SCHEMAS: RDFS + AN INTRODUCTION TO OWL

Schema information in RDF

- RDF allows to model the instance level of an ontology
- On the other hand, RDF provides very simple mechanisms to constrain the instance level through ontology schema information

Which means have you already discussed to specify schema constraints?

Schema information in RDF

- RDF allows to model the instance level of an ontology
- On the other hand, RDF provides very simple mechanisms to constrain the instance level through ontology schema information
  - we can specify the type of a resource of a predicate (rdf:type as property)
  - we can specify that a given predicate is indeed a property (rdf:Property as object)
- modeling complex domains requires more sophisticated ontology schema information

RDF Typing

- Languages for specifying
  - constraints over individuals (subjects, objects) and predicates used in RDF
  - i.e., an ontology *schema* an RDF graph is an instance of
- Two main proposals
  - RDF Schema (RDFS)
  - Web Ontology Language (OWL)
- Different expressive power
- Bringing the Semantics to the Semantic Web!

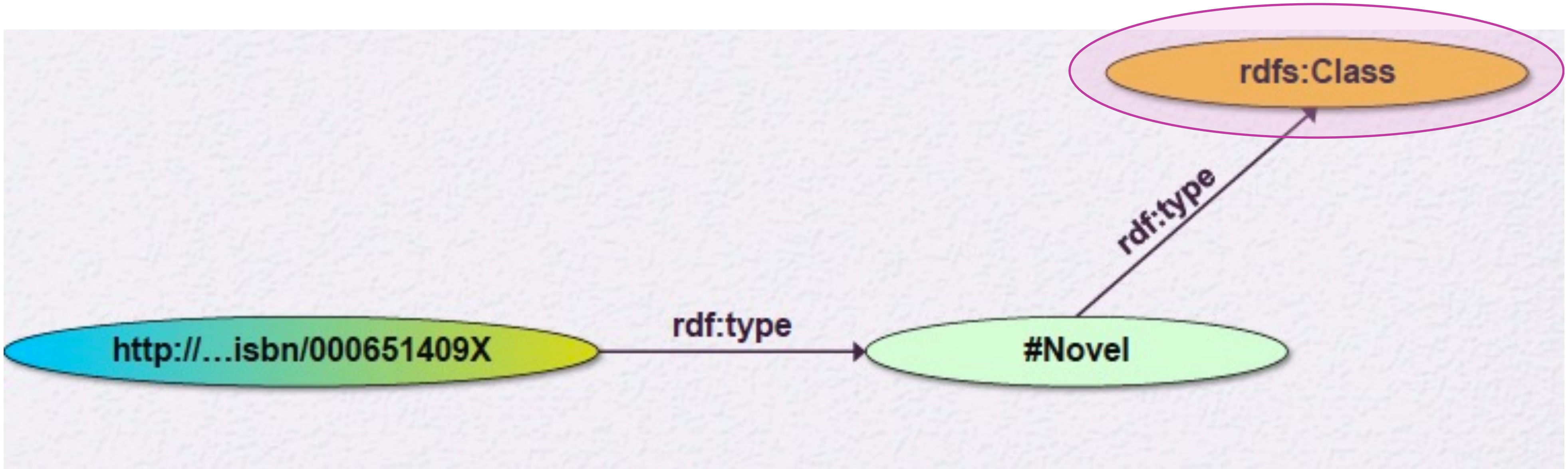
RDFS - RESOURCE DESCRIPTION FRAMEWORK SCHEMA

RDF Schema (RDFS)

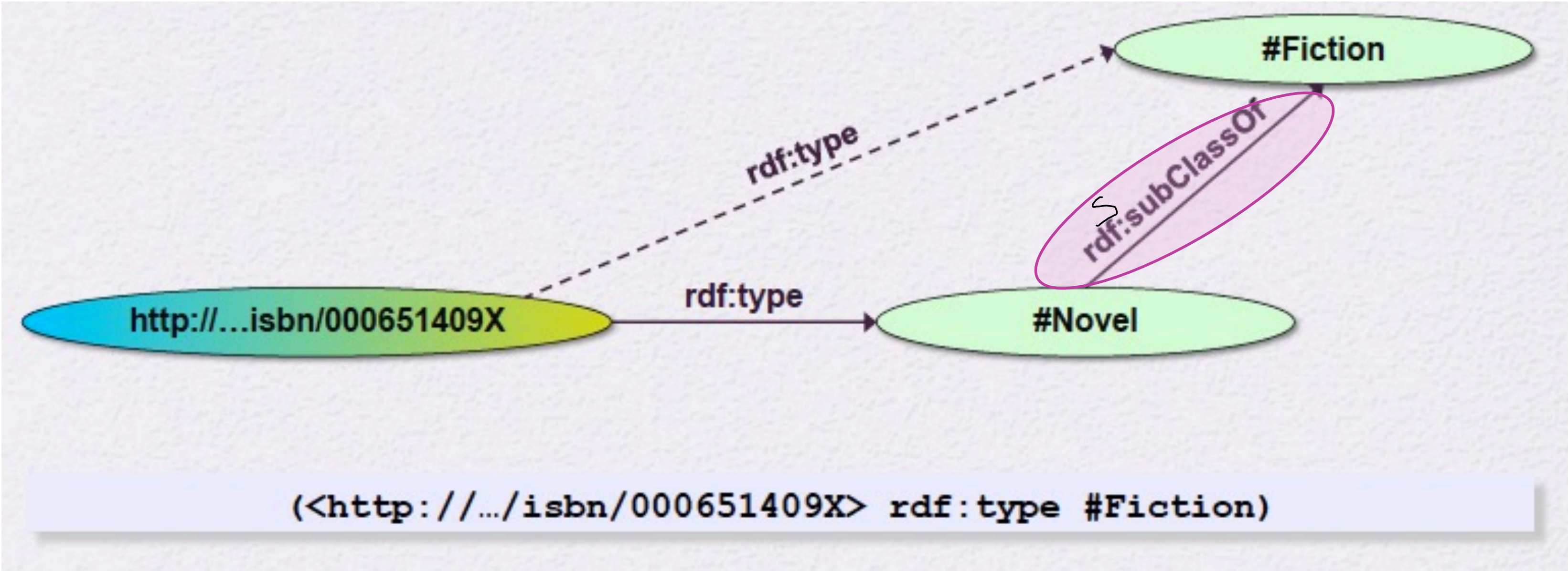
- First step towards the "extra knowledge"
- Vocabulary (defining terms)
  - use the term "novel" (a class)
- Schema (defining types)
  - "The Glass Palace" is a novel
    to be more precise: <http://.../000651409X> is a novel
- Taxonomy (defining hierarchies)
  - Any "novel" is a "fiction"

