Resource Description Framework

The Semantic Web

- The Semantic Web is a Web in which the resources (things) are semantically described, through the usage of an ontology

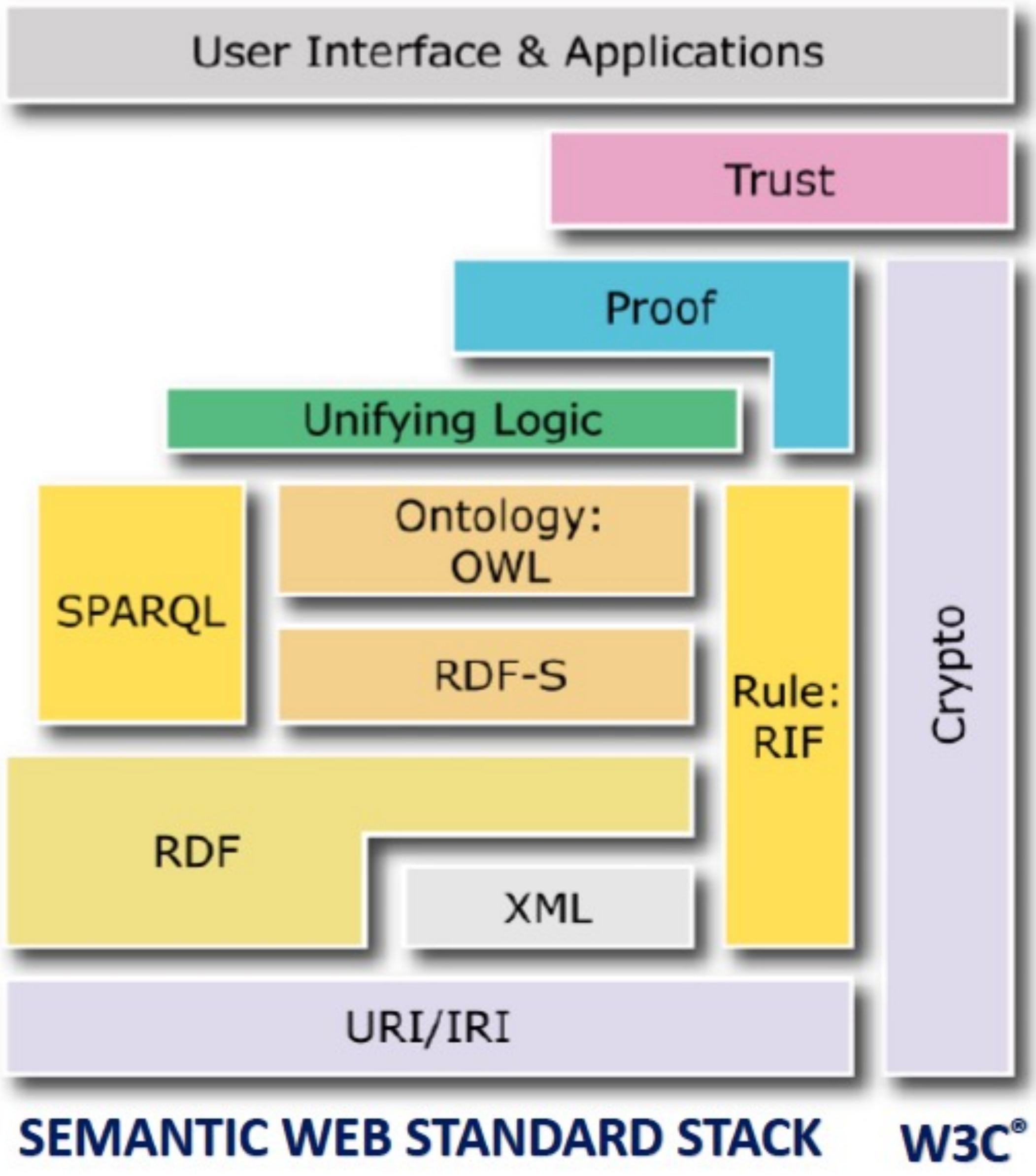
- A resource is anything that can be referred to by a URI (Uniform Resource Identifiers)
  - a web page, identified by an URL
  - a fragment of an XML document, identified by an element node of the document or an XPath expression
  - a web service
  - a thing, an object, a property, etc.

- Examples
  - http://www.example.org/file.html
    http://www.example.org/file.html#home
  - http://www.example.org/file2.xml#xpath(//q[@a=b])
  - http://www.example.org/form?a=b&c=d

THE TREE
THE DOOR
THE HOUSE
GARB CAN
SHIRT
PANTS
THE CAT
THE DOG

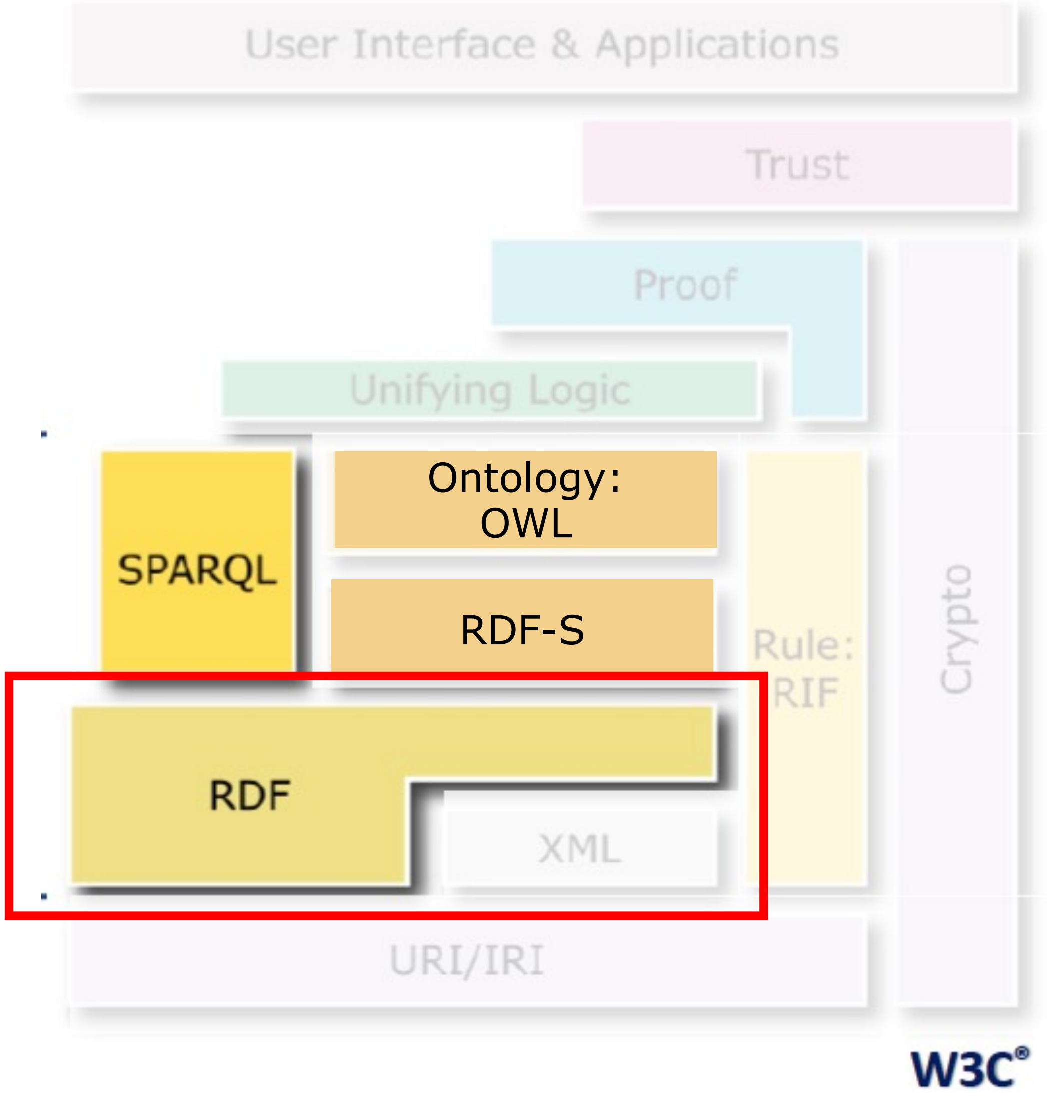
“Now! *That* should clear up a few things around here!”

The Semantic Web Standard Stack



The Semantic Web Standard Stack

- RDF: a very simple ontology language
- RDFS: Schema for RDF
  - Can be used to define richer ontologies
- OWL: a much richer ontology language



Resource Description Framework

- RDF stands for
  - Resource: pages, dogs, ideas...everything that can have a URI
  - Description: attributes, features, and relations of the resources
  - Framework: model, languages and syntaxes for these descriptions

- A W3C standard since 2004
- Description of arbitrary things

- A very simple ontology language
- Models ontology instances, ontology concepts, ontology relations

- RDF is the data model for the Semantic Web
  - provides a simple language for describing annotations about Web resources identified by URIs
  - these are facts

RDF Data Model

- A schema-less data model that features
  - unambiguous identifiers and
  - named relations between pairs of resources
- Data are represented as a set of triples (subject, predicate, object)
  - subject: a resource (identified by an URI)
  - predicate: a resource, representing a property (identified by an URI)
  - object: a resource (identified by an URI) or a literal (a constant value with some annotation)
- when the object is a literal, the triple expresses that a given subject has a given value for a given property
- RDF triples can be represented as a graph (RDF graph)

