Everything About PostGIS


paul.ramsey@crunchydata.ca
Motivation
I want a computer map!
By golly, my friend, you need to buy a... GIS!
Wait! Stop!

- Do you have PostgreSQL?
- Do you have PostGIS?

You already have a GIS!
What? (is PostGIS?)
“spatial” database
• **Types**
  ○ string, float, date

• **Indexes**
  ○ b-tree, hash

• **Functions**
  ○ strlen(text)
  ○ pow(float, float)
  ○ now()
• **Spatial** Types
  ○ geometry, geography

• **Spatial** Indexes
  ○ t-tree, quad-tree, kd-tree

• **Spatial** Functions
  ○ ST_Length(geometry)
  ○ ST_Buffer(geometry, R)
  ○ ST_X(geometry)
Reasoning by Analogy

PostGIS is to PostgreSQL as ORACLE SPATIAL is to ORACLE.
Spatial Database Marketplace

PostGIS

Oracle Spatial

And everybody else:

- SpatiaLite
- BigQuery
- Teradata
- Netezza
- OmniSci
- ....
Spatial Database Marketplace

Open Geospatial Consortium
“Simple Features for SQL Version 1.2”

International Standards Organization
SQL/MM Part 3: Spatial/Temporal
We now support the following geographic types: Point, Linestring, Polygon, Multi-polygon, and Collections. Functions will look and feel very familiar to users of PostGIS as they follow the SQL/MM Spatial standard.
Why? (PostGIS?)
Enterprise Architecture

Why does City of Metropolis manage their spatial data as a collection of files?
Enterprise Architecture

- Central point of truth
- Scalable access
- Data integrity and transactional control
- Common access pattern (SQL)
Enterprise Database

- **Integration with third party applications**
  - Desktop
  - Middleware
  - Scripting

- **Baseline Functionality**
  - Spatial types to support business
  - Spatial indexes to support business

- **Enterprisey-ness**
  - Scalability, security, reliability, support
PostGIS Desktop Integration
PostGIS Middleware Integration
PostGIS Language Integration

Ruby, C, C++, DCOM, JDBC, ODBC, node.js, Scala, Groovy, Mono, Haskell, Rust, ...
Example PostGIS Hybrid Architecture
PostGIS Enterprise Reference Users (Gov’t)
PostGIS Enterprise Reference Users (Corp)
PostGIS Cloud Users

- Google Cloud Platform
- Azure
- Amazon RDS
- Google Cloud SQL PostgreSQL
PostGIS Cloud Options

Your own managed cluster?

- Database-as-a-service
- Crunchy PostgreSQL “operator” for k8s
- Deploy on any cloud

Deployed on?

- Google Cloud Platform
- Azure
- Amazon web services
- Server hosting
History
“Managing changing data in shape files is a pain in the _____!”

[Diagram showing SHP files and a FGDB container]
BC Watershed Atlas

BC Digital Road Atlas

distance() indexes (v0.1)

BC Corporate Watersheds

“lightweight” geometry (v1.0)

“we need a spatial type!”
GIS without the GIS
GIS without the GIS

- Move **workload** from low-performance scripts to high-performance SQL
- Pipe analysis directly from database to the web, **without spatial middleware**
- Give developers **direct access** to powerful GIS analytics engine
How? (PostGIS?)
CREATE EXTENSION postgis;
it's a feature frenzy!!! (ack!)
freedom!
N*SQL
is the ultimate hotness
YeSQL!
GIS without the GIS
“What parcels are within 1km of the fire?”
SELECT owner_phone
FROM parcels
WHERE ST_DWithin(
    geom,
    'POINT()',
    1000
);
“How far did the bus travel last week?”
SELECT  Sum(ST_Length(geom))
FROM    vehicle_paths
WHERE   id = 12
        AND  date > Now() - '7d';
“Find me the nearest truck to the transformer.”
SELECT  t.*  
FROM  trucks t  
ORDER BY  t.geom <-> ( 
  SELECT  geom  
  FROM  transformers  
  WHERE  trans_id = 12  
)  
LIMIT  1;
“What trucks are in the service depots?”
SELECT yards.id, trucks.*
FROM trucks t
JOIN yards y
ON ST_Contains(y.geom, t.geom)
WHERE y.code = 'service';
Spatial Database Basics
CREATE TABLE my_table ( 
    id INTEGER SERIAL PRIMARY KEY, 
    name VARCHAR(28),
    geom GEOMETRY(Point, 26910) 
)

