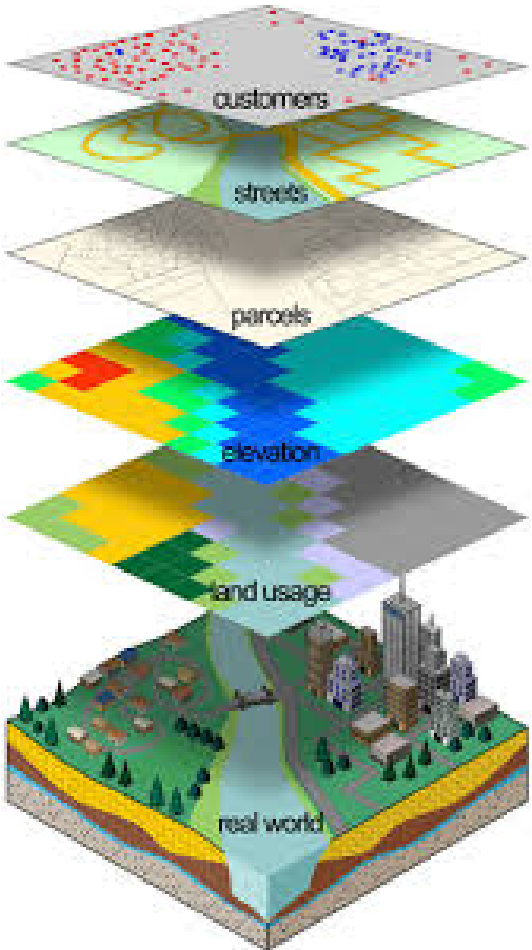
Plus loin avec PostGIS

Plus de spatial !

Vincent Picavet – Oslandia – www.oslandia.com

Post -
G eographical
I nformation
S ystem

Capturer, creér, stocker, analyser, partager, visualiser la donnée relative à l’espace



PostGIS

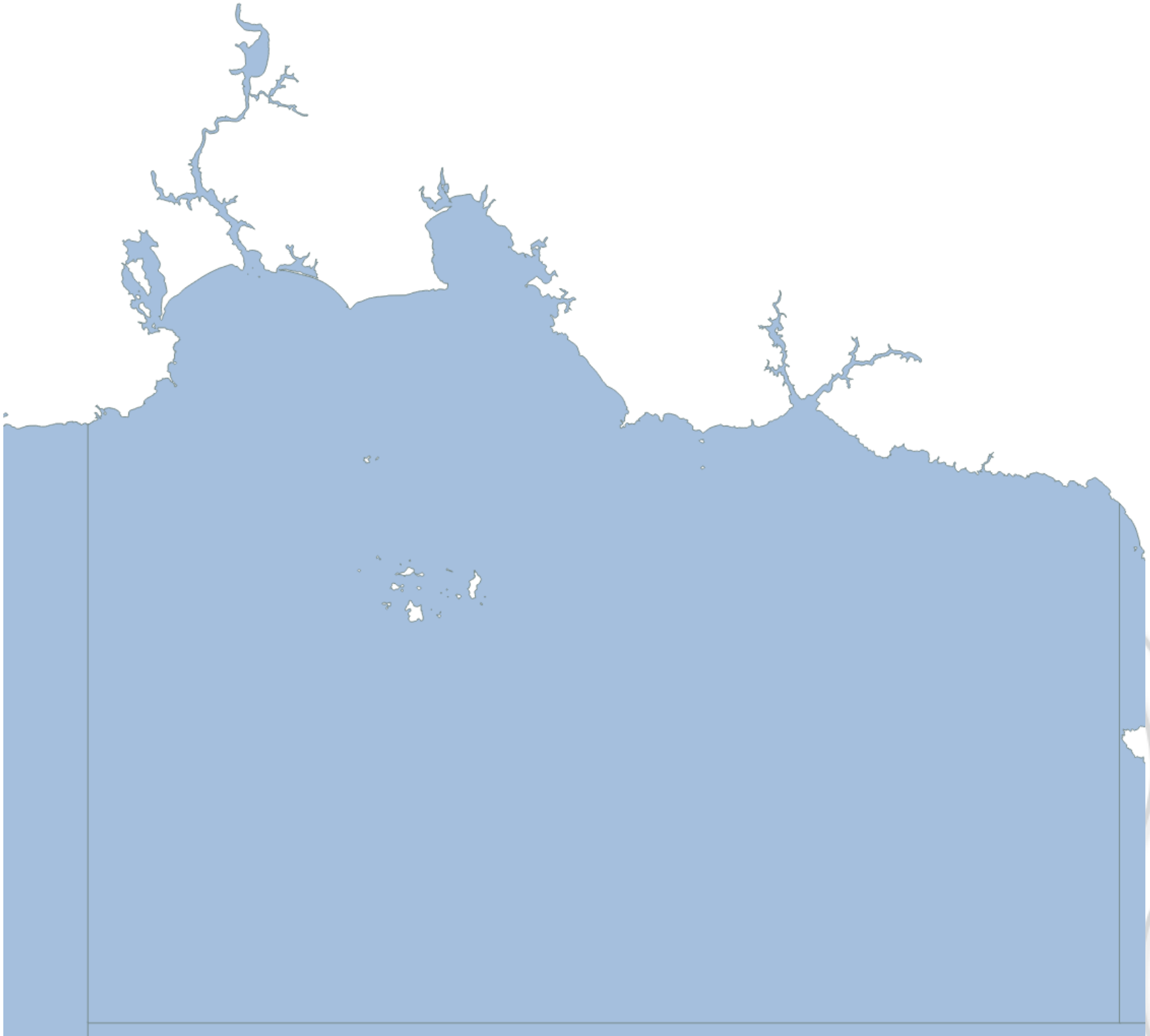
PostGIS 101

- Extension PostgreSQL
- Types : points, lignes, polygones...
- Indexes spatiaux
- Plein de fonctions !

```
SELECT
    d.*, count(*) / ST_Area(d.geom)
FROM
    clients as c
JOIN
    districts as d
ON
    ST_Contains(d.geom, c.geom)
GROUP by
    d.gid;
```

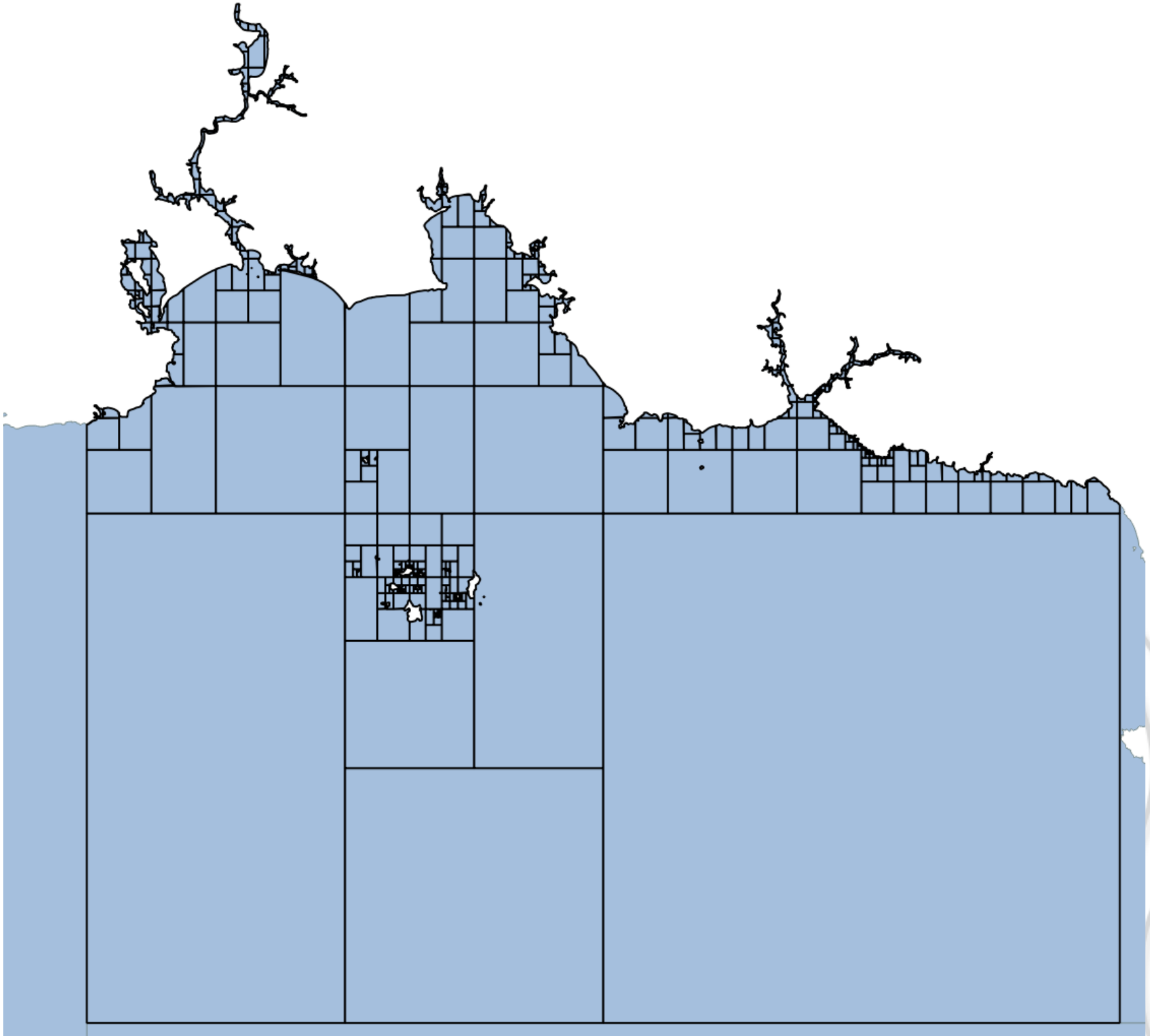
PostGIS 2.2.2

- Sortie le 22/03/2016
- Installeur Windows avec :
  - PostGIS 2.2, PgRouting 2.1.0, SFCGAL, OGR FDW, pgPointCloud...
- Nouvelles Fonctionnalités
  - KNN Recheck, KNN Geography
  - Fonctions 3D
  - Functions temporelles
  - Améliorations raster
  - Autres fonctions (simplify, TWKB…)



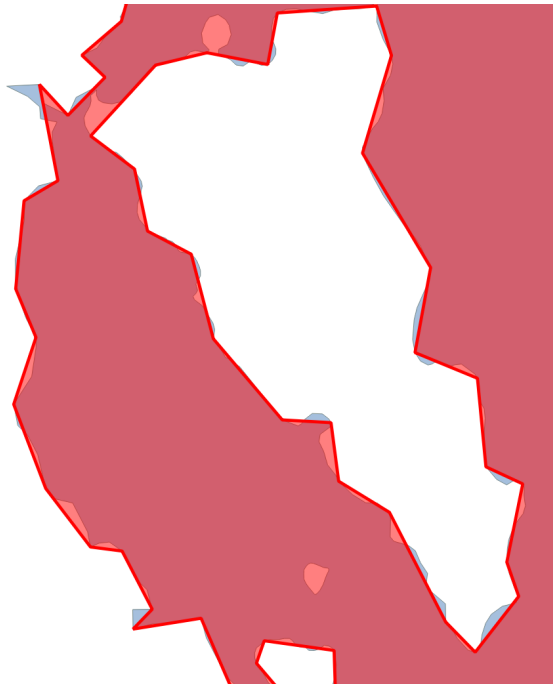
Subdivide

```
SELECT
     ST_Subdivide(geom, 50) as geom
FROM
     water_polygon
WHERE
     gid = 16389;
```



