# RDQuery* - Querying Relational Databases on-the-fly with RDF-QL

Cristian Pérez de Laborda Matthäus Zloch Stefan Conrad

Institute of Computer Science
Heinrich-Heine-Universität Düsseldorf
Universitätsstr. 1
D-40225 Düsseldorf, Germany
{perezdel, conrad}@cs.uni-duesseldorf.de, matthaeus.zloch@uni-duesseldorf.de

## ABSTRACT

One of the main drawbacks of the Semantic Web is the lack of semantically rich data, since most of the information is still stored in relational databases. We present RDQuery, a wrapper system which enables Semantic Web applications to access and query data actually stored in relational databases using their own built-in functionality. RDQuery automatically translates SPARQL and RDQL queries into SQL. The translation process is based on the Relational.OWL representation of relational databases and does not depend on the local schema or the underlying database management system.

## 1. INTRODUCTION

With his vision of a Semantic Web, Tim Berners-Lee inspired the database and knowledge representation communities to build up the next generation Web. Despite its sophisticated technologies like RDF [3] and OWL [4], the Semantic Web still has to face its major drawback, the lack of data. In fact, data is usually still stored in relational databases where it cannot be accessed directly by Semantic Web applications. Consequently, a well-defined mapping of relational to semantic data is required.

Although we can convert the schema of a relational database automatically into an RDF/OWL ontology and represent its data items as instances of this data source specific ontology [6], barely a database is static. Consequently, this data and schema extract may rapidly become outdated. Indeed, a schema or data extraction could be initiated, whenever a data or schema modification occurs within the database. Nevertheless, dealing with dynamic data sources, a direct access to such data sources would be preferable.

*RDQuery is published under GNU GPL and can be downloaded at `http://sourceforge.net/projects/rdquery/`.

*Posters and Demos of the 15th International Conference on Knowledge Engineering and Knowledge Management, EKAW 2006* Podebrady, Czech Republic, 2nd-6th October, 2006

## 2. RDQuery

RDQuery is a wrapper system which makes relational databases accessible for Semantic Web applications using an RDF query language (RDF-QL). RDQuery currently supports RDQL [9] and its successor SPARQL [8], which will hopefully be recommended soon by the W3C as the de facto standard query language for RDF. Nevertheless, RDQuery may easily be adapted to future developments adding specific parsers for other query languages. Figure 1 gives an overview of the RDQuery system architecture and depicts the path passed by a query until it reaches the relational database as its destination.

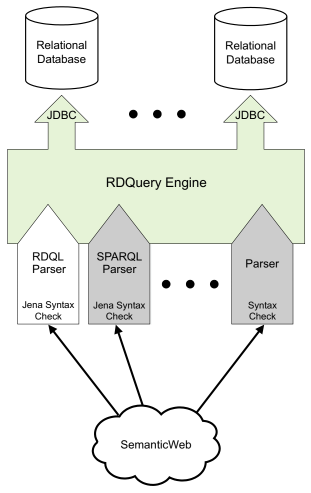

**Figure 1: RDQuery System Architecture**

First, the syntax of the query is validated and its relevant parts (e.g. the `WHERE` clause) are extracted using the built-in syntax checker of the JENA Framework [2]. Thereupon, the relevant parts of the query are once again parsed using an own JavaCC-based [1] grammar, in order to detect the properties of the query. Based on this information, the corresponding SQL query is built up. The resulting query is then executed and processed on the original database without having to translate the original database into a Relational.OWL representation, which thus only exists virtually.

The query translation is based on the results presented in [5] and [7], where we examined possible RDQL and SPARQL correspondents for the basic expressions of the relational algebra. Each of the five main operations $\{\sigma, \pi, \cup, -, \times\}$ of the relational algebra has characteristic appearances within a Semantic Web query. A selection, i.e. the `WHERE` part of an SQL query, corresponds to a triple similar to `{?x dbinst:TABLE.COLUMN 'value'}`, where `?x` is a free variable and `TABLE.COLUMN`, the column where the `value` shall be matched. Similar mappings can be given for the remaining operations of the relational algebra.

