Middleware for Heterogeneous Database Schema

Chetan H. Bhave[1], Gajanan K. Naik[2],Gauravkumar V. Tore[3],Chetan A. Patil[4]

U.G. Students
Computer Engineering Department
SSBT's COET, Bambhori, Jalgaon, India

***Abstract*—The heterogeneous databases are useful to have access to global information. Due to integration of various applications, heterogeneous databases need to interact with each other. The communication must also satisfy the transactional requirements. Since the vendors of heterogeneous databases are different, it results in different schemas. Such databases never interact with each other without any middleman. This interface must provide all transactions related services. There are few existing solution but they have limited transactional capabilities. As there is no common interface to communicate with heterogeneous databases, centralization issue arises. Simultaneous operations are not performed on such system resulting in performance loss and increase in response time. In our proposed solution we are developing database middleware. The Database Middleware is way to have common interaction mechanism. This type of middleware allows the heterogeneous database having different schemas to communicate with each other.**

***IndexTerms*— Database; Heterogeneous; Middleware; Schemas.**

.

I. INTRODUCTION

The development of Information Systems, network communications and the World Wide Web, has permitted access to autonomous, distributed and heterogeneous data sources[1]. An increasing number of databases, especially those published on the Web, are becoming available to external users. User requests are converted to queries over several sources with different information quality. Integration of schemas on existing databases into a global unified schema is an approach developed over 20 years ago. However information quality cannot be guaranteed after integration, because quality is dependent on the design of the information and its provenance[2]. Even greater levels of inconsistency exist when information is retrieved from different information sources. This calls for an efficient solution which can support this notion of Information System and at the same time maintain information quality.

Multidatabase systems provide integrated global access to autonomous,heterogeneous local databases via a simple global request. A central activity required for processing aglobal request is to resolve the logical heterogeneity as the result of the local autonomy of multidatabases namely, schema integration and data integration[1]. Schema integration resolves schematic heterogeneity such as differences in attribute name and domain and differences in data format and structure. Data integration on the other hand, has to solve the following problems: entity identificationidentify object instances in different databases that model the same real world entities[1]. Missing or inconsistent data some data items may be recorded in one database but not in others or several databases record the same data item but give it different values.

Heterogeneous database communication is always been an issue[1].There is lot of development in the era of operating system but operating system does not provide an service that will facilitates the communication between heterogeneous applications.The vendors of such databases are different so schema of such databases is different[3].So for this purpose database middleware is come into the reality.

A heterogeneous database system is an automated or semi-automated system for the integration of heterogeneous, disparate database management systems to present a user with a single, unified query interface[3].Heterogeneous database systems (HDBs) are computational models and software implementations that provide heterogeneous database integration[1]. The integrated, local DBSs are autonomous and can also be used as stand-alone systems .System having heterogeneous bases allows the data to be store at one center .This data is actually collected from different centers[4] . Heterogeneous system usually result when individual sites have implemented their own database and integration is considered at a later stage.

Main Objective Of this Project is to develop middleware for Heterogeneous Database Schema with following objectives-

- To develop middleware which will handle the transactions across heterogeneous databases.
- To develop middleware which will handle concurrent access to databases.
- To develop middleware which will hide heterogeneity of the database.

The Middleware for heterogeneous database schemehas been proposed in this paper with its architecture, result, technical discussion and conclusion.

