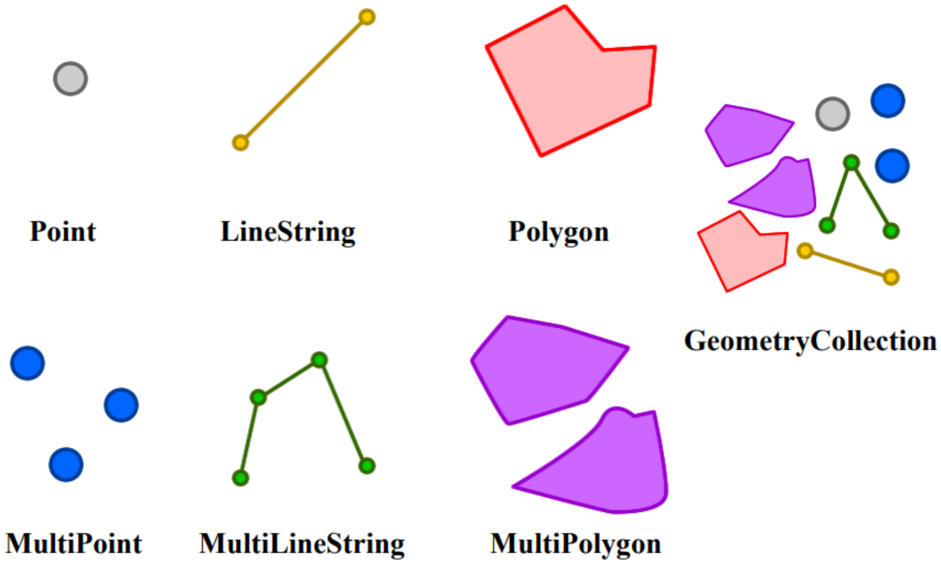
Manipuler la geographie

- Geometry
- Operateurs
- Encodage
- Outils / Bibliotheques
- Indexation / Sharding
- Utilisation OSM

Travailler avec les geométries

- Modèles :
    - Points
    - Ligne (multi)
    - Polygones (multi)
- Coordonnées X/Y/Z - M
- OGC - ST_Geometry - http://www.opengeospatial.org/

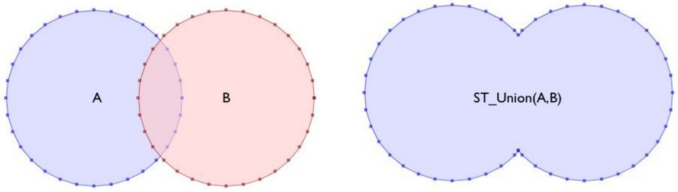


Problématiques associées à la donnée géographique

- Système de coordonnées (WGS 84/ WebMercator) - CRS/SRS
  - Bibliotheques de reprojection
  - Calcul de distances / Précision de la donnée
- Computational Geometry / Précision des nombres
  - Implémentation des opérateurs geometriques
- Taille en octets des geometries
  - contour de région précis -> 100 ko
- Indexation / Volumes
  - accès "Random"

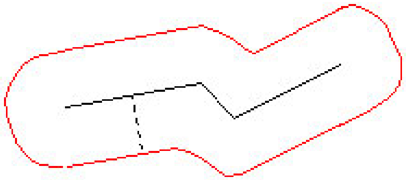
Operateurs

- Tests booleen
  - intersection, touche, à l'intérieur, ont des frontières communes, …
- Modifications / Création
  - union / différence / intersection

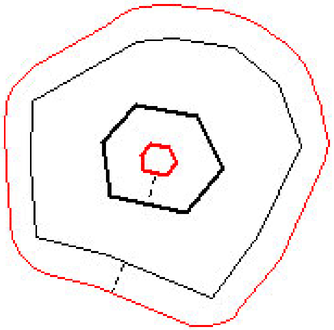


Operateurs Vectoriels (2)

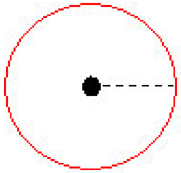
- Buffer / Delaunay / Voronoi …



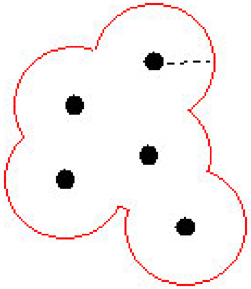
Buffering a Linestring



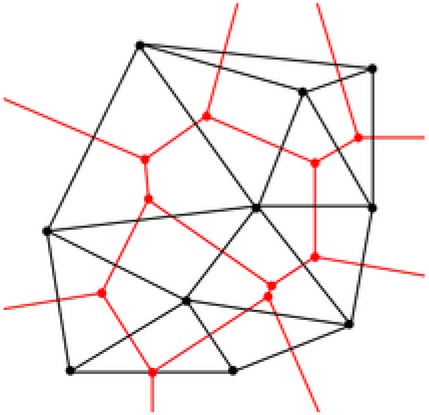
Buffering a Polygon with
1 interior ring



