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

This section explains the **advanced** operations supported by **spatialite-gis** 

We'll assume you've already acquired a sound familiarity with **basic** operations.

#### Table of contents:

- 1 Layer management: extent / drop table / Spatial Index
- 2 Shapefile import / export
- 3 Creating a new DB table / layer
- 4 Attributes editing
- 5 Inserting new entities / drawing
- 6 Geometry editing: moving a Point or a Vertex
- 7 Geometry editing: deleting a Vertex
- 8 Geometry editing: interpolating a Vertex
- 9 Geometry editing: inserting a sub-geometry into a complex Geometry
- 10 Geometry editing: removing a sub-geometry from a complex Geometry

#### 1- Layer management: extent / drop table / Spatial Index

🥩 Planet Heart - Geopolitics	
Files	
///*********	🔟   🐵 🖪   🗟 🔺 🌴 🌾   🥒 🔏 🖧 🐇   🛠   🥹   到
C:\Documents and Sett	ings\si ies.Geometry [Cities]
	resh Layers
in the country in the	entify On
Sho	w DB table
	ing disabled er configuration
	anced
	Second Spatial Index Second Build Spatial Index Second Build MBR cache
	Export as Shapefile           Image: Comparison of the state of
	Delece (Tils Layer
RefSys: 4326 [WGS 84]	Scale 1:307335817 178°18'36"5 146°20'29"W

**spatialite-gis** allows you to perform various common utility task related with layer (*DB table*) management:

- you can **update the layer statistics**: this one is a batch, deferred task, specifically aimed to determine the **extent** for a single layer [*and the whole project's full extent*]. You are strongly suggested to update layer's statistics every time you've inserted, deleted or updated lots and lots of entities.
- you can create or remove a Spatial Index [either an R\*Tree one, or an MBR-cached one, at your free choice]. Please note: creating a Spatial Index for a layer storing less than some thousands entities doesn't make any speed improvement at all. Conversely, creating a Spatial Index for a layer storing some hundred thousands [or millions] entities make a dramatic speed improvement.
- and finally you can **drop** [*delete*] a complete layer. In this occurrence spatialite-gis take care of dropping the corresponding DB table and any further layer-related definition. Please note: this one is an **unrecoverable operation**; so, think twice before confirm !!!

#### 2 – Shapefile import / export

🥩 Planet Heart - Geopolitics	
Files	
〃〃♀♀♀♀♀》♀  ∑  ④ 3   ④	🗚   🗄 🌾 🖌 K K K K 😽 😰 🛃
C:\Documents and Settings\sa	
Cities.Geometry [Cities]	
🗐 🖓 🕅 Refresh Layers	▶
🖃 🐨 🛜 Cou 🍚 Hide	
(1) Identify On	
Show DB table	
Editing disabled	
Layer configuration	
Advanced	🛂 📊 Update Layer Statistics
	💩 Build Spatial Index
	🍥 Build MBR cache
	🛞 Export as Shapefile
	🚮 Delete this Layer
RefSys: 4326 [WGS 84] Scale 1:307335817	' 175°50'10"N 189°16'05"W

Importing and / or exporting GIS datasets using the popular and widespread **Shapefile** format usually is a common task: so **spatialite-gis** allow you to perform both operations.

Load Shapefile		×
Path:	C:\Documents and Settings\sandro\Desktop\GisTest\TEST	
Table name:	TEST	
GeomColumn name:	Geometry	]
ſ	Charset Encoding	
SRID: -1	BIG5-HKSCS:1999 BIG5-HKSCS:2001 CP850 DOS/OEM Western Europe CP862 DOS/OEM Hebrew CP866 DOS/OEM Cyrillic CP874 DOS/OEM Thai CP932 DOS/OEM Japanese CP936 DOS/OEM Chinese CP949 DOS/OEM Korean CP950 DOS/OEM Chinese/Big5 CP1133 Laotian CP1250 Windows Central Europe CP1251 Windows Cyrillic CP1252 Windows Latin 1 CP1253 Windows Greek	

While importing some shapefile you'll be required to select a **SRID** and a **charset encoding** [may be you already know everything about this, because the companion app **spatialite-gui** works exactly in same identical way]

#### 3 – Creating a new DB table / layer



spatialite-gis allows you to create from scratch a completely new layer [DB table].

