Masterarbeit

Untersuchungen zum Einsatz von Apache CouchDB als Geodatenbank

Prüfer: Prof. Dr.-Ing. Baumann, M.Eng. Kirschke

Gliederung

- Apache CouchDB
- Geodatenbank
- Praktische Beispiele im GIS-Kontext
- Ergebnisse

Apache CouchDB

- **C**luster **o**f **u**nreliable **c**ommodity **h**ardware **D**ata**b**ase
- Gattung: Document Store
  - strukturierte Datensammlungen wie JSON, XML
  - schemalose Ablage; mehr oder weniger unstrukturiert in nichtrelationaler Datenbank
- Features (Auswahl):
  - MapReduce-Framework
  - Multi-Version Concurrency Control (MVCC)
  - HTTP-basierte RESTful API

Dokumente

JSON

```
{
  "Name": "Mustermann",
  "Vornamen": "Anton",
  "maennlich": true,
  "Faecher": [ "BWL",
               "Informatik",
               "Mathematik" ],
  "Benotung":
  {
    "BWL": 3.3,
    "Informatik": 2.3,
  }
}
```

GeoJSON

```
{
  "type": "Polygon",
  "coordinates": [
    [
      [0, 0],
      [1000, 1000],
      [1000, 0],
      [0, 0]
    ]
  ]
}
```

„enge Bindung“ von Sachdaten und Geometrie

Zugriff

- via HTTP API (GET):
  - `http://127.0.0.1:5984/db01/8a871db09abe878ff000a89`

    Host: `127.0.0.1`; Port: `5984`; Datenb.: `db01`; Dokument-ID: `8a871db09abe878ff000a89`
  - entspricht in etwa: `select * from db01 where id="8a871db09abe878ff000a89"`
- umfänglicher Zugriff via cURL:

```
curl -X PUT
http://127.0.0.1:5984/db01/8a871db09abe878ff00698b
-H "Content-Type: application/json" -d
"{\"Semester\":\"3\",
\"Modulart\":\"Wahlpflichtmodul\",
\"Dozent\":\"Baumann\", \"PA\":\"Projekt\"}"
```

Data Processing

- Views (MapReduce)
- Shows
- Lists
- Spatial (GeoCouch)
- Dokumenten-validierung
- Update-Funktionen
- Filter

Map-Funktion:

```
function(doc)
{
  emit(key, value);
}
```

Reduce-Funktion:

```
function(keys, values, rereduce)
{
  return <something>;
}
```

GeoCouch

- entwickelt von V. Mische[1]
- Speicherung in der Spatial-Eigenschaft eines Dokumentes (view)
- R-Tree-Indizierung
- Möglichkeit n-dimensionaler Datenabfragen
- Abfragen über Auswahlrechteck
- URL-Parameter zur Steuerung

[1] GeoCouch: A spatial index for CouchDB, Universität Augsburg, Institut für Geographie, 2010

Geodatenbank

- Speicherung (vier Datenbanken)
  - Rasterdaten (Topographische Karte)
    - Inline Attachment
    - Standalone Attachment
  - Vektordaten (SHP-Daten für Punkte, Linien, Fl.)
    - cURL-Befehlsketten (Stapelverarbeitung)
    - shp2geocouch
    - ogr2ogr (GDAL)
- Datenabfrage
  - thematisch/geometrisch (GeoCouch)

Beispiele I: ein Feature


```
{
    "_id":"000002172",
    "_rev":"1-aecab8c76aa033550a5280679ad0e",
    "type":"Feature",
    "properties":{
        "NAME":"Dessau", "AGS":"15001000", "GEMTEIL":"Ja",
        "GEMEINDE":"Dessau-Roßlau", "VERWGEM":"Dessau-Roßlau",
        "srcname":"GN250", "HOEHEG":62, "EWZG":25071
    },
    "geometry":{
        "type":"Point",
        "coordinates":[12.2418788650386005 66,
                       51.8308546802786906 94]
    }
}
```