Buffering a Point

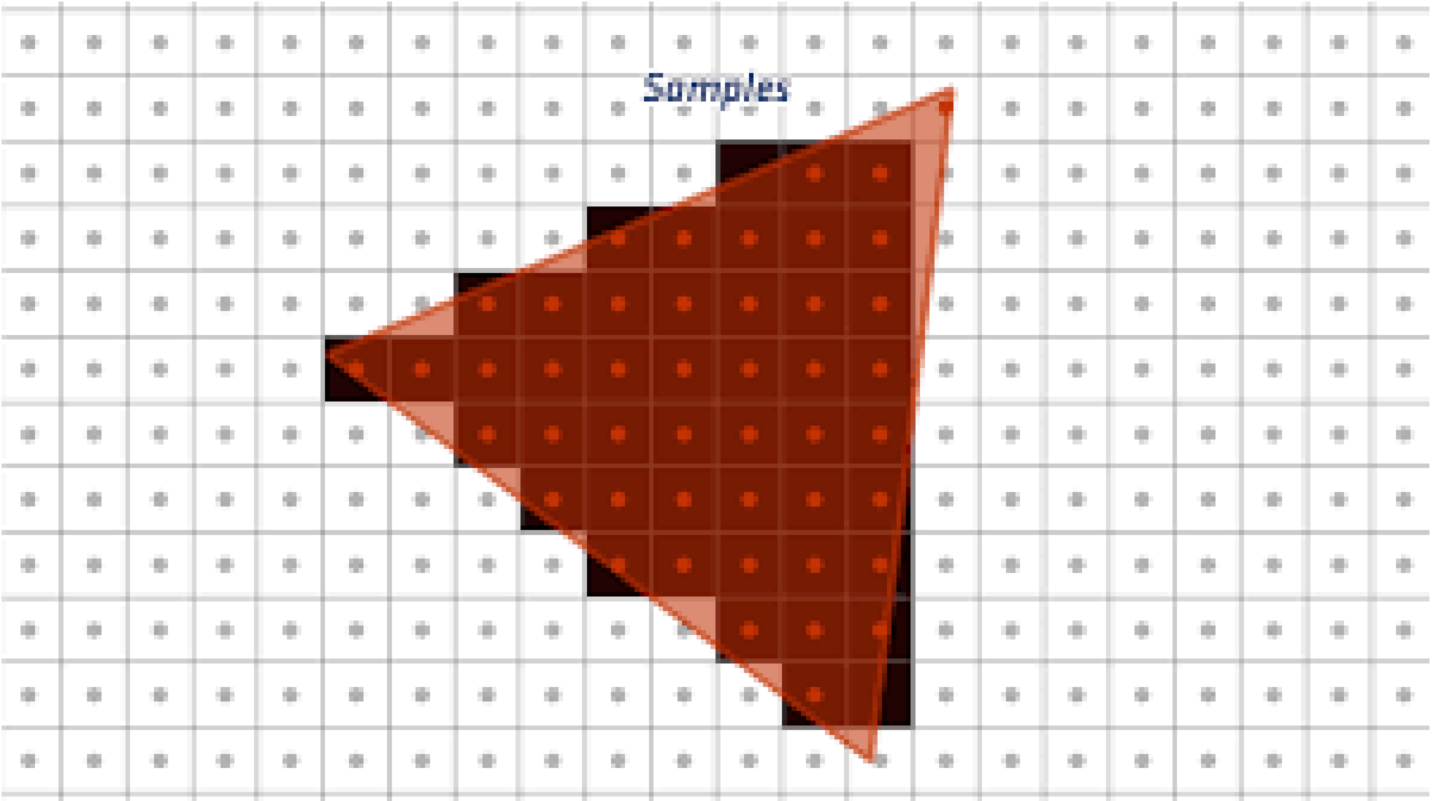


Buffering a Multipoint



Operateurs Raster (3)

