MySQL 5.7: JSON und GIS

DOAG 2016, Nürnberg

Cédric Bruderer

MySQL Support Engineer, FromDual GmbH

cedric.bruderer@fromdual.com

Über FromDual GmbH

www.fromdual.com

Support

Beratung

remote-DBA

Schulung

Über mich

- Cédric Bruderer
- Ausbildung

  2010 – 2014: Lehre zum Informatiker

  Teilnahme an den Schweizer Berufsmeisterschaften
- Junior Engineer

  In einem international tätigen Unternehmen.
- MySQL Support Engineer bei FromDual

  seit Oktober 2015

Inhalt

- JSON
  - Eintragen und Auslesen
  - Generierte Spalten
  - Indexierung
- GIS
  - Spatial Index
  - Datentypen
  - GIS Funktionen
- GeoJSON
  - Real-Life Anwendungen

Was ist JSON?

- JavaScript Object Notation
- Kann mit XML verglichen werden
  - Ist aber leichter zu lesen
- Seit dem Jahr 2000, Standard seit 2013
- Unabhängig von einer Sprache
  - PHP, Perl, Python, C, …

```
{"employees":[
    {"firstName":"John", "lastName":"Doe"},
    {"firstName":"Anna", "lastName":"Smith"},
    {"firstName":"Peter", "lastName":"Jones"}
]}
```

JSON vs XML


```
{"employees":[
    {"firstName":"John", "lastName":"Doe"},
    {"firstName":"Anna", "lastName":"Smith"},
    {"firstName":"Peter", "lastName":"Jones"}
]}
```

```
<employees>
    <employee>
        <firstName>John</firstName> <lastName>Doe</lastName>
    </employee>
    <employee>
        <firstName>Anna</firstName> <lastName>Smith</lastName>
    </employee>
    <employee>
        <firstName>Peter</firstName> <lastName>Jones</lastName>
    </employee>
</employees>
```

http://www.w3schools.com/json/

JSON in MySQL

Wie bekomme ich das in die/aus der DB?

```
mysql> CREATE TABLE json_test  (jdoc JSON);

mysql> INSERT INTO json_test
       VALUES ('{"key1": "value1", "key2": "value2"}');

mysql> SELECT * FROM json_test;
+-----------------------------------+
| jdoc                              |
+-----------------------------------+
| {"key1": "value1", "key2": "value2"} |
+-----------------------------------+

mysql> SELECT JSON_EXTRACT(jdoc, "$.key1") FROM json_test;
+-----------------------------+
| JSON_EXTRACT(jdoc, "$.key1") |
+-----------------------------+
| "value1"                    |
+-----------------------------+
```

JSON in MySQL

Oder so...

```
mysql> SELECT address_details->"$.service_type" FROM address_list;
+-----------------------------------+
| address_details->"$.service_type" |
+-----------------------------------+
| "fire brigade"                    |
| "police station"                  |
+-----------------------------------+
```

JSON indexieren

- **Nur über generierte Spalten:**

```
mysql> CREATE TABLE address_list (
        id INT UNSIGNED NOT NULL AUTO_INCREMENT,
        jdoc JSON,
        lastname VARCHAR(50) AS (JSON_UNQUOTE(JSON_EXTRACT(jdoc, "$.lastname")),
        PRIMARY KEY (id),
        KEY lastname (lastname)
        ) ENGINE=InnoDB;
```

- **`JSON_UNQUOTE` entfernt Anführungszeichen**
- **`JSON_EXTRACT` liest den Wert aus dem Feld**

Dynamisches Schema

- Dank JSON, kein fixes Schema nötig
- Tabelle kann alles beinhalten

```
mysql> CREATE TABLE json_test (
       id INT UNSIGNED NOT NULL AUTO_INCREMENT,
       jdoc JSON
       )ENGINE=InnoDB;
```

- Informationen zu einem Objekt
  - Öffnungszeiten eines Geschäfts
  - Haltestellen einer Buslinie

GIS

- Geographic Information System
- Dient dem Speichern von Koordinaten
  - Längen- und Breitengrade
  - GPS
- Ermöglicht verschiedene „Berechnungen“ basierend auf Koordinaten
  - Distanzen
  - Anzahl Objekte in einem Ort

Spatial Indexes

- Indexieren von GIS
- Seit MySQL 5.7 auch InnoDB
  - GIS konnte schon vorher eingetragen werden, aber Index ging nur für MyISAM!

```
mysql> ALTER TABLE geo_data ADD SPATIAL INDEX(coordniates);
```

http://dev.mysql.com/doc/refman/5.7/en/creating-spatial-indexes.html

Datentypen

- `POINT(X, Y)`
  - X und Y, Längen- und Breitengrad
  - Adresse, Ort
- `LINESTRING(X Y, X Y)`
  - Linie, Strecke
  - Strasse, Fluss, ...
- `POLYGON(X Y, X Y, X Y, X Y)`
  - Form
  - Ortschaft, Grenzen, Gebiet, ...

