World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:18, No:8, 2024

Post Pandemic Mobility Analysis through Indexing and Sharding in MongoDB: Performance Optimization and Insights

Karan Vishavjit, Aakash Lakra, Shafaq Khan

***Abstract*—**The COVID-19 pandemic has pushed healthcare professionals to use big data analytics as a vital tool for tracking and evaluating the effects of contagious viruses. To effectively analyse huge datasets, efficient NoSQL databases are needed. The analysis of post-COVID-19 health and well-being outcomes and the evaluation of the effectiveness of government efforts during the pandemic is made possible by this research's integration of several datasets, which cuts down on query processing time and creates predictive visual artifacts. We recommend applying sharding and indexing technologies to improve query effectiveness and scalability as the dataset expands. Effective data retrieval and analysis are made possible by spreading the datasets into a sharded database and doing indexing on individual shards. Analysis of connections between governmental activities, poverty levels, and post-pandemic wellbeing is the key goal. We want to evaluate the effectiveness of governmental initiatives to improve health and lower poverty levels. We will do this by utilising advanced data analysis and visualisations. The findings provide relevant data that support the advancement of UN sustainable objectives, future pandemic preparation, and evidence-based decision-making. This study shows how Big Data and NoSQL databases may be used to address problems with global health.

***Keywords*—**COVID-19, big data, data analysis, indexing, NoSQL, sharding, scalability, poverty.

I. Introduction

THE focus of this study is to accurately understand the effects of the COVID-19 pandemic, which continues to pose enormous problems globally, advanced data-driven techniques are required. In Phase 1 of our study, we tackled the challenges of COVID-19 datasets [1], [2] and looked at how government policies affected people's health, happiness, and levels of poverty. Our goal is to gather important insights about the pandemic's impact and government initiatives using NoSQL databases, sharding, indexing, and predictive modelling approaches. These insights will drive future preparedness and recovery measures to improve global health and well-being.

Our study suggests using sharding and indexing strategies on a NoSQL database, notably MongoDB, to satisfy the demand for increased query performance and scalability. We intend to overcome the limits of conventional relational databases in processing large and highly variable COVID-19 datasets by integrating Big Data Analytics with NoSQL technology, implementing sharding [3], and indexing [4].

In order to enable parallel processing and improve query speed, we suggest using sharding to divide data among many clusters. To further improve the performance of query execution, we will simultaneously apply indexing to each shard. These methods are used in our study in an effort to accelerate the query processing, increase scalability, and provide priceless visual artifacts that reveal post-pandemic health, well-being, and poverty levels. Additionally, we recognise and address the shortcomings of previous research, emphasizing the adoption of sharding and thorough analyses of various datasets to fill in the gaps.

II. Related Works

A. An Empirical Performance Comparison between MySQL and MongoDB [5]

In the COMEX Database, the effectiveness of relational (MySQL) and non-relational (MongoDB) databases was compared. In data manipulation, query processing, and aggregations, MongoDB outperformed MySQL. MongoDB exceeded MySQL in terms of performance and query processing speed. MongoDB outperforms MySQL in terms of massive dataset handling and scalability. The inclusion of sharding and indexing increased MongoDB's performance even more.

B. Assessment of SQL and NoSQL Systems to Store and Mine COVID-19 Data [6]

For storing COVID-19 data, the study compared SQL (relational) and NoSQL (MongoDB and Cassandra) databases. In terms of query retrieval time, CPU use, and size, nonrelational databases (e.g., MongoDB) beat relational databases. MongoDB performed better when dealing with big amounts of data and unstructured datasets. Relational databases (SQL) are best suited for structured data and join queries, whereas NoSQL (MongoDB) is best suited for unstructured data. Relational databases are better suited for complicated queries requiring numerous joins, whereas NoSQL databases may be better suited for basic, denormalized queries.