Simplify

```
SELECT
     gid
     , ST_SimplifyVW(geom, 300) as geom
FROM
     water_polygon
WHERE
     gid = 16389;
```



TWKB

- Format binaire
- Tiny Well Known Binary
- Avec IDs

```
SELECT
     ST_AsTWKB(array_agg(geom), array_agg(gid))
FROM
     mytable;
                      st_astwkb
-----------------------------------------------
\x040402020400000202
```

Raster

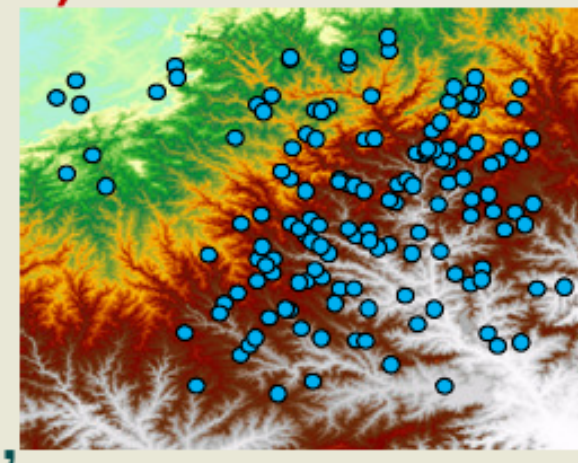
Rasters

- Analyse Raster / vecteur
- Nouveau type de données
  - Ressemble à geometry
  - Mais pour les rasters
- Multirésolution, multibandes, couverture tuilée
- Import/export (GDAL)
- Fonctions
  - Statistiques, reprojection, edition, calculs
  - Fonctions Vector/raster
  - De + en + de fonctions & plus rapides

Rasters

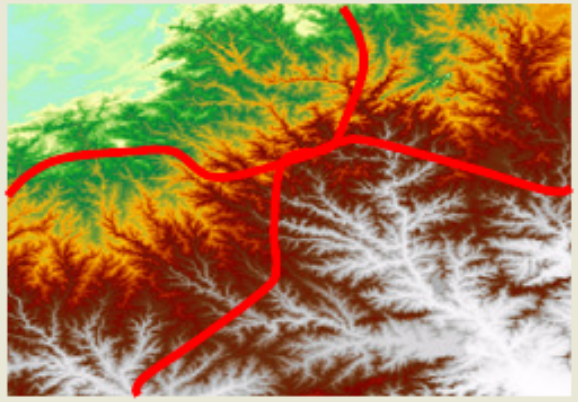
Extract ground elevation values for lidar points…

- SELECT pointID, ST_Value(rast, geom) elevation
  FROM lidar, srtm WHERE ST_Intersects(geom, rast)



Intersect a road network to extract elevation values for each road segment

- SELECT roadID,
  (ST_Intersection(geom, rast)).geom road,
  (ST_Intersection(geom, rast)).val elevation
  FROM roadNetwork, srtm WHERE ST_Intersects(geom, rast)



