The RDF Data Model

Werner Nutt

Acknowledgment

These slides are based on the slide set

- RDF
  By Mariano Rodriguez (see http://www.slideshare.net/marianomx)

- History and Motivation
- Naming: URIs, IRIs, Qnames
- RDF Data Model: Triples, Literals, Types
- Modeling with RDF: BNodes, n-ary Relations, Reification
- Containers

- History and Motivation
- Naming: URIs, IRIs, Qnames
- RDF Data Model: Triples, Literals, Types
- Modeling with RDF: BNodes, n-ary Relations, Reification
- Containers

RDF stands for …

***R**esource **D**escription **F**ramework*

History

- RDF originated as a format for structuring metadata about Web sites, pages, etc.
  - Page author, creator, publisher, editor, …
  - Data about them: email, phone, job, …
- First version in W3C Recommendation of 1999
  - specified serialization in XML
- Metadata = Data ➔ RDF is a general data format
- Berners-Lee, Hendler, and Lassila proposed RDF as the model for data exchange on the Semantic Web

  (see their paper in Scientific American, 2001)

RDF is…

… the data model of Semantic Technologies and of the Semantic Web

Two Views of RDF

- Intuitively, an RDF data set is a

  labeled, directed graph

  ➔ *what are the nodes? and what are the edge labels?*

- Technically, an RDF data contains

  triples

  of the form

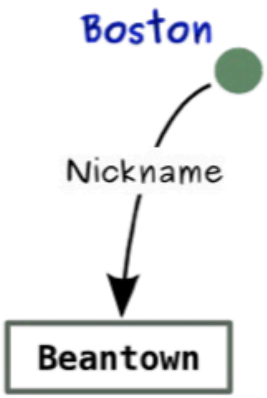
  Subject Predicate Object .

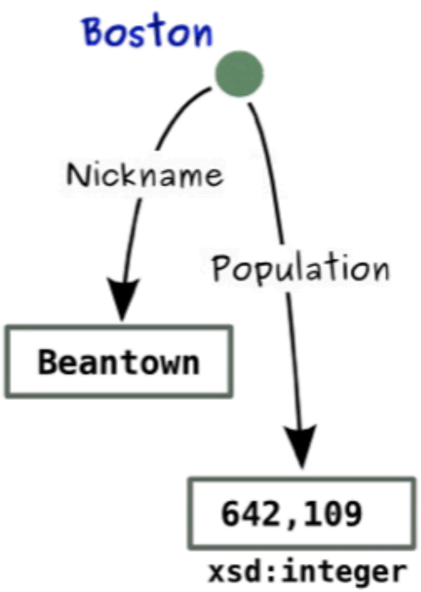
  ➔ *what are subjects, predicates, and objects?*

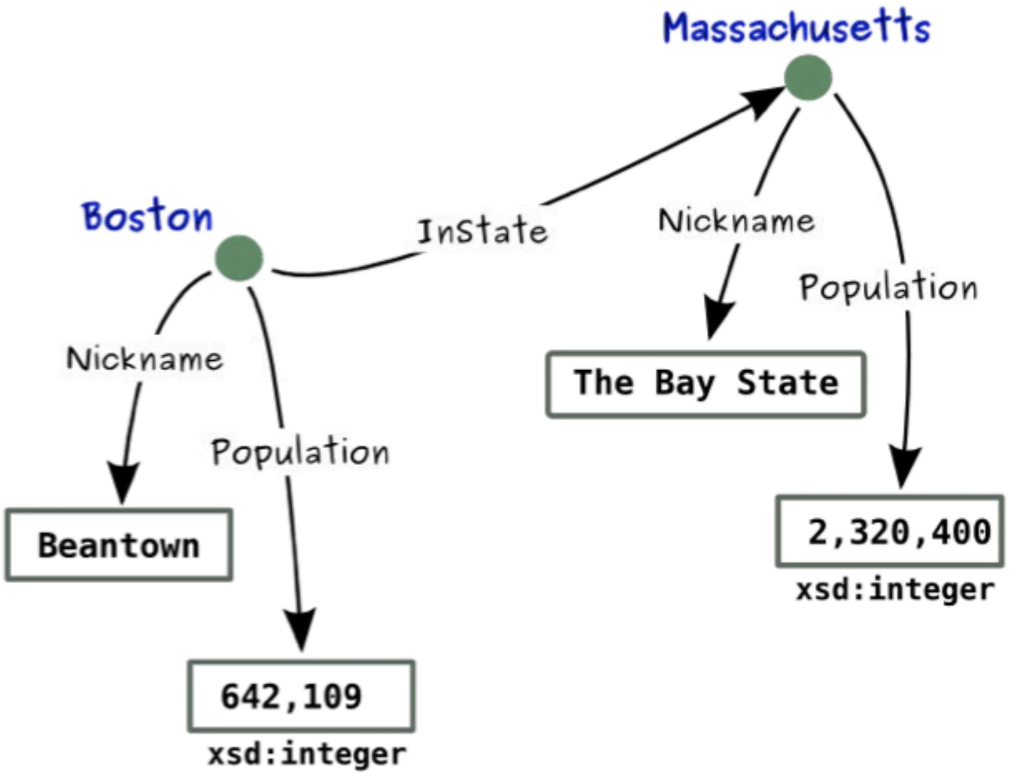
Example

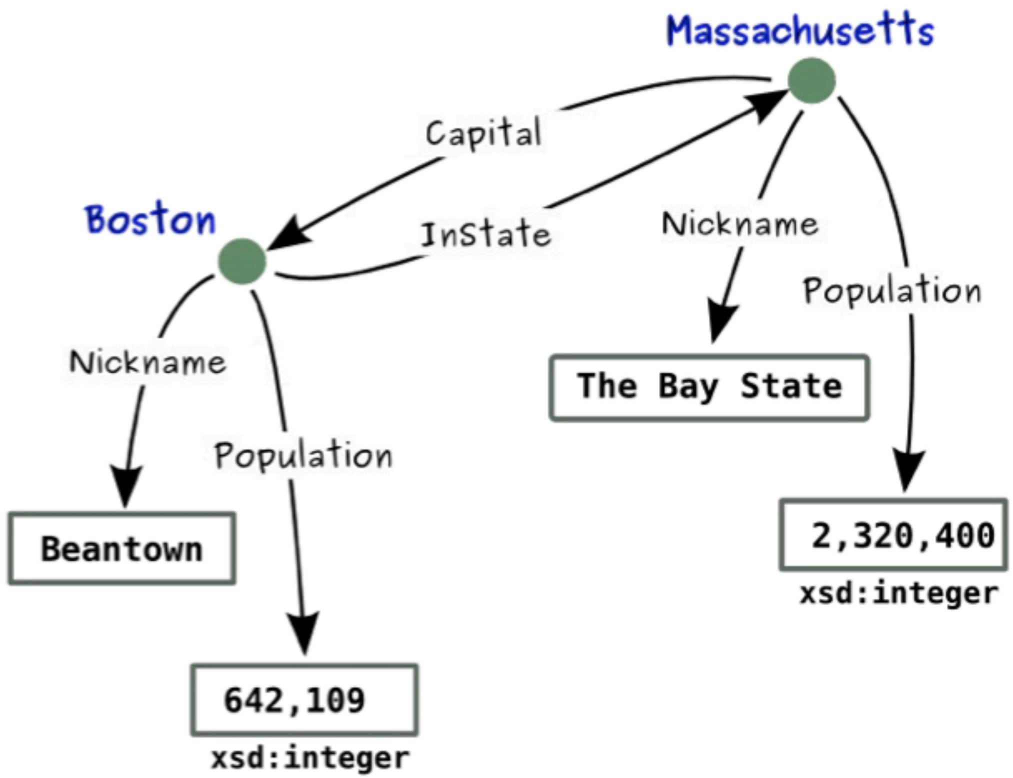
- Info about Boston, People, ISWC 2010, …

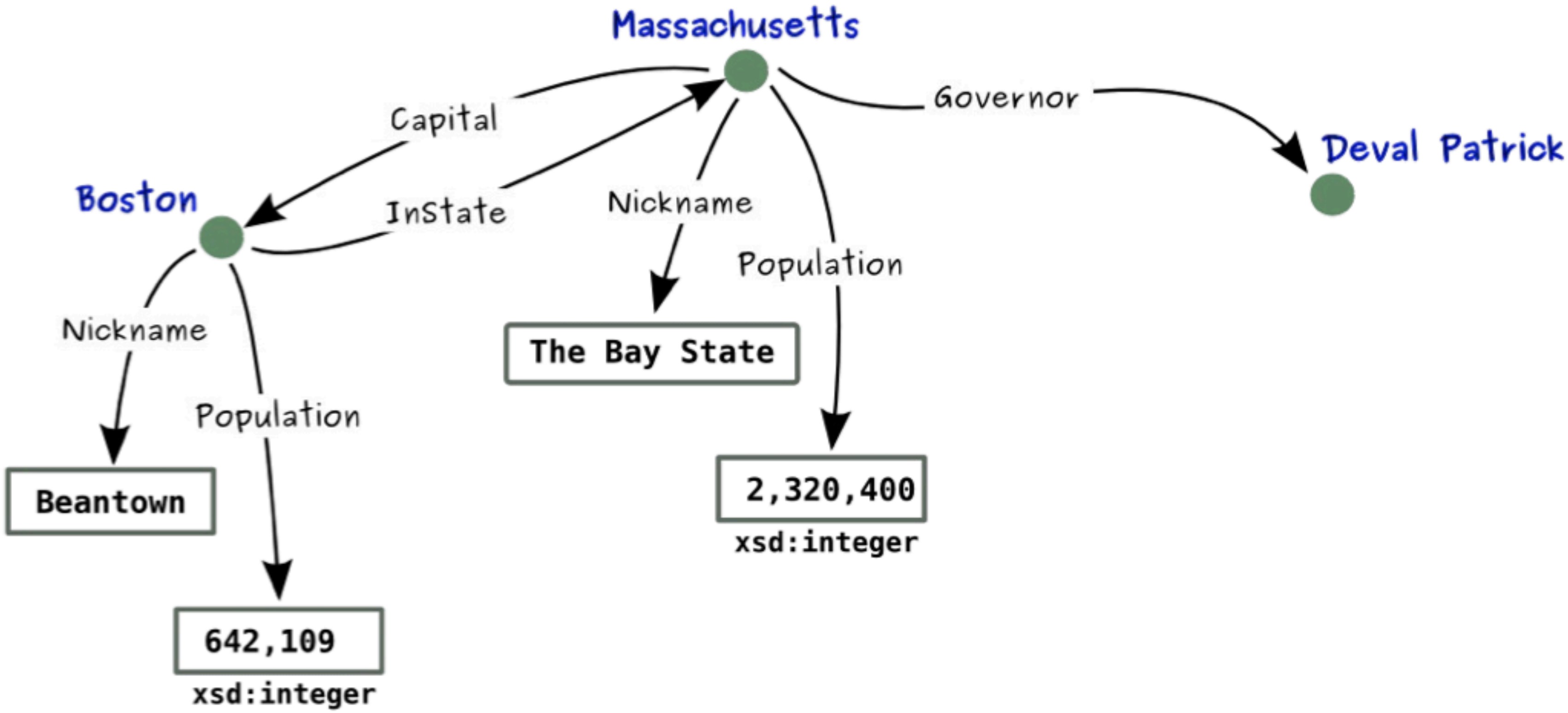
[taken from the tutorial of Sandro Hawke on RDF at ISWC 2010, which took place in …]

