Web Semantics: Science, Services and Agents on the World Wide Web 33 (2015) 141–169

Contents lists available at ScienceDirect

Web Semantics: Science, Services and Agents on the World Wide Web

journal homepage: www.elsevier.com/locate/websem

Efficient SPARQL-to-SQL with R2RML mappings

Mariano Rodríguez-Muro [1], Martin Rezk *

KRDB Research Centre, Free University of Bozen-Bolzano, Italy

ARTICLE INFO

Article history:
Received 17 October 2013
Received in revised form
9 January 2015
Accepted 13 March 2015
Available online 23 March 2015

Keywords:
OBDA
SPARQL
SQL
R2RML
RDF
RDB-to-RDF

ABSTRACT

Existing SPARQL-to-SQL translation techniques have limitations that reduce their robustness, efficiency and dependability. These limitations include the generation of inefficient or even incorrect SQL queries, lack of formal background, and poor implementations. Moreover, some of these techniques cannot be used over arbitrary DB schemas due to the lack of support for RDB to RDF mapping languages, such as R2RML. In this paper we present a technique (implemented in the -ontop- system) that tackles all these issues. We propose a formal approach for SPARQL-to-SQL translation that (i) generates efficient SQL by combining optimization techniques from the logic programming and SQL optimization fields; (ii) provides a well-defined specification of the SPARQL semantics used in the translation; and (iii) supports R2RML mappings over general relational schemas. We provide extensive benchmarks using the -ontop- system for Ontology Based Data Access (OBDA) and show that by using these techniques -ontop- is able to outperform well known SPARQL-to-SQL systems, as well as commercial triple stores, by several orders of magnitude.

1. Introduction

In an Ontology-Based Data Access (OBDA) framework, queries are posed over a conceptual layer and then translated into queries over the data layer. The conceptual layer is given in the form of an ontology that defines a shared vocabulary, and the data layer is in the form of one or more existing data sources. In this context, the most widespread data model for the conceptual layer and its matching query language are RDF (the Resource Description Framework) and SPARQL. Today, most enterprise data (data layer) is stored in relational databases, thus it is crucial that OBDA frameworks support RDB-to-RDF mappings. The new W3C standard for RDB-to-RDF mappings, R2RML [1], was created towards this goal.

R2RML mappings are used to expose relational databases as virtual RDF graphs. These virtual graphs can be *materialized*, generating RDF triples that can be used with RDF triple stores, or they can also be kept virtual and queried only during query execution. The virtual approach avoids the cost of materialization and (may) allow to profit from the more than 30 years maturity of relational systems (e.g., efficient query answering, security, robust transaction support, etc.). One of the most promising approaches for on-the-fly query answering over virtual RDF is query answering by query rewriting, that is, translating the original SPARQL query into an equivalent SQL query. This SQL query is then delegated to the DBMS for execution. In order to use the advantages provided by the DBMS, the query rewriting technique must produce "reasonable" SQL queries, that is, not excessively large or too complex to be efficiently optimized by the DB engine. Thus, the query rewriting technique needs to tackle two different issues: (i) a query translation problem that involves RDB-to-RDF mappings over arbitrary relational schemas, and (ii) a query optimization problem. There exist a number of systems and techniques related to this problem, such as the ones described in [2–4], However, each of these approaches has limitations that affect critical aspects of query answering over virtual RDF. These limitations include the generation of inefficient or even incorrect SQL queries, lack of formal background, and poor implementations. Moreover, some of them lack support for arbitrary DB schemas since they do not support RDB to RDF mapping languages, such as, R2RML.

The approach presented in this paper, and depicted in Fig. 1, deals with all the aforementioned issues. First, the SPARQL query and the R2RML mappings are translated into a Datalog program; the Datalog program is not meant to be executed, instead we view this program as a formal representation of the query and the mappings that we can manipulate and then transform into SQL. Second, we perform a number of structural and semantic optimizations on the Datalog program, including optimization with respect to database metadata. We do this by adapting well

* Corresponding author.
E-mail addresses: mrodrig@us.ibm.com (M. Rodríguez-Muro), mrezk@inf.unibz.it (M. Rezk).

[1] Mariano Rodríguez-Muro is currently working at IBM T.J. Watson Research Center.

http://dx.doi.org/10.1016/j.websem.2015.03.001
1570-8268/

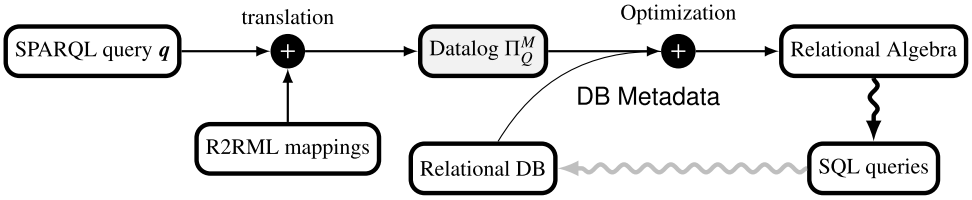

Fig. 1. Proposed approach for SPARQL to optimized SQL through Datalog with R2RML mappings.

known techniques for optimization of logic programs and SQL query optimization. Once the program has been optimized the final step is to translate it to relational algebra/SQL, and to execute it over the relational database. The technique presented in this paper is able to deal with all aspects of the translation, including URI and RDF Literal construction, RDF typing, and SQL optimization. This technique has been implemented in the -ontop- system[2] for OBDA, a mature open source system that is currently being used in a number of projects and that currently outperforms other similar systems, sometimes by several orders of magnitude. -ontop- is available as a SPARQL endpoint, as a OWLAPI and Sesame query engine and as a Protege 4 plugin.

