Day 04: questions from the course on OWL.

Q5.1 What can we deduce?

```
ex:Man owl:intersectionOf (ex:Male ex:Human) .
ex:Woman owl:intersectionOf (ex:Female ex:Human) .
ex:Human owl:unionOf (ex:Man ex:Woman) .
ex:Jane a ex:Human .
ex:John a ex:Man .
ex:James a ex:Male .
ex:Jane a ex:Female .
```

Answer

Jane is also a Woman and Human
John is a Human Male

Q5.2 What are we defining and inferring?

```
@prefix ex: <http://example.org/>

ex:Father rdfs:subClassOf [
   a owl:Class ;
   owl:intersectionOf ( ex:Parent ex:Man )
] .

ex:Jim a ex:Man, ex:Parent .
ex:Jack a ex:Father .
```

Answer

So here we know only that Jack is a Father, Man and Parent. (we cannot say the Jim)
It's in subclass of a cursive so only one direction.

Q5.3 What can we deduce?

```
ex:hasSpouse a owl:SymmetricProperty .
ex:hasChild  owl:inverseOf ex:hasParent .
ex:hasParent rdfs:subPropertyOf ex:hasAncestor .
ex:hasAncestor a owl:TransitiveProperty .
ex:Jim  ex:hasChild  ex:Jane .
ex:Jane ex:hasSpouse ex:John .
ex:Jim  ex:hasParent ex:James .
```

Answer

Jane hasParent Jim
John hasSpouse Jane
James hasChild Jim
Jim has Ancestor James
Jane has Ancestor Jim and James

Q5.4 What can we deduce?

```
ex:Human owl:equivalentClass foaf:Person .
foaf:name owl:equivalentProperty ex:name .
ex:JimmyPage a ex:Human ;
             owl:sameAs ex:JamesPatrickPage .
ex:JimmyHendrix owl:differentFrom ex:JimmyPage .
```

Answer

Foaf:Person is equivalentClass as ex:Human
Ex:name is also equivalentProperty as foaf:name
Ex:JimmyPage is a human also person (foaf:person) and it's the same as ex:JamesPatrickPage and different from ex:JimmyHendrix

Q5.5 What are we defining and inferring?

```
ex:UnluckyPerson owl:equivalentClass [
```

```
  a owl:Class ;
  owl:intersectionOf (
    ex:Person
    [ a owl:Class ; owl:complementOf ex:Lucky ]
  )
] .
```

Answer

If you are a person and the complement of lucky, you are unluckyPerson

Q5.6 What can we deduce?

```
ex:Human rdfs:subClassOf
  [ a owl:Restriction ;
    owl:onProperty ex:hasParent ;
    owl:allValuesFrom ex:Human ] .
ex:Tom a ex:Human .
ex:Tom ex:hasParent ex:James, ex:Jane.
```

Answer

James and Jane become human

Q5.7 What are we defining and inferring?

```
@prefix ex: <http://example.org/>
ex:PersonList rdfs:subClassOf
  [
    a owl:Restriction ;
    owl:onProperty rdf:first ;
    owl:allValuesFrom ex:Person
  ] ,  [
    a owl:Restriction ;
    owl:onProperty rdf:rest ;
    owl:allValuesFrom ex:PersonList
  ] .

ex:value rdfs:range ex:PersonList .
ex:abc ex:value (ex:a ex:b ex:c) .
```

Answer

OnProperty first must be a person
onProperty of rest must be personlist
(a, b, c) become a personlist
A become a person, (b c) become a person list
B become a person, (c) become a person list. And C become a person

Q5.8 What are we defining and inferring?

```
@prefix ex: <http://example.org/>
ex:Human rdfs:subClassOf [
  owl:intersectionOf (
    [
      a owl:Restriction ;
        owl:onProperty ex:hasBiologicalFather ;
        owl:maxCardinality 1
    ] , [
      a owl:Restriction ;
        owl:onProperty ex:hasBiologicalMother ;
        owl:maxCardinality 1
    ] )
] .
```

```
ex:Jane a ex:Human ;
        ex:hasBiologicalFather ex:James , ex:Jhon .
```

