

Simplified RDB2RDF Mapping

Claus Stadler, Jörg Unbehauen, Patrick Westphal, Mohamed Ahmed Sherif and Jens Lehmann



presented by Axel-Cyrille Ngonga Ngomo

2015 May 19

- 1 *Motivation*
- 2 *R2RML in a Nutshell*
- 3 *SML in a Nutshell*
- 4 *SML Step by Step Example*
- 5 *Evaluation*

- 1 *Motivation*
- 2 *R2RML in a Nutshell*
- 3 *SML in a Nutshell*
- 4 *SML Step by Step Example*
- 5 *Evaluation*

Motivation - RDB2RDF Approaches

Several tools exist that implemented different approaches for mapping relational databases to RDF, of which R2RML became a W3C standard (<http://www.w3.org/TR/r2rml/>).

```
1 map:eventTitle a d2rq:PropertyBridge;  
2   d2rq:belongsToClassMap map:  
3     Conference;  
4   d2rq:property :eventTitle;  
5   d2rq:column "Conferences.Name";  
6   d2rq:datatype xsd:string;
```

D2RQ

```
1 [MappingDeclaration] @collection [[  
2   mappingId      Book collection  
3   target         :BID_{id} a :Book .  
4   source         SELECT id FROM books  
5 ]]
```

Ontop

```
1 graph <http://localhost/testdata/  
2   products#>  
3 subject prd:product_iri(PRODUCT.  
4   PRODUCT_ID)  
5 predicate rdf:type  
6 object prd:Product
```

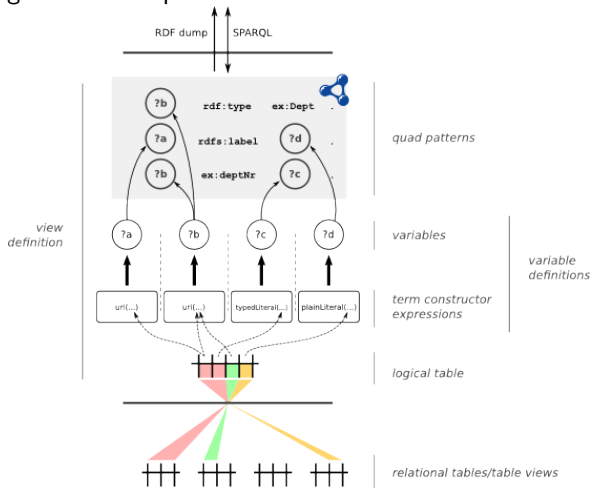
Virtuoso RDF views

```
1 <#emps>  
2   rr:logicalTable [  
3     rr:tableName "employees"  
4   ] ;  
5   rr:subjectMap [  
6     rr:template "http://ex.org/{id}"  
7     rr:class foaf:Person  
8   ] .
```

R2RML

From Tables to Triples

All these approaches iterate tables and on every row they first create RDF terms and then arrange them to triples:



Our Approach

- In SQL, there is the well known CREATE VIEW statement to create views from tables and other views.
- Quad stores essentially use a table with four columns to store RDF data.
- Current RDB2RDF approaches are quite different from how views are created in SQL.
- Our approach is to blend the traditional SQL CREATE VIEW statements with SPARQL CONSTRUCT queries:

```
1  PREFIX foaf: <http://xmlns.com/foaf/0.1/>
2  PREFIX ex: <http://example.org/>
3  CREATE VIEW emps AS
4      CONSTRUCT {
5          ?s a foaf:Person
6      }
7      With
8          ?s = uri(ex:, ?id)
9      From
10         employees
```

- Definition of the compact **Sparqlification Mapping Language** (SML) mapping language with equal expressiveness to R2RML
- A unified formal model of RDB2RDF mapping languages.
- User Study which compares SML to R2RML
- Tooling: SML/R2RML Converters and Syntax Highlighters

Outline

- 1 *Motivation*
- 2 *R2RML in a Nutshell*
- 3 *SML in a Nutshell*
- 4 *SML Step by Step Example*
- 5 *Evaluation*

An **R2RML mapping** is an RDF resource that must be described with the following properties:

- Exactly one **rr:logicalTable**, which refers to the view's logical table, i.e. an SQL query, SQL table or SQL view.
- Exactly one **rr:subjectMap**, which defines the subject of the triples created from this mapping
- Zero or more instances of **rr:predicateObjectMap**, that attach a set of predicate/object pairs using **rr:predicateMap** and **rr:objectMap** to the corresponding subject.
- Each of **rr:subjectMap**, **rr:predicateMap** and **rr:objectMap** must be further described to specify what RDF terms to create from every row of the logical table.