Create DB Spatial Table
Vector Layer - DB Table
Table Name:
Description:
Geometry Column
Column Name: Geometry SRID: 4326
Geometry Type
POINT     OLINESTRING     OPOLYGON
Columns
PEID INTEGER PRIMARY KEY AUTOINCREMENT
Column details
Column Name: PkId
Description: Unique ID
Data Type
INTEGER OFLOAT OTEXT OBOOLEAN     New Column
Format hints Save Changes
Length: 1 Decimal digits: 1 Delete Column
Create Table Cancel

The **create DB table panel** is very similar to the **table layout panel** we've already seen while explaining basic operations.

Any Spatial Table found into an SQLite / SpatiaLite DB may be accessed as a layer by **spatialite-gis** The minimal requirements to define a valid Spatial Table are the followings:

- you've to select a **table name** [and, may be, an optional human readable name too]
- you've to select a geometry column name, specifying its geometry type and SRID.
- you've to define a **primary key** of the **numeric auto-increment** type [*this task is automatically performed*]
- then you can freely add any other column [*attribute*] as required. You can defer this task to a later time, because you can use the **table layout panel** to add columns to any already existing table.

### 4 – Attributes editing

🥩 Planet Heart - Geopolitics	
Files	
🖋 🗩 🖶 🖷 📽 📽 📚 国 🛛 📾 🛛 👘 🍖	/ & & A   & &   🛠   🕹   🛃
C:\Documents and Settings\sa Cities Cities Cities Countries Countries	
Countries.Geometry [Countries]	
🖉 Refresh Layers	
🐨 Hide	Contraction of the second
<ol> <li>Identify On</li> </ol>	
🖳 Show DB table	
🔒 Editing disabled	
Layer configuration	<u> </u>
Advanced	
Scale 1:307335817 17°20'04"5	188°17'33"W

By default **spatialite-gis** disables any editing operation. You have to explicitly enable editing.

Identify Entitie	25		
-Current Layer -			
Vector Layer:	Countries.Geometry		e alias names
Description:	Countries		
-Selected entitie:	s	Current entity	
	ntity label		Delete this entity
<b>1</b> 110 It.	aly		Attribute value
		PK_UID	110
		ID	110
		FIPS_CNTRY	IT
		GMI_CNTRY	ITA
		country name	Repubblica Italiana
		national flag	
		Quit	changes Restore

The most immediate way to edit an entity is the one to use the Identify panel. As you can easily notice, when editing mode is enabled, you are allowed to:

- **delete** the currently selected entity
- modify any attribute value
- make any change to a permanent one, i.e. **updating** the corresponding DB row
- and you can recall the original values using the **restore** button [*obviously, before actually saving changes*]

Vector Layer: Admin.Geometry Desc: Administrative regions													
ecter	dColumns —		Filter Clause		Orde	er By							
WID	icolamns	~	CNTRY_NAM	F		nused							
met	ryType("Geo	ometry'		L			×						
rid("Geometry") Italy unused V Z-/													
Area("Geometry") Comparison operator unused 💟 🗌 Z-A													
	Centroid("G oid("Geomet		•=	$0 \Leftrightarrow ($	Like								
entr	oid("Geomet	rý")) 👝	0<	0<= (	IsNull	Qu	Jery						
UID			U (O>	○>= (	NotNull	Vuse a	lias names						
5-	ected entitie												
-58		35											
	ROWID	PK_UID	FIPS_ADMIN	GMI_ADMIN	ADMIN_NAME	FIPS_CNTRY	GMI_CNTRY	-					
1	1013	1013	IT01	ITA-ABR	Abruzzi	IT	ITA						
2	1014	1014	IT02	ITA-BSL	Basilicata	IT	ITA						
3	1015	1015	IT03	ITA-CLB	Calabria	IT	ITA						
4	1016	1016	IT04	ITA-CMP	Campania	IT	ITA						
5	1017	1017	IT05	ITA-ERM	Emilia-Romagna	IT	ITA						
6	1018	1018	ITO6	ITA-FVG	Friuli-Venezia Giulia	IT	ITA						
7	1019	1019	IT07	ITA-LAZ	Lazio	IT	ITA						
8	1020	1020	IT08	ITA-LIG	Liguria	IT	ITA						
the second se	1021	1021	IT09	ITA-LMB	Lombardia	IT	ITA						
9	1022	1022	IT10	ITA-MRC	Marche	IT	ITA						
9 1(	1023	1023	IT11	ITA-MOL	Molise	IT	ITA						
-	1025	1024	IT12	ITA-PMN	Piemonte	IT	ITA						
1(		1024						_					
1( 1) 12		1024	IT13	ITA-PUG	Puglia	13         1025         IT13         ITA-PUG         Puglia         IT         ITA                 >							

