

CLASE
5

Bases de datos espaciales

República Argentina
Ministerio de Educación
Programa Nacional Mapa Educativo

Curso de capacitación
Bases de datos espaciales

CLASE 5
Material de lectura. Versión 1

Temas de esta clase:

Función POPULATE_GEOMETRY_COLUMNS
SQL espacial: análisis de conjunto
Análisis espacial con PostGIS: funciones de geoprocso.
Geocodificación de domicilios.
Geocodificación en base a datos alfanuméricos.

Referencias:

A lo largo del documento encontraremos íconos y recuadros que requieren de una especial atención de los lectores:



ACTIVIDADES: son consignas de actividades para realizar la práctica con gvSIG y PGAdmin acompañando la lectura. En la presente clase hay 4 actividades para resolver.



¡IMPORTANTE! Indica una actividad que no debe omitirse para poder desarrollar correctamente la práctica de la clase.

1 POPULATE_GEOMETRY_COLUMNS

Para empezar esta clase, vamos a ver una función que nos será muy útil a la hora de registrar los numerosos campos geométricos que iremos creando a lo largo del curso. POPULATE_GEOMETRY_COLUMNS se utiliza para evitar el llenado manual del registro de las columnas geométricas. Se pasa como parámetro el nombre de la tabla cuyo campo geométrico deseamos registrar.

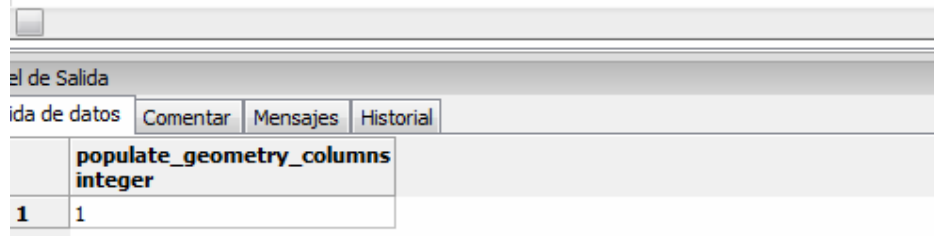
Por ejemplo, si deseamos registrar la columna de la vista “salta.v_mancha_urbana”, corremos la siguiente consulta:

```
SELECT POPULATE_GEOMETRY_COLUMNS  
( 'salta.prueba_populate'::regclass );
```


Los dos puntos y la palabra “regclass” indican que se debe transformar el contenido de la información obtenida a partir de la tabla que indicamos, para que pueda ser utilizado para llenar el registro de la tabla “geometry_columns”.

El resultado de la consulta muestra la cantidad de registros que han sido creados en “geometry_columns” (uno por cada campo geométrico registrado).

```
SELECT POPULATE_GEOMETRY_COLUMNS('salta.prueba_populate'::regclass);
```



	populate_geometry_columns
1	1

Y el producto de esta operación se visualiza al abrir la tabla “geometry_columns” con el botón .

	oid	f_table_catalog	f_table_schema	f_table_name	f_geometry_name	coord_dimension	srid	type
		[PK] character varying(256)	[PK] character varying(256)	[PK] character varying(256)	[PK] character varying(256)	integer	integer	character varying(256)
26	18357	"	salta	hidrografia	the_geom	2	22183	MULTILINESTR
27	18373	"	salta	pacientes	the_geom	2	22183	POINT
28	18342	"	salta	plaza_ficticia	the_geom	2	22183	MULTIPOLYGON
29	18327	"	salta	plazas	the_geom	2	22183	MULTIPOLYGON
30	18744	"	salta	prueba_populate	the_geom	2	22183	MULTIPOLYGON
31	18749	"	salta	prueba_simplify	the_geom	2	22183	MULTILINESTR
32	18231	"	salta	salud	the_geom	2	22183	POINT
33	18761	"	salta	semaforos	the_geom	2	22183	POINT
34	18292	"	salta	transporte	the_geom	2	22183	MULTILINESTR
35	18311	"	salta	transporte_especial	the_geom	2	22183	MULTILINESTR
36	18718	"	salta	transporte_especial_nuevo	the_geom	2	22183	MULTILINESTR
37	18851	"	salta	v_geocodificar	geocodificar	2	22183	POINT
38	18734	"	salta	v_mancha_urbana	the_geom	2	22183	MULTIPOLYGON
39	18820	"	salta	v_polygon	st_makepolygon	2	22183	POLYGON
40	18915	"	salta	v_riego2	the_geom	2	22183	MULTIPOLYGON
41	18769	"	salta	v_semaforos	the_geom	2	22183	POINT

A partir de ahora tenemos tres opciones para registrar las columnas geométricas. Una es entrando a la tabla y modificando los valores de las celdas, otra es con una consulta SQL, usando el comando “INSERT”; y finalmente la opción que acabamos de ver: POPULATE_GEOMETRY_COLUMNS.

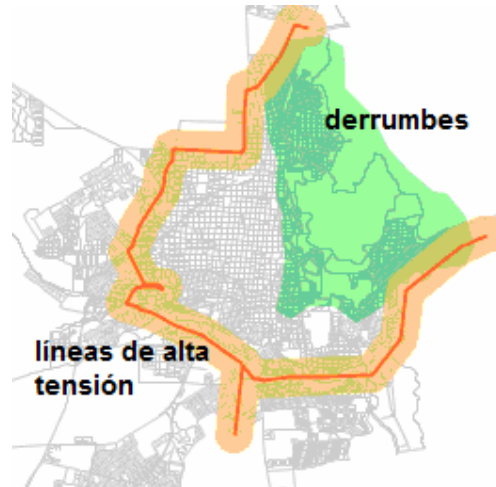
SQL espacial: Análisis de conjunto.

En este apartado veremos funciones de PostGIS que nos permiten realizar análisis de conjunto entre elementos de las capas, para obtener como resultado nuevas geometrías. A su vez, estas nuevas geometrías sirven para realizar nuevos procesos de análisis espacial.

1.1 ST_UNION

Esta primera función devuelve como resultado una geometría que representa la unión de dos o más geometrías.

Volvamos a los ejemplos de la Ciudad de Salta. Si queremos obtener una sola geometría de toda el área de la ciudad que es vulnerable frente a riesgos ambientales, podemos sumar el área de influencia de la línea de alta tensión y la zona de derrumbe.



La forma en que se estructura esta función para unir dos geometrías es:

```
SELECT ST_UNION (GEOMETRIA1, GEOMETRIA2);
```

Debemos reemplazar “GEOMETRIA” 1 y 2 con las consultas que nos traen las dos geometrías que deseamos unir:

```
SELECT ST_UNION (
  ( SELECT the_geom FROM salta.zona_derrumbe ),
  (SELECT BUFFER (the_geom, 500) FROM salta.alta_tension ));
```

Si corremos esta consulta, nos devolverá un campo geométrico que contiene la unión de las dos geometrías. Pero, debemos recordar que para poder visualizar este campo geométrico desde una vista de gvSIG, necesitamos crear una vista en Postgres, y dicha vista debe tener un campo que se pueda utilizar como clave primaria. Y es por esto que una de las dos geometrías de la función ST_UNION no la vamos a traer como subconsulta, sino como campo de una tabla que invocaremos en “FROM”:

```
SELECT gid, ST_UNION (the_geom,
  (SELECT BUFFER (the_geom, 500) FROM salta.alta_tension ) )
FROM salta.zona_derrumbe;
```

De esta manera, hemos logrado que la consulta nos devuelva un campo numérico con valores únicos que puede ser utilizado como clave primaria en una vista.

