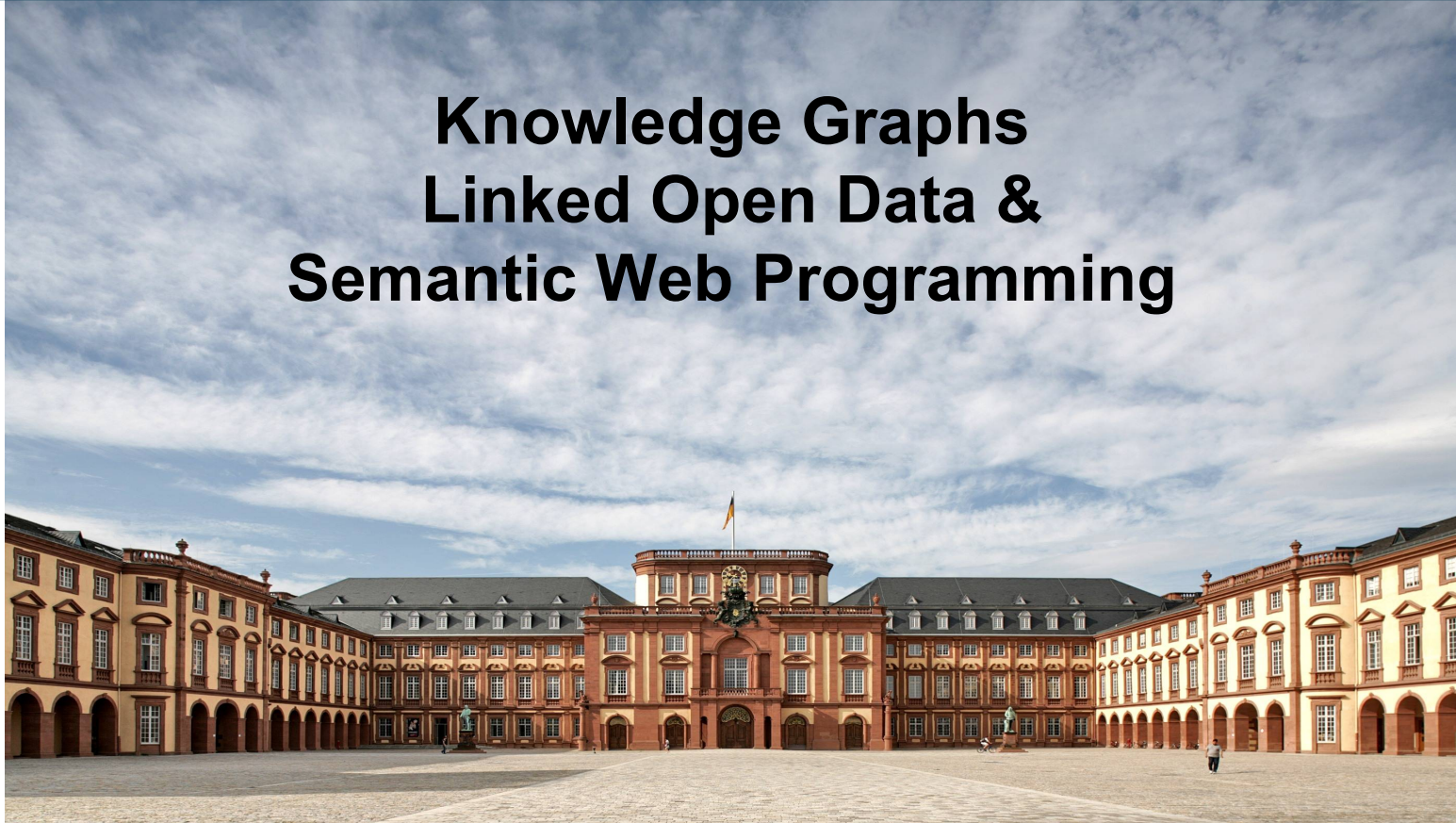


Knowledge Graphs Linked Open Data & Semantic Web Programming



Overview

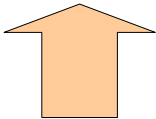
- Linked Open Data
 - Principles
 - Examples
 - Vocabularies
- Microdata & schema.org
- Introduction to Semantic Web Programming with rdflib & Jena

Linked Open Data

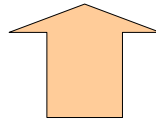
- What we've got to know up to now
 - RDF as a universal language for encoding knowledge
 - RDF Schema for describing vocabularies (i.e., classes and properties)
- How can we publish such knowledge?
- Linked Open Data
 - uses techniques like URIs, RDF, RDF schema
 - for publishing knowledge on the Web

Why “Linked” Open Data?

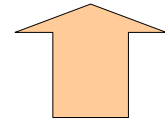
```
:p a :Physician .  
:p :hasDegree "Dr." .  
:p :hasName "Mark Smith" .  
:p :hasAddress :a .  
:a :street "Main Street" .  
:a :number "14"^^xsd:int .  
:a :city "Smalltown" .  
:p :hasOpeningHours [  
  a rdf:Bag ;  
  [ :day :Monday;  
    :from "9"^^xsd:int;  
    :to "11"^^xsd:int;  
  ]  
  ...  
]
```



```
:s a :City .  
:s :name "Smalltown" .  
:s :lat "49.86"^^xsd:double .  
:s :long "8.65"^^xsd:double .  
:s :district "Birmingham" .  
...
```



```
:d a :District .  
:d :name "Birmingham" .  
:d :pop "347891"^^xsd:int .  
:d :locatedIn "England" .  
...
```



Why “Linked” Open Data?

- Information is scattered on the Web
 - Publishing your own knowledge graph online just adds a scattered piece
 - “information silos”
- HTML has a concept for interlinking scattered information
 - known as *hyperlink*
 - More information at `W3C`
- Linked Open Data uses that principle, too

Why “Linked” Open Data?

```
:p a :Physician .  
:p :hasDegree "Dr." .  
:p :hasName "Mark Smith" .  
:p :hasAddress :a .  
:a :street "Main Street" .  
:a :number "14"^^xsd:int .  
:a :city  
  <http://.../smalltown> .  
:p :hasOpeningHours [  
  a rdf:Bag ;  
  [ :day :Monday;  
    :from "9"^^xsd:int;  
    :to "11"^^xsd:int;  
  ]  
  ...
```



```
:s a :City .  
:s :name "Smalltown" .  
:s :lat "49.86"^^xsd:double .  
:s :long "8.65"^^xsd:double .  
:s :district  
  <http://.../birmingham> .  
...
```



```
:d a :District .  
:d :name "Birmingham" .  
:d :pop "347891"^^xsd:int .  
:d :locatedIn "England" .  
...
```




Why “Linked” Open Data?

- Linked Open Data is RDF data
 - which is provided in a distributed manner
- URIs
 - have been used as simple identifiers so far
 - in LOD: links to data
 - resolvable!
 - "dereferencable URIs" (URLs)
 - can be used together with content negotiation, RDFa, etc.

Why “Linked” Open Data?

- Example:
 - `<#Heiko> :worksIn <http://dbpedia.org/resource/Mannheim> .`

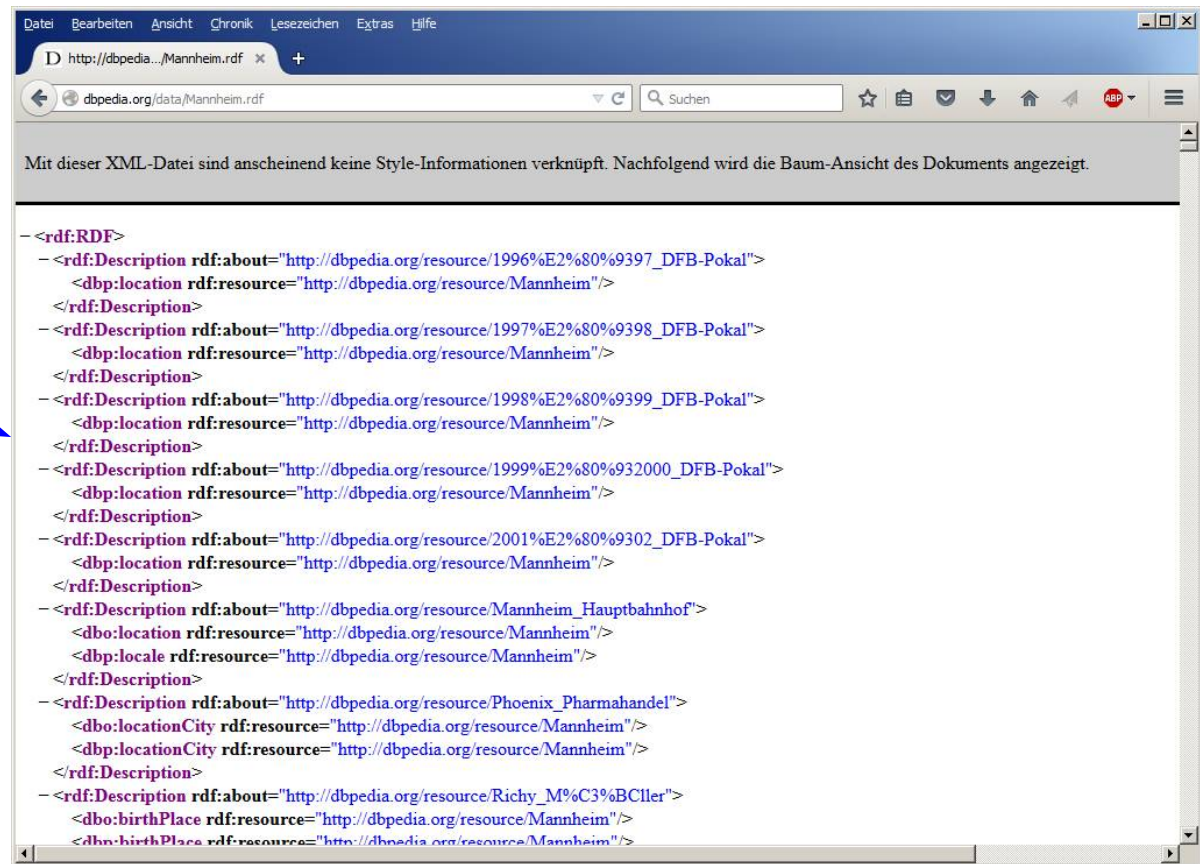


The screenshot shows a web browser window displaying the dbpedia.org page for Mannheim. The page title is "About: Mannheim". The browser's address bar shows "dbpedia.org/page/Mannheim". The page content is a table of properties and their values for Mannheim.

dbo:administrativeDistrict	■ dbr:Karlsruhe
dbo:areaCode	■ 0621
dbo:areaTotal	■ 144960000.000000 (xsd:double)
dbo:country	■ dbr:Germany
dbo:elevation	■ 97.000000 (xsd:double)
dbo:federalState	■ dbr:Baden-Württemberg
dbo:leaderParty	■ dbr:Social_Democratic_Party_of_Germany
dbo:leaderTitle	■ Lord Mayor
dbo:populationAsOf	■ 2008-12-31 (xsd:date)
dbo:populationMetro	■ 2362046 (xsd:integer)
dbo:populationTotal	■ 311142 (xsd:integer)
dbo:postalCode	■ 68001–68309
dbo:thumbnail	■ http://commons.wikimedia.org/wiki/Special:FilePath/SchlossMannheimEHof.jpg?width=300
dbo:wikiPageExternalLink	■ http://www.mannheim.de/ ■ http://home.mannheim.army.mil/sites/local/ ■ http://www.bertha-benz.de/indexen.php?inhalt=home/ ■ http://www.mann-hs.eu.dodea.edu/ ■ http://www.mann-ms.eu.dodea.edu/ ■ http://www.stadtpark-mannheim.de/ ■ http://www.vn.de/ ■ http://www.wishyouwerehere.de/
dbo:wikiPageID	■ 99627 (xsd:integer)
dbo:wikiPageRevisionID	■ 640007849 (xsd:integer)
dbp:align	■ center
dbp:aprHighC	■ 16.200000 (xsd:double)
dbp:aprLowC	■ 5 (xsd:integer)
dbp:aprMeanC	■ 10.700000 (xsd:double)
dbp:aprPrecipitationMm	■ 49.300000 (xsd:double)
dbp:aprRecordHighC	■ 28.100000 (xsd:double)
dbp:aprRecordLowC	■ -6.400000 (xsd:double)
dbp:aprSun	■ 180.200000 (xsd:double)
dbp:art	■ City

Why “Linked” Open Data?

- Example:
 - `<#Heiko> :worksIn <http://dbpedia.org/resource/Mannheim> .`



HTML Links vs. Links in Linked Open Data

- Compare

Heiko works in `Mannheim.`

to

`:Heiko :worksIn <http://dbpedia.org/resource/Mannheim> .`

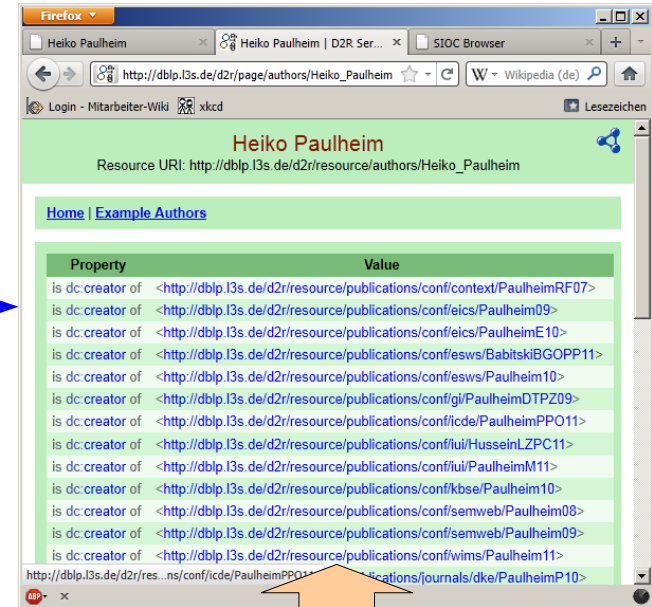
- Observation:

- Links in Linked Open Data are always *explicitly* typed
- The semantics of the link is thus interpretable
 - given that the predicate is defined in a schema

Links in Linked Open Data

- Important special case: owl:sameAs*

```
:Heiko  
  owl:sameAs  
    <http://dblp.l3s.de/d2r/page/  
      authors/Heiko_Paulheim>
```



* We don't know OWL yet, never mind, we'll get to that...

Links in Linked Open Data

- Important special case: `owl:sameAs`*
- Links two *identical* resources
 - This is required due to the non-unique naming assumption
- One of the most commonly misused concepts in the Semantic Web...
- Use:
 - Two datasets with information about the same person
- Abuse:
 - A dataset with information about a person and the person's homepage
 - The Starbucks in O7 and the company Starbucks
 - The state and the city of Hamburg
 - The parliament as an institution and the parliament as a building

* We don't know OWL yet, never mind, we'll get to that...

