WEB3.0 - Le langage de requêtes **SPARQL** (recherche de motifs dans un graphe)

ANTELME Mathis

1. Le langage **SPARQL**

À partir du fichier `univ1.rdfs` présent dans l'archive du TP (contenant un schéma et des données), du cours, d'une présentation de SPARQL et de la recommandation W3, écrire les requêtes **SPARQL** permettant d'obtenir les informations suivantes:

> **1.** *Déterminer les personnes qui sont enseignantes. On attend le résultat suivant:*

```
<results>
    <result>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#FredMartin</uri>
        </binding>
    </result>
    <result>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#HerveBlanchon</uri>
        </binding>
    </result>
    <result>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#RobertDupond</uri>
        </binding>
    </result>
</results>
```

La requête:

```
PREFIX ex:  <http://example.org#>

SELECT ?x
WHERE
{
    ?x rdf:type ex:Teacher .
}
```

> *Est-ce que **CORESE** supporte l'inférence ?*

Oui car sinon on n'aurait pas accès à toutes les données que l'on souhaite.

> *Que se passe-t-il lorsque l'on décoche l'option **RDFS** dans le menu engine et que l'on exécute de nouveau la requête ?*

Il nous manque le résultat suivante: `<binding name='x'><uri>http://www.univ-larochelle.fr#FredMartin</uri></binding>`, ce qui prouve bien que **CORESE** supporte l'inférence.

> **2.** *Déterminer les personnes employées par l'université et de la Rochelle et leur statut. On attend le résultat suivant:*

```
<results>
    <result>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#FredMartin</uri>
        </binding>
        <binding name= 'type'>
            <uri>http://example.org#Person</uri>
        </binding>
    </result>
    <result>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#FredMartin</uri>
        </binding>
        <binding name= 'type'>
            <uri>http://example.org#Phd-Student</uri>
        </binding>
    </result>
    <result>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#HerveBlanchon</uri>
        </binding>
        <binding name= 'type'>
            <uri>http://example.org#Person</uri>
        </binding>
    </result>
    <result>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#HerveBlanchon</uri>
        </binding>
        <binding name= 'type'>
            <uri>http://example.org#Teacher</uri>
        </binding>
    </result>
    <result>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#RobertDupond</uri>
        </binding>
        <binding name= 'type'>
            <uri>http://example.org#Person</uri>
        </binding>
    </result>
    <result>
```

```
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#RobertDupond</uri>
        </binding>
        <binding name= 'type'>
            <uri>http://example.org#Teacher</uri>
        </binding>
    </result>
</results>
```

La requête:

```
PREFIX ex:  <http://example.org#>

SELECT ?x ?type
WHERE
{
    ?x rdf:type ?type .
    ?x ex:employed-by <http://www.univ-larochelle.fr> .
}
```

> **3.** *Par rapport à la requête précédente, on souhaiterait récupérer uniquement les résultats suivants (ne pas récupérer les réponses liées au type `Person`):*

```
<results>
    <result>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#FredMartin</uri>
        </binding>
        <binding name= 'type'>
            <uri>http://example.org#Phd-Student</uri>
        </binding>
    </result>
    <result>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#HerveBlanchon</uri>
        </binding>
        <binding name= 'type'>
            <uri>http://example.org#Teacher</uri>
        </binding>
    </result>
    <result>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#RobertDupond</uri>
        </binding>
        <binding name= 'type'>
            <uri>http://example.org#Teacher</uri>
        </binding>
```

```
    </result>
</results>
```

La requête:

```
PREFIX ex:  <http://example.org#>

SELECT ?x ?type
WHERE
{
    ?x rdf:type ?type .
    ?x ex:employed-by <http://www.univ-larochelle.fr>
    FILTER (?type != ex:Person)
}
```

> **4.** *Déterminer les personnes étant en relation avec l'université. Le résultat attendu est le suivant:*

```
<results>
    <result>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#RobertDupond</uri>
        </binding>
        <binding name= 'type'>
            <uri>http://example.org#Teacher</uri>
        </binding>
    </result>
    <result>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#HerveBlanchon</uri>
        </binding>
        <binding name= 'type'>
            <uri>http://example.org#Teacher</uri>
        </binding>
    </result>
    <result>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#HenriDurand</uri>
        </binding>
        <binding name= 'type'>
            <uri>http://example.org#Student</uri>
        </binding>
    </result>
    <result>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#FredMartin</uri>
        </binding>
        <binding name= 'type'>
            <uri>http://example.org#Phd-Student</uri>
```

```
        </binding>
</results>
```

