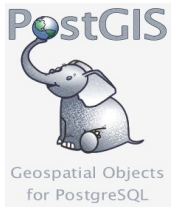
6. Dezember 2010

PostGIS - das Wo? in der Datenbank



Stefan Keller und Andreas Neumann

Einleitung

- **Die Ergänzung von (bestehenden) Informationen mit Ortsbezug erschliesst völlig neue Möglichkeiten**
  - **z.B. Suche nach Bars in der Nähe**
  - **Geodaten sind die Basis dafür!**

- **Diese Präsentation soll Informatiker in die Welt der Geodaten einführen**

Geodaten

- **Def. Geodaten:**
  - **einfach Daten mit Raumbezug – und trotzdem unterschätzt!**

- **Raumbezug:**
  - **Punkt, Linie, Fläche mit (gemeinsamem) Koord.-Referenzsystem, z.B. Lat./Lon.=48.72,9.16**
  - **Adresse: Plieninger Str. 100, D-70567 Stuttgart**

- **Def. Geoinformationssysteme (GIS):**
  - **Geo-DB und weitere SW-Komponenten (EVAP)**

What's so special about spacial?

- **5 Gründe warum Geodaten "komplex" sind:**
  1. **Komplexe Datenstrukturen**
     - **da nicht-relational, siehe Datentypen gleich**
  1. **Umfangreiche Metadaten**
     - **u.a. Koordinaten-Referenzsysteme, siehe Theorie**
  1. **Komplexe Konsistenzbedingungen und Grafik**
     - **siehe gleich**
  1. **Grosse Datenmengen (d.h. komplex in der Verarbeitung)**
     - **OpenStreetMap zurzeit 171 GB (XML ausgepackt)**
  1. **Teure Datenerfassung und -nachführung**

Geschichte von IT, DBMS und PostGIS

- Erste kommerzielle Computer Ende 50er Jahre (LEO I, UNIVAC)
  - erste GIS ab 1960 (Symap Uni. Harvard)
- Erste DBMS (Oracle) 1978
  - erstes (Server-)GIS 1982 (ARC/INFO)
  - Postgres 1986 (Stonebreaker)
- Erste Desktop CAD (AutoCAD) 1982
  - 1986 Desktop-GIS (Mapinfo)
- Erste kommerz. DBMS mit Geodaten-Typen 1986 (Oracle 7.3)
  - 1996: PostgreSQL Open Source und 2001: PostGIS
- Erste Webapplikationen (u.a. E-Mail) frühe 90er;
  - 'Geo-Web' ab Mitte 2000 (Google Maps); HTML5 ab 2010

Fazit: Zehn Jahre Rückstand von IT-Mainstream zu GIS

(Credits: Muki Haklay, University College London)

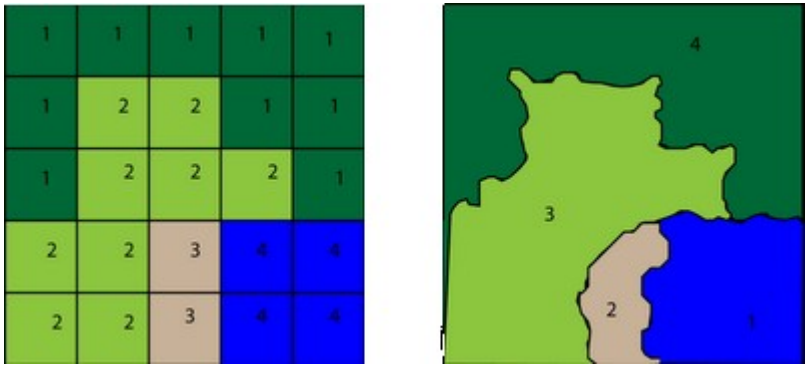
Moral von der Geschicht‘…

- Sie dokumentiert, dass Geodaten “komplex” sind
- Der „Rückstand“ erklärt, warum GIS seit 20 Jahren als "Emerging Technology" bezeichnet werden
- Alles das hat (hatte) zur Folge, dass
  - … viele Firmen im GIS-Business aufgeben mussten (z.B. Location-Based Services)
  - … GIS-Firmen ca. 10 Jahre durchhalten müssen

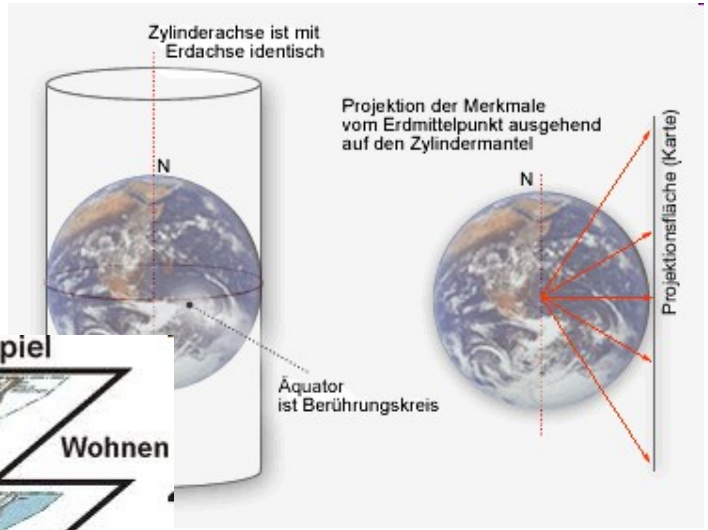
> Womit alle vor Selbstüberschätzung gewarnt seien!

