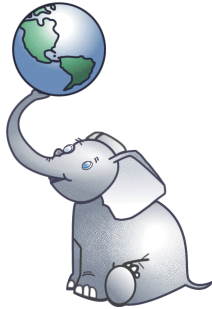
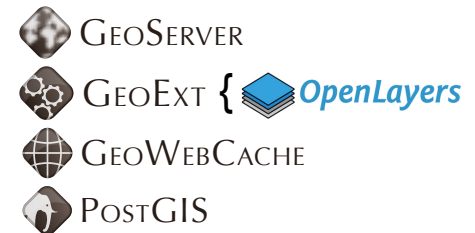


# The State of PostGIS



pramsey  
@  
 OPENGEO  
.org



Whole product  
Adding value w/ bundling, ease-of-use,  
integration testing,  
support for all components

## Spatial Database?

PostGIS is a spatial database,  
but what is a spatial database?

## Database

- **Types**
  - string, float, date
- **Indexes**
  - b-tree, hash
- **Functions**
  - strlen(string), pow(float, float), now()

database provides  
random access data storage system  
types, indexes, functions for STANDARD TYPES

## SPATIAL Database

- **Spatial Types**

- geometry, geography

- **Spatial Indexes**

- r-tree, quad-tree, kd-tree

- **Spatial Functions**

- ST\_Length(geometry), ST\_X(geometry)

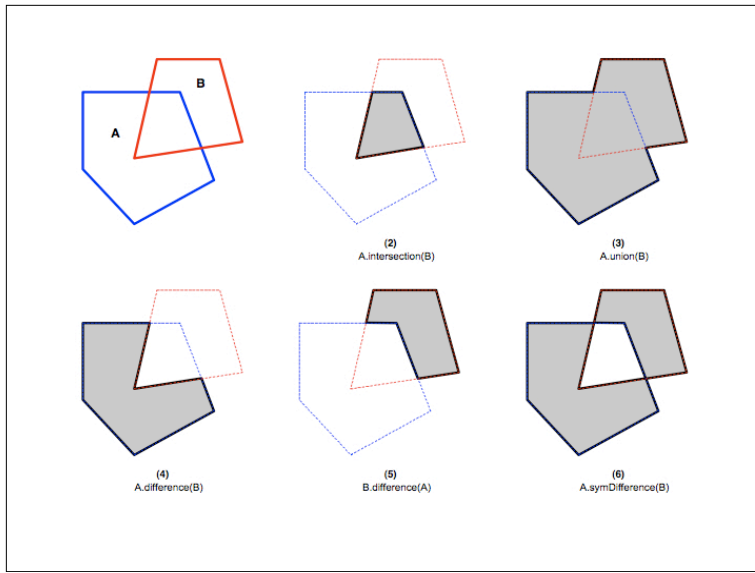
spatial database provides  
random access data storage system  
SPATIAL types, SPATIAL indexes, SPATIAL  
functions

Open Geospatial  
Consortium (OGC)



Simple Features for SQL  
(SFSQL)

spatial databases follow international standard  
SFSQL = Guideline for function names/behaviors  
PostGIS is certified compliant, has been through  
testing



SFSQL standard defines relationships between geometries like intersection, union, difference, symmetric difference



spatial databases your likely to run into are PostGIS, Oracle and SQL Server. Others include DB2 Spatial, Informix Spatial, Netezza, Teradata, SpatialLite. Spatial functionality has gone from being a rarity to being a standard feature.





Is MySQL a spatial database?

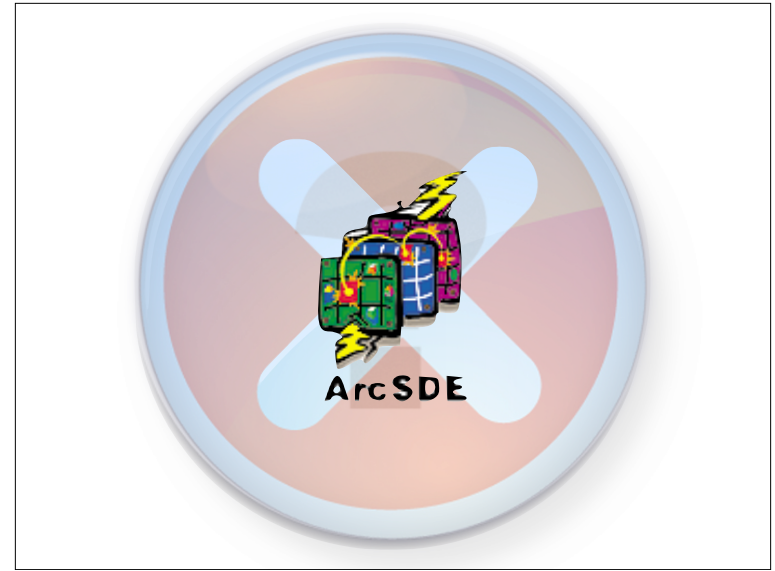
Not yet. (Still)

There are spatial types and a spatial index.

