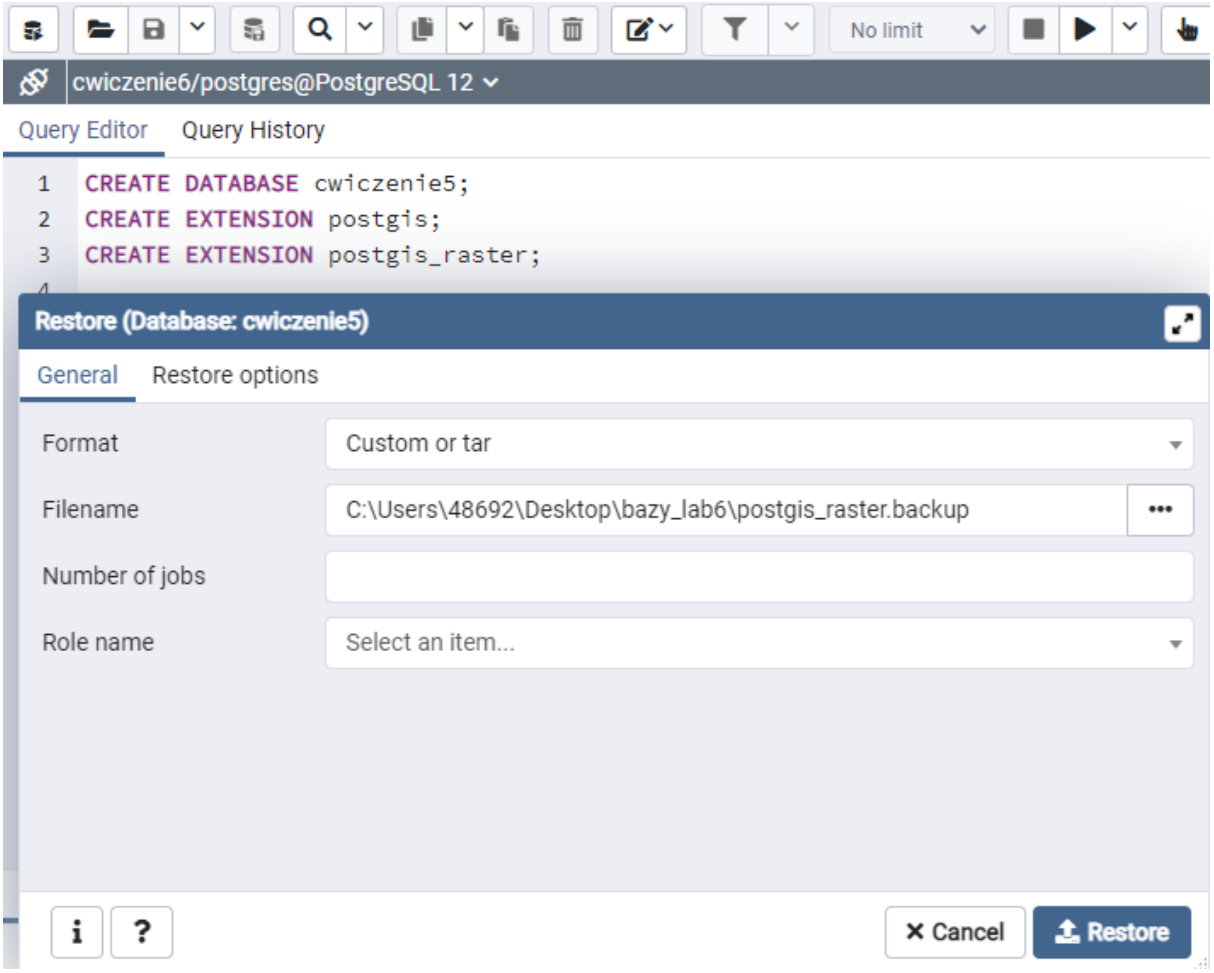
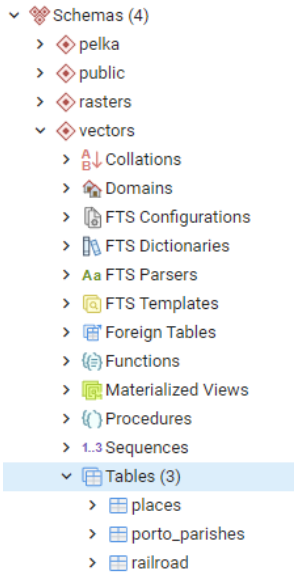
Bazy danych przestrzennych – ćwiczenia z PostGIS raster

Nowa baza danych:



Struktura bazy danych:



Ładowanie danych rastrowych:

Przykład 1 – ładowanie rastru przy użyciu pliku .sql

(C:\Program Files\PostgreSQL\13\bin)

raster2pgsql.exe -s 3763 -N -32767 -t 100x100 -I -C -M -d "C:\Users\48692\Desktop\bazy_lab6\rasters\srtm_1arc_v3.tif" rasters.dem > "C:\Users\48692\Desktop\bazy_lab6\rasters\dem.sql"

```
C:\Program Files\PostgreSQL\13\bin>raster2pgsql.exe -s 3763 -N -32767 -t 100x100 -I -C -M -d "C:\Users\48692\Desktop\baz
y_lab6\rasters\srtm_1arc_v3.tif" rasters.dem > "C:\Users\48692\Desktop\bazy_lab6\rasters\dem.sql"
Processing 1/1: C:\Users\48692\Desktop\bazy_lab6\rasters\srtm_1arc_v3.tif
```

Przykład 2 – ładowanie rastru bezpośrednio do bazy

raster2pgsql.exe -s 3763 -N -32767 -t 100x100 -I -C -M -d "C:\Users\48692\Desktop\bazy_lab6\rasters\srtm_1arc_v3.tif" rasters.dem | psql -U postgres -d cwiczenie5 -h localhost -p 5432

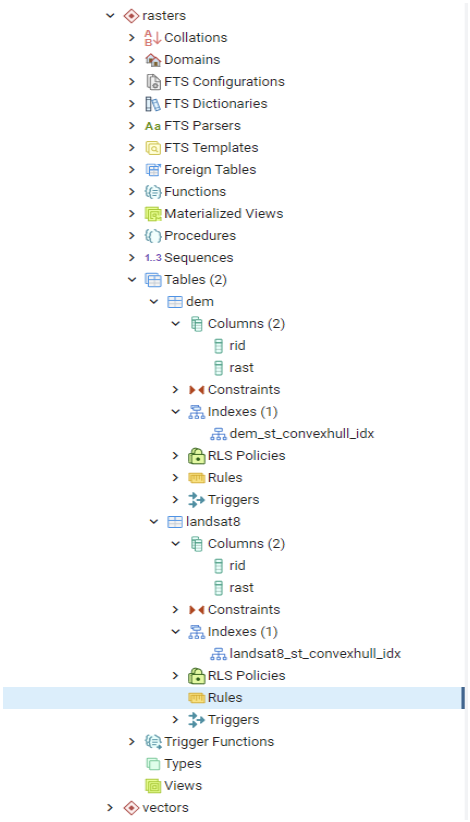
```
C:\Program Files\PostgreSQL\13\bin>raster2pgsql.exe -s 3763 -N -32767 -t 100x100 -I -C -M -d "C:\Users\48692\Desktop\baz
y_lab6\rasters\srtm_1arc_v3.tif" rasters.dem | psql -U postgres -d cwiczenie5 -h localhost -p 5432
Processing 1/1: C:\Users\48692\Desktop\bazy_lab6\rasters\srtm_1arc_v3.tif
Password for user postgres:
BEGIN
NOTICE:  table "dem" does not exist, skipping
DROP TABLE
CREATE TABLE
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
```

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

raster2pgsql.exe -s 3763 -N -32767 -t 128x128 -I -C -M -d "C:\Users\48692\Desktop\bazy_lab6\rasters\Landsat8_L1TP_RGBN.TIF" rasters.landsat8 | psql -d cwiczenie5 -h localhost -U postgres -p 5432

```
C:\Program Files\PostgreSQL\13\bin>raster2pgsql.exe -s 3763 -N -32767 -t 128x128 -I -C -M -d "C:\Users\48692\Desktop\baz
y_lab6\rasters\Landsat8_L1TP_RGBN.TIF" rasters.landsat8 | psql -d cwiczenie5 -h localhost -U postgres -p 5432
Processing 1/1: C:\Users\48692\Desktop\bazy_lab6\rasters\Landsat8_L1TP_RGBN.TIF
Password for user postgres:
BEGIN
NOTICE:  table "landsat8" does not exist, skipping
DROP TABLE
CREATE TABLE
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
```

Schemat rasters:

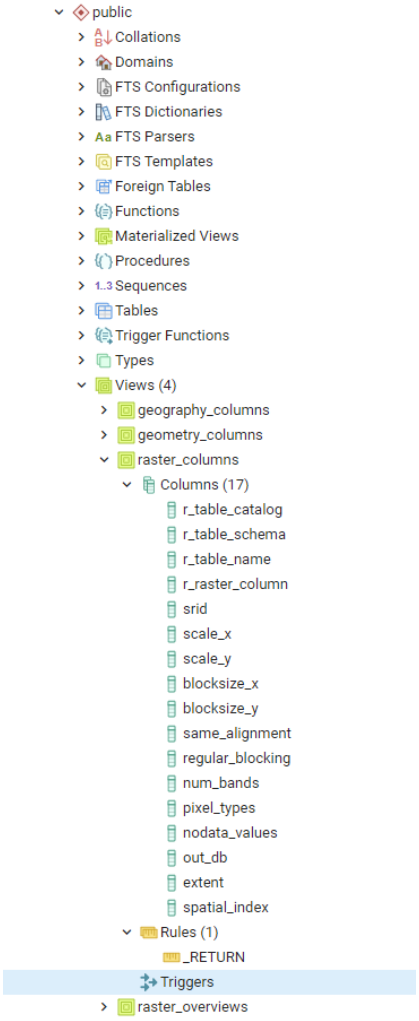


SELECT * FROM rasters.dem;

Data Output | Explain | Messages | Notifications

	rid [PK] integer	rast raster
1	1	01000001006172BF3E4D5A374080318D6907CA3EC0F25D96820DCBECC04257316BD52C0941000000000000000000000000000000000B30E0000
2	2	01000001006172BF3E4D5A374080318D6907CA3EC0DC0406BD24A7EBC04257316BD52C0941000000000000000000000000000000000B30E0000
3	3	01000001006172BF3E4D5A374080318D6907CA3EC0C6AB75F73B83EAC04257316BD52C0941000000000000000000000000000000000B30E0000
4	4	01000001006172BF3E4D5A374080318D6907CA3EC0B152E531535FE9C04257316BD52C0941000000000000000000000000000000000B30E0000
5	5	01000001006172BF3E4D5A374080318D6907CA3EC09BF9546C6A3BE8C04257316BD52C0941000000000000000000000000000000000B30E0000
6	6	01000001006172BF3E4D5A374080318D6907CA3EC085A0C4A68117E7C04257316BD52C0941000000000000000000000000000000000B30E0000
7	7	01000001006172BF3E4D5A374080318D6907CA3EC0704734E198F3E5C04257316BD52C0941000000000000000000000000000000000B30E0000

Schemat public.raster_columns:



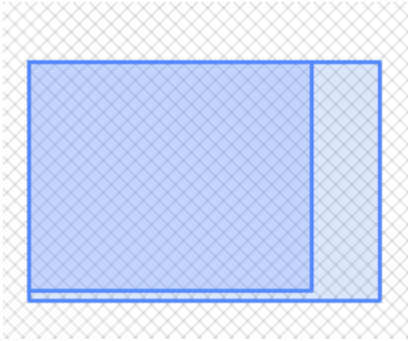
SELECT * FROM public.raster_columns;

Data Output | Explain | Messages | Notifications

	r_table_catalog name	r_table_schema name	r_table_name name	r_raster_column name	srid integer	scale_x double precision	scale_y double precision	blocksize_x integer	blocksize_y integer	same_alignment boolean	regular_blocking boolean	num_bands integer
1	cwiczenie5	rasters	dem	rast	3763	23.3527411668	-30.7891756029	100	100	true	false	1
2	cwiczenie5	rasters	landsat8	rast	3763	30.3114020783	-29.7057939174	128	128	true	false	4

Data Output | Explain | Messages | Notifications

pixel_types text[]	nodata_values double precision[]	out_db boolean[]	extent geometry	spatial_index boolean
{16BSI}	{-32767}	{f}	0103000020B30E0000010000000500000F25D96820DCBECC0E73716F3B9080241F25D96820DCBECC042	true
{16BUI,16BUI,16BUI,16BUI}	{65535,65535,65535,65535}	{f,f,f,f}	0103000020B30E0000010000000500000002068193DBBCBECC078F977B959C201412068193DBBCBECC071	true