An alternative way to perform attribute editing is the one using the show DB table feature (*obviously*, *enabling editing mode*). In this example we have set a filter in order to show only the regions of Italy.

		ayer							
/ector Layer: Admin.Geometry Desc: Administrative regions									
elect	edC	olumns —		Filter Clause			rder By		
ow:		T. (1)C	<u>^</u>	CNTRY_NAM	E	✓	unused	*	Z-
GeometryType("Geometry' Srid("Geometry")				Italy			unused	~	Z-
		eometry") ometry")		Comparison	operator		unused	~	 z.
sTe	d(Ce	entroid("G		•	$0 \Leftrightarrow 0$				
		d("Geomet d("Geomet		ŏ<	0<= (	IsNull	Qu	Jery	
			· / // 🔽	Ŏ>	O>= (	NotNull			
							🗹 Use a	lias names	
r <sup>s</sup>	5elec	ted entitie	es						
		ROWID	PK_UID	FIPS_ADMIN	GMI_ADMIN	ADMIN_NAME	FIPS_CNTRY	GMI_CNTR)	^
	1	1013	1013	IT01	ITA-ABR	Abruzzi	IT	ITA	
	2	1014	1014	IT02	ITA-BSL	Basilicata	IT	ITA	
	3	1015	1015	ІТОЗ	ITA-CLB	Calabria	IT	ITA	
	4	1016	1016	IT04	ITA-CMP	Campania	IT	ITA	
	5	1017	1017	1105	ITA-ERM	Emilia-Romagna	IT	ITA	
	6	1018	1018	IT06	ITA-FVG	Friuli-Venezia Giuli	ia IT	ITA	
	7	1019	1019	IT07	ITA-LAZ	Lazio	IT	ITA	
	8	1020	1020	IT08	ITA-LIG	Liguria	IT	ITA	
	9	1021	1021	IT09	ITA-LMB	Lombardia	IT	ITA	
	10	1022	1022	IT10	ITA-MRC	Marche	IT	ITA	
	11	1023	1023	IT11	ITA-MOL	Molise	IT	ITA	
	12	1024	1024	IT12	ITA-PMN	Piemonte	IT	ITA	
	13	1025	1025	IT13	ITA-PUG	Puglia	IT	ITA	~
13 1025 1025 IT13 ITA-PUG Puglia IT ITA V									

Now we have selected any region belonging to Souther Italy, and then we'll press the **bring to top any selected row** button.

