Asian Journal of Information Technology 10 (6): 234-239, 2011
ISSN: 1682-3915

Integration Intelligent Algorithm and XML Heterogeneous Database Access

Mohd Kamir Yusof, Ahmad Faisal Amri Abidin, Sufian Mat Deris and Surayati Usop
Universiti Sultan Zainal Abidin, 21300 Kuala Terengganu, Terengganu, Malaysia

Abstract: A good methodology for web query processing is needed to improve web query processing. In order to improve web processing in web based application, new and robust methods should be identified. In this research, web query processing for heterogeneous database access issue will determine and provide a proposed method to overcome this issue. The issue is time performance for searching and retrieving process in heterogeneous database access. In order to overcome this problem, intelligent algorithm was designed. This intelligent algorithm was integrated with XML language. XML is powerful tool for data exchange. Experiments have been done by comparing XML and integration between XML and proposed intelligent algorithm. The result indicates the integration between XML and proposed intelligent algorithm is better compared to XML in term of time performance.

Key words: XML, intelligent algorithm, heterogeneous database, time performance, Malaysia

INTRODUCTION

Web applications provided more information to web users. The main purpose of web applications is to provide useful information to web users. Useful information means this information can be used for analysis to make a decision etc. by web users. In order to produce the useful information to web users, good methodology is needed for web query processing. The web query processing is important element must be considered during web application development. Two major processes in web query processing are searching and retrieving process. The searching process occurs when web user request a query for example data mining. Generally, system will search data from databases. In this process, system will search any related data. Meanwhile, in retrieving process, system will retrieve any related data based on query entered by web user. Finally, the result or information after searching and retrieving process will display to web user. These processes also occur in heterogeneous databases. Heterogeneous database is a hot issue in database domain for researchers to find out a solution for query and integrating data from heterogeneous data sources. In the heterogeneous databases, data sources from different location will integrate using suitable method or technique. The suitable method for querying and integrating data from heterogeneous data sources is important to improve web query processing. XML is popular method was used in heterogeneous data sources. In this method, XML schema mapped directly to data sources. In this method, web query processing become faster for searching and retrieving process. Web application also displayed information to web users in a few seconds. In this case, web users are no needed to wait a long time a result from web application. However, a web query processing depends on query and number of data sources. For example, a query is data mining and number of data sources is 8. XML will carry a query and mapping to all data sources for searching and retrieving process. After that XML will carry a result. JQuery will load XML file and display information to web user. Based on example above, XML mapped to all data sources but not all the data sources contain data about data mining. In this case, intelligent algorithm is needed to help XML map to certain data sources that contain data about data mining. Motivation in this research is to develop algorithm and integrated this algorithm with XML for querying and integrating data from heterogeneous data sources.

HETEROGENEOUS DATABASE

Heterogeneous database means data sources are allocate at different places (Sujansky, 2001). These data sources are integrated using suitable method. The integration of data sources is important for web query processing. All data sources from different location must be identified for integration purpose. The data sources location must be included in main server. Generally, web query processing occurs at all data sources for data searching and retrieving. Good method or technique must be implemented in web query processing in order to improve web query processing especially in term of time processing. Many researchers are still looking and

Corresponding Author: Mohd Kamir Yusof, Universiti Sultan Zainal Abidin, 21300 Kuala Terengganu, Terengganu, Malaysia

develop more robust method for querying and integrating data from heterogeneous data sources (Butenuth *et al*., 2007). This happen caused of unprecedented increase in the availability of information due to the success of the World Wide Web has generated an urgent need for new and robust methods that simplify the querying and integration of data. Unified global integration schema is one of the methods for querying data sources. This method provided a relational schema in order to facilitate efficient global processing. However, their global schemas become hard to manage as the number and types of data sources increase (Collins *et al*., 2002). Wrapper and mediators is another approaches for querying data sources. This approach involves a system based on mediators and wrapper. Wrapper and mediators system are sophisticated applications that abstract the data source from the users. In addition, they translate queries into the terms of the data sources and integrate the results. The wrapper-mediator approach is remarkably scalable and allows of an increasing number of data sources. The most popular method for querying data sources is XML. In this research, XML will use to map only certain data sources. In addition to help XML to map only certain data sources, intelligent algorithm will design.

