

Bazy danych przestrzennych – ćwiczenia z PostGIS raster

1. Utworzenie nowej bazy danych „postgis_raster”
2. Dodanie do bazy rozszerzenia „postgis” oraz „postgis_raster”
3. Załadowanie kopii bazy danych zawierającej niezbędną strukturę wraz z danymi (Restore -> „postgis_raster.backup”)
4. Zmiana nazwy schematu „schema_name” na „schema_piekarska”
5. Ładowanie danych za pomocą narzędzia raster2pgsql (wspomniane narzędzie stworzy nowy plik .sql, który zostanie załadowany do bazy danych)

Przykład 1 – ładowanie rastru przy użyciu pliku .sql

```
C:\"Program Files"\PostgreSQL\14\bin\raster2pgsql.exe -s 3763 -N -32767 -t 100x100 -I -C -M -d C:\Users\Anna\Desktop\Studia\s5\BDP\cw\cw6\Dane\rasters\srtm_1arc_v3.tif rasters.dem > C:\Users\Anna\Desktop\Studia\s5\BDP\cw\cw6\Dane\dem.sql
```

Przykład 2 – ładowanie rastru bezpośrednio do bazy

```
C:\"Program Files"\PostgreSQL\14\bin\raster2pgsql.exe -s 3763 -N -32767 -t 100x100 -I -C -M -d C:\Users\Anna\Desktop\Studia\s5\BDP\cw\cw6\Dane\rasters\srtm_1arc_v3.tif rasters.dem > C:\Users\Anna\Desktop\Studia\s5\BDP\cw\cw6\Dane\dem.sql | psql -d postgis_raster -h localhost -U postgres -p 5432
```

Przykład 3 – załadowanie danych landsat 8 o wielkości kafelka 128x128 bezpośrednio do bazy danych.

```
C:\"Program Files"\PostgreSQL\14\bin\raster2pgsql.exe -s 3763 -N -32767 -t 100x100 -I -C -M -d C:\Users\Anna\Desktop\Studia\s5\BDP\cw\cw6\Dane\rasters\Landsat8_L1TP_RGBN.TIF rasters.landsat8 > C:\Users\Anna\Desktop\Studia\s5\BDP\cw\cw6\Dane\landsat8.sql | psql -d postgis_raster -h localhost -U postgres -p 5432
```

6. Tworzenie rastrów z istniejących rastrów i interakcja z wektorami

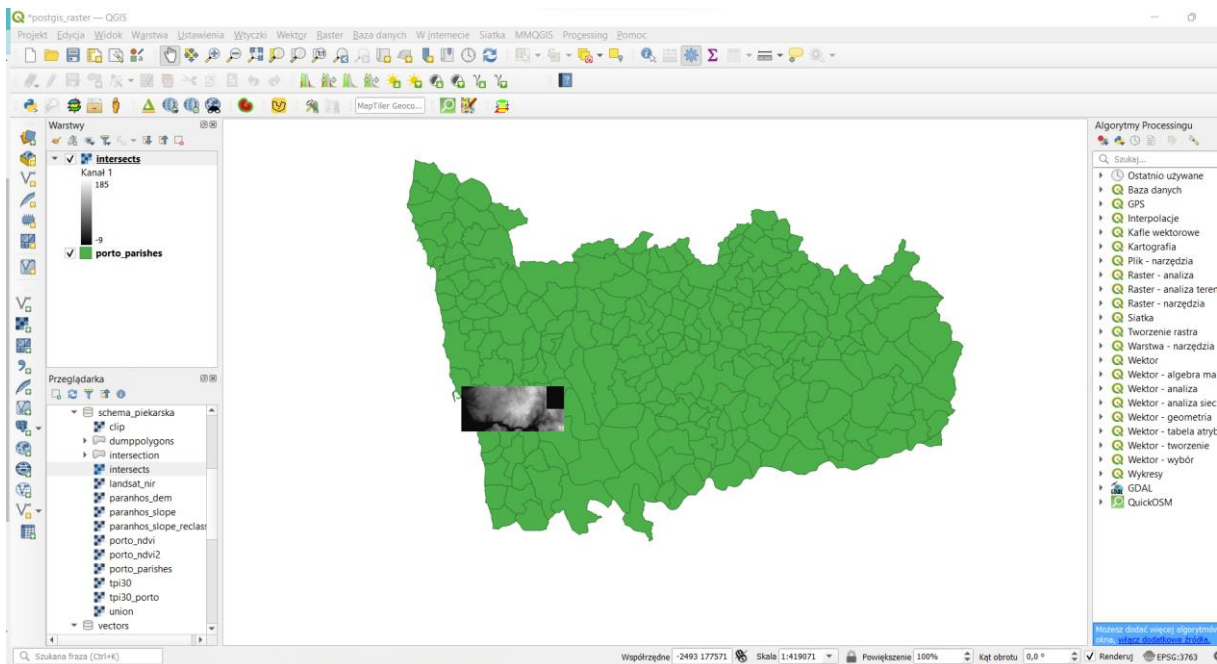
Przykład 1 - ST_Intersects

Przecięcie rastra z wektorem.

Pokazuje jak wyodrębnić kafelki nakładające się na geometrię. Opcjonalnie można utworzyć tabelę z wynikiem zapytania.

```
CREATE TABLE schema_piekarska.intersects AS  
SELECT a.rast, b.municipality
```

```
FROM rasters.dem AS a, vectors.porto_parishes AS b
WHERE ST_Intersects(a.rast, b.geom) AND b.municipality
ilike 'porto';
```



W przypadku tworzenia tabel zawierających dane rastrowe sugeruje się wykonanie poniższych kroków:

1. Dodanie serial primary key:

```
ALTER TABLE schema_piekarska.intersects
ADD COLUMN rid SERIAL PRIMARY KEY;
```

2. Utworzenie indeksu przestrzennego:

```
CREATE INDEX idx_intersects_rast_gist ON
schema_piekarska.intersects
USING gist (ST_ConvexHull(rast));
```

3. Dodanie raster constraints:

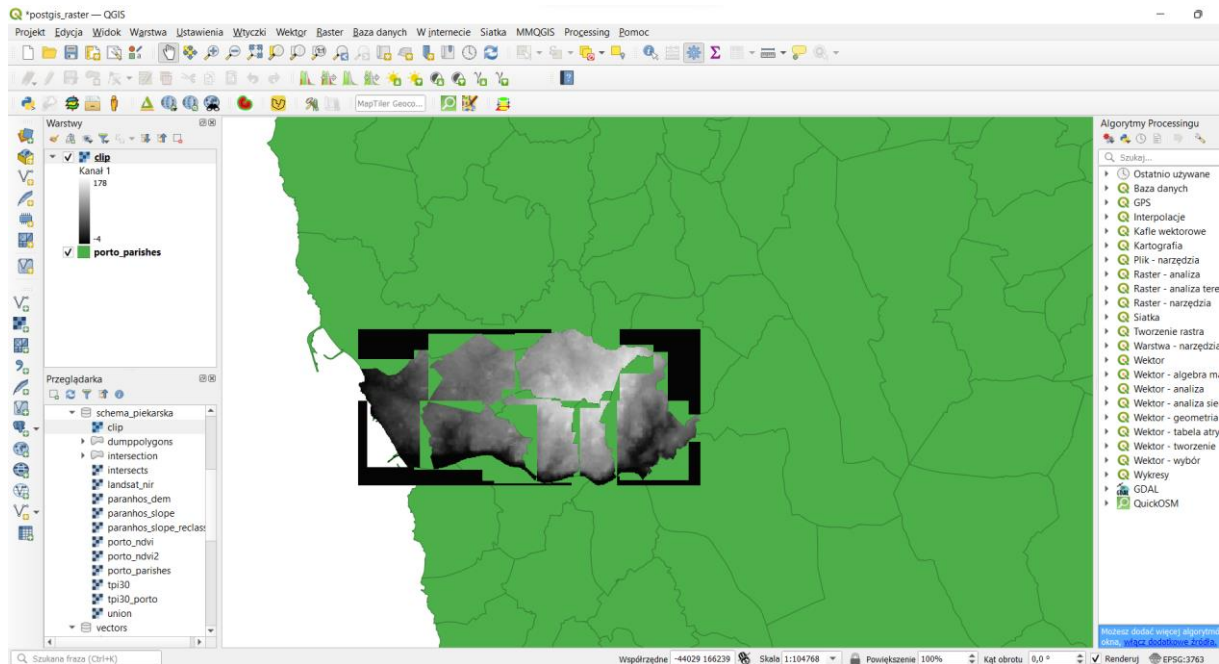
```
-- schema::name table_name::name raster_column::name
SELECT
AddRasterConstraints('schema_piekarska'::name,
'intersects'::name, 'rast'::name);
```

