International Journal of Engineering Research in Computer Science and Engineering (IJERCSE) Vol 3, Issue 4, April 2016

Survey of various Query Mapping Techniques from SQL to No SQL

[1] Piyush Nikam [2] Tushar Patil, [3] Gayatri Hungund, [4] Ankit Pagar, [5] Aditi Talegaonkar,
[5] Ms. Soudamini Pawar
[1][2][3][4][5] Department of Computer Engineering, Akurdi, Pune.
Savitribai Phule Pune University, Pune

Abstract: **- Dynamic assessment and handling of big data has become tedious task nowadays. Database scalability is a burning issue, when it comes to usage of traditional Relational Database Management System (R DBMS), the scalability becomes a constraint after a certain data limit is reached. In order to deal with this problem parallel databases was proposed as a solution which was later replaced by a more effective and scalable solution known as No SQL database. Dynamic data handling ability of No SQL databases made it easy to store various types of data. Moreover, No SQL databases proved to be more scalable than the existing relational databases. The paper provides a survey of various query mapping techniques from SQL to No SQL. It also provides a comparative study of various mapping techniques based on various factors such as approach used, time complexity and databases used.**

Keywords **- SQL, No SQL, and Map Reduce, Hive, Migrate, Map.**

I. INTRODUCTION

The traditional relational databases which store data in structured format in contrast to which the No SQL databases store the data dynamically The data handling capacity of the No SQL databases is more than that of the relational databases which proves to be a major advantage to store Big Data. The No SQL databases store the data in different forms like Key-value pair, Graphs, Document oriented, Column based. No SQL database of type key-value stores data in the form of key value pair. Key for every tuple is unique. Example of the key-value databases are: Oracle No SQL Database, redis, Couch DB. The No SQL databases of type Document oriented encapsulates data in document form and encodes data in some standard format. The encoding of data is done with help of XML and JSON. The examples of Document stored databases are Mongo DB, Apache Couch DB, Qizx. The Graph based No SQL databases represents the whole data in the form of graph. This data can be road maps, network topologies, public transport links. Examples of Graph based databases are: Allegro, Neo4J, Infinite Graph, Orient DB, Virtuoso, Stardog. The column based type of No SQL databases the data is grouped into columns rather than rows of data. These different columns are grouped logically into column families. The reading and writing of data is done through columns
rather than rows. Examples of such type of databases are: Accumulate, Cassandra, Druid, HBase, Vertica.

A. NEED OF SWITCHING FROM SQL TO NO SQL

1. Scaling:

In case of the relational databases as size of data increases the vendor has to buy costly servers store huge amount of data. In contrast, use of No SQL databases store huge amount of data by scaling vertically and horizontally, thus, reducing the cost required to buy large data servers to store such large amount of data.

2. Handling large amount of data:

R DBMS provides the increase in the volume for data storage but this increase is insufficient for some enterprises. But Hadoop provides storage of data in the distributed form with thousands of nodes which handle large amount of data efficiently.

3. Cheaper maintenance cost of No SQL servers:

The maintenance of R DBMS data server is more expensive and requires the constant assistance. But No SQL data server requires less assistance over the server and is cheaper than that of the R DBMS.

4. No fixed data model:

The R DBMS has to follow the rigid structure to store data in the databases. The No SQL databases do not follow any rigid structure to store data so we can store different types of data in the No SQL databases.

5. Caching facility:

To increase the data retrieval speed and efficiency of system the No SQL databases provides the cache in the memory. But to use the cache in R DBMS the use of different infrastructure is required.

B. SIGNIFICANCE OF MAPPING

The relational databases have limited data handling capacity and store structured data. No SQL databases on the other hand have dynamic data handling capacity and can store data in different formats. Due to this major advantage, many organizations are switching to No SQL. Also, if the current database management system of an organization is outdated and there is a need to shift to latest databases, then people having knowledge of new database technology are needed to handle the newly adopted database management system which may lead to financial crisis. Thus the query entered in SQL if mapped to corresponding No SQL format will benefit the organization by reducing the learning curve of user.

