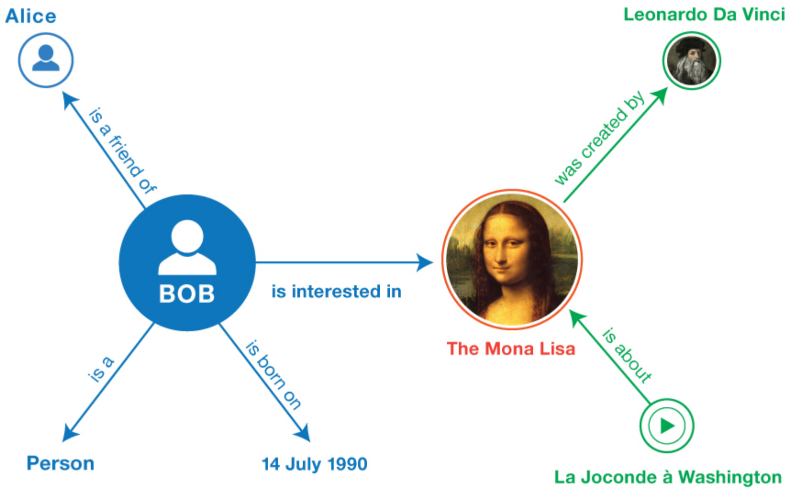
Resource Description Framework (RDF)

VL Semantic Technologies

Bernd Neumayr
(with contributions from Dieter Steiner)

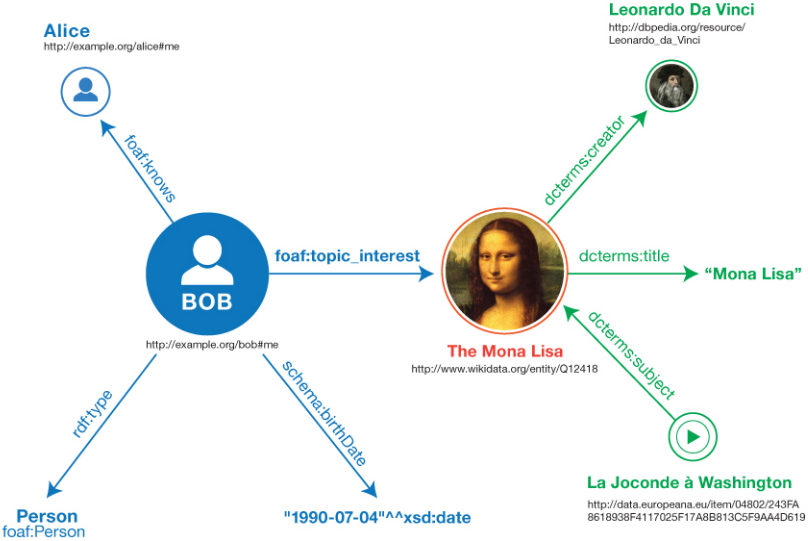
Department for Business Informatics – Data & Knowledge Engineering

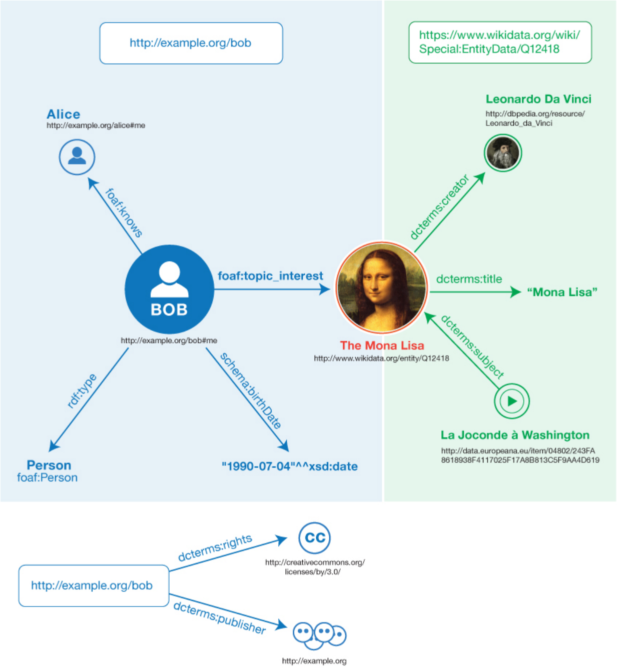
Informal Representation of an RDF Graph



Source: `http://www.w3.org/TR/rdf11-primer/`

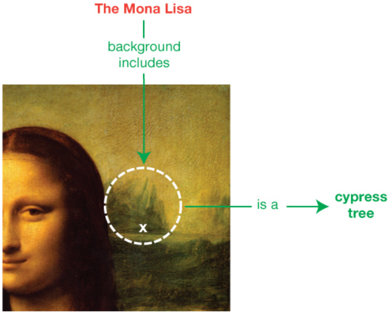
RDF Graph


Alice
http://example.org/alice#me
foaf:knows
Leonardo Da Vinci
http://dbpedia.org/resource/
Leonardo_da_Vinci
dcterms:creator
BOB
http://example.org/bob#me
foaf:topic_interest
dcterms:title
"Mona Lisa"
The Mona Lisa
http://www.wikidata.org/entity/Q12418
dcterms:subject
rdf:type
schema:birthDate
La Joconde à Washington
http://data.europeana.eu/item/04802/243FA
8618938F4117025F17A8B813C5F9AA4D619
Person
foaf:Person
"1990-07-04"^^xsd:date


http://example.org/bob
https://www.wikidata.org/wiki/Special:EntityData/Q12418
Alice
http://example.org/alice#me
foaf:knows
Leonardo Da Vinci
http://dbpedia.org/resource/Leonardo_da_Vinci
dcterms:creator
BOB
http://example.org/bob#me
foaf:topic_interest
The Mona Lisa
http://www.wikidata.org/entity/Q12418
dcterms:title
"Mona Lisa"
dcterms:subject
rdf:type
schema:birthDate
La Joconde à Washington
http://data.europeana.eu/item/04802/243FA8618938F4117025F17A8B813C5F9AA4D619
Person
foaf:Person
"1990-07-04"^^xsd:date
dcterms:rights
cc
http://creativecommons.org/licenses/by/3.0/
http://example.org/bob
dcterms:publisher
http://example.org

Source: `http://www.w3.org/TR/rdf11-primer/`

Blank Nodes



Quelle: `http://www.w3.org/TR/rdf11-primer/`

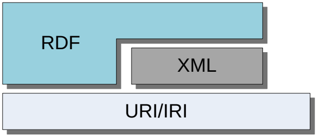
Resource Description Framework

Data Model for the Web of Data

1. URI/IRI – Uniform / Internationalized Resource Identifiers
2. Resource Description Framework (RDF)
3. Linked Data – Web of Data
4. RDF Schema

Overview

RDF within the Semantic Web Stack

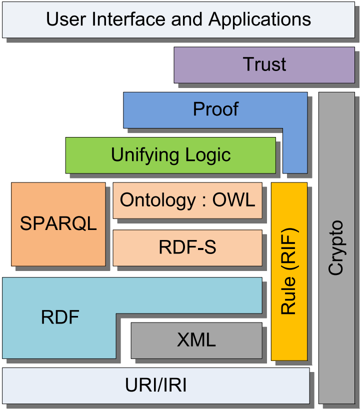


enabling technologies for the Semantic Web

- Identification mechanism
- Data model
- Data format

Overview

RDF within the Semantic Web Stack



enabling technologies for the Semantic Web

- Identification mechanism
- Data model
- Data format

URI/IRI – Uniform / Internationalized Resource Identifiers

Resources and their global identification

Everything is a Resource

- Files: documents, multimedia files, ontologies
- Things: objects, persons, products, abstract things such as countries, time periods, . . .
- Concepts: classes, relations
- Services
    - Services, in the sense of SOA, independent of a concrete implementation
    - Web Services, concrete implementations of services
    - . . .

Information vs. Non-Information Resource

Non-Information Resource

- Thing, Real-World-Entity
- e.g., the person ‘Bob’ is identified by `http://example.org/bob#me`

Information Resource

- Document, file on the Web; description, picture, or representation of a Non-Information Resource
- e.g., the RDF file that describes ‘Bob’ is identified by `http://example.org/bob`

- Information- and Non-Information Resources are disjoint
- ‘Ceci n’est pas une pipe’ `http://collections.lacma.org/node/239578`

Information vs. Non-Information Resource

Non-Information Resource

- Thing, Real-World-Entity
- e.g., the person ‘Bob’ is identified by `http://example.org/bob#me`

Information Resource

- Document, file on the Web; description, picture, or representation of a Non-Information Resource
- e.g., the RDF file that describes ‘Bob’ is identified by `http://example.org/bob`

- Information- and Non-Information Resources are disjoint
- ‘Ceci n’est pas une pipe’ `http://collections.lacma.org/node/239578`

Information vs. Non-Information Resource

Non-Information Resource

- Thing, Real-World-Entity
- e.g., the person ‘Bob’ is identified by `http://example.org/bob#me`

Information Resource

- Document, file on the Web; description, picture, or representation of a Non-Information Resource
- e.g., the RDF file that describes ‘Bob’ is identified by `http://example.org/bob`

- Information- and Non-Information Resources are disjoint
- ‘Ceci n’est pas une pipe’ `http://collections.lacma.org/node/239578`