Przykład 2 - ST_Clip

Obcinanie rastra na podstawie wektora.

```
CREATE TABLE schema_piekarska.clip AS
SELECT ST_Clip(a.rast, b.geom, true), b.municipality
FROM rasters.dem AS a, vectors.porto_parishes AS b
```

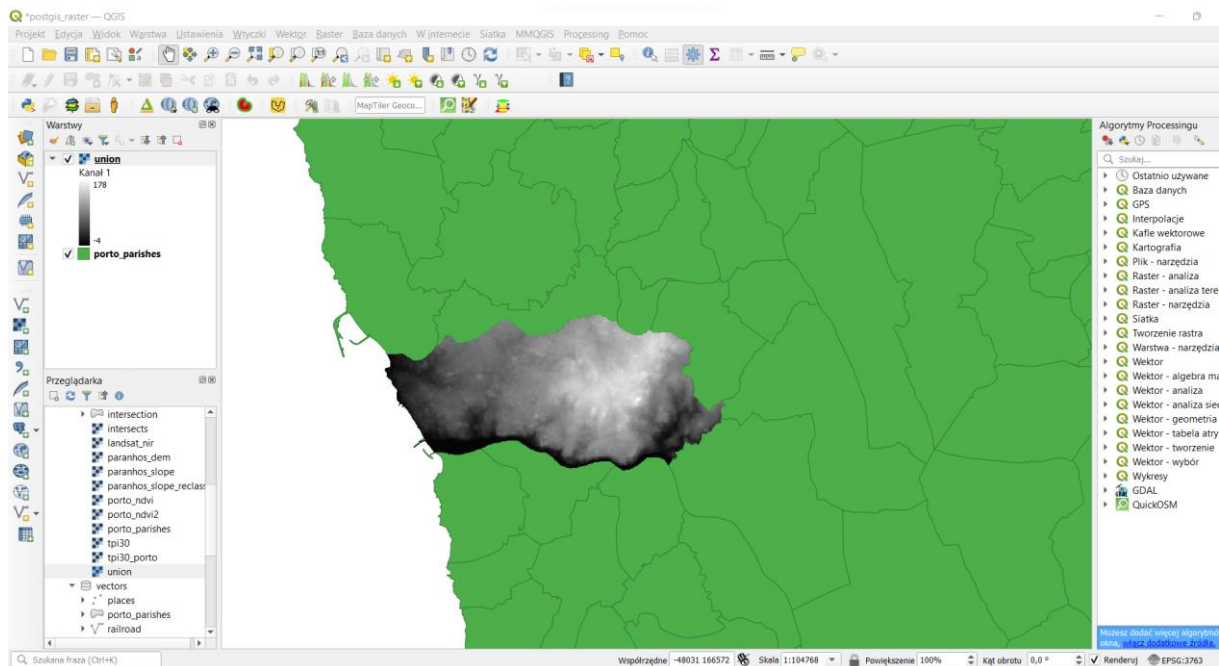
```
WHERE ST_Intersects(a.rast, b.geom) AND b.municipality
like 'PORTO';
```



Przykład 3 - ST_Union

Połączenie wielu kafelków w jeden raster.

```
CREATE TABLE schema_piekarska.union AS
SELECT ST_Union(ST_Clip(a.rast, b.geom, true))
FROM rasters.dem AS a, vectors.porto_parishes AS b
WHERE b.municipality ilike 'porto' and
ST_Intersects(b.geom,a.rast);
```



Oprócz powyższego przykładu, `st_union` pozwala również na operacje na nakładających się rastrach opartych na danej funkcji agregującej, a mianowicie `FIRST`, `LAST`, `SUM`, `COUNT`, `MEAN` lub `RANGE`. Na przykład, jeśli mamy wiele rastrów z danymi o opadach atmosferycznych i potrzebujemy średniej wartości, możemy użyć `st_union` lub `map_algebra`. Aby uzyskać więcej informacji na temat `st_union`, sprawdź dokumentację: https://postgis.net/docs/RT_ST_Union.html

7. Tworzenie rastrów z wektorów (rastrowanie)
Poniższe przykłady pokazują rastrowanie wektoru.

Przykład 1 - ST_AsRaster

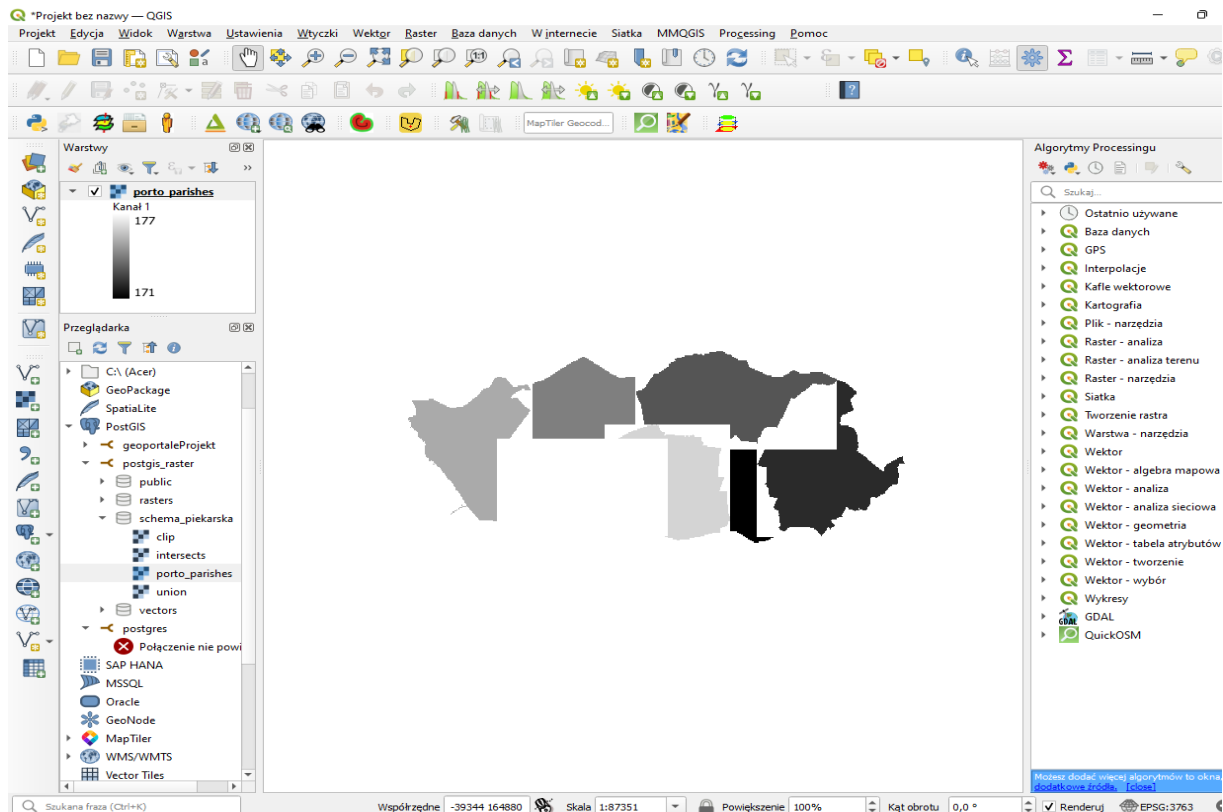
Przykład pokazuje użycie funkcji `ST_AsRaster` w celu rastrowania tabeli z parafiami o takiej samej charakterystyce przestrzennej tj.: wielkość piksela, zakresy itp.

```
CREATE TABLE schema_piekarska.porto_parishes AS WITH r
AS (SELECT rast
      FROM rasters.dem
      LIMIT 1 )
SELECT ST_AsRaster(a.geom,r.rast,'8BUI',a.id,-32767) AS
rast
FROM vectors.porto_parishes AS a, r
WHERE a.municipality ilike 'porto';
```

Przykładowe zapytanie używa piksela typu '8BUI' tworząc 8-bitową nieoznaczoną liczbę całkowitą (8- bit unsigned integer). Unsigned integer może reprezentować tylko nieujemne liczby całkowite; signed integer mogą również reprezentować liczby całkowite ujemne. Aby uzyskać więcej informacji o typach rastrowych PostGIS, zapoznaj się z dokumentacją:

https://postgis.net/docs/RT_ST_BandPixelType.html

Wynikowy raster to jedna parafia na rekord, na wiersz tabeli. Użyj QGIS lub ArcGIS do wizualizacji wyników.



