WEEK 03
SPATIAL DATA

Instructor: Yanan Wu
TA: Vanchy Li

Spring 2025

WEEK 03

LECTURE SESSION

Instructor: Yanan Wu
TA: Vanchy Li

Spring 2025

3.1
GEOMETRY

GEOMETRY

- The **geometry** data type is the core data type in **PostGIS** used to store **spatial objects**. It can represent **geometric shapes** such as:
  - **Points** (e.g., a location on a map)
  - **Lines** (e.g., a road or river)
  - **Polygons** (e.g., a building footprint or city boundary)
  - **Collections** of geometries (e.g., MULTIPOINT, MULTILINESTRING, MULTIPOLYGON)
- The **geometry data type** in PostGIS supports **two-dimensional (2D)**, **three-dimensional (3D)**, and even **four-dimensional (4D)** spatial data.

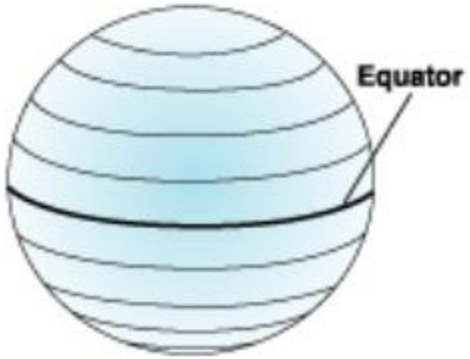
SPATIAL REFERENCE SYSTEM (SRS)

- A Spatial Reference System (SRS) (also called a Coordinate Reference System (CRS)) defines how geometry is referenced to locations on the Earth.

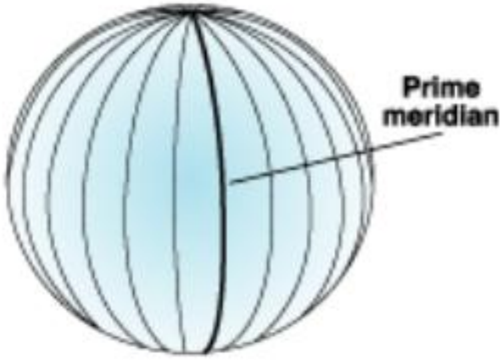
- Three types of SRS;
  - A **geodetic** SRS uses angular coordinates (longitude and latitude) which map directly to the surface of the earth.
  - A **projected** SRS uses a mathematical projection transformation to "flatten" the surface of the spheroidal earth onto a plane. It assigns location coordinates in a way that allows direct measurement of quantities such as distance, area, and angle. The coordinate system is Cartesian, which means it has a defined origin point and two perpendicular axes (usually oriented North and East). Each projected SRS uses a stated length unit (usually meters or feet). A projected SRS may be limited in its area of applicability to avoid distortion and fit within the defined coordinate bounds.
  - A **local** SRS is a Cartesian coordinate system which is not referenced to the earth's surface. In PostGIS this is specified by a SRID value of 0.

GEOGRAPHIC COORDINATE SYSTEM (OR GEODETIC)

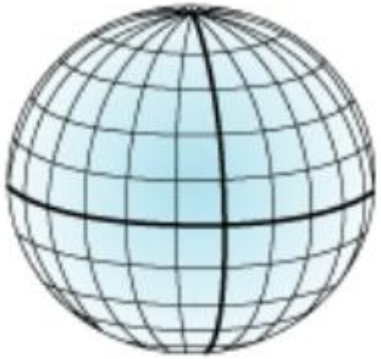
- A **geographic coordinate system** (**GCS**) is a spherical or geodetic coordinate system for measuring and communicating positions directly on Earth as latitude and longitude



Parallels
(Lines of latitude)



Meridians
(Lines of longitude)

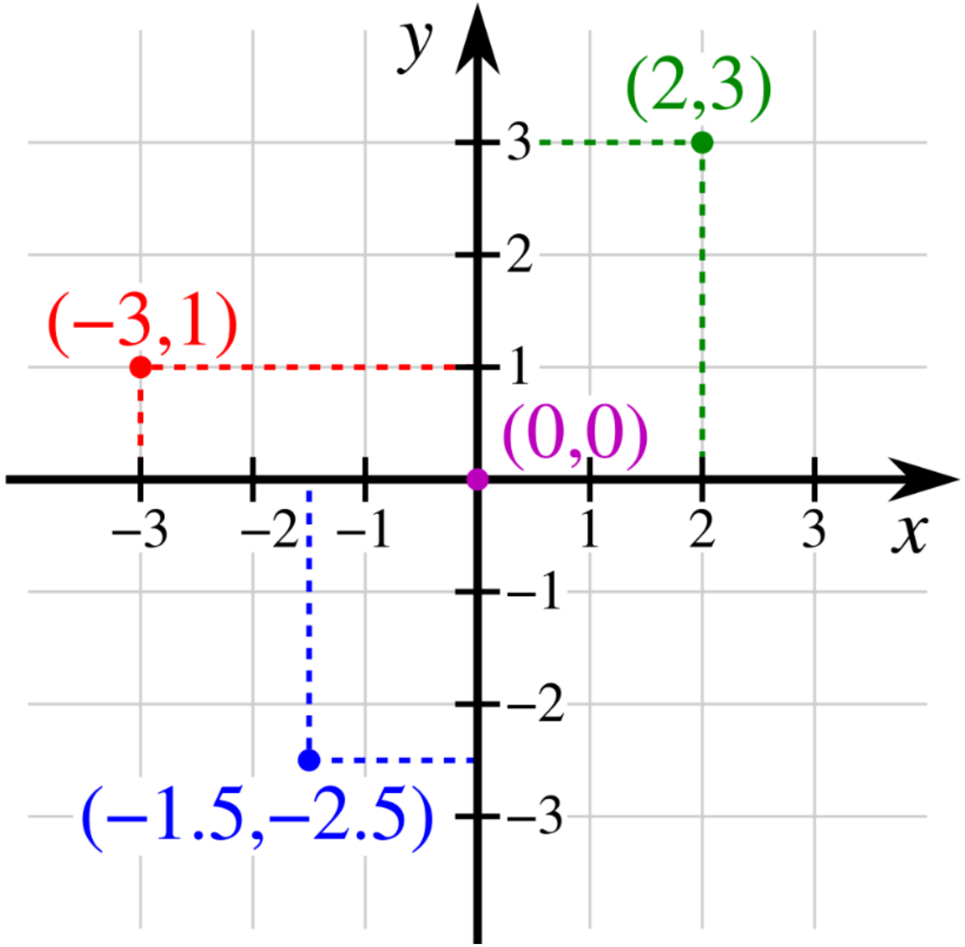


Graticular
Network

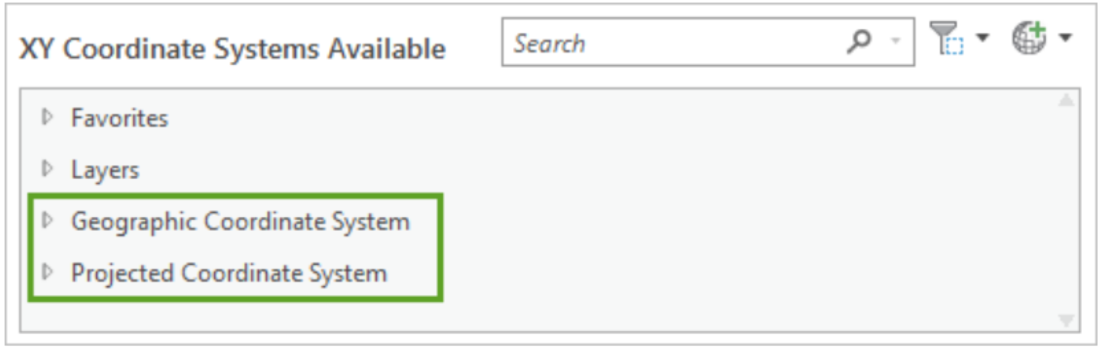
PROJECTED COORDINATE SYSTEM (OR PLANAR, GRID)