ST oder MBR Funktionen

- MBR
  - Minimum Bounding Rectangle (Kleinstmögliches Rechteck)
  - Neigt zur Ungenauigkeit
- ST
  - Genauer wie MBR, weil korrekte form verwendet wird

GIS Funktionen

- **`ST_X(POINT)`, `ST_Y(POINT)`**
  - **Gibt X oder Y Koordinate eines Punkts zurück**

```
mysql> SELECT ST_X(POINT(56.7, 53.34));
+---------------------------+
| ST_X(POINT(56.7, 53.34)) |
+---------------------------+
|                      56.7 |
+---------------------------+


mysql> SELECT ST_Y(POINT(56.7, 53.34));
+---------------------------+
| ST_Y(POINT(56.7, 53.34)) |
+---------------------------+
|                     53.34 |
+---------------------------+
```

  - **`X(POINT)` und `Y(POINT)` sind seit MySQL 5.7.6 deprecated**

GIS Funktionen

- **ST_Touches(g1, g2)**
  - **Prüft ob sich zwei Geometrien berühren.**

```
mysql> SET @g1 = ST_GEOMFROMTEXT('POINT(2 0)');

mysql> SET @g2 = ST_GEOMFROMTEXT('LINESTRING(2 0, 0 2)');

mysql> SELECT ST_TOUCHES(@g1,@g2);
+---------------------+
| ST_TOUCHES(@g1,@g2) |
+---------------------+
|                   1 |
+---------------------+
```

GIS Funktionen

- `ST_Within(g1, g2)`
  - **Prüft ob eine Geometrie innerhalb einer anderen ist.**

```
mysql> SET @g1 = ST_GEOMFROMTEXT('POLYGON((174 149, 20 40, 50 60, 125 100, 174 149))');

mysql> SET @g2 = ST_GEOMFROMTEXT('POLYGON((175 150, 20 40, 50 60, 125 100, 175 150))');

mysql> SELECT ST_WITHIN(@g1,@g2);
+--------------------+
| ST_WITHIN(@g1,@g2) |
+--------------------+
|                  1 |
+--------------------+
```

  - `ST_Contains(g1, g2)` **prüft, ob eine Form komplett in einer anderen Form ist.**

GIS Funktionen

- **ST_Area(Polygon)**
  - **Berechnet die Fläche einer Form und gibt diese zurück.**

```
mysql> SET @poly = 'Polygon((0 0,0 3,3 0,0 0))';

mysql> SELECT ST_Area(ST_GeomFromText(@poly));
+----------------------------------+
| ST_Area(ST_GeomFromText(@poly)) |
+----------------------------------+
|                              4.5 |
+----------------------------------+
```

GIS Funktionen

- **ST_Intersection(g1, g2)**
  - **Gibt Schnittpunkte zurück.**

```
mysql> SET @g1 = ST_GeomFromText('LineString(1 1, 3 3)');

mysql> SET @g2 = ST_GeomFromText('LineString(1 3, 3 1)');

mysql> SELECT ST_AsText(ST_Intersection(@g1, @g2));
+-------------------------------------+
| ST_AsText(ST_Intersection(@g1, @g2)) |
+-------------------------------------+
| POINT(2 2)                          |
+-------------------------------------+
```

GIS Funktionen

- `ST_Intersects(g1, g2)`
  - **Prüft ob sich zwei Geometrien schneiden.**

```
mysql> SET @g1 = ST_GeomFromText('LineString(1 1, 3 3)');

mysql> SET @g2 = ST_GeomFromText('LineString(1 3, 3 1)');

mysql> SELECT ST_Intersects(@g1, @g2);
+--------------------------+
| ST_Intersects(@g1, @g2)  |
+--------------------------+
|                        1 |
+--------------------------+
```

GIS Funktionen

- `ST_Distance(g1, g2)`
  - **Berechnet die Distanz zwischen zwei Punkten und gibt diese zurück.**

```
mysql> SET @g1 = POINT(1,1);
mysql> SET @g2 = POINT(2,2);

mysql> SELECT ST_Distance(@g1, @g2);
+----------------------+
| ST_Distance(@g1, @g2) |
+----------------------+
|    1.4142135623730951 |
+----------------------+
```

ACHTUNG !

