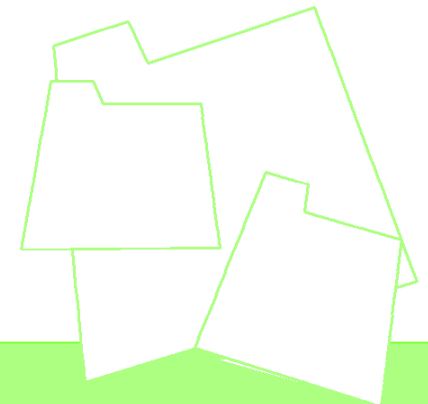


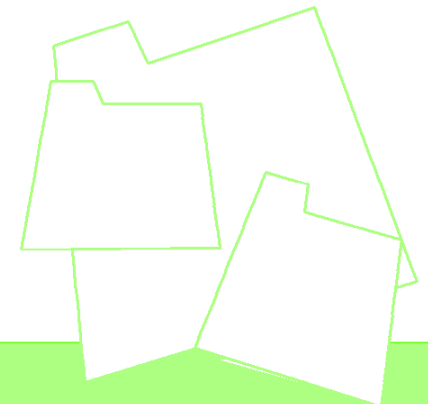
# Geoprozessieren mit PostgreSQL/PostGIS und R

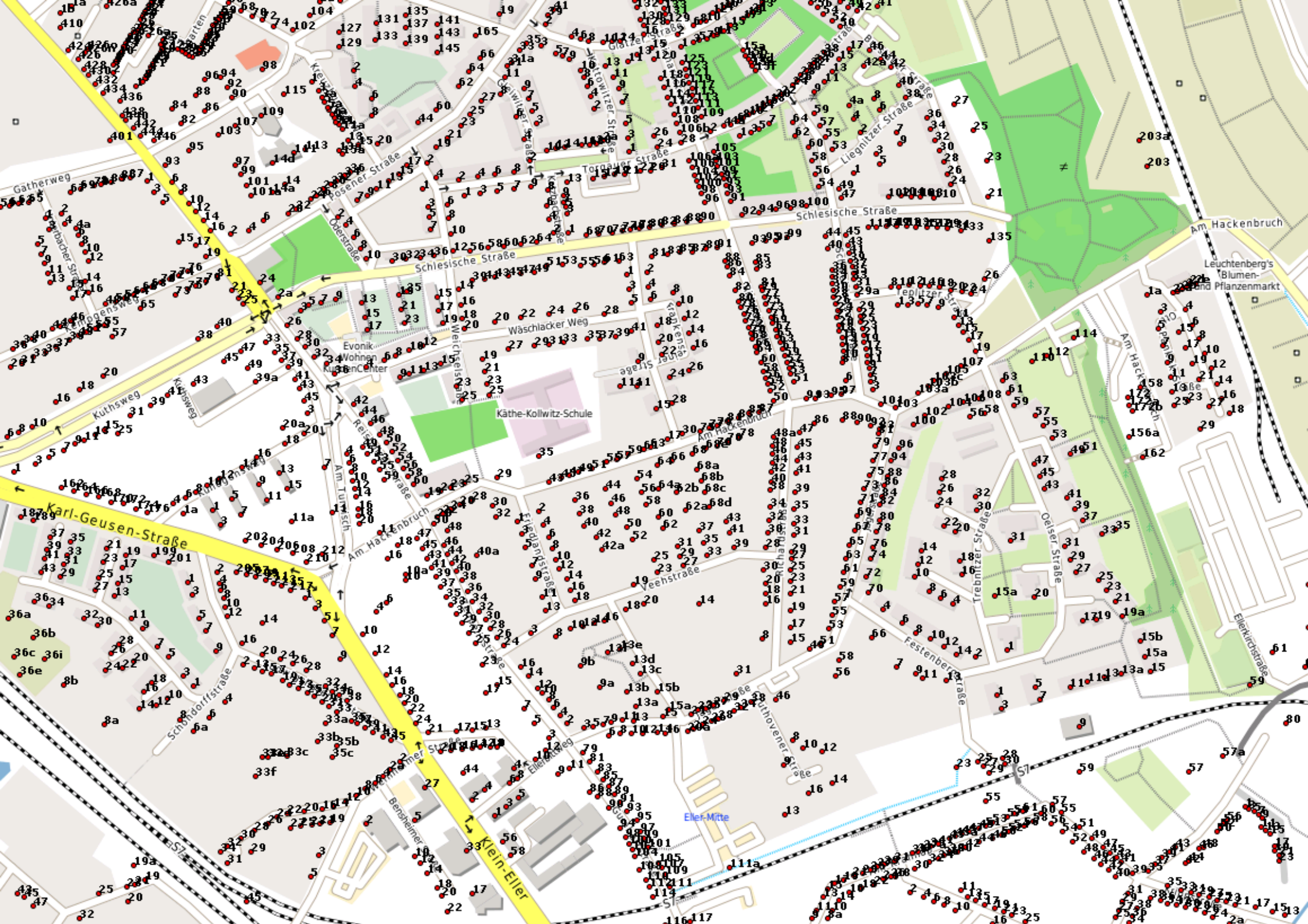
Automatisierte Ableitung einer Gebietsgliederung  
aus Punktdaten



## Typische Ausgangsdaten

id	Stadtteil	Straschl	Haus_Nr	Geometrie	Attribute
1	99	32501	11a	POINT	Wahlbezirk
2	98	29333	51	POINT	Baublock
...	...	...	...	...	...
9999	97	4711	31	POINT	plz
...	...	...	...	...	...

