INTEGRATING RELATED XML DATA INTO MULTIPLE DATA WAREHOUSE SCHEMAS

Soumya Sen[1], Ranak Ghosh[2], Debanjali Paul[3], Nabendu Chaki[4]

[1,4]University of Calcutta Kolkata -700009 West Bengal, India
[1]iamsoumyasen@gmail.com
[4]nabendu@ieee.org
[2,3]Barrackpore Rastraguru Surendranath College, Kolkata - 700120 West Bengal, India
[2]ranakghoshmail@gmail.com
[3]debanjali.rimi@gmail.com

ABSTRACT

XML is a worldwide standard to represent data in web based system. Numbers of organizations use XML for e-commerce and internet based applications. On the other hand, data warehouse is used by most of the organizations for making decisions on their business by analyzing operational data. Integration of business logic based on data warehouse and XML based system has therefore emerged as a demanding area of research interest. This paper illustrates an approach to integrate XML data based on multiple related XML Schemas, to its compatible data warehouse schemas based on relational online analytical processing (ROLAP). The novelty of the work is that more than one type of data warehouse schemas could be identified from a single related XML schema. A new data structure, Schema Graph has been proposed in the process.

KEYWORDS

XML, XML Schema, Data Warehouse, Fact Constellation Schema, Star Schema, Snowflake Schema, Schema Graph

1. INTRODUCTION

Data warehouse [9] is subject-oriented, non-volatile, integrated, time variant collection of data which helps in developing strategic decisions. The popular way of describing data warehouse is through multi-dimensional model [9]. Multidimensional model is based on some central theme which is represented by fact table [9]. A fact table is constructed from multiple dimension tables [9]. The association between these fact table and dimension tables are generally represented through three data warehouse schemas namely star schema, snowflake schema and fact constellation.

XML is well established standard for semi-structured data and also poses several benefits in web environment. Many organizations thus prefer the use of XML extensively. Data could come from different heterogeneous sources [5], some of them be may be semi-structured. In order to integrate these heterogeneous data they could be converted to XML to take the advantages of XML standard. XML data is associated with DTD [10] or XML schema [10]. XML provides

Document Type Definition (DTD), which explains precisely what elements could appear as document and what the contents of the elements and attributes are. Some of the approaches show how XML data based on DTD [6] [7] [8] have been converted to data warehouse schema. However DTD have some limitations. DTD do not have any built-in data types; also do not support user-derived data types and allow only limited control over cardinality. Whereas XML schemas are more powerful to represent XML document structure and overcome the limitations of XML DTD. An XML schema could works as a fact repository [12]. The increasing use of Web and the requirement of integrating data from different data sources expedite the research in converting XML schema to equivalent data warehouse system. Retrieving multi-dimensional data model [11] from XML sources became important over the years. XML document could be summarized [13] and represented in the form of data warehouse. Moreover due to the transmission of data over public domain like web, secure XML data warehouses [14] are need to be formed. Several researches have been done for integrating XML data in data warehouse [1] [2] [3] [4].

The paper [1] converts XML schema either to star schema or snowflake schema. Moreover this paper works only with single XML schema. The paper [2] proposes a method to design multiple cubes of multidimensional model from XML schema. The paper [3] proposes a method to convert the contents of the XML schema to multiple schemas of the multi dimensional model. However all these generated schemas are converted to star schema only. The paper [4] proposes XML schema conversion to OLAP cube. This paper focuses on converting XML schema to corresponding data warehouse schema by identifying fact and dimension tables.

However, the existing methods studied and cited here are only capable to identify a single data warehouse schema from a given XML schema. In this paper, a framework is proposed to detect more than one data warehouse schema from the given XML schema and support the identification of all three types of data warehouse schemas.

The process of obtaining the data warehouse schema from the XML schema has been proposed here. At first XML schema is converted to a schema graph (described in Section 3.2). Schema graph is a new data structure proposed here for the conversion process. In next stage the fact and dimension tables are being identified from the schema graph (described in Section 3.3). Once these tables have been identified, the measure of fact table is taken from the user. Depending on the relationship between dimension tables and fact tables one or more star schemas, snowflake schemas or fact constellation schema are being identified (described in Section 3.4).

