PostGIS Workshop
Einführung in Geodatenbanken mit PostGIS

05.04.2011, FOSSGIS 2011, Heidelberg

<stephan.holl@intevation.de> | <harald.schwenk@agentur-geoinfo.de>

Beispieldaten

- Die im Workshop verwendeten Beispieldaten sind unter der folgenden URL zu beziehen:
- http://ftp.intevation.de/users/stephan/fossgis-2011/ws/postgis/postgis-fossgis2011.tar.gz

Über die Referenten

- Stephan Holl,
  - Intevation GmbH
  - stephan.holl@intevation.de
  - http://www.intevation.de/geospatial
- Harald Schwenk,
  - agentur geoinfo
  - harald.schwenk@agentur-geoinfo.de
  - http://www.agentur-geoinfo.de

PostgreSQL

- Features
  - AKID (**A**tomar, **K**onsistent, **I**soliert, **D**auerhaft)
  - SQL 92, Query Optimizer
  - Volltext-Suche
  - Seperation, Replikation
  - Hot-Backup, Write-ahead Logs / PITR
  - Stored Procedures
  - Trigger / Rules
- Freie Software: BSD

PostGIS: Geschichte

- 2001:
  - SFSQL als Designgrundlage
  - Anbindung UMN MapServer
- 2002:
  - Verbesserte Basisfunktionen, Index
- 2003 - 2005:
  - GEOS-Anbindung
  - Lightweight Geometries
  - PostGIS 1.0.0
- 2006: OpenGIS SFSQL compliance
- 2007: CurveTypes

PostGIS: Geschichte

- 2009
  - Performance-Optimierungen
  - Stabilitätsoptimierungen
- 2010
  - PostGIS 1.5.2, Septermber 24
- 2011
  - PostGIS 2.0: Raster-Support
  - PostGIS 2.0: 3D
  - Idealer Release: Juni 2011

PostGIS Installation GNU/Linux

- Distributionen liefern Pakete für PostgreSQL
- Ab PostGIS 1.1.0 vereinfachte Installation:
  - Vorbedingungen:
    - PostgreSQL Laufzeitsystem
    - PostgreSQL Entwicklungspaket
    - PostGIS Quellen
  - Installieren:
    - ./configure [weitere Optionen]
    - make
    - make install
- Ab PostGIS 1.5.0 Geometry-Types

