XML based Heterogeneous Database Integration System Design and Implementation

Yogesh R. Rochlani
Department of Computer Science & Engineering
Sipna COET, SGBAU, Amravati (MH), India
yrochlani174@gmail.com

Prof. A.R. Itkikar
Department of Computer Science & Engineering
Sipna COET, SGBAU, Amravati (MH), India
abhijit_itkikar@rediffmail.com

Abstract—in the past decade, research works in heterogeneous database integration have established a good and solid framework to alleviate this task. However, there are still works that need to be accomplished to make these achievements easily implementable. In our project, we shall develop a software tool using XML for integrating and querying disparate heterogeneous information as unified XML views.

I. INTRODUCTION

Data integration is the process of combining data that is residing in different sources and providing users with a unified view of these data. The process involves standardization of data definition by using a common conceptual schema across a collection of data sources. Integrated data will be consistent and logically compatible in different systems or databases, and can use across time and users.

Nowadays organizations not only are increasing the data volume, but also they have to work with a large variety of data sources with different types of data. One of the main challenges for these and others environments will be how to work and integrate this heterogeneous information from different types of databases. Mainly when organizations get into the web it has to manage new types of web data such as XML or HTML, for example to exchange information with other organizations or to publish its information on the web.

In recent years, there have been many research projects focusing on heterogeneous information integration system. The goal of such a system is to intercept the user queries and to find the more adequate data from several heterogeneous resources, to answer the queries of the user, and to turn over the result in a transparent way to the users. The user does not need to know the nature, the type or the localization of the data, in which language they are programmed and on which operating system they are lodged, or no other system aspects which do not form part of the interface of the required sources.

The system based on XML, as an exchange model, (i.e., it is rich, clear, extensible and secure), makes it the best candidate for supporting the integrated data model. In addition, using XML views for local data sources hides the local specificities of each system. Furthermore, the richness of the XML schema model simplifies wrapper mappings. Also, the emergence of XQuery as a powerful universal query language for XML makes it possible to query XML global and local views in a uniform way based on a standard interface.

In particular, our data integration system provides access to pre-existing data sources which were created independently in Oracle Database and SQL Server respectively. In particular, a data integration
system requires flexible mechanism for describing contents of sources that may have overlapping contents.

II. LITERATURE REVIEW AND RELATED WORK

Data integration has received significant attention since the early days of databases. In the recent years, there have been several works focusing on heterogeneous information integration. Most of them are based on common Mediator Architecture [12]. In this architecture, mediators provide a uniform user interface to views of heterogeneous data sources. They resolve queries over global concepts into sub queries over data sources. Mainly, they can be classified into structural approach and semantic approach.

Structural approach need to specify mappings between the unified schema and the schema of original sources, and transform a query into specialized queries to match the schema of the original databases. The integration is done by providing or automatically generating a global unified schema that characterizes the underlying data sources. According to the mapping direction, the approaches are classified into two categories: global-as-view and local-as-view [13]. In global-as-view approach, each item in the global schema is defined as a view over the source schemas. In local-as-view approach, each item in source schema is defined as a view over the global schema. The local-as-view approach better supports a dynamic environment, where data sources can be added to the data integration system without the need to restructure the global schema.

On the other hand, semantic approach integration is obtained by sharing a common ontology among the data sources. Semantic approach addresses not the structuring of the architecture of the integration, but how to resolve semantic conflicts between heterogeneous data sources.

There are several well-known research projects and prototypes such as Garlic [2], Tsimmis [3], MedMaker [9], Mix [10], and DDXMI [11]. Garlic uses structural approach and takes a global-as-view mapping approach. Tsimmis and MedMaker also have structural approaches and take a global-as-view approach and uses common data model, e.g., OEM (Object Exchange Model). Mix uses XML as the data model, an XML query language XMAS was developed and used as the view definition language there. DDXMI (for Distributed Database XML Metadata Interface) is built on XML Metadata Interchange. DDXMI is a master file including database information, XML path information (a path for each node starting from the root), and semantic information about XML elements and attributes.

