Webanwendung zur Extraktion von Teildatensätzen aus DBpedia

Christian Ernst, Dominik Strohscheer, Hans Angermann
Till Nestler, Marvin Hofer, Robert Bielinski, Jonas Rebmann

Inhaltsverzeichnis

Recherchebericht

Begriffe

DBpedia

DBpedia ist ein Crowd-Sourced-Community-Projekt, welches sich mit der Sammlung strukturierter Daten im RDF Format aus der Web-Enzyklopädie Wikipedia beschäftigt. Es werden komplexe Anfragen an Wikipedia und die Verlinkung verschiedenster Datensätze zu Wikipedia ermöglicht.[1] Der DBpedia RDF-Datensatz wird mit Hilfe des SPARQL-Endpunktes von OpenLink Vituoso zugänglich gemacht.

Ontologie

Eine Ontologie ist ein Netzwerk aus Begrifflichkeiten und den zwischen ihnen bestehenden Beziehungen. Sie dient dem Datenaustausch zwischen Anwendungen. Eine formale Sprache zur Beschreibung von Ontologien ist zum Beispiel das RDF-Schema.

Metadaten

Metadaten enhalten Informationen über die Merkmale anderer Daten, aber nicht diese Daten selbst.

URI

Ein Uniform Resource Identifier (kurz URI) ist eine Zeichenfolge die eine abstrakte oder physische Ressource identifiziert.[2] Zum Beispiel gehören dazu Dateien oder Webseiten.

IRI

Generalisierung von URIs, die den UNICODE Zeichensatz unterstützt.[3]

URL

Ein Uniform Resource Locator (kurz URL) ist eine Unterart der URIs und identifiziert und lokalisiert eine Ressource.[4]

Datenbanksystem

System zur dauerhaften und effizienten Speicherung und Verwaltung großer Datenmengen.

Abfragesprache

Eine Abfragesprache ist eine Sprache zur Suche nach Information innerhalb einer Datenbank. Das Ergebnis einer solchen Abfrage (Query) ist eine Teilmenge der zugrundeliegenden Informationen (Filterung).

SPARQL

SPARQL (SPARQL Protocol and RDF Query Language) ist eine graphenbasierte Abfragesprache für RDF.[5]

Webanwendung

Eine Webanwendung ist ein Anwendungsprogramm, das in einem Webbrowser angezeigt und bedient wird. Sie liegt auf einem Webserver auf den über ein Netzwerk zugegriffen werden kann und dann von einem Klienten geladen wird.

Semantic Web

Das Semantic Web macht Daten zwischen Rechnern einfacher austauschbar und einfacher verwertbar. Verknüpfungen zwischen Objekten die dem Menschen durch den Kontext klar werden, werden im Semantic Web auch den Maschinen zugänglich gemacht.

inkonsistente Daten

Daten gelten als inkonsistent, wenn aus der Gültigkeit dieser Daten ein Zustand folgt der widersprüchlich ist.

Ein Film kann beispielsweise keine zwei verschiedenen Laufzeiten haben.

RDF

Resource Description Framework (kurz RDF) gilt als grundlegender Baustein des Semantic Webs zur Beschreibung von Ressourcen und deren Metadaten bestehend aus Tripeln der Form (Subjekt, Prädikat, Objekt).

Tripel-Store

Ein Tripel-Store ist eine Datenbank in der die Daten in Form von Tripeln (Subjekt,Prädikat,Objekt) gespeichert sind und für Abfragen zur Verfügung stehen.

Linked Data

Linked Data beschreibt eine Methode strukturierte Daten zu veröffentlichen, sodass diese untereinander verlinkt werden können und so durch semantischer Anfragen nützlicher werden.

Mapping

Mapping beschreibt die Formatierung eines Datensatzes auf ein anderes formales Schema.

Literal

Literale sind atomare Werte die Werte direkt Darstellen. Literale beinhalten keine weiteren Verweise oder Weiterleitungen. Im Kontext von RDF kommen Literale nur als Objekte vor, da diese keinen neuen Tripel bilden können.

