# Studie: Das Datenmodell von OpenStreetMap

OSM-User: Kumakyoo

19. März 2024

**Zusammenfassung:** Diese Studie untersucht das Datenmodell von OpenStreetMap anhand der derzeit gesammelten Daten zu Deutschland. Die Datentypen, deren Attribute und Tags, sowie die am häufigsten gesammelten Elemente werden dabei analysiert.

Bei der Analyse wurde der Schwerpunkt auf Schwachstellen gelegt. Diese werden jeweils beschrieben und anhand von Beispielen aus dem aktuellen Datensatz illustriert.

Aus dieser Untersuchung ergeben sich fast zwangsläufig einige Verbesserungsideen. Die fünf wichtigsten wurden am Ende der Studie zusammengestellt.

## Inhaltsverzeichnis

# 1 Datenmodell

## 1.1 Datentypen

Das Ziel von OpenStreetMap (OSM) ist es, die reale Welt in einem Datenmodell zu repräsentieren. Dabei wird eine gewisse Auswahl von realen Objekten auf die 2-dimensionale Erdoberfläche projiziert und auf das Wesentliche abstrahiert. Diese Abstraktion wird in der OSM-Datenbank gespeichert.

Hierfür stehen drei Datentypen zur Verfügung: Punkte (`node`), Wege[1] (`way`) und Relationen (`relation`), die unterschiedlich eingesetzt werden, siehe Abbildung 1: `node` wird benutzt, um punktförmige Objekte darzustellen, beispielsweise eine Straßenlampe. Zudem wird `node` in einigen Fällen auch für Adressdaten genutzt.

Mit dem Datentyp `way` werden sowohl linienhafte Objekte, beispielsweise ein Zaun, als auch Flächen ohne Löcher, beispielsweise eine Wiese, modelliert.

Flächen mit Löchern, etwa ein See mit einer Insel, oder Flächen, die aus mehreren unzusammenhängenden Teilen bestehen, werden in einer Multipolygon-Relation gespeichert. Relationen werden zudem auch für (mehr oder minder abstrakte) Zusammenhänge zwischen den Daten genutzt, beispielsweise für den Verlauf einer Buslinie.

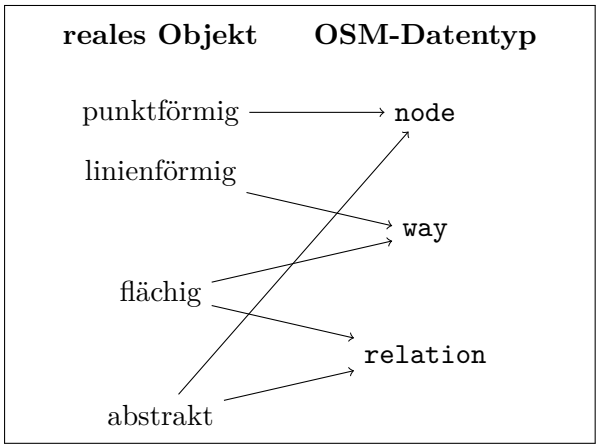


Abbildung 1: Wie reale Objekte mit OpenStreetMap-Datentypen modelliert werden.

Bereits auf dieser Ebene sind Probleme erkennbar:

- Die OSM-Datentypen werden für reale Objekte mit unterschiedlichen Eigenschaften genutzt. Insbesondere beim Datentyp `way` erschwert das Rückschlüsse auf die realen Objekte. So ist es schon vorgekommen, dass Flächen in einer Karte nur durch ihren Umriss dargestellt wurden, da der Algorithmus, der die Objekte verarbeiten musste, die Fläche für einen geschlossenen Weg gehalten hatte.[2]

- Flächige Objekte und abstrakte Dinge werden auf zwei unterschiedliche Arten gespeichert. Das erschwert die algorithmische Behandlung dieser Objekte und kann im schlimmsten Fall sogar dazu führen, dass Objekte mit ansonsten identischen Eigenschaften ganz unterschiedlich behandelt werden. Etwa: Ein See ohne Insel wird in der Karte eingezeichnet, ein See mit Insel nicht.

## 1.2 Attribute

Jedem der drei Datentypen werden zusätzliche Informationen beigefügt. Neben einer eindeutigen Nummer[3] (ID) zum Identifizieren des Objekts (und allerhand Metadaten, die uns hier nicht interessieren) gibt es für alle drei Datentypen eine beliebige Menge an Schlüssel-Wert-Paaren (Tags genannt).

Jeder Datentyp hat zudem weitere Attribute, die nur zu diesem Datentyp gespeichert werden:

**Punkt:** Geographische Breite und Länge des Punkts im WGS84-Koordinatensystem. Beispielsweise befindet sich laut OSM-Datenbank an 50,79888° nördlicher Breite und 8,78398° östlicher Länge ein Briefkasten (ID 2583954852).[4]

[1] Korrekter: Streckenzüge.

[2] Siehe zum Beispiel `https://github.com/gravitystorm/openstreetmap-carto/issues/3453`

[3] Die IDs werden für jeden Datentyp einzeln hochgezählt, was bedeutet, dass eine ID erst zusammen mit dem Datentyp ein Element eindeutig referenziert.

[4] Alle Angaben beziehen sich auf den Datenbank-Stand vom 1.1.2024. In der aktuellen Datenbank können diese bereits wieder geändert worden sein.

**Weg:** Eine Liste von Punkten, die einen Streckenzug definieren. Beispielsweise findet man unter der (Weg-)ID 156258695 einen Pfad, der die beiden Punkte mit den (Punkt-)IDs 1667905641 und 360857041 verbindet.

**Relation:** Eine Liste aus Elementen aller drei Datentypen (als Mitglieder bezeichnet), wobei jedem Element der Liste eine Rolle zugeordnet ist. Beispielsweise beschreibt die Relation mit der ID 9591937 einen See mit Insel. Die Mitglieder dieser Relation sind die beiden Wege mit den IDs 690275530 und 118857819, wobei ersterer die Rolle `outer`, also die Außenbegrenzung des Sees und zweiterer die Rolle `inner`, also die Grenze zur Insel, zugewiesen ist.

