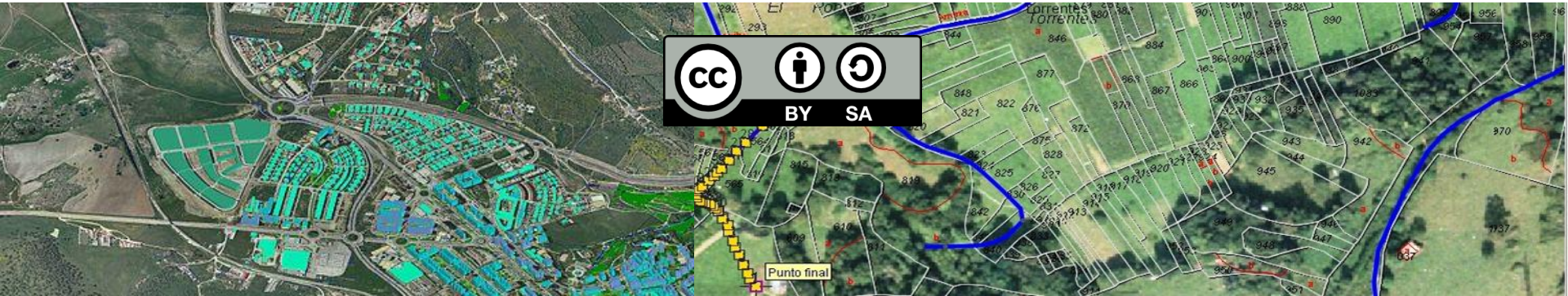
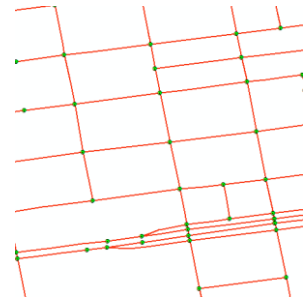


ANÁLISIS DE REDES: RUTA MÁS CORTA

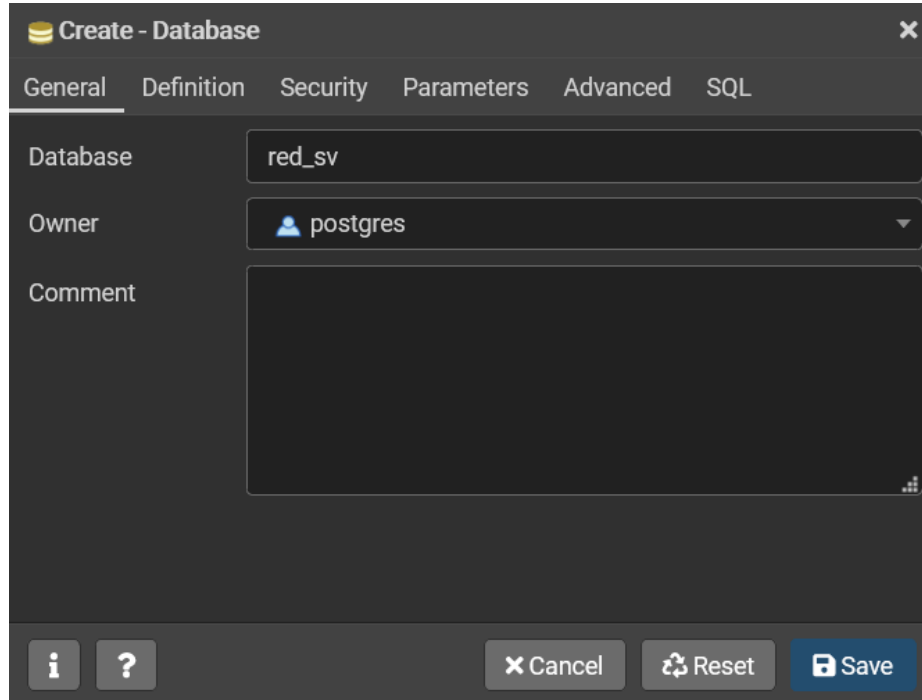


ANÁLISIS DE REDES CON POSTGIS: CREACIÓN DE LA RED

1. Creación de base de datos
2. Habilitación de capacidad para análisis de redes con extensión pgRouting
3. Preparación de datos: corte de red vial y reproyección de SRC
4. Insertar SRC proyectado a la nueva base de datos (Lambert)
5. Importar datos de red vial con: nombre, sentidos y tipo calle.
6. Crear topología de red (se añaden fuente y destino a la red vial)
7. Agregar **distancia de tramo** y **costo** a la red
8. Realizar algoritmos de análisis de redes de transporte



