Spatial Database Management
GEP 664 / EES 79903
Class #6: Spatial database fundamentals

Frank Donnelly

Dept of EEGS, Lehman College CUNY

Spring 2017

Today's Topics

PostgreSQL & PostGIS

Spatial Databases

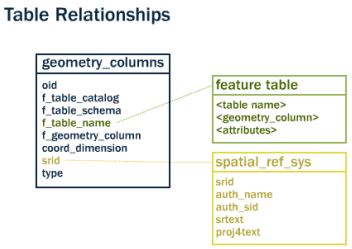
Geographic vector features stored as series of coordinates in a dedicated column, with a geometry data type and sub-types:

- POINT(0 0)
- LINESTRING(0 0,1 1,1 2)
- POLYGON((0 0,4 0,4 4,0 4,0 0),(1 1, 2 1, 2 2, 1 2,1 1))

The data types and sub-types allow you to model real-world features, like the fast food example in *PostGIS in Action* Chapter 1.

Spatial Reference Systems

Geometry columns are referenced with internal metadata tables that store coordinate systems.




```
SELECT *
FROM geometry_columns;
```

	f_table_cata character va	f_table_schen character var	f_table_name character varying(256)	f_geometry_column character varying(256)	coord_dim integer	srid integer	type character varying(
1	gep664	nyc	subway_complexes	geometry	2	2263	POINT
2	gep664	nyc	subway_stations	geometry	2	2263	POINT
3	gep664	nyc	census_tracts	geometry	2	2263	MULTIPOLYGON

Spatial Relationships & Analysis

When features have geometry they can be compared spatially. Spatial functions begin with the prefix "ST_"

```
SELECT bname, stop_name, trains
FROM boroughs, stations
WHERE bname='Bronx' AND
ST_Within (stations.geom, boroughs.geom)
```

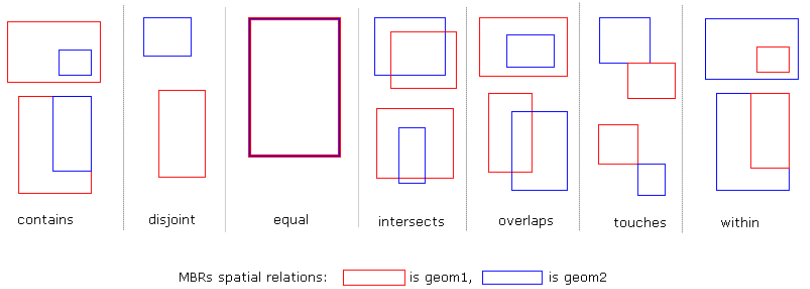


Image source: https://www.gaia-gis.it/spatialite-2.1/SpatiaLite-manual.html

Spatial Data Types

PostGIS adds support for four spatial data types:

- **Geometry:** the most common, uses planar Cartesian grid
- **Geography:** uses geodetic system, based on spherical surface
- **Raster:** continuous grid of pixels of equal size
- **Topology:** models the interconnectedness of features, all boundaries are shared

Geometry Columns

geometry_columns is a view stored in the public schema, use it to see the spatial attributes of features in the database.

```
SELECT *
FROM geometry_columns;
```

	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	gep664	nyc	census_tracts	geometry	2	2263	MULTIPOLYGON
2	gep664	nyc	subway_complexes	geometry	2	2263	POINT
3	gep664	nyc	subway_stations	geometry	2	2263	POINT

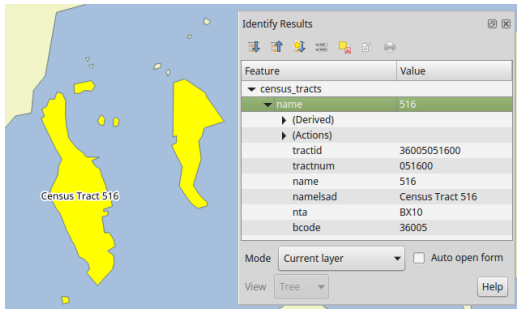
There are separate views for geography_columns and raster_columns.

Geometry Subtypes

- Geometry subtypes are type modifiers
- i.e. varchar(10), geometry(point)
- Points, Linestrings, and Polygons
- Subtypes come in single and multi options
- There is a geometry collection type for storing mixed features (it has limited uses)
- Store X,Y coordinates but also Z for elevation / depth, M for measurements

Singlepart versus Multipart

Multipart subtypes are used to store geometry for features that have multiple, disparate parts.



This census tract consists of many islands but should be stored as a single feature in the table, and not as several features.

