



Ortsuche

Preamblel

SQLpostgres

Scharfe Suche
Unschärfe
Suche

PostgreSQL

Geometrie
unaccent
FuzzyStrMatch
Trigramm
FTS

Fazit

Ortsuche mit PostgreSQL und PostGIS

Stephan Wagner

ITos GmbH, CH-9642 Ebnat-Kappel

FOSSGIS
13. Juni 2013

Ortsuche

Preamble

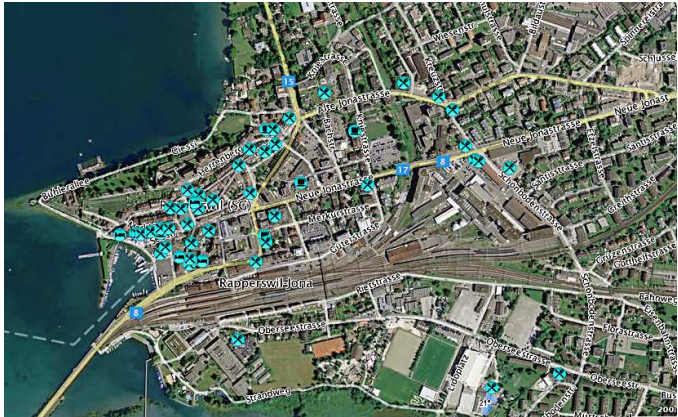
SQLpostgres

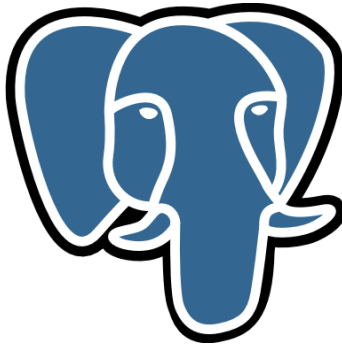
Scharfe Suche
Unschärfe
Suche

PostgreSQL

Geometrie
unaccent
FuzzyStrMatch
Trigramm
FTS

Fazit





Hinweise an den *Mahaut*



Preamblel

Abhängigkeit Ortsbezeichner

Ortsuche

Preamblel

SQLpostgres

Scharfe Suche
Unschärfe
Suche

PostgreSQL

Geometrie
unaccent
FuzzyStrMatch
Trigramm
FTS

Fazit

- Ortskoordinaten
 - Koordinatensystem
 - Einheiten
- Ortsname
 - Sprache
 - Schreibweise
 - Rechtschreibung
 - Wie hei[ss,ß]t das Ding schon wieder?

- Geographisch: (PostgreSQL oder PostGIS-Geometrietypen)
 - Koordinate Ortszentrum (**Punkt**)
 - Umhüllende des Ortes (Linie)
 - Ortsfläche
- Sprachlich:
 - Fribourg (französisch)
 - Frybùrg (alemannisch)
 - Friboua, Fribôrg (franko-provenzalisch)
 - Freiburg, Freiburg Ü(e)chtland
 - Freiburg, Freiburg im Breisgau, Friburg im Brisgau (alemannisch)
- Stichworte: Nähe, Radius, Abstand, Nearest Neighbour, kNN-Gist-Index \Rightarrow Nach Abstand sortiert

- 1 Spracheinstellung und gewähltes Wörterbuch beeinflussen Suchergebnis
- 2 Einbettung (PL/pgSQL-)Funktionen
- 3 Autocomplete

- `ls /usr/share/postgresql/9.2/extension` oder
`SELECT * FROM pg_available_extensions`
listet bereits installierte Erweiterungen.
- `CREATE EXTENSION 'ExtensionName' WITH SCHEMA
'Schema' VERSION 'Version' FROM 'AlteVersion'`
- **Debian:** `aptitude install postgresql-contrib-9.2`
- Ansonsten kompilieren (bsp. PostGIS 2.x)



