Presentation metadata

Open Data Support is funded by the European Commission under SMART 2012/0107 'Lot 2: Provision of services for the Publication, Access and Reuse of Open Public Data across the European Union, through existing open data portals'(Contract No. 30-CE-0530965/00-17).

OPEN DATA SUPPORT

Training Module 1.3

Introduction to RDF & SPARQL

Learning objectives

By the end of this training module you should have an understanding of:

- The Resource Description Framework (RDF).
- How to write/read RDF.
- How you can describe your data with RDF.
- What SPARQL is.
- The different types of SPARQL queries.
- How to write a SPARQL query.

Content

This module contains ...

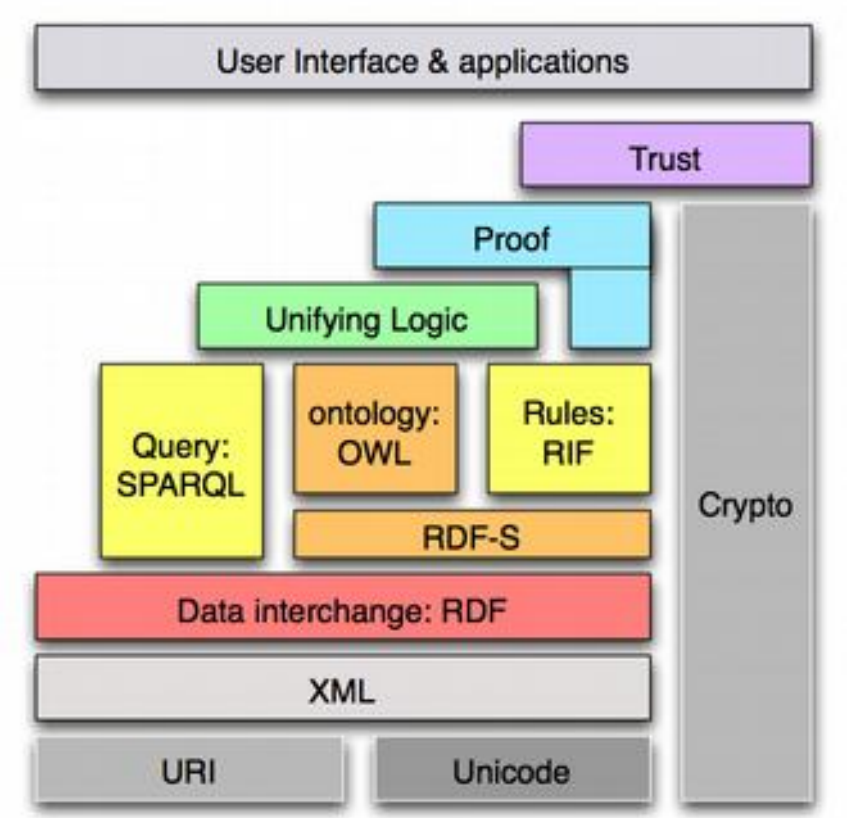
- An introduction to the Resource Description Framework (RDF) for describing your data.
  - What is RDF?
  - How is it structured?
  - How to represent your data in RDF.
- An introduction to SPARQL on how you can query and manipulate data in RDF.
- Pointers to further reading, examples and exercises.

Resource Description Framework

An introduction on RDF.

RDF in the stack of Semantic Web technologies

- RDF stands for:
  - **Resource**: Everything that can have a unique identifier (URI), e.g. pages, places, people, dogs, products...
  - **Description**: attributes, features, and relations of the resources
  - **Framework**: model, languages and syntaxes for these descriptions
- RDF was published as a **W3C recommendation** in 1999.
- RDF was originally introduced as a data model for **metadata.**
- RDF was generalised to cover **knowledge of all kinds**.



