

Unit 1 – Preliminaries

RDF, RDFS, OWL & SPARQL

Axel Polleres

DERI, National University of Ireland, Galway

Reasoning Web Summer School 2008

Unit Outline

1. Motivation – Aggregating Linked Open Data by Rules
2. How can I publish data? RDF
3. How can I query that data? SPARQL
4. What does that data mean? Ontologies described in RDFS + OWL
5. What's next?

Building a review panel for our book

- You want to publish a book containing the lectures of this Summer School. . .



Building a review panel for our book

- You want to publish a book containing the lectures of this Summer School. . .
- Who are the right reviewers?



Building a review panel for our book

- You want to publish a book containing the lectures of this Summer School. . .
- Who are the right reviewers?
- Which reviewers are in conflict?



Building a review panel for our book

- You want to publish a book containing the lectures of this Summer School. . .
- Who are the right reviewers?
- Which reviewers are in conflict?
- Observation: Much of the necessary data is available on the Web!



Building a review panel for our book

- You want to publish a book containing the lectures of this Summer School. . .
- Who are the right reviewers?
- Which reviewers are in conflict?
- Observation: Much of the necessary data is available on the Web!



Building a review panel for our book

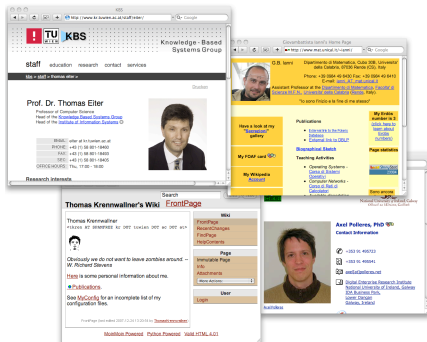
- You want to publish a book containing the lectures of this Summer School. . .
- Who are the right reviewers?
- Which reviewers are in conflict?
- Observation: Much of the necessary data is available on the Web!



Questions:

- Where do I get the right data?
- Which rules and query languages do I use to aggregate this data?
- Which systems are out there to support me?

Where is the data? 1/4



Rules and Ontologies for the Semantic Web¹

- Thomas Eiter¹, Christos Bessis²,
Thomas Krennwallner¹ and Axel Polleres¹
- ¹ Institute for Information Systems, University of Vienna, Austria
(E-mail: {eiter, krennwallner, polleres}@logic.at)
 - ² Department of Mathematics, University of Cyprus, Nicosia, Cyprus
(E-mail: bessi@math.ucy.ac.cy)

Abstract. Rules and ontologies (R/Os) play a key role in Semantic Web applications. In this paper, we discuss the role of R/Os in the Semantic Web and the challenges that arise from their use. We focus on the role of R/Os in the Semantic Web and the challenges that arise from their use. We focus on the role of R/Os in the Semantic Web and the challenges that arise from their use.

1 Introduction

The vision of the Semantic Web is to make it possible for machines to understand the content of the Web. This vision is based on the idea of using a common language to describe the content of the Web. This language is called the Semantic Web language. The Semantic Web language is based on the idea of using a common language to describe the content of the Web. This language is called the Semantic Web language.

¹Excellent tutorial here: <http://www4.wiwiw.fu-berlin.de/bizer/pub/LinkedDataTutorial/>

Where is the data? 1/4

Rules and Ontologies for the Semantic Web¹

- Thomas Eiter¹, Christos Bessis²
- ¹ Institute for Information Systems, University of Vienna, Austria
- ² Department of Informatics, University of Cyprus, Nicosia, Cyprus
- ³ Digital Enterprise, University of Southampton, Southampton, UK

Abstract. Rules and ontologies play a key role in the Semantic Web. In this paper, we review the state-of-the-art in the area of rules and ontologies. We focus on the following topics: (1) the role of rules in the Semantic Web, (2) the role of ontologies in the Semantic Web, (3) the role of rules and ontologies in the Semantic Web, (4) the role of rules and ontologies in the Semantic Web, (5) the role of rules and ontologies in the Semantic Web.

1 Introduction

The use of the Semantic Web is increasing rapidly. It is an important part of the Web 2.0. The Semantic Web is a Web of data. It is a Web of information. It is a Web of knowledge. It is a Web of... (The text continues with a detailed discussion of the Semantic Web and its applications, including a list of references at the bottom.)

¹ Excellent tutorial here: <http://www4.wiwiw.fu-berlin.de/bizer/pub/LinkedDataTutorial/>

Where is the data? 1/4

Rules and Ontologies for the Semantic Web*

- Thomas Eiter, "Consistent Rule Sets for Ontology Reasoning in OWL and OWL-DL", *Journal of Artificial Intelligence Research*, 2004
- Thomas Eiter, "A Reasoning Framework for OWL-DL with Rules", *Journal of Artificial Intelligence Research*, 2005
- Thomas Eiter, "A Reasoning Framework for OWL-DL with Rules", *Journal of Artificial Intelligence Research*, 2005
- Thomas Eiter, "A Reasoning Framework for OWL-DL with Rules", *Journal of Artificial Intelligence Research*, 2005

1. Introduction

The use of the Semantic Web is a key technology for the Semantic Web. It is an extension of the World Wide Web (WWW) that allows data to be shared and reused across different applications, enterprises, and communities. The Semantic Web is based on the use of standards developed by the World Wide Web Consortium (W3C), including the Resource Description Framework (RDF), the Web Ontology Language (OWL), and the SPARQL query language. The Semantic Web is a vision of a web where data is organized and integrated in a way that allows machines to understand and process the information. This is achieved through the use of ontologies, which are formal models of concepts and relationships. The Semantic Web is a key technology for the Semantic Web, and it is the foundation for many other technologies, including the Semantic Web Query Language (SPARQL) and the Semantic Web Services (SWS).

- A lot of Web data already available “out there” in a machine-readable format (RDF)

¹Excellent tutorial here: <http://www4.wiwi.fu-berlin.de/bizer/pub/LinkedDataTutorial/>

Where is the data? 1/4

The collage shows several web pages:

- Knowledge-Based Systems Group**: A website with navigation links like 'staff', 'education', 'research', 'contact', and 'services'.
- Prof. Dr. Thomas Eiter**: A personal profile page for a professor of Computer Science at the University of Vienna, including contact information and a photo.
- DBLP Publications**: A list of academic papers, such as 'Magalhães-Otero, Maria-Sotaria, Thomas Eiter: Worst-case Optimal Conjunctive Query Answering for an Expressive Description Logic without Bounded Axioms'.
- Thomas Krennwallner's Wiki FrontPage**: A personal page with a bio, a list of publications, and a note about not wanting to be known around.

Rules and Ontologies for the Semantic Web*

- Thomas Eiter, "Conceptual Data Models for Knowledge Representation and Query Processing", in: *Handbook of Knowledge Representation*, vol. 4, ed. by F. Heintz, J. Dix, and J. Dix, North-Holland, Amsterdam, 2008, pp. 1-40.
- Thomas Eiter, "Axiomatic Reasoning in Description Logics", in: *Handbook of Knowledge Representation*, vol. 4, ed. by F. Heintz, J. Dix, and J. Dix, North-Holland, Amsterdam, 2008, pp. 41-100.
- Thomas Eiter, "Axiomatic Reasoning in Description Logics", in: *Handbook of Knowledge Representation*, vol. 4, ed. by F. Heintz, J. Dix, and J. Dix, North-Holland, Amsterdam, 2008, pp. 41-100.

1 Introduction

The use of the Semantic Web is a key technology for the Semantic Web. It is an extension of the World Wide Web (WWW) that uses standards from the World Wide Web Consortium (W3C) to describe and link data. The Semantic Web is a vision of a web of data that can be understood by machines. It is a web of data that can be used to create new applications and services. The Semantic Web is a web of data that can be used to create new applications and services. The Semantic Web is a web of data that can be used to create new applications and services.

- A lot of Web data already available “out there” in a machine-readable format (RDF)
- More and more of it follows the *Linked Data* principles¹, i.e.:

¹Excellent tutorial here: <http://www4.wiwi.fu-berlin.de/bizer/pub/LinkedDataTutorial/>

Where is the data? 1/4

Rules and Ontologies for the Semantic Web*

- Thomas (Eiter), Christof Bessler†
- Thomas Krennwallner and Axel Polleres
- † Department of Information Systems, University of Vienna
- † Department of Information Systems, University of Vienna
- † Department of Information Systems, University of Vienna

Abstract. Rules and ontologies for the Semantic Web are the backbone of the Semantic Web. They are used to describe and reason about data. The Semantic Web is a collection of interlinked data sets that can be accessed and processed by machines. The Semantic Web is a collection of interlinked data sets that can be accessed and processed by machines. The Semantic Web is a collection of interlinked data sets that can be accessed and processed by machines.

1. Introduction

The idea of the Semantic Web is to make it possible for machines to understand and process data. The Semantic Web is a collection of interlinked data sets that can be accessed and processed by machines. The Semantic Web is a collection of interlinked data sets that can be accessed and processed by machines.

- A lot of Web data already available “out there” in a machine-readable format (RDF)
- More and more of it follows the *Linked Data* principles¹, i.e.:
 - 1 Use URIs as names for things

¹Excellent tutorial here: <http://www4.wiwi.fu-berlin.de/bizer/pub/LinkedDataTutorial/>

Where is the data? 1/4

Rules and Ontologies for the Semantic Web*

- 1. <http://www.w3.org/2001/sw/> - W3C Semantic Web Working Group
- 2. <http://www.w3.org/2001/sw/ontologies/> - W3C Semantic Web Ontologies
- 3. <http://www.w3.org/2001/sw/rules/> - W3C Semantic Web Rules

1. Introduction

The idea of the Semantic Web is to make it possible for machines to understand and process information on the Web. This is achieved by using a set of standards and protocols that allow data to be described and linked in a machine-readable format. The Semantic Web is a vision of a Web where data is organized and linked in a way that makes it easy for machines to understand and process. This is achieved by using a set of standards and protocols that allow data to be described and linked in a machine-readable format. The Semantic Web is a vision of a Web where data is organized and linked in a way that makes it easy for machines to understand and process. This is achieved by using a set of standards and protocols that allow data to be described and linked in a machine-readable format.

- A lot of Web data already available “out there” in a machine-readable format (RDF)
- More and more of it follows the *Linked Data* principles¹, i.e.:
 - 1 Use URIs as names for things
 - 2 Use HTTP dereferenceable URIs so that people can look up those names.

¹Excellent tutorial here: <http://www4.wiwiw.fu-berlin.de/bizer/pub/LinkedDataTutorial/>

Where is the data? 1/4

Rules and Ontologies for the Semantic Web*

- Thomas Eiter, "Consistent Rule Sets for Knowledge Representation and Query Processing", in: *Handbook of Knowledge Representation*, vol. 4, North-Holland, Amsterdam, 2008, pp. 1-40.
- Thomas Eiter, "A Logic Programming Approach to the Semantic Web", in: *Handbook of Knowledge Representation*, vol. 4, North-Holland, Amsterdam, 2008, pp. 41-70.
- Thomas Eiter, "A Logic Programming Approach to the Semantic Web", in: *Handbook of Knowledge Representation*, vol. 4, North-Holland, Amsterdam, 2008, pp. 41-70.

1 Introduction

The idea of the Semantic Web is to make the data on the Web more machine-readable and thus more useful for computers. This is achieved by using a set of standards called the Semantic Web standards, which are based on the principles of the Semantic Web. The Semantic Web is a vision of a Web where data is represented in a machine-readable format, allowing computers to process and analyze the data automatically. This is achieved by using a set of standards called the Semantic Web standards, which are based on the principles of the Semantic Web. The Semantic Web is a vision of a Web where data is represented in a machine-readable format, allowing computers to process and analyze the data automatically. This is achieved by using a set of standards called the Semantic Web standards, which are based on the principles of the Semantic Web.

- A lot of Web data already available “out there” in a machine-readable format (RDF)
- More and more of it follows the *Linked Data* principles¹, i.e.:
 - 1 Use URIs as names for things
 - 2 Use HTTP dereferenceable URIs so that people can look up those names.
 - 3 When someone looks up a URI, provide useful information.

