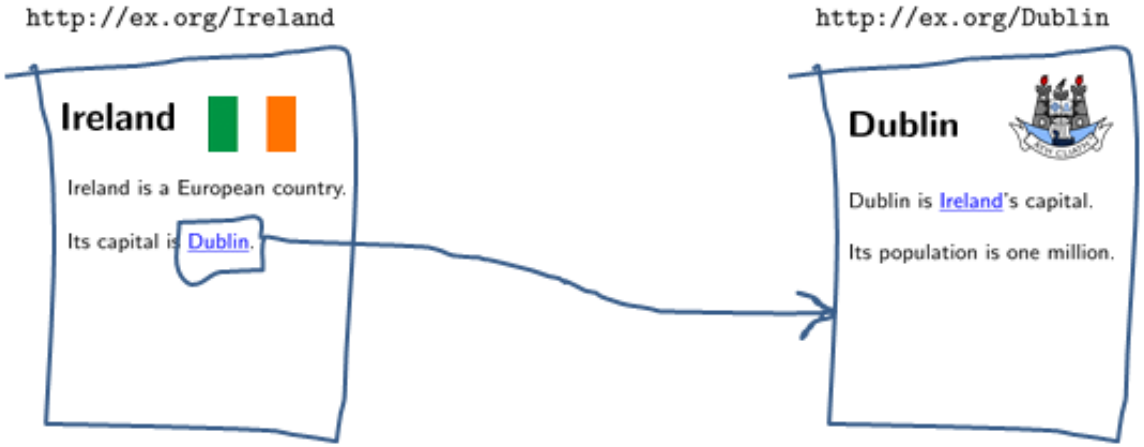
CC6202-1
La Web de Datos
Primavera 2015

Lecture 10: Conclusion

Aidan Hogan
aidhog@gmail.com

We've gotten from here …

The current Web is document-centric




```
<html>
 <body>
  <h1>Ireland</h1>
  <div class="flag"><img src="flag.jpg" /></div>
  <p>Ireland is a European country.</p>
  <p>Its capital is <a href="http://ex.org/Dublin">Dublin</a>.</p>
  ...
 </body>
</html>
```

To here (with (4) Links!)

(1) Data, (2) Rules/Ontologies, (3) Query

INPUT: "(x,partOf,y)"

DATA:

http://ex.org/Ireland

Ireland 

(Ireland,partOf,Europe)
(Ireland,a,Country)
(Ireland,capital,Dublin)

http://ex.org/Dublin

Dublin 

(Dublin,population,1000000)

RULES: $(a, \text{capital}, b) \rightarrow (b, \text{partOf}, a)$
$(c, \text{partOf}, d), (d, \text{partOf}, e) \rightarrow (c, \text{partOf}, e)$

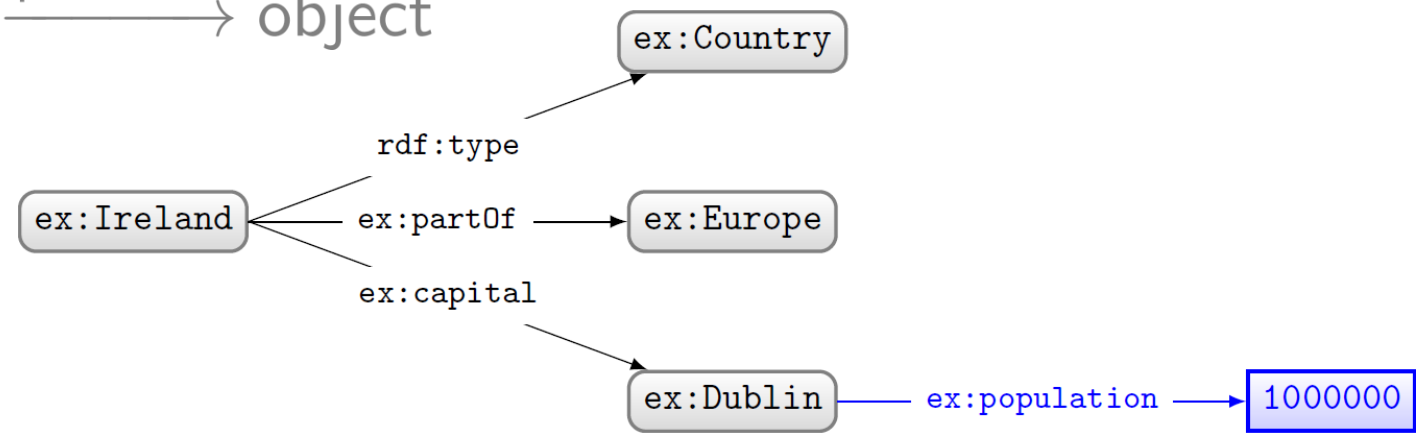
OUTPUT: $\{(x \mapsto \text{Ireland}, y \mapsto \text{Europe}), (x \mapsto \text{Dublin}, y \mapsto \text{Ireland})$
$(x \mapsto \text{Dublin}, y \mapsto \text{Europe})\}$

DATA? RDF!

Use RDF as a core data model

subject	*predicate*	*object*
ex:Ireland	ex:partOf	ex:Europe
ex:Ireland	rdf:type	ex:Country
ex:Ireland	ex:capital	ex:Dublin
ex:Dublin	ex:population	1,000,000