XML

XML stand for Extensible Markup Language. The XML is becoming standard for internet data representation as a mark-up language and for data exchange over the internet (Ozgur and Gundem, 2006). XML has been widely accepted as data format for modeling and exchanging data for internet application (El-Sayed *et al*., 2005). Beside that XML used for storing semi-structured data such product catalogs, book inventory, etc. XML allows users to ask very powerful queries on the web (Atay *et al*., 2007). A World wide Consortium (W3C) working draft of an XML query language has been proposed as a standard for querying XML.

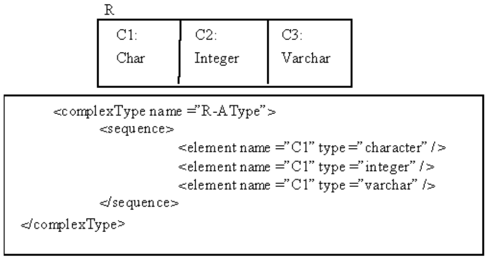
Relational database into XML

Step 1: Create a schema tag with the correct XML namespace information (Bray *et al*., 1999) (Fig. 1).

Step 2: Create complex type for each relation to hold the individual records for that relation. Therefore, for each relation R with attributes A1,…,An, create a complex type R-AType and include A1,…,An as elements with correct simple types. In addition, if A1,…,An is allowed to be null (Fig. 2).

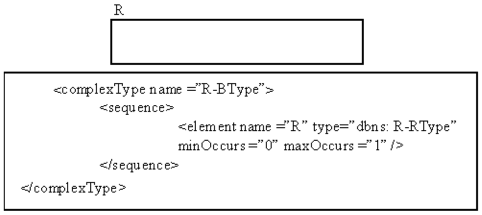
```
        <?xml version ="1.0" encoding ="UTF-8"?>
        <schema
                targetNamespace =”BookInventoryDB”
                xmlns =””
                xmlns:dbns =””
                elementFormDefault =”unqualified”
                attributeFormDefault =”unqualified”>
  </schema>
```

Fig. 1: XML namespace



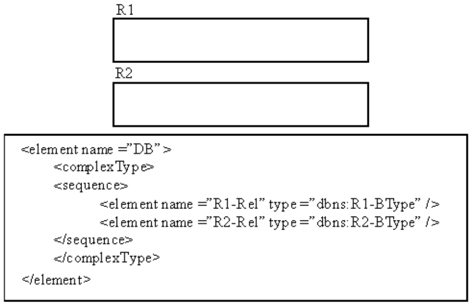
```
        <complexType name =”R-AType”>
                <sequence>
                        <element name =”C1” type =”character” />
                        <element name =”C1” type =”integer” />
                        <element name =”C1” type =”varchar” />
                </sequence>
  </complexType>
```

Fig. 2: Complex type for each relation



```
        <complexType name =”R-BType”>
                <sequence>
                        <element name =”R” type=”dbns: R-RType”
                        minOccurs =”0” maxOccurs =”1” />
                </sequence>
  </complexType>
```

Fig. 3: Complex type R-BType

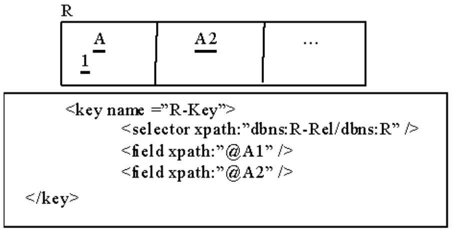


```
 <element name =”DB” >
      <complexType>
      <sequence>
           <element name =”R1-Rel” type =”dbns:R1-BType” />
           <element name =”R2-Rel” type =”dbns:R2-BType” />
      </sequence>
      </complexType>
 </element>
```

Fig. 4: Database element

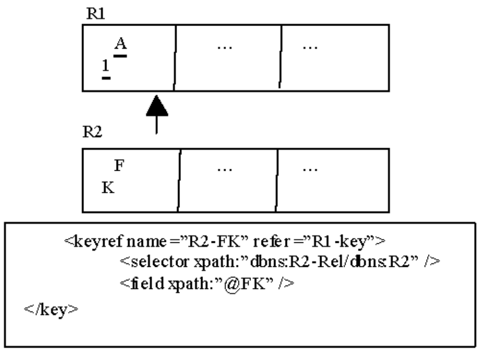
Step 3: Create a complex type R-BType and include an element R of type R-Type. Set the minOccurs attribute to 0 and max. Occurs attribute to 1 for each element (Fig. 3).

