Breakmarking UniProt RDF: SPARQL that make your database cry....

Jerven Bolleman
Developer
UniProtKB/Swiss-Prot

Friday 4 April 14
UniProt.rdf

UniProt

UniProt RDF data shape challenges

why SPARQL

Benchmark

Friday 4 April 14

Talk two things
uniprot.rdf
SPARQL!
UniProt.rdf

UniProt

UniProt RDF data shape challenges

why SPARQL

Benchmark

Friday 4 April 14

Talk two things

uniprot.rdf

Quality!
SPARQL?
Give me a better pipette

SPARQL does not make a biologist happy
It makes you happier so you can make the biologist happy
• Everything possible with SPARQL is possible with Clay tablets
• Information stays information
• Only difference is number of slaves, um I mean PhD students you need
• Clay is more expensive than FLASH ;)
• Excellent retention times :D
SPARQL against

- RDBMS
  - R2RML -> D2RQ, Ultrawrap, XSPARQL...
- Programs
  - SADI...
- Triplestore
  - Mark logic, Jena, Virtuoso, OWLIM, uRiKA, Oracle spatial, Oracle NoSQL, IBM DB2, etc...
- Biological flat file formats
  - sparql-bed
- CSV/TSV/Spreadsheets
  - Tarql, Sparqlify

No matter what query language you currently use:
Translating from SPARQL is possible
Data storage is decoupled from querying
Only speed for some query types is affected
UniProt.rdf

UniProt RDF data shape challenges

why SPARQL

Benchmark

Friday 4 April 14

Talk two things

uniprot.rdf

Quality!
Growing & Living knowledgebase

• Dataset grew from
  – 80 million in 2006
  – 12 billion 2014
  – more every 4 weeks regular release

• Data model changes over time
  – owl:sameAs -> skos:exactMatch
  – FALDO for positions
  – more structure
  – sha checksums
  – uniparc (drop reification 2014_05)
63% more triples in a year

In 364 days! Doubling time 15 months instead of 18 months!
Information growth is faster than entry growth!
250% in 18 months instead of 200%
RDF normalization

- Entry based view is 40% repetitive data
  - 1 annotation in 12 entries (average)
  - high throughput papers
  - species names etc...

- Significantly changes number of triples
UniProt.rdf: An **improving** experience

- 2006 powerful server of the day
  - 80 **million** triples take a week to load in a triple store
  - **SERQL** queries **may** return results
- 2014 powerful server of the day
  - 12 **billion** triples take a week to load in a triple store (32 hours for key-value store)
  - **SPARQL** queries **do** return results
UniProt.rdf

UniProt

why SPARQL

UniProt RDF data shape challenges

Benchmark

Talk two things
uniprot.rdf
Quality!
## Stats

<table>
<thead>
<tr>
<th></th>
<th>UniProt</th>
<th>LUBM</th>
<th>SP2Bench</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graphs</td>
<td>16</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>types</td>
<td>164</td>
<td>19</td>
<td>13</td>
</tr>
<tr>
<td>object properties</td>
<td>58</td>
<td>25</td>
<td>20...</td>
</tr>
<tr>
<td>datatype properties</td>
<td>81</td>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>
Benchmarks should help us make choices
Benchmarks should help us make choices

Key Value
DESCRIBE

Analytics
SELECT

OLTP
INSERT/DELETE

Graph Path

Lack of well known SPARQL benchmarks

LUBM

SP²Bench

CONSTRUCT
{
}
WHERE {
  GRAPH<http://purl.uniprot.org/uniprot/P05067>{
  }
}
Our benchmark

- Rule base classification turned into SPARQL
  - 1250 queries (now 1600)
  - 54 BGP avg per query (construct)
    - 175 max
    - 20 min
  - 50% have negation
- Run on 10 billion + triples

Halve that for select.
Equivalent variants of queries

- MINUS or NOT EXISTS
  - May be different semantics, made sure queries match
- UNION or VALUES