2. PROPOSED DATA STRUCTURE

Schema Graph: A schema graph is a representation of entities found in the XML Schema. The graph has the following properties:

a. It is a level wise separable graph.

b. The entities encountered in the XML are represented through vertices of the graph.

c. The name of the vertices of the Graph would be same as the name of the entities.

Holder Element (HE): The elements that have no predecessor are in the Schema Graph are called Holder Elements. They are placed in Level-1 of the graph

Contained Element (CE): The elements that are directly connected to the HE’s are called Contained Element. They are placed in Level-2 of the graph.

Secondary Elements (SE): The elements that are directly connected to the CE's are called Secondary Elements. They are placed in Level-3 of the graph.

If elements in the graph appear as connected to SE, they would be placed in level-4. The new vertices that would be connected to the vertices of level-4 would be placed in level-5. Subsequently new level could be created if the new entities appear in the graph connected to the previous level.

All the entities are represented through rectangular vertex in Schema Graph. The attributes are represented in the Schema Graph in an Oval shaped vertex. They are connected to the corresponding entities. In Figure 1 entities are shown only.

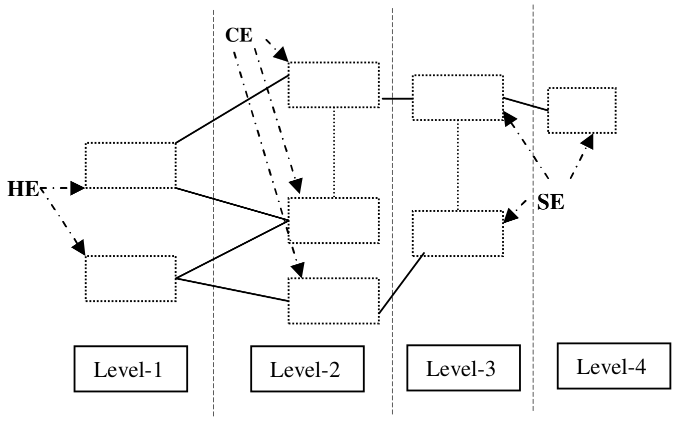

Figure 1. Schema Graph Along with HE, CE and SE

3. **CONVERSION OF RELATED XML SCHEMA TO WAREHOUSE SCHEMA**

3.1. Overview of The Proposed Framework

The modeling of Warehouse Schema from a related XML Schema is performed in two steps. At first Schema graph is formed from the XML schema (described in section 3.2). The elements of the schema graph are identified as HE, CE and SE. In the next stage fact tables and dimension tables are identified. Moreover if any element doesn't have any primary key then a key attribute is added to it (described in section 3.3). Each HE would correspond to a fact table and would make an entry in the fact table, the key attribute of the CE's that are connected to the HE are entered in the corresponding fact table for that HE. CE's would be the dimension tables. If SE's are found connected with CE the primary keys of SE's are placed in CE. If SE's are present even after level-3, the primary keys of the higher level are placed in the table corresponding to the SE of immediate lower level. Next the warehouse schema structure is identified (section 3.4). There are three separate methodologies for three data warehouses schemas. Identification of star schema, snowflake schema and fact constellation are described in section 3.4.1, 3.4.2 & 3.4.3 respectively.