The spatial types are defined in MyISAM tables only, no transactions

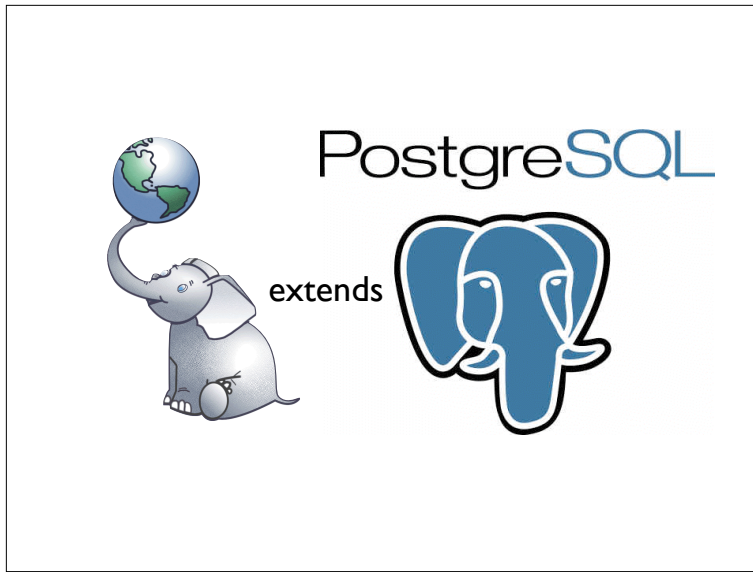
The functions only work against bounding boxes.

There are patches to add wider support for true geometry tests ([http://forge.mysql.com/wiki/GIS\\_Functions](http://forge.mysql.com/wiki/GIS_Functions)) but they have not been added to either the general releases (v5.1) or to the development releases (v5.5) thus far.




Is SDE a spatial database?

No, it is spatial “middleware”. It sits on top of a database and interprets requests. Depending on the database it sits on top of, it might pass queries back to a native spatial database (Oracle, PostGIS, ST\_Geometry) but SDE itself is not a spatial database.



PostGIS is great spatial database,  
and it is build on top of a great standard  
database,  
PostgreSQL.  
The core database provides transaction  
management,  
disk storage routines, SQL processing and  
planning.  
PostGIS provides spatial types, functions and  
indexes



PostgreSQL

- **Open Source (BSD)**
- **“Enterprise” Database**
  - ACID, hot backup, replication, partitioning
  - triggers, constraints, foreign keys, user functions
  - PL/PGSQL, PL/Perl, PL/TCL, PL/Java, PL/R
- **Corporate support**
  - Enterprise DB
  - SRA

By building on PostgreSQL, we get these good things!

What does  
PostGIS do?

That is interesting, but... what does it do?  
PostGIS answers questions that have a spatial component.  
It can answer questions on very large collections of spatial data,  
and against very complex data models.

“What parcels are  
within 1km of this  
fire?”

That sounds like a GIS question!  
How many lines of code should it take to solve it?

```
SELECT owner_phone  
FROM parcels  
WHERE ST_DWithin(  
    geom,  
    'POINT()',  
    1000 );
```

One line!

Using one spatial function, a coordinate, and a table of parcel data, we can generate a classic GIS “alert list” of people to phone about the fire.

“How far did the bus  
travel last week?”

```
SELECT
  Sum(ST_Length(geom))
FROM
  vehicle_paths
WHERE
  (v_id = 12)
AND
  (v_date > Now() - '7d');
```

History!

One SQL statement to answer a location services query.

SQL in a database is very powerful, **more powerful** than desktop GIS in terms of amount of code required and the size of datasets than can be queried.



I didn't write PostGIS. I've been a major developer since 2008.  
The first versions (0.1 to 0.7) were done by Dave Blasby on my left.  
I did the build system, documentation and Java components.  
This picture is actually a little too new, at the time of PostGIS 0.1  
(around 2001) there were only 5 people in company.

“Managing changing data in shape files is a pain in the \_\_\_\_\_!”

PostGIS was the answer to the question “how are we going to manage this changing data?!”

Name	Type
csekani-20010412.dbf	DBF File
csekani-20010412.shp	SHP File
csekani-20010412.shx	SHX File
csekani-20010421.dbf	DBF File
csekani-20010421.shp	SHP File
csekani-20010421.shx	SHX File
haida-19991213.dbf	DBF File
haida-19991213.shp	SHP File
haida-19991213.shx	SHX File
haida-20000213.dbf	DBF File
haida-20000213.shp	SHP File
haida-20000213.shx	SHX File
haida-20000219.dbf	DBF File
haida-20000219.shp	SHP File
haida-20000219.shx	SHX File
klahoose-20011023.dbf	DBF File
klahoose-20011023.shp	SHP File
klahoose-20011023.shx	SHX File
klahoose-20011203.dbf	DBF File
klahoose-20011203.shp	SHP File
klahoose-20011203.shx	SHX File

Table	Date	Geometry
Table	19991213	POLYGON()
Table	20000213	POLYGON()
Table	20000219	POLYGON()
Table	20010412	POLYGON()
Table	20010421	POLYGON()
Table	20011023	POLYGON()
Table	20011203	POLYGON()

We had a collection of “areas of interest” for treaty negotiations that were passed through modeling software. The negotiators would provide ever changing versions they wanted analyses on, so knowing what the history was, was important.

<X>

Database model could provide history, easy publication distribution, and automation of analysis runs.  
(That was the theory, in fact we never got around to using the database for this purpose.)

