On the Semantics of R2RML and its Relationship with the Direct Mapping

Juan F. Sequeda

Department of Computer Science, University of Texas at Austin
jsequeda@cs.utexas.edu

Abstract. The W3C Relational Database to RDF (RDB2RDF) standards are positioned to bridge the gap between Relational Databases and the Semantic Web. The standards consist of two interrelated and complementary specifications: Direct Mapping of Relational Data to RDF and R2RML: RDB to RDF Mapping Language. In this paper we present initial results on the formal study of the R2RML mapping language by defining its semantics using Datalog. We prove that there are a total of 57 distinct Datalog rules which can be used to generate RDF triples from a relational table. Additionally, we provide insights on the relationship between R2RML and Direct Mapping.

1 Introduction

To live up to the promise of web-scale data integration, the Semantic Web will have to include the content of existing relational databases. In September 2012, two interrelated and complementary W3C standards, Direct Mapping [1] and R2RML [2], were standarized. R2RML is a mapping language which allows users to manually define mappings. Direct Mapping is the default and automatic way to translate relational databases into RDF without any input from a user, which can be represented in R2RML. The Direct Mapping has been well studied. The W3C specifications present the denotational and datalog-based semantics of the Direct Mapping [1]. Additionally, the W3C Direct Mapping has been augmented to include a direct mapping from the relational schema to OWL in order to prove fundamental correctness properties [3]. However, to the best of our knowledge, there has not been a thorough study of the R2RML language and its relationship with the Direct Mapping. As a matter of fact, the semantics of R2RML have not even been formally defined.

Our methodology is to study the problem of mapping relational databases to RDF from two different perspectives: logical and physical. Inspired by the relational query processing workflow, we propose the following workflow. Mappings are expressed in a declarative language: *R2RML*. A parser can translate the mapping into a logical expression: the *logical mapping*. A logical mapping can be rewritten into other equivalent logical mappings which may be better for performance. A logical mapping is translated into a executable program, the *physical mapping*. A physical mapping can then be implemented and optimized in different ways. This paper address the first part of the proposed workflow and presents initial results on the formal study of R2RML by define its semantics using Datalog. We prove that there are a total of 57 distinct Datalog

rules which can be used to generate RDF triples from a relational table. Additionally, we provide insight on the relationship between R2RML and Direct Mapping. It is our hypothesis that a core subset of R2RML has the same expressive power as the Direct Mapping, if views are allowed as input.

2 R2RML

R2RML is a language for expressing mappings from a relational database to RDF. The input of an R2RML mapping $\mathcal{M}$ is a relational schema $\mathbf{R}$ and an instance I of $\mathbf{R}$. The output is an RDF graph. Consider a schema of a database with of tables `EMP(EMPNO,ENAME,DID)` and `DEPT(DEPTNO, DNAME, LOC)`. Moreover, we have the following constraints about the schema of the university: `EMPNO` is the primary key of `EMP`, `DEPTNO` is the primary key of `DEPT` and `DID` is a foreign key in `EMP` that references attribute `DEPTNO` in `DEPT`. An R2RML mapping is represented as an RDF graph itself and also has an associated RDFS schema[1]. For readibility, the RDF Turtle syntax is the recommended syntax to write R2RML mappings.

Example 1. An R2RML Mapping for the example database.

```
@prefix rr: <http://www.w3.org/ns/r2rml#>.
@prefix ex: <http://example.com/ns#>.

<#TriplesMap1>
 rr:logicalTable [ rr:tableName "emp" ];
 rr:subjectMap [ rr:template "http://ex.com/employee/{empno}";
                 rr:class ex:Employee; ];
 rr:predicateObjectMap [ rr:predicate ex:name;
                         rr:objectMap [ rr:column "ename" ]; ];
 rr:predicateObjectMap [ rr:predicate ex:department;
                         rr:objectMap [
                          rr:parentTriplesMap <#TriplesMap2>;
                            rr:joinCondition [
                               rr:child "dept";
                               rr:parent "deptno"; ]; ]; ].
<#TriplesMap2>
 rr:logicalTable [ rr:tableName "dept" ];
 rr:subjectMap [ rr:template "http://ex.com/dept/{deptno}";
                 rr:class ex:Department; ];
 rr:predicateObjectMap [ rr:predicate ex:name;
                         rr:objectMap [ rr:column "DNAME" ]; ].
```

The Datalog rules defining the R2RML semantics can be found at `http://www.cs.utexas.edu/~jsequeda/r2rml`. We briefly explain the components of R2RML with the running example. An R2RML mapping consists of a set of Triple Maps. In the example, there are two Triples Map: `<#TriplesMap1>` and `<#TriplesMap2>`. Each TripleMaps consists of exactly one LogicalTable, exactly one SubjectMap and a set (which may be empty) of Predicate-Object Maps. A LogicalTable is either an existing table/view in the database or a SQL query (known also as an R2RML view). In `<#TriplesMap1>`, the Logical Tables is the table name `"emp"`. A SubjectMap generates an RDF term for the subject and optionally an rdf:type statement. A PredicateObjectMap is a pair of PredicateMap and ObjectMap which generates the RDF terms for the predicate and object respectively of a triple, that is associated to the subject generated by the SubjectMap. An ObjectMap can also be a Referencing Object Map which allows using the subjects of another Triples Map as an object. Since both Triples Maps may be based on different logical tables, it may require a join between the logical tables.

[1] http://www.w3.org/ns/r2rml

SubjectMap, PredicateMap and ObjectMap are all Term Maps which is a function that generates an RDF term from the database. There are three ways of creating an RDF term, hence three types of Term Maps: 1) Constant-valued Term Map, 2) Column-valued Term Map, and 3) Template-valued Term Map. In `<#TriplesMap1>`, the SubjectMap is a Template-valued Term Map because it generates an IRI from a template and the value of the `"empno"` column. The PredicateMap is a Constant-valued Term Map because it generates a constant IRI: `ex:name`. The ObjectMap is a Column-valued Term Map because it generates a Literal from the value of the `"ename"` column.