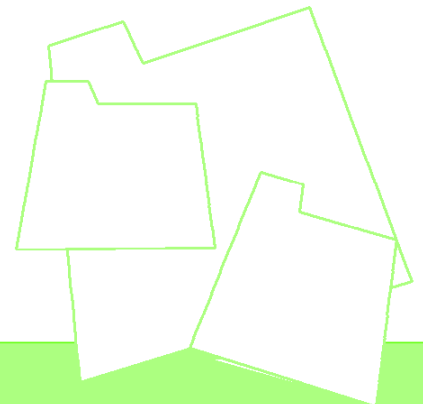




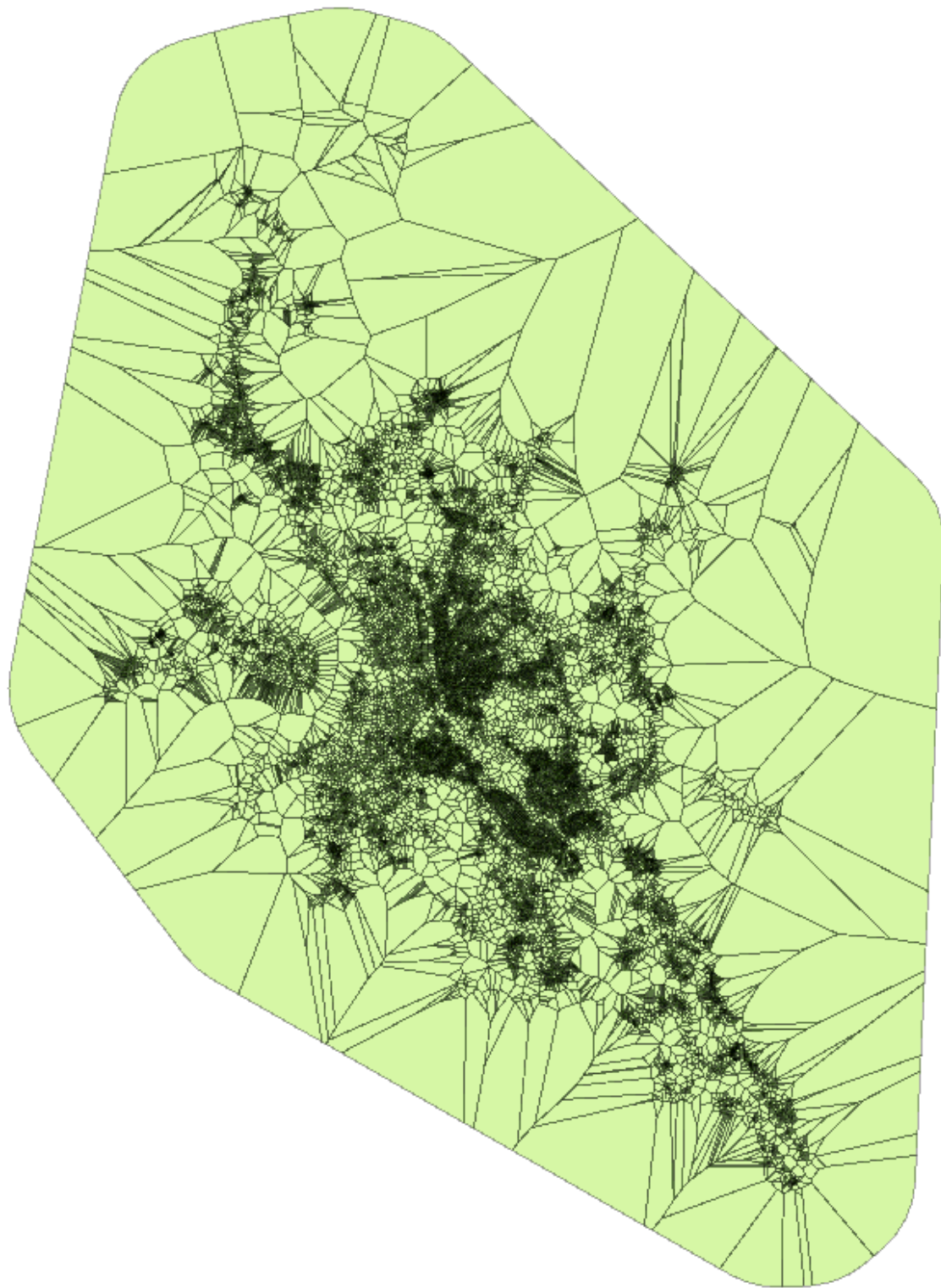
Die Beziehung der Punkte zueinander soll Flächenhaft dargestellt werden!

Eine Konvexe Hülle umspannt die Punktmenge ohne Auswertung der Nachbarschaftsbeziehungen!