La requête:

```
PREFIX ex:  <http://example.org#>

SELECT DISTINCT ?x ?type
WHERE {
    ?x a     ?type ;
    ?link <http://www.univ-larochelle.fr> .
    filter (?type != ex:Person) .
}
```

> **5.** *Déterminer les personnes ayant comme responsable* `Robert Dupond`*. Le résultat attendu est le suivant:*

```
<results>
    <result>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#FredMartin</uri>
        </binding>
        <binding name= 'x'>
            <uri>http://www.univ-larochelle.fr#HerveBlanchon</uri>
        </binding>
</results>
```

La requête:

```
PREFIX ex:  <http://example.org#>
PREFIX ulr: <http://www.univ-larochelle.fr#>

SELECT ?x
WHERE {
    ulr:RobertDupond  ex:manager-of ?employes.
    ?employes         rdfs:member   ?x
}
```

> **6.** *Construire le graphe* ***RDF*** *suivant à partir d'une requête* ***SPARQL*** *:*

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf= 'http: www.w3.org 1999 02 22-rdf-syntax-ns#' xmlns:ex=
'http: example.org#' xmlns:ulr= 'http: www.univ-larochelle.fr#'>
```

```
    <rdf:Description rdf:about= 'http: www.univ-
larochelle.fr#HerveBlanchon'>
        <ex:is-managed-by rdf:resource= 'http: www.univ-
larochelle.fr#RobertDupond' />
    </rdf:Description>
    <rdf:Description rdf:about= 'http: www.univ-larochelle.fr#FredMartin'>
        <ex:is-managed-by rdf:resource= 'http: www.univ-
larochelle.fr#RobertDupond' />
    </rdf:Description>
</rdf:RDF>
```

La requête:

```
PREFIX ex: <http://example.org#>
PREFIX ulr: <http://www.univ-larochelle.fr#>

CONSTRUCT {  ?employe ex:is-managed-by ?manager . }
WHERE {
    ?manager  ex:manager-of ?employes.
    ?employes rdfs:member   ?employe .
}
```

> **7.** *À partir du schéma `univ2.rdfs` (utilisant cette fois-ci des nœuds anonymes) déterminer le nom des personnes qui sont enseignantes. Si elles possèdent un responsable, le faire apparaître. Le résultat attendu est le suivant:*

```
<results>
    <result>
        <binding name= 'teacher_name'>
            <literal>Herve Blanchon</literal>
        </binding>
        <binding name= 'manager_name'>
            <literal>Robert Dupond</literal>
        </binding>
    </result>
    <result>
        <binding name= 'teacher_name'>
            <literal>Fred Martin</literal>
        </binding>
        <binding name= 'manager_name'>
            <literal>Robert Dupond</literal>
        </binding>
    </result>
    <result>
        <binding name= 'teacher_name'>
            <literal>Robert Dupond</literal>
        </binding>
```

```
    </result>
</results>
```

La requête:

```
PREFIX ex:  <http://example.org#>
PREFIX ulr: <http://www.univ-larochelle.fr#>

SELECT ?teacher_name ?manager_name
WHERE {
    ?teacher ex:employed-by <http://www.univ-larochelle.fr> ;
    ex:name ?teacher_name .
    OPTIONAL {
        ?manager ex:manager-of ?employes ;
        ex:name       ?manager_name .
        ?employes rdfs:member ?teacher
    }
 }
```

> **8.** *Déterminer le nom des personnes qui sont inscrites ou employées à l'université de La Rochelle. En modifiant votre requête, distinguer les noms dans chaque catégorie. Le premier résultat attendu est le suivant:*

