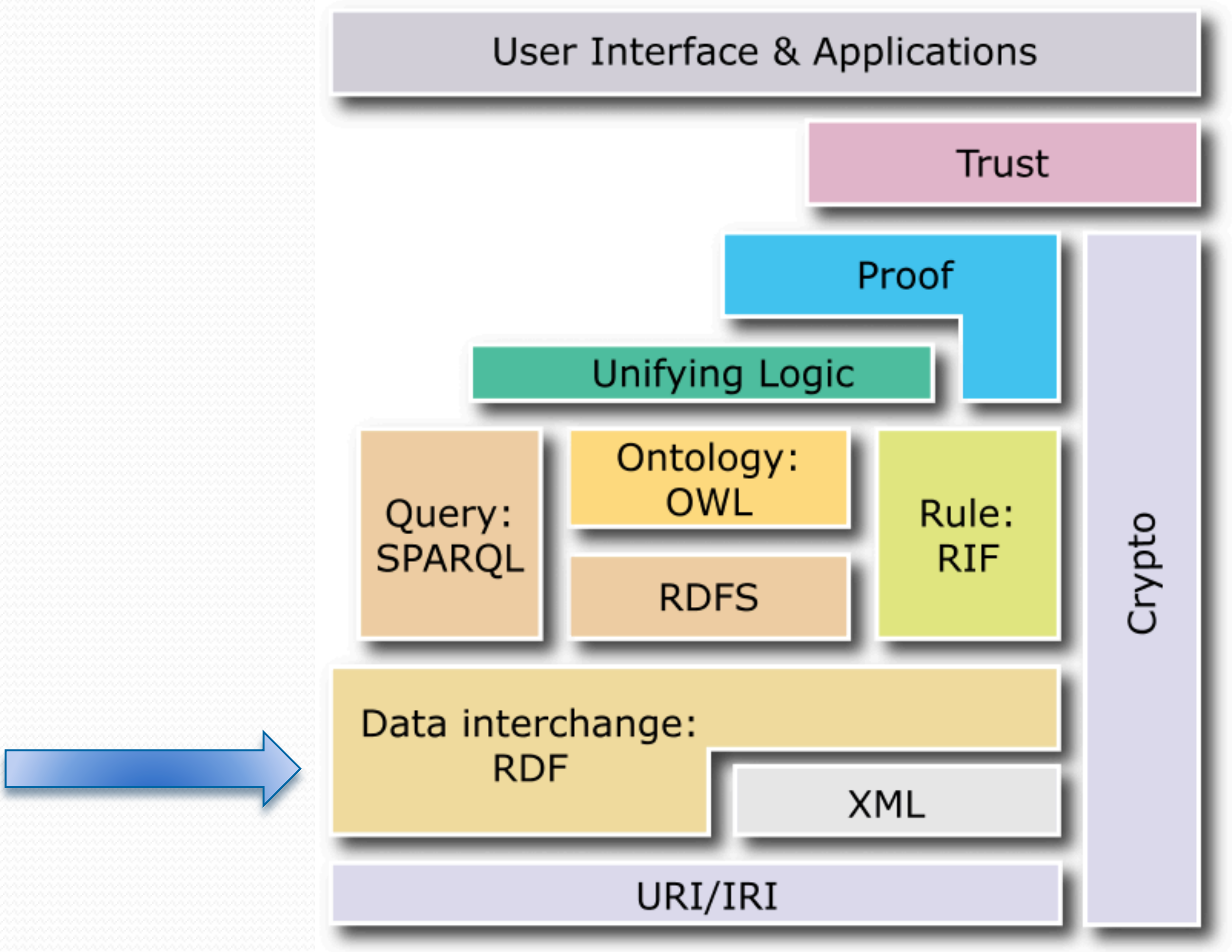
Resource Description Framework (RDF)

What is Resource Description Framework (RDF)?

Semantic Web Layer Cake



What is Resource Description Framework (RDF)?

- RDF
  - Stands for **Resource Description Framework**
  - A standard model for data interchange on the Web.
  - A W3C Recommendation (https://www.w3.org/RDF/)
  - Written in XML
  - Intended to be read and understand by machines
  - Allows for efficient and sophisticated data interchange, searching, cataloging, classification, navigation etc.
  - Based on a well defined set of rules and constraints

RDFResource Description Framework (RDF)?

An example of RDF document:

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:uol="http://www.cs.le.ac.uk/rdf#">
<rdf:Description rdf:about= "http://www.cs.le.ac.uk/rdf#Leicester">
  <uol:population>328939</uol:population>
  <uol:isLocatedIn rdf:resource="http://www.cs.le.ac.uk/rdf#Leicestershire"/>
</rdf:Description>
</rdf:RDF>
```

“So it’s written in XML .....”

“Yes”

“What’s the relationship between XML and RDF?”

RDF documents are written in XML.

The XML language used by RDF is called RDF/XML

RDF: Basics

- RDF documents are written as XML but other syntactic representations of RDF are also **possible**.
- Many other non-XML serialization formats for RDF
  - For example, N3 (Notation3)