Preamblel

Versionen

Ortsuche

Preamblel

SQLpostgres

Scharfe Suche
Unschärfe
Suche

PostgreSQL

Geometrie
unaccent
FuzzyStrMatch
Trigramm
FTS

Fazit

- PostgreSQL 9.2
- PostGIS 2.0.3
- pg_trgm 1.0
- fuzzystrmatch 1.0
- unaccent 1.0

- Stephan Wagner
- Geschäftsführer *ITos GmbH*
- Web-(GIS)-Lösungen
- PostgreSQL - Django (SQL, Python, HTML, CSS, JS)
- ⇒ System-Architektur und -Administration



Inhaltsverzeichnis

- 1 **Preamblel**
- 2 **SQLpostgres**
 - Scharfe Suche
 - Unscharfe Suche
- 3 **PostgreSQL**
 - Geometrie
 - unaccent
 - FuzzyStrMatch
 - Trigramm
 - FTS
- 4 **Fazit**



SQLpostgres

Suche nach Spalteninhalt

Ortsuche

Preamblel

SQLpostgres

Scharfe Suche
Unschärfe
Suche

PostgreSQL

Geometrie
unaccent
FuzzyStrMatch
Trigramm
FTS

Fazit

- `SELECT * FROM myTable;`
- `SELECT row2, row1 * 1.5 AS row1new FROM myTable;`



SQLpostgres

Einschränkende Suche: WHERE

Ortsuche

Preamblel

SQLpostgres

Scharfe Suche
Unschärfe
Suche

PostgreSQL

Geometrie
unaccent
FuzzyStrMatch
Trigramm
FTS

Fazit

- `SELECT row1 FROM myTable WHERE row2 = 'OrtsName';`
- `SELECT row1 FROM myTable WHERE row2 > 'OrtsName1' AND row2 < 'OrtsName2';`
- `SELECT row2 FROM myTable WHERE row1 < (SELECT ABS(AVG(row1) - row1) FROM myTable;`



SQLpostgres

Suche Beschleunigen: INDEX

Ortsuche

Preamblel

SQLpostgres

Scharfe Suche
Unschärfe
Suche

PostgreSQL

Geometrie
unaccent
FuzzyStrMatch
Trigramm
FTS

Fazit

- Auf Suchspalten INDEX legen
- Ab PostgreSQL 9.1 GiST kNN-INDEX
- Ab PostgreSQL 9.3 SELECT nur auf INDEX möglich
- **Beachte:**
 - Planer-Ausgabe analysieren (EXPLAIN ANALYSE)
 - impliziter SORT
 - RAM für INDEX, SORT (PG tuning)



SQLpostgres

Einschränkende Suche: CASE WHEN

Ortsuche

Preamblel

SQLpostgres

Scharfe Suche

Unscharfe

Suche

PostgreSQL

Geometrie

unaccent

FuzzyStrMatch

Trigramm

FTS

Fazit

```
SELECT CASE WHEN row1='OrtNameFR' THEN 'OrtNameDE'
           WHEN row2='OrtNameDE' THEN 'OrtNameDE'
           ELSE NONE
        END
FROM myTable;
```

Mustersuche auf Begriff:

- `SELECT * FROM myTable WHERE row2 LIKE 'Ortsna%';`
- `SELECT row1 FROM myTable WHERE row2 ILIKE '%rtsNa%';`
- `SELECT row1 FROM myTable WHERE lower(row2) LIKE '%rtsna%';`

Mustersuche auf ganze Zeichenkette:

- `SELECT * FROM myTable WHERE row1 SIMILAR TO 'Ortsna%';`
- `SELECT * FROM myTable WHERE row2 SIMILAR TO '(Ort|ort)%';`
- `SELECT * FROM myTable WHERE row2 SIMILAR TO '[0,o]rt%';`

Vergleichsoperator	Beschreibung
!	Negation
~	keysensitiver Vergleich
~ *	keyinsensitiver Vergleich

