EXERCISES ON SPARQL

Use DBpedia SPARQL endpoint to solve the exercises : http://dbpedia.org/sparql

EXERCISE 1

Find 50 example concepts in the DBpedia dataset.

SELECT DISTINCT ?concept WHERE { ?s a ?concept . } LIMIT 50

EXERCISE 2

Find the resource of Madonna (the singer). Tip: don't forget to use the language tag in your query: "Madonna"@en

SELECT DISTINCT * WHERE {?x ?y "Madonna"@en }

EXERCISE 3

Describe Madonna using one of the URIs retrieved by the previous query.

DESCRIBE <http://dbpedia.org/resource/Madonna_(entertainer)>

EXERCISE 4

The behavior of a DESCRIBE query is up to the server. Create a query to retrieve all properties around Madonna. In other words: all predicates and objects where she is the subject, and all predicated and subjects where she is the object.

One solution is to use unions:

```
SELECT DISTINCT ?property ?value {
        { <http://dbpedia.org/resource/Madonna_(entertainer)> ?property ?value. }
        UNION
        { ?value ?property <http://dbpedia.org/resource/Madonna_(entertainer)>. }
}
```

The problem, however, is that one 'loses' the direction of the properties. The following query adds "is … of" to a property of which

Madonna was the object.

```
SELECT DISTINCT ?property ?value {
        { <http://dbpedia.org/resource/Madonna_(entertainer)> ?property ?value. }
        UNION
        {
                ?value ?prop <http://dbpedia.org/resource/Madonna_(entertainer)>.
                BIND(CONCAT("IS ", ?prop, " OF") AS ?property).
        }
}
```

You can also change the query above not to use the properties' URIs, but the labels of these properties.

EXERCISE 5

List the people and their names who are born in Brussels.

```
SELECT ?person ?name WHERE {
        ?person <http://dbpedia.org/ontology/birthPlace> <http://dbpedia.org/resource/Brussels> .
        ?person <http://dbpedia.org/property/name> ?name.
}
```

EXERCISE 6

Are there people who were born in Brussels and died in Paris? (use an ASK query)

```
ASK WHERE {
        ?person <http://dbpedia.org/ontology/birthPlace> <http://dbpedia.org/resource/Brussels> .
        ?person <http://dbpedia.org/property/deathPlace> <http://dbpedia.org/resource/Paris>.
}
```

EXERCISE 7

Find 20 people (URI and place of death) who were born in Ghent, but died elsewhere.

```
SELECT DISTINCT * WHERE {
        ?person <http://dbpedia.org/ontology/birthPlace> <http://dbpedia.org/resource/Ghent> .
        ?person <http://dbpedia.org/property/deathPlace> ?deathplace.
        FILTER(<http://dbpedia.org/resource/Ghent> != ?deathplace).
} LIMIT 20
```

You will see duplicates in the results. This is because sometimes both city of death as well as country of death are recorded as places of death. Another problem is that — when the country of death is recorded next to Ghent — that person is also in the result set.

EXERCISE 8

Give a list of people and their names. You will see duplicate people, which aggregate can you use to return just one of the names?

```
SELECT ?person SAMPLE(?name) WHERE {
        ?person a <http://xmlns.com/foaf/0.1/Person>.
        ?person <http://xmlns.com/foaf/0.1/name> ?name.
} GROUP BY ?person LIMIT 2
```

EXERCISE 9

Give a list of countries and their French ('FR') labels.

```
SELECT ?country ?label WHERE {
        ?country a <http://schema.org/Country>.
        ?country <http://www.w3.org/2000/01/rdf-schema#label> ?label.
        FILTER(langMatches(lang(?label), "fr"))
} LIMIT 20
```

EXERCISE 10

Choose two cities, and make everyone born in one city know all persons born in the other and vice versa. (use a CONSTRUCT query)

```
CONSTRUCT {
        ?person1 <http://dbpedia.org/property/knows> ?person2.
        ?person2 <http://dbpedia.org/property/knows> ?person1.
}
WHERE {
        ?person1 <http://dbpedia.org/ontology/birthPlace> <http://dbpedia.org/resource/Brussels> .
        ?person2 <http://dbpedia.org/ontology/birthPlace> <http://dbpedia.org/resource/Ghent> .
} LIMIT 10
```

RESOURCES

- http://dbpedia.org/resource/Madonna_(entertainer)
- http://dbpedia.org/ontology/birthPlace
- http://dbpedia.org/resource/Brussels
- http://dbpedia.org/property/name
- http://dbpedia.org/property/deathPlace
- http://dbpedia.org/resource/Paris
- http://dbpedia.org/resource/Ghent
- http://xmlns.com/foaf/0.1/Person
- http://xmlns.com/foaf/0.1/name
- http://dbpedia.org/property/knows