3D

PostGIS 3D / SFCGAL

SFCGAL

=

CGAL

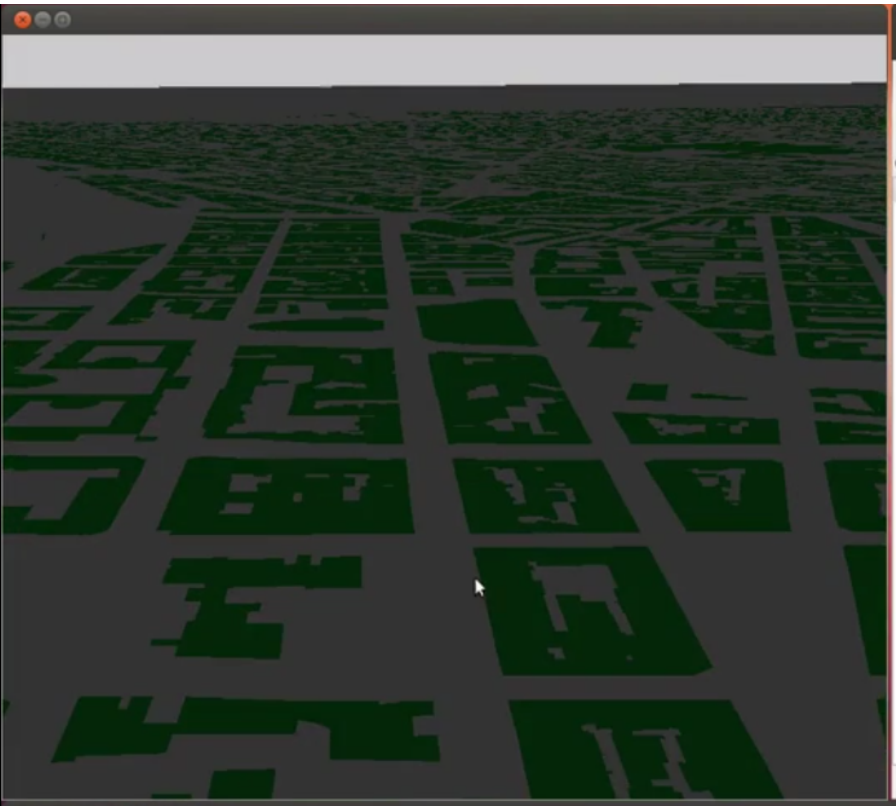
+

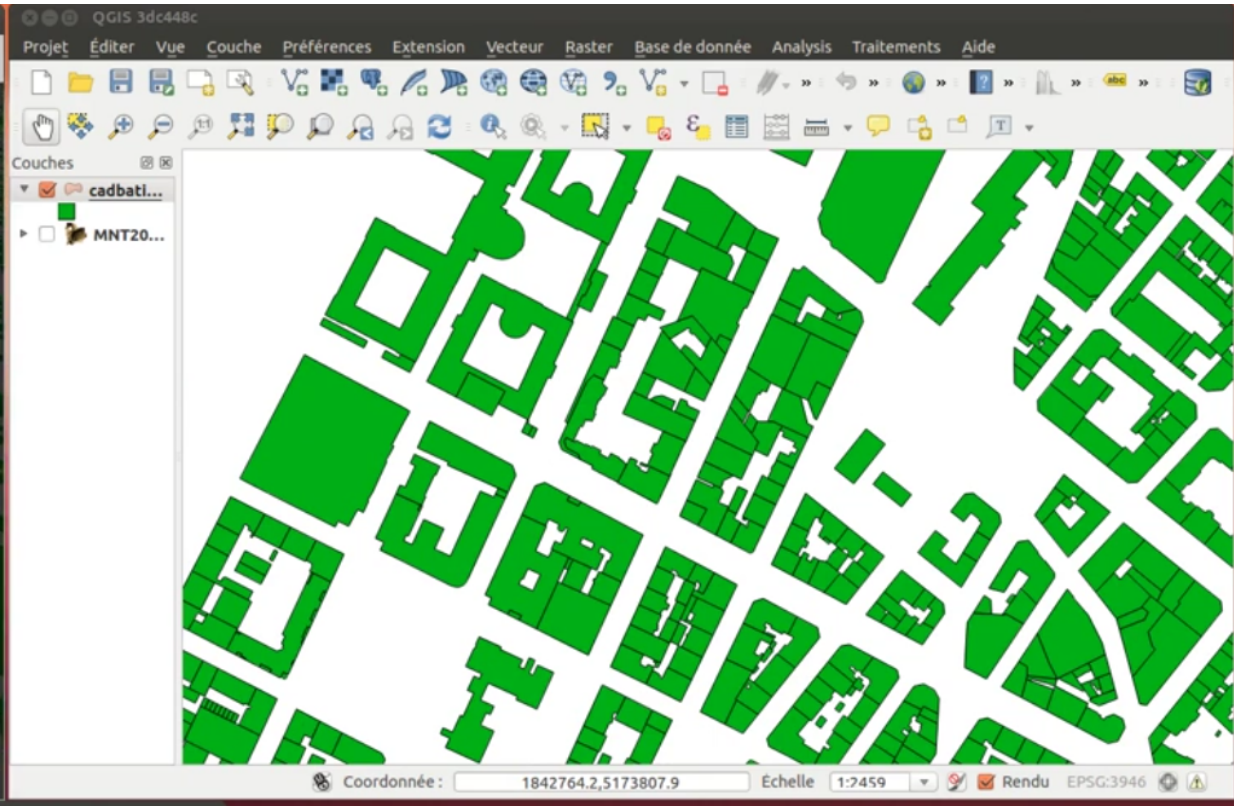
CERTIFIED OGC® COMPLIANT

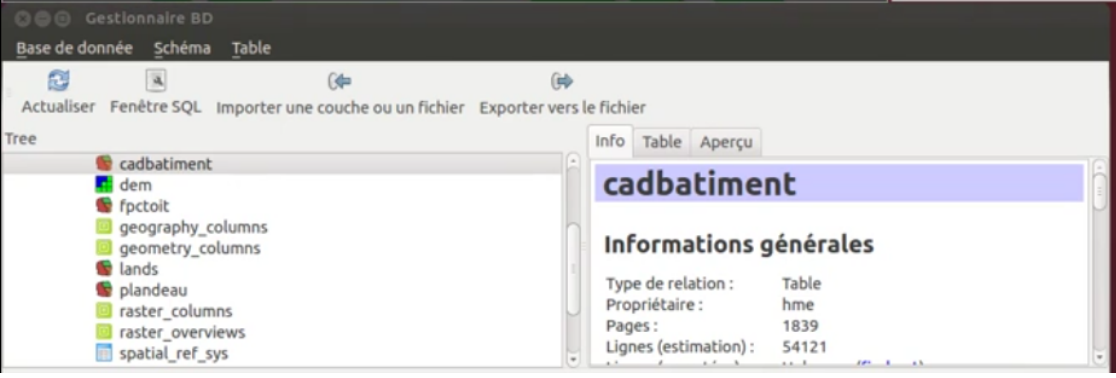
ISO 19107:2013

ISO 19125:2013

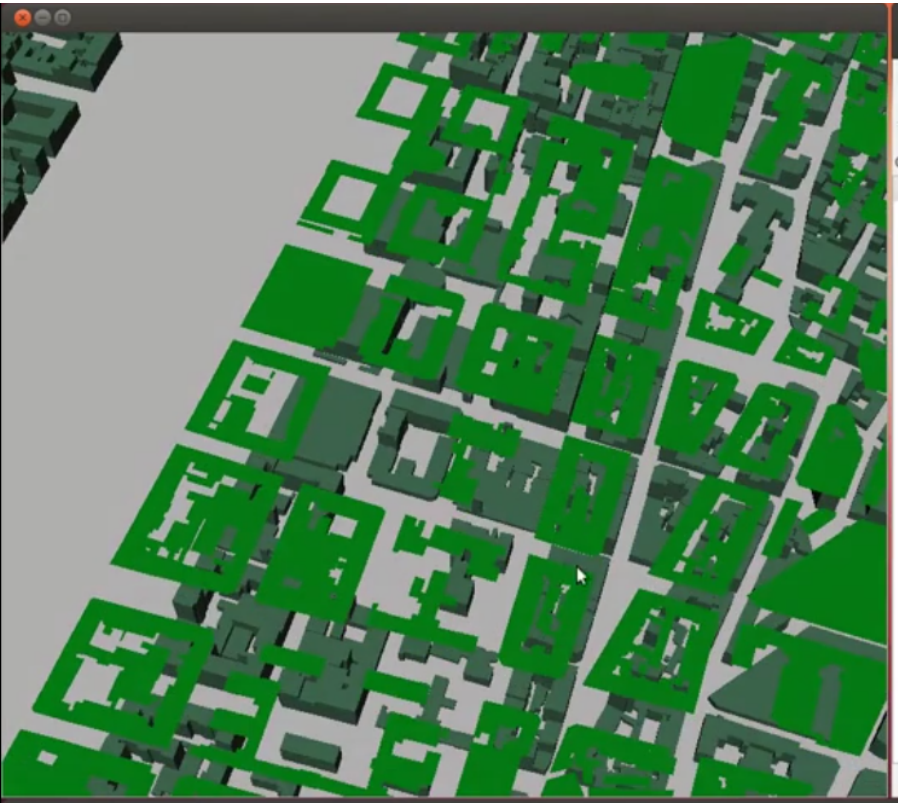
CGAL

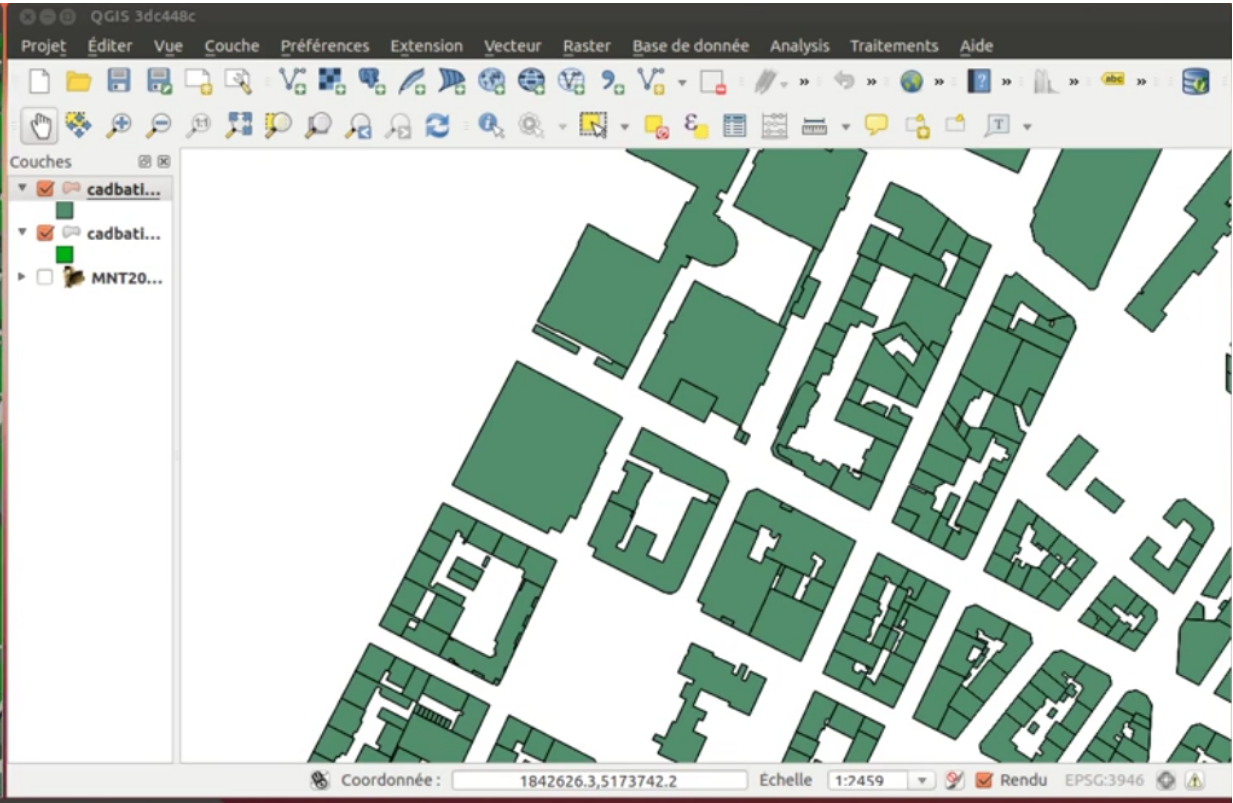


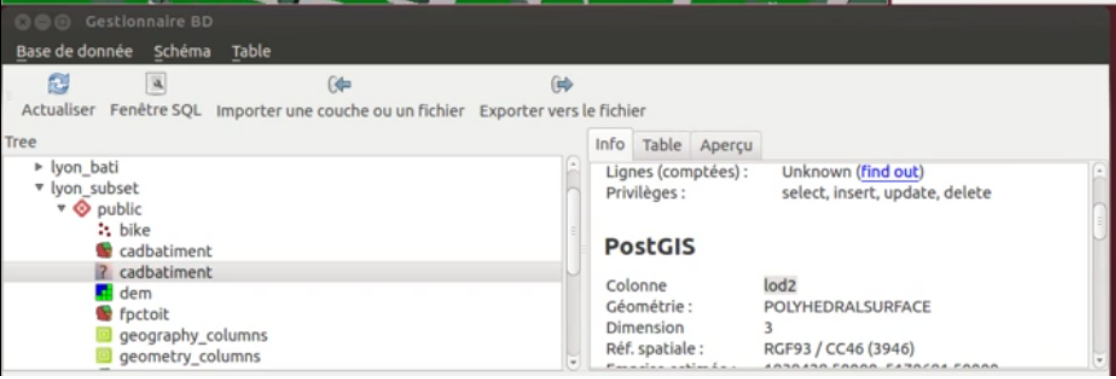

QGIS 3dc448c
Projet Éditer Vue Couche Préférences Extension Vecteur Raster Base de donnée Analysis Traitements Aide
Couches
cadbati...
MNT20...
Coordonnée : 1842764.2,5173807.9
Échelle 1:2459
Rendu
EPSG:3946

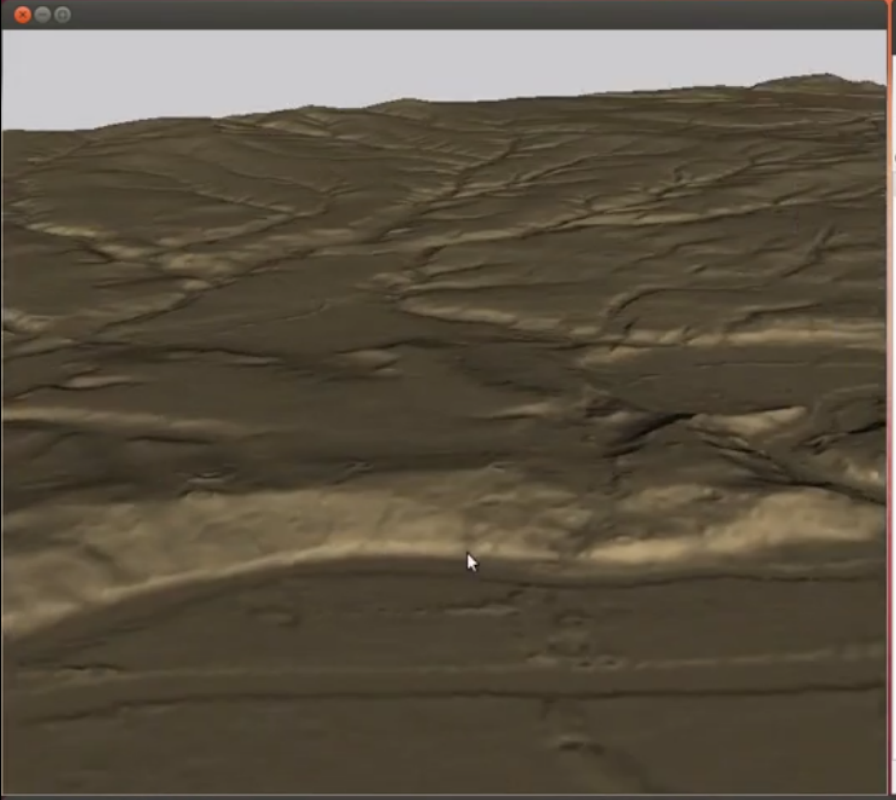

Gestionnaire BD
Base de donnée Schéma Table
Actualiser
Fenêtre SQL
Importer une couche ou un fichier
Exporter vers le fichier
Tree
cadbatiment
dem
fpctoit
geography_columns
geometry_columns
lands
plandeau
raster_columns
raster_overviews
spatial_ref_sys
Info Table Aperçu
cadbatiment
Informations générales
Type de relation : Table
Propriétaire : hme
Pages : 1839
Lignes (estimation) : 54121

```
ALTER TABLE
    buildings
ADD COLUMN
    geom3d geometry('POLYHEDRALSURFACEZ',3946);


UPDATE
    buildings
SET
    geom3d = ST_Extrude(ST_Force2D(geom), 0, 0, height);


-- Create spatial indexes
CREATE INDEX buildings_geom3d_idx
        ON buildings USING GIST(lod2);
```




QGIS 3dc448c
Projet Éditer Vue Couche Préférences Extension Vecteur Raster Base de donnée Analysis Traitements Aide
Couches
cadbati...
cadbati...
MNT20...
Coordonnée : 1842626.3,5173742.2
Échelle 1:2459
Rendu
EPSG:3946


Gestionnaire BD
Base de donnée Schéma Table
Actualiser
Fenêtre SQL
Importer une couche ou un fichier
Exporter vers le fichier
Tree
lyon_bati
lyon_subset
public
bike
cadbatiment
cadbatiment
dem
fpctoit
geography_columns
geometry_columns
Info Table Aperçu
Lignes (comptées) : Unknown (find out)
Privilèges : select, insert, update, delete
PostGIS
Colonne lod2
Géométrie : POLYHEDRALSURFACE
Dimension 3
Réf. spatiale : RGF93 / CC46 (3946)

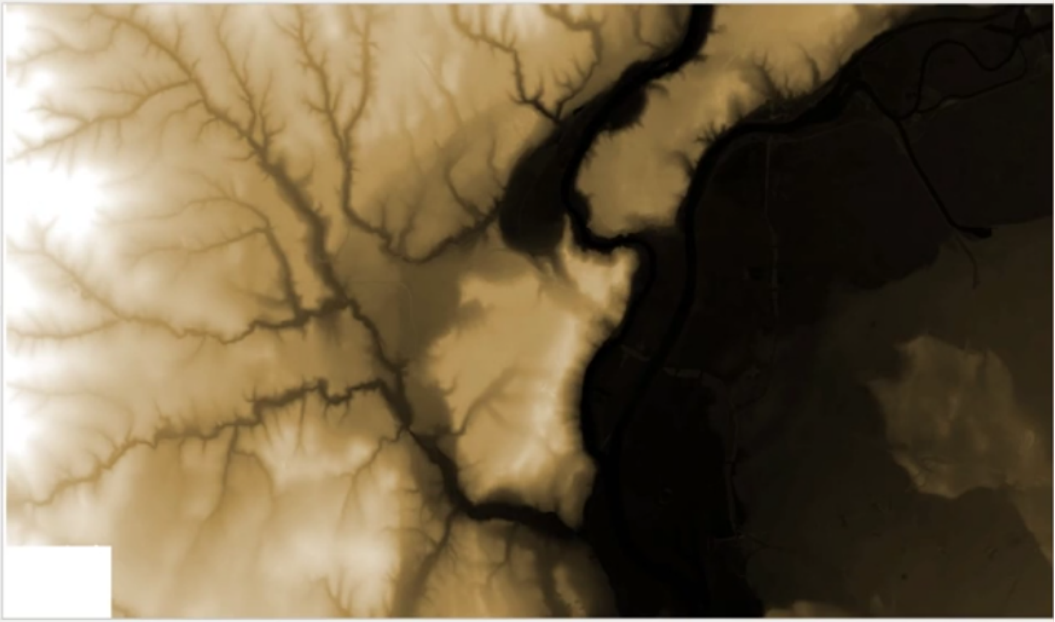


QGIS 3dc448c

Projet Éditer Vue Couche Préférences Extension Vecteur Raster Base de donnée Analysis Traitements Aide

Couches

MNT20...



