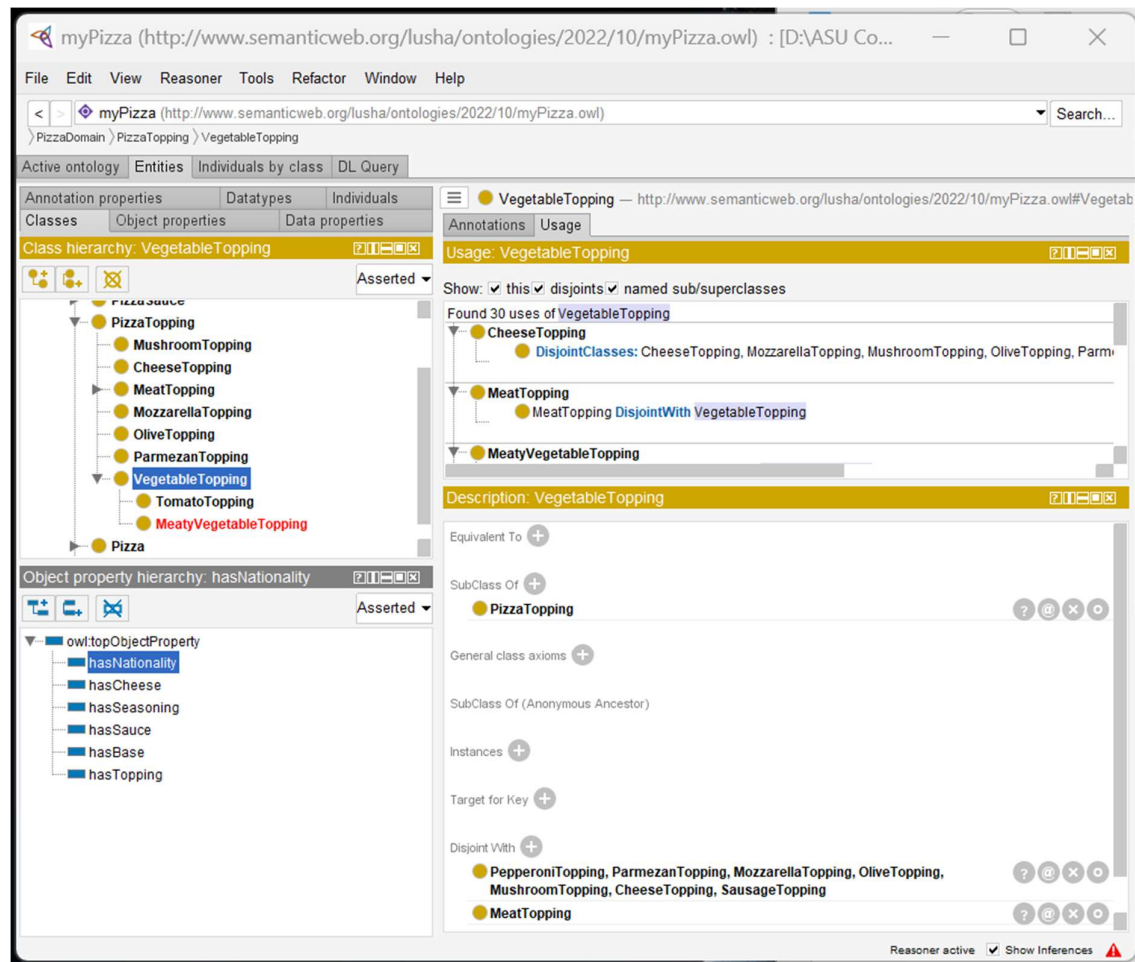# Programming Assignment 3 - Building an OWL Ontology

Ques 1: Do any of your classes come out as inconsistent?

Yes, I could find the inconsistencies in MeatyVegetableTopping as it was the one that was glowed in red.

The MeatVegetableTopping was unable to be added on both Vegetable and Meat Toppings initially, but they could be added in its description directly. Then we have added some disjoints in the Toppings Class. These Topping Classes were disjoint of one another. That raised an issue while running Reasoner and shown in the picture above.

One of the ways to overcome this type of inconsistency is to remove the disjoint between meat and vegetable toppings, that overlap with one another. Another way is to remove the MeatVegetableTopping altogether but add them conditionally on the pizza it is supposed to be present as both meat and vegetable should be present in subclasses function.

Ques 6 and 7: Execute at least two (non-trivial) DL queries of your own choice.

1. The first DL query is –

hasTopping some MeatTopping

This will check from the 'hasTopping' object and linked that with the class 'MeatTopping' that can or cannot have (at least could have) some Meat as their toppings on their Pizza.

The results we have found are shown below. The MeatyPizza is a direct subclass of the expression mentioned above which has the exact expression mentioned above. But the AmericanPizza and ClassicMeatPizza are just subclasses because it doesn't directly involve MeatTopping but involve it's subparts like PepperoniTopping and SausageTopping.

2. The second DL query is –

hasSauce some PestoSauce

This will check from the 'hasSauce' object and linked that with the class 'PestoSauce' that can or cannot have (at least could have) some Pesto as their sauce on their Pizza.

Here, the direct subclasses and the subclasses has the same values, because both the AmericanPizza and VeganPizza directly have this condition and they could have them or not if they chose to.

3. The third DL query is –

hasNationality only American

This expression would check for the 'hasNationality' object property and specifically checks if they are all American origin, so having 'America' Nationality.

Here, the direct subclasses and the subclasses has the same values, because both the AmericanPizza, SohoPizza and VeganPizza directly have this condition and they should all be having them.