

PostgreSQL



Map  
Server

PostGIS

Geo...

Sin Enredos



Andres Herrera

Image Silhouette by pixiv.com/productions.com

## Advertencia:

*Este documento no pretende ser una guía de estudio, una guía de bolsillo, ni mucho menos un documento de referencia bibliográfico de consulta, aquí tan solo pretendo explicar por medio de una serie de sencillos ejemplos y pasos la forma mas fácil de sacarle provecho a MapServer, Php-MapScript y al motor de bases de datos PostgreSQL y su extensión espacial PostGIS en nuestros desarrollos.*

*Esta es la primera y única versión de este documento, el autor no se hace responsable sobre el uso indebido de la información aquí consignada.*

*La iconografía usada pertenece a sus correspondientes autores.*



Este documento esta protegido bajo una Licencia Creative Commons, Usted es libre de: (1) copiar, distribuir y comunicar públicamente la obra (2) hacer obras derivadas, Bajo las condiciones siguientes: **Reconocimiento**. Debe reconocer los créditos de la obra de la manera especificada por el autor o el licenciador. **No comercial**. No puede utilizar esta obra para fines comerciales. **Compartir bajo la misma licencia**. Si altera o transforma esta obra, o genera una obra derivada, sólo puede distribuir la obra generada bajo una licencia idéntica a ésta. (1)Al reutilizar o distribuir la obra, tiene que dejar bien claro los términos de la licencia de esta obra. (2)Alguna de estas condiciones puede no aplicarse si se obtiene el permiso del titular de los derechos de autor.



## Introducción

En esta guía debido a su contenido informal, esta concebida bajo el principio de KISS (Keep It Simple Stupid<sup>1</sup>), así que no entraremos en formalismos y en definiciones complejas de cada uno de los componentes de software y las metodologías a utilizar a lo largo del desarrollo, por otro lado; se requiere una serie de conocimientos básicos lograr un completo entendimiento, de lo contrario es recomendable consultar con “San Google<sup>2</sup>”.

Esta guía se desarrollara tratando de abordar el mayor número interrogantes que se presentan a lo largo del aprendizaje de este tipo de tecnologías, junto a esta guía se entregan una serie de ejemplos básicos, algunas herramientas y utilerías.

Algunos de mi autoría y otros, recopilados de diferentes fuentes, realizando ligeros cambios. A manera de “PLUS” introduciremos algunas utilidades disponibles, las cuales nos permitirán conocer otras formas no tradicionales de realizar las cosas.

**En esta versión del documento, trabajaremos sobre dos ambientes operativos:**

**Linux en su distribución “ Ubuntu 8.04 – Hardy Heron” y Windows XP.**

---

<sup>1</sup> Principio KISS, [http://es.wikipedia.org/wiki/Principio\\_KISS](http://es.wikipedia.org/wiki/Principio_KISS)

<sup>2</sup> Google, <http://www.google.com>

## Herramientas

En esta guía marcharemos de la mano de herramientas de código abierto, de libre acceso y disponibles en la red, el pool de herramientas a trabajar en esta guía, son aquellas que no pueden faltar en la manga de un mago de la información geográfica.



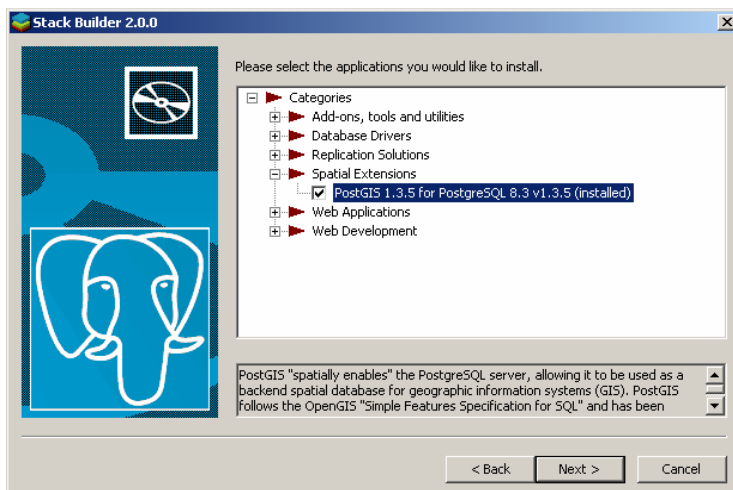
### PostgreSQL + PostGIS

Para el desarrollo de la mayoría de los ejemplos y ejercicios contenidos en esta guía, se requiere el motor de bases de datos PostgreSQL con la extensión de soporte de datos espaciales PostGIS, correctamente instalado y configurado en el equipo de trabajo.

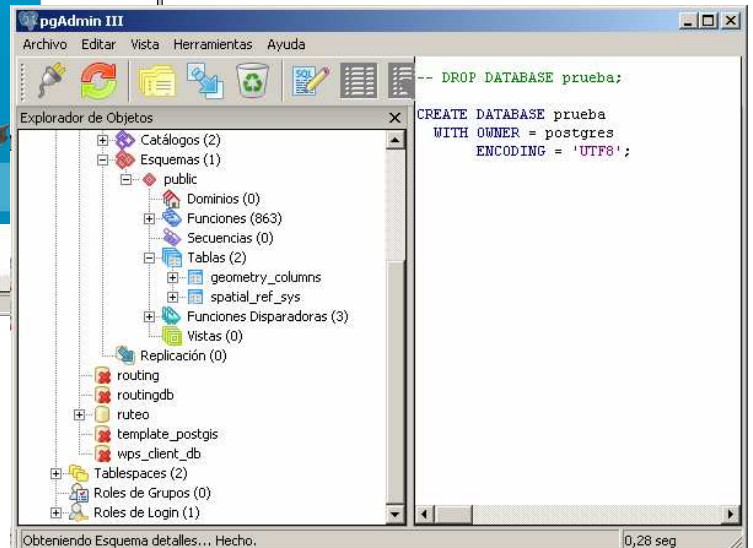
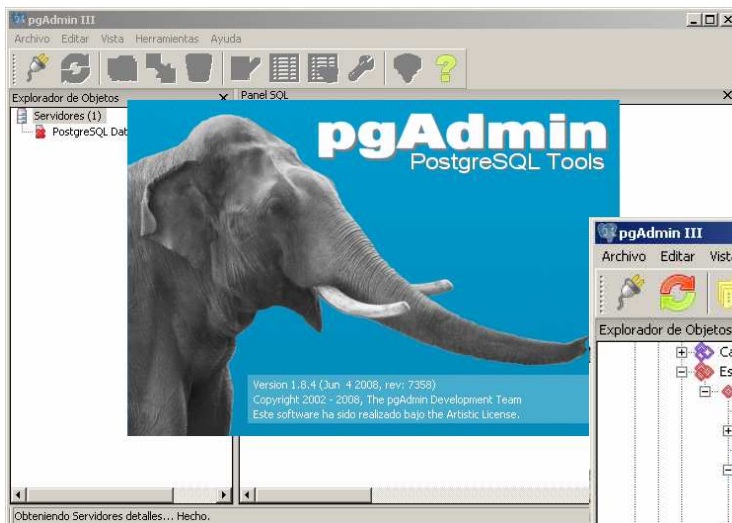


Descargamos el instalador de la URL: <http://www.master.postgresql.org/download/mirrors-ftp/binary/v8.3.5/win32/postgresql-8.3.5-1.zip>, La instalación se resume a unos simples clicks. Aunque debemos tener particular cuidado, con los datos proporcionados para el **nombre de usuario**, **contraseña** y puerto, que asignaremos a la cuenta de trabajo.

**Extensión de PostGIS:** Simplemente la debemos activar dirigiéndonos a: **Inicio -> Programas -> PostgreSQL 8,3 -> Application Builder Stack**. Seleccionando de la lista La última versión de PostGIS y haciendo clic en "Siguiente". Una vez instalada esta aparecerá en el marco del "*Spatial Extensions*"



**Cliente PgAdmin III:** En la instalación del PostgreSQL seleccionamos el cliente PgAdmin III contenido en el paquete, una vez instalado podemos iniciar dicha herramienta dirigiéndonos a: **Inicio -> Programas -> PostgreSQL 8.3 -> pgAdmin III**



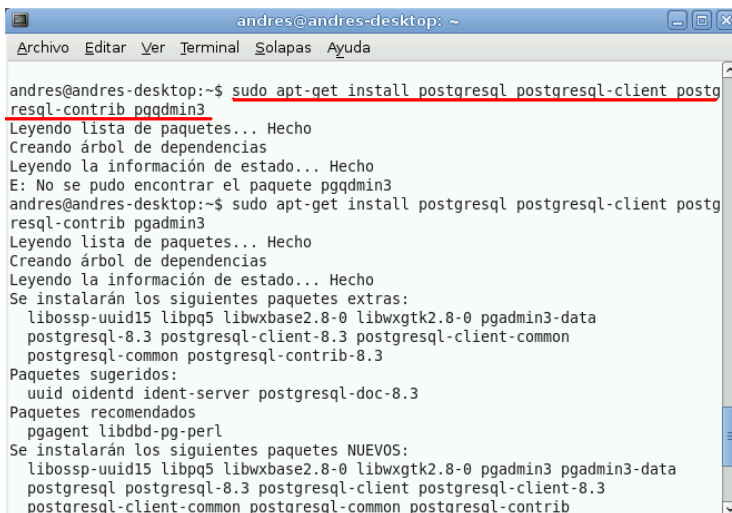
**Checkeamos instalación PostGIS:**  
Verificando tablas:

*geometria\_columns y spatial\_ref\_sys*



Desde nuestra consola de comandos digitamos:

**\$ sudo apt-get install postgresql postgresql-client postgresql-contrib postgresql-8.3-postgis postgis pgadmin3**



La herramienta **APT**, administra el acceso a repositorios disponibles de los paquetes de software, utilizando el fichero **sources.list**

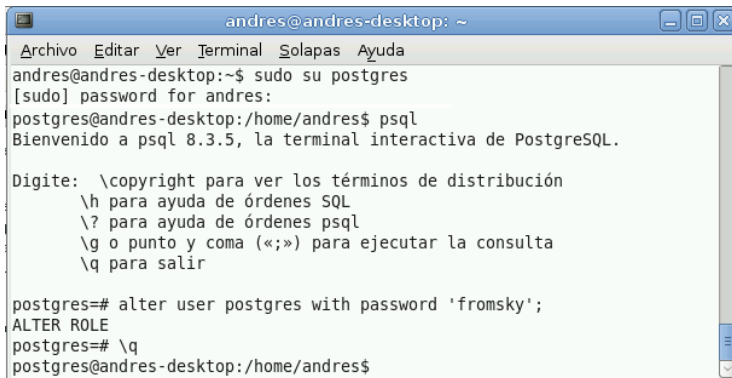
**\$ sudo nano /etc/apt/sources.list**

\* Una vez descargados, e instalados los paquetes mediante **APT**, procedemos a configurar el motor PostgreSQL.

```

$ sudo su postgres
$ psql
$ alter user postgres with password 'su_password';
$ createdb prueba
$ createlang plpgsql prueba
$ psql -d prueba -f /usr/share/postgresql-8.3-postgis/lwpostgis.sql
$ psql -d prueba -f /usr/share/postgresql-8.3-postgis/spatial_ref_sys.sql

```



```

andres@andres-desktop: ~
Archivo Editar Ver Terminal Solapas Ayuda
andres@andres-desktop:~$ sudo su postgres
[sudo] password for andres:
postgres@andres-desktop:/home/andres$ psql
Bienvenido a psql 8.3.5, la terminal interactiva de PostgreSQL.

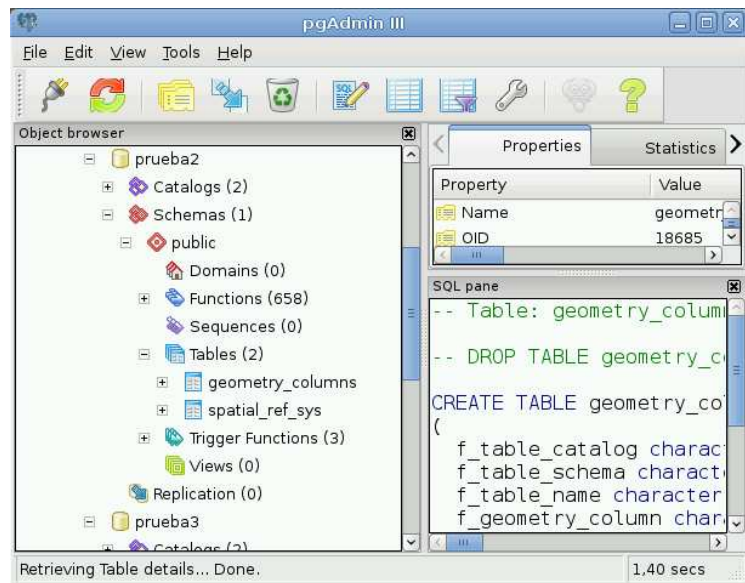
Digite: \copyright para ver los términos de distribución
        \h para ayuda de órdenes SQL
        \? para ayuda de órdenes psql
        \g o punto y coma («;») para ejecutar la consulta
        \q para salir

postgres=# alter user postgres with password 'fromsky';
ALTER ROLE
postgres=# \q
postgres@andres-desktop:/home/andres$

```

**Checkeamos instalación PostGIS:**  
verificando tablas:

*geometry\_columns* y *spatial\_ref\_sys*



**Tip !**

Creemos un template, que nos simplifica la vida, a la hora de crear nuevas bases de datos espaciales.

```

$ sudo su postgres
$ createdb -E UTF8 -O postgres -U postgres template_postgis
$ createlang plpgsql -d template_postgis -U postgres
$ psql -d template_postgis -U postgres -f /usr/share/postgresql-8.3-postgis/lwpostgis.sql
$ psql -d template_postgis -U postgres -f /usr/share/postgresql-8.3-postgis/spatial_ref_sys.sql
$ createdb -U postgres -T template_postgis prueba2

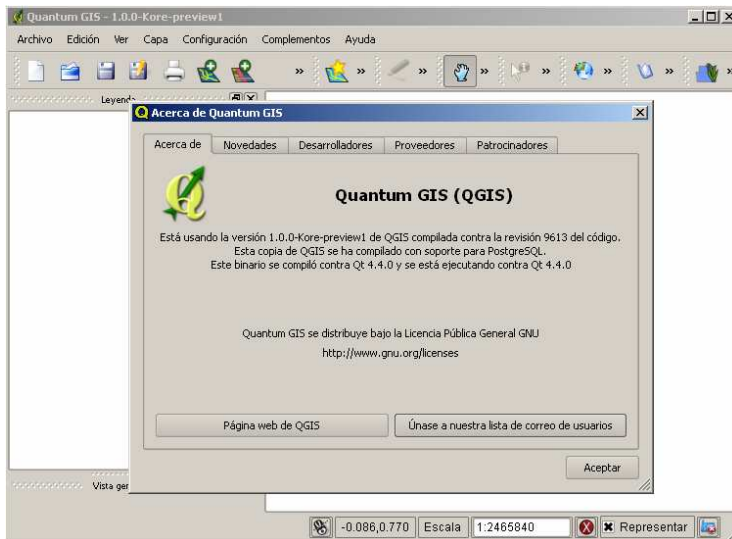
```



## Quantum GIS (QGIS)



QGIS es un proyecto OpenSource multiplataforma, para el tratamiento de información espacial, podemos obtenerla desde la siguiente URL: <http://download.osgeo.org/qgis/win32/QGIS-1.0.0preview2-Setup.exe>

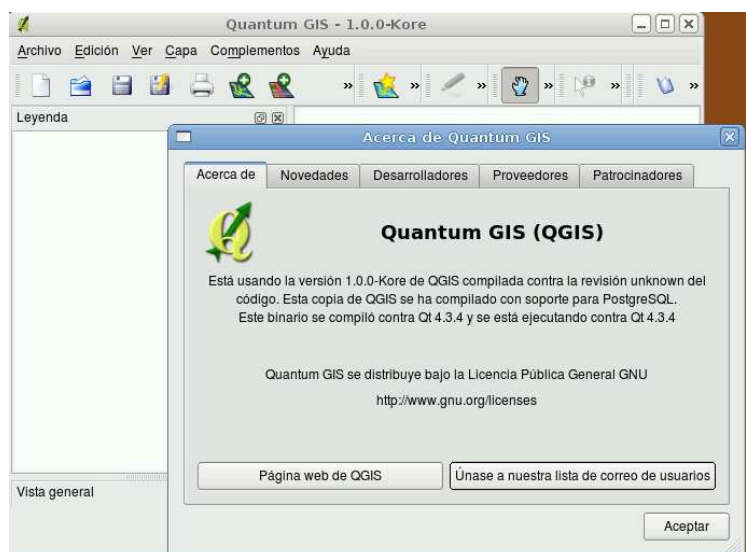


Agregue las siguientes líneas en: `/etc/apt/sources.list`

```
deb http://ppa.launchpad.net/jef-norbit/ubuntu hardy main
deb-src http://ppa.launchpad.net/jef-norbit/ubuntu hardy main
```

Instale mediante APT:

```
$ sudo apt-get update
$ sudo apt-get install qgis
```



## FWTools

Es un paquete de utilerías y herramientas de línea de comandos para el trabajo con información espacial entre estas herramientas están: (GDAL, PROJ .4, etc. ...).



La instalación es igualmente sencilla, descargamos el instalador de la siguiente URL: <http://home.gdal.org/fwtools/FWTools228.exe> una vez descargado, procedemos a ejecutar el instalador. Ya instalado: **Inicio -> Programas -> FWTools 2.2.8 -> FWTools Shell**

```
C:\Archivos de programa\FWTools2.2.8>cd bin
C:\Archivos de programa\FWTools2.2.8\bin>dir
El volumen de la unidad C es WindowsXP
El número de serie del volumen es: 5C35-E6E5

Directorio de C:\Archivos de programa\FWTools2.2.8\bin
30/12/2008 12:57 <DIR> .
30/12/2008 12:57 <DIR> ..
17/06/2008 12:01 2.563.143 a.exe
29/10/2008 13:12 27.136 adrg.dll
12/07/2005 20:49 2.973.559 bgd.dll
29/10/2008 13:12 9.728 bmp2tiff.exe
29/10/2008 13:12 10.240 cs2cs.exe
08/03/2008 08:30 24.576 dtcanada.dll
29/10/2008 13:12 28.672 dted.dll
12/07/2005 20:49 24.635 dtusa.dll
```



```
$ wget http://home.gdal.org/fwtools/FWTools-linux-2.0.6.tar.gz
$ tar xzvf FWTools-linux-0.9.5.tar.gz
$ cd FWTools-linux-0.9.5
$ sudo ./install.sh
```

```
andres@andres-desktop: ~/FWTools-2.0.6/bin_safe
Archivo Editar Ver Terminal Solapas Ayuda
andres@andres-desktop:~/FWTools-2.0.6$ ls
bin      fwtools_env.csh  info      pics        sbin        wms
bin_safe fwtools_env.sh  install.sh pymod        share        xmlconfig
conf     html             lib       ramps       symbols
demo-data include          man        README.1ST  tools
andres@andres-desktop:~/FWTools-2.0.6$ cd bin_safe/
andres@andres-desktop:~/FWTools-2.0.6/bin_safe$ ls
addtiff  gdalmanage      legend      pngtogd2    tiff2bw
bmp2tiff gdal_merge.py   listgeo     ppm2tiff    tiff2pdf
cs2cs    gdal_rasterize  mapscriptvars proj         tiff2ps
dap-config gdalindex       mapserv     python      tiff2rgba
epsg_tr.py gdaltransform   mapserver-config ras2tiff     tiffcmp
fax2ps    gdal_translate  mscrypt     raw2tiff    tiffcp
fax2tiff  gdalwarp        nad2bin     rgb2pct.py  tiffcrop
gcps2vec.py gdlib-config    nad2nad     rgb2ycbcr   tiffdither
gcps2wld.py geod            nearblack   scalebar     tiffdump
gdal2xyz.py geos-config     ogdi_import shp2img      tiffgt
gdaladdo  geotifcp        ogdi_info   shp2pdf      tiffinfo
```

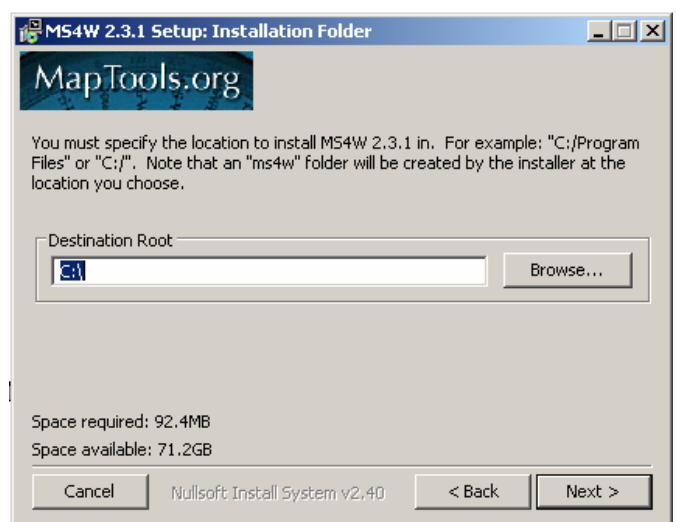
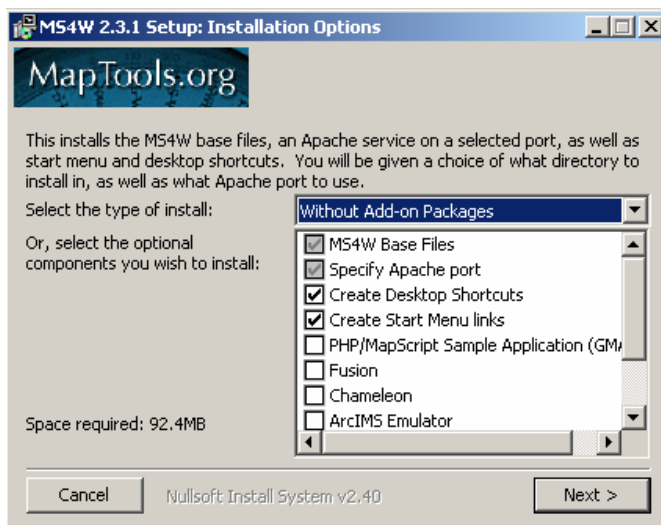
Una vez instalado, los binarios a ser utilizados se encuentran en: ( **bin\_safe** )

## Apache+MapServer+PhpMapScript

En particular, existen paquetes que nos facilitan la vida, en Windows, andaremos de la mano del (MS4W) el cual instala un ambiente pre-configurado de servidor Web, en este caso Apache además crea una completa instalación de Php5, MapServer, MapScript y otra serie de componentes útiles, en Linux trabajaremos con (FGS), ambos compilados y empaçados por maptools.org

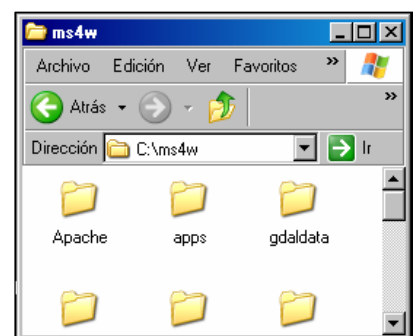
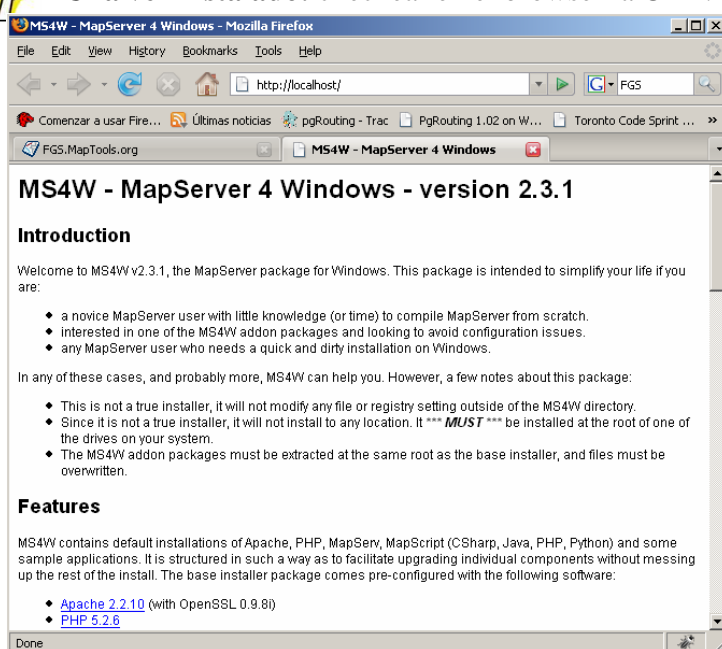


**MS4W (MapServer para Windows):** descargamos el instalador de la URL: <http://www.maptools.org/dl/ms4w/ms4w-2.3.1-setup.exe>, la instalación de este paquete se resume a unos simples clicks en otras palabras “*Siguiente -> Siguiente*”.



**Nota de instalación:** instalamos en el directorio raíz (C: o D:) en la ruta C:\ms4w\

**Una vez instalado:** checkear en el browser la URL: <http://localhost/>







**FGS en Linux):** descargamos el paquete de la URL:

[http://www.maptools.org/dl/fgs/releases/1.0/1.0.0/self-installers/fgs-mapserver\\_5.0.2-fgs\\_1.0.0-linux-i386.bin](http://www.maptools.org/dl/fgs/releases/1.0/1.0.0/self-installers/fgs-mapserver_5.0.2-fgs_1.0.0-linux-i386.bin) , la instalación es muy simple, a comparación de descargar e instalar paquete por paquete.

```
$ sudo sh fgs-mapserver_5.0.2-fgs_1.0.0-linux-i386.bin
```

+ Asignar path por defecto: **/opt/fgs/**

+ Asignar puerto escucha del apache: **80 o 8080**

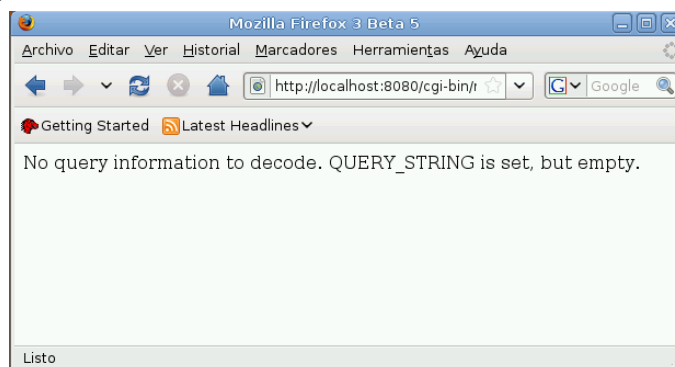
```
$ sudo echo "./opt/fgs/setenv.sh" >> ~/.bashrc
```

```
$ cd /opt/fgs/www/conf
```

```
$ cp php5.ini.template php5.ini
```

```
$ adduser apache
```

```
$ fgs start
```



**Una vez instalado:** chequear en el browser la URL: **http://localhost:puerto/cgi-bin/mapserv/**

Si el mensaje: **"No Query information to decode, QUERY\_STRING" is set, but empty**" es desplegado, FGS ha sido instalado correctamente en el sistema.

### ¿Problemas?



Si tiene problemas hasta este momento, en cualquiera de los pasos de instalación y/o configuración de cada una de las herramientas, se recomienda visitar los **sitios oficiales, foros y wikis** generalmente aquí se encuentra información de calidad y amables personas que estarán dispuestas a colaborar al máximo en la búsqueda de la solución relacionada a su problema.

**Por ultimo;** en la red encontramos distribuciones que nos facilitan la vida, donde ya todo el entorno viene pre-configurado y listo para trabajar en un ambiente GIS, estas son algunas de las más comunes.

