# PGRaster

Lubia Vinhas

http://trac.osgeo.org/postgis/wiki/WKTRaster/Documentation01

# PostGIS Raster

É um projeto ainda em desenvolvimento que visa desenvolver o suporte a dados raster (matriciais) no PostGIS.

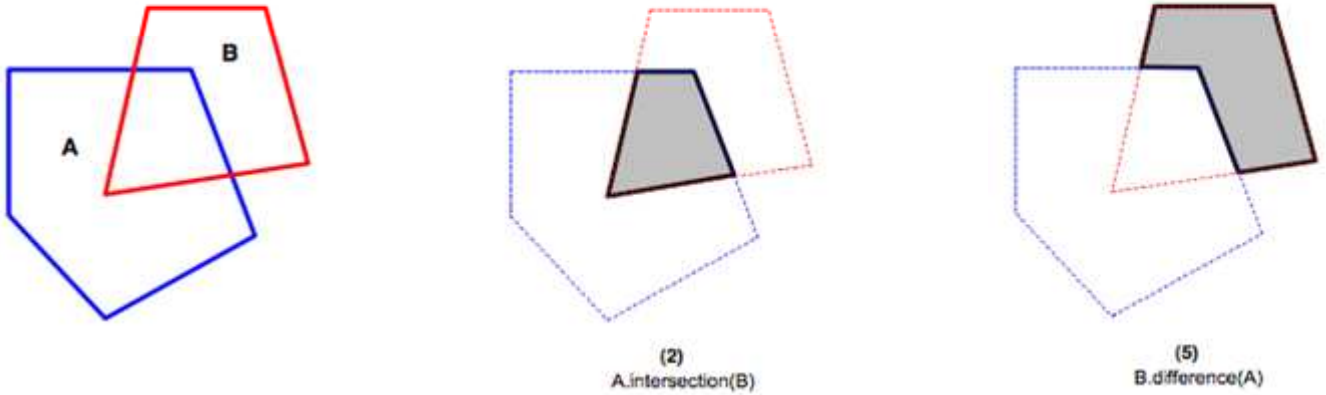
Implementar o tipo RASTER equivalente ao tipo GEOMETRY

Oferecer um conjunto único de funções SQL que operem de maneira uniforme sobre dados matriciais e vetoriais

Parte do PostGIS 2.0

# Arquitetura

Operações sobre dados vetores, operam nas geometria, em geral produzem resultados que são geometrias. Ex:

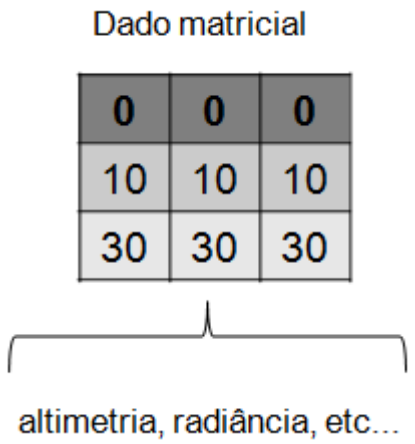



Ou seja, envolvem apenas a componente espacial

# Arquitetura

Dados matriciais, a componente espacial e os valores estão integrados

As operações típicas sobre dados matriciais podem operar sobre a componente espacial, os valores, ou ambos



# WKT Raster

"Coverage": se refere a uma cobertura contínua de uma região por uma representação matricial ou vetorial

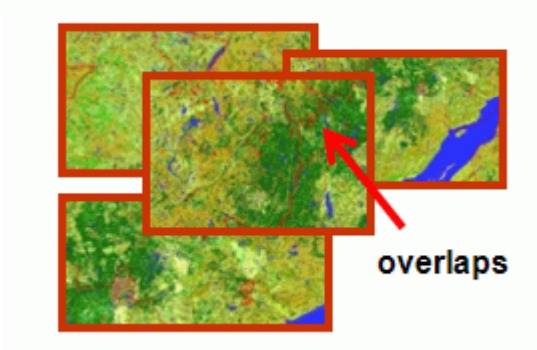
RASTER é mais um tipo permitido para as colunas (assim como o tipo GEOMETRY)