Links in Linked Open Data

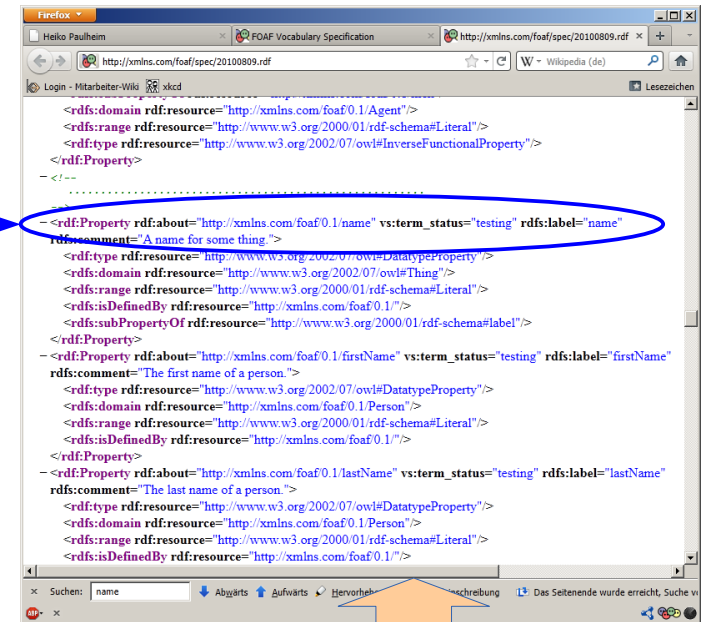
- Alternatives to abusing `owl:sameAs`*
 - General link to other resources
`rdfs:seeAlso`
 - Link to (HTML) homepage:
e.g., `foaf:homepage`

* We don't know OWL yet, never mind, we'll get to that...

Linking to a Schema

- Another important special case:
 - linking to a schema
 - luckily, everything is identified by a URI (also properties and classes)

:Heiko
<http://xmlns.com/foaf/0.1/name>
"Heiko Paulheim" .

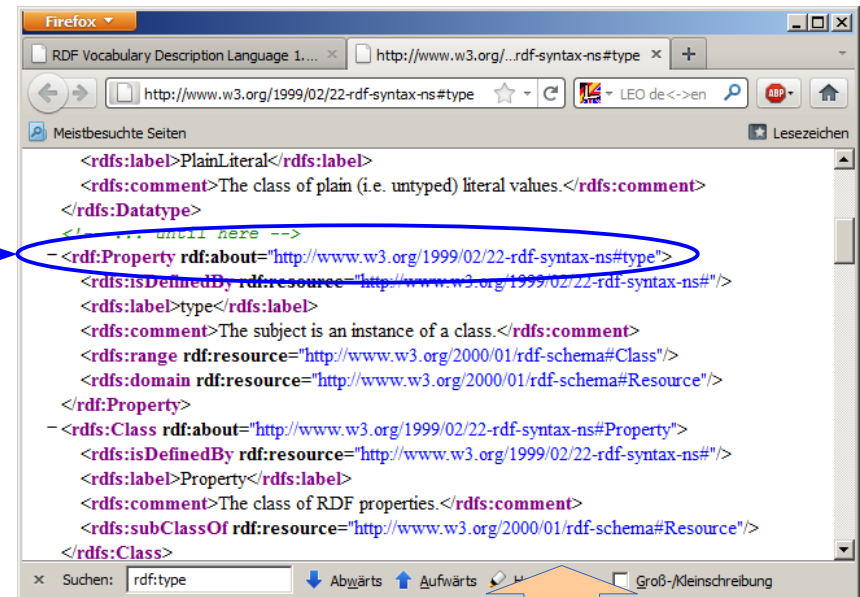


Linking to a Schema

- btw: this also works for “built in” schemas

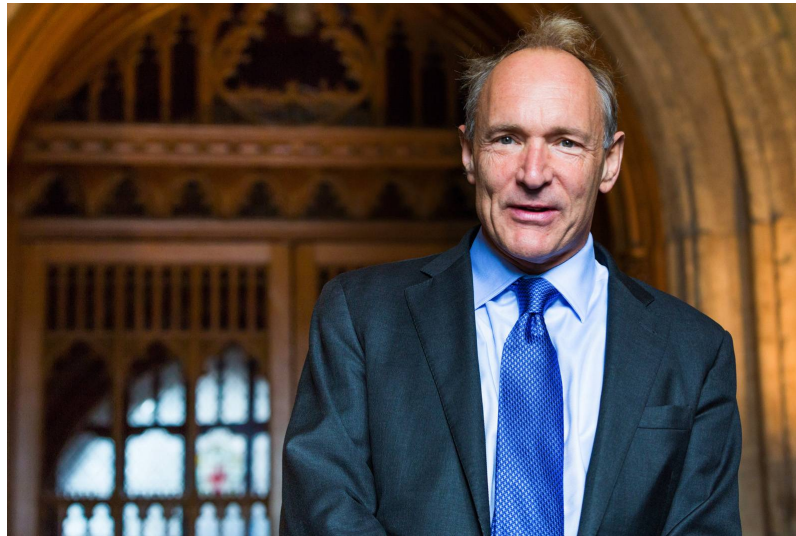
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

:Heiko **rdf:type** :Person .
...



Four Principles of Linked Open Data

- The four Principles by Tim Berners-Lee (2006)
 - 1) Use URIs to identify things
 - 2) Use derefencable URIs
 - 3) Provide useful information upon derefencable URIs, use standards
 - 4) Add links to other datasets

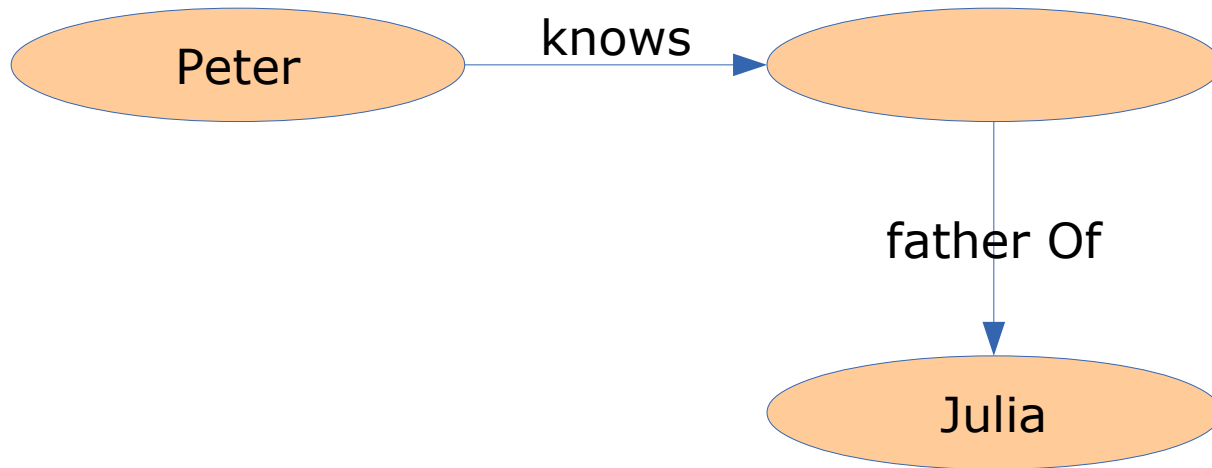


What Data to Serve at a URI?

- Basic principle: provide a complete *RDF molecule* at the URI
- Definition of a complete RDF molecule:
 - All triples that have the URI as a subject or an object
 - Every blank node is connected by at least two predicates

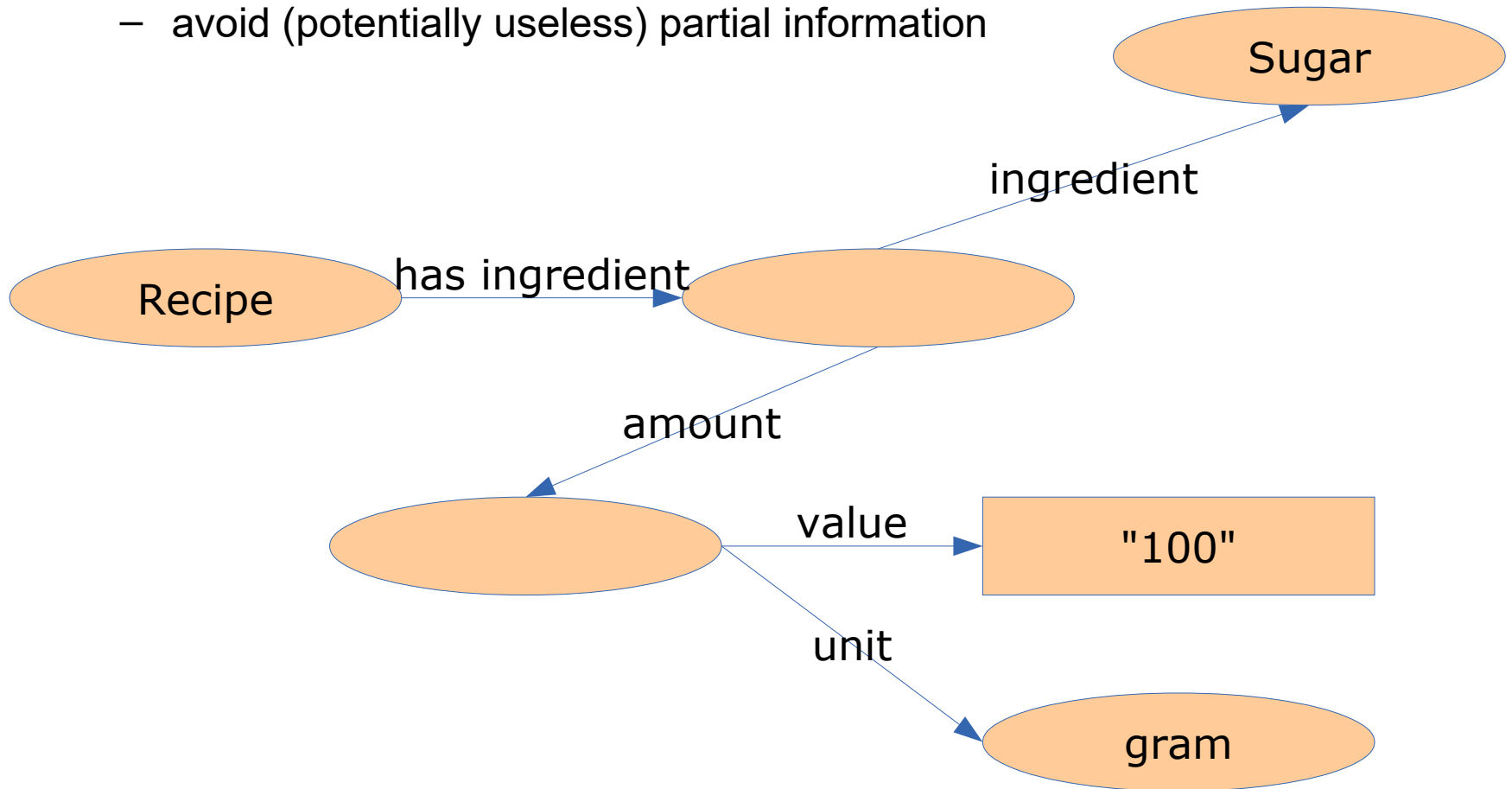
RDF Molecules

- Avoid dead ends in browsing



RDF Molecules

- Recap: Blank Nodes for multi-valued predicates
 - avoid (potentially useless) partial information

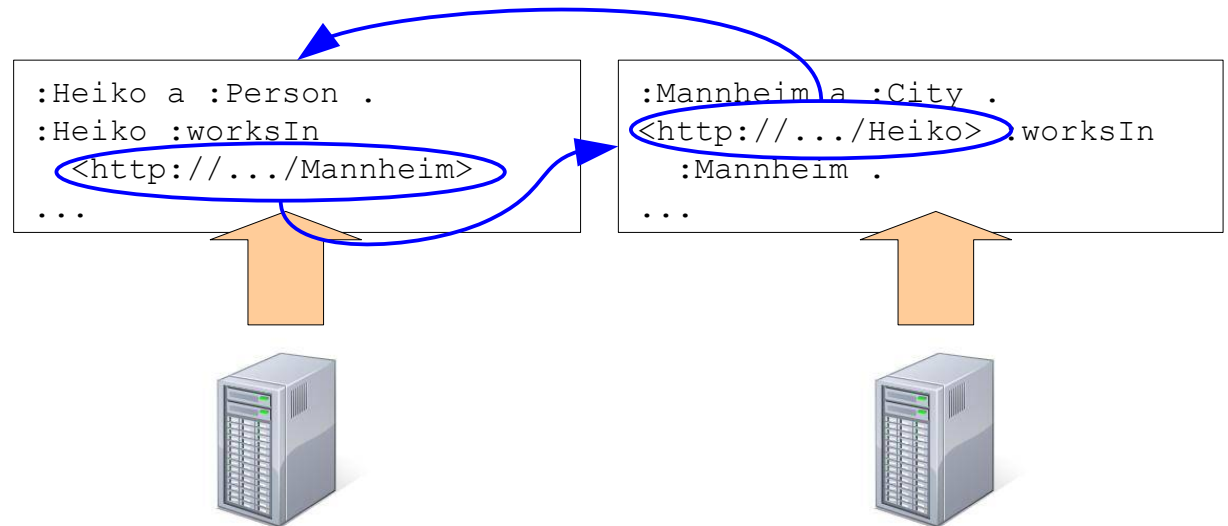


RDF Molecules: Theory and Practice

- Definition of a complete RDF molecule:
 - All triples that have the URI as a subject or an object
 - Every blank node is connected by at least two predicates
- Consequences:
 - Triples are duplicated (in the subject's and the object's molecule)
 - redundancy, depending on serving strategy
 - Molecules can become very big

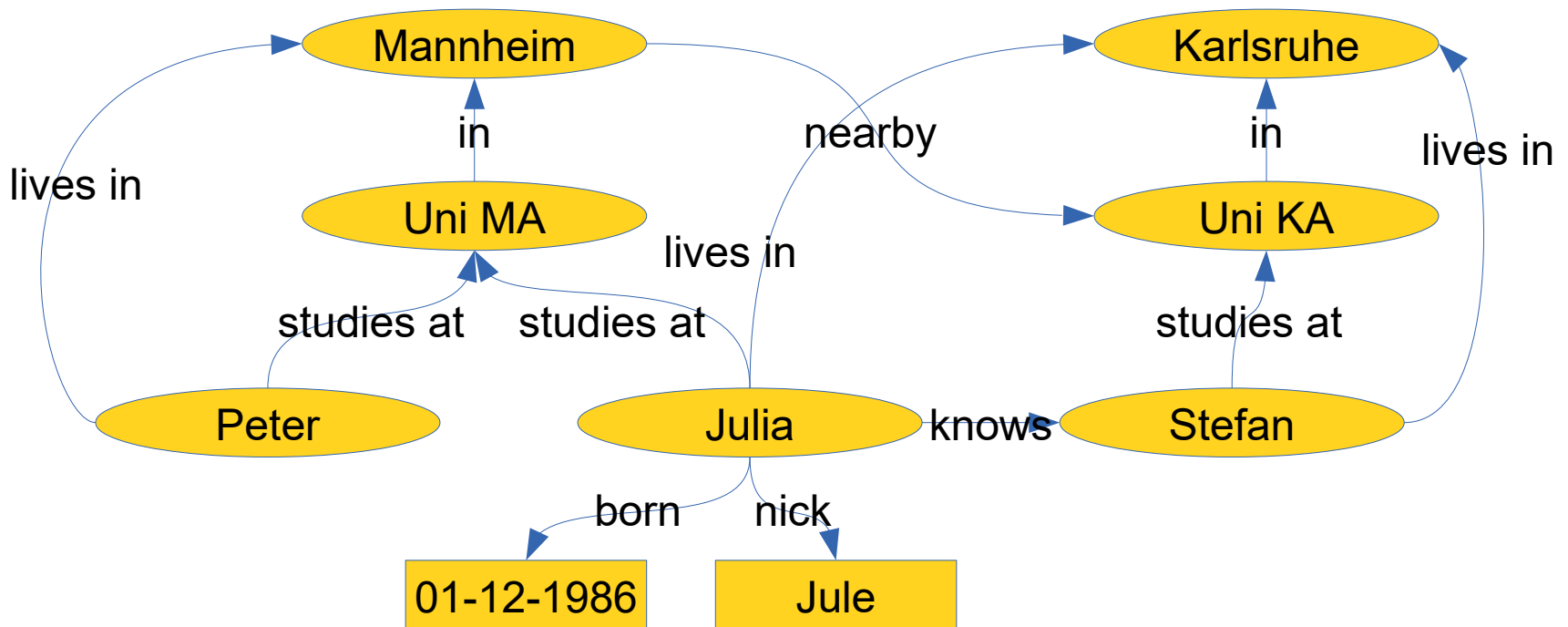
RDF Molecules: Theory and Practice

- In theory, all triples have to be served
- Pragmatic approach:
 - Which information is interesting for a user?
 - For a person: the city of residence
 - but for a city: all persons who reside here?



