Database Team Project: First Draft

MSDS7330 summer 2019

Pankaj Kumar & Samuel Kadyebo

[kumarp@mail.smu.edu](mailto:kumarp@mail.smu.edu), [skadyebo@smu.edu](mailto:skadyebo@smu.edu)

# RDBMS NoSQL Hybrid Database Plan With Travel Industry Focus

**Abstract**—A recent study from the Center of Government Excellence at John Hopkins University (GovEX) has found that “By far, the most prevalent type of data is unstructured data, with only a very small percentage of data produced being truly structured. Even semi-structured data (ie data in CSV, JSON, or XML formats) accounts for a small percentage of the data that’s produced. Unstructured data is difficult to easily parse, process, store, and retrieve.” This cited GovEx finding is in sync with what is happening in the current Big Data world whereof the increasing amount of data that need to be processed in a company are affecting the performances of the database ecosystem system generally. The Relational Database Management System (RDBMS) is a commonly used database system, but which is also known to having a decreased performance when handling high amount of data. NoSQL is evolving as an alternative storage option that is different than RDBMS and known for faster performance when handling high amount of data. For team project, we want to contribute to the understanding of a viable Big Data Database ecosystem system solution when we compare the performances between RDBMS MySQL and a hybrid model of MySQL and NoSQL MongoDB while being used on a web application. We implement a simple Travel Repository Prototype website to evaluate the performance of using different database models. The website is equipped with travel related profiling data in/with viewable features. Experiments are performed using randomly generated user profiles and repository data. Based on our evaluations, we seek to answer the question of whether the hybrid database of MySQL and MongoDB achieves higher read and write performances than the MySQL alone. Meanwhile MySQL can handle more sensitive data and maintaining data consistency. The hybrid model database MySQL and MongoDB uses less disk space than MySQL alone, but uses more RAM. We have also observed that there is not much difference in CPU usage for both database models.

**Keywords**—MySQL, MongoDB, CQRS, hybrid database, REST, web application, database performance

1. INTRODUCTION:

The usage of a database ecosystem system is a common thing in development of a production system. According to Paradaens et al., a database system is a collection of programs that run on a computer and help the user to collect, change, protect, and manage information [1]. In the instant case, a Travel Repository is a repository of information related to places from point of the view of the people interested in travelling to a location. The Relational database model is widely used in the last 30 years and thus it is common for a company to use the relational database for the database in their system. The commonly used Relational Database Management System (RDBMS) is MySQL which is recorded to be the most used data storage for web applications because of its speed in reading data. The popularity of MySQL is also supported by its low implementation cost and it is usually free and an open source project.

In 2009, a new method of storing data was introduced, which is called NoSQL (Not Only SQL). The NoSQL is first used in 1998 by Carlo Strozzi. Then, Eric Evans reintroduced the NoSQL in 2009 [2]. The major difference between NoSQL and RDBMS is that the NoSQL does not need a fixed table scheme like RDBMS. There are few types of NoSQL system such as key-value, document, graph, and multi-model [3]. The key-value type stores data as a ‘key’ and a ‘value’ for each of the data. The document type stores the data as a ‘document’ where different documents can contain different data within the same collection. The graph type stores data as a ‘graph’ that interconnected to each other. Meanwhile, the multi model is a type of NoSQL that uses multiple methods to store data, such as a combination of document and graph. The commonly used NoSQL are MongoDB and Cassandra. MongoDB is one of the NoSQL that uses document system.

Although RDBMS have a good reputation, RDBMS tends to have decreased performance with the increasing Big Data ecosystem data usage in the system. The replication of the relational database can be limited. The relational databases are also made based on consistency rather than availability [4]. NoSQL, on other hand, has a different reputation. It is known to be good at storing high amount of data. The scheme free characteristic of NoSQL makes the scalability of the database better. However, these advantages come with sacrifices, such as weaker security or lesser data consistency.

The increasing amount of data that needs’ to be processed in a company are affecting the performances of the company database systems. Hardware upgrades are generally also needed to make the system performance back to its expected level [5]. But the other more important effect of the technology advances is Big Data ecosystem. The big data which is more common to be found in big companies, means that managing the big data can be a key in competing with other companies [6]. Therefore, we propose a hybrid database system of MySQL and MongoDB to improve the performance of a web application by harnessing the capabilities of the RDMS and NoSQL database models. The ability of the RDBMS in storing sensitive data complements the ability of NoSQL in processing high amount of data. The proposed hybrid database system is implemented using MySQL for relational database and MongoDB for NoSQL.

II. RESEARCH METHODOLOGY:

Our motivation for this hybrid database project has been buttressed with some *use-case* reviews like the Solving MySQL Schema Migration Pain: Craigslist *whereof* their archive utilizes MongoDB while live listings are stored on a MySQL database [7], and Right Tool for the Job: E-commerce at OpenSky *whereof* the catalog takes advantage of MongoDB’s flexibility while the order management is stored with MySQL[8]. When it comes to the travel planning for any purpose, we have some common questions in our mind. We want to know a lot of about the place we plan to visit. There are some basic things that are associated with all the places. For example, weather, infrastructure, places of attractions, budget for comfortable stay distance from where someone wants to travel.

These types of information are the ones which have a common rigid structure and a relational schema seems to fit this requirement perfectly.

