Mapping Process of Relational Data to OWL Class Instances using XSLT

Abderrahim Marzouk
Département de Mathématiques & Informatique Laboratoire IR2M
University Hassan 1er FST de Settat Maroc

ABSTRACT

Ontologies are widely used in many domains and evolve as domains change [7]. From the other side, the relational database technology has ensured the best facilities for storing, updating and manipulating the information of problem domain. We propose in this paper a transformation process from existing data in relational databases into OWL class instances using XSLT. This process allows us to mapping some constraints on columns such as foreign key while maintaining semantic constraints. Our approach start by translate the relational data into adequate XML document conforming to a suitable XML schema.

Since our process is based on XSLT Stylesheets, its transformation rules can be modified in a very flexible manner in order to consider different mapping strategies and future requirements.

Keywords

Relational databases, XML, XSD, XSLT and OWL class instances.

1. INTRODUCTION

In this paper we propose a set of rules XSLT generating OWL class instances from data stored in relational database in the order to profit from the advantages of Web Ontology Language OWL (logic and reasoning should be applied) and to make the relation data information in the Web systems[5, 6, 7].

The remainder of this paper is organized as follows: Section 2 describes our mapping process. Section 3 presents mappings between relational data and ontological concepts. The process for mapping the given relational data into OWL class instances is explained in Section 4. In Section 5, we give a mapping example. Section 6 concludes the paper and considers the future work

2. DESCRIPTION OF MAPPING PROCESS

A relational data is represented by an XML document conforming to the below XML Schema [Fig. 1]. The elements in this document will be transformed into OWL elements using XSLT.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:element name="rows">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="table"
minOccurs="0"  maxOccurs="unbounded">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="row"  minOccurs="0" maxOccurs="unbounded">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="column"  minOccurs="0" maxOccurs="unbounded">
<xsd:complexType>
  <xsd:attribute name="name" type="xsd:string"/>
  <xsd:attribute name="type" type="xsd:string"/>
  <xsd:attribute name="value" type="xsd:string"/>
</xsd:complexType>
</xsd:element>
<xsd:element name="primaryKey">
 <xsd:complexType>
  <xsd:attribute name="name" type="xsd:string"/>
  <xsd:attribute name="type" type="xsd:string"/>
  <xsd:attribute name="value" type="xsd:string"/>
 </xsd:complexType>
</xsd:element>
<xsd:element name="foreignKey" minOccurs="0" maxOccurs="unbounded">
<xsd:complexType>
  <xsd:attribute name="name" type="xsd:string"/>
```

```
<xsd:attribute name="type" type="xsd:string" />
<xsd:attribute name="value" type="xsd:string"/>
<xsd:attribute name="foreignTable" type="xsd:string"/>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
<xsd:attribute name="name" type="xsd:string"/>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:schema>
```

Fig 1: The file data.xsd

3. MAPPING ROWS

In the simplest case, a RDB table T maps to an OWL class denoted by C (T).

Each row R in the table T is mapped to an OWL individual denoted by I(R). The term "individual" is used to refer to instances of OWL classes [1, 2].

The OWL class instances are populated from a table corresponding.

Each individual is characterized by a name (individual Name) and a type (individual Type) which is the class instantiated by the individual.

The mapping process will generate one OWL named individual for each row in the database tables. The named individual is created using the language tag *<owl:NamedIndividual rdf:about="instanceID">*, where instanceID is the unique identifier (such as the name) of the instance. This identifier is choosing as a string obtained by concatenating the name of the table and the value of the primary key corresponding to the mapped row.

An individual is declared as an instance of a class by specifying *<rdf:type rdf:resource = "className">* where className is the class to which the instance belongs [3, 4].

It is represented as shown in Fig 2.