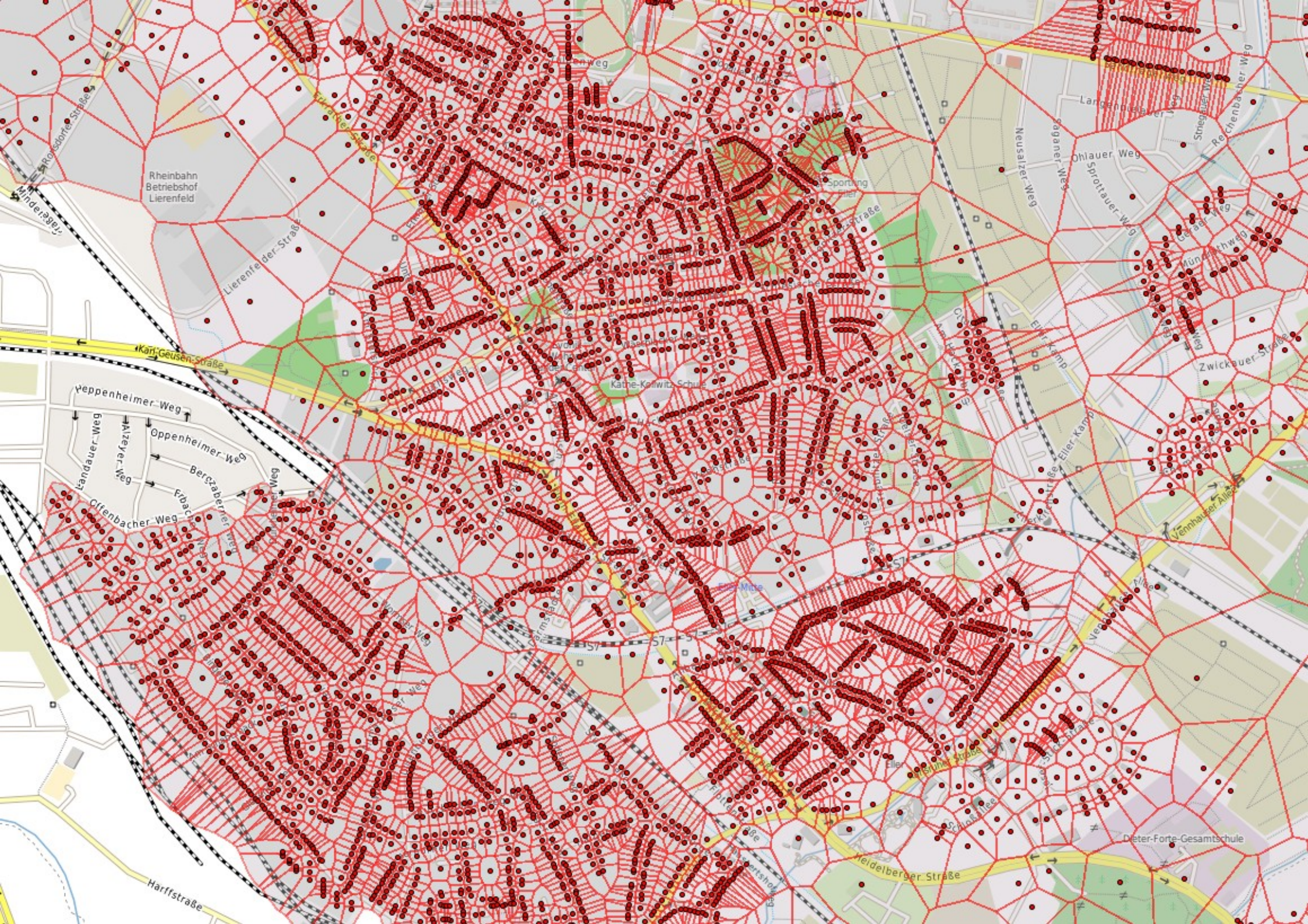
Mit dem Voronoi-Diagramm werden Regionen erzeugt in deren Zentren jeweils ein Punkt ist!











Rheinbahn  
Betriebshof  
Lierenfeld

Kathe-Kolwitz Schule

Dieter-Forte-Gesamtschule

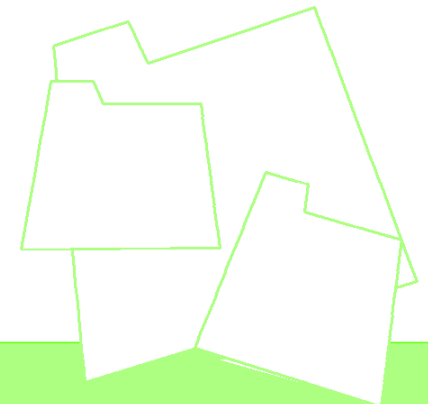
Map labels include: Minderweg, Rosdorfer Straße, Lierenfelder-Strasse, Karl-Geusen-Straße, Oppenheimer-Weg, Bergzaberner-Weg, Offenbacher Weg, Harffstraße, Heidelberger Straße, and various other street names like Ohlauer Weg, Spröttauer-Weg, and Zwickauer-Straße.



Für die Berechnung eines Voronoi-Diagramm muss der  
Funktionumfang von PostgreSQL/PostGIS erweitert  
werden!

R hat diese Rechenfunktion  
und

R ist eine Sprache  
und Umgebung für statistische Datenbearbeitung!

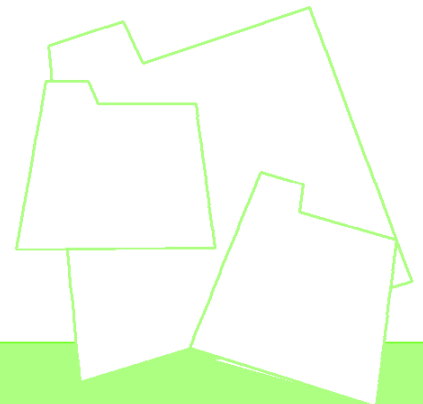


PL/R ist eine Spracherweiterung um mit PostgreSQL  
Funktionen aus R zu nutzen!