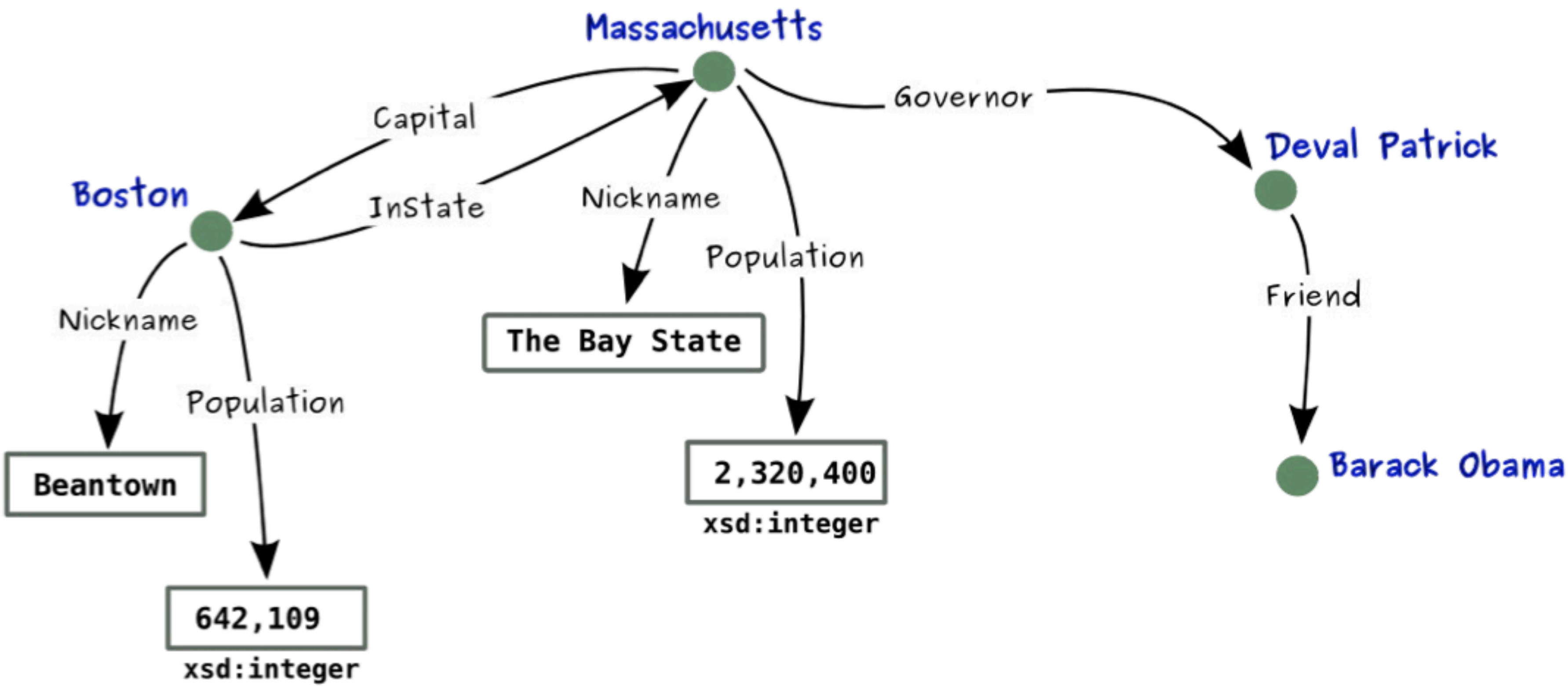

Boston
Nickname
Beantown

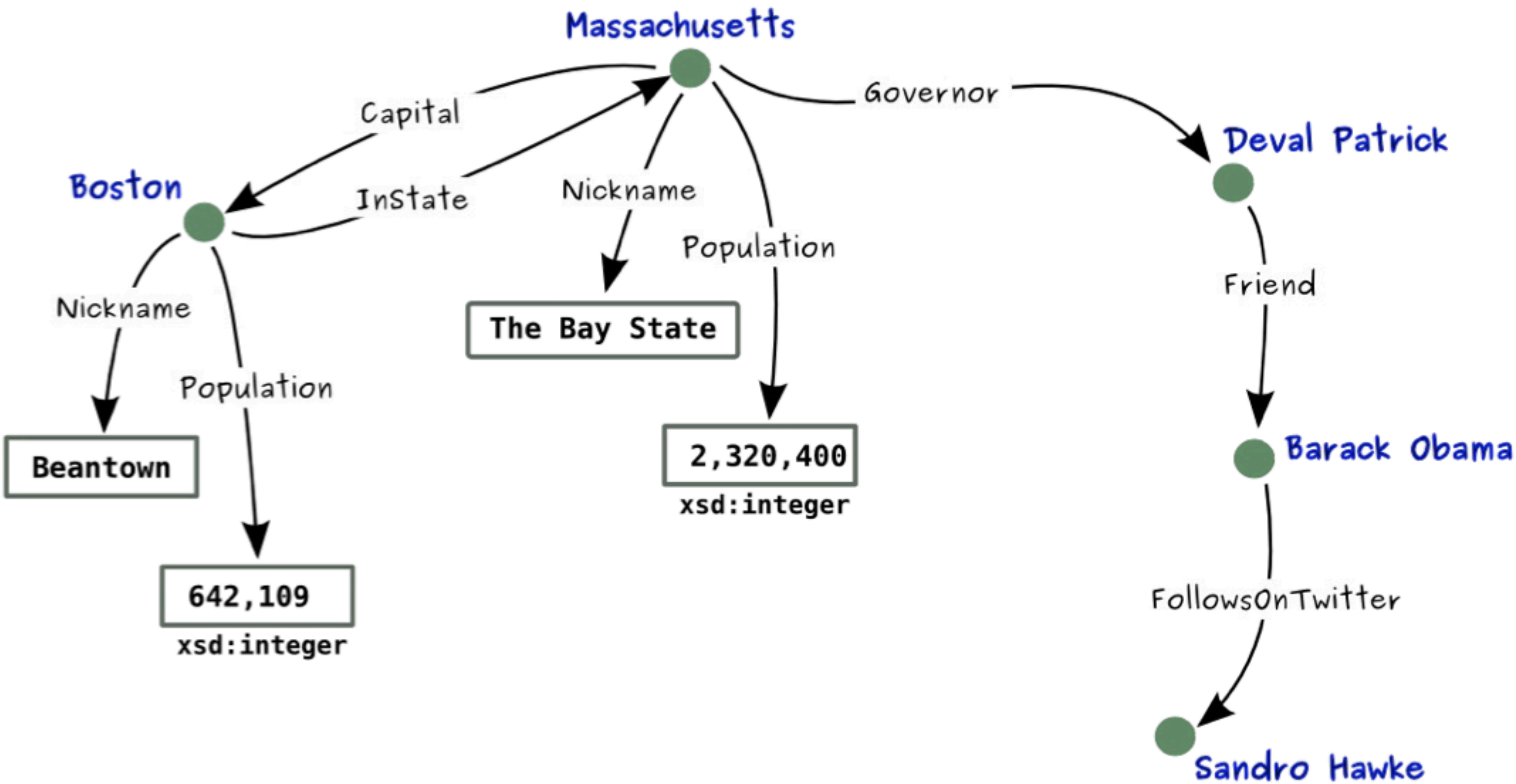

Boston
Nickname
Population
Beantown
642,109
xsd:integer


Massachusetts
Boston
InState
Nickname
Population
The Bay State
Nickname
Population
Beantown
2,320,400
xsd:integer
642,109
xsd:integer

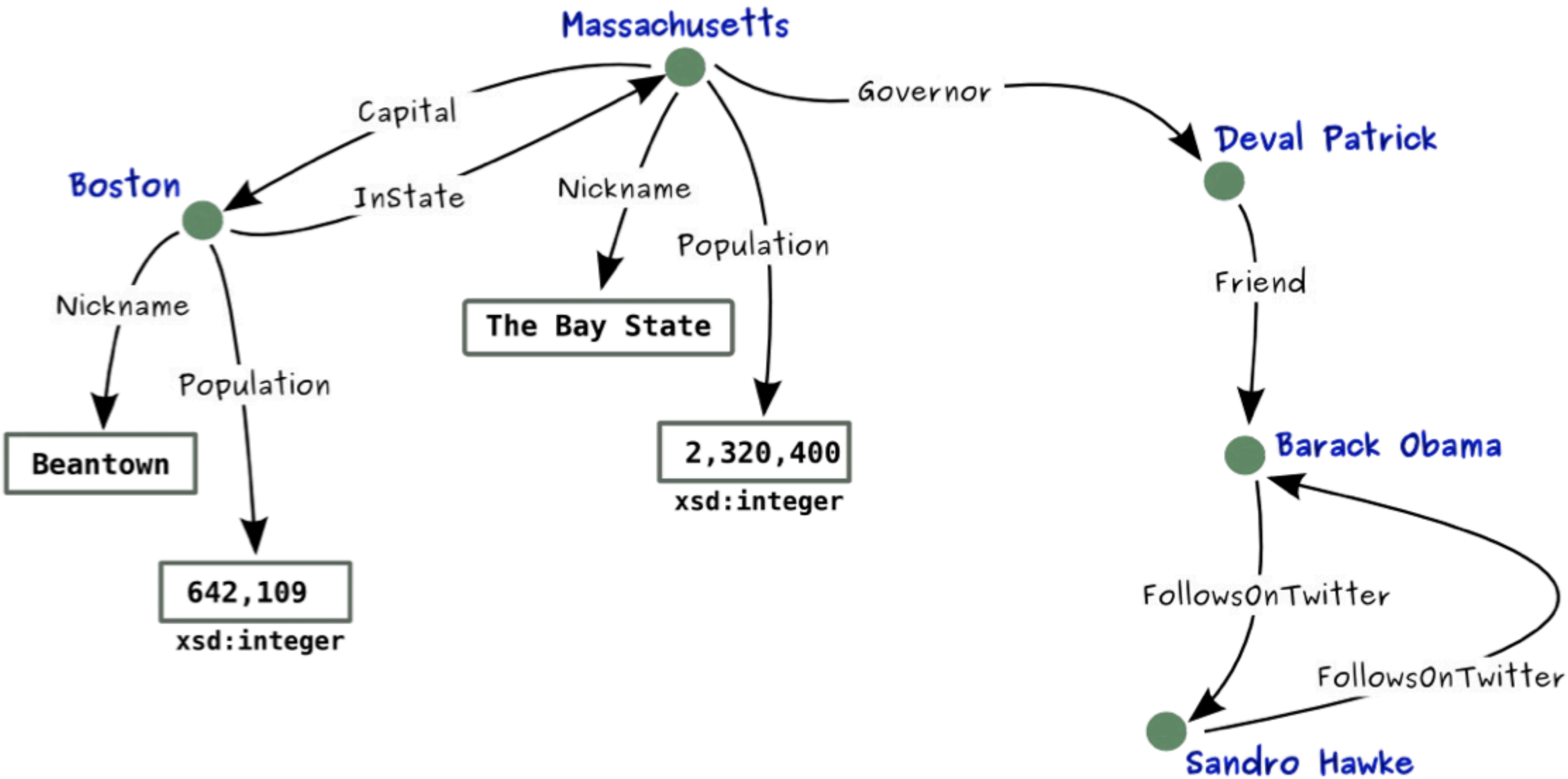

Massachusetts
Boston
Capital
InState
Nickname
Population
Nickname
Population
The Bay State
Beantown
2,320,400
xsd:integer
642,109
xsd:integer

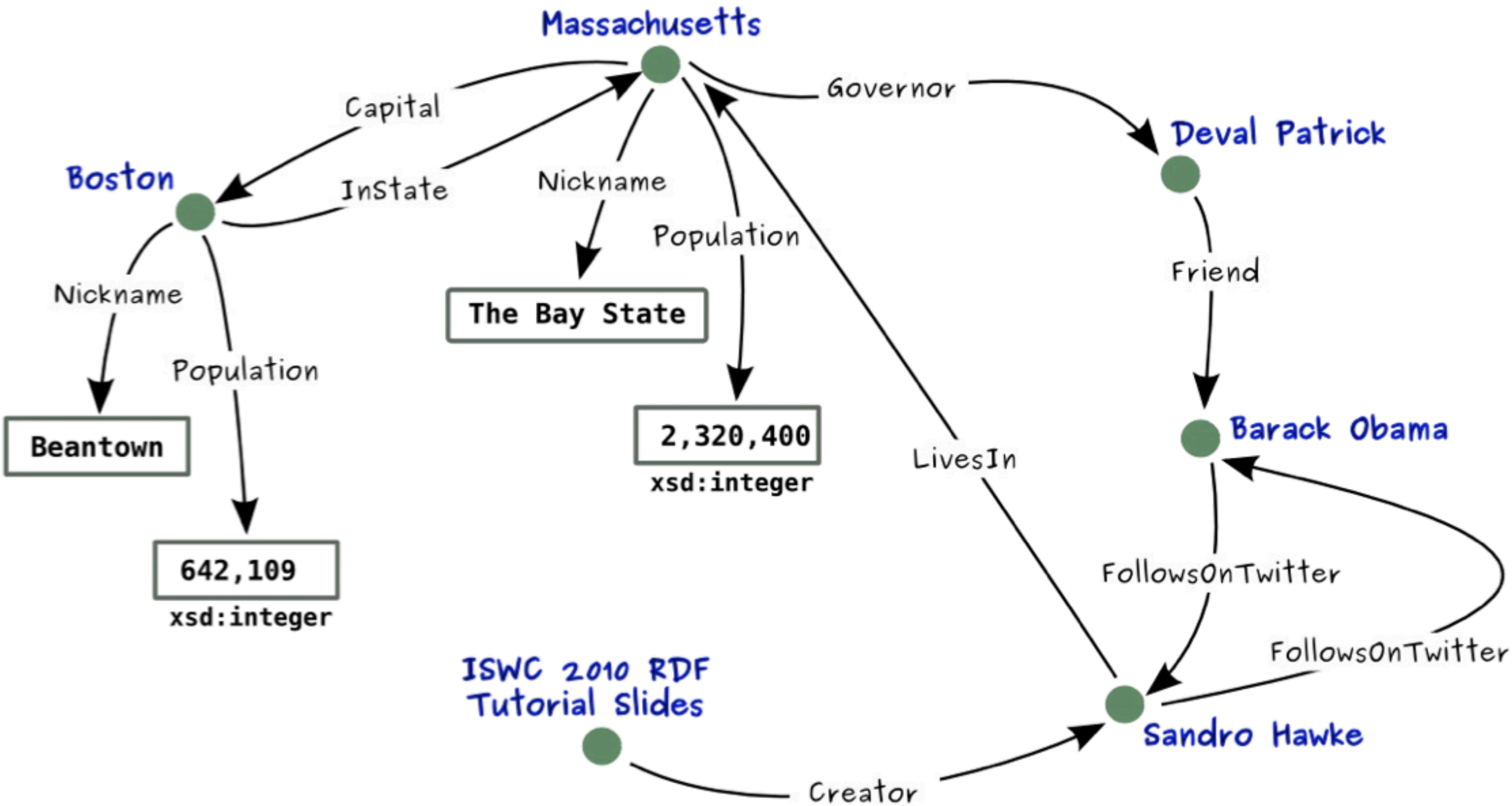

Massachusetts
Capital
Governor
Deval Patrick
Boston
InState
Nickname
Population
Nickname
Population
The Bay State
Beantown
2,320,400
xsd:integer
642,109
xsd:integer

