# Introduction to RDF

CS 479/CS 579
Fall 2006

## Introduction

- XML:
  - metalanguage for defining markup
  - provides a **syntax** for describing data
  - no **semantics!**
- E.g., "Enrico teachers Semantic Web"

```
<course name="semantic web">
   <teacher> Enrico </teacher>
</course

<teacher name="Enrico">
   <teaches>Semantic Web</teaches>
</teacher>

<cs_classes>
   <class>
     <name>Semantic Web</name>
     <instructor>Enrico </instructor>
   </class>
</cs_classes>
```

## Introduction

- RDF: a data model to describe semantics
- RDF=Resource Description Framework
  - It is a model for data description
  - It provides a syntax for exchange
- Thus, language for encoding metadata about resources that can be identified on the Web

## Introduction

- RDF Properties
  - Framework for describing resources on the web (e.g., properties of web pages, content of pictures, content to search engines)
  - Designed to be machine-readable
  - It is given an XML syntax
  - RDF is a W3C recommendation

## Introduction

- RDF offers structure for description
- RDFS (RDF Schema) offers the vocabulary of the domain
  - terms
  - what properties apply to which objects
  - relationships
- E.g.,

  teacher is a *subclass* of staff

## Introduction

- RDFS should allow semantic meaningful conclusions

```
<staff>Son Tran</staff>
<professor>Joe Pfeiffer</professor>
<course name="Semantic Web">
  <teacher>Enrico Pontelli</teacher>
</course>
```

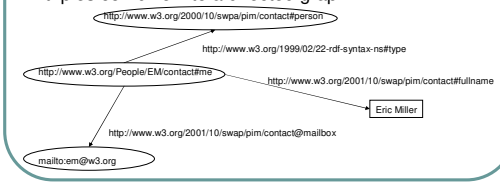
- a query "Who is staff?" should return
  - Son Tran (can be obtained using //staff )
  - Joe Pfeiffer (professor is a subset of staff)
  - Enrico Pontelli (courses are taught by staff members

## Introduction

- In the mid-nineties, R. V. Guha at Apple developed a metadata format named Meta Content Framework (MFC).
- They built a cool application called Hotsauce driven by MFC.
- R. V. Guha and T. Bray met at Nescape and combined MFC and XML.
- Microsoft pushed the Channel Definition Format (CDF) based on XML.
- Netscape submitted “MFC in XML” to the W3C.
- Microsoft was upset, W3C got anxious.
- W3C decided for a general-purpose metadata format: RDF.
- RDF is neither MFC nor CDF.

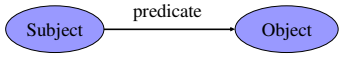
## Basic Idea of RDF

- Descriptions based on triples
  - individuals/objects/entities… (on the Web)
  - properties
  - values of properties
- triples combine into a directed graph




## Basic Idea of RDF

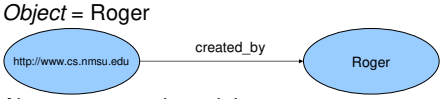
- Basic Model:




- Subject = *Resource*
  - the “thing” we are talking about
  - identified by a URI
- Predicate = *Property*
  - special type of resource
  - describe relations between resources
- Object = *Property Value*
  - itself a resource or a literal value

## Basic Idea of RDF

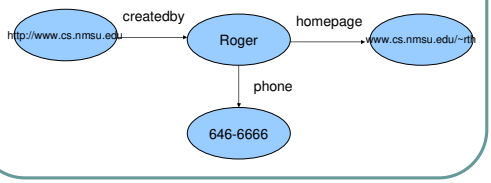
- E.g.
  *The author of http://www.cs.nmsu.edu is Roger*
- *Subject* = http://www.cs.nmsu.edu
- *Property* = created_by
- *Object* = Roger




- Also represented as triple:
  (http://www.cs.nmsu.edu, created_by, Roger)

## Basic Idea of RDF

- each triple is a *Statement*
- Note description as directed graph
- Property values can be resources – statements can be "linked"




## Basic Idea of RDF

- Needs:
  - machine processable identifiers for subjects, objects, and predicates
    - URIs or, more precisely, URI References (URI + fragment identifier)
  - machine processable language for encoding statements
    - XML

