

PostGIS: A Standards Based Geographic Extension for PostgreSQL

Frank Warmerdam
President, OSGeo



Overview

- Brief background
- PostGIS Details
- PostGIS Examples
- Survey of Simple Features 1.1 Geometries
- Simple Features 1.2 Geometry
- A survey of simple features based software
- Commentary

What is a Spatial Database?

- Support for a “Geometry” Type
- Indexing for the Geometry Type
- Functions for the Geometry Type
- Database that can answer GIS questions:
quickly
on large volumes of data

Why a Spatial Database?

(instead of just using files)

- Transactional Integrity
 - Multiple Users, Multiple Edits
- Unified Storage, Management, Access
SQL Everywhere
- Because Databases are Better than Files
NOT!

Unified

Storage

**Database
Guys**

**Corporate
Database**

GIS
Guys

**GIS
Files**

PostGIS

- Geographic Extension for PostgreSQL
- Based on OGC Simple Features for SQL
- By Refractions Research (Victoria, BC)
- First release in 2001
- GPL licensed (likely why not in main src tree)
- R-Tree- over- GiST used for spatial index
- Introduces:
 - new geometry types
 - many new functions
 - new support tables

<http://www.postgis.org>

Some Geometry Functions

- Area(POLYGON)
- Distance(GEOMETRY,GEOMETRY)
- Contains(GEOMETRY,GEOMETRY)
- Intersection(GEOMETRY,GEOMETRY)
- Intersects(GEOMETRY,GEOMETRY)
- Union(GEOMETRY,GEOMETRY)
- Buffer(GEOMETRY,double)
- ConvexHull(GEOMETRY)
- Perimeter(GEOMETRY)
- Crosses(GEOMETRY,GEOMETRY)
- Transform(GEOMETRY,integerSRID)

Some Accessor Functions

- Dimension(GEOMETRY)
- AsText(GEOMETRY)
- ST_X(POINT)
- ST_Y(POINT)
- NumPoints(GEOMETRY)
- PointN(GEOMETRY,integer)
- NumGeometries(GEOMETRY)
- GeometryN(GEOMETRY,integer)
- GeometryType(GEOMETRY)

GIS Questions

“How many people live within 5 miles of the toxic gas leak?”

```
SELECT sum(population)
FROM census_tracks
WHERE
    distance(census_geom,
              'POINT(210030 3731201)')
    < (5 * 1609.344)
```

GIS Questions

“What is the area of municipal parks inside the Westside neighbourhood?”

```
SELECT sum(area(park_geom))  
FROM parks, nhoods  
WHERE  
    contains(nd_geom,park_geom)  
    AND nhood_name = 'Westside'
```

GIS Questions

“What is the maximum distance a student has to travel to school?”

```
SELECT  
  max(distance(student_location,  
               school_location))  
FROM students, schools  
WHERE students.school_id = schools.id;
```

Create a Table Simply

```
CREATE TABLE ROADS  
(ID int4,  
  NAME varchar(255),  
  GEOM geometry)
```

Create a Table Properly

```
CREATE TABLE ROADS  
  (ID int4,  
   NAME varchar(255))
```

```
SELECT AddGeometryColumn  
  ('roads','geom',423,'LINESTRING',2)
```

'roads': Table name

'geom': Geometry column name

423: SRID (coordinate system)

'LINESTRING': geometry type constraint

2: Dimension

Insert Data

```
INSERT INTO roads  
( road_id, road_geom, road_name)  
VALUES  
(1,  
  GeomFromText(  
    'LINESTRING(19123 24311,19110 23242)',  
    242),  
  'Jeff Rd.')
```

