NoSQL Database Implementation Case Study in Thai Tech company, Pantip

Maneerat Wongjaroenporn
Heinz College of Information Systems and Public Policy
Carnegie Mellon University
Pittsburgh, Pennsylvania 15232
Email: m.wongjaroenporn@gmail.com

Abstract—NoSQL database has been implemented more and more in recent years. Its scalability and flexibility have attracted many companies in every industry to migrate from traditional relational database management system (RDBMS) to the NoSQL database. This paper will analyse the NoSQL database implementation, specifically MongoDB, in a Thai Tech company, Pantip in 2013, discuss the problem that the company had encountered from sharding and suggest alternative approaches that Pantip might choose from the current technology setting such as Postgres, Cassandra, or Redis.

Keywords—MySQL, NoSQL, MongoDB, Thai Tech Company, Database Migration, Shard Key, MongoDB Sharding

I. Introduction

In industry, there are several approaches to scale the database system after the company has grown to a certain point. It's not only about how much storage the company has, but also how quickly and accurately we can retrieve the data from the database.[7]

Pantip.com was established in 1996 as a community forum for technical discussion. In 2011, Pantip has redesigned the whole website and the number of users grew exponentially from 800,000 users to approximately 5 million users per day within 2 years. This massive growth led to the consideration of changing the database management system. The big technology companies such as Facebook, Amazon, Google ,and Netflix are all using NoSQL because of their dependence on large data volume to grow exponentially.[2] Therefore, the company decided to migrate from MySQL to MongoDB in 2013. [17]

This paper will mainly discuss about MongoDB implementation in Pantip case in 2013, their configuration issue ,and the alternative approaches for their system in 2020.

II. Fundamental Concept on MongoDB Sharding

Sharding, sometimes referred to as horizontal partitioning, is a common technique to horizontally scale the data storage and cache for the database system. Each set of data, named shards, will be assigned to the different chunk. [14] Sharding is largely used to support deployments with gigantic data sets and operations with high throughput. [11] In MongoDB, sharding happens at the collection level. Even though it provides and a lot of benefits in scaling, it is also recognized as one of the most complex features provided by MongoDB to get comfortable with because picking the right shard key with a sharding strategy is very tricky.

A. Sharding Strategy

There are two main sharding methodologies: range sharding and hashed sharding. Range-based sharding involves dividing data into contiguous ranges determined by the shard key values while hashed-based sharding uses either a single field hashed index or a compound hashed index as the shard key to partition data across the cluster.

B. Shard Key

MongoDB uses the shard key to chunk the data. The shard key can be either a single or compound indexed field that determines the distribution of the collection's document among the clusters. [10] The proper selection of the shard key is crucial to prevent bottleneck conditions.

C. Hot Chunk

Hot Chunk or Hot Shard is an issue from selecting a poor shard key choice. In general cases, using a monotonically increasing or decreasing shard key is the main cause.[10] This will result in all inserts targeting the single shard that currently storing the highest or lowest shard key value. Fig. 1

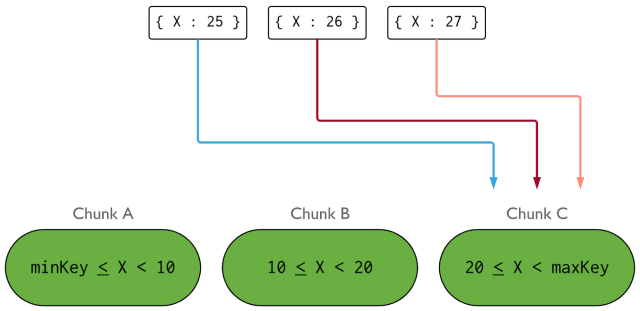

Fig. 1. A sharded cluster using the field X as the shard key. If the values for X are monotonically increasing, the distribution of inserts may look similar to the figure.[10]

D. Pre-Splitting Chunk Range

In most situations, a sharded cluster will create, split, and distribute chunks automatically without user intervention. However, MongoDB sometimes cannot create enough chunks to distribute data to support the required throughput.[9] The most common cause is the ingestion of imbalance data into the cluster, such as using monotonically increasing or decreasing shard keys. The pre-splitting technique is then introduced to minimize rebalancing. It is based on the known distribution of shard key values from the existing data in the database. The main pitfall for manually splitting chunks is unpredictable chunk ranges and sizes as well as inefficient balancing behavior.