Point Subtype

Representation:

```
POINT(0 0)
MULTIPOINT((0 0),(1 2))
```

Spatial functions:

ST_X return the X coordinate: ST_X(geom)

ST_Y return the Y coordinate: ST_Y(geom)

Image source: http://workshops.boundlessgeo.com/postgis-intro/geometries.html

Linestring Subtype

Representation:

```
LINESTRING(0 0,1 1,1 2)
MULTILINESTRING((0 0,1 1,1 2),(2 3,3 2,5 4))
```

Spatial functions:

ST_Length returns length of the line

ST_StartPoint returns first coordinate

ST_EndPoint returns last coordinate

ST_NPoints returns number of coordinates in the line

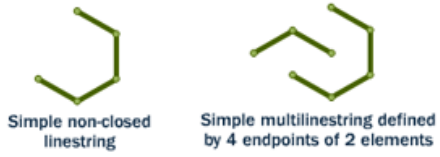


Image source: http://workshops.boundlessgeo.com/postgis-intro/geometries.html

Polygon Subtype

Representation:

```
POLYGON((0 0,4 0,4 4,0 4,0 0),(1 1, 2 1, 2 2, 1 2,1 1))
MULTIPOLYGON(((0 0,4 0,4 4,0 4,0 0),(1 1,2 1,2 2,1 2,1
((-1 -1,-1 -2,-2 -2,-2 -1,-1 -1)))
```

Spatial functions:

ST_Area returns area of the polygon
ST_Perimeter returns perimeter of the polygon
ST_Centroid calculates and returns center point of polygon
ST_NRings returns the number of rings

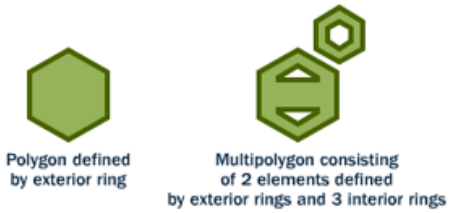


Image source: http://workshops.boundlessgeo.com/postgis-intro/geometries.html

WKT and WKB

Geometry stored in PostGIS is not readily viewable; it must be transformed when queried.

The OpenGIS specification defines 2 ways of expressing spatial objects: the Well-Known Text (WKT) and the Well-Known Binary (WKB) forms. PostGIS has 2 commands for each: an OGC compliant form that outputs subtype and coordinates, and a PostGIS specific form that also outputs the SRID.

- ST_AsText (OGC compliant)
- ST_AsEWKT (PostGIS specific)
- ST_AsBinary (OGC compliant)
- ST_AsEWKB (PostGIS specific)

Sample Data - NYC Spatial Features

We'll begin using some spatial features in the nyc schema of the gep664 database as examples:

spatial data : census tracts, subway stations, subway complexes (with ridership)

attribute data : 2010 census data by tract, 2010 census variable descriptions, neighborhood tabulation areas, subway closure notes

Source: NYC Geodatabase

Retrieving Geometry

```
SELECT stop_id, stop_name, trains, geometry
FROM nyc.subway_stations
ORDER BY stop_id;
```

	stop_id character varying(3)	stop_name character varying(38)	trains character varying(13)	geometry geometry(Point,2263)
1	101	Van Cortlandt Park	1	0101000020D7080000474EDD4F86E42E4118DE09D59C111041
2	103	238 St	1	0101000020D70800000FB37C6899DF2E411CBFDF880BEF0F41
3	104	231 St	1	0101000020D7080000634D0AD10DD72E41334B0061D8AC0F41
4	106	Marble Hill - 225	1	0101000020D7080000FA69605445CC2E41212050FAE57B0F41

```
SELECT stop_id, stop_name, trains, ST_AsText(geometry) AS
    geom
FROM nyc.subway_stations
ORDER BY stop_id;
```

	stop_id character varying(3)	stop_name character varying(38)	trains character varying(13)	geom text
1	101	Van Cortlandt Park	1	POINT(1012291.1559853 263271.208045454)
2	103	238 St	1	POINT(1011660.70407638 261601.441833013)
3	104	231 St	1	POINT(1010566.90828173 259483.047363842)
4	106	Marble Hill - 225	1	POINT(1009186.66479808 257916.74722314)

