



Shri Vile Parle Kelavani Mandal's  
**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**  
(Autonomous College Affiliated to the University of Mumbai)  
NAAC Accredited with "A" Grade (CGPA : 3.18)



### DEPARTMENT OF INFORMATION TECHNOLOGY

**COURSE CODE:** DJ19ITL801

**DATE:** 15-02-2024

**COURSE NAME:** Semantic Web Technology Laboratory

**CLASS:** BE IT

**NAME:** Rishikesh Sharma

**SAP ID:** 60003200102

### EXPERIMENT NO. 4

**CO/LO:** Query ontologies using SPARQL

**AIM:** To execute SPARQL Queries on RDF Dataset.

#### DESCRIPTION OF EXPERIMENT:

SPARQL (SPARQL Protocol and RDF Query Language) is the query language we use to shape and return linked data from a triplestore. SPARQL queries contain triple patterns, much like the data itself, which utilise the relationships to quickly navigate any linked data. This language is common for all linked data so queries can traverse across multiple RDF databases at once. The easiest way to start learning SPARQL is by example so let's start with a simple query and build from it. The following query returns every triple in the triplestore:

```
SELECT ?subject ?predicate ?object
WHERE {
  ?subject ?predicate ?object .
}
```

This query selects all triples matching the pattern: ? subject ?predicate ?object which is all triples. The ? indicates a variable so ? example is a variable called "example". These variables match every possible entity, predicate or literal that fit the patterns in the query. In this query we are selecting every variable in the pattern so we can represent that with a \* character. The only difference in results is that this query only returns 200 triples. We usually set limits in queries like these as there can be billions of matching triples! If you want the next batch, you can add OFFSET 200 on the next line. This allows you to batch process results if needed. Obviously to retrieve useful results you will need to fix variables. Fixing the predicate we can get twenty entities with their labels:



Shri Vile Parle Kelavani Mandal's  
**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**  
 (Autonomous College Affiliated to the University of Mumbai)  
 NAAC Accredited with "A" Grade (CGPA : 3.18)



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

```
SELECT *
```

```
WHERE {
```

```
  ?entity rdfs:label ?name .
```

```
}
```

```
LIMIT 20
```

Entities are connected to their human-readable names with the predicate `rdfs:label` so? `entity rdfs:label ?name` matches any triple with the predicate `rdfs:label` and returns `?entity` and `?name`. We have defined a prefix at the top of the query so that we can make the query more readable. Removing the top line and replacing `rdfs:label` with `<http://www.w3.org/2000/01/rdf-schema#label>` will return the same results. In larger queries you will notice the benefits of prefixing as the same prefix can be used multiple times in one query.

## SOFTWARE/TOOLS USED: Graph DB

### OBSERVATION:

#### Query 1

The screenshot shows the GraphDB SPARQL query editor interface. The query is as follows:

```

1 PREFIX walls: <http://wallscope.co.uk/ontology/olympics/>
2 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3 SELECT DISTINCT ?name
4 WHERE {
5   ?instance walls:athlete ?athlete ;
6   walls:medal <http://wallscope.co.uk/resource/olympics/medal/Gold> .
7   ?athlete rdfs:label ?name .
8 }
9

```

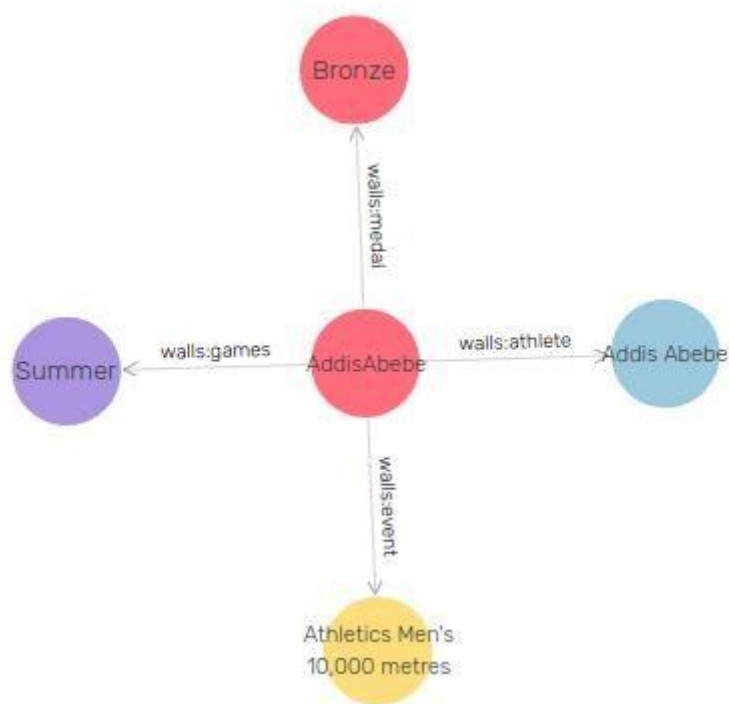
The results are displayed in a table with the following data:

	name
1	"Hermann Schreiber"@en
2	"Franz Pfnr"@en
3	"Henri Jean Oreiller"@en

The interface also shows a sidebar with options like Import, Explore, SPARQL, Monitor, Setup, and Help. The bottom status bar indicates "Showing results from 1 to 1,000 of 10,414. Query took 0.2s, minutes ago."