Przykład 2 - ST_Union

Drugi przykład łączy rekordy z poprzedniego przykładu przy użyciu funkcji ST_UNION w pojedynczy raster.

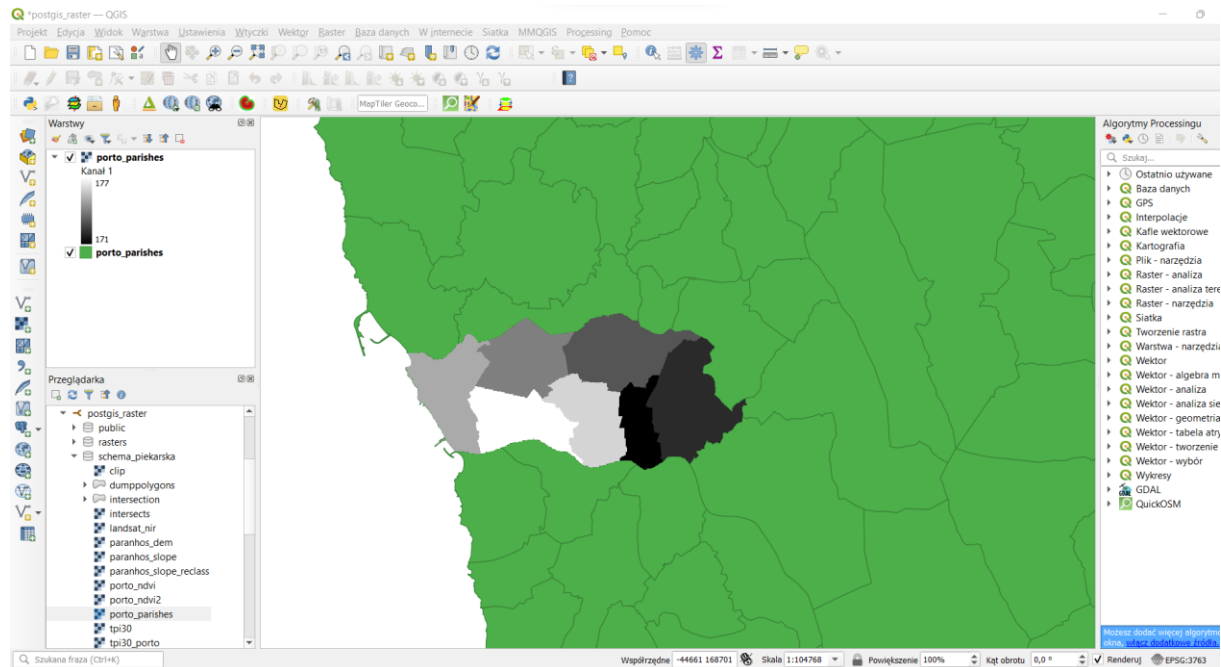
```
DROP TABLE schema_piekarska.porto_parishes; --> drop
table porto_parishes first
CREATE TABLE schema_piekarska.porto_parishes AS
WITH r AS (SELECT rast
            FROM rasters.dem
            LIMIT 1)
SELECT st_union(ST_AsRaster(a.geom,r.rast,'8BUI',a.id,-
32767)) AS rast
FROM vectors.porto_parishes AS a, r
WHERE a.municipality ilike 'porto';
```

Przykład 3 - ST_Tile

Po uzyskaniu pojedynczego rastra można generować kafelki za pomocą funkcji ST_Tile.

```
DROP TABLE schema_name.porto_parishes; --> drop table
porto_parishes first
CREATE TABLE schema_name.porto_parishes AS
WITH r AS (SELECT rast
            FROM rasters.dem
            LIMIT 1 )
```

```
SELECT
st_tile(st_union(ST_AsRaster(a.geom,r.rast,'8BUI',a.id,
-
32767)),128,128,true,-32767) AS rast
FROM vectors.porto_parishes AS a, r
WHERE a.municipality ilike 'porto';
```



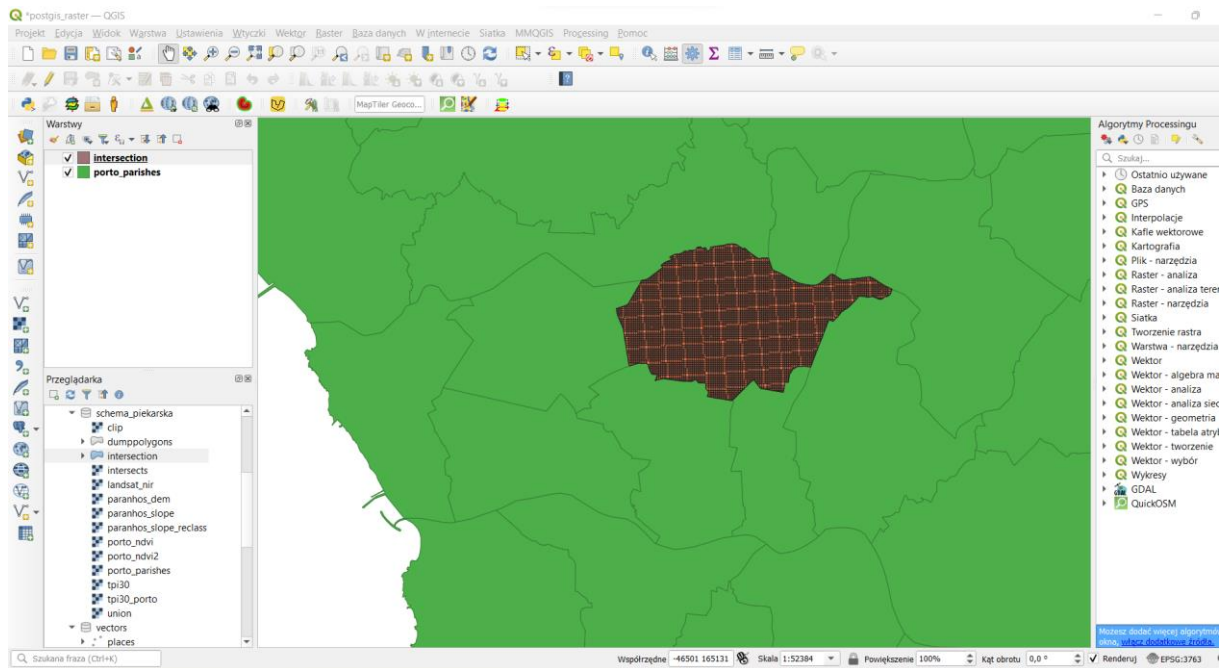
8. Konwertowanie rastrów na wektory (wektoryzowanie)

Poniższe przykłady użycia funkcji ST_Intersection i ST_DumpAsPolygons pokazują konwersję rastrów na wektory. PostGIS posiada więcej funkcji posiadających podobną funkcjonalność, aby dowiedzieć się więcej odwiedź https://postgis.net/docs/RT_reference.html#Raster_Processing_Geometry

Przykład 1 - ST_Intersection

Funkcja ST_Intersection jest podobna do ST_Clip. ST_Clip zwraca raster, a ST_Intersection zwraca zestaw par wartości geometria-piksel, ponieważ ta funkcja przekształca raster w wektor przed rzeczywistym „klipem”. Zazwyczaj ST_Intersection jest wolniejsze od ST_Clip więc zasadnym jest przeprowadzenie operacji ST_Clip na rastrze przed wykonaniem funkcji ST_Intersection.

