Aperçu de SPARQL

Exemples tirés
https://www.w3.org/TR/2013/REC-sparql11-query-20130321/

1 Requête simple

Un premier exemple dont les parties seront détaillées par la suite.

Données

```
@prefix foaf:  <http://xmlns.com/foaf/0.1/> .

_:a  foaf:name   "Johnny Lee Outlaw" .
_:a  foaf:mbox   <mailto:jlow@example.com> .
_:b  foaf:name   "Peter Goodguy" .
_:b  foaf:mbox   <mailto:peter@example.org> .
_:c  foaf:mbox   <mailto:carol@example.org> .
```

Requête

```
PREFIX foaf:   <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox
WHERE
  { ?x foaf:name ?name .
    ?x foaf:mbox ?mbox }
```

Résultat

name	mbox
"Johnny Lee Outlaw"	<mailto:jlow@example.com>
"Peter Goodguy"	<mailto:peter@example.org>

2 Requête simple et variables libres (*blank nodes*)

La clause de restriction (WHERE) dans sa forme la plus simple.

Données

```
@prefix foaf:  <http://xmlns.com/foaf/0.1/> .

_:a  foaf:name   "Alice" .
_:b  foaf:name   "Bob" .
```

Requête

```
PREFIX foaf:   <http://xmlns.com/foaf/0.1/>
SELECT ?x ?name
WHERE  { ?x foaf:name ?name }
```

Résultat 1

x	name
_:c	"Alice"
_:d	"Bob"

Résultat 2

x	name
_:r	"Alice"
_:s	"Bob"

3 Oh triplets de toutes les formes (BASE et PREFIX)

Variantes sur la forme des triplets.

Forme 0

```
SELECT  ?title
WHERE   { <http://example.org/book/book1>  <http://purl.org/dc/elements/1.1/title>  ?title }
```

Forme 1

```
PREFIX  dc: <http://purl.org/dc/elements/1.1/>

SELECT  ?title
WHERE   { <http://example.org/book/book1>  dc:title  ?title }
```

Forme 2

```
PREFIX  dc: <http://purl.org/dc/elements/1.1/>
PREFIX  : <http://example.org/book/>

SELECT  $title
WHERE   { :book1  dc:title  $title }
```

Forme 3

```
BASE    <http://example.org/book/>
PREFIX  dc: <http://purl.org/dc/elements/1.1/>

SELECT  $title
WHERE   { <book1>  dc:title  ?title }
```

Forme ...

D'autres formes existent...

4 Requête avec extension

Clause de restriction (WHERE) et conjonction (avec opérateur `.`)

Données

```
@prefix foaf:  <http://xmlns.com/foaf/0.1/> .

_:a  foaf:givenName  "John" .
_:a  foaf:surname  "Doe" .
```

Requête

```
PREFIX foaf:   <http://xmlns.com/foaf/0.1/>
SELECT ( CONCAT(?G, " ", ?S) AS ?name )
WHERE  { ?P foaf:givenName ?G . ?P foaf:surname ?S }
```

Résultat

name
"John Doe"

Requête équivalente

Factorisation du sujet par le biais de l'opérateur `;` .

```
PREFIX foaf:   <http://xmlns.com/foaf/0.1/>
SELECT ( CONCAT(?G, " ", ?S) AS ?name )
WHERE  { ?P foaf:givenName ?G ; foaf:surname ?S }
```

5 Requête avec restriction sur les attributs (*nodes*)

Utilisation des *filtres* (FILTER).

Données

```
@prefix dc:   <http://purl.org/dc/elements/1.1/> .
@prefix :     <http://example.org/book/> .
@prefix ns:   <http://example.org/ns#> .

:book1  dc:title  "SPARQL Tutorial" .
:book1  ns:price  42 .
:book2  dc:title  "The Semantic Web" .
:book2  ns:price  23 .
```

Requête 1

```
PREFIX  dc:  <http://purl.org/dc/elements/1.1/>
SELECT  ?title
WHERE   { ?x dc:title ?title
          FILTER regex(?title, "^SPARQL")
        }
```

Résultat 1

title
"SPARQL Tutorial"

