Comparative study on Relational data sets and XML data sets in J2EE system

1.B.Hemanth Kumar, Asst.,Prof.,
2. Prof. M.Surendra Prasad Babu,
3. M.Kiran Kumar, Lecturer

[1,3]Dept., of Information Technology,
RVR & JC College of Engineering,
Guntur, A.P. India

2. Dept., of CS & SE
AU college of Engineering,
Andhra University,
Visakhapatnam,A.P. India.

Abstract:--In the real world extracting information entirely depends up on the type of the information, the type of the structures used to maintain the information and the type of the application programs used. Most commonly available information is in the form of text; most of the systems use relational database systems to maintain text information. Another technology used to maintain the text information is the XML technology. The application performance with respect to response time is an important criterion for Internet-based services. The time factor plays a main role for the end user to survive in severe applications. Getting information from the xml data sets is more efficient than the relational data sets. In this paper we have studied different types of XML and RDB(Relational Data Base) data sets and compared them with respect to access time in J2EE system. The analyses has been made purely in J2EE environment and we have been used the technology JDBC to extract data from relational data sets and Java XML parsers to parse and extract the xml data sets.

Key words:-xml, rdb, parsers, jdbc

I. INTRODUCTION *(HEADING 1)*

Performance is the main criteria in real world applications; the application performance basically depends up on the response time and decreases when the response time is more. The extraction of data on the Internet depends upon the type of the data structures used to maintain the data and the type of the application programs used to retrieve the data. Application programs request the data sets and the database servers respond to these requests with respect to the type of the data structures available on the server. In this section we have given some introduction on the technologies used to analyze the application performance on the relational data sets and xml data sets. The section 2 contains discussion about the application program used to extract the information from the relational data sets as well as from the XML data sets. The section 3 discusses the design issues of the data sets and the section 4 contains the performance analysis on different data sets and last conclusions based on the experimental results.

A. XML:

One of the most significant recent additions to J2EE is its support for Web services[12] and its integration of the eXtensible Markup Language(XML) [2] to facilitate faster development of the enterprise applications. To support Web services, new application-programming Interfaces (API) such as API for XML-based Remote Procedure Calls (JAX-RPC) have been added. XML was accepted as a standard by the World Wide Web Consortium (W3C). In the initial days, XML was primarily used as a more flexible markup language for use in Web pages; however, as developers familiarized themselves with XML and its related technologies, a whole host of new uses such as database manipulation, configuration manipulation, and service description have been presented. XML has its root in Standard Generalized Markup Language (SGML) and Hypertext Markup Language. XML represented a concerted effort to take the best principles from SGML and develop a simpler and more generic language. The basic concept of the XML is to use ***tags*** similar to the markup tags used in HTML to identify data in order to make it easier for another application to understand the context of the data. An XML document can be created, read, and manipulated by using a J2EE component. A J2EE application can generate an XML document based on business rules that are encoded into one or more J2EE components.

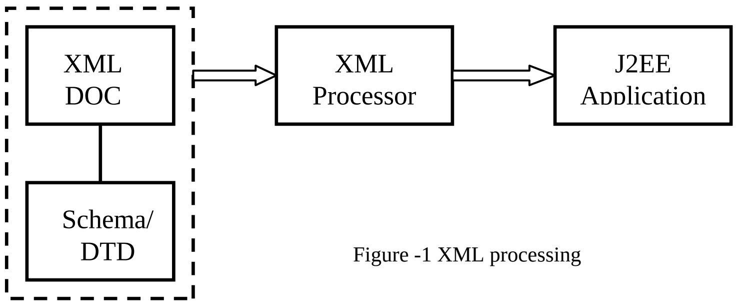

Figure -1 XML processing

B. Relational Data Base

The primary goal of the DBMS[13] is to provide an environment that is both convenient and efficient to use in retrieving and storing the database information. Relational database system consists of a collection of tables representing both the data and the relationships among those tables. Each table has multiple columns and each column has a unique name. A row in a table represents a relation ship among a set of values.