Answer

A human only one biological father, and one biological mother

The system would deduce that James and Jhon are the same person

Q5.9 What are we defining and inferring?

```
@prefix ex: <http://example.org/>
ex:Wealthy a owl:Class ;
     owl:equivalentClass  [
       a owl:Class ; owl:intersectionOf  (
         [ a owl:Restriction ;
           owl:onProperty ex:hasChild ;
           owl:allValuesFrom ex:Wealthy
         ] ,
         [ a owl:Restriction ;
           owl:onProperty ex:hasChild ;
           owl:someValuesFrom ex:Wealthy
         ]
     )] .
ex:Tom a ex:Wealthy ; ex:hasChild ex:Tim .
```

Answer

1. All your child has to be wealthy
2. You need to at least have one child
3. For people who have no child would all be wealthy
4. So Tim is wealthy too

Day 04: Answers to the practical session on OWL.

Software requirements

- The RDF XML online validation service by W3C: https://www.w3.org/RDF/Validator/
- The RDF online translator: http://rdf-translator.appspot.com/
- The SPARQL Corese engine: http://wimmics.inria.fr/corese

A, Query data augmented by an OWL schema

Make a copy of the human.rdfs file, name it humans.owl and use it for the rest of the session. For each of the following statements, specify a SPARQL query that shows that the difference before and after running the OWL inferences: you will find that answers to these queries are different depending on whether you load the ontology humans.rdfs or the humans.owl you modified.

1. Declare that `hasSpouse` is a symmetrical property and do the same for and `hasFriend` .

Code added to the schema:

```
<#hasSpouse> a rdf:Property, owl:SymmetricProperty ;
<#hasFriend> a rdf:Property, owl:SymmetricProperty ;
```

Query:

```
prefix : <http://www.inria.fr/2007/09/11/humans.rdfs#> .
select * where {?x ?p ?y. filter(?p= :hasFriend || ?p= :hasSpouse)}
```

Result before addition to the schema:

Graph | XML/RDF | Table | Validate

num	?x	?p	?y
1	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Eve>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Alice>
2	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#David>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>
3	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Alice>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>
4	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Jack>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Alice>
5	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Alice>
6	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Karl>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Sophie>
7	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Sophie>
8	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Eve>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#David>
9	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Flora>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>
10	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Jennifer>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>
11	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Karl>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Catherine>
12	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#William>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>

Result after addition to the schema:

Graph | XML/RDF | Table | Validate

num	?x	?p	?y
1	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Alice>
2	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Sophie>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Karl>
3	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Eve>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Alice>
4	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#David>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>
5	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Alice>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>
6	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Alice>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Eve>
7	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Alice>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Jack>
8	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Alice>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>
9	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#David>
10	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Jack>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Alice>
11	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Alice>
12	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Karl>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasFriend>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Sophie>
13	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Sophie>
14	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Jennif...
15	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Sophie>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>
16	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Eve>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#David>
17	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#David>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Eve>
18	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Flora>
19	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Flora>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>
20	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#William>
21	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Jennifer>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>
22	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Catherine>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Karl>
23	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Karl>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Cathe...
24	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#William>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>

Explanation:

At first I only have 12 result but after I have 24. Take the first data Eve for example. She has friend Alice. But in the first result Alice does not have friend Eve. While adding Symmetric property to has friend and has Spouse it would automatically make the relationship both side.

2. Declare that `hasChild` is the inverse property of the `hasParent` property.

Code added to the schema:

```
<#hasChild>
  a rdf:Property ;
  owl:inverseOf <#hasParent> ;
```

Query:

```
prefix : <http://www.inria.fr/2007/09/11/humans.rdfs#> .
select * where {  ?x ?p ?y. filter(?p= :hasChild || ?p= :hasParent )}
```

Result before addition to the schema:

Graph | XML/RDF | Table | Validate

num	?x	?p	?y
1	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>
2	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Jack>
3	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Pierre>
4	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Jack>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>
5	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Flora>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Pierre>
6	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasParent>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Sophie>
7	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Mark>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasParent>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>
8	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Lucas>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasParent>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Catherine>
9	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Lucas>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasParent>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Karl>
10	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Cather...	<http://www.inria.fr/2007/09/11/humans.rdfs#hasParent>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>

Result after addition to the schema:

Graph | XML/RDF | Table | Validate

num	?x	?y	?p
1	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>
2	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Mark>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>
3	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Sophie>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>
4	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Jack>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>
5	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Pierre>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>
6	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Jack>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>
7	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Flora>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Pierre>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>
8	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Catherine>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>
9	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Catherine>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Lucas>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>
10	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Karl>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Lucas>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>
11	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Jack>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasParent>
12	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasParent>
13	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Sophie>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasParent>
14	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Mark>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasParent>
15	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Jack>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasParent>
16	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Pierre>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasParent>
17	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Pierre>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Flora>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasParent>
18	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Lucas>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Catherine>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasParent>
19	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Lucas>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Karl>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasParent>
20	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Catherine>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#hasParent>

Explanation:

By adding has parent inverse of has child means that if we only have has child the opposite relationship has parent would be set. Likewise if we only have has Parent relation then has child inverse would be setup. See first data Harry for example. He has child John while John has parent Sophie. Then the new query John would have parent Harry and Sophie would has child John.

3. Declare `hasAncestor` as transitive property.

Code added to the schema:

```
<#hasAncestor>
  a rdf:Property, owl:TransitiveProperty ;
```

Query:

```
prefix : <http://www.inria.fr/2007/09/11/humans.rdfs#> .
select * where {?x :hasAncestor ?y}
```

Result before addition to the schema:

?x	?y
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Sophie>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Mark>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Lucas>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Catherine>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Lucas>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Karl>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Catherine>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>

Result after addition to the schema:

?x	?y
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Jack>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Sophie>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Jack>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Mark>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Mark>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#John>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Mark>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Sophie>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Mark>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Mark>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Jack>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Jack>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Pierre>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Pierre>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Flora>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Lucas>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Lucas>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Catherine>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Lucas>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Karl>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Catherine>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>

Explanation:

By doing transitive property we can see that the parents of our parent would be our ancestor too even if we didn't specify it. Take 2 first data for example Mark has ancestor John and John has ancestor Sophie. Now Mark has ancestor Sophie too (as well as Harry because from the last exercise we get that harry is John's father)

4. Declare the disjunction between `Male` and `Female`. Violate the constraint in the data, check the results and then remove the violation you created.

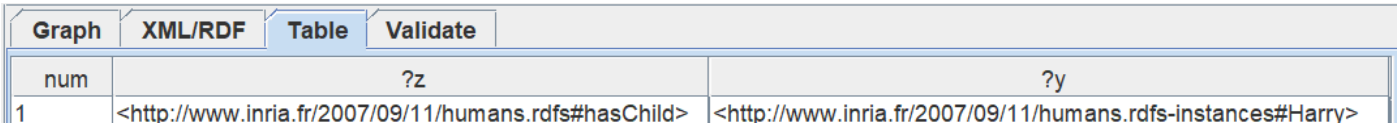
- Code added to the schema:

```
<#Male>
  a owl:Class, rdfs:Class ;
  owl:disjointWith <#Female>;
```

- Query:

```
prefix : <http://www.inria.fr/2007/09/11/humans.rdfs#> .
SELECT ?z ?y
WHERE {?x ?z ?y. ?y :name "Harry"}
```

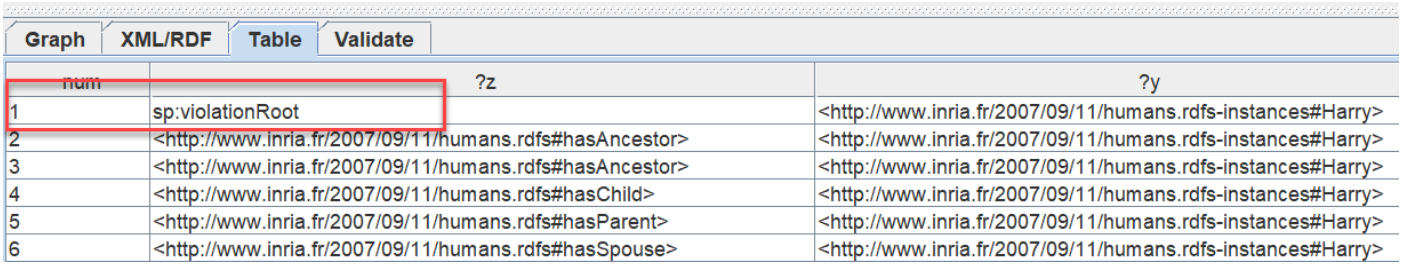
- Result before addition to the schema:



Graph | XML/RDF | Table | Validate

num	?z	?y
1	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>

- Result after addition to the schema:



Graph | XML/RDF | Table | Validate

num	?z	?y
1	sp:violationRoot	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>
2	<http://www.inria.fr/2007/09/11/humans.rdfs#hasAncestor>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>
3	<http://www.inria.fr/2007/09/11/humans.rdfs#hasAncestor>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>
4	<http://www.inria.fr/2007/09/11/humans.rdfs#hasChild>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>
5	<http://www.inria.fr/2007/09/11/humans.rdfs#hasParent>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>
6	<http://www.inria.fr/2007/09/11/humans.rdfs#hasSpouse>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Harry>

- Explanation:

```
<Man rdf:ID="Harry">
    <rdf:type rdf:resource="&humans;#Female"/>
    <name>Harry</name>
    <hasChild rdf:resource="#John"/>
    <hasSpouse rdf:resource="#Sophie"/>
</Man>
```

I have added Female to Harry, who was a Man (that would auto inherit Person, Male and Animal class). Which violate the rule we set up about the disjointWith in the schema. We can query and see the sp:violationRoot result.

5. Declare that the class `Professor` is the intersection of the class `Lecturer` and `Researcher` class.

Code added to the schema:

```
<#Professor>
  a rdfs:Class, owl:Class;
  owl:intersectionOf(<#Lecturer> <#Researcher>).
```

Query:

```
prefix : <http://www.inria.fr/2007/09/11/humans.rdfs#> .
select * where {?x a :Lecturer; a ?y}
```

```
prefix : <http://www.inria.fr/2007/09/11/humans.rdfs#> .
select * where {?x a :Professor}
```

Result before addition to the schema:

num	?x	?y
1	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Eve>	<http://www.inria.fr/2007/09/11/humans.rdfs#Person>
2	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Eve>	<http://www.inria.fr/2007/09/11/humans.rdfs#Lecturer>
3	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#Female>
4	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#Person>
5	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#Lecturer>
6	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#Researcher>
7	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#Animal>

num	?x

Result after addition to the schema:

n...	?x	?y
1	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Eve>	<http://www.inria.fr/2007/09/11/humans.rdfs#Animal>
2	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Eve>	<http://www.inria.fr/2007/09/11/humans.rdfs#Person>
3	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Eve>	<http://www.inria.fr/2007/09/11/humans.rdfs#Lecturer>
4	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Eve>	<http://www.inria.fr/2007/09/11/humans.rdfs#Academic>
5	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Eve>	owl:Thing
6	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#Animal>
7	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#Female>
8	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#Person>
9	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#Lecturer>
10	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#Researcher>
11	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#Professor>
12	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#Academic>
13	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	owl:Thing

Graph | XML/RDF | Table | Validate

num	?x
1	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>

Explanation:

We define that only if we have both Lecturer and Researcher then it is a Professor using owl:intersectionOf. Before the owl. Professor was automatically added to the relationship of Laura after adding owl:intersectionOf.

6. Declare that the `Academic` class is the union of classes `Lecturer` and `Researcher`.

Code added to the schema:

```
<#Academic>
  a rdfs:Class, owl:Class;
  owl:unionOf ( <#Lecturer> <#Researcher> ).
```

Query:

```
prefix : <http://www.inria.fr/2007/09/11/humans.rdfs#> .
select * where {?x a ?y. filter (?y = :Lecturer || ?y = :Researcher)}
```

```
prefix : <http://www.inria.fr/2007/09/11/humans.rdfs#> .
select * where {?x a :Academic}
```

Result before addition to the schema:

?x	?y
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Eve>	<http://www.inria.fr/2007/09/11/humans.rdfs#Lecturer>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#David>	<http://www.inria.fr/2007/09/11/humans.rdfs#Researcher>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>	<http://www.inria.fr/2007/09/11/humans.rdfs#Researcher>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#Lecturer>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#Researcher>

Graph | XML/RDF | Table | Validate

num	?x

Result after addition to the schema:

?x	?y
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Eve>	<http://www.inria.fr/2007/09/11/humans.rdfs#Lecturer>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#David>	<http://www.inria.fr/2007/09/11/humans.rdfs#Researcher>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>	<http://www.inria.fr/2007/09/11/humans.rdfs#Researcher>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#Lecturer>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs#Researcher>

Graph | XML/RDF | Table | Validate

num	?x
1	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Eve>
2	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#David>
3	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Gaston>
4	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>

Explanation:

We define that both Lecturer and Researcher are Academic using owl:unionOf. Before the owl. We can see the Lecturer and researcher, but not Academic. Academic was automatically added to the relationship after adding owl:union.