Wege bauen auf dem Datentyp Punkt auf. Hierfür können Punkte ohne Tags, aber auch Punkte mit Tags genutzt werden. Insbesondere ist es möglich, dass die Punkte, die einen Weg definieren, zusätzlich noch ein anderes reales Objekt abbilden. Beispielsweise kann ein Punkt, der für die Definition einer Straße benutzt wird, gleichzeitig ein Verkehrszeichen darstellen.

Diese Doppelnutzung birgt das Risiko, dass die Punkt-Objekte bei Korrekturen am Weg-Objekt versehentlich an eine falsche Stelle geschoben oder gar gelöscht werden. Beides dürfte in den veränderten Datensätzen kaum auffallen; solche Fehler werden typischerweise erst dann bemerkt (oder unbemerkt korrigiert), wenn jemand die Situation vor Ort erneut überprüft.

Relationen gehen noch einen Schritt weiter und bauen gleich auf allen drei Datentypen auf. Auch hier können die Elemente der Relation andere Objekte darstellen. Teilweise, wie beispielsweise bei Buslinien, ist das erwünscht, teilweise, wie beispielsweise bei Multipolygonen, birgt es das selbe Risiko, wie bei den Wegen.

Ob erwünscht oder nicht, die Doppelnutzung erschwert das Editieren von Weg-Elementen, die Teil einer Relation sind. Selbst erfahrene Mapper scheuen gelegentlich vor Veränderungen oder Korrekturen solcher Elemente zurück, aus Angst, bei der Relation etwas kaputt zu machen oder weil der Aufwand der Korrektur durch das anschließende Reparieren zahlreicher Relationen stark ansteigt.

Hinzu kommt noch, dass Relationen direkt oder indirekt rekursiv sein, sich also selbst als Mitglied enthalten können. Derzeit gibt es in Deutschland 6 rekursive Relationen: Die Relationen mit den IDs 1492100, 1697974, 2326300 und 11148575 enthalten sich jeweils selbst. Die Relationen mit den IDs 1212338 und 3067061 enthalten sich gegenseitig. Gleiches gilt für die Relationen mit den IDs 14439213 und 14462251. Vermutlich sind es allesamt Fehler.

Da es bislang keine Konventionen für rekursive Relationen gibt, kann man davon ausgehen, dass ein Teil der Anwendungssoftware für OSM-Daten mit Rekursionen nicht (korrekt) umgehen kann. Endlosschleifen und fehlerhafte Ergebnisse sind zu erwarten.

## 1.3 Tags

Bei den Tags handelt es sich um Schlüssel-Wert-Paare[5], die jedem der drei Datentypen in beliebiger Menge zugeordnet werden können, wobei allerdings jeder Schlüssel nur einmal vorkommen kann. Sowohl der Schlüssel, als auch der Wert eines Tags ist eine beliebige, maximal 255 Zeichen lange Zeichenkette.

### 1.3.1 Schlüssel

Es gibt keine Vorgaben, welche Schlüssel benutzt werden dürfen, jeder Mapper kann einfach neue Schlüssel erfinden. Insgesamt werden derzeit in Deutschland 28.772 verschiedene Schlüssel benutzt.[6] 9.385 davon nur ein einziges Mal, beispielsweise `cheese` mit dem Wert `yes` beim Punkt mit der ID 6140874678. Dieser Punkt modelliert einen Fachhandel für alkoholische Getränke. Der Tag soll wohl angeben, dass das Geschäft auch Käse verkauft. Aber auch Tippfehler, wie `commerical` (i und c vertauscht) beim Weg 151346662, kommen vor.

[5] Im Folgenden verwenden wir die übliche Konvention, Schlüssel-Wert-Paare in der Form *Schlüssel*=*Wert* zu schreiben. Man beachte dabei aber, dass es auch Schlüssel und Werte geben kann, die ein Gleichheitszeichen enthalten. Bei den Schlüsseln kommt dies in der Relation 8131479 gehäuft vor. Diese definiert die Standardwerte der StVO. Bei den Werten ist das vor allem, aber nicht nur, bei Webadressen der Fall.

[6] Ohne die in den nächsten Abschnitten erleuterten Subschlüssel und Schlüssel mit Life-Cycle-Präfixen kommt man auf 5.504 verschiedene Schlüssel.

Allerdings haben sich im Laufe der Zeit in der OSM-Gemeinschaft zahlreiche Schlüssel etabliert und für jeden Schlüssel gibt es Vorgaben, wie die zugehörigen Werte aussehen sollten. Beispielsweise gibt `maxspeed=20 mph` bei einem Element an, dass dem Element eine Maximalgeschwindigkeit von 20 Meilen pro Stunde zugeordnet ist. Abweichungen von diesen Konventionen werden oft als Fehler angesehen.

Gemäß dieser Konventionen haben Tags dreierlei Funktion: Zuerst einmal wird damit beschrieben, um was für ein Objekt es sich bei dem Element handelt (`highway=residantial` beschreibt eine Wohnstraße). Dann werden damit dem Element auch Eigenschaften des Objekts hinzugefügt, beispielsweise der Name der Straße. Zuletzt kommen noch Meta-Informationen hinzu, wie beispielsweise die Informationsquelle.

Da nie klar definiert wurde, welche Schlüssel der Definition eines Elements dienen, ist es oft nicht möglich, aus den Daten zu ermitteln, um was für ein reales Objekt es sich handelt. Insbesondere kann es passieren, dass man ein Element überhaupt keinem Objekt zuordnen kann und es kann auch passieren, dass man ein Element mehreren Objekten zuordnen muss.

Ersteres findet sich beispielsweise beim Punkt mit der ID 75853940: Dort ist der Name „Eiserner Ranga“ hinterlegt, aber ohne weitere Informationen, was das sein soll.

Ein Beispiel für ein Element, welches mehreren Objekten zugeordnet ist, findet man beim Punkt mit der ID 185542943: Dieser bezeichnet sowohl ein Hotel als auch ein Restaurant. Es ist nicht so ungewöhnlich, dass sich Hotel und Restaurant in einem Haus befinden. Nur: Gelten die angegebenen Öffnungszeiten (täglich 10–23 Uhr) für das Restaurant oder das Hotel oder beides?

