

INSTITUTO TECNOLÓGICO DE BUENOS AIRES – ITBA

ESCUELA DE INGENIERÍA Y GESTIÓN

INVENTARIO VIAL GEORREFERENCIADO

AUTOR: Arequipa Patricia Noemi (Leg. N° 105045)

TUTOR: Aizemberg Ariel

**TRABAJO FINAL PRESENTADO PARA LA OBTENCIÓN DEL TÍTULO DE ESPECIALISTA EN CIENCIA
DE DATOS**

BUENOS AIRES

SEGUNDO CUATRIMESTRE, 2020

CONTENIDO

1- INTRODUCCIÓN.....	1
2- ESTADO DE LA CUESTION	1
3- DEFINICION DEL PROBLEMA.....	2
4- JUSTIFICACION DEL ESTUDIO	3
5- ALCANCES DEL TRABAJO Y LIMITACIONES	3
6- HIPOTESIS.....	4
7- OBJETIVOS	4
Objetivo General	4
Objetivos Específicos.....	4
8- METODOLOGIA	4
Análisis del data set.....	5
Preparación del data set.....	6
Modelado	7
Evaluación	10
9- CONCLUSION	16
10- REFERENCIA BIBLIOGRAFICA	17

1- INTRODUCCIÓN

La Dirección Nacional de Vialidad tiene el objetivo de construir, mantener y conservar la Red Vial Nacional para garantizar una buena conectividad de caminos mejorando así el desarrollo económico del país. Para una gestión vial adecuada, el organismo cuenta con un Inventario Vial, que consiste en una base de datos sin georreferenciar, que describe físicamente toda la Red Vial Nacional, el mismo tiene carácter oficial y está disponible para uso de todos los trabajadores pertenecientes a Vialidad como así también para todo público que precise esta información.

Cuando una persona desee situarse o conocer algún punto de ubicación generalmente accede a una herramienta de búsqueda de ubicaciones (google maps) que le permitirá geolocalizar la dirección de interés, donde podrá saber cómo llegar, que lugares se encuentra cercano y cuánto tiempo tardará en llegar al mismo, de esta manera le resulta fácil orientarse en tiempo y espacio.

Si se aplica este criterio al inventario vial, ayudara a sus usuarios a situarse geoespacialmente en la red vial, localizando tramos, objetos puntuales y/o lugares de interés. Es por ello que en este trabajo se pretende georreferenciar el inventario, agregando al mismo las coordenadas geográficas, para ello se utilizara las herramientas informáticas adecuadas, que permitan trabajar mediante funciones, con el almacenamiento y gestión de datos espaciales.

Los datos set a utilizar, tienen un periodo de actualización anual y están subidos en la web oficial de Vialidad Nacional (<https://www.argentina.gob.ar/obras-publicas/vialidad-nacional>), como así también se pueden solicitar vía mail. El inventario se encuentra en formato xls (año 2020) mientras que las geometrías georreferenciadas se encuentran en formato shp o kml (año 2019). Finalmente, los softwares a utilizar PostgreSQL.12 y QGis 3.16 son de versión libre.

2- ESTADO DE LA CUESTION

Las Rutas Nacionales son administradas por la Dirección Nacional de Vialidad (DNV), donde las mismas se identifican mediante códigos y algunas características que la definen, todo esto se encuentra registrado en una base de datos denominado Inventario Vial, cuyos atributos almacenados son tanto descriptivos como numéricos.

El objetivo del Inventario Vial *"... es proporcionar una clara visión del patrimonio vial y proveer los datos reales del estado actual de la Red Nacional de Caminos"* [1], es por ello que esta base debe mantenerse actualizada y disponible para uso tanto de usuarios viales y externos.

La información disponible acerca de cómo realizar inventarios georreferenciados, hacen referencia a trabajos de tesis que abordan este tema indicando la importancia de geolocalizar puntos característicos [2], y la manera de llevar a cabo estos tipos de proyectos., es identificando y relevando puntos fijos con un dispositivo GPS, de esta manera se va generando una base de datos georreferenciada que luego permitirá ir graficando el inventario, esta metodología sería la óptima e ideal.

En Vialidad Nacional se trabaja con un relevamiento vial donde se identifica a lo largo de la ruta, objetos fijos mediante distancias de longitud entre puntos, pero sin georreferenciar. Una idea para geolocalizar estos datos, es utilizando la ingeniería inversa, donde se pueda aproximar puntos a líneas de rutas georreferenciadas, por lo cual se debe analizar y buscar la metodología apropiada para lograr esta vinculación.

Entre las herramientas más usadas, para realizar consultas y vincular base de datos con coordenadas georreferenciadas nos encontramos con PostgreSQL/PostGIS. Existen muchas investigaciones [3] que analizan diferentes herramientas, intentando buscar nuevos algoritmos y comparar los tiempos de procesamiento y consulta, para luego elegir la herramienta más eficaz que permita realizar el procesamiento en el menor tiempo posible, lo cual no implica que una herramienta sea mejor o peor, todas resuelven el problema, pero en diferentes tiempos.

3- DEFINICION DEL PROBLEMA

Actualmente la Red Vial Nacional consta de un total de 40.000 km, de los cuales el 90% se encuentra pavimentado. A fin de gestionar obras, proyectos e intervenciones sobre las Rutas Nacionales, siempre se hace referencia al Inventario Vial que es el encargado de llevar un registro de las Rutas Nacionales para la DNV, y para identificar algunas características georreferenciadas de las rutas, Vialidad también creó un visor denominado SigVial. Este visor solo permite visualizar las rutas sin la posibilidad de realizar modificaciones, seleccionar tramos determinados, ni agregar características.

4- JUSTIFICACION DEL ESTUDIO

A la hora de intervenir un tramo de Ruta Nacional, se usa las tablas formato xls del Inventario Vial, para identificar las características del mismo, luego para orientarse se utiliza el visor (SigVial) para geolocalizar el tramo en cuestión a intervenir y finalmente una vez que se localiza el tramo, recién se puede continuar con el trabajo y/o investigación solicitado, pero hasta ahora solo se logró ubicar el tramo de ruta geoespacialmente, si se requiere realizar un mapa específico, se debe bajar la capa rutas del visor, se vuelve a identificar el tramo nuevamente y se debe agregar a la misma las características que se desee mostrar para poder mapear el tramo en estudio, todo esto genera más horas de trabajo.

Esta situación puede simplificarse si se cuenta con un inventario georreferenciado, que ayudaría a sus usuarios a ubicarse, orientarse y al tener coordenadas geográficas se podrá mapear, todo en un solo paso, minimizando horas de labor. Es por ello que en este trabajo se intentara obtener las coordenadas geográficas, si bien no contamos con dispositivo GPS para relevar todos los puntos, contamos con herramientas informáticas que permiten aproximar puntos a líneas georreferenciadas.

5- ALCANCES DEL TRABAJO Y LIMITACIONES

Para el desarrollo de este estudio se trabajó con data set públicos pertenecientes a la Dirección Nacional de Vialidad, como se mencionó anteriormente la red vial cuenta con 40.000 km por lo cual se trabajará con un tramo de ruta, hasta encontrar el algoritmo adecuado de aproximación, para luego replicarlo a las demás rutas nacionales.

