

# PostGIS in real ;-) action

Falko Engel

FOSSGIS 2014, Berlin

# Gliederung

- › **PostGIS**
- › **Projekt:** Wälder mit natürlicher Entwicklung (NWE)
- › **Daten:** Projekt mit heterogener Datenlage
- › **Überblick:** in PostGIS durchgeführte Arbeiten
- › **Schwerstarbeit:** Verschneidung zweier großen Tabellen



# PostGIS

*„PostGIS is a spatial database extender for PostgreSQL object-relational database. It adds support for geographic objects allowing location queries to be run in SQL.“*

postgis.net

## Datentyp

- › geometry

## Tabellen

- › geometry\_columns
- › spatial\_ref\_sys

## Funktionen

- › mehrere 100
- › z.B. ST\_Intersection

# Projekt NWE5

## Natürliche Waldentwicklung (5 %) als Ziel der Nationalen Strategie zur biologischen Vielfalt

- › Hintergrund: CBD 1992 (Rio) / NBS 2007
- › Ziel NBS: 5 % NWE bis 2020
- › Ziel Projekt: Belastbare Bilanz aktueller und geplanter Wälder mit natürlicher Entwicklung
- › Auftraggeber: BMU / BfN
- › Projektlaufzeit: 2011 - 2014
- › Abschlussveranstaltung: Oktober 2013, Landesvertretung des Landes Niedersachsen, Berlin
- › Projektpartner: NW-FVA, Uni Freiburg, ILN Bühl



Bundesministerium  
für Umwelt, Naturschutz  
und Reaktorsicherheit



# Daten

- › Waldbesitzer (über Dachverbände) kontaktiert
- › Angefragte Daten: Geodaten und Sachinformationen zu nutzungsfreien Waldflächen
- › ca. 130 Einzellieferungen
- › unterschiedliche räumliche Bezugssysteme: ca. 10
- › insg. ca. 400.000 ha Fläche
- › Zeilen in überlagerungsfreier Gesamttabelle: 167.037
- › Vertizes in Gesamttabelle: 4.801.847

# Mit PostGIS bearbeitete Aufgaben im Projekt

## In Form von Funktionen (PL/pgSQL)

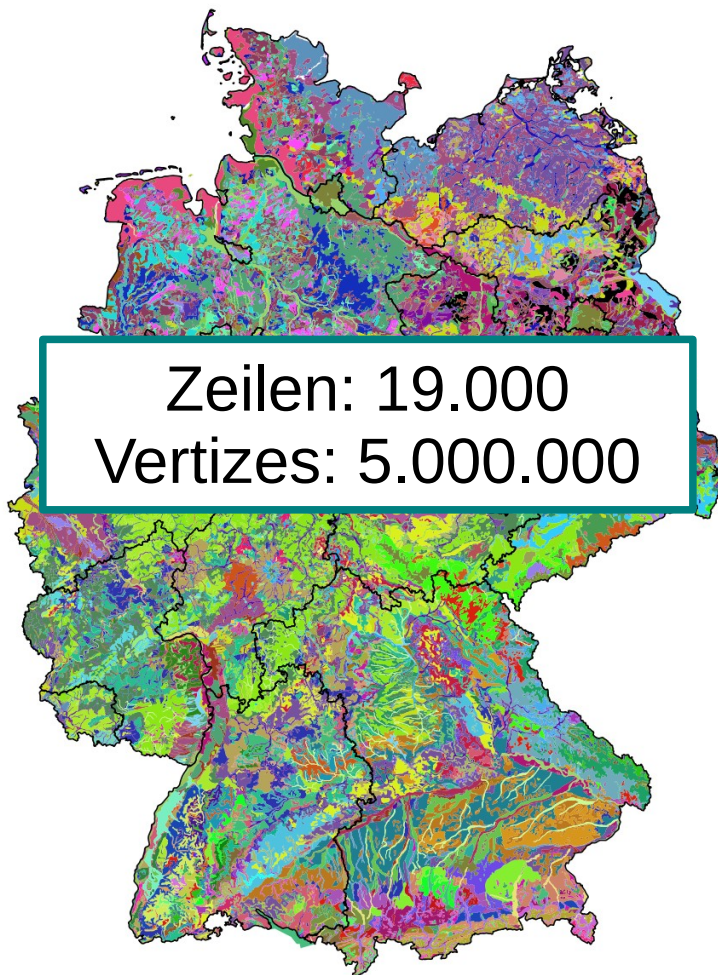
- › Überlagerungsanalyse
- › Duplikate im Geometrien suchen
- › Vereinheitlichung des räumlichen Bezugssystems
- › Verschneidung NWE-Flächen mit überregionalen Daten
- › Aufspüren und reparieren fehlerhafter Geometrien
- › serverbasierte Bereitstellung der Geodaten für mehrere Projektbearbeiter
- › weitere nicht-räumlich Aufgaben in PostgreSQL