Spatial Index

```
CREATE INDEX roads_geom_index  
ON roads  
USING GIST(geom)
```

geometry_columns

```
CREATE TABLE geometry_columns (  
  f_table_catalog  VARRCHAR(256) NOT NULL,  
  f_table_schema   VARCHAR(256) NOT NULL,  
  f_table_name     VARCHAR(256) NOT NULL,  
  f_geometry_column VARCHAR(256) NOT NULL,  
  coord_dimension  INTEGER NOT NULL,  
  srid             INTEGER NOT NULL,  
  type             VARCHAR(30) NOT NULL  
)
```

part of OGC specification
important to spatial applications
from AddGeometryColumn()

spatial_ref_sys

```
CREATE TABLE spatial_ref_sys (  
  srid          INTEGER NOT NULL PRIMARY KEY,  
  auth_name     VARCHAR(256),  
  auth_srid     INTEGER,  
  srtext        VARCHAR(2048),  
  proj4text     VARCHAR(2048)  
)
```

Defines Coordinate System

Part of OGC specification

Important to spatial applications

List is prepopulated

PostGIS Application Support

Web Mapping:

- MapServer, MapGuide, GeoServer

Desktop GIS :

- Udig, QGIS, JUMP, GRASS

Proprietary GIS:

- Cadcorp SIS, ArcGIS 9.3(?)

ETL:

- FME, GDAL/ OGR

PostGIS Installation

- Included in standard PostgreSQL Win Installer (buried back in the extra packages)
- PostgreSQL+ PostGIS MacOS X binaries available
- Elsewhere installing from source pretty easy
Optionally depends on GEOS and PROJ.4
Two SQL scripts need to be run to setup postgis types, and setup support tables

OGC and Simple Features

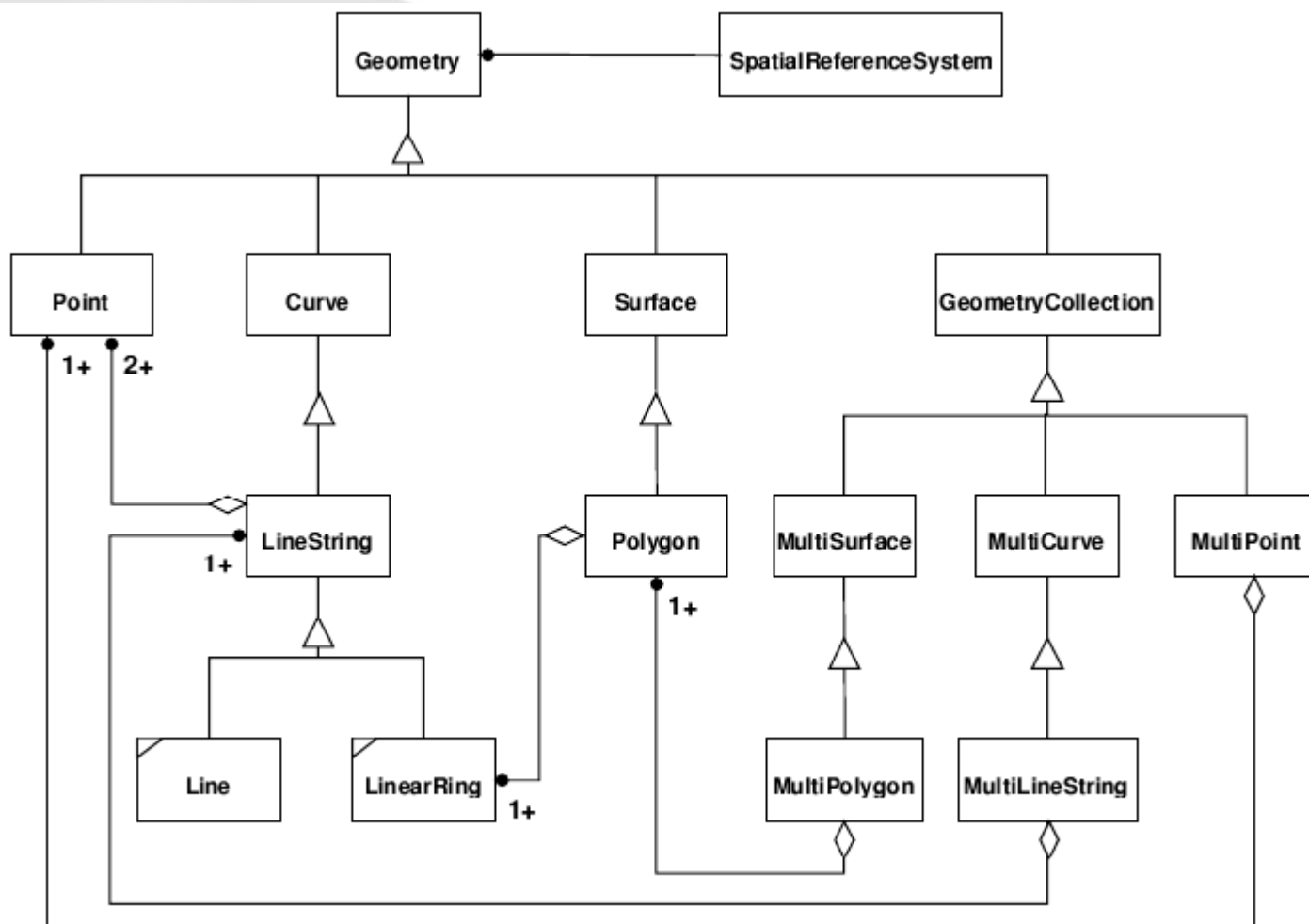
OGC is the “Open Geospatial Consortium”