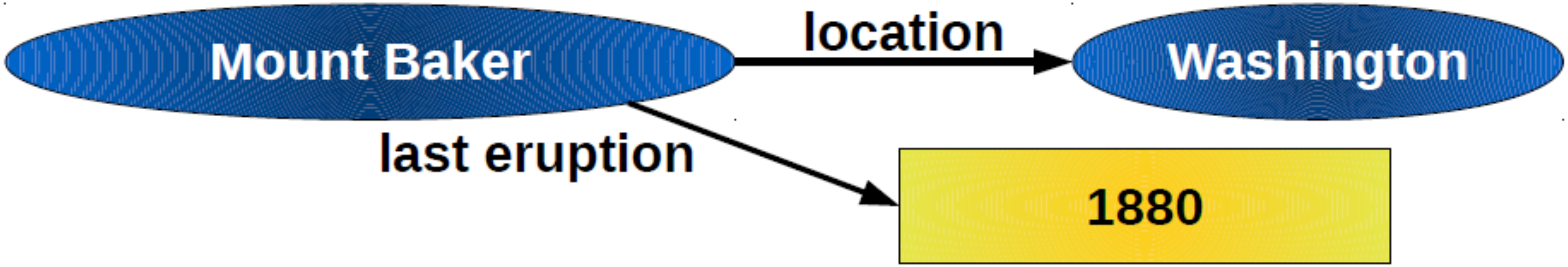
Example

resource

property

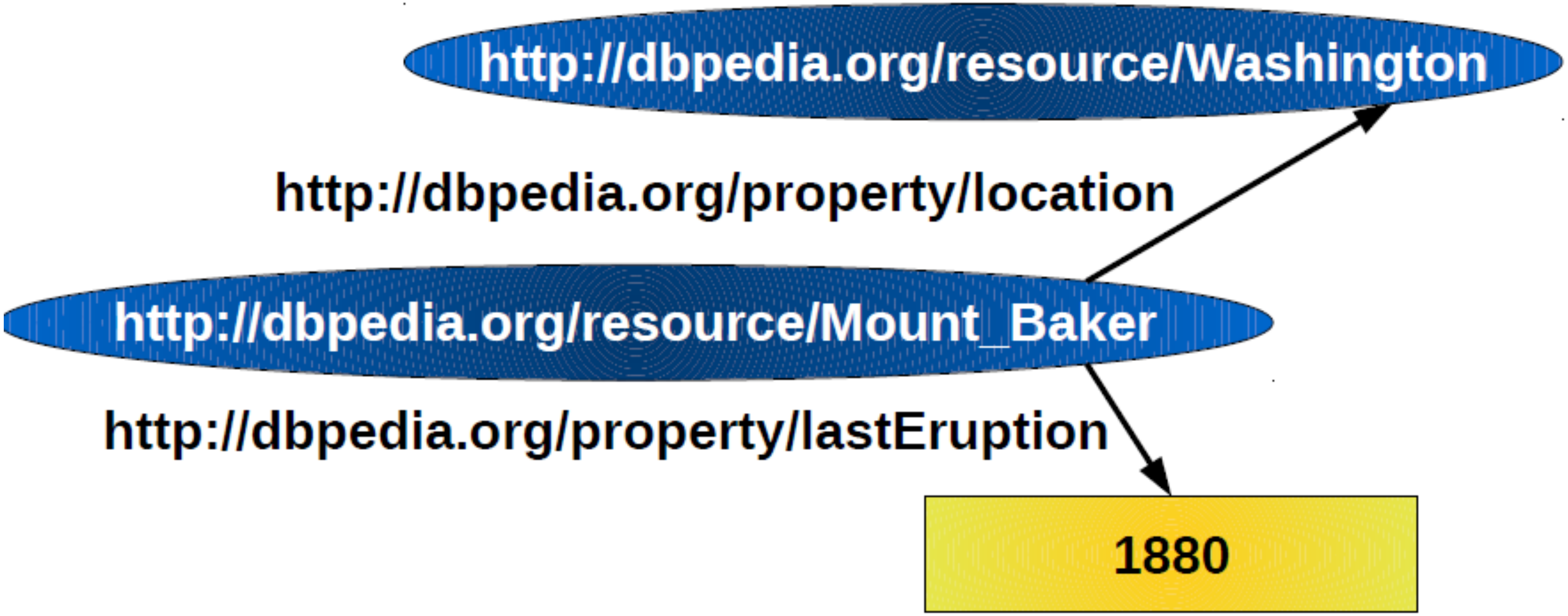
- **Example:**
  - (Mount Baker , last eruption , 1880)
  - (Mount Baker , location , Washington)

literal



RDF graph with URIs

- (http://dbpedia.org/resource/Mount_Baker,
  http://dbpedia.org/property/lastEruption, 1880)
- (http://dbpedia.org/resource/Mount_Baker,
  http://dbpedia.org/property/location,
  http://dbpedia.org/resource/Washington)


http://dbpedia.org/resource/Washington
http://dbpedia.org/property/location
http://dbpedia.org/resource/Mount_Baker
http://dbpedia.org/property/lastEruption
1880

RDF Data model: URI recap

- URI: Uniform Resource Identifier
  - a web page, identified by an URL
  - a fragment of an XML document, identified by an element node of the document or an XPath expression
  - a web service
  - a thing, an object, a property
  - …
- Examples
  - http://www.example.org/file.html
  - http://www.example.org/file.html#home
  - http://www.example.org/file2.xml#xpath(//q[@a=b])
  - http://www.example.org/form?a=b&c=d
  - http://dbpedia.org/resource/Berlin

DBLP Example

Basic bulding blocks

Basic bulding blocks

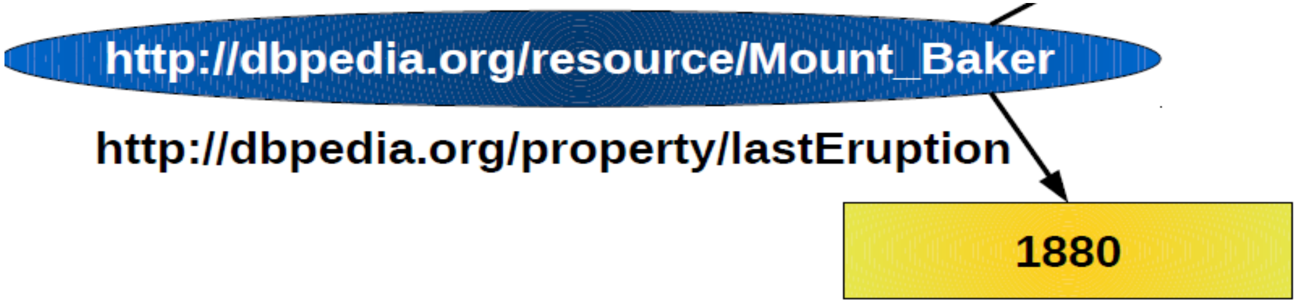
- Resources
  - denote things
  - are identified by a URI
  - can have one or multiple types

- Literals
  - are values like strings or integers
  - can only be objects, not subjects or predicates (graph view: they can only have incoming edges)
  - can have a datatype or a language tag (but not both)

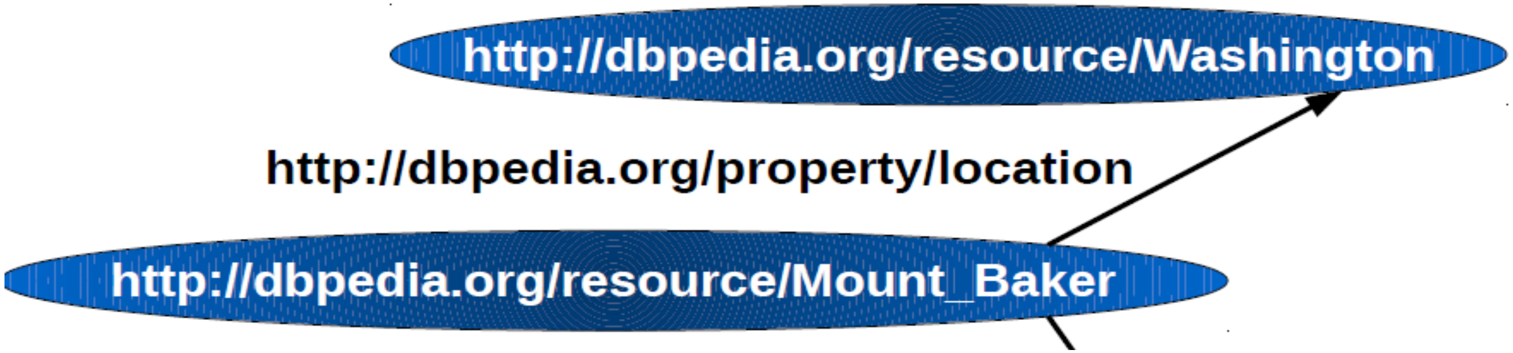
- Properties (Predicates)
  - Link resources to other resources and to literals

Resource versus Literal

- A literal is an atomic value
  - can only be object
  - i.e., a literal terminates always a graph

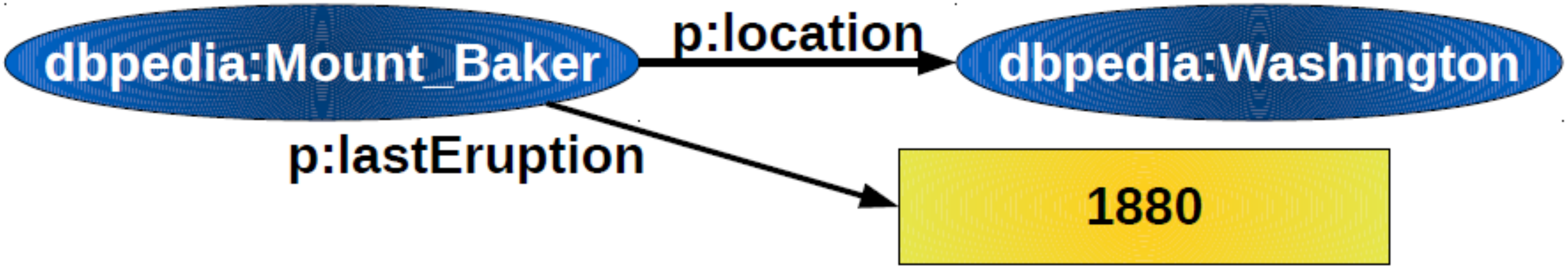

http://dbpedia.org/resource/Mount_Baker
http://dbpedia.org/property/lastEruption
1880

- A resource can be a subject itself

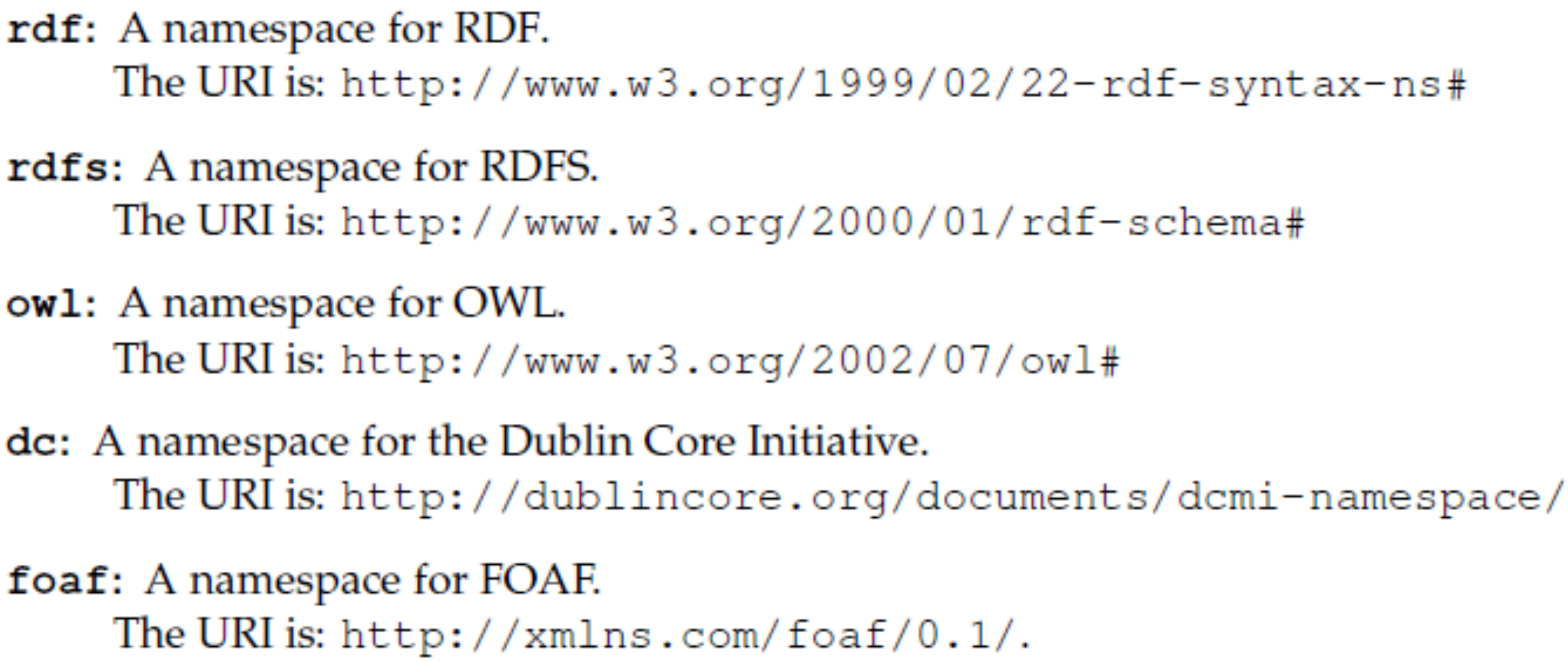

http://dbpedia.org/resource/Washington
http://dbpedia.org/property/location
http://dbpedia.org/resource/Mount_Baker