Etwas Theorie…

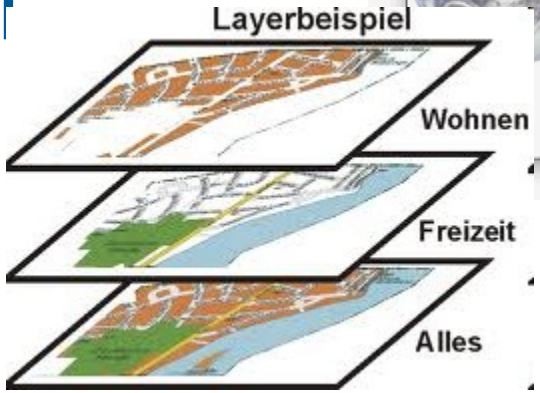
- Vektor
  und Raster
  sowie 'mixed‘



- Koordinaten-Referenz-Systeme (SRS, CRS), z.B. EPSG:4326 für ‚GPS‘



- Koordinaten mit gleic
  SRS => Layer-Prinzip
  - Wirken wie „Fremdschlüssel“



Geodatentypen: Grundlagen

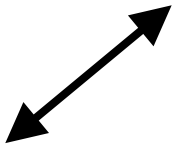
- **Geometriedatentypen in PostGIS:**
  - **Point, LineString/Curve, Polygon, etc.**
  - **MultiPoint, MultiLineString, MultiPolygon, etc.**
  - **Dazu 3D-Datentyp Geography**
  - **Standards OpenGIS Consortium, ISO 19000, SQL/MM**

- **2D-Daten:**
  - **POINT(8.55 47.37) -- Zürich**
  - **LINESTRING(0 0, 1 1, 1 2)**
  - **POLYGON((0 0,4 0,4 4,0 4,0 0), (1 1, 2 1, 2 2, 1 2,1 1))**

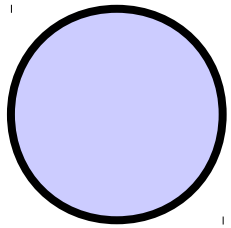
- **3D-Daten:**
  - **POINT(8.55 47.37 410) -- Zürich auf 410m Höhe**

Geodatentypen: Funktionen

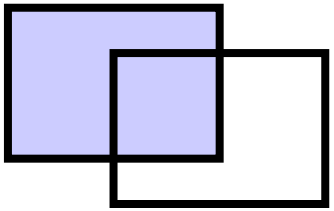
- ST_Distance(geom, geom) -> units



- ST_DWithin(geom, geom, distance) -> boolean



- ST_Intersect(geom, geom) -> geom



- ST_Transform(geom, srid) -> geom – z.B. SRID=4326

  z.B. `POINT(683946.7 246796.8)` zu `POINT(8.55 47.36)`

- …
  - Alle mit Präfix „ST_“ (Spatial Type)
  - … und noch ca. 300 Funktionen mehr…

Ein-/Ausgabe-Formate

- **Vektor-Daten**
  - **KML, Shapefile, GeoJSON, etc.**

  > ogr2ogr

  MapInfo, DGN, CSV, GPX, SQLite, PostGIS…

- **Raster-Daten**
  - **gif, png, jpg, tif, etc.**

  > gdalinfo --formats

  VRT, GeoTIFF, HDF, HGT, RLE, XPM, SGI, ERS, …

Konsistenzbedingungen, Ebene/Layer und Grafik

- „Statische Konsistenzbedingungen“
  - Was ist eine gültige Linie? 2D-Fläche? 3D-Objekt?

- Ebene/Layer = Gruppe von Entitäten mit Beziehungen
  - Die Geometrie (und zugeordnetes SRID) ist die Beziehung!
  - IT = ~ „UML-Package“; CAD = farbige Vektoren mit IDs

- Grafik
  - Generierung von Grafik aus Koordinaten (-listen)… (ist etwa so, wie aus XML-Rohtext sowohl Abstract als auch Buch generieren)
  - Kunst der Visualisierung: Chronische Platznot auf Bildschirm und Papier

Gut zu wissen…

- **Abhilfe zum Platzmangel:**
  - **Multirepräsentation!**
  - **Kartogr. Zoom (Zoom versus Skalierung)**

- **Drei Stolpersteine (aus Sicht Informatiker)**
  - **Viele Begriffe, Formate, Implementationen**
  - **Metadaten (SRS), Konsistenzbedingungen**
  - **Was erscheint wo? (Zoom-Level)**