Cada entrada de uma coluna do tipo RASTER, descreve o pedaço lá armazenado (tamanho do pixel, georeferenciamento, largura e altura, número de bandas, no-data por banda, tipo por banda)

Existe uma tabela para guardar o dado raster, uma tabela para armazenar a referência espacial e os metadados

# Possíveis arranjos de tabelas raster

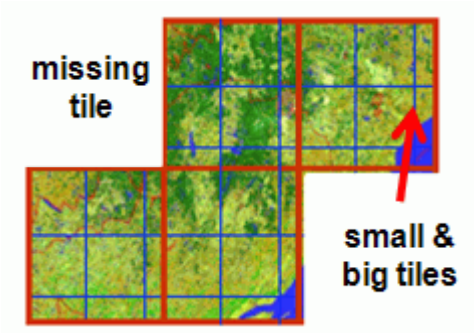
Repositórios de imagens não particionadas (non-tiled) e não relacionadas




| Forma de cada raster | Tiles faltantes | Tiles de mesmo tamanho | Sobreposição de tiles | Formam uma coverage |
|---|---|---|---|---|
| qualquer | sem tiles | sem tales | sim | não |

# Possíveis arranjos de tabelas raster

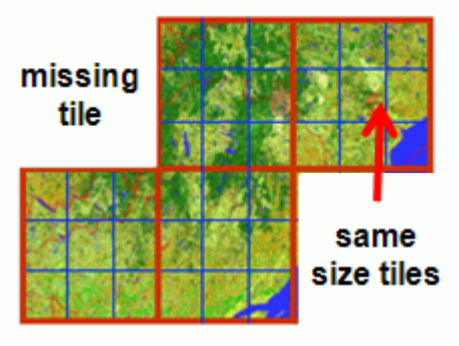
Coverage irregularmente particionada




| Forma de cada raster | Tiles faltantes | Tiles de mesmo tamanho | Sobreposição de tiles | Formam uma coverage |
|---|---|---|---|---|
| qualquer | sim | não | não | sim |

# Possíveis arranjos de tabelas raster

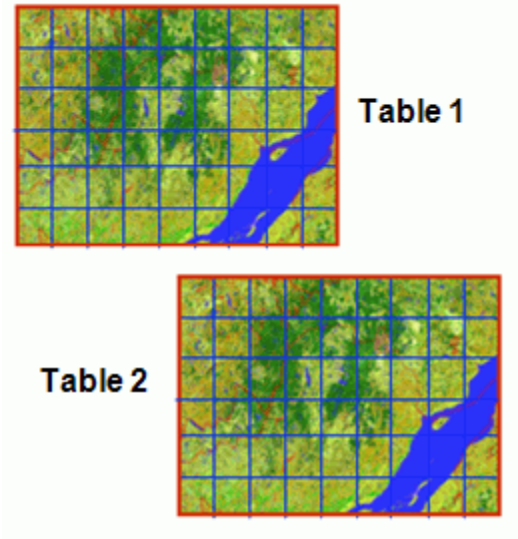
Coverage regularmente particionada




| Forma de cada raster | Tiles faltantes | Tiles de mesmo tamanho | Sobreposição de tiles | Formam uma coverage |
|---|---|---|---|---|
| qualquer | sim | sim | não | sim |

# Possíveis arranjos de tabelas raster

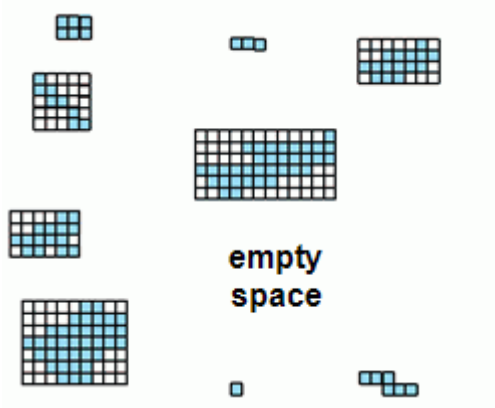
Imagens particionadas (2 tabelas de 54 tiles)




| Forma de cada raster | Tiles faltantes | Tiles de mesmo tamanho | Sobreposição de tiles | Formam uma coverage |
|---|---|---|---|---|
| retangular | não | sim | não | não (em uma única tabela) |