- RDFS defines resources and classes:
  - everything in RDF is a "resource"
  - "classes" are also resources, but…
  - …they are also a collection of possible resources (i.e., "individuals" "fiction", "novel", …)

Classes, resources in RDFS



Inferred properties IN RDFS



- is not in the original RDF data...
- ...but can be inferred from the RDFS rules
- RDFS environments return that triple, too

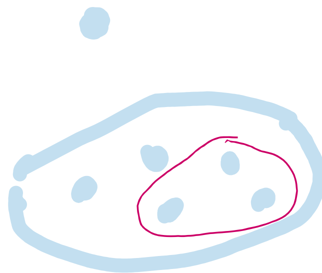
RDF Schema (RDFS)

- Type of constraints
  - [*objects and subjects as instances of certain classes*]
    → already in RDF, beyond the pure schema
  - inclusion statements between classes and between properties
  - semantic relations between the “domain”/the “range” of a property and some classes

- RDFS formalizes these notions as RDF triples, using some specific properties and objects

RDF Schema

- Declaration of instances (beyond the pure schema)
  - ⟨ Dupond rdf:type AcademicStaff ⟩



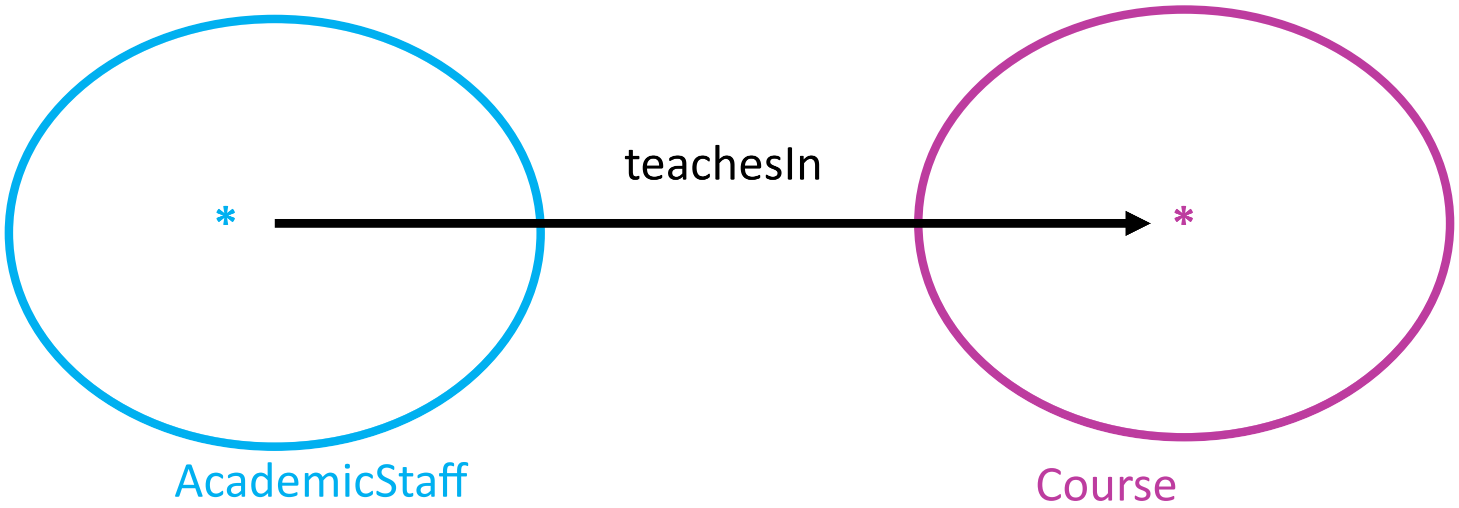
- Declaration of classes and subclass relationships
  - ⟨ Staff rdf:type rdfs:Class ⟩
  - ⟨ Java rdfs:subClassOf CSCourse ⟩
- Declaration of relations (properties in RDFS terminology)
  - ⟨ RegisteredTo rdf:type rdf:Property ⟩
- Declaration of subproperty relationships
  - ⟨ LateRegisteredTo rdfs:subPropertyOf RegisteredTo ⟩
- Declaration of domain and range restrictions for predicates
  - ⟨ TeachesIn rdfs:domain AcademicStaff ⟩ subject
  - ⟨ TeachesIn rdfs:range Course ⟩ object
  - TeachesIn(AcademicStaff, Course)

