Relational Databases as Semantic Web Endpoints

Matthias Hert

Department of Informatics, University of Zurich,
Binzmuehlestrasse 14, CH-8050 Zurich, Switzerland
hert@ifi.uzh.ch

Abstract. This proposal explores the promotion of existing relational databases to Semantic Web Endpoints. It presents the benefits of ontology-based read and write access to existing relational data as well as the need for specialized, scalable reasoning over that data. We introduce our approach for translating SPARQL/Update operations to SQL, describe how scalable reasoning can be realized by using the power of the database system, and outline two case studies for evaluating our approach.

1 Motivation

Relational Databases (RDBs) are used in most current enterprise environments to store and manage data. While RDBs are well suited to handle large amounts of data they were not designed to preserve the semantics of that data. This is especially relevant if data from different parties needs to be exchanged or integrated.

The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries. Although invented for the Web, these technologies have proven to be useful in other domains as well. Recent research[1] has shown that an ontology-based access to relational data reduces the barriers for data exchange and integration. The expressive and formal semantics increases the value of the existing data and enables new applications on that data.

Ontology-based access to relational data can be achieved in two different ways: (1) All relational data is converted at once to a Semantic Web format, eliminating the need for the original database or (2) the relational database is preserved and RDF data is provided on demand through a mediator that translates between Semantic Web requests and SQL. The one-time conversion approach is often not feasible due to the following reasons: (1) *Scalability* of RDF storage systems is still a topic of research, while relational databases were highly optimized over the past thirty years; (2) The existence of *legacy applications* prohibits the elimination of the relational database; (3) Mirroring the data in an RDF store does not only increase the total data size but also leads to *consistency* issues if the original data or the RDF mirror is updated. In contrast, the on

[1] e.g. the Linked Data project at http://linkeddata.org

demand provision has the drawback that it does not constitute a native Semantic Web data source and therefore leaves behind a conceptual gap between the two worlds of relational data and the Semantic Web.

The goal of this thesis is to promote relational databases to Semantic Web endpoints that enable ontology-based read/write access and offer scalable reasoning support. This is achieved with a wrapper in front of the database that mediates between the relational and the Semantic Web representations. Therefore, we profit from the strength of both technologies, the scalability in terms of response time and database size as well as the explicit, formal semantics that enable data exchange, integration, and reasoning.

2 State of the Art

Ontology-based Access to RDBs: The W3C developed SPARQL [1] as the standard query language for RDF which also provides a protocol [2] for remote read access to RDF data on so-called SPARQL endpoints. Relational.OWL [3] defines an ontology to represent relational structures (tables, attributes, keys, etc.) in RDF. This enables Semantic Web technologies to access the relational data, but the structure of the RDF data is limited to the database schema. RDQuery [4] enables SPARQL-based access to relational data by translating the queries to SQL via mappings expressed in Relational.OWL. D2R Server [5] wraps existing RDBs into SPARQL endpoints based on its own declarative mapping language D2R MAP [6] that maps database tables and attributes to corresponding classes and properties in an user-defined ontology. This allows to reuse existing vocabularies and avoids the exposure of the database schema to the ontology representation. OpenLink Virtuoso[2] is a commercial database system featuring RDF Views that allow querying the relational data with SPARQL. The World Wide Web Consortium (W3C) has recognized the importance of mapping relational data to the Semantic Web by starting the RDB2RDF incubator group (XG)[3] to investigate the need for standardization. The XG recommends that the W3C starts a working group to produce a standard RDB to RDF mapping language.

SPARQL/Update is a language to express updates to an RDF store that reuses the syntax of SPARQL. It supports operations to insert and delete RDF data in existing graphs as well as create and remove graphs in an RDF store [7]. To the best of our knowledge, none of the existing RDB to RDF mapping approaches support data manipulations via SPARQL/Update.

