

# Introduction to Knowledge Graphs and Semantic Web Technologies

## SPARQL

Olaf Hartig

[olaf.hartig@liu.se](mailto:olaf.hartig@liu.se)

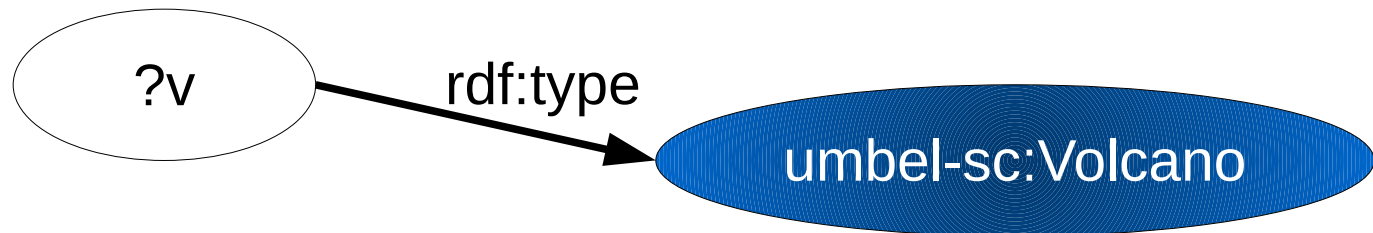
# SPARQL

- Family of W3C standards related to querying RDF data
- SPARQL **Query Language**
  - Declarative query language for RDF data
  - Our focus now
- SPARQL **Update**
  - Declarative update language for RDF data
- SPARQL **Protocol**
  - Communication between SPARQL processing services (a.k.a. SPARQL endpoints) and clients
- and more
  - e.g., formats for serializing query results



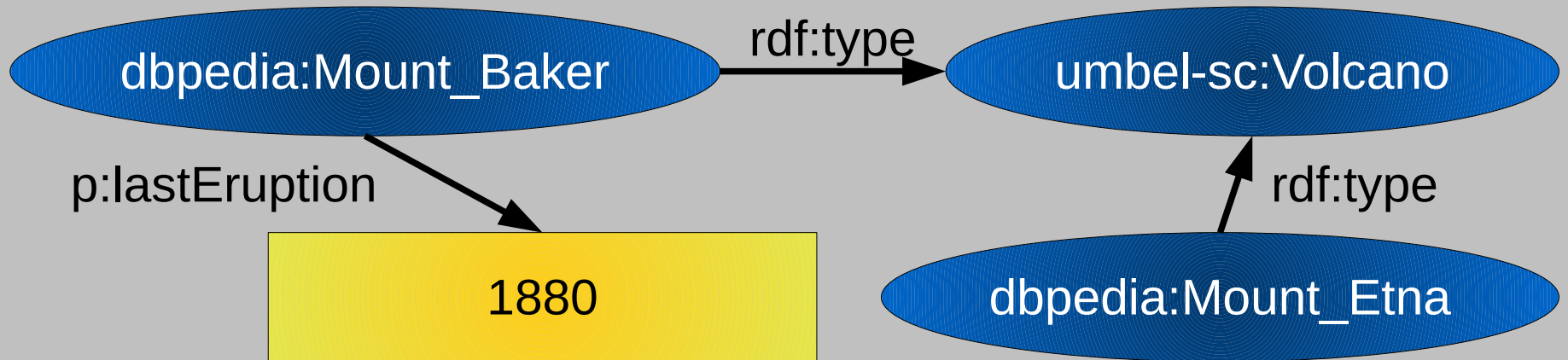
# Main Idea of SPARQL Queries

- Pattern matching:
  - Describe subgraphs of the queried RDF graph
  - Subgraphs that match the description yield a result
  - Mean: **graph patterns** (essentially, RDF graphs with variables)

