Journal of Computer Science 7 (2): 314-319, 2011
ISSN 1549-3636

An Efficient Middleware for Storing and Querying XML Data in Relational Database Management System

Mohammed Adam Ibrahim Fakheraldien, Jasni Mohamad Zain and Norrozila Sulaiman
Faculty of Computer Systems and Software Engineering,
University Malaysia Pahang, Lebuhraya Tun Razak, 26300 Kuantan, Pahang, Malaysia

Abstract: Problem statement: In this study, we propose a middleware that provides a transformation utility for storing and querying XML data in relational databases using model mapping method. **Approach:** To store XML documents in RDBMS, several mapping approaches can be used. We chose structure independent approach. In this middleware the model mapping method XParent and free of cost available technologies MYSQL, PhpMyAdmin and PHPclasses are used as examples. **Results:** This middleware stores XML tables and does not require a direct extension of SQL thus this middleware can be used with any relational databases management system with little changes in the middleware interface. The middleware offers two alternative methods -namely XParent and XReal- for storing XML in the database. **Conclusion:** The key to proposed middleware is to store XML document in a relational database through a user interface and with an XPath query processor. We present a comparative experimental study on the performance of insertion and retrieval of two types of XML documents with a set of XPath queries executed though the XPath. XML and Relational databases cannot be kept separately because XML is becoming the universal standard data format for the representation and exchanging the information whereas most existing data lies in RDBMS and their power of data capabilities cannot be degraded so the solution to this problem a middleware prototype is required. The proposed schema dependent solutions have a drawback that evens a small change in the logical structure of XML documents influence on the database schemas and several problems occur during the updating process. A new efficient data middleware is proposed in the study to face these issues.

Key words: Extensible Markup Language (XML), relational database, XParent, Structured Query Language (SQL), middleware, Database Administrators (DBAs)

INTRODUCTION

Today's data exchange between organizations has become challenging because of the difference in data format and semantics of the meta-data which used to describe the data. Now day' XML emerged as a major standard for representing data on the World Wide Web while the dominant storage mechanism for structured data is the relational databases, which has been an efficient tool for storing, searching, retrieving data from different collection of data. The ability to map XML data in relational databases is difficult mission and challenging in the world of all IT organization so there is a need to develop an interfaces and tools for mapping and storing XML data in relational databases.

Taking up emerging requirements, database vendor such as IBM, Oracle and Microsoft are enabling their product for XML. There is a need arise to manage XML data and other data stored in relational data seamlessly at a time efficiently. The native-XML databases usually have limited support for relational data. XML -Enabled databases like IBM, Oracle and Microsoft have mature and proven techniques for relational data processing but XML-extensions have not been mature enough yet. In these vendor specific RDBMS; Database Administrators (DBAs) have to express how to map XML data into their systems and the XML storage are tailored to one particular system and are hard-coded to some default mapping on behalf of the users, so they cannot be used for any other relational backend. For solution to these problems a middleware is required for storing and querying xml data in any RDBMS.

XML: The extensible Markup Language (XML) is quickly becoming the de facto standard for data

Corresponding Author: Mohammed Adam Ibrahim Fakheraldien, Faculty of Computer Systems and Software Engineering, University Malaysia Pahang, Lebuhraya Tun Razak, 26300 Kuantan, Pahang, Malaysia

exchange over the Internet and now it plays a central role in data management, transformation, and exchange. Since it s introduction to industry in the late 1990s, XML (Amirian and Alesheikh, 2008) has achieved widespread support and adoption among all the leading software tools, server, and database vendor s. As importantly, XML has become the lingua franca for data by lowering the cost of processing, searching, exchanging, and re-using information. XML provides a standardized, self-describing means for expressing information in a way that is readable by humans and easily verified, transformed, and published. This allows both information workers and automated applications to better find and uses the information they need. In addition, data can be transmitted to remote services anywhere on the Internet using XML-based Web services to take advantage of the new ubiquity of connected software applications. The openness of XML (Augeri *et al*., 2007). allows it to be exchanged between virtually any hardware, software, or operating system. Simply put, XML opens the door for information interchange without restriction. For its features in good description and transmission, the hot topic is to seek the best way for storing XML documents in order to get high query processing efficiency. At present, storing XML document in relational database is a promising way for that relational database is mature.