Ahora generaremos la vista y la registraremos para poder visualizar el resultado desde gvSIG.

```
CREATE VIEW salta.v_zonas_riesgo AS
SELECT gid, ST_UNION (the_geom,
```

```
(SELECT BUFFER (the_geom, 500) FROM salta.alta_tension ) ) AS
the_geom
FROM salta.zona_derrumbe;
```

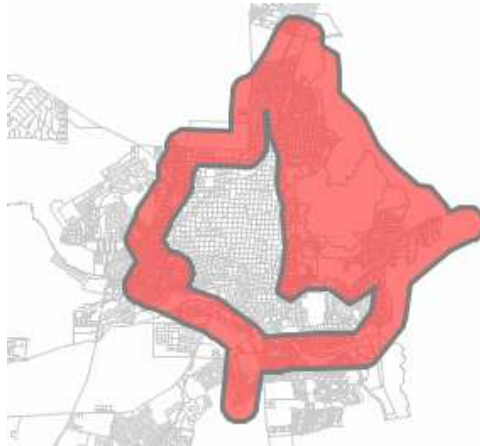
La vista la hemos creado en el esquema “salta” y además, hemos renombrado como “the_geom” el campo geométrico, que por la función aplicada se llamaba “st_union”.

```
INSERT INTO geometry_columns VALUES ('', 'salta',
'v_zonas_riesgo', 'the_geom', 2, 22183, 'MULTIPOLYGON');
```

También podemos usar la opción “POPULATE_GEOMETRY_COLUMNS”:

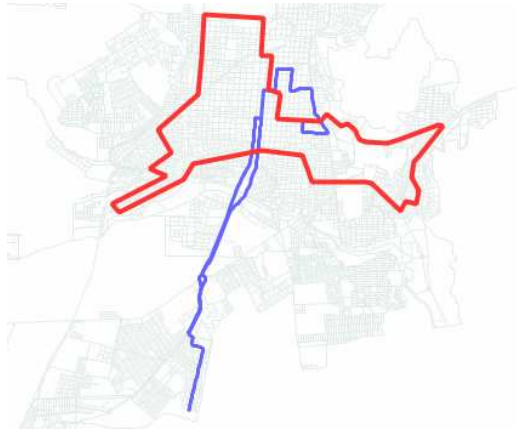
```
SELECT POPULATE_GEOMETRY_COLUMNS ('salta.v_zonas_riesgo' ::
regclass);
```

La consulta para registrar el campo geométrico de la vista inserta el valor “salta” para el campo referido al esquema.



También se pueden unir líneas o puntos. El resultado de la unión de dos líneas puede ser una nueva línea más grande, o bien una multilínea. Una multilínea es un solo elemento, que se corresponde con un solo registro de la tabla, pero que geoméricamente está compuesto de una o más geometrías simples no adyacentes entre sí.

Veamos un ejemplo en donde se una dos líneas. En la clase anterior utilizamos la tabla “salta.transporte_especial” que representaba una línea de transporte adaptada para personas con discapacidad. En esta clase veremos que se ha ampliado el recorrido a través de una nueva línea que recorre las zonas este y oeste de la ciudad.



Si deseamos representar la totalidad de la red de transporte especial, necesitaremos unir las dos líneas. Esta operación puede ser útil, por ejemplo, para volver a calcular la cantidad de pacientes que quedan por fuera del área de influencia del transporte adaptado.

```
SELECT gid, ST_UNION (the_geom,
(SELECT the_geom FROM salta.transporte_especial_nuevo) )
FROM salta.transporte_especial;
```

Si creamos una vista, y la registramos, podremos ver como las dos líneas se unieron para formar una sola:

```
CREATE VIEW salta.v_transporte_especial_unido AS
SELECT gid, ST_UNION (the_geom,
(SELECT the_geom FROM salta.transporte_especial_nuevo) ) AS
the_geom
FROM salta.transporte_especial;
```

```
INSERT INTO geometry_columns VALUES ('', 'salta',
'v_transporte_especial_unido', 'the_geom', 2, 22183,
'MULTILINESTRING');
```

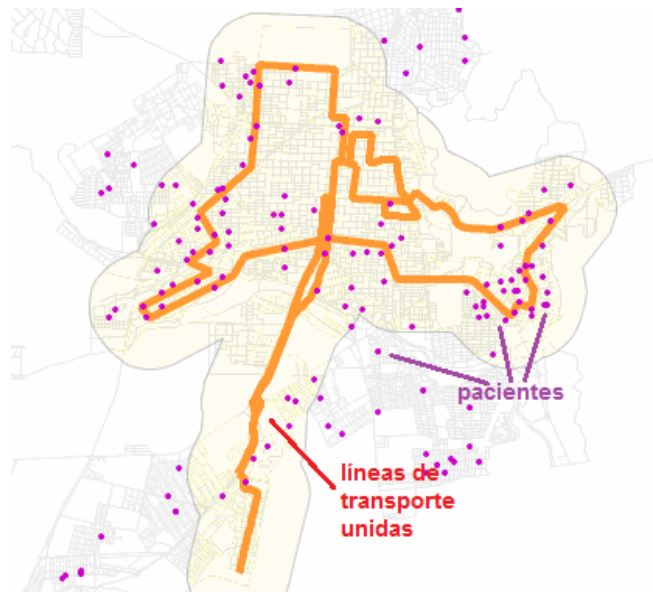
En este caso, al registrar el campo geométrico en la tabla “geometry_columns” hemos tenido la precaución de colocar el tipo de geometría correcta para polilíneas: “MULTILINESTRING”.



Ahora que hemos visualizado y comprobado la unión, podemos volver a correr la consulta para conocer la cantidad de personas con discapacidad que continúan fuera del área de cobertura del transporte especial:

```
SELECT * FROM salta.pacientes WHERE ST_DISJOINT (the_geom,
(SELECT buffer (the_geom,1000) FROM
salta.v_transporte_especial_unido));
```

Como vemos, el resultado anterior que solo tenía en cuenta la primera línea de transporte nos devolvía 138 pacientes, mientras que la segunda consulta, con las dos líneas unidas, nos dice que esta vez hay 61 personas sin cobertura.



También se pueden realizar otras operaciones geométricas sobre estas nuevas geometrías, como el cálculo de la longitud, superficie, perímetro, etc.

```
SELECT LENGTH (ST_UNION (the_geom,
(SELECT the_geom FROM salta.transporte_especial_nuevo) ) ) as
the_geom
FROM salta.transporte_especial;
```

Aquí hemos calculado la longitud de la nueva red de transporte adaptado a las capacidades diferentes: 46 kilómetros.



ACTIVIDAD 1

- Restaurar el archivo "clase5.backup" y correr las consultas del archivo "clase5.sql".
- Calcular la superficie en hectáreas de la zona de riesgo ambiental de la Ciudad de Salta (riego por alta tensión + riesgo de derrumbe).
- Escribir en el documento la respuesta y la consulta SQL que se utilizó para calcular esta superficie.

Además de la unión de dos geometrías, la función ST_UNION también puede unir un conjunto de geometrías. Lo único que hay que tener en cuenta es que la unión solo funciona cuando todos los elementos a unir son del mismo tipo de geometría y tienen el mismo SRID.

Por ejemplo, si queremos obtener un solo gran polígono que represente toda la mancha urbana de la ciudad de Salta, podemos unir todos los polígonos de los barrios:

```
SELECT ST_UNION (the_geom) FROM salta.barrios;
```

Esto nos trae como resultado una sola geometría. Para poder transformarlo en vista, necesitamos un campo numérico y único, pero no podemos traer el campo "gid" de la tabla "salta.barrios", porque el sistema no sabría de cuál de todos los registros unidos obtener el valor de "gid". Es por eso que le pediremos a Postgres que nos traiga uno solo de todos los valores de "gid" a través de una de las funciones de agregación:

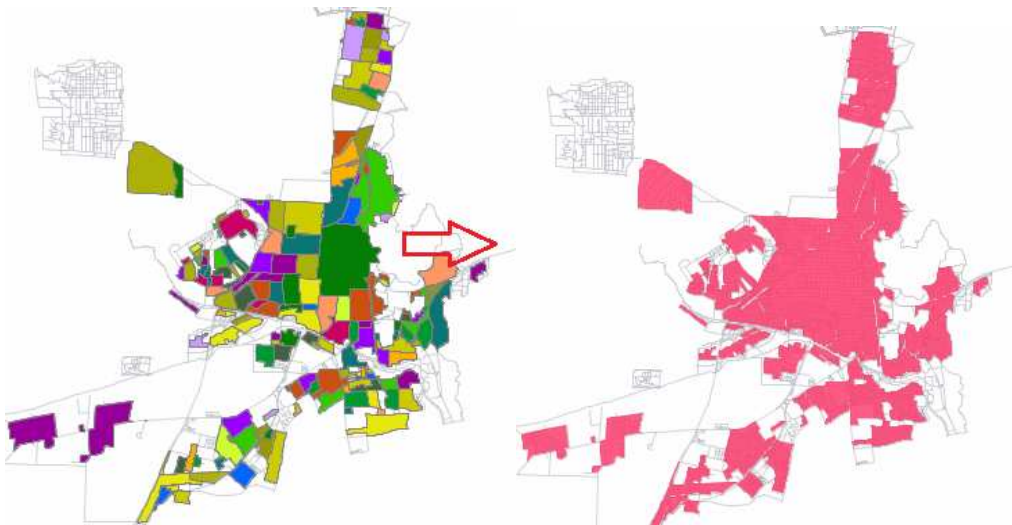
```
SELECT min (gid), ST_UNION (the_geom) FROM salta.barrios;
```

En este caso, optamos por pedirle el valor más pequeño que encuentre. No es importante el valor de "gid" que nos trae, solo nos importa quedarnos con un solo valor, porque necesitamos ese campo como clave única.