- Rasterisation -> Matrices



Exemples - Operateurs ST_* OGC

Constructors	Operations	Relationship Tests	Accessors
ST_AsBinary ST_AsGeoJSON ST_AsJSON ST_AsShape ST_AsText - Notes on Hive usage of ST_AsText ST_GeomCollection ST_GeomFromGeoJSON ST_GeomFromJSON ST_GeomFromShape ST_GeomFromText ST_GeomFromWKB ST_GeometryType ST_LineFromWKB ST_LineString ST_MLineFromWKB ST_MPointFromWKB ST_MPolyFromWKB ST_MultiLineString ST_MultiPoint ST_MultiPolygon ST_Point ST_PointFromWKB ST_PointZ ST_PolyFromWKB ST_Polygon ST_SetSRID	ST_Aggr_ConvexHull ST_Aggr_Intersection ST_Aggr_Union ST_Bin ST_BinEnvelope ST_Boundary ST_Buffer ST_ConvexHull ST_Difference ST_Envelope ST_ExteriorRing ST_InteriorRingN ST_Intersection ST_SymmetricDiff ST_Union	ST_Contains ST_Crosses ST_Disjoint ST_EnvIntersects ST_Equals ST_Intersects ST_Intersects ST_Overlaps ST_Relate ST_Touches ST_Within	ST_Area ST_Centroid ST_CoordDim ST_Dimension ST_Distance ST_GeodesicLengthWGS84 ST_GeometryN ST_Is3D ST_IsClosed ST_IsEmpty ST_IsMeasured ST_IsSimple ST_Length ST_M ST_MaxM ST_MaxX ST_MaxY ST_MaxZ ST_MinM ST_MinX ST_MinY ST_MinZ ST_NumGeometries ST_NumInteriorRing ST_NumPoints ST_PointN - Notes on Hive usage of ST_PointN ST_IsRing ST_SRID ST_StartPoint ST_EndPoint ST_X ST_Y ST_Z

Encodages des Geometries - Stockage

- OGC (WKT / WKB) (Texte ou Binaire)
- GeoJSON (JSON)
- GML (XML)
- KML (XML)
- Shape (Binaire)

Important dans les applications Big Data -> volume, specification

WKT (Well known Text)

```
POINT(6 10)
LINESTRING(3 4,10 50,20 25)
POLYGON((1 1,5 1,5 5,1 5,1 1))
MULTIPOINT((3.5 5.6), (4.8 10.5))
MULTILINESTRING((3 4,10 50,20 25),(-5 -8,-10 -8,-15 -4))
MULTIPOLYGON(((1 1,5 1,5 5,1 5,1 1),(2 2,2 3,3 3,3 2,2 2)),((6
```

WKB (Well Known Binary)

```
POINT(2.0 4.0) is represented as:

00 00000001 4000000000000000 4010000000000000,

where:
1-byte integer 00 or 0: big endian
4-byte integer 00000001 or 1: POINT (2D)
8-byte float 4000000000000000 or 2.0: x-coordinate
8-byte float 4010000000000000 or 4.0: y-coordinate
```

GeoJSON

Utilisation Web essentiellement - https://tools.ietf.org/html/rfc7946

```
{
  "type": "Feature",
  "geometry": {
    "type": "Point",
    "coordinates": [125.6, 10.1]
  },
  "properties": {
    "name": "Dinagat Islands"
  }
}
```

Chez ESRI -> FeatureSet

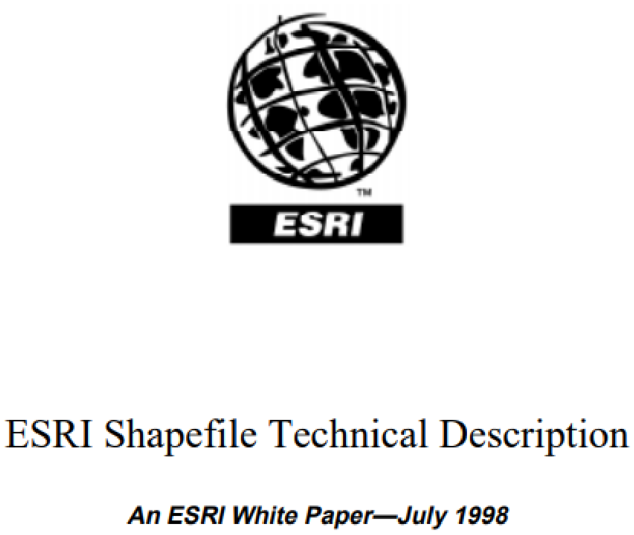
GUI : http://geojson.io

Shape - ESRI

Format binaire

https://www.esri.com/library/whitepapers/pdfs/shapefile.pdf

Equivalent au WKB, plus ancien, utilisé dans le format "ShapeFile" .shp - très connu.