E. Shard Zones

In a sharded cluster, we can create zones to represent a group of shards and associate one or more ranges of shard key values to that zone. It is a part of MongoDB data awareness center. This feature allows queries to be routed to particular MongoDB deployments considering physical locations or configurations of MongoDB instances.[8] Zones are useful when there is a need to isolate specific data to a particular shard.

III. Company Background

Pantip.com is the most popular discussion forum in Thailand. It's ranked 5th most-visited website, following google.co.th, youtube.com, google.com, and facebook.com, for Thai people.[1] It operates like Reddit, the biggest discussion community in the world. In 2018, There are more than 4 million active users and more than 15 million active forums daily.[18] The site is mainly populated in dynamic JavaScript and the main data request is the form of text.

Pantip used MySQL as the main database system since the website had launched until 2013. Prior to the migration to MongoDB, Pantip had one dedicated HDD to collect the interesting forum and keep it forever. The forum would be selected manually by the admin. Other forums with no update within one month will be deleted from the system forever.[4]

The main value of Pantip lies with the content creators and user views. The annual revenue is around $4 million and the main revenue stream is the advertisement on the platform which is a similar model with Reddit. Even though writing operation is important for the content creators to write new and available content, we can say that reading performance should have a higher priority because it is the main source of the business.

IV. Case Study

1) Database Migration: On September 28, 2013, Pantip.com had closed the website for 22 hours, the longest time since it had been launched, for the database migration. It had moved from MySQL to MongoDB because of the need for scalability's power in a document-oriented database. The CTO of the company has mentioned that prior to the migration the forum had the expiration date. They aimed to keep all forums in the database forever. Therefore, they decided to move to MongoDB as it has an incredible capability in both read and write requests. They also lost the trust of MySQL stability after Oracle has acquired MySQL AB. [17]

2) Problem after the migration: The company chose topic_id and comment_no as a compound shard key. Fig. 2 and decided to use range sharding. The topic_id was generated by the running number when the new forum was posted.Therefore, this choice of shard key easily led to the hot chunk problem because the new forums were likely to be more updated and the most of right chunks would be overloaded. Fig. 3

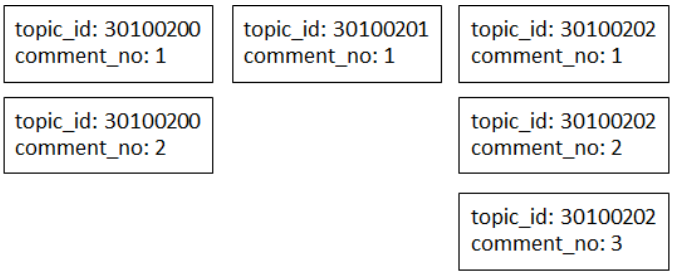

Fig. 2. Database Schema to store comments in each forum [17]

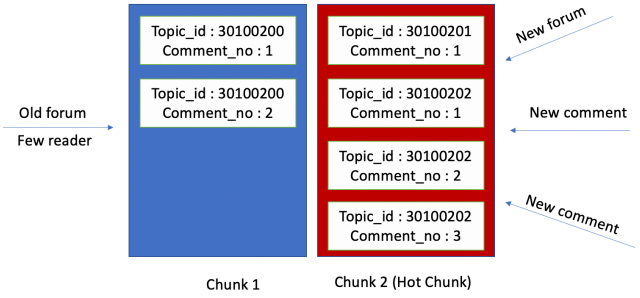

Fig. 3. Hot Chunk occurs at the right most chunks

People are generally interested in a new drama or current situation topic. Therefore, the more users commented on the new forum, the more right chunk was growing. As a result, they had to change the shard key or sharding methodology before the database became ridiculously imbalance. It was a very challenging task on the operating website. Besides closing the website, they had to back up all the data and upload everything back to the system. Fig. 4 There were around 100 million documents at the time that they realized the need of changing the sharding technique.

