The System of XML Extraction from Relational Databases Based on Document Object Model

Thidar Win, Khin Than Kyi
Computer University (Kalay)
thidarwin1999@gmail.com, khinthankyi@gmail.com

Abstract

eXtensible Markup Language (XML) is a self-describing meta-language and fast emerging a standard for Web data exchange among various applications. XML is also a new technology that attracts a great deal of interest and dominates some areas of information system development. Relational database is a mature technology that dominates the implementation of database system. This paper implements the irregular varies structures to regular structures and can be extracted XML document by using the table-based mapping into relational databases. It proposes the system that extracts the XML document from the data stored in the database. It provides a generalised mapping between relational databases and XML document that is based on Document Access Definition (DAD) and the tree structure of the Document Object Model (DOM). In this paper, the system is proposed for storing nodes of input XML document as a relational tuple by node numbering scheme. It reduces the redundant components of XML document, collects and stores the elements and contents according to their same paths.

Keywords: XML, DAD, DOM Parser, XPath, Relational Databases.

1. Introduction

In recent years, the Web has been growing incredibly and has become main information interchange among various organizations. So also, XML is emerging as a standard for representing and exchanging structured and semi-structured data among applications over the Web. XML documents refer to the Internet resources, and XML database system should include the Internet resource management. XML is a new technology that attracts a great deal of interest to manage XML data as a database.

There have been many XML storage solutions based on the conventional database such as Relational Database Management System and Object Database Management System. Relational database is a mature technology that dominates the implementation of database system.

This paper describes the generalized mapping between relational databases and XML document based on the tree structure of the Document Object Model (DOM) and Document Access Definition (DAD). It provides a methodology for translating the conceptual schema of relational database into XML schema through DOM and DAD. It also describes RTX (Relational Translation to XML) which automates the major functions involved in the solution which achieves a mapping from relational database to XML format.

2. Background

In this subsection, we describe some database for storing structured documents. There have been several proposals on approaches to XML databases. Lore is an object-based storage in which all elements are stored as objects and linked by the use of labels [6]. However, most work has focused on storing XML in relational DBMS or indexes [4].The former approach has been the most popular due to the success and maturity of the relational DBMS. The latter approach is based on storing the XML document as building separate indexes on top [1]. Unlike the approach, these strategies do not use native layout of XML data and are limited to the generic optimization strategies built into relational databases. Some XML storage models have been proposed in recent years. The native storage of XML data has been addressed that XML documents are stored by splitting the XML tree into a tree of pages [7]. Each page corresponds to a disk block. In this manner, they reduce the number of blocks read to traverse the tree.

3. Document Access Definition (DAD)

A mapping file called the Document Access Definition (DAD) is used to map from relational data to XML document. DAD tools help to enable database and table columns for XML [3].

DAD file: It specifies to store data by using the XML column method, or defining XML collection for composition or decomposition.

Location paths: A location path specifies the location of an element or attribute within an XML document. The XML Extender uses the location path to navigate the structure of the XML document and locate elements and attributes.

For example, a location path of /BookCatalog /Book/Title points to the element as shown in the following example:

```
<BookCatalog>
        <Book>
            <Title>.
```

3.1 Mapping Relational Databases to XML Document

An XML document is only a database that is indexing and storing in the structured relational databases. There are two types of mappings as follow [5].

1. Object-Relational Mapping
2. Table-Based Mapping

3.1.1 Object-Relational Mapping

The object-relational mapping is used by all XML-enabled relational databases. It models the data in the XML document as a tree of objects that are specific to the data in the document. In this model, element types with attributes, element content, or mixed content (*complex element types*) are generally modeled as classes. The model is then mapped to relational databases using traditional object-relational mapping techniques. The DOM models are the documents itself and are the same for all XML documents, while the model describe above models the data in the document and is different for each set of XML documents that conforms to a given XML schema.