Coordonnée : 1830932,5172876 Échelle 1:79970 Rendu EPSG:3946

Gestionnaire BD

Base de donnée Schéma Table

Actualiser Fenêtre SQL Importer une couche ou un fichier Exporter vers le fichier

Tree

- PostGIS
  - boston2
  - lyon_bati
  - lyon_subset
    - public
  - paris3d
  - paris_bati
  - tempus_nantes
  - test_db

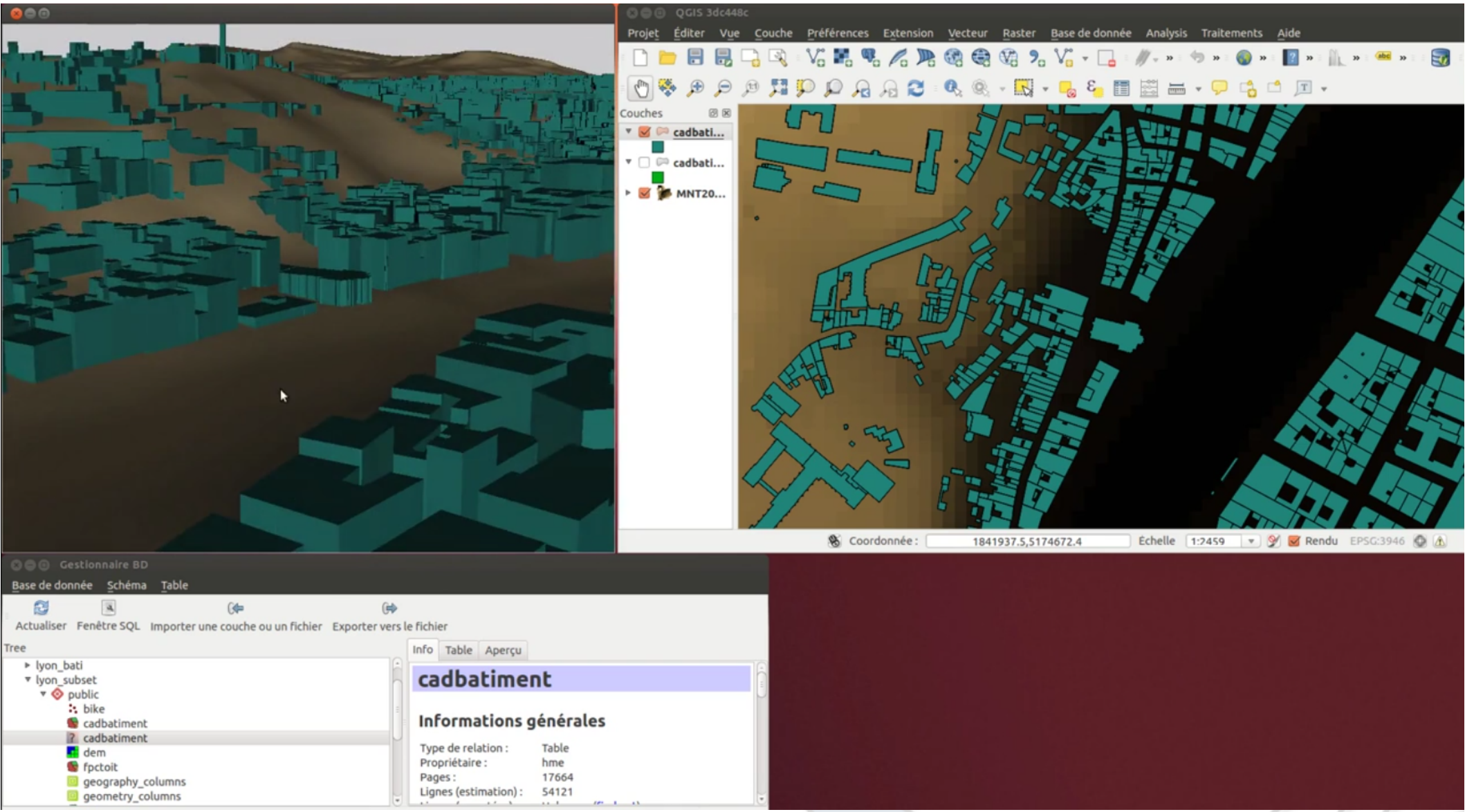
Info Table Aperçu

lyon_subset

Non connecté

```sql
-- Compute altitude for each building
ALTER TABLE buildings ADD COLUMN altitude integer;

 SELECT
     gid,
     -- Get the raster (elevation) value at the
     -- building's centroid point
     st_value(rast, st_centroid(geom)) as elevation
 FROM
     buildings
 ON
     -- Spatial join between raster and geometry
     ST_Intersects(rast, geom)
```

```
WITH hh AS
(
    ...
)
UPDATE
    buildings
SET
    -- Keep the altitude
    altitude = elevation
    -- Translate the buildings to their real heights
    , geom3d = ST_Translate(geom3d, 0, 0, elevation)
FROM
    hh
WHERE
    buildings.gid = hh.gid;
```


QGIS 3dc448c
Projet Éditer Vue Couche Préférences Extension Vecteur Raster Base de donnée Analysis Traitements Aide
Couches
cadbati...
cadbati...
MNT20...
Coordonnée : 1841937.5,5174672.4
Échelle 1:2459
Rendu
EPSG:3946
Gestionnaire BD
Base de donnée Schéma Table
Actualiser
Fenêtre SQL
Importer une couche ou un fichier
Exporter vers le fichier
Tree
lyon_bati
lyon_subset
public
bike
cadbatiment
cadbatiment
dem
fpctoit
geography_columns
geometry_columns
Info Table Aperçu
cadbatiment
Informations générales
Type de relation : Table
Propriétaire : hme
Pages : 17664
Lignes (estimation) : 54121

Viewer : Horao

- OpenSceneGraph-based
- Independant + QGIS plugin
- Synchronization with QGIS
- MNT, 3D, custom queries
- www.horao.net

