COMP318

Ontologies and Semantic Web

SPARQL - Part 2

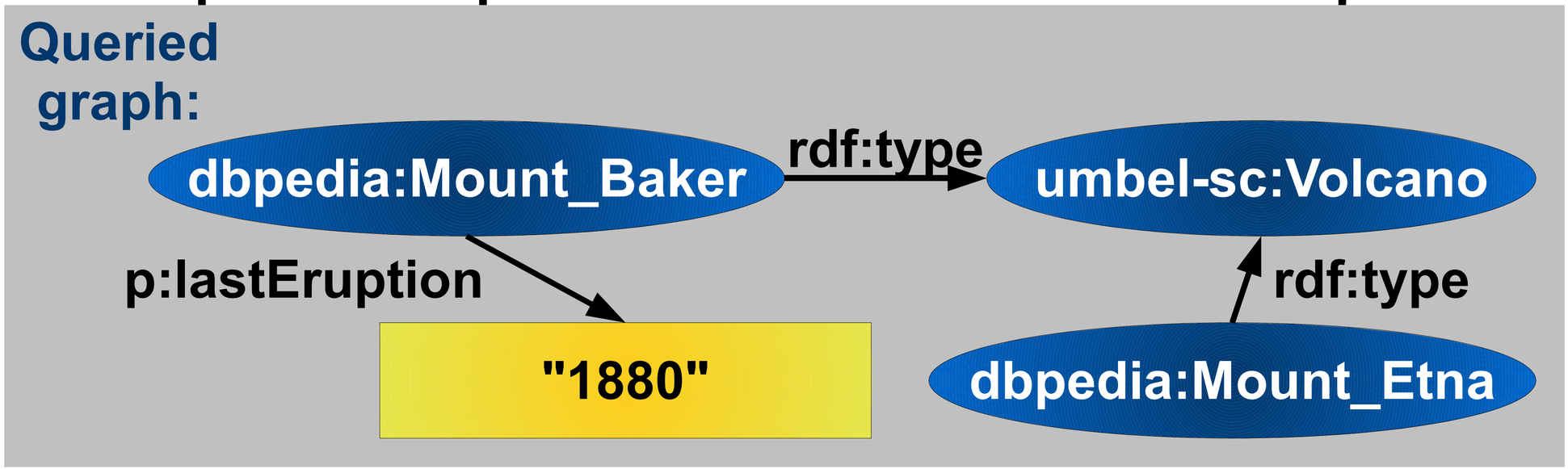
Dr Valentina Tamma

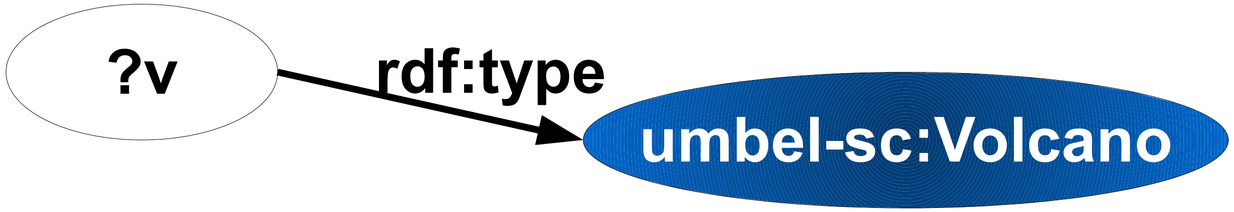
V.Tamma@liverpool.ac.uk

Where were we

- RDF data model & RDFS schema language
  - Vocabulary and model
- Principles of SPARQL

Main principle of SPARQL queries



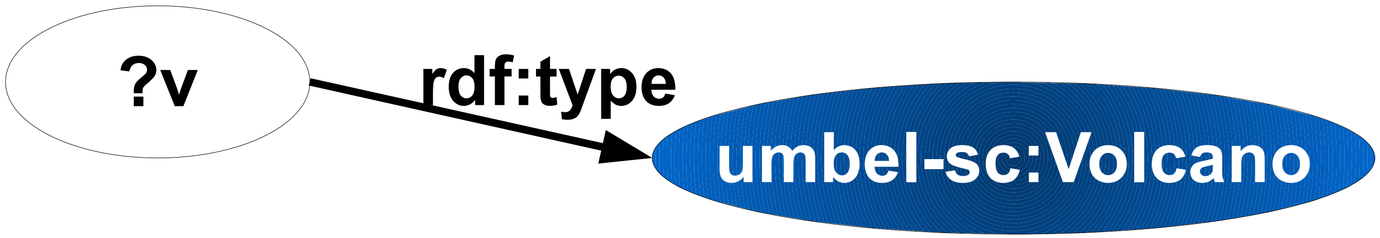


Results:

?v
dbpedia:Mount_Baker
dbpedia:Mount_Etna

Main principle of SPARQL queries

- SPARQL based on the principle of pattern matching
  - Describe subgraphs of the queried RDF graph
  - Subgraphs that match your description yield a result
    - i.e. graph patterns (i.e. RDF graphs with variables)



Query Result Forms

- `SELECT`: Projection of query results
- `CONSTRUCT`: Returning RDF Graph
- `DESCRIBE`: Returning descriptions of RDF resources
  - not treated here
- `ASK`: “yes/no” query

Selecting variables: SELECT

- Filtering variables to return
- Variables: **`?string`**

  `?x ?title ?name`
  - variables can match any node (resource or literal) in the RDF document
- Variables in **`SELECT`** are distinguished variables
- **`DISTINCT`** for non duplicate results
- Syntax:

```
SELECT var1, ... ,varn

SELECT ?x,?title

SELECT *
```