Suche in etwa 'Karlstrasse', Strasse in der Schreibweise ß:

- `SELECT * FROM myTable WHERE row2
'[C,c,K,k]arl+[ß]+$';`



PostgreSQL

Über SQL hinaus

Ortsuche

Preamblel

SQLpostgres

Scharfe Suche
Unschärfe
Suche

PostgreSQL

Geometrie
unaccent
FuzzyStrMatch
Trigramm
FTS

Fazit

SQLpostgres

PostgreSQL

- Koordinatensystem
- Einheiten
- **PostgreSQL:**
 - Punkt (*point*), Gerade (*lseg*), Linie (*path*), Polygon (*polygon*), Rechteck (*box*), Kreis (*circle*)
- **PostGIS:**
 - GiST-kNN-Index mit `<->` als Distanzoperator
 - `SELECT ort FROM ortsnamen ORDER BY geom <-> st_setsrid(st_makepoint(-90,40),4326) LIMIT 10;`
 - \Rightarrow Die 10 zum Testpunkt nächstgelegenen Punkte

- Extension aus dem Umfeld von FTS
- Datei mit Konvertierungsregeln unter
\$SHAREDIR/tsearch_data/
- Filter, welcher Lexeme (~ Begriffe) von Akzenten befreit
- **Abfrage:** `SELECT unaccent('Freiburg im Üechtland');`
⇒ Freiburg im Uechtland



FuzzyStrMatch

PG-Extension

Ortsuche

Preamblel

SQLpostgres

Scharfe Suche
Unschärfe
Suche

PostgreSQL

Geometrie
unaccent
FuzzyStrMatch
Trigramm
FTS

Fazit

- Soundex
- (Double-)Metaphone
- Levenshtein



FuzzyStrMatch

Soundex

Ortsuche

Preamble

SQLpostgres

Scharfe Suche
Unschärfe
Suche

PostgreSQL

Geometrie
unaccent
FuzzyStrMatch
Trigramm
FTS

Fazit

- **Algorithmus:** Robert Russell und Margaret Odel, 1918
- Phonetische Indizierung nach englischsprachigem Klangmuster
- Ergibt auch für Deutsch und Französisch brauchbare Ergebnisse
- **Code:** Erster Buchstabe plus drei Ziffern, bsp. W-213 für Wikipedia.
 - **Wort** mit mehr Buchstaben als benötigt \Rightarrow Schnitt
 - **Wort** mit wenig Buchstaben als benötigt \Rightarrow Nullen
- `SELECT SOUNDEX('Fribourg') \Rightarrow F616`
- `'Fryburg' \Rightarrow F616 \Leftarrow 'Freiburg im Üechtland'`
- `SELECT * FROM ort WHERE difference(ort.nm, 'Friburg') > 2;`
- Britney Spears \Rightarrow B635 S162 \Leftarrow Bewährten Superzicke 😊



FuzzyStrMatch

(Double-)Mataphone

Ortsuche

Preamble

SQLpostgres

Scharfe Suche
Unscharfe
Suche

PostgreSQL

Geometrie
unaccent
FuzzyStrMatch
Trigramm
FTS

Fazit

- **Algorithmus:** Lawrence Phillips, 2000
- Phonetische Indizierung nach englischsprachigem Klangmuster
- Absichtlich phonetisch unscharfe Darstellung (Sprechweisen)
- Unterscheidet feingranularer als SOUNDEX
- Gewichtung der Operationen möglich
- `SELECT metaphone('Fryburg im Üechtland', 7);` ⇒ `FRBRKMX`
- `SELECT dmetaphone('Fryburg im Üechtland');` ⇒ `FRPR`