- **Vorsicht beim `SELECT`:**

```
root@localhost [test]> SELECT address_id, address_position FROM address_list;
+------------+--------------------------+
| address_id | address_position         |
+------------+--------------------------+
|          1 |               $@      ??   |
|          2 |               ??      "@   |
+------------+--------------------------+
2 rows in set (0.00 sec)
```

`POINT` Werte werden nur mit der richtigen Funktion (`ST_X`, `ST_Y`) Lesbar dargestellt!

- **Die Welt ist (noch) flach**
  - **Höhenangaben (z.B.: Meter über Meer) müssten „von Hand“ angegeben werden.**

GeoJSON

- Verbindung von JSON und GIS
  - GIS für die Position
  - JSON für die Information

GeoJSON

- **Beispiel:**
  - **Einer Position können Detail-Informationen zugewiesen werden.**

```
CREATE TABLE `geo_json` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `position` point NOT NULL,
  `jdoc` json DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB;
```

GeoJSON

- **Einträge hinzufügen:**

```
INSERT INTO geo_json
(
   position,
   jdoc
) VALUES (
   POINT(47.6951103,8.6385015),
   '{
      "type":"Bar",
      "name":"The Mule"
   }'
);
```

GeoJSON

- **Einträge auslesen:**

```
mysql> SELECT
    -> ST_X(position) as x,
    ->     ST_Y(position) as y,
    ->     JSON_UNQUOTE(JSON_EXTRACT(jdoc, "$.type")) as type,
    ->     JSON_UNQUOTE(JSON_EXTRACT(jdoc, "$.name")) as name
    -> FROM geo_json;
+------------+-----------+----------------+------------------------------+
| x          | y         | type           | name                         |
+------------+-----------+----------------+------------------------------+
| 47.6951103 | 8.6385015 | Bar            | The Mule                     |
|  47.699331 | 8.6322252 | Restaurant     | Restaurant M?hlental         |
| 47.6997696 |  8.638405 | Public Service | Feuerwehrzentrum Schaffhausen |
|  47.698033 | 8.6346874 | Restaurant     | Gasthaus Adler               |
|  47.698033 | 8.6346874 | Bank           | Schaffhauser Kantonalbank    |
+------------+-----------+----------------+------------------------------+
```

GeoJSON

- **Einträge editieren:**

```
mysql> UPDATE geo_json
    -> SET jdoc = '{"name": "Restaurant Muehlental", "type": "Restaurant"}'
    -> WHERE id=2;
```

```
mysql> SELECT
    -> ST_X(position) as x,
    ->     ST_Y(position) as y,
    ->     JSON_UNQUOTE(JSON_EXTRACT(jdoc, "$.type")) as type,
    ->     JSON_UNQUOTE(JSON_EXTRACT(jdoc, "$.name")) as name
    -> FROM geo_json;
+------------+-----------+----------------+-------------------------------+
| x          | y         | type           | name                          |
+------------+-----------+----------------+-------------------------------+
| 47.6951103 | 8.6385015 | Bar            | The Mule                      |
|  47.699331 | 8.6322252 | Restaurant     | Restaurant Muehlental         |
| 47.6997696 |  8.638405 | Public Service | Feuerwehrzentrum Schaffhausen |
|  47.698033 | 8.6346874 | Restaurant     | Gasthaus Adler                |
|  47.698033 | 8.6346874 | Bank           | Schaffhauser Kantonalbank     |
+------------+-----------+----------------+-------------------------------+
```