PASOS 1 Y 2: CREACIÓN DE BD CON EXTENSIÓN ESPACIAL + PGROUTING



Create - Database

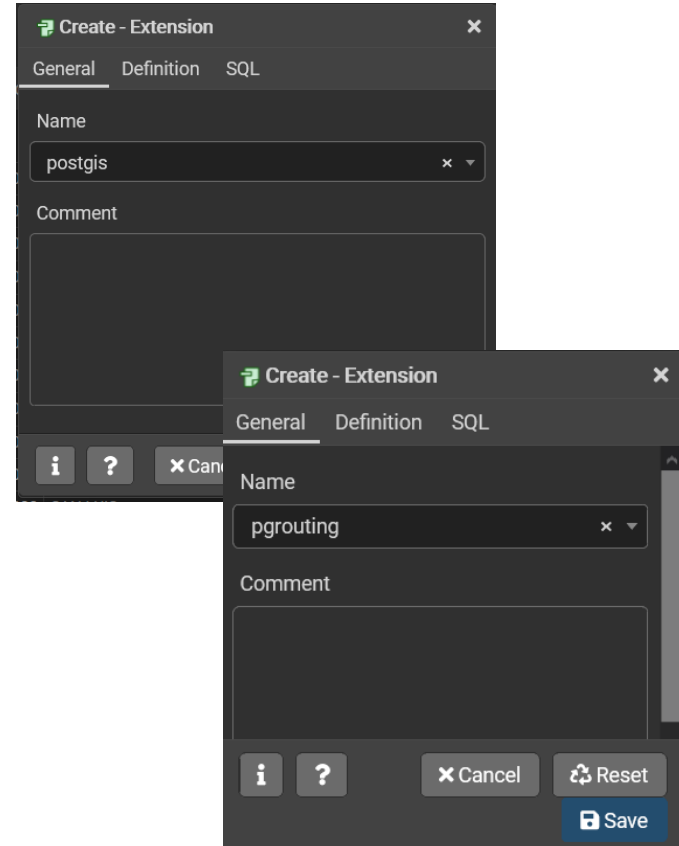
General Definition Security Parameters Advanced SQL

Database: red_sv

Owner: postgres

Comment:

Buttons: i, ?, Cancel, Reset, Save



Create - Extension

General Definition SQL

Name: postgis

Comment:

Buttons: i, ?, Cancel, Reset, Save

Create - Extension

General Definition SQL

Name: pgrouting

Comment:

Buttons: i, ?, Cancel, Reset, Save

4. INSERCIÓN DE SRC LAMBERT EN NUEVA BASE



red_sv/postgres@PostgreSQL 13 ▾

Query Editor

Query History

```
1 INSERT into spatial_ref_sys (srid, auth_name, auth_srid, proj4text, srtext)
2 values ( 100000, 'EPSG', 100000, '+proj=lcc +lat_1=13.31666666 +lat_2=14.25 +lat_0=13.783333333333333
```

5. CONEXIÓN A POSTGIS DESDE QGIS PARA IMPORTAR CAPA

Crear una nueva conexión a PostGIS

La conexión a red_sv tuvo éxito.

Información sobre la conexión

Nombre: red_sv

Servicio:

Anfitrión: localhost

Puerto: 5432

Base de datos: red_sv

Modo SSL: deshabilitar

Autenticación

Configuraciones Básica

Nombre de usuario: postgres ☐ Guardar

Contraseña: ☐ Guardar

Warning: credentials stored as plain text in archivo de proyecto.

Convertir a configuración

Probar conexión

IMPORTAR CAPA DE RED VIAL DE MUNICIPIO

Importar capa vectorial

Entrada: redvial_turin_lcc

☐ Importar sólo objetos espaciales seleccionados Actualizar opciones

Tabla de salida

Esquema: public

Tabla: redvial_turin

Opciones

☒ Clave primaria: id

☒ Columna de geometría: geom

☒ SRID de origen: Default CRS: EPSG:4326 - WGS 1984

☒ SRID de destino: Default CRS: EPSG:4326 - WGS 1984

☒ Codificación: UTF-8

☐ Sustituir la tabla de destino (si existe)

☐ No estimular a multi parte

☐ Pasar nombres de campos a minúsculas

☒ Crear índice espacial

☐ Comentario:

Aceptar Cancelar

Proveedores

Info Tabla Vista preliminar

- GeoPackage
- Oracle Spatial
- PostGIS
 - catastro
 - cnr
 - cnr_lcc
 - nyc
 - pgrouting
 - red_sv
 - public
 - geography_columns
 - geometry_columns
 - redvial_turin
 - spatial_ref_sys
 - rutas
 - world
- Spatialite
- Capa virtual

ACTUALIZAR EL SRC A LA RED VIAL

Base de datos Esquema Tabla

Importar capa/archivo... Exportar a archivo...