Current Layer       Vector Layer:       Admin.Geometry       Desc:       Administrative regions															
ector L	ayer: A	dmin.Geon	netry	Des	c: Administrative re	gions									
5elected	Columns	;	Filter Clau	use	C	order By									
				NAME	✓	- unused		🔽 🗌 Z-A							
GeometryType("Geometry' Srid("Geometry")			Italy			- unused		🗸 🗌 Z-A							
IsValid("Geometry") Area("Geometry")			Compari	ison operator —		- unused		✓ Z-A							
AsText(	Centroid	( <sup>''</sup> Geometr		00		anasoa									
	oid("Geor oid("Geor		ŏ<	○<=	◯ IsNull		Query								
PK UID		incery //		○ >=											
						🗹 Us	e alias names								
Sel	ected en	tities													
	DWID	PK_UID	FIPS_ADMIN	GMI_ADMIN	ADMIN_NAME	FIPS_CNTRY	GMI_CNTRY	r (							
1	13	1013	IT01	ITA-ABR	Abruzzi	IT	ITA Southern								
2	14	1014	IT02	ITA-BSL	Basilicata	IT	ITA	I							
3	15	1015	IT03	ITA-CLB	Calabria	IT	ITA	I							
4	16	1016	IT04	ITA-CMP	Campania	IT	ITA	I							
5	23	1023	IT11	ITA-MOL	Molise	IT	ITA	I							
6	25	1025	IT13	ITA-PUG	Puglia	IT	ITA	I							
7	17	1017	1105	ITA-ERM	Emilia-Romagna	IT	ITA	I							
	18	1018	IT06	ITA-FVG	Friuli-Venezia Giulia	IT	ITA	I							
8	19	1019	IT07	ITA-LAZ	Lazio	IT	ITA	I							
8		1020	IT08	ITA-LIG	Liguria	IT	ITA	I							
9 10	20		IT09	ITA-LMB	Lombardia	IT	ITA	I							
9 10 11	21	1021			Manual a	IT	ITA	I							
9 10 11		1021 1022	IT10	ITA-MRC	Marche										
9 10 11 12	21 22			ITA-MRC ITA-PMN	Marche Piemonte	IT	ITA	I							
9 10 11 12	21 22	1022	IT10					13         24         1024         IT12         ITA-PMN         Piemonte         IT         ITA         II                >         >         >							

All right: any selected row is now shown on the table top, and the background is highlighted: this actually means any change we'll apply will automatically propagate to the whole selected block.

Please note: if you wish to do so, you can obviously apply a change to a single cell. Simply don't use the selected block at all; you can remove the selected block pressing the **remove rows group** button.

Current Layer Vector Layer: Admin.Geometry Desc: Administrative regions								
	dColumn	is	Filter Cla	use		Order By		
WIE		"Geometry'		NAME	✓	- unused		🖌 🗌 Z
d("G	eometry	") !	Italy			- unused		🗸 🗆 z
	"Geomet Seometry		Compar	ison operator —		- unused		v 🗆 z
Text	(Centroi	d("Geometr	•=	○<>				
Cent Cent	roid("Geo roid("Geo	ometry")) ometry"))	Ŏ<	○<=	◯ IsNull		Query	
UIC		Sincery //	✓ ○>	○>=		_		
						🗹 Us	e alias names	
_Se	lected e	ntities						
	DWID	PK_UID	FIPS_ADMIN	GMI_ADMIN	ADMIN_NAME	FIPS_CNTRY	GMI_CNTRY	<b> </b>
1	13	1013	IT01	ITA-ABR	Abruzzi	IT	ITA Southern	þ. I
2	2 14	1014	IT02	ITA-BSL	Basilicata	IT	ITA Southern	þ.
3	<b>;</b> 15	1015	IT03	ITA-CLB	Calabria	IT	ITA Southern	2
4	i 16	1016	IT04	ITA-CMP	Campania	IT	ITA Southern	p =
1	; 23	1023	IT11	ITA-MOL	Molise	IT	ITA Southern	9
E	i 25	1025	IT13	ITA-PUG	Puglia	IT	ITA Southern	9
	<b>1</b> 7	1017	IT05	ITA-ERM	Emilia-Romagna	IT	ITA	I
8	18	1018	IT06	ITA-FVG	Friuli-Venezia Giulia	IT	ITA	II
9	19	1019	IT07	ITA-LAZ	Lazio	IT	ITA	I
1	<b>o</b> 20	1020	IT08	ITA-LIG	Liguria	IT	ITA	I
1	1 21	1021	IT09	ITA-LMB	Lombardia	IT	ITA	I
1	<b>2</b> 22	1022	IT10	ITA-MRC	Marche	IT	ITA	I
	<mark>3</mark> 24	1024	IT12	ITA-PMN	Piemonte	IT	ITA	II 🤜
1					1			>

Please note: any modified cell is now shown highlighted: this is because such changes are not yet been saved into the DB, but simply are temporarily cached in memory. You have to explicitly press the **save** button in order to make any change to be permanently recorded

into the underlaying DB.

