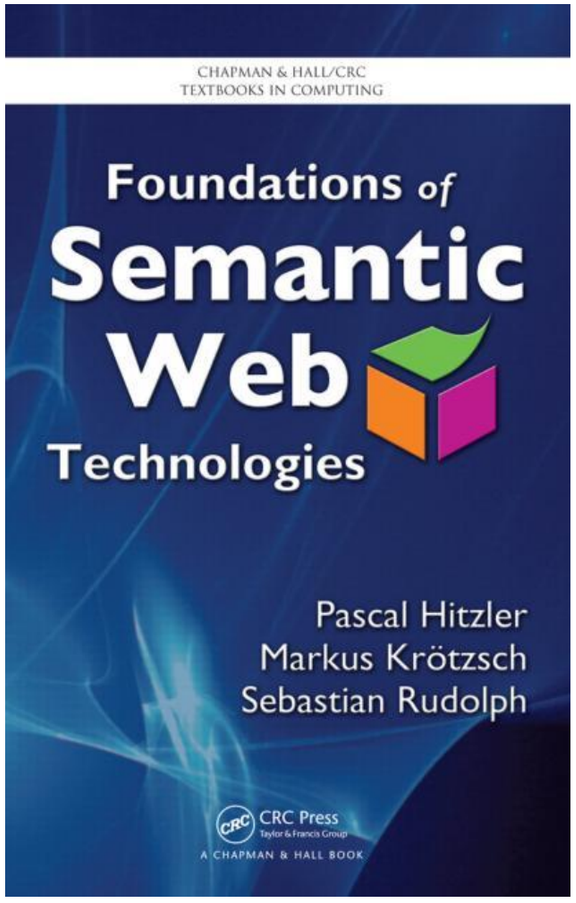
Australian National University

COMP3425/COMP8410 Data Mining

Semantic Web and Linked Data

Attribution

- This material has been developed by Kerry Taylor, incorporating material previously developed by Kerry, Armin Haller and David Ratcliffe at various times.
- This book is a source of many of the examples and is recommended for further reading.
- Pascal Hitzler, Markus Krotzsch, Sebastian Rudolph, **Foundations of Semantic Web Technologies,** CRC Press 2009.



What is the Semantic Web?

"The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation."

T. Berners-Lee, J. Hendler, O. Lassila, "The Semantic Web", Scientific American, May 2001

By 2009 "linked data" had become *the* "application of the Semantic Web"

Fundamentals: Semantic Web "Layer Cake"

