

Engenharia de Dados e Conhecimento

2019/2020

A Linguagem SPARQL

Introdução



- SPARQL – *Simple Protocol and RDF Query Language*
- Linguagem padrão pelo W3C para pesquisa de RDF
- Consiste num conjunto de especificações, entre as quais:
 - SPARQL 1.1 Query Language (21 March 2013)
 - <https://www.w3.org/TR/sparql11-query/>
 - SPARQL Query Results XML Format (Second Edition)
 - <https://www.w3.org/TR/rdf-sparql-XMLres/>
 - descreve um formato XML para “serializar” resultados
 - SPARQL 1.1 Graph Store HTTP Protocol
 - <https://www.w3.org/TR/sparql11-http-rdf-update/>
 - descreve um conjunto de operações HTTP para gerir uma coleção de grafos RDF

SPARQL 1.1
Query Language

Searching Data

SPARQL 1.1 Query Language



- Existem 4 formas de pesquisa através de SPARQL
 - SELECT
 - ASK
 - DESCRIBE
 - CONSTRUCT
- A forma mais usada é a SELECT e todas elas são baseadas em padrões de triplos

SPARQL - SELECT



. Estrutura de uma pesquisa SELECT

diretiva "base" (só pode haver uma)

BASE <URI>

lista de prefixos (podem existir múltiplos)

PREFIX pref: <URI>

...

seleção das variáveis resultado

SELECT ...

grafo a pesquisar (opcional)

FROM ...

padrão de pesquisa

WHERE {

...

}

modificadores

ORDER BY ...

SPARQL – Exemplo de Grafo



- . Exemplo de um grafo RDF em N3

```
@prefix fb: <http://rdf.freebase.com/ns/> .

fb:en.hollywood_homicide
    fb:film.film.directed_by fb:en.ron_shelton ;
    fb:film.film.starring fb:en.harrison_ford ,
                          fb:en.kurupt ,
                          fb:en.robert_wagner ;
    fb:film.film.initial_release_date "2003" .

...
```

SPARQL - SELECT



- Exemplo de pesquisa
 - Que realizador aparece no seu próprio filme?

```
PREFIX fb:<http://rdf.freebase.com/ns/>
SELECT ?who ?film
WHERE{
    ?film fb:film.film.directed_by ?who .
    ?film fb:film.film.starring ?who .
}
```

- Resposta:

```
who          film
fb:en.bob_saget fb:en.becoming_dick
```

SPARQL - SELECT



- Pesquisa com união
 - Filmes do “Ron Shelton”, incluindo a data se existir

```
PREFIX fb: <http://rdf.freebase.com/ns/>
SELECT ?film ?reldate
WHERE {
    ?film fb:film.film.directed_by fb:en.ron_shelton .
    OPTIONAL { ?film fb:film.film.initial_release_date ?reldate .}
}
```

- Resposta:

film	reldate
fb:en.dark_blue	
fb:en.hollywood_homicide	2003

SPARQL - SELECT



- Pesquisa com restrição
 - Filmes do “Ron Shelton”, que NÃO têm data

```
PREFIX fb: <http://rdf.freebase.com/ns/>
SELECT ?film ?reldate
WHERE {
    ?film fb:film.film.directed_by fb:en.ron_shelton .
    OPTIONAL { ?film fb:film.film.initial_release_date ?reldate .}
    FILTER (!bound(?reldate))
}
```

- Resposta:

film	reldate
fb:en.dark_blue	

SPARQL - SELECT



- Pesquisa
 - Nomes de atores com nome “russell” case insensitive

```
PREFIX fb: <http://rdf.freebase.com/ns/>
SELECT distinct ?who ?film
WHERE {
    ?film fb:film.film.starring ?star .
    ?star fb:type.object.name ?who .
    FILTER regex(?who, "russell", "i")
}
```

SPARQL - SELECT



- Pesquisa
 - Filmes com data superior a 2002

```
PREFIX fb: <http://rdf.freebase.com/ns/>
SELECT ?film ?when
WHERE {
    ?film fb:film.film.initial_release_date ?when .
    FILTER (?when > "2002")
}
```

- Resposta:

film	when
fb:en.hollywood_homicide	2003
fb:en.body_of_lies	2008

SPARQL - SELECT



- Pesquisa com múltiplos grupos padrão
 - Nomes de realizadores e atores
 - É feita uma conjunção entre os resultados dos grupos

```
PREFIX fb: <http://rdf.freebase.com/ns/>
SELECT ?name
WHERE {
  {
    ?film fb:film.film.directed_by ?person .
    ?person fb:type.object.name ?name
    filter regex(?name, "^... ", "i")
  }
  {
    ?film fb:film.film.starring ?actor .
    ?actor fb:type.object.name ?name
    filter regex(?name, "^b", "i")
  }
}
```

SPARQL - SELECT



- Pesquisa com múltiplos grupos padrão
 - Nomes de realizadores e atores
 - É feita uma união entre os resultados dos grupos

```
PREFIX fb: <http://rdf.freebase.com/ns/>
SELECT ?name
WHERE {
  {
    ?film fb:film.film.directed_by ?person .
    ?person fb:type.object.name ?name
    filter regex(?name, "^... ", "i")
  }
  UNION
  {
    ?film fb:film.film.starring ?actor .
    ?actor fb:type.object.name ?name
    filter regex(?name, "^b", "i")
  }
}
```

SPARQL



. CONSTRUCT

- Esta forma de pesquisa devolve como resultado a criação de novos grafos, em vez do resultado de uma coleção de variáveis.

