International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622
International Conference On Industrial Automation And Computing (ICIAC- 12th & 13th April 2014)

RESEARCH ARTICLE OPEN ACCESS

Creating Materialized View over the Integration of Heterogeneous Databases

Ankush J. Paturde*, Anil V. Deorankar**
*(Department of Computer Science and Engineering, Government College of Engineering, Amravati-444604
Email: ankushpaturde@gmail.com)
** (Department of Computer Science and Engineering, Government College of Engineering, Amravati-444604
Email: avdeorankar@gmail.com)

ABSTRACT
With the development of computer network and database, traditional database has been increasing unable to meet the needs of data sharing and interoperability. Meanwhile, it is impossible to abandon all the existing database systems; therefore, the research of simultaneously accessing and processing data from a number of databases has become an inevitable trend. For the Health care information system its not the issue to retrieve the information from their own databases. But when we want the information other than the own databases, then its an issue to get that information to our system. And the data which we want from other health care organizations may not be in same format. To solve this problem the proposed architecture is to integrate different geographically dispersed databases that are heterogeneous with regard to their logical schemas.For the Integration of heterogeneous databases MyAccess , MySQL, SQL and Oracle databases are taken. These databases are having different characteristics of data types and semantic conflictions may occure while integrating heterogeneous databases. Using java technology, XML, SQL Language, heterogeneous databases integration system is proposed and designed and key technologies are also described in detail. For the queries which are frequently fired to the databases for retrieving the information, the materialized view is created. By creating materialized view the information which requires most to the user is stored in the dataware house. Because of which response to the frequent queries is fast, no need to search the the whole database. It reduces the overhead for the frequent queries and increases response time.

Keywords **–** Data Integration, data transformation, Heterogeneous databases, Java Technology, SQL query, Wamp server, XML

I. INTRODUCTION

In E-Business, distributed information needs to be integrated or interoperated to provide the unique view of information from disparate sources. When we design the relevant databases dependently, it does not create heterogeneity problems to integrate the data sources. However, when databases have been designed independently, there are heterogeneity problems such as different terminology, data types, units of measurement, domains, scopes, and so on. The structures of these heterogeneous databases are different obviously, and semantic conflictions may occur. Data integration shields the heterogeneity of the various heterogeneous data sources, and carries out unified operation to different data sources through heterogeneous data integration system. The data forms involved in heterogeneous database are mainly structured data, semi-structured data and unstructured data three types. Structured data widely exists in a variety of information system database, the most common relational database. Semi-structured data commonly has Web pages as the chief representative. Unstructured data has common files, email and various documents. A practical information integration system should have intelligence, openness and initiative. Intelligence is to carry out unified processing, filtering, reduction, abstraction, integration and induction works for the structured, semi-structured and unstructured data from different databases. Openness is a heterogeneous and distributed database, which must solve the mismatching problem of the information expression with the structure. Initiative is to regulate the existing Internet data representation, exchange and service mechanism to provide proactive service mechanism. Then creating materialized view of the frequent queries. Materialized views are constructed and stored in a data warehouse with the purpose of improving response time of analytical queries. Since all possible views cannot be materialized, as it would violate the storage constraint, the aim is to construct those materialized views that maximize the profit in terms of their ability to answer future user queries.Proposed work is to integrate databases to provide a unified representation, storage and data management for various heterogeneous data environment, which is the basic function the heterogeneous data integration system must implement. Most queries posed by a user on a data warehouse are likely to be domain specific. Accordingly, it is more appropriate to construct

materialized views. Creating Materialized view will increase the likelihood of being able to answer the user query on its own. introduction of the paper should explain the nature of the problem, previous work, purpose, and the contribution of the paper. The contents of each section may be provided to understand easily about the paper.