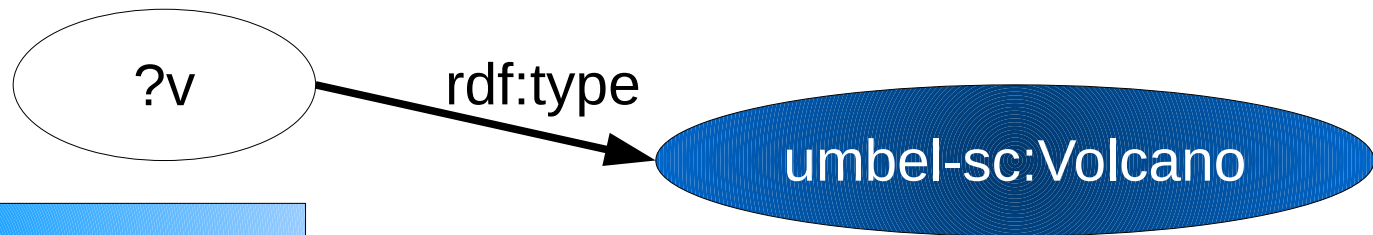


# Main Idea of SPARQL Queries

Queried RDF graph:

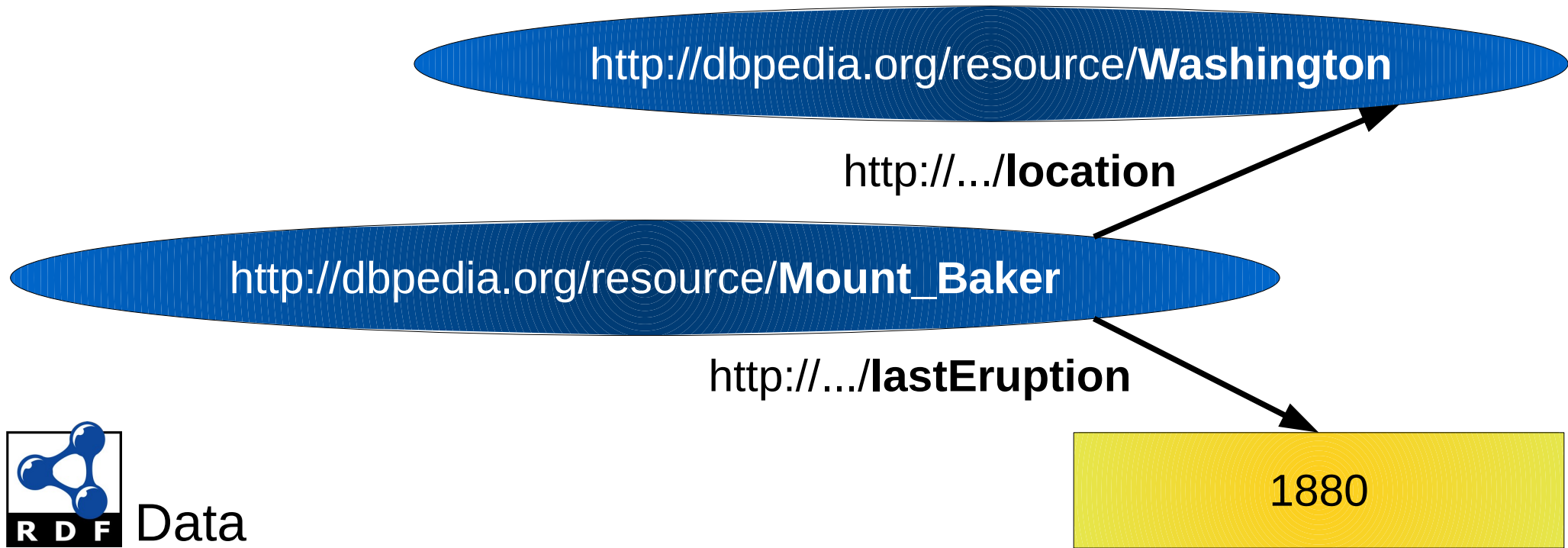


Result:



<code>?v</code>
<code>dbpedia:Mount_Baker</code>
<code>dbpedia:Mount_Etna</code>

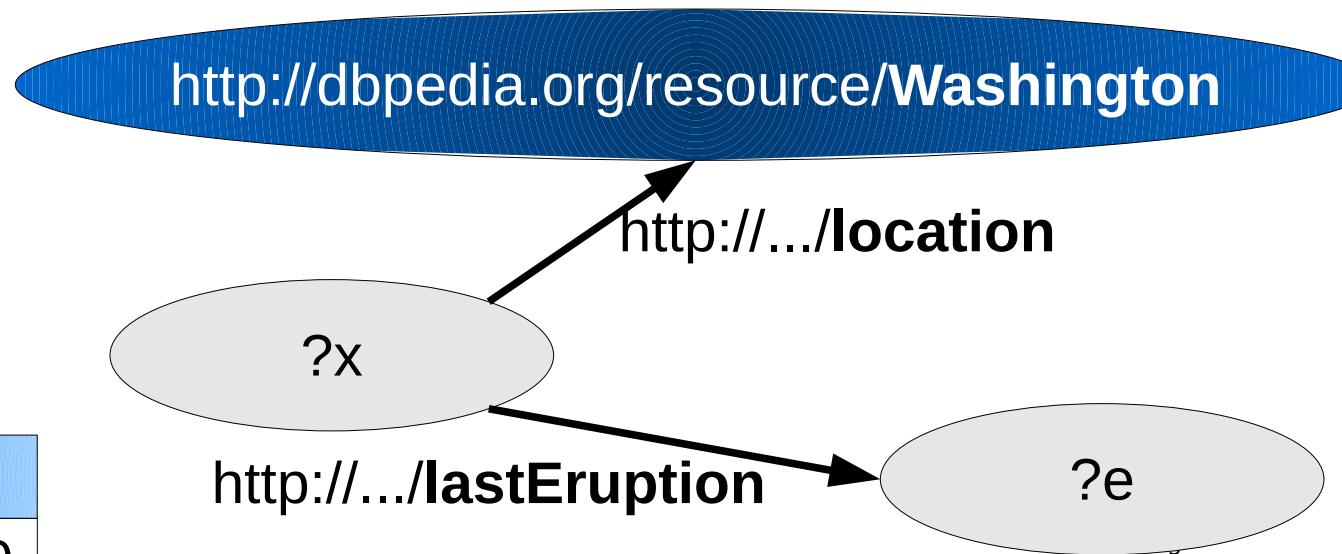
# Another Example



Data



Query



Query Result:

<code>?x</code>	<code>?e</code>
<code>http://.../Mount_Baker</code>	<code>1880</code>

# Components of SPARQL Queries

# Components of a SPARQL Query

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX umbel-sc: <http://umbel.org/umbel/sc/>
SELECT ?v
FROM <http://example.org/myGeoData>
WHERE {
    ?v rdf:type umbel-sc:Volcano .
}
ORDER BY ?v
```

# Components of a SPARQL Query

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX umbel-sc: <http://umbel.org/umbel/sc/>
SELECT ?v
FROM <http://example.org/myGeoData>
WHERE {
    ?v rdf:type umbel-sc:Volcano .
}
ORDER BY ?v
```

- Prologue:
  - **Prefix definitions** for using compact URIs (CURIEs)



# Components of a SPARQL Query

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX umbel-sc: <http://umbel.org/umbel/sc/>
SELECT ?v
FROM <http://example.org/myGeoData>
WHERE {
    ?v rdf:type umbel-sc:Volcano .
}
ORDER BY ?v
```

- Result form specification:
  - SELECT for tabular results (similar to SELECT in SQL)
  - Other forms: DESCRIBE, CONSTRUCT, and ASK

# Components of a SPARQL Query

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX umbel-sc: <http://umbel.org/umbel/sc/>
SELECT ?v
FROM <http://example.org/myGeoData>
WHERE {
    ?v rdf:type umbel-sc:Volcano .
}
ORDER BY ?v
```