The contributions of this paper are four: (i) a formal approach for SPARQL-to-SQL translation that generates efficient SQL by adapting and combining optimization techniques from logic programming the query optimization; (ii) a rule based formalization of R2RML mappings that can be integrated into our technique to support mappings to arbitrary database schemas; (iii) a discussion of the SQL features that are relevant in the context of SPARQL-to-SQL systems and that should be avoided to guarantee good performance in today's relational engines, together with experiments that validate these observations; (iv) an extensive evaluation comparing -ontop- with well known RDB2RDF systems and triple stores, showing that using the techniques presented here -ontop- can outperform them.

The rest of the paper is organized as follows: In Section 2 we briefly survey other works related to SPARQL–SQL translation. In Section 3 we introduce the necessary background. In Section 4 we present the core technique for translation of SPARQL to SQL. In Section 5 we show how to incorporate R2RML mappings into our approach. In Section 6 we provide a discussion on the SQL features that degrade performance of query execution. In Section 7 we describe how to optimize our technique with respect to the issues discussed in Section 7 by applying techniques from logic programming and SQL query optimization. In Section 8 we provide an evaluation of the performance of the technique. In Section 9 we conclude the paper. All proofs are given in the Appendix.

2. Related work

In this section we briefly survey related works regarding SPARQL query answering. We focus on two different but closely related topics: RDF stores and SPARQL to SQL translations.

RDF stores. Several RDF stores, such as RStar [5] and Virtuoso 6.1 [6], use a single table to store triples. This approach has the advantage that it is intuitive, flexible, and the mappings between the conceptual and data layer (if needed) are trivial. On the other hand such approach cannot use the known optimizations developed for normalized relational DBs—many of which are currently used in -ontop-. Our approach uses existing relational databases together with R2RML mappings to obtain a virtual representation of the RDF graph. Virtuoso 7 also provides column-wise compressed storage [7], which may be much faster than traditional row stores. -ontop- (and any other general SPARQL-to-SQL techniques) may also benefit from the performance of column-stores that support SQL, e.g., MonetDB.

In addition to the RDF stores mentioned above, we explore the commercial RDF stores Stardog and OWLIM more in detail in Section 8.

Stardog[3] is a commercial RDF database developed by Clark & Parsia that supports SPARQL 1.1. Although it is a triplestore, Stardog uses query-rewriting techniques [8] to perform reasoning.

OWLIM[4] (renamed GraphDB) is a commercial triplestore developed by Ontotext that allows to query, reason, and update RDF data.

SPARQL-to-SQL. Regarding SPARQL-to-SQL translations, there have been several approaches in the literature [2–4]. In addition one can also include here translations from SPARQL to Datalog [9–11] given that: (i) SPARQL (under set semantics) has the same expressive power of non-recursive safe Datalog with default negation [11]; and (ii) any recursion-free safe Datalog program is equivalent to an SQL query [12]. The work in [9] extends and improve the ones in [10,11] by modeling SPARQL 1.1 (under bag semantics, where duplicates are allowed), modeling non-safe queries,[5] and modeling the W3C standard semantics of the SPARQL Optional operator. Since [10] was published before the publication of the SPARQL standard specification, the semantics presented there is not compliant with the current SPARQL semantics specification. To keep the presentation simple, in this paper we will use the "academic" set semantics of SPARQL, as in [13,10,11]. We build and re-use several results from the works mentioned above, however we extend this line of research in several ways. First, we include R2RML mappings in the picture; second, we provide a (concrete) SQL translation from the Datalog program obtained from the input query; and third, we optimize and evaluate the performance of this approach. It is worth noticing that not any SQL query correctly translated from the Datalog program is acceptable, since (i) one has to deal the mismatch between types in SPARQL and SQL; and (ii) the syntactic form of SQL queries can severely affect their performance.

In [2] the authors introduce an SPARQL-to-SQL translation technique that focuses in the generation of efficient SQL queries. By efficient queries, we mean queries that result in efficient query plans when executed by the DB engine. The work in [2] has an objective similar to ours, however, it is different in that [2] covers a smaller fragment of SPARQL 1.0 and the technique lacks formal semantics, i.e., lacks proofs of soundness and completeness of the approach. In addition, the technique in [2] relies on a mapping language that lacks support for URI templates and is less expressive than R2RML.

In [3] the authors propose a translation function that takes a query and two many-to-one mappings: (i) a mapping between the triples and the tables, and (ii) a mapping between pairs of the form

[2] http://ontop.inf.unibz.it/.

[3] http://stardog.com/.

[4] http://www.ontotext.com/.

[5] Non-safe queries: queries with filter expressions that mention variables that do not occur in the graph pattern being filtered.