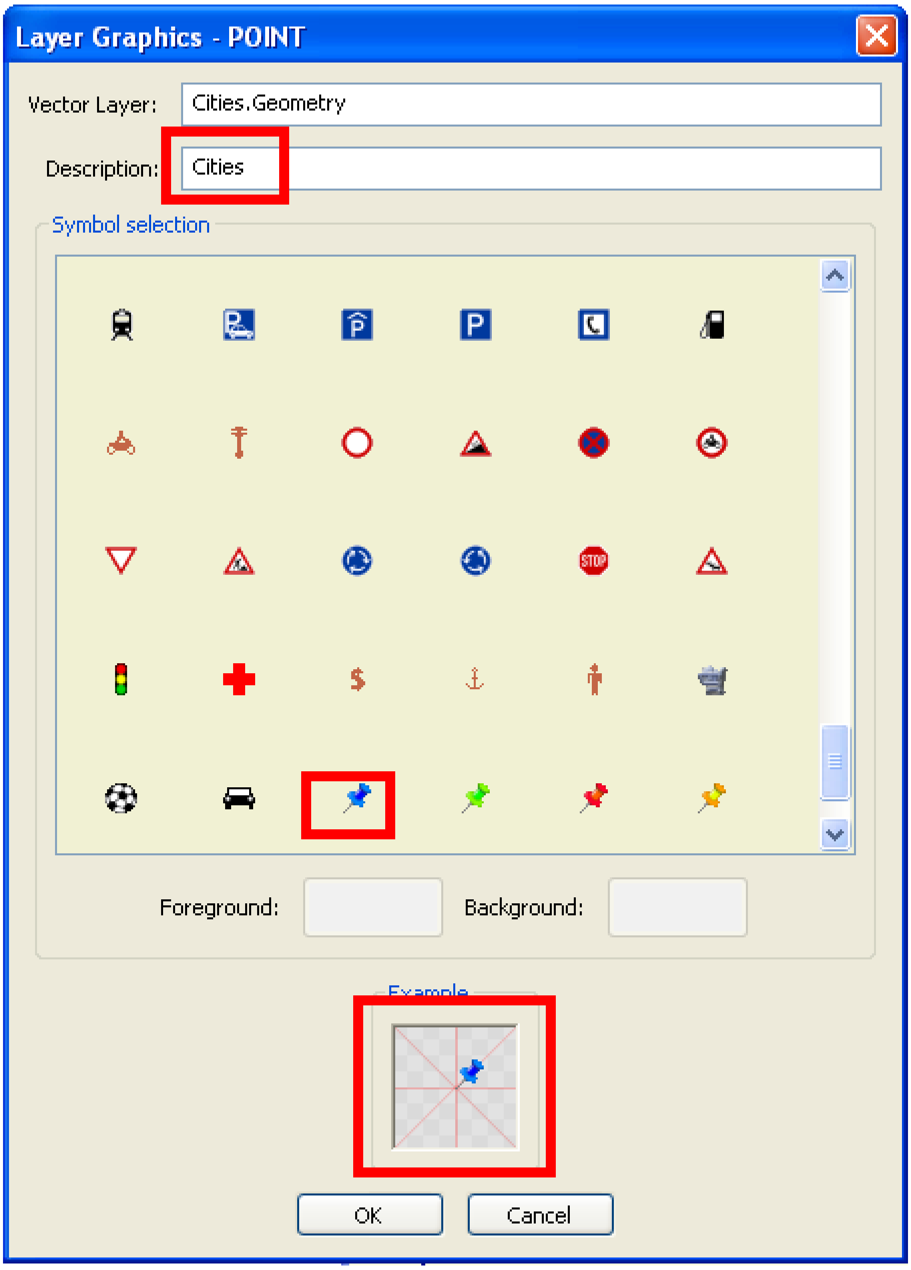
- Cities.Geometry
- Countries.Geometry
- Admin.Geometry

RefSys: 4326 [WGS 84]   Scale 1:307335817   80°44'57"S   170°43'54"W

In order to set your preferred **presentation ordering** for the various themes / layers, you simply have to **drag** the tree control's items.



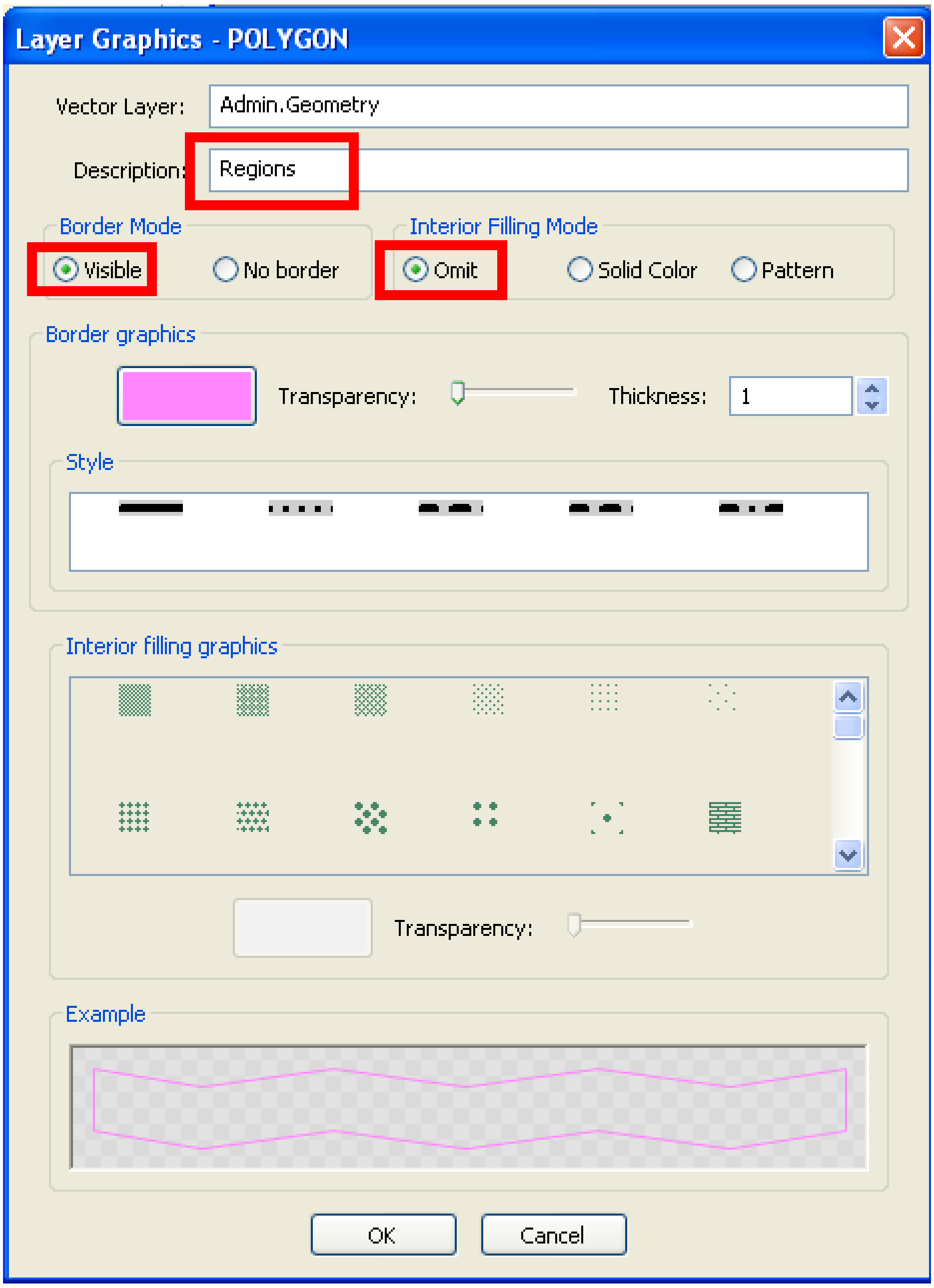
The next step is obviously the one to set an optimized configuration for any theme / layer
We'll start out trip customizing the **Graphic configuration**.



Layer Graphics - POINT

Vector Layer: Cities.Geometry

Description: Cities



Layer Graphics - POLYGON

Vector Layer: Admin.Geometry

Description: Regions

Border Mode: ● Visible ○ No border

Interior Filling Mode: ● Omit ○ Solid Color ○ Pattern

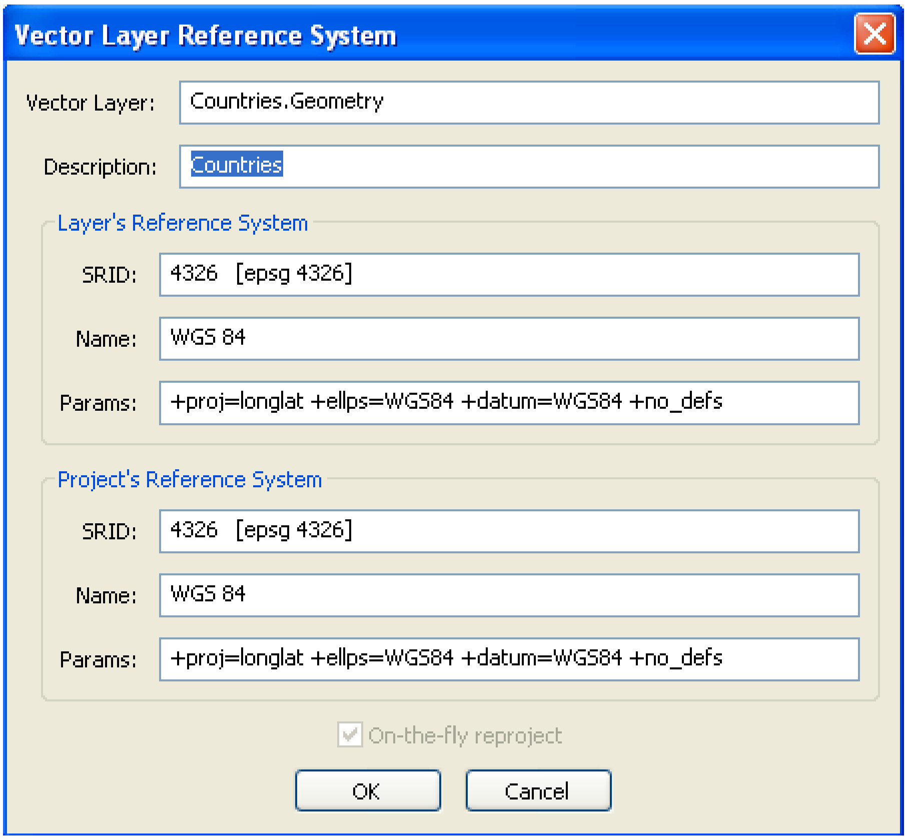
Border graphics

Transparency:

Thickness: 1

Obviously the graphic configuration details are quite different depending on the Geometry Class specifically interested. Anyway, you can easily learn by trial and error anything you have to know about.