C. Parsers

XML is parsed by a piece of software called an XML parser[2]. The functionality required of an XML parser has been defined by the W3C. Basically, an XML parser [5]provides an application with access to the elements within a document, thus becoming the link between the document and the application. The parser is responsible for making sure that the XML document is well formed and if it is valid. Figure-1 provides a high-level overview of the processing of an XML document. An XML document along with its associated schema/DTD is input into an XML parser. The parser checks the document to make sure that is well formed; If the schema/DTD is also available the parser checks the validity of the XML data according to what has been defined in the schema or DTD. The parser then provides the access methods for another application to access the data of the original XML document. The document can be parsed in two ways: by using Document Object Model (DOM) parser and by using Simple API for XML (SAX) parser[2]. A DOM parser[2] reads an entire document at once and holds a corresponding tree-like structure in memory. Each element within the document is a node in the tree. Using the org.w3c.dom.Node interface specified by the W3C, another application can traverse or modify the tree. On the other hand a SAX parser is an event-driven parser. It reads the document sequentially without storing the structure into memory, thus making SAX more efficient than DOM. At the start of the document, SAX parser begins by making callback functions based on the tags it encounters. Most XML parsers support SAX, although the W3C standards group has not officially accepted it.

D. JDBC.

Business applications usually store data in databases. Currently, the relational-database management systems (RDBMS) are the most popular ones. They store data in tables that consists of rows and columns, and understand Structure Query Language (SQL). One of the technology available for accessing relational data is JDBC(Java Database Connectivity) . JDBC drivers[1] play the role of middleman between a java program and an RDBMS. The drivers are available from the database vendors and also from the vendors of J2EE application servers. **Sun Microsystems** also provides a reference implementation of drivers for each specification. The list of available JDBC drivers can be found at the following URL: http://developers.sun.com/product/jdbc/drivers. Practically every J2EE application [11] saves, retrieves, and manipulates the information stored in a database using the web services provided by a J2EE component. A J2EE component supplies the database access using Java data objects contained in the JDBC application program interface (API). Java data objects have methods that open a connection to a database management system(DBMS) and then transmit queries to insert, retrieve, modify, or delete the data stored in a database.

II. DATA EXTRACTION

We have developed an application program in Java to extract information from the relational data sets as well as from the XML data sets. The application program maintains the response time for each data set. The inputs for this application program are relational data set and xml data set. The same type of data and the same number fields are maintained in relational data set and xml data set, i.e., the input given to this application program has the same quantity and same type of data in the relational data set as well as in the xml data set.

A. Extraction of data from the relational data sets

The JDBC technology has been used to extract data from the relational data sets.

Steps to calculate the time interval to extract data from relational data sets are:

i) Create a DSN (data source name) on the system which is an interface to the java application to connect relational data base.
ii) set Initial time : Initial time=system time in nanoseconds before data extraction.
iii) In application program load the data base drivers by using the statement
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
iv) Establish a connection between the application program and the relational database by using connection statement and DSN as shown
Connection con= DriverManager.getConnection("jdbc:odbc:dsn","username","password");
v) Create statement object by using connection: Statement stmt=con.createStatement();
vi) Execute query by using statement object which returns Reultset that contains data: Resultset results= stmt.executeQuery(qry).
vii) Extract data from Resultset.
viii) Set Final time: Final time= system time in nanoseconds after data extraction
ix) Time interval= Final time – Initial time.