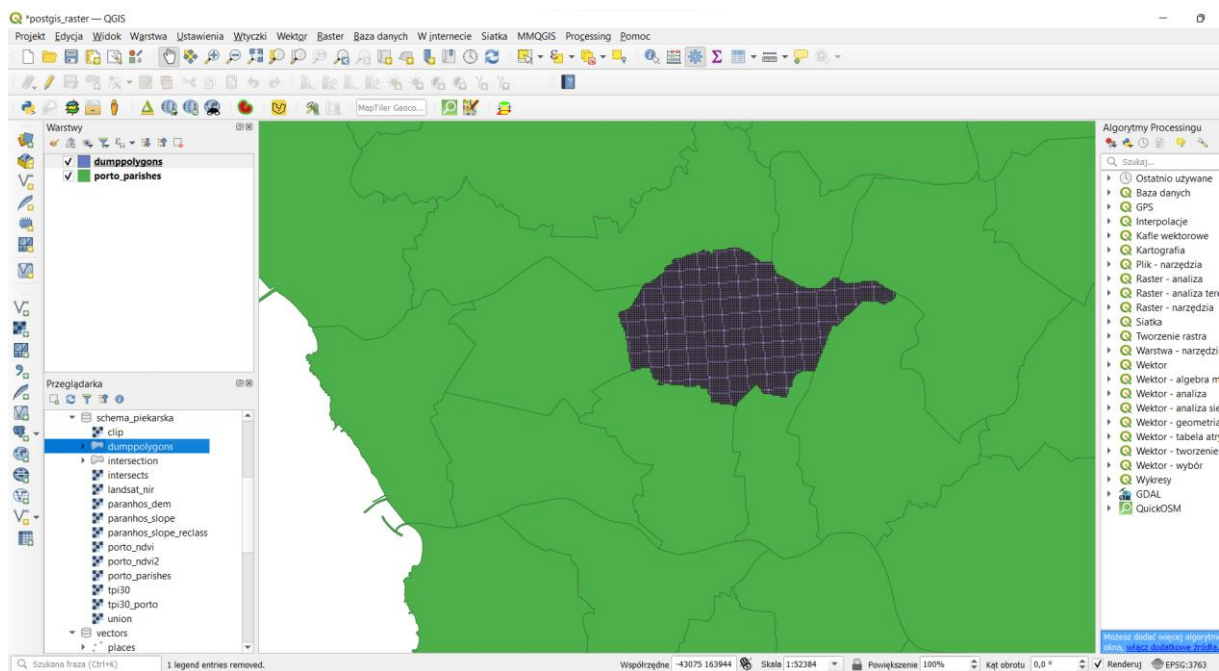
```
CREATE TABLE schema_piekarska.intersection as
SELECT
a.rid, (ST_Intersection(b.geom,a.rast)).geom, (ST_Intersection(b.geom,a.rast)).val
FROM rasters.landsat8 AS a, vectors.porto_parishes AS b
WHERE b.parish ilike 'paranhos' and
ST_Intersects(b.geom,a.rast);
```

Przykład 2 - ST_DumpAsPolygons

ST_DumpAsPolygons konwertuje rastry w wektory (poligony).

```
CREATE TABLE schema_piekarska.dumppolygons AS
SELECT
a.rid, (ST_DumpAsPolygons(ST_Clip(a.rast,b.geom))) .geom,
(ST_DumpAsPolygons(ST_Clip(a.rast,b.geom))) .val
FROM rasters.landsat8 AS a, vectors.porto_parishes AS b
WHERE b.parish ilike 'paranhos' and
ST_Intersects(b.geom,a.rast);
```



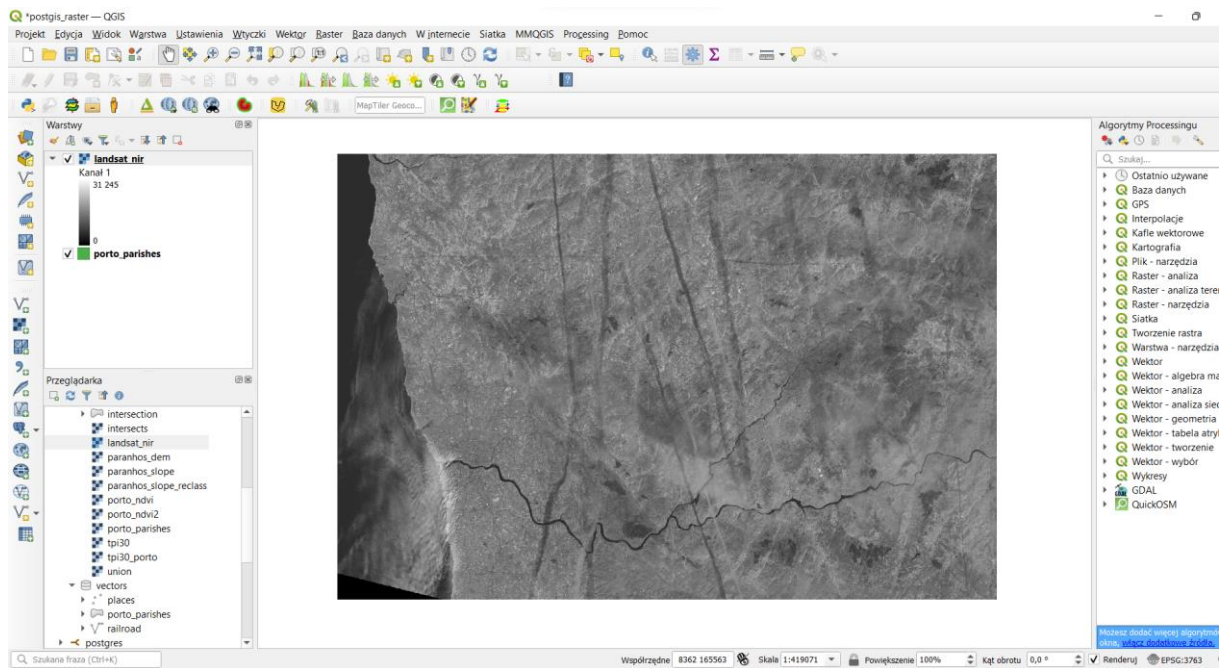
Obie funkcje zwracają zestaw wartości geomval, po więcej informacji dotyczących typu geomval sprawdź: <https://postgis.net/docs/geomval.html>

9. Analiza rastrów

Przykład 1 - ST_Band

Funkcja ST_Band służy do wyodrębniania pasm z rastra

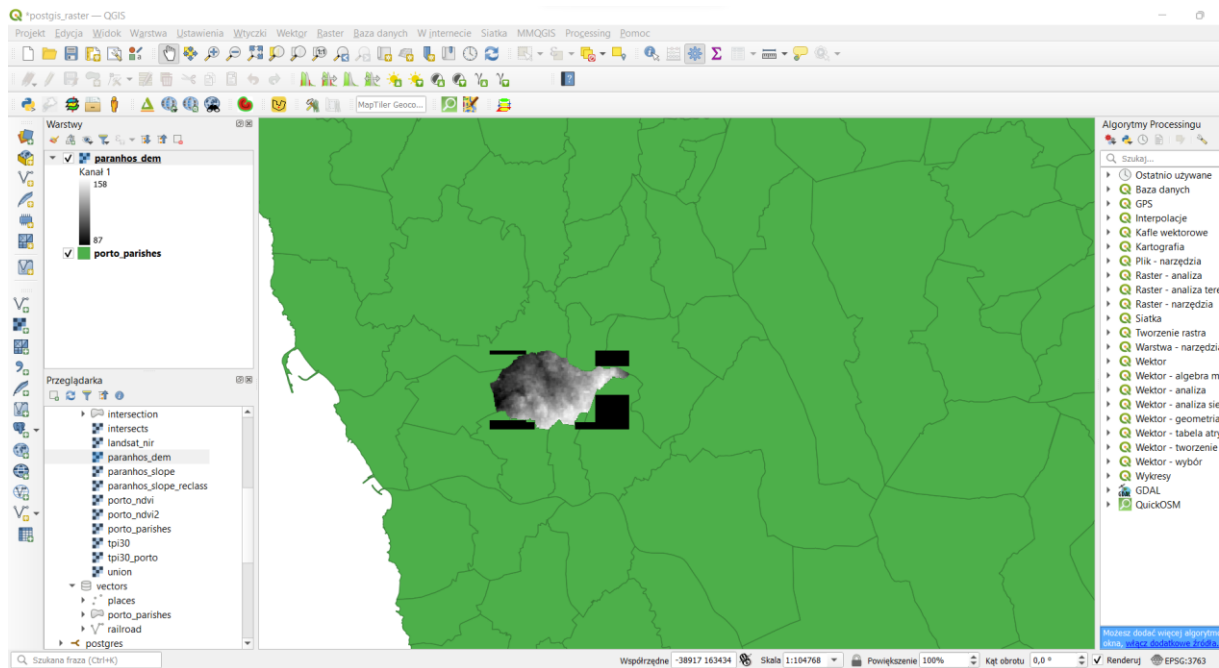
```
CREATE TABLE schema_piekarska.landsat_nir AS
SELECT rid, ST_Band(rast,4) AS rast
FROM rasters.landsat8;
```



Przykład 2 - ST_Clip

ST_Clip może być użyty do wycięcia rastra z innego rastra. Poniższy przykład wycina jedną parafię z tabeli vectors.porto_parishes. Wynik będzie potrzebny do wykonania kolejnych przykładów.

