Big Data Analytics for Semantic Data
BigSem Tutorial

Module 2: Libraries for Semantic Data Access

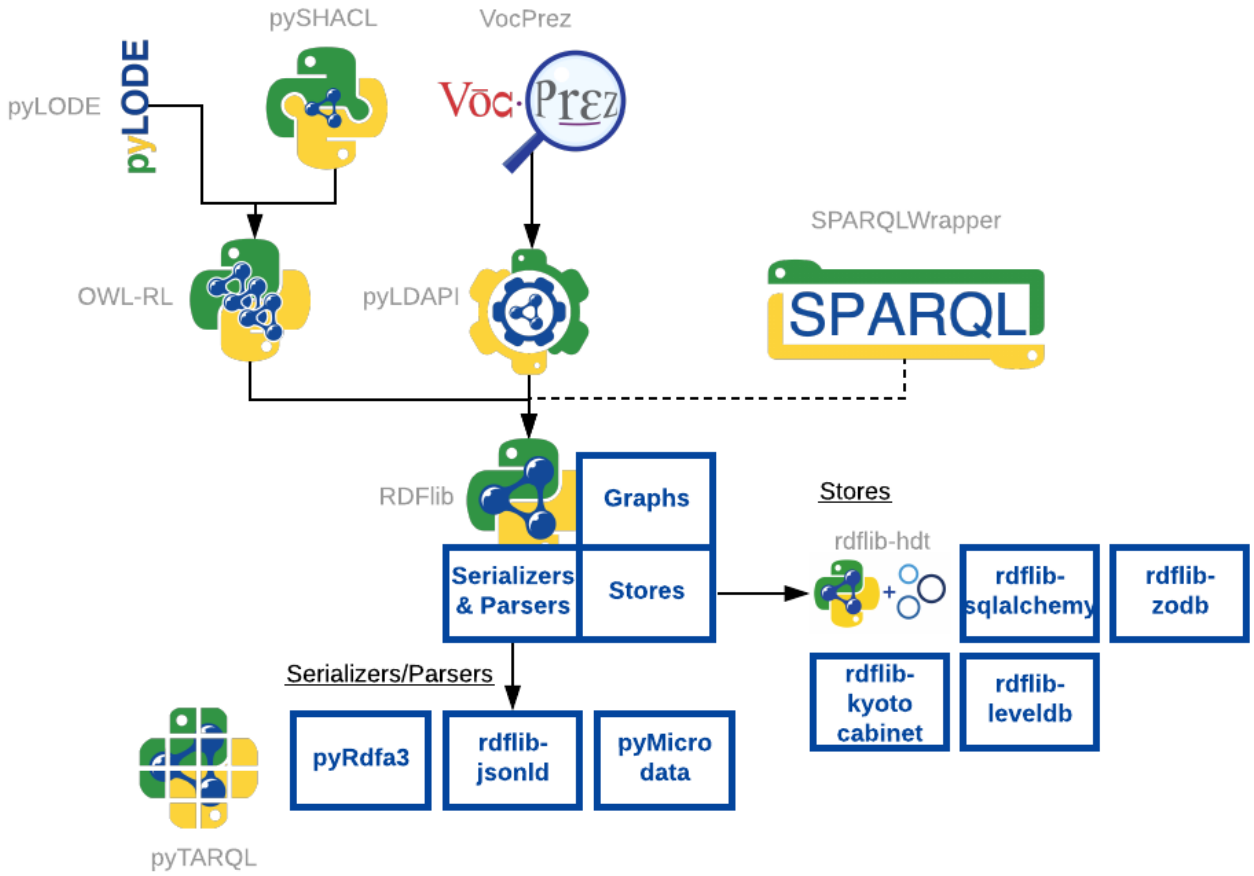
Chelmis Charalampos, Bedirhan Gergin
University at Albany, SUNY