### SlaxGIS

<http://geomatica.como.polimi.it/software/slaxGIS/index.php>

### UbuntuGIS

<https://wiki.ubuntu.com/UbuntuGIS>

### FOSS4G2008 Live CD

<http://blog.ominiverdi.org/index.php?/categories/2-GIS-liveCD>

### DebianGIS

<http://wiki.debian.org/DebianGis>

### OpenSuseGIS

<http://cartosig.upv.es/es/software/opensuse-gis>

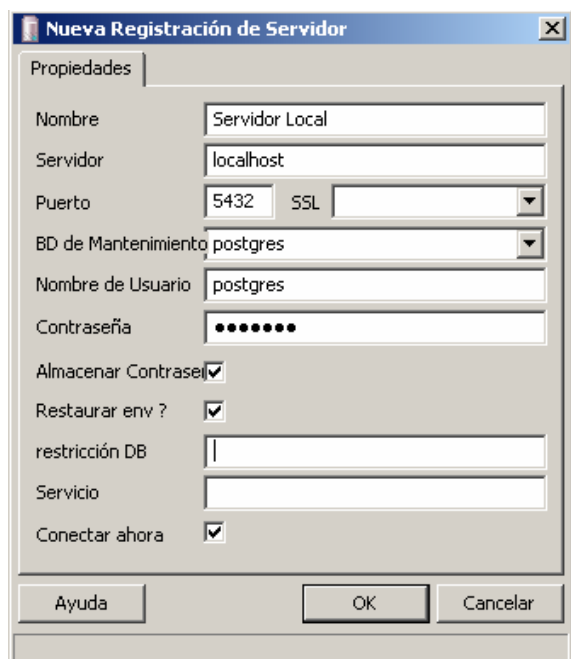
## Ahora SI ! Manos a la Obra !

Para iniciar, crearemos una base de datos de trabajo, la cual llamaremos: ( **tutopostgis** )

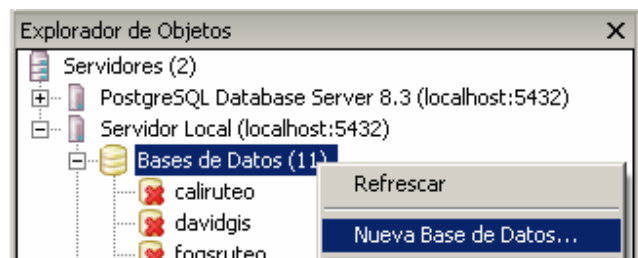


Lo haremos desde la forma fácil y rápida:

- 1) Iniciamos PgAdmin III.
- 2) Creamos la conexión con el servidor de base de datos, indicando parámetros de conexión.

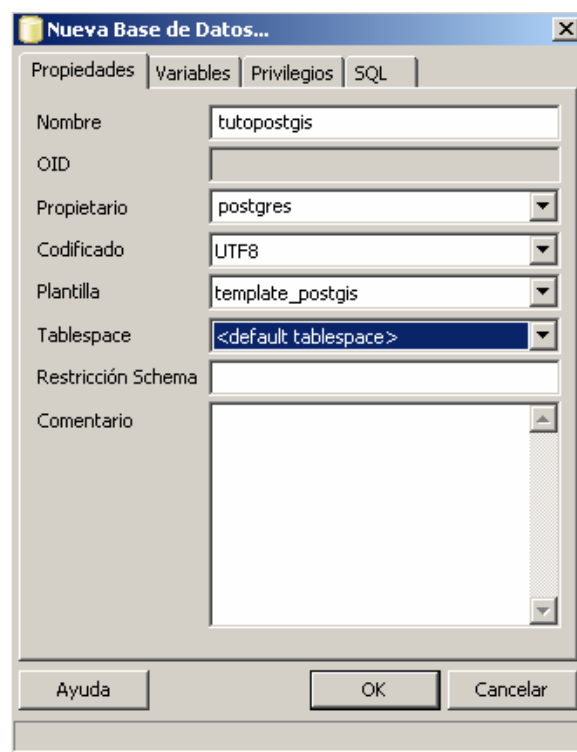
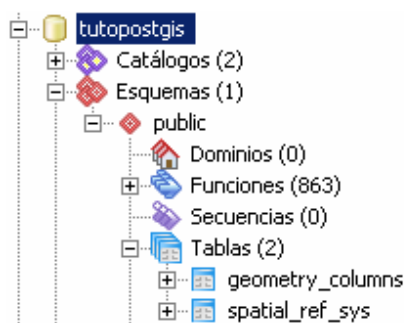


- 3) En la raíz (**Bases de Datos**), seleccionamos: **Nueva Base de Datos**.



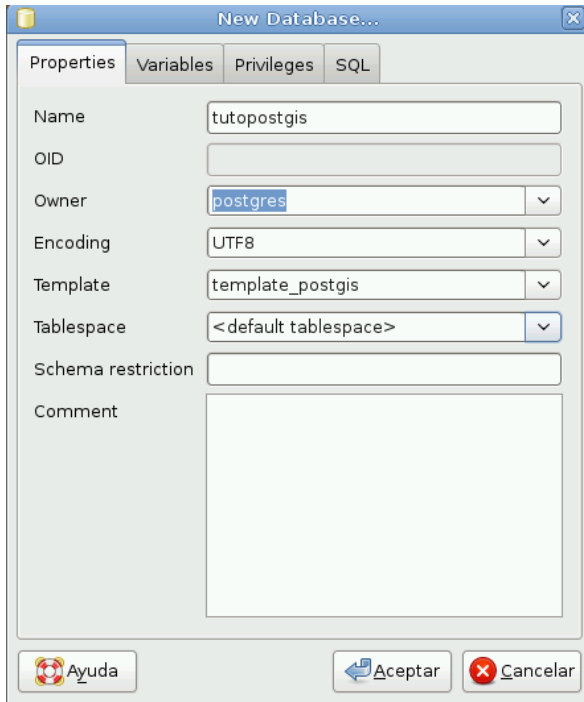
- 4) Asignamos nombre: (**tutopostgis**), encoding (**UTF8**), plantilla (**template\_postgis**).

- 5) Finalmente verificamos la existencia, de las tablas: **geometry\_columns** y **spatial\_ref\_sys**



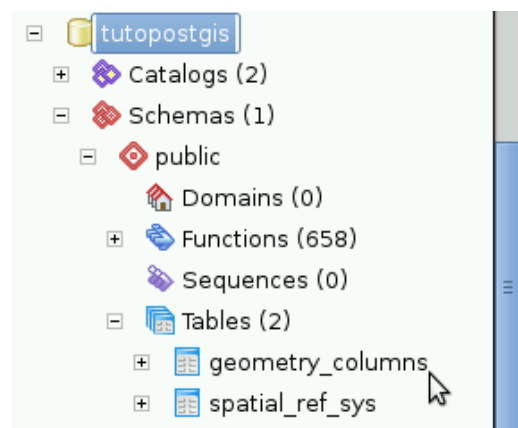


Iniciando el cliente PgAdmin III, y realizando los pasos inmediatamente anteriores (1,2,3,4), creamos nuestra base de datos de trabajo (**tutopostgis**)



5) Finalmente verificamos la existencia, de las tablas:

*geometry\_columns* y *spatial\_ref\_sys*



- The funniest way :)

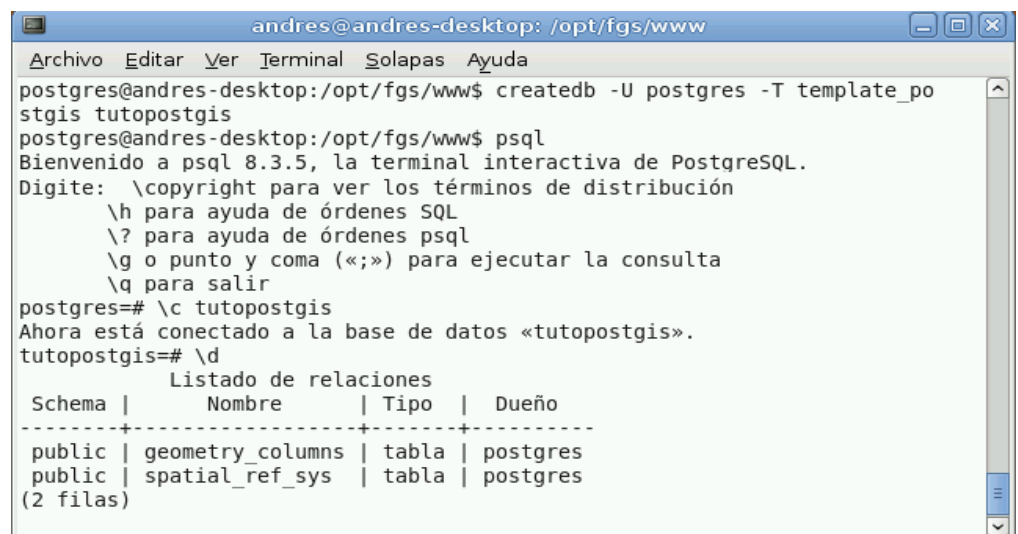
\$ sudo su postgres

\$ createdb -U postgres -T template\_postgis tutopostgis      **\*\*Nota<sup>3</sup>**

\$ psql

\c tutopostgis

\d



<sup>3</sup> Checkear creación **template**, en capítulo “Herramientas -> Postgres+PostGIS”

**Nuestra primera tabla espacial:** ya nuestra base de datos soporta objetos espaciales gracias a PostGIS. Bien!, ahora creemos nuestra primera tabla, donde almacenaremos geometrías espaciales de tipo **PUNTO**.



Ya sea, a través de PgAdmin III, o la Terminal interactiva de PostgreSQL en Windows o Linux, la secuencia SQL que indicaremos será la siguiente:

Creamos una tabla llamada (**tabla1**):

```
CREATE TABLE tabla1
(
    gid serial NOT NULL,
    id int4,
    nombre varchar(20),
    CONSTRAINT tabla1_pkey PRIMARY KEY (gid)
);
```

Agregamos una columna, espacial a (**tabla1**), la cual tendrá el nombre (**the\_geom**):

```
SELECT AddGeometryColumn('tabla1','the_geom',-1,'POINT',2);
```

Insertamos un registro a la tabla (**Punto: con coordenadas 1, 1**):

```
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (1, 'Primer Punto',
GeometryFromText('POINT(1 1)'));
```

Visualizamos el resultado:

```
SELECT gid, id, nombre, AsText(the_geom) AS geometria FROM tabla1;
```

Panel de Salida				
Salida de datos				
Comentar Mensajes Historial				
	gid integer	id integer	nombre character var	geometria text
1	1	1	Primer Punto	POINT(1 1)

Ahora visualicémoslo gráficamente a través de QGIS.

- 1) Iniciar QGIS
- 2) Añadir capa PostGIS. ( **Capa -> Añadir Capa PostGIS**)
- 3) Crear una nueva conexión PostGIS, indicando parámetros.

**Crear una nueva conexión a PostGIS**

Información sobre la conexión

Nombre:

Servidor:

Base de datos:

Puerto:

Nombre de usuario:

Contraseña:

☒ Guardar contraseña

☐ Buscar sólo en la tabla de columnas de la geometría

☐ Buscar sólo en el esquema "público"

Probar conexión

Aceptar Cancelar Ayuda

4) Conectar a conexión previamente creada.

**Añadir tabla(s) PostGIS**

Conexiones de PostgreSQL

Conexion PostgreSQL

Conectar Nueva Editar Borrar

Esquema	Tabla	Tipo	Columna de geometría
public	tabla1	POINT	the_geom

Opciones de búsqueda...

Ayuda Añadir Cerrar

Una vez conectados a la base de datos, se listarán las tablas que poseen geometría, las cuales QGIS tendrá la capacidad de desplegar en su entorno.



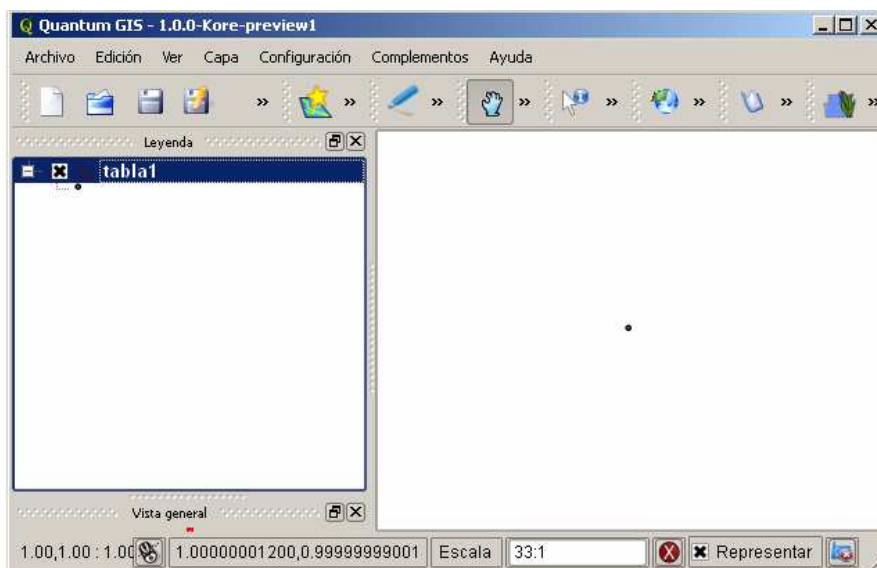
En este caso, nuestra tabla: **(tabla1)**, con geometría de tipo **(POINT)**, cuya columna geométrica es **(the\_geom)**.

5) seleccionamos **(tabla1)**, listada y presionamos **(Añadir)**.



**Well Done!**

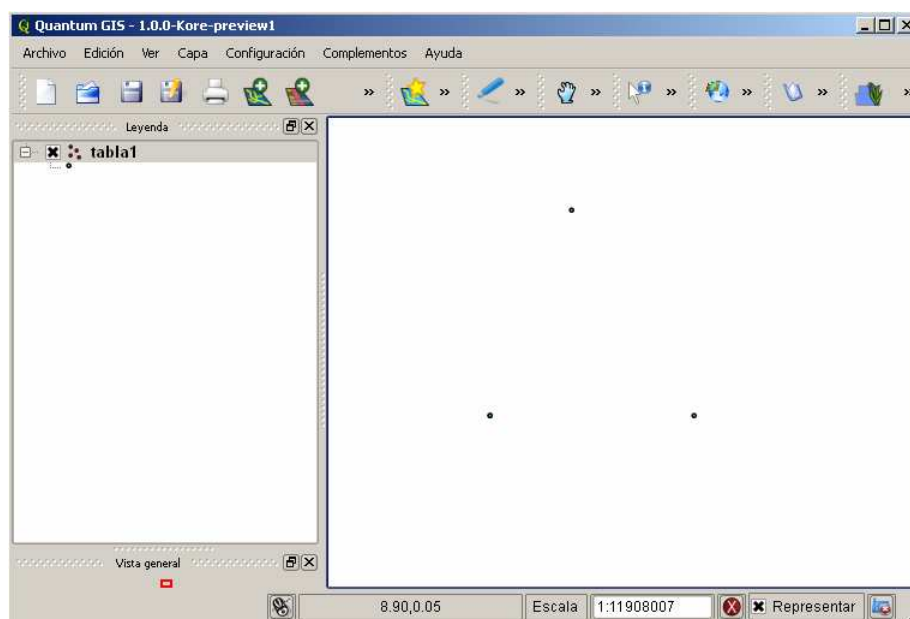
Hemos creado, almacenado y desplegado gráficamente nuestro primer objeto espacial,



**Agregando más elementos a nuestra tabla**

**Vértices de un triángulo:**

```
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (1, 'Primer Punto',  
GeometryFromText('POINT(1 1)'));  
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (2, 'Segundo Punto',  
GeometryFromText('POINT(6 1)'));  
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (3, 'Tercero Punto',  
GeometryFromText('POINT(3 6)'));
```

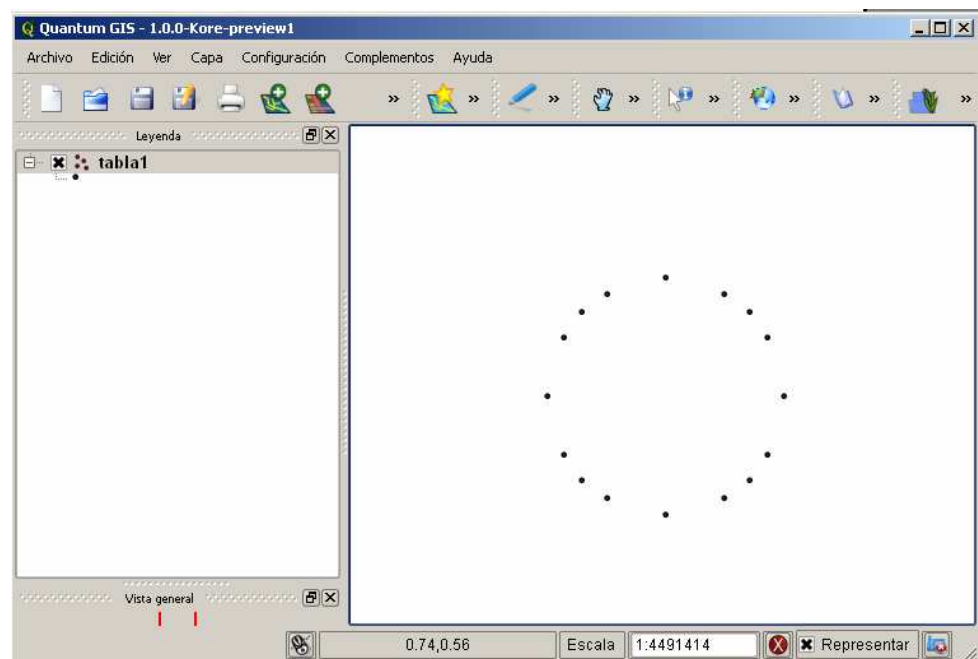


### Jugando un poco con la inserción:

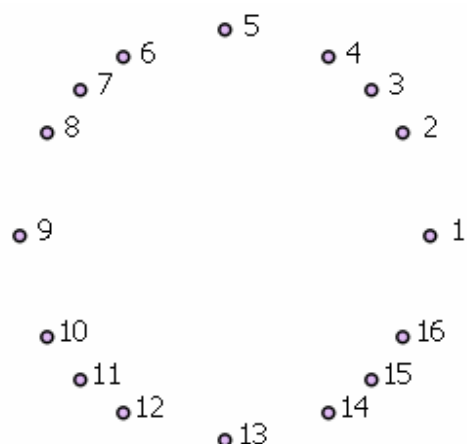
```
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (1, 'circulo', GeometryFromText('POINT(1 0)'));
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (2, 'circulo', GeometryFromText('POINT(0.866025403784439
0.5)'));
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (3, 'circulo', GeometryFromText('POINT(0.707106781186548
0.707106781186548)'));
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (4, 'circulo', GeometryFromText('POINT(0.5
0.866025403784439)'));
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (5, 'circulo', GeometryFromText('POINT(0 1)'));
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (6, 'circulo', GeometryFromText('POINT(-0.5
0.866025403784439)'));
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (7, 'circulo', GeometryFromText('POINT(-0.707106781186548
0.707106781186548)'));
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (8, 'circulo', GeometryFromText('POINT(-0.866025403784439
0.5)'));
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (9, 'circulo', GeometryFromText('POINT(-1 0)'));
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (10, 'circulo', GeometryFromText('POINT(-0.866025403784439
-0.5)'));
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (11, 'circulo', GeometryFromText('POINT(-0.707106781186548
-0.707106781186548)'));
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (12, 'circulo', GeometryFromText('POINT(-0.5 -
0.866025403784439)'));
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (13, 'circulo', GeometryFromText('POINT(0 -1)'));
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (14, 'circulo', GeometryFromText('POINT(0.5 -
0.866025403784439)'));
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (15, 'circulo', GeometryFromText('POINT(0.707106781186548
-0.707106781186548)'));
INSERT INTO tabla1 (id, nombre, the_geom) VALUES (16, 'circulo', GeometryFromText('POINT(0.866025403784439
-0.5)'));
```



Como resultado tenemos un círculo de radio 1, centrado en el origen (0,0), construido con 16 puntos.

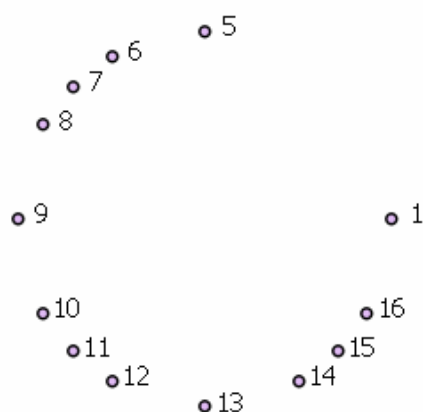


### ¿Como eliminar un registro?



Eliminaremos los registros que tienen asociados el **id 2, 3 y 4**.

```
DELETE FROM tabla1 where id = 2;  
DELETE FROM tabla1 where id = 3;  
DELETE FROM tabla1 where id = 4;
```



### ¿Como modificar un registro?

Modificaremos, el atributo alfanumérico de posición del registro con **id= 5**, desplazándolo hacia el centro (**coordenadas 0, 0**)

```
UPDATE tabla1 set the_geom=GeometryFromText('POINT(0 0)') where id=5;
```

### Optimizando nuestra tabla espacial

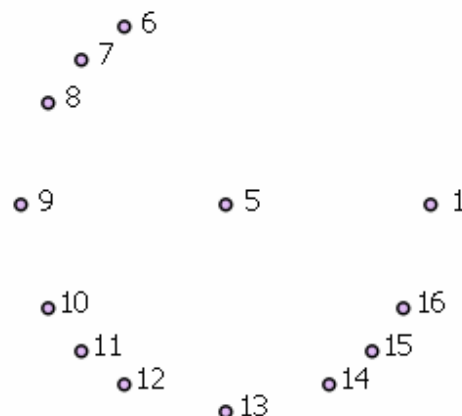
Ahora crearemos un índice espacial usando GIST.

```
CREATE INDEX geom_tabla1_idx  
ON tabla1  
USING gist (the_geom);
```



Para estos sencillos ejemplos, no notaremos la diferencia, en cuanto a tiempos y velocidad en las operaciones sobre tablas espaciales, pero con grandes volúmenes de datos y bases grandes, es vital indexar estas para alcanzar un performance ideal.

**Nota:** Podrá encontrar los fuentes de este ejemplo, en el archivo: **example\_1.sql**



## Exportando Información en formato ESRI SHP a PostgreSQL

En este ítem aprenderemos a importar y a exportar hacia y desde nuestra base de datos espacial, información en formato ESRI Shape.



**Usando Shp2Pgsql:** shp2pgsql es una herramienta vía consola que permite la conversión de nuestro archive en formato shp a sql.

1) Copiar **example\_2\_poligono.shp** , **example\_2\_poligono.dbf** , **example\_2\_poligono.shx** , **example\_2\_poligono.prj** al path donde se encuentra al binario **shp2pgsql.exe** , por lo general (instalación por defecto: **C:\Archivos de programa\PostgreSQL\8.3\bin** )

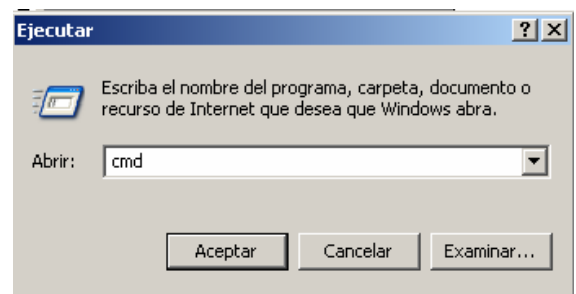
2) Iniciar la línea de comandos (cmd.exe)

3) Desde cmd.exe ir hasta el path:

**\$ cd C:\Archivos de programa\PostgreSQL\8.3\bin**

4) Ejecutar:

**\$ shp2pgsql.exe example\_2\_poligono example\_2\_poligono > example\_2\_poligono.sql**



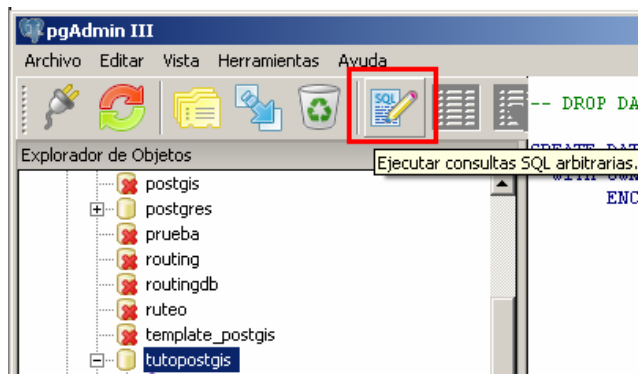
```
C:\WINDOWS\system32\cmd.exe
C:\Archivos de programa\PostgreSQL\8.3\bin>shp2pgsql.exe shp/example_2_poligono
example_2_poligono > shp/example_2_poligono.sql
Shapefile type: Polygon
Postgis type: MULTIPOLYGON[2]
C:\Archivos de programa\PostgreSQL\8.3\bin>_
```

5) Verificar archivo generado (**example\_2\_poligono.sql**). (Simplemente lo abrimos en un editor de texto) y debe presentar una estructura similar a la siguiente:

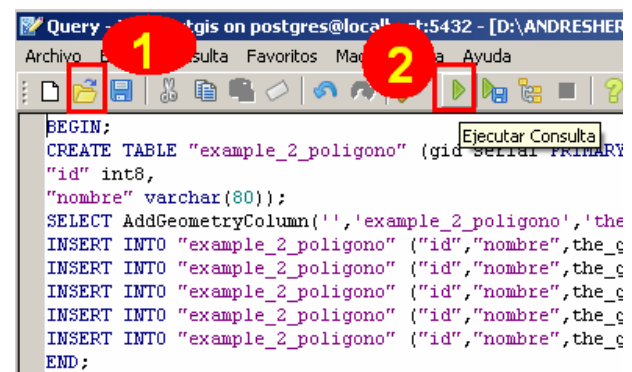
```
example_2_poligono.sql
1 BEGIN;
2 CREATE TABLE "example_2_poligono" (gid serial PRIMARY KEY,
3 "id" int8,
4 "nombre" varchar(80));
5 SELECT AddGeometryColumn('','example_2_poligono','the_geom','-1','MULTIPOLYGON',2);
6 INSERT INTO "example_2_poligono" ("id","nombre",the_geom) VALUES ('1','Poligono1','01060
7 INSERT INTO "example_2_poligono" ("id","nombre",the_geom) VALUES ('2','Poligono2','01060
8 INSERT INTO "example_2_poligono" ("id","nombre",the_geom) VALUES ('3','Poligono3','01060
9 INSERT INTO "example_2_poligono" ("id","nombre",the_geom) VALUES ('4','Poligono4','01060
10 INSERT INTO "example_2_poligono" ("id","nombre",the_geom) VALUES ('5','Poligono5','01060
11 END;
```

6) Ahora tan solo resta exportar a nuestra base de datos (**tutopostgis**) el archivo SQL, generado (**example\_2\_poligono.sql**)

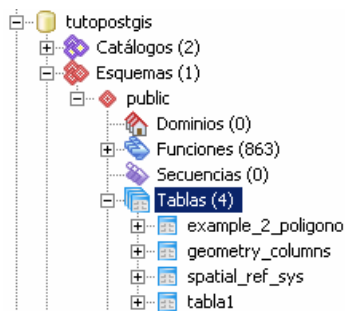
7) Ejecutar PgAdmin III, e iniciar el cuadro de dialogo (Ejecutar consultas SQL arbitrarias).



8) Abrir el archivo (**example\_2\_poligono.sql**)(1) y ejecutar la consulta (2).

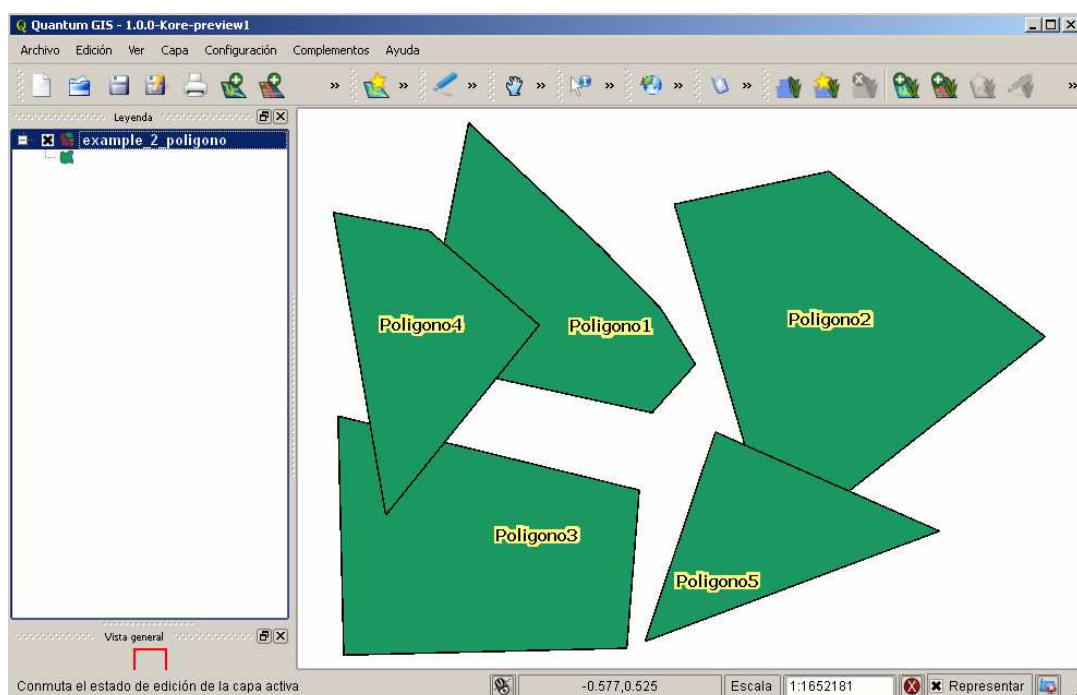


9) Verificamos la creación de la nueva tabla: (**example\_2\_poligono**)



\* Ahora visualicemos el resultado en QGIS.