```
<owl:NamedIndividual
rdf:about="tableNameprimaryKeyValue">
   <rdf:type rdf:resource="className"/>
        .....
</owl:NamedIndividual>
```

Fig 2: The XML structure of named individual

After the new individual I(R) has been created, it must be filled by the values of all columns of the row, including primary keys and foreign keys. Thus, sub-elements will be added to I(R) using the following rules:

- A subelement *<columnName rdf:datatype="dataType of column">* for each column in a row R that is neither primary key nor foreign key. The value of this element is the same as the value of the column.

- A subelement *<primaryKeyName rdf:datatype="dataType of primary key">* for the primary key of row R. The value of this element is the same as the value of the primary key.

- A subelement *<className-targetClassName rdf:resource = "targetClassName-foreignKeyValue"/>* for each foreign key in a row R of a table T that references row R' in another table T', where targetClassName is the OWL class corresponding to a table T'.

Note that the name of the target individual I (R') is obtained by concatenating the name of the table T' referenced and the value of the foreign key, since this value is the same as the value of primary key of the row R' corresponding to the target individual I(R').

Example: Below are a sample database tables Professor, Department [3] and an individual corresponding to the first row in table Professor.

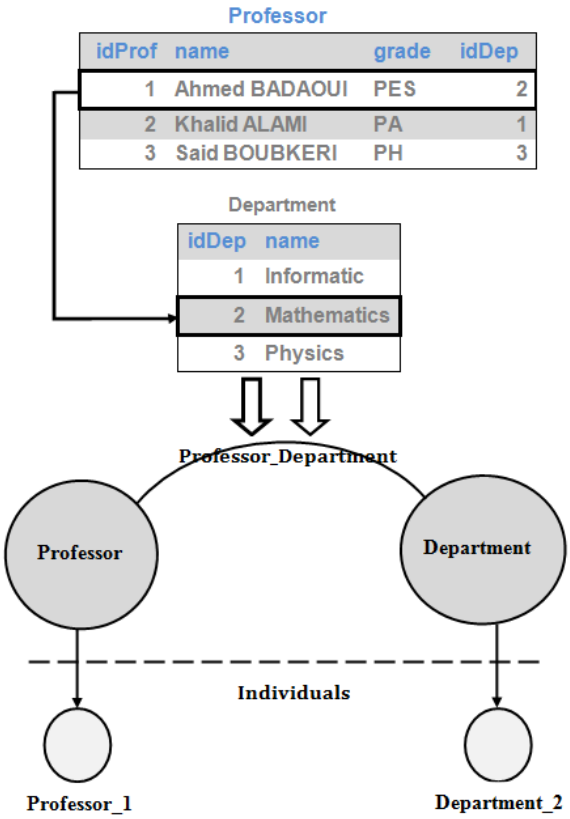

Fig 3: Example of an individual and his target individual

For the first row in table Professor having a foreign key referencing the primary key of the table Department, we create an individual named Professor_1 connecting to a target individual Department_2 as shown in the below figure [Fig. 4].

