Semantic Data

Practice 2 : RDF/RDFS

Jean-Louis Binot

Types of exercices

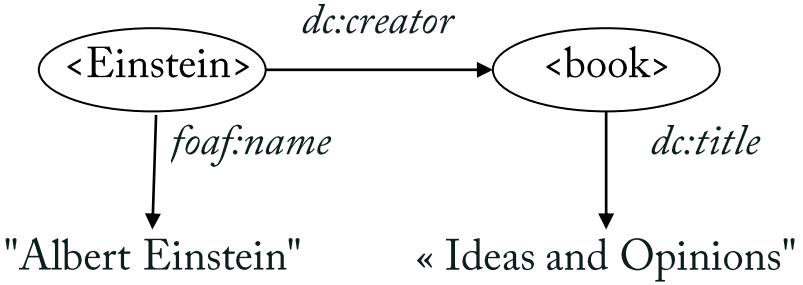
1. Manipulating the RDF Turtle language.
   - Find triples from Turtle, express natural language information in Turtle…
2. Transform triples in a labeled graph and vice-versa.
3. Express a model in RDFS (Turtle syntax).
4. Applying RDFS inference rules.

Note : useful validation services.

- https://www.w3.org/RDF/Validator/: RDF/XML validator from W3C (RDF/XML syntax only).
- http://rdfvalidator.mybluemix.net/: validator and converter between RDF/XML, Turtle and JSON LD.

1. Manipulating the RDF Turtle language.

- **Exercise 1** : express in valid Turtle : *Albert Einstein is the author of Ideas and Opinions.*
  - Use Dublin Core and Foaf properties. Find URIs for Einstein and his book. Add Einstein's name and the book's title.
  - How many triples will you need ?
  - **Answer: 3.**



PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>

<https://fr.wikipedia.org/wiki/Albert_Einstein> dc:creator
<https://www.goodreads.com/book/show/497195.Ideas_and_Opinions> .

<https://fr.wikipedia.org/wiki/Albert_Einstein> foaf:name "Albert Einstein" .

<https://www.goodreads.com/book/show/497195.Ideas_and_Opinions> dc:title "Ideas and opinions" .

Triples of the Data Model

Number	Subject	Predicate	Object
1	https://fr.wikipedia.org/wiki/Albert_Einstein	http://xmlns.com/foaf/0.1/name	"Albert Einstein"
2	https://fr.wikipedia.org/wiki/Albert_Einstein	http://purl.org/dc/elements/1.1/creator	https://www.goodreads.com/book/show/497195.Ideas_and_Opinions
3	https://www.goodreads.com/book/show/497195.Ideas_and_Opinions	http://purl.org/dc/elements/1.1/title	"Ideas and opinions"

1. Manipulating the RDF Turtle language.

- **Exercise 2 :** Simplify the syntax of the code of the previous exercise by
  - Using a default namespace for Wikipedia;
  - Using syntactic sugar where possible.

```
PREFIX : <https://fr.wikipedia.org/wiki/>
PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX foaf:  <http://xmlns.com/foaf/0.1/>
PREFIX dc:    <http://purl.org/dc/elements/1.1/>

:Albert_Einstein
      dc:creator  <https://www.goodreads.com/book/show/497195.Ideas_and_Opinions> ;
      foaf:name   "Albert Einstein" .
<https://www.goodreads.com/book/show/497195.Ideas_and_Opinions> dc:title "Ideas and opinions" .
```

1. Manipulating the RDF Turtle language.

Exercise 3 : how many triples are there in the following document ? What does it express ?

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rel: <http://purl.org/vocab/relationship/>
PREFIX univ: <http://www.example.org/univ#>