#### 5 - Inserting new entities / drawing

<b>G</b> Planet Heart - Geopolitics		
		🖉 K K 🖧   K K   💥 🙆 🗐
C:\Documents and Settin		
	Scale 1:2376976 43°33'54"N	11°58'14"E

In order to begin a new entity insertion you have prior to:

- make the relevant layer to be the currently selected one
- enable editing operations for this layer
- select the draw tool

To examine a first example (*the most simple one*), we'll now suppose to insert a POINT-type entity, i.e. a new City.

You simply have to position the mouse cursor over the most appropriate coordinates and then perform a **left mouse button click**: and immediately you'll notice a **blinking** point over the map showing the selected point; you've just defined a valid **geometry**, and now you have to assign the corresponding **attributes** in order to complete the new entity insertion.

Insert Entity			
Current Layer			
Vector Layer:	Cities.Geometry		Use alias names
Description:	Cities		Use allas names
Compton			
New Entity			
	Attribute value		
PK_UID			
IMS_ID	9999		
GEO_CODE	9999		
WUP_AGGL	Arezzo		
CNTRY_NAME	Italy		
UN_CODE			
GEO_SUBREG	Western Europe		
GEO_REGION	Europe		
WUP_CAPIT		-	
CAPIT_1_0	acbd		
Y_2003	90	•	
	Terreth Febbur	Abort	
	Insert Entity		

Simultaneously, an **Insert Entity** panel will be shown. Simply fill any required attribute using appropriate values: if **spatialite-gis** detects any data type incoherency then the corresponding value will be highlighted using a colored background.



Please note: spatialite-gis refuses to insert invalid entities (i.e. rows causing any SQL constraint violation); so you have to correct any issue before actually trying to perform an INSERT.



All right: once you've set appropriate values to each required attribute, the new entity you've just now inserted is visible over the map.



As a second example (*a most complex one*), we'll now suppose to insert a POLYGON-type entity, i.e. a new imaginary and fictious Country.

Drawing a polygonal surface is a little bit more complex than defining a single point: procedures related with lines and polygons drawing have already been exposed in the **basic** manual section [*please, re-read the* **measure polygon tool** page if required].

Simply you have to:

- click the left mouse button for each one vertex you intend to place.
- and then press the END key in order to terminate drawing.
- you can use the CTRL+U key [UNDO] in order to discard any wrong or misplaced vertex.



Anyway, inserting a POLYGON entity [and a LINESTRING entity as well] follows the same identical steps required in order to insert a POINT entity:

- 1. first, defining a valid **Geometry** [drawing]
- $2. \ \ \, filling the attribute values using the <math display="inline">\textbf{Insert Entity}$  panel
- 3. once you've completed both steps, the new entity will be immediately shown over the map

#### 💋 Planet Heart - Geopolitics Files 🛞 🖪 🗟 🗄 👘 🌾 🥒 🦧 🚜 🌒 କ 💉 📚 i 関 ۵Į C:\Documents and Settings\sa . 🖃 Cities 🔝 Regions 😚 Countries $\boxtimes$ × < > Scale 1:1085040 44°04'46"N 11°44'42"E

#### 6 – Geometry editing: moving a Point or a Vertex

Assigning new coordinates to some already existing Point (i.e. *moving to a different location*) is a very simple task:

- 1. be sure to set the relevant layer as the currently selected one
- 2. be sure to allow this layer to be editable
- 3. select the **move vertex** tool
- 4. then place the mouse cursor over the Point you intend to move: a special marker will now be shown to confirm this one Point is the currently selected one.
- 5. now hold the mouse left button pressed down, and start dragging.
- 6. you'll notice that a second special marker will be shown, following the mouse motion.
- 7. and in order to make the new position to be permanently assigned to this point (i.e. *completing the move point op*), simply release the mouse left button.



And you can apply the same operation to any Linestring or Ring vertex as well. In this case you'll immediately notice how the figure changes accordingly to the mouse motion.



#### 7 – Geometry editing: deleting a Vertex

For Linestring and Ring geometries you can apply another draw tool in order to edit geometries: i.e. the **delete vertex** tool.