¹Excellent tutorial here: <http://www4.wiwiw.fu-berlin.de/bizer/pub/LinkedDataTutorial/>

The screenshot shows a web browser window with the following content:

- Address Bar:** <http://www.kbs.wu-wi.at/kbaff/teier/>
- Page Header:** TU W I E N K B S Knowledge-Based Systems Group
- Navigation:** staff education research contact services
- Content:**
 - Prof. Dr. Thomas Eiter
 - Professor of Computer Science
 - Head of the [Knowledge-Based Systems Group](#)
 - Head of the [Institute of Information Systems IV](#)
 - SPRAC: office for human and machine communication
 - PHONE: +43 (0)1 85 887-1840
 - FAX: +43 (0)1 85 887-1845
 - MOB: +43 (0)1 85 887-1845
 - OFFICE/HOURS: THU 17:00 - 18:00
 - Research interests:
- Publications:**
 - DBLP publications from the [DBLP Bibliography Service](#) - EAQ
 - Computer Science - Ask others: ACM-DB, Google - CiteSeer - CSE - Google - MSN - Yahoo
 - [Home Page](#)
- Research Interests:**
 - Thomas Krennwallner's Wiki
 - Formal Prolog
 - 2004
 - 211. Magdalena Orla, Maria Stojan, Thomas Eiter: Classifying in Optimal Conjunctive Query Answering for an Expressive Description Logic without Boundedness. AAAI-2004, 864-819
 - 212. Thomas Eiter, Magdalena Orla, Jan Stokke: Error Warnings in Answer Disjunctive A Hornlike Approach. AAAI-2004, 850-810
 - 229. Magdalena Orla, Maria Stojan, Thomas Eiter: Conjunctive Query Answering in SH using DL. Dagstuhl Seminar
 - 231. Thomas Eiter, Magdalena Orla, Ursula Gloger, Dominik Leifer: Repair Involutions for query answering from incomplete databases. ACM-THIN, Dublin, 2004, 29-31
 - 232. Knowledge-based Data Mining: Thomas Eiter: Knowledge-based Data Mining: A knowledge-based intelligent reasoning system for expert representation/management. Advances in Intelligent Systems, 2004, 931-937
 - 233. Y. Chen, Thomas Eiter, Maria Stojan, Magdalena Orla: Maintaining good of agents in a dynamic environment: Formalism and policy construction. Artif. Intell. 172(1-2): 1429-1469 (2004)
 - 234. Thomas Eiter, Gabor Krennwallner, Thomas Leifer, Roman Stollmann: Horn Fragments of Conjunctive Query Answering in SH with Disjunctive Horns for the Semantic Web. Artif. Intell. 172(1-2): 1485-1508 (2004)
 - 235. Thomas Eiter, Roman Stollmann: Semantic Inference in Answer set programming. Artif. Intell. 164: 147-172 (2004)
 - 236. Thomas Eiter, Jan Stokke, Wolfgang Leifer: Answering the effects of action sequences. J. Applied Logic 1(3): 381-417 (2004)

- [illegible]

4/45

Where is the data? 2/4

Obtaining Machine-Readable RDF data

(i) directly by the publishers, (ii) by GRDDL transformations, or (iii) by 3rd-party wrappers:

Where is the data? 2/4

Obtaining Machine-Readable RDF data

(i) **directly by the publishers**, (ii) by GRDDL transformations, or (iii) by 3rd-party wrappers:

FOAF/RDF linked from a home page: personal data (foaf:name, foaf:phone, etc.), relationships foaf:knows, rdfs:seeAlso)

The image shows two side-by-side browser windows. The left window displays a personal homepage for G.B. Ianni, an Assistant Professor at the Dipartimento di Matematica, Cubo 30B, Università della Calabria. The page includes a photo, contact information, and links to publications and teaching activities. A red circle highlights the 'My FOAF card' link, with a red arrow pointing to the right-hand window. The right-hand window displays the raw RDF data for the FOAF card, showing various properties like foaf:name, foaf:phone, and foaf:knows.

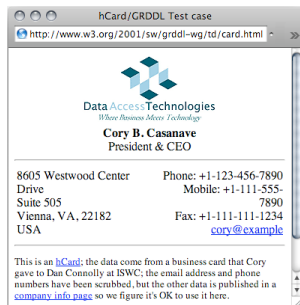
Where is the data? 3/4

Obtaining Machine-Readable RDF data

(i) directly by the publishers, (ii) by **GRDDL transformations**, or (iii) by 3rd-party wrappers: GRDDL (**G**leaning **R**esource **D**escriptions from **D**ialects of **L**anguages.) [Connolly (ed.), 2007]

Simple principle:

- extract RDF directly from HTML or XML files
- typically using XSLT transformations (other languages: XQuery, XSPARQL, etc.)
- useful for common Microformats 🟢, e.g. hCard, hCal:




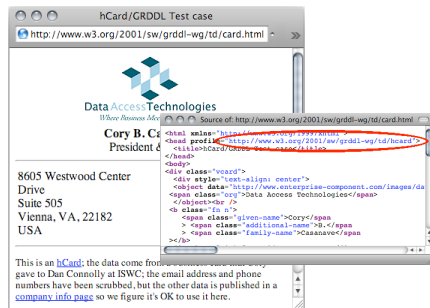
Where is the data? 3/4

Obtaining Machine-Readable RDF data

(i) directly by the publishers, (ii) by **GRDDL transformations**, or (iii) by 3rd-party wrappers: GRDDL (**G**leaning **R**esource **D**escriptions from **D**ialects of **L**anguages.) [Connolly (ed.), 2007]

Simple principle:

- extract RDF directly from HTML or XML files
- typically using XSLT transformations (other languages: XQuery, XSPARQL, etc.)
- useful for common Microformats , e.g. hCard, hCal:




- profile `http://www.w3.org/2001/sw/grddl-wg/td/hcard ...`

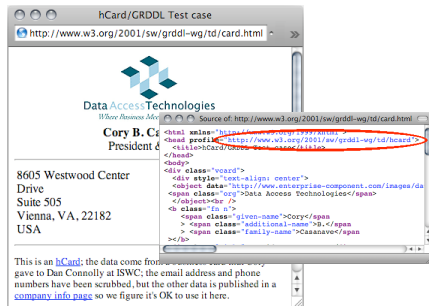
Where is the data? 3/4

Obtaining Machine-Readable RDF data

(i) directly by the publishers, (ii) by **GRDDL transformations**, or (iii) by 3rd-party wrappers: GRDDL (Gleaning Resource Descriptions from Dialects of Languages.) [Connolly (ed.), 2007]

Simple principle:

- extract RDF directly from HTML or XML files
- typically using XSLT transformations (other languages: XQuery, XSPARQL, etc.)
- useful for common Microformats , e.g. hCard, hCal:



hCard/GRDDL Test case

http://www.w3.org/2001/sw/grddl-wg/td/card.html

Data Access Technologies

Where Business Meets Technology

Cory B. Casanova
President & CEO

8605 Westwood Center
Drive
Suite 505
Vienna, VA, 22182
USA

This is an [hCard](#); the data come from a company info page that Dan Connolly at ISWC; the email address and phone numbers have been scrubbed, but the other data is published in a [company info page](#) so we figure it's OK to use it here.

- profile <http://www.w3.org/2001/sw/grddl-wg/td/hcard> ...
- ... points to XSL transformation <http://www.w3.org/2006/vcard/hcard2rdf.xsl>

Where is the data? 3/4

Obtaining Machine-Readable RDF data

- (i) directly by the publishers, (ii) by GRDDL transformations, or (iii) by 3rd-party wrappers:

GRDDL (**G**leaning **R**esource **D**escriptions from **D**ialects of **L**anguages.) [Connolly (ed.), 2007]

Simple principle:

- extract RDF directly from HTML or XML files
- typically using XSLT transformations (other languages: XQuery, XSPARQL, etc.)
- useful for common Microformats 🟢, e.g. hCard, hCal:

[illegible]

- **profile** <http://www.w3.org/2001/sw/grddl-wg/td/hcard> ...
- ...points to XSL transformation <http://www.w3.org/2006/vcard/hcard2rdf.xsl>
- ...which – executed on the original HTML file – extracts RDF.

Where is the data? 4/4

Obtaining Machine-Readable RDF data

(i) directly by the publishers, (ii) by GRDDL transformations, or (iii) by 3rd-party wrappers:

²<http://dblp.13s.de/d2r/>

Where is the data? 4/4

Obtaining Machine-Readable RDF data

(i) directly by the publishers, (ii) by GRDDL transformations, or (iii) by 3rd-party wrappers:

L3S' RDF export of the DBLP citation index:²

The left screenshot shows the DBLP website profile for Thomas Eiter. The right screenshot shows the same profile page with an RDF export view, displaying a table of properties and values in RDF format. A red arrow points from the left screenshot to the right one.

Property	Value
is:co:creator of	<http://dblp.l3s.de/d2r/resource/publications/book/mr/Subrahmanian2000>
is:co:creator of	<http://dblp.l3s.de/d2r/resource/publications/conf/aaai/BanaIEZ05>
is:co:creator of	<http://dblp.l3s.de/d2r/resource/publications/conf/aaai/BrewkaE07>
is:co:creator of	<http://dblp.l3s.de/d2r/resource/publications/conf/aaai/CalvaneseEC07>
is:co:creator of	<http://dblp.l3s.de/d2r/resource/publications/conf/aaai/EgyGTY00>
is:co:creator of	<http://dblp.l3s.de/d2r/resource/publications/conf/aaai/EnteFS08>
is:co:creator of	<http://dblp.l3s.de/d2r/resource/publications/conf/aaai/EnteFTW05>
is:co:creator of	<http://dblp.l3s.de/d2r/resource/publications/conf/aaai/EnteM08>
is:co:creator of	<http://dblp.l3s.de/d2r/resource/publications/conf/aaai/EnteM02>
is:co:creator of	<http://dblp.l3s.de/d2r/resource/publications/conf/aaai/EnteW06>
is:co:creator of	<http://dblp.l3s.de/d2r/resource/publications/conf/aaai/OrtizCE06>
is:co:creator of	<http://dblp.l3s.de/d2r/resource/publications/conf/aaai/OrtizS08>
is:co:creator of	<http://dblp.l3s.de/d2r/resource/publications/conf/aaai/OrtizS09>
is:co:creator of	<http://dblp.l3s.de/d2r/resource/publications/conf/aaai/OrtizS06>
is:co:creator of	<http://dblp.l3s.de/d2r/resource/publications/conf/aaai/OrtizS03>
is:co:creator of	<http://dblp.l3s.de/d2r/resource/publications/conf/aaai/OrtizS04>

²<http://dblp.l3s.de/d2r/>

Where is the data? 4/4

Obtaining Machine-Readable RDF data

(i) directly by the publishers, (ii) by GRDDL transformations, or (iii) by 3rd-party wrappers:

L3S' RDF export of the DBLP citation index:²

The left screenshot shows the DBLP profile for Thomas Eiter. The right screenshot shows the corresponding RDF export page, which lists various properties and their values as URIs. A red arrow points from the left page to the right page.

Thomas Eiter
Resource URI: http://dblp.l3s.de/d2r/resource/authors/Thomas_Eiter

Property	Value
is:co:creator of	< http://dblp.l3s.de/d2r/resource/publications/books/mr/Subrahmanian2000 >
is:co:creator of	< http://dblp.l3s.de/d2r/resource/publications/conf/aaai/Banaiz2005 >
is:co:creator of	< http://dblp.l3s.de/d2r/resource/publications/conf/aaai/BrewkaE07 >
is:co:creator of	< http://dblp.l3s.de/d2r/resource/publications/conf/aaai/CalvaneseEC07 >
is:co:creator of	< http://dblp.l3s.de/d2r/resource/publications/conf/aaai/EgyGFW00 >
is:co:creator of	< http://dblp.l3s.de/d2r/resource/publications/conf/aaai/EiterFWS08 >
is:co:creator of	< http://dblp.l3s.de/d2r/resource/publications/conf/aaai/EiterFW05 >
is:co:creator of	< http://dblp.l3s.de/d2r/resource/publications/conf/aaai/EiterM08 >
is:co:creator of	< http://dblp.l3s.de/d2r/resource/publications/conf/aaai/EiterM02 >
is:co:creator of	< http://dblp.l3s.de/d2r/resource/publications/conf/aaai/EiterM06 >
is:co:creator of	< http://dblp.l3s.de/d2r/resource/publications/conf/aaai/OrtizCE06 >
is:co:creator of	< http://dblp.l3s.de/d2r/resource/publications/conf/aaai/OrtizSE07 >
is:co:creator of	< http://dblp.l3s.de/d2r/resource/publications/conf/aaai/OrtizMP07 >
is:co:creator of	< http://dblp.l3s.de/d2r/resource/publications/conf/aaai/OrtizMP09 >
is:co:creator of	< http://dblp.l3s.de/d2r/resource/publications/conf/aaai/OrtizMP04 >

- Gives unique URIs to authors, documents, etc. on DBLP! E.g.,
http://dblp.l3s.de/d2r/page/authors/Thomas_Eiter,
<http://dblp.l3s.de/d2r/page/publications/conf/rweb/EiterIKP08>
- Provides RDF version of all DBLP data + query interface!

²<http://dblp.l3s.de/d2r/>

How can I query that data? SPARQL

SPARQL – W3C approved standardized query language for RDF:

- look-and-feel of “SQL for the Web”
- allows to ask queries like
- *“All documents created by Thomas Eiter”*
- *“Names of all persons who co-authored with authors of the present paper”*
- *“Names of persons who know Axel Polleres or who are known by Axel Polleres”*
- *“All people who have published in the Reasoning Web Series but have not co-authored with any of the authors of the present paper”*

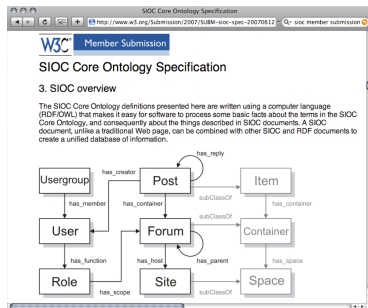
Example:

```
SELECT ?D
FROM <http://dblp.13s.de/d2r/page/authors/Thomas_Eiter>
WHERE {?D dc:creator <http://dblp.13s.de/d2r/resource/authors/Thomas_Eiter>}
```


What does the data mean? 2/2

Data, i.e. the used *vocabulary* to write down RDF is described by *ontologies*, themselves published in RDF, e.g.:

- Friend-of-a-Friend (FOAF) [Brickley and Miller, 2007]
- Socially-Interlinked-Online-Communities (SIOC) [Bojārs *et al.*, 2007]
- Dublin Core [Nilsson *et al.*, 2008]



Unit Outline

1. Motivation – Aggregating Linked Open Data by Rules
2. How can I publish data? RDF
3. How can I query that data? SPARQL
4. What does that data mean? Ontologies described in RDFS + OWL
5. What's next?

