ARBEITSGRUPPE DATENBANKEN UND INFORMATIONSSYSTEME

Spatial Databases: Project Documentation (orphaned)

SETUP-GUIDE AND DOCUMENTATION FOR SPATIAL-WEATHER-PROJECT

Christian Wirth (4498611)
Miriam Seel (dropped out)

February 28, 2015

Contents

1 Task

The goal of the project is to work with weather data and OpenStreetMap (OSM). The steps to be taken are as follows [1]:

- Find data sources for:
  - Medium-range (3–7 days) and short-range (12–48 hours) weather forecasts
  - Current and historical weather data from weather stations
  - OpenStreetMap data for districts and cities
- Model your data in an ER and store the data in PostGIS.
- Overlay OSM areas with forecast and historical weather data.
- Visualize the probability of forecasts compared to historical weather data on a map.

2 Project

2.1 Responsibilities

Christian:

- PostGIS-server set-up & configuration
- Download & importing OSM-data
- Backend-logic
- Frontend

Miriam:

- Parsing & importing weather-data
- Schema & data-model
- Queries & query-optimisation
- Documentation

[1] Task as stated in the Project-description by Daniel Kressner

2.2 Environment

- virtual server (orphaned)
- PostGIS (orphaned)
- Extensions activated (orphaned)
  - Enable PostGIS (includes raster) -> CREATE EXTENSION postgis;
  - Enable Topology -> CREATE EXTENSION postgis_topology;
  - Fuzzy matching -> CREATE EXTENSION fuzzystrmatch;
    Fuzzy string matching was activated in order to implement a fuzzy string search for locations.
- Frontend: Leaflet
- Backend
  - **Option A:** Express for Nodejs

    Our approach for the Backend follows a guide on Boomphisto' Blog [2], which explains the set-up of a Node-server with Express, a framework providing controllers and libraries for PostGIS as well as Leaflet. The article claims, that Express provides libraries in JavaScript to access the PostGIS-server and returns GeoJSON, which can be directly forwarded. Sadly this project had to be aborted, before we were able to verify this claim and evaluate how much additional work has be done to get it up and running.
    If you follow the guide mentioned above you can start the nodejs-server, but won't see anything apart from a blank page, since we abandoned the project during the set-up of the ejs-based user interface[3] for express
  - **Option B:** Leaflet with yet undefined backend logic in python.

    Accessable by:

    `spatialdb_project/weatherdb/index.html`

    This was our fall-back-approach, in case we would run into essential problems with Option A, since we knew, that Option B already worked for other teams.

[2] http://boomphisto.blogspot.de/2011/07/nodejs-express-leaflet-postgis-awesome.html (last access: 28.02.2015)
[3] http://codeforgeek.com/2014/06/express-nodejs-tutorial/ (last access: 28.02.2015)

2.3 Saving weather-data

Openweathermap offers weather forecasts for free as XML or json. We decided to download the data as json-files, because the size of data is much smaller (http://openweathermap.org/forecast). This Data has a timestamp in Unix-Time, so it will be necessary to convert that format to date-format.


```
{"cod":"200","message":0.0139,
"city":{
  "id":2950159,
    "name":"Berlin",
  "coord":{"lon":13.41053,"lat":52.524368},
  "country":"DE","population":0,
  "sys":{"population":0}},
"cnt":1,
"list":[{
  "dt":1417860000, //unix-time
  "temp":{
    "day":4.46,
    "min":3.79,
    "max":4.46,
    "night":3.79,
    "eve":4.46,
    "morn":4.46},
  "pressure":1023.77,
  "humidity":100,
  "weather":[{"id":600,"main":"Snow","description":"
     light snow",
        "icon":"13d"}],"speed":2.17,"deg":226,"clouds
           ":88,
        "snow":0.25}]}
```

Parameter	**Description**
city.id	City identification
city.name	City name
city.country	Country (GB, JP etc.)
coord.lat	City geo location, lat
coord.lon	City geo location, lon
cnt	Number of lines returned by this API call
dt	Data receiving time, unix time, GMT
temp.day	Day temperature
temp.min	Min daily temperature
temp.max	Max daily temperature
temp.night	Night temperature
temp.eve	Evening temperature
temp.morn	Morning temperature
humidity	Humidity
pressure	Atmospheric pressure
wind.speed	Wind speed, mps
wind.deg	Wind direction, degrees (meteorological)
wind.gust	Wind gust, mps
clouds.all	Cloudiness
weather.id	Weather condition id
weather.main	Group of weather parameters (Rain, Snow, Extreme etc.)
weather.description	Weather condition within the group
weather.icon	Weather icon id
rain	Precipitation volume for last 3 hours, mm
snow	Snow volume for last 3 hours, mm

On the owm-HP a fix json-file exists with all weatherstations (cities) where weather is meassured. It is structured as follows:

```
{"_id":2947416,"name":"Bochum","country":"DE",
  "coord":{"lon":7.21667,"lat":51.48333}}
{"_id":567322,"name":"Chvizhepse","country":"RU",
  "coord":{"lon":40.085278,"lat":43.631111}}
```

At first we created a new file with all the german cities with groovy.The number of rows decreased from 209579 to 28783.

Listing 1: groovy-code for cityfilter

```
class FilterCityList {

  public static void filterCitiesFileByCountryDe(def fileName){

    def file = new File(fileName)
    StringWriter writer = new StringWriter()
    def fw = new FileWriter("/germanCities.txt")
    file.filterLine(writer) { line ->
      line.contains("\"country\":\"DE\"")
    }
    println writer.toString()
    fw.write(writer.toString())
    fw.close()
  }
}


class FilterCityListTest {

  @Test
  public void testFilterCitiesFileByCountryDe() {
    FilterCityList.filterCitiesFileByCountryDe("/city.list.json")
    assertTrue(true);
  }

}
```

With this new file we created all the URLs to get weatherforecasts. This list of URLs is used for the daily download.

2.4 Schema

weatherStation(cityId, cityName, cityCountry, longitude, latitude)
forecastData(weatherstationId, dataTime, tempDay, tempMin, tempMax, tempNight, tempEve, tempMorn, humidity, pressure, windSpeed, windDirection, windGust, cloudiness, weatherCondition, weatherDecription, weatherIcon, rain, snow, insertdate)
weatherData(weatherstationId, currentFlag, dataTime, tempDay, tempMin, tempMax, tempNight, tempEve, tempMorn, humidity, pressure, windSpeed, windDirection, windGust, cloudiness, weatherCondition, weatherDecription, weatherIcon, rain, snow, insertdate)
...

3 Appendix

3.1 Link to Repository

`https://github.com/CrusaderW/spatialdb_project`