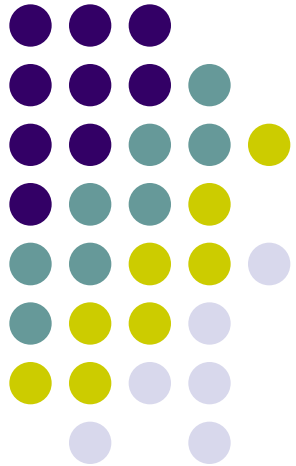
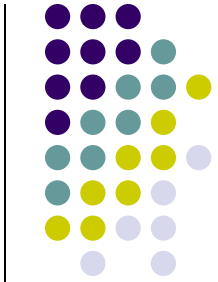
RDF

Acknowledgement to Ian Davis from Talis
http://research.talis.com/2005/rdf-intro/

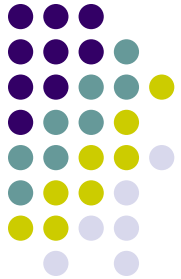


Overview



- Why RDF?
- What is RDF?
- URI
- Data Integration

Why RDF?



Why RDF

- Why should I use RDF, why not just XML?
- How to express following graph

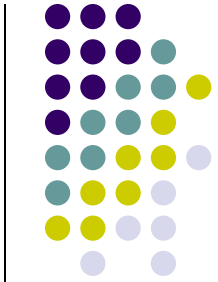
Why RDF

- In RDF, it is trivial (simply a triple):
  - triple(page, hasAuthor, Ora)
- In XML, there are many ways:

```
Way1:
<author>
     <uri>page</uri>
     <name>Ora</name>
</author>

Way2:
<document href="page">
   <author>Ora</author>
</document>

Way3:
<document>
   <details>
    <uri>href="page"</uri>
    <author>
        <name>Ora</name>
    </author>
    </details>
</document>
```

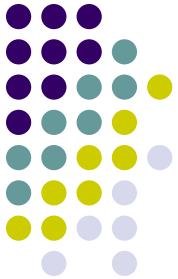
```
Way4:
<document>
   <author>
    <uri>href="page"</uri>
    <details>
        <name>Ora</name>
    </details>
    </author>
</document>

Way5:
<document
href="http://www.w3.org/test/p
age" author="Ora" />
```

Why RDF



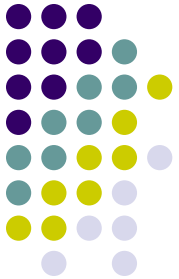
- RDF will provide machine the knowledge tree (semantic graph), where order is not important. – RDF data model is labeled (unordered) graph with two kinds of nodes (resources and literals).
- While XML will give a person a document tree where the order or serilaization is important. – XML data model is node-labeled tree (with left-to-right ordered)

What can we use RDF?



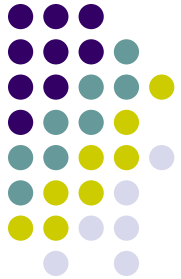
- Representing information about resources in the WWW
  - Describing propoerties for shopping items, such as price and availability
  - Describing time schedules for web events
  - Describing information about web pages, such as content, author, created and modified date
  - Describing content and rating for web pictures
  - Describing content for search engines
  - Describing electronic libraries

What is RDF



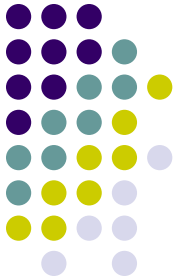
- It provides a model for data, and a syntax so that independent parties can exchange and use it.
- RDF is designed mainly to be read and understood by computers
- RDF is not designed for being displayed to people
- RDF is written in XML
  - Any XML processor, parsers can parse and process RDF
  - The XML language used for RDF is called RDF/XML

RDF – Web standard



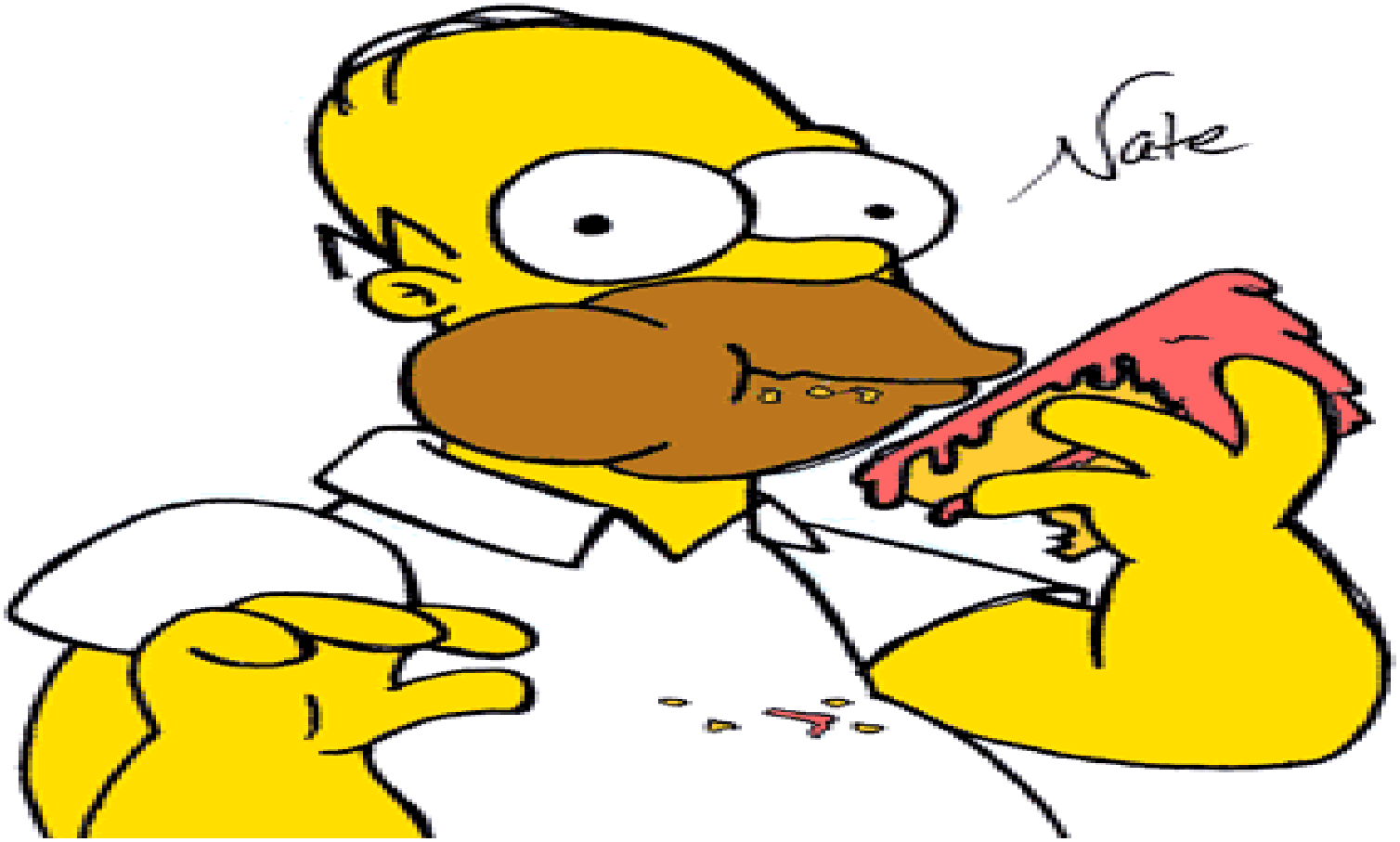
- RDF became a W3C Recommendation in 2004
- A W3C Recommendation is understood by the industry and the Web community as a web standard.
- A W3C Recommendation is a stable specification developed by a W3C Working Group and reviewed by the W3C Membership.

RDF and the Semantic Web

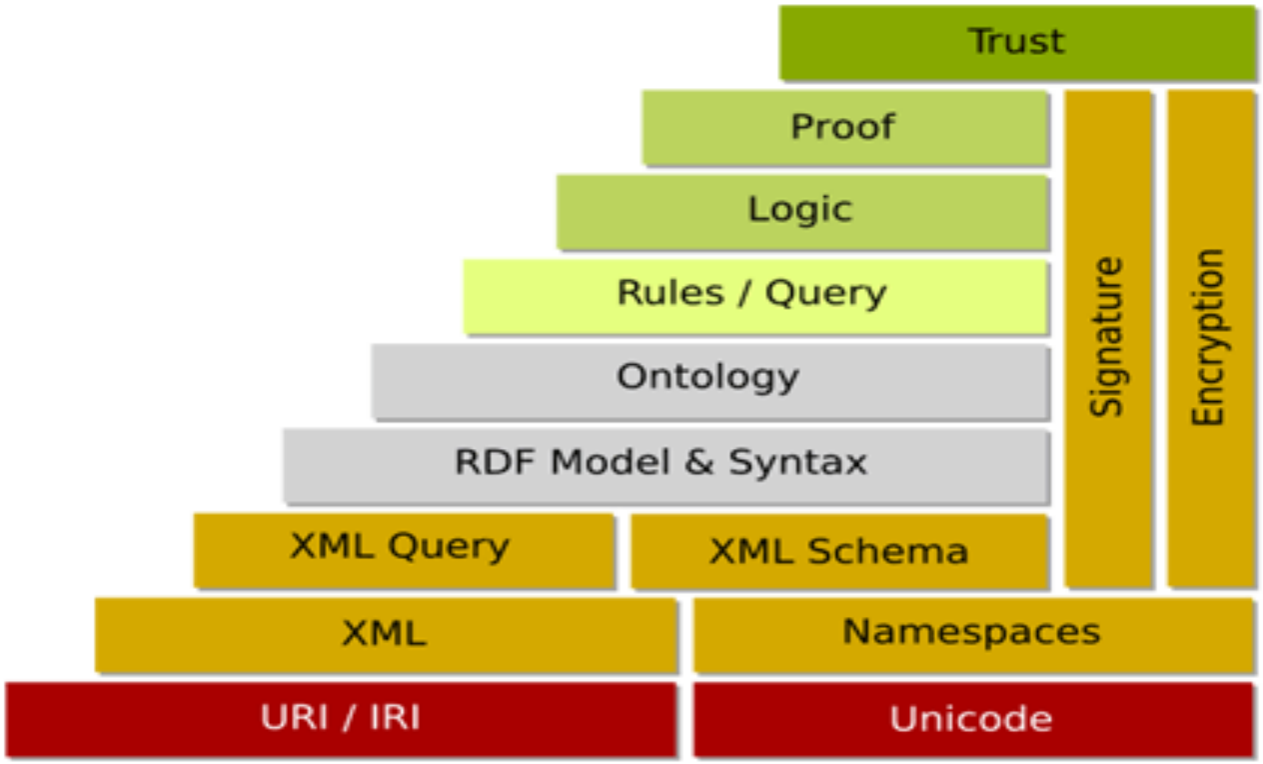


- RDF is the key part of the Semantic Web activities. It helps to realize the vision of the Semantic Web that:
  - Web information should have exact meaning
  - Web information can be understood and processed by computers
  - Computers can integrate information from the Web

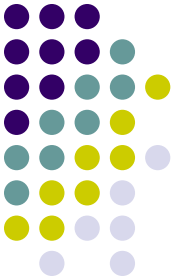
SW Layer Cake



SW Layer Cake

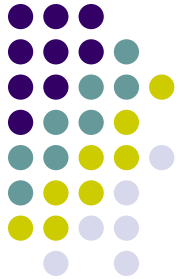


Why do we need RDF?