Information vs. Non-Information Resource

Non-Information Resource

- Thing, Real-World-Entity
- e.g., the person ‘Bob’ is identified by `http://example.org/bob#me`

Information Resource

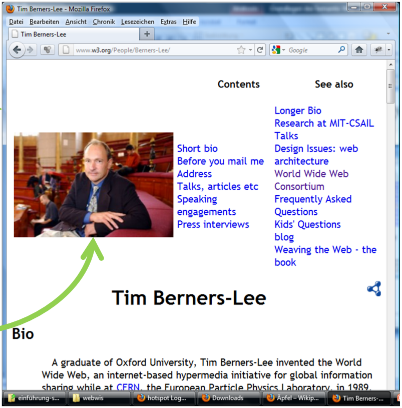
- Document, file on the Web; description, picture, or representation of a Non-Information Resource
- e.g., the RDF file that describes ‘Bob’ is identified by `http://example.org/bob`

- Information- and Non-Information Resources are disjoint
- ‘Ceci n’est pas une pipe’ `http://collections.lacma.org/node/239578`

Information Resources

http://www.w3.org/People/Berners-Lee/index.html

http://www.w3.org/Press/Stock/Berners-Lee/2001-europaeum-eighth.jpg


Tim Berners-Lee - Mozilla Firefox
Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe
Tim Berners-Lee
www.w3.org/People/Berners-Lee/
Google
einführung-s...
webwis
hotspot Log...
Downloads
Apfel – Wikip...
Tim Berners-...

Contents

See also

Short bio
Before you mail me
Address
Talks, articles etc
Speaking engagements
Press interviews

Longer Bio
Research at MIT-CSAIL
Talks
Design Issues: web architecture
World Wide Web Consortium
Frequently Asked Questions
Kids' Questions
blog
Weaving the Web - the book

Tim Berners-Lee

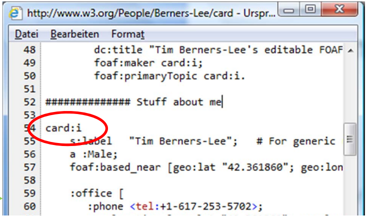
Bio

A graduate of Oxford University, Tim Berners-Lee invented the World Wide Web, an internet-based hypermedia initiative for global information sharing while at CERN, the European Particle Physics Laboratory, in 1989.

Information vs. Non-Information Resources

http://www.w3.org/People/Berners-Lee/card#i

http://www.w3.org/People/Berners-Lee/card



```
        dc:title "Tim Berners-Lee's editable FOAF
        foaf:maker card:i;
        foaf:primaryTopic card:i.

############### Stuff about me

card:i
    s:label   "Tim Berners-Lee";   # For generic
    a :Male;
    foaf:based_near [geo:lat "42.361860"; geo:lon

    :office [
       :phone <tel:+1-617-253-5702>;
```

No Unique Name Assumption

in databases: Unique Name Assumption

- Each tuple/object is identified by exactly one ID/OID
- Different objects in the database represent different things
- Or alternatively: There are only names and no objects. (Herbrand-Universe, Datalog)

in the Web: No Unique Name Assumption

- A single resource can be identified by multiple different IRIs.
- Every IRI identifies exactly one resource.

No Unique Name Assumption

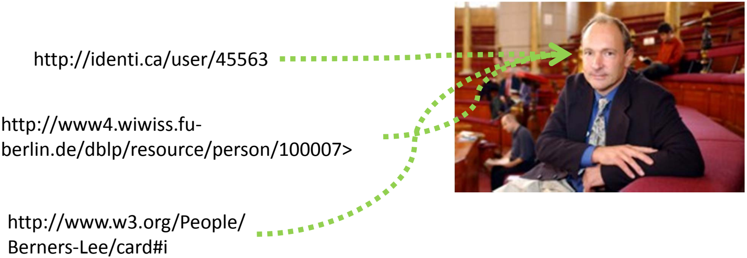
in databases: Unique Name Assumption

- Each tuple/object is identified by exactly one ID/OID
- Different objects in the database represent different things
- Or alternatively: There are only names and no objects. (Herbrand-Universe, Datalog)

in the Web: No Unique Name Assumption

- A single resource can be identified by multiple different IRIs.
- Every IRI identifies exactly one resource.

No Unique Name Assumption in the Web



http://identi.ca/user/45563

http://www4.wiwiss.fu-berlin.de/dblp/resource/person/100007>

http://www.w3.org/People/Berners-Lee/card#i

Structure of HTTP-URIs

`http://www.foo.com/data/persons?family=black#john`

- Scheme
  `http`
- Authority
  `www.foo.com`
- Path
  `/data/persons`
- Query (optional)
  `?family=black`
- Fragment (optional, not part of an HTTP-Request)
  `john`

Structure of HTTP-URIs

```
http://www.foo.com/data/persons?family=black#john
```

- Scheme
  `http`
- Authority
  `www.foo.com`
- Path
  `/data/persons`
- Query (optional)
  `?family=black`
- Fragment (optional, not part of an HTTP-Request)
  `john`

Structure of HTTP-URIs

`http://www.foo.com/data/persons?family=black#john`

- Scheme
  `http`
- Authority
  `www.foo.com`
- Path
  `/data/persons`
- Query (optional)
  `?family=black`
- Fragment (optional, not part of an HTTP-Request)
  `john`

Structure of HTTP-URIs

`http://www.foo.com/data/persons?family=black#john`

- Scheme
  `http`
- Authority
  `www.foo.com`
- Path
  `/data/persons`
- Query (optional)
  `?family=black`
- Fragment (optional, not part of an HTTP-Request)
  `john`

Structure of HTTP-URIs

`http://www.foo.com/data/persons?family=black#john`

- Scheme
  `http`
- Authority
  `www.foo.com`
- Path
  `/data/persons`
- Query (optional)
  `?family=black`
- Fragment (optional, not part of an HTTP-Request)
  `john`

Structure of HTTP-URIs

`http://www.foo.com/data/persons?family=black#john`

- Scheme
  `http`
- Authority
  `www.foo.com`
- Path
  `/data/persons`
- Query (optional)
  `?family=black`
- Fragment (optional, not part of an HTTP-Request)
  `john`

Abbreviations of IRIs: Namespace Prefixes

without Namespace Prefixes

```
<http://www.dbpedia.org/resource/Linz>
<http://example.org/famblack.rdf#Jim>
```

with Namespace Prefixes

```
@prefix dbpedia: <http://www.dbpedia.org/resource/>.
@prefix f: <http://example.org/famblack.rdf#>.

dbpedia:Linz
f:Jim
```

Types of IRIs, Standards

Uniform Resource Locator (URL)

identification through access mechanism permits dereferencing, e.g., using HTTP or FTP

Uniform Resource Name (URN)

no dereferencing possible, e.g., urn:ISBN

Internationalized Resource Identifier (IRI)

Internationalized URI; allows Unicode characters

Types of IRIs, Standards

JKU

Uniform Resource Locator (URL)

identification through access mechanism permits dereferencing, e.g., using HTTP or FTP

Uniform Resource Name (URN)

no dereferencing possible, e.g., urn:ISBN

Internationalized Resource Identifier (IRI)

Internationalized URI; allows Unicode characters

Types of IRIs, Standards

Uniform Resource Locator (URL)

identification through access mechanism permits dereferencing, e.g., using HTTP or FTP

Uniform Resource Name (URN)

no dereferencing possible, e.g., urn:ISBN

Internationalized Resource Identifier (IRI)

Internationalized URI; allows Unicode characters

IRI/URI – Summary

- ‘Everything’ is a resource
- Non-Information Resource vs. Information Resource
- Every IRI identifies exactly one resource
- No Unique Name Assumption: Multiple IRIs can identify the same resource

additional information:

- `http://www.w3.org/TR/webarch/#id-resources`
- `http://www.ietf.org/rfc/rfc3986`

Resource Description Framework (RDF)

A Data Model for the Web of Data

- Overview
- Data Model
- Serialization
- RDF Dataset, Named Graphs
- Summary

RDF

RDF is a data model for a

- **linked** (RDF Statement = Link),
- **decentralized** (distributed and without central control mechanism),
- **machine interpretable** (uniform and easily interpretable data model: a subject is described by a predicate and an object),
- **conceptual** (close to the mental conceptions of humans: direct representation of entities and their relations and properties)

description of resources.

RDF

RDF is a data model for a

- **linked** (RDF Statement = Link),
- **decentralized** (distributed and without central control mechanism),
- **machine interpretable** (uniform and easily interpretable data model: a subject is described by a predicate and an object),
- **conceptual** (close to the mental conceptions of humans: direct representation of entities and their relations and properties)

description of resources.

RDF

RDF is a data model for a

- **linked** (RDF Statement = Link),
- **decentralized** (distributed and without central control mechanism),
- **machine interpretable** (uniform and easily interpretable data model: a subject is described by a predicate and an object),
- **conceptual** (close to the mental conceptions of humans: direct representation of entities and their relations and properties)

description of resources.

RDF

RDF is a data model for a

- **linked** (RDF Statement = Link),
- **decentralized** (distributed and without central control mechanism),
- **machine interpretable** (uniform and easily interpretable data model: a subject is described by a predicate and an object),
- **conceptual** (close to the mental conceptions of humans: direct representation of entities and their relations and properties)

description of resources.