## Basic Idea of RDF

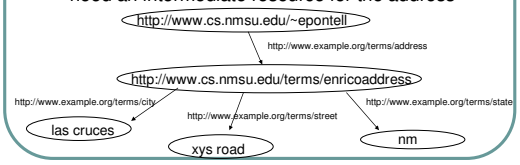
- "www.cs.nmsu.edu created by Roger"
  - subject: http://www.cs.nmsu.edu
  - predicate: http://purl.org/dc/elements/1.1/creator
  - object: http://www.cs.nmsu.edu/~rth
- Objects can be also constant values (*literals*)
- Statements written as triples
  <http://…> <http://…> <http://…>
  - to make it short: use XML qualified names (Qnames)
    - prefix dc: namespace http://purl.org/dc/elements/1.1
    - prefix nmsu: namespace http://www.cs.nmsu.edu
    - becomes: nmsu:index.html dc:creator nmsu:~rth
- URIs might point to URL containing further information about the definitions (but not required by RDF)
- use of URIs allows us
  - to avoid ambiguities
  - to share of vocabularies
    - two people using dc:creator they are referring to the same property
  - RDF statements about predicates

## Basic Idea of RDF

- Note:
  - RDF only defines graph syntax, triples, an meanings of a few concepts
  - everything else is user defined externally from RDF
    - for RDF the following have the same meaning
      nmsu:index.html dc:creator nmsu:~rth
      fy:abeerc ed:asdfwer fadfadf:33313s
    - RDF understands statements as much as a DBMS understands its data
      - sharing of vocabularies essential

## Basic Idea of RDF

- Some additional thoughts
  - often the value of a property is a complex entity – a date, an address, etc.
  - "Enrico's address is xys road, las cruces, nm
  - need an intermediate resource for the address



## Basic Idea of RDF

- RDF allows blank nodes
  - in triples: _:name as blank node identifier
    - nmsu:~epontell terms:address _:enricoaddress
    - _:enricoaddress terms:city “las cruces”
    - _:enricoaddress terms:street “xyz road”
    - …
  - allows only as subjects or objects
- Frequently needed to encode n-ary relations
  - RDF supports only binary predicates
- Frequently needed to represent entities with no real URIs
  - e.g., enrico represented by http://www.cs.nmsu.edu/~epontell
  - prevents us from talking about enrico’s web page

## Basic Idea of RDF

- Properties are only binary
- Predicates of greater arity have to be partitioned in multiple binary ones
- referee(italy, france, Elizondo):
  - ref(game, Elizondo)
  - team1(game, italy)
  - team2(game, france)

## Basic Idea of RDF

- Literals (e.g., “27”) meant to be interpreted as decimal number
- RDF allows typed literals
  - string + URI identifying the type
    - <http://www.cs.nmsu.edu/~epontell> <http://www.example.org/terms/age> "27"^^<http://www.w3.org/2001/XMLSchema#integer>
    - nmsu:index.html exterms:creation-date "1999-08-16"^^xsd:date
- Note
  - RDF has no explicit data types
  - Externally defined types (in the example, use XML Schema)
    - Note: only some XML Schema datatypes are accepted

## RDF/XML

- XML Syntax

```
<?xml version="1.0" encoding="UTF-16"?>
<RDF>
   <Description about="http://www.cs.nmsu.edu">
     <createdby>Roger</createdby>
   </Description>
</RDF>
```

- More complete (using namespaces)

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:mydomain="http://www.cs.nmsu.edu/~epontell">
<rdf:Description rdf:about="http://www.cs.nmsu.edu">
  <mydomain:createdby>Roger</mydomain:createdby>
</rdf:Description>
</rdf:RDF>
```

## Basic Idea of RDF

- W3C provides RDF validator
  http://www.w3.org/RDF/Validator/

## RDF XML Syntax

- Description of some CDs

| Title | Artist | Country | Company | Price | Year |
|---|---|---|---|---|---|
| Dark Side of the moon | Pink Floyd | UK | Sony | $12.99 | 1973 |
| Abacab | Genesis | USA | Atlantic | $9.99 | 1981 |
| … | | | | | |

## RDF XML Syntax

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-
   ns#" xmlns:cd="http://www.recshop.fake/cd#">
<rdf:Description
   rdf:about="http://www.recshop.fake/cd/DarkSideoftheMoon">
   <cd:artist>Pink Floyd</cd:artist>
   <cd:country>UK</cd:country>
   <cd:company>Sony</cd:company>
   <cd:price>12.99</cd:price>
   <cd:year>1973</cd:year>
</rdf:Description>
<rdf:Description rdf:about="http://www.recshop.fake/cd/Abacab">
   <cd:artist>Genesis</cd:artist>
   <cd:country>USA</cd:country>
   <cd:company>Atlantic Records</cd:company>
   <cd:price>9.99</cd:price>
   <cd:year>1981</cd:year>
</rdf:Description>
   . . .
</rdf:RDF>
```