II. SYSTEM ARCHITECTURE

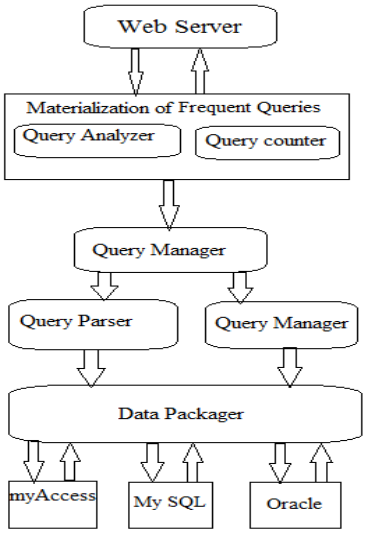

fig 2.1 architecture of integration of databases with materialized view.

2.1 Creating DSN of various relational databases

The DSN(Data Source Name) of various relational databases Can be created for the data integration such as –

- MyAccess
- MySQL
- Oracle
- SQL Server

2.2 The Key technologies of database integration system

Various technologies can be used for the database integration, which are as follows

2.2.1 JavaBean Technology

JavaBean is a software component model to describe Java. In the Java model, the functions of the Java program can be infinitely expanded by JavaBean, and new applications can be rapidly generated through the JavaBean combination. JavaBean also can achieve code reuse, while has very great significance for the program maintenance. Through the Java virtual machine JavaBean can be run correctly. JavaBean provides for the Java component-based development system. And the query manager and data packager in this system are all the JavaBean components based on the Java language.

2.2.2 Connection pool

Connection pool is a kind of entity which manages the connection as a resource, and a typical example of such resource is the database connection. The basic idea of the connection pool is to pre-establish some connections to store in the memory for use. To establish a database connection will consume considerable system resources, but once established, the query can be sent to obtain results through it.

2.3 Implimentation Design

After creating the databases we have to integrate them for global visualization of the dispersed databases. The integrating data is based on horizontal integration where the tuples of different databases are joined. As we have integrated the heterogeneous databases, we can retrieve the information from that integrated data. Now we are applying Data materialization technique on this integrated data. Because of this technique the time require to access the most frequent information will be very less and is useful for the decision making.

2.4 Creating Domain For Materialized View

A data warehouse contains data from multiple disparate data sources and hence may cater to various domains. Most queries posed by a user on a data warehouse are likely to be domain specific. Accordingly, it is more appropriate to construct materialized views with respect to individual domains. This will increase the likelihood of a materialized view being able to answer the user query on its own. One way to create such domains is by grouping closely related queries, from the previously posed queries on the data warehouse, into clusters. Each of these clusters will represent a domain.

2.4 Finding Similarity Between Two Clusters

The similarity between two clusters is computed using Jaccard's coefficient. The similarity is defined as the proportion of number of relations in common accessed by queries in the two clusters. Suppose Qi1, Qi2 . . . , Qim are queries in cluster Ci and Ri1, Ri2, . . . Rir are relations accessed by queries Qi1, Qi2, . . . Qim and Qj1, Qj2, . . . Qjn are queries in cluster Cj and Rj1, Rj2, . . . Rjs are relations accessed by queries Qj1, Qj2, . . . Qjn then the degree of similarity SIM(Ci, Cj) between two clusters Ci and Cj can be defined as

$$\text{SIM}\left(C_i, C_j\right) = \frac{|\{R_{i1}, R_{i2}, \ldots R_{ir}\} \cap \{R_{j1}, R_{j2}, \ldots R_{js}\}|}{|\{R_{i1}, R_{i2}, \ldots R_{ir}\} \cap \{R_{j1}, R_{j2}, \ldots R_{js}\}|} = \frac{|\text{Rel}(C_i) \cap \text{Rel}(C_j)|}{|\text{Rel}(C_i) \cup \text{Rel}(C_j)|}$$

where Rel(Ci) and Rel(Cj) are relations in cluster Ci and Cj respectively.

The clusters can be merged only when the degree of similarity between them is greater than zero i.e.