Note, that R2RML offers a set of **shortcut properties**, which we do not discuss for brevity.

Example of an R2RML mapping

Generic form of an R2RML mapping without the use of shortcuts:

- R2RML Example:

```
1 @prefix foaf: <http://xmlns.com/foaf/0.1/> .
2
3 <#emps>
4   rr:logicalTable [ rr:tableName "employees" ] ;
5   rr:subjectMap [ rr:template "http://example.org/{id}" ];
6   rr:predicateObjectMap [
7     rr:predicateMap [ rr:constant rdf:type ] ;
8     rr:objectMap [ rr:constant foaf:Person ]
9   ] .
```

- 1 *Motivation*
- 2 *R2RML in a Nutshell*
- 3 *SML in a Nutshell*
- 4 *SML Step by Step Example*
- 5 *Evaluation*

A SML view comprises:

- A name
- A **CONSTRUCT** clause for which quads to create
- A **FROM** clause for the underlying logical table.
- a **WITH** clause that creates RDF terms from the columns of the logical table and assigns them to variables
- Optionally, a **CONSTRAINT** clause, where URI prefixes of variables can be stated (can be used for pruning joins in SPARQL-to-SQL rewriters).

Example of an SML View

- SML Example:

```
1  PREFIX foaf: <http://xmlns.com/foaf/0.1/>
2  PREFIX ex: <http://example.org/>
3  CREATE VIEW emps AS
4      CONSTRUCT {
5          ?s a foaf:Person
6      }
7      With
8          ?s = uri(ex:, ?id)
9      From
10         employees
```

Creating RDF Terms in SML and R2RML

SML RDF term constructor	R2RML term map
bNode(?COL)	... [rr:column "COL" ; rr:termType rr:blankNode]
bNode(expr)	... [rr:template "asTemplate(expr)" ; rr:termType rr:blankNode]
uri(expr)	... [rr:(constant column template) "asTemplate(expr)"; rr:termType rr:IRI]
plainLiteral(?COL)	... [rr:column "COL"]
plainLiteral(expr)	... [rr:template "asTemplate(expr)"]
typedLiteral(?COL, xsd:int)	... [rr:column "COL" ; rr:datatype xsd:int]
typedLiteral(expression, xsd:int)	... [rr:template "asTemplate(expr)" ; rr:datatype xsd:int]

Table : Transformation of SML term constructors to R2RML term maps

- 1 *Motivation*
- 2 *R2RML in a Nutshell*
- 3 *SML in a Nutshell*
- 4 *SML Step by Step Example*
- 5 *Evaluation*

SML Mapping Example

- The following slides demonstrate how to map relational data to RDF with the Sparqlification Mapping Language (SML).
- Thereby, these prefixes are used:

Prefixes	
prefix	IRI
rdfs	http://www.w3.org/2000/01/rdf-schema#
ogc	http://www.opengis.net/ont/geosparql#
geom	http://geovocab.org/geometry#
lgd	http://linkedgeodata.org/triplify/
lgd-geom	http://linkedgeodata.org/geometry/

SML - Mapping Example: The Goal (1/4)

Input Table

nodes	
id	geom
1	POINT(0 0)
2	POINT(1 1)

- How to map tables to RDF?
 - How to introduce the commonly used distinction in GIS between feature and geometry?