GeoJSON


```
UPDATE geo_json
SET jdoc = '
{
   "name": "Restaurant Muehlental",
   "type": "Restaurant",
   "open": "Mon: Closed, Tue - Sun: 11:30 - 14:00 / 17:30 - 23:30"
}'
WHERE id=2;
```

```
mysql> SELECT
    -> ST_X(position) as x,
    ->     ST_Y(position) as y,
    ->     JSON_UNQUOTE(JSON_EXTRACT(jdoc, "$.type")) as type,
    ->     JSON_UNQUOTE(JSON_EXTRACT(jdoc, "$.name")) as name,
    ->     JSON_UNQUOTE(JSON_EXTRACT(jdoc, "$.open")) as open
    -> FROM geo_json
    -> WHERE id=2\G
*************************** 1. row ***************************
   x: 47.699331
   y: 8.6322252
type: Restaurant
name: Restaurant Muehlental
open: Mon: Closed, Tue - Sun: 11:30 - 14:00 / 17:30 - 23:30
1 row in set (0.00 sec)
```

Real-Life Anwendungen

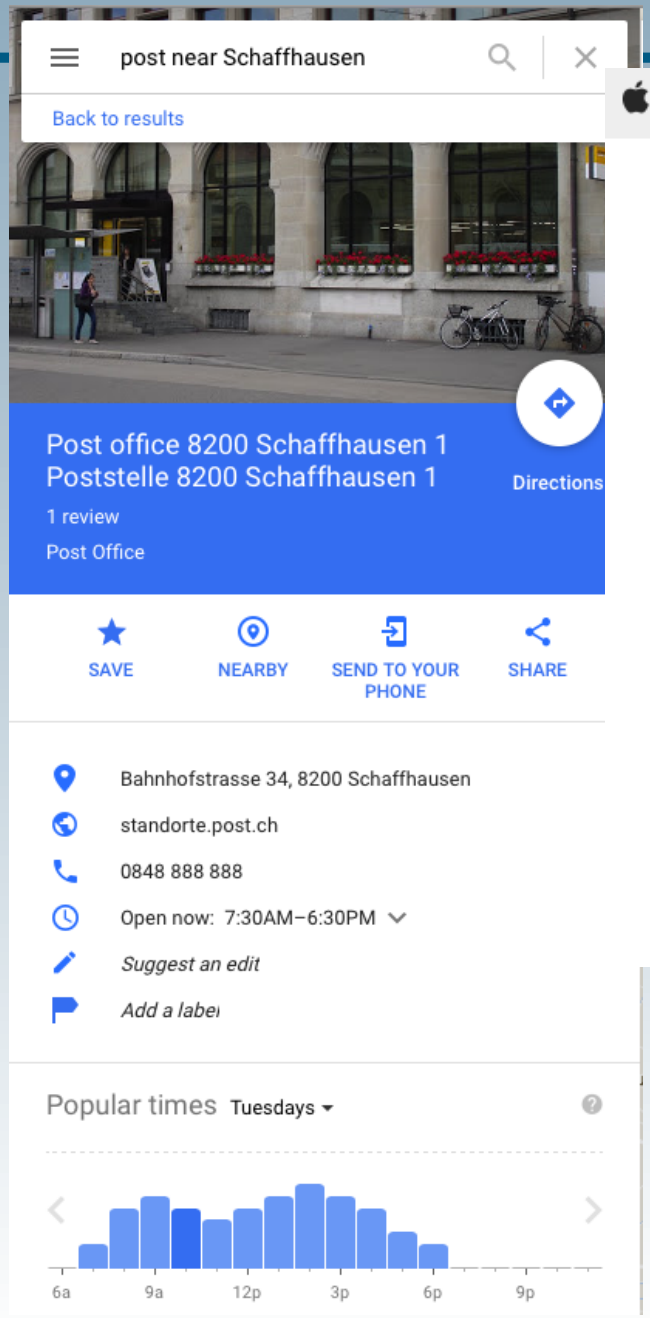
- **Stadtkarte**
  - **Ort wird über GIS identifiziert**
  - **Informationen zum Ort werden in JSON abgelegt**