II. GENERAL MAPPING TECHNIQUES

The different mapping techniques that can be used to map data from Relational databases to No SQL are as follows:

1. The SQL query entered is converted into parse tree and maps this tree to the interface of No SQL system.

2. After establishing connection with MySQL we have to select the No SQL database to which data is to be migrated. We can create embedded document which contains information about two or more tables in a single document. Text files are generated using this above information and converted into the format which is accepted by Pentaho data integration tool which generates collection in Mongo DB.

3. Q Map per
   a. Entered SQL query is converted into query graph model.
   b. Query rewriter applies S-H and H-H rules to query generated model.
   c. Query compiler translates this query generated model into MapReduce directed acyclic graph.
   d. Plan evaluator is used to select the best one from these DAG's.

4. Data can be migrated using java hibernate from SQL to No SQL. Data mapping module using object relational mapping maps the fired SQL query to the appropriate No SQL query and the data is fetched from No SQL database which increases the scalability.

III. MAPPING TECHNIQUES

Wu Chung Chung et al. [1] have proposed a framework Jackhare which uses J DBC for connecting to databases. The scanner, parser and code generator are implemented using Druid. Apache derby is used as metadata server that stores table name, column name which can be mapped to table and column name of relational database.

The query is checked for the existence of WHERE clause after which the query is checked for the condition specified by the where clause for the name of rows and input tables that are to be used. The last step is to output the data fields in the SELECT statement. The logical flow is converted to Hadoop Map Reduce jobs which consists of Map phase that produces keyvalue pairs and the Reducer phase combines the output in ordered list according to keys.
Leonardo Rocha et al. [2] have proposed a No SQL layer which contains a mapping module that lies between the application and databases. The user enters a query in SQL format and data is retrieved from the No SQL databases. MySQL proxy is used as it uses client server protocol for communication between different MySQL servers. The query is parsed in the form of xml file which has two attributes: The class of web service used and method to be executed. The query parameter is checked to get information about SQL table or database and also check for additional where clauses.

The query is checked for its type that is if it is one of type SELECT, INSERT, etc. and corresponding method is called. The mapping is done according to the type of statements. Each statement has separate java code which maps the SQL query to No SQL format using Mongo DB and json Java packages. Objects of DB Collection and DB Object are created and finally JSON. parse method is called. The result is stored and converted to string using toString() method. The mapped query is executed on Mongo DB and results are sent to mediator which forwards them to application.

The relational database system faced a slowdown in speed when data storage reached upto 2GB. Therefore Nawal N. Moatassem [6] proposed a solution by conducted experiment which involved the databases to pass through CSV files before migration to No SQL database. Thirteen tables were migrated by open connection to SQL and Mongo DB one table is read at a time. CSV file contained the table

in which each line represent a row and was imported to Mongo DB using following command: mongoimport -d database -c table -type csv -headerline -file /tmp/tab.csv

Julian Rith et al. [3] have proposed a middleware to map SQL queries to No SQL format. The queries are first divided into two categories before mapping. First category includes CRUD operation statements and second contains DDL statements. The queries containing like and where clause are also categorized under category two. The middleware is developed in C# on the basis of .Net framework. The SQL queries are directly provided as input to framework which processes a query and generates intermediate representation of query using standard parse tree. After which, the parse tree is mapped to corresponding No SQL query using connectors. If the mapping is unsuccessful the error message is displayed. The parser uses another tool for language recognition which is powerful parser generator for translating structured text. ANTLR generates a parser that can build parse tree.

Rupali Arora et al. [4] have proposed an algorithm to transfer from R DBMS to Mongo DB. Firstly, connection is established between databases and user is allowed to select the database from existing relational databases and database is created in Mongo DB. From the selected list of tables the table whose embedded documents are created in Mongo DB are chosen. Columns are selected and then Pentoho tool is used for loading data in Mongo DB and gone corresponding rules in it.

Xu et al. [7] proposed a tool in which query rewriting equivalent HiveQL queries are generated with increase in probability of optimization of mapping rules. It can also be used with MapReduce level optimizers. It can generate different versions of HiveQL that can reduce identical results. These methods to create different versions vary and leads to different performance. To select the best method it uses cost estimation.

Cost estimation is based on data between mapper and reducer or in network. Hive optimizer will look for the chances to merge jobs. QMapper estimator is used to estimate the cost of MapReduce DAGs. A MapReduce DAGs consist of MapReduce job as a node and the directed edge between jobs indicate the data flow. Finally it measures the intermediate data between Maps and Reduces. The method with minimum cost will be selected.