- Dataset specification:
  - Specify the RDF dataset to be queried (use URIs that identify particular RDF graphs in your RDF database)

# Components of a SPARQL Query

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX umbel-sc: <http://umbel.org/umbel/sc/>
SELECT ?v
FROM <http://example.org/myGeoData>
WHERE {
    ?v rdf:type umbel-sc:Volcano .
}
ORDER BY ?v
```

- Query Pattern:
  - WHERE clause specifies the graph pattern to be matched

# Components of a SPARQL Query

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX umbel-sc: <http://umbel.org/umbel/sc/>
SELECT ?v
FROM <http://example.org/myGeoData>
WHERE {
    ?v rdf:type umbel-sc:Volcano .
}
ORDER BY ?v
```

- Solution modifiers (for SELECT queries):
  - Modify the **result set** as a whole (not single solutions)
  - Keywords: DISTINCT, ORDER BY, LIMIT, and OFFSET

# Simple Types of Graph Patterns

# Basic Graph Pattern

- Set of triple patterns (i.e., RDF triples with variables)
- Variable names prefixed with “?”

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX umbel-sc: <http://umbel.org/umbel/sc/>
SELECT ?name
WHERE {
    ?v rdf:type umbel-sc:Volcano .
    ?v rdfs:label ?name .
}
```

# Basic Graph Pattern

- Set of triple patterns (i.e., RDF triples with variables)
- Variable names prefixed with “?”
- Turtle syntax
  - Including syntactic sugar (e.g., property and object lists)

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX umbel-sc: <http://umbel.org/umbel/sc/>
SELECT ?name
WHERE {
    ?v rdf:type umbel-sc:Volcano ;
        rdfs:label ?name .
}
```

# Basic Graph Pattern (Example)

```
dbpedia:Mount_Etna rdf:type umbel-sc:Volcano ;                               Data*
    rdfs:label "Etna" .
dbpedia:Mount_Baker rdf:type umbel-sc:Volcano.
dbpedia:Beerenberg rdf:type umbel-sc:Volcano,
    umbel-sc:NaturalElevation ;
    rdfs:label "Beerenberg"@en ;
    rdfs:label "Бере́нберг"@ru .
```

- **Question:** What are the names of all (known) volcanos?

```
SELECT ?v ?name WHERE {                                     Query*
    ?v rdf:type umbel-sc:Volcano ;
    rdfs:label ?name . }
```

\*Prefix declarations omitted

Result:

?v	?name
dbpedia:Mount_Etna	"Etna"
dbpedia:Beerenberg	"Beerenberg"@en
dbpedia:Beerenberg	"Бере́нберг"@ru



# Basic Graph Pattern (Example)

```
dbpedia:Mount_Etna rdf:type umbel-sc:Volcano ;                               Data*
                        rdfs:label "Etna" .
dbpedia:Mount_Baker rdf:type umbel-sc:Volcano.
dbpedia:Beerenberg  rdf:type umbel-sc:Volcano,
                        umbel-sc:NaturalElevation ;
                        rdfs:label "Beerenberg"@en ;
                        rdfs:label "Бере́нберг"@ru .
```

- **Question:** What are *all* (known) volcanos and their names?

```
SELECT ?v ?name WHERE {
  ?v rdf:type umbel-sc:Volcano ;
  rdfs:label ?name . } Query*
```

Mount Baker is **missing!**

Result:

?v	?name
dbpedia:Mount_Etna	"Etna"
dbpedia:Beerenberg	"Beerenberg"@en
dbpedia:Beerenberg	"Бере́нберг"@ru