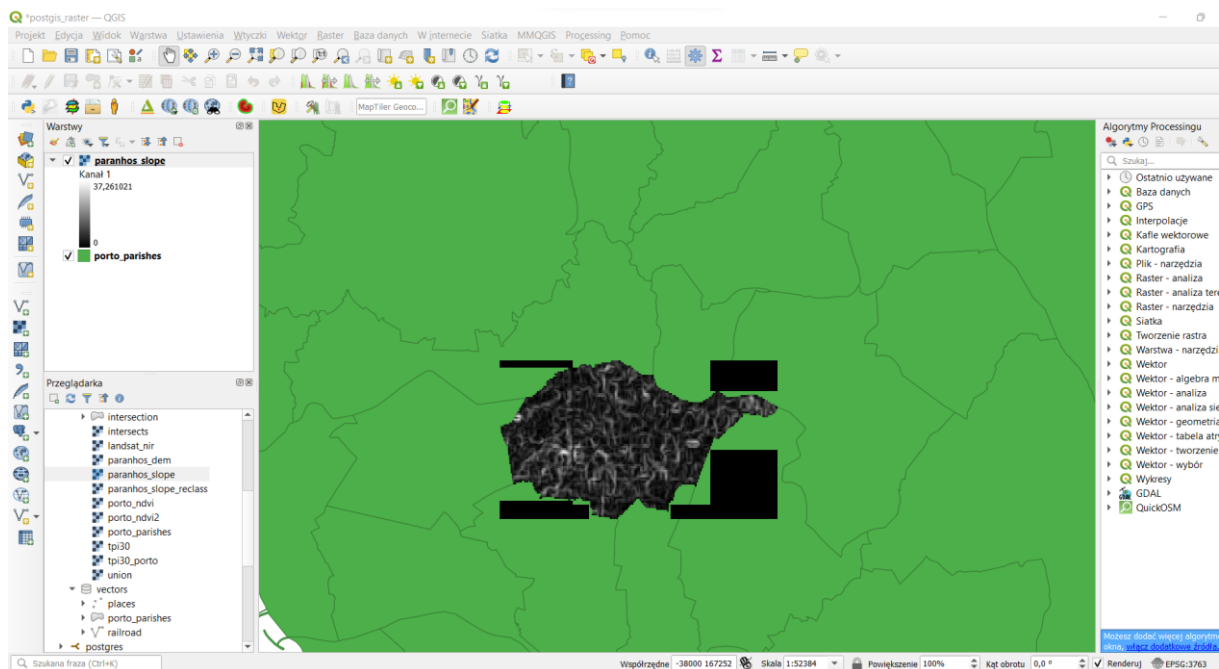
```
CREATE TABLE schema_piekarska.paranhos_dem AS
SELECT a.rid, ST_Clip(a.rast, b.geom, true) as rast
FROM rasters.dem AS a, vectors.porto_parishes AS b
WHERE b.parish ilike 'paranhos' and
ST_Intersects(b.geom, a.rast);
```

Przykład 3 - ST_Slope

Poniższy przykład użycia funkcji ST_Slope wygeneruje nachylenie przy użyciu poprzednio wygenerowanej tabeli (wzniesienie).

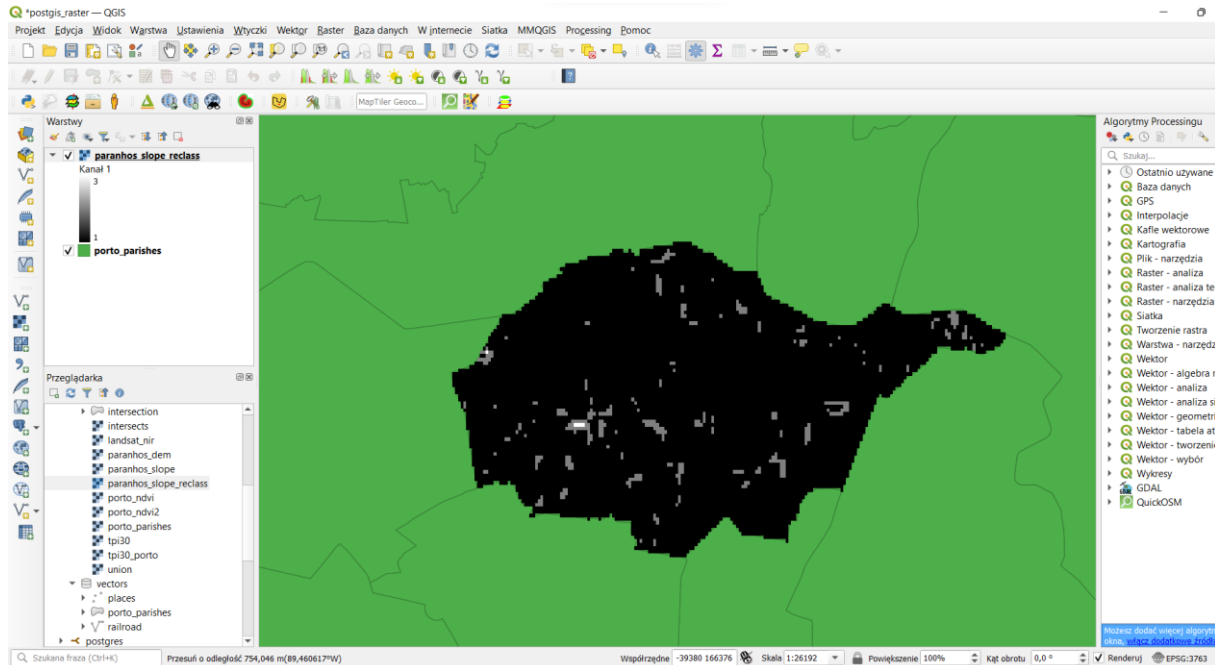
```
CREATE TABLE schema_piekarska.paranhos_slope AS
SELECT a.rid, ST_Slope(a.rast, 1, '32BF', 'PERCENTAGE') as
rast
FROM schema_piekarska.paranhos_dem AS a;
```



Przykład 4 - ST_Reclass

Aby zreklasifikować raster należy użyć funkcji ST_Reclass.

```
CREATE TABLE schema_piekarska.paranhos_slope_reclass AS
SELECT a.rid, ST_Reclass(a.rast, 1, '0-15]:1, (15-30]:2,
(30-9999:3', '32BF', 0)
FROM schema_piekarska.paranhos_slope AS a;
```



Przykład 5 - ST_SummaryStats

Aby obliczyć statystyki rastra można użyć funkcji ST_SummaryStats. Poniższy przykład wygeneruje statystyki dla kafelka.

```
SELECT st_summarystats(a.rast) AS stats
FROM schema_piekarska.paranhos_dem AS a;
```

Przykład 6 - ST_SummaryStats oraz Union

Przy użyciu UNION można wygenerować jedną statystykę wybranego rastra.

```
SELECT st_summarystats(ST_Union(a.rast))
FROM schema_piekarska.paranhos_dem AS a;
```

ST_SummaryStats zwraca złożony typ danych. Więcej informacji na temat złożonego typu danych znajduje się w dokumentacji:

<https://www.postgresql.org/docs/current/static/rowtypes.html>

Przykład 7 - ST_SummaryStats z lepszą kontrolą złożonego typu danych

```
WITH t AS (SELECT st_summarystats(ST_Union(a.rast)) AS
stats
FROM schema_piekarska.paranhos_dem AS a)
SELECT (stats).min, (stats).max, (stats).mean FROM t;
```

Przykład 8 - ST_SummaryStats w połączeniu z GROUP BY

Aby wyświetlić statystykę dla każdego poligonu "parish" można użyć polecenia

```
GROUP BY
WITH t AS (SELECT b.parish AS parish,
st_summarystats(ST_Union(ST_Clip(a.rast, b.geom,true)))
AS stats
FROM rasters.dem AS a, vectors.porto_parishes AS b
WHERE b.municipality ilike 'porto' and
ST_Intersects(b.geom,a.rast)
GROUP BY b.parish)
SELECT parish, (stats).min, (stats).max, (stats).mean
FROM t;
```

Przykład 9 - ST_Value

Funkcja ST_Value pozwala wyodrębnić wartość piksela z punktu lub zestawu punktów. Poniższy przykład wyodrębnia punkty znajdujące się w tabeli vectors.places.

