XML DATA HANDELING, STORAGE, INDEXING AND QUERY LANGUAGES

Megha Soni
Dept. of CS, Poornima Institute of Engineering & Technology
Jaipur, India

***Abstract*— As new standards for technology specifications, XML has emerged as a building block for information representation and data exchange. The process of storing, indexing, querying and deleting the XML documents have become the major issue in data world along with the data ambiguity and uncertainty.**

The basic intention of this paper is to describe some of the main problems related with the XML documentation and XML storage and querying. For resolving the issues, the adopted methodology is titled as fuzzy data management, which has been incorporated in database management systems in different ways. This paper presents a method of storing and querying xml data in relational database. XPath is used to get the xml documents, views xml documents as a tree and the information stored in the node and structure of the tree is stored in one table. LOAD XML statement is used to convert xml tree representation into table. SQL is used for query purpose.

***Keywords*- XPath, Fuzzy, LOAD, RDBMs, XML, SQL.**

I. INTRODUCTION

XML stands for Extensible Markup Language. It was designed in order to overcome the limitations of HTML. Both XML and HTML are derived from SGML which stands for Standard Generalized Markup Language, mother of all markup language. A markup language uses tags to define elements within a document. Its markup files contain standard words instead of programming syntax and that's why it is human readable. It works both as markup language and data format. It is important to understand that that XML is not a substitute for HTML. HTML focuses on design of the webpages and how data will be displayed on the web pages. XML focuses on how data will be stored and transmitted and what data is. XML was designed for well-formed data. HTML has predefined tags whereas XML provides flexibility to the users to define their own tags. XML tags are extensible as compared to HTML tags which are limited. XML tags are case sensitive but HTML tags are not.

HTML tags are used for displaying data whereas XML tags are used for describing data. HTML doesn't follow any strict rules

and some of the mistakes cab be ignored, but XML is strictly rule based. Purpose of XML is to exchange data between applications in vendor, language and platform independent manner over web. Its documents are used as textual database. XHTML, WSDL, WAP, SMIL etc are the intent languages which are based on XML. Any database data can be easily transformed into XML format. Data can be inserted or updated into the database tables. UTF-8 is used to encode content in XML files. It is very easy to describe data in table structure format and hence XML is self-describing data. Its tree like structure is easy to understand and is human readable. A database is an organized collection of data in which we can apply insertion, deletion, updation and many more. Database Management System is the software which is used to manage database. Data can be represented in well-understood manner using many traditional databases. But real world problem may contain some complex data and their representation is still very much of research issue. A reliable system is needed to store and allow efficient access to these data. The use of relation database for such purpose has grown considerable interest and the XML data in relational database looks promising. Though XML has been used for data exchange in many domains but how to manage XML data efficiently has become a prime hotspot for researchers.

XQuery and XPath are most important XML query language. XQuery is a functional programming and query language which is used to query a group of XML data. It is used to extract and manipulate data from either xml documents or relational database and ms office documents. A tree model with seven nodes is used to represent it. Seven nodes are processing instructions, elements, document nodes, attributes, namespaces, text nodes and comments. It is used to create syntax for new xml documents. The type and attribute of each tag usable for certain

XML documents can be well defined using XML schema or DTD.

XPath is a XML path language that is used to select nodes from an XML documents using query. Numbers, strings and Boolean types from another xml documents are computed using XPath. It is represented as tree and is used to navigate it using different nodes.

Fuzzy Markup Language (FML) is a specific purpose markup language which is based on XML. The structure and behavior of a fuzzy system independent of the hardware architecture is described using FML. The major advantage of using XML to describe a fuzzy system is hardware and software interoperability. All that is needed to read an FML file is the appropriate schema for that file and an FML parser. This Approach makes it a lot easier to exchange fuzzy system between software.

II. PROBLEM DEFINITION

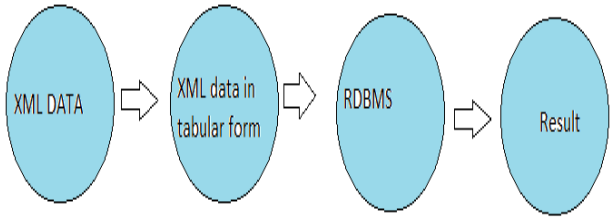
Fuzzy Database Management system has been a hotspot for research in various database management systems as imprecision and uncertainty exists in many real world problems. A lot of interest has been created by introducing native support for XML data in relational database systems (RDBMS). It has created a view of leveraging the powerful and reliable data management services provided by RDBMS. There are a lot of works available on xml-to-relational storage but the problem of storing fuzzy XML data in RDBMS is not explained satisfactorily anywhere. This paper is going to introduce a methodology to store and query fuzzy xml data in relational database. Since xml data can leverage powerful and reliable data management service, storing and

querying of xml data using relational database system has attracted considerable interest.

