Hybrid Database System of MySQL and MongoDB in Web Application Development

Gregorius Ongo
Computer Science Department,
BINUS Graduate Program - Master of Computer Science
Bina Nusantara University
Jakarta, Indonesia 11480
gregorius.ongo@binus.ac.id

Gede Putra Kusuma
Computer Science Department,
BINUS Graduate Program - Master of Computer Science
Bina Nusantara University
Jakarta, Indonesia 11480
inegara@binus.edu

***Abstract*—The increasing amount of data that need to be processed in a company are affecting the performances of the database system. Relational Database Management System (RDBMS) is a commonly used database system, but also known to having a decreased performance when handling high amount of data. NoSQL is an alternative storage option that is different than RDBMS and known for faster performance when handling high amount of data. In this contribution, we compare the performances between RDBMS MySQL and a hybrid model of MySQL and NoSQL MongoDB when being used on a web application. We implemented a simple social media website to evaluate the performance of using different database models. The website is equipped with chat and profile view features. Experiments are performed using randomly generated user profiles and chat data. Based on our evaluations, 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 also observed that there is not much difference in CPU usage for both database models.**

Keywords—MySQL, MongoDB, hybrid database, web application, database performance

I. Introduction

The usage of a database system is a common thing in development of a 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]. Relational database is widely used in the last 30 years. 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) are MySQL, Oracle, and PostgreSQL. Each RDBMS is well known and have their own superiority. MySQL is known for its faster query execution than the others RDBMS. Even though PostgreSQL have a relatively slower performance than MySQL, PostgreSQL have more features that can help user in managing data. Oracle is usually picked when the user need to handle big and complex data. However, Oracle is more expensive to be implemented than the other databases. On the other hand, MySQL 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 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 made 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 need to be processed in a company are affecting the performances of the database system. Hardware upgrades are also needed to make the system performance back to its expected level [5]. The other effect of the technology advances is big data. The big data is more common to be found in big companies, and managing 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. RELATED WORKS

Hybrid database model is a database system that uses two or more different database models in a system [7]. This model is preferred because sometimes the stored data is not always suitable with the database model being used. Therefore, Cook tried to combine two databases that have different models: object oriented database and relational database [8]. From the experiments, Cook founds some benefits in using multiple database models in a system, such as:

- Flexibility. In complex data types, the data may need to be forced to fit the storage model that is being used in the system. Hybrid database model can increase the flexibility of the system to store different types of data.
- Increased performance. Leveraging the strengths of multiple storage models improves the system performance.
- Logical distribution. The distribution of data is better and the number of forced data is reduced due to the flexibility of hybrid database model.
- Built for the web. The increased performance and the flexibility on storing the data can enhance the experience on web usage by the user.

Hybrid database is not a solution for every problem, but it is an option in making a database that can store normalized data and can run aggregated query with good performance [9]. For example, an e-commerce company can use MongoDB to store the company catalogue and use MySQL to store the transactions. As a result, the platform performance is increased and the codebase is simplified.

III. RESEARCH METHODOLOGY

In this contribution, we implemented a simple social media website to evaluate the performance of using different database models. The website is equipped with features: chatting between users and profile view. This simple social media website is chosen as a test case to showcase the performances of MySQL only and Hybrid of MySQL and MongoDB database models in handling mixed of structured and unstructured data, which is commonly found in web applications.

Two identical websites are built in PHP using CodeIgniter framework in Windows 10 operating system. GUI manager for the database are also employed. HeidiSQL is used to manage MySQL and RoboMongo for MongoDB. Both websites are identical in all aspects except for their database model. The first website is using MySQL as the database, while the second website is using hybrid database of MySQL and MongoDB.