See also:
http://www.w3.org/RDF/

Example: RDF description of an organisation

Nike, Dahliastraat 24, 2160 Wommelgem

```
<rdf:RDF
  xmlns:rov="http://www.w3.org/TR/vocab-regorg/ "
  xmlns:org="http://www.w3.org/TR/vocab-org/"
  xmlns:locn="http://www.w3.org/ns/locn#" >

<rov:RegisteredOrganization rdf:about="http://example.com/org/2172798119">
  <rov:legalName> "Nike"< /rov:legalName>
  <org:hasRegisteredSite rdf:resource="http://example.com/site/1234"/>
</rov:RegisteredOrganization>

<locn:Address rdf:about="http://example.com/site/1234"/>
  <locn:fullAddress>" Dahliastraat 24, 2160 Wommelgem"</locn:fullAddress>
</locn:Address>

</rdf:RDF>
```

RDF structure

Triples, graphs and syntax.

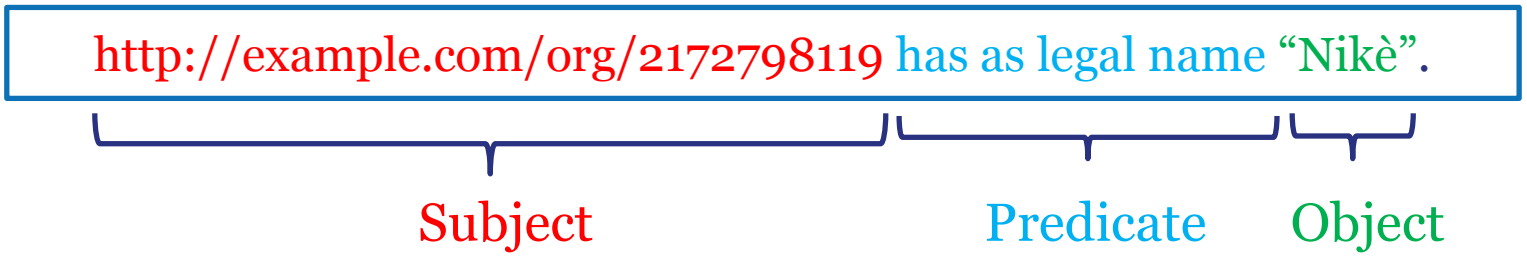
What is a triple?

RDF is a general syntax for representing data on the Web.

Every piece of information expressed in RDF is represented as a **triple**:

- ***Subject*** – a resource, which may be identified with a URI.
- ***Predicate*** – a URI-identified reused specification of the relationship.
- ***Object*** – a resource or literal to which the subject is related.

Example: name of a legal entity:



RDF is graph based

Graph =

A collection of triples

RDF Syntax
RDF/XML

```
<rdf:RDF

  xmlns:rov="http://www.w3.org/TR/vocab-regorg/ "
  xmlns:org="http://www.w3.org/TR/vocab-org/"
  xmlns:locn="http://www.w3.org/ns/locn#" >

<rov:RegisteredOrganization rdf:about="http://example.com/org/2172798119">
  <rov:legalName> "Niké"< /rov:legalName>
  <org:hasRegisteredSite rdf:resource="http://example.com/site/1234"/>
</rov:RegisteredOrganization>

<locn:Address rdf:about="http://example.com/site/1234"/>
  <locn:fullAddress>" Dahliastraat 24, 2160 Wommelgem"</locn:fullAddress>
</locn:Address>

</rdf:RDF>
```

Definition of prefixes

Description of data – triples

Graph

Subject

Predicate

Object

RDF/XML is currently the only syntax that is standardised by W3C.

RDF Syntax
Turtle

```
@prefix rov: <http://www.w3.org/TR/vocab-regorg/> .
@prefix org: <http://www.w3.org/TR/vocab-org/> .
@prefix locn: <http://www.w3.org/ns/locn#> .

< http://example.com/org/2172798119 >
 a <rov:RegisteredOrganization> ;
  rov:legalName "Niké ";
  org:hasRegisteredSite <http://example.com/site/1234> .

<http://example.com/site/1234>
  a <locn:Address> ;
  locn:fullAddress "Dahliastraat 24, 2160 Wommelgem" .
```

Definition of prefixes

Description of data – triples

Graph

Subject

Predicate

Object

Turtle will be standardised in RDF 1.1.