Java

Java ist ein objektorientierte Programmiersprache die mit dem Ziel entwickelt wurde möglichst simpel zu sein. Java ist an C und C++ angelehnt.[6]

Unittests

Unittests sind Testprogramme die kleine Einheiten ('units', zum Beispiel einzelne Klassen) des Projekts auf ihre Formale richtigkeit mithilfe von Testfällen prüfen.

Je nach wahl der Programmiersprache können Unittest-Werkzeuge entweder teil der Sprache selbst oder ihrer Standandbibliothek sein oder als externes Unittest-framework verfügbar sein.

CI

Continuous integration (CI) bezeichnet das automatische Erstellen von Softwarepaketen auf einem Server bzw. das Automatische durchführen von Unittests.

Verbreitete CI Systeme sind zum Beispiel Jenkins und Travis CI.

Konzepte

Semantic Web

Das Semantic Web ist eine Erweiterung des klassischen „Web of Documents" durch ein „Web of Data". Der Begriff beschreibt die Vision eines „Web of Linked Data" des W3Cs (World Wide Web Consortium). Das Ziel ist es Computern neue Möglichkeiten zu geben mit Daten, deren Inhalten und den Zusammenhängen zwischen ihnen zu arbeiten.

Das Semantic Web beschreibt einen großen Graphen der allen Sachen von Interesse mit einer eindeutigen Adresse als Knoten anlegt und mit eindeutig benannten Kanten miteinander verbindet.

Standards im Semantic Web:

Zur Identifikation der Entitäten und den weitergehenden Daten werden URIs benutzt.

Als gemeinsames Datenmodell zur Repräsentation von Aussagen wird RDF verwendet, dessen Vokabular in RDFS deklariert wird.

Als Anfragesprache wird SPARQL verwendet.

SPARQL

SPARQL ist eine RDF-Abfragesprache die mit SQL vergleichbar ist.

Durch ein kleines Beispiel kann man sich die Funktionsweise von SPARQL klarmachen:

```
1 PREFIX abc: http://example.com/example/
2 SELECT ?name ?ort
3 WHERE {
4     ?person a    abc:mitarbeiter.
5     ?person      abc:name     ?name.
6     ?person      abc:ort      ?ort.
7 }
```

Als erstes wird ein Präfix angegeben. Dadurch wird ein URI verkürzt dargestellt. Hier kann man statt `abc:` auch jederzeit `<http://example.com/example>` schreiben.

Die Eigentliche Abfrage beginnt bei dem `SELECT` Hier werden die abzufragenden Variablen angegeben. Zu beachten ist, dass Variablen immer mit einem `?` beziehungsweise einem `$` beginnen. Außerdem wird durch die Verwendung der

SELECT Klausel die Abfrage in Tabellenform zurückgegeben, alternativ kann man durch CONSTRUCT die Abfrage auch in RDF-Format zurückgeben oder durch ASK eine einfache Wahr/Falsch Abfrage tätigen.

Was genau gesucht wird wird in der WHERE Klausel definiert. Hier wird eine weitere Variable

?person erschaffen die einen Mitarbeiter in http://example.com/example/mitarbeiter‘ darstellt.

Nun werden Tripel (Subjekt, Prädikat, Objekt) betrachtet die ?person als Subjekt haben.

Unter ?name werden zu jedem Subjekt ?person mit dem Prädikat abc: name die zugehörigen Objekte ausgegeben.

Unter ?ort werden demnach zu jedem Subjekt ?person mit dem Prädikat abc:ort die zugehörigen Objekte ausgegeben.

RDF

RDF ist ein Modell zum beschreiben von Aussagen bestehend aus dem Tripel Subjekt, Prädikat und Objekt. Dabei wird von jedem Tripel eine Relation zwischen dem Subjekt und dem Objekt beschrieben.[7] Ein Beispiel wäre “Person1 ist Mitarbeiter in Abteilung1” wobei ‘Person1’ das Subjekt, ‘ist Mitarbeiter in’ das Prädikat und ‘Abteilung1’ das Objekt wäre.

Somit sind Subjekt und Objekt zwei Knoten eines gerichteten Graphen die durch das Prädikat als Kante verbunden werden.