SPARQL - CONSTRUCT



- Construção de novos triplos

- Partindo da união de resultados de 2 grupos de padrões

```
PREFIX fb: <http://rdf.freebase.com/ns/>
```

```
CONSTRUCT {
```

```
    ?who <http://employment.history/was_employed_in> ?year  
}
```

```
WHERE {
```

```
{
```

```
    ?film fb:film.film.starring ?who .
```

```
    ?film fb:film.film.initial_release_date ?year .
```

```
}
```

```
UNION
```

```
{
```

```
    ?film fb:film.film.directed_by ?who .
```

```
    ?film fb:film.film.initial_release_date ?year .
```

```
}
```

```
}
```

SPARQL



- ASK

- Esta forma de pesquisa verifica a validade de uma determinada declaração.
- Pode incluir variáveis.
- O resultado de uma pesquisa deste tipo é um resultado lógico: “verdadeiro” ou “falso”.

SPARQL - ASK



- Pesquisa

- Pergunta se existe um filme no qual os atores “Bob Saget e Harrison Ford” contracenaram

```
PREFIX fb: <http://rdf.freebase.com/ns/>
```

```
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
```

```
ASK {
```

```
    ?film fb:film.film.starring fb:en.bob_saget .
```

```
    ?film fb:film.film.starring fb:en.harrison_ford .
```

```
}
```

SPARQL



• DESCRIBE

- Esta forma de pesquisa também devolve novos grafos como resultado.
- O resultado pretende ser uma descrição acerca de um recurso do qual se conhece pouco ou nada.
- O resultado depende muito mais do processador que da pesquisa executada.
- Por esta razão, os resultados deste tipo de pesquisa depende muito da implementação dos processadores de SPARQL.

SPARQL - DESCRIBE



• Pesquisa

- Pede uma descrição da entidade dada pelo URI

PREFIX mov:<<http://movies.org/pred/>>

DESCRIBE <http://movies.org/en/bad_taste>

- Resultados

	Subject	Predicate	Object
1	http://movies.org/en/bad_taste	http://movies.org/pred/directed_by	http://movies.org/en/peter_jackson
2	http://movies.org/en/bad_taste	http://movies.org/pred/name	"Bad Taste"^^xsd:string
3	http://movies.org/en/bad_taste	http://movies.org/pred/starring	http://movies.org/en/peter_jackson
4	http://movies.org/en/bad_taste	http://movies.org/pred/starring	http://movies.org/guid/9202a8c04000641f80000000010cf91e
5	http://movies.org/en/bad_taste	http://movies.org/pred/starring	http://movies.org/guid/9202a8c04000641f80000000010cf924
6	http://movies.org/en/bad_taste	http://movies.org/pred/starring	http://movies.org/guid/9202a8c04000641f80000000010cf92b
7	http://movies.org/en/bad_taste	http://movies.org/pred/starring	http://movies.org/guid/9202a8c04000641f800000000434a77d

SPARQL - DESCRIBE



- Pesquisa

- Pede uma descrição de todos os atores que participaram no filme “Blade Runner”

```
PREFIX mov:<http://movies.org/pred/>
```

```
DESCRIBE ?actor
```

```
WHERE{
```

```
    ?film mov:name "Blade Runner" .
```

```
    ?film mov:starring ?actor
```

```
}
```

	Subject	Predicate	Object
1	http://movies.org/authority/imdb/title/tt0060934	http://movies.org/pred/starring	http://movies.org/en/james_hong
2	http://movies.org/authority/imdb/title/tt0060934	http://movies.org/pred/starring	http://movies.org/en/joe_turkel
3	http://movies.org/authority/imdb/title/tt0066601	http://movies.org/pred/starring	http://movies.org/en/harrison_ford
4	http://movies.org/authority/imdb/title/tt0070842	http://movies.org/pred/starring	http://movies.org/en/rutger_hauer

SPARQL 1.1 *Update*

Inserting and Deleting

SPARQL UPDATE - INSERT



- Inserção direta de triplos

```
PREFIX fb: <http://rdf.freebase.com/ns/>
```

```
INSERT DATA
```

```
{  
    fb:Ze_Manel fb:name "José Manuel Gomes" ;  
                fb:nick  "Zé Cabra" .  
}
```

SPARQL UPDATE - INSERT



- Inserção indireta de triplos

```
PREFIX fb: <http://rdf.freebase.com/ns/>
```

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
```

```
INSERT {?s rdf:type fb:celebrities.celebrity}
```

```
WHERE
```

```
{
```

```
  ?s fb:nick "Zé Cabra" .
```

```
}
```

SPARQL UPDATE - DELETE



- Remoção direta de triplos

```
PREFIX fb: <http://rdf.freebase.com/ns/>
```

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
```

```
DELETE DATA
```

```
{  
    fb:Ze_Manel rdf:type fb:celebrities.celebrity ;  
                fb:name "José Manuel Gomes" ;  
                fb:nick "Zé Cabra" .  
}
```


SPARQL UPDATE - DELETE



- Remoção indireta de triplos

```
PREFIX fb: <http://rdf.freebase.com/ns/>
```

```
DELETE { ?s ?p ?o }
```

```
WHERE
```

```
{
```

```
    ?s fb:nick "Zé Cabra" .
```

```
    ?s ?p ?o .
```

```
}
```