Relational Databases Conceptual Preservation

Ricardo André Pereira Freitas
José Carlos Ramalho

Department of Informatics – University of Minho
Braga – Portugal
freitas@fam.ulusiada.pt, jcr@di.uminho.pt

Abstract. Previously, a neutral format was adopted to pursue the goal of platform independence and to achieve a standard format in the digital preservation of relational databases, both data and structure (logical model). Currently, we intend to address the preservation of relational databases by focusing on the conceptual model of the database, considering the database semantics as an important preservation "property". For the representation of this higher level of abstraction present in databases we use an ontology based approach. At this higher abstraction level exists inherent Knowledge associated to the database semantics that we tentatively represent using "Web Ontology Language" (OWL). We developed a prototype (supported by case study) and define a mapping algorithm for the conversion between the database and OWL.

Key words: Digital Preservation, Relational Databases, Ontology, Conceptual Models, Knowledge, XML, Digital Objects

1 Introduction

This work addresses the issue of Digital Preservation and focuses on a specific class of digital objects: Relational Databases (RDBs). These kinds of archives are important to several organizations (they can justify their activities and characterize the organization itself) and are virtually in the base of all dynamic content in the Web.

In previous work [1] we adopted an approach that combines two strategies and uses a third technique — migration and normalization with refreshment:

- Migration which is carried in order to transform the original database into the new format – Database Markup Language (DBML) [2];
- Normalization reduces the preservation spectrum to only one format;
- Refreshment consists on ensuring that the archive is using media appropriate to the hardware in usage throughout preservation [3].

This previous approach deals with the preservation of the Data and Structure of the database, i.e., the preservation of the database logical model. We developed a prototype that separates the data from its specific database management environment (DBMS). The prototype follows the Open Archival Information System

(OAIS) [4] reference model and uses DBML neutral format for the representation of both data and structure (schema) of the database. The prototype is based on a web application with multiple interfaces. Three information packages are the base of the archival process: Submission Information Package (SIP), Archival Information Package (AIP) and Dissemination Information Package (DIP).

1.1 Conceptual Preservation

In this paper, we address the preservation of relational databases by focusing on the conceptual model of the database (the information system – IS). It is intended to raise the representation level of the database up to the conceptual model and preserve this representation. For the representation of this higher level of abstraction on databases we use an ontology based approach. At this level there is an inherent Knowledge associated to the database semantics that we represent using OWL [5]. We evolved the prototype (supported by case study) and established an algorithm that enables the mapping process between the database and OWL.

Our hypothesis concentrates on the potentiality of reaching relevant stages of preservation by using ontologies to preserve of RDBs. This lead us to the preservation of the higher abstraction level present in the digital object, which corresponds to the database conceptual model. At this level there is an inherent **Knowledge** associated to the database semantics (Fig. 1).

Digital Object	Preservation Levels	Relational Database
Experimented Object	**Ontology**	**Conceptual Model**
Conceptual Object	DBML	Logical Model
Logical Object	–	Original Bitstream
Physical Object	–	Physical Media

Fig. 1. Preservation Policy

In the following section, we overview the preservation of relational databases and relation between ontologies and databases establishing the state-of-the-art and referring to related work. The prototype and the mapping process from RDBs to OWL is detailed in section 3. At the end we draw some conclusions and specify some of the future work.

2 Relational Databases and Ontologies

There is a direct relation between ontologies and databases: a database has a defined scope and intends to model reality within that domain for computing (even when it is only virtual or on the web); ontology in ancient and philosophical significance means the study of being, of what exists [6].

2.1 Relational Databases Preservation

Strategies for digital preservation are detailed in [7] and preservation properties of digital objects are addressed in [8].

Considering the nature of the digital artifacts that we are addressing – relational databases – there is an European strategy encompassed in the "Planets Project" [9] to enable their long term access. The project adopted the SIARD [10] solution, which is based on the migration of database into a normalized format (XML – eXtensible Markup Language [11]). The SIARD was initially developed by the Swiss Federal Archives (SFA).

Another approach, also based on XML, relies on the main concept of "extensibility" – XML allows the creation of other languages [12] (it can be called as a meta language). The DBML [2] (Database Markup Language) was created in order to enable representation of both **DATA** and **STRUCTURE** of the database.