```
@prefix uol: <http://www.cs.le.ac.uk/rdf#>.
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<http://www.cs.le.ac.uk/rdf#Leicester>
  uol:population "328939"^^xsd:int ;
  uol:isLocatedIn <http://www.cs.le.ac.uk/rdf#Leicestershire>.
```

*https://www.w3.org/TeamSubmission/n3/

RDF: Basics

- The fundamental components of RDF are:
  - **Resources:** anything defined through ***URIs***
    - e.g http://www.cs.le.ac.uk/rdf#Leicester
  - **Properties**: describe **binary** relations
    - e.g. uol:isLocatedIn ("is located in") relation defined in Locations domain
  - **Statements**: assign a value to a property associated with a specific resource
    - e.g. http://www.cs.le.ac.uk/rdf#Leicester
      - uol:isLocationIn

        http://www.cs.le.ac.uk/rdf#Leicestershire
      - uol:population 328939
      - uol:postcode_prefix "LE"
    - Property value could be identified through a **URI**, it could also be a **string** literal content or a **number**

Resource

- We can think of a resource as an object, a “thing” we want to talk about.
- Resources could be “physical” or “abstract” objects
  - e.g. Book, location, person,
  - e.g. Homepage, student number, activity
- Every resource (physical or abstract) can be identified by a URI (Universal Resource Identifier), this could be:
  - A URL (web address): http://www.cs.le.ac.uk
  - Some other unique identifier :

    e.g. http://www.cs.le.ac.uk/rdf#Leicester

    (references unique location “Leicester”)

URI (Uniform Resource Identifier)

- In real life we use names to refer to resources: “Bob”, “The Moon”, “83 London Road”, “Leicester”, “LE1 7RH”, “Today’s weather”.
- But, names are ambiguous.
- To resolve this problem we use URIs to name things in the Web.
  - e.g. http://www.cs.le.ac.uk/rdf#Leicester

URI to Name Anything

- We can create a URI to refer to anything we want to talk about, including:
  - Network-accessible things, such as an HTML documents.
  - Things that are not network-accessible, such as humans, corporations, and books in a library.
  - Abstract concepts that don’t physically exist, like that of a “unicorn”.

URIs and RDF

- RDF uses *URI references* to define its resources.
- A *URI reference* (or URIref) is a URI, together with an optional fragment identifier at the end.
- The URIreference:
  - http://www.example.org/index.html#section2
  - consists of:
    - the URI http://www.example.org/index.html
    - the fragment identifier: section2.
  - A resource is identifiable by a URI reference

Properties

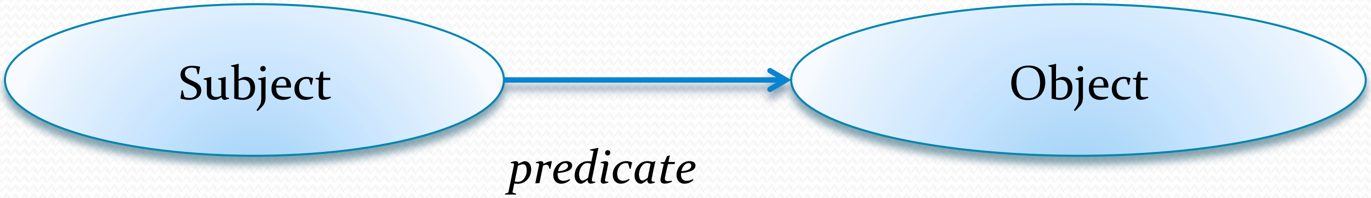
- Properties are special kinds of resources
- properties describe binary relations between resources e.g.
  - *Book* "written by" *Person*,
  - *Book* "has title" *BookTitle*
  - *City* "is located in" *County*
  - *City* "population" *Number*
  - *Person A* "is a friend of" *Person B*
- Properties are also identified by URI
  - http://www.cs.le.ac.uk/rdf#isLocatedIn
  - Or URI abbreviation using prefixes uol:isLocatedIn

Statements

- RDF statements assert the properties of resources
- A RDF statement can be seen as a triple
- Example of statements
  - John is a Professor
  - John has an email address “abc@le.ac.uk”
  - Book “Harry Potter” is written by J. K. Rowling
  - Leicester is located in Leicester
  - Leicester has a population of 328939

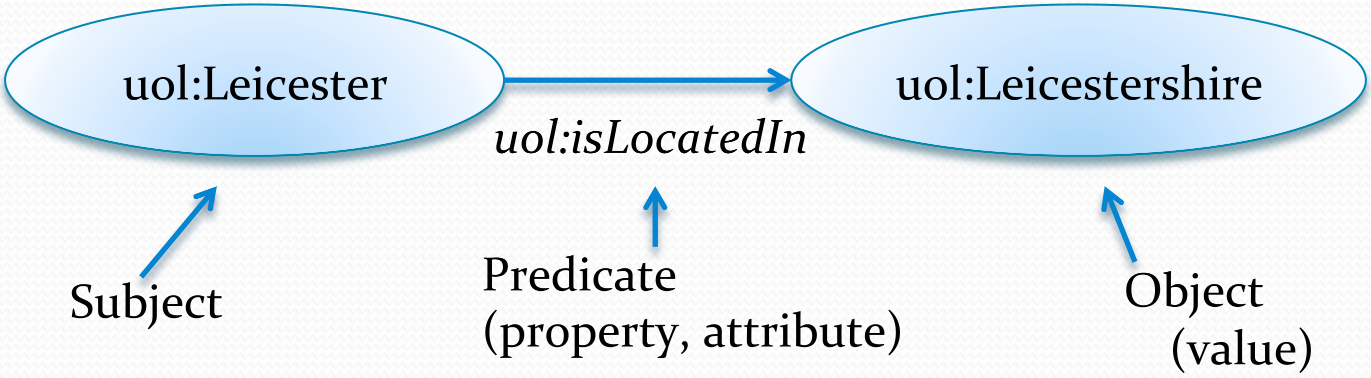
RDF: Basics

- Basic building block in RDF: a **triple**
  - A statement can be seen as a triple
  - **<subject, predicate, object>**
- *"Leicester is located in Leicestershire"* is a statement
  - ***subject:*** *Leicester*
  - ***predicate:*** *is located in*
  - ***object:*** *Leicestershire*



RDF: Basics

Triple <subject, predicate, object>



- The triple **(S,P,O)** can be considered as a logical formula **P(S,O)**
  - Binary predicate **P** relates resource **S** to resource **O**
  - RDF offers **ONLY** binary predicates (properties)
  - Any n-ary relation in RDF has to be converted into a set of binary relations

*For simplicity reasons we use URI abbreviation (prefixed name) in the diagram above
uol:Leicester is equivalent to http://www.cs.le.ac.uk/rdf#Leicester

RDF as a Directed Graph

- Let’s look at this RDF document again

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:uol="http://www.cs.le.ac.uk/rdf#">
<rdf:Description rdf:about=
          "http://www.cs.le.ac.uk/rdf#Leicester">
  <uol:population>328939</uol:population>
  <uol:isLocatedIn rdf:resource=
          "http://www.cs.le.ac.uk/rdf#Leicestershire"/>
</rdf:Description>
</rdf:RDF>
```

RDF as a Directed Graph

- Notation
  - nodes and arcs.
  - directed from ***subject*** to the ***object***.
  - nodes: unirefs, blank nodes, literals.
  - uniref: URI reference, provides a unique identity to the resource. Shown as an ellipse within the graph.
  - blanks: shown as an empty circle.
  - literals: a character string, a data type. Shown as rectangles.

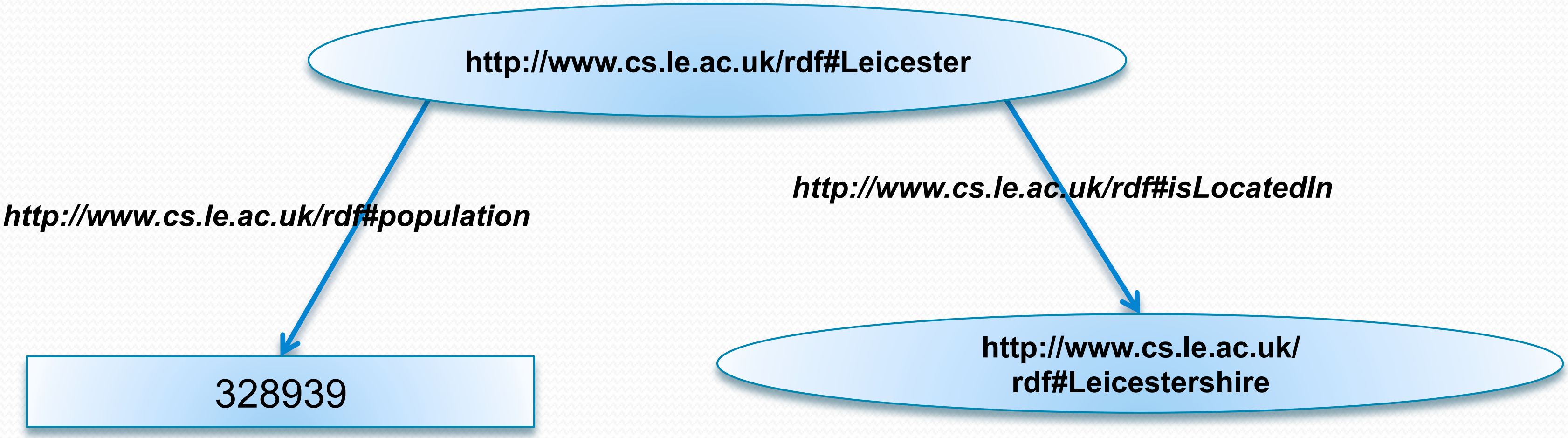
Default, most popular, but not the most efficient way of storing and retrieving data for machine processing.

