Volume 3, Issue 9, September-2016, pp. 506-513 ISSN (O): 2349-7084

International Journal of Computer Engineering In Research Trends

Available online at: www.ijcert.org

Relational Database to NoSQL Conversion by Schema Migration and Mapping

Neelima Kuderu[1], Dr. Vijaya Kumari[2]

[1]M.S Part Time Research Program, Dept., of CS&E, JNTU, Ananthapur,
[1]Email: neelimakenpi@gmail.com
[2]Professor, JNTUH, Hyderabad

Abstract: In pc software development, migration from a Data Base Management System (DBMS) to NOSQL, specifically applications like material administration systems with distinct attributes is a challenge for coders and database directors. Changes in the application rule in purchase to comply with new DBMS tend to be generally vast, causing migrations infeasible. In order to tackle this problem that is nagging we present RDBDMS-to-NoSQL by schema migration and query mapping to help conveniently migrating from relational DBMS to NoSQL DBMS. This framework is presented in two components: (1) schema mapping by reverting the forms that are regular, (2) mapping SQL query to NoSQL. Initial one is a set of practices enabling migration that is smooth relational DBMS to NOSQL DBMS. The latter provides a persistence layer to process database needs, becoming capable to translate and execute these demands in any DBMS, going back the information in a format that is suitable really. Experiments shown that the proposed design is scalable and robust to manage volume that is huge of of applications such CMS that contrasted to conventional relational DBMS techniques.

Index Terms: Big Data, RDBMS, NoSQL, SQL, Schema Migration, SQL Mapping, CMS, Denormalization

1 INTRODUCTION

Collecting, saving and integrating large quantities of information are rapidly becoming a need among pc software designers in business, also by researchers in analysis settings. Crisis informatics [1] is one study area in which this need hasn't been higher. Crisis informatics researches exactly how information and interaction technology are utilized in emergency and hazard reaction. An emerging branch of this control investigates just how users of the public make use of social media and various other types of computer system interaction that is mediated aid one another during times of mass disaster [2]. The analysis of this kind of data relies greatly on a robust and data being scalable infrastructure. The nature that is ephemeral of information (e.g. CMS revisions) requires collection is done in real time and with uncompromising dependability.

In computer software development, migration from information Base Management System (DBMS) to NOSQL, especially applications like material management systems with distinct characteristics is a challenge for programmers and database administrators. Modifications in the application rule in order to comply with brand-new DBMS tend to be generally vast, causing migrations infeasible.

Content management systems (CMSs) tend to be able to allow folks, who have no skill that is manage that is technical, quickly develop, edit, organize and publish online contents. For example, CMSs are usually utilized to establish e-commerce stores that are online community portals, private blog sites, or company websites. Thus, CMSs play the part of obtaining information then are quickly extended to become internet that is intelligent. Most CMSs that is popular,, WordPress and Joomla, rely on relational databases or Relational Databases, e.g., MySQL, PostgreSQL, etc. Nonetheless, due to the explosive development of huge data, the well-structured feature of Relational Databases may restrict the scalability to deal with scaling that is horizontal. Therefore, regarding the versatility and the feasibility of synchronous processing, the data storage space techniques takes Not Only SQL (NoSQL) databases into consideration. Exactly how to migrate the CMS that is initial pc software Relational Databases to NoSQL Databases becomes one rising and analysis problem that is crucial.

2 RELATED WORK

As a result of interest in the NoSQL databases, an entire large amount of works discussed and reviewed the SQL and NoSQL databases. As an example, Lombardo et al. [1] discussed problems about complex data frameworks into the NoSQL database. Li and Manoharan [2] analyzed the overall performance between SQL and key-value NoSQL databases. Boicea et al. [3] contrasted Oracle, for example. the standard database that is SQL and MongoDB, for example., the document NoSQL database from theoretical variations, feature restrictions and system design. Additionally, Grolinger et al. [4] identified dilemmas and difficulties in handling information which can be huge MapReduce. Four categories being main (1) data storage, (2) big data analytics, (3) online processing, and (4) safety and privacy. Naheman and Wei [5] considered the comparison between HBase and other NoSQL databases, e.g., BigTable, Cassandra, CouchDB, MongoDB, etc. Based on

their observance and evaluation, NoSQL databases may well not always improve performance than SQL databases. Scavuzzo et al. [6] provided an approach to information which are migrate different column-oriented NoSQL databases. One metamodel that is intermediate recommended to portray information in a common format and responsible for the interpretation from a source database into the target one.