As the relational model and XML model are not identical, it is important to first shred and then load the xml data into relational table. After this XML query should be translated over the original data into equivalent SQL queries over the table. All storage problems has not been mentioned in a single work although a lot of work is available on XML-relational storage. There is often a little or no information available about query translation in the works done on mapping strategies. Usually they are tied to a specific backend and use proprietary mapping language, which requires steep curves as well as they are often unable to represent desirable mapping which is required.

III. PROPOSED METHODOLOGY

In this paper we are going to review a methodology which can be used to store and query XML data into relational database. We can represent the methodology into an architectural form as follows:



A. *XML Query Language*

XML (eXtensible Markup Language) is used to describe data. It is a flexible way to create information format. It contains markup format to describe page and file contents. In XML structure of data is embedded with data and that's why XML is called self-describing. When data arrives there is no need to pre-built structure to store data. An element is the basic building block of XML which is defined by tags. It has a beginning and an ending tag. XML focuses on how data will be stored and transmitted and what data is. Content of the element is described by the element name. The structure of the element describes the relation between elements. XML power resides in its simplicity that it can convert large chunk of data into XML documents, a meaningful piece that can provide structure and organization in data.

XQL stands for XML Query Language. It is a query language for XML in the same way as SQL is query language for relational tables. It is a way to locate and filter elements and text in XML. XML files are used to transmit collection of data between computers on web. XQL provides a tool for finding out specific items in XML files. It is based on XSL and hence it is closely related to XPath. An important reason behind the design of XQL is that XML has its own implied data models a feature which is neither found in the relational database or object oriented database or object relational database. Documents in XML are represented in the form of structured and labeled tree with nodes to represent entity elements, information comments etc. In XQL, nodes from one or more documents are returned as a result of query.

XPath is used to navigate through elements and attributes in XML path. It is a W3 recommendation. We can select nodes or sub-nodes with the help of XPath.

B. *Structured Query Language*

Relational database is a database which is consisting of multiple data sets which are organized in the form of tables i.e. rows and column. It is used to access data in relation to another data in database. Relational database management system is used to manage relational database. It uses SQL (Structured Query Language) for querying data. It uses four forms of functional dependencies which are as follows:

- **One-to-one** in which a record in table maps to another record in another table
- **One-to-many** in which a record in table maps to many record in another table
- **Many-to-one** in which many records in a table maps to another record in another table
- **Many-to-many** in which many record in a table maps to many record in another table

SQL is used to select, insert, update and modify in database.

SQL are divided into four main categories:

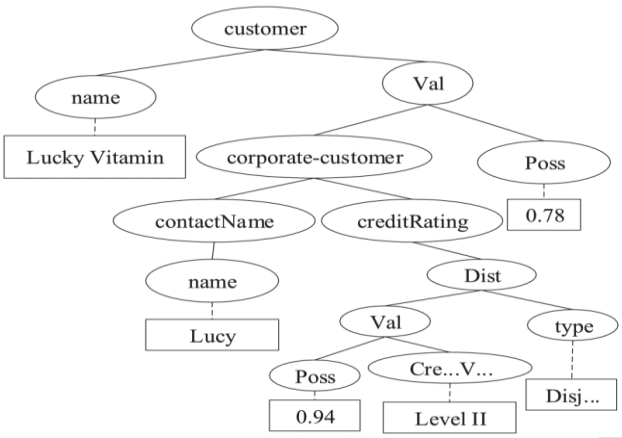
- **DDL** is Data Definition Language which is used to define structure of database. Some of the commands are Create, Alter, Drop, Rename, Truncate, and Comment
- **DML** is Data Manipulation Language which is used to manipulate data. Some of the commands are Select, Insert, update and Delete.
- **DCL** is Data Control Language which is used to control access to data in database. Some of the commands are grant and revoke.
- **TCL** is transactional Control Language which is used to manage transaction in database. Some of the commands are commit, rollback, savepoint and set transaction.

C. *Fuzzy XML Data Model*

A fuzzy xml is intended as a syntax tree which is consists of elements and their attributes. We use several fuzzy data construct for fuzzy data model. It uses a possibility attribute "Pass" along with "Val" which specifies the possibility of a given element in given XML document. Possibility distribution of an element is given by pair <Val Poss> and </Val>.

```
<customer>
 <name> Lucky Vitamin</name>
 <Val Poss=0.78>
  <corporate-customer>
   <contactName>Lucy</ contactName >
    < creditRating >
     <Dist type="disjunctive">
      <Val Poss=0.94>
       <creditRating_value >Level II </creditRating_value >
       </Val>
       </Dist >
     </ creditRating >
  </ corporate-customer >
</customer>
```

A fragment of fuzzy of xml document



Tree representation of above fuzzy data

D. *XML to Relational Storage*

We can store XML data into relational database with the help of XPath mapping technique. It is used to locate and process items in XML. It allows programmers to have a higher level of abstraction. XPath uses a syntax which is somewhat like informally finding geographical direction. It specifies route rather than pointing to specific set of words. XML data tree have three main kind of nodes namely element, attribute and text nodes. Leaf nodes and non-leaf nodes are the two sub-categories of element nodes. There are three special nodes which are possibility attribute, Val construct and Dist Construct.