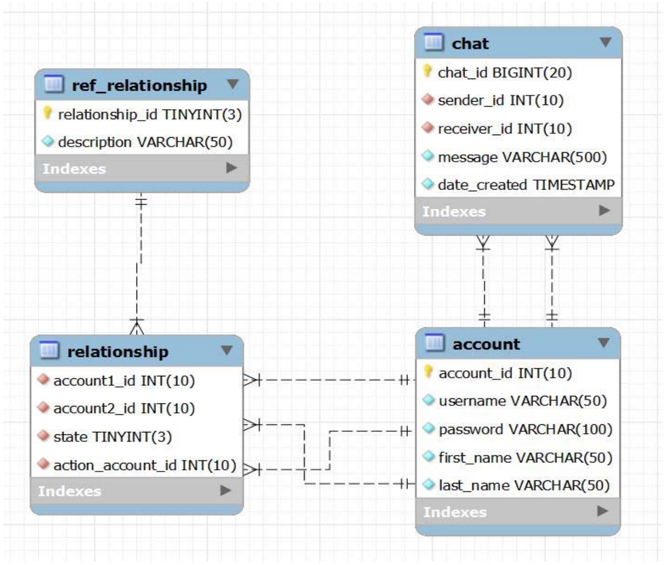


Fig. 1. MySQL database design.

The MySQL database design, as shown in Fig. 1, has four tables: account, chat, relationship, and ref_relationship tables. The account table is used for storing account data. In our experiment, 1000 user accounts are created randomly. The account_id is the primary key of the account table. The relationship table is a table that stores relationship data between users, such as friend request, friend, and block. Account1_id and account2_id will be used as index, and foreign key that refer to account_id of account table. The ref_relationship table will store the description of the state from the relationship table. Chat table will stores all the user chat data. Chat_id is the primary key of this table. Sender_id, and receiver_id will be used as foreign key that refer to account_id from account_table.

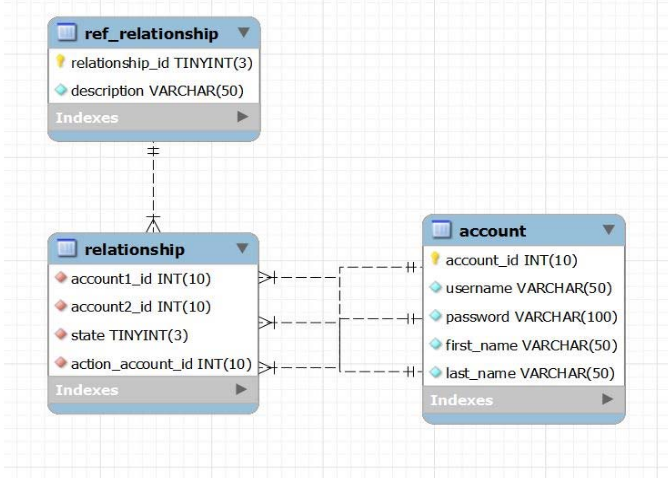
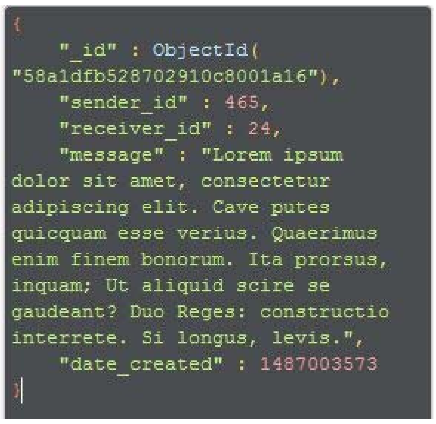


Fig. 2. MySQL design for hybrid database.

The MySQL design for hybrid database is shown in Fig. 2. It is similar to the MySQL design for the MySQL only system in Fig. 1. Please note that there is no chat table in the MySQL on hybrid database. In hybrid database model, the chat data will be stored in MongoDB.



```
{
    "_id" : ObjectId(
"58a1dfb528702910c8001a16"),
    "sender_id" : 465,
    "receiver_id" : 24,
    "message" : "Lorem ipsum
dolor sit amet, consectetur
adipiscing elit. Cave putes
quicquam esse verius. Quaerimus
enim finem bonorum. Ita prorsus,
inquam; Ut aliquid scire se
gaudeant? Duo Reges: constructio
interrete. Si longus, levis.",
    "date_created" : 1487003573
}
```

Fig. 3. MongoDB chat data structure.

The chat data structure for MongoDB is shown in Fig. 3, which consist of _id, sender_id, receiver_id, message, and date_created. The _id will store the data id in MongoDB. The sender_id store the id of the chat sender, and the receiver_id will store the id of the chat receiver. Message will store the chat message. Date created will store the UNIX timestamp when the data is made.