Example with namespaces

- **Using**
  - *dbpedia* for prefix *http://dbpedia.org/resource/*
  - *p* for prefix *http://dbpedia.org/property/*
- **we have**
  - (dbpedia:Mount_Baker, p:lastEruption, 1880)
  - (dbpedia:Mount_Baker, p:location, dbpedia:Washington)



Some standard namespaces



`rdf:` A namespace for RDF.
The URI is: `http://www.w3.org/1999/02/22-rdf-syntax-ns#`

`rdfs:` A namespace for RDFS.
The URI is: `http://www.w3.org/2000/01/rdf-schema#`

`owl:` A namespace for OWL.
The URI is: `http://www.w3.org/2002/07/owl#`

`dc:` A namespace for the Dublin Core Initiative.
The URI is: `http://dublincore.org/documents/dcmi-namespace/`

`foaf:` A namespace for FOAF.
The URI is: `http://xmlns.com/foaf/0.1/`.

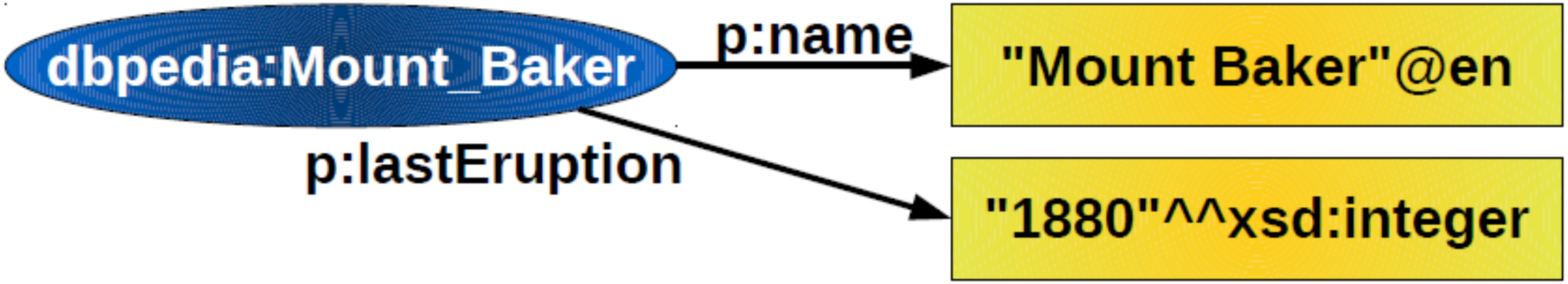
- **Dublin Core** is a popular standard in the field of digital libraries.
- The **Friend of a Friend (FOAF)** initiative aims at creating a “social” Web of machine-readable pages describing people, the links between them and the things they create and do.

Literals

- **Literals may occur in the object position of triples**
- **Represented by strings**
- **Literal strings interpreted by datatypes**
  - Datatype identified by a URI
  - Common to use the XML Schema datatypes
  - No datatype: interpreted as xsd:string
- **Untyped literals may have language tags (e.g. @de)**

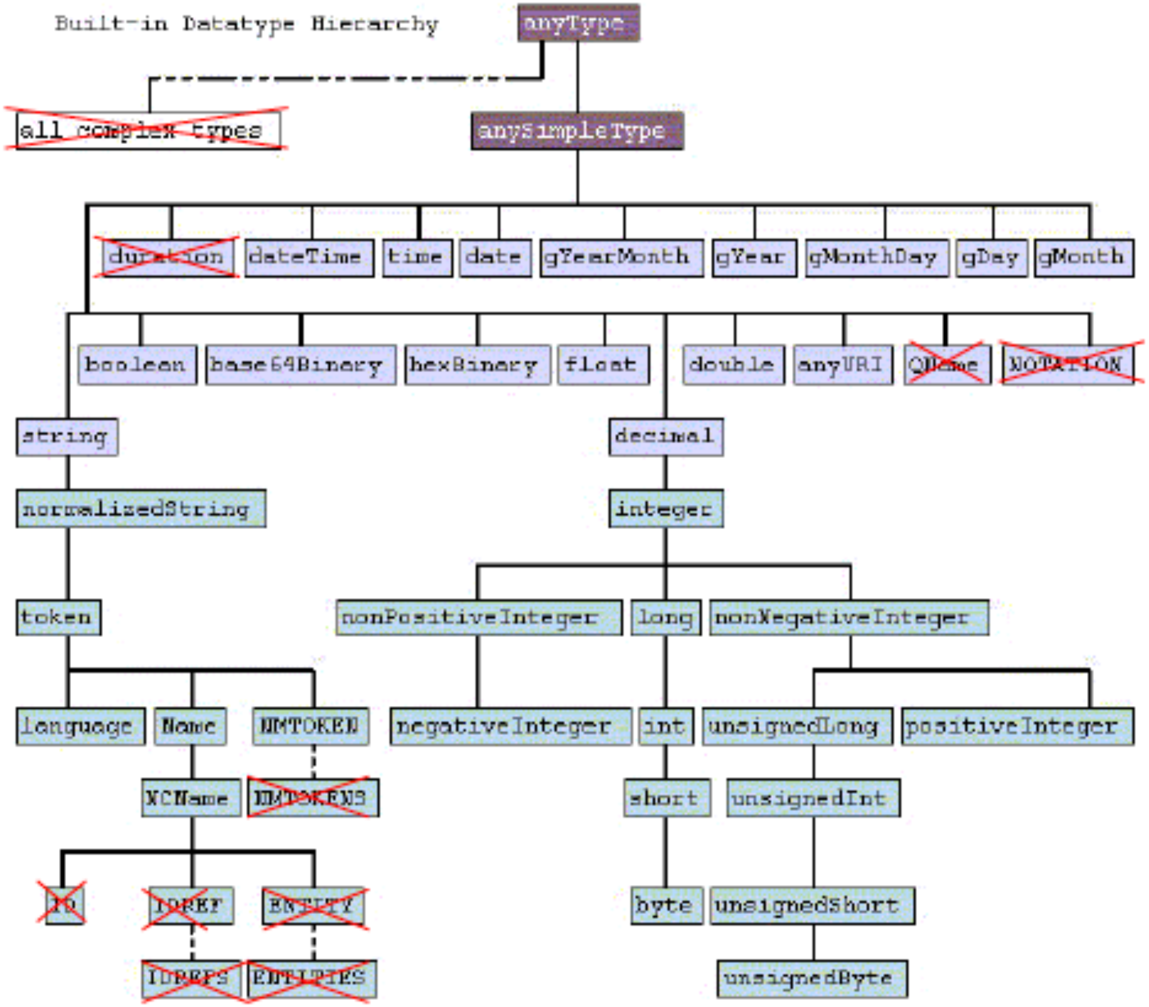
Language codes according to ISO 963

But also …''Monte Baker''@it



Literals: type

- (Almost) all XML Schema datatypes may be used
- Exception:
  - XML specific types
  - The underspecified type "duration"
  - sequence types



XML Schema Part 2: Datatypes Second Edition
http://www.w3.org/TR/xmlschema-2/

RDF Data Model – Some «schema» information

- In RDF, one can distinguish between individuals (objects) and properties (relationships)

- This is not mandatory but it can be done using two RDF resources:
  - rdf:type, which can be used as a property
  - rdf:Property, which can be used as a resource

- Still triplets but providing a very light schema information

- Example
  - < location rdf:type rdf:Property: > the resource location is a property
  - <Mount_Baker rdf:type Volcano>: the resource Mount_Baker is an instance of class Volcano

Blank nodes

- Sometimes, you may not precisely know the resource which is involved in some relationship with some other resources
- but you do know that the relationship exists
- two options:
  - create an extra URI, but in this case the resource will be visible on the Web
  - create an "internal" resource, visible only to your set of triples, in terms of a blank node

Blank nodes

- A blank node (or anonymous resource) is a subject or an object in an RDF triplet or an RDF graph that is not identified by a URI and is not a literal

- A blank node is referred to by a notation _:p where p is a local name that can be used in triplets (in the context of the same RDF graph) for stating several properties of the corresponding blank node

- Blank nodes require attention when merging
  - blanks nodes with identical nodeIDs in different graphs are different

Example

RDF triples

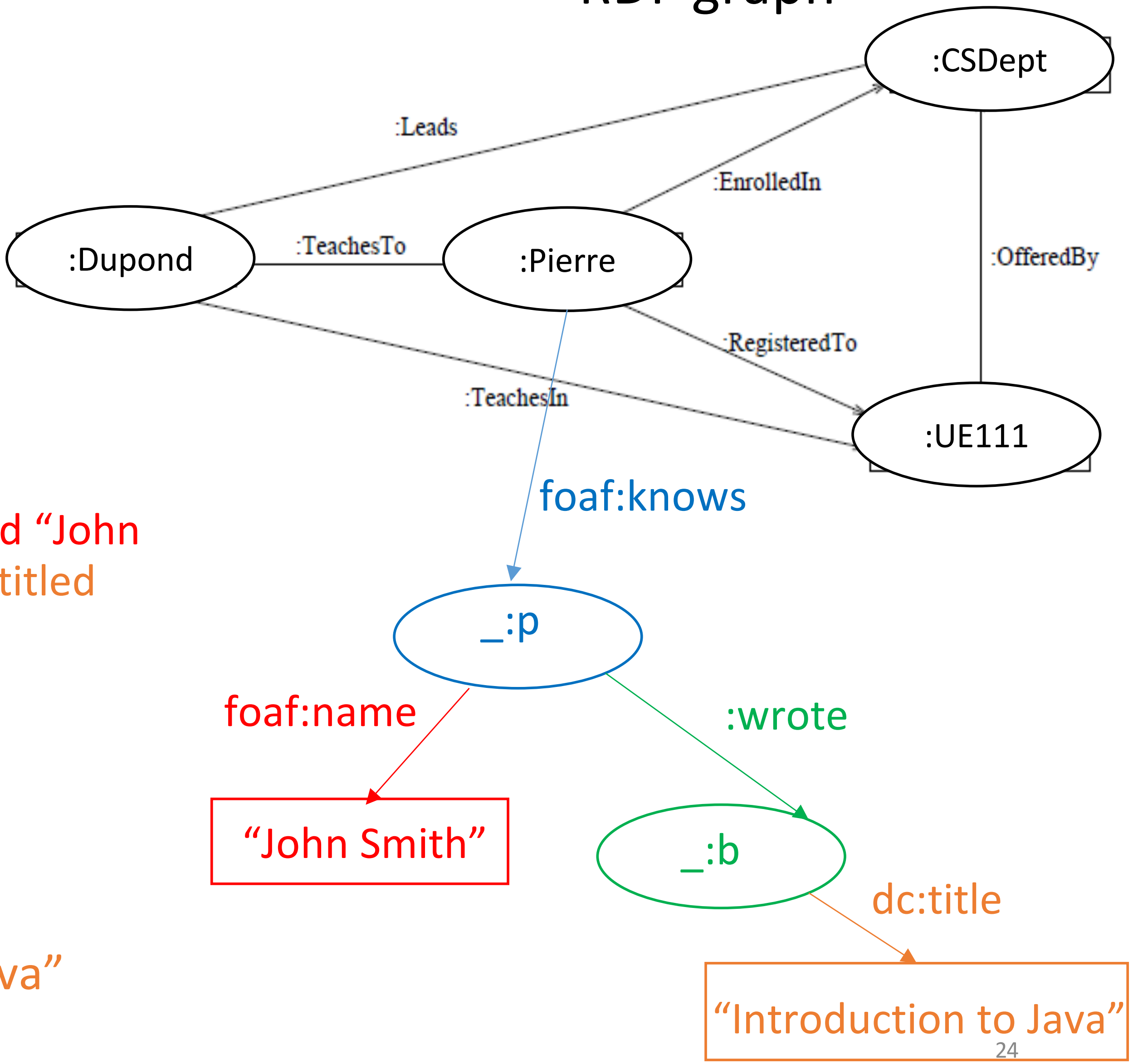
⟨ :Dupond :Leads :CSDept ⟩
⟨ :Dupond :TeachesIn :UE111 ⟩
⟨ :Dupond :TeachesTo :Pierre ⟩
⟨ :Pierre :EnrolledIn :CSDept ⟩
⟨ :Pierre :RegisteredTo :UE111 ⟩
⟨ :UE111 :OfferedBy :CSDept ⟩

