



Der richtige Treffer

Aus einer Menge an Orten soll der Nutzer möglichst einfach den nächstgelegenen finden. Zeit, sich mit der geografischen Suche in PHP zu beschäftigen.

Für die Arbeit mit Standort-Informationen gibt es eine Reihe von APIs/Datenquellen und Bibliothek. Es kann beispielsweise durchaus aufschlussreich sein (mit gewisser Treffergenauigkeit) die Besucher einer Website anhand ihrer IP-Adresse in der passenden Landessprache zu begrüßen oder eingegebene Straßen- und Orts-Angaben gegen eine Datenbank zu validieren.

Hierzu verfügt zum Beispiel das Projekt Open Street Map über eine wahre Flut von Geo-Informationen – neben Straßen beispielsweise auch über die genauen Aufstellplätze vieler Briefkästen oder Apotheken. Möchte man aber nur mal kurz eine Postleitzahl oder einen Ort nachschlagen, empfiehlt es, sich aufgrund des Datenumfangs eventuell auf andere Quellen zurückzugreifen. Ein Klassiker in Bezug auf Daten zu Postleitzahlen,

Länder, Bundesländern, Orten und Städten/Gemeinden ist Open Geo DB (opengeodb.giswiki.org/wiki/OpenGeoDB). Die hier bereitgestellten Daten enthalten neben Deutschland die benachbarten Länder Belgien, Litauen, Österreich und die Schweiz. Sie werden als SQL-Dumps zum Import in eine lokale Datenbank bereitgestellt. Die Basisarbeit ist durch Download und Einspielen der benötigten Dateien schnell erledigt.

1. *opengeodb-begin.sql*

Diese Datei enthält die Grundstrukturen der Datenbank. Sie stellt die notwendigen Tabellen und Feld-Definitionen bereit.

2. *AT.sql* - Österreich

BE.sql - Belgien

CH.sql - Schweiz

DE.sql - Deutschland

LI.sql - Lichtenstein

Je nach gewünschtem Land können hier die einzelnen Daten importiert werden. Hierbei handelt es ganz allgemein um Standorte (Locations), Koordinaten (coordinates) und damit verbundene Daten (Text- Werte und numerische Werte). Hierbei liegen die Angaben teilweise auch in mehreren Sprachen vor.

3. *opengeodb-end.sql*

Zur einfacheren Suche werden hier die Indizes auf die soeben importierten Tabellen angelegt.

4. *changes.sql*

Letzte Änderungen werden in einer separaten Datei geliefert und helfen dann beim Update.

5. *extra.sql*

Zusätzliche Daten wie etwa Kontinente finden sich in diesem SQL-Dump.

6. *AThier.sql*

BEhier.sql

CHhier.sql

DEhier.sql

LIhier.sql

Die Hierarchiedaten enthalten die eigentlichen Strukturen. Hier sind die Beziehungen und Stufen der einzelnen Einträge definiert. Jene Tabelle gibt vor, bei welchen Einträgen es sich beispielsweise um eine Stadt handelt und in welchem Bundesland diese zu finden ist.

Hierarchische Daten

Am interessantesten sind hierbei die Hierarchie-Daten, welche Sie in der Tabelle *geodb_hierarchies* finden. Ein Standort (location) wird hierbei über *loc_id* angegeben, dessen Hierarchie-Stufe über das Feld *level*. Um einen einfachen Zugriff auf übergeordnete Stufen zu ermöglichen, sind in weiteren Feldern *id_lv11* bis *id_lv19* außerdem alle übergeordneten Stufen zu finden – zu einer Stadt also zum Beispiel die Location-IDs für Kontinent, Land, Bundesland und Kreis.

Um über die Datenbank auch historische Datenstände abbilden zu können, existieren in den verschiedenen Tabellen der Open Geo DB außerdem noch Spalten die angeben, von/bis wann jene Informationen (wie die Zugehörigkeit zu oder der Name eines Kreises) gültig sind. Dies muss ent-



Offene Datenbank: OpenGeo bietet für Deutschland, Österreich, die Schweiz und Litauen Geoinformationen.

sprechend beachtet werden, wenn man jeweils mit den aktuellen Daten arbeiten möchte.