Es gibt auch Fälle, in denen Schlüssel mal als Objektdefinition und mal als Eigenschaft eines Objekts dienen: `highway=crossing` identifiziert beispielsweise eine Querungsstelle einer Straße (etwa einen Zebrastreifen). Mit `crossing=unmarked` kann dann die Querungsstelle weiter beschrieben werden. In diesem Fall bedeutet `unmarked`, dass keine Markierungen auf der Straße vorhanden sind, die Querungsstelle als solche aber schon offiziell als Querungsstelle existiert.

Aber dann gibt es auch noch Querungsstellen, die nicht offiziell vorgesehen sind, an denen aber dennoch oft gequert wird. Diese erhalten den zusätzlichen Eintrag `informal=yes`. Soweit wäre das kein Problem, aber die Konventionen sehen vor, dass in diesem Fall, da die Querungsstelle nicht offiziell ist, der Tag `highway=crossing` nicht gesetzt werden darf. Am Ende hat man ein Objekt, bei dem die Objekt-Identifikation fehlt. Diese Funktion muss durch zwei andere Tags (`crossing` und `informal`), die eigentlich beschreibenden Charakter haben, hilfsweise übernommen werden.

### 1.3.2 Subschlüssel

Eine OSM-Konvention sieht vor, dass Schlüssel durch Doppelpunkte strukturiert werden können. Wir bezeichnen solche Schlüssel als Subschlüssel.[7]

Die Funktion dieser Subschlüssel wurde nie genau geklärt und Subschlüssel werden demnach auch nicht konsequent für bestimmte Dinge eingesetzt. Insbesondere werden die Schlüsselteile mal als Präfix und mal als Suffix verwendet. Ohne Anspruch auf Vollständigkeit konnten in den Daten die folgenden Nutzungsarten gefunden werden:

**Ausnahmen:** Neben einem Schlüssel wird zusätzlich noch ein Subschlüssel zu diesem Schlüssel benutzt, um Ausnahmen zu notieren. So bedeutet die Angabe `oneway=yes` bei einer Straße, dass in dieser Straße nur in einer Richtung gefahren werden darf (Einbahnstraße). `oneway:bicycle=no` gibt dann zusätzlich noch an, dass diese Beschränkung für Fahrräder nicht gilt, diese also in beide Richtungen fahren dürfen.

Es kommt allerdings auch oft vor, dass der Originalschlüssel (hier `oneway`) nicht mit angegeben wird, da für diesen ein Standardwert existiert. Beispielsweise findet sich recht häufig `oneway:bicycle=`

[7] Da die Nutzung dieser Strukturierung sehr unterschiedlich ausfällt, ist es nicht einfach, hierfür einen geeigneten Namen zu finden. Die deutsche Wiki-Seite (`https://wiki.openstreetmap.org/wiki/DE:Attribut`) spricht von „Namensräumen“, die englische (`https://wiki.openstreetmap.org/wiki/Tags`) zusätzlich von Super- und Subkategorien. Nichts davon trifft auf alle Nutzungsarten zu.

`yes` bei straßenbegleitenden gemischten Fuß- und Radwegen, da hier Fußgänger in beide Richtungen gehen dürfen, Radfahrer aber (ohne gesonderte Ausschilderung) nur in der Richtung fahren dürfen, in der die Fahrzeuge auf der benachbarten Fahrbahn fahren.

Problematisch ist hierbei, dass Software, die die Tags auswertet, diese zusätzlichen Schlüssel möglicherweise nicht berücksichtigt[8] und dadurch Fehler macht.

**Zuordnung:** Mit `emergency=fire_hydrant` wird ein Hydrant gemappt. Ein Teil der Eigenschaften des Hydranten werden aber nicht als eigenständige Tags (etwa `diameter=150`) sondern mit Subschlüsseln zum Schlüssel `fire_hydrant` angegeben (also `fire_hydrant:diameter=150`). Der eigentliche Schlüssel `fire_hydrant` existiert nicht und wird auch nicht benutzt. Wohl aber gibt es weitere Tags, die sich auf den Hydranten beziehen, die ohne Subschlüssel angegeben werden: `ref` steht für die Referenznummer des Hydranten.

Diese Zusätze blähen die Datenbank unnötig auf. Notwendig werden sie nur, weil es manchmal vorkommt, dass zwei unterschiedliche Objekte durch das selbe Element gemappt werden. In einigen Fällen kommt es sogar vor, dass diese Zuordnungs-Zusätze nur dann genutzt werden, wenn es für eine Unterscheidung notwendig ist und sonst nicht, etwa bei `opening_hours:hotel` im Punkt 29262146.

Für Programme besteht hierbei das Problem, dass möglicherweise Schlüssel übersehen werden, weil diese auf unterschiedliche Weise angegeben werden können.

**Kennzeichnung der Zusammengehörigkeit:** Adressen werden in OpenStreetMap in ihre Bestandteile aufgeteilt und diese einzeln gespeichert. Die Adresse „Hölsterloh 1 in 59929 Brilon“ wird in OSM als `addr:city=Brilon`, `addr:country=DE`, `addr:housenumber=1`, `addr:postcode=59929` und `addr:street=Hölsterloh` gemappt, siehe Node mit der ID 23656400. Die Subschlüssel dienen hier ausschließlich dem Zweck, diese zusammengehörenden Daten als zusammengehörend zu markieren.[9]

**Zusatzinformationen:** Mit `maxspeed` wird eine Maximalgeschwindigkeit gespeichert. Mit `maxspeed:type` wird angegeben, auf welche Art der Staat diese Maximalgeschwindigkeit angeordnet hat, eine zusätzliche Information zur eigentlichen Angabe.

Solche Zusatzinformationen werden insbesondere auch dann notwendig, wenn Fuß- und Radwege, sowie Querungsstellen nicht eigenständig gemappt werden, sondern nur als Tags einer Straße. Dadurch müssen alle Eigenschaften dieser Wege durch zusätzliche Tags angegeben werden, was die algorithmische Auswertung der Tags erschwert. In einigen Fällen ist es sogar gar nicht mehr möglich, Eigenschaften sinnvoll anzugeben.