URIs, RDF, RDFS, SPARQL, OWL

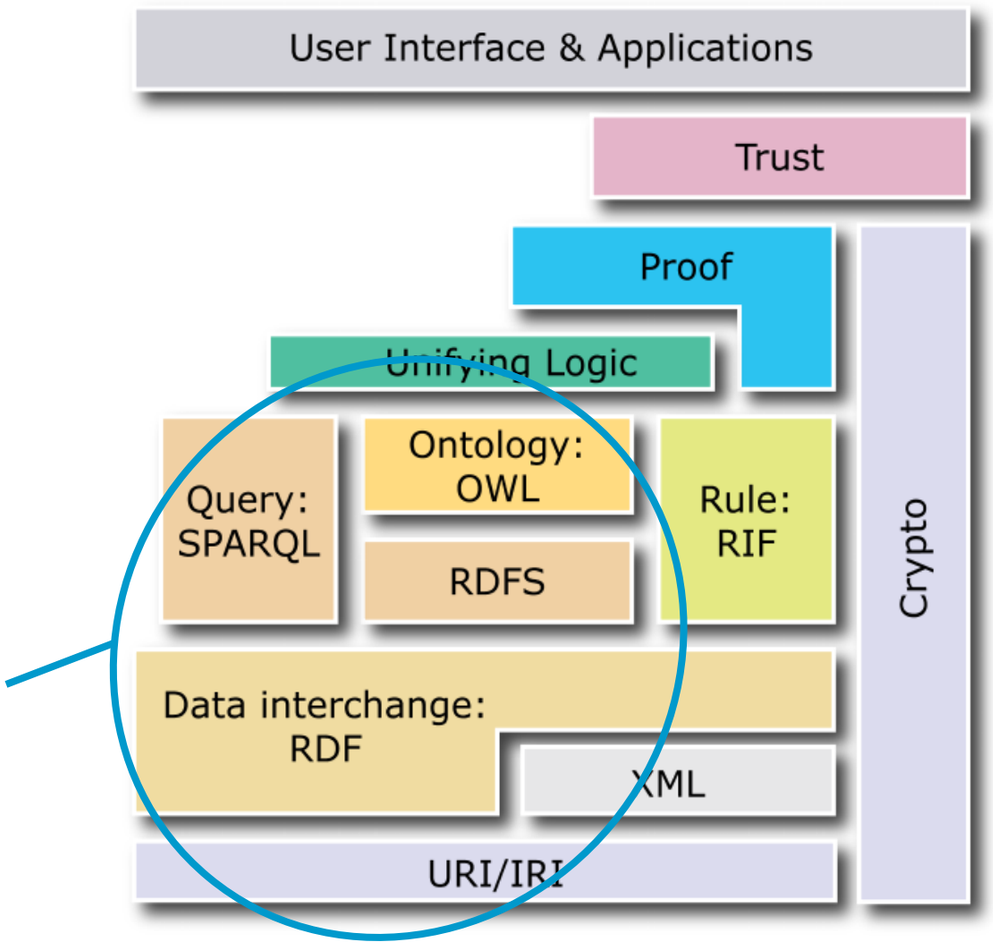


image source: W3C

The Semantic Web aka Linked Open Data

- *Web of data* where web content is processed by machines, without direct human readers, but not a separate Web.
- The web as a huge, dynamic, evolving database of facts, rather than pages, that can be interpreted and presented in many ways (mashups).
- Fundamental importance of *ontologies* to describe the fact that represents the data.
- Where is the “semantics”?
  - RDF(S) emphasises labelled links as the source of meaning: essentially a graph model either stored locally in a triple store or distributed as linked open data. A label (that is a URI) uniquely identifies a concept and reuse of the label implies the *same* meaning.
  - OWL adds emphasis on inference as the source of meaning: a label also refers to a package of logical axioms with a proof theory over a model-theoretic semantics.
  - Mostly, the two notions of meaning coincide.

Now what do we need to build it?

1. Identity
   - so things can be referred to

2. Relationships
   - so things can be connected to/ described by other things

3. Inference
   - so general statements about things can be applied to particular things and don't need to be restated every time

4. A way of encoding all this
   - normatively in RDF 1.1, Turtle, RDF/XML, N-Triples, N-Quads, TriG, JSON-LD

And the **Open World Assumption –** all information to hand is always only partial, i.e. from the absence of a statement alone, a deductive reasoner cannot (and must not) infer that the statement is false.

URIs

Universal Resource Identifier

Designing URIs for Linked Data

- Tim Berners-Lee's "four rules" (2006)
  http://www.w3.org/DesignIssues/LinkedData.html

1. Use URIs as names for things
2. Use HTTP URIs so that people can look up those names
3. When someone looks up a URI, provide useful information, using the standards (RDF*, SPARQL)
4. Include links to other URIs so that they can discover more things.

- See also "Cool URIs for the Semantic Web", (2008)
  http://www.w3.org/TR/cooluris/

Identity: Uniform Resource Identifiers

- "Things" in the semantic web, even very abstract ones, are uniquely named by URIs.
- A URI is an identifier, not necessarily a locator as is a URL.
- IRIs (International Resource Identifiers) allow extended characters and can be used in place of URIs.