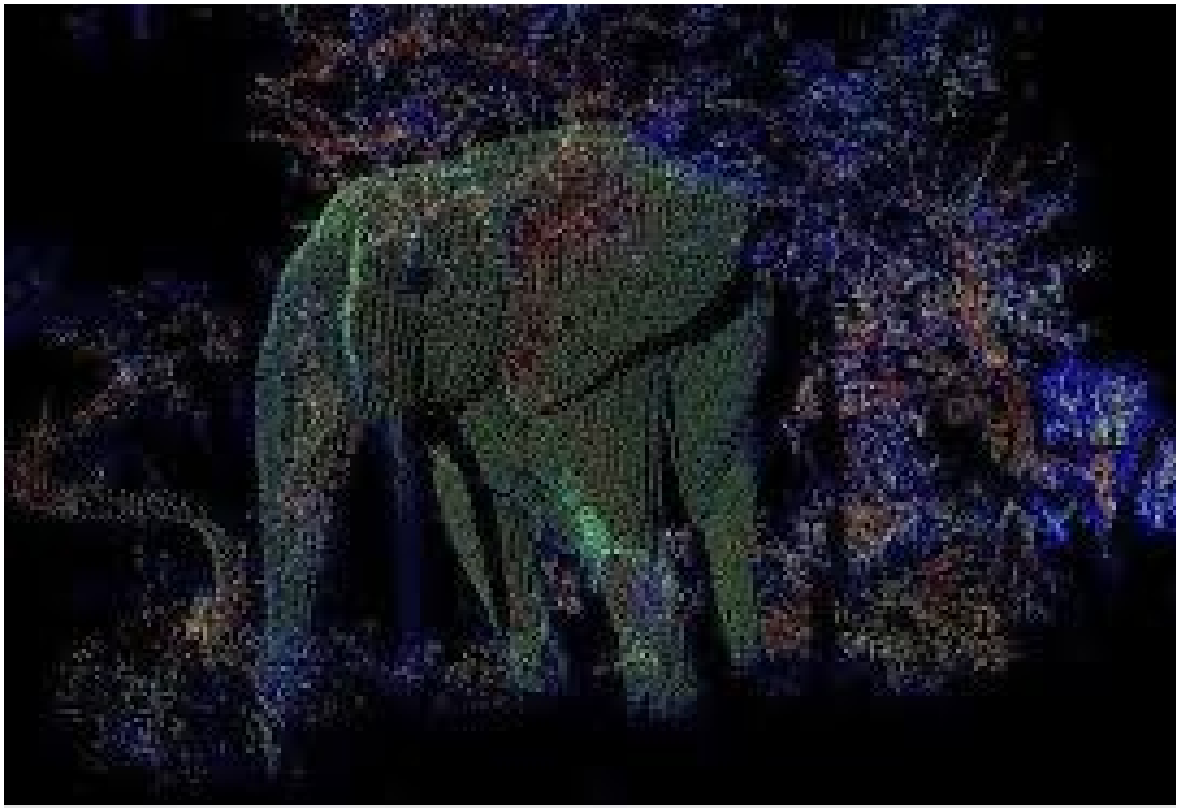
PointCloud

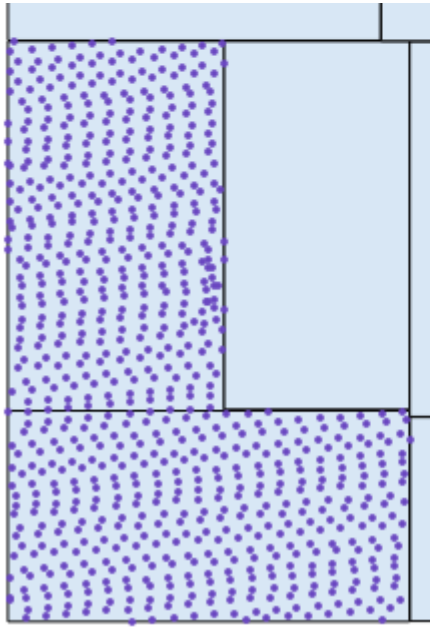

TOMTOM
www.tomtom.com
TOMTOM
We're mapping your world.
SICK

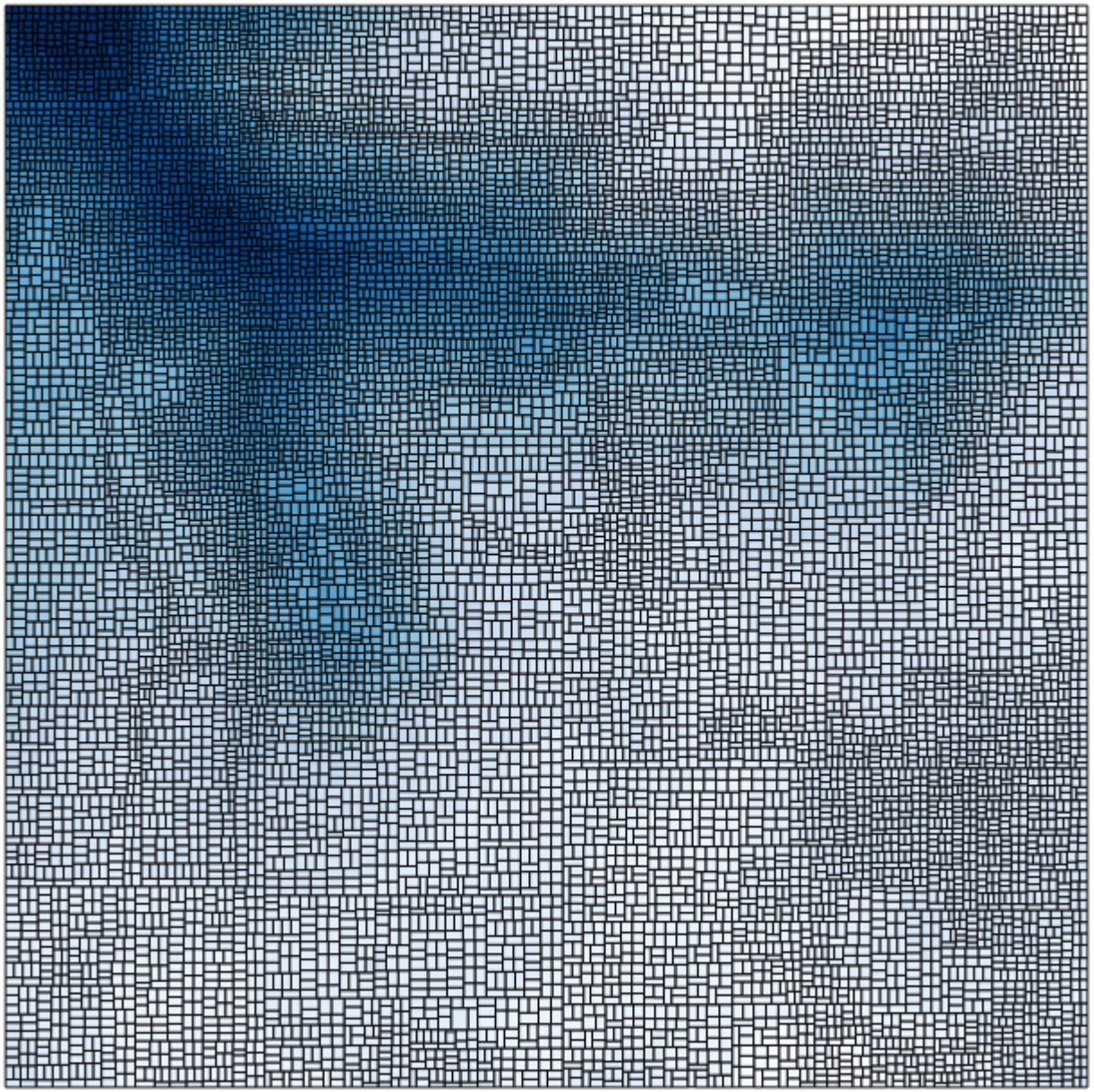
PointCloud

- Extension PostgreSQL + PostGIS
- Paul Ramsey – Natural resources Canada
- LIDAR – Jeux de données gigantesques
- N-Dimensional
- PDAL I/O

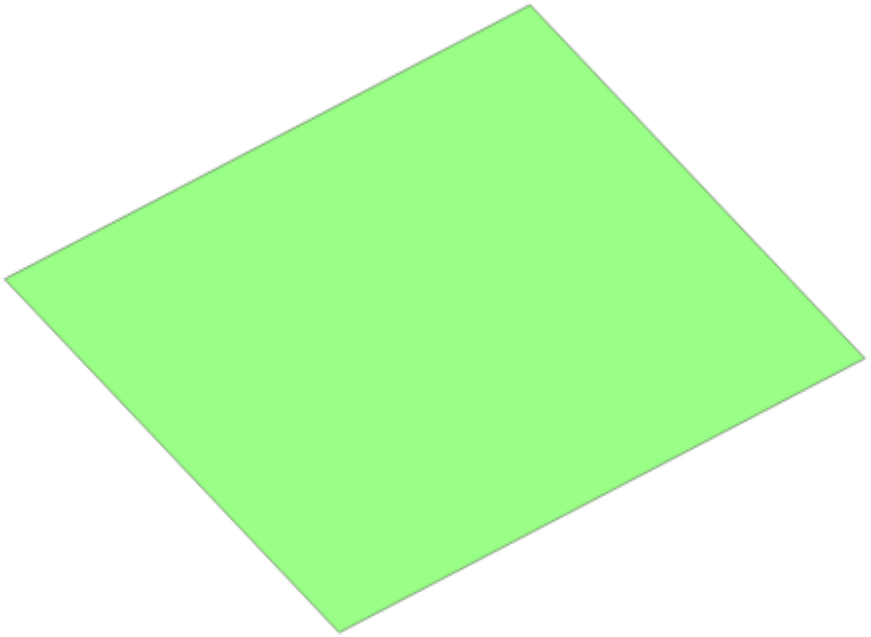
- Point patches
- Indexes
- Fonctions

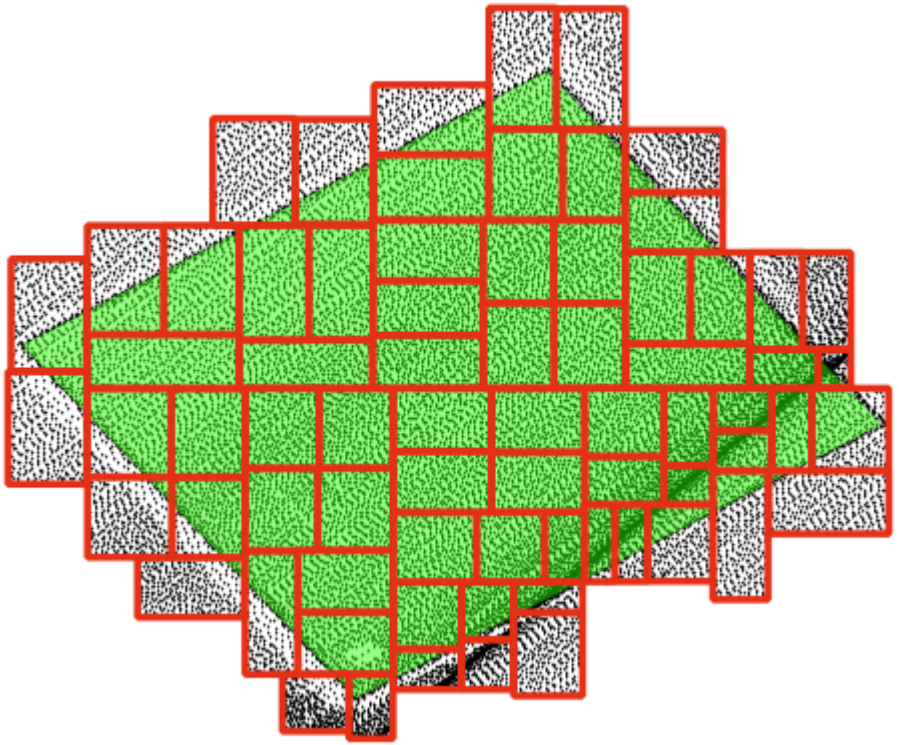


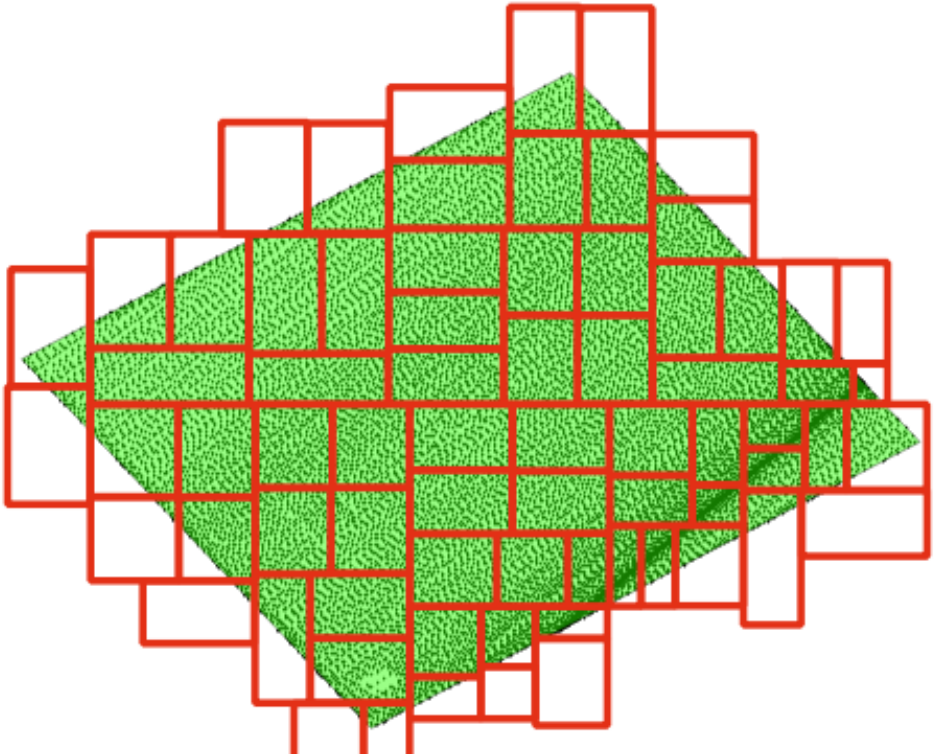


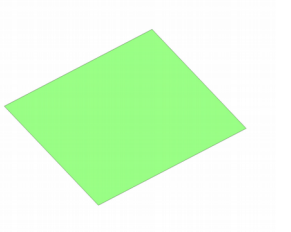



고무래길 (Gomuraegil)
고무래로
고무래로
고무래로
서초대로65길 (Seocho-daero 65-gil)
서초대로65길
서초대로65길
서초대로65길 (Seocho-daero
서운로 (Seoun-ro)
서운로 (Seoun-ro)
경부고속도로 (Gyeongbu Expressway)
경부고속도로 (Gyeongbu Expressway)
경부고속도로 (Gyeongbu Expressway)
경부고속도로 (Gyeo
지하도

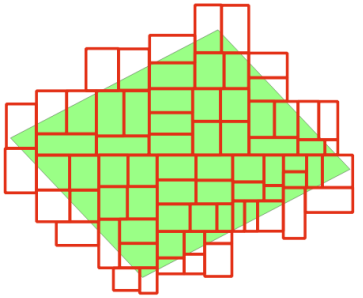


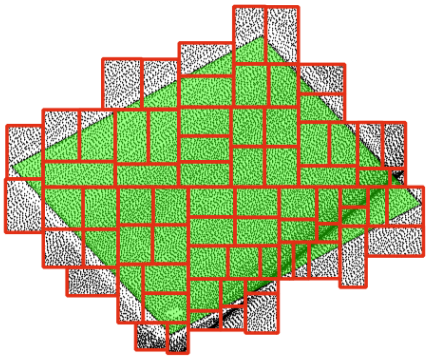




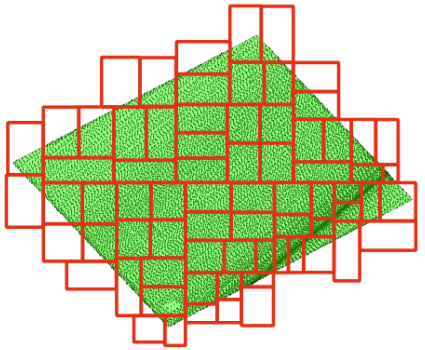


```
-- The building we are interested in
WITH building AS (
  SELECT geom FROM buildings WHERE buildings.gid = 81719
),
-- All the patches that intersect that building
patches AS (
  SELECT pa FROM pcpatches
            JOIN building ON PC_Intersects(pa, geom)
),
-- All the points in that patch
pa_pts AS (
  SELECT PC_Explode(pa) AS pts FROM patches
),
```





```
  ...
-- All the points in our one building
building_pts AS (
  SELECT pts FROM pa_pts
             JOIN building ON ST_Intersects(geom, pts::geometry)
)
-- Summarize those points by elevation
SELECT Avg(PC_Get(pts, 'z')) AS lidar_meters FROM building_pts;
```



441.075 metres !