Semantic Web Data: The Resource Description Framework (RDF)

- RDF is describing *resources* per triples/statements
Subject Predicate Object.

Semantic Web Data: The Resource Description Framework (RDF)

- RDF is describing *resources* per triples/statements
Subject Predicate Object.
- “simplest possible database schema”, data just a set of triples:

Semantic Web Data: The Resource Description Framework (RDF)

- RDF is describing *resources* per triples/statements

Subject Predicate Object.

- “simplest possible database schema”, data just a set of triples:

axel isA Person .

axel hasName “Axel Polleres”.

axel knows gb .

axel knows thomas.

thomas hasCreated an Article

titled “Rules and Ontologies for the Semantic Web”.

Semantic Web Data: The Resource Description Framework (RDF)

- RDF is describing *resources* per triples/statements

Subject Predicate Object.

- “simplest possible database schema”, data just a set of triples:

axel isA Person .

axel hasName “Axel Polleres”.

axel knows gb .

axel knows thomas.

∃X thomas hasCreated X . X isA Article .

X hasTitle “Rules and Ontologies for the Semantic Web”.

Semantic Web Data: The Resource Description Framework (RDF)

- RDF is describing *resources* per triples/statements

Subject Predicate Object.

- “simplest possible database schema”, data just a set of triples:

axel isA Person .

axel hasName “Axel Polleres”.

axel knows gb .

axel knows thomas.

$\exists X$ *thomas hasCreated X . X isA Article .*

X hasTitle “Rules and Ontologies for the Semantic Web”.

- abstracts away from tables (RDBMS) or tree-like (XML) schemas

Semantic Web Data: The Resource Description Framework (RDF)

- RDF is describing *resources* per triples/statements

Subject Predicate Object.

- “simplest possible database schema”, data just a set of triples:

axel isA Person .

axel hasName “Axel Polleres”.

axel knows gb .

axel knows thomas.

$\exists X$ *thomas hasCreated X . X isA Article .*

X hasTitle “Rules and Ontologies for the Semantic Web”.

- abstracts away from tables (RDBMS) or tree-like (XML) schemas
- triples can be viewed as edges of a labeled,directed graph.

Semantic Web Data: The Resource Description Framework (RDF)

- RDF is describing *resources* per triples/statements

Subject Predicate Object.

- “simplest possible database schema”, data just a set of triples:

axel isA Person .

axel hasName “Axel Polleres”.

axel knows gb .

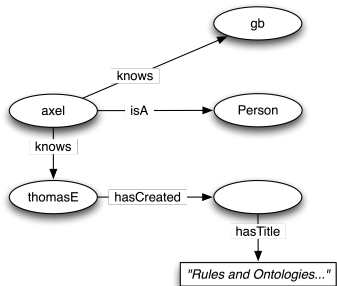
axel knows thomas.

$\exists X$ *thomas hasCreated X . X isA Article .*

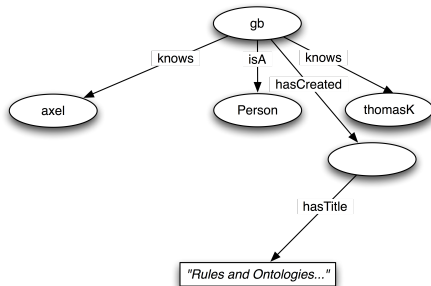
X hasTitle “Rules and Ontologies for the Semantic Web”.

- abstracts away from tables (RDBMS) or tree-like (XML) schemas
- triples can be viewed as edges of a labeled,directed graph.
- main advantage: Graphs are easy to merge! (Trees,Tables aren't)

axel isA Person .
 axel knows gb .
 axel knows thomasE.
 thomasE hasCreated X . X isA Article .
 X hasTitle "Rules and Ontologies..." .



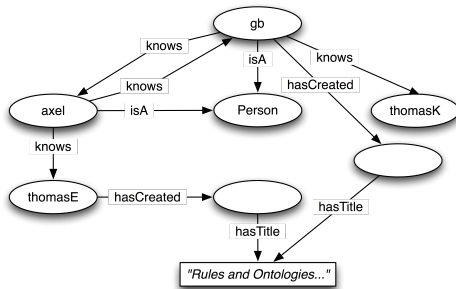
gb isA Person .
 gb knows axel .
 gb knows thomasK.
 gb hasCreated Y . Y isA Article .
 Y hasTitle "Rules and Ontologies..." .



Observe: the "existential variables" became "blank" nodes in the Graph.

axel isA Person .
 axel knows gb .
 axel knows thomasE.
 thomasE hasCreated X . X isA Article .
 X hasTitle "Rules and Ontologies..." .

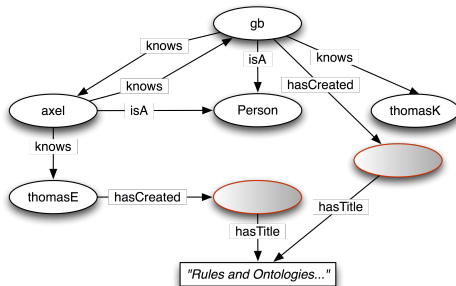
gb isA Person .
 gb knows axel .
 gb knows thomasK.
 gb hasCreated Y . Y isA Article .
 Y hasTitle "Rules and Ontologies..." .



Observe: the "existential variables" became "blank" nodes in the Graph.

axel isA Person .
 axel knows gb .
 axel knows thomasE.
 thomasE hasCreated **X** . **X** isA Article .
X hasTitle "Rules and Ontologies..." .

gb isA Person .
 gb knows axel .
 gb knows thomasK.
 gb hasCreated **Y** . **Y** isA Article .
Y hasTitle "Rules and Ontologies..." .



Observe: the "existential variables" became "blank" nodes in the Graph. **Note that we have no reason to assume that the two blank nodes are the same.**

Resources in RDF, Syntax

- Resources are identified by URIs (to encourage web-wide unique identifiers)
- There are special URIs, defined in vocabularies (FOAF, SIOC, RDF, etc.)
- Objects can be literals,

axel isA Person .

axel hasName "Axel Polleres".

Resources in RDF, Syntax

- Resources are identified by URIs (to encourage web-wide unique identifiers)
- There are special URIs, defined in vocabularies (FOAF, SIOC, RDF, etc.)
- Objects can be literals,

axel isA Person .

axel hasName "Axel Polleres".

becomes:

```
<http://polleres.net/foaf.rdf#me> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>  
    <http://xmlns.com/foaf/0.1/Person>.  
<http://polleres.net/foaf.rdf#me> <http://xmlns.com/foaf/0.1/name>  
    "Axel Polleres".
```


Resources in RDF, Syntax

- Resources are identified by URIs (to encourage web-wide unique identifiers)
- There are special URIs, defined in vocabularies (FOAF, SIOC, RDF, etc.)
- Objects can be literals, **occasionally with a datatype**

axel isA Person .

axel hasName "Axel Polleres".

becomes:

```
<http://polleres.net/foaf.rdf#me> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>  
    <http://xmlns.com/foaf/0.1/Person>.  
<http://polleres.net/foaf.rdf#me> <http://xmlns.com/foaf/0.1/name>  
    "Axel Polleres"^^<http://www.w3.org/2001/XMLSchema#string>.
```


Resources in RDF, Syntax

- Resources are identified by URIs (to encourage web-wide unique identifiers)
- There are special URIs, defined in vocabularies (FOAF, SIOC, RDF, etc.)
- Objects can be literals, occasionally with a datatype

axel isA Person .

axel hasName "Axel Polleres".

becomes:

```
<http://polleres.net/foaf.rdf#me> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
    <http://xmlns.com/foaf/0.1/Person>.
<http://polleres.net/foaf.rdf#me> <http://xmlns.com/foaf/0.1/name>
    "Axel Polleres"^^<http://www.w3.org/2001/XMLSchema#string>.
```

Ugly to read... more compact syntaxes like Turtle [Beckett, 2007] allow prefix definitions á la XML:

```
@prefix : <http://polleres.net/foaf.rdf#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix xs: <http://www.w3.org/2001/XMLSchema#> .
:me rdf:type foaf:Person .
:me foaf:name "Axel Polleres"^^xs:string.
```


More on RDF – Shortcuts in Turtle Syntax

```
@prefix : <http://polleres.net/foaf.rdf#>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .

:me rdf:type foaf:Person .
:me foaf:name "Axel Polleres" .
:me foaf:knows <http://dblp.l3s.de/d2r/data/authors/Giovambattista_Ianni> .
:me foaf:knows <http://dblp.l3s.de/d2r/page/authors/Thomas_Eiter> .
<http://dblp.l3s.de/d2r/page/authors/Thomas_Eiter> dc:creator X .
X rdf:type foaf:Document .
X dc:title "Rules and Ontologies for the Semantic Web".
```


More on RDF – Shortcuts in Turtle Syntax

```
@prefix : <http://polleres.net/foaf.rdf#>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .

:me rdf:type foaf:Person .
:me foaf:name "Axel Polleres" .
:me foaf:knows <http://dblp.l3s.de/d2r/data/authors/Giovambattista_Ianni> .
:me foaf:knows <http://dblp.l3s.de/d2r/page/authors/Thomas_Eiter> .
<http://dblp.l3s.de/d2r/page/authors/Thomas_Eiter> dc:creator _:X .
_:X rdf:type foaf:Document .
_:X dc:title "Rules and Ontologies for the Semantic Web".
```

- Blank nodes in Turtle are written as `_:Varname`

More on RDF – Shortcuts in Turtle Syntax

```
@prefix : <http://polleres.net/foaf.rdf#>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .

:me rdf:type foaf:Person ;
    foaf:name "Axel Polleres" ;
    foaf:knows <http://dblp.l3s.de/d2r/data/authors/Giovambattista_Ianni> ,
               <http://dblp.l3s.de/d2r/page/authors/Thomas_Eiter> .
<http://dblp.l3s.de/d2r/page/authors/Thomas_Eiter> dc:creator _:X .
_:X rdf:type foaf:Document ;
    dc:title "Rules and Ontologies for the Semantic Web" .
```

- Blank nodes in Turtle are written as `_:Varname`
- Turtle allows shortcuts:
 - Same subject triples can be grouped together with `';' , ','`

More on RDF – Shortcuts in Turtle Syntax

```

@prefix : <http://polleres.net/foaf.rdf#>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .

:me rdf:type foaf:Person;
    foaf:name "Axel Polleres" ;
    foaf:knows <http://dblp.l3s.de/d2r/data/authors/Giovambattista_Ianni> ,
               <http://dblp.l3s.de/d2r/page/authors/Thomas_Eiter> .
<http://dblp.l3s.de/d2r/page/authors/Thomas_Eiter> dc:creator [
    rdf:type foaf:Document ;
    dc:title "Rules and Ontologies for the Semantic Web" ] .

```

- Blank nodes in Turtle are written as `_:Varname`
- Turtle allows shortcuts:
 - Same subject triples can be grouped together with `','`
 - Blank nodes can be grouped/replaced using "bracket syntax" `'[, ']`

More on RDF – Shortcuts in Turtle Syntax

```
@prefix : <http://polleres.net/foaf.rdf#>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .

:me a foaf:Person;
    foaf:name "Axel Polleres" ;
    foaf:knows <http://dblp.13s.de/d2r/data/authors/Giovambattista_Ianni> ,
               <http://dblp.13s.de/d2r/page/authors/Thomas_Eiter> .
<http://dblp.13s.de/d2r/page/authors/Thomas_Eiter> dc:creator [
    a foaf:Document ;
    dc:title "Rules and Ontologies for the Semantic Web" ] .
```

- Blank nodes in Turtle are written as `_:Varname`
- Turtle allows shortcuts:
 - Same subject triples can be grouped together with `',' , ','`
 - Blank nodes can be grouped/replaced using "bracket syntax" `'[, ']`
 - `rdf:type` is often abbreviated with `a`.

Collecting RDF from the Web

- For us this is enough so far to “read” RDF on the Web.

³<http://librdf.org/>

⁴<http://jena.sourceforge.net/>

Collecting RDF from the Web

- For us this is enough so far to “read” RDF on the Web.
- For published RDF data there exists a machine-readable XML syntax. Lots of tools and APIs to read/process/convert this data (Redland (C++),³ Jena (Java),⁴ etc.)