# Optional Graph Pattern

```
dbpedia:Mount_Etna rdf:type umbel-sc:Volcano ;                               Data*
    rdfs:label "Etna" .
dbpedia:Mount_Baker rdf:type umbel-sc:Volcano.
dbpedia:Beerenberg rdf:type umbel-sc:Volcano,
    umbel-sc:NaturalElevation ;
    rdfs:label "Beerenberg"@en ;
    rdfs:label "Бере́нберг"@ru .
```

```
SELECT ?v ?name WHERE {                                                    Query*
    ?v rdf:type umbel-sc:Volcano .
    OPTIONAL {
        ?v rdfs:label ?name .
    }
}
```

Using OPTIONAL may result in unbound variables in some solutions of the query result

?v	?name
dbpedia:Mount_Etna	"Etna"
dbpedia:Beerenberg	"Beerenberg"@en
dbpedia:Beerenberg	"Бере́нберг"@ru
dbpedia:Mount_Baker	

# UNION Clauses

```
dbpedia:Mount_Etna rdf:type umbel-sc:Volcano ;                               Data
    rdfs:label "Etna" ;
    p:location dbpedia:Italy .
dbpedia:Mount_Baker rdf:type umbel-sc:Volcano ;
    p:location dbpedia:United_States .
dbpedia:Beerenberg rdf:type umbel-sc:Volcano ;
    rdfs:label "Beerenberg"@en ;
    p:location dbpedia:Norway .
```

Question: Which volcanos are located in the Italy or in Norway?

```
SELECT ?v WHERE {                                                            Query
    ?v rdf:type umbel-sc:Volcano ;
    p:location      ?      . }
```

# UNION Clauses

```
SELECT ?v WHERE {                                     Query
  { ?v rdf:type umbel-sc:Volcano ;
    p:location dbpedia:Italy }
  UNION
  { ?v rdf:type umbel-sc:Volcano ;
    p:location dbpedia:Norway }
}
```

**Semantically  
equivalent**



```
SELECT ?v WHERE {
  ?v rdf:type umbel-sc:Volcano .
  { ?v p:location dbpedia:Italy }
  UNION
  { ?v p:location dbpedia:Norway }
}
```

Query

# Constraints on Solutions

- Syntax: Keyword FILTER followed by filter expression

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX umbel-sc: <http://umbel.org/umbel/sc/>
PREFIX p: <http://dbpedia.org/property/>

SELECT ?v
WHERE {
    ?v rdf:type umbel-sc:Volcano ;
        p:lastEruption ?le .
    FILTER ( ?le > 1900 )
}
```

# Unary Operators in Constraints

<b>Operator</b>	<b>Type(A)</b>	<b>Result type</b>
! A	xsd:boolean	xsd:boolean
+ A	numeric	numeric
- A	numeric	numeric
BOUND(A)	variable	xsd:boolean
isURI(A)	RDF term	xsd:boolean
isBLANK(A)	RDF term	xsd:boolean
isLITERAL(A)	RDF term	xsd:boolean
STR(A)	literal / URI	simple literal
LANG(A)	literal	simple literal
DATATYPE(A)	literal	simple literal

# Binary and Other Operators

- Logical connectives `&&` and `||`
  - for `xsd:boolean`
- Comparison operators `=`, `!=`, `<`, `>`, `<=`, and `>=`
  - for numeric datatypes, `xsd:dateTime`, `xsd:string`, `xsd:boolean`
- Comparison operators `=` and `!=`
  - for other datatypes
- Arithmetic operators `+`, `-`, `*`, and `/` (for numeric datatypes)
- Furthermore:
  - `REGEX(String,Pattern)` or `REGEX(String,Pattern,Flags)`
  - `langMATCHES(A,B)`
  - etc.

# Constraints (Example)

```
dbpedia:Mount_Etna rdf:type umbel-sc:Volcano ;                               Data
                    rdfs:label "Etna" .
dbpedia:Beerenberg rdf:type umbel-sc:Volcano,
                    umbel-sc:NaturalElevation ;
                    rdfs:label "Beerenberg"@en ;
                    rdfs:label "Береберг"@ru .
```

- **Question:** What volcanos have an “e” in their name?