Esquema	Tabla	Tipo	Columna de geometría	Sql
public	example_2_poligono	MULTIPOLY...	the_geom	
public	tabla1	POINT	the_geom	







**Usando Shp2Pgsql:** Simplemente desde la línea de comandos ejecutar:

**\$ shp2pgsql example\_2\_poligono example\_2\_poligono > example\_2\_poligono.sql**

```

andres@andres-desktop: ~
Archivo Editar Ver Terminal Solapas Ayuda
andres@andres-desktop:~$ shp2pgsql example_2_poligono example_2_poligono > ex
ample_2_poligono.sql
Shapefile type: Polygon
Postgis type: MULTIPOLYGON[2]
andres@andres-desktop:~$

```

2) Ahora para exportar a nuestra base de datos (**tutopostgis**) el archivo SQL, generado (**example\_2\_poligono.sql**) , ejecutar:

**\$ psql -h localhost -U postgres -d tutopostgis < example\_2\_poligono.sql**

```

andres@andres-desktop: ~
Archivo Editar Ver Terminal Solapas Ayuda
andres@andres-desktop:~$ psql -h localhost -U postgres -d tutopostgis < exam
ple_2_poligono.sql
BEGIN
NOTICE: CREATE TABLE creará una secuencia implícita «example_2_poligono_gid_
seq» para la columna serial «example_2_poligono.gid»
NOTICE: CREATE TABLE / PRIMARY KEY creará el índice implícito «example_2_pol
igono_pkey» para la tabla «example_2_poligono»
CREATE TABLE
          addgeometrycolumn
-----
public.example_2_poligono.the_geom SRID:-1 TYPE:MULTIPOLYGON DIMS:2
(1 fila)

INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
COMMIT
andres@andres-desktop:~$

```



**Usando FwTools:** Otra forma de exportar rápidamente nuestro archivo shp a la base de datos, es haciendo uso de la herramienta **ogr2ogr**.

1) Inicar FWtools: **Inicio -> Programas ->FWTools-> FWtools Shell** y ejecutar la siguiente línea:

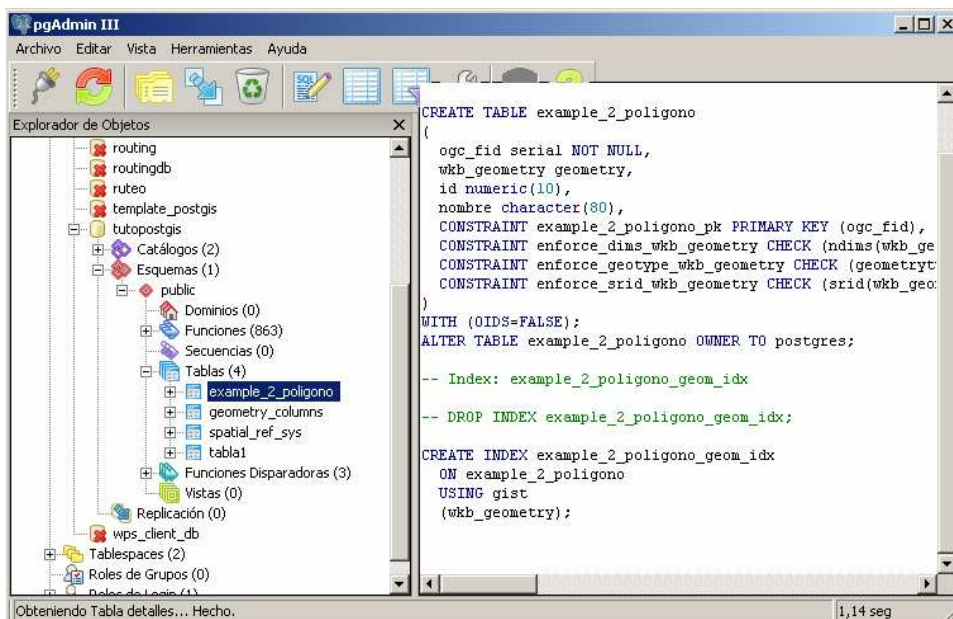
**\$ ogr2ogr -f "PostgreSQL" PG:"host=localhost user=postgres dbname=tutopostgis" example\_2\_poligono.shp**

```

C:\FWTools Shell
D:\ANDRESHERRERA\PUBLICACIONES\POSTGIS\data\shp>ogr2ogr -f "PostgreSQL" PG:"host
=192.168.1.11 user=postgres dbname=tutopostgis" example_2_poligono.shp
D:\ANDRESHERRERA\PUBLICACIONES\POSTGIS\data\shp>_

```

2) Verificar la creación de la tabla.



Usando FWTools, instalado en el entorno, simplemente corremos la siguiente línea:

**\$ ogr2ogr -f "PostgreSQL" PG:"host=localhost user=postgres dbname=tutopostgis" example\_2\_poligono.shp**

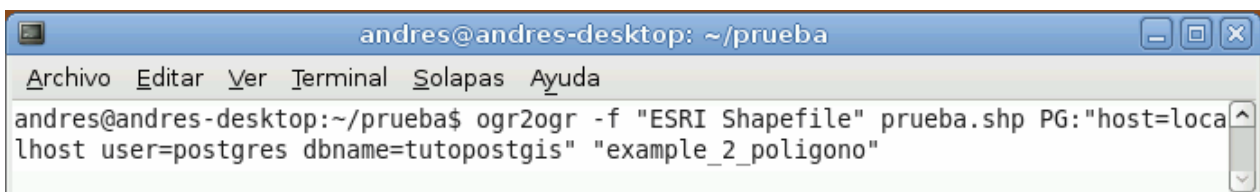


## Importando Información de PostgreSQL a formato ESRI SHP



Usando FWTools: Tanto en Windows como en Linux, ejecutamos el comando **ogr2ogr** con la siguiente sintaxis:

**\$ ogr2ogr -f "ESRI Shapefile" prueba.shp PG:"host=localhost user=postgres dbname=tutopostgis" "example\_2\_poligono"**

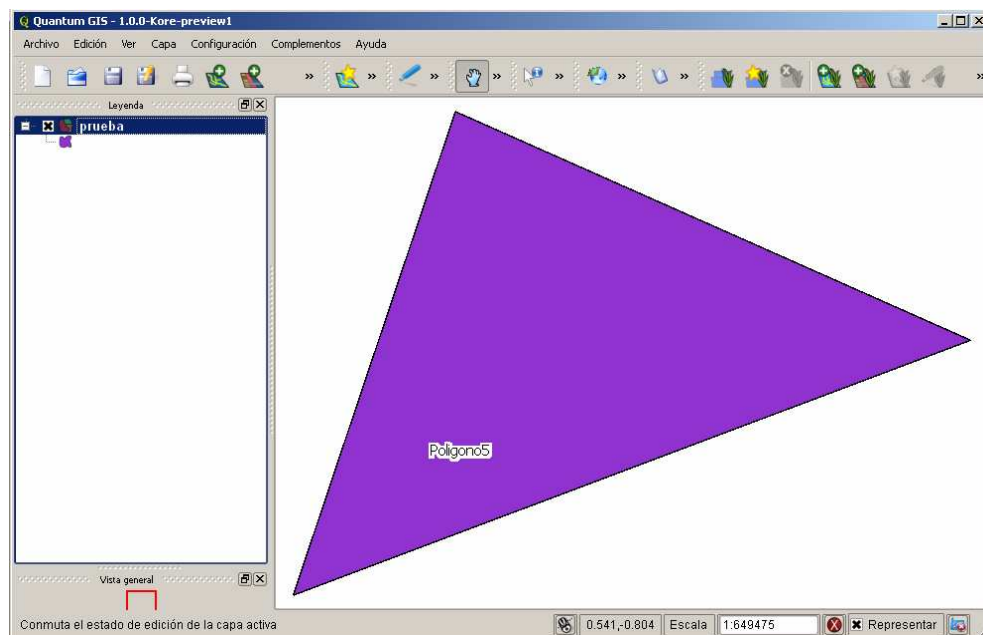


Este nos entrega como resultado, el volcamiento de la tabla espacial seleccionada en formato ESRI SHP.

```
andres@andres-desktop: ~/prueba
Archivo  Editar  Ver  Terminal  Solapas  Ayuda
andres@andres-desktop:~/prueba$ ls
prueba.dbf  prueba.prj  prueba.shp  prueba.shx
andres@andres-desktop:~/prueba$
```

**Filtrando:** variando un poco la sintaxis del comando, podemos importar la información aplicando filtros SQL.

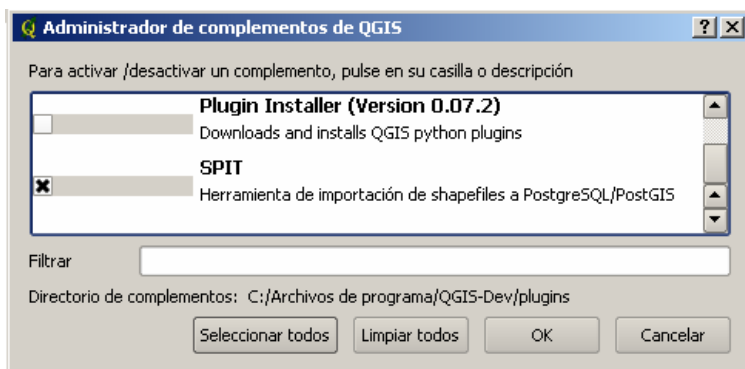
**\$ ogr2ogr -f "ESRI Shapefile" prueba.shp PG:"host=localhost user=postgres dbname=tutopostgis" -sql "SELECT \* FROM example\_2\_poligono where id=5"**



**Usando QGIS:** También podemos usar QGIS como herramienta para importar y exportar de SHP a PostgreSQL y de PostgreSQL a SHP, simplemente debemos seguir esta serie de pasos.

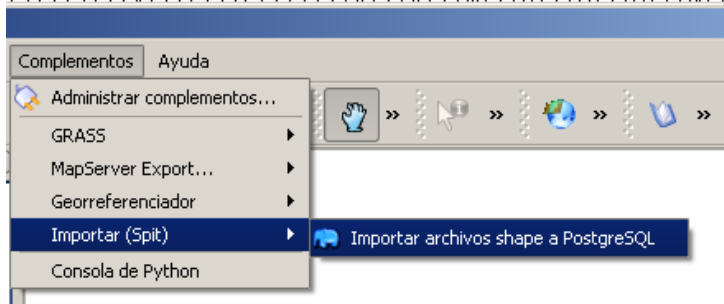
## Exportar a PostGIS.

- 1) Iniciar QGIS.
- 2) Iniciar dialogo de Administración de complementos.

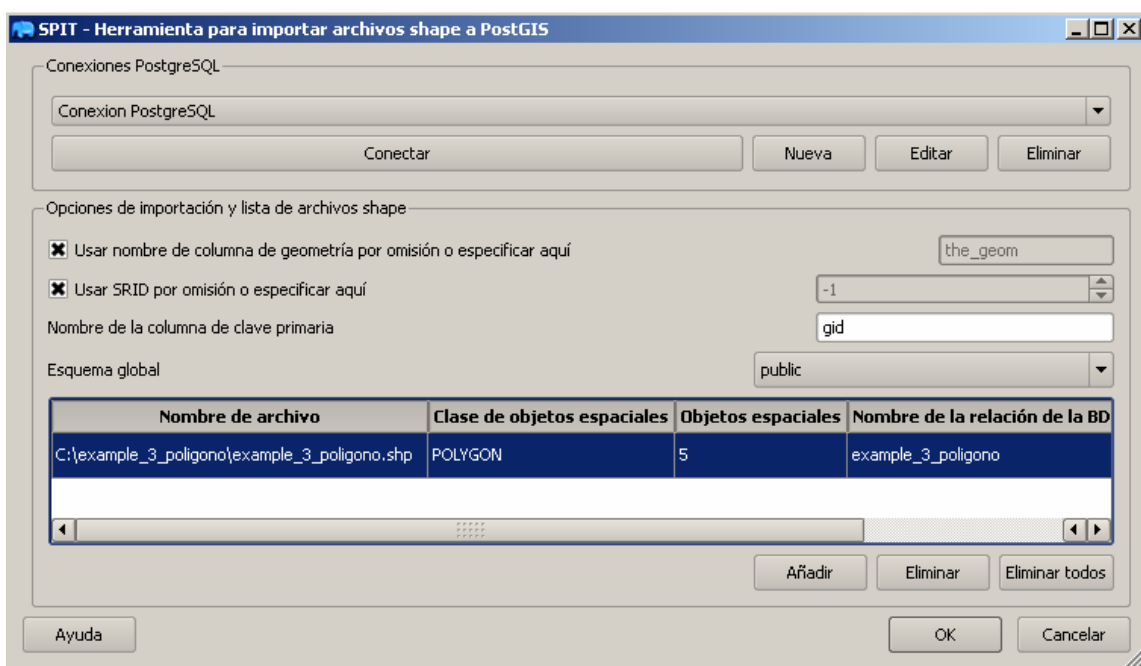


- 3) Activar complemento SPIT.

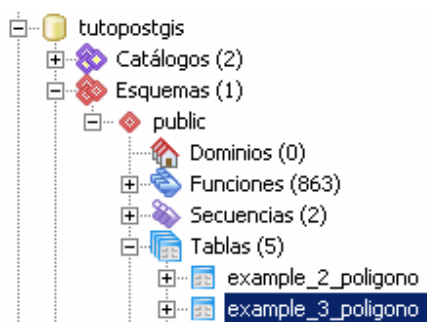
#### 4) Iniciar dialogo Importar (Spit).



#### 5) Realizar conexión con base de datos PostgreSQL (Conectar), (Añadir) SHP a exportar.



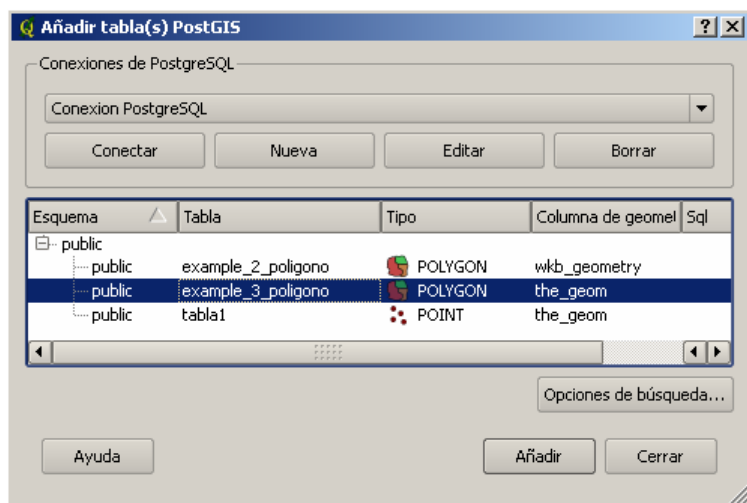
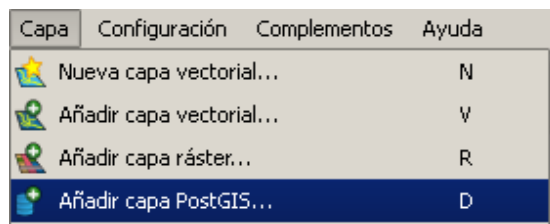
#### 6) Verificar en la base de datos PostgreSQL, que **example\_3\_poligono** se ha exportado correctamente.



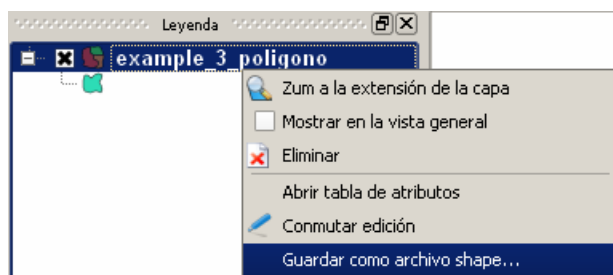
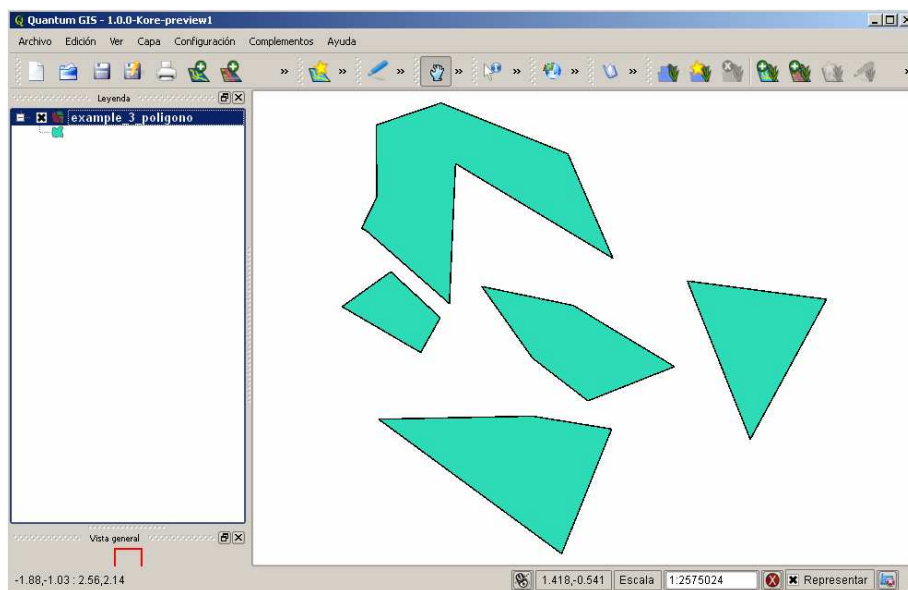


## Importar de PostgreSQL a SHP:

- 1) Iniciamos dialogo para añadir una nueva capa PostGIS. Capa -> Añadir capa PostGIS.



- 2) Realizamos conexión con la base de datos (**Conectar**), seleccionamos tabla a importar (**example\_3\_poligono**) y presionamos ( **Añadir**).



- 4) Click derecho sobre (layer: **example\_3\_poligono**) y presionar sobre (**Guardar como archivo shape...**).



## Geometrías

PostGIS internamente almacena, las geometrías espaciales en columnas, serializa estas en un formato binario denominado (**WKB**) Well Know Binary.

### PUNTO

```
CREATE TABLE puntos_prueba ( gid serial NOT NULL, id int4, nombre varchar(20),  
CONSTRAINT puntos_prueba_pkey PRIMARY KEY (gid) );  
SELECT AddGeometryColumn('puntos_prueba','the_geom',-1,'POINT',2);  
INSERT INTO puntos_prueba (id, nombre, the_geom) VALUES (1, 'Primer Punto',  
GeomFromText('POINT(1 5)'));
```

### LINESTRING

```
CREATE TABLE linestring_prueba ( gid serial NOT NULL, id int4, nombre varchar(20),  
CONSTRAINT linestring_prueba_pkey PRIMARY KEY (gid) );  
SELECT AddGeometryColumn('linestring_prueba','the_geom',-1,'LINESTRING',2);  
INSERT INTO linestring_prueba (id, nombre, the_geom) VALUES (1, 'Primer LineString',  
GeomFromText('LINESTRING (0 0, 1 5 , 6 8)'));
```

### MULTILINESTRING

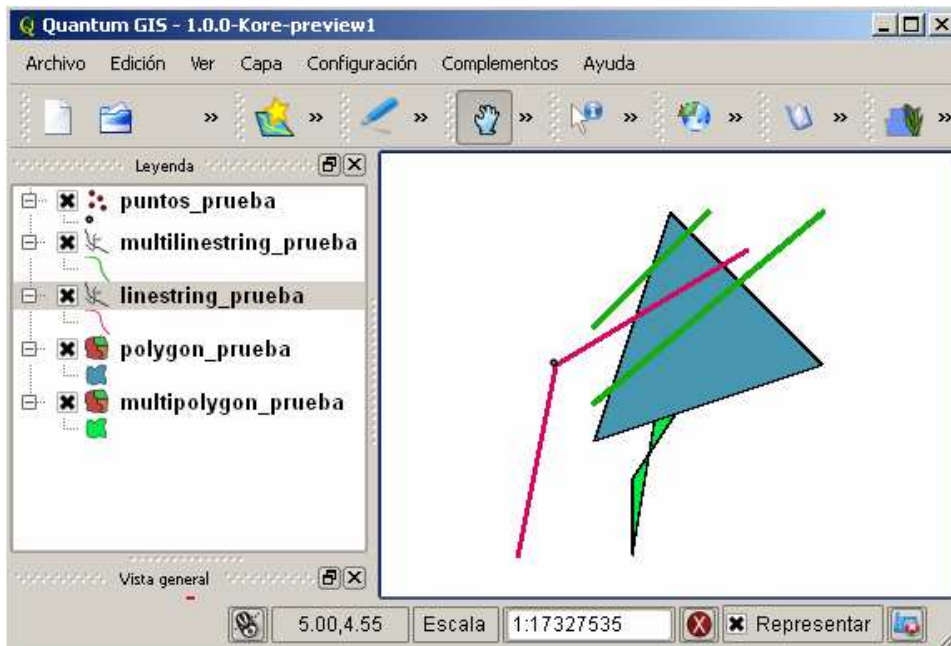
```
CREATE TABLE multilinestring_prueba ( gid serial NOT NULL, id int4, nombre varchar(30),  
CONSTRAINT multilinestring_prueba_pkey PRIMARY KEY (gid) );  
SELECT AddGeometryColumn('multilinestring_prueba','the_geom',-  
1,'MULTILINESTRING',2);  
INSERT INTO multilinestring_prueba (id, nombre, the_geom) VALUES (1, 'Primer  
MultiLineString', GeomFromText('MULTILINESTRING((2 4, 8 9), (2 6, 5 9)')));
```

### POLYGON

```
CREATE TABLE polygon_prueba ( gid serial NOT NULL, id int4, nombre varchar(20),  
CONSTRAINT polygon_prueba_pkey PRIMARY KEY (gid) );  
SELECT AddGeometryColumn('polygon_prueba','the_geom',-1,'POLYGON',2);  
INSERT INTO polygon_prueba (id, nombre, the_geom) VALUES (1, 'Primer Poligono',  
GeomFromText('POLYGON(( 2 3, 8 5, 4 9, 3 6 , 2 3 ))'));
```

### MULTIPOLYGON

```
CREATE TABLE multipolygon_prueba ( gid serial NOT NULL, id int4, nombre varchar(30),  
CONSTRAINT multipolygon_prueba_pkey PRIMARY KEY (gid) );  
SELECT AddGeometryColumn('multipolygon_prueba','the_geom',-1,'MULTIPOLYGON',2);  
INSERT INTO multipolygon_prueba (id, nombre, the_geom) VALUES (1, 'Primer MultiPoligono',  
GeomFromText('MULTIPOLYGON( (( 2 3, 8 5, 4 9, 3 6 , 2 3 )) , (( 4 6, 5 5, 3 2, 3 0 , 4 6 )) )'));
```



## Algunas Funciones básicas de PostGIS

Algunas funciones básicas de PostGIS, comunes y útiles a la hora de desarrollar aplicaciones complejas.