See also:
http://www.w3.org/2009/12/rdf-ws/papers/ws11

RDF Syntax
RDFa

embedding RDF data in HTML

```
<html>
<head> ... </head>
<body>
...
<div resource="http://example.com/org/2172798119"
typeof= "http://www.w3.org/TR/vocab-regorg/RegisteredOrganization">
<p>
<span property=" http://www.w3.org/TR/vocab-regorg/legalName">Nike<span>
Address: <span property="http://www.w3.org/ns/locn#fullAddress"> Dahliastraat
24, 2160 Wommelgem </span>
</p></div>
 </body>
```

Subject

Predicate

Object

See also:
http://www.w3.org/TR/2012/NOTE-rdfa-primer-20120607/

How to represent data in RDF

Classes, properties and vocabularies

RDF Vocabulary

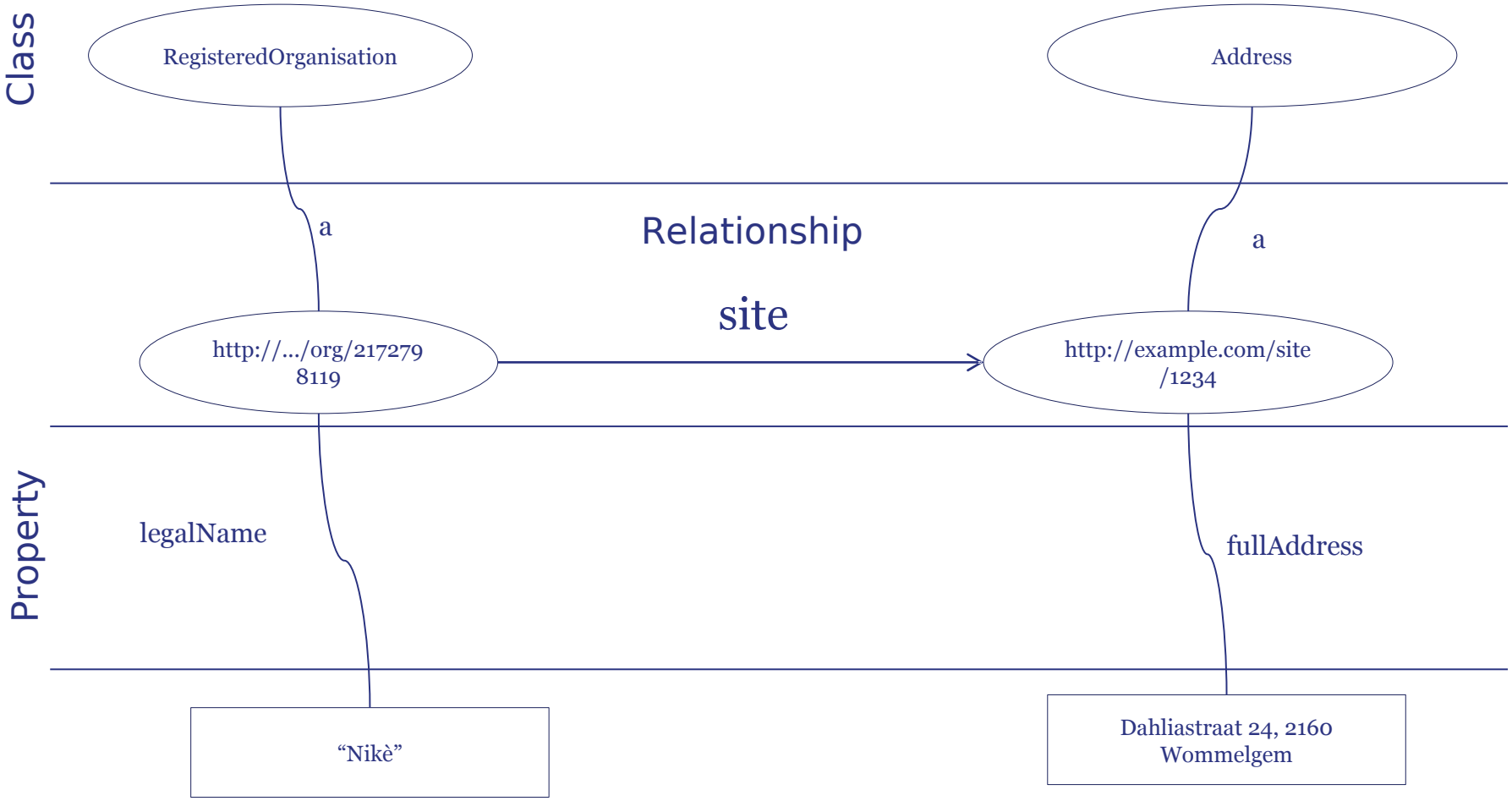
"A vocabulary is a data model comprising classes, properties and relationships which can be used for describing your data and metadata."