Step 4: Create database element. Assign DB with relation R1,….Rn, add a DB element to shema and insert an anonymous complex type. Then, include R1-BType, Rn-BType as elements of corresponding complex types Ri,….,BType (Fig. 4).



```
<key name ="R-Key">
        <selector xpath:"dbns:R-Rel/dbns:R" />
        <field xpath:"@A1" />
        <field xpath:"@A2" />
</key>
```

Fig. 5: Assign a primary key



```
<keyref name ="R2-FK" refer ="R1-key">
        <selector xpath:"dbns:R2-Rel/dbns:R2" />
        <field xpath:"@FK" />
</key>
```

Fig. 6: Assign a foreign key

Step 5: Assign a primary key. If a primary key exists with attributes A1,..Aj, insert tag R-key to the database element. Add a selector tag with the xpath value set to dbns: Rel/dbns: R to the key tag. In addition, insert field tags for the attributes A1,…,Aj (Fig. 5).

Step 6: Assign a foreign key. For each relation R with foreign keys FK,…,FKn, insert a keyref tag R-FKi for each foreign key. Insert a selector tag with the xpath value set to R-Rel/R and field tag (s) with xpath value set to @FK (Fig. 6).

PROPOSED INTELLIGENT ALGORITHM

This algorithm is needed to ensure XML map to only certain data sources (Fig. 7).

Based on algorithm in Fig. 7 information (K) was entered by web user will assign as an initial query. First, this initial will match with any keywords from Data Scheme (DS). All information about data sources destination, database scheme and relationship among tables are stored in data scheme file. Second, if k is not in data scheme file, searching and retrieving process will occur at all data sources. After data is found, all information about data sources destination, database scheme and relationship among tables will store in data scheme file. Third, if k is available in data scheme file, searching and retrieving process will directly point to certain data sources. Finally, the result will display to web user through web user interface. Table 1 shows how to generate data scheme structure. This important to ensure searching and retrieving process occur to only certain data sources. Table 1 also relationship between data source and database scheme will identified. The relation among tables in database scheme is based on primary key. For instance in DS, relationship between T1 and T2 is based on X1 and relationship between T2 and T3 based on Y1. Finally, structure mapping was created, DS → T1 → T2 → T3. The structure mapping and query/keywords was enter by web user will store into data scheme file.

```
Begin
K → query; //initial query (i.e., Data Mining)
Searching from data scheme file
Assign data scheme file equal to DS
While (DS = NULL)
{
     SDD = DS; // Assign value for structure data sources
     (contains data sources location)
     IF (K = SDD)
     {
          K → { $(DSN_1, TN_1, TR_1), (DSN_2, TN_2, TR_2), (DSN_n, TN_n, TR_n)$ }
          // K point to data source location, table name and their relations.
          // XML document will point directly to data source location.
     }
     ELSE
     {
          // search all data sources.
          SDD = { $(DSN_1, TN_1, TR_1), (DSN_2, TN_2, TR_2), (DSN_n, TN_n, TR_n)$ }
          // store SDD into data scheme file.
     }
}
End
```

Fig. 7: Proposed algorithm

Table 1: Structure mapping

Queries/ Keywords	Data source	Database scheme and their relationship	Structure mapping
Q1	DS1	T1 ($\underline{X}_1$, X_2, X_3, X_4)	DS1→T1→T2→T3
		T2 (Y_1, $\underline{X}_1$, Y_2, Y_3, Y_4)	
		T3 (Z_1,$\underline{Y}_1$,Z_2)	
Q2	DS2	T8 ($\underline{X}_1$, X_2, X_3)	DS2→T8→T9
		T9 (Y_1, $\underline{X}_1$, Y_2, Y_3)	
	DS3	T12 ($\underline{X}_1$, X_2, X_3)	DS3→T12→T13
		T13 (Y_1, $\underline{X}_1$, Y_2, Y_3, Y_4)	
Q3	D5	T4 ($\underline{X}_1$, X_2, X_3, X_4)	D5→T4→T5
		T5 (Y_1, $\underline{X}_1$, Y_2, Y_3)	
	D7	T14 ($\underline{X}_1$, X_2)	D7→T14→T15
		T15 (Y_1, $\underline{X}_1$, Y_2, Y_3)	
	D8	T19 ($\underline{X}_1$, X_2, X_3)	D8→T19→T20
		T20 (Y_1, $\underline{X}_1$, Y_2, Y_3, Y_4)	
Q5	DS1	T1 ($\underline{X}_1$, X_2, X_3, X_4)	DS1→T1→T2→T3
		T2 (Y_1,$\underline{X}_1$, Y_2, Y_3, Y_4)	
		T3 (Z_1,$\underline{Y}_1$,Z_2)	
	DS3	T12 ($\underline{X}_1$, X_2, X3)	DS3→T12→T13
		T13 (Y_1, $\underline{X}_1$, Y_2, Y_3, Y_4)	
:	:	:	:
:	:	:	:
:	:	:	:
Qn	DSn	Tn (X_n, X_{n+1}, X_{n+1}, X_{n+i})	Dsn→Tn