III. ANALYSIS OF PROBLEM

As described in the introduction, the task of a data integration system is to provide a uniform interface to a collection of data sources. Information systems are expected to be a completely new generation of software systems. Their main task is to operate at a global level over existing data sources. It is important to consider that these sources have characteristics making the integration process very difficult:

Heterogeneity:

The data sources are mostly developed for a special purpose. This often results in different solutions for storing information of the same real-world objects. Information can be stored in databases with different models (e.g. relational), or be available as Web Services.

It is difficult because these kinds of sources are accessed through different
• Interfaces,
• Protocols and
• Languages.

Even the information system built using same data model can cause mapping conflicts due to different understandings of the real world. To integrate or link the data stored in heterogeneous data sources, a critical problem includes entity matching, i.e., matching
records representing semantically corresponding entities in the real world, across the sources.

Data integration involves schematic conflicts and semantic conflicts: Schema integration is the process of merging autonomously developed DB schema into a unified, global schema to provide transparency through a unified view. Schema integration has been variously described as a 3, 4 or 5-step process [16] and involves the tasks of pre-integration (schema translation into common data model form), comparison (process of semantic conflict identification), conformance (making conflicts compatible for merging by similar representation), and merging (integrating schemas) including restructuring (refining schema)[16].

Semantic integration, resolves differences in conceptual representation of data by determining equivalence between schema constructs and removing ambiguity among component databases.

IV. SYSTEM DESIGN

A. System Design Modules:
Our System consists of two modules:

- Schema Integration module and
- Query Interface module.

Schema Integration:
In this module our system integrates two local source schemas to define Global Schema. Our mediator system is a data integration middleware that firstly receives two heterogeneous database names from user, retrieves database schema from respective local data sources. Each data source may contain one or more tables in it. Now each primary key of first data source is compared to each primary key of another data source on the basis of datatype, length and attribute name. If any of the primary key of one data source is found equal to any of the primary key of other data source on any of the given basis then question is asked to the user whether these fields are logically same if user confirms that those fields are logically same then two fields are merged and this process is repeated for the each attribute of table schema else those fields are not merged. In this way our system defines the global schema.

Query Interface:
This module aims to receive query, create xml documents and retrieve data from xml document. Our system receives XQuery from user applications based on global schema; this XQuery is analyzed syntactically as well as semantically. Syntactic analysis is done in accordance with XML grammar and semantic analysis in accordance with referred global schema view. After query is analyzed, an xml document is created which consist of data based on the domain of the query, and then query is executed on that xml document. To maintain freshness of data whenever the query from user is received new xml document is created which contains the updated information of the data source.

B. Architecture of the mediator

Fig 1 shows the architecture of the mediator which includes three sections

- Local DB Schema,
- Global Schema integration and
- Query interface section.

Local DB Schema:
This section consists of multiple local data sources schema that user wish to integrate. These data sources must be of Relational data model.

Global Schema integration:
This section defines the mediated schema of the integrated data sources which includes mapping schema from source element to global element.

One of the main problems arising in the data integration consists in carrying out the correspondence between a data source schema and the Global Schema. Generally, it is a question of laying down the rules which make it possible to bind the elements of the Schema of a source to those of the Global Schema (inter-Schema’s correspondence). This makes it
possible to the mediator to answer the queries of the user which are submitted on the Global Schema.

Query Interface:

After defining global schema, Mediator receives an XQuery formulated in terms of the unified Schema. The query is analyzed by XQuery engine and is addressed to specific data sources. This process creates xml document based on source descriptions by global Schema and mapping Schema, which play an important role in query execution plan optimization. Finally, the query is executed by the query execution engine on xml documents, retrieves data and present output to user in transparent way.

Fig 1 Architecture of Mediator

This mediator is the result of a detailed study of the advantages and disadvantages of several existing mediators. The implementation of the core is based on the management technology of the objects distributed around the two data models: the relational and the XML model.