E. *LOAD XML and the concept*

The LOAD XML statement reads data from XML file into table.

```
LOAD XML
    [LOW_PRIORITY | CONCURRENT] [LOCAL]
    INFILE 'file_name'
    [REPLACE | IGNORE]
    INTO TABLE [db_name.]tbl_name
    [CHARACTER SET charset_name]
    [ROWS IDENTIFIED BY '<tagname>']
    [IGNORE number {LINES | ROWS}]
    [(field_name_or_user_var
        [, field_name_or_user_var] ...)]
    [SET col_name={expr | DEFAULT},
        [, col_name={expr | DEFAULT}] ...]
```

The 'filename' and 'tagname' should be string. The LOAD XML acts as the compliment of running MySQL client in XML output mode to write data from a table to an xml file we can use *–xml* and *–e* option from system shell.

```
shell> mysql --xml -e 'SELECT * FROM mydb.mytable' > file.xml
```

The XML concept is discussed below:

Let us suppose we have a table named "person" which is initially empty as shown below.

```
USE test;

CREATE TABLE person (
    person_id INT NOT NULL PRIMARY KEY,
    fname VARCHAR(40) NULL,
    lname VARCHAR(40) NULL,
    created TIMESTAMP
);
```

Now let us suppose we have a XML file whose content are as follows:

```
<list>
  <person person_id="1" fname="Kapek" lname="Sainnouine"/>
  <person person_id="2" fname="Sajon" lname="Rondela"/>
  <person person_id="3"><fname>Likame</fname><lname>Örrtmons</lname></person>
  <person person_id="4"><fname>Slar</fname><lname>Manlanth</lname></person>
  <person><field name="person_id">5</field><field name="fname">Stoma</field>
    <field name="lname">Milu</field></person>
  <person><field name="person_id">6</field><field name="fname">Nirtam</field>
    <field name="lname">Sklöd</field></person>
  <person person_id="7"><fname>Sungam</fname><lname>Dulbåd</lname></person>
  <person person_id="8" fname="Sraref" lname="Encmelt"/>
</list>
```

We can use the following statement to import data in *person* table from *person.xml* file:

```
mysql> LOAD XML LOCAL INFILE 'person.xml'
    ->   INTO TABLE person
    ->   ROWS IDENTIFIED BY '<person>';

Query OK, 8 rows affected (0.00 sec)
Records: 8  Deleted: 0  Skipped: 0  Warnings: 0
```

We can verify that 8 rows were imported into person table by the help of simple *Select statement.*

```
mysql> SELECT * FROM person;
+-----------+--------+------------+---------------------+
| person_id | fname  | lname      | created             |
+-----------+--------+------------+---------------------+
|         1 | Kapek  | Sainnouine | 2007-07-13 16:18:47 |
|         2 | Sajon  | Rondela    | 2007-07-13 16:18:47 |
|         3 | Likame | Örrtmons   | 2007-07-13 16:18:47 |
|         4 | Slar   | Manlanth   | 2007-07-13 16:18:47 |
|         5 | Stoma  | Nilu       | 2007-07-13 16:18:47 |
|         6 | Nirtam | Sklöd      | 2007-07-13 16:18:47 |
|         7 | Sungam | Dulbåd     | 2007-07-13 16:18:47 |
|         8 | Sreraf | Encmelt    | 2007-07-13 16:18:47 |
+-----------+--------+------------+---------------------+
8 rows in set (0.00 sec)
```

We can achieve the inverse of the above operation that is dumping MySQL table data into XML file by the help of *mysql* client:

```
shell> mysql --xml -e "SELECT * FROM test.person" > person-dump.xml
shell> cat person-dump.xml
<?xml version="1.0"?>

<resultset statement="SELECT * FROM test.person" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <row>
        <field name="person_id">1</field>
        <field name="fname">Kapek</field>
        <field name="lname">Sainnouine</field>
  </row>

  <row>
        <field name="person_id">2</field>
        <field name="fname">Sajon</field>
        <field name="lname">Rondela</field>
  </row>

  <row>
        <field name="person_id">3</field>
        <field name="fname">Likema</field>
        <field name="lname">Örrtmons</field>
  </row>

  <row>
        <field name="person_id">4</field>
        <field name="fname">Slar</field>
        <field name="lname">Manlanth</field>
  </row>

  <row>
        <field name="person_id">5</field>
        <field name="fname">Stoma</field>
        <field name="lname">Nilu</field>
  </row>

  <row>
        <field name="person_id">6</field>
        <field name="fname">Nirtam</field>
        <field name="lname">Sklöd</field>
  </row>

  <row>
        <field name="person_id">7</field>
        <field name="fname">Sungam</field>
        <field name="lname">Dulbåd</field>
  </row>

  <row>
        <field name="person_id">8</field>
        <field name="fname">Sreraf</field>
        <field name="lname">Encmelt</field>
  </row>
</resultset>
```

