Spatial Database Management
GEP 664 / GEP 380
Class #12: Rasters, Other Database Formats

Frank Donnelly

Dept of EEGS, Lehman College CUNY

Spring 2019

Today's Topics

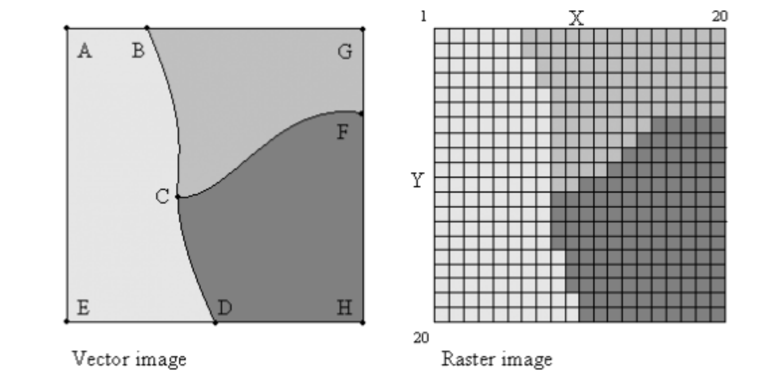
Rasters

SQLite / Spatialite

ArcGIS

Next Class

Rasters



Vector image

Raster image

Vector		
Polygon ID	Coordinates	Soil Type
1	A,B,C,D,E	Chalk
2	B,C,F,G	Clay
3	C,F,H,D	Gravel

Raster	
Grid Ref.	Item
x=1, y=1	Chalk
X=2, y=1	Chalk
X=3, y= 1	Chalk
X=4 ... etc.	...
X=20, y=20	Gravel

Image from http://www.arts-humanities.net/wiki/gis_geographic_information_system_archaeology

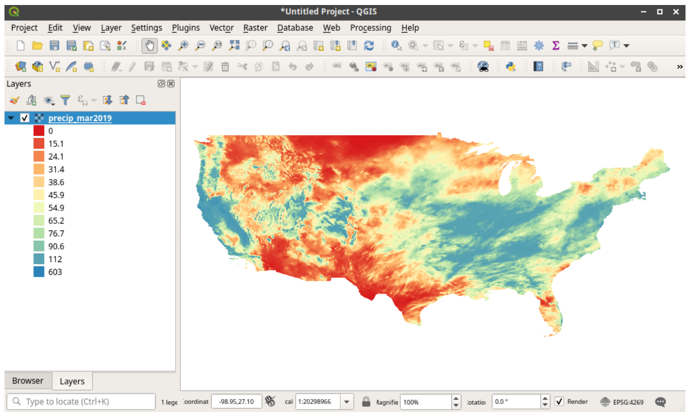
Rasters in PostGIS

Spatial databases are largely a vector-based world. Raster support is recent.

- All imported data is converted and stored in a PostGIS raster format
- Rasters have their own internal tables (raster_columns) and functions
- Tables typically consist of rows where each row is a tile, and the raster column for that row contains all the pixels
- Alternative to storing raster in the database is storing it outside, with a reference table in the db

Example - PRISM Data

We'll use some monthly precipitation data from PRISM as an example - `http://prism.oregonstate.edu/`



Importing Data

Use the raster2pgsql command line tool, stored in the bin folder in your installation of PostgreSQL.

- Windows users: this would be in: Program Files - PostgreSQL - 10 - bin
- Windows users must navigate to that folder in the command line to run the program
- Mac and Linux users can execute the program from any location in the shell
- Make life simpler: move import files to a temporary folder near the top of your directory tree
- The GDAL command line tool can be used to get info about rasters and do preprocessing before import

raster2pgsql

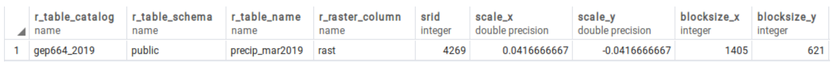
Use switches to set options. Type raster2pgsql to see them all.

```
raster2pgsql -s 4269 -C
C:\workspace\prism\PRISM_ppt_provisional_4kmM3_201903_bil.bil
precip_mar2019 | psql -h localhost -U postgres -p 5432 -d gep664_2019
```