RDF Molecules: Theory and Practice

- Example Graph



The Five Star Schema

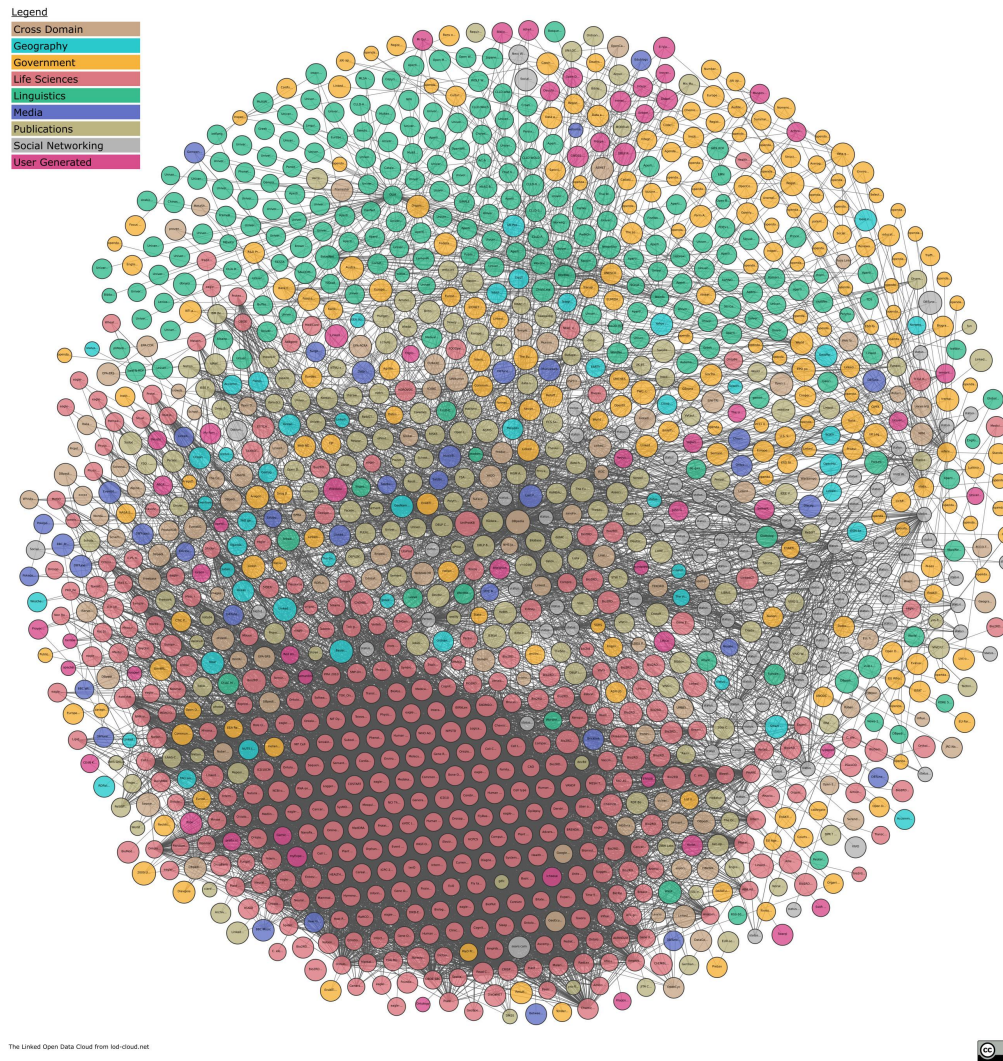
- Five Star Scheme (Tim Berners-Lee, 2010)
 - * Available on the web with an open license
 - ** Available as machine-readable, structured data
 - *** like ** plus using a non-proprietary format
 - **** like*** plus using open standards by the W3C
 - ***** like **** plus links to other datasets



Linked Open Data Best Practices

- as defined by Heath and Bizer, 2011
 - 1) Provide dereferencable URIs
 - 2) Set RDF links pointing at other data sources
 - 3) Use terms from widely deployed vocabularies
 - 4) Make proprietary vocabulary terms dereferencable
 - 5) Map proprietary vocabulary terms to other vocabularies
 - 6) Provide provenance metadata
 - 7) Provide licensing metadata
 - 8) Provide data-set-level metadata
 - 9) Refer to additional access methods

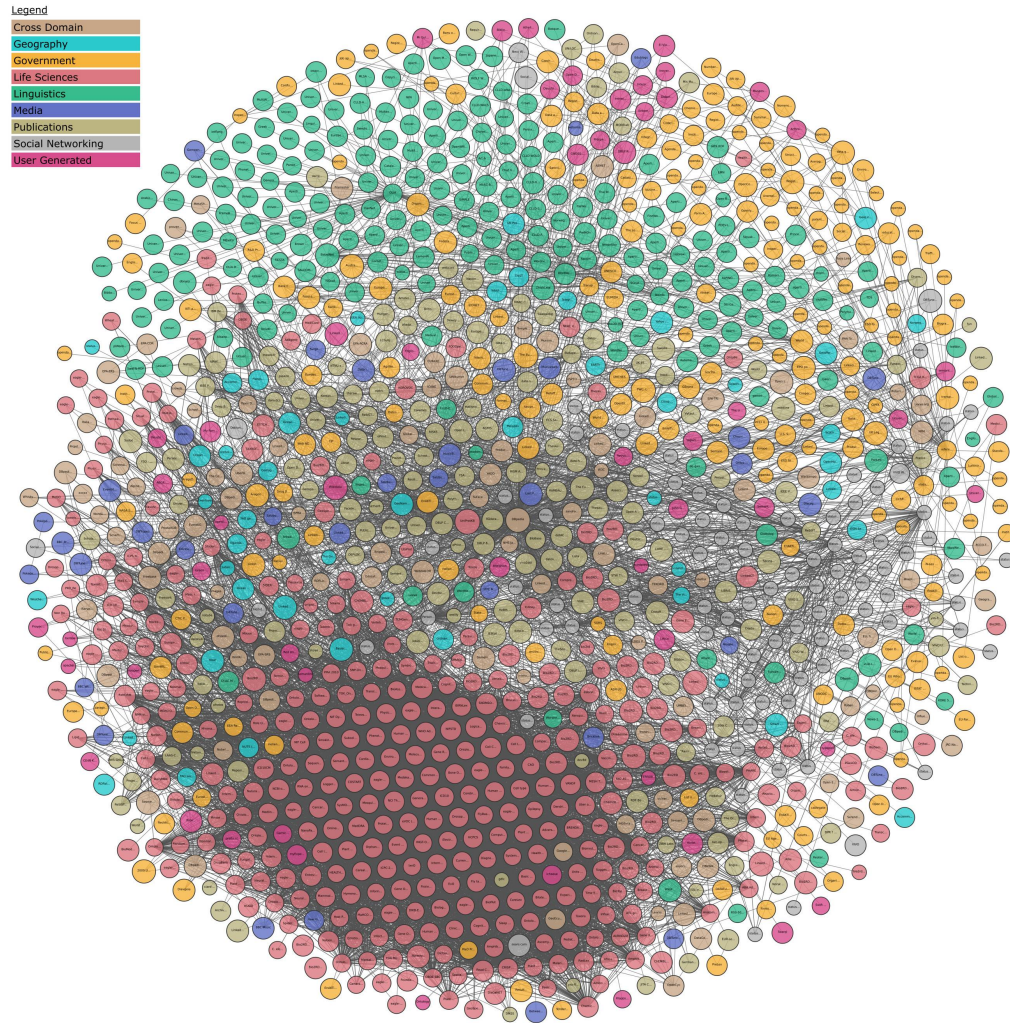
The Linked Open Data Cloud



<http://lod-cloud.net/>

What is the Linked Open Data Cloud?

- Viewpoint 1: a set of interconnected knowledge graphs
 - People have published ~1,000 knowledge graphs
 - They are linked to one another
- Viewpoint 2: one huge knowledge graph
 - In its entirety, the LOD cloud forms a large knowledge graph
 - This graph is very heterogeneous (i.e., uses different schemata)



The Linked Open Data Cloud from lod-cloud.net



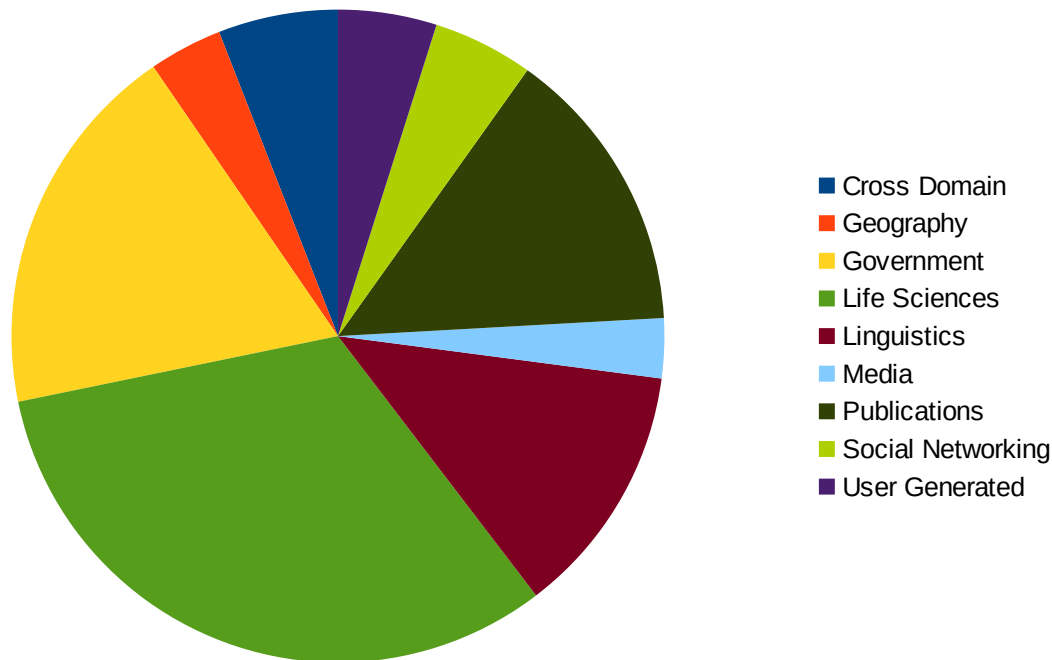
The Linked Open Data Cloud

- In numbers:
 - >1,250 Data sets
 - Several billion triples
 - Several million interlinks
- Topical domains:
 - Government
 - Publications
 - Life sciences
 - User-generated content
 - Cross-domain
 - Media
 - Geographic
 - Social web

<http://lod-cloud.net/>

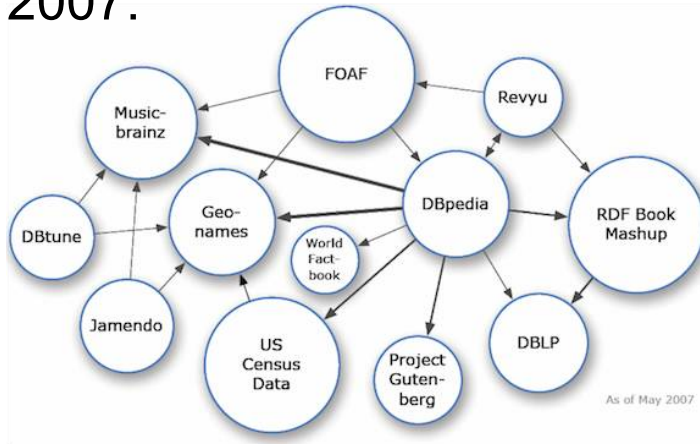
The Linked Open Data Cloud

- Domains by number of datasets in Linked Open Data
 - As of 2019
 - Classified based on data provider tags
 - More than half of the datasets are government and life sciences

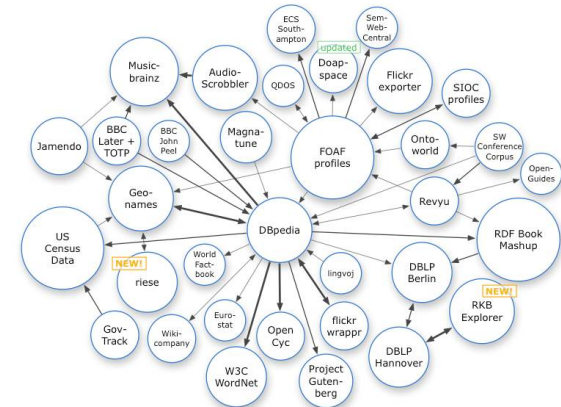


A Short History of Linked Open Data

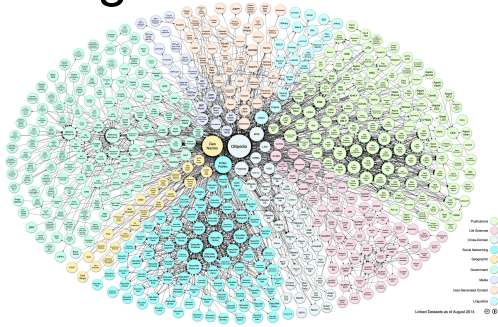
- May 2007:



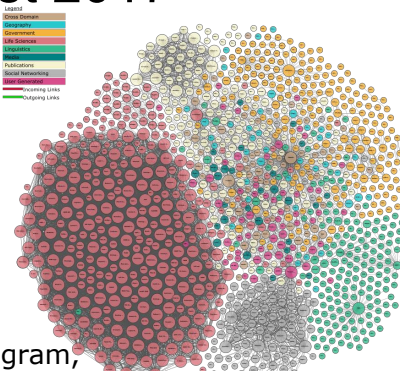
- March 2008:



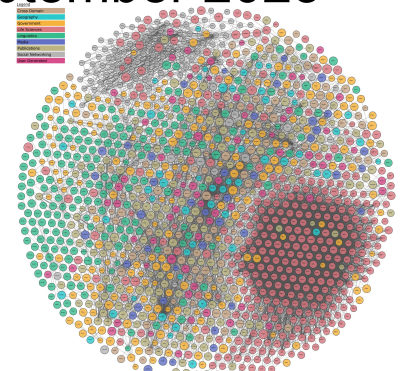
- August 2014



- August 2017



- September 2023



Linking Open Data cloud diagram,
by Richard Cyganiak and Anja Jentzsch. <http://lod-cloud.net/>

Examples: Government Data

The screenshot shows the data.gov website. At the top, there's a navigation bar with links like DATA, TOPICS, IMPACT, APPLICATIONS, DEVELOPERS, and CONTACT. Below this is a search bar and a filter section. The main content area displays "11,717 datasets found" and lists several datasets, including "Demographic Statistics By Zip Code" and "Popular Baby Names". A sidebar on the left shows filters for location and topics.