We can create a copy of *person* table and can very that the dump is valid like this.

```
mysql> USE test;
mysql> CREATE TABLE person2 LIKE person;
Query OK, 0 rows affected (0.00 sec)

mysql> LOAD XML LOCAL INFILE 'person-dump.xml'
    ->   INTO TABLE person2;
Query OK, 8 rows affected (0.01 sec)
Records: 8  Deleted: 0  Skipped: 0  Warnings: 0

mysql> SELECT * FROM person2;
+-----------+--------+------------+---------------------+
| person_id | fname  | lname      | created             |
+-----------+--------+------------+---------------------+
|         1 | Kapek  | Sainnouine | 2007-07-13 16:18:47 |
|         2 | Sajon  | Rondela    | 2007-07-13 16:18:47 |
|         3 | Likema | Örrtmons   | 2007-07-13 16:18:47 |
|         4 | Slar   | Manlanth   | 2007-07-13 16:18:47 |
|         5 | Stoma  | Nilu       | 2007-07-13 16:18:47 |
|         6 | Nirtam | Sklöd      | 2007-07-13 16:18:47 |
|         7 | Sungam | Dulbåd     | 2007-07-13 16:18:47 |
|         8 | Sreraf | Encmelt    | 2007-07-13 16:18:47 |
+-----------+--------+------------+---------------------+
8 rows in set (0.00 sec)
```

VI. CONCLUSION

We have seen a methodology above to store a XML data into relational database. The major issue in this is the uncertainty in XML data. And it's still very much a topic of research. In the above case study we have seen how to store and retrieved xml data using XPath and LOAD XML data technique. LOAD XML reads data from XML file into table. It acts as a compliment of mysql client in XML. We can achieve the reverse by dumping MySQL table data into XML file by the help of MySQL client.

REFERENCES

[1] W3C XML Path Language Specification, Latest. http://www.w3.org/TR/xpath

[2] Naresh Kumar, Satyanand Reddy V.E.S Murthy, "Storing, Querying, and Validating Fuzzy XML data in Relational Database", IJCSIT

[3] Amer-Yahia. S, Du. F and Freire. J, "A comprehensive solution to the XML-to-relational mapping problem," In: Proceedings of WIDM, pp 31–38, 2004.

[4] Abiteboul. S and Senellart. P, "Querying and updating probabilistic information in XML," In: Proceedings of EDBT, pp 1059–1068, 200

[5] Gaurav. A and Alhajj. R, "Incorporating fuzziness in XML and mapping fuzzy relational data into fuzzy XML", In: Proceedings of

[6] Valova. I, Milano. G, Bowen. K and Gueorguieva. N (2011) Bridging the fuzzy, neural and evolutionary paradigms for automatic target recognition. ApplIntell 35(2):211–225.

[7] Zajaczkowski. J and Verma. B (2012) Selection and impact of different topologies in multi-layered hierarchical fuzzy systems. ApplIntell36(3):564–584

[8] Xiaojie Yuan, "XML Data Storage and Query Optimization", JOURNAL OF SOFTWARE, VOL. 8, NO. 4, APRIL 2013

[9] Silivia Stefanova, "How to Store and Query XML Data"

[10] Artur Afonso de Sousa, Rui Pedro Duarte, José Luís Pereira, João Álvaro Carvalho," XML DATA STORAGE AND MANAGEMENT"

[11] "Indices and Querying in XML Databases", HAL Id: hal-01447949 https://hal.archivesouvertes.fr/hal-01447949v2

Submitted on 28 Jan 2017

[12] Shanmugasundaram, J., Tufte, K., He, G., Zhang, C., DeWitt, D. and Naughton, J., September 1999, Relational Databases for Querying XML Documents: Limitations and Opportunities, Proceedings of the 25th VLDB Conference, Edinburgh, Scotland.

[13] Tamino, 2003, Tamino - The XML power database. http://www.softwareag.com/Tamino

[14] Xindice, 2003, Frequently Asked Questions. http://xml.apache.org/xindice/faq.html

[15] LOAD XML SYNTAX, "https://dev.mysql.com/doc/refman/5.6/en/load-xml.html"