

Postgre/PostGIS Tutorial

Re-projecting Data



University
of Colorado
Denver

Created by: Ricardo Oliveira
ricardo.oliveira@ucdenver.edu

July 2014

On the previous tutorials we learned how to perform basic operations such as import data and ask basic queries. On this tutorial, and all that will follow, we will seek to answer a real-world question and therefore we will expand our Postgres knowledge based on such needs.

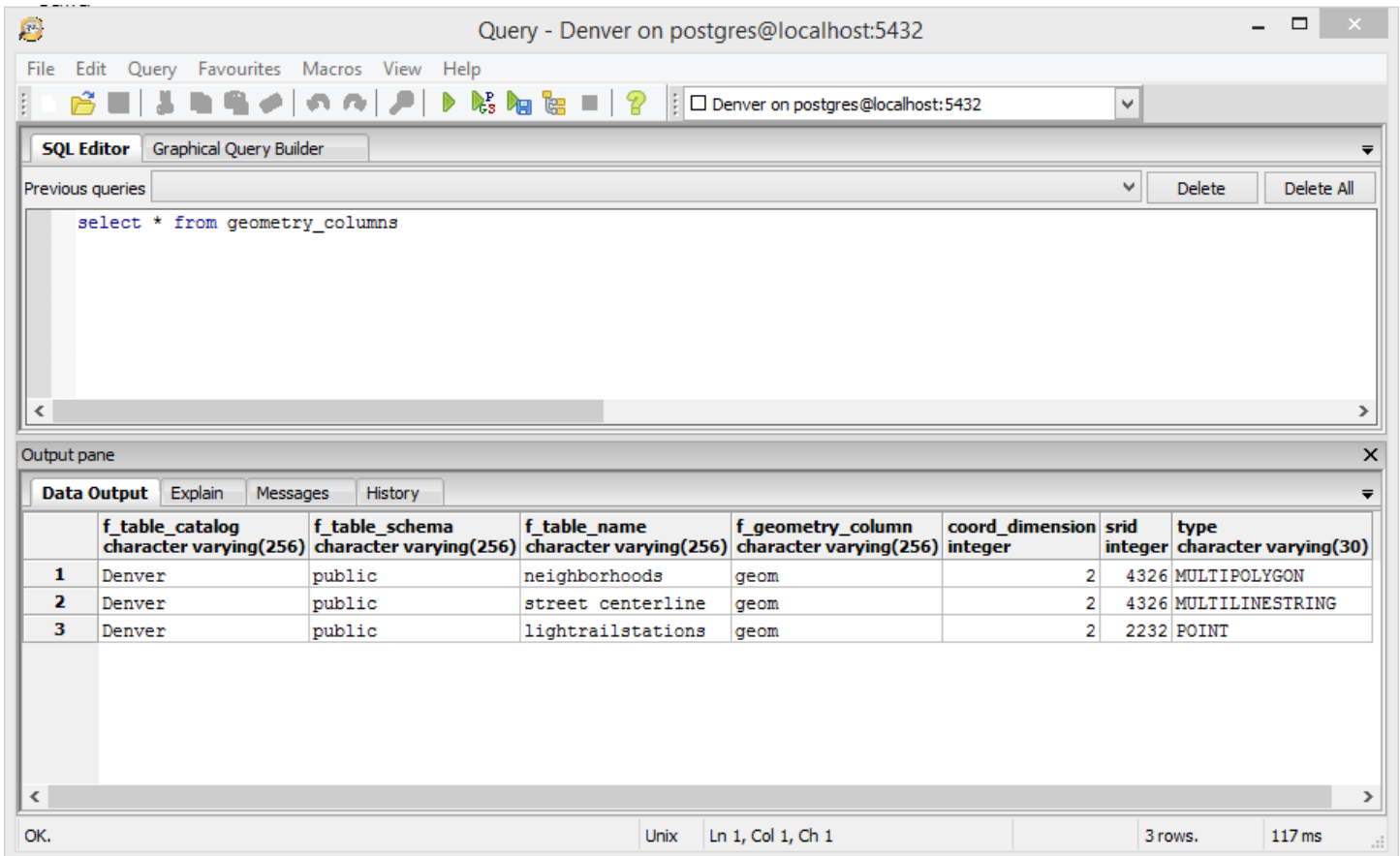
The question is: **The City of Denver is interested to know how the RTD's infrastructure relates and serves the city, the City of Denver asked us to create a report with such information.**

We should be able to answer this question by just using PostGIS.

