PostGIS
GEO in your DB

Vincent Picavet
Oslandia

OSLANDIA

Context

Bisonvert.net

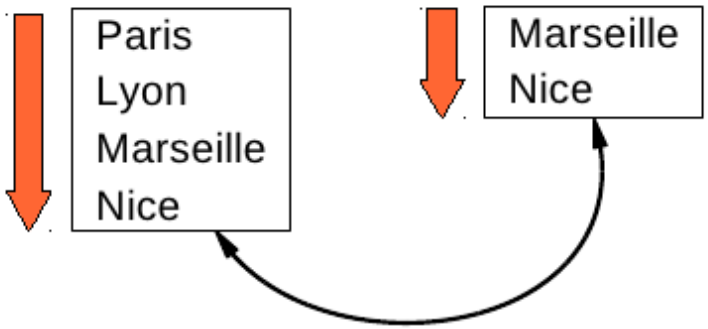
Car-sharing free software

Goal :

match people doing the same journey

Current method

Match from/to via names



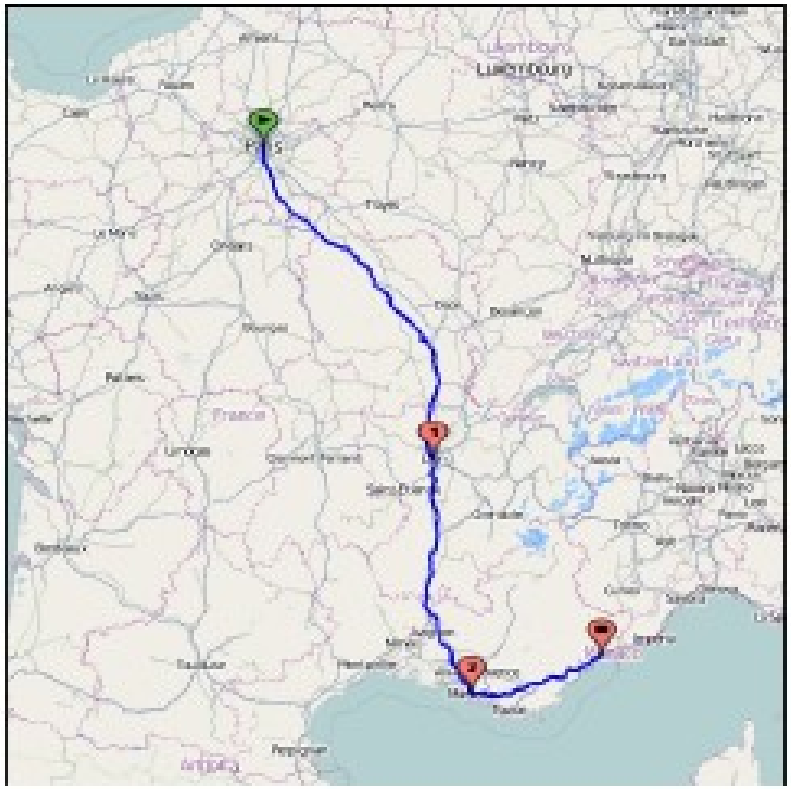
OSLANDIA

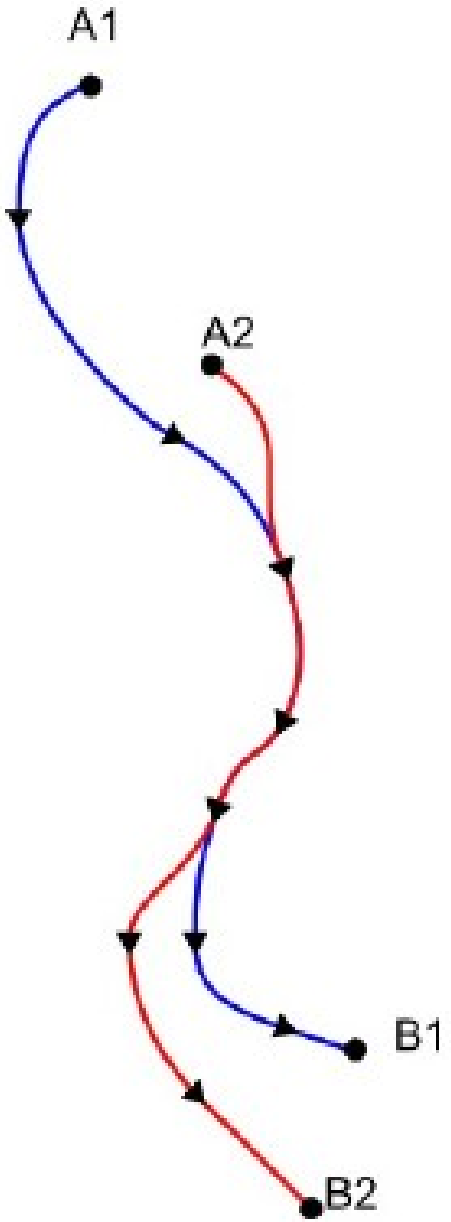
Solution :