Boundless

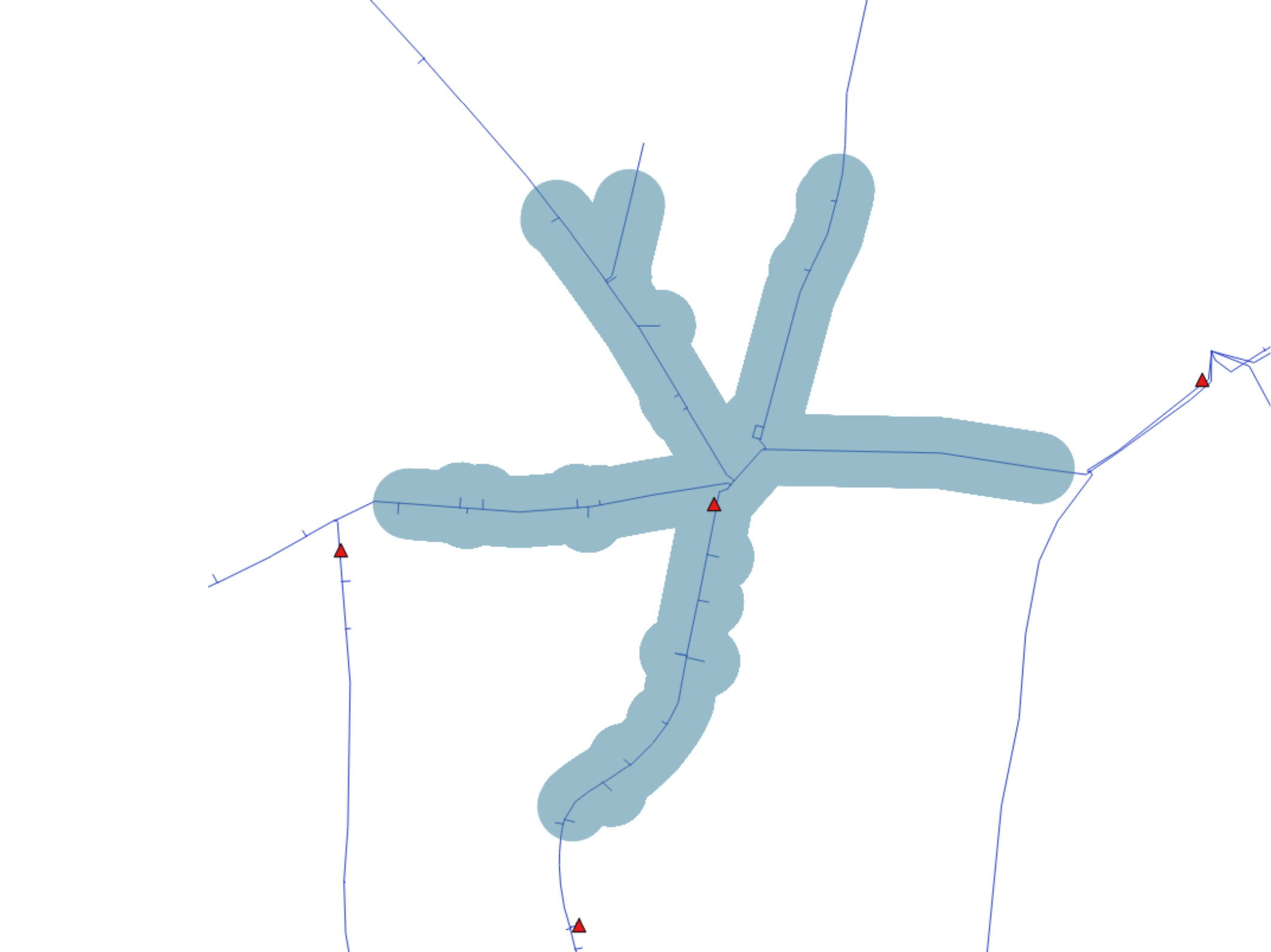
Au feu !


OTTIWA F.D.
PHOTOGRAPHER
L13

Au feu !

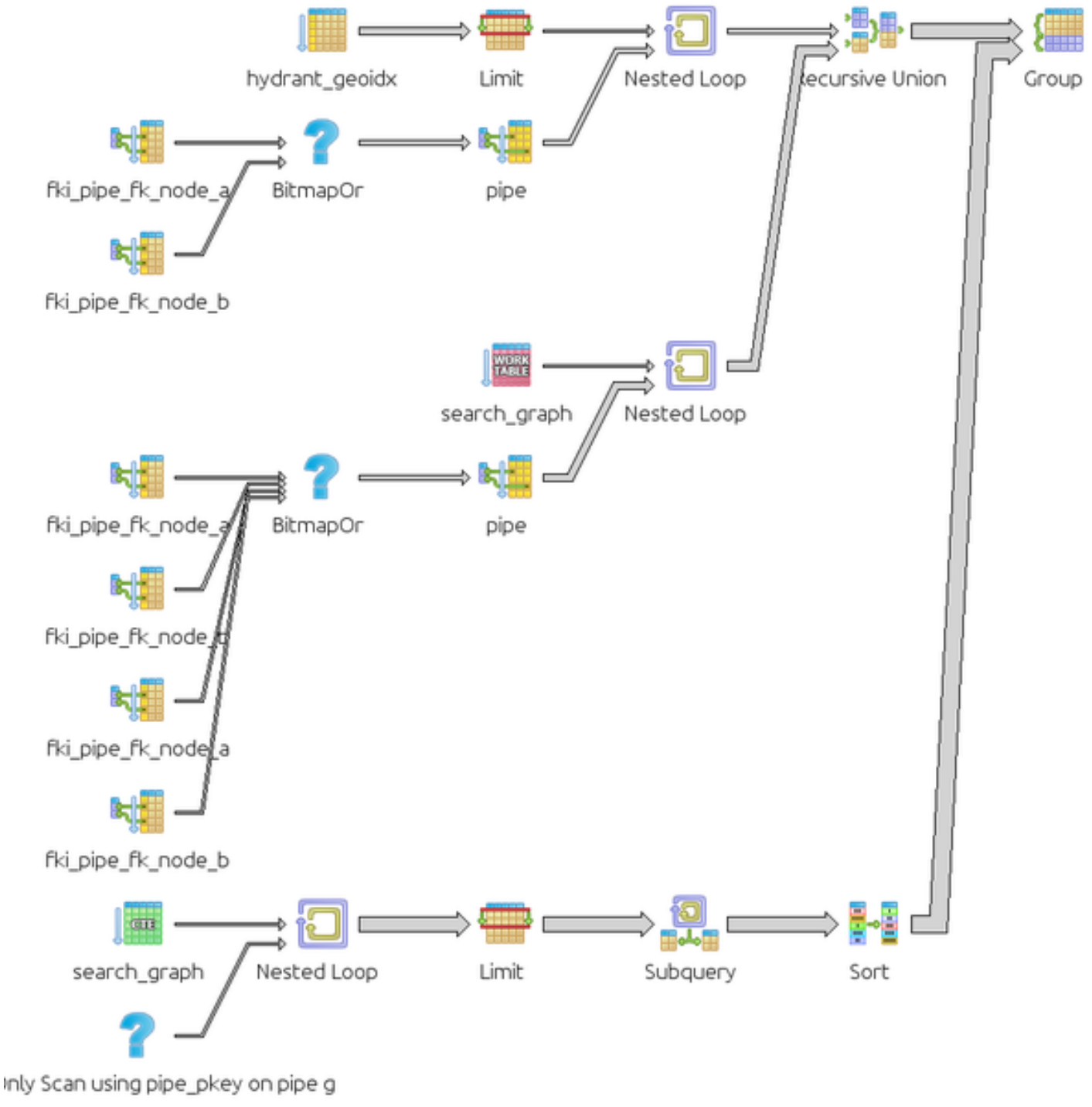
À partir d’un point de la carte, trouver les zones où on peut amener de l’eau

- **En entrée : réseau topologique de canalisations**
- **Il faut suivre les rues (= les canalisations)**
- **La lance à incendie fait 220m**



Au feu !

1. Trouver l’hydrant le plus proche
2. Trouver les canalisations connectées
3. Suivre le réseau sur 200m
4. À la fin des canalisations atteignables, récupérer uniquement la partie < 200m total
5. Créer une zone de 20m autour du trajet


hydrant_geoidx
Limit
Nested Loop
ecursive Union
Group
fki_pipe_fk_node_a
BitmapOr
pipe
fki_pipe_fk_node_b
WORK TABLE
search_graph
Nested Loop
fki_pipe_fk_node_a
BitmapOr
pipe
fki_pipe_fk_node_b
fki_pipe_fk_node_a
fki_pipe_fk_node_b
CTE
search_graph
Nested Loop
Limit
Subquery
Sort
nly Scan using pipe_pkey on pipe g

1. Hydrant le plus proche

- KNN-GIST search : PostgreSQL 9.1+
- RECHECK : PG 9.5+
- Utilise les indexes !
- Recherche de plus proches voisins spatiale

```
select * from
    -- our hydrant
    qwat_od.hydrant as h
order by
    h.geometry <-> st_makepoint(845499.23,6473465.22, 2154)
limit 1;
```

- NEW : Pas besoin de seconde passe pour les geométries autres que les points

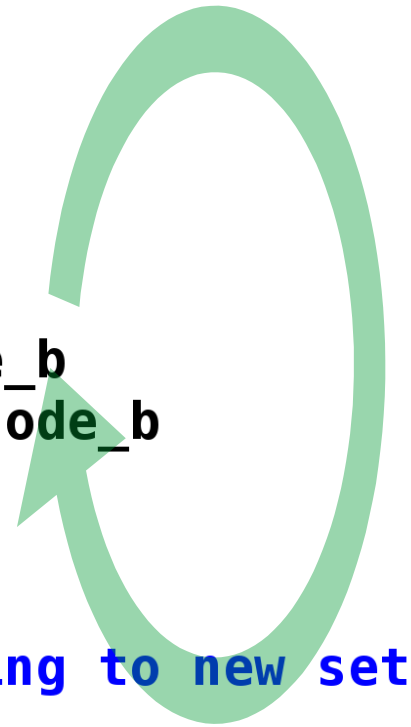
```
select
    g.id as gid
    , g.fk_node_a as nodea
    , g.fk_node_b as nodeb
    , g.geometry as geom
    , 1 as depth, ARRAY[g.id] as path
    , st_length(g.geometry) as length
    , false as cycle
    , st_length(g.geometry) > 200 as islast
    , true as forward_geom
from (
    -- our hydrant here
    ...
) as h
-- Get connected pipes
join
    qwat_od.pipe as g
    -- forward or backward
on h.fk_node = g.fk_node_a
   or h.fk_node = g.fk_node_b
```

