Experiment No. 1B
Web X.0 Lab

Aim: To study Semantic Web Tool.

Case Study: A Case Study on Semantic Web Search using Ontology Modeling

Abstract

The Semantic Web works on the existing Web which presents the meaning of information as well-defined vocabularies understood by the people. Semantic Search, at the same time, works on improving the accuracy of a search by understanding the intent of the search and providing contextually relevant results. The paper describes a semantic approach towards web search through a PHP application. The goal was to parse through a user's browsing history and return semantically relevant web pages for the search query provided. The browser used for this purpose was Mozilla Firefox. The user's history was stored in a MySQL database, which, in turn, was accessed using PHP. The ontology, created from the browsing history, was then parsed for the entered search query and the corresponding results were returned to the user providing a semantically organized and relevant output.

Introduction

The Semantic Web is, necessarily, a vision for the future of the World Wide Web where the available information is given a meaning, providing logical connections of terms and making it easier of the machine to integrate data and process the information available on the Web

In the paper published by Doms A. and Schroeder M. in 2002, the first semantic search engine for biomedical texts using the Gene Ontology was published.

These semantic search engines are designed to search for information in the World Wide Web. The objective of this research is to provide the user with semantically relevant results for the entered search query based on the browsing history of the user. Swoogle is a crawler-based

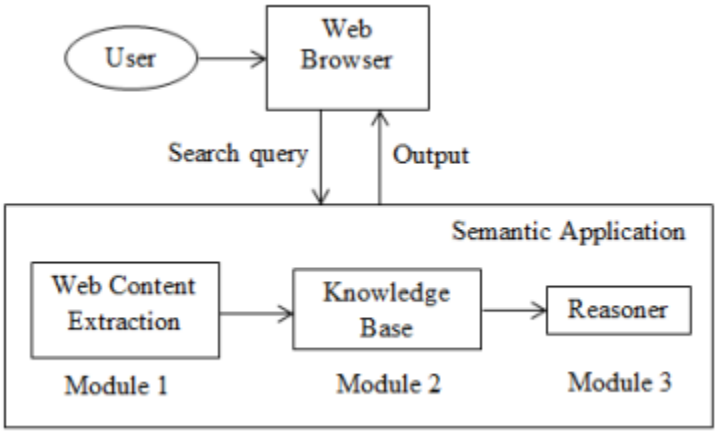
indexing and retrieval system that searches for Semantic Web documents, instance data, terms, ontologies, etc. published on the Web.

Architecture

The research had been carried out in three modules:

1. Web Content Extraction
2. Semantic Knowledge Base
3. Reasoner

The architecture of the system, in accordance to the modules, is shown below



Architecture of the system

Here, the user enters the search query into the semantic application. The semantic application in itself consists of the three modules, wherein, the browsing history of the user was contained in module 1, the ontology created in accordance to the browsing history was contained in module 2 and module 3 parsed through the created ontology to return the result. This result was then displayed to user through the semantic application on the browser

Methodology

1. **Web Content Extraction**
   The coding for this module was done using python. Also, the browser under consideration was Mozilla Firefox. Mozilla Firefox stores all its data into a SQLite database. The following figure shows the SQLite Manager that was used to access the browsing history from the Mozilla Firefox web browser
   From the moz.places database, the user's browsing history was accessed and stored into a MySQL database using python, the python packages used being sqlite3 and MySQLdb

   The code selects values from the sqlite3 database and stores it into a MySQL database, to be accessed in the PHP application. This python code was made to run in the background by adding a scheduler (the sched package) for the code to execute every 15 minutes. The web pages visited, in the meantime, would get updated into the MySQL table in intervals of 15minutes.

   *Other python packages like **BeautifulSoup4** and **NLTK** were used to extract the details of the visited URLs, like the title, description, etc. from the Web and were in turn stored into the MySQL database*

2. **Semantic Knowledge Base**

   The semantic knowledge base creation was done using the Protégé-OWL editor. Since the knowledge base domain, considered here, was "Apple", the ontology so created consisted of subclasses AppleInc (for the company) and AppleFruit (for the fruit). The various URLs obtained from Module 1 were added into the ontology as individuals for the corresponding classes, specifying the various schema features, OWL property restrictions, annotation properties, etc.