Relational databases model is designed to support an information system at its operational level. Thus, RDBs are complex and their data can be distributed into several entity relations that related to each other through specific attributes (foreign to primary keys) in order to avoid redundancy and maintain consistency [13].

Both approaches (SIARD and DBML) adopt the strategy of Migration of the database to XML, why? A neutral format that is hardware and software (platform) independent is the key to achieve a standard format to use in digital preservation of relational databases. This neutral format should meet all the requirements established by the designated community of interest.

2.2 Ontologies

The notion of ontology then emerges due to the need of expressing concepts in different domains (ontologies as collections of information). An ontology can provide readable information to machines [14] at a conceptual level (higher abstraction level). Ontologies also enable the integration and interpretability of data/information between applications and platforms.

Behind ontologies there is the need of knowledge representation for machine interpretation. Two technologies: a) the RDF (Resource Description Framework) [15] triples give support for the meaning in simple sentences b) and XML [11] is used for structuring documents [6]. The RDF document consist on a set of triples, – *object, property, value* – that we can also define as – *subject, predicate, object* [16].

2.3 Related Work

Work related to RBDs and ontologies transformations proliferate and is addressed continuously. Considering the RDF [15], OWL [5], ontologies and RDBs, several frameworks, mapping approaches and tools exist: Virtuoso RDF View

[17]; D2RQ [18]; Triplify [19]; RDBToOnto [20]; R2O [21]; Dartagrid Semantic Web toolkit [22]; SBRD Automapper [23]; XTR-RTO [24]; RDB2OWL [25]; DB2OWL [26]; R2RML [27]; OntER [28]; DM2OWL [29]; OWLFromDB [30] and also "Concept hierarchy as background knowledge" proposal [14] among others.

Several of these approaches and tools are referenced and analyzed in the W3C (World Wide Web Consortium) [31] Incubator group survey [32] and also in [14].

The conversion from databases into an ontology could be characterized as a process in the scope of reverse engineering [28]. While some approaches and works try to establish a mapping language or a mapping process [33], others use different techniques and strategies for the database translation [29] into an ontology (e.g. OWL).

The R2RML (RDB to RDF Mapping Language) [27] working draft submitted to W3C is designed for mapping the data within the attributes of a **table** into pairs: property, object. Each record within a table share the same subject in this RDF triple map relation. This approach supports the input of "logical" tables from the source database, which can be an existing table, a view or a valid SQL query.

R2O [21] approach is based on a mapping document generation (mapping language). Virtuoso RDF View establishes a set of RDF statements by defining for each table: *primary key* (subject), *attribute* (predicate), *value* (object). In the RDB2OWL [25] a different strategy is used since it is created a mapping RDB schema. The "Concept hierarchy as background knowledge" proposal [14] gives special attention to the data preparation before conversion and to the knowledge that resides on the database.

3 From RDB to OWL

This section presents the work developed to convert databases to ontology, based on a mapping process (mapping algorithm), for preservation. We intend to preserve a snapshot of the database (or a frozen database) by preserving the OWL generated from the database.

We start by concentrating our efforts on detailing the mapping process and analyzing the created algorithm. Then the conducted tests and some of the results are also presented.

3.1 Mapping Process of RDBs to OWL – Prototype

Our work implements the conversion from RDBs into OWL through an algorithm that performs the mapping process. The developed prototype enables the connection to a DSN (Data Source Name), extracts the data/information needed and gives the initial possibility of selecting the tables of interest (for conversion). It is assumed that the source database is normalized (3NF).

Lets start by enumerating the properties of RDBs that are addressed and incorporated in the ontology (OWL): a) **Tables** names; b) **Attributes** names and

data types; c) **Keys** primary keys, foreign keys (relationships between tables); d) **Tuples** data. These elements are extracted from the database into multidimensional arrays (Fig. 2). We also summarize the mapping process in figure 2. From the conceptual mapping approach and some DBMS heuristics we start to manually convert a relational database (case study database) into OWL using Protégé [34]. The algorithm was then designed based on the defined mapping and from the code analysis (Protégé – OWL/XML format).