Syntax (absolute):

<scheme> : <path> [? <query>] [# <fragment>]

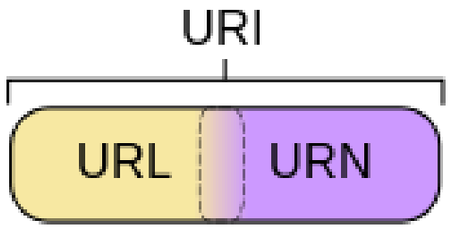
Examples:

ANU website:

http://www.anu.edu.au/

The actual ANU itself**:**

http://anu.edu.au/resource#id



URIs are more general than URLs (and less so than IRIs!)

URIs include URNs (e.g., urn:oid:2.16.840)

Used extensively in the Linked Open Data web

Many constraints/recommendations about URIs to come…

Where does a URI come from?

- Could be a common namespace which declares a vocabulary
  - e.g. http://www.w3.org/1999/02/22-rdf-syntax-ns#Description
  - e.g. http://purl.org/dc/terms/
- Can be made up on the fly – does not need to be resolvable but is meant to be *unique*, i.e. only deliberately reused, and *stable* i.e. unchanging
- In some areas careful allocation schemes exist (e.g. DOIs)
- URLs are often used because:
  - inventor has some control over the names.
  - Its nice to have something returned to a browser via a URI
- But can be confusing– is the thing intended to be the document, or perhaps just the thing that the document is about?
  - Using a fragment identifier that does not occur in the doc does this
  - Can also use content-negotiation/http redirect protocol to give the client what they ask for (very commonly used for linked data)

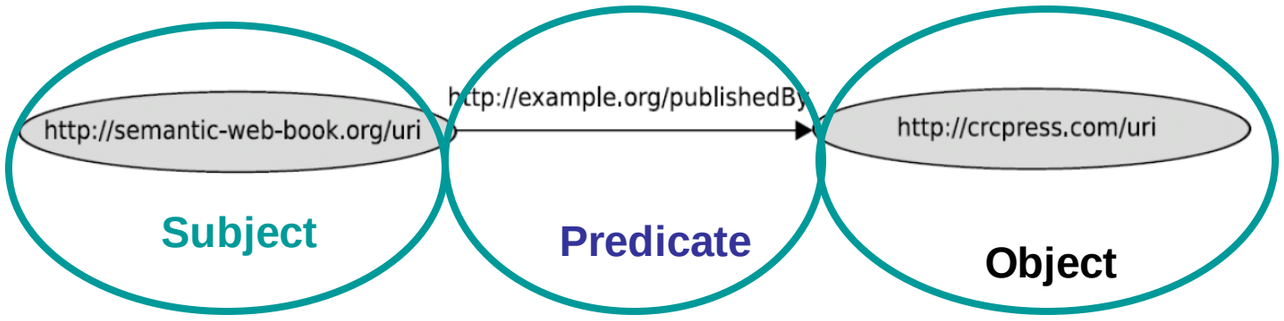
RDF

Resource Description Framework

Relationships: RDF

- RDF (Resource Description Framework) originally developed for representation of Metadata (data about resources e.g. Dublin Core), but now designed for data itself.
- Uses a flexible graph data model which doesn't require much design in advance, is easily extensible and seems to fit with the Web idea of decentralised creation and rich interconnections
- Graphs are defined by nodes (usually labelled), which are connected by directed (one-way), labelled arcs.