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
<http://www.mat.unical.it/~ianni/foaf.rdf> a foaf:PersonalProfileDocument .
<http://www.mat.unical.it/~ianni/foaf.rdf> foaf:maker _:me .
<http://www.mat.unical.it/~ianni/foaf.rdf> foaf:primaryTopic _:me .
_:me a foaf:Person .
_:me foaf:name "Giovambattista Ianni" .
_:me foaf:homepage <http://www.gibbi.com> .
_:me foaf:phone <tel:+39-0984-496430> .
_:me foaf:knows [ a foaf:Person ;
                  foaf:name "Wolfgang Faber" ;
                  rdfs:seeAlso <http://www.kr.tuwien.ac.at/staff/faber/foaf.rdf> ] .
_:me foaf:knows [ a foaf:Person .
                  foaf:name "Axel Polleres" ;
                  rdfs:seeAlso <http://www.polleres.net/foaf.rdf> ] .
_:me foaf:knows [ a foaf:Person .
                  foaf:name "Thomas Eiter" ] .
_:me foaf:knows [ a foaf:Person .
                  foaf:name "Alessandra Martello" ] .
```

³<http://librdfs.org/>

⁴<http://jena.sourceforge.net/>

Collecting RDF from the Web

- For us this is enough so far to “read” RDF on the Web.
- For published RDF data there exists a machine-readable XML syntax. Lots of tools and APIs to read/process/convert this data (Redland (C++),³ Jena (Java),⁴ etc.)

@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
 @prefix foaf: <http://xmlns.com/foaf/0.1/> .

foaf:name "Alessandra Martello"] .

³<http://librdf.org/>

⁴<http://jena.sourceforge.net/>

Unit Outline

1. Motivation – Aggregating Linked Open Data by Rules
2. How can I publish data? RDF
3. How can I query that data? SPARQL
4. What does that data mean? Ontologies described in RDFS + OWL
5. What's next?

How can I query/aggregate RDF data? SPARQL

- First “ingredient”: a standardized query language – SPARQL [Prud’hommeaux and Seaborne, 2007] – based on graph pattern matching

Prologue:	P	prefix <i>prefix</i> : <namespace-URI>
Head:	C S A	construct { <i>template</i> } select <i>variable list</i> ask
Body:	D W M	from / from named <dataset-URI> where { <i>pattern</i> } order by <i>expression</i> limit <i>integer</i> > 0 offset <i>integer</i> > 0

...construct a new RDF graph
...select matching resources from a graph
...boolean query

How can I query/aggregate RDF data? SPARQL

- First “ingredient”: a standardized query language – SPARQL [Prud’hommeaux and Seaborne, 2007] – based on graph pattern matching

Prologue:	P	prefix <i>prefix</i> : <namespace-URI>
Head:	C	construct { <i>template</i> }
	S	select <i>variable list</i>
	A	ask
Body:	D	from / from named <dataset-URI>
	W	where { <i>pattern</i> }
	M	order by <i>expression</i>
		limit <i>integer</i> > 0
		offset <i>integer</i> > 0

...construct a new RDF graph
 ...select matching resources in a graph
 ...boolean query

- Let us start with select queries and focus on the different **patterns**:
 - basic graph patterns (Conjunctive queries)
 - filters
 - unions of patterns
 - optional Patterns
 - graph Patterns

Basic Graph Patterns (Conjunctive queries)

“select all names of persons known by G.B. from his FOAF file”

```
prefix foaf: <http://xmlns.com/foaf/0.1/>
select ?N
from <http://www.mat.unical.it/~ianni/foaf.rdf>
where {
    <http://www.mat.unical.it/~ianni/foaf.rdf#me> foaf:knows ?X .
    ?X a foaf:Person .   ?X foaf:name ?N .
}
```

- graph patterns (where part) allow Turtle syntax

⁵We assume here that the only people declared “known” in G.B.’s FOAF file are those known by him.

Basic Graph Patterns (Conjunctive queries)

“select all names of persons known by G.B. from his FOAF file”

```
prefix foaf: <http://xmlns.com/foaf/0.1/>
select ?N
from <http://www.mat.unical.it/~ianni/foaf.rdf>
where {
  [ foaf:knows
    [ a foaf:Person; foaf:name ?N ] ]
}
```

- graph patterns (where part) allow Turtle syntax
- all Turtle shortcuts allowed⁵

⁵We assume here that the only people declared “known” in G.B.’s FOAF file are those known by him.

Basic Graph Patterns (Conjunctive queries)

“select all names of persons known by G.B., Axel, and Thomas K. from their FOAF files”

```
prefix foaf: <http://xmlns.com/foaf/0.1/>
select ?N
from <http://www.mat.unical.it/~ianni/foaf.rdf>
from <http://www.polleres.net/foaf.rdf>
from <http://www.postsubmeta.net/foaf>
where {
    [ foaf:knows
      [ a foaf:Person; foaf:name ?N ]]
}
```

- graph patterns (where part) allow Turtle syntax
- all Turtle shortcuts allowed⁵
- merge of several graphs can be queried at once

⁵We assume here that the only people declared “known” in G.B.’s FOAF file are those known by him.

FILTERs in Basic Graph Patterns

“select all names of persons known by the authors from their FOAF files”

```
prefix foaf: <http://xmlns.com/foaf/0.1/>
select ?N
where {
  [ foaf:knows
    [a foaf:Person ; foaf:name ?N] ]
}
```

- graph patterns (where part) allow Turtle syntax
- all Turtle shortcuts allowed
- Dataset can also be implicit, depending on the implementation. . .
so, let's assume we have a Web crawl of FOAF data . . .

FILTERs in Basic Graph Patterns

“select all names of persons known by the authors from their FOAF files”

```
prefix foaf: <http://xmlns.com/foaf/0.1/>
select ?N
where {
  [ foaf:knows
    [a foaf:Person ; foaf:name ?N] ]
  filter ( ?N != "Thomas Eiter" && ?N != "Giovambattista Ianni"
          && ?N != "Thomas Krennwallner" && ?N != "Axel Polleres" )
}
```

- graph patterns (where part) allow Turtle syntax
- all Turtle shortcuts allowed
- Dataset can also be implicit, depending on the implementation...
so, let's assume we have a Web crawl of FOAF data ...
- ...i.e., we have to filter out the authors' names from the result.

UNIONs

*"Names of persons who know Axel Polleres **or** who are known by Axel Polleres"*

```
prefix foaf: <http://xmlns.com/foaf/0.1/>
select ?N
from ...
where {
  { [ foaf:name "Axel Polleres" ] foaf:knows [foaf:name ?N ] }
  union
  { [ foaf:name ?N ] foaf:knows [foaf:name "Axel Polleres" ] }
}
```


UNIONs

*"Names of persons who know Axel Polleres **or** who are known by Axel Polleres"*

```
prefix foaf: <http://xmlns.com/foaf/0.1/>
select ?N
from ...
where {
  { [ foaf:name "Axel Polleres" ] foaf:knows [foaf:name ?N ] }
  union
  { [ foaf:name ?N ] foaf:knows [foaf:name "Axel Polleres" ] }
}
```

- **union** s allow alternative matching of several patterns, similar to UNIONs in SQL.

OPTIONALs 1/2 – Partial Matching

“Select all names of persons known by G.B. from his FOAF file and – if available – their `rdfs:seeAlso` links”

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix foaf: <http://xmlns.com/foaf/0.1/>

select ?N ?L
from <http://www.mat.unical.it/~ianni/foaf.rdf>
where {<http://www.mat.unical.it/~ianni/foaf.rdf#me> foaf:knows ?X .
      ?X foaf:name ?N . ?X rdfs:seeAlso ?L
}
```

- “Normal” basic graph pattern doesn’t work here, returns only those ?X with a `rdfs:seeAlso` link.

?N	?L
"Wolfgang Faber"	<http://www.kr.tuwien.ac.at/staff/faber/foaf.rdf>
"Axel Polleres"	<http://www.polleres.net/foaf.rdf>

OPTIONALS 1/2 – Partial Matching

“Select all names of persons known by G.B. from his FOAF file and – if available – their `rdfs:seeAlso` links”

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix foaf: <http://xmlns.com/foaf/0.1/>

select ?N ?L
from <http://www.mat.unical.it/~ianni/foaf.rdf>
where {<http://www.mat.unical.it/~ianni/foaf.rdf#me> foaf:knows ?X .
      ?X foaf:name ?N . optional { ?X rdfs:seeAlso ?L }
}
```

- “Normal” basic graph pattern doesn't work here, returns only those ?X with a `rdfs:seeAlso` link.
- optional allows **partial variable bindings** in the solutions.

?N	?L
"Wolfgang Faber"	<http://www.kr.tuwien.ac.at/staff/faber/foaf.rdf>
"Axel Polleres"	<http://www.polleres.net/foaf.rdf>
"Thomas Eiter"	
"Alessandra Martello"	

OPTIONALs 2/2 – Set difference

“Select all names of persons known by G.B. from his FOAF file who don't have a `rdfs:seeAlso` links”

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix foaf: <http://xmlns.com/foaf/0.1/>

select ?N
from <http://www.mat.unical.it/~ianni/foaf.rdf>
where {<http://www.mat.unical.it/~ianni/foaf.rdf#me> foaf:knows ?X .
      ?X foaf:name ?N . optional { ?X rdfs:seeAlso ?L }
      filter ( ! bound(L) )
    }
```

- optional allows partial variable bindings in the solutions.

OPTIONALS 2/2 – Set difference

“Select all names of persons known by G.B. from his FOAF file who don't have a `rdfs:seeAlso` links”

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix foaf: <http://xmlns.com/foaf/0.1/>

select ?N
from <http://www.mat.unical.it/~ianni/foaf.rdf>
where {<http://www.mat.unical.it/~ianni/foaf.rdf#me> foaf:knows ?X .
      ?X foaf:name ?N . optional { ?X rdfs:seeAlso ?L }
      filter ( ! bound(L) )
}
```

- optional allows partial variable bindings in the solutions.
- The negated `bound()` function in the filter allows to suppress unbound values.

?N
"Thomas Eiter"
"Alessandra Martello"

OPTIONALS 2/2 – Set difference

“Select all names of persons known by G.B. from his FOAF file who don't have a `rdfs:seeAlso` links”

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix foaf: <http://xmlns.com/foaf/0.1/>

select ?N
from <http://www.mat.unical.it/~ianni/foaf.rdf>
where {<http://www.mat.unical.it/~ianni/foaf.rdf#me> foaf:knows ?X .
      ?X foaf:name ?N . optional { ?X rdfs:seeAlso ?L }
      filter ( ! bound(L) )
}
```

- `optional` allows partial variable bindings in the solutions.
- The negated `bound()` function in the `filter` allows to suppress unbound values.
- This is similar to set difference (NOT EXISTS) in SQL or “negation as failure” in Logic Programming rules.

?N
"Thomas Eiter"
"Alessandra Martello"

OPTIONALS 2/2 – Set difference

“Select all names of persons known by G.B. from his FOAF file who don't have a `rdfs:seeAlso` links”

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix foaf: <http://xmlns.com/foaf/0.1/>

select ?N
from <http://www.mat.unical.it/~ianni/foaf.rdf>
where {<http://www.mat.unical.it/~ianni/foaf.rdf#me> foaf:knows ?X .
      ?X foaf:name ?N . optional { ?X rdfs:seeAlso ?L }
      filter ( ! bound(L) )
}
```

- optional allows partial variable bindings in the solutions.
- The negated bound() function in the filter allows to suppress unbound values.
- This is similar to set difference (NOT EXISTS) in SQL or “negation as failure” in Logic Programming rules.
- Many more useful filter functions available in SPARQL

?N
"Thomas Eiter"
"Alessandra Martello"

GRAPH patterns

“Select all names of persons directly known by G.B. or the names of persons appearing in the files linked by `rdfs:seeAlso` links.”

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix foaf: <http://xmlns.com/foaf/0.1/>

select ?N
from <http://www.mat.unical.it/~ianni/foaf.rdf>
where {<http://www.mat.unical.it/~ianni/foaf.rdf#me> foaf:knows ?X .
      { { ?X foaf:name ?N .}
        union
        { ?X rdfs:seeAlso ?L . graph ?L{ [a foaf:Person ] foaf:name ?N } }
      }
```

- named **graph** patterns allow to match pattern in remote graphs

GRAPH patterns

“Select all names of persons directly known by G.B. or the names of persons appearing in the files linked by `rdfs:seeAlso` links.”

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix foaf: <http://xmlns.com/foaf/0.1/>

select ?N
from <http://www.mat.unical.it/~ianni/foaf.rdf>
from named???
where {<http://www.mat.unical.it/~ianni/foaf.rdf#me> foaf:knows ?X .
      { { ?X foaf:name ?N .}
        union
        { ?X rdfs:seeAlso ?L . graph ?L{ [a foaf:Person ] foaf:name ?N } }
      }
```

- named graph patterns allow to match pattern in remote graphs
- the set of named graphs [Carroll et al., 2005] typically needs to be statically declared in the dataset in current SPARQL implementations (**from named** clause), details see [Prud'hommeaux and Seaborne, 2007], i.e. most SPARQL engines will not deliver the “expected” result here.

CONSTRUCT

construct queries in SPARQL allow to generate new RDF graphs from the results of a query, e.g.

“Create a graph which establishes ‘foaf:knows’ relations for all persons who I have co-authored with according to DBLP.”

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix foaf: <http://xmlns.com/foaf/0.1/>
prefix dc: <http://purl.org/dc/elements/1.1/>
prefix: <http://dblp.l3s.de/d2r/resource/authors/>

construct { <http://polleres.net/foaf.rdf#me> foaf:knows ?Y }
where { ?D dc:creator :Axel_Polleres;
       dc:creator ?Y . FILTER( ?Y != :Axel_Polleres )
}
```


CONSTRUCT

construct queries in SPARQL allow to generate new RDF graphs from the results of a query, e.g.

“Create a graph which establishes ‘foaf:knows’ relations for all persons who I have co-authored with according to DBLP.”