<https://www.uliege.be/cms/c_10548125/fr/jean-louis-binot> a foaf:Person;
        foaf:name "Jean-Louis Binot"@en, " Jean-Louis Binot"@fr ;
        rel:collaboratesWith [
        univ:professor [rdfs:label "Integrated Software Project"@en ] ] .
```

Jean-Louis Binot is a person whose name is "Jean-Louis Binot" in English and in French, and who collaborates with someone who is a university professor of something named the "Integrated Software Project".

Answer : 6

Number	Subject	Predicate	Object
1	https://www.uliege.be/cms/c_10548125/fr/jean-louis-binot	http://purl.org/vocab/relationship/collaboratesWith	genid:UA0
2	https://www.uliege.be/cms/c_10548125/fr/jean-louis-binot	http://xmlns.com/foaf/0.1/name	"Jean-Louis Binot"@fr
3	https://www.uliege.be/cms/c_10548125/fr/jean-louis-binot	http://xmlns.com/foaf/0.1/name	"Jean-Louis Binot"@en
4	https://www.uliege.be/cms/c_10548125/fr/jean-louis-binot	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://xmlns.com/foaf/0.1/Person
5	genid:UA1	http://www.w3.org/2000/01/rdf-schema#label	"Integrated Software Project"@en
6	genid:UA0	http://www.example.org/univ#professor	genid:UA1

1. Manipulating the RDF Turtle language.

- **Exercise 4 :** is it possible to express in RDF : *Jean has not married Marie* ?

Answer : no, because RDF does not directly support negation.

One might think to express the triple *ex:Jean ex:notMarry ex:Marie*, but the property *notMarry* would not have any semantic relationship with the property *Marry*.

Note: it is however possible in Owl.

1. Manipulating the RDF Turtle language.

- **Exercise 5a:** is it possible to express in Turtle: *Jean and Marie know the same (unknown) person* ?

  - Answer : yes. To represent an unknown entity, one uses a blank node, and a blank node can appear in object position in a triple.
    How many triples do we need ?
  - Answer : 2. We need to express the triples :
    <Jean knows x>
    <Marie knows x>

Solution :

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

ex:Jean foaf:knows _:b1 .
ex:Marie foaf:knows _:b1 .

Triples of the Data Model

Number	Subject	Predicate	Object
1	http://example.org/Marie	http://xmlns.com/foaf/0.1/knows	genid:UA0
2	http://example.org/Jean	http://xmlns.com/foaf/0.1/knows	genid:UA0

1. Manipulating the RDF Turtle language.

- **Exercise 5b :** expand the previous solution to show that all entities are of class Person.

Solution :

```
PREFIX foaf:  <http://xmlns.com/foaf/0.1/>
PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ex:    <http://example.org>

ex:Jean a foaf:Person ; foaf:knows _:b1 .
ex:Marie a foaf:Person ; foaf:knows _:b1 .
_:b1 a foaf:Person .
```

1. Manipulating the RDF Turtle language.

- **Exercise 6a : e**xpress the following statement in Turtle :

Albert Einstein married Mileva Marić in 1903 and Elsa Löwenthal in 1919.

Is the following representation appropriate ?

```
PREFIX : <http://example.org/>
PREFIX dbo:  <http://dbpedia.org/ontology/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
:Albert_Einstein
        dbo:spouse :Mileva_Marić ;
         dc:date 1903 ;
        dbo:spouse :Elsa_Löwenthal ;
         dc:date 1919 .
```

Answer: no. Why ?

The two dates and the two marriages are not associated to each other.

What would be a suitable solution ?

(after an exercice from Fiorelli, university di Roma Tor Vergata)

. Manipulating the RDF Turtle language.

- **Exercise 6b :** a suitable solution (Bio is a vocabulary for biographical information).

Albert Einstein married Mileva Marićin 1903 and Elsa Löwenthal in 1919.

```
PREFIX : <http://example.org/>
PREFIX bio:  <http://purl.org/vocab/bio/0.1/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dc: <http://purl.org/dc/elements/1.1/>

:marriage1 rdf:type bio:Marriage;
           bio:partner :Albert_Einstein;
           bio:partner :Mileva_Marić ;
           dc:date 1903  .

:marriage2 rdf:type bio:Marriage;
           bio:partner :Albert_Einstein;
           bio:partner :Elsa_Löwenthal;
           dc:date 1919 .
```

1. Manipulating the RDF Turtle language.

Suggested exercise 1 : correct the mistakes in the following Turtle document.