Preparing Data

You may have notice that not our data is equal, since we used two different data sources, the City of Denver Open Catalog and Open Colorado, we may expect some differences. Let's check their properties on Postgres. Open pgAdmin and run the following query:

```
select * from geometry_columns
```



The screenshot shows the pgAdmin interface with the SQL Editor open. The query 'select * from geometry_columns' is entered and executed. The output pane displays a table with the following data:

	f_table_catalog character varying(256)	f_table_schema character varying(256)	f_table_name character varying(256)	f_geometry_column character varying(256)	coord_dimension integer	srid integer	type character varying(30)
1	Denver	public	neighborhoods	geom		2 4326	MULTIPOLYGON
2	Denver	public	street centerline	geom		2 4326	MULTILINESTRING
3	Denver	public	lightrailstations	geom		2 2232	POINT

In order to make spatial queries we must to have our data in the same projected system. For instance, our light rail stations are in Colorado Central which uses US feet as a measure unit, while all our other data are in WGS 84 which is a coordinate system. If we want to work with spatial relationships all our data must be in the same unit of measure, and since all our data is geographic located in the same place, Denver, it is a good a idea to have them in the same projected system.

Let's re-project our neighborhoods and street centerline data into Colorado Central.

TIP: In order to have relevant information about your project always in hand create a new notepad document to store document such as SRID in the data folder and/or specific queries that you may use again.

Let's first deal with the neighborhoods data.

```
alter table neighborhoods
alter column geom type geometry (MULTIPOLYGON, 2232)
using ST_Transform (geom,2232)
```

TIP: When dealing with long queries it is always a good idea to enter each part in a new line, pressing enter you move to the line below since f5 is the running shortcut. This helps to organize each the query in a way that makes it easier to read, also it helps when debugging a error in the query.

Now let's understand what each part does in our query.

`Alter table` is used when we want to change some aspect of our data, we used this on the fourth tutorial to change the name of the neighborhoods data. Here we are declaring that we want to change something on the neighborhoods data.

