SPARQL again

Example 12 (Matching RDF Literals):

Data

```
@prefix dt:   <http://example.org/datatype#> .
@prefix ns:   <http://example.org/ns#> .
@prefix :     <http://example.org/ns#>.
@prefix xsd:  <http://www.w3.org/2001/XMLSchema#> .

:x   ns:p     "cat"@en .
:y   ns:p     "42"^^xsd:integer .
:z   ns:p     "abc"^^dt:specialDatatype .
```

Queries

```
SELECT ?v WHERE { ?v ?p "cat" }

SELECT ?v WHERE { ?v ?p "cat"@en }

SELECT ?v WHERE { ?v ?p 42 }

SELECT ?v WHERE { ?v ?p "abc"^^<http://example.org/datatype#specialDatatype> }
```


Result

```
v
-
```

Result

```
        v
<http://example.org/ns#x>
```

Result

```
        v
<http://example.org/ns#y>
```

Result

```
        v
<http://example.org/ns#z>
```


Example 13 (Syntax for Triple Patterns):

version 1

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

version 2

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

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

version 3

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

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


Example 14 (Predicate-Object Lists):

version 1

```
?x  foaf:name  ?name .
?x  foaf:nick  "Alice" .
?x  foaf:nick  "Alice_" .
```

version 2

```
 ?x  foaf:name ?name ; foaf:nick  "Alice" , "Alice_" .
```

Example 15a (RDF Collections):

version 1

```
_:b0  rdf:first  1 ;
      rdf:rest   _:b1 .
_:b1  rdf:first  ?x ;
      rdf:rest   _:b2 .
_:b2  rdf:first  3 ;
      rdf:rest   _:b3 .
_:b3  rdf:first  4 ;
      rdf:rest   rdf:nil .
_:b0  :p         "w" .
```

version 2

```
(1 ?x 3 4) :p "w" .
```


Example 15b (RDF Collections):

version 1

```
_:b0  rdf:first  1 ;
      rdf:rest   _:b1 .
_:b1  rdf:first  _:b2 .
_:b2  :p         :q .
_:b1  rdf:rest   _:b3 .
_:b3  rdf:first  _:b4 .
_:b4  rdf:first  2 ;
      rdf:rest   rdf:nil .
_:b3  rdf:rest   rdf:nil .
```

version 2

```
(1 [:p :q] ( 2 ) ) .
```


RDF Dataset

```
RDF data stores may hold multiple RDF graphs and record
information about each graph.

An RDF Dataset comprises one graph, the default graph, which does
not have a name, and zero or more named graphs, where each named
graph is identified by an IRI.

 Default graph: FROM <URI_1> ...  FROM <URI_n>
 Named graph:   FROM NAMED <URI_1> ...  FROM NAMED <URI_n>

Default graphs from different {\tt FROM}-clauses are merged (i.e.
union after having renamed multiply used blank nodes in different
graphs to make them unique) to form one default graph.

The graph that is used for matching a basic graph pattern is the
active graph indicated by the GRAPH keyword.
```


Example 16a (Querying a Data Set):

Data

```
# Default graph (stored 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> .
```

Query

```
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 }
}
```


Result

```
   who               g                          mbox
Bob Hacker   <http://example.org/bob> <mailto:bob@oldcorp.example.org>
Alice Hacker <http://example.org/Alice> <mailto:alice@work.example.org>
```


Example 16b (Querying a Data Set):

Data

```
# 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 .
```


Query

```
PREFIX  data:  <http://example.org/foaf/>
PREFIX  foaf:  <http://xmlns.com/foaf/0.1/>
PREFIX  rdfs:   <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?mbox ?nick ?ppd
FROM NAMED <http://example.org/foaf/aliceFoaf>
FROM NAMED <http://example.org/foaf/bobFoaf>
WHERE
{
 GRAPH data:aliceFoaf
 {
   ?alice foaf:mbox <mailto:alice@work.example> ;
          foaf:knows ?whom .
   ?whom  foaf:mbox ?mbox ;
          rdfs:seeAlso ?ppd .
   ?ppd  a foaf:PersonalProfileDocument .
 } .
 GRAPH ?ppd
 {
     ?w foaf:mbox ?mbox ;
        foaf:nick ?nick
 }
}
```

Result

```
        mbox                    nick                  ppd
<mailto:bob@work.example>  "Robert"    <http://example.org/foaf/bobFoaf>
```