- Collaborative development of specifications for geospatial services
- Industry driven
- About Open Standards, not Open Source

Simple Features

- Abstract geometry model
- Base of “Simple Features for SQL”
- First concrete OGC spec (mid 90's)

Simple Features Geometries (1.0)



Point

- 2D (x,y) point location

WKT (Well Known Text) Representation:

`POINT(- 117.25 35.0)`

Line String

- Chain of point location
- No restrictions on self-intersection
- Duplicate points ok

`LINESTRING(0 10, 20 15, 30 15)`

Line String

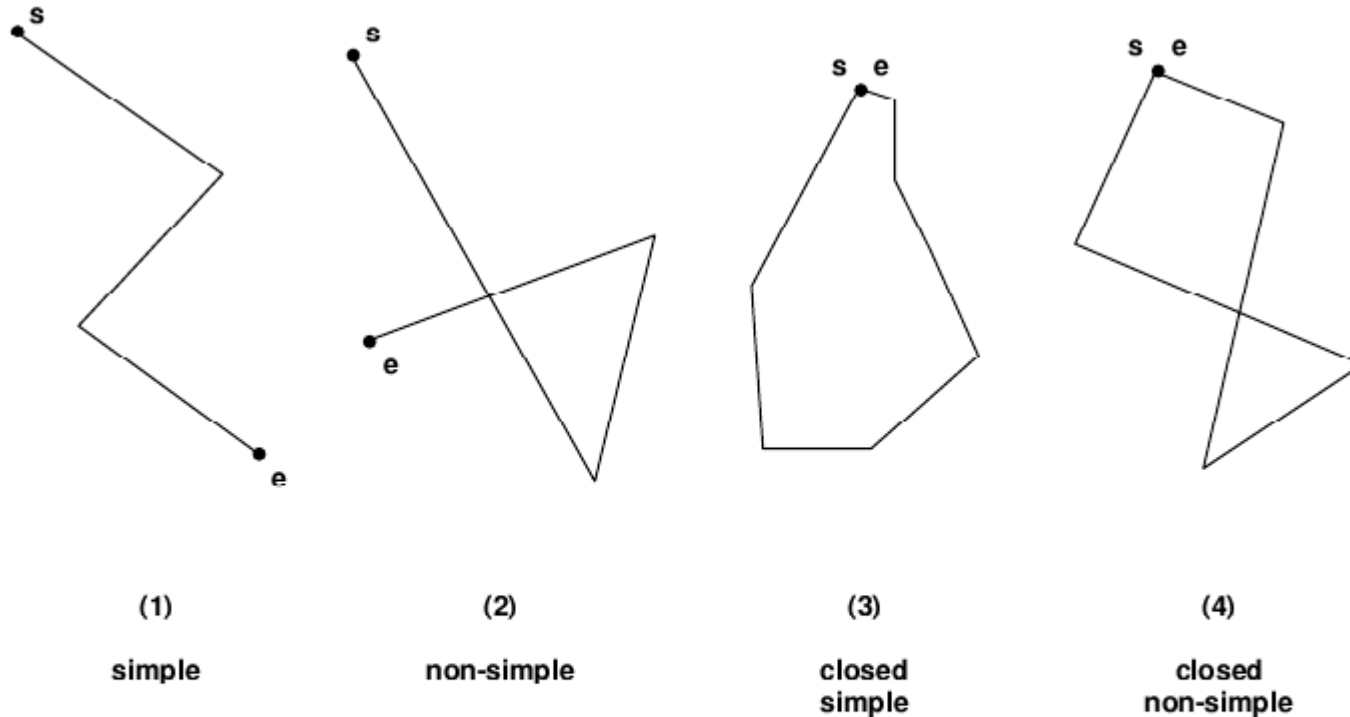