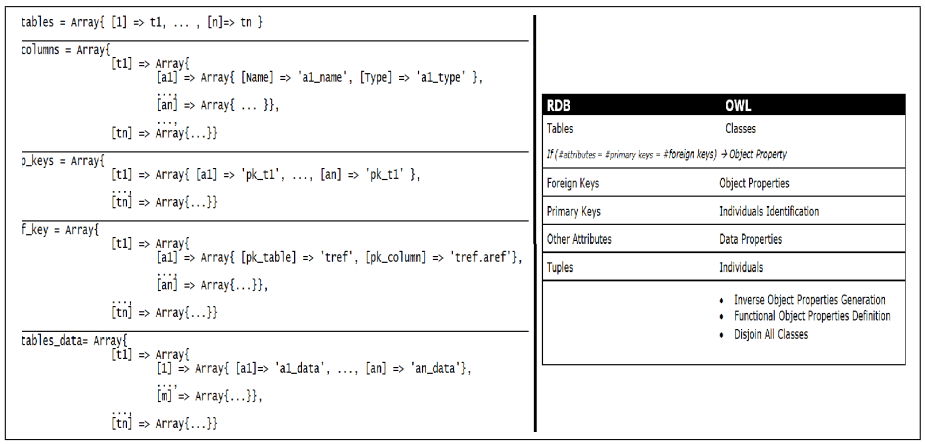

Fig. 2. Multidimensional Array Structure & Mapping Process Summarized

For each `table` on the database we define a **class** on the ontology with the exception of those tables where all attributes constitute a composed primary key (combination of foreign keys). These link tables used in the relational model to dismount a many-to-many relationship, are not mapped to OWL classes, instead they give origin to **object properties** in the ontology. These object properties have on their domain and range the correspondent classes (database tables) involved in the relationship (Fig. 3).

The `foreign keys` of the tables mapped directly to OWL classes also give origin to **object properties** of the correspondent OWL classes (tables). The `attributes` of the several tables are mapped to **data properties** within the analogous OWL classes with the exception of the attributes that are foreign keys (Fig. 4).

The algorithm generates inverse object properties for all relationships among the classes. If the object properties are generated directly from a 1-to-many relationship (which is the last case) it is possible to define one of the object properties as functional (in one direction).

The `tuples` of the different tables are mapped to **individuals** in the ontology and are identified by the associated `primary key` in the database. A tuple in a database table is mapped to an individual of a class (Fig. 4).

```
//    Classes (tables) & ObjectProperties (link tables - non_classes)
FOREACH [ table ]
        IF [ ( |columns[table]| = |p_keys[table]| ) AND ( |p_keys[table]| = |f_keys[table])| ) ] THEN
                non_class[] = table
                FOREACH [ columns[table] - 1 ]

                NEW 'ObjectProperty'
                        Property_Description = 'is_' + f_keys[table][columns[table]][pk_table] + '_Of'
                        Domain = f_keys[table][columns[table]][pk_table]
                        Range = f_keys[table][next(columns[table])][pk_table]

                NEW 'ObjectProperty'
                        Property_Description = 'has_' + f_keys[table][columns[table]][pk_table]
                        Domain = f_keys[table][next(columns[table])][pk_table]
                        Range = f_keys[table][columns[table]][pk_table]

                NEW 'InverseObjectProperties'
                        Property_Description = 'is_' + f_keys[table][columns[table]][pk_table] + '_Of'
                        Property_Description = 'has_' + f_keys[table][columns[table]][pk_table]

                END FOR
        ELSE
        class[] = table
        END IF
END FOR
```

