SGD LAB EXP - 7

Name: Aditi Chhajed; Reg. No.: 221081009

Branch: IT; **Course Instructor**: Prof. Vedashree Awati

Aim:

Spatial Relationship Functions.

Theory:

1. ST_Contains()

- Checks if one geometry fully contains another geometry.
- <u>Example</u>: Imagine a park represented as a polygon and a bench within the park represented as a point. If you check whether the park polygon `ST_Contains()` the bench point, it would return `TRUE` because the point is entirely inside the park boundary.

2. ST_Within()

- The converse of `ST_Contains()`. It tests whether a geometry is fully within another geometry.
- <u>Example</u>: Consider a house (point) inside a city boundary (polygon). If you query whether the house `ST_Within()` the city, it would return `TRUE` since the house is entirely within the city's boundaries.

3. ST_Covers() and ST_CoveredBy()

- `ST_Covers()` checks if one geometry contains all points of another, including if they touch boundaries. `ST_CoveredBy()` checks if a geometry is covered by another.
- <u>Example</u>: If a walking trail (line) exactly aligns with a park boundary (polygon), `ST_Contains()` may return `FALSE` (because it touches but doesn't lie strictly inside). However, `ST_Covers()` would return `TRUE` since the park covers all points of the trail.

4. ST_Intersects()

- Tests if two geometries share any space, whether by overlapping, touching, or containment.
- <u>Example</u>: Two roads crossing at an intersection would return `TRUE` if tested with `ST_Intersects()` since they share common space where they cross.

5. ST_Disjoint()

Returns `TRUE` if two geometries share no space.

• <u>Example</u>: If a lake (polygon) and a mountain (polygon) do not touch or overlap, they are `ST_Disjoint()`, meaning they share no space and are separate.

6. ST_Overlaps()

- Determines if two geometries share some but not all points. They must have the same dimension and not be contained entirely within one another.
- <u>Example</u>: Two adjacent city districts with a common border but that do not fully contain each other would return `TRUE` for `ST_Overlaps()`.

7. ST_Touches()

- Returns `TRUE` if the geometries have at least one point in common but their interiors do not overlap.
- <u>Example</u>: Two countries sharing a border would return `TRUE` with `ST_Touches()`, as they meet along their boundary without sharing interior space.

8. ST_DWithin()

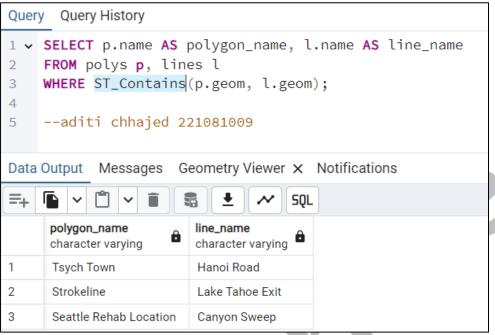
- Tests if two geometries are within a specified distance of one another.
- <u>Example</u>: Checking if a school (point) is within 500 meters of a park (polygon) would return `TRUE` if the distance is less than or equal to 500 meters.

9. ST_DFullyWithin()

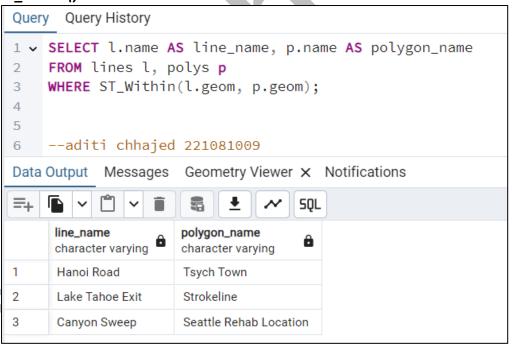
- Similar to `ST_DWithin()`, but every point of the first geometry must be within the distance of the second.
- <u>Example</u>: A river (line) must be entirely within 1 kilometer of a national park boundary (polygon) for `ST_DFullyWithin()` to return `TRUE`. If any part of the river extends beyond this distance, it returns `FALSE`.

Implementation:

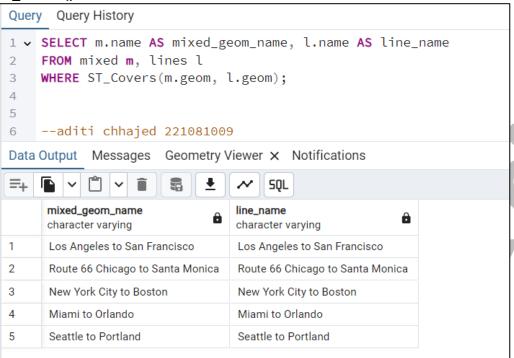
1. ST_Contains:



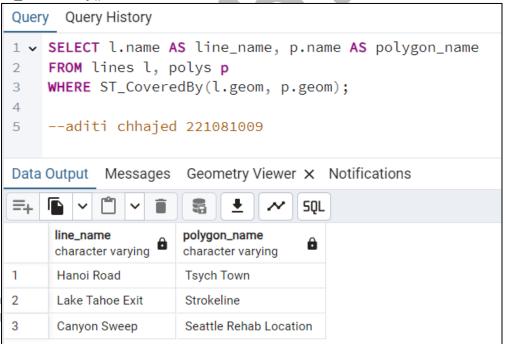
2. ST_Within():



