PostGIS - Gérer des données géographiques dans PostgreSQL

Vandy Berten – Webinar 27/09/2022

Smals
ICT for society

Smals Research 2022

Innovation with new technologies

Consultancy & expertise

Internal & external knowledge transfer

Support for going live

2022

- Low Code Development
- AI-Augmented Software Development
- Resilient Application Architectures
- On-premise Cloud: Anthos
- GIS for Analytics
- Near real time Translation
- Passwordless Authentication using FIDO
- Flexible Authorization Management
- Recommender Systems
- Named Entity Recognition Service
- Approach for detecting AI Cases
- Pseudonymization Service
- Format-Preserving Encryption Service
- Synthetic Data

Content

Context	• What is PostGIS, what kind of data • Data source
PostGIS queries	• Basic queries • Spatial join
Data load	• How to feed PostGIS
Indexing	• How to speed up queries

Previous webinar: https://www.smalsresearch.be/webinar-gis-analytics-follow-up/ (05-05-2022)

Python notebook : https://github.com/SmalsResearch/GISAnalytics/ → Webinar_PostGIS

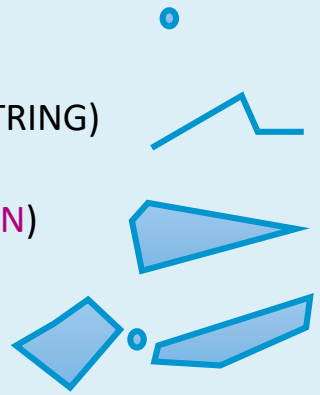
Smals
ICT for society

Context & data

PostGIS

- PostGIS (https://postgis.net/) is an extension (plugin) of PostgreSQL
- **Post**greSQL + **G**eographical **I**nformation **S**ystem

- Add new (spatial) data types :
  - POINT (+MULTIPOINT)
  - LINESTRING (+MULTILINESTRING)
  - POLYGON (+MULTIPOLYGON)
  - GEOMETRYCOLLECTION



- Add set of new (spatial) operations "ST_" (spatial type):
  - Comparison: ST_Contains, ST_Within, ST_Intersects
  - Aggregation: ST_Union, ST_Collect
  - Combinations: ST_Intersection, ST_Difference
  - Measurement: ST_Distance, ST_Area
  - Projection handling: ST_Transform
  - …

- Add a new (set of) spatial index type(s)

(Minimal) Installation:

- Install PostgreSQL
- Install PostGIS binary (or build sources):

```
sudo apt-get install postgresql-14-postgis-3
```

- `CREATE EXTENSION postgis;`

Geographical data

- Geographical point :

Degrés (géographique)

Mètres (cartésien)

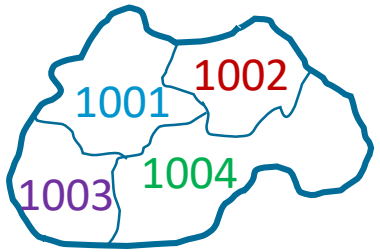
Latitude (Y)	Longitude (X)	CRS (Coordinate Reference System)/ SRID (Spatial Reference ID)
• 50.8361263,	4.3382716	(WGS 84 – EPSG:4326)
• 50°50'10.1"N,	4°20'17.8"E	
• 6592292.39,	482944.07	(WGS 84 – EPSG:3857)
• 169482.70,	147859.16	(Lambert 1972 – EPSG:31370)

- We don’t consider indirect geographical data (addresses, city names…)

Smals ICT for society

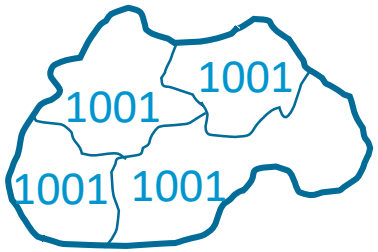
Belgium division schemes – Postal scheme

- Postal schemes (BPost): main commune – localities



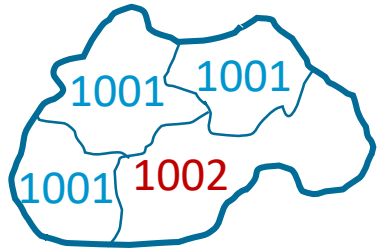
Main commune : Bruxelles(-Ville)

Zipcode	Locality
1000	Bruxelles
1020	Laeken
1120	Neder-Over-Hembeek
1130	Haren



Main commune : Jemeppe-Sur-Sambre

Zipcode	Locality
5190	Jemeppe-sur-Sambre
5190	Spy
5190	Onoz
5190	Balâtre
5190	…



Main commune : Assesse

Zipcode	Locality
5330	Assesse
5330	Maillen
5330	Sart-Bernard
5332	Crupet
5333	…

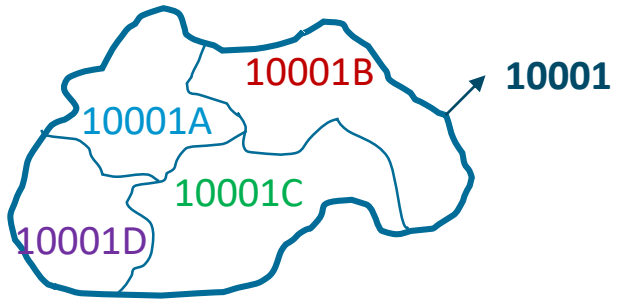
- No (numerical) ID for a commune (only « commune name ») → 581 values
- Locality ID : Zipcode + locality name
- Lots of « special codes » (1110 NATO, 1044 RTBF, …)

https://www.bpost.be/fr/codes-postaux

Smals ICT for society

Belgium division schemes – statistical scheme

- Statistical scheme (Statbel/Cadastre – SPF Economy):
  - Main commune (NIS5) → 581 values
  - Sections (NIS6)



Main commune : Bruxelles(-Ville)

NIS5		NIS6
21004	BRUXELLES PENTAGONE	21004A
21004	BRUXELLES-RUE DE LA LOI	21004B
21004	BRUXELLES-LOUISE	21004C
21004	BRUXELLES-CHAUSSEE D'ANVERS	21004D
21004	BRUXELLES-LAEKEN	21004E
21004	BRUXELLES-NEDER-OVERHEEMBEEK	21004F
21004	BRUXELLES-HAREN	21004G

- No 1-1 mapping Zipcode-NIS5 (or NIS6)
- No special code

- In our demo: do they agree at the commune level?

https://statbel.fgov.be/fr/propos-de-statbel/methodologie/classifications/geographie

Smals ICT for society

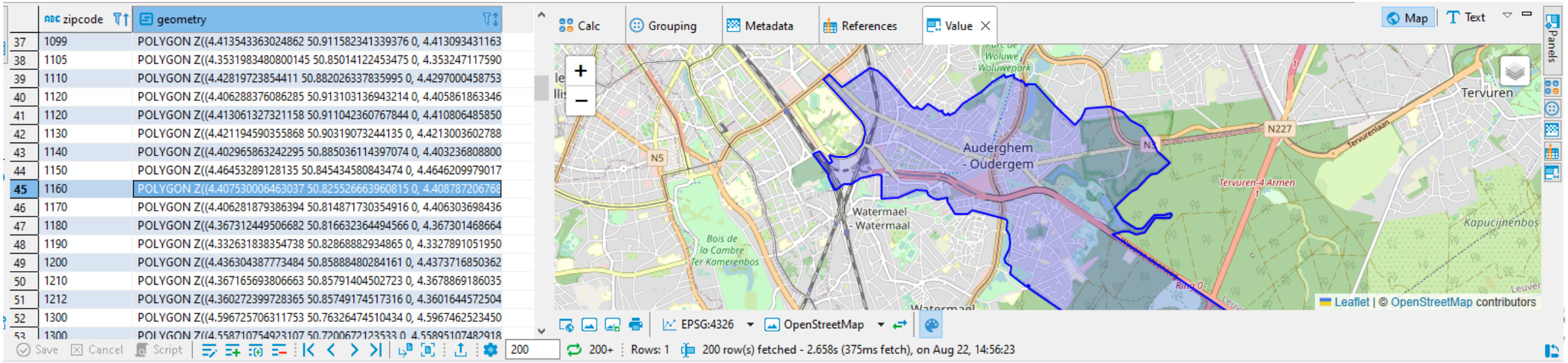
Data sources

- We will load 4 OpenData sources:

Subject	Source	Shapes	Format	CRS	URL
Zipcode list (mapping zipcode-main commune)	BPost	Non geo	Excel	[NA]	https://www.bpost2.be/(...)/zipcodes_alpha_fr_new.xls
Zipcode boundaries	BPost	Polygons	Shapefile	3812	https://bgu.bpost.be/(...)_x-shapefile_3812.zip
Zipcode centers	Agence du Numérique	Points	GeoJSON	4326	https://www.odwb.be/(...)/code-postaux-belge/exports/geojson
NIS code boundaries (statistical code level)	Statbel	Polygons	Shapefile	31370	https://statbel.fgov.be/(...)/sh_statbel_statistical_sectors_31370_20200101.shp.zip

→ To be standardized!!