The screenshot shows the data.gov.uk website header. It includes the text "data.gov.uk | Find open data" and links for "Publish your data", "Documentation", and "Support". Below this is a blue banner with the text: "We've been improving data.gov.uk to help you find and use open government data. Discover what's changed and get in touch to give us your feedback. Don't show this message again".

Search results

The screenshot shows search results for "Organogram of Staff Roles & Salaries" on data.gov.uk. It includes a search bar, a filter section with "Filter by" and "Publisher", and a list of results. The top result is "Organogram of Staff Roles & Salaries" published by "Serious Fraud Office" and last updated on "18 October 2016". The description states: "Organogram (organisation chart) showing all staff roles. Names and salaries are also listed for the Senior Civil Servants. Organogram data is released by all central government departments and...".

The screenshot shows the EU Open Data Portal website. It features a navigation bar with links like Home, Data, Applications, Linked data, Visualisations, Developers' corner, and About. The main content area is titled "About linked data" and explains that linked data is a standard way to represent data on a wide range of topics. It also includes a section for "SPARQL" queries and a "Need help?" section with a "Contact us" button.

Linguistics Example: BabelNet

Keyboard

http://babelnet.org/rdf/keyboard_n_EN



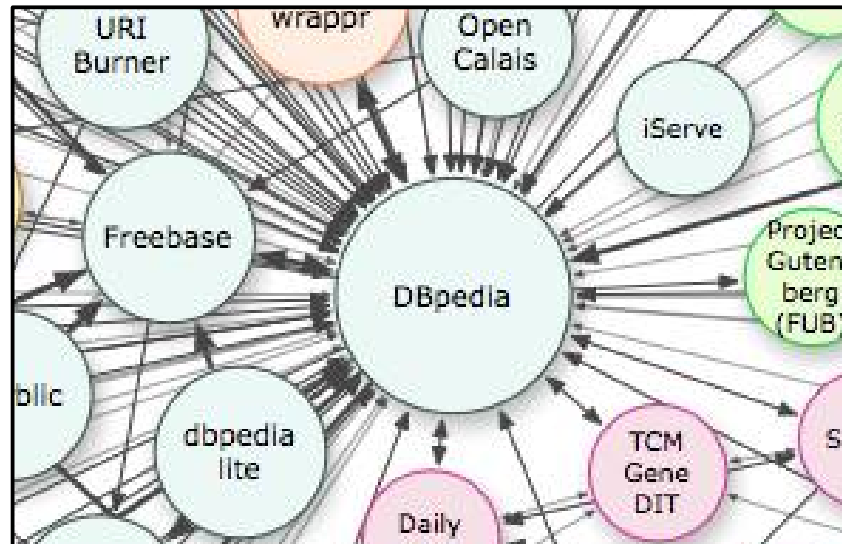
lemon: LexicalEntry

Property	Value
lemon:canonicalForm	bn: keyboard_n_EN/canonicalForm
Is lemon:entry of	bn: lexicon_EN
rdfs:label	<ul style="list-style-type: none">• keyboard• keyboard_(computer)• keyboard_(computing)• keyboard_(instrument)• keyboard_(music)• keyboard_(musical_instrument)• keyboard_(typing)
lemon:language	EN
lexinfo:partOfSpeech	lexinfo: noun
lemon:sense	11

[As Turtle](#) | [As RDF/XML](#) | [As N-Triple](#)

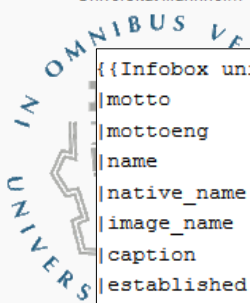
Cross-Domain Example: DBpedia

- General knowledge on almost five million entities
- Hundreds of millions of triples
- Linked to ~100 other datasets
 - the most interlinked dataset



<http://lod-cloud.net/>

DBpedia: How It Is built

University of Mannheim	
Universität Mannheim	
	
Motto	Infobox university
	motto = ''In Omnibus Veritas Suprema Lex Esto'' ([[Latin]])
	mottoeng = Truth in everything should be the supreme law
	name =University of Mannheim
	native_name =Universität Mannheim
	image_name =Uni_Mannheim_Siegel.gif
	caption =[[Seal (emblem) Seal]] of the UMA
	established =1763: Theodoro Palatinae 1907: Handelshochschule
	type =[[Public University Public]]
	endowment =€115 [[million]]
	academic_staff =800 (full time)
	administrative_staff = 550 (full time)
	Schools =5
	rector =[[Ernst-Ludwig von Thadden]]
	chancellor =[[Susann-Annette Storm]]
	students =12,151 <small>''(HWS 2013/14)''</small><ref name="univstatistik">{{Coord 49.4832 8.4647 region:DE-BW type:edu source=Studierendenstatistik_hws13.pdf title= Studierendenstatistik der Universität Mannheim date=2013-12-13 url=http://www.uni-mannheim.de/statistik/ format=PDF author=Studienamt der Universität Mannheim accessdate=2014-01-06 archiveurl=http://www.webcitation.org/103333333 archivedate=2014-01-06}}</ref>
	undergrad =6,915<ref name="uni-mannheim.de"/>
	postgrad =4,965<ref name="uni-mannheim.de"/>
	doctoral =249<ref name="uni-mannheim.de"/>
	profess =
	city =[[Mannheim]]
	state =[[Baden-Württemberg]]
	country =[[Germany]]
	coor = {{Coord 49.4832 8.4647 region:DE-BW type:edu source=Studierendenstatistik_hws13.pdf title= Studierendenstatistik der Universität Mannheim date=2013-12-13 url=http://www.uni-mannheim.de/statistik/ format=PDF author=Studienamt der Universität Mannheim accessdate=2014-01-06 archiveurl=http://www.webcitation.org/103333333 archivedate=2014-01-06}}
Undergraduates 6,915 ^[1]	
Postgraduates 4,965 ^[1]	
Doctoral students 249 ^[1]	

```

<rdf:RDF>
  <rdf:Description rdf:about="http://dbpedia.org/resource/Mannheim_Centre_for_European_Social_Research">
    <dbpedia.org/resource/University_of_Mannheim"/>
    <dbpedia.org/resource/Wolfgang_Franz">
    <dbpedia.org/resource/University_of_Mannheim"/>
    <dbpedia.org/resource/University_of_Mannheim"/>
    <dbpedia.org/resource/University_of_Mannheim"/>
    <dbpedia.org/resource/Heinz_K%C3%B6nig">
    <dbpedia.org/resource/University_of_Mannheim"/>
    <dbpedia.org/resource/Roman_Inderst">
    <dbpedia.org/resource/University_of_Mannheim"/>
    <dbpedia.org/resource/University_of_Mannheim"/>
    <dbpedia.org/resource/Claus_E._Heinrich">
    <dbpedia.org/resource/University_of_Mannheim"/>
    <dbpedia.org/resource/University_of_Mannheim"/>
    <dbpedia.org/resource/University_of_Mannheim"/>
    <dbpedia.org/resource/Susann-Annette_Storm">
    <dbpedia.org/resource/University_of_Mannheim"/>
    <dbpedia.org/resource/Bruno_Sälzer">
    <dbpedia.org/resource/University_of_Mannheim"/>
    <dbpedia.org/resource/Heinz_König">
    <dbo:award rdf:resource="http://dbpedia.org/resource/University_of_Mannheim"/>
  </rdf:Description>

```

DBpedia: Further Sources

Coordinates:  49°29'20"N 8°28'9"E

Climate [\[edit \]](#)

Climate data for Mannheim, Germany for 1981–2010 (Source: DWD) [hide]													
Month	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Year
Record high °C (°F)	16.4 (61.5)	20.2 (68.4)	26.1 (79)	28.1 (82.6)	32.2 (90)	36.6 (97.9)	39.0 (102.2)	39.8 (103.6)	32.6 (90.7)	28.2 (82.8)	19.7 (67.5)	16.5 (61.7)	39.8 (103.6)
Average high °C (°F)	4.7 (40.5)	6.7 (44.1)	11.6 (52.9)	16.2 (61.2)	20.6 (69.1)	23.7 (74.7)	26.1 (79)	25.9 (78.6)	21.2 (70.2)	15.3 (59.5)	8.9 (48)	5.3 (41.5)	15.50 (59.9)
Daily mean °C (°F)	1.8 (35.2)	2.8 (37)	6.7 (44.1)	10.7 (51.3)	15.2 (59.4)	18.2 (64.8)	20.3 (68.5)	19.9 (67.8)	15.6 (60.1)	10.7 (51.3)	5.7 (42.3)	2.8 (37)	10.85 (51.53)
Average low °C (°F)	−1.3 (29.7)	−0.8 (30.6)	2.3 (36.1)	5.0 (41)	9.4 (48.9)	12.4 (54.3)	14.5 (58.1)	14.2 (57.6)	10.6 (51.1)	6.7 (44.1)	2.5 (36.5)	−0.0 (32)	6.28 (43.3)
Record low °C (°F)	−18.7 (−1.7)	−18.7 (−1.7)	−13.6 (7.5)	−6.4 (20.5)	−0.1 (31.8)	4.0 (39.2)	4.7 (40.5)	5.3 (41.5)	2.5 (36.5)	−5.0 (23)	−8.7 (16.3)	−18.3 (−0.9)	−18.7 (−1.7)
Average precipitation mm (inches)	40.9 (1.61)	43.1 (1.697)	50.8 (2)	49.3 (1.941)	72.5 (2.854)	66.6 (2.622)	76.0 (2.992)	57.7 (2.272)	54.1 (2.13)	56.4 (2.22)	53.5 (2.106)	54.1 (2.13)	675.0 (26.575)
Mean monthly sunshine hours	55.2	85.6	124.0	180.2	214.1	219.1	235.1	222.1	164.1	108.8	59.0	44.9	1,712.2

Source: Data derived from [Deutscher Wetterdienst](#)^[12]

Categories: [Cities in Baden-Württemberg](#) | [Mannheim](#) | [Historic Jewish communities](#) | [Karlsruhe \(region\)](#) | [Populated places on the Rhine](#) | [University towns in Germany](#) | [Planned capitals](#) | [History of the Palatinate \(region\)](#)

DBpedia: Contents

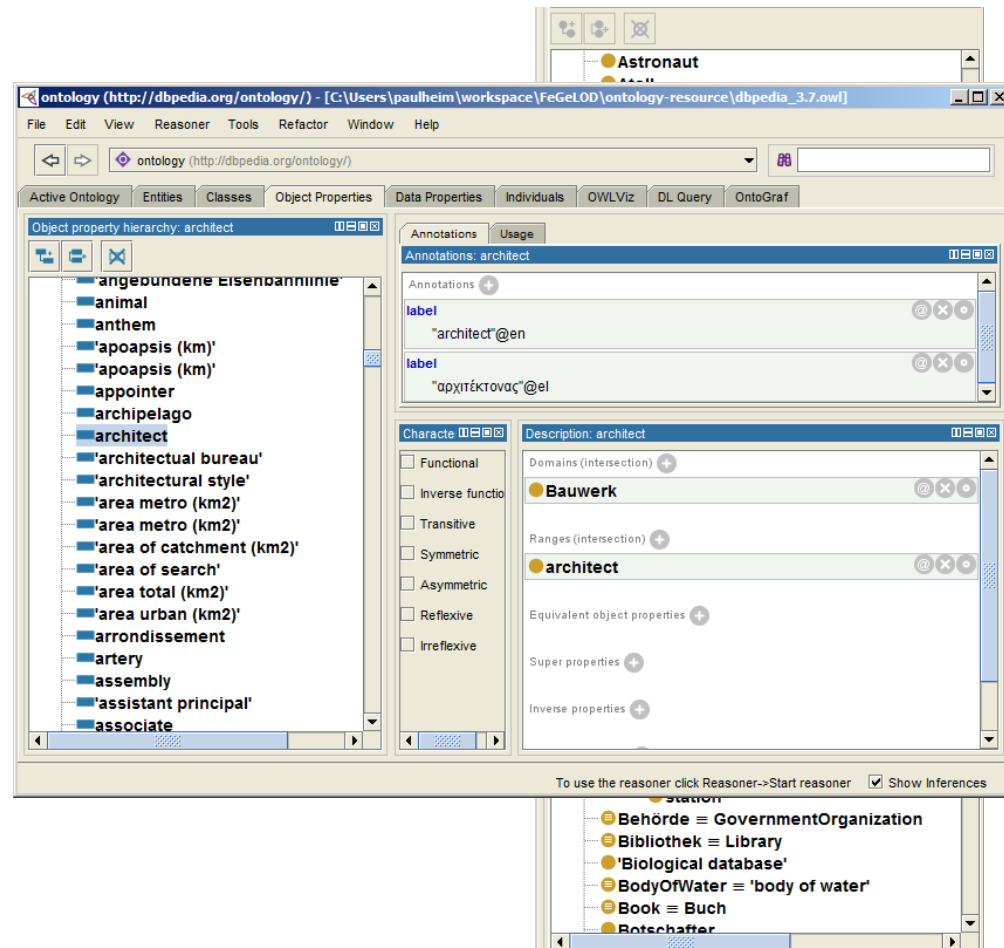
- Data from different infoboxes (extracted from multiple languages)
- Redirects and disambiguations
- External web links
- Abstracts in multiple languages
- Instance type information
 - DBpedia Ontology
 - YAGO*
 - schema.org*
 - DOLCE**
 - ...and others