Figure 2.2—(1) a simple LineString, (2) a non-simple LineString, (3) a simple, closed LineString (a LinearRing), (4) a non-simple closed LineString

Polygon

- Polygon with one outer ring, and zero or more inner rings (holes)
- Polygons are closed (last point of ring equals first point)
- Rings may not cross
- Rings may touch
- Polygon interior is a connected point set
- Winding direction of rings not significant

```
POLYGON((0 0,10 10,10 0,0 0),  
         (3 1,4 1,4 2,3 1))
```

Polygon

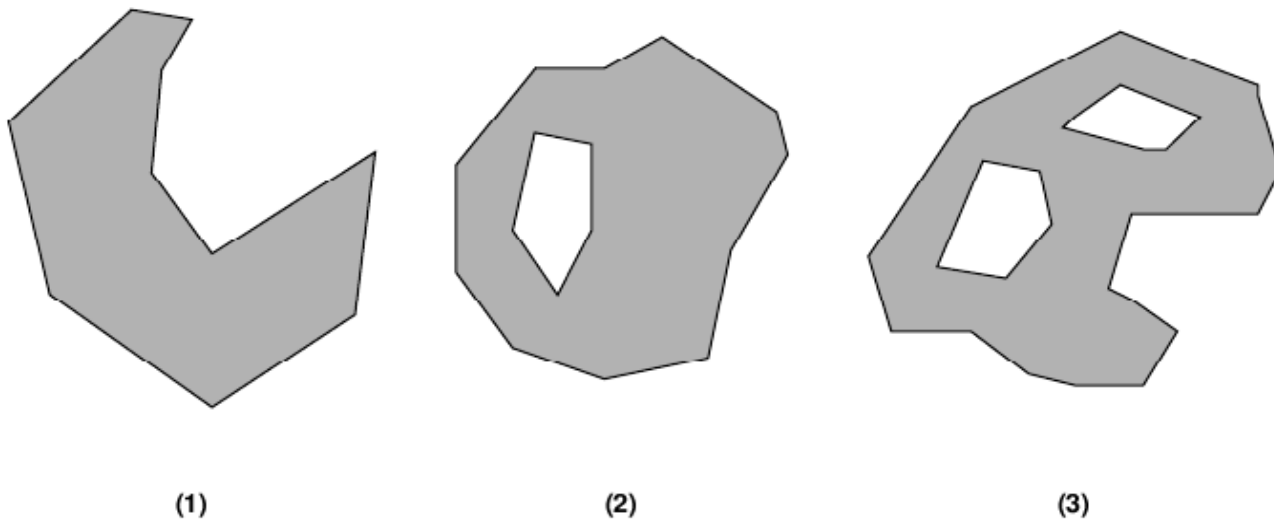


Figure 2.4—Examples of Polygons with 1, 2 and 3 rings respectively.