Proveedores

- GeoPackage
- Oracle Spatial
- PostGIS
 - catastro
 - cnr
 - cnr_lcc
 - nyc
 - pgrouting
 - red_sv
 - public
 - geography_columns
 - geometry_columns
 - redvial_turin
 - spatial_ref_sys
 - rutas
 - world
- Spatialite
- Capa virtual

Info Tabla Vista preliminar Consulta (red_sv) X

Consulta guardada Nombre Guardar Borrar Cargar archivo Guardar como archivo

```
1 select UpdateGeometrySRID('public', 'redvial_turin', 'geom', 100000) ;
```

Ejecutar 1 filas, 0.245 segundos Crear una vista Limpiar Historial de consultas

updategeometrysrid

| | |
|---|--|
| 1 | public.redvial_turin.geom SRID changed to 100000 |
|---|--|

☐ Cargar como capa nueva

Cancelar

6. CREAR TOPOLOGÍA

```
ALTER TABLE redvial_turin ADD COLUMN source integer;  
ALTER TABLE redvial_turin ADD COLUMN target integer;  
SELECT pgr_CreateTopology('redvial_turin', 0.001, 'geom', 'id');
```

The screenshot shows a PostgreSQL query editor interface. At the top, there are tabs for 'Info', 'Tabla', 'Vista preliminar', and a query icon. Below these is a toolbar with buttons for 'Consulta guardada', 'Nombre', 'Guardar', 'Borrar', 'Cargar archivo', and 'Guardar como archivo'. The main area contains three lines of SQL code: `1 ALTER TABLE redvial_turin ADD COLUMN source integer;`, `2 ALTER TABLE redvial_turin ADD COLUMN target integer;`, and `3 SELECT pgr_CreateTopology('redvial_turin', 0.001, 'geom', 'id');`. Below the code is a status bar showing 'Ejecutar', '1 filas, 0.936 segundos', 'Crear una vista', 'Limpiar', and 'Historial de consultas'. At the bottom, there is a result pane showing the output of the first command: `ogr_createtopolog;` and a row with the value '1 OK'.

Info Tabla Vista preliminar Consulta (red_sv) X

Consulta guardada Nombre Guardar Borrar Cargar archivo Guardar como archivo

```
1 ALTER TABLE redvial_turin ADD COLUMN source integer;  
2 ALTER TABLE redvial_turin ADD COLUMN target integer;  
3 SELECT pgr_CreateTopology('redvial_turin', 0.001, 'geom', 'id');
```

Ejecutar 1 filas, 0.936 segundos Crear una vista Limpiar Historial de consultas

ogr_createtopolog;

1 OK

CREAR TOPOLOGÍA

Administrador de BBDD

Base de datos Esquema Tabla

Importar capa/archivo... Exportar a archivo...

Proveedores

- GeoPackage
- Oracle Spatial
- PostGIS
 - catastro
 - cnr
 - cnr_lcc
 - nyc
 - red_sv
 - public
 - geography...
 - geometry...
 - red_turin
 - red_turin_v...
 - redvial_turin

| | Info | Tabla | Vista preliminar | | | | | |
|---|------|---------|------------------|--------------|--------|-------------|--------|--------|
| | | highway | z_order | other_tags | oneway | tipo_via | source | target |
| 1 | | y | 4 | "lanes"=>"2" | 00 | terciaria | 110 | 187 |
| 2 | | ential | 3 | "lanes"=>"2" | 00 | residencial | 1 | 2 |
| 3 | | ential | 3 | NULL | 00 | residencial | 3 | 4 |
| 4 | | ential | 3 | NULL | 00 | residencial | 5 | 6 |
| 5 | | ential | 3 | NULL | 00 | residencial | 7 | 8 |
| 6 | | ential | 3 | NULL | 00 | residencial | 8 | 5 |
| 7 | | ential | 3 | NULL | 00 | residencial | 5 | 9 |