- In DBeaver:



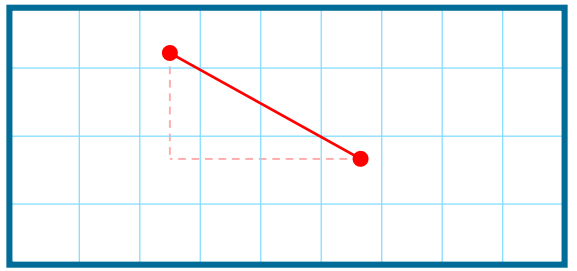
Basic functions

Simple distance

```
SELECT a.locality,
       b.locality,
       ST_Distance(a.geometry, b.geometry),
       ST_DistanceSphere(a.geometry, b.geometry)
FROM gistest.zipcodes_centers a,
     gistest.zipcodes_centers b
WHERE a.zipcode='1160' AND b.zipcode='5190'
```

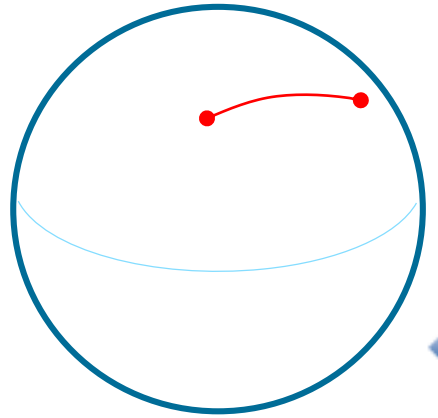
Geometric Euclidian distance (ok only for metric CRS)

$$0{,}43783 = \sqrt{(4.433 - 4.995)^2 + (50.815 - 50.828)^2}$$



	locality	locality	st_distance	st_distancesphere
1	Auderghem	Moustier-Sur-Sambre	0.4378300821	43,089.00047741
2	Auderghem	Saint-Martin	0.3807369552	38,121.84729085
3	Auderghem	Balâtre	0.3778559061	38,134.34560066
4	Auderghem	Onoz	0.4006119939	39,630.80735948
5	Auderghem	Jemeppe-Sur-Sambre	0.4164304942	42,209.58949063
6	Auderghem	Ham-Sur-Sambre	0.4421927051	44,618.0699368
7	Auderghem	Mornimont	0.4508334992	44,379.13904389
8	Auderghem	Spy	0.4295900467	41,768.06793197

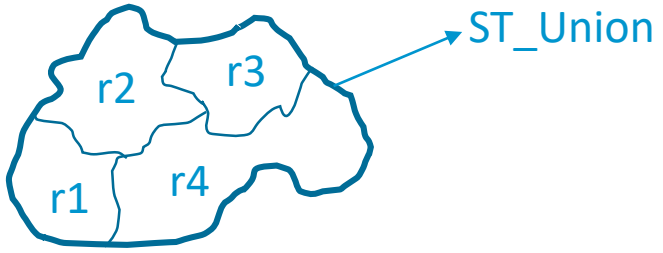
Geographical distance (in meters)



Smals
ICT for society

Aggregation

```
SELECT c_provi,
       t_provi_fr,
       t_regio_fr,
       ST_Union(geometry) as geometry
FROM gistest.statistical_sectors
GROUP BY c_provi, t_provi_fr, t_regio_fr
```



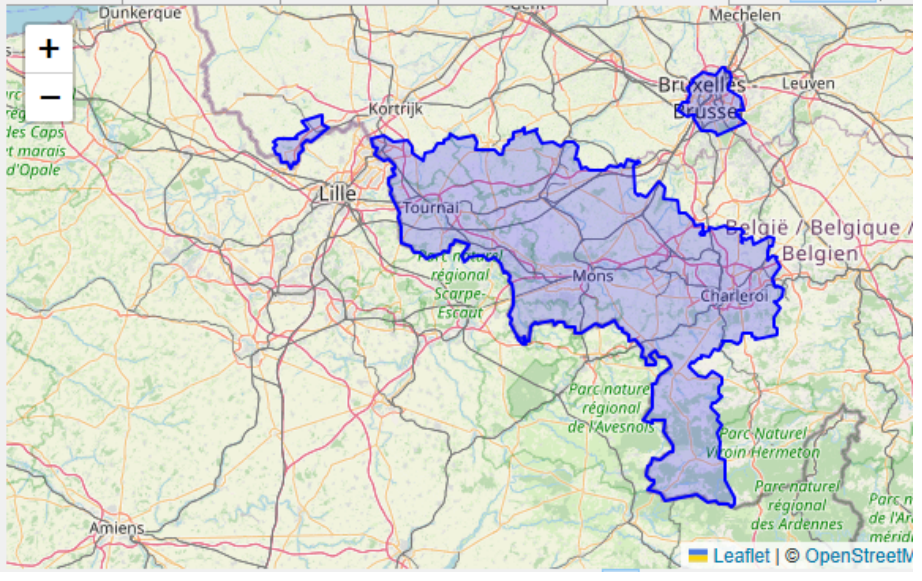
statistical_sectors 1

SELECT c_provi, t_provi_fr, t_regio_fr, ST_Union(geometry) as geometry FROI | Enter a SQL expression to filter results (use Ctrl+Space)

	c_provi	t_provi_fr	t_regio_fr	geometry
1	10000	Province d'Anvers	Région flamande	MULTIPOLYGON Z(((4.935794074100193 51.409299441363345 0,
2	-	[NULL]	Région de Bruxelles-Capitale	POLYGON Z((4.297395339460635 50.81001442302577 0, 4.29735
3	90000	Province de Namur	Région wallonne	POLYGON Z((4.513943097267026 49.94891534289222 0, 4.51384
4	20001	Province du Brabant flamand	Région flamande	POLYGON Z((5.107842576285907 50.722848094043826 0, 5.1073
5	40000	Province de Flandre orientale	Région flamande	POLYGON Z((3.4916480229646765 50.757922055392285 0, 3.491
6	30000	Province de Flandre occidentale	Région flamande	POLYGON Z((2.842299598820434 50.73559228368382 0, 2.84249
7	50000	Province du Hainaut	Région wallonne	MULTIPOLYGON Z(((2.911669461934257 50.70399969450178 0,
8	80000	Province du Luxembourg	Région wallonne	POLYGON Z((5.0124933683789274 49.99737467979133 0, 5.0124
9	20002	Province du Brabant wallon	Région wallonne	POLYGON Z((4.145748827328134 50.636597982452905 0, 4.1457
10	60000	Province de Liège	Région wallonne	POLYGON Z((5.351476312487963 50.38838397246115 0, 5.35143
11	70000	Province du Limbourg	Région flamande	MULTIPOLYGON Z(((5.895636190850145 50.745452548013176 0,

Grid | Text | Spatial | Record

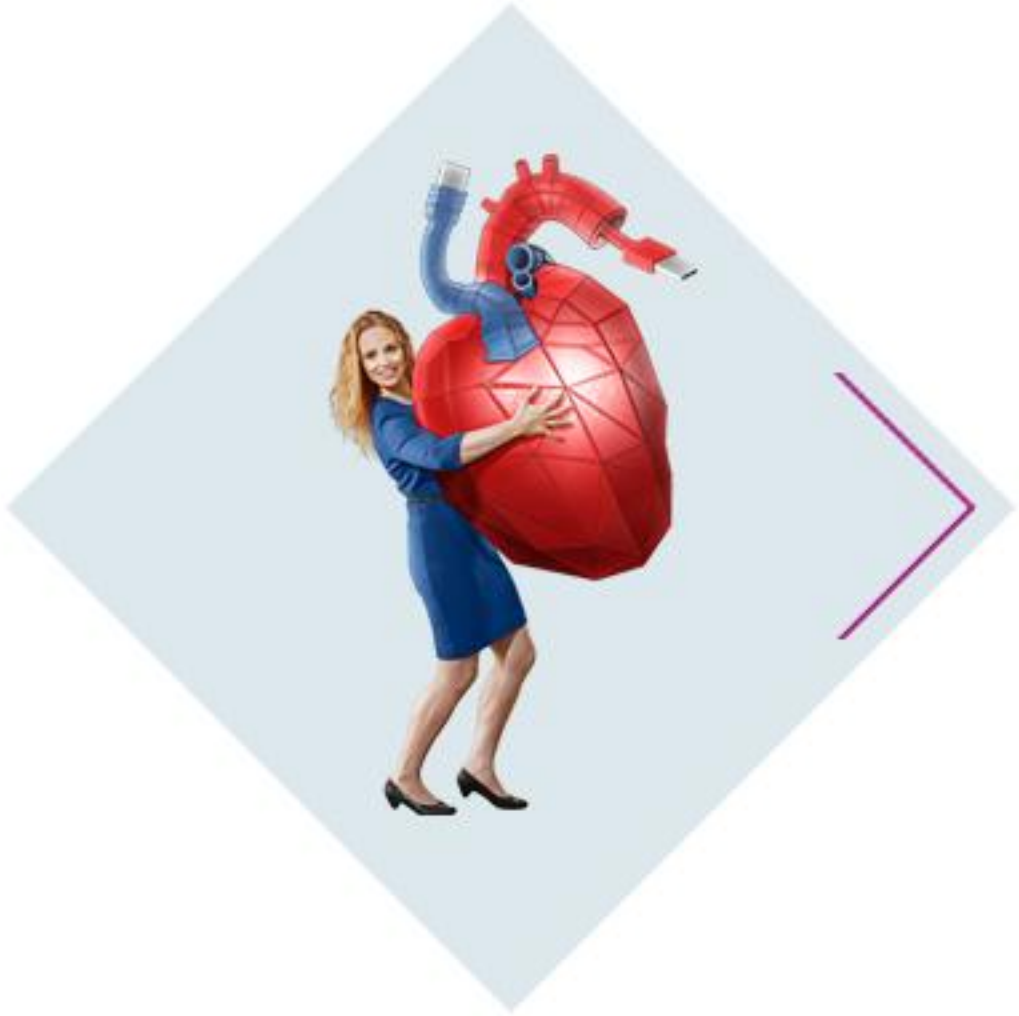
Calc | Grouping | Metadata | References | Value | Map