- The **projected coordinate system** assumes a flat, two-dimensional space where spatial measurements are made using **X** and **Y** coordinates.
  - **X-coordinate**: Represents the horizontal position.
  - **Y-coordinate**: Represents the vertical position.

- It is a type of spatial reference system that represents locations on Earth using Cartesian coordinates (x, y) on a planar surface created by a particular map projection.



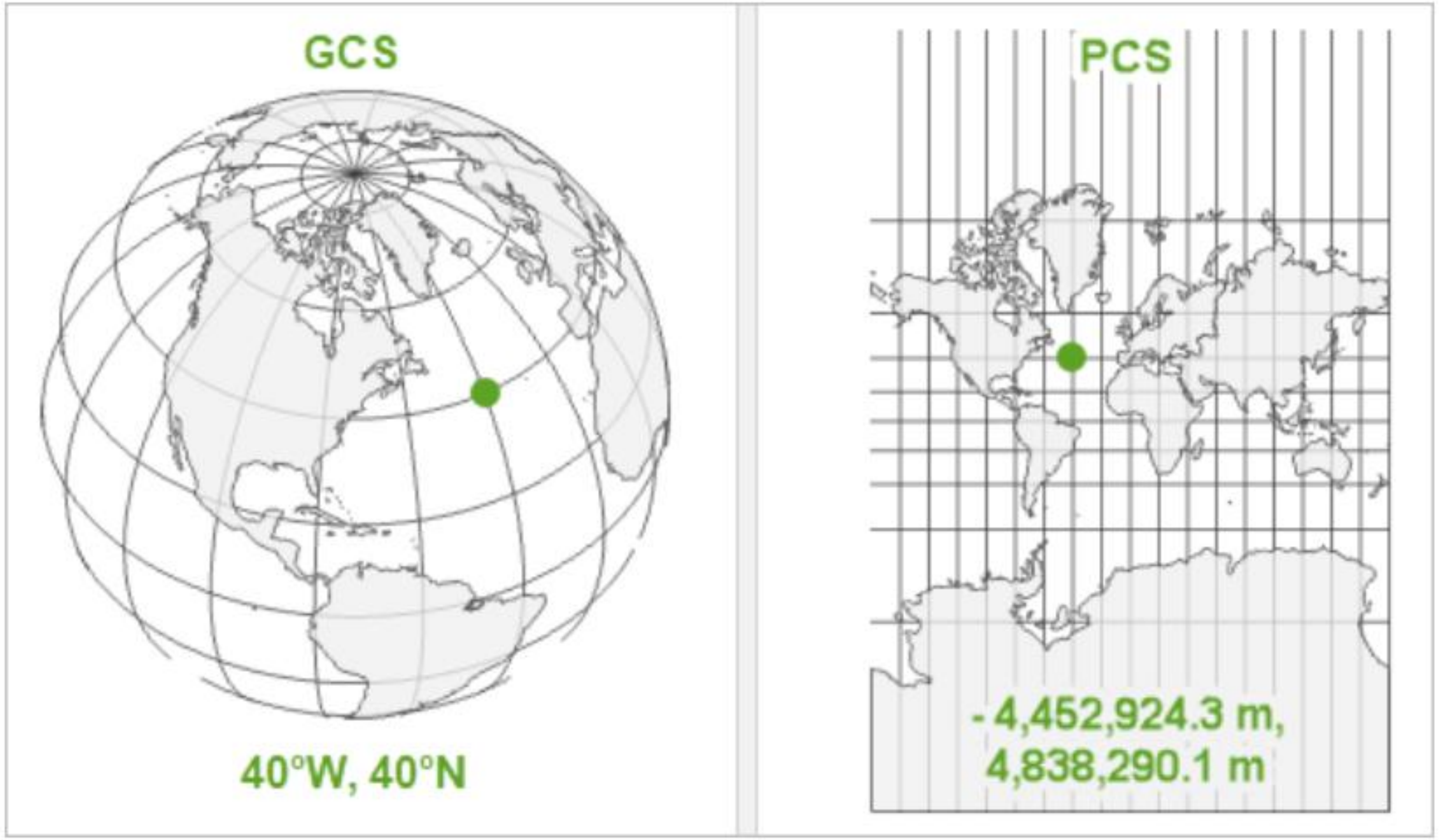
SETTING COORDINATE SYSTEM IN ARCGISPRO


XY Coordinate Systems Available
Search
Favorites
Layers
Geographic Coordinate System
Projected Coordinate System

A Geographic Coordinate System (GCS) represents locations on a round surface, recording them in angular units (typically degrees).

In contrast, a Projected Coordinate System (PCS) represents locations on a flat, two-dimensional plane, using linear units (usually meters).