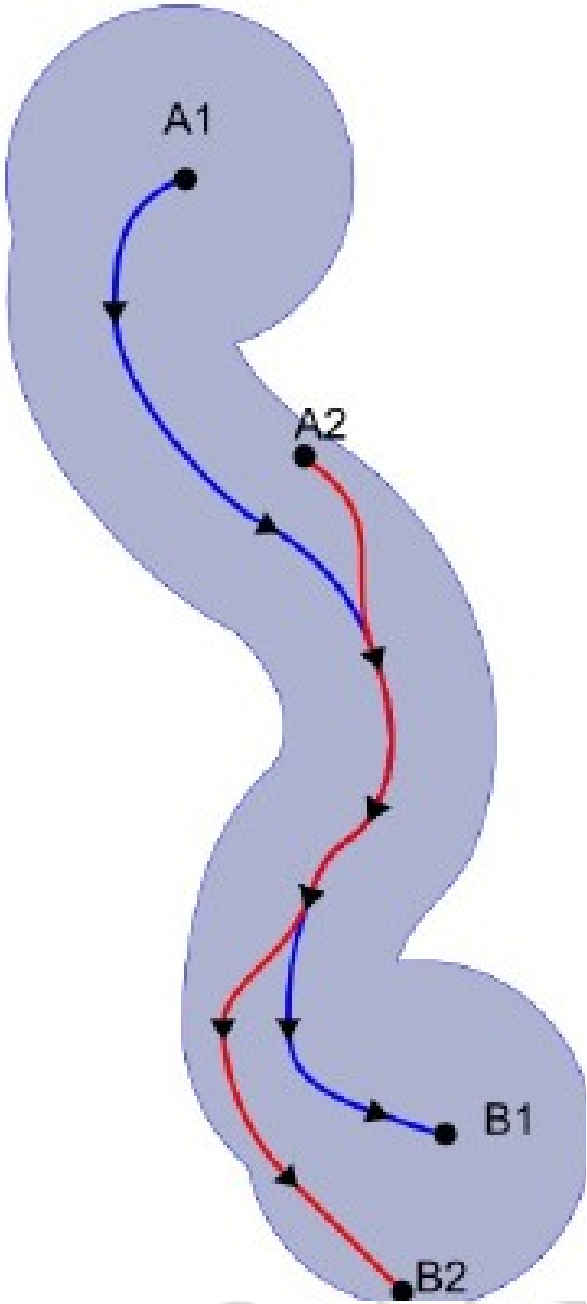
Use real paths

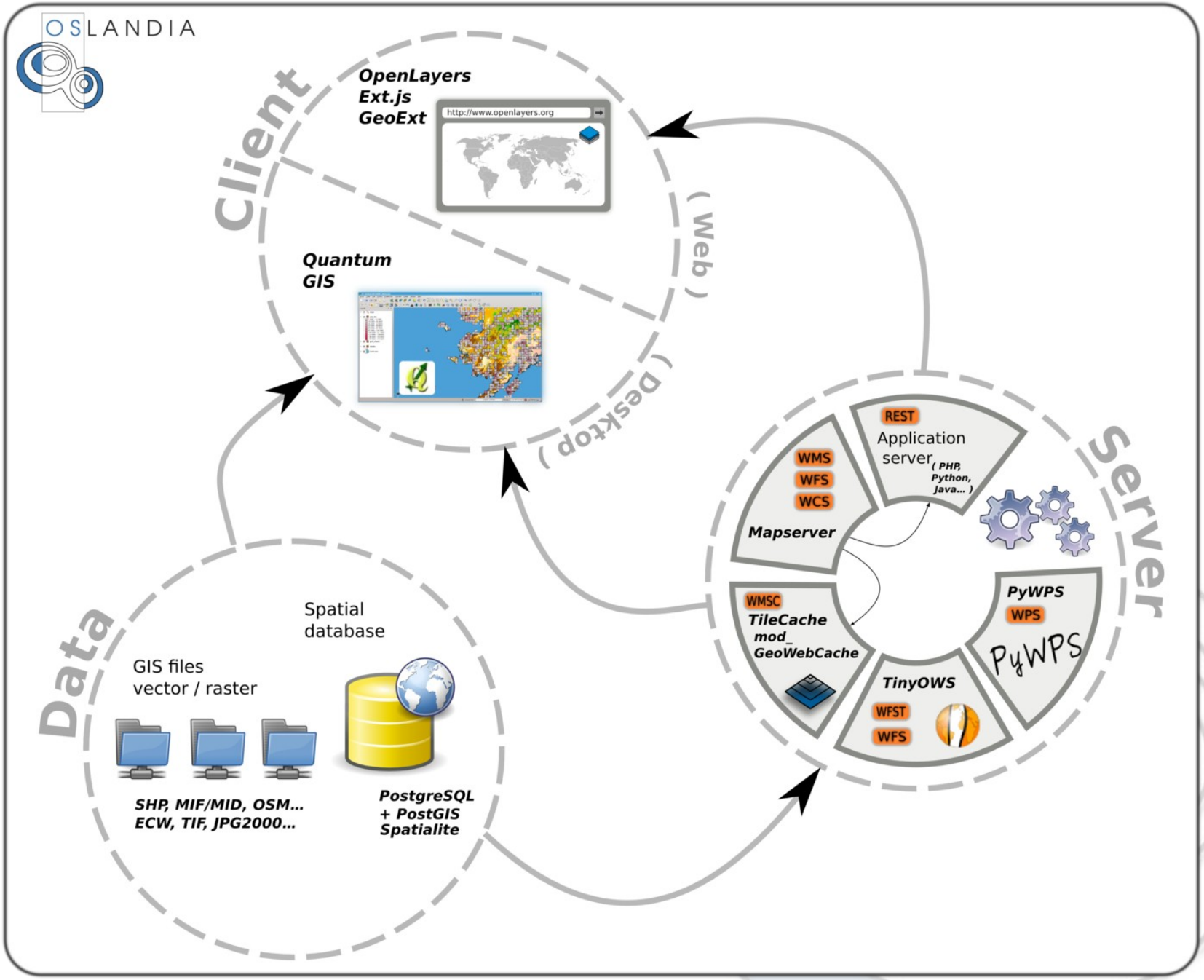
1/ Compute path (routing)

2/ Match paths
(Spatial analysis)








OSLANDIA
Client
OpenLayers
Ext.js
GeoExt
http://www.openlayers.org
Quantum
GIS
(Web)
(Desktop)
Server
REST
Application
server
(PHP,
Python,
Java...)
WMS
WFS
WCS
Mapserver
WMSC
TileCache
mod_
GeoWebCache
PyWPS
WPS
PyWPS
TinyOWS
WFST
WFS
Data
Spatial
database
GIS files
vector / raster
SHP, MIF/MID, OSM...
ECW, TIF, JPG2000...
PostgreSQL
+ PostGIS
Spatialite

PostGIS Project

OSLANDIA

Spatial RDBMS

- Geometry + attributes = «feature»
- SQL queries
  - Attributes
  - Spatial
- High load
  - Big data
  - Long and complex processing
- Acceptable performances
- Respect standards

OSLANDIA

Standards

International standards

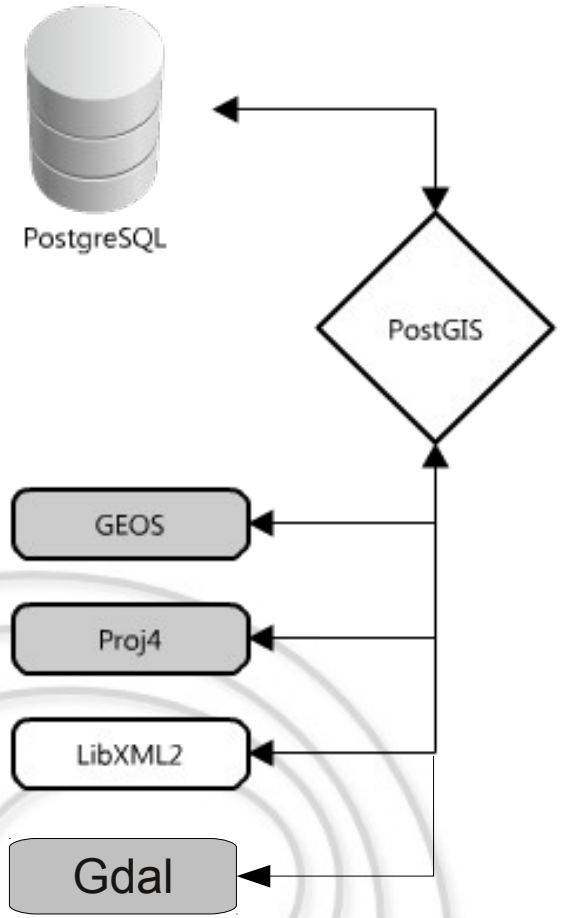
Specifications

- OGC SFS (Simple Feature for SQL)
- ISO SQL/MM part 3

What's in :

- Geometry types
- Spatial functions prototypes
- Tables and additional processing for referential integrity

Overview



Plugin PostgreSQL
Written in C
Based on external libs →
Implements OGC SFS 1.1
+ ISO SQL/MM
+ lots of extra features

History

2001 first alpha version

2003 version 0.8 – production ready

2005 version 1.0

- Core rewrite, LWGEOM
- OGC SFS 1.1 compatibility

2006 version 1.2

- Aim at ISO SQL/MM (curves, ST_ prefixes…)

2009 version 1.4

- PSC and OSGeo incubation

2011 version 1.5.3

- Geodesy

2012 version 2.0

- Raster

2013 version 2.1

- 3D

Users

Public users

- **IGN : French nat. geographical institute**
- **IRSN : Institut de Radioprotection et de sûreté Nucléaire**
- **EEA : European Environment Agency**
- **NOAA …**

Private sector

- **France Telecom, Infoterra, Digital Globe, Mediapost, Mappy, NY Times…**
- **Heroku !**

A lot of others !

OSLANDIA

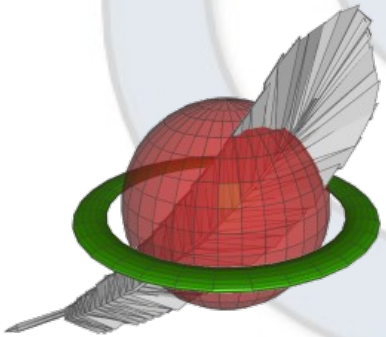
Community

Worldwide
Thousands of users
Very active postgis-users ML

Commiters :
LisaSoft
OpenGeo
Oslandia
CadCorp
Paragon Corporation
Refractions Research
Sandro Santilli
Sirius
Some others and individuals…

What else ?