Beispiel mit SQL (1)

```
-- Gegeben Staedte-Tabelle der Schweiz:
   CREATE TABLE staedte(
       gid integer PRIMARY KEY,
       name character varying,
       geom geometry);

-- Räumliche Geometrie-Spalte und -Index dazufügen:
   SELECT AddGeometryColumn(
       'public', 'staedte', 'geom', 21781, 'POINT',2);
   CREATE INDEX staedte_gist ON staedte USING gist (geom);

-- Ausgabe 1: Lokales SRS:
   SELECT name, ST_AsText(geom), ST_SRID(geom)
   FROM   staedte AS sta
   WHERE  name='Zürich'
   "Zürich";"POINT(683946.7 246796.8)";21781
```

Beispiel mit SQL (2)

```
-- Ausgabe 2: SRS mit lon/lat
SELECT name, ST_AsText(ST_Transform(geom,4326))
FROM   staedte AS sta
WHERE  name='Zürich' LIMIT 1;
"Zürich";"POINT(8.55 47.36)"

-- Ausgabe 3
SELECT 'http://maps.google.com/?q='
       || ST_Y(ST_Transform(geom,4326))::text
       || ','
       || ST_X(ST_Transform(geom,4326))::text as lon_lat
FROM   staedte AS sta
WHERE  name='Zürich' LIMIT 1;
"http://maps.google.com/?q=47.36,8.55"
```

Beispiel mit SQL (3)

```
-- Die 5 nächsten Orte von Zürich im Umkreis von 5000m:
SELECT name as stadt,
       round((ST_Distance(sta.geom, sta2.geom)/1000)::numeric, 3)
           as dist_km
FROM   staedte sta,
       (SELECT (ST_Transform(
                   ST_GeomFromText('POINT(8.55 47.37)',4326),21781))

           as geom) sta2
WHERE  ST_DWithin(sta.geom,sta2.geom,10000)
       AND sta.name != 'Zürich'
ORDER  BY 2 LIMIT 5;
```

```
"Aussersihl";0
"Seebach";1.85302748159533
"Zürichhorn";1.85302925037029
"Wiedikon";2.51788777503899
"Witikon";2.51788786503948
```

Beispiel mit SQL (4)

```
-- oder:
SELECT name as stadt,
       round((ST_Distance(sta.geom, sta2.geom)/1000)::numeric, 3) as dist_km
FROM  staedte sta,
      (SELECT geom FROM staedte WHERE name = 'Zürich') sta2
WHERE ST_DWithin(sta.geom,sta2.geom,10000)
      AND sta.name != 'Zürich'
ORDER BY 2 LIMIT 5;

-- oder:
SELECT sta.name as stadt,
       round((ST_Distance(sta.geom, sta2.geom)/1000)::numeric, 3) as dist_km
FROM  staedte sta, staedte sta2
WHERE ST_DWithin(sta.geom, sta2.geom, 10000)
      AND sta2.name = 'Zürich'
      AND sta.name != 'Zürich'
ORDER BY 2 LIMIT 5;
```

Erfahrungen mit PostGIS

- Schnell
  - … jedenfalls schneller als z.B. MS SQL Server (vgl. "HSR Benchmark“)
- Einfacher zu warten
  - (als z.B. Oracle)
- Stabil und produktionsreif
  - genügt Enterprise-Anforderungen
- Gute Dokumentation
  - und guter Support
- Gute Implementierung der Standards (OGC etc.)

PostGIS: Schwächen und Potential

- **Schwächen**
  - ST_Extent (ähnlich wie Count) zu laaangsaaam!
  - Geodaten-Benchmarks sind nötig!

- **Fehlende Features**
  - 3D-Index (siehe Roadmap OpenGeo)
  - Weitere Geometrie-Datentypen, z.B.
    - Point Cloud (Laser Radar; siehe Roadmap OpenGeo)
  - Neue Features siehe Release 2.0

- **Potential**
  - Noch bessere Performance: Extent, Nearest-neighbour Suche
  - Backend von GIS: zuwenig genutzt (nur Speicherung)

Verbreitung von PostGIS

- IGN Paris (France)
- Verkehrsministerium (Niederlande)
- Swisstopo Bern, Kantone Solothurn und Thurgau, Gemeinden (Schweiz)
- Im Hochschulunterricht...
- OpenStreetMap.org (VGI): Migriert im April 2009 von MySQL zu PostgreSQL; (bald PostGIS?)
- Vermutlich in versch. Verwaltungen von Deutschland, Kanada, Brasilien, Südafrika, etc. (bitte melden :->)