```
<owl:NamedIndividual rdf:about="Professor_1">
<rdf:type rdf:resource="Professor"/>
<idProf rdf:datatype="&xsd;integer">1</idProf>
<name rdf:datatype="&xsd;string">Ahmed Badaoui
</name>
<grade rdf:datatype="&xsd; string ">PES</grade>
<Professor_Department rdf:resource="Department_2"/>
</owl:NamedIndividual>
```

Fig 4: Example of OWL instance for a class C(Professor)

4. MAPPING SCENARIO

In this section, we describe our mapping process from relational data given as XML document to OWL class instances which raise the XML source documents to OWL ontology. The XML document contains a description of relational data.

Knowing that XSLT (XML StyleSheet Language Transformation) is a W3C standard widely used, that our source document is in XML and our destination language is OWL. The transformation of relational data to OWL instances is implemented as a set of XSLT stylesheets that take as input an XML description of relational data.

Fig 5: Mapping diagram

The following file called transformation.xsl contains a set of rules XSLT transforming an XML description of existing relational data to OWL class instances.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE xsl:stylesheet [
<!ENTITY owl "http://www.w3.org/2002/07/owl#">
<!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-
ns#">
<!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#">
<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">
]>
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:owl="&owl;" xmlns:rdf="&rdf;" xmlns:rdfs="&rdfs;"
xmlns:xsd="&xsd;" version="1.0">
<xsl:output method="xml" indent="yes" />
<xsl:template match="/">
<rdf:RDF xmlns:owl="&owl;" xmlns:rdf="&rdf;"
xmlns:rdfs="&rdfs;" xmlns:xsd="&xsd;">
<xsl:apply-templates select="//row"/>
</rdf:RDF>
</xsl:template>
<xsl:template match="//row">
<xsl:element name="owl:NamedIndividual">
<xsl:attribute name="rdf:about">
<xsl:value-of
select="concat(../@name,'_',./primaryKey/@value)" />
</xsl:attribute>
<xsl:element name="rdf:type">
<xsl:attribute name="rdf:resource">
<xsl:value-of select="../@name" />
</xsl:attribute>
</xsl:element>
<xsl:apply-templates select=".//primaryKey"/>
<xsl:apply-templates select=".//column"/>
<xsl:apply-templates select=".//foreignkey"/>
</xsl:element>
</xsl:template>
<xsl:template match="//row//column">
<xsl:element  name="{@name}">
<xsl:attribute name="rdf:datatype">
<xsl:value-of select="concat('xsd:',@type)" />
</xsl:attribute>
<xsl:value-of select="@value"/>
</xsl:element>
</xsl:template>
<xsl:template match="//row//primaryKey">
<xsl:element  name="{@name}">
<xsl:attribute name="rdf:datatype">
<xsl:value-of select="concat('xsd:',@type)" />
</xsl:attribute>
<xsl:value-of select="@value" />
</xsl:element>
</xsl:template>
<xsl:template match="//row//foreignkey">
<xsl:element
name="{concat(../../@name,'_',@foreignTable)}">
<xsl:attribute name="rdf:resource">
<xsl:value-of select="concat(@foreignTable,'_',@value)" />
</xsl:attribute>
</xsl:element>
</xsl:template>
</xsl:stylesheet>
```

Fig 6: The file transformation.xsl

5. A MAPPING EXEMPLE

We now present an example of mapping an existing relational data into OWL instances. Consider the two tables Professor and Department that are shown in Fig 3. The XML description of this relational data conforming to the XML Schema [see Fig. 1] is stored in a following text file called data.xml. This file is the input of our transformation process.

```
<?xml version="1.0" encoding="UTF-8" ?>
<rows>
<table name="Department">
  <row>
<primaryKey  name="idDep" value="1" type="int" />
   <column name="name" value="Informatic" type="string"/>
  </row>
  <row>
   <primaryKey name="idDep" value="2" type="int" />
   <column name="name" value="Mathematics" type="string"/>
  </row>
  <row>
   <primaryKey name="idDep" value="3" type="int" />
   <column name="name" value="Physics" type="string"/>
  </row>
</table>
<table name="Professor">
   <row>
    <column name="name" value=" Ahmed Badaoui" type="string"/>
    <column name="grade" value="PES" type="string"/>
    <primaryKey  name="idProf"  value="1" type="int"/>
    <foreignkey name="idDep" value="2" type="int" foreignTable="Department"/>
   </row>
   <row>
    <column name="name" value="Khalid Alami" type="string"/>
    <column name="grade" value="PA" type="string"/>
    <primaryKey  name="idProf"  value="2" type="int"/>
    <foreignkey name="idDep" value="1" type="int" foreignTable="Department"/>
   </row>
   <row>
    <column name="name" value="Said Boubkeri" type="string"/>
    <column name="grade" value="PH" type="string"/>
 <rdf:type rdf:resource="Professor"/>
    <primaryKey  name="idProf"  value="3" type="int"/>
    <foreignkey name="idDep" value="3" type="int" foreignTable="Department"/>
   </row>
</table>
</rows>
```

Fig 7: The file data.xml

We use the following PHP script called transformation.php to transform the file data.xml to an OWL file using the XSLT StyleSheet.

```
<?php
$doc = new DOMDocument();
$doc->load("data.xml");
$xsl = new XSLTProcessor();
$xslt = new DOMDocument();
$xslt->load("transformation.xsl");
$xsl->importStyleSheet($xslt);
echo $xsl->transformToXML($doc);
?>
```