B. Extraction of data from the xml data sets.

The XML DOM parser has been used to extract data from the XML data sets.
Steps to calculate the time interval to extract data from the XML data sets are:

i) set Initial time : Initial time=system time in nanoseconds before data extraction.
ii) Create an instance of DocumentBuilderFactory and ignore the white space in the element content:
DocumentBuilderFactory DBF=DocumentBuilderFactory.newInstance();
DBF.setIgnoringElementContentWhitespace(true);
ii) Create a document builder: DocumentBuilder DB=BDF.newDocumentBuilder(); Thedocument builder contains the parser to parse the XML document
iii) Parse the XML document . When parsing the document by using Document Builder it will return the reference of the XML document. The reference is used to get the elements of the XML document.
Document DOC=BD.parse(new File(“name of xml document”));
iv) Get the root element from the document: Element ROOT=DOC.getDocumentElement();
v) Get child nodes of the root element : NodeList CDLT=ROOT.getChildNodes(); CDLT.getLength() gives the number of child elements in the list.
vi) Get child elements from this child nodes.
first child element chel1: Element chld1=CDLT.item(0), secondchild element chel2: Element chld1=CDLT.item(1), get all child nodes by using any control structure.
vii) Get Tag name of the element: Element..getElementsByTagName(node name);
viii) Get the value of the element : Tagname.getData(); extract values of all elements.
ix) Set Final time: Final time= system time in nanoseconds after data extraction
x) Time interval= Final time – Initial time.

III. DESIGN OF DATA SETS

Relational data bases are designed as number of tuples and number of columns. Each relational data set represents one table in a relational data base; XML data sets are designed from these tables. In XML the root element represents the table name and each row represents one sub-element of the root element. This sub-element name is treated as record, i.e., for each row of the table there exists one element of type record in the XML data set. The columns of the tables are treated as child elements of the element record. The value of the child element is the corresponding value of the relational data set. The schema of the relational data set represents the Document Type Definition of the XML data set.

IV. COMPARATIVE STUDY

All experiments were conducted on top of Windows 7, Oracle 8i software installed on a 3.00GHz PC with 3.00GB RAM, 400 GB hard disk. Figure-2 shows a comparison for some sample experiments with different frequencies for relational data sets and XML data sets. The frequency of a data set in a relational model is the combination of number of tuples and columns. The frequency of a data set in xml is the combination of the number of elements and its sub-elements. We have conducted experiments for 200 data sets with different frequencies. As shown in the figure-2, the time interval to extract information from the relational data set is always greater than the time interval to extract information from the xml data set for the same frequency of data sets. The following table (Table-1) shows the experimental results conducted for different number of fields and different number of records for the same data in the relational system as well as in the XML system. The last column represents the time difference between the RDB and the XML data sets.

Table -1 : EXPERIMENTAL RESULTS.

No. of FIELDS	No.of Records	Time interval (nanoseconds) for Relational data **RDB**	Time interval (nanoseconds) For XML data **XML**	Time difference between RDB and XML (nanoseconds) **RDB-XML**
1	10	185863649	32742720	153120929
1	20	169388172	34855792	134532380
1	60	211172042	46726206	164445836
1	120	170528002	57525405	113002597
1	180	178389040	67048567	111340473
1	200	235734885	79651216	156083669
2	40	147962165	43849833	104112332
2	60	183208142	51650790	131557352
2	140	208520972	78132126	130388846
2	180	254103935	89137512	164966423
2	200	190170357	96174486	93995871
3	40	158172845	52445838	105727007
3	100	181092339	76240261	104852078
3	160	204947184	98692422	106254762
3	200	249104931	112701851	136403080
4	40	150436064	54106256	96329808
4	60	171419315	66073277	105346038
4	100	192110697	87556292	104554405
4	140	164658167	105201640	59456527
4	180	192065294	122567066	69498228
5	40	145578729	58903852	86674877
5	100	175492868	97683677	77809191
5	180	258743818	137364716	121379102
5	200	261818526	152911670	108906856
6	20	149777564	48018286	101759278
6	60	207620783	79828387	127792396
6	100	195012331	106145865	88866466
6	140	234522685	135466705	99055980
6	160	190369033	142581171	47787862
6	200	198996444	163097088	35899356
7	20	152162025	51958705	100203320
7	60	196010493	86218809	109791684
7	100	200702605	115556035	85146570
7	120	209900443	130744893	79155550
7	140	273827194	142856996	130970198
7	180	222648516	166308685	56339831
7	200	227383300	180359762	47023538
8	40	227462156	78699481	148762675
8	80	255842184	108155161	147687023
8	100	238174988	126036054	112138934
8	180	220558315	181062981	39495334
8	200	270900640	195710771	75189869
9	40	197432293	78362209	119070084
9	60	261236833	97946872	163289961
9	80	228541222	116513915	112027307
9	120	312164605	150141463	162023142
9	160	286177231	182103473	104073758
9	200	240423926	209497287	30926639
10	20	149744793	58389750	91355043
10	60	231196047	103679135	127516912
10	100	272150732	142202592	129948140
10	140	278076892	178176709	99900183