```
<results>
    <result>
        <binding name= 'name'>
            <literal>Herve Blanchon</literal>
        </binding>
    </result>
    <result>
        <binding name= 'name'>
            <literal>Fred Martin</literal>
        </binding>
    </result>
    <result>
        <binding name= 'name'>
            <literal>Robert Dupond</literal>
        </binding>
    </result>
    <result>
        <binding name= 'name'>
            <literal>Henri Durand</literal>
        </binding>
    </result>
</results>
```

> *Le second résultat attendu est le suivant:*

```
<results>
    <result>
        <binding name= 'employe_name'>
            <literal>Herve Blanchon</literal>
        </binding>
    </result>
    <result>
        <binding name= 'employe_name'>
            <literal>Fred Martin</literal>
        </binding>
    </result>
    <result>
        <binding name= 'employe_name'>
            <literal>Robert Dupond</literal>
        </binding>
    </result>
    <result>
        <binding name= 'student_name'>
            <literal>Fred Martin</literal>
        </binding>
    </result>
    <result>
        <binding name= 'student_name'>
            <literal>Henri Durand</literal>
        </binding>
    </result>
</results>
```

La requête du premier résultat:

```
PREFIX ex:  <http://example.org#>
SELECT DISTINCT ?name

WHERE {
    {
        [] ex:name ?name ;
        ex:employed-by <http://www.univ-larochelle.fr>
    }
    UNION
    {
        [] ex:name ?name ;
        ex:registered-at <http://www.univ-larochelle.fr>
    }
}
```

La requête du second résultat:

```
PREFIX ex: <http://example.org#>

SELECT DISTINCT ?employe_name ?student_name
WHERE {
    {
        [] ex:name ?employe_name ;
        ex:employed-by <http://www.univ-larochelle.fr>
    }
    UNION {
        [] ex:name ?student_name ;
        ex:registered-at <http://www.univ-larochelle.fr>
    }
}
```

---

2. Utilisation d'une base RDF de taille importante avec **SPARQL**

La plate-forme Jamendo offre différents services de diffusion de musique. Une grande partie de son catalogue est disponible sous forme de triplets RDF (1100000 triplets) sur le site dbtune.org et s'appuie sur plusieurs schémas dont « Music Ontology ». À partir des fichiers `jamendo.rdf` (données) et `musicontology-level1.rdfs` (schéma) présents dans l'archive du TP écrire les requêtes **SPARQL** permettant d'obtenir les informations suivantes en vous limitant à 20 résultats: Avant de débuter les requêtes, regarder le schéma **RDFS**, il est relativement important mais il est structuré: d'abord sont présentées les classes puis les relations entre ces classes.

> **1.** *Donner les noms des artistes musicaux:*

```
PREFIX bbc: <http://purl.org/ontology/mo/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?artist_name
WHERE {
    ?artist a bbc:MusicArtist .
    ?artist foaf:name ?artist_name
}
LIMIT 20
```

On obtient les résultats suivants:

```
Cicada
Hace Soul
vincent j
NoU
Margin of Safety
Bobywan
Les Clip's
Carter Hotel
```

```
La Tumba
King Dubby
vavrek
Suerte
My Name Is Fantastik
KEPAY
t r y ^ d
Buzzworkers
Mr Nuts
Stian
isotrak
433 erOs
```

> **2.** *Pour chaque artiste, donner par ordre alphabétique sur le nom de l'artiste le nom de ses albums;*

```
PREFIX mo: <http://purl.org/ontology/mo/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>

SELECT ?artist_name ?record_name
WHERE {
    ?artist a mo:MusicArtist .
    ?artist foaf:name ?artist_name .
    ?artist foaf:made ?record .
    ?record a mo:Record .
    ?record dc:title ?record_name
}
ORDER BY ?artist_name
LIMIT 20
```

On obtient les résultats suivants:

```
! M U H ? | HUM
# NUTSHELL # | # DRZWI #
#2 Orchestra | Mercutio's Dead
#2 Orchestra | Being Alive Killed The Best In Me
#Blockout | Get High
#Dance 75# | #Dance 75# Volume 1
#Dance 75# | Dakadium
#Dance 75# | #Dance 75 Mixtape#
#Dance 75# | Can You Do Better ?
#NarNaoud# | Green Vision
#ZedMeta# | Petit aperÃ§u
#Zorglups# | First Demo
$ArnoDj13$ | $ArnoDj13$
&amp;ND | Hit the road
(((Niko))) | Geometry Of Art
(own+line) | Down
```