You are warmly suggested to set some useful and meaningful **layer's name**: you have the opportunity to set it using quite any layer-related dialog (*and there are lots of such dialogs*)



Vector Layer Reference System

Vector Layer: Countries.Geometry

Description: Countries

Layer's Reference System

SRID: 4326 [epsg 4326]

Name: WGS 84

Params: +proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs

Project's Reference System

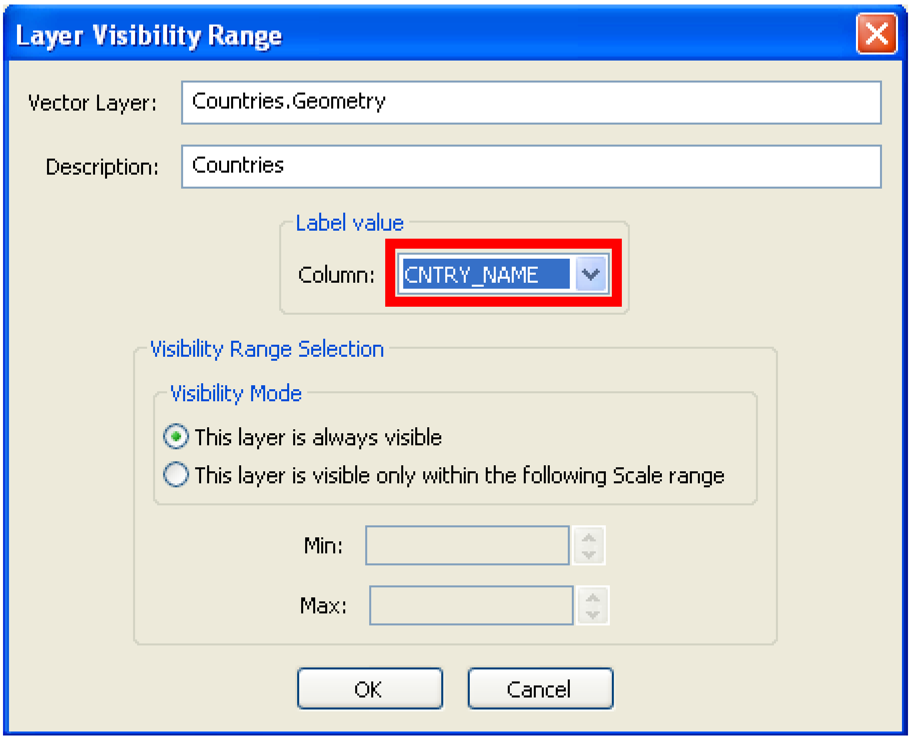
SRID: 4326 [epsg 4326]

Name: WGS 84

Params: +proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs

☑ On-the-fly reproject

Another useful option is to check (or set) the **layer's own SRID**.
Do you remember ? There is a Project's general SRID as well: if both the layer and the project belong to the same SRID, there is no problem at all.
But when they use different SRIDs, it may be really useful activating the **on-the-fly coordinate reprojection** feature.



Layer Visibility Range

Vector Layer: Countries.Geometry

Description: Countries

Label value

Column: CNTRY_NAME

Visibility Range Selection

Visibility Mode

● This layer is always visible

○ This layer is visible only within the following Scale range

Min:

Max:

You can set some appropriate **conditional visibility range** for your layers.
And you can set the **label column** as well, i.e. the column containing some clearly understandable name making easy to identify each individual entity.

It's very alike this item will be placed in a more appropriate dialog in future releases to come.

All right: now we've properly set our own custom project's layout. i.e., since now we have set:

- the general project's attributes
- the layers' presentation order
- the layers' specific attributes [*graphics, conditional visibility range, label column* ...]

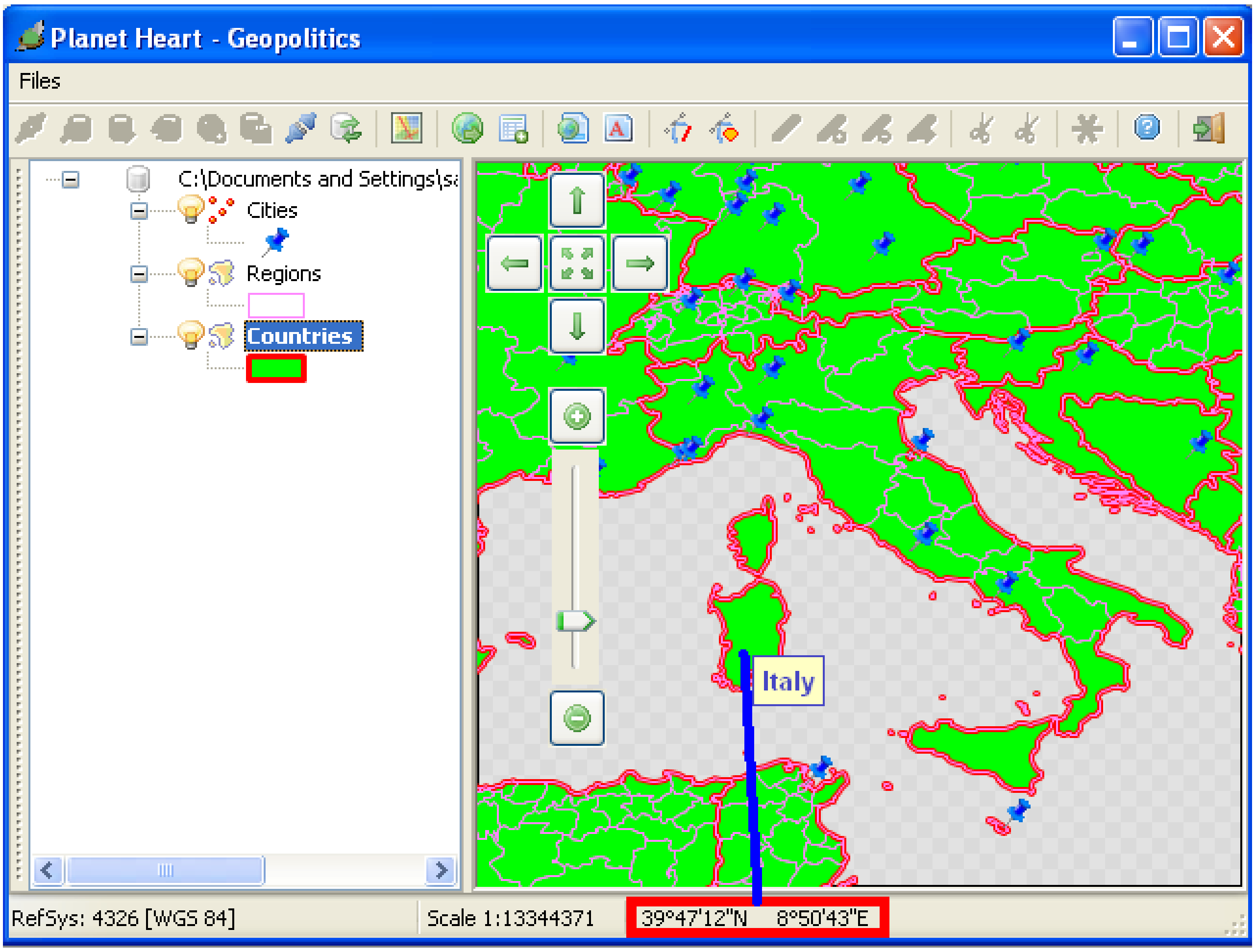
All this miscellaneous settings are now persistently stored within the DB itself. i.e., you can now transfer this DB to a different location, may be using a completely different OS, and your preferred project's settings will immediately be available, because they are directly stored into the same DB.

Hint: start the companion app **spatialite-gui**, and check to following DB tables:

- `layer_params`
- `layer_statistics`
- `layer_sub_classes`
- `layer_table_layout`
- `pattern_bitmaps`
- `symbol_bitmaps`

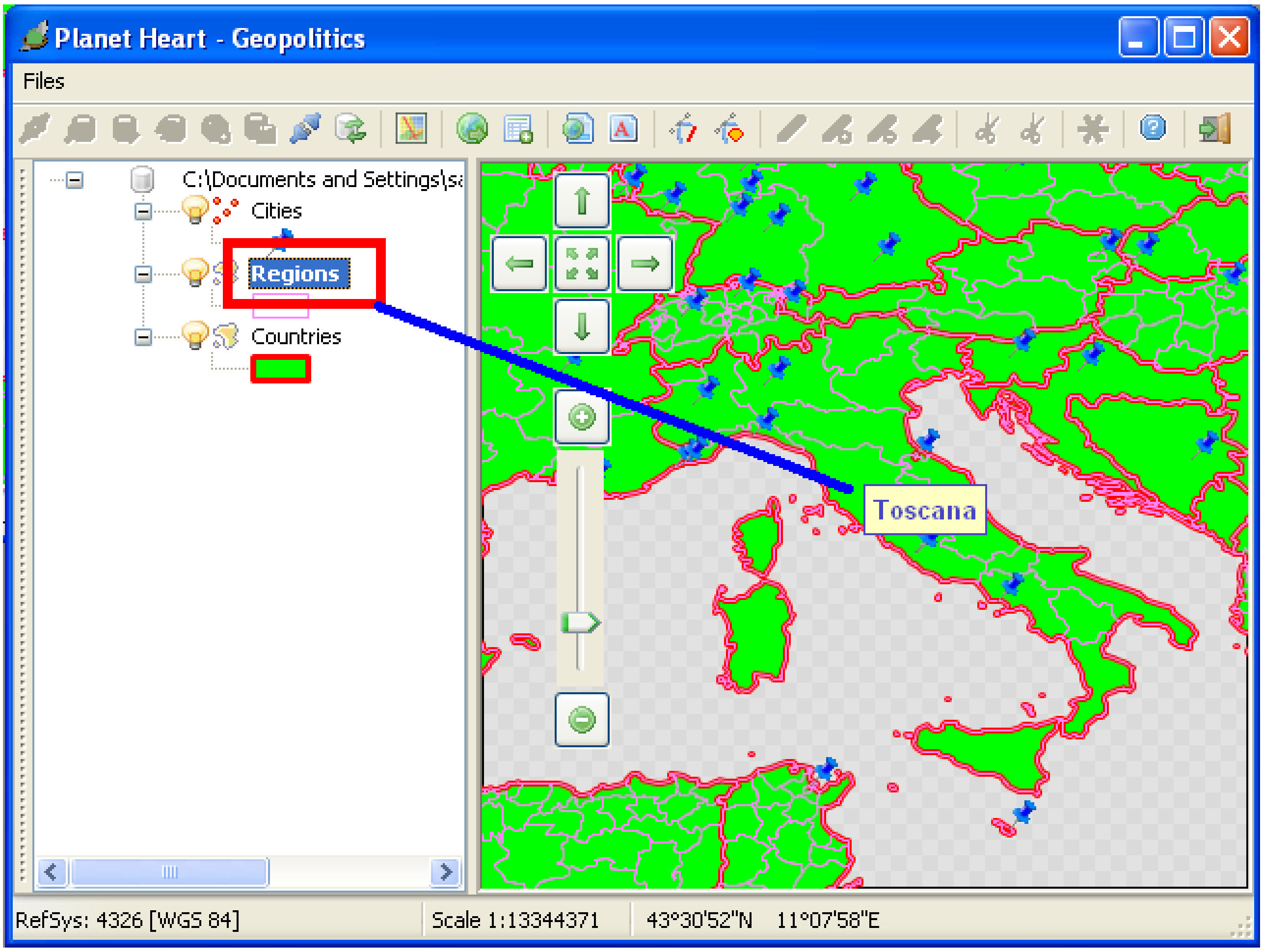
the whole project's configuration [*and any related stuff*] is stored within these tables.