Ahora sí, creamos la vista que contiene toda la mancha urbana de Salta:

```
CREATE VIEW salta.v_mancha_urbana AS
SELECT min (gid) AS gid, ST_UNION (the_geom) AS the_geom
FROM salta.barrios;
```

```
INSERT INTO geometry_columns VALUES ('', 'salta',
'v_mancha_urbana', 'the_geom', 2, 22183, 'MULTIPOLYGON');
```



1.2 ST_INTERSECTION

Esta función devuelve la geometría del sector compartido entre dos geometrías. Por ejemplo, si tenemos la zona de derrumbes, y queremos saber cuál es la superficie del barrio “CENTRO” de la ciudad de Salta afectado por este riesgo, aplicamos la función ST_INTERSECTION entre estos dos polígonos. Al igual que en ST_UNION, esta función también requiere que los elementos analizados sean del mismo tipo de geometría y que tengan el mismo SRID.

```
SELECT gid, ST_INTERSECTION (the_geom ,  
(SELECT the_geom FROM salta.zona_derrumbe) ) AS the_geom  
FROM salta.barrios WHERE nom_barrio = 'CENTRO';
```

Al utilizar esta consulta en la definición de la vista, vemos como el nuevo polígono solo cubre la zona del barrio “CENTRO” que se intersecta con el polígono de la zona de derrumbe.

Si queremos conocer la superficie de esta zona, solo agregamos la función “AREA” a la consulta anterior.

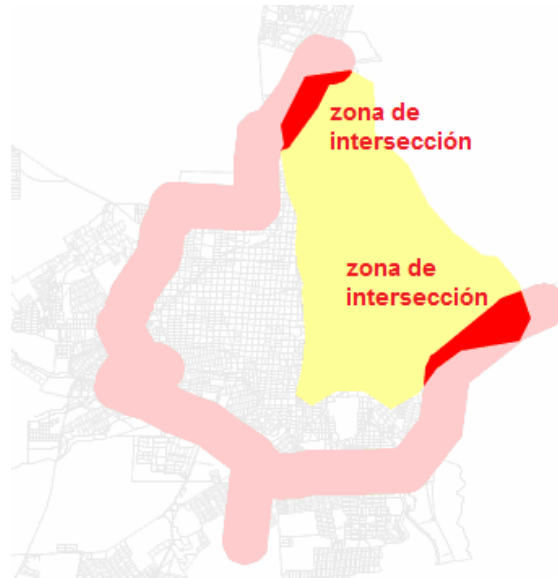
```
SELECT AREA (ST_INTERSECTION (the_geom ,  
(SELECT the_geom FROM salta.zona_derrumbe) ) ) / 10000  
FROM salta.barrios WHERE nom_barrio = 'CENTRO';
```

En este caso, además del cálculo del área a partir de la nueva geometría, hemos dividido el resultado (por 10.000), que originalmente estaba expresado en metros cuadrados, para obtener el valor de superficie en hectáreas, que es más familiar para trabajar sobre un sector de la ciudad.

Veamos otro ejemplo, también con polígonos. Si queremos delimitar el área más afectada por los riegos ambientales, tenemos las herramientas para visualizar la intersección entre el área de influencia de las líneas de alta tensión, y el área de derrumbes.

```
SELECT gid, ST_INTERSECTION (the_geom ,  
(SELECT BUFFER (the_geom, 500) FROM salta.alta_tension ) ) AS  
the_geom  
FROM salta.zona_derrumbe;
```

El resultado es la zona afectada por ambos riesgos.



1.3 ST_DIFFERENCE

Esta función devuelve la porción de la geometría A que no se intersecta con la geometría B. A diferencia de las dos funciones anteriores, en este caso, el orden de las geometrías en la consulta sí es importante, ya que en el resultado que deseamos obtener es la geometría A con el análisis espacial aplicado.

Tomemos como ejemplo las zonas de la mancha urbana de Salta que están fuera del alcance de los centros de Salud. En este caso la geometría A es la mancha urbana de Salta, y la geometría B es el área de influencia de 750 metros de los centros de Salud.

Para obtener A, simplemente seleccionamos la geometría de la vista que hicimos más atrás en esta misma clase:

```
SELECT the_geom FROM salta.v_mancha_urbana;
```

Por otro lado, para obtener B, como un solo polígono tenemos 2 alternativas:

a) Unir los puntos de los centros de Salud, y luego generar una sola área de influencia del multipolígono:

```
SELECT BUFFER ((SELECT ST_UNION (the_geom) FROM
salta.salud), 750);
```

b) Generar un área de influencia para cada punto y luego unirlos.

```
SELECT ST_UNION (BUFFER (the_geom, 750)) FROM salta.salud;
```

Luego, creamos la nueva consulta para analizar las geometrías:

```
SELECT ST_DIFFERENCE (
(SELECT the_geom FROM salta.v_mancha_urbana),
(SELECT ST_UNION (BUFFER (the_geom, 750)) FROM salta.salud));
```

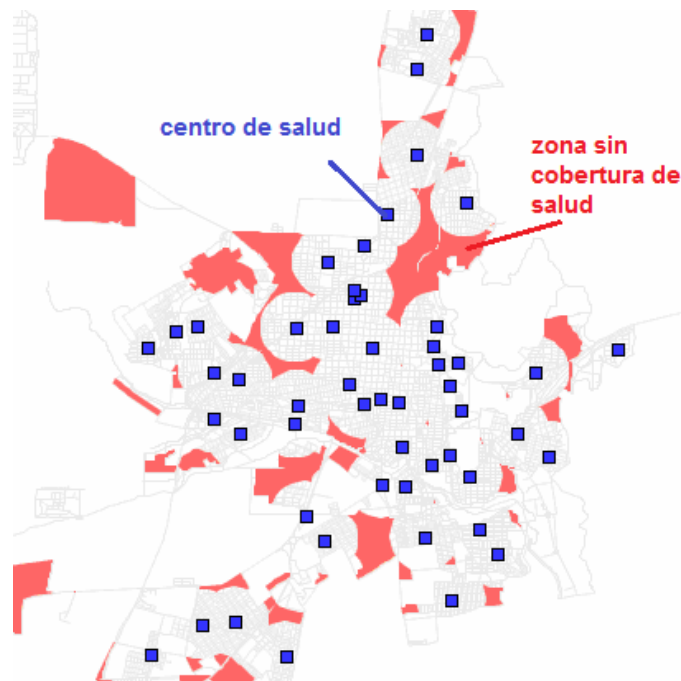
Pero, recordemos que necesitamos obtener un campo numérico para convertirse en la clave primaria de la vista:

```
SELECT gid, ST_DIFFERENCE (the_geom,
(SELECT ST_UNION (BUFFER (the_geom, 750)) FROM salta.salud) )
FROM salta.v_mancha_urbana
```