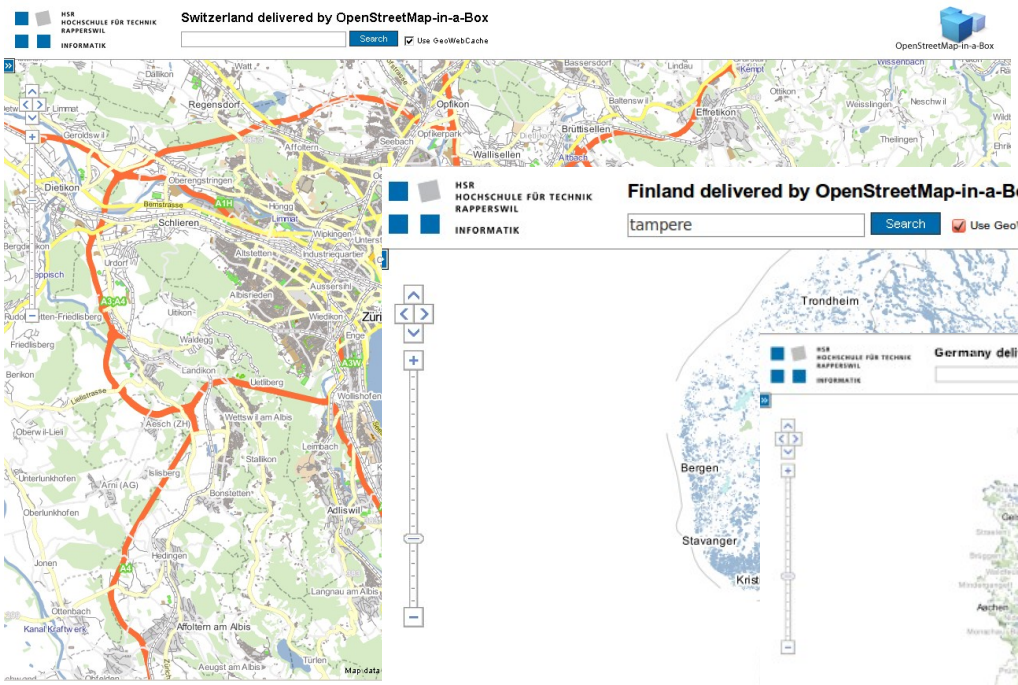
Stadt Uster: Erfahrungen mit PostGIS

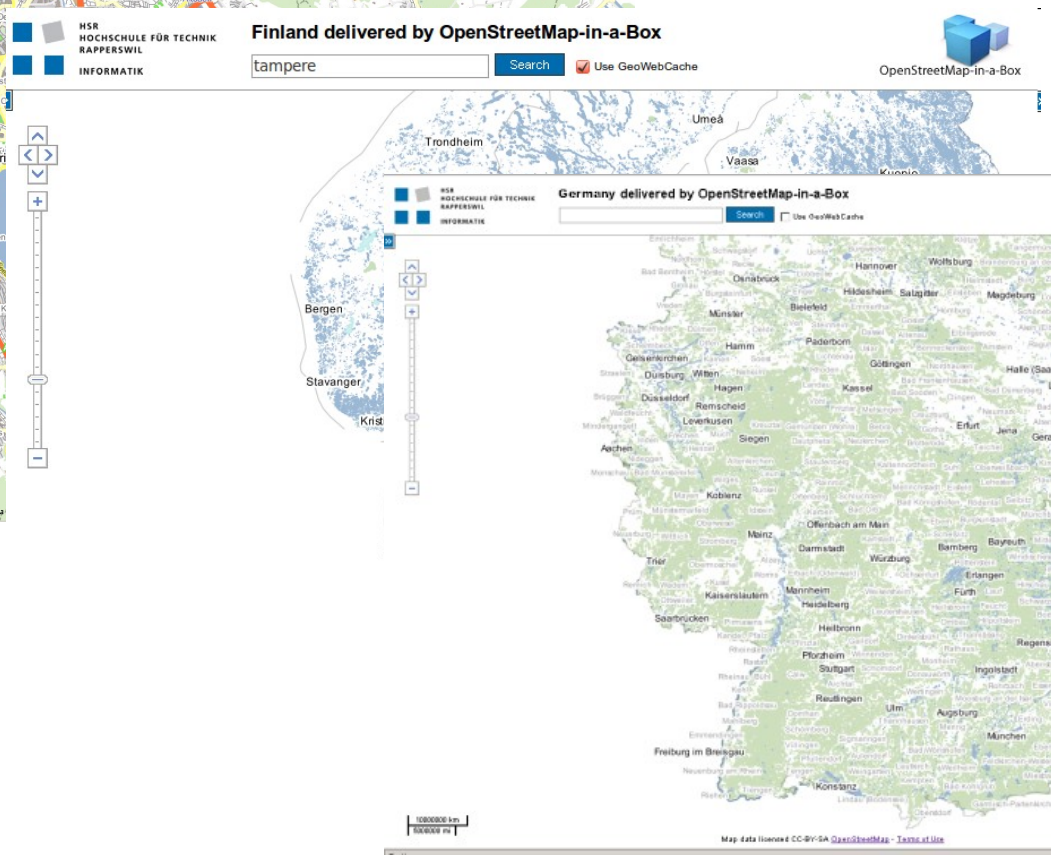
- Zentrales Geodata-Warehouse
- Guter Support in verschiedener GIS-Software
- Sehr stabil, gute Performanz
- Einfache Wartung
- Einfach zu lernen (gute Dokumentation)
- Gutes Ökosystem (Behörden, Hochschulen, Firmen)
- Gut erweiterbar
- Gute und schnelle Weiterentwicklung