## RDF XML Syntax

- Basics
  - The first line of the RDF document is the XML declaration. The XML declaration is followed by the root element of RDF documents: **<rdf:RDF>**.
  - The **xmlns:rdf** namespace, specifies that elements with the rdf prefix are from the namespace "http://www.w3.org/1999/02/22-rdf-syntax-ns#".
  - The **xmlns:cd** namespace, specifies that elements with the cd prefix are from the namespace "http://www.recshop.fake/cd#".
  - The **<rdf:Description>** element contains the description of the resource identified by the **rdf:about** attribute.
  - The elements: **<cd:artist>, <cd:country>, <cd:company>,** etc. are properties of the resource.

## RDF XML Syntax

- rdf:RDF element
  - root element of an RDF document.
  - It defines the XML document to be an RDF document.
  - It also contains a reference to the RDF namespace

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
. . Description goes here .
</rdf:RDF>
```

## RDF XML Syntax

- **The <rdf:Description> Element**
  - The <rdf:Description> element identifies a resource with the about attribute.
  - The <rdf:Description> element contains elements that describe the resource:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:cd="http://www.recshop.fake/cd#">
<rdf:Description rdf:about="http://www.recshop.fake/cd/Abacab">
         <cd:artist>Genesis</cd:artist>
         <cd:country>USA</cd:country>
         <cd:company>Atlantic</cd:company>
         <cd:price>9.99</cd:price>
         <cd:year>1981</cd:year>
</rdf:Description> </rdf:RDF>
```

## RDF/XML

- Another example

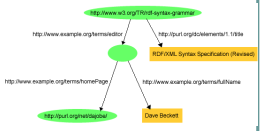
```
<?xml version="1.0"?>
<rdf:RDF >
         xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dc="http://purl.org/dc/elements/1.1/"
         xmlns:exterms="http://www.example.org/terms/">
         <rdf:Description rdf:about="http://www.example.org/index.html">
                  <exterms:creation-date>August 16, 1999</exterms:creation-date>
                  <dc:language>en</dc:language>
                  <dc:creator rdf:resource="http://www.example.org/staffid/85740"/>
         </rdf:Description>
</rdf:RDF>
```

## RDF/XML

- If value of property is a resource
  - cannot place the URI as value <property>http://….</property>
    - it would indicate that the value is that string, not that resource
  - use empty element tag + rdf:resource

## RDF/XML

- Blank Nodes:
  - realized with blank node identifiers




```
<?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:exterms="http://example.org/stuff/1.0/">
        <rdf:Description rdf:about="http://www.w3.org/TR/rdf-syntax-grammar">
                <dc:title>RDF/XML Syntax Specification (Revised)</dc:title>
                <exterms:editor rdf:nodeID="abc"/>
        </rdf:Description>
        <rdf:Description rdf:nodeID="abc">
                <exterms:fullName>Dave Beckett</exterms:fullName>
                <exterms:homePage rdf:resource="http://purl.org/net/dajobe/"/>
        </rdf:Description>
</rdf:RDF>
```

## RDF XML Syntax

- Datatypes can be associated to property values
  - rdf:datatype attribute

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
          xmlns:exterms=http://www.example.org/terms/
          xmlns:xsd="http://www.w3.org/2001/XMLSchema#">
<rdf:Description rdf:about="http://www.example.org/index.html">
        <exterms:creation-date rdf:datatype="&xsd;date">1999-08-16
        </exterms:creation-date>
</rdf:Description>
</rdf:RDF>
```

## RDF XML Syntax

- Properties as Attributes
  - property elements can be replaced by attributes

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
          xmlns:cd="http://www.recshop.fake/cd#">
<rdf:Description rdf:about="http://www.recshop.fake/cd/Abacab"
          cd:artist="Genesis"
          cd:country="USA"
          cd:company="Atlantic"
          cd:price="9.99"
          cd:year="1981" />
</rdf:RDF>
```