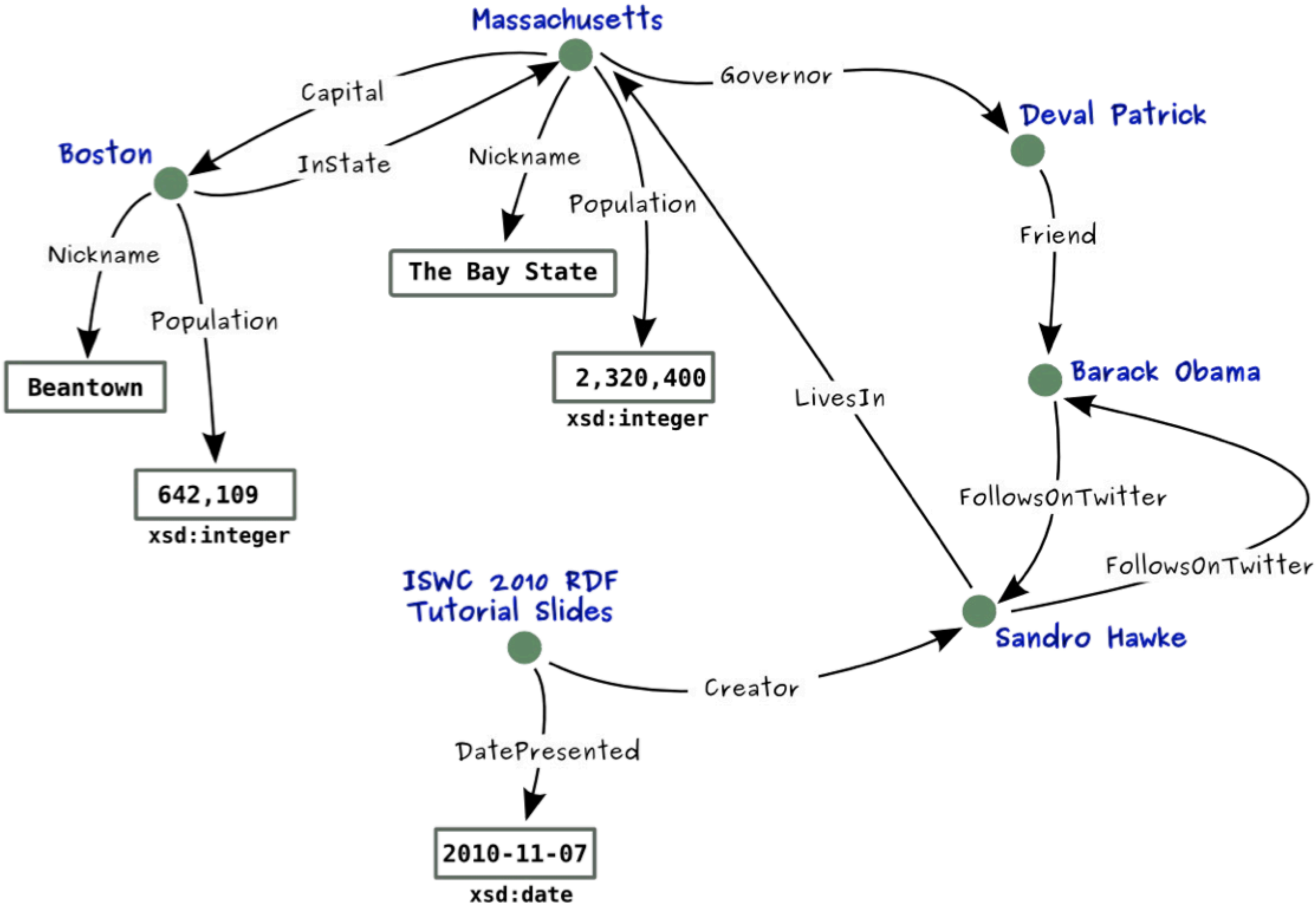

Massachusetts
Capital
Governor
Deval Patrick
Boston
InState
Nickname
Population
Friend
Nickname
The Bay State
Population
Beantown
2,320,400
xsd:integer
Barack Obama
642,109
xsd:integer



Massachusetts
Capital
Governor
Deval Patrick
Boston
InState
Nickname
Population
Nickname
Friend
The Bay State
Population
Beantown
Barack Obama
2,320,400
xsd:integer
642,109
xsd:integer
FollowsOnTwitter
Sandro Hawke


Massachusetts
Capital
InState
Nickname
Population
Governor
Deval Patrick
Boston
Nickname
Population
The Bay State
Beantown
642,109
xsd:integer
2,320,400
xsd:integer
Friend
Barack Obama
FollowsOnTwitter
FollowsOnTwitter
Sandro Hawke


Massachusetts
Capital
InState
Governor
Deval Patrick
Boston
Nickname
Population
Friend
The Bay State
Nickname
Population
Beantown
2,320,400
xsd:integer
Barack Obama
LivesIn
642,109
xsd:integer
FollowsOnTwitter
FollowsOnTwitter
ISWC 2010 RDF
Tutorial Slides
Sandro Hawke
Creator



Massachusetts
Capital
InState
Governor
Deval Patrick
Boston
Nickname
Population
The Bay State
Nickname
Friend
Beantown
Population
2,320,400
xsd:integer
Barack Obama
LivesIn
642,109
xsd:integer
FollowsOnTwitter
FollowsOnTwitter
ISWC 2010 RDF
Tutorial Slides
Sandro Hawke
Creator
DatePresented
2010-11-07
xsd:date

- History and Motivation
- Naming: URIs, IRIs, Qnames
- RDF Data Model: Triples, Literals, Types
- Modeling with RDF: BNodes, n-ary Relations, Reification
- Containers

Unambiguous Names

- How many things are named "Boston"?
  How about "Riverside"?

- What is meant by "Nickname"?
  And what by "LivesIn"?

➔ We need unambiguous identifiers

On the Web, we have URLs, URIs, and IRIs

URLs, URIs and IRIs

- We know basic Web addresses
  - http://google.com
  - https://gmail.com
  - http://www.inf/unibz.it/
  - http://www.inf/unibz.it/~nutt/

- **URL** (= Uniform Resource Locator)
  Web address of an information resource
  (Web page, image, zip file, …)

URLs, URIs and IRIs (cntd)

- **URI** (= Uniform Resource Identifier)
  In most cases, looks like a URL, but might identify something else (person, place, concept)
  - Every URL is also a URI, but not vice versa
  - Was known as URN (= Uniform Resource Names)
  - Supports ISBN numbers (e.g., urn:isbn:0-486-27557-4)
- **IRI** (= Internationalized Resource Identifier)
  - Uses Unicode instead of ASCII
    (e.g., http://ヒキワリ.ナットウ.ニホン)
  - Every IRI can be turned into a URI (%-encoding)

URLs ⊆ URIs ⊆ IRIs

URI, URL and IRI: Syntax

scheme:[//authority]path[?query][#fragment]

- scheme: type of URI, e.g. http, ftp, mailto, file, irc
- authority: typically a domain name
- path: e.g. /etc/passwd/
- query: optional; provides non-hierarchical information. Usually for parameters, e.g. for a web service
- fragment: optional; often used to address part of a retrieved resource, e.g. section of a HTML file.

IRI design is important for semantic applications. More later.

The URI Quiz

Using URIs to identify things

① ensures that there are not two different names (URIs) for the same thing

② ensures that one name (URI) is not used for two different things

③ makes it easier to avoid using one name (URI) for two different things

What is correct?

QNames

- Used in RDF as shorthand for long URIs:
  If
    prefix foo is bound to http://example.com/
  then
    foo:bar
  expands to
    http://example.com/bar
- Not quite the same as XML namespaces.
- Practically relevant due to syntactic restrictions on formats for data exchange (in particular, XML)

Necessary to fit any example on a page!
Simple string concatenation

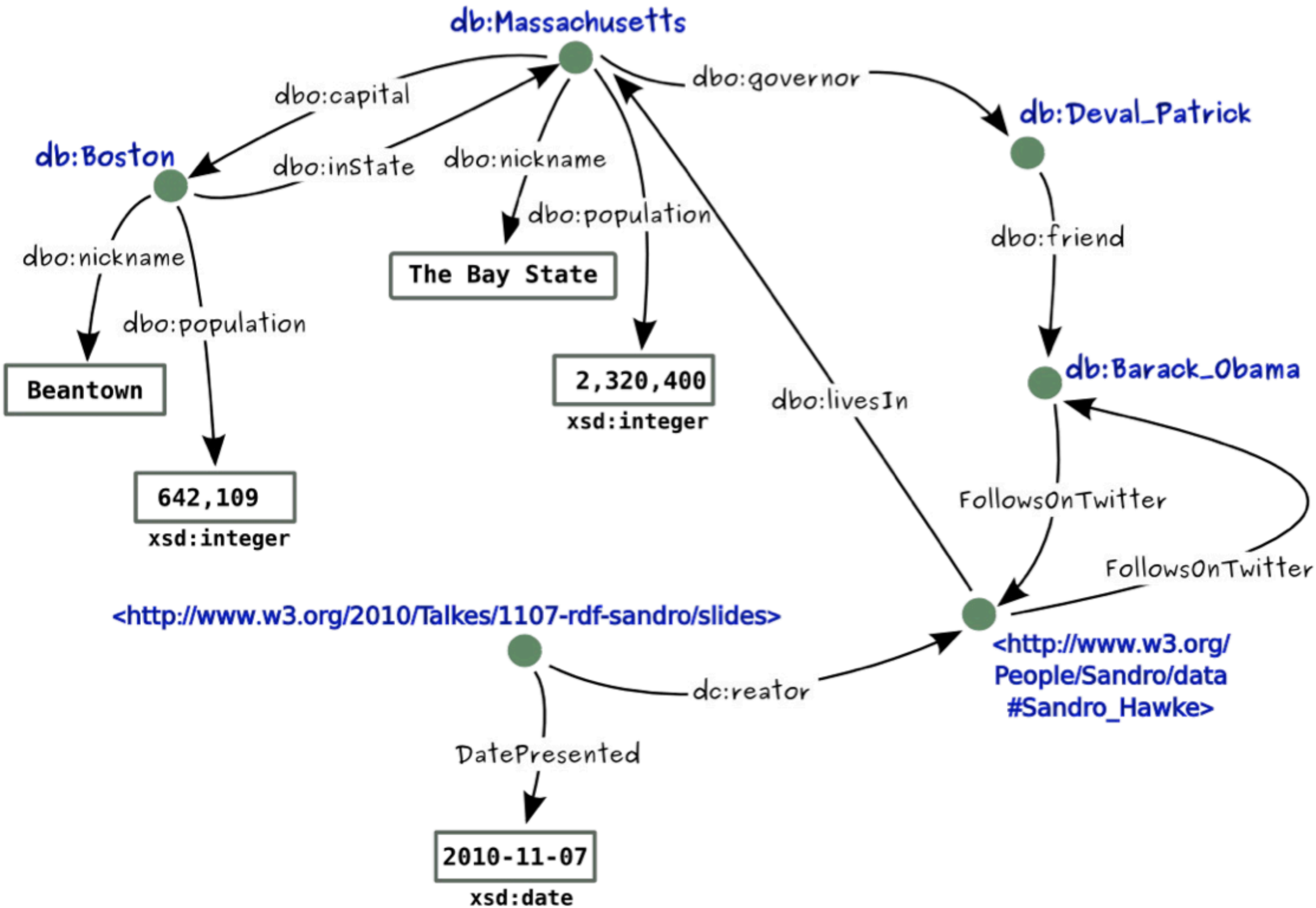
Unambiguous Names (cntd)

- How many things are named "Boston"?