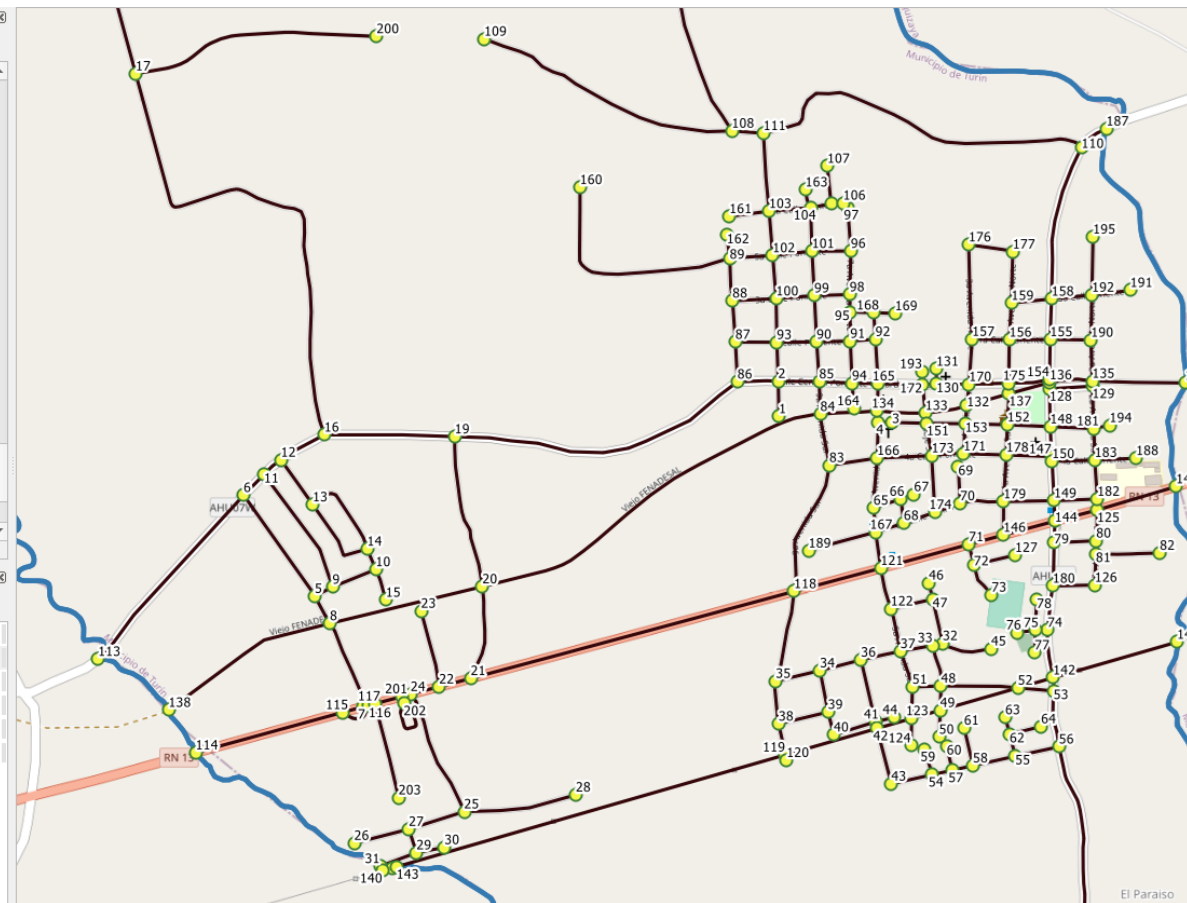
CREAR TOPOLOGÍA

Navegador (2)


- Inicio
- C:\
- D:\
- GeoPackage
- Spatialite
- PostGIS
 - catastro
 - cnr
 - cnr_lcc
 - nyc
 - red_sv
 - public
 - world
- MSSQL
- Oracle
- DB2
- WMS/WMTS
- XYZ Tiles
 - Cyclostrm
 - Mapbox


Capas

- ☐ turin_qneat3
- ☒ red turin vertices pgr
- ☒ red_turin
- ☐ Ruta más corta
- ☒ municipios
- ☒ OpenStreetMap