Polygon

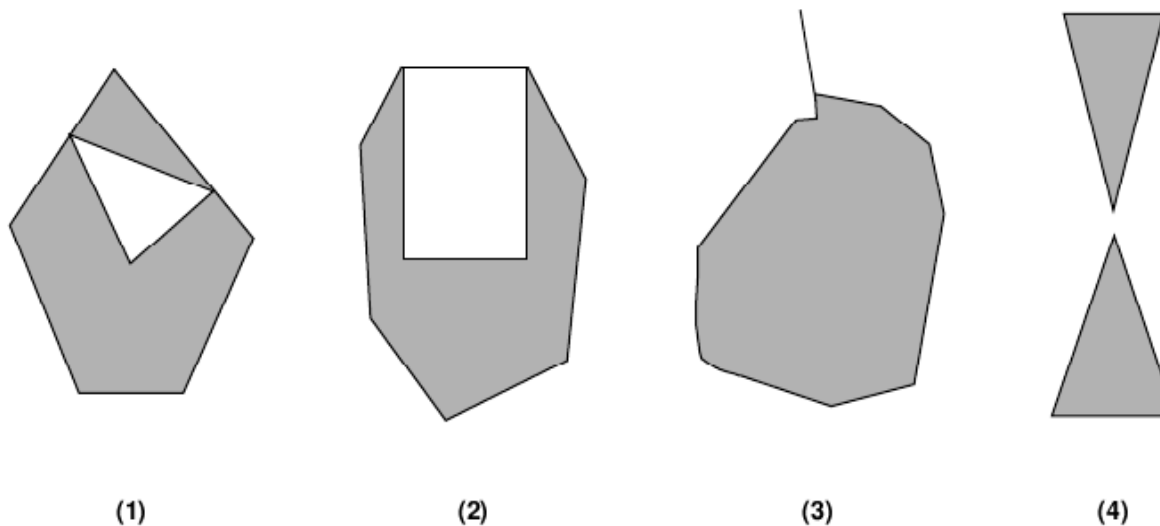


Figure 2.5—Examples of objects not representable as a single instance of Polygon. (1) and (4) can be

Multi-Polygon

- A collection of polygons
- May be nested (and island in a lake)
- May **not** be overlapping
- May touch at a point
- May **not** touch along an edge

```
MULTIPOLYGON(((0 0,10 10,10 0,0 0),  
                (3 1,4 1,4 2, 3 1)),  
              ((20 20, 30 30, 30 20, 20 20)))
```

Multi-Polygon

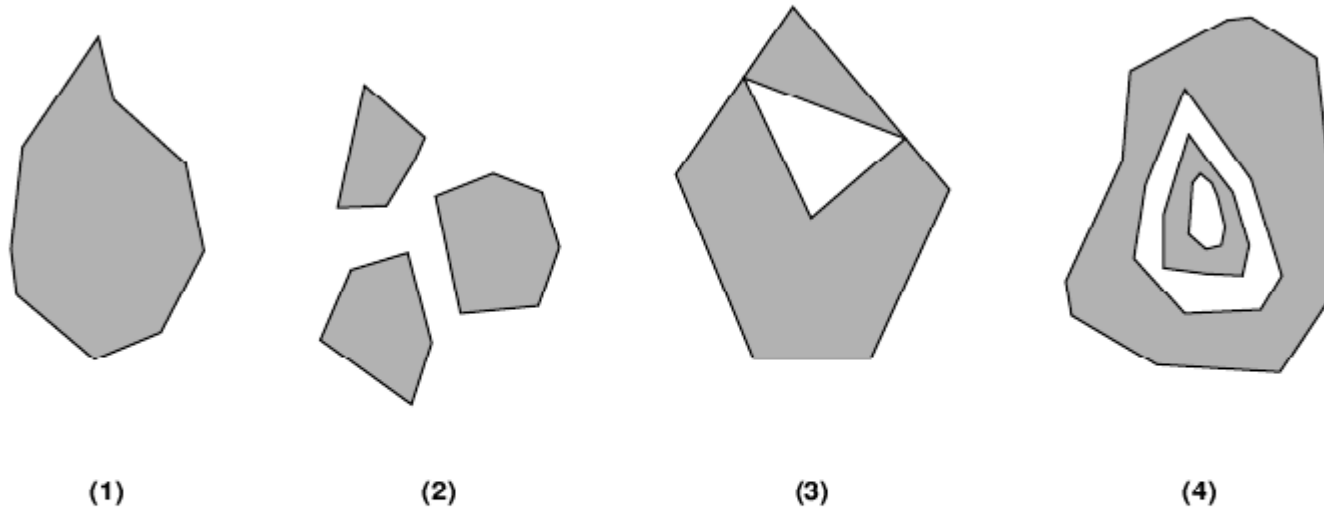
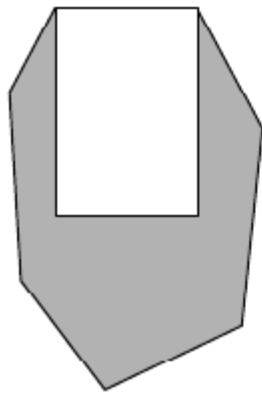
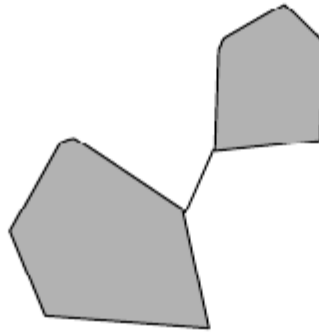


