A Comparative Analysis of Managing XML Data in Relational Database

Kamsuriah Ahmad

Faculty of Information Science and Technology
National University of Malaysia
kam@ftsm.ukm.my

Abstract. The eXtensible Markup Language (XML) has recently emerged as a standard for data representation and interchange on the web. Based on its popularity used in most application, the critical issues are to store and to query XML data to exploit the full power of this technology. Since relational database is widely used technology for storing and querying, therefore replacing it with pure XML database is not a good choice and very expensive process. It is thus crucial to map XML data into relational data and this process is one that occurs frequently. Many existing methods exist in the literature, and defining what the best mapping method is explicitly important. The intention of this paper is to the existing mapping methods in terms of generating good relational schema. At the end a new mapping method is developed to overcome the limitations the limitations and shows that it is efficient in terms of removing relation redundancy.

Keywords: Mapping method, comparative analysis, XML, relational database.

1 Introduction

XML has emerged as a standard data representation and interchange on the web. However, relational database is often used to store the data based on its popularity and reliability. Some mechanism is needed to map XML data to relational data. Until today there is no fast, easy, automated, and most important, free solution existed on the mapping from XML to relational. Currently, there are several approaches for storing XML data, ranging from using files to full-fledged database management systems such as relational, object-relational, object-oriented or native XML database systems. The methods can be classified according to the following categories:

i.Files: This approach takes an entire XML document as a single object in a file and directly mapped and stored as one big text columns such as CLOB (Character Large Object), in much the same way as to store an image file. The advantages of storing XML as CLOB are: there is no preprocessing and the originality of XML document is preserved. However the disadvantage of this approach is that we cannot get help from the database management system: not able to search effectively on the document contents, or to do queries since the data has to be parsed, loaded into memory, processed and, for update operations, we need to dump back to disk. It might cause a large number of nested tables and redundant data in the relations. Therefore the processes are not efficient and impractical.

ii.Native Database System: In this approach a native XML database was developed to directly support XML data model and queried by special purpose engines. The advantages of these approaches are that XML data can be stored and retrieved in their original formats and no additional mappings or translations are needed. Furthermore most native XML databases have the ability to optimize the query techniques. The disadvantages of this approach are that, due to the document-centric nature of XML database, complex searches or aggregation might be cumbersome. On the other hand since XML may contain lots of redundant data then the update operations are not efficient. As reported, the current state of native XML database is still unsettled and does not have any firm standard on its structure [19], this makes it unsuitable to manage huge XML data.

iii.RDBMS: In this approach, XML data is mapped into rows and columns of a relational table, which is known by the term shredding. The queries posed in semi-structured query languages are translated into SQL queries. The results of these queries are later translated back to XML, where all the processes are done internally. Therefore, the processes of mapping XML to relational tables occur frequently. Database vendors such as IBM, Microsoft, Oracle and Sybase are currently building tools to assist in mapping XML documents into relational tables. These vendors are competing against one another and a standard for the XML data type and the methods of mapping is yet to be defined [3].

Different approaches, either files, native database, or DBMS have their own strength and weaknesses [3]. It is still unclear which of these three approaches is going to find a wide-spread acceptance. In theory, native database systems should work best, but it is going to take a long time before such systems are mature and scale well for large amounts of data. Furthermore it is argued that XML can be effectively used as a database language. This is because XML language is best in supporting other applications, such as user-defined tagging documents, cross-referencing between documents and in fact it just stands as a database that store trees. Thus, XML will never be an ideal database language [15]. Therefore a more efficient database language for XML is needed, and relational database language is the best alternatives. Relational database systems are mature and scale very well, and they have the additional features that XML data can co-exist, making it possible to build applications with little extra effort. To optimize the use of relational database systems for managing XML data, recent work has concentrated on models and methods to map from XML to relational. It is believed that the effort on finding the best mapping algorithm will still continue in the future. Currently many existing methods exist in the literature on the mapping process. This gives rise to the following problem: Are some mappings ‘better’ than the others? To approach this problem, the classical relational database design through normalization technique is referred. The technique is based on the functional dependency that exists in the semantics of the data. Therefore in this study we made an approach that the mapping method is considered to be in good criteria if it can produce relational schema without relation redundancies that may lead to anomaly problems. This paper will study and make comparison of the existence approaches and discuss why the existence approaches are still insufficient for reducing relation redundancy and failed to achieve the optimal relational schema for XML. This paper is organized as follows: section 2 will discuss different ways to map XML to relations, section 3 will discuss the proposed mapping method (XtoR), section 4 provides a motivating example and compares the relational schema generated by XtoR and the existing methods. At the end the conclusion and future enhancement are presented.

