



**WEB SÉMANTIQUE ET ONTOLOGIES**  
**WEB DES DONNÉES**  
**DONNÉES LIÉES (LINKED DATA)**

## **4 – INTERROGER LES DONNÉES AVEC SPARQL**

Philippe GENOUD – Danielle ZIEBELIN - LIG-Steamer

[Prénom.Nom@imag.fr](mailto:Prénom.Nom@imag.fr)



# Outline

- “Theoretical” Session (morning)
  - Introduction
  - Distributing Data on the web with RDF
    - Naming the Data : URIs (Uniform Resources Identifiers)
    - The RDF Data model
  - Querying Linked Data with SPARQL
  - Semantic modelling
    - RDFS
    - OWL
  - From Open Data to Linked Open Data
  - Conclusion
- Hands-on session (afternoon)
  - From a CSV file to linked data
  - Querying linked data (SPARQL)

# Querying Linked Data with SPARQL

## Linked Data: 3<sup>rd</sup> Principle

*When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL).*



Most apps use only a subset of the stack

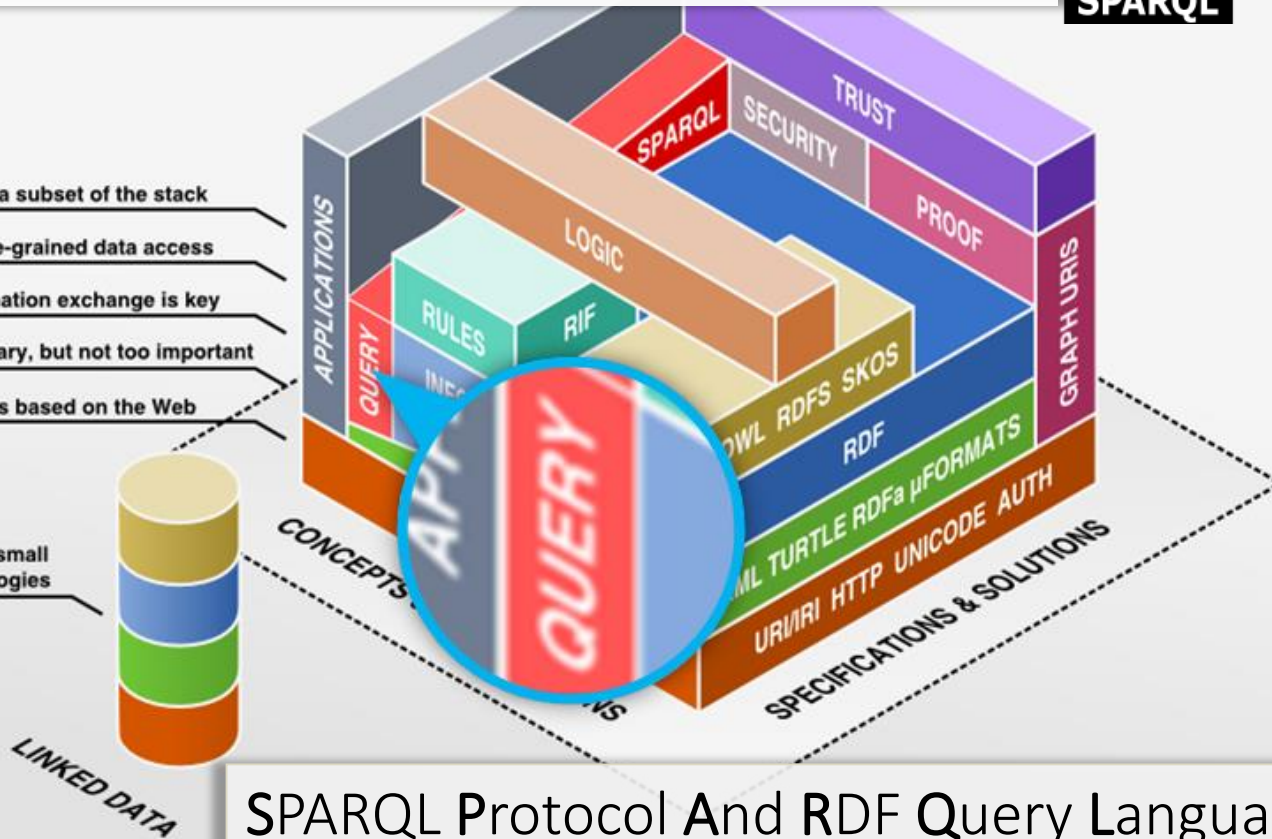
Querying allows fine-grained data access