Figure 2.6—Examples of MultiPolygons

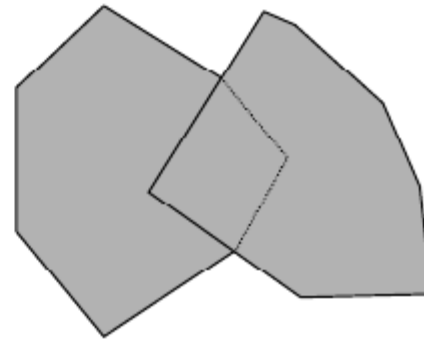
Multi-Polygon



(1)



(2)



(3)

Figure 2.7—Geometric objects not representable as a single instance of a MultiPolygon.

Multi Line String

- A collection of linestrings

```
MULTILINESTRING((0 0,10 10,10 0),  
                (3 1,4 1,4 2,5 1))
```

Multi Point

- A collection of points

```
MULTIPOINT((0 0),(10 10),(10 0))
```

Geometry Collection

- A collection of geometries

```
GEOMETRYCOLLECTION(  
  POINT(0 5),  
  LINESTRING(3 5, 2 9, 1 3),  
  POLYGON((0 0, 10 10, 10 0, 0 0)))
```

```

classDiagram
    class Point
    class Curve
    class LineString
    class Line
    class Polygon

    Point "1+" -- "2+" LineString
    Curve <|-- LineString
    LineString <|-- Line
    LineString <|-- Polygon
  
```

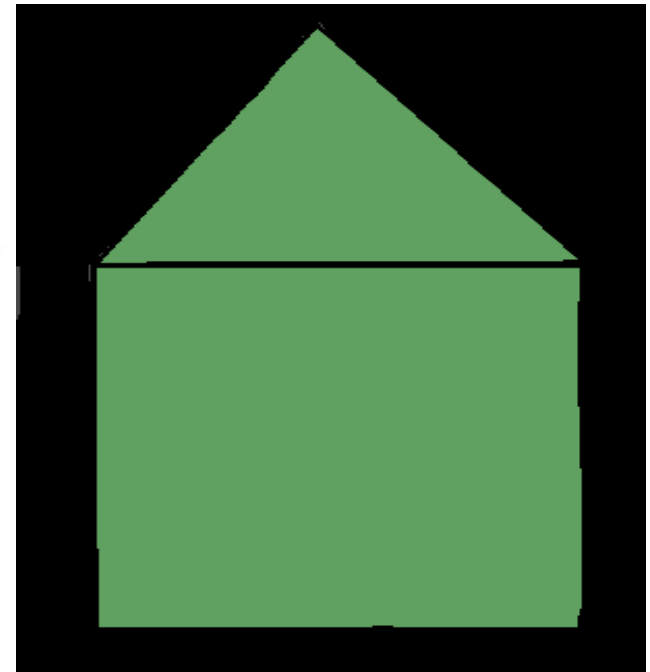
Simple Features 1.2

- Extends vertices to 3D/ 4D (Z/ Measure)
- Geometric operations done in 2D
- Adds Polyhedral Surface
- Adds TIN
- Alters defacto 3D/ 4D WKT/ WKB formats
- Adds Annotation Text to feature model