Die Typen der Standorte lassen sich anhand der *loc_id* der Tabelle *geodb_locations* zuordnen. Deren Bedeutung ist wie folgt:

- » 100100000 Erdeitel
- » 100200000 Staat/Land
- » 100300000 Bundesland
- » 100400000 Regierungsbezirk
- » 100500000 Landkreis
- » 100600000 Politische Gliederung
- » 100700000 Ortschaft
- » 100800000 Postleitzahlgebiet
- » 100900000 Ortsteil

Nachzulesen ist das auch unter opengeodb.giswiki.org/wiki/OpenGeoDB_-_Dateninhalt. Die als Wiki organisierte Website ist auch ansonsten eine gute Informationsquelle.

Pro Standort existieren dann verschiedene Angaben je nach Typ in Tabellen *geodb_textdata* beziehungsweise *geodb_intdata* oder *geodb_floatdata*. Hier einige Beispiele:

- » 500100002 Sortiername
- » 500100000 Name
- » 200200000 Breitengrad (latitude)
- » 200300000 Längengrad (longitude)
- » 500300000 Postleitzahl
- » 500500000 KFZ-Kennzeichen

Für die Stadt „Neuss“ können wir auf diesem Wege in *geodb_textdata* einen Eintrag mit dem Namen (500100000) „Neuss“ finden, welcher in *geodb_locations* als „Politische Gliederung“ (100600000) markiert ist. Dieser besitzt (derzeit) die Location-ID 21554.

Ein Blick in die Hierarchie-Tabelle *geodb_hierarchies* führt zu einem Eintrag der angibt, dass es sich hier um einen eine Gliederung auf Ebene (Level) 6 handelt und die übergeordneten Ebenen-IDs 104,105,117,181,195,21554 lauten. Hierzu lassen sich dann anhand der DB ebenfalls beispielsweise die Namen



Hochsicherheits-Webserver.

internet24 bietet mit seinen extrem leistungsfähigen Webservern, hohen Sicherheitsstandards und dem exzellenten Experten-Service optimale Bedingungen für individuelle Server-Projekte sowie für dedizierte Root-, managed und virtuelle Server.

- gespiegelte Festplatten
- Datensicherung frei konfigurierbar
- individuelle Firewall-Lösungen
- Serverüberwachung und -entstörung (24/7/365)
- hocheffektiver Spam-/Virensfilter
- garantierte Verfügbarkeit:
99,99 % Internet-Anbindung,
99,5 % Hardware

Root ab mtl.
79,00 EUR*

Managed ab mtl.
139,00 EUR*

*zzgl. einmalige Einrichtungsgebühr von 149,- EUR


```
?>
```

Sofern sie eine größere Anzahl von Zugriffen oder eine Massen-Geokodierung von Daten planen, ist zu beachten, dass Geo Names derzeit ein Limit von 30.000 Anfragen pro Tag/IP sowie 2.000 Anfragen pro Stunde hat. Bei größeren Abfragemengen oder gehobenen Ansprüchen an Verfügbarkeit/Geschwindigkeit empfiehlt es sich allerdings ohnehin, mit lokalen Daten zu arbeiten.

Umkreissuche

Um die Entfernung zwischen jenen Städten zu ermitteln, benötigt man etwas Mathematik, um den Abstand auf einer Kugel zu berechnen – in unserem Fall der Erde. Aus den Winkelangaben für Längen- und Breitengrad zusammen mit dem Erdradius (6.371 km) lassen sich die Punkte im 3-dimensionalen Raum abbilden. In PHP ermitteln wir hierzu zum Gradmaß Werte im Bogenmaß und bestimmen die Werte für X/Y/Z

```
<?php
```

```
$lambda = $lon * pi() / 180;
```

```
$phi = $lat * pi() / 180;
```

```
$Erdradius = 6371;
```

```
$geoKoordX = $Erdradius * cos($phi) *  
cos($lambda);
```

```
$geoKoordY = $Erdradius * cos($phi) *  
sin($lambda);
```

```
$geoKoordZ = $Erdradius * sin($phi);
```

```
?>
```

und verwenden diese dann zur Berechnung des Abstands:

```
<?php
```

```
$Entfernung = 2 * $Erdradius *
```

```
arcsin(  
sqrt(  
pow($x1 - $x2, 2) // Hinweis: pow  
($Basis, $Exponent) berechnet  
+ pow($y1 - $y2, 2) // „$Basis hoch  
$Exponent“.  
+ pow($z1 - $z2, 2) // Hier also  
„($z1 - $z2)^2“  
)/ (2 * $Erdradius)  
);
```

```
?>
```

