Projekt-Bericht und Dokumentation im Kurs Wissensrepräsentation SoSe15

Lukas Hodel Richard Remus

21. Juli 2015

Contents

1 Aufgabenstellung

Die gegebene Aufgabenstellung lautete:

Führen Sie ein Semantic Web / Linked Data Projekt durch. Dies sollte mehrere der folgenden Aspekte umfassen:

- Verknüpfen von verschiedenen Datenquellen
- Konvertierung von Daten ins RDF-Format (mit entsprechender T-Box)
- Entwicklung von Ontologien
- Linked Data Anwendung
- Informationsextraktion/Web Scaping und Konvertierung in RDF
- Entity Resolution
- ..

Wir haben im Kurs bereits Erfahrungen mit dem Lobbyradar des ZDF und DBpedia gemacht.

2 Vorgehen

- Als Erstes werden wir die Daten des Lobbyradar auswerten und entscheiden, welche Informationen sinnvoll in RDF konvertierbar sind.
- Danach versuchen wir eine Ontologie anhand der gefundenen Beziehungen abzuleiten.
- Sind die wichtigen Informationen identifiziert, werden sie in einen RDF-Graphen exportiert.
- Auch soll die Anwendung eine Personen- und Organisations-Suche bieten. Die Ergebnisse sollen dann wiederum in den bestehenden Graphen eingepflegt werden, anschliessend wird ein Teilgraph angezeigt, in dessen Zentrum die gesuchte Entität steht und der die Relationen zu anderen Entiäten aufzeigt.
- Danach werden wir eine Anbindung zu DBpedia entwickeln, in der Sparql-Queries zu den Entitäten des Graphen erstellt formuliert und abgeschickt werden können.

3 Analyse Lobbyradar

Lobbyradar enthält Knoten welche in JSON persistiert sind. Damit werden alle Entities und Relations abgebildet. Die Knoten enthalten schon für sich eine große Zahl an Properties:


```
{'_id': '54bd3c768b934da06340f4c6',
 'aliases': [],
 'created': 'datetime.datetime(2015, 1, 19, 17, 18, 46, 529000)',
 'data': [{'auto': 'True',
   'created': 'datetime.datetime(2015, 1, 19, 17, 19, 9, 420000)',
   'desc': 'Quelle',
   'format': 'link',
   'id': '3dc1416e38deac59076cd3f0d4e1235de79cb530207d06c28053134cc8aa7732',
   'key': 'source',
   'updated': 'datetime.datetime(2015, 1, 19, 17, 19, 9, 420000)',
   'value': {'remark': 'created by lobbyliste importer',
    'url': 'http://bundestag.de/blob/189476/8989cc5f5f65426215d7e0233704b20a/
    lobbylisteaktuell-data.pdf'}},
  {'auto': 'True',
   'created': 'datetime.datetime(2015, 1, 19, 17, 19, 9, 420000)',
   'desc': 'Titel',
   'format': 'string',
   'id': '65873d29bd0ab6af91ef341689d28f4f0658cc851494a0ef34cfd07dd0cf5d42',
   'key': 'titles',
   'updated': 'datetime.datetime(2015, 1, 19, 17, 19, 9, 420000)',
   'value': 'Gesch\xe4ftsf\xfchrer'},
  {'auto': 'True',
   'created': 'datetime.datetime(2015, 1, 19, 17, 18, 46, 529000)',
   'desc': 'Titel',
   'format': 'string',
   'id': '1c98f43f0787b6dfcad25c38849ef3895bd26624a79c8fa9e6203c360d120fad',
   'key': 'titles',
   'updated': 'datetime.datetime(2015, 1, 19, 17, 18, 46, 529000)',
   'value': '2. Vorsitzender'}],
 'importer': 'lobbyliste',
 'name': 'Markus Hoymann',
 'search': ['markus hoymann'],
 'slug': 'markus hoymann',
 'tags': ['lobbyist', 'lobbyismus', 'executive'],
 'type': 'person',
 'updated': 'datetime.datetime(2015, 1, 19, 17, 19, 9, 426000)'}
```


Deshalb haben wir entschieden, von all diesen, nur die wichtigsten Eigenschaften abzubilden. Für Personen und Organisationen:

- id
- name
- created
- updated
- type *(Person, Organisation)*
- alias *(nur bei Organisationen)*
- tags *(als Interessengebiete)*

Für die Relationen zwischen diesen Entitäten benötigen wir eine Klassifikation.