Polyhedral Surface

- A surface consisting of adjacent polygons
- Stored as collection of polygons
- TIN is special case, all triangles

```
POLYHEDRALSURFACE(  
  ((0 10,0 0,10 0,10 10,0 10)),  
  ((0 10,5 15, 10 10,0 10)))
```



SQL-MM

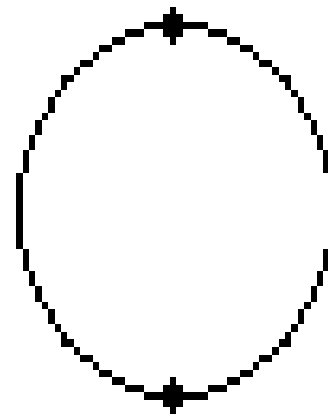
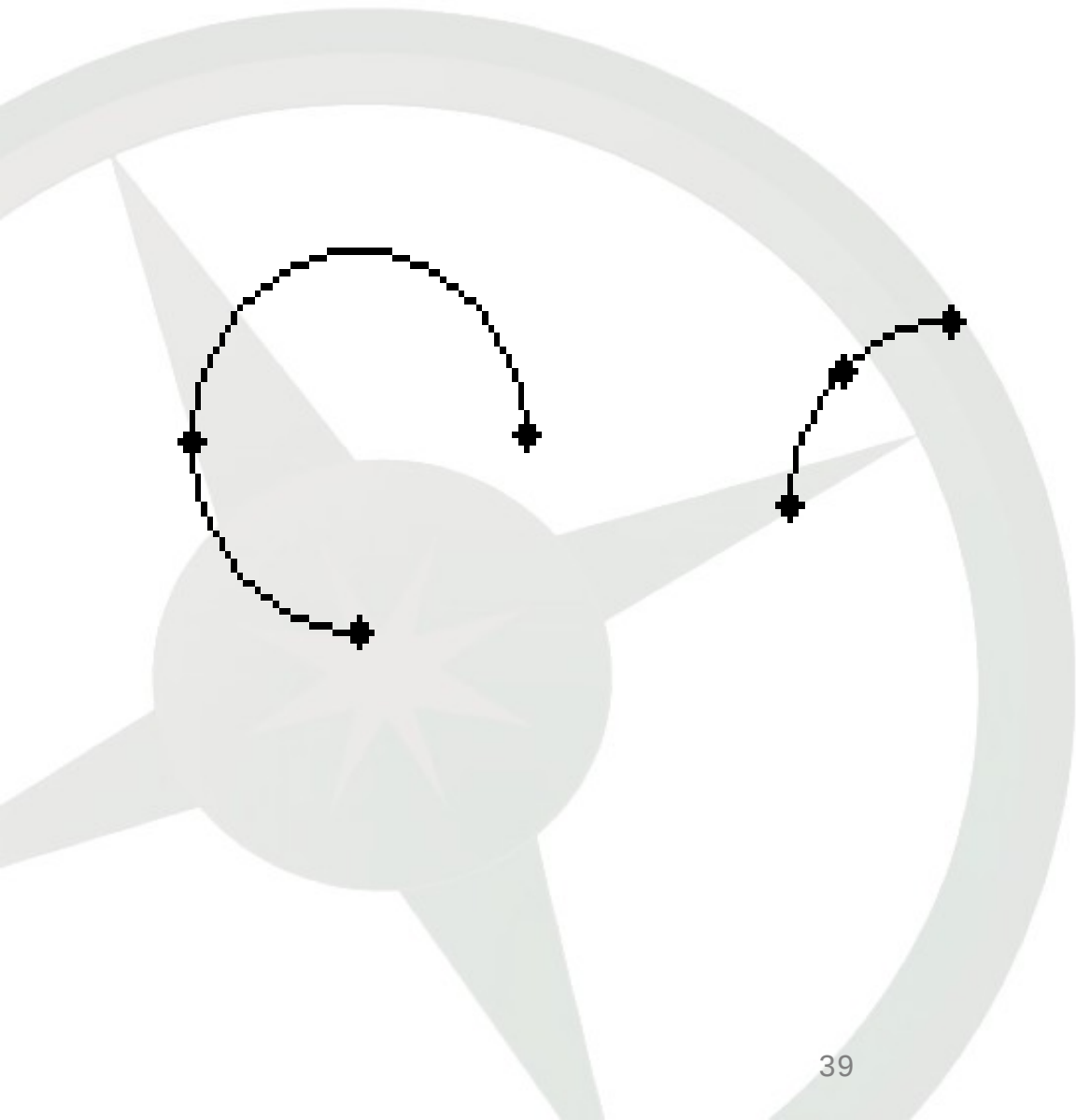
- ISO SQL Geometry Specification
- SF 1.2 aligned with SQL- MM
- PostGIS supports these SQL- MM types:
 - CircularString (arcs of a circle)
 - CompoundCurve (arcs+ linestrings)
 - CompoundSurface (curved polygons)
 - MultiCurve
 - MultiSurface
- SQL- MM also addresses topology, networks, directions, angles, ...