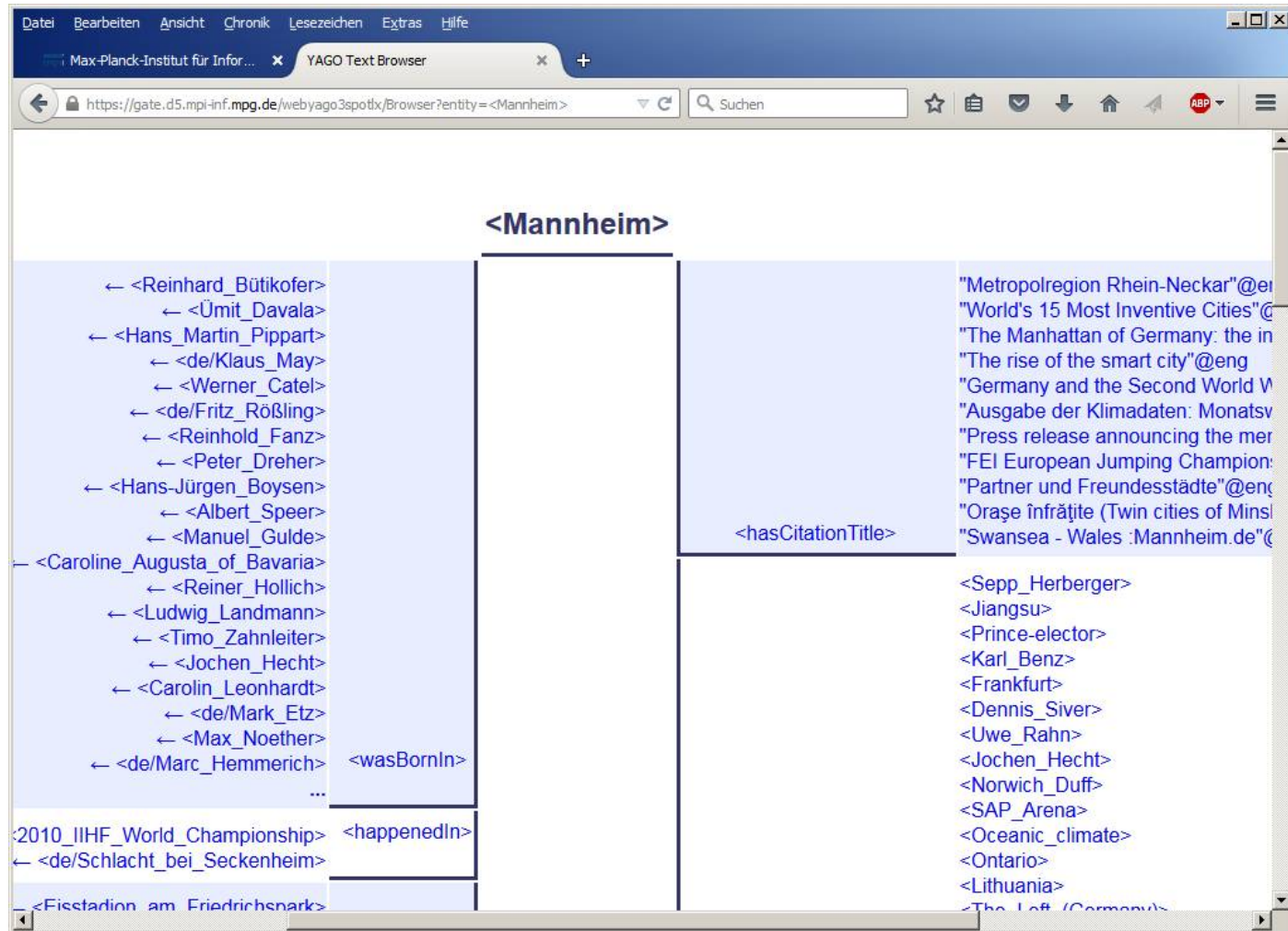
* later today

** in a few weeks

The DBpedia Ontology

- Classes:
 - ~1,800 classes
 - partial hierarchy
- Properties:
 - ~1,200 relations
 - many with domain/range
 - ~1,700 data properties
 - i.e., literal-valued
 - a bit of hierarchy





YAGO

- Also derived from Wikipedia
 - ~4.6M entities
 - ~26M statements
- Uses Wikipedia categories for typing
 - a class hierarchy of ~500,000 types
- Tries to capture time
 - i.e., statements that held true for a period of time
 - e.g., soccer players playing for teams
 - uses reification

Search: eng ▾

<id_1u5xrvs_1ul_zxcbb2>

<Miroslav_Klose> <playsFor> <FC_Bayern_Munich>	hasFactId		<extractionSource>	<http://en.wikipedia.org/wiki/Miroslav_Klose> → <http://en.wikipedia.org/wiki/Miroslav_Klose> →
			<occursUntil>	"2011-##-##"^^xsd:date →
			<occursSince>	"2007-##-##"^^xsd:date →

Wikidata

- Collaboratively edited knowledge base
- Size
 - ~15M instances
 - ~66M statements
- Ontology
 - ~23k classes
 - ~1.6k properties
- Special
 - provenance information
 - i.e., evidence: where did that statement come from?

Wikidata



Main page
Community portal
Project chat
Create a new item
Item by title
Recent changes
Random item
Help
Donate

Print/export

Create a book
Download as PDF
Printable version

Tools

What links here
Related changes
Special pages
Permanent link
Page information
Concept URI
Cite this page

Item [Discussion](#)

Trent Reznor (Q282722)

American musician

No aliases defined

[► In more languages](#)

[\[edit\]](#)

Statements

sex or gender

CHILD

male

[\[edit\]](#)

[▼ 4 references](#)

[\[edit\]](#)

imported from Swedish Wikipedia

[\[edit\]](#)

imported from Virtual International Authority File

[\[edit\]](#)

imported from Italian Wikipedia

[\[edit\]](#)

stated in Integrated Authority File

retrieved 27 April 2014

[\[add reference\]](#)

Wikipedia (33 entries) [\[edit\]](#)

[\[Collapse\]](#)

[be_x_old](#) Трэнт Рэзнар

[be](#) Трэнт Рэзнар

[bg](#) Трент Резнър

[cs](#) Trent Reznor

[da](#) Trent Reznor

[de](#) Trent Reznor

[en](#) Trent Reznor

[es](#) Trent Reznor

[et](#) Trent Reznor

[fa](#) ترنت رزرنر

[fi](#) Trent Reznor

[fr](#) Trent Reznor

[gl](#) Trent Reznor

[hu](#) Trent Reznor

[id](#) Trent Reznor

[is](#) Trent Reznor

[it](#) Trent Reznor

[ja](#) トレント・レズナー

[ka](#) ტრენტ რეზნორი

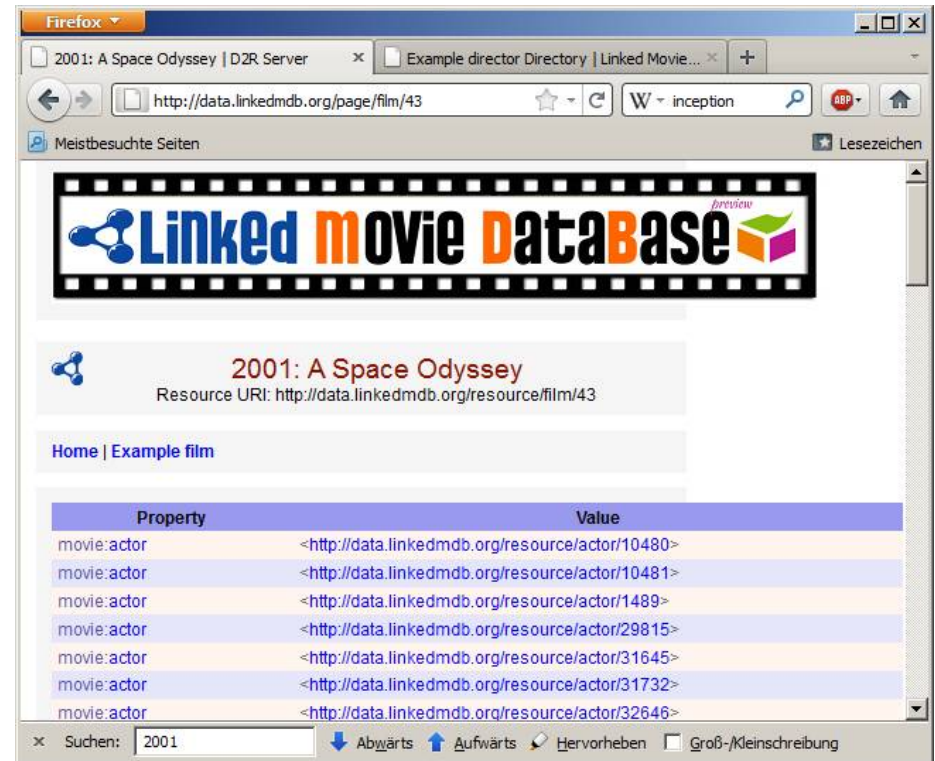
[ko](#) 트렌트 레즈너

[lv](#) Trents Reznors

[nl](#) Trent Reznor

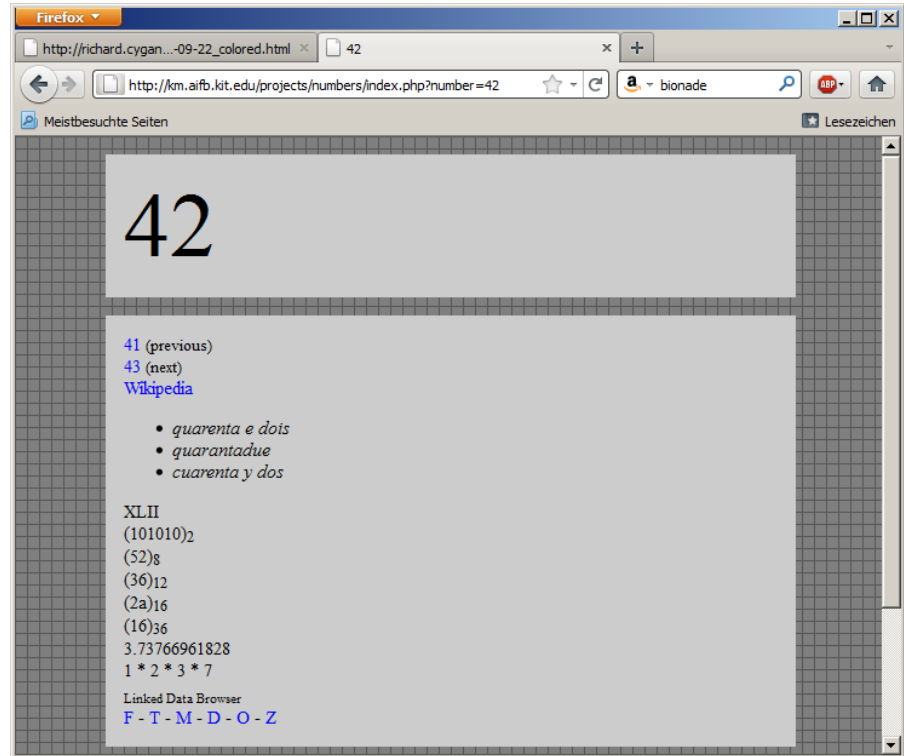
Further Example Datasets

- Linked Movie Database
 - Movies, actors, directors...
- MusicBrainz
 - Artists, albums, ...
- Open Library
 - books, authors, publishers
- DBLP
 - computer science publications



Further Example Datasets

- Linked Open Numbers
 - Numbers and their names in different languages
 - roman and arabic notations, binary, hex etc.



Vocabularies

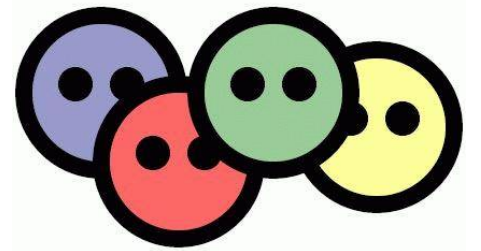
- Recap: LOD Best Practices, Principle 3:
 - Use terms from widely deployed vocabularies
- So, what are common widely deployed vocabularies?

Dublin Core

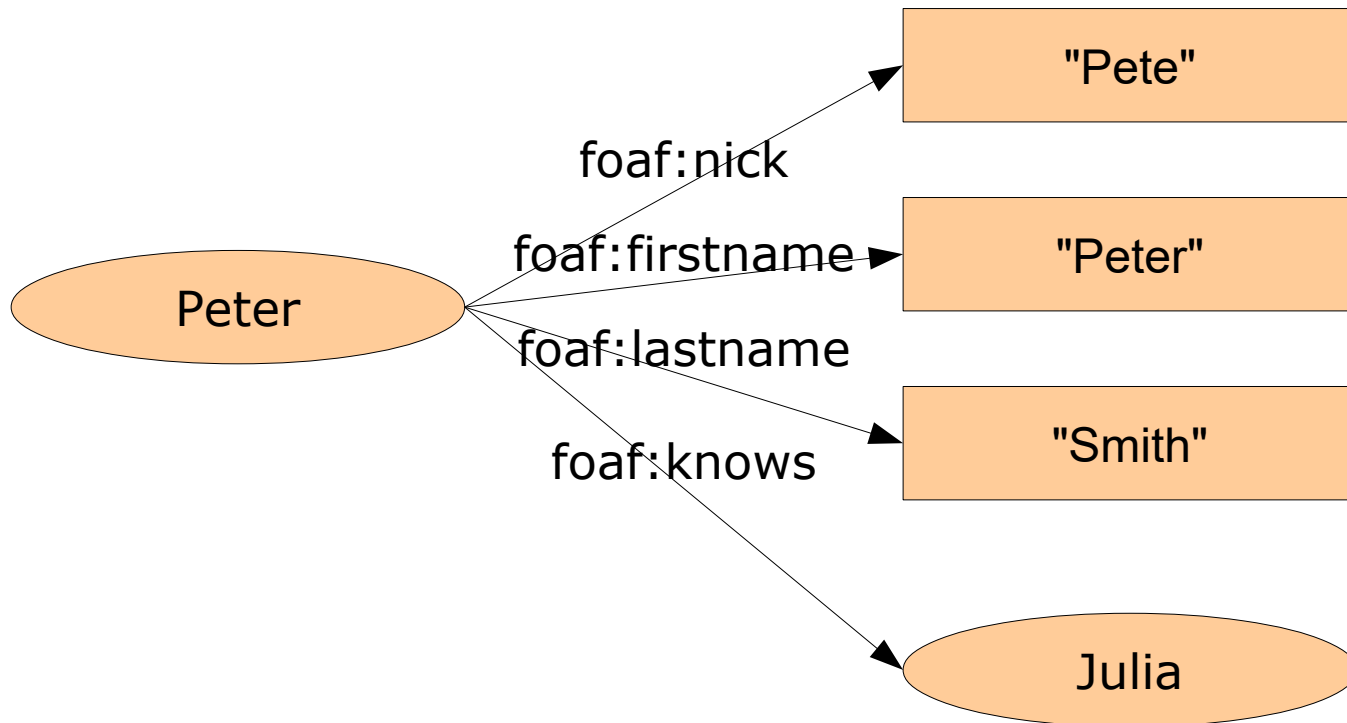
- We have already encountered this
- Usage: Metadata for resources and documents
- Namespace `http://purl.org/dc/elements/1.1/`
- Common prefix: `dc`
- defines properties, e.g.,
 - creator
 - subject
 - date
- Resources: DCMI Type Vocabulary:
 - Text
 - Image
 - Software
 - ...