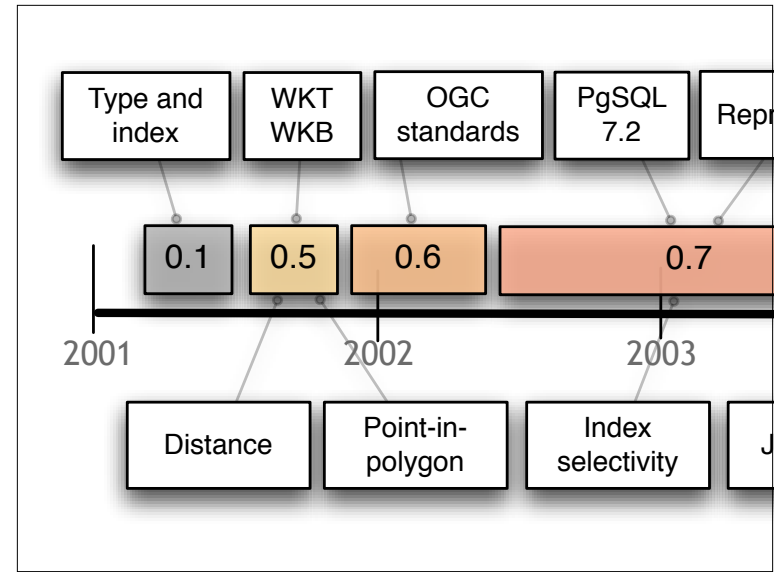
## Database Advantages

- Query all areas at once
- Publish in one step
- Manage in one place



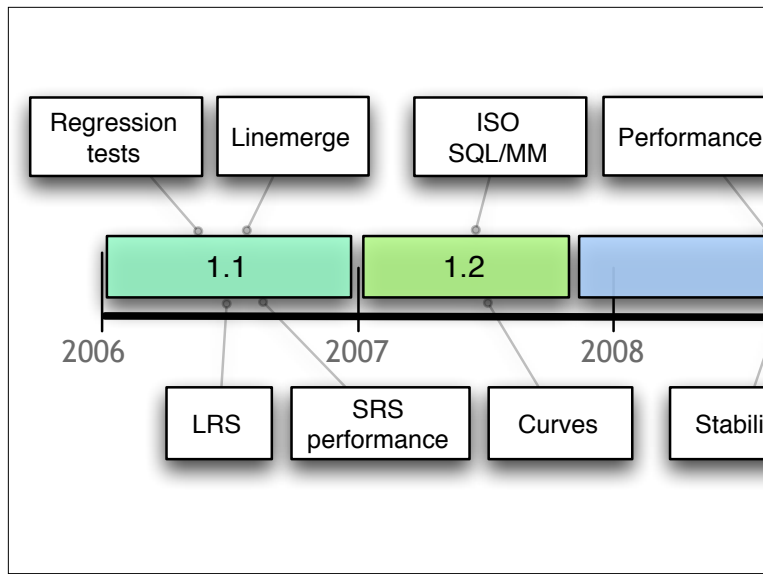
# History!!!

That was the beginning, but there is more.

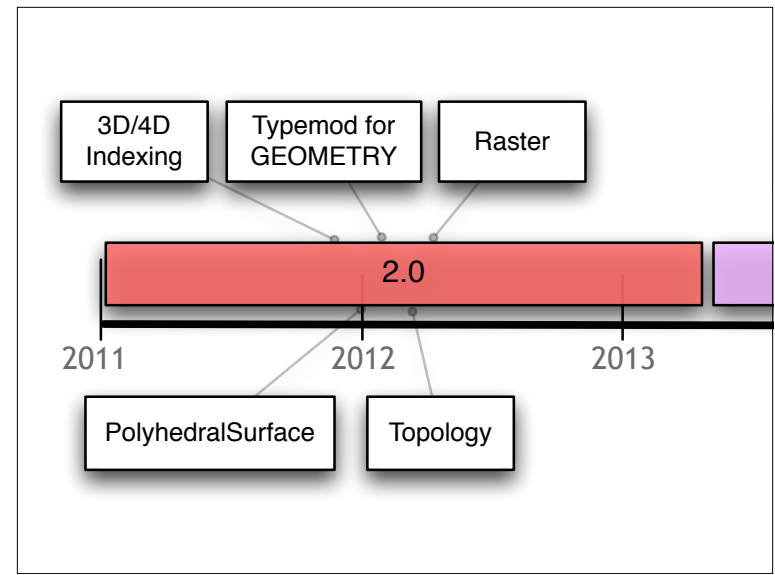


Basic functionality all done in year one (2001).  
Mapserver! P-i-P!  
Started to implement SFSQL standards in 2002.  
Complete full SFSQL in 2004.  
Changed to a new binary format in 2005 and  
complete the transition and released 1.0 in 2006.





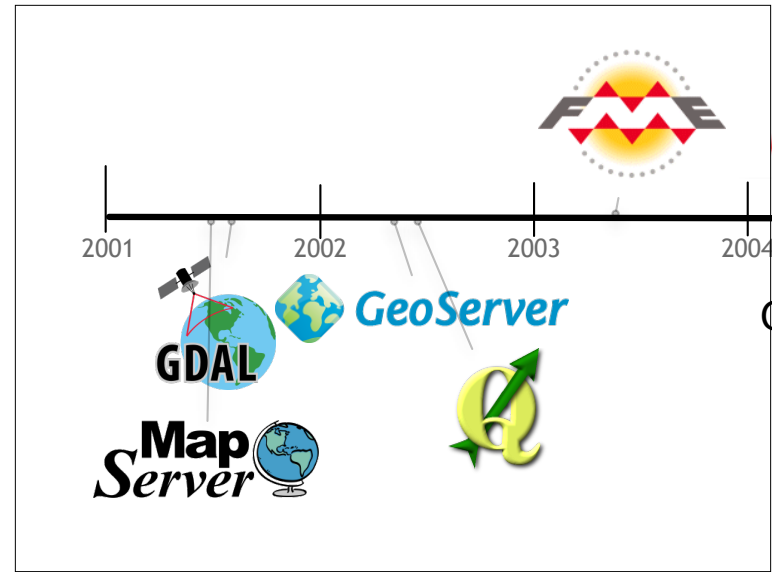
2006 and 2007 added more functions, both ISO standard ones and speciality ones like LineMerge and Polygonize.  
 2008 things stabilized and performance was the key focus  
 2009 more performance with prepared geometry and developer improvements like code reorganization  
 2010 and 1.5 added geography and GUI tools for loading