Las metodologías a utilizar para georreferenciar el inventario, nos llevará a obtener aproximaciones de puntos a través de distancias, con lo cual se podrá tener algunas diferencias al momento de mapear. Lo ideal y preciso es contar con dispositivo GPS e ir ubicando los puntos, pero al no contar con los mismos, las herramientas informáticas permitirán obtener un primer acercamiento al inventario georreferenciado.

Una vez obtenidas las coordenadas geográficas se sumará al inventario actual como un nuevo atributo, de manera que se distribuya la información de forma oficial, para todo usuario que precise de esta información. Esto incentivara a generar capas georreferenciadas con diferentes temáticas que, al compartirla a futuro, se vincularían entre sí, permitiendo realizar análisis a nivel global.

6- HIPOTESIS

El inventario vial de la DNV describe Rutas Nacionales en Argentina, mediante distancias y descripción de objetos fijos que las rodean, intentando representar la realidad, A su vez contamos con líneas georreferenciadas que describen las mismas Rutas Nacionales, por lo cual es posible vincular ambas ya que las mismas intentan definir lo mismo, pero con diferentes formatos.

7- OBJETIVOS

Objetivo General

Generar un algoritmo práctico que permita vincular y ajustar lo mejor posible los puntos de Inventario a puntos georreferenciados, obteniendo así las coordenadas geográficas de cada punto que describe a las rutas, esto permitirá mapear el Inventario vial y compartir el mismo de manera oficial.

Objetivos Específicos

- Análisis y selección de las bases de datos a utilizar.
- Filtrado y transformación del data set, a fin de encontrar los atributos a vincular
- Búsqueda de herramientas informáticas libres como PostGIS que permitirá vincular datos espaciales.
- Encontrar el mejor algoritmo que permita la mejor aproximación a la realidad,
- Visualización de las coordenadas obtenidas junto a sus atributos.

8- METODOLOGIA

En este trabajo se pretende obtener puntos georreferenciados del inventario vial, para ello se vinculará el data set xls, con otro de formato shp. Las herramientas informáticas a utilizar es PostGis, que permite hacer consulta de bases espaciales y no espaciales, proporcionando una forma sistemática de crear, recuperar, actualizar y administrar datos, esta herramienta permitirá conectar ambas bases, utilizando la mejor función que permita aproximar los puntos sin georreferenciar al data set geométrico.

La herramienta QGis permitirá verificar y visualizar geoespacialmente los resultados que se va obteniendo generando de esta manera la capa shp del inventario vial, a fin de poder mapear los puntos deseados.

Análisis del data set

- Rutas.SHP, data set obtenido de la página oficial de la DNV (<https://www.argentina.gob.ar/obras-publicas/vialidad-nacional/sig-vial>), consiste en una capa georeferenciada del tipo MULTILINESTRING, con 272 registro referidos a las rutas nacionales, diferenciadas por su numeración con origen y fin sin cortes (rutas dibujadas con longitud total), en sentido ascendente y descendente.

Analizando el **SRID** (*Spatial Reference System Identifier*), vemos que el Sistema de Coordenadas es WGS 84 con unidades en grados, como se muestra a continuación.

```
GEOGCS["WGS 84", DATUM["World Geodetic System 1984", SPHEROID["WGS 84", 6378137.0, 298.257223563, AUTHORITY["EPSG","7030"]], AUTHORITY["EPSG","6326"]], PRIMEM["Greenwich", 0.0, AUTHORITY["EPSG","8901"]], UNIT["degree", 0.017453292519943295], AXIS["Geodetic longitude", EAST], AXIS["Geodetic latitude", NORTH], AUTHORITY["EPSG","4326"]]
```

- Inventario2020.XLSX, data set en formato Excel que muestra una tabla con 206934 registros de datos puntuales sin georeferenciar, que describen las Rutas Nacionales de Argentina. En la *Figura 01- Inventario vial 2020*, se observa cómo se obtiene la base original.

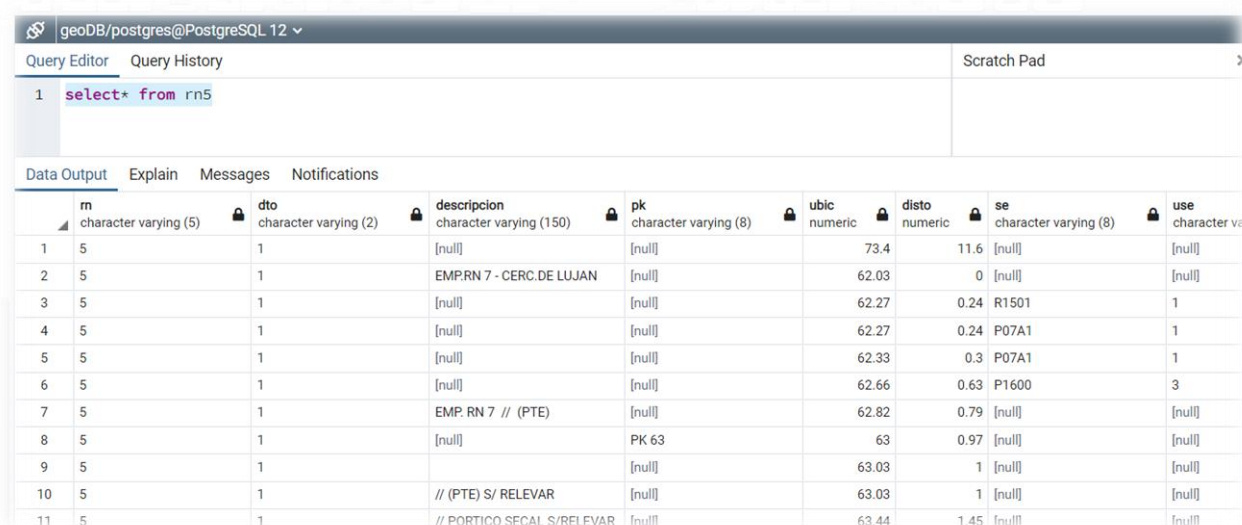
Ruta		Descripción		Postes	Unic.	Dist. Origen	Señal	Estruc-tura	Intersección	Camino	Tramo	Tramo	
Ruta		Descripción		Kilométricos								Tramo	
0005 01	EMP RN 7 - CERC DE LUJAN			62.03	0.00				02	0007	1 A B	2011	8 8 S 3.0 A1 7.3 2 S 3.0 S 3.0 A3 7.3 2 S 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				62.27	0.24	R1501	1					2011	8 8 S 3.0 A1 7.3 2 S 3.0 S 3.0 A3 7.3 2 S 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				62.27	0.24	P07A1	1					2011	8 8 S 3.0 A1 7.3 2 S 3.0 S 3.0 A3 7.3 2 S 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				62.33	0.30	P07A1	1					2011	8 8 S 3.0 A1 7.3 2 S 3.0 S 3.0 A3 7.3 2 S 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				62.66	0.63	P1600	3					2011	8 8 S 3.0 A1 7.3 2 S 3.0 S 3.0 A3 7.3 2 S 3.0 4 R L T AP LUJAN - MERCEDES
0005 01	EMP RN 7 // (PTE)			62.82	0.79			03 A	OO	0007	1 A	2011	8 8 S 3.0 A1 7.3 2 S 3.0 S 3.0 A3 7.3 2 S 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				63.00	0.97							2011	8 8 S 3.0 A1 7.3 2 S 3.0 S 3.0 A3 7.3 2 S 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				63.03	1.00			02 A				2011	8 8 S 3.0 A1 7.3 2 S 3.0 S 3.0 A3 7.3 2 S 3.0 4 R L T AP LUJAN - MERCEDES
0005 01	// (PTE) S/ RELEVAR			63.03	1.00			02 O				2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01	// PORTICO SEÑAL S/RELEVAR			63.44	1.45			17 A				2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				63.65	1.68			37 O				2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				63.86	1.91			14 B				2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01	POSTE S.O.S. (D) (I)			64.04	2.10			38				2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01	PASO PEATONAL // S/N S/RELEVAR			64.24	2.32			20 B				2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				64.24	2.32							2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01	EMP. EX R.N. N° 7 // (B/N) S/RELEVAR			64.89	3.02			09 B	OO	2 A		2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01	F.C.O. SARMIENTO // (B/N) S/RELEVAR			64.93	3.06			10 B				2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01	// (B/N) S/RELEVAR			65.16	3.31			09 B	01	A		2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01	CALLE DR. NEGRI // (B/N) S/RELEVAR			65.31	3.48			09 B	01	A		2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				66.00	4.23							2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01	CALLE N°653 // (B/N) S/RELEVAR			66.30	4.53			09 B	01	A		2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				67.00	5.22							2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01	EMP. R.P. N° 47 // (B/N) S/RELEVAR			67.17	5.39			09 B	11	0047	2 A	2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				67.36	5.58			01 A				2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				67.36	5.58			01 O				2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				67.43	5.65							2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				67.43	5.65							2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 7 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				67.62	5.84			01 A				2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 1 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				67.62	5.84			01 O				2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 1 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01	PASO PEATONAL // (S/N) S/RELEVAR			67.81	6.03			20 B				2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 1 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01				68.00	6.16							2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 1 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01	A* GUTIERREZ // (PTE) S/RELEVAR			68.62	6.79			02 A				2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 1 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES
0005 01	A* GUTIERREZ // (PTE) S/RELEVAR			68.62	6.79			02 O				2018	3.65 3.45 A 3.0 A1 7.3 2 S 3.0 1 7.5 S 3.0 A3 7.3 2 A 3.0 4 R L T AP LUJAN - MERCEDES