**Abgeleiteter Schlüssel:** Manche Objekte, wie beispielsweise Straßen, werden in OSM als Weg und nicht als Fläche gespeichert, da dies für zahlreiche Anwendungen praktischer ist. Andere Anwendungen benötigen aber die genaue Fläche. Deswegen kann diese zusätzlich gemappt werden und der Schlüssel, der für das ursprüngliche Objekt genutzt wurde, erhält dann den Präfix `area`. Am häufigsten ist dies derzeit bei `area:highway` anzutreffen.

**Genauere Spezifizierung des Schlüssels:** Manchmal lässt ein Schlüssel Spielraum bei der Interpretation, so kann sich `cycleway` bei straßenbegleitenden Radwegen auf den Radweg links oder rechts beziehen. Und Objekte können mehrere Referenznummern `ref` haben. Bei ersteren kann man dann mit `cycleway:left` klarstellen, auf welchen Radweg man sich bezieht, bei zweiterem würde `ref:IFOPT` sich auf die Internationale Haltestellen-Nummer beziehen.

[8] Da es sehr viele Schlüssel und Subschlüssel gibt und letztere auch noch ganz unterschiedliche Nutzungsarten aufweisen, ist es algorithmisch sehr schwierig, für jeden Schlüssel zu bestimmen, was dieser bedeuten soll. Selbst Menschen haben ja oft Probleme damit.

[9] Diese Nutzung entspricht am ehesten der Bezeichnung „Namensraum“.

Hinzu kommt, dass es mindestens einen Subschlüssel gibt, der problematisch ist: `building:part`. Der `building`-Präfix gehört normalerweise in die Zuordnungs-Kategorie, damit werden Eigenschaften eines Gebäudes (Farbe, Anzahl der Stockwerke etc.) genauer spezifiziert. Im speziellen Fall von `building:part` hingegen wird nicht eine Eigenschaft des Gebäudes beschrieben, sondern ein Objekt als Teil eines Gebäudes identifiziert. `part_of_building` oder `building_part` als eigenständiger Schlüssel wäre hier meiner Ansicht nach angebrachter gewesen.

Ein weiteres Problem stellen Subsubschlüssel, Subsubsubschlüssel und so weiter dar. Zum einen gibt es hierbei bunte Mischungen aus den obigen Nutzungsarten, zum anderen ist oft die Reihenfolge der Schlüsselteile vertauschbar: `lanes:bus:backward` oder `bus:lanes:backward`? Ersteres kommt 238 mal in der Datenbank vor, zweiteres 1279 mal. Verwirrend ist auch `cycleway:left:separation:right`, was die rechte Abgrenzung des Radwegs, der links der Straße entlang führt, angibt.

Zudem steigt die Anzahl der Möglichkeiten mit jedem weiteren Sub-Level exponentiell an. Das Extrem, das man in Deutschland findet, ist `fuel:gas:bottled:de:11kg:red:Tyczka_Totalgaz=yes`.[10] Damit wird eine bestimmte Gasflaschenart bezeichnet. Es ist fraglich, ob derartige Angaben noch sinnvoll sind. Die Auswertung dürfte nur noch in Spezialfällen sinnvoll möglich sein, da für allgemeine Algorithmen auf jeder Stufe zahlreiche Alternativen bekannt sein müssten.

Noch ein Beispiel: Eine Straße, bei der normalerweise Tempo 50 gilt, nachts aber Tempo 30, wird mit `maxspeed=50` und `maxspeed:conditional=30 @ (22:00-06:00)` getaggt. Wie schon gesehen ist es durchaus möglich, dass Software diesen zweiten Tag nicht auswertet und dadurch falsche Maximalgeschwindigkeiten für die Nachtstunden ermittelt.

Es geht aber noch komplizierter: Es ist nämlich nicht immer so, dass auf beiden Straßenseiten die selben Geschwindigkeitsbegrenzungen gelten. So könnte es sein, dass auf einer Seite zudem noch vorgeschrieben ist, dass bei Nässe nur 10 km/h gefahren werden darf. Das würde dann mit `maxspeed=50`, `maxspeed:forward:conditional=30 @ (22:00-06:00)` und `maxspeed:backward:conditional=30 @ (22:00-06:00); 10 @ wet` angegeben. Hier wird ein Schlüssel durch zwei Subsubschlüssel genauer spezifiziert, ein verbindender Subschlüssel fehlt.

### 1.3.3 Life-Cycle-Präfixe

Ebenfalls durch einen Doppelpunkt getrennt werden sogenannte Life-Cycle-Präfixe. Damit können Objekte, die nicht mehr (wurde abgerissen) oder noch nicht (ist geplant) existieren, in die Datenbank eingetragen werden. Ein Beispiel wäre `demolished:amenity=bench`, was bedeutet, dass sich an der Stelle Überreste einer Sitzbank befinden, die derzeit aber nicht benutzt werden kann.

Problem dabei: Bei einem durch Doppelpunkt getrennten Schlüssel kann nicht ohne weiteres entschieden werden, ob es sich um einen Schlüssel mit Life-Cycle-Präfix handelt oder um einen Subschlüssel. Theoretisch könnte man dies mit einer Liste der Life-Cycle-Präfixe abgleichen. Diese Liste ist aber, wie so vieles bei OSM, nicht ganz klar definiert.

### 1.3.4 Werte

Zu jedem Schlüssel sind bestimmte Werte vorgesehen. Dabei kann es sich beispielsweise um ein schlichtes `yes` oder `no` handeln oder eine Zahl oder ein kompliziertes Schema.

Das wohl komplizierteste Schema sind die `conditional`-Ausdrücke, die wir oben schon gesehen haben. Hierbei wird durch `@` getrennt ein Wert angegeben, der nur bei einer bestimmten Rahmen-Bedingung gilt. Diese Bedingung wiederum kann, insbesondere wenn es sich um eine zeitliche Beschränkung handelt, selbst sehr komplex werden. Auch hier ist es nicht ganz einfach, solche Ausdrücke algorithmisch korrekt auszuwerten und es kann auch passieren, dass Teile der Bedingung vom Programm überhaupt nicht verarbeitet werden können. Zudem fällt es auch vielen Mappern schwer, diese Ausdrücke korrekt einzugeben.