➔ So, we use URIs. Instead of "Boston":
  - http://dbpedia.org/resource/Boston
  - QName: db:Boston

➔ And instead of "Nickname" we use:
  - http://example.org/terms/nickname
  - QName: dbo:nickname

Note: we have to say somewhere that "db" is a shorthand for "http://dbpedia.org/resource/Boston/"


db:Massachusetts
dbo:capital
dbo:governor
db:Deval_Patrick
db:Boston
dbo:inState
dbo:nickname
dbo:population
dbo:friend
dbo:nickname
The Bay State
dbo:population
Beantown
2,320,400
xsd:integer
db:Barack_Obama
dbo:livesIn
642,109
xsd:integer
FollowsOnTwitter
FollowsOnTwitter
<http://www.w3.org/2010/Talkes/1107-rdf-sandro/slides>
<http://www.w3.org/People/Sandro/data#Sandro_Hawke>
dc:reator
DatePresented
2010-11-07
xsd:date

RDF is…

a schema-less data model that features unambiguous identifiers and named relations between pairs of resources.

Why RDF? What's Different Here?

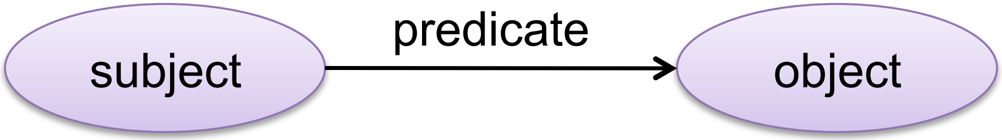
- The graph data structure makes *merging data* with shared identifiers *trivial* (as we saw earlier)
- Triples act as a *least common denominator* for expressing data
- URIs for naming remove ambiguity
  - …the same identifier means the same thing

- History and Motivation
- Naming: URIs, IRIs, Qnames
- RDF Data Model: Triples, Literals, Types
- Modeling with RDF: BNodes, n-ary Relations, Reification
- Containers

RDF is…

A labeled, directed graph of relations between resources and literal values.

- RDF graphs are sets of *triples*
- Triples are made up of a *subject*, a *predicate*, and an *object* (spo)



- Resources and relationships are named with *URIs*

Triple

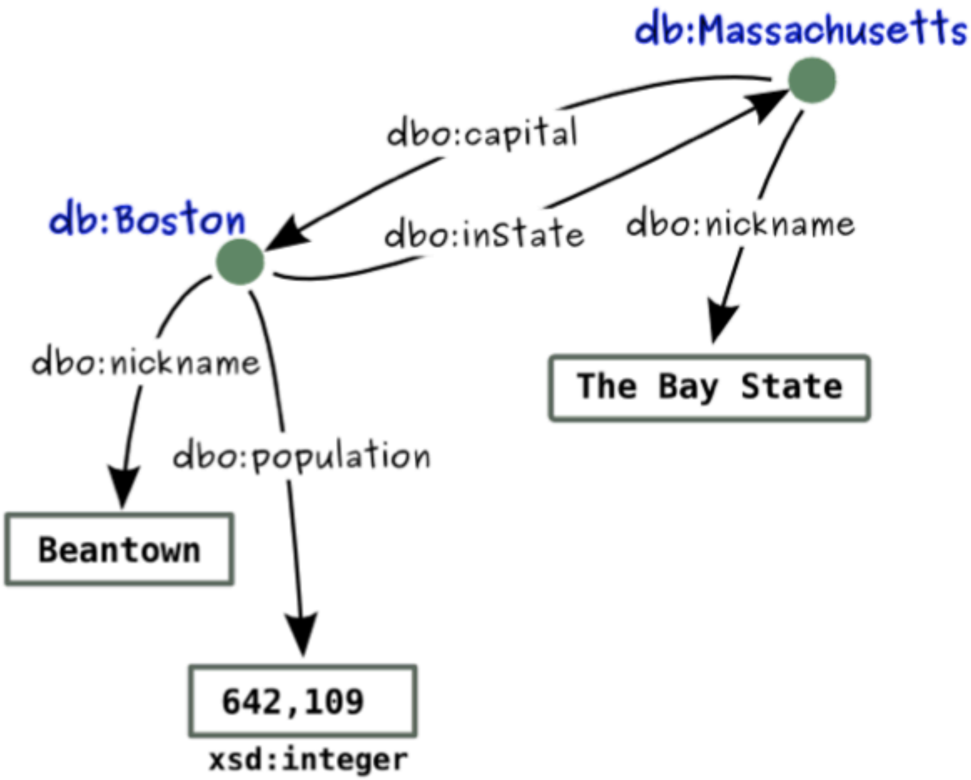
- Resources are: IRI (denotes an object)

- **Subjects**: Resource or blank-node
- **Predicates**: Resource
- **Object**: Resource, literal or blank-node

A triple is also called a “statement”

Turtle Syntax

- Turtle = Terse RDF Triple Language
  - Simple syntax for RDF
  - defined by Dave Beckett as a subset of Tim Berners-Lee and Dan Connolly's Notation3 (N3) language
  - standardized by a W3C recommendation since February 2014
- In Turtle, triples are directly listed as such: S P O
  - IRIs are in < angle brackets >
  - End with full-stop "."
  - Whitespaces are ignored

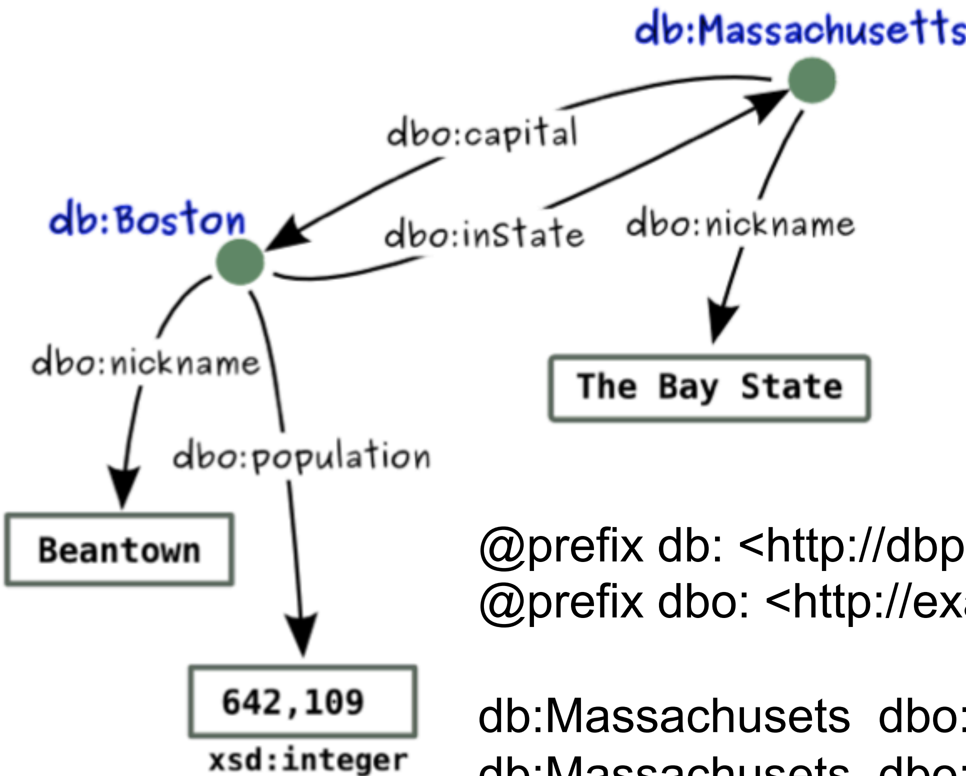

db:Massachusetts
dbo:capital
dbo:inState
dbo:nickname
db:Boston
The Bay State
dbo:nickname
dbo:population
Beantown
642,109
xsd:integer

In Turtle

```
<http://dbpedia.org/resource/Massachusets>  <http://example.org/terms/captial>
      <http://dbpedia.org/resource/Boston>  .

<http://dbpedia.org/resource/Massachusets> <http://example.org/terms/nickname>
      “The Bay State”  .

<http://dbpedia.org/resource/Boston> <http://example.org/terms/inState>
<http://dbpedia.org/resource/Massachusets>  .

<http://dbpedia.org/resource/Boston> <http://example.org/terms/nickname>
“Beantown”  .

<http://dbpedia.org/resource/Boston> <http://example.org/terms/population>
“642,109”^^xsd:integer  .
```

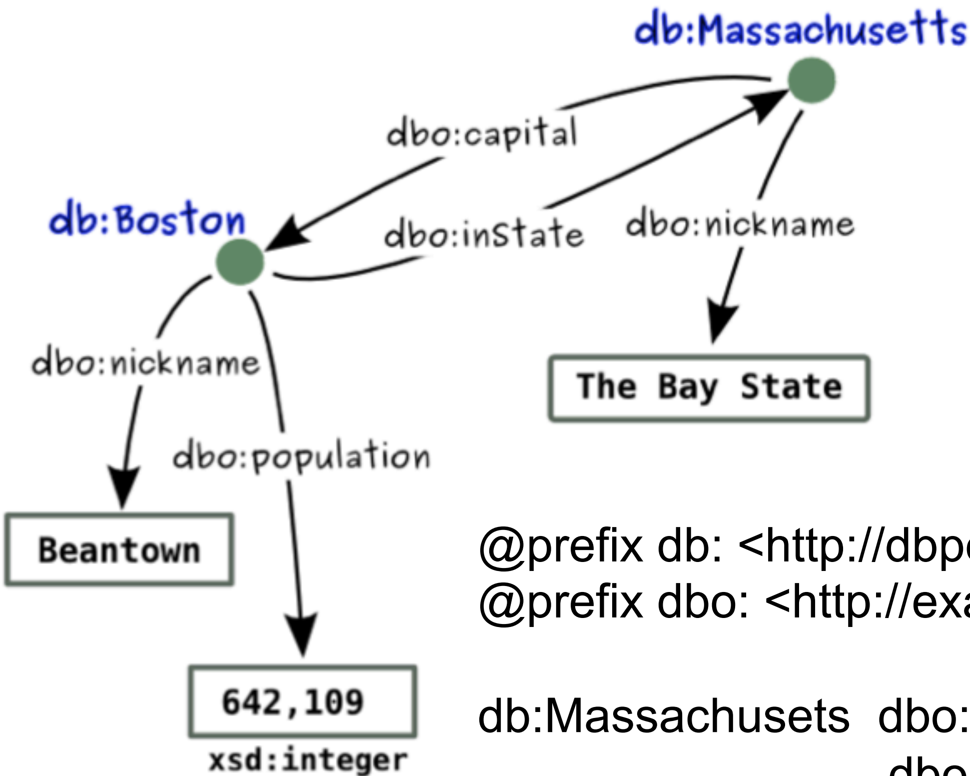
Shortcuts

- Prefixes (simple string concatenation)
- Grouping of triples with the same subject
  using semi-colon ‘;’
- Grouping of triples with the same subject and predicate
  using comma ‘,’



@prefix db: <http://dbpedia.org/resource/>
@prefix dbo: <http://example.org/terms/>

db:Massachusets	dbo:capital	db:Boston .
db:Massachusets	dbo:nickname	“The Bay State” .
db:Boston	dbo:inState	db:Massachusets .
db:Boston	dbo:nickname	“Beantown” .
db:Boston	dbo:population	“642,109”^^xsd:integer .




```
@prefix db: <http://dbpedia.org/resource/>
@prefix dbo: <http://example.org/terms/>

db:Massachusets  dbo:captial     db:Boston  ;
                 dbo:nickname    "The Bay State"  .
db:Boston        dbo:inState     db:Massachusets ;
                 dbo:nickname    "Beantown"  ;
                 dbo:population  "642,109"^^xsd:integer  .
```

Literals

- Represent data values
- Encoded as strings (the value)
- Can be interpreted by means of datatypes
- Literals without a type are treated the same as string (but they are not equal to strings)
- An literal without a type is called plain literal.
  A plain literal may have a language tag.
- Datatypes are not defined by RDF,
  people commonly use datatypes from XML Schema (XSD)
- RDF does not require implementation support for any datatype.
  However, systems generally implement most of XSD datatypes.

Literals (cont.)