Anwendungen von PostGIS: OpenStreetMap-in-a-Box

osminabox - A Ready-Made Map and Geodata Server Including a Highly Configurable Converter Which Synchronizes OpenStreetMap Data

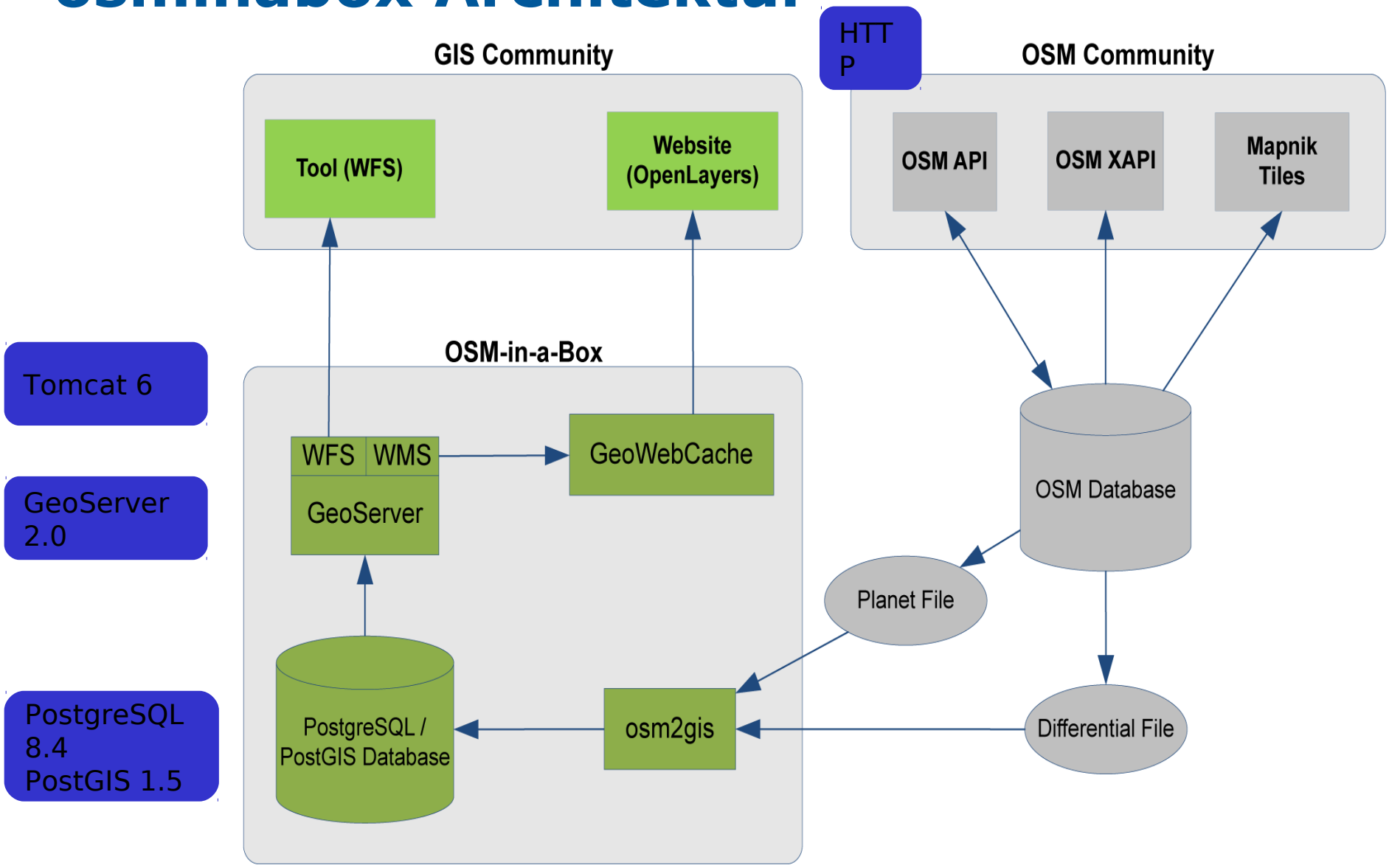
osminabox Website (Showcases)