- Different engines have different optimizations
  - Try to find the form they have optimized
Example

```sparql
SELECT (COUNT(DISTINCT ?this) AS ?countTotal)
WHERE {
  ?this a up:Protein .
  ?this up:reviewed false .
  ?this rdfs:seeAlso panther:PTHR11361 .
  {?this rdfs:seeAlso smart:SM00533> .}
UNION {?this rdfs:seeAlso pfam:PF05192> .}
FILTER (NOT EXISTS {
  {?this rdfs:seeAlso interpro:IPR006153 .}
UNION {?this rdfs:seeAlso interpro:IPR000626 .}
UNION {?this rdfs:seeAlso interpro:IPR005061 .}
UNION {?this rdfs:seeAlso interpro:IPR007720 .}
UNION {?this rdfs:seeAlso interpro:IPR003583 .}
UNION {?this rdfs:seeAlso interpro:IPR004771 .}
UNION {?this rdfs:seeAlso interpro:IPR016040 .}
UNION {?this rdfs:seeAlso interpro:IPR003148 .}
UNION {?this rdfs:seeAlso interpro:IPR006055 .}
UNION {?this rdfs:seeAlso interpro:IPR000727 .}
UNION {?this rdfs:seeAlso interpro:IPR013520 .}
UNION {?this rdfs:seeAlso interpro:IPR000160 .}
UNION {?this rdfs:seeAlso interpro:IPR000873 .}}
)
?this up:organism ?taxon .
  {?taxon rdfs:subClassOf* taxonomy:2157. }
UNION {?taxon rdfs:subClassOf* taxonomy:2759. }
UNION {?taxon rdfs:subClassOf* taxonomy:2 .}
```
Example

```
SELECT (COUNT(DISTINCT ?this) AS ?countTotal)
WHERE {
    ?this a up:Protein .
    ?this up:reviewed false .
    ?this rdfs:seeAlso panther:PTHR11361 .
VALUES (?theseLinks) {(smart:SM00533) (pfam:PF05192) }
?this rdfs:seeAlso ?theseLinks .
VALUES (?notTheseLinks) { (interpro:IPR006153)
    (interpro:IPR000626)
    (interpro:IPR005061)
    (interpro:IPR007720)
    (interpro:IPR003583)
    (interpro:IPR004771)
    (interpro:IPR016040)
    (interpro:IPR003148)
    (interpro:IPR006055)
    (interpro:IPR000727)
    (interpro:IPR013520)
    (interpro:IPR000160)
    (interpro:IPR000873)}
MINUS { ?this rdfs:seeAlso ?notTheseLinks . }
VALUES (?supertaxon) { (taxonomy:2157)
    (taxonomy:2759)
    (taxonomy:2) }
?this up:organism/rdfs:subClassOf* ?supertaxon . }
```
Example:

```
SELECT (COUNT(DISTINCT ?this) AS ?countTotal)
WHERE {
  ?this a up:Protein .
  ?this up:reviewed false .
  ?this rdfs:seeAlso panther:PTHR11361 .
VALUES (?theseLinks) {(smart:SM00533) (pfam:PF05192) }
?this rdfs:seeAlso ?theseLinks .
VALUES (?notTheseLinks) { (interpro:IPR006153)
  (interpro:IPR000626)
  (interpro:IPR005061)
  (interpro:IPR007720)
  (interpro:IPR003583)
  (interpro:IPR004771)
  (interpro:IPR016040)
  (interpro:IPR003148)
  (interpro:IPR006055)
  (interpro:IPR00727)
  (interpro:IPR013520)
  (interpro:IPR000160)
  (interpro:IPR000873)}
MINUS { ?this rdfs:seeAlso ?notTheseLinks . }
VALUES (?supertaxon) {(taxonomy:2157)
  (taxonomy:2759)
  (taxonomy:2) }
?this up:organism/rdfs:subClassOf* ?supertaxon . }
```

2 billion

56 million

1 million taxnodes
Hardware/Software qualitative

- Owlim 5.2 (5.4 is faster)
  - Completes the test
  - 256GB ram/200GB java heap
  - 2 slow disks (5.4 gets SSDs)
  - 64 core AMD
- uRiKa
  - Matthorn at CSCS 2TB Ram
- Oracle almost 12c
  - 1/4 exadata
Future systems to test

- **Virtuoso 7.1**
  - Promising candidate
  - Not reviewed yet

- **BigData**
  - 1.0 never finished loading
  - Retest after 2 years
    - Looking into the cluster
jervenbolleman

jerven.bolleman@isb-sib.ch

answers@semanticweb.com

Friday 4 April 14
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX up:<http://purl.uniprot.org/core/>
PREFIX ko:<http://purl.uniprot.org/ko/>
WHERE
{
?protein up:organism ?taxon ;
  rdfs:seeAlso ko:K00399 ;
  up:annotation ?annotation ;
  ^(up:member/up:sequenceFor) ?cluster .
?annotation a up:Pathway_Annotation ;
  rdfs:seeAlso ?pathway .
}