2 Managing XML Data in Relational Database

The mapping from XML to relational is not an easy task to accomplish because the data model of an XML document is fundamentally different from that of a relational database. Especially, the structure of an XML document is in hierarchy, and the XML elements may be nested and repeated, while relational model is a flat representation of data with tables and columns. During the data exchange, XML might come with or without a schema (DTD or XML Schema). The existence or the absence of a schema greatly influences the mapping procedure. When the schema of XML data is not available, a generic mapping is used. XML document can be seen as a tree model and the mapping is based on the relationship between the nodes and edges of XML model. But when a schema is available structural constraint information of an XML document from a schema is used to guide the mapping design. Recently, studies in the context of integrity constraint for XML paying particular attention to the class of keys and functional dependencies [15] as renewed interest to adopt these constraints in the mapping framework. It is believed that if the mapping considers the presence of semantic constraints then the relational schema generated will be good. Therefore, the mapping approach can be discussed into three different categories: (i) model-based approach, (i) structural-based approach, and (iii) semantic-based approach.

2.1 Model-Based Approach

This mapping is based on path expression in the XML tree in the absence of schema type. Basically this approach will traverse the tree and store the path for every node visited in a table. Even though the main idea is the same but various strategies have been proposed that improved from the previous one. Among the approaches that fall under this category are Edge [4], XRel [18] and XPev [14]. However these three approaches are in the absence of schema type, therefore relational table is used to store the path information. The information is based on the structure of XML tree and ignores totally the semantics aspect of XML. The problem with relational schema generated by these methods is that the information is split into small pieces that may end up increasing the storage size of the database. In fact most of the targetIDs are zeroes and lots of data duplication in ordinal column as in Edge approach. Furthermore, in order to process query faster, an index (tag, data) is mandatory and more tables need to be joint during the query processing. Since the methods do not consider the semantics constraints, the criteria of generating good relational schema that reduced redundancy is not achieved.

2.2 Structural-Based Approach

This approach is based on the existence of type definition such as XML DTD or XML Schema, which conforms to XML document. By analyzing the structural properties of the schema, it then automatically converts a DTD or XML Schema into relational schemas. DTDs may contain arbitrary regular expressions, such as recursive, disjunction and set value, which need further analysis. The approaches that can be classified under these categories are Inlining [16], LegoDB [9] and CPI [10]. However the semantic keys are not used efficiently where system generated ID, parentID and parentCODE are used widely. Studies as shown that [11] these IDs will generate data and relation redundancy with respect to the constraints exists in XML documents. This approach do not take into account the information about semantic dependencies, therefore it is not efficient in reducing the redundancy of data that may exist in XML.

2.3 Semantics-Based Approach

Since studies on integrity constraints for of XML data started to emerge, there have been efforts considering capturing semantics of XML for the mapping. These semantics constraints are provided along with XML documents and become an input to the system. The constraints information guides the design of relational schema during the mapping processed. The constraints used are keys, foreign keys, and functional dependencies both in the absence or presence of the schema. As been proved in relational database, when semantics are given, redundancy can be reduced with respect to the constraints through normalization process. The mapping under this category can be divided into two: key based approach (X2R [6], Davidson [7], Liu [12]) and functional dependency based approach (Lv&Yan [13], RRXS [3]).