Standardized information exchange is key

Formats are necessary, but not too important

The Semantic Web is based on the Web

Linked Data uses a small selection of technologies

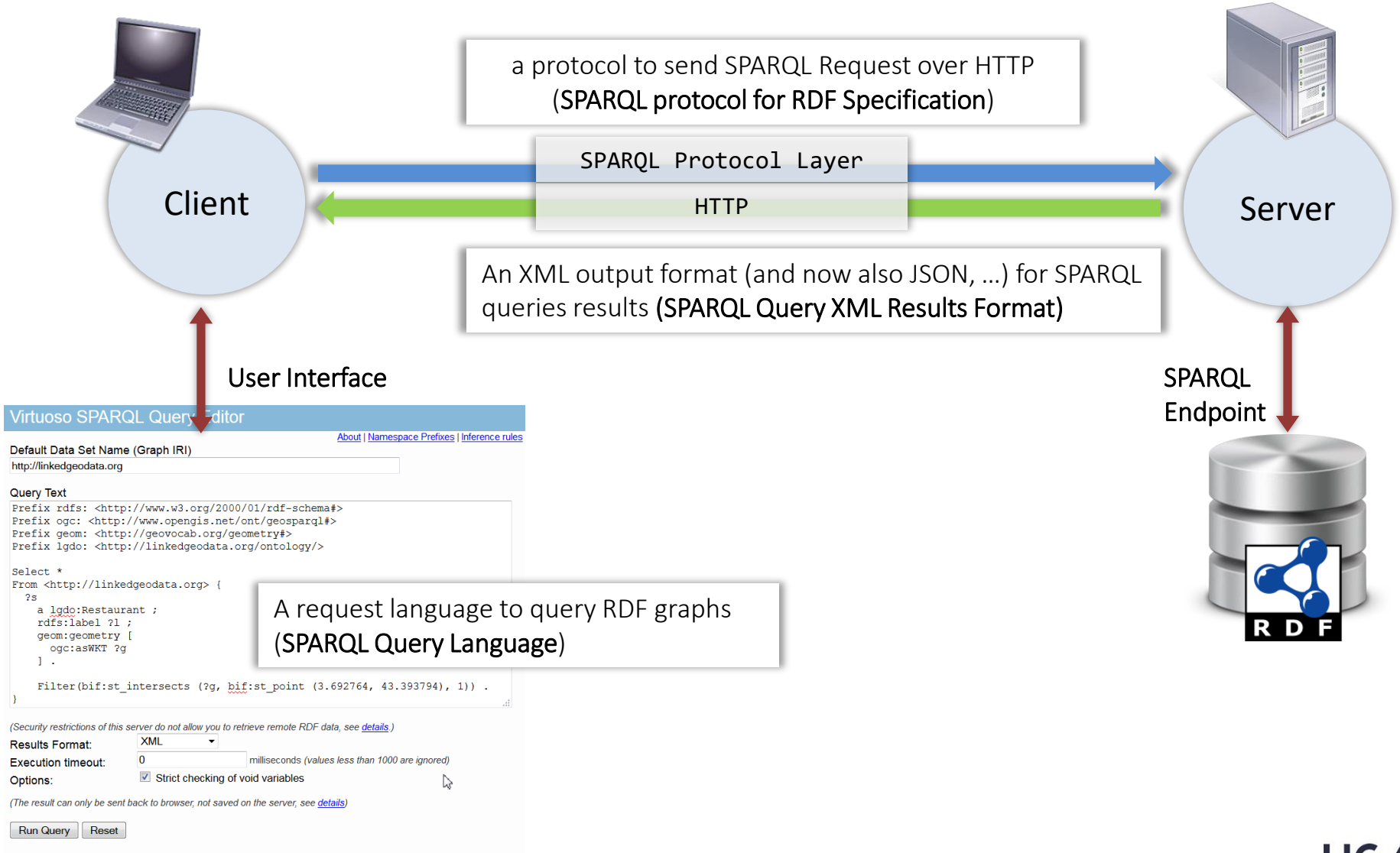


SPARQL Protocol And RDF Query Language

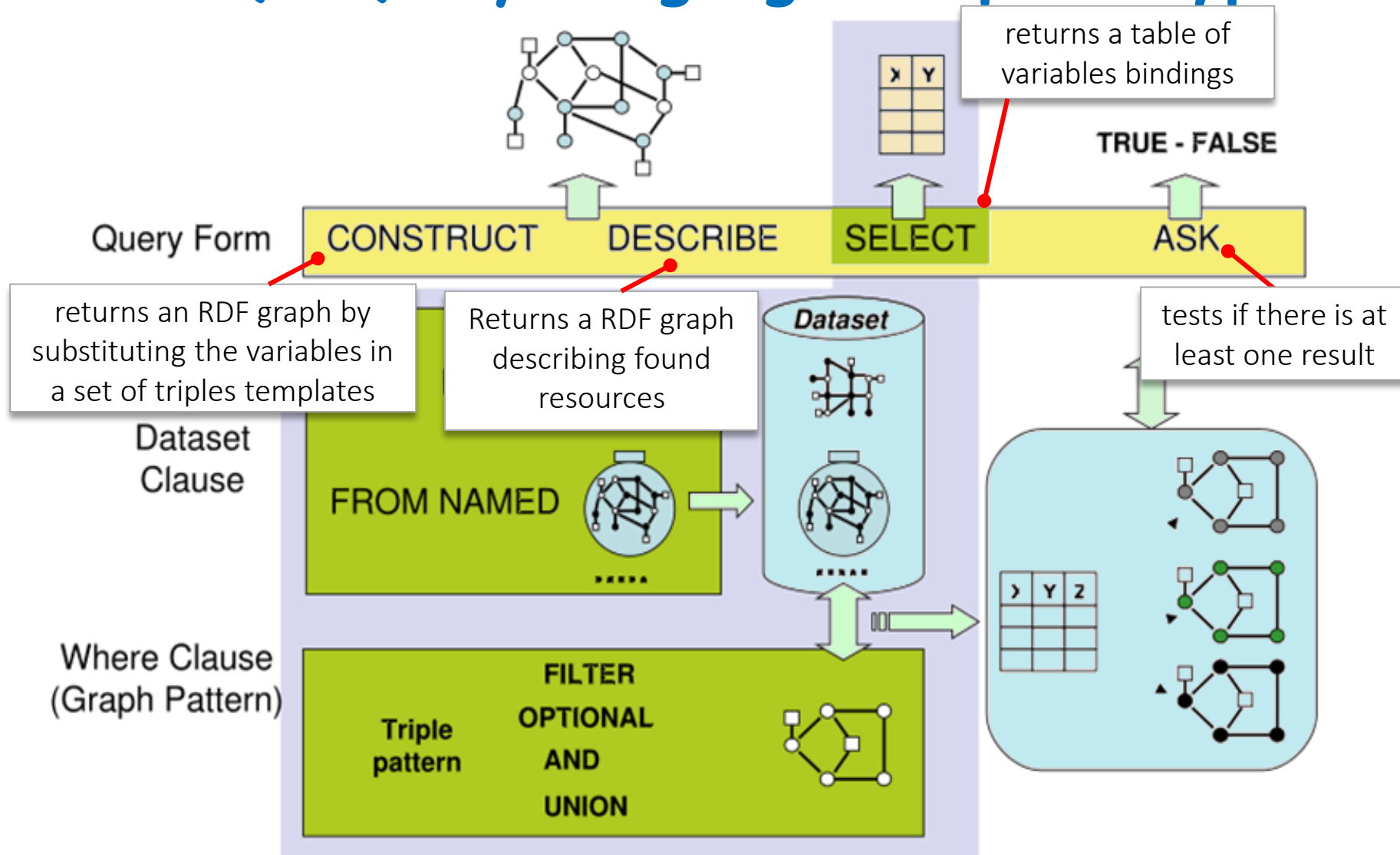
# SPARQL : introduction

- **RDF (Resource Description Framework)**
  - Flexible and extensible way to represent information about resources of the web
- **SPARQL (SPARQL Protocol And RDF Query Language)**
  - A W3C standard
    - SPARQL 1.0 recommendation - January 2008,
    - SPARQL 1.1 recommendation – March 2013
  - <http://www.w3.org/TR/2013/REC-sparql11-overview-20130321/>
  - a request language to access a RDF graph (SPARQL Query Language Specification) inspired from SQL
  - a protocol to submit request through HTTP GET, HTTP POST or SOAP (SPARQL protocol for RDF Specification)
  - an XML format for the results (SPARQL Query XML Results Format), and now JSON

# SPARQL Protocol And RDF Query Language overview



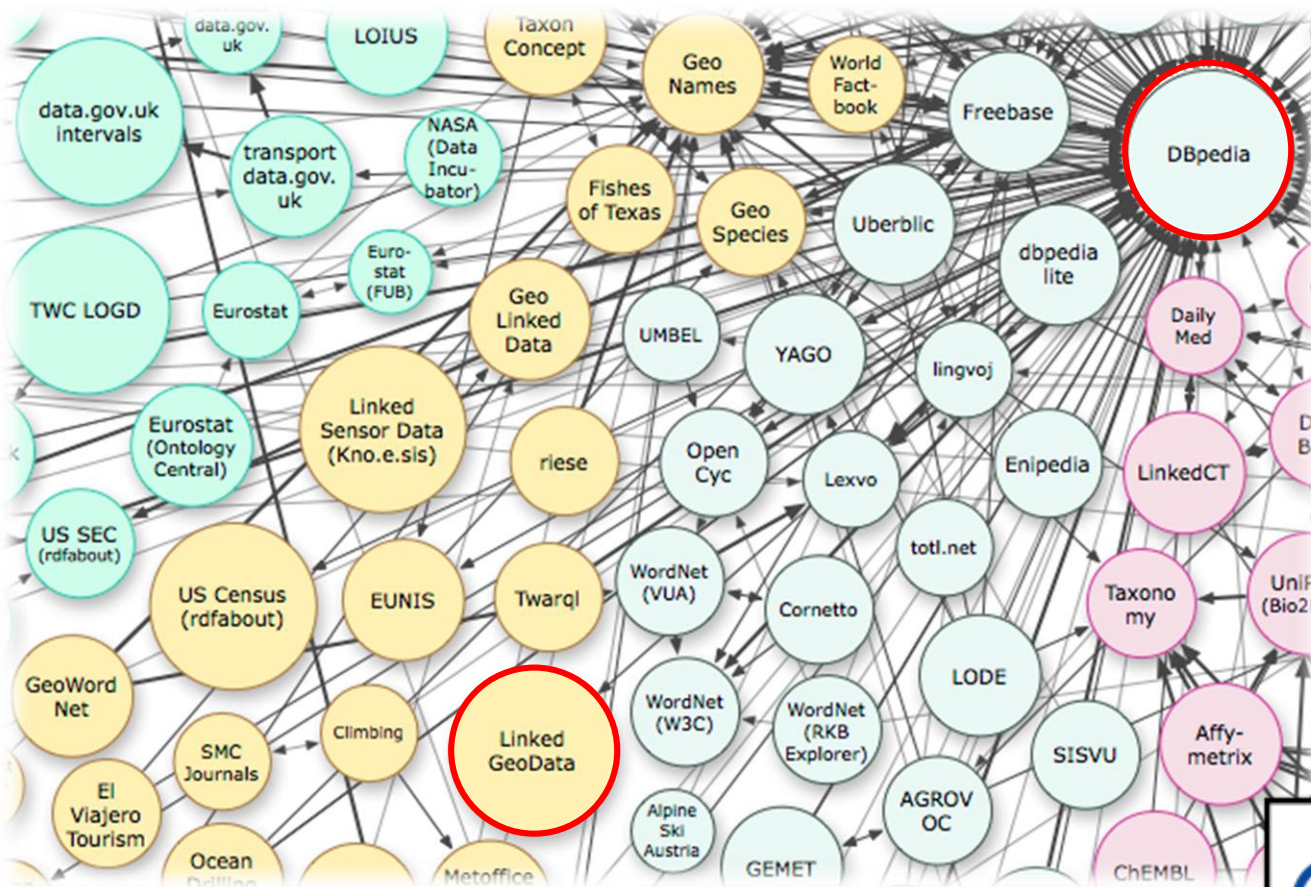
# SPARQL Query Language: request types



from: Pérez, Arenas and Gutierrez, Chapter 1: On the Semantics of SPARQL, Semantic Web Information Management: A Model Based Perspective, Springer 2010