3) Problem after changing the sharding methodology: The company changed to a hashed-based sharding strategy which randomly assigns the topic_id and comment_id

Fig. 4. The process to change the shard key

to a different shard in stead of ranged-based sharding. This methodology can distribute the data into each shard perfectly so the write performance was really good. On the other hand, there were too many distributions and it required a scatter gather query. Sometimes it needed to go through almost 100 chunks to construct one forum, resulting in very slow read performance. Their temporary mitigation is buying a new SSD. Fig. 5[17]

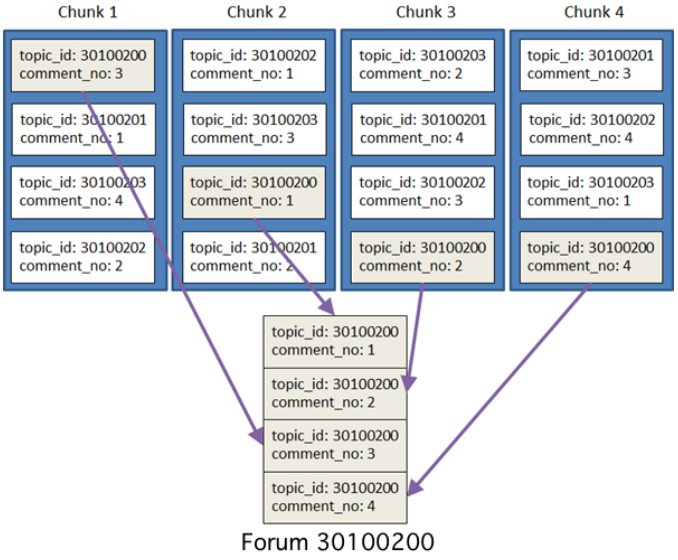

Fig. 5. Read request to construct one forum from hashed shard key

4) Mitigation from sharding problem: The company had to change the sharding method again. This time they decided to use the pre-splitting chunk range method. Fig. 6 This technique would manually assign each document to destinated chunk without waiting for MongoDB to randomly assign it. It was highly recommend when we need to load a high amount of data to the collection. [5]

V. Analysis

From the use case above, we can see that Pantip had encountered problems from selecting shard key and different sharding techniques. Currently, Pantip hasn't publicly mentioned about their current performance or any updates from the latest sharding technique. In this section, I

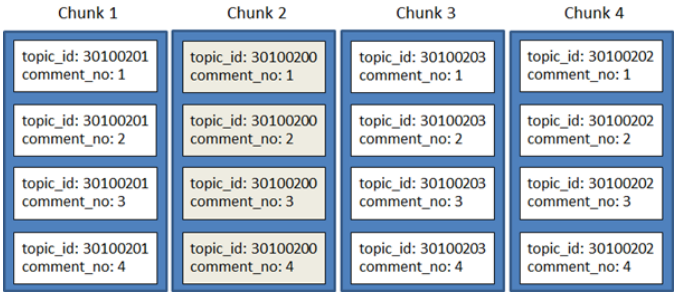

Fig. 6. Data chunk after using pre-splitting technique [17]

will analyse data modeling decision, load balancer issue, security issue and suggestion on the alternative approach.

A. Sharding Decision

MongoDB has stated in its blog that picking a good shard key is very tricky. Choosing a wrong shard key can totally trash the performance of the system's cluster and getting the wrong sharding might lead to the data migration to the new collection with a right shard strategy. [21] Pantip has unfortunately fallen into both cases. Hot shard led to poor write scalability and hashed-based sharding decision led to poor read scalability. In order to pick an appropriate shard key, there are 5 criterion to consider: cardinality, write distribution, read distribution, read targeting and read locality.

1) Cardinality: The cardinality of a shard key determines the maximum number of chunks the balancer can create.[10] Pantip should pick a shard key which can be subdivided into small ranges to avoid ending with putting too many documents in a single chunk. Pantip chose topic_id as one of the compound shard key but didn't mention about the number of cardinality. The topic_id is hard to divide into small ranges so Pantip might consider more than a combination of topic_id and comment_id. The categories of the forum can be a good candidate to determine the cardinality as every topic has to fall into one of 35 categories.

2) Write Distribution: In ideal situation, a write load should be uniformly distributed over the shards in the cluster. The most common mistake is using a monotonically increasing shard key like topic_id in ranged-based sharding which generally limits the write load scalability. Pantip can use zones in sharded clusters to help improve the locality of data and the write performance. Fig. 7

3) Read Distribution: Similar to write distribution, we also want to avoid hot shards for reads. The compound shard key that Pantip chose definitely causes the hot shard in the most left one in range sharding. A mitigation for this problem is using a high-granularity field like category_id or hash to determine the cardinality.