RDF/XML

RDF/XML ist eine Syntax um einen RDF Graphen als XML Datei darzustellen.

Das obige Beispiel lässt sich folgend in RDF/XML darstellen:

```
<rdf:RDF [...] xmlns:ex="http://example.com/">
    <rdf:Description rdf:about="http://example.com/Person1">
        <ex:IstMitarbeiter rdf:resource="http://example.com/Abteilung1" />
    </rdf:Description>
</rdf:RDF>
```

Das beschriebene Element Person1ist das Subjekt, IstMitarbeiter das Prädikat und Abteilung1 das Objekt des RDF Tripels.

rdf:Description beschreibt ein Element. Die URI des zu Beschreibenden Elements wird wird durch rdf:about festgelegt. Das Objekt kann direkt nach dem Prädikat durch rdf:resource oder als eigenständiger rdf:Description Teil angegeben werden:

```
<rdf:RDF [...] xmlns:ex="http://example.com/">
    <rdf:Description rdf:about="http://example.com/Person1">

        <ex:IstMitarbeiter>
            <rdf:Description rdf:about="http://example.com/Abteilung1" />
                <ex:AbteilungName>MusterAbteilung</ex:AbteilungName>
            </rdf:Description>
        </ex:IstMitarbeiter>
    </rdf:Description>
</rdf:RDF>
```

So ist es möglich das Prädikat noch weiter zu beschreiben.

Literale werden einfach als Inhalt eines Prädikaten Abschnitts angegeben. Wie in diesem Beispiel

```
<ex:AbteilungName>MusterAbteilung</ex:AbteilungName>
```

Turtle

Alternative zu RDF/XML die den Vorteil bietet für den Menschen sehr leicht lesbar zu sein.

In Turtle lässt sich das obige Beispiel so darstellen:

```
@prefix ex: <http://www.example.com/>.

    ex:Person1  ex:IstMitarbeiter   ex:Abteilung1.
```

Merkmale:

URIs werden in spitzen Klammern angegeben, Literale in Anführungszeichen.

URIs können wie bei SPARQL durch die Definition eines Präfix abgekürzt werden. Hier steht ex: für <http://www.example.com/>.

Endet eine Zeile mit einem Punkt ist das Tripel abgeschlossen und die nächste Zeile beginnt wieder mit einem neuen Subjekt. Möchte man ein Subjekt beibehalten beendet man die Zeile mit einem Semikolon und die nächste Zeile beginnt mit einem Prädikat. Möchte man auch das beibehalten beendet man die Zeile mit einem Komma.

Beispiel:

```
@prefix ex: <http://www.example.com/>.

    ex:Person1  ex:IstMitarbeiter   ex:Abteilung1;

            ex:Arbeitszeit      ex:Schicht1,

                        ex:Schicht2.
```

Aspekte

Aktueller Stand / Nutzen

DBpedia hat Unmengen an Daten gesammelt und gespeichert. Das Problem ist, dass diese Daten teilweise Inkonsistenzen aufweisen. Zum Beispiel können für einen Film mehrere Einträge vorhanden sein, die alle unterschiedliche Lauflängen aufweisen. Diese Inkonsistenzen sind natürlich unerwünscht und sollen im Laufe der Zeit behoben werden, doch dies ist für so ein riesiges Datennetz wie DBpedia sehr umständlich. Deswegen wird eine Webanwendung benötigt, die hilft, dieses Problem leichter zu beheben.

Unser Ziel

Unser Ziel ist es eine Anwendung mit passendem Webinterface zu schreiben, die genau diese Inkonsistenzen aufzeigt, die Fehlereigenschaften zur Lösung des Problems untersucht, behebt, Statistiken anzeigt und die konsistenten Datensätze zum Download in einem Format der Wahl (RDF/XML, Turtle, Ntriples) bereitstellt. Zur Umsetzung dieser Anwendung steht uns der OpenLink Virtuoso SPARQL-Endpunkt von DBpedia und das Tool RDFUnit zur Verfügung.