Relational Databases: Today, the dominant storage mechanism for structured enterprise data is the relational database, which has proven itself an efficient tool for storing, searching for, and retrieving information from massive collections of data. Relational databases specialize in relating individual data records grouped by type in tables. Developers can join records together as needed using SQL (Structured Query Language) and present one or more records to end-users as meaningful information. The relational database model revolutionized enterprise data storage with its simplicity, efficiency, and cost effectiveness. Relational databases have been prevalent in large corporations since the 1980s, and they will likely remain the dominant storage mechanism for enterprise data in the foreseeable future. Despite these strengths, relational databases lack the flexibility to seamlessly integrate with other systems, since this was not historically a requirement of the database model (Wilson, 2001). In addition, although relational databases share many similarities, there are enough differences between the major commercial implementations to make developing applications to integrate multiple products difficult. Among the challenges are differences in data types, varying levels of conformance to the SQL standard, proprietary extensions to SQL, and so on.

MATERIALS AND METHODS

Structure Independent Mapping Approach: The structure independent mapping approach is explained with a sample XML document shown in Fig. 1. In this study, we employ the data model of XPath (Haw and Lee, 2008) to represent XML documents. In the XPath data model, XML documents are modeled as an ordered and directed labeled tree. There are seven types of nodes. In this study, we consider only the following four types of nodes for the sake of simplicity: root, element, text and attribute. The root node is a virtual node pointing to the root element of an XML document.

The XRel Method: This approach (Association for Computing Machinery, 2001) stores XML documents in four different tables; Element table stores only the document structure. Text table holds only text data. Attribute table stores attribute values. Path table keeps unique paths in XML documents. The key to this method is the path table and regions associated with the inner nodes in the tree.

Element (DocID, PathID, Start, End, Index)
Attribute (DocID, PathID, Start, End,Value)
Text (DocID, PathID, Start, End, Value)
Path (PathID, Pathexp)

The database attributes DocID, PathID, Start, End, and value represent document identifier, start position of a region, end position of a region, and string-value, respectively. Each node is associated with start and end positions. A region (or the pair of start and end positions) implies a containment between elements with regards to the ancestor-descendant and parent-child relationships. For example, a node, ni, is reachable from another node nj, if the region of ni is included in the region of nj. In our study, we modified some attributes of tables in XRel approach: we added a ParentID column to Element, Attribute, and Text tables to find parent nodes easily. We put NodeID and last descendant node id (EndDescID) attributes instead of start and end columns in Element table. The XRel four tables (Path, Element, Attribute and Text) can be seen in Table1-4 for the XML document given in Fig. 1as following:

Table 1: Path table

Path	PathEXP
1	/address_book
2	/address_book/card
3	/address_book/card/@no
4	/address_book/card/name
5	/address_book/card/name/surname
6	/address_book/card/name/given
7	/address_book/card/name/other
8	/address_book/card/title
9	/address_book/card/address
10	/address_book/card/address/street
11	/address_book/card/address/city
12	/address_book/card/address/state
13	/address_book/card/address/zip
14	/address_book/card/name/contact
15	/address_book/card/contact/phone

Table 2: Element table

Nod ID	EndDesc ID	Path ID	Parent ID	Ordinal
1	51	1	0	1
2	25	2	1	1
5	11	4	2	1
6	7	5	5	1
8	9	6	5	1
10	11	7	5	1
12	13	8	2	1
14	22	9	2	1
15	16	10	14	1
17	18	11	14	1
19	20	12	14	1
21	22	13	14	1
23	25	14	2	1
24	25	15	23	1

```
<address_book>
 <card no="1">
  <name>
        <surname>Adam</surname>
 <given>John</given>
        <other>Tom</other>
        </name>
          <title>Prof.Dr<title>
        <address>
<street>3rd Floor, C11 ABC</sreet>
         lakeside</street>
        <city>Nyala</city>
        <state>AB<state>
        <zip>12345</zip>
        </address>
         <contact>
        <phone>989937</phone>
          </contact>
</card>
   </address_book>
```

Fig. 1: Sample XML

Table 3: Attribute table

Nod ID	Path ID Value	Parent	ID
3	3	2	1

Table 4: Text table

Nod ID	Path ID	Parent ID	Value
7	5	6	Adam
9	6	8	John
11	7	10	Tom
13	8	12	Prof.DR
16	9	15	3rd Floor, C11 ABC
18	10	17	Nyala
20	11	19	AB
22	12	21	12345