EXPERIMENTAL RESULTS

Prototype architecture: The architecture must be designed before implementation phase. The major

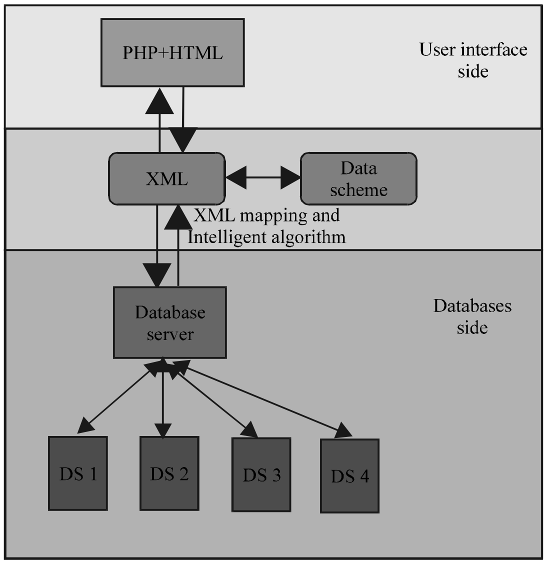


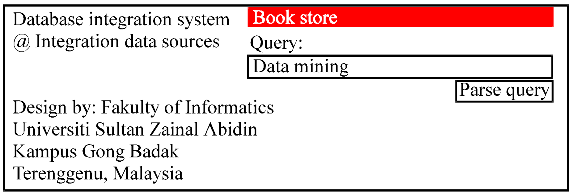
Fig. 8: Architecture

components were identified to design the architecture. Based on Fig. 8, three major components in this architecture are user interface, XML mapping and intelligent algorithm side and databases server. The user interface allows web users enter a query (i.e., data mining). System will search and retrieve any related data based on the query. This interface will display the result to web user. The query from web user will convert to XML document. Two methods to read XML document are Simple API for XML parsing (SAX) and Document Object Method (DOM). In this DOM was applied to read XML document because DOM is versatile compared to SAX (Williams *et al*., 2005). The XML document will communicate with data schema to check data (query). This communication will control by intelligent algorithm. This intelligent will find accurate data from data scheme. After that XML will map to database server. The database server connected to data sources. Connection between database server and data sources is need so that searching and retrieving process can occur smoothly. The data sources contain lot of data. Normally, data sources allocated at different places. These three components are related with each other in order to improve web query processing.

Implementation and results: In this implementation, 6 data sources are connected to a database server. The purpose of this connection is to ensure searching and retrieving process occurs at all these data sources. Table 2 shows the software and hardware specifications for each data sources.

Table 2: Software and hardware specification for data sources

Data sources	Specifications
Data source 1, Data source 2, Data source 3, Data dource 4, Data source 5, Data source 6	Operating system: Linux Processor: Intel (R) Core (TM) i3 CPU 2.93 GHz RAM: 2.00 GB System type: 32-bit operating system

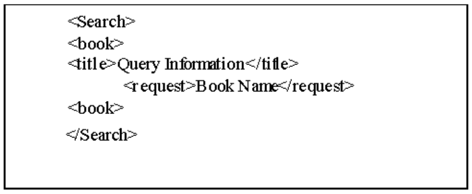


Database integration system
@ Integration data sources

Book store
Query:
Data mining

Design by: Fakulty of Informatics
Universiti Sultan Zainal Abidin
Kampus Gong Badak
Terenggenu, Malaysia

Fig. 9: User interface



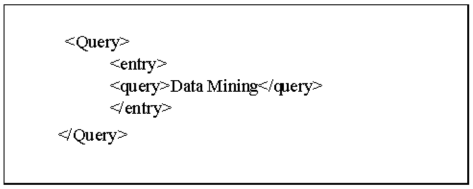
```
<Search>
<book>
<title>Query Information</title>
        <request>Book Name</request>
<book>
</Search>
```