```
SELECT ?v WHERE {                                                            Query
  ?v rdf:type umbel-sc:Volcano ;
     rdfs:label ?name .
  FILTER( REGEX(STR(?name), "e") )
}
```

?v
dbpedia:Beerenberg
dbpedia:Beerenberg



# Constraints (Example, cont'd)

```
dbpedia:Mount_Etna rdf:type umbel-sc:Volcano ;                               Data
                    rdfs:label "Etna" .
dbpedia:Beerenberg rdf:type umbel-sc:Volcano,
                    umbel-sc:NaturalElevation ;
                    rdfs:label "Beerenberg"@en ;
                    rdfs:label "Береенберг"@ru .
```

- **Question:** What volcanos have an “e” in their name?

```
SELECT ?v WHERE {                                                           Query
  ?v rdf:type umbel-sc:Volcano ;
     rdfs:label ?name .
  FILTER( REGEX(STR(?name), "e", "i") )
}
```

?v
dbpedia:Mount_Etna
dbpedia:Beerenberg
dbpedia:Beerenberg

# Property Path Patterns

- Navigation in an RDF graph (à la Xpath)

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

```
SELECT ?n
```

```
WHERE {
```

```
  ?p foaf:name "Olaf Hartig".
```

```
  ?p foaf:knows+/foaf:name ?n .
```

```
}
```

# Assigning Values to Variables

- In SELECT clause

```
PREFIX ex: <http://example.org/>
SELECT ?p (2023 - ?y AS ?age)
WHERE {
  ?p a ex:Person .
  ?p ex:yearOfBirth ?y .
}
```

- In BIND clause inside WHERE

```
PREFIX ex: <http://example.org/>
SELECT ?p ?age
WHERE {
  ?p a ex:Person .
  ?p ex:yearOfBirth ?y .
  BIND( 2023 - ?y AS ?age )
  FILTER( ?age > 30 )
}
```

# Querying RDF Datasets

that contain Named Graphs

# Idea

- SPARQL queries are evaluated over an RDF dataset
- An **RDF dataset** consists of:
  - one **default graph** and
  - zero or more **named graphs** (identified by an URI)
- Keyword **GRAPH** makes one of the named graphs the **active graph** used for pattern matching

# Examples of GRAPH Clauses

```
dbpedia:Mount_Etna rdfs:seeAlso <http://example.org/d1> .  
dbpedia:Mount_Baker rdfs:seeAlso <http://example.org/d2> .
```

Default  
Graph



```
dbpedia:Mount_Etna    http://example.org/d1  
    rdf:type umbel-sc:Volcano ;  
    rdfs:label "Etna" .
```

```
dbpedia:Mount_Baker  http://example.org/d2  
    rdf:type umbel-sc:Volcano .
```

```
dbpedia:Beerenberg   http://example.org/d3  
    rdf:type umbel-sc:Volcano ;  
    rdfs:label "Beerenberg"@en .
```

# Examples of GRAPH Clauses

```
dbpedia:Mount_Etna rdfs:seeAlso <http://example.org/d1> .  
dbpedia:Mount_Baker rdfs:seeAlso <http://example.org/d2> .
```

Default  
Graph



```
dbpedia:Mount_Etna http://example.org/d1  
rdf:type umbel-sc:Volcano ;  
rdfs:label "Etna" .
```

```
dbpedia:Mount_Baker http://example.org/d2  
rdf:type umbel-sc:Volcano .
```

```
SELECT ?v WHERE {  
  GRAPH <http://example.org/d1> {  
    ?v rdf:type umbel-sc:Volcano .  
  }  
}
```

```
http://example.org/d3
```

```
umbel-sc:Volcano ;  
rdfs:label "Beerenberg"@en .
```

**?v**

dbpedia:Mount\_Etna

# Examples of GRAPH Clauses