RDF as a Directed Graph

Each tripe consists a ***subject, a predicate and an object***.
A set of such triples become a **RDF Graph.**

A RDF graph is a directed graph with labeled nodes and arcs

- **from** the resource (the **subject** of the statement)
- **to** the value (the **object** of the statement)


http://www.cs.le.ac.uk/rdf#Leicester
http://www.cs.le.ac.uk/rdf#population
http://www.cs.le.ac.uk/rdf#isLocatedIn
328939
http://www.cs.le.ac.uk/
rdf#Leicestershire

Objects could be **resources** or **literals (strings/numbers)**

Blank Node

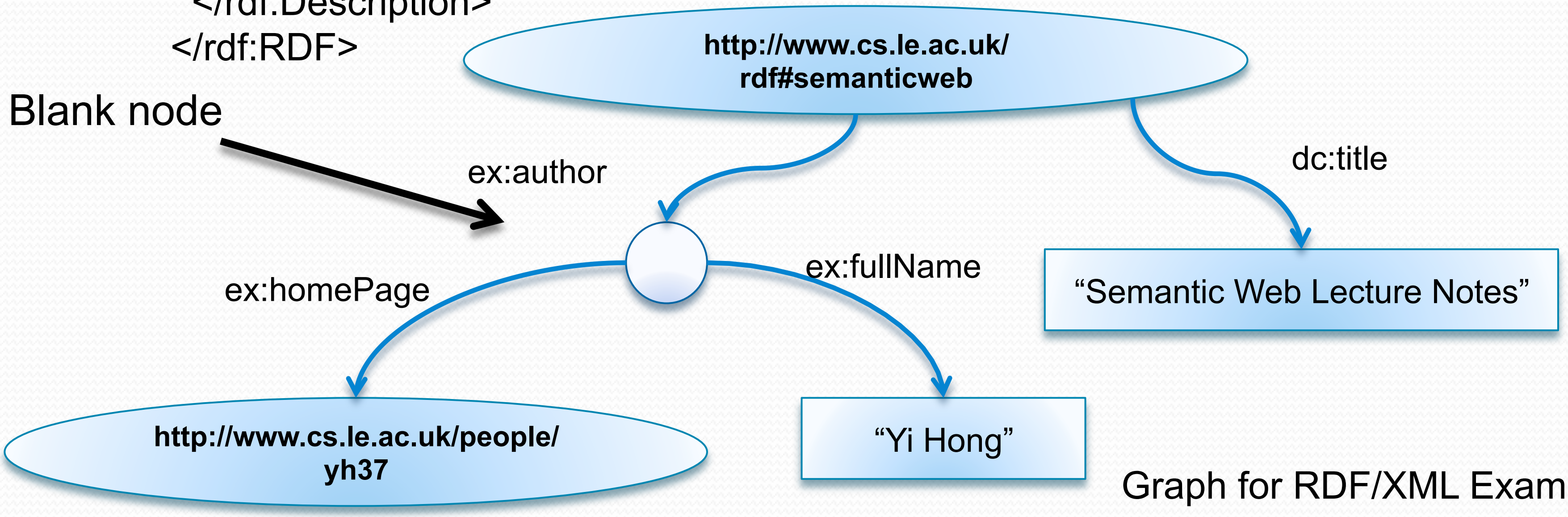
- It is possible to have a node without URI identifier
- Blank nodes in the RDF graph are distinct but have no URI identifier

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
            xmlns:dc="http://purl.org/dc/elements/1.1/"
            xmlns:ex="http://example.org/stuff/1.0/">

  <rdf:Description rdf:about="http://www.cs.le.ac.uk/rdf#semanticweb"
    dc:title="Semantic Web Lecture Notes">
    <ex:author rdf:nodeID="abc"/>
  </rdf:Description>
  <rdf:Description rdf:nodeID="abc" ex:fullName="Yi Hong">
    <ex:homePage rdf:resource="http://www.cs.le.ac.uk/people/yh37"/>
  </rdf:Description>
</rdf:RDF>
```

When blank node needs to be referred to in the RDF/XML multiple times,

Blank Node

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
            xmlns:dc="http://purl.org/dc/elements/1.1/"
            xmlns:ex="http://example.org/stuff/1.0/">

  <rdf:Description rdf:about="http://www.cs.le.ac.uk/rdf#semanticweb"
    dc:title="Semantic Web Lecture Notes">
    <ex:author rdf:nodeID="abc"/>
  </rdf:Description>
  <rdf:Description rdf:nodeID="abc" ex:fullName="Yi Hong">
    <ex:homePage rdf:resource="http://www.cs.le.ac.uk/people/yh37"/>
  </rdf:Description>
</rdf:RDF>
```


http://www.cs.le.ac.uk/rdf#semanticweb
Blank node
ex:author
dc:title
ex:homePage
ex:fullName
"Semantic Web Lecture Notes"
http://www.cs.le.ac.uk/people/yh37
"Yi Hong"

Graph for RDF/XML Example

RDF/XML (1)

- The top-level XML document element of an RDF document rdf:RDF
- The content of this element is a number of descriptions, which use rdf:Description tags.
- Every description makes statements about a resource, usually identified in 2 ways:
  - an about attribute, referencing an existing description
  - an ID attribute, can only be used once within this document
- <rdf:Description rdf:about=”#Leicester”/>

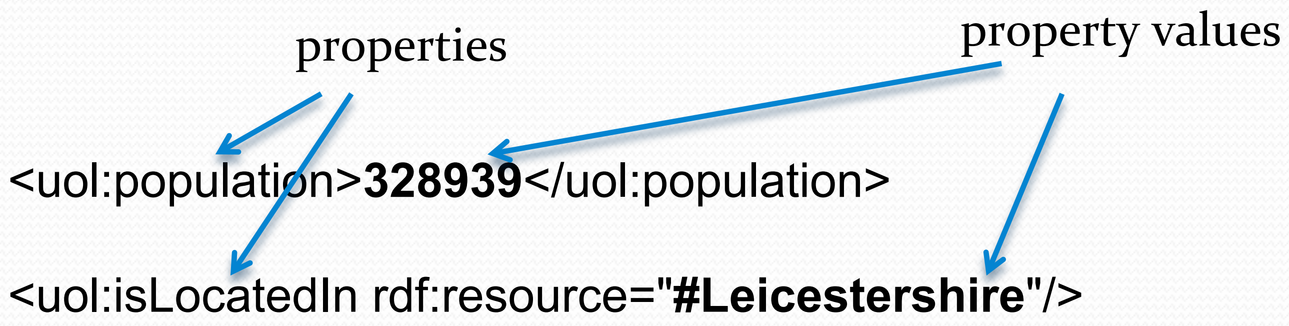
RDF/XML (2)

- Abbreviating URIs: using **xml:base** for shortening URIs

```
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:uol="http://www.cs.le.ac.uk/rdf#"
xml:base="http://www.cs.le.ac.uk/rdf#"
>
<rdf:Description rdf:about="#Leicester">
  <uol:population>328939</uol:population>
  <uol:isLocatedIn rdf:resource="#Leicestershire"/>
</rdf:Description>
</rdf:RDF>
```

RDF/XML (3)

- The **rdf:Description** element makes a statement about the resource Leicester
  - Within rdf:Description element, the property is used as a tag
  - If the property value is a literal (string, number or date): the content is the value of the property.
  - If the property value is a non-literal: its URI should be specified in property rdf:resource .