Inkonsistenzen Herausfiltern

Zum herausfiltern der Inkonsistenzen werden bestimmte SPARQL Anfragen an den Virtuoso Endpoint gestellt. Häufig erkennt man die Inkonsistenz daran, dass zwei Werte, die eigentlich gleich sein sollten, einen unterschiedlichen Datentyp aufweisen. Mit einer SPARQL Anfrage können wir also alle gespeicherten Werte und ihren Datentyp ausgeben um Inkonsistenzen dieser Art zu finden. Für zahlreiche weitere mögliche Fehlerquellen bietet RDFUnit passende SPARQL Schablonen. Desweiteren ist es nötig, nachdem man eine Inkonsistenz gefunden hat, den Fehlerhaften Datensatz zu analysieren. Dazu sollte man hinter jeder Eigenschaft die Anzahl der Einträge mit dem jeweiligen Datentyp anzeigen, um so erkennen zu können wie inkonsistent der Datensatz ist. Außerdem ist es ratsam die Möglichkeit zu bieten, für eine frei wählbare Eigenschaft anzuzeigen ob und wie viele Ressourcen mehrere Tripel mit dieser Eigenschaft haben.

RDFUnit

RDFUnit ist ein Tool welches manuelle und automatische Tests über ein RDF Datenset laufen lassen kann.
Es benutzt SPARQL Anfrage Schablonen um häufige Fehler-Zustände in den Datensets zu beschreiben und nach Anwendung auf einen bestimmtes Datenset diese Fehler herauszufiltern.[8]

Inkonsistenzen beheben

Zum Beheben der Inkonsistenzen wird muss für jede Eigenschaft gewählt werden welchen Datentyp sie haben soll. Standardmäßig wäre dies der häufigste. Hat diese Eigenschaft für eine Ressource mehrere unterschiedliche Werte, so muss ein Verfahren zur Auswahl eines Wertes wie zum Beispiel größter, kleinster, oder bestimmter Typ gewählt werden. Dazu muss gewählt werden welchen rdf:type Ressourcenobjekte haben sollen (von welcher Klasse sie eine Instanz sind), welche Eigenschaften auf einander gemappt werden sollen und ob inkonsistente Daten angepasst bzw. ausgeschlossen werden.

Wahl der Programmiersprache

Da das Erkennen und Beheben der Inkonsistenzen nicht Teil von DBpedia selbst, sondern eine eigenständige Anwendung werden soll, steht uns die Wahl der Programmiersprache für diesen Teil der Zielsetzung offen.

Bisher existiert wohl schon ein Tool mit diesem Zweck, das in Java Programmiert ist. Java bietet auch weitere Vorzüge:

- Java ist wohl die einzige Programmiersprache mit der wir alle - im Rahmen des Studiums - schon Erfahrungen gesammelt haben.
- es besitzt eine sehr Umfangreiche Standardbibliothek weshalb man sich kaum mit externen Bibliotheken und Dokumentationen beschäftigen muss.
- es ist Just-In-Time Kompiliert und läuft in einer Virtuellen Maschine weshalb die Fertigstellung eines Softwarepakets verhältnismäßig einfach ist.[9]
- Java ist gut an SPARQL angebunden.[10]

Allerdings hat Java auch Nachteile:

- es ist weniger geeignet dafür, das Webinterface für die Anwendung zu schreiben weshalb eine weitere, zweite Sprache verwendet werden sollte.
- Java fehlen Aufgrund des Konzepts möglichst einfach zu sein viele Funktionen die bei größeren Projekten bzw. der Verarbeitung größerer Datenmengen von großen Vorteil sind. Beispiele dafür sind Manuelle Speicherverwaltung, Kopie-Konstruktoren, Operatorenüberladung, RAII[11] und CTFE[12] sowie viele andere Kompiler-Abstraktionen[13][14].