ESRI Shapefile Technical Description

An ESRI White Paper—July 1998

Bibliotheques Geometriques

- OGR / GDAL (C++ / Wrappers)
- ESRI-Geometry-API (JVM)
- JTS (JVM)
- GEOS (C++)
- CGAL (C++)

…

Manipulation geometries (1/3)

Python - OGR

```
from osgeo import ogr

wkt = "LINESTRING (1181866.263593049 615654.4222507705, 1205917
geom = ogr.CreateGeometryFromWkt(wkt)
for i in range(0, geom.GetPointCount()):
    # GetPoint returns a tuple not a Geometry
    pt = geom.GetPoint(i)
    print "%i). POINT (%d %d)" %(i, pt[0], pt[1])
```

https://pcjericks.github.io/py-gdalogr-cookbook/geometry.html

Exemple création buffer (2/3)

```
from osgeo import ogr

wkt = "POINT (1198054.34 648493.09)"
pt = ogr.CreateGeometryFromWkt(wkt)
bufferDistance = 500
poly = pt.Buffer(bufferDistance)
print "%s buffered by %d is %s" % (pt.ExportToWkt(), \
        bufferDistance, poly.ExportToWkt())
```

https://pcjericks.github.io/py-gdalogr-cookbook/geometry.html

Exemple JVM (java)

```
// geometry is shape binary encoded
byte[] content = way.getAs("geometry");
System.out.println(Arrays.asList(content));

Geometry g = GeometryEngine.geometryFromEsriShape(content,
                Type.Polyline);

// we can work on geometries, compute buffers, rasterize  ...

// dump geojson geometry
System.out.println(GeometryEngine.geometryToGeoJson(g));
```

https://github.com/Esri/geometry-api-java

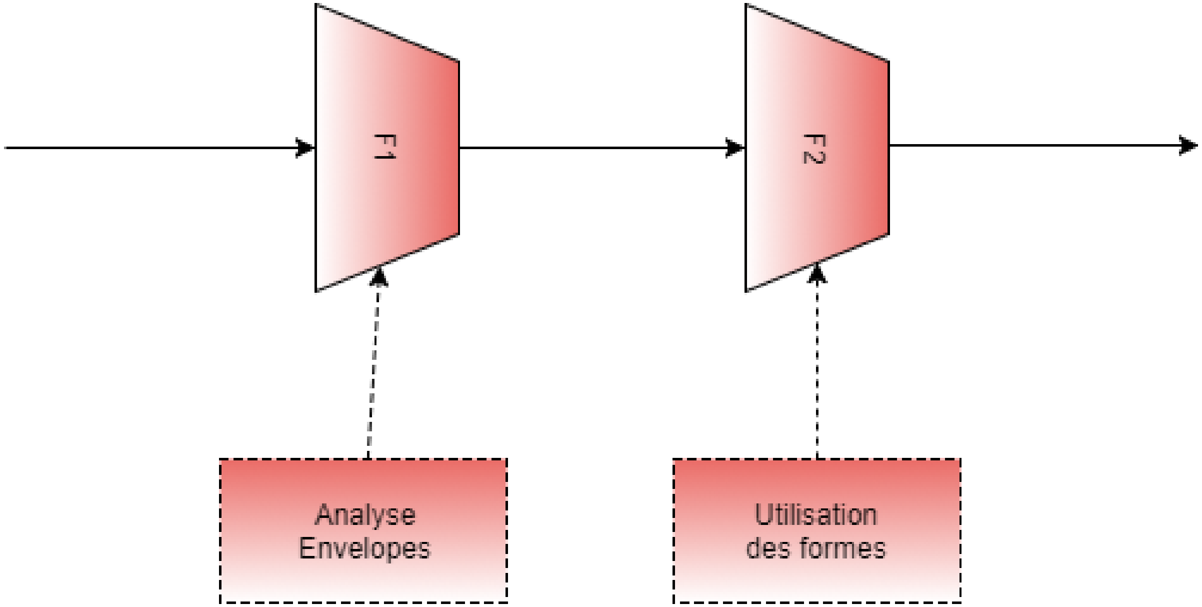
```
<dependency>
  <groupId>com.esri.geometry</groupId>
  <artifactId>esri-geometry-api</artifactId>
  <version>1.2.1</version>
</dependency>
```

Indexation / Sharding / Jointures

Retrouver rapidement des informations par "proximité" spatiale.