GeoServer + GeoExt/OpenLayers

osminabox Architektur



osminabox: Erfahrungen mit PostGIS

- **Robust**
- **Schnell**
- **Umfangreiche Funktionen**

Vektor-Raster Analyse in Postgis
Das wktraster-Projekt

Ziele

- Hybride Speicherung/Datenverarbeitung
- Gleiche/ähnliche SQL-Befehle wie bei Vektordatenverarbeitung
- Wachsende Anzahl von Analysemöglichkeiten
- Sehr flexible Speichermöglichkeiten
  - Innerhalb/ausserhalb DB
  - Eine Zelle per Record, ein Tile per Record, gesamter Rasterdatensatz per Record
  - Jeder Record kann andere Georeferenzierung haben
- Metadaten können direkt vom Rasterdatentyp abgeleitet werden

Mögliche Anwendungen

- Backend für Desktop-GIS
- Backend für ModelBuilder
- Backend für Reports/Berichte
- Backend für Web-GIS
- Backend für WPS
- Toolbox für Entwickler

Datenspeicherung in DB

PostGIS
Geospatial Objects for PostgreSQL

Vorteile:

- **Alles in einer DB/Dump**
- **Schnellere Analyse**
- **Rasterbearbeitung kann Teil von DB-Transaktion sein**
- **Besser getestet als Raster ausserhalb DB**

Nachteile:

- **Bläst Dumpfile auf (Backup)**
- **Raster können nicht von beliebigen anderen Tools bearbeitet werden**

Datenspeicherung ausserhalb DB

PostGIS
Geospatial Objects for PostgreSQL

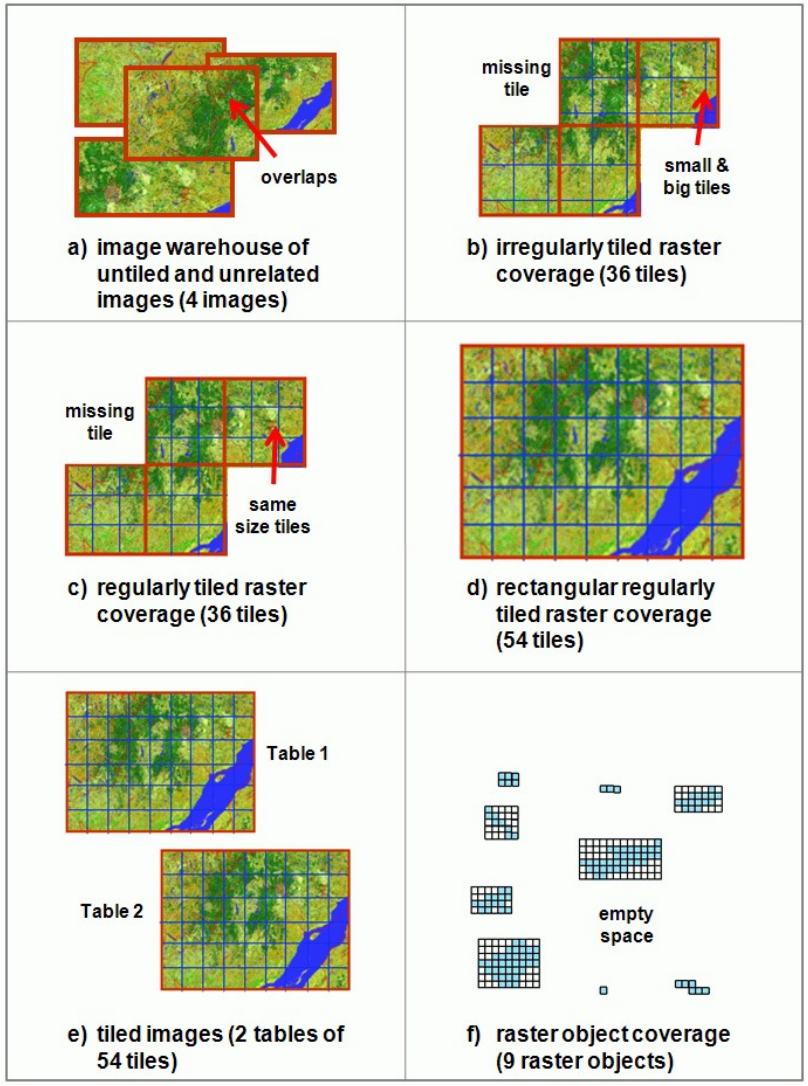
Vorteile:

- Auch andere Applikationen können auf Raster zugreifen
- Einfacheres Backup (kein riesiger DB-Dump)

Nachteile:

- Derzeit ”read-only”
- Langsamerer Zugriff für Analysen
- Weniger getestet
- Nicht im DB-Dump drinnen (Problem der verwaisten Referenzen)
- Kann nicht Teil von DB-Transaktionen sein