Example 16c (Querying a Data Set):

Data

```
# Default graph @prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix g:  <tag:example.org,2005-06-06:> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
g:graph1 dc:publisher "Bob" .
g:graph1 dc:date "2004-12-06"^^xsd:date .
g:graph2 dc:publisher "Bob" .
g:graph2 dc:date "2005-01-10"^^xsd:date .

# Graph: locally allocated IRI: tag:example.org,2005-06-06:graph1
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
_:a foaf:name "Alice" .
_:a foaf:mbox <mailto:alice@work.example> .
_:b foaf:name "Bob" .
_:b foaf:mbox <mailto:bob@oldcorp.example.org> .

# Graph: locally allocated IRI: tag:example.org,2005-06-06:graph2
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
_:a foaf:name "Alice" .
_:a foaf:mbox <mailto:alice@work.example>.
_:b foaf:name "Bob" .
_:b foaf:mbox <mailto:bob@newcorp.example.org> .
```


Query

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

SELECT ?name ?mbox ?date
WHERE
 {  ?g dc:publisher ?name ;
       dc:date ?date .
   GRAPH ?g
     { ?person foaf:name ?name ; foaf:mbox ?mbox }
 }
```


Result

```
  name            mbox                        date
"Bob"   <mailto:bob@oldcorp.example.org> "2004-12-06"^^xsd:date
"Bob"   <mailto:bob@newcorp.example.org> "2005-01-10"^^xsd:date
```

Solutions

Query patterns generate an unordered collection of solutions, each solution being a partial function from variables to RDF terms.

These solutions are then treated as a sequence (a solution sequence), initially in no specific order; any sequence modifiers are then applied to create another sequence.

Solution Sequence Modifier

* Order modifier: put the solutions in order, expl. ORDER BY ?x DESC(?y)

* Projection modifier: choose certain variables, SELECT

* Distinct modifier: ensure solutions in the sequence are unique, SELECT DISTINCT

* Reduced modifier: permit elimination of some non-unique solutions, SELECT REDUCED

* Offset modifier: control where the solutions start from in the overall sequence of solutions, expl. OFFSET 10

* Limit modifier: restrict the number of solutions, expl. LIMIT 5

Query Forms

SPARQL has four query forms. These query forms use the solutions from pattern matching to form result sets or RDF graphs. The query forms are:

SELECT
: Returns all, or a subset of, the variables bound in a query pattern match.

CONSTRUCT
: Returns an RDF graph constructed by substituting variables in a set of triple templates.

ASK
: Returns a boolean indicating whether a query pattern matches or not.

DESCRIBE
: Returns an RDF graph that describes the resources found.

Example 17 (CONSTRUCT: Templates with Blank Nodes):

Data

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

_:a    foaf:givenname   "Alice" .
_:a    foaf:family_name "Hacker".
_:b    foaf:firstname   "Bob" .
_:b    foaf:surname     "Hacker" .
```

Query

```
PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
PREFIX vcard:   <http://www.w3.org/2001/vcard-rdf/3.0#>

CONSTRUCT { ?x  vcard:N _:v .
            _:v vcard:givenName ?gname .
            _:v vcard:familyName ?fname }
WHERE
{
   { ?x foaf:firstname ?gname } UNION  { ?x foaf:givenname   ?gname } .
   { ?x foaf:surname   ?fname } UNION  { ?x foaf:family_name ?fname } .
}
```


Result

```
@prefix vcard: <http://www.w3.org/2001/vcard-rdf/3.0#> .

_:v1 vcard:N         _:x .
_:x vcard:givenName  "Alice" .
_:x vcard:familyName "Hacker" .

_:v2 vcard:N         _:z .
_:z vcard:givenName  "Bob" .
_:z vcard:familyName "Hacker" .
```

Example 19 (ASK):

Data

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

_:a  foaf:name       "Alice" .
_:a  foaf:homepage  <http://work.example.org/alice/> .
_:b  foaf:name       "Bob" .
_:b  foaf:mbox      <mailto:bob@work.example> .
```

Query1: yes

```
PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
ASK  { ?x foaf:name "Alice" }
```

Query2: no

```
PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
ASK  { ?x foaf:name "Alice" ;
          foaf:mbox  <mailto:alice@work.example> }
```