Sofern man zu einem gegebenen Ursprungspunkt alle Einträge innerhalb der Tabelle `geodb_coordinates` finden möchte die innerhalb eines bestimmten Radius `$r` liegen, sind jene Berechnungen auch direkt in SQL möglich.

```
<?php
```

```
// ...
```

```
$sql .= 'WHERE ( (2 * $Erdradius) *  
,
```

```
ASIN(  
SQRT(  
POWER(, . $UrsprungX .'^ - , .  
$Erdradius . , * COS(lat * PI() /  
180) * COS(lon * PI() / 180), 2)  
+ POWER(, . $UrsprungY .'^ - , .  
$Erdradius . , * COS(lat * PI() /  
180) * SIN(lon * PI() / 180), 2)  
+ POWER(, . $UrsprungZ .'^ - , .  
$Erdradius . , * SIN(lat * PI() /  
180), 2)  
)/ , . (2 * $Erdradius) . ,  
) <= , . $r ;  
?>
```

Gewinnen mit MeinTelefonbuch

Anzeige

Mit MeinTelefonbuch startet DasTelefonbuch ein neues, innovatives Angebot innerhalb seiner Online-Plattform dastelefonbuch.de.

Private Nutzer haben mit diesem kostenlosen Service erstmalig die Möglichkeit, ihre persönlichen Kontakte in einem Online-Adressverzeichnis zu führen, zu aktualisieren und zu synchronisieren. Dabei bietet MeinTelefonbuch Zugriff auf den kompletten Adressdatenbestand von rund 30 Mio. Kontakten von DasTelefonbuch.



Persönliches Adressbuch

Sichere Archivierung von privaten Adressen, Rufnummern, E-Mail-Adressen, geschäftlichen Kontakten, Messenger-Daten oder Social-Network-Profile: Mittels passwortgeschütztem Login lassen sich diese Daten zu jeder Zeit, von jedem Rechner mit Internet-Anschluss aus abrufen oder mit anderen elektronischen Adressverzeichnissen synchronisieren. Zusätzlich kann der Nutzer seine eigenen Kontaktdaten für jedes Mitglied von MeinTelefonbuch individuell differenziert freigeben. Än-

dern sich Daten von so verknüpften Kontakten, werden diese im eigenen Verzeichnis automatisch aktualisiert.

Direkte, teils kostenlose Verbindungen

Mit nur einem Mausklick kann über MeinTelefonbuch eine direkte Verbindung (Anruf, SMS, E-Mail) zu jeder in Deutschland vergebenen Rufnummer hergestellt werden. Festnetzverbindungen und E-Mail-Versand sind **grundsätzlich kostenlos**. Ein Startguthaben und sogenannte Credits bieten den Usern ein Kontingent kostenfreier Mobilfunkverbindungen an.

Privater Selbsteintrag – auf Wunsch verschlüsselt

Die Veröffentlichung und Verwaltung der eigenen Kontaktdaten auf www.dastelefonbuch.de wird vereinfacht: Änderungen werden zeitnah auf den neuesten Stand gebracht. Neben den Basiskontaktdaten können Informationen, wie Messenger-, VoIP- oder Community-Daten, Internet-Adressen oder Fotos ergänzt werden. Mit der Verschlüsselungsoption ist man erreichbar, ohne Rufnummern oder E-Mail-Adressen preiszugeben.

GEWINNSPIEL:

Mit MeinTelefonbuch können Sie jetzt ein iPad 16 GB WiFi + 3G, exkl. Vertrag gewinnen, wenn Sie folgende Frage beantworten:

Auf wie viele öffentliche Adressdaten haben Sie von www.meintelefonbuch.de aus Zugriff?

- A) ca. 2 Mio.
- B) ca. 30 Mio.
- C) ca. 100 Mio.

Die Lösung bitte per Mail an meintelefonbuch@wekanet.de Einsendeschluss ist der 31.12.2010

Keine Wandlung oder Auszahlung der Gewinne. Die Teilnahme ist für Mitarbeiter der betreffenden Firmen untersagt. Der Rechtsweg ist ausgeschlossen.





Informationsquelle: Unter www.mamat-online.de/umkreissuche/opengeodb.php finden Sie eine gute Einführung in die Berechnungen für Umkreissuchen.

```
(2 * $Erdradius), 2);
```

```
?>
```

Eine interessante Informationsquelle zur Umkreissuche ist die Website von Philipp Mamat: www.mamat-online.de/umkreissuche/opengeodb.php.