Along with this common structured information every place also has some very specific information that people are interested in. Sometimes we discover facts after visiting the place and sometimes we know the facts and the people might be more curious or interested in exploring a place after discovering such facts. For example, some places might be famous for historical significance and some for natural beauty. There are further levels of details like its background, associated legends etc. Similarly, there are different cultural aspects associated with which place that are unique to that place.

One challenge while trying to capture second type of information in traditional database design is that we can have a rigid structure that applies to all places. If we plan to have broad categories, we might end up having either a lot of them or missing some and fitting them in something similar on expense of exactness. Other problem is could be sparsity of data. Though we can model a relational schema for such data but schema on read approach is better suited.

Following are the issues we are the requirements we are trying to meet:

### Primary:

1. Given a location we should be able to perform queries on its basic details like location, economy attractions
2. We should be able to get information like historical, tourism , review and other info in unstructured format.

### Secondary :

1. Given a source and destination we should be able to query for optimal routes between those in terms of distance , fare or other criteria.
2. Given a source and list of destination we should be able to query for better choice for travel based on some criteria like total cast , number of hours that can spent. Etc.

## Example Data storage (Entity sets):

Destinations. Distance matrix. Route information. Budget. Attractions.

## Data Storage

Since unstructured data can almost include anything. Including images posted in reviews and video etc. We have might have to deal with data that can grow big in terms of volume and variety ( 2 of 3 vs of big data). Our choice for such data is a nosql database which is mongodb. We may also include some continuously changing information related to a location that are of interest to any traveler. This might include events , whether forecast, or other new establishments. We plan to keep our database scalable for such data even if we are not implementing these features now.

For structured information we’ll use mysql as relational database.

|  |  |  |
| --- | --- | --- |
| **Type** | **DB** | **Language** |
| Structured Info | Relational DB: mySQL | Python |
| Unstructured Info | noSQL: MongoDB | Python |

## Source of data:

|  |  |
| --- | --- |
| **Type** | **Source** |
| Structured Info | Wikipedia  Google map  Corresponding government sites. |
| Unstructured Info | Travel blogs.  Website of the places/ attractions  factbooks |

## Users of data:

People managing travel agencies, tourism departments of government, people interested in visiting new places

## Growth and scalability:

We can start with repository of one specific geographical location and then follow the same pattern of data gathering and storage to build repository for other locations.

III. RESULTS

We implement a simple Travel Repository Prototype website to evaluate the performance metrics of using different database models. We look at: System Specifications, Read and Write Performance Comparison, Database Performance Comparison Based on User Profile Data, Disk Usage, CPU, and RAM Comparison, etc.

A close up of a logo

Description automatically generated

IV. CONCLUSIONS AND FUTURE WORKS:

From the experimental results and analysis, we hope to conclude whether that the hybrid database model of MySQL and MongoDB would improve the web application performance on large database size. We know that MongoDB have a relatively better write time than MySQL; but MySQL write procedure is more consistent. In term of disk space, CPU and RAM usage, we evaluate that the hybrid model database MySQL and MongoDB uses less disk space than MySQL. But that it comes with a price of higher RAM requirement. We also want to see whether that there is not much difference in CPU usage for both the using MySQL and hybrid database. Our theory is also that the scheme free of MongoDB need to be considered when using MongoDB in hybrid model database with MySQL, in order to avoid system failure cause by the less consistent data from MongoDB, and further that the Hybrid database model of MySQL and MongoDB can be used by splitting the data (CQRS). Sensitive and potentially duplicate data can be stored to MySQL, while high amount of small data, with high read or write traffic, are stored to MongoDB. We use this mechanism in hybrid database ensure maximum performance in storing and reading data. We look at whether the implementation of hybrid model database of MySQL and MongoDB, there are more complexity compared to MySQL system, and further that the cost of the implementing need to be considered by comparing them with the benefit that can be received. With the current evaluations being performed on a single system, we suggest repeating the evaluation through experiments on another operating system and different web framework in future works. We do find thus far that further evaluation on the performance of the proposed hybrid database on bigger and more complex data is recommended for more conclusive results.

REFERENCES:

[1] J. Paradaens, P. D. Bra, M. Gyssens and D. V. Gucht, The Structure of the Relational Database Model, Heidelberg: Springer-Verlag Berlin Heidelberg, 2012.

[2] A. Lith and J. Mattsson, "Investigating storage solutions for large data - A comparison of well performing and scalable data storage solutions for real time extraction and batch insertion of data," Unpublished master's thesis, 2010.

[3] R. Cattell, "Scalable SQL and NoSQL Data Stores," SIGMOD Record, vol. 39, no. 4, pp. 12-27, 2010.

[4] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall and W. Vogels, "Dynamo: Amazon's Highly Available Key-Value Store," SIGOPS Operating Systems Review, vol. 41, no. 6, pp. 205-220, 2007.

[5] R. Hect and S. Jablonski, "NoSQL Evaluation: A Use Case Oriented Survey," in International Conference on Cloud and Service Computing, Washington DC, 2011.

[6] J. Manyika, M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh and A. H. Byers, Big Data: The Next Frontier for Innovation, Competition, and Productivity, San Fransisco: McKinsey Global Institute, 2011.

[7] Craigslist, https://www.theserverside.com/feature/How-NoSQL-MySQL-and-MogoDB-worked-together-to-solve-a-big-data-problem

[8] Opensky, <https://www.mongodb.com/blog/post/hybrid-deployments-mongodb-and-mysql-3-examples>