**Calculo de Area:** Calculamos el área de una geometría Cerrada (**Polígono**).

```
SELECT area(the_geom) as "Area" FROM example_2_poligono;
```

```
tutopostgis=# select area(the_geom) as "Area" from example_2_poligono;
Area
-----
0.552920420343564
0.966642907598125
0.763425418709411
0.416345906551526
0.353739415665932
<5 rows>
```

**Calculo de Perímetro:** Calculamos el perímetro de una geometría cerrada (**Polígono**).

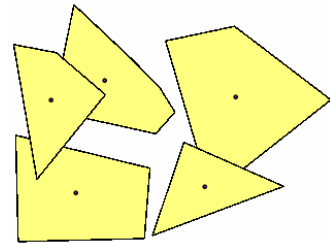
```
SELECT perimeter(the_geom) as "Perimetro" FROM example_2_poligono;
```

```
tutopostgis=# select perimeter(the_geom) as "Perimetro" from example_2_poligono;
Perimetro
-----
3.18499570718423
4.10657316302225
3.59185896168714
2.87648123418619
2.82794639085621
<5 rows>
```

**Calculo de Centroide:** Calculamos el centroide de cada polígono y lo representamos en OGC WKB, igualmente podemos extraer las coordenadas de dicho centroide.

```
select centroid(the_geom) as "Centroide WKB",
       x(centroid(the_geom)) as "Coord X Centroide",
       y(centroid(the_geom)) as "Coord Y Centroide" from example_2_poligono;
```

	Centroide WKB geometry	Coord X Centroide double precision	Coord Y Centroide double precision
1	0101000000DA0722276	-0.638112141067036	0.324137083502631
2	010100000006AEE7794C	0.425372475669718	0.185085236886242
3	0101000000005ADF92E9	-0.873644625530517	-0.592741301846271
4	0101000000C3C67A7AC	-1.07758281558601	0.167135440761013
5	0101000000B827A973C	0.175178121532914	-0.53721290603427



**Información de Dimensión:** Obtenemos la dimensión de la geometría, en este caso: **2D**

```
SELECT dimension(the_geom) as "Dimension" FROM example_2_poligono;
```

```
tutopostgis=# select dimension(the_geom) as "Dimension" from example_2_poligono;
 Dimension
-----
          2
          2
          2
          2
          2
<5 rows>
```

**Información tipo geometría:** Obtenemos el tipo de geometría.

```
SELECT geometrytype(the_geom)as "Tipo Geometria" from example_2_poligono;
```

```
tutopostgis=# select geometrytype(the_geom) as "Tipo Geometria" from example_2_poligono;
 Tipo Geometria
-----
MULTIPOLYGON
MULTIPOLYGON
MULTIPOLYGON
MULTIPOLYGON
MULTIPOLYGON
<5 rows>
```

**Información SRID:** Obtenemos el SRID correspondiente a la geometría.

```
SELECT getsrid(the_geom)as "SRID" from example_2_poligono;
```

```
tutopostgis=# select getsrid(the_geom) as "Tipo Geometria" from example_2_poligono;
 Tipo Geometria
-----
          -1
          -1
          -1
          -1
          -1
<5 rows>
```

**Información NPoints:** Obtenemos el número de vértices de la geometría.

```
SELECT npoints(the_geom) as "NPOINTS" from example_2_poligono;
```

```
tutopostgis=# select npoints<the_geom> as "NPOINTS" from example_2_poligono;
 NPOINTS
-----
      10
       5
       7
       5
       6
(5 rows)
```

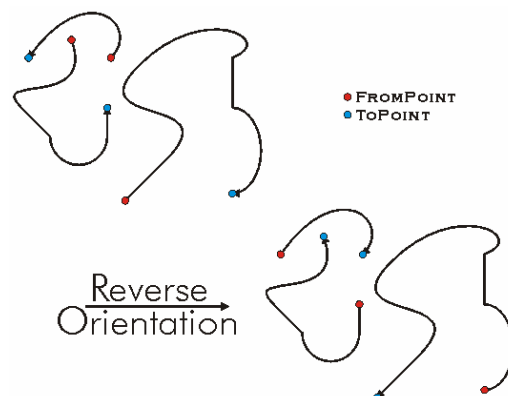
**Revertir:** Revertimos la dirección de una geometría.

```
SELECT reverse(the_geom) as "Reverse WKB", asText(reverse(the_geom)) as "Reverse Text" ,
asText((the_geom)) as "Normal" from example_2_poligono;
```

Muy útil en geometrías de tipo (**lineal**). Veamos claramente<sup>4</sup>:

```
SELECT GeometryFromText('LINESTRING(1 0 , 5 0)') as "Linea WKB" ,
asText(GeometryFromText('LINESTRING(1 0 , 5 0)')) as "Linea Text",
reverse(GeometryFromText('LINESTRING(1 0 , 5 0)')) as "Reverse Linea WKB",
asText(reverse(GeometryFromText('LINESTRING(1 0 , 5 0)'))) as "Reverse Linea Text";
```

Salida de datos		Comentar	Mensajes	Historial
	Linea WKB geometry	Linea Text text	Reverse Linea WKB geometry	Reverse Linea WKB text
1	010200000000200000000000	LINESTRING(1 0,5 0)	01020000000020000000000000	LINESTRING(5 0,1 0)



**Rotar:** Rotamos una geometría.

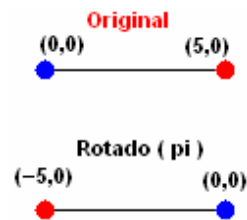
```
select rotate(the_geom , 45 ) as "Rotate WKB" , astext(rotate(the_geom , 45 )) as "Rotate Text"
from example_2_poligono;
```

Veamos otro ejemplo:

```
select asText(GeometryFromText('LINESTRING(0 0,5 0)')) as "Original Text",
astext(rotate(GeometryFromText('LINESTRING(0 0 , 5 0)') , pi() )) as "Rotate Text";
```

<sup>4</sup> Grafica tomada de <http://edndoc.esri.com/arcobjects/9.2/java/api/arcobjects/com/esri/arcgis/geometry/>

Salida de datos   Comentar   Mensajes   Historial		
	Original Text text	Rotate Text text
1	<code>LINESTRING(0 0,5 0)</code>	<code>LINESTRING(0 0,-5 0)</code>



## Representación geometría en GEOJSON:

```
select st_asgeojson(the_geom) as "GeoJSON" from example_2_poligono;
```

	GeoJSON text
1	<code>{"type":"MultiPolygon","coordinates":[[[[[-0.905090497737557,0.9549773755656</code>
2	<code>{"type":"MultiPolygon","coordinates":[[[[[0.401470588235294,0.77918552036199</code>
3	<code>{"type":"MultiPolygon","coordinates":[[[[[-1.378754862417878,-0.1071677919966</code>
4	<code>{"type":"MultiPolygon","coordinates":[[[[[-1.395242215032740,0.6306412375184</code>
5	<code>{"type":"MultiPolygon","coordinates":[[[[[0.801697520897690,-0.5234734455218</code>

## Representación geometría en GML:

```
select st_asgml(the_geom) as "GML" from example_2_poligono;
```

	GML text
1	<code>&lt;gml:MultiPolygon&gt;&lt;gml:polygonMember&gt;&lt;gml:Polygon&gt;&lt;gml:outerBoundaryIs&gt;&lt;gml:LinearRing&gt;&lt;gml:coordinates&gt;-0.905090497737557;0.954977375565611;-0.425226244343891;-0.50361990950226</code>
2	<code>&lt;gml:MultiPolygon&gt;&lt;gml:polygonMember&gt;&lt;gml:Polygon&gt;&lt;gml:outerBoundaryIs&gt;&lt;gml:LinearRing&gt;&lt;gml:coordinates&gt;0.401470588235294;0.779185520361991;1.185407239819005;-0.180542986425339</code>
3	<code>&lt;gml:MultiPolygon&gt;&lt;gml:polygonMember&gt;&lt;gml:Polygon&gt;&lt;gml:outerBoundaryIs&gt;&lt;gml:LinearRing&gt;&lt;gml:coordinates&gt;-1.378754862417878;-0.107167791996606;-0.286467751683236;0.375087271988122</code>
4	<code>&lt;gml:MultiPolygon&gt;&lt;gml:polygonMember&gt;&lt;gml:Polygon&gt;&lt;gml:outerBoundaryIs&gt;&lt;gml:LinearRing&gt;&lt;gml:coordinates&gt;-1.39524221503274;-0.630641237518491;-1.049007810120628;-0.564691827059041</code>
5	<code>&lt;gml:MultiPolygon&gt;&lt;gml:polygonMember&gt;&lt;gml:Polygon&gt;&lt;gml:outerBoundaryIs&gt;&lt;gml:LinearRing&gt;&lt;gml:coordinates&gt;0.80169752089769;0.523473445521885;-0.265858560914658;0.923291746432301;-0</code>

## Representación geometría en SVG:

```
select st_assvg(the_geom) as "SVG" from example_2_poligono;
```

	SVG text
1	<code>M -0.905090497737557 -0.954977375565611 -0.425226244343891 -0.50361990950226</code>
2	<code>M 0.401470588235294 -0.779185520361991 1.185407239819005 -0.180542986425339</code>
3	<code>M -1.378754862417878 0.107167791996606 -0.286467751683236 0.375087271988122</code>
4	<code>M -1.39524221503274 -0.630641237518491 -1.049007810120628 -0.564691827059041</code>
5	<code>M 0.80169752089769 0.523473445521885 -0.265858560914658 0.923291746432301 -0</code>

## Representación geometría en Text:

```
select st_astext(the_geom) as "TEXT" from example_2_poligono;
```

	TEXT text
1	<code>MULTIPOLYGON((( (-0.905090497737557 0.954977375565611,-0.425226244343891 -0.50361990950226</code>
2	<code>MULTIPOLYGON((( (0.401470588235294 0.779185520361991,1.185407239819005 -0.180542986425339</code>
3	<code>MULTIPOLYGON((( (-1.37875486241788 -0.107167791996606,-0.286467751683236 0.375087271988122</code>
4	<code>MULTIPOLYGON((( (-1.39524221503274 0.630641237518491,-1.049007810120628 -0.564691827059041</code>
5	<code>MULTIPOLYGON((( (0.80169752089769 -0.523473445521885,-0.265858560914658 0.923291746432301 -0</code>



**Distancia:** Selecciona todas las geometrías de la columna the\_geom que estén a una distancia menor a **0.3** unidades del punto (0,0).

```
SELECT id,nombre,the_geom , astext(the_geom) FROM example_2_poligono
WHERE ST_Distance(the_geom, GeomFromText('POINT(0 0)', -1)) < 0.3;
```

	id bigint	nombre character var	the_geom geometry	astext text
1	1	Poligono1	010600000001C	MULTIPOLYGON
2	2	Poligono2	010600000001C	MULTIPOLYGON
3	5	Poligono5	010600000001C	MULTIPOLYGON

**Obtener Extend:** Obteniendo la máxima extensión de la capa.

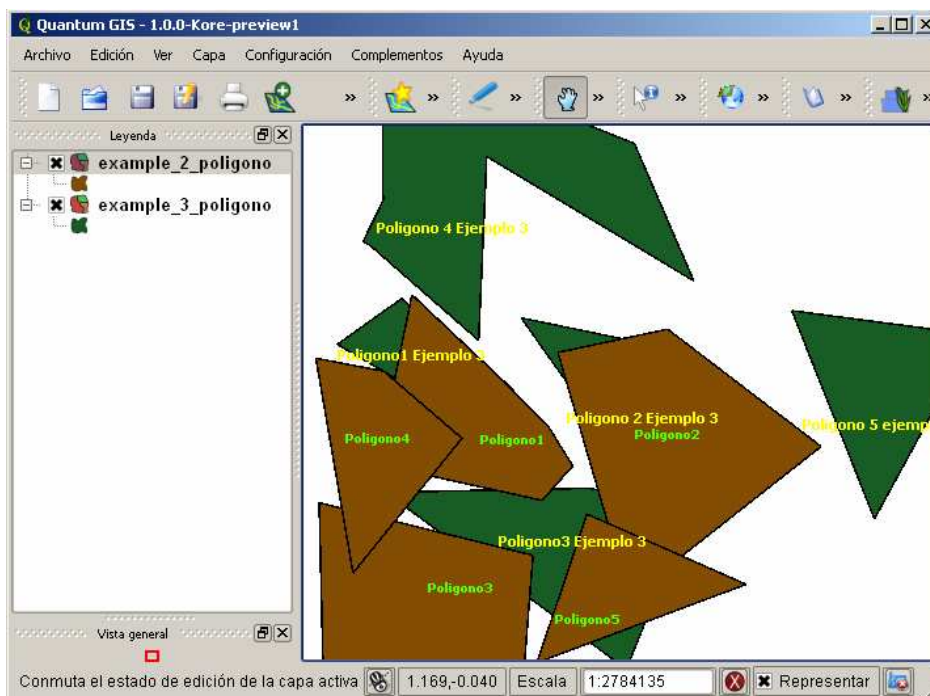
```
SELECT st_extent(the_geom), xmin(st_extent(the_geom)) , ymin(st_extent(the_geom)) ,
xmax(st_extent(the_geom)) , ymax(st_extent(the_geom))from example_2_poligono;
```

	st_extent box2d	xmin double precis	ymin double precis	xmax double precis	ymax double precis
1	BOX(-1.3952421	-1.3952423334	-0.97275382280	1.18540728092	0.95497739315

**Nota:** Podrá encontrar las fuentes de estos ejemplos, en el archivo: **example\_3.sql**

### Consultas y relaciones espaciales:

En QGIS cargaremos las capas PostGIS ( **example\_2\_poligono** y **example\_3\_poligono** ).



### Sobre posición (Overlaps) Operador-> **&&** :

- Encuentra las geometrías que se sobreponen al box **BOX3D(0 5,5 0)**.

```
SELECT id,nombre,the_geom FROM example_2_poligono WHERE the_geom && 'BOX3D(0 5,5 0)::box3d;
```

	id bigint	nombre character var	the_geom geometry
1	2	Poligono2	0106000000001C

- Encuentra las geometrías que se sobreponen al punto **POINT (-1 0)**.

```
SELECT id,nombre,the_geom FROM example_2_poligono WHERE the_geom && GeomFromText('POINT(-1 0)', -1);
```

	id bigint	nombre character var	the_geom geometry
1	1	Poligono1	0106000000001C
2	4	Poligono4	0106000000001C

### Geométricamente Idénticas Operador-> **~=** :

Mediante este operador podemos testear si una geometría es idénticamente a otra, de tal forma que si estas son idénticas, obtenemos una respuesta booleana (**True**).

```
SELECT GeomFromText('POLYGON((0 1,1 1,1 0,0 1))') ~= GeomFromText('POLYGON((0 1,1 1,1 0,0 1))') as identical;
```

```
tutopostgis=# SELECT GeomFromText('POLYGON((0 1,1 1,1 0,0 1))') ~= GeomFromText('POLYGON((0 1,1 1,1 0,0 1))') as identical;
            identical
-----
t
<1 row>
```

### Geométricamente Iguales Operador -> **=** :

Mediante este operador postgis verifica si una geometría tiene un bounding box idéntico al comparado.

```
SELECT id,nombre,the_geom FROM example_2_poligono
WHERE the_geom = (GeomFromText('MULTIPOLYGON(((0.80169752089769 -
0.523473445521885,-0.265858560914658 -0.923291746432301,-0.265858560914658 -
0.923291746432301, -0.265858560914658 -0.923291746432301,-0.0103045953842892
-0.164873526148625,0.80169752089769 -0.523473445521885)))', -1));
```

	id bigint	nombre character var	the_geom geometry
1	5	Poligono5	0106000000001C

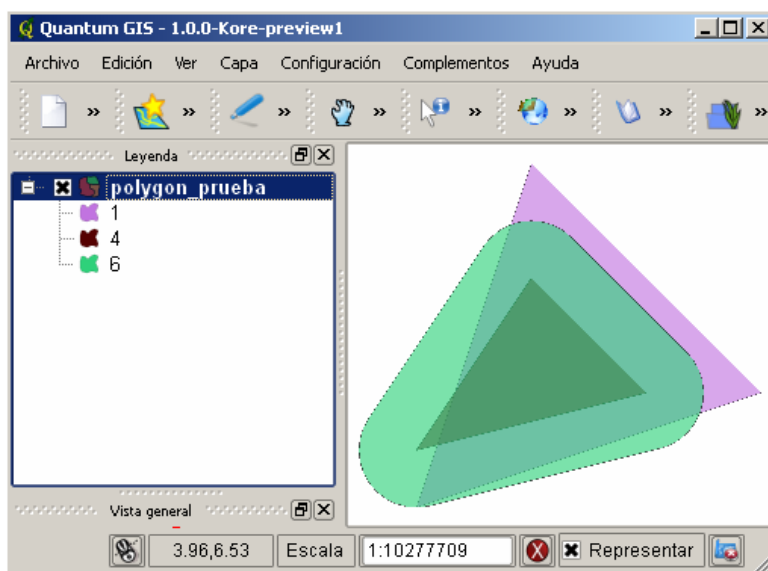
**Buffer:** Se genera un buffer o área de influencia alrededor de un elemento especial.

```
SELECT buffer(GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))') , 4 ) , asText(  
buffer(GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))') , 4 ) );
```

	buffer geometry	astext text
1	010300000001000000240000	POLYGON((6.82842712474619 9.82842712474619

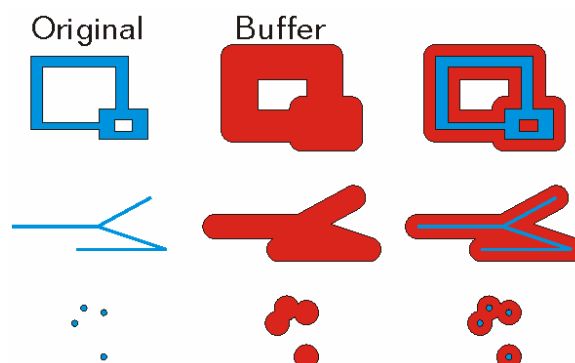
Visualicemos un buffer espacialmente vía QGIS, usando la tabla **polygon\_prueba** para almacenar el resultado.

```
INSERT INTO polygon_prueba (id, nombre, the_geom) VALUES (4, 'Poligono 2',  
GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))'));  
INSERT INTO polygon_prueba (id, nombre, the_geom) VALUES (6, 'Poligono Buffer 4',  
buffer(GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))') , 2) );
```



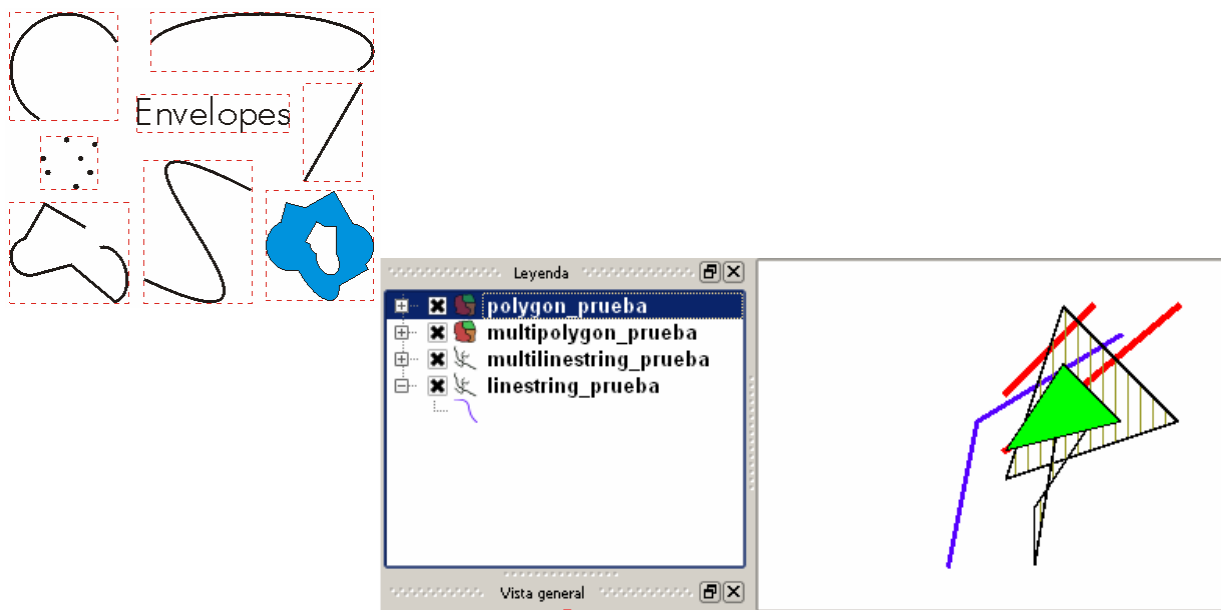
4-> Polígono original.  
6-> Buffer a polígono.

- Buffer aplicado a diferentes entidades.<sup>5</sup>

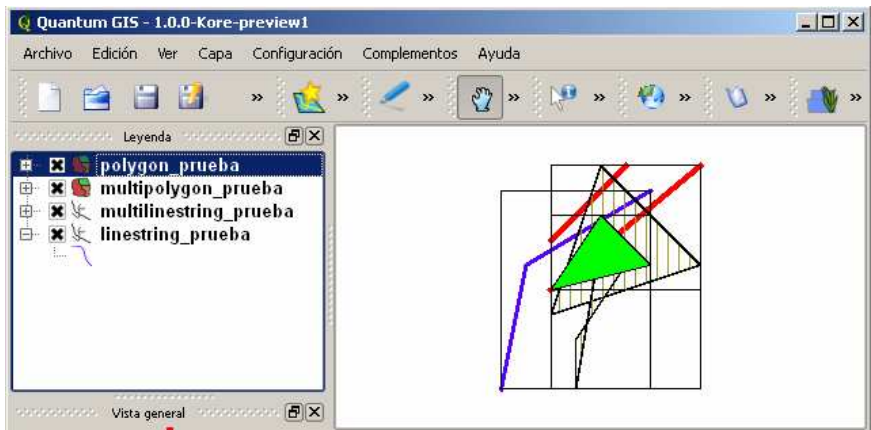


<sup>5</sup> Grafica tomada de: <http://edndoc.esri.com/arcobjects/9.2/java/api/arcobjects/com/esri/arcgis/>

**Envelope<sup>6</sup>:** Retorna una geometría envolvente.



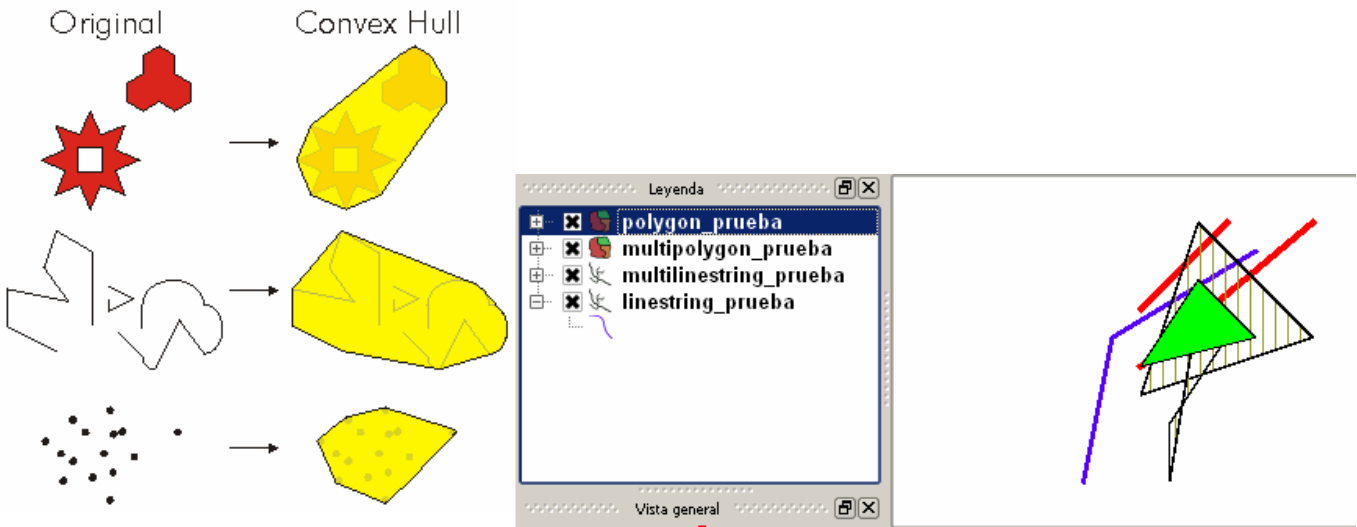
```
SELECT asText(envelope(GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))' ) ),
asText(envelope(GeomFromText('POINT(1 5)' ) ) ,
asText(envelope(GeomFromText('LINESTRING (0 0, 1 5 , 6 8)' ) ) ,
asText(envelope(GeomFromText('MULTILINESTRING((2 4, 8 9), (2 6, 5 9))' ) ) ,
asText(envelope(GeomFromText('MULTIPOLYGON( (( 2 3, 8 5, 4 9, 3 6 , 2 3 ) ) , (( 4 6, 5
5, 3 2, 3 0 , 4 6 ) )' ) ) ) );
```



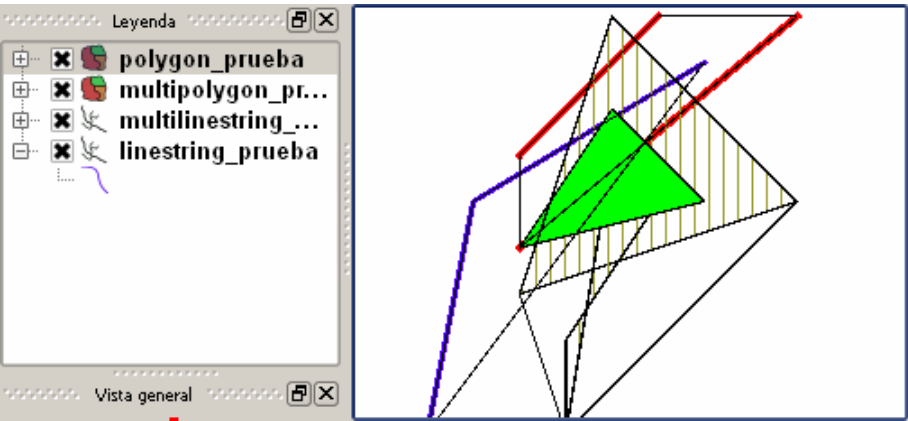
	astext text	astext text	astext text	astext text	astext text
1	POLYGON((2 4,2 7,6 7,6 4,2 4))	POINT(1 5)	POLYGON((0 0,0 8,6 8,6 0,0 0))	POLYGON((2 4,2 9,8 9,8 4,2 4))	POLYGON((2 0,2 9,8 9,8 0,2 0))

<sup>6</sup> Grafica tomada de: <http://edndoc.esri.com/arcobjects/9.2/java/api/arcobjects/com/esri/arcgis/>

**Conex Hull** <sup>7</sup>: Retorna la envolvente convexa de una geometría.



```
SELECT asText(convexhull(GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))' ))),
asText(convexhull(GeomFromText('POINT(1 5)') ) ) ,
asText(convexhull(GeomFromText('LINESTRING (0 0, 1 5 , 6 8)') ) ),
asText(convexhull(GeomFromText('MULTILINESTRING((2 4, 8 9), (2 6, 5 9))') ) ),
asText(convexhull(GeomFromText('MULTIPOLYGON( (( 2 3, 8 5, 4 9, 3 6 , 2 3 )) , (( 4 6,
5 5, 3 2, 3 0 , 4 6 )) ')') ) );
```



	astext text	astext text	astext text	astext text	astext text
1	POLYGON((2 4,4 7,6 5,2 4))	POINT(1 5)	POLYGON((0 0,1 5,6 8,0 0))	POLYGON((2 4,2 6,5 9,8 9,2 4))	POLYGON((3 0,2 3,4 9,8 5,3 0))

<sup>7</sup> Grafica tomada de: <http://edndoc.esri.com/arcobjects/9.2/java/api/arcobjects/com/esri/arcgis/>

**Touch**<sup>8</sup>: Retorna (TRUE) si las geometrías " presentan un toque espacial".

Base Geometry			
Comparison Geometry			
	No touch relationship possible		

```
SELECT touches(GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))' ) ,
GeomFromText('POINT(4 7)' ) );
```

```
tutopostgis=# SELECT touches<GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))' )
, GeomFromText('POINT(4 7)' ) >;
touches
-----
t
<1 row>
```

```
SELECT touches(GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))' ) ,
GeomFromText('LINESTRING (0 0, 6 5 , 6 8)' ) );
```

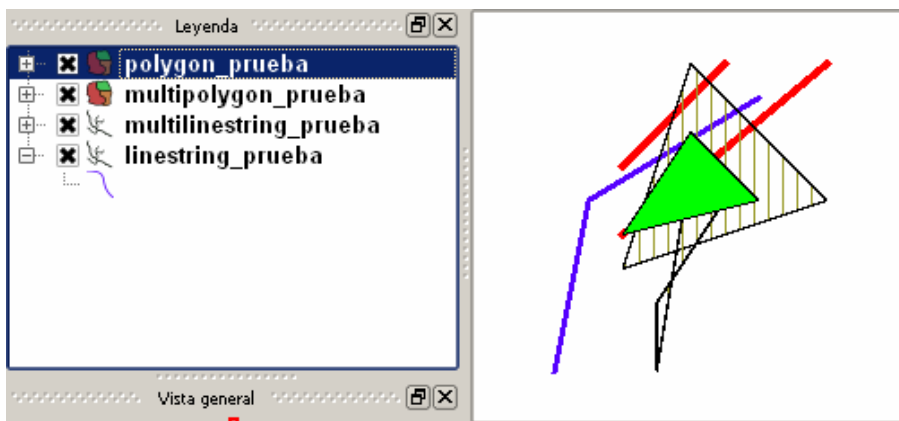
```
tutopostgis=# SELECT touches<GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))' )
, GeomFromText('LINESTRING (0 0, 6 5 , 6 8)' ) >;
touches
-----
t
<1 row>
```

```
SELECT touches(GeomFromText('LINESTRING (0 0, 6 5 , 6 8)' ) , GeomFromText('POINT(0 0)' )
);
```

```
tutopostgis=# SELECT touches<GeomFromText('LINESTRING (0 0, 6 5 , 6 8)' ) , GeomF
romText('POINT(0 0)' ) >;
touches
-----
t
<1 row>
```

<sup>8</sup> Grafica tomada de: <http://edndoc.esri.com/arcobjects/9.2/java/api/arcobjects/com/esri/arcgis/>





```
SELECT touches( ( select the_geom from polygon_prueba ) , ( select the_geom from
multilinestring_prueba ) );
```

```
tutopostgis=# SELECT touches< < select the_geom from polygon_prueba > , < select
the_geom from multilinestring_prueba > >;
touches
-----
f
<1 row>
```

**Contains**<sup>9</sup>: Retorna (TRUE) si las geometrías " presentan una contención espacial".

Base Geometry			
Comparison Geometry		Point	Line
		No containment relationship possible	
		No containment relationship possible	

```
SELECT contains(GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))') ,
GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))') );
```

```
tutopostgis=# SELECT contains<GeomFromText<'POLYGON<<4 7,2 4,6 5,6 5,6 5,4 7>>'>
, GeomFromText<'POLYGON<<4 7,2 4,6 5,6 5,6 5,4 7>>'>> >;
contains
-----
t
<1 row>
```

<sup>9</sup> Grafica tomada de: <http://edndoc.esri.com/arcobjects/9.2/java/api/arcobjects/com/esri/arcgis/>

```
SELECT contains(GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))'),
GeomFromText('POLYGON((0 0,4 4,6 2,6 3,6 1,0 0))') );
```

```
tutopostgis=# SELECT contains<GeomFromText<'POLYGON<<4 7,2 4,6 5,6 5,6 5,4 7>>'>
, GeomFromText<'POLYGON<<0 0,4 4,6 2,6 3,6 1,0 0>>'> >;
contains
-----
f
<1 row>
```

```
SELECT contains(GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))'),
GeomFromText('LINESTRING (0 0, 6 5 , 6 8)') );
```