properties

property values

<uol:population>**328939**</uol:population>

<uol:isLocatedIn rdf:resource="**#Leicestershire**"/>

Concise Representation

```
<!DOCTYPE rdf:RDF [
    <!ENTITY xsd  "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY rdf  "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
    <!ENTITY uol  "http://www.cs.le.ac.uk/rdf#" >
  ]>

<rdf:RDF
  xml:base  = "&uol;"
  xmlns:rdf = "&rdf;"
  xmlns:uol=  "&uol;"
  xmlns:xsd = "&xsd">
>
<rdf:Description rdf:about="#Leicester">
  <uol:population>328939</uol:population>
  <uol:isLocatedIn rdf:resource="#Leicestershire"/>
</rdf:Description>
</rdf:RDF>
```

XML Schema data types

- In RDF, typed literals (primitive data types such as string, integer and data) can be used
  - Usually through the use of XML Schema
- In which case the XML Schema namespace has to be declared in the namespace block

```
<rdf:RDF
xmlns:xsd = "http://www.w3.org/2001/XMLSchema#" .. >
```

To explicitly says `population` must be an integer and `postcode` must be a string :

```
<rdf:Description rdf:about="#Leicester">
  <uol:population rdf:datatype="&xsd;int">328939</uol:population>
  <uol:postcode rdf:datatype="&xsd;string">LE</uol:population>
</rdf:Description>
```

Using `rdf:type`

- Declare a statement which formally defines a description

```
<rdf:Description rdf:about="#Leicester">
  <rdf:type rdf:resource="&uol;City"/>
  <uol:population rdf:datatype=“&xsd;int”>328939</uol:population>
  <uol:postcode rdf:datatype=“&xsd;string”>LE</uol:population>
</rdf:Description>
```

- The above RDF fragment declares resource `Leicester` to be of type `City`.
- `City` is defined in a vocabulary (as a class using RDF Schema, we’ll talk about RDF Schema later ..)

Using a typed node element to replace an `rdf:type`

- RDF/XML allows **rdf:type** to be expressed more concisely
  - replacing the **rdf:Description** node element name with the namespaced-element corresponding to the URI of the value of the type relationship

For example:

```
<rdf:Description rdf:about="#Leicester">
  <rdf:type rdf:resource="&uol;City"/>
  <uol:population rdf:datatype="&xsd;int”>328939</uol:population>
</rdf:Description>
```

It’s equivalent to

```
<uol:City rdf:about="#Leicester">
    <uol:population rdf:datatype="&xsd;int”>328939</uol:population>
</uol:City>
```

A Complete Example: `Map.rdf`

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:uol="http://www.cs.le.ac.uk/rdf#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xml:base="http://www.cs.le.ac.uk/rdf">
  <uol:City rdf:ID="Leicester">
    <uol:isLocationIn>
      <uol:County rdf:ID="Leicestershire"/>
    </uol:isLocationIn>
    <uol:population rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >328939</uol:population>
    <uol:postcode rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >LE</uol:postcode>
  </uol:City>
</rdf:RDF>
```

*Created using Protégé 3.4.8

XML vs RDF

XML

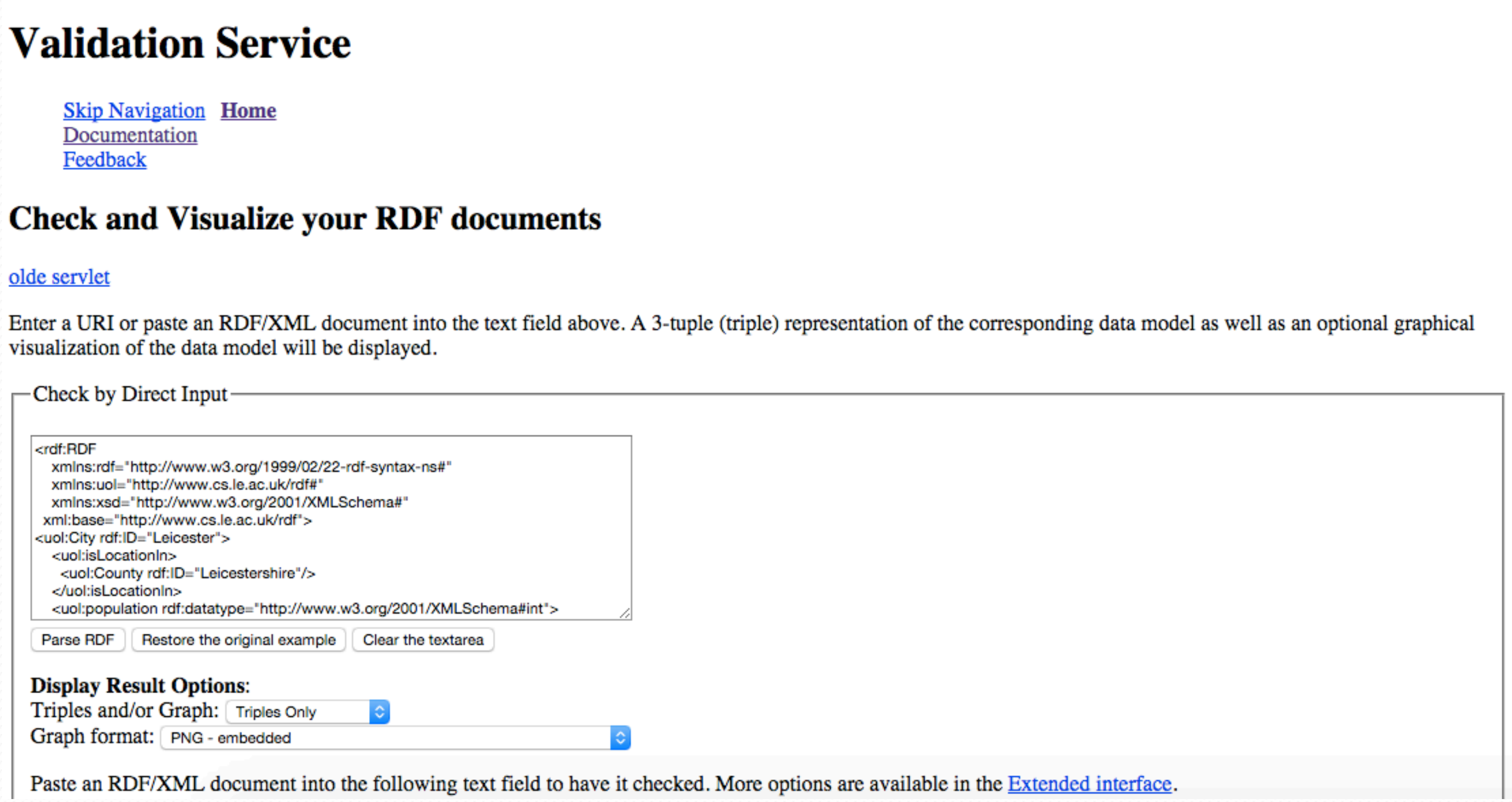
```
<?xml version="1.0"?>
<City name="Leicester">
    <county>Leicestershire</county>
    <population>328939<population>
    <postcode>LE</postcode>
  </City>
```

RDF

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:uol="http://www.cs.le.ac.uk/rdf#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xml:base="http://www.cs.le.ac.uk/rdf">
<uol:City rdf:ID="Leicester">
    <uol:isLocationIn>
      <uol:County rdf:ID="Leicestershire"/>
    </uol:isLocationIn>
    <uol:population rdf:datatype="http://www.w3.org/2001/XMLSchema#int">
     328939</uol:population>
    <uol:postcode rdf:datatype="http://www.w3.org/2001/XMLSchema#string" >
     LE</uol:postcode>
  </uol:City>
</rdf:RDF>
```

Online: W3C Validation Service

- https://www.w3.org/RDF/Validator/



Validation Service

Check and Visualize your RDF documents

olde servlet

Enter a URI or paste an RDF/XML document into the text field above. A 3-tuple (triple) representation of the corresponding data model as well as an optional graphical visualization of the data model will be displayed.