7. Create a class `Organization` and its sub class `University`. Create a new property `mainEmployer`, with domain `Person` and range `Organization`. Use a restriction to declare that any `Professor` has for main employer a `University`.

Code added to the schema (new property, new classes and new restriction):

- New Property

```
<#mainEmployer> a rdf:Property;
    rdfs:domain <#Person>;
    rdfs:range <#Organization>.
```

- New Classes

```
<#Organization> a rdfs:Class .
<#University> a rdfs:Class ;
    rdfs:subClassOf <#Organization>.
```

- New restriction (or equivalent class)

```
<#Professor>
  a rdfs:Class, owl:Class;
  owl:intersectionOf(<#Lecturer> <#Researcher>);
  rdfs:subClassOf  [a owl:Restriction;
    owl:onProperty <#mainEmployer>;
    owl:allValuesFrom <#University>].
```

Code added to the data (just declare the main employer of a Professor):

```
<Person rdf:ID="Laura">
    <hasFriend rdf:resource="#Alice"/>
    <mainEmployer rdf:resource="#DSTI">
    <name>Laura</name>
</Person>
```

Query:

```
prefix : <http://www.inria.fr/2007/09/11/humans.rdfs#> .
select * where {?x :mainEmployer ?y. optional {?y a ?z}}
```

Result before addition to the schema:

?x	?y	?z
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#DSTI>	

Result after addition to the schema:

?x	?y	?z
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#DSTI>	<http://www.inria.fr/2007/09/11/humans.rdfs#Organization>
<http://www.inria.fr/2007/09/11/humans.rdfs-instances#Laura>	<http://www.inria.fr/2007/09/11/humans.rdfs-instances#DSTI>	<http://www.inria.fr/2007/09/11/humans.rdfs#University>

Explanation:

From above we can see that Laura is a researcher and lecturer which would automatically make her a professor. By adding the restriction we can tell that all the mainEmployer of the Laura should be a class university and Organization.

8. Use a restriction to declare that any person must have a parent who is a woman. For this last statement, you need to run the rule engine after loading the ontology and data.

Code added to the schema:

```
<#Person>
  a rdfs:Class, owl:Class ;
  owl:equivalentClass  [a owl:Restriction;
    owl:onProperty <#hasParent>;
    owl:someValuesFrom <#Woman>];
```

Query:

```
prefix : <http://www.inria.fr/2007/09/11/humans.rdfs#> .
select ?y where {?x a ?y; :name "Sandy". }
```

Result before addition to the schema:

?y
<http://www.inria.fr/2007/09/11/humans.rdfs#Female>
<http://www.inria.fr/2007/09/11/humans.rdfs#Animal>

Result after addition to the schema:

Graph | XML/RDF | Table | Validate

num	?y
1	<http://www.inria.fr/2007/09/11/humans.rdfs#Female>
2	<http://www.inria.fr/2007/09/11/humans.rdfs#Person>
3	<http://www.inria.fr/2007/09/11/humans.rdfs#Animal>

Explanation:

I've added new data

```
<Female rdf:ID="Sandy">
  <name>Sandy</name>
  <hasMother rdf:resource="#Catherine"/>
</Female>
```

This give me a Female (not a person) to the data who has a Parent that is woman. After adding the schema. Sandy automatically become a person.

B, Make your own OWL models:

For each one of the following OWL primitives imagine a definition that could use it and provide that definition in OWL using your preferred syntax (RDF/XML or N3/Turtle). For instance a possible definition using owl:TransitiveProperty would be a definition of the Ancestor property. For each primitive in the following list you imagine the definition of a class or property that was not given in the course and you give that definition in English and in OWL.