Ponieważ geometria punktów jest wielopunktowa, a funkcja ST_Value wymaga geometrii jednopunktowej, należy przekonwertować geometrię wielopunktową na geometrię jednopunktową za pomocą funkcji (ST_Dump(b.geom)).geom.

```
SELECT b.name,st_value(a.rast, (ST_Dump(b.geom) ).geom)
FROM rasters.dem a, vectors.places AS b
WHERE ST_Intersects(a.rast,b.geom)
ORDER BY b.name;
```

Data Output

	name character varying (48)	st_value double precision
1	Aldeia São Miguel	96
2	Alpendurada e Matos	145
3	Amarante	71
4	Baião	581
5	Cabeceiras de Basto	[null]
6	Castelo de Paiva	284
7	Celorico de Basto	227
8	Cinfães	405
9	Espinho	14
10	Fafe	338
11	Fajozes	53
12	Felgueiras	320
13	Gondomar	123
14	Guifões	69
15	Guimarães	197
16	Lousada	289
17	Maia	111
18	Marco de Canaveses	193
19	Matosinhos	29
20	Paços de Ferreira	300

Query History Explain Notifications Messa

Przykład 10 - ST_TPI

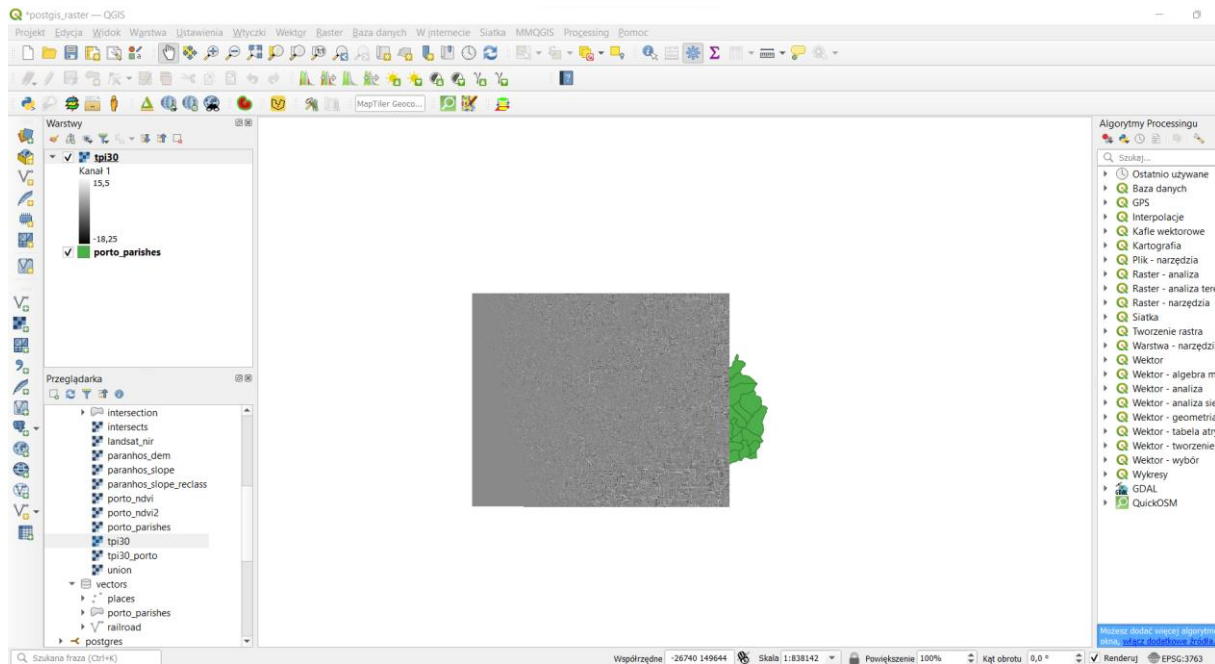
Topographic Position Index (TPI)

TPI porównuje wysokość każdej komórki w DEM ze średnią wysokością określonego sąsiedztwa wokół tej komórki. Wartości dodatnie reprezentują lokalizacje, które są wyższe niż średnia ich otoczenia, zgodnie z definicją sąsiedztwa (grzbietów). Wartości ujemne reprezentują lokalizacje, które są niższe niż ich otoczenie (doliny). Wartości TPI bliskie zeru to albo płaskie obszary (gdzie nachylenie jest bliskie zeru), albo obszary o stałym nachyleniu. Więcej informacji na temat TPI można znaleźć tutaj: www.jennessent.com/downloads/tpi-poster-tnc_18x22.pdf

```
CREATE TABLE schema_piekarska.tpi30 as
SELECT ST_TPI(a.rast,1) as rast
FROM rasters.dem a;
```

```
SELECT 589
```

```
Query returned successfully in 37 secs 482 msec.
```



Poniższa kwerenda utworzy indeks przestrzenny:

```
CREATE INDEX idx_tpi30_rast_gist
ON schema_piekarska.tpi30
USING gist (ST_ConvexHull(rast));
```

-- Dodanie constraintów:

```
SELECT AddRasterConstraints('schema_piekarska'::name,
'tpi30'::name, 'rast'::name);
```

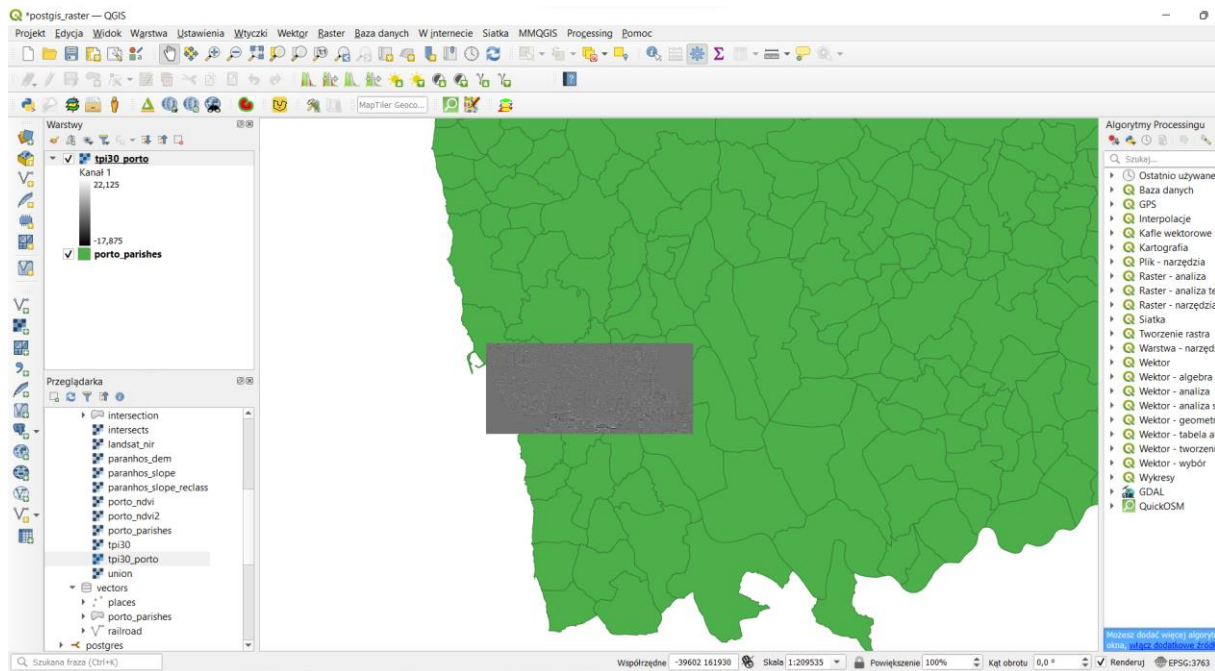
10. Problem do samodzielnego rozwiązania