Fig. 3. Algorithm – Classes and Non Classes

The object properties that relates individuals in different classes are only defined in one direction. If in the inverse pair of object properties exists one property that is functional, is that one that it is defined; if not, the generated object property assertion is irrelevant.

```
//   Sub Classes of Thing & Disjoint all & Object and Data Properties
class_disjoint[] = class
FOREACH [ class ]
        NEW class 'SubClassOf' owl:Thing
        FOREACH [ class_disjoint ]
                IF  [ class IN class_disjoint ] THEN
                        NEW 'DisjointClasses'
                                Class_Description = class
                                Class_Description = class_disjoint
                END IF
        END FOR
        pop(class_disjoint)

        FOREACH [ f_keys[table] as fk ]
                NEW 'ObjectProperty'
                        Property_Description = 'is_' + fk['pk_table'] + '_Of'
                        Domain = fk['pk_table']
                        Range = class
                NEW 'ObjectProperty'
                        Property_Description = 'has_' + fk['pk_table']
                        Domain = class
                        Range = fk['pk_table']
                NEW 'InverseObjectProperties'
                        Property_Description = 'is_' + fk['pk_table'] + '_Of'
                        Property_Description = 'has_' + fk['pk_table']
                NEW 'FunctionalObjectProperty'
                        Property_Description = 'is_' + fk['pk_table'] + '_Of'
        END FOR

        FOREACH [ columns[table] as table_data ]
                IF [ f_keys[$table][table_data['Name']]['pk_column'] != table_data['Name'] ] THEN
                        NEW 'DataProperty'
                                Property_Description = 'has_' + table_data['Name']
                                Domain = class
                                Range = data_type
                END if
        END FOR
END FOR
```

```
//   tuples -> Individuals   //
FOREACH [ class ]
        FOREACH [ tables_data[table] as tuple ]
                primary_key = class
                FOREACH [ p_keys[table] as pk)
                        primary_key = primary_key + pk
                END FOR

                NEW 'classAssertion'
                        Class_Description = class
                        NamedIndividual = primary_key
                FOREACH [ tuple as ktet ]
                        IF [ NOT [ kt IN array_keys(f_keys[table]) ] ]
                                NEW 'DataPropertyAssertion'
                                        DataProperty = class + '_has_' + kt
                                        NamedIndividual = primary_key
                                        Literal = t
                        ELSE
                                NEW 'ObjectPropertyAssertion'
                                        ObjectProperty = f_keys[table][kt]['pk_table']
                                        NamedIndividual = primary_key
                                        NamedIndividual = f_keys[table][kt]['pk_table'] + '_' + t
                        END IF
                END FOR
        END FOR
END FOR
//   tuples -> objectProperties (link tables)   //
FOREACH [ non_class ]
        FOREACH [ columns[table] - 1 ]
                FOREACH [ tables_data[table] as tuple ]
                        NEW 'ObjectPropertyAssertion'
                                ObjectProperty = f_keys[table][columns[table]]['pk_table']
                                NamedIndividual = f_keys[table][next(columns[table])]['pk_table'] +
                                        '_' + tuple[f_keys[table][next(columns[table])]['pk_column']]
                                NamedIndividual = f_keys[table][columns[table]]['pk_table'] +
                                        '_' + tuple[f_keys[table][columns[table]]['pk_column']]
                END FOR
        END FOR
END FOR
```

Fig. 4. Algorithm – Structure Generation & Individuals

3.2 Prototype – Tests and Results

The algorithm was then tested with the case study database. Figure 5 shows the database logical model and the ontology conceptual approach. It was necessary to do some adjustments in order to achieve a consistent ontology. Then we successfully use the HermiT 1.3.3 reasoner [35] to classify the ontology. The inverse "object properties assertions" that the algorithm do not generates for the individuals were inferred. Some equivalent (and inverse functionality) object properties were also inferred.

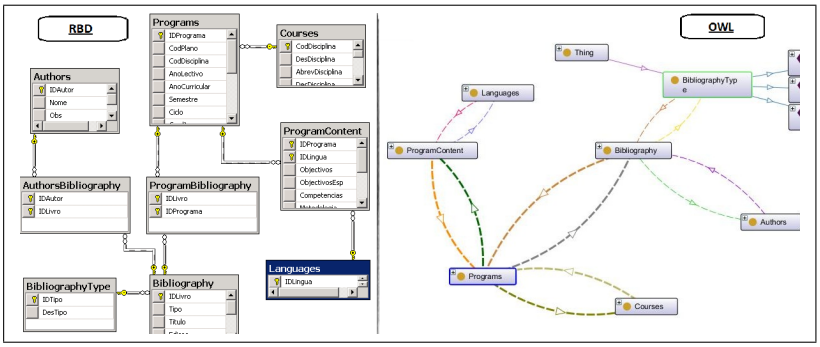

Fig. 5. RDB Logical Model *vs* Ontology Overview

In Figure 6 we present an example of the generated ontology. This example focus on the relationship that exists between the two tables ("Authors" and "Bibliography") where the link table "AuthorsBibliography" is mapped into an object property (and inverse object property) relating the correspondent mapped classes. It is also shown a portion of the generated OWL document where we demonstrate the results of mapping a table attribute into a data property of a class.