Fig 8: The PHP Script transformation.php to perform XSL transformation

When we apply this previous XSL Stylesheet to the XML file, the following OWL output is obtained:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:owl="&owl;" xmlns:rdf="&rdf;" xmlns:rdfs="&rdfs;"
xmlns:xsd="&xsd;">
 <owl:NamedIndividual rdf:about="Department_1">
 <rdf:type rdf:resource="Department"/>
 <idDep rdf:datatype="xsd:int">1</idDep>
<name rdf:datatype="xsd:string">Informatic</name>
 </owl:NamedIndividual>
 <owl:NamedIndividual rdf:about="Department_2">
 <rdf:type rdf:resource="Department"/>
 <idDep rdf:datatype="xsd:int">2</idDep>
 <name rdf:datatype="xsd:string">Mathematics</name>
 </owl:NamedIndividual>
  <owl:NamedIndividual rdf:about="Department_3">
  <rdf:type rdf:resource="Department"/>
 <idDep rdf:datatype="xsd:int">3</idDep>
  <name rdf:datatype="xsd:string">Physics</name>
 </owl:NamedIndividual>
  <owl:NamedIndividual rdf:about="Professor_1">
  <idProf rdf:datatype="xsd:int">1</idProf>
```

```
<name rdf:datatype="xsd:string"> Ahmed Badaoui</name>
<grade rdf:datatype="xsd:string">PES</grade>
<Professor_Department rdf:resource="Department_2"/>
</owl:NamedIndividual>
<owl:NamedIndividual rdf:about="Professor_2">
<rdf:type rdf:resource="Professor"/>
<idProf rdf:datatype="xsd:int">2</idProf>
<name rdf:datatype="xsd:string">Khalid Alami</name>
<grade rdf:datatype="xsd:string">PA</grade>
<Professor_Department rdf:resource="Department_1"/>
</owl:NamedIndividual>
<owl:NamedIndividual rdf:about="Professor_3">
<rdf:type rdf:resource="Professor"/>
<idProf rdf:datatype="xsd:int">3</idProf>
<name rdf:datatype="xsd:string">Said Boubkeri</name>
<grade rdf:datatype="xsd:string">PH</grade>
<Professor_Department rdf:resource="Department_3"/>
</owl:NamedIndividual>
</rdf:RDF>
```

Fig 9: The Output of the mapping

6. CONCLUSION AND FUTUR WORK

This paper has presented a mapping process which makes a transformation from relational data to OWL instances. This transformation preserves semantics of some constraints of the row's columns in order to obtain a usable ontology for needs in shared information. For the realization of this transformation we developed a suitable XML schema for relational data. Our process is implemented running the XSLT transformation over the XML instances document, produces an OWL output document containing the OWL class instances. We plan to apply the current process to transform object data to OWL class instances.

7. REFERENCES

[1] Etminani, K, Kahani, M,Yanehsari, N.R. Building ontologies from relational databases. Networked Digital Technologies, 2009.

[2] LU Yan-hui, MA Zong-min, WANG Yu-xi. Study on the OWL Ontology Construction Approach Based on Relational Database. Computer Science, July 2009.

[3] Jamal Bakkas, Mohamed Bahaj, Abderrahim Marzouk. Direct Migration Method of RDB to Ontology while Keeping Semantics International Journal of Computer Applications (0975 – 8887) Volume 65– No.3 March 2013.

[4] Shihan Yang,Ying Zheng,Xuehui Yang. Semi automatically Building Ontologies from Relational Databases. ICCSIT, July 2010.

[5] Guntars Bumans. Mapping between Relational Databases and OWL Ontologies: an example. Scientific Papers, University Of Latvia Computer Science And Information Technologies, Vol. 756, 2010.

[6] W. hu, y. Qu, Discovering Simple Mappings between Relational Database Schemas and Ontologies. In: Proc. of the 6th International Semantic Web Conference 2007

[7] Ernestas Vysniauskas, Lina Nemuraite, Transforming ontology representation from owl to relational database**,** issn 1392 – 124x information technology and control, 2006, vol.35, no.3a