Einrichtung einer Datenbank

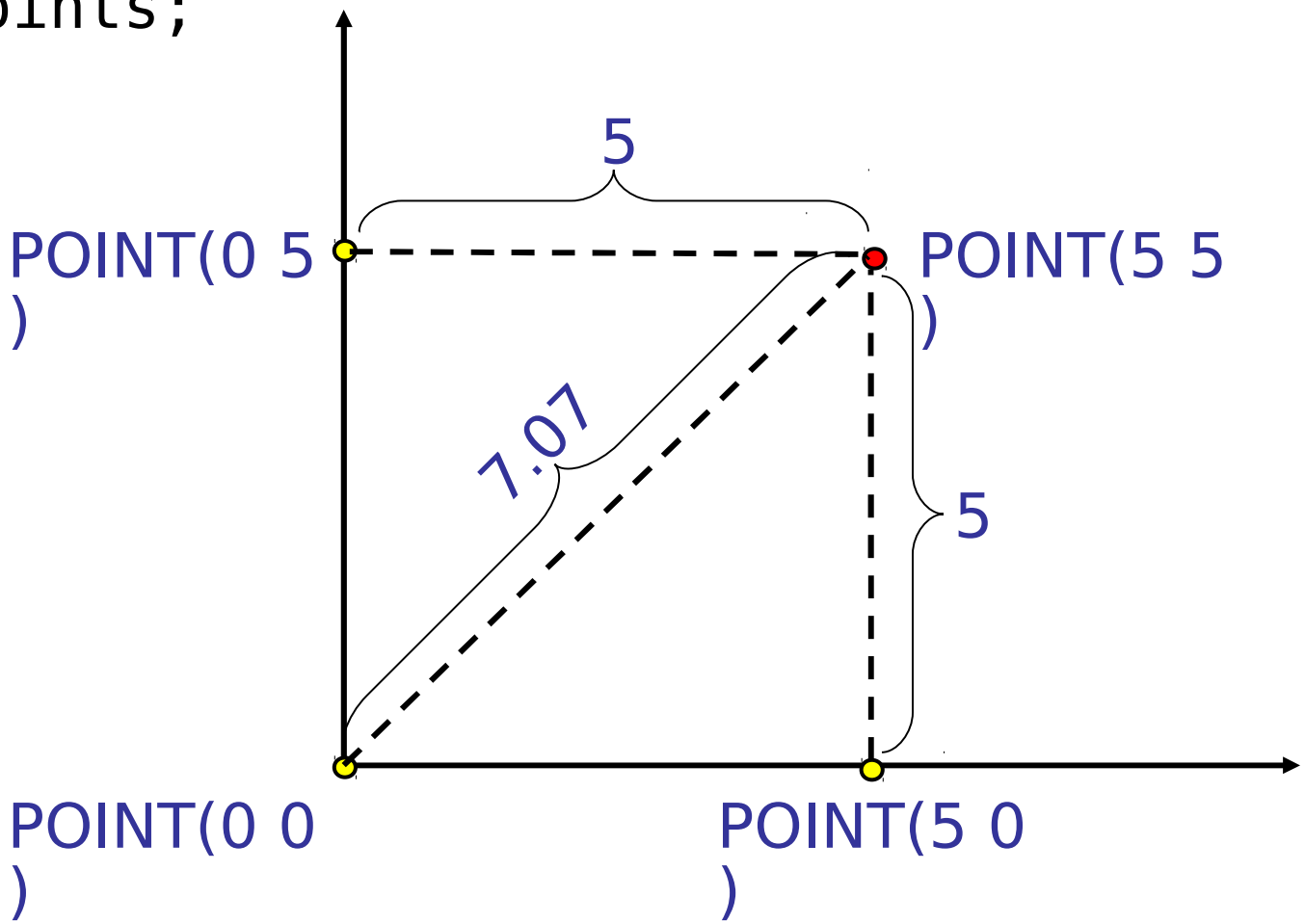
- createdb <datenbankname>
- createlang plpgsql <datenbankname>
- psql -f lwpostgis.sql <datenbankname>
- psql -f spatial_ref_sys.sql <datenbankname>

Einfache Abfragen

```
create table points (pt geometry, name varchar);
insert into points values
       ('POINT(0 0)', 'Ursprung');
insert into points values
       ('POINT(5 0)', 'X-Achse');
insert into points values
       ('POINT(0 5)', 'Y-Achse');
select name, st_AsText(pt),
       st_Distance(pt, 'POINT(5 5)') from points;
```

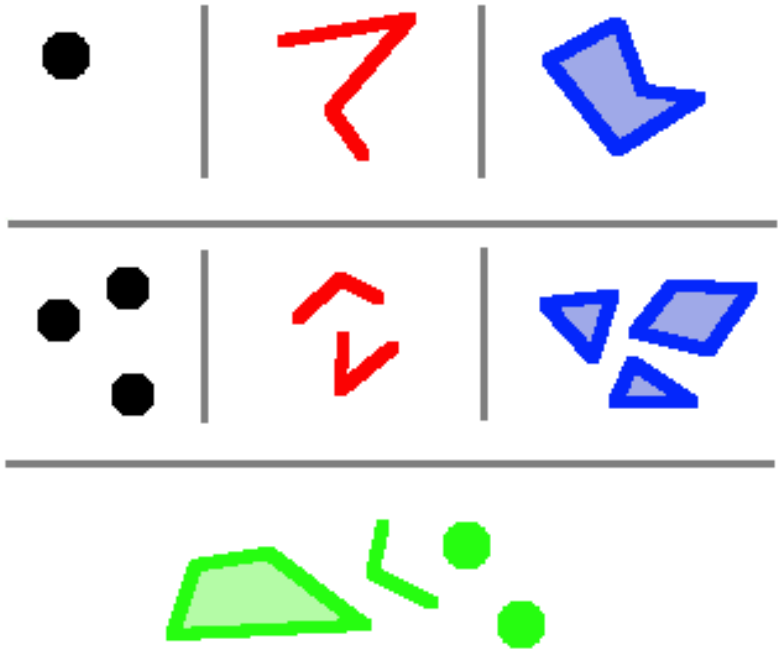
Einfache Abfragen

```
SELECT name, st_AsText(pt),
       st_Distance(pt, 'POINT(5 5)') from
  points;
```