CREATE TABLE my_table ( 
    geog GEOGRAPHY(Point, 4326) 
)
PostGIS Core - SFSQL

POINT

LINESTRING

POLYGON
PostGIS Core - SFSQL

MULTI POINT
MULTI LINESTRING
MULTI POLYGON
PostGIS Core - SFSQL

GEOMETRY COLLECTION
PostGIS Core - ISO

CIRCULAR STRING

COMPOUND CURVE

- CURVEPOLYGON
- MULTICURVE
- MULTISURFACE
PostGIS Core - ISO Extended

POLYHEDRAL SURFACE
PostGIS Core - ISO Extended

TRIANGLE

TIN
PostGIS Core - OSS

GEOMETRY

GEOGRAPHY

$\mathbb{R}_3$

$S_2$
PostGIS Core - Indexing

R-TREE

KD-TREE
GIS without the GIS
“Everything is related to everything else, but near things are more related than distant things.”
Spatial is the foreign primary universal key
Spatial Joins

- `ST_Intersects(geom, geom)`
- `ST_Contains(geom, geom)`
- `ST_DWithin(geom, geom, radius)`
Spatial Joins
SELECT
census.*, customers.*
FROM census
JOIN customers
ON ST_Contains(
census.geom,
customers.geom)
);
find the nearest
Indexed Nearest Neighbor

```
SELECT *  
FROM customers  
ORDER BY geom <-> ST_MakePoint(-124, 42)  
LIMIT 1
```
Indexed Nearest Neighbor
Indexed Nearest Neighbor

2,082,965 GNIS Points
Indexed Nearest Neighbor

SELECT id, name, state, kind
FROM geonames
ORDER BY geom <-> (SELECT geom FROM geonames WHERE id = 4781416)
LIMIT 10

2,082,965 GNIS Points
Spatial Joins

```sql
SELECT c.*, s.store_id
    ST_Distance(c.geom, s.geom) AS dist
FROM
    customers AS c
CROSS JOIN LATERAL
    (SELECT store_id, geom
    FROM stores
    ORDER BY
        customers.geom <-> stores.geom
    LIMIT 1) AS s
```
PostGIS Internal Architecture
CREATE FUNCTION st_area
RETURNS float8
AS 'postgis.so', 'st_area'
LANGUAGE 'c';

PG_FUNCTION_INFO_V1(st_area);
Datum st_area(PG_FUNCTION_ARGS) {}
CREATE FUNCTION test
RETURNS text
AS 'extension.so', 'test'
LANGUAGE 'c';

PG_FUNCTION_INFO_V1(test);
Datum test(PG_FUNCTION_ARGS) {}
PostgreSQL

PostGIS

postgis-3.so

libgeos.so
libxml.so
libprotobuf-c.so

libproj.so
libjson-c.so

libcgal.so
intl.so

libgdal.so
\[
(\theta, \phi) \quad \xrightarrow{\text{Forward}} \quad (x, y)
\]

\[
(\theta_1, \phi_1) \quad \xrightarrow{\text{Pipeline}} \quad (\theta_2, \phi_2)
\]
Coordinate Systems

Plate Carree Projection

Sinusoidal Projection

ST_Transform(geometry, srid)

Behrmann Projection

Albers Equal Area Conic Projection
Proj 6 (proj.org)

- Time dependent datums
- Vertical transformations
- Built in EPSG database
- WKT2, WKT1, URN format support
- Direct datum transformations
  - (no trip through WGS84)
Coordinate Systems

Geometry Engine Open Source

Exactly characterize relationships between any pair of geometries.

aka “dimensionally extended nine-intersection model” (DE9IM)
Spatial QA/QC

- A valid dock has one end on the edge of a lake
- The interior of a dock should be entirely on the interior of the lake

`ST_Relate(dock, lake)`

`'1FF00F212'`
Geometry Engine Open Source

Exactly characterize relationships between any pair of geometries.

aka “dimensionally extended nine-intersection model” (DE9IM)
Geometry Engine Open Source

Generate new geometries for set-wise operations.

- Union
- Difference
- SymDifference
- Intersection
ST_Union(geometry[])
Generate new geometries for fun!

- Buffer
- Delauney / Voronoi
Dilate
and
Erode
Too detailed!
Dilate! ST_Buffer(g, 100)
Erode! ST_Buffer(g, -100)
Generate new geometries for fun!