Oracle spatial (and locator)
ESRI ArcSDE
IBM DB2
MS SQL Server (>2008)
Actian
Spatialite
Sybase (last version)



OSLANDIA

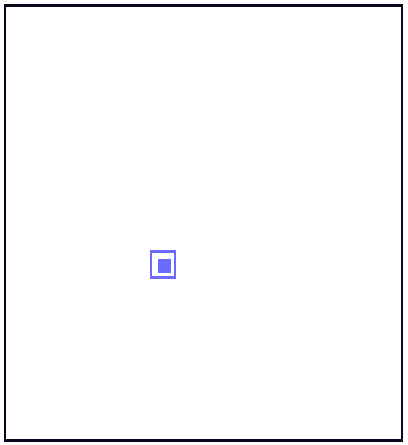
Basics

OSLANDIA

Geometry

- Geometry (or HEWKB)
  - Native database storage
  - Binary format with hexadecimal encoding
- WKT (Well Known Text)
  - Textual representation
- Dimensions
  - 2D, 3D, or 4D
- Projection system id (SRID)

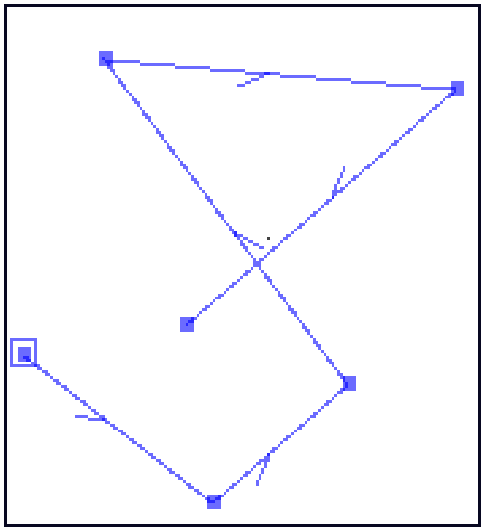
Point



POINT (10 10)

OSLANDIA

Line

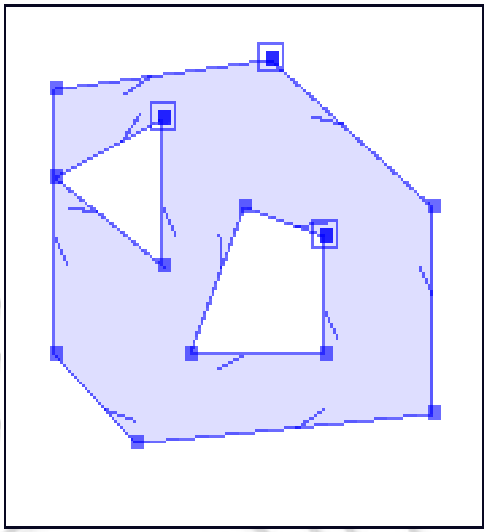


```
LINESTRING
 (
  0 5, 5 1, 9 4, 2 14, 14 13, 4 4
 )
```

OSLANDIA

```
POLYGON
(
 (9 13,13 9,13 3,4 2,1 4,1 12, 9 13),
 (5 11,5 6,1 9,5 11),
 (10 7, 10 4, 6 4, 8 8, 10 7)
)
```

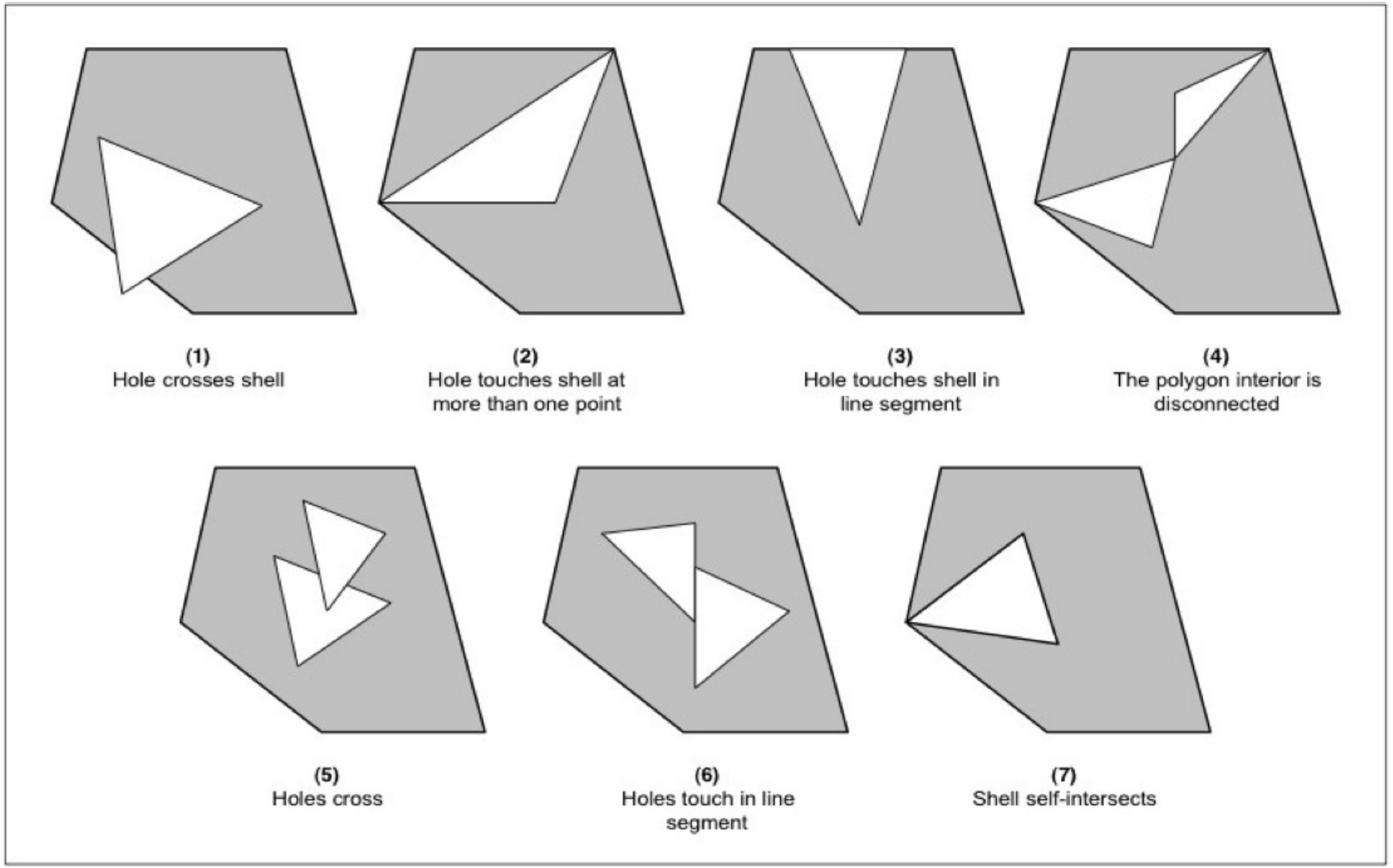
Polygon



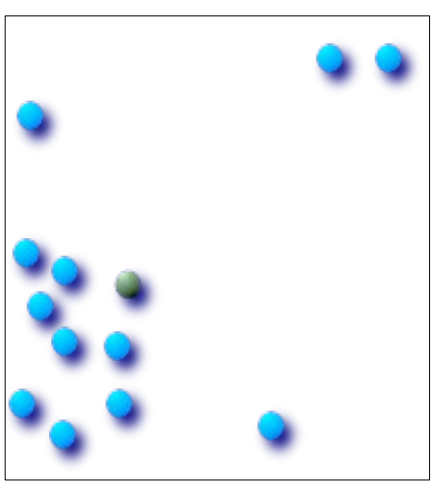
1) Mandatory first ring is external ring
2) Rings coordinates must be closed

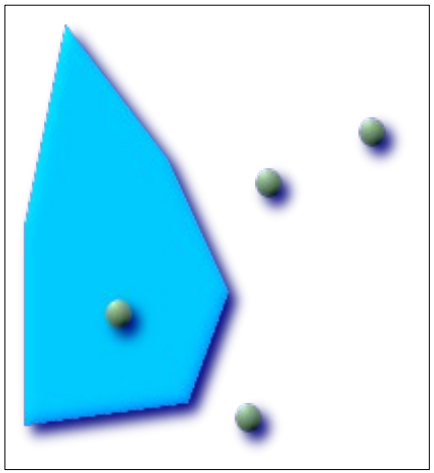
OSLANDIA

SFS (in)validity



Different projection systems cannot be mixed
Neither can different dimensions





MULTIPOINT
MULTILINESTRING
MULTIPOLYGON
GEOMETRYCOLLECTION

Multi / Aggregate

OSLANDIA

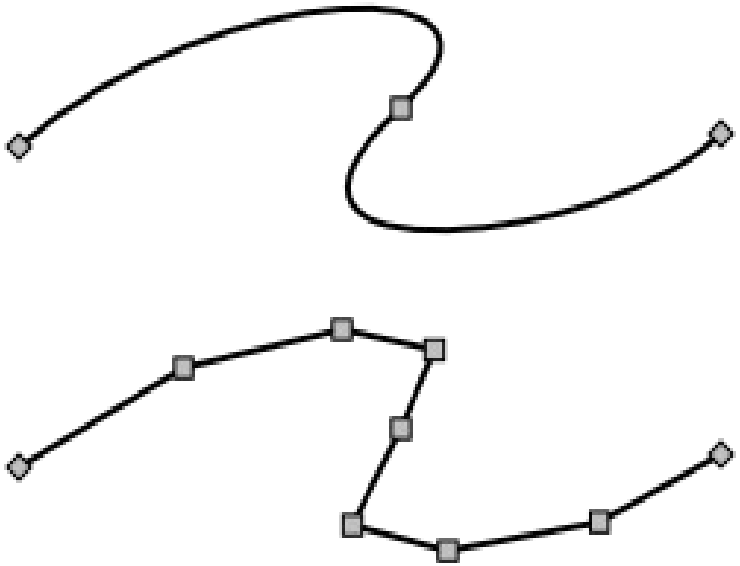
Curves

«curves» types :

CIRCULARSTRING

COMPOUNDCURVE

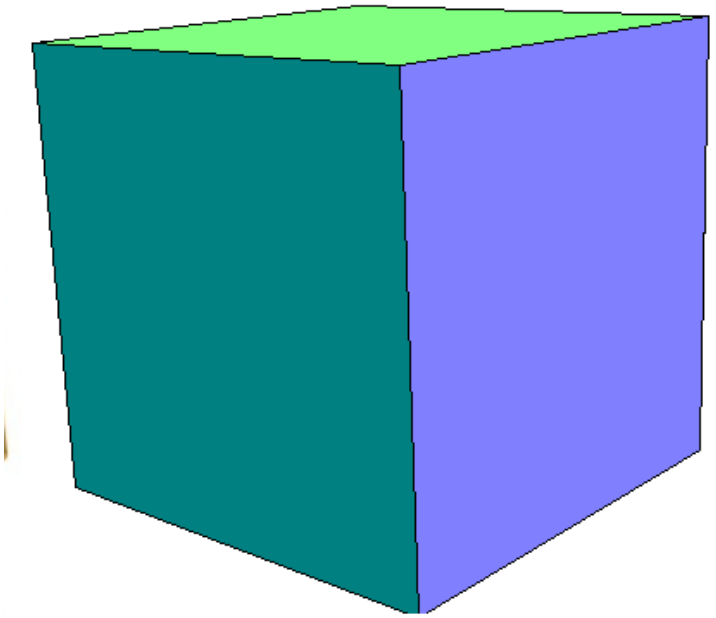
MULTISURFACE



OSLANDIA

PolyhedralSurface

```
PolyhedralSurface(((0 0 0, 0 0 1, 0 1 1, 0 1 0, 0 0 0)),
                  ((0 0 0, 0 1 0, 1 1 0, 1 0 0, 0 0 0)),
                  ((0 0 0, 1 0 0, 1 0 1, 0 0 1, 0 0 0)),
                  ((1 1 0, 1 1 1, 1 0 1, 1 0 0, 1 1 0)),
                  ((0 1 0, 0 1 1, 1 1 1, 1 1 0, 0 1 0)),
                  ((0 0 1, 1 0 1, 1 1 1, 0 1 1. 0 0 1)))
```



metadata

	oid	f_table_catalog [PK] character va	f_table_schema [PK] character v	f_table_name [PK] character v	f_geometry_column [PK] character varyi	coord_dimension integer	srid integer	type character varying(30)
1	709958	''	public	dept	the_geom	2	27582	MULTIPOLYGON
2	709957	''	public	world	the_geom	2	4326	MULTIPOLYGON

geometry_columns : spatial fields catalog

	srid [PK] integer	auth_name character var	auth_srid integer	srtext character varying(2048)	proj4text character varying(2048)
1	2000	EPSG	2000	PROJCS["Anguilla 1957 / British We	+proj=tmerc +lat_0=0 +lon_0=-62 +k=0.999
2	2001	EPSG	2001	PROJCS["Antigua 1943 / British We	+proj=tmerc +lat_0=0 +lon_0=-62 +k=0.999
3	2002	EPSG	2002	PROJCS["Dominica 1945 / British W	+proj=tmerc +lat_0=0 +lon_0=-62 +k=0.999

spatial_ref_sys: projection systems catalog

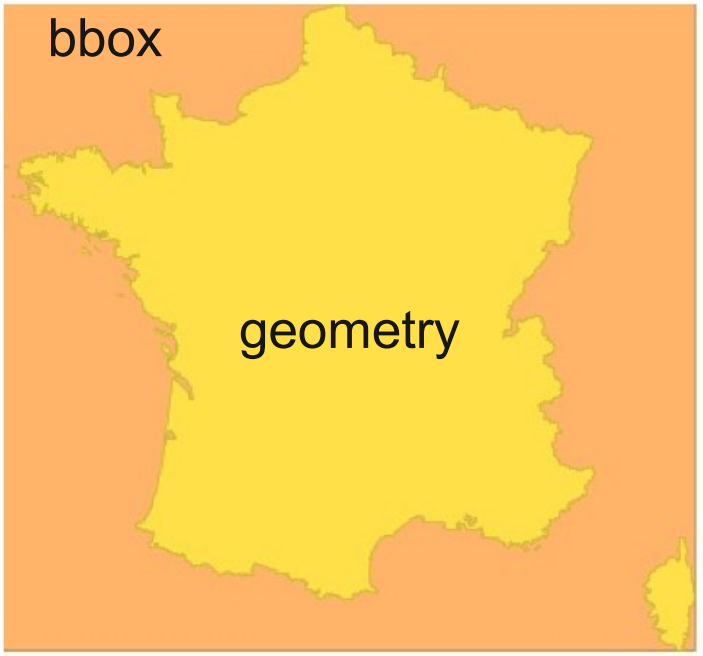
geography_columns : geography fields view