   The property restriction allValuesFrom was used to differentiate between the URLs of the two Apple subclasses, where the subclasses AppleInc and AppleFruit were defined to be disjoint with each other. The various restrictions applied on the classes in the ontology is shown in the figure below.

3. **Reasoner**

   The ontology was queried using DL query, which displays the corresponding classes, subclasses, individuals, etc. pertaining to the query. This query retrieved information from the ontology as created in the previous module

   The ontology, saved as an OWL file, was accessed, parsed and queried in the PHP application, where the query entered by the user was passed as the query to the file. The result obtained was displayed by the application, providing a semantically relevant list of URLs for the entered search query from the user's list of visited URLs.

   The semantic application was provided with user login, thereby keeping the application and the database more user-specific

Result and Further work

The application, hence, aids the search of the user by providing a list of useful and relevant web pages earlier visited by him, in accordance to the query, thereby providing him with results that would be relevant and helpful.

The major area of further work in the system would be to make the application work for web browsers other than Mozilla Firefox. Accessing databases of Google Chrome, Safari, Internet Explorer, etc. and storing them in the MySQL database along with the existing database would make the application work for the other web browsers as well.

Conclusion

With the increasing amount of data being stored on the Web every day, Semantic Search would ensure the provision of useful and relevant information to the user. This application works on making the procedure more user-friendly by providing contextually relevant visited pages to the user along with the searches available to the

user otherwise through the various search engines. Understanding the user's query and providing just the content required is the objective of the application.

Description:

Semantic web

The Semantic Web, Web 3.0, the Linked Data Web, the Web of Data, it represents the next major evolution in connecting and representing information. It enables data to be linked from a source to any other source and to be understood by computers so that they can perform increasingly sophisticated tasks on our behalf.

From a technical point of view, the Semantic Web consists primarily of three technical standards:

1. **RDF** (Resource Description Framework) - The data modeling language for the Semantic Web. All Semantic Web information is stored and represented in the RDF.

2. **SPARQL** (SPARQL Protocol and RDF Query Language) - The query language of the Semantic Web. It is specifically designed to query data across various systems.

3. **OWL** (Web Ontology Language) - The schema language, or knowledge representation (KR) language, of the Semantic Web. OWL enables you to define concepts composably so that these concepts can be reused as much and as often as possible. Composability means that each concept is carefully defined so that it can be selected and assembled in various combinations with other concepts as needed for many different applications and purposes.

Semantic Web technologies as a whole have made tremendous strides in the last decade. Some highlights include:

1. The Open Linked Data movement has grown massively every single year and contains far more information than any single resource anywhere on the Web.
2. Large organizations, such as Merck, Johnson & Johnson, Chevron, Staples, GE, the US Department of Defense, NASA, and others now rely on Semantic Web technologies to run critical daily operations.
3. The Semantic Web standards viz. RDF, SPARQL, OWL, and others were merely drafts in 2001, but they have now been formalized and ratified.
4. Truly, an entire industry has been born in the past ten years, complete with multiple trade shows on several continents, a growing user community, and active standards bodies.

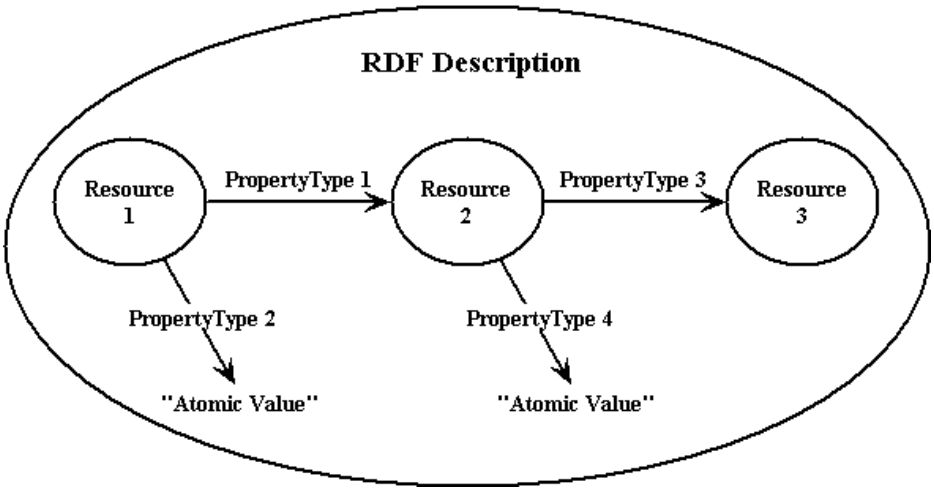
That said, significant room for growth still can be found.