V. IMPLEMENTATION

System aims user friendly mediation platform for the integration and provides user querying disparate heterogeneous information system. To implement both of the design modules we need two backend relational database servers and one frontend software application that can be connected to two or more backend database servers independently. For two backend database servers we selected two most popular and featured relational database servers

- MS-SQL Server and
- Oracle Database Server.

**MS-SQL Server:**

Microsoft SQL Server is a relational database management system developed by Microsoft. As a database, it is a software product whose primary function is to store and retrieve data as requested by other software applications, be it those on the same computer or those running on another computer across a network (including the Internet). Its primary query languages are T-SQL and ANSI SQL.

SQL Server 2008 aims to make data management self-tuning, self organizing, and self maintaining with the development of SQL Server Always On technologies, to provide near-zero downtime. SQL Server 2008 also includes support for structured and semi-structured data, including digital media formats for pictures, audio, video and other multimedia data. In current versions, such multimedia data can be stored as BLOBs (binary large objects), but they are generic bit streams. Intrinsic awareness of multimedia data will allow specialized functions to be performed on them. According to Paul Flessner, senior Vice President, Server Applications, Microsoft Corp., SQL Server 2008 can be a data storage backend for different varieties of data: XML, email, time/calendar, file, document, spatial, etc as well as perform search, query, analysis, sharing, and synchronization across all data types.

Other new data types include specialized date and time types and a spatial data type for location-dependent data. Better support for unstructured and semi-structured data is provided using the new FILESTREAM data type, which can be used to reference any file stored on the file system. Structured data and metadata about the file is stored in SQL Server database, whereas the unstructured component is stored
in the file system. Such files can be accessed both via Win32 file handling APIs as well as via SQL Server using T-SQL; doing the latter accesses the file data as a BLOB. Backing up and restoring the database backs up or restores the referenced files as well. SQL Server 2008 also natively supports hierarchical data, and includes T-SQL constructs to directly deal with them, without using recursive queries.

The version of SQL Server Management Studio included with SQL Server 2008 supports IntelliSense for SQL queries against a SQL Server 2008 Database Engine. SQL Server 2008 also makes the databases available via Windows Power Shell providers and management functionality available as Cmdlets, so that the server and all the running instances can be managed from Windows PowerShell.

In SQL Server 2008 we created a database named "dbo" with two tables “Personal” and “VoterDetails”.

Personal table contains personal information of the person with attributes FULLNAME (primary key), AGE, Residence_Address, Mobile_No, RelationShip, and EmailID.

VoterDetails table contains voting card information of the person with VotterID (primary key), FULLNAME (foreign key), Village, District, and State.

**Oracle Database Server:**

The Oracle RDBMS stores data logically in the form of tablespaces and physically in the form of data files ("datafiles"). Tablespaces can contain various types of memory segments, such as Data Segments, Index Segments, etc. Segments in turn comprise one or more extents. Extents comprise groups of contiguous data blocks. Data blocks forms the basic units of data storage.

In Oracle DB Server we created a database named “ORCL” with two tables “PERSONALDETAILS” and “DRIVING_LICENSE”.

PERSONALDETAILS table contains personal information of the person with attributes NAME (primary key), ADDRESS, PINCODE, BIRTH_DATE, AGE, CITY.

DRIVING_LICENSE table contains driving license information of the person with attributes LICENSE_NO (primary key), NAME (foreign key), LICENSE_TYPE, DATE_OF_ISSUE, EXPIRY_DATE, CITY, STATE.

For implementation of Schema Integration module and Query Engine module, the frontend software application we selected is Microsoft Visual Studio 2008.

**Microsoft Visual Studio 2008:**

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop console and graphical user interface applications along with Windows Forms or WPF applications, web sites, web applications, and web services in both native code together with managed code for all platforms supported by Microsoft Windows, Windows Mobile, Windows CE, .NET Framework, .NET Compact Framework and Microsoft Silver light.