1. owl:oneOf
2. owl:unionOf
3. owl:intersectionOf
4. owl:complementOf
5. owl:disjointWith

   or owl:AllDisjointClasses
   or owl:disjointUnionOf
6. owl:ObjectProperty
7. owl:DatatypeProperty
8. owl:SymmetricProperty

   or owl:AsymmetricProperty
9. owl:inverseOf
10. owl:TransitiveProperty
11. owl:propertyDisjointWith
12. owl:ReflexiveProperty

    or owl:IrreflexiveProperty
13. owl:propertyChainAxiom
14. owl:FunctionalProperty
15. owl:InverseFunctionalProperty
16. owl:hasKey
17. owl:allValuesFrom
18. owl:someValuesFrom
19. owl:hasValue
20. owl:maxCardinality

    or owl:minCardinality
21. owl:qualifiedCardinality

	OWL	ENGLISH
1	`<#Gender> a owl:Class ;` `  owl:oneOf (<#Male> <#Female>).`	Male and Female are both in Gender Class
2	`<#Class> a owl:Class;` `  owl:unionOf(<#Student> <#Teacher>).`	A class class is a union of student and teacher class

3	`<#AnnualReport> a owl:Class;`
	`  owl:intersectionOf(<#Report>`
	`<#AnnualDoc>).`
		When a thing is both report and annual document then it is an annual report

4	`<#Failed> a owl:Class;` `  owl:complementOf <#Passed>.`	If a person doesn't passed the exam then he failed the exam
5	`<#Vegetable> a owl:Class;` `owl:disjointUnionOf (<#Fruit> <#Meat>).`	A food can only be vegetable or Fruit or Meat. They can not be both nor all for the same food.

6	`<#hasFriend> a owl:SymmetricProperty`
	`,owl:ObjectProperty;`
	`  rdfs:domain f:Person .`
		We define a hasFriend relationship property that is symmetric between 2 person (resource).

7	`<#bornYear> a owl:DatatypeProperty;`
	`  rdfs:domain f:Person ;`
	`  rdfs:range xsd:integer.`
		We define a relationship between a resource and a literal value (a person to an integer)

8	`<#hasFriend> a owl:SymmetricProperty`
	`,owl:ObjectProperty;`
	`  rdfs:domain f:Person .`
		We define a hasFriend relationship property that is symmetric between 2 person that is if a person is a friend of

		another person, then the inverse hasFriend relationship would be set up.
9	<#hasSchool> owl:inverseOf <#hasStudent>.	If a person hasSchool A then A would have student for that person
10	<#sameReligion> a owl:TransitiveProperty, owl:SymmetricProperty.	With this transitiveProperty we can link all the person who has the same religion together. A sameReligion as B and B sameReligion as C then A would be same religion as A
11	<#hasFailed> a owl:propertyDisjointWith <#hasPassed>.	A student can only failed or passed for every single subject.
12	<#knows> a owl:ReflexiveProperty. <#hasNeighbor> a owl:IrrefleiveProperty.	A person can know others but a person also know themselve. A person cannot have themselve as a neighbor
13	<#Grandma> owl:propertyChainAxiom (<#Parent><#Mather>).	The mother of your parent is your grandma.
14	<#hasMother> a owl:FunctionalProperty.	A person can have one and only mother if A has mother B and C, B and C should be the same.
15	<#hasHusband> a owl:FunctionalProperty, owlInverseFunctionalProperty.	A can only have on and only B as husband. If today A C both have husband B then AC are the same.
16	<#Student> owl:hasKey (<#cohort> <#studentSequence>)	In dsti from the cohort number (S19 for example) and the studentsequence number can make a student identification
17	<#HappyPerson> a owl:Class ; owl:equivalentClass [rdf:type owl:Class ; owl:intersectionOf ([a owl:Restriction ; owl:onProperty <#hasSpouse> ; owl:allValuesFrom <#Happy>], [a owl:Restriction ; owl:onProperty <#hasSpouse>; owl:someValuesFrom <#Happy>;])] .	A person is a happy person then the husband or wife of that person is happy too.
18	<#Author> owl:equivalentClass [a owl:Restriction; owl:onProperty <#creatorOf>; owl:someValuesFrom <#Book>];	A creator of a book would be also link as a class author.
19	<#Dog> rdfs:subClassOf [a owl:Restriction; Owl:onProperty <#nbLegs> 4].	A Dog would have 4 legs, but it doesn't means that 4 legs animal are all dogs
20	<#Human> rdfs:subClassOf [a owl:Restriction; owl:onProperty <#hasArm>; owl:maxCardinality 2].	A Human can only have maximum 2 arms
21	<#Car> rdfs:subClassOf [a owl:Restriction; owl:onProperty <#controlledBy>; owl:onClass <#Human> owl:qualifiedCardinality 1].	A Car have to be controlled by one person.