SPARQL

Nicolò Pratelli

Istituto di Scienza e Tecnologie dell'Informazione -
Consiglio Nazionale delle Ricerche – Pisa

UNIVERSITÀ DI PISA

A.A. 2024-2025

Istituto di Scienza e Tecnologie
dell'Informazione "A. Faedo"

In seminar 2

In the last seminar you learned how to:

- define **classes** and **properties** in RDFS
- manage the **special predicate** “a”
- use **ontologies** like DC and FOAF
- search **prefixes** on prefix.cc
- deal with a **knowledge base** like Wikidata
- (and we successfully installed **Blazegraph**)

SPARQL

- SPARQL is a recursive acronym for “SPARQL Protocol and RDF Query Language” and it is a language for **retrieving** and **manipulating** RDF data
- The latest version available is **SPARQL 1.1**
- RDF Datasets are made accessible via **SPARQL endpoint** provided by a SPARQL service
- SPARQL answers queries by matching **graph patterns** that are constructed using variables
- A variable is composed of a **question mark** followed by a variable name, such as `?something`
- The graph patterns are made of **triples**, and the syntax is very **similar to Turtle**

Prefixes in SPARQL

Prefixes in SPARQL have different syntax compared to Turtle:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
```

3 differences may cause mistakes:

- no @ before PREFIX
- uppercase PREFIX
- no dot at the end of the line

The latest version of Turtle supports this syntax, so you can use SPARQL-style prefixes in Turtle (but not the opposite)

SPARQL SELECT Query Structure

The SELECT query is composed of:

- some prefixes
- a `SELECT` clause
- (optional) dataset (`FROM`) clause
- a `WHERE` clause
- (optional) solution modifiers

that occur in this order.

SPARQL Query Example

```
# Extract the IRI and the label of all resources that have
a label
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT ?iri ?label
WHERE {
      ?iri rdfs:label ?label  .
}
```

iri	label
<http://www.wikidata.org/entity/Q4653>	Sherlock Holmes
<http://www.wikidata.org/entity/Q187349>	Dr. John Watson

Some SPARQL Operators

Today we will use the following SPARQL operators:

1. the `FILTER` operator is used to restrict solutions to those for which the filter expression evaluates to TRUE
2. the `DISTINCT` operator is used to remove duplicate solutions from the results
3. the `ORDER BY` operator is used to sort the results
4. the `LIMIT` operator is used to report only a subset of the results
5. the `OPTIONAL` operator is used when you want to capture a triple you don’t know if is present in the graph
6. the `UNION` operator is used to unite two graph patterns

Filtering the Results

```
# Extract IRI and label of texts, filtered by date
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dctypes: <http://purl.org/dc/dcmitype/>
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
SELECT ?iri ?label
WHERE {
    ?iri rdfs:label ?label .
    ?iri rdf:type dctypes:Text .
    ?iri dcterms:issued ?date .
    FILTER  (?date  <   "1889"^^xsd:date)
}
```

?iri	?label
<http://www.wikidata.org/entity/Q223131>	A Study in Scarlet

Filter is **inside** the WHERE clause

Removing Duplicates

```
# Extract the family name of all resources, and remove
duplicate results
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT DISTINCT ?familyName
WHERE {
    ?iri foaf:familyName ?familyName .
}
```

?familyName
Holmes
Watson
Moriarty

We have only one result for Holmes, while we have people with the same family name (i.e. Mycroft and Sherlock),

Ordering the Results

```
# Extract IRI and label of all resources that have a
label, and order the results by label
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT ?iri ?label
WHERE {
      ?iri rdfs:label ?label  .
} ORDER BY ?label
```

?iri	?label
<http://www.wikidata.org/entity/Q187349>	Dr. John Watson
<http://www.wikidata.org/entity/Q4653>	Sherlock Holmes

Limit

```
# Extract IRI and label of all resources that have a
label, and report one result
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT ?iri ?label
WHERE {
      ?iri rdfs:label ?label  .
} LIMIT 1
```

?iri	?label
<http://www.wikidata.org/entity/Q4653>	Sherlock Holmes
~~<http://www.wikidata.org/entity/Q187349>~~	~~Dr. John Watson~~

?iri	?label
<http://www.wikidata.org/entity/Q4653>	Sherlock Holmes

Retrieving Optional Patterns

```
# Extract label of all resources, and optionally date of birth
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dcterms: <http://purl.org/dc/terms/>
SELECT ?label ?date
WHERE {
    ?iri rdfs:label ?label  .
    OPTIONAL
    { ?iri dcterms:date ?date  . }
}
```

OPTIONAL is **inside** the WHERE clause

label	date
Sherlock Holmes	
Dr. John Watson	
Arthur Conan Doyle	1859-05-22

Uniting Graph Patterns

```
# Extract given name or family name of all people
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name
WHERE {
    ?iri rdf:type foaf:Person .
    {?iri foaf:givenName ?name . }
    UNION
    {?iri foaf:familyName ?name . }
}
```

UNION is **inside** the WHERE clause

name
Sherlock
John
Watson
Holmes

Instructions for Exercises

- Download the knowledge base from Moodle, open it in a text editor and look at its contents
- Launch the Blazegraph JAR file
- Go to the Blazegraph web interface and define a new namespace called “Seminar3” (no spaces!)
- Upload the knowledge base into **Blazegraph**
- Execute queries on the knowledge base
- Request a **review** when finished

Exercises

Write a query to display in Blazegraph:

1. IRI and label of all people
2. given name of all people with family name “Alighieri”
3. IRI and label of all Dante’s texts in alphabetical order
4. label and date of the first two Dante’s texts in chronological order
5. family names of all people without duplicates
6. label and date of all Dante’s texts created after 1310
7. given names of all people, and optionally the family name of their teacher
8. IRIs and labels of the texts in Italian or French language
9. labels of all texts created by Dante Alighieri or Brunetto Latini with date < 1300