4 – Map navigation

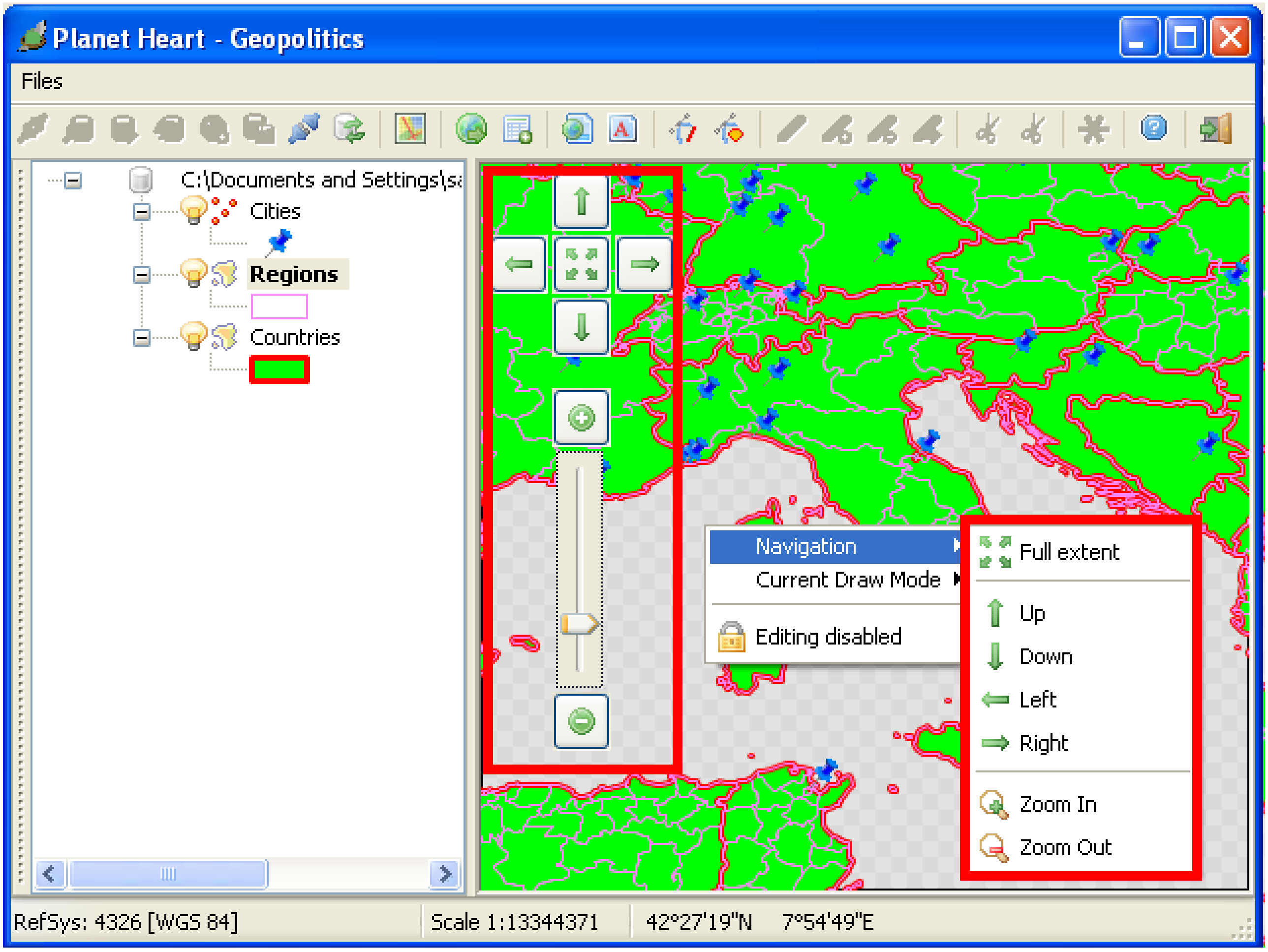


All right: once you've properly set your preferred project's layout (i.e. *the one you found more useful and practical for your specific purposes*), you are ready to begin your first **map navigation** experience.
You'll immediately notice that:

- map coordinates corresponding to the current cursor position are shown on the bottom status bar
- if the **selected layer** supports labeling, a small **tip** will dynamically identify the entity under the cursor cursor position. This is because **spatialite-gis** implements an **on-the-fly-identify** feature.



Setting the currently **selected layer** is really easy: just click the corresponding **tree item**.



Navigation commands are really intuitive: the simplest way is to use the **map buttons** and the **zoom slider**.
Alternatively, you can use the **context menu** as well (*simply perform a* ***right button click*** *over the map to cause the context menu to be shown*).
And there is a third way too, i.e. the one to use **keyboard shortcuts**.

5 – Keyboard shortcuts

The following is the keyboard shortcuts' list available during **map navigation**:

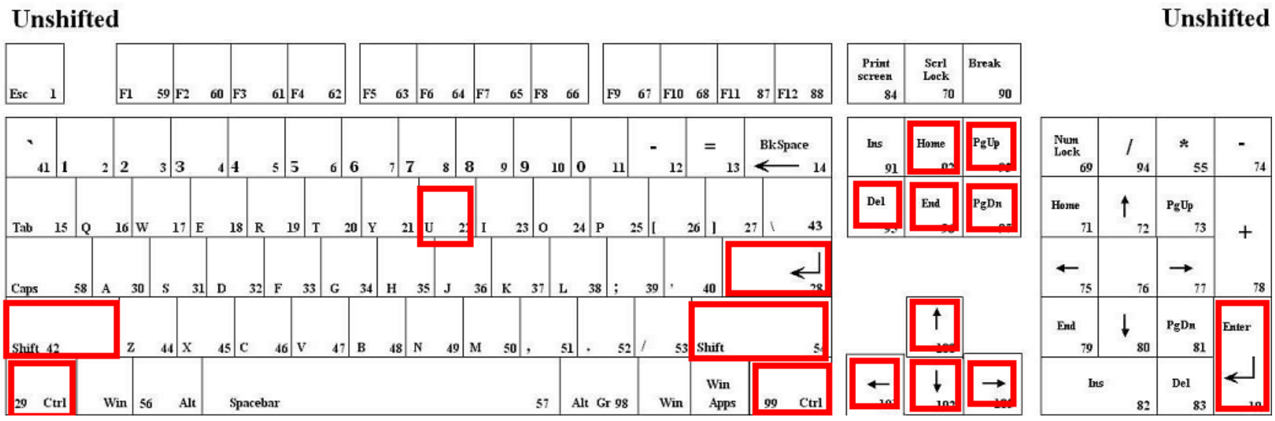
Key or Key combination	Function
HOME	Zoom to project's full extent
UP arrow key	Move the map viewpoint in the corresponding direction Moderate effect: useful for fine, precision positioning
DOWN arrow key	
LEFT arrow key	
RIGHT arrow key	
SHIFT + UP arrow key	Same as above Strongest effect: useful for quick, coarse positioning
SHIFT + DOWN arrow key	
SHIFT + RIGHT arrow key	
SHIFT + LEFT arrow key	
PAGE UP	Zoom In / Out Moderate effect: useful for fine, precision positioning
PAGE DOWN	
SHIFT + PAGE UP	Same as above Strongest effect: useful for quick, coarse positioning
SHIFT + PAGE DOWN	

You can use this keyboard shortcut to activate the Identify feature:

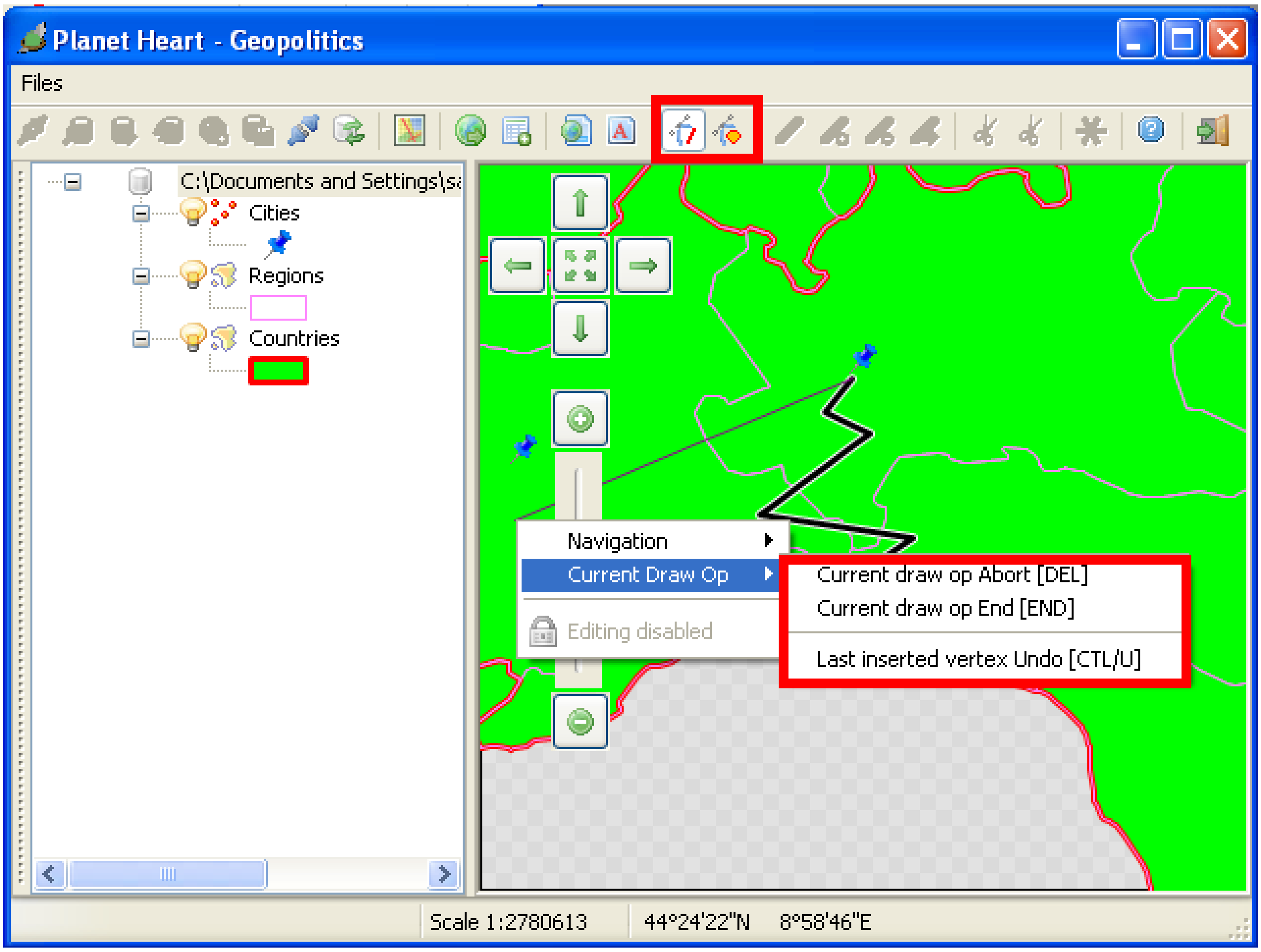
Key or Key combination	Function
RETURN	Identifies the entity (belonging to the currently selected layer) spatially selected by the current mouse positioning over the map

And the following keyboard shortcuts are available during any **drawing operation**:

Key or Key combination	Function
CTRL + U	Undo Removes the latest digitized vertex from the current draw You can recursively use the UNDO function: i.e., pressing three times CTRL+U removes the three latest digitized vertices.
DEL	Aborts the current drawing operation, with no further consequence
END	Terminates and positively confirms the current drawing operation Usually, this implies an INSERT or an UPDATE op to be performed.