The XParent Method: XParent (Jiang *et al*., 2002) is a model mapping schema. Like Xrel, XParent uses the Path index table, but it uses the Edge and Edge-value approach to store the parent-child relationship. This relationship is maintained in a separate table (DataPath), again, to reduce the join cost. XParent is chosen because it outperforms significantly current state-of-the-art model mapping approaches like Edge and particularly for query performance (Ali, 2006). The XParent schema stores the node information of the XML data graph of an XML document into four tables:

LabelPath (ID, Len, Path)
DataPath (Pid, Cid)
Element (PathID,Did,Ordinal)
Data (PathID, Did, Ordinal, Value)

Table LabelPath stores the information of label-paths of an XML data graph. The attributes ID and Len denote the unique ID and the length of each label-path, respectively. The attribute Path denotes the name of the corresponding label-path which is a sequence of node names in the label-path. Table DataPath stores the information of parent-child relationships of an XML data graph. The attributes Pid and Cid denote the node number of the corresponding parent node and child node of an edge, respectively. Table Data stores the information of the nodes of the XML data graph if the corresponding elements or attributes in an XML document have a value. The attributes PathID and Did denote the ID of label-path (i.e., the foreign key of the ID in Table LabelPath) and the node number of the node, respectively. The attribute Ordinal denotes the ordinal number of the node among its sibling-nodes with the same name. The attribute Value denotes the value of the node, where the value of anode is the value of the corresponding element or attribute in the XML document. Table Element is similar with Table Data, except that Table Element does not store the value of each node.