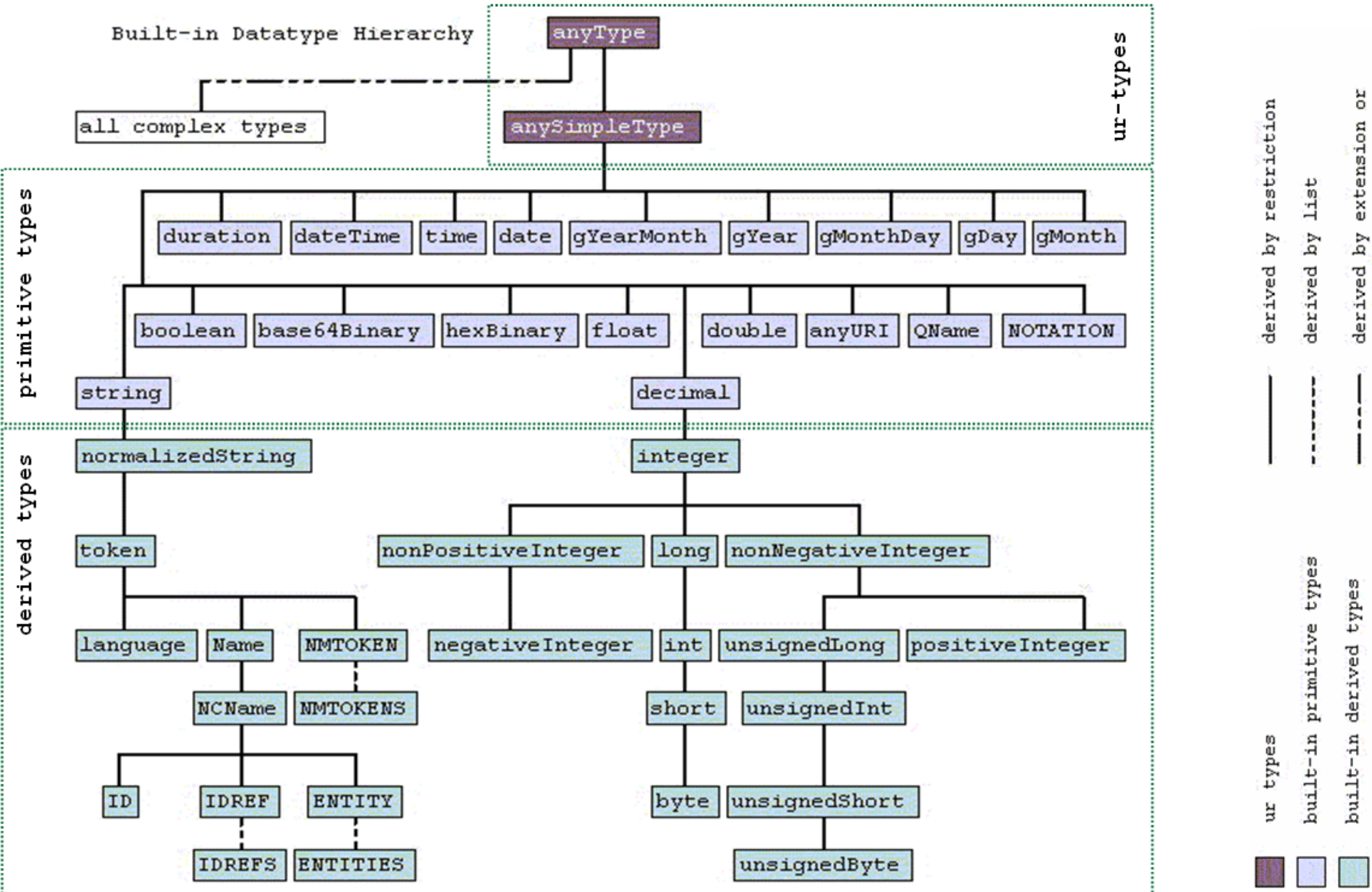
- Typed literal:
  - “Mariano”^^xsd:string, “12-12-12”^^xsd:date

- Plain literal and literals with language
  - “France”   “France”@fr   “Frankreich”@de

- Equalities under simple RDF interpretation (lexical form matters):
  - “Mariano” != “Mariano”@es != “Mariano”^^xsd:string
  - “001”^^xsd:integer = “1”^^xsd:integer

Literals (cont.)

- Equalities under typed interpretation (lexical form does not matter):
  - “123”^^xsd:integer = “0123”^^xsd:integer
  - Type hierarchy:
    “123.0”^^xsd:decimal = “00123”^^xsd:integer


Built-in Datatype Hierarchy
anyType
all complex types
anySimpleType
ur-types
primitive types
duration
dateTime
time
date
gYearMonth
gYear
gMonthDay
gDay
gMonth
boolean
base64Binary
hexBinary
float
double
anyURI
QName
NOTATION
string
decimal
derived types
normalizedString
integer
token
nonPositiveInteger
long
nonNegativeInteger
language
Name
NMTOKEN
negativeInteger
int
unsignedLong
positiveInteger
NCName
NMTOKENS
short
unsignedInt
ID
IDREF
ENTITY
byte
unsignedShort
IDREFS
ENTITIES
unsignedByte
derived by restriction
derived by list
derived by extension or
ur types
built-in primitive types
built-in derived types

Type definition

- Datatypes can be defined by the user, as with XML
- New “derived simple types” are derived by restriction, as with XML. Complex types based on enumerations, unions and list are also possible. Example:

```
<xsd:schema ...>
    <xsd:simpleType name="humanAge">
      <xsd:restriction base="integer">
       <xsd:minInclusive value="0">
       <xsd:maxExclusive value="150">
      </xsd:restriction>
    </xsd:simpleType>
    ...
</xsd:schema>
```

- History and Motivation
- Naming: URIs, IRIs, Qnames
- RDF Data Model: Triples, Literals, Types
- Modeling with RDF: BNodes, N-ary Relations, Reification
- Containers

Modeling with RDF

- Lets revisit our motivational examples and do some modeling in RDF ourselves.
- Given the following relational data, generate an RDF graph

Exercise:
Data set “A”: A *simplified* Book Store

Sellers

<ID>	Author	Title	<Publisher>	Year
ISBN0-00-651409-X	id_xyz	The Glass Palace	id_qpr	2000

Authors

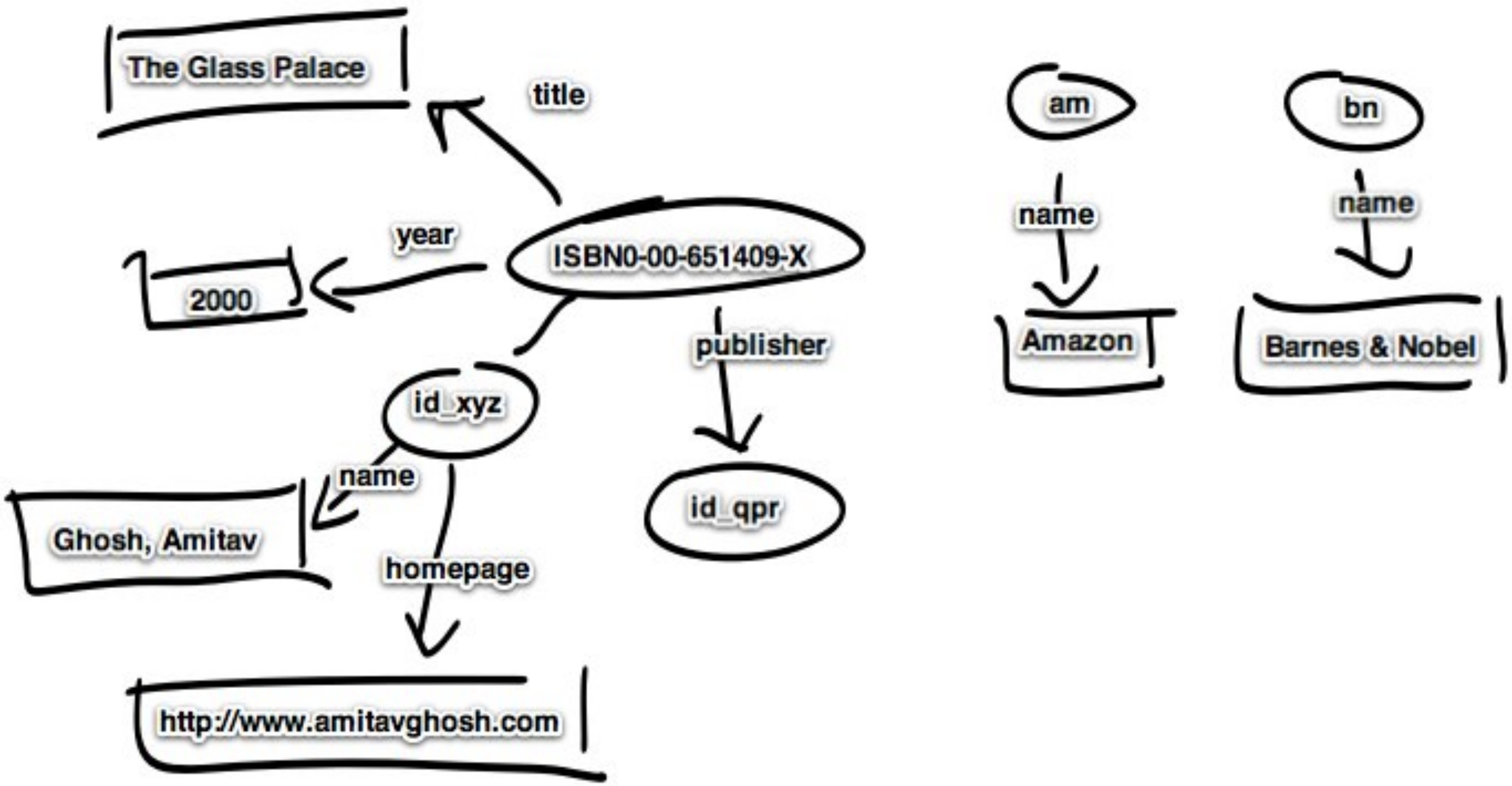
<ID>	Name	Home page
id_xyz	Ghosh, Amitav	http://www.amitavghosh.com

Stores

<ID>	Publisher Name
am	Amazon
bn	Barnes & Nobel

Generate an RDF graph.
Keys are marked with <>.
Primary keys are underscored.
Steps:
1) Generate the graph
2) Adjust identifiers
3) Adjust names of relations and types

Relational to Graph (not yet RDF)



Insert …

- Types

- Proper URIs, e.g., using

  @prefix ex: http://example.org/

- Blank nodes

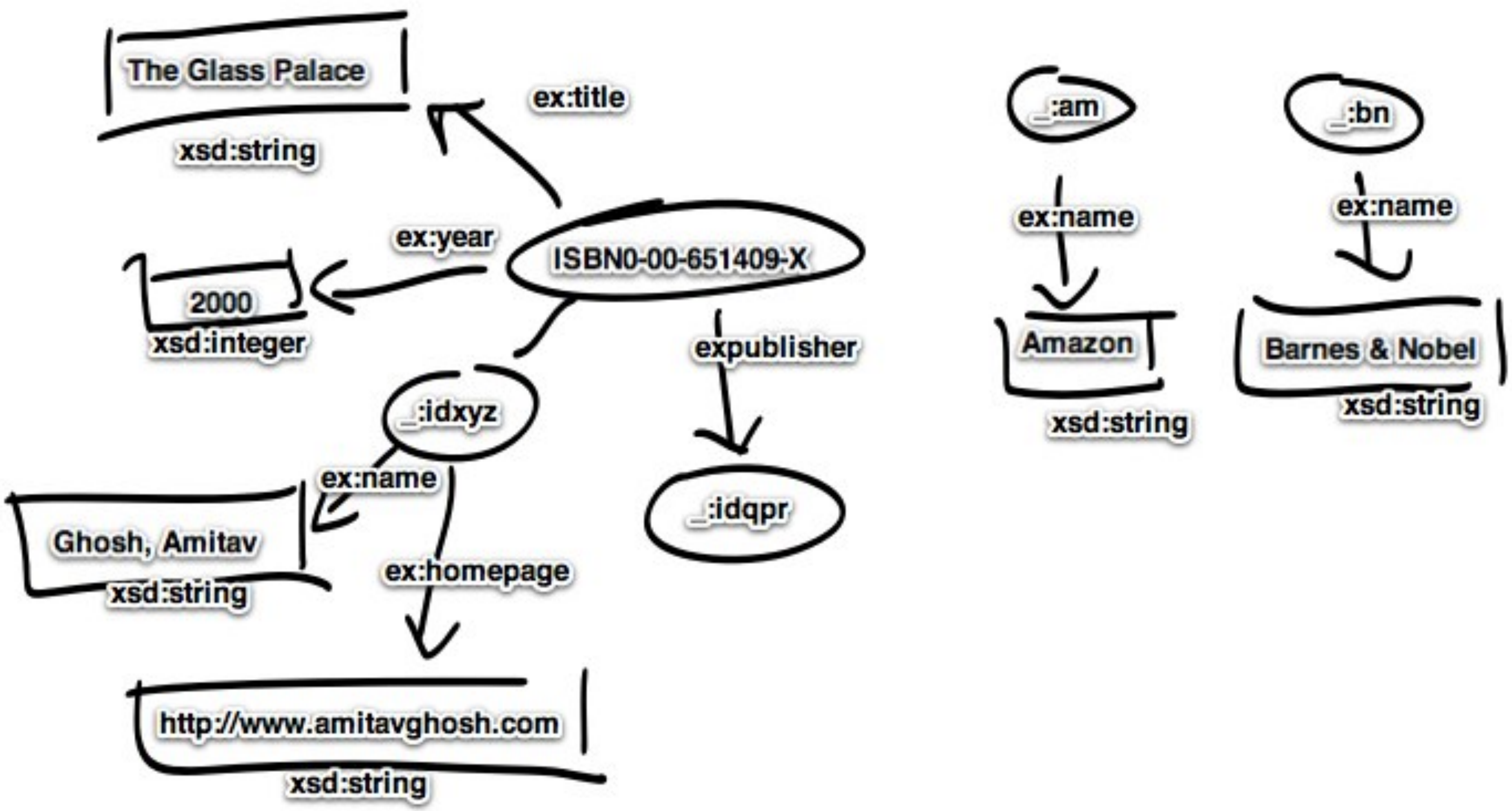
Blank Nodes

- Nodes without a IRI
  - Unnamed resources
  - Complex nodes (later)

- Representation of blank nodes is syntax-dependent
  - In Turtle we use underscore followed by colon, then an ID
  - `_:b0` `_:nodeX`

- The scope of the ID of a blank node is only the document to which it belongs.
  That is, two different RDF files that contain the blank node `_:n0` do not refer to the same node.