2. Les canalisat
connectées

3. Suivons le réseau
CTE récursive !

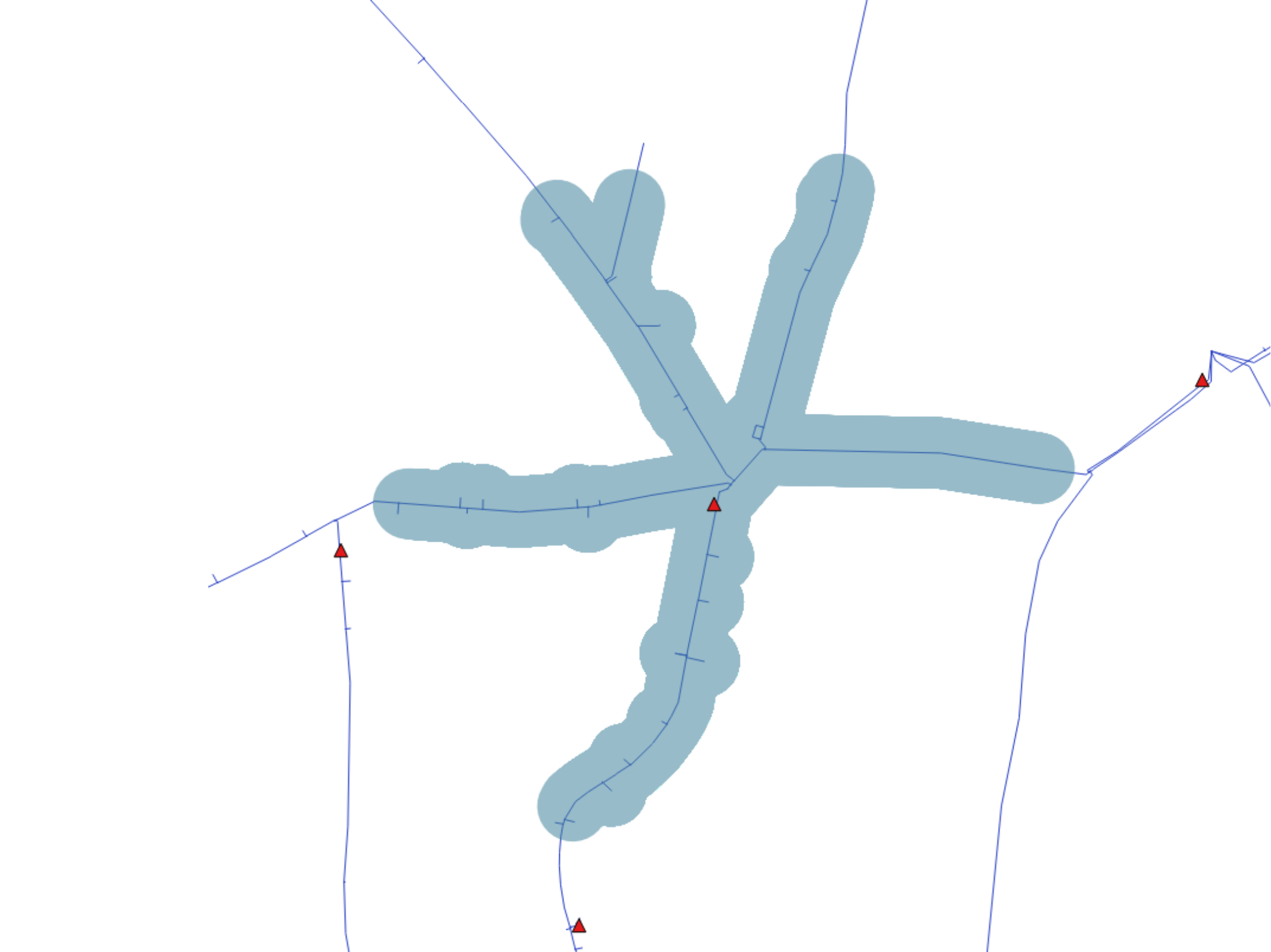
```
WITH RECURSIVE search_graph(...) AS
    -- Initial dataset : pipes connected to our hydrant ( see #2 )
    SELECT ... FROM qwat_od.hydrant JOIN qwat_od.pipe ...
    -- Recursive dataset
+   UNION ALL
    SELECT ...
    FROM
        search_graph as sg
    JOIN
        qwat_od.pipe as g
    ON
        -- Previous dataset pipes are connected
        -- to other pipes one side or the other
        sg.nodeb = g.fk_node_a or sg.nodeb = g.fk_node_b
        or sg.nodea = g.fk_node_a or sg.nodea = g.fk_node_b
    WHERE
        -- no loop
        and sg.gid <> g.id and not cycle
        -- when total computed length > 200, add nothing to new set
        and sg.length < 200
```

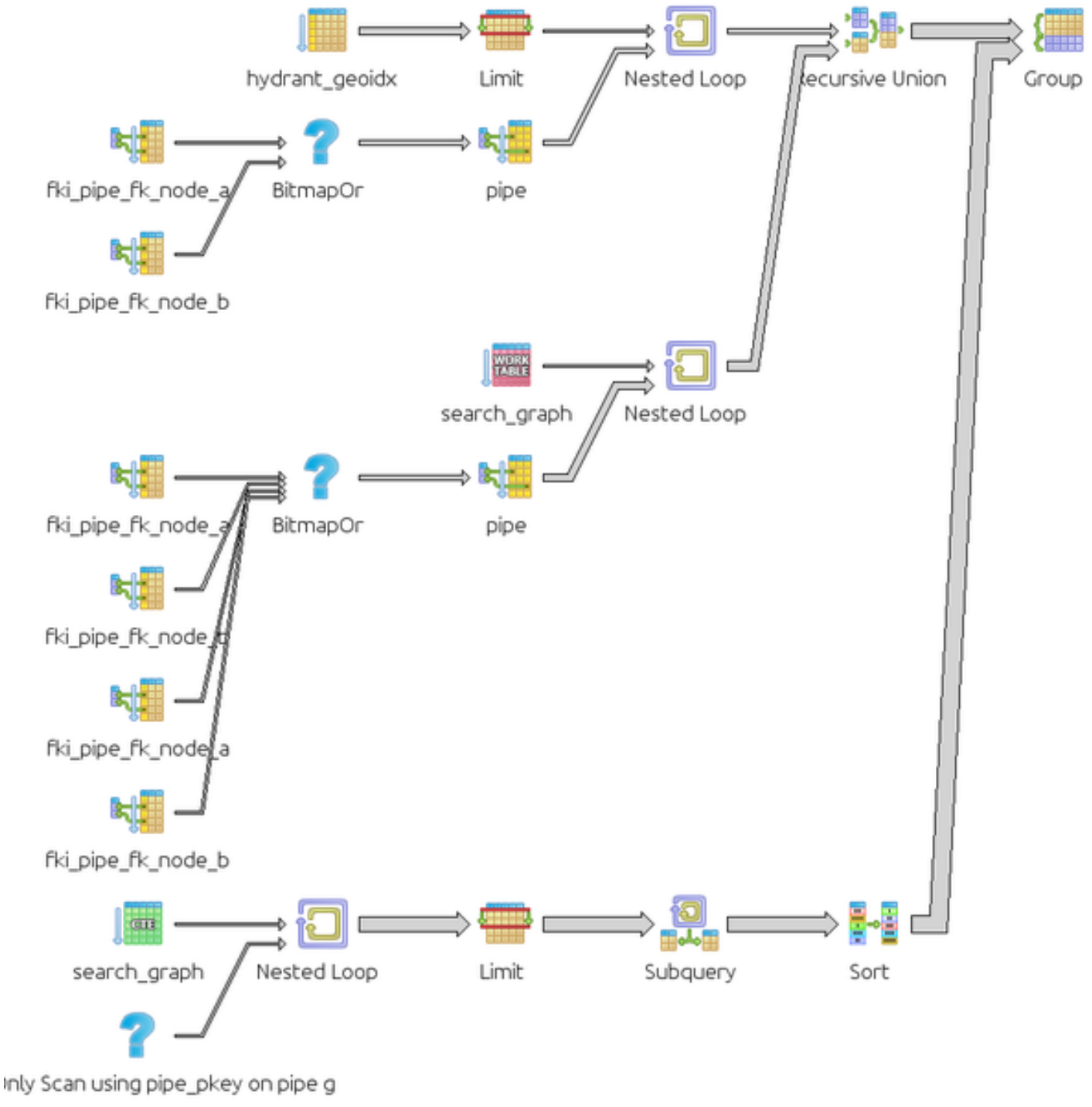
Le calcul des **cycles et de la longueur** est laissé en exercice pour l'utilisateur



5. Buffer sur le chemin

```
select sg.gid, sg.geom from (
select
        -- if pipe is part of path, buffer it
        , case
            when not islast then st_buffer(sg.geom, 20)
            -- if pipe is last of path, take only the portion to reach 200m
            -- and buffer it
            else st_buffer(
                st_linesubstring(sg.geom, 0,
                  (200 + st_length(sg.geom) - sg.length) / st_length(sg.geom))
                 , 20)
        end as geom
from
        search_graph as sg
-- limit the recursive search in case we
-- failed the stop criteria
limit 50000
```




hydrant_geoidx
Limit
Nested Loop
ecursive Union
Group
fki_pipe_fk_node_a
BitmapOr
pipe
fki_pipe_fk_node_b
WORK TABLE
search_graph
Nested Loop
fki_pipe_fk_node_a
BitmapOr
pipe
fki_pipe_fk_node_b
fki_pipe_fk_node_a
fki_pipe_fk_node_b
CTE
search_graph
Nested Loop
Limit
Subquery
Sort
nly Scan using pipe_pkey on pipe g

La donnée est ailleurs

oracle_fdw

- FDW pour BdD Oracle (Laurenz Albe)
- Nouveau : Support Oracle Spatial (Vincent Mora)
- Natif, R+W, rapide !
- Points, lignes & polygones
- Combinaison avec Vues Materialisées

- Systèmes hétérogènes
- Migration

- Support ora2pg

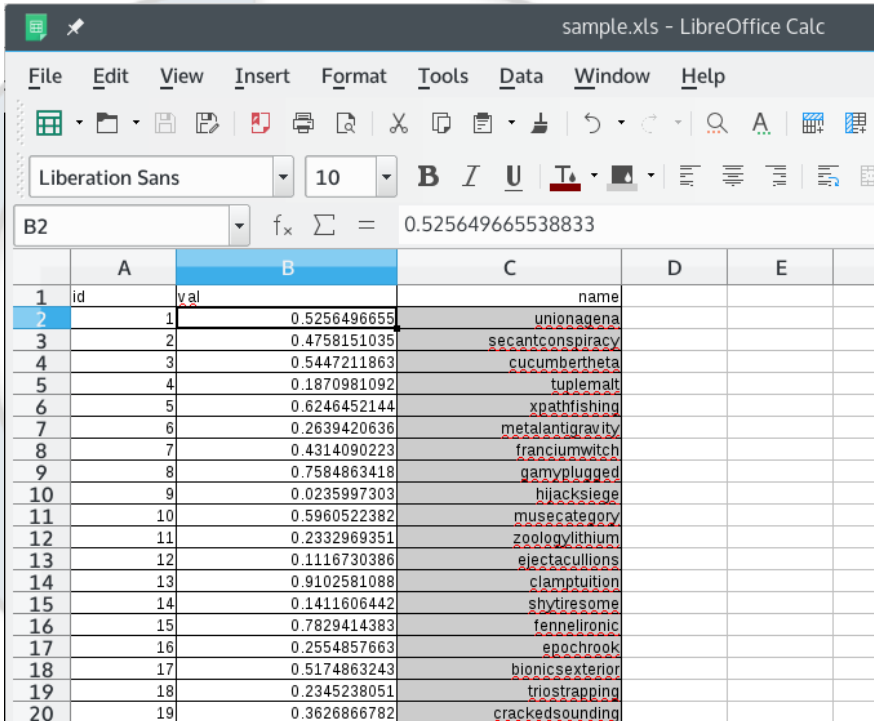
ogr_fdw

- OGR = Accès donnée vecteur
- Foreign Data Wrapper
- Données Géo / non-géo
- Beaucoup de formats :
  - Shapefile, SpatiaLite, WFS, GeoJSON...

ogr_fdw

- OGR = Accès données vecteur
- Foreign Data Wrapper
- Données Géo / non-géo
- Beaucoup de formats :
  - Shapefile, SpatiaLite, WFS, GeoJSON…

… Fichiers Excel !