1. Despite recent huge strides on the part of Schema.org, Facebook's Open Graph, and others, the vision of an entire Web of interoperable data has still not yet been realized.
2. Notwithstanding significant early corporate adoption by a select few frontrunners, most companies have not yet started using Semantic Web technologies.

3. The learning curve for using Semantic Web technologies is perceived to be steep because few educational resources currently exist for users new to the concepts, and still fewer resources can be found that discuss when and how to apply the technologies to real world scenarios.

RDF (Resource Description Framework)

RDF is a standard model for data interchange on the Web. RDF has features that facilitate data merging even if the underlying schemas differ, and it specifically supports the evolution of schemas over time without requiring all the data consumers to be changed.
RDF extends the linking structure of the Web to use URIs to name the relationship between things as well as the two ends of the link (this is usually referred to as a "triple"). Using this simple model, it allows structured and semi-structured data to be mixed, exposed, and shared across different applications.
This linking structure forms a directed, labeled graph, where the edges represent the named link between two resources, represented by the graph nodes. This graph view is the easiest possible mental model for RDF and is often used in easy-to-understand visual explanations.



Knowledge graphs, represented in RDF, provide the best framework for data integration, unification, linking and reuse, because they combine:

1. **Expressivity** - The standards in the Semantic Web stack – RDF(S) and OWL – allow for a fluent representation of various types of data and content: data schema, taxonomies and vocabularies, all sorts of metadata, reference and master data. The RDF* extension makes it easy to model provenance and other structured metadata.

2. **Formal semantics** - All standards in the Semantic Web stack come with well specified semantics, which allow humans and computers to interpret schema, ontologies and data in unambiguous manner.

3. **Performance** - All the specifications have been thought out, and proven in practice, to allow for efficient management of graphs of billions of facts and properties.

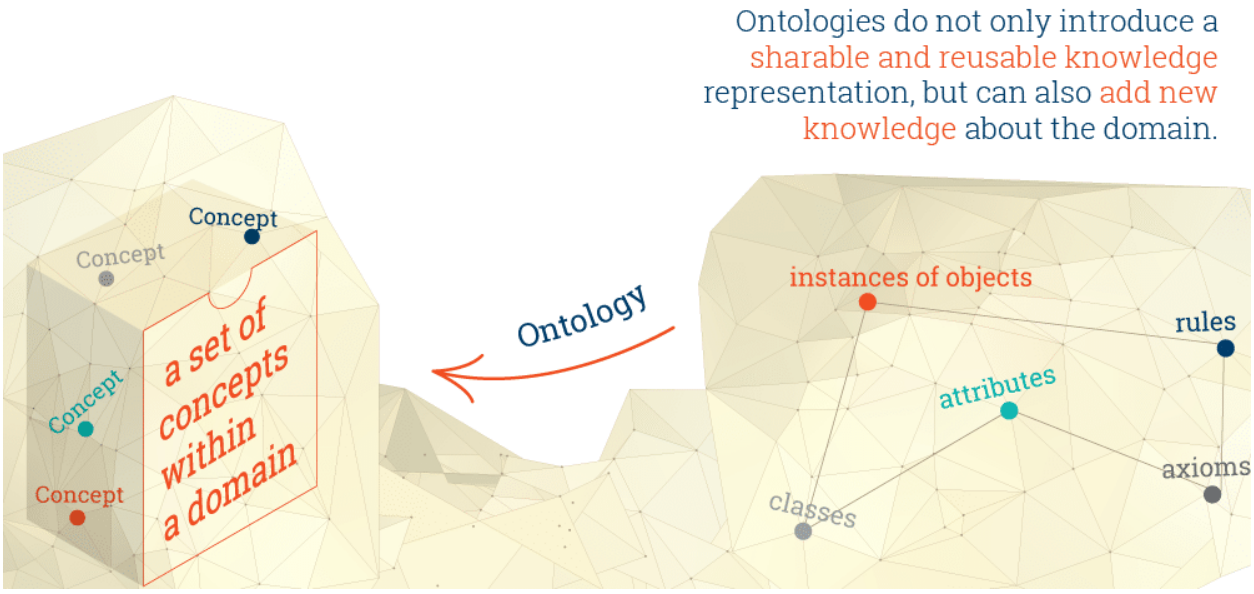
4. **Interoperability** - There is a range of specifications for data serialization, access (SPARQL Protocol for end-points), management (SPARQL Graph Store) and federation. The use of globally unique identifiers facilitates data integration and publishing.