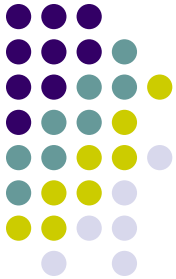


- The web is a global, universal information space for documents
- Can we do same for data?
- Make the web into a big database?
- RDF is the data format for that database

What is RDF?



The relational model



- A typical relational database table for books

isbn	title	author	publisherID	pages
0596002637	Practical RDF	Shelley Powers	7642	350
0596000480	Javascript	David Flanagan	3556	936
. . .	. . .	. . .	. . .	. . .
. . .	. . .	. . .	. . .	. . .

The rows represent the things you are storing

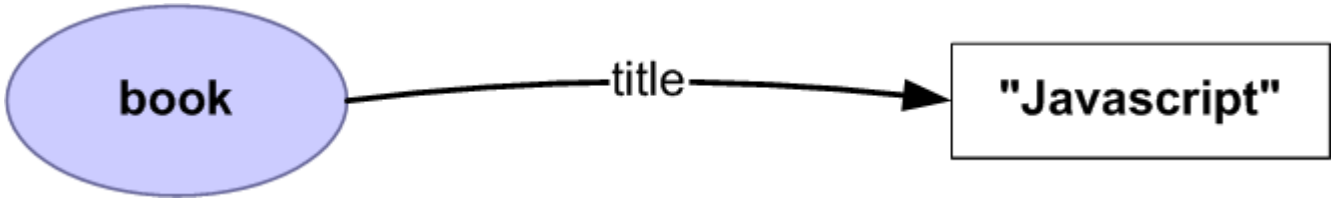
The columns represent the properties or attributes of those things

The intersection gives the value of that property for that thing

The book has a title with value "Javascript"

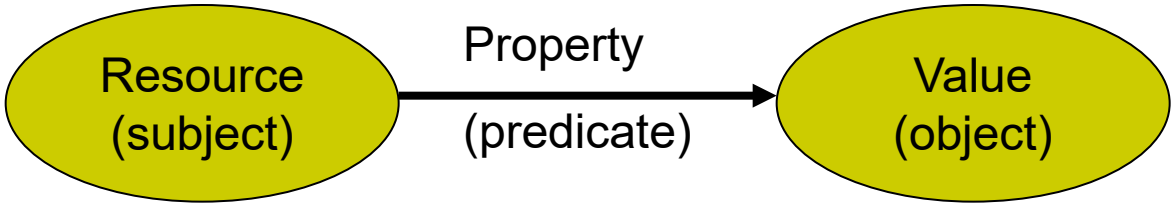
The relational model

- The book has a title with value “Javascript”



Basic Ideas behind RDF

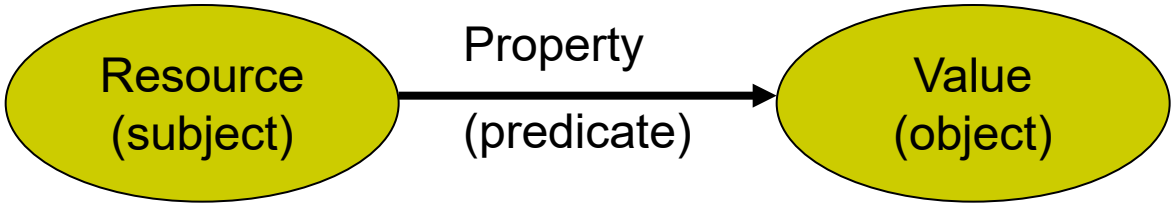
- RDF uses Web identifiers (URIs) to identify resources
- RDF describes resources with properties and property values
  - Everything can be represented as triples
- The essence of RDF is the (s,p,o) triple



Subject has a ***property*** with value "***object*** " (s,p,o)

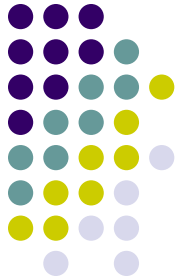
RDF Triples

- Triple
  - A Resource (Subject) is anything that can have a URI: URIs or blank nodes
  - A Property (Predicate) is one of the features of the Resource: URIs
  - A Property value (Object) is the value of a Property, which can be literal or another resource: URIs, literal, blank nodes



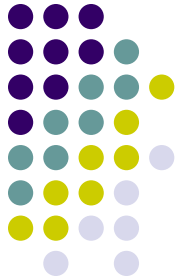
Literals can be the object of an RDF statement, but cannot be the subject or the predicate

RDF Data Model



- Any expression in RDF is a collection of triples (subject, predicate, object)
- A set of such triples is called an RDF graph
  - The nodes of an RDF graph are its subjects and objects
  - The direction of the arc is significant: it always points toward the object.
- A assertion of an RDF triple says the relationship (indicated by teh predicate) holds between subject and object.
- The meaning of an RDF graph is conjunction (AND) of the statements corresponding to all the triples it contains
- RDF does not provide means to express negation (NOT) or disjunction (OR)

RDF design goal



- Having a simple data model
- Have formal semantics and provable inference
- Using an extensible URI-based vocabulary
- Using an XML-based syntax
- Supporting use of XML Schema datatypes
- Allowing anyone to make statements about any resource

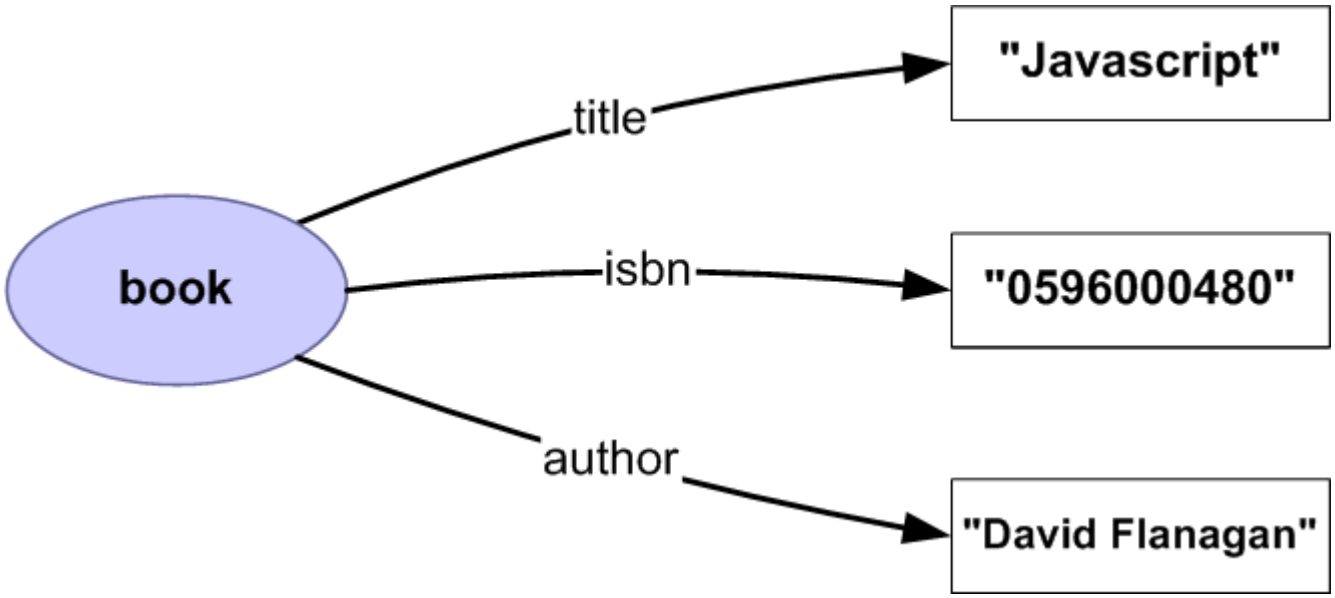
More properties

- What about showing more properties

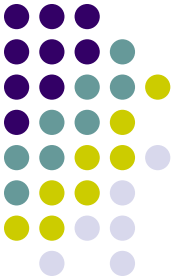
isbn	title	author	publisherID	pages
0596002637	Practical RDF	Shelley Powers	7642	350
0596000480	Javascript	David Flanagan	3556	936
. . .	. . .	. . .	. . .	. . .
. . .	. . .	. . .	. . .	. . .