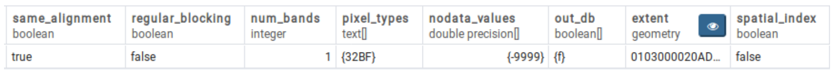
- Run the tool, -s specifies the SRID for the layer, -C enforces common db constraints
- (adding -R after srid would keep the file outside the db)
- Provide full path to the import file, followed by name of new table in the db (can also specify schema.table)
- Add a pipe | followed by switches to connect to the database in psql: -h host, -U username, -p port, -d database

Raster Column

```
SELECT *
FROM raster_columns;
```



	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
1	gep664_2019	public	precip_mar2019	rast	4269	0.0416666667	-0.0416666667	1405	621

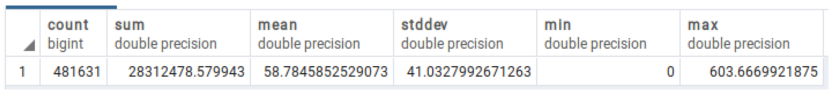


same_alignment boolean	regular_blocking boolean	num_bands integer	pixel_types text[]	nodata_values double precision[]	out_db boolean[]	extent geometry	spatial_index boolean
true	false	1	{32BF}	{-9999}	{f}	0103000020AD...	false

Pixel Statistics - Summaries

ST_SummaryStats takes the raster band as input

```
SELECT (stats).*
FROM (
SELECT ST_SummaryStats(rast,1) AS stats
FROM precip_mar2019)
AS summary;
```

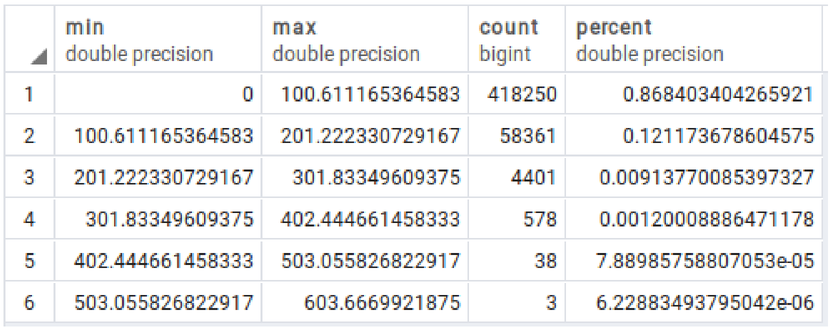


	count bigint	sum double precision	mean double precision	stddev double precision	min double precision	max double precision
1	481631	28312478.579943	58.7845852529073	41.0327992671263	0	603.6669921875

Pixel Statistics - Histogram

ST_Histogram takes the raster band and number of summary buckets as input

```
SELECT (stats).*
FROM (
SELECT ST_Histogram(rast,1,6) AS stats
FROM precip_mar2019)
AS summary;
```



	min double precision	max double precision	count bigint	percent double precision
1	0	100.611165364583	418250	0.868403404265921
2	100.611165364583	201.222330729167	58361	0.121173678604575
3	201.222330729167	301.83349609375	4401	0.00913770085397327
4	301.83349609375	402.444661458333	578	0.00120008886471178
5	402.444661458333	503.055826822917	38	7.88985758807053e-05
6	503.055826822917	603.6669921875	3	6.22883493795042e-06

Creating and Clipping

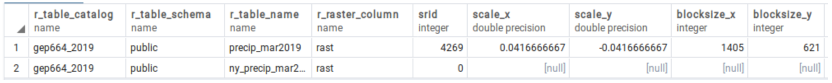
Clip the raster using geometry of state boundaries (from the Census TIGER files) and store in a new table. Spatial index must be created on convex hull of the raster.

```
CREATE TABLE ny_precip_mar2019 (
rid serial PRIMARY KEY,
rast raster);

INSERT INTO ny_precip_mar2019(rid, rast)
SELECT p.rid, ST_CLIP(p.rast, s.geom)
FROM precip_mar2019 p, state_bndy s
WHERE s.stusps='NY';

CREATE INDEX ny_precip_mar2019_idx ON ny_precip_mar2019
USING gist( ST_ConvexHull(rast) );
```

Adding Constraints

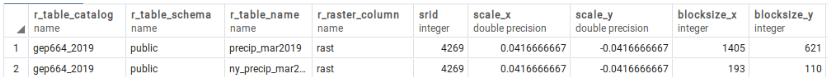
Check the raster catalog after loading and no constraints exist. Add constraints to update the catalog, and check again.

```
SELECT * FROM raster_columns;
```