You want to add

Pierre knows someone named “John Smith” that wrote a book entitled “Introduction to Java”

:Pierre foaf:knows _:p
_:p foaf:name “John Smith”
_:p :wrote _:b
_:b dc:title “Introduction to Java”

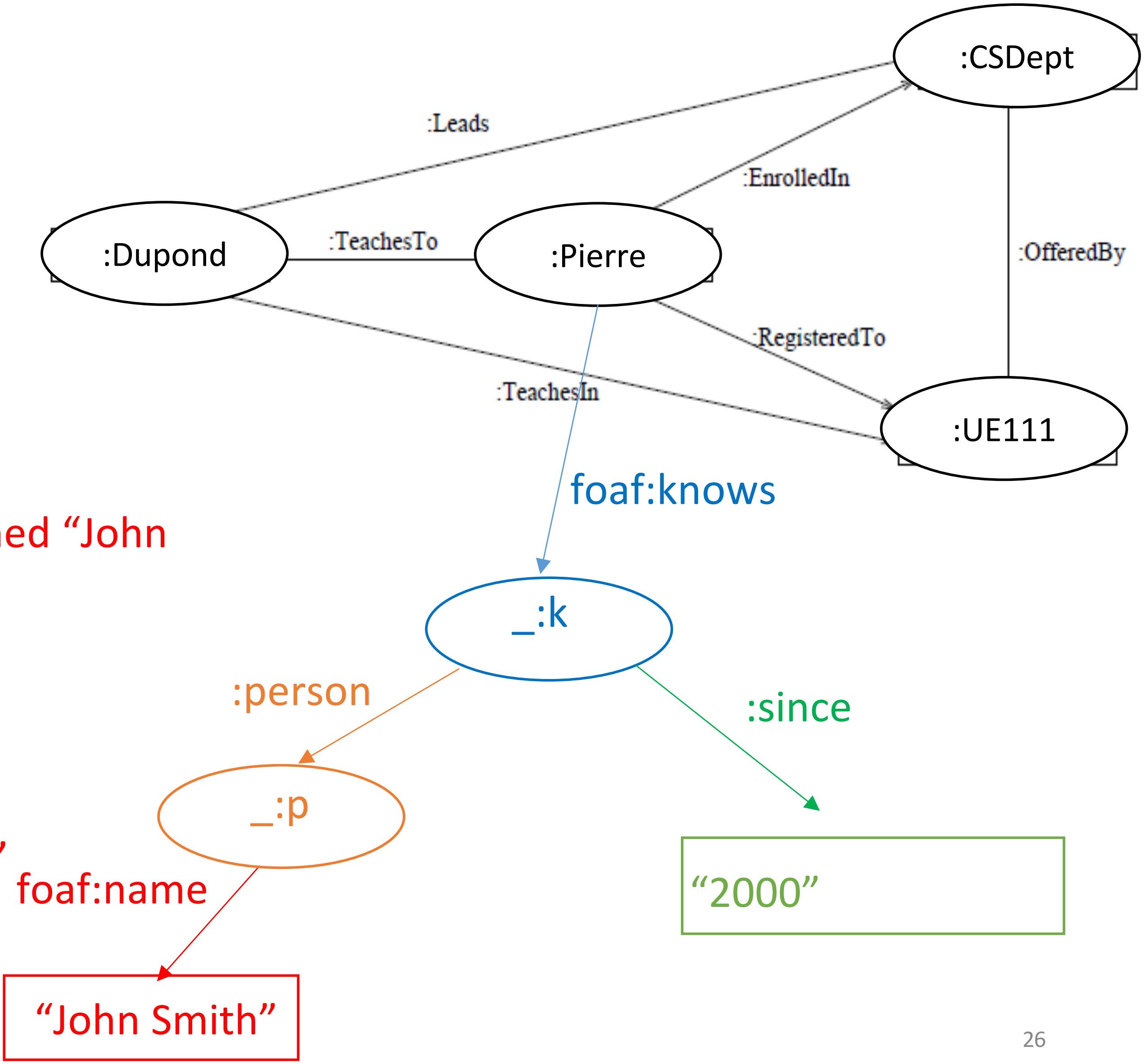
RDF graph



Blank nodes and n-ary associations

- RDF predicates always connect a subject and an object
- In the sense of predicate logic, they are binary predicates
  - Pierre[1] knows someone[2]
  - knows(Pierre, someone)
  - :Pierre foaf:knows _:p .
- Sometimes, n-ary predicates are needed
  - Pierre[1] knows someone[2] since 2000[3]
  - knows(Pierre, someone, 2000)
- N-ary predicates can be modeled using blank nodes

Example



You want to add

Pierre knows someone named "John Smith" since year 2000

:Pierre foaf:knows _:k
_:k :person _:p
_:p foaf:name "John Smith"
_:k :since ''2000''

Reification in RDF

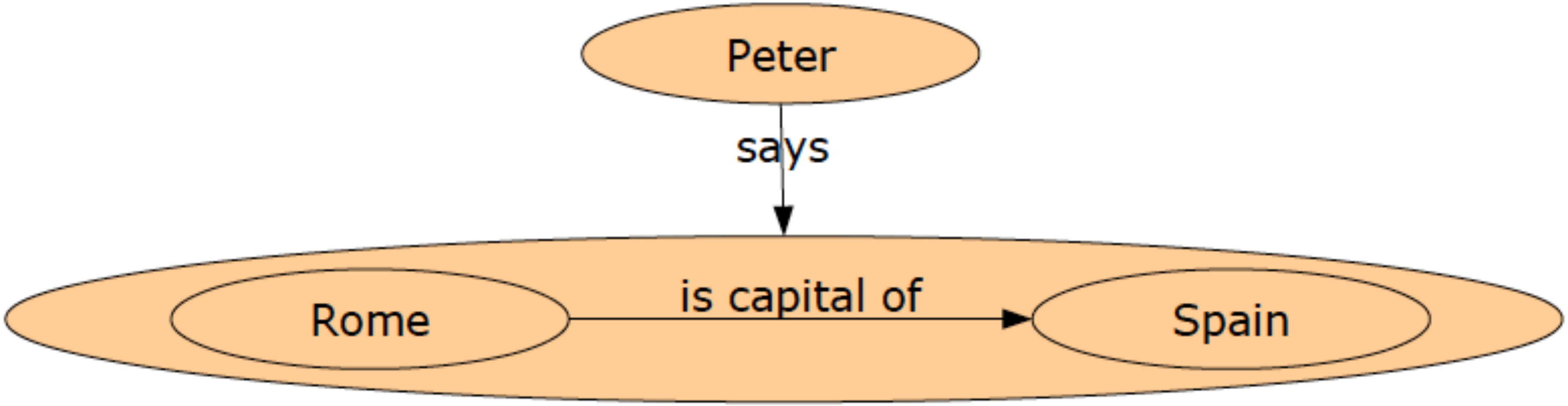
- Latin res ("Thing"), facere ("make")
  - an explication
  - making a statement, an opinion etc.

- In RDF: Statements about statements

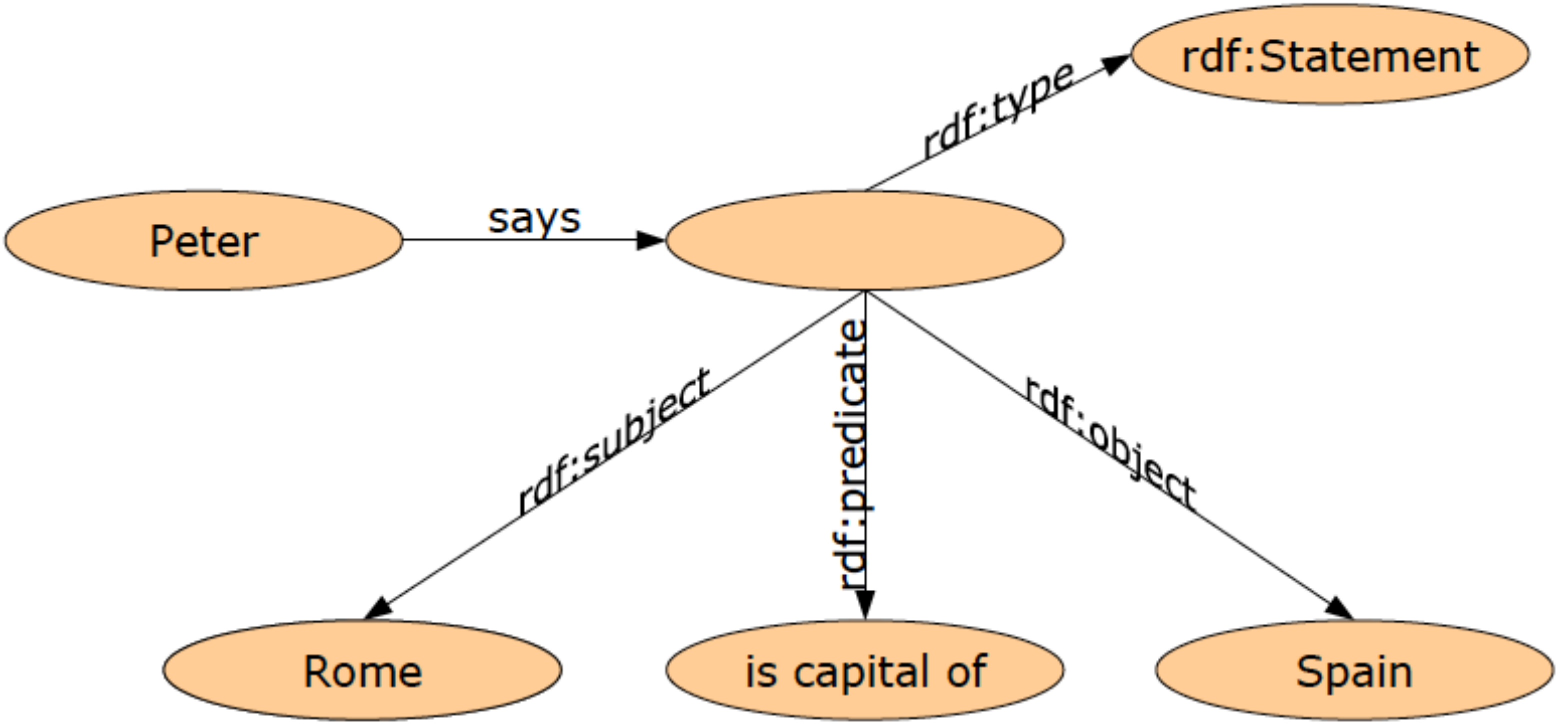
- "Peter says that Rome is the capital of Spain.»