(PL/R - R Procedural Language for PostgreSQL)

Ein Tutorial für die ergänzenden Funktionen finden sich mit  
diesem Link:

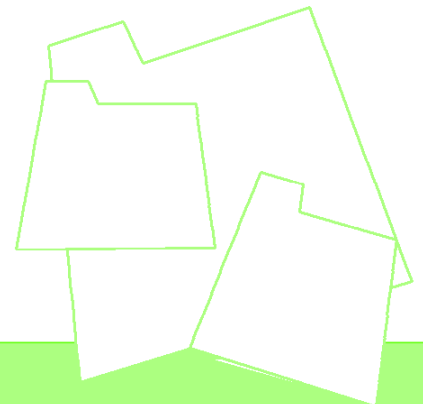
[http://www.bostongis.com/PrinterFriendly.aspx?content\\_name=postgresql\\_plr\\_tut02](http://www.bostongis.com/PrinterFriendly.aspx?content_name=postgresql_plr_tut02)





## Die Übergabe der Daten an R:

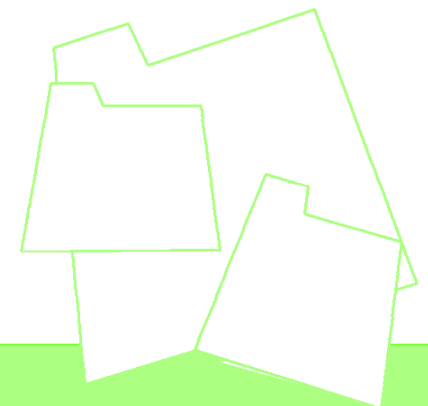
```
INSERT INTO adressen_voronoi(gid, fk_stadtteil, fk_strschl, hs_nr, the_geom)
  (SELECT id, fk_stadtteil, fk_strschl, hs_nr, fk_sozialraum, polygon
    FROM voronoi
    (
      '(SELECT gid, the_geom FROM adressen WHERE fk_attribut LIKE "%abc%"
      )AS subselect', 'subselect.the_geom', 'subselect.gid'
    ) JOIN adressen ON (id = gid)
  );
```



Für eine Gebietsgliederung könnten die Daten ...

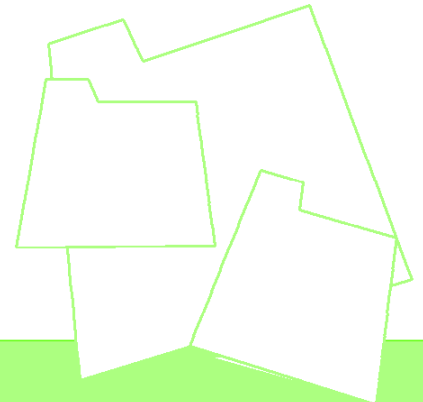
kwbez	kwbez_name	straschl	strasse	von_u_hs	bis_u_hs	von_g_hs	bis_g_hs	stibez
1	Mitte	33333	Nordstraße	1	999	2	998	1999
1	Mitte	12121	Ostallee			44	998	1888
1	Mitte	9999	Wall	1	3			1888
1	Mitte	9999	Wall	3a	999			1999
...	...	...	...	...	...			...
2	Nord	12345	Im Tal	1	999			2435
2	Nord	12345	Im Tal			2	44	2433
2	Nord	12345	Im Tal			46	998	2432
...	...	...	...	...	...			...

daher kommen.