Domain and range

domain range

s, teachesIn, o

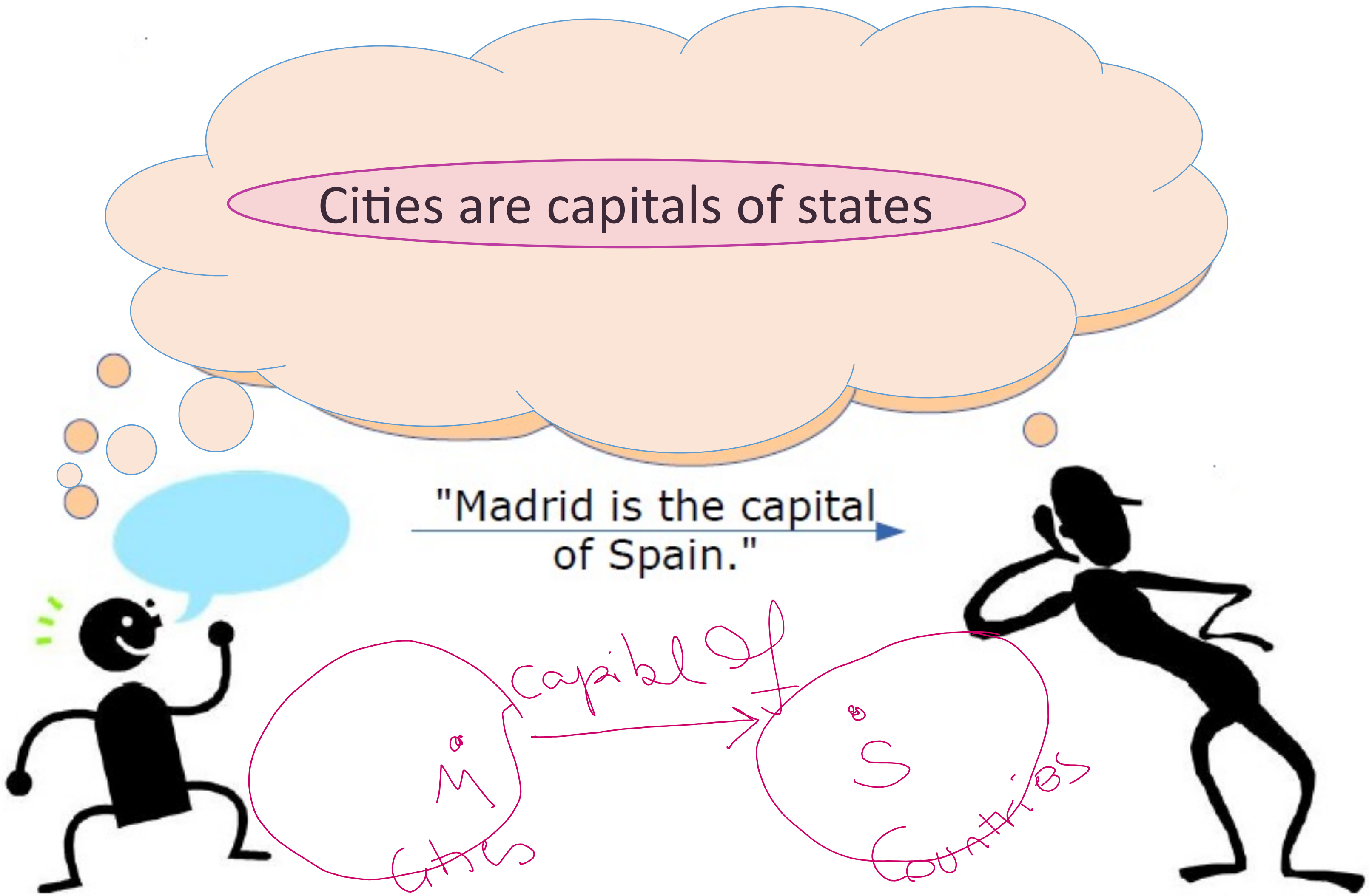
AcademicStaff Course



A very simple example

- Let's look at that sentence:
  - "Madrid is the capital of Spain."
- Published on the Semantic Web (i.e., using RDF):
  - :Madrid :capitalOf :Spain .
- How many pieces of information can we (i.e., humans) derive from that sentence?

A very simple example – adding simple semantics (schema information)



A very simple example

- States, cities, and capitals

a is an alternative syntax (qname, shorthand) for rdf:type

```
:State a rdfs:Class .
:City a rdfs:Class .
:locatedIn a rdf:Property .
:capitalOf rdfs:subPropertyOf :locatedIn .
:capitalOf rdfs:domain :City .
:capitalOf rdfs:range :State .
```

Definition of the Terminology (T-Box)

RDFS

```
:Madrid :capitalOf :Spain .
```

Definition of the Assertions (A-box)