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix foaf: <http://xmlns.com/foaf/0.1/>
prefix dc: <http://purl.org/dc/elements/1.1/>
prefix: <http://dblp.l3s.de/d2r/resource/authors/>

construct { <http://polleres.net/foaf.rdf#me> foaf:knows ?Y }
where { ?D dc:creator :Axel_Polleres;
       dc:creator ?Y . FILTER( ?Y != :Axel_Polleres )
}
```

- “Output pattern” is a basic graph pattern
- similar to “views” in SQL
- May be viewed as a “rules language” itself.

Exercise

Using the SPARQL interface to DBLP at <http://dblp.13s.de/d2r/snorql/> write a query that outputs the following:

Task

Names of people who have published in the Reasoning Web Series but have not co-authored with any of the authors of the present paper

- Can you do it in one query?
- Which of the constructs discussed do you need?

SPARQL summary

- We have only “scratched the surface” here
- Extensions of SPARQL (updates (DELETE, INSERT, ...), aggregate functions (SUM, MAX, COUNT,...), etc.) currently being discussed in W3C, e.g. [esw-wiki, ; Polleres *et al.*, 2007]
- Rigid investigation of SPARQL’s semantics and complexity [Pérez *et al.*, 2006; Gutiérrez *et al.*, 2004]
- Peculiarities in SPARQL’s semantics (multiset semantics, joins over unbound variables, etc. [Pérez *et al.*, 2006; Polleres and Schindlauer, 2007])
- SPARQL itself may be viewed as a “rules language” (construct)
- Translation of SPARQL to rules [Schenk and Staab, 2008; Polleres, 2007]
- SPARQL only does RDF graph pattern matching, what about ontologies? ... Let’s take a look at this next!

Unit Outline

1. Motivation – Aggregating Linked Open Data by Rules
2. How can I publish data? RDF
3. How can I query that data? SPARQL
4. What does that data mean? Ontologies described in RDFS + OWL
5. What's next?

What does RDF data mean?

- *Ontologies* are formal descriptions of what the *vocabulary* used in an RDF document means.

⁶“data” rather than “ontology”, in DL terminology this distinction is often called ABox vs. TBox.

What does RDF data mean?

- *Ontologies* are formal descriptions of what the *vocabulary* used in an RDF document means.
- By vocabulary, we mean here mostly:
 - *properties*, i.e., predicates
 - *classes*, i.e., objects of `rdf:type` triples
 - (*individuals*, i.e., concrete objects)⁶

⁶“data” rather than “ontology”, in DL terminology this distinction is often called ABox vs. TBox.

What does RDF data mean?

- *Ontologies* are formal descriptions of what the *vocabulary* used in an RDF document means.
- By vocabulary, we mean here mostly:
 - *properties*, i.e., predicates
 - *classes*, i.e., objects of `rdf:type` triples
 - (*individuals*, i.e., concrete objects)⁶
- Ontologies describe *relations* among properties, classes and individuals (subclasses, subproperties, equivalence, domain, range, etc.)

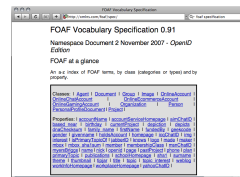
⁶“data” rather than “ontology”, in DL terminology this distinction is often called ABox vs. TBox.

What does RDF data mean?

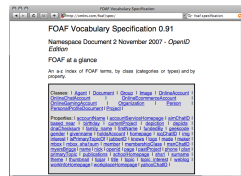
- *Ontologies* are formal descriptions of what the *vocabulary* used in an RDF document means.
- By vocabulary, we mean here mostly:
 - *properties*, i.e., predicates
 - *classes*, i.e., objects of `rdf:type` triples
 - (*individuals*, i.e., concrete objects)⁶
- Ontologies describe *relations* among properties, classes and individuals (subclasses, subproperties, equivalence, domain, range, etc.)
- The W3C has published two standards to describe ontologies, namely *RDF Schema (RDFS)* [Brickley and Guha (eds.), 2004] and the *Web Ontology language (OWL)* [Patel-Schneider *et al.*, 2004]
 - **RDFS** ... simple schema language with minimal expressivity, mostly expressible in simple forward chaining inference rules (*Horn Rules*)
 - **OWL** ... higher expressivity, foundations in *Description Logics*
 - both RDFS and OWL ontologies are RDF graphs themselves, i.e., OWL and RDFS provide “an RDF vocabulary to describe RDF vocabularies”

⁶“data” rather than “ontology”, in DL terminology this distinction is often called ABox vs. TBox.

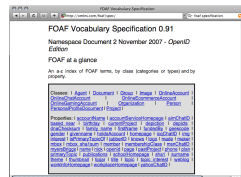
- *Each Person is a Agent* (subclass)



- *Each Person is a Agent* (subclass)
- *The img property is more specific than depiction* (subproperty)



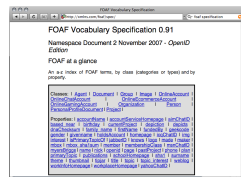
- *Each Person is a Agent* (subclass)
- *The img property is more specific than depiction* (subproperty)
- *img is a relation between Persons and Images* (domain/range)



Example Vocabulary 1 – The FOAF ontology:

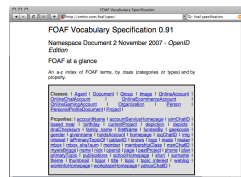
- **Properties:** name, knows, homepage, primaryTopic etc.
- **Classes:** Person, Agent, Document, Organisation, etc.
- **Relations:** e.g.

- *Each Person is a Agent* (subclass)
- *The img property is more specific than depiction* (subproperty)
- *img is a relation between Persons and Images* (domain/range)
- *knows is a relation between two Persons* (domain/range)



- **Properties:** name, knows, homepage, primaryTopic etc.
- **Classes:** Person, Agent, Document, Organisation, etc.
- **Relations:** e.g.

- *Each Person is a Agent* (subclass)
- *The img property is more specific than depiction* (subproperty)
- *img is a relation between Persons and Images* (domain/range)
- *knows is a relation between two Persons* (domain/range)
- *homepage denotes **unique** homepage of an Agent* (uniquely identifying property)



Examples 2 – A simple ontology about reviewers:

- **Properties:** title, isAuthorOf, publishedIn, etc.
- **Classes:** Senior, Paper, Publication, etc.
- **Relations:**
 - *A Publication is a Paper which has been published* (subclass + existential condition on property)

⁷reuse of external ontologies!

Examples 2 – A simple ontology about reviewers:

- **Properties:** title, isAuthorOf, publishedIn, etc.
- **Classes:** Senior, Paper, Publication, etc.
- **Relations:**
 - *A Publication is a Paper which has been published* (subclass + existential condition on property)
 - *isAuthorOf is the opposite of Dublin Core's dc:creator Property*⁷

⁷reuse of external ontologies!

Examples 2 – A simple ontology about reviewers:

- **Properties:** title, isAuthorOf, publishedIn, etc.
- **Classes:** Senior, Paper, Publication, etc.
- **Relations:**
 - *A Publication is a Paper which has been published* (subclass + existential condition on property)
 - *isAuthorOf is the opposite of Dublin Core's dc:creator Property*⁷
 - *A Senior researcher is a foaf:Person who isAuthorOf 10+ Publications* (subclass + condition on cardinality)

⁷reuse of external ontologies!

Examples 2 – A simple ontology about reviewers:

- **Properties:** title, isAuthorOf, publishedIn, etc.
- **Classes:** Senior, Paper, Publication, etc.
- **Relations:**
 - *A Publication is a Paper which has been published* (subclass + existential condition on property)
 - *isAuthorOf is the opposite of Dublin Core's dc:creator Property*⁷
 - *A Senior researcher is a foaf:Person who isAuthorOf 10+ Publications* (subclass + condition on cardinality)
 - *Each item can be publishedIn at most one venue* (functional property)
 -
 -

⁷reuse of external ontologies!

RDF(S) vocabulary: RDF and RDFS themselves are vocabularies!

- **Properties:** `rdf:type`, `rdfs:domain`, `rdfs:range`, `rdf:subClassOf`, `rdf:subPropertyOf`, `rdf:first`, `rdf:rest` etc.
- **Classes:** `rdf:XMLLiteral`, `rdf:Literal`, `rdfs:Resource`, `rdfs:Property`, `rdfs:Class`, `rdf:List`, etc.
- **Relations:**

RDF(S) vocabulary: RDF and RDFS themselves are vocabularies!

- **Properties:** `rdf:type`, `rdfs:domain`, `rdfs:range`, `rdf:subClassOf`, `rdf:subPropertyOf`, `rdf:first`, `rdf:rest` etc.
- **Classes:** `rdf:XMLLiteral`, `rdf:Literal`, `rdfs:Resource`, `rdfs:Property`, `rdfs:Class`, `rdf:List`, etc.
- **Relations:** The semantics of the RDFS vocabulary is defined in [Hayes, 2004]; it is a “meta vocabulary” used to define the semantics of other vocabularies

The Semantics of RDF graphs:

```

@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
<http://www.mat.unical.it/~ianni/foaf.rdf> a foaf:PersonalProfileDocument.
<http://www.mat.unical.it/~ianni/foaf.rdf> foaf:maker _:me .
<http://www.mat.unical.it/~ianni/foaf.rdf> foaf:primaryTopic _:me .
_:me a foaf:Person .
_:me foaf:name "Giovambattista Ianni" .
_:me foaf:homepage <http://www.gibbi.com> .
_:me foaf:phone <tel:+39-0984-496430> .
_:me foaf:knows [ a foaf:Person ;
                  foaf:name "Wolfgang Faber" ;
                  rdfs:seeAlso <http://www.kr.tuwien.ac.at/staff/faber/foaf.rdf> ].
_:me foaf:knows [ a foaf:Person .
                  foaf:name "Axel Polleres" ;
                  rdfs:seeAlso <http://www.polleres.net/foaf.rdf> ].
_:me foaf:knows [ a foaf:Person .
                  foaf:name "Thomas Eiter" ] .
_:me foaf:knows [ a foaf:Person .
                  foaf:name "Alessandra Martello" ] .

```


The Semantics of RDF graphs:

Each RDF graph can be viewed as a first-order formula:

```

$$\begin{aligned} & \exists me, b1, b2, b3, b4 \\ & (triple(foaf.rdf, rdf:type, PersonalProfileDocument) \\ & \wedge triple(foaf.rdf, maker, me) \\ & \wedge triple(foaf.rdf, primaryTopic, me) \\ & \wedge triple(me, rdf:type, Person) \\ & \wedge triple(me, name, "Giovambattista Ianni") \\ & \wedge triple(me, homepage, http://www.gibbi.com) \\ & \wedge triple(me, phone, tel:+39-0984-496430) \\ & \wedge triple(me, knows, b2) \wedge triple(b1, type, Person) \\ & \wedge triple(b1, name, "Wolfgang Faber") \\ & \wedge triple(b1, rdfs:seeAlso, http://www.kr.tuwien...) \\ & \wedge triple(me, knows, b1) \wedge triple(b1, rdf:type, Person) \\ & \wedge triple(b2, name, "Axel Polleres") \\ & \wedge triple(b2, rdfs:seeAlso, http://www.polleres...) \\ & \wedge triple(me, knows, b3) \wedge triple(b1, rdf:type, Person) \\ & \wedge triple(b3, name, "Thomas Eiter") \\ & \wedge triple(me, knows, b4) \wedge triple(b1, type, Person) \\ & \wedge triple(b4, name, "Alessandra Martello")) \end{aligned}$$

```


The Semantics of RDF graphs:

Alternatively, especially in the SW community, F-Logic has gained popularity:

```

$$\begin{aligned} &\exists me, b1, b2, b3, b4 \text{ (foaf.rdf\#PersonalProfileDocument} \\ &\quad \wedge \text{foaf.rdf[maker} \rightarrow me] \\ &\quad \wedge \text{foaf.rdf[primaryTopic} \rightarrow me] \\ &\quad \wedge me\#\text{Person} \wedge \dots) \end{aligned}$$

```

- special notion for class membership (rdf:type): #
- frames $s[p \rightarrow o]$ naturally represent triples

The Semantics of RDF graphs:

Alternatively, especially the OWL favors unary/binary predicate representation:

```

$$\begin{aligned} &\exists me, b1, b2, b3, b4 \text{ (PersonalProfileDocument(foaf.rdf)} \\ &\quad \wedge \text{maker(foaf.rdf, me)} \\ &\quad \wedge \text{primaryTopic(foaf.rdf, me)} \\ &\quad \wedge \text{Person(me)} \wedge \dots) \end{aligned}$$

```

- unary predicates for `rdf:type` predicates
- binary predicates for all other predicates

The Semantics of the RDFS vocabulary:

The formal semantics of RDF(S) [Hayes, 2004] is accompanied by a set of (informative) entailment rules ... can be written down as first-order formulas again:

$$\begin{aligned} \forall S, P, O \quad & (\text{triple}(S, P, O) \supset \text{triple}(S, \text{rdf:type}, \text{rdfs:Resource})) \\ \forall S, P, O \quad & (\text{triple}(S, P, O) \supset \text{triple}(P, \text{rdf:type}, \text{rdf:Property})) \\ \forall S, P, O \quad & (\text{triple}(S, P, O) \supset \text{triple}(O, \text{rdf:type}, \text{rdfs:Resource})) \\ \forall S, P, O \quad & (\text{triple}(S, P, O) \wedge \text{triple}(P, \text{rdfs:domain}, C) \supset \text{triple}(S, \text{rdf:type}, C)) \\ \forall S, P, O, C \quad & (\text{triple}(S, P, O) \wedge \text{triple}(P, \text{rdfs:range}, C) \supset \text{triple}(O, \text{rdf:type}, C)) \\ \forall C \quad & (\text{triple}(C, \text{rdf:type}, \text{rdfs:Class}) \supset \text{triple}(C, \text{rdfs:subClassOf}, \text{rdfs:Resource})) \\ \forall C_1, C_2, C_3 \quad & (\text{triple}(C_1, \text{rdfs:subClassOf}, C_2) \wedge \\ & \quad \text{triple}(C_2, \text{rdfs:subClassOf}, C_3) \supset \text{triple}(C_1, \text{rdfs:subClassOf}, C_3)) \\ \forall S, C_1, C_2 \quad & (\text{triple}(S, \text{rdf:type}, C_1) \wedge \text{triple}(C_1, \text{rdfs:subClassOf}, C_2) \supset \text{triple}(S, \text{rdf:type}, C_2)) \\ \forall S, C \quad & (\text{triple}(S, \text{rdf:type}, C) \supset \text{triple}(C, \text{rdf:type}, \text{rdfs:Class})) \\ \forall C \quad & (\text{triple}(C, \text{rdf:type}, \text{rdfs:Class}) \supset \text{triple}(C, \text{rdfs:subClassOf}, C)) \\ \forall P_1, P_2, P_3 \quad & (\text{triple}(P_1, \text{rdfs:subPropertyOf}, P_2) \wedge \\ & \quad \text{triple}(P_2, \text{rdfs:subPropertyOf}, P_3) \supset \text{triple}(P_1, \text{rdfs:subPropertyOf}, P_3)) \\ \forall S, P_1, P_2, O \quad & (\text{triple}(S, P_1, O) \wedge \text{triple}(P_1, \text{rdfs:subPropertyOf}, P_2) \supset \text{triple}(S, P_2, O)) \\ \forall P \quad & (\text{triple}(P, \text{rdf:type}, \text{rdf:Property}) \supset \text{triple}(P, \text{rdfs:subPropertyOf}, P)) \end{aligned}$$

plus the axiomatic triples from [Hayes, 2004, Sections 3.1 and 4.1].

The Semantics of the RDFS vocabulary:

The formal semantics of RDF(S) [Hayes, 2004] is accompanied by a set of (informative) entailment rules ... can be written down as first-order formulas again:

$$\begin{aligned}
 &\forall S, P, O \ (triple(S, P, O) \supset triple(S, rdf:type, rdfs:Resource)) \\
 &\forall S, P, O \ (triple(S, P, O) \supset triple(P, rdf:type, rdf:Property)) \\
 &\forall S, P, O \ (triple(S, P, O) \supset triple(O, rdf:type, rdfs:Resource)) \\
 &\forall S, P, O \ (triple(S, P, O) \wedge triple(P, rdfs:domain, C) \supset triple(S, rdf:type, C)) \\
 &\forall S, P, O, C \ (triple(S, P, O) \wedge triple(P, rdfs:range, C) \supset triple(O, rdf:type, C)) \\
 &\forall C \ (triple(C, rdf:type, rdfs:Class) \supset triple(C, rdfs:subClassOf, rdfs:Resource)) \\
 &\forall C_1, C_2, C_3 \ (triple(C_1, rdfs:subClassOf, C_2) \wedge \\
 &\quad triple(C_2, rdfs:subClassOf, C_3) \supset triple(C_1, rdfs:subClassOf, C_3)) \\
 &\forall S, C_1, C_2 \ (triple(S, rdf:type, C_1) \wedge triple(C_1, rdfs:subClassOf, C_2) \supset triple(S, rdf:type, C_2)) \\
 &\forall S, C \ (triple(S, rdf:type, C) \supset triple(C, rdf:type, rdfs:Class)) \\
 &\forall C \ (triple(C, rdf:type, rdfs:Class) \supset triple(C, rdfs:subClassOf, C)) \\
 &\forall P_1, P_2, P_3 \ (triple(P_1, rdfs:subPropertyOf, P_2) \wedge \\
 &\quad triple(P_2, rdfs:subPropertyOf, P_3) \supset triple(P_1, rdfs:subPropertyOf, P_3)) \\
 &\forall S, P_1, P_2, O \ (triple(S, P_1, O) \wedge triple(P_1, rdfs:subPropertyOf, P_2) \supset triple(S, P_2, O)) \\
 &\forall P \ (triple(P, rdf:type, rdf:Property) \supset triple(P, rdfs:subPropertyOf, P))
 \end{aligned}$$

plus the axiomatic triples from [Hayes, 2004, Sections 3.1 and 4.1].

The Semantics of the RDFS vocabulary:

Note 1:

All those rules were Datalog expressible, i.e. no negation, no function symbols.

The Semantics of the RDFS vocabulary:

Note 1:

All those rules were Datalog expressible, i.e. no negation, no function symbols.

Note 2:

Writing entailment rules in unary/binary representation would yield second order, e.g.:

$$\forall S, C_1, C_2 (\text{triple}(S, \text{rdf:type}, C_1) \wedge \text{triple}(C_1, \text{rdfs:subClassOf}, C_2) \supset \text{triple}(S, \text{rdf:type}, C_2))$$

The Semantics of the RDFS vocabulary:

Note 1:

All those rules were Datalog expressible, i.e. no negation, no function symbols.

Note 2:

Writing entailment rules in unary/binary representation would yield second order, e.g.:

$$\forall S, C_1, C_2 (C_1(S) \wedge \text{rdfs:subClassOf}(C_1, C_2) \supset C_2(S))$$

RDFS Semantics Example: The FOAF ontology

FOAF Ontology:

- *Each Person is a Agent* (subclass)
- *The img property is more specific than depiction* (subproperty)
- *img is a relation between Persons and Images* (domain/range)
- *knows is a relation between two Persons* (domain/range)
- *homepage denotes **unique** homepage of an Agent* (uniquely identifying property)
- ⋮

RDFS: Semantics

$$\begin{array}{c} \vdots \\ \forall S, C_1, C_2 \ (triple(S, rdf:type, C_1) \wedge triple(C_1, rdfs:subClassOf, C_2) \supset triple(S, rdf:type, C_2)) \\ \vdots \end{array}$$

Data:

```
:me rdf:type foaf:Person .
```


RDFS Semantics Example: The FOAF ontology

FOAF Ontology in RDF:

- `foaf:Person rdfs:subClassOf foaf:Agent .`
- `foaf:img rdfs:subPropertyOf foaf:depiction .`
- `foaf:img rdfs:domain foaf:Person ; rdfs:range foaf:Image .`
- `foaf:img rdfs:domain foaf:Person ; rdfs:range foaf:Person .`
- ???

⋮

RDFS: Semantics

⋮
 $\forall S, C_1, C_2 \ (triple(S, rdf:type, C_1) \wedge triple(C_1, rdfs:subClassOf, C_2) \supset triple(S, rdf:type, C_2))$
 ⋮

Data:

`:me rdf:type foaf:Person .`
`:me rdfs:type foaf:Agent .`

RDFS Semantics Example: The FOAF ontology

FOAF Ontology in RDF:

- `foaf:Person rdfs:subClassOf foaf:Agent .`
- `foaf:img rdfs:subPropertyOf foaf:depiction .`
- `foaf:img rdfs:domain foaf:Person ; rdfs:range foaf:Image .`
- `foaf:img rdfs:domain foaf:Person ; rdfs:range foaf:Person .`
- *homepage denotes unique homepage of an Agent ???*
- ⋮

RDFS: Semantics

$$\begin{array}{c} \vdots \\ \forall S, C_1, C_2 \ (triple(S, rdf:type, C_1) \wedge triple(C_1, rdfs:subClassOf, C_2) \supset triple(S, rdf:type, C_2)) \\ \vdots \end{array}$$

Data:

```
:me rdf:type foaf:Person .
:me rdf:type foaf:Agent .
```


The OWL vocabulary:

- *homepage* denotes **unique** homepage of an *Agent* (uniquely identifying property)

For expressing this, we need more than the RDFS vocabulary. **OWL** is again an RDF vocabulary, extending RDF(S), fixed semantics that adds more expressivity on top of RDFS:

- **Properties:** owl:sameAs, owl:differentFrom, owl:inverseOf, owl:onProperty, owl:allValuesFrom, owl:someValuesFrom, owl:minCardinality, owl:maxCardinality etc.
- **Classes:** owl:Restriction, owl:DatatypeProperty, owl:ObjectProperty, owl:FunctionalProperty, owl:InverseFunctionalProperty, owl:SymmetricProperty etc.
- **Relations:** The semantics of OWL is defined in [Patel-Schneider *et al.*, 2004]
 - in terms of its RDF reading (OWL Full semantics), and

⁸ OWL DL puts restrictions on the use of the OWL and RDF vocabulary, e.g. classes may not be used as instances, etc., for instance `one rdf:type integer . integer rdf:type simpleDatatype .` would not be allowed.

The OWL vocabulary:

- *homepage* denotes **unique** homepage of an *Agent* (uniquely identifying property)

For expressing this, we need more than the RDFS vocabulary. **OWL** is again an RDF vocabulary, extending RDF(S), fixed semantics that adds more expressivity on top of RDFS:

- **Properties:** owl:sameAs, owl:differentFrom, owl:inverseOf, owl:onProperty, owl:allValuesFrom, owl:someValuesFrom, owl:minCardinality, owl:maxCardinality etc.
- **Classes:** owl:Restriction, owl:DatatypeProperty, owl:ObjectProperty, owl:FunctionalProperty, owl:InverseFunctionalProperty, owl:SymmetricProperty etc.
- **Relations:** The semantics of OWL is defined in [Patel-Schneider *et al.*, 2004]
 - in terms of its RDF reading (OWL Full semantics), and
 - in terms of its Description Logics reading (OWL DL semantics)⁸

⁸ OWL DL puts restrictions on the use of the OWL and RDF vocabulary, e.g. classes may not be used as instances, etc., for instance `one rdf:type integer . integer rdf:type simpleDatatype .` would not be allowed.

The Semantics of the OWL vocabulary (DL reading):

Description Logics:

- syntactic variant of first-order logic with equality
- especially tailored for talking about concepts (classes, sets) and roles (properties)
- dedicated symbols for class membership and subclass/subproperty relation:

`foaf:Person rdfs:subClassOf foaf:Agent`

$Person \sqsubseteq Agent$

`:me rdf:type foaf:Person`

$me \in Person$

OWL DL in two slides: 1/2

Expressing property characteristics:

OWL property axioms as RDF triples	DL syntax	FOL short representation
$P \text{ rdfs:domain } C .$	$\top \sqsubseteq \forall P^- . C$	$\forall x, y. P(x, y) \supset C(x)$
$P \text{ rdfs:range } C .$	$\top \sqsubseteq \forall P. C$	$\forall x, y. P(x, y) \supset C(y)$
$P \text{ owl:inverseOf } P_0 .$	$P \equiv P_0^-$	$\forall x, y. P(x, y) \equiv P_0(y, x)$
$P \text{ rdf:type owl:SymmetricProperty.}$	$P \equiv P^-$	$\forall x, y. P(x, y) \equiv P(y, x)$
$P \text{ rdf:type owl:FunctionalProperty.}$	$\top \sqsubseteq \leq 1P$	$\forall x, y, z. P(x, y) \wedge P(x, z) \supset y = z$
$P \text{ rdf:type owl:InverseFunctionalProperty.}$	$\top \sqsubseteq \leq 1P^-$	$\forall x, y, z. P(x, y) \wedge P(z, y) \supset x = z$
$P \text{ rdf:type owl:TransitiveProperty.}$	$P^+ \sqsubseteq P$	$\forall x, y, z. P(x, y) \wedge P(y, z) \supset P(x, z)$

OWL DL in two slides: 1/2

Expressing property characteristics:

OWL property axioms as RDF triples	DL syntax	FOL short representation
$P \text{ rdfs:domain } C$	$\top \sqsubseteq \forall P^-.C$	$\forall x, y. P(x, y) \supset C(x)$
$P \text{ rdfs:range } C$	$\top \sqsubseteq \forall P.C$	$\forall x, y. P(x, y) \supset C(y)$
$P \text{ owl:inverseOf } P_0$	$P \equiv P_0^-$	$\forall x, y. P(x, y) \equiv P_0(y, x)$
$P \text{ rdf:type owl:SymmetricProperty.}$	$P \equiv P^-$	$\forall x, y. P(x, y) \equiv P(y, x)$
$P \text{ rdf:type owl:FunctionalProperty.}$	$\top \sqsubseteq \leq 1P$	$\forall x, y, z. P(x, y) \wedge P(x, z) \supset y = z$
$P \text{ rdf:type owl:InverseFunctionalProperty.}$	$\top \sqsubseteq \leq 1P^-$	$\forall x, y, z. P(x, y) \wedge P(z, y) \supset x = z$
$P \text{ rdf:type owl:TransitiveProperty.}$	$P^+ \sqsubseteq P$	$\forall x, y, z. P(x, y) \wedge P(y, z) \supset P(x, z)$

Expressing complex class descriptions:

OWL complex class descriptions*	DL syntax	FOL short representation
owl:Thing	\top	$x = x$
owl:Nothing	\perp	$\neg x = x$
owl:intersectionOf ($C_1 \dots C_n$)	$C_1 \sqcap \dots \sqcap C_n$	$C_1(x) \wedge \dots \wedge C_n(x)$
owl:unionOf ($C_1 \dots C_n$)	$C_1 \sqcup \dots \sqcup C_n$	$C_1(x) \vee \dots \vee C_n(x)$
owl:complementOf (C)	$\neg C$	$\neg C(x)$
owl:oneOf ($o_1 \dots o_n$)	$\{o_1, \dots, o_n\}$	$x = o_1 \vee \dots \vee x = o_n$
owl:restriction ($P \text{ owl:someValuesFrom } (C)$)	$\exists P.C$	$\exists y. P(x, y) \wedge C(y)$
owl:restriction ($P \text{ owl:allValuesFrom } (C)$)	$\forall P.C$	$\forall y. P(x, y) \supset C(y)$
owl:restriction ($P \text{ owl:value } (o)$)	$\exists P.\{o\}$	$P(x, o)$
owl:restriction ($P \text{ owl:minCardinality } (n)$)	$\geq nP$	$\exists y_1 \dots y_n. \bigwedge_{k=1}^n P(x, y_k) \wedge \bigwedge_{i < j} y_i \neq y_j$
owl:restriction ($P \text{ owl:maxCardinality } (n)$)	$\leq nP$	$\forall y_1 \dots y_{n+1}. \bigwedge_{k=1}^{n+1} P(x, y_k) \supset \bigvee_{i < j} y_i = y_j$

*For reasons of legibility, we use a variant of the OWL abstract syntax [Patel-Schneider et al., 2004] in this table.

OWL DL in two slides: 2/2

Relating Class descriptions:

 $C_1 \text{ rdfs:subClassOf } C_2$ $C_1 \text{ owl:equivalentClass } C_2$ $C_1 \text{ owl:disjointWith } C_2$ $C_1 \sqsubseteq C_2$ $C_1 \equiv C_2$ $C_1 \sqcap C_2 \sqsubseteq \perp$

Relating individuals:

 $o_1 \text{ owl:sameAs } o_2$ $o_1 \text{ owl:differentFrom } o_2$ $o_1 = o_2$ $o_1 \neq o_2$

OWL DL in two slides: 2/2

Relating Class descriptions:

 $C_1 \text{ rdfs:subClassOf } C_2$ $C_1 \sqsubseteq C_2$ $C_1 \text{ owl:equivalentClass } C_2$ $C_1 \equiv C_2$ $C_1 \text{ owl:disjointWith } C_2$ $C_1 \sqcap C_2 \sqsubseteq \perp$

Relating individuals:

 $o_1 \text{ owl:sameAs } o_2$ $o_1 = o_2$ $o_1 \text{ owl:differentFrom } o_2$ $o_1 \neq o_2$

Examples:

```
<http://www.polleres.net/foaf.rdf#me> owl:sameAs  
<http://dblp.l3s.de/d2r/resource/authors/Axel_Polleres> .
```

```
<http://polleres.net/foaf.rdf#me> owl:differentFrom  
<http://www.mat.unical.it/~ianni/foaf.rdf#me> .
```


OWL Example: The FOAF ontology

- *homepage* denotes **unique** homepage of an *Agent* (uniquely identifying property)

...in OWL/RDF syntax:

```
foaf:homepage rdf:type owl:InverseFunctionalProperty .
```


OWL Example: The FOAF ontology

- *homepage* denotes **unique** homepage of an *Agent* (uniquely identifying property)

...in OWL/RDF syntax:

```
foaf:homepage rdf:type owl:InverseFunctionalProperty .
```

...in DL syntax:

$$\top \sqsubseteq \leq 1 \textit{homepage}^-$$

OWL Example: The FOAF ontology

- *homepage* denotes **unique** homepage of an *Agent* (uniquely identifying property)

...in OWL/RDF syntax:

```
foaf:homepage rdf:type owl:InverseFunctionalProperty .
```

...in DL syntax:

$$\top \sqsubseteq \leq 1 \textit{homepage}^-$$

Example inference:

```
<http://www.polleres.net/foaf.rdf#me> foaf:homepage  
  <http://www.polleres.net/> .  
<http://dblp.13s.de/d2r/resource/authors/Axel_Polleres> foaf:homepage  
  <http://www.polleres.net/> .
```


OWL Example: The FOAF ontology

- *homepage* denotes **unique** homepage of an *Agent* (uniquely identifying property)

...in OWL/RDF syntax:

```
foaf:homepage rdf:type owl:InverseFunctionalProperty .
```

...in DL syntax:

$$\top \sqsubseteq \leq 1 \textit{homepage}^-$$

Example inference:

```
<http://www.polleres.net/foaf.rdf#me> foaf:homepage
  <http://www.polleres.net/> .
<http://dblp.13s.de/d2r/resource/authors/Axel_Polleres> foaf:homepage
  <http://www.polleres.net/> .