**Example:** The SPARQL query

```
CONSTRUCT {?a ?b ?c}
WHERE {{?a ?b ?c}.
      {?a rdf:type db:customers}.
      {?a db:customers.City 'Berlin'}.
   FILTER (?b=db:customers.ContactName)}
```

is automatically recognized by RDQuery as the SPARQL correspondent of a selection, followed by a projection. It thus translates the given query automatically into the following SQL query:

```
SELECT customers.ContactName
FROM   customers
WHERE  customers.City = "Berlin"
```

After the query execution on the original database, the user may opt for an RDF processable representation of the query result. This feature of RDQuery is especially important for Semantic Web applications using a query language, which is not closed within RDF (e.g. RDQL), where the result of such a query is not a valid RDF graph, but a list of possible variable bindings.

The whole query transformation process is identical for any relational database and does not depend on the local schema or the underlying database management system. Nevertheless, the queries have to match the instances of the Relational.OWL ontology. For a detailed description on how to simulate the main operators of the relational algebra in RDQL and SPARQL, we again refer to [5] and [7].

## 3. DEMONSTRATION

The presentation of the RDQuery system consists of two main parts. We will first introduce the Java-based user interface of RDQuery, where the users can interactively query relational databases using RDQL and SPARQL, the RDF query languages currently implemented in the system. The GUI enables the users to follow the translation process, to verify the generated SQL query, and to regard the result set returned from the database quickly. Furthermore, the users can access their own query history and get a general idea of the tables stored in the corresponding database. We will start with the simulation of the basic relational algebra operators and get to more complex queries containing several join operations. Thereby we will describe the basic functionality of RDQuery and explain in-depth, how the queries are parsed and translated into SQL.

In the second part of the presentation we will demonstrate how Semantic Web applications can use the API of RDQuery to query and access information actually stored in relational databases, as if this data would actually be a part of the Semantic Web. Additionally, we will show how to create a mapping from the relational model to an arbitrary ontology simply using RDQuery and SPARQL. For this purpose we will create a SPARQL query, which actually maps the data stored in a typical relational database to instances of the 'Friend of a Friend' (FOAF) ontology. This data is then accessed by an application to perform several reasoning tasks, e.g. find people within the same social network, working on related projects, living in the same city, or listening to similar music. These reasoning tasks are all processed by the application without actually noticing, that the data is stored in and modeled for a relational database and not for the Semantic Web.

To illustrate the independency of the translation process from the concrete database schema and the underlying database management system, all queries presented in both parts of the presentation will be performed using several databases stored in different database systems.

## 4. REFERENCES

[1] JavaCC - Java Compiler Compiler. `https://javacc.dev.java.net/`, 2006.

[2] Jena - A Semantic Web Framework for Java. `http://jena.sourceforge.net/`, 2006.

[3] F. Manola and E. Miller. RDF primer. `http://www.w3.org/TR/2004/REC-rdf-primer-20040210/`, 2004. W3C Recommendation.

[4] D. L. McGuinness and F. van Harmelen. OWL Web Ontology Language Overview. `http://www.w3.org/TR/2004/REC-owl-features-20040210/`, 2004. W3C Recommendation.

[5] C. Pérez de Laborda and S. Conrad. Querying Relational Databases with RDQL. In *Berliner XML Tage*, pages 161–172, 2005.

[6] C. Pérez de Laborda and S. Conrad. Relational.OWL - A Data and Schema Representation Format Based on OWL. In *Conceptual Modelling 2005, Second Asia-Pacific Conference on Conceptual Modelling (APCCM2005), Newcastle, NSW, Australia, January/February 2005*, volume 43 of *CRPIT*, pages 89–96. Australian Computer Society, 2005.

[7] C. Pérez de Laborda and S. Conrad. Bringing Relational Data into the Semantic Web using SPARQL and Relational.OWL. In *Semantic Web and Databases, Third International Workshop, SWDB 2006, Proceedings of the 22nd International Conference on Data Engineering Workshops, ICDE 2006, 3-7 April 2006, Atlanta, GA, USA*. IEEE Computer Society, 2006.

[8] E. Prud'hommeaux and A. Seaborne. SPARQL Query Language for RDF. `http://www.w3.org/TR/2006/CR-rdf-sparql-query-20060406/`, 2006. W3C Candidate Recommendation.

[9] A. Seaborne. RDQL - A Query Language for RDF. `http://www.w3.org/Submission/2004/SUBM-RDQL-20040109/`, 2004. W3C Member Submission.