Retaining Semantics in Relational Databases by Mapping them to RDF

Madhav Krishna
Netaji Subhas Institute of Technology, Delhi, India
madhkrish@gmail.com

Abstract

Across various domains, the relational model for databases is employed extensively for the purpose of storing, managing and retrieving data. The translation, however, of data modeling schemes such as entity-relationship (ER) diagrams into relational databases is accompanied by a certain loss of semantics as expressed by these models. With the growth of the semantic web, an effort has to be made in the direction of expressing these relational databases in a form and language that may be machine processable and such that they make the semantics as expressed by the database more explicit. This paper presents a methodology for expressing relational databases in the Resource Description Framework (RDF) language, which has even been referred to as the language of the Semantic Web.

1. Introduction

In this paper, the focus is on mapping relational databases onto an RDF representation, keeping in mind the need for an efficient methodology for a semantically rich representation of data. RDF is a machine-readable language that can be employed to describe various classes of objects (resources), their properties and the relationships among them through the use of *statements*. In order to express such a description, RDF uses a number of *triples* (*n3* notation) [1]:
<Subject> <Predicate> <Object>
The above is equivalent to the triad:
<Resource> <Property> <Property Value>

This triple is expressed in the form of an RDF statement which can easily be represented as a directed graph in which a node represents the subject, another node for the object and a connecting arc for the predicate, directed from the subject node towards the object node. A resource, physical or abstract, can be defined as per [2] as a "conceptual mapping to an entity or a set of entities, not necessarily the entity which corresponds to that mapping at any particular instance in time. Thus, a resource can remain constant even when its content - the entities, to which it currently corresponds - changes over time, provided that the conceptual mapping is not changed in the process." What gives RDF its uniqueness in representing relationships between resources is that RDF is specific to use on the web; it utilizes Uniform Resource Identifiers (URIs) to represent resources and properties, and in some cases, property values as well. A URI is a string of characters identifying an abstract or physical resource. The "uniformity" [2] provides many advantages such as:
a) Making the use of different types of resource identifiers (that is, when the mechanisms used to access these resources differ) in the same context becomes possible.
b) Uniformity is introduced across different types of resource identifiers, in the semantics of common syntactical conventions.
c) The introduction of new types of resource identifiers also becomes possible, without interfering with existing identifiers which can be used in many different contexts.
d) It becomes possible for new applications to utilise a pre-existing set of resource identifiers.

Since URIs are used instead of words to identify resources, predicates and objects in statements, a set of URIs is referred to as a *vocabulary*. Such a set of URIs in a vocabulary is often defined for a certain predefined, common purpose. The URIs in such vocabularies are often referred to as URI references or URIrefs. URIrefs support Unicode, thus allowing them to be written in different languages. Often, the URIrefs in vocabularies are organized such that they share a common namespace. Such a namespace is typically a URIref under the control of the organization which defines the vocabulary. For instance, an organization such as "forexample" might define a vocabulary consisting of URIrefs starting with the prefix *http://www.forexample.org/defineterms* for terms that it may use for naming entities in its databases -

"employee-name" or "date-of-birth". In order to be able to define the vocabularies (terms) which RDF users intend to use in RDF statements, and specifically to indicate that they are describing specific kinds or classes of resources, RDF Schema or OWL (provides additional formal semantics) has to be used.

Much of the related research work attempting to express data in a machine readable form, while utilizing terms defined ontologically, has concentrated on the use of UML (e.g. [3], [4]) for this purpose. UML has many disadvantages when it comes to dealing with the problem at hand. It is specified by a combination of English, the object constraint language (OCL, which is basically a set of well formed rules) and the language itself and is therefore, far from being precise or unambiguous. Apart from lacking formal semantics, UML possesses the undesired characteristics of being large and complex with a highly steep learning curve. Work has also been done on producing XML schemas [5] from models expressed in ontology language OIL. This work reported, however, that the XML Schema notion of type inheritance does not correspond well to inheritance in object-oriented models.

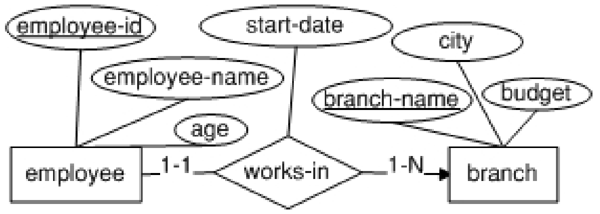

Figure 1. A binary ER diagram

This paper is organized as follows - Section 2 describes the proposed methodology of mapping relational data to an RDF format with the extensive use of user-defined URIref vocabularies; Section 3 presents an illustration of the same. Finally, Section 4 concludes the paper and briefly presents an idea for the future direction of research.

