Knowledge Engineering and Semantic Web

Exercise Sheet: 7

Will be discussed on: July 20,2021

Leibniz Universität Hannover

TUTORS:
Yaser Jaradeh, Salomon Kabenamualu, Vitalis Wiens

LECTURE SLIDES: The lecture slides can be accessed through the following link:

https://slidewiki.org/playlist/237

QUESTIONS: Please don't hesitate to ask any questions. Questions help you and your peers.
PRINT: Please consider the environment before printing the exercise.

Required Slides `https://slidewiki.org/deck/90732-2/07-owl-semantics-and-reasoning/`

1 ACL to NNF

Convert the following ACL axioms into Negation Normal Form (NNF)

- $\neg\left(A \sqcup \neg B\right)$
- $\neg(\neg(A \sqcup \neg B) \sqcap \neg C)$
- $\neg(A \sqsubseteq B) \sqcup (C \sqsubseteq D)$
- $\neg(\forall r.A \sqcup B)$

Required Slides `https://luh-kesw.github.io/SummerTerm2021/slides/KESW-SPARQL.pdf`

2 Review Questions

1. Which statements are true or false?
   (a) SPARQL stands for "SPARQL Protocol and RDF Query Language".
   (b) SPARQL endpoints expose only one graph.
   (c) SPARQL queries must have prefix definitions.
   (d) SPARQL queries must have the where clause.
   (e) All statements in a SPARQL must be closed by a '.'
   (f) SPARQL queries can only retrieve variables.
   (g) SPARQL responses are RDF triples.

3 Learning by Doing

Open the DBpedia endpoint in your browser : http://dbpedia.org/sparql/

1. Run the example query :
   SELECT DISTINCT ?Concept WHERE [] a ?Concept LIMIT 100

(a) Explain in your own the query. Particularly explain the individual commands. (SELECT,DISTINCT, WHERE, LIMIT)

(b) How could you extend / modify the query to get the next 10 entries.

2. Create a SPARQL query to find all triples about Nikola Tesla.

(a) Without using prefixes.

(b) Using prefixes

(c) How can you modify the query so the result will be provided in a triple format.

(d) Return the number of triples associated with Nikola Tesla.

(e) Create a SPARQL query that will return the individual properties and their counts (given the subject is Nikola Tesla.

(f) Create a SPARQL query that will return all different labels for Nikola Tesla

Consider the following knowledge base about people who work for an exemplary company and solve the tasks 2 to 4.

```
@prefix ex:<http://example.org#> .
@prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd:<http://www.w3.org/2001/XMLSchema#>.
@prefix owl: <http://www.w3.org/2002/07/owl#>.

ex:p1   ex:name          "William"@en;
        ex:salary        "23000"^^xsd:integer;
        ex:birthYear     "1989"^^xsd:integer;
        ex:friendWith    ex:p3;
        ex:knows         ex:p2,ex:p4;
        ex:workingStatus "fullTime";
        ex:jobTitle      ex:Programmer;
        ex:nationality   ex:American;
        ex:email         "william@fake.com";
        ex:workingProject ex:pr1.

ex:p2   ex:name           "Baerble"@de;
        ex:salary         "43000"^^xsd:integer;
        ex:birthYear      "1977"^^xsd:integer;
        ex:knows          ex:p1, ex:p3, ex:p4;
        ex:workingStatus  "fullTime";
        ex:jobTitle       ex:Manager;
        ex:nationality    ex:German;
        ex:workingProject ex:pr2.

ex:p3   ex:name           "Abdolreza"@de;
        ex:salary         "8000"^^xsd:integer;
        ex:birthYear      "1995"^^xsd:integer;ex:friendWith   ex:p1;
        ex:knows          ex:p2;
        ex:workingStatus  "partTime";
        ex:jobTitle       ex:Programmer;
        ex:nationality    ex:Libyan;
        ex:email          "abdolreza@fake.com";
        ex:workingProject ex:pr2.

ex:p4   ex:name           "Paul"@en;
```

```
        ex:salary         "24000"^^xsd:integer;
        ex:birthYear      "1963"^^xsd:integer;
        ex:knows          ex:p1, ex:p2;
        ex:workingStatus  "Retired";
        ex:jobTitle       ex:Manager;
        ex:nationality    ex:American;
        ex:workingProject x:pr2.

ex:pr1   a              ex:Project;
         ex:startYear   "2013"^^xsd:gYear;
         ex:supervisor  ex:p4;
         ex:headWorker  ex:p1.

ex:pr2   a             ex:Project;
         ex:supervisor ex:p2;
         ex:advisor    ex:p3.

ex:headWorker rdfs:subClassOf ex:Manager.

ex:friendWith rdfs:subPropertyOf ex:knows;
              a                  owl:symmetricProperty.
```

4 Explain the queries below in your own words and find their results.

1. ```
   PREFIX ex:<http://example.org#>
   SELECT ?name
   {
   ?p   ex:name   ?name;
        ex:salary   ?salary.
   FILTER(?salary>15000)}
   ```

2. ```
   PREFIX ex:<http://example.org#>
   ASK {
   ?person ex:name   ?name;
           ex:salary   ?salary;
           ex:nationality   ex:German .
   FILTER(?salary >= 40000)}
   ```

3. ```
   PREFIX ex:<http://example.org#>
   SELECT (COUNT(?name) as ?count)
   {
   ?p   ex:name   ?name;
        ex:workingStatus   ?stat.
   MINUS{?p   ex:workingStatus   "Retired"}
   OPTIONAL{?p   ex:email   ?email.}
   FILTER(!bound(?email))
   }
   ```

4. ```
   PREFIX ex:<http://example.org#>
   SELECT (SUM(?salary) as ?sum)
   {
   ?p   ex:salary   ?salary;
   {?p   ex:workingStatus   ?status.
   FILTER(?status="partTime")} UNION
   {?p   ex:workingStatus   ?status.
   FILTER(?status="fullTime")}
   }
   ```

5. ```
   PREFIX ex:<http://example.org#>
   SELECT DISTINCT ?p ?job ?name2
   {
   ```
   ```

```
?p   ex:name   ?name;
     ex:jobTitle   ?job;
     ex:knows   ?p2.
?p2   ex:name   ?name2.
FILTER(lang(?name2)="en")
}
```

5 Write SPARQL queries to answer the following requests.

1. The average age of all Working Employees in the year 2016.
2. The salary and email (if it's given) of American employees.
3. Names of people with a salary of less than 20,000 who are not American.
4. Names of supervisors of projects which American people work in.
5. Does any American worker aged over 30 works for the company who is payed more than 30000 annually?