# Possíveis arranjos de tabelas raster

Atributos raster de objetos discretos




| Forma de cada raster | Tiles faltantes | Tiles de mesmo tamanho | Sobreposição de tiles | Formam uma coverage |
|---|---|---|---|---|
| qualquer | sem tiles reais | sem tiles reais | sim | sim |

# Estrutura do tipo `raster`

| Attribute | Description | Storage | Accessible with function... |
|---|---|---|---|
| version | WKB format version (now 0). | unsigned 16 bit integer | Not accessible. |
| number of bands | Number of raster bands stored in the raster. | unsigned 16 bit integer | ST_NumBands() |
| width | Width of the raster. | unsigned 16 bit integer | ST_Width() |
| height | Height of the raster. | unsigned 16 bit integer | ST_Height() |
| **Georeference information** | | | |
| pixelsizex | Pixel size in the x-direction in the same map units as the coordinate system. Same as parameter A in a world file. | 64 bit double | ST_PixelSizeX() |
| pixelsizey | Pixel size in the y-direction in the same map units as the coordinate system. Same as parameter E in a world file. | 64 bit double | ST_PixelSizeY() |
| upperleftx | X-coordinate of the center of the upper left pixel. Same as parameter C in a world file. | 64 bit double | ST_UpperLeftX() |
| upperlefty | Y-coordinate of the center of the upper left pixel. Same as parameter F in a world file. | 64 bit double | ST_UpperLeftY() |
| rotationx | Rotation about x-axis. Same as parameter B in a world file. | 64 bit double | ST_RotationX() |
| rotationy | Rotation about y-axis. Same as parameter D in a world file. | 64 bit double | ST_RotationY() |
| srid | Spatial reference id. | 32 bit integer | ST_SRID() |

# Estrutura do tipo `raster`

| Attribute | Description | Storage | Accessible with function... |
|---|---|---|---|
| **Band information** (one set per band) | | | |
| isoffline | Flag specifying if the band data is stored in the database or as a file in the file system. | 1 bit | If ST_BandPath() return an empty string, the data is stored in the database. |
| hasnodatavalue | Flag specifying if the stored nodatavalue is significant or not. | 1 bit | ST_BandHasNoDataValue() |
| pixeltype | Pixel type for this band. | 4 bits | ST_BandPixelType() |
| nodatavalue | Nodata value for the band. | Depend on band pixel type. | ST_BandNodataValue() |
| **Band data** (one set per band) for **in-db** raster | | | |
| values[] | Value of each pixel. | Depends on band pixel type. | ST_ Value() |
| **Band data** (one set per band) for **out-db** raster | | | |
| bandnumber | Number of the out-db band. | unsigned 8 bit integer | Not accessible yet. |
| path | Path to the out-db raster file. | string | ST_BandPath() |