ISWC 2024

goo.gl/6Nrc1r

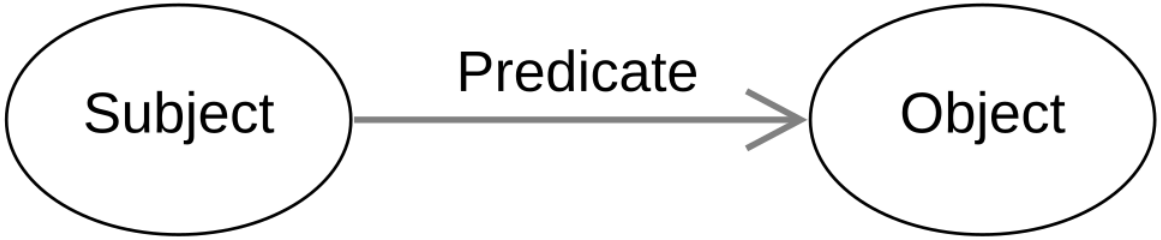
RDFLib: Knowledge Graphs in Python

- A Python library for working with RDF, a data model used for KGs and the semantic web



Big Parenthesis (More on RDF)

- Each triple consists of a subject, predicate, and object, which together represent data in a structured format
  - Subject - The resource being described
  - Predicate - A characteristic or attribute of the subject
  - Object - The value of the attribute or another resource



- Common RDF serialization formats:

  -Turtle - Human-readable and concise

  -RDF/XML - XML-based format

  -JSON-LD - JSON format for RDF data

SPARQL Basics

- Query language for RDF data, allows retrieving and manipulating data in RDF format

- Basic Components of a SPARQL Query:

  SELECT - Retrieves specific data

  WHERE - Specifies patterns to match in the data

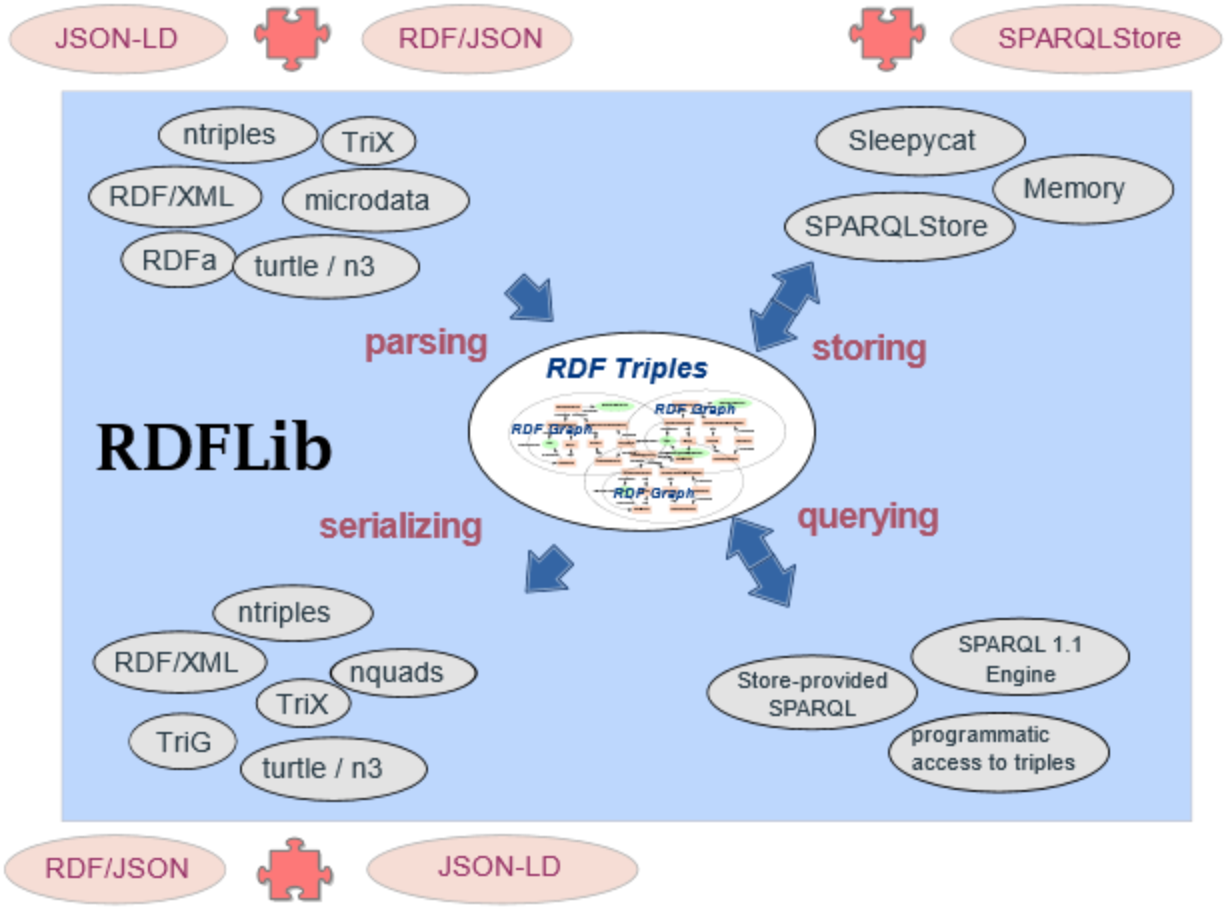
```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT * WHERE {
  ?sub ?pred ?obj .
}
LIMIT 10
```