Spatial index

Better spatial filter performance

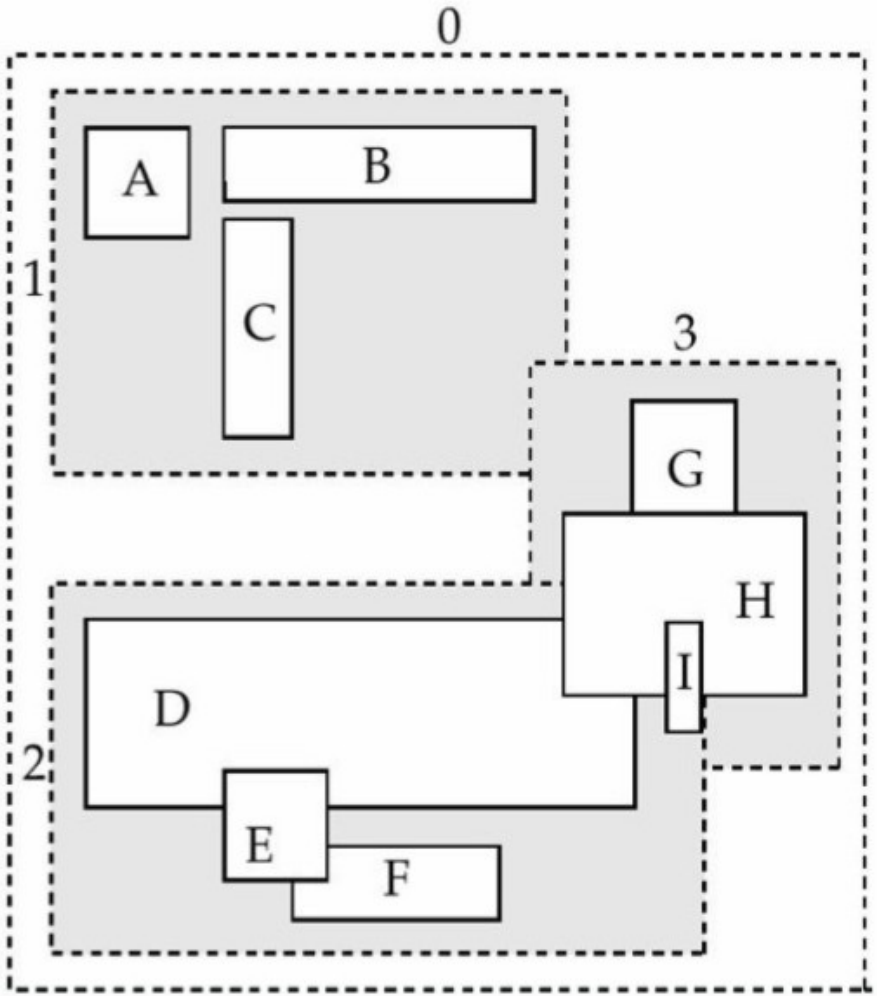
Geometry approximation with bbox

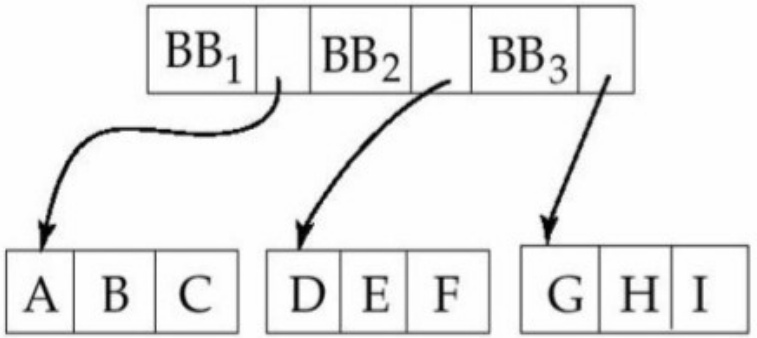
Spatial index creation :




```
CREATE INDEX index_name ON table_name
USING GIST(geom_column_name);
```

R-TREE

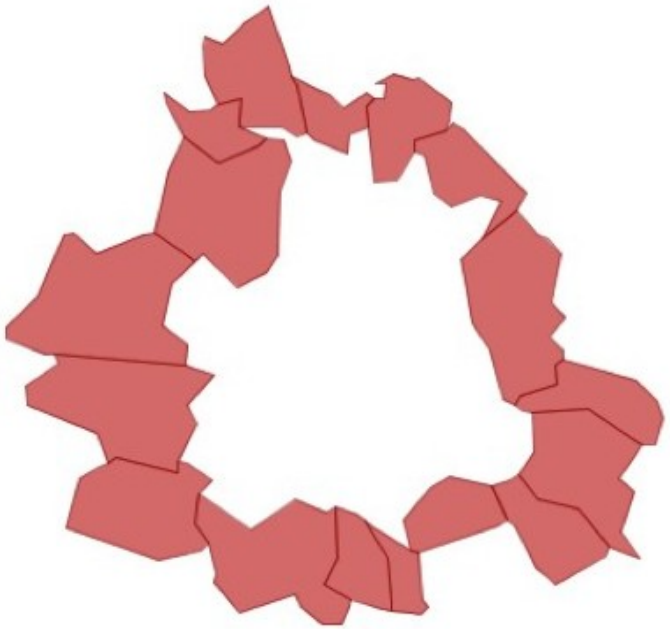




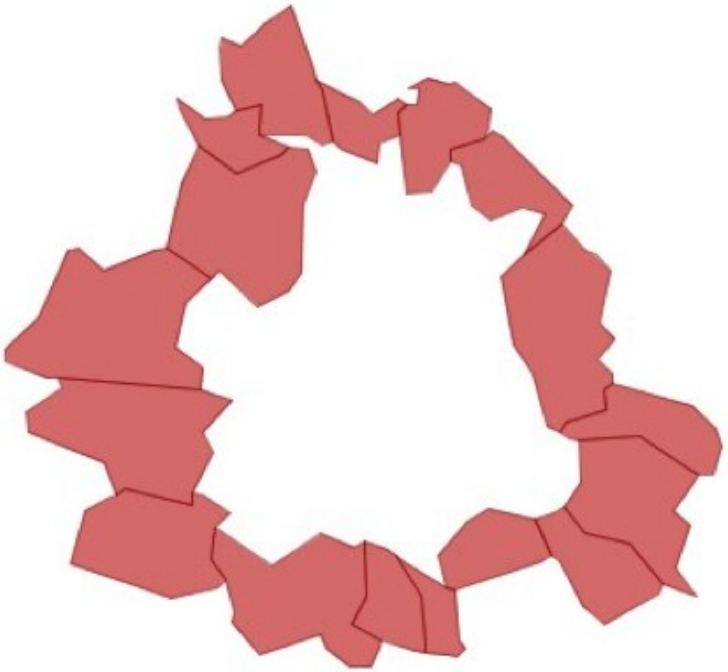
Bounding boxes are grouped in regions of the index

Indexes / perf

```
SELECT
  c1.nom
FROM
  communes c1, communes c2
WHERE
  c2.nom = 'Toulouse'
  AND ST_Touches(c1.the_geom, c2.the_geom);
```



Sans index: temps = 150 ms



Avec index: temps = 30 ms

Documentation

Documentation:

Regina Obe,
Kevin Neufeld

Improved in 1.4

Name

ST_Simplify — Returns a "simplified" version of the given geometry using the Douglas-Peuker algorithm.

Synopsis

```
geometry ST_Simplify(geometry geomA, float tolerance);
```

Description

Returns a "simplified" version of the given geometry using the Douglas-Peuker algorithm. Will actually do something only with (multi)lines and (multi)polygons but you can safely call it with any kind of geometry. Since simplification occurs on a object-by-object basis you can also feed a GeometryCollection to this function.

Note! Note that returned geometry might loose its simplicity (see ST_IsSimple)

Note! Note topology may not be preserved and may result in invalid geometries. Use (see ST_SimplifyPreserveTopology) to preserve topology.

Performed by the GEOS module.