Figura 01- Inventario vial 2020

Preparación del data set

De la base original del inventario se consideró una Ruta Nacional, la cual se utilizara como muestra, hasta encontrar el algoritmo deseado para luego replicarlo a las demás Rutas.

Para ello filtramos de la base Excel la RN 05 y la guardamos en extensión CSV, para así poder importar la misma a Postgres con el fin de cruzar la información con los datos georreferenciados. A continuación, se muestra el nuevo data set denominado rn5 en Postgres *Figura 02- Base rn5 Postgres*.



The screenshot shows a PostgreSQL query editor window titled 'geoDB/postgres@PostgreSQL 12'. The 'Query Editor' tab is active, displaying the query: `select* from rn5`. Below the query editor, the 'Data Output' tab shows the results of the query in a table. The table has 11 rows and 10 columns. The columns are: 'rn' (character varying (5)), 'dto' (character varying (2)), 'descripcion' (character varying (150)), 'pk' (character varying (8)), 'ubic' (numeric), 'disto' (numeric), 'se' (character varying (8)), and 'use' (character varying (8)). The data is as follows:

	rn	dto	descripcion	pk	ubic	disto	se	use
1	5	1	[null]	[null]	73.4	11.6	[null]	[null]
2	5	1	EMP.RN 7 - CERC.DE LUJAN	[null]	62.03	0	[null]	[null]
3	5	1	[null]	[null]	62.27	0.24	R1501	1
4	5	1	[null]	[null]	62.27	0.24	P07A1	1
5	5	1	[null]	[null]	62.33	0.3	P07A1	1
6	5	1	[null]	[null]	62.66	0.63	P1600	3
7	5	1	EMP. RN 7 // (PTE)	[null]	62.82	0.79	[null]	[null]
8	5	1	[null]	PK 63	63	0.97	[null]	[null]
9	5	1	[null]	[null]	63.03	1	[null]	[null]
10	5	1	// (PTE) S/ RELEVAR	[null]	63.03	1	[null]	[null]
11	5	1	// PORTICO SECAL S/RELEVAR	[null]	63.44	1.45	[null]	[null]

Figura 02- Base rn5 Postgres

La capa georreferenciada Rutas. Shp, la llevamos al programa Qgis, la misma se puede visualizar en la *Figura 03- Rutas.shp en Qgis*, y tiene un sistema de referencia espacial EPSG 4326 (WGS 84), en la imagen se muestra un gráfico por categoría de rutas en donde se visualiza que cada ruta esta dibujada de inicio a fin, esta misma luego se exporta al Postgis *Figura 04- Rutas en Postgis*.