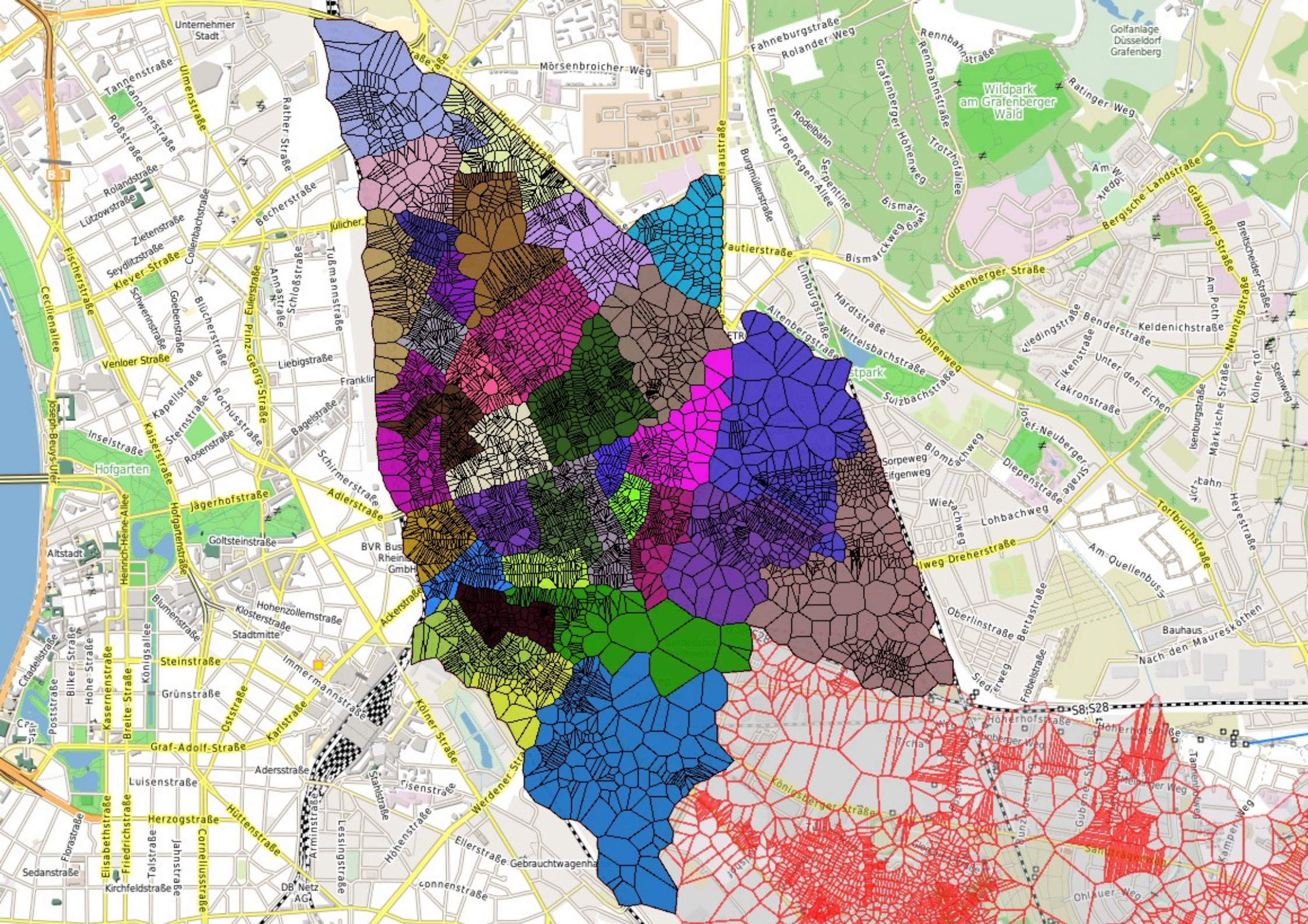




```
(SELECT c.fk_strschl, c.hs_nr, c.stibez, c.kwbez FROM adr,
 (SELECT a.fk_strschl, a.hs_nr, b.stibez, b.kwbez FROM adr a, kwbez b
  WHERE a.fk_strschl = b.fk_strschl AND
    (
      (
        (round(to_number(a.hs_nr||'.0', '999D9'),0) >=
          round(to_number(b.von_u_hs||'.0', '999D9'),0) AND
        round(to_number(a.hs_nr||'.0', '999D9'),0) <=
          round(to_number(b.bis_u_hs||'.0', '999D9'),0) AND
        (round(to_number(a.hs_nr||'.0', '999D9') / 2,1)::text) LIKE '%.5')
      OR
        (round(to_number(a.hs_nr||'.0', '999D9'),0) >=
          round(to_number(b.von_g_hs||'.0', '999D9'),0) AND
        round(to_number(a.hs_nr||'.0', '999D9'),0) <=
          round(to_number(b.bis_g_hs||'.0', '999D9'),0) AND
        (round(to_number(a.hs_nr||'.0', '999D9') / 2,1)::text) LIKE '%.0')
      )
    )
  ) AS c
WHERE adr.fk_strschl||'.'||adr.hs_nr = c.fk_strschl||'.'||c.hs_nr);
```

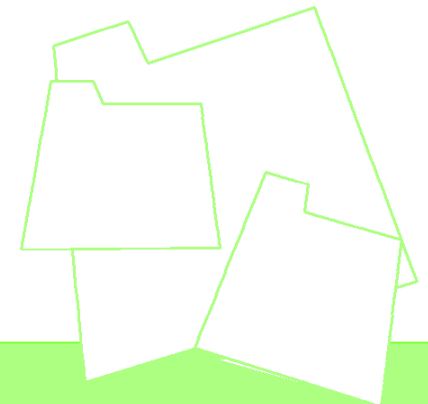








- . Regionen vereinigen (ST\_Union)
- . Flächen stutzen (ST\_Intersection)
- . Fläche für die Löcher erstellen (ST\_Difference)
- . Fläche (aus ST\_Difference) aufbrechen (ST\_Dump)
- . Flächen (aus ST\_Dump) zuordnen und vereinigen (ST\_Union)



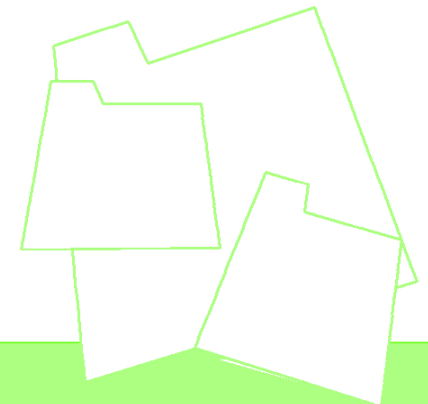






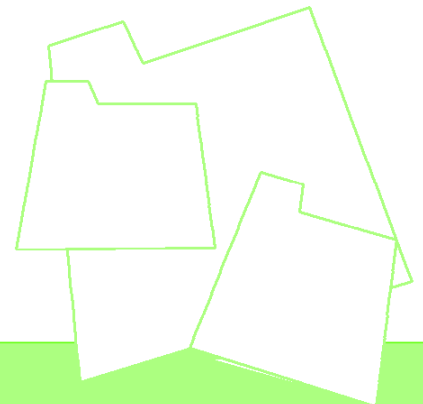
## Regionen vereinigen

```
INSERT INTO / UPDATE ...  
  (SELECT a.wbez, b.stibez, ST_Multi(ST_Union(b.the_geom)) FROM  
    adressen b JOIN wahlbezirke_fossgis_2010 a ON ((b.stibez)::int = a.stbez)  
  WHERE  
    b.wbez >= 101 AND b.kwbez <= 104  
  GROUP BY b.stibez, a.wbez  
  )  
;
```