Weiterhin kann es vorkommen, dass einem Schlüssel mehrere Werte zugeordnet werden.

[10] Man versuche mal, dieses Ungetüm in die weiter oben in diesem Abschnitt aufgeführten Nutzungsarten aufzuspalten. Mir ist das nicht gelungen.

Dies geschieht, indem man die Werte durch ein Semikolon trennt. Beispielsweise `kerb=flush;lowered` für geteilte Bordsteine bei Querungsstellen.[11] Mehrere Werte sind vor allem deswegen problematisch, weil Software oft nur einen Wert erwartet und mit diesen Mehrfachwerten nicht umgehen kann. Außerdem kann es auch hier passieren, dass die Reihenfolge unterschiedlich ist: Der obige Tag könnte genausogut auch als `kerb=lowered;flush` angegeben werden (kommt 50 mal in der Datenbank vor, die andere Variante 238 mal).

Hinzu kommt, dass gelegentlich die Werte der Schlüssel HTML-Entitäten (etwa `&`) enthalten, die durch ein Semikolon abgeschlossen werden. Diese Semikolons gelten dann nicht als Werte-Trenner.

Weiter kann es auch zu semantischen Ungenauigkeiten führen wie im Geschwindigkeits-Beispiel oben: `maxspeed:backward:conditional=30 @ (22:00-06:00); 10 @ wet`: Wenn es nachts nass ist, darf man dann 10 km/h oder 30 km/h fahren? Natürlich 10 km/h. Uns Menschen ist das klar. Aber kann auch ein Computerprogramm damit richtig umgehen?

## 1.4 Die wichtigsten Elemente

### 1.4.1 Bei Relationen

Es gibt derzeit 804.645 Relationen. 804.314 davon, also fast alle, haben als Objekt-Identifikation den `type`-Tag, der angibt, um was für eine Relation es sich handelt. Von den verbleibenden 331 Relationen haben 40 einen Life-Cycle-Präfix vor dem `type`-Tag. Bei den anderen 291 handelt es sich sehr wahrscheinlich um fehlerhafte Einträge, da der `type`-Tag bei Relationen vorgeschrieben[12] ist.

Fünf dieser Relationen haben überhaupt keine Tags.[13]

Die häufigsten Relationen-Typen sind in Tabelle 1 aufgeführt.

| **Anzahl** | % | `type` |
|---|---|---|
| 244.580 | 30,4 | `multipolygon` |
| 164.480 | 20,5 | `route` |
| 138.278 | 17,2 | `restriction` |
| 105.200 | 13,1 | `public_transport` |
| 41.346 | 5,1 | `boundary` |
| 35.813 | 4,5 | `destination_sign` |
| 13.629 | 1,7 | `route_master` |
| 11.427 | 1,4 | `building` |
| 11.318 | 1,4 | `waterway` |
| 7.981 | 1,0 | `site` |
| 6.467 | 0,8 | `TMC` |
| 5.204 | 0,7 | `enforcement` |
| 4.245 | 0,5 | `turnlanes:turns` |
| 2.754 | 0,3 | `street` |
| 1.649 | 0,2 | `bridge` |
| 1.437 | 0,2 | `tmc` |
| 1.282 | 0,2 | `tmc:point` |
| 993 | 0,1 | `network` |
| 711 | 0,1 | `superroute` |
| 654 | 0,1 | `collection` |
| 561 | 0,1 | `tmc:link` |
| 513 | 0,1 | `junction` |
| 464 | 0,1 | `turnlanes:lengths` |
| | | ... |
| 291 | 0,0 | *Ohne Typ* |

Tabelle 1: Die häufigsten `type`-Werte von Relationen

Der am häufigsten vorkommende Typ `multipolygon` nimmt dabei eine besondere Stellung ein, da damit Flächen, die sich nicht mit einem einzigen Weg darstellen lassen, gemappt werden. Wir werden uns diese wegen der besonderen Stellung gleich noch gesondert ansehen.

Man kann der Tabelle 1 auch entnehmen, dass nur eine kleine Anzahl an Relationen-Typen in der Datenbank in nennenswerter Anzahl vorkommt. In einigen Fällen mag das daran liegen, dass eine derartige Relation in der Realität

[11] Bei einem Teil des Bordsteins wird dieser auf Straßenniveau abgesenkt (`kerb=flush`). Das erleichtert allen Verkehrsteilnehmern mit Rädern das Leben: Leute mit schweren Rollkoffern genauso, wie Leute mit Rollatoren oder Rollstuhlfahrern. Der andere Teil des Bordsteins hingegen ist nur auf ca. 3 cm abgesenkt (`kerb=lowered`). Das ist vor allem für Blinde wichtig, da sie so die genaue Position des Bordsteins gut erfühlen können.

[12] 100%ige Vorschriften gibt es bei OpenStreetMap nicht. Es wird dort jedem Mapper zugebilligt, eigene Ideen umzusetzen, auch wenn diese nicht etabliert sind. Allerdings haben sich im Laufe der Zeit mehr oder minder starke Konventionen eingebürgert. Viele davon haben auch einen formalen Abstimmungsprozess durchlaufen. Der `type`-Tag bei Relationen dürfte (obwohl ohne formalen Abstimmungsprozess) eine der stärksten Konventionen bei OpenStreetMap sein.

[13] IDs 1501386, 16306903, 16558835, 16848666 und 16908684.

nicht oft vorkommt. In den meisten Fällen dürfte aber die bessere Erklärung sein, dass Relationen bei vielen Mappern recht unbeliebt sind[14] und sich deswegen nur einige wenige wichtige Relationen halten konnten.

Auffällig sind auch die TMC-Einträge[15]: Hier herrscht ein ziemliches Chaos. Der Tag wird oft mit Großbuchstaben geschrieben, manchmal aber auch mit Kleinbuchstaben. Manchmal wird er gar nicht vergeben und stattdessen identifizieren Subschlüssel den Typ.