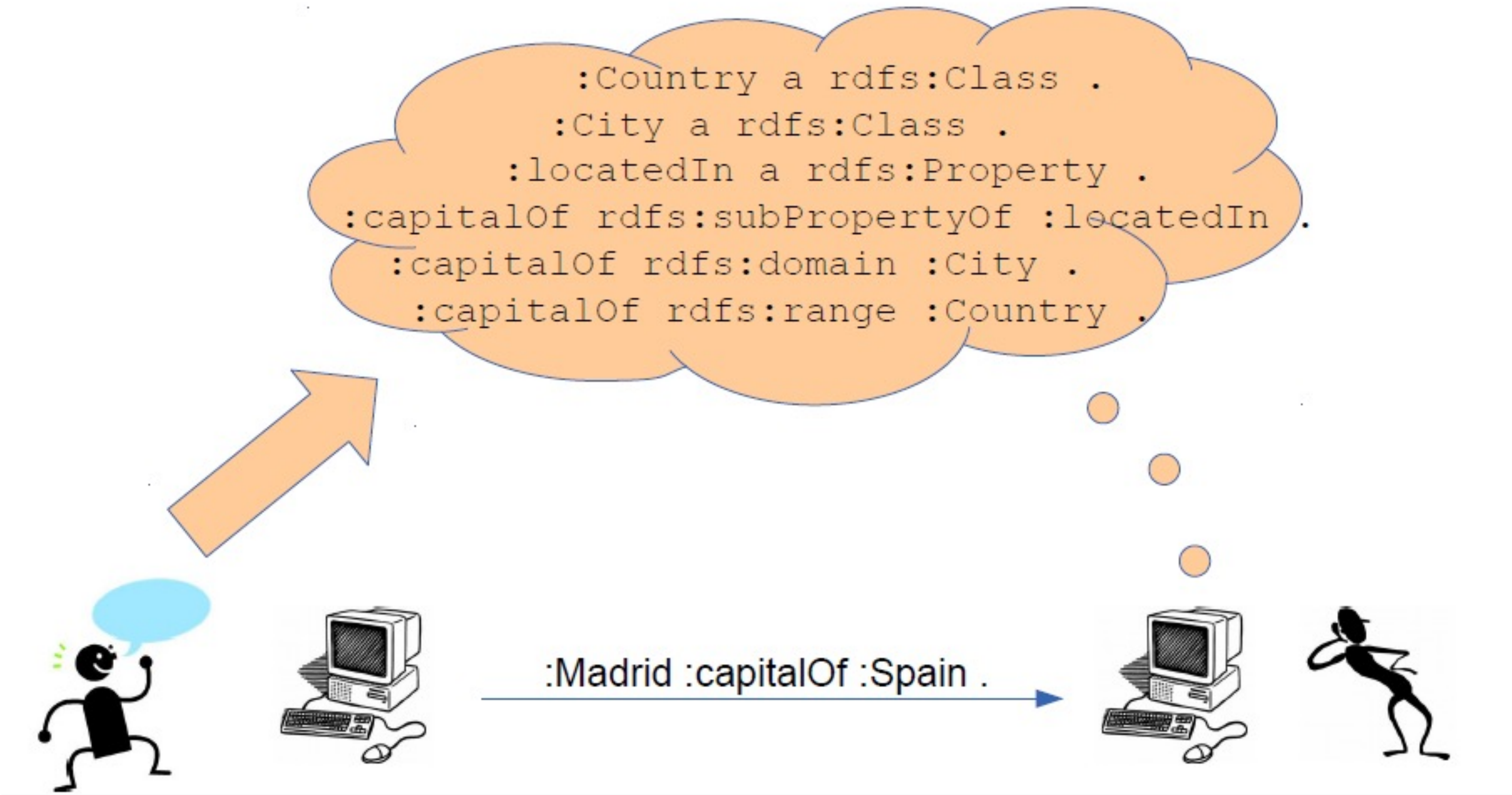
RDF

RDFS Logical Semantics

$\forall \ldots (\cdots \Rightarrow \ldots)$

RDF and RDFS statements	**FOL translation**
$\langle$ `i rdf:type C` $\rangle$	$C(i)$
$\langle$ `i P j` $\rangle$	$P(i,j)$
$\langle$ `C rdfs:subClassOf D` $\rangle$	$\forall X\,(C(X) \Rightarrow D(X))$
$\langle$ `P rdfs:subPropertyOf R` $\rangle$	$\forall X \forall Y\,(P(X,Y) \Rightarrow R(X,Y))$
$\langle$ `P rdfs:domain C` $\rangle$	$\forall X \forall Y\,(P(X,Y) \Rightarrow C(X))$
$\langle$ `P rdfs:range D` $\rangle$	$\forall X \forall Y\,(P(X,Y) \Rightarrow D(Y))$

What do we gain now?



What do we gain now?

$\langle \texttt{P rdfs:domain C} \rangle$ | $\forall X \forall Y (P(X,Y) \Rightarrow C(X))$ |

```
  :Madrid :capitalOf :Spain .
+ :capitalOf rdfs:domain :City
→ :Madrid a :City .
```

$\langle \texttt{P rdfs:range D} \rangle$ | $\forall X \forall Y (P(X,Y) \Rightarrow D(Y))$ |

```
  :Madrid :capitalOf :Spain .
+ :capitalOf rdfs:range:Country
→ :Spain a :Country .
```

```
  :Madrid :capitalOf :Spain .
+ :capitalOf rdfs:subPropertyOf :locatedIn .
→ :Madrid :locatedIn :Spain .
```

$\langle \texttt{P rdfs:subPropertyOf R} \rangle$ | $\forall X \forall Y (P(X,Y) \Rightarrow R(X,Y))$ |

RDFS Operational Semantics

- The logical formulas representing the semantics of RDFS are very useful in practice and are very adapted to deductive reasoning on RDF
- This means that
  - given facts and rules
  - we can derive new facts
- The corresponding tools are called reasoners
- The logical formulas can be interpreted as rules (tuple generating dependencies) that may be thought of as a factory for generating new facts

RDFS Operational Semantics

- Deduction rules are an interpretation function
- Simple reasoning algorithm (a.k.a. *forward chaining*):

```
Given: an RDF Graph G
a set of deduction rules R
Entailment E = G
Repeat
        M := { }
        For all rules in R
                For each statement S in G
                        Apply R to S
                        If E does not contain consequence
                                Add consequence to M
        Add all elements in M to E
bis M = { }
```

