Asian Journal of Information Technology 16 (2-5): 312-316, 2017
ISSN: 1682-3915
© Medwell Journals, 2017

A Framework for Data Portability from Relational Database to Document Database

Nikitha Ramakrishnan and Monisha Singh
Department of Computer Science, Christ University, Hosur Road, Bengaluru, Karnataka, India

Abstract: Cloud computing is advancing day by day. Several services are offered to the consumers by the cloud providers. The main challenge faced by any cloud consumers is the portability of data between the applications. It is difficult to move data from one application to another, since each application has different data models. In this study, a framework is proposed which transfers data from relational data model to key-value store and vice versa. Using the proposed framework one can easily convert, transform and exchange data between different data models. The research also performs evaluation methodology in order to show the efficiency of the system. The proposed system enables one to either minimize code modification or eliminate the need to modify the code.

Key words: NoSQL, data portability, key-value store, relational database, document database, eliminate

INTRODUCTION

Cloud computing is growing unexpectedly. It is a new paradigm in which the resources can be shared. It is on demand computing where computers and other devices are provided with data and information. It is a term which is used to explain diverse scenarios where computing resources are shared as a service over the internet (Shawish and Salama, 2014). Rather than deploying personal hardware and software, cloud computing allows to share the resources over internet. In short cloud computing means delivery of on-demand computing resources. Cloud computing enables in reducing the cost incurred in hardware and software resource management. One of the major advantage of cloud computing is the pay as you go scheme which allows access to offerings like IaaS (Infrastructure as a Service), PaaS (Platform as a Service) and SaaS (Software as a Service). Further, cloud computing also provides DaaS (Database as a Service) that facilitates different data or storage models to end users. There are several cloud providers who provide these services to the end customers. Currently, new administrations and components are provided to the consumers with a view to solve their issues in an efficient and cost effective way. However, one of the major challenges faced by cloud is the portability and interoperability of services.

Different cloud providers use different data models to store the data. This makes it difficult for a user to migrate the software from one cloud storage to another. Cloud portability methods the capability of transferring information and application components effortlessly and the reuse of those information and components irrespective of the cloud provider, operating system, storage format or APIs. There can be numerous reasons why a cloud user wants to move from one service provider to another. For example, locating different approach to make it value powerful, contract termination, modifications in business or technology approaches, felony problems or due to extended downtime or service failures. Cloud based storage model should be able to store large amount of data.

Essentially, there are methods for modeling cloud based data storage, particularly, NoSQL and relational databases. NoSQL allows to process large volume of data compared to relational databases. Since, the cloud providers implement different strategies for managing the data models, there are numerous NoSQL implementations. If a consumer wants to move from one cloud provider to another, it becomes difficult as the implementation varies from cloud to cloud. It could require us to begin the coding and implementation from the very beginning to accommodate the data. So, there ought to be a mechanism where only few changes are required to transfer the data.

NoSQL stands for "Not Only SQL" which is a is used because of its simplicity in design, simpler horizontal scaling and high availability (Cattell, 2011) non-relational database designed for the storage and retrieval of data. NoSQL (Sellami *et al.*, 2014). NoSQL databases can be classified into the following categories based on various data models service.

Corresponding Author: Nikitha Ramakrishnan, Department of Computer Science, Christ University, Hosur Road, Bengaluru, Karnataka, India

Key-value stores: Key-value store consists of a pair of unique key and value. The values are represented as a set of attributes. Example, Amazon DynamoDB, Microsoft Azure, Google Data Store.

Extensible record stores: Extensible record store contains tables with variable size and it is spread across multiple servers horizontally and vertically. Example, Apache Cassandra, Apache HBase, Amazon SimpleDB.

Document stores: The system consists of objects that has variable number of attributes with the possibility of having objects which are nested. Example, CouchDB, MongoDB.

Literature review: According to Calder *et al.* (2011), the portability of data between different cloud based databases is a challenging task. Every database has different structure and a unique way in which the data is stored which makes it difficult to access the data from them. In this research, the various challenges faced in the transfer of data among different NoSQL data model and the ways to tackle these problems are discussed. The framework supports three NoSQL databases, namely extensible data stores, document stores and key-value stores from different data models like Amazon SimpleDB, MongoDB and Google datastore respectively. It allows the portability of data among different NoSQL data models automatically without any need to write manual scripts to transfer the data. This framework also allows in converting the data types that are incompatible with the relative data type that are supported by the new backend. Only few lines of query is required to transfer the data from one model to another. The API used by the framework is implemented in such a way that it can be extended to support other data models in the future.