	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
1	gep664_2019	public	precip_mar2019	rast	4269	0.0416666667	-0.0416666667	1405	621
2	gep664_2019	public	ny_precip_mar2...	rast	0	[null]	[null]	[null]	[null]

```
SELECT AddRasterConstraints('ny_precip_mar2019'::name,
'rast'::name);
SELECT * FROM raster_columns;
```

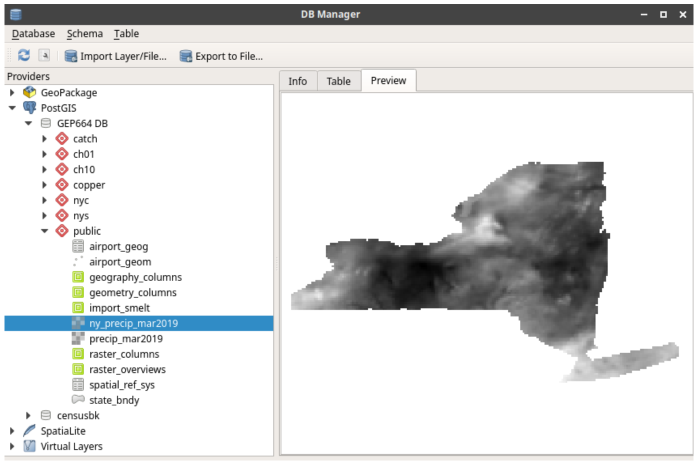


	r_table_catalog name	r_table_schema name	r_table_name name	r_raster_column name	srid integer	scale_x double precision	scale_y double precision	blocksize_x integer	blocksize_y integer
1	gep664_2019	public	precip_mar2019	rast	4269	0.0416666667	-0.0416666667	1405	621
2	gep664_2019	public	ny_precip_mar2...	rast	4269	0.0416666667	-0.0416666667	193	110

Constraints are applied to the entire table; if you are loading multiple rasters (tiles) into one table, don't apply constraints until everything is loaded.

QGIS Support

Rasters cannot be added to projects from the Browser or through the Add PostGIS layers interface. Use the Database Manager to preview and add raster layers.



Raster Processing

These functions modify the underlying pixels of the raster.

ST_Transform	convert from one SRS for another
ST_Rescale	changes pixel size by specifying specific pixel size
ST_Resize	similar to rescale, except you specify percentage of the original
ST_Resample	changes pixel size by specifying width and height for entire raster
ST_Reclass	change the actual value of the pixels

For expanded details on raster analysis and processing, see *PostGIS in Action* Chapter 12 (Chapter 7 just covers the basics)

Conversion and Exporting

- ST_AsRaster converts vectors to rasters
- There are several functions for converting rasters to vectors. Create convex hulls, envelopes, or actual polygons.
- There are several functions for exporting rasters. There are specific functions for common image formats (like ST_AsTiff) or the ST_AsGDALRaster for 20 other formats.

Today's Topics

Rasters

SQLite / Spatialite

ArcGIS

Next Class

SQLite

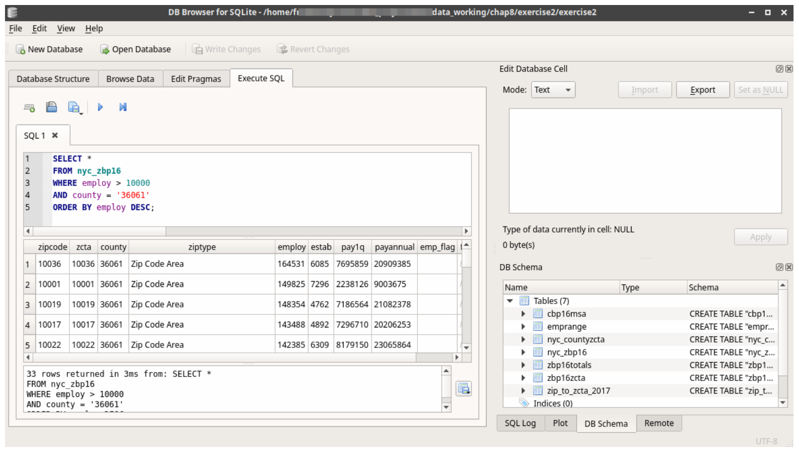
SQLite is a public-domain, file-based database that was specifically created for easily deploying and embedding databases in software applications.