6 – Measure tools



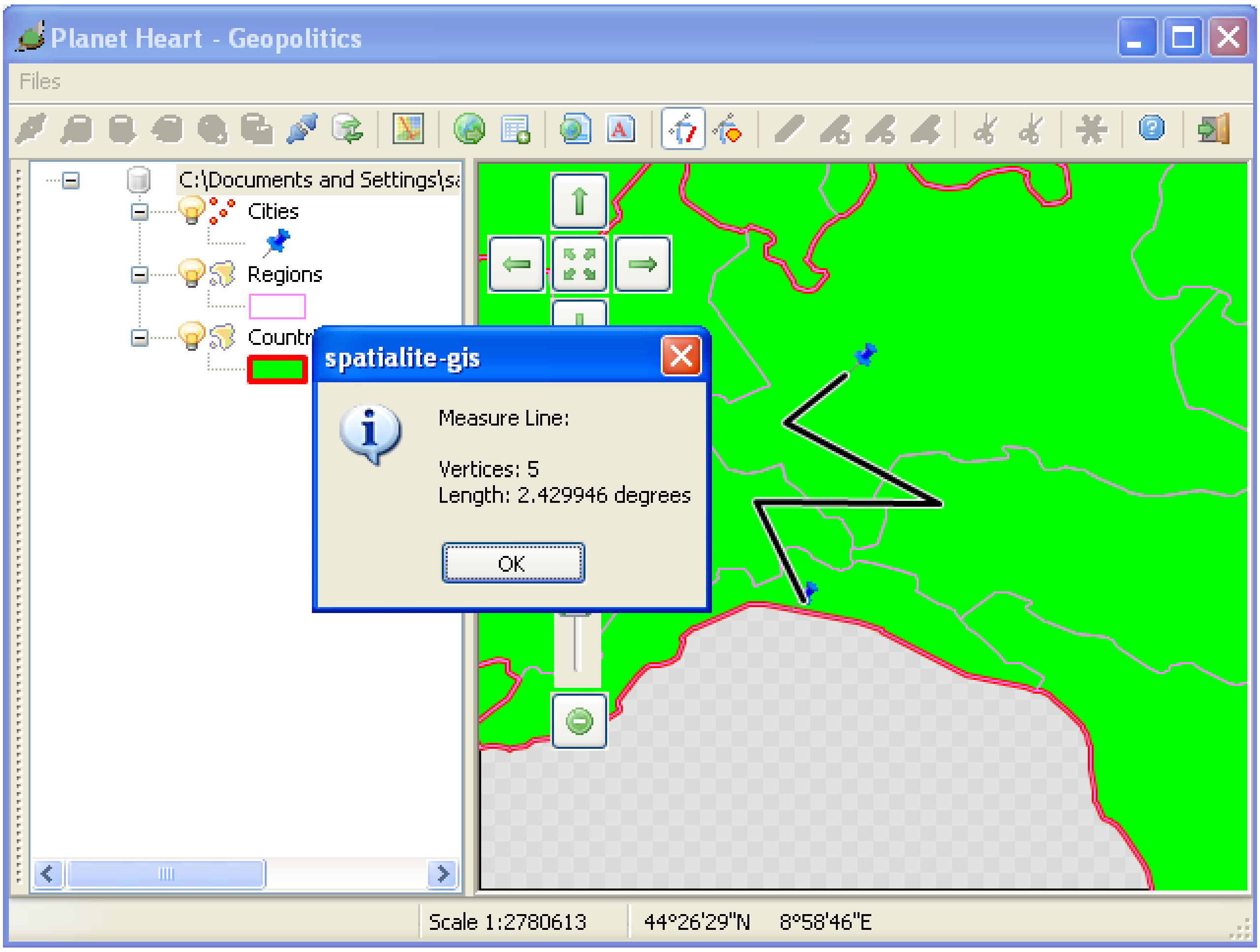
Planet Heart - Geopolitics
Files
C:\Documents and Settings\s
Cities
Regions
Countries
Navigation
Current Draw Op
Editing disabled
Current draw op Abort [DEL]
Current draw op End [END]
Last inserted vertex Undo [CTL/U]
Scale 1:2780613
44°24'22"N
8°58'46"E

Let now see how we can **take geometric measures** over the map. To do such a thing we can use:

- the **measure length** tool
- and/or the **measure area** tool

And this offers us the opportunity to examine how drawing ops does really work in **spatialite-gis**:

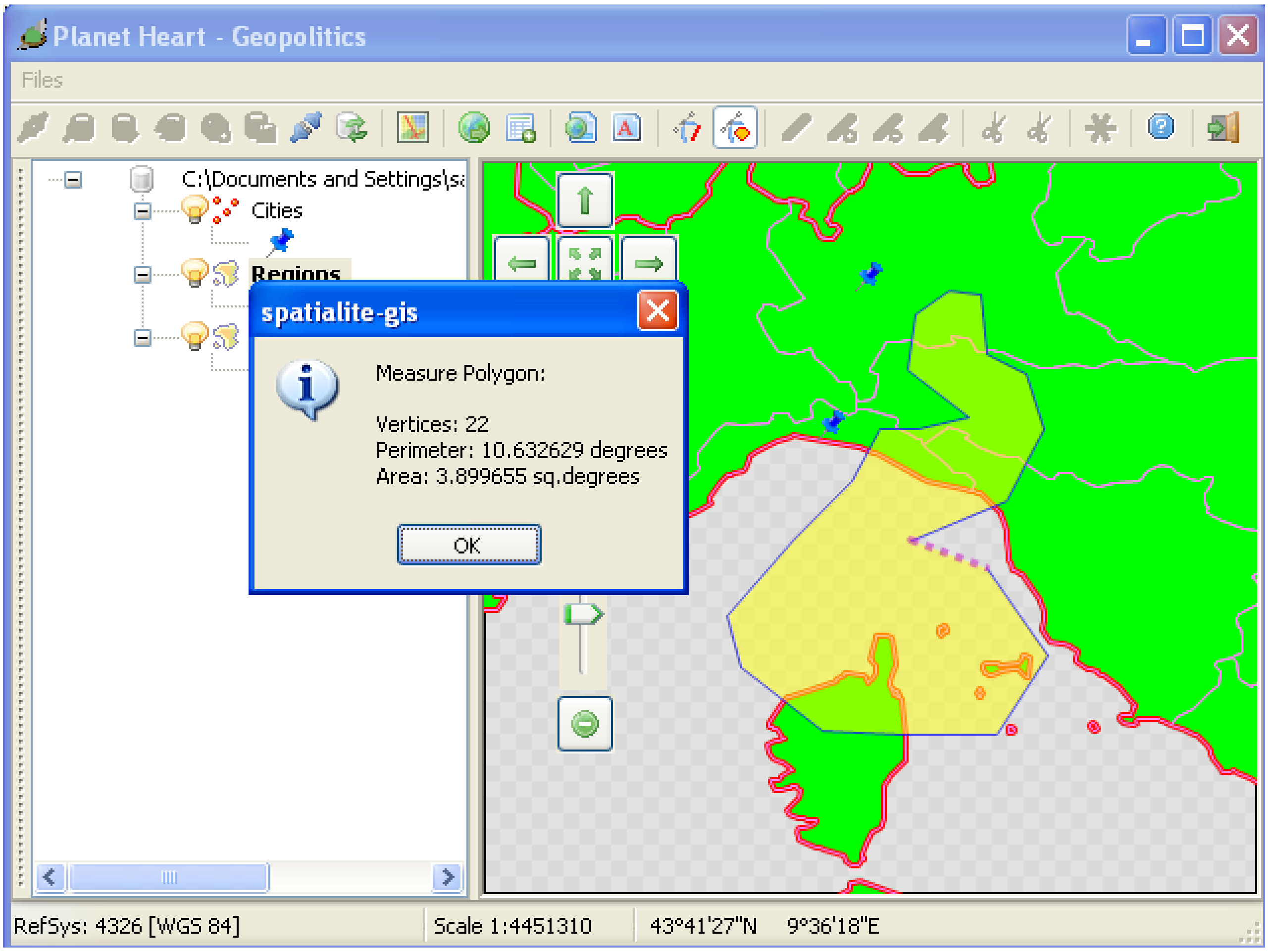
- begin any draw op using some corresponding tool (the measure length and area tools actually belongs to this family)
- then perform a **left button mouse-click** to set any vertex you intend to insert into your draw
  - when drawing a **line**, no special task is performed, so you'll simply see your line grow and accumulate subsequent vertices, as the draw goes on.
  - when drawing a **polygon**, you'll instead notice that a fictious segment will be automatically added at each time, in order to ensure the draw to be a **closed figure** anyway.
- you can use **keyboard shortcuts** or the **context menu** to perform special tasks (e.g. UNDO or ABORT).
- Once you've completed your draw, simply press the **END** key to confirm.



Measure Line:

Vertices: 5
Length: 2.429946 degrees

This one is an arbitrary example of **measure length** result

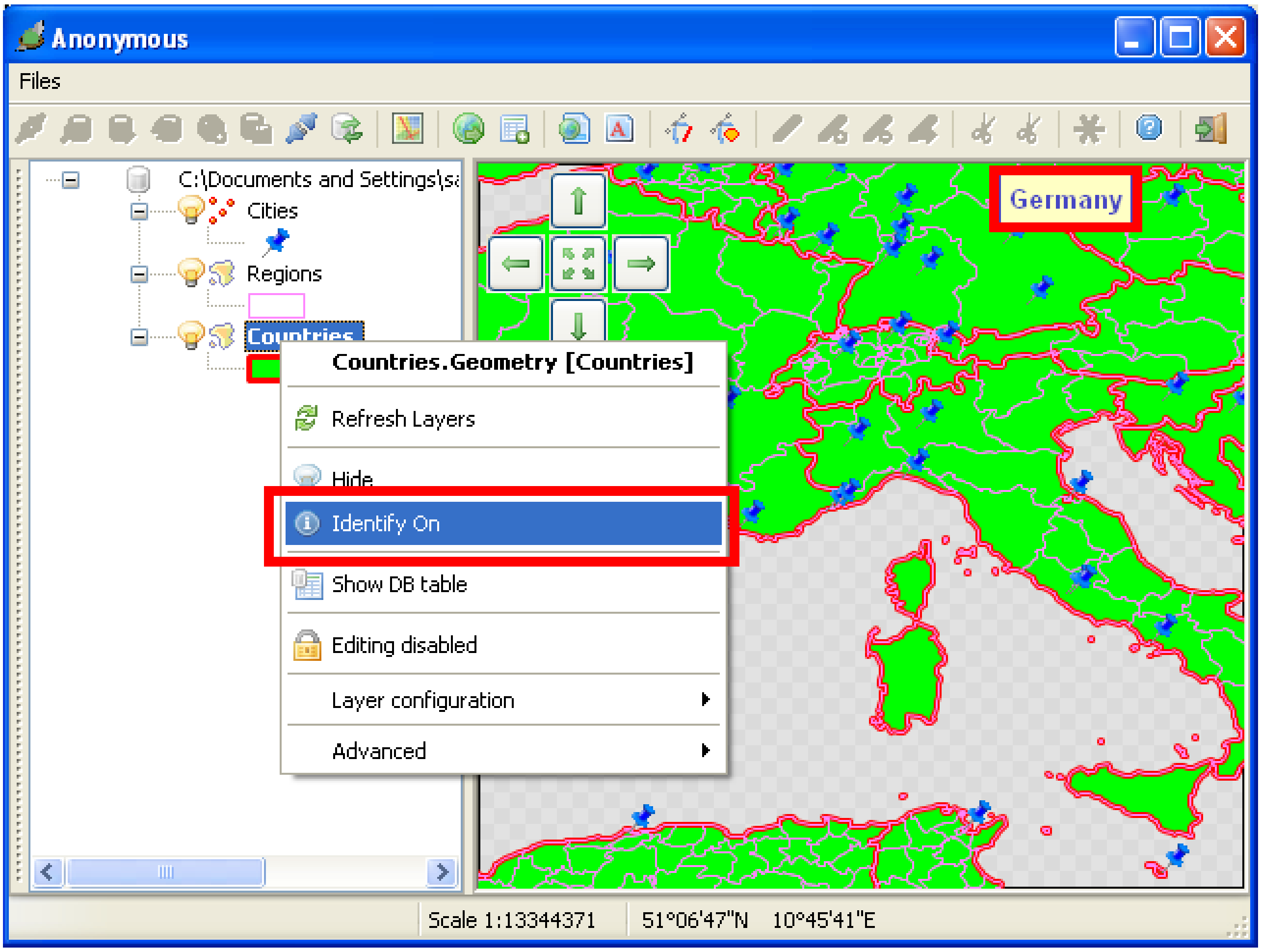


Measure Polygon:

Vertices: 22
Perimeter: 10.632629 degrees
Area: 3.899655 sq.degrees

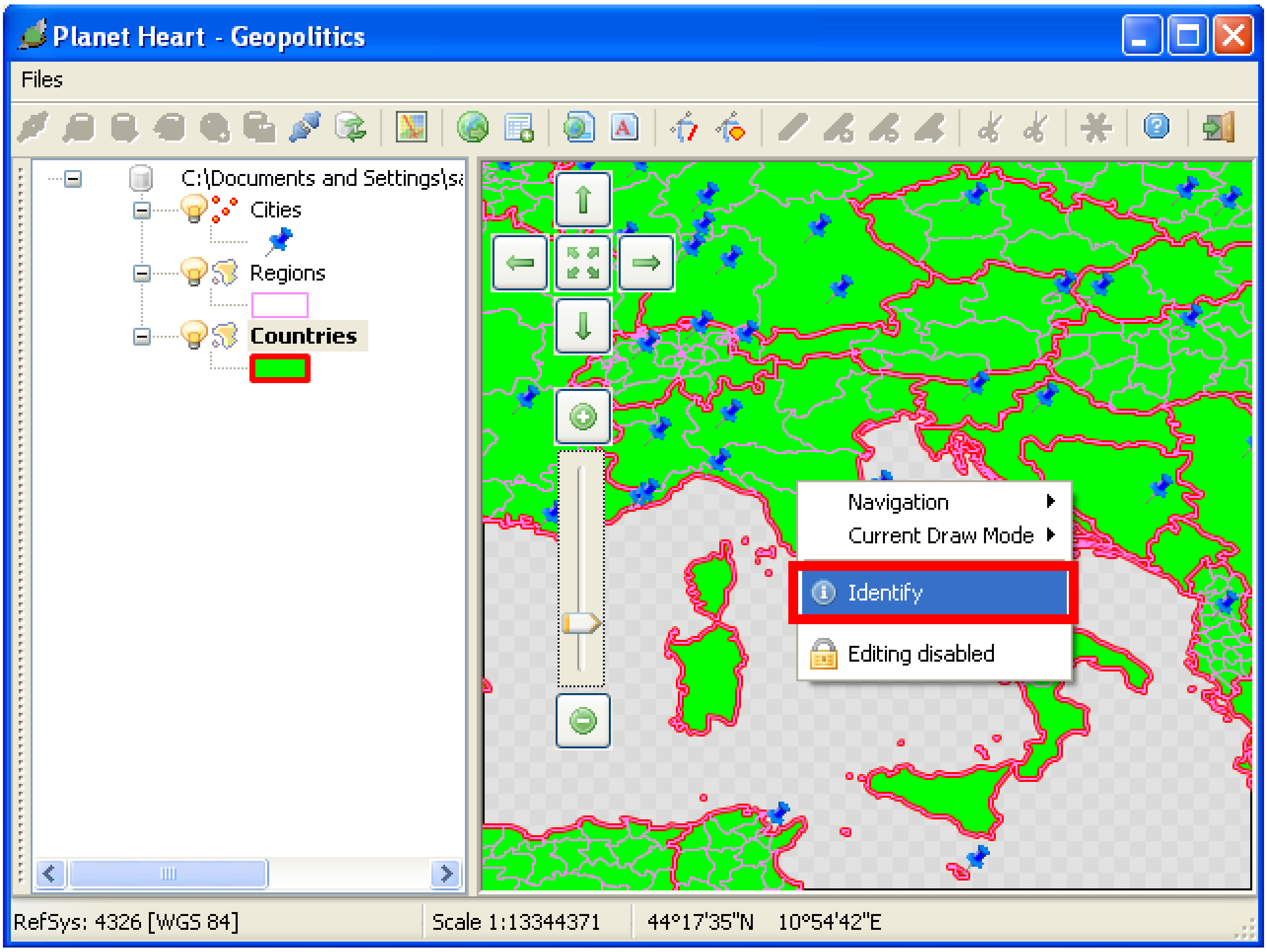
And this one is an arbitrary example of **measure area** result
Please note: the project we are currently using adopts **geographic coordinates** [**WGS 84**], and consequently the measures we've got are expressed in **degrees** and **square degrees**.
If you wish to obtain more conventional units (i.e. **meters**), you have to use some **planar** (*aka* ***projected***) **reference system** (e.g., *one belonging to the UTM family*).

7 – Identify



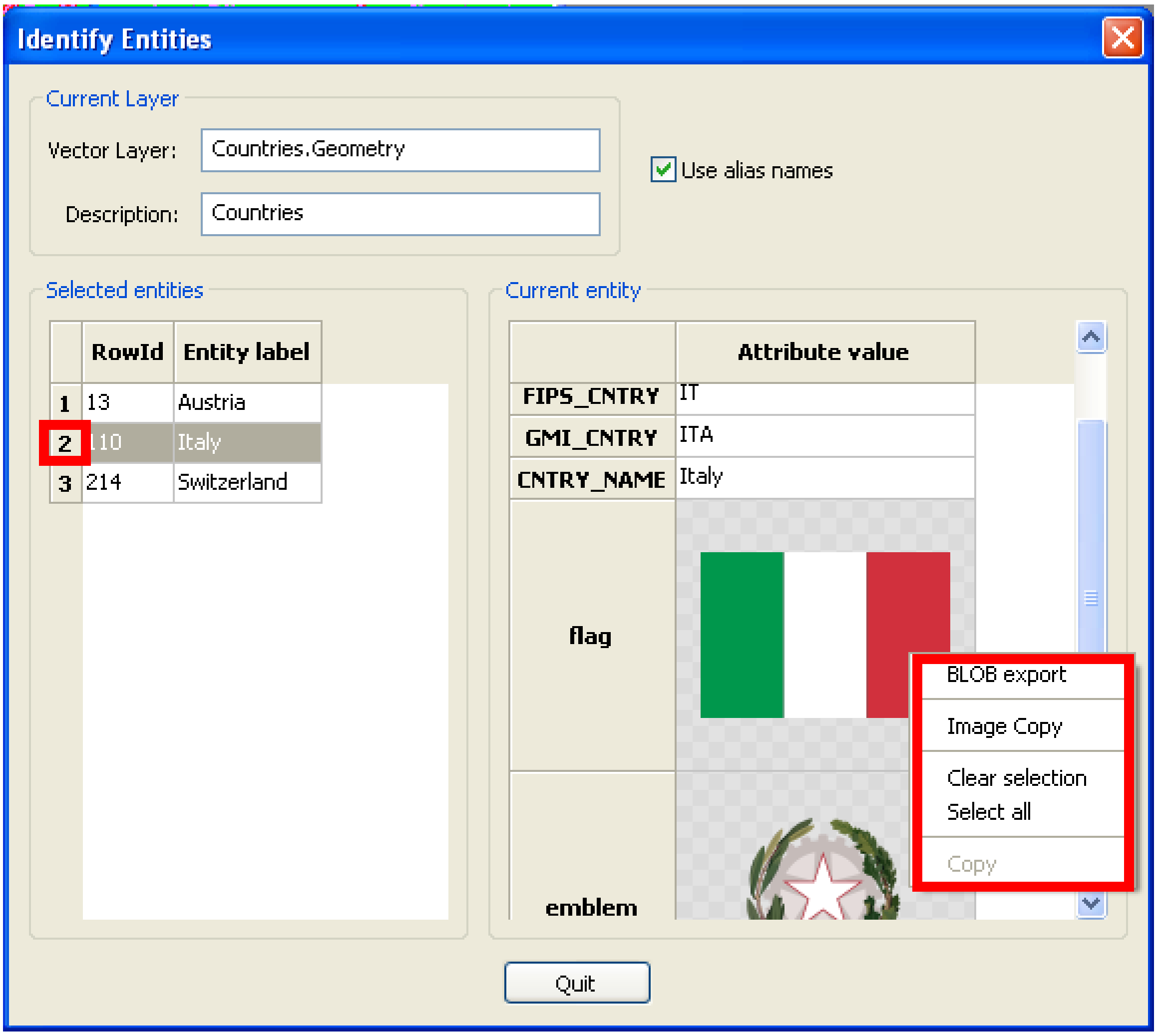
Usually the **Identify** feature is activated by default, but you can obviously disable this setting, if you wish to do so.

Once the Identify feature is activated for the currently **selected layer**, then an appropriate **label** will be dynamically shown over the map, accordingly with the current mouse positioning: we've already encountered this **on-the-fly-identify** feature.



In order to get the full attributes identifying any entity, you can use indifferently one of the following actions:

- simply **click the mouse left button**
- select the **Identify** item on the **context menu**: you can make the context menu to be shown clicking the **mouse right button**
- using a keyboard shortcut, i.e. pressing the **RETURN** key



	RowId	Entity label
1	13	Austria
2	110	Italy
3	214	Switzerland

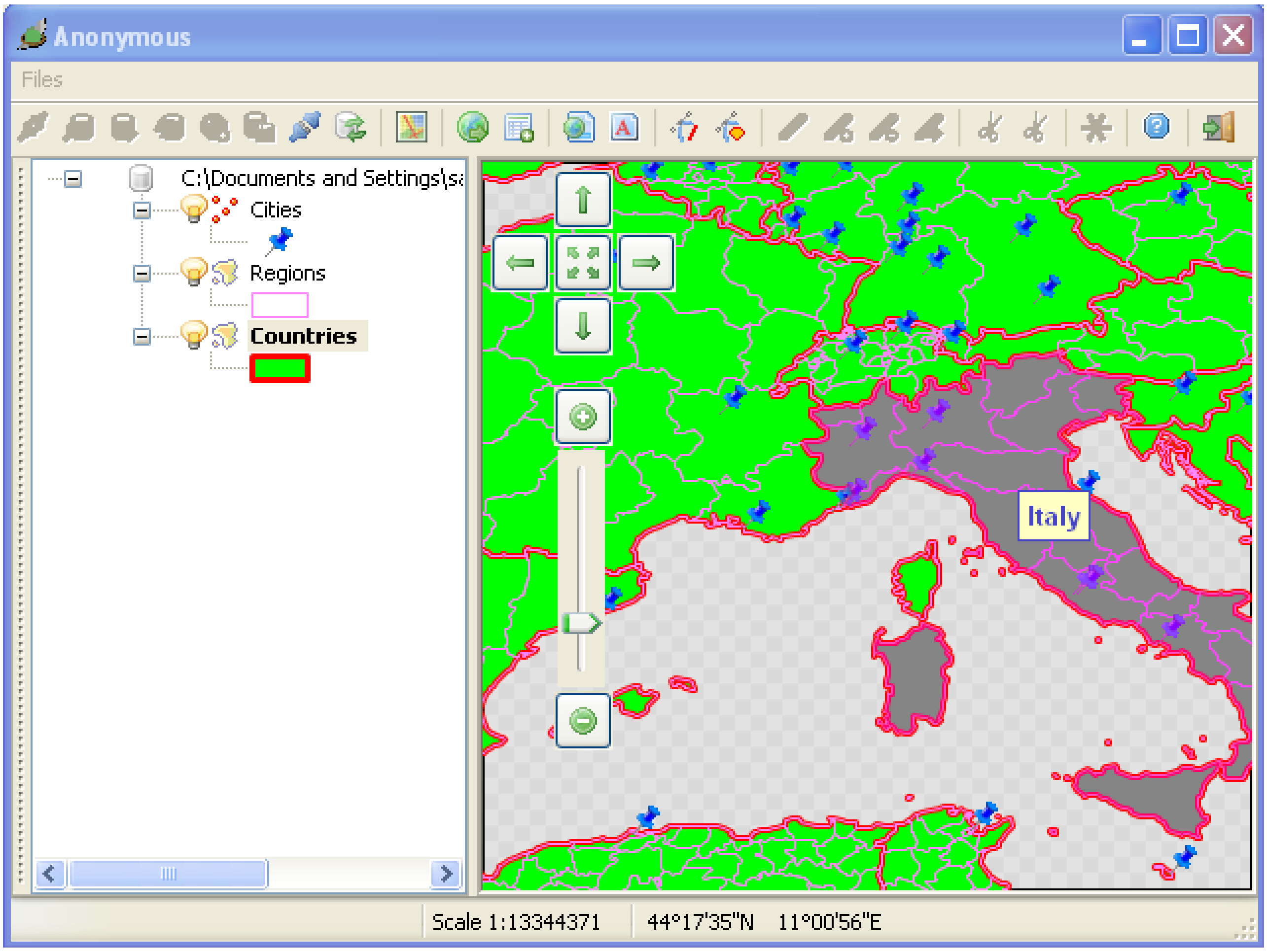
	Attribute value
FIPS_CNTRY	IT
GMI_CNTRY	ITA
CNTRY_NAME	Italy