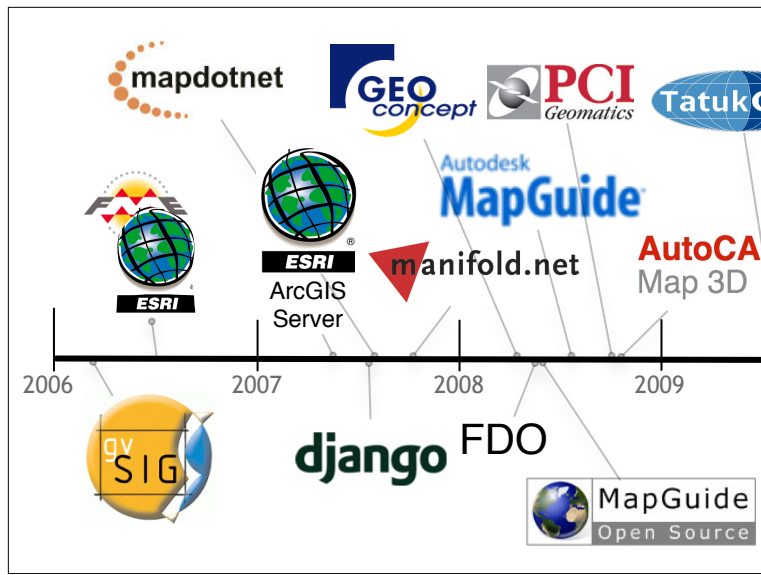


This year and most of last year has been about 2.0.  
 Changing the binary format again, adding multi-dimensional indexes, rasters, topology, 3D objects and more.  
 We're looking forward to XP, which will be the ultimate release, including a sentient interface.  
 Which may lead to some problems with the Vista release, but we remain optimistic.

# More History!!!

More to history than just software features.  
PostGIS has gone from curiosity to industry standard.





lots of proprietary companies at the end of this timeline!

“Why are these companies supporting PostGIS?”



because it makes them money



GeoConcept and CadCorp both supported PostGIS because IGN (national mapping agency of France) wanted PostGIS-enabled tools. It gave them access to more business.



Even the big three  
ESRI, MapInfo and InterGraph have come on board  
because their customers have been saying  
they want it.

Who is using  
PostGIS?

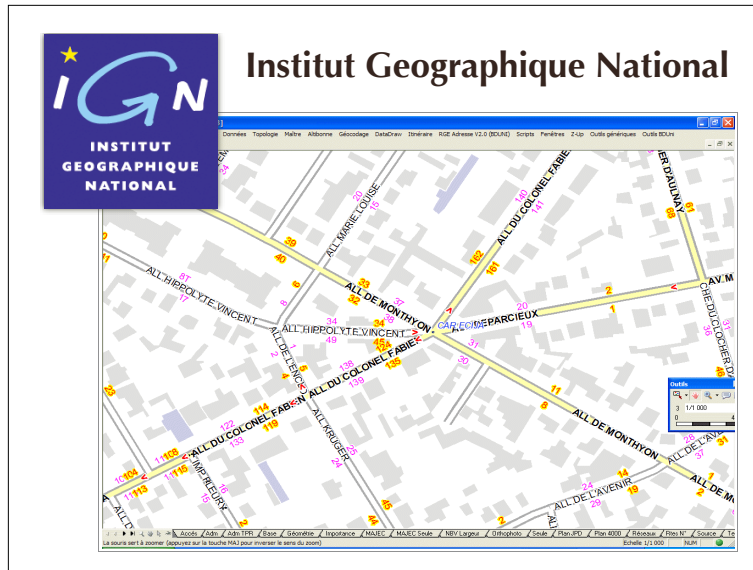
And who are those customers?  
They are legion.



On the private sector side,  
 WSI also known as the Weather channel  
 The New York Times, running PostGIS behind  
 their Django instance.  
 InfoTerra in the UK, managing all of Ordnance  
 Survey in their PostGIS  
 DigitalGlobe, managing their web-based image  
 delivery  
 SAIC, BAE, Ball Systems, all doing defence  
 systems.  
 RedFin and Zonar, startup companies,



On the government side,  
 National levels in France and Portugal and Canada  
 Regional levels like MN, and Quebec  
 Local levels like Pierce County, the City of St Paul,  
 and many more  
 and quasi-governmental outfits like NREL and  
 NavCanada all use PostGIS.



So, I mentioned (twice)  
that IGN is using PostGIS  
...what is IGN?  
...what is BDuni?



## Database Evaluation

- PostGIS? DB2? Oracle?
- Can it handle 100M spatial features?
- Can it do spatial transactions?
- Yes! Yes! Yes!

## Scalability

"Enterprise"	1 Dual-Core	2 Quad-Core
Oracle	\$40,000	\$160,000
IBM DB2	\$36,400	\$145,600
MS SQL Server	\$25,000	\$50,000
IBM Informix	\$50,000	\$200,000
PostGIS	\$0	\$0

Vendors like to talk about "scalability" as if customers have infinite money to address their problems. With modern hardware, the main cost driver in scaling up an installation is no longer hardware, it's software. You can buy a dual-quad server with 24Gb of memory from Dell for less than \$10K. Your software licensing for that server will be many times your hardware cost, and that is not a good scalability situation.