Fig. 10: Search.xml

The prototype architecture was implemented as a web application using PHP, JQuery and HTML. PHP and HTML used to create user interface and JQuery used to load XML document. This development environment was chosen because it would make the system portable and easily accessible through the World Wide Web. This web application can access anywhere and anywhere as long as internet connection exists. User interface (Fig. 9) is created using PHP to gather information from web user. After that the information from web user will convert to XML document. The XML document will communicate with data scheme. This purpose of this communication is to check availability of query in data dictionary. The data dictionary is allocated at data scheme. Next process is searching and retrieving process. XML will directly map to data sources. A result from the retrieving process will convert to XML document. In this implementation, DOM is used to read XML document. The result will display to web user through web user interface.

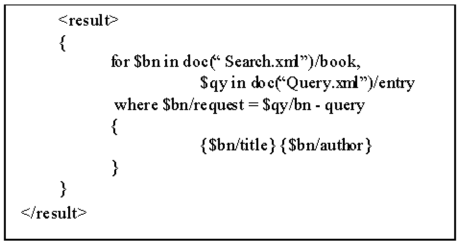
Web user is needed to key in a query (i.e., data mining). Then, click on parse query button. XML document will create. Figure 10 shows XML document for a query Data Mining.

In Fig. 11, information (Data Mining) enter by web user was carried for searching and retrieving process. This XML was created as a Query.xml.



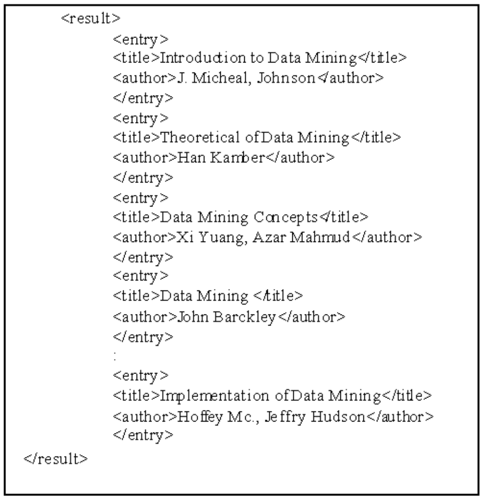
```
    <Query>
        <entry>
        <query>Data Mining</query>
        </entry>
   </Query>
```

Fig. 11: Query.xml



```
      <result>
      {
              for $bn in doc(" Search.xml")/book,
                          $qy in doc("Query.xml")/entry
               where $bn/request = $qy/bn - query
              {
                          {$bn/title} {$bn/author}
              }
      }
 </result>
```

Fig. 12: Query



```
     <result>
          <entry>
          <title>Introduction to Data Mining</title>
          <author>J. Micheal, Johnson</author>
          </entry>
          <entry>
          <title>Theoretical of Data Mining</title>
          <author>Han Kamber</author>
          </entry>
          <entry>
          <title>Data Mining Concepts</title>
          <author>Xi Yuang, Azar Mahmud</author>
          </entry>
          <entry>
          <title>Data Mining </title>
          <author>John Barckley</author>
          </entry>
          :
          <entry>
          <title>Implementation of Data Mining</title>
          <author>Hoffey Mc., Jeffry Hudson</author>
          </entry>
 </result>
```

Fig. 13: Result.xml

In Fig. 12, sear ching and retrieving process occurs. This query is important to search and retrieve any relevant from certain data source. In this part, communication between XML mapping and data scheme is needed. This communication is to identified and point directly to certain data sources that contain data mining information.

Result.xml document will create after searching and retrieving process have been done. After result is found all information about data mining will convert into XML document (Result.xml). JQuery will use to load this XML document and the result will display to web user through web user interface. The web user interface was created using PHP and HTML (Fig. 13).