```
PREFIX vcard: http://www.w3.org/2006/vcard/ns#
PREFIX foaf http://xmlns.com/foaf/0.1/
http://example.org/people/jlb#me a foaf:Person vcard:Individual
name "Jean-Louis Binot" .  # Hint: foaf has this property
vcard:hasEmail "jean-louis.binot@uliege.be" ;
vcard:hasTelephone  [ a vcard:Voice ,  vcard:hasValue  11111111] ;
```

Check your answer with a validator.

1. Manipulating the RDF Turtle language.

Suggested exercise 2 : write an RDF model representing the following statements :

"Ideas and opinions" has been created by Einstein.
Book1 and Book2 have been created by the same unknown author.
Amazon states that "Ideas and opinions" has been published by Broadway Books.

Use the Dublin Core ontology and assume that an appropriate default namespace exists for the resource mentioned (e.g. *:Einstein)*.

1. Manipulating the RDF Turtle language.

- **Suggested exercise 3 :** express the following statements in RDF Turtle :

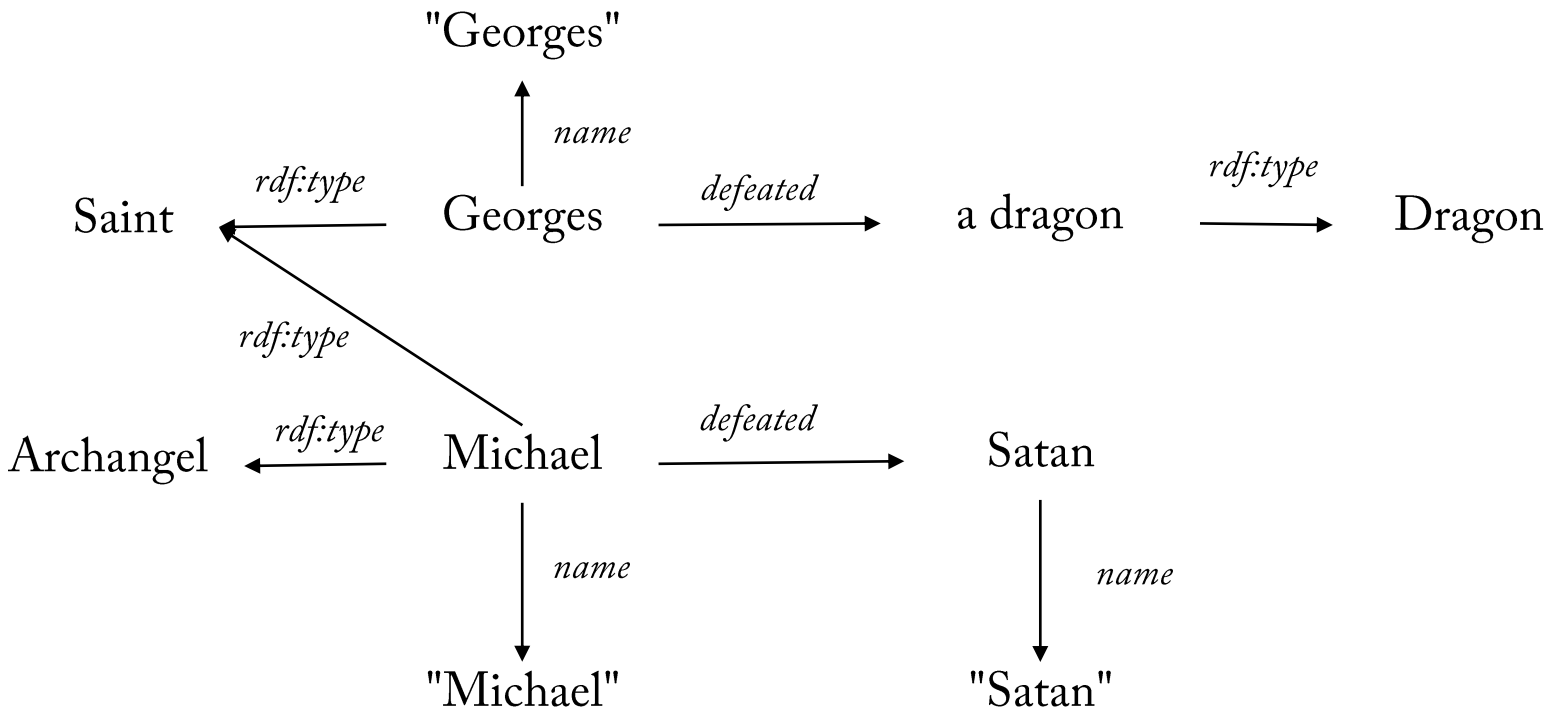
a) *Georges lives in Brussels and Marie in a city located in Wallonia.*

b) *Three students, Pierre, Jean and Marie are living in Liège, located in Belgium. They collectively follow the course of Logic, given by Willard Quine.*

Assume that an appropriate default namespace exist for the resources mentioned.

1. Manipulating the RDF Turtle language.

- **Suggested exercise 4 :** model the following graph in RDF Turtle, using appropriate vocabularies and prefixes :

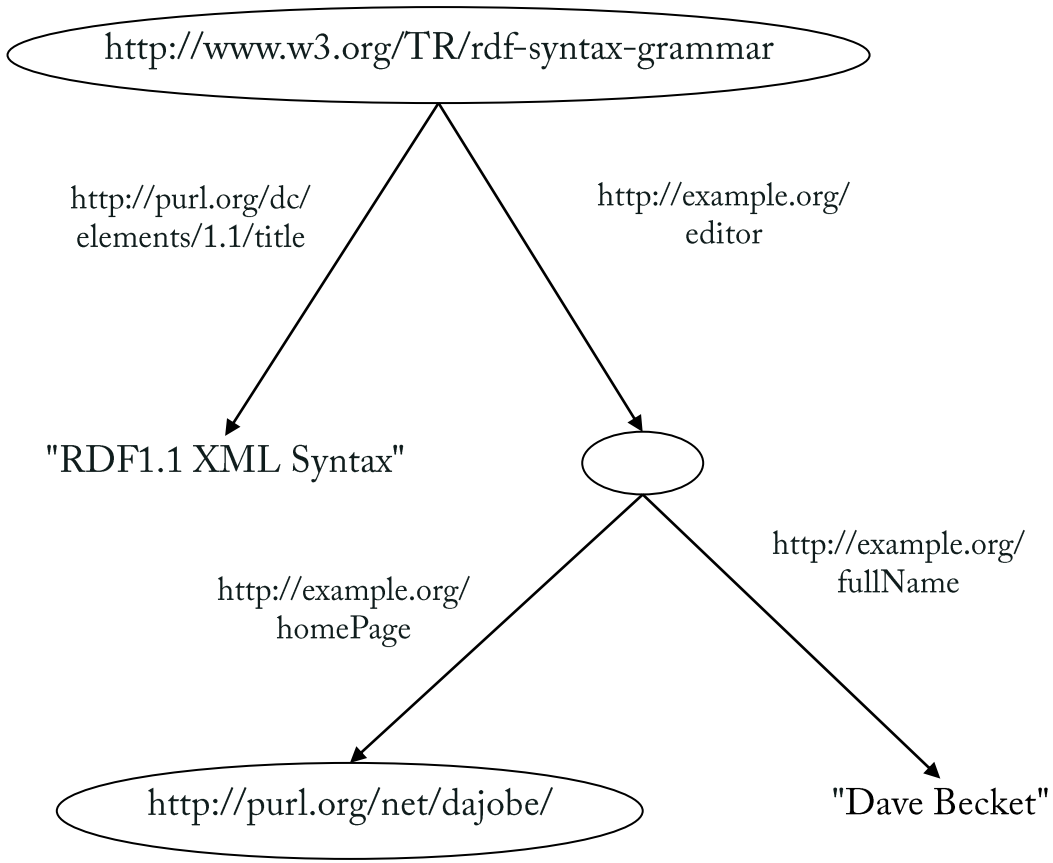


2. Transform triples in a labeled graph

- **Exercise 1 :** draw the graph corresponding to the following Turtle code:

```
PREFIX ex:    <http://example.org/>
PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dc:    <http://purl.org/dc/elements/1.1/>

<http://www.w3.org/TR/rdf-syntax-grammar>
      ex:editor  [ ex:fullName  "Dave Beckett" ;
                   ex:homePage  <http://purl.org/net/dajobe/>
                   ] ;
      dc:title   "RDF1.1 XML Syntax" .
```



(inspired from the document RDF1.1 XML syntax)

2. Transform triples in a labeled graph

- **Suggested exercise 1 :** draw the graph corresponding to the following Turtle code :

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

:JLBinot a :Teacher;
:teaches "SemanticData" , "Integrated Software Project" ;
foaf:member :ULiege .
[] foaf:knows :JLBinot; foaf:member [a :University] .
```

3. Express a model in RDFS (Turtle syntax).

- **Exercise 1**: write an RDF/RDFS model representing the following statements:
  - C1 and C2 are classes;
  - I1 is an instance of C1;
  - I2 and I3 are instances of C2;
  - P1 is a property of domain C1 and of range C2.

  Use appropriate abbreviations.

```
PREFIX rdf:
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs:
  <http://www.w3.org/2000/01/rdf-schema#rdfs>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX : <http://example.org/terms/>

:C1 a rdfs:Class .
:C2 a rdfs:Class .

:I1 a :C1 .
:I2 a :C2 .
:I3 a :C2 .

:P1 a rdf:Property ;
    rdfs:domain :C1 ;
    rdfs:range :C2 .
```

3. Express a model in RDFS (Turtle syntax).

Exercise 2 : write an RDFS document defining the following elements (Turtle syntax):

concepts :

- *lions, tigers, wildlife, animals, zoos, cities, places.*

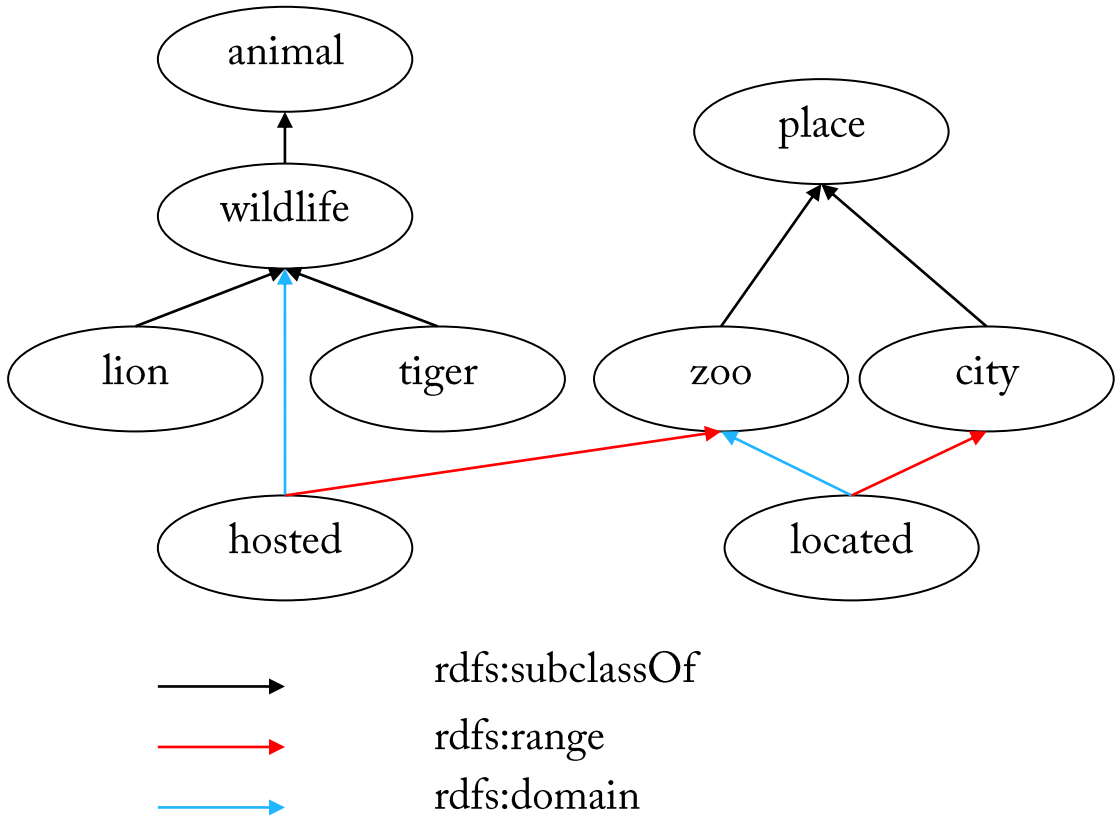
properties (specify in each case domain and range) :

- *hosted* (a wildlife animal is hosted in a zoo);
- *located* (a zoo is located in a city).

Add some instances.

Check your syntax with a validator.

- First draw a graph-based model



3. Express a model in RDFS

Exercise 2 continued : write an RDFS document defining the following elements (Turtle syntax):

concepts :

- *lions, tigers, wildlife, animals, zoos, cities, places.*

properties (specify in each case domain and range) :

- *hosted* (a wildlife animal is hosted in a zoo);
- *located* (a zoo is located in a city).

Add some instances.

