Apache Jena

- Home
- Download
- Learn
  - Tutorials
  - Overview
  - RDF core API tutorial
  - SPARQL tutorial
  - Manipulating SPARQL using ARQ
  - Using Jena with Eclipse
  - How-To's
  -
  - References
  - Overview
  - Javadoc
  - RDF API
  - RDF I/O
  - ARQ (SPARQL)
  - RDF Connection - SPARQL API
  - Elephas - tools for RDF on Hadoop
  - Text Search
  - TDB
  - SDB
  - SPARQL over JDBC
  - Fuseki
  - Permissions
  - Assembler
  - Ontology API
  - Inference API
  - Command-line tools
  - Extras
- Javadoc
  - Jena Core
  - ARQ
  - TDB
  - Fuseki
  - Elephas
  - Text Search
  - Spatial Search
  - Permissions
  - JDBC
  - All Javadoc
- Ask
- Get involved
  - Contribute
  - Report a bug
  -
  - Project
  - About Jena
  - Roadmap
  - Architecture
  - Project team

SPARQL Tutorial - Alternatives in a Pattern

Another way of dealing with the semi-structured data is to query for one of a number of possibilities. This section covers `UNION` patterns, where one of a number of possibilities is tried.

UNION - two ways to the same data

Both the vCard vocabulary and the FOAF vocabulary have properties for people's names. In vCard, it is vCard:FN, the "formatted name", and in FOAF, it is foaf:name. In this section, we will look at a small set of data where the names of people can be given by either the FOAF or the vCard vocabulary.

Suppose we have an RDF graph that contains name information using both the vCard and FOAF vocabularies.

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

_:a foaf:name   "Matt Jones" .

_:b foaf:name   "Sarah Jones" .

_:c vcard:FN    "Becky Smith" .

_:d vcard:FN    "John Smith" .
```

A query to access the name information, when it can be in either form, could be (q-union1.rq):

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

SELECT ?name
WHERE
{
   { [] foaf:name ?name } UNION { [] vCard:FN ?name }
}
```

This returns the results:

```
-----------------
| name          |
=================
| "Matt Jones"  |
| "Sarah Jones" |
| "Becky Smith" |
```

```
| "John Smith"  |
-----------------
```

It didn't matter which form of expression was used for the name, the ?name variable is set. This can be achieved using a `FILTER` as this query (q-union-1alt.rq) shows:

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

SELECT ?name
WHERE
{
  [] ?p ?name
  FILTER ( ?p = foaf:name || ?p = vCard:FN )
}
```

testing whether the property is one URI or another. The solutions may not come out in the same order. The first form is more likely to be faster, depending on the data and the storage used, because the second form may have to get all the triples from the graph to match the triple pattern with unbound variables (or blank nodes) in each slot, then test each `?p` to see if it matches one of the values. It will depend on the sophistication of the query optimizer as to whether it spots that it can perform the query more efficiently and is able to pass the constraint down as well as to the storage layer.

UNION - remembering where the data was found.

The example above used the same variable in each branch. If different variables are used, the application can discover which sub-pattern caused the match (q-union2.rq):

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

SELECT ?name1 ?name2
WHERE
{
   { [] foaf:name ?name1 } UNION { [] vCard:FN ?name2 }
}
```

```
---------------------------------
| name1         | name2         |
=================================
| "Matt Jones"  |               |
| "Sarah Jones" |               |
|               | "Becky Smith" |
|               | "John Smith"  |
---------------------------------
```

This second query has retained information of where the name of the person came from by assigning the name to different variables.

OPTIONAL and UNION

In practice, `OPTIONAL` is more common than `UNION` but they both have their uses. `OPTIONAL` are useful for augmenting the solutions found, `UNION` is useful for concatenating the solutions from two possibilities. They don't necessary return the information in the same way:

Query(q-union3.rq):

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

SELECT ?name1 ?name2
WHERE
{
  ?x a foaf:Person
  OPTIONAL { ?x  foaf:name  ?name1 }
  OPTIONAL { ?x  vCard:FN   ?name2 }
}
```

```
---------------------------------
| name1         | name2         |
=================================
| "Matt Jones"  |               |
| "Sarah Jones" |               |
|               | "Becky Smith" |
|               | "John Smith"  |
---------------------------------
```

but beware of using `?name` in each `OPTIONAL` because that is an order-dependent query.

Next: Named Graphs