3.2. Procedure to Build Schema Graph From Related XML Schema

In order to build the schema graph from the XML schema at first the entities are identified. In XML schema those elements which are not nested within some elements are termed as HE and are placed at level-1 of the schema graph. After that if some element is being found to be declared within the body of HE they are being classified as CE and placed at level-2 of the schema graph. Again if new elements are declared within body of CE they are being placed at level-3 of the schema graph and are termed as SE. If the nested elements are further found for SE they are placed in the next level of the schema graph and are also termed as SE. This scanning would be continued till all the elements which are being nested are identified. Though new levels are created in the schema graph based on the higher level of nesting all of them are termed as SE, that is, elements in level-3 or more than level-3, all are being termed as SE. This algorithm is shown in Figure 2.

```
a) Find out those entities in XML schema that have no predecessor and
denote them as the starting vertices or holders for the entire graph. These
entities would be known as HE. They would placed in the Level-1 of the
graph.
b) For all  HE (i=1 to n) perform following : (n is the total number of
HE)
   i. Find the sequence of elements under i-th HE :
      If it is an element then create a vertex for it into the graph and
      connect it with i-th HE. These elements (vertices) would be denoted
      as CE. CE would be placed in the level-2 of the graph.
      Else if it is an attribute it would be considered as an attribute of the
      corresponding HE.
  ii. For all  CE (j=1 to m) perform following: (m is the number of CE in
       the HE)
 iii. Scan the XML Schema for j-th CE:
      If it is an element then place it into the graph and connect it  with its
      CE. These elements would be known as SE. SE would be placed in
      the level-3 of the graph.
      Else if it is an attribute place it would be considered as an attribute of
      CE.
  iv. For every SE (k=1 to p) : (p is the total number of SE at that  level)
      Repeat the steps to include the entities and attributes as they
      encountered. Whenever a new entity is added new level is created for
      it.
  End For /* SE */
 End For /* CE */
End For /* HE */
```

Figure 2. Algorithm to Construct Schema Graph from XML Schema

3.3. Identifying Fact and Dimension Tables

While drawing the Schema Graph three entities have been specified: HE, CE, SE. The CE's are chosen as the dimension table and each HE would prompt towards a fact-table. If there exists any SE, then the primary keys of them would be included in the corresponding CE's. Similarly, if there are entities beyond level-3, then the primary keys of entities of level (i+1) appear as foreign key in the connected entity of level i. If any entity is found without having a primary key, then primary key is added to it as: Name of entity + "_id"

3.4. Identification of the Schema Structure of Data Warehouse

Once the fact and dimension tables are identified, the corresponding DW schema need to be build. Here three popular data warehouse schemas are identified namely star schema, snowflake schema and fact constellation schema. This is done by checking the nature of connection among elements within the schema graph. If the schema is found as disconnected, more than one schema is identified. The numbers of distinct components in the schema is same as the number of DW schema identified.

3.4.1 Procedure Star Schema

A data warehouse schema is identified as star schema if the schema graph consists of HE and CE's only. Every fact table is named as the name of HE + "Fact". The primary keys of each of the connected CE are placed in fact table and the CE's are act as dimension tables in the star schema. This algorithm is shown in Figure 3.

```
Partition the Schema Graph Level wise.
Identify HE
For HE:
   a. Form a Fact-Table with the name of HE + "Fact" and Primary key of
      the HE
   b. Specify the CE connected with this HE; include the primary  keys of
       each CE  into the Fact-Table.
End For /*HE*/
```

Figure 3. Algorithm to Construct Star Schema

3.4.2 Procedure Snowflake Schema

A data warehouse schema is identified as snowflake schema if the schema graph consists of HE, CE and SE provided that HE's are not connected. At least one SE should exist for some CE in the schema graph to be identified as snowflake schema. Every fact table is named as the name of HE + "Fact". The primary key of the connected CE are placed in fact table and the CEs act as dimension tables. Similarly the dimension tables would contain the primary key of the tables represented by SE. If further levels of SE are found they are represented in the schema by placing the primary key in the immediate previous level of SE tables. The algorithm is shown in Figure 4.

```
Partition the Schema Graph Level wise.
Identify the HE
For each HE:
   a. Form a Fact-Table with the name of HE + "Fact" and primary key of the
HE
   b. Specify the CE connected with this HE, include the primary keys of each
CE into the Fact-Table.
   c. For each CE find SEs, if any.
        If found Connect it with its CE using the primary key of the SE.
   d.  For each SE:
    Check if there are any SE:
      If further level of SE is found the primary key of the new SE of
      the immediate higher level is placed in the SE of current level;
   End For /* SE */
 End For /* CE */
End For /* HE */
```