### 1.4.2 Bei Multipolygonen

Es gibt keinen eindeutigen Tag, der den Typ eines Multipolygons bestimmt. Stattdessen übernehmen zahlreiche Tags diese Rolle gemeinsam, ohne das dies hinreichend dokumentiert ist.

Ersatzweise wurde hier jeweils der am häufigsten vorkommende Tag in der Datenbank, der nicht offensichtlich andere Bedeutung hat, als Typ-Bestimmung herangezogen und dies so lange mit den verbleibenden Multipolygonen wiederholt, bis weitere Typen sich nicht mehr klar benennen ließen. Das Ergebnis ist in Tabelle 2 zu sehen.

Die drei häufigsten Typen sind `landuse`, `natural` und `building`. Mit den ersten beiden werden oft sehr große Landflächen gemappt. Es ist nicht verwunderlich, dass hierbei, allein aufgrund der Größe der Gebilde, oft Löcher in den Flächen vorkommen. Zusammen decken diese drei Typen schon fast 90% der Multipolygone ab.

2.116 Multipolygone konnten keinem Typ zugeordnet werden, entweder, weil kein Typ existiert, oder weil dieser so selten vorkommt, dass er aus den anderen Tags nicht mehr hervorsticht.

Umgekehrt sind 6.119 Multipolygone mehr als einem Typ zugeordnet, die beiden Multipolygone mit den IDs 4063929 und 6431041 sind sogar gleich 4 Typen zugeordnet.

| Anzahl | % | Typ |
|---|---|---|
| 138.865 | 56,8 | `landuse` |
| 54.493 | 22,8 | `natural` |
| 24.948 | 10,2 | `building` |
| 9.212 | 3,8 | `leisure` |
| 5.895 | 2,4 | `amenity` |
| 3.593 | 1,5 | `highway` |
| 2.492 | 1,0 | `public_transport` |
| 1.872 | 0,8 | `place` |
| 1.765 | 0,7 | `building:part` |
| 1.737 | 0,7 | `area:highway` |
| 1.459 | 0,6 | `man_made` |
| 949 | 0,4 | `golf` |
| 909 | 0,4 | `tourism` |
| 548 | 0,2 | `power` |
| 2.116 | 0,9 | *Ohne häufigen Typ* |

Tabelle 2: Die Multipolygon-Typen. Die Liste wurde durch Auswertung der Daten in der Datenbank und Einträgen im OSM-Wiki[16] bestimmt.

Ersteres ist der Luisenplatz in Darmstadt, der sowohl als Überwachungsanlage (`man_made=surveillance`), Busbahnhof (`amenity=bus_station`), Haltestelle (`public_transport=station`) und Fußgängerzone (`highway=pedestrian`) gemappt ist, als Platz aber übrigens nicht.

Zweiteres ist der Bismarckturm in Leipzig, der eigentlich genausogut als `way`-Fläche hätte gemappt werden können, da das Multipolygon nur aus den vier Seitenwänden des Turms besteht. Laut Daten ist der Turm sowohl ein Gebäude (`building=yes`), als auch Teil eines Gebäudes[17] (`building:part=yes`), ein Turm (`man_made=tower`) und ein Aussichtspunkt (`tourism=viewpoint`).

### 1.4.3 Bei Wegen

In der Datenbank sind 64.164.557 Wege gespeichert. 453.480 davon haben keine Tags und dienen lediglich als Elemente einer Relation. Die

[14] Woran das liegt, ist nicht ganz klar. Möglicherweise liegt es daran, dass man viele Relationen, da sie abstrakte Zusammenhänge repräsentieren, nicht sehen kann und somit auch keine rechte Rückmeldung bekommt, ob das, was man gemappt hat, auch richtig ist. Auch ist das Editieren von Relationen und Elementen die Teil einer Relation sind vergleichsweise schwierig, was oft auch dazu führt, dass Mapper auf Änderungen verzichten. Ob bessere Unterstützung durch die Editoren das verbessern könnte, ist unklar.

[15] Traffic Message Channel, ein digitaler Verkehrsnachrichtendienst.

[16] `https://wiki.openstreetmap.org/wiki`

[17] Aus den Daten erschließt sich nicht, von welchem Gebäude der Turm ein Teil sein soll, vermutlich ist diese Angabe falsch, oder sie bezieht sich darauf, dass der Turm früher mal ein Teil eines Gebäudes gewesen war.

anderen 63.711.077 Wege wurden weiter untersucht:

Genau wie bei den Multipolygonen, gibt es bei den Wegen keinen eindeutigen Tag, der den Typ des Wegs identifiziert. Mit der selben Methode wie dort wurden hilfsweise die Typen in Tabelle 3 bestimmt. Auffällig ist, dass es zwei Typen (`building` und `highway`) gibt, die über 80% der Wege ausmachen.

| Anzahl | % | Typ |
|---:|---:|---|
| 36.945.839 | 58,0 | `building` |
| 14.465.646 | 22,7 | `highway` |
| 4.455.110 | 7,0 | `landuse` |
| 2.055.619 | 3,2 | `natural` |
| 1.506.258 | 2,4 | `waterway` |
| 1.208.786 | 1,9 | `amenity` |
| 1.167.865 | 1,8 | `barrier` |
| 505.903 | 0,8 | `leisure` |
| 387.395 | 0,6 | `railway` |
| 347.786 | 0,5 | `man_made` |
| 339.488 | 0,5 | `building:part` |
| 295.805 | 0,5 | `power` |
| 114.462 | 0,2 | `boundary` |
| 92.832 | 0,1 | `shop` |
| 79.687 | 0,1 | `indoor` |
| 79.184 | 0,1 | `public_transport` |
| 68.541 | 0,1 | `historic` |
| 67.610 | 0,1 | `golf` |
| 57.911 | 0,1 | `tourism` |
| 53.002 | 0,1 | `area:highway` |
| 21.186 | 0,0 | `place` |
| 17.217 | 0,0 | `allotments` |
| 16.815 | 0,0 | `aeroway` |
| 13.449 | 0,0 | `traffic_calming` |
| 13.323 | 0,0 | `playground` |
| 215.819 | 0,3 | *Ohne häufigen Typ* |

Tabelle 3: Die Weg-Typen. Die Liste wurde durch Auswertung der Daten in der Datenbank und Einträgen im OSM-Wiki bestimmt.