Functions and Retrieving Geometry

```
SELECT tractid, namelsad,
ST_AsText(ST_Centroid(geometry)) AS center
FROM nyc.census_tracts;
```

	tractid character varying(11)	namelsad character varying(20)	st_astext text
1	36005000100	Census Tract 1	POINT(1016743.81969954 227485.577111285)
2	36005000200	Census Tract 2	POINT(1023248.8171318 234696.087564549)
3	36005000400	Census Tract 4	POINT(1025130.41233168 234251.232903636)
4	36005001600	Census Tract 16	POINT(1023532.5305191 237637.296734385)
5	36005001900	Census Tract 19	POINT(1007839.01189344 231875.489676178)

```
SELECT tractid, namelsad,
ST_AsText(ST_Envelope(geometry)) AS bbox
FROM nyc.census_tracts;
```

	tractid character varying(11)	namelsad character varying(20)	bbox text
1	36005000100	Census Tract 1	POLYGON((1013915.8411759 225567.403537144,1013915.8411759 230144.520948263,
2	36005000200	Census Tract 2	POLYGON((1021042.88616282 231725.380059893,1021042.88616282 236332.69508832
3	36005000400	Census Tract 4	POLYGON((1023080.5146628 231725.380059893,1023080.5146628 236731.687608556,
4	36005001600	Census Tract 16	POLYGON((1022317.86658099 236244.477116773,1022317.86658099 239221.30939262
5	36005001900	Census Tract 19	POLYGON((1003162.87145305 229259.630587584,1003162.87145305 234353.04175736

Summary Functions

Coordinates and measurements that are returned are based on the feature's underlying spatial reference system.

```
SELECT tractid, namelsad, ST_Area(geometry) as area_sqft,
ST_Area(geometry)/2.788e+7 AS area_sqmi,
ST_Perimeter(geometry) AS per_ft,
ST_Perimeter(geometry)/5280 AS per_mi
FROM nyc.census_tracts;
```

	tractid character varying(11)	namelsad character varying(20)	area_sqft double precision	area_sqmi double precision	per_ft double precision	per_mi double precision
1	36005000100	Census Tract 1	16999855.9907697	0.609750932237076	17276.2299442922	3.27201324702505
2	36005000200	Census Tract 2	4900966.3978947	0.17578789088575	13463.0160075607	2.54981363779559
3	36005000400	Census Tract 4	9820750.75666681	0.352250744500244	24371.1211851008	4.61574264869333
4	36005001600	Census Tract 16	5221245.09031671	0.187275648863584	9672.23739891704	1.83186314373429
5	36005001900	Census Tract 19	17691917.2229319	0.634573788483929	28592.4731672748	5.41524113016568

Rounding and Casting

Use ROUND by itself to display values as integers. To round and preserve decimal places, you must CAST the output of the geometry calculation as numeric and then ROUND.

```
SELECT tractid, namelsad,
ROUND(ST_Area(geometry)) as area_sqft,
ROUND(CAST(ST_Area(geometry)/2.788e+7 AS numeric),3) AS
      area_sqmi
FROM nyc.census_tracts;
```

	tractid character varying(11)	namelsad character varying(20)	area_sqft double precision	area_sqmi numeric
1	36005000100	Census Tract 1	16999856	0.610
2	36005000200	Census Tract 2	4900966	0.176
3	36005000400	Census Tract 4	9820751	0.352
4	36005001600	Census Tract 16	5221245	0.187
5	36005001900	Census Tract 19	17691917	0.635

Combining Regular and Spatial Functions

Select the northernmost subway station.

```
SELECT stop_id, stop_name, trains
FROM nyc.subway_stations
WHERE ST_Y(geometry) IN (
SELECT MAX(ST_Y(geometry))
FROM nyc.subway_stations);
```

	stop_id character varying(3)	stop_name character varying(38)	trains character varying(13)
1	201	Wakefield - 241	2

Adding and Inserting Geometry

Add to a new table with CREATE TABLE or to an existing table with ALTER TABLE. Specify: type(subtype, srid)

```
CREATE TABLE nyc.weather_station2 (
station_id varchar(20) PRIMARY KEY,
station_name text,
elevation numeric(6,1),
geom geometry(point, 4269));
```

Inserting coordinates manually as WKT. You must designate the SRS for each feature to match the geometry column .

```
INSERT INTO nyc.weather_station
VALUES ('WBAN:04781', 'ISLIP AIRPORT NY US', 25.6,
ST_GeomFromText('POINT(40.7939 73.1017)',4269)),
('WBAN:54780', 'MONTAUK AIRPORT, NY US', 2.1,
    ST_GeomFromText('POINT(41.07306 71.92333)',4269));
```

Adding and Building Geometry

Add to a new table with CREATE TABLE or to an existing table with ALTER TABLE. Specify: type(subtype, srid)

```
ALTER TABLE nyc.weather_station
ADD COLUMN geom geometry(point, 4269);
```

If you already have coordinates stored in designated fields you can build geometry. Wrap SRS function around Point function.

```
UPDATE nyc.weather_station
SET geom = ST_SetSRID(ST_Point(lon,lat),4269)
```

ST_Point works for 2D; use ST_MakePoint for 3D.