	A	B	C	D	E
1	id	val	name		
2	1	0.5256496655	unionagena		
3	2	0.4758151035	secantconspiracy		
4	3	0.5447211863	cucumbertheta		
5	4	0.1870981092	tuplemalt		
6	5	0.6246452144	xpathfishing		
7	6	0.2639420636	metalantigravity		
8	7	0.4314090223	franciumwitch		
9	8	0.7584863418	gamyplugged		
10	9	0.0235997303	hijacksiege		
11	10	0.5960522382	musecategory		
12	11	0.2332969351	zoologylithium		
13	12	0.1116730386	ejectacullions		
14	13	0.9102581088	clamptuition		
15	14	0.1411606442	shytiresome		
16	15	0.7829414383	fennelironic		
17	16	0.2554857663	epochrook		
18	17	0.5174863243	bionicsexterior		
19	18	0.2345238051	triostrapping		
20	19	0.3626866782	crackedsounding		

```
./ogr_fdw_info -s /tmp/sample.xls -l Sheet1

CREATE EXTENSION ogr_fdw;

CREATE SERVER myserver
  FOREIGN DATA WRAPPER ogr_fdw
  OPTIONS (
    datasource '/tmp/sample.xls',
    format 'XLS' );

CREATE FOREIGN TABLE sheet1 (
  fid integer,
  id integer,
  val real,
  name varchar )
  SERVER myserver
  OPTIONS ( layer 'Sheet1' );
```

```
pggis=# select * from sheet1 where val < 0.2;
 fid | id |     val     |        name
-----+----+-------------+--------------------
   5 |  4 |  0.187098   | tuplemalt
  10 |  9 | 0.0235997   | hijacksiege
  13 | 12 |  0.111673   | ejectacullions
  15 | 14 |  0.141161   | shytiresome
  26 | 25 |  0.175975   | broodbugs
  28 | 27 | 0.0276925   | lumberwhizzer
  30 | 29 |  0.141428   | sastor
  36 | 35 | 0.0231349   | pancakeplush
  37 | 36 | 0.0174379   | lithuaniancostly
  48 | 47 |  0.131352   | kaputalkane
(10 rows)
```

Testez vous même !

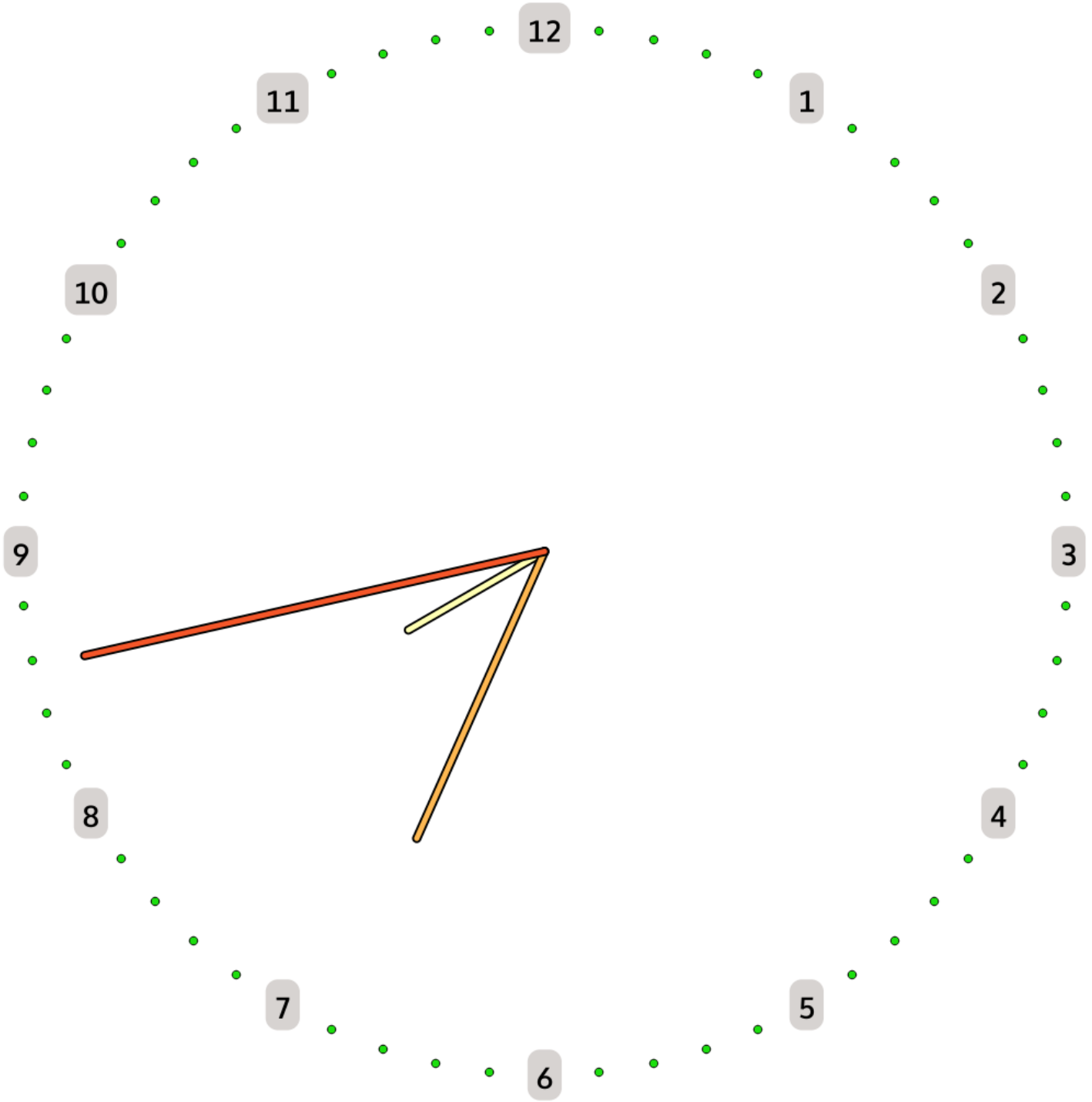
docker-pggis

- Container avec outils BdD SIG
- Dernières versions
  - PG 9.5
  - PostGIS 2.2 + SFCGAL + GEOS 3.5
  - pgPointCloud master
  - PDAL
  - OGR_FDW
- Lancez tout en deux lignes
- Basé sur phusion baseimage
- Pas en production !

https://github.com/vpicavet/docker-pggis

```
docker build -t oslandia/pggis .
docker run --rm -P --name pggis_test oslandia/pggis
```

L'horloge PostGIS


12
1
2
3
4
5
6
7
8
9
10
11

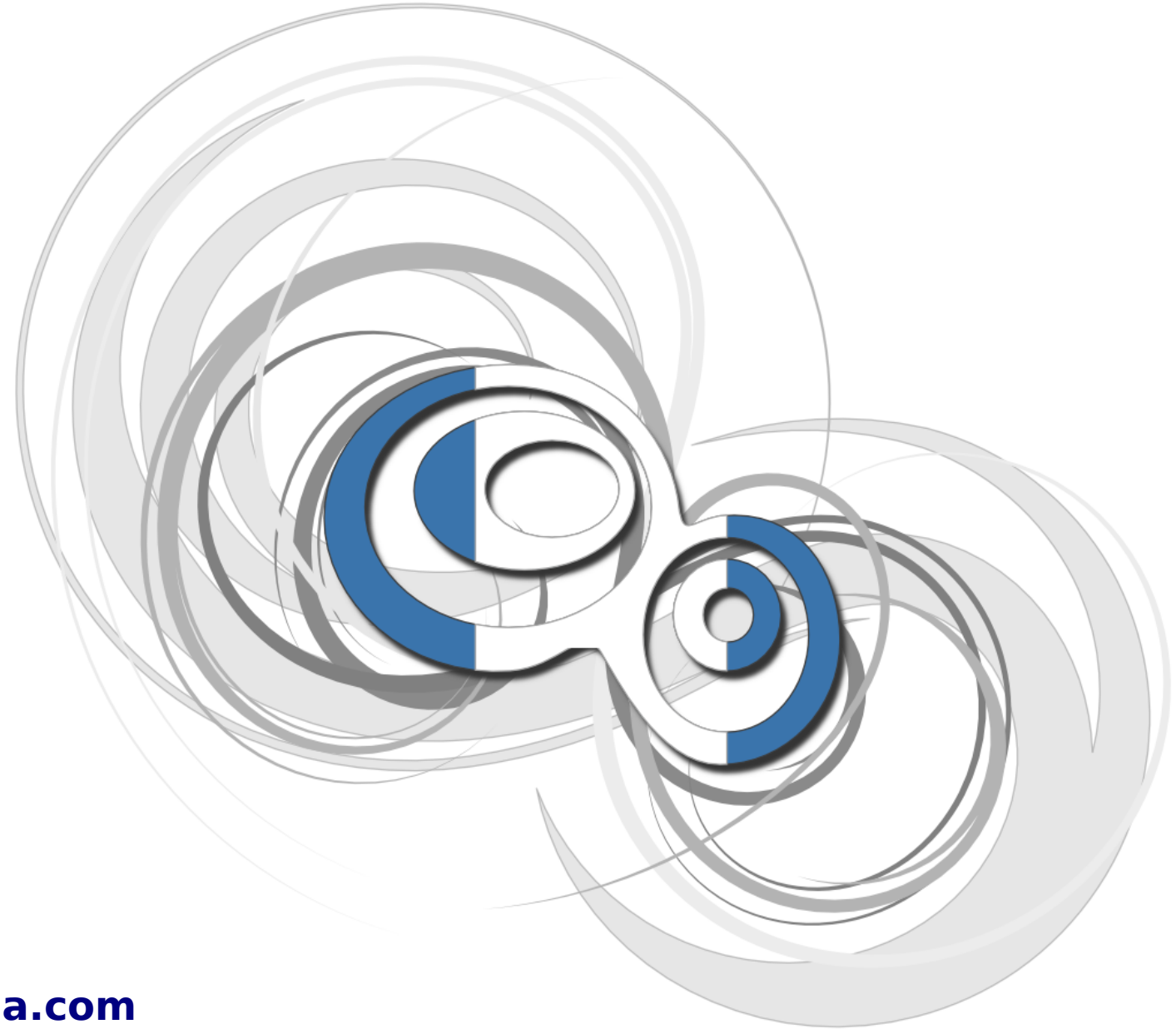
```
create or replace view ticks as
select
    m as id
    , case when m % 5 = 0 then 'h' else 'm' end as typ
    , m / 5 as h
    , m as m
    , st_makepoint(
        cos(m::double precision * -pi() / 30.0 + pi() / 2)
        , sin(m::double precision * -pi() / 30.0 + pi() / 2)
    )
from
    generate_series(1, 60) as m;
```

```
create or replace view clock as
with hms (id, unit, length, nb, val) as (
        values (1, 'h', 0.3, 12, extract('hour' from current_time))
            , (2, 'm', 0.6, 60, extract('minute' from current_time))
            , (3, 's', 0.9, 60, extract('second' from current_time))
    )
select
    hms.id
    , st_makeline(
        st_makepoint(0, 0)
        , st_makepoint(
            hms.length * cos(val::double precision *-2* pi() / nb + pi() / 2)
            , hms.length * sin(val::double precision *-2*pi() / nb + pi() / 2)
        )
    ) as geom
from
    hms;
```

Merci !

vincent.picavet@oslandia.com
@vpicavet

OSLANDIA



www.oslandia.com
@Oslandia_en / @Oslandia_Team