II. ARCHITECTURE

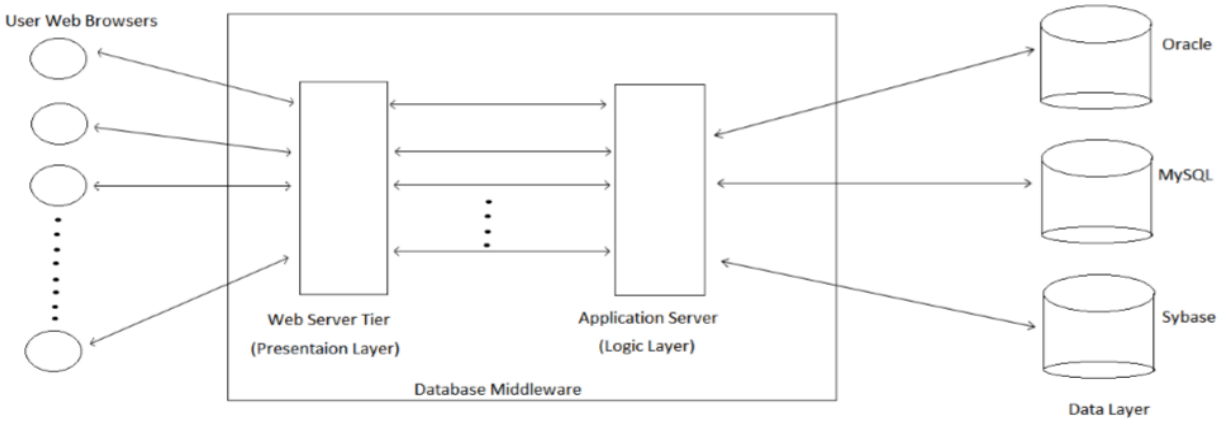

Figure:1Architecture of Proposed Solution

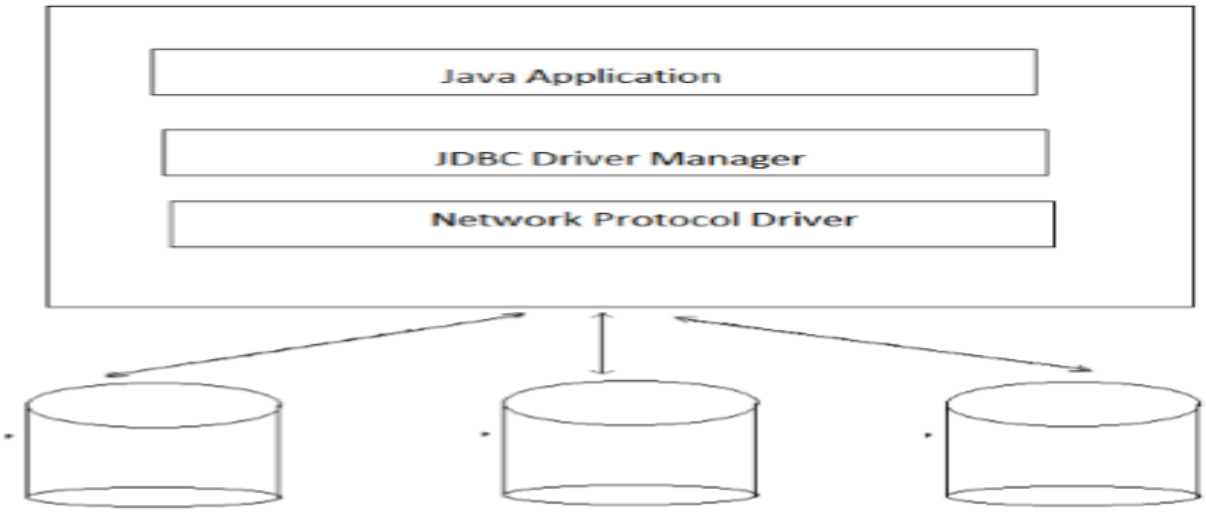

Figure:2 Architecture of Proposed Solution

In our proposed solution Database Middleware is divided into two Layers or Tiers. One is presentation layer and another one is logic layer.When this middleware is a part of the N-tier architecture then we can work on a multiple databases[5].

When multiple clients are trying to access or request a web server at a same time then web server creates a separate thread for each request.Then, these requests are forwarded to logic layer. Logic layer is connected with three or more different databases.In our architecture we are using MySQL,Oracle, Sybase databases. Logic layer performs certain operations like load balancing,Concurrency control,Heterogeneous database connectivity and satisfaction of ACID properties.

In our proposed solution we are using Tomcat as a web server which creates an environment for the development and execution of a java program.

Java application program creates an environment with a Java Database Connectivity Driver Manager with the help of JDBC API(Application Program Interface) to form a Network driver or 3-Type driver.JDBC API needs a driver to connect with a multiple databases.JDBC Driver Manager is a connectivity software which forms the communication between Java Application program and Multiple databases.Network driver or Type-3 driver resides on a application server which converts JDBC calls into driver based calls.

