Converting Relational to Qualified XML Schema with Referential Integrity Constraint

Myint Myint lwin, Thi Thi Soe Nyunt, Yuzana
University of Computer Studies, Yangon
lwin.myintmyint@gmail.com, thithisn@gmail.com, yuzana@gmail.com

Abstract

Data Exchanging is one of the most important sections in Web Application. XML is flexible and platform independent which provide for data exchanging. Most of the features such as information exchanging and information extraction are depended on the XML schema. To provide these features, XML schema needs to be qualified to exchange data over the heterogeneous applications. The understanding of the human reader is the important quality factor of XML schema. This paper presents the converting method for relational database to qualified XML schema with two main processes (1) highly nested with referential integrity constraints depends on the database design and (2) grouping the common attributes in the relations which can be applied for the large database. The quality of the resulted XML schema document is described with the complexity measurements.

1. Introduction

In today world, data exchanging is essential part in Web application. However, the heterogeneity introduces the some problem in Web. To solve the heterogeneity problem, XML is defined as the only one standard for data exchanging because it has the strongest expressive power, and incorporates many commonly-recurring schema constraints in its language specification [3]. XML also has some advantages such as platform-independence data representation and transport format and has been easily adopted in diverse fields due to its flexible nature and ease of implementation. Most of the data or information is stored traditional storage system such as relational database. They are popular because their mature and stability. On the other hand, Web technology is popular for business or organizations. The data in the relational database are not compatible for Web application. Therefore, the relational data are necessary to convert into XML format. As a result, the relational to XML conversion method has been proposed with the different point of views such as structural or semantic. But the previous conversion methods did not consider the design factors such as user understandability and maintainability effort. This paper presents the new conversion method for relational database to XML schema with integrity constraints and element group by considering the quality of XML schema.

This paper presents about introduction in section 1. The section 2 describes the related works. The motivation of our research is shown in section 3. The section 4 presents the architecture of proposed system. The transformation example of XML schema and complexity measurement are described in section 5 and 6. Finally concludes in section 7.

2. Related Works

Relational to XML conversion are currently popular in research fields because the volume of information within the today world is staggering, but the limitations of existing technology can make it difficult to access [8]. Many researchers proposed their conversion methods with structural or semantic points of views. The earliest conversion method is Flat Translation (FT) [10] and each relation is converted as element and each attribute of the relation is either converted as subelement or attribute. It is the simplest conversion method but it does not consider the nesting idea. It did not present the relationship between the relations and it is not a efficient conversion method. The Nesting based Translation (NeT) [5] was proposed to overcome nesting problem. It applied the nested structured by nest operator such as "*" and "+". That method is better than FT and useful for decreasing data redundancy. However, it considers one table at a time and not covers for the whole relational database. It also does not include the relationship of all tables. Both FT and NeT are structured method and they did not consider the semantic aspects. Constraints-based Translation (CoT) [6] method was developed to solve the problem which occurred in NeT. It applied the Inclusion Dependency (INDs) of relational schema which based on the foreign key constraints. It is mostly associated with the usage of sub element and IDREF attributes for translation purpose. Moreover, it considers not only the structural part such as tables and columns but also the semantic part such as the constraints and referential integrity (RI). But it can only provide the explicit RI. If the implicit RI exists, it cannot extract RI and cannot generate the exact XML document. The ConvRel algorithm [2] detects the relation between tables and extracts the RI by applying the idea of parent-child relationship. It also provide the N:M relationships modeled as a combination between a nested structure and keyref. All of the above translation methods did not consider the maintainability effort and user understandability. When the databases are larger, some of the relation may have the common attributes. It may introduce the complex and less modular. The quality of the XML schema document includes size of the document, line of code, number of simple type or complex type etc. Moreover, the previous conversion methods only presented their method and did not measure the complexity or quality measurement of the XML schema.