RDF

RDF is a data model for a

- **linked** (RDF Statement = Link),
- **decentralized** (distributed and without central control mechanism),
- **machine interpretable** (uniform and easily interpretable data model: a subject is described by a predicate and an object),
- **conceptual** (close to the mental conceptions of humans: direct representation of entities and their relations and properties)

description of resources.

Related Technologies

- Frames
- Conceptual Graphs
- Topic Maps
- Metadata Frameworks
- RSS (Really Simple Syndication)
- . . .

RDF Statement (Triple)

subject

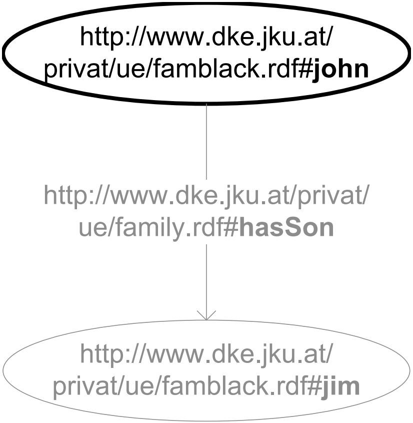
- IRI (Resource) or
- Blank Node

predicate

- IRI (Property)

object

- IRI (Resource),
- Blank Node or
- Literal


http://www.dke.jku.at/privat/ue/famblack.rdf#**john**
http://www.dke.jku.at/privat/ue/family.rdf#**hasSon**
http://www.dke.jku.at/privat/ue/famblack.rdf#**jim**

RDF Statement (Triple)

subject

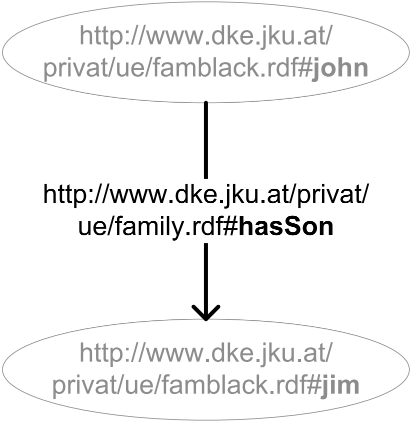
- IRI (Resource) or
- Blank Node

predicate

- IRI (Property)

object

- IRI (Resource),
- Blank Node or
- Literal



http://www.dke.jku.at/privat/ue/famblack.rdf#**john**

http://www.dke.jku.at/privat/ue/family.rdf#**hasSon**

http://www.dke.jku.at/privat/ue/famblack.rdf#**jim**

RDF Statement (Triple)

subject

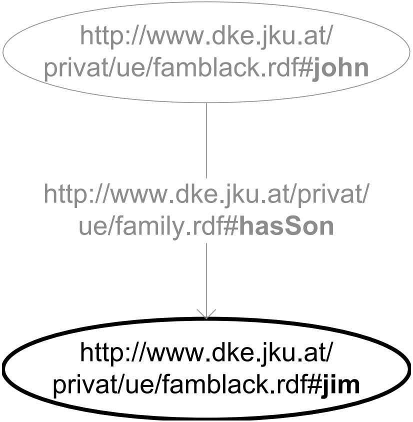
- IRI (Resource) or
- Blank Node

predicate

- IRI (Property)

object

- IRI (Resource),
- Blank Node or
- Literal


http://www.dke.jku.at/privat/ue/famblack.rdf#**john**
http://www.dke.jku.at/privat/ue/family.rdf#**hasSon**
http://www.dke.jku.at/privat/ue/famblack.rdf#**jim**

RDF Statement (Triple)

subject

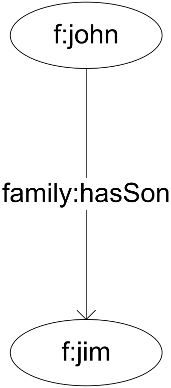
- IRI (Resource) or
- Blank Node

predicate

- IRI (Property)

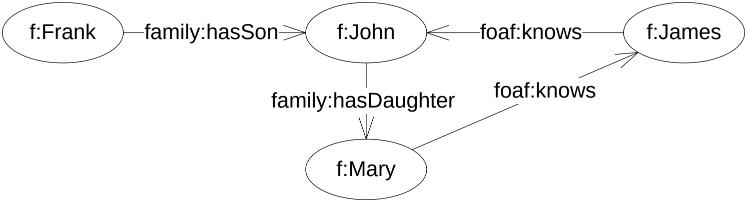
object

- IRI (Resource),
- Blank Node or
- Literal



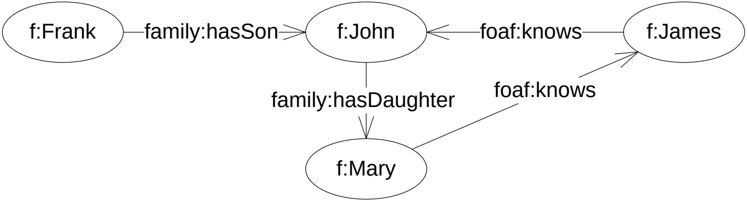
RDF Graph = Set of Statements

An RDF graph is a directed, labelled graph.



RDF Graph = Set of Statements

An RDF graph is a directed, labelled graph.

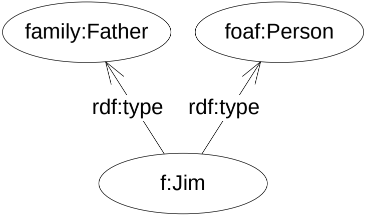


corresponds to a ternary relation (set of triples):

```
subject   predicate           object
---------------------------------------
f:Frank   family:hasSon       f:John
f:James   foaf:knows          f:John
f:John    family:hasDaughter  f:Mary
f:Mary    foaf:knows          f:James
```

Classification

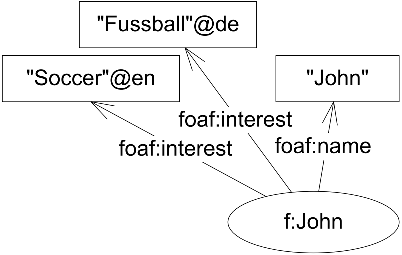
- `rdf:type`
- Multiple classifications possible
- No distinction between classes and individuals



Literals

Plain Literal

- Character string with optional language definition
- Represents itself
- in RDF 1.1, every literal has a type



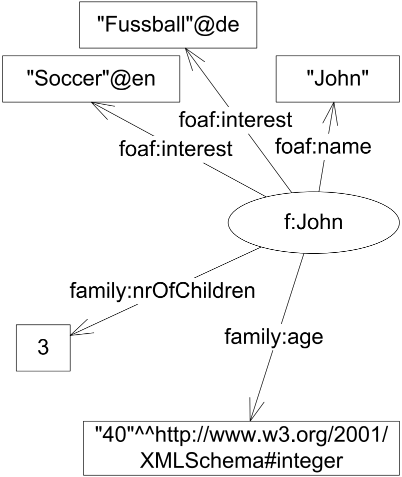
Literals

Plain Literal

- Character string with optional language definition
- Represents itself
- in RDF 1.1, every literal has a type

Typed Literal

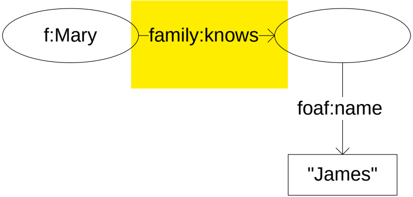
- Character string and data type URI
- Represents element from the data type's value space



Blank Nodes – Unnamed Nodes

- Auxiliary nodes
- Specification of IRI not necessary

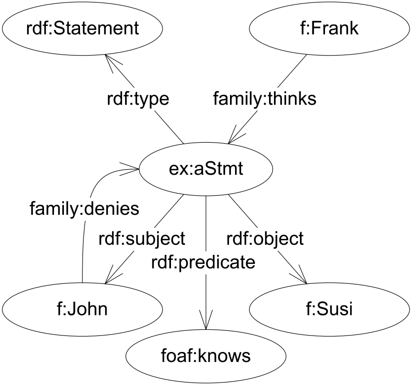
Mary knows someone who is called James.



Reification: Statements about Statements

- Statements as resources
- Particular vocabulary:
  - `rdf:subject`
  - `rdf:predicate`
  - `rdf:object`
  - `rdf:Statement`

Frank thinks that “John knows Susi”, John denies this.



Container and Collections

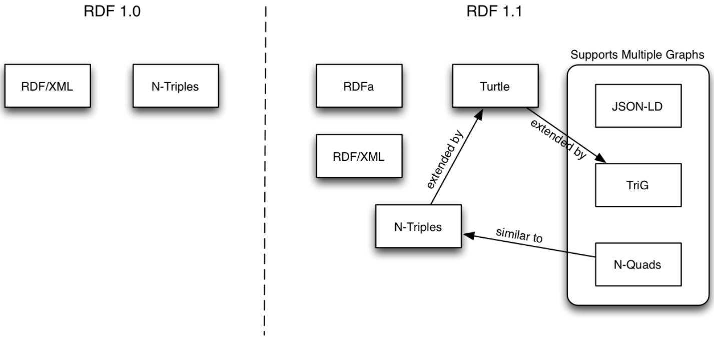
Container

- Adding additional elements is possible.
- `rdf:Bag`, `rdf:Seq`, `rdf:Alt`

Collection (Finalized List)

- No additional elements can be added.
- `rdf:List`, `rdf:first`, `rdf:rest`, `rdf:nil`

Serialization of RDF



Source: `http://www.w3.org/TR/rdf11-new/`

Serialization of RDF

Turtle family

Turtle and TriG offer a convenient, abbreviated notation for N-Triples and N-Quads.