Following are the components of Visual Studio that are used in our project:

**Microsoft Visual C#:**

Microsoft Visual C#, Microsoft’s implementation of the C# language, targets the .NET Framework, along with the language services that lets the Visual Studio IDE support C# projects. While the language services are a part of Visual Studio, the compiler is available separately as a part of the .NET Framework. The Visual C# 2008, 2010 and 2012 compilers support versions 3.0, 4.0 and 5.0 of the C# language specifications, respectively. Visual C# supports the Visual Studio Class designer, Forms designer, and Data designer among others.

**Microsoft Visual Web Developer:**

Microsoft Visual Web Developer is used to create web sites, web applications and web services using ASP.NET. Either C# or VB.NET languages can be
used. Visual Web Developer can use the Visual Studio Web Designer to graphically design web page layouts.

Using Visual C# language we made connection from front end application to backend MS-SQL Server using SqlConnection class instance and to Oracle database server using OleDbConnection class instance. Multiple DataSet instances and DataTable instances are created to store the retrieved data from local data source to frontend. Database names are taken as input from user in textboxes and are passed to SqlConnection and OleDbConnection objects respectively to create a connection with backend servers. Now schemas from both the databases are retrieved using respective sql queries. These retrieved schemas are stored in DataSet class instances. MS-SQL server consists of INFORMATION_SCHEMA table which contains information about the schemas of all databases available in it. Hence to retrieve schema of given input SQL server database name, query is fired on INFORMATION_SCHEMA table. Oracle database server consists of ALL_TAB_COLUMNS table which contains information about the schemas of all databases available in it. Hence to retrieve schema of given input Oracle server database name, query is fired on ALL_TAB_COLUMNS table.

After retrieving schema and storing it in DataSet, primary keys are compared from one data source to other data source on the basis of datatype, length, or attribute name if it matches, system asks user for confirmation of logical similarity of these two fields. If primary keys are logically similar then process is repeated for all attributes of both tables and two tables are integrated to form one table. Message Boxes are used to ask the questions from user and based on these answers given by user the global schema is defined. Tree nodes are used to show the schema of integration of two data sources. Once global schema is defined, user can interface with integrated schema using XQuery.

We selected XML as a global platform because it is the best candidate for supporting the integrated data model. Whenever XQuery is given as input xml document is created for storing integrated data and allow user to query xml documents in XQuery syntax.

XML:
Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. It is defined in the XML 1.0 Specification produced by the W3C, and several other related specifications all gratis open standards.

The design goals of XML emphasize simplicity, generality, and usability over the Internet It is a textual data format with strong support via Unicode for the languages of the world. Although the design of XML focuses on documents, it is widely used for the representation of arbitrary data structures, for example in web services.

XQUERY:
XQuery is a query and functional programming language that is designed to query collections of XML data. XQuery 1.0 was developed by the XML Query working group of the W3C.

The mission of the XML Query project is to provide flexible query facilities to extract data from real and virtual documents on the World Wide Web, therefore finally providing the needed interaction between the Web world and the database world. Ultimately, collections of XML files will be accessed like databases.

Now whenever user gives the input in XQuery format, data is retrieved from the local data sources in datasets and dataset’s WriteXml() method is used to create new xml document and store data in xml data form. Then XQueryNavigatorCollection class adds the respective xml documents based on XQuery and XQueryExpression’s Execute () method is used to analyze and execute the XQuery. The output is then
given to gridview object to display it to users on web page.

In implementing Schema integration module and Query Execution module the platform we used is Windows XP O.S.

VI. CONCLUSION AND FUTURE WORK

A. Conclusion

Our system satisfies almost all requirements for a mediator allowing an efficient integration of heterogeneous information systems. Besides the integration of different kinds of data sources it offers now a more flexible way of extending the system.

B. Future Work

There are still works need to be done in future. The solution for data integration problem that we provided in this dissertation is primarily optimized for the case of two databases.

However, this can also be extended towards multiple databases. Data model integrated here are two instances of Relational databases, it can also be extended to integrate two Object Oriented Data Sources, or Object Oriented data source to Relational data source.

REFERENCES