TABLE 1:
Comparison between various mapping techniques

Paper name	Mapping technique	Technology	Key features
JackHare: a framework for SQL to NoSQL translation using MapReduce [1]	• The compiler scans and parses the ANSI-SQL query. • Lookup the related table name, column families and column qualifier of HBase • Generate MapReduce code • According to the query commands and metadata. • Access HBase and execute the MapReduce job.	• MySQL • MongoDB • HBase • Hive • Java • XML • Druid	• Without framework o Hive: 700gb in 10000 sec. o MySQL: 700gb in 5000 sec. • With framework o JackHare: 700gb in 4000 sec.
Speaking in Tongues: SQL access to NoSQL Systems [3]	• Queries are first divided into two categories (CRUD & DDL) before mapping. • SQL queries are directly provided as input to framework • Framework generates intermediate representation of query using standard parse tree. • The parse tree is mapped to corresponding NoSQL query using connectors	• MongoDB • Cassandra • SQL • C# • .NET	• Without framework o RDBMS: 700-800 operations per sec. o MongoDB: 1200 operations per sec. o Cassandra: 600 operations per sec. • With framework o RDBMS: 700-750 operations per sec. o MongoDB: 950 operations per sec. o Cassandra: 500 operations per sec.
QMapper: A Tool for SQL Optimization on Hive Using Query Rewriting [7]	• Entered SQL query is converted into query graph model. • Query rewriter applies S-H and H-H rules to query generated model. • Query compiler translates this query generated model into MapReduce directed acyclic graph. • Plan evaluator is used to select the best one from these DAGs.	• Hadoop • SQL • HiveQL	• Without framework o HiveQL: 1100 sec per query. • With YSmart: o 1100 sec per query o Optimized query: 650 sec per query.
A Frame-work for Migrating Relational Datasets to NoSQL [2]	• MySQL proxy is used • The query is parsed in the form of xml file • The query is checked for its type accordingly the type and corresponding method is called. • Each statement has separate java code which maps the SQL query to NoSQL format using MongoDB and jsonJava packages.	• MongoDB • SQL • JSQL parser • XML • JSON	• Runtime of NoSQL is less than that of MySQL • For larger volume of data NoSQL layer is efficient than MySQL
Integration and Virtualization of Relational SQL and NoSQL Systems including MySQL and MongoDB [16]	• Parsing step is a parse tree representing the SQL query. • The query translator takes a parse tree as input and translates it into a relational operator tree. • System optionally generates schema to represent data in NoSQL	• SQL • MongoDB	• Join combines a table with 1500 records. • Time to read base relation: • MongoDB: 0.9 sec. • MySQL : 1.5 sec.

Ramon Lawrence et. al. have built a virtual architecture for translation of SQL queries into source specific APIs which consists of SQL query parser that generates a parse tree from the SQL query. The generated parse tree is converted to operator tree which consists of various operators such as selection, projection, join and grouping. While converting the parse tree to operator tree field and table names are checked from relational schema. After query translation No SQL schema is generated. For Mongo DB, schema is generated by sampling data in each collection and creating most general relational model that covers data [16]. After the schema generation, it is stored in Mongo DB collection to avoid regeneration of schema.

IV. DISCUSSION ABOUT VARIOUS MAPPING TECHNIQUES

HBase is used as a flexible data store not only to store data on large storage system but also manipulate the data in a transparent way. In this framework the SQL queries is broken down into tokens which can then be matched with corresponding No SQL queries [1].

This paper proposes No SQL framework which provides two services which are data migration and data mapping. Data mapping module creates equivalent collection in No SQL database corresponding to database in SQL. Data mapping module uses mediator which maps the SQL queries to No SQL queries and fetches data from No SQL database [2].

To optimize the data placement this paper introduces a method called correlation method on Sqoop. This is done by grouping the related data to minimize the data transformation cost. It considers table correlation along with size for data locality and data efficiency. It focuses on frequently used operations such as JOIN. The design is such that it analyzes the tables which are frequently used for this operation and distributes these tables on the same processing node [5].

This paper identifies a set of SQL queries to access No SQL system. It translates SQL queries to connected No SQL database queries. Cassandra and Mongo DB are the databases used. The mapping framework is developed in C# on the basis of .NET framework [3].