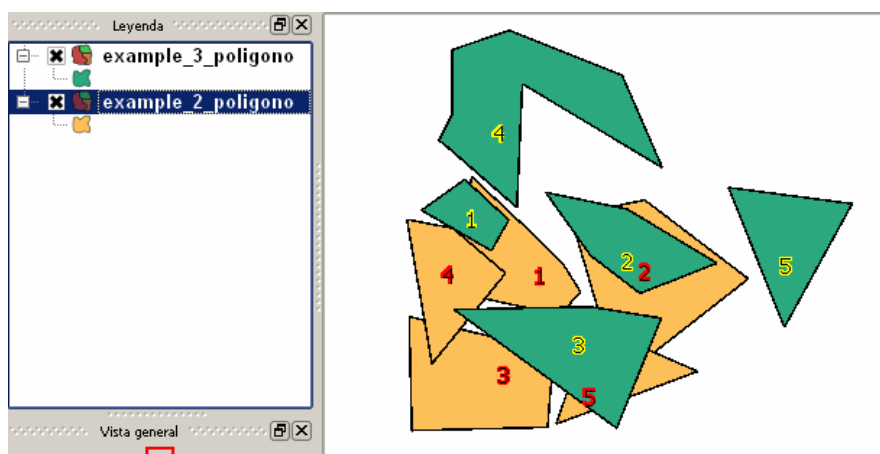
```
tutopostgis=# SELECT contains<GeomFromText<'POLYGON<<4 7,2 4,6 5,6 5,6 5,4 7>>'>
, GeomFromText<'LINESTRING <0 0, 6 5 , 6 8>'> >;
contains
-----
f
<1 row>
```

```
SELECT contains(GeomFromText('POLYGON((10 7,2 4,6 45,45 5,6 9,10 7))'),
GeomFromText('LINESTRING (9 25, 23 12)') );
```

```
tutopostgis=# SELECT contains<GeomFromText<'POLYGON<<10 7,2 4,6 45,45 5,6 9,10 7>>'>
, GeomFromText<'LINESTRING <9 25, 23 12>'> >;
contains
-----
t
<1 row>
```







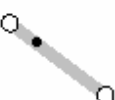
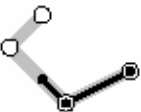


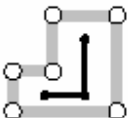

Consulta algo mas compleja, usando **envelope**, **geounion** y **contains** .

```
SELECT contains( envelope((SELECT geomunion( (select the_geom from example_3_poligono
where id=5) , (select the_geom from example_3_poligono where id=3)))) ,
( select the_geom from example_2_poligono where id=2 ));
```



```
tutopostgis=# SELECT contains< envelope<<SELECT geomunion< <select the_geom from
example_3_poligono where id=5> , <select the_geom from example_3_poligono where
id=3>>>> ,< select the_geom from example_2_poligono where id=2 >>;
contains
-----
t
<1 row>
```

**WITHIN**<sup>10</sup>: Retorna (TRUE) si las geometrías " presentan una contención espacial".

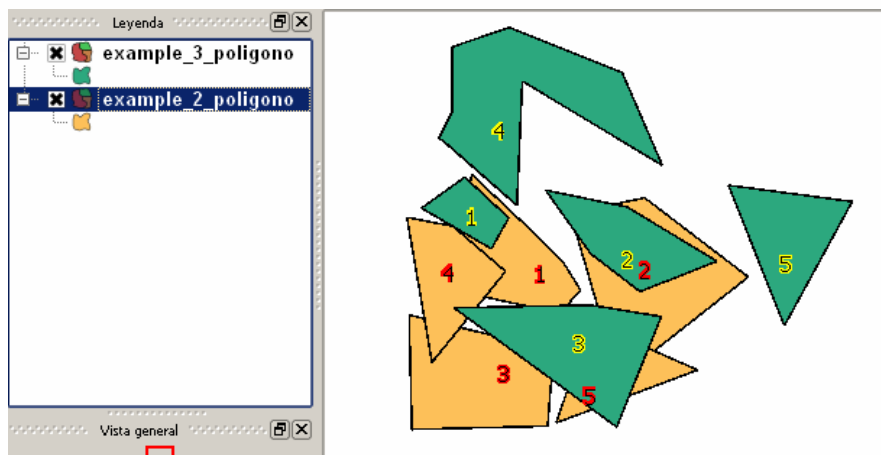
		Base Geometry		
				
Comparison Geometry			No within relationship possible	No within relationship possible
				No within relationship possible
				

```
SELECT within(GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))') ,
GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))') );
```

```
tutopostgis=# SELECT within<GeomFromText<'POLYGON<<4 7,2 4,6 5,6 5,6 5,4 7>>'> ,
  GeomFromText<'POLYGON<<4 7,2 4,6 5,6 5,6 5,4 7>>'> >;
  within
-----
t
<1 row>
```

```
SELECT within(GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))') ,
GeomFromText('POLYGON((0 0,4 4,6 2,6 3,6 1,0 0))') );
```

```
futopostgis=# SELECT within<GeomFromText<'POLYGON<<4 7,2 4,6 5,6 5,6 5,4 7>>'> ,
  GeomFromText<'POLYGON<<0 0,4 4,6 2,6 3,6 1,0 0>>'> >;
  within
-----
f
<1 row>
```



<sup>10</sup> Grafica tomada de: <http://edndoc.esri.com/arcobjects/9.2/java/api/arcobjects/com/esri/arcgis/>

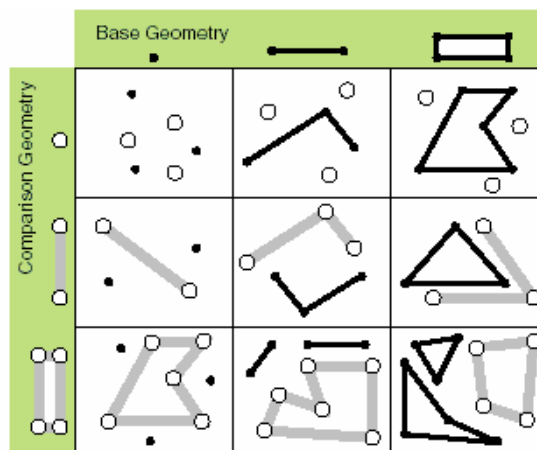
```
SELECT within(( select the_geom from example_2_poligono where id=1 ), envelope((SELECT
geomunion( (select the_geom from example_3_poligono where id=4) ,
(select the_geom from example_3_poligono where id=3)))) );
```

```
tutopostgis=# SELECT within<< select the_geom from example_2_poligono where id=1
> , envelope<<SELECT geomunion< <select the_geom from example_3_poligono where
id=4> , <select the_geom from example_3_poligono where id=3>>>> >;
within
-----
t
<1 row>
```

```
SELECT within(( select the_geom from example_2_poligono where id=1 ),
convexhull((SELECT geomunion( (select the_geom from example_3_poligono where id=4) ,
(select the_geom from example_3_poligono where id=3)))) );
```

```
tutopostgis=# SELECT within<< select the_geom from example_2_poligono where id=1
> , convexhull<<SELECT geomunion< <select the_geom from example_3_poligono whe
re id=4> , <select the_geom from example_3_poligono where id=3>>>> >;
within
-----
f
<1 row>
```

**DISTJOINT**<sup>11</sup>: Retorna (TRUE) si las geometrías “no presentan ninguna similitud espacial”, o no tienen elementos en común.



```
SELECT disjoint(GeomFromText('POLYGON(((4 7,2 4,6 5,6 5,6 5,4 7)))' ) ,
GeomFromText('POLYGON(((4 7,2 4,6 5,6 5,6 5,4 7)))' );
```

```
tutopostgis=# SELECT disjoint<GeomFromText<'POLYGON<<4 7,2 4,6 5,6 5,6 5,4 7>>>'>
, GeomFromText<'POLYGON<<4 7,2 4,6 5,6 5,6 5,4 7>>>'> >;
disjoint
-----
f
<1 row>
```

<sup>11</sup> Grafica tomada de: <http://edndoc.esri.com/arcobjects/9.2/java/api/arcobjects/com/esri/arcgis/>

```
SELECT disjoint(GeomFromText('POLYGON((0 0,2 4,6 2,4 5,6 5,0 0))') ,
GeomFromText('POINT(3 2)') );
```

```
tutopostgis=# SELECT disjoint<GeomFromText('POLYGON<<0 0,2 4,6 2,4 5,6 5,0 0>>')
, GeomFromText('POINT(3 2)')> >;
disjoint
-----
t
<1 row>
```

```
SELECT disjoint(GeomFromText('POLYGON((0 0,2 4,6 2,4 5,6 5,0 0))') ,
GeomFromText('LINESTRING (3 8, 6 9 , 6 8)') );
```

```
tutopostgis=# SELECT disjoint<GeomFromText('POLYGON<<0 0,2 4,6 2,4 5,6 5,0 0>>')
, GeomFromText('LINESTRING (3 8, 6 9 , 6 8)')> >;
disjoint
-----
t
<1 row>
```

**CROSSES**<sup>12</sup>: Retorna (TRUE) si las geometrías “presentan un cruce espacial”.

Base Geometry				
		●	—	▭
Comparison Geometry	○	No crossing relationship possible	No crossing relationship possible	No crossing relationship possible
	○ ○ ○	No crossing relationship possible		
	○ ○ ○ ○ ○	No crossing relationship possible		No crossing relationship possible

```
SELECT crosses(GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))') ,
GeomFromText('LINESTRING(3 3 , 4 5)') );
```

```
tutopostgis=# SELECT crosses<GeomFromText('POLYGON<<4 7,2 4,6 5,6 5,6 5,4 7>>')
, GeomFromText('LINESTRING(3 3 , 4 5)')> >;
crosses
-----
t
<1 row>
```

<sup>12</sup> Grafica tomada de: <http://edndoc.esri.com/arcobjects/9.2/java/api/arcobjects/com/esri/arcgis/>

```
SELECT crosses(GeomFromText('LINESTRING(3 3 , 4 5 , 5 4)'),
GeomFromText('LINESTRING(2 3 , 4 4 , 4 5 )') );
```

```
tutopostgis=# SELECT crosses<GeomFromText<'LINESTRING(3 3 , 4 5 , 5 4)') , GeomF
romText<'LINESTRING(2 3 , 4 4 , 4 5 )')>> );
crosses
-----
t
<1 row>
```

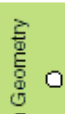





```
SELECT crosses(GeomFromText('LINESTRING(3 3 , 4 5 , 5 4)'),
GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))') );
```

```
tutopostgis=# SELECT crosses<GeomFromText<'LINESTRING(3 3 , 4 5 , 5 4)') , GeomF
romText<'POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))')>> );
crosses
-----
t
<1 row>
```

```
SELECT crosses(GeomFromText('LINESTRING(6 5, 7 4)'), GeomFromText('POLYGON((4 7,2
4,6 5,6 5,6 5,4 7))') );
```

```
tutopostgis=# SELECT crosses<GeomFromText<'LINESTRING(6 5, 7 4)') , GeomFromText
<'POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))')>> );
crosses
-----
f
<1 row>
```

**OVERLAPS<sup>13</sup>**: Retorna (TRUE) si las geometrías “presentan una superposición espacial”.

Base Geometry			
Comparison Geometry			
			
		No overlap relationship possible	No overlap relationship possible
		No overlap relationship possible	No overlap relationship possible
		No overlap relationship possible	No overlap relationship possible

```
SELECT overlaps(GeomFromText('POLYGON((0 0,4 4,6 2,6 3,6 1,0 0))') ,
GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))') );
```

<sup>13</sup> Grafica tomada de: <http://edndoc.esri.com/arcobjects/9.2/java/api/arcobjects/com/esri/arcgis/>

```
tutopostgis=# SELECT overlaps(GeomFromText('POLYGON((0 0,4 4,6 2,6 3,6 1,0 0))') ,
, GeomFromText('POLYGON((4 7,2 4,6 5,6 5,6 5,4 7))') );
overlaps
```

```
-----
f
<1 row>
```

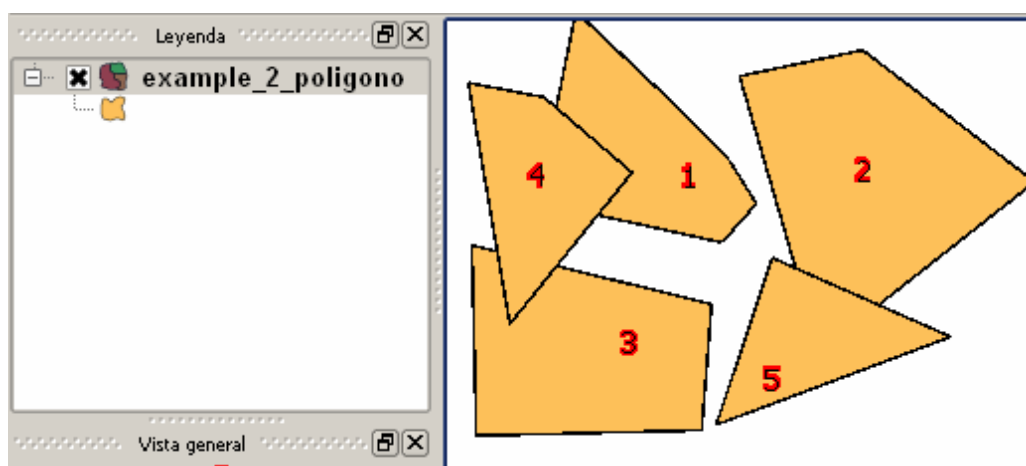
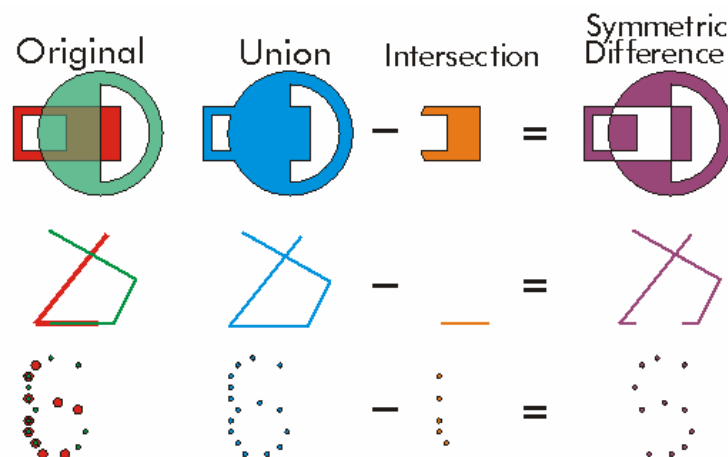
```
SELECT overlaps(GeomFromText('LINESTRING(3 38, 6 38, 5 40, 8 42, 8 42, 8 42)'),
GeomFromText('LINESTRING(5 40, 8 42, 5 42, 4 40)'));
```

```
tutopostgis=# SELECT overlaps(GeomFromText('LINESTRING(3 38, 6 38, 5 40, 8 42, 8
42, 8 42)') , GeomFromText('LINESTRING(5 40, 8 42, 5 42, 4 40)') );
overlaps
```

```
-----
t
<1 row>
```

**Nota:** Podrá encontrar las fuentes de estos ejemplos, en el archivo: **example\_4.sql**

### Unión, Intersección y Diferencia Simétrica<sup>14</sup>

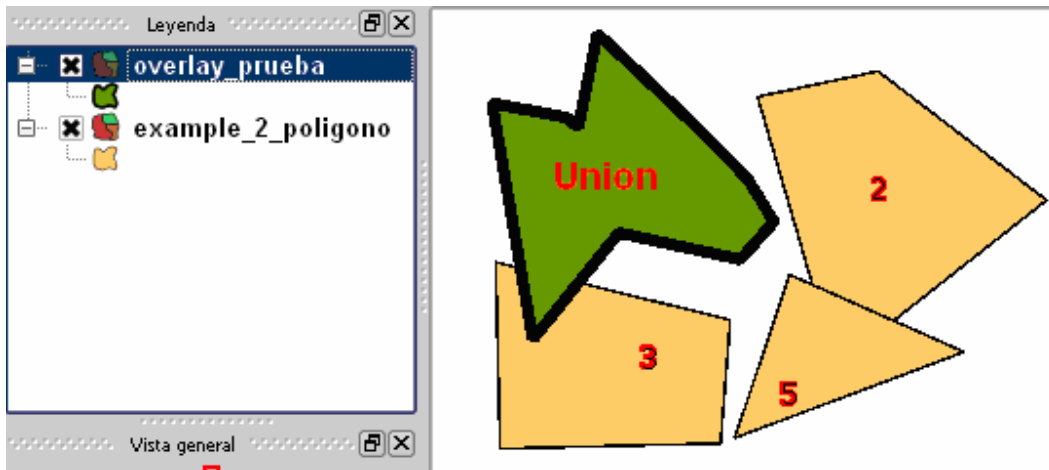


<sup>14</sup> Grafica tomada de: <http://edndoc.esri.com/arcobjects/9.2/java/api/arcobjects/com/esri/arcgis/>



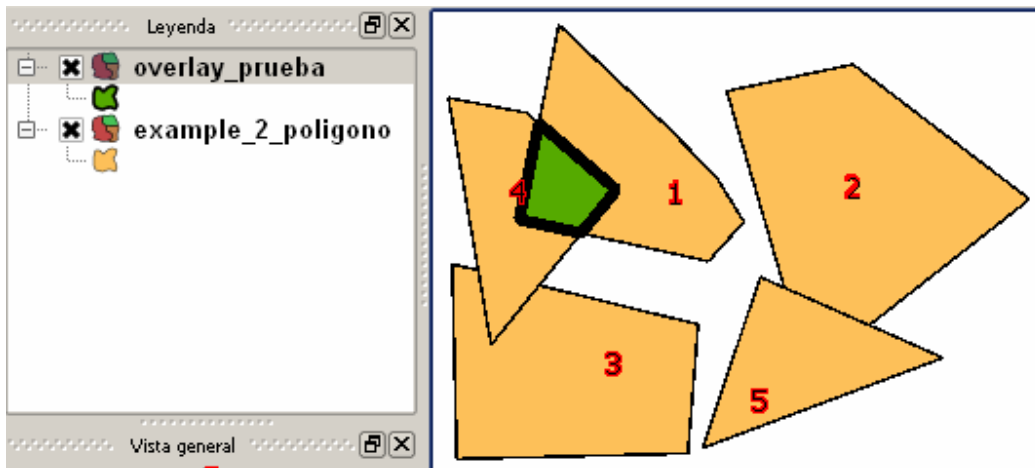
**Unión: ST\_Union** - Retorna una geometría que representa la unión de dos geometrías.

```
SELECT (St_Union( ( SELECT the_geom FROM example_2_poligono where id=1) ,  
(SELECT the_geom FROM example_2_poligono where id=4)));
```



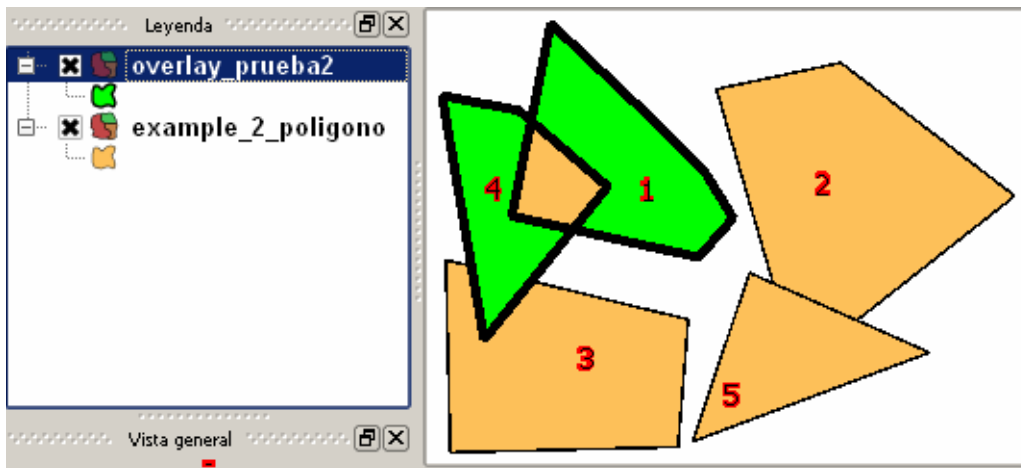
**Intersección: ST\_Intersection** - Retorna una geometría que representa la intersección unión de dos geometrías.

```
SELECT (ST_Intersection( ( SELECT the_geom FROM example_2_poligono where id=1) ,  
(SELECT the_geom FROM example_2_poligono where id=4)));
```



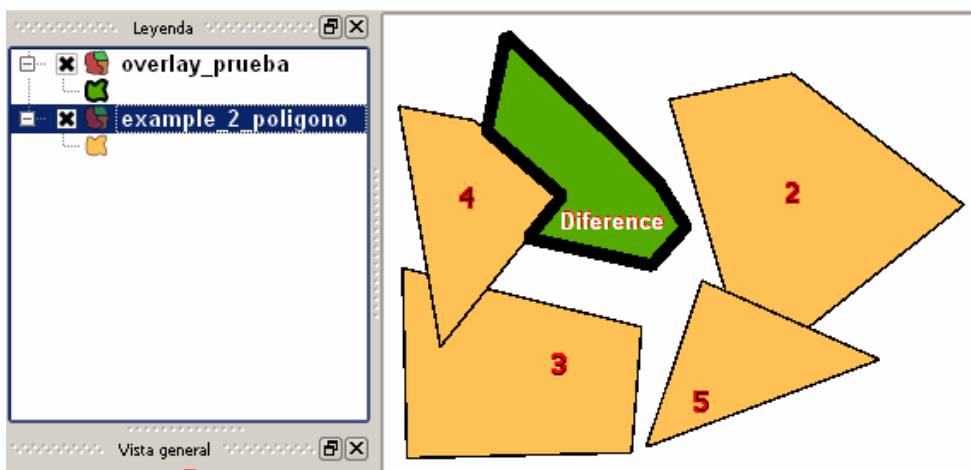
**Diferencia Simétrica: ST\_SymDifference** - Retorna una geometría que representa la diferencia simétrica de dos geometrías.

```
SELECT (ST_SymDifference( ( SELECT the_geom FROM example_2_poligono where id=1) ,  
(SELECT the_geom FROM example_2_poligono where id=4)));
```

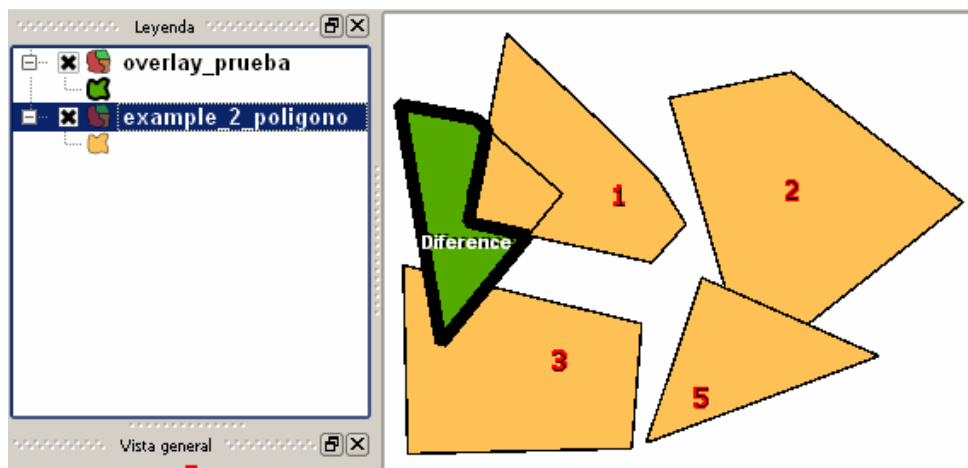


**Diferencia: ST\_Difference** - Retorna una geometría que representa la diferencia de las dos geometrías.

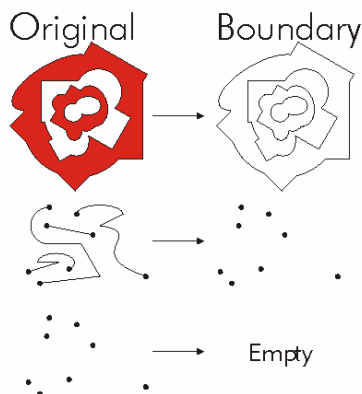
```
SELECT (ST_Difference( ( SELECT the_geom FROM example_2_poligono where id=1) ,
(SELECT the_geom FROM example_2_poligono where id=4)));
```



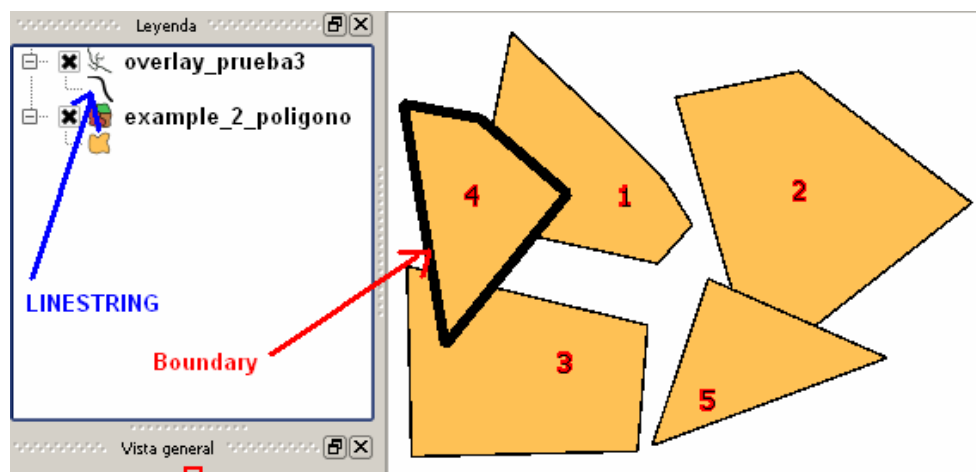
```
SELECT (ST_Difference( ( SELECT the_geom FROM example_2_poligono where id=4) ,
(SELECT the_geom FROM example_2_poligono where id=1)));
```



**Frontera<sup>15</sup>: ST\_Boundary** - Retorna una geometría que representa la frontera geométrica.



```
SELECT (ST_Boundary(( SELECT the_geom FROM example_2_poligono where id=4 )));
```



**Views (Vistas):** Una vista es una consulta, que refleja el contenido de una o más tablas, desde la que se puede acceder a los datos como si fuera una tabla. **¡Creemos una vista!**

```
CREATE VIEW vista_example AS
SELECT the_geom, area(the_geom), perimeter(the_geom), astext(the_geom), srid(the_geom)
FROM example_2_poligono;
```

```
tutopostgis=# CREATE VIEW vista_example AS SELECT the_geom, area(the_geom), perimeter(the_geom), astext(the_geom), srid(the_geom) FROM example_2_poligono;
CREATE VIEW
```

Una vez creada la vista:

```
SELECT * from vista_example;
```

	the_geom geometry	area double precis	perimeter double precis	astext text	srid integer
1	01060000000010	0.55292042034	3.18499570718	MULTIPOLYGON	-1
2	01060000000010	0.96664290759	4.10657316302	MULTIPOLYGON	-1
3	01060000000010	0.76342541870	3.59185896168	MULTIPOLYGON	-1
4	01060000000010	0.41634590655	2.87648123418	MULTIPOLYGON	-1
5	01060000000010	0.35373941566	2.82794639085	MULTIPOLYGON	-1

<sup>15</sup> Grafica tomada de: <http://edndoc.esri.com/arcobjects/9.2/java/api/arcobjects/com/esri/arcgis/>

**Triggers ( Disparadores ) y PL/pgSQL :** Un trigger o un disparador es un evento que se ejecuta cuando se cumple determinada condición establecida al realizar una operación de inserción (INSERT), actualización (UPDATE) o borrado (DELETE). . **¡Creemos un trigger!**

**Creemos Tabla:**

```
CREATE TABLE tablatrigger
(
  gid serial NOT NULL,
  x1 int4, y1 int4, x2 int4, y2 int4,
  geom1 geometry, geom2 geometry,
  buff_geom1 geometry, buff_geom2 geometry,
  linegeom1_geom2 geometry,
  buffdistance int4,
  distance_calc double precision,
  distance_postgis double precision,
  CONSTRAINT tablatrigger_pkey PRIMARY KEY (gid)
);
```

**Creemos Función en pl/pgSQL:** partiendo de un par de coordenadas (**x1,y1,x2,y2**), generara una geometría de punto para cada una de ellas (**geom1, geom2**), un parámetro adicional(**buffdistance**), servirá para indicar el tamaño de la geometría correspondiente al buffer a crear(**buff\_geom1,buffgeom2**), igualmente crear una línea entre los dos puntos generados(**linegeom1\_geom2**),finalmente calcula la distancia euclidiana entre los dos puntos, operando usando funciones matematicas simples (**distance\_calc**), y (**distance\_postgis**) calcula usando función postgis: length(geometry,geometry).

```
CREATE OR REPLACE FUNCTION funcionDisparadora()
RETURNS "trigger" AS
$BODY$
BEGIN

  NEW.geom1:=SetSRID(MakePoint(new.x1, new.y1),-1) ;
  NEW.geom2:=SetSRID(MakePoint(new.x2, new.y2),-1) ;
  NEW.buff_geom1:=buffer(SetSRID(MakePoint(new.x2, new.y2),-1),new.buffdistance);
  NEW.buff_geom2:=buffer(SetSRID(MakePoint(new.x2, new.y2),-1),new.buffdistance);
  NEW.linegeom1_geom2:= ST_MakeLine(SetSRID(MakePoint(new.x1, new.y1),-1) ,
SetSRID(MakePoint(new.x2, new.y2),-1) );
  NEW.distance_calc:= sqrt( pow((new.x2 - new.x1),2) + pow((new.y2 - new.y1),2));
  NEW.distance_postgis:= length( ST_MakeLine(SetSRID(MakePoint(new.x1, new.y1),-1) ,
SetSRID(MakePoint(new.x2, new.y2),-1) ) );

  RETURN NEW;
END
$BODY$
LANGUAGE 'plpgsql' VOLATILE;
```

**Creamos Trigger:**

```
CREATE TRIGGER triggerEjemplo
BEFORE INSERT OR UPDATE
ON tablatrigger
FOR EACH ROW
EXECUTE PROCEDURE funcionDisparadora();
```

**Realizamos la inserción de un elemento:**

```
INSERT INTO tablatrigger(x1, y1, x2, y2, buffdistance) VALUES (1,2,4,5,3);
```

**Probamos si que nuestro trigger y función realizaron su tarea:**

```
SELECT * FROM tablatrigger ;
```

	gid integer	x1 integer	y1 integer	x2 integer	y2 integer	geom1 geometry	geom2 geometry	buff_geom1 geometry
1	1	1	2	4	5	0101000000	0101000000	010300000001

buff_geom2 geometry	linegeom1_geom2 geometry	buffdistance integer	distance_calc double precision	distance_postgis double precision
010300000000	010200000000200000C	3	4.24264068711928	4.24264068711928

```
SELECT
x1,y1,x2,y2,astext(geom1),astext(geom2),astext(buff_geom1),astext(buff_geom2),astext(linegeom1
_geom2),buffdistance,distance_calc,distance_postgis FROM tablatrigger;
```

	x1 integer	y1 integer	x2 integer	y2 integer	astext text	astext text
1	1	2	4	5	POINT(1 2)	POINT(4 5)

astext text	astext text	astext text	buffdistance integer	distance_calc double precis	distance_pos double precis
POLYGON((7 5,	POLYGON((7 5,	LINESTRING(1 3	3	4.24264068711	4.24264068711

**Nota:** Podrá encontrar las fuentes de estos ejemplos, en el archivo: [example\\_5.sql](#)

Lo siguiente a tratar en la guía corresponde a la actualización de la guía **MapServer for Dummies (MFD)**, se han conservado la mayoría de ejemplos, y a sugerencia de variedad de lectores, los ejemplos se han migrado a la plataforma Linux en esta entrega.