The Identify operation may eventually select more entities (i.e., not a single one), because it works using a spatial proximity criterion. Accordingly to this, the Identify panel:

- on the leftmost side the Selected Entities list is shown
- on the rightmost side the Attributes List belonging to the Selected Entity is shown

You can simply select the specific Entity of your interest clicking the corresponding item on the list

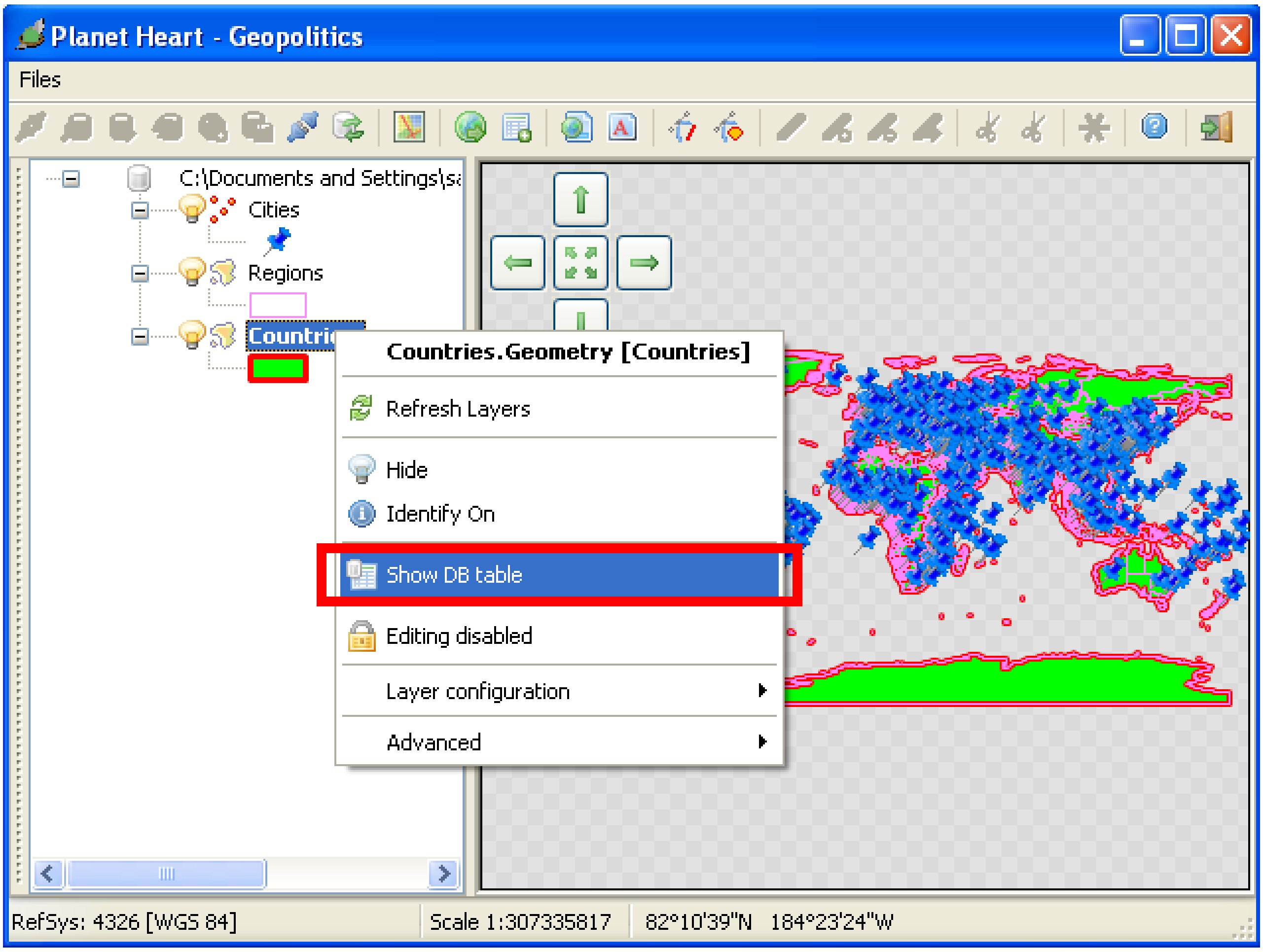
Please note: **spatialite-gis** actually handles **images** as well as they simply where any other ordinary value type: actually a **thumbnail** is shown, instead of the full-resolution image.
You can easily **export** or **copy** the full-resolution image anyway: to do such a thing you simply have to use the **context menu** functions.



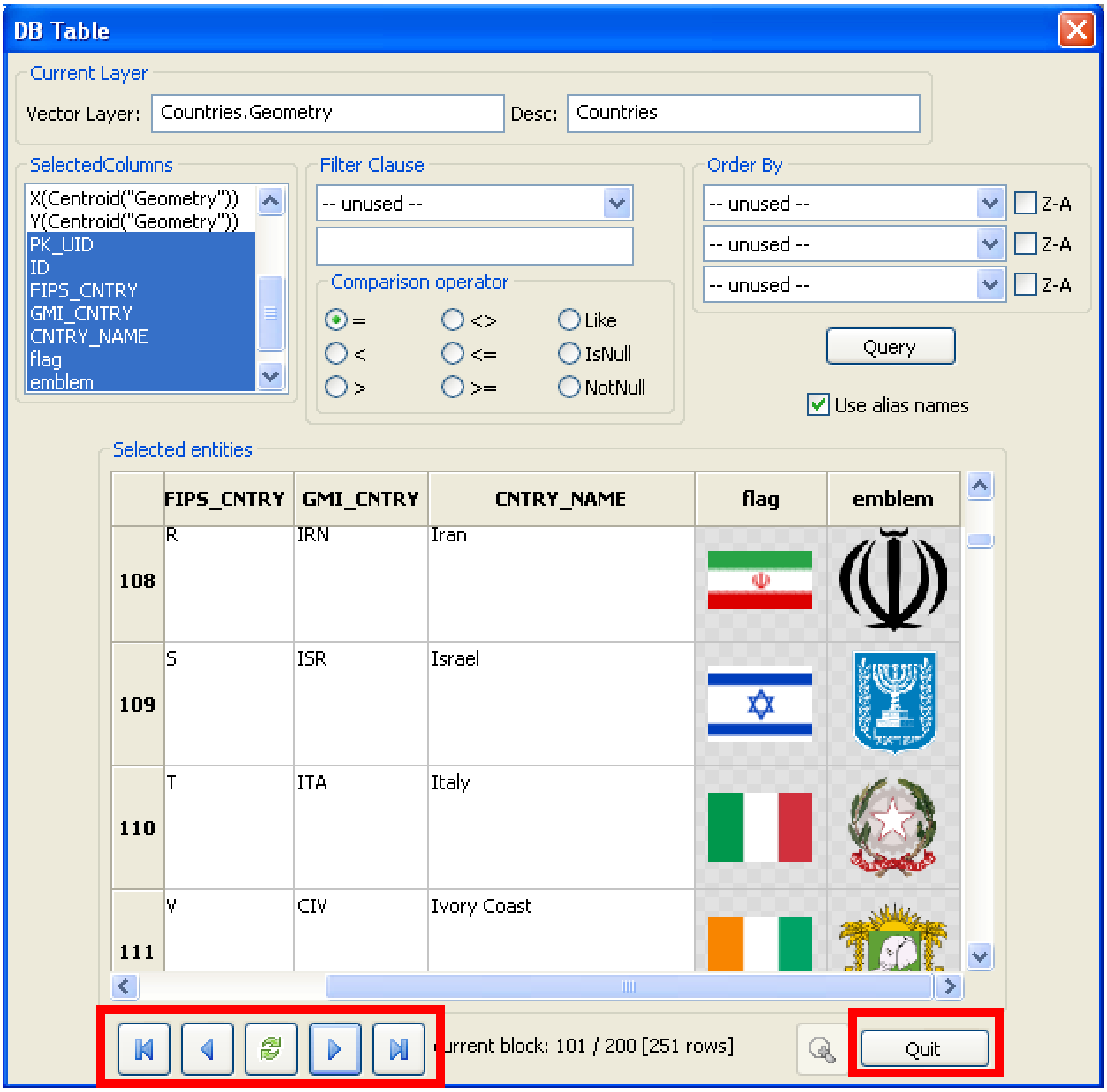
Scale 1:13344371   44°17'35"N   11°00'56"E

And that's not all: each time you select some specific Entity on the Identify panel, then the corresponding Geometry will be highlighted on the Map (***blinking***).

8 – Table querying



spatialite-gis supports a second, alternative way allowing to search, query and identify entities, i.e. the **show DB table function**.