V. CONCLUSION

As we have seen the limited size handling capacity of the relational database scalability is a major constraint. Apart from this using few commercial licensed relational databases is costly. On the contrary No SQL databases provide support for storing unstructured data with the capability to handle dynamic data and ability to scale data horizontally on commodity hardware. To get these advantages of No SQL databases using ANSI SQL queries we have discussed various mapping techniques which help us in retrieving data in real time from No SQL database using ANSI SQL queries. The mapping process reduces the learning curve and helps user to perform operations on No SQL databases using ANSI SQL Queries thus making it economically feasible. Few mapping techniques even help reducing data retrieval time by performing some optimizing operations.

REFERENCES

[1] Chung, W. C., Lin, H. P., Chen, S. C., Jiang, M. F., & Chung, Y. C. (2014). JackHare: a framework for SQL to No SQL translation using MapReduce. *Automated Software Engineering*, *21*(4), 489-508.

[2] Rocha, L., Vale, F., Cirilo, E., Barbosa, D., & Mourão, F. (2015). A Framework for Migrating Relational Datasets to No SQL. *Procedia Computer Science*, *51*, 2593-2602.

[3] Rith, J., Lehmayr, P. S., & Meyer-Wegener, K. (2014, March). Speaking in tongues: SQL access to No SQL systems. In *Proceedings of the 29th Annual ACM Symposium on Applied Computing* (pp. 855-857). ACM.

[4] Arora, R., & Aggarwal, R. R. (2013). An Algorithm for Transformation of Data from MySQL to No SQL (Mongo DB). *International Journal of Advanced Studies in Computer Science and Engineering (IJASCSE)*, *2*(1).

[5] Hsu, J. C., Hsu, C. H., Chen, S. C., & Chung, Y. C. (2014, July). Correlation Aware Technique for SQL to No SQL Transformation. In *Ubi-Media Computing and Workshops (UMEDIA), 2014 7th International Conference on* (pp. 43-46). IEEE.

[6] Moatassem, N. N. (2015). *A Study of Migrating Biological Data from Relational Databases to No SQL Databases* (Doctoral dissertation, Youngstown State University).

[7] Xu, Y., & Hu, S. (2013, May). Qmapper: a tool for sql optimization on hive using query rewriting. In *Proceedings of the 22nd international conference on World Wide Web companion* (pp. 211-212). International World Wide Web Conferences Steering Committee.

[8] Codd, E. F. (1970). A relational model of data for large shared data banks.*Communications of the ACM*, *13*(6), 377-387.

[9] Lee, R., Luo, T., Huai, Y., Wang, F., He, Y., & Zhang, X. (2011, June). Ysmart: Yet another sql-to-mapreduce translator. In *Distributed Computing Systems (ICDCS), 2011 31st International Conference on* (pp. 25-36). IEEE

[10] Manoj, V. (2014). Comparative study of No SQL Document, Column Store Databases and Evaluation Of Cassandra. *International Journal of Database Management Systems (IJDMS)*, *6*(4), 11-26.

[11] Kumar, R., Gupta, N., Charu, S., Bansal, S., & Yadav, K. (2014). Comparison of SQL with HiveQL. *International Journal for Research in Technological Studies*, *1*(9), 2348-1439.

[12] Choi, Y. L., & Yoon, S. H. (2014). Improving Database System Performance by Applying No SQL. *Journal of information processing systems*, *10*(3), 355-364.

[13] Cattell, R. (2011). Scalable SQL and No SQL data stores. *ACM SIGMOD Record*, *39*(4), 12-27.

[14] Schram, A., & Anderson, K. M. (2012, October). MySQL to No SQL: data modeling challenges in supporting scalability. In *Proceedings of the 3rd annual conference on*

Systems, programming, and applications: software for humanity(pp. 191-202). ACM.

[15] Roijackers, J., & Fletcher, G. H. L. (2012). Bridging sql and No SQL. *Master's thesis, Eindhoven University of Technology*.

[16] Lawrence, R. (2014, March). Integration and virtualization of relational SQL and No SQL systems including MySQL and Mongo DB. In Computational Science and Computational Intelligence (CSCI), 2014 International Conference on (Vol. 1, pp. 285-290). IEEE