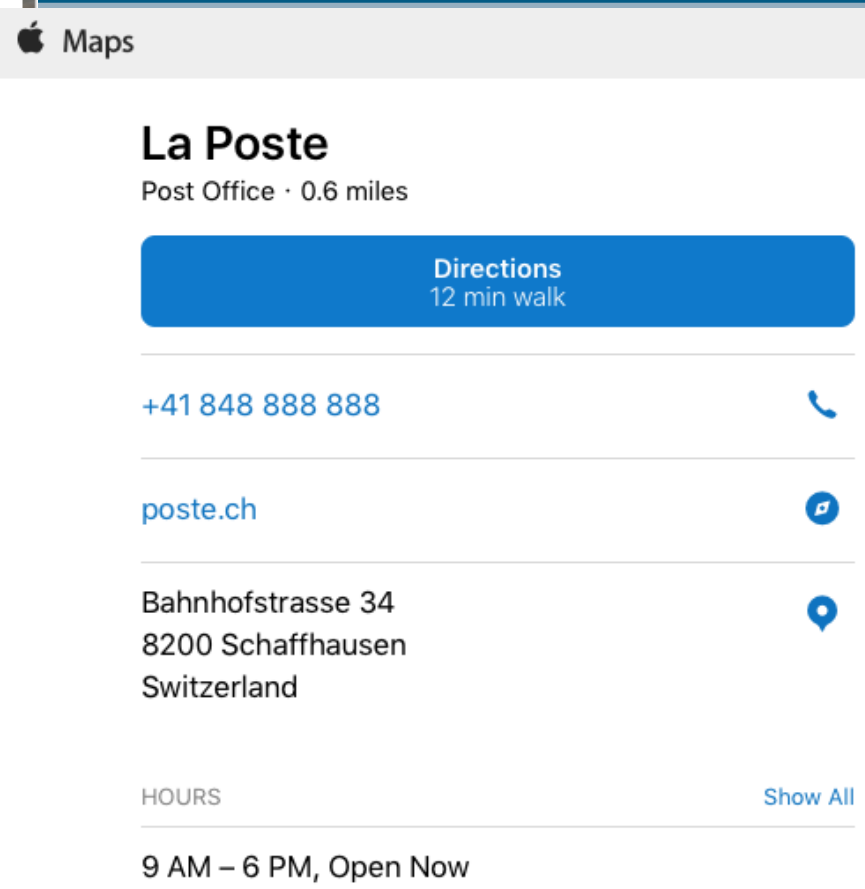
Feuerwehrzentrum Schaffhausen mit Informationen zum öffentlichen Verkehr, den Strassen und dem Gebäude.

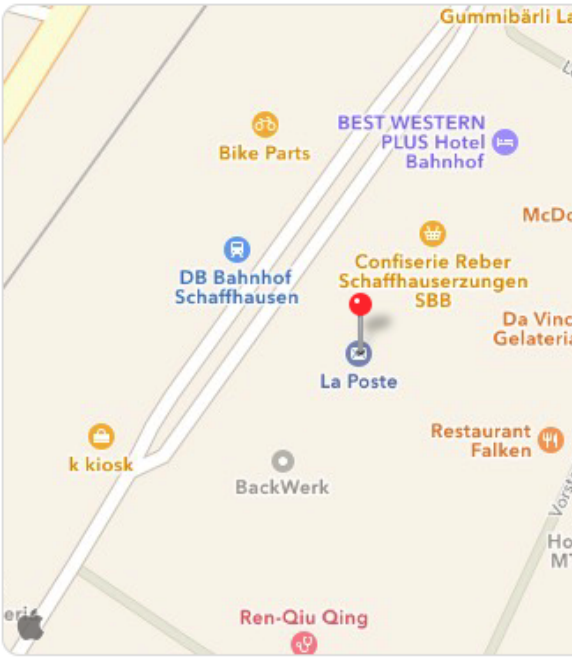
http://fwsh.ch/beta/index.php/organisation/feuerwehrzentrum
https://www.openstreetmap.org/#map=18/47.70111/8.63591