This paper presents the RDB to XML conversion method with referential integrity and grouping the common attributes. It reduces the maintainability efforts. It also provides code modularity and user understandability by applying element groups in the target XML schema.

3. Motivation

In the current world, the daily data are stored in traditional database such as RDB. On the other hand, information sharing in distributed computing environment becomes widespread, XML has become more popular as a universal data format for exchanging structured information via Internet communications. Therefore, the relational data are required to convert into the flexible XML format. However, deploying XML documents is a challenging problem for an application without using supporting schema technology. Using schema not only provides common understanding about exchanged data but also the ability of ease access methods for XML documents to be validated. With the successful design and implementation of schema, the developers can have the capability of increasing productivity, improving software reliability, minimizing development time, and decreasing time to market [4]. To provide the conversion process (RDB to XML) more efficiently, the proposed method is presented.

The proposed method considers both semantic and structural points of view. It includes the concept of integrity constraints to maintain semantic and the modularity of the schema code. Moreover, it can provide other XML schema quality features such as user understandability and reduce the maintainability efforts.

The proposed conversion method chooses the W3C XML schema language to describe the resulted XML schema because it is most suitable for the relational database and it can also provide the domain constraints such as data types.

4. Architecture of the proposed system

The architecture of the proposed system is shown in Figure 1. It shows the processing steps to convert the relational database to XML schema. Therefore, the relational database is taken as input and finally produces the qualified XML schema document. It includes two main processes. They are detecting the integrity constraint to get the highly nested and detecting the general same attributes in the relations to provide the simplicity, user understandability and reduce the maintainability effort. In this paper, the integrity constraints portion is emphasized and presented to get the highly nested structure because the detection of the common attribute was presented in the previous work [11]. Integrity constraints are essential to get the qualified XML schema document. The important integrity constraints are domain constraints and referential integrity constraints. The proposed system extracts these constraints from the metadata of the relational database. Database metadata enables dynamic database access. Moreover it can provide the understanding of database schema, users, tables, views, stored procedure and identifying the primary/foreign keys for a given table etc. Therefore, the metadata is the one of the important things for converting process and can be used for different concepts. For the database point of view, the technical metadata are important. It includes the system metadata which defines the data structures such as tables, fields, data types in the relational engine. As a result, the system metadata of the relational database are applied to get the highly nested structure. Unfortunately, some of the database cannot detect foreign keys because database designer did not define explicitly them. In that case, the proposed system tries to detect foreign keys with the following steps because foreign key (FK) constraints play mainly as actor in the integrity constraints. To extract the FK from each relation, the proposed system uses three steps to detect the FK with certain ways.

First step: detect FK directly from metadata and not require many efforts

Second step: detect FK from indirectly which is extracted from the domain constraints and obey the FK features

Third step: detect FK using the inclusion dependency which the attribute values are same.

The proposed system detects the FK from the relations using the required steps depend on the database design. The proposed system uses only the necessary steps because it intends to reduce the processing time as possible. If the database design is good, the processing step requires only first step to detect FK. Some of the database design expresses the FK during design implementation using the InnoDB engine. For this type of database, only the first step is required to detect FK. It is the fastest method because it can detect the FK from metadata directly.

The second step can be applicable for the some types of database which did not exactly defined FK. But the FK obey some domain constraints. For example, student (SID, name, address), teacher (TID, name, address, dept, SID). In this example, both tables have SID attributes. The second step can exactly define SID as FK if each SID has the same domain constraints such as data type, length.

The third step is used when the first two steps cannot detect FK. It used the concepts of inclusion dependency. It can detect in the data values level. When the attribute values are same and have some relationships (one to one or one to many etc). This step is applicable for the some databases which did not defined FK explicitly.