C. Query Optimization Using Indexing and Sharding in MongoDB [7]

MongoDB's speed was greatly enhanced by indexing and

Karan Vishavjit, Aakash Lakra, and Shafaq Khan are with Department of Computer Science, University of Windsor, Canada (e-mail: vishavj@uwindsor.ca, lakraa@uwindsor.ca, shafaq.khan@uwindsor.ca)

sharding, especially for bigger datasets. Evaluation of complex queries was lacking, which made it difficult to comprehend performance problems in the actual world. MongoDB's indexing employs data structures to speed up data retrieval, whereas MySQL uses B-trees. In contrast to MySQL, which has to modify its structure in order to accommodate changes, MongoDB's dynamic architecture makes it simple to modify data without impacting already-existing entries. MySQL needs relational tables for data organization, but MongoDB's document-based approach supports layered structures, making it suited for hierarchical data representation.

III. Methodology

A. Technology Used

1) *Database technology:* MongoDB, a NoSQL database, will continue to be used because of its effectiveness and scalability in managing huge, unstructured datasets like COVID-19 data. The implementation will involve setting up a MongoDB cluster with several shards to hold the various COVID-19 data subsets.
2) *Sharding and Indexing:* To spread data over several MongoDB instances (shards) [8] and to assure parallel processing for enhanced query performance and scalability, we will use sharding. To improve query execution and retrieval time, indexing will be applied to each individual shard.
3) *Config Server and Mongos Routers:* The MongoDB cluster will come with a Config Server, which serves as a central repository for cluster configuration, and Mongos Routers, which serve as query routers to route application requests to the proper shards.

B. Datasets Used and Data Collection Methods

1) *The COVID-19 data:* which contain details on infection rates, governmental regulations, patterns of movement, wellbeing indices, and poverty levels, will serve as the main dataset for this study. The information will come from trustworthy and reputable sources, including government databases, health organizations, and research libraries.
2) *Data Collection:* Reliable sources of data will be accessed. At several phases of the data gathering process, data validation and cleaning will be carried out to guarantee data quality and accuracy.

C. Data Analysis Methods

1) *Query Processing:* The proposed model attempts to handle complicated queries to study relationships between post-pandemic mobility, well-being, health, and poverty levels. To improve query speed, efficient query processing techniques like sharding and indexing will be used.
2) *Data Visualization:* Techniques for data visualization will be used to derive insightful conclusions and convey data well [9]. To help with the understanding and analysis of COVID-19 data, visual artifacts such as interactive dashboards, graphs, and charts will be made.
3) *Data Quality and Validation:* To guarantee accurate and trustworthy outcomes, some subsets of data will go through stringent data quality checks and validation. Outliers, discrepancies, and missing or incorrect data points will all be found and dealt with throughout this procedure.

D. Evaluation and Conclusion

The proposed model and approach will be judged on how effectively they can manage vast and complicated datasets, improve query processing, and offer insightful information on the relationships between COVID-19, well-being, health, and poverty levels. The data will be evaluated, and judgments on how well sharding, indexing, and data analysis techniques work to enhance database efficiency and scalability will be made.

By proving the effective use of sharding and indexing in MongoDB for COVID-19 data processing, we want to advance the field of advanced database topics through our study. The results will help with evidence-based decision-making and upcoming pandemic preparedness activities by shedding light on the effects of governmental actions, post-pandemic health outcomes, and poverty levels.

IV. Results

Post implementation of indexing and sharding on COVID-19 dataset in MongoDB, our research yielded significant improvements in performance and query efficiency. These techniques proved high effective in optimizing query execution.

Query Performance Optimization: Indexing [10] has greatly improved query performance by reducing time required to retrieve specific information. Figs. 4 and 5 show that there is significant difference in the time in milliseconds it took to execute same query.

Efficient Data Distribution: From Fig. 2 we can see sharding has effectively divided the data into smaller, manageable chunks across shards. Not only does this ensures even data distribution, but it also enhances data retrieval efficiency by directing queries to specific shards (refer to Fig. 8), thereby reducing the necessity to scan the entire dataset for each query.

Improved Overall Performance: By enabling sharding on database and hash-based indexing there is significant improvement in the overall performance as seen in Fig. 8 for data retrieval with large and complex datasets like COVID-19. We can clearly analyse from the results that index has examined 972 records out of 1,27,872 records and fetched 684 records by directly going into shard A in 4 m/s, compared to 262 m/s it took to retrieve same results without implementing this approach.