Es importante mantener la mancha urbana como la primera geometría del análisis (A). Al convertir en vista y registrarlo en “geometry_columns”, vemos que el resultado es la mancha urbana sin las zonas cercanas a los centros de salud.

```
CREATE VIEW salta.v_zona_sin_salud AS
SELECT gid, ST_DIFFERENCE (
the_geom,
(SELECT ST_UNION (BUFFER (the_geom, 750)) FROM salta.salud) ) AS
the_geom
FROM salta.v_mancha_urbana;
```

```
INSERT INTO geometry_columns VALUES ('', 'salta',
'v_zona_sin_salud', 'the_geom', 2, 22183, 'MULTIPOLYGON');
```



1.4 ST_SYMDIFFERENCE

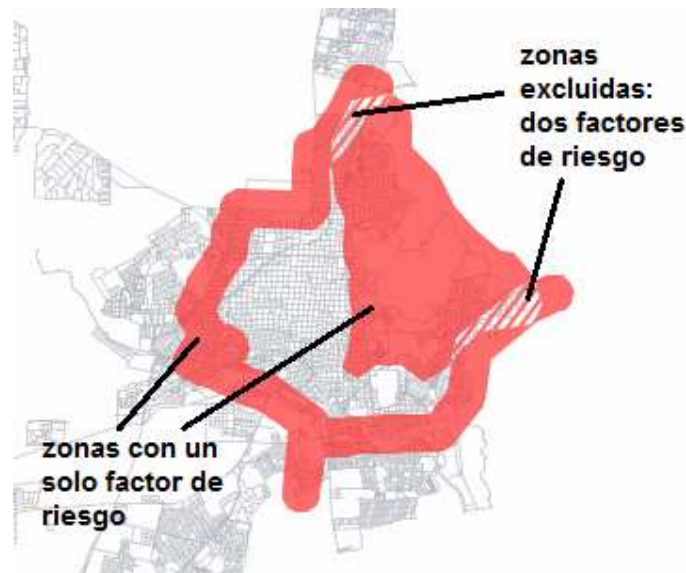
La función “ST_SYMDIFFERENCE” o diferencia simétrica, es la opuesta a “ST_INTERSECTION”. Devuelve las porciones de geometría que no se comparten entre dos elementos.

Esta función también requiere de que las dos geometrías intervinientes en el análisis sean del mismo tipo de geometría y que tengan el mismo SRID.

Retomando el ejemplo de la zona afectada por los dos tipos de riesgo ambiental que existe en la ciudad, ahora vamos a generar un polígono (o multipolígono) correspondiente a las zonas que solo están afectadas por un tipo de riesgo ambiental.

```
SELECT gid, ST_SYMDIFFERENCE (the_geom ,  
(SELECT BUFFER (the_geom, 500) FROM salta.alta_tension ) ) AS  
the_geom  
FROM salta.zona_derrumbe;
```

El resultado es un multipolígono con la zona que solo tiene riesgo por derrumbe y la zona que solo se ve amenazada por la proximidad a las líneas de alta tensión.



1.5 Vistas o consultas complejas.

Para poder realizar nuevos procedimientos de análisis espacial utilizando las geometrías resultantes de ST_UNION, ST_INTERSECTION o ST_SYMDIFFERENCE tenemos dos opciones:

1. Una es crear vistas con estas consultas, de tal forma que luego sea más simple invocar las geometrías resultantes.
2. La otra opción es no crear vistas, e integrar el texto de las consultas realizadas a nuevas consultas.

Obviamente la segunda opción significa generar consultas con una enorme y creciente complejidad, que pueden presentar muchas dificultades para manejarlas. Pero esta opción tiene la ventaja de que evita crear nuevos objetos en la base de datos, que quizá luego no volveremos a utilizar.

Veamos estas dos opciones aplicadas a un ejemplo: Queremos conocer la superficie de la mancha urbana de Salta que está libre de riesgos ambientales.

Con la opción “a”, usando vistas:

```
SELECT ST_DIFFERENCE ( the_geom ,  
(SELECT the_geom FROM salta.v_zonas_riesgo) )  
FROM salta.v_mancha_urbana;
```

Con la opción “b”, usando subconsultas:

```
SELECT ST_DIFFERENCE (  
(SELECT ST_UNION (the_geom) AS the_geom  
FROM salta.barrios),  
(SELECT ST_UNION (the_geom,  
(SELECT BUFFER (the_geom, 500) FROM salta.alta_tension ) ) AS  
the_geom  
FROM salta.zona_derrumbe));
```

Queda a criterio de cada usuario la decisión de utilizar vistas que se invocan unas a otras, o bien consultas con subconsultas sucesivas en su interior.

La recomendación de los docentes de este curso es que **solo se creen vistas cuando el contenido de dichas vistas vaya a ser utilizado nuevamente** y en forma frecuente. En los demás casos, se sugiere almacenar las consultas complejas en archivos “.sql” debidamente documentados y alojados.

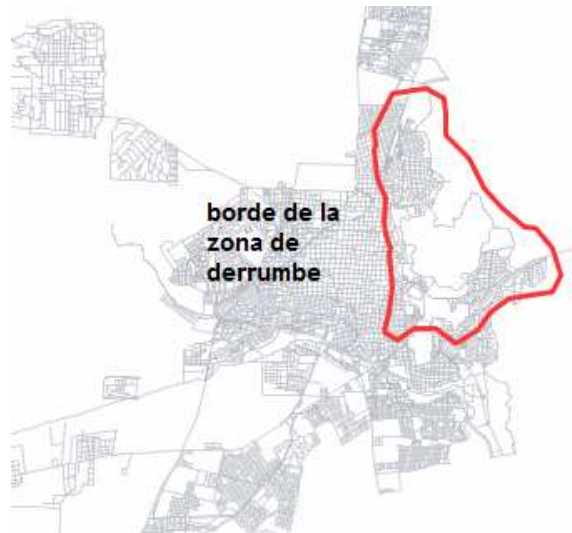
2 Funciones de geoprocursos

Veremos algunas funciones de PostGIS que nos permiten realizar geoprocursos sobre los campos geométricos existentes. Este apartado contiene solo algunas de las funciones más utilizadas. Para obtener más información se recomienda recurrir al manual de PostGIS o al resto de la bibliografía recomendada.

2.1 *ST_BOUNDARY*

La función *ST_BOUNDARY* devuelve la geometría lineal del borde de un polígono. Por ejemplo, podemos obtener el borde del polígono de la zona de derrumbe, para graficar la línea de protección a desplegar en caso de catástrofe.

```
SELECT ST_BOUNDARY (the_geom) FROM salta.zona_derrumbe;
```



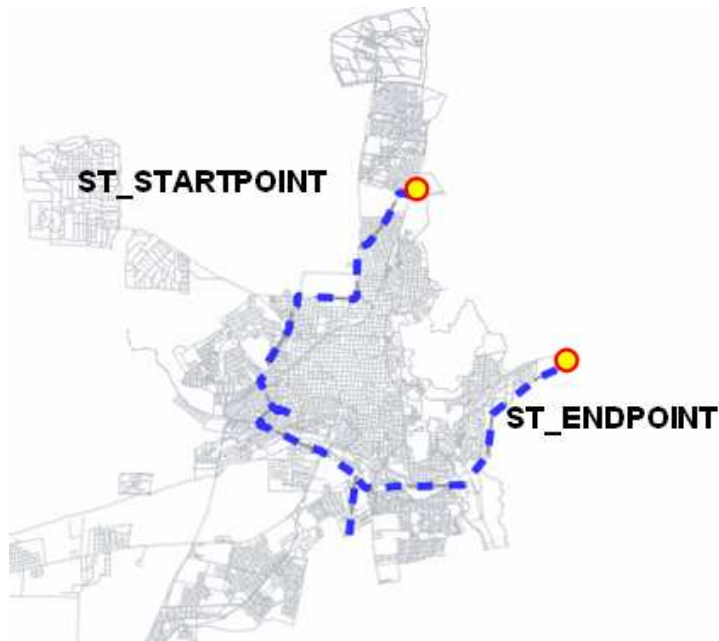
2.2 *ST_STARTPOINT* y *ST_ENDPOINT*

Estas dos funciones se utilizan para obtener la geometría del primero y del último vértice de la figura. Generalmente se utiliza con líneas y polilíneas para conocer el punto de partida o de llegada de un recorrido.