Check by Direct Input

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:uol="http://www.cs.le.ac.uk/rdf#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xml:base="http://www.cs.le.ac.uk/rdf">
<uol:City rdf:ID="Leicester">
   <uol:isLocationIn>
     <uol:County rdf:ID="Leicestershire"/>
   </uol:isLocationIn>
   <uol:population rdf:datatype="http://www.w3.org/2001/XMLSchema#int">
```

Display Result Options:
Triples and/or Graph: Triples Only
Graph format: PNG - embedded

Paste an RDF/XML document into the following text field to have it checked. More options are available in the Extended interface.

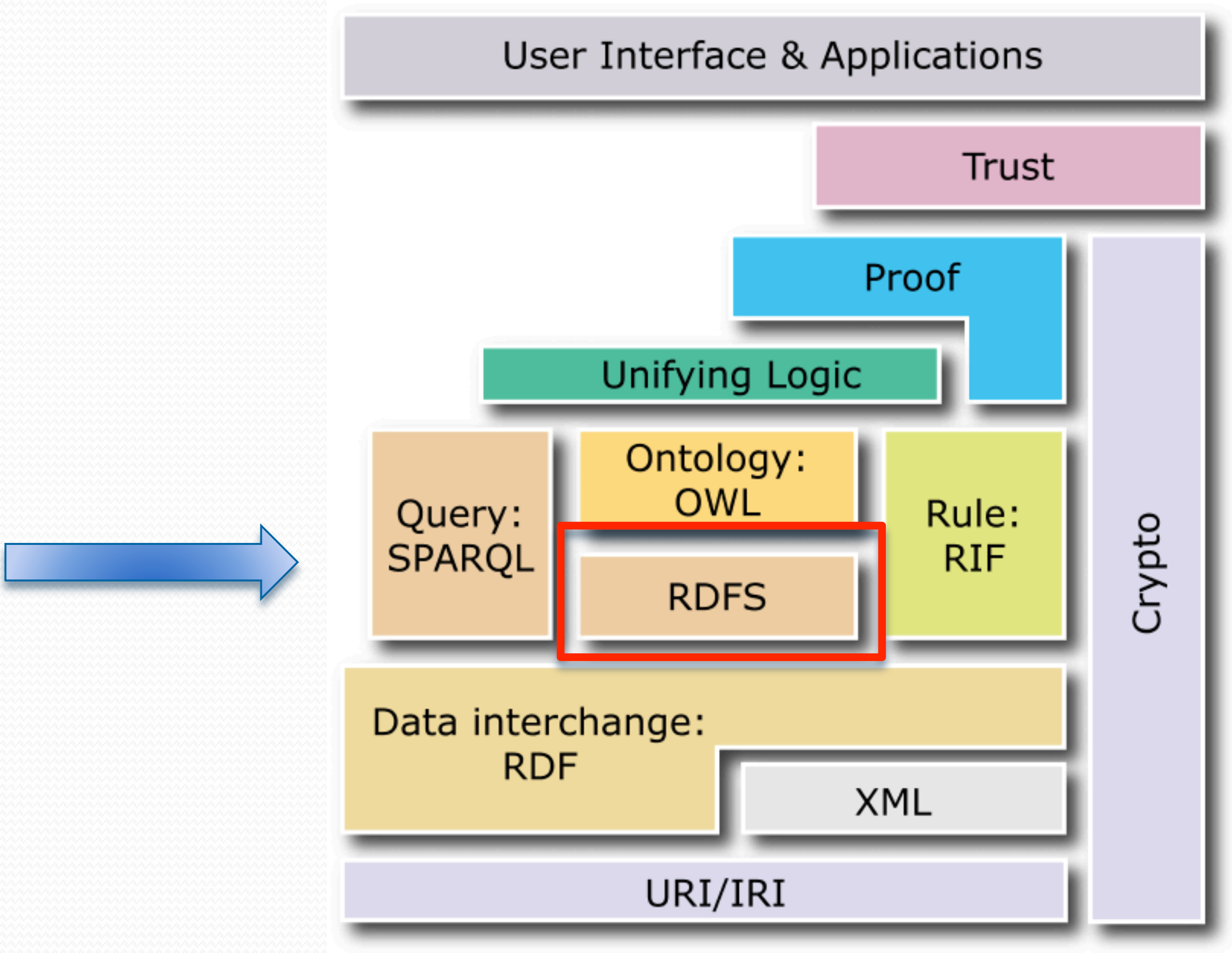
Essential Reading

- Semantic Web Primer, Chapter 3
- Semantic Web for the Working Ontologist, Chapter 3
- For everything you ever wanted to know about RDF,
- http://www.w3.org/RDF/
- Dave Beckett's Resource Description Framework (RDF) Resource Guide http://planetrdf.com/guide/
- The Semantic Web: roles of XML and RDF, Stefan Decker,
- http://www.ontoknowledge.org/oil/downl/IEEE00.pdf RDF Primer, E. Miller,
- http://www.w3.org/TR/rdf-primer/

Introduction to RDF Schema (RDF-S)

What is Resource Description Framework (RDF)?

Semantic Web Layer Cake



RDF Schema (1)

- RDF makes no assumptions about any specific domain, nor does it define the semantics of such a domain
- To define the ***terminology*** related to any domain a schema language such as RDF Schema (RDF-S) can be used
- RDF Schema (RDF-S)
  - provides a data-modelling vocabulary for RDF data.
  - defines the classes and properties used in RDF and what kinds of relation can be applied to which resources

RDF Schema (2)

- Relation between RDF and RDF Schema is not the same as that between XML and XML Schema
  - XML schema constraints the structure of XML documents
  - RDFS defines the vocabulary used within an RDF data model
- through RDFS its also possible to
  - specify what properties can be applied to a resource
  - specify the values that such properties can take
  - declare relationships between objects

RDF Schema (3)

- The user can describe any particular domain using:
  - **Classes** and **Properties**: differentiate between individual objects (or instances) and classes of object
    - John is a Person, Person is a class
    - Leicester is a City, City is a class
  - Restrict properties to apply to certain things
    - e.g. property population can only be applied to regions (i.e. Cities); Only student can have studentID property; isFriendOf can only be applied to a Person
- Class Hierarchies and Inheritance
  - Student is a subclass of Person; Undergraduate student is a subclass of Student.
- Property Hierarchies
  - Property is “hasBrother” is a sub property of “hasSibling”

Core Class

- **rdfs:Resource**, the class of all resources
- **rdfs:Class**, the class of all classes
- **rdfs:Literal**, the class of all literals (string)
- **rdf:Property**, the class of all property

“Meta-classes”

```
 <rdf:RDF
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#".. >

<rdfs:Class rdf:ID=”City">
    <rdfs:comment> a large human settlement</rdfs:comment>
     …
 </rdfs:Class>