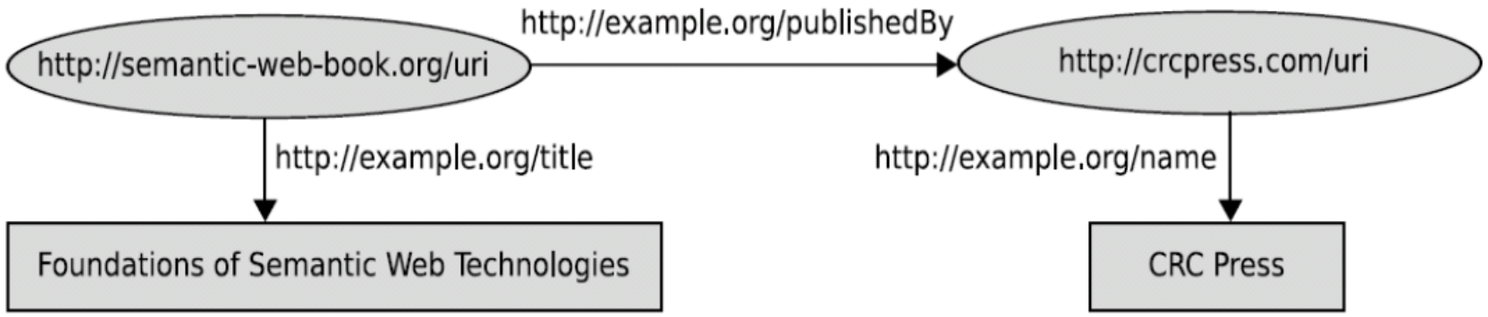
e.g. **The semantic-web-book is published by the CRC Press**



A Graph is a Set of Triples

3 *triples* (or *statements*):

```
The semantic-web-book  is published by the CRC press.
The semantic-web-book has a title “Foundations of Semantic Web Technologies”.
The CRC press is named “CRC Press”.
```



RDF is comprised of

- ***URIs***, ***Literals*** and ***Blank nodes***
  - **Subjects** must be ***URIs*** or ***Blank nodes.***
  - **Predicates** (also called **properties**) must be ***URIs***.
  - **Objects** may be ***URIs*** or ***Blank nodes*** or ***Literals***.

- Serialisation encoding (e.g. Turtle, RDF/XML, JSON-LD)
- XSD data types for Literals (e.g. `xsd:string`, `xsd:boolean`, `xsd:decimal`)
- Many syntactic shortcuts for encoding complex structures e.g. lists
- A formal model-theoretic semantics

Namespaces

- RDF language primitives are defined (by RDFS) in the namespace

```
http://www.w3.org/1999/02/22-rdf-syntax-ns#
```

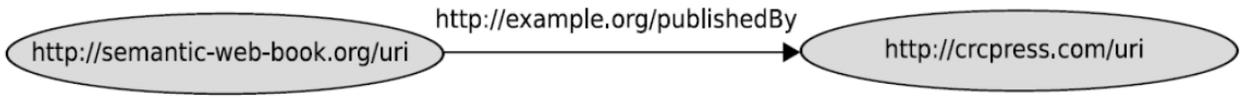
- Which is conventionally abbreviated by “rdf”, so the first part of an RDF document in text/turtle serialisation looks like this:

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix example: <http://example.org/> .
```

- We have introduced the essential “rdf” namespace, and also our own “example” namespace here.

- A *prefix* simply defines a simple macro abbreviation for the leftmost part of a URI.

Turtle syntax for an RDF triple

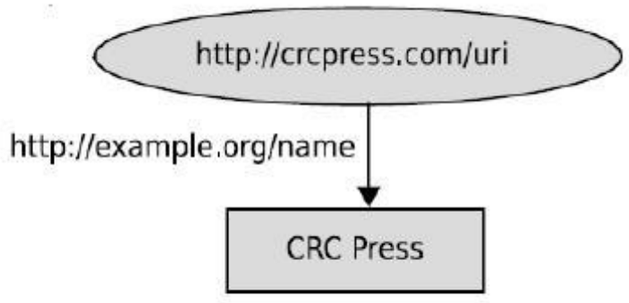



```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix ex: <http://example.org/> .

