14/06/2017 EX5

lunedì 4 gennaio 2021 14:43

Exercise 5

(a) Write an OWL ontology that formalizes the domain described at point (a) of Exercise 4.

(b) Add to the above ontology the axioms formalizing the following statements:

1. add a new property `isWrittenBy` and state that it is the inverse of `isWriterOf`;
2. add a new class `WrittenByMultipleAuthors` and state that it corresponds to the class of movies written by at least two writers;
3. add a new class `LowBudgetMovie` and state that it corresponds to the class of movies played by at most 5 actors;
4. add the new class `ItalianMovie` and state that such a class corresponds to the class consisting of every movie such that all its writer(s) and director(s) were born in Italy.

Then, tell whether the resulting OWL ontology is redundant, i.e.: can some of the axioms constituting the ontology be deleted without changing the meaning (that is, the models) of the ontology? if so, identify and list such axioms.

a)

1) Declaration(Class(myns:Person))
   Declaration(Class(myns:Director))
   Declaration(Class(myns:Writer))
   Declaration(Class(myns:Actor))
   Declaration(Class(myns:Country))
   Declaration(Class(myns:Movie))
   Declaration(Class(myns:Comedy))
   Declaration(Class(myns:Drama))
   Declaration(Class(myns:Man))
   Declaration(Class(myns:Woman))
2) subClassOf(myns: Man myns:Person)
   subClassOf(myns: Woman myns:Person)
3) subClassOf(myns: Comedy myns:Movie)
   subClassOf(myns: Drama myns:Movie)
4) Declaration(ObjectProperty(myns:actsIn))
   Declaration(ObjectProperty(myns:bornIn))
   Declaration(ObjectProperty(myns:filmedIn))
   Declaration(ObjectProperty(myns:isDirectorOf))
   Declaration(ObjectProperty(myns:isWriterOf))
5) subClassOf(ObjectSomeValuesFrom(myns:isDirectorOf owl:Thing) myns:Director)
   subClassOf(ObjectSomeValuesFrom(ObjectInverse(myns:isDirectorOf) owl:Thing) myns:Movie)
6) subClassOf(ObjectSomeValuesFrom(myns:filmedIn owl:Thing) myns:Movie)
   subClassOf(ObjectSomeValuesFrom(ObjectInverse(myns:filmedIn) owl:Thing) myns:Country)
7) subClassOf(ObjectSomeValuesFrom(myns:bornIn owl:Thing) myns:Person)
   subClassOf(ObjectSomeValuesFrom(ObjectInverse(myns:BornIn) owl:Thing) myns:Country)
8) subClassOf(ObjectSomeValuesFrom(myns:actsIn owl:Thing) myns:Actor)
   subClassOf(ObjectSomeValuesFrom(ObjectInverse(myns:ctsIn) owl:Thing) myns:Movie)
9) subClassOf(DataSomeValuesFrom(myns:hasBoxOfficeGross owl:Thing) myns:Movie)
   subClassOf(owl:Thing DataAllValuesFrom(myns:hasBoxOfficeGross xsd:integer))
10) ObjectPropertyAssertion(myns:isDirectorOf myns:Ann myns:XYZ)
    ObjectPropertyAssertion(myns:isWriterOf myns:Ann myns:XYZ)
11) ObjectPropertyAssertion(myns:actsIn myns:Joe myns:ABC)
    ObjectPropertyAssertion(myns:actsIn myns:Paul myns:ABC)

12) ObjectPropertyAssertion(myns:filmedIn myns:ABC myns:France)
13) ClassAssertion(myns:Woman myns:Ann)
14) ClassAssertion(myns:Man myns:Paul)

b)

1) Declaration(ObjectProperty(myns:isWritterBy))
   EquivalentObjectProperty(myns:isWrittenBy ObjectInverse(isWriterOf))
2) Declaration(Class(myns:WrittenByMultipleAuthors))
   EquivalentClasses(myns:WriterByMyltipleAuthors ObjectIntersectionOf(myns:Movie
   ObjectMinCardinality(2 myns:isWrittenBy myns:Writer)))
3) Declaration(Class(myns:LowBudgetMovie))
   EquivalentClasses(myns:LowBudgetMovie ObjectIntersectionOf(myns:Movie
   ObjectMaxCardinality(2 ObjectInverseOf(myns:actsIn) myns:Actor)))
4) Declaration(Class(myns:ItalianMovie))
   EquivalentClasses(myns:ItalianMovie ObjectIntersectionOf(myns:Movie
   ObjectAllValuesFrom(myns:isWrittenBy
   ObjectHasValue(myns:bornIn myns:Italy))
   ObjectAllValuesFrom(ObjectInverseOf(myns:isDirectorOf)
   ObjectHasValue(myns:bornIn myns:Italy)))