RDF/XML

widespread; good tool support

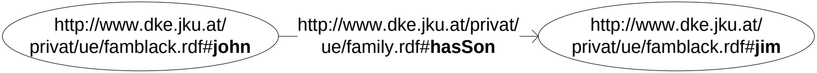
JSON-LD

JSON for Linked Data

RDFa

Embedding of RDF in HTML, Google Rich Snippets; alternative: Microdata (schema.org), Microformats

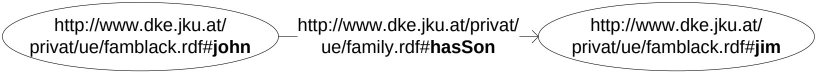
Turtle


http://www.dke.jku.at/
privat/ue/famblack.rdf#**john**
http://www.dke.jku.at/privat/
ue/family.rdf#**hasSon**
http://www.dke.jku.at/
privat/ue/famblack.rdf#**jim**

```
<http://example.org/famblack.rdf#John>
 <http://example.org/family.rdf#hasSon>
   <http://example.org/famblack.rdf#Jim>.
```

abbreviated notation using prefixes:

Turtle

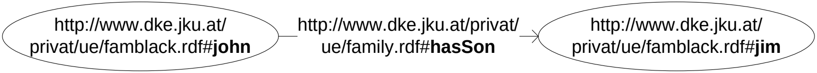

http://www.dke.jku.at/ privat/ue/famblack.rdf#**john**
http://www.dke.jku.at/privat/ ue/family.rdf#**hasSon**
http://www.dke.jku.at/ privat/ue/famblack.rdf#**jim**

```
<http://example.org/famblack.rdf#John>
 <http://example.org/family.rdf#hasSon>
   <http://example.org/famblack.rdf#Jim>.
```

abbreviated notation using prefixes:

```
@prefix family: <http://example.org/family.rdf#>.
@prefix f: <http://example.org/famblack.rdf#>.
```

Turtle



http://www.dke.jku.at/privat/ue/famblack.rdf#**john**

http://www.dke.jku.at/privat/ue/family.rdf#**hasSon**

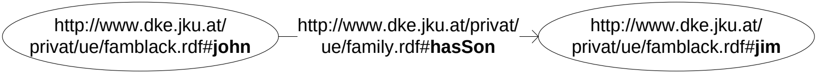
http://www.dke.jku.at/privat/ue/famblack.rdf#**jim**

```
<http://example.org/famblack.rdf#John>
 <http://example.org/family.rdf#hasSon>
   <http://example.org/famblack.rdf#Jim>.
```

abbreviated notation using prefixes:

```
@prefix family: <http://example.org/family.rdf#>.
@prefix f: <http://example.org/famblack.rdf#>.

f:John family:hasSon f:Jim.
```

RDF/XML



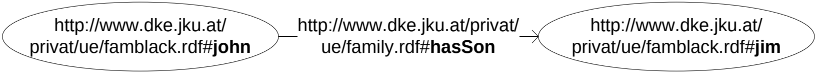
http://www.dke.jku.at/privat/ue/famblack.rdf#**john**

http://www.dke.jku.at/privat/ue/family.rdf#**hasSon**

http://www.dke.jku.at/privat/ue/famblack.rdf#**jim**

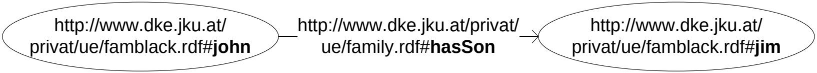
```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:f="http://example.org/famblack.rdf#"
  xmlns:family="http://example.org/family.rdf#">






</rdf:RDF>
```

RDF/XML


http://www.dke.jku.at/
privat/ue/famblack.rdf#**john**
http://www.dke.jku.at/privat/
ue/family.rdf#**hasSon**
http://www.dke.jku.at/
privat/ue/famblack.rdf#**jim**

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:f="http://example.org/famblack.rdf#"
  xmlns:family="http://example.org/family.rdf#">
  <rdf:Description rdf:about=
        "http://example.org/famblack.rdf#John">


  </rdf:Description>
</rdf:RDF>
```

RDF/XML



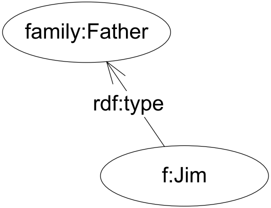
http://www.dke.jku.at/privat/ue/famblack.rdf#**john**

http://www.dke.jku.at/privat/ue/family.rdf#**hasSon**

http://www.dke.jku.at/privat/ue/famblack.rdf#**jim**

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:f="http://example.org/famblack.rdf#"
  xmlns:family="http://example.org/family.rdf#">
  <rdf:Description rdf:about=
        "http://example.org/famblack.rdf#John">
    <family:hasSon rdf:resource=
          "http://example.org/famblack.rdf#Jim"/>
  </rdf:Description>
</rdf:RDF>
```

Classification



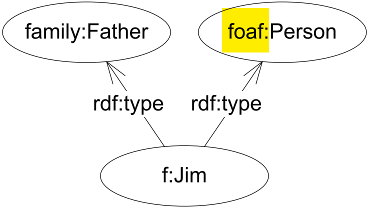
Turtle:

```
f:Jim a                    family:Father.
```

RDF/XML:

```
<family:Father rdf:about=
  "http://example.org/famblack.rdf#Jim">


</family:Father>
```

Classification



Turtle:

```
f:Jim a foaf:Person, family:Father.
```

RDF/XML:

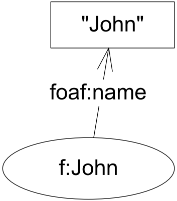
```
<family:Father rdf:about=
  "http://example.org/famblack.rdf#Jim">
  <rdf:type rdf:resource=
      "http://xmlns.com/foaf/0.1/Person"/>
</family:Father>
```

Literals (Turtle)

```
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.
```

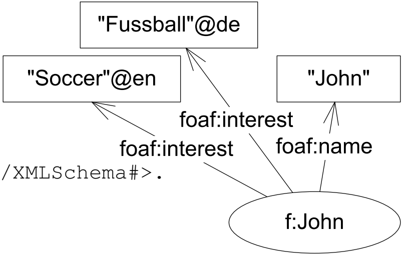
.

Literals (Turtle)



```
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.

f:John
    foaf:name "John";
                          .
```

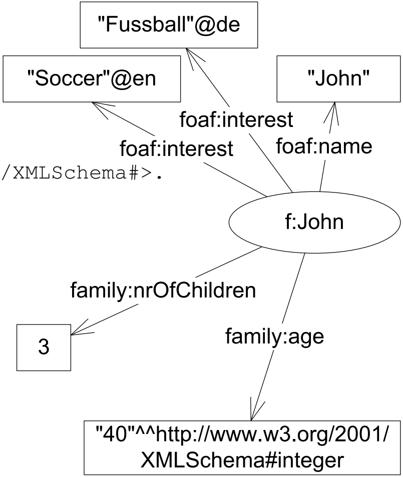
Literals (Turtle)



```
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.

f:John
    foaf:name "John";
    foaf:interest "Soccer"@en,
                  "Fussball"@de;

                        .
```

Literals (Turtle)

```
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.

f:John
    foaf:name "John";
    foaf:interest "Soccer"@en,
                  "Fussball"@de;
    family:age "40"^^xsd:integer;
    family:nrOfChildren 3.
```



Literals (RDF/XML)

```
<rdf:Description rdf:about=
      "http://example.org/famblack.rdf#John">
   <foaf:name>John<foaf:name>








 </foaf:Person>
```

Literals (RDF/XML)

```
<rdf:Description rdf:about=
      "http://example.org/famblack.rdf#John">
   <foaf:name>John<foaf:name>
   <foaf:interest xml:lang="de">
       Fussball
   </foaf:interest>
   <foaf:interest xml:lang="en">
       Soccer
   </foaf:interest>









 </foaf:Person>
```

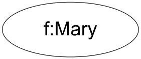
Literals (RDF/XML)

```
<rdf:Description rdf:about=
      "http://example.org/famblack.rdf#John">
   <foaf:name>John<foaf:name>
   <foaf:interest xml:lang="de">
       Fussball
   </foaf:interest>
   <foaf:interest xml:lang="en">
       Soccer
   </foaf:interest>
   <family:nrOfChildren rdf:datatype=
     "http://www.w3.org/2001/XMLSchema#integer">
       3
   </family:nrOfChildren>
   <family:age rdf:datatype=
      "http://www.w3.org/2001/XMLSchema#integer">
       40
   </family:age>
 </foaf:Person>
```

Blank Nodes

Turtle:

```
f:Mary

        .
```



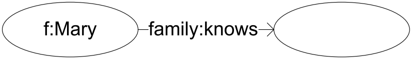
RDF/XML:

```
<rdf:Description rdf:about=
  "http://example.org/famblack.rdf#Mary">



</rdf:Description>
```

Blank Nodes

Turtle:

```
f:Mary family:knows [

       ].
```



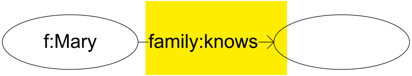
RDF/XML:

```
<rdf:Description rdf:about=
  "http://example.org/famblack.rdf#Mary">



</rdf:Description>
```