We have conducted the experiments depending on the number of data items available in the relational data sets and the XML data sets. As per the experimental results as shown in the Table-2, Figure-3 shows the timeline for extracting data from respective data sets. Figure2 shows that the timeline for relational data sets is always above the timeline of XML data sets. As the number of data items increases, the time interval increases in both the relational data sets as well as the XML date sets. The experimental result shows , in any case the time interval of the XML data sets is always less than the time interval of the relational data sets.

TABLE 2

No of data items	Time interval in nanoseconds	
	RDB data	XML data
10	185863649	32742720
30	163293375	36812859
40	155089944	36619985
60	143267322	40261707
80	147962165	43849833
90	179260895	44123611
120	158172845	52445838
150	219050148	53689786
180	206607259	58398967
200	196692888	59974383
240	171419315	66073277
280	156566363	71179470
320	255234206	76656040
360	230284934	80892433
400	192110697	87556292
440	227424946	91511048
480	191771376	99950023
520	199375022	100204343
550	200856562	106252031
600	175304774	107127301
650	181101215	112746912
660	247501181	112738719
720	251685337	118888135
780	237479621	125071346
840	234522685	135466705
900	231993142	136534848
960	190369033	142581171
980	273827194	142856996
1050	235749563	148703618
1120	241884642	155657299
1190	218221305	161989347
1200	208964411	159960592
1280	241095397	167439639
1360	253548869	174559224
1440	220558315	181062981
1520	263266271	189037013
1530	272494832	189972021
1620	252793762	200114768
1710	264774096	212465487
1800	240423926	209497287
1900	231242473	215619734
2000	238807886	222404437