# SPARQL in action with LinkedGeoData



OpenStreetMap.org



# SPARQL Query Language: SELECT

- find all the restaurants that are less than 1km from Fort Saint-Pierre (Théâtre de la Mer – Sète)

```
Prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
Prefix ogc:  <http://www.opengis.net/ont/geosparql#>
Prefix geom: <http://geovocab.org/geometry#>
Prefix lgdo: <http://linkedgeodata.org/ontology/>
```

```
Select ?s, ?l From <http://linkedgeodata.org>
```

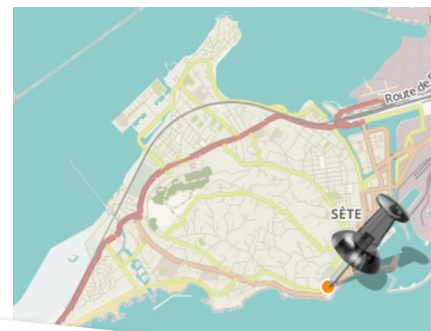
```
Where
```

```
{
```

```
  ?s    a lgdo:Restaurant ;
        rdfs:label ?l ;
        geom:geometry [
          ogc:asWKT ?g
        ] .
```

```
  Filter(bif:st_intersects (?g, bif:st_point (3.692764, 43.393794), 1)) .
```

```
}
```





# SPARQL Query Language: SELECT

- SPARQL based on :
  - RDF serialization with Turtle
  - Graph Pattern Matching

```
Prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
Prefix ogc: <http://www.opengis.net/ont/geosparql#>
Prefix geom: <http://geovocab.org/geometry#>
Prefix lgdo: <http://linkedgeodata.org/ontology/>
```

```
Select ?s, ?l From <http://linkedgeodata.org>
```

```
Where
```

```
{
```

variable

?s

```
  a lgdo:Restaurant ;
  rdfs:label ?l ;
  geom:geometry [
    ogc:asWKT ?g
  ] .
```

s?

rdf:type

lgdo:Restaurant

Graph Pattern: RDF triple containing one or more variables at any position (subject, predicate, object)

```
Filter(bif:st_intersects (?g, bif:st_point (3.692764, 43.393794), 1)) .
```

```
}
```

# SPARQL Query Language

## Select

- Graph patterns can be combined to construct complex (conjunctive) requests.

Prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>

Prefix ogc: <http://www.opengis.net/ont/geosparql#>

Prefix geom: <http://geovocab.org/geometry#>

Prefix lgdo: <http://linkedgeodata.org/ontology/>

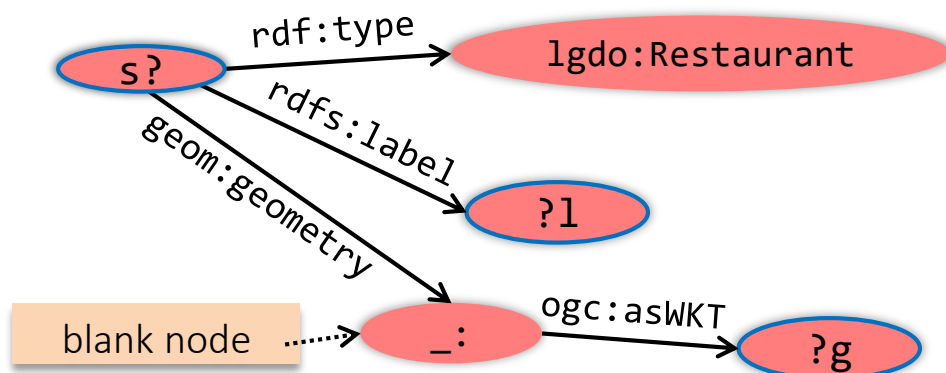
Select ?s, ?l From <http://linkedgeodata.org>

Where

{

variable → ?s

```
?s a lgdo:Restaurant ;  
  rdfs:label ?l ;  
  geom:geometry [  
    ogc:asWKT ?g  
  ] .
```



```
Filter(bif:st_intersects (?g, bif:st_point (3.692764, 43.393794), 1)) .
```

}

# SPARQL Query Language

## Select

- The SELECT clause indicates which variables to consider in the result
- \* all the variables (like SQL)

Prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>

Prefix ogc: <http://www.opengis.net/ont/geosparql#>

Prefix geom: <http://geovocab.org/geometry#>

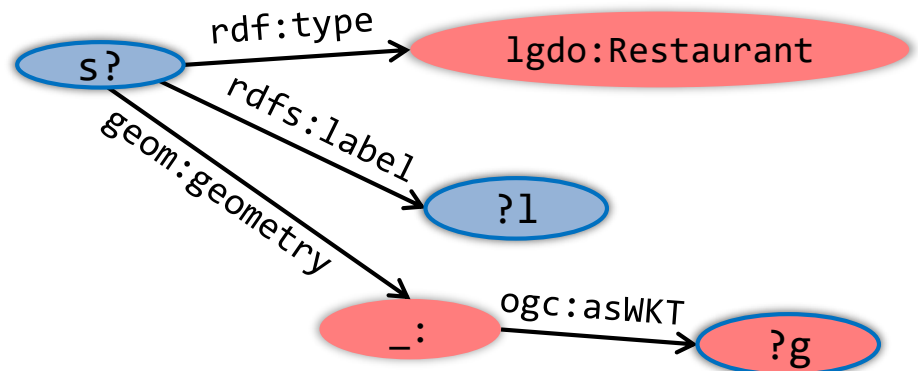
Prefix lgdo: <http://linkedgeodata.org/ontology/>

Select ?s, ?l From <http://linkedgeodata.org>

Where

{

```
?s    a lgdo:Restaurant ;
      rdfs:label ?l ;
      geom:geometry [
        ogc:asWKT ?g
      ] .
```



```
Filter(bif:st_intersects (?g, bif:st_point (3.692764, 43.393794), 1)) .
```

}

# SPARQL Query Language

## Select clause - DataSets

- The RDF data-store service can handle one or more RDF graphs, the SPARQL query is executed against a data set (RDF Dataset) that represents a collection of one or more graphs.

Prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>

Prefix ogc: <http://www.opengis.net/ont/geosparql#>

Prefix geom: <http://geovocab.org/geometry#>

Prefix lgdo: <http://linkedgeodata.org/ontology/>

Select ?s, ?l From <http://linkedgeodata.org>

Where

{

```
?s    a lgdo:Restaurant ;
      rdfs:label ?l ;
      geom:geometry [
        ogc:asWKT ?g
      ] .
```

graph on which the  
query is executed

```
Filter(bif:st_intersects (?g, bif:st_point (3.692764, 43.393794), 1)) .
```

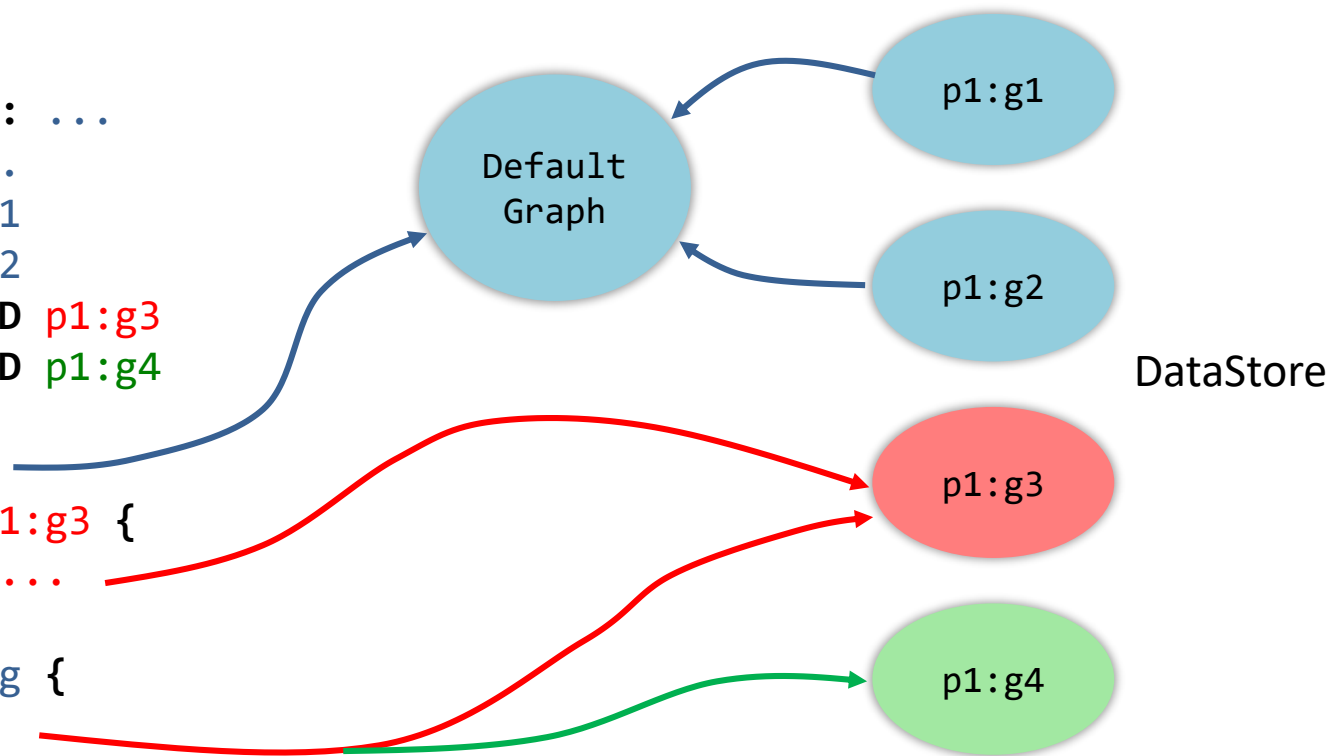
}

# SPARQL Query Language

## Select - DataSets

- a SPARQL request can associate different graph patterns to different named graphs.
  - FROM defines the default graph (this may be merging several graphs)
  - FROM NAMED defines graphs which then can be explicitly named in the WHERE part of the query through the GRAPH keyword

```
PREFIX p1: ...  
SELECT ...  
FROM p1:g1  
FROM p2:g2  
FROM NAMED p1:g3  
FROM NAMED p1:g4  
WHERE {  
  .....  
  GRAPH p1:g3 {  
    .....  
  }  
  GRAPH ?g {  
    .....  
  }  
}
```





# SPARQL Query Language

## Select - Filters

- Possibility to filter results
  - A filter allows to constrain solution values

Prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>

Prefix ogc: <http://www.opengis.net/ont/geosparql#>

Prefix geom: <http://geovocab.org/geometry#>

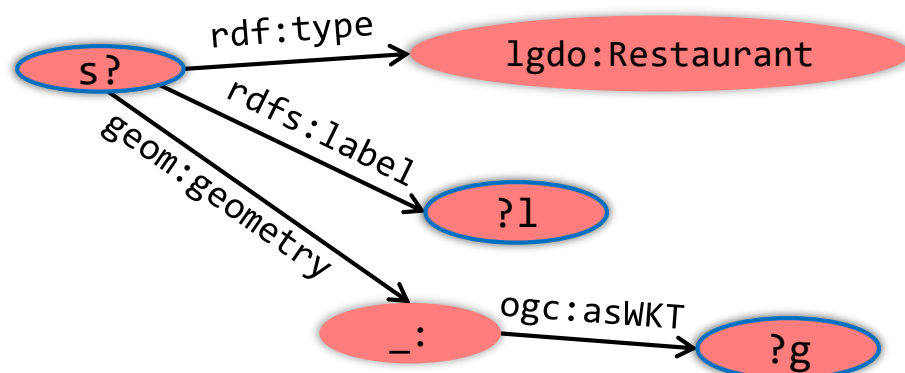
Prefix lgdo: <http://linkedgeodata.org/ontology/>

Select ?s, ?l From <http://linkedgeodata.org>

Where

{

```
?s a lgdo:Restaurant ;  
  rdfs:label ?l ;  
  geom:geometry [  
    ogc:asWKT ?g  
  ] .
```



```
Filter(bif:st_intersects (?g, bif:st_point (3.692764, 43.393794), 1)) .
```

Filter on ?g variable

# SPARQL Query Language

## Select -Filters

- Filters : allow to restrict the values in a solution
  - Boolean expression the request solutions must satisfy.
  - rich expression language based on Xpath, XQuery and special operators defined by SPARQL.  
see section 17 of SPARQL 1.1 Query Language specification document  
<https://www.w3.org/TR/2013/REC-sparql11-query-20130321/#expressions>
  - Relational operators: `<`, `>`, `=`, `<=`, `>=`, `!=`
    - Boolean operators: `&&`, `||`, `!`
    - Arithmetic operator: `+` `*` `-` `/`
    - Variable binding tests: `isURI(?x)`, `isBlank(?x)`, `isLiteral(?x)`, `bound(?x)`
    - Regular expressions: `regex(?x, "A.*")`
    - Access to attributes/values `lang()`, `datatype()`, `str()`
    - Casting ((re-)typing functions) `xsd:integer(?x)`
    - External functions / extensions

# SPARQL Query Language

## Solution Modifiers

- The set of solution produced by graphs pattern matching can be modified in various ways:
  - Projection - keep only selected variables
  - OFFSET/LIMIT - chop the number solutions (best used with ORDER BY)
    - OFFSET the start index,
    - LIMIT the number of solutions to be returned.
      - Using LIMIT alone useful to ensure not too many solutions are returned, to restrict the effect of some unexpected situation
  - ORDER BY - sorted results
  - DISTINCT - yield only one row for one combination of variables and values.
- The solution modifiers OFFSET/LIMIT and ORDER BY always apply to all result forms

example

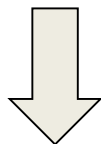
```
prefix foaf: <http://xmlns.com/foaf/0.1/>
SELECT DISTINCT * where {
    ?x foaf:name ?name;
    foaf:age ?age.
}
ORDER BY ?name DESC(?age)
```

# SPARQL Query Language

## ASK query

- ASK – ask a boolean query.
  - to verify that there is at least one response.
  - Is there a student over 30 years?

```
PREFIX ufrimag: <http://www.ufrimag.fr/onto#>
ASK {
  ?etudiant ufrimag:inscrit ?x .
            ufrimag:age ?age .
  FILTER (?age > 30)
}
```



the boolean result

```
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
  <head> ... </head>
  <boolean> true </boolean>
</sparql>
```