- Implementation:
  - RDF Statements are considered resources themselves
  - Can be subject or object of other statements
  - Reification can have multiple levels
  - “Peter says that Wikipedia states that Rome is the capital of Spain.”

Example



Example


rdf:Statement
rdf:type
Peter
says
rdf:subject
rdf:predicate
rdf:object
Rome
is capital of
Spain

RDF - Syntaxes

Abstract and concrete syntax

- Triples provides an abstract RDF syntax
- Several concrete languages have been provided to represent the same information
  - Triple notation
  - Turtle: simple, human readable notation for listing RDF tuples, introduce some shorthands
  - RDF/XML
  - ...

Triple notation

- A W3C standard (2004)
- Triples consist of a subject, predicate, and object
- An RDF document is an unordered set of triples

- Simple triple:

```
<http://dbpedia.org/resource/Mount_Baker>
<http://dbpedia.org/property/location>
<http://dbpedia.org/Washington> .
```

- Literal with language tag:

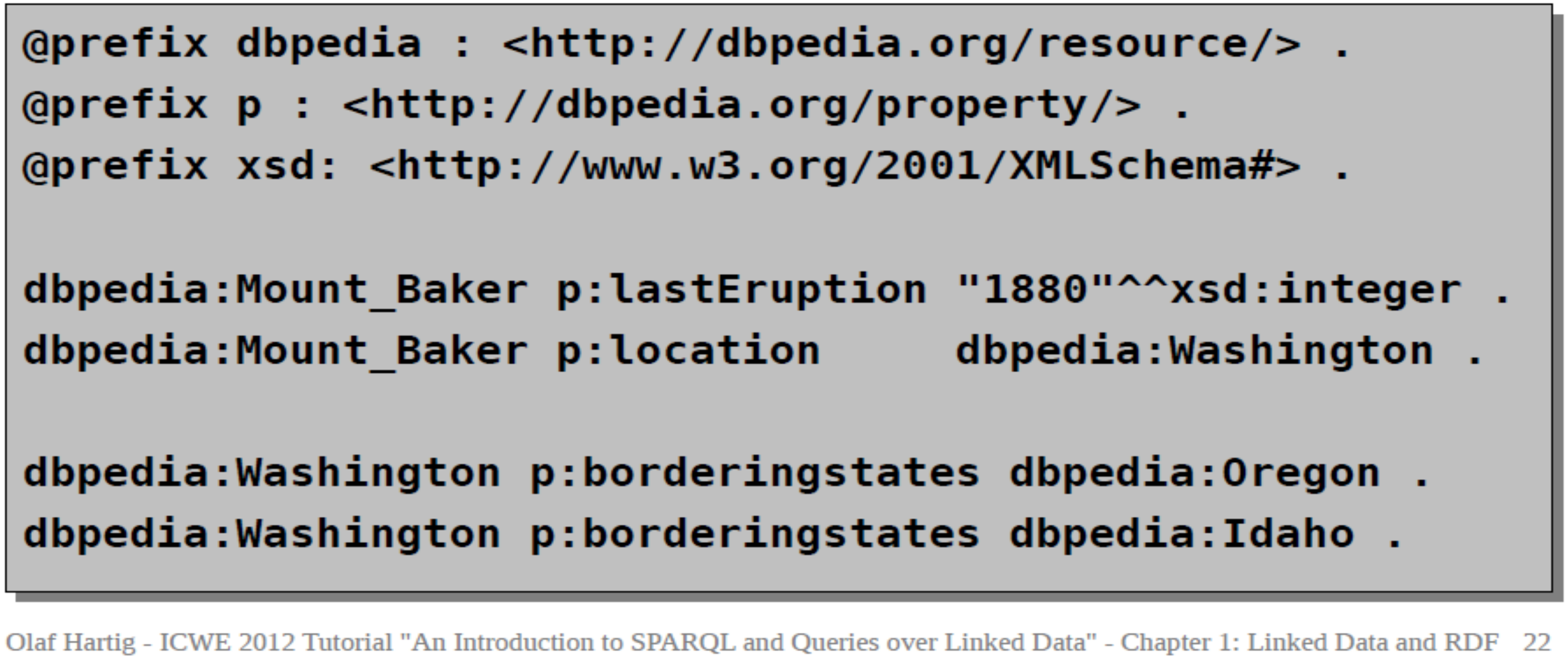
```
<http:// dbpedia.org/resource/Mount_Baker>
<http://dbpedia.org/property/name>
"Mount Baker"@en .
```

- Typed literal:

```
<http://dbpedia.org/resource/Mount_Baker>
<http://dbpedia.org/property/lastEruption>
"1800"^^<http://www.w3.org/2001/XMLSchema#integer> .
```

Turtle notation

- A simplified triple notation
- Namespaces as central definition



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

dbpedia:Mount_Baker p:lastEruption "1880"^^xsd:integer .
dbpedia:Mount_Baker p:location     dbpedia:Washington .

dbpedia:Washington p:borderingstates dbpedia:Oregon .
dbpedia:Washington p:borderingstates dbpedia:Idaho .
```

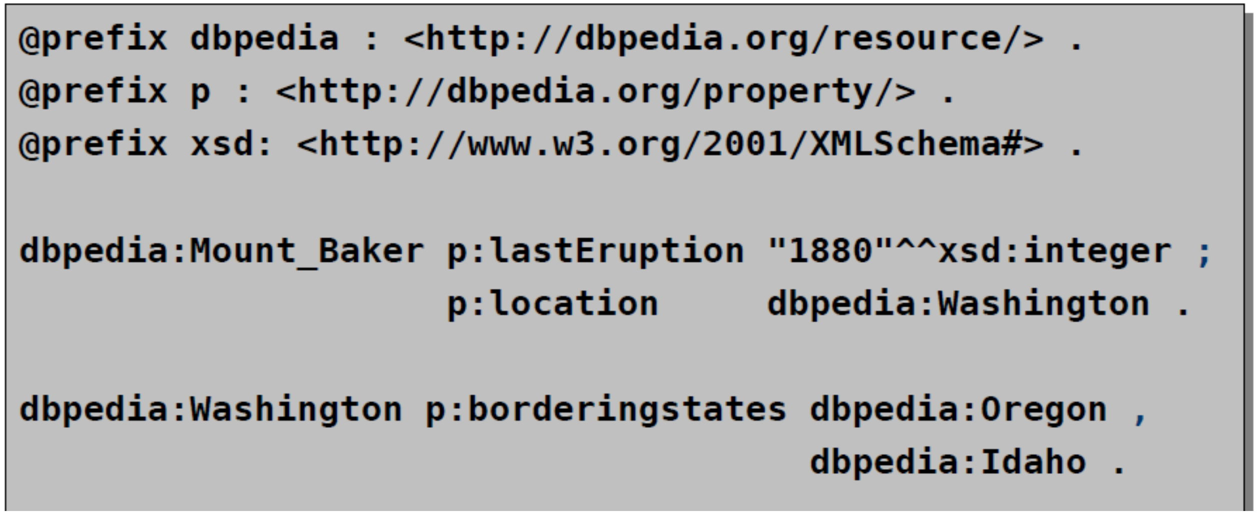
Olaf Hartig - ICWE 2012 Tutorial "An Introduction to SPARQL and Queries over Linked Data" - Chapter 1: Linked Data and RDF 22

- A default namespace

```
@prefix :http://www.example.org
```

Turtle notation

- A simplified triple notation
- Triples sharing
  - the same subject: lists separated by «;»
  - the same subject+predicate: lists separated by «,»



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

dbpedia:Mount_Baker p:lastEruption "1880"^^xsd:integer ;
                    p:location     dbpedia:Washington .

dbpedia:Washington p:borderingstates dbpedia:Oregon ,
                                     dbpedia:Idaho .
```

- Shorthand notation for rdf:type:

  `dbpedia:Mount_Baker a dbpedia:StratoVolcano`

Turtle notation: blank nodes

- **Variant 1:** explicitly named with an underscore

```
:Dieter_Fensel dc:creator _:x .
_:x   a :Book ;
      dc:subject "Semantic Web" .
```

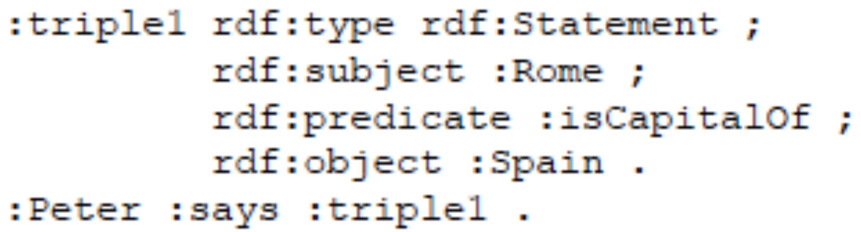
- **Variant 2:** unnamed with square brackets

```
:Dieter_Fensel dc:creator [ a :Book;
                            dc:subject "Semantic Web" ].
```

- Notes:
  - both are equivalent
  - changing blank node names does not change the semantics!