3.1.2 Table-Based Mapping

The table-based mapping is used by many of the middleware products that transfer data between XML document and relational database. It models XML documents as a single table or set of tables. The structure of the XML document must be as follows, where the <database> element and additional <table> elements do not exist in the single-table case.

```
<database>
    <table>
        <row>
        <column1>...</column1>
        <column2>...</column2>
        </row>
    </table>
    <table>...
    </table>
</database>
```

Figure 1. Table-Based Mapping of XML

It may be possible to specify whether column data is stored as child elements or attributes, as well as what names to use for each element or attribute. In addition, products that use table-based mapping often optionally include table and column meta-data at the start of the document or attributes of each table and column element. The term "table" is usually interpreted loosely. When transferring data from the database to XML, a "table" can be any result set or a "table" can be a table or an updateable view. The table-based mapping is useful for serializing relational data.

3.2 DOM Parser

A parser is an application that reads XML document and determines if it meets the XML syntax requirements to be well-formed.

DOM is both the programming interface hierarchy and method for locating and exposing data for manipulation and represents the tree view of the XML document. DOM is the only available method of XML data representation from the relational databases [2].

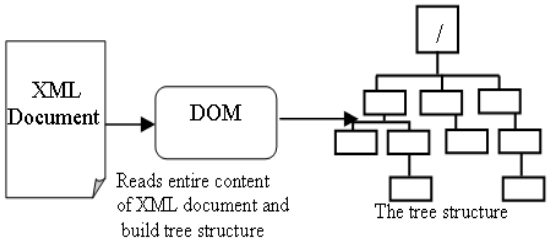

Figure 2. DOM Parser

4. Proposed System Design

In relational databases, data-set about XML document converted and stored in relational databases.

This system deals with DAD approach by which all multiple XML documents are collectively grouped according to their paths, and stored together in the repository. This system stores XML contents in user-defined types and map contents.

All root-to-leaf paths of XML document, where leaf is either attribute or element, are assigned a unique ID (i.e., path ID) and stored in the path table. Each node (i.e., text string, attribute value, and element) of the XML document is captured in the appropriate table separately.

As the result, the system is implemented as two outputs. First output is presented XML document by extracting from relational databases and the second can be viewed as the tree structure.

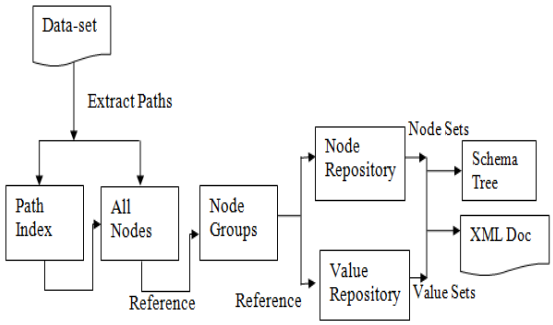

Figure 3. Proposed System Design

5. System Implementation

This system has been implemented two functions: Storing XML document to relational databases and Extraction XML document from the relational databases.

5.1 Storing XML Document to Relational Databases

The first function is the specification of different tables for relational database in order to store various structures into a system. The numbering scheme determines the ancestor-descendant relationship between elements in the hierarchy of XML data.

The following Figure is the sample XML document to store in the relational databases.

```
<xml version="1.0"?>
<BookCatalog>
    <Book>
         <Bookid="1001">
        <Title>Java</Title>
        <Authors>Peter</Authors>
        <Price>$5</Price>
    </Book>
    <Bookid="1002">
          <Title>HTML</Title>
        <Authors>Jone</Authors>
         <Price>$7</Price>
  </Book>
</BookCatalog>
```

Figure 4. Sample XML Document

XML document can be mapped into the following tables:

- *NODE* Table
- *PATH* Table
- *VALUE* Table

The *NODE* table stores elements and attributes of XML document. The actual values of elements and attributes are stored in the corresponding *VALUE type* table.