4) Read Targeting: The MongoDB query can perform either a targeted query or a scatter query. Pantip might choose to do range-based sharding because of the read tar-

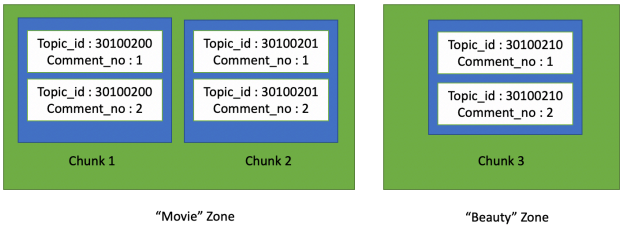

Fig. 7. Example of using zone in sharded cluster for Pantip case

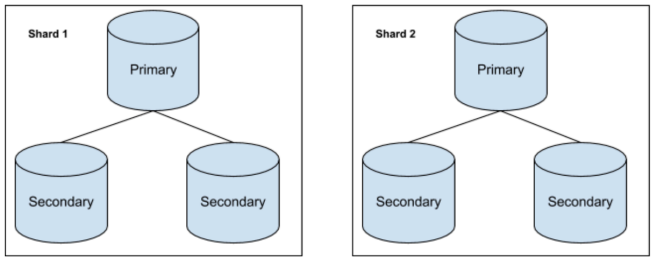

Fig. 8. The replication inside sharding. The read requests will be mainly handled by secondary server while the primary server will takes care of write requests

geting. Therefore, each chunk is mainly used to construct one forum. However, this consideration has lost when the company changed the strategy to hash sharding because the same topic_id keys which are used to construct one forum are scattered all over. Using pre-splitting chunk range was the good decision to deal with this criterion.

5) Read Locality: This criterion is important for Pantip to consider because reading the data to construct the forum on the website definitely need to use a range query. For example,"Show me the latest 20 comments on this posting" can be done easily with the current compound shard key choice. They can do a simple query "find({topic_id:301000200}.sort({comment_no:-1}.limit(20))". However, "Show me the latest 10 posting" might be hard to if Pantip didn't add the cardinality in categories type.

As it is impossible to create the perfect shard key, Pantip can mainly focus on the read distribution and read locality because of its main operation. The number of readers is definitely greater than the writer's one. Giving more priority on read performance should be a priority. Therefore, the current choice of shard key is fine but Pantip need to determine the cardinality to increase the performance.

B. Load Balancer

For any database, the load balancing of incoming requests from client is an important foundation to ensure scalability. The CTO mentioned that after they encountered the hot chunk problem and need to migrate the data collection, they had not managed the time to deal with load balancer at all. [17] MongoDB Load Balancing can be done by several approached. One of them is replication. Usually, primary server will take some time to handle the write requests and propagates the changes to secondary servers. Therefore, there is a possibility that the secondary server will return stale data as is it not in sync with the primary server.[6] Pantip.com content is not sensitive to the stale data so I think replication in sharding is suitable for manage the load balancing. Fig. 8

C. Security

There are several vulnerabilities inside NoSQL because it were not initially considered as a priority features. Moreover, sharding poses more security risks than standalone NoSQL deployment [19] These are the security issues that are related to Pantip operation and they need to be aware of.

1) Encryption: Encryption is one of the serious concern in MongoDB because the data files are never encrypted by default. There is a number of sensitive data kept in the system such as user's national identification which is used for registration in the website. Therefore, Pantip needs to make sure that the encryption has been done in the application layer before sending to the database server.

2) Authentication: When running in sharded mode, MongoDB also doesn't support authentication. If authentication has been enabled, then the databases supports two types of users: read-only and read-write[12]. In pantip.com, all users can write or create a new post and read any posts, therefore, users should have full read-write access to all of the databases defined in the cluster. In this case, Pantip can use a reverse proxy behind the RESTful API to add distinct permissions to any particular role like admin to make stronger permissions.

3) Auditing: Reddit has once mentioned that not having a proper data logging and monitoring can lead to several problems in scaling system. [3] Auditing in MongoDB only happens when a new Namespace is created but there is no logs for subsequent operations like updates or queries. The mitigation for this issue is implementing authorization feature using Kerberos of MongoDB enterprise.

