Tutorial on SPARQL: SPARQL Protocol and RDF Query Language

Presented By

Biswanath dutta

DRTC, ISI, Bangalore

Outline

- Introduction to SPARQL
- SPARQL Query Forms
- SELECT Query form
- Simple query
- Multiple match
- Optional Clause
- Filter Clause (string, arithmatic)
- Matching alternatives
- Algebra
- Query for properties and schema
- Queries Manipulation

Introduction

- SPARQL is a query language for RDF graph traversal
  - SPARQL query language specification
- A protocol* for using SPARQL via HTTP
  - SPARQL Protocol for RDF Specification
- SPARQL queries and manipulates RDF graph content on the web or in the RDF store
- SPARQL is independent of any particular serialization format (e.g., RDF/XML, N3, Turtle)
- Inspired by SQL

- SPARQL 1.0: W3C Recommendation (15th January 2008) (https://www.w3.org/TR/rdf-sparql-query/)
- SPARQL 1.1: W3C Recommendation (21st March 2013) (https://www.w3.org/TR/sparql11-query/)

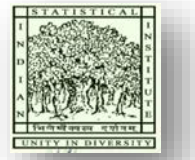
*a means of conveying SPARQL queries from query clients to query processors.

Introduction (contd…2)

- The SPARQL query result is usually displayed in Tabular form.

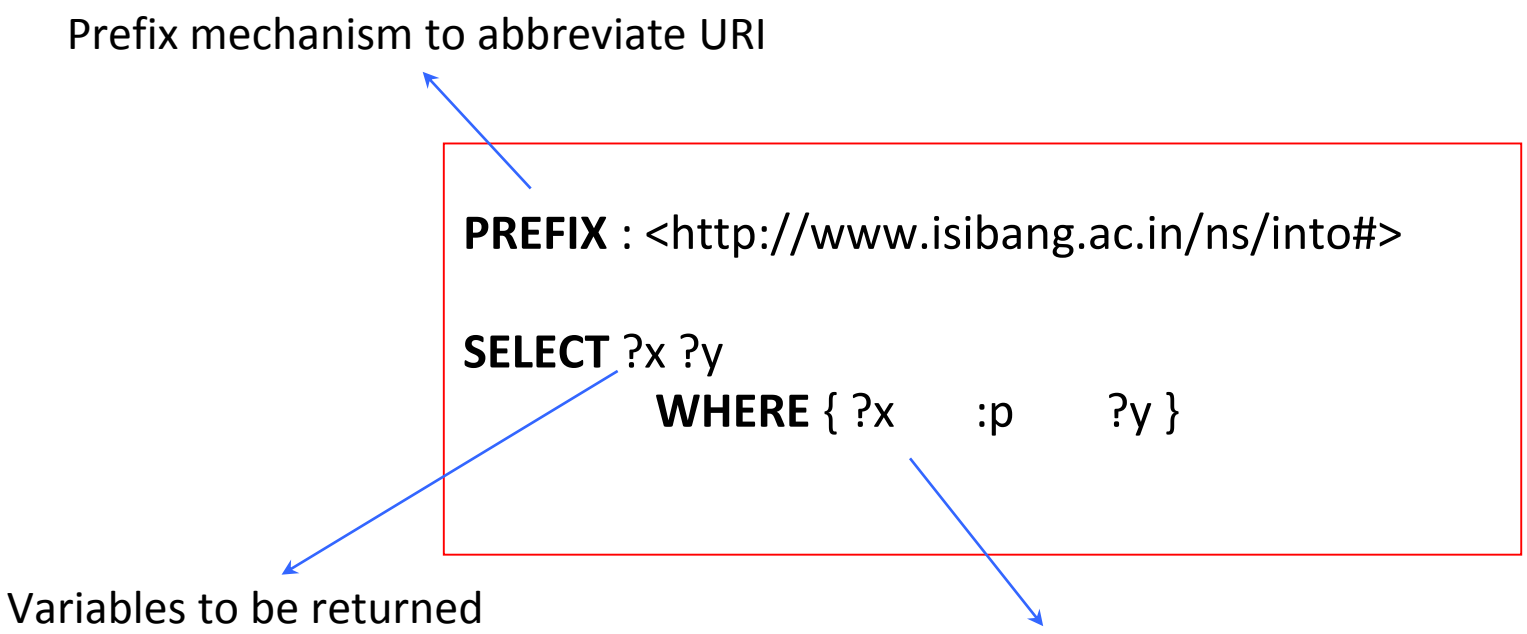
- SPARQL query results can be exchanged using any of the following formats:
  - Extensible Markup Language (XML)
  - JavaScript Object Notation (JSON)
  - Comma Separated Value (CSV)
  - Tab Separated Value (TSV)

SPARQL Four Query Forms



- These query forms use the solutions from pattern matching to form result sets or RDF graphs.
- **SELECT query**
  - Returns all, or a subset of, the variables bound in a query pattern match.
  - Extract raw values, the results are returned in a table format.
- **CONSTRUCT query**
  - Returns an RDF graph constructed by substituting variables in a set of triple templates.
  - Extract information and transform the results into valid RDF.
- **ASK query**
  - Returns a boolean indicating whether a query pattern matches or not.
  - Provides a simple True/False result for a query.
- **DESCRIBE query**
  - Returns an RDF graph that describes the resources found.
  - Extract an RDF graph from the SPARQL endpoint, the content of which is left to the endpoint to decide based on what the maintainer deems as useful information.