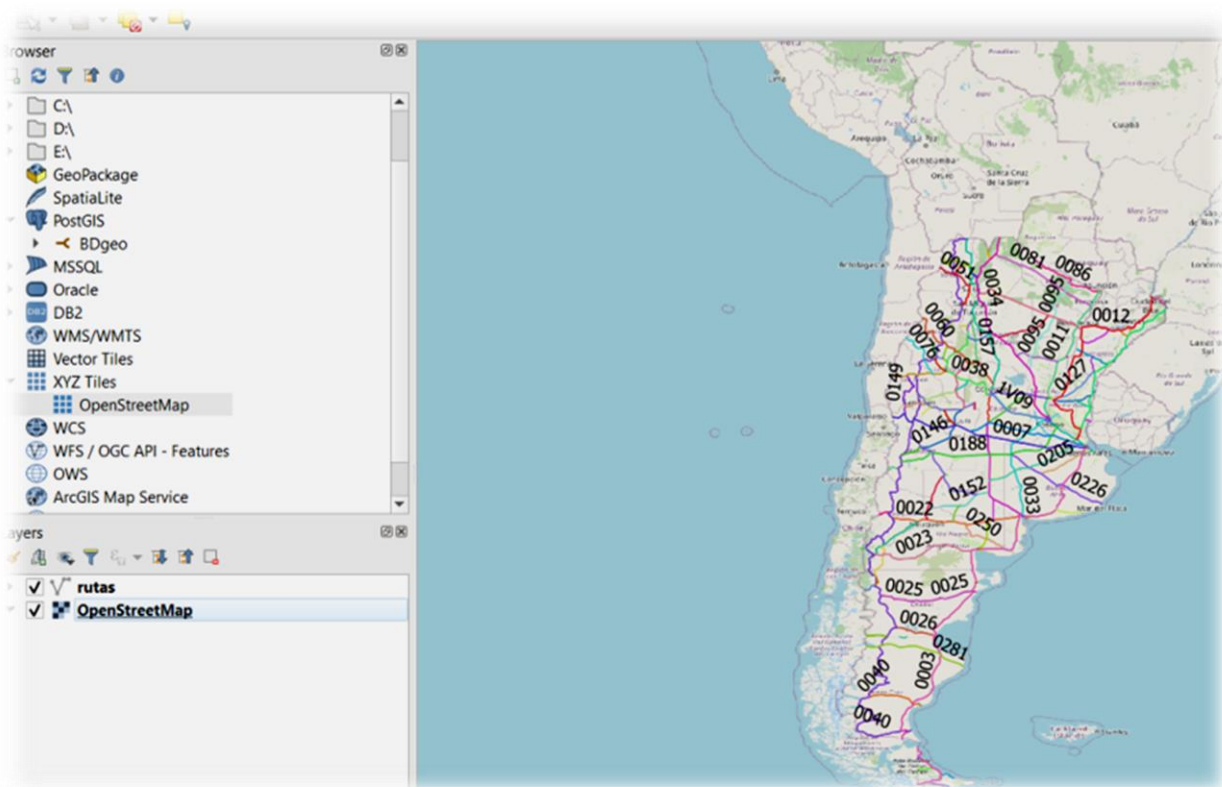


Figura 03- Rutas.shp en Qgis

id	geom	id_ruta	cod_ruta	sentido	distancia	dibujo	progresiva	progresiv0	tipo_calza	ano_ruta	progresiv1	progresiv2
[PK] Integ	geometry	integer	character vary	character vary	numeric	numeric	numeric	numeric	character vary	integer	integer	integer
1	87 0105000020E6100...	1762	0005	A	542.1	543.984	62.03	606.65	P	2018	62030	606650
2	250 0105000020E6100...	1756	0005	D	542.1	543.636	62.03	606.65	P	2018	62030	606650

Figura 04- Rutas en Postgis.

Modelado

Para encontrar el algoritmo indicado se procedió a la búsqueda de funciones relacionada a datos geométricos establecidas para el programa Postgres, de las cuales encontramos `ST_LineInterpolatePoint`(geometría a_linestring ,float8 a_fraction).

Esta función devuelve un punto interpolado a lo largo de una línea. El primer argumento debe ser Linestring, en nuestro caso contamos con Multilinestring, por lo cual deberemos realizar alguna transformación previa. Mientras que el segundo argumento es un float8 entre 0 -1 y representa una fracción de la longitud total de la cadena lineal en la que se debe ubicar el punto.

A continuación, desglosamos los argumentos para verificar si los resultados son los deseados

- Linestring

En nuestro caso el data set SHP cuenta con líneas Multilinestring, por lo cual debemos llevarla a Linestring para poder utilizar la sentencia anterior, para ello utilizamos la siguiente función **ST_LineMerge**(geometría de una cadena múltiple) ; que devuelve un conjunto de LineString formados por Multilinestring, en el caso que no se puede fusionar, se devolverá el Multilinestring original.

La query utilizada es la siguiente:

```
--Tablas auxiliar paso de Multilinestring a Linestring
CREATE MATERIALIZED VIEW
rutline AS
    SELECT cod_ruta,distancia_,
    ST_AsText(ST_LineMerge(geom)) AS geom2
    FROM rutas
    WHERE sentido='A';
```

- Float

Para buscar la distancia de corte realizo una simple regla de tres, considerando la longitud de dibujo al respecto de la longitud real que se considera en el inventario y se calculó la fracción de distancia donde debo colocar cada punto deseado.

```
--Cálculo del punto
SELECT (a.disto / c.distancia_) as total
FROM rn5 a, rutas c
WHERE c.cod_ruta='0005'
```

Con estas transformaciones y verificaciones previas ahora se puede armar el algoritmo para replicarlo a todos los puntos de la RN 05 automáticamente. Seguidamente, se muestra las query utilizadas y la *Figura 04- RN05 georreferenciada-Postgres*, muestra el resultado obtenido donde se puede visualizar que la transformación se realizó con éxito.

```
--Creamos la columna con el pto georeferenciado
SELECT AddGeometryColumn('public','rn5','geom',4326,'POINT',2);
```

```

---Actualizamos la columna geom con coordenadas
UPDATE rn5 SET geom = st_lineinterpolatepoint(g.geom2, (rn5.disto
/ g.distancia_))
FROM rutline g
WHERE g.cod_ruta='0005'

```

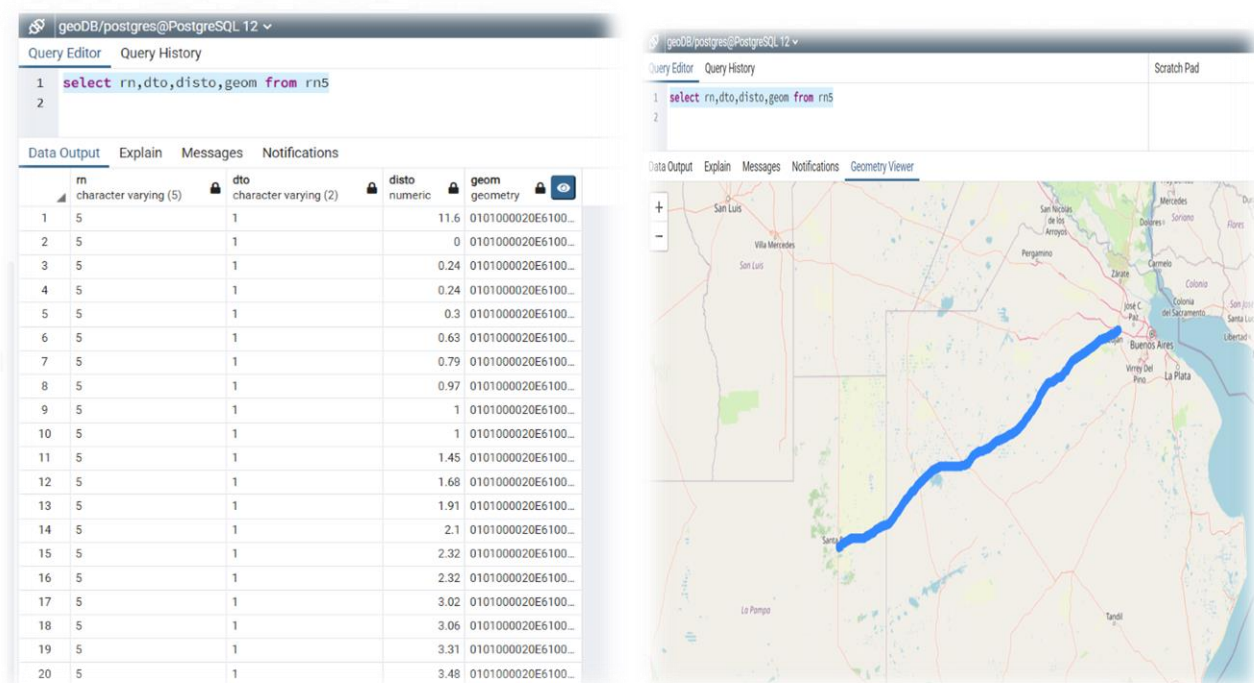


Figura 05- RN05 georreferenciada-Postgres

Al vincular el nuevo data set georreferenciado entre los dos programas Postgres y Qgis, se puede representar la nueva capa en ambos programas, en la *Figura 06- RN05 georreferenciada-Qgis* se muestra lo recién explicado.