Die verschiedenen Bezeichnungen für die Ämter und Anstellungsverhältnisse der Personen, also ihre Positionen in der Lobbyradar-Datenbank, sind jedoch stark arbiträr, gehorchen keiner Syntax und liegen nur als Literal vor. Es ist leider nicht ohne weiteres möglich einer Relation (Property) in RDF eine zusätzliche Bezeichnung als Literal zu geben. Dazu würden wir Hilfsknoten benötigen, welche den Graphen und somit die Suche stark verkomplizieren würden.

Da in der Datenbank auch viele Positionen mit semantisch gleichem Inhalt unterschiedliche Bezeichnungen haben, entweder weil keine einheitliche Benennung vorgegeben ist (*z.B. Bundesminister der Finanzen, Finanzminister, Minister der Finanzen*) oder weil die Importeure Tippfehler gemacht haben (*executive, ececutive*), haben wir uns entschieden, die wichtigsten Relationen semantisch zu gruppieren und für diese zusammenfassende Properties zu erstellen.

Die properties wurden so in der Datei "ontology.ttl" hierarchisch zusammengefasst.

Diese Zuordnung mussten wir händisch erledigen, aus Zeitgründen konnten wir also die restlichen Daten nicht Berücksichtigen.

4 Aufbau einer Ontologie

4.1 Konvertierung der Daten in RDF

Um die Daten in RDF übertragen zu können, mussten wir zunächst eine hierarchische Ontologie in `ontology.ttl` nach Turtle-Syntax erstellen.

4.2 T-Box

Zunächst mussten wir für einige Entitäten neue Klassen anlegen, da es zum Beispiel für Dinge wie *BTCertUID* oder *Bundesland* noch keine RDFS-Klassen gibt.

Die wichtigsten Klassen sind dabei:

```
# Partei
:Party a rdfs:Class ;
  rdfs:subClassOf org:Organization .

# Regierung
:Government a rdfs:Class ;
  rdfs:subClassOf org:Organization .

# Politiker
:Politician a rdfs:Class ;
  rdfs:subClassOf foaf:Person .

# Lobbyist
:Lobbyist a rdfs:Class ;
  rdfs:subClassOf foaf:Person .
```

Auch mussten wir eigene Properties anlegen, zum Beispiel:

```
# Mitglied
:isMemberOf a rdf:Property ;
  rdfs:domain foaf:Person ;
  rdfs:range :Organization .

#Vorsitzender
:isExecutiveOf rdfs:isSubPropertyOf :isMemberOf .

# steht in Verbindung mit einer Regierung
:isRelatedToGovernment rdfs:isSubPropertyOf :isMemberOf ;
  rdfs:domain foaf:Person ;
  rdfs:range :Government .
```

Mit diesen drei Properties oder deren subProperties werden die Relationen zwischen Personen und Organisationen oder Personen und Regierungen dargestellt. Diese subProperties sind beispielsweise:

```
# Stellvertretender Vorsitz, Vize-Präsident
:isVicePresidentOf rdfs:isSubPropertyOf :isExecutiveOf .

# Obmann
:isChairmanOf rdfs:isSubPropertyOf :isMemberOf .

#Staatssekretär
isSecretaryOfStateOf rdfs:isSubPropertyOf :isRelatedToGovernment .
```

Die Zuordnung war nicht trivial, da wir uns über passende Oberbegriffe gedanken machen mussten und die Einordnung von Hand geschehen musste.