Table 3: Comparison results between XML mapping and proposed algorithm (1st iteration)

	XML mapping		XML mapping+ Proposed algorithm	
Query	No. of relevant data	Time performance	No. of relevant data	Time performance
Data mining	45	10 sec	45	10 sec
Data warehouse	56	15 sec	56	15 sec
Information system	33	8 sec	33	8 sec
The intelligent information	27	6 sec	27	6 sec
Data dictionary	127	14 sec	127	14 sec
Management system	49	10 sec	49	10 sec
Management information system	139	15 sec	139	15 sec
Transport system	24	3 sec	24	3 sec
Core activities sport	67	15 sec	67	15 sec
Dream	101	9 sec	101	9 sec

Table 4: Comparison results between XML mapping and proposed algorithm (2nd iteration)

	XML mapping		XML mapping+ Proposed algorithm	
Query	No. of relevant data	Time performance	No. of relevant data	Time performance
Data mining	45	10 sec	45	7 sec
Data warehouse	56	15 sec	56	12 sec
Information system	33	8 sec	33	6 sec
The intelligent information	27	6 sec	27	4 sec
Data dictionary	127	14 sec	127	12 sec
Management system	49	10 sec	49	9 sec
Management information system	139	15 sec	139	11 sec
Transport system	24	3 sec	24	2 sec
Core activities sport	67	15 sec	67	10 sec
Dream	101	9 sec	101	4 sec

Table 3 shows the comparison between XML mapping and combination between XML and proposed intelligent algorithm. Two measurements in this comparison are number of relevant data and time performance. In Table 3, queries were entered by web user for the first time. The result shows no differentiation in term time number of relevant data or time performance between XML mapping and combination between XML mapping and proposed intelligent algorithm. Table 4 shows these same queries were entered by web users for the second time. In Table 4, the result shows no differentiation between XML mapping and combination between XML mapping and proposed intelligent algorithm in term of number of relevant data. But the result shows combination between XML mapping and proposed intelligent algorithm is better compared to XML mapping in term of time performance. In this experiments, 10 queries were enter by web user was tested and the results shows combination between XML mapping and proposed intelligent algorithm able to improve web query processing for heterogeneous database access.

Analysis of proposed intelligent algorithm: Table 3 and 4 shows the comparison between XML mapping and combination between XML mapping and proposed intelligent algorithm (Fig. 7). In Table 4, combination between XML and proposed intelligent algorithm shows time performance decrease from 10-7 sec for data mining information. This time performance for web query processing was decreased about 30%. Same thing was happened to data warehouse information. The result shows time performance for web query processing was deducted from 15-12 sec. In term time performance, implementation of proposed intelligent algorithm for web query processing was increased about 10-30% average for number of word between 1 and 2. Different with number of words equal to 3, implementation of proposed intelligent algorithm in web query processing was decreased about 15-11 sec for Management Information system in term of time performance. In term of time performance, this proposed intelligent algorithm was increased web query processing about 30-40% according to sample of queries where the number of word equal to 3. In conclusion, based on analysis results in Table 4, implementation of the proposed intelligent algorithm was improved web query processing for heterogeneous database access. This improving of web query processing in term of time performance was proved by tested of sample of queries in Table 4.

CONCLUSION

This research demonstrates integration between proposed intelligent algorithms and XML. The proposed algorithm was created and implemented by integration with XML. The main function this algorithm is to store a keyword or information with source destination in data scheme. This data scheme with communicate with XML document. This integration between data scheme and XML in needed to improve web query processing for heterogeneous. Based on experiment the result indicate the proposed intelligent algorithm able to improve web query processing for heterogeneous database access in term of time performance.

REFERENCES

Atay, M., A. Chebotko, D. Liu, S. Lu and F. Fotouhi, 2007. Efficient schema-based XML-to-relational data mapping. Inform. Syst., 32: 458-476.

Bray, T., D. Hollander and A. Layman, 1999. Namespaces in XML. http://www.w3.org/TR/1999/REC-xml-names-19990114.

Butenuth, M., G.V. Gosseln, M. Tiedge, C. Heipke, U. Lipeck and M. Sester, 2007. Integration of heterogeneous geospital data in a federated database. J. Photogrammetry Remote Sens., 62: 328-346.

Collins, S.R., S. Navathe and L. Mark, 2002. XML schema mapping for heterogeneous database access. Inform. Software Technol., 44: 251-257.

El-Sayed, M., K. Dimitrova and E.A. Rundensteiner, 2005. Efficiently supporting order in XML query processing. Data Knowledge Eng., 54: 355-390.

Ozgur, A. and T.I. Gundem, 2006. Efficient indexing technique for XML-based electronic product catalogs. Electr. Commerce Res. Appl., 5: 66-77.

Sujansky, W., 2001. Heterogeneous database integration in biomedicine. J. Biomed. Inform., 34: 285-298.

Williams, M.E., G.R. Consolazio and M.I. Hoit, 2005. Data storage and extraction in engineering software using XML. Adv. Eng. Software, 36: 709-719.