subject —predicate→ object



A Summary of RDF Terms

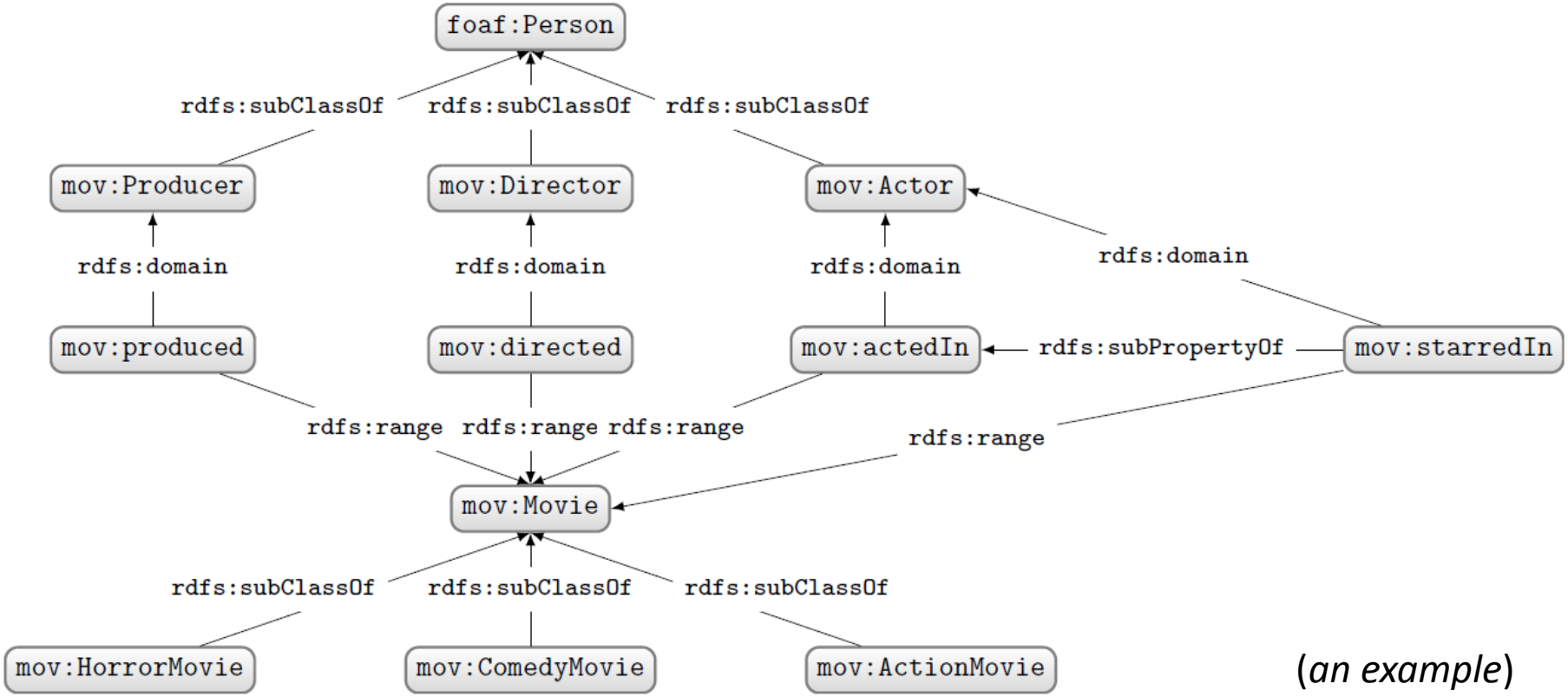
1. IRIs (Internationalised Resource Identifiers)
   - Used to name generic things
2. Literals
   - Used to refer to datatype values
   - Strings may have a language tag
3. Blank Nodes
   - Used to avoid naming things
   - A little mysterious right now

subject	*predicate*	*object*
[IRI, Blank Node]	[IRI]	[IRI, Blank Node, Literal]

RULES/ONTOLOGIES? RDFS!

RDF Schema: lightweight semantics

Let's model an RDF Schema for Movies, including different types of movies, some different types of people involved, and how they are related.



(*an example*)

RDFS: Describe “schema” in RDF

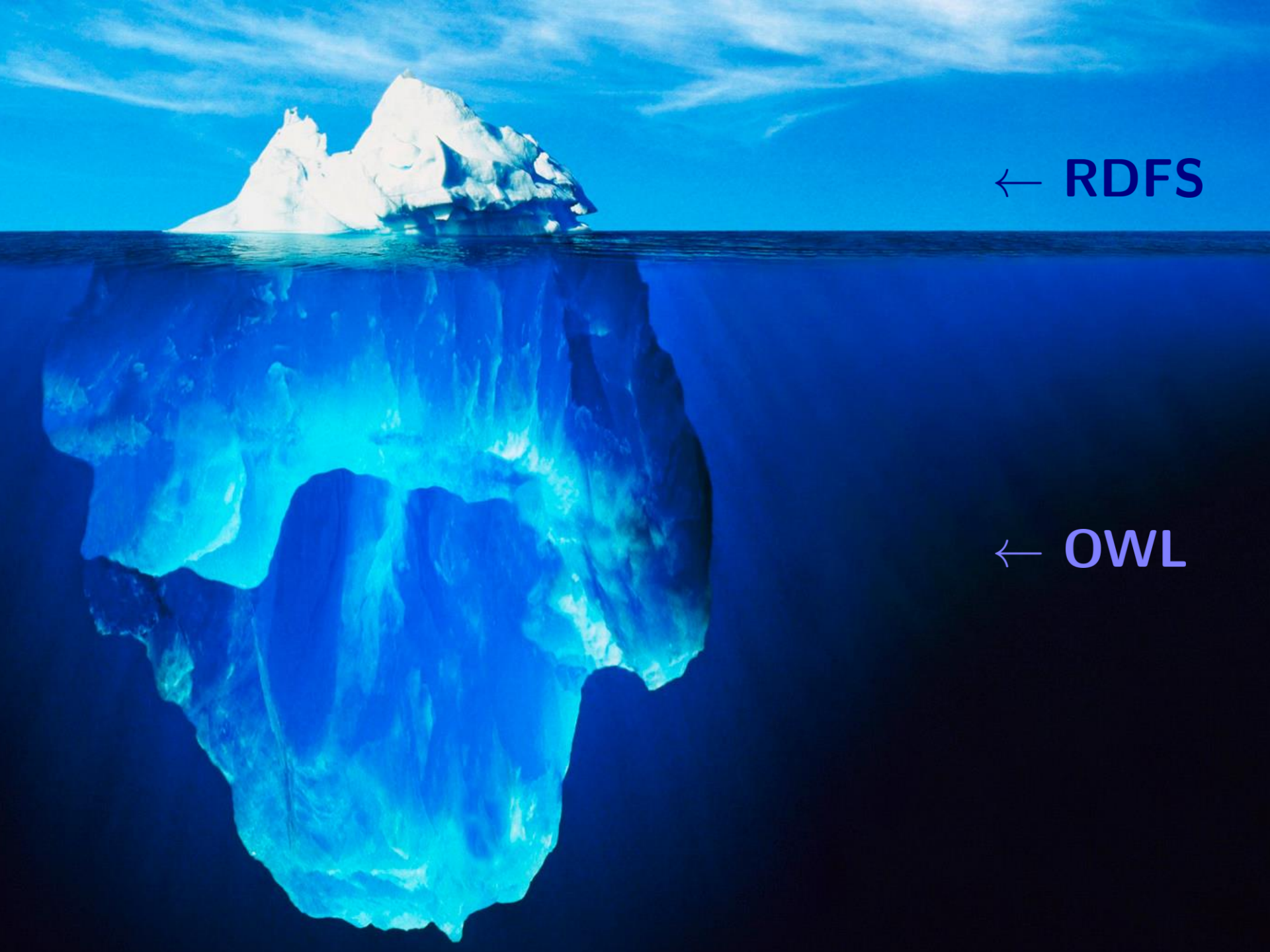
- Sub-class:
  - `ex:CapitalCity rdfs:subClassOf ex:City .`
- Sub-property:
  - `ex:hasCapitalCity rdfs:subPropertyOf ex:hasCity .`
- Domain:
  - `foaf:familyName rdfs:domain foaf:Person .`
- Range:
  - `ex:hasCapitalCity rdfs:range ex:CapitalCity .`
  - `foaf:familyName rdfs:range xsd:string .`