Blank Nodes

Turtle:

```
f:Mary family:knows [

       ].
```



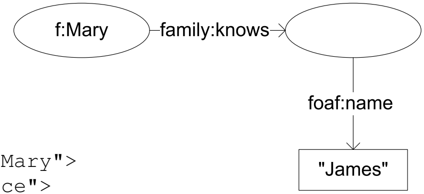
RDF/XML:

```
<rdf:Description rdf:about=
  "http://example.org/famblack.rdf#Mary">
  <foaf:knows rdf:parseType="Resource">

  </foaf:knows>
</rdf:Description>
```

Blank Nodes

Turtle:

```
f:Mary family:knows [
           foaf:name "James"
       ].
```



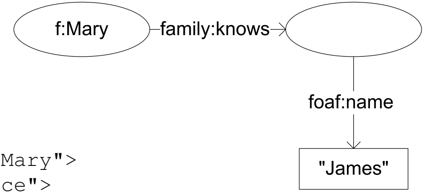
RDF/XML:

```
<rdf:Description rdf:about=
  "http://example.org/famblack.rdf#Mary">
  <foaf:knows rdf:parseType="Resource">

  </foaf:knows>
</rdf:Description>
```

Blank Nodes

Turtle:

```
f:Mary family:knows [
            foaf:name "James"
        ].
```



RDF/XML:

```
<rdf:Description rdf:about=
  "http://example.org/famblack.rdf#Mary">
  <foaf:knows rdf:parseType="Resource">
    <foaf:name>James</foaf:name>
  </foaf:knows>
</rdf:Description>
```

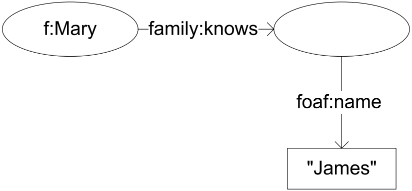
Blank Nodes - Alternative Turtle Notation

Turtle:

```
f:Mary family:knows [
    foaf:name "James"].
```

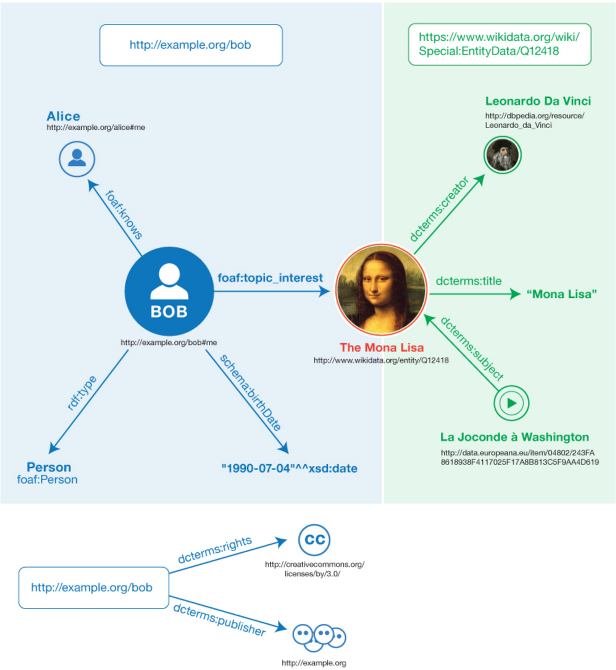
alternative Turtle notation:

```
f:Mary family:knows _:1.
_:1 foaf:name "James".
```



RDF Dataset, Named Graphs

- An RDF dataset consists of one or more RDF graphs.
  - one unnamed Default Graph
  - any number of Named Graphs
- Named Graphs are identified by an IRI or a Blank Node.
- Graphs can be linked.
- Statements can be made about graphs.
- A dataset can be serialized using N-Quads, TriG, or JSON-LD.



Quelle: `http://www.w3.org/TR/rdf11-primer/`

Serialization of an RDF Dataset using TriG

```
BASE   <http://example.org/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX schema: <http://schema.org/>
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX wd: <http://www.wikidata.org/entity/>

GRAPH <http://example.org/bob>
  {
    <bob#me>
        a foaf:Person ;
        foaf:knows <alice#me> ;
        schema:birthDate "1990-07-04"^^xsd:date ;
        foaf:topic_interest wd:Q12418 .
  }

GRAPH <https://www.wikidata.org/wiki/Special:EntityData/Q12418>
  {
    wd:Q12418
        dcterms:title "Mona Lisa" ;
        dcterms:creator <http://dbpedia.org/resource/Leonardo_da_Vinci> .

    <http://data.europeana.eu/item/04802/243FA8618938F4117025F17A8B813C5F9AA4D619>
       dcterms:subject wd:Q12418 .
  }

<http://example.org/bob>
    dcterms:publisher <http://example.org> ;
    dcterms:rights <http://creativecommons.org/licenses/by/3.0/> .
```

Source: http://www.w3.org/TR/rdf11-primer/

RDF – Summary

- RDF is a data model for the linked, decentralized, machine interpretable, conceptual description of resources.
- Every statement is a triple in the form of a subject-predicate-object expression.
- Sets of RDF statements form a graph.
- There are multiple formats for serializing RDF, e.g., Turtle, RDF/XML.
- To be continued ...

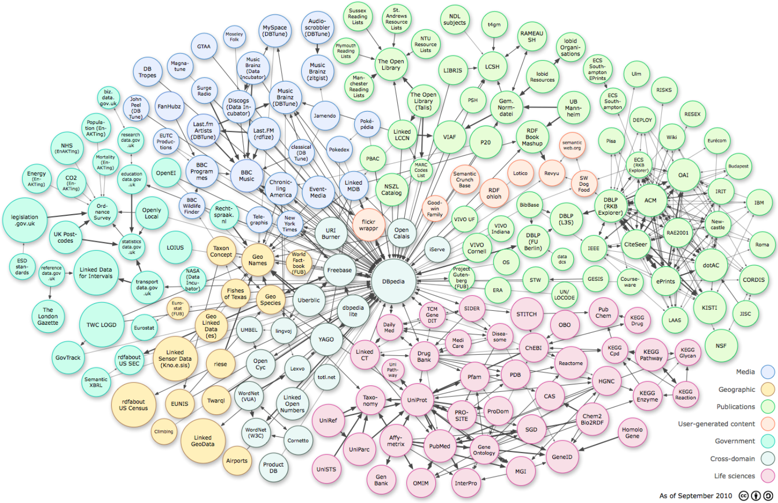
Further Information

- RDF 1.1 Primer
  `http://www.w3.org/TR/rdf11-primer/`
- Current Recommendations: RDF 1.1
  `http://www.w3.org/standards/techs/rdf`
- List of tools for working with RDF
  `http://www.w3.org/RDF/`

Linked Data – Web of Data

The Web as *Decentralized* Database

Linked Data Cloud



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

Linked Data – Data Sources

- DBPedia `http://dbpedia.org`
- Wikidata `http://www.wikidata.org/`
- Google Knowledge Graph Search API `https://developers.google.com/knowledge-graph/`
- GeoNames `http://www.geonames.org/`
- data.gov.uk `http://data.gov.uk/data/search?res_format=RDF`
- datahub `http://datahub.io/dataset`
- Freebase; foundation for Google Knowledge Graph `http://www.freebase.com`, `https://developers.google.com/freebase/`

Web of Documents → Web of Data

- Starting point: Vast amount of unstructured data and knowledge in the Web represented as hypertext
- Problem: Knowledge and data extraction is difficult and only partially supported by machines. Document Retrieval (search engines), Browsing, reading
- Goal: Query (analogous to database)

Web of Documents → Web of Data

JKU

- Starting point: Vast amount of unstructured data and knowledge in the Web represented as hypertext
- Problem: Knowledge and data extraction is difficult and only partially supported by machines. Document Retrieval (search engines), Browsing, reading
- Goal: Query (analogous to database)

Web of Documents → Web of Data

- Starting point: Vast amount of unstructured data and knowledge in the Web represented as hypertext
- Problem: Knowledge and data extraction is difficult and only partially supported by machines. Document Retrieval (search engines), Browsing, reading
- Goal: Query (analogous to database)

Databases → Web of Data

- Starting point: Vast amount of structured data in databases
- Problem: Data integration
  - heterogeneous data models
  - heterogeneous conceptualizations
  - heterogeneous database schemas
  - instance level: duplicates, referencing
- Goal: simple, inter-organizational integration of databases
- First steps (Web of Data):
  - RDF as data model for integration (DB→RDF Mappings)
  - Ontologies, Ontology Mappings
  - Schema-Mappings, Schema-Ontology-Mappings
  - IRIs as global IDs, linking, sameAs

Databases → Web of Data

- Starting point: Vast amount of structured data in databases
- Problem: Data integration
  - heterogeneous data models
  - heterogeneous conceptualizations
  - heterogeneous database schemas
  - instance level: duplicates, referencing
- Goal: simple, inter-organizational integration of databases
- First steps (Web of Data):
  - RDF as data model for integration (DB→RDF Mappings)
  - Ontologies, Ontology Mappings
  - Schema-Mappings, Schema-Ontology-Mappings
  - IRIs as global IDs, linking, sameAs