III. RESULT AND DISCUSSION

Due to integration of various applications, heterogeneous databases need to interact witheach other. The communication must also satisfy all the transactional requirements. Sincethe vendors of heterogeneous databases are different, it results in different schemas. Suchdatabases never interact with each other without any middleman. This interface must provide all transactions related services.

So, with our proposed architecture applications can easily communicate between heterogeneous databases globally and provide global transaction. So to prove this, consider anexample in banking context where we would have to transfer account from one country toanother country. So, consider there is one bank customer which live in India and have an account in India. Customer has to transfer account to UK. In our proposed solution we have maintain information of customers into two databases i.e. Oracle and MySQL. Oracle contains information of customers from USA, Russia, India as shown in figure 3and MySQL maintained information of customers from UK, Japan as shown in figure 5.

Figure 7 contains authentication information tableof users such as account number, password,country name before transfer. figure 3 shows oracle database table which contains information aboutAccount Number, Country, Current Balance, Currency. As shown in figure 3 and figure7 ,customer with Account Number '123010' and country 'India' has to transfer account to 'UK' i.e. fromOracle database to MySQL database.

'UK' customer information is maintained in MySQL and Account Number '123010' ispresent in Oracle database. So we have to transfer all the information of customer withAccount Number '123010' from Oracle to MySQL.

After transferring account, entry from Oracle database is deleted and newtable is obtained as shown in figure 5. Also, new entry is created in MySQL database table with unique account number '321010' and balance is converted from 'RS' into 'Pound'. Also information from login figure 7 is changed such as account number and country name and new table is obtained as shown in figure 8.

As in many cases, When lots of multiple users simultaneously login into system. They fired queries on main databases due to this applications frequently needs to interact with databases and large significant amount of time goes waste on doing low priority tasks. But the main task is to provide transaction functionality with high performance speed acrossglobally. So, to solve above problem we proposed an architecture which maintains one localdatabase that contained all the authentication information of particular domain and otherdatabases which contains transactional data. So due to separate and distinct local databaseauthentication related queries spanned with local database and main databases do otherhigh priority transactional tasks. Due to this performance of system significantly increasedand therefore system load is balanced in some amount.

ACCNO	COUNTRY	CURBAL	CURRENCY	OLDCOUNTRY
123010	India	42486.5294	RS	UK
456001	USA	774.3858	DOLLAR	Null
456002	USA	10000.0000	DOLLAR	Null
456003	USA	14963.0000	DOLLAR	Null
456004	USA	21479.0000	DOLLAR	Null
456005	USA	1100.0000	DOLLAR	Null
789001	RUSSIA	1848.3969	RUBLE	Null
789002	RUSSIA	754.0000	RUBLE	Null
789003	RUSSIA	5472.0000	RUBLE	Null
789004	RUSSIA	964.0000	RUBLE	Null
789005	RUSSIA	6314.0000	RUBLE	Null
123002	India	12982.8156	RS	Null
123003	India	3000.0000	RS	Null
123004	India	1278.3000	RS	Null
123005	India	1234.9000	RS	Null

Figure 3 Oracle Database Customer Information before Account Transfer

ACCNO	COUNTRY	CURBAL	CURRENCY	OLDCOUNTRY
456001	USA	774.3858	DOLLAR	Null
456002	USA	10000.0000	DOLLAR	Null
456003	USA	14963.0000	DOLLAR	Null
456004	USA	21479.0000	DOLLAR	Null
456005	USA	1100.0000	DOLLAR	Null
789001	RUSSIA	1848.3969	RUBLE	Null
789002	RUSSIA	754.0000	RUBLE	Null
789003	RUSSIA	5472.0000	RUBLE	Null
789004	RUSSIA	964.0000	RUBLE	Null
789005	RUSSIA	6314.0000	RUBLE	Null
123002	India	12982.8156	RS	Null
123003	India	3000.0000	RS	Null
123004	India	1278.3000	RS	Null
123005	India	1234.9000	RS	Null

