Exercises on OWL

Riccardo Rosati

Corso di Laurea Magistrale in Ingegneria Informatica
Sapienza Università di Roma
2018/2019
Exercise 1

Write an OWL ontology representing the following statements:

• URI1 and URI2 are classes
• URI3 is a property
• URI4 is an instance of class URI1, and URI5 and URI6 are instances of class URI2
• URI3 has domain URI1 and range URI2
• (URI6, URI4) is an instance of property URI3
Exercise 1: Solution

Declaration(Class(myns:URI1))
Declaration(Class(myns:URI2))

Declaration(ObjectProperty(myns:URI3))

ClassAssertion(myns:URI1 myns:URI4)
ClassAssertion(myns:URI2 myns:URI5)
ClassAssertion(myns:URI2 myns:URI6)

SubClassOf(
    ObjectSomeValuesFrom(myns:URI3 owl:Thing)
    myns:URI1)
Exercise 1: Solution (continued)

SubClassOf(
    ObjectSomeValuesFrom(
        ObjectInverseOf(myns:URI3)
        owl:Thing)
    myns:URI2)

ObjectPropertyAssertion(myns:URI3 myns:URI6
    myns:URI4)
Exercise 2

Write an OWL ontology that formalizes knowledge about the domain of people, in particular the classes person, man, woman, and the properties parent, mother, father.

Try to express all the knowledge you have about such classes and properties (e.g.: every man is a person, every woman is a person, every mother is a woman, etc.).
Exercise 2: Solution

SubClassOf(myns:man myns:person)  (every man is a person)
SubClassOf(myns:woman myns:person)  (every woman is a person)

SubObjectPropertyOf(myns:hasMother myns:hasParent)
(hasMother is a subproperty of hasParent)
SubObjectPropertyOf(myns:hasFather myns:hasParent)
(hasFather is a subproperty of hasParent)

SubClassOf(
  ObjectSomeValuesFrom(
    ObjectInverseOf(myns:hasMother)
    owl:Thing)
  myns:woman)  (every mother is a woman)
SubClassOf(
    ObjectSomeValuesFrom(
        ObjectInverseOf(myns:hasFather)
        owl:Thing)
    myns:man)  (every father is a man)

ClassAssertion(myns:man myns:Joe)  (Joe is a man)

ObjectPropertyAssertion(myns:hasMother myns:Joe
    myns:Ann)  (Ann is the mother of Joe)
Exercise 3

Add to the ontology of Exercise 2 the following information:

- Man and woman are disjoint classes
- Every person has a mother
- Every person has a father
- Every person has exactly two parents
- Every person has a father, who is a man
- Every person has a mother, who is a woman
- Every person has a father and a mother
Exercise 3: Solution

1) `DisjointClasses(myns:man myns:woman)` (man and woman are disjoint classes)

2) `SubClassOf(
   myns:person
   ObjectSomeValuesFrom(myns:hasMother owl:Thing))` (every person has a mother)

3) `SubClassOf(
   myns:person
   ObjectSomeValuesFrom(myns:hasFather owl:Thing))` (every person has a father)
4) SubClassOf(
    myns:person
    ObjectExactCardinality(2 myns:hasParent))
(every person has exactly two parents)

5) SubClassOf(
    myns:person
    ObjectSomeValuesFrom(myns:hasFather myns:man))
(every person has a father who is a man)

6) SubClassOf(
    myns:person
    ObjectSomeValuesFrom(myns:hasMother myns:woman))
(every person has a mother who is a woman)
Exercise 3: Solution (continued)

7) SubClassOf(
   myns:person
   ObjectIntersectionOf(
      ObjectSomeValuesFrom(myns:hasMother owl:Thing)
      ObjectSomeValuesFrom(myns:hasFather owl:Thing)))
(every person has a mother and a father)

Notice that axiom 7) is equivalent to the above pair of axioms 2) and 3)