Aggregation – commune boundaries

```
-- Create a table with NIS boundaries
CREATE TABLE gistest.statbel_communes AS
-- or: CREATE VIEW
      SELECT cnis5_2020, -- Code NIS5
             t_mun_fr, -- Commune name
             ST_Union(geometry) AS geometry
      FROM gistest.statistical_sectors
      GROUP BY cnis5_2020, t_mun_fr, t_mun_nl
```

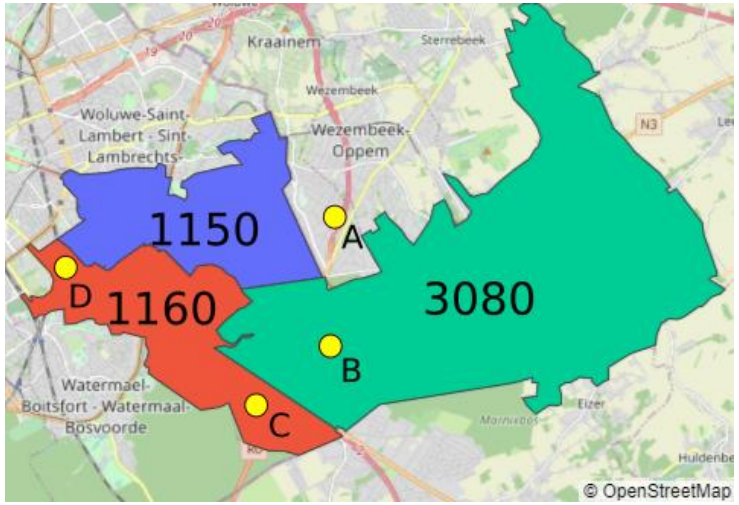

```
-- Create a table with postal boundaries
-- Assuming 'main_commune2' has been filled
-- for special zipcodes (see notebook for details)
CREATE TABLE gistest.bpost_communes AS
      SELECT main_commune2 as main_commune,
             ST_Union(geometry) AS geometry
      FROM gistest.zipcodes zp
      JOIN gistest.zipcodes_boundaries zp_bnd
             ON zp.zipcode = zp_bnd.zipcode
      GROUP BY main_commune2
```



Spatial join

Spatial join

"Classical" join:



observations

ID	ZIPCODE
A	
B	3080
C	1160
D	1160
…	…

zipcodes

ZIPCODE	PROVINCE
1150	Bruxelles-Capitale
1160	Bruxelles-Capitale
3080	Brabant Flamand
…	…

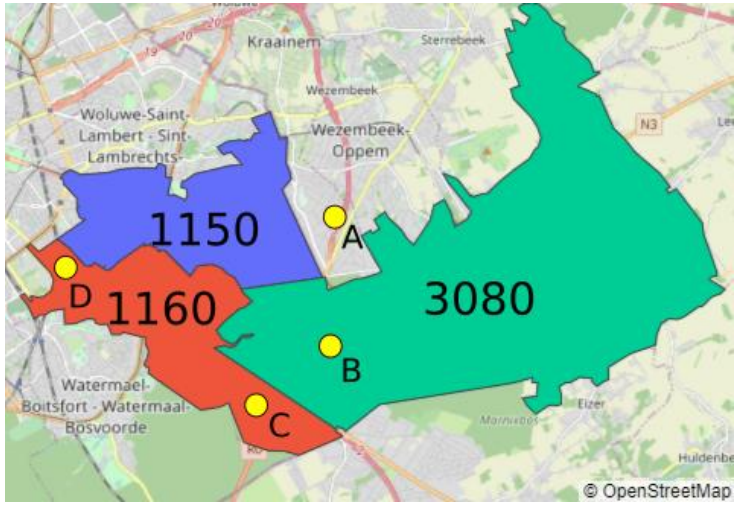
```
SELECT o.ID, o.zipcode, z.province
FROM observations o
LEFT JOIN zipcodes z
ON o.zipcode = z.zipcode
```

ID	ZIPCODE	PROVINCE
A		
B	3080	Brabant Flamand
C	1160	Bruxelles-Capitale
D	1160	Bruxelles-Capitale
…		

Smals ICT for society

Spatial join

Spatial join:



observations

ID	GEOM
A	point(x_a, y_a)
B	point(x_b, y_b)
C	point(x_c, y_c)
D	point(x_d, y_d)
…	…

provinces

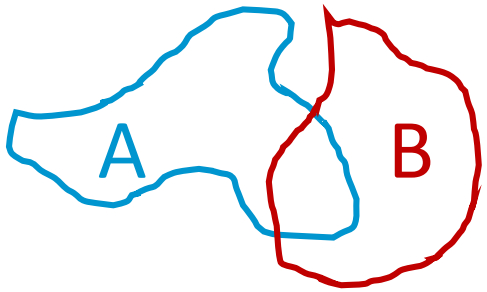
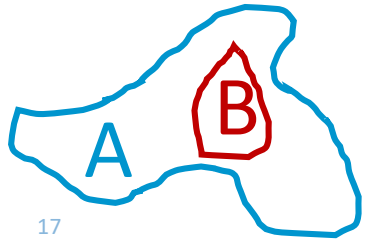
PROVINCE	GEOM
Bruxelles-Capitale	polygon(pt_1, pt_2,…)
Brabant Flamand	polygon(…)
Brabant Wallon	polygon(…)
…	…

```
SELECT o.ID, p.province
FROM observations o
LEFT JOIN provinces p
ON ST_Contains (p.geom, o.geom)
```

ID	PROVINCE
A	Brabant Flamand
B	Brabant Flamand
C	Bruxelles-Capitale
D	Bruxelles-Capitale
…	

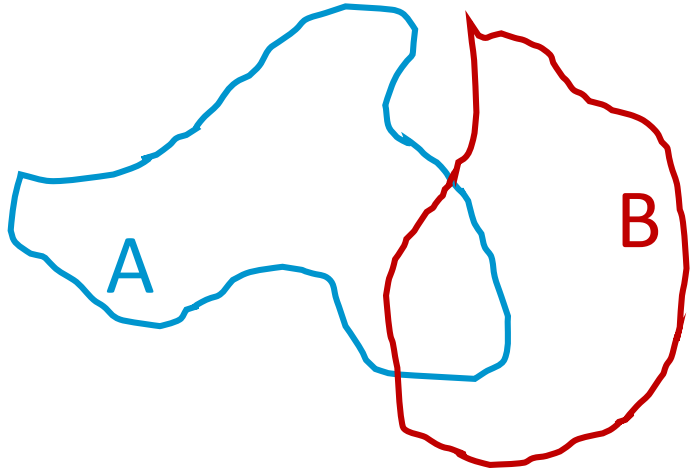
Smals ICT for society

Spatial operators (boolean)

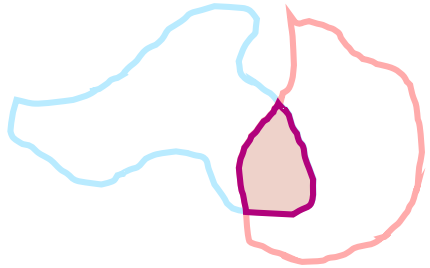
	ST_Intersects(A, B)	ST_Touches(A, B)	ST_Contains(A, B) = ST_Within(B, A)
	✓	✗	✗
	✗	✓	✗
	✓	✗	✓

Smals
ICT for society

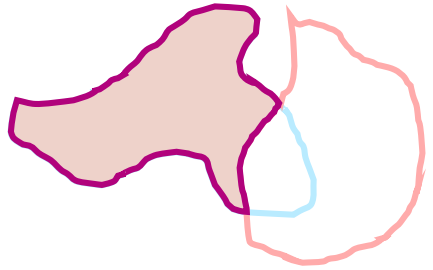
Spatial operators (constructive)