- **Algorithmus:** Wladimir Levenshtein, 1965
- **Distanz:** Anzahl von Einzeloperationen, um Zeichenkette 1 auf 2 umzuwandeln
- **Haupteinsatzbereich:** Duplikatserkennung, Rechtschreibprüfung
- Bei grössen Datensätzen langsam: $O(N \times M)$
- `SELECT levenshtein('Fribourg', 'Freibourg');` \Rightarrow 1
- `SELECT levenshtein('Fribourg', 'Fryburg im Üechtland');` \Rightarrow 15

- **Module:** pg_trgm
- **Trigramm:** Zerlegung eines Textes in 3-Zeichen-Fragmente (n-Gramm)
- **Trigramme:** Zur Weissagung dienende, altchinesische Symbole
- **Dice-Koeffizient d :** Anteil übereinstimmender N-Gramme zweier Terme ($0 \leq d \leq 1$)

- $$d(a, b) = \frac{2|T(a) \cap T(b)|}{|T(a)| + |T(b)|}$$

- **Beispiel:**

- Term a = "wirk", Term b = "work"

- $T(a) = \{\{w, \{wi, wir, irk, rk\}, k\}\}$

- $T(b) = \{\{w, \{wo, wor, ork, rk\}, k\}\}$

- $T(a) \cap T(b) = \{\{w, k\}, rk\}$

- $d(wirk, work) = \frac{2 \cdot 3}{6+6} = \frac{1}{2}$

d.h., Die Ähnlichkeit oder Distanz beträgt 0.5

- Seit PostgreSQL 8.3 fester Bestandteil
- Resultatmenge: *Dokumente*, welche mit dem Suchbegriff möglichst nah Übereinstimmen
- Basis-Funktionalität:
 - ① *Tokens* auf *Lexeme* reduziert, Stopwords gefiltert: *Bréil sur Royas* \Rightarrow *breil roya* (Datentyp *tsvector* oder *tsvector*)
 - ② *GIN* oder *kNN-GiST*-Index auf Spalte *Lexem*
 - ③ Suchanfrage mit *to_tsquery()* und dem Matching-Operator *&&*
 - ④ Ausgabe des Suchergebnisses nach Relevanz

- Simple, **Synonym**, **Thesaurus**, Ispell, **Snowball** (Stemming-Algorithmus)
- *ls /usr/share/postgresql/9.2/tsearch_data*
- CREATE TEXT SEARCH DISCTIONARY ...
- **Synonym**: Ur-Begriff \sqsubset Synonym
- **Thesaurus**: Ur-Satz : Synonym-Satz

- ❶ *Distanz* verbindet Ortsnamensuche und Ortskoordinatensuche
- ❷ Ortsnamensuche mit Standard-SQL quasi statisch
- ❸ PostgreSQL bietet Alternativen:
 - Soundex, Metaphone
 - Levenshtein
 - Trigramm
 - FTS
 - unaccent

Ortsuche

Preamblel

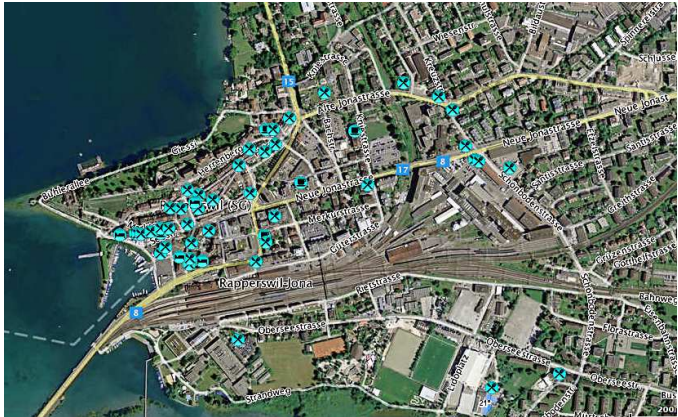
SQLpostgres

Scharfe Suche
Unscharfe
Suche

PostgreSQL

Geometrie
unaccent
FuzzyStrMatch
Trigramm
FTS

Fazit



oder
FOSSGIS e.V. Sektempfang