SIM(Ci, Cj) > 0 for all clusters Ci ≠ Cj

Suppose,

Q1

SELECT EmpName, DeptName FROM Employee, WorksIn, Dept

WHERE Employee.EmpId=WorksIn.EmpId AND WorksIn.DeptId=Department.DeptId

Q2

SELECT StudName, CourseTitle FROM Student, Opted, Course, Taught
WHERE Student.StudId=Opted.StudId AND Opted.CourseId=Course.CourseId AND
Course.CourseId=Taught.CourseId AND NoOfHours = 40

Q3

SELECT EmpName, HireDate FROM Employee, WorksIn
WHERE Employee.EmpId=Works_In.EmpId AND DeptId=20

A similarity matrix is constructed with similarity between two clusters computed using Jaccard's Coefficient as given above. The similarity matrix for the above queries, is shown in Table 1.

TABLE I: SIMILARITY MATRIX FOR QUERIES Q1,Q2 AND Q3

SIM	Q1	Q2	Q3
Q1	1	0	0.66
Q2	0	1	0
Q3	0.66	0	1

The clusters are merged in an iterative manner using the similarity matrix where, at each step, the clusters having maximum similarity are merged first. The approach uses the queries posed in the past to create clusters of closely related queries. Each such cluster specifies a domain. The approach then identifies frequent queries from amongst the queries in each domain. These frequent queries reflect the information that has been accessed frequently in the past. Thus, the materialized views constructed using these frequent queries are likely to answer most of the future queries. As a result, the query response time would be reduced. This in turn would facilitate the decision making process.

III. CONCLUSIONS

In this paper the proposed work is to integrate the disperate information from the various databases. It is useful to retrieve the information from the heterogeneous databases just like the information of students of various departments, information about the health care system from different organizations, etc. And creating the materialized view of the frequent queries. Due to which for the frequent queries we don't have to search the different databases. Firstly materialized data will be search and then after goes for the different databases to search. Because of which we get the fast response to the frequent queries taking very short time.

REFERENCES

[1] BRITO, M. S. et al. An Architecture for Integrating Databases with Replication Support Based on the OGSA-DAI Middleware. 12th IEEE International Conference on Computational Science and Engineering

[2] Wu Xiaoli,Yao Yuan "XML-based Heterogeneous Database Integration System Design and Implementation". Department of Computer Science and Engineering, Henan University of Urban Construction IEEE 2010.

[3] ZHANG Zhen–you WANG Hongong–hui 'Research of Heterogeneous Database Integration Based on XML and JAVA Technology' International Conference on E-Learning, E-Business, Enterprise Information Systems, and E-Government 2009.

[4] C. H. Goh, S. E. Madnick, & M. D. Siegal, "Context inter-change: overcoming the challenges of large-scale interoperable database systems in a dynamic environment", in Proc. Inf. and Knowledge Management, MD USA,1994.

[5] R. D. Holowczak, & W. S. Li, "A survey on attribute correspondence and heterogeneity metadata representation", nstitute of Electrical & Electronics Engineers.

[6] K. Abdulla, "A new approach to the integration of heterogeneous databases and information systems", Dissertation, University of Miami, Florida, 1998.

[7] Chaiyaporn Chirathamjaree. 'A Data Model for Heterogeneous Data Sources' EngineeringSchool of Computer & Information Science, Edith Cowan University, Australia

[8] W. Kim, I. Choi, I. Gala, & M. Scheevel, "On resolving schematic heterogeneity in multidatabase systems". Distributed and Parallel Database, vol.1, no.3, pp.251-279, 1993.

[9] R. J. Miller, "Using schematically heterogeneous structures", SIGMOD'98, WA, USA, pp.189-200, 1998.

[10] P. Missier, M. Rusinkiewicz, & W. Jin, "Multidatabase languages", Management of heterogeneous and autonomous database systems. CA: Morgan Kaufmann Publishers, 1999.

[11] Wu Xiaoli and Yao Yuan "XML-based Heterogeneous Database Integration System Design and Implementation" Department of Computer Science and Engineering, Henan University of Urban Construction 2011.

[12] Mingli Wu and Yebai Li "Investigations on XML-based Data Exchange between Heterogeneous Databases". Ninth Web Information Systems and Applications Conference 2012.