Query Form: SELECT



Query Form: SELECT

- Syntax: SELECT var_1, var_2, … var_n
  - Here, var refers to any string variable, such as, ?x, ?name, ?email, ?title, ?experts

SELECT ?x ?y

Alternatively:

SELECT ?researcher ?researchInterest

WHERE

- Graph patterns to match a set of triples
- Syntax: WHERE {

subject predicate object .
subject predicate object .
}

```
PREFIX : <http://www.isibang.ac.in/ns/into#hasResearchInterest#>

SELECT ?x ?y
        WHERE { ?x : hasResearchInterest ?y }
```

Alternative query:

```
SELECT ?researcher ?researchInterest
    WHERE { ?researcher <http://www.isibang.ac.in/ns/into#hasResearchInterest> ?researchInterest}
```

Class inf. of Institutional Ontology (excerpt) (**Turtle** format)

```
###  http://www.isibang.ac.in/ns/into#ADIS

:ADIS rdf:type owl:Class ;

     rdfs:subClassOf :CourseSchedule ;

     rdfs:comment "This course is equivalent to MLISc
degree"^^xsd:string .


###  http://www.isibang.ac.in/ns/into#AcademicProgramme

:AcademicProgramme rdf:type owl:Class .


###  http://www.isibang.ac.in/ns/into#AdministrativeStaff

:AdministrativeStaff rdf:type owl:Class ;

                rdfs:subClassOf :Employee .


###  http://www.isibang.ac.in/ns/into#Agent

:Agent rdf:type owl:Class .
```

```
###  http://www.isibang.ac.in/ns/into#Article

:Article rdf:type owl:Class ;

        rdfs:subClassOf :Publication .


###  http://www.isibang.ac.in/ns/into#Assistant

:Assistant rdf:type owl:Class ;

         rdfs:subClassOf :Employee .


###  http://www.isibang.ac.in/ns/into#AssistantProfessor

:AssistantProfessor rdf:type owl:Class ;

                rdfs:subClassOf :Faculty ,

                                [ rdf:type owl:Restriction ;

                                  owl:onProperty :supervise ;

                                  owl:maxQualifiedCardinality
"4"^^xsd:nonNegativeInteger ;

                                  owl:onClass :ResearchFellow

                                ] .
```

A Simple Query and Result

```
SELECT ?class ?sub_class
        WHERE { ?class rdfs:subClassOf ?sub_class}
```

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX : <http://www.isibang.ac.in/ns/into#>

SELECT ?class ?sub_class
        WHERE { ?class rdfs:subClassOf ?sub_class}
```

class	sub_class
ResearchAssociate	hasResearchInterest some ResearchTopic
ResearchFellow	Student
Faculty	Employee
Proceedings	Publication
ResearchFellow	hasSupervisor min 1 Faculty
Student	Person
AppliedScience	Discipline
Short-termCourse	AcademicProgramme
Professor-in-Charge	AdministrativeStaff
WorkshopArticle	Article
Division	Organization
Diploma	AcademicProgramme

Execute

Data in Institutional Ontology

Data in Turtle (1):

```
###  http://www.isibang.ac.in/ns/into#KiranKanta
:BiswanathDutta rdf:type owl:NamedIndividual ,
                    :SeniorResearchFellow ;
            :hasAffiliation :DRTC ,
                        :ISIBC ;
            :hasGender :male ;
            :hasQualification :BSc ,
                         :MLISc ;
            :hasResearchInterest :Ontology ,
                           :SWS ,
                           :SemanticWebBased-eGovernance ;
            foaf:age 26 ;
            foaf:email “kiran@drtc.isibang.ac.in"^^xsd:string ;
            :firstName “Kiran"^^xsd:string ;
            :homepage
"http://drtc.isibang.ac.in/~kiran"^^xsd:anyURI ;
            :lastName “Kanta"^^xsd:string ;
            :rollNo "RS039"^^xsd:string .

.....
```

Data in Turtle (2):

```
###  http://www.isibang.ac.in/ns/into#DebeshDas
:DebeshDas rdf:type owl:NamedIndividual ,
                    :AssociateProfessor ;
            :editorOfEvents :DL2005 ,
                        :ICSD2007 ;
            :hasAffiliation :DRTC ,
                        :ISIBC ;
            :hasGender :male ;
            :hasQualification :MLISc ,
                         :MPhil ;
            :hasResearchInterest :AI ,
                           :DigitalLibrary ,
                           :NLP ,
                           :Ontology ;
            :supervise :BiswanathDutta ,
                    :NGuha ,
                    :SKSunny ;
            foaf:age "57"^^xsd:int ;
            :firstName "Debesh"^^xsd:string ;
            :hasTeachingExperience 10 ;
            :homepage "http://drtc.isibang.ac.in/~ard"^^xsd:anyURI ;
            :lastName "Das"^^xsd:string ;
            :rollNo "ASSP039"^^xsd:string .
.....
```

A Simple Query and Result

```
PREFIX : <http://www.isibang.ac.in/ns/into#>

SELECT ?researcher ?topic
WHERE {?researcher :hasResearchInterest ?topic}
```

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX : <http://www.isibang.ac.in/ns/into#>

SELECT ?researcher ?topic
WHERE {?researcher :hasResearchInterest ?topic}
```

researcher	topic
JayashankarS	AutomaticClassification
Mathew	AbstractAlgebra
IKRavichandraRao	Bibliometrics
DebeshDas	DigitalLibrary
KSRaghavan	KnowledgeManagement
TSSKRao	Geometry
VimalKumar	DigitalLibrary
DebeshDas	NLP
SumanBera	LinkedData
DebeshDas	Ontology
NGuha	Ontology
NGuha	DigitalLibrary

Execute

Multiple Match

Graph patterns to match a set of triples

Syntax: WHERE {

subject predicate object .

subject predicate object .