Real-Life Anwendungen

www.fromdual.com



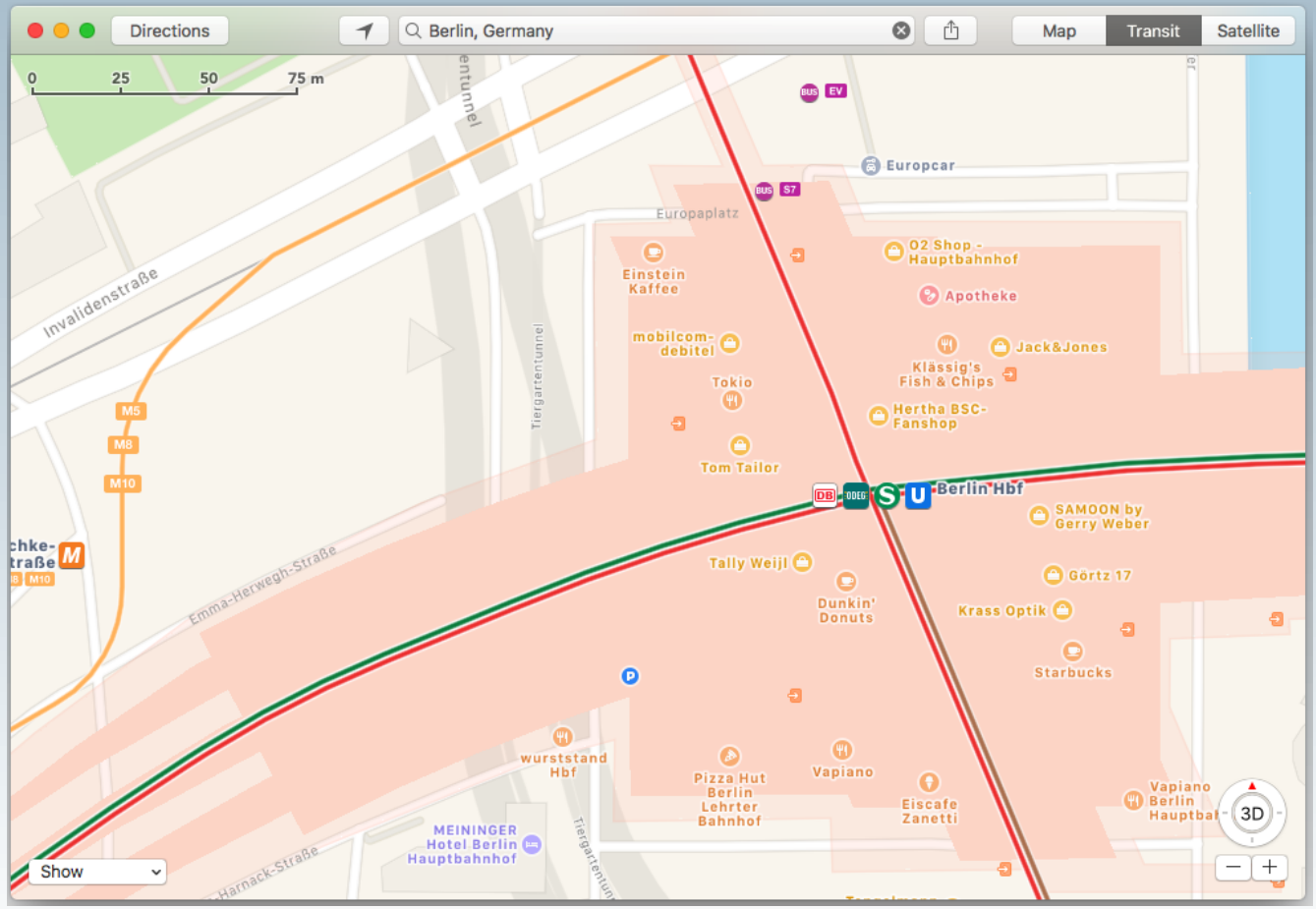




Karten mit Informationen zu Geschäften

Real-Life Anwendungen

- Wo ist der nächste Briefkasten?
- Wo ist die nächste Pizzeria?
- ÖPNV
  - Strecken
  - Haltestellen
  - Fahrzeiten?



Real-Life Anwendungen

- Lokalisierung
  - Apple: Find Friends
  - Facebook: Nearby Friends
  - Google: Timeline
- „Event-Tracking“
  - Einbrüche
  - Diebstähle
  - Wasserrohrbrüche
  - Feuer

Lesenswert

- MySQL JSON Funktionen
  https://dev.mysql.com/doc/refman/5.7/en/json-function-reference.html
- JSON Tuning
  https://www.percona.com/blog/2016/03/07/json-document-fast-lookup-with-mysql-5-7/
- MySQL GIS Funktionen
  http://dev.mysql.com/doc/refman/5.7/en/spatial-function-reference.html
- How To für GIS
  https://www.percona.com/blog/2016/02/03/new-gis-features-in-mysql-5-7/
  http://mysqlserverteam.com/mysql-5-7-and-gis-an-example/

Q & A

Fragen ?

Diskussion?

Wir haben Zeit für ein persönliches Gespräch...

- **FromDual bietet neutral und unabhängig:**
  - **Beratung**
  - **Remote-DBA**
  - **Support für MySQL, Galera, Percona Server und MariaDB**
  - **Schulung**

www.fromdual.com/presentations