# Programming Assignment 3: Building an OWL Ontology

Amey Bhilegaonkar

Q1. Do any of your classes come out as inconsistent? Explain why and describe a way to resolve the inconsistency?

Ans: Yes, 2 of the classes came out as inconsistent when we added disjoints. When we introduced the topping MeatyVegetableTopping and assigned it to both MeatTopping and VegetableTopping categories, we observed inconsistency in the classes. This is because the same topping cannot be assigned to two different topping categories.
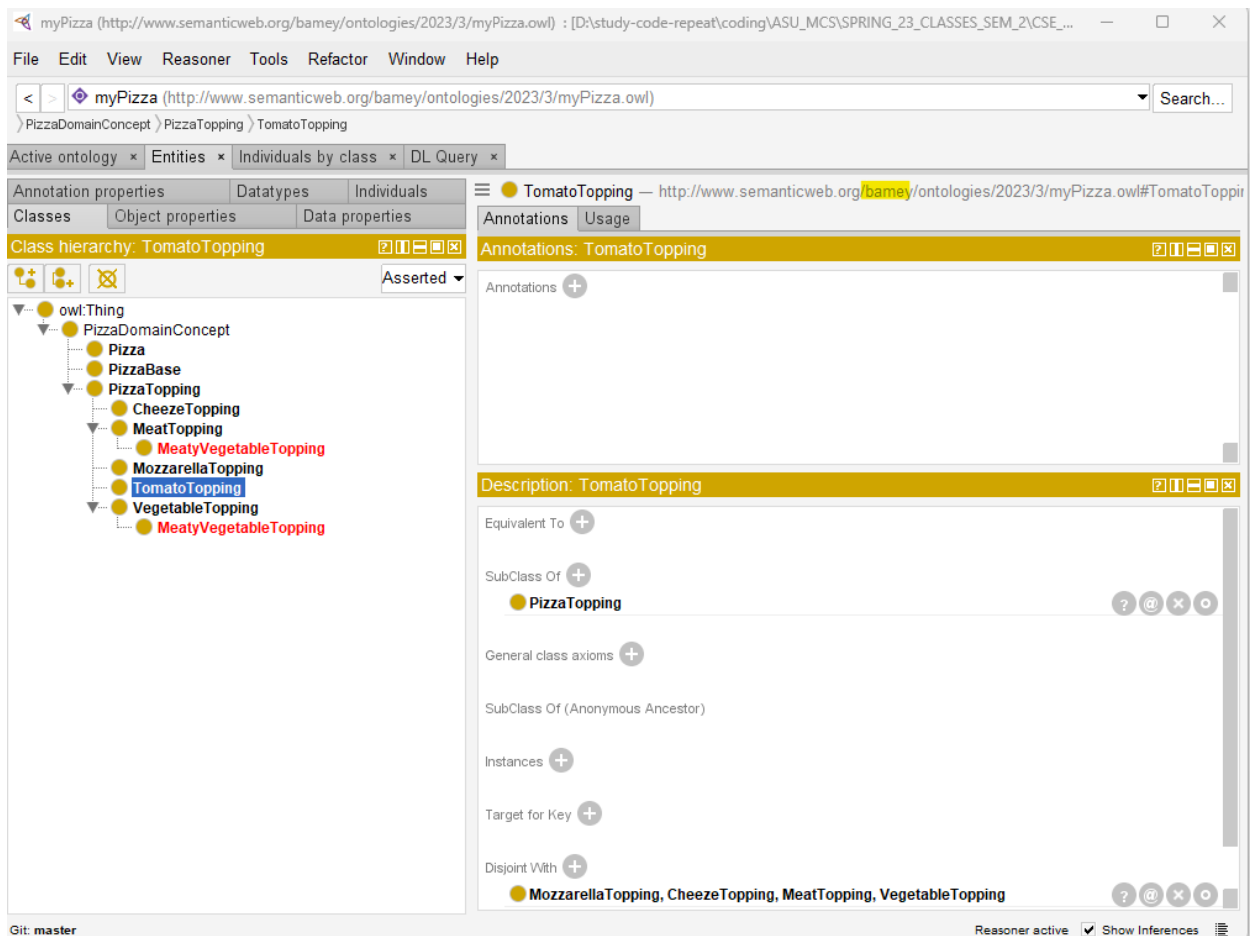
Explanation of inconsistency:

We were trying to make things a bit more organized by creating a new category called "MeatyVegetableTopping" that would be a subcategory of both "MeatTopping" and "VegetableTopping". But then, we ran into a bit of a problem when we tried to make "MeatTopping" and "VegetableTopping" totally separate from each other. You see, if we made "MeatyVegetableTopping" totally separate from both "MeatTopping" and "VegetableTopping", then we'd also be making it separate from itself! And that

just doesn't make sense. So basically, we've got an inconsistency on our hands. It's not ideal, but it's something we'll need to address and figure out a solution for.