}

E.g.,

PREFIX : <http://www.isibang.ac.in/ns/into#>

PREFIX foaf:<http://xmlns.com/foaf/0.1/>

SELECT ?firstname ?lastname ?age

WHERE {?p :firstName ?firstname .

?p :lastName ?lastname .

?p foaf:age ?age .}

LIMIT 10

- The above query involves two triple patterns, each triple ends with a '.'
  - the dot ('.') after the last triple can be omitted.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX : <http://www.isibang.ac.in/ns/into#>
PREFIX foaf:<http://xmlns.com/foaf/0.1/>

SELECT ?firstname ?lastname ?age
WHERE {?p :firstName ?firstname .
       ?p :lastName ?lastname .
       ?p foaf:age ?age .}
```

firstname	lastname	age
"S."^^<http://www.w3.org/2001/XMLSchema#string>	"Jayashankar"^^<http://www.w3.org/2001/XMLSchema#string>	"47"^^<http://www.w3.org/2001/XMLSchema#int>
"Kiran"^^<http://www.w3.org/2001/XMLSchema#string>	"Kanta"^^<http://www.w3.org/2001/XMLSchema#string>	"26"^^<http://www.w3.org/2001/XMLSchema#integer>
"IK Ravichandra"^^<http://www.w3.org/2001/XMLSchema#string>	"Rao"^^<http://www.w3.org/2001/XMLSchema#string>	"65"^^<http://www.w3.org/2001/XMLSchema#int>
"Anand"^^<http://www.w3.org/2001/XMLSchema#string>	"Pandey"^^<http://www.w3.org/2001/XMLSchema#string>	"24"^^<http://www.w3.org/2001/XMLSchema#int>
"KS"^^<http://www.w3.org/2001/XMLSchema#string>	"Raghavan"^^<http://www.w3.org/2001/XMLSchema#string>	"61"^^<http://www.w3.org/2001/XMLSchema#int>
"Mathew"^^<http://www.w3.org/2001/XMLSchema#string>	"R"^^<http://www.w3.org/2001/XMLSchema#string>	"33"^^<http://www.w3.org/2001/XMLSchema#int>
"TSSK"^^<http://www.w3.org/2001/XMLSchema#string>	"Rao"^^<http://www.w3.org/2001/XMLSchema#string>	"63"^^<http://www.w3.org/2001/XMLSchema#int>
"Udaya"^^<http://www.w3.org/2001/XMLSchema#string>	"Varadarajan"^^<http://www.w3.org/2001/XMLSchema#string>	"25"^^<http://www.w3.org/2001/XMLSchema#integer>
"Nayana"^^<http://www.w3.org/2001/XMLSchema#string>	"Guha"^^<http://www.w3.org/2001/XMLSchema#string>	"25"^^<http://www.w3.org/2001/XMLSchema#int>
"Debesh"^^<http://www.w3.org/2001/XMLSchema#string>	"Das"^^<http://www.w3.org/2001/XMLSchema#string>	"57"^^<http://www.w3.org/2001/XMLSchema#int>

Execute

Optional Clause

- SPARQL also allows to define OPTIONAL blocks.
- They offer the ability to query for data but not to fail query when that data does not exist.
- Optional blocks define additional graph patterns that do bind to the graph when they can be matched, but do not cause solutions to be rejected if they are not matched.

Optional Pattern Matching

```
PREFIX : <http://www.isibang.ac.in/ns/into#>
PREFIX foaf:<http://xmlns.com/foaf/0.1/>
SELECT ?firstname ?lastname ?age
WHERE {?p :firstName ?firstname .
       ?p :lastName ?lastname .
       OPTIONAL {?p foaf:age ?age .} }
```

```
SELECT ?firstname ?lastname ?age
WHERE {?p :firstName ?firstname .
       ?p :lastName ?lastname .
       OPTIONAL {?p foaf:age ?age .} }
LIMIT 5
OFFSET 2
```

OFFSET causes the solutions generated to start after the specified number of solutions.

firstname	lastname	age
"Anand"^^<http://www.w3.org/2001/XMLSchema#string>	"Pandey"^^<http://www.w3.org/2001/XMLSchema#string>	"24"^^<http://www.w3.org/2001/XMLSchema#int>
"Radharani"^^<http://www.w3.org/2001/XMLSchema#string>	"Rit"^^<http://www.w3.org/2001/XMLSchema#string>	
"Vimal"^^<http://www.w3.org/2001/XMLSchema#string>	"Kumar"^^<http://www.w3.org/2001/XMLSchema#string>	
"Debesh"^^<http://www.w3.org/2001/XMLSchema#string>	"Das"^^<http://www.w3.org/2001/XMLSchema#string>	"57"^^<http://www.w3.org/2001/XMLSchema#int>
"Sanjeev kumar"^^<http://www.w3.org/2001/XMLSchema#string>	"Sunny"^^<http://www.w3.org/2001/XMLSchema#string>	