# SPARQL Query Language

## CONSTRUCT query

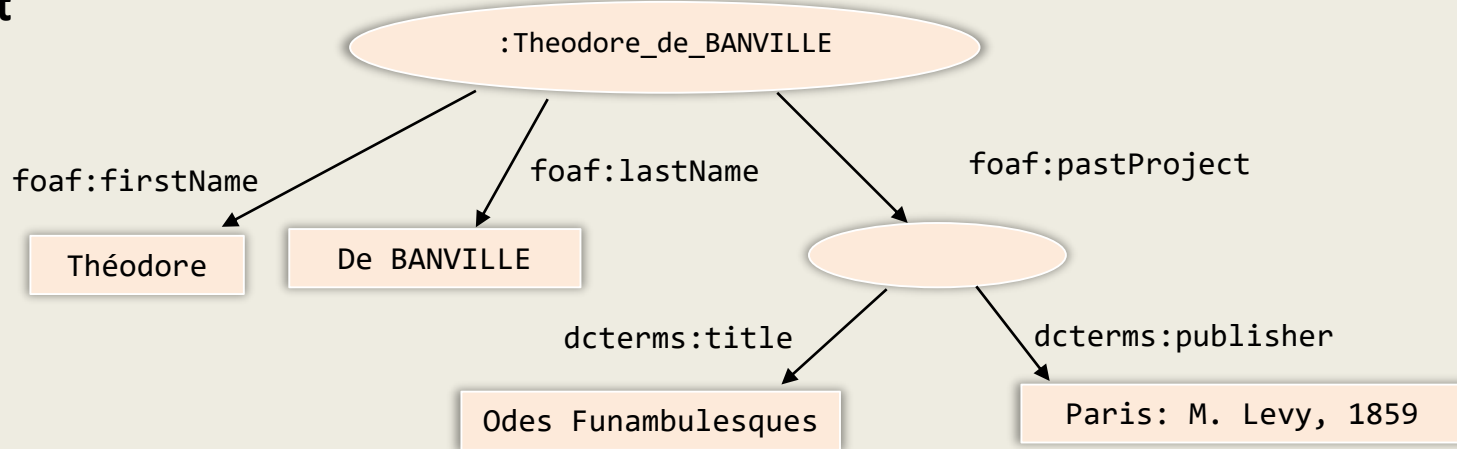
- **SELECT** returns a flat list of variables bindings
  - the application program is in charge of processing these bindings (sometimes by converting solution tuples into triples and adding them to a RDF graph)
- **CONSTRUCT** allows you to directly product a RDF graph containing the variables values
  - the WHERE and FILTER clause works the same way as the SELECT form
  - bindings of the variables are inserted into a new graph constructed from template triples specified in the CONSTRUCT clause (which replace the SELECT clause).



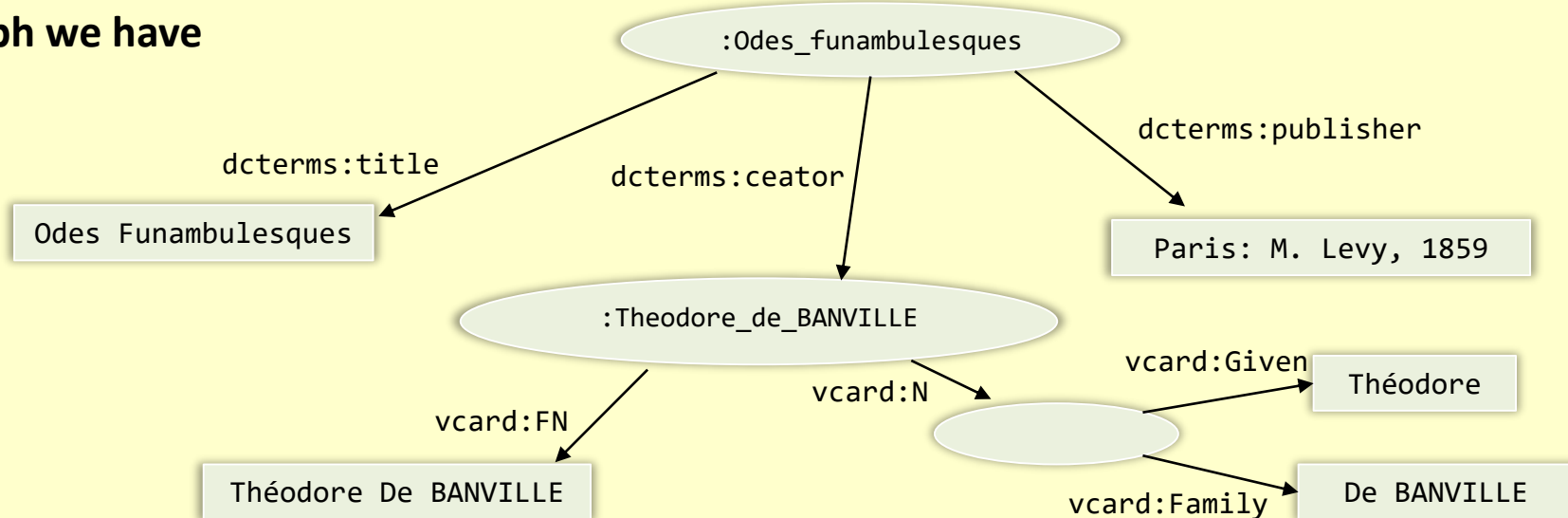
# SPARQL Query Language

## CONSTRUCT query

### The graph we want



### The graph we have

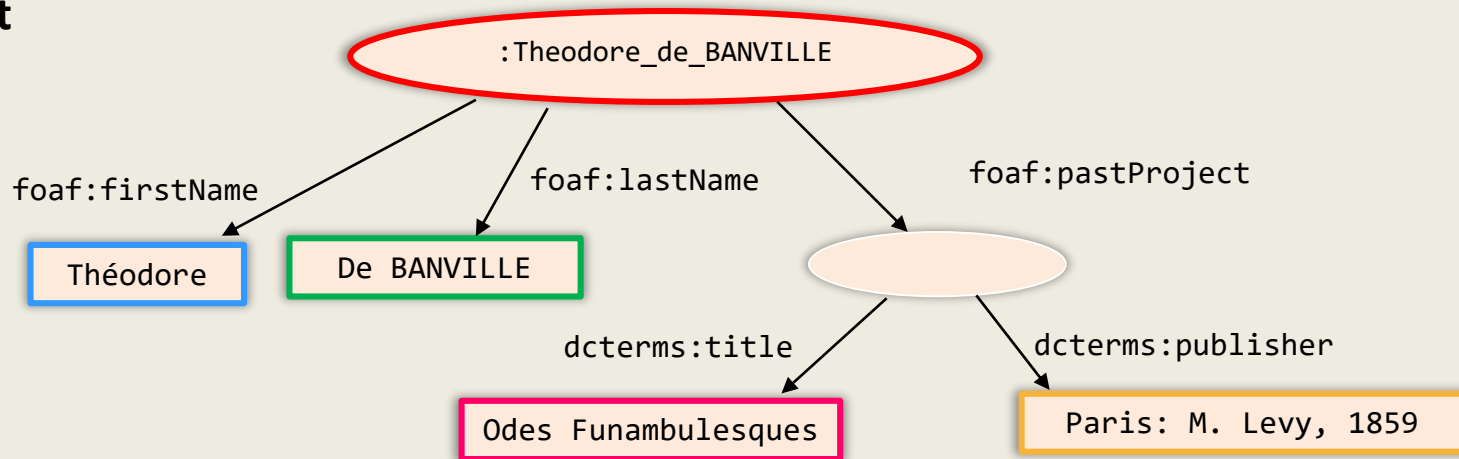


D'après : <http://pagesperso-systeme.lip6.fr/Jean-Francois.Perrot/inalco/XML/RDF/SPARQL/IntroSPARQL.html>