Oracle Enterprise Edition pricing 2007-07: \$40K \* NCORES \* 0.5 (for Standard Edition, use \$15K instead of \$40K)

IBM (2007-07) uses "value unit" now, <https://www-112.ibm.com/software/howtobuy/passportadvantage/valueunitcalculator/vucalc.wss>

100 value units for Intel dual core chip, 400 value units for dual Intel quads

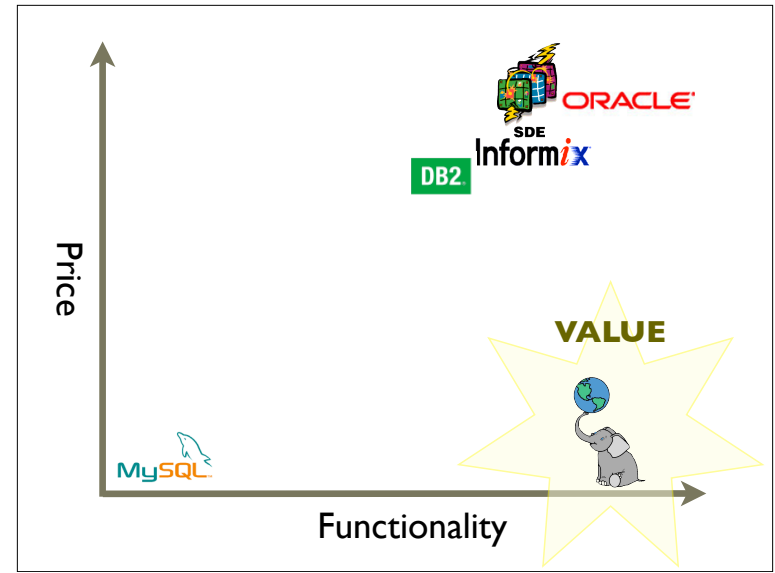
DB2 Enterprise = \$364 / value unit

DB2 Workgroup = \$100 / value unit

Informix Enterprise = \$500 / value unit

Microsoft (page updated 2005) SQLServer <http://www.microsoft.com/sql/howtobuy/default.msp>

Only multiply by processor, not cores! Nice! <http://www.microsoft.com/sql/howtobuy/multicore.msp>



other databases have good features  
or good prices  
postgis has the best price/functionality



## Cool Features!



OK, so that business talk is fine, but talk some tech to me.

SQL  
Goodness!



You have heard of the “NoSQL” movement, perhaps?  
The number one feature of PostGIS is that it puts the power of hundreds of complex, performance optimized spatial functions at your fingertips through powerful declarative language we call SQL

Costed,  
Planned  
Spatial Queries!

Try this on  
MySQL!

```
SELECT ...  
FROM geotable_a a  
JOIN geotable_b b  
  ON ST_Intersects(b.geo,a.geo)  
JOIN attrtable_c c  
  ON (b.id = c.id)  
JOIN attrtable_d d  
  ON (a.id = d.id)
```

The spatial type is fully integrated into the query planner.  
Even the PostgreSQL gurus at PgCon didn't know this.  
(So I added this slide)  
That means that complex multi-table queries like this one execute efficiently.

Fun  
Formats!



```
ST_AsGeoJSON()  
ST_AsGML()  
ST_AsKML()  
ST_GeomFromGML()  
ST_GeomFromKML()
```

Thanks to the work of Oslandia, PostGIS supports a veritable zoo of XML and other hipster formats like JSON for both output and input.

## Geometry Construction!

ST\_Polygonize()  
ST\_LineMerge()  
**ST\_Union()**  
ST\_Collect()  
ST\_BuildArea()

We have exposed the geometry processing capabilities of the GEOS C++ library to allow fancy construction of geometries (come to my Friday talk for more on these)

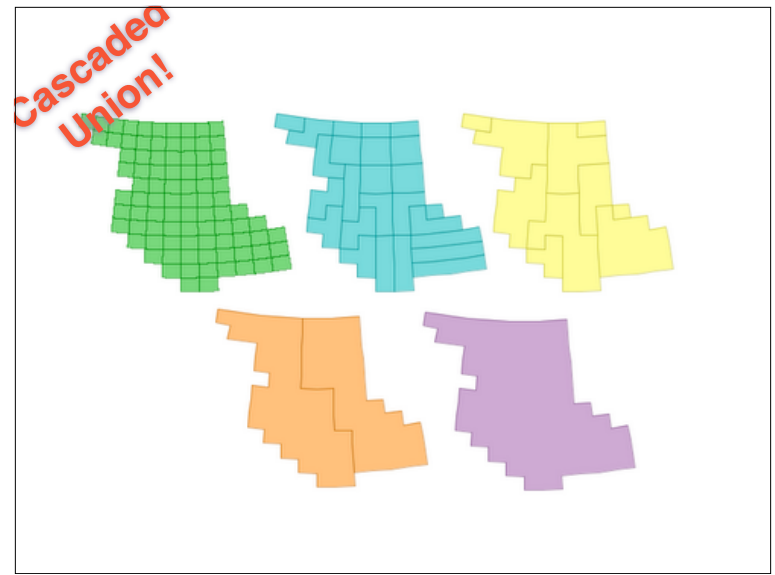
## Linear Referencing!

ST\_LocateAlong()  
ST\_LocateBetween()  
ST\_AddMeasure()  
ST\_Line\_Locate\_Point()

There is a set of linear referencing functions so you can build data models that include both measured and proportional linear references.



And our standard functions keep getting faster.  
 The cascaded union improvement came in version 1.4  
 The picture is the example we were sent:  
 “why is this operation so slow” he asked  
 Cascaded union merges polygons in the optimal  
 order.  
 It made this example 40 times faster.



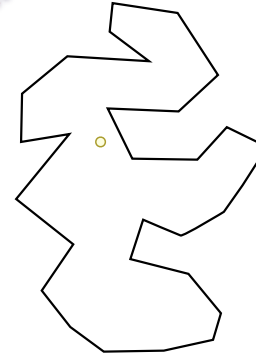
Cascaded union does that melting process in a  
 particular order  
 Merging neighbors first  
 This example became 5 times faster.

Prepared  
Geometry!