Beispielhafte Alternativen zu Java sind

- **Python**: Eine Skriptsprache die zwar weniger Performance bietet aber leicht verständlich, konsistent und auch zum Entwickeln des Webinterface-backends[15] geeignet ist.[16]
- **Rust**: Eine moderne Programmiersprache die verschiedenste Paradigmen unterstützt, ein hohes Maß an Optimierung erlaubt, sehr sichere Programmierung aber zugleich auch *low-level* Zugriff erlaubt[17]. Rust in eine sehr Junge Sprache aber inzwischen stabil mit weit entwickelten Werkzeugen.[18]
- **Ruby**: Ruby ist eine interpretierte Programmierpsrache die, ähnlich wie Python, viele Paradigmen unterstützt allerdings mehr auf Objektorientierung ausgelegt ist.[19] Ruby sollte zur Entwicklung unseres Web-Backends[20] sowie der Kernanwendung geeignet sein[21]. Ruby funktioniert gut mit verbreiteten CI-Systemen wie Travis CI[22] und verfügt über ein Unittest Framework[23].
- **PHP**: *PHP Hypertext Preprocessor* ist eine Skriptsprache die zur Erstellung dynamischer Webseiten entwickelt wurde[24]. Deshalb wäre php in erster Linie für die Serverseitige Programmierung unseres Webinterfaces, weniger aber für das Backend geeignet. Vorteile von php sind die einfache Erlernbarkeit und die breite Verfügbarkeit auf Webservern. Anderweitig ist die Verfügbarkeit von php eher schlecht.

Die Entscheidende Frage ist wohl ob die Teammitglieder im Rahmen des Projekts eine neue Programmiersprache lernen oder das bekannte Java verwenden wollen und wie gut die jeweiligen Sprachen an die zu verwendenden Datenbanksysteme angebunden ist.

Durchführung von Tests

Neben dem manuellen Testen der Anwendung sollten im Laufe der Entwicklung Unittests geschrieben werden. Es wäre möglich auf dem Projektserver einen CI Server einzurichten und die Unittests regelmäßig automatisch durch zu führen um die Qualität der Anwendung sicher zu stellen.

Besonders sollte auch das Webinterface im Bezug auf Sicherheitslücken getestet werden, zum Beispiel auch mithilfe eines *Web application security scanners*.

Fußnoten

1 http://wiki.dbpedia.org/

2 https://tools.ietf.org/html/rfc3986

3 https://tools.ietf.org/html/rfc3987

4 https://tools.ietf.org/html/rfc3986#section-1.1.3

5 http://www.w3.org/TR/sparql11-query/

6 https://docs.oracle.com/javase/specs/jls/se8/jls8.pdf

7 http://www.w3.org/TR/rdf-syntax-grammar/

8 http://aksw.org/Projects/RDFUnit.html

9 https://docs.oracle.com/javase/specs/jls/se8/jls8.pdf

10 http://www.w3.org/wiki/SparqlImplementations

11 Resource Acquisition Is Initialization: Ressourcen werden Belegt wenn ihr Gültigkeitsbereich betreten- und freigegeben wenn er Verlassen wird. Dabei kann eine Destruktor-Methode z.B. abhängige Ressourcen freigeben. In Java werden Ressourcen zwangsläufig immer von einem Garbage-Collector verwaltet was in einigen Anwendungsfällen zu Problemen führt so unterstützt Java zum Beispiel infolgedessen keine Destruktoren und Ressourcen müssen oft manuell freigegeben werden.

12 Compile time function execution: Die Ausführung von Funktionen deren Parameter bekannt sind zur Zeit der Kompilierung.

13 Gemeint sind Abstraktionen die zur Laufzeit keinen Overhead erzeugen also zur Kompilierzeit ausgeführt werden. So wird in Java Reflexion zum Beispiel zwangsläufig in Laufzeit durchgeführt was neben Overhead auch andere Probleme bringt.

14 http://docs.oracle.com/javase/tutorial/reflect/index.html

15 Gemeint ist der Serverseitige Teil des Webinterfaces.

16 https://wiki.python.org/moin/WebFrameworks

17 https://de.wikipedia.org/wiki/Rust_%28Programmiersprache%29

18 https://www.hosteurope.de/blog/rust-1-0-es-ist-serviert/

19 https://www.ruby-lang.org/en/about/

20 http://guides.rubyonrails.org/getting_started.html

21 https://rubygems.org/gems/sparql/versions/1.99.0

22 https://docs.travis-ci.com/user/languages/ruby/

23 https://en.wikibooks.org/wiki/Ruby_Programming/Unit_testing

24 http://php.net/