RFDS Operational semantics: Example

- **if** < r rdf:type A > and < A rdfs:subClassOf B >
  then < r rdf:type B >

  where r, A, and B are variables

- This means that:
  - **if** we know two triplets matching the patterns
    - < r rdf:type A > and
    - < A rdfs:subClassOf B >

    for some values of r, A, B, **then**
  - we can infer the triplet < r rdf:type B > with the values of r, B taken to be those of the match

RDFS Operational Semantics

1. **if** < r rdf:type A > and < A rdfs:subClassOf B >
   then < r rdf:type B >

2. **if** < r P s > **and** < P rdfs:subPropertyOf Q >
   then < r Q s >

3. **if** < P rdfs:domain C > **and** < x P y >
   then < x rdf:type C >

4. **if** < P rdfs:range D > **and** < x P y >
   then < y rdf:type D >

A very simple example

- States, cities, and capitals

```
:State a rdfs:Class .
:City a rdfs:Class .
:locatedIn a rdf:Property .
:capitalOf rdfs:subPropertyOf :locatedIn .
:capitalOf rdfs:domain :City .
:capitalOf rdfs:range :State .

:Madrid :capitalOf :Spain .
```

Definition of the Terminology (T-Box)

RDFS

Definition of the Assertions (A-box)

RDF

:Madrid a :City
:Spain a :Country

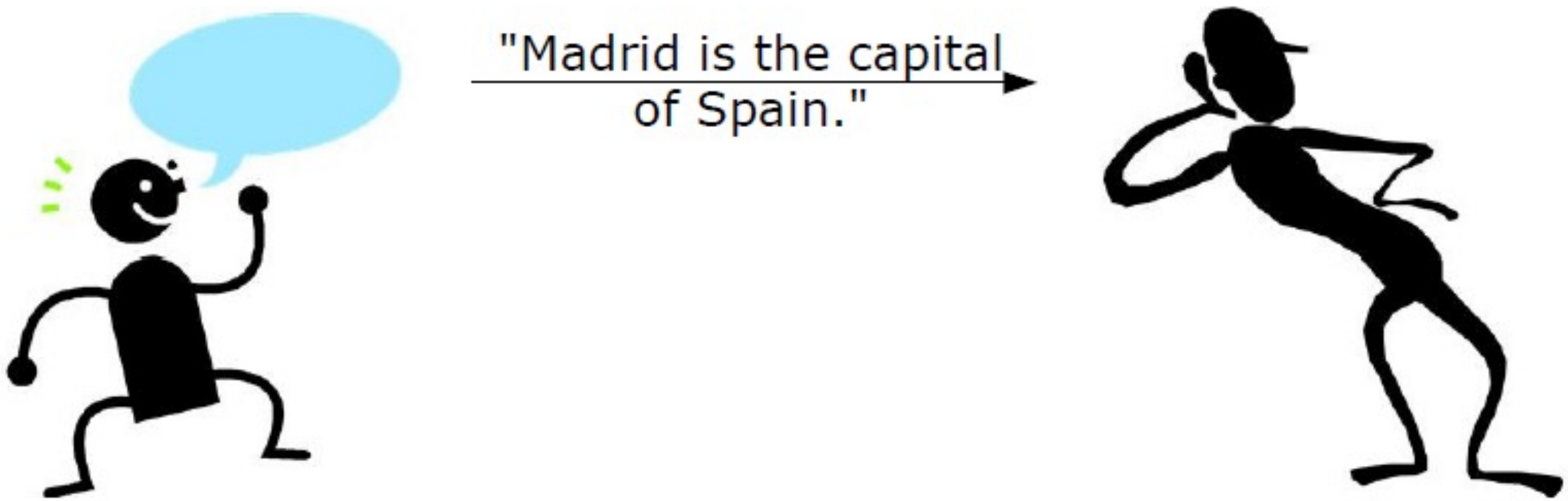
:Madrid :locatedIn :Spain

Inferred Assertions

OWL
WEB ONTOLOGY LANGUAGE

What is missing up to now?

- Our mission: make computers understand information on the Web
- But what does *understand* actually mean?