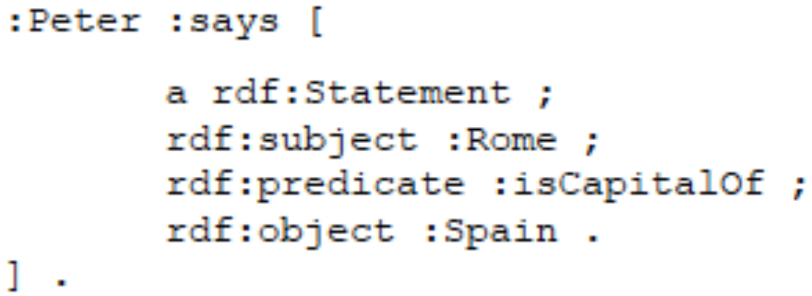
Turtle notation: reification

- Variant 1: Named Statement (with URI)



```
:triple1 rdf:type rdf:Statement ;
         rdf:subject :Rome ;
         rdf:predicate :isCapitalOf ;
         rdf:object :Spain .
:Peter :says :triple1 .
```

- Variant 2: Unnamed Statement (Blank Node)



```
:Peter :says [

       a rdf:Statement ;
       rdf:subject :Rome ;
       rdf:predicate :isCapitalOf ;
       rdf:object :Spain .
] .
```

Other notations

- XML/RDF (W3C standard)
  - Encodes RDF in XML
  - Suitable for machine processing (plenty of XML tools!)
- JSON-LD (W3C standard)
  - Encodes RDF in JSON
  - Useful for serializing RDF data

RDF Semantics

RDF Semantics

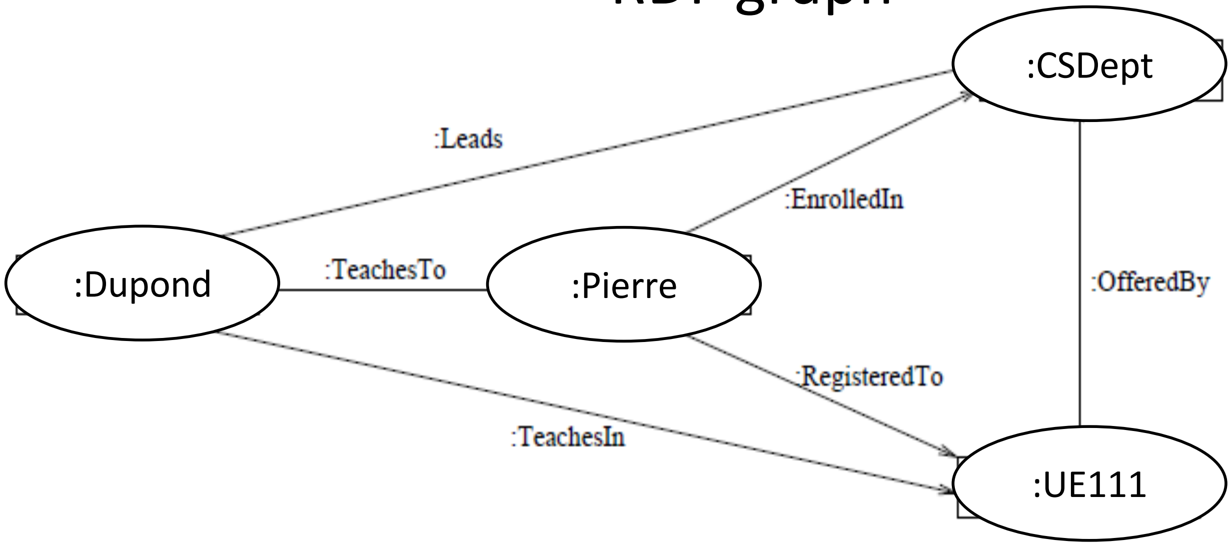
- A triplet $\langle s \quad P \quad o \rangle$ is interpreted in first-order logic (FOL) as a fact $P(s, o)$
- Example:

(atomic formula without variables)

RDF triples

$\langle$:Dupond :Leads :CSDept $\rangle$
$\langle$:Dupond :TeachesIn :UE111 $\rangle$
$\langle$:Dupond :TeachesTo :Pierre $\rangle$
$\langle$:Pierre :EnrolledIn :CSDept $\rangle$
$\langle$:Pierre :RegisteredTo :UE111 $\rangle$
$\langle$:UE111 :OfferedBy :CSDept $\rangle$

RDF graph



RDF semantics

```
Leads(Dupond, CSDept)
TeachesIn(Dupond, UE111)
TeachesTo(Dupond,Pierre)
EnrolledIn(Pierre, CSDept)
RegisteredTo(Pierre, UE111)
OfferedBy(UE111, CSDept)
```

RDF Semantics

- Blank nodes, when they are in place of the subject or the object in triplets, are interpreted as existential variables
- Therefore a set of RDF triplets, possibly with blank nodes as subjects or objects, is interpreted as a conjunction of positive literals in which all the variables are existentially quantified
- Giving a FOL semantics to triplets in which the predicates can be blank nodes is also possible but a little bit tricky

Example

- Pierre knows someone named “John Smith” wrote a book entitled “Introduction to Java”

```
  :Pierre foaf:knows _:p
_:p foaf:name “John Smith”
_:p wrote _:b
_:b dc:title “Introduction to Java”
```

$$\exists p \exists b[knows(Pierre,p) \wedge name(p,\text{“John Smith”}) \wedge wrote(p,b) \wedge title(b,\text{“Introduction to Java”})]$$

RDF vs Labeled Property Graph

RDF vs Labeled property graphs

- The logical model is graph-based
- What are the differences between NoSQL labeled property graphs (LPG) and RDF graphs?

- More details at
  - https://neo4j.com/blog/rdf-triple-store-vs-labeled-property-graph-difference/
  - https://allegrograph.com/articles/rdf-graph-vs-property-graph-the-graph-show/

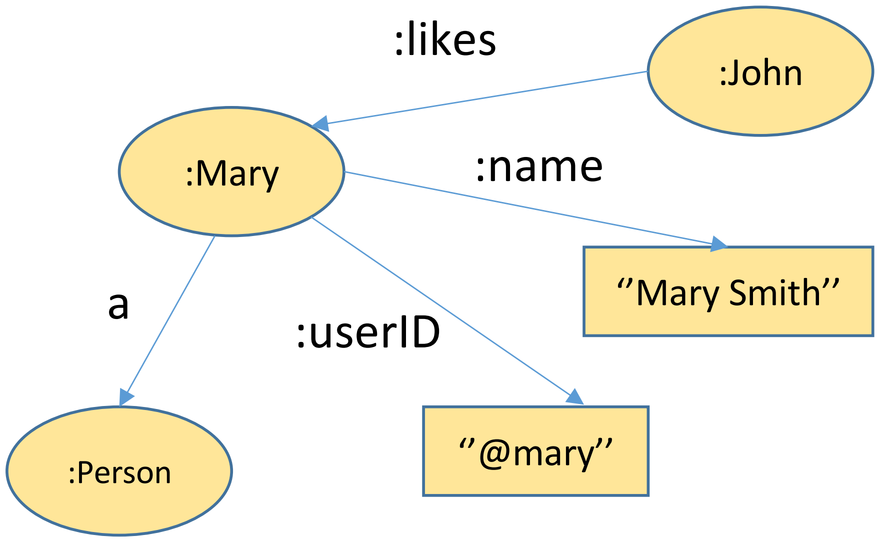
RDF vs Labeled property graphs: general structure

- LPG are more compact

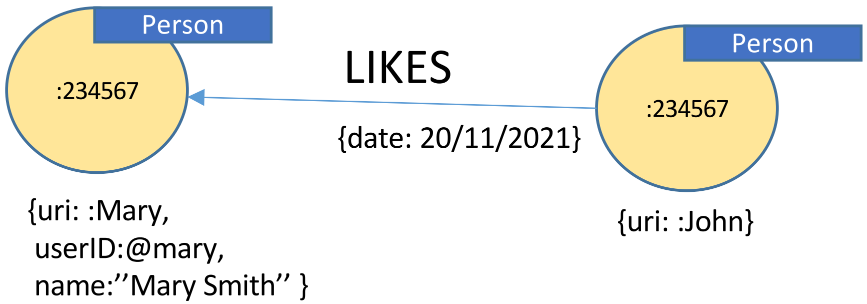
- In LPG, nodes and edges have an internal structure representing part of the information as key-value pairs

Example

- Mary is a person
- Mary user ID is @mary
- Mary name is ''Mary Smith''
- John likes Mary



- There is a person that is described by: her name, Mary Smith, her user ID, @mary. She has a globally unique identifier :Mary
- There is another person with a globally unique identifier, :John
- :John likes :Mary in date 20/11/2021



RDF vs Labeled property graphs: edge instances

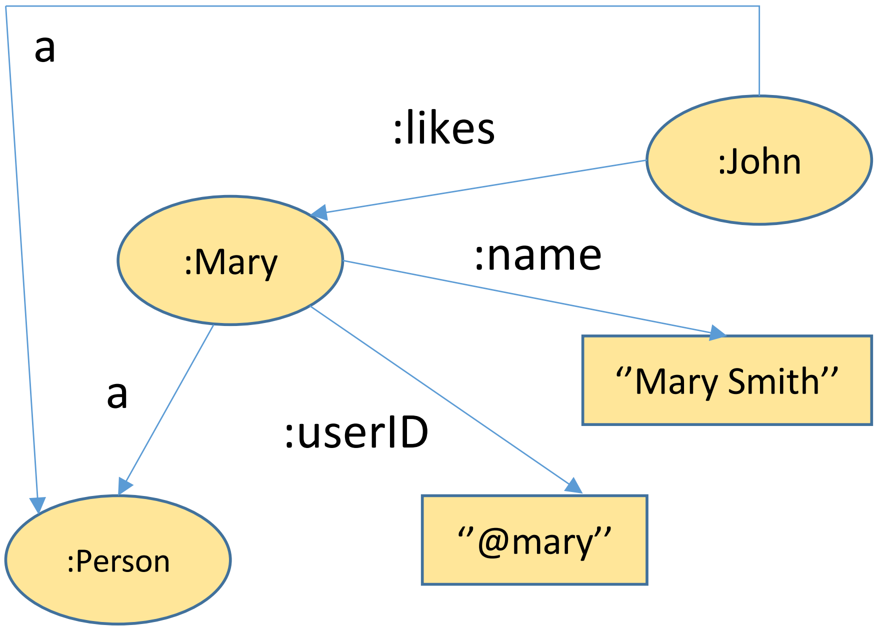
- RDF does not uniquely identify instances of relationships of the same type

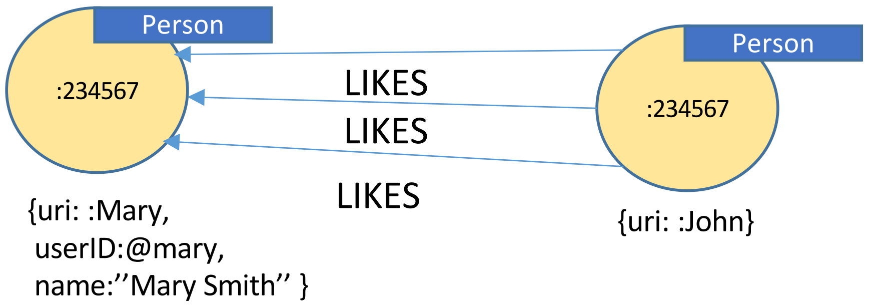
- It's not possible to have connections of the same type between the same pair of nodes because that would represent exactly the same triple, with no extra information