Apply RDFS reasoning using “rules”

ID	if G matches	then G RDFS$_D$-entails
rdfD1	?x ?p ?l . (?l a literal with datatype IRI dt(?l) $\in D$)	?x ?p _:b . _:b a dt(?l) .
rdfD2	?x ?p ?y .	?p a rdf:Property .
rdfs1	?u $\in D$	?u a rdfs:Datatype .
rdfs2	?p rdfs:domain ?c . ?x ?p ?y .	?x a ?c .
rdfs3	?p rdfs:range ?c . ?x ?p ?y .	?y a ?c .
rdfs4a	?x ?p ?y .	?x a rdfs:Resource .
rdfs4b	?x ?p ?y .	?y a rdfs:Resource .
rdfs5	?p rdfs:subPropertyOf ?q . ?x ?p ?y .	?x ?q ?y .
rdfs6	?p a rdf:Property .	?p rdfs:subPropertyOf ?p .
rdfs7	?p rdfs:subPropertyOf ?q . ?q rdfs:subPropertyOf ?r .	?p rdfs:subPropertyOf ?r .
rdfs8	?c a rdfs:Class .	?c rdfs:subClassOf rdfs:Resource .
rdfs9	?c rdfs:subClassOf ?d . ?x a ?c .	?x a ?d .
rdfs10	?c a rdfs:Class .	?c rdfs:subClassOf ?c .
rdfs11	?c rdfs:subClassOf ?d . ?d rdfs:subClassOf ?e .	?c rdfs:subClassOf ?e .
rdfs12	?p a rdfs:ContainerMembershipProperty .	?p rdfs:subPropertyOf rdfs:member .
rdfs13	?d a rdfs:Datatype .	?d rdfs:subClassOf rdf:Literal .

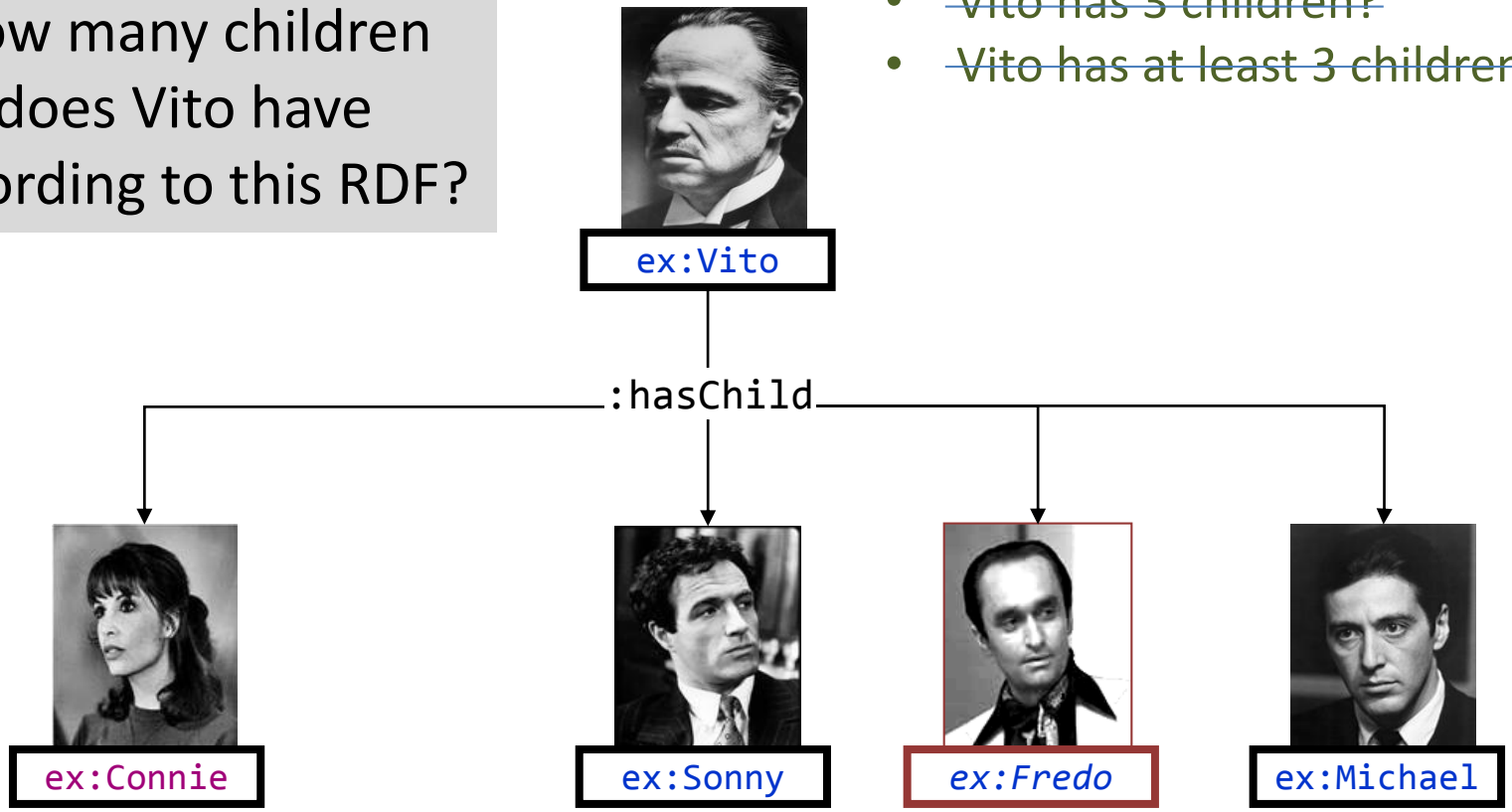
(Don’t worry about rdfD1, rdfs1, rdfs12, rdfs13)

RULES/ONTOLOGIES? OWL!


← RDFS
← OWL

Open World Assumption (OWA)

How many children does Vito have according to this RDF?