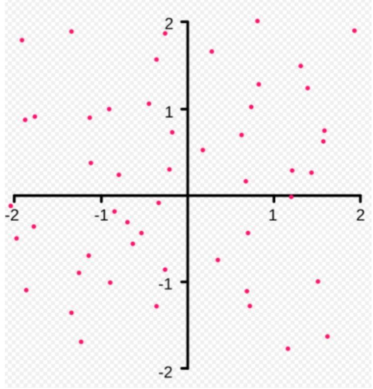
GCS AND PCS



SUBTYPE OF GEOMETRY - POINTS

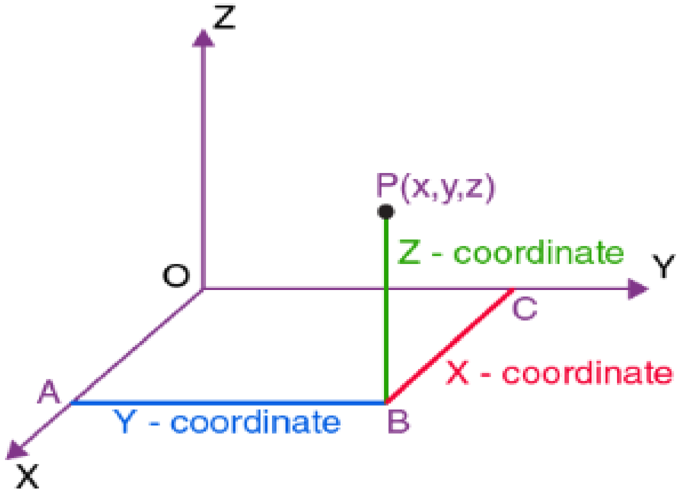
- POINT

A point in 2D space specified by its X and Y coordinates



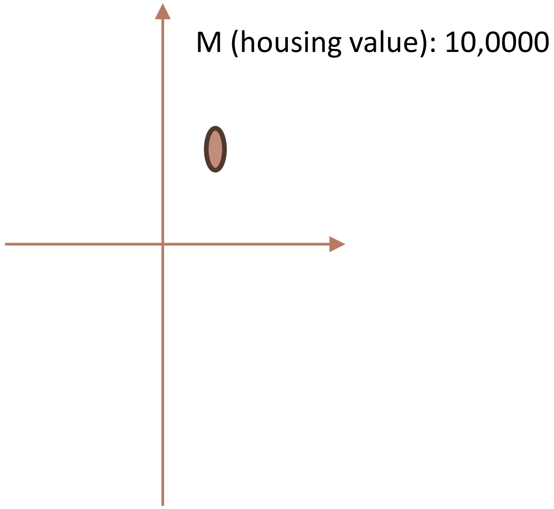
- POINTZ

A point in 3D space specified by its X, Y, and Z coordinates



- POINTM

point in 2D space with a measured value specified by its spatial X and Y coordinates plus an M value

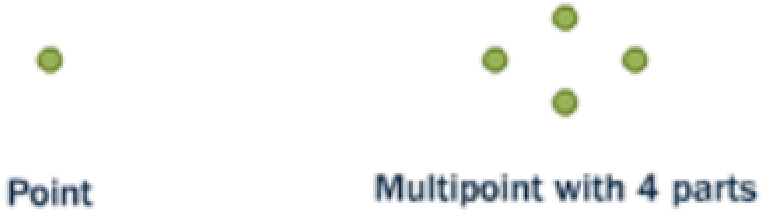


🤔 What is POINTZM?

SUBTYPE OF GEOMETRY - MULTIPOINTS

- A spatial point represents a city on the earth

- A multipoints with four points within one geometry to represent four cities on the earth



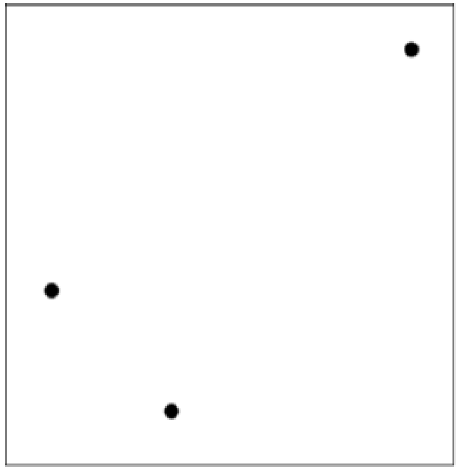
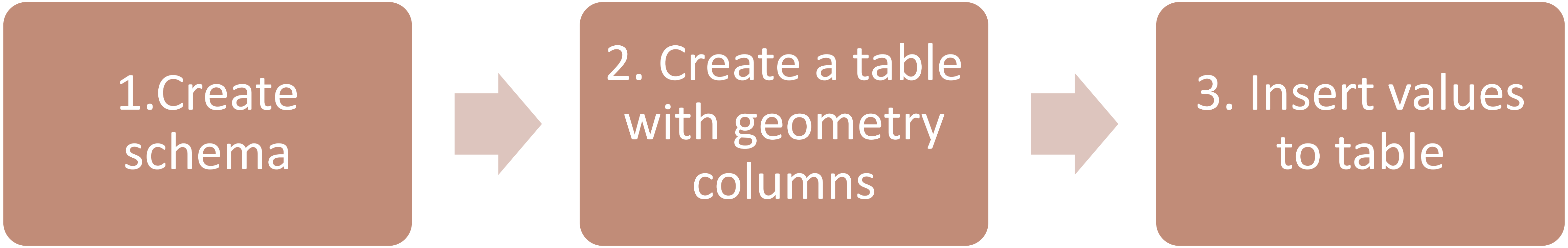


Figure 2.8 A single multipoint `geometry` (not three distinct points!)

3.1.1CREATE POINT

CREATE POINTS WITH SPATIAL DATA IN POSTGIS



1.Create schema

2. Create a table with geometry columns

3. Insert values to table

1. CREATE SCHEMA

```
CREATE SCHEMA ch03;
```

- A **schema** in PostGIS (and PostgreSQL) is a **logical container** used to organize and manage **database objects**, such as **tables, views, functions, and spatial data**.
- Schema provides a **namespace** to avoid naming conflicts between objects and helps manage **database permissions** more effectively.
- Think of a schema as a **folder** inside a database, where you can group related objects together to keep things organized.

2. CREATE TABLE: SYNTAX

```
Syntax:

CREATE TABLE table_name (
    column1 datatype,
    column2 datatype,
    column3 datatype,
   ....
);
```

2. CREATE TABLE: EXAMPLE

```
CREATE TABLE ch03.clarku (
id serial PRIMARY KEY,
p geometry(POINT),
pz geometry(POINTZ),
pm geometry(POINTM),
pzm geometry(POINTZM),
p_srid geometry(POINT,4326));
```

- id: This is a unique identifier for each row in the table.
- serial: This is an auto-incrementing integer column.
- PRIMARY KEY: This ensures the id is unique and cannot be NULL.

- p: The column name.
- geometry(POINT): This defines a 2D point geometry with X and Y coordinates.

Example: A point representing a latitude/longitude location or X/Y coordinates in a planar system

3. INSERT VALUE: EXAMPLE

```
INSERT INTO ch03.clarku (p, pz, pm, pzm, p_srid)
VALUES ( ST_GeomFromText('POINT(-71.8231 42.2510)’),
ST_GeomFromText('POINTZ(-71.8231 42.2510 100)’),
ST_GeomFromText('POINTM(-71.8231 42.2510 200)’),
ST_GeomFromText('POINTZM(-71.8231 42.2510 300 400)’),
ST_SetSRID(ST_GeomFromText('POINT(-71.8231 42.2510)’), 4326));
```

WELL-KNOWN BINARY (WKB)

- **Well-Known Binary (WKB)** is a binary encoding standard used to represent geometric objects such as points, lines, and polygons in spatial databases,