PostGIS Geometrietypen

1. Punkte
2. Linien
3. Polygone
4. MultiPunkte
5. MultiLinien
6. MultiPolygone
7. Kollektionen
8. Kurventypen
9. 3D-Typen



PostGIS Geometrietypen I

- POINT(5 5)
- LINESTRING(0 5, 5 0);
- POLYGON((0 0, 2 0, 0 2, 0 0))
- MULTIPOINT((5 3), (2 5));
- MULTILINESTRING ...
- MULTIPOLYGON ...
- GEOMETRYCOLLECTION(
  POINT(...),
  LINESTRING(...), ...
)

PostGIS Geometrietypen II

- CIRCULARSTRING(0 0,1 1,1 0)
- COMPOUNDCURVE(CIRCULARSTRING(0 0,1 1,1 0),1 0,0 1))
- CURVEPOLYGON(CIRCULARSTRING(0 0,4 0,4 4,0 4,0 0),(1 1,3 3,3 1,1 1))
- MULTICURVE((0 0,5 5),CIRCULARSTRING(4 0,4 4,8 4))
- MULTISURFACE(CURVEPOLYGON(CIRCULARSTRING()))

OGC Standards

- Tabelle spatial_ref_sys:

```
  Spalte   |           Typ            | Attribute
-----------+--------------------------+-----------
 srid      | integer                  | not null
 auth_name | character varying(256)   |
 auth_srid | integer                  |
 srtext    | character varying(2048)  |
 proj4text | character varying(2048)  |
```

- SRID=4326;POINT(52.8 8.4)

OGC Standards

- Tabelle geometry_columns:

```
      Spalte       |          Typ             | Attribute
-------------------+--------------------------+-----------
 f_table_catalog   | character varying(256) | not null
 f_table_schema    | character varying(256) | not null
 f_table_name      | character varying(256) | not null
 f_geometry_column | character varying(256) | not null
 coord_dimension   | integer                | not null
 srid              | integer                | not null
 type              | character varying(30)  | not null
```

- Funktion AddGeometryColumns:

```
AddGeometryColumn(<schema_name>, <tabellen_name>,
                  <spalten_name>, <srid>, <type>
                  <dimension>)
```

- Ohne <schema_name> aktuelles Schema

```
SELECT AddGeometryColumn('roads', 'geom', 423, 'LINESTRING', 2);
```

OGC Standards

- Validierung der Geometrien
  - Simple Feature beschränkt Varianten
  - PostGIS Funktion:
    - isvalid(<geom>)
    - liefert als NOTICE Hinweise bzgl. Invalidität
  - keine automatische Prüfung beim Einfügen
  - explizit anlegen:
    - ```
      ALTER TABLE roads
        ADD CONSTRAINT geometrie_valide_check
        CHECK (isvalid(geom));
      ```
      ```