In this structure, _id, sender_id, and receiver_id will be used as indexes. The data from Fig. 2 and Fig. 3 are combined in MySQL and MongoDB to make a hybrid database model for the second website. In this research, the response time data will be collected using CodeIgniter benchmarking tool. While, RAM data will be collected using CodeIgniter benchmarking tool and process explorer. The databases disk usage will be collected using the GUI manager for the corresponding database, HeidiSQL for MySQL and RoboMongo for MongoDB.

Besides the system performances, an analysis on how hard the implementation of the hybrid database will be compared to implementation of MySQL only. The analysis includes how much backend code need to be changed so the system can use two different databases at the same time, and is there any library that can support the usage of the database on the framework that is used to build the website.

IV. Experiments

A. System Specifications

The entire experiments are performed on hardware with the following specifications:

• Processor	: Intel I5-3570K 3.4 GHz;
• RAM	: 8 GB;
• Input device	: Keyboard, mouse;
• Monitor	: Yes;
• HDD	: 20 GB.

B. Read and Write Performance Comparison

We randomly generated chat data between users of the website for the experiments. There are three datasets being used in this experiment, which are summarized in Table I. We evaluate the read and write performances of two websites with different database models. The first website uses MySQL database and the second web uses hybrid database of MySQL and MongoDB. For each dataset, the read and write response times of the two websites are collected 10 times. Their read and write performances are then measured by the average value of their response times.

TABLE I. The three chat datasets for experiments.

Datasets	Size
First Dataset	100,000 data
Second Dataset	500,000 data
Third Dataset	1,000,000 data

The average read response times of the two database models on different dataset sizes are shown in Table II. For the first dataset, there is not much difference on the read response times between the two websites. The average read response time for MySQL website is 0.0205 second and for website that use hybrid database is 0.0226 second. The difference between them is only 0.0021 second, where MySQL is a little bit faster than MongoDB.

TABLE II. Summary of average read response times.

Database Model	Average Read Response Time (s)		
	First Dataset	Second Dataset	Third Dataset
MySQL Database	0.0205	0.5589	2.2938
Hybrid Database	0.0226	0.0311	0.0430
Response speed-up:	-0.0021	0.5278	2.2508

In the second dataset, the experimental data consists of five hundred thousand chat data. As shown in Table II, the average read response time for website that use MySQL is 0.5589 second, and for the website that use hybrid database is 0.0311 second. There is a significant increase in MySQL read response time compared to the Hybrid database read response time. The website with Hybrid database achieved a read response speed-up of 0.5278 second compared to the website using MySQL. From results of first and second datasets, we can also see an increment in the average read response time by 0.5384 for the website that uses MySQL. On the other hand, the website that use hybrid database only suffers 0.0085 second increase in response time.

On the third dataset, the website that use MySQL has average read response time of 2.2938 second and for the website that use hybrid database have an average read response time of 0.0430 second. The gap between their read response time is now widen to 2.2508 second. Both websites suffer from higher read response times for larger dataset sizes. However, these increments are more significant on the website using MySQL database compared to the website using hybrid database.

Besides the read response times, the write response time for both websites are also analyzed. We measured the time needed to insert 10 new chats on existing datasets of both websites.

The sizes of the existing datasets on the website are varied according to the dataset sizes in Table I. The average write response times of the two database models on different dataset sizes are summarized in Table III.

TABLE III. SUMMARY OF AVERAGE WRITE RESPONSE TIMES.

Database Model	Average Write Response Time (s)		
	First Dataset	Second Dataset	Third Dataset
MySQL Database	0.3166	0.4313	0.5500
Hybrid Database	0.0165	0.0166	0.0310
Response speed-up:	0.3001	0.4147	0.5190

As shown in Table III, there is increasing average write response times when storing new chat data on larger dataset sizes. For the website with MySQL database, the average write response time is 0.3166 second to save 10 new chat data to the first dataset, 0.4313 second to the second dataset, and 0.5500 second to the third dataset. On the other hand, for the website with hybrid database, the average time to save 10 new chat data is 0.0165 second to the first dataset, 0.0166 second to the second dataset, and 0.0310 second to the third dataset. MySQL and hybrid database have a steady increase in average write response time with larger dataset sizes. However, MySQL database shows a higher increase in the average write response times for inserting new chat data compared to hybrid database. Also, the average write response time for the MySQL database is constantly higher than the hybrid database.