- RDF Vocabularies are ***sets of terms*** used to describe things.
- A term is either a ***class or a property***.
  - Object type properties (relationships)
  - Data type properties (attributes)

What are classes, relationships and properties?

- **Class.** A construct that represents things in the real and/or information world, e.g. a person, an organisation, a concepts such as “health” or “freedom”.
- **Relationship.** A link between two classes; for the link between a document and the organisation that published it (i.e. organisation *publishes* document), or the link between a map and the geographic region it depicts (i.e. map *depicts* geographic region). In RDF relationships are encoded as object type properties.
- **Property.** A characteristic of a class in a particular dimension such as the legal name of an organisation or the date and time that an observation was made.

Examples of classes, relationships and properties



Reusing RDF vocabularies

- Reuse greatly **aids interoperability** of your data.

  Use of dcterms:created, for example, the value for which should be a data typed date such as 2013-02-21^^xsd:date, is immediately processable by many machines. If your schema encourages data publishers to use a different term and date format, such as ex:date "21 February 2013" – data published using your schema will require further processing to make it the same as everyone else's.

- Reuse **adds credibility** to your schema.

  It shows it has been published with care and professionalism, again, this promotes its reuse.

- Reuse is **easier** and **cheaper**.

  Reusing classes and properties from well defined and properly hosted vocabularies avoids your having to replicate that effort.

See also:
https://joinup.ec.europa.eu/community/semic/document/cookbook-translating-data-models-rdf-schemas
http://www.slideshare.net/OpenDataSupport/model-your-data-metadata

Where can I find existing vocabularies?

http://lov.okfn.org/

http://joinup.ec.europa.eu/

Well-known vocabularies

See also:
http://www.w3.org/wiki/TaskForces/CommunityProjects/LinkingOpenData/CommonVocabularies

Friend-of-a-Friend (FOAF)	Vocabulary for describing people
Core Person Vocabulary	Vocabulary to describe the fundamental characteristics of a person, e.g. the name, the gender, the date of birth…
DOAP	Vocabulary for describing projects
DCAT-AP	Vocabulary based on the Data Catalogue vocabulary (DCAT) for describing public sector datasets in Europe.
ADMS	Vocabulary for describing interoperability assets.
Dublin Core	Defines general metadata attributes
Registered Organisation Vocabulary	Vocabulary for describing organizations, typically in a national or regional register
Organization Ontology	for describing the structure of organizations
Core Location Vocabulary	Vocabulary capturing the fundamental characteristics of a location.
Core Public Service Vocabulary	Vocabulary capturing the fundamental characteristics of a service offered by public administration
schema.org	Agreed vocabularies for publishing structured data on the Web elaborated by Google, Yahoo and Microsoft

Model your own vocabulary as an RDF Schema

If there is no suitable authoritative reusable vocabulary for describing your data, use conventions for describing your own vocabulary:

- RDF Schema (RDFS)
- Web Ontology Language (OWL)

Example: definition of a class :

```
cpsv:PublicService a rdfs:Class, owl:Class;
        rdfs:label "Public Service"@en;
        rdfs:comment "This class represents the service itself. As noted in
        the scope, a public service is the capacity to carry out a procedure
        and exists whether it is used or not. It is a set of deeds and
        acts performed by or on behalf of a public agency for the  benefit of a
        citizen, a business or another public agency."@en.
```

See also:
http://www.slideshare.net/OpenDataSupport/model-your-data-metadata

Introduction to SPARQL

The RDF Query Language

About SPARQL

SPARQL is the standard language to query graph data represented as RDF triples.