# Schwerstarbeit

## Verschneidung großer Daten

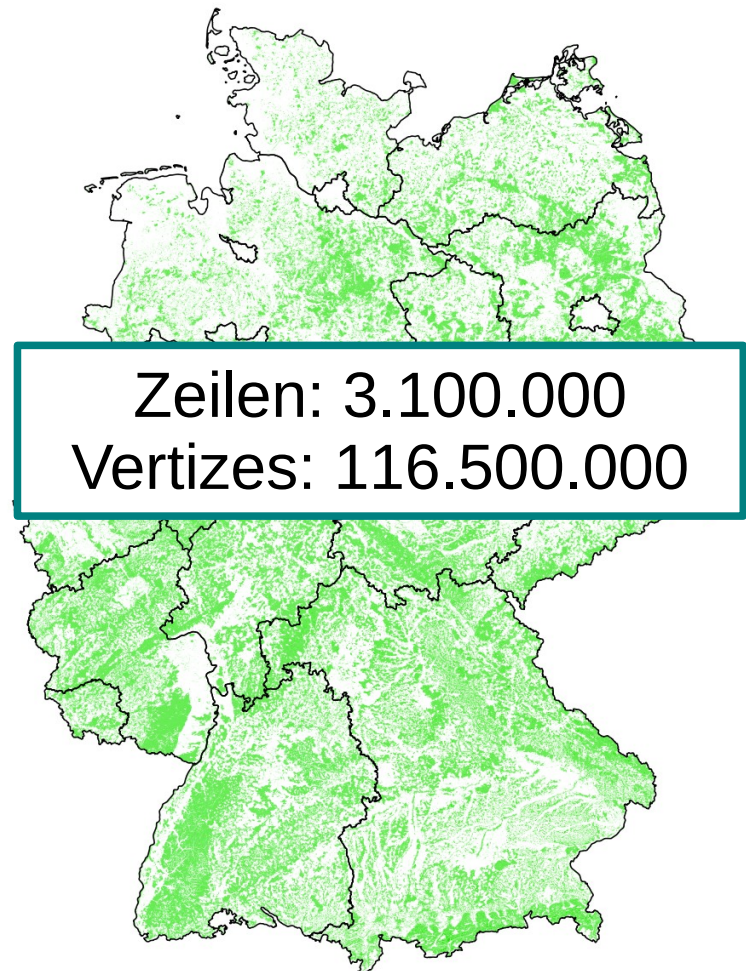
Karte der potentiellen natürlichen Vegetation Deutschlands (pnV)

Quelle: Suck & Bushart 2011 / BfN



Digitales Landbedeckungsmodell (DLM-DE), nur Wald

Quelle: BKG 2012



# Schwerstarbeit

## Ausgangslage

- › **Fragestellung:** Wie verteilt sich die Fläche der pnV-Einheiten auf den aktuellen Wald des DLM-DE? (Grundlage einer Repräsentanzanalyse)
- › **Methode:** Verschneidung beider Datentabellen mittels ST\_Intersection
- › **Problem:** Direkte Verschneidung der Gesamtdaten führte zu
  - › hoher Arbeitsspeicherauslastung (swap)
  - › extrem langen Laufzeiten