Book table

More properties



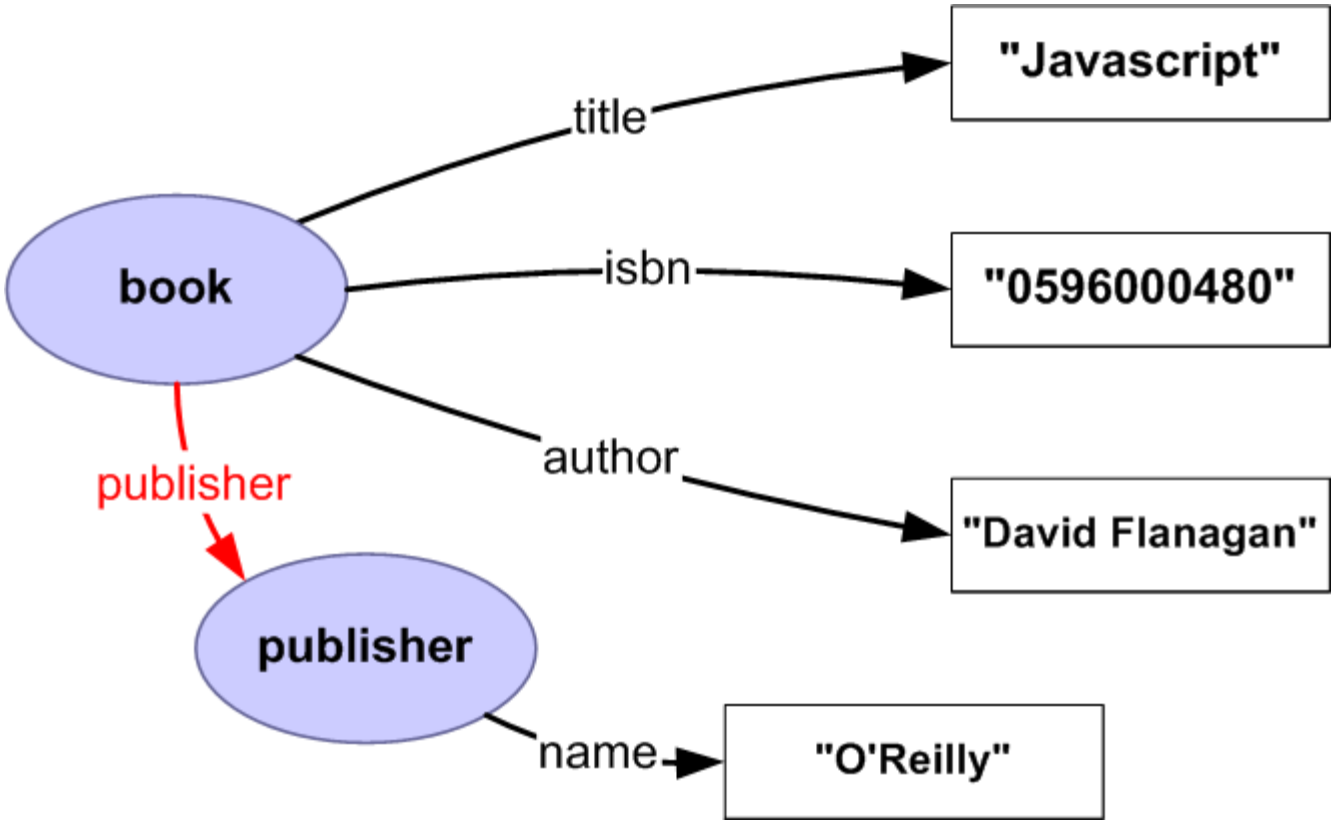
Relations between entities



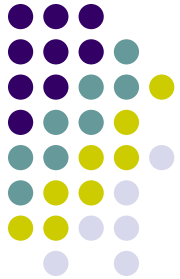
- Publishers table: publisherID is primary key and exists as foreign key in book table

publisherID	name	. . .	. . .
3556	O'Reilly	. . .	. . .
7311	Wrox	. . .	. . .
3209	Manning	. . .	. . .
. . .	. . .	. . .	. . .

Relations between entities

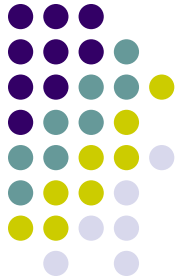


RDF is a graph



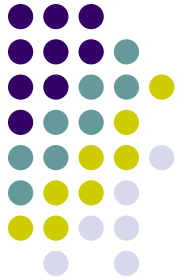
- An (s, p, o) triple can be viewed as a labeled graph
- The formal semantics of RDF is also described using graphs
- Think in terms of graphs, not XML or documents
- Nodes in graph are things (resources), arcs are relationship between things (resources)

Datatypes



- RDF takes XML Schema Datatypes, such as
  - Integer, date, number, …
- RDF predefines just one datatype:
  - rdf:XMLLiteral: for embedding XML in RDF
- RDF provides no mechanism for defining new datatypes
  - XML Schema Datatypes provides an extensible framework suitable for defining new datatypes for use in RDF.

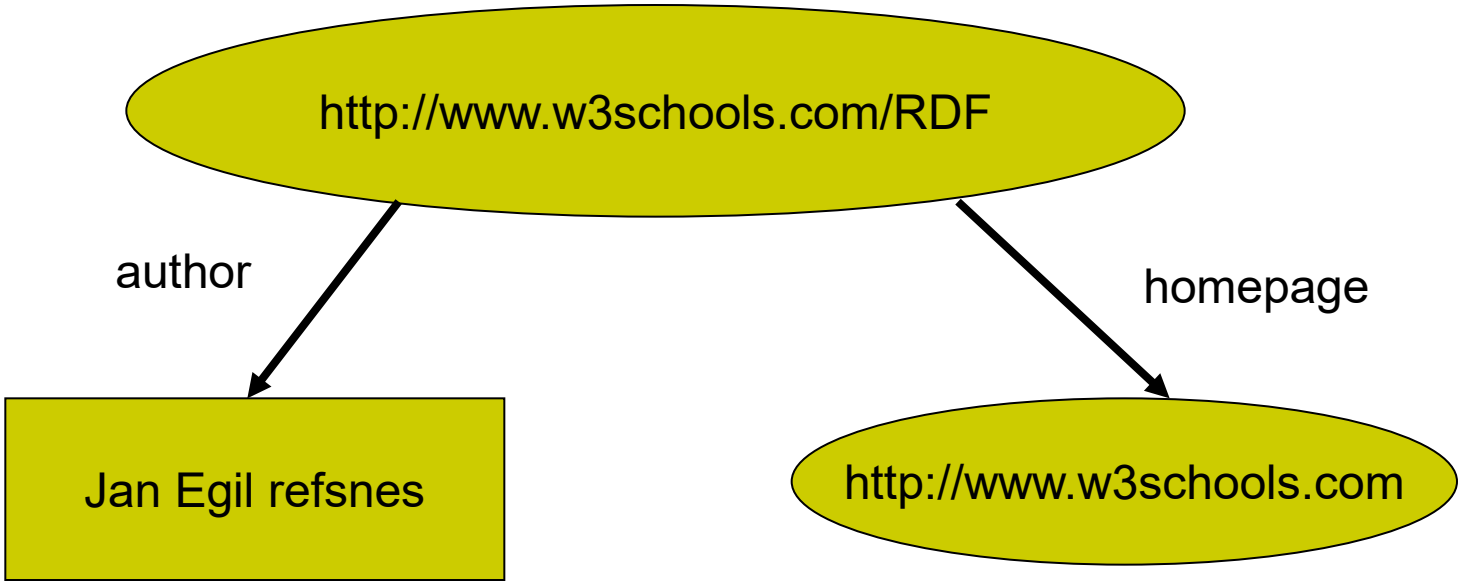
URI-based vocabulary and node identification



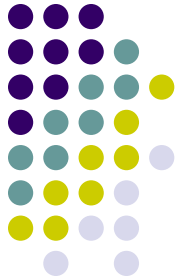
- A node can be
  - A node with URI
  - A literal, or
  - A blank node
- URI reference or literal can be used as node identifiers
- A blank node has no node identifier (has no name), but
  - Convention provides a way to use a blank node identifier to distinguish blank nodes from other nodes.
  - When merging different RDF graphs, different blank nodes need to be distinctly identified.

Simple example of RDF

```
<RDF>
  <Description about="http://www.w3schools.com/RDF">
    <author>Jan Egil refsnes</author>
    <homepage>http://www.w3schools.com</homepage>
  </Description>
</RDF>
```

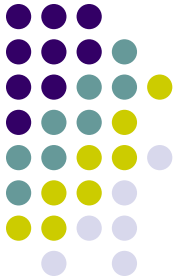


RDF Statements



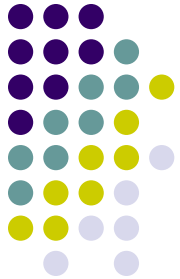
- The combination of a Resource, a Property, and a Property value forms a Statement
  - A Resource – the subject of a Statement
  - A Property – the predicate of a Statement
  - A Property value – the object of a Statement

Statement examples



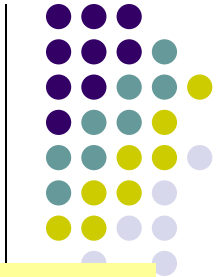
- Statement: “The author of http://www.w3schools.com/RDF is Jan Egil Refsnes”
  - Subject: http://www.w3schools.com/RDF
  - Predicate: author
  - Object: Jan Egil Refsnes
- Statement: “The homepage of http://www.w3schools.com/RDF is http://www.w3schools.com”
  - Subject: http://www.w3schools.com/RDF
  - Predicate: homepage
  - Object: http://www.w3schools.com

RDF syntax



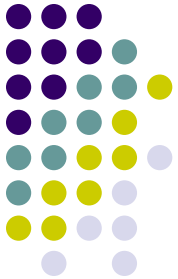
- Starting with <rdf:RDF> and end with </rdf:RDF>
- <rdf:Description> is the main element to define the subject, predicate and object of the statement
- RDF Namespace
  - http://www.w3.org/1999/02/22-rdf-syntax-ns#,
- File format: .rdf

RDF Example



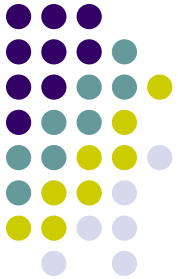
```
<?xml version="1.0" encoding="UTF-8" ?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cd="http://www.recshop.fake/cd#">

<rdf:Description
  rdf:about="http://www.rechshop.fake/cd/Empire Burlesque">
    <cd:artist>Bob Dylan</cd:artist>
    <cd:country>USA</cd:country>
    <cd:company>Columbia</cd:company>
    <cd:price>10.90</cd:price>
    <cd:year>1985</cd:year>
</rdf:Description>
<rdf:Description
  rdf:about="http://www.rechshop.fake/cd/Hide your heart">
    <cd:artist>Bonnie Tyler</cd:artist>
    <cd:country>UK</cd:country>
    <cd:company>CBS Records</cd:company>
    <cd:price>9.90</cd:price>
    <cd:year>1988</cd:year>
</rdf:Description>
<!-- more cds -->
</rdf:RDF>
```