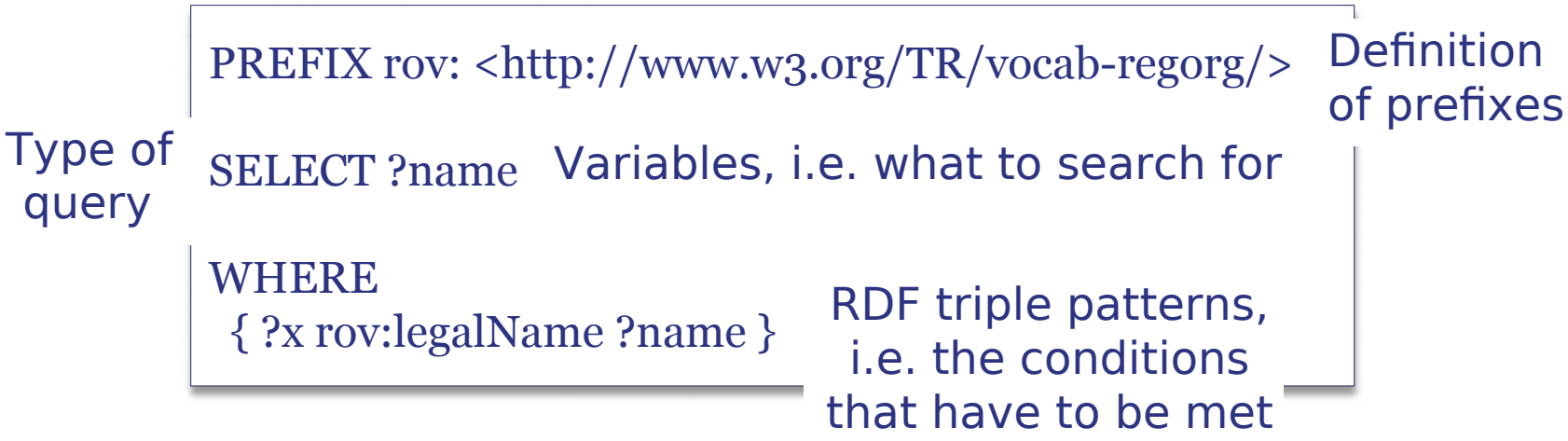
- **S**PARQL **P**rotocol and **R**DF **Q**uery **L**anguage
- One of the three core standards of the Semantic Web, along with RDF and OWL.
- Became a W3C standard January 2008.
- SPARQL 1.1 now in Working Draft status.

Types of SPARQL queries

- **SELECT**
  Return a table of all X, Y, etc. satisfying the following conditions ...
- **CONSTRUCT**
  Find all X, Y, etc. satisfying the following conditions ... and substitute them into the following template in order to generate (possibly new) RDF statements, creating a new graph.
- **DESCRIBE**
  Find all statements in the dataset that provide information about the following resource(s) ... (identified by name or description)
- **ASK**
  Are there any X, Y, etc. satisfying the following conditions ...

See also:
http://www.euclid-project.eu/modules/chapter2

Structure of a SPARQL Query



SELECT – return the name of an organisation with particular URI

Sample data

```
comp:A rov:haslegalName “Niké” .
comp:A org:hasRegisteredSite site:1234 .

Comp:B rov:haslegalName “BARCO” .

site:1234 locn:fullAddress “Dahliastraat 24, 2160 Wommelgem .
```

Query

```
PREFIX comp: < http://example/org/org/>
PREFIX org: < http://www.w3.org/TR/vocab-regorg/ >
PREFIX site: <http://example.org/site/>
PREFIX rov: <http://www.w3.org/TR/vocab-regorg/>

SELECT ?name

WHERE
  { ?x org:hasRegisteredSite site:1234 .
    ?x rov:haslegalName ?name  .}
```

Result

name
“Niké”

SELECT - return the name and address of organis

Sample data

```
comp:A rov:haslegalName “Niké” .
comp:A org:hasRegisteredSite site:1234 .

Comp:B rov:haslegalName “BARCO” .

site:1234 locn:fullAddress “Dahliastraat 24, 2160 Wommelgem” .
```

Query

```
PREFIX org: < http://www.w3.org/TR/vocab-regorg/ >
PREFIX locn:< http://www.w3.org/ns/locn#>
PREFIC rov:<http://www.w3.org/TR/vocab-regorg/>

SELECT ?name ?address

WHERE
 { ?x org:hasRegisteredSite ?site.
   ?x rov:haslegalName ?name  .
   ?site locn:fullAddress ?address  . }
```

Result

name	address
“Niké”	“Dahliastraat 24, 2160 Wommelgem”

SELECT - Return all books under a certain price (1

Sample data

```
@prefix dc:   <http://purl.org/dc/elements/1.1/> .
@prefix :     <http://example.org/book/> .
@prefix ns:   <http://example.org/ns#> .

:book1  dc:title  "SPARQL Tutorial" .
:book1  ns:price  42 .
:book2  dc:title  "The Semantic Web" .
:book2  ns:price  23 .
```

SELECT – Return all books under a certain price (2

Query

```
PREFIX dc:   <http://purl.org/dc/elements/1.1/> .
PREFIX :     <http://example.org/book/> .
PREFIX ns:   <http://example.org/ns#> .

SELECT ?book ?title

WHERE
 { ?book dc:title ?title .
   ?book ns:price ?price . FILTER ( ?price < 40 )
 }
```

Result

book	title
:book2	“The Semantic Web”

CONSTRUCT – Create a new graph with another la name

Sample data

```
comp:A rov:haslegalName “Niké” .
comp:A org:hasRegisteredSite site:1234 .

comp:B rov:haslegalName “BARCO” .

site:1234 locn:fullAddress “Dahliastraat 24, 2160 Wommelgem” .
```

Query

```
PREFIX comp: < http://example/org/org/>
PREFIX org: < http://www.w3.org/TR/vocab-regorg/ >
PREFIC rdfs: <http://www.w3.org/TR/rdf-schema/>

CONSTRUCT {?comp rdfs:label ?name}

WHERE
 { ?comp org:haslegalName ?name. }
```

Resulting graph

```
@prefix comp: <http://example/org/> .
@prefix rdfs: <http://www.w3.org/TR/rdf-schema/>

comp:a rdfs:label “Niké" .
comp:b rdfs:label “BARCO" .
```

DESCRIBE – Return all triples of organisations reg at a particular site

Sample data

```
comp:A rov:haslegalName “Niké” .
comp:A org:hasRegisteredSite site:1234 .

comp:B rov:haslegalName “BARCO” .

site:1234 locn:fullAddress “Dahliastraat 24, 2160 Wommelgem” .
```

Query

```
PREFIX comp: <http://example/org/>
PREFIX site: <http://example/site>
PREFIX org: < http://www.w3.org/TR/vocab-regorg/

DESCRIBE ?organisation

WHERE
{?organisation org:hasRegisteredSite site:1234}
```

Result

```
@prefix comp: <http://example/org/> .
@prefix org: <http://www.w3.org/TR/vocab-regorg/> .

comp:A has:legalName “Niké” .
comp:A org:hasRegisteredSite site:1234 .
```

DESCRIBE – Return all triples associated to a part resource (organisation)

Sample data

```
comp:A rov:haslegalName “Niké” .
comp:A org:hasRegisteredSite site:1234 .

comp:B rov:haslegalName “BARCO” .

site:1234 locn:fullAddress “Dahliastraat 24, 2160 Wommelgem” .
```

Query

```
PREFIX comp: <http://example/org/>

DESCRIBE comp:A
```

Result

```
@prefix comp: <http://example/org/> .
@prefix org: <http://www.w3.org/TR/vocab-regorg/> .

comp:A rov:haslegalName “Niké” .
comp:A org:hasRegisteredSite site:1234 .
```

ASK – Are there any organisations having “1234” a registered site?