Cure *et al.* (2011) proposed a framework that adapted the integration of relational facts to a border circumstance in which each NoSQL and relational databases can be included. One essential extension is composed within the efficient answering of queries expressed over these records resources. The denormalized characteristic of NoSQL databases results in various overall performance costs for several feasible query translations. Accordingly, an optimized translation query needs to be generated so that the data can be integrated. The study proposed an access right of entry to course based mapping solution that takes advantage of the design options of each information source, combined options to deal with conflicts among various sources and a query language such that the distance among the SQL query expressed through the person and the query of the data sources are bridged. A prototype implementation is presented wherein the goal schema is represented as an arrangement of relations and which permits the integration of the most prominent NoSQL database models, particularly file and column family stores. This study used techniques to incorporate data originating from NoSQL stores and relational databases into single virtualized database.

Bansel (2015) proposed a migration framework which enhances the facts portability among different NoSQL implementations together with report, columnar and graph is presented. In thisa study, a comparative study of various methodologies that is used in the transfer of data is made. Based on the performance of different methods, the best one is chosen. The consistency of the information is ensured by standardizing the data and classifying the tiers. The study also proposed a model for the effective transfer of data among NoSQL data stores like Microsoft Azure table, MongoDB and Neo4j. A technique for the migration of information among the heterogeneous NoSQL datastores like file, columnar and graph based databases is presented in this research.

Sakr (2014) offered a solution for permitting portability among column family databases and graph databases as cloud databases with the aid of providing layout styles. The study gave a formal manner for migrating information among HBase as a column own family database to Neo4j as graph database. HBase database does not support Foreign key but in some table' s layout the designer use a column as a foreign key. Accordingly, design pattern that transfers records from HBase to Neo4j wishes to recognize those columns. The capability of graph databases do not equal of column family databases. In result for migrating information shape Hbase to Neo4j the quantity of data is a problem. For destiny work clustering set of rules will be implemented to 2 dimensional patterns as a way to reach the pleasant end result because in graph databases relationships and nodes have many residences, so these kinds of data must be stored in a column circle of relatives or a column. For gaining the fine end result those information needs to be labeled in an appropriate column family or column.

Shirazi *et al.* (2012) says that one of the essential for web hosting the database tier of software packages in cloud environments. Hard and fast of novel challenges that were brought on by using the reliance on cloud computing platforms and faced through utility developers and designers of cloud database systems and mentioned opportunity research directions for tackling them are mentioned.

Wu *et al.* (1997) proposes a gateway-free system called the Butterfly Methodology where the migration of data from legacy system to any target system is made easier.

DeCandia *et al.* (2007) presents a highly available key-value storage system called dynamo that a number of Amazon's core services use to produce "always-on-experience". Dynamo lacks consistency in certain conditions in order to achieve high availability. To provide a better interface for developers, object versioning and a method to assist the conflict in application is used.

Chang *et al*. (2006) describes megastore's linguistics and replication formula. Megastore combines the quantifiability of NoSQL datastore and conventional RDBMS system. It provides a storage system to satisfy the wants of today's online services.

Brad proposes a Windows Azure Storage (WAS) cloud storage system gives the customer the flexibility to store any number of knowledge at any time. WAS has different datatypes like blobs, tables and queues. This helps the customers to access the information at any point of time from any place.

MATERIALS AND METHODS

Problem statement: The difficulty in migrating the software from one specific cloud based storage to another is due to the use of different data models that introduce the diversity in cloud services. There are two ways in which the data is stored in the cloud, namely NoSQL and relational databases. The traditional Relational Data Base Management System (RDBMS) suffers from poor performance and also it does not support horizontal scaling. On the other hand, the NoSQL database acquires the ability to under take large volume of data thereby eliminating the problems that the traditional RDBMS system exhibits. Hence, the data portability across different cloud based data stores needs to be attained which is cost effective.