RDF Validator



- It parses your RDF document, checks the syntax, and generate tabular and graphical views of your RDF document.
- W3C RDF Validation Service
  - http://www.w3.org/RDF/Validator/

RDF Validator



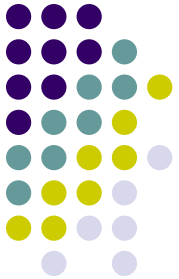
```
<?xml version="1.0" encoding="UTF-8" ?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:si="http://www.rechshop.fake/siteinfo#">

  <rdf:Description rdf:about="http://www.w3schools.com/RDF">
    <si:author>Jan Egil refsnes</si:author>
    <si:homepage>http://www.w3schools.com</si:homepage>
  </rdf:Description>

</rdf:RDF>
```

validation

RDF Validator

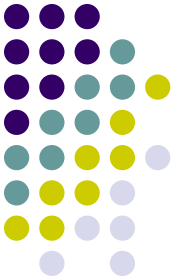
No	Subject	Predicate	Object
1	http://www.w3schools.com/RDF	http://www.rechshop.fake/siteinfo#author	"Jan Egil refsnes"
2	http://www.w3schools.com/RDF	http://www.rechshop.fake/siteinfo#homepage	"http://www.w3schools.com"

RDF main elements



- <rdf:RDF>: the root element
- <rdf:Description>: defining a resource

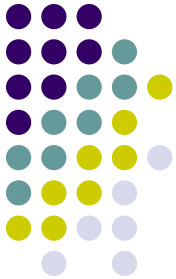
<rdf:RDF>



- It is the root element of an RDF document
- It declares the XML document to be an RDF document
- It contains a reference to the RDF namespace

```
<?xml version="1.0" encoding="UTF-8" ?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:si="http://www.rechshop.fake/siteinfo#">

  .  .  .

</rdf:RDF>
```

<rdf:Description>

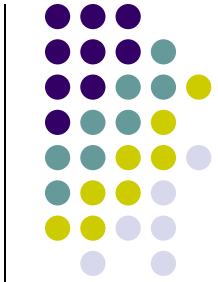


- It defines a resource using “about” attribute
- It contains elements that describing the resource (property, property values)

```
<?xml version="1.0" encoding="UTF-8" ?>
<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.rechshop.fake/cd/Empire Burlesque">
    <cd:artist>Bob Dylan</cd:artist>
    <cd:country>USA</cd:country>
    <cd:company>Columbia</cd:company>
    <cd:price>10.90</cd:price>
    <cd:year>1985</cd:year>
</rdf:Description>

</rdf:RDF>
```

Property is defined as element

rdf:about and rdf:ID



- Both are attribute for rdf:Description to represent the subject of the statement.
  - If the subject is URI, then use rdf:about
  - If the subject is literal, then use rdf:ID
  - If the subject is a blank node, then use rdf:nodeID

```
<rdf:Description
  rdf:about="http://www.rechshop.fake/cd/Empire Burlesque">
</rdf:Description>

<rdf:Description
  rdf:ID="Empire Burlesque">
</rdf:Description>

<rdf:Description rdf:nodeID="abc">
</rdf:Description>
```

rdf:about and rdf:ID

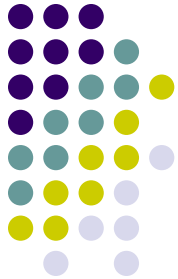
- You can use a relative URI in rdf:about and resolve it based on the base URI.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:dc="http://purl.org/dc/elements/1.1/"
xml:base="http://spam.com/eggs/" >

<rdf:Description rdf:about="listing.rdf#local-record">
<dc:title>Local Record</dc:title>
</rdf:Description>
</rdf:RDF>
```

The whole URI for local-record is:

http://spam.com/eggs/listing.rdf#local-record

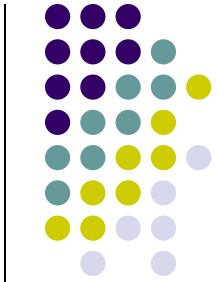
Properties as attributes



- Property elements can also be defined as attributes.

```
<?xml version="1.0" encoding="UTF-8" ?>
<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.rechshop.fake/cd/Empire Burlesque">
    cd:artist="Bob Dylan"
    cd:country="USA"
    cd:company="Columbia"
    cd:price="10.90"
    cd:year="1985"
/>

</rdf:RDF>
```

rdf:resource



- It is an attribute to indicate that the property element's value is another URI resource

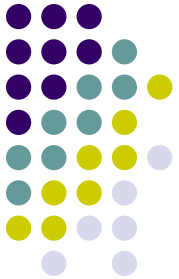
```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:exterms="http://www.example.org/terms/">

<rdf:Description rdf:about="http://www.example.org/index.html">
   <exterms:creation-date>August 16, 1999</exterms:creation-date>
</rdf:Description>

<rdf:Description rdf:about="http://www.example.org/index.html">
   <dc:language>en</dc:language>
</rdf:Description>

<rdf:Description rdf:about="http://www.example.org/index.html">
   <dc:creator rdf:resource="http://www.example.org/staffid/85740"/>
</rdf:Description>

</rdf:RDF>
```

Properties as Resources

- Property elements can also be defined as resources.

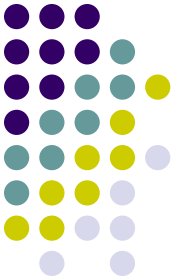
```
<?xml version="1.0" encoding="UTF-8" ?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cd="http://www.recshop.fake/cd#">

<rdf:Description
  rdf:about="http://www.rechshop.fake/cd/Empire Burlesque">
   <cd:artist rdf:resource="http://www.rechshop.fake/cd/dylan"/>
   <cd:country rdf:resource="http://www.rechshop.fake/cd/USA"/>
   <cd:company
        rdf:resource="http://www.rechshop.fake/cd/Columbia"/>
   <cd:price rdf:ID="10.90"/>
   <cd:year rdf:ID="1985"/>
</rdf:Description>

</rdf:RDF>
```

RDF Container Elements



- It is used to describe group of things
  - <Bag>: a list of members without order
  - <Seq>: a list of members with order
  - <Alt>: a list of members that only one can be selected
  - rdf:parseType="Collection": enumerates the specified members (the group only contains the specified members listed in the collection)

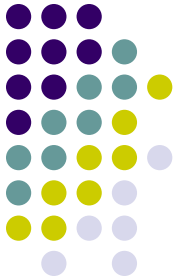
<rdf:Bag>



- It is used to describe a list of values without order
- It can contain duplicate values

```
<?xml version="1.0" encoding="UTF-8" ?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cd="http://www.rechshop.fake/cd#">
<rdf:Description rdf:about="http://www.rechshop.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>
```

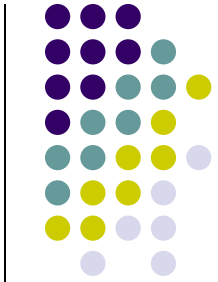
<rdf:Seq>



- It is used to describe a list of values with order
- It can contain duplicate values

```
<?xml version="1.0" encoding="UTF-8" ?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cd="http://www.rechshop.fake/cd#">
<rdf:Description rdf:about="http://www.rechshop.fake/cd/Beatles">
  <cd:artist>
    <rdf:Seq>
      <rdf:li>John</rdf:li>
      <rdf:li>Paul</rdf:li>
      <rdf:li>George</rdf:li>
      <rdf:li>Ringo</rdf:li>
    </rdf:Seq>
  </cd:artist>
</rdf:Description>
</rdf:RDF>
```

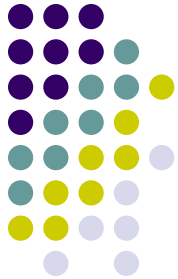
<rdf:Alt>



- It is used to describe a list of alternative values that the user can select only one of it

```
<?xml version="1.0" encoding="UTF-8" ?>
<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:format>
    <rdf:Alt>
      <rdf:li>CD</rdf:li>
      <rdf:li>Record</rdf:li>
      <rdf:li>Tape</rdf:li>
    </rdf:Alt>
  </cd:format>
</rdf:Description>

</rdf:RDF>
```

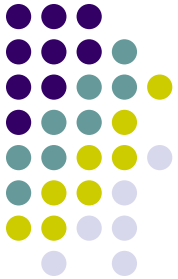
rdf:parseType=“Collection”



- It is used to describe group that contains ONLY the specified members
- It is described as the attribute rdf:parseType=“Collection”