With Proper URI's and BNodes



Insert also…

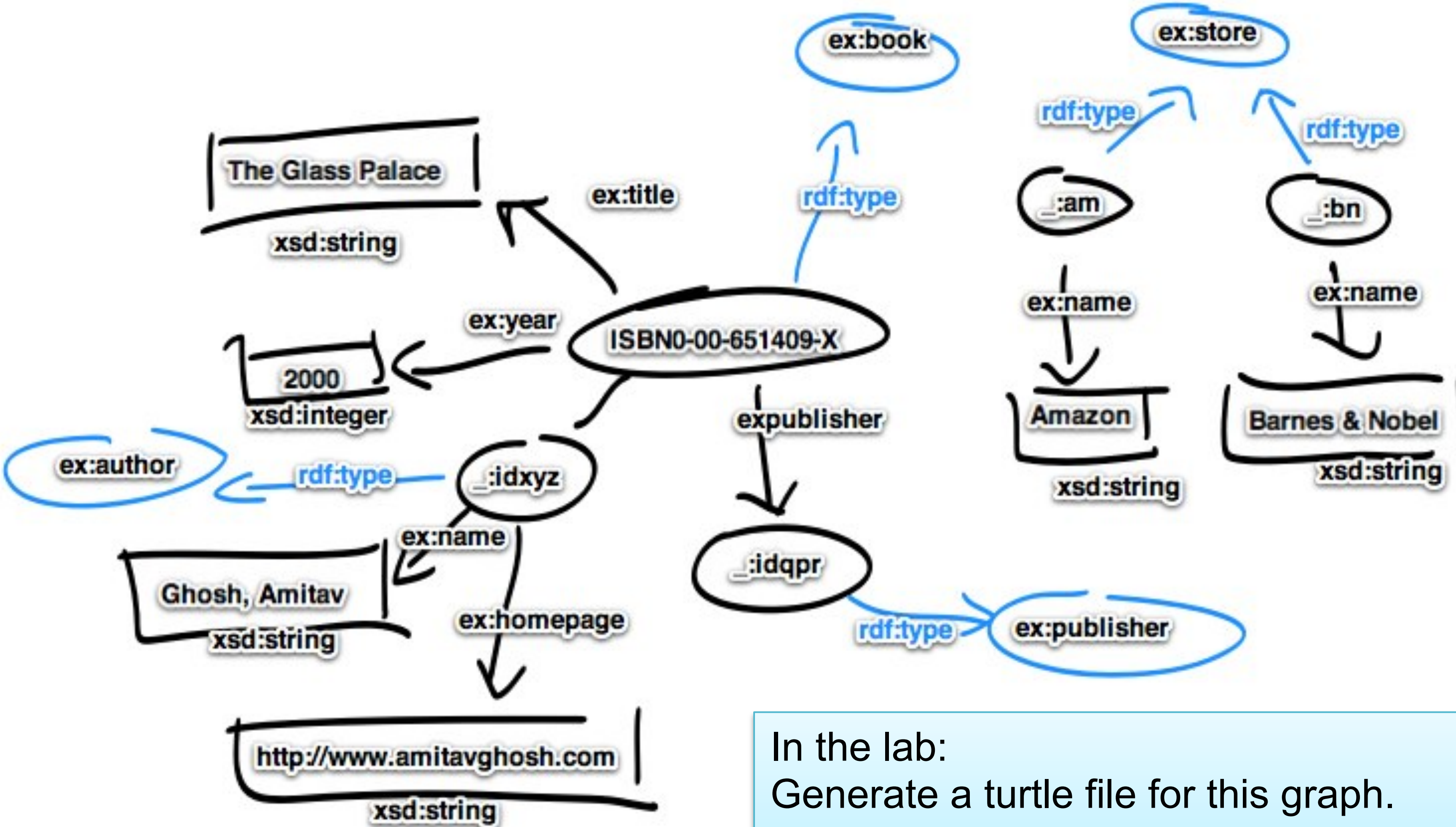
- Classes/Types
  - invent new class names for your vocabulary
  - class names are URIs (like everything else)

- Class membership links, using the predicate

  rdf:type

  - Syntax: dbo:Boston rdf:type ex:City
    with
    @prefix ex:http//example.org/

Complete with rdf:type



In the lab:
Generate a turtle file for this graph.
Additionally, transform it into n3 and RDF/XML file using Sesame or Jena

Data set "A+": The Simplified Book Store

Sellers

<ID>	Author	Title	<Publisher>	Year
ISBN0-00-651409-X	id_xyz	The Glass Palace	id_qpr	2000

Authors

<ID>	Name	Home page
id_xyz	Ghosh, Amitav	http://www.amitavghosh.com

Stores

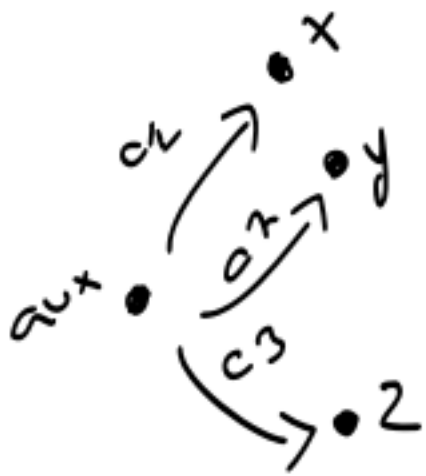
<ID>	Publisher Name
am	Amazon
bn	Barnes & Nobel

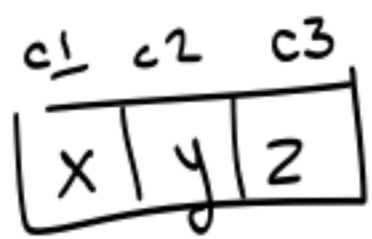
Sold-By

<Book>	<Store>	Price
ISBN0-00-651409-X	am	22.50
ISBN0-00-651409-X	bn	21.00

N-ary Relations

- Not all relations are binary
- All n-ary relations can be "encoded" as a set of binary relations using auxiliary nodes.
- This process is called "reification" in conceptual modeling (do not confuse with reification in RDFS, to come later).





Data set “A+”: The Simplified Book Store

Sellers

ID	Author	Title	Publisher	Year
ISBN0-00-651409-X	id_xyz	The Glass Palace	id_qpr	2000

Authors

ID	Name	Home page
id_xyz	Ghosh, Amitav	http://www.amitavghosh.com

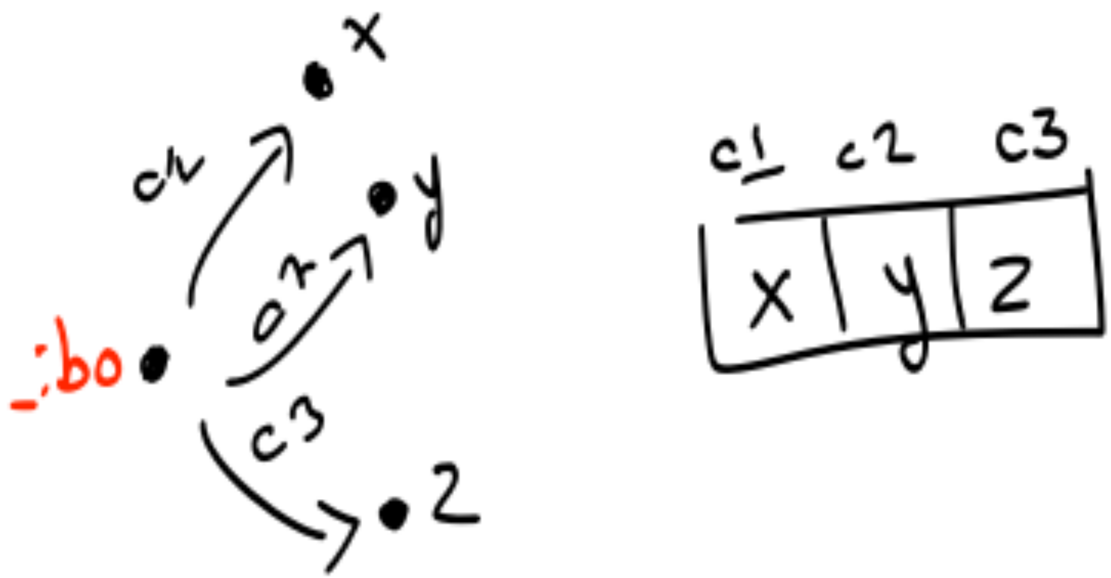
Stores

ID	Publisher Name
am	Amazon
bn	Barnes & Nobel

Sold-By

Book	Store	Price
ISBN0-00-651409-X	am	22.50
ISBN0-00-651409-X	bn	21.00

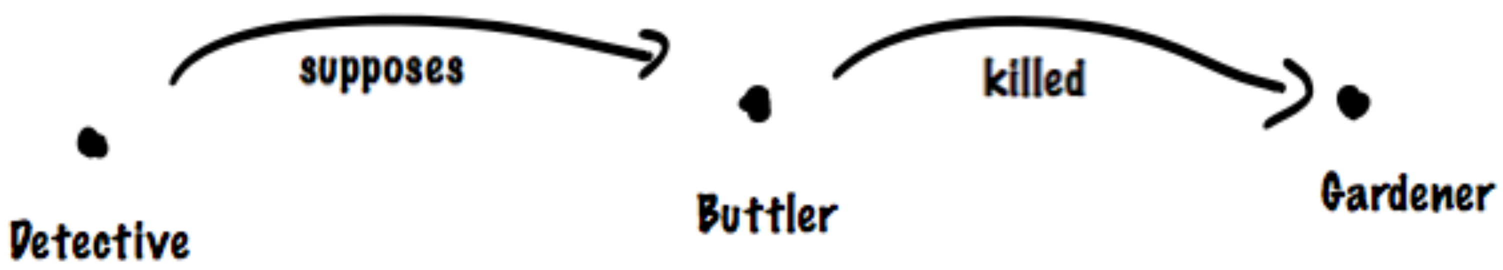
Use Blank Nodes to Model Tuples



RDF Reification

- How would you state in RDF:

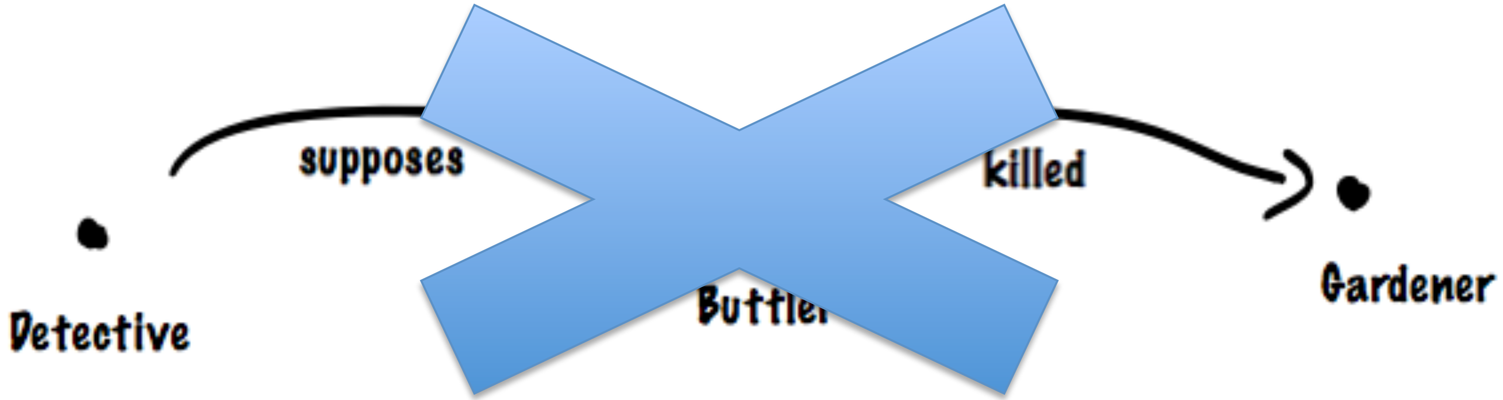
  "The detective supposes that the butler killed the gardener"



RDF Reification

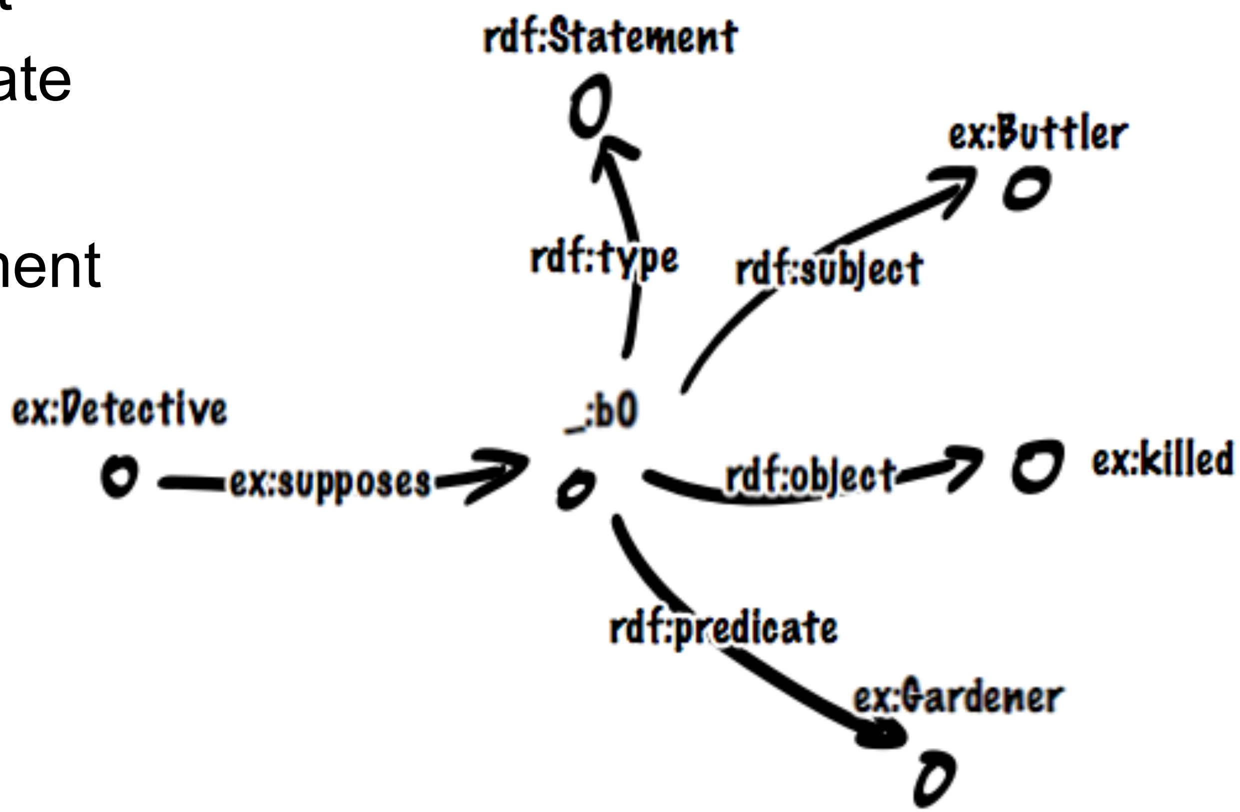
- How would you state in RDF:

  "The detective supposes that the butler killed the gardener"



RDF Reification

- Reification allows to state statements about statements
- Use special vocabulary:
  - rdf:subject
  - rdf:predicate
  - rdf:object
  - rdf:Statement



RDF Reification

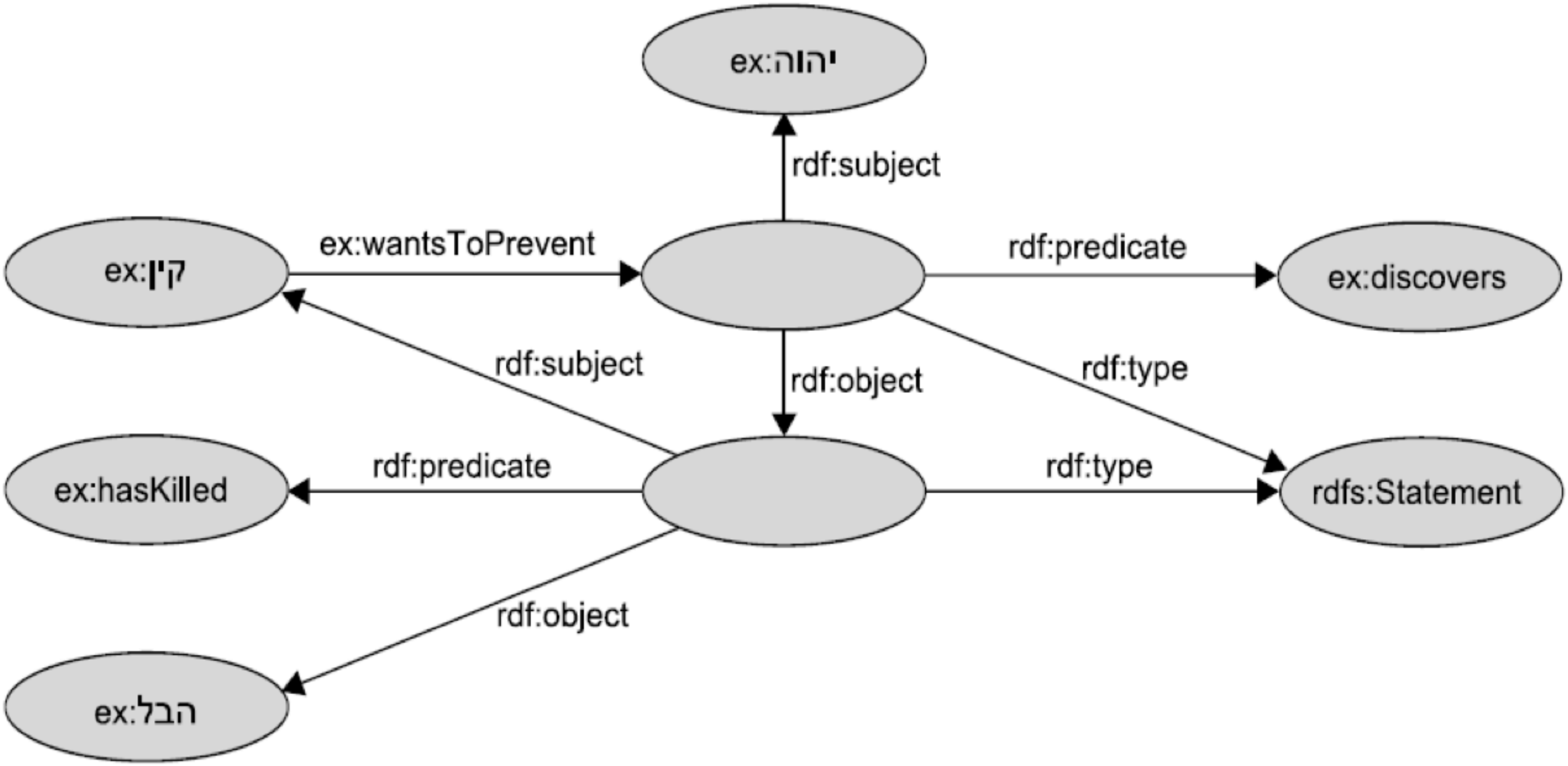
- Reification allows to state statements about statements
- Use special vocabulary:
  - rdf:subject
  - rdf:predicate
  - rdf:object
  - rdf:Statemen

Note: The triple

<Buttler> <Killed> <Gardener>

Is *not* in the graph.

A Reification Puzzle



Know the story?

Exercise

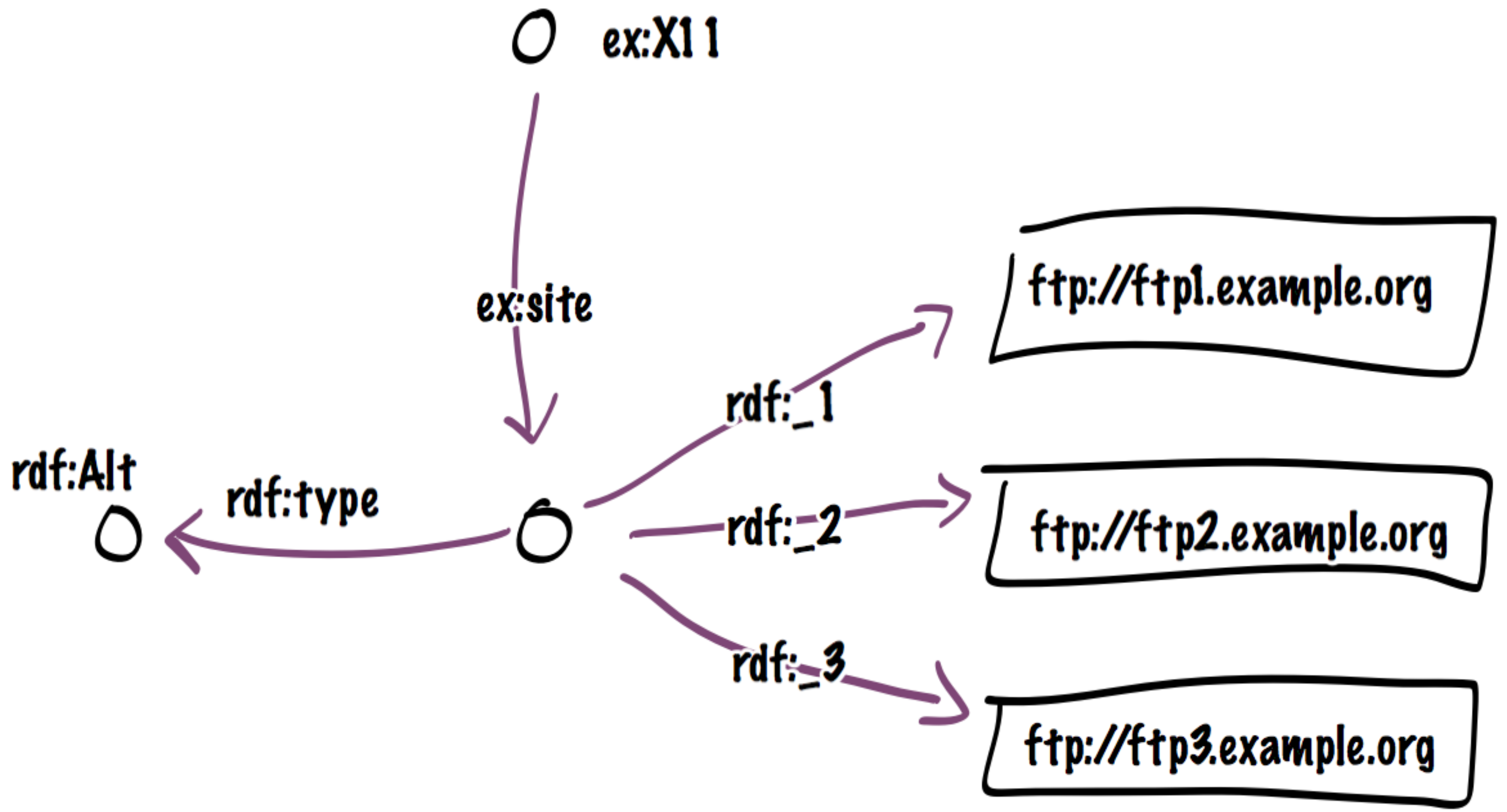
- Express the following natural language sentences as a graph:
  - Mary saw Eric eat ice cream
  - The professor explained that the scientific community regards evolution theory as correct