Arten der Speicherung



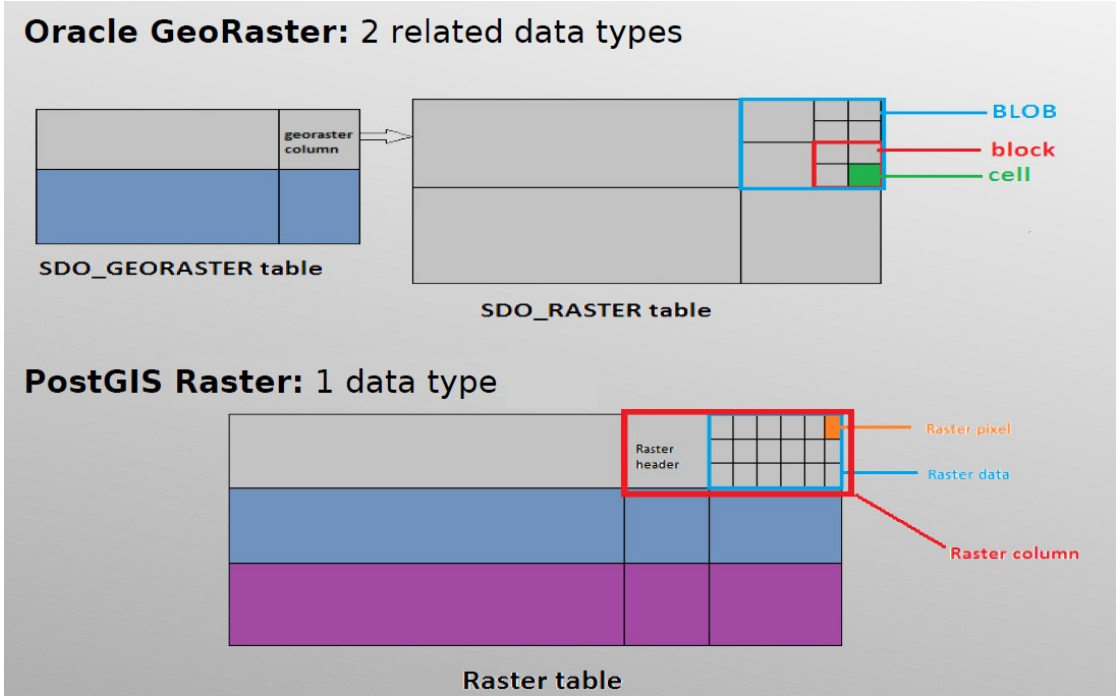
Verschiedene Raster können unterschiedliche:

- Zellgrössen
- Anzahl Bänder
- Ursprünge/Grids
- Projektionen

haben und können trotzdem in einem gemeinsamen SQL-Statement verwendet werden

Metadaten

Keine separaten Metadatentabellen wie bei Oracle



Format, Bänder und Pyramiden

- Speicherformat: wkb
- Beliebige Anzahl Bänder pro Raster
- Pyramiden werden vor dem Einlesen generiert (gdal)
  Müssen im Vorhinein festgelegt werden
- Pyramiden sind in separaten Tabellen gespeichert

Import

- Import über gdal2wktraster (resp. Raster2pgsql)
  - Alle Formate die gdal unterstützt (>100 Formate)
  - Import in 2 Schritten: Erzeugung .sql-File, dann einlesen des SQL-Files

```
gdal2wktraster.py -t dtm.dtmav -M -r dtm_av.txt -o
dtmav.sql -s 21781 -k 200x200
```

- Optional können beim Import auch Pyramiden generiert werden
- Geplant: direkter Import ohne Umweg über SQL-File

Export

- Über gdal (gdal_translate, gdal_warp) – wie alle anderen Rasterdaten auch
- ST_AsBinary()
- In Implementierung: ST_AsText(), ST_AsJPEG, ST_AsTiff, ST_AsPNG
- Vektorobjekte werden beim Export automatisch rasterisiert

Befehlsgruppen

- Raster Management Functions
- Raster and Raster Band Accessors and Constructors
- Raster Pixel Accessors and Setters
- Raster Editors and Band Editors
- Raster Outputs
- Raster Processing Functions
- Raster Operators
- Raster Spatial Relationship Operators

In Implementierung

- ST_Resample(raster): nearest neighbour, bilinear, bicubic
- ST_Reclass(rast|geom, expr_text) zum Reklassifizieren von Rasterdaten
- ST_SelectByValue(raster|geometry, ‘expression’) PixelSelektionen über Bedingungen/ranges
- ST_MapAlgebra(raster|geometry, [raster|geometry,...], ‘mathematical expression’,‘raster’ |’geometry’) - Map-Algebra über 2 verschiedene Datensätze

Ideen für SQL-Abfragen

- Rasterwerte abfragen für Punkte/Flächen/Linien (z.B. Gebäudehöhen, Baumhöhen (DOM/DTMAV))
- Reklassifizierungen
- Map-Algebra
- Geländemodellierung
- Kürzeste Wege finden mit Raster die Gewichte als ”Hindernisse” beinhalten
- Raster ausschneiden aufgrund von Vektorgeometrien (z.B. Alle Hausdächer aus Orthofoto)