5. **Standardization** - All the above is standardized through the W3C community process, to make sure that the requirements of different actors are satisfied – all the way from logicians to enterprise data management professionals and system operations teams.

Often RDF is criticized because it doesn't allow for descriptions or properties to be attached to the edges in the graph and this is perceived as a disadvantage compared to Property Graphs. This concern has been addressed with RDF-Star (abbreviated RDF*), which allows one to make statements about other statements and this way to attach metadata to the edges in the graph

Ontology

In computer science and information science, an ontology encompasses a representation, formal naming and definition of the categories, properties and relations between the concepts, data and entities that substantiate one, many, or all domains of discourse. More simply, an ontology is a way of showing the properties of a subject area and how they are related, by defining a set of concepts and categories that represent the subject.



Currently, most of the technologies that employ data modeling languages (like SQL) are designed using a rigid "Build the Model, then Use the Model" mindset.

For example, suppose you want to change a property in a relational database. You had previously thought that the property was single-valued, but now it needs to be multi-valued. For almost all modern relational databases, this change would require you to delete the entire column for that property and then create an entirely new table that holds all of those property values plus a foreign key reference.

This is not only a lot of work, but it will also invalidate any indices that deal with the original table. It will also invalidate any related queries that your users have written. In short, making that one change can be very difficult and complicated. Often, such changes are so troublesome that they are simply never made.
By contrast, all data modeling statements (along with everything else) in ontological languages for data are incremental, by their very nature. Enhancing or modifying a data model after the fact can be easily accomplished by modifying the concept.

SPARQL

SPARQL is the standard query language and protocol for Linked Open Data and RDF databases. Having been designed to query a great variety of data, it can efficiently extract information hidden in non-uniform data and stored in various formats and sources.

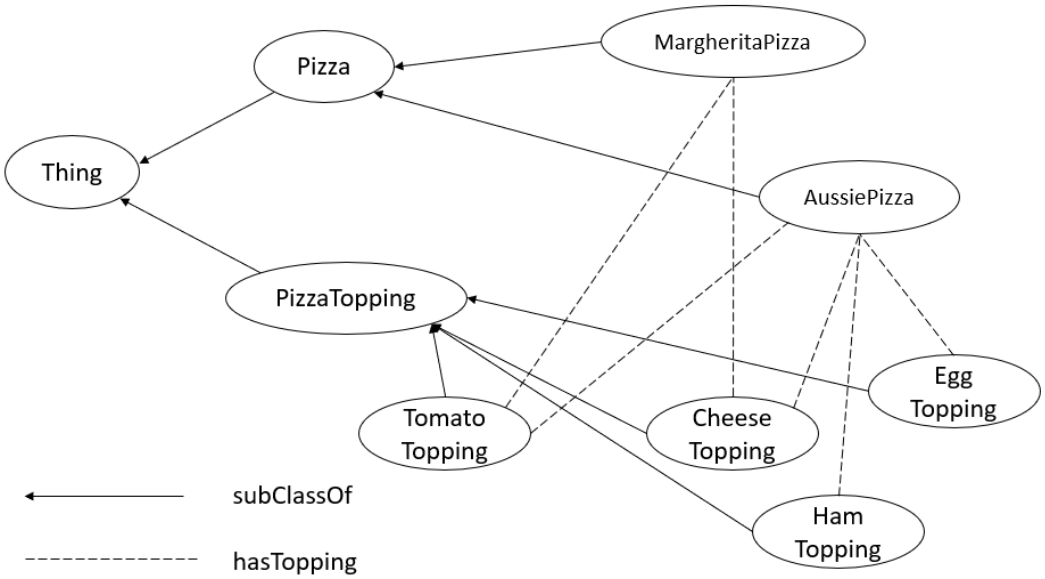
Just like SQL allows users to retrieve and modify data in a relational database, SPARQL provides the same functionality for NoSQL graph databases like Ontotext's GraphDB.
In addition, a SPARQL query can also be executed on any database that can be viewed as RDF via a middleware. For example, a relational database can be queried with SPARQL by using a Relational Database to RDF mapping software.