<rdf:Prtoperty rdf:ID=”population">
    <rdfs:comment>all the inhabitants of a particular place
    </rdfs:comment>
    …
 </rdf:Property>
```

http://www.w3.org/2000/01/rdf-schema#

Core Properties

- **rdf:type**, relates a resource to its class
- **rdfs:subClassOf**, relates a class to one of its superclasses
- **rdfs:subPropertyOf**, relates a property to one of its superproperties

```
<rdfs:Class rdf:ID=”Student”>

<rdfs:Class rdf:ID=”UndergraduateStudent">
   <rdfs:subClassOf rdf:resource=”#Student"/>
 </rdfs:Class>
```

```
<rdf:Property rdf:ID=”knows”>
<rdf:Property rdf:ID=”isFriendOf">
   <rdfs:subPropertyOf rdf:resource=”#knows"/>
 </rdf:Property>
```

Core Properties

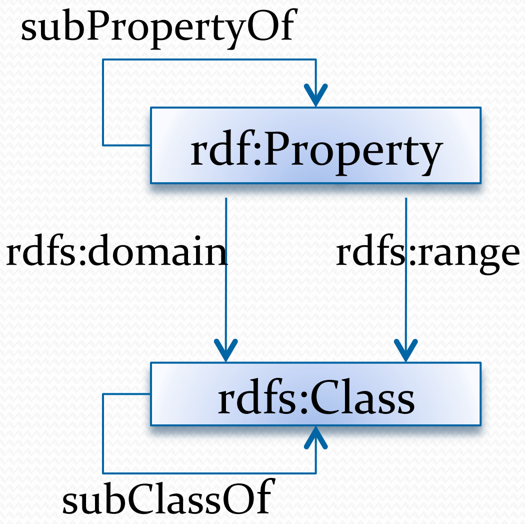
- **rdf:type**, relates a resource to its class
- **rdfs:subClassOf**, relates a class to one of its superclasses
- **rdfs:subPropertyOf,** relates a property to one of its superproperties

Multiple inheritance: when inheriting from two or more classes.

```
<rdfs:Class rdf:ID=”Policelady">
  <rdfs:subClassOf>
    <rdfs:Class rdf:ID=”Police"/>
    <rdfs:Class rdf:ID=”Lady"/>
  </rdfs:subClassOf>
 </rdfs:Class>
```

Core Properties

- **rdfs:domain**, specifies the domain of a property P
  - The class of those resources that may appear as subjects in a triple with predicate P
  - If the domain is not specified, then any resource can be the subject
- **rdfs:range**, specifies the range of a property P
  - The class of those resources that may appear as values (of objects) in a triple with predicate P



RDF : Triples

----

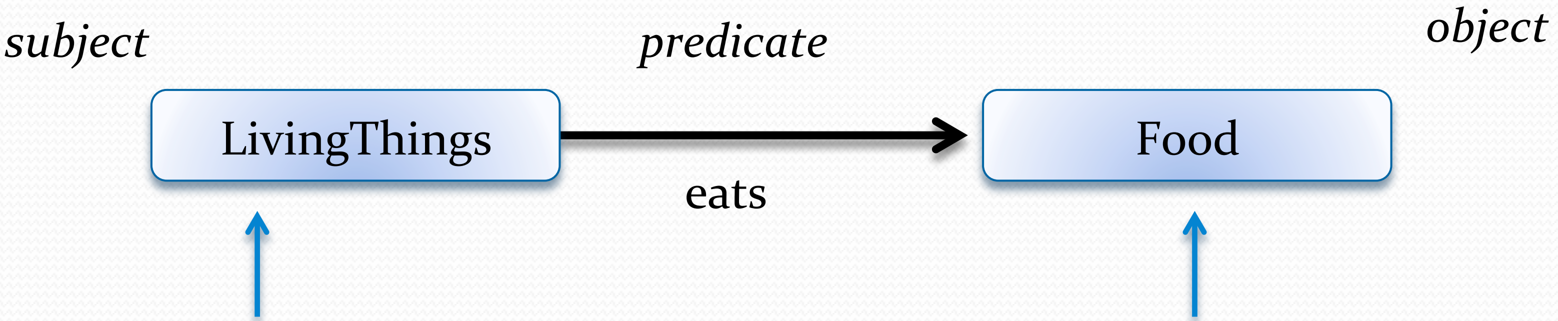
property ->predicate

domain -> subject

range -> object

Core Properties

- **rdfs:domain**, specifies the domain of a property P
  - The class of those resources that may appear as subjects in a triple with predicate *P*
  - If the domain is not specified, then any resource can be the subject
- **rdfs:range**, specifies the range of a property P
  - The class of those resources that may appear as values (of objects) in a triple with predicate *P*



subject

predicate

object

LivingThings

eats

Food

A **domain** of a property limits the individuals to which the property can be applied.

The **range** of a property limits the individuals that the property may have as its value

Core Properties

- **rdfs:domain**, specifies the domain of a property P
  - The class of those resources that may appear as subjects in a triple with predicate *P*
  - If the domain is not specified, then any resource can be the subject
- **rdfs:range**, specifies the range of a property P
  - The class of those resources that may appear as values (of objects) in a triple with predicate *P*

```
<rdf:Property rdf:ID="eat">
   <rdfs:domain rdf:resource=”#LivingThings"/>
   <rdfs:range rdf:resource="#Food"/>
 </rdf:Property>
```

Core Properties

Domains and **Ranges :** Examples

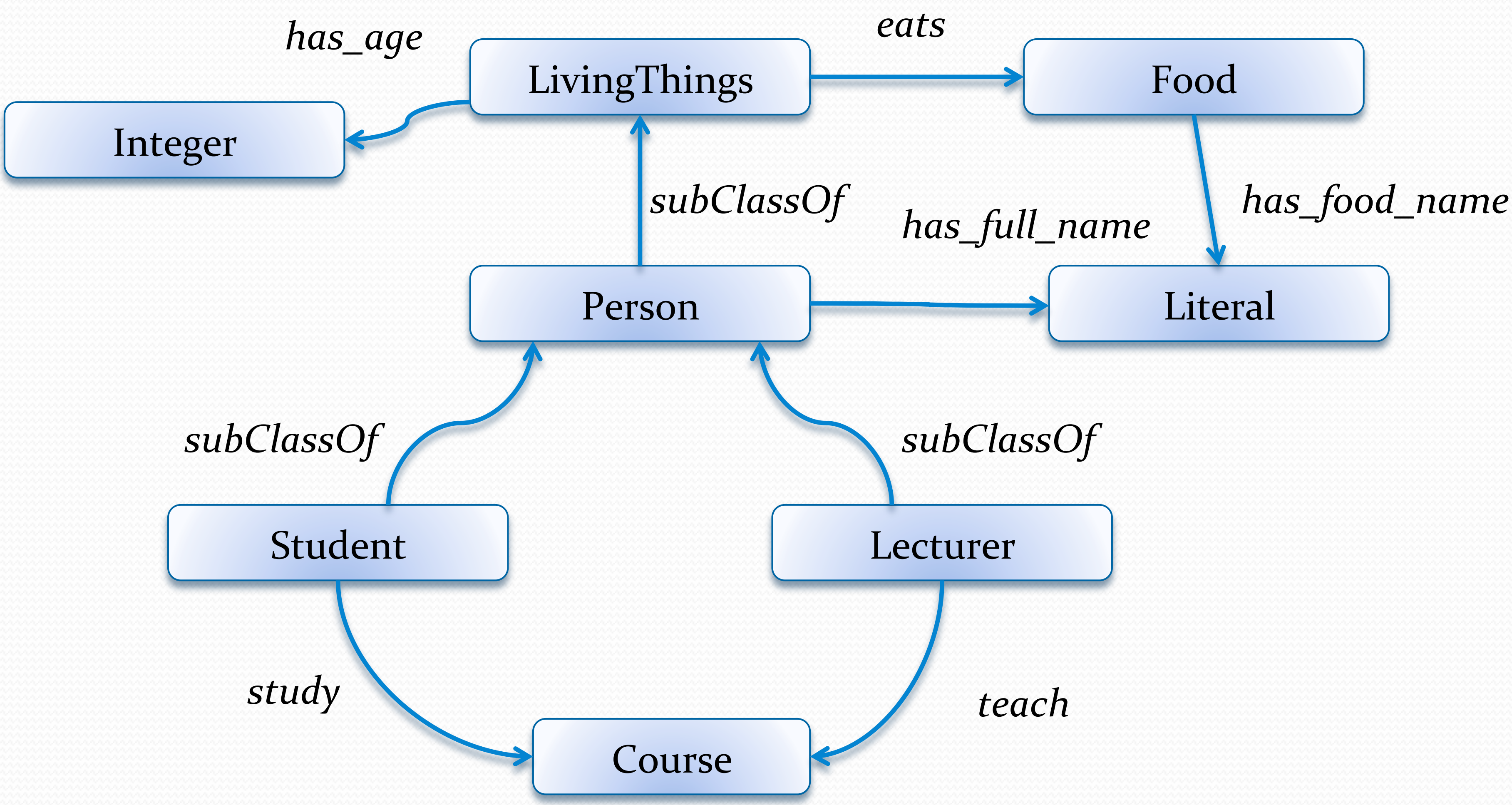
Property range could be any built-in data type or RDF-S class