Endpoints

- Web service that allows users to query an RDF database over the web using SPARQL
- It provides direct access to the data stored in RDF format, enabling querying, filtering, and retrieving specific information

RDFLib

- RDFLib allows:
  - you to parse and serialize RDF data
  - Query data using SPARQL
  - Manipulate RDF graphs programmatically



RDFLib – Creating KG

- An RDF Graph is used to store triples

```
from rdflib import Graph
# Create an empty graph
g = Graph()
print(f'Created an empty graph with {len(g)} triples.')
```

RDFLib – Adding Triples

- We can programmatically add triples to the graph using RDFLib
- Triples are added by defining a subject, predicate, and object, which represent the data

```
from rdflib import URIRef, Literal, Namespace
# Define a namespace for our RDF data
EX = Namespace('http://example.org/')

# Add a few triples to the graph
g.add((URIRef(EX.Alice), URIRef(EX.name), Literal('Alice')))
g.add((URIRef(EX.Bob), URIRef(EX.knows), URIRef(EX.Alice)))

print(f'Graph now contains {len(g)} triples after adding.')
```

RDFLib – Loading and Serializing

- We can load RDF data from an external file or URL into the graph

```
# Parsing an RDF file (assuming we have an RDF file available)
# You can replace 'example.rdf' with a path to your own RDF file.
g.parse('example.rdf')
print(f'Graph has {len(g)} triples after parsing.')
```

- Once we have RDF data in a graph, we can serialize (export) it into various formats such as Turtle, XML, and JSON-LD.

```
# Serializing RDF data to XML format
g.serialize(destination='output.rdf', format='xml')
print('Serialized the RDF graph to XML format and saved it as output.rdf')
```

RDFLib – Querying with SPARQL

- It allows us to query the graph for specific triples based on patterns
- We will query for all the subjects, predicates, and objects in the graph

```
qres = g.query(
    '''
    SELECT ?subject ?predicate ?object
    WHERE {
      ?subject ?predicate ?object.
    }
    '''
)
# Print the results of the query
for row in qres:
    print(f'{row.subject} {row.predicate} {row.object}')
```