```
<?xml version="1.0" encoding="UTF-8" ?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cd="http://www.rechshop.fake/cd#">
<rdf:Description rdf:about="http://www.rechshop.fake/cd/Beatles">
 <cd:artist rdf:parseType="Collection">
  <rdf:Description rdf:about="http://recshop.fake/cd/Beatles/George"/>
  <rdf:Description rdf:about="http://recshop.fake/cd/Beatles/John"/>
  <rdf:Description rdf:about="http://recshop.fake/cd/Beatles/Paul"/>
  <rdf:Description rdf:about="http://recshop.fake/cd/Beatles/Ringo"/>
 </cd:artist>
</rdf:Description>
</rdf:RDF>
```

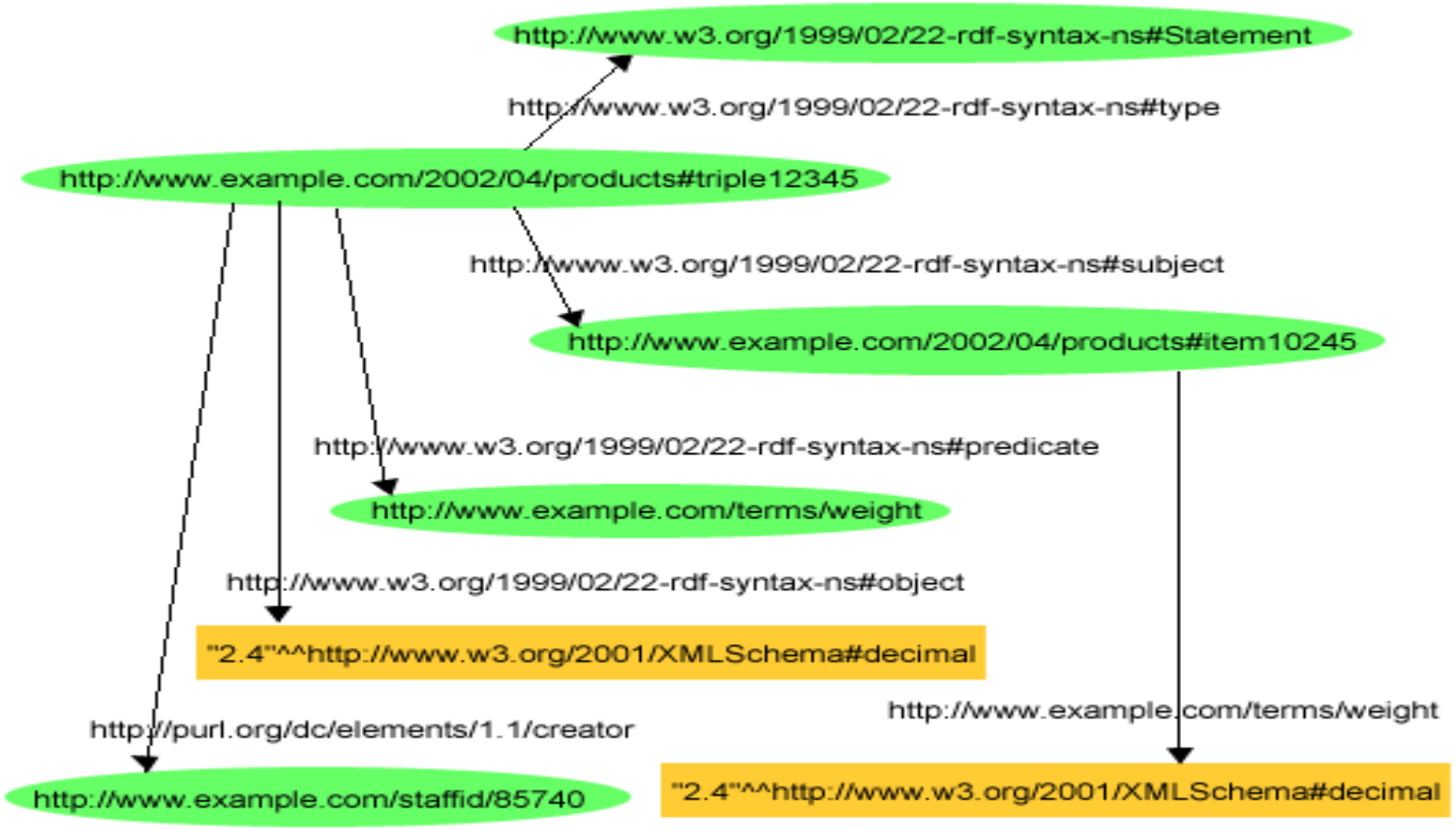
RDF Reification



- RDF provides a built-in vocabulary for describing RDF statements.
- A description of a statement using this vocabulary is called a reification of the statement
  - rdf:Statement, rdf:subject, rdf:predicate, rdf:object

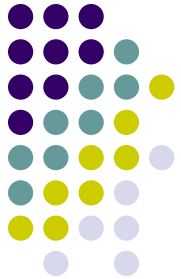
```
<rdf:Statement rdf:about="#triple12345">
  <rdf:subject
        rdf:resource="http://www.example.com/2002/04/products#item10
        245"/>
  <rdf:predicate
        rdf:resource="http://www.example.com/terms/weight"/>
  <rdf:object rdf:datatype="&xsd;decimal">2.4</rdf:object>

  <dc:creator rdf:resource="http://www.example.com/staffid/85740"/>

</rdf:Statement>
```

RDF Reification



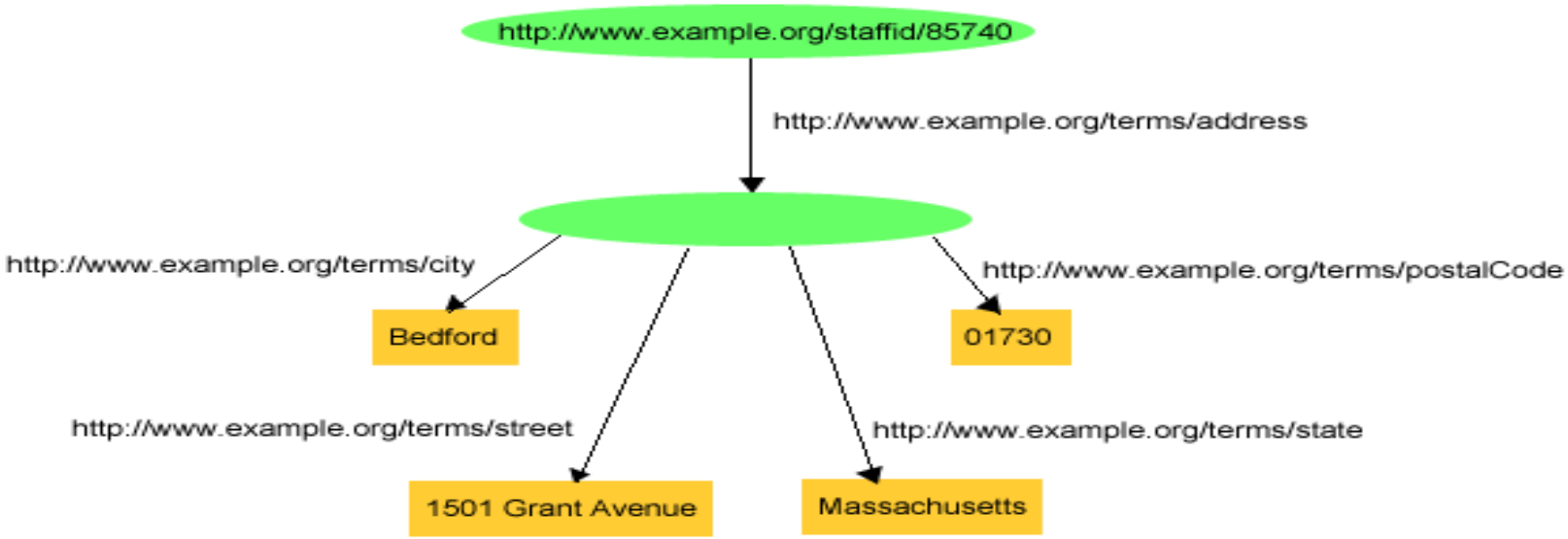

```
exproducts:triple12345 rdf:type rdf:Statement .
exproducts:triple12345 rdf:subject exproducts:item10245 .
exproducts:triple12345 rdf:predicate exterms:weight .
exproducts:triple12345 rdf:object "2.4"^^xsd:decimal .
exproducts:triple12345 dc:creator exstaff:85740 .
```

Blank node



- Sometimes a node without name or URI, it just provides the necessary connectivity between various parts of the graph.
- Blank node is called anonymous resource in RDF

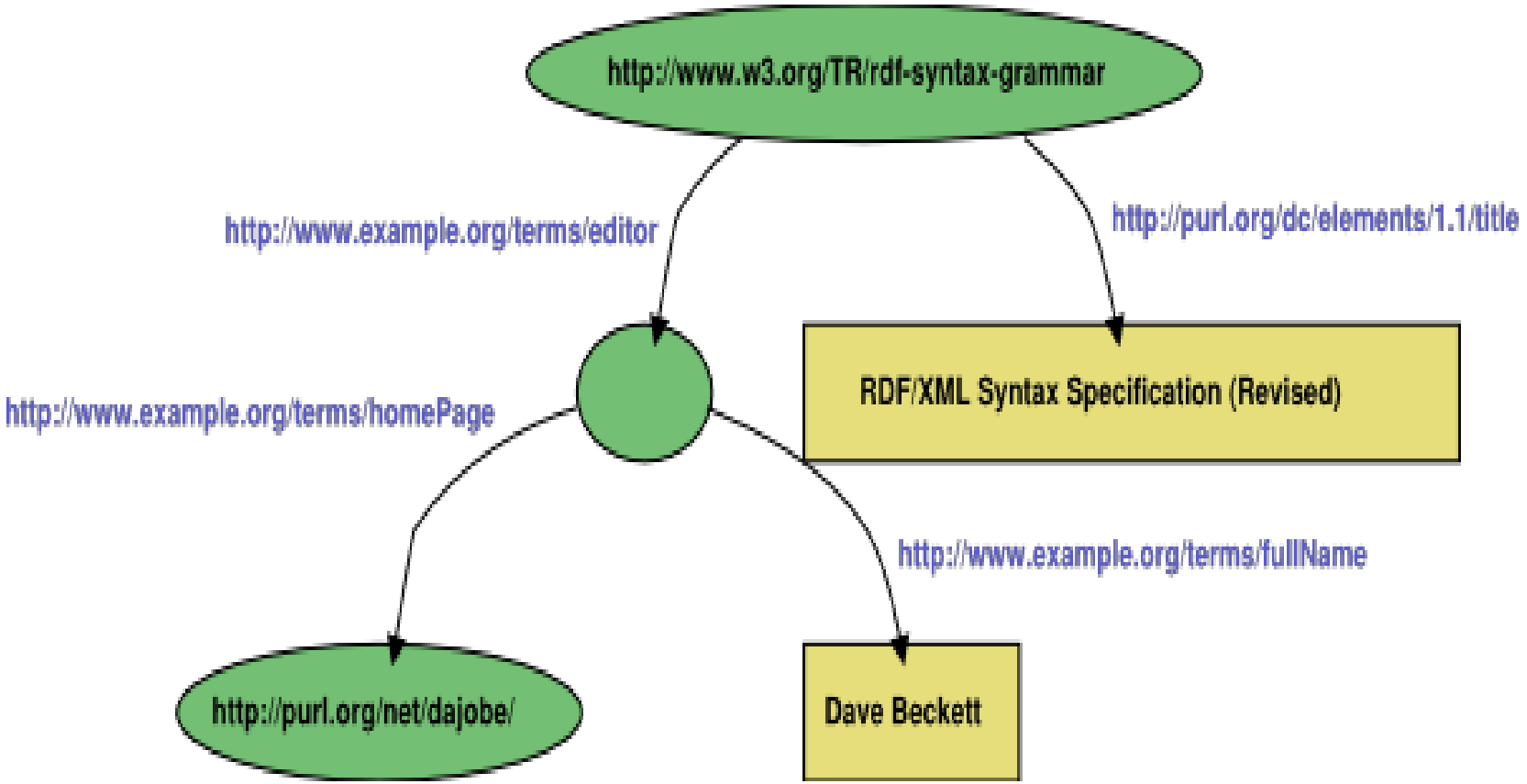
Blank Node



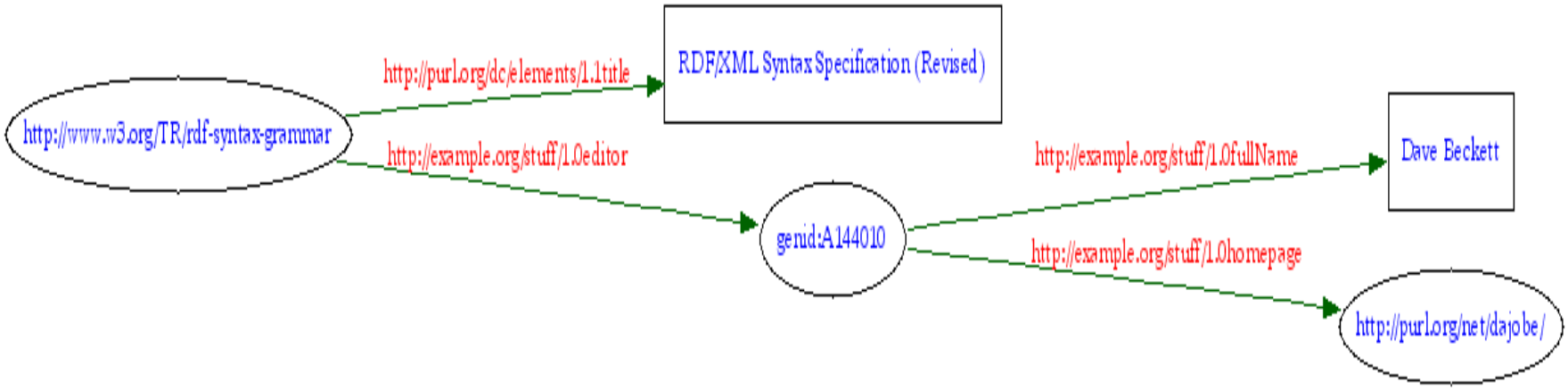
If we gives _:johnaddress as the identifier for the blank node