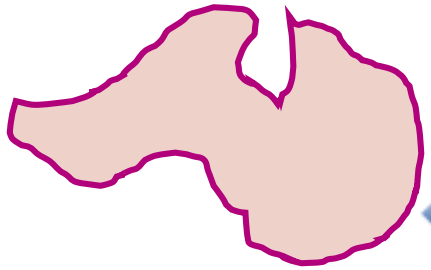
`ST_Intersection(A, B):`



`ST_Difference (A, B):`

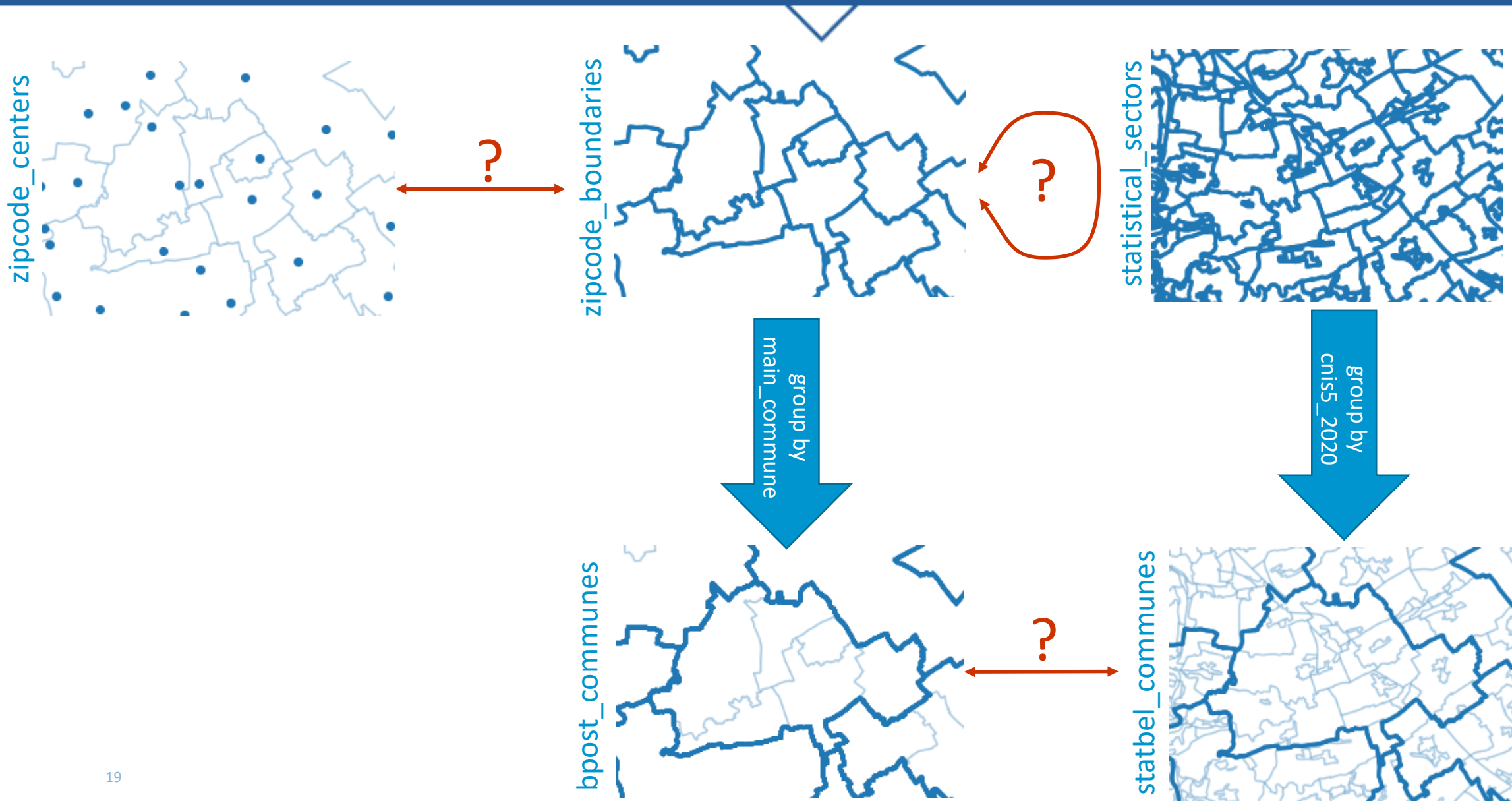


`ST_Union (A, B):`



Smals
ICT for society

Data quality use cases



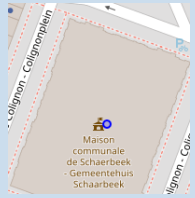
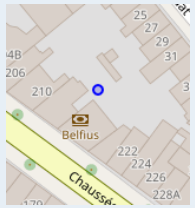
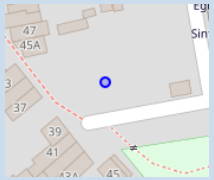
Smals
ICT for society

Spatial join– point vs polygone

Bpost – zipcode boundaries

zipcode	geom
1030	
1060	
1130	
1160	

AdN – centre of zipcodes

zipcode	geom
1030	
1060	
1160	
7300	

Contains

Bp.zipcode	Bp.geom	AdN.zipcode	AdN.geom
1030	…	1030	…
1060	…	1060	…
1130	…	7300	…
1160	…	1160	…

53 “mismatches” (in AdN) :

- 1130 et 3700 Haren as same place
- “8600 Driekapellen” located “Driekapellenstraat, 1541 St. Pt Kap.”
- “6831 Noirfontaine” located “Rue de Noirfontaine, 4400 Flémalle”
- Commune name vs “lieu dit” (Laar, Schriek, Berg, Voort…)

Smals
ICT for society

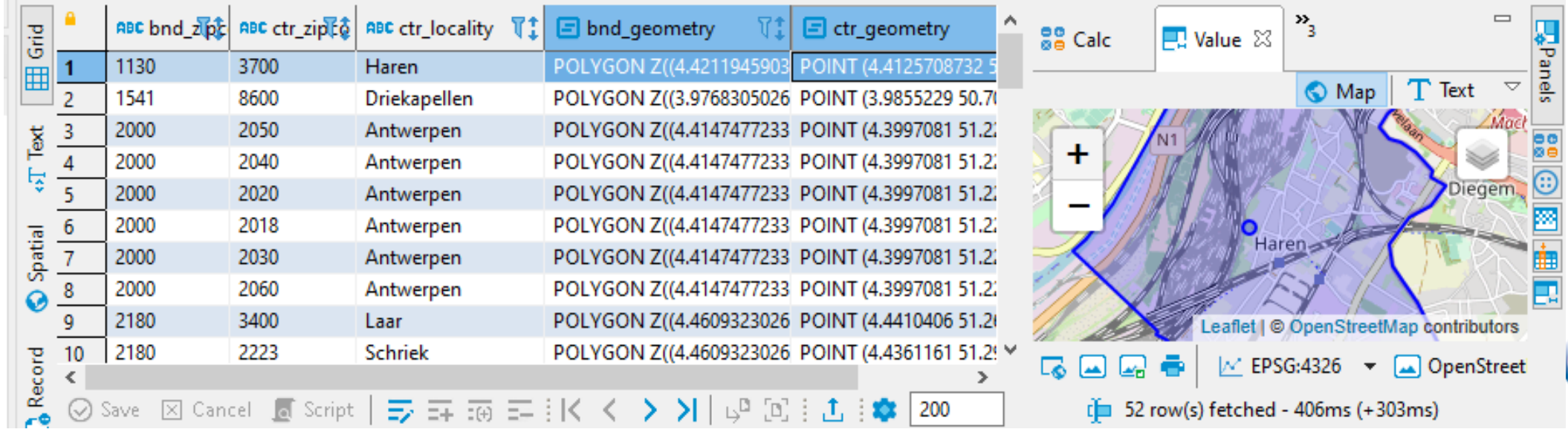
Spatial join: Mismatch zip centers – zip boundaries

```
SELECT bnd.zipcode  AS bnd_zipcode,
       ctr.zipcode  AS ctr_zipcode,
       ctr.locality AS ctr_locality,
       bnd.geometry AS bnd_geometry,
       ctr.geometry AS ctr_geometry
FROM gistest.zipcodes_boundaries bnd
JOIN gistest.zipcodes_centers ctr
       ON ST_Contains(bnd.geometry, ctr.geometry)
WHERE bnd.zipcode != ctr.zipcode
```

COLUMN SELECTION

SPATIAL JOIN

ROW SELECTION



Smals
ICT for society

Spatial join – Internal mismatch @ BPost

COLUMN SELECTION

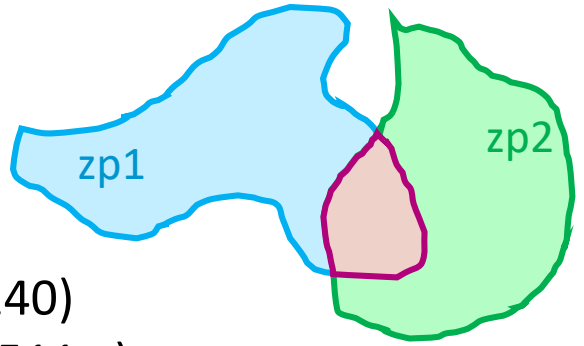
```
SELECT zp1.zipcode, zp2.zipcode,
       zp1.geometry, zp2.geometry,
       ST_Intersection(zp1.geometry, zp2.geometry),
       ST_Area(ST_Intersection(zp1.geometry, zp2.geometry))/ST_Area(zp1.geometry)
              AS common_ratio
```

SPATIAL JOIN

```
FROM gistest.zipcodes_boundaries zp1
JOIN gistest.zipcodes_boundaries zp2
       ON ST_Intersects(zp1.geometry, zp2.geometry)
```

ROW SELECTION

```
WHERE zp1.zipcode < zp2.zipcode
  AND ST_Area(ST_Intersection(zp1.geometry, zp2.geometry))/ST_Area(zp1.geometry)>0.01
```



Only with special codes, we observe :

