System Architektur v0.0.1

24.09.16 - Andrei

Überblick

1. mongodb konfigurieren
   - Sensordaten simulieren
2. statische Datenbank Schema
   - OSM Daten abholen
3. Parken Abfragen

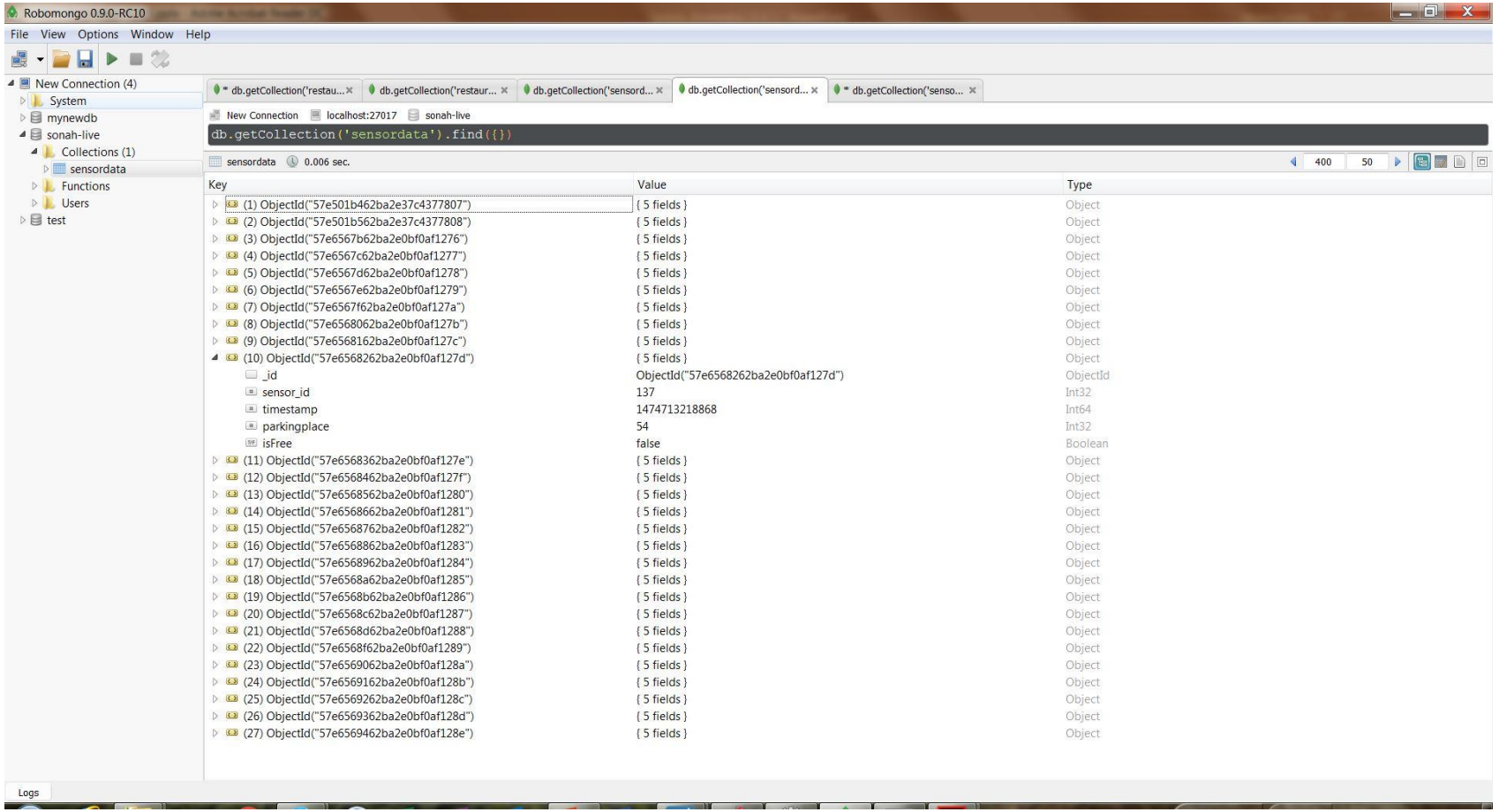
mongodb

- Darein werden Sensordaten abgespeichert:
  - Sensor id
  - Timestamp
  - Parkplatz
  - isFree – ist jetzt der Platz frei oder belegt? (TRUE oder FALSE)