Requête 2

```
PREFIX  dc:  <http://purl.org/dc/elements/1.1/>
SELECT  ?title
WHERE   { ?x dc:title ?title
          FILTER regex(?title, "web", "i" )
        }
```

Résultat 2

title
"The Semantic Web"

Requête 3

```
PREFIX  dc:  <http://purl.org/dc/elements/1.1/>
PREFIX  ns:  <http://example.org/ns#>
SELECT  ?title ?price
WHERE   { ?x ns:price ?price .
          FILTER (?price < 30.5)
          ?x dc:title ?title . }
```

Résultat 3

title	price
"The Semantic Web"	23

6 Requête avec disjonction (UNION)

Données

```
@prefix dc10:  <http://purl.org/dc/elements/1.0/> .
@prefix dc11:  <http://purl.org/dc/elements/1.1/> .

_:a  dc10:title     "SPARQL Query Language Tutorial" .
_:a  dc10:creator   "Alice" .
_:b  dc11:title     "SPARQL Protocol Tutorial" .
_:b  dc11:creator   "Bob" .
_:c  dc10:title     "SPARQL" .
_:c  dc11:title     "SPARQL (updated)" .
```

Requête

```
PREFIX dc10:  <http://purl.org/dc/elements/1.0/>
PREFIX dc11:  <http://purl.org/dc/elements/1.1/>

SELECT ?title
WHERE  { { ?book dc10:title  ?title } UNION { ?book dc11:title  ?title } }
```

Résultat

title
"SPARQL Protocol Tutorial"
"SPARQL"
"SPARQL (updated)"
"SPARQL Query Language Tutorial"

Requête

```
PREFIX dc10:  <http://purl.org/dc/elements/1.0/>
PREFIX dc11:  <http://purl.org/dc/elements/1.1/>

SELECT ?x ?y
WHERE  { { ?book dc10:title ?x } UNION { ?book dc11:title ?y } }
```

Résultat

x	y
	"SPARQL (updated)"
	"SPARQL Protocol Tutorial"
"SPARQL"	
"SPARQL Query Language Tutorial"	

Requête

```
PREFIX dc10:  <http://purl.org/dc/elements/1.0/>
PREFIX dc11:  <http://purl.org/dc/elements/1.1/>

SELECT ?title ?author
WHERE  { { ?book dc10:title ?title . ?book dc10:creator ?author }
         UNION
         { ?book dc11:title ?title . ?book dc11:creator ?author }
       }
```

Résultat

title	author
"SPARQL Query Language Tutorial"	"Alice"
"SPARQL Protocol Tutorial"	"Bob"

7 Requête avec critère facultatif (OPTIONAL)

Données

```
@prefix foaf:       <http://xmlns.com/foaf/0.1/> .
@prefix rdf:        <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

_:a  rdf:type        foaf:Person .
_:a  foaf:name       "Alice" .
_:a  foaf:mbox       <mailto:alice@example.com> .
_:a  foaf:mbox       <mailto:alice@work.example> .

_:b  rdf:type        foaf:Person .
_:b  foaf:name       "Bob" .
```

Requête

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox
WHERE  { ?x foaf:name  ?name .
         OPTIONAL { ?x  foaf:mbox  ?mbox }
       }
```

Résultat

name	mbox
"Alice"	<mailto:alice@example.com>
"Alice"	<mailto:alice@work.example>
"Bob"	

8 Requête avec emboitement (cas OPTIONAL)

Données

```
@prefix dc:   <http://purl.org/dc/elements/1.1/> .
@prefix :     <http://example.org/book/> .
@prefix ns:   <http://example.org/ns#> .

:book1  dc:title  "SPARQL Tutorial" .
:book1  ns:price  42 .
:book2  dc:title  "The Semantic Web" .
:book2  ns:price  23 .
```

Requête

```
PREFIX  dc:  <http://purl.org/dc/elements/1.1/>
PREFIX  ns:  <http://example.org/ns#>
SELECT  ?title ?price
WHERE   { ?x dc:title ?title .
          OPTIONAL { ?x ns:price ?price . FILTER (?price < 30) }
        }
```

Résultat

title	price
"SPARQL Tutorial"	
"The Semantic Web"	23

9 Requête avec négation (NOT EXISTS)

Données

```
@prefix  :       <http://example/> .
@prefix  rdf:    <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix  foaf:   <http://xmlns.com/foaf/0.1/> .