- Real overlaps (1830 or 1931 vs 1931 ; 7010 vs 7020)
- Missing “holes” for special codes (6075 or 6099 within 6022, 1049 within 1140)
- Several special codes with the same region (1043/1044, 1046/1049, 7510/7511…)

Smals
ICT for society

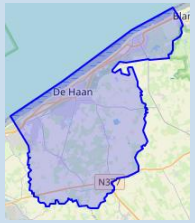
Spatial join– polygone vs polygone

Bpost – commune boundaries

commune	geom
VILVOORDE	
HAM-SUR-HEURE	
DE HAAN	

+

Statbel – commune boundaries

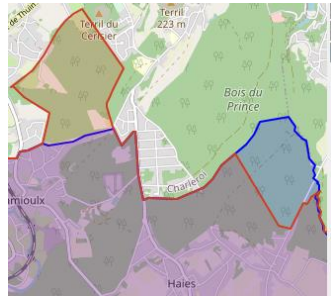
commune	geom
Vilvorde	
Ham-sur-Heure-Nalinnes	
De Haan	

Intersects

AND

$$\frac{Area(Intersection(a,b))}{Area(Union(a,b))}$$

$$\in [50\%, 99.5\%]$$

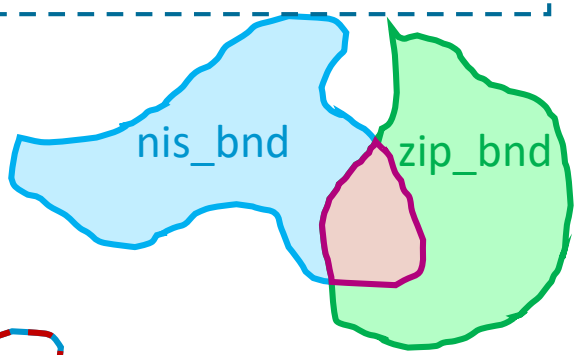


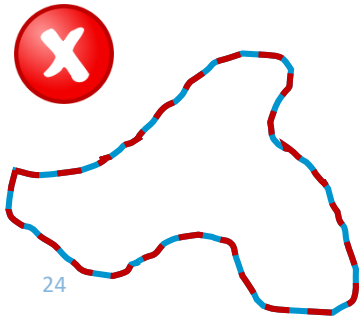
- Coast: with (Statbel) or without (BPost) beach
- Known situations: 1040 = Etterbeek + Europe (Bxl) ; 1050 = Ixelles + Louise-Roosevelt (Bxl)
- Errors?

Smals ICT for society

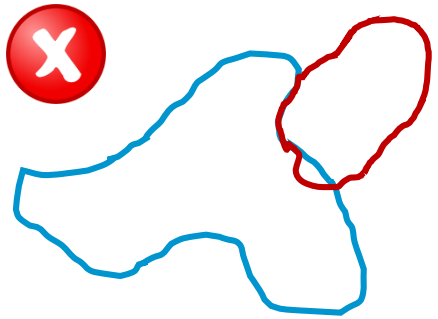
Spatial join: Mismatch NIS – Zip boundaries

```
SELECT cnis5_2020, t_mun_fr, main_commune,                    COLUMN SELECTION
       nis_bnd.geometry   AS nis_geom,
       zip_bnd.geometry   AS zip_geom,
FROM gistest.statbel_communes nis_bnd
JOIN gistest.bpost_communes zip_bnd                           SPATIAL JOIN
       ON ST_Intersects(nis_bnd.geometry, zip_bnd.geometry)
WHERE ST_Area(ST_Intersection(nis_bnd.geometry, zip_bnd.geometry))/   ROW SELECTION
        ST_Area(nis_bnd.geometry) between 0.5 and 0.99
```

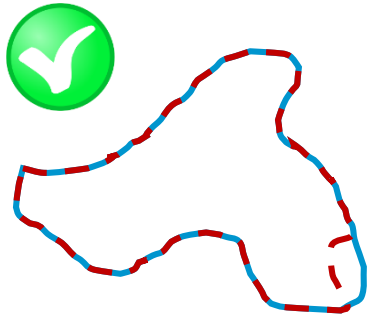




>0.99



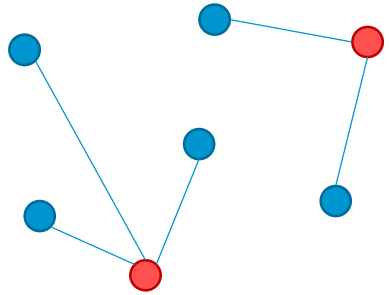
<0.5



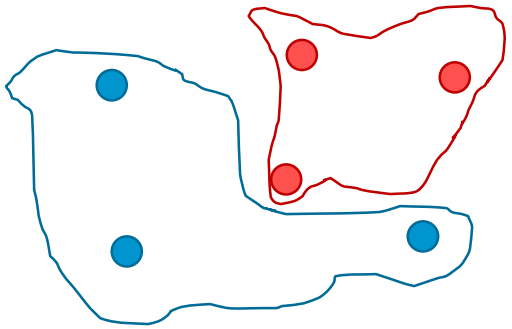
Smals
ICT for society

Spatial join: use cases

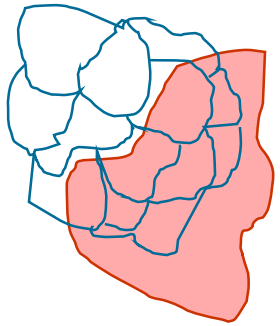
- Join point >< point (ST_ClosestPoint):
  - Closest inspection antenna/inspector home for each worksite (as the crow flies)

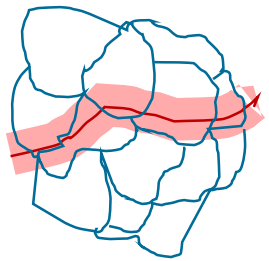


- Join point >< polygon (ST_Contains, ST_Within):
  - Select a region (commune, province, neighborhood, business area) to a set of points (pharmacy, worksites…)
  - Aggregate (count, sum, mean…) points per region



- Join polygon >< polygon (ST_Intersects, ST_Contains, ST_Touches):
  - Compute population of a given area
  - Get sectors within a buffer around a (set of) geometry





Data import

Using SQL

- Using “WTK” (Well Known Text):

```
INSERT INTO app(p_id, geom)
VALUES(2, ST_GeomFromText('POINT(-71.060316 48.432044)', 4326));
```

- Using ST_MakePoint:

```
INSERT INTO app(p_id, geom)
VALUES(2, ST_SetSRID(ST_MakePoint(-71.060316, 48.432044), 4326));
```

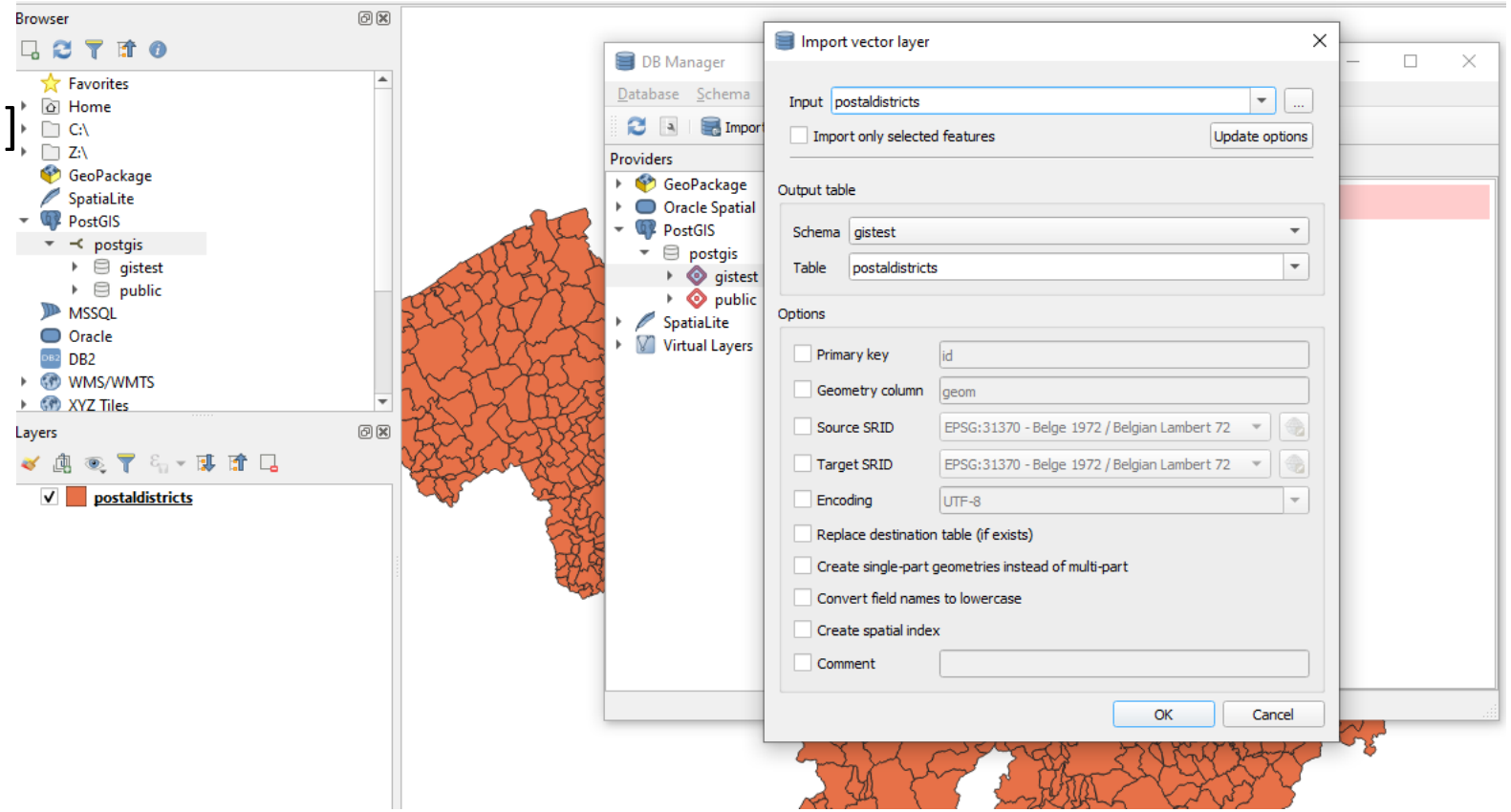
Import shapefiles in Python

```
import geopandas as gpd   # To handle shapefile (or geojson)
import sqlalchemly as sa  # To push data on Postgres/PostGIS

# Download https://bgu.bpost.be/assets/9738c7c0-5255-11ea-8895-34e12d0f0423_x-
shapefile_3812.zip
# to "zipcode_boundaries.zip"

# Dataload
zipcode_boundaries = gpd.read_file(f"zip://zipcode_boundaries.zip/3812")
zipcode_boundaries = zipcode_boundaries.to_crs("epsg:4326")

# Create DB connection
engine = sa.create_engine(f"postgresql+psycopg2://{usr}:{pwd}@{host}:{port}/{db}")

# Push data into DB
zipcode_boundaries.to_postgis(name="zipcode_boundaries",
                              con=engine,
                              if_exists="replace",
                              schema="gistest")
```