Shri Vile Parle Kelavani Mandal's  
**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**  
 (Autonomous College Affiliated to the University of Mumbai)  
 NAAC Accredited with "A" Grade (CGPA : 3.18)



## Query 2

The screenshot shows the GraphDB web interface. The left sidebar contains navigation options: Import, Explore, SPARQL (selected), Monitor, Setup, and Help. The main area displays a SPARQL query:

```

1 PREFIX walls: <http://wallscope.co.uk/ontology/olympics/>
2 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3 SELECT ?name (COUNT(?name) As ?noOfMedals)
4 WHERE {
5   ?instance walls:athlete ?athlete ;
6             walls:medal ?medal .
7   ?athlete rdfs:label ?name .
8 }
9 GROUP BY ?name
10 ORDER BY DESC(?noOfMedals)
  
```

Below the query editor, there are tabs for 'Table', 'Raw Response', 'Pivot Table', and 'Google Chart'. The 'Table' tab is active, showing a table with two columns: 'name' and 'noOfMedals'. The table displays the top 3 results of the query.

	name	noOfMedals
1	"Michael Fred Phelps, II"@en	"28"^^xsd:integer
2	"Larysa Semenivna Latynina (Diriy-)"@en	"18"^^xsd:integer
3	"Nikolay Yefimovich Andrianov"@en	"15"^^xsd:integer

At the bottom of the table, it says "Showing results from 1 to 1,000 of 28,203. Query took 0.6s, minutes ago."



### Query 3

SPARQL Query & Update

```

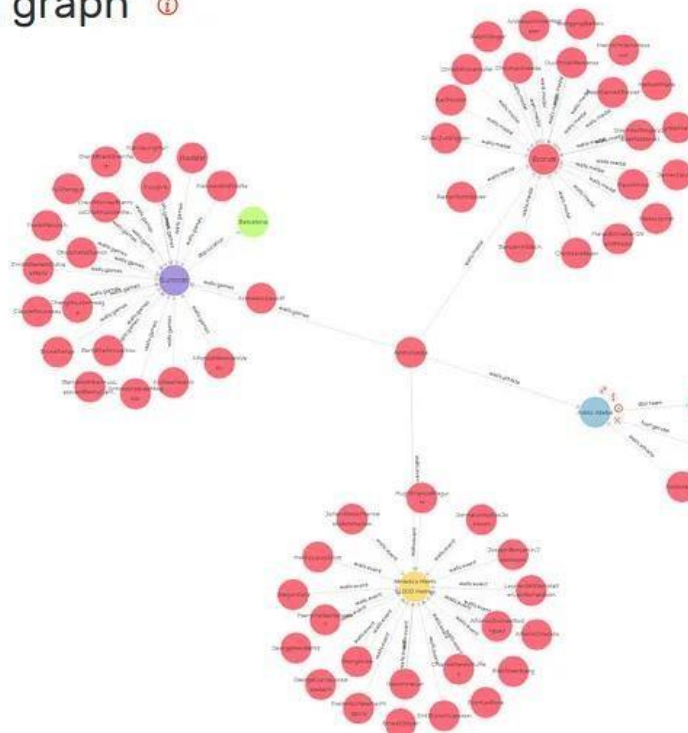
1 PREFIX walls: <http://wallscope.co.uk/ontology/olympics/>
2 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3 PREFIX dbo: <http://dbpedia.org/ontology/>
4 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
5 PREFIX foaf: <http://xmlns.com/foaf/0.1/>
6 SELECT ?countryCode
7       (AVG(?height) As ?avgHeight)
8       (AVG(?weight) As ?avgWeight)
9 WHERE {
10   ?noc dbo:ground ?team ;
11       rdfs:Label ?countryCode .
12   ?athlete rdf:type foaf:Person ;
13           dbo:team ?team .
  