# Schwerstarbeit: Lösungsansatz

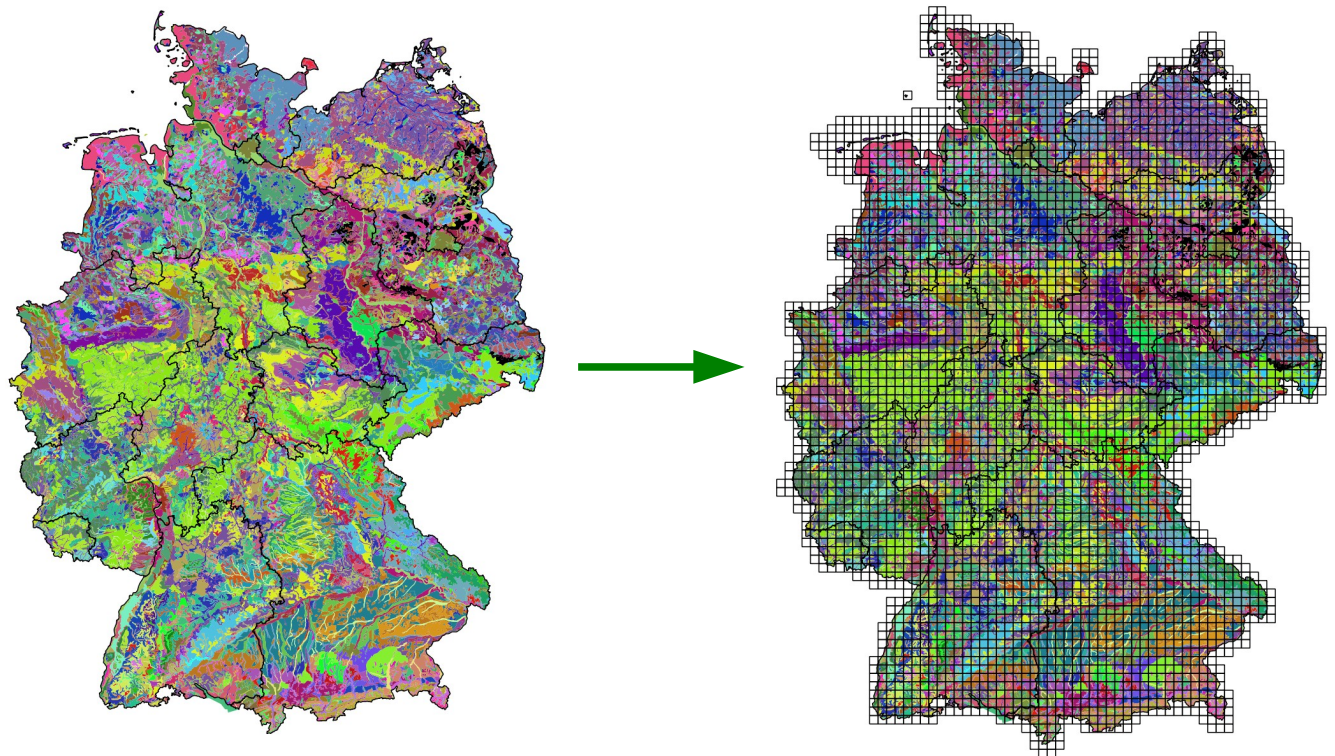
## Lösungsansatz

- › **Lösung:** komplexe Strukturen vereinfachen
- › **Methode:** vorgelagerte Verschneidung mit einem systematisch und einfach aufgebauten Datensatz (hier: Blattschnitt der DTK25)