Figure 4. Algorithm to Construct Snowflake Schema

3.4.3 Procedure Fact-Constellation

A data warehouse schema is identified as fact constellation if the fact tables are found connected. The number of HE's in a component of the schema is same as the number of fact tables for the component. Every fact table is named as the name of HE + "Fact". The primary keys of the connected CEs are placed in fact table. If further levels of SEs are found, these are represented by placing their primary keys in the immediate previous level of SEs. This algorithm is shown in Figure 5.

```
Partition the Schema graph level wise;
Identify the HE
For each HE:
   a. Form a Fact-Table with the HE +"Fact" and Primary key of the HE
   b. Specify the CE connected with this HE, include the primary keys of
      each CE into the Fact-Table.
   c. For each CE check if there are any SE:
      If Found
     Connect SE with its CE including the primary key of the SE in the
     CE;
   d. For each SE:
      Check if there are any SE:
       If further level of SE is found the primary key of the new SE of
       the immediate higher level is placed in the SE of current level
      End For /* SE */
    End For /* CE */
   End For /* HE */
```

Figure 5. Algorithm to Construct Fact Constellation

4. EXAMPLE

In this section an illustrative example is presented to explain the conversion methodology of an XML Schema to its equivalent data warehouse schema

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http??www.w3.org/2001/xmlschema">
<xsd:elementname="sales">
<xsd:complexType>
<xsd:sequence>
<xsd:elementname="Item" type="ItemType">
<xsd:sequence>
<xsd:elementname="name" type="xsd:string" use="required"/>
<xsd:elementname="Title" type="xsd:string" use="required"/>
</xsd:sequence>
<xsd:elementname="branch" type="BranchType">
<xsd:sequence>
<xsd:elementname="type" type="xsd:string" use="required"/>
<xsd:elementname="Branch_id" type="xsd:string" use="required"/>
</xsd:sequence>
<xsd:elementname="Location" type="LocationType"/>
<xsd:elementname="Sales_id" type="xsd:string" use="required"/>
<xsd:attributename="unit" type="xsd:positiveinteger"
use="required"/>
</xsd:sequence>
</xsd:complextype>
</xsd:element>
<xsd:elementname="shipping">
<xsd:complexType>
<xsd:sequence>
<xsd:elementname="shipper" type="ShipperType">
<xsd:sequence>
```