4.3 A-Box

5 Aufbau des Graphen

6 Benutzung des Programms, Lobbyradar

Zu erst muss mal die Datei "Graph" und deren Funktionen importiert werden. Wichtig zu wissen ist das die libraries *pymongo*, *bson*, *rdflib*, *networkx* und *matplotlib* vorhanden sind.

```
import Graph
from Graph import search_persons, person_connections,  \
                  search_organizations, \
                  organization_connections, plot_tripples, \
                  search_governmental, search_sparql
%matplotlib inline # um schöne plotts zu erhalten
```

6.1 Nach Personen suchen

Mit der Methode **search_persons** kann mit Freitext nach Personen gesucht werden.

```
search_persons(name="angela m", limit=10)
```

```
[u'Angela Merkel', u'Angela Marquardt']
```

6.2 Verbindungen einer Person ausgeben

Hat man den korrekten Namen der Person, kann mit Hilfe der Methode **person_connections** nach "connections" also Organisationen gesucht werden. Es wird eine Liste von Triples mit ('personenname', 'property', 'Organisationslabel') zurück gegeben.

```
angela_connections = person_connections("Angela Merkel")
angela_connections[:2]
```

```
[(u'Angela Merkel',
  u'http://example.org/isOtherwiseRelatedToGovernment',
  u'Bundeskanzleramt'),
 (u'Angela Merkel',
  u'http://example.org/isOtherMemberOf',
  u'Atlantik-Br\xfccke e.V.')]
```

6.3 Resultat plotten

Die Methode **plot_tripples** visualisiert nun die tripples in einem Graphen. Dabei werden die "Objekte" Grün und die "Subjekte" rot dargestellt. In unserem Fall sind die Personen grün und die Organisationen rot dargestellt.

```
plot_tripples(angela_connections)
```

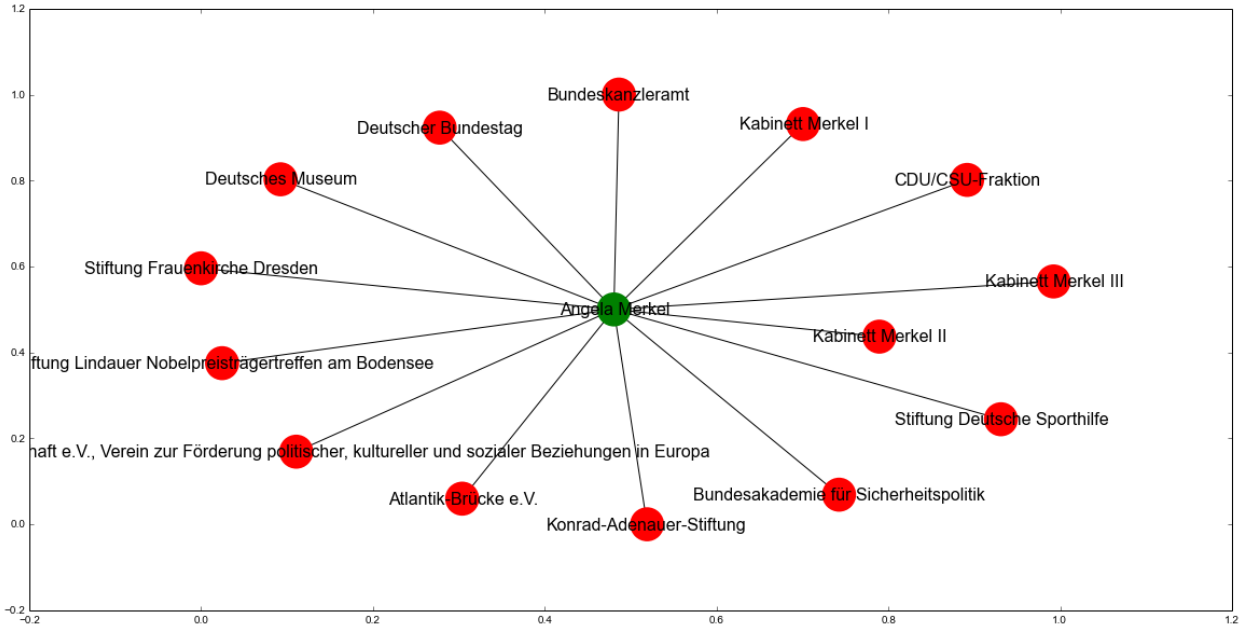

Figure 1: png

6.4 Nach Organisationen suchen

Mit der Methode **search_organizations** kann mit Freitext nach Organisationen gesucht werden.

```
search_organizations("SPD")
[u'SPD', u'SPD-NRW', u'SPD-Bundestagsfraktion', u'SPD Vorpommern']
```