The increasing response time in writing new data in MySQL can be caused by the static structure of MySQL database. Each stored data needs to be checked for consistency. The larger the size of the database, the longer it takes to check for consistency. In hybrid database that use MongoDB, the data is stored directly, and the data checking can only be done in the website application level. Even though MySQL have longer response time in writing the data, the MySQL procedure is more secure because it keeps the integrity of data better than MongoDB.

C. *Database Performance Comparison Based on User Profile Data*

Another analysis is performed to compare the performance of MySQL and hybrid databases in storing sensitive data, such as user profile data. There is no significant difference found on the read response time when showing the profile page. However, MySQL database has some features that can support on storing sensitive data. Features comparison between MySQL and MongoDB are shown in Table IV.

TABLE IV. FEATURE COMPARISON BETWEEN MYSQL AND MONGODB DATABASE.

Features	MySQL	MongoDB
Data scheme	Schema-free	Yes
Triggers	No	Yes
Foreign Keys	No	Yes
Transaction Concepts	Document-level transaction	ACID

From Table IV, we can see that MongoDB does not have foreign key, so normalization cannot be done in MongoDB. As a result, there will redundant data on MongoDB. Normalization on MySQL can help to reduce redundant data and hence reduce the database size. Furthermore, MySQL can keep the consistency and the durability of the data with the table scheme, transaction feature, and the write concept of 'all or nothing' in ACID. These features can help to minimize the chance of error happening in the system because of the changing data type. Therefore, MySQL is used to store the profile data in our hybrid database model.

D. *Disk Usage, CPU, and RAM Comparison*

An experiment is performed to evaluate the disk usage of the websites with a different database model when storing chat data of different sizes. The disk usage of the two websites using MySQL and hybrid databases are summarized in Table V.

TABLE V. SUMMARY OF DISK USAGE.

Database Model	Disk usage (MB)		
	First Dataset	Second Dataset	Third Dataset
MySQL Database	32.10	149.30	289.90
Hybrid Database	23.35	131.64	263.35
Difference:	8.75	17.36	26.55

There is not much difference in disk usage between the MySQL and hybrid database. As shown in Table V, the disk usage of hybrid database is 37.5% smaller than MySQL database for the first dataset. While for the second and third datasets, the hybrid database is 12.91% and 10.08% smaller than MySQL database respectively. The percentage of disk size difference become smaller as the size of the dataset become larger. Larger disk usage of the MySQL database may due to fixed data type being used. This result also indicates that NoSQL database model such as MongoDB is more efficient in storing unstructured data than SQL database.

Based on our observations, there is not much difference in CPU usage between the two websites. Even though the hybrid database uses more applications than the MySQL, the CPU usage is only slightly larger. However, the hybrid database uses more RAM than the MySQL database because hybrid database has more applications that need to stand by to process data when needed. Hybrid database uses around 10 to 17 MB more RAM than MySQL.

V. CONCLUSIONS AND FUTURE WORKS

From the experimental results and analysis, we can conclude that the hybrid database model of MySQL and MongoDB improves the web application performance on large database size. The 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, hybrid model database MySQL and MongoDB uses less disk space than MySQL. But it comes with a price of higher RAM requirement. There is not much difference in CPU usage for both websites using MySQL and hybrid database. 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. Hybrid database model of MySQL and MongoDB can be used by splitting the data. 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. This mechanism in hybrid database ensure maximum performance in storing and reading data. In the implementation of hybrid model database of MySQL and MongoDB, there are more complexity compared to MySQL system. The cost of the implementing need to be considered by comparing them with the benefit that can be received.

The current evaluations are performed on a single system, we suggest repeating the evaluation through experiments on another operating system and different web framework in future works. 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] G. Powell, Beginning Database Design, Indianapolis: Wiley Publishing, 2006.

[8] R. Cook, "Is a hybrid database in your future?," SunWorld, February 1997. [Online]. Available: http://sunsite.uakom.sk/sunworldonline/swol-02-1997/swol-02-objects.html. [Accessed 23 February 2017].

[9] J. Madison, "Building a Hybrid Data Warehouse Model," Oracle, April 2007. [Online]. Available: http://www.oracle.com/technetwork/articles/madison-models-086845.html. [Accessed 23 February 2017].