Check your syntax with a validator.

```
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
PREFIX :  <http://example.org/>

:lion  rdfs:subClassOf  :wildlife .
:tiger  rdfs:subClassOf  :wildlife .
:wildlife rdfs:subClassOf  :animal .
:zoo  rdfs:subClassOf  :place .
:city  rdfs:subClassOf  :place .

:hosted  rdfs:range  :zoo .
:hosted  rdfs:domain  :wildlife .

:located  rdfs:domain  :zoo .
:located rdfs:range  :city .

:lion1 rdf:type :lion .
:tiger1 rdf:type :tiger .
:tiger1 :hosted :zoo1 .
:zoo1 :located  :city1 .
```

Express a model in RDFS

- **Suggested exercise 1 :** model the simple RDFS ontology about pizzas described below:
  - A pizza has two types of ingredients : a base and a topping.
  - We distinguish two kinds of pizza bases : DeepPan base, and ThinAndCrispy base.
  - We distinguish four kinds of pizza toppings : Cheese, Meat, Fish and Vegetable toppings.
  - We consider three kinds of pizza : Cheesy, Named and Meaty.
  - We distinguish four kinds of named pizza : American, AmericanHot, Margherita, and Siciliana.

- **Suggested exercise 2 :** write an RDFS model about a family (parents, mother, father, children, son, daughter). Organize classes into a hierarchy. Extend the model with other concepts (brother, sister, aunt, uncle). Try to distinguish between what should be a class and what should be a property. Add some instances.

4. Applying RDFS inference rules.

Exercise 1 : you are given the following schema (prefixes omitted) :

:Person a rdfs:Class .
:Student a rdfs:Class .
:Student rdfs:subClassOf :Person .
:University a rdfs:Class .
:enrolledAt a rdf:Property .
:memberOf a rdf:Property .
:memberOf rdfs:domain :Person .
:memberOf rdfs:range :University .
:enrolledAt rdfs:subPropertyOf :memberOf .
:enrolledAt rdfs:domain :Student .

and the statement:

:Michel :enrolledAt :ULiege.

Show how a reasoner would conclude :

:Michel a :Person .

- Two ways:

:Michel a :Student . (from domain of enrolledAt)
:Michel a :Person . (from :Student rdfs:subClassOf :Person .)

:Michel :memberOf :ULiege
(from :enrolledAt rdfs:subPropertyOf :memberOf .)
:Michel a :Person . (from domain of :memberOf)

4. Applying RDFS inference rules.

Suggested exercise 1 : which additional triplets can be inferred from the following RDFS code ?

```
PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
PREFIX ex:   <http://example.org/>
PREFIX foaf:  <http://xmlns.com/foaf/0.1/#>

ex:Person  a  rdfs:Class .
ex:Student  a rdfs:Class ;
      rdfs:subClassOf  ex:Person .
ex:Teacher  a rdfs:Class ;
      rdfs:subClassOf  ex:Person .

ex:John  ex:teaches ex:George .
ex:teaches  a            rdf:Property ;
      rdfs:domain     ex:Teacher;
      rdfs:range        ex:Student ;
      rdfs:subPropertyOf  foaf:knows .
```

Technical note: writing IRIS in Turtle

```
# A triple with all absolute IRIs
<http://one.example/subject1> <http://one.example/predicate1> <http://one.example/object1> .

@base <http://one.example/> .
<subject2> <predicate2> <object2> .          # relative IRIs, e.g. http://one.example/subject2

BASE <http://one.example/>
<subject2> <predicate2> <object2> .          # relative IRIs, e.g. http://one.example/subject2

@prefix p: <http://two.example/> .
p:subject3 p:predicate3 p:object3 .          # prefixed name, e.g. http://two.example/subject3

PREFIX p: <http://two.example/>
p:subject3 p:predicate3 p:object3 .          # prefixed name, e.g. http://two.example/subject3

@prefix p: <path/> .                         # prefix p: now stands for http://one.example/path/
p:subject4 p:predicate4 p:object4 .          # prefixed name, e.g. http://one.example/path/subject4

@prefix : <http://another.example/> .        # empty prefix
:subject5 :predicate5 :object5 .             # prefixed name, e.g. http://another.example/subject5

:subject6 a :subject7 .                      # same as :subject6 <http://www.w3.org/1999/02/22-rdf-syntax-
ns#type> :subject7 .
<http://伝言.example/?user=أكرم&amp;amp;channel=R%26D> a :subject8 . # a multi-script subject IRI .
```

Notes

- The Turtle language originally permitted only the syntax including the '@' character for writing prefix and base directives. The case-insensitive 'PREFIX' and 'BASE' forms were added to align Turtle's syntax with that of SPARQL.
- The '@prefix' and '@base' directives require a trailing '.' after the IRI, the equivalent 'PREFIX' and 'BASE' must not have a trailing '.' after the IRI part of the directive.

(source https://www.w3.org/TR/turtle/)

THANK YOU