6.5 Verbindungen einer Organisationen ausgeben

Nun kann mit der Methode **organization_connections** nach Personen-Verbindungen dieser Organisation gesucht werden. Zurück kommt wieder ein Tripple. Analog zur Methode **person_connections**

```
org_conn_tripple = organization_connections("SPD-Bundestagsfraktion")
org_conn_tripple[:4]

[(u'Bernhard Daldrup',
  u'http://example.org/isOtherMemberOf',
  u'SPD-Bundestagsfraktion'),
 (u'Gabriele Hiller-Ohm',
  u'http://example.org/isOtherMemberOf',
  u'SPD-Bundestagsfraktion'),
 (u'Daniela De Ridder',
  u'http://example.org/isOtherMemberOf',
  u'SPD-Bundestagsfraktion'),
 (u'Nina Dr. Nina Scheer',
  u'http://example.org/isOtherMemberOf',
  u'SPD-Bundestagsfraktion')]
```

6.6 Verbindungen einer Organisationen plotten

Diese können dann wieder mit der Methode **plot_tripples** geplottet werden. Wenn der Graphen zu gross wird, kann die Größe des Plots selbst über figsize angegeben werden.

```
plot_tripples(org_conn_tripple, figsize=(25,25))
```

6.7 Nach Bundes-Organisationen suchen

Mit der Methode **search_govenrmental** kann ausschliesslich nach Bundesorganisationen gesucht werden.