About the emergent element huge information, more and more scientists which are recent on how best to integrate SQL and NoSQL. For example, Hsu et al. [7] proposed and created the technique this is certainly correlation-aware Sqoop that is used to transform the data kept in the standard relational database to the HBase. The proposed correlation-aware technique is able to analyze which tables are often utilized for the query this is certainly crosstable then transform these data into the same Hadoop data-node. For HBase, i.e., column-oriented NoSQL database, Zhao et al. [8] combined all associated tables into different column families. Hence, the connections in all SQL tables are preserved entirely into one NoSQL that is solitary dining table. Regarding SQL table nesting circumstances, Zhao et al. [9] proposed a transforming algorithm this is certainly graphbased. The proposed algorithm might cost more space to boost the question overall performance after schema conversion for MongoDB, in other words. document NoSQL database. Sellami et al. [10] developed PaaS that is available database (ODBAPI) that is a streamlined and REST-based API in order to execute the CRUD operations, i.e., create, read, update and erase, on the SQL and NoSQL databases. Li et al. [11] designed a data that is query-oriented (QODM) method, in which the information kept framework plus the information question demands should be validated then the NoSQL database schema will be made by the proposed QODM.

Some works enabled the cross-table question throughout the NoSQL database in addition to move the SQL database into the NoSQL database. Gadkari et al. [12] proposed a model that is theoretical apply the cross-table question on the HBase and get the produced outcomes faster. Wei et al. [13] implemented a middleware level called Cloud TPS make it possible for join queries and hold powerful persistence this is certainly transactional NoSQL databases, e.g., SimpleDB and HBase. Regarding the document NoSQL database, Lawrence [14] offered a virtualization system which allows to question and join information making use of a SQL query and automatically result in the API that is underlying to MongoDB. Eventually, Hieu et al. [15] considered the key-value NoSQL database and analyzed the MapReduce join strategies, including theta-join, all pair partition join, repartition join, broadcasting join, semi join, presplit semi join.

The benchmarking this is certainly explored are evinced as sturdy under contextual factors centered for instance the item, structure associated with the RDBMS. The constraints noticed frequently for several of these existing models tend to be, (i) they are bound into the RDBMS structure for the particular product, (ii) the tree structure development from the SQL structure is recursive, thus the computational complexity is complement and (iii) the objective of all those models restricted to draw out the outcomes much faster under SQL mapping to NoSql and optimization of migrating from SQL to NoSQL is the very least considerable.

In order to overcome the limitations of the existing designs explored, a RDBDMS-to-NoSQL this is certainly common by migration and query mapping to support conveniently migrating from relational DBMS to NoSQL DBMS. This framework is presented in multifold and are (1) checking out schema by denormalizing the RDBMS table frameworks, that will help to draw out the Schema of the target RDBMS regardless of item specific structure, therefore the proposal is common to virtually any RDBMS aside from their particular product and structure, (2) migrating the schema to NoSQL tree construction that eliminates redundant tree structure development, therefore the proposition is sturdy, scalable and computationally linear and (3) reforming SQL queries to gain access to NoSQL data, this period enables the qualitative evaluation of the proposed SQL to NOSQL migration framework, which evinces the compatibility of this resultant NoSQL structure for the proposition to the SQL queries.

3 SCHEMA MIGRATION AND QUERY MAPPING FRAMEWORK

This paper proposed the migration that is complete of databases to NoSQL, keeping only 1 database, eliminating the need to decide between a supply or another. Moreover, this tactic keeps the application unchanged and considering the information as being stored in a model that is relational. The key challenge is to provide an entire abstraction regarding the relational model for NoSQL model, since they will be completely different w.r.t in order to achieve this goal. Their structures, additionally the means to make use of it as an abstraction layer.

3.1 Schema Migration Layer

The Relational Database has the well-structured data and supports the query that is cross-table. However, the well-structured characteristic of the Relational Database also limits scaling that is horizontal. Hence, the contribution that is key of paper is reverts the normal forms of the Relational Database schemas and then migrate into a NoSQL Database autonomously.