## Flächen stutzen (Referenzfläche erforderlich)

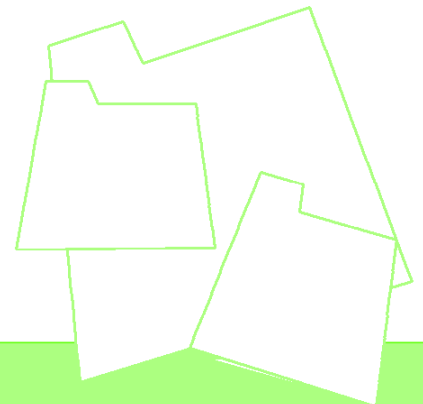
```
UPDATE stimmbezirke c SET the_geom = foo.the_geom FROM
  (SELECT a.gid, ST_Multi(ST_Intersection(a.the_geom ,ST_Union(b.the_geom)))
                                     AS the_geom
   FROM stimmbezirke a, bezirk_fossgis_2010 b
   GROUP BY a.the_geom, a.gid
  ) foo
WHERE c.gid = foo.gid;
```



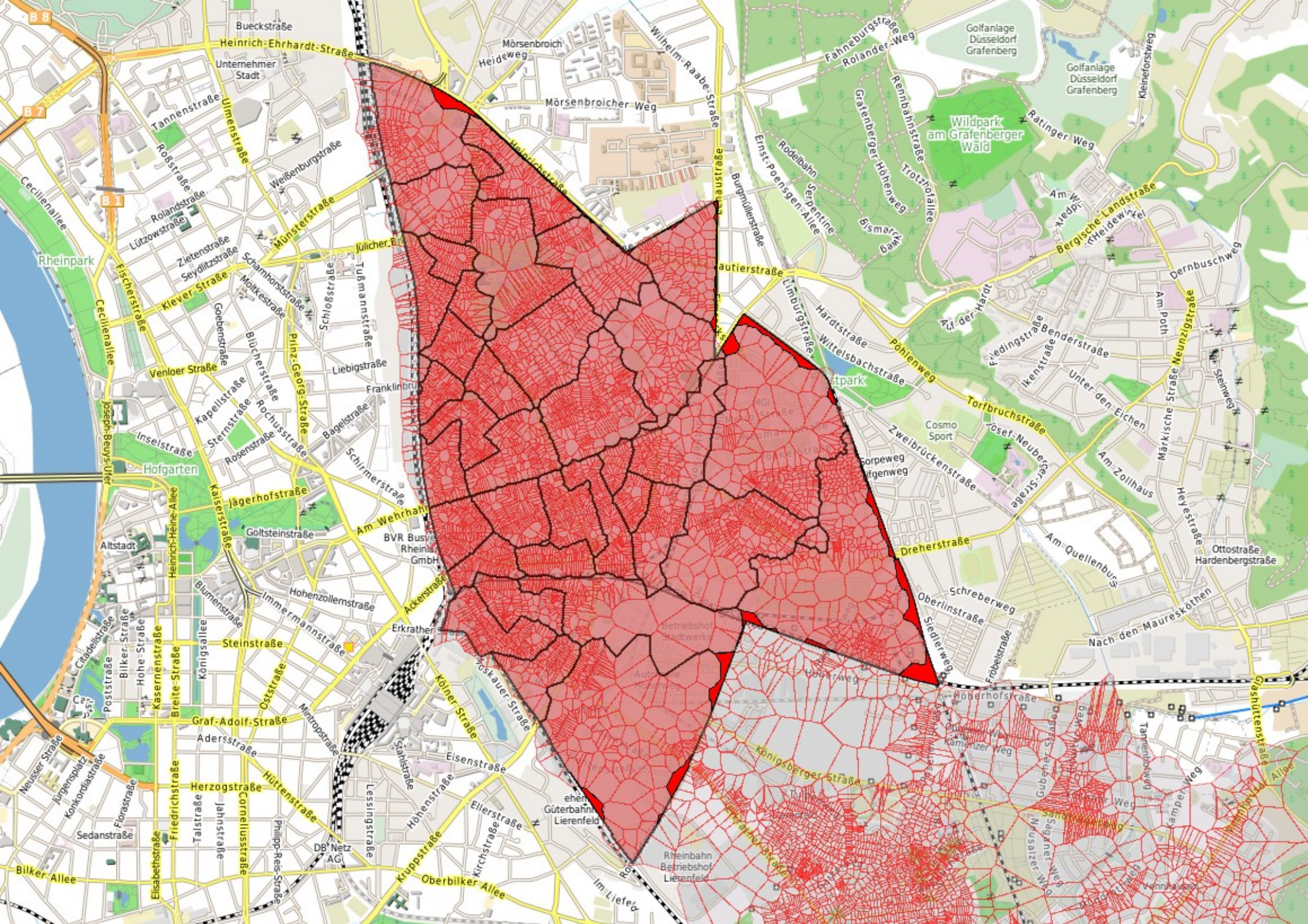


## Fläche für die Löcher erstellen (Referenzfläche erforderlich)

```
INSERT INTO enklave (stadtbezirk, the_geom)
  (SELECT b.stadtbezirk,
        ST_Difference(ST_Union(b.the_geom), ST_Union(a.the_geom)) AS the_geom
   FROM stimmbezirke a, bezirk_fossgis_2010 b
  GROUP BY b.stadtbezirk
 )
;
```