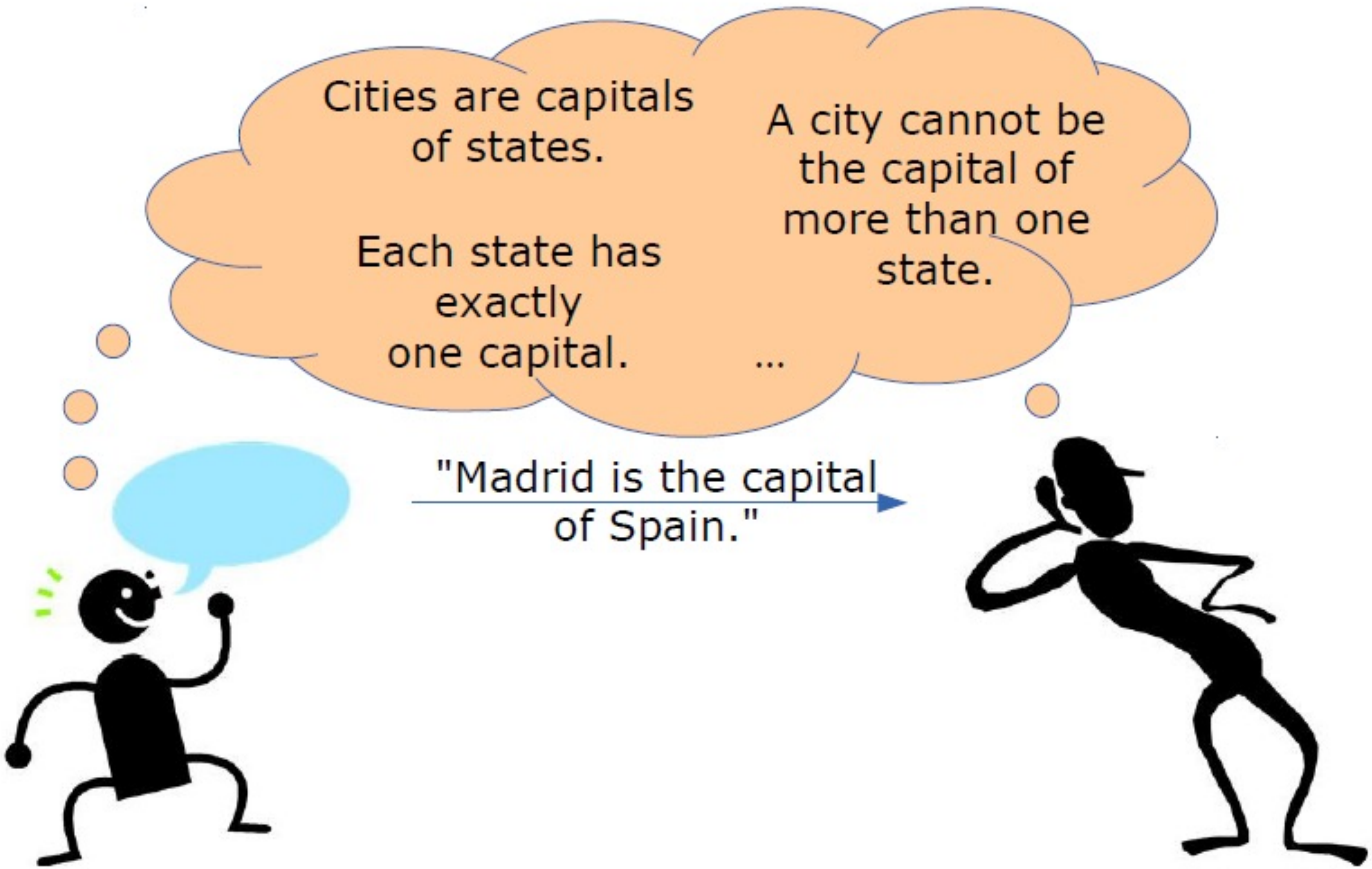
Semantics

- Let's look at that sentence:
  - "Madrid is the capital of Spain."

- Published on the Semantic Web (i.e., using RDF):
  - :Madrid :capitalOf :Spain .

- How many pieces of information can we (i.e., humans) derive from that sentence?
  - (1 piece of information = 1 statement <S,P,O>)

Semantics – How does it work?



Semantics

- Let's look at that sentence:
  - "Madrid is the capital of Spain."

- We can get the following information:
  - "Madrid is the capital of Spain."
  - "Spain is a state."
  - "Madrid is a city."
  - "Madrid is located in Spain."
  - "Barcelona is not the capital of Spain."
  - "Madrid is not the capital of France."
  - "Madrid is not a state."
  - ...

What have we gained?

- Let's look at that sentence:
  - "Madrid is the capital of Spain."

- We can get the following information:
  - "Madrid is the capital of Spain." ✔
  - "Spain is a state." ✔
  - "Madrid is a city." ✔
  - "Madrid is located in Spain." ✔
  - "Barcelona is not the capital of Spain." ✖
  - "Madrid is not the capital of France." ✖
  - "Madrid is not a state." ✖
  - ...

What we cannot express (up to now)?

- "Every state has *exactly one* capital"
  - Property cardinalities
- "Every city can only be the capital of one state."
  - Functional properties
- "A city cannot be a state at the same time."
  - Class disjointness
- ...

- For those, we need more expressive languages than RDFS!

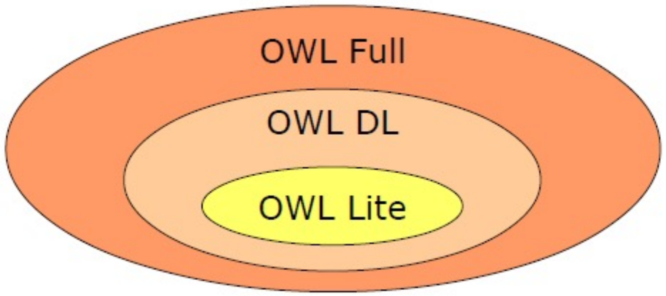
Expressive Ontologies using OWL

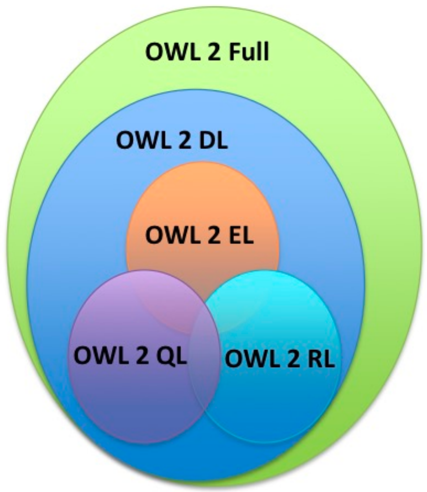
OWL extends RDFS with the possibility to express additional constraints

OWL was designed to find a reasonable balance between

- expressivity of the language and
- efficient reasoning, i.e. scalability

This also motivates the different sublanguages





Expressive Ontologies using OWL

- "Barcelona is not the capital of Spain." ✖
- Why not?
  - Countries have exactly one capital
  - Barcelona and Madrid are not the same