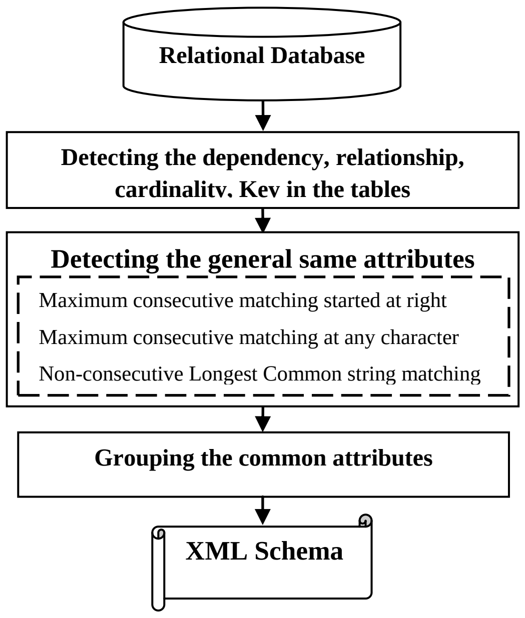

Relational Database
Detecting the dependency, relationship, cardinality, Key in the tables
Detecting the general same attributes
Maximum consecutive matching started at right
Maximum consecutive matching at any character
Non-consecutive Longest Common string matching
Grouping the common attributes
XML Schema

Figure 1. Architecture of the proposed system

In this process, the primary key of each relation is extracted and determine it may be outer or inner element. The proposed method determines the outer element and inner element as the parent/child relation. According to the integrity constraints, the referencing relation as parent and referenced relation as child is defined because a tuple t1 in relation R1 said to reference a tuple t2 in relation R2 if t1[FK]=t2[PK]. To define outer or inner element, the primary key of each relation is checked with the condition (PK(ri) ⊆ $\forall column(rj)$ – PK(rj)). Which means the primary key of each relation is compared with all attributes except primary key of other relation in the database. If IR is existed, ri is defined as the inner element of rj. That step processes for all the relation in the database. After that, the proposed system creates the nested structure according to the child list of each relation.

The next process is the detecting of the common attributes in the tables and converting them as the element group in the resulted schema document. The proposed method does not emphasis on the domain specific. Therefore, the string similarity algorithms are used to collect the common attributes in the relational database. In the matching steps, the distinct attributes in each relation are compared with all attributes of different relation. If the selected common attributes are not distinct in the relation, they are determined as the unqualified to become the common attributes. In this step, three strings matching algorithms are used to get the common attributes. They are non-consecutive longest common substring algorithm, maximum consecutive substring at left algorithm and maximum consecutive at right substring algorithm and these algorithms are already presented in our previous paper [11]. The first algorithm takes two attributes as input and finds the common characters in the input strings. The second algorithm also accepts the two attributes as input and these input strings are compared starting from the left to produce the maximum left consecutive string. The final algorithm produce the maximum right consecutive string by finding the common sub attribute which is consecutive at right. Finally the similarity values are applied to calculate the total similarity value of two strings. The total similarity value [1] is calculated by using the following equation.

$$\alpha = w_1v_1 + w_2v_2 + w_3v_3 \text{ ------- } (1)$$

where α is the total similarity value of two string. Then, w1,w2 and w3 are weights of each normalized values and w1+w2+w3=1. v1,v2 and v3 are similarity value of each string matching algorithm.

5. Transformation Example of XML schema

The following figure shows the relations in the NorthWind database.

Products	Suppliers
<u>**ProductID**</u>	<u>**SupplierID**</u>
ProductName	CompanyName
SupplierID	ContactName
CategoryID	Address
QuantityPerUnit	City
UnitPrice	Region
UnitsInStock	PostalCode
UnitsOnOrder	Country
ReorderLevel	Phone
Discontinued	Fax
	HomePage