```
dbpedia:Mount_Etna rdfs:seeAlso <http://example.org/d1> .
dbpedia:Mount_Baker rdfs:seeAlso <http://example.org/d2> .
```

Default  
Graph



```
dbpedia:Mount_Etna http://example.org/d1
rdf:type umbel-sc:Volcano ;
rdfs:label "Etna" .
```

```
dbpedia:Mount_Baker http://example.org/d2
rdf:type umbel-sc:Volcano .
```

```
SELECT ?v WHERE {
  GRAPH ?g {
    ?v rdf:type umbel-sc:Volcano .
  }
}
```

```
http://example.org/d3
umbel-sc:Volcano ;
rdfs:label "Beer-  
bergenberg" .
```

```
dbpedia:Mount_Etna
dbpedia:Mount_Baker
dbpedia:Beerenberg
```



# Examples of GRAPH Clauses

```
dbpedia:Mount_Etna rdfs:seeAlso <http://example.org/d1> .  
dbpedia:Mount_Baker rdfs:seeAlso <http://example.org/d2> .
```

Default  
Graph



```
dbpedia:Mount_Etna http://example.org/d1  
rdf:type umbel-sc:Volcano ;  
rdfs:label "Etna" .
```

```
dbpedia:Mount_Baker http://example.org/d2  
rdf:type umbel-sc:Volcano .
```

```
SELECT ?g ?v WHERE {
```

```
  GRAPH ?g {  
    ?v rdf:type
```

```
  }
```

```
}
```

**?g**

**?v**

<http://example.org/d1>

<http://example.org/d2>

<http://example.org/d3>

dbpedia:Mount\_Etna

dbpedia:Mount\_Baker

dbpedia:Beerenberg

# Examples of GRAPH Clauses

```
dbpedia:Mount_Etna rdfs:seeAlso <http://example.org/d1> .  
dbpedia:Mount_Baker rdfs:seeAlso <http://example.org/d2> .
```

Default  
Graph



```
dbpedia:Mount_Etna http://example.org/d1  
rdf:type umbel-sc:Volcano ;  
rdfs:label "Etna" .
```

```
dbpedia:Mount_Baker http://example.org/d2  
rdf:type umbel-sc:Volcano .
```

```
SELECT ?v WHERE {  
  _:x rdfs:seeAlso ?g  
  GRAPH ?g {  
    ?v rdf:type umbel-sc:Volcano .  
  }  
}
```

**?v**

```
dbpedia:Mount_Etna  
dbpedia:Mount_Baker
```

# Solution Modifiers

# DISTINCT

Removes duplicates from the result set

```
dbpedia:Mount_Etna rdf:type umbel-sc:Volcano ;  
                    rdfs:label "Etna" .  
dbpedia:Mount_Baker rdf:type umbel-sc:Volcano.  
dbpedia:Beerenberg rdf:type umbel-sc:Volcano,  
                    umbel-sc:NaturalElevation ;  
                    rdfs:label "Beerenberg"@en ;  
                    rdfs:label "Бере́нберг"@ru .
```

Data

```
SELECT ?type  
WHERE { _:x rdf:type ?type }
```

Query

**?type**

```
umbel-sc:Volcano  
umbel-sc:Volcano  
umbel-sc:NaturalElevation  
umbel-sc:Volcano
```

# DISTINCT

Removes duplicates from the result set

```
dbpedia:Mount_Etna rdf:type umbel-sc:Volcano ;  
                    rdfs:label "Etna" .  
dbpedia:Mount_Baker rdf:type umbel-sc:Volcano.  
dbpedia:Beerenberg rdf:type umbel-sc:Volcano,  
                    umbel-sc:NaturalElevation ;  
                    rdfs:label "Beerenberg"@en ;  
                    rdfs:label "Бере́нберг"@ru .
```

Data