Tworzenie rastrów z istniejących rastrów i interakcja z wektorami:

Przykład 1 - ST_Intersects:

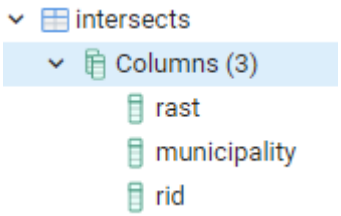
Przecięcie rastra z wektorem.

```
CREATE TABLE pelka.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';

SELECT * FROM pelka.intersects;
```

rast raster	municipality character varying (254)
01000001006172BF3E4D5A374080318D6907CA3EC044951356C7ABE3C082283B553DAA0441000000000000000000000000000000(	PORTO
01000001006172BF3E4D5A374080318D6907CA3EC044951356C7ABE3C0474F11FE054A0441000000000000000000000000000000(	PORTO
01000001006172BF3E4D5A374080318D6907CA3EC044951356C7ABE3C082283B553DAA0441000000000000000000000000000000(	PORTO
01000001006172BF3E4D5A374080318D6907CA3EC02E3C8390DE87E2C082283B553DAA0441000000000000000000000000000000(	PORTO
01000001006172BF3E4D5A374080318D6907CA3EC044951356C7ABE3C0474F11FE054A0441000000000000000000000000000000(	PORTO
01000001006172BF3E4D5A374080318D6907CA3EC02E3C8390DE87E2C0474F11FE054A0441000000000000000000000000000000(	PORTO
01000001006172BF3E4D5A374080318D6907CA3EC018E3F2CAF563E1C0474F11FE054A0441000000000000000000000000000000(	PORTO

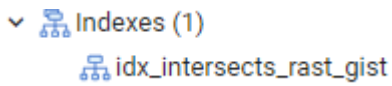
1. dodanie serial primary key:

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



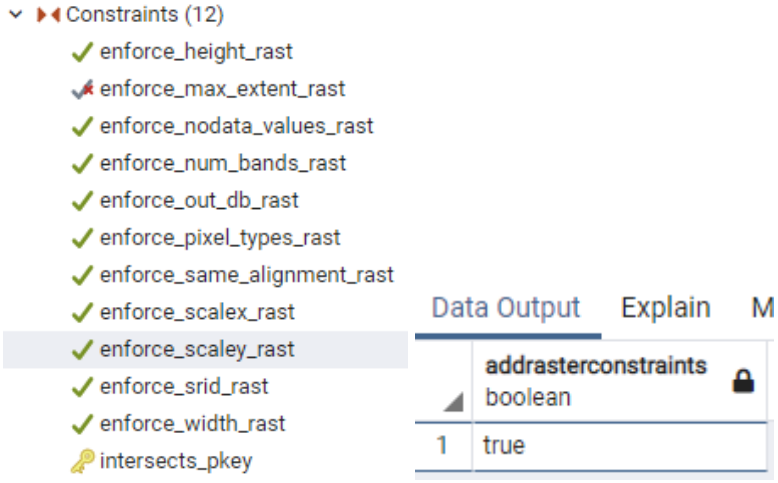
2. utworzenie indeksu przestrzennego:

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



3. dodanie raster constraints:

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



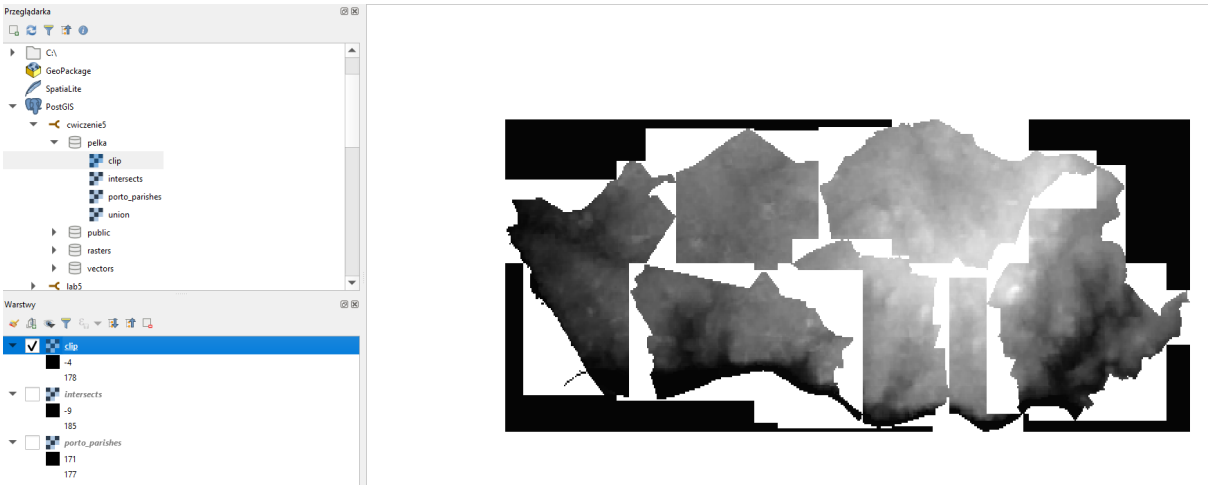
Przykład 2 - ST_Clip:

Obcinanie rastra na podstawie wektora.

```
CREATE TABLE pelka.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';
```

Data Output | Explain | Messages | Notifications | Geometry Viewer

	st_clip raster	municipality character varying (254)
1	01000001006172BF3E4D5A374080318D6907CA3EC02C288A402D31E3C0D0979D709B5404410000000(	PORTO
2	01000001006172BF3E4D5A374080318D6907CA3EC0574768B43454E3C0474F11FE054A044100000000	PORTO
3	01000001006172BF3E4D5A374080318D6907CA3EC025E358E1A7B3E2C079DC7A05D068044100000000(	PORTO
4	01000001006172BF3E4D5A374080318D6907CA3EC02E3C8390DE87E2C08B6D93EAFA7D044100000000(	PORTO
5	01000001006172BF3E4D5A374080318D6907CA3EC00962D67B5CE2E2C0474F11FE054A044100000000(	PORTO
6	01000001006172BF3E4D5A374080318D6907CA3EC02E3C8390DE87E2C0474F11FE054A044100000000(	PORTO
7	01000001006172BF3E4D5A374080318D6907CA3EC018E3F2CAF563E1C018D5974A973B044100000000(	PORTO



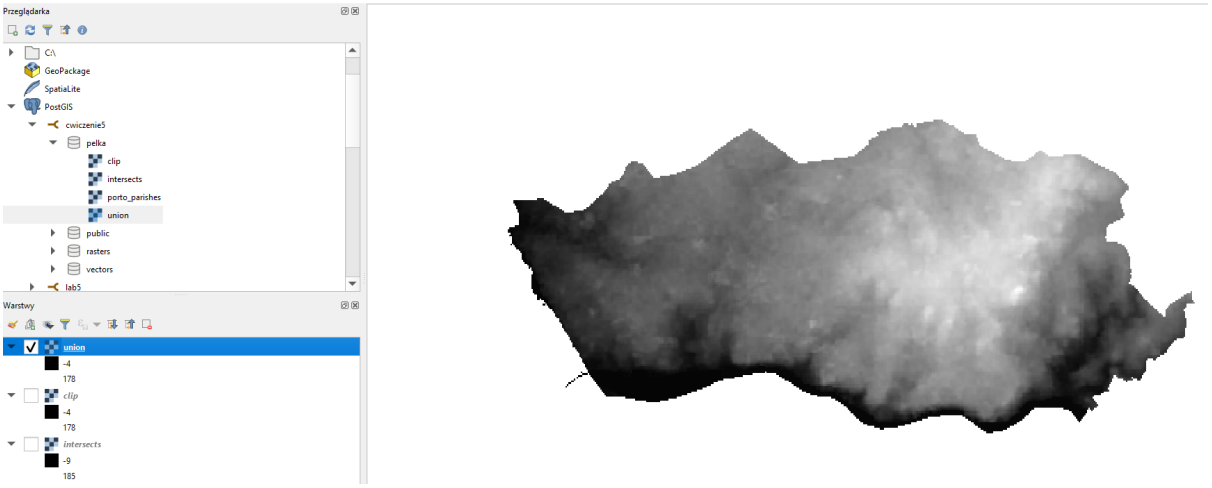
Przykład 3 - ST_Union:

Połączenie wielu kafelków w jeden raster.

```
CREATE TABLE pelka.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);
```

Data Output | Explain | Messages | Notifications | Geometry Viewer

	st_union raster
1	01000001006172BF3E4D5A374080318D6907CA3EC0D6994F2F0BE0E6C0D9E35DC00896044100000000000000000000000000000000B30E0000F201AC00450180018001800180018001800180018001800180018001800180018001800180018001800180018001800180018001800



Tworzenie rastrów z wektorów (rastrowanie):

Przykład 1 - ST_AsRaster:

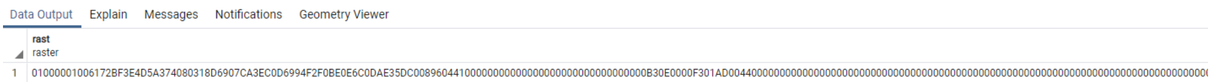
```
CREATE TABLE pelka.porto_parishes AS
WITH r AS (
SELECT rast FROM rasters.dem
LIMIT 1
)
SELECT ST_AsRaster(a.geom,r.rast,'8BUI',a.id,-32767) AS rast
FROM vectors.porto_parishes AS a, r
WHERE a.municipality ilike 'porto';
```

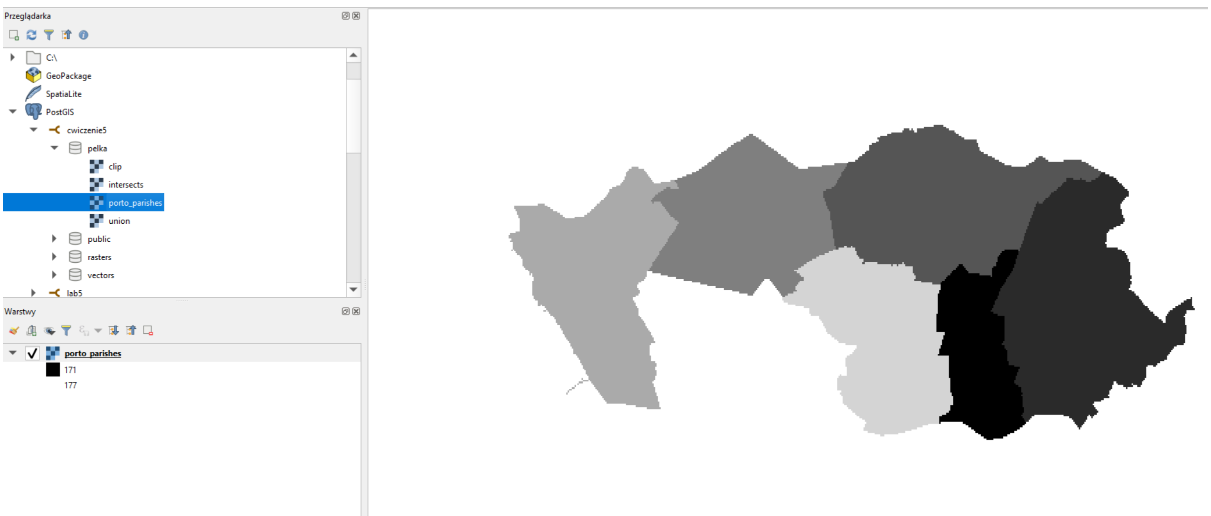
Data Output | Explain | Messages | Notifications | Geometry Viewer

	rast raster
1	01000001006172BF3E4D5A374080318D6907CA3EC0574768B43454E3C0D0979D709B5404410000000000000000000000000000000
2	01000001006172BF3E4D5A374080318D6907CA3EC00A62D67B5CE2E2C08A6D93EAFA7D0441000000000000000000000000000000
3	01000001006172BF3E4D5A374080318D6907CA3EC0D268172037B9E5C0CA65352F399104410000000000000000000000000000000
4	01000001006172BF3E4D5A374080318D6907CA3EC0D6994F2F0BE0E6C05E060A8BE77F0441000000000000000000000000000000
5	01000001006172BF3E4D5A374080318D6907CA3EC09A49D3957D46E4C0D9E35DC00896044100000000000000000000000000000
6	01000001006172BF3E4D5A374080318D6907CA3EC087977E37109EE4C0A33014118856044100000000000000000000000000000
7	01000001006172BF3E4D5A374080318D6907CA3EC00FB04D4A53D9E5C0474F11FE054A0441000000000000000000000000000000