The data in the tables are inserted in a <column> tag for each column according to elements between tags ("<" and ">") of XML document.

It is defined four attributes: name, type, path and value.

Name: it specifies the name of the column that is created in the inside table.

Type: it indicates the SQL data type in the table for each indexed element or attribute.

Path: it specifies the location path in the XML document for each element or attribute to be indexed.

5.1.1 The Schema of Tables

In order to store XML document in relational database tables are constructed by using table-based mapping. These tables are classified into path table, all nodes table, same node groups table, node repository and value repository.

(1) Path Table

In this table, it maintains path expressions. The leaf nodes of the index tree contain references pointing to the associated node groups.

The hierarchical relationship between nodes is expressed by operators "/". It is the XPath to specify the content of Path Expression and Path Identifier.

Eg: /BookCatalog/Book

Table 1.XPath Expression and Path identifier

Path Expression	Path ID	NodeID
/BookCatalog	101	1
/BookCatalog/Book	102	1.1, 1.2
/BookCatalog/Book/@id	103	1.1.1, 1.2.1
/BookCatalog/Book/Title	104	1.1.2, 1.2.2
/BookCatalog/Book/Auth	105	1.1.3, 1.2.3
/BookCatalog/Book/Price	106	1.1.4, 1.2.4

XPath is a mechanism to locate a particular node or portion inside the XML document's tree structure. It is like a "file path" used in computer's file system to locate files.

Table 2. All Nodes in Storage System

Node ID	Parent Node ID	Node type	NodeName	Value
1	0	Element	BookCatalog	
1.1	1	Element	Book	
1.1.1	1.1	Attribute	@id	1001
1.1.2	1.1	Element	Title	Java
1.1.3	1.1	Element	Author	Peter
1.1.4	1.1	Element	Price	$5
1.2	1	Element	Book	
1.2.1	1.2	Attribute	@id	1002
1.2.2	1.2	Element	Title	HTML
1.2.3	1.2	Element	Author	Jone
1.2.4	1.2	Element	Price	$7

Table 2 shows all nodes in the storage system.

(2) Same Node Groups

Node groups, a set of node-references are grouped together taken from the same paths belong to the document.

Table 3. Same Node Groups Table

Path ID	Node ID	Parent ID	Same Node Groups
101	1	0	BookCatalog
102	1.1	1	Book
102	1.2	1	Book
103	1.1.1	1.1	@id
103	1.2.1	1.1	@id
104	1.1.2	1.1	Title
104	1.2.2	1.2	Title
105	1.1.3	1.1	Author
105	1.2.3	1.2	Author
106	1.1.4	1.1	Price
106	1.2.4	1.2	Price

(3)Node repository

Node repository maintains root node and all the internal nodes with the relationship of parent node and child node.

Table 4. Node Repository

Path ID	Node ID	Parent Node ID	NodeName	Node Child ID
101	1	0	BookCatalog	1.1,1.2
102	1.1	1	Book	1.1.1,1.1.2,1.1.3,1.1.4
103	1.1.1	1.1	@id	
104	1.1.2	1.1	Title	
105	1.1.3	1.1	Author	
106	1.1.4	1.1	Price	
102	1.2	1	Book	1.2.1,1.2.2,1.2.3,1.2.4
103	1.2.1	1.2	@id	
104	1.2.2	1.2	Title	
105	1.2.3	1.2	Author	
106	1.2.4	1.2	Price	

(4) Value repository

Value repository contains the external node values.

Table 5. Value Repository

Path ID	Parent ID	Node Value
103	1.1.1	1001
103	1.2.1	1002
104	1.1.2	Java
104	1.2.2	HTML
105	1.1.3	Peter
105	1.2.3	Jone
106	1.1.4	$5
106	1.2.4	$7

5.2 Extracting XML from the Database

The second function is that extraction process could be taken to achieve the input/update process. It describes a simple relational schema for a document. The extraction of data from database relies on a mapping between the stored tables of parent-child relationship and the respective portions of the database schema.

