Store and Process Geodata in a Relational Database

Andreas Petutschnig

Maptime Salzburg – 12.12.2018

University of Salzburg, dept. of Geoinformatics – Z_GIS
andreas.petutschnig@sbg.ac.at

UNIVERSITÄT SALZBURG #maptime!

Introduction

Mission statement

- Target group: spatial data users
- Introduce you to the concept of spatial databases
- Long term goal: Shapefile must die!
- Encourage you to try PostGIS

Databases and datatypes

Why use DBs?

- Reliable
- Multi user access/management
- Indexing can speed up processes
- Complex queries are easy to store/repeat/extend
- Lots of possibilities for automation

Why use spatial DBs?

- Spatial functions $\rightarrow$ `st_*` prefix
- Search based on location
- Interpolation
- Geometry calculations
- Spatial indexing
- Spatial joins

Spatial datatypes

- Vector and raster support
- `geography`
  - Calculations on a sphere
  - Few PostGIS functions
  - Longer calculation times
- `geometry`
  - Calculations on a plane
  - More (all?) PostGIS functions
  - Shorter calculation times

Practical examples

Generate sample data – Random Points

```
CREATE TABLE public.testpoint (
    id serial PRIMARY KEY,
    geom geometry);
```

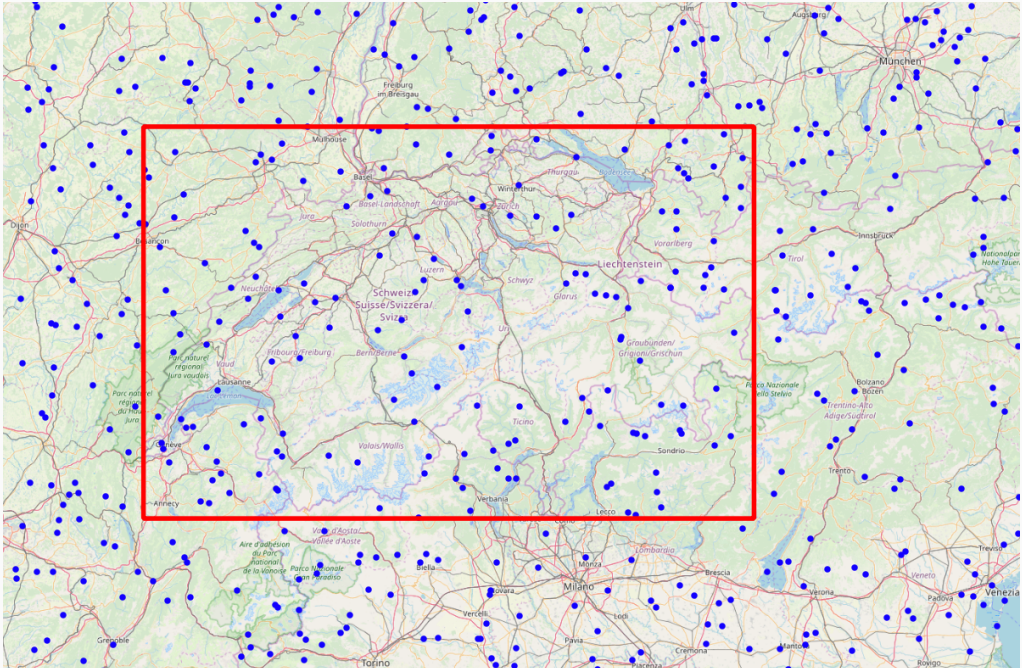
```
WITH x AS
    (select generate_series (1,1000),
     st_setSRID (st_makepoint (
        (random () * 10)::float + 5,
        (random () * 5)::float + 45),
     4326) geom)
INSERT INTO testpoint (geom) SELECT geom FROM x;
```