Query Patterns: FROM

- Specifies the dataset(s) to be queried
- Can be the default dataset
  - Then **`FROM`** can be omitted
  - **`FROM`** and **`NAMED FROM`** each with a URI to specify one or more dataset

Query Patterns: WHERE

- Graph pattern to match
  - Triple patterns are just like triples, but any part of a triple pattern can be replaced with a variable
- Set of triples:
  - { (**s p o** .)* }
    - Subject: URI, QName, Blank node, Variable
    - Predicate: URI, QName, Variable
    - Object: URI, QName, Blank node Literal, Variable

- Example:

```
{
    _:author ex:hasName ?name .
    _:author ex:authors :lotr .
}
```

- Optional triples: `OPTIONAL` triple

```
OPTIONAL :john ont:hasAge ?age
```

GRAPH PATTERNS

- Different types of graph patterns for the query pattern (WHERE clause):
  - Basic graph pattern (BGP)
  - `Group` graph pattern
  - `Optional` graph pattern
  - `Union` graph pattern
  - `Graph` graph pattern (Constraints)

Example RDF Dataset (Turtle)

```
@prefix : <http://example.org/data#> .
@prefix ont: <http://example.org/myOntology#> .
@prefix vcard: <http://www.w3.org/2001/vcard-rdf/3.0#> .

:john
      vcard:FN "John Smith" ;
      vcard:N [
            vcard:Given "John" ;
            vcard:Family "Smith" ] ;
      ont:hasAge 32 ;
      ont:marriedTo :mary .
:mary
      vcard:FN "Mary Smith" ;
      vcard:N [
            vcard:Given "Mary" ;
            vcard:Family "Smith" ] ;
      ont:hasAge 29 .
```

BASIC GRAPH PATTERNS

- Set of triple patterns (i.e. RDF triples with variables)
- Variable names prefixed with “?” or “$” (e.g. `?v`, `$v`)
- Turtle syntax (similar to N3)
  - Syntactic sugar as in N3 (e.g. property and object lists)
- Blank nodes in SPARQL queries
  - Permitted as *subject* and *object* of a triple pattern
  - Like non-selectable variables

SPARQL Queries: all full names

"Return the full names of all people in the graph"

```
PREFIX vCard:
     <http://www.w3.org/2001/vcard-rdf/3.0#>
SELECT ?fullName
WHERE {?x vCard:FN ?fullName}
```

result:

```
fullName
=================
"John Smith"
"Mary Smith"
```

```
:john
     vcard:FN "John Smith" ;
     vcard:N [
          vcard:Given "John" ;
          vcard:Family "Smith" ] ;
     ont:hasAge 32 ;
     ont:marriedTo :mary .
:mary
     vcard:FN "Mary Smith" ;
     vcard:N [
          vcard:Given "Mary" ;
          vcard:Family "Smith" ] ;
     ont:hasAge 29 .
```

SPARQL Queries: properties

"Return the relation between John and Mary"

```
PREFIX : <http://example.org/myOntology#>
SELECT ?p
WHERE { :john ?p :mary }
```

result:

```
p
=================
<http://example.org/myOntology#marriedTo>
```

```
:john
      vcard:FN "John Smith" ;
      vcard:N [
            vcard:Given "John" ;
            vcard:Family "Smith" ] ;
      ont:hasAge 32 ;
      ont:marriedTo :mary .
:mary
      vcard:FN "Mary Smith" ;
      vcard:N [
            vcard:Given "Mary" ;
            vcard:Family "Smith" ] ;
      ont:hasAge 29 .
```

SPARQL Queries: complex patterns

"Return the spouse of a person by the name of John Smith"

```
PREFIX vCard: <http://www.w3.org/2001/vcard-rdf/
   3.0#>
PREFIX ont: <http://example.org/myOntology#>
SELECT ?y
WHERE {?x vCard:FN "John Smith".
        ?x ont:marriedTo ?y}
```

result:

```
y
==================
<http://example.org/data#mary>
```

```
:john
      vcard:FN "John Smith" ;
      vcard:N [
            vcard:Given "John" ;
            vcard:Family "Smith" ] ;
      ont:hasAge 32 ;
      ont:marriedTo :mary .
:mary
      vcard:FN "Mary Smith" ;
      vcard:N [
            vcard:Given "Mary" ;
            vcard:Family "Smith" ] ;
      ont:hasAge 29 .
```

SPARQL Queries: blank nodes

"Return the name and the first name of all people in the KB"

```
PREFIX vCard: <http://www.w3.org/2001/vcard-rdf/3.0#>
SELECT ?name, ?firstName
WHERE {?x vCard:N ?name .
       ?name vCard:Given ?firstName}
```

result:

```
        name          firstName
=============================
_:a "John Smith"    "John"
_:b "Mary Smith"    "Mary"
```

```
:john
      vcard:FN "John Smith" ;
      vcard:N [
            vcard:Given "John" ;
            vcard:Family "Smith" ] ;
      ont:hasAge 32 ;
      ont:marriedTo :mary .
:mary
      vcard:FN "Mary Smith" ;
      vcard:N [
            vcard:Given "Mary" ;
            vcard:Family "Smith" ] ;
      ont:hasAge 29 .
```

COMP318

Ontologies and Semantic Web

End of SPARQL - Part 2

Dr Valentina Tamma

`V.Tamma@liverpool.ac.uk`