The retrieved data can be visualised to extract meaningful insights. In Fig. 9, we can see the same data represented as time series, allowing us to observe the percentage change across various categories. The values are dynamic and based on the query results. By passing the retrieved data directly into these graphs, we can better understand the changes.

```
Shard a at a/localhost:2005
{
  data: '11.44MiB',
  docs: 58108,
  chunks: 2,
  'estimated data per chunk': '5.72MiB',
  'estimated docs per chunk': 29054
}
---
Shard b at b/localhost:2006
{
  data: '13.85MiB',
  docs: 69764,
  chunks: 2,
  'estimated data per chunk': '6.92MiB',
  'estimated docs per chunk': 34882
}
```

Fig. 1 Individual Shard Distribution

```
Totals
{
  data: '25.29MiB',
  docs: 127872,
  chunks: 4,
  'Shard a': [
    '45.23 % data',
    '45.44 % docs in cluster',
    '206B avg obj size on shard'
  ],
  'Shard b': [
    '54.76 % data',
    '54.55 % docs in cluster',
    '208B avg obj size on shard'
  ]
}
```

Fig. 2 Collective Shard Data Distribution

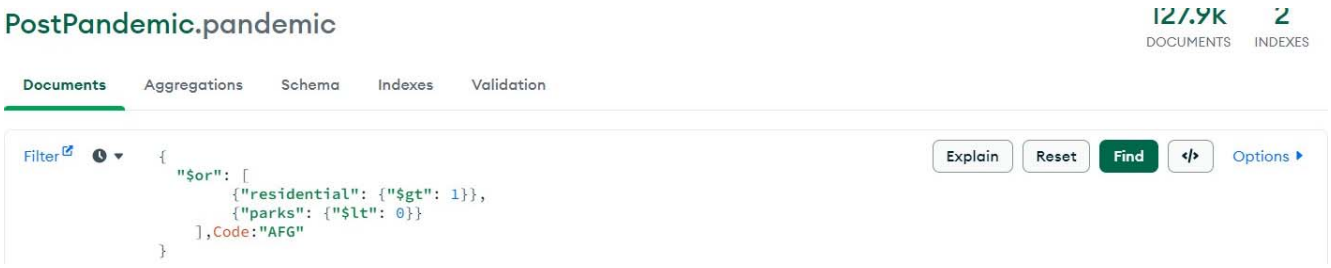

Fig. 3 Query Filter on Country code Residential and Park percentage Change without Sharding and Indexing

Fig. 4 Residential and Parks percentage change results

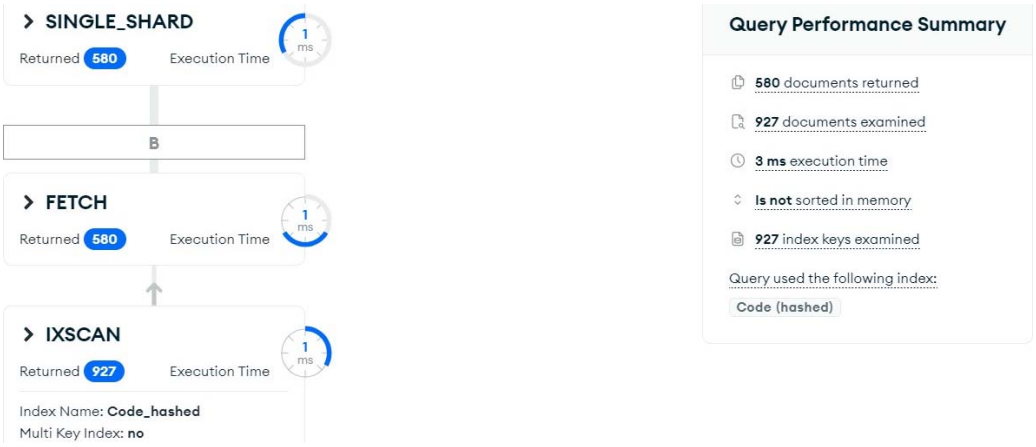

Fig. 5 Residential and Parks percentage change results with Sharding and Indexing