## RDF/XML

- We assumed that entities already have URIs
- A catalog (http://www.example.com/2002/04/products)
  - each item has an entry
  - we want to avoid writing different URI for each entry

```
<?xml version="1.0"?>
<rdf:RDF
        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:exterms="http://www.example.com/terms/"
        xmlns:xsd "http://www.w3.org/2001/XMLSchema#">
        <rdf:Description rdf:ID="item10245">
                <exterms:model rdf:datatype="&xsd;string">Overnighter</exterms:model>
                <exterms:sleeps rdf:datatype="&xsd;integer">2</exterms:sleeps>
                <exterms:weight rdf:datatype="&xsd;decimal">2.4</exterms:weight>
                <exterms:packedSize rdf:datatype="&xsd;integer">784</exterms:packedSize>
        </rdf:Description>
...other product descriptions...
</rdf:RDF>
```

## RDF/XML

- ITEM10245 is a fragment identifier
- short form for http://www.example.com/2002/04/products#item10245
- identifier must be unique in the document
  - within the document item can be referenced as #item10245 or using full URI
  - from outside the document, the full URI has to be used; e.g., someone adds info about the product:

```
<?xml version="1.0"?>
<rdf:RDF
        xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:sportex="http://www.exampleRatings.com/terms/">
        <rdf:Description rdf:about="http://www.example.com/2002/04/products#item10245">
                <sportex:ratingBy rdf:datatype="&xsd;string">Richard Roe</sportex:ratingBy>
                <sportex:numberStars rdf:datatype="&xsd;integer">5</sportex:numberStars>
        </rdf:Description>
</rdf:RDF>
```

- The base could be explicitly specified with xml:base in the rdf:RDF element

## RDF XML Syntax

- Extend RDF example with information about Genesis

```
<rdf:Description rdf:about="Genesis">
    <cd:singer>Phil Collins</cd:singer>
</rdf:Description>
```

- We wish to maintain a connection between the occurrences of Genesis:

```
…
<rdf:Description rdf:about="http://www.recshop.fake/cd/Abacab">
        <cd:artist rdf:resource="#Genesis" /> . .
…
<rdf:Description rdf:ID="#Genesis">
        <cd:singer>Phil Collins</cd:singer>
…
```

## RDF XML Syntax

- Type names can be associated to the objects described by RDF statements
  - e.g., we have RDF statements describing courses and instructors

```
<rdf:Description rdf:about="CS579">
    <rdf:type rdf:resource="&univ;course" />
      <univ:name>Semantic Web</univ:name>
      <univ:teacher rdf:resource="#tran">
</rdf:Description>
<rdf:Description rdf:about="#tran">
   <rdf:type rdf:resource="&univ;professor" />
      <univ:name>Son Cao Tran</univ:name>
      <univ:age rdf:datatype="&xsd;integer">78</univ:age>
</rdf:Description>
```

## RDF XML Syntax

- types allows simplified notation:

```
<rdf:Description rdf:ID="cs473">
  <rdf:type rdf:resource="&uni;cscourse" />
  <uni:name> Architectures </uni:name>
  <uni:taughtby>Pfeiffer</uni:taughtby>
</rdf:Description>
```

can be written as

```
<uni:course rdf:ID="cs473"
        uni:name="Architectures"
        uni:taughtby="Pfeiffer" />
```

## RDF XML Syntax

- Container elements
  - collect resources/attributes that we want to treat as a whole
  - e.g., all courses by the same professor, books with multiple authors
- Three types
  - *Bag*: unordered container, possibly with multiple occurrences
  - *Seq*: ordered container
  - *Alt*: set of alternatives (typically for a single value of a property)

## RDF XML Syntax

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:cd="http://www.recshop.fake/cd#">
<rdf:Description rdf:about="http://www.recshop.fake/cd/Beatles">
    <cd:artist>
        <rdf:Bag>
            <rdf:li>John</rdf:li>
            <rdf:li>Paul</rdf:li>
            <rdf:li>George</rdf:li>
            <rdf:li>Ringo</rdf:li>
        </rdf:Bag>
    </cd:artist>
</rdf:Description>
</rdf:RDF>
```

Implicit Blank Node for the Bag

## RDF XML Syntax

- source code of X11 may be found at ftp.example.org, ftp1.example.org, or ftp2.example.org

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:s="http://example.org/packages/vocab#">
    <rdf:Description rdf:about="http://example.org/packages/X11">
        <s:DistributionSite>
            <rdf:Alt>
                <rdf:li rdf:resource="ftp://ftp.example.org"/>
                <rdf:li rdf:resource="ftp://ftp1.example.org"/>
                <rdf:li rdf:resource="ftp://ftp2.example.org"/>
            </rdf:Alt>
        </s:DistributionSite>
    </rdf:Description>
</rdf:RDF>
```

