RDOTE - Transforming Relational Databases into Semantic Web Data

Konstantinos N. Vavliakis[1,2], Theofanis K. Grollios[1], and Pericles A. Mitkas[1,2]

[1]Electrical and Computer Engineering Department, Aristotle University of Thessaloniki, GR541 24, Thessaloniki, Greece
[2]Informatics and Telematics Institute, CERTH, GR570 01, Thessaloniki, Greece
kvavliak@issel.ee.auth.gr,fgroll@auth.gr,mitkas@eng.auth.gr

Abstract. During the last decade, there has been intense research and development in creating methodologies and tools able to map Relational Databases with the Resource Description Framework. Although some systems have gained wider acceptance in the Semantic Web community, they either require users to learn a declarative language for encoding mappings, or have limited expressivity.
Thereupon we present *RDOTE*, a framework for easily transporting data residing in Relational Databases into the Semantic Web. *RDOTE* is available under GNU/GPL license and provides friendly graphical interfaces, as well as enough expressivity for creating custom RDF dumps.

Keywords: Relational Databases to Ontology Transformation, RDB2RDF, RDF Dump

1 Introduction

The large volume of data residing in relational databases led to the creation of systems for instantiating ontology schemata using relational information, with some, like D2RQ, gaining wider acceptance in the Semantic Web community. Unfortunately, all these tools require advanced user skills as they are either built upon complex declarative languages and lack friendly user interfaces or they provide GUIs with limited expressivity.

We present *RDOTE*, a system able to map multiple Relational Databases (RDB) into different ontology schemata and integrate them into a single ontology file. *RDOTE* is online[1,2] available under the GNU/GPL license and provides drag 'n drop operations, drop down lists and recommendation mechanisms, that allow users to define all the necessary mappings between tables/columns and classes/properties, in order to create domain-specific mappings according to a selected ontology schema.

The main contribution of *RDOTE* towards Semantic Web researchers is two-fold: a) it can transform datasets currently residing in (one or many) Relational

[1] http://sourceforge.net/projects/rdote/
[2] http://www.youtube.com/watch?v=pk7izhFeuf0

Databases into Semantic Web data through a friendly interface and b) it can quickly instantiate an ontology schema with real data, allowing easy experimentation with large ontology datasets.

2 Background Information - Relevant Work

Throughout related bibliography, one may find numerous methodologies and systems for publishing data residing in Relational Databases into the Semantic Web. D2RQ [1] is the mostly embraced by the Semantic Web Community. D2RQ offers a powerful declarative language for mapping Relational Databases to ontologies, nevertheless, no graphical user interfaces are provided. Less prevalent systems, like RDB2Onto [4] are highly configurable too but they also lack friendly user interfaces. On the other hand, systems like SquirrelRDF [6] offer a simplistic approach to publish RDF data from Relational Databases (still absent GUI), which may not be expressive enough in case of complex databases/mappings. Dartgrid [2] and ODEMapster plugin for the NeOn Toolkit [5] currently offer a graphical user interface, but they have limited scope and applicability, as well as limited expressivity compared to *RDOTE*. Finally Virtuoso RDF View [3] comes with a graphical user interface, which is available only for the Virtuoso database, rather than other popular relational DBMS.

3 *RDOTE* Functionality

An overview of *RDOTE*'s functionality is depicted in Figure 1. *RDOTE* lies in the category of Domain Semantics-driven Mapping Generation tools, all mappings are formulated via graphical user interfaces and stored in text files. *RDOTE* has a generalized application domain and is currently applied on two test cases: one in the bibliographic domain and one in the art object documentation domain, available online.

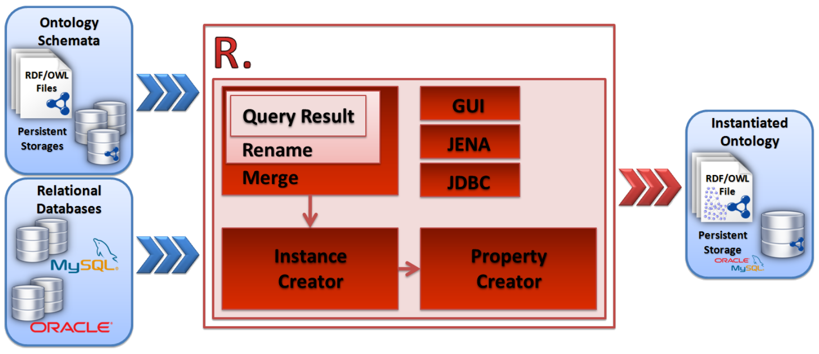

Fig. 1. *RDOTE* Architecture

For the complete transformation of a RDB to an ontology, one has to take the following steps:

1. Connect to the desired RDBMS and load the respective ontology schema: Users first have to connect to the desired RDBs (MySQL and Oracle supported) and load an ontology schema containing the TBox of the desired domain. *RDOTE* can load ontology files in various formats (RDF/XML, N3, N-triples), and/or persistent ontologies. *RDOTE*'s GUI depicts the TBox in a tree representation form, as well as the ABox of the loaded ontology. *RDOTE* also presents the tables contained in each connected RDB and the respective columns.
2. Write SQL queries that select the desired tuples to be processed as an RDF graph: Each query represents a result set that is used to populate an Ontology Class or acts as a dataset of literals for linking instances of a class with a datatype property. For unambiguous creation of instances, the primary key is required in the URI, whereas the user selected columns are required for populating datatype properties. Thus, along with the user defined SQL query, *RDOTE* automatically selects the primary keys that have been defined for the tables participating in the SQL query.
3. Define renaming options and merging strings in case there are queries containing multiple selected columns: This optional step is responsible for renaming result sets, based on regular expression pattern matching. Each tuple matching the defined regular expression will be renamed when used as a literal. Moreover, this step allows users to define merging strings for SQL queries that select data from multiple columns.
4. Connect queries defined in Step 2 with ontology classes (Class Mappings): Next the Class Mappings that are responsible for the creation of all the ontology instances have to be defined. SQL queries are connected to ontology classes and in this way for each tuple of a SQL query, an instance of the respective class is created. In case one wishes to use the actual data instead of just creating URIs, *RDOTE* provides the possibility of copying the actual tuple information into any datatype property of the ontology schema.
5. Define conditional links of Class Mappings with other Class Mappings via object properties or, in case of datatype properties, with SQL queries: For each Property Mapping, users can select a Class Mapping, drag 'n drop a property from the ontology tree and in case of an object property, select a second Class Mapping, whereas in the case of datatype property, select a SQL query. Next a join condition between the first Class Mapping and the second Class Mapping/SQL Query has to be defined, either by the user or *RDOTE*, which can propose possible join conditions as calculated by a greedy graph algorithm traversing the SQL schema. Users can then insert any other conditional restriction SQL92 supports.
6. Instantiate the ontology schema and store it either in text file format or in a persistent repository: Finally users can launch *RDOTE*'s engine, which instantiates the loaded ontology schema and creates an RDF dump of the selected relational data which are stored either in text file format (RDF/XML,

N3, N-triples) or in a persistent storage format (MySQL and Oracle). *RDOTE* first creates all the instances and then it links them with object properties or adds literals using datatype properties.

All steps previously described are realized through *RDOTE*'s friendly interfaces (screenshots are available in *RDOTE*'s homepage), while at any step in the mapping definition process, users can save their project, that is all their mappings and database/ontology connections, and resume later.

The maximum supported RDF dump size *RDOTE* can create depends on the output format and naturally on the numbers of triples. In the case of text format, this size is also significantly dependent on Java maximum heap size. In our case, for maximum Java heap size of 1024MB, *RDOTE* successfully created 2 million triples in RDF/XML-ABBREV format in less than five minutes. Memory limitations do not exist in the case of persistent storage, but time limitations begin to emerge. In this case, *RDOTE* managed to create 10 million triples in less than five hours.

4 Conclusions - Future Work

RDOTE provides the Semantic Web research community and domain experts with the necessary means for easily enriching Ontology schemata with the vast amount of data currently residing in relational databases. It also enables quick instantiations of new ontology schemata for testing and experimentation. By allowing easy transportation of legacy data into semantically aware data structures, *RDOTE* aspires to bring the Semantic Web vision one step closer.

RDOTE is constantly updated. In the near future we expect to incorporate an export/import mechanism for D2RQ compliant mapping files, as well as a query builder graphical user interface together with complementary interfaces and mechanisms that will facilitate and hasten the mapping creation process even further. Finally, further evaluation and testing on large datasets is pending.

References

1. Bizer, C., Seaborne, A.: D2rq - treating non-rdf databases as virtual rdf graphs. In: Poster presented in Internatiotal Semantic Web Conference 2004 (November 2004)
2. Chen, H., Wu, Z.: Dartgrid iii: A semantic grid toolkit for data integration. In: First International Conference on Semantics, Knowledge and Grid. p. 12. IEEE Computer Society (2005)
3. Erling, O., Mikhailov, I.: Rdf support in the virtuoso dbms. In: Conference on Social Semantic Web. LNI, vol. 113, pp. 59–68. GI (2007)
4. Laclavk, M.: Rdb2onto: Relational database data to ontology individual mapping in: Tools for acquisition, organisation and presenting of information and knowledge. Tech. rep. (2008)
5. Rodriguez, J.B., Gómez-Pérez, A.: Upgrading relational legacy data to the semantic web. In: WWW '06: Proceedings of the 15th international conference on World Wide Web. pp. 1069–1070. ACM, New York, NY, USA (2006)
6. Steer, D.: Squirrelrdf. Tech. rep., HP (2006)