3. ST_Covers():



4. ST_CoveredBy():



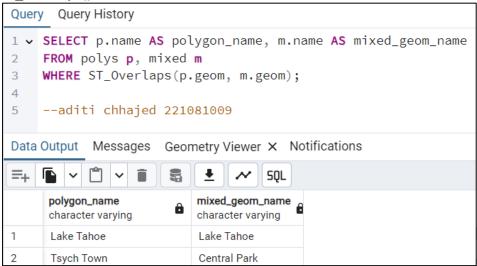
5. ST_Intersects():

```
Query Query History
1 		 SELECT l.name AS line_name, p.name AS polygon_name
     FROM lines l, polys p
2
     WHERE ST_Intersects(l.geom, p.geom);
3
4
5
     --aditi chhajed 221081009
Data Output Messages Geometry Viewer X Notifications
                                          SQL
=+
      line_name
                            polygon_name
      character varying
                            character varying
1
      New York City to Boston
                             New York Community Park
      Miami to Orlando
2
                             Miami-Orlando Line
                             Central Park
      Hanoi Road
3
4
      Hanoi Road
                             Tsych Town
5
      Lake Tahoe Exit
                             Lake Tahoe
6
      Lake Tahoe Exit
                             Strokeline
7
      Canyon Sweep
                             Seattle Rehab Location
```

6. ST_Disjoint():

```
Query Query History
1 v SELECT pt.name AS point_name
2
    FROM pts pt
3
    WHERE NOT EXISTS (
4
         SELECT 1
5
         FROM lines l
         WHERE ST_Disjoint(pt.geom, l.geom) = FALSE
6
7
    );
8
    --aditi chhajed 221081009
Data Output Messages Geometry Viewer X Notifications
                                       SQL
     point_name
     character varying
     Central Perk
1
2
     Grand Canyon Entry
3
     Houston
4
     Lake Canoe
```

7. ST_Overlaps():



8. ST_Touches():



9. ST_Dwithin():

36

Central Perk

Data	Output Messages	Ge	eometry Viewer × Notifications	
=+		750	<u> </u>	
	point_name character varying		e_name aracter varying	
1	New York		s Angeles to San Francisco	
2	New York		Route 66 Chicago to Santa Monica	
3	New York		New York City to Boston	
4	New York		Miami to Orlando	
5	New York		Seattle to Portland	
6	New York		Hanoi Road	
7	New York		ardstone Highway	
8	New York		ke Tahoe Exit	
9	New York		acadamia	
10			nyon Sweep	
11			Salem Park	
12	_		s Angeles to San Francisco	
13	-		oute 66 Chicago to Santa Monica	
14	Los Angeles New York City to Boston			
15	Los Angeles	Mi	ami to Orlando	
	point_name character varying	â	line_name character varying	
16	Los Angeles		Seattle to Portland	
17	Los Angeles		Hanoi Road	
18	Los Angeles		Hardstone Highway	
19	Los Angeles		Lake Tahoe Exit	
20	Los Angeles		Macadamia	
21	Los Angeles		Canyon Sweep	
22	Los Angeles		Salem Park	
23	Chicago		Los Angeles to San Francisco	
24	Chicago		Route 66 Chicago to Santa Monica	
25	Chicago		New York City to Boston	
26	Chicago		Miami to Orlando	
27	Chicago		Seattle to Portland	
28	Chicago		Hanoi Road	
29	Chicago		Hardstone Highway	
30	Chicago		Lake Tahoe Exit	
31	Chicago		Macadamia	
32	Chicago		Canyon Sweep	
33	Chicago		Salem Park	
34	Central Perk		Los Angeles to San Francisco	
35	Central Perk		Route 66 Chicago to Santa Monica	
			- ·	

New York City to Boston



10. ST_DFullyWithin():

```
Query History
Query
1 		 SELECT l.name AS line_name, p.name AS polygon_name
2
     FROM lines l, polys p
     WHERE ST_DFullyWithin(l.geom, p.geom, 200);
3
4
     --aditi chhajed 221081009
5
              Messages Geometry Viewer x Notifications
Data Output
=+
                                              SQL
       line_name
                                        polygon_name
                                                                â
       character varying
                                        character varying
1
       Los Angeles to San Francisco
                                        Central Park
2
       Los Angeles to San Francisco
                                        Lake Tahoe
3
       Los Angeles to San Francisco
                                        Yellowstone National Park
       Los Angeles to San Francisco
                                        Grand Canyon National Park
4
5
       Los Angeles to San Francisco
                                        Acadia National Park
       Los Angeles to San Francisco
                                        Tsych Town
6
7
       Los Angeles to San Francisco
                                        New York Community Park
       Los Angeles to San Francisco
                                        Seattle-Portland Dockyard
8
9
       Los Angeles to San Francisco
                                        Strokeline
       Los Angeles to San Francisco
                                        Seattle Rehab Location
10
11
       Los Angeles to San Francisco
                                        Miami-Orlando Line
                                        Central Park
12
       Route 66 Chicago to Santa Monica
                                        Lake Tahoe
13
       Route 66 Chicago to Santa Monica
14
       Route 66 Chicago to Santa Monica
                                        Yellowstone National Park
15
       Route 66 Chicago to Santa Monica
                                        Grand Canyon National Park
16
       Route 66 Chicago to Santa Monica
                                        Acadia National Park
Total rows: 121 of 121 Query complete 00:00:00.123
                                                            Ln 4, Col 1
```

Putting all 121 results is out of the scope of this pdf.

Conclusion:

In summary, using PostGIS spatial relationship functions allowed me to analyze and understand how different geographic features relate to one another.

By using functions like `ST_Contains()` and `ST_Within()`, I can determine whether a specific geometry lies entirely within another.

With functions like `ST_Intersects()` or `ST_Overlaps()`, I can identify shared spaces or overlapping areas.

These tools helped me explore boundaries, distances, adjacency, and coverage between geographic entities.

Whether I need to check if a point is inside a polygon, find features within a specific distance, or identify touching boundaries, these functions provide me with powerful ways to analyze and interact with spatial data for insightful results and meaningful spatial relationships.