**pnV-Karte**  
und  
**Blattschnitt der**  
**DTK 25**

Quelle: BKG, 2014

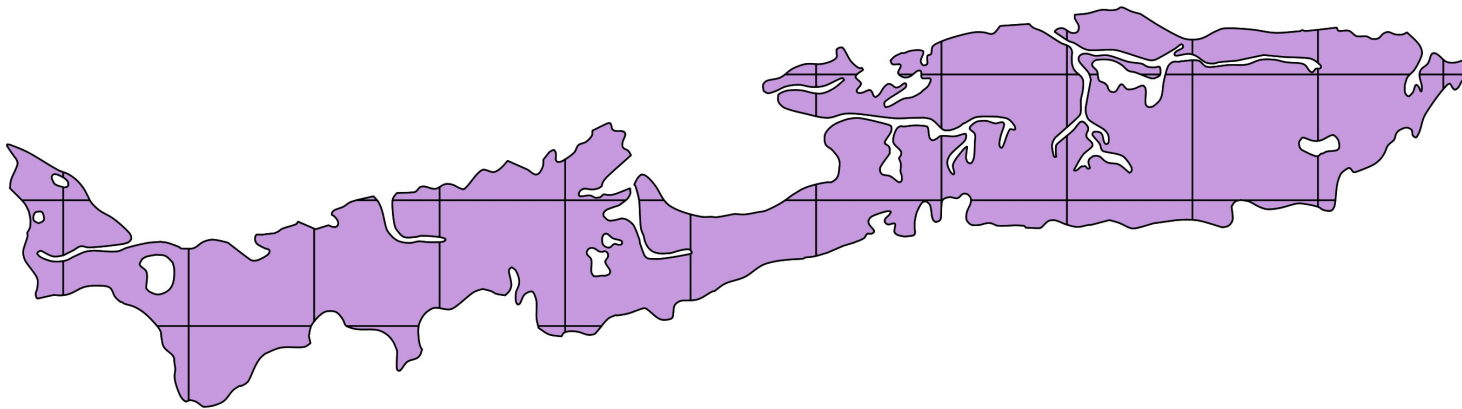
→ verschneiden



# Schwerstarbeit

## Vereinfachung

dadurch erreichen wir:

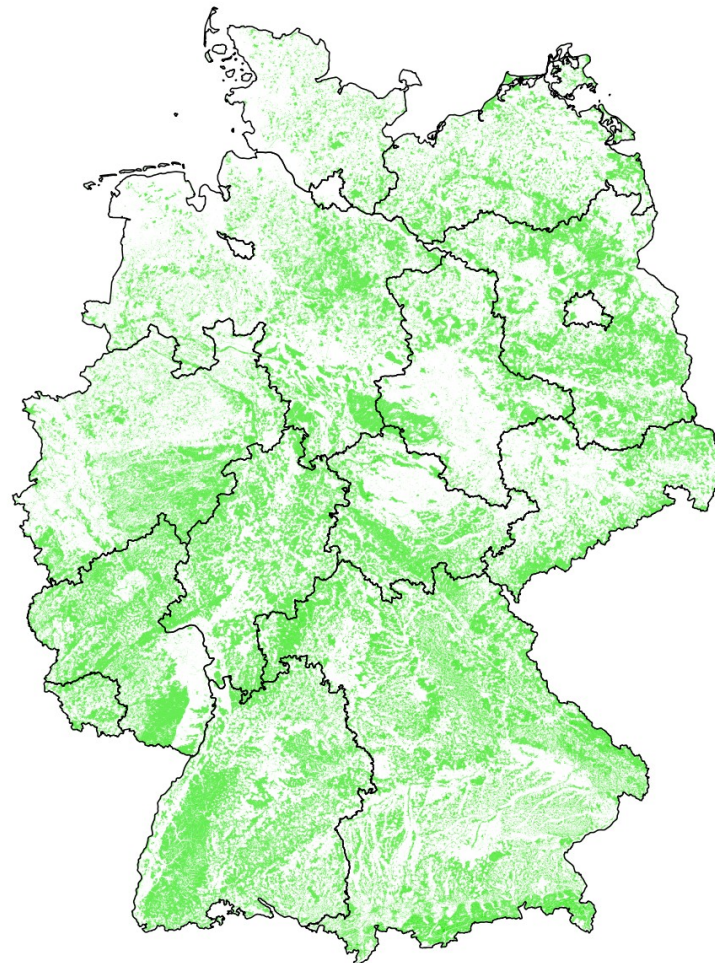
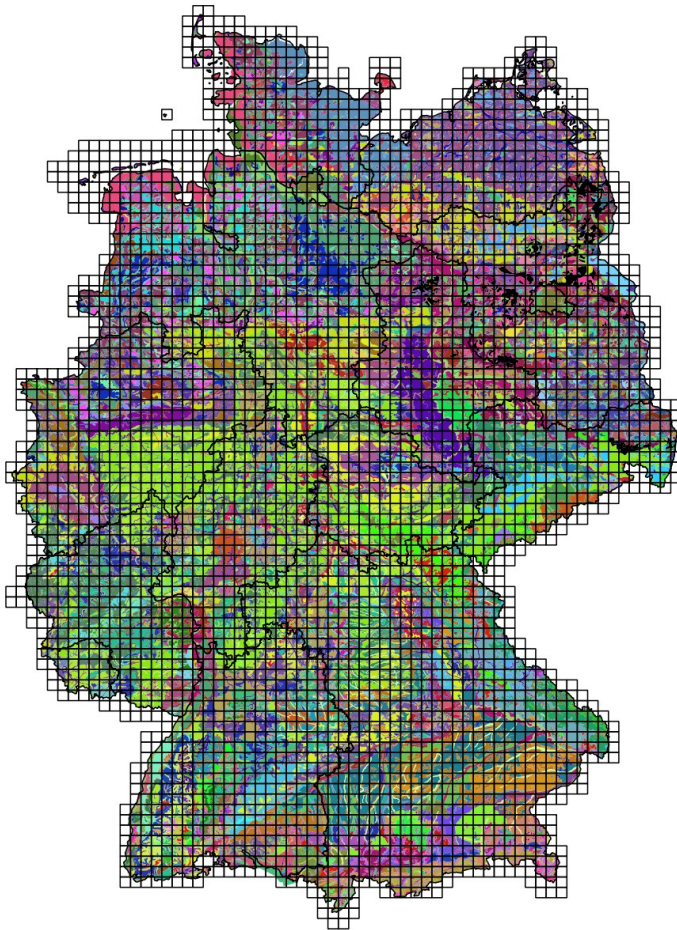