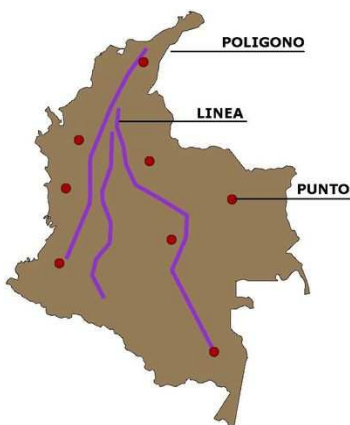
## PHP y PostgreSQL

Podemos realizar un pequeño script el cual nos permitirá checkear la conexión de postgres con php.

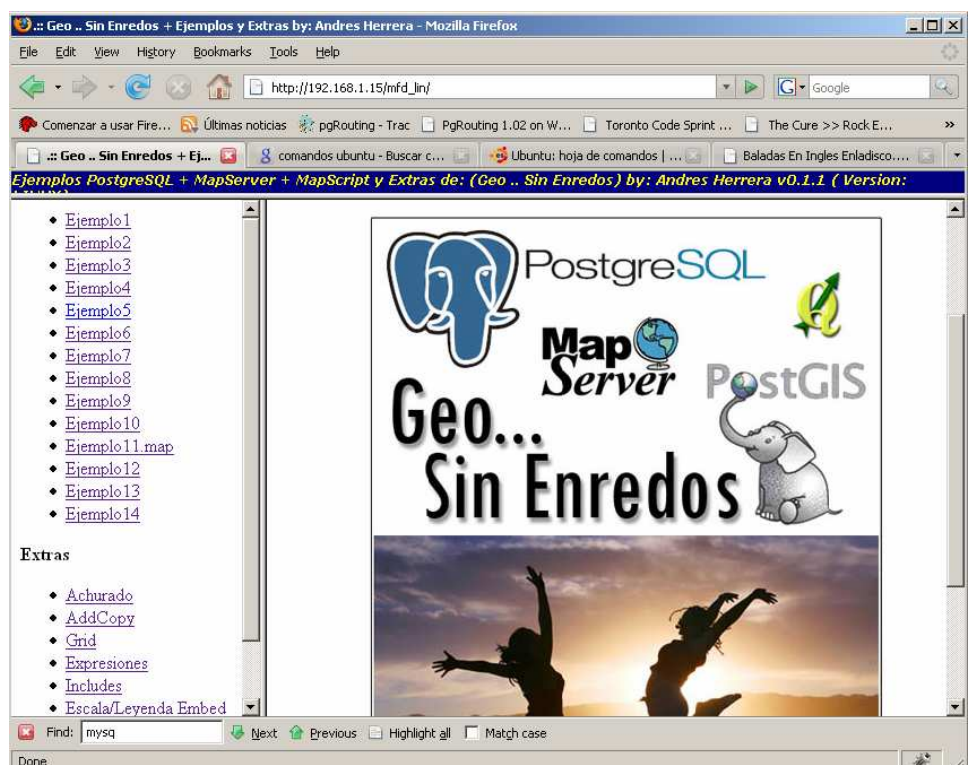
```
<?php
    $connection = "host=localhost port=5432 dbname=prueba user=postgres password=pass";
    $conecta=pg_connect($connection);

    if(!$conecta)
    {
        die("No se pude establecer la conexión con la base de datos.");
    }
else
{
    print("Conexión establecida con éxito !!");
}
?>
```

## Ejemplos de MapServer y PHP MapScript (MFD)



Junto a este documento, se adjuntan una serie de ejemplos, prácticos totalmente útiles y funcionales. Con relación a la información espacial contenida en el paquete; se incluyen 3 coberturas de tipo vectorial (punto, línea, polígono), las cuales están referenciadas al sistema geocéntrico WGS84, con las cuales se desarrollaran gran parte de los ejemplos a los que se hace referencia.



## ¿Como Acceder a los Ejemplos?

Es de esperar, que en este punto el ambiente operativo y aplicativo este correctamente funcional.



Descomprimir (**mfd\_win.zip**) en ( **C:\ms4w\Apache\htdocs\** ) y acceder a los ejemplos a digitando en el browser la siguiente URL: **http://localhost/mfd\_win/index.php**



En solo tres pasos, tendremos nuestros ejemplos funcionando en FGS.

```
$ sudo ln -sf /opt/fgs/www/htdocs /var/www
$ sudo ln -sf /opt/fgs/www/cgi-bin /usr/lib/cgi-bin
```

Descomprimir ( **mfd\_lin.zip** ) en ( **/var/www** )

```
chmod 777 /var/www/tmp
```

Acceder a los ejemplos a digitando en el browser la siguiente URL:  
**http://localhost/mfd\_lin/index.php**

## Mapas Estáticos

En el **Ejemplo1**<sup>16</sup> elaboraremos un mapa estático donde representaremos las 3 entidades básicas (punto, línea, polígono) que hemos definido anteriormente.

Echémosle un vistazo al archivo **ejemplo1.map**, este coincide con la estructura de mapfile propia de mapserver. Puntos importantes a destacar son:

Bloque: **EXTENT -88 -5 -62 13** corresponde a las coordenadas **[minx], [miny], [maxx], [maxy]** de nuestra información espacial, podemos visualizar estos limites, apoyándonos en herramientas como son Quantum GIS, GRASS, gvSIG, o cualquier tipo de software que nos permita visualizar información geográfica en formato ESRI shape.



Ventana de Propiedades de la capa, software Quantum GIS

<sup>16</sup> Ver archivo MFD.zip que acompaña esta guía.





En la consola FWTools; ubicándonos en el path donde se encuentra la información espacial, digitamos: **ogrinfo -al archivo.shp**

```

C:\ms4w\Apache\htdocs\MFD\shapes>ogrinfo -al lineas.shp
INFO: Open of 'lineas.shp'
      using driver 'ESRI Shapefile' successful.

Layer name: lineas
Geometry: Line String
Feature Count: 3
Extent: (-76.606100, -0.996208) - (-70.764584, 10.921424)
Layer SRS WKT:
GEOGCS["GCS_Assumed_Geographic_1",
  DATUM["North_American_Datum_1927",

```



**ogrinfo -al archivo.shp**

```

server@server-desktop:/var/www/mfd_lin/shapes$ ogrinfo -al lineas.shp
INFO: Open of 'lineas.shp'
      using driver 'ESRI Shapefile' successful.

Layer name: lineas
Geometry: Line String
Feature Count: 3
Extent: (-76.606100, -0.996208) - (-70.764584, 10.921424)

```

```

SHAPEPATH "shapes/"
FONTSET "misc/fonts/fonts.txt"
SYMBOLSET "misc/symbols/symbols.sym"

```

Aquí definimos la ubicación de nuestra información especial (shapes), el archivo de fuentes y la librería de símbolos.



```

WEB
  IMAGEPATH "C:/ms4w/Apache/htdocs/mfd_win/tmp/"
  IMAGEURL "tmp/"
END

```

**/var/www/mfd\_lin/tmp/**

Aquí definimos el path temporal donde mapserver renderizara las imágenes.

```

UNITS dd

```

Aquí definimos la unidad de medida con la que nuestra información espacial esta representada.

Este primer ejemplo contiene dos archivos importantes Ejemplo1.map y Ejemplo1.php el primero correspondiente al MAPFILE donde se definen las propiedades del mapa a generar y cada capa (LAYER) de información espacial a ser representado, el segundo;

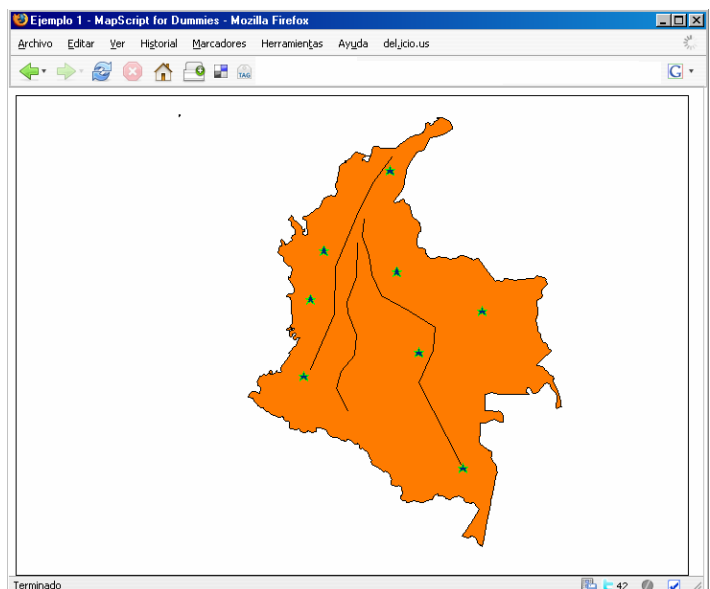
LAYER NAME "Poligonos" STATUS ON <b>DATA "poligono.shp"</b> TYPE POLYGON CLASS STYLE COLOR 255 123 0 OUTLINECOLOR 0 0 0 END END END	<pre>&lt;?php if (!extension_loaded("MapScript")) {     dl('php_mapscript.'.PHP_SHLIB_SUFFIX); }  \$mapObject = ms_newMapObj("ejemplo1.map"); \$mapImage = \$mapObject-&gt;draw(); \$urlImage = \$mapImage-&gt;saveWebImage();  ?&gt; &lt;img src="&lt;?php echo \$urlImage; ?&gt;" border="1" &gt;</pre>
--	---

**ejemplo1.php**, corresponde a un script en php que usa la Librería mapscript para crear el objeto del mapa e invocar el mapserver para que genere la imagen, una vez renderizada la imagen es llamada desde un bloque de código en HTML y desplegada en cualquier navegador, el resultado al digitar la url:

[http://localhost/mfd\\_win/ejemplo1.php](http://localhost/mfd_win/ejemplo1.php)



[http://localhost/mfd\\_lin/ejemplo1.php](http://localhost/mfd_lin/ejemplo1.php)



## No más shapefiles

En este ejemplo no usaremos más los archivos shapes, y crearemos un repositorio remoto (base de datos), el cual contendrá la información vectorial, como primer paso convertiremos nuestros archivos fuente ESRI shp a SQL, siguiendo las siguientes recomendaciones.

*Algunos aspectos a tener en cuenta cuando trabajemos con archivos:*

**No usar espacios nombres de archivo, No usar caracteres especiales, Usar nombres cortos y de fácil recordación.**



Recordemos que ubicándonos en el path donde se encuentra la información espacial, y siguiendo la siguiente sintaxis: **shp2pgsql [shp] [shx] > salida.sql**, sin indicar la extensión de los archivos de entrada, por ejemplo si nuestra misión es convertir el shape de **linea.shp**, digitaríamos en la línea de comandos: **shp2pgsql linea linea > linea.sql**  
 Como resultado obtenemos una salida en formato .SQL

## ¡Trabajo Tedioso!

Cuando tenemos una gran cantidad de coberturas en formato ESRI Shape, las cuales deseamos insertar en nuestra base de datos, resulta ser una tarea bastante tediosa convertir una a una, escribiendo una sencilla rutina en batch podemos ahorrarnos unos cuantos minutos de nuestras vidas y convertir grandes volúmenes de información usando del comando shp2pgsql para luego ser insertadas en nuestra base de datos geográfica.

```
@echo *****
@echo all_shapes to pgsq1 v0.1
@echo .
@echo fandresherrera(at)hotmail(dot)com
@echo .
@echo *****
@echo off
color 20
for %%x in (*.shp) do shp2pgsql %%~nx %%~nx >
%%~nx.sql
pause
exit
```



Encontrara  
está y mas  
herramientas en  
**misc/tools**



Un script mas completo y complejo:

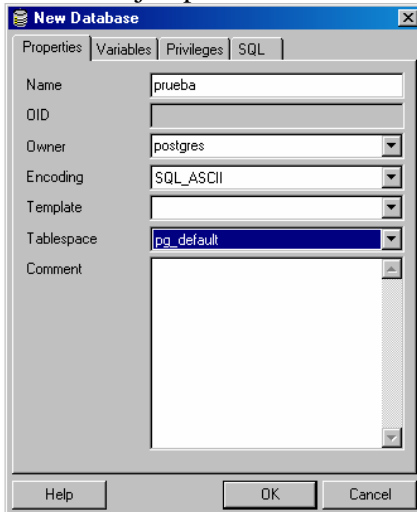
```
#!/bin/bash
# Copiar los shapefiles de un directorio a Postgis empleando shp2pgsql 20080811
#
# Germán Carrillo ( carrillo.german@gmail.com )
# GeoTux (http://geotux.tuxfamily.org)
##
echo;echo "Iniciando el cargue...";echo
rm -f log.txt
bd="nombrebd";U="usuario";esquema="public" #Parámetros
Shapefiles=$(find -name "*.shp");numShapefiles=${#Shapefiles[*]}
if [ $numShapefiles = 0 ]; then
    echo "No hay shapefiles en el directorio "`pwd`"
else
    for file in $( find -name "*.shp" )
    do
        filename=`basename $file`;fullname=`pwd`/$filename
        echo;echo "Nombre: $filename"
        shp2pgsql -s 4326 -d -g the_geom -D -i -I -S -N skip $fullname $esquema.${filename%.shp} |
psql - d $bd -U $U >> log.txt 2>&1
    done
    echo;echo "Terminando el cargue..."
    echo "Más información del proceso en el archivo log.txt";echo
fi
exit 0
#Fin del script
```

## Creando el repositorio

Recordemos que estos procedimientos ya los habíamos realizado al inicio del documento, así que ya sabemos diversas formas de realizarlo.



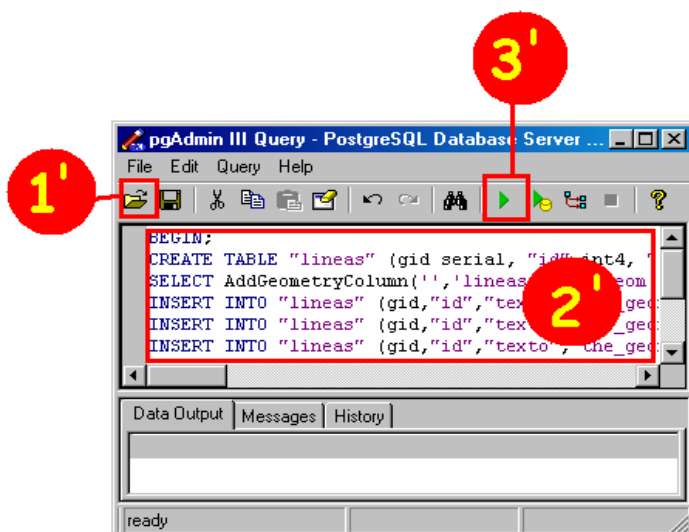
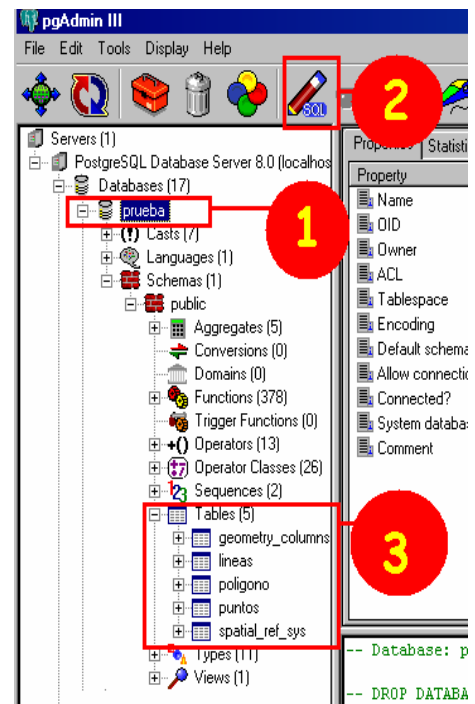
Para este ejemplo crearemos una nueva base de datos en PostgreSQL, la cual llamaremos (**prueba**).



Los pasos para crear esta;

- 1) Iniciamos pgAdmin III
- 2) Nos conectamos al servidor de PostgreSQL( indicamos usuario y contraseña).
- 3) Vamos a **Edit -> New Database**
- 4) En la ventana de **New Database**, configuramos la base de datos escogiendo encoding **SQL\_ASCII**.

Una vez creada la base de datos procedemos a volcar el contenido de cada una de nuestras geometrías convertidas a SQL, (1) primero es conveniente seleccionar la base de datos creada, seguidamente picando el icono de (2) (Execute arbitrary SQL query's). La ventana emergente nos permitirá seleccionar el archivo que deseamos volcar a través del icono (1') (open file), aquí seleccionamos el archivo .SQL el cual deseamos introducir en la base de datos, en (2') pre-visualizamos el archivo correspondiente a la geometría a introducir... para la el **ejemplo2** introduciremos los archivos correspondientes a **poligonos.sql** , **lineas.sql** , **puntos.sql** . Con (3') ejecutamos la sentencia SQL. Verificamos la correcta creación de las tablas en la base de datos en (3).



## ¡Manos a la obra !

Nuestro **ejemplo2**, corresponde a la generación de un mapa estático en mapserver y mapscript, similar al anterior, salvo una serie de ligeros cambios en la estructura MAP contenidos en el **ejemplo2.map**, cambios que nos permitirán conectarnos a la base de datos previamente creada. Echémosle un vistazo al archivo ejemplo2.map, este coincide con la estructura de mapfile propia de mapserver. Puntos importantes en este archivo a destacar son:

Bloque:

**LAYER**

**CONNECTIONTYPE** postgis

**NAME** "Poligonos"

Aquí definimos el tipo de conexión; postgis para nuestro ejemplo

**CONNECTION** "user=postgres password=1234567 dbname=prueba host=localhost"

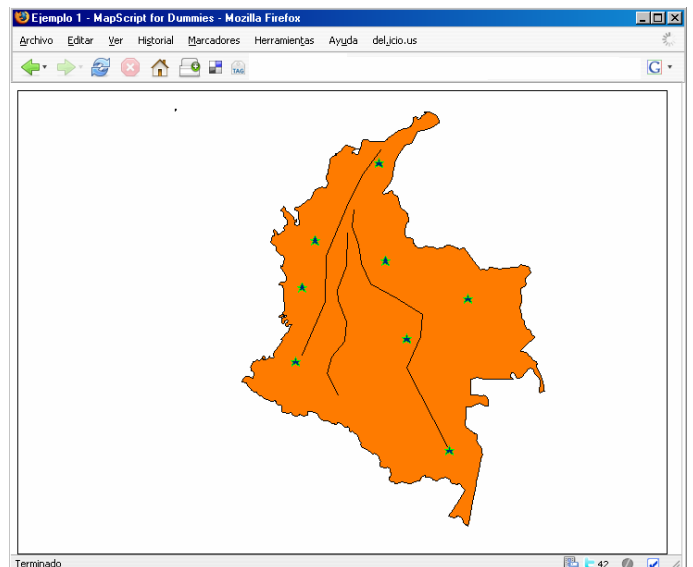
**DATA** "the\_geom FROM poligono as poligono using unique gid using SRID=-1"

Para realizar la conexión con la base de datos, indicamos los parámetros correspondientes, y realizamos un simple query (SELECT) a la tabla deseada, trayendo el campo de **the\_geom** el cual corresponde a la geometría en formato **WKT**.

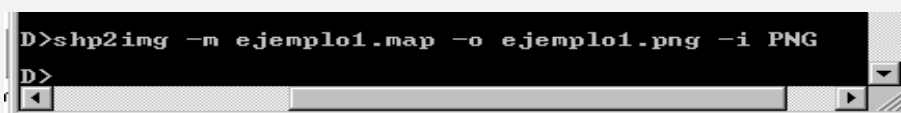


Para finalizar, el resultado lo podemos ver digitando en nuestro navegador la url: [http://localhost/mfd\\_win/ejemplo2.php](http://localhost/mfd_win/ejemplo2.php) como vemos el resultado corresponde al despliegue de un mapa con las entidades básicas (punto, línea, polígono), las cuales se encuentran en la base de datos postgresSQL.

[http://localhost/mfd\\_lin/ejemplo2.php](http://localhost/mfd_lin/ejemplo2.php)



Ubicándonos en el path en donde se encuentran nuestros archivos MAP, digitamos la siguiente línea: **shp2img -m ejemplo1.map -o ejemplo1.png -i PNG**, Esta es otra forma de generar un mapa estático partiendo de la estructura .map.



nplo9.php



ejemplo10.map



ejemplo10.php



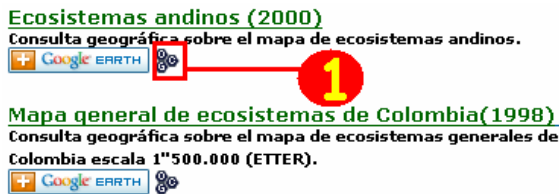
ejemplo1.png



ejemplo1.map

## Accediendo a Repositorios distribuidos en WMS

En el ejemplo3 vamos a conectarnos a un repositorio OGC WMS, para este ejemplo usaremos el recurso proporcionado por el instituto humboldt<sup>17</sup>, el cual nos proporciona 3 tipos de mapas (1) diferentes servidos a través de WMS a los cuales tenemos libre acceso.

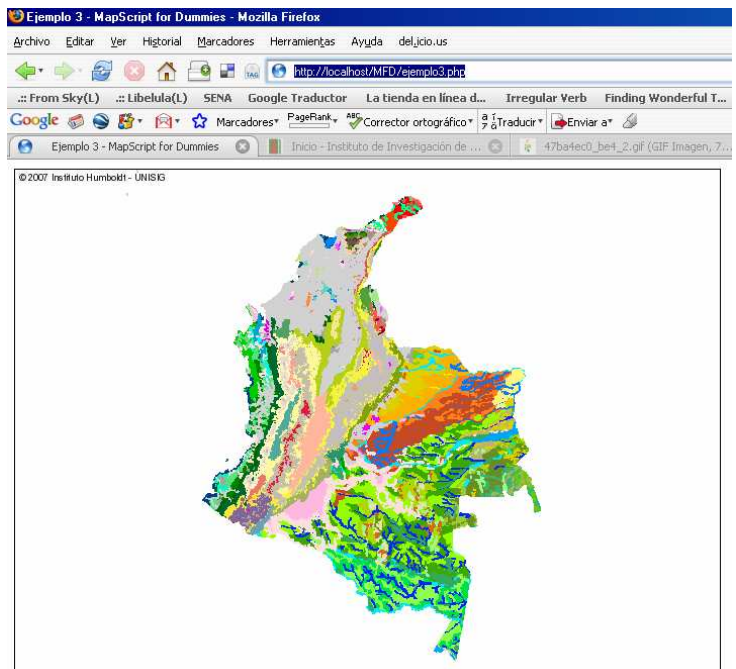


En **ejemplo3.map** observemos los cambios, para poder realizar la respectiva conexión WMS. Puntos importantes en este archivo a destacar son:

Bloque:

```
PROJECTION
    "init=epsg:4326"
END
```

Forzó la proyección. Según humboldt; el mapa esta referenciado en WGS84, para esto consultamos el código EPSG<sup>18</sup> equivalente a WGS, este corresponde al **4326**.



```
LAYER
    NAME "Ecosistemas"
    TYPE RASTER
    STATUS ON
    CONNECTION
    "http://www.humboldt.org.co/unisig/ogc/wxs?service=wms&servicename=Ecosistemas_de_Colombia&request=getcapabilities"
    CONNECTIONTYPE WMS
    METADATA
        "wms_srs" "EPSG:4326"
        "wms_name" "Ecosistemas"
        "wms_server_version" "1.1.1"
        "wms_format" "image/png"
    END
END
```

Como podemos ver clara mente la estructura del mapfile se conserva, la definición de layer cambia en su contenido, puesto que WMS nos entrega una capa renderizada en formato raster, debemos definir esta en **TYPE RASTER**, igualmente los parámetros de conexión y el tipo de conexión cambia drásticamente con respecto a ejemplos anteriores. Un nuevo bloque de atributos **METADATA** se define, aquí definiremos parámetros propios de la conexión, **wms\_rsr**: tipo e

<sup>17</sup> <http://www.humboldt.org.co/humboldt/>

<sup>18</sup> <http://www.epsg.org/CurrentDB.html>

proyección, **wms\_name**: nombre de la capa que deseamos obtener, **wms\_server\_version**: la versión del servidor remoto WMS del cual estamos obteniendo la capa, **wms\_format** : formato en el que deseamos que el servidor nos entregue la imagen renderizada, para finalizar, el resultado lo podemos ver digitando en nuestro navegador la url: [http://localhost/mfd\\_win/ejemplo3.php](http://localhost/mfd_win/ejemplo3.php)



## Añadiendo Controles

[http://localhost/mfd\\_lin/ejemplo3.php](http://localhost/mfd_lin/ejemplo3.php)

En el ejemplo4 añadiremos una serie de controles básicos (**Zoom IN**, **Zoom OUT**, **pan**), con los cuales nuestro mapa ya nunca más será un aburrido mapa estático.

