Cross Comparison of Spatially Enabled Databases: PostGIS, SQL Server and JASPA

Leo Hsu and Regina Obe

http://www.postgis.us
http://www.bostongis.com
http://www.postgresonline.com
http://www.paragon corporation.com
Cross Comparison
Key Focus / Audience

**PostGIS**
Low Price, High Performance
GIS and Research Professionals
Companies with high GIS processing need and unpredictable scale-out
Programmers / Database Specialists (no GIS skills)
Affordable to all

**SQL Server 2008 (Spatial)**
Existing users of SQL Server
Adequate GIS for most use-cases
Tight integration with Microsoft Stack (e.g. Sharepoint, Azure, Reporting Services)
Programmers / Database Specialists (no GIS skills)
Affordable price (if you don’t have too many servers)

**JAvaSPAtial (JASPA)**
Easy to change and lots of processing features (similar to PostGIS)
Adequate performance for most use-cases
Education, GIS Professionals with Java programming skills
Affordable to all
Demonstration of the power of PL/Java (particularly in PostgreSQL)
Cross Comparison
Main Platform Differentiation

PostGIS [http://www.postgis.org/documentation](http://www.postgis.org/documentation)
Spatially enables PostgreSQL
C,C++ (GEOS), PL/PgSQL, Proj4, libxml (PostGIS 2.0 has extra dependency on libgdal)
Each version works with 2-3 versions of PostgreSQL
Packaged separately from PostgreSQL
Generally available as precompiled add-on package for most OS
Operating System Agnostic

SQL Server 2008 (Spatial)
Spatially enables SQL Server 2008 and packaged in
.NET / C++
Only works with version designed for
Only works on Windows 2003 or above

JavaSPAtial (JASPA) [http://jaspa.forge.osor.eu/](http://jaspa.forge.osor.eu/)
The best user documentation I have ever seen ☺ (Creative Commons Share-Alike License)
Spatially enables PostgreSQL, H2, perhaps Oracle one day?
Java (JTS), PL/Java, PL/PgSQL, GeoTools
Currently works for PostgreSQL 8.3, 8.4, 9.0
(windows / Linus binaries for 8.3 and 8.4)
Patterned after PostGIS – almost identical function names
Operating System Agnostic
Can coexist with PostGIS on same PostgreSQL, and even same database. All
functions etc. are installed in schema called jaspa.
Cross Comparison
Function Differentiations (Current)

**PostGIS 1.5**
OGC Geometry / Geography (Geodetic)
Basic curve support – needs improvement
Basic 3D – no spatial relationship functions
Lots of output options WKT, WKB, (KML,GML (2,3)), GeoJSON, SVG
Input Options – WKT, WKB, GML (2,3), KML
Reprojection support via Proj4

**SQL Server 2008**
OGC Geometry / Geography (Geodetic) – better Geodetic than PostGIS
Great tie in with Microsoft Stack – e.g Reporting Services
Input/ Output functions – WKT, WKB, GML but only 3.
No reprojection support
No aggregate functions (need to use the SQL Spatial codeplex add-on project)

**JASPA 0.1**
OGC Geometry (no Geography (Geodetic))
Same functionality as PostGIS 1.5 minus Geography
and some other native PostGIS
Has input/output functions same as PostGIS via GeoTools
Additional functions above PostGIS 1.5 – e.g. ST_DelaunayTriangles, ST_Snap, ST_NodeLine, more robust Union support than PostGIS
(JTS is more robust than GEOS e.g less topological precision issues)
Reprojection support via GeoTools
Cross Comparison
Third Party Support

**PostGIS 1.5**
The best in OS arena
OpenStreetMap loader, GDAL, packaged shp2pgsql
MapServer, GeoServer
QGIS (even preliminary support for raster), OpenJump, GvSig (even PostGIS 3D support and PostGIS raster support for next release) and a lot more
Safe FME, CadCorp, AutoCad (beta), ArcGIS (via ZigGIS) (only 1.3 and 1.4 support via ArcSDE), Manifold, MapInfo

**SQL Server 2008**
ESRI ArcGIS SDE
Safe FME, Manifold, MapInfo, Autocad
MapServer, GeoServer
GDAL
Viewer built into SQL Manager

**JASPA 0.1**
Packaged loader (shp2jaspapg)
MapServer
OpenJump – if you use the ad hoc query tool and choose PostGIS driver – OpenJump can’t tell the difference
Cross Comparison Syntax (Create a Geometry point)

**PostGIS 1.5**
SELECT ST_GeomFromText('POINT(82.572258 35.708483)', 4326);
SELECT ST_SetSRID(ST_Point(82.572258, 35.708483),4326);

**SQL Server 2008**
SELECT Geometry::STGeomFromText('POINT(82.572258 35.708483)', 4326);

**JASPA 0.1**
(NOTE: the jaspa. is not necessary if you add to your db search_path and don’t have PostGIS installed in same db)
SELECT jaspa.ST_GeomFromText('POINT(82.572258 35.708483)', 4326);
SELECT jaspa.ST_SetSRID(jaspa.ST_Point(82.572258, 35.708483),4326);
Cross Comparison Syntax (Create a Geography point)