3.2 Reverting Normal Forms and Row-key Selection

So far as we all know that there is no explicit or procedure that is standard column-oriented NoSQL Database to create the column-oriented NoSQL schema. The column-oriented NoSQL Database only has the DDI design principle, i.e., (1) denormalization, (2) duplication and (3) intelligent keys in order words. First, denormalization and replication mean to re-aggregate associated SQL tables into a noSQL that is big no matter if some information should be duplicate or redundant in the NoSQL table. Then, each row should choose one key that is intelligent line key for unique recognition. No cross-table query is needed in the NoSQL Database since all related SQL tables are merged into a large table. Consequently, we're motivated to check out the DDI design principle and avoid any query that is cross-table the NoSQL Database.

About the table that is traditional design, the tall-narrow design pattern lets a table have few columns but the majority of rows. The flat-wide design pattern lets a table have actually few rows but many columns on the other hand. The proposed process will try to create the NoSQL Database with the tall-narrow design pattern since HBase, which is utilized in this paper for implementation, can only split at the row boundary. To experience the tall-narrow design, the selected row key must keep the cardinality that is greatest. The row type in the NoSQL table ought to be the concatenation of several primary secrets inside our proposed process since primary keys are useful to identify each row in the SQL tables. Upcoming, the relationship is taken by us among SQL tables into consideration. A is related to another table called Table B by a foreign key for instance, one SQL table called table. However, Table B is additionally related to another table called Table C. Based on our observation and studies, the chained relationship is longer, the principal key of the table that is corresponding i.e., Table C in this example, keeps higher cardinality. Thus, in this example, Table C's primary key would be selected as the row key into the NoSQL table. The line key will be the concatenation of these SQL tables' primary keys if a lot more than one SQL table with the same chained relationship.

3.3 Schema Migration

As portrayed in Figure 1, the recommended schema migration is independent, i.e., create the NoSQL schema instantly after the SQL schema analysis. All SQL schemas in a single special table called “information schema”. Hence, we can parse and draw out each table's major key and international tips from the Relational Database in our execution, Relational DBMS stores.

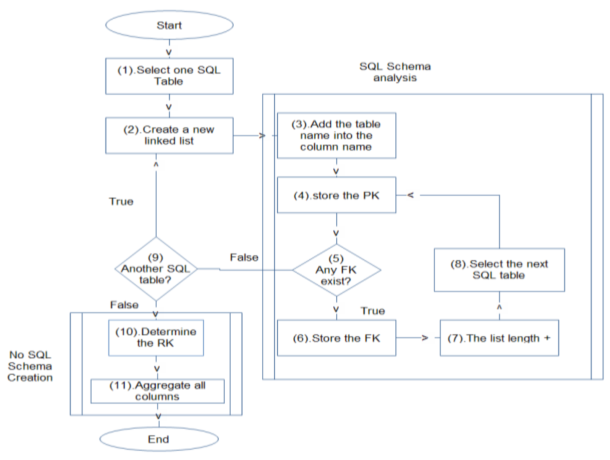


Figure 1: The flowchart of the proposed schema migration

3.4 The schema migration execution flow:

1) One table is selected through the Database that is relational for SQL schema analysis.
2) A linked list is done to keep the sum total outcome of the SQL schema evaluation. The functionality that is primary of linked listing is used to calculate the chained relationship.
3) Considering that the column this is certainly same might evince around in different tables, the recommended migration should add the table title into each column title for obvious recognition.
4) The key (PK) this is certainly primary kept given that row Key (RK) candidate associated with the NoSQL table.
5) The proposed migration determines whether there clearly was any key this is certainly foreignpFK) been around in this SQL table or not.
    a) If true, head to Step 6.
    b) If untrue, head to Step 9.
6) The foreign key (FK) is stored to aggregate the table that is relevant.
7) On the basis of the key that is foreignFK), the related dining table is included to the linked list. Then, the record length is incremented by one.
8) We find the table that is related the next table and repeat the SQL schema evaluation again, head to step 4, until all relevant table tend to be included in to the connected list.
9) The recommended migration determines whether there was just about any table that will be perhaps not analyzed yet or perhaps not.
    a) If real, head to Step 2.
    b) If untrue, head to move 10.
10) The recommended migration determines the line secret (RK) based on the kept keys that are primaryPKs).
11) The suggested migration aggregates all columns into one NoSQL table.3.3.3 Data Mapping Module

An abstraction layer is provided by this module (i.e., persistence layer) between the application and the NoSQL DBMS. The goal with this module is to enable a database that is seamless, preventing any change in the application code before changing the information model used. Therefore, application developers will continue creating queries within the model that is relational nevertheless the data will be fetched in a NoSQL Database, benefiting from the advantages of performance and scalability offered by this management system. Figure 2 illustrates the operation that is working of module.