```
<rdf:Property rdf:ID="population">
   <rdfs:domain rdf:resource="#Place"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</rdf:Property>
```

Hierarchies of classes support inheritance of a property domain and range: Property **eat** can also be applied to class ***Person*** because it’s a subclass of *LivingThings*

```
<rdf:Property rdf:ID="eat">
   <rdfs:domain rdf:resource="#LivingThings"/>
   <rdfs:range rdf:resource="#Food"/>
 </rdf:Property>
 <rdfs:Class rdf:ID=”Person">
    <rdfs:subClassOf rdf:resource=”#LivingThings"/>
  </rdfs:Class>
```

Example (1) RDF-S



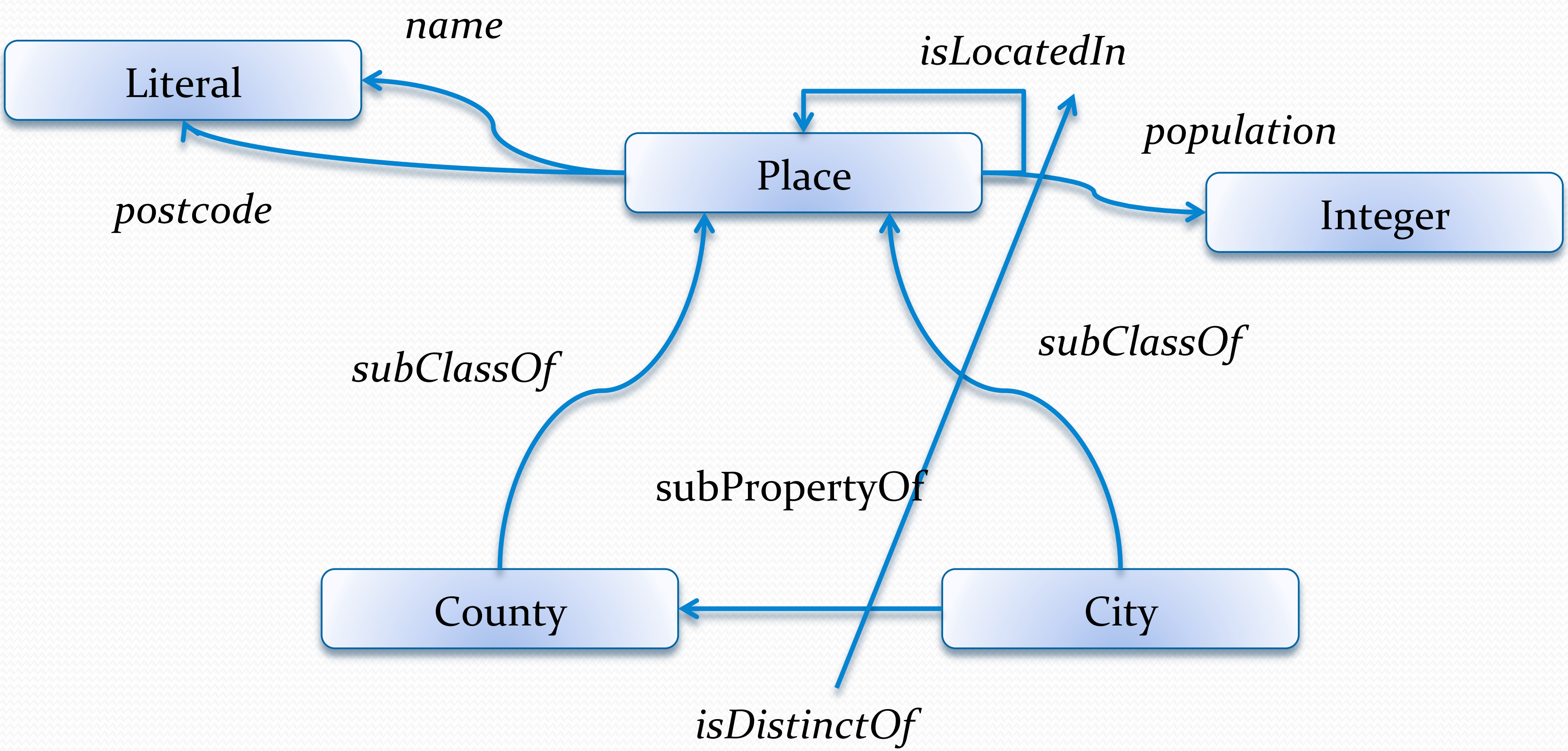
RDF-S: Complete Example (2)

```
<rdfs:Class rdf:ID="Student">
  <rdfs:subClassOf>
    <rdfs:Class rdf:ID="Person"/>
  </rdfs:subClassOf>
 </rdfs:Class>
 <rdfs:Class rdf:about="#Person">
  <rdfs:subClassOf>
    <rdfs:Class rdf:ID="LivingThing"/>
  </rdfs:subClassOf>
 </rdfs:Class>
 <rdfs:Class rdf:ID="Food"/>
 <rdfs:Class rdf:ID="Course"/>
 <rdfs:Class rdf:ID="Lecturer">
  <rdfs:subClassOf rdf:resource="#Person"/>
 </rdfs:Class>
 <rdf:Property rdf:ID="has_age">
  <rdfs:domain rdf:resource="#LivingThing"/>
  <rdfs:range rdf:resource=”&xsd;int"/>
 </rdf:Property>
 <rdf:Property rdf:ID="teach">
  <rdfs:domain rdf:resource="#Lecturer"/>
  <rdfs:range rdf:resource="#Course"/>
 </rdf:Property>
 <rdf:Property rdf:ID="eat">
  <rdfs:domain rdf:resource="#LivingThing"/>
  <rdfs:range rdf:resource="#Food"/>
 </rdf:Property>
```