AKA Geodetic

**PostGIS 1.5**
SELECT ST_GeogFromText('POINT(82.572258 35.708483)', 4326);

**SQL Server 2008**
SELECT Geography::STGeomFromText('POINT(82.572258 35.708483)', 4326);

**JASPA 0.1**
Geography what?
WHY would we cross-breed PostGIS and JASPA?

- PostGIS has faster union support, but JASPA (built on JTS) bugs fixed in JTS may appear faster in JASPA than PostGIS.
- PostGIS has geography, more in-built functions – ST_ClosestPoint, upcoming raster and 3D, but JTS functions appear in JASPA before they get ported to GEOS and PostGIS – e.g. ST_DelaunayTriangles etc. – gives us a sneak preview of what may come in PostGIS.
- A little easier to prototype in Java than C (though plpgsql is still much easier)
- Can cross compare GEOS / JTS issues
HOW?

Install JASPA and PostGIS in the same database. We have PostGIS installed in public and JASPA in jaspa.

-- Our PostGIS table (1000 points) –
CREATE TABLE postgis_test(gid serial PRIMARY key, geom public.geometry);
INSERT INTO postgis_test(geom)
SELECT ST_Point(x*random(),y*random())
FROM generate_series(1,200, 5) As x CROSS JOIN generate_series(1,100, 4) As y;
PASPAGIS
Before
PASPAGIS
Delaunay Triangles Test

-- Creates 1972 triangles and pretty fast (under 1.5 secs on standard Windows 7 box) --
CREATE TABLE postgis_deltriang(gid serial PRIMARY key, geom public.geometry);

INSERT INTO postgis_deltriang(geom)
SELECT
    (ST_Dump(jaspa.ST_DelaunayTriangles(
        jaspa.ST_GeomFromEWKB(
            ST_AsEWKB(
                ST_Collect(geom)
            )
        )::public.geometry)).geom
FROM postgis_test;
PASPAGIS
Delaunay Triangles Test

-- Creates 1972 triangles (Shorter syntax but relying on autocasting) --
-- Works because JASPA’s native form is bytea (presumably EWKB) -- it has
  no type called geometry
-- PostGIS can autocast to bytea that Jaspa surprisingly understands
CREATE TABLE postgis_deltriang(gid serial PRIMARY key, geom
  public.geometry);

INSERT INTO postgis_deltriang(geom)
SELECT (ST_Dump(
    jaspa.ST_DelaunayTriangles( ST_Collect(geom) )::geometry )
  ).geom
FROM postgis_test;
PASPAGIS
After

Triangles

Triangles overlaid with original points
Cross Comparison
Function Improvements
PostGIS 2.0 (Alpha)

PostGIS 2.0 (Experimental Builds available for Windows 32-bit)
Raster (main focus raster/geometry operations and raster analysis)
Better 3D (TINS, Polyhedral Surfaces)
3D relationship functions (ST_3DIntersects, ST_3DClosestPoint,
ST_3DShortestLine, ST_3DDistance, ST_3DDWithin ..and more (for
everything but TINS)
GML input/output for 3D (all 3D geometries)
Lots more functions – ST_Split, ST_Snap, ST_ValidDetail,
ST_ConcaveHull, ST_MakeValid and more ->
http://www.postgis.org/documentation/manual-
svn/ch12.html#NewFunctions_2_0
Improved Topology support following SQL/MM ISO with additional goodies
like AsGML
3D spatial index (maybe) (but definitely by 2.1)
KNN GIST (if using PostgreSQL 9.1+) (maybe) (but definitely by 2.1)
Lots of functions rely on GEOS 3.3 (And a LOT OF WORK is being done in
GEOS 3.3 to make Union etc. more robust).
Cross Comparison
Function Improvements
SQL Server Denali 2010 (Alpha)

Faster indexing and index wizard support than 2008
Nearest Neighbor Query Plan (similar idea to coming KNN GIST in PostGIS)
Faster relationship checks for Points
Better CLR support
Curved support for both Geometry and Geography (sounds impressive)
Improved Geography (single hemisphere limit lifted)
New functions STShortestLineTo (both for geometry and geography) (similar to PostGIS 1.5 ST_ShortestLine (for geometry))
Aggregate functions (e.g. UnionAggregate, EnvelopeAggregate (similar to PostGIS ST_Extent for geometry), CollectionAggregate, ConvexHullAggregate - etc even for Geography)
Persist Computed Column that returns spatial type (e.g. can create a computed column that returns a point like STCentroid, similar to PostgreSQL functional indexes, but it’s persisted so no need for recalc if output)
Cross Comparison
Function Improvements JASPA 0.2
Roadmap

JASPA 0.2
Geography
Support for HSQLDB
PostGIS in Action

Buy PostGIS in Action:
http://www.postgis.us/page_buy_book