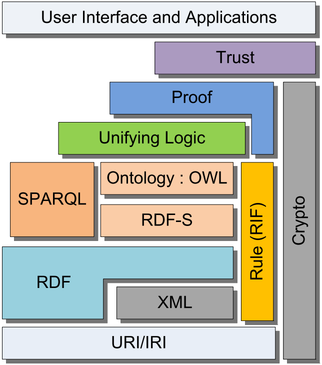
Databases → Web of Data

- Starting point: Vast amount of structured data in databases
- Problem: Data integration
  - heterogeneous data models
  - heterogeneous conceptualizations
  - heterogeneous database schemas
  - instance level: duplicates, referencing
- Goal: simple, inter-organizational integration of databases
- First steps (Web of Data):
  - RDF as data model for integration (DB→RDF Mappings)
  - Ontologies, Ontology Mappings
  - Schema-Mappings, Schema-Ontology-Mappings
  - IRIs as global IDs, linking, sameAs

Databases → Web of Data

- Starting point: Vast amount of structured data in databases
- Problem: Data integration
  - heterogeneous data models
  - heterogeneous conceptualizations
  - heterogeneous database schemas
  - instance level: duplicates, referencing
- Goal: simple, inter-organizational integration of databases
- First steps (Web of Data):
  - RDF as data model for integration (DB→RDF Mappings)
  - Ontologies, Ontology Mappings
  - Schema-Mappings, Schema-Ontology-Mappings
  - IRIs as global IDs, linking, sameAs

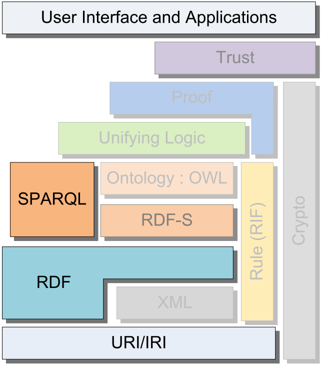
Semantic Web Stack



Semantic Web is more difficult than initially thought.
Technologies are in part still

- **not mature**:
  Unifying Logic, Proof, Trust
- **too complex** for efficient computation:
  Reasoning, . . .
- **too difficult** in their application:
  Ontology Engineering, . . .

Semantic Web Stack



Semantic Web is more difficult than initially thought.
Technologies are in part still

- **not mature**:
  Unifying Logic, Proof, Trust
- **too complex** for efficient computation:
  Reasoning, . . .
- **too difficult** in their application:
  Ontology Engineering, . . .

Keep it Simple!

Related Approaches and Technologies

- Mashups (ad-hoc, not generic)
- Semantic Search (automatic identification of entities and relationships in hypertext; very complex)
- Web Data Extraction, Lixto
- Database integration
- Distributed databases
- . . .

Linked Data on the Web

- Distributed and linked RDF data
- Non-local IRIs in RDF documents as links to other documents
- RDF Statements as typed links

Four rules:

- Use IRIs as names for things.
- Use HTTP IRIs so that people can look up those names.
- When someone looks up a IRI, provide useful information.
- Include links to other IRIs. so that they can discover more things.

(Berners-Lee, `http://www.w3.org/DesignIssues/LinkedData.html`)

Linked Data on the Web

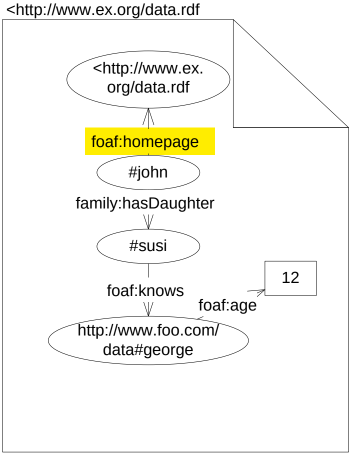
- Distributed and linked RDF data
- Non-local IRIs in RDF documents as links to other documents
- RDF Statements as typed links

Four rules:

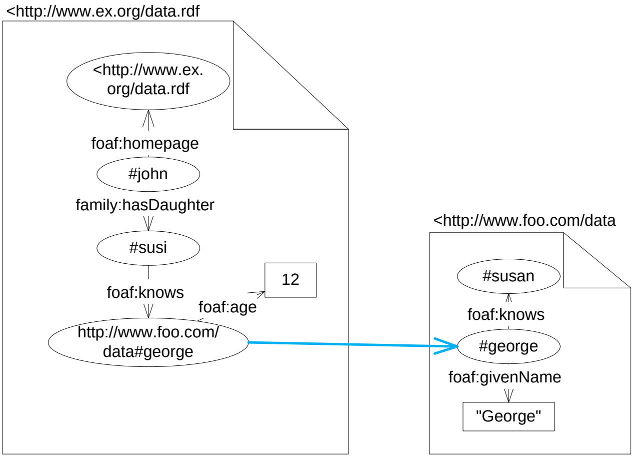
- Use IRIs as names for things.
- Use HTTP IRIs so that people can look up those names.
- When someone looks up a IRI, provide useful information.
- Include links to other IRIs. so that they can discover more things.

(Berners-Lee, `http://www.w3.org/DesignIssues/LinkedData.html`)

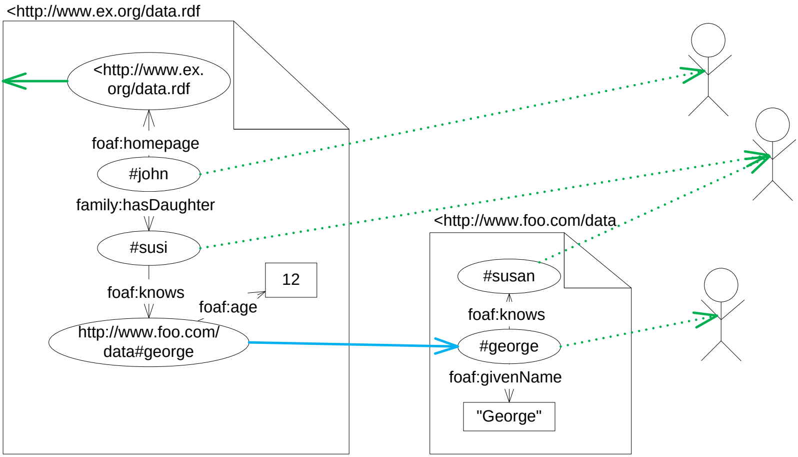
Distributed RDF



Distributed RDF



Distributed RDF



URIs for Non-Information Resources

- How to name things and concepts and how to access their description?
- Variant 1: Hash-URIs
- Variant 2: 303 Response

URIs for Non-Information Resources

- How to name things and concepts and how to access their description?
- Variant 1: Hash-URIs
- Variant 2: 303 Response

Variant 1: Hash-URIs

- Tim Berners-Lee
  `http://www.w3.org/People/Berners-Lee/card#i`
- Description of Tim-Berners-Lee
  `http://www.w3.org/People/Berners-Lee/card`

Variant 1: Hash-URIs

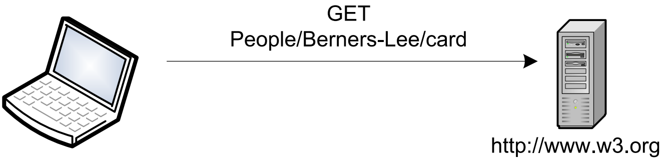
- Tim Berners-Lee
  `http://www.w3.org/People/Berners-Lee/card#i`
- Description of Tim-Berners-Lee
  `http://www.w3.org/People/Berners-Lee/card`

Dereferencing of
`http://www.w3.org/People/Berners-Lee/card#i`?

Variant 1: Hash-URIs

- Tim Berners-Lee
  `http://www.w3.org/People/Berners-Lee/card#i`
- Description of Tim-Berners-Lee
  `http://www.w3.org/People/Berners-Lee/card`

Dereferencing of
`http://www.w3.org/People/Berners-Lee/card#i`?



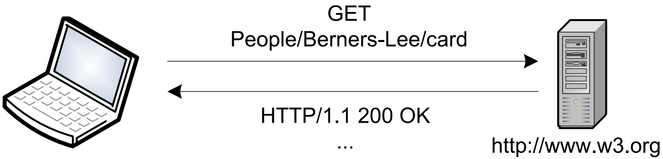
GET
People/Berners-Lee/card

http://www.w3.org

Variant 1: Hash-URIs

- Tim Berners-Lee
  `http://www.w3.org/People/Berners-Lee/card#i`
- Description of Tim-Berners-Lee
  `http://www.w3.org/People/Berners-Lee/card`

Dereferencing of
`http://www.w3.org/People/Berners-Lee/card#i`?



Variant 2: 303 Response

- Linz
  `http://de.dbpedia.org/resource/Linz`
- Description of Linz
  `http://de.dbpedia.org/data/Linz`

Variant 2: 303 Response

- Linz

  `http://de.dbpedia.org/resource/Linz`
- Description of Linz

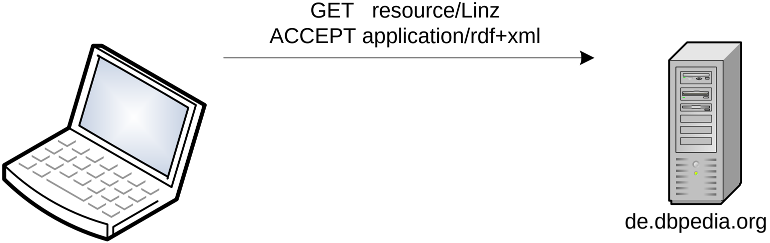
  `http://de.dbpedia.org/data/Linz`

Dereferencing of `http://de.dbpedia.org/resource/Linz`?