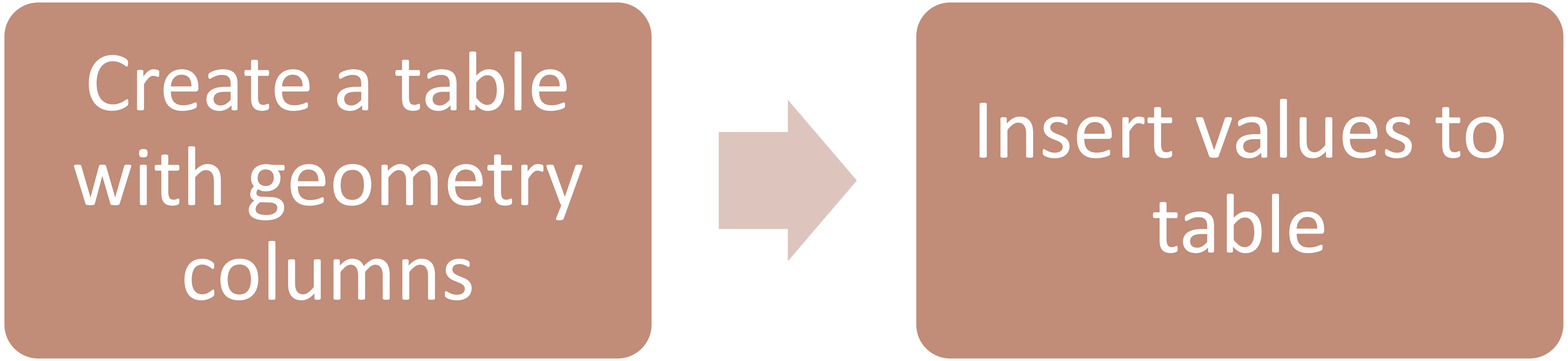
- PostGIS supports WKB as part of the Open Geospatial Consortium (OGC) standards, allowing spatial data to be stored, retrieved, and processed efficiently.

01010000009B559FABADF451C0E3A59BC420204540

Component	Value	Explanation
Byte Order	01	Little-endian
Geometry Type	01000000	POINT (1)
X Coordinate	9B559FABADF451C0	Longitude (-71.8231)
Y Coordinate	E3A59BC420204540	Latitude (42.2510)

3.1.2 CREATE MULTIPOINT

CREATE TABLE WITH MULTIPOINT IN POSTGIS



Create a table with geometry columns

Insert values to table

CREATE MULTIPOINT GEOMETRY

```
CREATE TABLE ch03.restaurants (
    id serial PRIMARY KEY,
    name varchar(50),
    geom geometry(MULTIPOINT, 4326));
```

INSERT VALUES TO TABLE

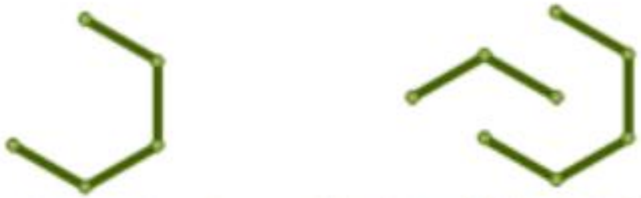
```
INSERT INTO ch03.restaurants (name, geom)

VALUES

('BBQ', ST_GeomFromText('MULTIPOINT(-71.824 42.249,-71.8256 42.2486,
-71.8268 42.2479)', 4326));
```

SUBTYPE OF GEOMETRY - LINESTRINGS

- A linestring is a path between locations. It takes the form of an ordered series of two or more points.
- Roads and rivers are typically represented as linestrings.
- A linestring is said to be closed if it starts and ends on the same point.
- It is said to be simple if it does not cross or touch itself (except at its endpoints if it is closed).
- A linestring can be both closed and simple.

SUBTYPE OF GEOMETRY - MULTILINESTRINGS

- Multilinestring is a collection of linestrings.



Simple non-closed linestring

Simple multilinestring defined by 4 endpoints of 2 elements

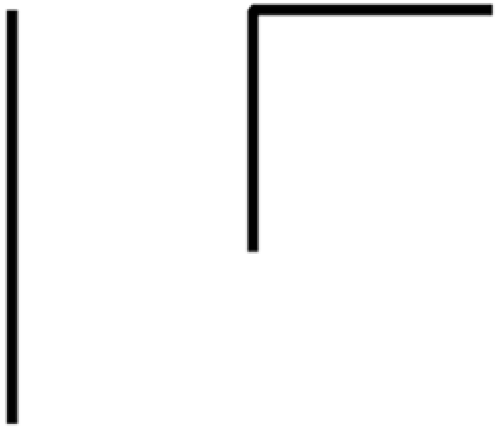


Figure 2.9 Multilinestrings

3.1.3 CREATE LINESTRINGS

CREATE A LINESTRINGS

```
CREATE TABLE ch03.streets (
id serial PRIMARY KEY,
name varchar(20),
line_str geometry(LINESTRING),
line_srid geometry(LINESTRING));

INSERT INTO ch03. streets(name, line_str, line_srid)
VALUES
('main', ST_GeomFromText('LINESTRING( -71.82359 42.24951, -71.82160 42.25056,
-71.81836 42.25227)'),
              ST_GeomFromText('LINESTRING( -71.82359 42.24951, -71.82160
42.25056, -71.81836 42.25227)', 4326)),
('str_squre', ST_GeomFromText('LINESTRING(-71.8267 42.2536, -71.8259 42.2544,
-71.8240 42.2530,  -71.8249 42.2523, -71.8267 42.2536)'),
                      ST_GeomFromText('LINESTRING(-71.8267 42.2536, -71.8259
42.2544,  -71.8240 42.2530,  -71.8249 42.2523,  -71.8267 42.2536)', 4326));
```

3.1.4 CREATE MULTILINESTRINGS

CREATE MULTILINESTRING

```
DROP TABLE IF EXISTS ch04.multi_street;
CREATE TABLE ch04.multi_street (
    id serial PRIMARY KEY,
    name varchar(20),
    line_str geometry(MULTILINESTRING),
    line_srid geometry(MULTILINESTRING, 4326)
);

INSERT INTO ch04.multi_street(name, line_str, line_srid)
VALUES
('multi_street',
    ST_GeomFromText('MULTILINESTRING((-71.82359 42.24951, -71.82160 42.25056, -71.81836 42.25227),
(-71.8267 42.2536, -71.8259 42.2544,  -71.8240 42.2530,  -71.8249 42.2523, -71.8267 42.2536))'),

    ST_GeomFromText('MULTILINESTRING((-71.82359 42.24951, -71.82160 42.25056, -71.81836 42.25227),
    (-71.8267 42.2536, -71.8259 42.2544,  -71.8240 42.2530,  -71.8249 42.2523, -71.8267 42.2536))',
4326)
);
```

SUBTYPE OF GEOMETRY - POLYGONS

- Closed linestrings are the building blocks of polygons.
- Polygon: Composed of one outer linear ring and optionally one or more inner rings.