:alice  rdf:type   foaf:Person .
:alice  foaf:name  "Alice" .
:bob    rdf:type   foaf:Person .
```

Requête

```
PREFIX  rdf:    <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX  foaf:   <http://xmlns.com/foaf/0.1/>

SELECT ?person
WHERE
{
    ?person rdf:type  foaf:Person .
    FILTER NOT EXISTS { ?person foaf:name ?name }
}
```

Résultat

person
<http://example/bob>

10 Requête avec élimination (MINUS)

Données

```
@prefix :       <http://example/> .
@prefix foaf:   <http://xmlns.com/foaf/0.1/> .

:alice  foaf:givenName "Alice" ;
        foaf:familyName "Smith" .
:bob    foaf:givenName "Bob" ;
        foaf:familyName "Jones" .
:carol  foaf:givenName "Carol" ;
        foaf:familyName "Smith" .
```

Requête

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

SELECT DISTINCT ?s
WHERE {
   ?s ?p ?o .
   MINUS {
      ?s foaf:givenName "Bob" .
   }
}
```

Résultat

s
<http://example/carol>
<http://example/alice>

11 Relation entre NOT EXISTS et MINUS

Données

```
@prefix : <http://example/> .
:a :b :c .
```

Requête 1

```
SELECT *
{
  ?s ?p ?o
  FILTER NOT EXISTS { ?x ?y ?z }
}
```

Résultat 1

s	p	o

Requête 2

```
SELECT *
{
   ?s ?p ?o
   MINUS
     { ?x ?y ?z }
}
```

Résultat 2

s	p	o
<http://example/a>	<http://example/b>	<http://example/c>

12 Emboitement des critères (cas FILTER)

Données

```
@prefix : <http://example.com/> .

:a :p 1 .
:a :q 1 .
:a :q 2 .

:b :p 3.0 .
:b :q 4.0 .
:b :q 5.0 .
```

Requête 1

```
PREFIX : <http://example.com/>
SELECT * WHERE {
        ?x :p ?n
        FILTER NOT EXISTS {
                ?x :q ?m .
                FILTER(?n = ?m)
        }
}
```

Résultat

x	n
<http://example.com/b>	3.0

13 Expressions de relation (*path property*)

Syntaxe

Forme syntaxique	Dénomination	Description (en anglais)
iri	PredicatePath	An IRI. A path of length one.
^*elt*	InversePath	Inverse path (object to subject).
elt1 / *elt2*	SequencePath	A sequence path of *elt1* followed by *elt2*.
elt1 | *elt2*	AlternativePath	A alternative path of *elt1* or *elt2* (all possibilities are tried).
*elt**	ZeroOrMorePath	A path that connects the subject and object of the path by zero or more matches of *elt*.
elt+	OneOrMorePath	A path that connects the subject and object of the path by one or more matches of *elt*.
elt?	ZeroOrOnePath	A path that connects the subject and object of the path by zero or one matches of *elt*.
!(*iri*$_1$|…|*iri*$_n$)	NegatedPropertySet	Negated property set. An IRI which is not one of *iri*$_i$. !*iri* is short for !(*iri*).
!(^*iri*$_1$|…|^*iri*$_n$)	NegatedPropertySet	Negated property set where the excluded matches are based on reversed path. That is, not one of *iri*$_1$...*iri*$_n$ as reverse paths. !^*iri* is short for !(^*iri*).
!(*iri*$_1$|…|*iri*$_j$|^*iri*$_{j+1}$|…|^*iri*$_n$)	NegatedPropertySet	A combination of forward and reverse properties in a negated property set.
(*elt*)		A group path *elt*, brackets control precedence.