Dies ist allerdings über die Datenbank sehr aufwändig, da für jeden potenziellen Treffer die Berechnungen durchgeführt werden müssen. Die Berechnungen werden hierbei vom MySQL-Server durchgeführt und können erst dann mit dem gesuchten Radius verglichen werden. Eine erhebliche Optimierung ist möglich, wenn man die Koordinaten für die kartesischen Koordinaten bereits in dieser Form in der Datenbank abspeichert, sodass jene Berechnungen bei der späteren Abfrage eingespart werden können.

```
<?php
$sql = ,UPDATE data SET ,
,kartX = ,.$Erdradius . , * COS
(lat * PI() / 180) * COS(lon * PI()
/ 180).',
,kartY = ,.$Erdradius . , * COS
(lat * PI() / 180) * SIN(lon * PI()
/ 180).',
,kartZ = ,.$Erdradius . , * SIN
(lat * PI() / 180);
?>
```

Außerdem kann ein Teil der Berechnung auf die rechte Seite des Vergleich verlagert werden, sodass diese Berechnung nur einmalig ausgeführt wird. Dann vereinfacht sich die Abfrage zu:

```
<?php
// ...
$sql .= ,WHERE
POWER( , . $UrsprungX .' - KoordX, 2)
+ POWER( , . $UrsprungY .' - KoordY,
2)
+ POWER( , . $UrsprungZ .' - KoordZ,
2)
<= , . pow(2 * $Erdradius * sin($r /
```

Bei einer sehr großen Anzahl von zu prüfenden Koordinaten und einem verhältnismäßig kleinen Radius bestünde eine weitere Optimierung darin, um den Ursprungspunkt herum einen Würfel (minimale/maximale Werte für KoordX/KoordY/KoordZ) zu bestimmen und in der Datenbank einen Index über jene drei Spalten zu erstellen. Das Ergebnis wäre dann, dass nur noch die Punkte innerhalb jenen Bereichs überhaupt in die Betrachtung (und die Berechnung der Quadrate) Einzug halten müssen.

Ausblick

Das Arbeiten mit Geodaten stellt sehr interessante Möglichkeiten bereit. Wer neben den hier vorgestellten Basisdaten wie zum Beispiel Straßen oder Postleitzahlen tiefer einsteigen möchte, findet etwa im Rahmen des OpenStreetMap-Projekts eine reichhaltige Fülle an Geodaten bis hinunter auf Straßenebene, Geschäfte, Häuser, Briefkästen oder sogar Strommasten. Die Arbeit mit jenen Daten ist allerdings aufgrund der Fülle anfangs durchaus etwas unhandlich und erfordert Einarbeitung sowie die notwendigen Ressourcen zur Verarbeitung der Daten. Wer gewonnene Daten auf einfache Weise auf einer Website grafisch darstellen möchte, findet

Galileo: Die Erde ist eine Kugel
(Quelle Wikipedia, Demon-DeLuxe (Dominique Toussaint), de.wikipedia.org/w/index.php?title=Datei:Sphere_3d.png).

bei Openlayers vielfältige Möglichkeiten zur Arbeit mit interaktiven Karten. Über eine einfache JavaScript-API lassen sich so verschieb- und zoombare Karten in eine Website integrieren (sogenannte Slippy-Maps) und mit weiteren Daten wie Punkten oder Flächen und unterschiedlichen Layern flexibel erweitern.

Wie bei der Arbeit mit allen Daten aus fremden Quellen sei aber insbesondere auch bei der Arbeit mit Geodaten darauf hingewiesen, die Lizenzen der jeweiligen Quellen zu beachten. Gerade bei der Nutzung von kommerziellen Datenquellen zur Geokodierung oder nur in bestimmten Grenzen nutzbaren Daten/Diensten (etwa Google Maps) ist die weitere Verwendung von gewonnenen Daten oft stark eingeschränkt. So dürfen beispielsweise auch durch einen Benutzer auf einer Google-Karte händisch eingezeichnete Punkte/Koordinaten nicht frei verwendet oder verteilt werden, da sie auf Basis der Kartendaten ein abgeleitetes Werk darstellen und somit von der zugrunde liegenden Lizenz erfasst sind.

Rechtlich unproblematisch ist in den meisten Fällen hingegen die Möglichkeit, Daten aus verschiedenen Quellen lediglich auf dem Client zur Anzeige zusammenzuführen, zum Beispiel in Form einzelner Layer über einer Karte. Aber auch hier ist genau darauf zu achten, die etwa im Rahmen von Creative Commons geforderte Lizenz- und Namensnennung dann auch wirklich an geeigneter Stelle zu platzieren.

Stefan Neufeind und Tobias Hauser / ds