Figure 4 New Oracle Database Customer information after Account Transfer

ACCNO	COUNTRY	CURBAL	CURRENCY	OLDCOUNTRY
321001	UK	148966.2779	Pound sterling	Null
321002	UK	21469.0000	Pound sterling	Null
321003	UK	9631.0000	Pound sterling	Null
321004	UK	1500.0000	Pound sterling	Null
321005	UK	900.0000	Pound sterling	Null
654001	Japan	45978.0000	Yen	Null
654002	Japan	124789.0000	Yen	Null
654003	Japan	167618.0660	Yen	Null
654004	Japan	455810.0000	Yen	Null
654006	Japan	303467.0628	Yen	India

Figure 5 MySQL Database Customer Information Table beforeAccount Transfer

ACCNO	COUNTRY	CURBAL	CURRENCY	OLDCOUNTRY
321001	UK	148966.2779	Pound sterling	Null
321002	UK	21469.0000	Pound sterling	Null
321003	UK	9631.0000	Pound sterling	Null
321004	UK	1500.0000	Pound sterling	Null
321005	UK	900.0000	Pound sterling	Null
321010	UK	453.1864	Pound sterling	India
654001	Japan	45978.0000	Yen	Null
654002	Japan	124789.0000	Yen	Null
654003	Japan	167618.0660	Yen	Null
654004	Japan	455810.0000	Yen	Null
654006	Japan	303467.0628	Yen	India

Figure 6 MySQL Database Customer Information Table after Account Transfer

ACCNO	PASSWORD	COUNTRY
123010	@@india	India
123002	adminuser	India
123003	walkalone	India
123004	Password	India
123005	user123	India
456001	GeorgeSoman	USA
456002	amol234	USA
456003	Alkatai	USA
456004	Yogesh	USA
456005	abhijit	USA
789001	@chandan	Russia
789002	gajanan123	Russia
789003	@Gaurav	Russia
789004	Maskov	Russia
789005	@Dimitri	Russia
321001	kapil	UK

Figure 7 User Login Information database table before Account Transfer

ACCNO	PASSWORD	COUNTRY
321010	@@india	UK
123002	adminuser	India
123003	walkalone	India
123004	Password	India
123005	user123	India
456001	GeorgeSoman	USA
456002	amol234	USA
456003	Alkatai	USA
456004	Yogesh	USA
456005	abhijit	USA
789001	@chandan	Russia
789002	gajanan123	Russia
789003	@Gaurav	Russia
789004	Maskov	Russia
789005	@Dimitri	Russia
321001	kapil	UK

Figure 8 User Login Information database table after\ Account Transfer

IV. CONCLUSION AND FUTURE SCOPE

The proposed solution resolved conflicts between incompatible data sources. As for different sources there are different vendors. So there is no common interface to communicate with heterogeneous databases, centralization issue arises.

In this paper, we proposed an architecture which resolves centralization issues and also provide load balancing on main data sources which are mainly used for transaction.The solution also provides transaction between heterogeneous databases which are useful to access the global information.

In this paper, we have presented an architecture for database middleware which was formally proved to ensure Transaction guarantees in the general context of systems with multiple autonomous back-end databases. Compared to already existing solutions coping with this scenario, our proposal has the distinguishing feature of this type of middleware allows the heterogeneous database having different schemas to communicate with each other.

The solution have features like handling ACID property in heterogeneous environment, handle Global and Local transactions and resultant consistent data can simplify decision making.

This project can be extended to enable to automatically extract data from RFID tags and sensors which can be use for different purpose such as vehicles for traffic control system. Also, by embedding Artificial Intelligence (AI), it can be used for automatic and dynamic decision making.

V. REFERENCES

[1] JM. Smith et al., "MULTI DATABASE" Integrating Heterogeneous Distributed Database Systems" Proceedings of the 1981 National Computer Conference, Reston, VA: AFIPS Press, pp. 487-499.

[2] YI ZHANG XIN WANG. "Solution for data inconsistency and data integration" Technical report, JOURNAL OF INFORMATION SCIENCE AND ENGINEERING.

[3] Jens H. JahnkeJorg P. Wadsack. "Towards model-driven middleware maintenance" JOURNAL OF INFORMATION SCIENCE AND ENGINEERING.

[4] Stefan Tai Christoph Liebig. Middleware mediated transaction. Springer, 2004.

[5] W.D. Potter and L. Kerschberg, "A Unified Approach to Modeling Knowledge and Data" IFIP WG 2.6 Working Conference on Knowledge and Data, September 1986.