Przetwarzanie poprzedniego zapytania może potrwać dłużej niż minutę, a niektóre zapytania mogą potrwać zbyt długo. W celu skrócenia czasu przetwarzania czasami można ograniczyć obszar zainteresowania i obliczyć mniejszy region. Dostosuj zapytanie z przykładu 10, aby przetwarzać tylko gminę Porto. Musisz użyć `ST_Intersects`, sprawdź Przykład 1 - `ST_Intersects` w celach informacyjnych. Porównaj różne czasy przetwarzania. Na koniec sprawdź wynik w QGIS.

```
CREATE TABLE schema_piekarska.tpi30_porto AS
SELECT ST_TPI(a.rast,1) as rast
FROM rasters.dem AS a, vectors.porto_parishes AS b
WHERE ST_Intersects(a.rast, b.geom) AND b.municipality
ILIKE 'porto';
```

```
SELECT 25
```

```
Query returned successfully in 1 secs 374 msec.
```



```
CREATE TABLE schema_piekarska.tpi30_porto AS
SELECT ST_TPI(a.rast,1) as rast
FROM rasters.dem AS a, vectors.porto_parishes AS b
WHERE ST_Intersects(a.rast, b.geom) AND b.municipality
ILIKE 'porto';
```

```
SELECT AddRasterConstraints('schema_piekarska'::name,
'tpi30_porto'::name, 'rast'::name);
```

11. Algebra map

Istnieją dwa sposoby korzystania z algebry map w PostGIS. Jednym z nich jest użycie wyrażenia, a drugim użycie funkcji zwrotnej. Poniższe przykłady pokazują jak stosując obie techniki utworzyć wartości NDVI na podstawie obrazu Landsat8.

Wzór na NDVI:

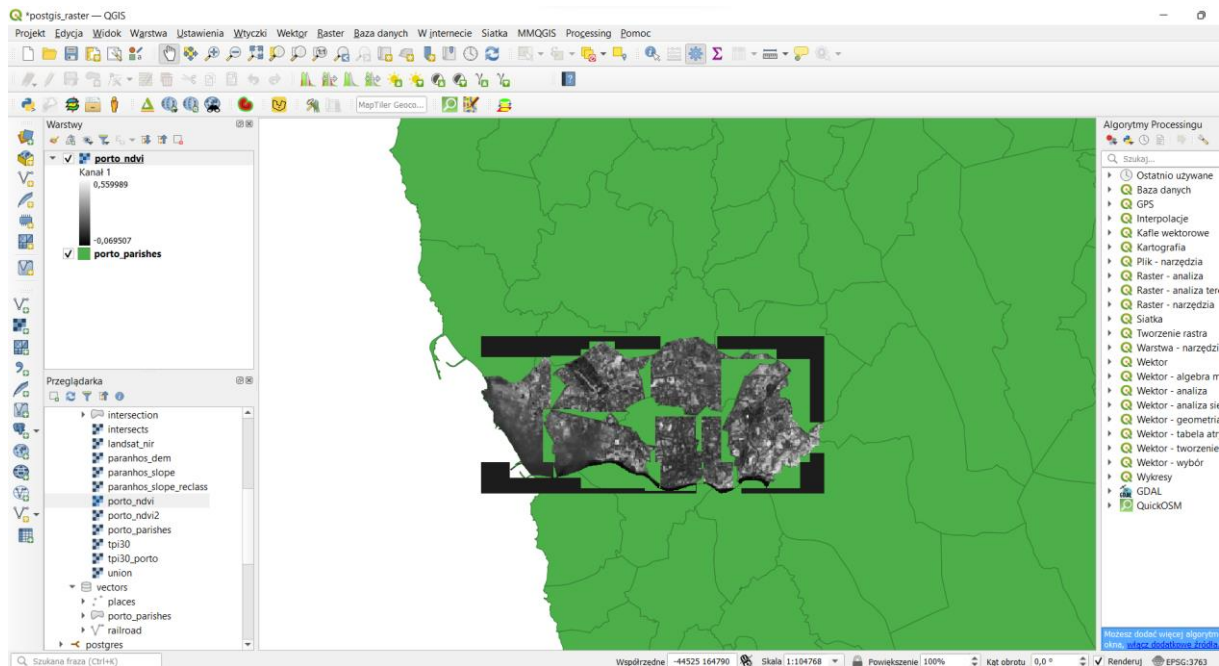
$$NDVI = (NIR - Red) / (NIR + Red)$$

Przykład 1 - Wyrażenie Algebry Map

```
CREATE TABLE schema_piekarska.porto_ndvi AS
WITH r AS (SELECT a.rid, ST_Clip(a.rast, b.geom, true) AS
rast
FROM rasters.landsat8 AS a, vectors.porto_parishes AS b
WHERE b.municipality ilike 'porto' and
ST_Intersects(b.geom, a.rast))
SELECT
r.rid, ST_MapAlgebra(
r.rast, 1,
r.rast, 4,
'([rast2.val] - [rast1.val]) / ([rast2.val] +
```



```
[rast1.val])::float','32BF'
) AS rast
FROM r;
```



Poniższe zapytanie utworzy indeks przestrzenny na wcześniej stworzonej tabeli:

```
CREATE INDEX idx_porto_ndvi_rast_gist ON
schema_piekarska.porto_ndvi
USING gist (ST_ConvexHull(rast));
```

Dodanie constraintów:

```
SELECT AddRasterConstraints('schema_piekarska'::name,
'porto_ndvi'::name, 'rast'::name);
```

Możliwe jest użycie algebry map na wielu rastrach i/lub wielu pasmach, służy do tego rastbandargset. Więcej informacji jest dostępnych w dokumentacji:

https://postgis.net/docs/RT_ST_MapAlgebra.html

Przykład 2 – Funkcja zwrotna

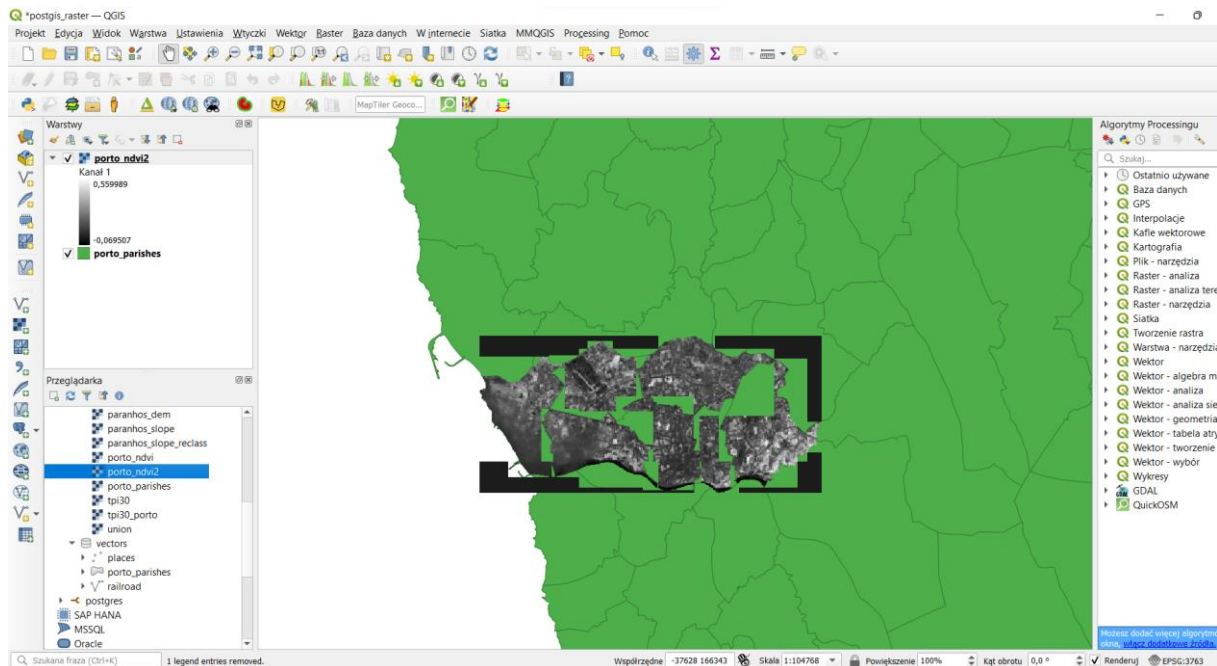
W pierwszym kroku należy utworzyć funkcję, które będzie wywołana później:

```
CREATE OR REPLACE FUNCTION schema_piekarska.ndvi(
value double precision [] [] [],
pos integer [][],
VARIADIC userargs text []
)
RETURNS double precision AS
$$
BEGIN
```

```
--RAISE NOTICE 'Pixel Value: %', value [1][1][1];-->For
debug purposes
RETURN (value [2][1][1] - value [1][1][1])/(value
[2][1][1]+value
[1][1][1]); --> NDVI calculation!
END;
$$
LANGUAGE 'plpgsql' IMMUTABLE COST 1000;
```