Beispiele II: eine thematische Abfrage

```
{
  "language": "javascript",
  "views": {
    "abfrage1": {
      "map": "function(doc) {
        if (doc.properties.GEMTEIL === 'Nein' &&
            doc.properties.EWZG > 1000 &&
            doc.properties.EWZG < 5000 &&
            doc.properties.HOEHEG > 240) {
        emit(
            doc.properties.NAME,
            [doc.properties.VERWGEM, doc.geometry]
        )
       }
      }",
      "reduce": "_count"
    }
  }
}
```

http://127.0.0.1:5984/gn250/_design/abfragen/_view/abfrage1?reduce=false

Beispiele III: JSON-Antwort


```
{
    "total_rows":2,
    "offset":0,
    "rows":[
        {
            "id":"000000881", "key":"Benndorf",
            "value":[
                "Mansfelder Grund-Helbra",
                {
                    "type":"Point",
                    "coordinates":[
                        11.4928704155916,
                        51.571213794011371
                    ]
                }
            ]
        },…
```

Beispiele IV: „Formatierte“ Antworten

- Lists-Antwort als GML-Fragment:

```
<gml:FeatureCollection>
  <gml:Point id="000000881" srsName="EPSG:4326">
    <gml:coordinates>
    11.4928704155916, 51.57121379401137</gml:coordinates>
  </gml:Point>
  <gml:Point id="000000504" srsName="EPSG:4326">
    <gml:coordinates>
    11.139179395695642, 51.6406396434085</gml:coordinates>
   </gml:Point>
</gml:FeatureCollection>
```

- Lists-Antwort in WKT:

```
POINT(11.4928704155916 51.57121379401137)
POINT(11.139179395695642 51.6406396434085)
```

Beispiele V: räuml. Abfrage (GeoCouch)

- Aufgabe:

  innerhalb der Grenzen P_{ul}(11,919231° ö.L., 51,721156° n.Br.), P_{or} (12,810265° ö.L., 51,989251° n.Br.) Orte ermitteln, deren Einwohnerzahl (EWZG) größer als 10.000 ist

```
curl -X PUT -d '{"spatial":{"punkte":"function(doc) {
if(doc.geometry) {
emit([doc.geometry,doc.properties.EWZG],
[doc.properties.NAME,doc.properties.EWZG]); }}"}}'
http://127.0.0.1/_design/abfragen_sp_md
```

http://127.0.0.1:5984/gn250/_design/abfragen_sp_md/_spatial/punkte?start_range=[11.919231,51.721156,10000]&end_range=[12.810265, 51.989251, null]

Praktische Beispiele im GIS-Kontext

- Python
- PHP + JavaScript
- .NET
- WebClient
  - Leaflet
  - JQuery

Python

```
# import der notwendigen Module
import couchdb

# Serververbindung herstellen
couch = couchdb.Server('http://localhost:5984')

# existierende Datenbank auswaehlen
db = couch['vg'] # existing

# Schleife über alle Kreise bzw. kreisfreien Städte
for id in db:
    # Designdokumente ausschließen
    if not id.startswith('_'):
        # Schlüsselwert des Saalekreises ermitteln und speichern
        if db[id]['properties']['GEN'] == 'Saalekreis':
            rs=db[id]['properties']['RS']

# existierende Datenbank auswählen
db = couch['gn250']
# Schleife über alle geogr. Namen
j=0
for id in db:
    # Designdokumente ausschließen
    if not id.startswith('_'):
        #print('aktuelles Dokument: '+id)
        if db[id]['properties']['AGS2'] == rs:
            if db[id]['properties']['EWZG'] > 2000:
                j+=1
                print(db[id]['properties']['NAME']+\
                      ' (EW: '+str(db[id]['properties']['EWZG'])+\
                      '): '+str(db[id]['geometry']['coordinates'][0])+' '+\
```