Aus den verbleibenden Wegen wurden 23 Typen bestimmt, die im Vergleich zu Relationen und Multipolygonen immer noch häufig vorkommen. 215.819 Wege lassen sich aber keinem dieser Typen zuordnen und eine weitere Typisierung ist schwierig. Umgekehrt sind 850.190 Wege mehreren Typen zugeordnet, 33 davon sogar gleich 5 verschiedenen Typen.

Ein schwieriges Problem, bei den Wegen, ist die Frage, ob damit ein Linienzug oder eine Fläche dargestellt wird. Ein einheitliches Verfahren, um das zu bestimmen, gibt es nicht.[18] Theoretisch gibt es den Tag `area=yes/no`, aber dieser wird nicht zur Unterscheidung zwischen Linie und Fläche genutzt, sondern zum Überschreiben des (ungenau definierten) Standardwerts, was die Sache eher schwieriger macht, als leichter.

### 1.4.4 Bei Punkten

Es gibt 393.631.442 Punkte in der OSM-Datenbank, 95% davon (genauer 375.755.235) haben keine Tags und dienen nur als Stützpunkte für Wege oder als Mitglieder einer Relation. Untersucht wurden nur die anderen 17.876.207 Punkte.

Wie schon bei Multipolygonen und Wegen gibt es bei den Punkten ebenfalls keinen Tag, der das Objekt klar identifiziert. Mit der gleichen Methode, wie dort, ist die Tabelle 4 entstanden: Im Vergleich zu Wegen und Multipolygonen, sind hier deutlich mehr Typen vorhanden.

Adressen spielen unter den Punkten eine Sonderrolle, da sie ein abstraktes Gebilde und keinen Punkt repräsentieren. Sie machen etwas über 20% aller Punkte aus. Zudem gibt es keinen Tag, der diesen Typ charakterisiert, weshalb hier der Hausnummern-Tag (`addr:housenumber`) verwendet wurde.

Ebenfalls häufig sind die Typen `natural`, `highway`, `amenity` und `power`. Danach wurden noch 37 weitere Typen identifiziert, wobei an einigen Stellen unklar ist, ob man noch von einem eigenen Typ sprechen kann. So wurde der „Typ“ `crossing` aus der Liste entfernt, da Elemente mit diesem in den meisten Fällen zum Typ `highway` gehören, manchmal, wie in Abschnitt 1.3.2 geschildert, aber nicht. Andererseits wurden einige Typen in der Liste gelassen, die in mehr als 50% der Fälle zu einem anderen Typen gehören. Die Grenzen lassen sich schlicht und ergreifend nicht klar ziehen.

Die beiden Schlüssel `public_transport` und `traffic_sign` sind problematisch, da sie teilweise zusammen mit anderen Schlüsseln verwendet werden (`railway` bzw. `highway`) und teilweise nicht. Zudem wird an Stelle

[18] Siehe https://wiki.openstreetmap.org/wiki/Area#Tags_implying_area_status

von `traffic_sign` mit Zusatz `direction` gelegentlich `traffic_sign:forward` oder `traffic_sign:backward` genutzt, um alles in einem Tag unterzubringen.

| Anzahl | % | Typ |
|---|---|---|
| 3.856.951 | 21,6 | `addr:housenumber` |
| 3.139.611 | 17,6 | `natural` |
| 2.014.104 | 11,3 | `highway` |
| 2.002.885 | 11,2 | `amenity` |
| 1.142.403 | 6,4 | `power` |
| 881.460 | 4,9 | `entrance` |
| 867.718 | 4,9 | `barrier` |
| 756.940 | 4,2 | `emergency` |
| 626.807 | 3,5 | `public_transport` |
| 547.044 | 3,1 | `railway` |
| 481.991 | 2,7 | `tourism` |
| 363.856 | 2,0 | `shop` |
| 272.506 | 1,5 | `man_made` |
| 248.584 | 1,4 | `historic` |
| 240.004 | 1,3 | `traffic_sign` |
| 202.882 | 1,1 | `noexit` |
| 197.933 | 1,1 | `place` |
| 121.761 | 0,7 | `TMC:*` |
| 121.131 | 0,7 | `leisure` |
| 99.483 | 0,6 | `ele` |
| 86.385 | 0,5 | `office` |
| 76.347 | 0,4 | `seamark:type` |
| 73.592 | 0,4 | `door` |
| 69.949 | 0,4 | `healthcare` |
| 42.413 | 0,2 | `traffic_calming` |
| 41.356 | 0,2 | `playground` |
| 35.894 | 0,2 | `sport` |
| 34.035 | 0,2 | `craft` |
| 23.332 | 0,1 | `advertising` |
| 19.877 | 0,1 | `building` |
| 18.108 | 0,1 | `marker` |
| 17.467 | 0,1 | `waterway` |
| 16.263 | 0,1 | `network:type` |
| 14.554 | 0,1 | `traffic_sign:forward` |
| 12.271 | 0,1 | `ford` |
| 12.202 | 0,1 | `pipeline` |
| 12.030 | 0,1 | `golf` |
| 11.753 | 0,1 | `aeroway` |
| 9.795 | 0,1 | `traffic_sign:backward` |
| 9.193 | 0,1 | `waterway:sign` |
| 5.509 | 0,0 | `aerialway` |
| 4.432 | 0,0 | `cemetery` |
| 755.524 | 4,2 | *Ohne häufigen Typ* |

Tabelle 4: Die Punkt-Typen. Die Liste wurde durch Auswertung der Daten in der Datenbank und Einträgen im OSM-Wiki bestimmt.

Ein weiterer Problemfall ist `ele`, ein Tag, der eigentlich nur in Zusammenhang mit anderen Tags verwendet werden sollte, um an besonderen Stellen (z.B. dem Gipfel eines Berges) die Höhe anzugeben. In der Anfangszeit von OpenStreetMap wurde dieser Tag jedoch von einigen Benutzern an jedem `node` angegeben. Man hatte damals gehofft, dadurch eine Höhenkarte zu ermöglichen. Diese Idee wurde schon lange verworfen, aber in den Daten sind noch zahlreiche Überbleibsel aus dieser Zeit vorhanden.