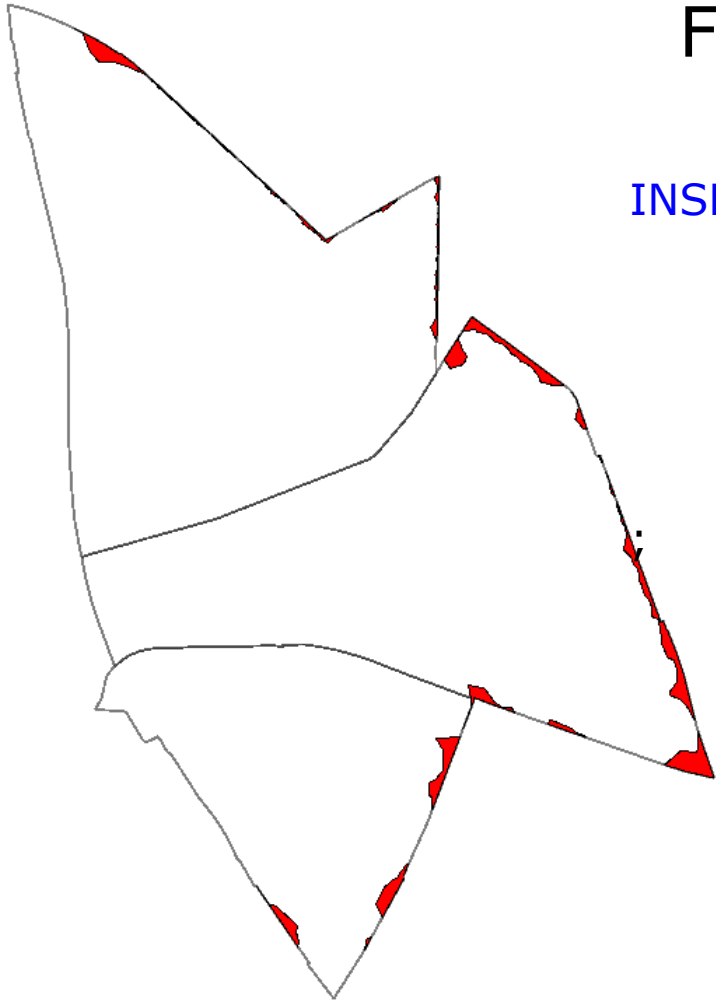




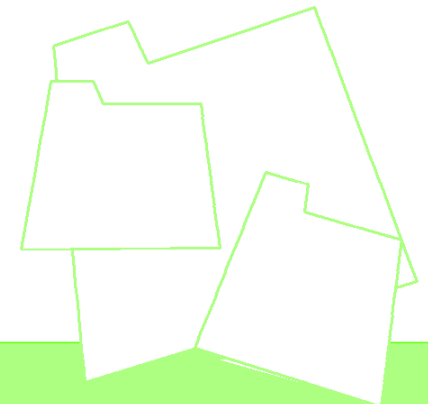




## Fläche aufbrechen

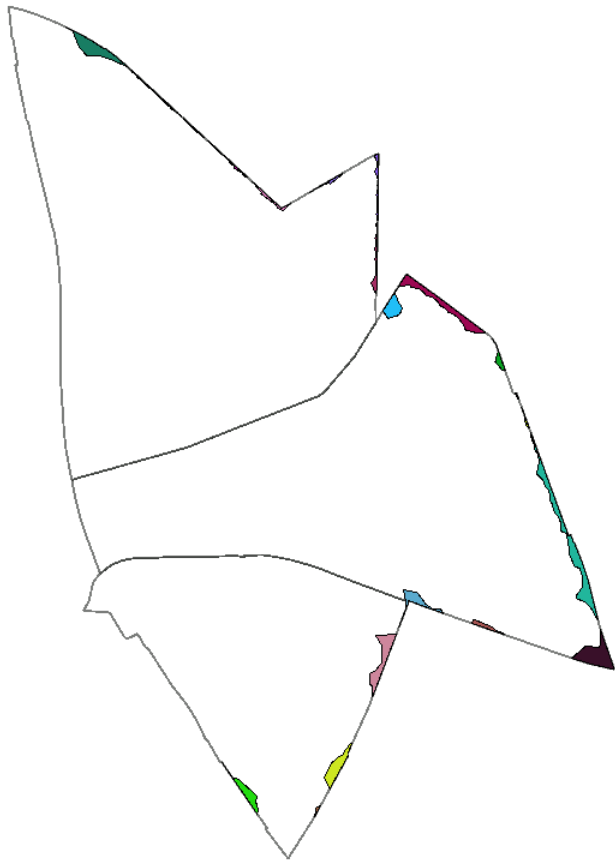


```
INSERT INTO enklave_n (gid, stadtbezirk, the_geom)
  (SELECT (ST_Dump(the_geom)).path[1] AS gid, stadtbezirk,
    (ST_Dump(the_geom)).geom AS the_geom
  FROM enklave
  )
```