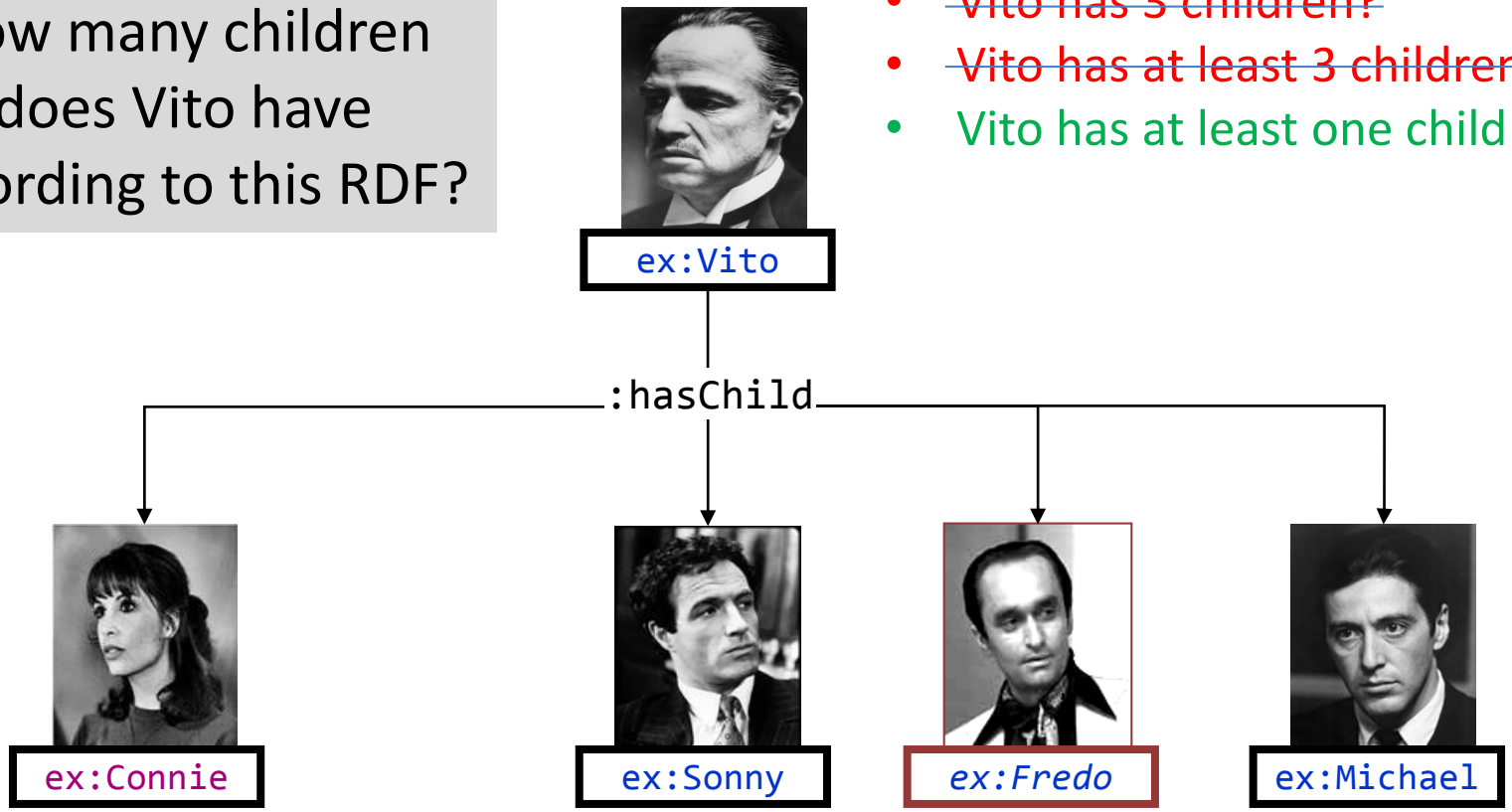


- ~~Vito has 3 children?~~
- ~~Vito has at least 3 children?~~

```
ex:Vito :hasChild ex:Connie, ex:Sonny, ex:Michael .
ex:Vito :hasChild ex:Fredo .
... ?
```

No Unique Name Assumption (No UNA)

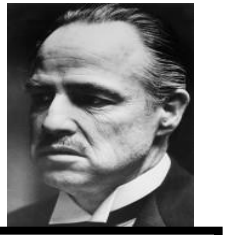
How many children does Vito have according to this RDF?



- ~~Vito has 3 children?~~
- ~~Vito has at least 3 children?~~
- Vito has at least one child!

```
ex:Vito :hasChild ex:Connie, ex:Sonny, ex:Michael .
ex:Vito :hasChild ex:Fredo .
... ?
```

All the features …



ex:Vito

ex:Carmela

ex:Sonny

ex:Connie

ex:Fredo

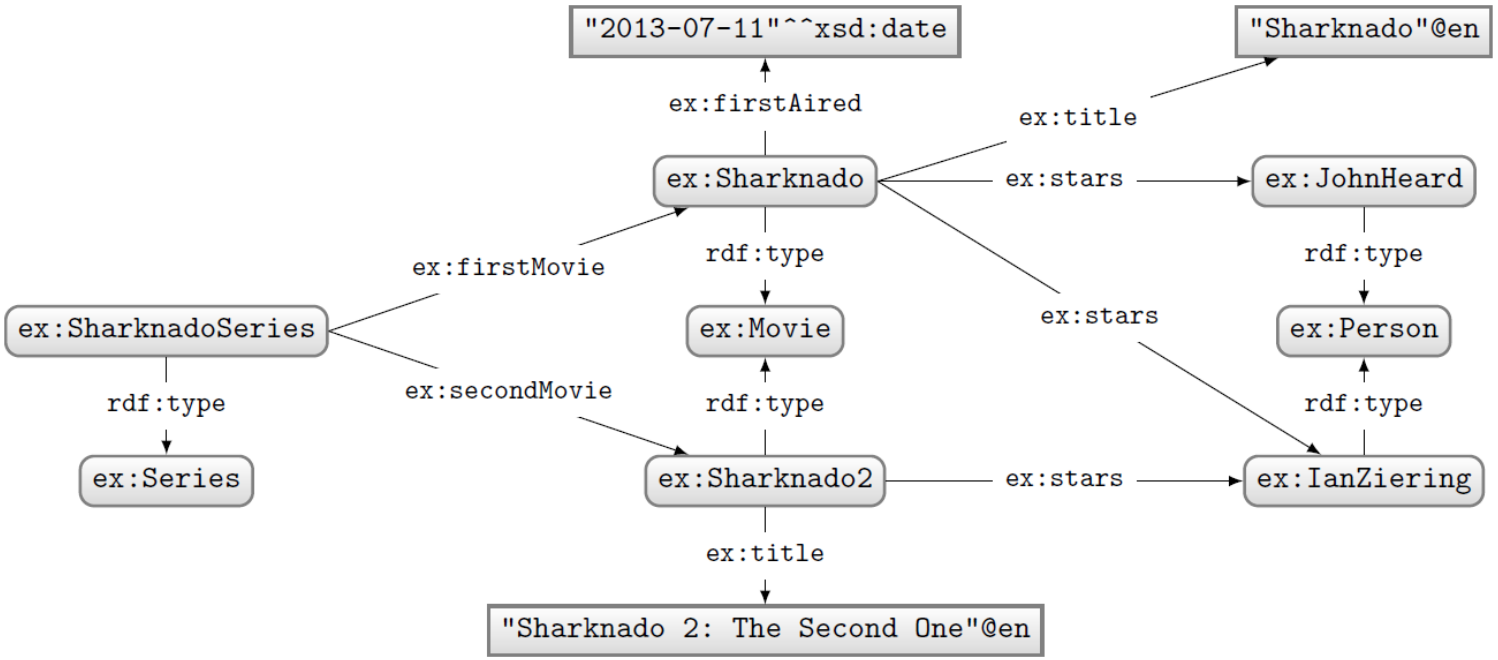
ex:Michael

ex:Vincent

ex:Mary

QUERY? SPARQL (1.1)