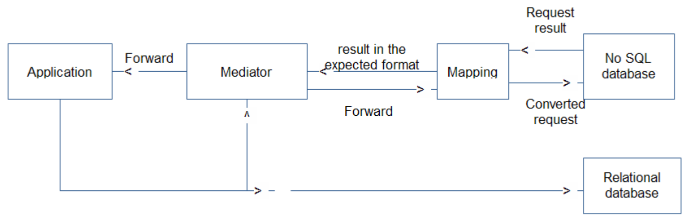


Figure 2: Data Mapping Module Working Diagram

The first task with this module is to intercept queries issued by the application to the DBMS, to be able to redirect them to the NoSQL that is suitable DBMS. In this sense, we created a sub-module called Mediator. We built Mediator using Relational DBMS Proxy [16], an source that is open that executes the communication between Relational DBMS server and customer application. The LUA is used by this proxy language [17] for performing data manipulation. It also comes with predefined functionalities in order to intercept queries, refine the sum total results and deliver signals regarding queries done successfully or with mistakes. We changed the Relational DBMS Proxy using LUA, so these operations are properly intercepted and forwarded to a sub-module that is second which is accountable for question conversion. We implemented the communication between your Mediator while the demand conversion sub-module utilizing the LuaSoap that is making use of library, which utilizes SOAP (Simple Object Access Protocol) protocol and XML files containing all the information related to the operations.

The transformation sub-module, called Convert, gets from Mediator all requests forwarded to the database that is relational converts them to questions supported by NoSQL. This sub-module was developed as a WebService Java that is utilizing language. It waits for new needs that are sent by Mediator as XML files. An example of XML files utilized in the request communication is presented below. The first attribute for the xmlns represents which class of the Web Service should be used (i.e., query Interceptor) in this example. The characteristic that is second which method should be executed (i.e., Intercepta). Furthermore, we highlight two necessary parameters: query and queryType, which represent, correspondingly, the request that is intercepted its type.

Convert evaluates the parameter queryType so as to identify what must be the procedure that is next be executed, according to the procedure kind (i.e. Select, Insert, Update or Delete). A corresponding method accounts for doing the operation onto the NoSQL Database and going back the effect of the operation for every single operation. All methods perform the steps being exact same below, with some peculiarities related to each transaction:

Information extraction about the question: The method evaluates the query parameter to be able to gather information regarding the SQL operations, such as tables, characteristics, etc., in addition to the criteria found in where clauses. We implemented this technique Java library JSQLParser that is using [18]. Exactly what differentiates each method in this step may be the given information that each one exploits. The existing joins, because well as functions for sorting and grouping for ex-ample, even though the matching operation towards the SQL Select method needs to treat the involved tables. An insert operation must address only the attributes and tables involved with the operations in the other hand.

Generation and implementation of equivalent procedure in NoSQL: This step corresponds to translate the SQL operations onto its equivalent NoSQL ones. The translation process is dependant on the NoSQL that is official form SQL operations mappings [19]. The collection Metadata, which stores the relationship of the database that is original its correspondence in NoSQL Database, is essential. This new operation is executed on NoSQL and the outcome is processed in the alternative during the end.

Mapping return results: the total results returned by NoSQL are then sent to Mediator, which is responsible to forward the result to your application. First, the total result header is built, which contains the correct identifications of tables, attributes, along with other elements pertaining to the end result. Then, each record came back by NoSQL is mapped, following the header built.

This mapping also supports SQL nested transactions, such as a Delete, in that the items to rely be removed on a Select. Initially, Schema Migration Model triggers the corresponding technique pertaining to the absolute most working that is outside. During the parser execution, when the nested transaction is identified, the method that is corresponding that operation is triggered so that you can completely resolve it. Then, it returns the mandatory information to execute the operation that is external. Considering once more the Delete procedure, the technique that is respective this operation, mapped by the parser execution, identifies the Select procedure and invokes the strategy accountable to manage it. Caused by the Select execution is then returned to the technique Delete. Each nested procedure regarding the method that is corresponding invoked. This scheme supports also the application of recursive telephone calls (age.g., a few queries that are nested, in which the limitation of these calls is only subject to the hardware resources availability.