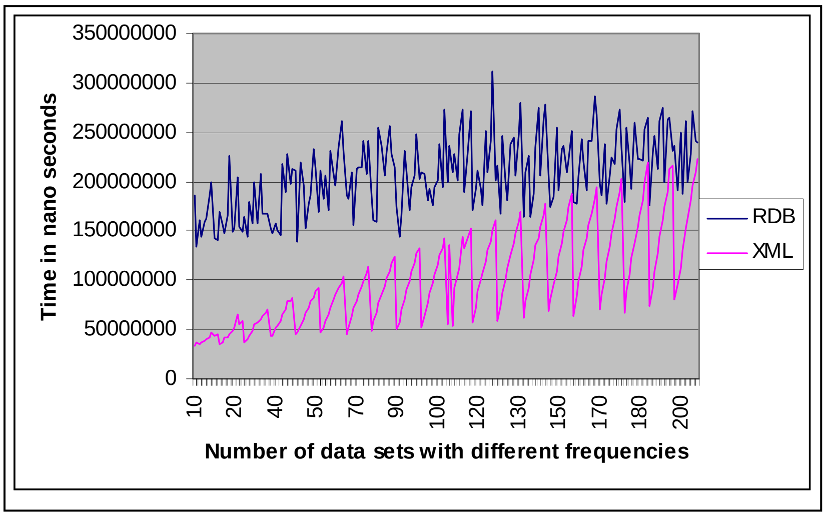

Figure 2: Time comparisons for XML and RDB data sets

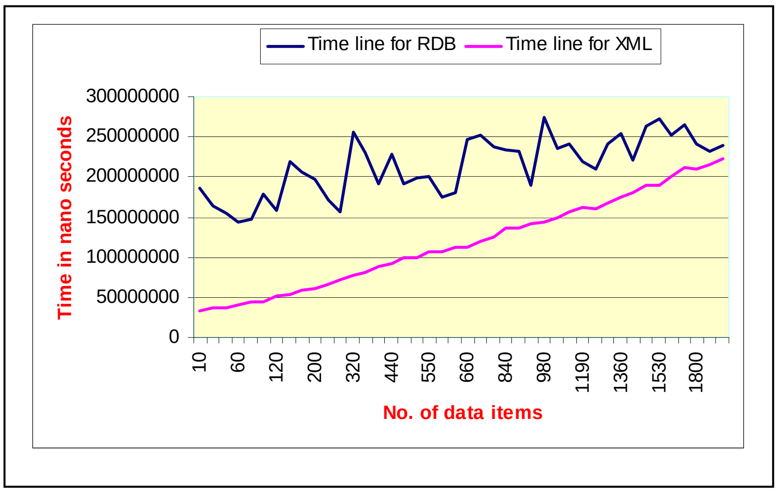

Figure 3:Time line showing for number of data items

V. CONCLUSIONS.

As per the results shown by the above experiments, to improve the efficiency of the system with respect to data extraction, it is always preferable to maintain the data in XML format than RDB. Applications which will process only text data can directly interact with xml data sets. As per W3C recommendations, in Semantic web[6] the data will be represented in RDF[6] format which follows the XML syntax. So the XML data sets can be implemented in the Semantic web systems. The disadvantage of maintaining the text information in RDB is that the application program has to identify the database server and again the database within that server for each query. It means the application needs two operations to extract information from the relational database which is a time consuming process. But when you are representing data in a XML application, the XML data and the application program can be maintained within in the same root. So to extract the information from the

XML data the application needs only one operation i.e. identifying the XML data. So in most of the web applications it is better to maintain the text information in the XML format than in the RDB format. Web services are dynamic technologies that will increasingly help to integrate independent systems, provided by business partners participating in multiple enterprises. We can build new databases based on open collaboration and standards (XML Web Services) that are accessible and interoperable and promote the use of XML Web Services. The performance of all the dynamic web services depends up on the response time. By minimizing the response time we can increase the performance of a system. All the experiments summarized in the paper are purely dependent up on the system configuration, the processor and the type of the application programs currently executing in the system

REFERENCES

[1] James MCGovern, Rehan Aditya, Yokovfain, “J2EE 1.4 BIBLE”, Wiley dream tech.

[2] Jim Koeg, “The Complete reference J2EE “ ,Tata Mcgraw Hill

[3] Internet & World Wide Web How to Program, 4/e , Harvey M. Deitel and Paul J. Deitel

[4] Chen Zhifeng “Research on the Application of XML in Web Services” 2009 IEEE

[5] Wei Zhang and Robert A. van Engelen “An Adaptive XML Parser for Developing High-Performance Web Services” 978-0-7695-3535-7/08 © 2008 IEEE

[6] Gottfried Vossen, Stephen Hagemann, “Unleashing web 2.0 from concepts to creativity” Morgan Kaufmann publishers,.

[7] Cay Hostman, John Wiley&sons, “Big Java “ Pearson.

[8] http://www.oracle.com/technetwork/java/javase/jdbc/index.html

[9] http://w3schools.com/xml/xml_http.asp

[10] J2EE Design Patterns By William Crawford, Jonathan Kaplan Publisher: O'Reilly Media

[11] Enterprise Applications in Java and J2EE Publisher: O'Reilly Media

[12] Java Web Services by David Chappell and Tyler Jewell

[13] Database System Concepts by Abraham Silberschatz, Henry F. Korth., McGraw-Hill.

[14] XML Schema Part 0: Primer, W3C Working Draft, “http://www.w3.org /TR/xmlschema-0/”.