Reasoning Over Data in RDBs: Reasoning over large ABoxes as found in mapped RDBs pose a scalability limitation to current reasoners. OWLDB [8] aims to overcome this by offloading the reasoning task to the relational database system. The inference rules of OWL DL are translated to equivalent SQL queries that are executed to materialize and store the inference inside the database.

[2] `http://virtuoso.openlinksw.com`
[3] `http://esw.w3.org/topic/Rdb2RdfXG`

MASTRO-I [9] is a data integration framework that uses DL-Lite [10] ontologies as the global schema. The family of DL-Lite description logics were designed to enable a maximum of expressiveness while remaining within the LOGSPACE data complexity that allows to perform reasoning as standard SQL query evaluation.

3 Proposed Approach

Current approaches for mapping relational data to the Semantic Web as described in Section 2 are limited to read-only access. The relational data is immutable to the Semantic Web layer, which means modifications to the RDF data cannot be propagated back into the RDB. However, applications of Semantic Web technologies apart from the Web domain (e.g. in enterprise environments) would benefit from the possibility of persistent modifications to RDF data. Furthermore, mapping industry-size relational data to RDF generates a large amount of instance data which blasts current in-memory reasoning approaches.

In this PhD thesis, we strive for a promotion of existing relational databases to Semantic Web Endpoints, i.e., ontology-based read and write access to RDBs as well as support for scalable reasoning over the relational data. The use of relational databases as the storage layer should ideally be transparent to the developer of ontology-based applications, however due to the conceptual differences (relational versus triple-orientied representation of the data) this cannot be realized in general. Our goal is to achieve a maximum of transparency while providing support to the developer in the cases where the underlying storage system is exposed (e.g. if a SPARQL/Update insert operation does not contain enough data to create a legal tuple in a database table). This will be achieved through a mediator in front of existing RDBs that, based on mappings, translates requests, results, and errors between the relational representation and the Semantic Web. Ontology-based read access will be provided through a SPARQL endpoint and write access through a corresponding SPARQL/Update endpoint. This enables a clear separation between data and applications, independence from any specific Semantic Web framework, and the possibility to remotely access the data. As shown in [11], SPARQL and relational algebra have equivalent expressive power, which means SPARQL queries can be translated completely to SQL for efficient execution by the database engine. We will investigate how the SPARQL/Update language can be translated to the SQL data manipulation language (DML) and in particular how each of the SPARQL/Update operations is mapped to SQL DML with respect to our application scenario. We will build a prototype implementation that can be set up in front of an existing database to enable ontology-based access to the relational data.

Reasoners that need to materialize all data in main memory are not feasible in our application scenario as the instance data in the database exceeds the main memory size. In addition, the support for data updates renders approaches based on pre-computing and caching of the inference unsuitable as it would need to be validated or recomputed after every update. Therefore, efficient reasoning needs

to be performed at query time to avoid scalability problems. The OWL 2 QL profile,[4] which is based on the DL-Lite description logic, was specifically designed for this kind of application as it allows to reduce reasoning to standard SQL query evaluation. We plan to integrate DL-Lite reasoning through query rewriting in our approach.

4 Evaluation Strategy

We plan to deploy our approach in two case studies. The first is derived from an industry project where we develop an ontology-based inventory management and verification system. Our industry partner employs relational databases to store data about individual parts and product configurations. This relational data is the foundation of many legacy applications that can neither be replaced nor adapted in the short-run, resulting in the absolute need to preserve the RDBs. The goal of the project is to establish a semantic layer on top of the existing systems that verifies new product configurations according to a set of rules. The layer is realized with Semantic Web technologies and implements our approach for read/write access to the data as well as for reasoning.

The second case study is motivated in the domain of software quality analysis where the application of Semantic Web technologies recently gained popularity. Software analysis builds on different data sets (source code repositories, bug tracking systems, etc.) typically stored in individual relational databases. Semantic Web technologies provide an ideal mean to integrate this data and reasoning is used for certain analyses. We plan to integrate our approach into the existing tool set as a semantic interface to the source data, enabling ontology-based and existing analysis tools to (co-)operate on the same data.