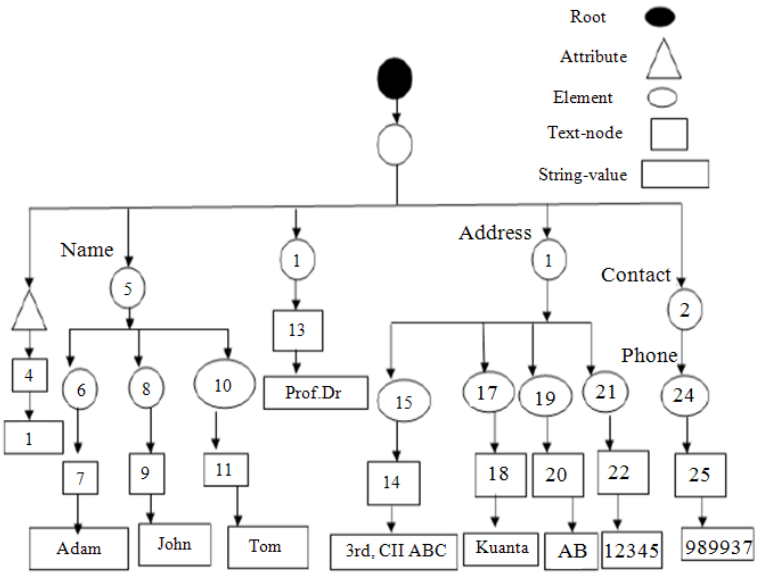

Fig. 2: T he XML Tree the sample XML

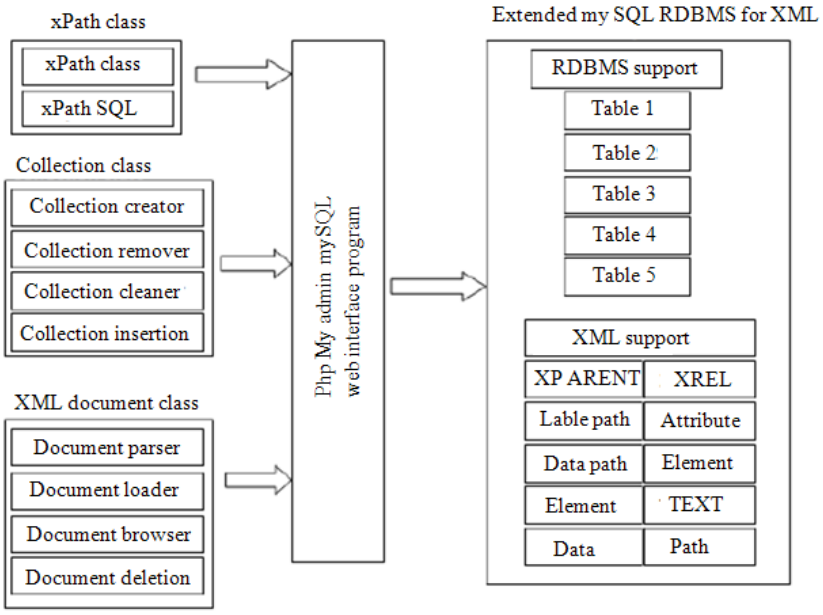

Fig. 3: Middelware structure

The middleware architecture and its implementation: The main idea for the implementation of the prototype is taken from (Şevkli *et al*., 2004) with modification of usage of available mapping strategy XParent instead of other method. We chose MYSQL as the DBMS for storing and retrieving XML documents using structure independent mapping approach because it is free of cost, open source and easily available. This implementation adds collection support to the MYSQL database. A collection is a set of XML document stored in fixed schema of tables. In this case it a set of fixed schema tables according to the proposed by XParent (Fakhraldien *et al*., 2010)In reality, MySQL database model does not change, but from the user point of view, inside a database there is not only tables but also collections. In addition, the users can not modify or

access to the tables of collection directly. Users know the existing collection names and types only. They can create, drop or browse collections. The users can insert, browse or delete XML documents into collections. The independent classes in PHP can be created and embedded to PhpMyAdmin program which is a web based interface between MySQL and the users. The PhpMyAdmin program provides all database operations with user friendly web interface. In these Middleware we used two different independent mapping approach methods. There are variations of these methods as well. We chose the XParent and XRel method, because XParent is a four table database schema (LabelPath, DataPath, Element and Data).DataPath keeps parent-child relationalships while XRel does not explicitly stores edges, for data paths. Instead, XRel records containment relationships using the notion of region. Therefore, it needs joints in order to check edge connections .The design objective of this middleware is to provide efficient software that can use commercially available RDBMS to manage XML documents. After the implementation it will become repository for both XML documents and relational data. The Fig. 2 and 3 bellow outlines the architecture of the middleware which adds XML support to the MYSQL database system. The three main classes (Collection, Document, and XPath) can be used to adapt the same interface to other database systems.

RESULTS

The proposed middleware can be use as an efficient solution with respect to query processing specially recursive XML quires and updating. In comparison to other middleware's which used in storing and querying XML documents in relational database this middleware can act as efficient mediator between XML and r relational database.

This experimental discusses the result of storage and retrieval time of XML documents and a set of XPath queries using XParent and XRel methods. Comprehensive experiments were conducted to study the performance of XParent, in comparison with other approaches. While among those RDBMS-based approaches, XParent perform significantly better than other model-mapping-based approaches such as Edge and XRel. One observation is that RDBMS-based approaches can outperform special-purpose XML repositories such as Lore and Tamino. All experiments were conducted on p4 350 MHz PC with 1 GB RAM, 40GB hard disk, windows XP using the middleware. Figure 4 presents some sample results of experiments with

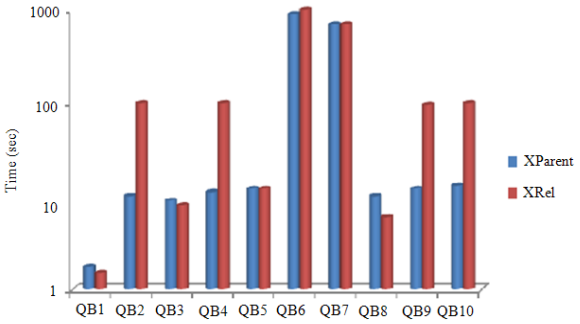

Fig. 4: Total Elapsed Time: Xparent Vs. XRel using BENSHO.4

Table 5: Quires sample

QB1	/site/regions/namerica/item[@id="item12345"]/name
QB2	/site/open_auction/open_/bidder[1]/in crease/
QB3	/site/open_auction/open_auction[bidde r[persone/@person="person6789"]/foll owingsibling:: bider[person/@person=person3 4567"]]/reserve/
QB4	Count(/site/closed_auctions/closed_auc tion[price>=50])
QB5	Count(/site/regions//items)
QB6	Count(/site//descripiti /site//annotation /site//email)
QB7	/site/regions/*/item[contains(descriptio n,"excellent")/name
QB8	/site/closed_auctions/closed_auction/a nnotation/description/parlist/listitem/p arlist/listitem/text/keword/empo
QB9	/site/closed_auctions/closed_auctions[a nnotation/description/parlist/listitem/p arlist/listitem/text/emph/keword/]/sell er/@person
QB10	/site/people/person/[not(homepage/te xt()/name/

XParent and XRel using the Benchmark database (with factor 0.4 (BENCH0.4), while table 5 present the ten XPath queries which used for the comparison between the two methods.