Przykład 2 - ST_Union:

DROP TABLE pelka.porto_parishes; --> drop table porto_parishes first
CREATE TABLE pelka.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';



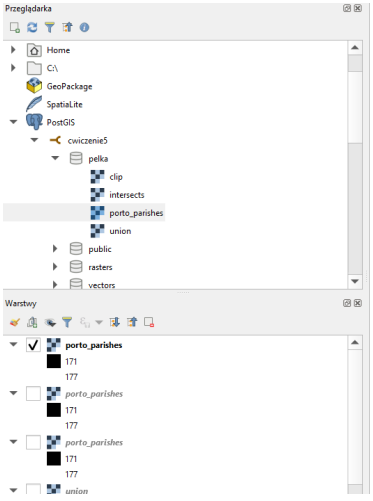


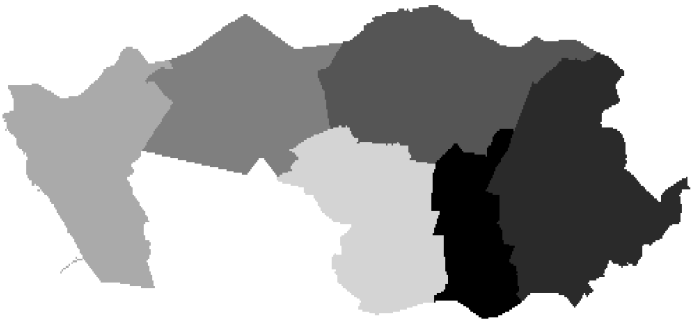
Przykład 3 - ST_Tile:

```
DROP TABLE pelka.porto_parishes; --> drop table porto_parishes first
CREATE TABLE pelka.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';
```

Data Output | Explain | Messages | Notifications | Geometry Viewer

	rast raster
1	01000001006172BF3E4D5A374080318D6907CA3EC0D6994F2F0BE0E6C0DAE35DC0089604410000000000000000000000000000000B30E0000800080004400000000000000000000000000
2	01000001006172BF3E4D5A374080318D6907CA3EC0B0A2635B666AE5C0DAE35DC0089604410000000000000000000000000000000B30E0000800080004400000000000000000000000000
3	01000001006172BF3E4D5A374080318D6907CA3EC08AAB7787C1F4E3C0DAE35DC0089604410000000000000000000000000000000B30E0000800080004400000000000000000000000000
4	01000001006172BF3E4D5A374080318D6907CA3EC064B48BB31C7FE2C0DAE35DC0089604410000000000000000000000000000000B30E0000800080004400000000000000000000000000
5	01000001006172BF3E4D5A374080318D6907CA3EC0D6994F2F0BE0E6C014AFB7A2E01A0441000000000000000000000000000000B30E0000800080004400000000000000000000000000
6	01000001006172BF3E4D5A374080318D6907CA3EC0B0A2635B666AE5C014AFB7A2E01A0441000000000000000000000000000000B30E0000800080004400B1B1B1B1B1B1B1B1B1B1B1
7	01000001006172BF3E4D5A374080318D6907CA3EC08AAB7787C1F4E3C014AFB7A2E01A0441000000000000000000000000000000B30E0000800080004400B0B0B0B0B0B0B0B0B0B0B0B0





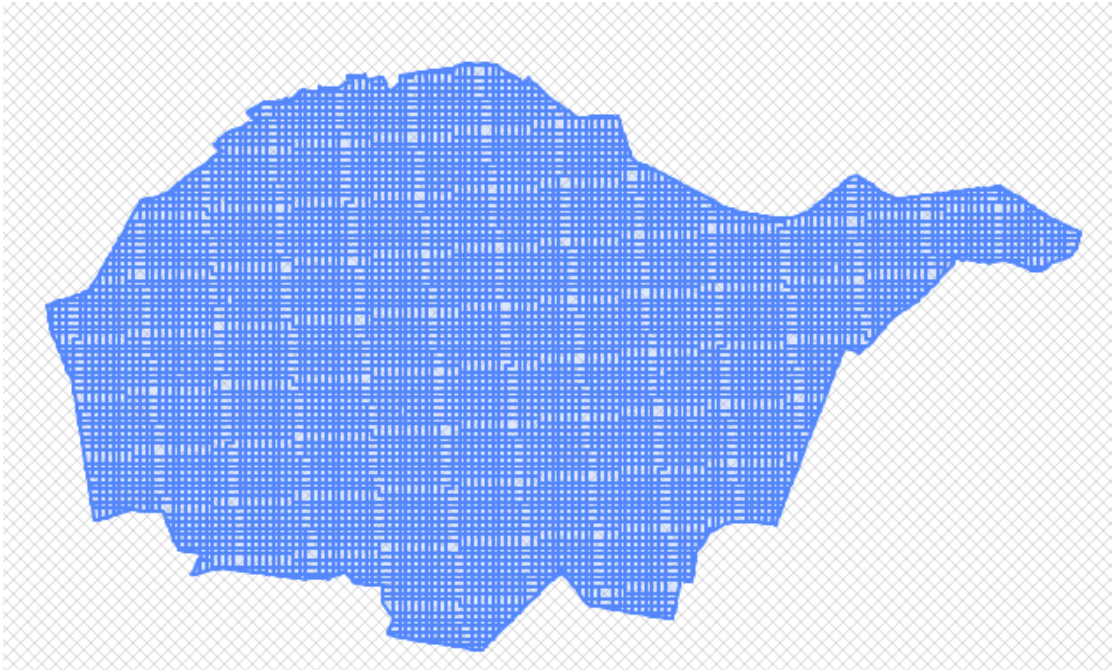
Konwertowanie rastrów na wektory (wektoryzowanie):

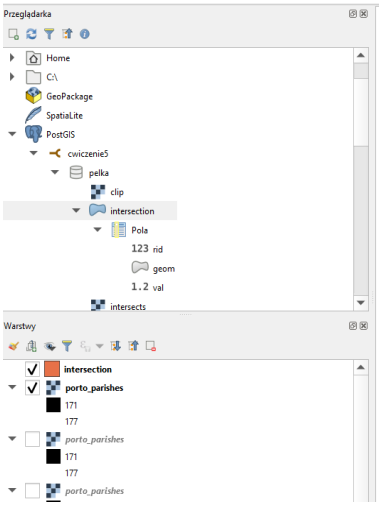
Przykład 1 - ST_Intersection:

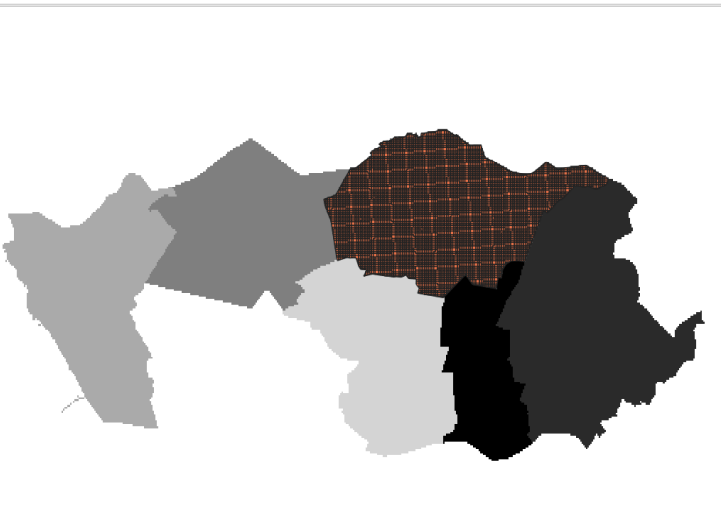
```
CREATE TABLE pelka.intersection AS
SELECT
a.rid,(ST_Intersection(b.geom,a.rast)).geom,(ST_Intersection(b.geom,a.rast)
).val
FROM rasters.landsat8 AS a, vectors.porto_parishes AS b
WHERE b.parish ilike 'paranhos' and ST_Intersects(b.geom,a.rast);
```

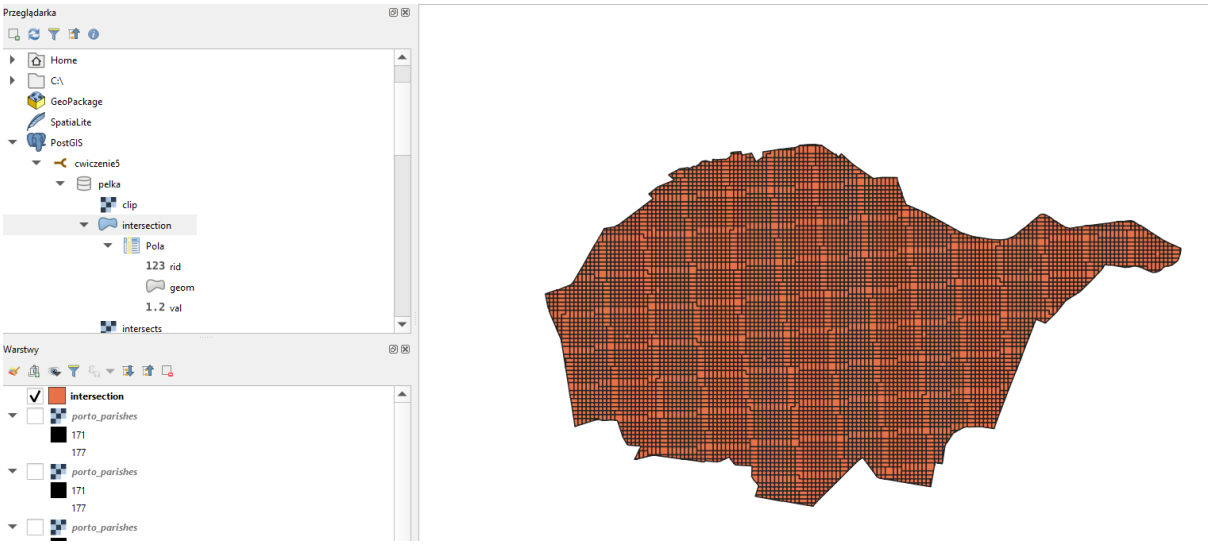
Data Output | Explain | Messages | Notifications | Geometry Viewer

	rid integer	geom geometry	val double precision
1	221	0103000020B30E0000010000000500000086F60B09B56E3C0B0B5B943A8950441086F60B0	10648
2	221	0103000020B30E0000010000000500000003EF15EB9D152E3C01EAC6512C29504413EF15EB9	12155
3	221	0103000020B30E0000010000000400000BF61697A576DE3C0187DD4649B940441BF61697A	9248
4	221	0103000020B30E0000010000000600000F5E367838D69E3C03BD5CB34BC940441F5E36783	10030
5	221	0103000020B30E0000010000000500000002C66668CC365E3C0D605C213DB9404412C66668C	10347
6	221	0103000020B30E0000010000000500000063E86495F961E3C07136B8F2F994044163E86495F	10126
7	221	0103000020B30E0000010000000500000009A6A639E2F5EE3C00C67AED1189504419A6A639E	10611







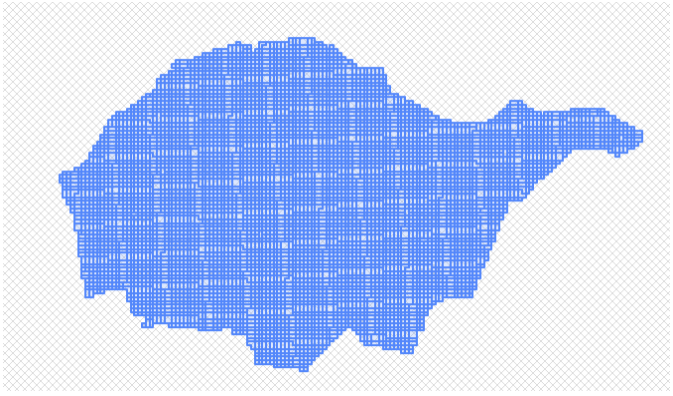


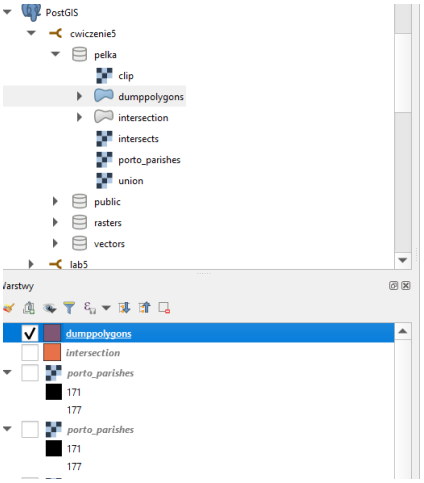
Przykład 2 - ST_DumpAsPolygons:

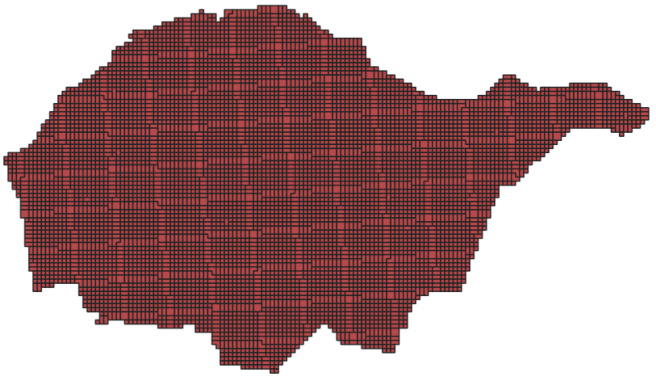
CREATE TABLE pelka.dumppolygons AS
SELECT
a.rid,(ST_DumpAsPolygons(ST_Clip(a.rast,b.geom))).geom,(ST_DumpAsPolygons(ST_Clip(a.rast,b.geom))).val
FROM rasters.landsat8 AS a, vectors.porto_parishes AS b
WHERE b.parish ilike 'paranhos' and ST_Intersects(b.geom,a.rast);

Data Output | Explain | Messages | Notifications | Geometry Viewer

	rid integer	geom geometry	val double precision
1	221	0103000020B30E0000010000000500000009A6A639E2F5EE3C0E32C703B809504419A6A639E2F5EE3C0D	12761
2	221	0103000020B30E0000010000000500000D0EC61A7655AE3C0E32C703B80950441D0EC61A7655AE3C0	14382
3	221	0103000020B30E0000010000000500000076F60B09B56E3C0E32C703B80950441076F60B09B56E3C0D	14090
4	221	0103000020B30E0000010000000500000009A4B78203B93E3C0DAE4F895929404419A4B78203B93E3C0D	10625
5	221	0103000020B30E0000010000000700000E3586F567F7CE3C0DAE4F89592940441E3586F567F7CE3C0C	8601
6	221	0103000020B30E0000010000000500000087DF6A712171E3C0DAE4F8959294044187DF6A712171E3C0C	10102
7	221	0103000020B30E0000010000000500000BE61697A576DE3C0DAE4F89592940441BE61697A576DE3C0C	11050







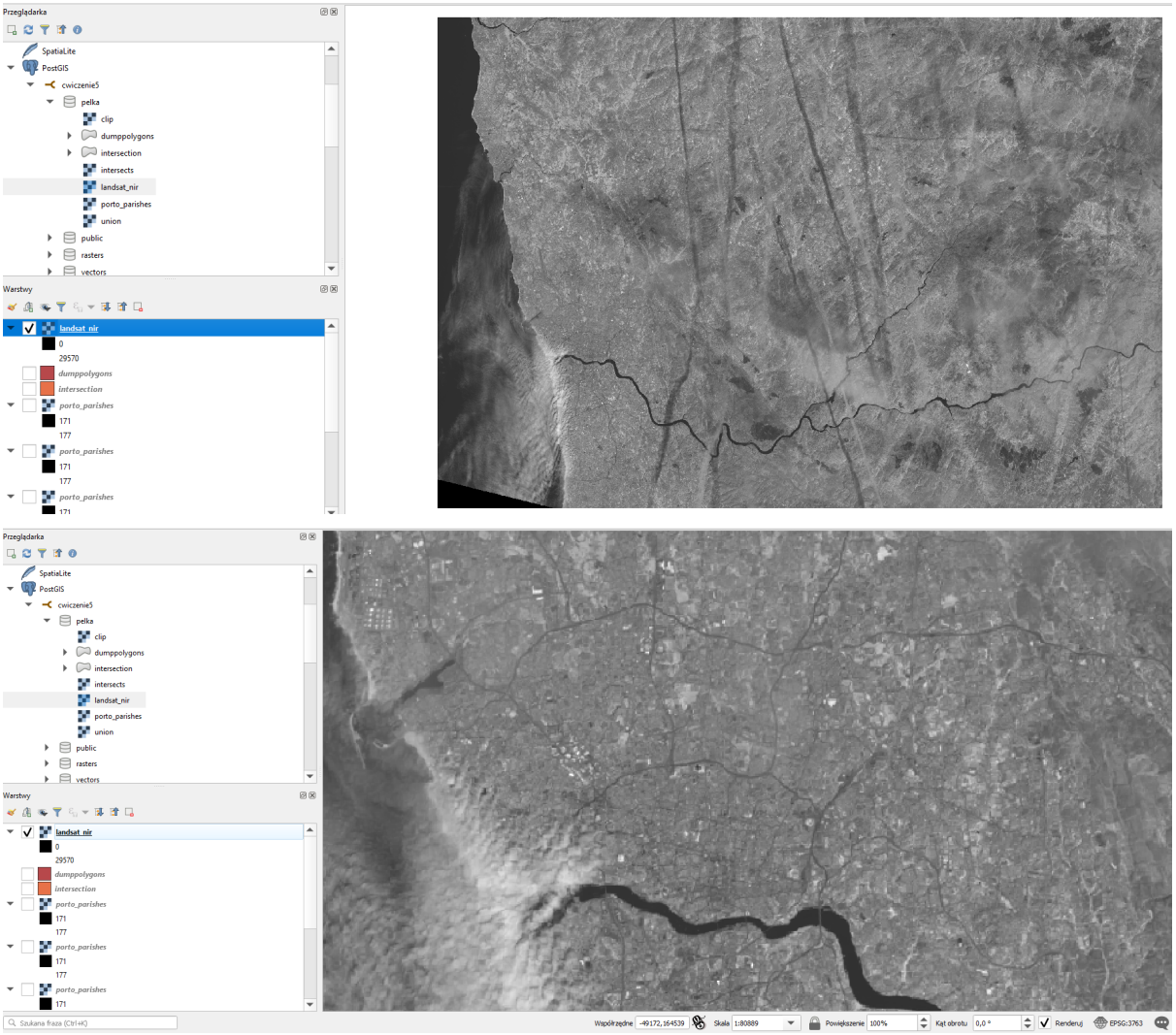
Analiza rastrów:

Przykład 1 - ST_Band:

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

Data Output | Explain | Messages | Notifications | Geometry Viewer

	rid integer	rast raster
1	1	01000001003849EE0BB84F3E404F2001E9AEB43DC02068193DBBCBECC071695283BC2C0941000
2	2	01000001003849EE0BB84F3E404F2001E9AEB43DC08C835ABCBFE6EAC071695283BC2C0941000
3	3	01000001003849EE0BB84F3E404F2001E9AEB43DC0F99E9B3BC401E9C071695283BC2C0941000
4	4	01000001003849EE0BB84F3E404F2001E9AEB43DC066BADCBAC81CE7C071695283BC2C0941000
5	5	01000001003849EE0BB84F3E404F2001E9AEB43DC0D2D51D3ACD37E5C071695283BC2C0941000
6	6	01000001003849EE0BB84F3E404F2001E9AEB43DC03EF15EB9D152E3C071695283BC2C0941000
7	7	01000001003849EE0BB84F3E404F2001E9AEB43DC0AB0CA038D66DE1C071695283BC2C0941000
8	8	01000001003849EE0BB84F3E404F2001E9AEB43DC02F50C26FB511DFC071695283BC2C0941000
9	9	01000001003849EE0BB84F3E404F2001E9AEB43DC00887446EBE47DBC071695283BC2C0941000
10	10	01000001003849EE0BB84F3E404F2001E9AEB43DC0E0BDC66CC77DD7C071695283BC2C0941000
11	11	01000001003849EE0BB84F3E404F2001E9AEB43DC0BAF4486BD0B3D3C071695283BC2C0941000
12	12	01000001003849EE0BB84F3E404F2001E9AEB43DC0285796D3B2D3CFC071695283BC2C0941000

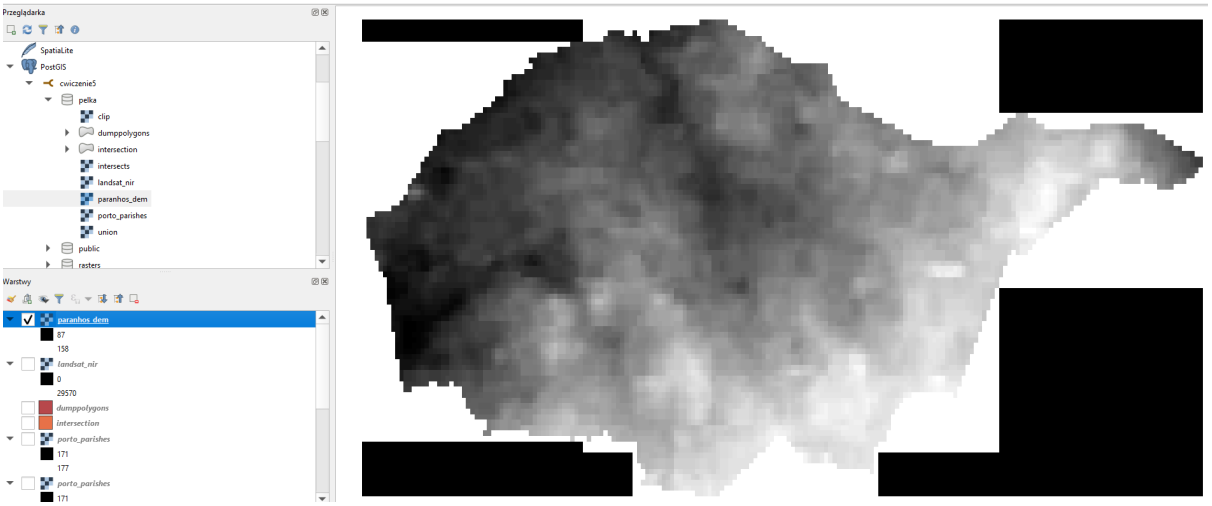


Przykład 2 - ST_Clip:

```
CREATE TABLE pelka.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);
```

Data Output | Explain | Messages | Notifications | Geometry Viewer

	rid integer	rast raster
1	380	01000001006172BF3E4D5A374080318D6907CA3EC09A49D3957D46E4C033B2707F2F9204410000000000000000000000000000000B30E000035004900450180018001800
2	381	01000001006172BF3E4D5A374080318D6907CA3EC044951356C7ABE3C0DAE35DC00896044100000000000000000000000000000000B30E000064004F00450180018001800
3	382	01000001006172BF3E4D5A374080318D6907CA3EC02E3C8390DE87E2C0D7D06D6CAD85044100000000000000000000000000000000B30E000031002000450180018001800
4	412	01000001006172BF3E4D5A374080318D6907CA3EC0187635E2BF88E3C0474F11FE054A044100000000000000000000000000000000B30E00003B0008004501800180890084

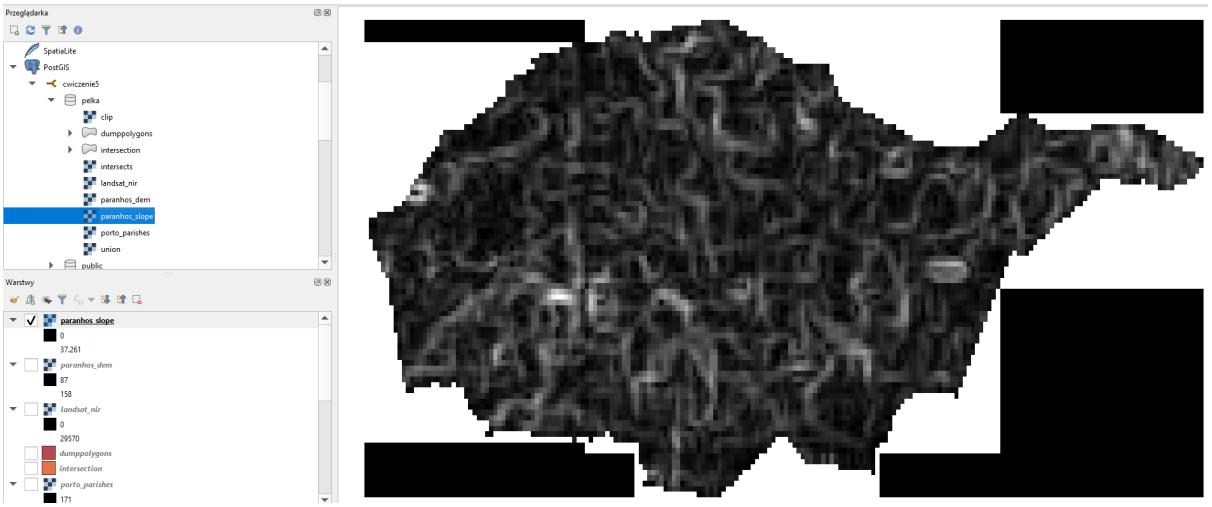


Przykład 3 - ST_Slope:

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

Data Output | Explain | Messages | Notifications | Geometry Viewer

	rid integer	rast raster
1	380	01000001006172BF3E4D5A374080318D6907CA3EC09A49D3957D46E4C033B2707F2F92044100000000000000000000000000000000B30E
2	381	01000001006172BF3E4D5A374080318D6907CA3EC044951356C7ABE3C0DAE35DC00896044100000000000000000000000000000000B30
3	382	01000001006172BF3E4D5A374080318D6907CA3EC02E3C8390DE87E2C0D7D06D6CAD85044100000000000000000000000000000000B3C
4	412	01000001006172BF3E4D5A374080318D6907CA3EC0187635E2BF88E3C0474F11FE054A044100000000000000000000000000000000B30E

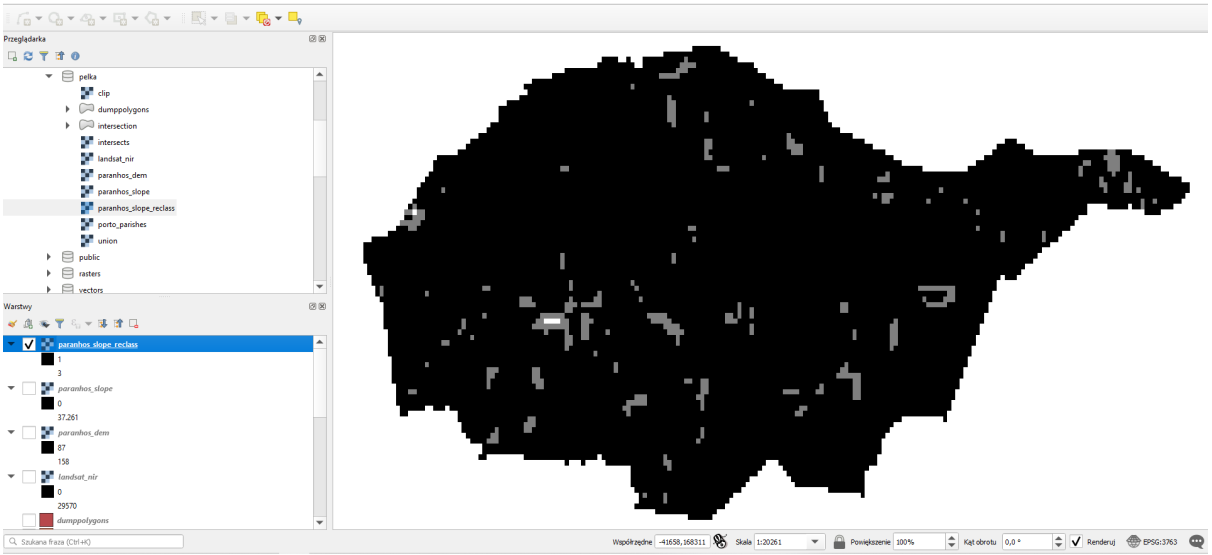


Przykład 4 - ST_Reclass:

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

Data Output | Explain | Messages | Notifications | Geometry Viewer

	rid integer	st_reclass raster
1	380	01000001006172BF3E4D5A374080318D6907CA3EC09A49D3957D46E4C033B2707F2F92044100000000000000000000000000000C
2	381	01000001006172BF3E4D5A374080318D6907CA3EC044951356C7ABE3C0DAE35DC00896044100000000000000000000000000000(
3	382	01000001006172BF3E4D5A374080318D6907CA3EC02E3C8390DE87E2C0D7D06D6CAD85044100000000000000000000000000000
4	412	01000001006172BF3E4D5A374080318D6907CA3EC0187635E2BF88E3C0474F11FE054A04410000000000000000000000000000



Przykład 5 - ST_SummaryStats:

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

Data Output | Explain | Messages | Notifications | Geometry

	stats summarystats
1	(2616,278385,106.41628440366972,11.622628762211638,87,143)
2	(6463,816615,126.35231316725978,14.0438229209133,94,158)
3	(682,95581,140.14809384164224,12.078072186605759,103,158)
4	(216,31874,147.5648148148148,4.262830628315728,137,158)

Przykład 6 - ST_SummaryStats oraz Union:

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

Data Output | Explain | Messages | Notifications | Geometry

	st_summarystats summarystats
1	(9977,1222455,122.52731281948482,16.908004202736272,87,158)

Przykład 7 - ST_SummaryStats z lepszą kontrolą złożonego typu danych:

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

Data Output | Explain | Messages | Notifications | Geometry V

	min double precision	max double precision	mean double precision
1	87	158	122.52731281948482

Przykład 8 - ST_SummaryStats w połączeniu z 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;
```

Data Output | Explain | Messages | Notifications | Geometry Viewer

	parish character varying (254)	min double precision	max double precision	mean double precision
1	Bonfim	1	159	107.5658842667906
2	Campanhã	0	178	74.66732213085449
3	Paranhos	87	158	122.52731281948482
4	Ramalde	48	108	77.58444444444444
5	União das freguesias de Aldoar, Foz do Douro e Nevogilde	-4	83	34.66735489791237
6	União das freguesias de Cedofeita, Santo Ildefonso, Sé, Miragaia, São Nicolau e Vitória	1	157	95.00277741039545
7	União das freguesias de Lordelo do Ouro e Massarelos	-1	117	49.50051440329218

Przykład 9 - ST_Value:

```
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 | Explain | Messages | Notificatic

	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

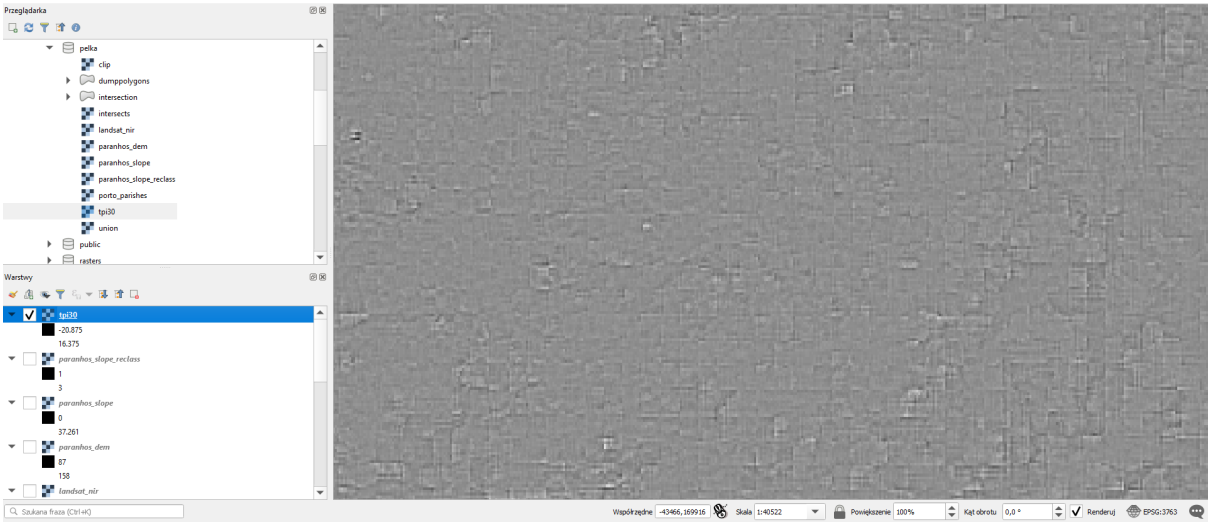
Topographic Position Index (TPI):

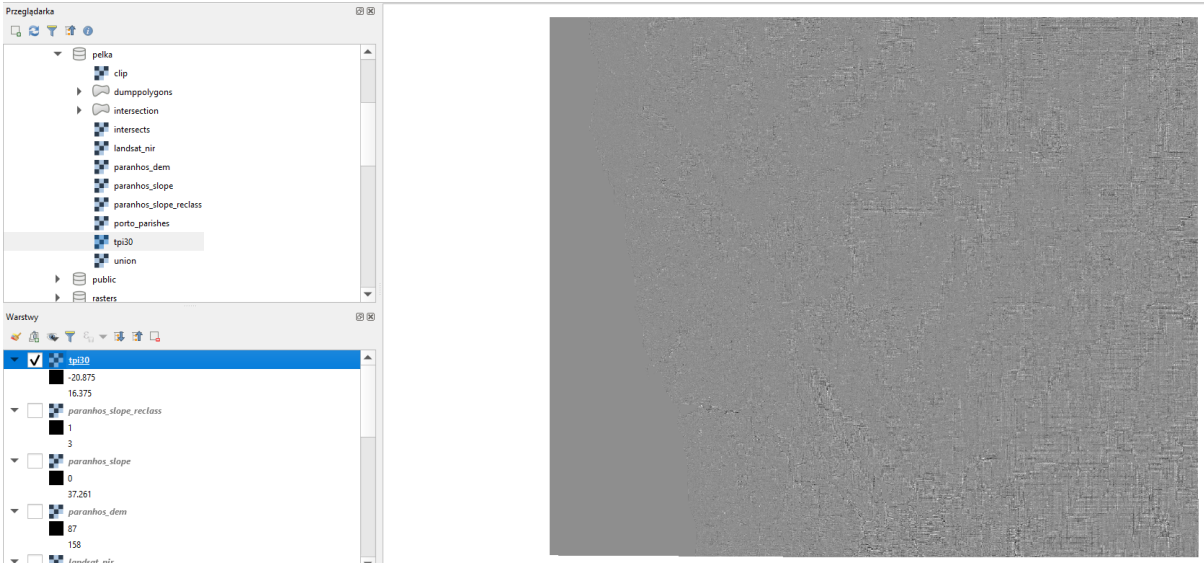
Przykład 10 - ST_TPI:

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

Data Output | Explain | Messages | Notifications | Geometry Viewer

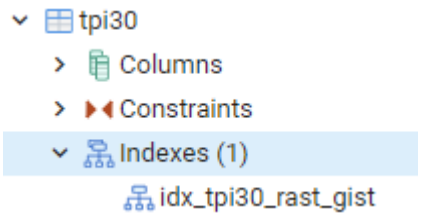
	rast raster
1	01000001006172BF3E4D5A374080318D6907CA3EC0F25D96820DCBECC04257316BD52C0941000000000000000000000000000000000B30E0000640064004A00FEFFC60000000000000000(
2	01000001006172BF3E4D5A374080318D6907CA3EC0DC0406BD24A7EBC04257316BD52C0941000000000000000000000000000000000B30E0000640064004A00FEFFC60000000000000000(
3	01000001006172BF3E4D5A374080318D6907CA3EC0C6AB75F73B83EAC04257316BD52C0941000000000000000000000000000000000B30E0000640064004A00FEFFC6000080BE0000C0BE0(
4	01000001006172BF3E4D5A374080318D6907CA3EC0B152E531535FE9C04257316BD52C0941000000000000000000000000000000000B30E0000640064004A00FEFFC60000C03E0000F03F00
5	01000001006172BF3E4D5A374080318D6907CA3EC09BF9546C6A3BE8C04257316BD52C0941000000000000000000000000000000000B30E0000640064004A00FEFFC60000203F0000003F00
6	01000001006172BF3E4D5A374080318D6907CA3EC085A0C4A68117E7C04257316BD52C0941000000000000000000000000000000000B30E0000640064004A00FEFFC60000C03F000048400(
7	01000001006172BF3E4D5A374080318D6907CA3EC0704734E198F3E5C04257316BD52C0941000000000000000000000000000000000B30E0000640064004A00FEFFC6000000BF000080BE00
8	01000001006172BF3E4D5A374080318D6907CA3EC05AEEA31BB0CFE4C04257316BD52C0941000000000000000000000000000000000B30E0000640064004A00FEFFC60000003E000080BF0(
9	01000001006172BF3E4D5A374080318D6907CA3EC044951356C7ABE3C04257316BD52C0941000000000000000000000000000000000B30E0000640064004A00FEFFC60000003E0000803E00
10	01000001006172BF3E4D5A374080318D6907CA3EC02E3C8390DE87E2C04257316BD52C0941000000000000000000000000000000000B30E0000640064004A00FEFFC60000C0BE0000003E0(
11	01000001006172BF3E4D5A374080318D6907CA3EC018E3F2CAF563E1C04257316BD52C0941000000000000000000000000000000000B30E0000640064004A00FEFFC60000C03E000080BE0(
12	01000001006172BF3E4D5A374080318D6907CA3EC0038A62050D40E0C04257316BD52C0941000000000000000000000000000000000B30E0000640064004A00FEFFC6000038C0000028C00(