OrderDetails	**Categories**
OrderID **ProductID**	**CategoryID**
UnitPrice Quantity Discount	CategoryName Description Picture
Orders	**Employees**
OrderID	**EmployeeID**
CustomerID **EmployeeID** **ShipperID** OrderDate RequiredDate ShippedDate ShipVia Freight ShipName ShipAddress ShipCity ShipRegion ShipPostalCode ShipCountry	LastName FirstName Title TitleOfCourtesy BirthDate HireDate Address City Region PostalCode Country HomePhone Extension Photo Notes ReportsTo
Shippers	**Customers**
ShipperID	**CustomerID**
CompanyName Phone	CompanyName ContactName ContactTitle Address City Region PostalCode Country Phone Fax

Figure 2: Northwind database

It is the sample database of MS Access. In this table, the primary key of each relation is described by underline and the relationships of the relation are maintained by using the referential integrity constraints which are detected with the required FK detecting steps. After this step, the inner or outer elements are defined according to the result of child list.

Table 1. Result of Integrity Constraints Detecting Step

Table	Child List
OrderDetails	Products, Orders
Products	Suppliers, Categories
Orders	Employee,Shippers, Customers
Suppliers	-
Categories	-
Employees	-
Shippers	-
Customers	-

Moreover, some relations have common attributes. These attribute are detected with the string matching algorithms and calculate the similarity values. If the similarity values are satisfied, they are created as the element group in the target XML schema document. In this sample database, the proposed system can detect some common attributes in the relations such as CompanyName,ContactName, Phone and Fax which are exactly common in some relations. Moreover, some attributes Address, City, Region, PostalCode and Country are detected as the common attribute of the relations by using the string matching algorithms because they are partially common. The similarity values of these attributes are described in Table 2.

Table 2. Similarity values of the common attributes in the relations

String1	String2	Total Similarity Value (α)
Address	ShipAddress	0.63
City	ShipCity	0.495
Region	ShipRegion	0.59400004
PostalCode	ShipPostalCode	0.70714283
Country	ShipCountry	0.63