- Buffer
- Delauney / Voronoi

**ST_VoronoiPolygons()**
**ST_DelaunayTriangles()**
“Divide a polygon into an arbitrary number of similar sub-polygons.”
Geometry Processing

ST_GeneratePoints()
Geometry Processing

ST_ClusterKMeans()
Geometry Processing

ST_Centroid()
Geometry Processing

ST_VoronoiPolygons()
Geometry Processing

\texttt{ST\_Intersection()}

Geometry Processing
Geometry Engine Open Source
Another Library!
3D Support
New Algorithms
CREATE EXTENSION postgis_sfcgal

- ST_3DIntersection
- ST_Tesselate
- ST_3DArea
- ST_3DUnion
- ST_Extrude
- ST_ApproximateMedialAxis
- ST_ForceLHR
- ST_Orientation
- ST_Minkowski
- ST_Volume
- ST_StraightSkeleton
ST_Extrude(geometry, x, y, z)
ST_3DIntersection(geometry, geometry)
Multi-format Raster Processing

ST_Resize
ST_Rescale
ST_Rotate
ST_Transform
ST_AsJPEG
ST_AsPNG
ST_DumpAsPolygons
2D, 3D & 4D

TYPES & INDEXING

BUCK ROGERS
IN THE 3RD DIMENSION
Higher dimensionality

POINT Z (1 2 3)
LINESTRING Z (1 3 4, 1 4 5)
POLYGON Z ((0 0 2, 1 1 2, 1 0 2, 0 0 2))

ST_X('POINT(1 2 3)') = 3
Higher dimensionality input/output

```
ST_AsText()
ST_AsBinary()
ST_AsGML()
ST_AsGeoJSON()
```

```
{"type": "Point",
 "coordinates": [1,1,1]}
```
3D Calculations

ST_3dDistance(geom, geom)
ST_3dLength(geom)
ST_3dClosestPoint(geom, geom)
ST_3dPerimeter(geom)
ST_3dIntersects(geom, geom)
ST_3dDWithin(geom, geom, tolerance)
3D Calculations

\[ \text{ST\_Length} \quad (\text{geom}, \text{geom}) \]  
\[ \text{vs} \]  
\[ \text{ST\_3dLength} \quad (\text{geom}, \text{geom}) \]
Multi-dimensional Indexing

```
CREATE INDEX my_nd_x
ON my_table
USING GIST (  
  geom gist_geometry_ops_nd 
);
```
Multi-dimensional nearest neighbor!

```sql
SELECT * 
FROM starships 
ORDER BY geom <-> 'POINT Z (534 492 -166)' 
LIMIT 1
```
Extra-dimensionality (X, Y, Z, M)

POINT ZM (1 2 3 4)
LINESTRING M (1 2 3, 1 4 3)
LINESTRING Z (1 3 4, 1 4 5)

ST_M('POINT(1 2 3 4)') = 4
"The bridge is at mile 10.5 on Highway 12"
“The salmon habitat is from 3km to 5k above the confluence”

<table>
<thead>
<tr>
<th>rvr</th>
<th>fsh</th>
<th>from</th>
<th>to</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>101</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>rvr</th>
<th>geom</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td></td>
</tr>
</tbody>
</table>
Linear Referencing

LINESTRING M (x₁ y₁ m₁, x₂ y₂ m₂)

ST_LineInterpolatePoint(geom, float)
ST_LineLocatePoint(geom, geom)
ST_LocateAlong(geom, float, float)
ST_LocateBetween(geom, float, float)
ST_LineSubstring(geom, float, float)
Spatio-Temporal Functionality
LINESTRING M (x_1 y_1 t_1, x_2 y_2 t_2)
LINESTRING M \((x_1 \ y_1 \ t_1, x_2 \ y_2 \ t_2)\)

- ST_ClosestPointOfApproach(geom, geom)
- ST_IsValidTrajectory(geom)
- ST_DistanceCPA(geom, geom)
Curve Processing

\texttt{ST\_CurveToLine(\text{geometry})}
Curve Processing

ST_LineToCurve(geometry)
ST_MakeLine({point})
ST_BuildArea(multilinestring)
ST_BuildArea(multilinestring)
Geometry Processing

- `ST_GeometryN(geometry, integer)`
- `ST_PointN(geometry, integer)`
- `ST_ExteriorRing(geometry)`
- `ST_InteriorRingN(geometry, integer)`