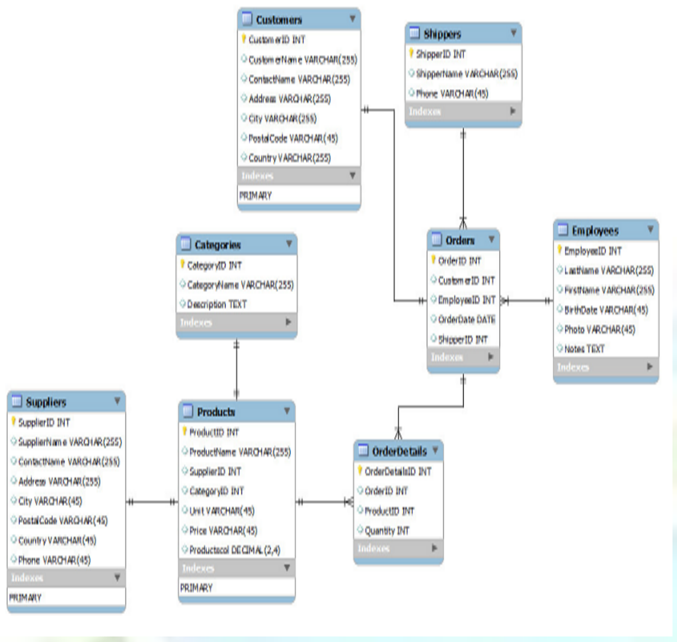
Finally, Mediator handles the XML sent by Convert. Mediator is responsible for transforming the XML in to the Relational DBMS format expected by the DBMS that is relational Resource. In order to perform this, Mediator utilizes the help of Relational DBMS-Proxy.

4 EXPERIMENTAL SETUP AND EVALUATION

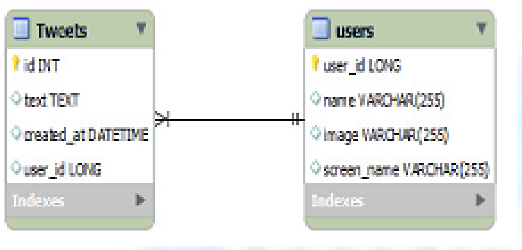
So as to evaluate the RDBMS-TO-NOSQL Conversion by schema mapping and migration, centering on two forms of analysis: qualitative and quantitative. While in the evaluation that is qualitative goal is to provide a proof of concept by showing the Schema Migration and Mapping Framework execution in practice, in the quantitative one we aim to verify whether the application of NoSQL, with our framework, leverages the system performance.

4.1 Experimental Setup

Within our evaluation, we used two databases as provided in Figure 3. 1st one comes from W3Schools (http://www.w3schools.com/). This database may capture characteristics quite different from those typically utilized in all the applications, such stability limitations, connections or various types of data although showing structure that is quick. The other database is certainly quick regarding an application that collects and places user tweets from Twitter. This database features a volume that is high of information that could help us, mainly, to the assessment this is certainly quantitative of Migration and Mapping Framework.



a. W3schools



b. Twitter

Figure 3: Database models used in the evaluation.

Both databases were implemented using Relational DBMS. An application, written in Java, was created with the purpose of executing several SQL operations for each database. We call this con-figuration DBMS that is relational scenario. For each arrangement application/database, we created another version that used Schema Migration and Mapping Framework to perform the migration that is entire of Relational Database to NoSQL, including the structure and the data themselves. The Schema Migration and Mapping Framework was also used to capture and map the operations requested by the applications and NoSQL for this latter configuration, which we call Schema Migration and Mapping Framework Scenario. Thus, we performed quantitative and qualitative comparisons of the total results achieved by each scenario. Although several operations that are SQL tested during the development process.

4.2 Qualitative Assessment on CMS Systems

The proposed RDBMS-to-NoSQL that is autonomous schema is assessed using two popular CMSs, i.e., WordPress and Joomla. First, we migrate the Relational Database schemas of WordPress and Joomla directly to the NoSQL Database by the Sqoop and then utilize HiveQL to perform the query that is cross-table. On the other hand, we also migrate the Relational Database schemas using the proposed schema migration cross and mechanism table querying is achieved by mapping module. For each CMS, we utilize the Talend Open Studio Big Data tool to generate up to randomly 10 million transactions.

