UNIVERSITY OF TWENTE.

PostGIS Raster

Why do I need PostGIS raster ???????

- Multi-users, e.g. one DB for multiple users!!

- Database advantages
  - Raster – Vector interaction.
  - Spatial and non spatial interaction.

- Dynamic analysis using one single SQL query
  - e.g. Using Landsat 8 imagery, what is the maximum NDVI for all municipalities where population is bigger than a specific value?

- Not suitable for all applications
  - Sometimes having your rasters inside a database will not bring any advantage!

Initial remarks

- Raster type is different from geometry type!
  - - The two types can work together.

- More than 140 raster functions and growing! → https://postgis.net/docs/RT_reference.html
  - ST_Clip
  - ST_Slope
  - ST_Reclass
  - ST_MapAlgebra → A super powerful tool!!

- Supports rasters in-database and out-of-db.
  - - In out-of-db mode, raster's will be accessed by postgresql in read only mode!

- **raster2pgsql** is the tool to load an existing raster into a db.

- **Gdal** supports reading postgis rasters and therefore can be used to export and access postgis rasters from outside.

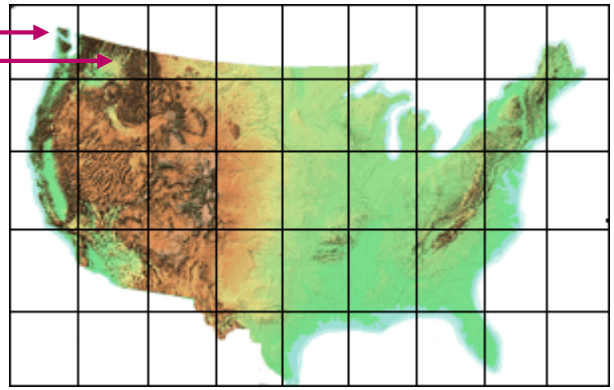
PostGIS raster tiles

- Tiles are not mandatory but most of the time they are recommended.
  - - Depends on the raster and database structure.

- We are going to use tiles.

- In PostGIS raster **each record represents one tile**.

	rid integer	rast raster
1	1	01000001006172BF3E4D5A374080318D6907CA3EC0F25D96820DCBECC04257316BD52C
2	2	01000001006172BF3E4D5A374080318D6907CA3EC0DC0406BD24A7EBC04257316BD52C
3	3	01000001006172BF3E4D5A374080318D6907CA3EC0C6AB75F73B83EAC04257316BD52C
4	4	01000001006172BF3E4D5A374080318D6907CA3EC0B152E531535FE9C04257316BD52C
5	5	01000001006172BF3E4D5A374080318D6907CA3EC09BF9546C6A3BE8C04257316BD52C
6	6	01000001006172BF3E4D5A374080318D6907CA3EC085A0C4A68117E7C04257316BD52C
7	7	01000001006172BF3E4D5A374080318D6907CA3EC0704734E198F3E5C04257316BD52C
8	8	01000001006172BF3E4D5A374080318D6907CA3EC05AEEA31BB0CFE4C04257316BD52C
9	9	01000001006172BF3E4D5A374080318D6907CA3EC044951356C7ABE3C04257316BD52C
10	10	01000001006172BF3E4D5A374080318D6907CA3EC02E3C8390DE87E2C04257316BD52C
11	11	01000001006172BF3E4D5A374080318D6907CA3EC018E3F2CAF563E1C04257316BD52C
12	12	01000001006172BF3E4D5A374080318D6907CA3EC0038A62050D40E0C04257316BD52C
13	13	01000001006172BF3E4D5A374080318D6907CA3EC0DA61A47F4838DEC04257316BD52C



Importing rasters

Raster2pgsql

- http://postgis.refractions.net/docs/using_raster.xml.html#RT_Raster_Loader
- - s : SRID
- - t : tile size
- -d : drop table, create new one and populate it with raster(s)
- - R : Register raster outside BD (filesystem)
- - N : “no data” value
- - I : Adds gist spatial index
- - C : Apply raster constraints -- srid, pixelsize etc. to ensure raster is properly registered in raster_columns view.
- - M : Vacuum analyze the raster table.

Enough Talk Let's Code

The workshop is available at

https://github.com/GIP-ITC-UniversityTwente/workshop-postgis-raster

The database server

Address: *gip.itc.utwente.nl*
Port: *5434*