Availability: 1.2.2

Examples

A circle simplified too much becomes a triangle, medium an octagon,

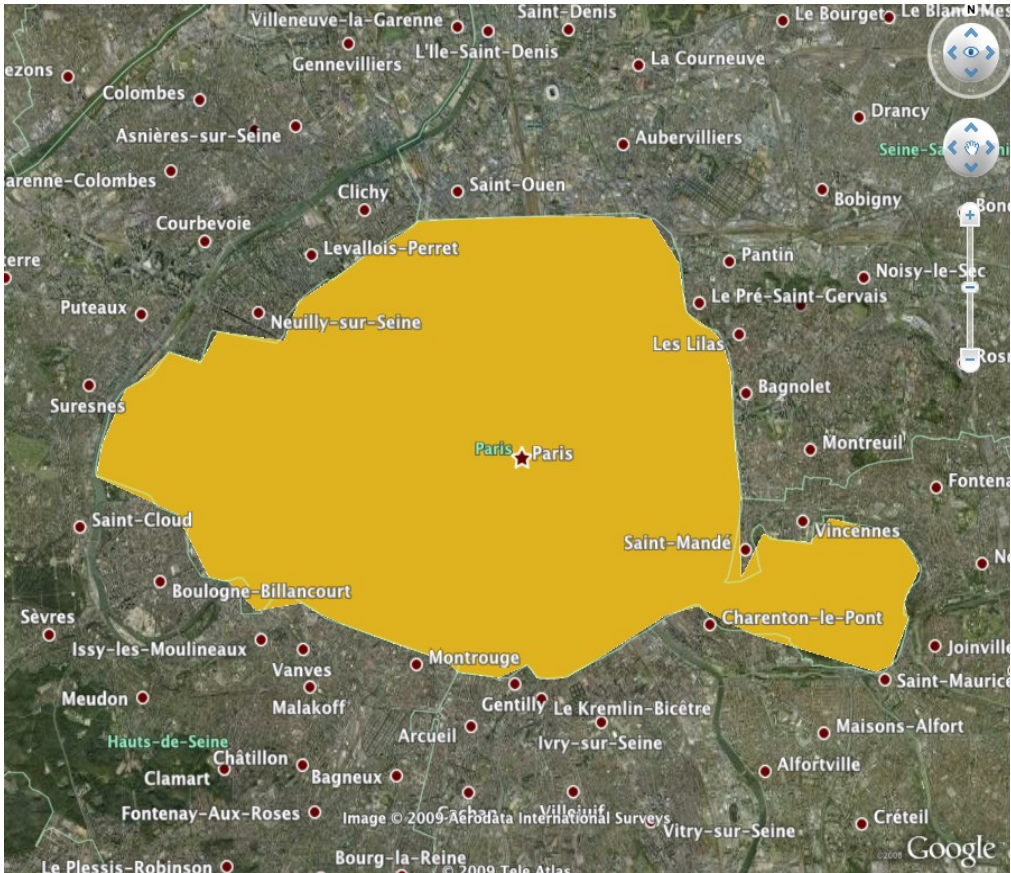
```
SELECT ST_Npoints(the_geom) As np_before, ST_NPoints(ST_Simplify(the_geom,0.1)) As np01_notbadcircle, ST_NPoints(ST_Si
ST_NPoints(ST_Simplify(the_geom,1)) As np1_octagon, ST_NPoints(ST_Simplify(the_geom,10)) As np10_triangle,
(ST_Simplify(the_geom,100) is null) As  np100_geometrygoesaway
FROM (SELECT ST_Buffer('POINT(1 3)', 10,12) As the_geom) As foo;
-result
 np_before | np01_notbadcircle | np05_notquitecircle | np1_octagon | np10_triangle | np100_geometrygoesaway
-----------+-------------------+---------------------+-------------+---------------+------------------------
        49 |                33 |                  17 |           9 |             4 | t
```

OSLANDIA

Functions

KML

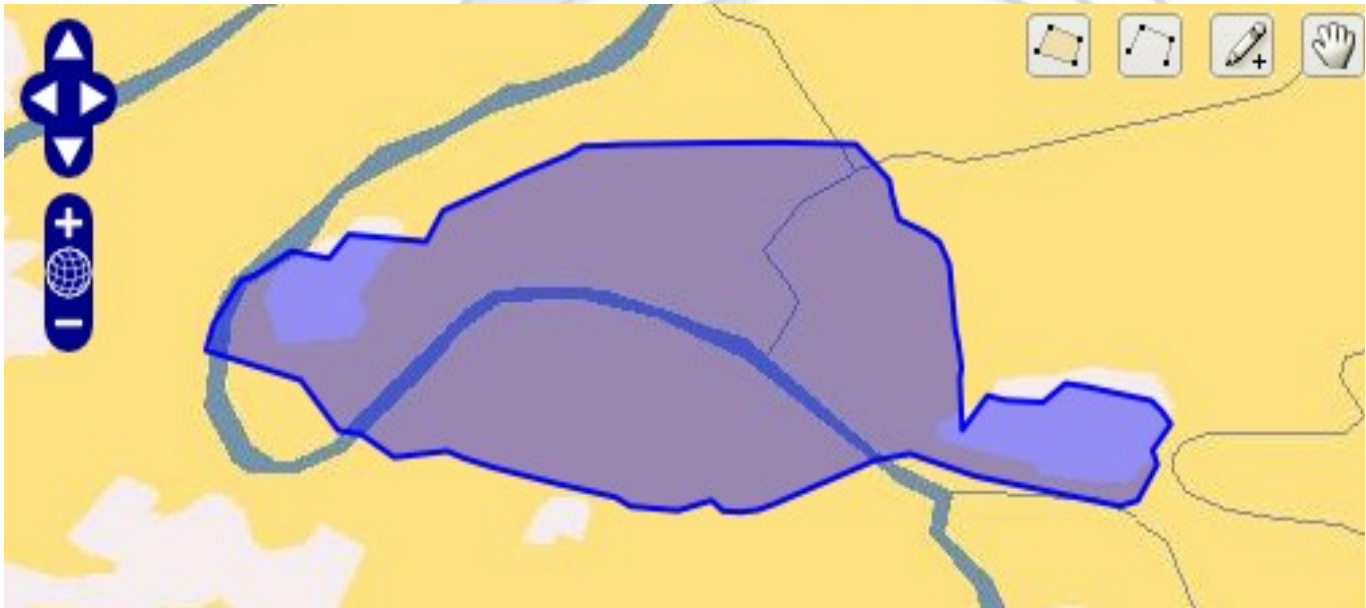
```
SELECT ST_AsKML(the_geom, 5)
FROM dept
WHERE code_dept='75' ;
```



OSLANDIA

GeoJSON

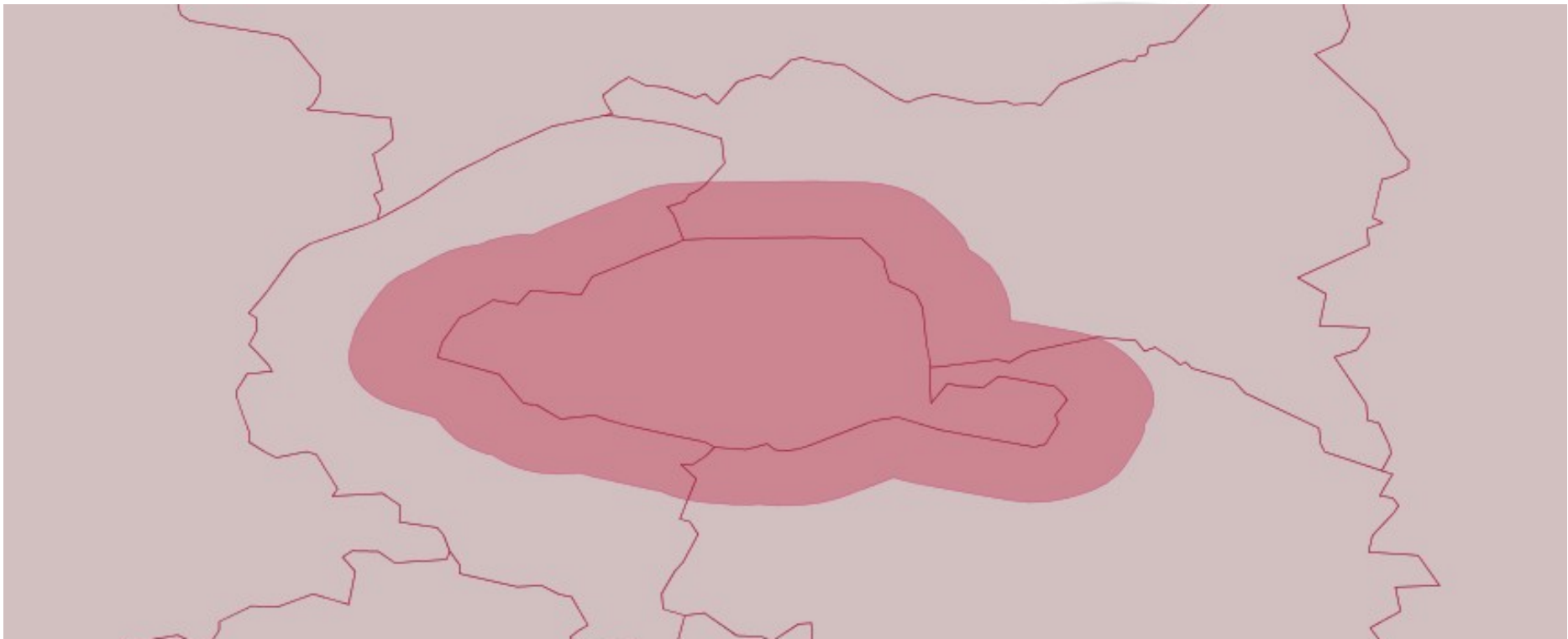
```
SELECT ST_AsGeoJSON(
  ST_Transform(the_geom, 4326), 5)
FROM dept
WHERE
  code_dept='75' ;
```