```
<xsd:elementname="name" type="xsd:string"
  use="required"/>
<xsd:elementname="Shipper_id" type=locationType
   use="required"/>
</xsd:sequence>
<xsd:elementname="Location" type="LocationType"/>
<xsd:attributename="Shipping_id" type=ShipperType
   use="required"/>
<xsd:attributename="location_to" type=LocationType
  use="required"/>
<xsd:attributename="unit" type="xsd:string"
   use="required"/>
</xsd:sequence>
</xsd:complextype>
</xsd:element>
<xsd:complexTypename="LocationType">
<xsd:sequence>
<xsd:elementname="street" type="xsd:string"
   use="required"/>
<xsd:elementname="city" type="xsd:string"
  use="required"/>
</xsd:sequence>
```

Figure 6. Example of XML Schema

In the XML schema (Figure 6) by applying the proposed methodology described in section 3.2 schema graph is drawn. It is found that the elements Sales and Shipping have no predecessors. Thus they would be identified as HE. For the HE Sales three CE's are identified namely they are Item, Branch and Location. All these CE's have no SE. For the HE Shipping two CE's are identified. They are Shipper and Location. Both of these CE don't have any SE. The attributes of every entity is connected with the corresponding entity. The schema graph obtained is shown in Figure 7.

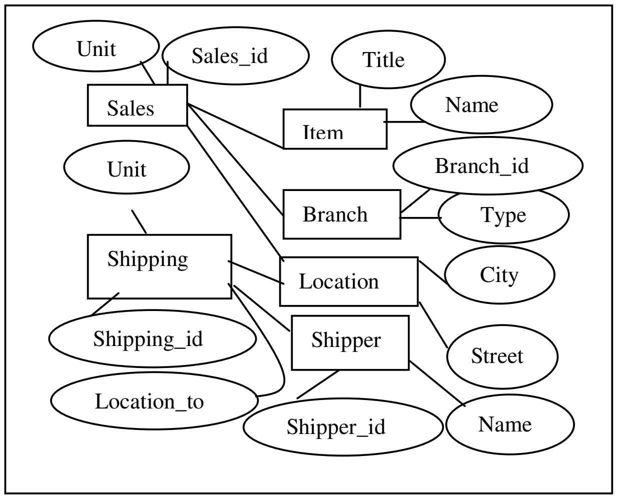

Figure 7. Schema Graph of XML Schema

Now fact and dimension tables are identified using the procedure described in section 3.3. As the schema graph contains two HE, two fact tables would be constructed namely Sales_Fact and Shipping_Fact. The dimension tables for Sales_fact would be Sales, Item, Branch, Location. The dimension tables for Shipping_Fact would be Shipping, Shipper, Location. Now primary keys are required to be identified.

We assume that the entity Sales, Shipping, Branch, Shipper have the primary key hence we need to add primary key for the entity Item and Location. The primary keys of the entity Item and Location are named as Item_id and Location_id (described in section 3.3). These new primary key attributes are shown in Figure 8.

The next step is to identify the proper data warehouse schema from the schema graph. As the HE Sales and Shipping are connected through the CE Location, procedure Fact-Constellation (described in section 3.4.3) is applied to build the data warehouse schema. In both of the fact tables there is an attribute user_measures which would be given by the users. This is shown in Figure 8.

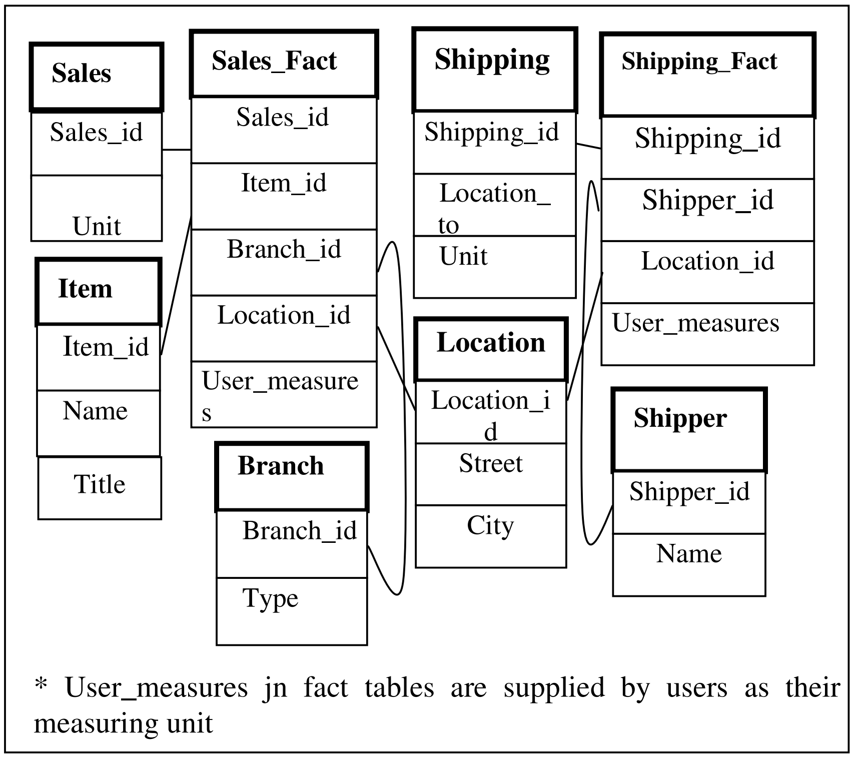

Figure 8. Fact Constellaion Schema from the Schema Graph of Figure-7

5. CONCLUSION

This paper provides a framework to convert a related XML schema to data warehouse schema which includes star schema, snowflake schema and also the fact constellation. Moreover it is capable to identify multiple schemas from a XML schema if they are related. This is an interesting and emerging area of research as a number of business organizations use data warehouse for their business analysis and use XML to handle semi-structured data and also to take the advantage of using web environment. Converting different types of semi-structured data to XML is another important research interest. This work could be extended further to construct data warehouse from different semi-structured data by adding an intermediate step to transform semi-structured data to XML. Thus the organizations looking to incorporate business intelligence can use this type of framework to build data warehouse architecture from heterogeneous data sources.

REFERENCES