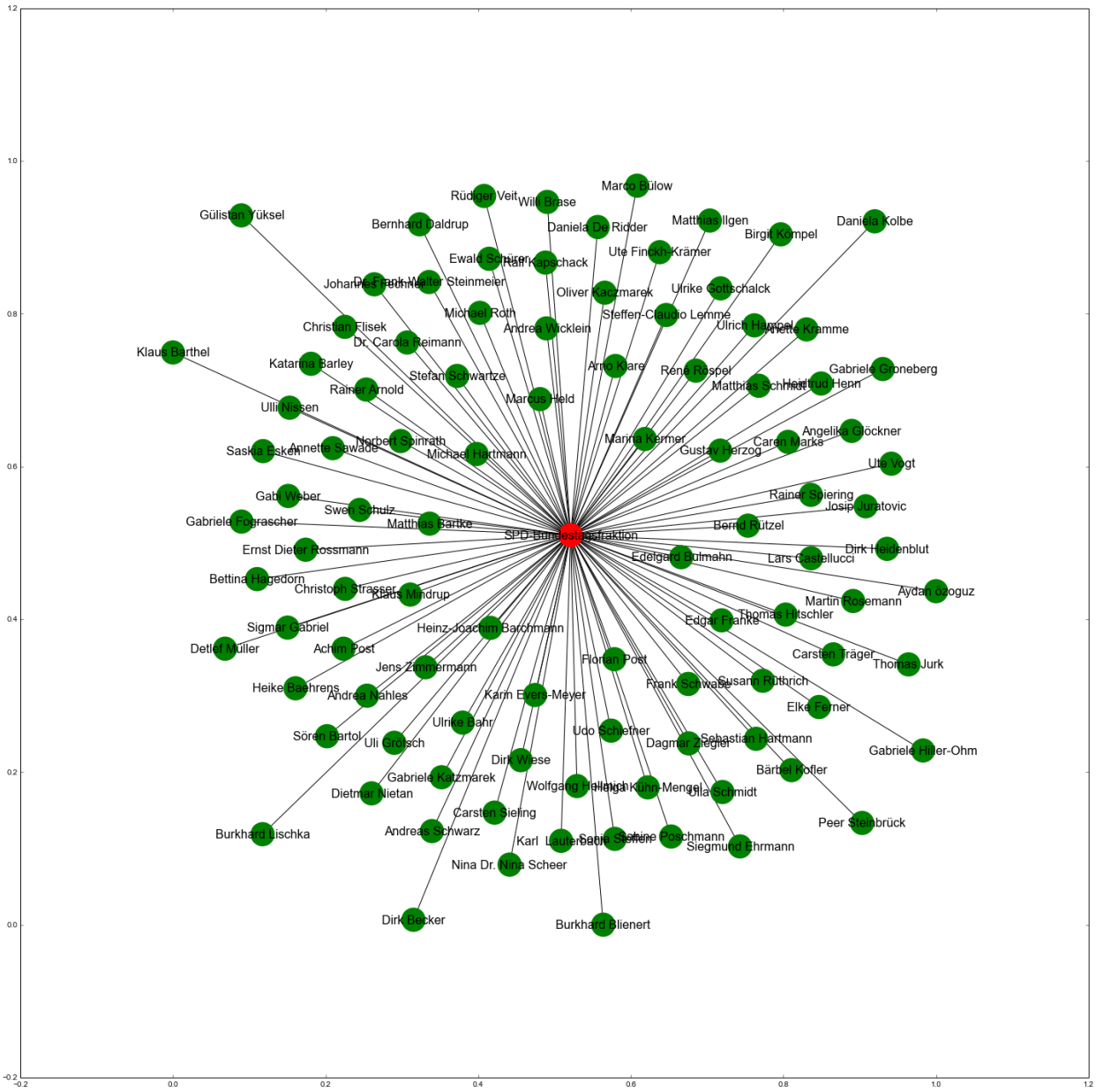
Marco Bülow
Rüdiger Veit
Willi Brase
Gülistan Yüksel
Bernhard Daldrup
Daniela De Ridder
Matthias Illgen
Daniela Kolbe
Birgit Kömpel
Ute Finckh-Krämer
Ewald Schurer
Karl Kapschack
Johannes Pflug
Frank-Walter Steinmeier
Oliver Kaczmarek
Ulrike Gottschalck
Michael Roth
Steffen-Claudio Lemme
Christian Flisek
Andrea Wicklein
Ulrich Hampel
Dr. Carola Reimann
Anette Kramme
Klaus Barthel
Katarina Barley
Arno Klare
Rene Röspel
Gabriele Groneberg
Stefan Schwartze
Matthias Schmidt
Hartmut Henn
Rainer Arnold
Markus Held
Ulli Nissen
Marina Kermer
Angelika Glöckner
Norbert Spinrath
Caren Marks
Saskia Esken
Annette Sawade
Michael Hartmann
Gustav Herzog
Ute Vogt
Gabi Weber
Rainer Spiering
Swen Schulz
Josip Juratovic
Gabriele Fograscher
Matthias Bartke
Bernd Rützel
SPD-Bundestagsfraktion
Ernst Dieter Rossmann
Dirk Heidenblut
Edelgard Bulmahn
Lars Castellucci
Bettina Hagedorn
Christoph Strasser
Aydan Özoguz
Klaus Mindrup
Martin Rosemann
Thomas Hitschler
Sigmar Gabriel
Heinz-Joachim Barchmann
Edgar Franke
Detlef Müller
Achim Post
Carsten Träger
Florian Post
Thomas Jurk
Jens Zimmermann
Susann Rüthrich
Heike Baehrens
Frank Schwabe
Andrea Nahles
Karin Evers-Meyer
Elka Ferner
Ulrike Bahr
Sören Bartol
Udo Schiefner
Sebastian Hartmann
Uli Grötsch
Dagmar Ziegler
Gabriele Hiller-Ohm
Dirk Wiese
Bärbel Kofler
Gabriele Katzmarek
Wolfgang Hellmich
Dietmar Nietan
Heiga Kühn-Mengel
Ulla Schmidt
Carsten Sieling
Peer Steinbrück
Burkhard Lischka
Andreas Schwarz
Karl Lauterbach
Sonja Steffen
Sabine Poschmann
Siegmund Ehrmann
Nina Dr. Nina Scheer
Dirk Becker
Burkhard Blienert

Figure 2: png

```
gov_organizations = search_governmental()
gov_organizations[:10]
```

```
[u'Nieders\xe4chsisches Kultusministerium',
 u'Bayrisches Staatsministerium f\xfcr Ern\xe4hrung, Landwirtschaft und Forsten',
 u'Bayrisches Staatsministerium f\xfcr Arbeit und Soziales, Familie und Integratio
 u'Ministerium f\xfcr Inneres und Kommunales Nordrhein-Westfalen',
 u'Hessisches Ministerium der Finanzen',
 u'Staatsministerium Baden-W\xfcrttemberg',
 u'Ministerium f\xfcr Landwirtschaft, Umwelt und Verbraucherschutz Mecklenburg-Vor
 u'Staatskanzlei Sachsen-Anhalt',
 u'Nieders\xe4chsisches Ministerium f\xfcr Ern\xe4hrung, Landwirtschaft und Verbra
 u'Ministerium f\xfcr Schule und Weiterbildung  Nordrhein-Westfalen']
```

```
plot_tripples(organization_connections("Bundesministerium der Justiz"))
```

