TALOS ERA CHAIR IN ARTIFICIAL INTELLIGENCE FOR HUMANITIES AND SOCIAL SCIENCES

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ
UNIVERSITY OF CRETE

"SPARQL"

Rafail Giannadakis
Research Assistant, TALOS-AI*4*SSH
giannadakis.uni@gmail.com

Funded by the European Union

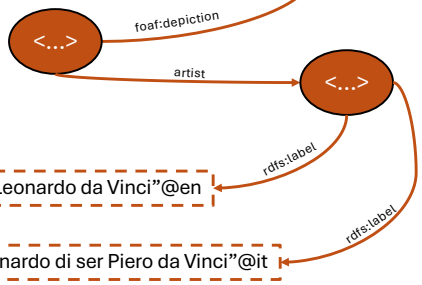
Horizon ERA Chair TALOS AI4SSH Project funded by the European Commission
Grant Agreement n° 101087269, https://cordis.europa.eu/project/id/101087269

Hello everyone! In this session of our MOOC, we will explore SPARQL!

Contents

- Recap of RDF
- What is SPARQL?
  - Query Graph Structure
  - Breakdown of SPARQL Queries
  - Executing SPARQL Queries
- Conclusion

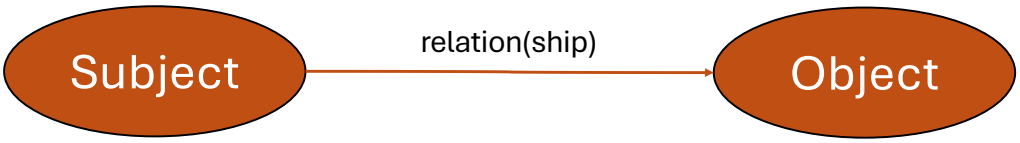
Mona Lisa, by Leonardo da Vinci. Originally: Galerie de tableaux en très haute définition, Public Domain, https://commons.wikimedia.org/w/index.php?curid=15642524



Here's what we'll cover today:
First, we'll briefly recap RDF.
Then, we'll define what SPARQL is, explore how SPARQL queries are structured, and break down their components.
Next, we'll see how to execute SPARQL queries with some examples.
Finally, we'll wrap up by discussing why SPARQL is such a powerful and useful tool.

Recap of RDF

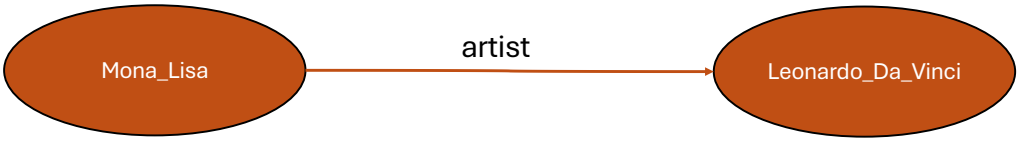
- RDF graphs=> represent web information
- Directed and labelled graphs
- Triples



Before we dive into SPARQL, let's refresh our understanding of RDF graphs.
These graphs represent knowledge using RDF—the Resource Description Framework standard.
More specifically, RDF is used to represent web information as directed, labeled graphs.
It does so by using **triples**, consisting of a **subject**, a **predicate** (or relationship), and an **object**.

Recap of RDF

- RDF graphs=> represent web information
- Directed and labelled graphs
- Triples



Think of it like this:
Mona_Lisa → artist → Leonardo Da Vinci
Each triple forms part of a larger graph that connects various pieces of information on the web. This graph structure is crucial because SPARQL queries are specifically designed to interact with and mirror these RDF graphs.

What is SPARQL?

- SPARQL Protocol and RDF Query Language
- Query and retrieve data stored or viewed in RDF format.
- "Trying to use the Semantic Web without SPARQL is like trying to use a relational database without SQL".
  Tim Berners-Lee

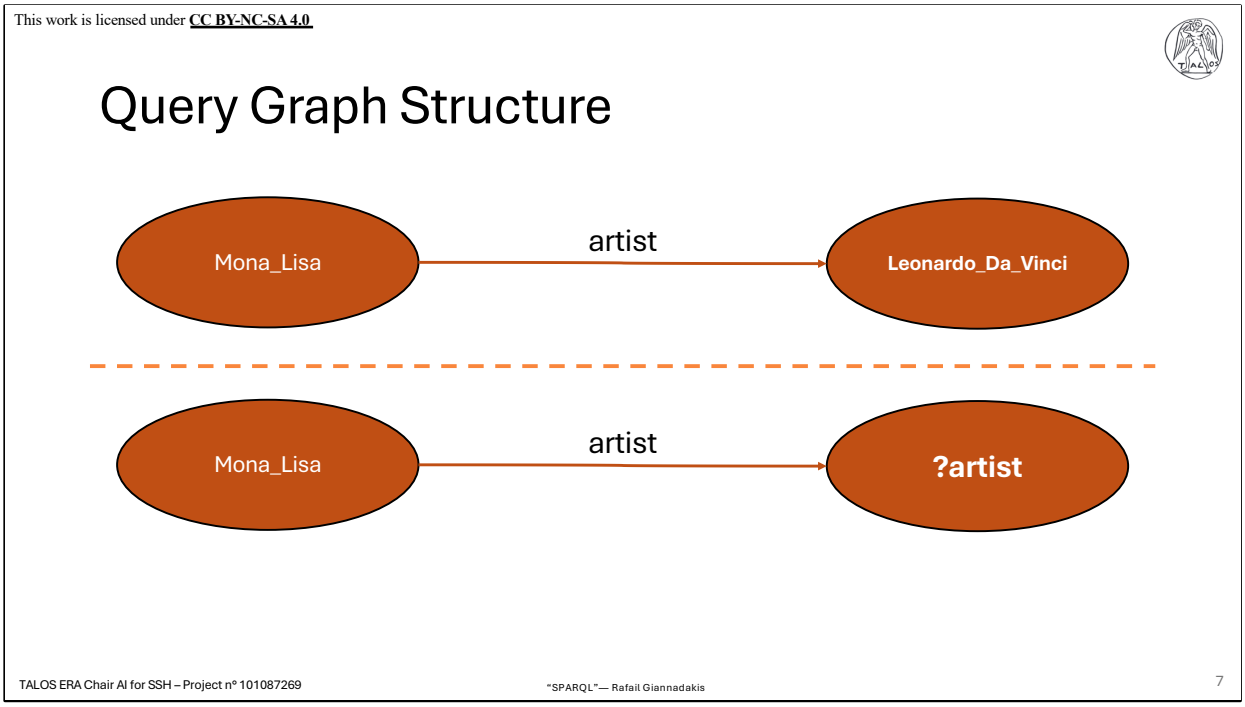
So, what exactly is SPARQL?
SPARQL stands for **SPARQL Protocol and RDF Query Language**.
It's the standard way to query and retrieve data stored or viewed in RDF format.
In this way, the user can retrieve data by querying databases or any data source that can be mapped to RDF.
Tim Berners-Lee, the inventor of the web, once said:
"Trying to use the Semantic Web without SPARQL is like trying to use a relational database without SQL."
So, if RDF is our data model, SPARQL is the tool we use to query and manipulate that data effectively.

Query Graph Structure

- RDF graphs
- SPARQL queries

Triples

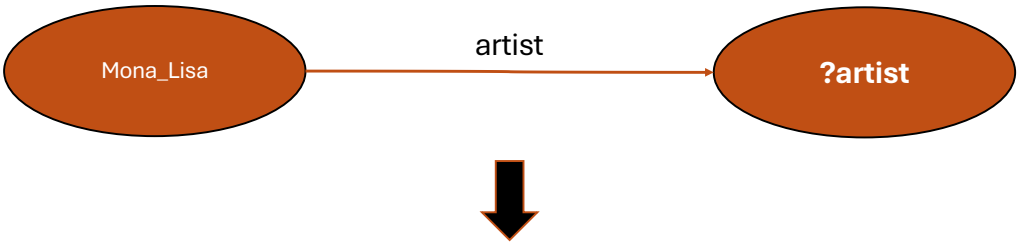
Just as RDF graphs consist of linked triples—made up of subjects, predicates, and objects—SPARQL query patterns also operate in a graph-oriented structure, meaning they are based on triples.



More specifically, in both RDF graphs and SPARQL queries, we have nodes and links with standardized or defined vocabularies.

But in the query graphs, we also use **variables** to represent data we are searching for, or data that are currently unknown.

Query Graph Structure




```
SELECT ?artist
WHERE { Mona_Lisa artist ?artist. }
```


Here's a simple example:
Say we want to find out who painted the Mona Lisa.
Our query form and pattern will look like this:
We want to **SELECT** and retrieve the value of the variable **?artist**, which is linked in the **WHERE** clause to the subject **Mona_Lisa**.
In essence, it mirrors the RDF triple; we just replace the object with a variable because it's unknown, and we select to retrieve it.

Breakdown of SPARQL Queries

```
PREFIX ex: <example-namespace-URI>
PREFIX rdfs: <www.w3.org/2000/01/rdf-schema>

SELECT ?artistName

FROM <example-dataset-URI>

WHERE {
ex:Mona_Lisa ex:artist ?artistURI.
?artistURI rdfs:label ?artistName.
}

LIMIT ...
```

Namespaces

Query Form

RDF Datasets URI

Query graph pattern

Modifiers

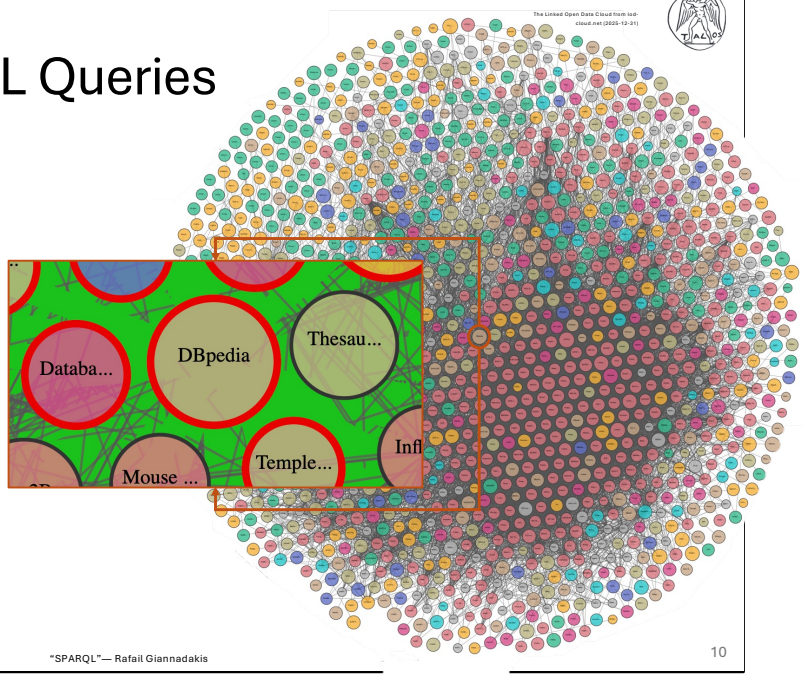
Now, let's break down the anatomy of a complete SPARQL query:
1.PREFIXes — These are the namespaces or vocabularies used in your query.
2.Query Form — The operation you want to perform. The most common query forms in SPARQL include:

1. **SELECT**: Returns all or a subset of variables matched in the query pattern.
2. **CONSTRUCT**: Builds a new RDF graph based on variable substitutions.
3. **ASK**: Returns a boolean indicating whether a query pattern matches.
4. **DESCRIBE**: Returns an RDF graph that describes resources found.

3.FROM: Optionally indicates the dataset URI you're querying.
4.WHERE: Defines the graph pattern you're matching. You can also include keywords here, such as filters etc.
5.Modifiers: Since query patterns generate an unordered collection of results, modifiers like **LIMIT**, **ORDER BY**, or **DISTINCT** can refine how results are returned.

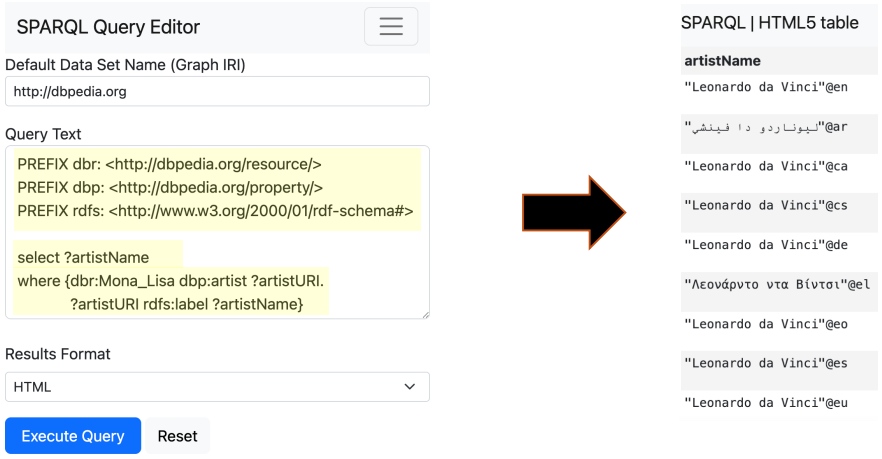
Executing SPARQL Queries

- SPARQL endpoints
- SPARQLer - General purpose processor
- DBpedia, BnF, data.europa.eu, Wikidata, Europeana...



To execute SPARQL queries, we use **SPARQL endpoints**.
You can use processors such as **SPARQLer** to query an RDF graph by providing its URI, or you can directly query well-known graphs via their own endpoints.
Some examples include **DBpedia, BnF, the European data portal, Wikidata, Europeana**, and of course there are more.
These endpoints return results in various formats—JSON, XML, HTML—depending on your needs.
Now, let's move on to some concrete examples using the DBpedia endpoint.

Executing SPARQL Queries



Let's revisit our question:
What's the name of the person who painted the Mona Lisa?
We first include the **PREFIXes** for the vocabularies used, even though DBpedia's endpoint allows us to omit them.
We use the **SELECT** query form to retrieve data for the variable **?artistName**.
And in the query pattern:
1.We link the **Mona_Lisa** resource to an object via the **artist** property.
2.That object becomes the subject to retrieve its labels using the RDF Schema property **rdfs:label**.
Since no filters are added, we get results for Leonardo da Vinci in multiple languages.

Executing SPARQL Queries

SPARQL Query Editor

Default Data Set Name (Graph IRI)

http://dbpedia.org

Query Text

```
PREFIX dbr: <http://dbpedia.org/resource/>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

select ?artistDepiction
where {dbr:Mona_Lisa dbp:artist ?artistURI.
       ?artistURI foaf:depiction ?artistDepiction}
LIMIT 5
```

Results Format

HTML

Execute Query Reset

SPARQL | HTML5 table

artistDepiction
http://commons.wikimedia.org/wiki/Special:FilePath/Leonardo_da_Vinci_-_RCIN_919000,_Verso_The_bones_and_muscles_of_the_arm_c.1510-11.jpg
http://commons.wikimedia.org/wiki/Special:FilePath/Da_Vinci_Vitruve_Luc_Viatour_(cropped).jpg
http://commons.wikimedia.org/wiki/Special:FilePath/Leonardo_da_Vinci_-_presumed_self-portrait_-_WGA12798.jpg
http://commons.wikimedia.org/wiki/Special:FilePath/Mona_Lisa,_by_Leonardo_da_Vinci,_from_C2RMF_retouched.jpg
http://commons.wikimedia.org/wiki/Special:FilePath/A_portrait_of_Leonardo,_by_Francesco_Melzi.jpg

Here's another example, like the previous query.
But instead of asking for the labels of the artist resource, this time, we request depictions—by utilizing the **foaf:depiction** property, which we limit to 5.

This shows the importance of accuracy, along with the flexibility of SPARQL: you can retrieve various types of data linked to the same resource simply by just changing the properties...

Executing SPARQL Queries

SPARQL Query Editor

Default Data Set Name (Graph IRI)

http://dbpedia.org

Query Text

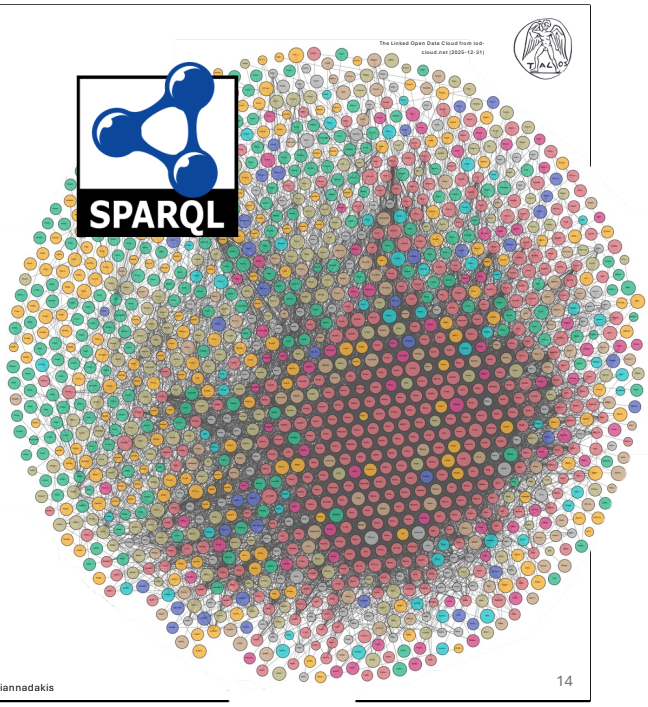
```
PREFIX dbr: <http://dbpedia.org/resource/>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

ASK
where {dbr:Mona_Lisa dbp:artist ?artistURI.
       ?artistURI rdfs:label "Leonardo da Vinci"@en}
```

true

Finally, here's one more query—but this time, we use the **ASK** query form instead of **SELECT**.
As we mentioned earlier, the **ASK** form checks whether a query pattern exists in the graph, returning **true** or **false**.
In this case:
1.We check if there is a resource in DBpedia linked to **Mona_Lisa** via the **artist** relation.
2.And if that resource has an English label equal to **"Leonardo da Vinci"**.
The query returns **true**, confirming that such a relationship exists in the graph under research.

Conclusion

- Powerful tool:
  - Query RDF data
  - Unlock the Semantic Web
- Main uses=> (re)search:
  - Semantics
  - Explore large datasets
  - Data from multiple sources



SPARQL is a powerful tool for querying RDF data and unlocking the full potential of the Semantic Web.
Without it, navigating or making sense of the vast web of linked data would be nearly impossible.
Let's quickly recap the main uses of SPARQL, especially in research and data-driven fields:
•**Semantics**: SPARQL allows us to perform semantic searches—querying not just raw data, but also the relationships and meanings behind it.
•**Explore large datasets**: Whether it's DBpedia, Wikidata, or Europeana, SPARQL is built to handle large, complex datasets efficiently.
•**Integrate data from multiple sources**: SPARQL's strength lies in pulling together data from different RDF datasets, enabling richer, more connected insights.

TALOS ERA CHAIR IN ARTIFICIAL INTELLIGENCE FOR HUMANITIES AND SOCIAL SCIENCES

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ
UNIVERSITY OF CRETE

Thank you!

Rafail Giannadakis
Research Assistant, TALOS-AI*4*SSH
giannadakis.uni@gmail.com

Funded by the European Union

Horizon ERA Chair TALOS AI4SSH Project funded by the European Commission
Grant Agreement n° 101087269, https://cordis.europa.eu/project/id/101087269

Thank you very much for your attention!