The proposed conversion method reduces line of codes and provides the more understandability of designer to retrieve information from the XML database. The resulted XML schema using element group is shown in below figure.

```
<?xml version=”1.1” encoding=”UTF-8”?>
<xsd:schema xmlns=”http://www.w3.org/2001/XMLSchema”>

<xsd:group name=”G1”>
   <xsd:sequence>
   <xsd:element name=”CompanyName” type=”xsd:string”/>
   <xsd:element name=”ContactName” type=”xsd:string”/>
   <xsd:element name=”Phone” type=”xsd:string”/>
   <xsd:element name=”Fax” type=”xsd:string”/>
  </xsd:sequence>
  </xsd:group>

 <xsd:group name=”G2”>
  <xsd:sequence>
  <xsd:element name=”Address” type=”xsd:string”/>
  <xsd:element name=”City” type=”xsd:string”/>
  <xsd:element name=”Region” type=”xsd:string”/>
 <xsd:element name=”PostalCode”  type=”xsd:string”/>
 <xsd:element name=”Country” type=”xsd:string”/>
 </xsd:sequence>
 </xsd:group>

   <xsd:element name=”NorthWind”>
   <xsd:element name=”OrderDetail”>

    <xsd:ComplexType>
     <xsd:element name=”Products”>
     <xsd:element name=”ProductID” type=”xsd:string”>
    <xsd:element name=”ProductName” type=”xsd:string”/>

      <xsd:ComplexType>
     <xsd:element name=”Suppliers”>
    <xsd:element name=”SupplierID” type=”xsd:string”>
     <xsd:ComplexType>

    <xsd:group ref=”G1”/>
    <xsd:group ref=”G2”/>
 <xsd:element name="HomePage" type="xsd:string"/>
</xsd:ComplexType>

</xsd:element>
</xsd:element>

<xsd:element name="Categories">
<xsd:element name="CategoryID" type="xsd:string">

 <xsd:ComplexType>
<xsd:element name="CategoryName"  type="xsd:string"/>
<xsd:element name="Description" type="xsd:string"/>
<xsd:element name="Picture" type="xsd:string"/>
</xsd:ComplexType>

</xsd:element>
</xsd:element>

 <xsd:element name="QuantityPerUnit" type="xsd:integer"/>
<xsd:element name="xsd:UnitPrice"  type="xsd:integer"/>
<xsd:element name="xsd:UnitsInStock" type="xsd:integer"/>
<xsd:element name="xsd:UnitsOnOrder"
                                         type="xsd:integer"/>
<xsd:element name="xsd:ReorderLevel"
                                        type="xsd:integer"/>
<xsd:element name="xsd:Discontinued"  type="xsd:string"/>
</xsd:ComplexType>

</xsd:element>
</xsd:element>

<xsd:element name="Orders">
<xsd:element name="OrderID" type="xsd:string">

<xsd:ComplexType>
<xsd:element name="Employees">
 <xsd:element name="EmployeeID" type="xsd:string">

<xsd:ComplexType>
<xsd:element name="LastName" type="xsd:string"/>
<xsd:element name="FirstName" type="xsd:string"/>
<xsd:element name="Title" type="xsd:string"/>
<xsd:element name="TitleOfCourtesty" type="xsd:string"/>
<xsd:element name="BirthDate" type="xsd:Date"/>
<xsd:element name="HireDate" type="xsd:Date"/>
<xsd:group ref="G2"/>
<xsd:element name="Homephone" type="xsd:string"/>
<xsd:element name="Extension" type="xsd:string"/>
<xsd:element name="Photo" type="xsd:string"/>
<xsd:element name="Notes" type="xsd:string"/>
<xsd:element name="Report To"  type="xsd:string"/>
</xsd:ComplexType>

</xsd:element>
</xsd:element>

<xsd:element name="Customers">
<xsd:element name="CustomerID"  type="xsd:string">

<xsd:ComplexType>
<xsd:group ref="G1"/>
<xsd:group ref="G2"/>
</xsd:ComplexType>
</xsd:element>
</xsd:element>
<xsd:element name="Shippers">
<xsd:element name="ShipperID" type="xsd:string">
<xsd:ComplexType>
<xsd:element name="CompanyName" type="xsd:string"/>

<xsd:element name="Phone" type="xsd:string"/>
</xsd:ComplexType>

</xsd:element>
</xsd:element>
```

```
</xsd:ComplexType>
</xsd:element>
</xsd:element>

</xsd:ComplexType>
</xsd:element>
</xsd:element>

</xsd:schema>
```

Figure 3. The Resulted XML schema document

6. Comparisons of complexity of the resulted XML schema

Many complexity measurement methods can be used to measure the quality of the XML schema documents (DTD or XSD). In this paper, the target XML schema is written with the XSD schema language and only XSD complexity measurement methods are used to show the complexity of the resulted XML schema. The count-based method is the earliest method which counts the number of element or attributes. The complexity metric is proposed by Dilke Basci and Sanjay Misra [4] which calculates not only provides the complexity due to all components of the schema file but also due to the imported components from other external schema file. They calculate the schema complexity with the following equation.

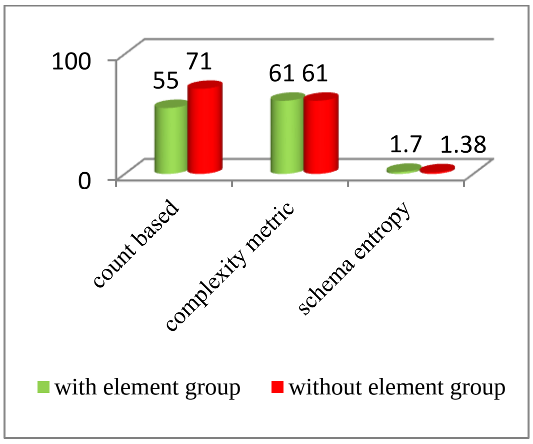
$$C(XSD) = C(Vg)+C(Gg)+C(Tg) \text{------}(2)$$