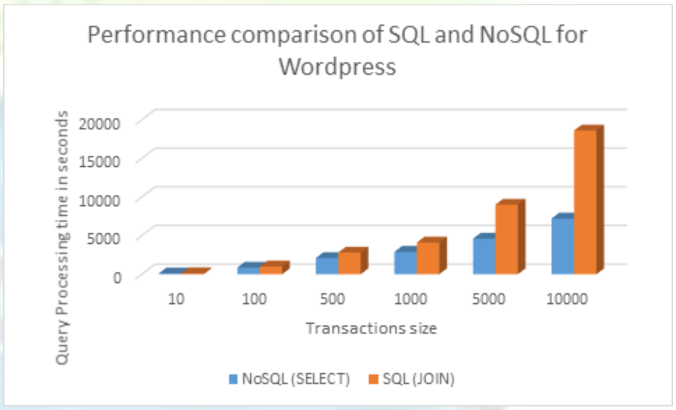


Figure 4: The performance comparison over the WordPress CMS

4.3 WordPress

WordPress is widely deployed to establish websites. According to the 2015 CMS investigation, WordPress was used by more than 23% of top websites. WordPress's Relational Database is composed of total 11 SQL tables. After our schema that is SQL analysis WordPress's SQL tables can be divided into two groups. One group is composed of 3 tables that are SQL the other one is composed of 6 SQL tables. The remaining 2 SQL tables are independent, i.e., no key that is foreign. Figure 4 depicts the performance comparison over the WordPress CMS. When the data size increases, two mechanisms require more time to complete the query. However, our proposed mechanism requires much less time while the data size is 10 million transactions. The proposed mechanism is able to save 45% query time on average.

TT

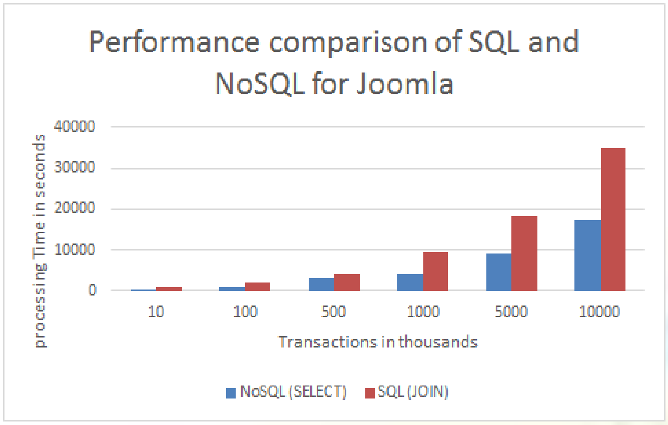


Figure 5: The performance comparison over the Joomla CMS

4.4 Joomla

Joomla is another CMS to publish web contents and estimated to be the second most CMS that is popular after. Joomla's Relational Database has much more SQL tables than WordPress. There are totally 34 tables that are SQL. Based on the total results of our SQL schema analysis, Joomla's SQL tables can be categorized into three groups. The group that is largest is composed of 17 SQL tables. Thus, Joomla's Relational Database has relationship that is complex. Figure 5 depicts the performance comparison over the Joomla CMS. Similar to the results that are wordPress's two mechanisms require more time to complete the query when the data size increases. Regarding the Joomla CMS, our proposed mechanism is able to save 48% query time on average.

4.5 Schema Migration Analysis

Content administration systems (CMS) are popular Internet system to publish web contents and able to increase functionalities that are novel e.g., intelligent data analysis or choice helps. But, most CMSs are nevertheless based on the conventional Relational Databases. Because the NoSQL Databases provide better support and scalability parallel processing to deal with data which can be big. This paper is motivated to propose an RDBMS-to-NoSQL that is schema that is autonomous migration. The CMS administrators or designers require not redesigning their database schemas in other words. According to the full total results that are experimental WordPress and Joomla, the proposed mechanism is ready to truly save 45~48% question time on average. The work that is considering that is future to divide columns into different column families.

5 CONCLUSION

This paper proposed a Schema-Migration and Mapping Framework to support developers at migrating automatically from relational databases to NoSQL while preserving the semantics of the database that is original. Schema Migration and Mapping Framework has a database persistence layer enabling applications to access data in NoSQL model seamlessly, with no need of modifying queries and application code. Schema Migration and Mapping Framework was implemented in Java, employing concepts of object-oriented programming, in order making it easily extensible to all or any of DBMS strategies.

REFERENCES

[1] S. Lombardo, E. Di Nitto, and D. Ardagna, "Issues in Handling Complex Data Structures with NoSQL Databases," Proceedings of the 14th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), pp. 443-448, Sept. 2012

[2] Yishan Li and S. Manoharan, "A performance comparison of SQL and NoSQL databases," Proceedings of IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM), pp. 15- 19, Aug. 2013.

[3] A. Boicea, F. Radulescu, and L.I. Agapin, "MongoDB vs Oracle -- Database Comparison," Proceedings of The 3rd International Conference on Emerging Intelligent Data and Web Technologies (EIDWT), pp.330- 335, Sept. 2012.

[4] K. Grolinger, M. Hayes, W.A. Higashino, A. L'Heureux, D.S. Allison, and M.A.M. Capretz, "Challenges for MapReduce in Big Data," Proceedings of IEEE World Congress on Services (SERVICES), pp.182- 189, Jun. 2014.

[5] W. Naheman and Jianxin Wei, "Review of NoSQL databases and performance testing on HBase," Proceedings of International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC), pp. 2304-2309, Dec. 2013.

[6] M. Scavuzzo, E. Di Nitto, and S. Ceri, "Interoperable Data Migration between NoSQL Columnar Databases," Proceedings of IEEE 18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations (EDOCW), pp.154-162, Sept. 2014.

[7] Jen-Chun Hsu, Ching-Hsien Hsu, Shih-Chang Chen, and Yeh-Ching Chung, "Correlation Aware Technique for SQL to NoSQL Transformation," Proceedings of the 7th International Conference on Ubi-Media Computing and Workshops (UMEDIA), pp. 43-46, Jul. 2014.

[8] Gansen Zhao, Libo Li, Zijing Li, and Qiaoying Lin, "Multiple Nested Schema of HBase for Migration from SQL," Proceedings of 9th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), pp.338-343, Nov. 2014.

[9] Gansen Zhao, Qiaoying Lin, Libo Li, and Zijing Li, "Schema Conversion Model of SQL Database to NoSQL," Proceedings of 9th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), pp.355-362, Nov. 2014.

[10] R. Sellami, S. Bhiri, and B. Defude, "ODBAPI: A Unified REST API for Relational and NoSQL Data Stores," Proceedings of IEEE International Congress on Big Data (BigData Congress), pp.653-660, Jun. 2014.

[11] Xiang Li, Zhiyi Ma, and Hongjie Chen, "QODM: A query-oriented data modeling approach for NoSQL databases," Proceedings of IEEE Workshop on Advanced Research and Technology in Industry Applications (WARTIA), pp.338-345, Sept. 2014.

[12] A. Gadkari, V.B. Nikam, and B.B. Meshram, "Implementing Joins over HBase on Cloud Platform," Proceedings of IEEE International Conference on Computer and Information Technology (CIT), pp. 547- 554, Sept. 2014.

[13] Zhou Wei, G. Pierre, and Chi-Hung Chi, "Scalable Join Queries in Cloud Data Stores," Proceedings of the 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), pp.547- 555, May 2012.

[14] R. Lawrence, "Integration and Virtualization of Relational SQL and NoSQL Systems Including MySQL and MongoDB," Proceedings of International Conference on Computational Science and Computational Intelligence (CSCI), pp.285-290, Mar. 2014.

[15] D. Van Hieu, S. Smanchat, and P. Meesad, "MapReduce join strategies for key-value storage," Proceedings of the 11th International Joint Conference on Computer Science and Software Engineering (JCSSE), pp.164-169, May 2014.

[16] Mysql proxy. http://dev.mysql.com/refman /5.6/en/mysql-proxy.html. Accessed:2014-09-24.

[17] Roberto Ierusalimschy, Luiz Henrique de Figueiredo, and Waldemar Celes Filho. Lua—an: Ex-tensible extension language. Softw. Pract. Exper., 26(6):635–652, June 1996.

[18] Jsql parser. http://jsqlparser.sourceforge.net/. Accessed:2014-09-24.

[19] Mongodb mapping chart. http://docs.mongodb.org/ manual/reference/sql-comparison. Accessed:2014-09-24.