You simply have to place the mouse cursor over the specific vertex you intend to remove [*a special marker will be shown to give you a visual feedback*], and then perform a **left mouse button click**, and the selected vertex will be immediately removed.

Please note: under some circumstances, removing a vertex will produce a **degenerated** geometry, such as a Linestring containing less than 2 vertices, or a Ring containing less than 4 vertices: in order to preserve anyway geometry consistency and correctness, **spatialite-gis** rejects any attempt to produce such degenerated geometries.



#### 8 – Geometry editing: interpolating a Vertex

And for Linestring and Ring geometries you can apply yet another draw tool in order to edit geometries: i.e. the **interpolate vertex** tool.

- 1. place the mouse cursor over the specific Segment you intend to interpolate: a special marker will now be shown to confirm this one Segment is the currently selected one.
- 2. now hold the mouse left button pressed down, and start dragging.
- 3. you'll notice that a second special marker will be shown, identifying the interpolated vertex, and following the mouse motion.
- 4. and in order to make the new interpolated vertex to be permanently inserted (i.e. *completing the interpolate vertex op*), simply release the mouse left button.

#### 9 – Geometry editing: inserting a sub-geometry into a complex Geometry

The followings may be assumed to be *complex* Geometries:

- a MultiPoint i.e. a collection of elementary Points
- a MultiLinestring i.e. a collection of elementary Linestrings
- a MultiPolygon i.e. a collection of elementary Polygons
- but any Polygon as well represents a complex geometry: a polygon is defined by its Exterior Ring, but may well be it contains any arbitrary number of Interior Rings [holes] as well. So, any Polygon may be assumed to be a collection of elementary Rings.

In the following example we'll examine how to add Interior Rings and further Polygons to some MultiPolygon; this one being the most complex case, you'll find really easy adapt by yourself this example to MultiLinestrings and / or MultiPoints.



First of all, we are required to **select the current entity**: we can accomplish this task using the **select complex geometry** tool. Actually we'll select the fictious *Utopia Island* we've just inserted.

Once an entity becomes the currently selected one, special markers will be shown to highlight this entity over the map.

And when a complex geometry has been selected, then any further drawn geometry will be added to this complex geometry collection, instead of being considered as a new, distinct entity [*as is the standard, ordinary behavior*].



Accordingly to the previously exposed rules, the new Ring we've just now drawn will be immediately appended as a new elementary Polygon, belonging to the Utopia Island MultiPolygon.



And when **spatialite-gis** can automatically detect if the new Ring has to be considered as an Interior Ring, so you can easily add **holes** to any Polygon.



So, in a very few steps, we've been able to define a quite complex geometry, i.e. a MultyPolygon containing two individual Polygons, each one of them containing internal holes.

## **10 – Geometry editing: removing a sub-geometry from a complex Geometry**

In the previous paragraph we've explored how to use **spatialite-gis** in order to add new elementary geometries into a complex Geometry.

Quite obviously, the opposite operation is supported as well, i.e. the one to remove elementary geometries from a complex Geometry. This includes:

- removing a **Point** from a **MultiPoint**
- removing a Linestring from a MultiLinestring
- removing a Polygon from a MultiPolygon
- removing an **Interior Ring** from a **Polygon**

Please note: under some circumstances, removing an elementary geometry from a collection will produce a **degenerated** geometry, i.e. a **NULL** [**empty**] one: in order to preserve anyway geometry consistency and correctness, **spatialite-gis** rejects any attempt to produce such degenerated geometries.



We'll continue using the fictious **Utopia Island** in this example.

You simply have to select the **delete sub-geometry tool**, and then select the individual ring you intend to remove; special markers will be shown to highlight this entity (and the specific item) over the map. In order to confirm deletion of this item, you simply have to perform a **mouse left button click**.

In the special case of polygon's rings, the following rules apply:

- Interior rings [holes] will be removed on an individual base
- Deleting an Exterior ring implicitly assumes deletion of the whole Polygon [*this including any interior ring as well*].



And in this case too we've been able to edit a complex geometry in a very few and simple steps.