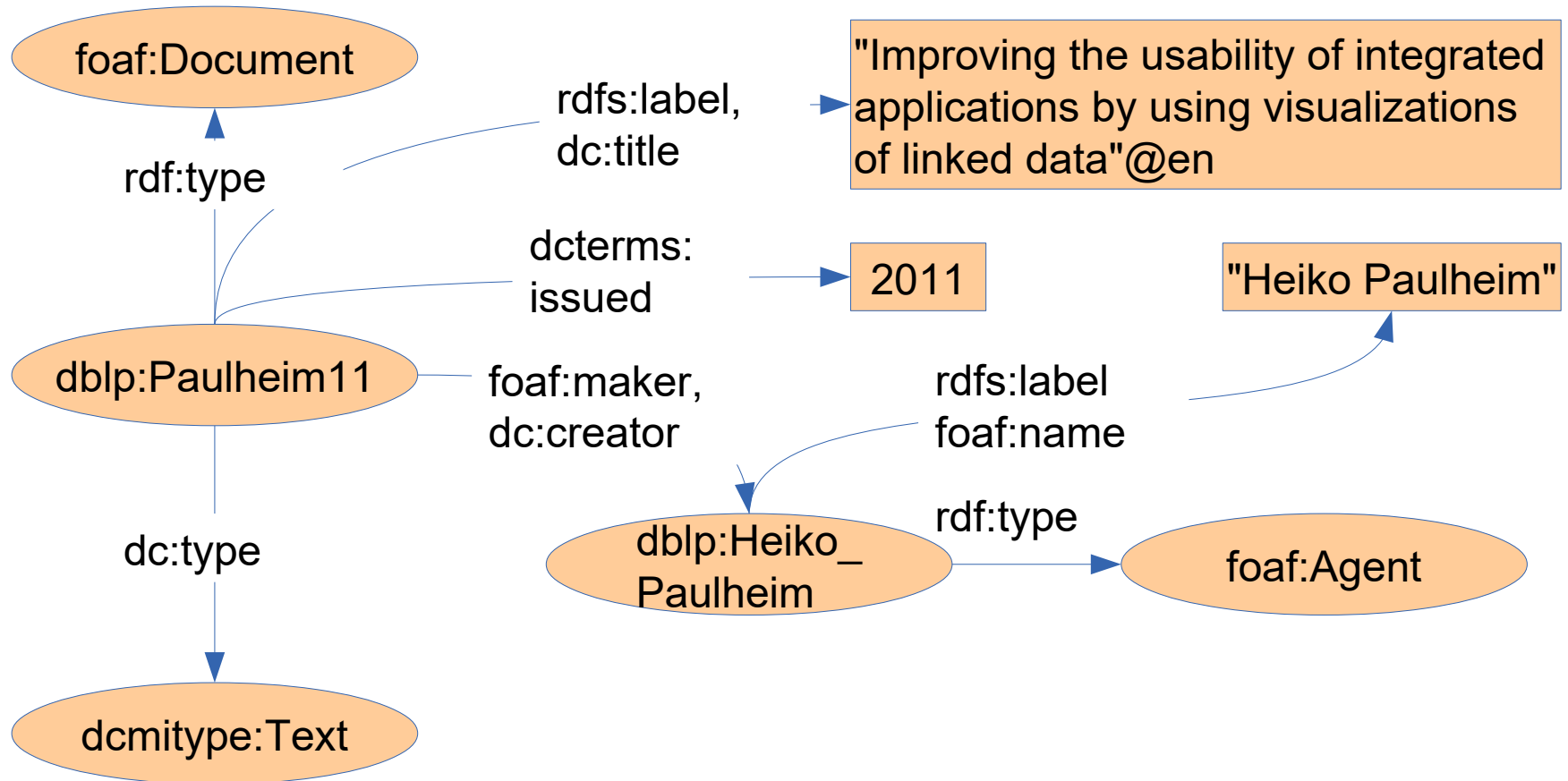
FOAF (Friend of a Friend)

- Persons and their relations
 - Created for personal home pages
 - but used widely beyond that
 - Namespace `http://xmlns.com/foaf/0.1/`
 - Common prefix: `foaf:`
- 
- Important classes
 - Person
 - Group
 - Organization
 - Project
 - ...
 - Important properties
 - name, firstName, lastName
 - phone, mbox, homepage
 - knows
 - currentProject, pastProject
 - ...

FOAF (Friend of a Friend)



DBLP: Combining FOAF and DC



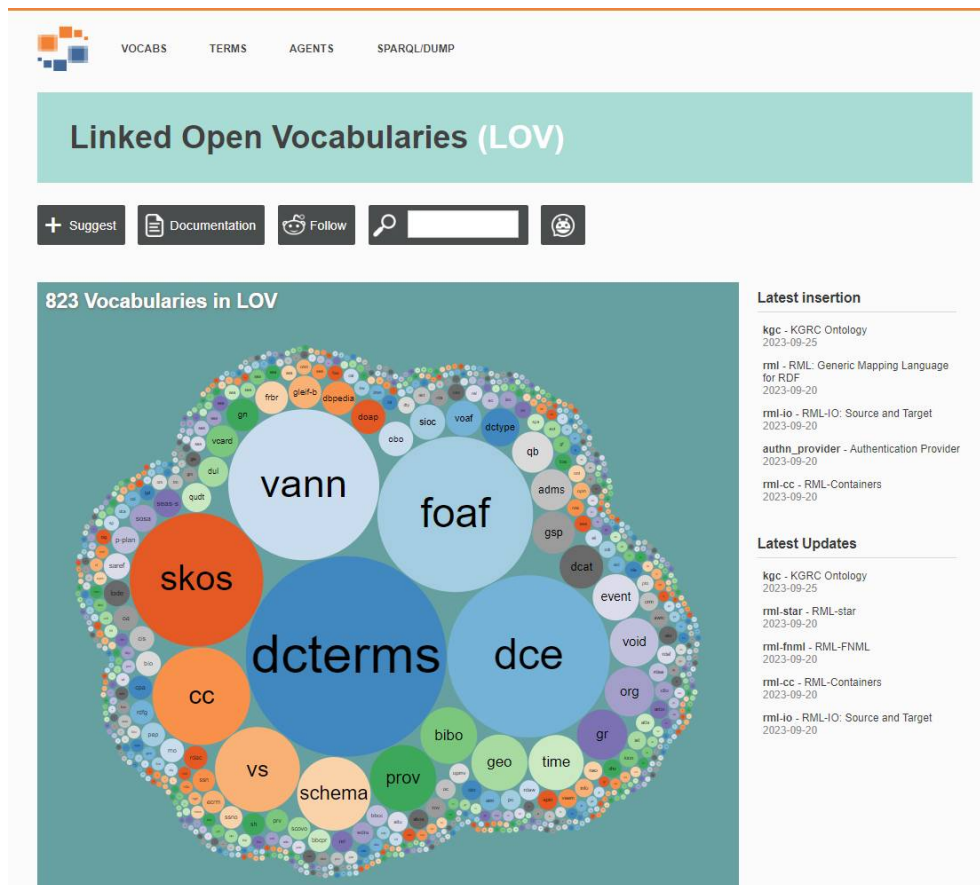
WGS 84

- Encodes geographic data
 - World Geodetic System 1984
 - 3D reference model
 - Namespace `http://www.w3.org/2003/01/geo/wgs84_pos#`
 - Common prefix: `geo:`
-
- Classes:
 - SpatialThing
 - Point
 - Properties:
 - latitude
 - longitude
 - altitude
 - location



Vocabularies

- Where to search for vocabulary terms?
 - One possibility: <https://lov.linkeddata.es/dataset/lov/>



Vocabularies

- Where to search for prefix definitions?
 - One possibility: <http://prefix.cc/>

The diagram illustrates the process of finding the RDFS vocabulary definition. It starts with the **prefix.cc** website, which is described as a "namespace lookup for RDF developers". A search box on the site contains the text "rdfs", and a "look up" button is next to it. Below the search box, there are examples of namespaces: "foaf", "foaf:knows", "dc:foaf", "rdfs", "dc:foaf", "geo", "sparql", and "http://xmlns.com/foaf/0.1/name". An arrow points from the search box to the URL <http://www.w3.org/2000/01/rdf-schema#>, which is also labeled "rdfs" and "Add alternative URI". Another arrow points from this URL to a browser window showing the RDFS vocabulary definition. The browser window has a address bar with the URL "w3.org/2000/01/rdf-schema#" and a content area displaying the following RDF code:

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .

<http://www.w3.org/2000/01/rdf-schema#> a owl:Ontology ;
    dc:title "The RDF Schema vocabulary (RDFS)" .

rdfs:Resource a rdfs:Class ;
    rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
    rdfs:label "Resource" ;
    rdfs:comment "The class resource, everything." .

rdfs:Class a rdfs:Class ;
    rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
    rdfs:label "Class" ;
    rdfs:comment "The class of classes." ;
    rdfs:subClassOf rdfs:Resource .

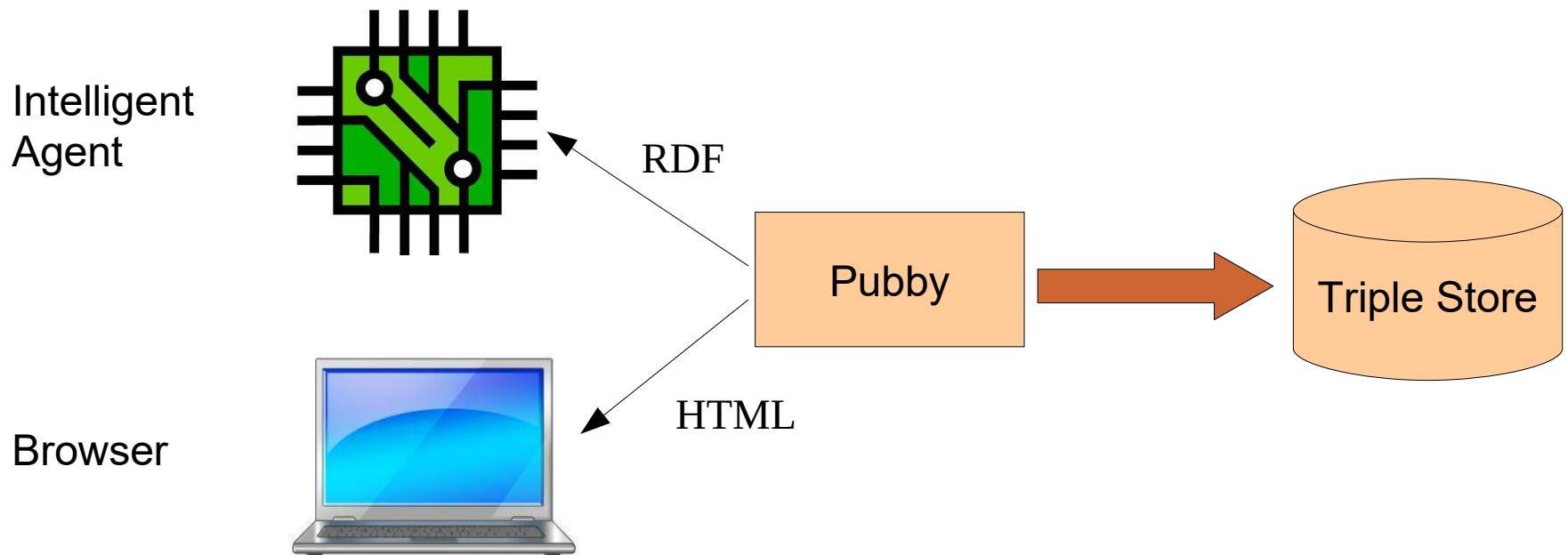
rdfs:subClassOf a rdf:Property ;
    rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
    rdfs:label "subClassOf" ;
    rdfs:comment "The subject is a subclass of a class." ;
    rdfs:range rdfs:Class ;
    rdfs:domain rdfs:Class .
```

Publishing Linked Open Data

- Possible variants
 - hand coded
 - from triple stores
 - from relational databases

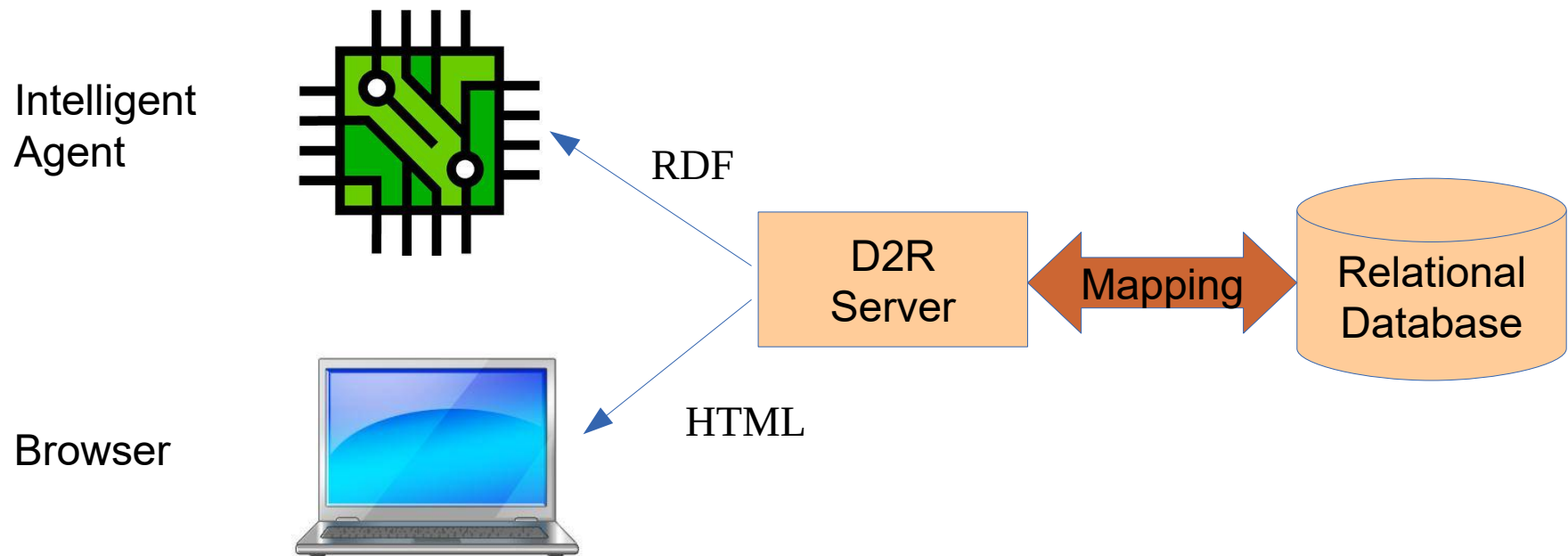
Linked Data from Triple Stores

- Triple Store: RDF storage engine
 - e.g., Virtuoso
- Pubby: Front end for triple stores
- Supports content negotiation etc.



Knowledge Graphs from Databases

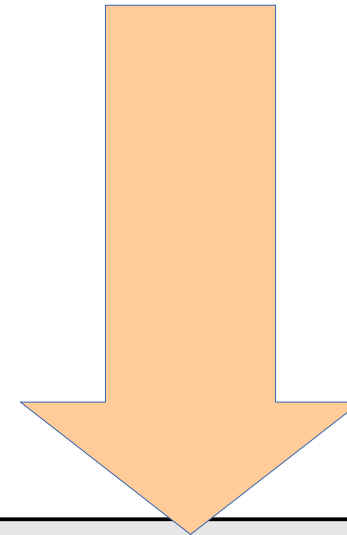
- D2R: Linked Open Data interface on relational databases
 - e.g., MySQL



Knowledge Graphs from Databases

ID (int)	name (text)	location (int)
1327890123	"Heiko"	"Mannheim"
...		

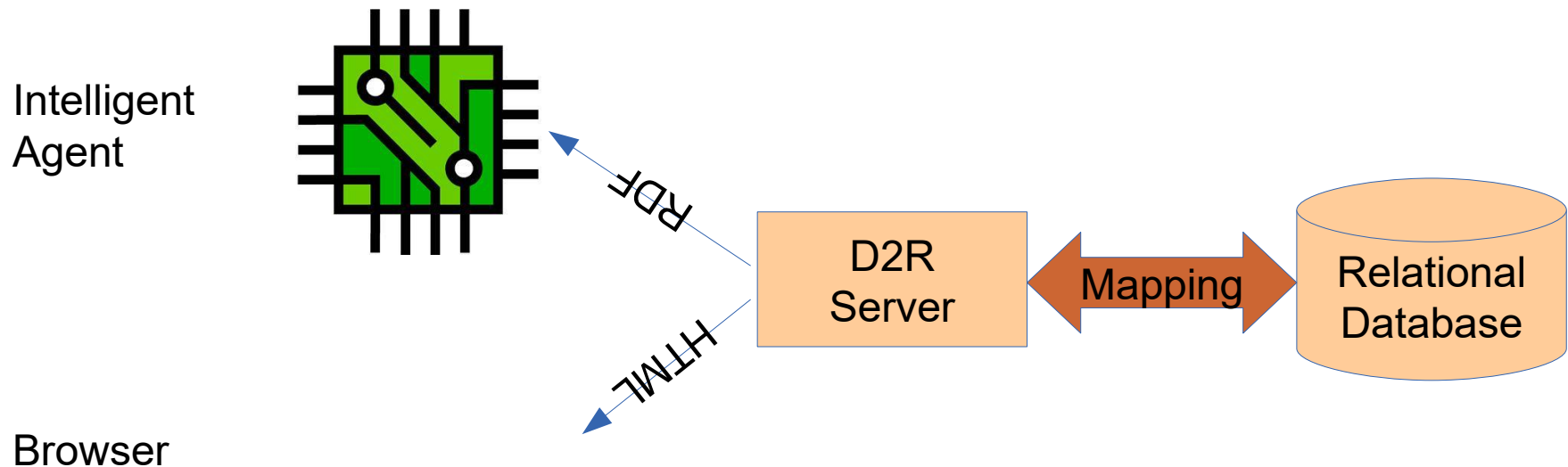
```
map:Person a d2rq:ClassMap;  
  d2rq:dataStorage map:Database1.  
  d2rq:class foaf:Person;  
  d2rq:uriPattern "http://foo.bar/p@@Person.ID@";  
  .  
map:personName a d2rq:PropertyBridge;  
  d2rq:belongsToClassMap map:Person;  
  d2rq:property foaf:name;  
  d2rq:column "Person.name";  
  d2rq:datatype xsd:string;  
  .  
map:location a d2rq:PropertyBridge;  
  d2rq:belongsToClassMap map:Person;  
  d2rq:property foaf:basedNear;  
  d2rq:column "Person.location";  
  d2rq:datatype xsd:string;  
  .
```