L'analyse des geometries est couteuse (CPU), il faut analyser tous les points.

Filtre primaire / secondaire



F1 : Utilisation des BBox (Bouding Box) -> 4 tests flottants

F2 : Utilisation de la geometrie

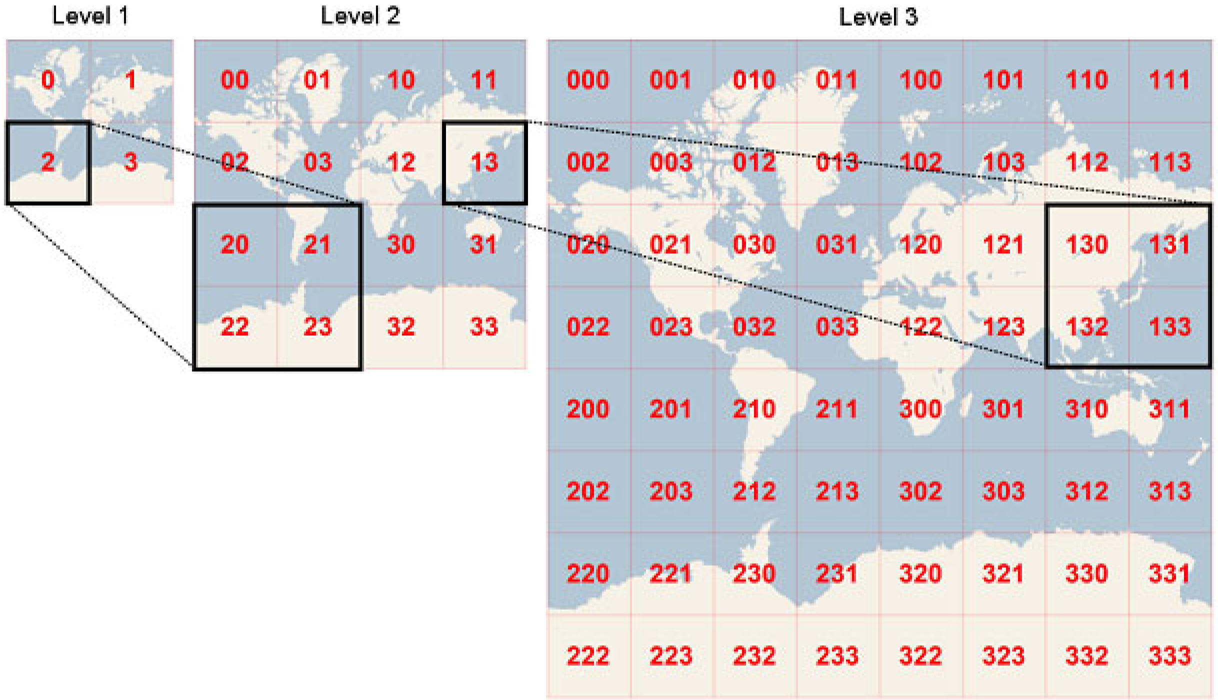
Formes d'indexes

- Grid
- GeoHash - DGGS (OGC)
- QuadTree - RTree
- TwinIndices - Format3
- ...

plus d'infos

https://en.wikipedia.org/wiki/Spatial_database

Focus on GeoHash



Exemple - Scala https://github.com/davidallsopp/geohash-scala

```
import com.github.davidallsopp.geohash.GeoHash._

encode(12.345, 123.456)  // latitude, longitude

    // res0: String = wdpy1r3fv6c9  // using default precision

encode(lat=12.345, lon=123.456)  // named arguments to avoid mi

    // res1: String = wdpy1r3fv6c9  // using default precision

encode(12.345, 123.456, 6)  // specify precision of 6 chars

    // res2: String = wdpy1r

decode("wdpy1r3fv6c9")

    // res2: (Double, Double) = (12.345000011846423,123.4559998
```

En environnement BigData

Big Data -> Parallelisme des traitements, "Bring Program on Datas", Stream the datas

Critères importants :

- Temps de parsing - CPU (Encoding)
- Taille des éléments stockés - I/O performance
- Volumes / Indexation - F1/F2