Sensordaten simulieren

- Je nach Speicherart unterscheiden sich folgende Strategien:
  - Pro Event (aktuelle Wahl)
  - Pro Sekunde
    - Falls mehrfache Events pro Sekunde
    - Vorteil: weniger Inserts, spart Schreiben-Zeit
    - Zehntelsekunde-Events sind in Gruppen gespeichert: 0..9
  - Pro Minute
    - Sekunde-Events 0..59 sind zusammen gespeichert

Mongodb - Screenshot



Statische Datenbank – Schema (1)

```
CREATE TABLE `sensors` (
        `SensorID` bigint(20) NOT NULL AUTO_INCREMENT,
        `Latitude` double NOT NULL,
        `Longitude` double NOT NULL,
        `Altitude` double NOT NULL DEFAULT '0',
        `ParkingAreaID` bigint(20) COMMENT 'Parking area to which it belongs',
        `Added` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
        `Type` varchar(30) COMMENT 'Type of sensor (optical, etc.)',
        PRIMARY KEY (`SensorID`)
);
```

Statische Datenbank – Schema (2)

```
CREATE TABLE `parkingareas` (
        `ParkingAreaID` bigint(20) NOT NULL AUTO_INCREMENT,
        `ParkingAreaName` varchar(30),
        `Access` varchar(20) COMMENT 'E.g. public, customers, residents, employees',
        `Type` varchar(20) COMMENT 'E.g. surface, multi-storey, underground, rooftop, sheds,
garage_boxes',
        `FeeID` bigint(20) NOT NULL,
        /* time interval when open – missing, mysql has no such inbuilt type*/
        `MaxHeight` double,
        PRIMARY KEY (`ParkingAreaID`)
);
```

Statische Datenbank – Schema (3)

```
CREATE TABLE `parkingspace` (
        `ParkingSpaceID` bigint(20) NOT NULL AUTO_INCREMENT,
        `Latitude` double NOT NULL,
        `Longitude` double NOT NULL,
        `Altitude` double NOT NULL DEFAULT '0',
        `ParkingAreaID` bigint(20) NOT NULL COMMENT 'Parking area to which it belongs',
        `SensorID` bigint(20) COMMENT 'Sensor to which it belongs, can be NULL',
        PRIMARY KEY (`ParkingSpaceID`)
);
```

Statische Datenbank – Schema (4)

```
CREATE TABLE `Fees` (
        `FeeID` int(11) NOT NULL AUTO_INCREMENT,
        /* time interval -- missing, mysql has no such inbuilt type */
        `CostPerHour` double,
        /* other type of cost?! */
        PRIMARY KEY (`FeeID`)
);
```

OSM Daten

- Daten werden durch Abfragen an OverpassAPI geholt
  - Parkautomaten (vending = parking_tickets)
  - Parkräume (amentiy = parking)
- Aktuell
  - `sensors` Tabelle wird mit Parkautomaten-Daten befüllt
  - `parkingareas` Tabelle wird mit Parkräume-Daten befüllt
  - sensor (n) : parkingarea (1) -> in OSM schwierig festzulegen!!

Parken Abfragen

- Einfache Variante: Finde die besten Parkmöglichkeiten in der Nähe von Position (lat, lon), jetzt!
- Vorgeschlagener Algorithmus:
  - Suche im Umkreis 500m
    - finde die Parkräume, die den aktuellen Standort geometrisch enthalten
    - Berechne Anteil an freien Plätzen für diese Parkräume (stand: aktuell)
    - Erstelle eine Rangliste; E.g. P1: 35% frei, P2: 24% frei, P3: 10% frei
    - Falls nicht genug Parkräume gefunden/die Raten zu niedrig sind
  - Suche im Umkreis 1000m
    - das gleiche
  - ...