SPARQL 1.0 Query Features



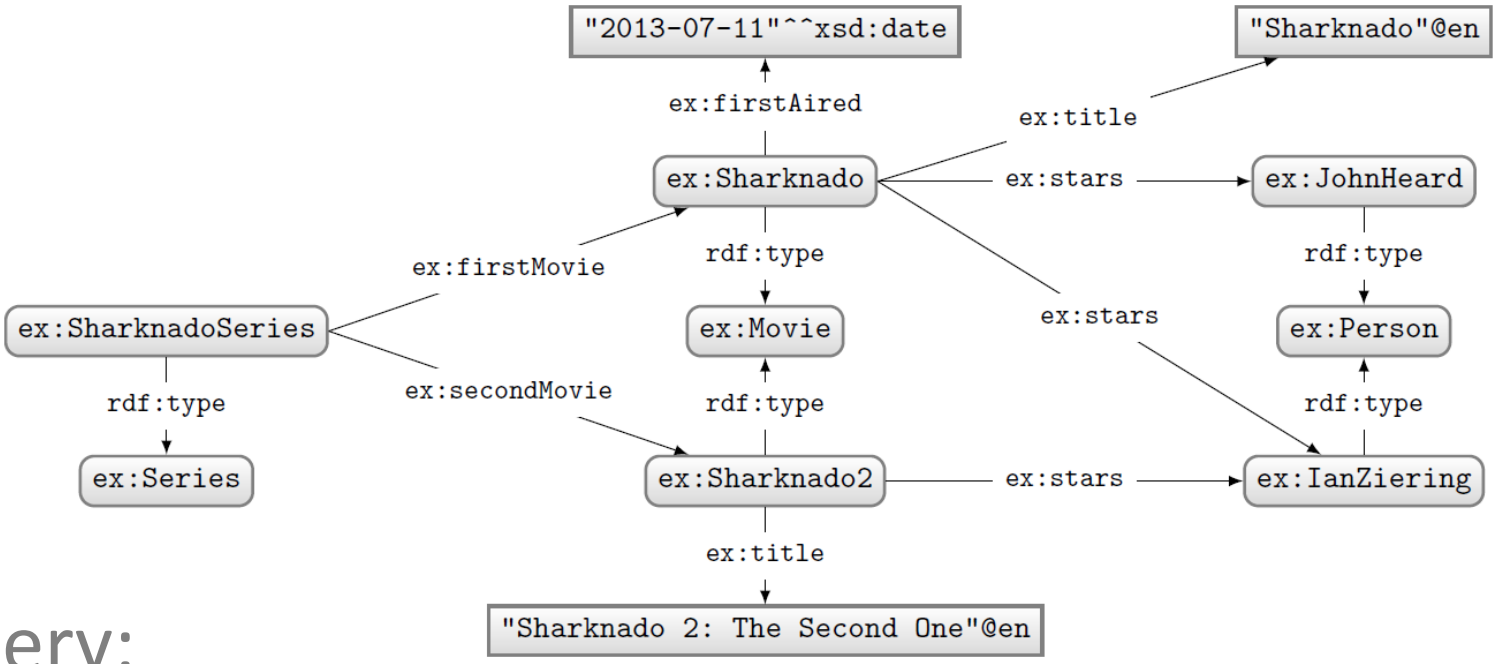
Query:

```
PREFIX ex: <http://ex.org/voc#>
SELECT *
WHERE {
  ex:Sharknado ex:stars ?star .
}
```

Solutions:

?star
ex:JohnHeard
ex:IanZiering

SPARQL 1.0 query features



Query:

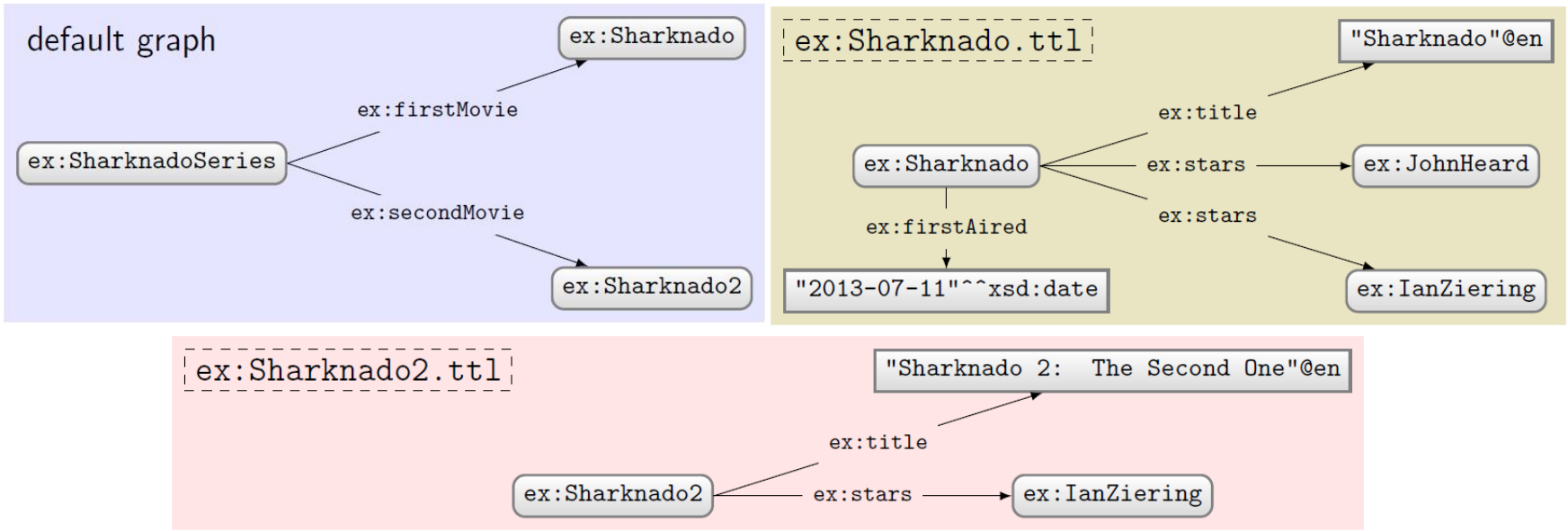
```
PREFIX ex: <http://ex.org/voc#>
SELECT *
WHERE {
  { ex:SharknadoSeries ex:firstMovie ?movie . }
  UNION
  { ex:SharknadoSeries ex:secondMovie ?movie . }
  OPTIONAL
  { ?movie ex:firstAired ?date . }
  ?movie ex:title ?title .
  FILTER(REGEX(STR(?title),"*[0-9]*"))
}
```

What solutions would this query return?