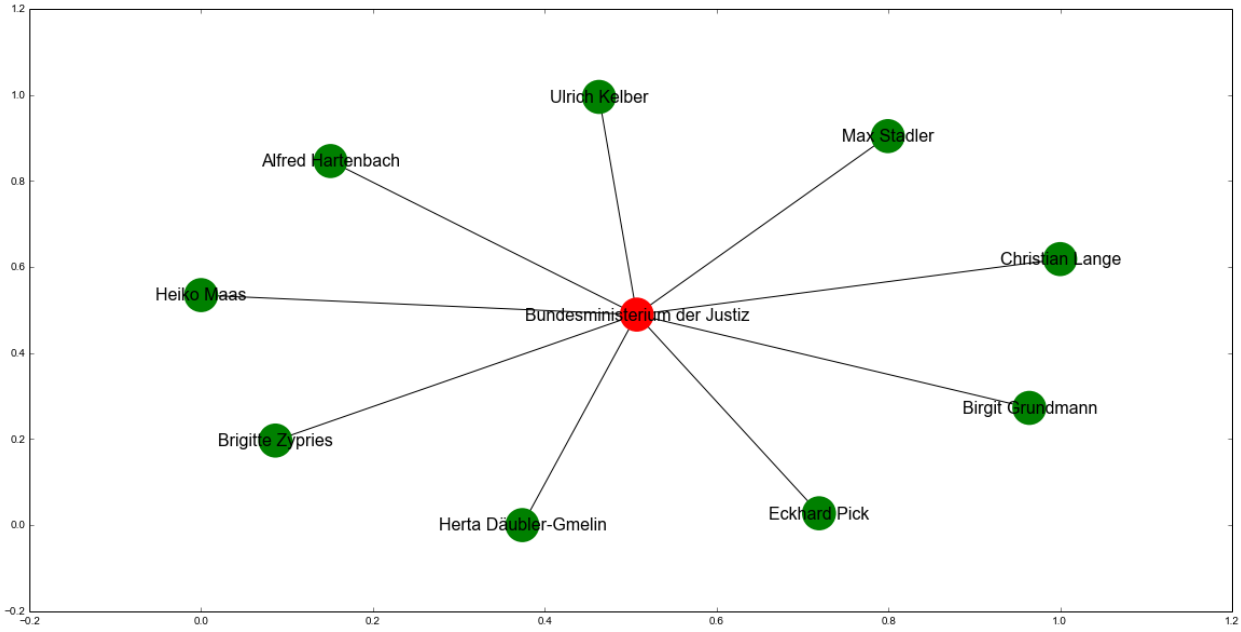

Figure 3: png

6.8 Eigene Sparql abfrage erstellen

Mit der Methode **search_sparql** kann eine eigene Sparql Abfrage abgeschickt werden. Zurück erhält man ein rdflib Abfrage-Resultat. Dieses kann dann selbst nach belieben verwendet werden.

Als Beispiel kann die Ontologie analysiert werden:

```
    res = search_sparql("""
        SELECT DISTINCT ?property ?subproperty
        WHERE {
            ?subproperty rdfs:isSubPropertyOf ?property .
        }
        LIMIT 100
    """)

    for row in res:
        print("%s -> %s" % row)
```

```
http://example.org/isMemberOf -> http://example.org/isOtherMemberOf
http://example.org/isMemberOf -> http://example.org/isOrdinaryMemberOf
http://example.org/isExecutiveOf -> http://example.org/isChairmanOfManagementOf
http://example.org/isExecutiveOf -> http://example.org/isAdministrationBoardMemberOf
http://example.org/isExecutiveOf -> http://example.org/isManagementMemberOf
http://example.org/isExecutiveOf -> http://example.org/isDirectorOf
http://example.org/isMemberOf -> http://example.org/isRelatedToGovernment
http://example.org/isExecutiveOf -> http://example.org/isChairmanOfDirectorsBoardOf
...
```