Back to OSM

Structuration des données OSM

- Node (1)

```
<node id="298884269" lat="54.0901746" lon="12.2482632"
  user="SvenHRO" uid="46882" visible="true" version="1"
  changeset="676636" timestamp="2008-09-21T21:37:45Z"/>
```

- Way (2)

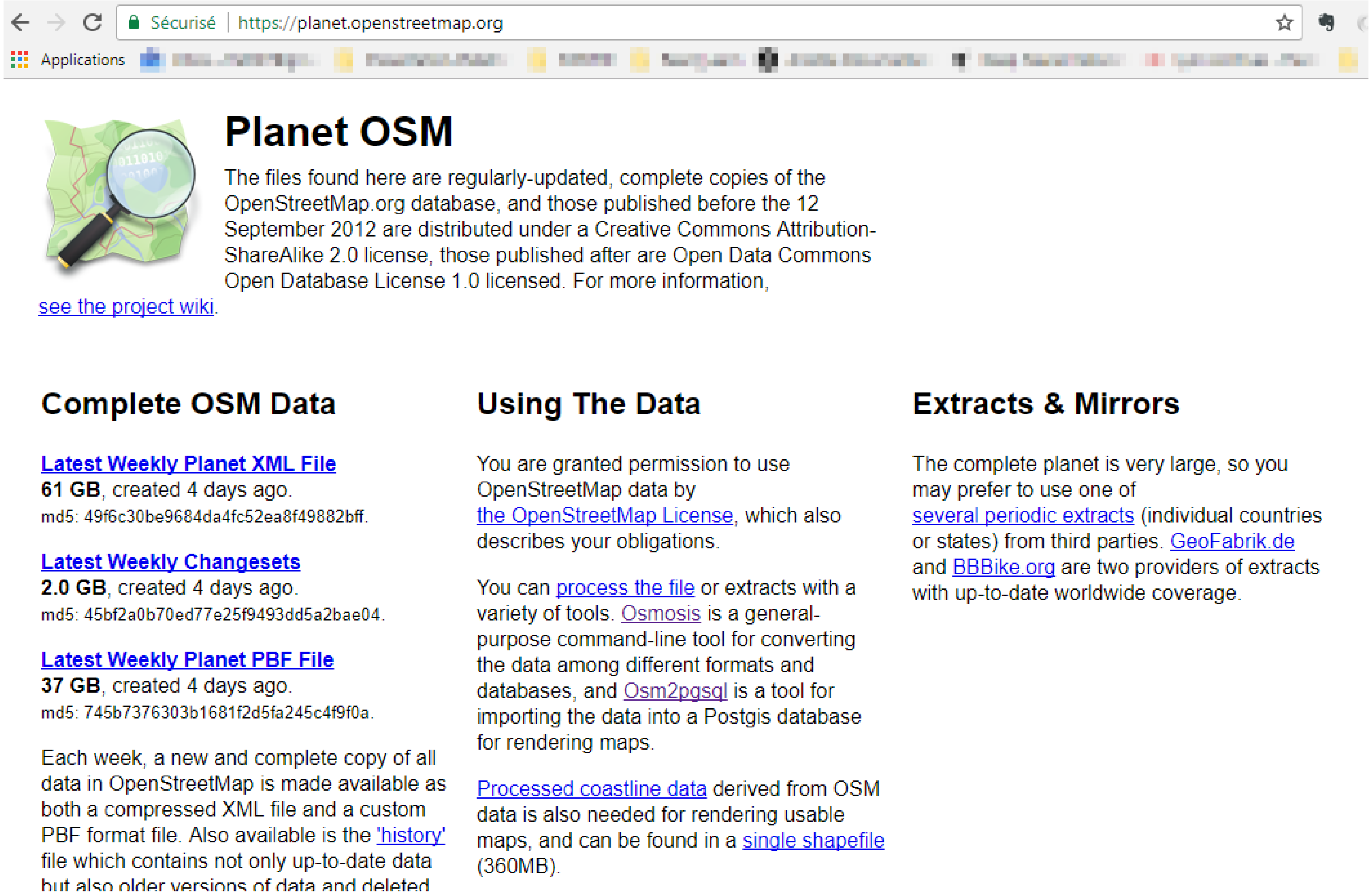
```
<way id="26659127" user="Masch" uid="55988" visible="true"
        version="5" changeset="4142606"
    timestamp="2010-03-16T11:47:08Z">
  <nd ref="292403538"/>
  <nd ref="298884289"/>
  ...
  <nd ref="261728686"/>
  <tag k="highway" v="unclassified"/>
  <tag k="name" v="Pastower Straße"/>
 </way>
```