It can be seen that, XParent outperforms XRel for most of the queries. In certain cases, such as QB2, QB4, QB9 and QB10, XParent can be faster than XRel up to 15 times (QB2), for instance.

DISCUSSION

From the presented results, the proposed middleware can be use as an efficient way for storing and queering XML data in relational database. This middleware has the following unique features.

XML data is stored in relational tables according to the XParent mapping-schema, an efficient, model-mapping approach without assistance of DTD.

- The visual query interface of XParent provides both expressive powers for professionals and user friendliness for native users.

Overall the performance of XRel and XParent methods is comparable in most cases with exception of long XPath queries where XRel is definitely faster.

We can say that the XParent method can certainly be considered for query processing in most cases. We think that the execution times for queries processing are adequate and comparable to commercial applications and databases. The middleware has flexibility to add any future proposed more efficient schema-oblivious mapping straggly as new collection. The mediator can also be used as benchmark tool for the researchers to compare various model mapping XML schemas by adding them as collection. Our implementation adds collection support to XML into the MYSQL database. A collection is a set of similar XML documents stored in fixed schema of tables, sometimes referred to as XML repository. In realty, MYSQL database model does not change, but from the user point of view, a database can contain tables and collections.

CONCLUSION

Based on experiment and result, it shows that the proposed middleware can be used as an efficient, affordable and quick solution until XML data processing matures. The key to Middleware approach is storing XMIL documents in the relational databases, providing a user interface for XML manipulation and adding an XPath query processor for XNML querying. The Middleware impleinented in this study can be used with any other database management system as it doesn't require any modification to DBMS itself. It provides collections or XML repositories to store XML documents in a database. Organizer should look for developing middleware's to store and quire XML documents into relational databases. That middleware should store XML files directly into a relational database by integrity and efficiently way.

REFERENCES

Ali, A.A., 2006. On optimistic concurrency control for real-time database systems. Am. J. Applied Sci., 3: 1706-1710. DOI: 10.3844/ajassp.2006.1706.1710

Amirian, P. and A.A. Alesheikh, 2008. Publishing geospatial data through geospatial web service and XML database system. Am. J. Applied Sci., 5: 1358-1368. DOI: 10.3844/ajassp.2008.1358.1368

Association for Computing Machinery, 2001. ACM transactions on Internet technology. 1st Edn., Association for Computing Machinery, USA.,

Augeri, C.J., D.A. Bulutoglu, B.E. Mullins, R.O. Baldwin and L.C. Baird, 2007. An analysis of XML compression efficiency. In Proceedings of the 2007 Workshop on Experimental Computer Science, June 13-14, ACM New York, NY, USA., DOI: 10.1145/1281700.1281707

Fakhraldien, M.A.I., J.M. Zain and S.N. Ihsan, 2010. A middleware prototype for storing and quering XML documents in RDB using XParent model mapping schema. Proceeding of the International Conference in Electronic and Information Engineering, Aug. 1-3, Kyoto, pp: 560-563. DOI: 10.1109/ICEIE.2010.5559745.

Haw, S.C. and C.S. Lee, 2008. TwigINLAB: A decomposition-matching-merging approach to improving XML query processing. Am. J. Applied Sci., 5: 1199-1205. DOI: 10.3844/ajassp.2008.1199.1205

Jiang, H., H. Lu, W. Wang and J.X. Yu, 2002. Path materialization revisited: An efficient storage model for xml data. Proceeding of the 13th Australasian Database Conference, (ADC'02), Australian Computer Society, Inc. Darlinghurst, Australia, pp: 85-94. DOI: 10.1145/563932.563916

Şevkli, Z., M. Mercan and A. Kurt, 2005. A middleware approach to storing and querying xml documents in relational databases. Adv. Inform. Sys., 3261: 223-233. DOI: 10.1007/978-3-540-30198-1_23

Wilson, J., 2001. Take a good look. 1st Edn., Puffin Books, USA., ISBN0141309423, pp: 90.