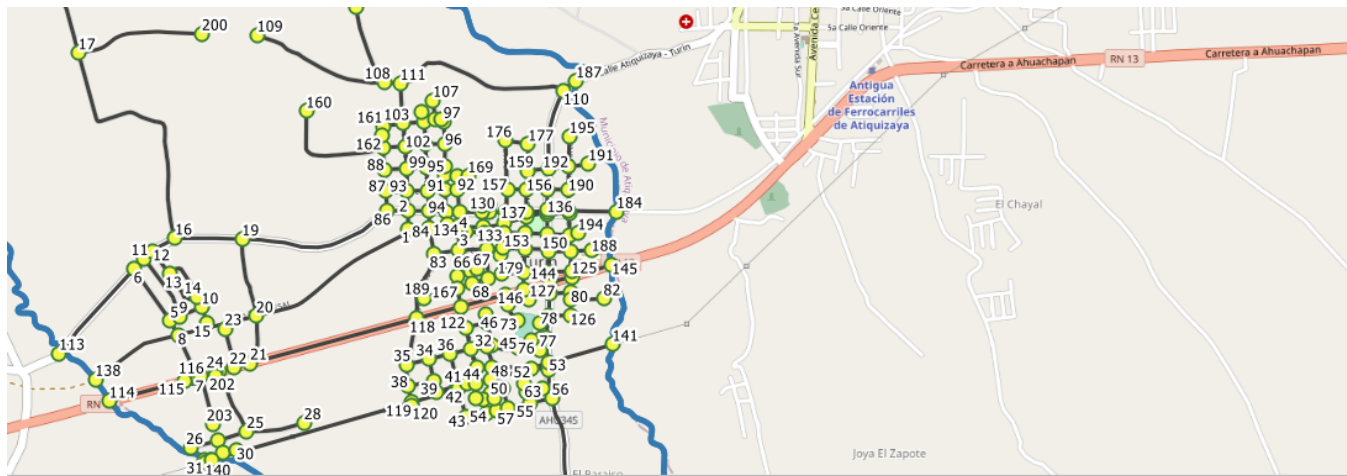
7. AGREGAR LONGITUD DE TRAMO Y COSTO

Info Tabla Vista preliminar  Consulta (red_sv) X

 Consulta guardada ▼ Nombre Guardar Borrar Cargar archivo Guardar como archivo

```
1 --select updategeometrysrid('public','red_turin','geom',100000);
2 -- agrega atributo para longitud de tramo
3 ALTER TABLE red_turin ADD column length double precision;
4 UPDATE red_turin SET length = st_length(geom)/1000;
5 -- agrega atributo para el costo de recorrer el tramo
6 ALTER TABLE red_turin ADD column cost double precision;
7 UPDATE red_turin SET cost = length/50*3600;
8
```