Explanation for MeatyVegetableTopping EquivalentTo owl:Nothing

○ Show regular justifications   ● All justifications
○ Show laconic justifications   ○ Limit justifications to
                                        2

Explanation 1   ☐ Display laconic explanation

Explanation for: MeatyVegetableTopping EquivalentTo owl:Nothing
    MeatyVegetableTopping **SubClassOf** MeatTopping
    MeatyVegetableTopping **SubClassOf** VegetableTopping
    DisjointClasses: CheeseTopping, MeatTopping, MozzarellaTopping, TomatoTopping, VegetableTopping

OK

Resolution: To fix the problem we encountered with the MeatyVegetableTopping category, one possible solution would be to adjust the way we've organized our classes. Instead of trying to make MeatTopping and VegetableTopping completely separate from each other, we could make them disjoint only from other types of toppings.

This means that while MeatTopping and VegetableTopping would still be separate categories, they wouldn't be completely separate from each other. This would allow us to include the MeatyVegetableTopping category as a subcategory of both MeatTopping and VegetableTopping, without causing any logical inconsistencies.

File   Edit   View   Reasoner   Tools   Refactor   Window   Help

< > ◈ myPizza (http://www.semanticweb.org/bamey/ontologies/2023/3/myPizza.owl)

> PizzaDomainConcept > Pizza > NamedPizza > PunePizza

| Active ontology × | Entities × | Individuals by class × | DL Query × |

| Classes | Object properties | Data properties | Annotation properties | Datatypes | Individuals |

Class hierarchy: PunePizza

Asserted ▾

- ▼ ● owl:Thing
  - ▼ ● PizzaDomainConcept
    - ▼ ● Nationality
      - ● Italian
      - ● Indian
      - ● American
    - ● Meat
    - ● Crunch
    - ● Crust
    - ● Spice
    - ▼ ● Pizza
      - ⊜ MeatyPizza
      - ▼ ⊜ NamedPizza
        - ⊜ PaneerPizza
        - ⊜ **PunePizza**
        - ● AmericanPizza
        - ● SohoPizza
        - ● MargheritaPizza
    - ● PizzaBase
    - ▼ ● PizzaTopping
      - ▼ ● IndianTopping
        - ● AlooTopping
        - ● SpicyTopping
      - ▼ ● MeatTopping
        - ● PepperoniTopping
        - ● MeatyVegetableTopping
      - ● CheezeTopping
      - ● MozzarellaTopping
      - ● TomatoTopping
      - ▼ ● VegetableTopping
        - ● ParmezanTopping
        - ● OliveTopping
        - ● MeatyVegetableTopping

≡ ● PunePizza — http://www.semanticweb.org/bamey/ontologies/2023/

| Annotations | Usage |

Annotations: PunePizza

Annotations ⊕

Description: PunePizza

Equivalent To ⊕
- ● hasNationality **some** Indian
- ⊜ PaneerPizza

SubClass Of ⊕
- ● hasSpice **some** SpicyTopping
- ● hasTopping **some** IndianTopping
- ● NamedPizza

General class axioms ⊕

SubClass Of (Anonymous Ancestor)
- ● hasBase **some** PizzaBase

Instances ⊕

Target for Key ⊕

Disjoint With ⊕

Disjoint Union Of ⊕

Descriptions:

I added following properties:

1. hasCrunch: Its corresponding class has 3 choices, soft, medium and hard crunch
2. hasCrust: Its corresponding choices are Lemon and Garlic Crust
3. hasNationality: Its corresponding choices are Indian, American, Italian
4. hasSpice: It choices are it has, spice toppings.

I added following Toppings:

1. Indian Toppings
2. Spicy Toppings

I added following Bases:

1. White Base
2. Wheat Base

I added following Pizzas with properties:

1. PunePizza:
   a. hasNationality: Indian
   b. hasCrust: Lemon
   c. hasSpice: Spicy Topping, Indian Topping
   d. hasCrunch: Medium Crunch
2. PaneerPizza:
   a. hasNationality: Indian
   b. hasCrust: Garlic
   c. hasSpice: Aloo Topping, Indian Topping
   d. hasCrunch: Hard Crunch

Q6. First query of your own choice.

Ans: NamedPizza and hasNationality some Indian and hasTopping some IndianTopping

Q6-1. Explanation of the First query.

Ans: This query finds all the NamedPizza types which have Indian Nationality, and only those which have IndianTopping.

Q6-2. Screenshot of the First query.

Ans:

Q7. Second query of your own choice.

Ans: hasCrust some LemonCrust

Q7-1. Explanation of the Second query.

Ans: This query finds all the hasCrust types which have LemonCrust

Q7-2. Screenshot of the Second query.

Ans: