A Study for Integrating SQL and NoSQL Databases

Krishnapriya VM[1], Libin Sebastian[2], Gibin George[3]

[1]Pursuing master's degree program in Computer Science in Santhigiri College Of Computer Science, Thodupuzha, Kerala, India
mca2022_krishnapriyavm@santhigiricollege.com

[2]Pursuing master's degree program inComputer Science in Santhigiri College Of Computer Science, Thodupuzha, Kerala, India
mca2022_libinsebastian@santhigiricollege.com

[3]*Assisstant Professor in Santhigiri College Of Computer Science, Thodupuzha, Kerala, India*
george.gibin@santhigiricollege.com

Abstract: ***In our day to day life we come across several data. Some will be confidential and some will not. Therefore, it's necessary to have a storage mechanism. When it is the storage of bulk data we come to the term database. Storage of data can be relational or non-relational. The storage of data in a relational format is called relational database or SQL database. Whereas, storing data not based on a perfect schema can be called NoSQL database. The former follows data storage in the form of tables or relation and the later follows non tabular format, i.e. will be based on type of data model. Now just think about a case where we want to store or retrieve data from both databases together. This can be done only through integration. But integration of both the databases following different schema is a complicated task where it is possible one too. That's what we present through this paper. In this paper we present certain integration mechanisms like Unity, Data Adapter, Novel Integration Data Methodology, JackHare framework using MapReduce and DualFetchQL. These approaches mentioned help us to integrate the both databases. Each mechanism has its own features and objectives which will be discussed in detail in this paper.***

Keywords: SQL, NoSQL, Databases, Integration.

1. Introduction

Integration of already existing information systems is becoming more inevitable in order to satisfy customer and business needs. Data retrieval from different database systems is a challenging task and has become a widely researched topic[1,2]. In most cases, we need data or information simultaneously from different database systems which are separated from each other (e.g. query patient information from separated hospital information systems or manage computer integrated manufacturing). Such cases, users need homogeneous logical views of data physically distributed over heterogeneous data sources. These views have to represent the collected information as if data were stored in a similar way. The basic problem originates from the fact, that origins were built separately and the need of information linking arose later. If the source system were designed and built irrespectively of each other, then the created database schemas and semantics of data may be significantly different. The main tasksof general data access are (i) to identify those data elements in source that match each other and (ii) to formulate such global database access methods that can maintain the differences of the source systems. To solve the problem we need an intermediatesolution to retrieve data from heterogeneous source systems and to deliver them to the destination. There exist a lot of solutions which aim to integrate data from separated relational database systems into a new one. However the problem is further complicated ,if the database are heterogeneous, namely they implement different categories of data models (e.g relation data model vsnon-relational data models). While every SQL database management systems are based on the well-known relational data model, NoSQL systems implement different semi-structured data models (column stores, key- value stores, document databases, graph databases)[3]As data models define how the logical structure of a database is modeled, they play a significant role in data access. The aim of this paper is to suggest a generalized solution to query relational and non-relational database systems simultaneously. The contribution of this work is to present a novel data integration method and workflow, which implement data integration of different source systems in the way that users do not need any programming skills.

2. Literature Review

Data base integration is the process of combining data from different databases into a single, unified view. This process that is ,database integration starts with the ingestion process, and includes steps such as cleansing, ETL mapping, and transformation. Data integration process enables analytics tools to produce effective, actionable intelligence. here is no universal approach to data integration. However, data integration solutions typically involve a some similar objects, including a network of data sources, a server, and clients accessing data from the server. In a typical data integration process, the client send a request to the server for data. The server then intakes the needed data from internal and external sources. The data is extracted from the sources database, then consolidated into a single, cohesive data set. This is sent back to the client for use. Therefore through this paper we present before some techniques or methods for integrating databases. Integration of data helps in time saving, proper management of data and so on. So let's see what are the methods which we can apply to integrate databases.

3. Structured Query Language(SQL)

SQL (Structured Query Language)[4] is used to store and handle data maintained in a relational database management system (RDBMS) or to process streams in a relational data stream management system (RDSMS). It is used for handling data in the form of structured data(that is table format).SQL offers two main advantages over older read write APIs like ISAM or VSAM. Firstly, it introduced the process of accessing many records with one single command. Secondly, it reduces the need to specify how to reach a record, example with or without an index. SQL have different types of statements, which may be informally classed as sublanguages that is, (DQL) data query language,[a] (DDL) data definition language ,[b] (DCL)data control language , and (DML)data manipulation language .SQL's scope involves the processing of data queries, data manipulation (insertion, updating and deletion), data description (creation and modification of schematics), and data access control.. Although itis a declarative language (4GL), SQL also includes procedural elements.SQLwas the first commercial languages to utilize Edgar F. Codd's relational model. The model was described in 1970 paper named as, "A Relational Model of Data for Large Shared Data Banks".[5] Despite not entirely adhering to the relational database model as invented by Codd, it became the most widely and commonly used database query languageThe standardized query language became a standard for the ANSI (American National Standards Institute) in 1986 and for the International Standardization Organization (ISO) in 1987.[11] Since then, to include the standard, a wider set of characteristics has been revised. Most SQL code needs some modifications, until being ported to separate database systems, in spite of the existence of requirements.

4.NoSQL

NoSQL("non-SQL" or "non-relational")[6] database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases(SQL). The databases are existed since the late 1960s,before that we use file system, but the name "NoSQL" was only coined in the early 21st century, triggered by the needs of Web 2.0 companies.NoSQL databases are used in big data and real-time web applications development .NoSQL systems are also sometimes called "Not only SQL" to emphasize that they may supports SQL-like query languages. Motivations for this method include simplicity of process design, easier horizontal scaling of system clusters (which is an issue for relational databases), better accessibility control and limiting the mismatch of relational impedance items. The data structures used by NoSQL databases (e.g. key–value pair, graph, or document) are different from those used by default in SQL or relational databases, making some operations faster in NoSQL. Sometimes the data structures used inNoSQL databases are also 'more flexible' than SQL database.SeveralNoSQL stores compromise consistency in of the CAP theorem, in favor of availability, and speed [7].The challenges to NoSQL database adoption include the use of a low-level query language instead of SQL query, such as a lack of standardized interfaces and major previous investments in existing relational databases. Additionally, some NoSQL database systems may exhibit lost writes and other forms of data loss. There are few different methods for storing data. One is key value, it is a simple form basic data structure is a dictionary or map. We can store a value such as integer, string or anythingalong with a key used to refer that value. Second format is graph format, that uses graph structures for semantic queries with nodes, edges and properties to represent and store data. It uses a collection data and nodes. Collections are the next model. The NoSQLdatabase do not have a specific schema in the same rigid way that relationaldatabase(SQL) have a schema. Each of the four main type of has an underlining structure that is used to store the data. The another model is document. This gives the logic for a data model in each case document database store data in the document data type which is similar to a JSON document or object. Last one is fields.

4. Different Methods Of Integration

Though it is a little hard to integrate these two databases, there exist some methods that are as follows:

- Unity
- Data Adapter
- Novel Integration Data Methodology
- JackHare framework using MapReduce
- DualFetchQL

5.1 Unity

The contribution of this work is a generalizable SQL query interface for both relational and NoSQL systems called Unity[8]. Unity allows SQL queries to be translated and executed using the API of the databases (NoSQL or SQL). Unity is a method for integration and virtualization it allows SQL queries that span multiple sourcesdatabses and uses its internal query engine to perform joins across sourcesdatabses. The key features of Unity are:

- A processor and optimizer for SQL queries that supports push-down filters and joins cross-source hash.
- A customisable SQL dialect translator that uses mappings in relational structures to compensate for various SQL variants.
- SQL to NoSQL translation that maps relational operators to NoSQL programming APIs in a query plan.
- SQL function translation allows functions to be mapped between different system implementation (e.g. different order of parameters, different names).
- Data virtualization allowing queries to join or grouping across both relational and NoSQL databases.

The Unity system architecture consists of a SQL query parser that converts the SQL query into a parsing tree and validates the query. The query translator converts the parse tree into a relational operator the tree consisting of operations like selection, projection, grouping, and join. The query optimizer determines join ordering and portions of the query plan to execute on each data source. The execution system interacts with the data sources to submit queries and retrieve results and then perform any additional operations.

5.2 Data Adapter

The main features of the data adapter are listed as follows[9].

1. SQL Interface to RDB and NoSQL Database. To access both RDB and NoSQL databases, we provide a general SQL interface. It consists of a SQL query parser and Apache Phoenix to connect HBase as a NoSQL database to a SQL translator and a MySQL JDBC driver to an RDB connector. The application does not need to change the queries or manage NoSQL queries with this SQL interface, and may remain the original framework design for both MySQL and

HBase access..

2. DB Converter. We are developing a database converter with a table synchronization mechanism to deal with database transformation. Using Apache Sqoop and Apache Phoenix bulk load software, the database converter converts data from MySQL to HBase.The synchronization mechanism synchronizes data after completing transformation for each MySQL table by patch- ing the blocked queries while transformation.

Query Approach. Three query approach modes are proposed: blocking transform mode (BT mode), blocking dump mode (BD mode), and direct access mode (DA mode). Each mode provides various policies that allow apps to access the RDB.

The data adapter system contains mainly two parts as shown in Fig. 1 DB Adapter and DB Converter. It is the duty of the DB Adapter to communicate with programs, two databases, and the DB converter.DB Converter controls the transformation of data from a relational database to HBase

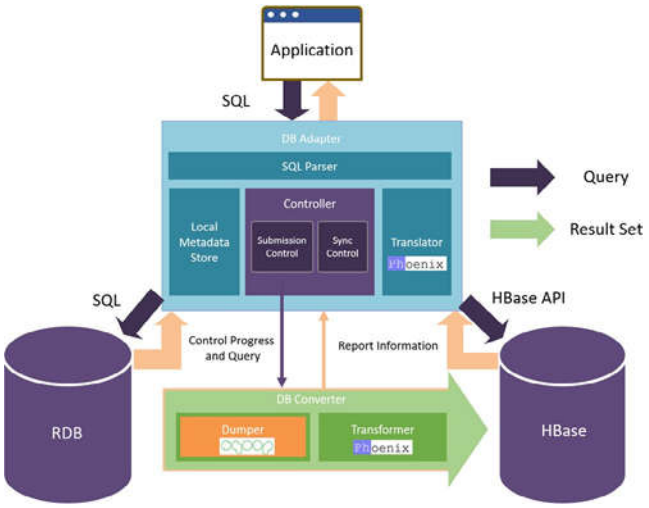

Figure 1: Data Adapter Architecture

and synchronization of uncertain tables. We describe the design and implementation of each component as follows.

Apache HBase is a portable, Hadoop framework-based NoSQL database. Tables in HBase are quite different from ones in MySQL. To solve this problem, Phoenix is used to create tables similar to MySQL tables. Row, key, column family and column qualifier of HBase are handle by Phoenix, too.

Apache Phoenix is a SQL translator for HBase. It allows users to access HBase using SQL commands. Phoenix accesses HBase with a coprocessor instead of creating MapReduce jobs, which allows query results returned faster. However, the value of rowkey and name of column family must be generated specifically when creating tables by Phoenix.The value of rowkey is the value of column of primary key and name of column family is ‘‘_0’’. We need to convert data according to the requirements, otherwise, Phoenix cannot access any data in HBase.

DB adapter can be designed to link different databases as data source. In this paper, it is designed to support MySQL and HBase. MySQL JDBC driver is used to connect with MySQL while Phoenix provides client and server jar files used to connect with HBase. We perform SQL queries through translator and it handles SQL statement translation. When users need differentNoSQL database instead of HBase, it is necessary to find a proper SQL translator for data adapter. In addition, we need to build new data converter methods to migrate data from RDB to NoSQL databases.

SQL parser is an interface which accepts queries from applications, parses queries, extracts and sends necessary information to controller. Parser can tell the difference between read and write queries and pass the information to controller to put write queries, which might be affected by transformation progresses, in a queue if necessary.

The Controller controls table transformation development, query flow, and table synchronization according to proposed modes of query approach. Queries that insert, remove or update operations on a table that is converted to HBase are placed by the controller in a queueData in tables cannot be altered for various techniques in particular phases. In controllers, submission control and sync control are two elements.Submission monitors not only converter communications, but also tracks the progress of transformation in local metadata stores. A table is called as a transformation unit. Order of table transformation by converter is also decided by submission control. Sync control is responsible for carrying out the process of synchronization after the transformation of each table.The SQLite database is used to record all relevant details, such as the transformation status of the table, and the controller and DB converter keep this database updated.

There are two components to the DB converter: the dumper and the transformer.Sqoop is used as the dumper for the transformer process to export data from RDB to CSV format files.First, Sqoop selects a table set and divides data into splits, including partial data files. Each partition is managed by a mapper..Sqoop does not block tablesbecause of performing operations using MapReduce. Applications submits queries to access tables which is involved in data transformation process. In transformer phase of DB converter, Phoenix Bulk Load is used to load CSV files and convert data into HFiles for HBaseviaMapReduce operation. The reason to use Phoenix Bulk Load is as Phoenix is a SQL translator in this system. Phoenix needs specific information while accessing tables in HBase. Phoenix Bulk Load not only coverts data into HBase tables but also generates information required by Phoenix. In this system, transformer and translator are considered as a set of components. The data format used in this system such as data type and schema mapping must be compatible with both transformer and translator. Otherwise, translator cannot understand the output result of transformer.

5.3 Novel Integration Data Methodology

To cover the structural and se- mantic heterogeneities of source data we suggest to set up general data models which describe main characteristics and relationships of source systems. These data models are formulated in JSON[10]. Heterogeneities of different access methods are resolved by database adapters which implement data manipulation functions according to the language and structure of source systems. General schemas is the remedy for this problemOur fundamental concept is to produce general, independent explanations of the data model, containing both structural and semantic information from source database systems.As a general schema, this general data model description is called general schema (GS). GSs are described in JSON objects, because JSON is a language-independent open-standard format, and its self-describing nature allows us to easily describe and understand the main characteristics of schemas.

General schemas are automatically generated from the source systems. In the case of relational DBMSs, it means, that GSs are generated from tables or views. As NoSQL systems are based on different data modeling approaches, they implement different database objects as well. In the case of document stores, the basic database object is the collection, and so general schemas are created from the collections.

Each general schema has unique name, which is generated from the source object. In addition, each GS contains the information regarding its source database object:

• Information about the source system: Information about the source system includes, for example, the name and the database type of the source DBMS. These data are required in the connection to the server and to generate queries.

• Structural description of the database object: Structural description includes names, data types and constraints of the elements (e.g. attributes) of the source object (e.g. table). Furthermore, general schemas define key attribute(s) (orobject IDs) as well, which is crucial information for joining related data.

• Semantic mapping of the elements of the database object: General schemas define an alias for each attribute or element in the object. These aliases guarantee the consistent name conversion for the elements in GSs. Aliases can be defined manually but using ontology-based methods, proposed in the literature, they can also be generated semi-automatically. With the use of aliases, users can reference attributes without having to includeits containing object (e.g. table) and database. If attributes (even in different databases) receive the same alias value, they are considered as semantically identical.

• Relationships of the database object with other objects: If the source object of a general schema refers to another object of the system, then GS generated from the referencing object contains information about this relationship. These links allow connecting the related data.

5.4 JackHare framework using MapReduce

For easy use of theMapReduce and HBase for ANSI-SQL users, the JackHare framework comprises front-end and back-end to translate ANSI-SQL queries and perform logical operations while accessing HBase[11]. Instead of RDB, JackHare aims to ease the learning curve of analyzing large-scale NoSQL database datasets. Figure 2 shows the architecture of JackHare framework. The front-end consists of a GUI, a com-piler and a JDBC driver, while the back-end consists of a scalable NoSQLdatabaseHBase. Users are able to select an application as a GUI, e.g. SQuirreL SQL client, to import the JDBC driver. The JDBC driver includes an ANSI-SQL query

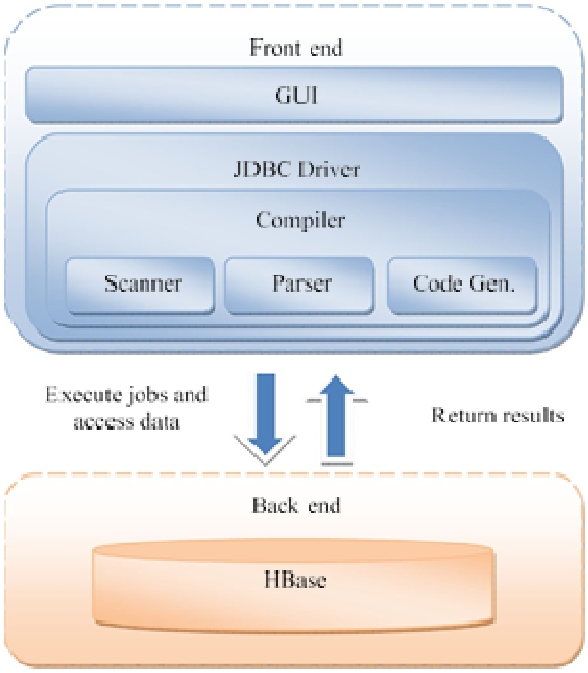

Figure 2: JackHare Framework Architecture

compiler to scan, parse and execute queries. The scanner, parser and code generator are implemented using the open source project, Druid. While an ANSI-SQL query submitted, the scanner breaks queries into tokens. The parser manipulates the abstract syntax tree which is comprised of tokens. The code generator produces MapReduce jobs to access HBaseand then perform logical operations according to the parsed commands. The outcomes of ANSI-SQL queries are returned and displayed on GUI.

The data models of tables in HBase are different from RDB. Here we define the data model used for HBase. The relation

between tables in RDB and HBase should be given for the code generator to precisely retrieve the information of the right column in right table to produce MapReduce job. The JackHare framework chooses Apache Derby as the metadata serverDerby stores the HBase table name, column families, and column qualifiers that can be mapped to the RDB database name, table name, and column name.

5.5 DualFetchQL

The DualFetch Function phase is the center of the architecture and responsible for the relationship between other components of the system as well as determines how the system functions[12]. It extracts from the Abstraction Layer the query entered by a client, decides the form of query and communicates with the necessary databases, and eventually sends the query result back to the Abstraction Layer for presentation.

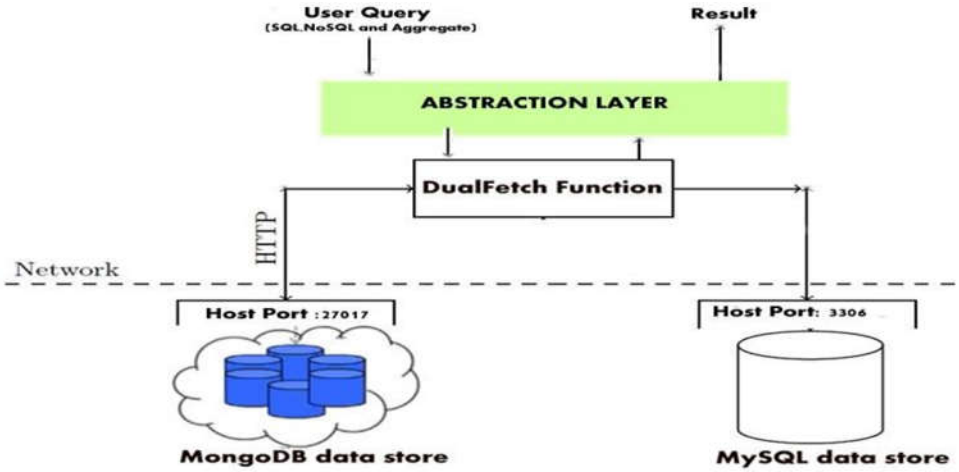

Figure 3: DualFetchQL Architecture

The architecture of the system developed is made up of four major phases,DualFetch Function, an Abstraction Layer, a MongoDB data store and a MySQL data store.

The DualFetch Function phase is termed as thethe center of the architecture and is responsible for the relationship between other components of the system as well as determines how the system works. It extracts the query entered by a client from the Abstraction Layer, determines thetype of query and interacts with the appropriate databases and, finally, sends back result of the query to the Abstraction Layer for presentation.

The Abstraction layer is the interface of the user. Contains mainly two parts; the part for inputingqueries and the other part for displaying the results.Whenan user starts execute action, the query entered by the user is passed to DualFetch Function phase. The DualFetchphase processes the query and returns necessary to Abstraction Layer.

The MongoDB data store is the environment where the server of the MongoDB resides..MongoDB related queries are transferred to the MongoDB data store by the DualFetch Function for execution. After query execution the corresponding result is given back to DualFetch Function.

The MySQL Data Store Stage hosts the MySQL database server..All SQL related queries are handed over by the DualFetch function for execution to the MySQL data store..The query is performed with the response returned to the Function DualFetch.

To implement the DualFetchQL System, we needed to choose one RDBMS and one NoSQL implementation. Here we use the MySQL server for SQL database manager and the MongoDB for NoSQL administration. These were chosen due to their popularity and also have Java interfaces that make them easier to interact with.

A software layer for querying SQL and NoSQL databases is provided by the DualFetchQL Framework..A new query syntax called aggregate query was developed that is used when both SQL and NoSQL database data is needed in a single view.

The DualFetchQL System analyse the queries to determine in which of the databases the required data is located and provide the output of the query on the result panel of the Layer.

6.Comparing the models.

The integration of mongodb and sql are needed for the management tools,account tools ,E -commerse etc..

In this session we have discussed about different integration methods. They areUnity,DataAdapter,Novel Integration Data Methodology,JackHare framework using MapReduceand finally DualFetchQL. Thus models uses different formats for integrating SQL and NoSQL. But few components of the Unity and Data Adapter are same like both use a parser. In the same way that all models have a one or more same components but that doesn't make the different models same**.**

To select a best model from the mentioned is difficult because each model has its own features and functional specialities. Also the structure of different integration methods had already disscussed above. Therefore we felt DualFetchQL model is better.It is a simple process for those who can handle SQL and NoSQL. The main feature of this model is usage of single software layer called DualFetch Function phase for the overall management of the system.In this model the query is divided into SQL and NoSQL query with the help of abstraction layer then the request query is translated to SQL and NoSQL query, the DualFetch layer will fetch the data's from there corresponding databases and are transferred back to the abstraction layer ..

7. Conclusion

In this paper we have presented several methods based on integration of SQL and NoSQL databases. The first one was Unity which is an integration and virtualization system allowing SQL queries over both relational and NoSQLsystems.The next is data adapter uses a general SQL layer accepts requests from application providers, so that the original application does not need to alter the design.. The data adapter is also responsible for the query flow during database transformation. A novel metamodelbased approach to data fusion, which enables

simultaneous querying of data from heterogeneous database systems. the JackHare framework with a comprehensive solution including SQL query compiler, JDBC driver and a systematical method using MapReduce for processing the unstructured data in NoSQLdatabase.Helps in employing NoSQL database like HBase, SQL user can import the JDBC driver to SQL client software to manipulate the large-scale data without being aware of NoSQL database DualFetchQL System that acts as a software layer for both SQL and NoSQL database query processing. We also developed a new query syntax called an aggregate query that is used in a single view when data from both SQL and NoSQL databases is needed. Therefore, every method has driven us to the concept: data integration. Data integration helps us to handle more easier. In one side database with relations and the other side is non-relational. So existence of such integration mechanism or methods while dealing with database makes the job more simple. Even though we have presented many methods, each methods are somewhere similar but are not the same. That is, for integration of databases there doesn't exist a perfect method: one and only one single method. So in future we ill try to put forward a perfect mechanism that is more efficient and reliable than the methods mentioned above.

References

[1] R. Asgari, M.G. Moghadam, M. Mahdavi, A. Erfanian, An ontology-based ap- proach for integrating heterogeneous databases, Open Comput. Sci. 5 (1) (2015) 41–50.

[2] R. Asgari, M.G. Moghadam, M. Mahdavi, A. Erfanian, An ontology-based ap- proach for integrating heterogeneous databases, Open Comput. Sci. 5 (1) (2015) 41–50.

[3] B.G. Tudorica, C. Bucur, A comparison between several nosql databases with comments and notes, in: 2011 RoEduNet International Conference 10th Edi- tion: Networking in Education and Research, 2011, pp. 1–5, doi:10.1109/ RoEduNet.2011.5993686.

[4] SQL-92, 4.22 SQL-statements, 4.22.1 Classes of SQL-statements "There are at least five ways of classifying SQL-statements:", 4.22.2, SQL statements classified by function "The following are the main classes of SQL-statements:"; SQL:2003 4.11 SQL-statements, and later revisions.

[5] Codd, Edgar F. (June 1970). "A Relational Model of Data for Large Shared Data Banks". Communications of the ACM. 13 (6): 377–87. CiteSeerX 10.1.1.88.646. doi:10.1145/362384.362685. S2CID 207549016.

[6] http://nosql-database.org/ "NoSQL DEFINITION: Next Generation Databases mostly addressing some of the points : being non-relational, distributed, open-source and horizontally scalable".H. Niu, M. Iwai, K. Sezaki, L. Sun, and Q. Du, __Ex- ploiting fountain codes for secure wireless delivery,'' IEEE Commun. Lett., vol. 18, no. 5, pp. 777–780, May 2014.

[7] Fowler, Martin. "NosqlDefinition". many advocates of NoSQL say that it does not mean a "no" to SQL, rather it means Not Only SQL.

[8] R. Lawrence, "Integration and Virtualization of Relational SQL and NoSQL Systems Including MySQL and MongoDB," 2014 International Conference on Computational Science and Computational Intelligence, Las Vegas, NV, 2014, pp. 285-290, doi: 10.1109/CSCI.2014.56..

[9] Ying-Ti Liao, Jiazheng Zhou, Chia-Hung Lu, Shih-Chang Chen, Ching-Hsien Hsu, Wenguang Chen, Mon-Fong Jiang, Yeh-ChingChung,Data adapter for querying and transformation between SQL and NoSQL database, Future Generation Computer Systems,https://doi.org/10.1016/j.future.2016.02.002. (http://www.sciencedirect.com/science/article/pii/S0167739X16300085)

[10] ÁgnesVathy-Fogarassy, TamásHugyák,Uniform data access platform for SQL and NoSQL database systems,Information Systems, https://doi.org/10.1016/j.is.2017.04.002. (http://www.sciencedirect.com/science/article/pii/S0306437916303398)

[11] Chung, WC., Lin, HP., Chen, SC. et al. JackHare: a framework for SQL to NoSQL translation using MapReduce. AutomSoftwEng 21, 489–508 (2014). https://doi.org/10.1007/s10515-013-0135-xO.V. Joldzic, D.R. Vukovic, The impact of cluster characteristics on HiveQL query optimization, in: 21stTelecommunications Forum, TELFOR, 2013, pp. 837–840.

[12] ThankGod S. Adeyi, Saleh E. Abdullahi, Sahalu.BJunaiduDepartment of Mathematics, Ahmadu Bello UniversityZaria, Kaduna State, NigeriaDualFetchQL System: A Platform for Integrating Relational andNoSQLDatabases.

Author Profile

Krishnapriya VM received Degree in Bachelor of Computer Applications from Mahatma Gandhi University Kottayam, Kerala in 2020.

Libin Sebastian received Degree in Bachelor of Computer Apllications from Mahatma Gandhi University Kottayam, Kerala in 2020.

Gibin George received Masters in Computer Application from Mahatma Gandhi University, Kottayam , Kerala in 2008.