SPARQL has four types of queries. It can be used to:

1. ASK whether there is at least one match of the query pattern in the RDF graph data;
2. SELECT all or some of those matches in a tabular form (including aggregation, sampling and pagination through OFFSET and LIMIT);
3. CONSTRUCT an RDF graph by substituting the variables in these matches in a set of triple templates;
4. DESCRIBE the matches found by constructing a relevant RDF graph.
5. The leading semantic graph databases that support SPARQL have intuitive SPARQL editors with autocomplete, explorer and many other features that facilitate building powerful SPARQL queries.

Pizza ontology

Creating a simple Pizza ontology in RDF and OWL using Topbraid Composer
The figure below shows a suggested information model for a simple Pizza ontology. We have 2 main classes, called Pizza and PizzaTopping and we'll be creating 2 pizza types - the classic, MargheritaPizza and the AussiePizza



In OWL, we use Description Logic to capture class semantics. We use class restrictions to narrow down the possible logical statements about that class and their set of instances. Using the MargheritaPizza example, we know that it is a pizza that has cheese and tomato toppings. To express this, we create a restriction on the subClassOf property for MargheritaPizza with the following:

1. There exists instances of Pizza where the hasTopping property is CheeseTopping
2. There exists instances of Pizza where the hasTopping property is TomatoTopping

These restrictions can then be used in software reasoners to infer a list of pizzas which have no TomatoToppings, which would include the BiancaPizza. Similarly, we can use software reasoners to infer a list of pizzas which have CheeseToppings, which would include the BiancaPizza, AussiePizza and the MargheritaPizza.