Beispiel: Gebäudehöhen extrahieren

```
SELECT MIN((int_result).val), MAX((int_result).val), AVG((int_result).val)
  FROM
  (SELECT ST_Intersection(dom.rast, ST_BUFFER(geb.the_geom,-0.5)) AS int_result
      FROM dtm.dom dom, av.gebaeude geb
       WHERE geb.nummer = 1884 AND ST_Intersects(dom.rast, geb.the_geom)
   ) foo;
```

Beispiel: DTM-Höhe entlang einer Strasse extrahieren

```
SELECT ST_Value(rast,(SELECT
    ST_Line_Interpolate_Point(ST_GeometryN(the_geom,1),0.1)
  FROM av_user.strassenstuecke WHERE text = 'Florastrasse' AND ordnung = 1))
  FROM dtm.dtmav
WHERE (SELECT ST_Line_Interpolate_Point(ST_GeometryN(the_geom,1),0.1)
  FROM av_user.strassenstuecke WHERE text = 'Florastrasse' AND ordnung = 1)
   && rast;
```

Tools/GUI

- Desktop-GIS:
  - QGIS
  - OpenJump
  - gvSig
- Mapserver:
  - UMN Mapserver
  - Geoserver
  - QGIS Server
- Libraries
  - Gdal/ogr

PostGIS-Raster Fazit

- Vektor/Rasteranalyse neu?
  Nein, aber einzigartige Realisierung
- Existierendes SQL-Knowhow kann auch für Rasteranalyse eingesetzt werden
- Transparente SQL-Funktionen/Operatoren für Vektor und Raster
- Berechnungen nur dann wenn nötig und nur für den gerade nötigen geografischen Ausschnitt
- Umfassender als Oracle Georaster

Mögliche Probleme

- Einige Funktionen noch nicht implementiert
- Tilegrösse und Pyramidenparameter können nur beim Import definiert werden
- Out of DB-Raster noch nicht genügend getestet
- Noch zuwenige Clients/Applikationen

PostGIS-Raster Infos/Credits

- Credits:
  - Talks at FOSS4G2010 (http://2010.foss4g.org):
    - Pierre Racine et al.: „Introducing Postgis WKT Raster…“, id=3221
    - Jorge Arévalo: „PostGIS WKT Raster, an Open Source Alternative…“ id=3814
    - Marc Jansen: „What you type is what you see…“, id=3630
  - Obe & Hsu, „PostGIS in Action“, Book by Manning (geplant 1. Quartal 2011, www.manning.com/obe/)

- Weblinks:
  - PostGIS Raster: http://trac.osgeo.org/postgis/wiki/WKTRaster
  - Postgis Raster Tutorial: http://trac.osgeo.org/postis/wiki/WKTRasterTutorial01/
  - PostGIS: www.postgis.org
  - PostGIS ‚stuff‘: www.postgresonline.org
  - GISpunkt Wiki: www.gis.hsr.ch/wiki

Ausblick

PostGIS Version 2

Support- und Buchhinweise

PostGIS Release 2.0 (ca. März 2011)

- Neuer Datentyp Raster:
  - siehe oben
- Neue Funktionen auf 3D Points, z.B.
  - ST_3DDWithin, ST_3DDistance, ST_3DIntersects
  - Geländedaten (topologische 3D-Typen)

Support, Bücher und Web

- **Support:**
  - **Users & Developers: www.postgis.org/support/**
  - **Entscheider D-A-CH: www.postgres-support.ch**
- **Bücher:**
  - **„PostGIS in Action“ von Obe & Hsu (geplant ab 1. Quartal 2011), Manning (siehe unten)**
- **Web:**
  - **PostGIS: www.postgis.org**
  - **PostGIS Blog: www.postgresonline.org**
  - **GISpunkt Wiki: www.gis.hsr.ch/wiki/PostGIS**

Noch nicht genug von PostGIS?

- **Mo. 6. September (heute):**
  - **14:00-14:50, "PostGIS 1.5 and beyond: a technical perspective", Mark Cave-Ayland and Olivier Courtin**
  - **15:20-16:10, "Openstreetmap -> PostGIS -> OpenLayers: Mit offenen Karten ins Web", Hartmut Holzgraefe**
- **Di. 7. September (morgen):**
  - **13:10-14:00, "Discover PostGIS: GIS for PostgreSQL", Vincent Picavet**
  - **17:00 (nach Closing): Birds-of-a-feather Session zu „Benchmarking PostgreSQL and other DBMS”?**
- **Mi. 8. September (übermorgen):**
  - **Workshops von Mark Cave-Ayland zu Introduction und Advanced PostGIS**

Dank / Credits

- Talks at FOSS4G2010 from Pierre Racine et al. sowie von Jorge Arévalo: http://2010.foss4g.org/presentations.php
- Obe & Hsu, „PostGIS in Action“, Manning (www.manning.com/obe/)

Stefan Keller
HSR Hochschule Rapperswil
www.gis.hsr.ch
sfkeller(ät)hsr(dot)ch

Andreas Neumann
GIS-Kompetenzzentrum Uster
www.uster.ch
a.neumann(ät)carto(dot)net