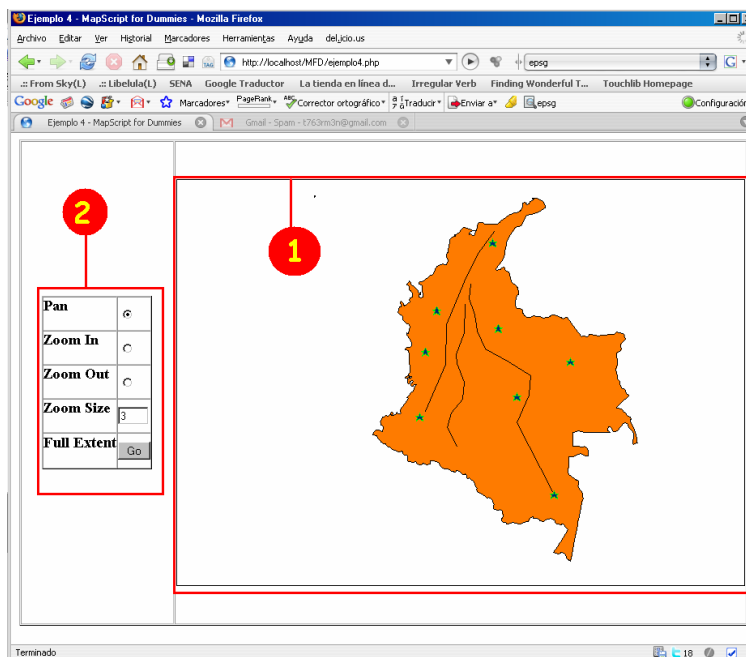
**Nota:** para la comprensión de este archivo es necesario conocimientos básicos de php y html (maneja y envío de formularios).

Para nuestro ejemplo, estructura del archivo .map se conserva (**ejemplo4.map**) igual a los anteriores, los cambios se han realizado sobre el archivo **ejemplo4.php**, donde mediante phpMapScript e interacciones de formularios en HTML, podemos lograr el efecto de zoom.

El resultado lo podemos ver digitando en nuestro navegador la url:

[http://localhost/mfd\\_win/ejemplo4.php](http://localhost/mfd_win/ejemplo4.php)

Donde (1) corresponde al mapa renderizado, (2) a las herramientas compuestas por elementos de formulario HTML (radiobutton), analizando el archivo **ejemplo4.php** Puntos importantes a destacar son:



[http://localhost/mfd\\_lin/ejemplo4.php](http://localhost/mfd_lin/ejemplo4.php)

```
$pointObject = ms_newpointObj();  
$pointObject->setXY($HTTP_POST_VARS["mapa_x"],$HTTP_POST_VARS["mapa_y"]);
```

Creación de objeto punto, con información de posición (x,y) capturada a partir de la interacción Mouse -> ventana de despliegue (1) .

```
$mapObject->zoompoint($zoomFactor,$pointObject,$mapObject->width,$mapObject->height,$extentRectObject);
```

Paso de parámetros a través de la función **zoompoint** hacia objeto correspondiente al mapa (2).





Usted puede aplicar transparencias a las capas definiendo dentro de la estructura de layer el atributo **TRANSPARENCY** e indicando un valor entre **(0-100)** donde 100 es opaco y 0 totalmente transparente.

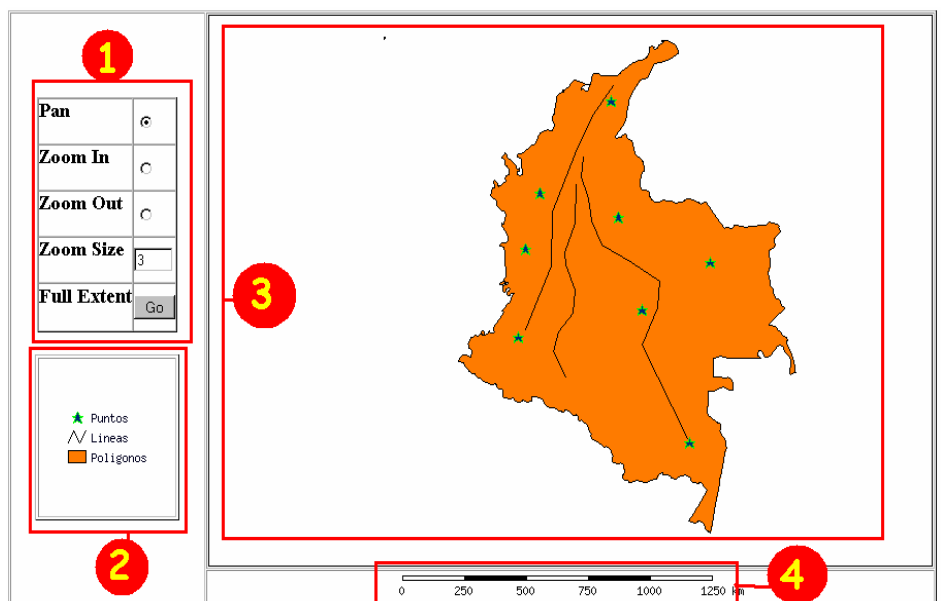
```
LAYER
  NAME "Poligonos "
  STATUS ON
  DATA "poligonos.shp"
  TRANSPARENCY 90
  TYPE POLYGON
  CLASS
    STYLE
      COLOR 255 250 250
      OUTLINECOLOR 100 150 155
    END
  END
END
```

## Más Dinamismo

En el ejemplo5 añadiremos más controles como: (escala grafica y leyenda), el resultado lo podemos ver digitando en nuestro navegador la url: [http://localhost/mfd\\_win/ejemplo5.php](http://localhost/mfd_win/ejemplo5.php)

Donde (1) corresponde a la barra de herramientas, exactamente igual a la del ejemplo anterior, (2) leyenda, (3) mapa renderizado, (4) escala grafica. Puntos importantes a destacar sobre los archivos **ejemplo5.map** y **ejemplo5.php** son:

```
LEGEND
  IMAGECOLOR 255 255 255
  KEYSIZE 18 12
  KEYSPPACING 5 5
  LABEL
    SIZE SMALL
    TYPE BITMAP
    BUFFER 0
    COLOR 0 0 30
    FORCE FALSE
    MINDISTANCE -1
    MINFEATURESIZE -1
    OFFSET 0 0
    PARTIALS TRUE
  END
  POSITION LL
  STATUS ON
END
```



Definición de las propiedades de la leyenda a visualizar este bloque de código se encuentra definido en el .map.

```
$mapLegend = $mapObject->drawLegend();
$urlLegend = $mapLegend->saveWebImage(MS_GIF, 0, 0, -1);
```

Creación y renderizado del objeto leyenda a través de phpmapscript.

```

```

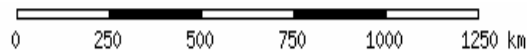
Despliegue de leyenda a través de HTML.

```
LAYER
  NAME "Poligonos"
  STATUS ON
  DATA "poligono.shp"
  TYPE POLYGON
  CLASS
    NAME "Poligonos"
    STYLE
      COLOR 255 123 0
      OUTLINECOLOR 0 0 0
    END
  END
END
```

★ Puntos  
 \ Lineas  
 ■ Poligonos

Debemos prestar un especial cuidado, dentro de cada layer al cual le deseemos generar leyenda, en la definición de la clase debemos agregar el atributo NAME, este es aquel que se mostrara una vez mapserver renderice la leyenda.

```
SCALEBAR
  IMAGECOLOR 255 255 255
  LABEL
    COLOR 0 0 0
    SIZE SMALL
  END
  SIZE 350 4
  COLOR 255 255 255
  BACKGROUND 0 0 0
  OUTLINECOLOR 0 0 0
  UNITS KILOMETERS
  INTERVALS 5
  STATUS ON
END
```



```
$mapScale = $mapObject->drawScaleBar();
$urlScale = $mapScale->saveWebImage(MS_GIF, 0, 0, -1);
```

Creación y renderizado del objeto de escala a través de phpmapscript.

```

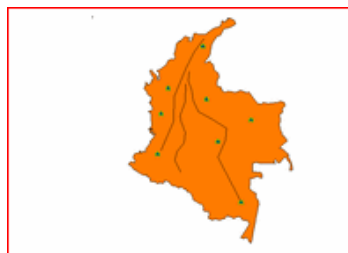
```

Despliegue de la escala grafica a través HTML.

## Añadiendo más objetos interesantes

En el **ejemplo6** añadiremos más controles como: (mapa de referencia y rosa de los vientos). Puntos importantes a destacar sobre los archivos **ejemplo6.map** y **ejemplo6.php** son:

```
REFERENCE
  IMAGE mapaclave.gif
  EXTENT -88 -5 -62 13
  STATUS ON
  COLOR -1 -1 -1
  OUTLINECOLOR 255 0 0
  SIZE 200 143
END
```



En nuestro mapa definimos un nuevo bloque de código el cual indica que usaremos una imagen **IMAGE** llamada mapaclave.gif como mapa de referencia, para nuestro ejemplo esta imagen fue una pre-visualización de nuestro mapa, grabada como un archivo adicional de nombre **mapaclave.gif** cuyas dimensiones son: 200 x 143px , debemos tener especial cuidado en proporcionar el correcto extent.

```
$keyMapImage = $mapObject->drawreferencemap();
$urlKeyMap = $keyMapImage->saveWebImage(MS_GIF, 0, 0, -1);
```

Creación y renderizado del objeto de referencia.

```

```

Despliegue del mapa de referencia a través HTML.

Bloque: en **misc/symbols/symbols.sym**

```
SYMBOL
NAME 'rosavientos'
TYPE PIXMAP
IMAGE 'norte.gif'
END
```



Agregaremos unas nuevas líneas al archivo symbols.sym, donde creamos un símbolo a través del renderizado de una imagen **norte.gif**

```
LAYER
NAME "Norte"
TYPE POINT
STATUS ON
TRANSFORM OFF
POSTLABELCACHE TRUE
FEATURE
POINTS 35 35
END
CLASS
SYMBOL 'rosavientos'
COLOR 0 0 0
OUTLINECOLOR 0 0 0
STYLE END
END
```

Definimos un nuevo layer de tipo punto (**POINT**) en el .map el cual usaremos para asignar un símbolo (**rosavientos**) en el punto de anclaje (**POINTS**). El resultado lo podemos ver digitando en nuestro navegador la url:

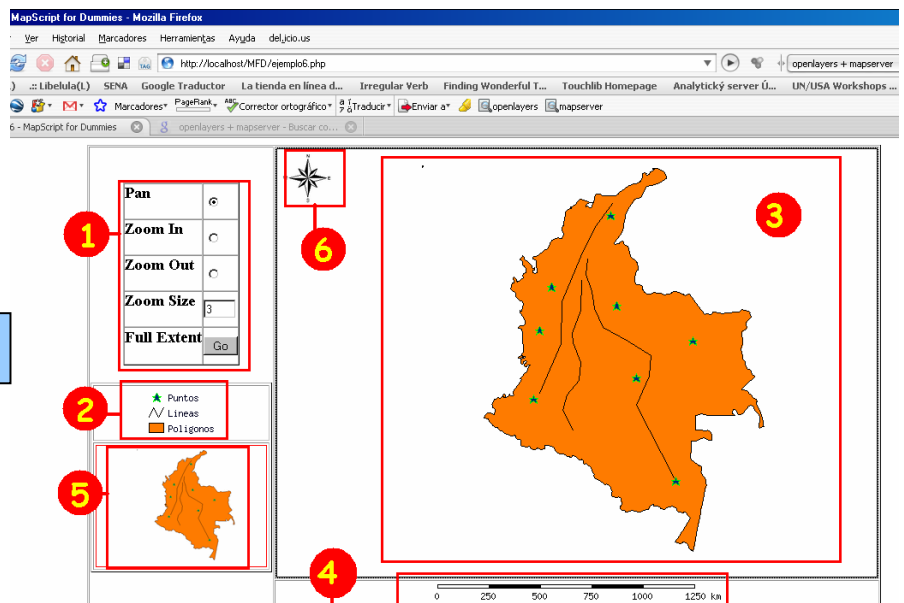


[http://localhost/mfd\\_win/ejemplo6.php](http://localhost/mfd_win/ejemplo6.php)

Donde (1) corresponde a la barra de herramientas, exactamente igual a la del ejemplo anterior, (2) leyenda, (3) mapa renderizado, (4) escala grafica, (5) mapa de referencia, (6) rosa de los vientos.



[http://localhost/mfd\\_lin/ejemplo6.php](http://localhost/mfd_lin/ejemplo6.php)



Usted puede añadir marcas de copyright de texto (2), o logo símbolos (1) a la imagen renderizada, mira los siguientes archivos. **addcopy.map** y **addcopy.php**, el resultado lo podemos ver digitando en nuestro navegador la url: [http://localhost/mfd\\_win/addcopy.php](http://localhost/mfd_win/addcopy.php)



## Manejo de Capas

En el **ejemplo7** añadiremos el control de capas (layer selector como en alguno de los software GIS). Puntos importantes a destacar sobre los archivos **ejemplo7.map** y **ejemplo7.php** son:

```
$item = $_REQUEST["layerselector"];
$allLayersObject=$mapObject->getAllLayerNames();
foreach ($allLayersObject as $idx => $layer)
{
    $layerObject=$mapObject->getLayerByName($layer);
    if( sizeof( $item ) > 0 )
    {
        if (in_array( $layerObject->name, $item ))
        {
            $layerObject->set( "status", MS_ON );
        }
        else
        {
            $layerObject->set( "status", MS_OFF );
        }
    }
}
```

- ☒ Poligonos
- ☒ Lineas
- ☒ Puntos

En este bloque le solicitamos al objeto de mapa que nos indique todas las capas por el cual esta compuesto **getAllLayerNames()** .Luego según la petición (encender / apagar ) proveniente de los check box en HTML, **getLayerByName(\$layer)** encendemos **\$layerObject->set( "status", MS\_ON )** o apagamos **\$layerObject->set( "status", MS\_OFF )** el layer correspondiente.

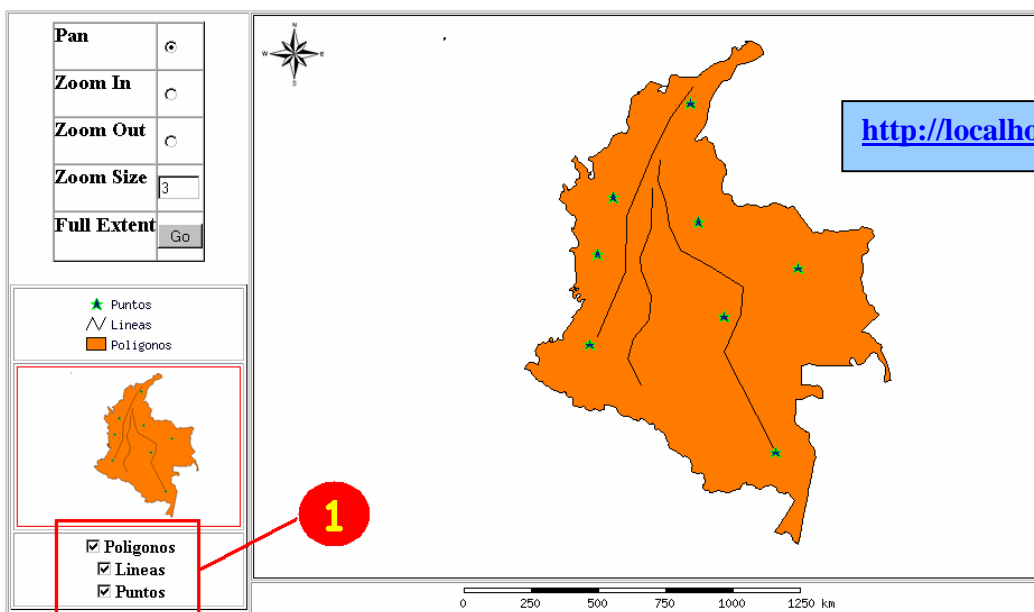
```

<?php
$allLayersObject=$mapObject->getAllLayerNames();
foreach ($allLayersObject as $idx => $layer)
{
    $layerObject=$mapObject->getLayerByName($layer);
    if ($layerObject->status==MS_ON)
    {
        if(($layerObject->name != "Norte"))
        {
            for($i=0;$layerObject->getClass($i);$i++) {
                $Class = $layerObject->getClass($i);
                echo "<input type='checkbox' value='{ $layerObject->name}'
                name='layersselector[]' onClick='document.frmlayerselector.submit();' checked>";
                echo "<span>{$Class->name}</span><br /> "; }
            }
        }
    else
    {
        if(($layerObject->name != "Norte"))
        {
            for($i=0;$layerObject->getClass($i);$i++) {
                $Class = $layerObject->getClass($i);
                echo "<input type='checkbox' value='{ $layerObject->name}'
                name='layersselector[]' onClick='document.frmlayerselector.submit();' > ";
                echo "<span>{$Class->name}</span><br /> "; }
            }
        }
    }
}
?>

```

En este bloque generamos código HTML (1) el cual será representado por el navegador mediante las cajas de chequeo (checkboxes), siempre mantendrá actualizadas las cajas que se encuentren encendidas y apagadas, adicionalmente agregamos un condicional para que nuestro layer de nombre **Norte**, siempre mantenga encendido.

El resultado lo podemos ver digitando en nuestro navegador la url:  
[http://localhost/mfd\\_win/ejemplo7.php](http://localhost/mfd_win/ejemplo7.php)



## Añadido eventos

```
$polLayer = $mapObject->getLayerByName("Poligonos");
$polLayer->set("status",MS_ON);
$punLayer = $mapObject->getLayerByName("Puntos");
$punLayer->set("status",MS_ON);
$linLayer = $mapObject->getLayerByName("Lineas");
$linLayer->set("status",MS_ON);
```

A través de mapscript traigo una instancia de las capas.

```
$dfKeyMapXMin = $mapObject->extent->minx;
$dfKeyMapYMin = $mapObject->extent->miny;
$dfKeyMapXMax = $mapObject->extent->maxx;
$dfKeyMapYMax = $mapObject->extent->maxy;
$dfWidthPix = doubleval($mapImage->width);
$dfHeightPix = doubleval($mapImage->height);
$onClickGeoX = GMapPix2Geo($_REQUEST['mapa_x'], 0, $dfWidthPix, $dfKeyMapXMin, $dfKeyMapXMax, 0);
$onClickGeoY = GMapPix2Geo($_REQUEST['mapa_y'], 0, $dfHeightPix, $dfKeyMapYMin, $dfKeyMapYMax, 1);
$my_point = ms_newpointObj();
$my_point->setXY($onClickGeoX,$onClickGeoY);
```

Genero punto con coordenadas mapa, invoco función que convierte de coordenadas píxel a mapa.

```
<?php
function GMapPix2Geo($nPixPos, $dfPixMin, $dfPixMax, $dfGeoMin, $dfGeoMax, $nInversePix)
{
    $dfWidthGeo = $dfGeoMax - $dfGeoMin;
    $dfWidthPix = $dfPixMax - $dfPixMin;
    $dfPixToGeo = $dfWidthGeo / $dfWidthPix;
    if (!$nInversePix)
        $dfDeltaPix = $nPixPos - $dfPixMin;
    else
        $dfDeltaPix = $dfPixMax - $nPixPos;
    $dfDeltaGeo = $dfDeltaPix * $dfPixToGeo;
    $dfPosGeo = $dfGeoMin + $dfDeltaGeo;
    return ($dfPosGeo);
}
?>
```

Función para convertir de coordenadas píxel, a coordenadas mapa.

```
//Query a Puntos
if(@$punLayer->queryByPoint($my_point, MS_SINGLE, 200) == MS_SUCCESS)
{
    $results = $punLayer->{resultcache};
    $punLayer->open();
    $rslt = $punLayer->getResult(0);
    $shape = $punLayer->getShape($rslt->tileindex, $rslt->shapeindex);
    $resultadoConsulta = $shape->values["texto"];
    echo "<center><br><br>Resultado de la Consulta Sobre Capa Puntos: <b> ".
        $resultadoConsulta . "</b></center>";
} else{ echo "No pudo realizar la consulta, vuelva a intentar !!!"; }
```

Mediante esta función desplegamos el resultado de abrir el layer, consultar el atributo **\$shape->values["texto"]** alfanumérico correspondiente al hacer **queryByPoint** del objeto seleccionado.

```
//Query a Poligonos
if(@$polLayer->queryByPoint($my_point, MS_SINGLE, 200) == MS_SUCCESS)
{
    $results = $polLayer->{resultcache};
    $polLayer->open();
    $rslt = $polLayer->getResult(0);
    $shape = $polLayer->getShape($rslt->tileindex, $rslt->shapeindex);
    $resultadoConsluta = $shape->values["CNTRY_NAME"];
    echo "<center>Resultado de la Consulta Sobre Capa Poligonos:<b> ".$resultadoConsluta . "</b></center>";
}else{ echo "No pudo realizar la consulta, vuelva a intentar !!"; }
```

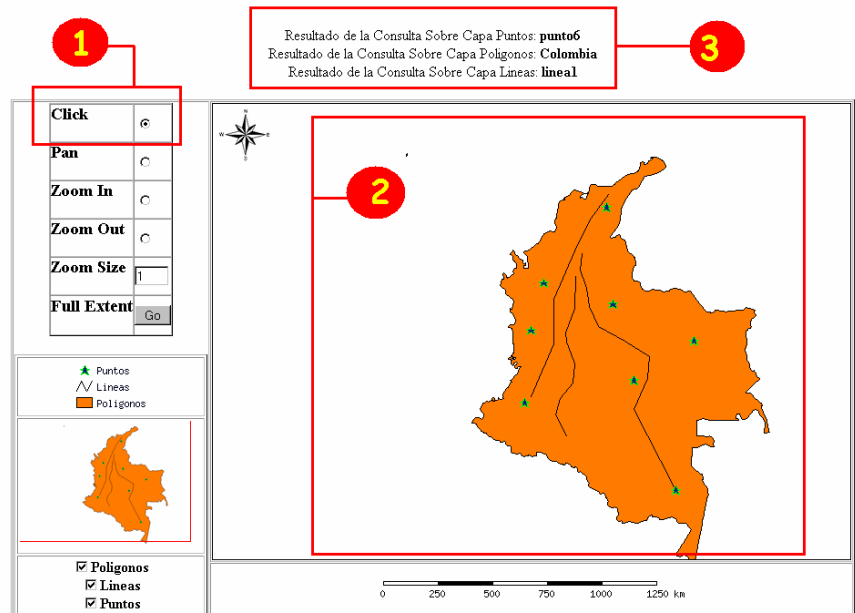
Mediante esta función desplegamos el resultado de abrir el layer, consultar el atributo **\$shape->values["CNTRY\_NAME"]** alfanumérico correspondiente al hacer **queryByPoint** del objeto seleccionado.

```
//Query a Lineas
if(@$linLayer->queryByPoint($my_point, MS_SINGLE, 200) == MS_SUCCESS)
{
    $results = $linLayer->{resultcache};
    $linLayer->open();
    $rslt = $linLayer->getResult(0);
    $shape = $linLayer->getShape($rslt->tileindex, $rslt->shapeindex);
    $resultadoConsluta = $shape->values["texto"];
    echo "<center>Resultado de la Consulta Sobre Capa Lineas:<b> ".
    $resultadoConsluta . "</b><br><br></center>";
}else{ echo "No pudo realizar la consulta, vuelva a intentar !!"; }
```

Mediante esta función desplegamos el resultado de abrir el layer, consultar el atributo **\$shape->values["texto"]** alfanumérico correspondiente al hacer **queryByPoint** del objeto seleccionado.

```
LAYER
NAME "Poligonos"
STATUS ON
DATA "poligono.shp"
TYPE POLYGON
CLASS
TEMPLATE 'poligonos.html'
NAME "Poligonos"
STYLE
    COLOR 255 123 0
    OUTLINECOLOR 0 0 0
END
END
TOLERANCE 20
END
```

Para cada layer, debemos agregar Atributo **TEMPLATE** y **TOLERANCE**.



El resultado lo podemos ver digitando en nuestro navegador la url: [http://localhost/mfd\\_win/ejemplo8.php](http://localhost/mfd_win/ejemplo8.php) , primero seleccionamos la herramienta click (1), luego picamos sobre algunas de las entidades graficas (2) y por ultimo visualizamos el resultado en (3).



[http://localhost/mfd\\_lin/ejemplo8.php](http://localhost/mfd_lin/ejemplo8.php)





```
LAYER
NAME "LayerMapInfo"
TYPE LINE
STATUS ON
CONNECTIONTYPE OGR
CONNECTION
"archivomapinfo.TAB"
CLASS
COLOR 255 0 0
NAME "LayerMapInfo"
END
END
```

Usted puede añadir capas no solo de ESRI shape, también puede añadir capas en formato **TAB** de **MapInfo**.

## No más archivos .MAP

En el **ejemplo9** dibujaremos nuestro mapa, sin necesidad de usar un archivo .map, puntos importantes a destacar este archivo **ejemplo9.php** son:

```
$mapObject = ms_newMapObj("");
$mapObject->set("name", "Pruebas");
$mapObject->set("shapepath", "C:/ms4w/Apache/htdocs/mfd_win/shapes/");
$mapObject->setSize(700,500);
$mapObject->setExtent(-88,-5,-62,13);
$mapObject->web->set("imagepath", "C:/ms4w/Apache/htdocs/mfd_win/tmp/");
$mapObject->web->set("imageurl", "tmp/");
```

Defino las propiedades basicas de creación de objeto del mapa.

```
$layerPoligono = ms_newLayerObj($mapObject);
$layerPoligono->set("name", "Poligonos");
$layerPoligono->set("type", MS_LAYER_POLYGON);
$layerPoligono->set("status", MS_ON);
$layerPoligono->set("data", "poligono.shp");
$clasePoligono = ms_newClassObj($layerPoligono);
$estiloPoligono = ms_newStyleObj($clasePoligono);
$estiloPoligono->color->setRGB(255,123,0);
$estiloPoligono->outlinecolor->setRGB(0, 0, 0);
```



**/var/www/mfd\_lin/shapes/**

Defino las propiedades de cada layer. Clases y estilos correspondientes a esta igual como en la estructura del MAPFILE (.map).

```
$symbolid = ms_newSymbolObj($mapObject, "star");
$soSymbol = $mapObject->getsymbolobjectbyid($symbolid);
$soSymbol->setpoints(Array(0, .375, .35, .375, .5, 0, .65, .375, 1, .375, .75, .625, .875, 1, .5, .75, .125, 1, .25, .625));
$soSymbol->set("filled", MS_TRUE);
$soSymbol->set("inmapfile", MS_TRUE);
```

Creamos símbolos y otras estructuras complejas.

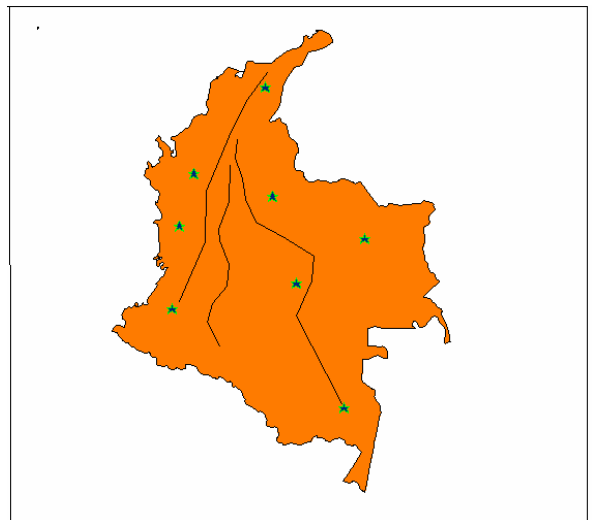
```
$estiloPuntos->color->setRGB(0, 34, 125);
$estiloPuntos->outlinecolor->setRGB(0, 255, 0);
$estiloPuntos->set("symbolname", "star");
$estiloPuntos->set("size", "10");
```

El resultado lo podemos ver digitando en nuestro navegador la url: [http://localhost/mfd\\_win/ejemplo9.php](http://localhost/mfd_win/ejemplo9.php) representa nuestro mapa de ejemplo el cual ha sido renderizado sin necesidad de definir un archivo externo .map.



