Interogarea bazelor de cunoștințe în format RDF utilizând SPARQL

Introducere

RDF și OWL permit descrierea formală a unor concepte și a relațiilor dintre acestea. Utilizând aceste limbaje putem specifica structura unui domeniu, proprietățile unui concept astfel încât calculatorul să le poată ”înțelege”. Cunoscând aceste concepte și relațiile dintre ele ne punem problema să regăsim rapid și eficient informația. În acest scop s-au dezvoltat câteva limbaje de interogare, cel mai folosit dintre acestea fiind SPARQL [1]. SPARQL este un limbaj de interogare pentru RDF și se bazează pe creearea unor șabloane ce vor fi căutate în interiorul grafului. RDF este un limbaj care definește grafuri de cunoștințe prin specificarea unor triplete de tipul Subiect Predicat Atribut, unde subiectul și atributul sunt tipic noduri în graful de cunoștințe iar predicatul reprezintă relația dintre ele (muchia din graf).
SPARQL permite definirea de subgrafuri prin specificarea unei succesiuni de triplete și caută încearcă să potrivească subgraful astfel format în graful de cunoștințe inițial

Acest document își propune să vă familiarizeze cu sintaxa SPARQL și să vă permită să construiți interogări simple utilizând acest limbaj.

Exemplele ce vor fi folosite în acest document sunt bazate pe baza de cunoștințe Dbpedia [2], o variantă formală a Wikipedia. Pentru a testa interogările din acest document puteți utiliza interfața web de la http://dbpedia.org/snorql/

Sintaxa

Sintaxa SPARQL este destul de similară cu sintaxa SQL.
O interogare tipică în SPARQL are următoarea structură

```
PREFIX : URI_NAMESPACE_Default
PREFIX prefix_alt_namespace: URI_alt_namespace
Select ?variabila1 ?variabila2
Where
{
?variabila1 predicat prefix_alt_namespace:atribut .
?variabila1 predicat2 ?variabila2
}
Order by ?variabila1
```

Prefix specifică spațiul de nume default pentru interogarea pe care dorim să o realizăm și poate specifica și prefixe pentru alte spații de nume pe care dorim să le utilizăm în interogare. Scopul definirii prefixelor este de a mări lizibilitatea interogării. Dacă nu

dorim să utilizăm prefixe atunci în interiorul clauzei **Where** vom folosi URI-urile integrale ale conceptelor pe care dorim să le utilizăm.
De exemplu, în Dbpedia spațiul de nume pentru proprietăți este http://dbpedia.org/property/ și cel pentru resurse este http://dbpedia.org/resource/.
Utilizând prefixe și considerând spațiul de nume pentru resurse ca fiind default vom putea scrie

```
PREFIX : <http://dbpedia.org/resource/>
PREFIX dbpedia2: <http://dbpedia.org/property/>
SELECT ?name
WHERE
{
:Albert_Einstein dbpedia2:name ?name .
}
```

Albert_Einstein reprezintă resursa din wikipedia și folosind proprietatea nume putem afla efectiv numele persoanei ce este descrisă folosind această resursă.

Interogarea de mai sus poate fi scrisă neutilizând prefixe în felul următor:

```
SELECT ?name
WHERE {
<http://dbpedia.org/resource/Albert_Einstein> <http://dbpedia.org/property/name>
?name .
}
```

Al doilea cuvânt cheie este SELECT. Select precizează ce variabile vor fi întoarse de interogare. Numele de variabile încep întotdeauna cu ”?”. În interiorul unei interogări pot fi folosite variabile ce nu vor fi întoarse și care sunt folosite doar pentru a lega diferite propoziții. În exemplul de mai jos variabila persoană este folosită pentru că reprezintă nodul care leagă mai multe proprietăți din graf. De interes pentru noi este doar numele persoanei pe care îl returnăm utilizând variabila *?name*

```
PREFIX : <http://dbpedia.org/resource/>
PREFIX dbpedia2: <http://dbpedia.org/property/>
PREFIX dbpediaontology: <http://dbpedia.org/ontology/>
SELECT ?name
WHERE {
?person dbpedia2:name ?name .
?person dbpediaontology:knownFor :General_relativity .
?person dbpediaontology:award :Nobel_Prize_in_Physics .
}
```

