

# Efficiently Querying Relational Databases Using OWL and SWRL

Martin O'Connor, Ravi Shankar, Samson Tu, Csongor Nyulas, Amar Das, Mark  
Musen

Stanford Medical Informatics, Stanford University School of Medicine,  
Stanford, CA 94305 [martin.oconnor@stanford.edu](mailto:martin.oconnor@stanford.edu)

**Abstract.** For the foreseeable future, most data will continue to be stored in relational databases. To work with these data in ontology-based applications, tools and techniques that bridge the two models are required. Mapping all relational data to ontology instances is often not practical so dynamic data access approaches are typically employed, though these approaches can still suffer from scalability problems. The use of rules with these systems presents an opportunity to employ optimization techniques that can significantly reduce the amount of data transferred from databases. To illustrate this premise, we have developed tools that allow direct access to relational data from OWL applications. We express these data requirements by using extensions to OWL's rule language SWRL. A variety of optimization techniques ensure that this process is efficient and scales to large data sets.

## 1 Introduction

As ontology development tools have been increasingly used to address real-world problems, scalability has become an important topic [4]. Initially, ontologies were stored in flat files and fully loaded into application memory when in use. This approach worked well for small ontologies, but it did not scale well. One of the first approaches to this problem was to store ontology information directly in relational databases. However, this approach generally used application-specific storage formats, and ontologies stored this way could not be used easily by other tools. More recent approaches have focused on triple stores [<http://simile.mit.edu/reports/stores/>], which use native representation of RDF triples to store ontologies. Triple stores are analogous to relational database management systems and provide efficient storage and retrieval of ontology information. RDF query languages like RDQL [<http://www.w3.org/Submission/RDQL/>] and SPARQL [<http://www.w3.org/TR/rdf-sparql-query/>] can provide SQL-like query functionality on triple stores. OWL ontologies can be stored in triple-store back ends without loss of semantics.

One approach to the problem would be to statically map a relational database to a triple-store. This approach suffers from several shortcomings, however. First, there is an issue of data duplication. Furthermore, there are questions about how frequently to update triple stores to reflect changes in associated relational database. Knowledge-driven applications requiring up-to-date information require frequent synchronization,

which may be cumbersome and problematic. And, of course, supporting knowledge-driven updates, means that synchronization issues arise in the reverse direction.

Ideally, knowledge-driven data requests would retrieve data from live relational databases. This approach requires automatic or semi-automatic dynamic mapping between relational databases and triple-based formats. It also requires a software layer to rewrite knowledge-level queries into SQL-queries for retrieving required data from a database. Further reasoning with retrieved knowledge could be performed in memory. If updates were allowed, the reverse transformation would also be supported. Many recent systems have implemented this approach or variants of it [6,7].

## 2 Implementation

To support knowledge-driven querying of relational databases, we have developed tools to map data dynamically from relational databases to concepts described in an OWL ontology. Our tools make extensive use of OWL's rule language SWRL [2]. SWRL is used both to specify the OWL-to-relational mapping and to provide a knowledge-level query interface to the system.

This work extends ontology development technologies that we have been producing over the past decade. In particular, we have used Protégé-OWL [5], an open source framework that provides a suite of tools for constructing OWL ontologies and knowledge-based applications, and an associated development environment called SWRLTab for working with SWRL rules [1]. SWRLTab supports the editing and execution of SWRL rules. It also supports the incorporation of user-defined libraries of methods—called *built-ins*—that can be used in rules. Several standard libraries are provided, including implementations for the core SWRL built-ins defined by the SWRL submission [2], a temporal library that can be used to reason with temporal information in SWRL rules, and libraries that allow Abox and Tbox querying. A query library is also provided, and can be used to effectively turn SWRL into a query language.

We used the query extensions provided by SWRLTab to develop a set of tools that can be used to perform knowledge-level queries on data stored in a relational database. We have devised an array of optimization strategies to improve the performance of the underlying relational-to-ontology mapping process. Our primary goal is to offload as much work as possible to the underlying RDBMS by exploiting knowledge of SWRL rules as well as additional information provided by a rule base author. A secondary goal is to reduce the amount of data retrieved from databases during rule processing.

Our optimization strategies include: (1) adding annotations to built-ins to describe the nature of the operations they perform and then using them to rewrite the underlying data retrieval SQL queries to exclude unnecessary data; (2) annotating individual SWRL rules to describe their major 'axis of evaluation' and using these annotated rules to exclude as much unneeded data as possible; (3) using the same annotation technique at the rule base level to control overall relational data access during rule evaluation; (4) making rule engine optimizations by providing a late

binding mechanism for mapped data so that information is retrieved only when needed; and finally, (5) using standard database optimization techniques, which are then enhanced by a knowledge-driven process to create database views based on the expected access pattern to data.

### 3 Conclusions

We have implemented an efficient dynamic OWL-to-relational mapping method and used SWRL to provide a high-level language that uses these mappings. An important benefit of our approach is that it allows knowledge-driven applications to work directly with relational data. We believe that SWRL provides a rich high-level language to specify the data requirements of these applications. In conjunction with relational-to-OWL mapping technology, it can also serve as an efficient means of dealing with legacy relational data.

**Acknowledgements.** This work was supported in part by the Centers for Disease Control and Prevention under grant number SPO-34603.

### References

1. O'Connor M.J., Knublauch, H., Tu, S.W., Grossof, B., Dean, M., Grosso, W.E., Musen, M.A. Supporting Rule System Interoperability on the Semantic Web with SWRL. Fourth International Semantic Web Conference Galway, Ireland (2005).
2. SWRL Submission: <http://www.w3.org/Submission/SWRL/>
3. Crubezy, M., O'Connor, M.J., Buckeridge, D.L., Pincus, Z.S., Musen, M.A. Ontology-Centered Syndromic Surveillance for Bioterrorism. *IEEE Intelligent Systems*, 20(5):26-35 (2005).
4. Vanessa Lopez, Marta Sabou, Enrico Motta PowerMap: Mapping the Real Semantic Web on the Fly. 5th International Semantic Web Conference, Athens, GA, USA (2006).
5. Knublauch, H. Ferguson, R.W., Noy, N.F. and Musen, M.A. The Protégé OWL Plugin: An Open Development Environment for Semantic Web applications Proc Third ISWC (ISWC 2004), Hiroshima, Japan, pp. 229-243 (2004)
6. Christian Bizer. D2R MAP: A Database to RDF Mapping Language. WWW 2003, Budapest, Hungary (2003).
7. Huajun Chen, Yimin Wang, Heng Wang, Yuxin Mao, Jinmin Tang, Cunyin Zhou, Ainin Yin, and Zhaohui Wu. Towards a Semantic Web of Relational Databases: a Practical Semantic Toolkit and an In-Use Case from Traditional Chinese Medicine. Fifth International Semantic Web Conference (2006).