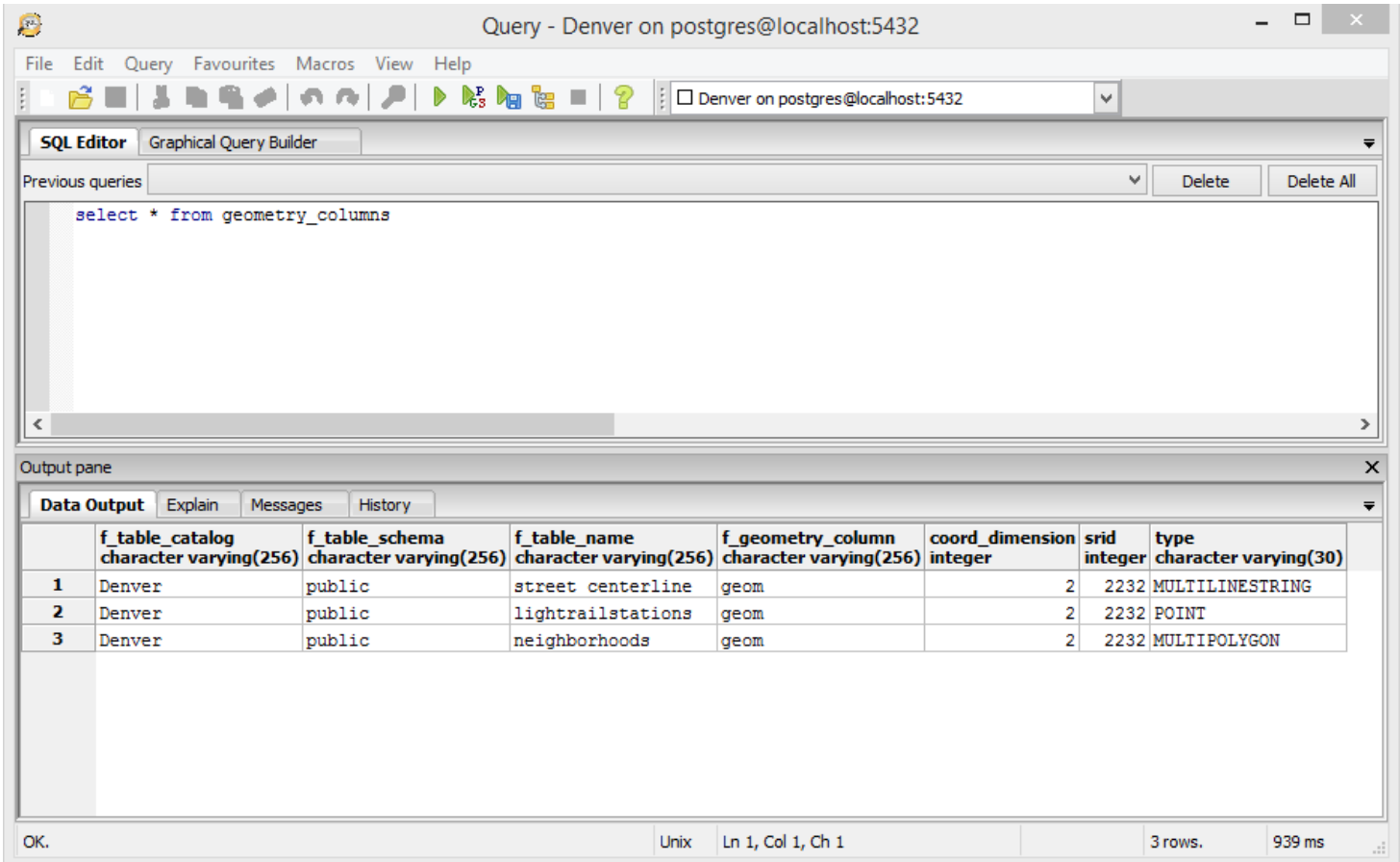
`Alter column` is used here to direct our query to the exact area we want to change. We specified the `geom` column, this column contains all the spatial data, on this specific column we specify what type of geometry we are dealing with in this case `MULTIPOLYGON`, and finally the SRID we are projecting to, 2232 (Colorado Central).

`Using` specifies which command will be applied to the column, in this case `ST_Transform` which returns a new geometry based on the SRID specified on the parameter.

Now let's repeat the process for the street data.

```
alter table street_centerline
alter column geom type geometry(MULTILINESTRING, 2232)
using ST_Transform(geom,2232)
```

Observe how in this case we used MULTILINESTRING instead of MULTIPOLYGON, this is because of the nature of the data.



The screenshot shows a PostgreSQL SQL Editor window titled "Query - Denver on postgres@localhost:5432". The SQL Editor pane contains the query: `select * from geometry_columns`. The Output pane displays the results of the query in a table format.

	f_table_catalog character varying(256)	f_table_schema character varying(256)	f_table_name character varying(256)	f_geometry_column character varying(256)	coord_dimension integer	srid integer	type character varying(30)
1	Denver	public	street_centerline	geom		2 2232	MULTILINESTRING
2	Denver	public	lightrailstations	geom		2 2232	POINT
3	Denver	public	neighborhoods	geom		2 2232	MULTIPOLYGON

At the bottom of the window, the status bar shows "OK.", "Unix", "Ln 1, Col 1, Ch 1", "3 rows.", and "939 ms".

Now that we re-projected our data we are ready to ask spatial questions.