```
SELECT ?firstname ?lastname ?age
WHERE {?p :firstName ?firstname .
       ?p :lastName ?lastname .
       OPTIONAL {?p foaf:age ?age .} }
```

firstname	lastname	age
"Mathew"^^<http://www.w3.org/2001/XMLSchema#string>	"R"^^<http://www.w3.org/2001/XMLSchema#string>	"33"^^<http://www.w3.org/2001/XMLSchema#int>
"KS"^^<http://www.w3.org/2001/XMLSchema#string>	"Raghavan"^^<http://www.w3.org/2001/XMLSchema#string>	"61"^^<http://www.w3.org/2001/XMLSchema#int>
"Anand"^^<http://www.w3.org/2001/XMLSchema#string>	"Pandey"^^<http://www.w3.org/2001/XMLSchema#string>	"24"^^<http://www.w3.org/2001/XMLSchema#int>
"Radharani"^^<http://www.w3.org/2001/XMLSchema#string>	"Rit"^^<http://www.w3.org/2001/XMLSchema#string>	
"Vimal"^^<http://www.w3.org/2001/XMLSchema#string>	"Kumar"^^<http://www.w3.org/2001/XMLSchema#string>	
"Debesh"^^<http://www.w3.org/2001/XMLSchema#string>	"Das"^^<http://www.w3.org/2001/XMLSchema#string>	"57"^^<http://www.w3.org/2001/XMLSchema#int>
"Sanjeev kumar"^^<http://www.w3.org/2001/XMLSchema#string>	"Sunny"^^<http://www.w3.org/2001/XMLSchema#string>	
"Udaya"^^<http://www.w3.org/2001/XMLSchema#string>	"Varadarajan"^^<http://www.w3.org/2001/XMLSchema#string>	"25"^^<http://www.w3.org/2001/XMLSchema#integer>
"Nayana"^^<http://www.w3.org/2001/XMLSchema#string>	"Guha"^^<http://www.w3.org/2001/XMLSchema#string>	"25"^^<http://www.w3.org/2001/XMLSchema#int>
"Kiran"^^<http://www.w3.org/2001/XMLSchema#string>	"Kanta"^^<http://www.w3.org/2001/XMLSchema#string>	"26"^^<http://www.w3.org/2001/XMLSchema#integer>
"IK Ravichandra"^^<http://www.w3.org/2001/XMLSchema#string>	"Rao"^^<http://www.w3.org/2001/XMLSchema#string>	"65"^^<http://www.w3.org/2001/XMLSchema#int>
"Pradip"^^<http://www.w3.org/2001/XMLSchema#string>	"Patra"^^<http://www.w3.org/2001/XMLSchema#string>	

Execute

Filter Clause

- SPARQL allows to pose restrictions on the values in query solutions.
- These restrictions are defined in FILTER clauses.
- These clauses are, to some extent, similar to WHERE clause of an SQL query.
- SPARQL filters can be used to restrict:
  - String values (using REGEX function),
  - Numeric values (using <, >, =, <=, >= and != operators).
- SPARQL also provides test functions:
  - BOUND, isURI, isBLANK, isLITERAL

Filter – String matching

SPARQL provides an operation to test strings, based on regular expressions

This allows to ask SQL "LIKE" style tests, although the syntax of the regular expression is different from SQL.

Syntax:

FILTER regex(?x, "pattern" [, "flags"])

*Regex invokes the XPath *fn:matches* function to match text against a regular expression pattern. The regular expression language is defined in XQuery 1.0 and XPath 2.0

**The flags argument is optional. The flag "i" means a case-insensitive pattern match is done.

Filter – String matching (contd…2)

```
SELECT ?name
        WHERE { ?person :firstName ?name .
        FILTER regex(?name, "^k", "i")
       }
```

Any name starts with "k"

```
SELECT ?name
        WHERE { ?person :firstName ?name .
        FILTER regex(?name, "ran", "i")
       }
```

Any name that contains "ran"

Filter - Arithmetic Expression

```
SELECT ?student ?age
WHERE {?student a :SeniorResearchFellow ;
                foaf:age ?age .
                FILTER (?age <= 26) }
```

```
SELECT ?student ?age
WHERE {?student a :SeniorResearchFellow;
               foaf:age ?age .
               FILTER (?age <= 26) }
```

student	age
KiranKanta	"26"^^<http://www.w3.org/2001/XMLSchema#integer>
Udaya	"25"^^<http://www.w3.org/2001/XMLSchema#integer>
NGuha	"25"^^<http://www.w3.org/2001/XMLSchema#int>

```
SELECT ?person ?age
WHERE {?person foaf:age ?age .
               FILTER (?age >= 26) }
```

person	age
JayashankarS	"47"^^<http://www.w3.org/2001/XMLSchema#int>
SSPanza	"62"^^<http://www.w3.org/2001/XMLSchema#int>
KiranKanta	"26"^^<http://www.w3.org/2001/XMLSchema#integer>
IKRavichandraRao	"65"^^<http://www.w3.org/2001/XMLSchema#int>
KSRaghavan	"61"^^<http://www.w3.org/2001/XMLSchema#int>
Mathew	"33"^^<http://www.w3.org/2001/XMLSchema#int>
TSSKRao	"63"^^<http://www.w3.org/2001/XMLSchema#int>
DebeshDas	"57"^^<http://www.w3.org/2001/XMLSchema#int>

Optional + Filter

```
SELECT ?student ?age
WHERE { ?student a :SeniorResearchFellow  .
            OPTIONAL { ?student foaf:age ?age .
                        FILTER (?age >= 26)
                                }
        }
```

```
SELECT ?student ?age
WHERE { ?student a :SeniorResearchFellow  .
            OPTIONAL { ?student foaf:age ?age .
                        FILTER (?age >= 26)
                        }
                }
```

student	age
KiranKanta	"26"^^<http://www.w3.org/2001/XMLSchema#integer>
Udaya	
NGuha	

```
SELECT ?student ?age
WHERE { ?student a :SeniorResearchFellow  .
            OPTIONAL { ?student foaf:age ?age .
                        FILTER (?age != 26)
                        } }
```