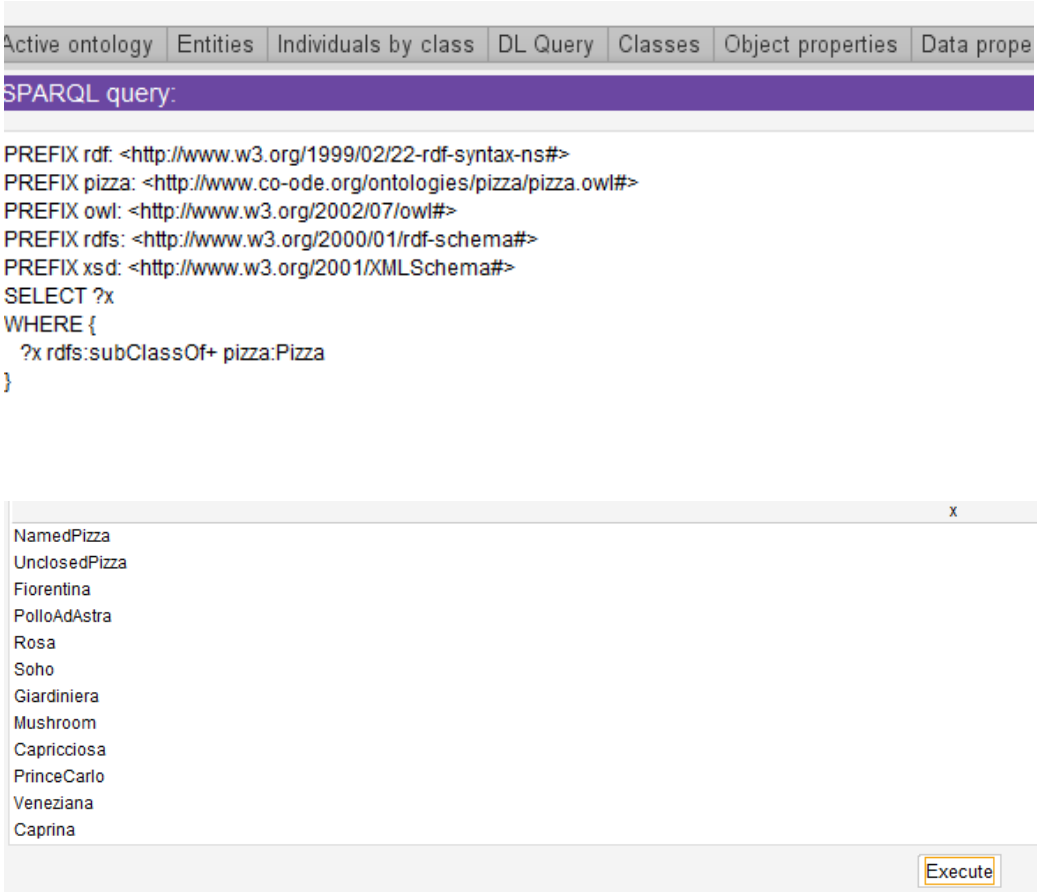
Output

Prefixes used

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX pizza: <http://www.co-ode.org/ontologies/pizza/pizza.owl#>

1. Query the Pizza ontology and list all subclasses of pizza:Pizza (direct and indirect)

```
SELECT ?x
WHERE {
  ?x rdfs:subClassOf+ pizza:Pizza
}
```



2. Pizzas with cheese topping

```
SELECT ?X ?topping WHERE {
   ?X rdfs:subClassOf ?Y .
   ?Y owl:someValuesFrom ?topping .
   ?topping rdfs:subClassOf* pizza:CheeseTopping
}
ORDER BY ?X
```

SPARQL query:

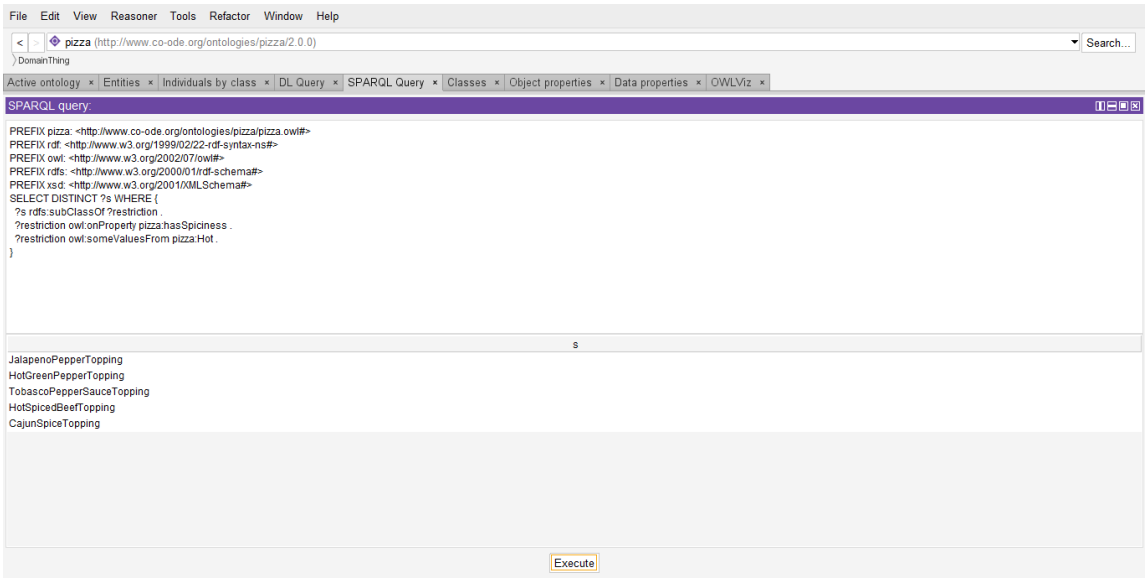
```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX pizza: <http://www.co-ode.org/ontologies/pizza/pizza.owl#>

SELECT ?X ?topping WHERE {
   ?X rdfs:subClassOf ?Y .
   ?Y owl:someValuesFrom ?topping .
   ?topping rdfs:subClassOf* pizza:CheeseTopping
}
ORDER BY ?X
```