DB Table

Current Layer

Vector Layer: Countries.Geometry Desc: Countries

SelectedColumns

X(Centroid("Geometry"))
Y(Centroid("Geometry"))
PK_UID
ID
FIPS_CNTRY
GMI_CNTRY
CNTRY_NAME
flag
emblem

Filter Clause: -- unused --

Comparison operator

● = ○ <> ○ Like
○ < ○ <= ○ IsNull
○ > ○ >= ○ NotNull

Order By

-- unused -- □ Z-A
-- unused -- □ Z-A
-- unused -- □ Z-A

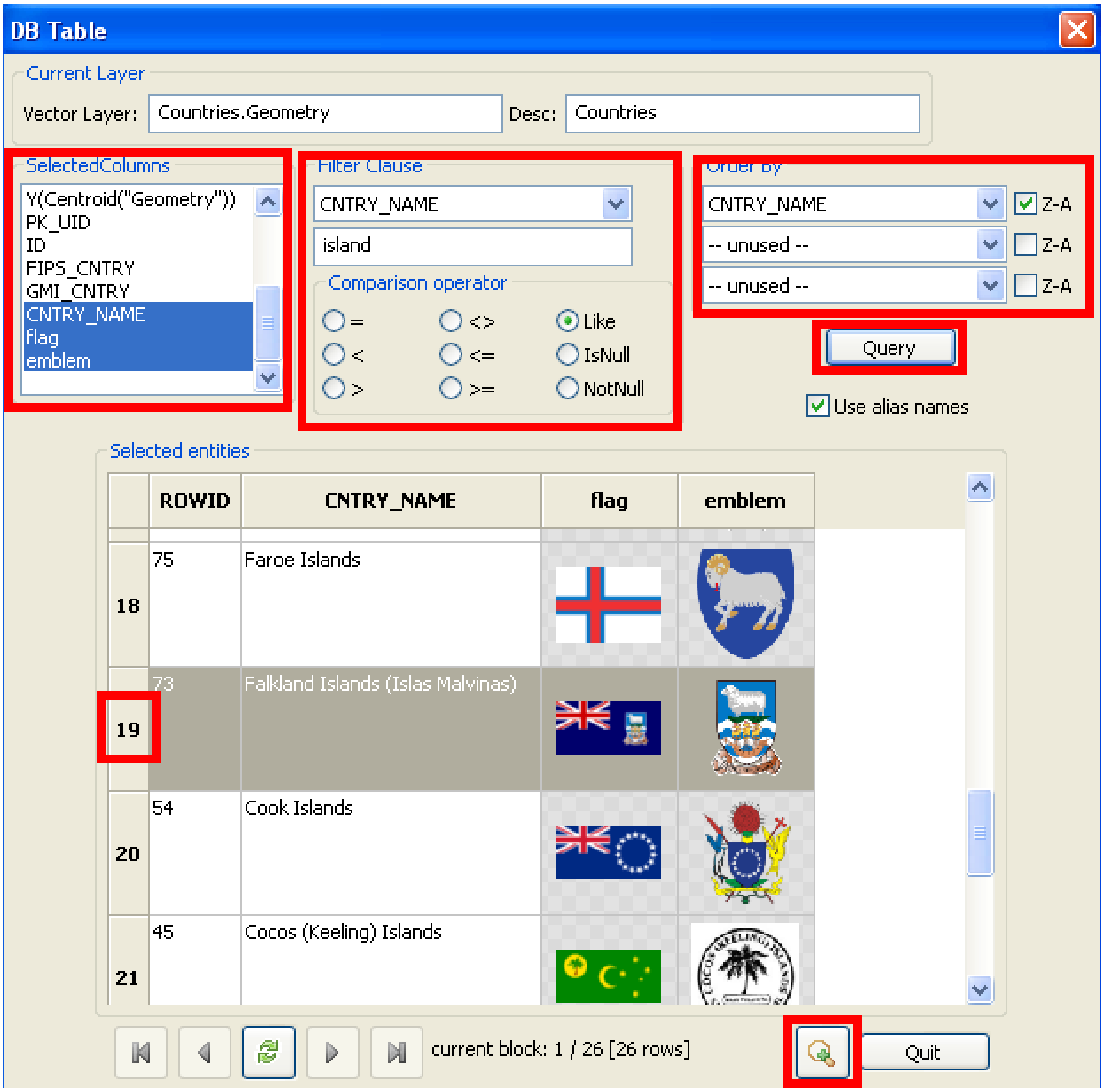
☑ Use alias names

Selected entities

	FIPS_CNTRY	GMI_CNTRY	CNTRY_NAME	flag	emblem
108	R	IRN	Iran		
109	S	ISR	Israel		
110	T	ITA	Italy		
111	V	CIV	Ivory Coast		

urrent block: 101 / 200 [251 rows]

The simplest way to query a table is performing an SQL statement such as: **`SELECT * FROM table_name`** and this one is exactly the way that **query DB table** works, if you don't apply any other specific setting. The **result set** is shown as a grid, holding 100 rows at each time to avoid wasting huge memory amounts: you can navigate the result set using the buttons shown above (*may be you are already accustomed to this, because this is exactly the same way you can examine a result set when using the companion app* ***spatialite-gui***).



DB Table

Current Layer

Vector Layer: Countries.Geometry Desc: Countries

SelectedColumns

- Y(Centroid("Geometry"))
- PK_UID
- ID
- FIPS_CNTRY
- GMI_CNTRY
- CNTRY_NAME
- flag
- emblem

Filter Clause

CNTRY_NAME

island

Comparison operator

○ = ○ <> ● Like
○ < ○ <= ○ IsNull
○ > ○ >= ○ NotNull

Order By

CNTRY_NAME ☑ Z-A
-- unused -- □ Z-A
-- unused -- □ Z-A

Query

☑ Use alias names

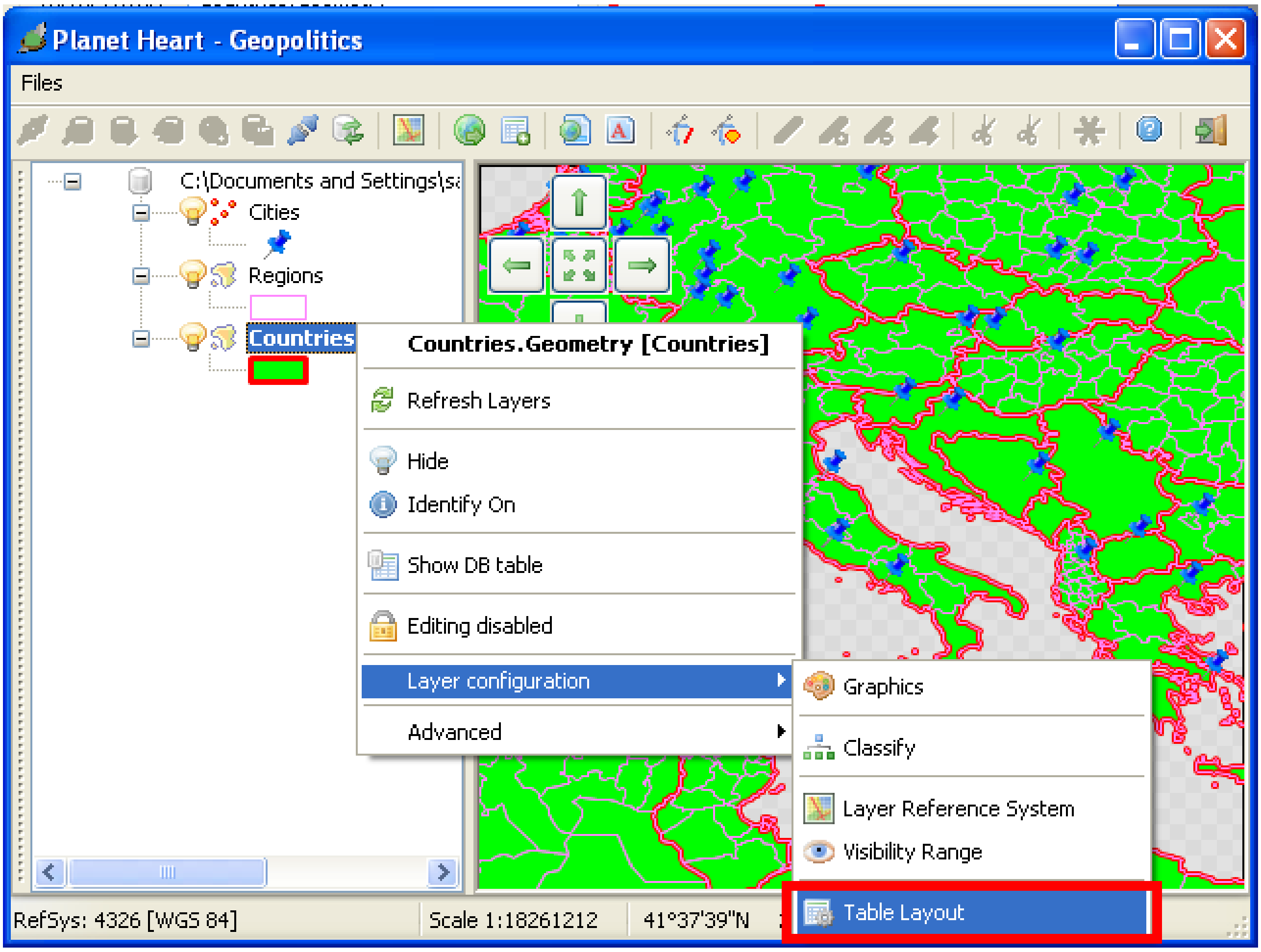
Selected entities

	ROWID	CNTRY_NAME	flag	emblem
18	75	Faroe Islands		
19	73	Falkland Islands (Islas Malvinas)		
20	54	Cook Islands		
21	45	Cocos (Keeling) Islands		

current block: 1 / 26 [26 rows]

You can flexibly customize the various query parameters (*filters*) in order to get more specific result sets. And you can select a specific entity and then automatically **zoom the map** in order to show that entity highlighted as a ***blinking*** one.

9 – Table layout



The architecture supported by the **SQLite SQL data engine** is a very peculiar one, in that **column data type** consistency and coherency is implemented in the most loose and weak way.
And that's not all: only the following five data types are supported by SQLite:

- **INTEGER** [*8, 16, 32 or 64 bits depending on value – corresponding to* ***TINYINT*** */* ***SMALLINT*** */* ***INTEGER*** */* ***BIGINT***]; this may be assumed to be equivalent to ***BOOLEAN*** as well
- **FLOAT** [*floating point – corresponding to* ***DOUBLE PRECISION***]; using the appropriate SQL functions, this may represent ***DATE*** or ***DATETIME*** as well [*Julian date format*]
- **TEXT** [*arbitrary length – corresponding to* ***CHAR*** */* ***VARCHAR*** */* ***TEXT***]
- **BLOB** [*arbitrary length – corresponding to* ***BLOB*** */* ***VARBINARY*** */* ***BYTEA***]
- **NULL**

So **spatialite-gis** supports a more detailed **table layout** definition, this allowing to use some more specific data type, enforcing data type coherency in input and get a cleaner data formatting.

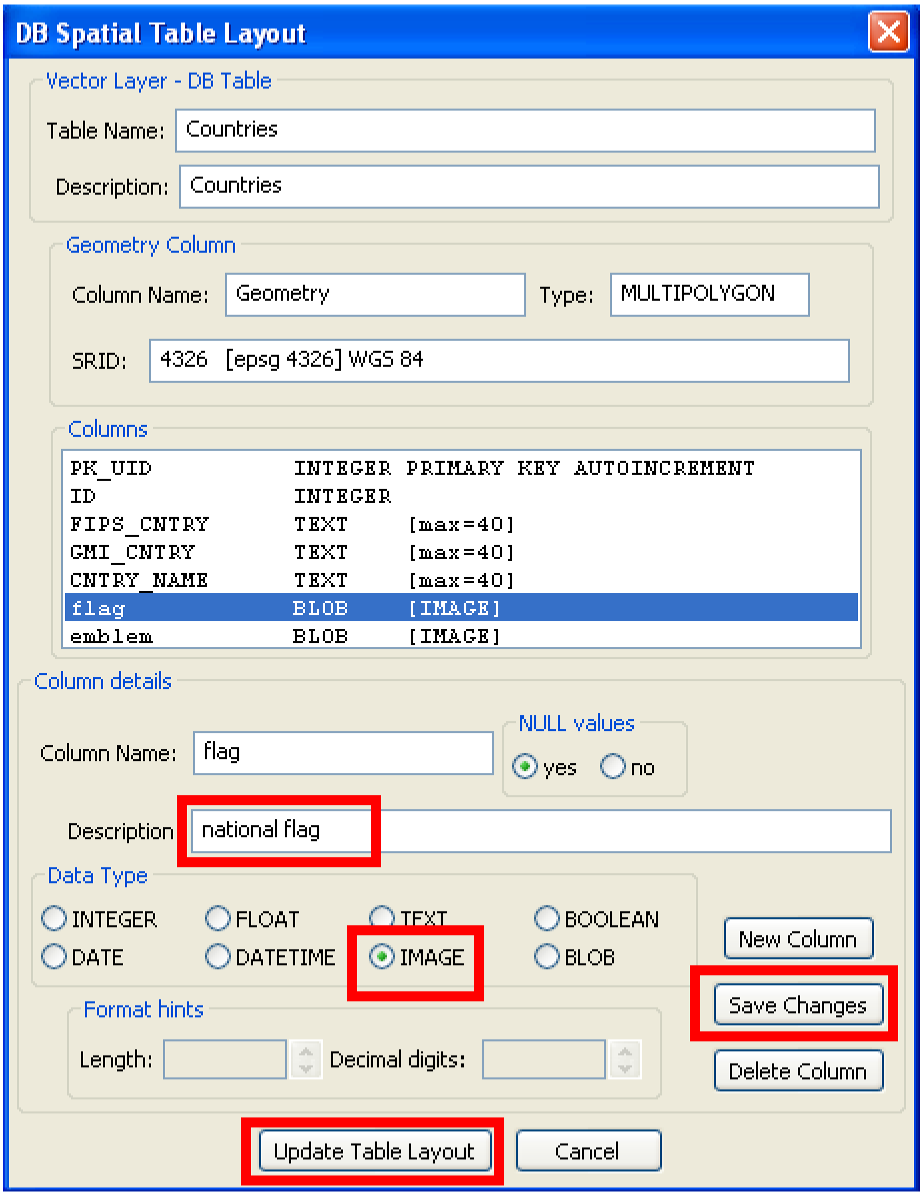


Table Name: Countries

Description: Countries

Column Name: Geometry Type: MULTIPOLYGON

SRID: 4326 [epsg 4326] WGS 84

```
PK_UID          INTEGER PRIMARY KEY AUTOINCREMENT
ID              INTEGER
FIPS_CNTRY      TEXT    [max=40]
GMI_CNTRY       TEXT    [max=40]
CNTRY_NAME      TEXT    [max=40]
flag            BLOB    [IMAGE]
emblem          BLOB    [IMAGE]
```