```
<rdf:Property rdf:ID="study">
  <rdfs:range rdf:resource="#Course"/>
  <rdfs:domain rdf:resource="#Student"/>
 </rdf:Property>
 <rdf:Property rdf:ID="has_food_name">
  <rdfs:domain rdf:resource="#Food"/>
  <rdfs:range rdf:resource="&xsd;string"/>
 </rdf:Property>
 <rdf:Property rdf:ID="has_full_name">
  <rdfs:domain rdf:resource="#Person"/>
  <rdfs:range rdf:resource="&xsd;string"/>
 </rdf:Property>
```

*complete source code can be found on the Blackboard

Example (2) RDF-S



RDF-S: Complete Example (2)

```
<rdfs:Class rdf:ID="City">
  <rdfs:subClassOf>
    <rdfs:Class rdf:ID="Place"/>
  </rdfs:subClassOf>
</rdfs:Class>
<rdfs:Class rdf:ID="County">
  <rdfs:subClassOf rdf:resource="#Place"/>
</rdfs:Class>
<rdf:Property rdf:ID="name">
  <rdfs:range rdf:resource="&xsd;string"/>
  <rdfs:domain rdf:resource="#Place"/>
</rdf:Property>
<rdf:Property rdf:ID="isDistrictOf">
  <rdfs:domain rdf:resource="#City"/>
  <rdfs:subPropertyOf>
    <rdf:Property rdf:ID="isLocationIn"/>
  </rdfs:subPropertyOf>
  <rdfs:range rdf:resource="#County"/>
</rdf:Property>
<rdf:Property rdf:ID="postcode">
  <rdfs:range rdf:resource=”&xsd;string"/>
  <rdfs:domain rdf:resource="#Place"/>
</rdf:Property>
<rdf:Property rdf:ID="population">
  <rdfs:domain rdf:resource="#Place"/>
  <rdfs:range rdf:resource="&xsd;int"/>
</rdf:Property>
```

```
<rdf:Property rdf:about="#isLocationIn">
  <rdfs:domain rdf:resource="#Place"/>
  <rdfs:range rdf:resource="#Place"/>
 </rdf:Property>
```

*complete source code can be found on the Blackboard

Instance Example

```
<uol:City rdf:ID="Leicester">
  <uol:isLocationIn>
    <uol:County rdf:ID="Leicestershire"/>
  </uol:isLocationIn>
  <uol:population rdf:datatype=“&xsd;int"
  >328939</uol:population>
  <uol:postcode rdf:datatype="&xsd;string"
  >LE</uol:postcode>
  <uol:isDistrictOf rdf:resource="#Leicestershire"/>
 </uol:City>
```

*complete source code can be found on the Blackboard

Container Classes

- **rdf:Bag** (Bag)
  - the container is intended to be unordered
- **rdf:Seq** (Sequence)
  - numerical ordering is intended to be significant
- **rdf:Alt** (Alternative)
  - select one of the members of the container (first member in the container is the default value)

Reification Vocabulary

- **rdf:Statement** : A RDF triple statement
  - **rdf:subject**: subject of the statement
  - **rdf:predicate**: predicate of the statement
  - **rdf:object**: object of the statement

- What is Reification?

```
<rdf:Description rdf:about="#Leicester">
   <uol:population rdf:datatype=“&xsd;int”>328939</uol:population>
</rdf:Description>

<rdf:Statement rdf:ID="Statement_1">
   <rdf:subject rdf:resource="&uol;Leicester"/>
   <rdf:predicate rdf:resource="&uol;population"/>
   <rdf:object rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >LE</rdf:object>
 </rdf:Statement>
```

Applications of RDF/S: FOAF

- FOAF (Friend of a Friend) Ontology
  - a way of providing affiliation and other social information about yourself
  - describing a network of friends and others we know, in such a way that automated processes such as web bots can find this information and incorporate it with other FOAF files
- RDF Schema for FOAF
  - http://xmlns.com/foaf/spec/index.rdf

Example: Tim Berners-Lee's FOAF file

```
<con:Male rdf:about="https://www.w3.org/People/Berners-Lee/card#i">
      <rdf:type rdf:resource="http://xmlns.com/foaf/0.1/Person"/>
      <web:uses rdf:resource="#findMyLoc"/>
      <s:label>Tim Berners-Lee</s:label>
       <con:preferredURI>https://www.w3.org/People/Berners-Lee/card#i</con:preferredURI>
      <space:preferencesFile rdf:resource="https://timbl.com/timbl/Data/preferences.n3"/>
      <space:storage rdf:resource="https://timbl.rww.io/"/>
      <account rdf:resource="http://en.wikipedia.org/wiki/User:Timbl"/>
      <account rdf:resource="http://twitter.com/timberners_lee"/>
      <account rdf:resource="http://www.reddit.com/user/timbl/"/>
      <based_near rdf:parseType="Resource">
         <geo:lat>42.361860</geo:lat>
         <geo:long>-71.091840</geo:long>
      </based_near>
      <family_name>Berners-Lee</family_name>
      <givenname>Timothy</givenname>
      <homepage rdf:resource="https://www.w3.org/People/Berners-Lee/"/>
      <img rdf:resource="http://www.w3.org/Press/Stock/Berners-Lee/2001-europaeum-eighth.jpg"/>
      <mbox rdf:resource="mailto:timbl@w3.org"/>
      <mbox_sha1sum>965c47c5a70db7407210cef6e4e6f5374a525c5c</mbox_sha1sum>
      <name>Timothy Berners-Lee</name>
      <nick>TimBL</nick>
      <nick>timbl</nick>
      <openid rdf:resource="https://www.w3.org/People/Berners-Lee/"/>
      <phone rdf:resource="tel:+1-(617)-253-5702"/>
      <title>Sir</title>
      <workplaceHomepage rdf:resource="http://www.w3.org/"/>
  </con:Male>
```

http://xmlns.com/foaf/spec/

RDF Tool

- RDF Store: stores RDF models in relational dbs (persistent storage)
  - JenaTDB can be hooked with MySQL, PostgreSQL or Oracle db
  - Sesame
  - Virtuoso Universal Server
  - 5store
- RDF validator:
  - https://www.w3.org/RDF/Validator/
- RDF Editors:
  - Protége´
    - For editing RDF/S please use version 3.4.x (Latest Protégé 5 was intended for OWL2): http://protege.stanford.edu/download/protege/3.4/installanywhere/Web_Installers/
- RDF visualiser
  - RDF Viz (visualises graphs)
- Search for schemas:
  - SWOOGLE: http://swoogle.umbc.edu/

Further Information

- Semantic Web Primer, Chapter 3
- RDF Primer, E. Miller
    - http://www.w3.org/TR/rdf-primer/
- RDF tutorial examples
    - http://www.zvon.org/xxl/RDFTutorial/General/contents.html
- The Semantic Web: roles of XML and RDF, Stefan Decker,
    - http://www.cs.ox.ac.uk/ian.horrocks/Publications/download/2000/Horrocks00n.pdf
- Jena RDF tutorial
    - https://jena.apache.org/tutorials/rdf_api.html