Aimed for RDF Output

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .  
...
```

```
lgd:node1 geom:geometry lgd-geom:node1 .  
lgd:node2 geom:geometry lgd-geom:node2 .
```

```
lgd-geom:node1 ogc:asWKT "POINT(0 0)"^^ogc:wktLiteral .  
lgd-geom:node2 ogc:asWKT "POINT(1 1)"^^ogc:wktLiteral .
```

SML - Mapping Example: SML Syntax Outline (2/4)

Input Table

nodes	
id	geom
1	POINT(0 0)
2	POINT(1 1)

```
Create View myNodesView As
Construct {
    ...
}
With
    ...
From
    ...
```

Aimed for RDF Output

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
...

lgd:node1 geom:geometry lgd-geom:node1 .
lgd:node2 geom:geometry lgd-geom:node2 .

lgd-geom:node1 ogc:asWKT "POINT(0 0)"^^ogc:wktLiteral .
lgd-geom:node2 ogc:asWKT "POINT(1 1)"^^ogc:wktLiteral .
```

SML - Mapping Example: Construct and From (3/4)

Input Table

nodes	
id	geom
1	POINT(0 0)
2	POINT(1 1)

```
Create View myNodesView As
Construct {
  ?n geom:geometry ?g .
  ?g ogc:asWKT ?o
}
With
  ...
From nodes
```

Aimed for RDF Output

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
...

lgd:node1 geom:geometry lgd-geom:node1 .
lgd:node2 geom:geometry lgd-geom:node2 .

lgd-geom:node1 ogc:asWKT "POINT(0 0)"^^ogc:wktLiteral .
lgd-geom:node2 ogc:asWKT "POINT(1 1)"^^ogc:wktLiteral .
```

SML - Mapping Example: Complete! (4/4)

Input Table

nodes	
id	geom
1	POINT(0 0)
2	POINT(1 1)

```
Create View myNodesView As
Construct {
  ?n geom:geometry ?g .
  ?g ogc:asWKT ?o
}
With
  ?n = uri(lgd:node, ?id)
  ?g = uri(lgd-geom:node, ?id)
  ?o = typedLiteral(?geom,
                    ogc:wktLiteral)
From nodes
```

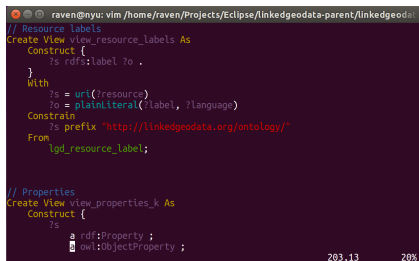
Aimed for RDF Output

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
...
lgd:node1 geom:geometry lgd-geom:node1 .
lgd:node2 geom:geometry lgd-geom:node2 .

lgd-geom:node1 ogc:asWKT "POINT(0 0)"^^ogc:wktLiteral .
lgd-geom:node2 ogc:asWKT "POINT(1 1)"^^ogc:wktLiteral .
```

Website: <http://sml.aksw.org>

- R2RML \leftrightarrow SML converter
- Syntax Highlighters for vim and CodeMirror (a JavaScript IDE component; used in the user study).
- SML in use at LinkedGeoData and Panlex

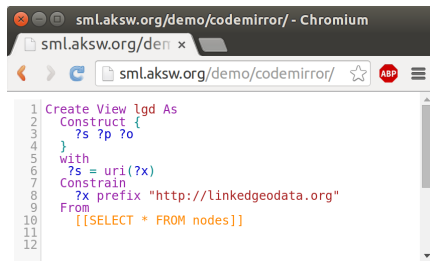


The screenshot shows a vim editor window with the following SML code:

```
// Resource labels
Create View view_resource_labels As
Construct {
  ?s rdfs:label ?o .
}
With
  ?s = uri(?resource)
  ?o = plainLiteral(?label, ?language)
Constrain
  ?s prefix "http://linkedgeodata.org/ontology/"
From
  lgd_resource_label;

// Properties
Create View view_properties_k As
Construct {
  ?s
    a rdf:Property ;
    owl:ObjectProperty ;
```

The status bar at the bottom indicates line 203, column 13, and a 20% zoom level.