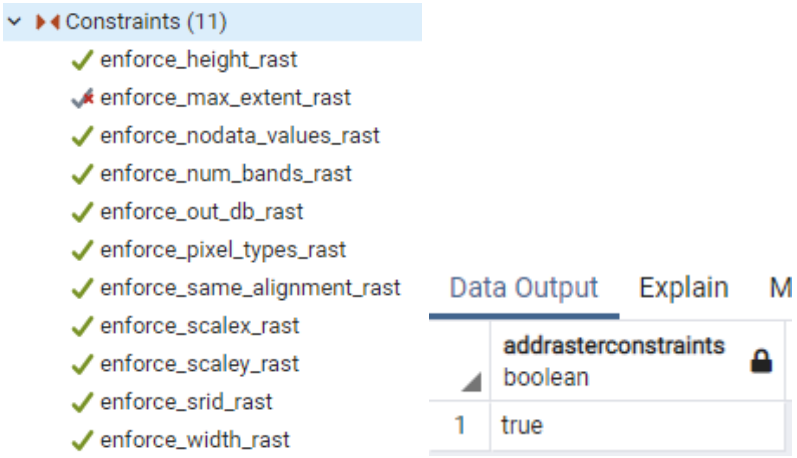
Tworzenie indeksu przestrzennego:

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



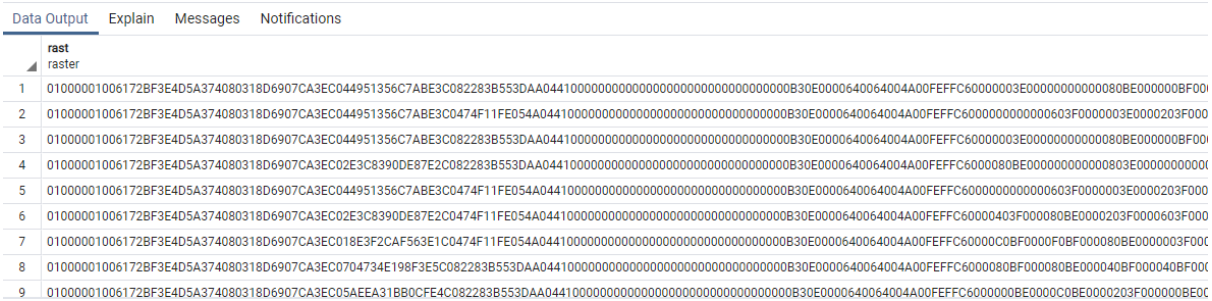
Dodanie constraintów:

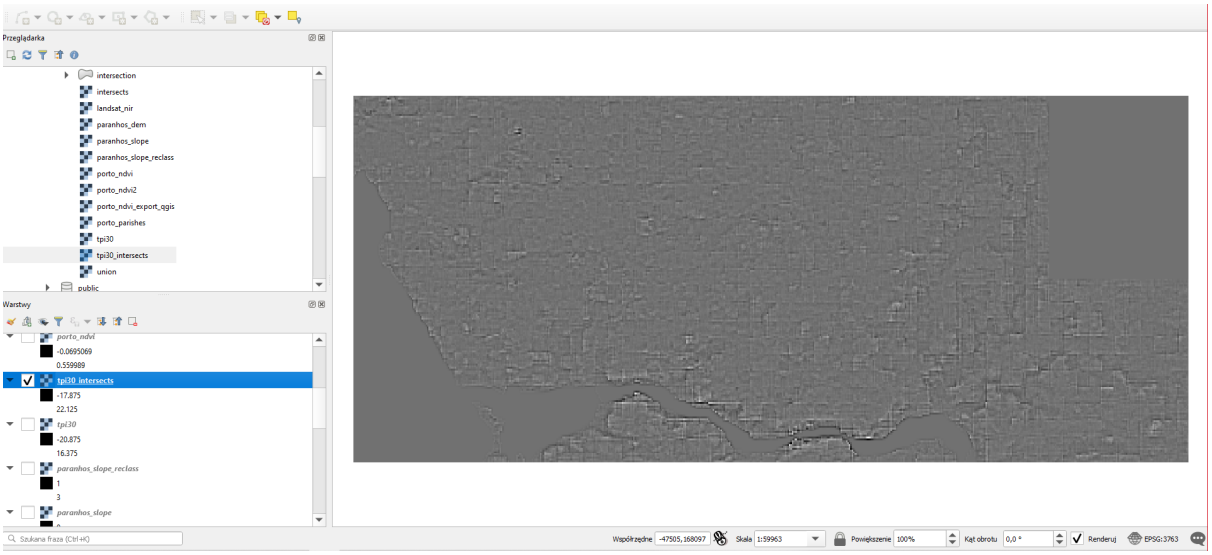
SELECT AddRasterConstraints('pelka'::name, 'tpi30'::name,'rast'::name);



Problem do samodzielnego rozwiązania:

```
CREATE TABLE pelka.tpi30_intersects AS
SELECT ST_TPI(a.rast,1) AS rast
FROM rasters.dem a, vectors.porto_parishes AS b
WHERE ST_Intersects(a.rast, b.geom) AND b.municipality ilike 'porto';

SELECT * FROM pelka.tpi30_intersects;

--Tworzenie indeksu przestrzennego:
CREATE INDEX idx_tpi30_rast_gist_intersects ON pelka.tpi30_intersects
USING gist (ST_ConvexHull(rast));

--Dodanie constraintów:
SELECT AddRasterConstraints('pelka'::name, 'tpi30_intersects'::name,'rast'::name);
```





PORÓWNANIE CZASÓW ZAPYTAŃ					
tpi30			**tpi30_intersects**		
Tworzenie tabeli	Tworzenie indeksu	Constraity	Tworzenie tabeli	Tworzenie indeksu	Constraity
36 secs 682 msec	56 msec	223 msec	1 sec 474 msec	45 msec	88msec

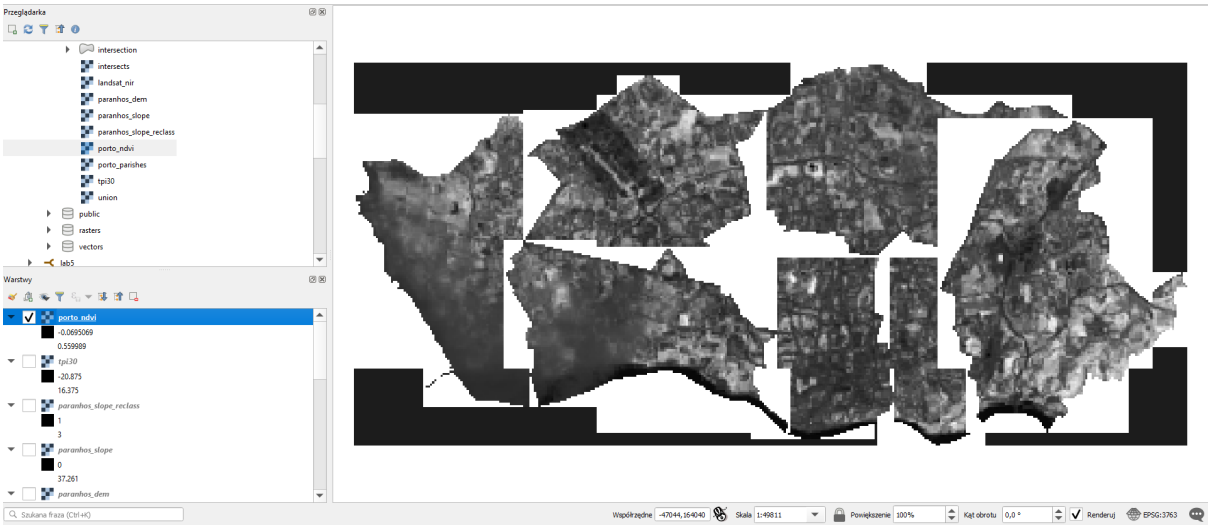
Algebra map:

Przykład 1 - Wyrażenie Algebry Map:

```
CREATE TABLE pelka.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;
```

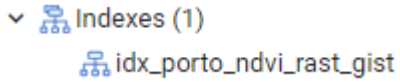
Data Output | Explain | Messages | Notifications | Geometry Viewer

	rid integer	rast raster
1	245	01000001003849EE0BB84F3E404F2001E9AEB43DC0086F60B09B56E3C01F1195122C320441000000000000000000000000000000B30E000001000A0(
2	246	01000001003849EE0BB84F3E404F2001E9AEB43DC03EF15EB9D152E3C075C249A272550441000000000000000000000000000000B30E00002E00490(
3	270	01000001003849EE0BB84F3E404F2001E9AEB43DC075735DC2074FE3C0E4374673AE110441000000000000000000000000000000B30E000032002400(
4	246	01000001003849EE0BB84F3E404F2001E9AEB43DC075B133BEF0E4E2C00224CA124B7E044100000000000000000000000000000B30E00006300750(
5	247	01000001003849EE0BB84F3E404F2001E9AEB43DC0AB0CA038D66DE1C079E13D89743B044100000000000000000000000000000B30E0000100023C
6	270	01000001003849EE0BB84F3E404F2001E9AEB43DC03E4E2033AFB3E2C0E4374673AE110441000000000000000000000000000000B30E00004B001E0(
7	221	01000001003849EE0BB84F3E404F2001E9AEB43DC0C069108BB315E5C0B6C41B00DC900441000000000000000000000000000000B30E00001D00090(
8	244	01000001003849EE0BB84F3E404F2001E9AEB43DC08A8A5008A0B8E5C02644A7A801820441000000000000000000000000000000B30E00002200460
9	245	01000001003849EE0BB84F3E404F2001E9AEB43DC0D2D51D3ACD37E5C0653CEA2E81880441000000000000000000000000000000B30E00004E00530(
10	244	01000001003849EE0BB84F3E404F2001E9AEB43DC0D4DDC44A29E0E6C014B4B85D26800441000000000000000000000000000000B30E00006000770(
11	268	01000001003849EE0BB84F3E404F2001E9AEB43DC0AF24956AEA66E6C0E4374673AE110441000000000000000000000000000000B30E00003600120(
12	221	01000001003849EE0BB84F3E404F2001E9AEB43DC0E31A995A96E6E3C0EC74E7E06D960441000000000000000000000000000000B30E000027000F0(



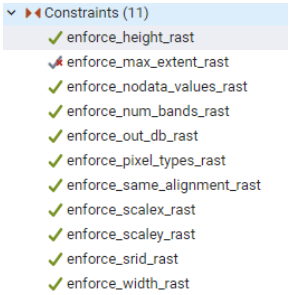
Utworzenie indeksu przestrzennego na wcześniej stworzonej tabeli:

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



Dodanie constraintów:

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



Data Output | Explain | M

	addrasterconstraints boolean
1	true

Przykład 2 – Funkcja zwrotna:

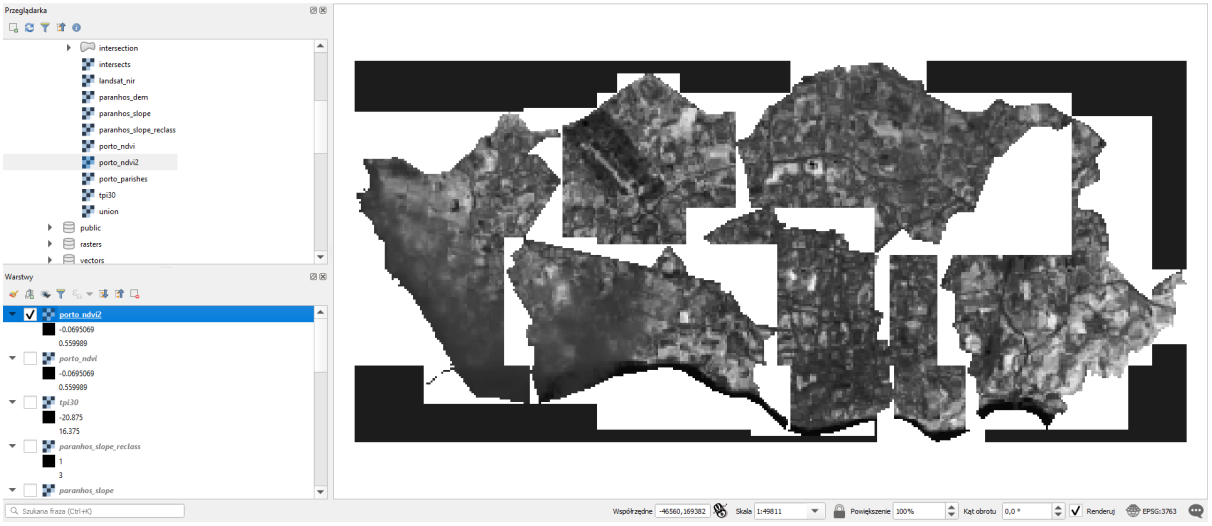
```
CREATE OR REPLACE FUNCTION pelka.ndvi(
value double precision [] [] [],
pos integer [][],
VARIADIC userargs text []
)
RETURNS double precision AS
$$
BEGIN
--RAISE NOTICE 'Pixel Value: %', value [1][1][1];-->For debug purposes
RETURN (value [2][1][1] - value [1][1][1])/(value [2][1][1]+value
[1][1][1]); --> NDVI calculation!
END;
$$
LANGUAGE 'plpgsql' IMMUTABLE COST 1000;
```