```
SELECT ?student ?age
WHERE { ?student a :SeniorResearchFellow  .
            OPTIONAL { ?student foaf:age ?age .
                        FILTER (?age != 26)
                        }
                }
```

student	age
KiranKanta	
Udaya	"25"^^<http://www.w3.org/2001/XMLSchema#integer>
NGuha	"25"^^<http://www.w3.org/2001/XMLSchema#int>

Optional + Filter

```
SELECT ?person ?birthDay
WHERE { ?person  :dateOfBirth  ?birthDay .
        FILTER ( ?birthDay < "2010-11-24T00:00:00"^^xsd:dateTime )  }
```

person	birthDay
VimalKumar	"1984-08-02T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
KSRaghavan	"2007-11-24T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
SKSunny	"1980-12-08T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
Radharani Rit	"2007-11-22T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>

```
SELECT ?givenName ?birthDay
WHERE {
?x foaf:givenName ?givenName .
OPTIONAL {?x :dateOfBirth ?birthDay} }
```

givenName	birthDay
"Debesh Satpathy"^^<http://www.w3.org/2001/XMLSchema#string>	
"Anand K."^^<http://www.w3.org/2001/XMLSchema#string>	
"Koti S."^^<http://www.w3.org/2001/XMLSchema#string>	"2007-11-24T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
"Radha Rani"^^<http://www.w3.org/2001/XMLSchema#string>	"2007-11-22T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
"Sanjeev"^^<http://www.w3.org/2001/XMLSchema#string>	"1980-12-08T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
"Vimal K."^^<http://www.w3.org/2001/XMLSchema#string>	"1984-08-02T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>

```
SELECT ?givenName ?birthDay
WHERE {
?x foaf:givenName ?givenName .
OPTIONAL {?x :dateOfBirth ?birthDay}
     FILTER ( bound(?birthDay) )     }
```

BOUND rturns true if var is bound to a value. Returns false otherwise.

givenName	birthDay
"Koti S."^^<http://www.w3.org/2001/XMLSchema#string>	"2007-11-24T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
"Radha Rani"^^<http://www.w3.org/2001/XMLSchema#string>	"2007-11-22T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
"Sanjeev"^^<http://www.w3.org/2001/XMLSchema#string>	"1980-12-08T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
"Vimal K."^^<http://www.w3.org/2001/XMLSchema#string>	"1984-08-02T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>

Optional + Filter

```
SELECT ?givenName ?birthDay
WHERE {
?x foaf:givenName ?givenName .
OPTIONAL {?x :dateOfBirth ?birthDay}
     FILTER ( bound(?birthDay) )     }
```

BOUND rturns true if var is bound to a value. Returns false otherwise.

givenName	birthDay
"Koti S."^^<http://www.w3.org/2001/XMLSchema#string>	"2007-11-24T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
"Radha Rani"^^<http://www.w3.org/2001/XMLSchema#string>	"2007-11-22T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
"Sanjeev"^^<http://www.w3.org/2001/XMLSchema#string>	"1980-12-08T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>
"Vimal K."^^<http://www.w3.org/2001/XMLSchema#string>	"1984-08-02T00:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>

```
SELECT ?givenName ?birthDay
WHERE {
?x foaf:givenName ?givenName .
OPTIONAL {?x :dateOfBirth ?birthDay}
     FILTER ( !bound(?birthDay) )     }
```

givenName	birthDay
"Debesh Satpathy"^^<http://www.w3.org/2001/XMLSchema#string>	
"Anand K."^^<http://www.w3.org/2001/XMLSchema#string>	

Matching Alternatives: UNION

- SPARQL supports **combining graph patterns so that one of several alternative graph patterns may match**.
- If there is a match of more than one alternatives, all the possible pattern solutions are found.
- Pattern alternatives are syntactically specified with the UNION keyword.

Data in Institutional Ontology

```
:AKPandey :firstName "Anand"^^xsd:string ;
          :lastName "Pandey"^^xsd:string ;
      foaf:familyName "Pandey"^^xsd:string ;
      foaf:givenName "Anand K."^^xsd:string .

:DebeshDas :firstName "Debesh"^^xsd:string ;
           :hasTeachingExperience 10 ;
           :lastName "Das"^^xsd:string ;
        foaf:familyName "Das"^^xsd:string ;
        foaf:givenName "Debesh Satpathy"^^xsd:string.

…….
```

Matching Alternatives: UNION (contd…2)

```
SELECT DISTINCT ?firstName ?lastName ?givenName ?familyName
WHERE {
        {?x :firstName ?firstName; :lastName ?lastName .}
  UNION
        {?y foaf:givenName ?givenName; foaf:familyName ?familyName .}
  }
```

*Here, modifier DISTINCT to filter out the duplicate results.