<http://semantic-web-book.org/uri>
    ex:publishedBy <http://crcpress.com/uri> .
```

In this case, the *object* is a URI (identifying the CRC Press)

Turtle Syntax for a triple with a literal object

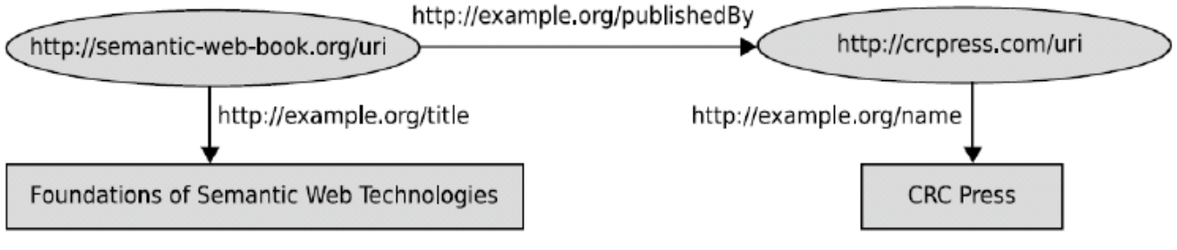


Here, the ex:name of the crcpress subject has a string value, “CRC Press”.

```
<http://crcpress.com/uri>
        ex:name “CRC Press” .
```

A graph of related triples: Abbreviations

- A subject with multiple properties (semantic-web-book)
- An object that is also a subject (crcpress)
- Relations with the same subject may be nested




```
http://semantic-web-book.org/uri
        ex:title "Foundations of Semantic Web Technologies" ;
        ex:publishedBy <http://crcpress.com/uri> .

<http://crcpress.com/uri> ex:name "CRC Press" .
<http://crcpress.com/uri> ex:name "CRC Press" , "Chapman and Hall" .
```

Base namespace: shorter than prefix

- A new base IRI can be defined using the '@base' or 'BASE' directive in Turtle, or rely on the base URI of the document (the URL it came from).
- A string without a prefix in a URI attribute value is interpreted as prefixed with the base string.

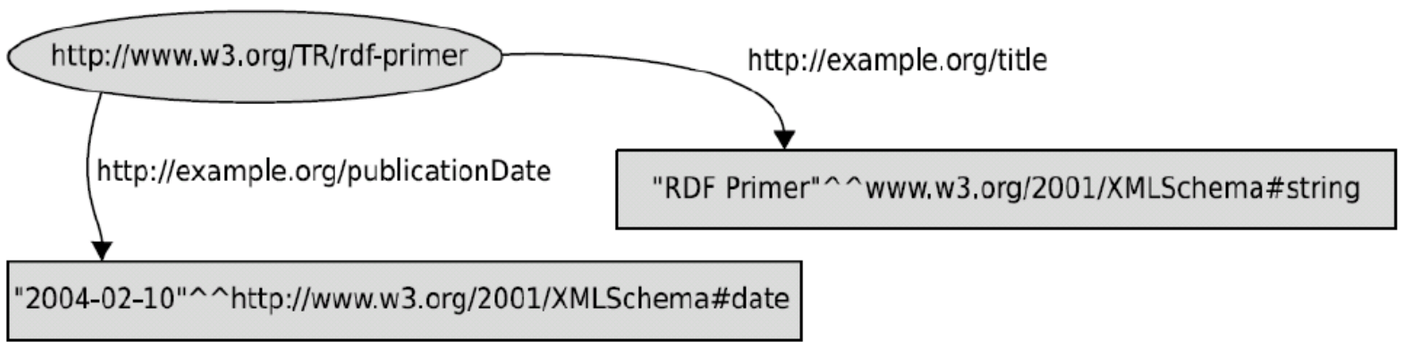
```
@base <http://semantic-web-book.org/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix ex: <http://example.org/> .

<uri> ex:title "Foundations of Semantic Web Technologies" ;
       ex:publishedBy <http://crcpress.com/uri> .