X	topping
American	MozzarellaTopping
AmericanHot	MozzarellaTopping
Cajun	MozzarellaTopping
Capricciosa	MozzarellaTopping
Caprina	MozzarellaTopping
Caprina	GoatsCheeseTopping
Fiorentina	ParmezanTopping
Fiorentina	MozzarellaTopping
FourSeasons	MozzarellaTopping
Giardiniera	MozzarellaTopping
LaReine	MozzarellaTopping
Margherita	MozzarellaTopping

Execute

3. Check for everything that is related to hot spiciness

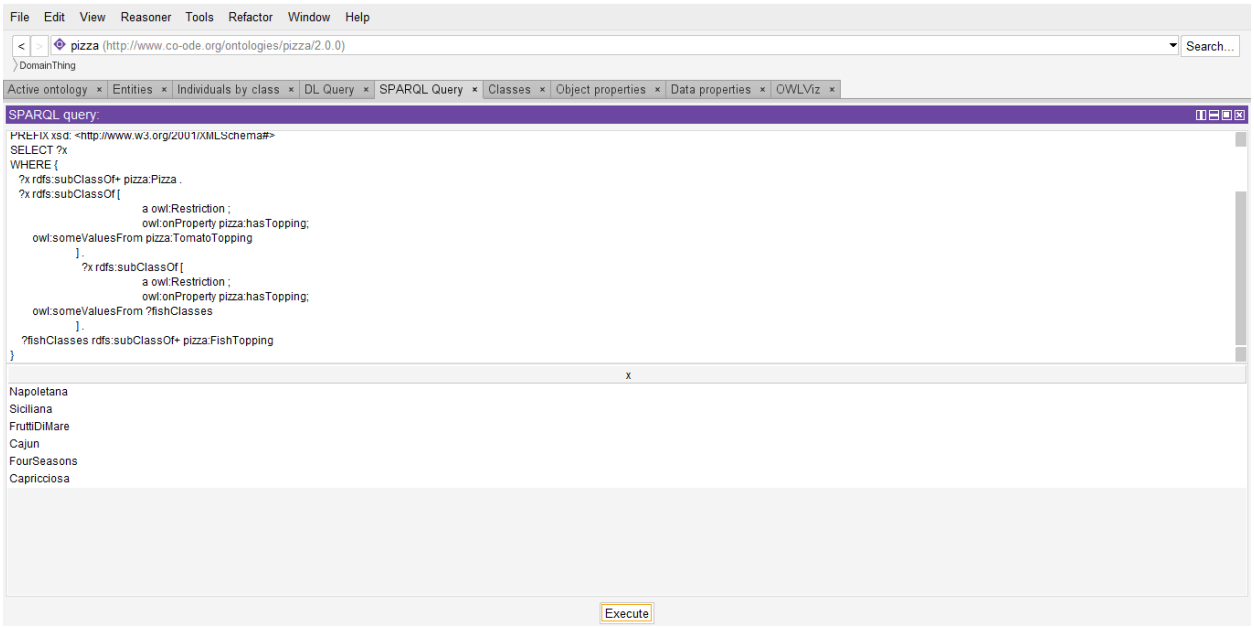
```
SELECT DISTINCT ?s WHERE {
 ?s rdfs:subClassOf ?restriction .
 ?restriction owl:onProperty pizza:hasSpiciness .
 ?restriction owl:someValuesFrom pizza:Hot .
}
```