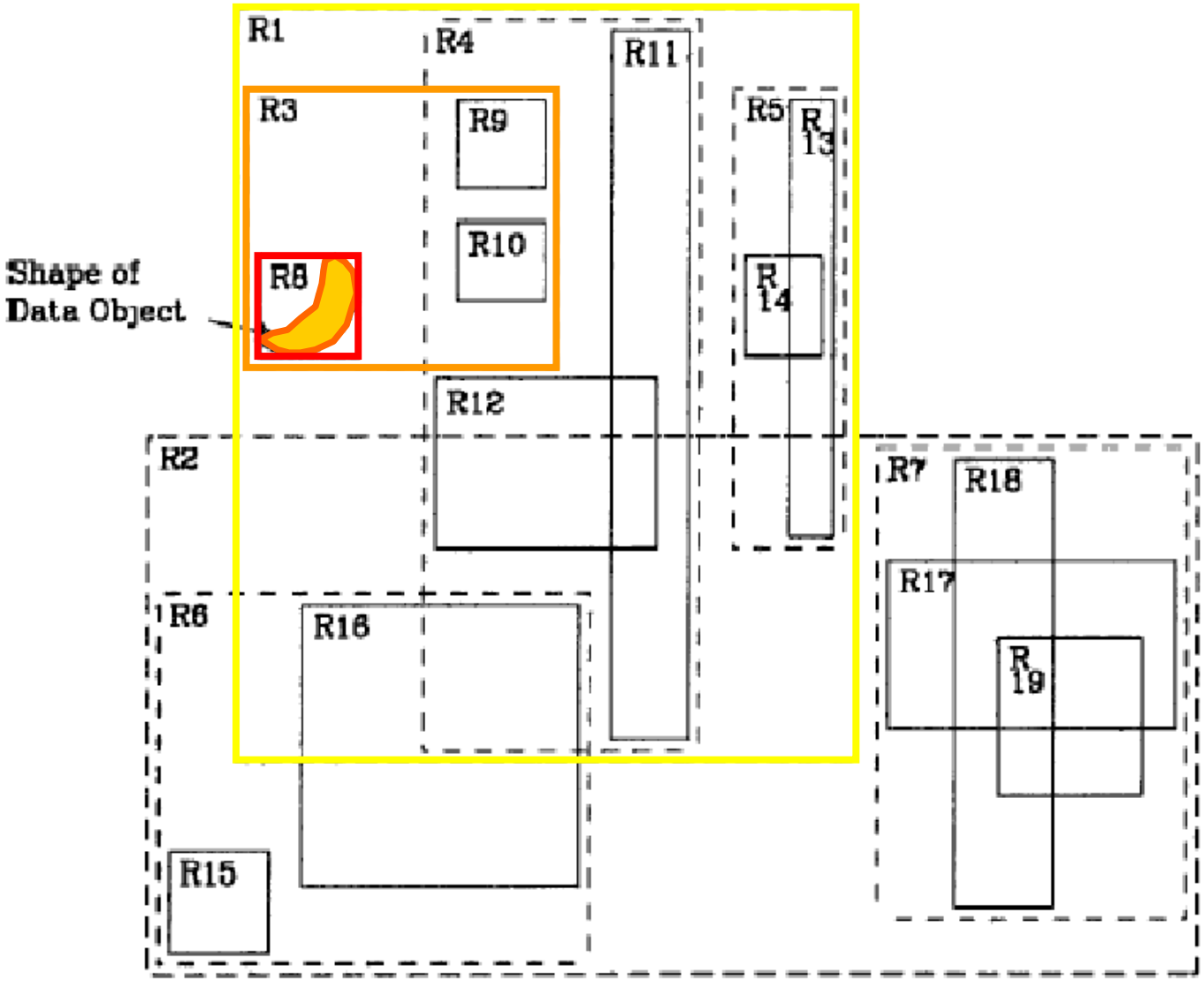
Import von Geo-Daten

- Shp2pgsql <shapefile> <tabelle>
  - Optionen:
    - -s: SRID
    - -D: Postgresql COPY (Bulk load)
    - -I: GiST-Index
  - Ausgabe SQL-Skript
  - Möglichkeit einer Pipe: „| psql .....“

Export von Geo-Daten

- Pgsql2shp <opts> <db> <tabelle>
  - Optionen:
    - -f <Ausgabefile>
    - -h, -p ...
    -
- Beschränkungen im Zielformat beachten!

Räumliche Indizes



Räumliche Indizes

- Erstellen eines Index:

```
CREATE INDEX bc_roads_gidx
  ON bc_roads
  USING GIST ( the_geom );
```

- Sammeln von Statistiken

```
VACUUM ANALYSE;
```

- Seit PostGIS 1.3 wird der Index automatisch beim BBOX-Vergleich genutzt, explizites Anfragen ist nicht mehr nötig.

Spatial Analysis

- Gesamtlänge aller Straßen in BC in Kilometern?

```
SELECT sum(st_length(the_geom))/1000
    FROM bc_roads;
```

Spatial Analysis

- Welches ist die größte Stadt, nach Fläche?

```
SELECT name
  FROM bc_municipality
  WHERE st_area(the_geom) =
    (SELECT max(st_area(the_geom))
       FROM bc_municipality);
```

- Alternative:

```
SELECT name, st_area(the_geom) AS
area FROM bc_municipality
  ORDER by area DESC LIMIT 1;
```

Spatial Analyis - Entfernungen

- Wieviele Wähler der Grünen Partei leben in einem höchstens 2 Kilometer vom Pub 'TABOR ARMS' entfernten Wahlbezirk?

```
SELECT sum(v.gp) AS Gruene_Waehler
  FROM bc_voting_areas v, bc_pubs p
  WHERE st_distance(v.the_geom,
                    p.the_geom) < 2000
    AND p.name like 'Tabor Arms%';
```