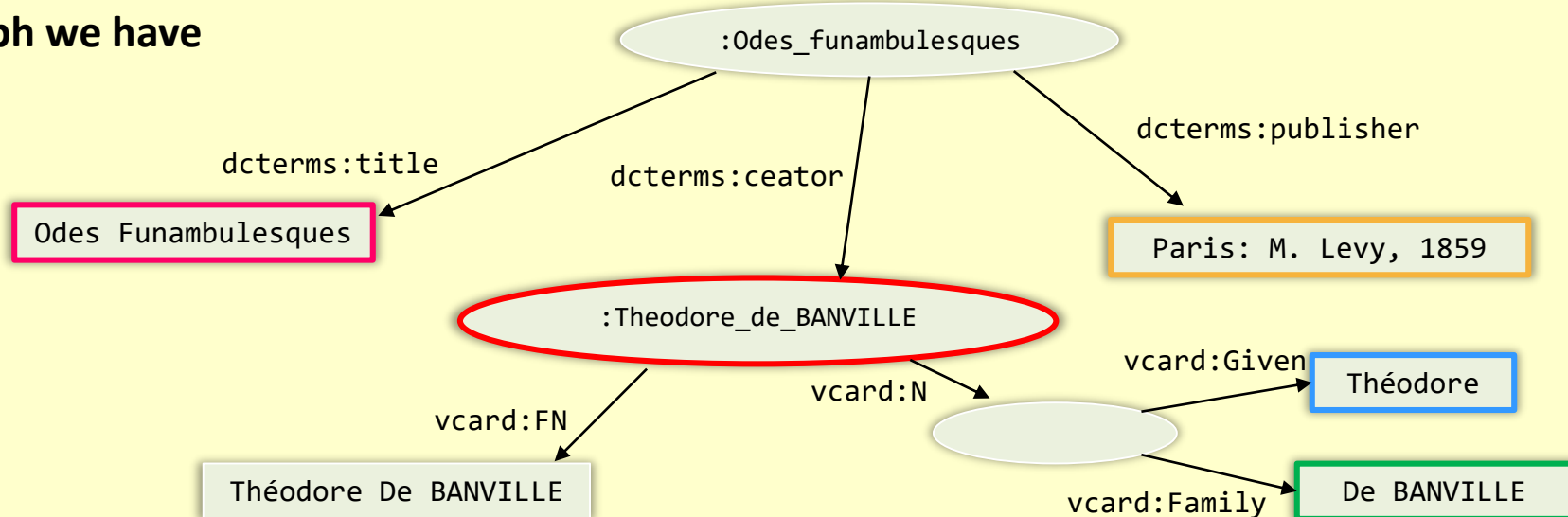
# SPARQL Query Language

## CONSTRUCT query

The graph we want

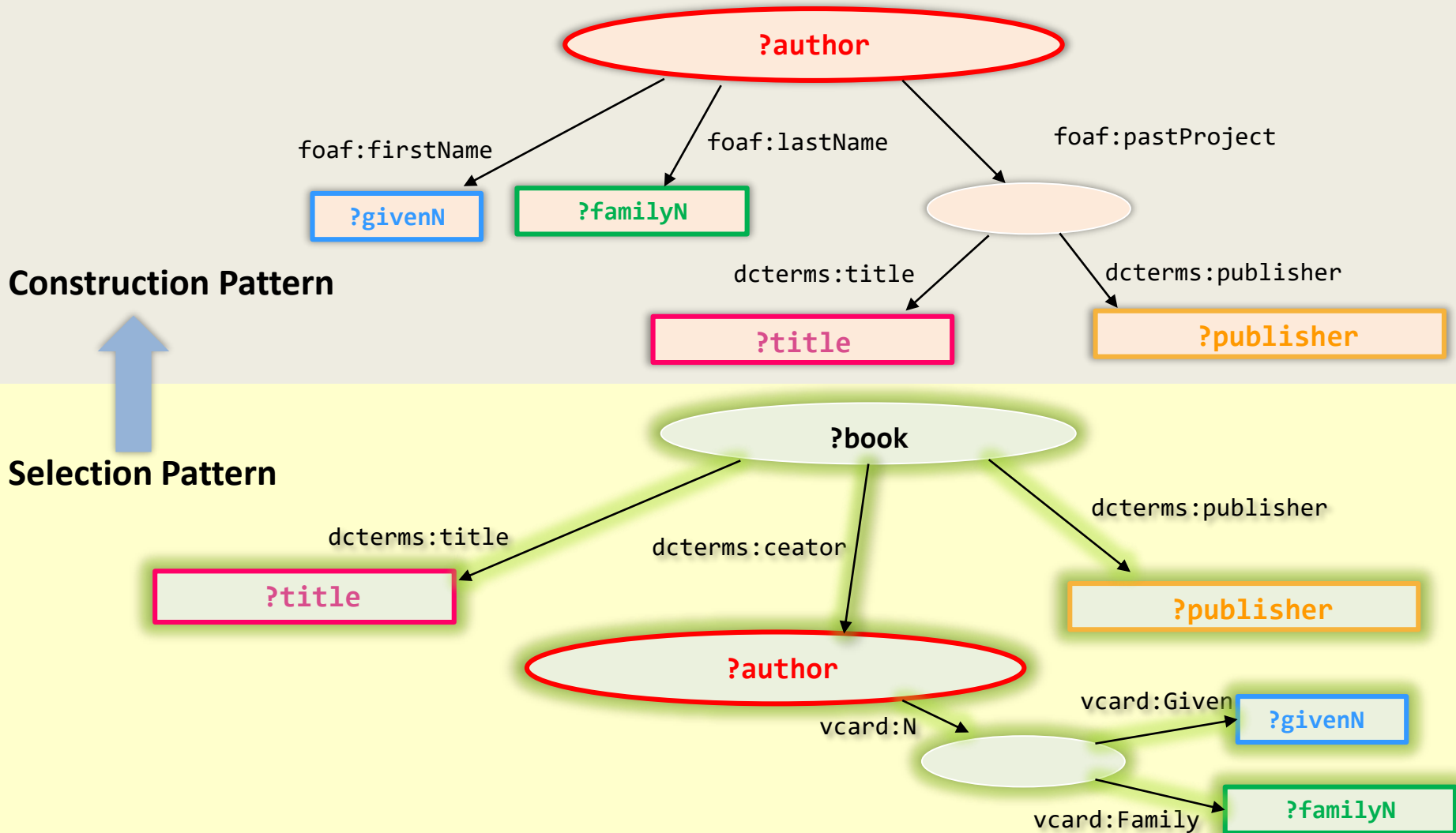


The graph we have



# SPARQL Query Language

## CONSTRUCT query

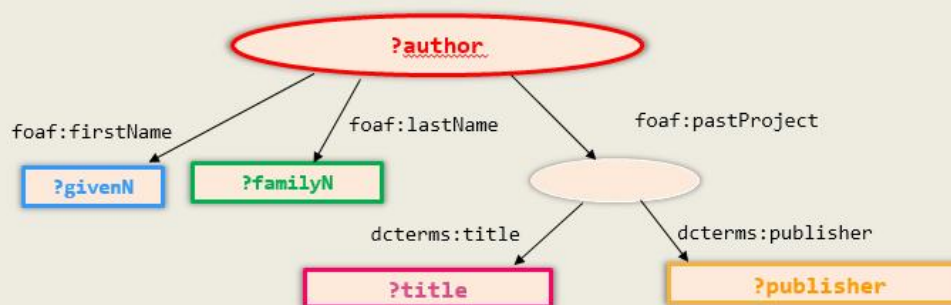


# SPARQL Query Language

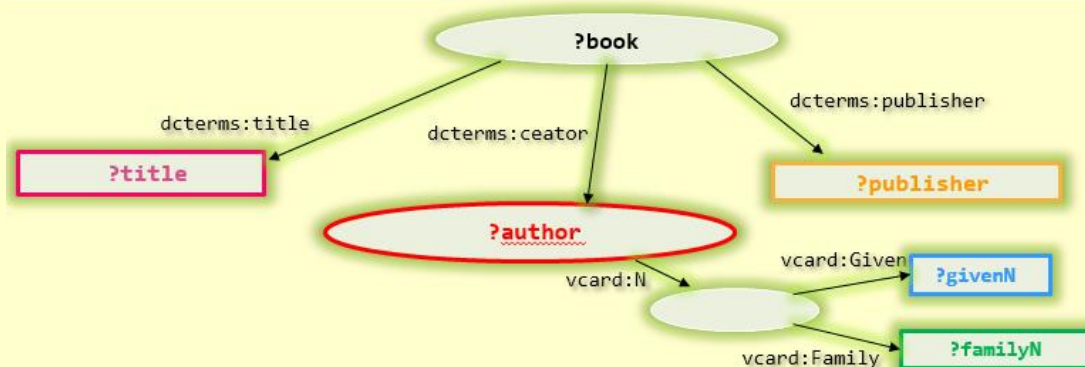
## CONSTRUCT query

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX vcard: <http://www.w3.org/2001/vcard-rdf/3.0#>
```

