Lab 2
Spatial Operators

Learning objectives:

- ✓ Get to know spatial operators in postgreSQL/PostGIS and QGIS

Pure textual queries and create table/view statements can only be run from the **SQL-Shell**.

Queries returning spatial data or a mixture of spatial and textual data must be run in **QGIS**. Note queries executed from QGIS will not return a 'text only' result.

This lab investigates the semantics of the following operators:

- `ST_Union`
- `ST_Intersection`

There are two `ST_UNION` variations:

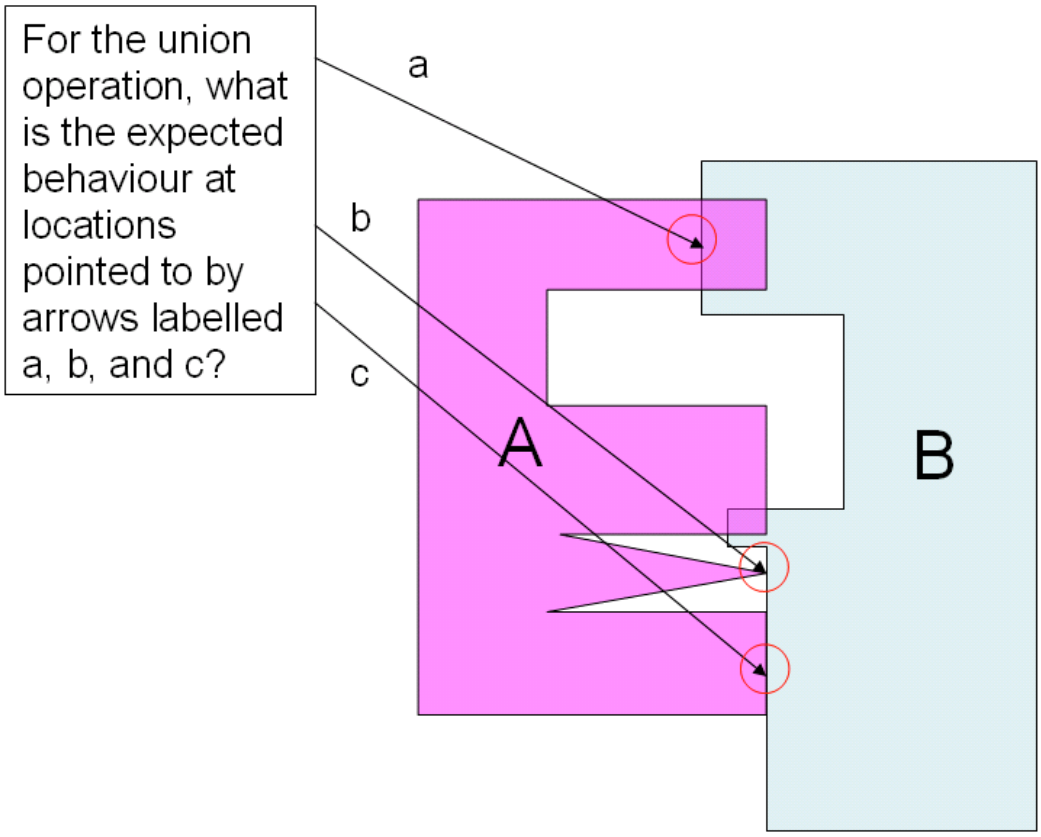
- Variant 1 unions **2** geometries resulting in a new geometry with no intersecting regions.
- Variant 2 is an aggregate function that takes a **set** of geometries and combines them into a single `ST_Geometry` object resulting in no intersecting regions.

We will focus on variant 1, which uses 2 geometry arguments
When you have completed this lab, try to develop similar queries for ST_Intersection.

The ST_UNION operator is seen as an aggregate function in postgreSQL. You have seen aggregate functions before in relational DBMS. Examples are `sum(), count(), min(),` etc.

