EasyChair Preprint

№ 1916

Data Exchange in Heterogeneous Databases in KSA Using Cloud Computing

Noof Awad Aldieef and Nabeel Khan

November 11, 2019

DATA EXCHANGE IN HETEROGENEOUS DATABASES IN KSA USING CLOUD COMPUTING

Noof Aldieef
Information Technology
Qassim University
Qassim, Saudi Arabia
411213660@qu.edu.sa

Dr.NabeelKhan
AssiatantProfessor
QassimUniversity
Qassim, Saudi Arabia
n.khan@qu.edu.sa

***Abstract*—Most institutions in Saudi Arabia, of all types, commercial, medical, social and government agencies, have cloud computing technology. Thus, there is a large amount of heterogeneous data available, which has become a significant obstacle to the query process and direct access to information. Therefore, heterogeneous data processing has become a new challenge for Saudi Arabia in computational computing. We know that different data processing in local environments is not an easy process, as there are difficulties in traditional database management systems (DBMS), as the situation is more complicated, more complex, in cloud computing environments. Many companies in the IT industry are interested in addressing this situation, producing many technologies to deal with it. This paper aims to develop a methodology for standardized data access in heterogeneous data systems. This allows users to query and interact with different structural databases. And make the most of the enormous amount of information stored. The main objective of the paper is to help users as well as developers to access data, in different databases, as a single database, easily, saving time, effort and cost, with a single click.**

***Index Terms*—Heterogeneous, DBMS ,Databases,Cloud.**

I. INTRODUCTION

The integration of existing information systems is becoming increasingly imperative to meet customer and business needs. Restoring public data from different database systems is a difficult task and has become a widely discussed topic. In many cases, we need information simultaneously from various database systems separated from each other ”such as querying patient information from separate hospital information systems or computerized manufacturing management. “In these cases, users need to homogeneous logical views of data physically distributed across heterogeneous data sources. These views should represent the information collected as if the data were stored in a standardized manner. A significant trend in cloud computing and data management is the understanding that there is ”no one size fits all “solution. Thus, there has been a blooming of different cloud data management infrastructures, referred to as NoSQL [1], specialized for different kinds of data and tasks and able to perform orders of magnitude better than traditional relational DBMS. Examples of new data management technologies include: graph databases (e.g. Sparksee [2], Neo4j), key-value data stores (e.g. HBase, Cassandra, Hyper Table), array data stores (e.g. SciDB), analytical cloud databases (e.g. Greenplum and Vertica), analytical cloud frameworks (e.g. Hadoop Map-Reduce, Cloudera Impala), document databases (e.g. MongoDB, CouchBase), and data stream management systems (e.g. Stream Cloud [3,4], Storm). This has resulted in a rich offering of services that can be used to build cloud data-intensive applications that can scale and exhibit high performance. However, this has also led to a wide diversification of DBMS interfaces and the loss of a common programming paradigm. This makes it very hard for a user to integrate her data sitting in specialized data stores, e.g. relational, documents and graph databases. For example, consider a user who, given a relational data store with authors, a document store with reviews, and a graph database with author friendships, wants to find out about conflicts of interests in the reviewing of some papers. The main solution today would be to write a program (e.g. in Java) that accesses the three data stores through their APIs and integrates the data (in memory). There are many solutions to integrate data from separate relational systems into a new system. However, the problem is further complicated if the source database systems are heterogeneous, that is, they implement different types of data models (such as the data model for different non-relational data models). While all relational database management systems rely on the popular relational data model, NoSQL systems apply different models of semi-structured data (column stores, master value stores, document databases, and graph databases) [3,5]. When data models determine how to modify the logical structure of a database, they play an important role in accessing the data. The aim of this paper is to propose a general solution for simultaneously relational and non-relational database systems. The

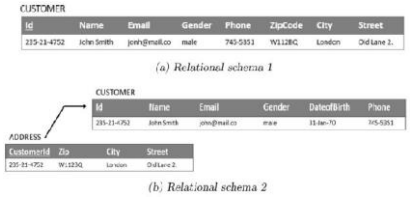
contribution of this work is to introduce a new method of data integration and workflow, which implements data integration for different source systems in a way that users do not need any programming skills. The main element of the proposed method is the JSON intermediary data model.The new cloud services are offered to companies ranging from virtual server creation to high-performance storage systems, providing high-speed, high-quality connectivity, public and private network interfaces, and cloud support services. The Kingdom has also launched the e-Government Program (Yesser) of the Ministry of Communications and Information Technology. The program will establish a communication network for government e-transactions, which connects government agencies to the e-government data center "Yesser", to be used to host the national e-government portal. This network enables the e-Government Center to serve as a link between government. entities, so that the mechanism of linkage between entities is unified and its cost is reduced [11].