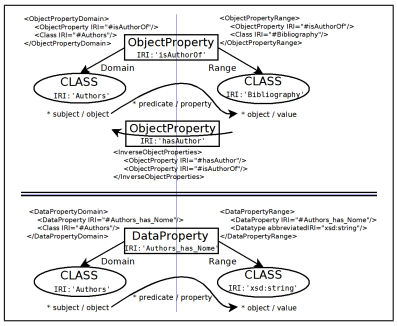

Fig. 6. Results Portion: tables "Authors" and "Bibliography" relationship & "Authors" attribute mapping

The next step consisted on testing the algorithm with other databases. We use one MySQL database and two MSSQL Server databases (the maximum tables size were about tens of thousands records). All databases used in this research are from the University Lusíada information system.

The results were very satisfactory because the algorithm achieve similar results of the ones obtained with the case study database only with minor inconsistencies. The processing time is an issue directly related to the dimension of

the database (it is necessary to test the algorithm with huge databases [millions of records] in machines with powerful processing capability).

4 Conclusion and Future Work

Ontologies and databases are related to each other because of their characteristics. Using ontologies in database preservation is an approach to capture the "knowledge" associated to the conceptual model of the database.

In previous work we preserve the database data and structure (logical model) by ingesting the database in a XML based format (DBML [2]) into an OAIS [4] based archive.

Here, we present the work developed in order to convert databases to ontology, based on a mapping process (mapping algorithm), for preservation. In order to preserve a snapshot of the database (or a frozen database) we preserve the ontology (OWL [5], also a XML based format) obtained from the application of developed algorithm to the source database. We tested the algorithm with few databases and the results were acceptable in terms of consistency of the generated ontology (and comparing to the results obtained with the case study database).

This generated ontologies will induce the development of a new database browser/navigation tool.

Ontologies also have other potentialities such as the asset of providing answers to questions that other standards are limited. For example, in terms of metadata, one issue that we intend to also address in future work.

We also anticipate the possibility of integration between Web Ontology Language (OWL) and Semantic Web Rule Language (SWRL [36]) to consolidate the asserted and inferred knowledge about the database and its information system.

References

1. R. Freitas, J. Ramalho, "Relational Databases Digital Preservation," Inforum: Simpósio de Informática, Lisboa, Portugal, 2009, ISBN: 978-972-9348-18-1; [Online]. Available: http://repositorium.sdum.uminho.pt/handle/1822/9740
2. M. Jacinto, G. Librelotto, J. Ramalho, P. Henriques, "Bidirectional Conversion between Documents and Relational Data Bases," 7th International Conference on CSCW in Design, Rio de Janeiro, Brasil, 2002.
3. Ricardo André Pereira Freitas, "Preservação Digital de Bases de Dados Relacionais," MSc Thesis, Escola de Engenharia, Universidade do Minho, Portugal, 2008.
4. Consultative Committee for Space Data Systems. "Reference Model for an Open Archival Information System (OAIS) – Blue Book," National Aeronautics and Space Administration, Washington, 2002.
5. "OWL – Web Ontology Language" [Online]. Available: http://www.w3.org/TR/owl-features/
6. Tim Berners-Lee, James Hendler and Ora Lassila, "The Semantic Web", Scientific American, May 2001.
7. K. Thibodeau, "Overview of Technological Approaches to Digital Preservation and Challenges in Coming Years," presented at The State of Digital Preservation: An International Perspective, Washington D.C., 2002.
8. A. Wilson, "Significant Properties Report," InSPECT Work Package 2.2, Draft/Version 2 (2007).
9. "PLANETS – Preservation and Long-term Access through NETworked Services" [Online]. Available: http://www.planets-project.eu/
10. "SIARD – Format Description," Swiss Federal Archives - SFA, 2008.