Wie auch schon bei den Relationen gibt es auch bei den Punkten keinen sinnvollen Typ für TMC-Einrichtungen. Stattdessen hat man es hier mit zahlreichen Subsubsubschlüsseln zu tun.

755.524 Elementen konnte überhaupt kein Typ zugeordnet werden. Das sind immerhin mehr als 4% aller Punkt-Elemente.

Etwa doppelt so viele, nämlich 1.535.366 Punkte sind mehr als einem Typen zugewiesen, zwei davon sogar 6: Mit der ID 653314156 die Praxis von Dr. Bernhard Warmbrunn, die sowohl Adresse (`addr:housenumber=13`), Türe (`door=yes`), Arztpraxis (`amenity=doctors`), Praxis eines Arztes (`office=physician`), Arbeitsplatz eines Arztes (`healthcare=doctor`) als auch Eingang (`entrance=yes`) ist, und mit der ID 3298561284 der Felsenkeller Schwandorf, der sowohl etwas Historisches (`historic=yes`), ein Kellereingang (`man_made=cellar_entrance`), eine Touristenattraktion (`tourism=attraction`), eine Adresse (`addr:housenumber=7`), ein Gebäude (`building=yes`) und ein Eingang (`entrance=yes`) ist.

In beiden Fällen wurden Dinge auf unterschiedliche Weisen gemappt: Ein Kellereingang ist ein Eingang und eine Arztpraxis ist die Praxis eines Arztes und natürlich dessen Arbeitsplatz.

Ein Beispiel, bei dem einfach nur verschiedene Dinge an den selben Punkt gemappt wurden, ist der Berggipfel *Tiefenrother Höhe* (ID 806437880) mit den Schlüsseln `natural=peak`, `highway=emergency_access_point`, `tourism=information`, `place=locality` und `ele=551`, wobei der Schlüssel `ele` hier allerdings ein Zusatz zum Berggipfel und kein echter Schlüssel ist.

Es werden vier Dinge (Berggipfel, Rettungpunkt, Informationstafel und Flurname) vermischt und es bleiben Unklarheiten: Beispielsweise ist nicht klar, auf was sich der Tag `ref=SI 441772` bezieht. Ist das die Referenznummer des Berges, des Rettungspunktes (vermutlich) oder der Informationstafel? Und was, wenn sowohl Informationstafel als auch Rettungspunkt eine Referenznummer haben? Wie wird das angegeben?

# 2 Verbesserungsvorschläge

Aus den bisherigen Überlegungen ergeben sich einige Verbesserungsideen. Die wichtigsten sind nachfolgend gelistet. Man beachte dabei, dass diese Ideen aufeinander aufbauen.

## 2.1 Einführung eines Flächen-Datentypes

Es wäre sinnvoll, zusätzlich zu Punkten, Wegen und Relationen noch einen vierten Datentyp „Flächen“ einzuführen. Die Flächen würden als `area` bezeichnet und enthielten neben den Tags und den üblichen Metadaten noch Außen- und Innen-Wege, die die Fläche wie bei einem Multipolygon bestimmen, wobei mindestens ein Außen-Weg vorhanden sein muss.

Multipolygone ließen sich vollautomatisch in dieses Format überführen. Bei den Wegen nur die, bei denen ziemlich klar ist, dass es sich um eine Fläche handelt. Der Rest müsste dort nach und nach händisch sortiert werden.

Mir ist klar, dass diese Änderung einen immensen Aufwand bedeutet – nicht nur die gesamte OSM-Infrastruktur müsste daran angepasst werden, auch der Rest des OSM-Ökosystems muss mitziehen. Eine solche Änderung kann deswegen nur Schritt für Schritt erfolgen.

Aber ich denke, es ist nach wie vor nicht zu spät dafür und wenn man vor der Aufgabe, die ganze Welt zu mappen nicht zurückschreckt, dann sollte man auch hier nicht schon kapitulieren, bevor man angefangen hat. Was sollen denn unsere Enkel von uns sagen, wenn wir das nicht hinbekommen?!?

## 2.2 Jedes Element steht für genau ein Objekt

Derzeit gibt es Elemente die mehrere Objekte auf einmal mappen. Beispielsweise eine Wiese und den Zaun drumherum. Eine konsequente Trennung dieser Dinge wäre sinnvoll, damit klar ist, auf welches Objekt sich die zusätzlichen Tags beziehen.

## 2.3 Einführung eines Was-ist-das-Tags

Die Datenstruktur leidet sehr darunter, dass man von einem Element nicht klar sagen kann, welches Objekt es repräsentiert. Ein Was-ist-das-Tag würde genau diese Aufgabe übernehmen. Als Name für diesen Tag würde sich `type` anbieten, da es diesen bei den Relationen mit genau der gewünschten Aufgabe bereits gibt.

Einziges Problem hierbei sind die Multipolygone. Bei diesen ist ja bereits `type=multipolygon` gesetzt. Nach Einführung eines Flächen-Datentyps wäre dieses Problem gelöst. Bis dahin könnte man hilfsweise für Multipolygone auf einen Ersatz-Tag ausweichen, beispielsweise `multipolygon:type`.

Beispiel: Eine Straße erhält zusätzlich noch den Tag `type=highway`, eine Einrichtung den Tag `type=amenity`. Ein Weg, der derzeit gleichzeitig Obstplantage und Zaun um die Obstplantage ist würde aufgespalten und der Weg (oder besser die Fläche) für die Obstplantage bekäme `type=landuse` während der Weg für den Zaun `type=barrier` bekäme.

## 2.4 Verschieben der Life-Cycle-Präfixe an den Was-ist-das-Tag

Wenn man den Life-Cycle-Präfix nur noch an dem Was-ist-das-Tag anbringt, ist klar, dass es sich um einen Life-Cycle-Präfix handelt und nicht um einen Subschlüssel.

## 2.5 Vereinfachung von Tags

Wenn jedes Element nur noch für genau ein Objekt steht, können einige Schlüssel vereinfacht werden. Insbesondere Zuordnungs-Subschlüssel können ersatzlos entfallen.