```
* Q u i r y * | FÃc lin Pour L'Autre
* Q u i r y * | Soeurs Siamoises
-;~Â◦Â§)[ k.ROCKSHIRE ](Â§Â◦~;- | the doggystyle EP
-=Kwada=- | DEMO
```

> **3.** *Pour l'artiste 2MaTao, donner le nom de ses albums;*

```
PREFIX mo: <http://purl.org/ontology/mo/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>

SELECT ?artist_name ?record_name
WHERE {
    ?artist a mo:MusicArtist .
    ?artist foaf:name ?artist_name .
    ?artist foaf:made ?record .
    ?record a mo:Record .
    ?record dc:title ?record_name
    FILTER regex(?artist_name,"2MaTao*")
}
LIMIT 50
```

On obtient les résultats suivants:

```
2MaTao | Synthesizer Project
2MaTao | Synthesizer Project vol.2
```

> **4.** *En étudiant la spécification **SPARQL** 1.1, déterminer comment donner, pour chaque artiste, le nombre d'albums qu'il a enregistré;*

```
PREFIX mo: <http://purl.org/ontology/mo/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>

SELECT ?artist_name (COUNT(?record_name) AS ?nb)
WHERE {
    ?artist a mo:MusicArtist .
    ?artist foaf:name ?artist_name .
    ?artist foaf:made ?record .
    ?record a mo:Record .
    ?record dc:title ?record_name
}
GROUP BY ?artist_name
LIMIT 20
```

On obtient les résultats suivants:

```
! M U H ? | 1
# NUTSHELL # | 1
#2 Orchestra | 2
#Blockout | 1
#Dance 75# | 4
#NarNaoud# | 1
#ZedMeta# | 1
#Zorglups# | 1
$ArnoDj13$ | 1
&amp;ND | 1
(((Niko))) | 1
(own+line) | 1
* Q u i r y * | 2
-;~Â◦Â§)[ k.ROCKSHIRE ](Â§Â◦~;- | 1
-=Kwada=- | 1
-DEMO- | 1
-mystery- | 3
...ChArLy's... | 1
...anabase* | 1
...with sad adieus | 1
```

> **5.** *Donner le nom des albums « taggés » acoustique mais pas electro;*

```
PREFIX mo: <http://purl.org/ontology/mo/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX tags: <http://www.holygoat.co.uk/owl/redwood/0.1/tags/>

SELECT ?record_name
WHERE {
    ?record a mo:Record .
    ?record dc:title ?record_name
    ?record tags:taggedWithTag <http://dbtune.org/jamendo/tag/acoustique>
    FILTER NOT EXISTS {  ?record  tags:taggedWithTag
<http://dbtune.org/jamendo/tag/electro> }

}
ORDER BY ?record_name
LIMIT 20
```

On obtient les résultats suivants:

```
" il me manque..."
.::le canzoni di A+ebu)_AYris::.
15 court-mÃc trages
```

```
1Ãc res Prises
2006
3 Obras de Teatro
5 Songs
A Jour
A MIS HERMANOS
A glimpse inside the bubble
A l'aube
A night in Istanbul
A tranparent pain
Accords et A,mes
Acoustic Demo
After Infinity 1
After Infinity III
Amour critique
Andrew, Just
AprÃ¨s la chute
```

> **6.** *Donner le nom de toutes les propriétés associées à la classe MusicArtist;*

```
PREFIX mo: <http://purl.org/ontology/mo/>

SELECT DISTINCT ?label1 ?label2
WHERE {
    {
        ?p a rdf:Property .
        ?p rdfs:label ?label1 .
        ?p rdfs:domain mo:MusicArtist .
    }
    UNION {
        ?p a rdf:Property .
        ?p rdfs:label ?label2 .
        ?p rdfs:range mo:MusicArtist .
    }
}
LIMIT 20
```

On obtient les résultats suivants:

```
biography
compiled
discography
djmixed
fanpage
remixed
sampled
supporting_musician
compiler
```

```
djmixed_by
remixer
sampler
supporting_musician
tribute_to
```

ANTELME Mathis