Por ejemplo, podemos visualizar los puntos de inicio y fin de la línea de alta tensión en el tramo que pasa por la ciudad de Salta, para colocar en estos dos puntos sensores de radiación proveniente de los cables electrificados.

```
SELECT ST_STARTPOINT (the_geom) FROM salta.alta_tension;
```

```
SELECT ST_ENDPOINT (the_geom) FROM salta.alta_tension;
```



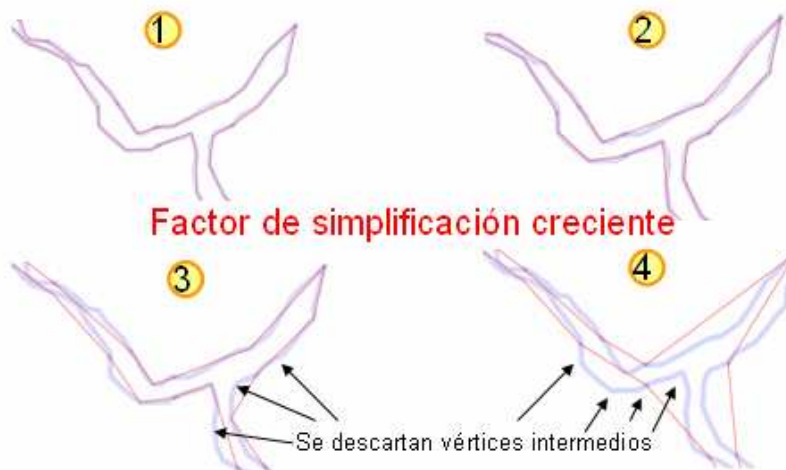
2.3 ST_SIMPLIFY

ST_SIMPLIFY es utilizado para eliminar vértices de alguna figura que a cierta escala resultan redundantes. Es similar, en cierta medida, al proceso de generalización propio de la cartografía, que representa de manera menos detallada los rasgos del territorio a medida que se reduce la escala del mapa.

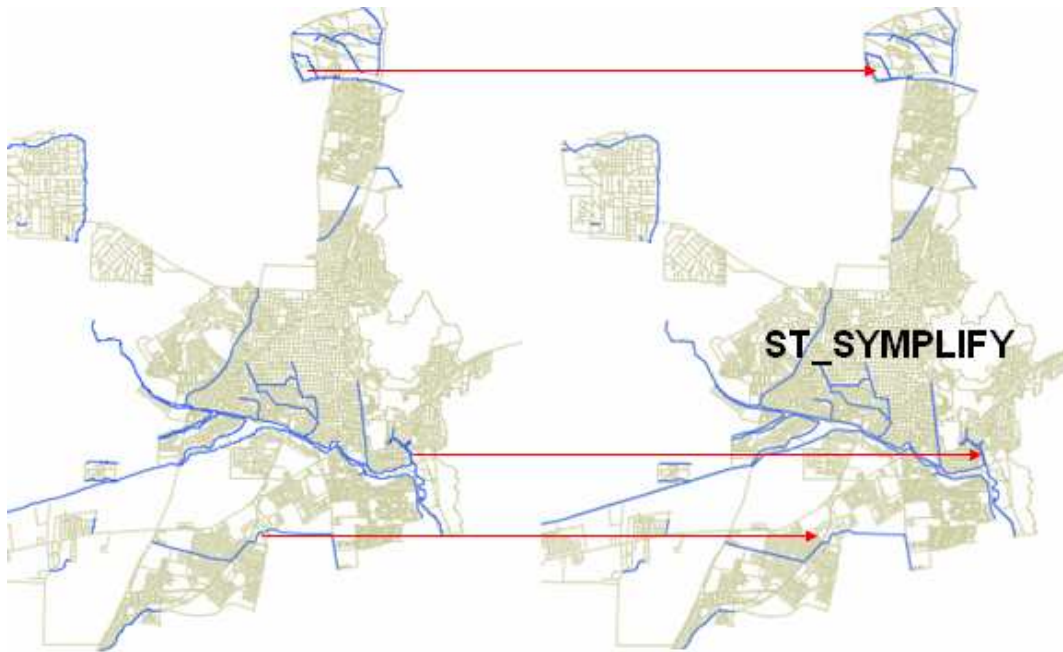
La forma en que se estructura la función es colocando una geometría, y luego un factor de simplificación, que es un número entero. Mientras más grande sea el número, más simple resultarán las geometrías.

```
SELECT ST_SIMPLIFY (the_geom , 25) from salta.hidrografia;
```

Como ejemplo, podemos simplificar el tendido de las líneas de hidrografía , para visualizar claramente en qué consiste esta función.



Esta función es muy útil para reducir el tamaño de la información cuando tenemos información relevada



Redujimos considerablemente la cantidad de vértices de la capa, y su tamaño en disco, pero a esta escala se sigue visualizando adecuadamente la información.

2.4 ST_SYMPPLIFYPRESERVE TOPOLOGY

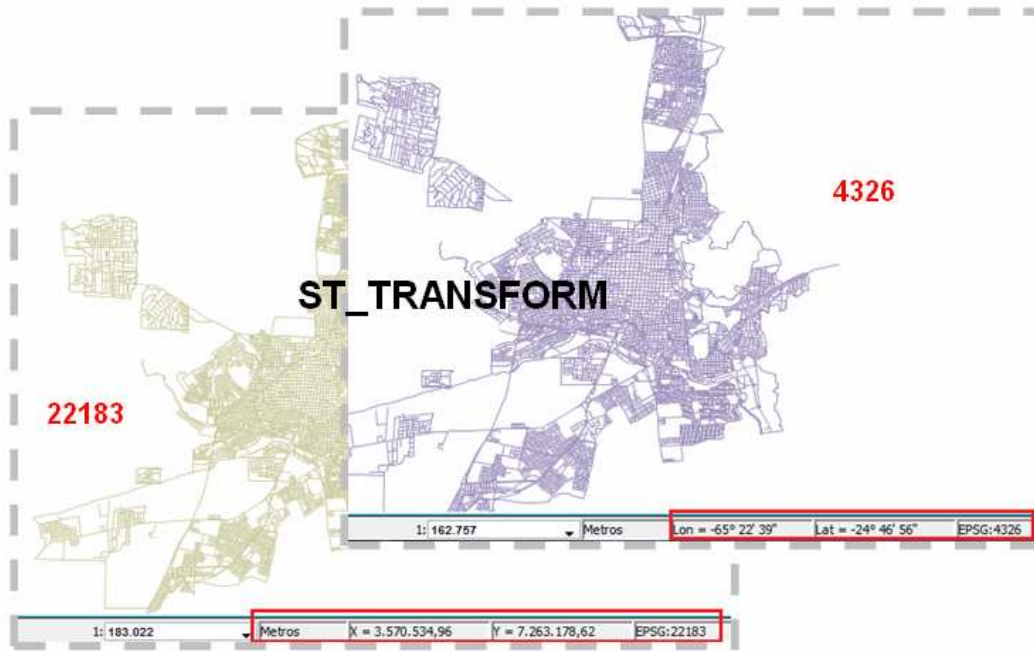
Es similar a ST_SYMPPLIFY, pero el algoritmo que utiliza garantiza la consistencia topológica del resultado. Esto quiere decir que las relaciones que se establecen entre los elementos en el espacio no se ven alteradas.

2.5 ST_TRANSFORM

Esta función es muy importante, porque nos permite transformar información de un sistema de referencia a otro. Esto nos puede evitar tener que reproyectar la información al ingresarla a una vista de gvSIG. También sirve para presentar a cada provincia en el sistema de referencia correspondiente, la información que originalmente tenemos consistida para todo el país y en coordenadas geográficas.

Por ejemplo, si queremos integrar la información de la provincia de Salta con la del resto del país, en primer lugar debemos transformarla a coordenadas geográficas. La forma de utilizar esta función es brindando el código del SRID correspondiente a la proyección a la que queremos llevar la información. Recordemos que dentro del propio campo geométrico ya está almacenado el SRID actual.

```
SELECT TRANSFORM ( the_geom, 4326) from salta.calles;
```

2.6 ST_MAKEPOINT

La función ST_POINT crea una geometría de punto a partir de los valores X e Y de la coordenada. De esta manera, podemos crear puntos tipeando las coordenadas en una consulta SQL, o bien obteniendo los valores de las coordenadas de una tabla con datos alfanuméricos.

```
SELECT MAKEPOINT ( 3560764, 7264713 );
```

Para ser realmente un campo geométrico de un SIG, necesita la información referida al sistema de referencia, y para ello utilizamos la función ST_SETSRID:

```
SELECT ST_SETSRID ( MAKEPOINT ( 3560764, 7264713 ), 22183 );
```

Ahora sí hemos generado un elemento **geográfico** en la base de datos.