11. XML, "Extensible Markup Language," in W3C – The World Wide Web Consortium [Online]. Available: http://www.w3.org/XML/
12. J. Ramalho, P. Henriques, "XML and XSL - Da Teoria à Prática," FCA - Editora Informática, 2002.
13. Edgar Codd, "A Relational Model of Data for Large Shared Data Banks," in Communications of the ACM, 1970.
14. H. Santoso, S. Hawa and Z. Abdul-Mehdia, "Ontology extraction from relational database: Concept hierarchy as background knowledge," Knowledge-Based Systems, Elsevier, 2010
15. "Resource Description Framework," [Online]. Available: http://www.w3.org/RDF/
16. G. P. Zarri, "RDF and OWL," Encyclopedia of Knowledge Management, 2006.
17. OpenLink Virtuoso Platform, "Automated Generation of RDF Views over Relational Data Sources," [Online]. Available: http://docs.openlinksw.com/virtuoso/rdfrdfviewgnr.html
18. C. Bizer, R. Cyganiak, "D2RQ – Lessons Learned," Position paper for the W3C Workshop on RDF Access to Relational Databases, Cambridge, USA, 2007.
19. S. Auer, S. Dietzold, J. Lehmann, S. Hellmann , D. Aumueller, "Triplify – Light-Weight Linked Data Publication from Relational Databases," proceedings of WWW 2009, Madrid, Spain
20. Farid Cerbah, "Learning highly structured semantic repositories from relational databases: the RDBToOnto tool," In Proceedings of the 5th European Semantic Web Conference, Spain, 2008.
21. J. Barrasa, A. Gomez-Perez, "Upgrading relational legacy data to the semantic web," 15th international conference on World Wide Web Conference (WWW 2006), Edinburgh, United Kingdom, 2006.
22. H. Chen, Z Wu, "DartGrid III: A Semantic Grid Toolkit for Data Integration," Proceedings of the First International Conference on Semantics, Knowledge, and Grid, 2005
23. M. Fisher, M. Dean, "Automapper: Relational Database Semantic Translation using OWL and SWRL," Proceedings of the IASK International Conference E-Activity and Leading Technologies, Porto, Portugal, 2007
24. J. Xu and W. Li, "Using Relational Database to Build OWL Ontology from XML Data Sources," CISW 2007 – Proceedings of the 2007 International Conference on Computational Intelligence and Security Workshops, IEEE Computer Society, Washington, DC, USA, (2007)
25. G. Bumans, K.Cerans, "RDB2OWL : a Practical Approach for Transforming RDB Data into RDF/OWL," Proceedings of the 6th International Conference on Semantic Systems ISEMANTICS 10, 1-3. Retrieved from http://portal.acm.org/citation.cfm?id=1839739
26. N. Cullot, R. Ghawi, K. Yetongnon,"DB2OWL: A Tool for Automatic Database-to-Ontology Mapping," . In Proc. of 15th Italian Symposium on Advanced Database Systems (SEBD 2007), pages 491-494, Torre Canne, Italy, June 2007.
27. "R2RML: RDB to RDF Mapping Language," W3C Working Draft, 24 March, 2011
28. J. Trinkunas, O. Vasilecas, "Building Ontologies from Relational Databases Using Reverse Engineering Methods," International Conference on Computer Systems and Technologies - CompSysTech07, ACM, 2007, ISBN: 978-954-9641-50-9
29. K. M. Albarrak , E. H. Sibley, "Translating relational & object-relational database models into OWL models," Proceedings of the 10th IEEE international conference on Information Reuse & Integration, Las Vegas, Nevada, USA, 2009
30. C. He-ping, H. Lu, C. Bin, "Research and Implementation of ontology automatic construction based on relational database," International Conference on Computer Science and Software Engineering. IEEE Computer Society, 2008.
31. "World Wide Web Consortium," [Online]. Available: http://www.w3.org/
32. "A Survey of Current Approaches for Mapping of Relational Databases to RDF," W3C Incubator Group, 2009
33. I. Myroshnichenko , M. C. Murphy, "Mapping ER Schemas to OWL Ontologies," Proceedings of the 2009 IEEE International Conference on Semantic Computing, p.324-329, September 14-16, 2009
34. http://protege.stanford.edu
35. http://hermit-reasoner.com/
36. "SWRL: A Semantic Web Rule Language Combining OWL and RuleML" [Online]. Available: http://www.w3.org/Submission/SWRL/