<http://crcpress.com/uri> ex:name "CRC Press" .
```

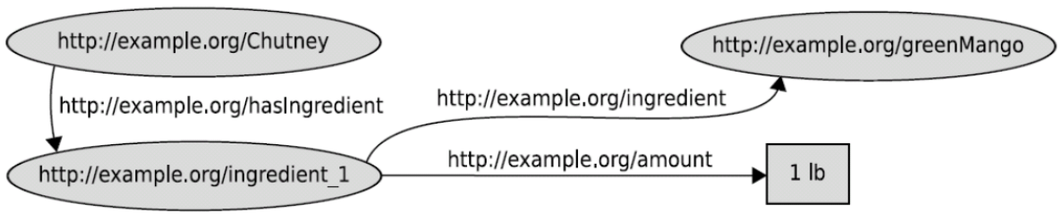
RDF Datatypes for Literals

Any URI can be used for the datatype, but generally XML Schema datatypes are used.




```
http://www.w3.org/TR/rdf-primer
ex:title
 "Foundations of Semantic Web Technologies"^^<http://www.w3.org/2001/XMLSchema#string> ;
ex:publicationDate
 "2004-02-10"^^<http://www.w3.org/2001/XMLSchema#date> .
```

N-ary or many-valued Relationships



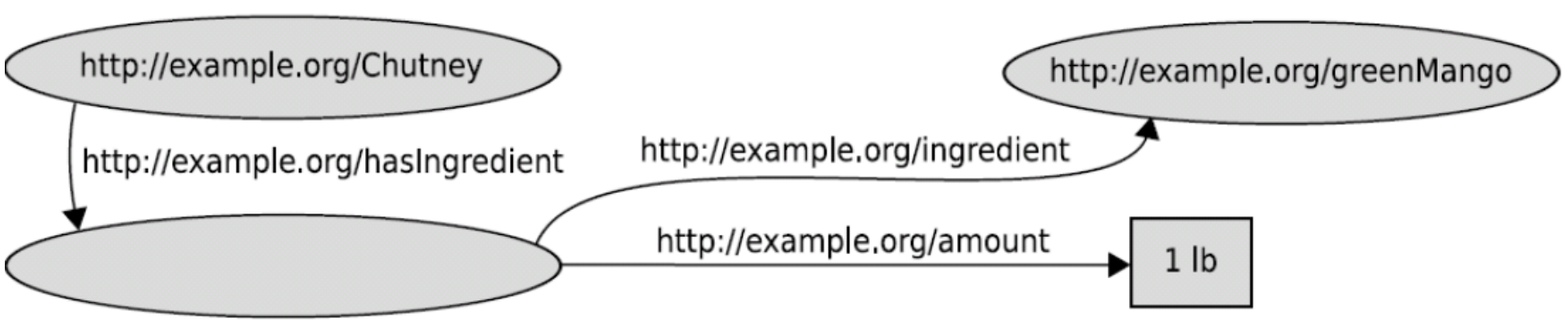
Representing a tuple relationship like *"a recipe has an ingredient and an amount of that ingredient"* can be done by introducing a new object to signify each pair (ingredient_1, ingredient_2, ingedient_3..) and a new property for all the pairs (hasIngredient)

```
ex:Chutney ex:hasIngredient ex:ingredient_1 .

ex:ingredient_1 ex:ingredient ex:greenMango ;
        ex:amount "1 kg" .
```

Blank nodes

- You don't really need to name all those introduced nodes.
- Nodes without a URI are called ***Blank nodes*** or ***bnodes***.
- There are no blank properties.




```
ex:Chutney ex:hasIngredient _:nodeXYZ .

_:nodeXYZ ex:ingredient ex:greenMango ;
          ex:amount "1 lb" .
```

Australian National University

RDFS

Resource Description Framework Schema

RDF Schema: RDFS

- Express *schema* knowledge, or terminological knowledge, about classes of things. Also called a *vocabulary* to be re-used.
- RDF Schema is a vocabulary expressed in RDF.
- Inference is needed to relate the knowledge about the class to the knowledge about the individual thing.
- As well as class hierarchies, RDFS supports property hierarchies.
- RDF Schema is tightly connected to RDF– commonly both are used together and often called RDF(S).
- The RDFS namespace is `http://www.w3.org/2000/01/rdf-schema#`

Classes and Instances

Classes are names for sets of things. A thing is asserted to belong to a class using the `rdf:type` property.

In Turtle serialisation the token 'a' in the predicate position represents the IRI `http://www.w3.org/1999/02/22-rdf-syntax-ns#type .`

Example: ***Princeton is a member of class University.***