- › 6077 Vertizes in Originalgeometrie
- › 36 Einzelgeometrien mit maximal 589 Vertizes
- › Blattschnittzellen einzeln abzuarbeiten: RAM wird geschont

# Schwerstarbeit

sukzessive Verschneidung  
mit DLM-DE

```
FOR i IN dtk_min .. dtk_max  
  LOOP  
    INSERT INTO new_tab  
      ST_Intersection  
  END LOOP;
```





# Schwerstarbeit

## SQL: Leere Tabelle mit Geometriespalte erzeugen

```
CREATE TABLE tmp1 (col_a integer, col_b varchar);
```

```
SELECT AddGeometryColumn('tmp1', 'the_geom', 25832,  
                          'POLYGON',2);
```

# Schwerstarbeit

## SQL: Blattschnitt mit pnV-Tabelle verschneiden

```
CREATE TABLE tmp2 AS
(
    SELECT
    a.pnv_code AS mycol1,
    b.gid AS tmpid,
    ST_Intersection(a.the_geom, b.the_geom AS the_geom

    FROM pnv_tab AS a

        INNER JOIN dtk25_tab AS b
        ON ST_Intersects(a.the_geom, b.the_geom)
);
```

# Schwerstarbeit

## SQL: Rasterelemente abarbeiten

```
FOR i IN array_lower(myarr, 1)
    .. array_upper(myarr, 1)
LOOP
    myelem := myarr[i];

    EXECUTE 'CODE';

END LOOP;
```

INSERT INTO tmp1

SELECT  
a.mycol1,  
b.mycol2,  
ST\_Intersection(a.the\_geom, b.the\_geom) AS the\_geom



RAM wird offensichtlich geschont

FROM tmp2 AS a

INNER JOIN dlm\_tab AS b  
ON ST\_Intersects(a.the\_geom, b.the\_geom)

WHERE a.tmpid = ' || myelem || '

# Schwerstarbeit

## In PostGIS integrierte bzw. weitere möglich Optimierungen bei Verschneidungen

- › räumliche Indizes (GIST): Search tree, der schnell durchsucht werden kann um räumliche Beziehungen zu prüfen
- › ST\_Intersection nur einsetzen wenn nötig
- › ST\_Simplify

# Fazit

- › Projekt mit komplexer Datenlage in PostgreSQL / PostGIS bearbeitet
- › PostGIS bietet Grundfunktionalität für alle nötigen Arbeiten mit Geodaten
- › Funktionalität ist durch schreiben eigener Funktionen beliebig ausschöpfbar
- › z.B.: Problem RAM-Auslastung konnte „kreativ“ durch Portionierung gelöst werden



**Vielen Dank!**

# Literatur

Bundesamt für Kartographie und Geodäsie (BKG) Hrsg. (2012): Digitales Landbedeckungsmodell DLM-DE.

Suck, R., Bushart, M. & Bundesamt für Naturschutz (2011): Karte der potentiellen natürlichen Vegetation Deutschlands.