Foundations of Semantic Web Technologies

Tutorial 5

Sebastian Rudolph, Lukas Schweizer

SS 2017

Exercise 5.1. Consider the following RDF document with information about celestial bodies.

```
@prefix  ex: <http://example.org/> .
ex:sun    ex:radius "1.392e6"^^xsd:double ;
          ex:satellite    ex:mercury , ex:venus ,
                           ex:earth ,    ex:mars   .
ex:mercury  ex:radius    "2439.7"^^xsd:double  .
ex:venus    ex:radius    "6051.9"^^xsd:double  .
ex:earth    ex:radius    "6372.8"^^xsd:double  ;
            ex:satellite  ex:moon .
ex:mars     ex:radius    "3402.5"^^xsd:double  ;
            ex:satellite  ex:phobos, ex:deimos  .
ex:moon     ex:name      "Mond@de", "Moon@en"  ;
            ex:radius    "1737.1"^^xsd:double  .
ex:phobos   ex:name      "Phobos" .
ex:deimos   ex:name      "Deimos" .
```

Specify SPARQL queries which yield the following results in the form of a table.

- Objects which orbit around the sun or around a satellite of the sun.

- Objects with a volume greater than $2 \cdot 10^{10}$ (km^3) together with the object – if it exists – of which they are a satellite. Assume for this that all celestial bodies are spherical.

- Objects with a satellite for which an English name is given, and which furthermore are satellites of an object with diameter greater than 3000 (km).

- Objects with two or more satellites. Assume for this that different URIs denote different objects.

Exercise 5.2. Translate the queries from Exercise 5.1 into expressions into SPARQL algebra.

1

Exercise 5.3. Compute the solutions to the expressions from Exercise 8.2 with respect to the knowledge base from Exercise 8.1 step by step.

Exercise 5.4. The game *Sudoku* is about completing incomplete tables with numbers while respecting certain rules. We consider the following simple 4×4 Sudoku:

			3
			4
2			
3			

You have to fill in numbers with values 1 to 4 in the empty slots in the table so that no number occurs twice in any row or any column, and so that no number is duplicated within any of the marked 2×2 squares.

We now want to use SPARQL for solving this Sudoku, i.e. we want to obtain all possible solutions by means of answers to a SPARQL query. In order to do this, set up a suitable RDF document and SPARQL query.

Exercise 5.5. This exercise focuses on the use of modifiers in SPARQL. Consider the following RDF document:

```
@prefix  ex: <http://example.org/> .
ex:a   ex:value  "1"^^xsd:integer ;
       ex:value  "3"^^xsd:integer .
ex:b   ex:value  "2"^^xsd:integer .
```

Which result would each of the following SPARQL queries return for this RDF input?

(a)
```
SELECT ?s ?v
   WHERE { ?s <http://example.org/value> ?v }
   ORDER BY ?v
```

(b)
```
SELECT ?s
   WHERE { ?s <http://example.org/value> ?v }
   ORDER BY ?v
```

(c)
```
SELECT ?s
   WHERE { ?s <http://example.org/value> ?v }
   ORDER BY DESC(?v) LIMIT 2
```

(d)
```
SELECT DISTINCT ?s
   WHERE { ?s <http://example.org/value> ?v }
   ORDER BY ?v
```

Which result would you expect the last query to return when `LIMIT 1` is added?