```
<http://www.princeton.edu> a <http://www.example.org/University> .
```

A thing can belong to multiple classes.

NB This is membership, *not* subset or subclass.

Class Hierarchies

- Classes can be named with `rdfs:Class` (by stating the class is an `rdf:type` of `rdfs:Class`)
- Classes can be arranged in class hierarchies using `rdfs:subClassOf`.
- `rdfs:subClassOf` is transitive and reflexive.

```
ex:Primates rdf:type rdfs:Class .

ex:Primates rdfs:subClassOf ex:Mammalia .
```

  - So **transitive**: something that is an `rdf:type` of Primate is also an `rdf:type` of Mammalia without needing explicit assertion.
  - And **reflexive**: Primates are also `rdfs:subClassOf` Primates

Property Hierarchies & Property Restrictions

- `rdf:Property` is the class of all properties
- Like classes, properties can be arranged in hierarchies using the transitive, reflexive `rdfs:subPropertyOf` property.

```
ex:hasSon rdfs:subPropertyOf ex:hasChild .
```

- `rdfs:domain` & `rdfs:range` (both instances of `rdf:Property`) state that a property can only hold between things of a certain `rdf:type`. Can be used to infer the `rdf:type` of a thing that is asserted to *have* the property (subject) or *be* the property (object). Also works for datatypes.

```
ex:hasSon rdfs:domain ex:Father .
ex:hasSon rdfs:range ex:Son .
ex:KirkDouglas ex:hasSon ex:MichaelDouglas .

ex:hasAge rdfs:range xsd:nonNegativeInteger .
```

RDFS properties for metadata

- `rdfs:label` – to give a readable/printable short label for the thing (URIs are rarely readable enough; tools may use the label)
- `rdfs:comment` – explanatory text of some kind
- `rdfs:seeAlso` – a thing (URI) that explains it somehow further
- `rdfs:isDefinedBy` – like seeAlso but usually refers to an RDF schema where it is first defined

```
ex:Primates rdf:type rdfs:class ;
       rdfs:label "Primates" ;
       rdfs:comment "Order of mammals. Primates are
characterized by an advanced brain. They mostly populate the
tropical earth regions. " ;
       rdfs:seeAlso dbpedia:Primates ;
       rdfs:subClassOf ex:Mammalia .
```

Australian National University

SPARQL

SPARQL Protocol and RDF Query Language

How to query RDF data: SPARQL

- SPARQL: SPARQL Protocol and RDF Query Language
- SPARQL 1.1. became a W3C recommendation in March 2013: http://www.w3.org/TR/sparql11-overview/
- Spec covers query language, protocol for interaction with a SPARQL endpoint, result format (XML, JSON, CSV, TSV).
- Query language is about graph pattern matching.
- You can query for unknown relationships.
- Warning: Many SPARQL endpoints do not provide RDFS inference
  - can be done with a reasoner (once off or incrementally) on load
  - can be done dynamically in response to query
  - may not be done at all! no obvious way to tell.

SPARQL query structure

prefix declarations
PREFIX foo: <http://example.com/resources/> …

dataset definition (stating which RDF graph(s) to use)
FROM …

result clause (identifying what to return)
SELECT …

query pattern (specify what to query for, = graph pattern + filter)
WHERE { … }

query modifiers (ordering, rearranging or modifying query results)
ORDER BY …

SPARQL Example

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX ex: <http://example.org/>

SELECT ?label
WHERE {
  ?subj rdfs:label ?label ;
        a ex:Mammalia
FILTER(REGEX(?label, "*Africa*"))
}
```