The screenshot shows a Chromium browser window at the URL sml.aksw.org/demo/codemirror/. The editor displays the following SML code:

```
1 Create View lgd As
2   Construct {
3     ?s ?p ?o
4   }
5   with
6     ?s = uri(?x)
7   Constrain
8     ?x prefix "http://linkedgeodata.org"
9   From
10    [[SELECT * FROM nodes]]
```

Outline

- 1 *Motivation*
- 2 *R2RML in a Nutshell*
- 3 *SML in a Nutshell*
- 4 *SML Step by Step Example*
- 5 *Evaluation*

We performed a user study with the goal to answer the following questions:

- Is SML easier to read than R2RML and does SML have a lower entry barrier than R2RML?
- Can people understand SML mappings or R2RML mappings faster?
- If given the choice, would people prefer SML or R2RML?

46 humans completed the survey of which 28 performed all tasks correctly.

- Participants first were asked to do a self-assessment on their familiarity with technologies related to RDB2RDF.
- Then they were presented 5 multiple-choice tasks each for R2RML and SML (10 tasks in total).
- Finally, after having completed the tasks, users could score their impression and preference on R2RML / SML.

User Study - Familiarity

Familiarity

- The topic of RDB2RDF is (or may become) relevant for one of my projects
(1=not at all ... 5=absolutely)

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

- I am familiar with the Turtle RDF syntax (1=not at all, 2=have seen it before, 3=know some basic concepts, 4=capable of working with it, 5=can write it from scratch)

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

- I am familiar with the SPARQL syntax
(1=not at all, 2=have seen it before, 3=know some basic concepts, 4=can write some simple queries from scratch, 5=can write rather sophisticated queries from scratch)

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

- I am familiar with the SQL syntax
(1=not at all, 2=have seen it before, 3=know some basic concepts, 4=can write some simple queries from scratch, 5=can write rather sophisticated queries from scratch)

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

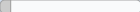
- I am familiar with R2RML
(1=not at all, 2=have seen it before, 3=know some basic concepts, 4=capable of using it with reference information, 5=can write mappings from scratch)

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

- I am familiar with SML
(1=not at all, 2=have seen it before, 3=know some basic concepts, 4=capable of using it with reference information, 5=can write mappings from scratch)

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

User Study - Task 1 - SML

0%  100%

Task 1 (SML): Find the Output [Warm Up]

One simple task for each R2RML and SML

- Mark all the triples that are generated from the given table using the given view.
(Please note that the symbol *a* is a shorthand for *rdf:type*.)

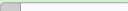
departments	
id (int)	name (text)
1	Development

```
1 Prefix ex: <http://example.com/>
2
3 Create View DepartmentsView As
4   Construct {
5     ?s a ex:Department
6   }
7   With
8     ?s = uri(ex:, ?id)
9   From
10    departments
11
```

Check any that apply

- ☐ `1` `<http://example.com/1> ex:id 1 .`
- ☐ `1` `<http://example.com/Department> a ex:Department .`
- ☐ `1` `<http://example.com/1> a ex:Department .`
- ☐ `1` `ex:Department a "1" .`
- ☐ I cannot make sense out of this mapping

User Study - Task 1 - R2RML

0%  100%

Task 1 (R2RML): Find the Output [Warm Up]

- Mark all the triples that are generated from the given table using the given view.
(Please note that the symbol *a* is a shorthand for *rdf:type*.)

employees	
id (int)	name (text)
1	Susan

```
1 @prefix rr: <http://www.w3.org/ns/r2rml#> .
2 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
3 @prefix ex: <http://example.com/> .
4
5 <EmployeesMap>
6   a rr:TriplesMap;
7   rr:logicalTable [ rr:tableName "employees" ] ;
8   rr:subjectMap [
9     rr:template "http://example.com/{id}"
10    ] ;
11   rr:predicateObjectMap [
12     rr:predicate rdf:type ;
13     rr:object ex:Employee
14   ] .
15
```

Check any that apply

- ☐ 1 <http://example.com/1> ex:id 1 .
- ☐ 1 ex:Employee a "1" .
- ☐ 1 <http://example.com/Susan> a ex:Employee .
- ☐ 1 <http://example.com/1> a ex:Employee .
- ☐ I cannot make sense out of this mapping

