Axel Polleres
axel.polleres@wu.ac.at

XQuery and SPARQL... Some thoughts...
Connection points to tie together Query languages

- E.g., XQUERY+SPARQL = XSPARQL
- For those who haven’t looked at it: SPARQL1.1
- Latest:
  - Other formats: RDB, CSV JSON
  - Streams, Windows and Big Data
**Transformation and Query Languages**

XML Transformation Language
Syntax: XML

- XSLT
- XQuery

- XPath is the common core
- Mostly used to select nodes from an XML doc

**XML world**

- XML Query Language
- non-XML syntax

**RDF world**

- Query Language for RDF
- Pattern based
- declarative

SPARQL

SPARQL XML Result format
SPARQL + Linked Data give you
Semantic search almost "for free"

- Which bands origin from Kitee?

```sparql
SELECT ?X
WHERE
{
}
```

- Try it out at http://dbpedia.org/sparql
SPARQL – Standard RDF Query Language and Protocol

- SPARQL (2008):
  
  ```sparql
  SELECT ?X
  WHERE
  {
  }
  ```

  - SQL “Look-and-feel” for the Web
  - Essentially “graph matching” by *triple patterns*
  - Allows conjunction (.), disjunction (UNION), optional (OPTIONAL) patterns and filters (FILTER)
  - Construct new RDF from existing RDF
  - Solution modifiers (DISTINCT, ORDER BY, LIMIT, ...)
  - A **standardized** HTTP based protocol:
Missing features in SPARQL1.0 (and why SPARQL1.1 was needed)

Based on implementation experience, in 2009 new W3C SPARQL WG founded to address common feature requirements requested urgently by the community:

http://www.w3.org/2009/sparql/wiki/Main_Page

1. Negation
2. Assignment/Project Expressions
3. Aggregate functions (SUM, AVG, MIN, MAX, COUNT, ...)
4. Subqueries
5. Property paths
6. Federated Queries
7. Updates
8. Entailment Regimes (RDFS, OWL, ...)

- Other issues for wider usability:
  - Result formats (JSON, CSV, TSV),
  - Graph Store Protocol (REST operations on graph stores)

- SPARQL 1.1 W3C Recommendation since March 2013
e.g., 6. Federated Queries in SPARQL1.1

Find which persons in DBPedia have the same birthday as Axel (foaf-file):

**SPARQL 1.1 has new feature SERVICE to query remote endpoints**

```
PREFIX dbpedia2: <http://dbpedia.org/property/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?N ?MyB
FROM <http://polleres.net/foaf.rdf>
{ [ foaf:birthday ?MyB ].

    SERVICE <http://dbpedia.org/sparql> { SELECT ?N WHERE {
        [ dbpedia2:born ?B; foaf:name ?N ]. } }

    FILTER ( Regex(Str(?B),str(?MyB)) )
}
```

 Doesn’t work in practice, ☹ as SERVICE endpoints often only returns limited results…
e.g., 6. Updates

- SQL has not only a query language, but also a Data manipulation language.

→ SPARQL Update to fill this gap:

```sparql
PREFIX ex: <http://example.org/>
DELETE { ?Item ex:price ?Pr }
INSERT { ?Item ex:price ?NewPr }
WHERE { ?Item ex:price ?Pr
          BIND (?Pr * 1.1 AS ?NewPr ) }
```

→ Allows to change/update an RDF Store from outside, again via standard HTTP protocol.
XSPARQL

**Idea:** One approach to conveniently query XML and RDF side-by-side: XSPARQL

- Transformation language
- Consume and generate XML and RDF
- Syntactic extension of XQuery, ie.

\[
\text{XSPARQL} = \text{XQuery} + \text{SPARQL}
\]
XSPARQL Language Specification

W3C Member Submission 20 January 2009

This version:  
http://www.w3.org/Submission/2009/SUBM-xsparql-language-specification-20090120/

Latest version:  
http://www.w3.org/Submission/xsparql-language-specification/

Authors:  
Axel Polleres • DERI, NUI Galway
Thomas Krennwallner • Institute of Information Systems, Vienna University of Technology
Nuno Lopes • DERI, NUI Galway
Jacek Kopecký • University of Innsbruck
Stefan Decker • DERI, NUI Galway

Copyright © 2009 by DERI Galway at the National University of Ireland, Galway, Ireland.

This work is supported by Science Foundation Ireland under grants number SFI/02/CE1/I131 and SFI/08/CE/I1380 and under the European Commission European FP6 project inContext (IST-034718).