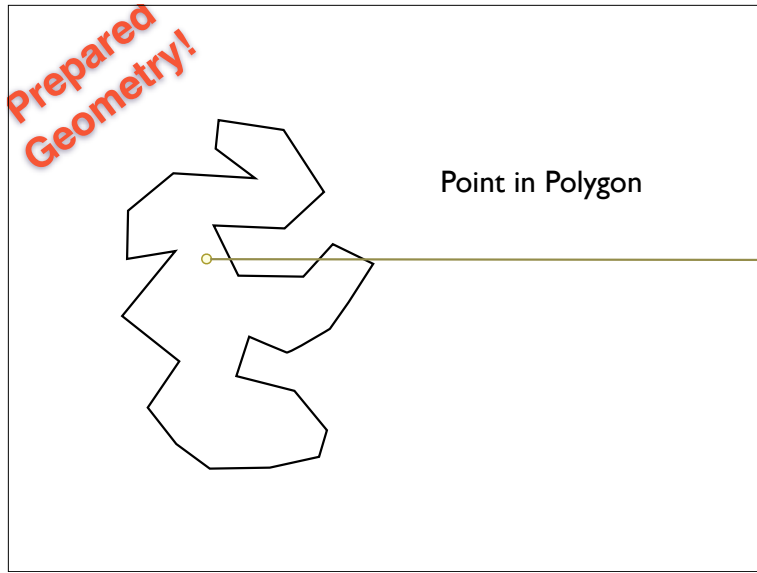
```
SELECT ...  
FROM points, polygons  
WHERE ST_Intersects (  
    polygons.geom,  
    points.geom  
)
```

Similarly, prepared geometries make standard spatial join queries faster.

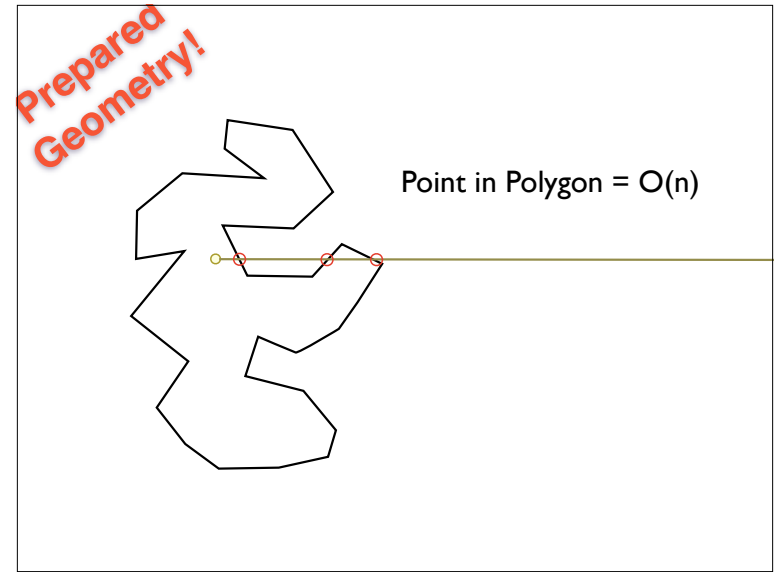
Prepared  
Geometry!



Calculating (for example) a point-in-polygon is quite expensive.



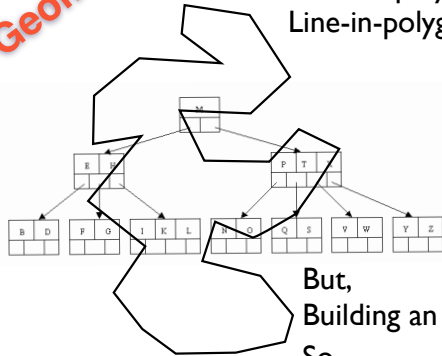
First you draw a stab line.  
Then you check every edge to see if it hits the line.  
Then you count up the number of hits.  
Odd => inside.  
Even => outside.



This point has three hits, so it's inside.  
Calculate cost is proportional to the number of edges.

Prepared  
Geometry!

Build spatial index on edges!  
Point-in-polygon ==  $O(\log(n))$   
Line-in-polygon ==  $O(m \cdot \log(n))$



But,  
Building an index takes  $O(n)$   
So,  
Cache index and re-use it!

But, if we index the edges, we can get the cost down to  $\log(N)$ !  
But indexing the edges takes  $O(n)$  time!  
So we temporarily cache the index the first time we build it  
and use it on subsequent queries that hit the same polygon.

Prepared  
Geometry!

Prepared geometry makes repeated tests on large geometries very fast.

(**ST\_Intersects**,  
**ST\_Contains**)

that's fun geek talk  
important to remember...  
between 2 and 5 times faster depending on complexity of inputs  
(more complex, means better improvement)

## Curves!

- CURVESTRING
- COMPOUNDCURVE
- CURVEPOLYGON
- ST\_CurveToLine()
- **ST\_LineToCurve()**



Since 1.2, we've had curve types which are part of ISO SQL/MM standard. And our curve support has been getting more complete with each release. You can convert curves to linestrings, and even convert linestrings to curves! Curve types are useful for storing CAD data, which uses curves.

## Geography!

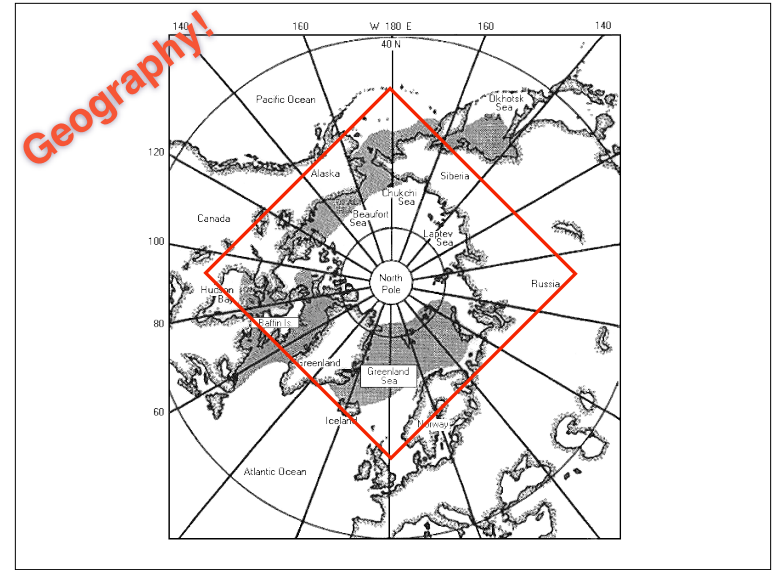


In 1.5, we added the geography type models data on a sphere.