4) Injection Attack: A potential attack that MongoDB has been suffered from implementing in JavaScript. Moreover, the Pantip website is deployed in dynamic JavaScript. JavaScript uses $where clause to evaluate records in collection and it is read only. Any changes in database are not possible, therefore, all input including the posting and comment should be sanitized to prevent the changes of injection. [12]

D. Alternative Approach in 2020

In 2013, MongoDB was the most suitable NoSQL database system to choose. Given that NoSQL database has become a standard part of many technology stacks, there are a number of alternative solution we can choose. In 2020, PostgreSQL, Cassandra and Redis are the top three most popular database systems after MySQL and

MongoDB. Therefore, I will analyze the practicability of these database with Pantip along with the current status of MySQL.

1) PostageSQL: PostgreSQL, also known as Postgres, is a free and open-source relational database management system that supports a certain feature commonly found in NoSQL database. One of the most common one is JSON data. It is designed to scale very well with a large number of cores at high concurrency levels and high availability. It's also good for highly structured data with relations between objects. Furthermore, it has a very strong security built-in at no extra cost.

PostgreSQL was implemented by Reddit before moving to Cassandra. Most data at Reddit is key-value and stores in Postgres, using a master-slave configuration. Reddit also did the sharding across four master databases. The read requests were managed by the slaved to keep the master dedicated to writes. However, the main difference is there is no expiration date on the comment in the old version of Reddit. [15]

Given that the data type stored in the system for Pantip is a structured data, Pantip was likely to encounter fewer problems on sharding and still enjoy the power of scalability from Postgres if they had chosen Postgres in 2013.

2) Cassandra: Cassandra is a column-family, highly scalable and distributed NoSQL database. This type of database could be modeled around query patterns. Cassandra also has a sharding feature and auto-sharding support. In terms of security, Cassandra has a lot of vulnerabilities similar to MongoDB. It does not provide a mechanism to automatically encrypt the data. The mitigation is managing the encryption before parsing to the database which is the same as MongoDB's case. Cassandra Query Language, CQL, is a parsed language and it is also valnerable to injection attacks exactly like MongoDB. [13]

Comparing the performance perspective, Cassandra performs better in application with heavy data load because it can have multiple master nodes in a cluster while MongoDB can have only one master node in the cluster at any given time. It's a suitable alternative for Pantip that needs the sharding to manage the I/O of the system. Cassandras also has higher writing capabilities because of multiple master nodes which leads to higher scalability[16]. MongoDB has higher flexibility in terms of schema but the structure in the Pantip website is quite static. Therefore, implementing Cassandra is still efficient for the system.

3) Redis: Redis is a key-value, highly available and distributed NoSQL data store. Unlike other NoSQL databases, Redis supports three kinds of partitioning : "client side partitioning", "proxy assisted partitioning" and "Query routing". Redis is a single-threded database so it mostly gets more done with running on one care. It does not provide authentication by default and it does not ensure any kind of access control mechanism. It purely depends on third party SSL implementation similar to MongoDB. [20]

Comparing MongoDB with Redis in terms of performance, Redis is better when working with rapidly changing data and foreseeable database size. However, it uses a lot more RAM to store the same amount of data and very difficult to deploy in large cloud deployments. Clustering also has to be done manually. Moreover, the memory cost will surge as the performance increase. Therefore, Redis might not be a good option for Pantip which needs to do range query and scale over time.

4) MySQL: In 2013, Pantip migrated their system to MongoDB due to the limited ability to scale of MySQL. Also, the Sun Microsystems acquisition by oracle had driven MySQL to the closed-source model instead of community driven one which led to a limitation in getting supports. CTO of Pantip lost his trust in MySQL because of these issues.

Fast forward to 2020, there are several approaches to scale on MySQL. By default, it can be either Vertical or Hybrid approaches. Horizontally scaling can be done by extended technologies such as CloudSQL, ProxySQL, MySQL NDB Cluster, etc. Master-slave replication and sharding are the main features of these tools to achieve scaling.

However, scalability shouldn't be the only criterion for choosing the database. The nature of data storage and operations are also critical. If Pantip keeps using MySQL and the number of table keeps growing as the users grow, the join operation must be extensive. Also, the insertion and retrieval will be slower compared to MongoDB as the nature of MySQL tables are normalized.

VI. Conclusion

In the conclusion, Pantip migration from MySQL to MongoDB is a very interesting use case for choosing the sharding strategy when implementing MongoDB. The main cause of their problem is choosing a monotonically increasing shard key along with range-based sharding. The typical solution to this problem is changing to hash sharding. Pantip had tried this approach and encountered the pitfall which was a poor read performance and it opposes the main operation and the nature of the businesses.