Column Name: flag

NULL values: ● yes ○ no

Description: national flag

Data Type: ○ INTEGER ○ FLOAT ○ TEXT ○ BOOLEAN ○ DATE ○ DATETIME ● IMAGE ○ BLOB

Length: Decimal digits:

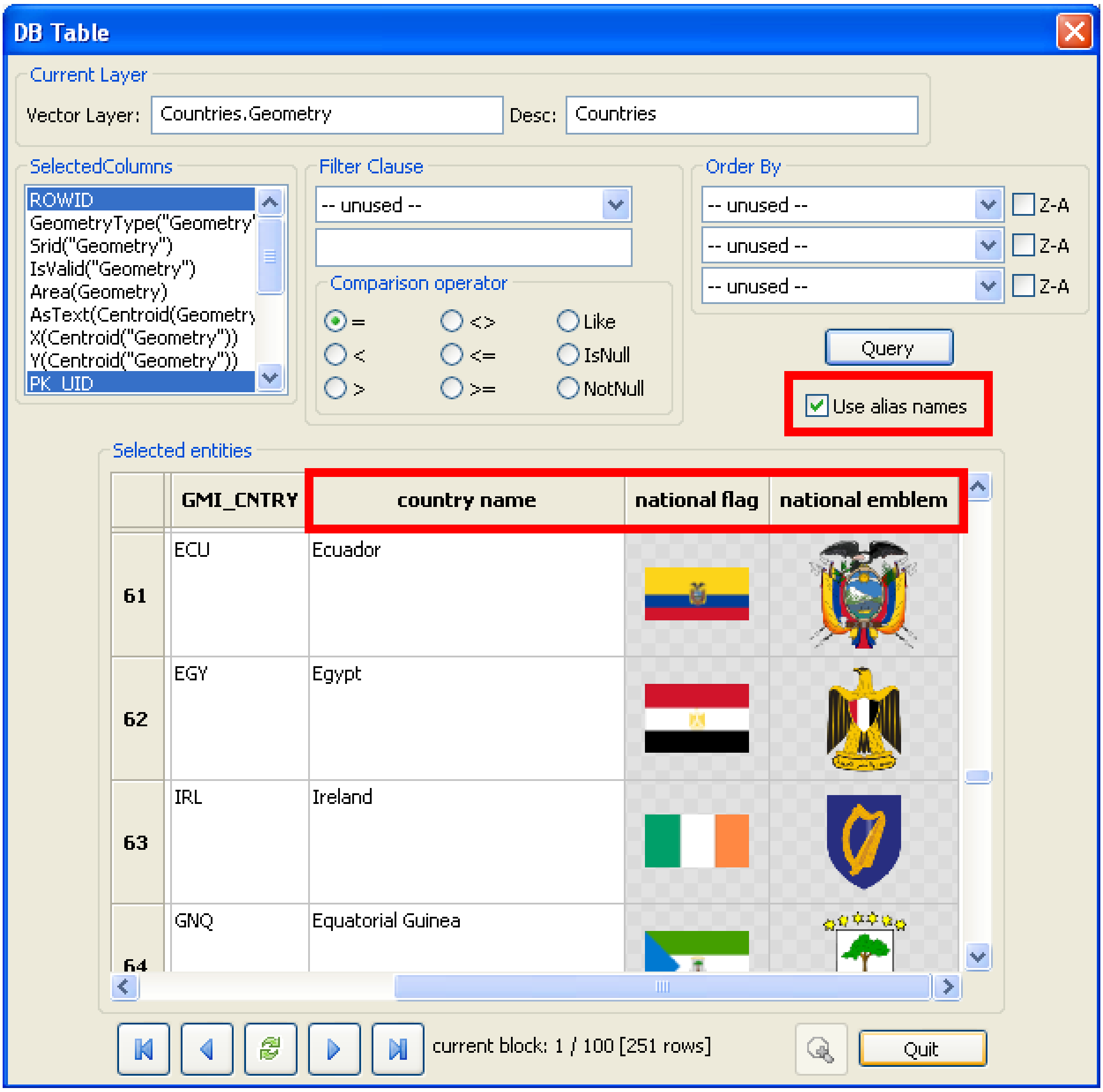
Using the **table layout** panel you can:

- assign an **alias name** to each column, i.e. a more descriptive, clear and understandable one.
- set a more specific **data type** for each column, this including BOOLEAN, DATE, DATETIME and IMAGE pseudo-types as well.
- assign specific **format hints**, such max allowable length for text strings, or decimal positions for numbers.

Please note: you can also **add further columns** to an already existing DB table; this task is actually accomplished performing implicit **`ALTER TABLE ADD COLUMN`** … SQL statements as required.

And please note too: due to the explicit limitations of the SQLite SQL data engine, the followings operations are instead **unsupported** [i.e. *they aren't implemented at all*]:

- dropping an already existing column
- renaming an already existing column
- changing the NOT NULL handling for an already existing column
- setting the NOT NULL clause for a column added on a second time after the initial table creation.



DB Table

Current Layer

Vector Layer: Countries.Geometry Desc: Countries

SelectedColumns

ROWID
GeometryType("Geometry"
Srid("Geometry")
IsValid("Geometry")
Area(Geometry)
AsText(Centroid(Geometry
X(Centroid("Geometry"))
Y(Centroid("Geometry"))
PK_UID

Filter Clause: -- unused --

Comparison operator

● = ○ <> ○ Like
○ < ○ <= ○ IsNull
○ > ○ >= ○ NotNull

Order By

-- unused -- □ Z-A
-- unused -- □ Z-A
-- unused -- □ Z-A

Query

☑ Use alias names

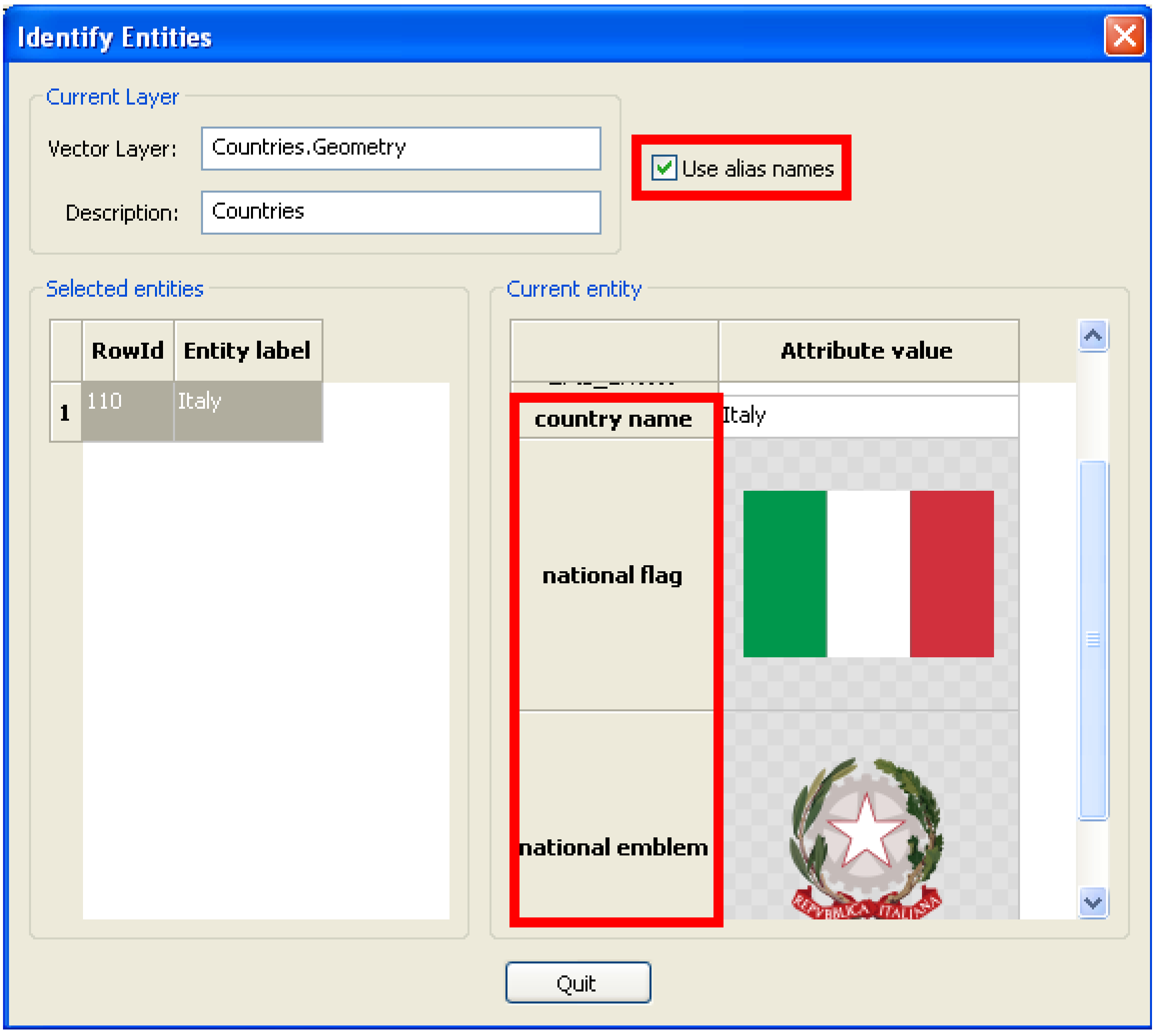
Selected entities

	GMI_CNTRY	country name	national flag	national emblem
61	ECU	Ecuador		
62	EGY	Egypt		
63	IRL	Ireland		
64	GNQ	Equatorial Guinea		

current block: 1 / 100 [251 rows]

Quit

Once you've set an appropriate, useful **alias name** for some table column [*attribute*], this one can be immediately seen on the **query DB table** panel.



Identify Entities

Current Layer

Vector Layer: Countries.Geometry

Description: Countries

☑ Use alias names

Selected entities

	RowId	Entity label
1	110	Italy

Current entity

	Attribute value
country name	Italy
national flag	
national emblem	

Obviously, the same thing happens on the **Identify** panel as well.