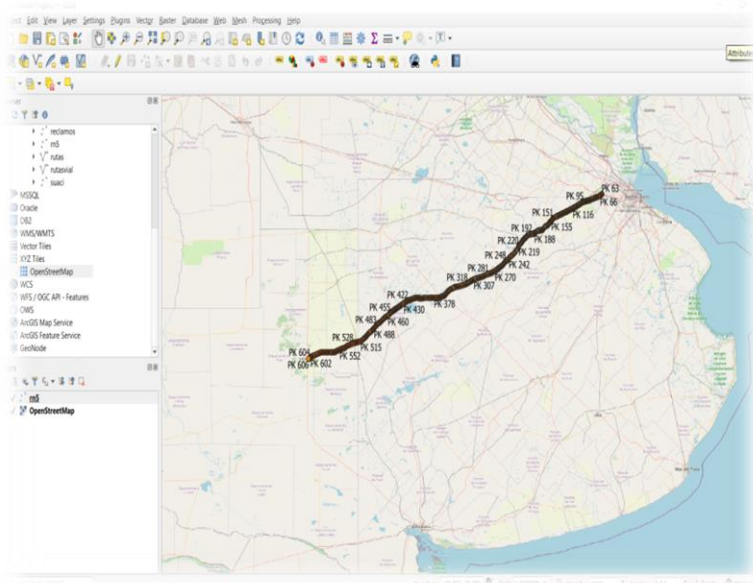


Figura 06- RN05 georreferenciada-Qgis

Evaluación

Una vez encontrado el algoritmo buscado, ahora se procede a replicar el mismo al data set completo del inventario2020, utilizando la función geométrica `ST_LineInterpolatePoint`(geometría a_linestring ,float8 a_fraction).

Esta función no resultó, debido a que al pasar Multilinestring a Linestring, quedaron 10 rutas sin transformar, como se muestra en la *Figura 07- RN Multilinestring, Postgres-Qgis*, donde para visualizar mejor el problema, para ello se exportó de postgres a qgis donde se efectuó un mapa temático por color de ruta, con lo cual se puede concluir que algunas transformaciones no se realizaron por estar dibujadas de manera no continua.

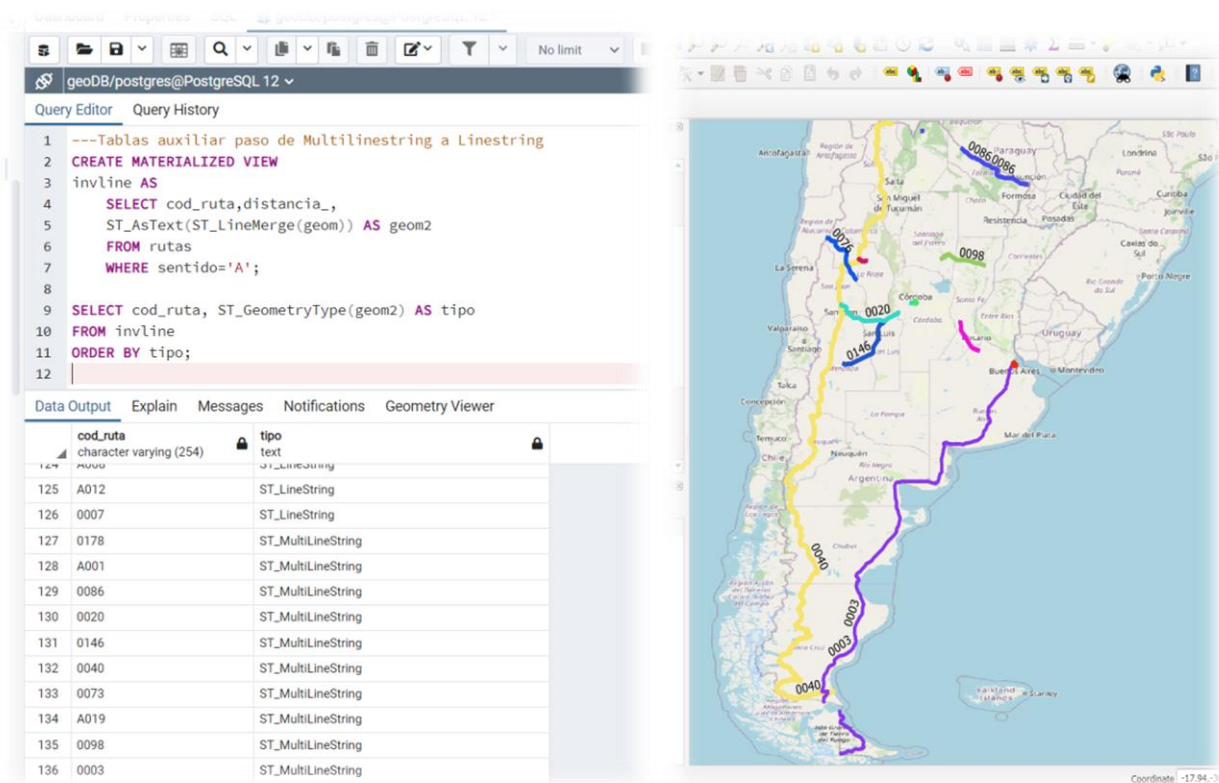
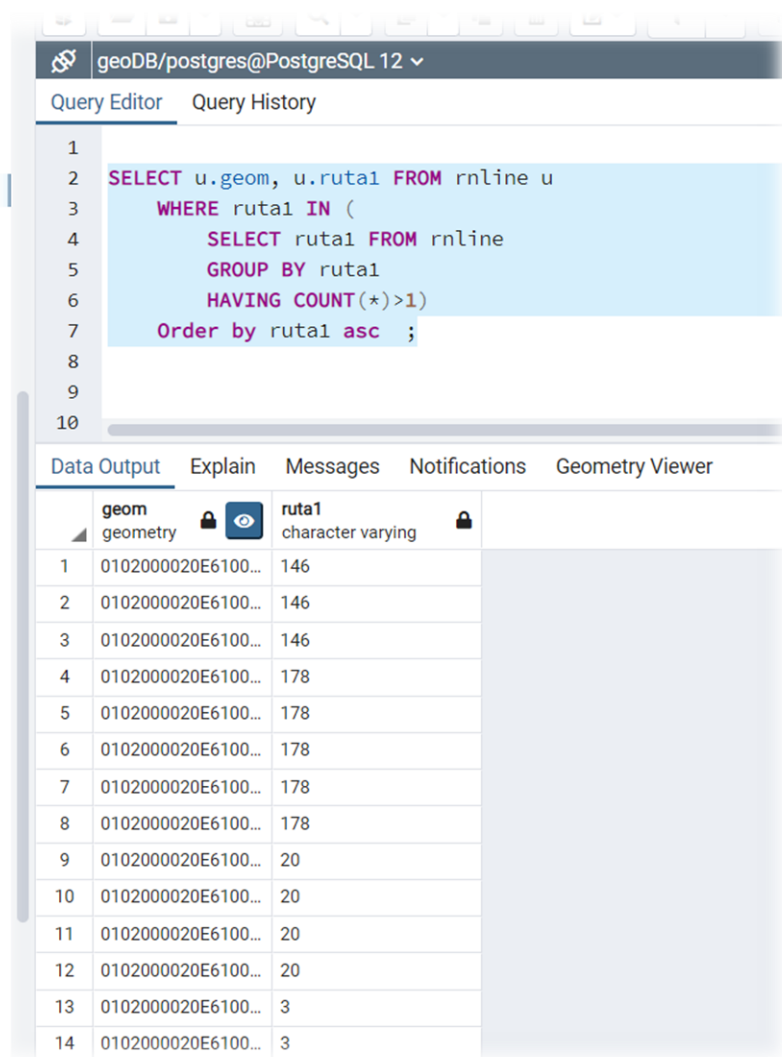


Figura 07- RN Multilinestring, Postgres-Qgis

Una forma de solucionar este tema, es utilizando una nueva función geométrica denominada `ST_Dump`(geometría g1), la cual se encarga de romper una cadena geométrica devolviendo un registro para cada uno de los componentes de la colección.