2. RDF and relational databases

Since the mapping between a relational database and an ER diagram is straight forward, for the purpose of representing an ER diagram in RDF, the correspondence between a relational database and RDF may be addressed directly. A relational database consists of tables, which consist of tuples or records. Each tuple consists of a set of attribute values. A tuple, therefore, is comprised of the contents of its attributes, just as an RDF node is nothing but the confluence of many property arcs. The correspondence between RDF and relational databases is:

- A tuple is an RDF subject.
- An attribute is an RDF predicate.
- An attribute value is an RDF object.

With the use of URIref vocabularies for the purpose of storing information about objects, it is imperative that shared vocabularies are developed and their usage encouraged in order to reflect a shared understanding of concepts. For example, in the triple: *<http://www.forexample.org/index.html> dc:creator <http://www.forexample.org/staff-Ids/5670>*

The term *dc:* is a prefix which designates the namespace *http://purl.org/dc/elements/1.1/* which in turn refers to the Dublin Core metadata [6] attribute set, a widely-used set of attributes (properties) for describing information of all kinds. Therefore, the predicate *dc:creator*, is an unambiguous reference to the "creator" attribute in the Dublin Core metadata attribute set. This triple is stating, therefore, that the relationship between the web page (*http://www.forexample.org/index.html*) and the creator of the page (a distinct person, identified by *http://www.forexample.org/staff-IDs/5670*) is exactly the concept identified by *http://purl.org/dc/elements/1.1/creator*. A program that is made familiar with the Dublin Core vocabulary will be able to recognize, at a certain level, what is meant by this relationship.

Further, using URIrefs to identify properties, and in some cases property values as well, is especially important for a number of reasons. Firstly, it distinguishes the properties which some organization may use from different properties which other organizations may use that would otherwise be identified by the same character string. For instance, forexample.org may use "name" to mean someone's full name written out as a string (e.g., "John Smith"), but someone else may intend "name" to refer something else (e.g., the name of a department). If forexample.org writes *http://www.forexample.org/terms/name* for its "name" property, and another organization writes *http://www.another.forexample.org/depts/terms/name* instead, it becomes clear that there are disparate properties involved. Also, using URIrefs to identify properties enables the properties to be treated as resources themselves. Since properties can be treated as resources, additional information can be recorded about them. For instance, the English description of what forexample.org means by "name" may be added simply by writing RDF statements with the property's URIref as the subject.

3. An illustration

We shall now consider the ER diagram in Figure 1; it is comprised of a single binary relationship (for the sake of simplicity). It models a database designed by, say, an organization called "forexample" for the purpose of recording information about its employees. In keeping with the idea of using a shared vocabulary, we shall assume that this organization has constructed its own URIref vocabulary for all the terms that it intends to use such as those for attributes, or as attribute values in certain cases.

Following from the ER diagram, we can have the following tables in a relational database (assuming arbitrary data):

Table 1. Company employees

employee-id	employee-name	age
576	John Smith	57
783	Jack Black	34
863	Will Son	36

Table 2. Company branches

branch-name	city	budget
CO	Flowerville	1000.22
EC	Moneyville	6000

Table 3. Relationship "works-in"

employee-id	branch-name	start-date
576	CO	21-12-1983
783	CO	22-12-1984
863	EC	14-12-1984

The corresponding RDF directed graph for the ER diagram of Figure 1 is shown in Figure 2; the entity sets and the relationship of Figure 1 have been treated as property values of the employee database. Only the first tuple (property "t1") of each table has been shown for the sake of convenience; other tuples (properties "t2", "t3") can be treated in a similar manner and have been omitted from the figure. Also, the namespace prefixes for the resources described in this RDF description have been omitted from the figure only for the sake of convenience. The organization may define a prefix such as *DBterms:* for the namespace *http://www.forexample.org/DBterms/* - a URI that the organization may associate with providing standard definitions for database terms such as "entity", "primary-key" etc.

Another salient feature of this RDF graph is that the primary key attributes (underlined in the ER diagram and in tables 1, 2 and 3) of the tables corresponding to entity sets "employee" (table 1) and "branch" (table 2) have been treated as resources, of which the other attribute values are RDF property values.

These primary-key resources have been linked through the resource "works-in" which serves as a relationship in this case. In our mapping, entities, relationships, key attributes and attribute values are all considered as resources. This enables our mapping to unambiguously refer to these resources and thereby, declare RDF statements about them. Further, this enables relationships to retain their own attributes; this obviates the need for mapping the latter to an entity set participating in that relationship.