Priorités

- IRI, identificateur préfixé
- Inversion de propriétés ! et !()
- Parenthèses ()
- Opérateurs unaires *, ?, +
- Opérateur inverse ^
- Opérateur binaire /
- Opérateur binaire |

Alternative

```
{ :book1 dc:title|rdfs:label ?displayString }
```

équivalent à

```
{ :book1
    <http://purl.org/dc/elements/1.1/title> | <http://www.w3.org/2000/01/rdf-schema#label>
    ?displayString }
```

Sequence

```
{ ?x foaf:mbox <mailto:alice@example> . ?x foaf:knows/foaf:name ?name . }
```

Exemple.

```
{ ?x foaf:mbox <mailto:alice@example> . ?x foaf:knows/foaf:knows/foaf:name ?name . }
```

Équivalent à

```
SELECT ?x ?name
{ ?x foaf:mbox <mailto:alice@example> .?x  foaf:knows [ foaf:knows [ foaf:name ?name ]]. }
```

Équivalent à

```
SELECT ?x ?name
{
  ?x  foaf:mbox <mailto:alice@example> .
  ?x  foaf:knows ?a1 .
  ?a1 foaf:knows ?a2 .
  ?a2 foaf:name ?name .
}
```

Filtering duplicates

Because someone Alice knows may well know Alice, the example above may include Alice herself.
This could be avoided with:

```
{ ?x foaf:mbox <mailto:alice@example> .
  ?x foaf:knows/foaf:knows ?y .
  FILTER ( ?x != ?y )
  ?y foaf:name ?name
}
```

Inverse

These two are the same query: the second is just reversing the property direction which swaps the roles of subject and object.

```
{ ?x foaf:mbox <mailto:alice@example> }
{ <mailto:alice@example> ^foaf:mbox ?x }
```

Inverse Path Sequence

Find all the people who know someone `?x` knows.

```
{
  ?x foaf:knows/^foaf:knows ?y .
  FILTER(?x != ?y)
}
```

which is equivalent to (?gen1 is a system generated variable):

```
{
  ?x foaf:knows ?gen1 .
  ?y foaf:knows ?gen1 .
  FILTER(?x != ?y)
}
```

Arbitrary length match

Find the names of all the people that can be reached from Alice by `foaf:knows`:

```
{
  ?x foaf:mbox <mailto:alice@example> .
  ?x foaf:knows+/foaf:name ?name .
}
```

Alternatives in an arbitrary length path:

```
{ ?ancestor (ex:motherOf|ex:fatherOf)+ <#me> }
```

Arbitrary length path match

Some forms of limited inference are possible as well. For example, for RDFS, all types and supertypes of a resource:

```
{ <http://example/thing> rdf:type/rdfs:subClassOf* ?type }
```

All resources and all their inferred types:

```
{ ?x rdf:type/rdfs:subClassOf* ?type }
```

Subproperty

```
{ ?x ?p ?v . ?p rdfs:subPropertyOf* :property }
```

Negated Property Paths

Find nodes connected but not by rdf:type (either way round):

```
{ ?x !(rdf:type|^rdf:type) ?y }
```

14 Spécification des sources (FROM) – cas simple

Données

```
# Default graph (located at http://example.org/foaf/aliceFoaf)
@prefix  foaf:  <http://xmlns.com/foaf/0.1/> .

_:a  foaf:name     "Alice" .
_:a  foaf:mbox     <mailto:alice@work.example> .
```

Requête

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT  ?name
FROM    <http://example.org/foaf/aliceFoaf>
WHERE   { ?x foaf:name ?name }
```

Résultat

name
"Alice"

15 Spécification des sources (FROM) - cas général

Données

```
# Default graph (located at http://example.org/dft.ttl)
@prefix dc: <http://purl.org/dc/elements/1.1/> .
<http://example.org/bob>    dc:publisher  "Bob Hacker" .
<http://example.org/alice>  dc:publisher  "Alice Hacker" .

# Named graph: http://example.org/bob
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
_:a foaf:name "Bob" .
_:a foaf:mbox <mailto:bob@oldcorp.example.org> .

