23/01/2019 Ex5

mercoledì 6 gennaio 2021 12:10

(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. every manager leads at most one division;
2. `City` and `Division` are disjoint classes;
3. a `SpecialDivision` is a division for which at least four managers work;
4. a `SpecialManager` is a manager that manages at least two male employees and at least two female employees;
5. a `RomanEmployee` is an employee who works in a division located in Rome and lives in Rome.

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 of the ontology? if so, identify and list such axioms.

a)

1) Declaration (Class (myns: Employee))
Declaration (Class (myns: Manager))
Declaration (Class (myns: TopManager))
Declaration (Class (myns: Division))
Declaration (Class (myns: Man))
Declaration (Class (myns: Woman))
Declaration (Class (myns: City))

2) subClassOf (myns: TopManager myns: Manager)
subClassOf (myns: Manager myns: Employee)

3) Declaration (ObjectProperty (myns: worksWith))
Declaration (ObjectProperty (myns: livesIn))
Declaration (ObjectProperty (myns: isManagerOf))
Declaration (ObjectProperty (myns: headsDivision))
Declaration (ObjectProperty (myns: locatedIn))

4) subObjectPropertyOf (myns: isManagerOf myns: worksWith)

5) subClassOf (ObjectSomeValuesFrom (ObjectInverseOf (myns: isManagerOf) owl:Thing) myns: Manager)
subClassOf (ObjectSomeValuesFrom (myns: isManagerOf owl:Thing) myns: Employee)

6) subClassOf (ObjectSomeValuesFrom ObjectInverseOf (myns: worksWith) owl:Thing myns: Employee)
subClassOf (ObjectSomeValuesFrom (myns: worksWith owl:Thing) myns: Employee)

7) subClassOf (ObjectSomeValuesFrom ObjectInverseOf (myns: livesIn) owl:Thing myns: Person)
subClassOf (ObjectSomeValuesFrom (myns: livesIn owl:Thing) myns: City)

8) subClassOf (ObjectSomeValuesFrom ObjectInverseOf (myns: locatedIn) owl:Thing myns: Division)
subClassOf (ObjectSomeValuesFrom (myns: locatedIn owl:Thing) myns: City)

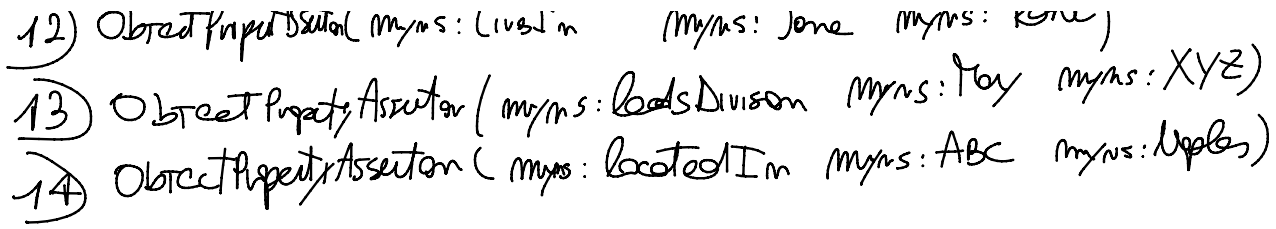
9) ClassAssertion (myns: Manager myns: Jane)

10) ClassAssertion (myns: Employee myns: Bob)
ClassAssertion (myns: Employee myns: ANN)

11) ObjectPropertyAssertion (myns: isManagerOf myns: Jane myns: Bob)

12) ObjectPropertyAssertion (myns: livesIn myns: Jane myns: Rome)

13) ObjectPropertyAssertion (myns: headsDivision myns: Roy myns: XYZ)



b)

1) subClassOf(myns:Manger ObjectMaxCardinality(1 myns:leadsDivision myns:Division))
2) DisjointClasses(myns:City myns:Division)
3) EquivalentClasses(myns:SpecialDivision ObjectIntersectOf(myns:Division ObjectMinCardinality(4 ObjectInverseOf(myns:worksInDivision) myns:Manger)))
4) EquivalentClasses(myns:SpecialManager ObjectIntersectOf(myns:Manager ObjectMinCardinality(2 myns:isManagerOf myns:Man) ObjectMinCardinality(2 myns:isManagerOf myns:Woman)))
5) EquivalentClasses(myns:RomanEmployee ObjectIntersectOf(myns:Employee ObjectSomeValuesFrom(myns:worksInDivision ObjectHasValue(myns:locatedIn myns:Rome)) ObjectHasValue(myns:LivesIn myns:Rome)))