AGREGAR LONGITUD DE TRAMO Y COSTO



red_turin :: Objetos totales: 274, Filtrados: 274, Seleccionados: 0

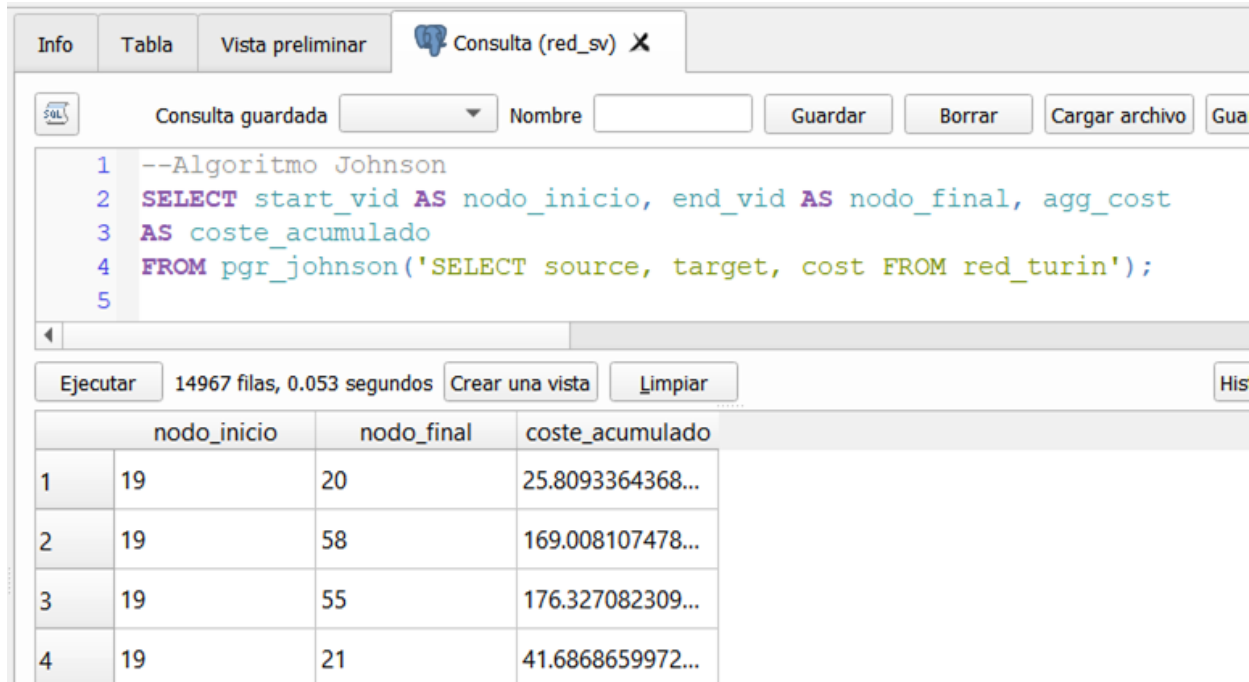


| way | z_order | other_tags | oneway | tipo_via | source | target | length | cost |
|-----|---------|------------|--------|-------------|--------|--------|------------------|------------------|
| 1 | al | 3 NULL | 00 | residencial | 33 | 48 | 0.09721065208... | 6.99916695000... |
| 2 | al | 3 NULL | 00 | residencial | 48 | 49 | 0.05662612365... | 4.07708090326... |
| 3 | al | 3 NULL | 00 | residencial | 46 | 47 | 0.03908192269... | 2.81389843396... |
| 4 | al | 3 NULL | 00 | residencial | 47 | 32 | 0.10561681814... | 7.60441090641... |
| 5 | al | 3 NULL | 00 | residencial | 58 | 55 | 0.10165242821... | 7.31897483134... |
| 6 | al | 3 NULL | 00 | residencial | 59 | 54 | 0.06209338253... | 4.47072354270... |
| 7 | al | 3 NULL | 00 | residencial | 54 | 57 | 0.04770429990... | 3.43470959348... |

9. APLICACIONES

FUNCIÓN PGR_JOHNSON

La función `pgr_johnson` nos devuelve un listado con todos los costes acumulados para cada par de nodos en nuestra red, siempre desde la ruta más corta entre ellos. El algoritmo de Johnson es una forma de encontrar los caminos más cortos entre todos los pares de vértices en una network.



The screenshot shows a database management system interface with a query editor and a results table. The query editor contains the following SQL code:

```
1 --Algoritmo Johnson
2 SELECT start_vid AS nodo_inicio, end_vid AS nodo_final, agg_cost
3 AS coste_acumulado
4 FROM pgr_johnson('SELECT source, target, cost FROM red_turin');
5
```

Below the query editor, the execution status is displayed: "Ejecutar 14967 filas, 0.053 segundos". There are buttons for "Crear una vista", "Limpiar", and "His".

The results table displays the output of the query, showing the start node, end node, and the accumulated cost for the shortest path between them.

| | nodo_inicio | nodo_final | coste_acumulado |
|---|-------------|------------|------------------|
| 1 | 19 | 20 | 25.8093364368... |
| 2 | 19 | 58 | 169.008107478... |
| 3 | 19 | 55 | 176.327082309... |
| 4 | 19 | 21 | 41.6868659972... |

FUNCIÓN PGR_DIJKSTRA

La función pgr_dijkstra

Consulta guardada Nombre

```
5 -- Algoritmo Dijkstra
6 SELECT seq, node AS nodo_origen, edge AS nodo_destino, cost AS
7   coste, agg_cost AS coste_acumulado
8 -FROM pgr_dijkstra('SELECT id::integer, source::integer,
9   target::integer, cost::double precision FROM red_turin',
10  189, 160, false);
11
```

Ejecutar 13 filas, 0.034 segundos

| | seq | nodo_origen | nodo_destino | coste | coste_acumulado |
|----|-----|-------------|--------------|------------------|------------------|
| 1 | 1 | 189 | 5 | 11.7377075914... | 0.0 |
| 2 | 2 | 167 | 90 | 4.06679739971... | 11.7377075914... |
| 3 | 3 | 65 | 90 | 8.37824017212... | 15.8045049911... |
| 4 | 4 | 166 | 10 | 8.13926936271... | 24.1827451632... |
| 5 | 5 | 83 | 50 | 8.70574246496... | 32.3220145259... |
| 6 | 6 | 84 | 50 | 5.46425544198... | 41.0277569909... |
| 7 | 7 | 85 | 87 | 6.65400925111... | 46.4920124329... |
| 8 | 8 | 90 | 53 | 6.71592237595... | 53.1460216840... |
| 9 | 9 | 93 | 53 | 6.9966121419244 | 59.8619440599... |
| 10 | 10 | 87 | 52 | 6.94134653748... | 66.858556201894 |
| 11 | 11 | 88 | 52 | 7.17120414062... | 73.7999027393... |
| 12 | 12 | 89 | 83 | 38.7468292409... | 80.971106880012 |
| 13 | 13 | 160 | -1 | 0.0 | 119.717936120... |