Sample data

```
comp:A rov:haslegalName “Niké” .
comp:A org:hasRegisteredSite site:1234 .

comp:B rov:haslegalName “BARCO” .

site:1234 locn:fullAddress “Dahliastraat 24, 2160 Wommelgem” .
```

Query

```
PREFIX org: < http://www.w3.org/TR/vocab-regorg/

ASK

WHERE
{?organisation org:hasRegisteredSite site:1234}
```

Result

TRUE

ASK – Is there a registered site for organisation "B

Sample data

```
comp:A rov:haslegalName "Niké" .
comp:A org:hasRegisteredSite site:1234 .

comp:B rov:haslegalName "BARCO" .

site:1234 locn:fullAddress "Dahliastraat 24, 2160 Wommelgem" .
```

Query

```
PREFIX comp: <http://example/org/>
PREFIX org: <http://www.w3.org/TR/vocab-regorg/>

ASK

WHERE
  {comp:B org:hasRegisteredSite ?site .}
```

Result

FALSE

SPARQL Update

Can be used for...

- Adding data (INSERT)
- Deleting data (DELETE)
- Loading RDF Graph (LOAD / LOAD .. INTO)
- Clearing an RDF Graph (CLEAR GRAPH)
- Creating RDF Graphs (CREATE GRAPH)
- Removing RDF Graphs (DROP GRAPH)
- Copying RDF Graphs (COPY GRAPH ... TO GRAPH)
- Moving RDF Graphs (MOVE GRAPH ... TO GRAPH)
- Adding RDF Graphs (ADD GRAPH TO GRAPH)

See also:
http://www.euclid-project.eu/modules/chapter2
http://www.w3.org/TR/sparql11-update/

INSERT – Add a registered site for "BARCO"?

Sample data

```
comp:A rov:haslegalName "Niké" .
comp:A org:hasRegisteredSite site:1234 .

comp:B rov:haslegalName "BARCO" .

site:1234 locn:fullAddress "Dahliastraat 24, 2160 Wommelgem" .
```

Query

```
PREFIX comp: <http://example/org/>
PREFIX org: <http://www.w3.org/TR/vocab-regorg/>

INSERT DATA
{
site:5678 locn:fullAddress "President Kennedypark 35, 8500 Kortrijk" .
comp:B org:hasRegisteredSite site:5678 .
}
```

Result

```
comp:A rov:haslegalName "Niké" .
comp:A org:hasRegisteredSite site:1234 .

comp:B rov:haslegalName "BARCO" .
comp:B org:hasRegisteredSite site:5678 .

site:1234 locn:fullAddress "Dahliastraat 24, 2160 Wommelgem" .
site:5678 locn:fullAddress "President Kennedypark 35, 8500 Kortrijk" .
```

INSERT/DELETE – Change the address for "Niké"?

Data

```
comp:A rov:haslegalName "Niké" .
comp:A org:hasRegisteredSite site:1234 .

comp:B rov:haslegalName "BARCO" .

site:1234 locn:fullAddress "Dahliastraat 24, 2160 Wommelgem" .
```

Query

```
PREFIX comp: <http://example/org/>
PREFIX org: <http://www.w3.org/TR/vocab-regorg/>

DELETE DATA
{
comp:A org:hasRegisteredSite site:1234 .
}

INSERT DATA
{
site:5678 locn:fullAddress "Rue de Loi 34, 1000 Bruxelles" .
comp:A org:hasRegisteredSite site:5678 .
}
```

Result

```
comp:A rov:haslegalName "Niké" .
comp:A org:hasRegisteredSite site:1000.

site:1234 locn:fullAddress "Dahliastraat 24, 2160 Wommelgem" .
site:1000 locn:fullAddress "Rue de Loi 34, 1000 Bruxelles" .
```

Summary

- RDF is a general way to express **data** intended for publishing on the **Web**.
- RDF data is expressed in **triples**: subject, predicate, object.
- **Different syntaxes** exist for expressing data in RDF.
- SPARQL is a standardised language to **query** graph data expressed as RDF.
- SPARQL can be used to **query** and **update** RDF data.