```
exstaff:85740 exterms:address _:johnaddress .
_:johnaddress exterms:street "1501 Grant Avenue" .
_:johnaddress exterms:city "Bedford" .
_:johnaddress exterms:state "Massachusetts" .
_:johnaddress exterms:postalCode "01730".
```

Blank node

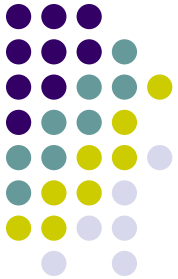

http://www.w3.org/TR/rdf-syntax-grammar
http://www.example.org/terms/editor
http://purl.org/dc/elements/1.1/title
RDF/XML Syntax Specification (Revised)
http://www.example.org/terms/homePage
http://www.example.org/terms/fullName
http://purl.org/net/dajobe/
Dave Beckett

Blank node



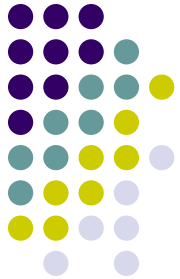
No	Subject	Predicate	Object
1	http://www.w3.org/TR/rdf-syntax-grammar	http://purl.org/dc/elements/1.1title	"RDF/XML Syntax Specification (Revised)"
2	**genid:A144010**	http://example.org/stuff/1.0fullName	"Dave Beckett"
3	http://www.w3.org/TR/rdf-syntax-grammar	http://example.org/stuff/1.0editor	**genid:A144010**
4	**genid:A144010**	http://example.org/stuff/1.0homepage	http://purl.org/net/dajobe/

Graph Rules



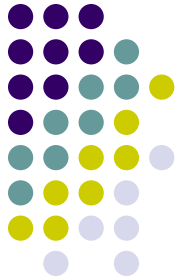
- Remember three types of node: URL, blank, or literal
- In a triple:
  - Subjects can be URLs or blank nodes
  - Predicates (properties) can be URLs
  - Objects (values) can be URLs, blank nodes or literals
- An RDF graph is a set of RDF triples
- The set of nodes of an RDF graph is the set of subjects and objects of triples

Graph Equivalence



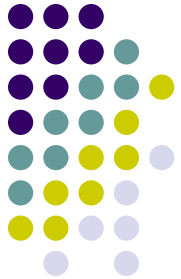
- The two RDF graph G and G’ are equivalent if:
  - Blank nodes in G map blank nodes in G’
  - Literal nodes in G map literal nodes in G’
  - URI nodes in G map URI nodes in G’
  - The triple (s,p,o) in G maps triple in G’

Writing down graphs

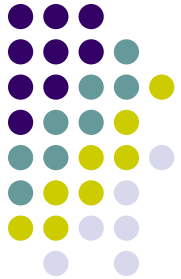


- A few options
  - Turtle: a text based format, easy to scribble, easy to read
  - RDF/XML: an XML based format, hard to read/write
  - Fundamental problem in XML: representing a graph with a tree

URI – Global Identification

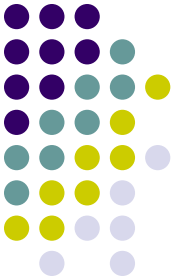


Global Naming



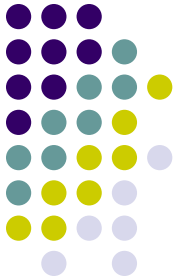
- For a web-scale database, need to be able to identify things globally and uniquely
- URLs already provide those capabilities
- RDF names things (resources) using URLs

Enough Names?



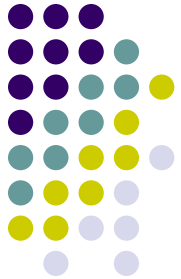
- There are an infinite number of possible URLs
- URLs name only one thing
- Create different URLs to name different things
- Do not use my names for your things
  - … unless you intend to refer to exactly the same thing

URI



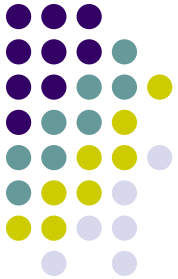
- Uniform Resource Identifier (URI) provides a general form of identifier for resources.
- It is not limited to identifying things that have network locations, or use other computer access mechanisms.
- URI can be created to refer to anything that needs to be referred to in a statement, such as:
  - Network-accessible things: electronic documents, online photo, service
  - Things that are not network-accessible: human beings, corporations, or books in library
  - Abstract concepts that do not physically exist: one concept

URI in RDF



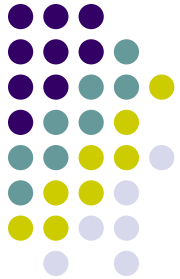
- RDF uses URIs (URI reference and fragment identifier) to identify the subjects, predicates, objects in statements