Wywołanie funkcji zwrotnej:

```
CREATE TABLE pelka.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],
'pelka.ndvi(double precision[],
integer[],text[])'::regprocedure, --> This is the function!
'32BF'::text
) AS rast
FROM r;
```

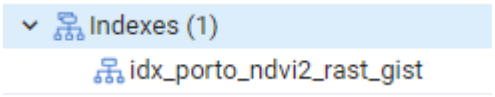
Data Output | Explain | Messages | Notifications | Geometry Viewer

	rid integer	rast raster
1	245	01000001003849EE0BB84F3E404F2001E9AEB43DC0086F60B09B56E3C01F1195122C320441000000000000000000000000000000000B30E000001000A0
2	246	01000001003849EE0BB84F3E404F2001E9AEB43DC03EF15EB9D152E3C075C249A272550441000000000000000000000000000000000B30E00002E00490
3	270	01000001003849EE0BB84F3E404F2001E9AEB43DC075735DC2074FE3C0E4374673AE110441000000000000000000000000000000000B30E00003200240
4	246	01000001003849EE0BB84F3E404F2001E9AEB43DC075B133BEF0E4E2C00224CA124B7E0441000000000000000000000000000000000B30E00006300750
5	247	01000001003849EE0BB84F3E404F2001E9AEB43DC0AB0CA038D66DE1C079E13D89743B0441000000000000000000000000000000000B30E0000100023
6	270	01000001003849EE0BB84F3E404F2001E9AEB43DC03E4E2033AFB3E2C0E4374673AE110441000000000000000000000000000000000B30E00004B001E0
7	221	01000001003849EE0BB84F3E404F2001E9AEB43DC0C069108BB315E5C0B6C41B00DC900441000000000000000000000000000000000B30E00001D0009
8	244	01000001003849EE0BB84F3E404F2001E9AEB43DC08A8A5008A0B8E5C02644A7A80182044100000000000000000000000000000000B30E0000220046
9	245	01000001003849EE0BB84F3E404F2001E9AEB43DC0D2D51D3ACD37E5C0653CEA2E818804410000000000000000000000000000000000B30E00004E0053
10	244	01000001003849EE0BB84F3E404F2001E9AEB43DC0D4DDC44A29E0E6C014B4B85D26800441000000000000000000000000000000000B30E0000600077
11	268	01000001003849EE0BB84F3E404F2001E9AEB43DC0AF24956AEA66E6C0E4374673AE110441000000000000000000000000000000000B30E00003600120
12	221	01000001003849EE0BB84F3E404F2001E9AEB43DC0E31A995A96E6E3C0EC74E7E06D960441000000000000000000000000000000000B30E000027000FC



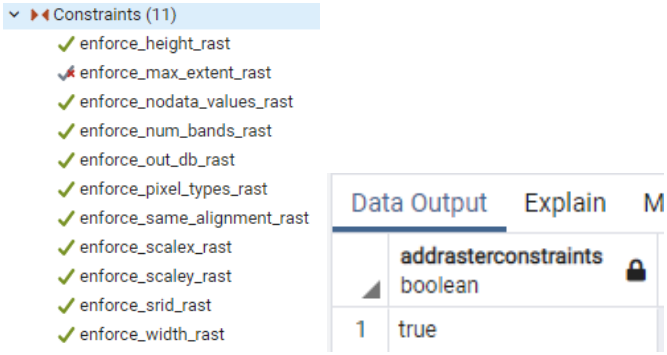
Dodanie indeksu przestrzennego:

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



Dodanie constraintów:

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



Przykład 3 - Funkcje TPI:

tpi:

```
DECLARE
    _rast public.raster;
    _nband integer;
    _pixtype text;
    _pixwidth double precision;
```

```
		_pixheight double precision;
		_width integer;
		_height integer;
		_customextent public.raster;
		_extenttype text;
	BEGIN
		_customextent := customextent;
		IF _customextent IS NULL THEN
			_extenttype := 'FIRST';
		ELSE
			_extenttype := 'CUSTOM';
		END IF;

		IF interpolate_nodata IS TRUE THEN
			_rast := public.ST_MapAlgebra(
				ARRAY[ROW(rast, nband)]::rastbandarg[],
				'public.st_invdistweight4ma(double precision[][][], integer[][],
text[])'::regprocedure,
				pixeltype,
				'FIRST', NULL,
				1, 1
			);
			_nband := 1;
			_pixtype := NULL;
		ELSE
			_rast := rast;
			_nband := nband;
			_pixtype := pixeltype;
		END IF;

		-- get properties
		_pixwidth := public.ST_PixelWidth(_rast);
		_pixheight := public.ST_PixelHeight(_rast);
		SELECT width, height INTO _width, _height FROM
public.ST_Metadata(_rast);

		RETURN public.ST_MapAlgebra(
			ARRAY[ROW(_rast, _nband)]::rastbandarg[],
			' public._ST_tpi4ma(double precision[][][], integer[][],
text[])'::regprocedure,
			_pixtype,
			_extenttype, _customextent,
			1, 1);
	END;
```

tpi4ma:

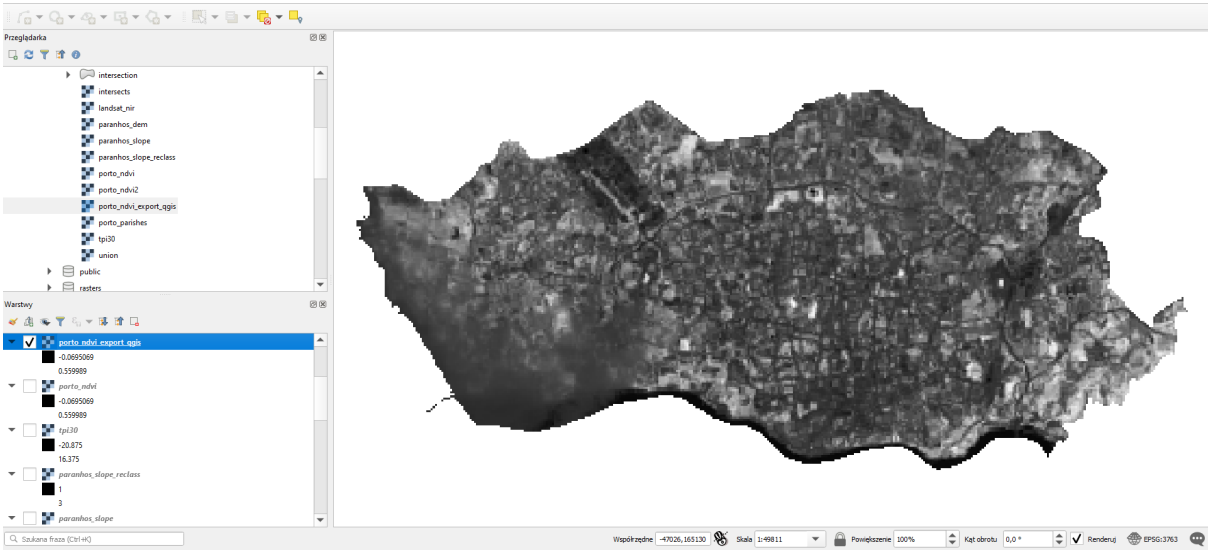
```
	DECLARE
		x integer;
		y integer;
		z integer;

		Z1 double precision;
		Z2 double precision;
		Z3 double precision;
		Z4 double precision;
		Z5 double precision;
		Z6 double precision;
		Z7 double precision;
		Z8 double precision;
		Z9 double precision;

		tpi double precision;
		mean double precision;
		_value double precision[][][];
		ndims int;
	BEGIN
		ndims := array_ndims(value);
		-- add a third dimension if 2-dimension
		IF ndims = 2 THEN
			_value := public._ST_convertarray4ma(value);
		ELSEIF ndims != 3 THEN
			RAISE EXCEPTION 'First parameter of function must be a
3-dimension array';
		ELSE
			_value := value;
		END IF;

		-- only use the first raster passed to this function
		IF array_length(_value, 1) > 1 THEN
			RAISE NOTICE 'Only using the values from the first raster';
		END IF;
		z := array_lower(_value, 1);

		IF (
			array_lower(_value, 2) != 1 OR array_upper(_value, 2) != 3 OR
			array_lower(_value, 3) != 1 OR array_upper(_value, 3) != 3
		) THEN
			RAISE EXCEPTION 'First parameter of function must be a 1x3x3
array with each of the lower bounds starting from 1';
		END IF;

		-- check that center pixel isn't NODATA
```

```
        IF _value[z][2][2] IS NULL THEN
                RETURN NULL;
        -- substitute center pixel for any neighbor pixels that are NODATA
        ELSE
                FOR y IN 1..3 LOOP
                        FOR x IN 1..3 LOOP
                                IF _value[z][y][x] IS NULL THEN
                                        _value[z][y][x] = _value[z][2][2];
                                END IF;
                        END LOOP;
                END LOOP;
        END IF;

        ------------------------------------------------
        --|   Z1= Z(-1,1) |  Z2= Z(0,1)  | Z3= Z(1,1)  |--
        ------------------------------------------------
        --|   Z4= Z(-1,0) |  Z5= Z(0,0) | Z6= Z(1,0)  |--
        ------------------------------------------------
        --|   Z7= Z(-1,-1)|  Z8= Z(0,-1)|  Z9= Z(1,-1)|--
        ------------------------------------------------

        Z1 := _value[z][1][1];
        Z2 := _value[z][2][1];
        Z3 := _value[z][3][1];
        Z4 := _value[z][1][2];
        Z5 := _value[z][2][2];
        Z6 := _value[z][3][2];
        Z7 := _value[z][1][3];
        Z8 := _value[z][2][3];
        Z9 := _value[z][3][3];

        mean := (Z1 + Z2 + Z3 + Z4 + Z6 + Z7 + Z8 + Z9)/8;
        tpi := Z5-mean;

        return tpi;
END;
```

Eksport danych:

Przykład 0 - Użycie QGIS:

```
CREATE TABLE pelka.porto_ndvi_export_qgis AS
SELECT ST_Union(rast)
FROM pelka.porto_ndvi;
```



Przykład 1 - ST_AsTiff:

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

Data Output	Ex
st_astiff bytea	
1	[binary data]

Przykład 2 - ST_AsGDALRaster:

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

Data Output	Explain
st_asgdalraster bytea	
1	[binary data]