## RDF/XML

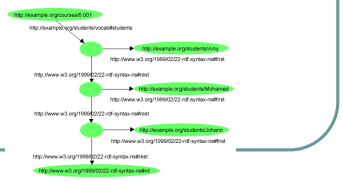
- Bag and Alt have subtle differences
  - e.g., if a book has two authors, we should use Bag
  - if a book has an author known with two different names, use Alt

## RDF XML Syntax

- Containers are not “closed”
  - no way of saying: “these are all the members of the container”
  - another graph may describe additional members
- Collections are closed

## RDF XML Syntax

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:s="http://example.org/students/vocab#">
    <rdf:Description rdf:about="http://example.org/courses/6.001">
        <s:students rdf:parseType="Collection">
            <rdf:Description rdf:about="http://example.org/students/Amy"/>
            <rdf:Description rdf:about="http://example.org/students/Mohamed"/>
            <rdf:Description rdf:about="http://example.org/students/Johann"/>
        </s:students>
    </rdf:Description>
</rdf:RDF>
```



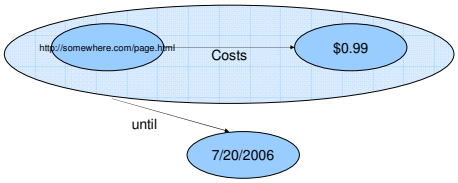
## RDF/XML

- Extended version:

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:s="http://example.org/students/vocab#">
    <rdf:Description rdf:about="http://example.org/courses/6.001">
        <s:students rdf:nodeID="sch1"/>
    </rdf:Description>

    <rdf:Description rdf:nodeID="sch1">
        <rdf:first rdf:resource="http://example.org/students/Amy"/>
        <rdf:rest rdf:nodeID="sch2"/>
    </rdf:Description>

    <rdf:Description rdf:nodeID="sch2">
        <rdf:first rdf:resource="http://example.org/students/Mohamed"/>
        <rdf:rest rdf:nodeID="sch3"/>
    </rdf:Description>

    <rdf:Description rdf:nodeID="sch3">
        <rdf:first rdf:resource="http://example.org/students/Johann"/>
        <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
    </rdf:Description>
</rdf:RDF>
```

## RDF XML Syntax

- Reification: statements about statements




## RDF/XML

- Statement:
  exproducts:item10245 exterms:weight "2.4"^^xsd:decimal
- We want to add that the creator of this statement is staff member 85740

```
<?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:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:exterms="http://www.example.com/terms/"
    xml:base="http://www.example.com/2002/04/products">
    <rdf:Description rdf:ID="item10245">
        <exterms:weight rdf:ID="triple12345" rdf:datatype="&xsd;decimal">2.4 </exterms:weight>
    </rdf:Description>

    <rdf:Description rdf:about="#triple12345">
        <dc:creator rdf:resource="http://www.example.com/staffid/85740"/>
    </rdf:Description>
</rdf:RDF>
```

## RDF Schema (RDFS)

- Statements describe individuals
- Class = set of individuals (instances of the class)
- E.g., rdf:type defines a class
- Classes allows us to restrict content of statements, e.g.,

  RoomSH124 teaches Semantic Web
- impose restrictions on which objects a property can be applied to
- RDF provides facilities to describe
  - classes
  - properties
  - which classes and properties should be used together

## RDF Schema (RDFS)

- Classes related via inheritance
  - professors are staff members
  - staff members are university employees
  - …
- A *subclass of* B: each instance of A is an instance of B
- Class hierarchy
- Class can have multiple superclasses

## RDF Schema (RDFS)

- Important: e.g., if schema says that
  *Courses can be only taught by academic members*
- Statement says
  *Enrico is a professor*
- Thus, Enrico cannot teach
- We need:
  *professor subset of academic members*