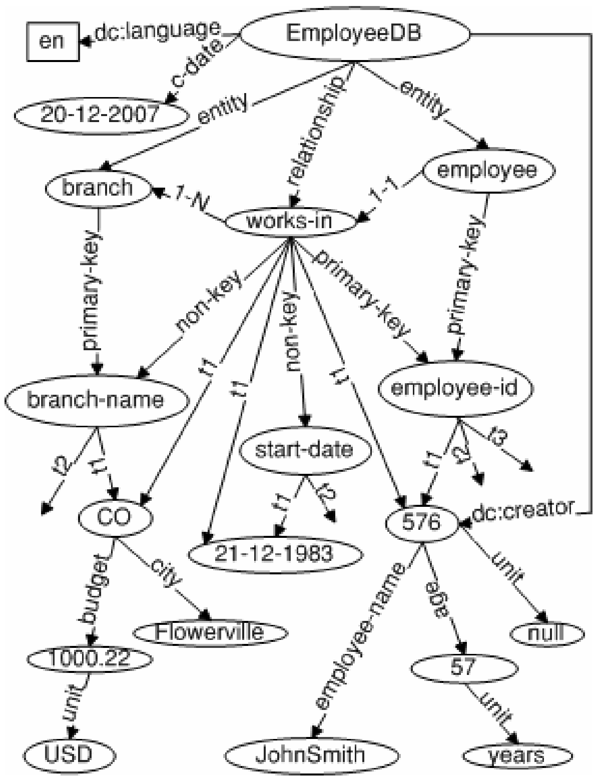

Figure 2. RDF equivalent of the ER diagram in figure 1

Participation constraints have been handled in the same manner as in a general ER diagram. In Figure 2, these constraints (1-N and 1-1) have been represented as property types on the edges between the nodes corresponding to the entity sets and the single relationship. The directions of the property arcs that represent these constraints also serve to specify the mapping constraint, which in this case is many-to-one from "employee" to "branch".

As can also be seen in Figure 2, the organization "forexample", in its shared vocabulary, has defined units for physical quantities as well. In our mapping, whenever an attribute value is a number, its unit must be specified. A special "null" unit has been defined for numbers which do not have a physical unit – for example, a phone number or an employee identification number.

Also worth mentioning is that through the use of RDF, a resource may be linked with others through RDF "properties" – thereby being able to express multiple relationships using the same number of resources. In Figure 2, the resource *576* which signifies an employee in the organization, with a unique ID, has also been identified as the creator of the database by the use of the Dublin Core [6] *dc:creator* property.

4. Conclusion and future research

One of the visions of the Semantic Web has been to enable computer software to locate for us, relevant resources on the Web and also extract, integrate and index the information contained within these resources. In this paper, it has been shown how this need can be partially fulfilled by demonstrating how a relational database may be represented in RDF while utilizing vocabularies that may be developed in, say, RDF Schema or OWL. Further, with RDF parsers being available easily today, data expressed in RDF can be parsed and processed easily by a machine for any desired purpose. The RDF query language, SPARQL, which allows for simple and efficient RDF graph querying is another strong incentive for the use of our mapping.

On the web, such mapping can be implemented by creating a backend system at the server end at the application server layer (using Jena – the Java RDF API) which may exploit the serializability of RDF graphs in XML and thereby, perform the necessary conversions. Given the dynamic nature of online databases, such conversions may be done on-the-fly.

The mapping as described in this paper may be enhanced further so as to incorporate higher features of ER diagram representations such as aggregation, specialization, mapping constraints etcetera. Work has already begun in this direction. The semantic integration of heterogeneous databases, especially that of online databases, has been a long-standing problem for the database community. The mapping proposed in this paper may prove to be an effective solution to this problem.

5. References

[1] F. Manola and E. Miller, "RDF primer", *W3C working draft*, World Wide Web Consortium, Cambridge, Massachusetts, USA, April 2002.

[2] T. Berners-Lee, R. Fielding and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998.

[3] D. Skogan, "UML as a Schema Language for XML based Data Interchange", *Proceedings of the 2nd International Conference on The Unified Modeling Language (UML'99)*, Fort Collins, Colorado, USA, October 1999.

[4] XPetal, Langdale Consultants Website, http://www.langdale.com.au/styler/xpetal/.

[5] M. Klein, D. Fensel, F. van Harmelen, and I. *Horrocks, "*The Relation between Ontologies and Schema-Languages: Translating OIL-Specifications to XML-Schema*", Proceedings of the Workshop on Applications of Ontologies and Problem-solving Methods*, 14th European Conference on Artificial Intelligence ECAI-00, Berlin, Germany, August 2000.

[6] Dublin Core Metadata Initiative, Dublin Core Website, http://www.dublincore.org/.