# raster_columns

| Attribute | PostgreSQL Type | Description |
|---|---|---|
| **r_table_catalog** | character varying(256) | Name of the catalog containing the table. This attribute exists only to follow the geometry_column table schema. It is generally left empty. |
| **r_table_schema** | character varying(256) | Name of the schema containing the table. Generally equal to "public". |
| **r_table_name** | character varying(256) | Name of the table containing a column of type raster. |
| **r_column** | character varying(256) | Name of the raster column in the table. |
| **srid** | integer | ID of the spatial reference system used for the raster column in this table. It is a foreign key reference to the PostGIS spatial_ref_sys table. |
| **pixel_types** | array[varchar] | Array of pixel types; one for every band. Index for first band is 1. Types are expressed as varchar (string) values:<br>"1BB" = 1-bit boolean<br>"2BUI" = 2-bit unsigned integer<br>"4BUI" = 4-bit unsigned integer<br>"8BSI" = 8-bit signed integer<br>"8BUI" = 8-bit unsigned integer<br>"16BSI" = 16-bit signed integer<br>"16BUI" = 16-bit unsigned integer<br>"32BSI" = 32-bit signed integer<br>"32BUI" = 32-bit unsigned integer<br>"32BF" = 32-bit float<br>"64BF" = 64-bit float |
| **out_db** | array[boolean] | Array of flags for in-db or out-db storage; one per band. True when raster data is not stored in the database but resides as files in the filesystem. Paths to these files (one per row) are determined by ST_BandPath(). Index for first band is 1. |
| **regular_blocking** | boolean | Flag specifying that the regular blocking constraint is enforced for this table. |
| **nodata_values** | array[double] | Array of nodata values; one for each band. Each nodata value is stored as a double (even when the pixel type is integer). Index for first band is 1. |
| **pixelsize_x** | double | Pixel size of the raster in the x-direction. In the units of the coordinate system determined by srid. |
| **pixelsize_y** | double | Pixel size of the raster in the y-direction. In the units of the coordinate system determined by srid. |
| **blocksize_x** | integer | Width of a block of raster data when regular_blocking is set to true. NULL otherwise. |
| **blocksize_y** | integer | Height of a block of raster data when "regular_blocking" is set to true. NULL otherwise. |
| **extent** | geometry | Polygon geometry encompassing all raster rows of this table. NULL if predefined bounds are not known. When "regular_blocking" is set to true this polygon is a simple rectangle. In other cases it might be an irregular polygon. |

# Referência

Consultar o manual

http://postgis.refractions.net/documentation/manual-svn/RT_reference.html

# Tutorial

Baseado no tutorial disponível em:
http://trac.osgeo.org/postgis/wiki/WKTRasterTutorial01

Dados:

dado SRTM sobre o município de Ilhabela do projeto Topodata (http://www.dsr.inpe.br/topodata/)

pontos aleatórios criados sobre a extensão do dado matricial

Software:

- PostGIS 2.0
- OpenJump

# Inserindo o dado matricial no PostGIS

1. Usar o plugin para o Quantun GIS desenvolvido pelo Mauricio de Paulo: http://mapeandoobrasil.blogspot.com/2010/12/postgis-raster-plugin-para-qgis.html
2. Usar o scritp python:
   a. raster2pgsql.py -s 4326 -I -r ilhabela.tif -F public.testerst -c -t altimetria -k 64x64 -o ilhabela.sql
   b. Abrir o arquivo “ilhabela.sql” no pgAdmin e rodar a consulta
   c. shp2pgsql -s 4326 -I PontosInteresse.shp > pontosinteresse.sql
   d. Abrir o arquivo “pontosinteresse.sql” no pgAdmin e rodar a consulta

# Usando o OpenJump

Abrir a interface "Layer->Run Datastore Query“

Criar uma conexão para o PostGIS

Usar as consultas:

select gid, ST_AsBinary(geom) from pontosinteresse;

# Usando a extensão

```
SELECT rid, (stats).*
FROM (SELECT rid, ST_SummaryStats(altimetria.rast) As stats
   FROM altimetria) As foo;

SELECT rid, ST_Value(rast,1,pontosinteresse.geom) As b1pval
FROM pontosinteresse, altimetria
WHERE ST_Intersects(altimetria.rast,pontosinteresse.geom);

SELECT rid, ST_Value(rast,foo.geom) As b1pval
FROM altimetria
CROSS JOIN (SELECT pontosinteresse.geom FROM pontosinteresse) AS foo;
```

# Usando a extensão

```
create table ptos_buffer as
select gid, st_buffer(geom,0.001) as geom from pontosinteresse;
```

```
SELECT SUM(ST_Area((gv).geom)*(gv).val)/SUM(ST_Area((gv).geom))
FROM (
  SELECT ST_Intersection(r.rast,1, p.geom) As gv
  FROM ptos_buffer As p INNER JOIN altimetria As r
  ON ST_Intersects(p.geom, r.rast)
  WHERE p.gid = 1) As foo;
```