SQL-MM: CircularString

- A string of partial circle arcs connected end to end (a LineString of arcs)
- Each arc defined by three points
 - arc start
 - a point on the arc
 - arc end
- Whole circles have same first/ last and the middle point is opposite side of circle

CIRCULARSTRING(0 0, 1 1, 1 0)

SQL-MM: CircularString



Software Survey

OGC Simple Features 1.0 for SQL

- Postgres/ PostGIS: Full implementation including all geometry functions
- MySQL: Supports geometry, and spatial indexing, very limited additional functions
- Oracle: Apparently compliant, many extensions
- MSSQL: Supports geometry, spatial indexing, very limited additional functions
- SpatialLite: Partial SFSQL.
- Ingres: Being implemented! 1.2?

Software Survey

Simple Features based/ inspired:

- OGR: Geometry is Simple Features 1.0
- FDO: Geometry is extension of SF
- QGIS: Geometry is Simple Features 1.0
- SDE: Close to SF + extensions
- GEOS: SF 1.0 geometry model

Not Simple Features:

- MapServer, OpenEV, GRASS
- MapInfo, Microstation

Format Survey

- GML: Geometry is SF (+ extensions in 3?)
- Shapefile: Not simple features
- Mapinfo: Not simple features

Most major GIS products do not exactly map to Simple Features, though they may be similar.

SF: What is missing?

- Nonlinear curves (ellipse/ spline/ etc)
- 3D solids
- Topology
- Non- planar surfaces
- Representation

Universal Geometry Model for GIS?

No, because:

- lack of real curves, hampers CAD links
- lack of topology

Yes, because:

- Understandable
- Lingua franca for interchange/ discussion
- Wide adoption

Takeaway Lessons

- PostgreSQL+ PostGIS is the leading spatial database combination
- PostGIS is standards based
- OGC Simple Features is useful, widely adopted way of expressing geometry

Opportunities

- Dracones talk here at 1:30!
- OSBootCamp 6 (Geobootcamp)
June 2nd, Ottawa (Carleton)
Free!
www.osbootcamp.org
- OSGeo Ottawa
Monthly meetings at a pub
wiki.osgeo.org/wiki/Ottawa_Chapter

Questions?

PostGIS:
www.postgis.org

OGC (standards):
opengeospatial.org



OSGeo.org
The Open Source
Geospatial Foundation