Table 1 shows a comparison for the mapping approaches. The mapping methods under semantic-based approach used semantic constraints that come with the XML documents. X2R mapping method used the information in key and foreign key constraints and used these constraints to guide the schema design. Even though this method able to generate a reduce redundancy relational schema with respect to the semantics in keys and keyrefs but the general concept of functional dependencies (that able to express dependency constraints among the attributes) is ignored, hence they failed to produce good relational schema for XML with respect to the constraints. Furthermore, the study of XFD implication is avoided; therefore it cannot infer other constraints that may exist, given the existing constraints. As the result, other semantics relationships of XML document cannot be identified. System generated ID is used as an ID to the relation and parentID is used to express the relationship between parent and child. Hence this method will generate relational table with data and relation redundancy. Study has shown that if the system generated ID is used when designing the relational schema instead of the given value-based keys, it will decrease the query performance [12]. Davidson's method performs the mapping in the presence of key constraints. The relational schema is mapped from a set of minimum covers propagated from key constraints. But again this approach does not consider functional dependencies therefore it cannot remove redundancies that may exist in XML. Hence this redundancy cannot be captured and it cannot produce a good relational schema. In relational, functional dependencies constraint is useful for generating optimal schema decomposition for XML thru its normalization step. Ironically, this constraint is the most neglected aspect by many researchers as the approach in model-based and structural-based. The model-based approach as in Edge, XRel and XPev claims to be efficient in processing the queries in the absence of schema definition but this approach ignores totally the semantic aspects in XML. Therefore this approach will produce redundancies in the schema, hence the criteria of producing an optimal relational schema for XML is not achieved.

Table 1. Comparison of the existing XML-Relational Mapping Techniques

	Schema	Model-based	Structural-based	Semantics-based
Edge	No	Path-based	No	No
XRel	No	Path-based	No	No
XPev	No	Path-based	No	No
Stored	Yes, DTD	No	Yes	No
Inlining	Yes, DTD	No	Yes	No
LegoDB	Yes, DTD	No	Yes	No
CPI	Yes, DTD	No	Yes	No
XtoR	Yes, XML-Schema	No	Yes	Yes
Davidson	No	No	No	Yes
Liu	Yes, DTD	No	No	Yes

The structural-based approach as in Inlining, LegoDB and CPI claims to be efficient in processing the queries. In the presence of type definition such as DTD or XML Schema, they need to deal with these type definitions that may contain arbitrary regular expressions, such as recursive, disjunction, set value and also to deals with null values and incomplete relations. Two evaluations studies [9], [21] on alternative storage strategies indicate that the shared-inlining algorithm [20] outperforms other strategies in data representation and performance across different datasets and different queries, when DTDs are available. The studies also indicate that the presence of DTD during the mapping is vital to achieve good performance and compact data representation of XML in relational settings across different datasets and different queries. This is one of the reasons DTD is included in our studies, and the XFDs is defined with respect to this schema. But most of the mapping approaches ignore the existence of semantics as expressed in functional dependencies, therefore the resulted relational schema may contain redundancy in the relations. It would be helpful if tools exist to facilitate the general problem of mapping between different data formats, taking the semantics of data into account. To facilitate such mapping we need a language in which to express transformations and constraints, and the ability to reason about the correctness of the transformations with respect to the constraints.

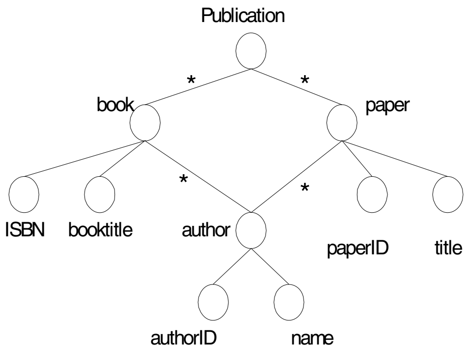
3 X2R: A New Method for Mapping XML to Relations

In this study, a transformation language that is able to extract the semantics information in XML and preserve it during the transformation is developed. However, we have to deal with DTDs that may contain arbitrary regular expressions, and also we have to deals with null values and incomplete relations. Functional dependency is used to define that X determine Y or X->Y. However functional dependency for XML (XFD) is more complicated since we need to deal with the hierarchical structure of XML and the path expression that can be used to express XML. Till now there is no standard definition for XFD, therefore a lot of attempts done to define ones. Studies had shown that different definition of XFD will have different expressive power [1]. The XFD that we adopt is an expression of the form: (C, Q : X -> Y), where C is the downward context path which is defined by an XPath expression from the root of the XML document, Q is a target path, X is an LHS (Left-Hand-Side) and Y is an RHS (Right-Hand-Side). Functional dependencies are used to specify constraints in the XML and use this constraint to infer a non-redundant relational schema for XML. The strategy adopted in this study, is to produce a relational design, which preserves structural and semantic constraints of the XML data while reduced redundancies. First, the structural of XML data is captured by the DTD and generate the DTD schema, which is the formal description of XML. Using the constraint-preserving algorithm, the redundant path is removed. Finally, by mapping paths in XFDs to relational attributes, a set of relational functional dependencies and a relational storage for the XML data is produced, which preserves the content and the structure information of the original XML document. The generated relational schema is able to remove redundancy as indicated by the XFDs, and enforced efficiently using relational primary key constraints.