There are many frameworks that helps in easy transfer of data from relational to different types of NoSQL databases, one NoSQL type to another, columnar database to graph database and so on. This study proposes a system that helps in converting relational database to key value store.

Framework: The main goal of the proposed system is to help the software developers to easily transfer the data from one application to other across various cloud databases without the need for the particular application to be re-engineered. This system allows in migrating data from relational database model to key-value store and vice versa. The data model and the cloud storage service interface can be adapted easily. The framework supports relational database services for its implementation. An adapter is implemented which hides the implementation details. The request from the user application which is on the framework's API is received by the adapter and is converted to the model which is supported by the database. The users of the proposed system need not know how the system works or how the operations are performed. The proposed framework enables one to either minimize code modification or eliminate the need to modify the code at all. The adapter ensures that the data model translation occurs according to the specified data model which is supported by the system and hence the users need not deal with it manually. The proposed data models helps in unifying the interfacing concepts with the NoSQL model and relational database by depending on a simple, general and standardized abstraction. Certainly, the data objects of the data model are represented as entities that consist of these.

Key: It is a unique identification attribute by which the entity type can be identified. It can be either integer data type or string or object type, so that compatibility with different databases is possible.

Property: It is a key-value attribute that describes the entity which can be string, long, Boolean or array.

RESULTS AND DISCUSSION

The evaluation is done using the following dataset given in Table1. Consider a relation schema R of degree n which is denoted by R(A_1, A_2, ..., A_n) where A_1, A_2, ..., A_n are attributes. An attribute A can be qualified with the relation name R to which it belongs using the dot notation R.A (Elmasri and Navathe, 2010). This is because the same name is used for two attributes in different relations. All attribute names in a particular relation must be distinct. The state of the whole database will correspond to the states of all its relations at a particular point in time. A university database that consists of 4963 students is taken as the data set.

Table 1 shows the university data set that was considered for the research. Department is a table in the database which contains 256 rows. The university has a total of 4936 students and 466 teaching staffs. All these data is stored in relational database model.

In order to evaluate the system, questionnaire was distributed to five different programmers which consisted of five questions. The following choices were given to the programmers for each of the question:

- Strongly disagree
- Disagree
- Neutral
- Agree

They were asked to rate, "how convenient the framework is to work with and how easy is it to transfer the data from relational model to key-value store". A statistical analysis was performed after gathering the data.

Table 1: Dataset

Name	No. of attributes	No. of rows
Department	4	256
Students	5	4963
Staff	5	466

Table 2: Student

Roll No.	Name	Age	Marks	Course
122015	Joseph	23	87	MCA
122016	Kumar	22	79	MCA
122017	Lisa	23	84	MCA
122018	Lenin	24	92	MCA

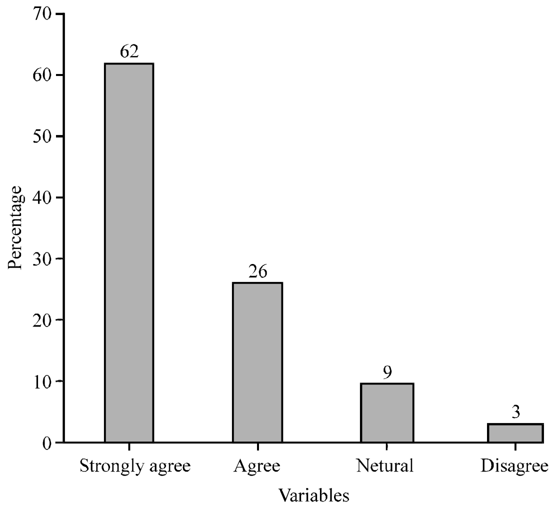

Fig. 1: Statistical analysis

About 62% of the people strongly agreed that the proposed API helps in the portability of data from relational model to key-value pair model without the need to rewrite the entire code and 26% agreed to that. On the other hand, 9% remained neutral and 3% disagreed.

Figure 1 depicts the graphically representation of the statistical analysis that is conducted. It is very clear from the figure that the proposed framework allows the portability of data from relational model to key value pair without much effort.

Table 2 depicts student table which is stored in relational database model. Student details like roll no, name, age, course and marks are present in Table 2. Using the framework the above data is transformed to key-value store as follows algorithm.