Example

- Given two nodes, one property resource can connect them just once

- Three instances of the same association between the same two nodes





RDF vs Labeled property graphs: attributes associated with edges

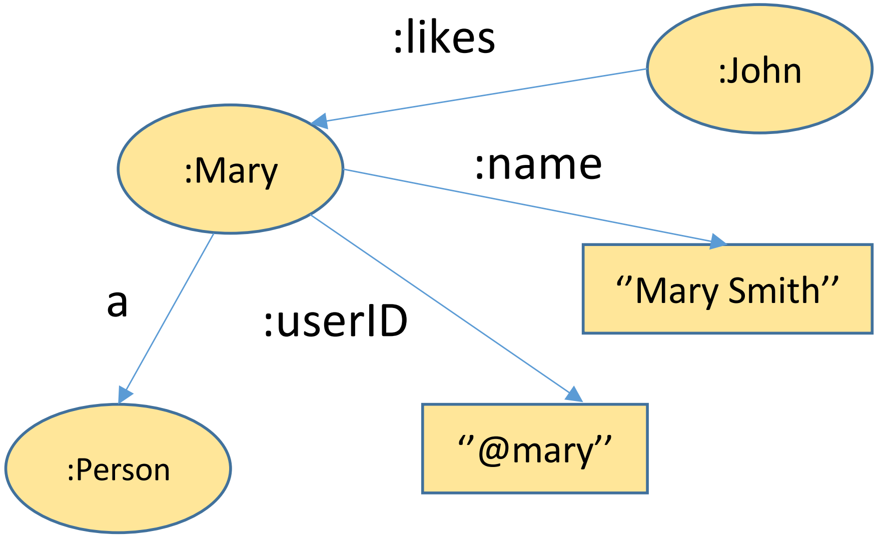
- Inability to qualify instances of relationships

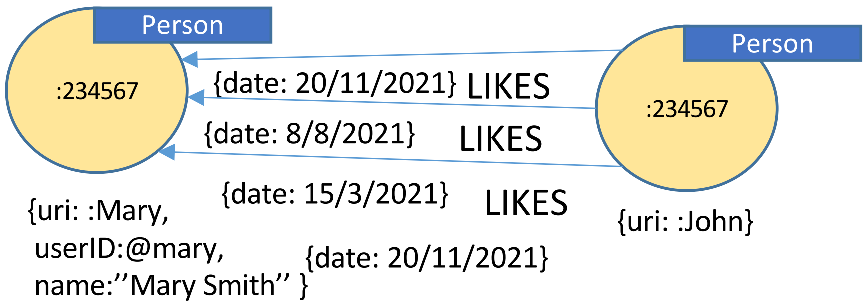
- because you can’t identify these unique instances in RDF, you cannot qualify them or give them attributes

Example

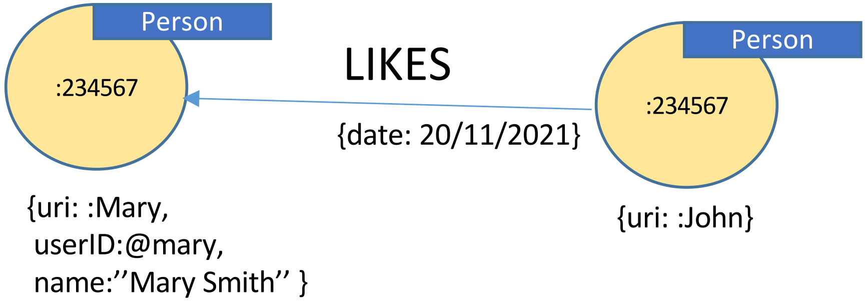
- You can add a triple like :likes :givenAtDate ''20/22/2011'' but this represent a general property of resource :like and not the specific instance used to connect :Mary and :John

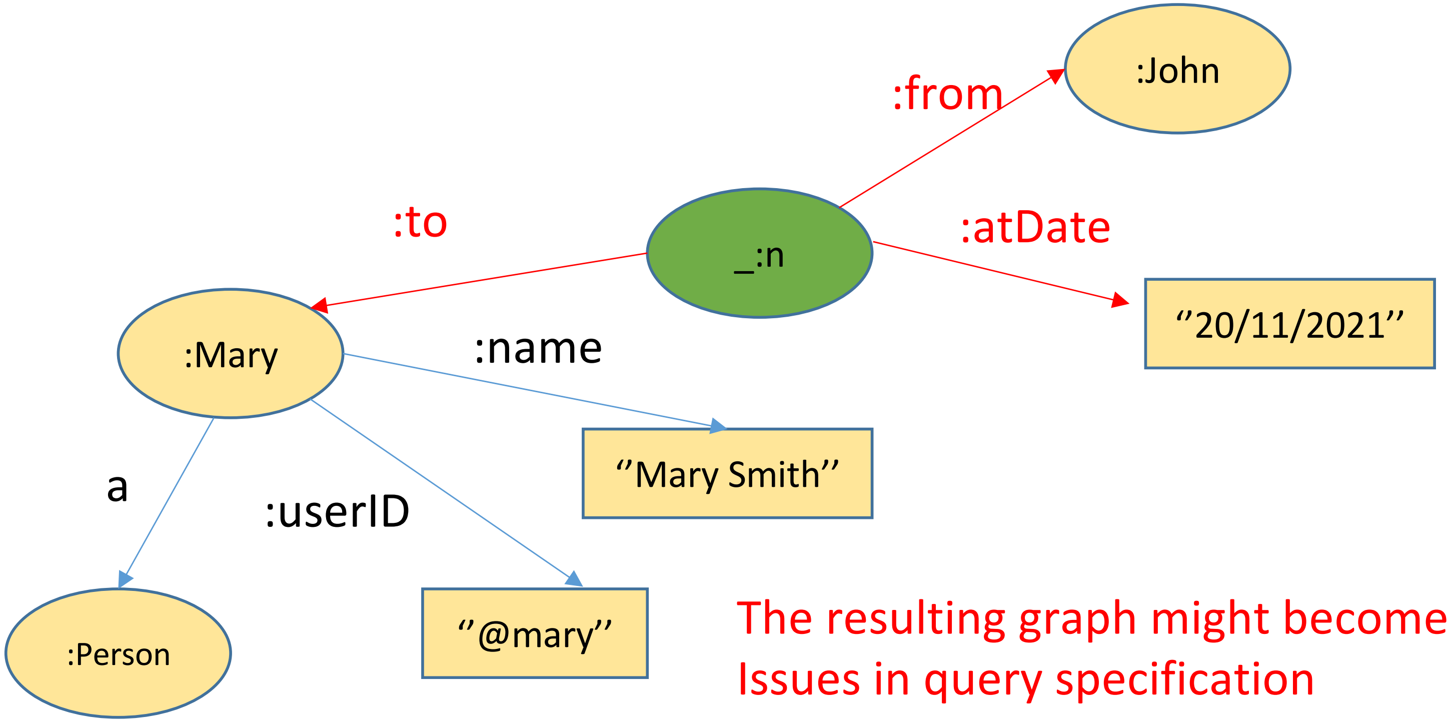
- Each 'likes' association, can be qualified by the reference date for the association





Solution 1: add new nodes



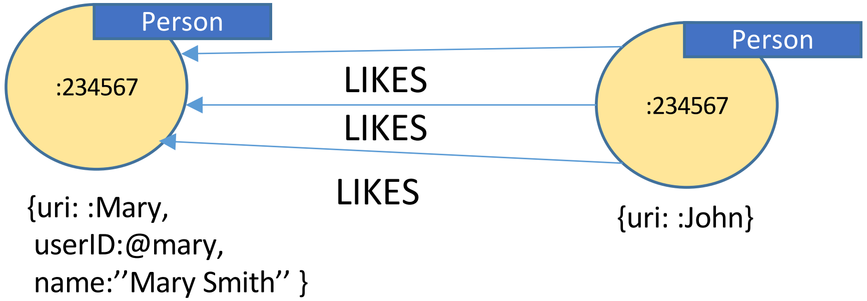


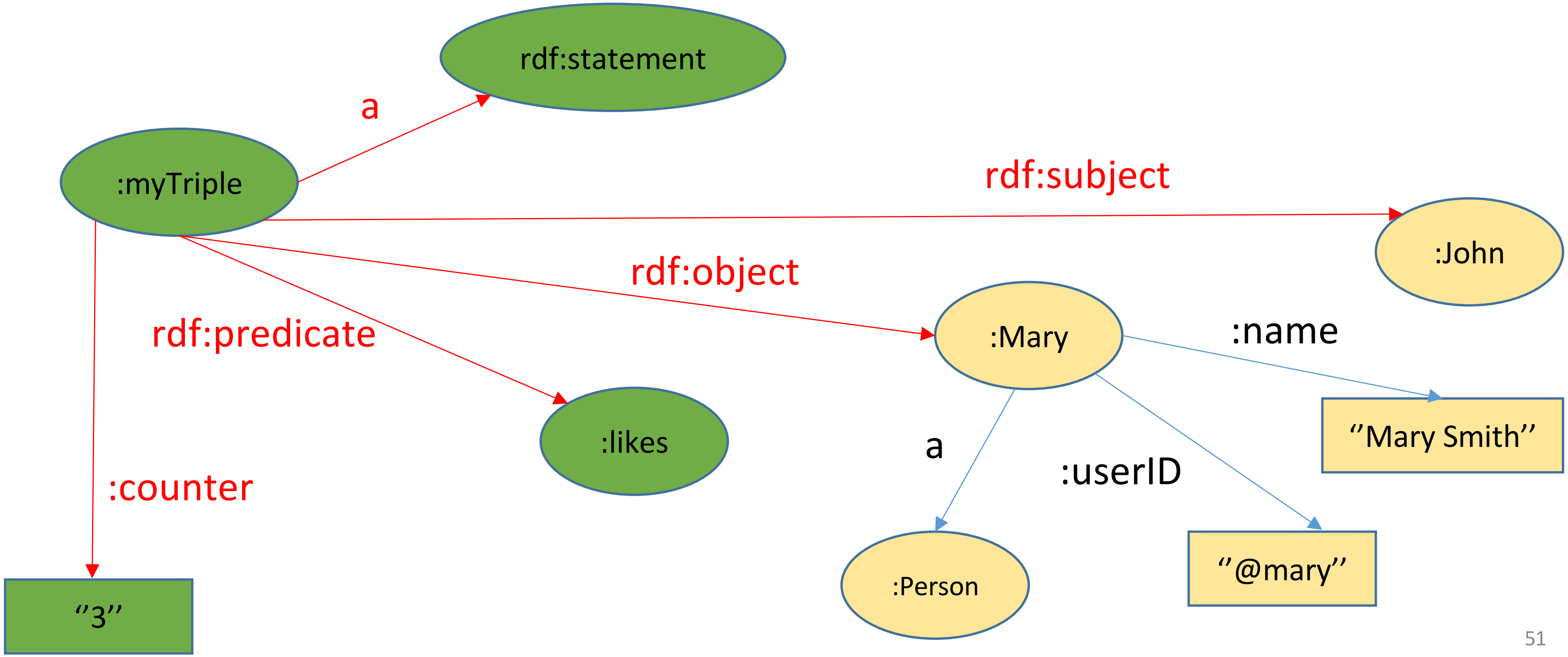
The resulting graph might become quite complex

Issues in query specification

Solution 2: reification

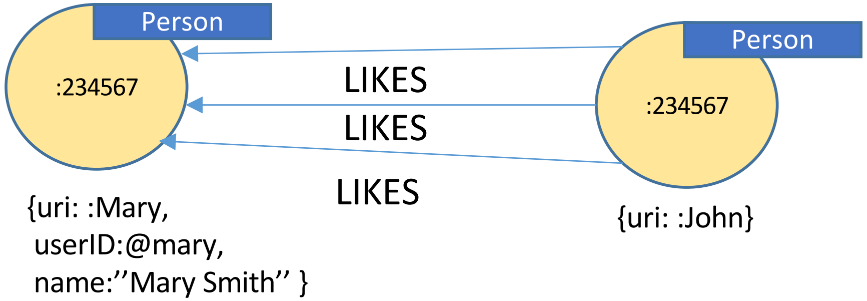
With reification, we create a metagraph on top of our graph that represents the statement that we have here