This document is available under the W3C Document License. See the W3C Intellectual Rights Notice and Legal Disclaimers for additional information.
# XSPARQL: Syntax overview (I)

## Data Input (XML or RDF)

### Prefix declarations

| P | declare namespace prefix="namespace-URI" 
   | or prefix prefix: <namespace-URI> |

## Body:

| FLW | for var [at posVar] in FLOWR’ expression 
    | let var := FLWOR’ expression 
    | where FLWOR’ expression 
    | order by FLWOR’ expression |

| F’ | for varlist [at posVar] 
   | from / from named ( <dataset-URI> or FLWOR’ expr.) 
   | where { pattern } 
   | order by expression 
   | limit integer > 0 
   | offset integer > 0 |

## Data Output (XML or RDF)

| C | construct 
   | { template (with nested FLWOR’ expressions) } |

| R | return XML+ nested FLWOR’ expressions |
### XSPARQL Syntax overview (II)

<table>
<thead>
<tr>
<th>Concept</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>XQuery or SPARQL prefix declarations</td>
<td><code>P</code> declare namespace prefix=&quot;namespace-URI&quot; or prefix prefix: &lt;namespace-URI&gt;</td>
</tr>
<tr>
<td>Any XQuery query</td>
<td><code>F</code> for var [at posVar] in FLOWR’ expression</td>
</tr>
<tr>
<td></td>
<td><code>FLW</code> let var := FLOWR’ expression</td>
</tr>
<tr>
<td></td>
<td><code>FLWO</code> where FLOWR’ expression</td>
</tr>
<tr>
<td></td>
<td><code>FLWOR</code> order by FLOWR’ expression</td>
</tr>
<tr>
<td></td>
<td><code>FLWOR</code> order by FLOWR’ expression</td>
</tr>
<tr>
<td></td>
<td><code>FLODWM</code> for varlist [at posVar]</td>
</tr>
<tr>
<td></td>
<td><code>FDWM</code> from / from named (&lt;dataset-URI&gt; or FLOWR’ expr.)</td>
</tr>
<tr>
<td></td>
<td><code>DW</code> where { pattern }</td>
</tr>
<tr>
<td></td>
<td><code>DM</code> order by expression</td>
</tr>
<tr>
<td></td>
<td><code>DM</code> limit integer &gt; 0</td>
</tr>
<tr>
<td></td>
<td><code>DM</code> offset integer &gt; 0</td>
</tr>
<tr>
<td>SPARQL FOR Clause represents a SPARQL query</td>
<td><code>C</code> construct { template (with nested FLOWR’ expressions) } or return XML+ nested FLOWR’ expressions</td>
</tr>
</tbody>
</table>

Legend:
- **P**: Declare namespace prefix
- **F**: For loop
- **L**: Let statement
- **W**: Where clause
- **O**: Order by clause
- **D**: From clause
- **M**: Where clause
- **C**: Construct clause
- **R**: Return clause
Federated Queries in SPARQL1.1

Find which persons in DBPedia have the same birthday as Axel (foaf-file):

**SPARQL 1.1 has new feature SERVICE to query remote endpoints**

```sparql
PREFIX dbpedia2: <http://dbpedia.org/property/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?N ?MyB
FROM <http://polleres.net/foaf.rdf>
{ [ foaf:birthday ?MyB ].

SERVICE <http://dbpedia.org/sparql> { SELECT ?N WHERE {
    FILTER ( Regex(str(?B),str(?MyB)) )
} }
}

Doesn’t work!!! ?MyB unbound in SERVICE query
```
e.g. Federated Queries

*Find which persons in DBPedia have the same birthday as Axel (foaf-file):*

**In XSPARQL:**

```sparql
prefix dbprop: <http://dbpedia.org/property/>
prefix foaf: <http://xmlns.com/foaf/0.1/>
prefix : <http://xsparql.deri.org/bday#>

let $MyB := for * from <http://polleres.net/foaf.rdf>
  where { [ foaf:birthday $B ]. }
  return $B

for * from <http://dbpedia.org/> endpoint <http://dbpedia.org/sparql>
  where { [ dbprop:born $B; foaf:name $N ].
    filter ( regex(str($B),str($MyB)) ) }  
  construct { :axel :sameBirthDayAs $N }
```

Specifies the endpoint to perform the query, similar to SERVICE in SPARQL1.1

Works! In XSPARQL bound values (?)MyDB are injected into the SPARQL subquery → More direct control over “query execution plan”
Connection points to tie together languages & formats

- Other formats: RDB, CSV JSON
  - RDB2RDF, REC out since Sept 2012
    [http://www.w3.org/TR/r2rml/](http://www.w3.org/TR/r2rml/)
    mapping language to transform RDB to RDF
  - JSON-LD, REC out since Jan 2014
    [http://www.w3.org/TR/json-ld/](http://www.w3.org/TR/json-ld/)
  - CSV on the Web WG
    [http://www.w3.org/2013/05/lcsv-charter.html](http://www.w3.org/2013/05/lcsv-charter.html)

- Streams, Windows and Big Data:
  - [http://www.w3.org/community/rsp/](http://www.w3.org/community/rsp/) RDF Stream Processing Community Group
  - Related?
    [http://www.w3.org/TR/xquery-30/#id-windows](http://www.w3.org/TR/xquery-30/#id-windows)

- Binary format/compression is becoming an issue:
Linked Data, RDF and SPARQL... Why at all?

• Standard protocol and standard means to integrate different datasets
• Standard way to query the data and the schema alongside (Schema & Data decoupled)
• „share [...] data in ways that are easily discoverable, useable, or understandable by the public“
• Increasingly supported/used in Open Data, e.g.

![European Union Open Data Portal](image)