Al aplicar esta nueva query obtendremos los Linestring de todas las rutas y efectivamente de las rutas cuyo dibujo no era continuo, esta misma se dividió en varios registros con el mismo número de RN, esto se observa mejor en la *Figura 08- RN Transformado por st_Dump, Postgres*.

```
CREATE TABLE
rut2 AS
SELECT rutas.cod_ruta,
       rutas.distancia_,
       (st_dump(geom)).geom AS geom2
FROM rutas
WHERE rutas.sentido::text = 'A'::text
```



The screenshot shows a PostgreSQL Query Editor window titled 'geoDB/postgres@PostgreSQL 12'. The 'Query Editor' tab is active, displaying a SQL query. Below the query, the 'Data Output' tab shows the results of the query. The results are displayed in a table with two columns: 'geom' (geometry) and 'ruta1' (character varying). The table contains 14 rows of data.

	geom geometry	ruta1 character varying
1	0102000020E6100...	146
2	0102000020E6100...	146
3	0102000020E6100...	146
4	0102000020E6100...	178
5	0102000020E6100...	178
6	0102000020E6100...	178
7	0102000020E6100...	178
8	0102000020E6100...	178
9	0102000020E6100...	20
10	0102000020E6100...	20
11	0102000020E6100...	20
12	0102000020E6100...	20
13	0102000020E6100...	3
14	0102000020E6100...	3

Figura 08- RN Transformado por st_Dump, Postgres

Este nuevo dataset obtenido rut2, ahora le agregamos la columna long para conocer la longitud total de cada línea de dibujo, esto lo realizamos porque ahora tenemos una misma ruta dividida en varios linestring, aplicando la función `st_length`, su resultado se muestra a continuación, *Figura 09- rut2, dataset linestring del inventario- Postgres.*

```
ALTER TABLE rut2 ADD COLUMN long numeric (15,2);
UPDATE rut2 SET long =
(SELECT cast (st_length(rut2.geom2)*100 as decimal (15,2)))
```

geoDB/postgres@PostgreSQL 12

Query Editor Query History

```

1 SELECT * FROM rut2
2 Order by cod_ruta
3
4 UPDATE rut2 SET long = (SELECT cast (st_length(rut2.geom2)*100 as decimal (15,2)))
5 FROM rut2
6
7

```

Data Output Explain Messages Notifications Geometry Viewer

	cod_ruta character varying (254)	distancia_ numeric	geom2 geometry	long numeric (15,2)
1	0001	49.26	0102000020E6100...	50.89
2	0003	3048.36	0102000020E6100...	366.14
3	0003	3048.36	0102000020E6100...	555.46
4	0003	3048.36	0102000020E6100...	650.27
5	0003	3048.36	0102000020E6100...	277.58
6	0003	3048.36	0102000020E6100...	833.69
7	0003	3048.36	0102000020E6100...	483.26
8	0005	542.1	0102000020E6100...	576.63
9	0007	1232.7	0102000020E6100...	1290.65
10	0008	694.4	0102000020E6100...	730.80
11	0009	1944.21	0102000020E6100...	1855.51
12	0011	983.49	0102000020E6100...	913.34
13	0012	1591.84	0102000020E6100...	1520.16
14	0014	1109.17	0102000020E6100...	1049.49
15	0016	706.43	0102000020E6100...	692.72
16	0018	226.38	0102000020E6100...	234.59

Figura 09- rut2, dataset linestring del inventario- Postgres.

Ahora que tenemos todas las rutas en formato linestring con su longitud de dibujo, se puede aplicar la función **ST_LineInterpolatePoint** (geometría a_linestring, float8) trabajada anteriormente. Para verificar que la misma funciona se obtiene primeramente los puntos georreferenciados de la RN 05, la cual ya se había estudiado, pero esta vez usando la base del inventario vial en su totalidad. En la *Figura 10- Puntos RN05 usando data set inventario20- Postgres*, se muestra los puntos georreferenciados obtenidos.

-Cálculo del punto float

```

SELECT a.rn, a.disto,m.long,(a.disto / m.long) AS punto
FROM (SELECT * FROM rut2 g
JOIN codigorn p ON g.cod_ruta = p.ruta2)m ,inventario20 a
WHERE a.rn = m.ruta1 and m.ruta1='5'
GROUP by a.rn,a.disto,m.long
order by a.rn, a.disto asc

```


-Verificamos la rn5

```
SELECT t.cod_ruta, ST_lineinterpolatepoint(k.geom2,t.punto)
FROM(
    SELECT m.cod_ruta,a.disto,m.long,cast((a.disto / m.long) as
decimal(15,3)) AS punto
    FROM (SELECT * FROM rut2 g
    JOIN codigorn p ON g.cod_ruta = p.ruta2)m ,inventario20 a
    WHERE a.rn = m.ruta1
    GROUP by m.cod_ruta,a.disto,m.long
    order by m.cod_ruta, a.disto asc)t, rut2 k
WHERE t.cod_ruta = k.cod_ruta and k.cod_ruta='0005'
```