[http://localhost/mfd\\_lin/ejemplo9.php](http://localhost/mfd_lin/ejemplo9.php)

De igual manera podemos acceder a repositorios remotos (bases de datos PostgreSQL) , el **ejemplo10**, presenta la misma salida grafica, ligeros cambios se realizan en el archivo **ejemplo10.php**, donde rescatamos el siguiente bloque de código.



```
$layerLineas->set("connectiontype","MS_POSTGIS");  
$layerLineas->set("connection","user=postgres password=1234567 dbname=prueba host=localhost");  
$layerLineas->set("data","the_geom FROM lineas as lineas using unique gid using SRID=-1");
```

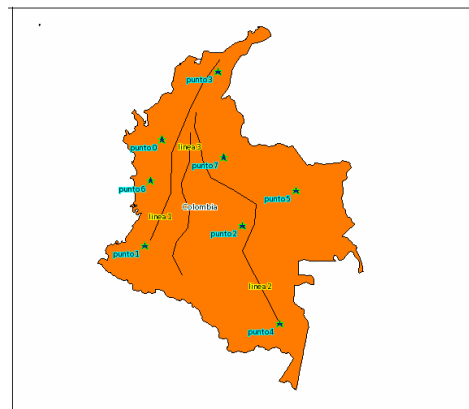
El resultado lo podemos ver digitando en nuestro navegador la url: [http://localhost/mfd\\_win/ejemplo10.php](http://localhost/mfd_win/ejemplo10.php)



[http://localhost/mfd\\_lin/ejemplo10.php](http://localhost/mfd_lin/ejemplo10.php)



```
LABELITEM "texto"  
LABELCACHE ON  
  
LABEL  
COLOR 0 0 0  
FONT sans  
TYPE TRUETYPE  
POSITION CC  
PARTIALS TRUE  
SIZE 7  
BUFFER 1  
OUTLINECOLOR 255 255 0  
END
```



Si desea añadir etiquetas (**labels**) a las entidades graficas, puedes ver un ejemplo digitando en el navegador la url: [http://localhost/mfd\\_win/labels.php](http://localhost/mfd_win/labels.php) el cual usa **labels.map** .



[http://localhost/mfd\\_lin/labels.php](http://localhost/mfd_lin/labels.php)

## Creando un Servidor WMS

Nuestro mapfile nos permite publicar mapas a través de wms, en el ejemplo11.map lograremos esto realizando unos ligeros cambios en la estructura, trozos a destacar los veremos a continuación.

```
WEB
  METADATA
    "wms_title"      "Ejemplo WMS de MapServer for Dummies"

#WINDOWS
  "wms_onlineresource" http://localhost/cgi-bin/mapserv.exe?map=../htdocs/mdf\_win/ejemplo11.map&

#LINUX
  wms_onlineresource" http://localhost/cgi-bin/mapserv?map=/var/www/mdf\_lin/ejemplo11.map&"
  "wms_srs"          "EPSG:4326"
  END
END
```

Ya no es necesario definir un path de salida, ahora tan solo basta definir un bloque de **METADATA**.

```
PROJECTION
  "init=epsg:4326"
END
```

Al igual que en el ejemplo de acceso a servicios WMS, para la publicación de nuestra información espacial es necesario definir una proyección de salida, en este caso usaremos WGS84 con su código EPSG equivalente.

```
LAYER
  NAME "Poligonos"
  METADATA
    "wms_srs"          "EPSG:4326"
    "wms_name"         "Poligonos"
    "wms_server_version" "1.1.1"
    "wms_format"       "image/png"
    "wms_transparent"  "true"
  END
  STATUS ON
  DATA "poligono.shp"
  TYPE POLYGON
  CLASS
    STYLE
      COLOR 255 123 0
      OUTLINECOLOR 0 0 0
    END
  END
END
```

Debemos definir para cada layer que deseamos hacer visible mediante WMS basta tan solo definir el bloque **METADATA** indicando los atributos de tipo de proyección ([wms\\_srs](#)), nombre accesible para esa capa ([wms\\_name](#)), versión del servidor wms ([wms\\_server\\_version](#)), formato de publicación ([wms\\_format](#)), si deseamos que nuestra capa sea publicada con soporte de transparencias ([wms\\_transparent](#)) indicamos trae, de lo contrario indicamos el atributo de false.

## Consumiendo servicios WMS

Accederemos a las capas servidas en WMS a través de un cliente pesado, en este caso se usara Quantum GIS, pero en general puedes usar cualquier tipo de software GIS el cual soporte acceso a datos de fuente WMS.

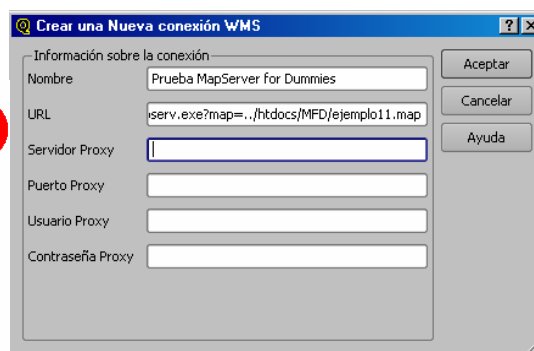
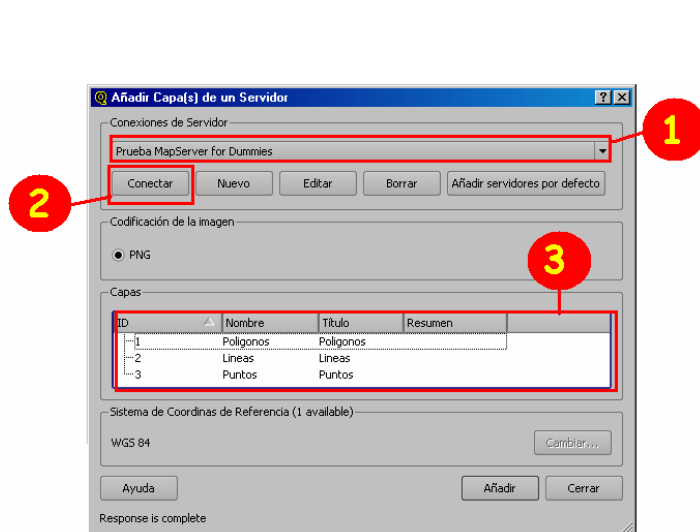
Para añadir una capa WMS en QuantumGIS, nos dirigimos al menú **Capas -> Añadir capa WMS** , en esta ventana emergente procedemos a crear una nueva conexión; indicando un nombre y la URL de acceso correspondiente a nuestro servicio previamente creado.



URL: [http://localhost/cgi-bin/mapserv.exe?map=../htdocs/mfd\\_win/ejemplo11.map](http://localhost/cgi-bin/mapserv.exe?map=../htdocs/mfd_win/ejemplo11.map)



URL: [http://localhost/cgi-bin/mapserv?map=/var/www/mfd\\_lin/ejemplo11.map](http://localhost/cgi-bin/mapserv?map=/var/www/mfd_lin/ejemplo11.map)



Una vez creada la conexión (1), procedemos a conectarnos con el servicio (2).

El software internamente se encarga de realizar

la petición **getCapabilities** al servidor.

Devolviendo las capas disponibles servidas (3), para nuestro ejemplo; capa: Polígonos, Líneas, Puntos.



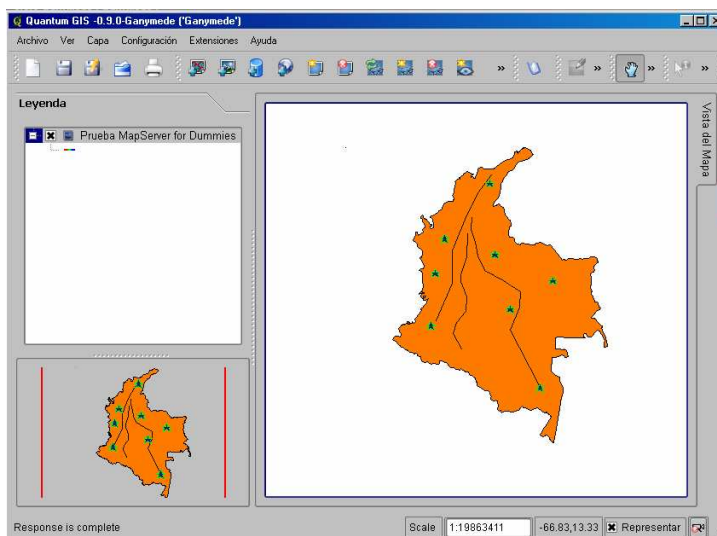
[http://localhost/cgi-bin/mapserv.exe?map=../htdocs/mfd\\_win/ejemplo11.map&service=WMS&request=getcapabilities](http://localhost/cgi-bin/mapserv.exe?map=../htdocs/mfd_win/ejemplo11.map&service=WMS&request=getcapabilities)



[http://localhost/cgi-bin/mapserv?map=/var/www/mfd\\_lin/ejemplo11.map&service=WMS&request=getcapabilities](http://localhost/cgi-bin/mapserv?map=/var/www/mfd_lin/ejemplo11.map&service=WMS&request=getcapabilities)

Añadimos las capas disponibles (3), el resultado es un mapa compuesto entregado por el mapserver. Igualmente podemos acceder a cada capa de forma independiente, y combinar estas con otro tipo de información espacial.

En **acceso\_wms.qgs** se guardo el proyecto de acceso al ejemplo presentado.



## OpenLayers

Al igual que podemos acceder a servicios wms, a través de clientes pesados, Openlayers nos permite acceder a estos mediante la interfaz web.

Openlayers es una librería OpenSource en JavaScript, la cual podemos obtener en la siguiente URL: <http://www.openlayers.org/> , OpenLayers hace muy fácil el despliegue de mapas dinámicos en la WEB, podemos acceder a nuestro WMS previamente creado digitando la URL:

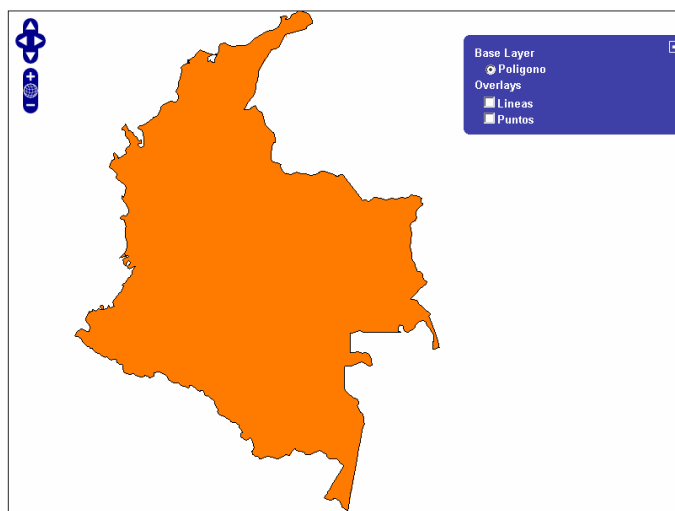


[http://localhost/mfd\\_win/ejemplo12.html](http://localhost/mfd_win/ejemplo12.html)



[http://localhost/mfd\\_lin/ejemplo12.html](http://localhost/mfd_lin/ejemplo12.html)

De este archivo podemos rescatar los siguientes bloques de código.



```
<script src="misc/lib/OpenLayers.js"></script>
```

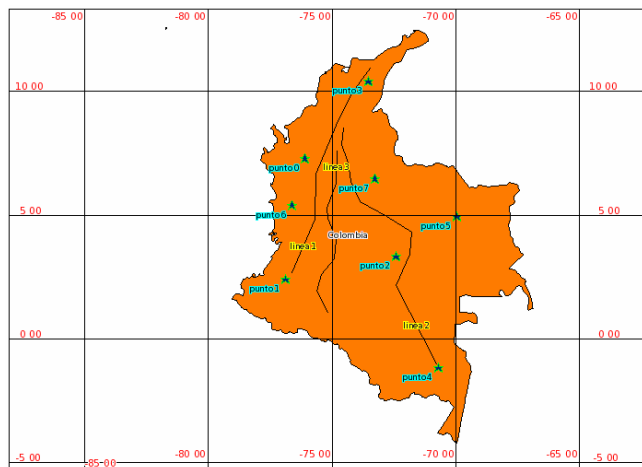
Iniciamos la librería JavaScript OpenLayers.

```
map = new OpenLayers.Map( 'map', { maxResolution: 'auto' } );
var layerPoligono = new OpenLayers.Layer.MapServer("Poligono",
"http://localhost/cgi-bin/mapserv.exe?map=../htdocs/MFD/ejemplo11.map",
{layers: 'Poligonos'}, {isBaseLayer: true,buffer: 1, gutter:0} );
```

Creamos el objeto mapa y agregamos la capa de mapserver haciendo una llamada a nuestro servicio WMS).

```
map.addLayers([layerPoligono,layerLineas,layerPuntos]);
```

Agregamos las capas a nuestro objeto de map.



Si desea añadir etiquetas (**grillas**) a la salida grafica resultante, remítase al ejemplo grilla.map y grilla.php, puedes ver el ejemplo digitando en el navegador la url: [http://localhost/mfd\\_win/grid.php](http://localhost/mfd_win/grid.php).

## Mapas Dinámicos

En la onda de las web2.0, mscross un cliente GIS Ajax OpenSource el cual lo podemos obtener en la siguiente URL: <http://sourceforge.net/projects/mscross> , escrito en JavaScript el cual nos permite realizar una implementación limpia y sencilla de nuestra información geo-espacial.

### Ejemplo13.php corresponde

De este archivo podemos rescatar los siguientes bloques de código.

```
<script src="misc/lib/mscross-1.1.9.js" type="text/javascript">
</script>
```

Implementamos la librería JavaScript mscross.

```
myMap1 = new msMap( document.getElementById("dc_main"), 'standardRight' );
myMap1.setCgi( '/cgi-bin/mapserv.exe' );
myMap1.setMapFile( '/ms4w/Apache/htdocs/mfd_win/ejemplo13.map' );
myMap1.setFullExtent( -88 , -62, -5);
myMap1.setLayers( 'Poligonos Lineas Puntos' );
```

Una vez creada la instancia de mscross, pasaremos parámetros de configuración.

```
myMap1.redraw();
```

Dibujo el mapa utilizando mscross.

```
<input CHECKED onClick="chgLayers()" type="checkbox" name="layer[0]" value="Poligonos">
```

Control de capas.

```
myMap1.setLayers( list );
```

Asigno a objeto map de mscross, la lista de layers a desplegar.

El resultado lo podemos ver digitando en nuestro navegador la url:

[http://localhost/mfd\\_win/ejemplo13.php](http://localhost/mfd_win/ejemplo13.php)

representa nuestro mapa visualizado utilizando la librería Ajax de mscross.



[http://localhost/mfd\\_lin/ejemplo13.php](http://localhost/mfd_lin/ejemplo13.php)

## Añadiendo Capas Raster

El resultado lo podemos ver digitando en

nuestro navegador la url: [http://localhost/mfd\\_win/ejemplo14.php](http://localhost/mfd_win/ejemplo14.php)

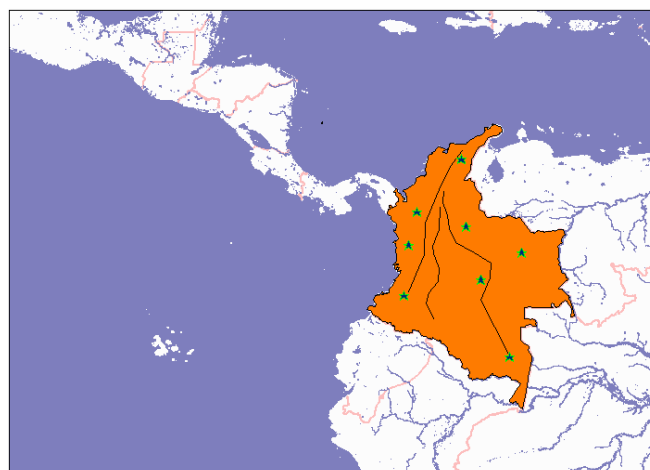
Una capa WMS es una capa raster.

Pero si necesitamos agregar otro tipo de dataset de tipo raster recomendamos checkear la documentación:

[http://mapserver.gis.umn.edu/docs/howto/raster\\_data](http://mapserver.gis.umn.edu/docs/howto/raster_data)

```
LAYER
  NAME landsat
  DATA "landsatimagen.tif"
  STATUS DEFAULT
  TYPE RASTER
  PROCESSING "BANDS=1,2,3"
  OFFSITE 71 74 65

  PROJECTION
    "init=epsg:4326"
  END
END
```



Agregando bandas 1,2,3 de una imagen landsat en .TIF



A través de **ogr2ogr** en FWtools, usted puede realizar múltiples conversiones de información espacial mediante la línea de comandos.

Por ejemplo; convertir de un ESRI shape a OGC GML tan solo basta digitar. **ogr2ogr -f "GML" puntos.gml puntos.shp** o convertir una cobertura a KML para ser visualizada en GoogleEarth digitando; **ogr2ogr -f KML puntos.kml puntos.shp** lo conseguiremos.



## Borrando Cache ( Solo ).

Cada vez que sea invocado mapserver realizándole la petición de renderizar una nueva vista o mapa este generara un archivo de cache (imagen) con la vista actual, si tenemos grandes volúmenes de consultas a la aplicación desarrollada, nuestro disco duro se llenara en un abrir y cerrar de ojos. Mediante este script borraremos el cache, digitando en la barra direcciones del navegador: [http://localhost/mfd\\_win/delcache.php](http://localhost/mfd_win/delcache.php) , este script viene acompañado de una sencilla rutina en batch ubicada en `tmp/del_cache.bat` la cual es invocada para realizar el borrado.

## Más Ejemplos

A manera de extra se presentan una serie de ejemplos.

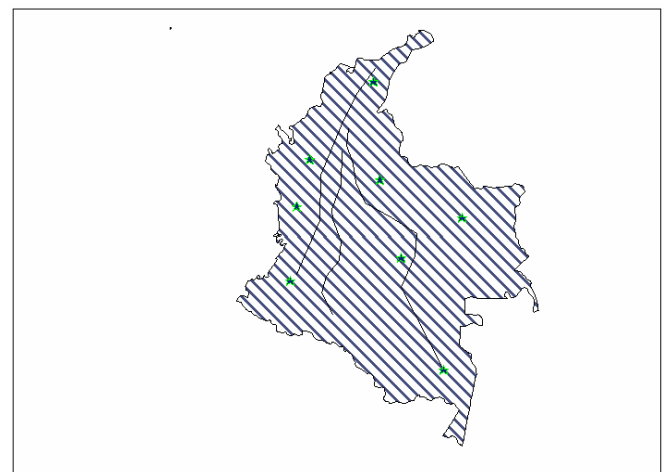
### Achurado

```
SYMBOL
  NAME 'achurado'
  TYPE HATCH
END
```

Defino el símbolo que será un achurado.

```
STYLE
  SYMBOL 'achurado'
  ANGLE -45
  SIZE 10
  WIDTH 3
  COLOR 69 78 124
  OUTLINECOLOR 0 0 0
END
```

Asigno el símbolo de achurado al estilo.



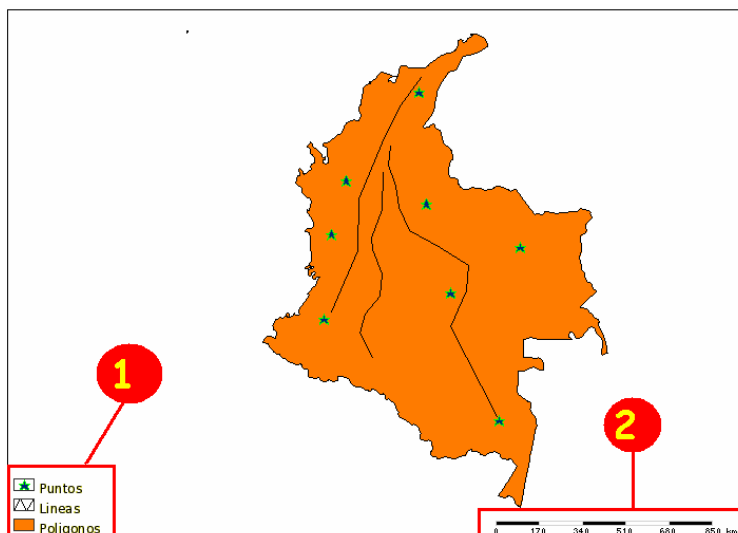
[http://localhost/mfd\\_lin/achurado.php](http://localhost/mfd_lin/achurado.php)



El resultado lo podemos ver digitando en nuestro navegador la url:

[http://localhost/mfd\\_win/achurado.php](http://localhost/mfd_win/achurado.php)

### Escala y Leyenda Embebida



Al igual que los objetos escala (2) y leyenda (1) se renderizan por separado, podemos embeber estos dentro del mismo mapa, checkea los archivos;

**escala\_leyendaembebida.php**  
**escala\_leyendaembebida.map**

El resultado lo podemos ver digitando en nuestro navegador la url:



[http://localhost/mfd\\_win/escala\\_leyendaembebida.php](http://localhost/mfd_win/escala_leyendaembebida.php)

[http://localhost/mfd\\_lin/escala\\_leyendaembebida.php](http://localhost/mfd_lin/escala_leyendaembebida.php)



## Includes

Al igual que en algunos lenguajes, la estructura MAP permite incluir trozos o fragmentos de archivos y ponerlos a funcionar dentro de la implementación, el ejemplo **includes.php** usa **includes.map** y este a su vez usa **includeslayer.map**. El resultado lo podemos ver digitando en nuestro navegador la url:



[http://localhost/mfd\\_win/includes.php](http://localhost/mfd_win/includes.php)

[http://localhost/mfd\\_lin/includes.php](http://localhost/mfd_lin/includes.php)

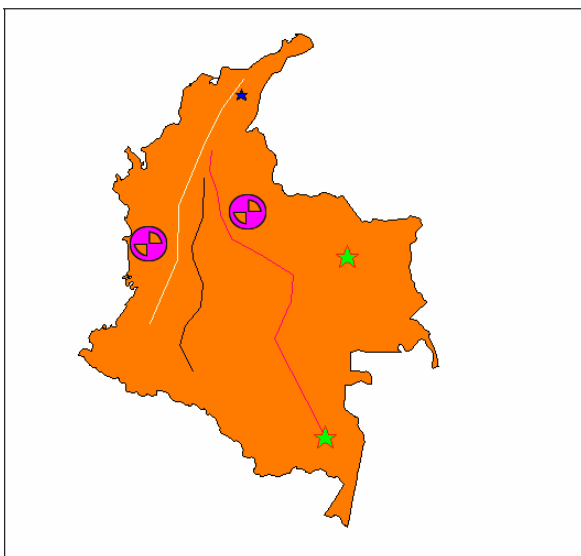


## SLD

Si alguna vez nos preguntamos como extraer el SLD<sup>19</sup> de nuestro mapfile, la respuesta la tenemos digitando en nuestro navegador la url: [http://localhost/mfd\\_win/generarSLD.php](http://localhost/mfd_win/generarSLD.php) una vez generado el resultado, en el navegador vamos a la opción Ver Código Fuente y guardamos esta salida como un documento .xml el cual podremos utilizar para definir el SDL

## Expressions

[http://localhost/mfd\\_lin/generarSLD.php](http://localhost/mfd_lin/generarSLD.php)



Si desea conocer mas acerca de la sintaxis y la potencialidad de expressions, remítase a <http://mapserver.gis.umn.edu/docs/howto/msexpressions> obtendrá información con un mayor nivel de detalle.

Para este ejemplo se ha compuesto una salida grafica que usa el archivo expresiones.map indicando el uso básico de este atributo **EXPRESSION**, el resultado lo podemos ver digitando en nuestro navegador la url: [http://localhost/mfd\\_win/expresiones.php](http://localhost/mfd_win/expresiones.php)

[http://localhost/mfd\\_lin/expresiones.php](http://localhost/mfd_lin/expresiones.php)



---

<sup>19</sup> Styled Layer Descriptor , <http://www.opengeospatial.org/standards/sld>

## Otras Herramientas

**misc/backup\_db.bat** : Realizar backup a base de datos PostgreSQL.

**misc/all\_shapes.bat** : Convertir todos los archivos Shapefile a SQL.

**misc/del\_cache.bat** : Borrar cache imágenes generados por MapServer.

**misc/fast\_shp2pgsql.bat:**

Insertar rápida y limpiamente todos archivos Shapefile los shp a una base de datos PostgreSQL.

**misc/pgdb2shp.bat** : Genera shp de tabla en base de datos PostgreSQL.

**PLUS:** Cargar shapefiles a Postgis por medio de scripts en Bash para GNU/Linux  
Escritos por German Carrillo (<http://geotux.tuxfamily.org/>)

**mfd\_lin/misc/bash\_shp2pgsql.sh** : Cargar todos los archivos Shapefile de un directorio a una base de datos de PostgreSQL/Postgis empleando la herramienta shp2pgsql.

**mfd\_lin/misc/bash\_Shapefile\_may\_a\_min.sh** : Cambiar extensiones de archivos Shapefiles de mayúsculas a minúsculas (.SHP a .shp).

**mfd\_lin/misc/bash\_ogr2ogr.sh** : Cargar todos los archivos Shapefile de un directorio a una base de datos de PostgreSQL/Postgis empleando la herramienta ogr2ogr.

## Enlaces

Algunos enlaces y documentos de interés;

<b>PostgreSQL</b>	<a href="http://www.postgresql.org">http://www.postgresql.org</a>
<b>PostGIS</b>	<a href="http://postgis.refractory.net">http://postgis.refractory.net</a>
<b>Ubuntu</b>	<a href="http://www.ubuntu-es.org">http://www.ubuntu-es.org</a>
<b>PhpExperto:</b>	<a href="http://phpexperto.blogspot.com/">http://phpexperto.blogspot.com/</a>
<b>MsCross:</b>	<a href="http://sourceforge.net/projects/mscross">http://sourceforge.net/projects/mscross</a>
<b>Ne@Polis</b>	<a href="http://umn.mapserver.ch/index_en.php">http://umn.mapserver.ch/index_en.php</a>
<b>FWTools</b>	<a href="http://fwtools.maptools.org/">http://fwtools.maptools.org/</a>
<b>Php/Mapscript</b>	<a href="http://mapserver.gis.umn.edu/docs/howto/phpmapscript-byexample">http://mapserver.gis.umn.edu/docs/howto/phpmapscript-byexample</a>
<b>MapServer</b>	<a href="http://ms.gis.umn.edu/">http://ms.gis.umn.edu/</a>
<b>GDAL</b>	<a href="http://www.gdal.org/ogr/">http://www.gdal.org/ogr/</a>
<b>PHP</b>	<a href="http://www.php.net">http://www.php.net</a>
<b>Instituto Humboldt</b>	<a href="http://www.humboldt.org.co/humboldt/">http://www.humboldt.org.co/humboldt/</a>
<b>GDAL</b>	<a href="http://www.gdal.org/ogr/">http://www.gdal.org/ogr/</a>
<b>Mapping Tools</b>	<a href="http://www.maptools.org">http://www.maptools.org</a>
<b>GeoTux</b>	<a href="http://geotux.tuxfamily.org">http://geotux.tuxfamily.org</a>

## Agradecimientos

Esta guía fue elaborada, a partir de navegación de muchos enlaces, saltando de enlace en enlace uno se topa con tutoriales, ejemplos, y otra serie de artículos que clarifican el panorama, así que quiero dar gracias a esas personas de las cuales tome cierto tipo de información.

*Jaime M. Tan Nozawa* / *phpExperto* - “Trabajando con MapServer”.

*Tyler Mitchell* / *O'Reilly Media Inc* - “Web Mapping Illustrated”.

*Jeff McKenna* / *DMSolutions* – “MapFile Reference”.


*Chris Tweedie* / *Chris GISmos* – “10 Easy steps for converting mxd to map & sld”.

*Jean David Techer* / “Manuel De L'utilisateur De Postgis”.

*Paul Ramsey* / <http://www.cleverelephant.ca/>

*Germán Carrillo* / <http://geotux.tuxfamily.org>

## Acerca del Autor

	<p><b>Fabio Andrés Herrera</b> Ingeniero Topográfico / Universidad del Valle Santiago de Cali - Colombia</p> <p><b>Sitio Personal:</b> <a href="http://andres.hrglobalideas.com">http://andres.hrglobalideas.com</a> <b>Corporativo:</b> <a href="http://www.hrglobalideas.com">http://www.hrglobalideas.com</a> <b>Blog:</b> <a href="http://andresherreracali.blogspot.com">http://andresherreracali.blogspot.com</a> <b>E-mail:</b> t763rm3n ( at ) gmail.com fandresherrera ( at ) hotmail.com <b>IM:</b> <b>t763rm3n</b> / en GTalk <b>Skype:</b> fabio.andres.herrera</p>
---	---