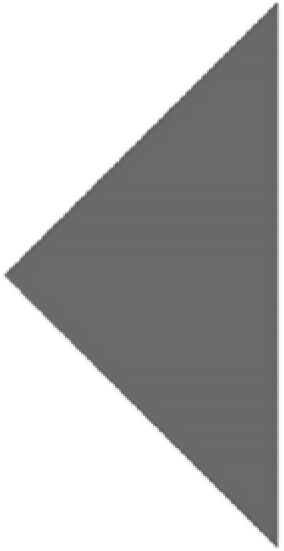


Figure 2.4
Triangular polygon

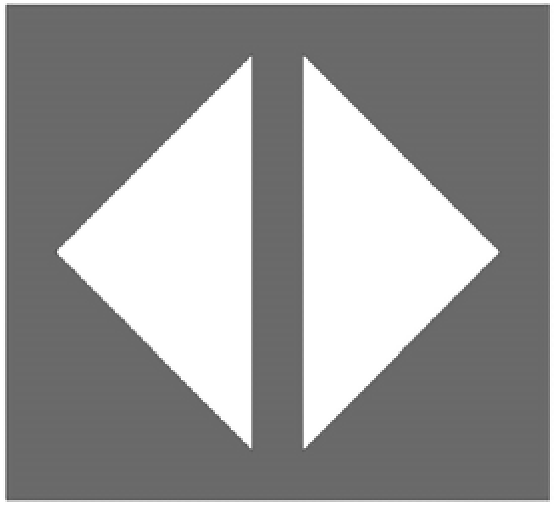
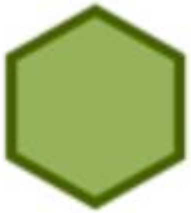


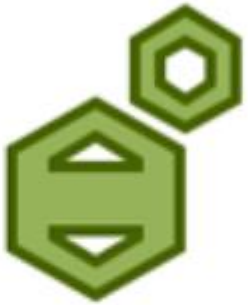
Figure 2.5 Polygon with interior rings (holes)

SUBTYPE OF GEOMETRY - MULTIPOLYGONS

- A polygon is a representation of an area.
- The outer boundary of the polygon is represented by a ring. This ring is a linestring that is both closed and simple as defined above.
- Holes within the polygon are also represented by rings.



Polygon defined
by exterior ring



Multipolygon consisting
of 2 elements defined
by exterior rings and 3 interior rings

GEOMETRYCOLLECTION

- The GEOMETRYCOLLECTION is a PostGIS geometry subtype that can contain heterogeneous geometries.

- Unlike multi-geometries, where the constituent geometries must be of the same subtype, GEOMETRYCOLLECTION can include points, linestrings, polygons, and their collection counterparts.

3.1.5 CREATE GEOMETRYCOLLECTION

CREATE GEOMETRYCOLLECTION

```
CREATE TABLE ch03.campus (
    id serial PRIMARY KEY,
    name varchar(50),
    geom geometry(GEOMETRYCOLLECTION, 4326));

INSERT INTO ch03.campus (name, geom)
VALUES
(   'campus_map',
    ST_GeomFromText(
        'GEOMETRYCOLLECTION(
        POLYGON((-71.8235 42.2510, -71.8229 42.2513, -71.8227 42.2510, -71.8233 42.2507,  -71.8235 42.2510)),
        LINESTRING(-71.8230 42.2509, -71.8223 42.2502),
        POINT(-71.8228 42.2508))',4326));
```

3.2 GEOGRAPHY

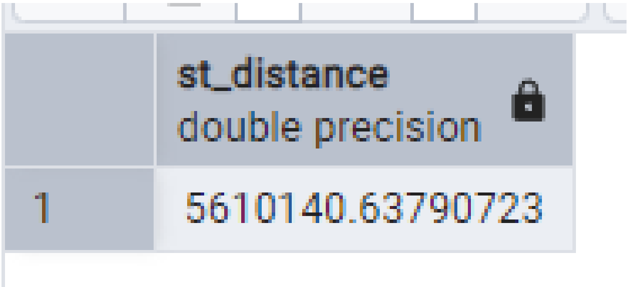
GEOGRAPHY

- `geography` starts by assuming that all your data is based on a geodetic coordinate system, specifically the WGS 84 lon/lat SRID of 4326.
- Unlike GEOMETRY, which assumes a flat plane, GEOGRAPHY accounts for the earth's curvature, making it more suitable for applications that span large geographic areas, such as tracking movement across regions or calculating great-circle distances.
- It specifies how spatial coordinates (such as longitude, latitude, or X/Y values) relate to the real world by defining the **coordinate system, projection, and datum**.

3.3 DIFFERENCE BETWEEN GEOGRAPHY AND GEOMETRY

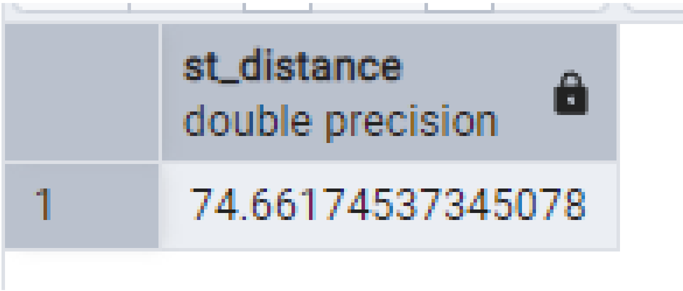
DISTANCE CALCULATION ON GEOMETRY AND GEOGRAPHY (FAR)

```
SELECT ST_Distance(
  'SRID=4326;POINT(-71.8011 42.2694)'::geography, -- Worcester
  'SRID=4326;POINT(2.5559 49.0083)'::geography     -- Paris
 );
```



	st_distance double precision
1	5610140.63790723

```
SELECT ST_Distance(
  'SRID=4326;POINT(-71.8011 42.2694)'::geometry, -- Worcester
  'SRID=4326;POINT(2.5559 49.0083)'::geometry     -- Paris
 );
```

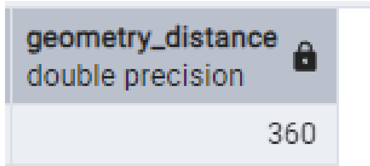


	st_distance double precision
1	74.6617453734507 8

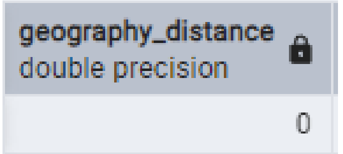
one degree is approximately 110.944 kilometers

DISTANCE BETWEEN NEAR LOCATIONS

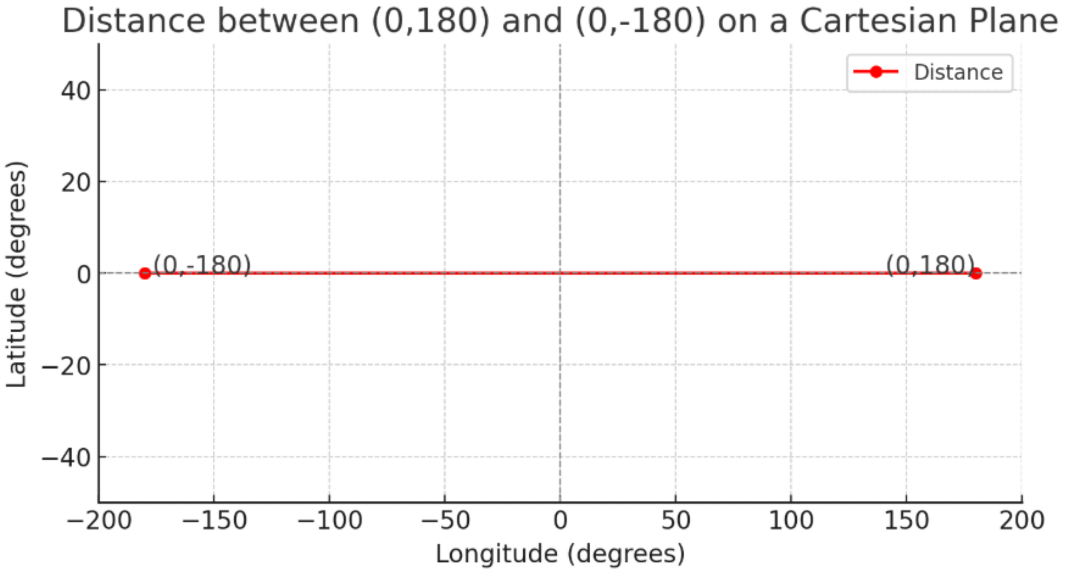
```
SELECT
   ST_Distance(ST_Point(0,180)::geography, ST_Point(0,-180)::geography) AS geography_distance,
   ST_Distance(ST_Point(0,180)::geometry, ST_Point(0,-180)::geometry) AS geometry_distance;
```