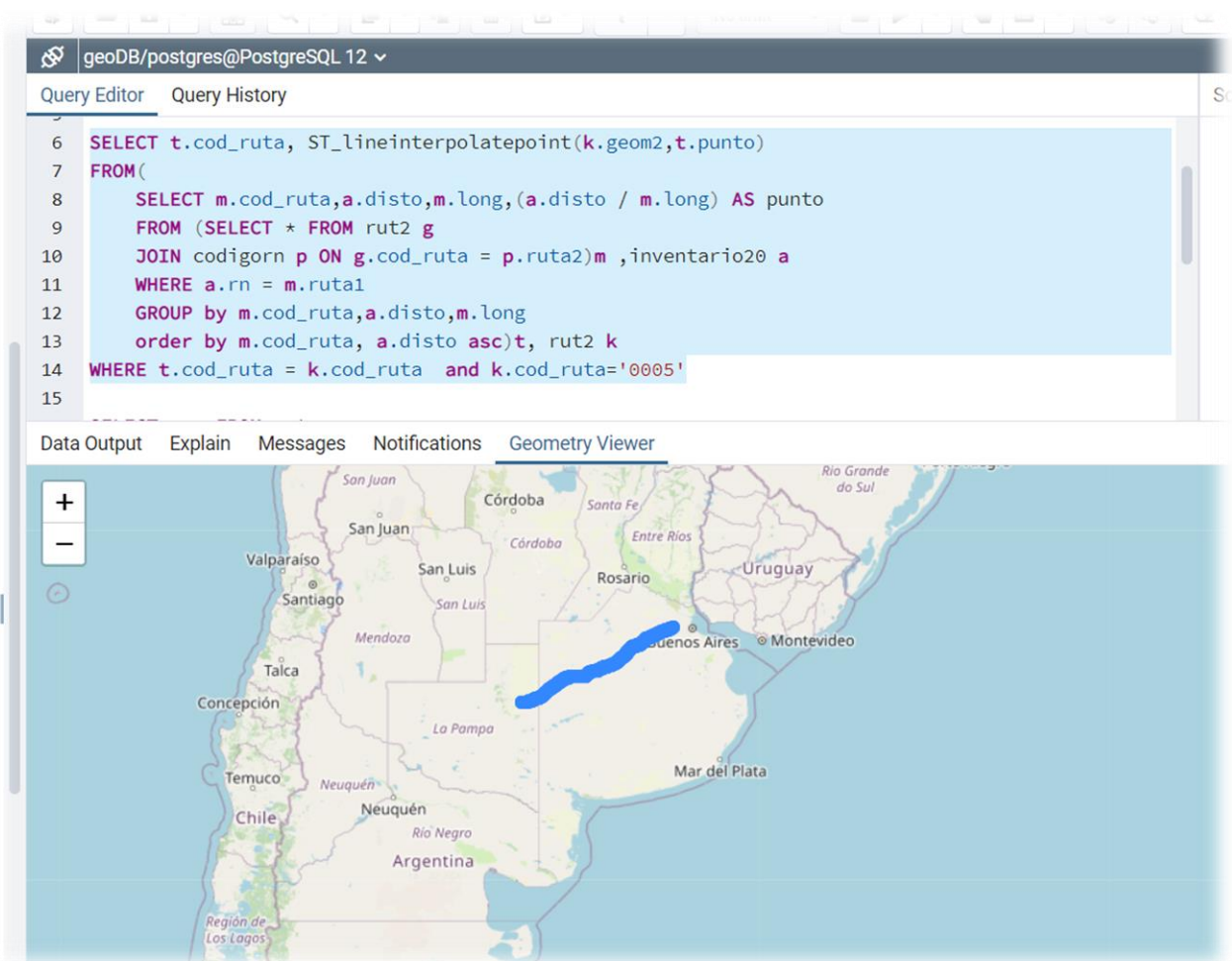


Figura 10- Puntos RN05 usando data set inventario20- Postgres,

Ahora calculamos los códigos de rutas no repetidos, para ir georreferenciando parte del inventario vial, como ya se conoce los algoritmos a utilizar se procede a usar los mismos, mostrando seguidamente el éxito de nuestra query en la *Figura 11- Puntos georreferenciados usando data set inventario20- Postgres.*

```

CREATE MATERIALIZED VIEW
rutline2 AS
WITH rutunic AS(
SELECT u.* FROM rut2 u
    WHERE cod_ruta IN (
        SELECT cod_ruta FROM rut2
        GROUP BY cod_ruta
        HAVING COUNT(*)=1))
SELECT t.cod_ruta,t.ruta1,t.disto,t.punto,
ST_lineinterpolatepoint(k.geom2,t.punto)      AS geom3
FROM(
    SELECT m.cod_ruta,m.ruta1,a.disto,m.long,cast((a.disto /
m.long)as decimal(15,3)) AS punto
    FROM (SELECT * FROM rut2 g
    JOIN codigorn p ON g.cod_ruta = p.ruta2)m ,inventario20 a
    WHERE a.rn = m.ruta1
    GROUP by m.cod_ruta, m.ruta1,a.disto,m.long
    order by m.cod_ruta, m.ruta1, a.disto asc)t, rut2 k,rutunic z

WHERE  t.cod_ruta  =  k.cod_ruta      and  k.cod_ruta=z.cod_ruta  and
t.punto<=1;

UPDATE inventario20 SET geom = m.geom3
FROM rutline2 m
WHERE m.ruta1= inventario20.rn and m.disto= inventario20.disto

```

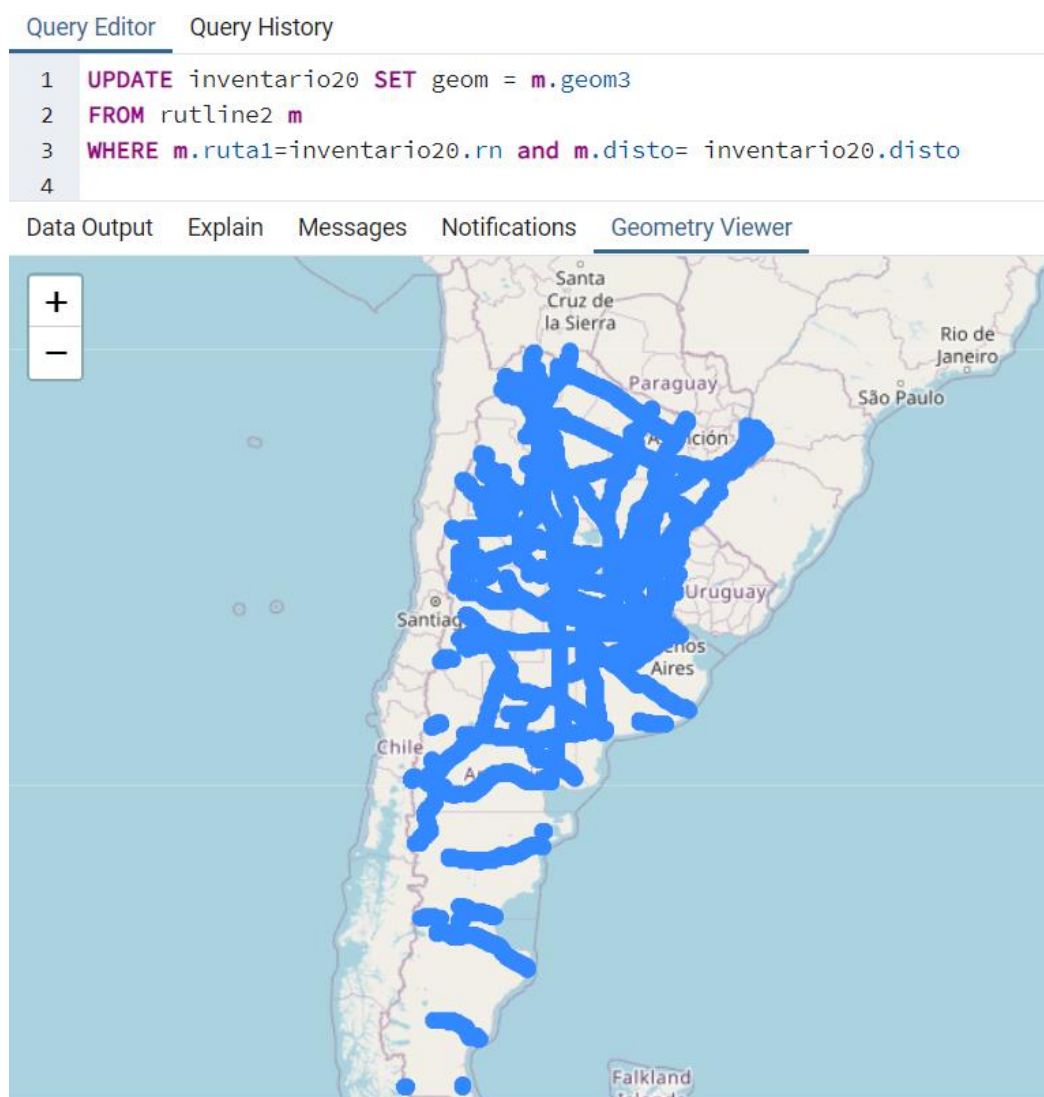



Figura 11- Puntos georreferenciados usando data set inventario20- Postgres.