```
SELECT DISTINCT ?firstName ?lastName ?givenName ?familyName
WHERE {
        {?x :firstName ?firstName; :lastName ?lastName .}
  UNION
        {?y foaf:givenName ?givenName; foaf:familyName ?familyName .}
  }
```


firstName	lastName	givenName	familyName
"Pradip"^^<http://www.w3.org/2001/XMLSchema#string>	"Patra"^^<http://www.w3.org/2001/XMLSchema#string>		
"S."^^<http://www.w3.org/2001/XMLSchema#string>	"Jayashankar"^^<http://www.w3.org/2001/XMLSchema#string		
"Udaya"^^<http://www.w3.org/2001/XMLSchema#string>	"Varadarajan"^^<http://www.w3.org/2001/XMLSchema#string>		
"KS"^^<http://www.w3.org/2001/XMLSchema#string>	"Raghavan"^^<http://www.w3.org/2001/XMLSchema#string>		
"IK Ravichandra"^^<http://www.w3.org/2001/XMLSchema#stri	"Rao"^^<http://www.w3.org/2001/XMLSchema#string>		
"Sanjeev kumar"^^<http://www.w3.org/2001/XMLSchema#strir	"Sunny"^^<http://www.w3.org/2001/XMLSchema#string>		
"Anand"^^<http://www.w3.org/2001/XMLSchema#string>	"Pandey"^^<http://www.w3.org/2001/XMLSchema#string>		
"Nayana"^^<http://www.w3.org/2001/XMLSchema#string>	"Guha"^^<http://www.w3.org/2001/XMLSchema#string>		
"Radharani"^^<http://www.w3.org/2001/XMLSchema#string>	"Rit"^^<http://www.w3.org/2001/XMLSchema#string>		
"TSSK"^^<http://www.w3.org/2001/XMLSchema#string>	"Rao"^^<http://www.w3.org/2001/XMLSchema#string>		
		"Anand K."^^<http://www.w3.org/2001/XMLSchema#string>	"Pandey"^^<http://www.w3.org/2001/XMLSchema#string>
		"Debesh Satpathy"^^<http://www.w3.org/2001/XMLSchema#s	"Das"^^<http://www.w3.org/2001/XMLSchema#string>

Matching Alternatives: UNION (contd…3)

```
SELECT DISTINCT ?firstName
WHERE {
        {?x :firstName ?firstName .}
   UNION
        {?y foaf:givenName ?firstName .}
   } ORDER BY ?firstName
```

firstName
"Anand"^^<http://www.w3.org/2001/XMLSchema#string>
"Anand K."^^<http://www.w3.org/2001/XMLSchema#string>
"Debesh"^^<http://www.w3.org/2001/XMLSchema#string>
"Debesh Satpathy"^^<http://www.w3.org/2001/XMLSchema#string>
"IK Ravichandra"^^<http://www.w3.org/2001/XMLSchema#string>
"KS"^^<http://www.w3.org/2001/XMLSchema#string>
"Kiran"^^<http://www.w3.org/2001/XMLSchema#string>
"Manju"^^<http://www.w3.org/2001/XMLSchema#string>
"Mathew"^^<http://www.w3.org/2001/XMLSchema#string>
"Mithun Raj"^^<http://www.w3.org/2001/XMLSchema#string>
"Nayana"^^<http://www.w3.org/2001/XMLSchema#string>
"Pradip"^^<http://www.w3.org/2001/XMLSchema#string>

Algebra – Count

- Counting the number of research topics of each researchers.
- The following query displays the individual researchers and the number of research topics they are working on. Here, the grouping is by “researhchers.”

SELECT ?researcher (COUNT (?topics) AS ?noOfTopics)

WHERE {?researcher :hasResearchInterest ?topics .}

GROUP BY ?researcher ORDER BY DESC(COUNT(?topics)) LIMIT 10

For Ascending order, use “ASC”

researcher	noOfTopics
DebeshDas	"4"^^<http://www.w3.org/2001/XMLSchema#integer>
KiranKanta	"3"^^<http://www.w3.org/2001/XMLSchema#integer>
IKRavichandraRao	"3"^^<http://www.w3.org/2001/XMLSchema#integer>
NGuha	"3"^^<http://www.w3.org/2001/XMLSchema#integer>
JayashankarS	"2"^^<http://www.w3.org/2001/XMLSchema#integer>
KSRaghavan	"2"^^<http://www.w3.org/2001/XMLSchema#integer>
SumanBera	"2"^^<http://www.w3.org/2001/XMLSchema#integer>
Mathew	"1"^^<http://www.w3.org/2001/XMLSchema#integer>
TSSKRao	"1"^^<http://www.w3.org/2001/XMLSchema#integer>
VimalKumar	"1"^^<http://www.w3.org/2001/XMLSchema#integer>

Algebra – Count (contd…2)

- Counting the number of researchers working in an area.
- The following query displays the research areas/ topics and the total number of researchers working in the area. Here, the grouping is by “researchers.”

SELECT ?topic (COUNT (?researcher) AS ?noOfResearcher)

WHERE {?researcher :hasResearchInterest ?topic .}

GROUP BY ?topic

topic	noOfResearcher
Geometry	"1"^^<http://www.w3.org/2001/XMLSchema#integer>
NLP	"1"^^<http://www.w3.org/2001/XMLSchema#integer>
DigitalLibrary	"4"^^<http://www.w3.org/2001/XMLSchema#integer>
AI	"1"^^<http://www.w3.org/2001/XMLSchema#integer>
Ontology	"5"^^<http://www.w3.org/2001/XMLSchema#integer>
Faceted_ontology	"1"^^<http://www.w3.org/2001/XMLSchema#integer>
KnowledgeManagement	"1"^^<http://www.w3.org/2001/XMLSchema#integer>
QuantitativeAnalysis	"1"^^<http://www.w3.org/2001/XMLSchema#integer>
SemanticWebBased-eGovernance	"1"^^<http://www.w3.org/2001/XMLSchema#integer>
SWS	"1"^^<http://www.w3.org/2001/XMLSchema#integer>
AbstractAlgebra	"1"^^<http://www.w3.org/2001/XMLSchema#integer>
LinkedData	"1"^^<http://www.w3.org/2001/XMLSchema#integer>

Execute

Query for properties and schema

- The following query tells “what are the sorts of things (i.e., metadata) the dataset knows about DebeshDas?” This query is very powerful. Because we may not know what properties are used to describe a resource.