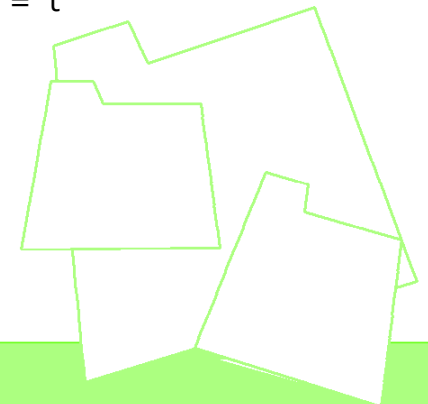


## Flächen zuordnen und vereinigen

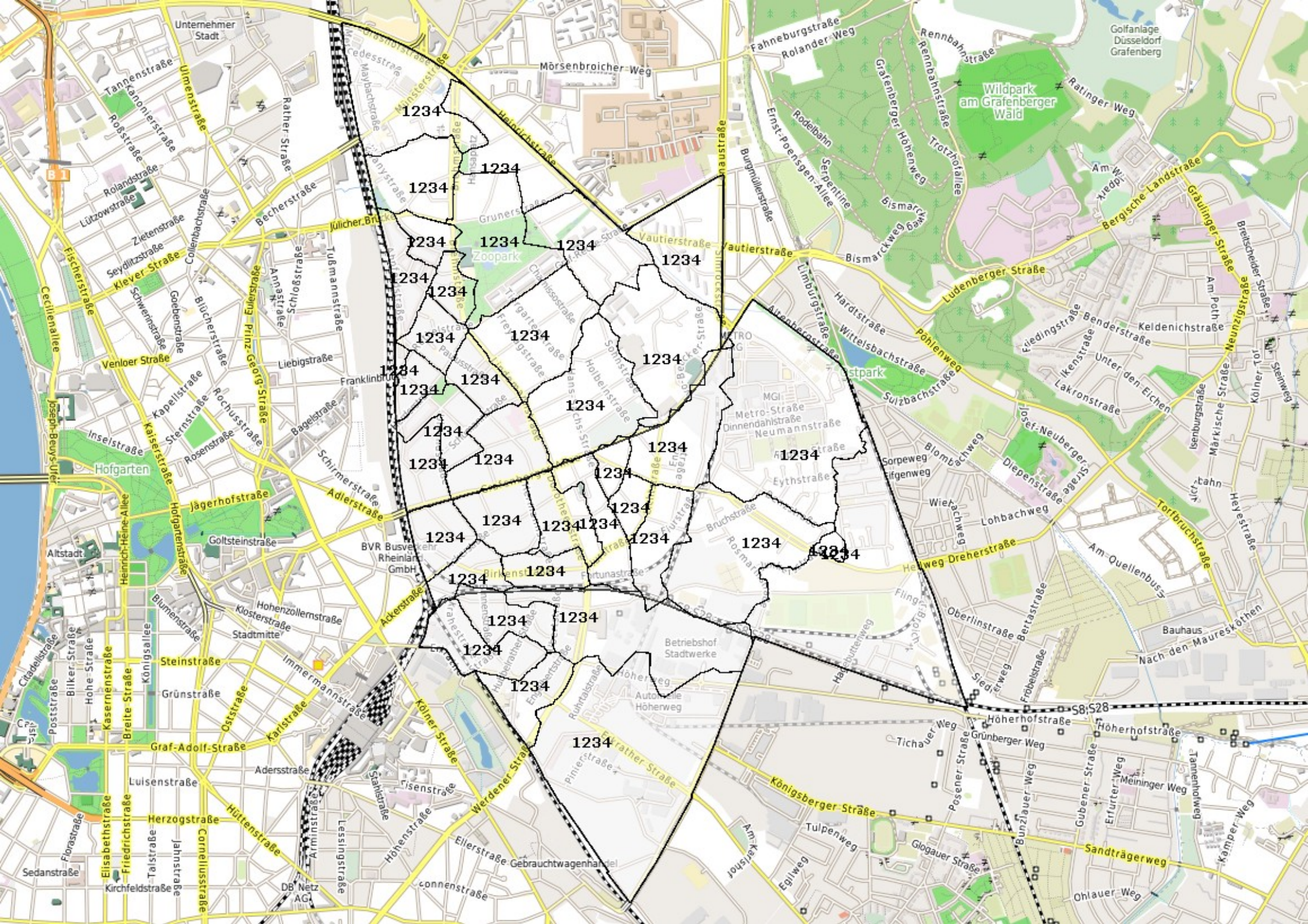


```

UPDATE enklave_n c SET stimmbezirk = foo3.stimmbezirk FROM
  (SELECT foo.gid, foo.stimmbezirk FROM
    (SELECT a.stimmbezirk, b.gid,
      (LENGTH(ST_Intersection(b.the_geom, a.the_geom)))
    FROM stimmbezirke a, enklave_n b
    GROUP BY b.gid, b.the_geom, a.the_geom, a.stimmbezirk
    HAVING ST_Overlaps(a.the_geom, ST_Buffer(b.the_geom, 0.01)) AND
      ST_IsValid(b.the_geom) = 't'
    ) AS foo
    WHERE foo.gid = foo.gid AND foo.length
      (SELECT max(foo2.length) FROM
        in(SELECT b.gid,
          (LENGTH(ST_Intersection(b.the_geom, a.the_geom)))
        FROM stimmbezirke a, enklave_n b
        GROUP BY b.gid, b.the_geom, a.the_geom
        HAVING ST_IsValid(b.the_geom) = 't'
        ) AS foo2
      GROUP BY foo2.gid)
    ) AS foo3
  WHERE c.gid = foo3.gid;
  
```









Vielen Dank für Ihre Aufmerksamkeit!