6.9 dbPedia informationen einer Person anzeigen

Mit der Methode **dbpedia_person** kann in dbpedia über eine spezifische Person informationen abgefragt werden. Momenan werden einfach die dbpedia uri und eine kleine biographie der Person ausgegeben. Wenn es mehrere Personen gibt die gleich heissen werden mehrere Zeilen zurückgegeben.

```
dbpedia_person("Angela Merkel")
```

```
[(u'http://dbpedia.org/resource/Angela_Merkel',
  u'beschreibung sehr lange...')]
```

Uri von Angela Merkel bei dbpedia http://dbpedia.org/resource/Angela_Merkel

Abstract von Angela Merkel aus dbpedia Angela Dorothea Merkel (* 17. Juli 1954 in Hamburg als Angela Dorothea Kasner) ist eine deutsche Politikerin. Bei der Bundestagswahl am 2. Dezember 1990 errang Merkel, die in der DDR als Physikerin ausgebildet wurde und auch tätig war, erstmals ein Bundestagsmandat; in allen darauffolgenden sechs Bundestagswahlen wurde sie in ihrem Wahlkreis direkt gewählt. Von 1991 bis 1994 war Merkel Bundesministerin für Frauen und Jugend im Kabinett Kohl IV und von 1994 bis 1998 Bundesministerin für Umwelt, Naturschutz und Reaktorsicherheit im Kabinett Kohl V. Von 1998 bis 2000 amtierte sie als Generalsekretärin der CDU. Seit dem 10. April 2000 ist sie Bundesvorsitzende der CDU und seit dem 22. November 2005 2013 mittlerweile in der

dritten Amtsperiode als Chefin von unterschiedlich zusammengesetzten Koalitionsregierungen deutsche Bundeskanzlerin. Sie ist die erste Frau und zugleich die achte Person in der Geschichte der Bundesrepublik, die dieses Amt innehat.

6.10 Eigene Sparql Anfrage an dbpedia senden

Möchte man nun noch genauere Informationen über die Person finden, kann man auch eine eigene Sparql Anfrage senden.

```
dbpedia_sparql_query("""
SELECT ?person ?birthDate
WHERE {
    ?person a dbpedia-owl:Person .
    ?person foaf:name ?name .
    ?person dbpedia-owl:birthDate ?birthDate .
    ?person dbpedia-owl:birthPlace ?birthPlace .
    FILTER(str(?person) = 'http://dbpedia.org/resource/Angela_Merkel')
}""")
```

zurück erhaltält man eine JSON in folgender Struktur, wobei die wichtigen informationen unter json[“results”][“bindings”] stehen. Auch wichtig zu wissen ist, dass die duplizierten Einträge aus den Einträgen aller Sprachen resultieren.


```
{
   "head":{
      "link":[

      ],
      "vars":[
         "person",
         "birthDate"
      ]
   },
   "results":{
      "distinct":false,
      "ordered":true,
      "bindings":[
         {
            "person":{
               "type":"uri",
               "value":"http://dbpedia.org/resource/Angela_Merkel"
            },
            "birthDate":{
```

```
                "type":"typed-literal",
                "datatype":"http://www.w3.org/2001/XMLSchema#date",
                "value":"1954-07-17"
            }
        },
        {
            "person":{
                "type":"uri",
                "value":"http://dbpedia.org/resource/Angela_Merkel"
            },
            "birthDate":{
                "type":"typed-literal",
                "datatype":"http://www.w3.org/2001/XMLSchema#date",
                "value":"1954-07-17"
            }
        },
        ...
    ]
  }
}
```


Dieses Json kann nun geparst und ausgelesen werden.

7 Nachbetrachtung

- rdflib sollte beim hinzufügen von Knoten prüfen, ob die gegebenen Tripel sinnhaft sind und auf Knoten verweisen, die bereits da sind. g.add() ist da nicht *verbose* genug und sollte ruhig mal meckern und nicht still alles hinnehmen.

8 Ausblick

- Daten genauer anschauen, z.B. Unterscheidung von Parteien zu anderen Organisationen
- Machine Learning zur Identifikation von Lobbyisten.