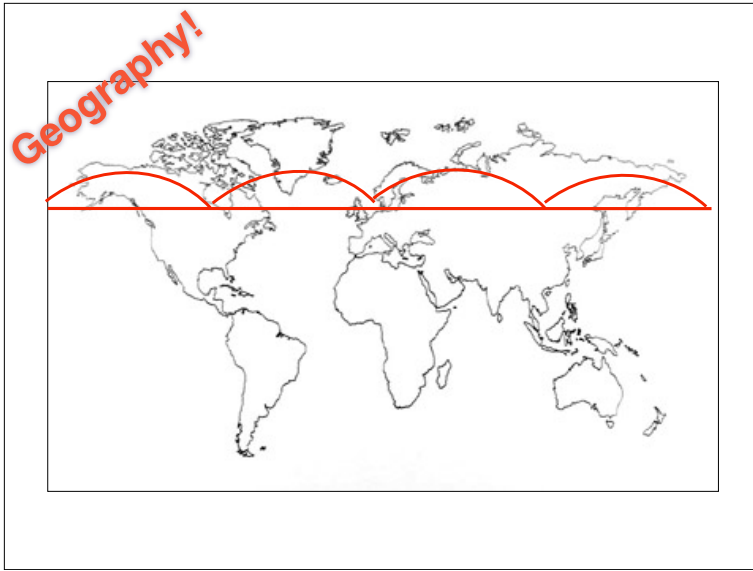




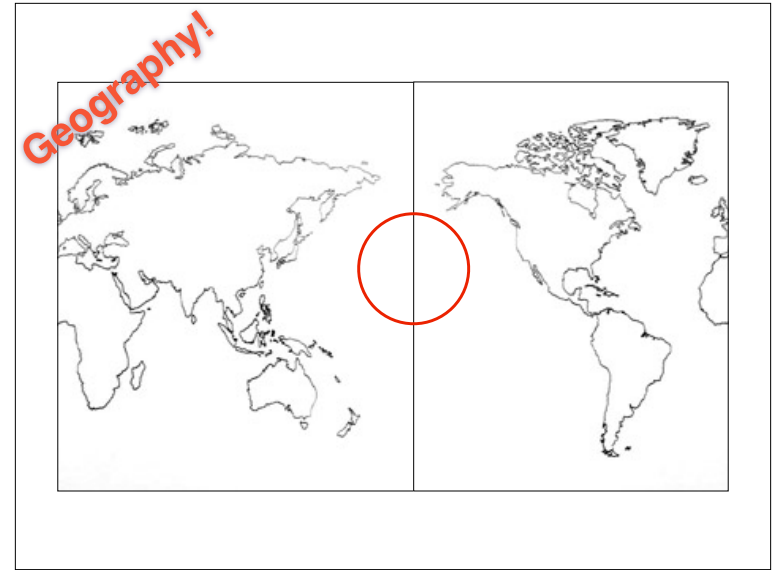
“geometry” type models data on a plane but that Simple Plate-Carre view, has lots of problem cases.