Clauza Where poate conține 1 sau mai multe triplete ce vor fi folosite în construcția șablonului ce va fi căutat în graful de cunoștințe.
Tripletele sunt întotdeauna de forma Subiect Predicat Atribut. Subiectul și predicatul sunt URI-uri ale unor concepte din baza de cunoștințe. Atributul poate fi un URI al unui concept sau poate fi o valoare. În cazul în care este valoare trebuie precizat tipul și eventual limba. De exemplu în wikipedia după fiecare variabilă de tip string se specifică

și limba folosind marcajul de limbă. Marcajul de limbă se specifică în SPARQL utilizând caracterul ”@” urmat de identificatorul limbii – ”en” de exemplu pentru engleză. Nespecificarea acestui marcaj poate duce la pierderea unor rezultate.
Astfel interogarea următoare nu va produce nici un rezultat

```
PREFIX dbpedia2: <http://dbpedia.org/property/>
SELECT distinct ?person
WHERE {
?person dbpedia2:name "Albert Einstein" .
}
```

Pentru a obține rezultatul dorit trebuie să specificăm marcajul de limbă

```
PREFIX dbpedia2: <http://dbpedia.org/property/>
SELECT distinct ?person
WHERE {
?person dbpedia2:name "Albert Einstein"@en .
}
```

În cadrul clauzei Where putem folosi operatorul FILTER pentru a specifica restricții care să se aplice asupra rezultatelor căutării. Restricțiile pot fi aritmetice, logice sau bazate pe expresii regulate asupra șirurilor de caractere. Tot în interiorul FILTER pot fi utilizate și teste de tip sau de verificare a marcajului de limbă.

Exemplul următor arată cum putem să-l găsim pe Einstein printre cei care au primit premiul Nobel pentru fizică utilizând expresii regulate. Regex primește ca parametri variabila pe care se aplică expresia regulată, expresia propriu-zisă și o listă de flag-uri. Flag-ul ”i” specifică faptul că este o căutare case-insensitive.

```
PREFIX dbpedia2: <http://dbpedia.org/property/>
PREFIX dbpediaontology: <http://dbpedia.org/ontology/>
SELECT distinct ?person
WHERE {
?person dbpedia2:name ?name .
?person dbpediaontology:award :Nobel_Prize_in_Physics .
FILTER regex(?name, "einstein", "i") }
```

Următorul exemplu arată cum putem să obținem folosind Filter un text doar într-o anumită limbă.

```
PREFIX dbpedia2: <http://dbpedia.org/property/>
PREFIX dbpediaontology: <http://dbpedia.org/ontology/>
SELECT distinct ?person ?abstract
WHERE {
?person dbpedia2:name "Albert Einstein"@en .
?person dbpedia2:abstract ?abstract . FILTER langMatches(lang(?abstract),"EN") . }
```

Un ultim exemplu arată cum putem folosi filter pentru a compara date și arată și cum putem specifica, folosind ”^^tipdata”, tipul unei anumite date. Interogarea găsește toți câștigătorii premiului Nobel pentru fizică născuți după 1940.

```
PREFIX dbpediaontology: <http://dbpedia.org/ontology/>
```

SELECT distinct ?person WHERE {
?person dbpediaontology:award :Nobel_Prize_in_Physics .
?person dbpediaontology:birthdate ?birthdate . FILTER (?birthdate>"1940-01-01"^^xsd:date) . }

Concluzii

SPARQL permite regăsirea rapidă a informațiilor formalizate într-o bază de cunoștințe. În contextul în care cât mai multe informații devin accesibile într-un mod formal agenților software un astfel de limbaj le permite acestora să navigheze cu ușurință printre informațiile existente și să regăsească noțiunile de care au nevoie.
Exemplele ce utilizează Dbpedia au scopul de a ilustra faptul că o cantitate impresionantă de cunoștințe ce este deja accesibilă oamenilor (proiectul wikipedia) poate fi acum utilizată și de către calculatoare. Serviciile inteligente care pot fi creeate utilizând aceste cunoștințe vor fi aduce mai aproape ideea web-ului semantic.

Referințe

1. http://www.w3.org/TR/rdf-sparql-query/
2. http://dbpedia.org