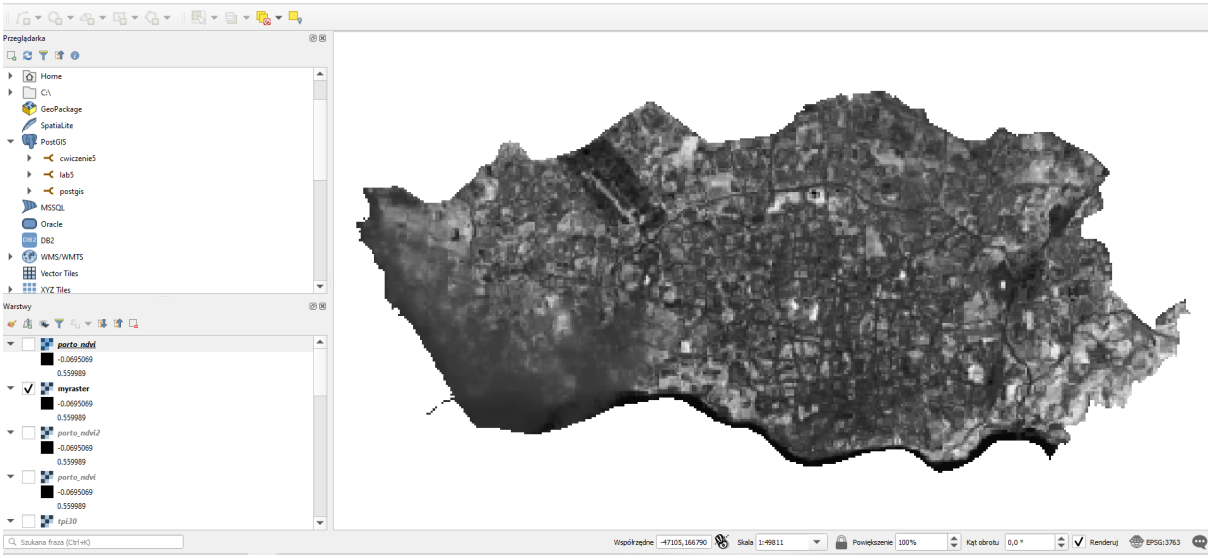
Przykład 3 - Zapisywanie danych na dysku za pomocą dużego obiektu (large object, lo):

```
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 pelka.porto_ndvi;

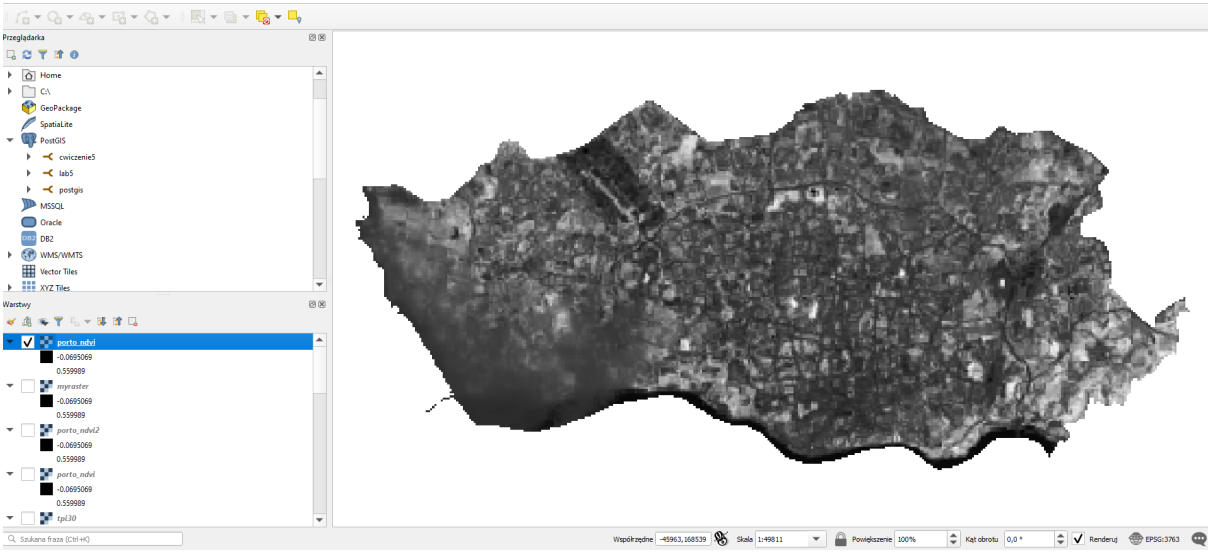
SELECT lo_export(loid, 'C:\Users\48692\Desktop\bazy_lab6\rasters\myraster.tiff') --> Save the file in a place
--where the user postgres have access. In windows a flash drive usualy works fine.
 FROM tmp_out;

SELECT lo_unlink(loid)
FROM tmp_out; --> Delete the large object.



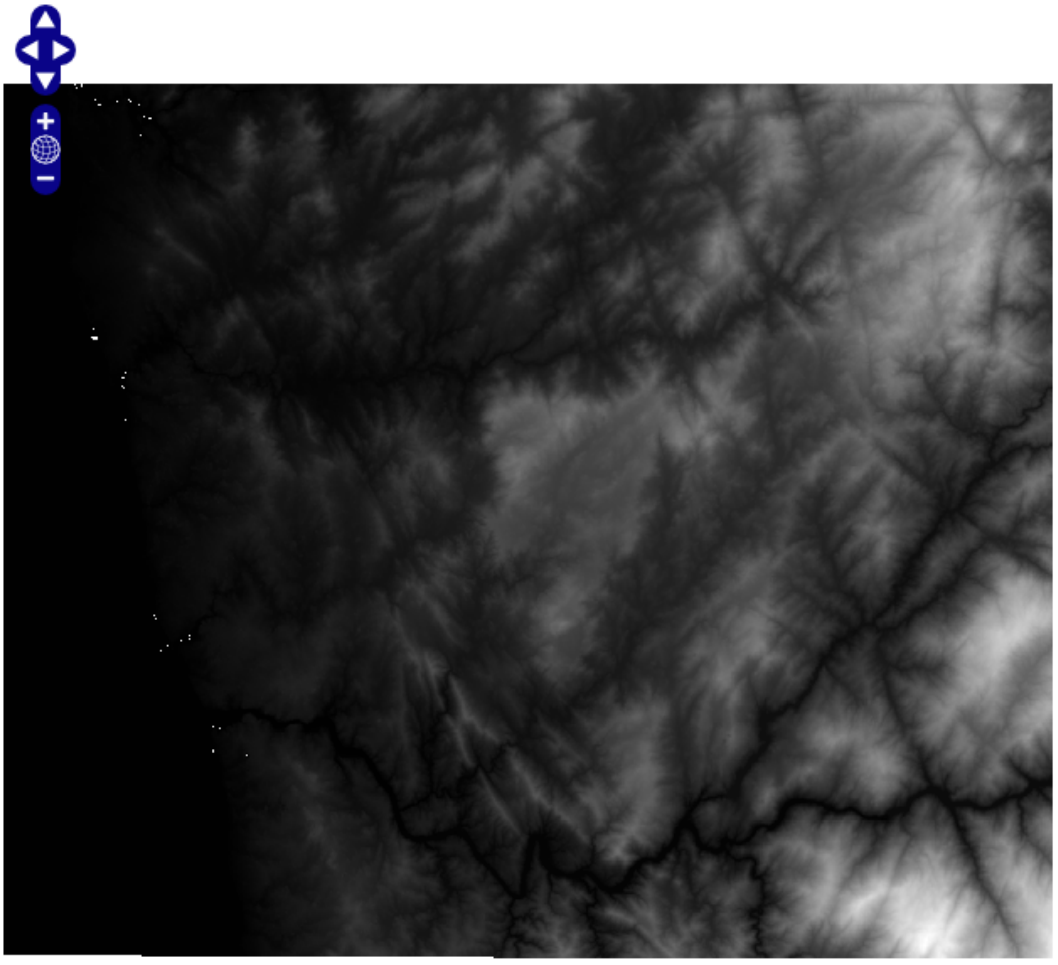
Przykład 4 - Użycie Gdal:

gdal_translate -co COMPRESS=DEFLATE -co PREDICTOR=2 -co ZLEVEL=9
PG:"host=localhost port=5432 dbname=cwiczenie5 user=postgres password=postgis
schema=pelka table=porto_ndvi mode=2"
"C:\Users\48692\Desktop\bazy_lab6\rasters\porto_ndvi.tiff"



Publikowanie danych za pomocą MapServer

http://localhost/cgi-bin/mapserv.exe?map=C:/Users/48692/Desktop/bazy_lab6/pliki/raster.map&MODE=browse&TEMPLATE=openlayers&LAYERS=all



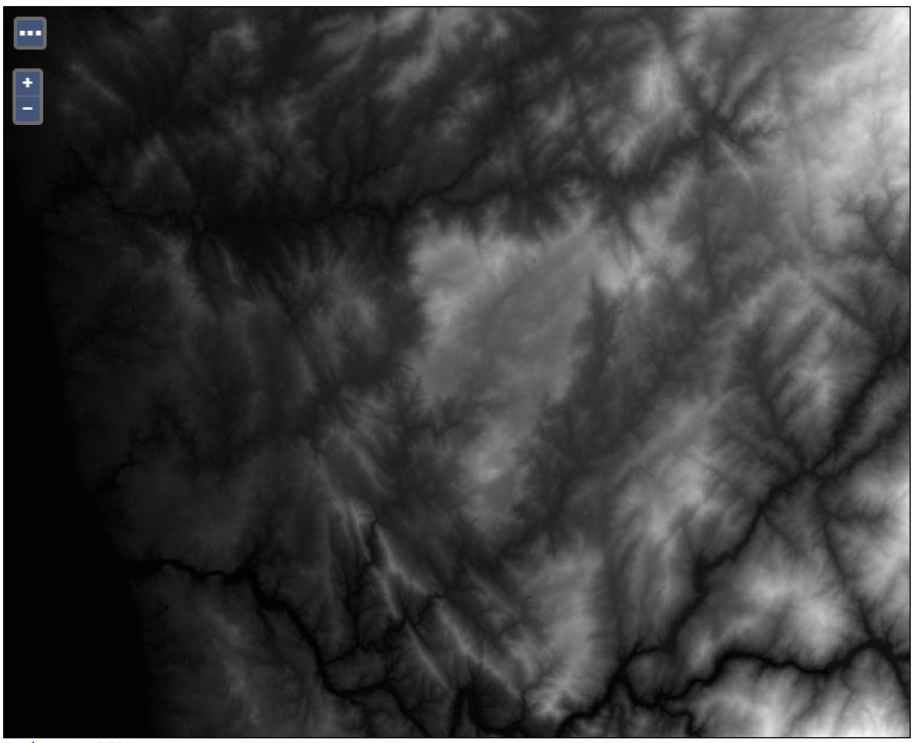
plik .map :

```
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=5433 dbname='cwiczenie6'
                user='postgres' password='postgres' schema='rasters' table='dem' mode='2'"
                PROCESSING "SCALE=AUTO"
                PROCESSING "NODATA=-32767"
                OFFSITE 0 0 0
                METADATA
                'wms_title' 'srtm'
                END
        END
END
```

Publikowanie danych przy użyciu GeoServera

```
CREATE TABLE public.mosaic (
    name character varying(254) COLLATE pg_catalog."default" NOT NULL,
    tiletable character varying(254) COLLATE pg_catalog."default" NOT NULL,
    minx double precision,
    miny double precision,
    maxx double precision,
    maxy double precision,
    resx double precision,
    resy double precision,
    CONSTRAINT mosaic_pkey PRIMARY KEY (name, tiletable)
);

insert into mosaic (name,tiletable) values ('mosaicpgraster','rasters.dem');
```

geoserver 2.17.4, postgres 12.9, postgis 3.0



Scale = 1 : 273K
Click on the map to get feature info

plik connect.pgraster.xml.inc :

```
<connect>
  <dstype value="DBCP"/>
  <username value="postgres"/>
  <password value="postgres"/>
  <jdbcUrl value="jdbc:postgresql://localhost:5433/cwiczenie6"/>
  <driverClassName value="org.postgresql.Driver"/>
```

```
  <maxActive value="10"/>
  <maxIdle value="0"/>
</connect>
```

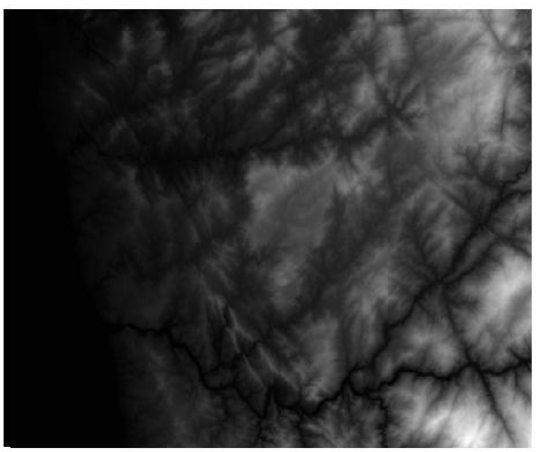
plik mapping.pgraster.xml.inc :

```
<spatialExtension name="pgraster"/>
 <mapping>
     <masterTable name="mosaic" >
            <coverageNameAttribute name="name"/>
            <maxXAttribute name="maxX"/>
            <maxYAttribute name="maxY"/>
            <minXAttribute name="minX"/>
            <minYAttribute name="minY"/>
            <resXAttribute name="resX"/>
            <resYAttribute name="resY"/>
            <tileTableNameAtribute name="tiletable" />
     </masterTable>
     <tileTable>
            <blobAttributeName name="rast" />
     </tileTable>
</mapping>
```

plik mosaicpgraster.pgraster.xml :

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE ImageMosaicJDBCConfig [
     <!ENTITY mapping PUBLIC "mapping"  "mapping.pgraster.xml.inc">
     <!ENTITY connect PUBLIC "connect"  "connect.pgraster.xml.inc">
]>

<config version="1.0">
     <coverageName name="mosaicpgraster"/>
     <coordsys name="EPSG:3763"/>
     <scaleop  interpolation="1"/>
     <axisOrder ignore="false"/>
     &mapping;
     &connect;
</config>
```

publikacja Geotiff:



Scale = 1 : 545K
Click on the map to get feature info