- Relation (3)

```
<relation id="56688" user="kmvar" uid="56190" visible="true"
  version="28" changeset="6947637"
  timestamp="2011-01-12T14:23:49Z">
 <member type="node" ref="294942404" role=""/>
 ...
 <member type="node" ref="364933006" role=""/>
 <member type="way" ref="4579143" role=""/>
 ...
 <member type="node" ref="249673494" role=""/>
 <tag k="name" v="Küstenbus Linie 123"/>
 <tag k="network" v="VVW"/>
 <tag k="operator" v="Regionalverkehr Küste"/>
 <tag k="ref" v="123"/>
 <tag k="route" v="bus"/>
 <tag k="type" v="route"/>
</relation>
```

Formats de Téléchargements

- XML -> format d'origine
- PBF -> protocol buffer - Très compact



Planet OSM

The files found here are regularly-updated, complete copies of the OpenStreetMap.org database, and those published before the 12 September 2012 are distributed under a Creative Commons Attribution-ShareAlike 2.0 license, those published after are Open Data Commons Open Database License 1.0 licensed. For more information, see the project wiki.

Complete OSM Data

Latest Weekly Planet XML File
61 GB, created 4 days ago.
md5: 49f6c30be9684da4fc52ea8f49882bff.

Latest Weekly Changesets
2.0 GB, created 4 days ago.
md5: 45bf2a0b70ed77e25f9493dd5a2bae04.

Latest Weekly Planet PBF File
37 GB, created 4 days ago.
md5: 745b7376303b1681f2d5fa245c4f9f0a.

Each week, a new and complete copy of all data in OpenStreetMap is made available as both a compressed XML file and a custom PBF format file. Also available is the 'history' file which contains not only up-to-date data but also older versions of data and deleted

Using The Data

You are granted permission to use OpenStreetMap data by the OpenStreetMap License, which also describes your obligations.

You can process the file or extracts with a variety of tools. Osmosis is a general-purpose command-line tool for converting the data among different formats and databases, and Osm2pgsql is a tool for importing the data into a Postgis database for rendering maps.

Processed coastline data derived from OSM data is also needed for rendering usable maps, and can be found in a single shapefile (360MB).

Extracts & Mirrors

The complete planet is very large, so you may prefer to use one of several periodic extracts (individual countries or states) from third parties. GeoFabrik.de and BBBike.org are two providers of extracts with up-to-date worldwide coverage.

Consommation des données OSM

plusieurs chemins

0 - OpenStreetmap.org -> extract XML (petit ensemble)

1 - fichiers -> Postgis / Postgres

2 - à partir de Fichiers uniquement

1 - PostGIS / Postgresql

Outils existants : Osmosis, osm2pgsql, ...

```
osm2pgsql -s -U postgres -d nameofdatabase  \
     /file/path/toosm/fileorpbf/name.osm
```

ex : Planet import on custom E5-1650 (32GB RAM - 6 cores) / SSD

```
Processing: Node(1507455k 301.6k/s) Way(141480k 37.49k/s) Relat
.....
Osm2pgsql took 40016s overall (11.1 hours)

real	666m56.457s
user	335m7.557s
sys	40m0.366s
```

Geofabrik / Openstreetmap.fr propose des extracts plus petits -> economie de temps

2 - à partir de fichiers

Etapes de lecture

- Parsing du fichier
- Reconstruction des geometries (Ways, Polygons)