There are three ways of generating RDF triples. *Table Triples* are triples describing an instance of a given class if a Subject Map has a `rr:class` associated to it. In this case, the subject is one of the three possible Term Maps, the predicate is `rdf:type` and the object is the given class. There are three possible rules to generate Table Triples. *Local Triples* are triples that are generated exclusively from a single logical table. Given that there are three ways of generating and RDF term for each the subject, predicate and object, there are 27 different possible cases to generating triples, hence 27 different rules. *Reference Triples* are triples that are generated for Referencing Object Maps, through a join conditions to another table Similar to Local Triples, there are also 27 distinct rules to generate triples. Figure 1 depicts al the possible 27 combinations.

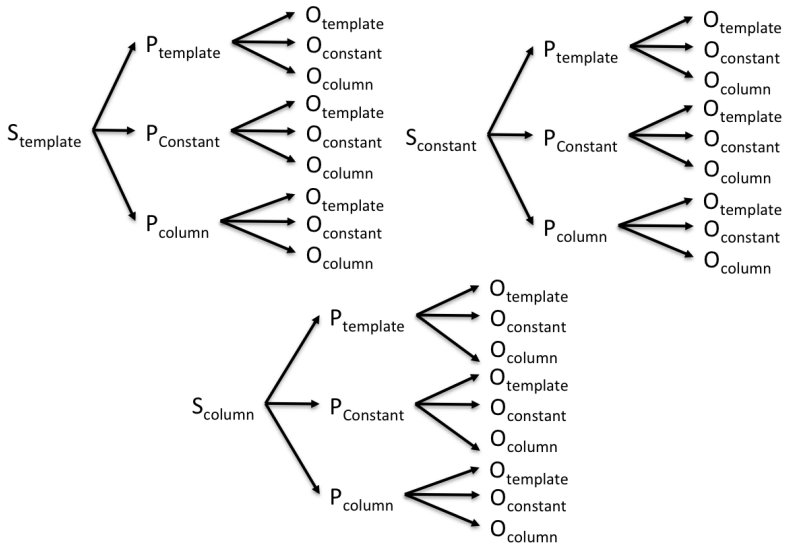

Fig. 1. The tree describes the space of 27 possible way of generating RDF Triples

Given that there are 3 rules to generate Table Triples, 27 rules to generate Local Triples and 27 rules to generate Reference Triples, the following theorem holds.

Theorem 1. *The total number of distinct Datalog rules which can be used to generate RDF triples from a table name is 57.*

3 Relationship with Direct Mapping

We now focus on a simple yet important subset of R2RML which we call $R2RML_{core}$ which consists of only three rules (out of the 57), one rule each for generating Table, Local and References Triples. In these rules, subjects are template-valued term maps, predicates are constant-valued term maps and objects are column-valued term maps. The following is an example of the a Datalog rule to generate Local Triples:

$$\begin{aligned}\textsc{Triple}(S,\,P,\,O) \leftarrow\ & \textsc{SubjectMap}(TM,\,SID),\,\textsc{PredicateObjectMap}(TM,\,POID),\\ & \textsc{SubjectTemplateValueTermMap}(SID,\,T,\,S),\\ & \textsc{PredicateConstantValueTermMap}(POID,\,P),\\ & \textsc{ObjectColumnValueTermMap}(POID,\,T,\,O)\end{aligned}$$

The motivation of $R2RML_{core}$ is two-fold. First, the W3C RDB2RDF Test Cases [5] consist of mainly subject template-valued term maps and predicate constant-valued term maps. Second, the anecdotal experience of the the author, who has implemented Ultrawrap [4], a system that supports R2RML, shows that subjects are usually template-valued term maps and predicates are constant-valued term maps.

Our hypothesis is that $R2RML_{core}$ is as expressive as the Direct Mapping, if views are allowed as input. We first would need to show how any mapping in $R2RML_{core}$ can be expressed in $\mathcal{DM}_{views}$. This means, that views and constraints would need to be created from the R2RML mapping. Additionally, there would need to be a mechanism to defining templates for IRIs. Subsequently, we would need to show how any $\mathcal{DM}_{views}$ mapping can be expressed as $R2RML_{core}$.

4 Conclusions

To the best of our knowledge, this is the first work presenting initial results on the formal study of the R2RML mapping language by defining its semantics using Datalog. and studying the relationship between R2RML and Direct Mapping. This is ongoing work where we are now considering additional features of R2RML such as SQL queries, languages and datatypes. We believe it is important to understand R2RML in order to know how better support users and build tools.

References

1. M. Arenas, A. Bertails, E. Prud'hommeaux, and J. Sequeda. Direct mapping of relational data to RDF. W3C Recomendation 27 September 2012, `http://www.w3.org/TR/rdb-direct-mapping/`.
2. S. Das, S. Sundara, and R. Cyganiak. R2RML: RDB to RDF mapping language. W3C Recomendation 27 September 2012, `http://www.w3.org/TR/r2rml/`.
3. J. Sequeda, M. Arenas, and D. P. Miranker. On directly mapping relational databases to rdf and owl. In *WWW*, pages 649–658, 2012.
4. J. Sequeda and D. P. Miranker. Ultrawrap: Sparql execution on relational data. *To appear in Journal of Web Semantics*, 2013.
5. B. Villazon-Terrazas and M. Hausenblas. R2RML and direct mapping test cases. W3C Working Group Note 14 August 2012, `http://www.w3.org/TR/rdb2rdf-test-cases/`.