  - Fragment identifier: http://www.example.org/index.html#section2
- RDF defines a resource as anything that is identifiable by a URI

URLs for lookup



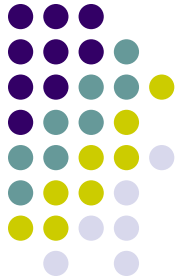
- This is the web – URLs can be retrieved
- Fetch any URL from a graph to lookup more RDF about it
- Which can be safely merged into the graph
- And may have other URLs which can be looked up
- What RDF might be provided at http://ex.org/thing/book
  - Title, author, isbn, publisher, …, anything

URI-based vocabulary



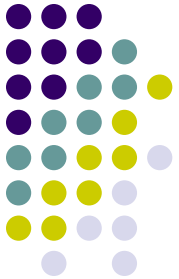
- RDF uses URI references to identify resources and properties
- A node can be a URI, a literal or blank
- A blank node means that a node is not a URI reference or a literal. It is a unique node that can be used in one or more RDF statements with no intrinsic name.
- RDF Triple
  - Subject (URI, blank node)
  - Predicate (URI)
  - Object (URI, literal, blank node)

URI plays a fundamental role



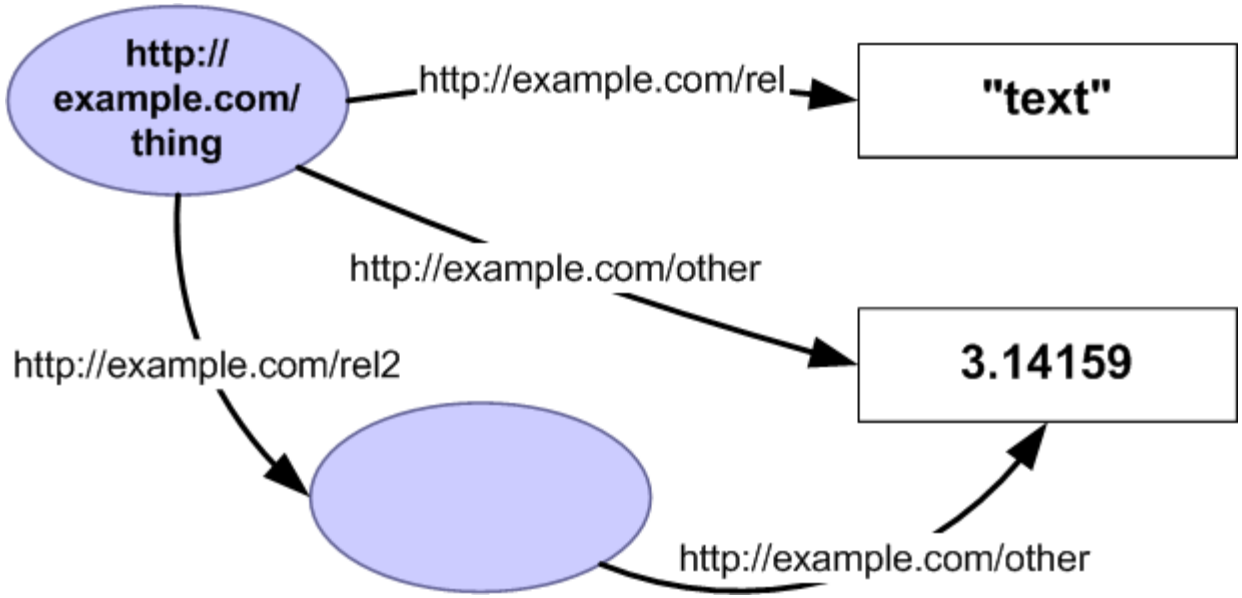
- URIs made the merge possible
- Semantics is added to existing Web resources via URIs
- URIs make it possible to link data with one another

Data Integration

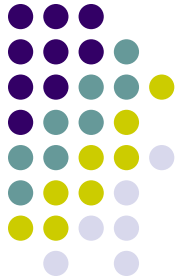


Graph Components

- Graphs can have
  - named things – resources
  - and named relations
  - and unnamed things which can also be related to other things

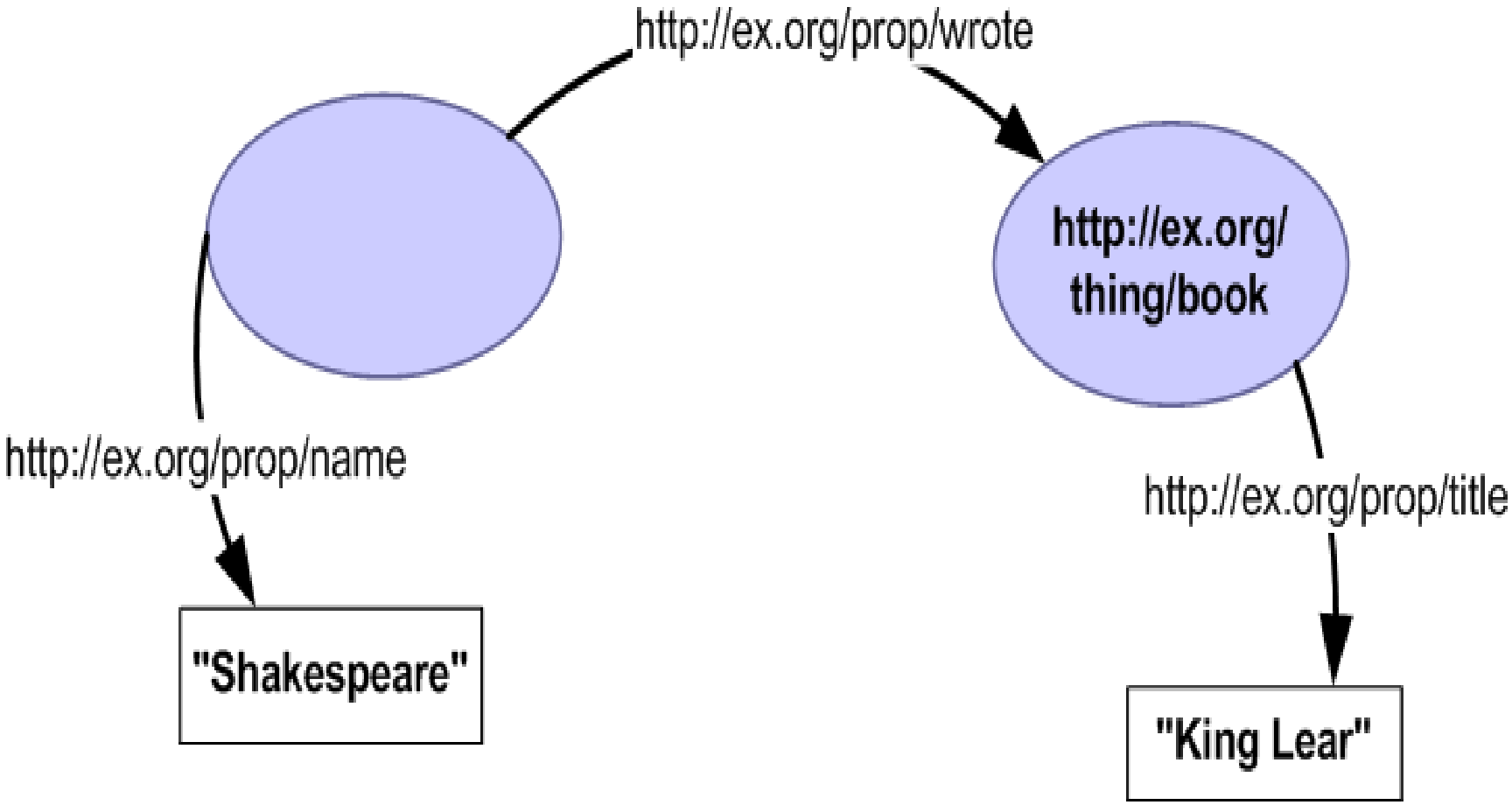


Merging

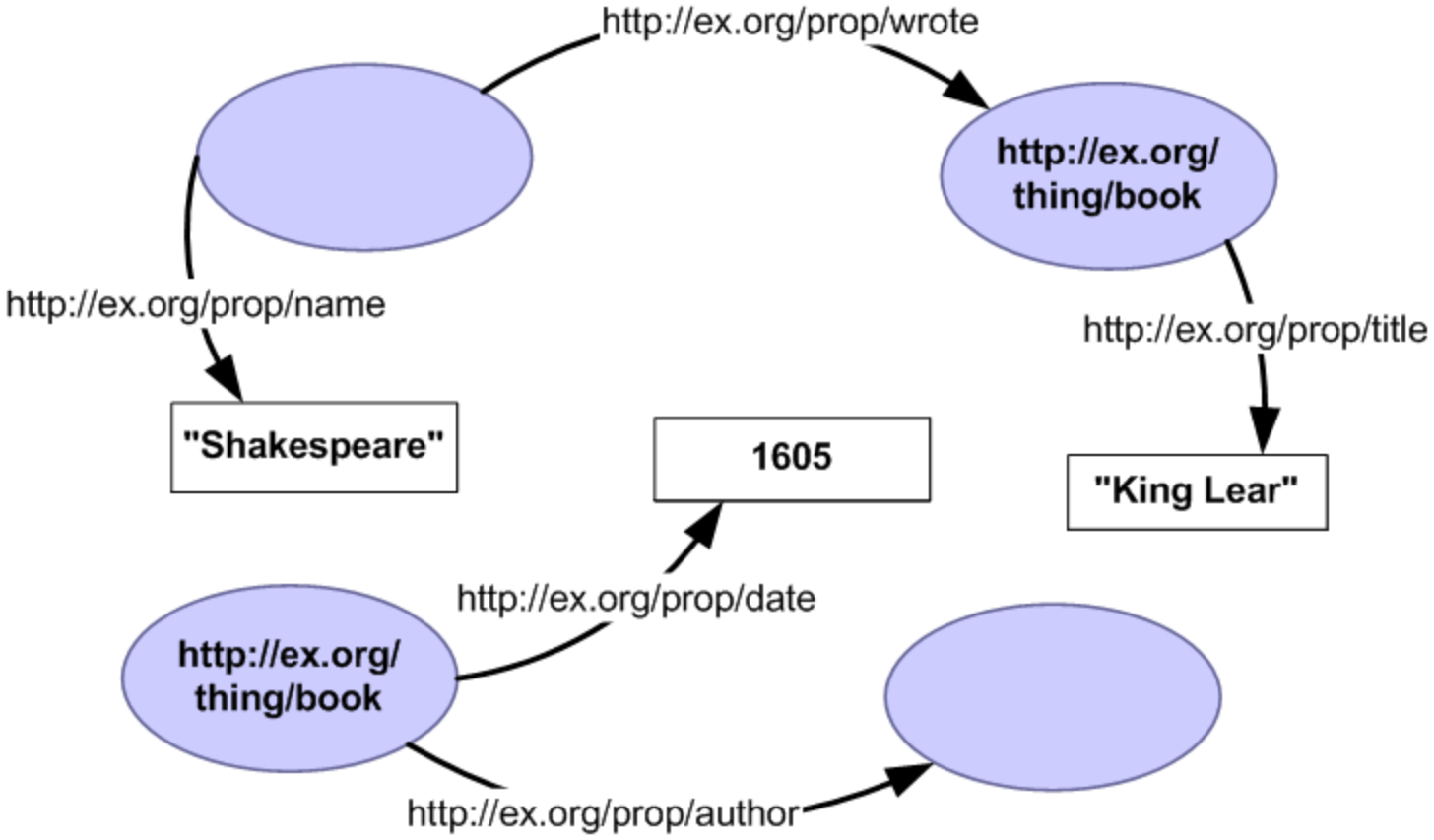


- Graphs from different sources can be merged
- Nodes with the same URL are considered identical
- Unlabelled nodes are kept separate
- No limitations on graphs that can be merged
- Any RDF can be merged with any other RDF

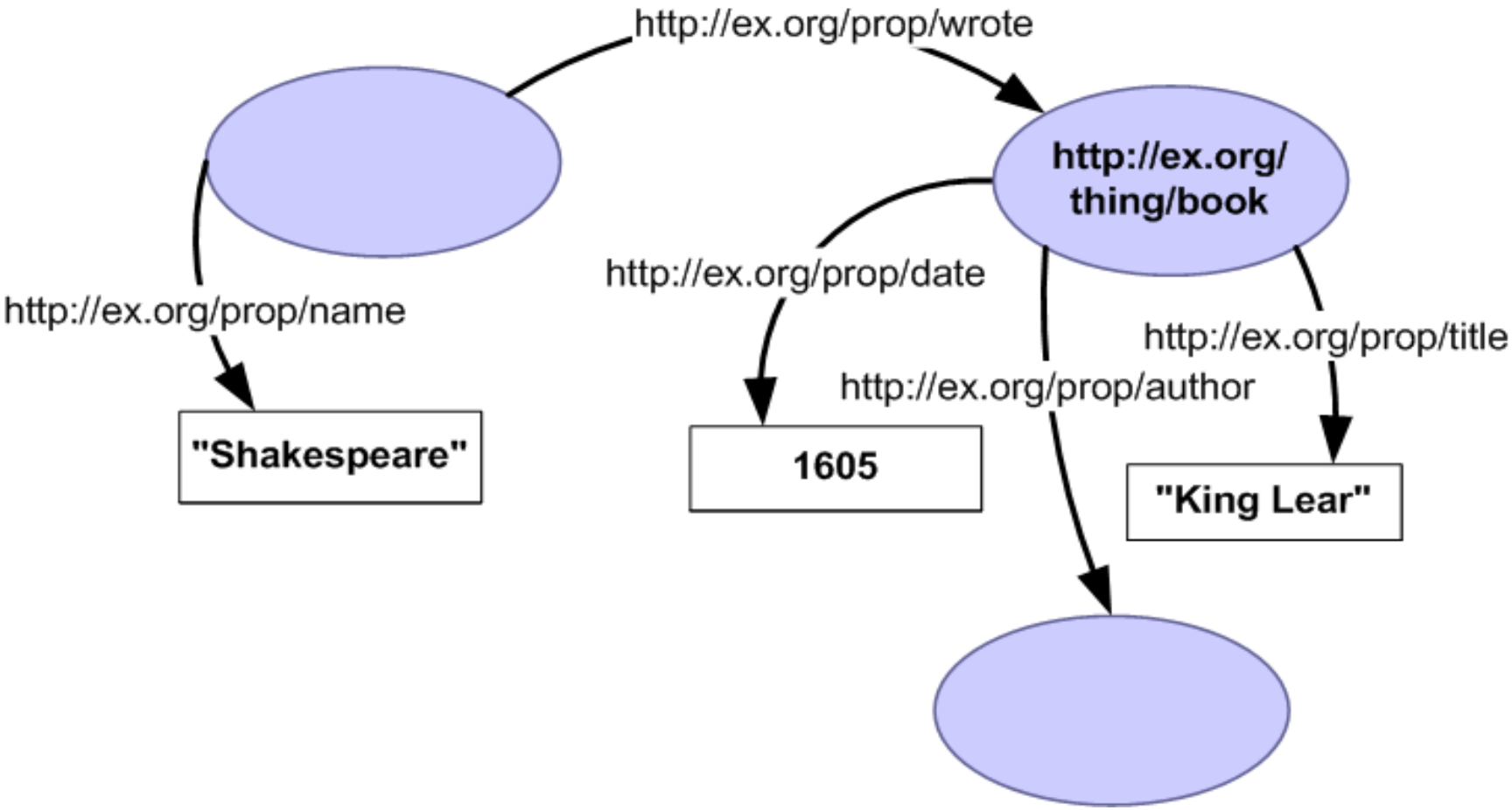
Merging


http://ex.org/prop/wrote
http://ex.org/
thing/book
http://ex.org/prop/name
http://ex.org/prop/title
"Shakespeare"
"King Lear"

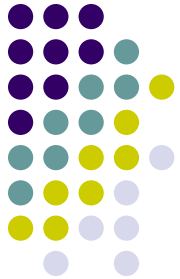
Merging


http://ex.org/prop/wrote
http://ex.org/
thing/book
http://ex.org/prop/name
http://ex.org/prop/title
"Shakespeare"
1605
"King Lear"
http://ex.org/prop/date
http://ex.org/
thing/book
http://ex.org/prop/author

Merging

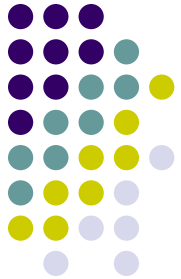

http://ex.org/prop/wrote
http://ex.org/
thing/book
http://ex.org/prop/name
http://ex.org/prop/date
http://ex.org/prop/title
http://ex.org/prop/author
"Shakespeare"
1605
"King Lear"

Merging is safe by design



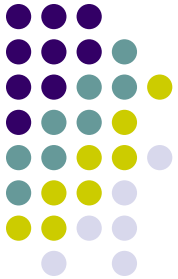
- Adding triples never changes the meaning of a graph
  - e.g. you cannot loose things by adding triples
  - You can not invalidate earlier conclusions
- Formally this means that RDF is monotonic
- It is a key part of the design for scalability

Smushing



- An informal term for “merging data based on knowledge of uniquely identifying properties”
- It is special type of merging
- Given IFPs it is possible to merge blank nodes as shown
- This means that it is possible to merge descriptions of things without needing to agree on URIs
- Which can be hard for things like people

Smushing

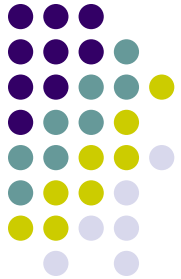


- FOAF defines the following IFPs
  - foaf:aimChatID, foaf:homepage, foaf:icqChatID, foaf:isPrimaryTopicOf, foaf:jabberID, foaf:mbox, foaf:msnChatID, foaf:yahooChatID
- So you can refer to people indirectly, e.g. Shakespeare

