XML and Web Technologies

INFO-H509

RDF, RDFS, OWL, and SPARQL

4 June 2021

BAKKALI Yahya : 000445166
HAUWAERT Maxime : 000461714

Université Libre de Bruxelles (ULB)

Contents

1 RDFS and OWL

In order to express the RDF schema for AIDA35K, we used several RDFS and OWL statements.

We set the domain and the range of all properties according to the provided schema. Each property only had one element as domain and one element as range because each property was present in only one location in the schema.

We decided to set the “hasPaper” property as the inverse of the “hasAuthor” property, because if an author has a paper, this paper should have him as an author.

We also decided to set the “hasYear” property as a functional property, because an article should not have multiple years.

2 SPARQL

2.1 Query 1

Return all unique affiliations present in the graph.

```
PREFIX aida35kschema: <http://aida.kmi.open.ac.uk/aida35k/ontology#>
SELECT DISTINCT ?Affiliation
WHERE {
    ?ad aida35kschema:hasAffiliation ?Affiliation.
}
```

We just find all the affiliations, then, we remove the duplicates.

2.2 Query 2

Count the distinct CSO topics present in the graph.

```
PREFIX aida35kschema: <http://aida.kmi.open.ac.uk/aida35k/ontology#>
SELECT (COUNT(DISTINCT ?cso) AS ?NbCsoTopics)
WHERE {
    ?a aida35kschema:hasCsoEnhancedTopic ?cso.
}
```

We just find all the CSO topics, then, we remove the duplicates and finally, we count them.

2.3 Query 3

Find the 10 CSO topics that have the highest number of papers; present the topics in descending order of popularity, and for each topic show its name and number of papers.

```
PREFIX aida35kschema: <http://aida.kmi.open.ac.uk/aida35k/ontology#>
SELECT (?cso as ?CsoTopic) (COUNT(?p) as ?Nb)
WHERE {
  ?p aida35kschema:hasCsoEnhancedTopic ?cso.
} GROUP BY ?cso ORDER BY DESC(?Nb) LIMIT 10
```

First, we find all the CSO topics then we group them topics, and we count the number of papers of each group. Then we sort the results by descending order and set the limit to 10 to get the top 10.

2.4 Query 4

Find papers of the conference series 'iswc', count their references (citations), and present them in decrease order of their count.

```
PREFIX aida35kschema: <http://aida.kmi.open.ac.uk/aida35k/ontology#>
SELECT (?paper as ?Paper) (COUNT(?r) as ?Nb)
WHERE {
    ?paper aida35kschema:hasConfSeries ?s.
    FILTER (?s = "iswc")
    ?paper aida35kschema:hasReference ?r.
} GROUP BY ?paper ORDER BY DESC(?Nb)
```

First, we find all the papers of the conference "iswc" and we get all the references of these papers. Then, we group the results by paper, and we count all their references. Finally, we sort the paper by this count.

2.5 Query 5

Find all authors that have an Affiliation with ULB, and present them along with their papers.

```
PREFIX aida35kschema: <http://aida.kmi.open.ac.uk/aida35k/ontology#>
SELECT (?author as ?Author) (?paper as ?Paper)
WHERE {
    ?aff aida35kschema:hasAffiliation ?name.
    FILTER (?name = "université_libre_de_bruxelles").
    ?paper aida35kschema:hasAffiliationDistribution ?aff.
    ?paper aida35kschema:hasAuthor ?author.
}
```

First we find all the affiliation distributions that have an affiliation with "université_libre_de_bruxelles". Then, we find all the papers that have these affiliation distributions as well as all the authors of these papers.

We have decided to go through the papers to get the affiliation because the query that goes directly from "author" to "affiliationDistribution" via the predicate "hasNetworkInDistribution" yielded no result. We also decided to present the authors with only their papers which have an affiliation with ULB. The output is a set of tuples containing the author and one of his paper for all his papers.

2.6 Query 6

Execute a remote service call to the Microsoft Academic Knowledge Graph SPARQL endpoint and retrieve the abstract for the paper with URI https://makg.org/entity/1642143707.

```
PREFIX dcterms: <http://purl.org/dc/terms/>
SELECT ?abstract
WHERE {
    SERVICE <https://makg.org/sparql>
    {
        SELECT ?abstract
        WHERE {
            <https://makg.org/entity/1642143707> dcterms:abstract ?abstract.
        }
    }
}
```

We just execute a remote call to MAKG and ask for the abstract of the specified URI by using the predicate "dcterms:abstract".

2.7 Query 7

Find all papers that have an Affiliation with ULB, and for each one retrieve its abstract from the remote MAKG SPARQL endpoint.

```
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX aida35kschema: <http://aida.kmi.open.ac.uk/aida35k/ontology#>
SELECT ?abstract
WHERE {
    {  SELECT ?newURI
        WHERE {
            ?aff aida35kschema:hasAffiliation ?name.
            FILTER (?name = "université_libre_de_bruxelles").
            ?paper aida35kschema:hasAffiliationDistribution ?aff.
            BIND (STR(?paper) as ?strURI)
            BIND (REPLACE(STR(?paper), "http://aida.kmi.open.ac.uk/aida35k/resource/p_",
                                "https://makg.org/entity/") as ?strNewURI)
            BIND (URI(?strNewURI) as ?newURI)
        }
    }
    SERVICE <https://makg.org/sparql>
    {
        SELECT ?abstract
        WHERE {
            ?newURI dcterms:abstract ?abstract.
        }
    }
}
```

First, we find the all papers that have an Affiliation with ULB just like the fifth query. Then, we retrieve all their abstracts just like the sixth query.