```
Select DISTINCT ?property
Where {:DebeshDas ?property ?value}
```

This query returns the properties along with their corresponding values.

```
SELECT DISTINCT ?property ?value
    WHERE {:DebeshDas ?property ?value}
```

property	value
rdf:type	owl:NamedIndividual
editorOfEvents	DL2005
hasTeachingExperience	"10"^^<http://www.w3.org/2001/XMLSchema#integer>
foaf:homepage	"http://drtc.isibang.ac.in/~ard"^^<http://www.w3.org/2001/XMLSchema#anyURI>
supervise	SKSunny
supervise	NGuha
lastName	"Das"^^<http://www.w3.org/2001/XMLSchema#string>
hasAffiliation	DRTC
hasQualification	MPhil
rdf:type	AssociateProfessor
foaf:familyName	"Das"^^<http://www.w3.org/2001/XMLSchema#string>
editorOfEvents	International Conference on Semantic Web and Digital Libraries

Execute

Query for properties and schema (Contd…2)

- This ability of querying the properties, make it possible to reverse-engineer schema information from the data itself. For instance, we change the query about properties used to describe :DebeshDas to find all properties used to describe any :Faculty.

```
Select DISTINCT ?property
Where {?x a :Faculty .
       ?x ?property ?object .}
```

property
hasEmployment
rdf:type
hasAffiliation
teachesCourse
hasResearchInterest

- Note: If we don't know about the class :Faculty, we can ask about that as well.

```
Select DISTINCT ?class
Where {?class rdfs:subClassOf  :Faculty}
```

class
AssociateProfessor
Lecturer
Professor
AssistantProfessor

Query for properties and schema (Contd…3)

- If we do not know anything about the data at all, we can find the classes used in the data.

```
Select DISTINCT ?class
Where {?x a ?class}
```

class
rdfs:Datatype
owl:NamedIndividual
owl:DatatypeProperty
owl:ObjectProperty
owl:Class
intoSeniorResearchFellow
owl:AnnotationProperty
Master'sDegree
owl:FunctionalProperty
intoCity
intoResearchTopic
intoBMath

- Also, alternatively, find all the properties used anywhere in the data.

```
Select DISTINCT ?property
Where {?x ?property ?y}
```

property
ontologyId
ontologyVersion
foaf:age
rdf:type
hashCode
sourceOntology
intosupervise
foaf:homepage
rdfs:subClassOf
intopostalCode
intoisAffiliationOf
intohasQualification

Query for Data Manipulation: Insert

```
SELECT ?student ?supervisor
WHERE{
?student :hasSupervisor
?supervisor}
```

	student	supervisor
1	into:AKPandey	into:JayashankarS
2	into:Amrita	into:Jaidev
3	into:Arun	into:Jaidev
4	into:KiranKanta	into:DebeshDas
5	into:NGuha	into:DebeshDas
6	into:SKSunny	into:DebeshDas
7	into:Mithun_Raj_M	into:JayashankarS
8	into:Radharani_Rit	into:KSRaghavan

The above query shows the students and their supervisors. We want to add more students and supervisors in our database.

```
INSERT{
into:Anita :hasSupervisor :Jisha
}
WHERE{
?student :hasSupervisor ?supervisor
}
```

Query for Data Manipulation: Insert (contd…2)

```
SELECT ?student ?supervisor
WHERE{
?student :hasSupervisor
?supervisor}
```

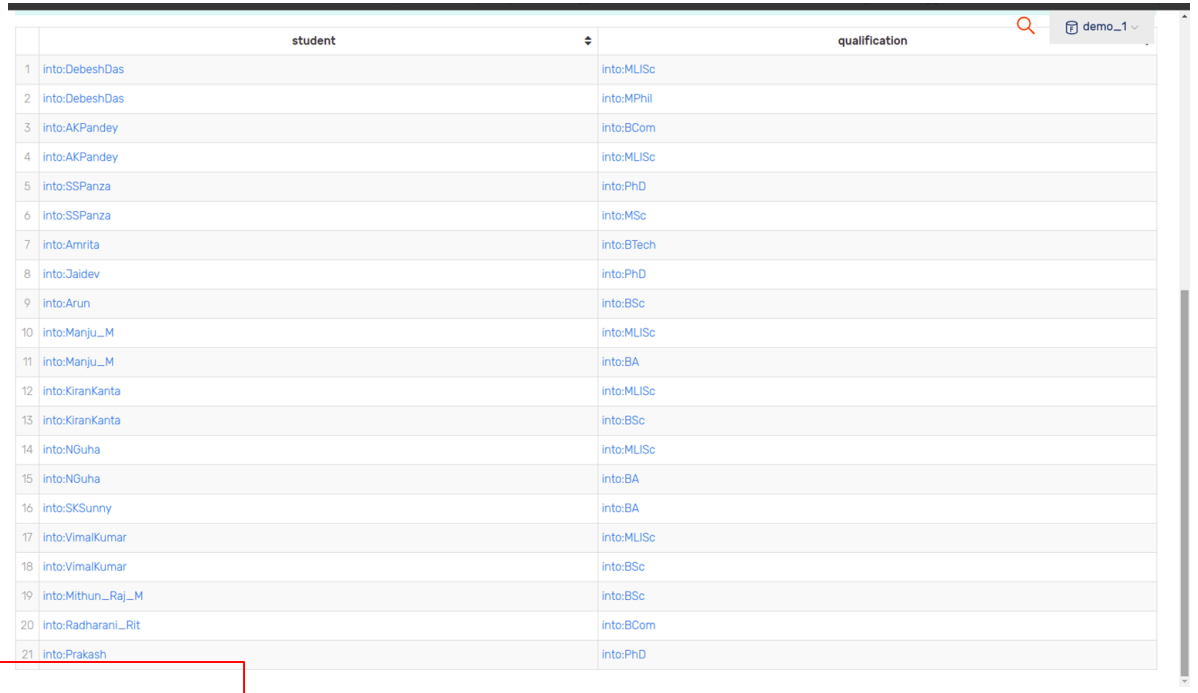
Updated result

	student	supervisor
1	into:AKPandey	into:JayashankarS
2	into:Amrita	into:Jaidev
3	into:Arun	into:Jaidev
4	into:KiranKanta	into:DebeshDas
5	into:NGuha	into:DebeshDas
6	into:SKSunny	into:DebeshDas
7	into:Mithun_Raj_M	into:JayashankarS
8	into:Radharani_Rit	into:KSRaghavan
9	into:Anita	into:Jisha

Query for Data Manipulation: Delete