Pantip ended up with using a pre-splitting chunk range. It was a good decision to deal with read performance but another strategy that Pantip should do is increasing cardinality by adding category_id.

For the alternative approach, if Pantip had migrated the system in this period, Postgres and Cassandra would have been the strongest candidates for Pantip to consider among four technologies listed.

It is crucial for Pantip to monitor the cluster, spend enough time and research so that they can foresee the plan for scaling requirements and avoid having similar issues that they encountered in the past.

References

[1] Aquaring. Pantip A Popular Internet Forum Among Thai People. https://www.aquaring.co.jp/en/gs/news/thai_pantip. 2018 (accessed September 23, 2020).

[2] Shanon Block. Why Amazon, Google, Netflix and Facebook Switched to NoSQL? https://shannonblock.org/why-amazon-google-netflix-and-facebook-switched-to-nosql/. 2018 (accessed September 23, 2020).

[3] KEVIN BURKE. Reddit's database has two tables? https://kevin.burke.dev/kevin/reddits-database-has-two-tables//. 2012 (accessed September 25, 2020).

[4] The cloud. Pantip, the legend of online community. https://readthecloud.co/pantip/. 2020 (accessed October 10, 2020).

[5] Juanroy. What is the MongoDB Pre-Splitting? http://juanroy.es/what-is-the-mongodb-pre-splitting/. 2016 (accessed September 25, 2020).

[6] Akash Kathiriya. An Overview of MongoDB and Load Balancing. https://www.mongodb.com/blog/post/on-selecting-a-shard-key-for-mongodb. 2019 (accessed October 9, 2020).

[7] W. Khan et al. "SQL Database with physical database tuning technique and NoSQL graph database comparisons". In: 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC). 2019, pp. 110–116.

[8] Ayushi Mangal. Zone Based Sharding in MongoDB. https://www.percona.com/blog/2018/06/13/zone-based-sharding-in-mongodb/. 2018 (accessed October 10, 2020).

[9] MongoDB. Create Chunks in a Sharded Cluster. https://docs.mongodb.com/manual/tutorial/create-chunks-in-sharded-cluster/. 2020 (accessed October 8, 2020).

[10] MongoDB. Shard Key - MongoDB Manual. https://docs.mongodb.com/manual/core/sharding-shard-key/. 2020 (accessed October 8, 2020).

[11] MongoDB. Sharding - MongoDB Manual. https://docs.mongodb.com/manual/sharding/. 2020 (accessed October 8, 2020).

[12] L. Okman et al. "Security Issues in NoSQL Databases". In: 2011IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications. 2011, pp. 541–547.

[13] Lior Okman et al. "Security Issues in NoSQL Databases". In: (Nov. 2011). doi: 10.1109/TrustCom.2011.70.

[14] L. Saino et al. "Load Imbalance and Caching Performance of Sharded Systems". In: IEEE/ACM Transactions on Networking 28.1 (2020), pp. 112–125.

[15] High Scalability. Reddit: Lessons Learned From Mistakes Made Scaling To 1 Billion Pageviews A Month. http://highscalability.com/blog/2013/8/26/reddit-lessons-learned-from-mistakes-made-scaling-to-1-billi.html. 2013 (accessed October 10, 2020).

[16] Sofija Simic. Cassandra Vs MongoDB - What Are The Differences? https://phoenixnap.com/kb/cassandra-vs-mongodb. 2020 (accessed October 10, 2020).

[17] [Apisil Trungkanon]. Pantip MongoDB[Why pantip is down so often? Lesson learn from MongoDB]. http://macroart.net/2013/10/mongodb-lessons-learned-on-pantip/. 2013 (accessed September 23, 2020).

[18] Wittawin.A. Pantip 101. https://www.thumbsup.in.th/pantip-101. 2019 (accessed September 24, 2020).

[19] A. Zahid, R. Masood, and M. A. Shibli. "Security of sharded NoSQL databases: A comparative analysis". In: 2014 Conference on Information Assurance and Cyber Security (CIACS). 2014, pp. 1–8.

[20] A. Zahid, R. Masood, and M. A. Shibli. "Security of sharded NoSQL databases: A comparative analysis". In: 2014 Conference on Information Assurance and Cyber Security (CIACS). 2014, pp. 1–8.

[21] William Zola. On Selecting a Shard Key for MongoDB. https://www.mongodb.com/blog/post/on-selecting-a-shard-key-for-mongodb. 2015 (accessed October 8, 2020).