Solutions:

?movie	?title	?date
ex:Sharknado2	"Sharknado 2: The Second One"@en	

Named Graphs



Query:

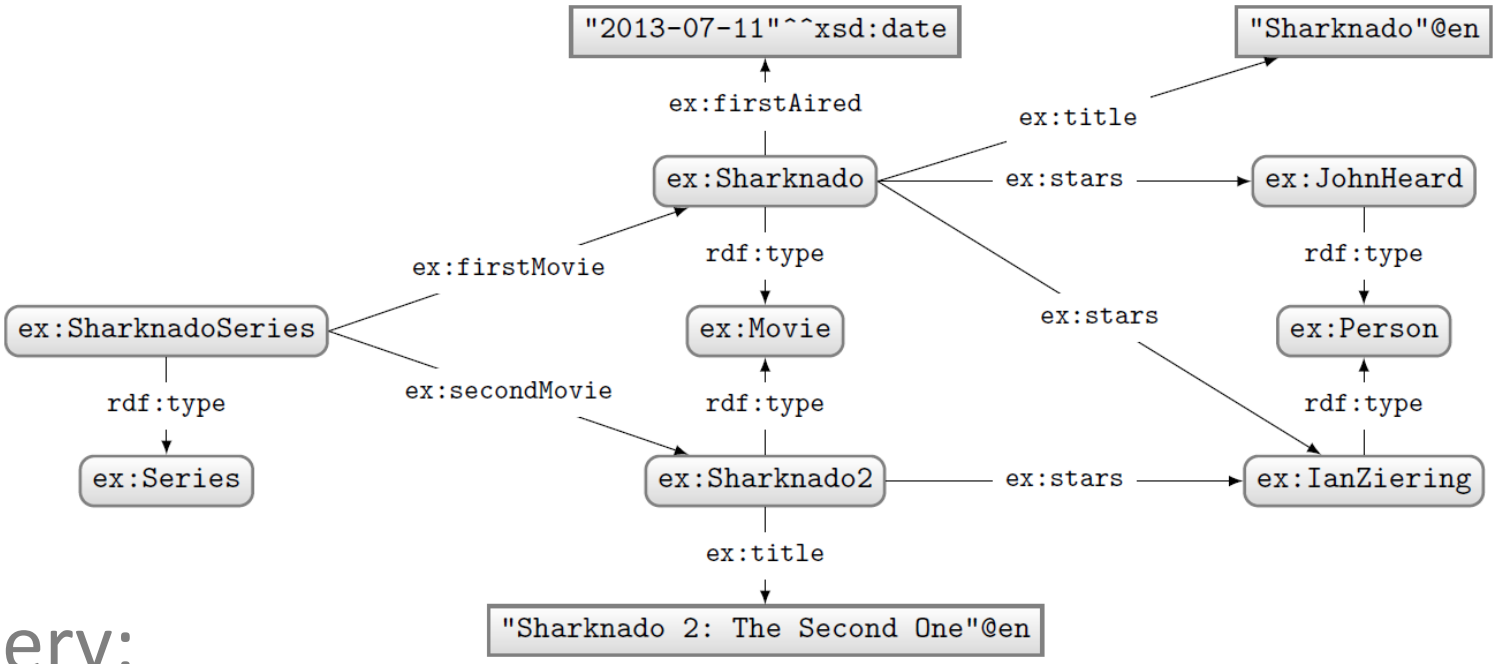
```
PREFIX ex: <http://ex.org/voc#>
FROM ex:Sharknado2.ttl
FROM NAMED ex:Sharknado.ttl
SELECT DISTINCT ?x ?q
WHERE {
  GRAPH ?g { ?s ?p ?o }
  ?x ?q ?o .
}
```

What solutions would this query return?

Solutions:

?x	?q
ex:Sharknado2	ex:stars

New SPARQL 1.1 Query Features



Query:

```
PREFIX ex: <http://ex.org/voc#>
SELECT (COUNT(?star) as ?count)
WHERE {
  ?movie ex:stars ?star .
}
```

Solutions:

?count
3

SPARQL 1.1 Federation

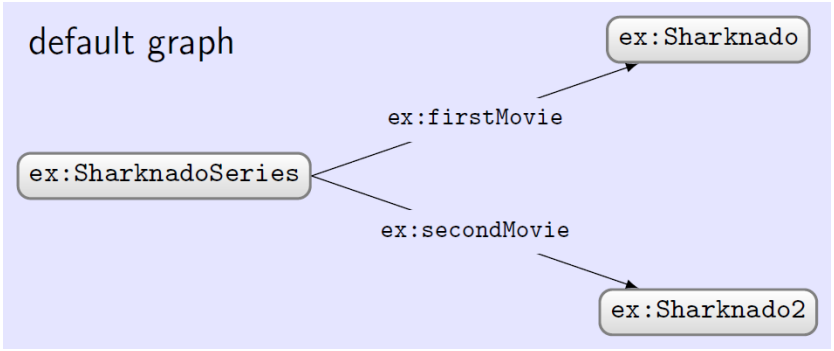
```
PREFIX movie: <http://data.linkedmdb.org/resource/movie/>
PREFIX dbpedia: <http://dbpedia.org/ontology/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?actor_name ?birth_date
FROM <http://dig.csail.mit.edu/2008/webdav/timbl/foaf.rdf> # placeholder graph
WHERE {
 {
  SERVICE <http://data.linkedmdb.org/sparql> {
    <http://data.linkedmdb.org/resource/film/675> movie:actor ?actor .
    ?actor movie:actor_name ?actor_name
  }
  BIND(STRLANG(?actor_name, "en") AS ?actor_name_en)
 }
  SERVICE <http://dbpedia.org/sparql> {
    ?actor2 a foaf:Person ; foaf:name ?actor_name_en ;
         dbpedia:birthDate ?birth_date .
  }
}
```

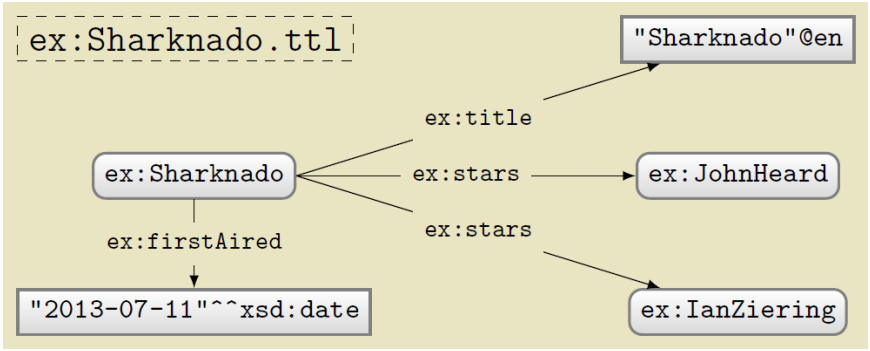
Get actors for Star Trek movie from LinkedMDB. Use DBpedia to get the birthdate of the actor