- Originally released in 2000, widely used
- Implements most of the SQL-92 standard
- Uses PostgreSQL as a reference platform
- Does not use a client-server database engine; it is embedded into end programs or used as a stand-alone database
- Uses data type affinities for columns, rather than strict types

SQLite Tools

SQLite : Command-line program from the project developers

DB Browser for SQLite : Free desktop program



DB Browser for SQLite - /home/f[redacted]data_working/chap8/exercise2/exercise2

```
SELECT *
FROM nyc_zbp16
WHERE employ > 10000
AND county = '36061'
ORDER BY employ DESC;
```

	zipcode	zcta	county	ziptype	employ	estab	pay1q	payannual	emp_flag
1	10036	10036	36061	Zip Code Area	164531	6085	7695859	20909385	
2	10001	10001	36061	Zip Code Area	149825	7296	2238126	9003675	
3	10019	10019	36061	Zip Code Area	148354	4762	7186564	21082378	
4	10017	10017	36061	Zip Code Area	143488	4892	7296710	20206253	
5	10022	10022	36061	Zip Code Area	142385	6309	8179150	23065864	

```
33 rows returned in 3ms from: SELECT *
FROM nyc_zbp16
WHERE employ > 10000
AND county = '36061'
```

Edit Database Cell — Mode: Text

Type of data currently in cell: NULL

0 byte(s)

DB Schema

Name	Type	Schema
Tables (7)		
cbp16msa		CREATE TABLE "cbp1...
emprange		CREATE TABLE "empr...
nyc_countyzcta		CREATE TABLE "nyc_c...
nyc_zbp16		CREATE TABLE "nyc_z...
zbp16totals		CREATE TABLE "zbp1...
zbp16zcta		CREATE TABLE "zbp1...
zip_to_zcta_2017		CREATE TABLE "zip_t...
Indices (0)		

Spatialite

Spatialite is the open source spatial extension to SQLite, similar to how PostGIS is the spatial extension to PostgreSQL.

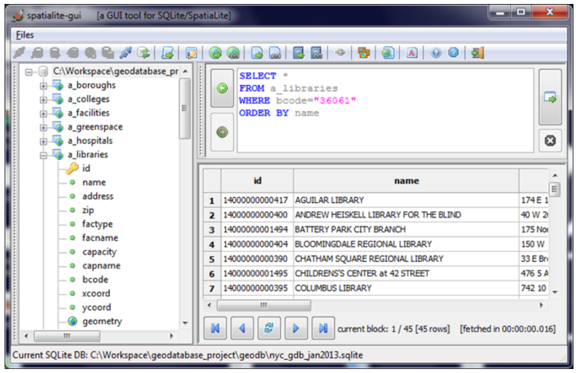
- Originally released in 2008
- Uses the same OGC standards for spatial SQL as PostGIS
- Roughly equivalent to PostGIS in supporting vector geometry
- No equivalent geography type, but there are functions for calculating geodetic distance
- Limited (but growing) support for rasters and topology

Spatialite Tools

Spatialite CLI : Command-line program from the project developers

Spatialite GUI : Graphic interface from the project developers

QGIS : Through the Database Manager



spatialite-gui [a GUI tool for SQLite/SpatiaLite]

```
SELECT *
FROM a_libraries
WHERE bcode="36061"
ORDER BY name
```

C:\Workspace\geodatabase_pr

- a_boroughs
- a_colleges
- a_facilities
- a_greenspace
- a_hospitals
- a_libraries
  - id
  - name
  - address
  - zip
  - factype
  - facname
  - capacity
  - capname
  - bcode
  - xcoord
  - ycoord
  - geometry

	id	name	
1	14000000000417	AGUILAR LIBRARY	174 E 1
2	14000000000400	ANDREW HEISKELL LIBRARY FOR THE BLIND	40 W 2
3	14000000001494	BATTERY PARK CITY BRANCH	175 No
4	14000000000404	BLOOMINGDALE REGIONAL LIBRARY	150 W
5	14000000000390	CHATHAM SQUARE REGIONAL LIBRARY	33 E Br
6	14000000001495	CHILDRENS'S CENTER at 42 STREET	476 5 A
7	14000000000395	COLUMBUS LIBRARY	742 10

current block: 1 / 45 [45 rows] [fetched in 00:00:00.016]

Current SQLite DB: C:\Workspace\geodatabase_project\geodb\nyc_gdb_jan2013.sqlite

SQLite Pluses

- Very easy to create and deploy
- File-based, so easy to copy and move around
- Provides many of the relational database benefits that shapefiles lack
- Provides the ability to do SQL and spatial SQL
- Can easily be tapped into with scripting and programming languages