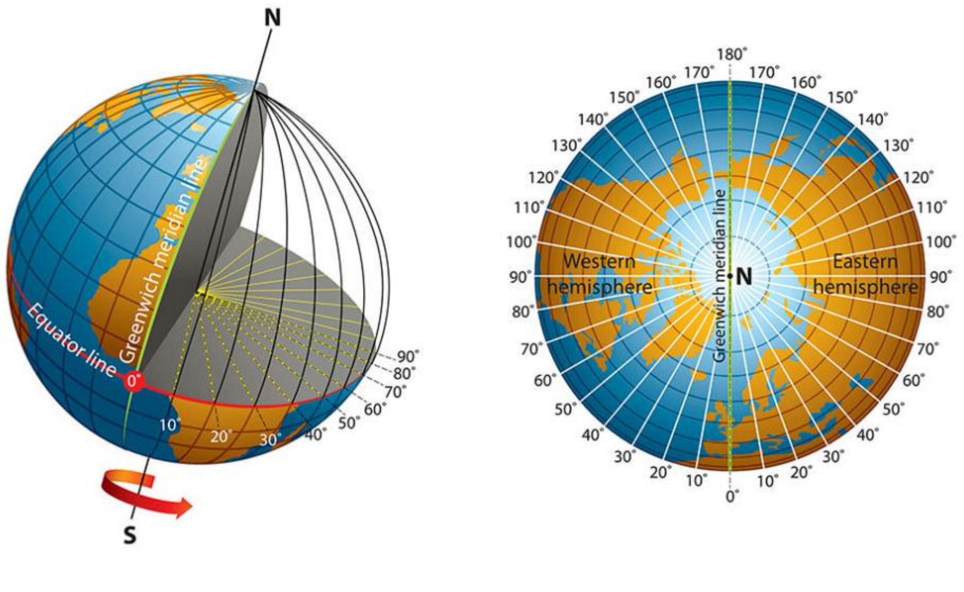


```
geometry_distance
double precision
                360
```



```
geography_distance
double precision
                  0
```





Longitude measures distance east or west of the prime meridian

CONCLUSION

- **Geography (Spherical Model) - More Accurate Distance Calculation:**
  - The geography data type treats coordinates as points on a spherical model of the Earth, **considering its curvature.**
  - When you use ::geography, it applies geodesic (great-circle) distance calculations, **which provide accurate real-world distances over large and small areas.**

- **Geometry (Planar Model) - Less Accurate for Larger Areas:**
  - The geometry data type assumes a flat Cartesian plane, which does not account for Earth's curvature.
  - The calculation treats latitude and longitude values as simple X-Y Cartesian coordinates (degrees), **which leads to distortion, especially for distances spanning larger areas or when further from the equator.**
  - Result: Distance in degrees, interpreted linearly in a flat space, leading to potential inaccuracies.

3.4 RASTER

RASTER

- Raster data represents geographic information using a grid of cells (pixels), where each cell has a value representing information such as elevation, land cover, or temperature.

- Common raster file formats: GeoTIFF, JPEG, PNG, ASCII Grid.

- Raster data is often used for continuous data representation, such as satellite imagery, terrain modeling, and environmental monitoring.

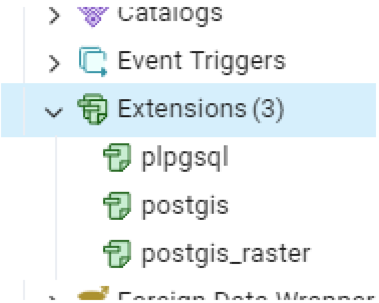
RASTER SUPPORT IN POSTGIS

- PostGIS extends PostgreSQL to support geographic objects, including raster data.
- Raster functionality in PostGIS allows storage, analysis, and manipulation of raster data within a spatial database.
- To use raster capabilities, PostGIS must be installed with raster support enabled.