4 Motivating Example

Publication dataset [16] as in Fig. 1 is used to illustrate the effectiveness of the proposed method. This dataset describes the publication that has many books and papers. Each books and papers contains information about the authors who published either books or papers. To evaluate the effectiveness of the proposed method (X2R), an experiment is conducted where the relational schema generated by X2R, RRXS, Lv&Yan methods is compared. RRXS and Lv&Yan methods which are under semantic-based approach are used in the comparison because they consider XFD in the mapping. Given the DTD graph, its schema and corresponding XFD that may exist in the Publication dataset, the relational schema generated by the three methods shown as in Fig.2.

From the generated schema, X2R method generates a good relational schema for Publication dataset when compares with the schema generates by RRXS and Lv&Yan method. The relational schema generates by RRXS in Fig. 2(ii) will produce two equivalent tables that belong to the same concept which are Author and Author1 table. The reason for this redundant creation is that the algorithm did not check the existence of already created table for the same concept of object (Author),



<!ELEMENT publication(book*, paper*)>
<!ELEMENT book (ISBN, booktitle, author*)>
<!ELEMENT paper(paperID, title, author*)>
<!ELEMENT author (name)>
<!ATTLIST author authorID

XFD1 //book: ISBN -> ε
XFD2 //paper: paperID -> ε
XFD3 //author: authorID -> name

Fig. 1. DTD graph, its schema and corresponding XFD

Book (ISBN, booktitle)
Author (authorID, name)
Paper(paperID, title)
Book-author(ISBN, aID)
Paper-author(paperID, authorID)

i.Schema generated by X2R

Book (ISBN, booktitle)
Author (authorID, name)
Author1(ID, authorID, name)
Paper (paperID, title)
Book-author(ISBN, aID)
paper-author1(ID, paperID, author1.ID)

ii.Schema generated by RRXS

Publication(ID)
Book(ID, ISBN, booktitle, author.ID)
Paper(ID, paperID, title, author.ID)
Author(ID, authorID, name)
Publication-book (publication.ID, book.ID)
Publication-paper(publication.ID, paper.ID)
Book-author(book.ID, author.ID)
Paper-author(paper.ID, author.ID)

F_1(publication.book.ISBN, publication.book.ID)
F_2(publication.paper.paperID, publication.paper.ID)
F_3(publication.paper.author.authorID, publication.paper.author.name)

iii.Schema generated by Lv&Yan

Fig. 2. Relational Schema generated by X2R, RRXS and Lv&Yan

it blindly created a new one. These redundant tables may lead to the update anomaly problems. The method by Lv&Yan will create two types of tables: based on structural DTD and based on semantics presented by XFD. Basically three steps involved in this algorithm: i) using structural DTD, a separate relation will be created for each non-leaf vertex and leaf, ii) for each parent-child relation between two vertexes connected by a * operator, a separate relation is created, and ii) for each XFD defined over DTD, a separate relation is created. Based on this algorithm the resulted relational schema is shown as in Fig. 2(iii). As observed, the F_3 table and the Author table are redundant. These tables are used to describe the same concept of object (Author), therefore this will lead to update anomaly problems. For instance if a new author is added to the publication, both the relations F_3 and author need to be updated for the database to satisfy the constraints. The relational schema generated by X2R overcomes the limitation of these methods by producing a good relational schema design for XML with no redundant relations. In fact the schema produced is correct with respect to keys and functional dependencies.