Smals
ICT for society

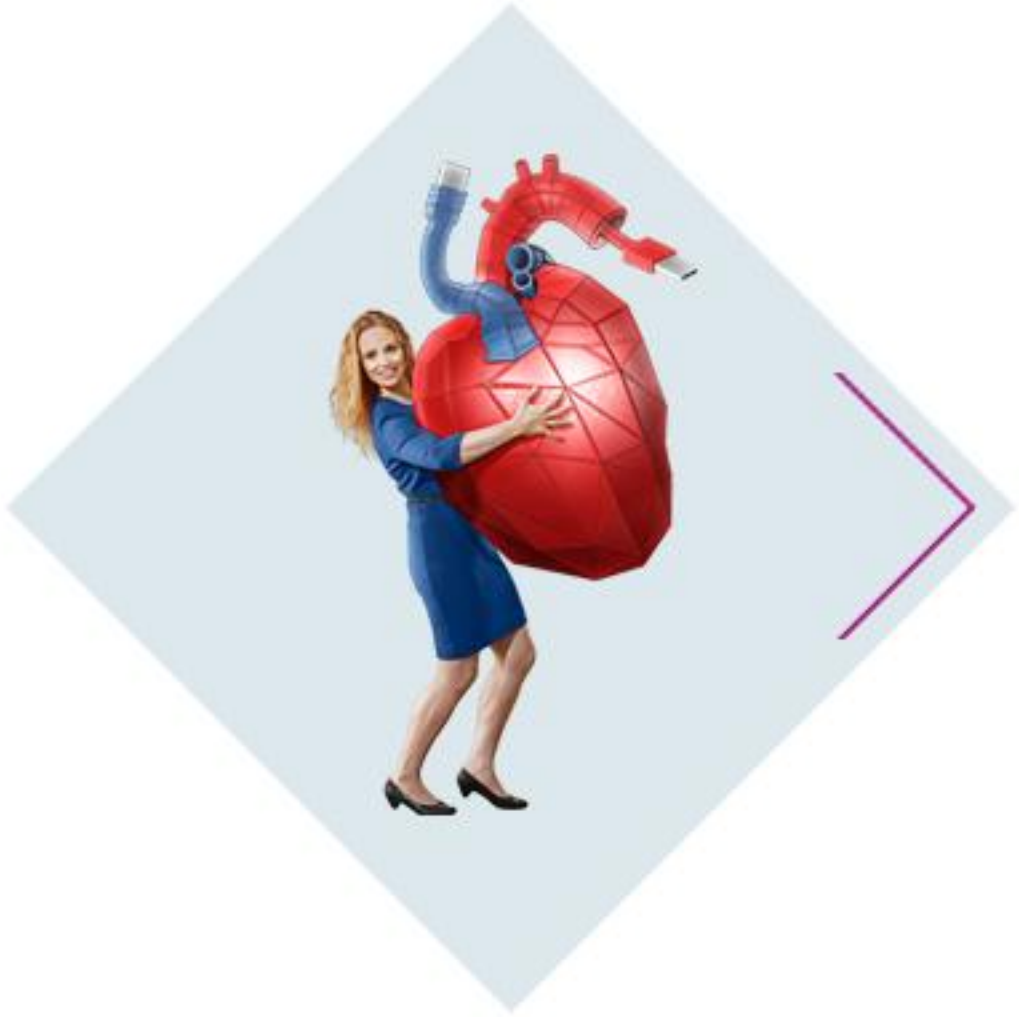
Import shapefiles via QGIS

- On www.geo.be, search “Cantons postaux”
- Select dataset “Cantons postaux”, download “projection = 31370”
- Open QGIS, drag&drop zipfile to “Layers” tab
- In Browser > PostGIS > new connection
- Database > DB Manager
- Select PostGIS > postgis > [schema]
- Import Layer/File…



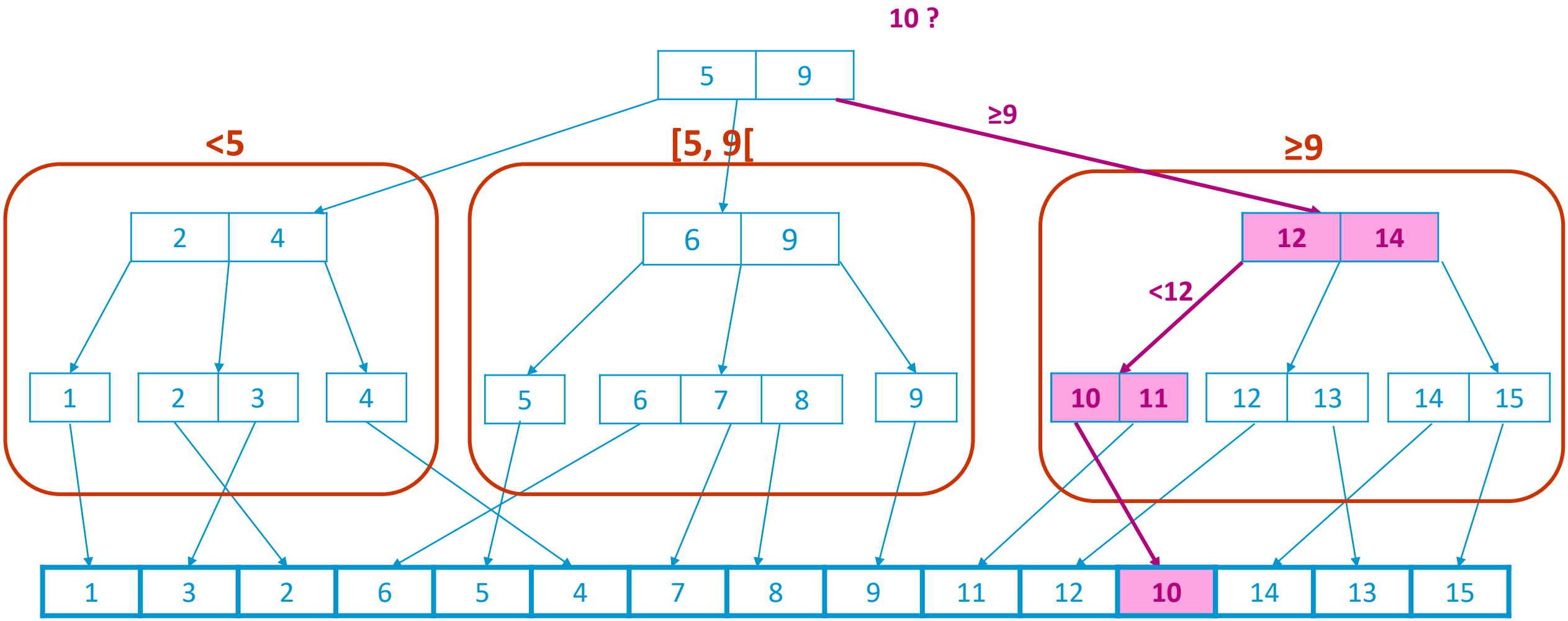
Command line

```
export PGPASSWORD=mydatabasepassword

ogr2ogr \
  -nln zipcode_boundaries \
  -nlt PROMOTE_TO_MULTI \
  -lco GEOMETRY_NAME=geom \
  -lco FID=gid \
  -lco PRECISION=NO \
  Pg:'dbname=gistest host=localhost user=pramsey port=5432' \
  zipcode_boundaries.shp

Alternative:
shp2pgsql -D -I -s 3812 zipcode_boundaries.shp zipcode_boundaries \
     | psql dbname=gistest user=postgres host=localhost
```