II. Saudi Arabia and cloud computing

Saudi-based Virtual Vision, a technology and technology development company, has launched e-cloud services in the Kingdom of Saudi Arabia, in collaboration with Cloud Sigma and Hewlett-Packard Enterprise. The cloud services provide the opportunity for Virtual Vision to expand its diversified portfolio of services and businesses in various sectors in the Kingdom and the region in general, while contributing to the continuous development of strengthening the infrastructure in the ICT sector in the country, in addition to achieving the objectives of the Saudi Vision 2030. The new cloud services are offered to companies ranging from virtual server creation to high-performance storage systems, providing high-speed, high-quality connectivity, public and private network interfaces, and cloud support services. The Kingdom has also launched the e-Government Program (Yesser) of the Ministry of Communications and Information Technology. The program will establish a communication network for government e-transactions, which connects government agencies to the e-government data center "Yesser", to be used to host the national e-government portal. This network enables the e-Government Center to serve as a link between government. entities, so that the mechanism of linkage between entities is unified and its cost is reduced [11].

III. Problems of data integration

While retrieving information from heterogeneous source systems, we must meet three main challenges: (1) Solving data heterogeneity (2) structural solution (data model) data heterogeneity (3) Bridging differences in data query syntax. Although structural heterogeneity can be considered a special kind of semantic heterogeneity, Semantic heterogeneity means differences in the meaning and interpretation of context from the same field. The semantic variation in the scheme level arises from synonyms and character symmetries [6]. A synonym problem occurs when the same real-world entity is named differently in different databses, and the symmetry problem occurs when different real-world objects (such as entities and attributes) have the same name in different databases [6]. These problems can be resolved by defining unique and clear attribute names, which can be created manually or semi-automatically using ontology. Structural heterogeneity (heterogeneity of the data model) arises from different modeling approaches and from the use of different data models. On the one hand, in a particular type of data model, real world objects can be designed in different ways. Due to different database design techniques and methods, database designers may create different schemes for the same data, and it may be difficult to reconcile these models. Schema Matching attempts to identify relevant attributes and entities in different schemas and define mapping rules between these schemas. For example, Figure1 - shows different presentations that have the same attributes in the relevant databases. Both relational models have almost the same qualities (with some semantic homogeneity) but in the first model, all attributes are grouped into one relationship, and in the second case, the attributes are divided into two relationships according to their meanings. Both relational models are true but the schemes applied are different. In addition, the second relational model also has an additional attribute (DateofBirth), which is not part of the first model [7].



(a) Relational schema 1

(b) Relational schema 2

Fig. 1. Different representations of the same real-world objects in relational models.

```
{
    "_id"       : ObjectID("542c2b97bac0595474108b48")
    "Name"      : "John␣Smith",
    "Email"     : "john@mail.co",
    "Gender"    : "male",
    "Birthdate" : "31-Jan-70"
    "Phone"     : "745-5351",
    "Address"   : {
                    "Zip"       : "W112BQ",
                    "City"      : "London",
                    "Street"    : "Old␣Lane␣2."
                  }
}
```

Fig. 2. Code1. Example for document-oriented data model: people collection.

The example above in JSON is shown in code 1. As we can

see, the structure and width of this model are very different from the relational models presented and may also contain semantic variations. However, data integration is more than a structural or semantic problem. Heterogeneity of respects must be so. Technically, it is fairly easy to connect different relational database networks (for example, via ODBC or JDBC), than connecting related systems to NoSQL databases. This problem arises from heterogeneity of data access methods. On the one hand, SQL is a standard programming language, designed to manage data stored in relational database management systems[8].

Although there are subtle differences between different dialects of SQL languages, these differences are not important. On the other hand, because NoSQL systems rely on different data models, they implement different access methods that are incompatible with the SQL language. Although NoSQL systems sometimes assert that they may support SQL-like query languages, these extensions are not equal to the standard SQL version. Following the example above, if we want to query the name and city of customers according to the three schemas, we have to formulate this query differently: As we can see, the first two queries are based on the standard SQL language and these queries differ from each other only because the schemas applied are different. However, the comparison between the first database queries and the third data access method, the sub-difference between them is obvious. Based on this homogeneity, data integrity can be observed as a difficult task. My goal is to support data integration by hiding the specific details and heterogeneities of the various source systems. The proposed solution described in the next section is based on a meta model approach, in a sense that the specific interfaces and schemas of the source systems are mapped into a common one [8].