Si tenemos una tabla con coordenadas, podemos reemplazar los valores de X e Y por el nombre de los campos de dicha tabla, y de esta manera generar un punto para cada registro. Por ejemplo, en la tabla "salta.semaforos" tenemos un listado de coordenadas de las esquinas donde hay semáforos. Para convertir ese listado de coordenadas en datos geométricos, utilizamos la siguiente consulta:

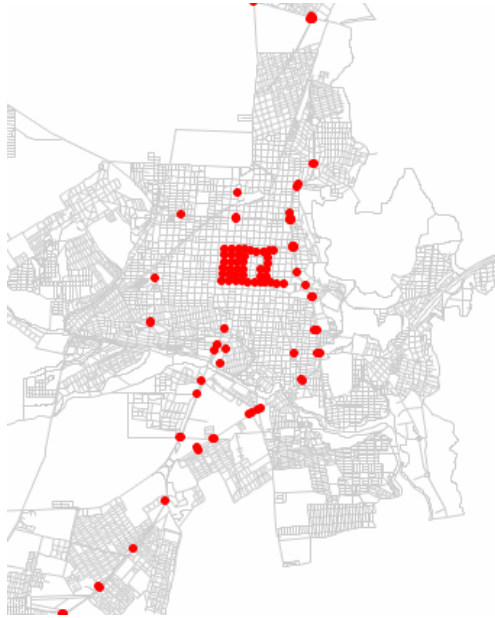
```
SELECT id, ST_SETSRID ( MAKEPOINT ( coord_x, coord_y ), 22183 )
FROM salto.semaforos;
```

Y luego, podemos crear la vista para visualizar los puntos en gvSIG.

```
CREATE VIEW salto.v_semaforos AS
SELECT id, ST_SETSRID ( MAKEPOINT ( coord_x, coord_y ), 22183 ) AS
the_geom
FROM salto.semaforos;

SELECT POPULATE_GEOMETRY_COLUMNS('salta.v_semaforos'::regclass);
```

El campo “id” funciona como clave primaria, y el campo geométrico es renombrado como “the_geom”.



2.7 ST_MAKELINE

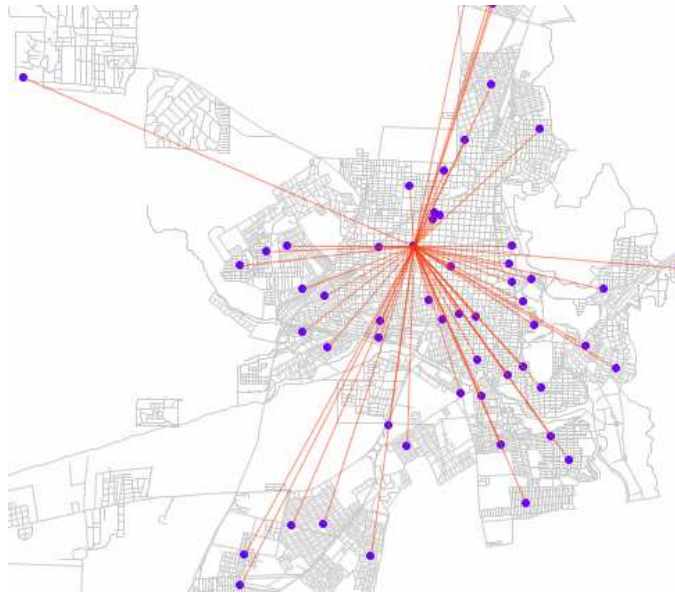
Crea una línea a partir de dos puntos. Por ejemplo, podemos crear una línea que vaya desde la central de emergencias médicas hacia cada centro de salud de la ciudad de Salta. La forma de seleccionar el punto de la central de emergencias es:

```
SELECT the_geom FROM salta.salud WHERE aoperatva = 'SAMEC';
```

Ahora, integramos esta consulta en la consulta que crea las líneas:

```
SELECT gid, ST_MAKELINE (the_geom ,  
  (SELECT the_geom FROM salta.salud WHERE aoperatva = 'SAMEC') )  
AS the_geom  
FROM salta.salud;
```

Como vemos, en este caso no hace falta usar la función “ST_SETSRID” porque las líneas heredan el SRID de los puntos que las originan.



También se puede utilizar `ST_MAKELINE` como función de agregación, es decir, pasándole un conjunto de puntos, para que cree una polilínea usando como vértices a todos estos puntos.

Por ejemplo, podríamos crear el recorrido de transporte para personas con discapacidad generando una parada en cada uno de los domicilios de los pacientes, de tal forma que ninguno quede afuera del servicio.

```
SELECT min (gid) as gid, ST_MAKELINE (the_geom) from  
salta.pacientes;
```

Como usamos `ST_MAKELINE` como función de agregación, pedimos solo el valor mínimo de `gid`, para obtener un campo que sirva de clave primaria.

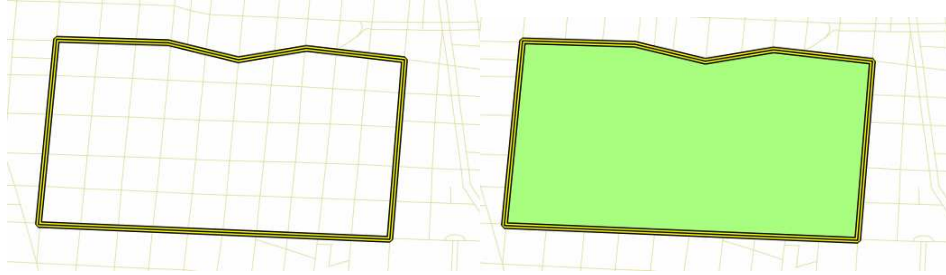
El resultado se ve a continuación, y es, por supuesto, solo un ejemplo:



2.8 ST_POLYGON

Por su parte, esta función crea polígonos a partir de polilíneas cerradas, es decir, polilíneas cuyo punto de inicio coincide con el punto final.

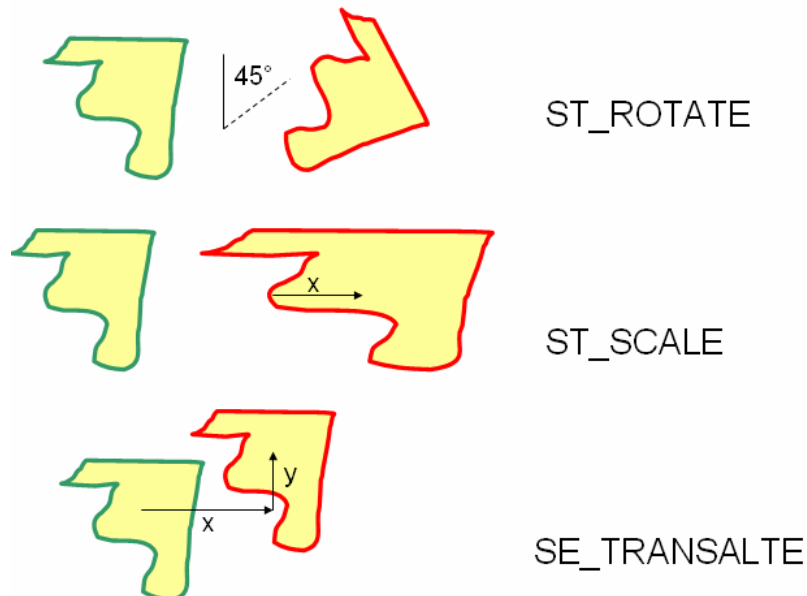
Supongamos que queremos crear un polígono que delimita el área de tránsito central de Salta. Podemos usar como límite la red de calles con carriles exclusivos para transporte público que se encuentra en la tabla "salta.carriles".



```
SELECT ST_MAKEPOLYGON (the_geom) FROM salta.carriles;
```

2.9 Rotar, escalar, trasladar

Existen otras funciones que es poco probable que se utilicen en SIG, pero que son muy comunes en entorno de diseño o dibujo:



ST_ROTATE para rotar una figura. El parámetro a colocar es el ángulo de rotación.