Thank you!

...and now YOUR question

Take the online test here!

This presentation has been created by Open Data S

Disclaimers

1. The views expressed in this presentation are purely those of the authors and may not, in any circumstances, be interpreted as stating an official position of the European Commission.
The European Commission does not guarantee the accuracy of the information included in this presentation, nor does it accept any responsibility for any use thereof.
Reference herein to any specific products, specifications, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favouring by the European Commission.
All care has been taken by the author to ensure that s/he has obtained, where necessary, permission to use any parts of manuscripts including illustrations, maps, and graphs, on which intellectual property rights already exist from the titular holder(s) of such rights or from her/his or their legal representative.

2. This presentation has been carefully compiled by PwC, but no representation is made or warranty given (either express or implied) as to the completeness or accuracy of the information it contains. PwC is not liable for the information in this presentation or any decision or consequence based on the use of it. PwC will not be liable for any damages arising from the use of the information contained in this presentation. The information contained in this presentation is of a general nature and is solely for guidance on matters of general interest. This presentation is not a substitute for professional advice on any particular matter. No reader should act on the basis of any matter contained in this publication without considering appropriate professional advice.

Authors:
Michiel De Keyzer, Nikolaos Loutas, Christophe Colas and Stijn Goedertier

References

Slide 6:

- Semantic Web Stack. W3C. http://www.w3.org/DesignIssues/diagrams/sweb-stack/2006a.png

Slides 18& 20:

- Linked Data Cookbook. W3C. http://www.w3.org/2011/gld/wiki/Linked_Data_Cookbook

Slide 21:

- Cookbook for translating data models to RDF schemas. ISA Programme. https://joinup.ec.europa.eu/community/semic/document/cookbook-translating-data-models-rdf-schemas

Slide 22:

- Common Vocabularies / Ontologies / Micromodels. W3C. http://www.w3.org/wiki/TaskForces/CommunityProjects/LinkingOpenData/CommonVocabularies

Slide 23-24:

- SPARQL Query Language for RDF. W3C. http://www.w3.org/TR/rdf-sparql-query/

Slide 24:

- Module 2: Querying Linked Data. EUCLID. http://www.euclid-project.eu/modules/course2

Slide 35:

- Module 2: Querying Linked Data. EUCLID. http://www.euclid-project.eu/modules/course2
- SPARQL 1.1 Update. W3C.. http://www.w3.org/TR/sparql11-update/

Further reading

Learning SPARQL. Bob DuCharme.

http://www.learningsparql.com/

Semantic Web for the working ontologist. Dean Allemang, Jim Hendler.

http://workingontologist.org/

EUCLID - Course 2: Querying Linked Data

http://www.euclid-project.eu/modules/course2

Related projects and initiatives

Joinup, https://joinup.ec.europa.eu/

Linked Open Vocabularies, http://okfn.org/

W3C GLD WG, http://www.w3.org/2011/gld/wiki/Main_Page
W3C Schools – Learn RDF
http://www.w3schools.com/rdf/default.asp

EUCLID, http://euclid-project.eu/

TopBraid Composer

Protégé Ontology Editor , http://protege.stanford.edu/

XML Summer School http://xmlsummerschool.com/

Be part of our team…

Find us on

Open Data Support
http://www.slideshare.net/OpenDataSupport

Open Data Support
http://goo.gl/y9ZZI

Join us on

http://www.opendatasupport.eu

Follow us

@OpenDataSupport

Contact us

contact@opendatasupport.eu