⊨
<http://www.polleres.net/foaf.rdf#me> owl:sameAs
  <http://dblp.13s.de/d2r/resource/authors/Axel_Polleres> .
```


OWL Example: A simple ontology about reviewers

$$\exists ex:title.\top \sqsubseteq ex:Paper \quad (i)$$

$$\exists ex:title^{\neg}.\top \sqsubseteq xsd:string \quad (ii)$$

$$ex:isAuthorOf^{\neg} \equiv dc:creator \quad (iii)$$

$$ex:Publication \equiv ex:Paper \sqcap \exists ex:publishedIn.\top \quad (iv)$$

$$\top \sqsubseteq \leq 1 \ ex:publishedIn^{\neg} \quad (v)$$

$$ex:Senior \equiv foaf:Person \sqcap \geq 10 \ ex:isAuthorOf \sqcap \quad (vi)$$

$$\exists ex:isAuthorOf.ex:Publication$$

$$ex:Club100 \equiv foaf:Person \sqcap \geq 100 \ ex:isAuthorOf \quad (vii)$$

⁹ (...) is a shortcut for rdf:Lists, expand to rdf:first, rdf:rest, rdf:nil triples.

¹⁰ (vi) is not exactly that, qualified number restrictions will likely be in OWL 2, though.

OWL Example: A simple ontology about reviewers

$\exists ex:title.\top \sqsubseteq ex:Paper$ (i)

$\exists ex:title^{\neg}.\top \sqsubseteq xsd:string$ (ii)

$ex:isAuthorOf^{\neg} \equiv dc:creator$ (iii)

$ex:Publication \equiv ex:Paper \sqcap \exists ex:publishedIn.\top$ (iv)

$\top \sqsubseteq \leq 1\ ex:publishedIn^{\neg}$ (v)

$ex:Senior \equiv foaf:Person \sqcap \geq 10\ ex:isAuthorOf \sqcap$ (vi)

$\exists ex:isAuthorOf.ex:Publication$

$ex:Club100 \equiv foaf:Person \sqcap \geq 100\ ex:isAuthorOf$ (vii)

- *A Publication is a Paper which has been published* (iv)

⁹ (...) is a shortcut for `rdf:Lists`, expand to `rdf:first`, `rdf:rest`, `rdf:nil` triples.

¹⁰ (vi) is not exactly that, qualified number restrictions will likely be in OWL 2, though.

OWL Example: A simple ontology about reviewers

$\exists ex:title.\top \sqsubseteq ex:Paper$ (i)

$\exists ex:title^{\neg}.\top \sqsubseteq xsd:string$ (ii)

$ex:isAuthorOf^{\neg} \equiv dc:creator$ (iii)

$ex:Publication \equiv ex:Paper \sqcap \exists ex:publishedIn.\top$ (iv)

$\top \sqsubseteq \leq 1\ ex:publishedIn^{\neg}$ (v)

$ex:Senior \equiv foaf:Person \sqcap \geq 10\ ex:isAuthorOf \sqcap$ (vi)

$\exists ex:isAuthorOf.ex:Publication$

$ex:Club100 \equiv foaf:Person \sqcap \geq 100\ ex:isAuthorOf$ (vii)

■ *A Publication is a Paper which has been published* (iv)

`ex:Publication owl:intersectionOf (ex:Paper [a owl:Restriction; owl:onProperty ex:publishedIn ;
owl:minCardinality 1]) .9`

⁹ (...) is a shortcut for `rdf:Lists`, expand to `rdf:first`, `rdf:rest`, `rdf:nil` triples.

¹⁰ (vi) is not exactly that, qualified number restrictions will likely be in OWL 2, though.

OWL Example: A simple ontology about reviewers

$\exists ex:title.\top \sqsubseteq ex:Paper$ (i)

$\exists ex:title^{\neg}.\top \sqsubseteq xsd:string$ (ii)

$ex:isAuthorOf^{\neg} \equiv dc:creator$ (iii)

$ex:Publication \equiv ex:Paper \sqcap \exists ex:publishedIn.\top$ (iv)

$\top \sqsubseteq \leq 1\ ex:publishedIn^{\neg}$ (v)

$ex:Senior \equiv foaf:Person \sqcap \geq 10\ ex:isAuthorOf \sqcap$ (vi)

$\exists ex:isAuthorOf.ex:Publication$

$ex:Club100 \equiv foaf:Person \sqcap \geq 100\ ex:isAuthorOf$ (vii)

■ *A Publication is a Paper which has been published* (iv)

`ex:Publication owl:intersectionOf (ex:Paper [a owl:Restriction; owl:onProperty ex:publishedIn ; owl:minCardinality 1]) .9`

■ *isAuthorOf is the opposite of Dublin Core's dc:creator Property* (iii)

⁹ (...) is a shortcut for `rdf:Lists`, expand to `rdf:first`, `rdf:rest`, `rdf:nil` triples.

¹⁰ (vi) is not exactly that, qualified number restrictions will likely be in OWL 2, though.

OWL Example: A simple ontology about reviewers

$\exists ex:title.\top \sqsubseteq ex:Paper$ (i)

$\exists ex:title^{\neg}.\top \sqsubseteq xsd:string$ (ii)

$ex:isAuthorOf^{\neg} \equiv dc:creator$ (iii)

$ex:Publication \equiv ex:Paper \sqcap \exists ex:publishedIn.\top$ (iv)

$\top \sqsubseteq \leq 1\ ex:publishedIn^{\neg}$ (v)

$ex:Senior \equiv foaf:Person \sqcap \geq 10\ ex:isAuthorOf \sqcap$ (vi)

$\exists ex:isAuthorOf.ex:Publication$

$ex:Club100 \equiv foaf:Person \sqcap \geq 100\ ex:isAuthorOf$ (vii)

■ *A Publication is a Paper which has been published* (iv)

`ex:Publication owl:intersectionOf (ex:Paper [a owl:Restriction; owl:onProperty ex:publishedIn ; owl:minCardinality 1]) .9`

■ *isAuthorOf is the opposite of Dublin Core's dc:creator Property* (iii)

`ex:isAuthorOf owl:inverseOf dc:creator .`

⁹ (...) is a shortcut for `rdf:Lists`, expand to `rdf:first`, `rdf:rest`, `rdf:nil` triples.

¹⁰ (vi) is not exactly that, qualified number restrictions will likely be in OWL 2, though.

OWL Example: A simple ontology about reviewers

$\exists ex:title.\top \sqsubseteq ex:Paper$ (i)

$\exists ex:title^{\neg}.\top \sqsubseteq xsd:string$ (ii)

$ex:isAuthorOf^{\neg} \equiv dc:creator$ (iii)

$ex:Publication \equiv ex:Paper \sqcap \exists ex:publishedIn.\top$ (iv)

$\top \sqsubseteq \leq 1 \ ex:publishedIn^{\neg}$ (v)

$ex:Senior \equiv foaf:Person \sqcap \geq 10 \ ex:isAuthorOf \sqcap$ (vi)

$\exists ex:isAuthorOf.ex:Publication$

$ex:Club100 \equiv foaf:Person \sqcap \geq 100 \ ex:isAuthorOf$ (vii)

■ *A Publication is a Paper which has been published* (iv)

`ex:Publication owl:intersectionOf (ex:Paper [a owl:Restriction; owl:onProperty ex:publishedIn ; owl:minCardinality 1]) .9`

■ *isAuthorOf is the opposite of Dublin Core's dc:creator Property* (iii)

`ex:isAuthorOf owl:inverseOf dc:creator .`

■ *A Senior researcher is a foaf:Person who isAuthorOf 10+ Publications* (vi)¹⁰

⁹ (...) is a shortcut for `rdf:Lists`, expand to `rdf:first`, `rdf:rest`, `rdf:nil` triples.

¹⁰ (vi) is not exactly that, qualified number restrictions will likely be in OWL 2, though.

OWL Example: A simple ontology about reviewers

$\exists ex:title.\top \sqsubseteq ex:Paper$ (i)

$\exists ex:title^{\neg}.\top \sqsubseteq xsd:string$ (ii)

$ex:isAuthorOf^{\neg} \equiv dc:creator$ (iii)

$ex:Publication \equiv ex:Paper \sqcap \exists ex:publishedIn.\top$ (iv)

$\top \sqsubseteq \leq 1 \ ex:publishedIn^{\neg}$ (v)

$ex:Senior \equiv foaf:Person \sqcap \geq 10 \ ex:isAuthorOf \sqcap$ (vi)

$\exists ex:isAuthorOf.ex:Publication$

$ex:Club100 \equiv foaf:Person \sqcap \geq 100 \ ex:isAuthorOf$ (vii)

■ *A Publication is a Paper which has been published* (iv)

`ex:Publication owl:intersectionOf (ex:Paper [a owl:Restriction; owl:onProperty ex:publishedIn ; owl:minCardinality 1]) .9`

■ *isAuthorOf is the opposite of Dublin Core's dc:creator Property* (iii)

`ex:isAuthorOf owl:inverseOf dc:creator .`

■ *A Senior researcher is a foaf:Person who isAuthorOf 10+ Publications* (vi)¹⁰

`ex:Senior owl:intersectionOf (foaf:Person [a owl:Restriction; owl:onProperty ex:isAuthorOf ; owl:minCardinality 10] [a owl:Restriction; owl:onProperty ex:isAuthorOf ; owl:someValuesFrom ex:Publication]) .`

⁹ (...) is a shortcut for `rdf:Lists`, expand to `rdf:first`, `rdf:rest`, `rdf:nil` triples.

¹⁰ (vi) is not exactly that, qualified number restrictions will likely be in OWL 2, though.

OWL Example: A simple ontology about reviewers

$\exists ex:title.\top \sqsubseteq ex:Paper$ (i)

$\exists ex:title^{\neg}.\top \sqsubseteq xsd:string$ (ii)

$ex:isAuthorOf^{\neg} \equiv dc:creator$ (iii)

$ex:Publication \equiv ex:Paper \sqcap \exists ex:publishedIn.\top$ (iv)

$\top \sqsubseteq \leq 1 \text{ } ex:publishedIn^{\neg}$ (v)

$ex:Senior \equiv foaf:Person \sqcap \geq 10 \text{ } ex:isAuthorOf \sqcap$ (vi)

$\exists ex:isAuthorOf.ex:Publication$

$ex:Club100 \equiv foaf:Person \sqcap \geq 100 \text{ } ex:isAuthorOf$ (vii)

■ *A Publication is a Paper which has been published* (iv)

`ex:Publication owl:intersectionOf (ex:Paper [a owl:Restriction; owl:onProperty ex:publishedIn ; owl:minCardinality 1]) .9`

■ *isAuthorOf is the opposite of Dublin Core's dc:creator Property* (iii)

`ex:isAuthorOf owl:inverseOf dc:creator .`

■ *A Senior researcher is a foaf:Person who isAuthorOf 10+ Publications* (vi)¹⁰

`ex:Senior owl:intersectionOf (foaf:Person [a owl:Restriction; owl:onProperty ex:isAuthorOf ; owl:minCardinality 10] [a owl:Restriction; owl:onProperty ex:isAuthorOf ; owl:someValuesFrom ex:Publication]) .`

■ *Each item can be publishedIn at most one venue* (v)

⁹ (...) is a shortcut for `rdf:Lists`, expand to `rdf:first`, `rdf:rest`, `rdf:nil` triples.

¹⁰ (vi) is not exactly that, qualified number restrictions will likely be in OWL 2, though.

OWL Example: A simple ontology about reviewers

$\exists ex:title.\top \sqsubseteq ex:Paper$ (i)

$\exists ex:title^{\neg}.\top \sqsubseteq xsd:string$ (ii)

$ex:isAuthorOf^{\neg} \equiv dc:creator$ (iii)

$ex:Publication \equiv ex:Paper \sqcap \exists ex:publishedIn.\top$ (iv)

$\top \sqsubseteq \leq 1 \text{ } ex:publishedIn^{\neg}$ (v)

$ex:Senior \equiv foaf:Person \sqcap \geq 10 \text{ } ex:isAuthorOf \sqcap$ (vi)

$\exists ex:isAuthorOf.ex:Publication$

$ex:Club100 \equiv foaf:Person \sqcap \geq 100 \text{ } ex:isAuthorOf$ (vii)

■ *A Publication is a Paper which has been published* (iv)

`ex:Publication owl:intersectionOf (ex:Paper [a owl:Restriction; owl:onProperty ex:publishedIn ; owl:minCardinality 1]) .9`

■ *isAuthorOf is the opposite of Dublin Core's dc:creator Property* (iii)

`ex:isAuthorOf owl:inverseOf dc:creator .`

■ *A Senior researcher is a foaf:Person who isAuthorOf 10+ Publications* (vi)¹⁰

`ex:Senior owl:intersectionOf (foaf:Person [a owl:Restriction; owl:onProperty ex:isAuthorOf ; owl:minCardinality 10] [a owl:Restriction; owl:onProperty ex:isAuthorOf ; owl:someValuesFrom ex:Publication]) .`

■ *Each item can be publishedIn at most one venue* (v)

`ex:publishedIn a owl:FunctionalProperty .`

⁹ (...) is a shortcut for `rdf:Lists`, expand to `rdf:first`, `rdf:rest`, `rdf:nil` triples.

¹⁰ (vi) is not exactly that, qualified number restrictions will likely be in OWL 2, though.

Reasoning with Ontologies

Tools:

- Special purpose DL reasoners:
Pellet [Sirin *et al.*, 2005], Racer [Haarslev and Möller, 2001], Fact++ [Tsarkov and Horrocks, 2006]

Reasoning with Ontologies

Tools:

- Special purpose DL reasoners:
Pellet [Sirin *et al.*, 2005], Racer [Haarslev and Möller, 2001], Fact++ [Tsarkov and Horrocks, 2006]
- General purpose FOL theorem provers:
SNARK [Stickel *et al.*,], SPASS [SPASS,], Vampire [Riazanov and Voronkov, 2002]

Reasoning with Ontologies

Tools:

- Special purpose DL reasoners:
Pellet [Sirin *et al.*, 2005], Racer [Haarslev and Möller, 2001], Fact++ [Tsarkov and Horrocks, 2006]
- General purpose FOL theorem provers:
SNARK [Stickel *et al.*,], SPASS [SPASS,], Vampire [Riazanov and Voronkov, 2002]
- **Rule/LP engines:** cf. Unit3

SPARQL & Ontologies

SPARQL on top of ontologies not (yet) entirely clear (cf. [Arenas *et al.*, , Unit 5b]):

SPARQL & Ontologies

SPARQL on top of ontologies not (yet) entirely clear (cf. [Arenas *et al.*, , Unit 5b]):

- Problem 1: infinite RDFS/OWL inferences on a finite graph

SPARQL & Ontologies

SPARQL on top of ontologies not (yet) entirely clear (cf. [Arenas *et al.*, , Unit 5b]):

- Problem 1: infinite RDFS/OWL inferences on a finite graph
- Problem 2: co-reference of blank nodes in SPARQL solutions

SPARQL & Ontologies

SPARQL on top of ontologies not (yet) entirely clear (cf. [Arenas *et al.*, , Unit 5b]):

- Problem 1: infinite RDFS/OWL inferences on a finite graph
- Problem 2: co-reference of blank nodes in SPARQL solutions
- Problem 3: SPARQL is SQL inspired (CWA), OWL/RDFS are (OWA):
e.g., OPTIONAL patterns are non-monotonic, RDFS+OWL inference are both monotonic, that can lead to query answers valid under simple RDF, but not under OWL entailment, etc.

SPARQL & Ontologies

SPARQL on top of ontologies not (yet) entirely clear (cf. [Arenas *et al.*, , Unit 5b]):

- Problem 1: infinite RDFS/OWL inferences on a finite graph
- Problem 2: co-reference of blank nodes in SPARQL solutions
- Problem 3: SPARQL is SQL inspired (CWA), OWL/RDFS are (OWA):
e.g., OPTIONAL patterns are non-monotonic, RDFS+OWL inference are both monotonic, that can lead to query answers valid under simple RDF, but not under OWL entailment, etc.

SPARQL & Ontologies

SPARQL on top of ontologies not (yet) entirely clear (cf. [Arenas *et al.*, , Unit 5b]):

- Problem 1: infinite RDFS/OWL inferences on a finite graph
- Problem 2: co-reference of blank nodes in SPARQL solutions
- Problem 3: SPARQL is SQL inspired (CWA), OWL/RDFS are (OWA):
e.g., OPTIONAL patterns are non-monotonic, RDFS+OWL inference are both monotonic, that can lead to query answers valid under simple RDF, but not under OWL entailment, etc.

OWL2 WG tries to define SPARQL DL ... stay tuned!

Unit Outline

1. Motivation – Aggregating Linked Open Data by Rules
2. How can I publish data? RDF
3. How can I query that data? SPARQL
4. What does that data mean? Ontologies described in RDFS + OWL
5. What's next?

What's next? Rules!

- Can I use my favorite rules engine/language to do Ontology Reasoning with RDFS+OWL?
- Can I use *rules* to aggregate the Web data I collected from various sites beyond the expressivity of RDFS+OWL?
- Can I combine RDFS, OWL, SPARQL and custom rules?

-  Marcelo Arenas, Claudio Gutierrez, Bijan Parsia, Jorge Pérez, Axel Polleres, and Andy Seaborne.
-  David Beckett.
Turtle - Terse RDF Triple Language, November 2007.
Available at <http://www.dajobe.org/2004/01/turtle/>.
-  Uldis Bojārs, John G. Breslin, Diego Berrueta, Dan Brickley, Stefan Decker, Sergio Fernández, Christoph Görn, Andreas Harth, Tom Heath, Kingsley Idehen, Kjetil Kjernsmo, Alistair Miles, Alexandre Passant, Axel Polleres, Luis Polo, and Michael Sintek.
SIOC Core Ontology Specification, June 2007.
W3C member submission.
-  Dan Brickley and R.V. Guha (eds.).
RDF vocabulary description language 1.0: RDF Schema, February 2004.
W3C Recommendation, available at <http://www.w3.org/TR/rdf-schema/>.
-  Dan Brickley and Libby Miller.
FOAF Vocabulary Specification 0.91, November 2007.
<http://xmlns.com/foaf/spec/>.
-  Jeremy Carroll, Christian Bizer, Pat Hayes, and Patrick Stickler.
Named graphs.
Journal of Web Semantics, 3(4), 2005.
-  Dan Connolly (ed.).
Gleaning Resource Descriptions from Dialects of Languages (GRDDL), September 2007.



ESW Wiki: SPARQL.

List of useful links for SPARQL implementations and extensions,
<http://esw.w3.org/topic/SPARQL/>.



Claudio Gutiérrez, Carlos A. Hurtado, and Alberto O. Mendelzon.

Foundations of semantic web databases.

In *PODS*, pages 95–106, 2004.



V. Haarslev and R. Möller.

RACER System Description.

In *Proceedings of the First International Joint Conference on Automated Reasoning (IJCAR 2001)*, volume 2083 of *Lecture Notes in Computer Science (LNCS)*, pages 701–705. Springer Verlag, 2001.



P. Hayes.

RDF semantics, 2004.

<http://www.w3.org/TR/rdf-mt/>.



Mikael Nilsson, Andy Powell, Pete Johnston, and Ambjörn Naeve.

Expressing dublin core metadata using the resource description framework (rdf), January 2008.

DCMI Recommendation.



Peter F. Patel-Schneider, Patrick Hayes, and Ian Horrocks.

OWL Web Ontology Language Semantics and Abstract Syntax, February 2004.

W3C Recommendation.



Jorge Pérez, Marcelo Arenas, and Claudio Gutierrez.

Semantics and complexity of sparql.

In *International Semantic Web Conference (ISWC 2006)*, pages 30–43, 2006.



Axel Polleres and Roman Schindlauer.

dlvhex-sparql: A SPARQL-compliant query engine based on dlvhex.

In *2nd International Workshop on Applications of Logic Programming to the Web, Semantic Web and Semantic Web Services (ALPSWS2007)*, volume 287 of *CEUR Workshop Proceedings*, pages 3–12, Porto, Portugal, September 2007. CEUR-WS.org.



Axel Polleres, François Scharffe, and Roman Schindlauer.

SPARQL++ for mapping between RDF vocabularies.

In *ODBASE 2007*, volume 4803 of *Lecture Notes in Computer Science (LNCS)*, pages 878–896, Vilamoura, Algarve, Portugal, November 2007. Springer.



Axel Polleres.

From SPARQL to rules (and back).

In *Proceedings of the 16th World Wide Web Conference (WWW2007)*, Banff, Canada, May 2007.



SPARQL Query Language for RDF, January 2007.

W3C Recommendation, available at

<http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115/>.



Alexandre Riazanov and Andrei Voronkov.

The design and implementation of vampire.

AI Communications, 15(2-3):91–110, 2002.



Simon Schenk and Steffen Staab.

Networked graphs: A declarative mechanism for sparql rules, sparql views and rdf data integration on the web.

In *Proceedings WWW-2008*, pages 585–594, 2008.



Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz.
Pellet: A practical OWL-DL reasoner.

Technical Report 68, UMIACS, University of Maryland, 2005.



Spass: An automated theorem prover for first-order logic with equality.

Available at <http://www.spass-prover.org/>.



Mark E. Stickel, Richard J. Waldinger, and Vinay K. Chaudhr.

A guide to snark.

Available at <http://www.ai.sri.com/snark/tutorial/tutorial.html>.



Dmitry Tsarkov and Ian Horrocks.

Fact++ Description Logic Reasoner: System Description.

In *Proceedings of the Third International Joint Conference on Automated Reasoning (IJCAR 2006)*, 2006.