3.4.1 CREATE RASTER

INSTALL POSTGIS_RASTER EXTENSION

```
CREATE EXTENSION postgis_raster;
```

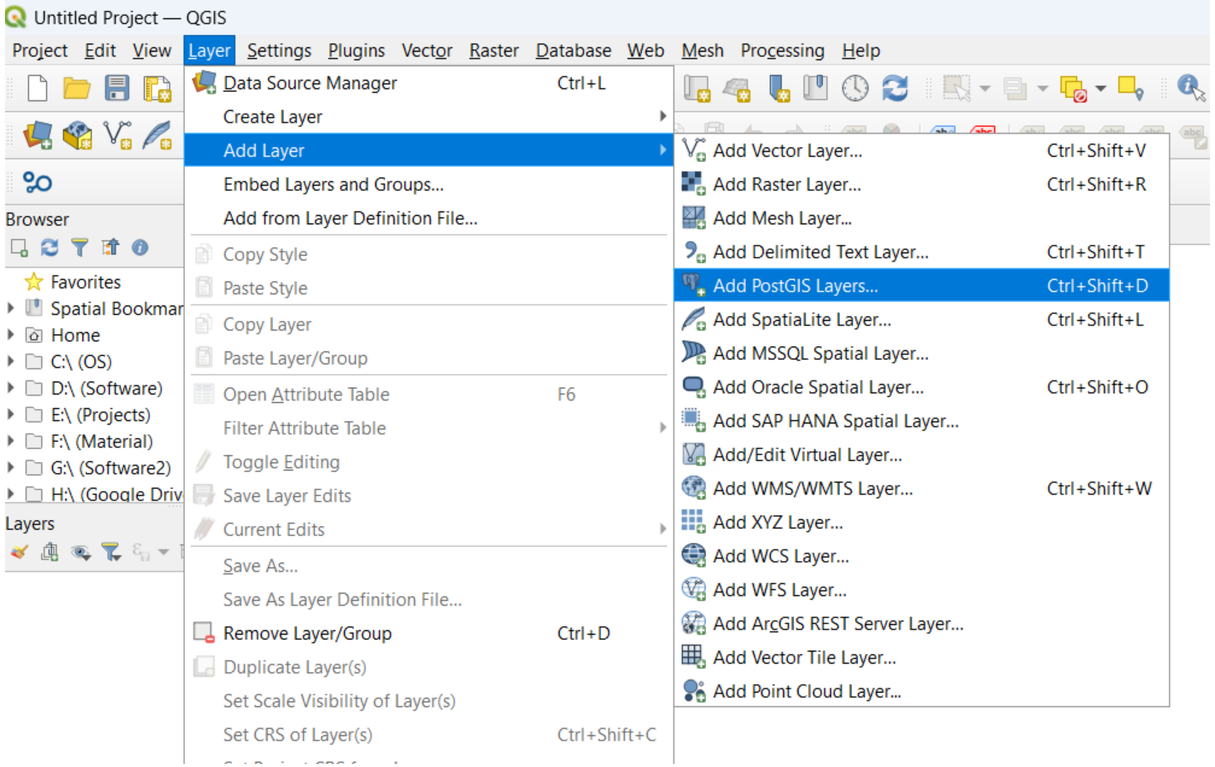


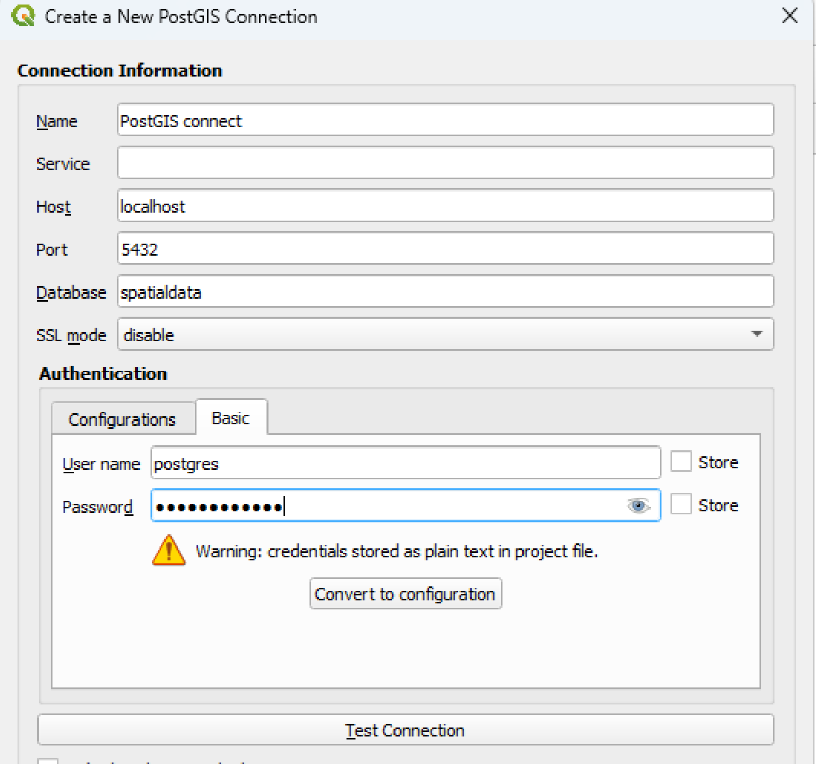
postgis_raster is used to create raster data from scratch and how to insert the data using SQL

CREATE RASTER

```
DROP TABLE IF EXISTS ch04.rasters02;
CREATE TABLE ch04.rasters02
   (rid SERIAL PRIMARY KEY,
     name varchar(255),
      rast raster);

INSERT INTO ch04.rasters02 (name, rast)
SELECT
'quad ' || x::text || ' ' || y::text,
ST_AddBand(
     ST_MakeEmptyRaster(
           100, 100,
           -71.824 + (x*0.01) ,42.249 - (y * 0.01),
           0.001, -0.001, 0, 0,4326),
           '16BUI'::text,1)
FROM generate_series(0,3) As x CROSS JOIN generate_series(0,3) As y;
```

3.4.2 CHECK RASTER IN QGIS


Untitled Project — QGIS
Project Edit View Layer Settings Plugins Vector Raster Database Web Mesh Processing Help
Data Source Manager Ctrl+L
Create Layer
Add Layer
Embed Layers and Groups...
Add from Layer Definition File...
Copy Style
Paste Style
Copy Layer
Paste Layer/Group
Open Attribute Table F6
Filter Attribute Table
Toggle Editing
Save Layer Edits
Current Edits
Save As...
Save As Layer Definition File...
Remove Layer/Group Ctrl+D
Duplicate Layer(s)
Set Scale Visibility of Layer(s)
Set CRS of Layer(s) Ctrl+Shift+C
Add Vector Layer... Ctrl+Shift+V
Add Raster Layer... Ctrl+Shift+R
Add Mesh Layer...
Add Delimited Text Layer... Ctrl+Shift+T
Add PostGIS Layers... Ctrl+Shift+D
Add SpatiaLite Layer... Ctrl+Shift+L
Add MSSQL Spatial Layer...
Add Oracle Spatial Layer... Ctrl+Shift+O
Add SAP HANA Spatial Layer...
Add/Edit Virtual Layer...
Add WMS/WMTS Layer... Ctrl+Shift+W
Add XYZ Layer...
Add WCS Layer...
Add WFS Layer...
Add ArcGIS REST Server Layer...
Add Vector Tile Layer...
Add Point Cloud Layer...
Browser
Favorites
Spatial Bookmar
Home
C:\ (OS)
D:\ (Software)
E:\ (Projects)
F:\ (Material)
G:\ (Software2)
H:\ (Google Driv
Layers


Create a New PostGIS Connection

Connection Information

Name: PostGIS connect

Service:

Host: localhost

Port: 5432

Database: spatialdata

SSL mode: disable

Authentication

Configurations | Basic

User name: postgres □ Store

Password: ●●●●●●●●●●●● □ Store

Warning: credentials stored as plain text in project file.

3.4.3 CHECK METADATA

METADATA TABLES

- PostGIS provides two tables to track and report on the geometry types available in a given database.
- The first table, *spatial_ref_sys*, defines all the spatial reference systems known to the database and will be described in greater detail later.
- The second table (actually, a view), *geometry_columns*, provides a listing of all “features” (defined as an object with geometric attributes), and the basic details of those features.