Orte im Saalekreis mit Einwohnerzahl > 2.000

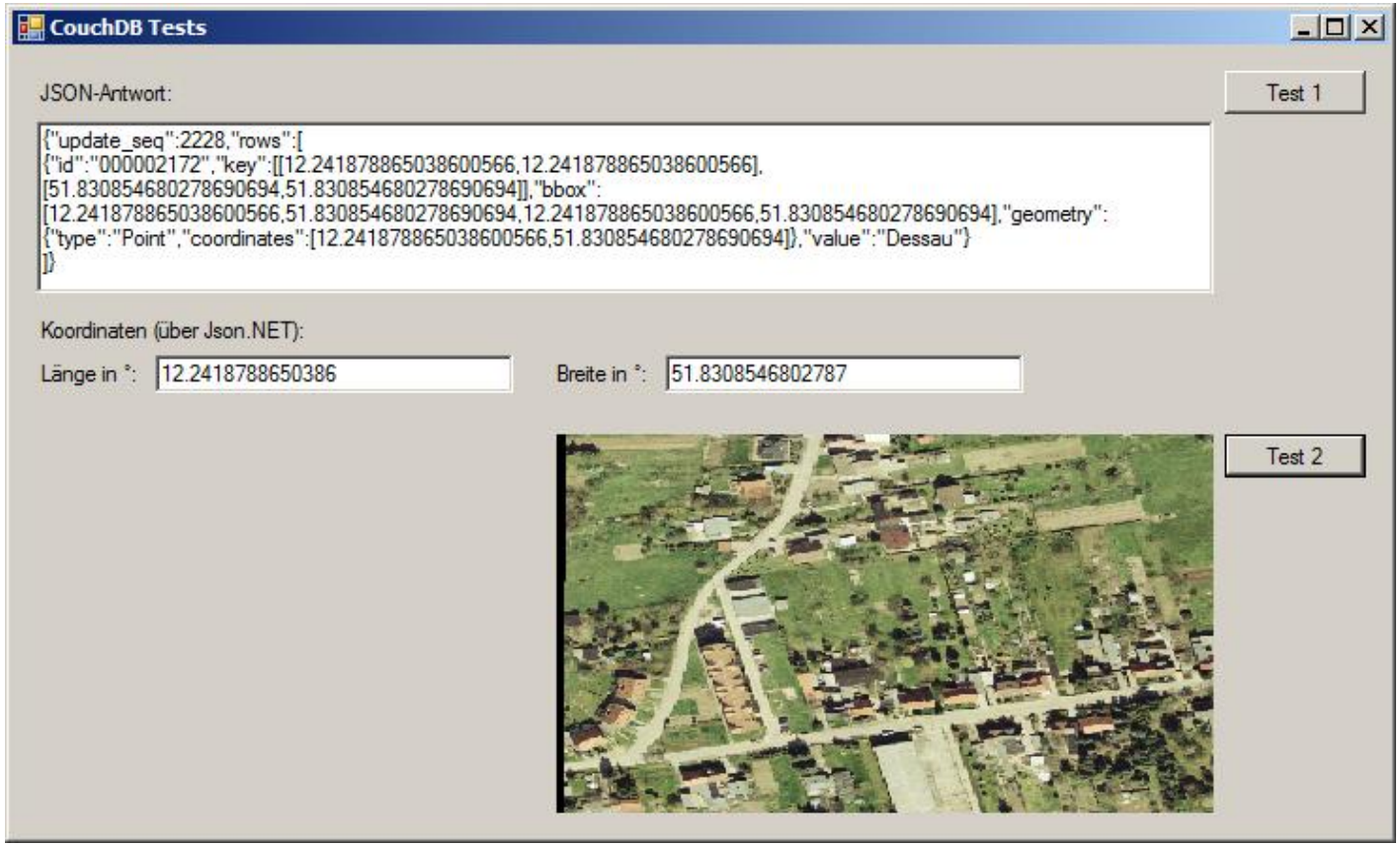
Zugriff via Python library for CouchDB

PHP + JavaScript

Orte als Marker in Google Maps-Karte

Zugriff via Data Access Library PHP-on-Couch

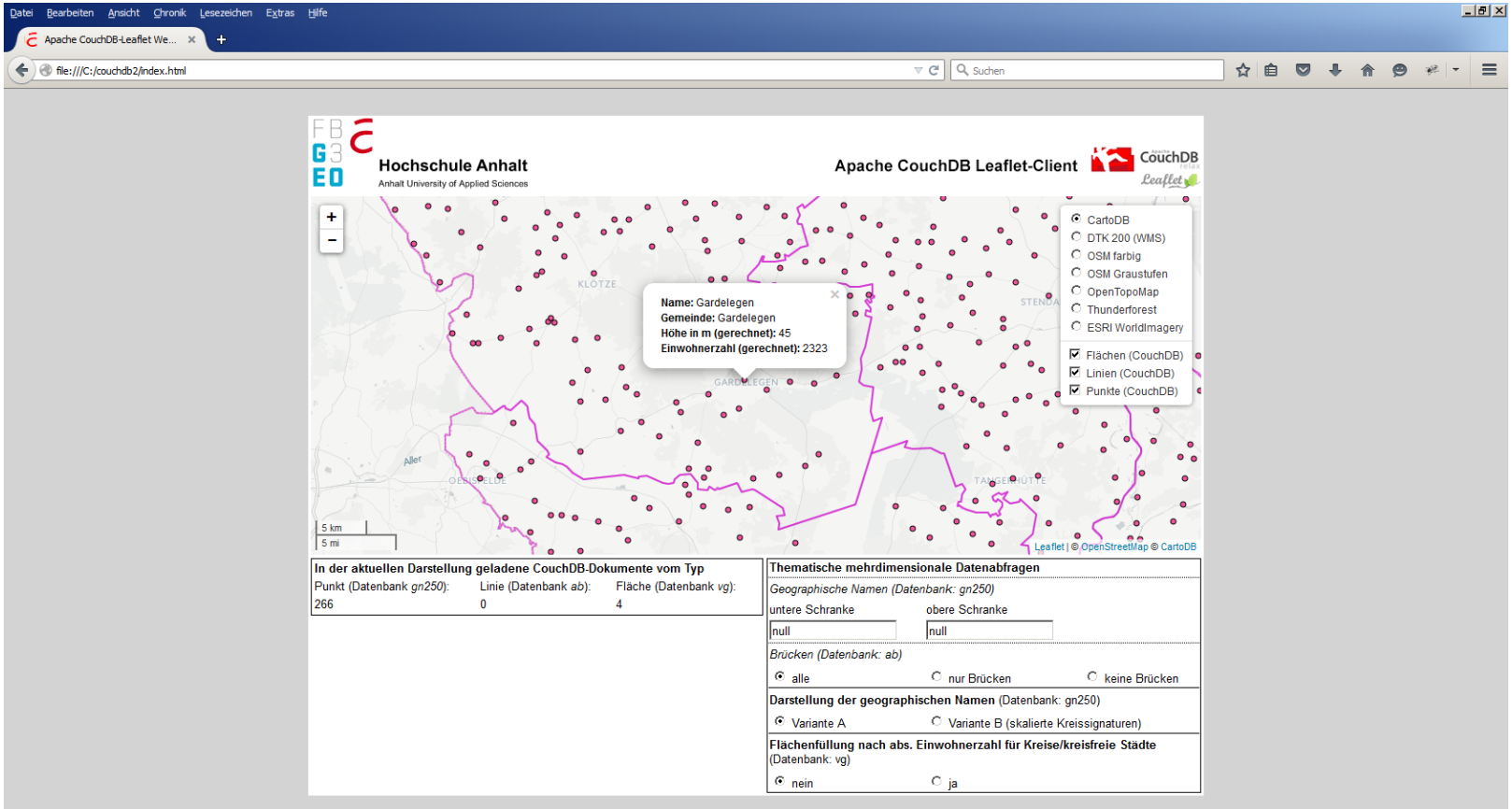
.NET