PLUGIN: PGROUTING LAYER

pgRouting Layer

Database: red_sv

pgr_ pgr_dijkstra

Edges SQL

Edge table: red_turin

Schema: public

Geometry: geom

☐ BBOX

Columns

Id: gid

Source: source

Target: target

Cost: cost

Reverse Cost: reverse_cost ☐

Execute

pgRouting Layer

Database: red_sv

pgr_ pgr_dijkstra

Arguments

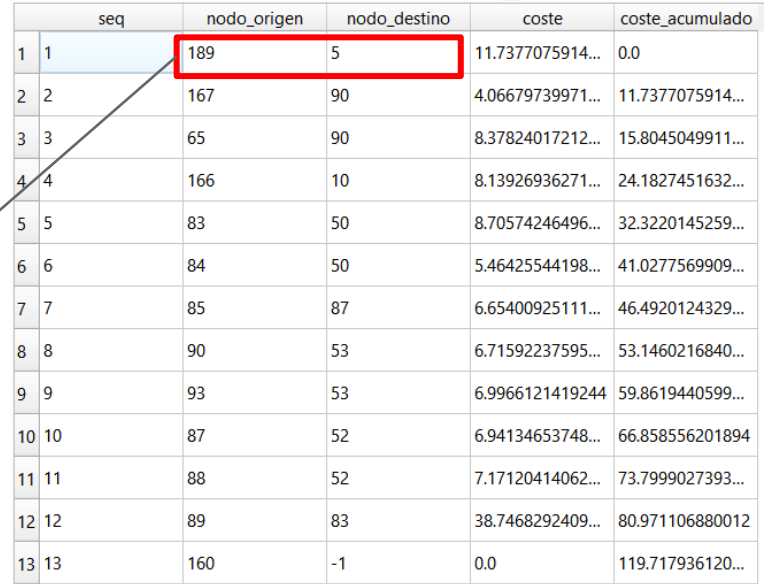
Edges SQL

From Vids: 189

To Vids: 160

☐ directed

Execute



HERRAMIENTAS PROCESAMIENTO: RUTA MÁS CORTA (P-P)

Ruta más corta (punto a punto)

Parámetros

Registro

Capa vectorial que representa la red

V red_turin [USER:100005]

...

☐ Objetos seleccionados solamente

Tipo de ruta a calcular

Más corto

Punto de inicio

-9993291.941598,1569811.875482 [EPSG:3857]

Punto final

-9993844.481303,1570687.986825 [EPSG:3857]

▼ Parámetros avanzados

Campo de sentido [opcional]

abc oneway

Valor para sentido de avance [opcional]

FT

Valor para dirección hacia atrás [opcional]

TF

Valor para ambas direcciones [opcional]

00

Dirección predeterminada

Ambas direcciones

Campo de velocidad [opcional]

Velocidad predeterminada (km/h)