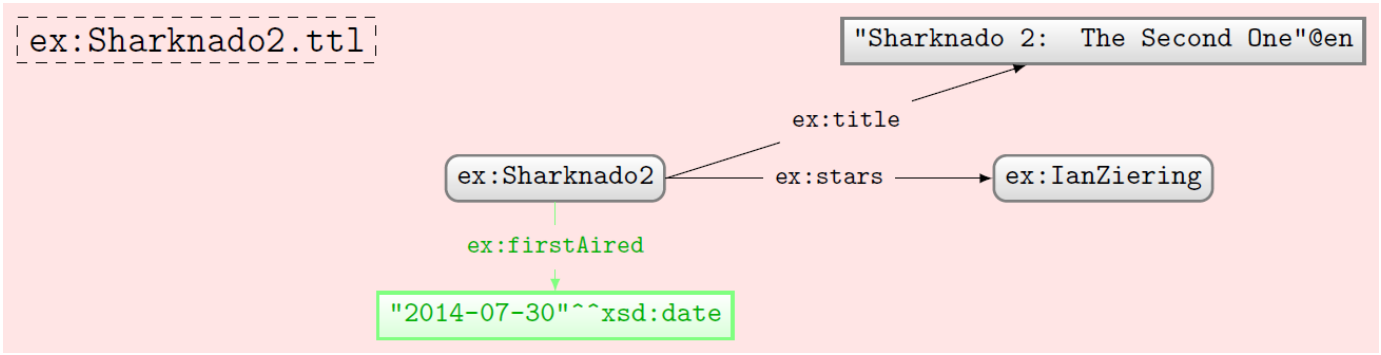
Can be run at http://sparql.org/sparql

Example borrowed from: http://www.cambridgesemantics.com/semantic-university/sparql-by-example

SPARQL 1.1 Update








```
PREFIX ex: <http://ex.org/voc#>
INSERT DATA {
  GRAPH ex:Sharknado2.ttl
    { ex:Sharknado2 ex:firstAired "2014-07-30"^^xsd:date . }
}
```

LINKS? LINKED DATA

IRIs link to other RDF documents …

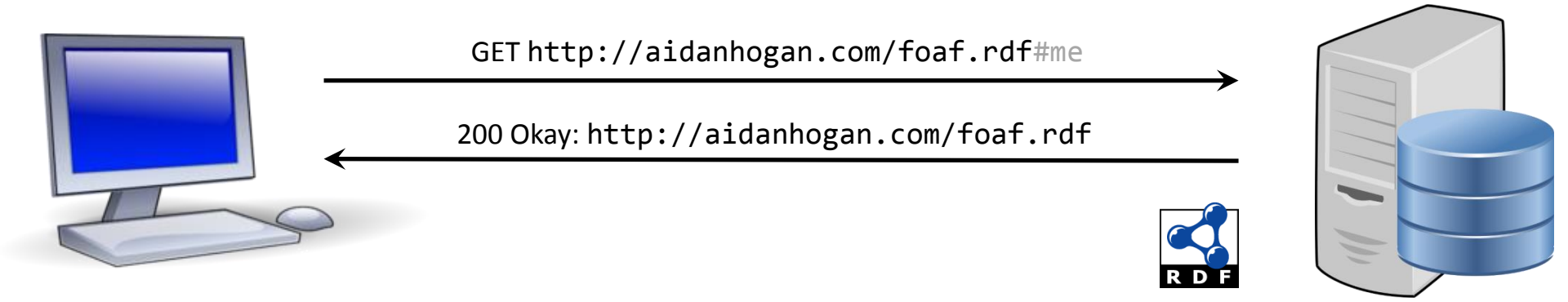
```
@base <http://anakena.dcc.uchile.cl/~calvarez/foaf.ttl> .

@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<#me> a foaf:Person;
        foaf:name "Camila Andrea Ãlvarez"@es;
        foaf:firstName "Camila"@es;
        foaf:surname "Ãlvarez"@es;
        foaf:birthday "02-24";
        foaf:gender "female";
        foaf:homepage <http://anakena.dcc.uchile.cl/~calvarez/cv.html>;
        foaf:based_near <http://sws.geonames.org/3871336/>;
        foaf:interest <http://dbpedia.org/resource/Tales_(series)>,
<http://dbpedia.org/resource/Embroidery>;
        foaf:img <http://images.evisos.cl/2009/06/03/erizo-de-tierra_9de6128c_3.jpg>;
        foaf:schoolHomepage <http://www.uchile.cl/>, <http://www.dcc.uchile.cl/>;
        foaf:knows <http://anakena.dcc.uchile.cl/~jasalas/web/foaf.ttl#me>,
<http://anakena.dcc.uchile.cl/~jogarrid/web/foaf.ttl#yo>,
<http://anakena.dcc.uchile.cl/~ekauffma/foaf.ttl#eli> .
```

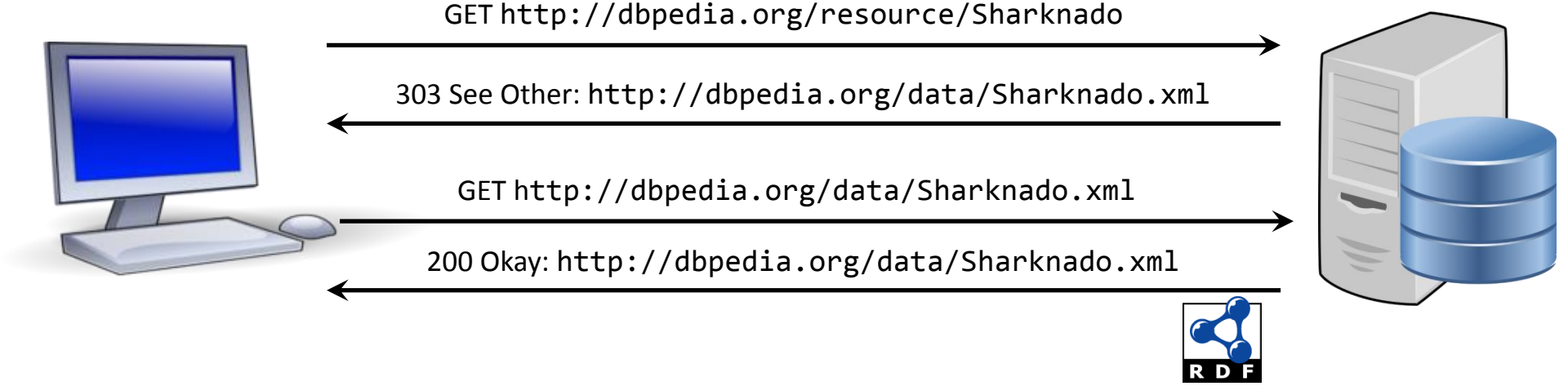
Four Principles of Linked Data

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.