where C(Vg) is the total complexity values of all global elements and attributes that can be included/imported from external XSDs or can be declared/defined in the current XSD, C(Gg) is the total complexity values of unreferenced global elements and attributes group that can be declared/defined in the current XSD and C(Tg) is the total complexity values of unreferenced global complex and user-defined/built-in simple type definitions/declarations of XML schema document. The final measurement method is schema entropy (SE) [3] which measures the quality of the XML schema document with understandability, maintainability and reusability. They show the complexity values is increasing when the reusable component are increasing. They calculate the complexity with entropy concept which defined as:

$$SE= -\sum_{i=0}^{n} P(Ci) \log_2 P(Ci) \text{ -------}(3)$$

where n is the number of distinct classes. The table 3 describes the complexity measurement of the output schema document.

Table 3. Complexity comparison of the resulted XML schema document



7. Conclusion

In this paper, a method for the generation of qualified XML schema from relational database is proposed. Detecting the referential integrity (RI) is also presented with three steps which depend on database design to reduce the RI detection time. It has the advantages of redundancy free schema documents by providing the highly nested structure. It also maintains the original integrity constraints in relational database. Moreover, it provides code modularity, user understandability and reduces the maintainability effort by using the element group in the target XML schema document and it is shown with complexity measurements.

References

[1]Aminul Islam, Diana Inkpen, Iluju Kiringa, *"Applications of corpus-based semantic similarity and word segmentation to database schema matching"*, Volume 17 Issue 5, Springer-Verlag New York, Inc. Secaucus, NJ, USA, August 2008.
[2] Angela Cristina Duta, Ken barker and Reda Alhajj,, *"ConvRel: relationship conversion to XML nested structure"*, Proceedings of the 2004 ACM symposium on Applied computing, ACM New York, NY, USA, 2004.
[3] Dilek Basci and Sanjay Misra, *"Entropy as a Measure of Quality of XML Schema Document*", The International Arab Journal of Information Technology, Vol. 8, No1, January 2011.
[4] Dilek Basci and Sanjay Misra, "*Measuring and Evaluating a Design Complexity Metric for XML Schema Documents*", Journal of Information Science and Engineering 25, 1405-1425, 2009.
[5] Dongwon Lee, Murali Mani and Wesley W. Chu, "*Nesting-based relational-to XML schema translation", In Proceedings of International Workshop on the Web and Databases, 2001, pp. 61-66.
[6] Dongwon Lee, Murali Mani and Wesley W. Chu, "*NeT & CoT: Translating relational schemas to XML schemas using semantic constraints", In Proceedings of the 11th ACM International Conference on Information and Knowledge Management, 2002, pp. 282-291.
[7] Dogwon Lee , Murali Mani and Wesley W. Chu, *"Effective Schema Conversions between XML and Relational Models"*, In European Conf. on Artificial Intelligence (ECAI), Knowledge Transformation Workshop (ECAI-OT), 2002.
[8] Erik-T-Ray, "*Learning XML*", First Edition, January 2001, ISBN: 0-59600-046-4.
[9] Joseph Fong, Anthony Fong, HK Wong and Philip Yu, "*Translating relational schema with constraints into XML schema"*, International Journal of Software Engineering and Knowledge Engineering IJSEKE, Volume 16, Issue 2, 2006, pp 201-243.
[10] Kanagaraj.S and Dr. Sunitha Abburu, "*Converting Relational Database Into XML Document*", International Journal of Computer Science Issues, Vol 9, Issue 2, No 1, March 2012.
[11] Myint Myint Lwin, Thi Thi Soe Nyunt, Yuzana, "*Generating the Good XML Schema from Relational Database by using String Matching Algorithms*", 10th International Conference on Computer Applications, February, 2012, pp 357-361.