[1] Sarbani Dasgupta, Soumya Sen, Nabendu Chaki; "A Framework To Convert XML Schema to ROLAP"; Proc. of the Second International Conference on Emerging Applications of Information Technology (EAIT 2011), Kolkata, India, 2011 ISBN : 978-1-4244-9683-9

[2] From XML schema to cube Parimala N and Payel pahwa International Journal of Computer Theory and Engineering Vol 1 No 3 August 2009.

[3] Conceptual design of data warehouses from xml schemas Payel pahwa and Parimala N 2nd International Conference on Intellectual Capital, knowledge management & Organizational Learning 21-22 Nov,2005 American University of Dubai, United Arab Emirates

[4] M. Jensen, T. Møller, and T.B. Pedersen, .Specifying OLAP Cubes On XML Data., Journal of Intelligent Information Systems, 2001.

[5] Frank S.C. Tseng, Chia Wei Chen: Integrating heterogeneous data warehouses using XML technologies, Journal of Information Science Volume-31, Issue:3 (June 2005) Page-209-229

[6] Boris Vrdoljak, Marko Banek, and Stefano Rizzi: Designing Web Warehouses from XMLSchemasY. Kambayashi, M. Mohania, W. Wöß (Eds.): DaWaK 2003, LNCS 2737, pp. 89-98, 2003. Springer-Verlag Berlin Heidelberg 2003

[7] Wolfgang Hummer, Andreas Bauer, Gunnar Harde: XCube – XML For Data Warehouses, DOLAP'03, November 7, 2003, USA.

[8] M. Golfarelli, S. Rizzi, and B. Vrdoljak, .Data warehouse design from XML sources., Proc. DOLAP'01, Atlanta, pp. 40-47, 2001.

[9] Data Mining Concepts and Technique,2nd Edition, Jiawei Han and Micheline Kamber, Morgan Kaufmann Publisher

[10] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, François Yergeau; "Extensible Markup Language (XML) 1.0 (Fifth Edition)"; W3C Recommendation 26 November 2008 www.w3.org/TR/REC-xml

[11] Yuan Sun; Hexin Chen; Mianshu Chen; Xinying Wang; Aijun Sang; "Multi-dimension Multimedia Retrieval Model Implementation Based on XML Database" International Conference on Signal Processing Syatems, 2009

[12] Rajugan, R.; Chang, E.; Dillon, T.S.; "Conceptual Design of an XML FACT Repository for Dispersed XML Document Warehouses and XML Marts", 5th International Conference on Computer and Information Technology, 2005

[13] Ramanath, M.; Kumar, K.S.; "A rank-rewrite framework for summarizing XML documents" 24th International Conference on Data Engineering Workshop, ICDEW 2008

[14] Belen Vela; Carlos Blanco; Eduardo Fernandez; E.Marcos "Model Driven Development of Secure XML Data Warehouses: A Case Study". EDBT 2010, Lausanne, Switzerland.

Authors

Soumya Sen is a faculty member in A. K. Choudhury School of Information Technology under University of Calcutta since March, 2009. He obtained his M. Tech. Degree in Computer science & Engineering from the University of Calcutta in 2007. His area of research is Data Warehouse and OLAP Tool. He is currently pursuing his PhD work as a part-time scholar.

Ranak Ghosh completed his M.Sc. degree in Computer Science in 2011 from Barrackpore Rastraguru Surendranath College under West Bengal State University. This work is part of his Masters dissertation work under the guidance of Soumya Sen

Debanjali Paul completed her M.Sc. degree in Computer Science in 2011 from Barrackpore Rastraguru Surendranath College under West Bengal State University. This work is part of her Masters dissertation work under the guidance of Soumya Sen

Nabendu Chaki is an Associate Professor in the Department Computer Science & Engineering, University of Calcutta, India. Besides editing several volumes in Springer proceedings, Nabendu has authored 2 text books and close to 100 refereed research papers in Journals and International conferences. His areas of research interests include distributed systems and software engineering.