Hash vs. Slash

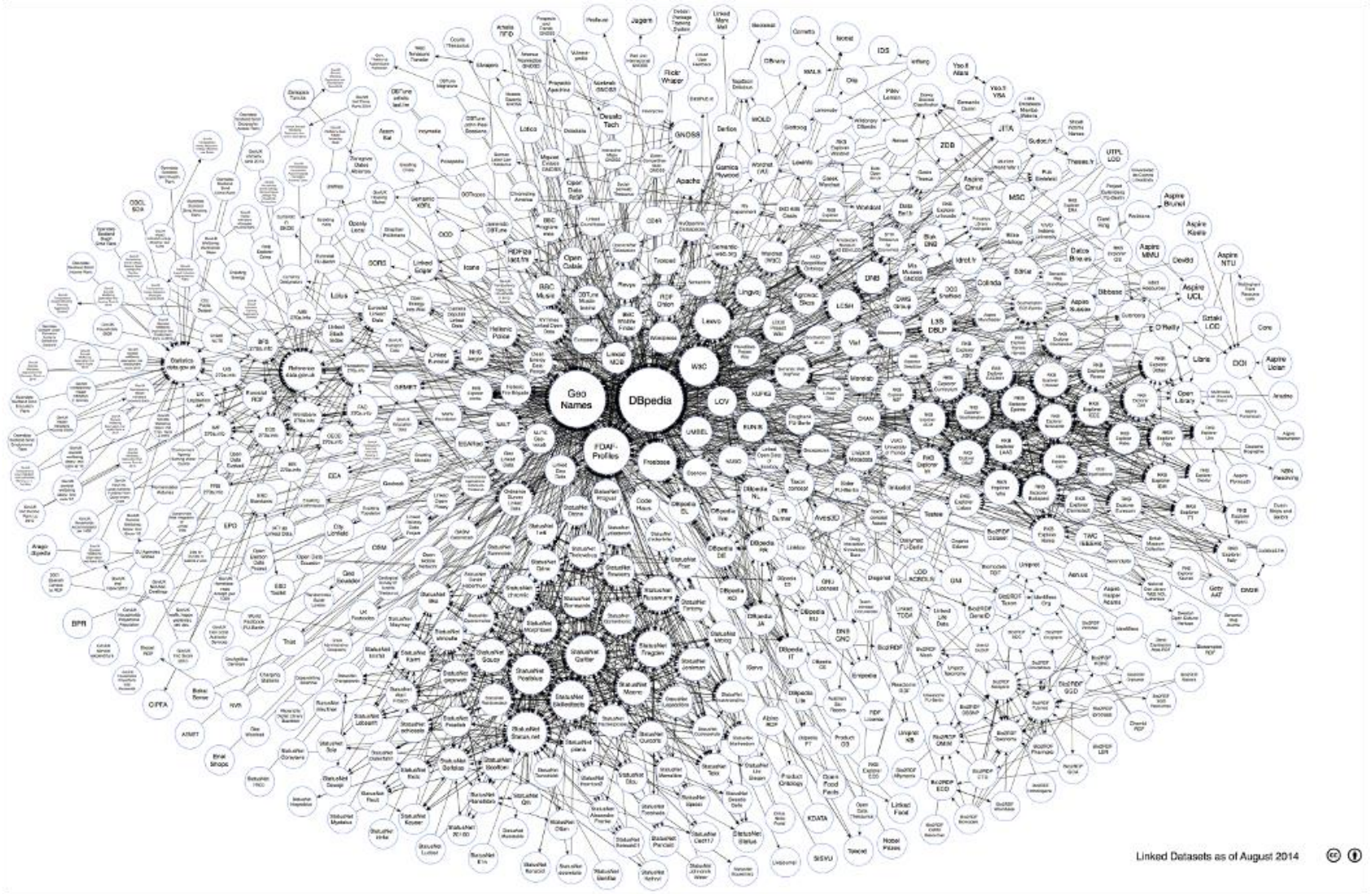




The 5 ★'s of Linked Open Data

- ★ Publish data under open licence
- ★★ Make the data "machine readable"
  - e.g., a Spreadsheet better than a PDF table
- ★★★ Use non-proprietary formats
  - e.g., a CSV text file better than Excel
- ★★★★ Use URIs to name your stuff (hint: RDF)
  - use unambiguous identifiers that can be linked/looked up
- ★★★★★ Provide links to other content
  - so consumers can follow links to find out more

Linked Data Cloud



Basics in place (with (4) Links!)

(1) Data, (2) Rules/Ontologies, (3) Query

INPUT: "$(x$,partOf,$y)$"

DATA:

http://ex.org/Ireland

Ireland 

(Ireland,partOf,Europe)
(Ireland,a,Country)
(Ireland,capital,Dublin)

http://ex.org/Dublin

Dublin 

(Dublin,population,1000000)

RULES: $(a, \text{capital}, b) \rightarrow (b, \text{partOf}, a)$
$(c, \text{partOf}, d), (d, \text{partOf}, e) \rightarrow (c, \text{partOf}, e)$

OUTPUT: $\{(x \mapsto \text{Ireland}, y \mapsto \text{Europe}), (x \mapsto \text{Dublin}, y \mapsto \text{Ireland})$
$(x \mapsto \text{Dublin}, y \mapsto \text{Europe})\}$


MANY OPEN RESEARCH QUESTIONS!

Who you gonna call …

Join us?

PROJECT …

Project, December 9th

- Presentation:
  - 5 minutes **strict**!
  - Idea, (source data), pipeline, methods used, example result, limitations/difficulties, lessons learned, (possible future work?)
- Code:
  - Submit code/ontology developed to u-cursos
- No report needed ☺

EXAM …

Exam, December 15th

- Four questions, best of three
  1. RDF (Lecture 2)
     - Good to know how to write Turtle
  2. RDFS/OWL (Lecture 3,4)
     - Lectures 5,6 not important ☺
  3. SPARQL (Lecture 7,8)
     - Will need to write (pseudo-)queries
       - *Minor* syntax errors not a problem but should know keywords and query structure
  4. Linked Data/Modelling (Lecture 9,2)
     - Modelling: given some facts/semantics in natural language, model them in RDF, RDFS, and OWL

Questions?

Thanks for taking the course!

- Hope it was interesting ☺
- First time teaching this … sorry for rough edges (e.g., problems in the lab)
- Any feedback?