Furthermore, we will evaluate the performance and expressiveness of our approach with the Berlin SPARQL Benchmark (BSBM).[5] BSBM is currently focused on SPARQL queries, therefore we plan to develop a set of representative SPARQL/Update operations to cover all features of our approach.

4.1 Current State of Our Research

In a first step, we defined a simple mapping language to bridge the conceptual gap between the relational representation of the data and an ontology. Mappings defined in this language not only contain the correspondences between database and ontology terms but also all structural information of the database schema needed to support write operations (e.g. primary keys, foreign key relationships, etc.). Based on this mapping language, we implemented a first prototype that supports inserting and deleting RDF data through SPARQL/Update. The prototype exploits the information stored in the mapping to support even complex insert operations spanning multiple tables with foreign key relationships. We also investigated how the SPARQL/Update operations are mapped to semantically equivalent SQL DML queries and their required order of execution.

[4] `http://www.w3.org/TR/owl2-profiles/#OWL_2_QL`

[5] `http://www4.wiwiss.fu-berlin.de/bizer/BerlinSPARQLBenchmark/`

5 Conclusion

In this proposal, we outlined the benefits that Semantic Web technologies can have in application scenarios where data is stored in relational databases. We presented the need for full read and write access as well as support for scalable reasoning and the advantages the preserving of the RDB has over the one-time data conversion. We introduced our approach that will enable ontology-based read and write access to relational data through SPARQL+SPARQL/Update endpoints as well as provide support for scalable reasoning performed at query time to use the power of the database engine and to avoid problems occurring from data updates. We further described where we plan to apply our approach for evaluation purposes.

References

1. Prud'hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF. `http://www.w3.org/TR/rdf-sparql-query/` (2008)
2. Clark, K.G., Feigenbaum, L., Torres, E.: SPARQL Protocol for RDF. `http://www.w3.org/TR/rdf-sparql-protocol/` (2008)
3. de Laborda, C.P., Conrad, S.: Relational.OWL - A Data and Schema Representation Format Based on OWL. In: Proceedings of the 2nd Asia-Pacific Conference on Conceptual Modelling. (2005)
4. de Laborda, C.P., Zloch, M., Conrad, S.: RDQuery - Querying Relational Databases on-the-fly with RDF-QL. In: Proceedings of the 15th International Conference on Knowledge Engineering and Knowledge Management. (2006)
5. Bizer, C., Cyganiak, R.: D2R Server - Publishing Releational Databases on the Semantic Web. In: Proceedings of the 5th International Semantic Web Conference. (2006)
6. Bizer, C.: D2R MAP - A Database to RDF Mapping Language. In: Proceedings of the 12th International World Wide Web Conference. (2003)
7. Seaborne, A., Manjunath, G., Bizer, C., Breslin, J., Das, S., Davis, I., Harris, S., Idehen, K., Corby, O., Kjernsmo, K., Nowack, B.: SPARQL Update - A language for updating RDF graphs. `http://www.w3.org/Submission/2008/SUBM-SPARQL-Update-20080715/` (2008)
8. Auer, S., Ives, Z.: Integrating Ontologies and Relational Data. Technical Report MS-CIS-07-24, Department of Computer and Information Science, University of Pennsylvania (2007)
9. Calvanese, D., Giacomo, G.D., Lembo, D., Lenzerini, M., Poggi, A., Rosati, R.: Mastro-i: Efficient Integration of Relational Data through DL Ontologies. In: Proceedings of the 20th International Workshop on Description Logics. (2007)
10. Calvanese, D., Giacomo, G.D., Lembo, D., Lenzerini, M., Rosati, R.: Tractable Reasoning and Efficient Query Answering in Description Logic: The DL-Lite Family. Journal of Automated Reasoning (2007)
11. Angles, R., Gutierrez, C.: The Expressive Power of SPARQL. In: Proceedings of the 7th International Semantic Web Conference. (2008)