```
<http://foo.bar/p1327890123> a foaf:Person .  
<http://foo.bar/p1327890123> foaf:name "Heiko" .  
<http://foo.bar/p1327890123> foaf:basedNear "Mannheim" .
```

Knowledge Graphs from Databases

- Note:
 - In this case, the knowledge graph does not *replicate* the data
 - It is only a “virtual” knowledge graph, providing a knowledge graph view on data from another system
 - Combining such virtual knowledge graphs can provide a *unified view* of data from different sources



Microdata and schema.org

- We have already seen that in the last lecture

HTML



```
<div itemscope  
  itemtype="http://schema.org/PostalAddress">  
  <span itemprop="name">Data and Web Science Group</span>  
</div>
```

```
_:1 a <http://schema.org/PostalAddress> .  
_:1 <http://schema.org/name> "Data and Web Science Group" .  
_:1 <http://schema.org/addressLocality> "Mannheim" .  
_:1 <http://schema.org/postalCode> "68131" .  
_:1 <http://schema.org/adressCounty> "Germany" .
```


Microdata and schema.org

- schema.org defines (among others)
 - products
 - product offers
 - businesses and local businesses (stores, cafés, ...)
 - books, movies, records
 - events
 - recipes
 - persons
 - ...

Movie

Thing > CreativeWork > Movie

A movie.

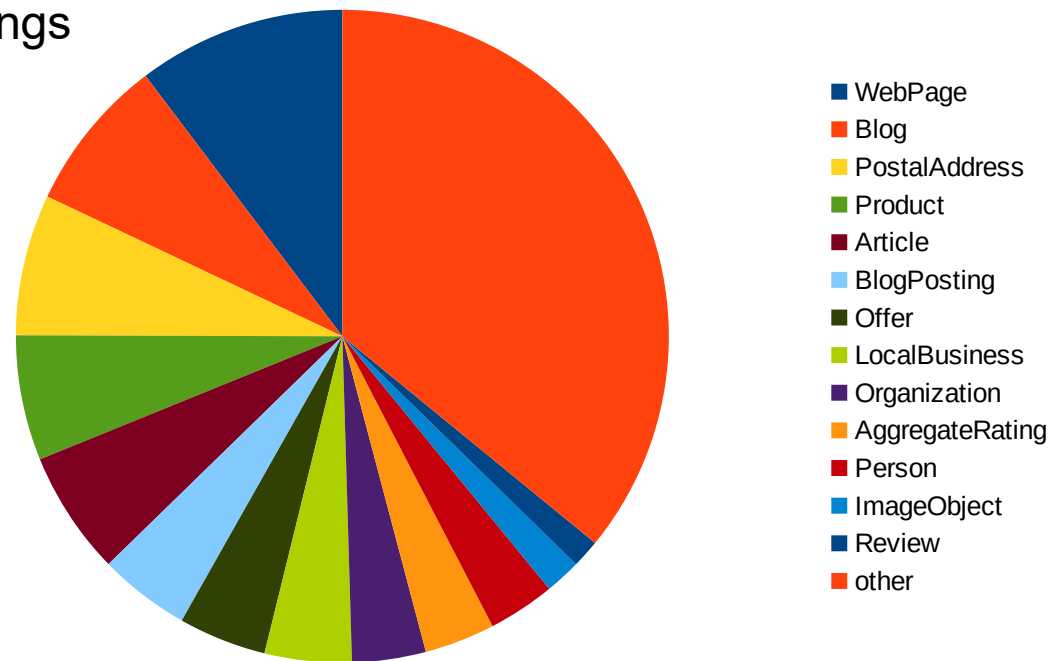
Usage: Between 10,000 and 50,000 domains

[more...]

Property	Expected Type	Description
Properties from Movie		
actor	Person	An actor, e.g. in tv, radio, movie, video games etc. Actors can be associated with individual items or with a series, episode, clip. Supersedes actors .
director	Person	A director of e.g. tv, radio, movie, video games etc. content. Directors can be associated with individual items or with a series, episode, clip. Supersedes directors .
duration	Duration	The duration of the item (movie, audio recording, event, etc.) in ISO 8601 date format .
musicBy	MusicGroup or Person	The composer of the soundtrack.
productionCompany	Organization	The production company or studio responsible for the item e.g. series, video game, episode etc.
subtitleLanguage	Text or Language	Languages in which subtitles/captions are available, in IETF BCP 47 standard format .
trailer	VideoObject	The trailer of a movie or tv/radio series, season, episode, etc.
Properties from CreativeWork		
about	Thing	The subject matter of the content.
accessibilityAPI	Text	Indicates that the resource is compatible with the referenced accessibility API (WebSchemas wiki lists possible values).
accessibilityControl	Text	Identifies input methods that are sufficient to fully control the described resource (WebSchemas wiki lists possible values).
accessibilityFeature	Text	Content features of the resource, such as accessible media, alternatives and supported enhancements for accessibility (WebSchemas wiki lists possible values).
accessibilityHazard	Text	A characteristic of the described resource that is physiologically dangerous to some users. Related to WCAG 2.0 guideline 2.3 (WebSchemas wiki lists possible values).
accountablePerson	Person	Specifies the Person that is legally accountable for the CreativeWork.
aggregateRating	AggregateRating	The overall rating, based on a collection of reviews or ratings, of the item.
alternativeHeadline	Text	A secondary title of the CreativeWork.

Deployment of schema.org

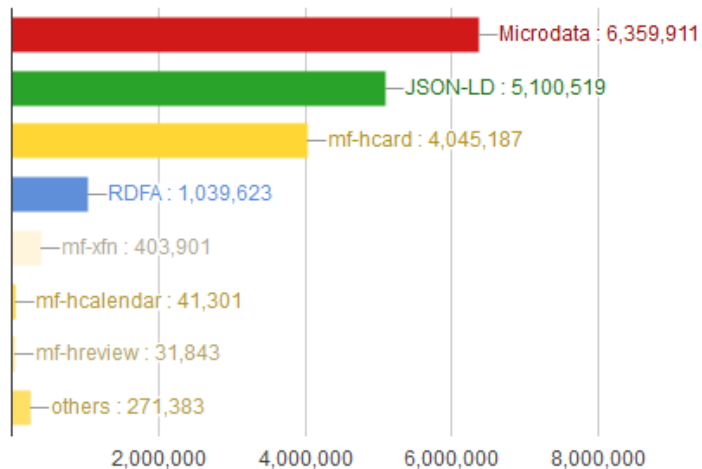
- Main topics of schema.org:
 - Meta information on web page content (web page, blog...)
 - Business data (products, offers, ...)
 - Contact data (businesses, persons, ...)
 - (Product) reviews and ratings
- ...and a massive long tail



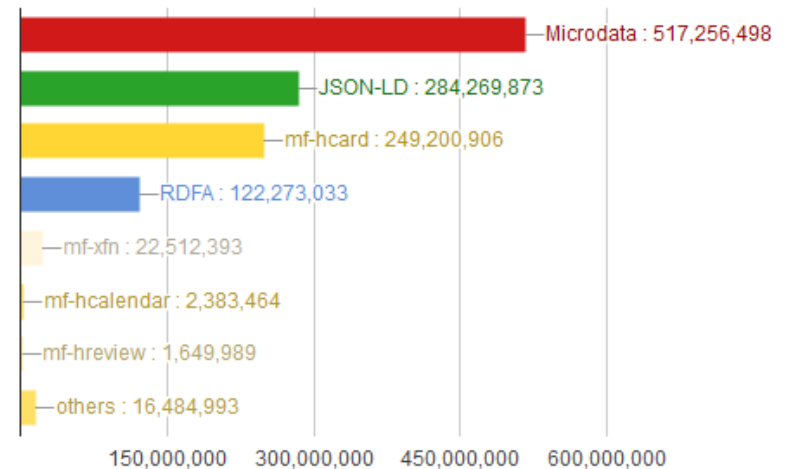
Growth of schema.org

- Note: schema.org is mainly used with Microdata
 - ...and Microdata is mainly used with schema.org

Domains with Triples



URLs with Triples



<http://webdatacommons.org/structureddata/>

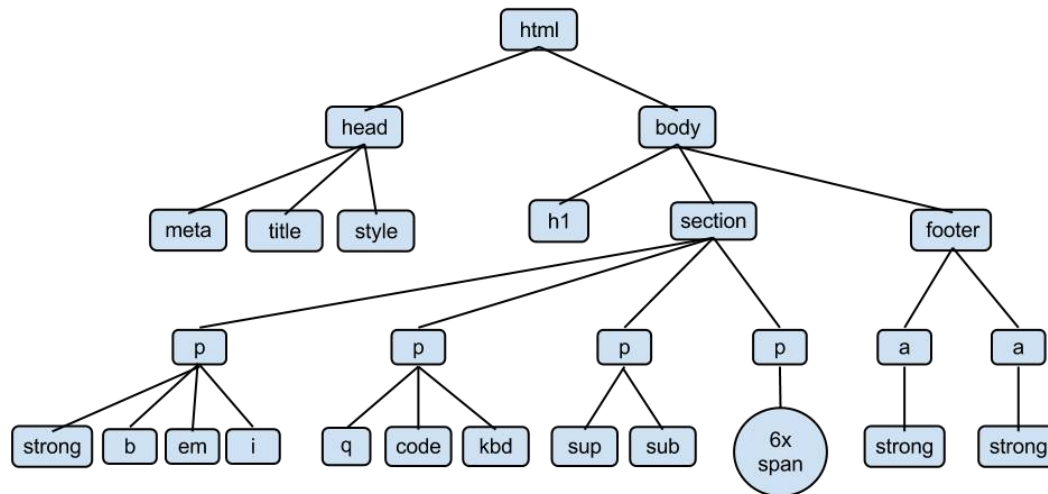
Microdata/schema.org vs. Linked Open Data

- Commonalities
 - Both encode machine-interpretable knowledge
 - Schema.org uses a standard vocabulary
 - Both can be encoded as RDF




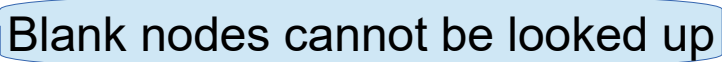
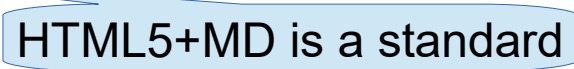
Microdata/schema.org vs. Linked Open Data

- Differences
 - Microdata is embedded in the DOM tree
 - i.e., the resulting RDF is always a set of trees
 - not a general directed graph
 - no cycles, no reification
 - Microdata uses only blank nodes and literals



Microdata/schema.org vs. Linked Open Data

- Linked Data Principles (Tim Berners-Lee 2006)

- Use URIs as names for things 
- Use HTTP URIs that can be looked up 
- When someone looks up a HTTP URI, provide useful information using a standard 

```
<div itemscope  
  itemtype="http://schema.org/PostalAddress">  
  <span itemprop="name">Data and Web Science Group</span>
```