```
CONSTRUCT {
  ?author foaf:firstName ?givenN;
  foaf:lastName ?familyN;
  foaf:pastProject [
    dc:title ?title;
    dc:publisher ?publisher
  ]
}
```



```
WHERE {
  ?book dc:title ?title;
  dc:creator ?author;
  dc:publisher ?publisher .
  ?author vcard:N [
    vcard:Given ?givenN;
    vcard:Family ?familyN
  ] .
}
```



# SPARQL Query Language

## DESCRIBE query

- **DESCRIBE** – Returns an RDF graph, based on what the query processor is configured to return.
  - SPARQL specification says : "the useful information the service has about a resource"
  - in theory this should help you understand the context of the resources returned... but there is no warranty.

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX ex: <http://example.com/>
DESCRIBE ex:karen ?friend {
  ex:karen foaf:knows ?friend .
}
```

asks for a description of karen ad her friends

Subject	Predicate	Object	
ex:karen	foaf:knows	ex:alex	×
ex:karen	foaf:name	"Karen"	×
ex:alex	foaf:name	"Alex"	×

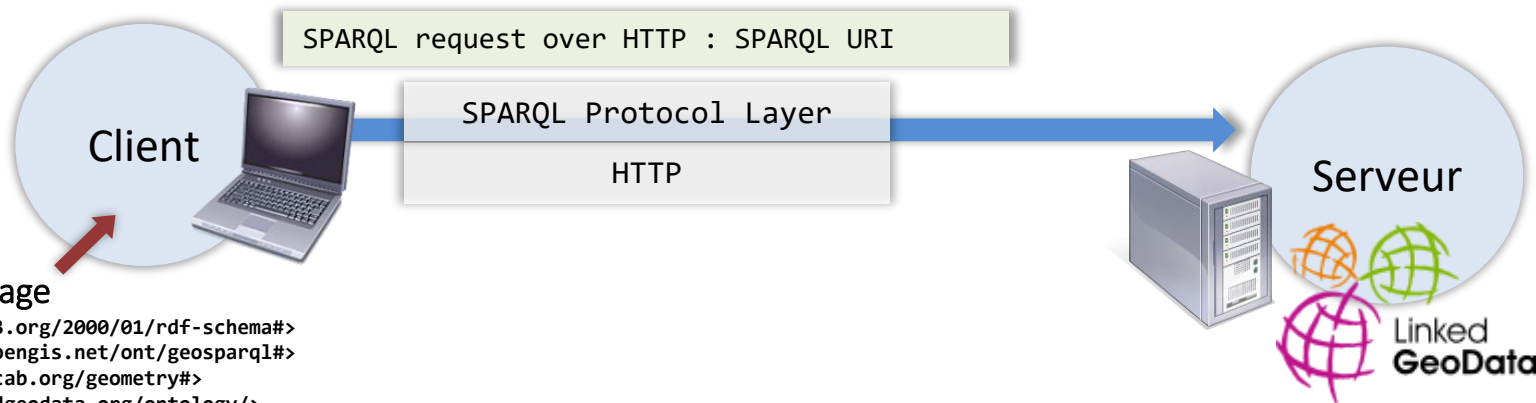
```
DESCRIBE <http://example.com/fish> ?x WHERE {
  ?x ?y <http://example.com/fish>
}
```

asks for a description of fish, and any resource directly related to fish.



# SPARQL Protocol

```
http://linkedgeo.org/sparql?default-graph-uri=http%3A%2F%2Flinkedgeo.org
&query=Prefix+rdfs%3A+%3Chttp%3A%2F%2Fwww.w3.org%2F2000%2F01%2Frdf-
schema%23%3E%0D%0APrefix+ogc%3A+%3Chttp%3A%2F%2Fwww.opengis.net%2Font%2Fgeospa
rql%23%3E%0D%0APrefix+geom%3A+%3Chttp%3A%2F%2Fgeovocab.org%2Fgeometry%23%3E%0
D%0APrefix+lgdo%3A+%3Chttp%3A%2F%2Flinkedgeo.org%2Fontology%2F%3E%0D%0A%0D%
0ASelect+*%0D%0AFrom+%3Chttp%3A%2F%2Flinkedgeo.org%3E+%7B%0D%0A++%3Fs%0D%
0A++++a+lgdo%3ARestaurant+%3B%0D%0A++++rdfs%3Alabel+%3FI+%3B++++%0D%0A++++geom
%3Ageometry+%5B%0D%0A+++++ogc%3AasWKT+%3Fg%0D%0A++++%5D+.%0D%0A%0D%0A++
++Filter%28bif%3Ast_intersects+%28%3Fg%2C+bif%3Ast_point+%283.692764%2C+43.393794%29
%2C+1%29%29+.%0D%0A%7D&format=application%2Fsparql-results%2Bxml
```



## SPARQL Query Language

```
Prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
Prefix ogc: <http://www.opengis.net/ont/geosparql#>
Prefix geom: <http://geovocab.org/geometry#>
Prefix lgdo: <http://linkedgeo.org/ontology/>
```

```
Select ?s, ?l From <http://linkedgeo.org>
Where {
  ?s a lgdo:Restaurant ;
  rdfs:label ?l ;
  geom:geometry [
    ogc:asWKT ?g
  ] .
  Filter(bif:st_intersects (?g, bif:st_point (3.692764, 43.393794), 1)) .
}
```

# SPARQL Protocol

- SPARQL request over HTTP (SPARQL URI) has 4 components:

the SPARQL endpoint URL

<http://linkedgeodata.org/sparql>

The RDF graphs to request (optional)

**default-graph-uri**=http://linkedgeodata.org  
**named-graph-uri**=...

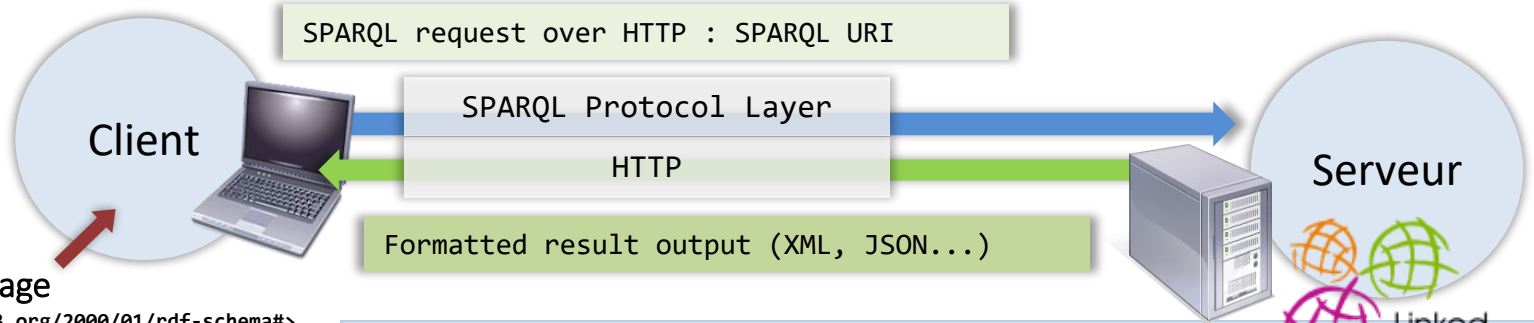
[http://linkedgeodata.org/sparql?default-graph-uri=http%3A%2F%2Flinkedgeodata.org&query=Prefix+rdfs%3A+%3Chttp%3A%2F%2Fwww.w3.org%2F2000%2F01%2Frdf-schema%23%3E%0D%0APrefix+ogc%3A+%3Chttp%3A%2F%2Fwww.opengis.net%2Font%2Fgeosparql%23%3E%0D%0APrefix+geom%3A+%3Chttp%3A%2F%2Fgeovocab.org%2Fgeometry%23%3E%0D%0APrefix+lgdo%3A+%3Chttp%3A%2F%2Flinkedgeodata.org%2Fontology%2F%3E%0D%0A%0D%0ASelect+\\*%0D%0AFrom+%3Chttp%3A%2F%2Flinkedgeodata.org%3E+%7B%0D%0A+++%3Fs%0D%0A++++a+lgdo%3ARestaurant+%3B%0D%0A++++rdfs%3Alabel+%3FI+%3B++++%0D%0A++++geom%3Ageometry+%5B%0D%0A+++++ogc%3AasWKT+%3Fg%0D%0A++++%5D+.%0D%0A%0D%0A++++Filter%28bif%3Ast\\_intersects+%28%3Fg%2C+bif%3Ast\\_point+%283.692764%2C+43.393794%29%2C+1%29%29+.%0D%0A%7D&format=application%2Fsparql-results%2Bxml](http://linkedgeodata.org/sparql?default-graph-uri=http%3A%2F%2Flinkedgeodata.org&query=Prefix+rdfs%3A+%3Chttp%3A%2F%2Fwww.w3.org%2F2000%2F01%2Frdf-schema%23%3E%0D%0APrefix+ogc%3A+%3Chttp%3A%2F%2Fwww.opengis.net%2Font%2Fgeosparql%23%3E%0D%0APrefix+geom%3A+%3Chttp%3A%2F%2Fgeovocab.org%2Fgeometry%23%3E%0D%0APrefix+lgdo%3A+%3Chttp%3A%2F%2Flinkedgeodata.org%2Fontology%2F%3E%0D%0A%0D%0ASelect+*%0D%0AFrom+%3Chttp%3A%2F%2Flinkedgeodata.org%3E+%7B%0D%0A+++%3Fs%0D%0A++++a+lgdo%3ARestaurant+%3B%0D%0A++++rdfs%3Alabel+%3FI+%3B++++%0D%0A++++geom%3Ageometry+%5B%0D%0A+++++ogc%3AasWKT+%3Fg%0D%0A++++%5D+.%0D%0A%0D%0A++++Filter%28bif%3Ast_intersects+%28%3Fg%2C+bif%3Ast_point+%283.692764%2C+43.393794%29%2C+1%29%29+.%0D%0A%7D&format=application%2Fsparql-results%2Bxml)

The query string  
(URL encoded)

Output format (MIME type)

**format**=application/sparql-results+xml  
(text/html, json...)

# SPARQL output format



## SPARQL Query Language

```
Prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
Prefix ogc: <http://www.opengis.net/ont/geosparql#>
Prefix geom: <http://geovocab.org/geometry#>
Prefix lgdo: <http://linkedgeodata.org/ontology/>
```

```
Select ?s, ?l From <http://linkedgeodata.org>
Where {
  ?s a lgdo:Restaurant ;
  rdfs:label ?l ;
  geom:geometry [
    ogc:asWKT ?g
  ] .
  Filter(bif:st_intersects (?g, bif:st_point (3.692764, 43.393794), 1))
}
```

```
<sparql xmlns="http://www.w3.org/2005/sparql-results#"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2001/sw/DataAccess/rf1/result2.xsd">
  <head>
    <variable name="s"/>
    <variable name="l"/>
  </head>
  <results distinct="false" ordered="true">
    <result>
      <binding name="s">
        <uri>http://linkedgeodata.org/triplify/node1304006380</uri>
      </binding>
      <binding name="l">
        <literal>L&#39;Hostal</literal>
      </binding>
    </result>
    <result>
      <binding name="s">
        <uri>http://linkedgeodata.org/triplify/node1952213273</uri>
      </binding>
      <binding name="l">
        <literal>L&#39;Ultima</literal>
      </binding>
    </result>
  </results>
</sparql>
```

Variables from the SELECT clause

one **result** element for each row in the result set

**binding** of one the variables of this result



# SPARQL 1.1: new functionalities

- W3C Recommendation, March 2013, 21
  - <http://www.w3.org/TR/2013/REC-sparql11-overview-20130321/>
- Query
  - New aggregation functions for results (**count**, **min**, **max**, **group by**, etc.)
  - Variables assignment
    - `SELECT (COUNT(DISTINCT ?s)) AS ?num` *number of distinct restaurants*
  - Negation - MINUS
    - **NOT EXISTS**, **EXIST** :filtering results depending on whether a graph pattern does or does not match in the context of the query solution being filtered,

```
PREFIX rdf:    <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX foaf:  <http://xmlns.com/foaf/0.1/>
```

```
SELECT ?person
WHERE
{
  ?person rdf:type foaf:Person .
  FILTER NOT EXISTS { ?person foaf:name ?name }
}
```

*Persons who don't have a name*

Example from

<http://www.w3.org/TR/2013/REC-sparql11-query-20130321/#negation>

- **MINUS** removing solutions related to another pattern.

# SPARQL 1.1: new functionalities

- Query...
  - subqueries: possibility to embed SPARQL queries within other queries
    - e.g. for limiting the number of results from some sub-expression within the query
  - subqueries are evaluated logically first, and the results are projected up to the outer query.

Example\*: Return a name (the one with the lowest sort order) for all the people that know Alice and have a name. • From <http://www.w3.org/TR/2013/REC-sparql11-query-20130321/#subqueries>

```
@prefix : <http://people.example/> . The data
:alice :name "Alice", "Alice Foo", "A. Foo" .
:alice :knows :bob, :carol .
:bob :name "Bob", "Bob Bar", "B. Bar" .
:carol :name "Carol", "Carol Baz", "C. Baz" .
```

y	minName
:bob	"B. Bar"
:carol	"C. Baz"

y
:bob
:carol

```
PREFIX : <http://people.example/>
SELECT ?y ?minName
WHERE {
  :alice :knows ?y .
  {
    SELECT ?y (MIN(?name) AS ?minName)
    WHERE {
      ?y :name ?name .
    } GROUP BY ?y
  }
}
```

1: inner query evaluation

2: outer query evaluation

3: results of 1. are joined with results of 2.

y	minName
:alice	"A. Foo"
:bob	"B. Bar"
:carol	"C. Baz"

# SPARQL 1.1: new functionalities

- Query...
  - Basic federated queries (**SERVICE, BINDING**)
    - To execute requests distributed over different SPARQL endpoints

Example\*: *is there anyone among Alice's friends with the same name as the resource identified by the IRI `<http://dbpedia.org/resource/Snoopy>` at Dbpedia?*

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name
WHERE {
  <http://example.org/alice#me> foaf:knows [ foaf:name ?name ] .
  SERVICE <http://dbpedia.org/sparql> {
    <http://dbpedia.org/resource/Snoopy> foaf:name ?name
  }
}
```

names of Alice's friend. This pattern is matched against the local SPARQL service

find out the name of <http://dbpedia.org/resource/Snoopy>  
evaluation of this pattern is delegated to the respective remote SPARQL service `http://dbpedia.org/sparql`

\* From <http://www.w3.org/TR/sparql11-overview/>

# SPARQL 1.1: new functionalities ...

- New serialization formats for request results (JSON...)
- CRUD operations
  - Graph update: INSERT, INSERT DATA, DELETE DATA, DELETE, DELETE WHERE, LOAD, CLEAR)
  - Graph management: CREATE, DROP, COPY, MOVE, ADD
- Entailments
  - RDF, RDFS, OWL, RIF
- ...

<http://www.w3.org/TR/2013/REC-sparql11-overview-20130321/>