The following Figure 5 is tree representation taken from Figure 4 of sample XML document. Each node in the schema tree is assigned a unique identifier.

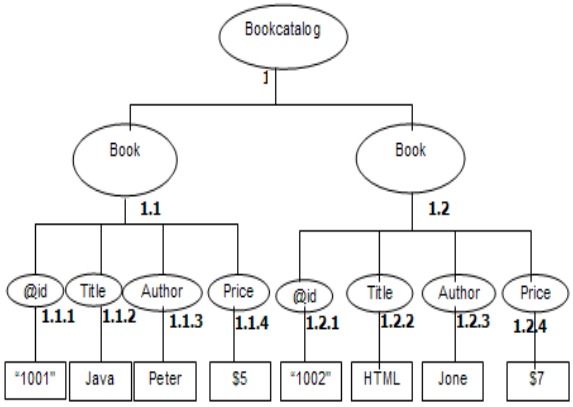

Figure 5. Tree-like for XML Document

5.2.1 Output Generation as a Tree

The user would indicate which data is to be extracted from the desirable portions of the database. The output XML document, extracting from relational databases, can be transferred into tree structure by using DOM. It provides the parent-child relationship of a pair of nodes.

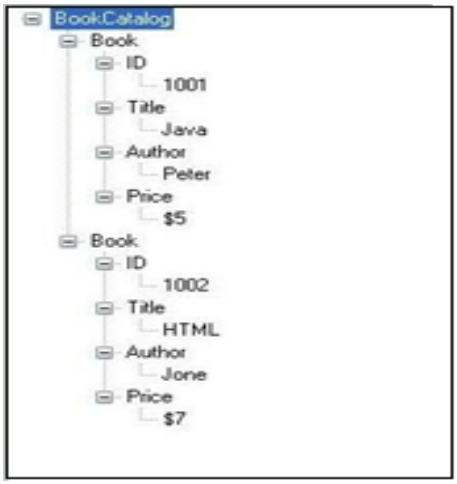

Figure 6. Schema Tree Based on DOM

6. Conclusion

XML is semi-structured data (irregular structure). The Web page using XML has large irregular data-set and it is difficult to transport on Web. In this paper, the system is introduced by mapping between XML and relational data models. It can be presented a generalized means of extracting XML data from relational database storing of XML data. The main features of this extraction mechanism are the production of the XML formatted based on the database structure and specified in a schema, expressed in an XML schema language. It implements the data characteristics and data usage models, to intend the impact storage result by the use of table-based mapping method which contains multiple tables. The storage result is implemented by the number of tables. The fundamental point of the system is to make the reader understand the fact that XML data can be always stored to relational database, and relational data can be always published as XML documents. Finally, this system is generated to the tree structure of XML by using DOM. XML indexing and storage system from relational database is implemented with DAD, DOM parser, Xpath and other useful methods.

7. References

[1] A.V.D Lee, W.C.Lee, and P.L.P.State, "Supporting XML Security Models using Relational Databases: ".

[2] A.Salminen Dept. of Computer Science and Information Systems, "Requirements for XML Document Database Systems:", Department of Computer Science University of Waterloo Waterloo, ON, Canada .

[3] B.Lewis lewisba@cs.latrobe.edu.au, "Generalised XML extraction from relational databases based on ad-hoc XML schema".

[4] E.P.Lim, W.K.Ng, "ENAXS: Efficient Native XML Storage System", NTU Nanyang Avenue, N4-B3C-13, Singapore.

[5] Igor, Stratis D, "Storing and Querying Ordered XML Using a Relational Database System".

[6] W.S.Han , K.H.Lee and B.S. Lee, "An XML Storage System for Object-Oriented/ Object-Relational DBMSs".

[7] M.Nicola, Kane.et.al, "Native XML Support in DB2 Universal Database ", Bert van der Linden IBM Silicon Valley Lab.