W kwerendzie algebry map należy wywołać zdefiniowaną wcześniej funkcję:

```
CREATE TABLE schema_piekarska.porto_ndvi2 AS
WITH r AS (
SELECT a.rid,ST_Clip(a.rast, b.geom,true) AS rast
FROM rasters.landsat8 AS a, vectors.porto_parishes AS b
WHERE b.municipality ilike 'porto' and
ST_Intersects(b.geom,a.rast)
)
SELECT
r.rid,ST_MapAlgebra(
r.rast, ARRAY[1,4],
'schema_piekarska.ndvi(double precision[],
integer[],text[])':::regprocedure, --> This is the
function!
'32BF'::text
) AS rast
FROM r;
```



Dodanie indeksu przestrzennego:

```
CREATE INDEX idx_porto_ndvi2_rast_gist ON  
schema_piekarska.porto_ndvi2  
USING gist (ST_ConvexHull(rast));
```

Dodanie constraintów:

```
SELECT AddRasterConstraints('schema_piekarska'::name,  
'porto_ndvi2'::name, 'rast'::name);
```

Przykład 3 - Funkcje TPI

Aktualnie zaimplementowana w PostGIS funkcja TPI wykorzystuje algebrę mapy z wywołaniem funkcji.

Schemat public zawiera dwie funkcje TPI:

- public._st_tpi4ma - funkcja używana w algebrze map
- public.st_tpi - funkcja, która wywołuje poprzednią funkcję. Istnieją dwie funkcje st_tpi, które różnią się liczbą dozwolonych wejść lecz obie te funkcje wykonują tę samą akcję.

Przeanalizuj kod wspomnianych funkcji oraz sposób ich wykonania.

Więcej informacji odnośnie algebry map w PostGIS znajduje się na stronach:

MapAlgebra z wyrażeniem:

https://postgis.net/docs/RT_ST_MapAlgebra_expr.html

MapAlgebra z wywołaniem funkcji:

https://postgis.net/docs/RT_ST_MapAlgebra.html

Obecna implementacja TPI w PostGIS obsługiwana przy użyciu funkcji ST_TPI pozwala tylko na obliczenie TPI z jedną komórką sąsiedztwa. Nowa implementacja TPI pozwalająca użytkownikowi określić komórki sąsiedztwa (wewnętrzny pierścień i pierścień zewnętrzny), za pomocą algebry mapy dostępna jest tutaj:

https://github.com/lcalisto/postgis_customTPI

12. Eksport danych

Poniższe przykłady pokazują jak wyeksportować rastry stworzone we wcześniejszych przykładach.

PostGIS może zapisywać rastry w różnych formatach plików. Poniższe przykłady używają funkcji ST_AsTiff i ST_AsGDALRaster, ale także funkcjonalności wbudowanej w Gdal.

Przykład 0 - Użycie QGIS

Po załadowaniu tabeli/widoku z danymi rastrowymi do QGIS, możliwe jest zapisanie/wyeksportowanie warstwy rastrowej do dowolnego formatu obsługiwanej przez GDAL za pomocą interfejsu QGIS.

Przykład 1 - ST_AsTiff

Funkcja ST_AsTiff tworzy dane wyjściowe jako binarną reprezentację pliku tiff, może to być przydatne na stronach internetowych, skryptach itp., w których programista może kontrolować, co zrobić z plikiem binarnym, na przykład zapisać go na dysku lub po prostu wyświetlić.

```
SELECT ST_AsTiff(ST_Union(rast))
FROM schema_name.porto_ndvi;
```

Przykład 2 - ST_AsGDALRaster

Podobnie do funkcji ST_AsTiff, ST_AsGDALRaster nie zapisuje danych wyjściowych bezpośrednio na dysku, natomiast dane wyjściowe są reprezentacją binarną dowolnego formatu GDAL.

```
SELECT ST_AsGDALRaster(ST_Union(rast), 'GTiff',
ARRAY['COMPRESS=DEFLATE',
'PREDICTOR=2', 'PZLEVEL=9'])
FROM schema_piekarska.porto_ndvi;
```

Uwaga:

Funkcje ST_AsGDALRaster pozwalają nam zapisać raster w dowolnym formacie obsługiwanych przez gdal. Aby wyświetlić listę formatów obsługiwanych przez bibliotekę uruchom:

```
SELECT ST_GDALDrivers();
```

Przykład 3 - Zapisywanie danych na dysku za pomocą dużego obiektu (large object, lo)

```
CREATE TABLE tmp_out AS
SELECT lo_from_bytea(0,
ST_AsGDALRaster(ST_Union(rast), 'GTiff',
ARRAY['COMPRESS=DEFLATE',
'PREDICTOR=2', 'PZLEVEL=9'])
) AS loid
FROM schema_piekarska.porto_ndvi;
-----
SELECT lo_export(loid, 'G:\myraster.tiff') --> Save the
file in a place where the user postgres have access. In
windows a flash drive usually works fine.
FROM tmp_out;
-----
SELECT lo_unlink(loid)
FROM tmp_out; --> Delete the large object.
```

Więcej informacji odnośnie eksportu danych rastrowych dostępna jest na stronie:

https://postgis.net/docs/RT_reference.html#Raster_Outputs

Przykład 4 - Użycie Gdal

Gdal obsługuje rastry z PostGISa. Polecenie `gdal_translate` eksportuje raster do dowolnego formatu obsługiwanego przez GDAL.

```
gdal_translate -co COMPRESS=DEFLATE -co PREDICTOR=2 -co
ZLEVEL=9 PG:"host=localhost port=5432
dbname=postgis_raster user=postgres password=postgis
schema=schema_piekarska table=porto_ndvi mode=2"
porto_ndvi.tiff
```

13. Publikowanie danych za pomocą MapServer

Ponieważ GDAL obsługuje rastry PostGIS, możliwe jest opublikowanie rastra jako WMS.

Należy pamiętać, że w takim przypadku zaleca się generowanie podglądów w celu uzyskania lepszej wydajności.

Poniższy przykład to plik mapowania z rastrem przy użyciu standardowych opcji i klauzuli WHERE.

Przykład 1 – Mapfile

MAP

```
NAME 'map'
SIZE 800 650
STATUS ON
EXTENT -58968 145487 30916 206234
UNITS METERS
```

WEB

METADATA

```
'wms_title' 'Terrain wms'
'wms_srs' 'EPSG:3763 EPSG:4326 EPSG:3857'
'wms_enable_request' '*'
'wms_onlineresource'
'http://54.37.13.53/mapservices/srtm'
```

END

END

PROJECTION

```
'init=epsg:3763'
```

END

LAYER

```
NAME srtm
TYPE raster
STATUS OFF
```

```
DATA "PG:host=localhost port=5432 dbname='postgis_raster'
user='sasig'"
```

```
password='postgis' schema='rasters' table='dem'  
mode='2'" PROCESSING  
        "SCALE=AUTO"  
        PROCESSING "NODATA=-32767"  
        OFFSITE 0 0 0  
        METADATA  
                'wms_title' 'srtm'  
        END  
END  
END
```

Przykładowy WMS jest dostępny pod adresem:

<https://sigap.calisto.pt/mapservices/srtm>

lub w przeglądarce:

<https://sigap.calisto.pt/mapservices/srtm?layer=srtm&mode=map>