Generate sample data – Polygon

```
CREATE TABLE public.testpolygon (
    id serial PRIMARY KEY,
    geom geometry);
```

```
INSERT INTO testpolygon (geom) VALUES (
    st_setSRID (st_geometryfromtext (
    'POLYGON((5.96 47.81, 10.49 47.81,
    10.49 45.82, 5.96 45.82, 5.96 47.81))'
), 4326));
```

Generate sample data

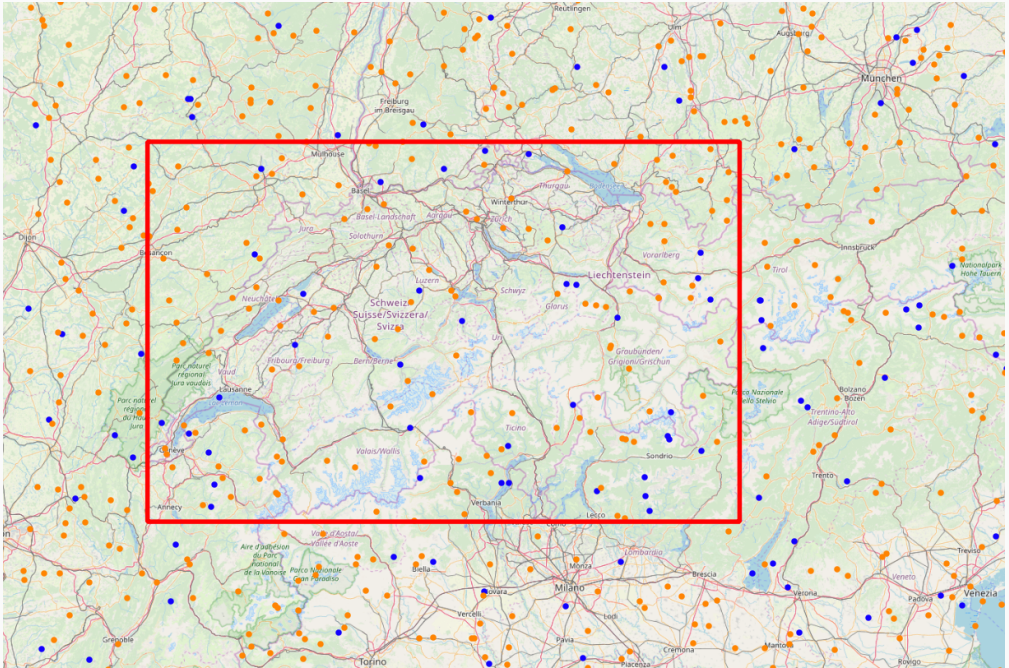


Non spatial query

```
SELECT count (*) FROM testpoint
    WHERE id > 200;
```

Result
800

Non spatial query

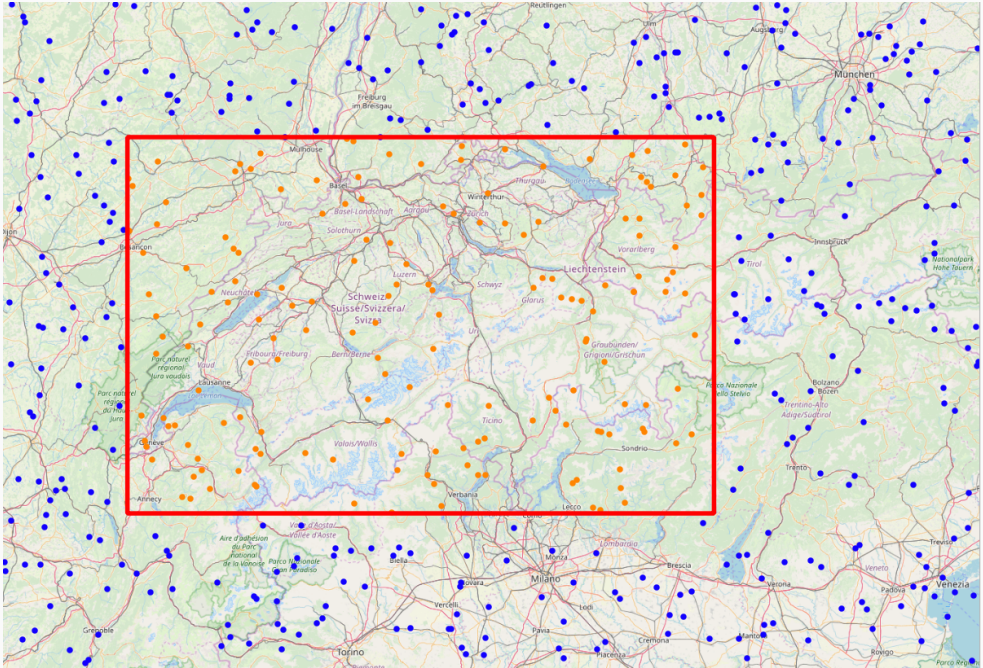


Spatial query

```
SELECT count (*) FROM testpoint pt
    JOIN testpolygon poly
    ON st_contains (poly.geom, pt.geom);
```

Result
169

Spatial query