```

Table Raw Response Pivot Table Google Chart

Filter query results Showing results from 1 to 227 of 227. Query took 1.1s, moments ago.

	countryCode	avgHeight	avgWeight
1	"SCG"@en	"188.324607329842931937172775"^^xsd:decimal	"84.094240837696335078534031"^^xsd:decimal
2	"BOH"@en	"188"^^xsd:decimal	"86"^^xsd:decimal
3	"SRB"@en	"186.063559322033898305084746"^^xsd:decimal	"81.669491525423728813559322"^^xsd:decimal
4	"CRO"@en	"185.420654911838790931989924"^^xsd:decimal	"83.44584382871536529294710"^^xsd:decimal
5	"LTU"@en	"184.014388489208633093525180"^^xsd:decimal	"79.600719424460431654676259"^^xsd:decimal

### Visual graph



### Addis Abebe

Addis Abebe<sup>en</sup>

Types:

foaf:Person

RDF rank:

0

dbo:height

160

dbo:weight

50

rdfs:label

Addis Abebe

foaf:age

21



Shri Vile Parle Kelavani Mandal's  
**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**  
 (Autonomous College Affiliated to the University of Mumbai)  
 NAAC Accredited with "A" Grade (CGPA : 3.18)



### Query 4

The screenshot shows the GraphDB SPARQL Query & Update interface. The query editor contains the following SPARQL query:

```

1 PREFIX walls: <http://wallscope.co.uk/ontology/olympics/>
2 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3 PREFIX foaf: <http://xmlns.com/foaf/0.1/>
4 PREFIX dbp: <http://dbpedia.org/property/>
5 SELECT ?year (MAX(?age) AS ?maxAge)
6 WHERE {
7   ?instance walls:games ?games ;
8             walls:event ?event ;
9             walls:athlete ?athlete .
10  ?event rdfs:subClassOf <http://wallscope.co.uk/resource/olympics/sport/Judo> .
11  ?games dbp:year ?year .
12
13
14

```

The results table shows the following data:

	year	maxAge
1	"1980"^^xsd:int	"39"^^xsd:int
2	"1984"^^xsd:int	"39"^^xsd:int
3	"1988"^^xsd:int	"40"^^xsd:int
4	"1992"^^xsd:int	"40"^^xsd:int
5	"1996"^^xsd:int	"40"^^xsd:int

### Query 5

The screenshot shows the GraphDB SPARQL Query & Update interface. The query editor contains the following SPARQL query:

```

1 PREFIX walls: <http://wallscope.co.uk/ontology/olympics/>
2 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3 PREFIX dbo: <http://dbpedia.org/ontology/>
4 PREFIX foaf: <http://xmlns.com/foaf/0.1/>
5 PREFIX dbp: <http://dbpedia.org/property/>
6 PREFIX noc: <http://wallscope.co.uk/resource/olympics/NOC/>
7 SELECT ?genderName (COUNT(?athlete) AS ?count)
8 WHERE {
9   ?instance walls:games ?games ;
10             walls:athlete ?athlete .
11  ?games dbp:location ?city .
12  ?athlete foaf:name ?name .
13
14

```

The results table shows the following data:

	genderName	count
1	"male"^^en	"108372"^^xsd:integer
2	"female"^^en	"55465"^^xsd:integer





Shri Vile Parle Kelavani Mandal's  
**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**  
 (Autonomous College Affiliated to the University of Mumbai)  
 NAAC Accredited with "A" Grade (CGPA : 3.18)



### Query 6

The screenshot shows the GraphDB SPARQL Query & Update interface. The query is as follows:

```

1 PREFIX walls: <http://wallscope.co.uk/ontology/olympics/>
2 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3 SELECT DISTINCT ?name
4 WHERE {
5   ?instance walls:athlete ?athlete ;
6             walls:medal ?medal .
7   ?athlete rdfs:label ?name .
8   FILTER(?medal = <http://wallscope.co.uk/resource/olympics/medal/Gold>)
9 }

```

The results table shows the following data:

	name
1	"Hermann Schreiber"@en
2	"Franz Pfnnr"@en
3	"Henri Jean Oreiller"@en
4	"Hubert Strolz"@en
5	"Josef Polig"@en

### Query 7

The screenshot shows the GraphDB SPARQL Query & Update interface. The query is as follows:

```

1 PREFIX walls: <http://wallscope.co.uk/ontology/olympics/>
2 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3 PREFIX dbp: <http://dbpedia.org/property/>
4 SELECT DISTINCT ?cityName
5 WHERE {
6   ?instance walls:games / dbp:location / rdfs:label ?cityName .
7 }

```

The results table shows the following data:

	cityName
1	"Athina"@en
2	"Paris"@en
3	"St Louis"@en
4	"London"@en
5	"Stockholm"@en
6	"Antwerpen"@en
7	"Chamonix"@en
8	"Amsterdam"@en
9	"Sankt Moritz"@en



Shri Vile Parle Kelavani Mandal's  
**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**  
(Autonomous College Affiliated to the University of Mumbai)  
NAAC Accredited with "A" Grade (CGPA : 3.18)



### **CONCLUSION:**

We have thus used GraphDB to execute SPARQL Queries on RDF Dataset

### **REFERENCES:**

- [1] <https://medium.com/wallscope/constructing-sparql-queries-ca63b8b9ac02>
- [2] <https://www.w3.org/TR/rdf-sparql-query/>