RDFLib Notebook

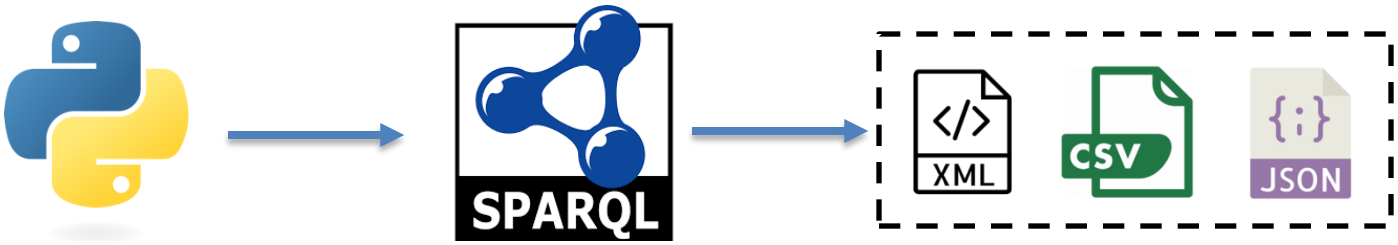
- Feel free to explore more on RDFLib!!

- You can find the hands-on notebook for RDFLib in our Github repository.

SPARQLWrapper: SPARQL Queries in Python

SPARQLWrapper is a Python library to interact with SPARQL endpoints.

- It simplifies querying RDF data using SPARQL from Python
- Commonly used with RDF data sources like DBpedia, Wikidata, or your own RDF stores



SPARQLWrapper in Action

- SparqlWrapper example:

```
# Importing the SPARQLWrapper library
from SPARQLWrapper import SPARQLWrapper, JSON

# Initializing the SPARQLWrapper with the DBpedia endpoint
sparql = SPARQLWrapper("http://dbpedia.org/sparql")
```

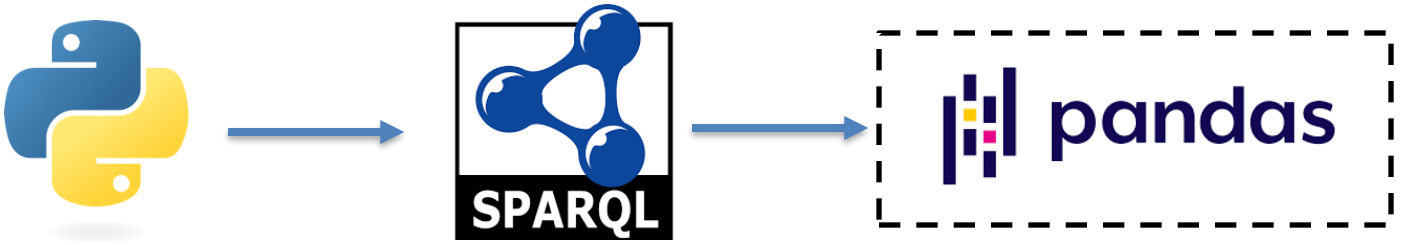
```
# Writing the SPARQL query
sparql.setQuery("""
    SELECT ?label
    WHERE {
      <http://dbpedia.org/resource/Python_(programming_language)> rdfs:label ?label .
      FILTER (lang(?label) = 'en')
    }
""")

# Setting the return format to JSON
sparql.setReturnFormat(JSON)
```

```
# Executing the query and fetching results
results = sparql.query().convert()
```

Sparql-dataframe: SPARQL Results in Pandas

- Sparql-dataframe is a Python library to interact with SPARQL endpoints and specifically returning result as dataframes
    - It simplifies querying RDF data using SPARQL from Python
    - Commonly used with RDF data sources like DBpedia, Wikidata, or your own RDF stores



Sparql-dataframe in Action

- Sparql-dataframe example:

```
# Writing the SPARQL query
query = """
    SELECT ?label ?birthPlace
    WHERE {
      ?person dbo:birthPlace ?birthPlace ;
              rdfs:label ?label .
      FILTER (lang(?label) = 'en')
    }
    LIMIT 10
"""
```

```
# Querying the endpoint and getting the results as a DataFrame
df = sparql_dataframe.get(endpoint_url, query)

# Displaying the resulting DataFrame
df.head()
```

Notebooks

- Feel free to explore more on SPARQLWrapper and Sparql-dataframe!!

- You can find the hands-on notebooks in our Github repository.

Step by Step Instructions

- Let's go to RDFLib tutorial and practice!!

- We will run a couple queries on RecipeKG dataset.

Coffee Break

- See you again at 3:40pm