- In OWL:

```
:capitalOf a owl:InverseFunctionalProperty .
:Madrid :capitalOf :Spain .
:Madrid owl:differentFrom :Barcelona .

ASK { :Barcelona :capitalOf :Spain . } → false
```

Expressive Ontologies using OWL

- "Madrid is not the capital of France." ✖
- Why not?
  - A city can only be the capital of one country
  - Spain and France are not the same
- Also:

```
:capitalOf a owl:FunctionalProperty .
:Madrid :capitalOf :Spain .
:Spain owl:differentFrom :France .
```

```
ASK { :Madrid :capitalOf :France . } → false
```

Expressive ontologies using OWL

- Madrid is not a state
- Why not?
  - A capital is a city and a city cannot be a state (states and cities are disjoint)

```
:Madrid :capitalOf :Spain .
:capitalOf rdfs:domain :City .
:City owl:disjointWith :State .
```

```
ASK {:Madrid :a :State .} -> false
```

What is OWL?

- Standard language (W3C) for representing vocabularies/ ontologies/schemas
- Much richer than RDF Schema and SKOS (W3C recommendations for light-weight vocabularies)
- Original OWL
  - *Published as W3C recommendation on 10.2.2004*
- OWL 2
  - *Latest W3C recommendation on 11.12.2012*
    Extends and replaces the old recommendation

OWL 2 Syntaxes

Turtle

```
ex:JohnSmith rdf:type ex:Person .
```

RDF/XML

```
<ex:Person rdf:about="#JohnSmith"/>
```

OWL/XML

```
<ClassAssertion>
    <Class IRI="Person" />
    <NamedIndividual IRI="JohnSmith" />
</ClassAssertion>
```

Functional-style

```
ClassAssertion( :Person :JohnSmith )
```

Manchester

```
Individual: JohnSmith
    Types: Person
```

Many examples, translated into all syntaxes:
OWL 2 Web Ontology Language: Primer
http://www.w3.org/TR/owl2-primer/

«Schema knowledge» in RDFS

rdf:type	rdf:Property
rdfs:Class	
rdfs:range	rdfs:domain
rdfs:subClassOf	rdfs:subPropertyOf

«Schema knowledge» in RDFS

```
ex:isMarriedTo rdfs:domain ex:Person .
ex:isMarriedTo rdfs:range ex:Person .
ex:instituteAIFB rdf:type ex:Institution
```

```
ex:pascal ex:isMarriedTo ex:instituteAIFB
```

- It likely is a modeling flaw
- It would result in rejecting the insertion in a database
- In RDFS it results in inferring

```
ex:instituteAIFB rdf:type ex:Person
```

- Multiple class membership is allowed
- No inconsistency

RDFS (and OWL) Semantics

- Open World Assumption The absence of a triple in a graph does not imply that the corresponding statement does not hold
- The fact that a statement is not true cannot be described in RDFS

- No Unique Name Assumption: differently named individuals can denote the same thing

The OWL View of Life

- OWL is not like a database system
- no requirement that the only properties of an individual are those mentioned in a class it belongs to
- no assumption that everything is known
- classes and properties can have multiple “definitions”
- statements about individuals need not be together (in the same document)

OWL Constructs

- Class (&instance) relations
- Property relations & characteristics
- Combining classes (set opns)
- Property restrictions and intensional classes

Class Relations

- Disjointness between classes:

OWL notation	FOL translation
⟨ C owl:disjointWith D ⟩	$\forall X\,(C(X) \Rightarrow \neg D(X))$

ex:isMarriedTo rdfs:domain ex:Person .
ex:isMarriedTo rdfs:range ex:Person .
ex:instituteAIFB rdf:type ex:Institution .
ex:Institution owl:disjointWith ex:Person

ex:pascal ex:isMarriedTo ex:instituteAIFB

ex:instituteAIFB rdf:type ex:Person

inconsistency

Class Relations
(back to our initial example)

- Madrid is not a state
- Why not?
  - A capital is a city and a city cannot be a state (states and cities are disjoint)

```
:Madrid :capitalOf :Spain .
:capitalOf rdfs:domain :City .
:City owl:disjointWith :State .
```

```
ASK {:Madrid :a :State .} -> false
```

Other class & instance relations

- owl:equivalentClass

- owl:sameAs
- owl:differentFrom

Needed because of lack of
Unique Name Assumption

Property Relations

- Constraints of functionality and symmetry on predicates:

OWL notation	FOL translation
⟨ P rdf:type owl:FunctionalProperty ⟩	$\forall X \forall Y \forall Z$ $(P(X,Y) \wedge P(X,Z) \Rightarrow Y = Z)$
⟨ P rdf:type owl:InverseFunctionalProperty ⟩	$\forall X \forall Y \forall Z$ $(P(X,Y) \wedge P(Z,Y) \Rightarrow X = Z)$
⟨ P owl:inverseOf Q ⟩	$\forall X \forall Y\, (P(X,Y) \Leftrightarrow Q(Y,X))$
⟨ P rdf:type owl:SymmetricProperty ⟩	$\forall X \forall Y\, (P(X,Y) \Rightarrow P(Y,X))$

