Training Day 11
Report

27 June 2024

Key Takeways:

1. Uploading Data to Fuseki in CSV Form

- **Convert CSV to RDF**
  - CSV files need to be converted into an RDF format because Fuseki accepts RDF data.
  - Use tools like csv2rdf or a custom script to convert CSV data into RDF.
- **Example Conversion Script Using Python**
  - Use the rdflib library in Python to convert CSV to RDF.
  - Example:

```
python  Copy
code  Import
csv
from rdflib import Graph, Literal, RDF, URIRef, Namespace
from rdflib.namespace import XSD

g = Graph()
ns1 = Namespace("http://example.org/")

with open('data.csv', 'r') as csvfile:
reader = csv.DictReader(csvfile)        for
row in reader:
        person = URIRef(ns1 + row['id'])
        g.add((person, ns1.name, Literal(row['name'])))
        g.add((person, ns1.age, Literal(row['age'],
datatype=XSD.integer)))
g.serialize(destination='out.rdf', format='xml')
```

- **Upload RDF Data to Fuseki**
  - Access the Fuseki web interface.
  - Navigate to the "Add Data" section.
  - Select the RDF file and upload it.
  - Example: out.rdf

2. Performing SPARQL Queries on Uploaded Data

- **Retrieve All Triples**
  - o Query:

```
Sparql
Copy code
 SELECT ?subject ?predicate ?object
 WHERE{
 ?subject ?predicate ?object.
 }
```

  - o **Explanation:** This query retrieves all triples in the dataset.

- **Retrieve Names and Ages**
  - o Query:

```
sparql
Copy code
PREFIX ns1: <http://example.org/>
SELECT ?name ?age
WHERE{
?person ns1:name ?name ;  ns1:age
?age .
}
```

  - o **Explanation:** This query retrieves the name and age for each individual.

- **Filter Individuals by Age**
  - o Query:

```
sparql
Copy code
PREFIX ns1: <http://example.org/>
SELECT ?name
WHERE{
?person ns1:name ?name ;  ns1:age
?age .
FILTER(?age > 5)
}
```

  - o **Explanation:** This query retrieves the names of individuals who are older than 5 years.

- **Count the Number of Individuals**
  - o Query:

```
sparql
Copy code
PREFIX ns1: <http://example.org/>
SELECT (COUNT(?person) AS ?numberOfIndividuals)
WHERE{
?person ns1:name ?name .
}
```

  - o **Explanation:** This query counts the total number of individuals in the dataset.

- **Retrieve Individuals with a Specific Name**
  - o Query:

```
sparql
Copy code
PREFIX ns1: <http://example.org/>
SELECT ?person ?age
WHERE{
?person ns1:name "f" ;  ns1:age
?age .
}
```

  - o **Explanation:** This query retrieves individuals whose name is "f" and their ages.

- **Retrieve Individuals Grouped by Age**
  - o Query:

```
sparql
Copy code
PREFIX ns1: <http://example.org/>
SELECT ?age (COUNT(?person) AS ?count)
WHERE{
?person ns1:age ?age .
}
GROUPBY?age
```

  - o **Explanation:** This query groups individuals by age and counts how many individuals are in each age group.

- **Retrieve Individuals with Names Starting with a Specific Letter**
  - Query:

```
sparql
Copy code
PREFIX ns1: <http://example.org/>
SELECT ?name
WHERE{
?person ns1:name ?name .
FILTER(STRSTARTS(?name, "s"))
}
```

  - **Explanation:** This query retrieves the names of individuals whose names start with the letter "s".

- **Retrieve Individuals with Ages in a Specific Range**
  - Query:

```
sparql
Copy code
PREFIX ns1: <http://example.org/>
SELECT ?name ?age
WHERE{
?person ns1:name ?name ;  ns1:age
?age .
FILTER(?age >= 4 && ?age <= 7)
}
```

  - **Explanation:** This query retrieves the names and ages of individuals whose ages are between 4 and 7, inclusive.

- **Retrieve the Youngest Individual**
  - Query:

```
sparql
Copy code
PREFIX ns1: <http://example.org/>
SELECT ?name ?age
WHERE{
?person ns1:name ?name ;  ns1:age
?age .
}
ORDERBY?age
LIMIT 1
```

- **Explanation:** This query retrieves the name and age of the youngest individual.

- **Retrieve the Oldest Individual**
  - Query:

```
sparql
Copy code
PREFIX ns1: <http://example.org/>
SELECT ?name ?age
WHERE{
?person ns1:name ?name ;  ns1:age
?age .
}
ORDERBYDESC(?age)
LIMIT 1
```

  - **Explanation:** This query retrieves the name and age of the oldest individual.