Spatial index

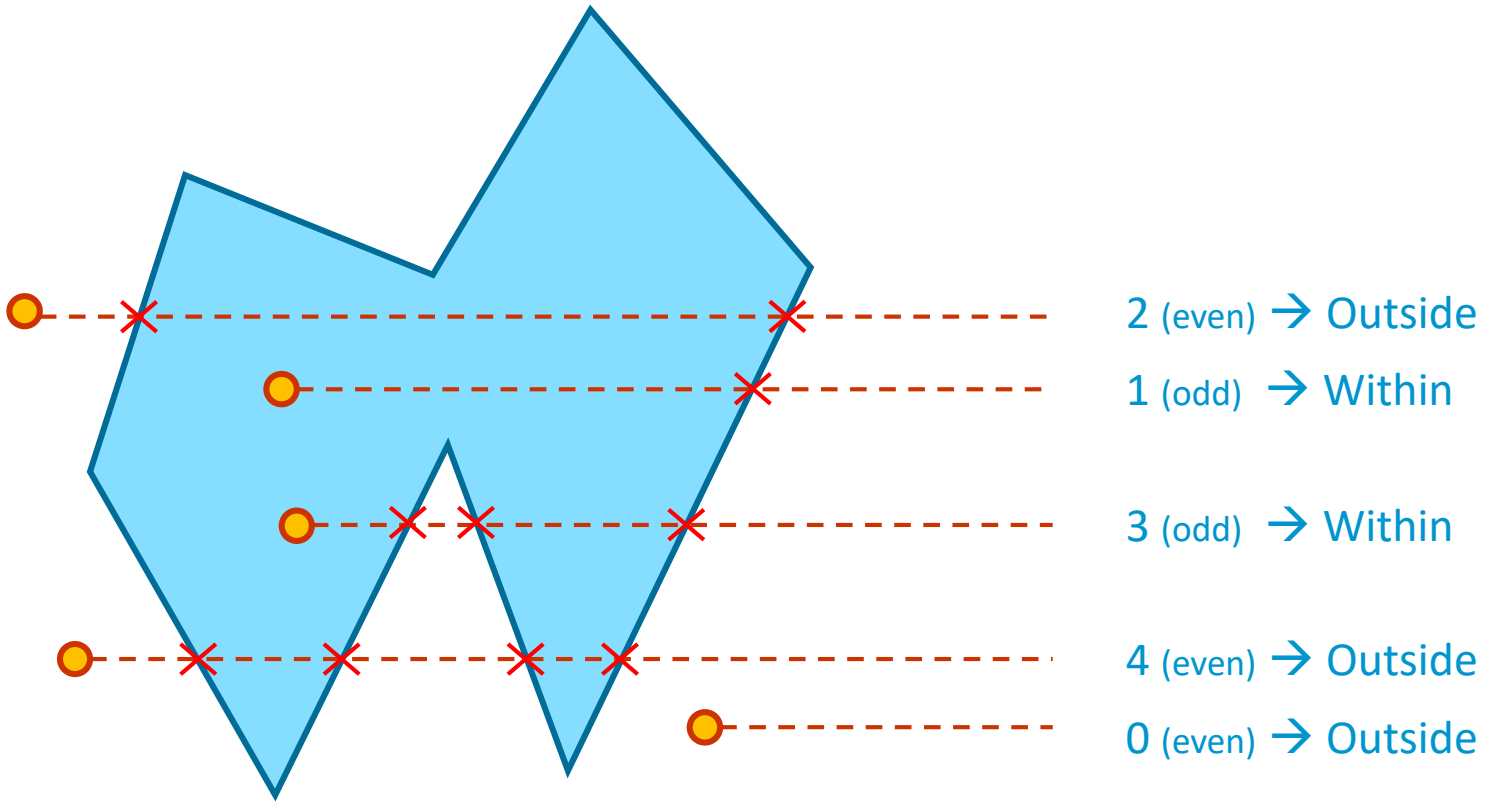
Basic indexing: B-Tree index



Smals
ICT for society

Spatial index

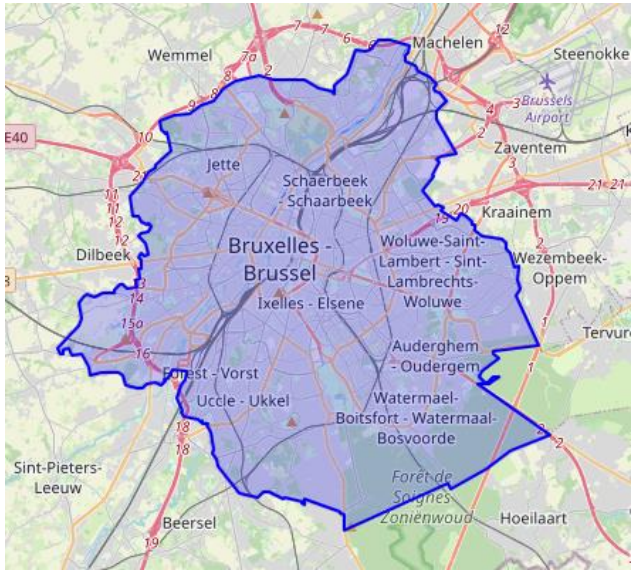
- With geometrical shapes, “join” question is “Is a point within a polygon?”
- Requires a complex geometric operation : O (nb points(polygon))