OSLANDIA

Buffer

```
SELECT ST_Buffer(the_geom, 2500)
FROM dept
WHERE code_dept='75';
```

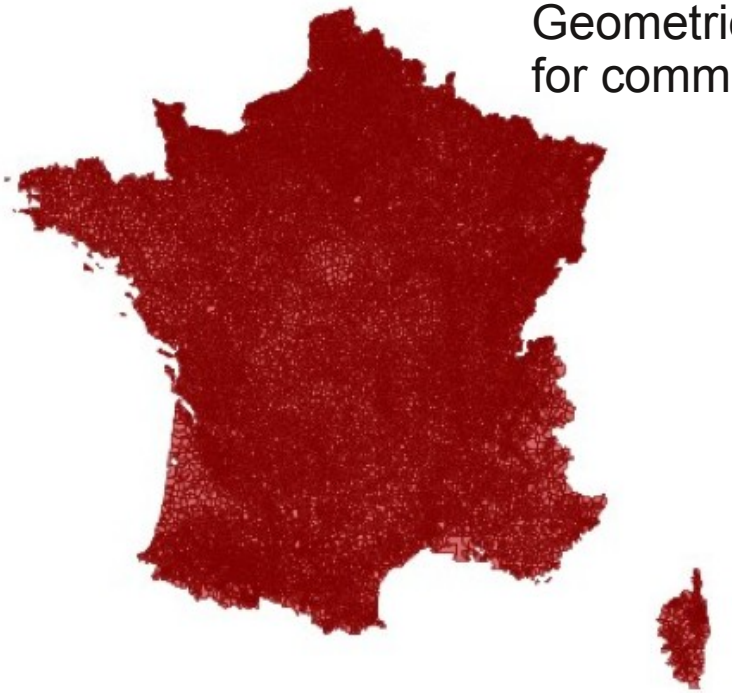


OSLANDIA

Aggregate

```
SELECT ST_Union(the_geom)
FROM commune
GROUP BY code_dept;
```

Geometries
for communes



Geometries for departments
built from communes

OSLA

Intersection

```
SELECT nom_dept
FROM dept
WHERE ST_Intersects(the_geom,
  (SELECT ST_Buffer(the_geom, 2500)
   FROM dept WHERE code_dept='75')
);
```

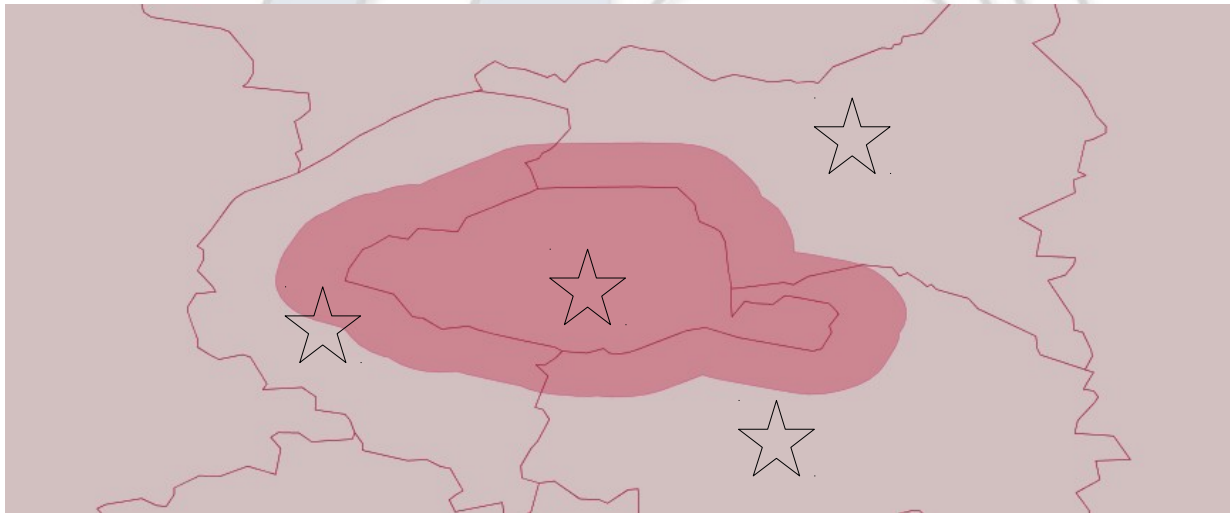
Results :

PARIS

HAUTS-DE-SEINE

SEINE-SAINT-DENIS

VAL-DE-MARNE



OSLANDIA

Distance

```
SELECT code_dept, round(
  ST_Distance(ST_Centroid(the_geom),
  (SELECT ST_Centroid(the_geom)
   FROM dept WHERE code_dept='75')) / 1000)
  AS distance
FROM dept ORDER BY distance LIMIT 4;
```

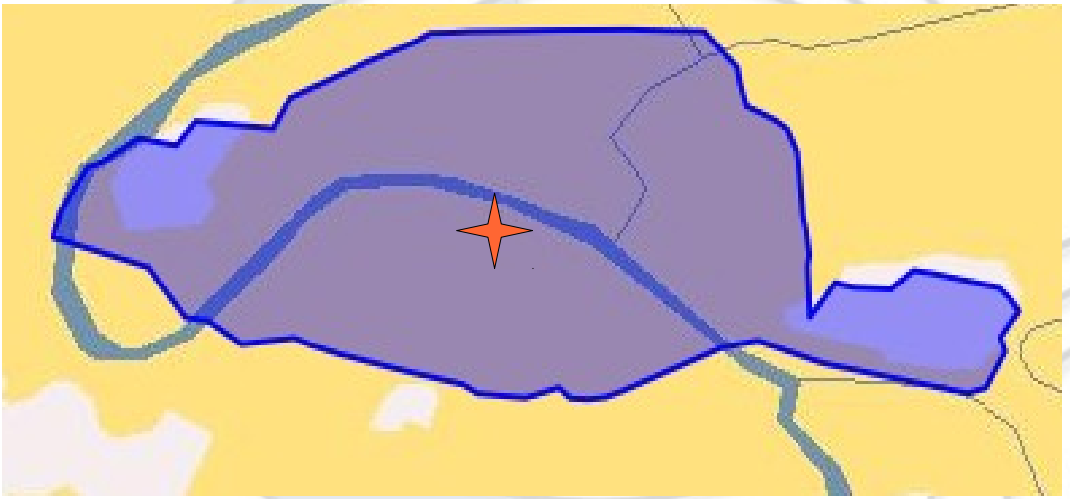
Results:

```
75|0
92|7
93|12
94|13
```

Creation

```
SELECT nom_dept
FROM dept
WHERE St_Within(
   GeometryFromText('POINT(600440 2428685)'
   , 27572), the_geom);
```