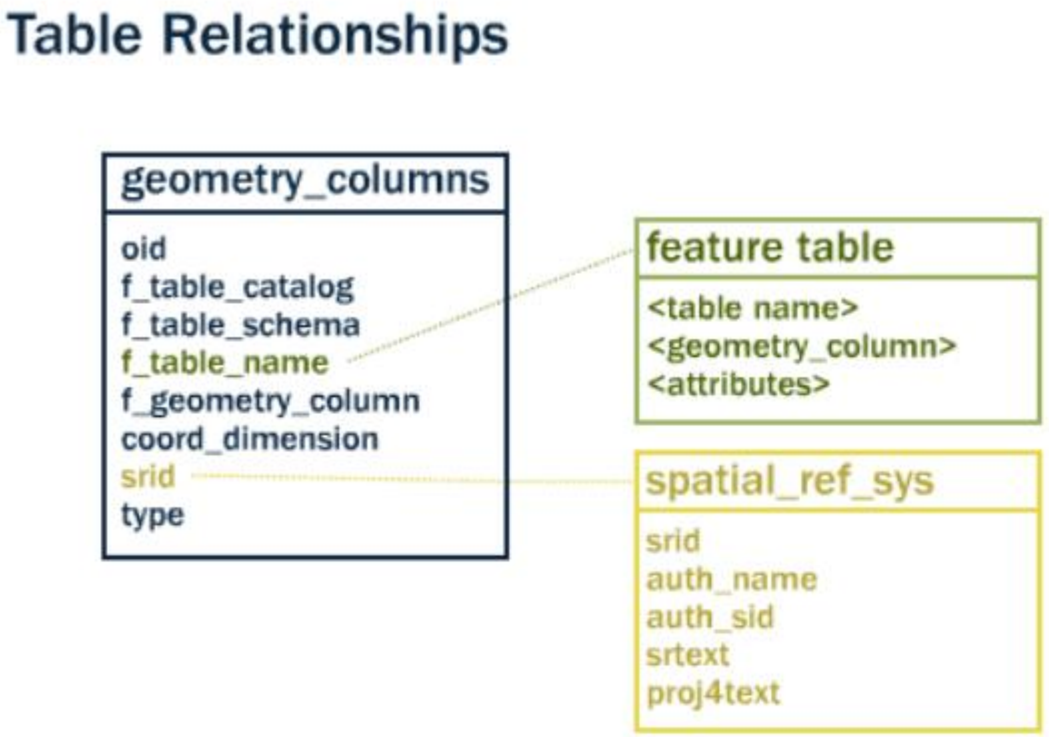


Table Relationships

geometry_columns

oid
f_table_catalog
f_table_schema
f_table_name
f_geometry_column
coord_dimension
srid
type

feature table

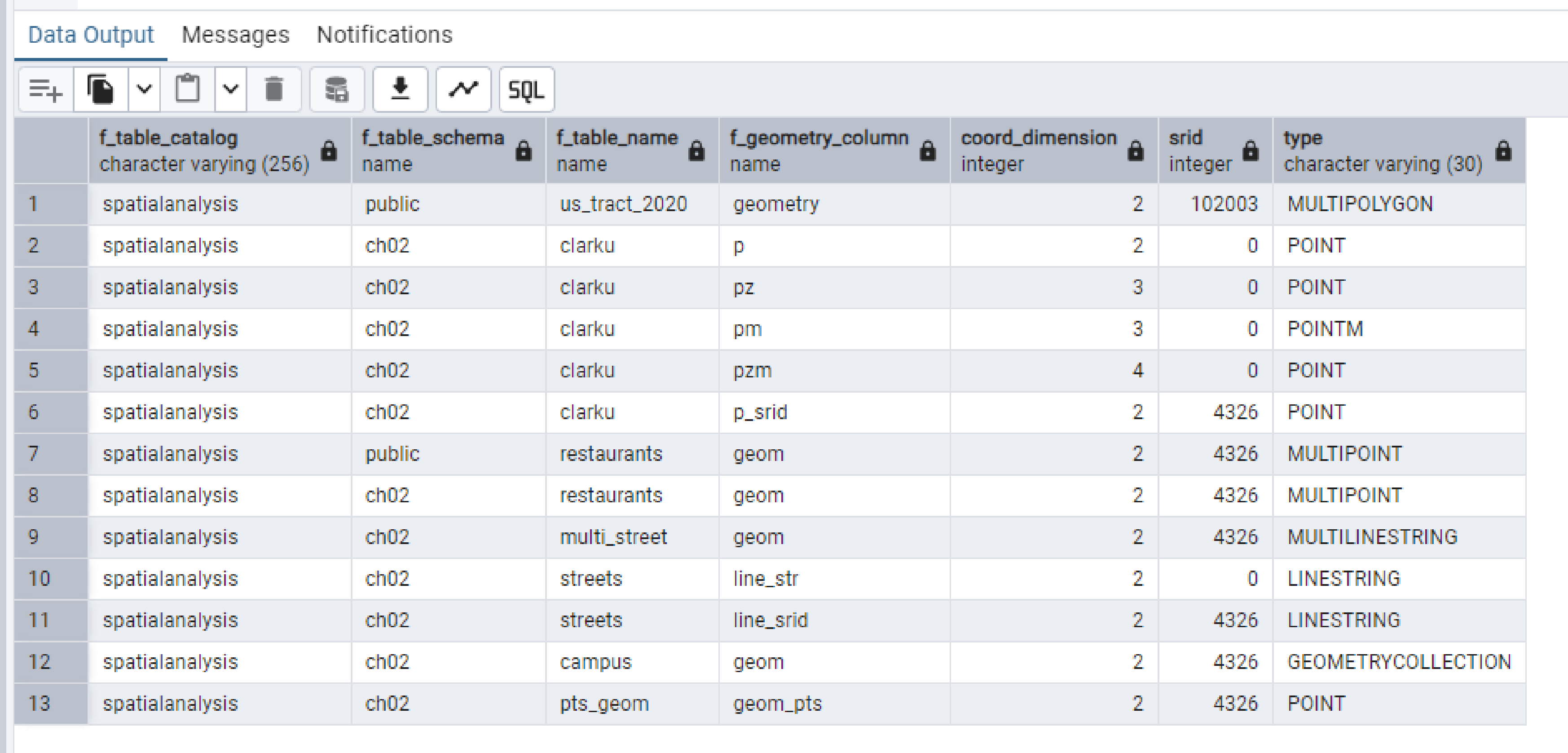
<table name>
<geometry_column>
<attributes>

spatial_ref_sys

srid
auth_name
auth_sid
srtext
proj4text

GEOMETRY_COLUMNS TABLE IN DATABASE

- SELECT * FROM geometry_columns;



	f_table_catalog	f_table_schema	f_table_name	f_geometry_column	coord_dimension	srid	type
	character varying (256)	name	name	name	integer	integer	character varying (30)
1	spatialanalysis	public	us_tract_2020	geometry	2	102003	MULTIPOLYGON
2	spatialanalysis	ch02	clarku	p	2	0	POINT
3	spatialanalysis	ch02	clarku	pz	3	0	POINT
4	spatialanalysis	ch02	clarku	pm	3	0	POINTM
5	spatialanalysis	ch02	clarku	pzm	4	0	POINT
6	spatialanalysis	ch02	clarku	p_srid	2	4326	POINT
7	spatialanalysis	public	restaurants	geom	2	4326	MULTIPOINT
8	spatialanalysis	ch02	restaurants	geom	2	4326	MULTIPOINT
9	spatialanalysis	ch02	multi_street	geom	2	4326	MULTILINESTRING
10	spatialanalysis	ch02	streets	line_str	2	0	LINESTRING
11	spatialanalysis	ch02	streets	line_srid	2	4326	LINESTRING
12	spatialanalysis	ch02	campus	geom	2	4326	GEOMETRYCOLLECTION
13	spatialanalysis	ch02	pts_geom	geom_pts	2	4326	POINT