IV. METHODOLOGY

As discussed earlier, the motivation is to access and execute operations on different SQL and NoSQL systems without knowing about them in a single application. We'll detail about the properties of this interface in this proposal. The main objective is how to design and access data rather than the scalability or productivity performance of NoSQL systems. The structure of the CAP window can be seen in Figure 3. MySQL, Mongodb, Cassandra and Neo4j were selected for this suggestion. Each designer framework will have input from the user through the GUI in the form of a query option and a SQL database to work on. Then, the SQL query should be mentioned in a way that specifies specific information about the segments the user wants to work on, the parameter that the user wants to target and performs the operation. The user is expected to have a basic knowledge of the SQL query [9]. The choose database option determines the process of selecting the database to perform. The various subsections of architecture can be explained as follows: 1.SQL query parser: which helps in parsing specially SQL queries. which helps in parsing specially SQL queries. the parser also verifies weather the query is syntactically and semantically proper.

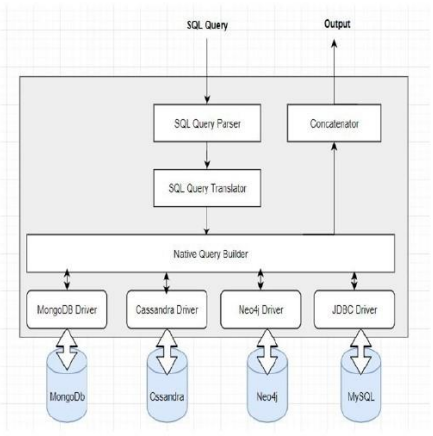


Fig. 3. System Architecture.

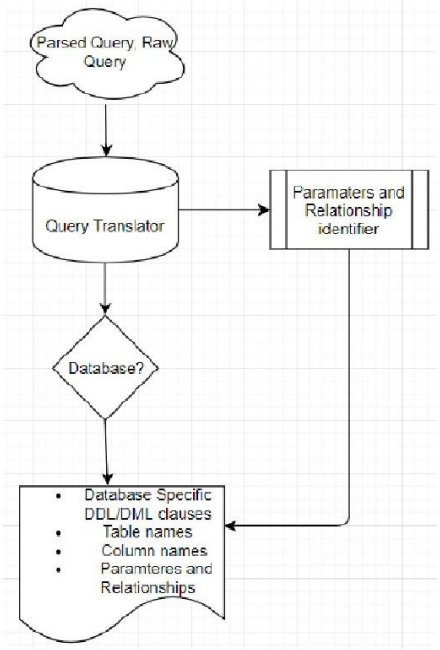


Fig. 4. Query Translator.

The parser breaks down SQL query into following important parts: -DDL/DML Clauses. -Table names. - Column names. -Parameters. -Attributes.

2 Query translator: the query translator accepts input in the form of String and syntax tree. One of the important tasks of query Translator is to find relationship between the attributes and values. Also, it prepares all set of parameters for native

query builders to write database specific queries, as seen in Figure 4.

Native query Builder:the mission of Native Query Builders is to develop specific queries for the original database and interact with the relevant database drivers to execute them. Metadata is obtained from different database drivers that help Native Query Builders obtain information about database / group / column family names, read / write formats, existing attributes, and so on. An example can be seen of sentences that are the originators of the original query: the statement describes a DDL statement. Sentences that are currently considered are of the following types: Select, Insert, Delete, and Update. The statement is important in order to understand the process that will run. Sentences for each database are different and can be presented as a tabular format [10]. 4.Concatenate the result :the interface helps fetch information from multiple databases at once. The structure of the connected databases is different and it is important to present the result uniformly. This is achieved by the Concatenate score category as seen in Figure 5. [10].