- `ST_Dump(geometricalcollection) => setof(geometry)`
- `ST_DumpRings(polygon) => setof(geometry)`
- `ST_DumpPoints(geometry) => setof(geometry)`
Geography
geography

geometry

$S_2$

$R_2$
“I want to find all the address points within one mile! My data is in lat/lon! Google Maps rocks!”
“GIS sucks, and I want to go home now.”
“Yeah, I own a freaking satellite, you got a problem with that?”
double R = 6371000; /* meters */
double d_lat = lat2-lat1; /* radians */
double d_lon = lon2-lon1; /* radians */
double sin_lat = sin(d_lat/2);
double sin_lon = sin(d_lon/2);
double a = sin_lat * sin_lat +
    cos(lat1) * cos(lat2) *
    sin_lon * sin_lon;
double c = 2 * atan2(sqrt(a),
    sqrt(1-a));
double d = R * c;

double dx = x2 - x1;
double dy = y2 - y1;
double d2 = dx * dx +
    dy * dy;
double d = sqrt(d2);
Geography supports...

- Indexes on spherical data
- Nearest neighbor searches
- ST_Intersects()
- ST_Distance()
- ST_DWithin()
- ST_Area()
- Casts to/from GEOMETRY
Raster Analysis

ST_Envelope(raster) →

ST_ConvexHull(raster)

ST_Polygon(raster)
Raster Analysis
Raster Analysis
Raster Analysis 😄
Raster Analysis 😊
Raster Analysis 😄
“What buildings will flood at 30 metre sea level rise?”
SELECT b.*
FROM buildings b
JOIN dem
ON ST_Intersects(b.geom, dem.rast)
WHERE ST_Value(dem.rast, ST_Centroid(b.geom)) < 30;
Topology Model

- For coverages & shared boundaries
- Parcels, admin boundaries
- Good tooling required
  eg QGIS PostGIS Topology Editor
Performance
PostGIS is **faster** than...

PostGIS is **slower** than...
PostGIS is **better** than...

PostGIS is **worse** than...
Performance Tweaks
Smallification

Polygon with 90,000 vertices

$\text{ST\_Subdivide}()$

Recursive subdivision
CREATE TABLE land_subdivided AS
SELECT ST_SubDivide(geom) AS geom,
       base250_id
FROM land

Polygons with no more than 250 vertices
Simplification
Simplification

ST_Simplify()

Douglas-Pueker Algorithm
Simplification

ST_SimplifyVW()
Visvalingam-Whyatt Algorithm
Sorting

random

order by ST_X()
Sorting

ORDER BY geom (v2.4)

ORDER BY geom (v3.0)
clustering!
(not nodes!)
ST_GeneratePoints(geom, n)
ST_GeneratePoints(geom, popn/100)
ST_ClusterDBScan(
  geom,
  1000,  -- proximity
  100   -- minsize
)
ST_ClusterKMeans(
  geom,
  5  -- K
)
GIS without the GIS
PostGIS Web Architectures
“Where’s the middleware?!"
Input / Output Formats

`ST_AsText(geometry)`

`ST_AsBinary(geometry)`

`ST_AsGML(geometry)`

`ST_AsSVG(geometry)`

`ST_AsGeoJSON(record)`

`ST_AsMVT(record)`

OGC Standards

Web Standards

Leaflet

OpenLayers

MapBox
Super Lightweight Architecture

Web Maps → MVT
Web Queries → GeoJSON

https://info.crunchydata.com/blog/dynamic-vector-tiles-from-postgis
GIS without the GIS
PostGIS
PostgreSQL
Ecosystem
- **Network traversals**
  - Djikstra
  - A*, C*, Shooting*
  - Travelling salesman
  - Trade areas
- **Build any network** you like
  - Utility network?
  - Edge weights, dynamic edge removal, etc
- **Forward Geocoder**
  - Input is an address **string**
  - Output is a address **location** point

- **Uses US Census data**
  - Limited to US states and territories

- **Included in PostGIS**
  - `create extension postgis_tiger_geocoder`
  - `create extension address_standardizer`
Uber H3

https://github.com/dlr-eoc/pgh3
ogr_fdw

PostgreSQL

https://github.com/pramsey/pgsql-ogr-fdw

+ dozens more
- **PostGIS Workshop**
  - https://postgis.net/workshops/postgis-intro/
- **QGIS Tutorial**
  - https://docs.qgis.org/3.4/en/docs/training_manual/
- **Web Middleware Options**
  - https://docs.geoserver.org/
  - https://mapserver.org/documentation.html
- **Web Client Options**
  - https://leafletjs.com/
  - https://openlayers.org/
  - https://docs.mapbox.com/mapbox-gl-js/api/
Everything About PostGIS


paul.ramsey@crunchydata.ca