Naming the Geometry Column

Like any other column, you can name the geometry column whatever you want. Some suggestions:

- geom : concise and unambiguous
- geometry : ok and often used, but violates conventions as it's a reserved keyword
- geom_(srid) : like geom_nad83 to indicate the srs of the geometry
- geo : don't do this; it can be confused with the geography type

Today's Topics

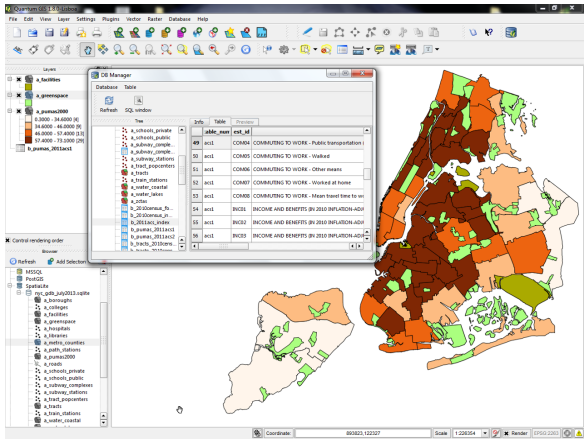
PostGIS & Desktop Support

Open Source:

- QGIS
- OpenJUMP
- uDig
- gvSIG

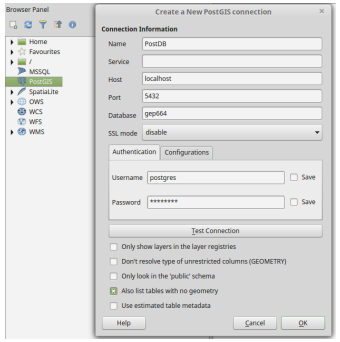
Proprietary:

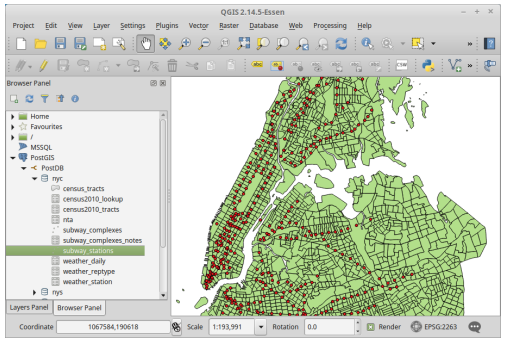
- ArcGIS
- MapINFO
- Manifold



Connect & Display in QGIS

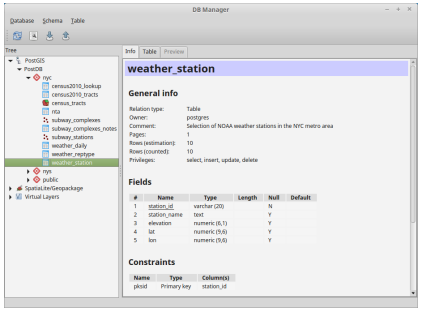
Connect to the database through the Browser panel. Drag and drop spatial features from the layers panel into the map view.

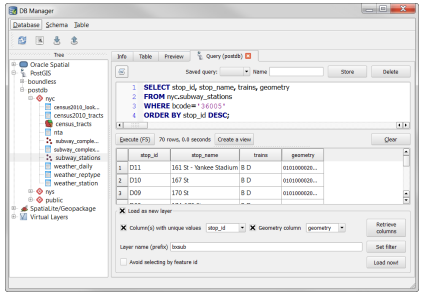




QGIS Database Manager

View tabular and spatial data in the DB Manager plugin.
Write SQL and spatial SQL queries and visualize the results.





Today's Topics

Spatial Databases & Geometry

PostGIS & Desktop GIS

Next Class

Due Next Class

The following are due at the beginning of our next class:

Assignment #6

Posted on the course website

Readings for Class #7

Listed in the syllabus, in the *PostGIS In Action* book

PostGIS in Action Readings

READ Chapters 3, 4, & 6

But in these chapters you can skim or skip the following:

- 3.2.4: Covering the globe when distance is a concern
- 4.3 Importing and exporting vectors with ogr2ogr
- 4.4 Importing OpenStreetMap data
- 4.5 Importing and exporting raster data
- 6.18 Geohash