COMP 6721 Applied Artificial Intelligence (Winter 2021)

Lab Exercise #09: Knowledge Graphs, Part II

Question 1 We'll continue the knowledge graph exercise from the previous week. Your next task is to develop a new *vocabulary* using RDF Schema (RDFS): The *Friends-of-Concordia-University (FOCU)* schema. This vocabulary should be able to express:

(a) Classes for `Student` and `Professor`, both of which are subclasses of `foaf:Person`

(b) A `University` class, which is a subclass of `foaf:Organization`

(c) A property describing that a `Student` is enrolled at a `University`

(d) A property describing that a `Professor` teaches `Students`

You can start with the following template (in Turtle format):

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix focu: <http://focu.io/schema#> .
@prefix focudata: <http://focu.io/data#> .

# RDF Schema for the FOCU Vocabulary

focu:[CLASSNAME]
  a rdfs:Class ;
  rdfs:label "[LABEL]"@en .

focu:[OTHERCLASSNAME]
  a rdfs:Class ;
  rdfs:subClassOf foaf:[CLASS] ;
  rdfs:label "[LABEL]"@en .

focu:[PROPERTY]
  a rdf:Property ;
  rdfs:label "[LABEL]"@en ;
  rdfs:comment "[COMMENT]"@en ;
  rdfs:domain uni:[DOMAIN_CLASS] ;
  rdfs:range uni:[RANGE_CLASS] .
```

(a) Add your RDFS triples for the four definitions above by filling in the missing parts in `[square brackets]`.

(b) Validate your new RDF schema using the tools mentioned in last week's lab and visualize them in form of a graph.

(c) Now add some triples using your new FOCU vocabulary that describe yourself as a student at Concordia. Add information about yourself (age, email) using the FOAF vocabulary as we did on the lecture Worksheet #8.

(d) Again, validate and visualize your triples.

Some notes:

- There are of course existing vocabularies for universities etc. and in a real system you would not create a new vocabulary duplicating existing definitions (just like we re-use FOAF to describe a person here). This is just for exercise purposes, so that you see how to define a minimal, working RDF Schema.

- We are using `focu.io` as the authority (domain name): This is again bad practice, since you do not control this domain name and thus cannot publish anything using this authority.

- The template above defines `focudata:` as the namespace for the triples describing university people; you could use another namespace here, like the `example.org` we've used before. The point here is that it is not good practice to mix schema information (in this case your FOCU schema) with data described using this schema into the same namespace. In other words, you want to be able to load the schema and the data separately from each other into a system.

- Can you add a triple that relates your new `University` definition to an existing concept in the FOAF vocabulary?

Question 2 Using your RDFLib program from the previous week, read the FOAF+FOCU triples about some people you created above. Next, your program should ask for a name and retrieve all the available information from the graph. For example, given the name Joe, it will print out all available triples like age, email, etc.:

```python
import rdflib
from rdflib import Literal, Namespace
from rdflib.namespace import FOAF, RDFS
```

Add the user-defined namespaces:

```python
# Create the user-defined namespaces
FOCU = Namespace("http://focu.io/schema#")
FOCUDATA = Namespace("http://focu.io/data#")
DBO = Namespace("http://dbpedia.org/ontology/")
OWL = Namespace("http://www.w3.org/2002/07/owl#")
DBP = Namespace("http://dbpedia.org/property/")
```

Here is the command to create a graph instance:

```python
g = rdflib.Graph()
```

Now parse the graph:

```python
g.parse("Lab9.ttl", format="turtle")
```

Use the below command to start accepting user input:

```python
user_input = input('Hello, I am your smart university agent. Who are you
looking for?')
```

```
Hello, I am your smart university agent. Who are you looking for?
> Joe

Joe is 22 years old and has the email address joe@example.com.
```

Implement searching and printing the information by going through the graph.

Hint: Use the RDFLib functions you learned in the previous lab to navigate the graph and retrieve the required information, i.e., age and email address. Here, the user inputs a person's first name, therefore find the triples matching the given first name and then proceed to retrieved the required information from the graph.

Next, if the person is enrolled at a university, your agent should determine in which city & country this person studies (using the triples about the university from DBpedia) and print out this information as well:

```
Hello, I am your smart university agent. Who are you looking for?
> Jane

Jane is 24 years old and studies at Concordia University,
which is located in Montreal, Canada.
```

Hint: The only new information we need here is the location. To obtain it, you can merge the DBpedia graph into your knowledge base and fetch all these additional information from there.

You can print out additional information about the university if you like.

Question 3 We'll now make the AI even smarter, by also supporting the query "What is `<X>`", where you retrieve triples about `<X>` from DBpedia, printing out the `comment` information, e.g.:

```
Hello, I am your smart university agent. How can I help you?
> What is Concordia University?

Concordia University (commonly referred to as Concordia) is a public
comprehensive university located in Montreal, Quebec, Canada.
Founded in 1974 following the merger of Loyola College and
Sir George Williams University...
```

To implement this, we'll start by checking whether a "What is `<X>`" pattern is present in the user's input:

```python
# Define pattern
pattern = "What is"

if pattern in user_input:
    # Get the entity name from the user's input
    input_user = user_input.split(pattern,1)[1].strip()
    uni_name = input_user.split("?",1)[0].strip()
    # Based on the entity name, get its description
```

Implement the following two steps:

(a) Create a DBpedia URI based on the input subject `<X>` and load the graph into your program. For example, if `<X>` is `Concordia University`, create the URI http://dbpedia.org/resource/Concordia_University.[1]

(b) Use the RDFLib functions as before to navigate and retrieve the required output from the graph.[2]

Notice how similar this is to an answer you'd get from a chatbot like Google Assistant or Alexa, Siri, etc.: When chatbots make use of a knowledge graph to answer questions, this is referred to as *grounding* (as opposed to pattern-matching as we've seen in bots like Eliza).

---

[1]Of course, this naïve approach of converting a string to a DBpedia URI does not always work in practice, but there are more sophisticated ways available of achieving this.

[2]Another way of implementing this would be to generate SPARQL queries, similar to how you've done it on Worksheet #08. However, we're not covering SPARQL in this lab.