```
SELECT ?student ?qualification
WHERE{
?student :hasQualification
?qualification
}
```

We see 21 results for the has qualification result.
We need to delete them



demo_1

	student	qualification
1	into:DebeshDas	into:MLISc
2	into:DebeshDas	into:MPhil
3	into:AKPandey	into:BCom
4	into:AKPandey	into:MLISc
5	into:SSPanza	into:PhD
6	into:SSPanza	into:MSc
7	into:Amrita	into:BTech
8	into:Jaidev	into:PhD
9	into:Arun	into:BSc
10	into:Manju_M	into:MLISc
11	into:Manju_M	into:BA
12	into:KiranKanta	into:MLISc
13	into:KiranKanta	into:BSc
14	into:NGuha	into:MLISc
15	into:NGuha	into:BA
16	into:SKSunny	into:BA
17	into:VimalKumar	into:MLISc
18	into:VimalKumar	into:BSc
19	into:Mithun_Raj_M	into:BSc
20	into:Radharani_Rit	into:BCom
21	into:Prakash	into:PhD

```
DELETE {?student :hasQualification :MPhil}
WHERE{
?student :hasQualification ?qualification
}
```

Query for Data Manipulation: Delete (contd…)

```
DELETE {?student :hasQualification
into:MPhil}
WHERE{
?student :hasQualification
?qualification
}
```

Updated result

	student	qualification
1	into:DebeshDas	into:MLISc
2	into:AKPandey	into:BCom
3	into:AKPandey	into:MLISc
4	into:SSPanza	into:PhD
5	into:SSPanza	into:MSc
6	into:Manju_M	into:MLISc
7	into:Manju_M	into:BA
8	into:KiranKanta	into:MLISc
9	into:KiranKanta	into:BSc
10	into:NGuha	into:MLISc
11	into:NGuha	into:BA
12	into:SKSunny	into:BA
13	into:VimalKumar	into:MLISc
14	into:VimalKumar	into:BSc
15	into:Mithun_Raj_M	into:BSc
16	into:Radharani_Rit	into:BCom

Remote query service: search DBPedia

```
PREFIX foaf:<http://xmlns.com/foaf/0.99/>
PREFIX dbo:<http://dbpedia.org/ontology/>
PREFIX :<http://dbpedia.org/resource/>

Select ?person ?institution
Where {
   SERVICE <https://dbpedia.org/sparql/> {
      ?person dbo:birthPlace :Berlin .
      ?person dbo:institution ?institution}
}
LIMIT 100
```

	person	institution
1	http://dbpedia.org/resource/Barbara_A._Schaal	http://dbpedia.org/resource/Washington_University_in_St._Louis
2	http://dbpedia.org/resource/Alex_F._T._W._Rosenberg	http://dbpedia.org/resource/Cornell_University
3	http://dbpedia.org/resource/Alex_F._T._W._Rosenberg	http://dbpedia.org/resource/Northwestern_University
4	http://dbpedia.org/resource/Alex_F._T._W._Rosenberg	http://dbpedia.org/resource/University_of_California,_Santa_Barbara
5	http://dbpedia.org/resource/Alexander_Schmidt_(mathematician)	http://dbpedia.org/resource/Heidelberg_University
6	http://dbpedia.org/resource/Hans_Heilbronn	http://dbpedia.org/resource/University_of_Bristol
7	http://dbpedia.org/resource/Hans_Heilbronn	http://dbpedia.org/resource/University_of_Cambridge
8	http://dbpedia.org/resource/Hans_Heilbronn	http://dbpedia.org/resource/University_of_Toronto
9	http://dbpedia.org/resource/Heinrich_Bruns	http://dbpedia.org/resource/Prussian_Staff_College
10	http://dbpedia.org/resource/Heinrich_Bruns	http://dbpedia.org/resource/Leipzig

References

1. SPARQL 1.1 Query Language (W3C Recommendation). http://www.w3.org/TR/sparql11-query/
2. Corno, Fulvio (2012). SPARQL and Linked Data. http://www.slideshare.net/
3. Ghaiwi, R. (2013). SPARQL.
4. SPARQL by example. https://www.w3.org/2009/Talks/0615-qbe/
5. DBPedia SPARQL Endpoint. http://dbpedia.org/sparql

Thank you!!!

Questions?

Contact:

Biswanath Dutta

dutta2005@gmail.com, bisu@drtc.isibang.ac.in, bisu@isibang.ac.in

Twitter: @biswanath_dutta