50.000000

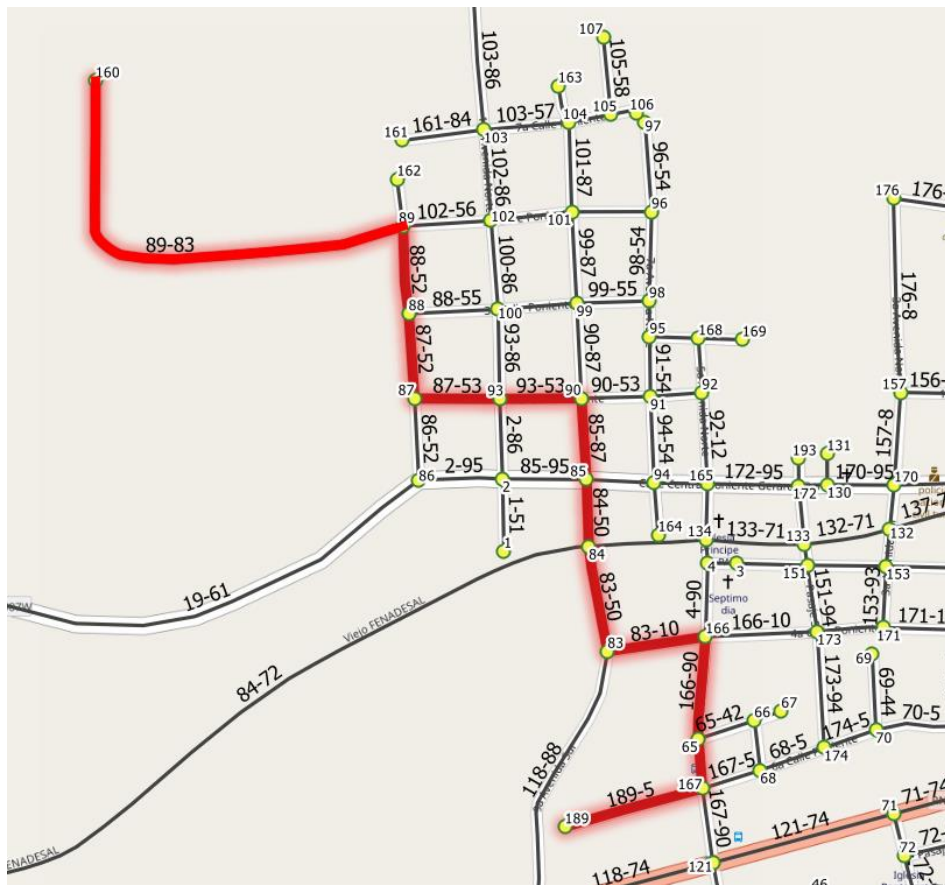
Tolerancia de topología

0.001000

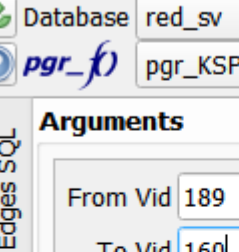
metros

Ruta más corta (punto a punto)

Este algoritmo calcula la ruta óptima (la más corta o la más rápida) entre los puntos inicial y final indicados.



PGROUTING: PGR_KSP, RUTAS ALTERNAS



pgRouting Layer

Database: red_sv

Schema: pgr_KSP

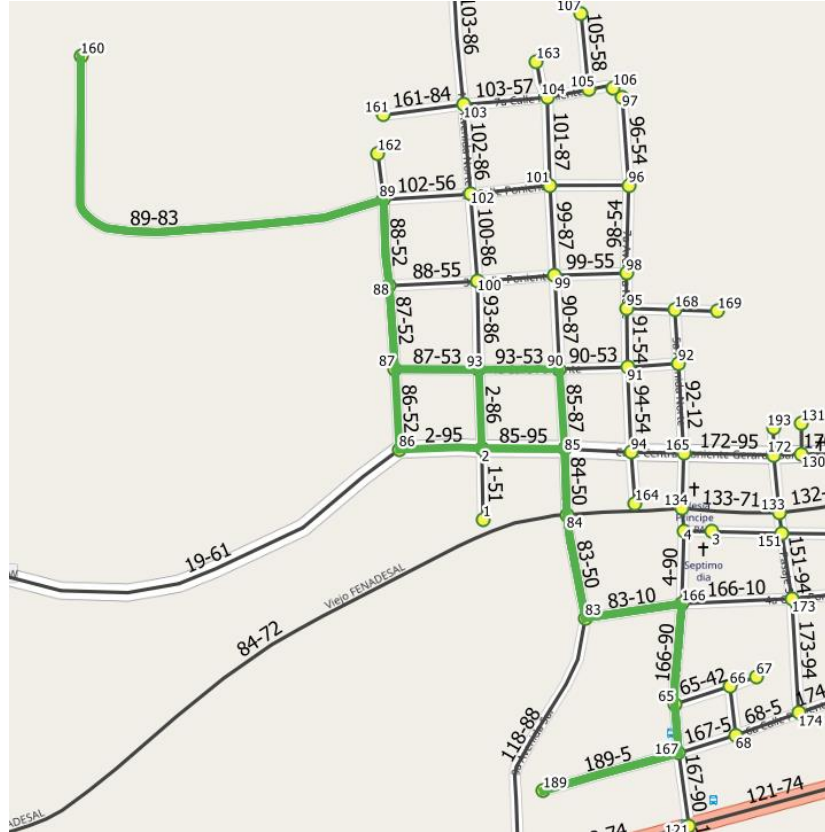
Arguments

From Vid: 189

To Vid: 160

K: 3

☐ directed ☐ heap_paths



RUTA MÁS CORTA (CAPA A PUNTO)