Key-value algorithm:

```
    {RollNo ->122015, Name->Joseph, Age->23, Mark->87,
Course->MCA}
    {RollNo ->122016, Name-> Kumar, Age->22, Mark->79,
Course->MCA}
    {RollNo ->122017, Name-> Lisa, Age->23, Mark->84,
Course->MCA}
    {RollNo ->122018, Name-> Lenin, Age->24, Mark->92,
Course->MCA}
```

The framework allows us to convert the data from relational model to key-value store without making many changes to the code automatically and hence there is no need to manually deal with the transformation of data.

CONCLUSION

The portability of data among different cloud based databases is a challenging task when a user wants to move from one service provider to another. Every other cloud platforms use different data models to store the data. The accessing of data from NoSQL database can also be done in different ways. In this study, a framework is proposed in order to handle the data portability issues. This framework allows us to transfer the data from relational database model to key-value store. There is an adapter which ensures that the data model translation occurs according to the specified data model that is supported by the system and hence the users need not deal with it manually. The proposed framework enables one to either minimize code modification or eliminate the need to modify the code.

RECOMMENDATION

For the future, the framework can be used to implement other database models like columnar and document.

REFERENCES

Bansel, A., 2015. Cloud based NoSQL data migration framework to achieve data portability. Ph.D Thesis, National College of Ireland, Dublin, Republic of Ireland.

Calder, B., J. Wang, A. Ogus, N. Nilakantan and A. Skjolsvold *et al*., 2011. Windows azure storage: A highly available cloud storage service with strong consistency. Proceedings of the 23th International ACM Symposium on Operating Systems Principles, October 23-26, 2011, ACM, New York, USA., ISBN:978-1-4503-0977-6, pp: 143-157.

Cattell, R., 2011. Scalable SQL and NoSQL data stores. ACM. Sigmod Rec., 39: 12-27.

Chang, F., J. Dean, S. Ghemawat, W.C. Hsieh and D.A. Wallach *et al*., 2006. Bigtable: A distributed storage system for structured data. Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation, November 6-8, 2006, Incline Village, Nevada, USA., pp: 205-218.

Cure, O., R. Hecht, L.C. Duc and M. Lamolle, 2011. Data Integration Over Nosql Stores Using Access Path Based Mappings. In: Database and Expert Systems Applications, Hameurlain, A., S.W. Liddle, KD. Schewe and X. Zhou (Eds.). Springer, Berlin, Germany, pp: 481-495.

DeCandia, G., D. Hastorun, M. Jampani, G. Kakulapati and A. Lakshman *et al.*, 2007. Dynamo: Amazon's highly available key-value store. Proceedings of the Symposium on Operating Systems Principles, October 14-17, 2007, Stevenson, WA., pp: 205-220.

Elmasri and Navathe, 2010. Fundamentals of Database Systems. 6th Edn., Addison-Wesley, San Francisco, California, ISBN-13:978-0-136-08620-8, Pages: 1172.

Sakr, S., 2014. Cloud-hosted databases: Technologies, challenges and opportunities. Cluster Comput., 17: 487-502.

Sellami, R., S. Bhiri and B. Defude, 2014. ODBAPI: A unified REST API for relational and NoSQL data stores. Proceedings of the 2014 IEEE International Congress on Big Data (BigData Congress), June 27-July 2, 2014, IEEE, France, ISBN: 978-1-4799-5057-7, pp: 653-660.

Shawish, A. and M. Salama, 2014. Cloud Computing: Paradigms and Technologies. In: Inter-Cooperative Collective Intelligence: Techniques and applications, Xhafa, F. and N. Bessis (Eds.). Springer, Berlin, Germany, ISBN:978-3-642-35016-0, pp: 39-67.

Shirazi, M.N., H.C. Kuan and H. Dolatabadi, 2012. Design patterns to enable data portability between clouds' databases. Proceedings of the 2012 12th International Conference on Computational Science and its Applications, June 18-21, 2012, IEEE, New York, USA., ISBN:978-1-4673-1691-0, pp: 117-120.

Wu, B., D. Lawless, J. Bisbal, J. Grimson and V. Wade *et al.*, 1997. Legacy system migration: A legacy data migration engine. Proceedings of the 17th International Conference on (DATASEM'97), October 12-14, 1997, Yale University, New Haven, Connecticut, pp: 129-138.