4. Find all Pizza classes that have TomatoTopping and all types of classes of FishTopping

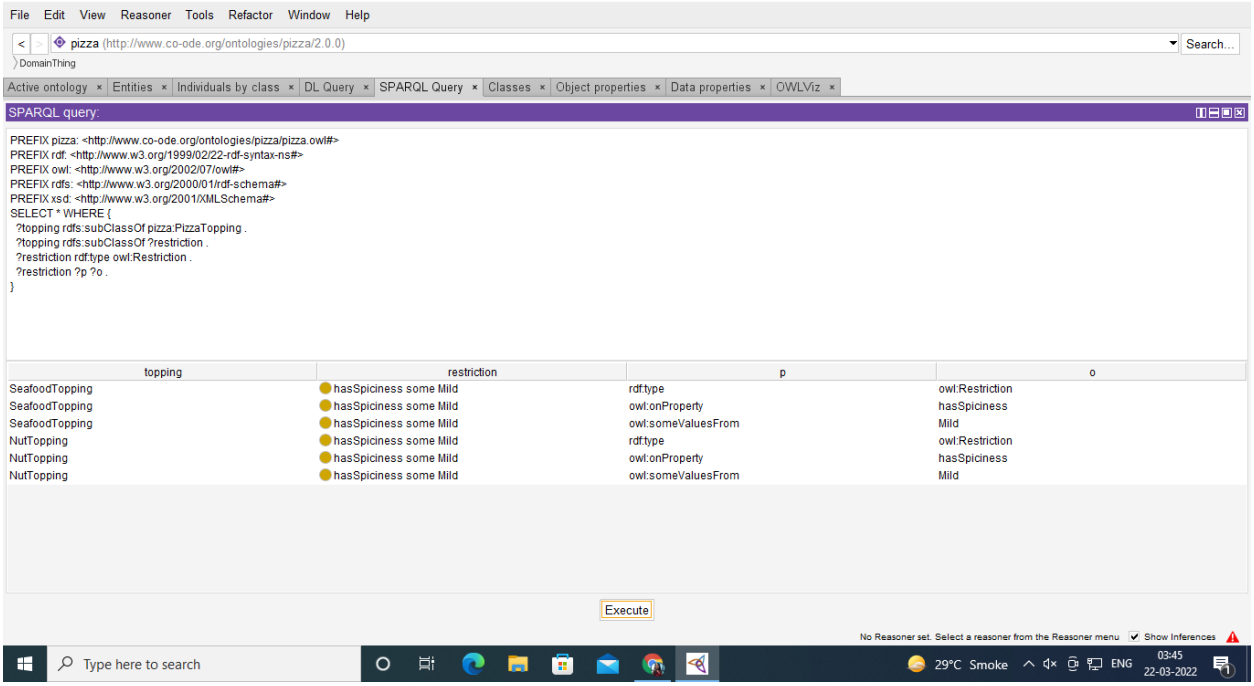
```
SELECT ?x
WHERE {
  ?x rdfs:subClassOf+ pizza:Pizza .
  ?x rdfs:subClassOf [
                a owl:Restriction ;
                owl:onProperty pizza:hasTopping;
    owl:someValuesFrom pizza:TomatoTopping
      ] .
       ?x rdfs:subClassOf [
                a owl:Restriction ;
                owl:onProperty pizza:hasTopping;
    owl:someValuesFrom ?fishClasses
```

```
        ] .
  ?fishClasses rdfs:subClassOf+ pizza:FishTopping
}
```



5. A query that lists all mild toppings

```
SELECT * WHERE {
 ?topping rdfs:subClassOf pizza:PizzaTopping .
 ?topping rdfs:subClassOf ?restriction .
 ?restriction rdf:type owl:Restriction .
 ?restriction ?p ?o .
}
```



Conclusion:

Thus we understood what Semantic Web Tool is, why Semantic Web Tool is important and how to use Semantic Web Tools and get hands-on experience on Protege.