ST_SCALE para escalar una figura. Se coloca un factor de escala con respecto a X, y otro con respecto a Y.

ST_TRANSLATE para desplazar una figura. Se pasa como parámetro la cantidad de unidades del mapa (metros o grados decimales, según la proyección) que se desea desplazar la geometría en los ejes de X e Y.

**ACTIVIDAD 2**

2 - Escribir las consultas SQL para realizar los siguientes procedimientos, y crear las vistas para visualizar las capas en GvSig:

- a) Unir todos los radios censales en un solo polígono para crear la fracción censal “16” del partido de Avellaneda (cod_depto = ‘035’), a partir de los datos de la tabla “radios_censales”.
- b) General el polígono correspondiente con la zona del barrio “CENTRO” de Salta que no está afectado por el riesgo de derrumbe.

3 Geocodificación de domicilios

Este procedimiento consiste en pasar los parámetros de nombre de calle y altura para que Postgres nos devuelva una geometría de punto correspondiente al domicilio.

Actualmente no existe una función dentro de PostGIS para geocodificar domicilios con el formato utilizado en Argentina (calle y número), pero en diferentes instituciones se utilizan funciones que en menor o mayor medida resuelven el problema de la geocodificación de domicilios.

En esta clase usaremos una función muy básica y escrita por alguien que no conoce demasiado de programación, pero que ha creado la función guiándose con el manual de PostGIS.

Para poder utilizar la función, primero debemos correr la siguiente consulta que la crea:

```
CREATE OR REPLACE FUNCTION geocodificar(calle character varying,
numero integer)
    RETURNS geometry AS
$BODY$
    SELECT ST_Line_Interpolate_Point (
geomfromtext (
    REPLACE (REPLACE (REPLACE(astext (the_geom), 'MULTI',''),
'((','(',')','))','')) ,22183),--the_geom
(($2 - desdei)/(hastad-desdei)::float)
    ) FROM salta.calles where nombre ilike $1 and desdei < $2
and hastad > $2
    limit 1;
$BODY$
LANGUAGE sql STABLE
COST 100;
ALTER FUNCTION geocodificar(character varying, integer) OWNER TO
postgres;
```

Una vez que corremos la función, podemos ir a la rama de funciones dentro del árbol de objetos del esquema “public”, y si refrescamos veremos que la función ha sido agregada.

The screenshot shows the pgAdmin interface. On the left, the 'geocodificar(character varying, integer)' function is highlighted in the tree view. On the right, the 'Properties' pane shows the function's details, and the 'Panel SQL' pane displays the function's definition:

```

-- Function: geocodificar(character varying, integer)
-- DROP FUNCTION geocodificar(character varying, integer);

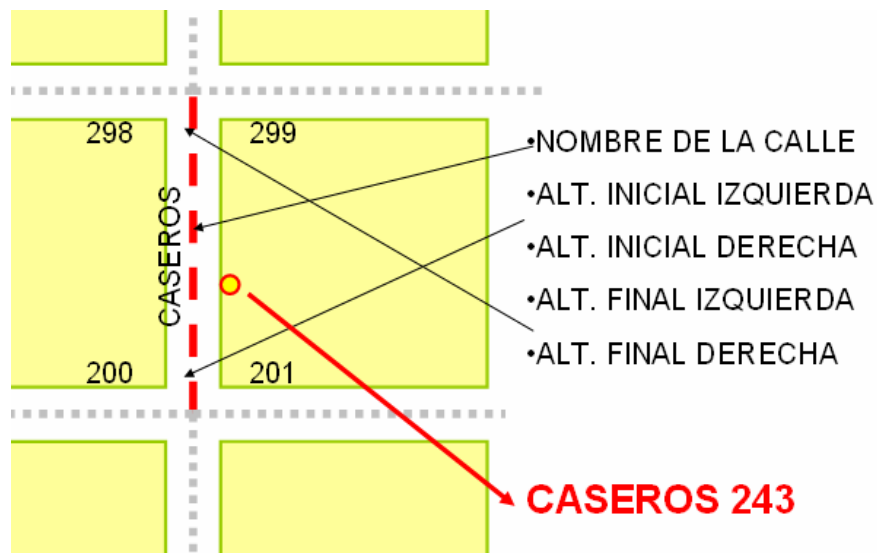
CREATE OR REPLACE FUNCTION geocodificar(calle character varying, numero
      RETURNS geometry AS
$BODY$
    SELECT ST_Line_Interpolate_Point (
geomfromtext (
      REPLACE(REPLACE(REPLACE(astext (the_geom), 'MULTI', ''), '(', ''), ')')
      ((#2 - desdei)/(hastad-desdei)::float)
        ) FROM salta.calles where numero ilike #1 and desdei < #2 and h
      limit 1;
$BODY$
LANGUAGE sql STABLE

```

Si leemos la función, sin entrar en detalles, veremos que la tabla “salta.calles” está implicada. Efectivamente esta tabla contiene los datos que nos servirán para encontrar los domicilios. Los registros de esta tabla son tramos de calle o cuadras. Y los campos que usaremos son:

- “nombre”, que guarda el nombre de la calle,
- “desdei”, que tiene el número más bajo del lado izquierdo de la cuadra,
- “desded”, número más bajo de la derecha,
- “hastaí”, número más alto de la izquierda,
- “hastad”. Número más alto de la derecha.

La función primero pide dos parámetros al usuario: el nombre de la calle y el número, que forman el domicilio. Con esos datos, Postgres busca todos los tramos que tengan ese nombre, y luego el tramo que coincida con el número. Finalmente calcula en qué parte de la cuadra debe colocar el punto en función del número máximo y del número mínimo esa cuadra.



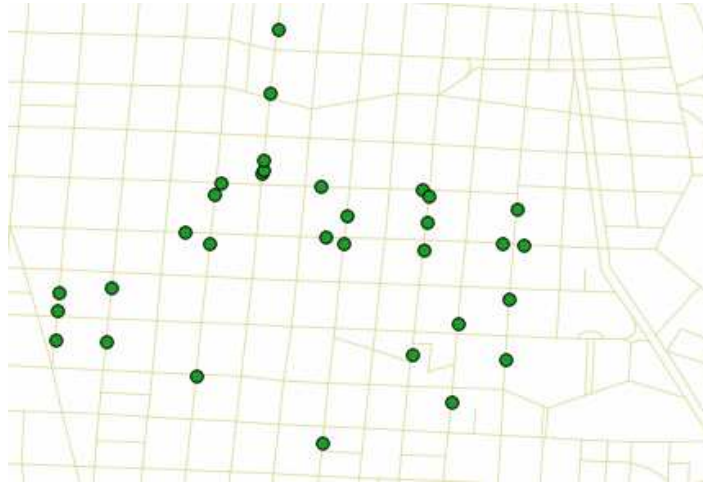
Lo que nos devuelve es una geometría referida al punto del domicilio:

```
SELECT GEOCODIFICAR ('CASEROS', 243);
```

Si tenemos un listado con domicilios en una tabla de postgres, como la tabla “salta.domicilios”, podremos geocodificar en forma masiva reemplazando los parámetros por el nombre de los campos en la consulta:

```
SELECT id, GEOCODIFICAR (calle, numero) FROM salto.domicilios;
```

Al hacer la vista, nos encontramos con los puntos desde gvSIG:



Como hemos visto, esta función es solo para hacer pruebas. Una buena herramienta de geocodificación debería dar, entre otras, las siguientes prestaciones:

- Buscar nombres de calle exactos o parecidos
- Buscar alturas exactas o cercanas
- Devolver el nivel de precisión de la localización conseguida
- Localizar a la izquierda o la derecha del eje de calle (par – impar)
- Buscar intersecciones de calles
- Devolver diferentes opciones ordenadas, empezando por las más acertadas
- Otras búsquedas: por barrio, número de manzana, lugares conocidos, etc.



ACTIVIDAD 3

- Geocodificar los domicilios de “salta.domicilios” y luego hacer un mapa temático que muestre cuáles domicilios fueron inspeccionados (campo “inspeccion”).
- Hacer una impresión de pantalla y añadir al documento de texto.

4 Geocodificación en base a datos alfanuméricos.

En esta clase vimos dos formas de georreferenciar: con la función ST_MAKEPOINT podemos crear puntos a partir de coordenadas almacenadas en una tabla; y también vimos una introducción a la geocodificación por domicilios. Pero existe otra forma de georreferenciar basándonos en información alfanumérica referida a entidades geográficas. Por ejemplo, podemos recibir un listado de localidades y parajes en donde se aplica una política pública relacionada con nuestra institución, y con esa información podemos visualizar los lugares en un mapa para analizar su distribución territorial.

Como ejemplo, tenemos la tabla “alerta_sismica” que solo contiene un listado de códigos de aglomerado. Estos códigos podemos utilizarlos para hacer JOIN con la tabla de localidades, y de esa manera obtener las localidades relacionadas con el programa de alerta por sismos.

```
SELECT a.* FROM localidades a JOIN alerta_sismica b  
ON a.cod_aglo = b.cod_aglo;
```

Y el resultado es un subconjunto de localidades.



Otra situación que suele darse, es que en lugar de listados con códigos de entidades, recibimos listados con **nombres** de entidades. Esto suele presentar dificultades, ya que pueden existir errores de tipeo, o nombres alternativos para cada entidad. En este caso, vamos a ver un ejemplo en donde la información muestra los lugares de Salta en donde se han instalado laboratorios con computadoras de acceso público. Lo que nos piden es mapear las localidades que tienen laboratorios, y luego analizar cuales son los laboratorios que faltan equipar y qué alternativas se puede dar desde el análisis espacial.

Los datos recibidos están en la tabla “laboratorios”. Vamos a unirlos con la tabla de localidades, pero esta vez usando el nombre de la localidad para hacer el JOIN.

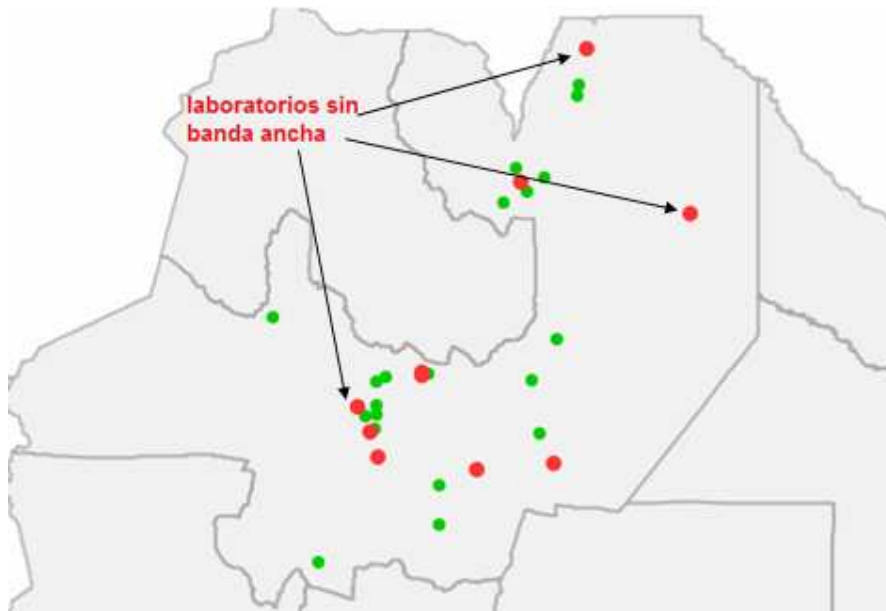
```
SELECT a.gid, a.the_geom, b.* FROM localidades a JOIN
laboratorios b
ON a.nom_loc = b.localidad;
```

Con esto comprobamos que la unión trae 38 registros, cuando en la tabla de laboratorios solo hay 30 registros. Lo que está ocurriendo es que la tabla de localidades tiene los puntos de todo el país, y al hacer el JOIN con el nombre de la localidad, puede ser que existan otras localidades con el mismo nombre en otras provincias, y que erróneamente se están uniendo a la tabla de laboratorios.

Para restringir la participación de las localidades en el JOIN, vamos a seleccionar solamente las localidades cuya geometría se intersecte con el polígono de la provincia de Salta. Recordemos que las provincias estaban en faja 4, y que por eso debemos reprojectarlas antes de analizar si se intersectan con las localidades.

```
SELECT a.gid, a.the_geom, b.* FROM localidades a JOIN
laboratorios b
ON a.nom_loc = b.localidad
WHERE INTERSECTS (a.the_geom, (SELECT TRANSFORM (the_geom, 22183)
FROM provincias WHERE nom_prov = 'SALTA'));
```

Ahora sí, hemos obtenido una tabla con geometrías que puede ser visualizada desde gvSIG.





ACTIVIDAD 4

- Geocodificar las localidades que tienen antenas de WIMAX (WIFI de largo alcance), usando la tabla "salta.localidades_wimax".
- Crear áreas de influencia las antenas de WIMAX utilizando el campo "alcance" para definir el radio de influencia. Atención: el alcance está expresado en KM.
- Calcular cuáles son los laboratorios SIN banda ancha que quedan por fuera de las áreas de cobertura de WIMAX.
- Agregar al documento de texto las consultas SQL escritas para este ejercicio y una impresión de pantalla de las áreas de influencia WIMAX y los laboratorios clasificados según presencia de banda ancha.

- Bibliografía

OLAYA, Víctor. (2011). *Sistemas de Información Geográfica*. Versión 1.0, Rev. 24 de marzo de 2011. Proyecto “Libro Libre SIG”.

http://forge.osor.eu/docman/view.php/13/577/Libro_SIG.zip

THE POSTGIS TEAM. Manual de PostGIS 1.5.3.

<http://www.postgis.org/download/postgis-1.5.3.pdf>

OBE, Regina y HSU Leo. (2011). *PostGIS in Action*. Editorial Manning. Stamford.

Índice

1 POPULATE GEOMETRY COLUMNS.....	1
1.1 ST UNION	2
1.2 ST INTERSECTION	7
1.3 ST DIFFERENCE	8
1.4 ST SYMDIFFERENCE	10
1.5 VISTAS O CONSULTAS COMPLEJAS.	11
2 FUNCIONES DE GEOPROCESOS.....	13
2.1 ST BOUNDARY	13
2.2 ST STARTPOINT Y ST_ENDPOINT	13
2.3 ST_SIMPLIFY	14
2.4 ST_SYMPLOYPRESERVE TOPOLOGY	15
2.5 ST_TRANSFORM	15
2.6 ST_MAKEPOINT	16
2.7 ST_MAKELINE	17
2.8 ST_POLYGON	19
2.9 ROTAR, ESCALAR, TRASLADAR	19
3 GEOCODIFICACIÓN DE DOMICILIOS	21
4 GEOCODIFICACIÓN EN BASE A DATOS ALFANUMÉRICOS.....	24

Usted es libre de compartir - copiar, distribuir, ejecutar y comunicar públicamente y de hacer obras derivadas de este documento

Este documento es una obra compartida bajo la licencia Creative Commons.

Atribución-NoComercial-CompartirIgual 2.5 Argentina (CC BY-NC-SA 2.5)



Atribución — Debe reconocer los créditos de la obra de la manera especificada por el autor o el licenciante (pero no de una manera que sugiera que tiene su apoyo o que apoyan el uso que hace de su obra).



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.

Aviso: Al reutilizar o distribuir la obra, tiene que dejar muy en claro los términos de la licencia de esta obra. La mejor forma de hacerlo es enlazar a la siguiente página:

<http://creativecommons.org/licenses/by-nc-sa/2.5/ar/>

Programa Nacional Mapa Educativo
Ministerio de Educación
República Argentina

Teléfono/Fax: 54 11 4129-1408
Correo electrónico: mapaedu_nac@me.gov.ar
www.mapaeducativo.edu.ar