Parsen und Laden in Windows-Forms-Elemente

.Net-Objekte: HTTPWebRequest / HTTPWebResponse

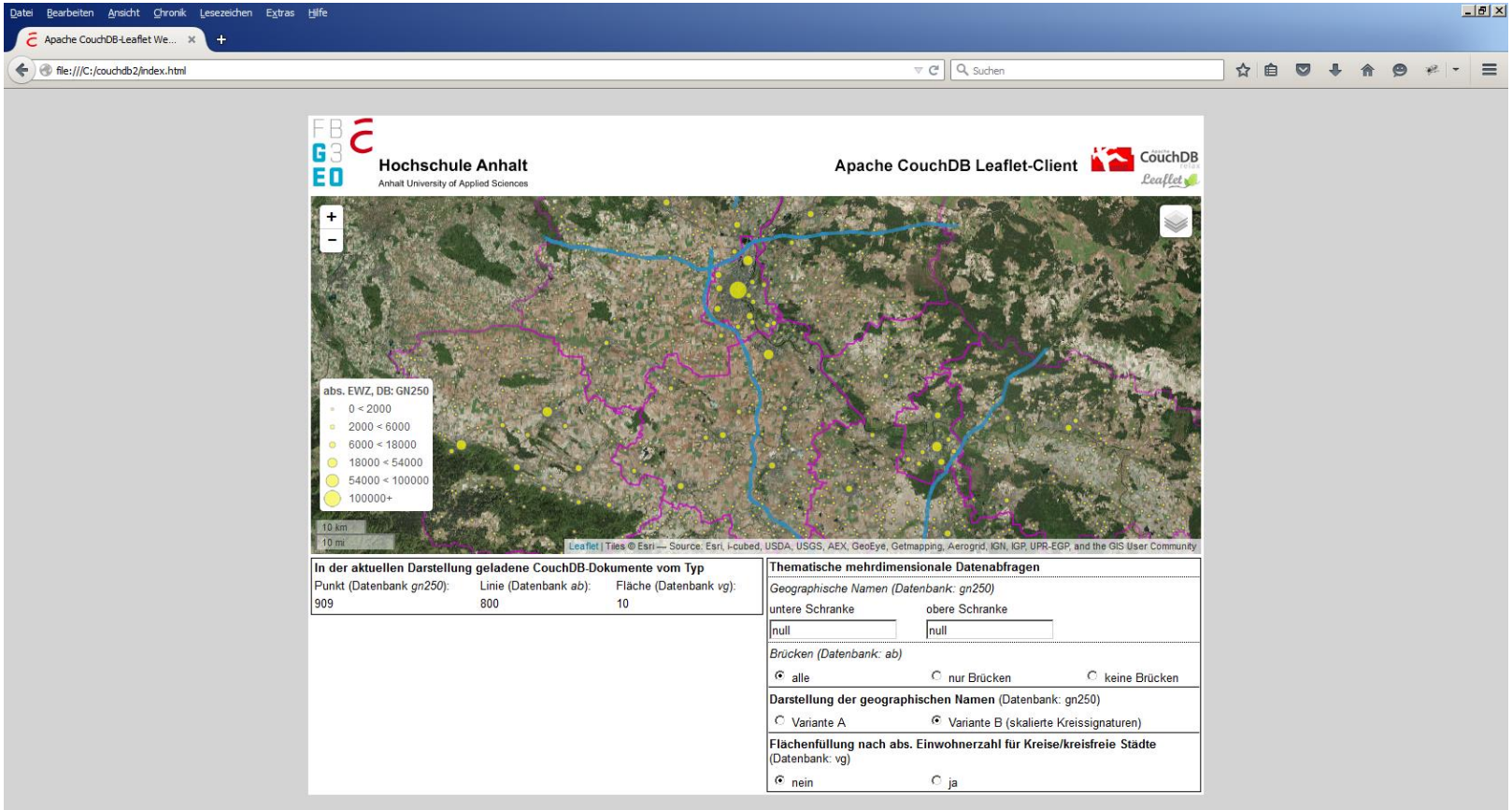
WebClient I



Laden von Punkten und Polygonen

Kopplung via `L.geoJson()` :: `jQuery.ajax()`, JSON als Schnittstelle

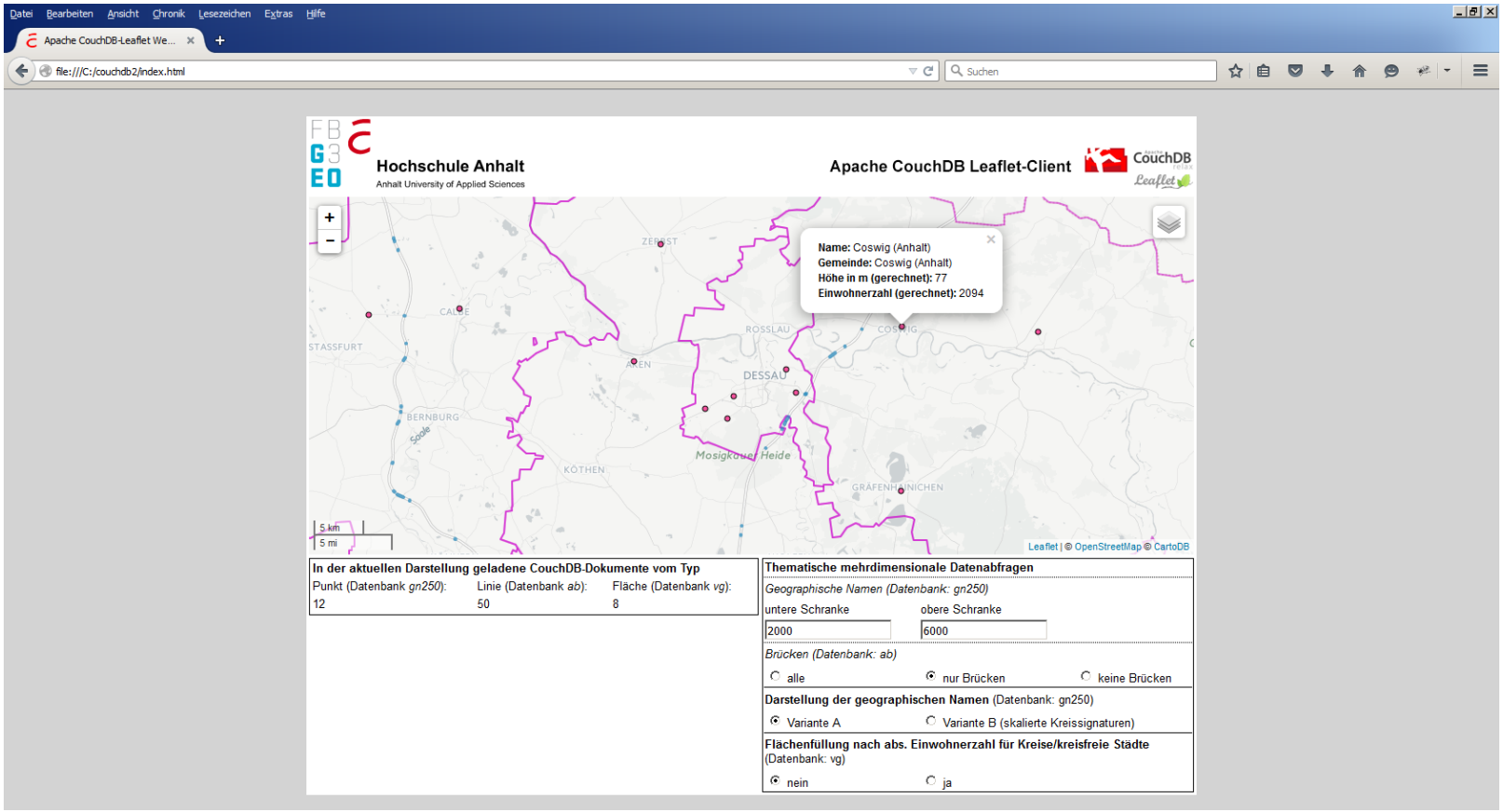
WebClient II



operatives Ändern der kartograph. Darstellung

gelb: Punkte (EW); lila: Flächen (Kreise); blau: Linien (Autobahnen)

WebClient III



n-dimensionale Abfragen

Siedlungen/Punkte: 2000 < x < 6000 EW; Autobahnen/Linien: nur Brücken

Ergebnisse

- hoher Durchsatz, geringe Ladezeiten
- Inbetriebnahme von Apache CouchDB mit GeoCouch auf Ubuntu 14.04 LTS
- Zugriff/Export/Import der Daten leicht möglich
- räumliche Abfrage (GeoCouch) nur über BBOX bzw. Punkt
- keine Features der räumlichen Analyse/Transformation/Topologie
- umständliche oder keine JOINs

Vielen Dank für die Aufmerksamkeit