## RDF Schema (RDFS)

- Note differences w.r.t. Java hierarchies
  - properties are global, not part of class declarations
  - one can add "globally" new declarations to existing classes
- Similarly: hierarchies of properties
  - *driverof* is a subproperty of *operatorof*
  - in general: P subproperty of Q if

$$\forall x, y.P(x, y) \rightarrow Q(x, y)$$

## RDF Schema (RDFS)

- RDF Schema builds on RDF
  - resources and properties
  - hierarchies = labeled graphs
  - E.g., professor is a subclass of academic staff:
    - subject = professor
    - property = subclass of
    - object = academic staff

## RDF Schema (RDFS)

- Core Classes:
  - rdfs:Resource = class of all resources
  - rdfs:Class = class of all classes
  - rdfs:Literal = set of all literals (strings)
  - rdf:Property = class of all properties
  - rdf:Statement = class of all reified statements

## RDF Schema (RDFS)

- E.g., class professor

```
<rdfs:Class rdf:ID="professor">
....
</rdfs:Class>
```

- Core Properties for defining relationships
  - rdf:type relates a resource to a class (instance of)
  - rdfs:subClassOf relates a class to a superclass
  - rdfs:subPropertyOf relates a property to a superproperty
- E.g., all professors are academic staff

```
<rdfs:Class rdf:ID="professor">
  <rdfs:subClassOf rdf:resource="#academicstaff" />
</rdfs:Class>
```

## RDF Schema (RDFS)

- subClassOf is meant to be transitive

```
<rdfs:Class rdf:ID="Van">
   <rdfs:subClassOf rdf:resource="#Vehicle" />
</rdfs:Class>
<rdfs:Class rdf:ID="MiniVan">
   <rdfs:subClassOf rdf:resource="#Van">
</rdfs:Class>
```

## RDF Schema (RDFS)

- Assume that the vehicle classes are described in http://www.example.org/schemas/vehicles
- One instance can be created

```
<?xml version="1.0"?>
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:ex="http://example.org/schemas/vehicles#"
   xml:base="http://example.org/things">
   <ex:MotorVehicle rdf:ID="companyCar"/>
</rdf:RDF>
```

## RDF Schema (RDFS)

- Core Properties for Restricting Properties
  - rdfs:domain defines domain of a property (class of resources that may appear as subjects of the property)
  - rdfs:range defines resources that can appear as objects of the property
- E.g., age applies to professors as resources and literals as object

```
<rdf:Property rdf:ID="age">
   <rdfs:domain rdf:resource="#professor" />
   <rdfs:range rdf:resource="&rdf;Literal"/>
</rdf:Property>
```

## RDF Schema (RDFS)

- Each property can have zero, one or more range property
  - zero: nothing known about range
  - more than one: then all the conditions should be met
    - if property ex:hasMother is defined to have range ex:Person and ex:Female, than the object of ex:hasMother should belong to both classes
- Each property can have zero, one, or more domain property
  - zero: applicable to any entity
  - more than one: the element to which the property is applied must belong to all the Classes

## RDF Schema (RDFS)

```
<rdf:Property rdf:ID="registeredTo">
   <rdfs:domain rdf:resource="#MotorVehicle"/>
   <rdfs:range rdf:resource="#Person"/>
</rdf:Property>

<rdf:Property rdf:ID="rearSeatLegRoom">
   <rdfs:domain rdf:resource="#PassengerVehicle"/>
   <rdfs:range rdf:resource="&xsd;integer"/>
</rdf:Property>

<rdfs:Class rdf:ID="Person"/>
```

## RDF Schema (RDFS)

- Specialize Properties and Classes
  - rdfs:subPropertyOf
    - property ex:primaryDriver
    - property ex:Driver
    - state that ex:primaryDriver is a specialization of ex:Driver
      ex:driver rdf:type rdf:Property
      ex:primaryDriver rdf:type rdf:Property
      ex:primaryDriver rdfs:subPropertyOf ex:Driver
    - if exstaff:Fred is a primaryDriver of ex:companyVan, then it is also a Driver for the ex:companyVan

```
<rdf:Property rdf:ID="driver">
   <rdfs:domain rdf:resource="#MotorVehicle"/>
</rdf:Property>
<rdf:Property rdf:ID="primaryDriver">
   <rdfs:subPropertyOf rdf:resource="#driver"/>
</rdf:Property>
```

## RDF Schema (RDFS)

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xml:base="http://example.org/schemas/vehicles">
    <rdfs:Class rdf:ID="MotorVehicle"/>
    <rdfs:Class rdf:ID="PassengerVehicle">
        <rdfs:subClassOf
rdf:resource="#MotorVehicle"/>
    </rdfs:Class>
    <rdfs:Class rdf:ID="Truck">
        <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
    </rdfs:Class>
    <rdfs:Class rdf:ID="Van">
        <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
    </rdfs:Class>
    <rdfs:Class rdf:ID="MiniVan">
        <rdfs:subClassOf rdf:resource="#Van"/>
        <rdfs:subClassOf rdf:resource="#PassengerVehicle"/>
    </rdfs:Class>
    <rdfs:Class rdf:ID="Person"/>

    <rdf:Property rdf:ID="registeredTo">
        <rdfs:domain rdf:resource="#MotorVehicle"/>
        <rdfs:range rdf:resource="#Person"/>
    </rdf:Property>

    <rdf:Property rdf:ID="rearSeatLegRoom">
        <rdfs:domain rdf:resource="#PassengerVehicle"/>
        <rdfs:range rdf:resource="&xsd;integer"/>
    </rdf:Property>
    <rdf:Property rdf:ID="driver">
        <rdfs:domain rdf:resource="#MotorVehicle"/>
    </rdf:Property>

    <rdf:Property rdf:ID="primaryDriver">
        <rdfs:subPropertyOf rdf:resource="#driver"/>
    </rdf:Property>
</rdf:RDF>
```

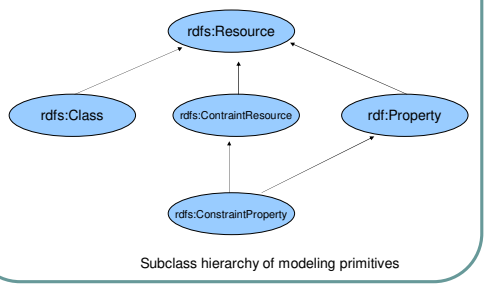
## RDF Schema (RDFS)

- An instance:

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:ex="http://example.org/schemas/vehicles#"
    xml:base="http://example.org/things">

<ex:PassengerVehicle rdf:ID="johnSmithsCar">
    <ex:registeredTo rdf:resource="http://www.example.org/staffid/85740"/>
    <ex:rearSeatLegRoom rdf:datatype="&xsd;integer">127</ex:rearSeatLegRoom>
    <ex:primaryDriver rdf:resource="http://www.example.org/staffid/85740"/>
 </ex:PassengerVehicle>
</rdf:RDF>
```

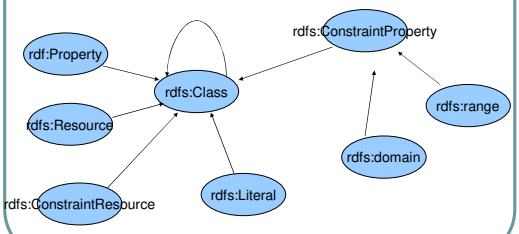
- rdfs:ConstraintResource  class of all constraints
- rdfs:ConstraintProperty all properties that define constraints (superclass of domain and range)

## RDF Schema (RDFS)




Subclass hierarchy of modeling primitives

## RDF Schema (RDFS)




## RDF Schema (RDFS)

- Containers
  - rdf:Bag class of bags
  - rdf:Seq class of all sequences
  - rdf:Alt class of all alternatives
  - rdf:Container superclass of all container classes
- Utility Properties: properties used to link resources described in many places
  - rdfs:seeAlso relates a resource to another one
  - rdfs:isDefinedBy subproperty of seeAlso, relates resource to the place where its definition can be found
  - rdfs:comment comment for a resource
  - rdfs:label:human friendly label

## RDF Schema (RDFS)

- Example: a university

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

<rdfs:Class rdf:ID="lecturer">
  <rdfs:comment>
        The class of lectures; all academic staff
  </rdfs:comment>
  <rdfs:subClassOf rdf:resource="#academicstaff"/>
</rdfs:Class>
```

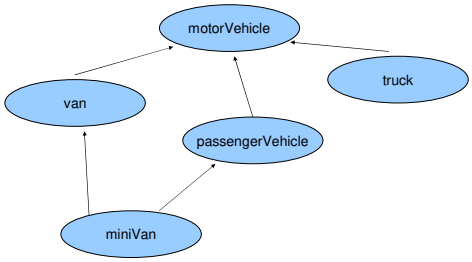
## RDF Schema (RDFS)

```
<rdfs:Class rdf:ID="academicstaff">
  <rdfs:comment> All the academic staff </rdfs:comment>
  <rdfs:subClassOf rdf:resource="#staffmember"/>
</rdfs:Class>

<rdfs:Class rdf:ID="staffmember">
  <rdfs:comment> all staff members </rdfs:comment>
</rdfs:Class>

<rdfs:Class rdf:ID="course">
  <rdfs:comment> all courses </rdfs:comment>
</rdfs:Class>
```

## RDF Schema (RDFS)

```
<rdf:Property rdf:ID="involves">
  <rdfs:comment>relates only courses to lectures
  </rdfs:comment>
  <rdfs:domain rdf:resource="#course" />
  <rdfs:range rdf:resource="#lecturer" />
</rdf:Property>

<rdf:Property rdf:ID="isTaughtBy">
  <rdfs:subPropertyOf rdf:resource="#involves" />
</rdf:Property>

<rdf:Property rdf:ID="phone">
  <rdfs:domain rdf:resource="#staffmember">
  <rdfs:range rdf:resource="&rdf;Literal">
</rdf:Property>

</rdf:RDF>
```

## RDF Schema (RDFS)




## RDF Schema RDFS

```
<rdf:RDF xmlns:rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#
   xmlns:rdfs=http://www.w3.org/2000/01/rdf-schema#>

<rdfs:Class rdf:ID="motorVehicle" />

<rdfs:Class rdf:ID="van">
   <rdfs:subClassOf rdf:resource="#motorVehicle" />
</rdfs:Class>

<rdfs:Class rdf:ID="truck">
   <rdfs:subClassOf rdf:resource="#motorVehicle" />
</rdfs:Class>

<rdfs:Class rdf:ID="passengerVehicle">
   <rdfs:subClassOf rdf:resource="#motorVehicle" />
</rdfs:Class>
```

## RDF Schema (RDFS)

```
<rdfs:Class rdf:ID="miniVan">
   <rdfs:subClassOf rdf:resource="#passengerVehicle"/>
   <rdfs:subClassOf rdf:resource="#van"/>
</rdfs:Class>
```

## Querying RDF

- RDF is encoded in XML
- but XPath is too low level to be useful
- E.g., retrieve titles of lecturers

```
<rdf:Description rdf:about="xxxxx">
  <rdf:type rdf:resource="&uni;lecturer"/>
  <uni:name>Enrico</uni:name>
  <uni:title>Professor</uni:title>
</rdf:Description>
```

use query:

```
/rdf:Description[rdf:type=http://www.mydomain.org/uni-ns#lecturer]/uni:title
```

## Querying RDF

- But the same can be written

```
<uni:lecturer rdf:about="xxxx">
  <uni:name>Enrico</uni:name>
  <uni:title>Professor</uni:title> </uni:lecturer>
```

which would require the query

```
//uni:lecturer/uni:title
```

- Better to have a uniform query language

## Querying RDF: RQL

- Basic Queries
  - Class: returns all the classes
  - Property: returns all the properties
  - To retrieve all instances of a class (course):

    course

    query will also return instances of subclasses
  - If we do not want instances of subclasses

    ^course
  - If we want to retrieve triples with a property (involve)

    involve

## Querying RDF

- Select-from-where
  - Select: number and order of retrieved data
  - From: navigate the data model
  - Where: constraints on solutions
- E.g., retrieve phone of staff members
  select X,Y from {X} phone {Y}
  - {X}phone{Y} is a resource-property-value
- Restrict only to lecturers
  select X,Y from lecturer{X}.phone{Y}
  - lecturer{X} instantiate X with instances of lecturer
  - from each one collect the value of phone property
  - '.' is an implicit join of two queries

## Querying RDF

- Retrieve names of courses taught by lecturer with ID 9999

```
select N
  from course{X}.isTaughtBy{Y},
       {C}name{N}
  where Y="9999" and X=C
```

## Querying RDF

- We can also query RDFS
  - $X and @X are schema variables
  - $X is a class variable
  - @X is a property variable

```
select X, $X, Y, $Y
  from {X:$X}phone{Y:$Y}
```

retrieves resource/values connected by property phone and their classes