5 Conclusion

We have investigated the problem of how to design a good relational schema for XML data with no redundant relation. A new method has been developed which given functional dependencies and DTD, redundancy in XML document can be detected and used this information for mapping to relations which can reduce relation redundancy and at the same time preserve the constraints as expressed in functional dependencies. This method can be efficiently operated, automated and eliminates unnecessary ID. As an immediate task, we would like to find efficient algorithm for mapping from relations to XML that based on functional dependencies which may appear in XML. Through this study it is hope that it will give contributions to the database community.

References

1. Ahmad, K., Mamat, A., Ibrahim, H., Mohd Noah, S.A.: Defining functional dependency for XML. Journal for Information Systems Research and Practices (2008)
2. Amer-Yahia, S.A., Du, F., Freire, J.: A Comprehensive Solution to the XML to Relational Mapping Problem. In: Proceedings of the 6th Annual ACM International Workshop on Web Information and Data Management, pp. 31–38 (2004)
3. Atay, M., Chebotko, A., Liu, D., Lu, S., Fotouhi, F.: Efficient Schema-based XML-to-Relational Data Mapping. Journal of Information System 32, 458–476 (2007)
4. Bohannon, P., Freire, J., Roy, P., Simeon, J.: From XML Schema to Relations: A Cost-Based Approach to XML Storage. In: Proceedings of the 18th International Conference on Data Engineering, pp. 64–74 (2002)
5. Chen, Y., Davidson, S., Hara, C., Zheng, Y.: RRXS: Redundancy Reducing XML Storage in Relations. In: Proceedings of 29th International Conference on Very Large Data Base, pp. 189–200 (2003)
6. Chen, Y., Davidson, S., Zheng, Y.: Constraint Preserving XML Storage in Relations. In: Proceeding of the 9th International Conference of Database Theory, pp. 7–12 (2002)
7. Davidson, S., Fan, W., Hara, C., Qin, J.: Propagating XML Constraints to Relations. In: Proceedings of the 19th International Conference on Data Engineering, pp. 543–554 (2003)
8. Fan, W.: XML Constraints: Specification, Analysis, and Application. In: Proceedings of the 16th International Workshop on Database and Expert Systems Applications, pp. 805–809 (2005)
9. Florescu, D., Kossman, D.: A Performance Evaluation of Alternative Mapping Schemes for Storing XML Data in A Relational Database. In: Proceedings of the VLDB (1999)
10. Lee, D., Chu, W.W.: CPI- Constraint-Preserving Inlining Algorithms For Mapping XML DTD to Relational Schema. Journal of Data Knowledge and Engeering 39(1), 3–25 (2001)

11. Lee, Q., Bressan, S.,Rahayu, W.: XShreX:Maintaining Integrity Constraints in the Mapping of XML Schema to Relational. Proceedings of the 17th International Conference on Database and Expert Systems Application, pp.492-496 (2006)
12. Liu, C., Vincent, M., Liu, J.: Constraint Preserving Transformation from Relational Schema to XML Schema. Journal of World Wide Web 9(1), 93–110 (2006)
13. Lv, T., Yan, P.: Mapping DTDs to Relational Schemas with Semantic Constraints. Journal of Information and Software Technology 48(4), 245–252 (2006)
14. Qin, J., Zhao, S., Yang, S., Dou, W.: XPEV: A Storage Model for Well-Formed XML Documents. In: Wang, L., Jin, Y. (eds.) FSKD 2005. LNCS (LNAI), vol. 3613, pp. 360–369. Springer, Heidelberg (2005)
15. Schewe, K.: Redundancy, Dependencies and Normal Forms for XML Databases. In: Sixteenth Australasian Database Conference, vol. 39 (2005)
16. Shanmugasundaram, J.: Relational Databases for Querying XML Documents: Limitations and Opportunities. In: Proceedings of the 25th VLDB Conference, pp. 302–314 (1999)
17. Tian, F., DeWitt, J., Chen, J., Zhang, C.: The Design and Performance Evaluation of Alternative XML Storage Strategies. SIGMOD Record 31(1), 5–10 (2002)
18. Yoshikawa, M., Amagasa, T., Shimura, T.: XRel: A Path-based Approach To Storage and Retrieval of XML Documents Using Relational Database. ACM Transactions on Internet Technology 1, 110–141 (2001)
19. Zhanga, S., Gana, J., Xua, J., Lva, G.: Study On Native XML Database Based GML Storage Model. In: The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Beijing, vol. XXXVII (2008)