Lab 2
Spatial Operators

I. Unions and Polygons



1. To draw a polygon in QGIS with coordinates that you specify, you can use ST_GeomFromEWKT in QGIS, go through Database > DB Manager… (for this to work you need to establish the connection between QGIS and your PostGIS instance first). In the window that opens, click on the SQL window button on the top left (2nd icon from left) and type the following into the query window:

   ```
   SELECT ST_GeomFromEWKT('POLYGON((10 20, 40 20, 40 30, 20 30, 40 35, 20 40, 40 40, 40 50, 20
   50, 20 60, 70 60, 70 70, 20 70, 10 70, 10 20))');
   ```

   Click Execute (F5), have load as new layer checked, set the geometry column field to st_geomfromewkt and click Load Now, accept the default coordinate system shown! The Shape A from the diagram will appear in your QGIS window.

   To have QGIS show you the 2nd shape B, you can repeat the process with:

   ```
   SELECT ST_GeomFromEWKT('POLYGON((40 00, 80 00, 80 80, 35 80, 35 55, 70 55, 70 45, 35 45, 35
   41, 40 41, 40 00))');
   ```

   Notice how QGIS displays Query Layer and Query Layer 2 in the Layer panel.

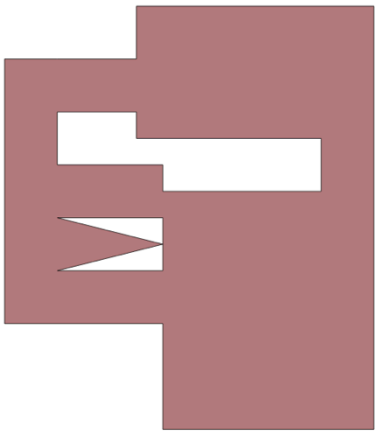
2. Now, let's create a UNION of the two shapes:
   In the query window, write the following:

   ```
   SELECT
   ST_Union(
   ST_GeomFromEWKT('POLYGON((10 20, 40 20, 40 30, 20 30, 40 35, 20 40, 40 40, 40 50, 20 50, 20
   60, 70 60, 70 70, 20 70, 10 70, 10 20))'),
   ```

Lab 2
Spatial Operators

```
    ST_GeomFromEWKT('POLYGON((40 00, 80 00, 80 80, 35 80, 35 55, 70 55, 70 45, 35 45, 35 41, 40
41, 40 00))') );
```

These are our two shapes. Remember to select st_union in the geometry field after hitting Execute (F5). Load the Geometry. And display the geometry:



You might have to de-select the other layers to be able to view this shape properly.

3. Go to pgAdmin 4 or your favourite shell to interact with your postGIS database. In the query window, type

```
SELECT
ST_Union(
ST_GeomFromEWKT('POLYGON((10 20, 40 20, 40 30, 20 30, 40 35, 20 40, 40 40, 40 50, 20 50, 20
60, 70 60, 70 70, 20 70, 10 70, 10 20))'),
ST_GeomFromEWKT('POLYGON((40 00, 80 00, 80 80, 35 80, 35 55, 70 55, 70 45, 35 45, 35 41, 40
41, 40 00))') );
```

This is the same SQL command as before. What does the output say?
Change the command to include:

```
SELECT ST_AsText(
ST_Union(
ST_GeomFromEWKT('POLYGON((10 20, 40 20, 40 30, 20 30, 40 35, 20 40, 40 40, 40 50, 20 50, 20
60, 70 60, 70 70, 20 70, 10 70, 10 20))'),
ST_GeomFromEWKT('POLYGON((40 00, 80 00, 80 80, 35 80, 35 55, 70 55, 70 45, 35 45, 35 41, 40
41, 40 00))') ));
```

Investigate the result. Is it as expected?

4. Staying in the shell for a moment, get the number of perimeter (or rings):

```
SELECT ST_NRings(ST_Union(
ST_GeomFromEWKT('POLYGON((10 20, 40 20, 40 30, 20 30, 40 35, 20 40, 40 40, 40 50, 20 50, 20
60, 70 60, 70 70, 20 70, 10 70, 10 20))'),
ST_GeomFromEWKT('POLYGON((40 00, 80 00, 80 80, 35 80, 35 55, 70 55, 70 45, 35 45, 35 41, 40
41, 40 00))') ));
```

Of course you can do that in QGIS too, but it won't show you the result on the map.

Can you explain how the following select differs from the previous one?
(Hint: look at the PostGIS manual http://postgis.net/docs/manual-1.5/)

```
SELECT ST_NumInteriorRings(ST_Union(
ST_GeomFromEWKT('POLYGON((10 20, 40 20, 40 30, 20 30, 40 35, 20 40, 40 40, 40 50, 20 50, 20
60, 70 60, 70 70, 20 70, 10 70, 10 20))'),
ST_GeomFromEWKT('POLYGON((40 00, 80 00, 80 80, 35 80, 35 55, 70 55, 70 45, 35 45, 35 41, 40
41, 40 00))') ));
```

5. Does the shape that you created contain the 'POINT(65 25)'? To find out run the following in the **SQL-Shell**.

```
select st_contains(
```

Lab 2
Spatial Operators

```
ST_Union(ST_GeomFromEWKT('POLYGON((10 20, 40 20, 40 30, 20 30, 40 35, 20 40, 40 40, 40 50, 20
50, 20 60, 70 60, 70 70, 20 70, 10 70, 10 20))'),
ST_GeomFromEWKT('POLYGON((40 00, 80 00, 80 80, 35 80, 35 55, 70 55, 70 45, 35 45, 35 41, 40
41, 40 00))') ),ST_GeomFromEWKT('POINT(65 25)'));
```

Does the shape that you created contain the 'POINT(50 50)'? Run the appropriate query in in the **SQL-Shell** and explain the result.

6. Go to QGIS,
   - Enable only the layers that show your two initial shapes
   - In the menu go to Vector – Geoprocessing Tools – Intersection…
   - In the menu choose your two layers and click run

   - Now run the union, try the other operators
   - Notice how a new layer opens for each result

II. Other Unions

1. Points

Duplicate points are removed:

```
SELECT ST_AsText(ST_Union(ST_GeomFromText('POINT(1 2)'),
        ST_GeomFromText('POINT(-2 3)') ) );
```

- Note that if you run this from QGIS, you don't need ST_AsText. Why?

result

```
MULTIPOINT(-2 3,1 2)
```

```
SELECT ST_AsText(ST_Union(ST_GeomFromText('POINT(1 2)'),
        ST_GeomFromText('POINT(1 2)') ) );
```

result

```
POINT(1 2)
```

2. Union handles Lines

```
SELECT ST_AsText(ST_Union(ST_GeomFromEWKT('LINESTRING(5 5, 10 10 )'),
ST_GeomFromEWKT('LINESTRING(5 5, 10 10)')));
```

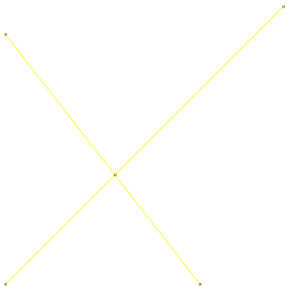
result

```
LINESTRING(5 5,10 10)
```

TIP: if you are viewing the information in QGIS, and you find the result does not centre, use the following icon from the top icon bar, in the middle:

3. Union and Intersection for intersecting Lines

```
SELECT st_asbinary(ST_union(ST_GeomFromEWKT('LINESTRING(0 0, 10 10 )'),
ST_GeomFromEWKT('LINESTRING(0 9, 7 0)')));
```



Lab 2
Spatial Operators

If we type the text version into the **SQL-Shell** we get the following;

```
SELECT ST_AsText(ST_union(ST_GeomFromEWKT('LINESTRING(0 0, 10 10 )'),
ST_GeomFromEWKT('LINESTRING(0 9, 7 0)')));

MULTILINESTRING((0 0,3.9375 3.9375),(3.9375 3.9375,10 10),(0 9,3.9375 3.9375),(
3.9375 3.9375,7 0))
```

What do we get when we intersect the same data?

```
SELECT ST_intersection(ST_GeomFromEWKT('LINESTRING(0 0, 10 10 )'), ST_GeomFromEWKT('LINESTRING(0
0, 10 10)'));
```

Try these queries in the **SQL-Shell** and with **QGIS**, Remember though that for the shell you need to show the result through ST_ASTEXT.

```
SELECT ST_Intersects(ga,gb) from
(SELECT ST_GeomFromText('LINESTRING (40 40, 180 180)') as ga,
ST_GeomFromText('LINESTRING (120 120, 20 200) ') as gb) as foo;

-- With geometry in QGIS, with just intersection point
SELECT ST_Intersects(ga,gb), ST_Intersection(ga,gb) from
(SELECT ST_GeomFromText('LINESTRING (40 40, 180 180)') as ga,
ST_GeomFromText('LINESTRING (120 120, 20 200) ') as gb) as foo;
```

- Hint: choose the right intersection geometry column!

4. Union handles duplicate polygons

Try the following in postgis shell and in QGIS (remember about no need for ST_ASTEXT in QGIS):

```
SELECT ST_AsText(
ST_Union(ST_GeomFromEWKT(
'POLYGON((10 10, 20 10, 20 20, 10 20, 10 10 ))'),
ST_GeomFromEWKT('POLYGON((10 10, 20 10, 20 20, 10 20, 10 10))')));
```

result
```
POLYGON((20 10,10 10,10 20,20 20,20 10));
```

5. Union with one point touching give multipolygon.
```
SELECT ST_AsText(
ST_Union(ST_GeomFromEWKT(
'POLYGON((10 10, 20 10, 20 20, 10 20, 10 10 ))'),
ST_GeomFromEWKT('POLYGON((20 20,40 20, 40 40,20 40, 20 20))')));
```

result
```
MULTIPOLYGON(((20 20,20 10,10 10,10 20,20 20)),((20 20,20 40,40 40,40 20,20 20)
))
```
Remember about the “Full Zoom” button in QGIS to rescale.

6. Union with one line in common, touching line removed
```
SELECT ST_AsText(
ST_Union(ST_GeomFromEWKT(
'POLYGON((10 10, 20 10, 20 20, 10 20, 10 10 ))'),
ST_GeomFromEWKT('POLYGON((20 10,40 10, 40 20, 20 20, 20 10))')));
```

result
```
POLYGON((20 10,10 10,10 20,20 20,40 20,40 10,20 10))
```

7. Union with overlap: both overlapping lines removed
```
SELECT ST_AsText(
ST_Union(ST_GeomFromEWKT(
'POLYGON((10 10, 20 10, 20 20, 10 20, 10 10 ))'),
ST_GeomFromEWKT('POLYGON((19 10,40 10, 40 19, 19 19, 19 10))')));
```

Lab 2
Spatial Operators

result
```
POLYGON((19 10,10 10,10 20,20 20,20 19,40 19,40 10,20 10,19 10))
```

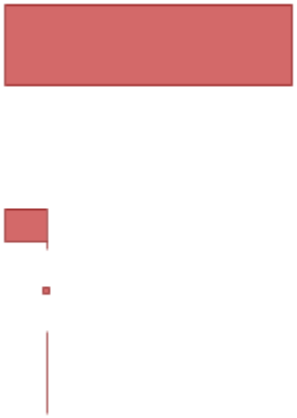
8. Union with disjoint, gives two polygons MULTIPOLYGON
```
SELECT ST_AsText(
ST_Union(ST_GeomFromEWKT(
'POLYGON((10 10, 20 10, 20 20, 10 20, 10 10 ))'),
ST_GeomFromEWKT('POLYGON((21 10,40 10, 40 21, 21 21, 21 10))')));
```

result
```
MULTIPOLYGON(((10 10,10 20,20 20,20 10,10 10)),((21 10,21 21,40 21,40 10,21 10))
```

III. ST_Intersection

Now try intersection of two polygons in QGIS.

```
SELECT
ST_intersection(
ST_GeomFromEWKT('POLYGON((10 20, 40 20, 40 30, 20 30, 40 35, 20 40, 40 40, 40 50, 20 50, 20 60, 70
60, 70 70, 20 70, 10 70, 10 20))'),
ST_GeomFromEWKT('POLYGON((40 00, 80 00, 80 80, 35 80, 35 55, 70 55, 70 45, 35 45, 35 41, 40 41, 40
00))') ));
```



IV. ST_Difference and ST_SymDifference

ST_Difference returns a geometry that represents that part of geometry A that does not intersect with geometry B. One can think of this as GeometryA - ST_Intersection(A,B). If A is completely contained in B then an empty geometry collection is returned.

```
SELECT ST_Difference(
                        ST_GeomFromText('LINESTRING(50 100, 50 200)'),
                        ST_GeomFromText('LINESTRING(50 50, 50 150)'));
```

ST_SymDifference — Returns a geometry that represents the portions of A and B that do not intersect. It is called a symmetric difference because ST_SymDifference(A,B) = ST_SymDifference(B,A).

```
--Safe for 2d - symmetric difference of 2 linestrings
SELECT ST_AsText(
        ST_SymDifference(
                ST_GeomFromText('LINESTRING(50 100, 50 200)'),
                ST_GeomFromText('LINESTRING(50 50, 50 150)')
        )
);
```

Change this query so that it can be executed in QGIS.

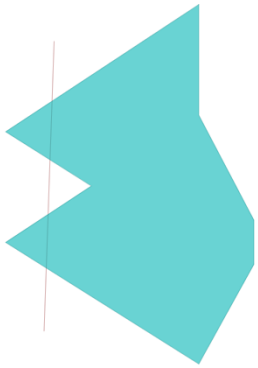
Lab 2

Spatial Operators

View geometry of a polygon and line using two SQL statements.
The SQL must be executed as two separate SQL queries in QGIS

```
SELECT ST_GeomFromText('POLYGON((2 4.5,3 2.6,3 1.8,2 0,-1.5 2.2,0.056 3.222,-1.5 4.2,2 6.5,2
4.5))');
SELECT ST_GeomFromText('LINESTRING(-0.62 5.84,-0.8 0.59)');
```

Gives the blue shape and red line below



The union of the above line and polygon give one object of type GEOMETRYCOLLECTION, as follows

Next we create a GEOMETRYCOLLECTION.

```
SELECT
 st_GeomFromText('GEOMETRYCOLLECTION (MULTIPOINT ((2 2), (3 4), (4 2), (2 1)),
LINESTRING (5 4, 6 2, 4 3, 3 2, 4 1),
POLYGON ((5 5, 5 6, 7 6, 7 5, 5 5)))');
```

How does ST_Difference work (geom1 – geom2)?
Subtract the line from the polygon

```
SELECT ST_Difference(g1.geom1, g1.geom2))
FROM (SELECT ST_GeomFromText('POLYGON((2 4.5,3 2.6,3 1.8,2 0,-1.5 2.2,0.056 3.222,-1.5 4.2,2 6.5,2
4.5))') As geom1,
ST_GeomFromText('LINESTRING(-0.62 5.84,-0.8 0.59)') As geom2) AS g1;
```

Swap the direction of subtraction (geom2 – geom1), subtract the polygon from the line.

```
SELECT ST_Difference(g1.geom2, g1.geom2)
FROM (SELECT ST_GeomFromText('POLYGON((2 4.5,3 2.6,3 1.8,2 0,-1.5 2.2,0.056 3.222,-1.5 4.2,2 6.5,2
4.5))') As geom1,
ST_GeomFromText('LINESTRING(-0.62 5.84,-0.8 0.59)') As geom2) AS g1;
```

V. Linear Interpolation

The next example shows how to construct a line from a table containing a set of points. We do not enforce any constraints on the POINT table. That is, we do not use the AddGeometryColumn function.

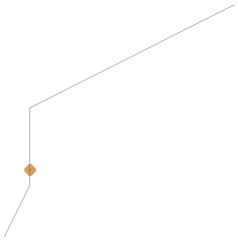
```
CREATE TABLE POINT (name varchar(1), g geometry);

INSERT INTO POINT (name, g) values (‘a’ , 'POINT(10 10)');
INSERT INTO POINT (name, g) values (‘b’ , 'POINT(20 30)');
INSERT INTO POINT (name, g) values (‘c’ , 'POINT(20 60)');
INSERT INTO POINT (name, g) values (‘d’ , 'POINT(100 100)');


select  st_astext(g) from POINT
```

Lab 2
Spatial Operators

```
select st_astext(ST_LineFromMultiPoint(st_collect(g)))
from (
select g
from POINT
order by name
) as line
```

Then try:

```
SELECT
ST_Line_Interpolate_Point(ST_LineFromMultiPoint(st_collect(g)),0.20)
FROM (select g FROM  POINT) as foo;
```



For this display both layers need to be activated. You get a point back.
See:
http://postgis.refractions.net/documentation/manual-2.0/ST_Line_Interpolate_Point.html
http://postgis.refractions.net/documentation/manual-2.0/ST_Line_Locate_Point.html