- History and Motivation
- Naming: URIs, IRIs, Qnames
- RDF Data Model: Triples, Literals, Types
- Modeling with RDF: BNodes, N-ary Relations, Reification
- Containers

Containers

- Groups of resources
  - **rdf:Bag**: Group, possibly with duplicates, no order
  - **rdf:Seq**: Group, possibly with duplicates,
    order matters
  - **rdf:Alt**: Group, indicates alternatives
- Use **rdf:type** to indicate one type of container
- Use **container membership properties** to enumerate:
  - rdf:_1, rdf:_2, rdf:_3, …, rdf:_n

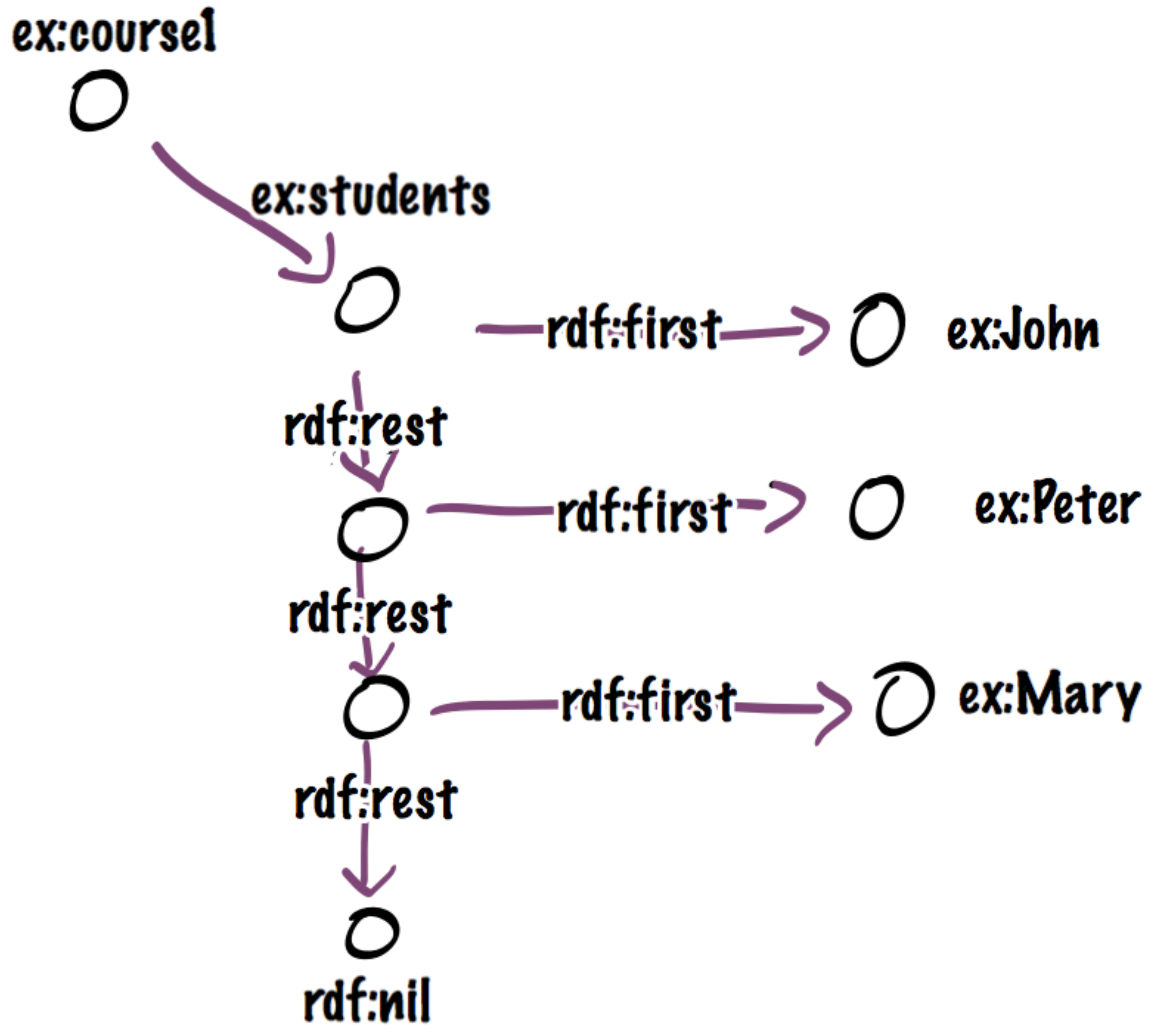
Example


ex:X11
ex:site
rdf:Alt
rdf:type
rdf:_1
rdf:_2
rdf:_3
ftp://ftp1.example.org
ftp://ftp2.example.org
ftp://ftp3.example.org

Collections (Closed Containers)

- Containers are open. No way to “close them”. Imposible to say “no other member exists”. Consider merging datasets.
- Group of things represented as a linked list structure
- The list is defined using the RDF vocabulary:
  - rdf:List,
  - rdf:first,
  - rdf:rest and
  - rdf:nil
- Each member of the list is of type rdf:List (implicitly)

Example


ex:course1
ex:students
rdf:first
ex:John
rdf:rest
rdf:first
ex:Peter
rdf:rest
rdf:first
ex:Mary
rdf:rest
rdf:nil

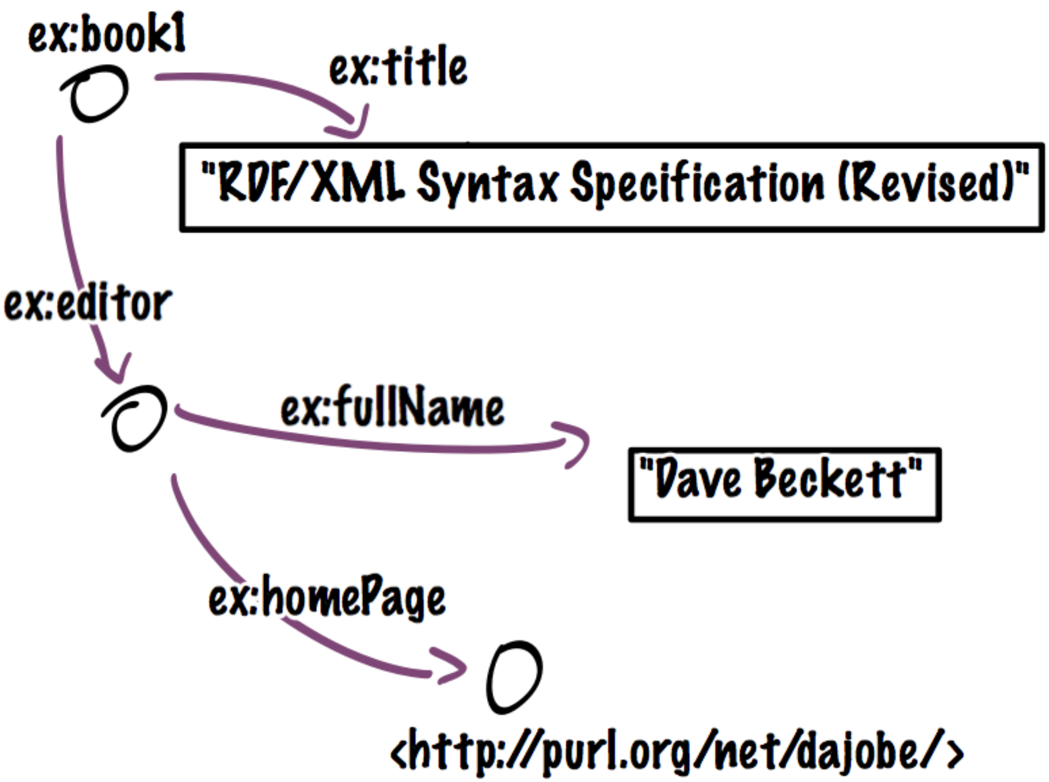
Turtle

- We already covered most of the nuts and bolts
- Missing: Blank nodes, containers

Blank Nodes

- Use square brackets to define a blank node

```
ex:book1
   ex:title "RDF/XML Syntax
             Specification (Revised)";
   ex:editor [
     ex:fullName "Dave Beckett";
     ex:homePage
      <http://purl.org/net/dajobe/>
   ].
```



Exercise

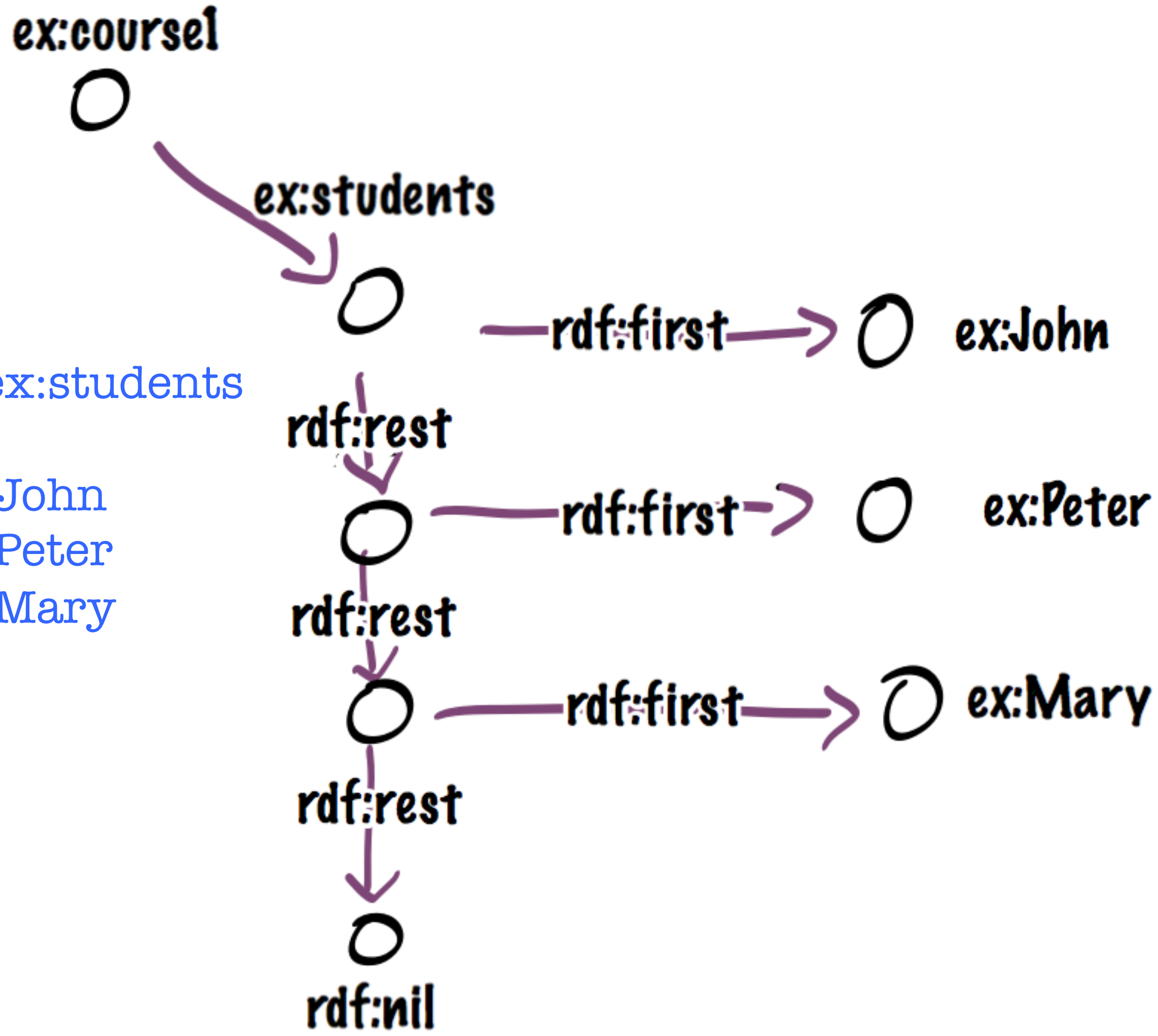
Write in Turtle syntax with blank nodes a representation of

“Mary saw Eric eat ice cream”.

Containers

- Use ( )

```
ex:coursel ex:students
   (
        ex:John
        ex:Peter
        ex:Mary
   ) .
```



Turtle

- Advantages and uses:
  - Easy to read and write manually or programmatically
  - Good performance for IO, supported by many tools
- The Turtle standard does not comprise containers

N-Triples

- Turtle minus:
  - No prefix definitions are allowed
  - No reference shortcuts (semi-colon, comma)
  - Every other shortcut ☺
- Very simple to parse/generate (even through scripts)
- Supported by most tools
- Very verbose. Wastes space/IO (problem is reduced with compression)

RDF/XML

- W3C Standard since 1999, revised in 2004
- Used to be the only standard
- Standard XML (works with any XML tools)