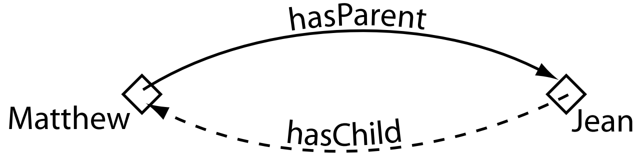
owl:TransitiveProperty

…

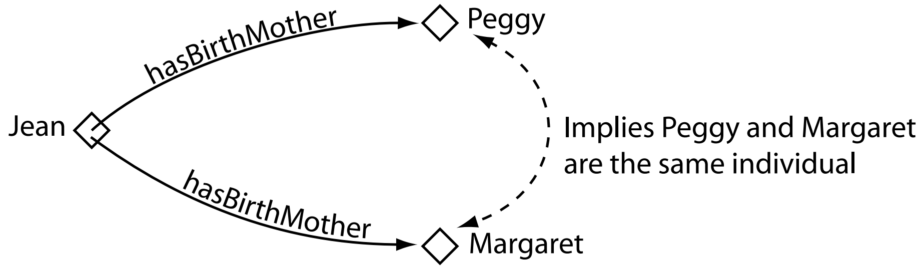
owl:equivalentProperty
owl:propertyDisjointWith

Property Relations

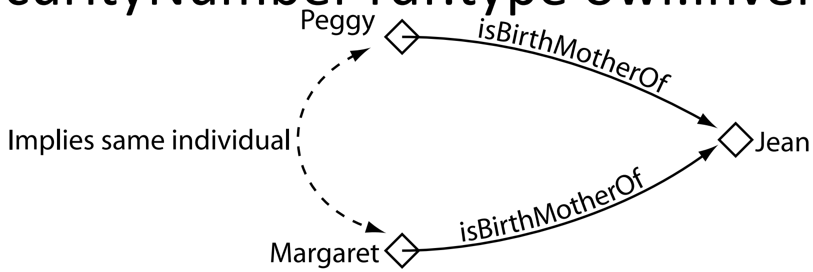
- hasParent owl:inverseOf hasChild



- hasBirthMother rdf:type owl:FunctionalProperty



- hasSocialSecurityNumber rdf:type owl:InverseFunctionalProperty

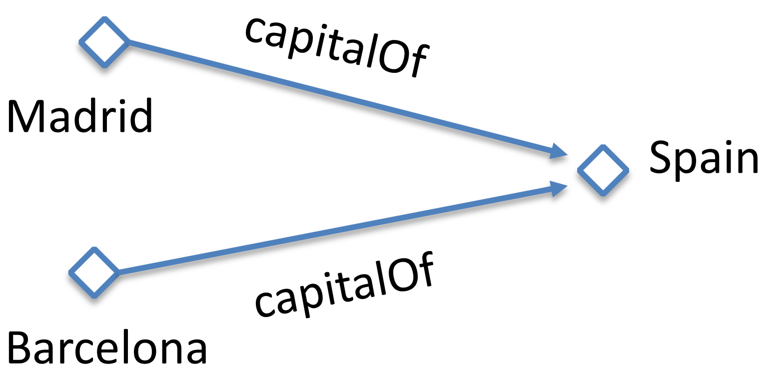


Property Relations (back to our initial example)

- "Barcelona is not the capital of Spain." ✗
- Why not?
  - Countries have exactly one capital
  - Barcelona and Madrid are not the same
- In OWL:

```
:capitalOf a owl:InverseFunctionalProperty .
:Madrid :capitalOf :Spain .
:Madrid owl:differentFrom :Barcelona .

ASK { :Barcelona :capitalOf :Spain . } → false
```



inverseFunctionality leads to the fact that any city capital of Spain is the same thing as Madrid (Barcelona = Madrid)
If we add owl:differentFrom we get an inconsistency

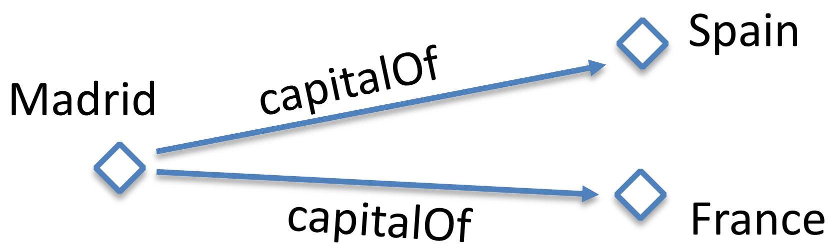
Property Relations (back to our initial example)

- "Madrid is not the capital of France." ✖
- Why not?
  - A city can only be the capital of one country
  - Spain and France are not the same

- Also:

```
:capitalOf a owl:FunctionalProperty .
:Madrid :capitalOf :Spain .
:Spain owl:differentFrom :France .

ASK { :Madrid :capitalOf :France . } → false
```



Functionality leads to
the fact that any country which capital is Madrid is the same thing as Spain (France = Spain)
If we add owl:differentFrom we get an inconsistency

Key Notions

- An ontology is a clear, unambiguous, formal model of some part of the real world that is shared by several people
- OWL can be used to describe ontologies
- OWL adds to RDFS the possibility to
  - clarify that two individuals are identical or different
  - express that two classes are equivalent or disjoint
  - declare a property to be the inverse of another property
  - construct new classes by taking the union, intersection or complement of other existing classes
  - define classes by specifying restrictions on the cardinality or possible values of properties
- OWL is based on description logics, which makes its semantics machine accessible

Suggested Reading

- Pascal Hitzler, Markus Krötzsch and Sebastian Rudolph. Foundations of Semantic Web Technologies. Chapman & Hall/CRC, 2009. (Chapter 4)

- Matthew Horridge. A Practical Guide To Building OWL Ontologies Using Protégé 4 and CO-ODE Tools. Edition 1.3.

- Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter F. Patel-Schneider, Sebastian Rudolph. OWL 2 Web Ontology Language Primer (Second Edition), 2012