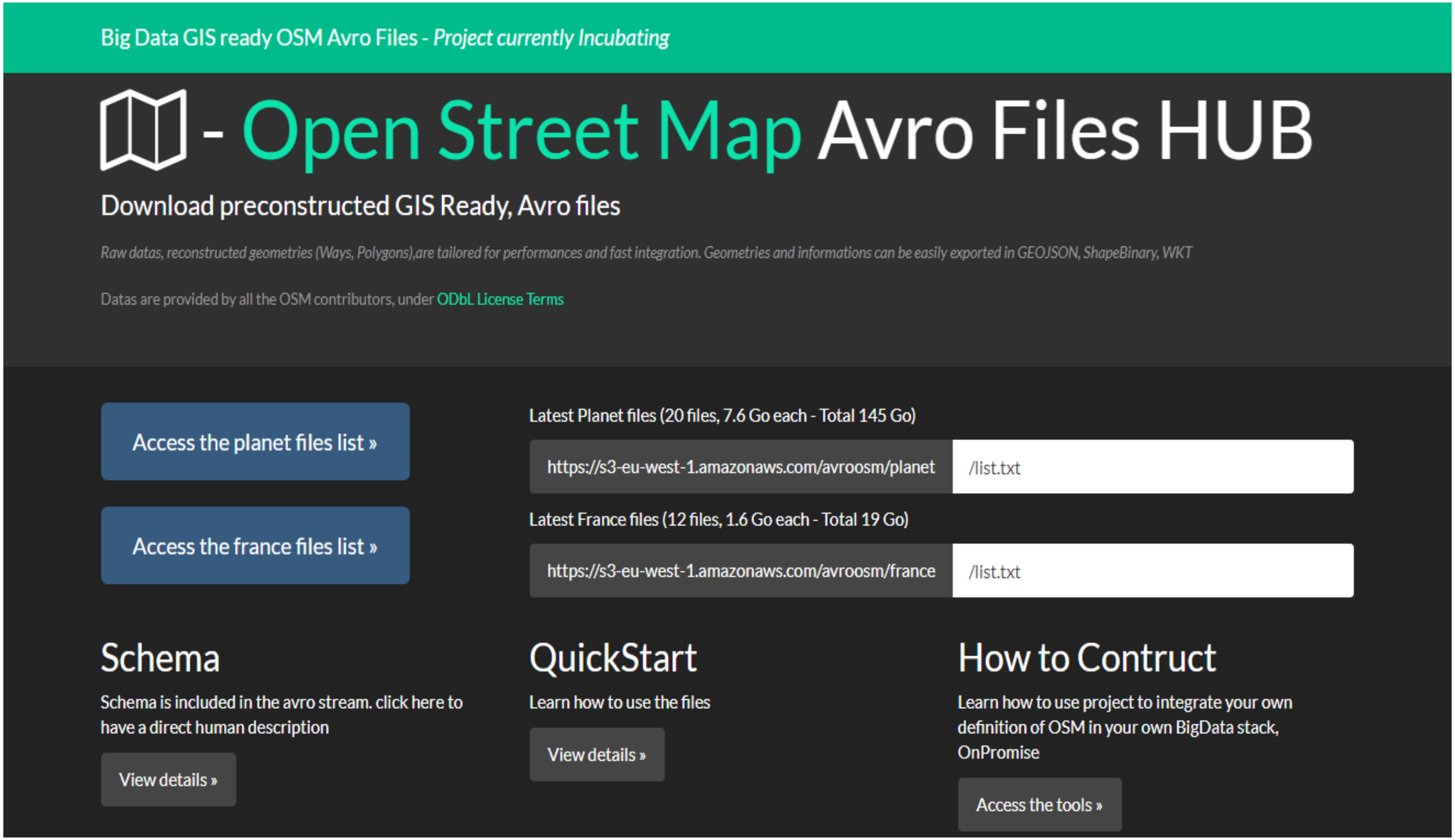
Librairies :

- osm-pbf-parser (Javascript)
- ruby osm parsing
- crosby.binary (JVM)
- imp osm (python)

-> ATTENTION : reconstruction des polylignes / polygones

Autre possibilité - AVRO (Big Data)

http://avroosm.s3-website-eu-west-1.amazonaws.com/index.html
http://bit.ly/2gPIw8B



Big Data GIS ready OSM Avro Files - *Project currently Incubating*

- Open Street Map Avro Files HUB

Download preconstructed GIS Ready, Avro files

Raw datas, reconstructed geometries (Ways, Polygons),are tailored for performances and fast integration. Geometries and informations can be easily exported in GEOJSON, ShapeBinary, WKT

Datas are provided by all the OSM contributors, under ODbL License Terms

Access the planet files list »

Access the france files list »

Latest Planet files (20 files, 7.6 Go each - Total 145 Go)

https://s3-eu-west-1.amazonaws.com/avroosm/planet /list.txt

Latest France files (12 files, 1.6 Go each - Total 19 Go)

https://s3-eu-west-1.amazonaws.com/avroosm/france /list.txt

Schema

Schema is included in the avro stream. click here to have a direct human description

View details »

QuickStart

Learn how to use the files

View details »

How to Contruct

Learn how to use project to integrate your own definition of OSM in your own BigData stack, OnPromise

Access the tools »

Polylignes et polygones déjà préconstruits

Stack Big Data

Librairies / Solutions Geospatiales (BigData)

Magellan - HortonWorks

Geomesa - Location Tech

ESRI-UDF - ESRI

…