Result : PARIS



OSLANDIA

Import GML

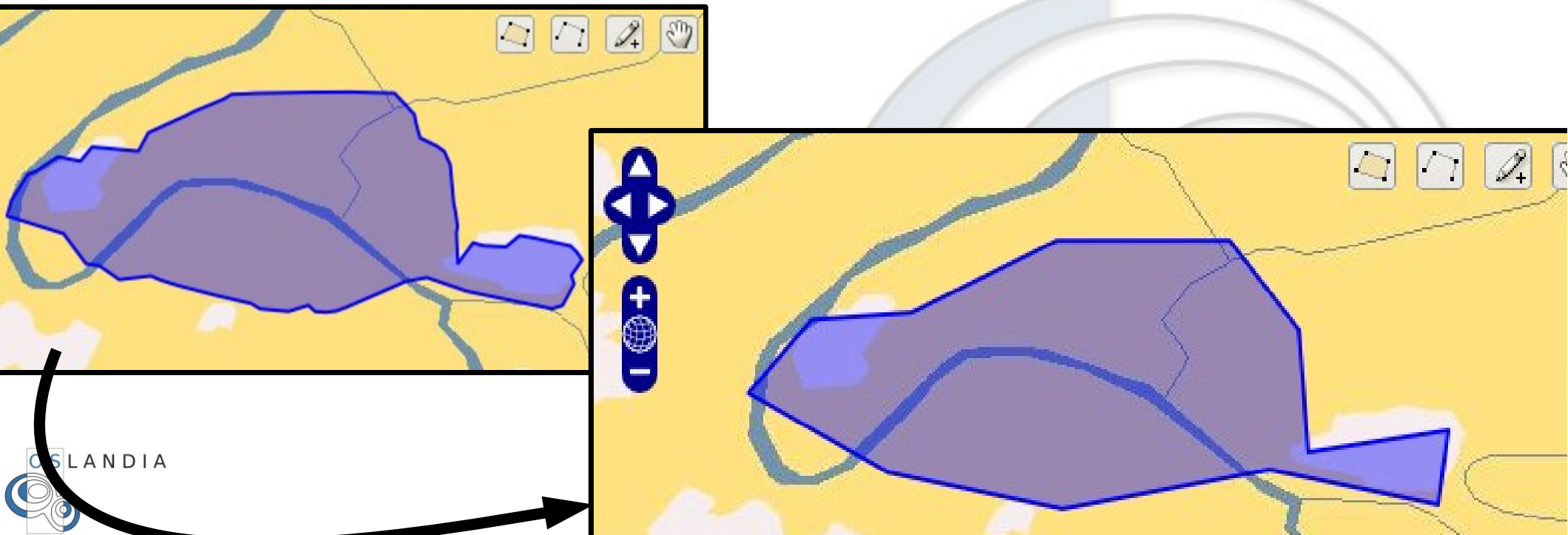
```
SELECT ST_AsText(
        ST_GeomFromGML(
            '<gml:Point srsName="EPSG:27572">
               <gml:pos srsDimension="2">
                  600440 2428686
               </gml:pos>
            </gml:Point>'
        )
    );
```

OSLANDIA

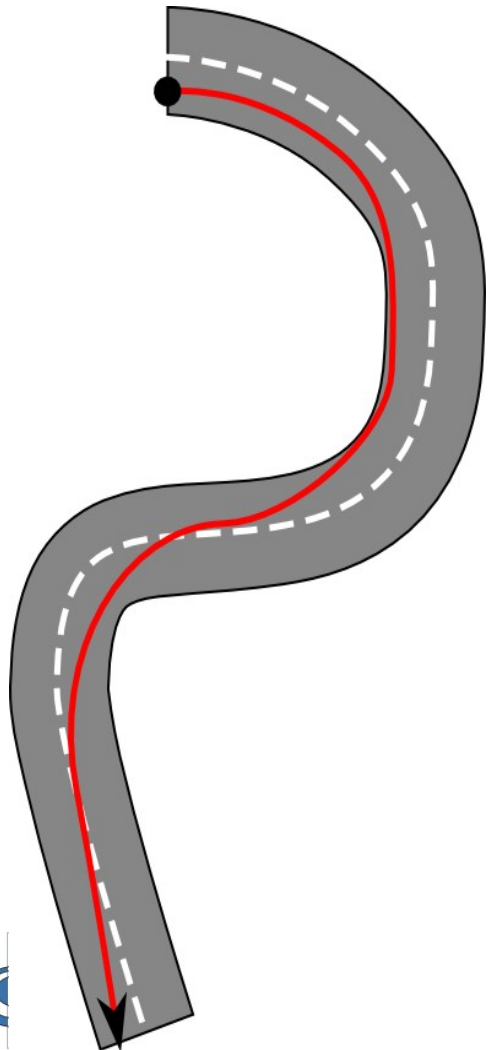
Generalization

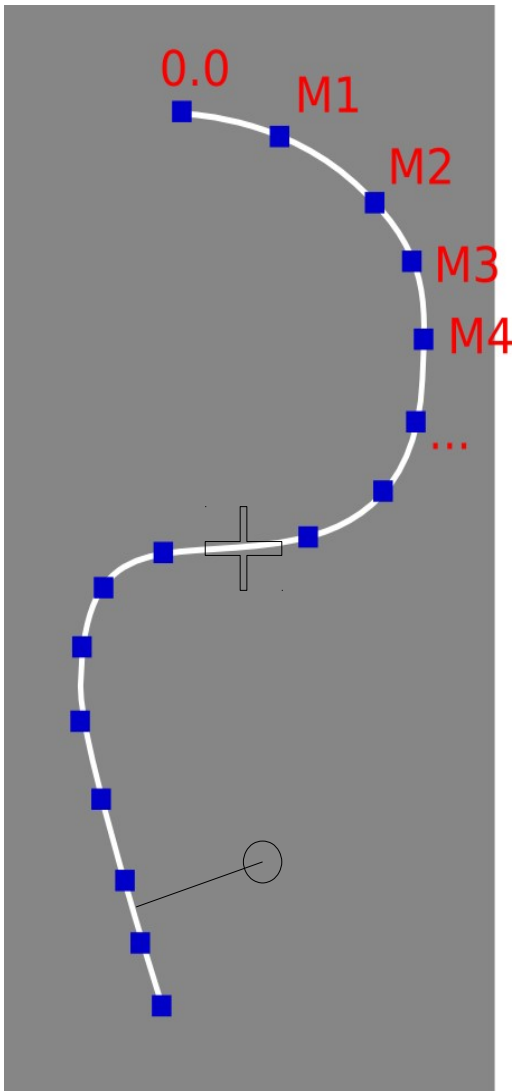
Algorithm : Douglas-Peuker

```
SELECT ST_AsGeoJSON(
            ST_Transform(
                ST_Simplify(the_geom, 800),
            4326), 5)
FROM dept WHERE code_dept='75';
```



Linear referencing





Functions for linear referencing (Road network for example)

Aka LRS

ST_line_interpolate_point(linestring, location)

ST_line_substring(linestring, start, end)

ST_line_locate_point(LineString, Point)

ST_locate_along_measure(geometry, float8)

PgRouting

PgRouting, additional module for graph routing

OSLANDIA

Gegraphy

Type 'geography' : latitude, longitude

= «geodetic support»

Functions for this type

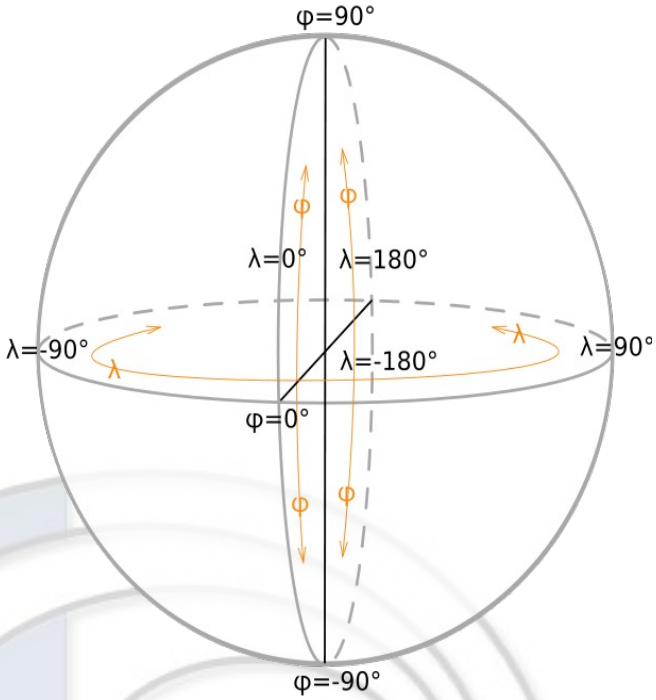
Area, distance, indexation

Only a subset

Import and export functions

GML, KML, GeoJSON

Use it or not according to use case



OSLANDIA