SQLite Minuses

- Not intended for direct multi-user access over a network
- Size limitations on files and tables
- Implements just a subset of SQL language
- Spatial support is largely limited to geometry type
- Spatial indexes must be called explicitly
- Limited documentation / tutorials for Spatialite

SQLite Features Not Supported

Nothing beyond the SQL-92 standard, and:

- No schemas
- No right or full outer joins
- Limited support for ALTER TABLE (you can only rename and add columns)
- No GRANT and REVOKE as permissions can only be set at the file level, not for users or objects
- No strict data types

Today's Topics

Rasters

SQLite / Spatialite

ArcGIS

Next Class

ArcGIS Formats

The ArcGIS formats are proprietary

- Personal Geodatabase is built on MS Access, and is no longer supported
- File Geodatabase is the default, entirely self-contained
- ArcSDE links ArcGIS to a number of large, proprietary databases (SQL Server, Oracle) as well as PostgreSQL

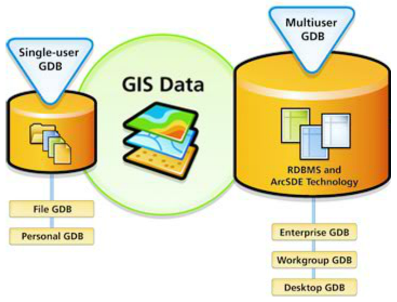
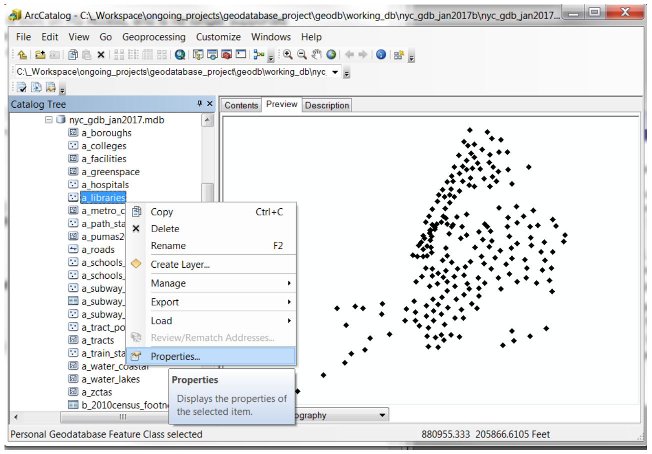


Image source: `http://www.esriuk.com/software/arcgis/geodatabase`

ArcGIS Interface

Interface for working with File and Personal Geodatabases is the ArcCatalog



Geodatabase Pluses

The file geodatabase offers many advantages over shapefiles:

- Gather features and attributes in one container
- Can handle vectors, rasters, and topology
- Easier to enforce integrity and entity constraints
- Create domains, subtypes, and indexes
- Explicitly link features together in relationship classes
- Easy to use GUI interface in ArcCatalog
- Supports access for multiple readers (but not writers)

Geodatabase Minuses

The file geodatabase has drawbacks relative to PostGIS

- Cannot do any SQL or spatial SQL; must use internal Arc tools or an enterprise-level db via ArcSDE
- Issues with backwards incompatibility and forced obsolescence
- As a proprietary format, it does not work well with other open source GIS software

PostGIS Support

You can connect to PostGIS and Spatialite databases via the ArcCatalog (from 10.2 forward), view data, overlay data with other file types, and perform analysis. Creating objects or db administration is problematic or not possible.

PostgreSQL is supported via enterprise-level ArcSDE (the only FOSS option; other options are proprietary).

PostgreSQL Beyond pgAdmin

Remember: PostgreSQL / Postgis is independent from a specific interface. Use what works best. External PostGIS tools (like the shapefile loader) can always be used outside.

psql	tried and true command-line, takes practice
pgAdmin 3	previous GUI for many years, no-longer supported after PostgreSQL 9.6 (but still alive)
pgAdmin 4	new version released with PostgreSQL 9.6
phpPgAdmin	common web-based client
DBeaver	one of many independent GUI tools
QGIS	with the database manager
OpenJUMP	Favorite GIS package of the *PostGIS in Action* authors for PostGIS

Today's Topics

Rasters

SQLite / Spatialite

ArcGIS

Next Class

Due Next Class

The following are due at the beginning of our next class:

Nothing! There are no readings or assignments.

We'll work with Python in our next class.