# Named graph: http://example.org/alice
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
_:a foaf:name "Alice" .
_:a foaf:mbox <mailto:alice@work.example.org> .
```

Requête

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
SELECT ?who ?g ?mbox
FROM <http://example.org/dft.ttl>
FROM NAMED <http://example.org/alice>
FROM NAMED <http://example.org/bob>
WHERE
  { ?g dc:publisher ?who . GRAPH ?g { ?x foaf:mbox ?mbox } }
```

Remarques sur les références multiples

- indépendance des sources nommées
- fusion des sources par défaut

16 Manipulation explicite des « graphes »

The following two graphs will be used in examples:

```
# Named graph: http://example.org/foaf/aliceFoaf
@prefix  foaf:     <http://xmlns.com/foaf/0.1/> .
@prefix  rdf:      <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix  rdfs:     <http://www.w3.org/2000/01/rdf-schema#> .

_:a  foaf:name     "Alice" .
_:a  foaf:mbox     <mailto:alice@work.example> .
_:a  foaf:knows    _:b .

_:b  foaf:name     "Bob" .
_:b  foaf:mbox     <mailto:bob@work.example> .
_:b  foaf:nick     "Bobby" .
_:b  rdfs:seeAlso  <http://example.org/foaf/bobFoaf> .

<http://example.org/foaf/bobFoaf>
     rdf:type      foaf:PersonalProfileDocument .

# Named graph: http://example.org/foaf/bobFoaf
@prefix  foaf:     <http://xmlns.com/foaf/0.1/> .
@prefix  rdf:      <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix  rdfs:     <http://www.w3.org/2000/01/rdf-schema#> .

_:z  foaf:mbox     <mailto:bob@work.example> .
_:z  rdfs:seeAlso  <http://example.org/foaf/bobFoaf> .
_:z  foaf:nick     "Robert" .

<http://example.org/foaf/bobFoaf>
     rdf:type      foaf:PersonalProfileDocument .
```

17 Accessing Graph Names

The query below matches the graph pattern against each of the named graphs in the dataset and forms solutions which have the `src` variable bound to IRIs of the graph being matched. The graph pattern is matched with the active graph being each of the named graphs in the dataset.

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

SELECT ?src ?bobNick
FROM NAMED <http://example.org/foaf/aliceFoaf>
FROM NAMED <http://example.org/foaf/bobFoaf>
WHERE
  {
    GRAPH ?src
    { ?x foaf:mbox <mailto:bob@work.example> .
      ?x foaf:nick ?bobNick
    }
  }
```

The query result gives the name of the graphs where the information was found and the value for Bob's nick:

src	bobNick
<http://example.org/foaf/aliceFoaf>	"Bobby"
<http://example.org/foaf/bobFoaf>	"Robert"

18 Restricting by Graph IRI

The query can restrict the matching applied to a specific graph by supplying the graph IRI. This sets the active graph to the graph named by the IRI. This query looks for Bob's nick as given in the graph `http://example.org/foaf/bobFoaf`.

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX data: <http://example.org/foaf/>

SELECT ?nick
FROM NAMED <http://example.org/foaf/aliceFoaf>
FROM NAMED <http://example.org/foaf/bobFoaf>
WHERE
  {
     GRAPH data:bobFoaf {
         ?x foaf:mbox <mailto:bob@work.example> .
         ?x foaf:nick ?nick }
  }
```

which yields a single solution:

nick
"Robert"

19 GROUP, HAVING, ORDER et les autres

- Order modifier: put the solutions in order
- Projection modifier: choose certain variables
- Distinct modifier: ensure solutions in the sequence are unique
- Reduced modifier: permit elimination of some non-distinct solutions
- Offset modifier: control where the solutions start from in the overall sequence of solutions
- Limit modifier: restrict the number of solutions

20 Les trois valeurs logiques (sic)

Table de calcul de la disjonction et de la conjonction pour les valeurs « vrai » (T), « faux » (F) et « erreur » (E) :

A	B	A || B	A && B
T	T	T	T
T	F	T	F
F	T	T	F
F	F	F	F
T	E	T	E
E	T	T	E
F	E	E	F
E	F	E	F
E	E	E	E

Quelques fonctions logiques parmi d'autres :

- bound
- isBlank
- if
- coalesce
- exists, not exists
- || (disjonction)
- && (conjonction)
- = (équivalence, pas l'égalité)
- sameTerm
- in, not in
- isIRI, isLiteral, isNumeric, ...

21 Et si on parlait de syntaxe ?

Consulter la section 19.8 de https://www.w3.org/TR/2013/REC-sparql11-query-20130321/

Syntaxe

Règles 1 à 138

Lexique

Règles 139 à 173