Variant 2: 303 Response

- Linz
  `http://de.dbpedia.org/resource/Linz`
- Description of Linz
  `http://de.dbpedia.org/data/Linz`

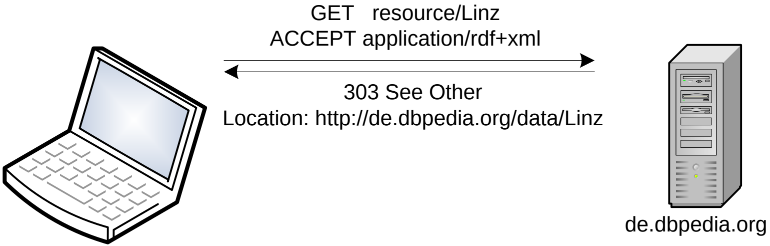
Dereferencing of
`http://de.dbpedia.org/resource/Linz`?



Variant 2: 303 Response

- Linz
  `http://de.dbpedia.org/resource/Linz`
- Description of Linz
  `http://de.dbpedia.org/data/Linz`

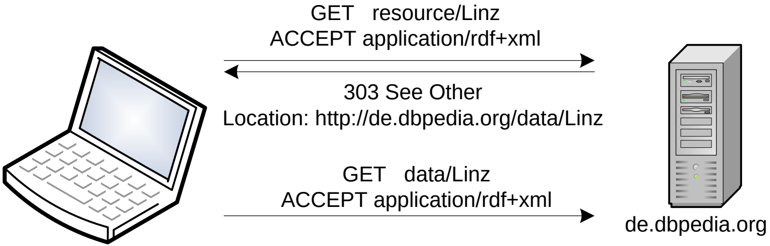
Dereferencing of
`http://de.dbpedia.org/resource/Linz`?


GET resource/Linz
ACCEPT application/rdf+xml
303 See Other
Location: http://de.dbpedia.org/data/Linz
de.dbpedia.org

Variant 2: 303 Response

- Linz
  `http://de.dbpedia.org/resource/Linz`
- Description of Linz
  `http://de.dbpedia.org/data/Linz`

Dereferencing of
`http://de.dbpedia.org/resource/Linz`?



GET resource/Linz
ACCEPT application/rdf+xml

303 See Other
Location: http://de.dbpedia.org/data/Linz

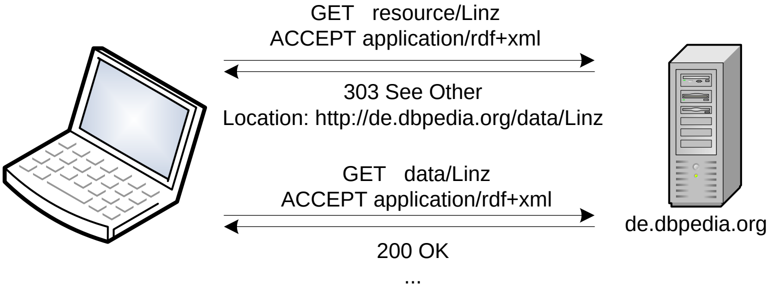
GET data/Linz
ACCEPT application/rdf+xml

de.dbpedia.org

Variant 2: 303 Response

- Linz
  `http://de.dbpedia.org/resource/Linz`
- Description of Linz
  `http://de.dbpedia.org/data/Linz`

Dereferencing of
`http://de.dbpedia.org/resource/Linz`?



GET resource/Linz
ACCEPT application/rdf+xml

303 See Other
Location: http://de.dbpedia.org/data/Linz

GET data/Linz
ACCEPT application/rdf+xml

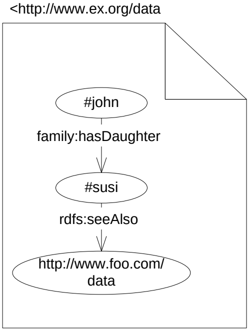
200 OK
...

rdfs:seeAlso

`rdfs:seeAlso` is used to indicate that additional information can be found at the referenced location.

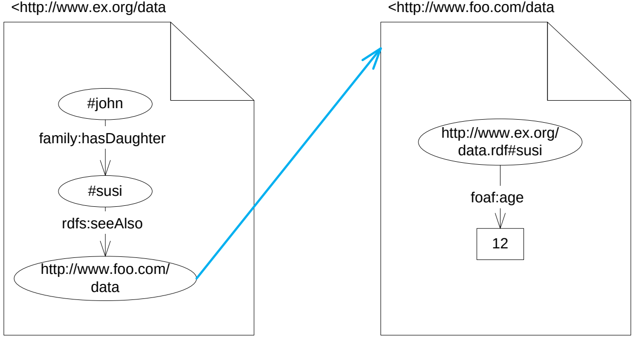
rdfs:seeAlso

`rdfs:seeAlso` is used to indicate that additional information can be found at the referenced location.



rdfs:seeAlso

`rdfs:seeAlso` is used to indicate that additional information can be found at the referenced location.

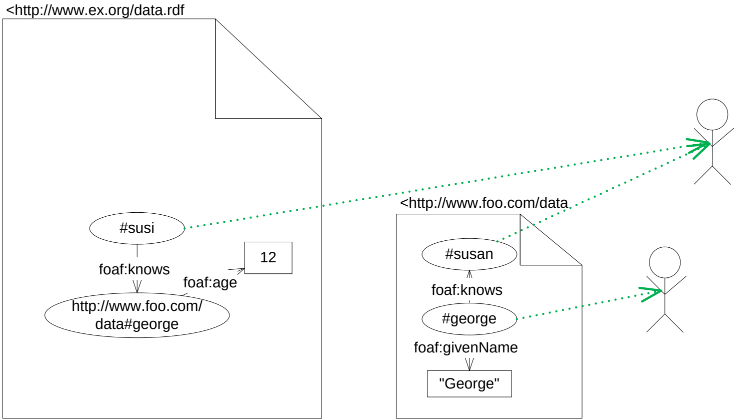


owl:sameAs

`owl:sameAs` is used to indicate that two IRIs identify the same Resource.

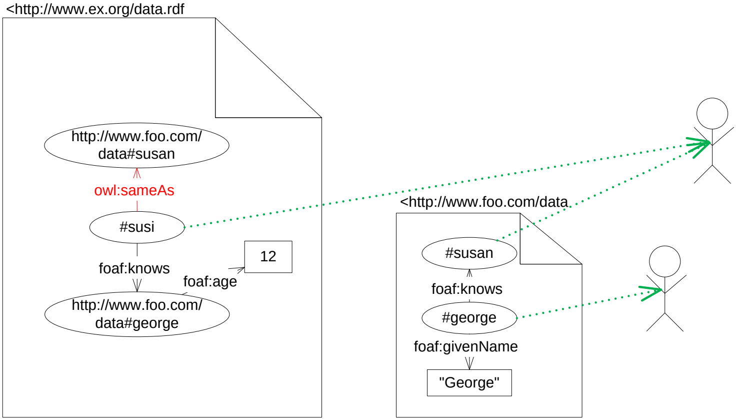
owl:sameAs

`owl:sameAs` is used to indicate that two IRIs identify the same Resource.



owl:sameAs

`owl:sameAs` is used to indicate that two IRIs identify the same Resource.



Summary

Linked Data on the Web

- Linked data are distributed, linked RDF data.
- HTTP-URLs are used as identifiers and locators of Information and Non-Information Resources.

Further Reading

- How to Publish Linked Data on the Web
  `http://wifo5-03.informatik.uni-mannheim.de/bizer/pub/LinkedDataTutorial/`
- Christian Bizer, Tom Heath, Tim Berners-Lee: Linked Data - The Story So Far. Int. J. Semantic Web Inf. Syst. 5(3): 1-22 (2009)
- Nigel Shadbolt, Kieron O'Hara, Tim Berners-Lee, Nicholas Gibbins, Hugh Glaser, Wendy Hall, m. c. schraefel: Linked Open Government Data: Lessons from Data.gov.uk. IEEE Intelligent Systems 27(3): 16-24 (2012)

RDF Schema

RDF Vocabulary Description Language

- Selected Language Elements
- RDFS Examples

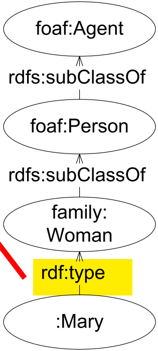
Class Hierarchies

"a" und "rdf:type" können beide dafür verwendet werden

Given:

```
:Mary a family:Woman.
family:Woman rdfs:subClassOf
                   foaf:Person.
foaf:Person rdfs:subClassOf
                   foaf:Agent.
```

Deduced:



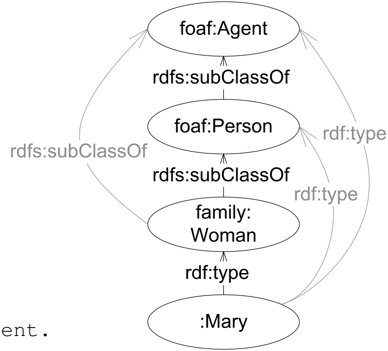
Class Hierarchies

Given:

```
:Mary a family:Woman.
family:Woman rdfs:subClassOf
                   foaf:Person.
foaf:Person rdfs:subClassOf
                   foaf:Agent.
```

Deduced:

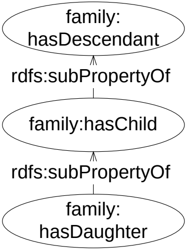
```
:Mary a foaf:Person.
:Mary a foaf:Agent.
family:Woman rdfs:subClassOf foaf:Agent.
```



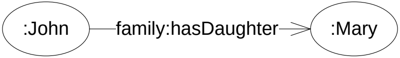
Property Hierarchies

Given:

```
family:hasDaughter
   rdfs:subPropertyOf
        family:hasChild.
family:hasChild
   rdfs:subPropertyOf
     family:hasDescendant.
:John family:hasDaughter :Mary.
```



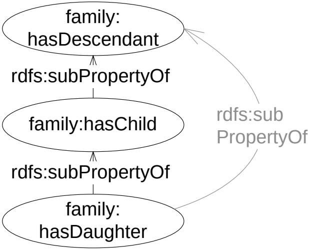
Deduced:



Property Hierarchies

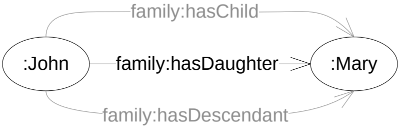
Given:

```
family:hasDaughter
   rdfs:subPropertyOf
        family:hasChild.
family:hasChild
   rdfs:subPropertyOf
     family:hasDescendant.
:John family:hasDaughter :Mary.
```



Deduced:

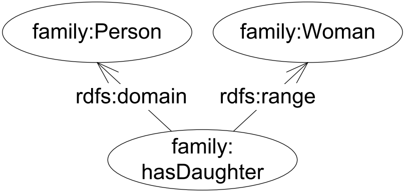
```
family:hasDaughter
   rdfs:subPropertyOf
        family:hasDescendant.
:John family:hasChild :Mary.
:John family:hasDescendant :Mary.
```



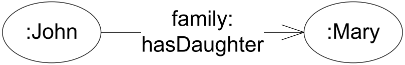
Domain and Range of Properties

Given:

```
family:hasDaughter
  rdfs:domain family:Person;
  rdfs:range family:Woman.
:John family:hasDaughter :Mary.
```



Deduced:



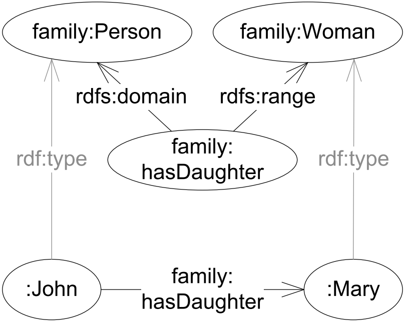
Domain and Range of Properties

Given:

```
family:hasDaughter
  rdfs:domain family:Person;
  rdfs:range family:Woman.
:John family:hasDaughter :Mary.
```

Deduced:

```
:John a family:Person.
:Mary a family:Woman.
```

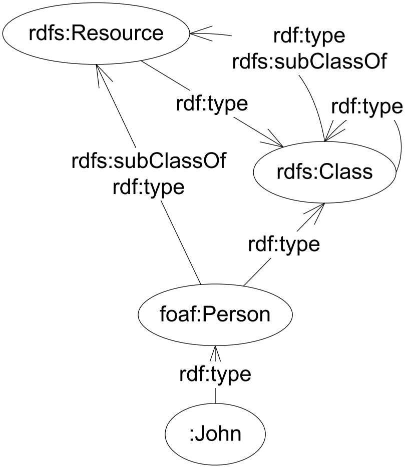


Metamodeling

No separation between model levels:

- Individuals, such as `:John`
- Classes, such as `family:Person`
- Metaclasses, such as `rdfs:Class`

A class can be a member of itself. Attention, this is not compatible with OWL DL!



RDF Schema – Summary

- RDF Schema is a semantic extension of RDF
- It is used to define simple vocabularies (Concepts/Classes and Properties)
- Further Reading:
  `http://www.w3.org/TR/rdf-schema/`

RDFS Examples

RDF Schema (1)

The following sample document is given:

```
...
:hasYoungDaughter rdfs:subPropertyOf :hasDaughter.
:hasDaughter rdfs:range :Women.
:hasChild rdfs:range :Person;
          rdfs:domain :Person.
:Women rdfs:subClassOf :Person.
:Maria :hasYoungDaughter :Susi.
```

Can the following statements be deduced?

- `:Maria a :Person`
- `:Maria :hasChild :Susi`
- `:Susi a :Women`
- `:Susi a :Person`
- `:Maria :hasDaughter :Susi`

RDF Schema (1)

The following sample document is given:

```
...
:hasYoungDaughter rdfs:subPropertyOf :hasDaughter.
:hasDaughter rdfs:range :Women.
:hasChild rdfs:range :Person;
          rdfs:domain :Person.
:Women rdfs:subClassOf :Person.
:Maria :hasYoungDaughter :Susi.
```

Can the following statements be deduced?

- `:Maria a :Person` no
- `:Maria :hasChild :Susi`
- `:Susi a :Women`
- `:Susi a :Person`
- `:Maria :hasDaughter :Susi`

RDF Schema (1)

The following sample document is given:

```
...
:hasYoungDaughter rdfs:subPropertyOf :hasDaughter.
:hasDaughter rdfs:range :Women.
:hasChild rdfs:range :Person;
          rdfs:domain :Person.
:Women rdfs:subClassOf :Person.
:Maria :hasYoungDaughter :Susi.
```

Can the following statements be deduced?

- `:Maria a :Person` no
- `:Maria :hasChild :Susi` no
- `:Susi a :Women`
- `:Susi a :Person`
- `:Maria :hasDaughter :Susi`

RDF Schema (1)

The following sample document is given:

```
...
:hasYoungDaughter rdfs:subPropertyOf :hasDaughter.
:hasDaughter rdfs:range :Women.
:hasChild rdfs:range :Person;
          rdfs:domain :Person.
:Women rdfs:subClassOf :Person.
:Maria :hasYoungDaughter :Susi.
```

Can the following statements be deduced?

- `:Maria a :Person` no
- `:Maria :hasChild :Susi` no
- `:Susi a :Women` yes
- `:Susi a :Person`
- `:Maria :hasDaughter :Susi`

RDF Schema (1)

The following sample document is given:

```
...
:hasYoungDaughter rdfs:subPropertyOf :hasDaughter.
:hasDaughter rdfs:range :Women.
:hasChild rdfs:range :Person;
          rdfs:domain :Person.
:Women rdfs:subClassOf :Person.
:Maria :hasYoungDaughter :Susi.
```

Can the following statements be deduced?

- `:Maria a :Person` no
- `:Maria :hasChild :Susi` no
- `:Susi a :Women` yes
- `:Susi a :Person` yes
- `:Maria :hasDaughter :Susi`

RDF Schema (1)

The following sample document is given:

```
...
:hasYoungDaughter rdfs:subPropertyOf :hasDaughter.
:hasDaughter rdfs:range :Women.
:hasChild rdfs:range :Person;
          rdfs:domain :Person.
:Women rdfs:subClassOf :Person.
:Maria :hasYoungDaughter :Susi.
```

Can the following statements be deduced?

- `:Maria a :Person` no
- `:Maria :hasChild :Susi` no
- `:Susi a :Women` yes
- `:Susi a :Person` yes
- `:Maria :hasDaughter :Susi` yes

RDF Schema (2)

Can the following information be modeled using RDF(S)? If so, provide a solution:

- Each person knows its children.
- Each car is a vehicle.
- Only persons can own things; everyone who owns something is a person.
- Every father has at least one child.

RDF Schema (2)

Can the following information be modeled using RDF(S)? If so, provide a solution:

- Each person knows its children.

```
:hasChild rdfs:subPropertyOf foaf:knows.
```

- Each car is a vehicle.
- Only persons can own things; everyone who owns something is a person.
- Every father has at least one child.

RDF Schema (2)

Can the following information be modeled using RDF(S)? If so, provide a solution:

- Each person knows its children.

```
    :hasChild rdfs:subPropertyOf foaf:knows.
```

- Each car is a vehicle.

```
    :Car rdfs:subClassOf :Vehicle.
```

- Only persons can own things; everyone who owns something is a person.

- Every father has at least one child.

RDF Schema (2)

Can the following information be modeled using RDF(S)? If so, provide a solution:

- Each person knows its children.

```
:hasChild rdfs:subPropertyOf foaf:knows.
```

- Each car is a vehicle.

```
:Car rdfs:subClassOf :Vehicle.
```

- Only persons can own things; everyone who owns something is a person.

```
:owns rdfs:domain :Person.
```

- Every father has at least one child.

beispiele:

RDF Schema (2)

Can the following information be modeled using RDF(S)? If so, provide a solution:

- Each person knows its children.

```
:hasChild rdfs:subPropertyOf foaf:knows.
```

- Each car is a vehicle.

```
:Car rdfs:subClassOf :Vehicle.
```

- Only persons can own things; everyone who owns something is a person.

```
:owns rdfs:domain :Person.
```

- Every father has at least one child.

```
no solution
```

extra hint:
für typenbeschreibung funktioniert Bsp "Mary is a Person"
:Mary a :Person . ODER   :Mary rdf:type :Person .

extra hint 2:
für Integer funktioniert Bsp "Mary ist 27 Jahre alt"
:Mary foaf :age 27 . ODER   :Mary foaf:age "27"^^xsd:integer .