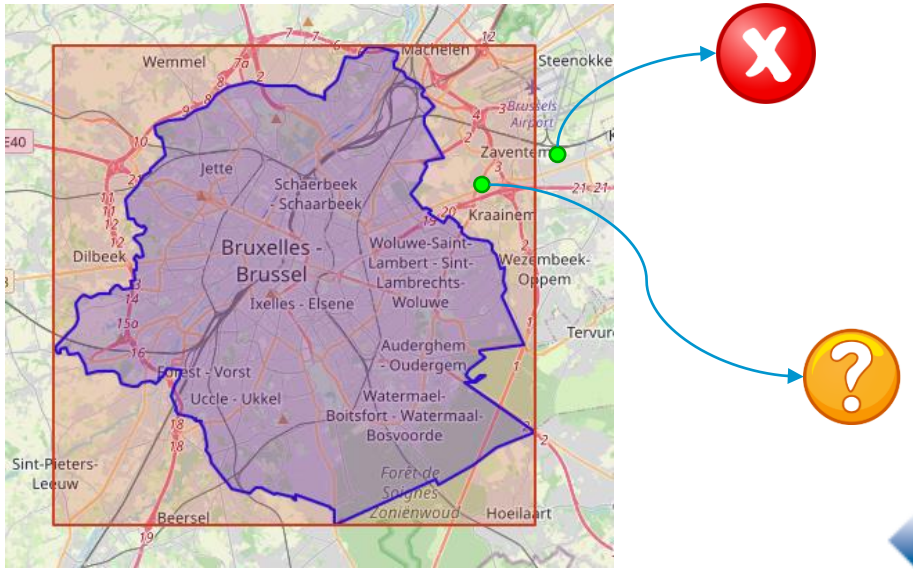


Smals
ICT for society

Spatial index

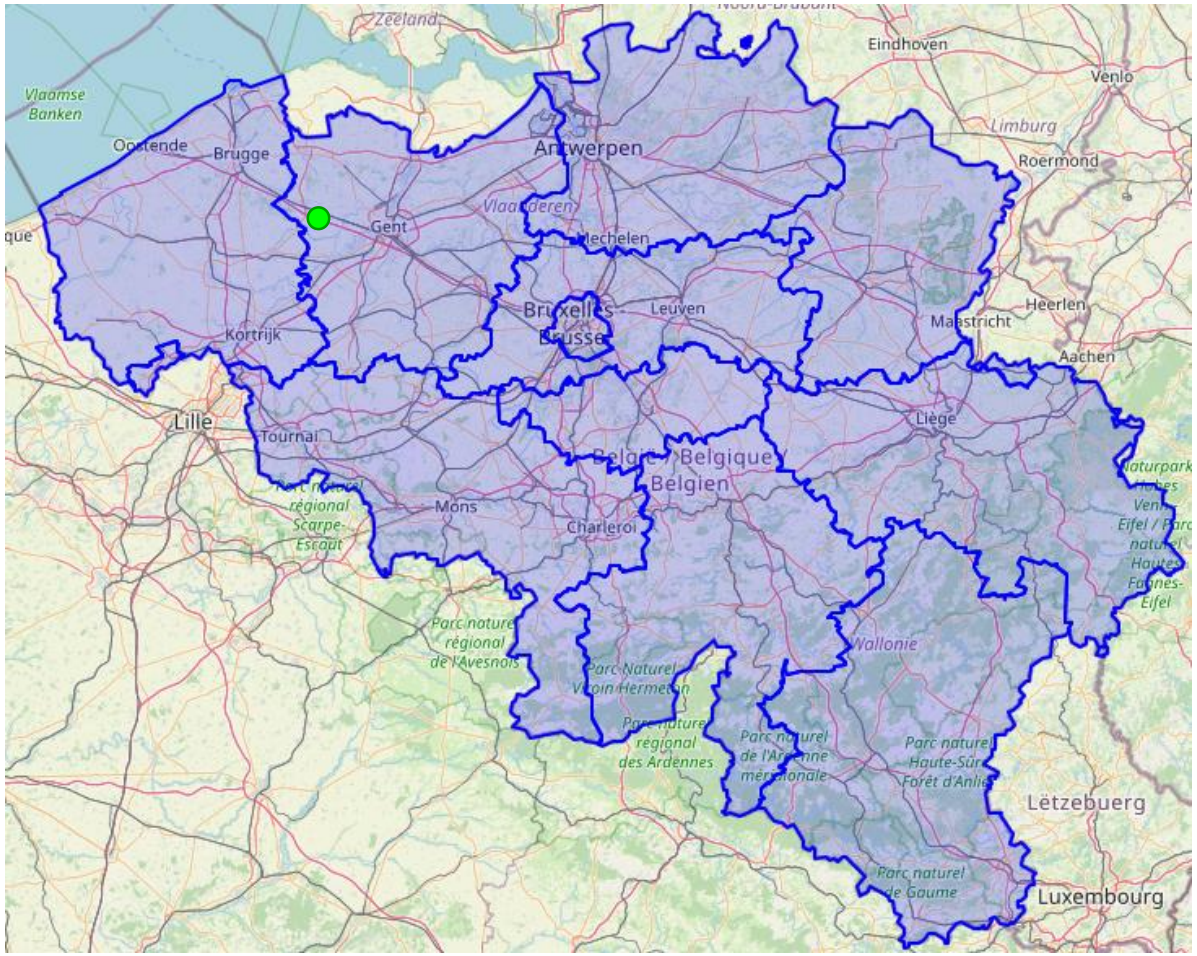
- Without indexing, spatial join between a list of N points and a list of M polygons requires NxM “is within a polygon” operations
- But: “Is a point within an horizontal rectangle?”: very fast (using B-tree)
- Basic concept for spatial index: first consider bounding boxes (or envelope) for polygons, and then exact polygon only for those matching with bounding box



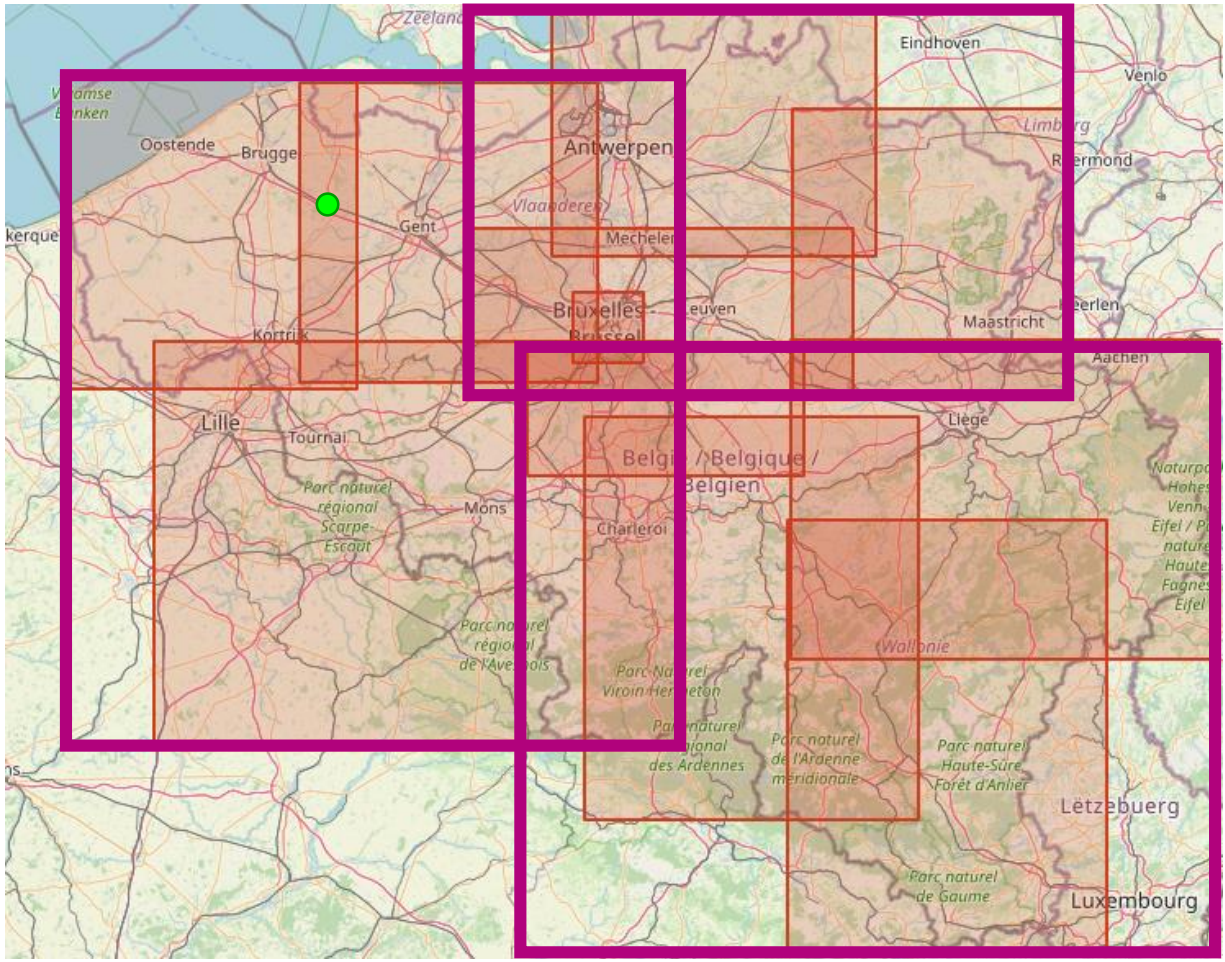


Smals ICT for society

R-tree



11 x “is ● within polygon”



11 x “is ● within rectangle” + 2 x “is ● within polygon”

(3 + 3) x “is ● within rectangle” + 2 x “is ● within polygon”

R-tree: performance

```
DROP INDEX IF EXISTS gistest.idx_zipcodes_boundaries_geometry;
DROP INDEX IF EXISTS gistest.idx_zipcodes_centers_geometry;
```

```
SELECT *
FROM gistest.zipcodes_boundaries bnd
JOIN gistest.zipcodes_centers ctr
ON ST_Contains(bnd.geometry, ctr.geometry);
```

```
SELECT *
FROM gistest.statistical_sectors bnd
JOIN gistest.zipcodes_centers ctr
ON ST_Contains(bnd.geometry, ctr.geometry);
```

```
CREATE INDEX idx_zipcodes_boundaries_geometry
ON gistest.zipcodes_boundaries
USING GIST (geometry);
```

```
CREATE INDEX idx_zipcodes_centers_geometry
ON gistest.zipcodes_centers
USING GIST (geometry);
```

	Zipcode	Statistical sectors
Number of geometries	1 286 x 2 575	19 794 x 2 575
Number of points	1,14M	3,5M
Avg. points per geom.	900	180
Timing without index	9,3 seconds	48 seconds
Timing with index	0,45 seconds	0,3 seconds
Gain	X 20	X 160

Index: a few remarks

- Works for comparison (polygon, point), but also (polygon, polygon)
- Index "lost" if any operation is performed before join: `ST_Transform, ST_Buffer,…`
  - Either build the appropriate column and index it
  - Or use "a.geom && b.geom" = true if bounding boxes intersect
- Alternatives to GIST: SPGIST, BRIN…
- After some data updates: **`VACUUM ANALYZE <table_name>`**
- As for "classical" PosgtreSQL: indexes takes space and time.
  For statistical sectors (~20K records, 3.5M points): ~ 1MB - 0,5sec.

Conclusion

Conclusion

- Not more difficult than “classical RDBMS”, but with powerful additional functionalities
- Added value:
  - To just plot simple points (lat, lon) onto a map → No added value, PostgreSQL + leaflet.js (or other) is enough
  - Distance computation (even for simple points)
  - Projection handling
  - Line, Polygon handling: spatial join, aggregation/combination…
  - Buffer: area 1km away from Seveso sites, river, pipeline, highway…
- Alternatives:
  - Oracle Spatial , Microsoft SQL Server Spatial
  - Similar (but less advanced) functionalities in ElasticSearch (geoshapes)

Conclusion – Use cases

- Clear potential for many eGov applications (see first webinar for inspiration)
- Can be used on Smals PostgreSQL DBs (ask DBAs… conditions TBD)… but no project so far?
- Known usages:
  - BestAddress (BOSA)
  - NGI/IGN & partners
  - (OpenStreetMap)
- Potential usages:
  - Quality checks
  - Enrichment: assign a zone (commune, province, business zone…) to a set of points
  - Selection: select addresses within a given area (i.e., companies within a known neighborhood)
  - Disaster management: area 10 km away from potential risk
  - Impact analysis: population living in a given area

From your side?
Happy to get feedback and discuss potential use cases!

Vandy Berten
Smals Research

Smals, ICT for society
02 787 57 11
Fonsnylaan 20 / Avenue Fonsny 20
1060 Brussel / 1060 Bruxelles