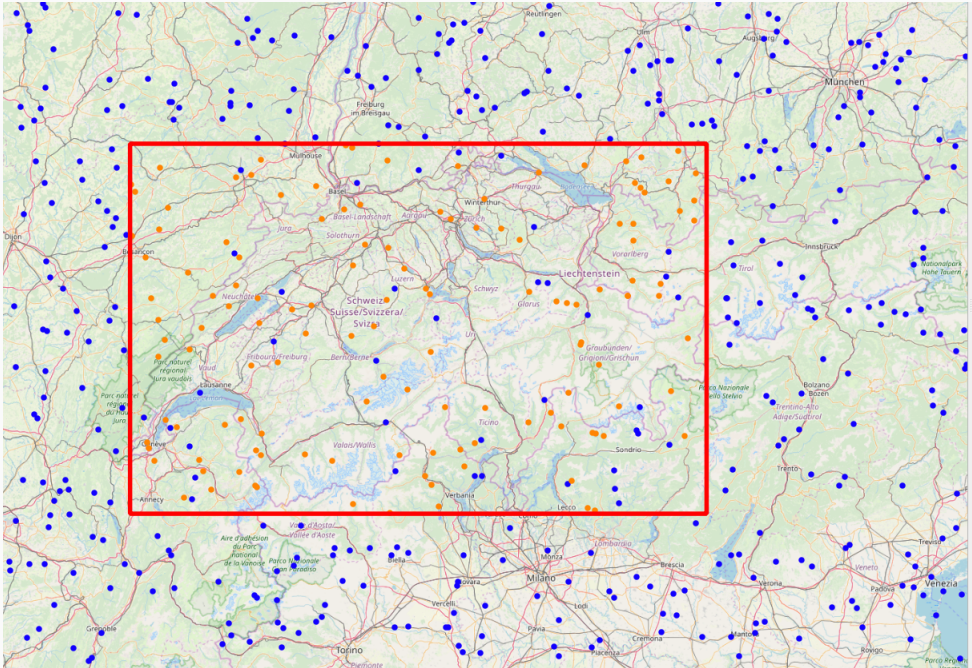


Combined query

```
SELECT count (*) FROM testpoint pt
    JOIN testpolygon poly
    ON st_contains (poly.geom, pt.geom)
    WHERE pt.id > 200;
```

Result
131

Combined query



Summary

Conclusions

- Spatial DBs can make your life much easier
- Try this at home

Useful links

Software

- PostgreSQL download
- PostGIS download

Docs

- PostgreSQL documentation
- PostGIS documentation

Other links

- GIS Stackexchange
- My website

Store and Process Geodata in a Relational Database

Andreas Petutschnig

Maptime Salzburg – 12.12.2018

University of Salzburg, dept. of Geoinformatics – Z_GIS
andreas.petutschnig@sbg.ac.at

UNIVERSITÄT SALZBURG #moptime!