Spatial Analyis - Entfernungen

- Optimierung: Einschränkung der zu prüfenden Wahlbezirke.

```
SELECT sum(v.gp) AS Gruene_Waehler
  FROM bc_voting_areas v, bc_pubs p
  WHERE st_distance(v.the_geom,
                    p.the_geom) < 2000
    AND v.the_geom &&
        st_expand(p.the_geom, 2000)
    AND p.name like 'Tabor Arms%';
```

Spatial Joins

- Verknüpfung zweier Tabellen anhand Beziehung zwischen Geometrien
- Alle Pubs, die näher als 250 m an einem Krankenhaus liegen:

```
SELECT h.name, p.name
    FROM bc_hospitals h, bc_pubs p
    WHERE st_distance(h.the_geom,
            p.the_geom) < 250;
```

Spatial Joins

- Zusammenführung von Datenbeständen:
  - Alle Wahlkreise in 'PRINCE GEORGE':

```
SELECT v.id
  FROM bc_voting_areas v,
       bc_municipality m
 WHERE st_intersects(v.the_geom,
                     m.the_geom)
   AND m.name = 'PRINCE GEORGE';
```

Räumliche Prädikate

- Verschiedene Prädikate, um Beziehung zwischen Geometrien zu untersuchen:
  - st_equals(geometry, geometry)*
    - Linie(0 0, 10 10), Linie(0 0, 5 5, 10 10)
  - st_disjoint(geometry, geometry)*
  - st_intersects(geometry, geometry)*
  - st_touches(geometry, geometry)*
    - Polygon((0 0, 1 0, 1 1, 0 0)) und
    - Polygon((1 1, 1 0, 2 0, 1 1))
  - st_Crosses()*
  - st_Within()*

Räumliche Prädikate

- st_Overlaps(geometry, geometry)*
- st_Contains(geometry, geometry)*
- st_Covers(geometry, geometry)*
- st_CoveredBy(geometry, geometry)*
- st_Relate(geometry, geometry, intersectionPatternMatrix)*
- st_Relate(geometry, geometry)*
  - Dimensionally Extended 9 Intersection Model (DE-9IM)

Verschneidungen

- Methoden zur Analyse / Erzeugung neuer Geometrien
- Prozentuale Anteile der Gemeinde Hudson's Hope an Wahlkreisen:

```
SELECT v.id, v.region,
 st_Area(st_Intersection(v.the_geom,
          m.the_geom))/
st_Area(v.the_geom)*100 as anteil
FROM bc_voting_areas v,
     bc_municipality m
WHERE v.the_geom && m.the_geom
      AND m.name = 'HUDSON''S HOPE';
```

Verschneidungen

- st_Intersection(geometry, geometry)*
- st_Difference(geometry A, geometry B)*
- st_SymDifference(geometry, geometry)*
- st_Union(geometry, geometry)*
- Auch als Aggregat:
  - st_Union(geometry set)
  - st_MemUnion(geometry set)

Projektionen

- Konsistenz der Daten
  - SELECT st_srid(the_geom)
    FROM bc_roads LIMIT 1;
- Umprojektion (Transformation):
  - SELECT st_astext(the_geom)
    FROM bc_roads LIMIT 1;
  - SELECT st_astext(
    st_transform(the_geom,4326))
    FROM bc_roads LIMIT 1;

Schulungstermine

- PostGIS-Schulungen 2011
  - Einführung (2 Tage)
    - 31.05 - 01.06.2011
    - 27. - 28.09.2011
    - 01. - 02.11.2011
  - PostGIS für Fortgeschrittene (2 Tage)
    - 02. - 03.06.2011
    - 29. - 30.09.2011
    - 03. - 04.11.2011
- Weitere Termine bieten wir auch gerne bei Ihnen Inhouse an! Fragen Sie nach!

http://www.intevation.de/geospatial

- **www.postgis.org**
- **www.postgresql.org**

- **Stephan Holl <stephan.holl@intevation.de>**
- **www.intevation.de/geospatial**

- **Harald Schwenk <harald.schwenk@agentur-geoinfo.de>**
- **www.agentur-geoinfo.de**