User Study - Readability

- I found the tasks too difficult (1=not at all ... 5=absolutely)

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

- I was able to make sense of the SML mappings (1=not at all ... 5=absolutely)

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

- I was able to make sense of the R2RML mappings (1=not at all ... 5=absolutely)

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

- I found SML to be easily readable (1=not at all ... 5=absolutely)

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

- I found R2RML to be easily readable (1=not at all ... 5=absolutely)

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

- I could imagine using SML for solving RDB2RDF mapping tasks
(1=not at all ... 5=absolutely)

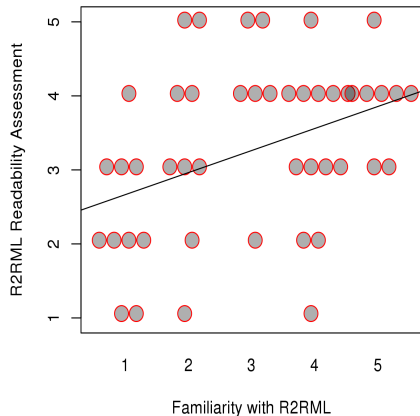
☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

- Which of the languages did you prefer over the other?
1=strong preference for R2RML, 2=weak preference for R2RML
3=indifferent
4=weak preference for SML, 5=strong preference for SML

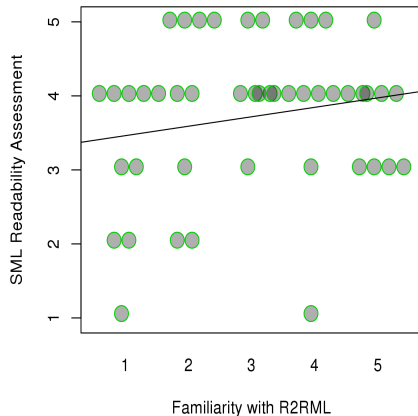
☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

Results: Readability

(a) R2RML Familiarity vs. R2RML Readability



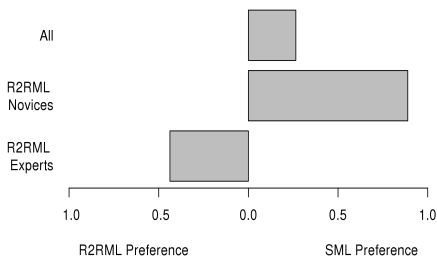
(b) R2RML Familiarity vs. SML Readability



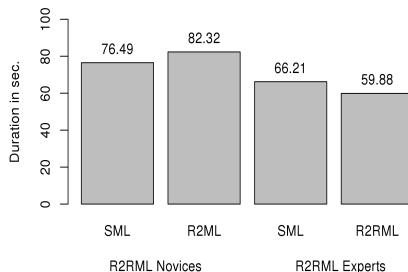
- Readability of SML better than R2RML for novices.

Results: Preference

Absolute Value of Overall Preference



Task 1 Duration



- Novice = Self assessment in R2RML familiarity ≤ 3
- Expert = Self assessment in R2RML familiarity ≥ 4

Conclusions and Future Work

- We introduced the novel Sparqlification Mapping Language (SML) and showed how it relates to R2RML
- Evaluation shows a favor in SML by RDB2RDF novices, providing evidence that SML could simplify RDB2RDF mapping.
- We provided tooling to bridge the gap between SML and R2RML

Future Work

- More testing of the converters (WIP)
- Possibly streamline some language features, such as
 - Usage SPARQL 1.1's **strdt** and **strlang** in favor of **plainLiteral** and **typedLiteral**
 - Introduction of a **FROM QUERY** syntax instead of interpreting content of triple quotes as an SQL query.



The End - Questions/Feedback?

SML Resources: <http://sml.aksw.org>



Claus Stadler
cstadler@informatik.uni-leipzig.de
AKSW/Uni Leipzig



Jens Lehmann
lehmann@informatik.uni-leipzig.de
AKSW/Uni Leipzig



<http://geoknow.eu>