Clauses	MongoDb	Cassandra	Neo4j
select	db.find()	SELECT keyspace.tablename	MATCH nodes
insert	db.CollectionName. insertOne()	INSERT into keyspace.tablename	CREATE node
update	db.CollectionName. updateMany({ }, { $set: { } })	ALTER columnName FROM keyspace.tablename	MATCH node and SET node
delete	db.people.drop()	DELETE [column_name (term)] FROM keyspace_name.table_name	MATCH node and DELETE node

Fig. 5. tabular format

V. PROPOSED SOLUTION

To develop a web application,using one of two languages: ASP.NET or PHP. The main components of a web application are:

- GUI application: to enter the query. The front end of the application consists of two pages: - First of a model with two parts, the query part and the base selection part. In the query pane the user has to enter a SQL query and in the database selection pane, the user, select one or more databases (Databases available). available). This can be seen in Figure 6.

Fig. 6. Application Home page .

This can be seen in Figure 5. page consists of a form used to enter a SQL query and a list of available databases. The user can choose one or more databases. Both inputs are very crucial for the system. Input is received by the frame, the operation is performed and returned to: front end, for another page namely the results page.

- Back-end system: the back-end system consists of request servlets, parsers, translators and general parser.

VI. EVALUATION

The project is tested in multiple scenarios by changing queries and database options. Access to standardized results from multiple databases. This scenario tests with several databases, and the success of the query process, for different types, of databases. Through the proposed query form as seen in Figure 7 and 8 [8].

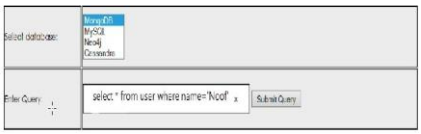


Fig. 7. Query Page.

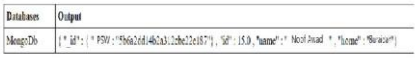


Fig. 8. Output Page.

VII. CONTRIBUTION

In this work, I proposed a unified interface for heterogeneous NoSQL databases. Under these papers, the research was theoretical, needed a practical application, set up a project, monitor the results, and increase the number of databases used. Finally, integration can be made with the cloud platform to create a unified query engine serving all government and private sectors in the Kingdom.

VIII. CONCLUSION AND FUTURE WORK

In this work, I proposed a unified interface for heterogeneous NoSQL databases. Under these papers, the research was theoretical, needed a practical application, set up a project, monitor the results, and increase the number of databases used. Finally, integration can be made with the cloud platform to create a unified query engine serving all government and private sectors in the Kingdom.

REFERENCES

[1] NoSQL Databases, http://nosql-database.org. [Last accessed on August 18, 2015].

[2] Martínez-Bazan, N., Muntés-Mulero,V.,Gómez-Villamor, S., Águila-Llorente, M.A., Dom´ınguez-Sal, D., Larriba-Pey, J-L.: EfficientGraph Management Based onBitmap Indices. Int.DatabaseEngineering Applications Symposium (IDEAS), pp. 110-119 (2012).

[3] Gulisano, V., Jiménez-Peris, R., Patiño-Martinez, M., Valduriez, P.: StreamCloud: A Large Scale Data Streaming System. IEEE Int. Conf. on Distributed Computing Systems (ICDCS), pp. 126-137 (2010).

[4] Gulisano, V., Jiménez-Peris, R., Patiño-Martinez, M., Soriente, C., Valduriez, P.: StreamCloud: An Elastic and Scalable Data Streaming System. IEEE Trans. On Parallel and Distributed Systems 23(12), 2351–2365 (2012).

[5] Godfrey, P., Gryz, J., Hoppe, A., Ma, W., Zuzarte, C.: Query rewrites with views for XML in DB2. IEEE Int. Conf. on Data Engineering, pp. 1339–1350 (2009).

[6] M. Ceruti , M.N. Kamel , Semantic heterogeneity in database and data dictionary integration for command and control systems, Technical Report, DTIC Docu- ment, 1994.

[7] Db-engines ranking, (http://db-engines.com/en/ranking). Accessed: 2016-08- 04. View publication.
.

[8] Agnes Vathy-Fogarassy, VeszprémUniform data access platform for SQL and NoSQL database systems,University of Pannonia Article in Information Systems · May 2017

[9] Atzeni, P., Bugiotti, F. and Rossi, L. (2012). Uniform access to non-relational database systems: The sos platform, International Conference on Advanced Information Systems Engineering, Springer, pp. 160 ,174, Core Rank A.

[10] SCHREINER, Geomar A.; DUARTE, Denio; DOS SANTOS MELLO, Ronaldo. Sqltokeynosql:a layer for relational to key-based nosql database mapping. In: Proceedings of the 17th International Conference on Information Integration and Web-based Applications Services.

[11] Secure Government Network (GSN). https://www.yesser.gov.sa/EN.