```
SELECT DISTINCT ?type  
WHERE { _:x rdf:type ?type }
```

Query

**?type**

umbel-sc:Volcano  
umbel-sc:NaturalElevation

# ORDER BY

Specify an order for the returned solution mappings

```
SELECT ?v WHERE { ?v rdf:type umbel-sc:Volcano ;  
                  rdfs:label ?name }  
ORDER BY ?name
```

Query

Order for different kinds of elements:

unbound variable < blank node < URI < literal

**ASC** for ascending (default) and **DESC** for descending

Hierarchical order criteria:

```
SELECT ?name WHERE { ?v rdf:type umbel-sc:Volcano ;  
                       p:lastEruption ?le ;  
                       rdfs:label ?name }  
ORDER BY DESC(?le), ?name
```

Query

# LIMIT and OFFSET

**LIMIT** – limits the number of solution mappings returned

```
SELECT ?name WHERE { ?v rdf:type umbel-sc:Volcano ; Query
                        rdfs:label ?name }

ORDER BY ?name

LIMIT 5
```

**OFFSET** – position/index of the first returned mapping  
(useful only in combination with ORDER BY)

```
SELECT ?name WHERE { ?v rdf:type umbel-sc:Volcano ; Query
                        rdfs:label ?name }

ORDER BY ?name

LIMIT 5 OFFSET 10
```

# Result Forms



# Components of a SPARQL Query

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX umbel-sc: <http://umbel.org/umbel/sc/>
SELECT ?v
FROM <http://example.org/myGeoData>
WHERE {
    ?v rdf:type umbel-sc:Volcano .
}
ORDER BY ?v
```

- Result form specification:
  - SELECT for projection (similar to SELECT in SQL)
  - Other forms: DESCRIBE, CONSTRUCT, and ASK  
~~(more about these later)~~ **now ...**

# SELECT and ASK

## SELECT

Sequence of solutions mappings

Selected variables separated by space (not by comma!)

Asterisk (“\*”) character selects all variables in the pattern

## ASK

Check if there is **at least one solution mapping**

Example: Do we have any volcanos?

```
ASK WHERE { ?v rdf:type umbel-sc:Volcano .  
            OPTIONAL { ?v rdfs:label ?name }  
            FILTER( ! BOUND(?name) )  
}
```

Query

# DESCRIBE

Returns an RDF graph with data about resources

Not deterministic (i.e. query processor determines the actual structure of the returned RDF graph)

Name the resource:

```
DESCRIBE <http://dbpedia.org/resource/Beerenberg> Query
```

Specify the resource with a query pattern:

```
DESCRIBE ?v WHERE {  
  ?v rdf:type umbel-sc:Volcano ; rdfs:label ?name .  
  FILTER ( STR(?name) = "Beerenberg" )  
}
```

 Query

Multiple variables possible or asterisk (“\*”) for all

# CONSTRUCT

Returns an RDF graph created from a template

Template: graph pattern with variables from the query pattern

```
CONSTRUCT { ?v rdfs:label ?name ;  
            rdf:type myTypes:VolcanoInTheUS }  
  
WHERE {  
  ?v rdf:type umbel-sc:Volcano ;  
      rdfs:label ?name ;  
      p:location ?l .  
  FILTER ( ?l = dbpedia:United_States )  
}
```

Query

# SPARQL Version 1.1

# Other Features of SPARQL Queries

- Sub-queries
- Negation (EXISTS, NOT EXISTS, MINUS)
- Assignments (e.g., BIND, VALUES, SELECT expressions)
- Aggregate functions (e.g., COUNT, SUM, AVG)
- Basic query federation (SERVICE)
- SPARQL Update – graph update (INSERT DATA, DELETE DATA, INSERT, DELETE, DELETE WHERE, LOAD, CLEAR)
- SPARQL Update – graph management (CREATE, DROP, COPY, MOVE, ADD)

[www.liu.se](http://www.liu.se)