```
<http://foo.bar/#1> a <http://schema.org/PostalAddress> .  
<http://foo.bar/#1> <http://schema.org/name> "Data and Web  
Science Group" .  
<http://foo.bar/#1> <http://schema.org/addressLocality>  
"Mannheim" .  
<http://foo.bar/#1> <http://schema.org/postalCode> "68131" .  
<http://foo.bar/#1> <http://schema.org/adressCounty> "Germany" .
```

Microdata/schema.org vs. Linked Open Data

- Linked Data Principles (TimBL 2006)
 - Use URIs as names for things
 - Use HTTP URIs that can be looked up
 - When someone looks up a HTTP URI, provide useful information using a standard
 - Include links to other URIs

This is possible with
schema.org/sameas



- Linkage within schema.org Microdata:
 - Only 0.02% of all data providers use schema.org/sameas

Microdata/schema.org vs. LOD

- Five Star Scheme (TimBL 2010)
 - * Available on the web with an open license
 - ** Available as machine-readable, structured data
 - *** as (**), using a non-proprietary format
 - **** plus: using open standards by the W3C
 - ***** plus: links to other datasets
- What's the license of web data?



Intermediate Summary

- Until today, we have dealt with the Semantic Web as a *vision*
- Today, we have seen two incarnations of that vision
 - Linked Open Data
 - schema.org/Microdata
- Both have a lot in common
- Linked Open Data:
 - A set of interconnected knowledge graphs, or a large knowledge graph
- schema.org/Microdata
 - A very large set of small knowledge graphs

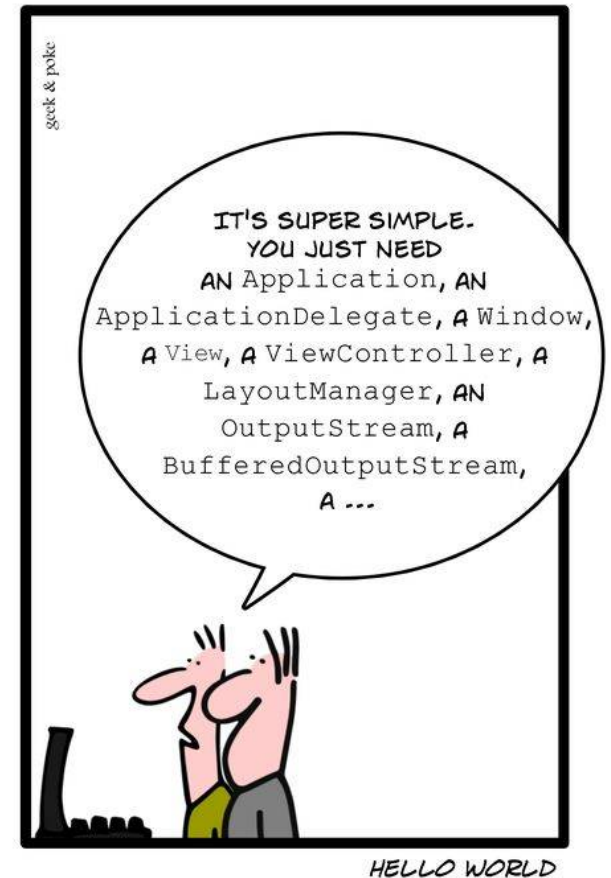
And Now for Something Completely Different



Programming with Knowledge Graphs

- Let's start with a simple application
 - a Hello World application for reading data from a knowledge graph

SIMPLY EXPLAINED



Using only Plain Java

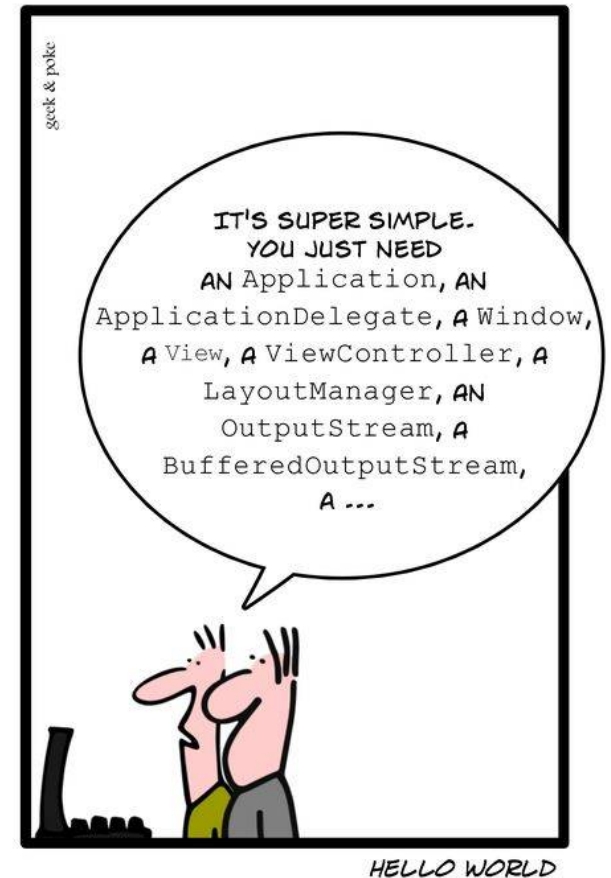
```
URL url = new URL("http://dbpedia.org/resource/Mannheim");
URLConnection conn = url.openConnection();
conn.addRequestProperty("Accept", "text/rdf+n3");
BufferedReader BR = new BufferedReader(
    new InputStreamReader(conn.getInputStream())
);

while (BR.ready()) {
    String triple = BR.readLine();
    StringTokenizer tokenizer = new StringTokenizer(triple, " ");
    String subject = tokenizer.nextToken();
    String predicate = tokenizer.nextToken();
    String object = tokenizer.nextToken();
    ...
}
```

Using only Plain Java

- Let's start with a simple application
 - a Hello World application for reading data from a knowledge graph
- Using plain Java is possible
 - but not very comfortable
 - there are more sophisticated frameworks

SIMPLY EXPLAINED



Programming with Jena

- Jena is a well-known Semantic Web programming framework
- started in 2000 at HP Labs
- Apache open source project since 2010
- Central concepts
 - Models (`class Model`) are RDF graphs
 - Resources (`class Resource`) are resources in RDF graphs
- Special features
 - Database connectors for persistence
 - Support for reasoning
 - Rule engines
 - Support for SPARQL (see next lecture)



Programming with Jena

- Reading a model from a derefencable URI

```
model.read("http://dbpedia.org/resource/Mannheim");
```

- Navigating within a model

```
Resource mannheim =  
    model.getResource("http://dbpedia.org/resource/  
                      Mannheim");
```

```
Resource countryOfMannheim =  
    model.getProperty(  
        "http://dbpedia.org/ontology/country").  
    getResource();
```


Programming with Jena

- Working with literals

```
Literal lit = mannheim.getProperty(  
    "http://www.w3.org/2000/01/rdf-schema#label").  
    getLiteral();  
  
lit.getString();  
lit.getLanguage();  
lit.getDatatype();
```

Programming with Jena

- Working with multi-valued relations

```
- StmtIterator iter = mannheim.getProperty(  
    "http://www.w3.org/2000/01/rdf-schema#label");  
- while(iter.hasNext()) {  
    Statement s = iter.next();  
    RDFNode node = s.getObject();  
    if(node.isLiteral())  
        System.out.println(node.asLiteral().getString());  
}
```

creates an iterator over all triples
with the subject node
and the given predicate

Iterators in Jena

- Jena uses the iterator pattern quite frequently

- e.g.:

```
StmtIterator iter = mannheim.getProperty(  
    "http://www.w3.org/2000/01/rdf-schema#label");
```

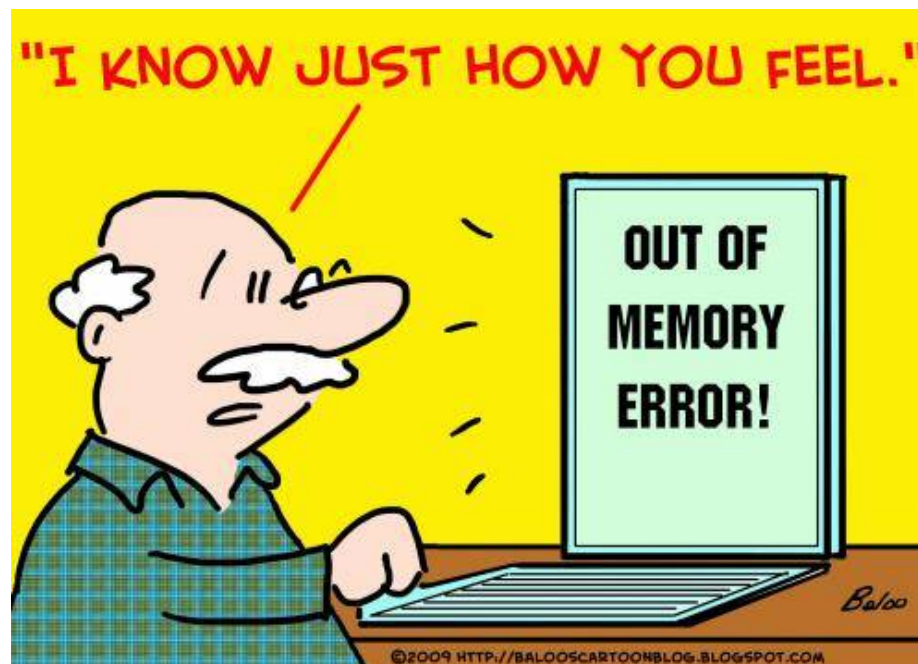
- But there is no such thing as

```
Collection<Statement> triples =  
mannheim.getProperty(  
    "http://www.w3.org/2000/01/rdf-schema#label");
```

- Why?

Iterators in Jena

- Knowledge graphs can be very large
- e.g., reading all triples from DBpedia
 - stored in List<Statement> would kill the main memory
 - iterators allow a more efficient memory use



Programming with Jena

- Manipulating models

```
p1.addProperty("http://xmlns.com/foaf/0.1/knows",p2);
```

- Watching model changes

```
class MyListener implements ModelChangeListener...  
MyListener listener = new MyListener();  
model.add(listener);
```

Reasoning with Jena

- Recap: we can derive information from a schema (T-Box) and data (A-box)
 - :knows rdfs:domain :Person .
 - :knows rdfs:range :Person .
 - :Peter :knows :Tom .
 - :Peter a :Person .
 - :Tom a :Person .
- Jena also supports reasoning

Reasoning with Jena

- Given: a schema and some data

```
Model schemaModel = ModelFactory.createDefaultModel();  
InputStream IS = new  
FileInputStream("data/example_schema.rdf");  
schemaModel.read(IS);
```

```
Model dataModel = ModelFactory.createDefaultModel();  
IS = new FileInputStream("data/example_data.rdf");  
dataModel.read(IS);
```

```
Model reasoningModel =  
    ModelFactory.createRDFSModel(schemaModel, dataModel);
```

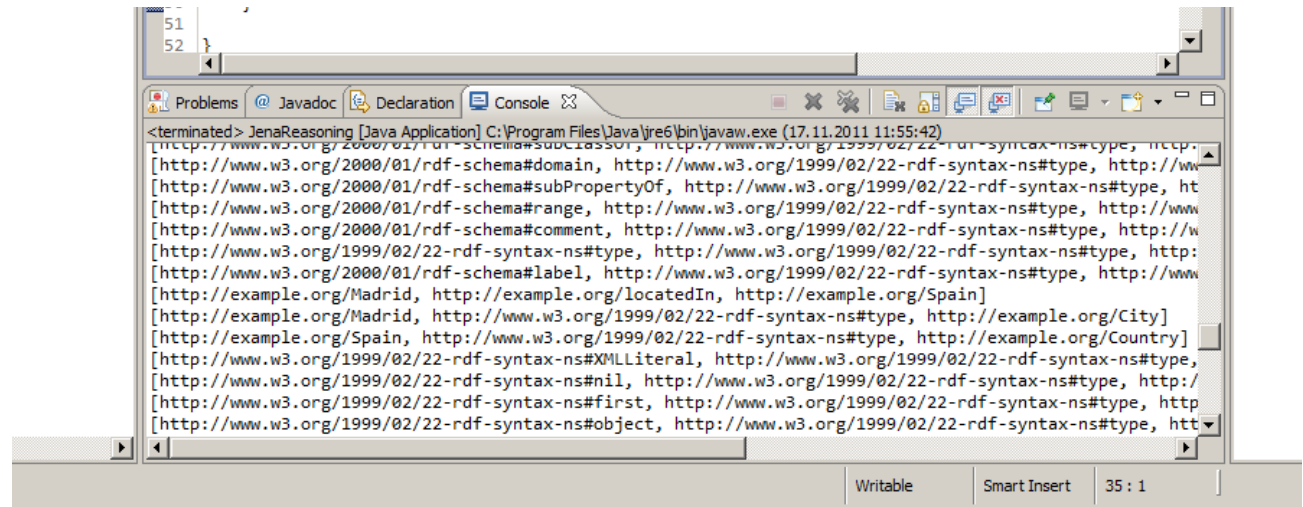
- Now, `reasoningModel` contains all derived facts

Reasoning with Jena

- Now, `reasoningModel` contains all derived facts

```
StmtIterator it =
    reasoningModel.listStatements();
while(it.hasNext()) {
    Statement s = it.next();
    System.out.println(s);
}
```

- Output:



Programming with RDFLib (Python)

- RDFLib is a Python library for working with RDF
- initial release 4 June, 2002 by Daniel Krech
 - Now being developed by the community at github:
<https://github.com/RDFLib/rdfLib/>
- it contains parsers and serializers for
 - RDF/XML, N3, NTriples, N-Quads, Turtle, TriX, RDFa and Microdata
- graph interface which can be backed by store implementations
 - memory storage
 - persistent storage on top of the Berkeley DB
- reasoning possible (<https://github.com/RDFLib/OWL-RL>)
- SPARQL 1.1 implementation (see next lecture)

Programming with RDFLib (Python)

- primary interface is a Graph
 - represented as a set of 3-item triples

```
[  
    (subject, predicate, object),  
    (subject1, predicate1, object1),  
    ...  
    (subjectN, predicateN, objectN)  
]
```

Programming with RDFLib (Python)

- Reading a model from a derefencable URI

```
import rdflib
g=rdflib.Graph()
g.load('http://dbpedia.org/resource/Mannheim')
```

- Print out all RDF triples

```
for s,p,o in g:
    print(s,p,o)
```

- Navigating within a graph

```
print(g.value(
    URIRef("http://dbpedia.org/resource/Mannheim"),
    URIRef("http://dbpedia.org/ontology/country")
))
```

Programming with RDFLib (Python)

- Most often reduced to basic triple matching
 - Graph.triples(subject, predicate, object)
 - each of them can be None (similar to null in Java)
- ```
for s,p,o in g.triples((None, RDF.type, FOAF.Person)):
 print("%s is a person"%s)
```
- Special functions for returning only specific parts
    - Graph.subjects(predicate, object) – returns only subjects
    - Graph.predicate(subject, object)
    - Graph.objects(subject, predicate)
    - Graph.subject\_objects(predicate)
    - Graph.subject\_predicates(object)
    - Graph.predicate\_objects(subject)
    - Graph.value(subject, predicate)
      - For just one value and not a generator/iterator

# Programming with RDFLib (Python)

- create URIs

```
mannheim = URIRef('http://example.com/Mannheim')
```

- create literals

```
mannheim_literal = Literal("Mannheim")
```

- Add triples to graph

```
g.add((mannheim, RDFS.label, mannheim_literal))
g.add((mannheim, RDFS.label, Literal("Mannheim", lang="de")))
```

- Serialize graph

```
print(g.serialize(format='n3'))
```

# Wrap-Up

- Today, we have seen
  - two incarnations of knowledge graphs as publicly available data
    - i.e., Linked Open Data
    - and Microdata/schema.org
- ...and we have learned how to write programs consuming data in knowledge graphs
  - Jena & RDFlib programming frameworks
  - loading RDF from files and from URLs
  - performing reasoning

# Questions?