```
_:person
  foaf:isPrimaryTopicOf
  <http://en.wikipedia.org/wiki/Shakespeare> .
```

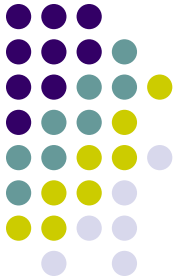
- This uniquely identifies the person as being the one that is the primary topic of that wikipedia page

RDF/XML



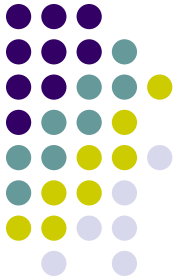
- RDF is expressed in XML – called RDF/XML
  - Subjects of triples are written as <rdf:Description rdf:about="URI">
  - Properties are written as child elements
  - Literal values are written as text content
  - URI value is written as rdf:resource="URI"

RDF/XML example



```
<?xml version="1.0" encoding="UTF-8" ?>

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:ex="http://example.com/schema#" >

<rdf:Description rdf:about="http://example.com/house">
  <ex:title>a house</ex:title>
  <ex:owner rdf:nodeID="owner"/>
</rdf:Description>

<rdf:Description rdf:nodeID="owner">
  <rdf:type rdf:resource="http://example.com/schema#Person"/>
  <ex:name>Fred</ex:name>
</rdf:Description>

</rdf:RDF>
```

RDF/XML example



- Without blank node labels:

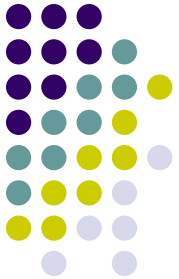
```
<?xml version="1.0" encoding="UTF-8" ?>

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-
ns#" xmlns:ex="http://example.com/schema#" >

<rdf:Description rdf:about="http://example.com/house">
  <ex:title>a house</ex:title>
  <ex:owner>
    <ex:Person>
      <ex:name>Fred</ex:name>
    </ex:Person>
  </ex:owner>
</rdf:Description>

</rdf:RDF>
```

RDF/XML example

- Blank node

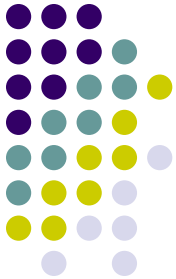
```
<?xml version="1.0" encoding="UTF-8" ?>

<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.w3.org/TR/rdf-syntax-grammar"
                   dc:title="RDF/XML Syntax Specification (Revised)">
    <ex:editor>
      <rdf:Description ex:fullName="Dave Beckett">
        <ex:homepage rdf:resource="http://purl.org/net/dajobe/" />
      </rdf:Description>
    </ex:editor>
  </rdf:Description>
</rdf:RDF>
```

Blank nodes require additional attention



- When merging
  - Blank nodes with identical nodeIDs in different graphs are different

Datatypes and languages



- Language use xml:lang, datatype use rdf:datatype

```
<?xml version="1.0" encoding="UTF-8" ?>

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:ex="http://example.com/schema#" >

<rdf:Description rdf:about="http://example.com/house">
  <ex:title xml:lang="en">a house</ex:title>
  <ex:owner>
    <ex:Person>
      <ex:name
        rdf:datatype="http://www.w3.org/2000/10/XMLSchema#string">
        Fred</ex:name>
    </ex:Person>
  </ex:owner>
</rdf:Description>
</rdf:RDF>
```

XML Literals

- RDF allows XML literals (XML content within an RDF graph) to be the object node of a predicate
- rdf:parseType="Literal"

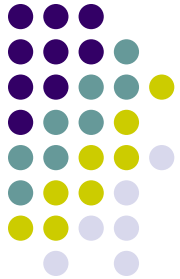
```
<?xml version="1.0" encoding="UTF-8" ?>

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:ex="http://example.org/stuff/1.0/">
  <rdf:Description rdf:about="http://example.org/item01">
    <ex:prop rdf:parseType="Literal"
             xmlns:a="http://example.org/a#"><a:Box required="true">
         <a:widget size="10" />
         <a:grommit id="23" /></a:Box>
    </ex:prop>
  </rdf:Description>
</rdf:RDF>
```

One triple: subject: http://example.org/item01;

predicate: http://example.org/stuff/1.0/prop;

object: "<a:Box xmlns:a="http://example.org/a#" required="true"> <a:widget size="10"></a:widget> <a:grommit id="23"></a:grommit></a:Box>"^^http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral

Typed Literals: rdf:datatype



- RDF allows typed literals to be the object node of a predicate
- Typed literals consist of a literal string and a datatype (XML datatype)

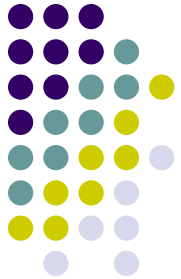
```
<?xml version="1.0" encoding="UTF-8" ?>

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:ex="http://example.org/stuff/1.0/">
  <rdf:Description rdf:about="http://example.org/item01">
    <ex:size rdf:datatype="http://www.w3.org/2001/XMLSchema#int">123</ex:size>
  </rdf:Description>
</rdf:RDF>
```

Subject: http://example.org/item01

Predicate: http://example.org/stuff/1.0/size

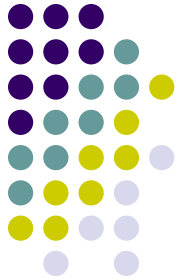
Object: "123“^^http://www.w3.org/2001/XMLSchema#int

Scaling the Web



- TBL made simplifications in the web's design to enable huge scalability
- Pre-web: all links were two-way
- Simplification: back-links are optional; most links work, some break
- Benefits: scalability; can link from my site to google without a link back

Partial Understanding



- In RDF, analogous simplification is that of partial understanding:
- Pre-web: require all data to be available before making decisions
- Simplification: enable partial understanding via monotonic property
- Benefits: scalability; can process data as it becomes available
- “missing is not broken”

References

- RDF tutorial
  http://research.talis.com/2005/rdf-intro/
- W3C RDF
  http://www.w3.org/RDF/
- RDF Resource Guide
  http://planetrdf.com/guide/