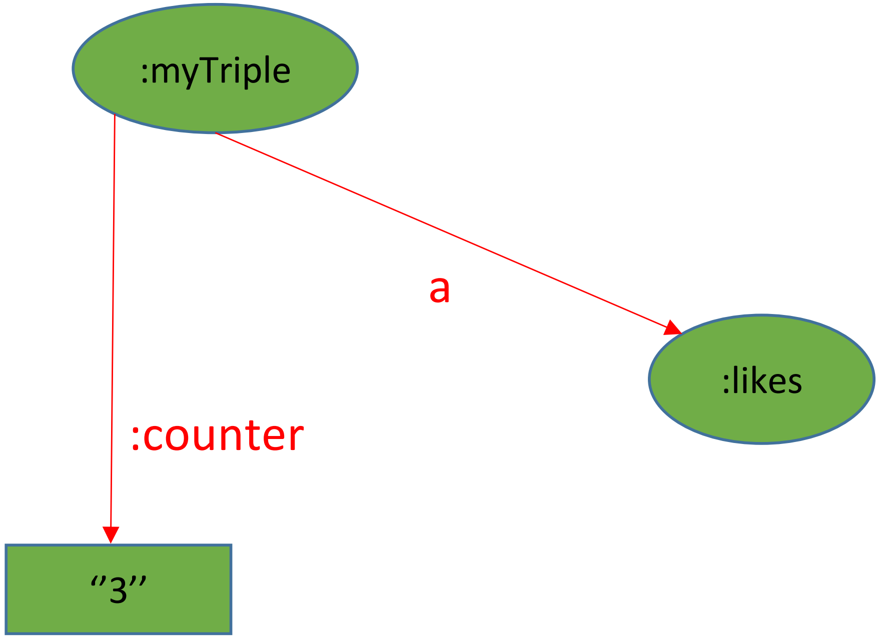


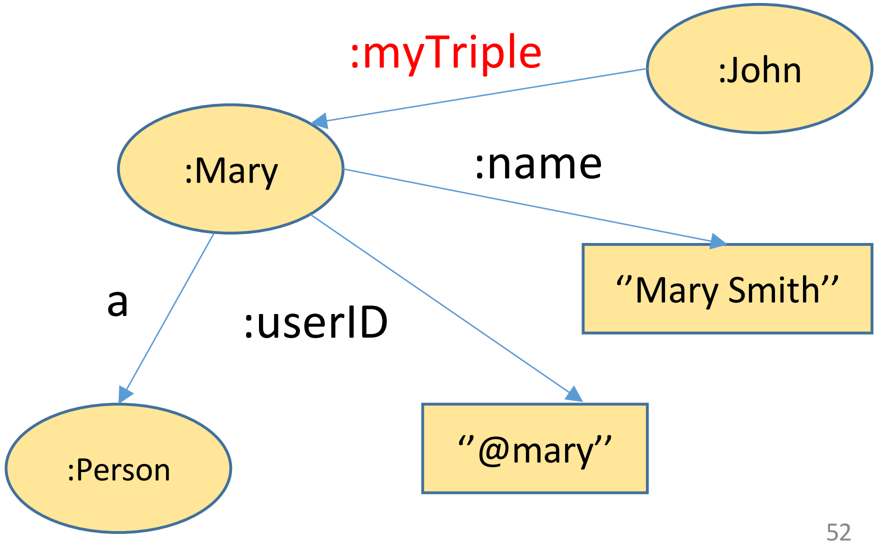


Solution 3: Singleton property

Similar to solution 2, more compact but two distint graphs



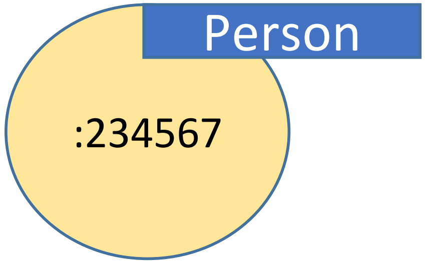




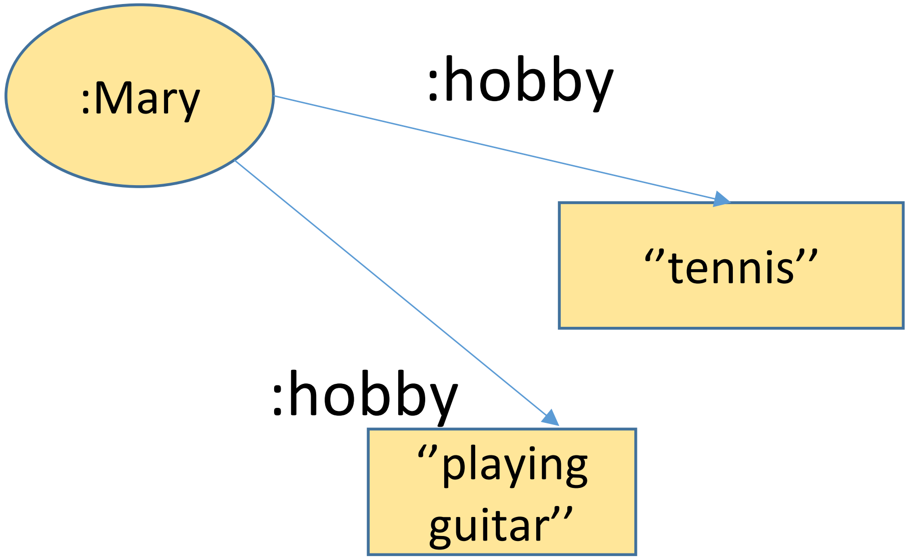
RDF vs Labeled property graphs: multivalued properties

- RDF can have multivalued properties
- In LPG you use arrays

Example



{uri: :Mary,
 hobbies: [“tennis’’, “playing guitar’’] }



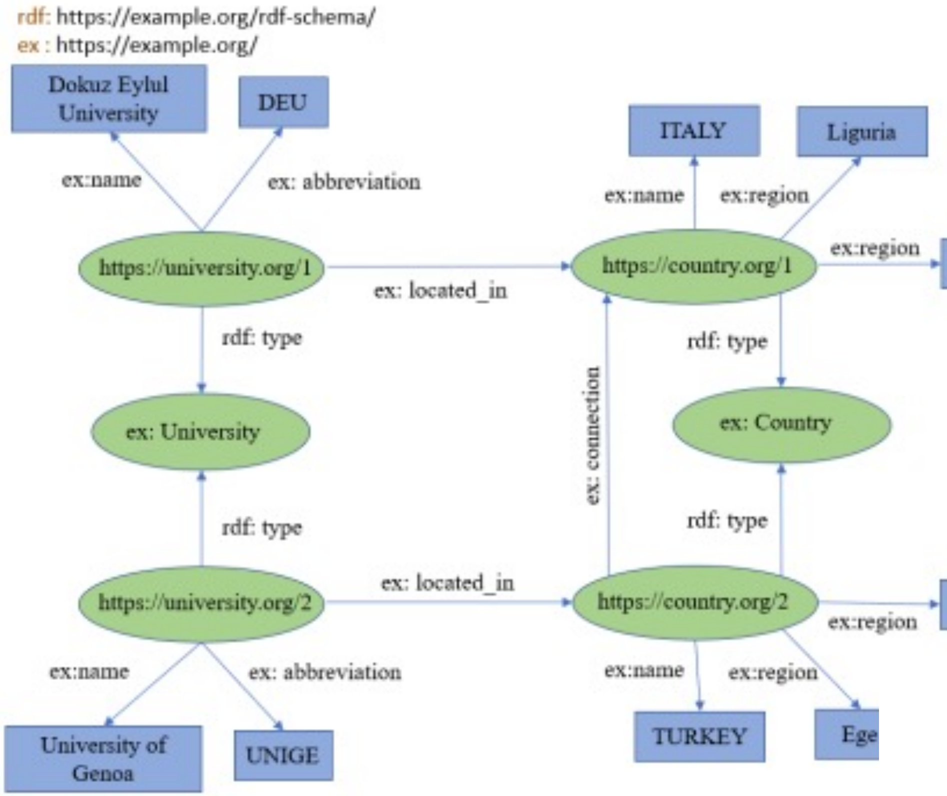
Property graph vs RDF Main differences

From L10-
Graph Data Management

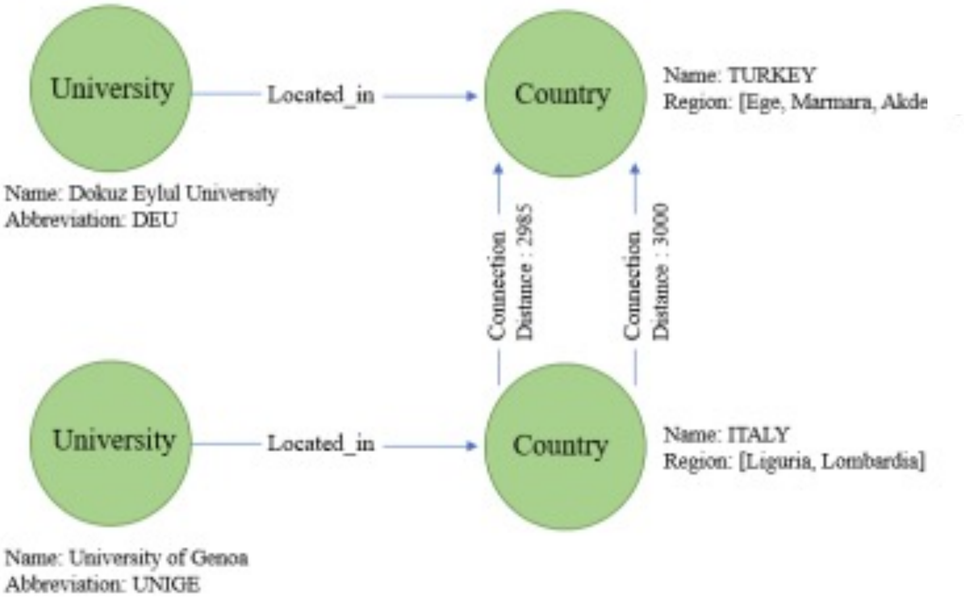
- In RDF
  - Each property need to be modeled as a new, unique, node
  - No properties for relationships
  - No multiple relationships of the same type between the same nodes (they can exist in property graphs which are actually multigraphs)
  - Array-valued vs multivalue properties

Example LPG vs RDF

From L10-
Graph Data Management



Is connection information exactly the same?
How would you modify the RDF graph?


University
Located_in
Country
Name: TURKEY
Region: [Ege, Marmara, Akde
Name: Dokuz Eylul University
Abbreviation: DEU
Connection
Distance : 2985
Connection
Distance : 3000
University
Located_in
Country
Name: ITALY
Region: [Liguria, Lombardia]
Name: University of Genoa
Abbreviation: UNIGE