Hasta ahora se trabajó con rutas continuas que son las gran mayorías, para trabajar con las rutas discontinuas, se deben analizar e identificar de manera separada el tramo que referencia a cada linestring con un mismo código de ruta, para luego poder vincular con el data set de inventario y asi aplicar las técnicas empleadas hasta ahora. A continuación, se se identifica la query a utilizar, seguida por la *Figura 12 Resumen Ruta 03 1ra sección*, donde se tiene una recopilación de imágenes Postgres junto al resultado obtenido para la Ruta Nacional Nro 3 primera parte de la sección.

-Verificamos la rn3 primera parte

```

SELECT t.cod_ruta, ST_lineinterpolatepoint(k.geom2,t.punto)
FROM(
    SELECT m.cod_ruta,a.disto,m.long,cast((a.disto / m.long) as
decimal(15,3)) AS punto

```

```

FROM (SELECT * FROM rut2 g
JOIN codigorn p ON g.cod_ruta = p.ruta2)m ,inventario20 a
WHERE a.rn = m.ruta1 and m.ruta1='3' and a.disto < m.long and
m.long=277.58
GROUP by m.cod_ruta,a.disto,m.long
order by m.cod_ruta, a.disto asc)t, rut2 k
WHERE t.cod_ruta = k.cod_ruta and k.cod_ruta='0003' and
k.long=277.58

```

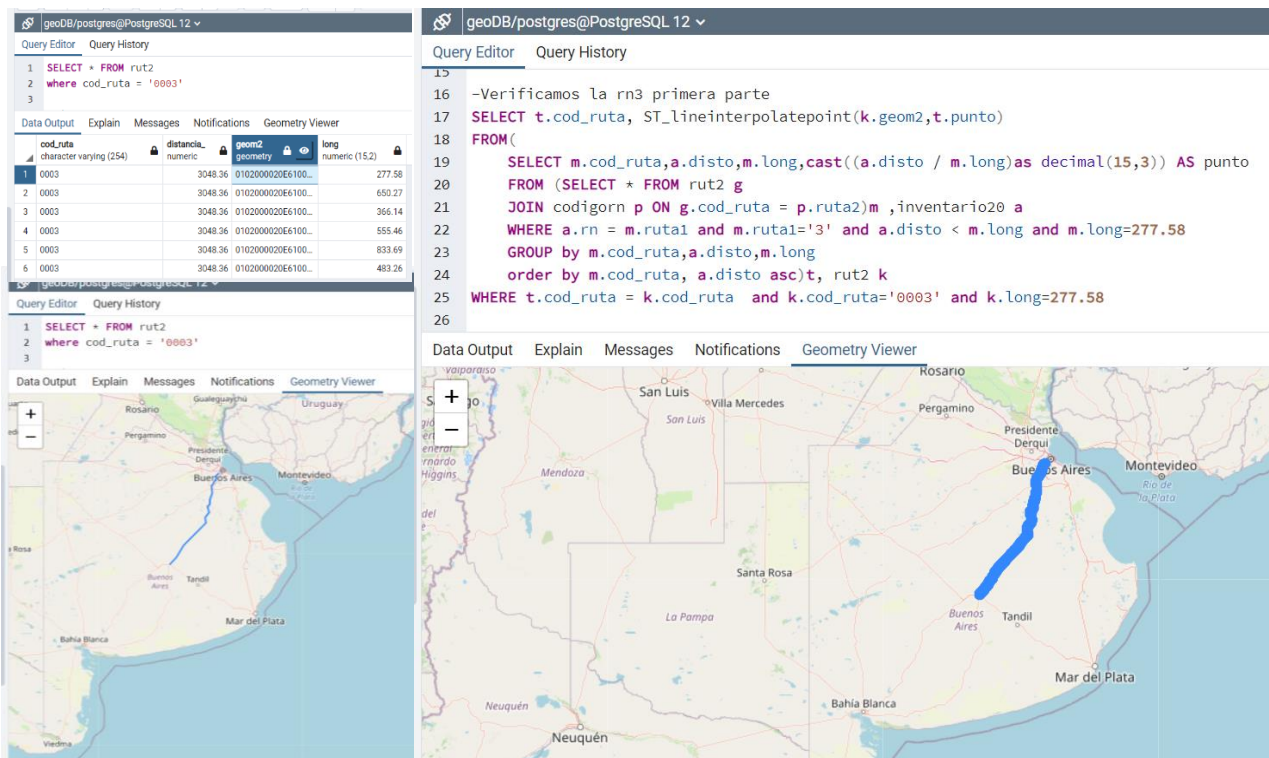


Figura 12 - Resumen Ruta 03 1ra sección

9- CONCLUSION

En función de todo lo analizado, se mostró como georreferenciar una tabla de puntos, mediante distancias lineales, si todas las rutas fueran continuas (cuya realidad no sería lógica) nuestro algoritmo hubiese sido un éxito total, no obstante, pudimos analizar gran parte de nuestro set de datos quedando solo 10 rutas por concluir. Esto no implica que es imposible realizarlo, todo lo contrario, estas rutas deben analizarse de manera separada, en este trabajo se mostró como trabajar con las rutas discontinuas haciendo referencia a un tramo de sección de la RN 03, de la misma manera se debe proseguir con los demás tramos discontinuos.

Todas las metodologías aplicadas en este trabajo, fue una gran herramienta de aproximación para obtener una lectura de puntos georreferenciados correspondiente al inventario vial 2020, a su vez se podría aplicar al Inventario 2021 y también a inventarios más históricos de Vialidad. Hoy en día es tan avanzada la innovación de herramientas informáticas, por lo cual Vialidad Nacional está trabajando en una aplicación móvil, que a futuro se utilizara para el relevamiento vial, obteniendo de esta manera un inventario georreferenciado de fácil almacenamiento, procesamiento y manejo para operarios, con lo cual se podrá obtener datos más exhaustivo de la red vial, con la finalidad de compartir toda esta información de manera oficial como se viene haciendo hasta el día de hoy a sus usuarios, para que los mismos puedan generar sus propios mapas en función a sus áreas de intereses.

En un futuro cercano todos los mapas y la información generada por las diferentes áreas pertenecientes a la Dirección Nacional de Vialidad, se podrán vincular entre sí, permitiendo obtener como resultado final la concreción de un Sistema Coordinado de Datos Viales.

10- REFERENCIA BIBLIOGRAFICA

[1] Autores División Relevamiento - Sección Inventario Vial (Vialidad Nacional)

“Guía de Relevamiento de Inventario Vial”

Página oficial de vialidad <https://www.argentina.gob.ar/obras-publicas/vialidad-nacional/sig-vial>

[2] Autores Dolorier Dolorier, Miguel Angel (2012)

“Manual de inventario vial georeferenciado: aplicaciones en señalización, accesibilidad y seguridad vial” <http://cybertesis.uni.edu.pe/handle/uni/15294>

[3] Autores Mateusz Ilba (2021) - Departamento de Geografía Social y Económica, Universidad de Economía de Cracovia, Cracovia, Polonia

"Algoritmo paralelo para mejorar el rendimiento de consultas espaciales en SQL: los casos de uso de bases de datos SQLite / SpatiaLite y PostgreSQL / PostGIS"

<https://www.sciencedirect.com/science/article/pii/S0098300421001382>

[4] Manual PostGIS 2.4.0 <http://postgis.net/stuff/postgis-2.4.0-es.pdf>