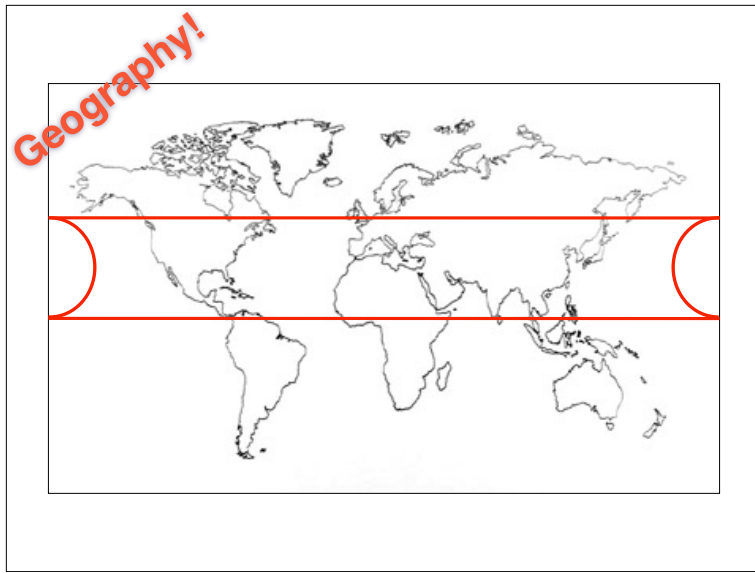
a polygon that covers the pol



is wrong if you interpret it on the plane

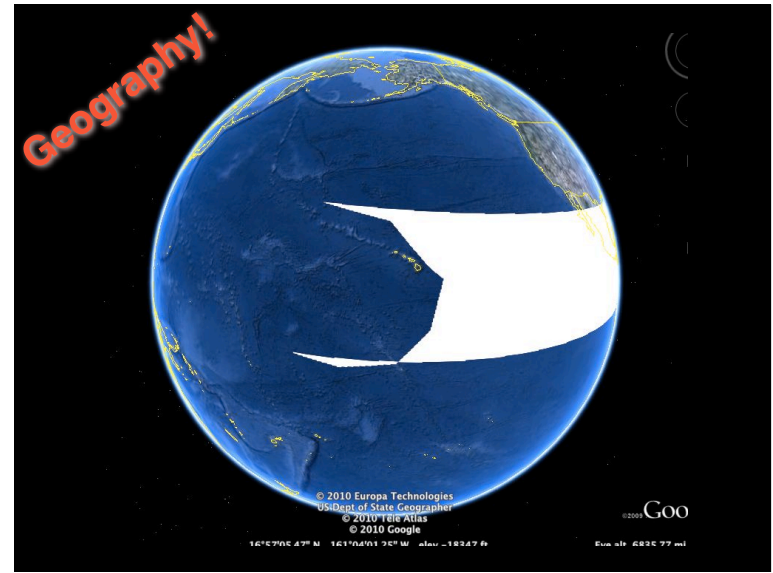


a polygon that crosses the dateline



thinks it crosses the whole world if you interpret it on the plane

this wasn't just a problem for postgis



even google earth has a hard time with the dateline and polygons also the poles

**Geography!**

GeoNewbies

Who is geography for?

“I want to find all the address points within one mile!  
My data is in lat/lon!  
Google Maps rocks!”





There two kinds of users who find GEOGRAPHY useful  
the first is “geonewbies”, users who do not know any GIS or anything about map projections  
GEOGRAPHY lets them work with lat/lon data without knowing about projections

**Geography!**

GeoHugies

Who is geography for?

“Yeah, I own a freaking satellite, you got a problem with that?”

The “geohugies” are really big organizations that have truly global data, that covers the poles and datelines and everything in between there is no map projection that works for them

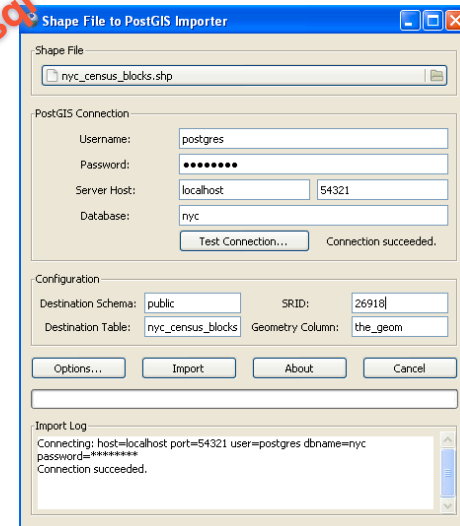
Geography!

- Indexes spherical data
- ST\_Intersects()
- ST\_Distance()
- ST\_DWithin()
- ST\_Area()
- **Casts** to/from **GEOMETRY**



the implementation of GEOGRAPHY currently only has support for a few functions  
but you can use casts to convert to GEOMETRY  
and access all the geometry function

shp2pgsql



PostGIS 1.5 added a GUI loader tool  
The 2.0 version includes the ability to load  
multiple  
files in a batch!

**NEW! 2.0!**

## TypMod

```
CREATE TABLE
  my_spatial_table
(
  id INTEGER,
  name VARCHAR(64),
  geo GEOMETRY(Point,26910)
);
```

In 2.0 (come on Friday to see Regina and Leo's talk)  
about the 2.0 "good stuff" (don't know if this counts)  
"TypMod" support will let you declare the type and srid and dimensionality of a geometry right at table CREATE time.

**NEW! 2.0!**

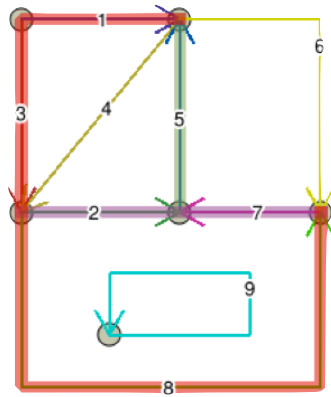
## 3D/4D Index

```
CREATE INDEX my_index
ON my_spatial_table
USING GIST (
  geom,
  gist_nd_geometry_ops
);
```

Yes, in 2.0 you will be able to create indexes and search in 3D and 4D!

NEW! 2.0!

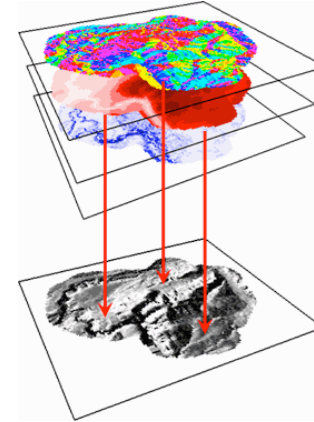
## Topology



In 2.0 you'll be able to create topologies!

NEW! 2.0!

## Raster for Analysis



raster is not for image storage  
for analysis  
vector->raster->vector  
map algebra, operations in raster space

NEW! 2.0!

## 3D Objects



polyhedral objects are good for modelling buildings  
3D and 4D indexes will be good for fast retrieval of those objects

NEW! 2.0!

## Indexed Nearest-Neighbor Search

sponsored by   
vizzuality

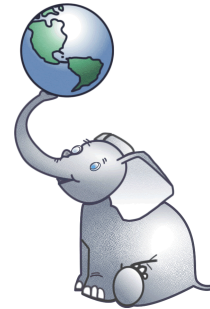
And (this hasn't been coded yet, but before October!)  
index-based nearest-neighbor searching!  
Find me the closest X to Y.



# 2.0!?!?

When is 2.0 coming?!?!?  
We have to stabilize the features we have, and  
close tickets  
Realistically, not before early 2012.  
Yes, we want it to be done too!

## The State of PostGIS



pramsey  
@  
 OPENGEO  
.org

Thanks!