Fig. 6 Query Filter on Country code and Date Range

Fig. 7 Records withing Date Range without Sharding and Indexing

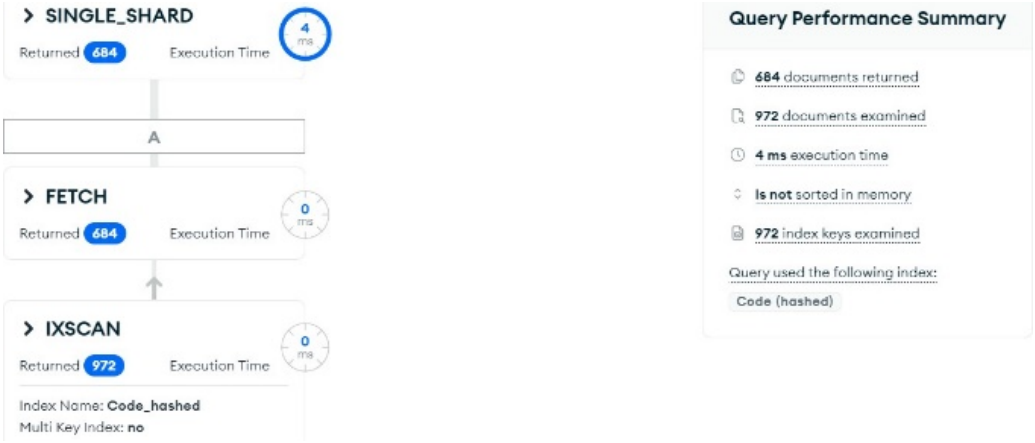

Fig. 8 Records withing Date Range with Sharding and Indexing

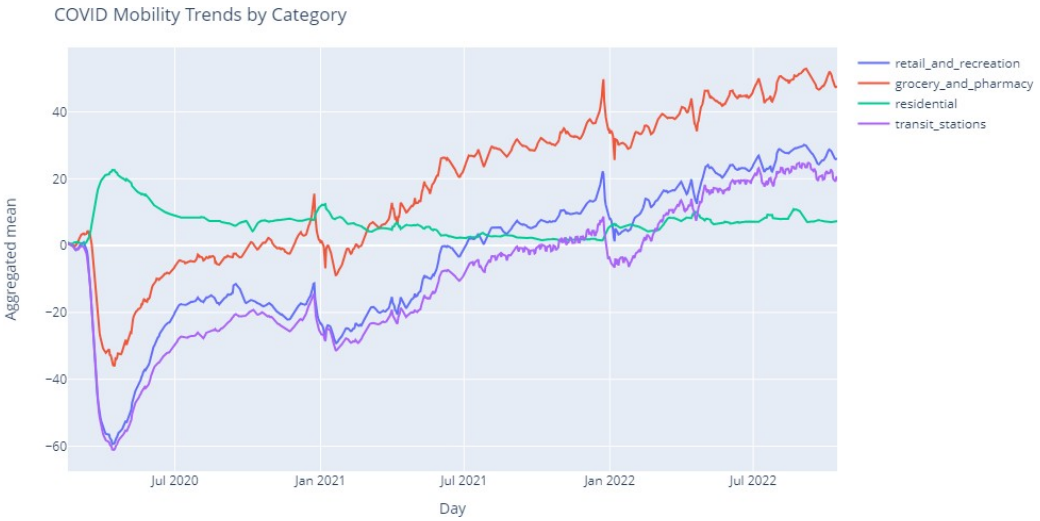

Fig. 9 Mobility Trends by Category

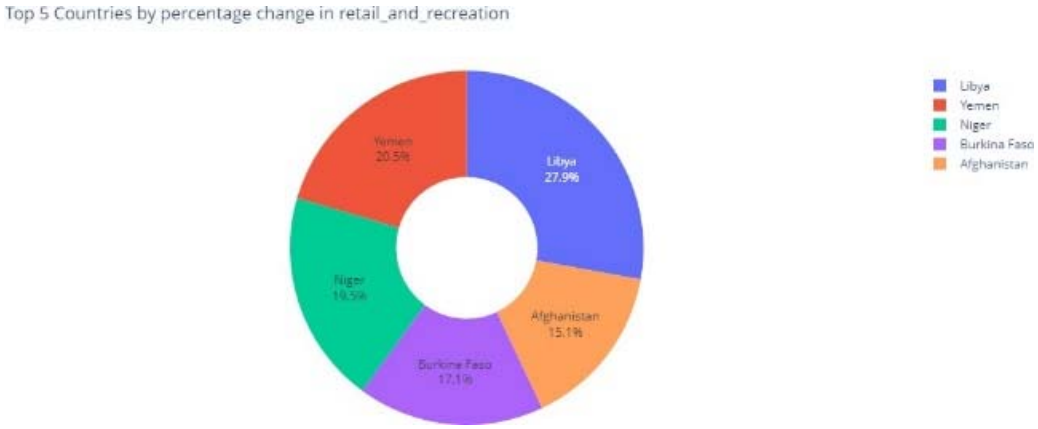

Fig. 10 Top 'n' Percentage Change in Retail and Recreation

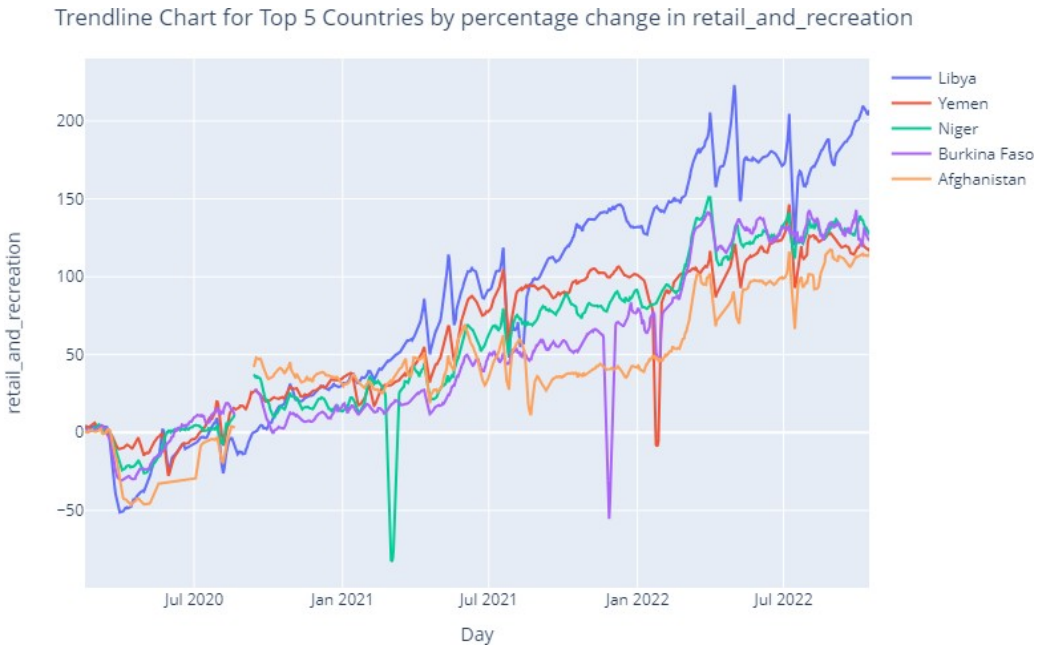

Fig. 11 Top 'n' Percentage Change in Retail and Recreation Time Series

Fig. 12 Time series Change "Canada"

Fig. 13 Time Series Change "India"

V. Limitations

Queries executed on non-indexed key values tend to result in longer durations as they necessitate full data scans, consequently consuming more memory and thereby leading to suboptimal performance. Looking ahead, as data continue to grow through insertions and updates, index management becomes increasingly complex, particularly when dealing with millions or even billions of records. Inadequate updates may give rise to latency and performance issues. Furthermore, query execution times tend to rise when data retrieval involves multiple shards, leading to cross-shard operations. Heavy write operations can introduce index contention, with multiple threads and processes vying to update the same index concurrently. Looking forward, frequent data deletions and updates may trigger index fragmentation, which can subsequently undermine system performance. Lastly, in the future, the occurrence of a shard becoming a hotspot, experiencing higher loads, while others remain underutilized, may lead to imbalanced resource utilization within the system. These considerations underscore the importance of proactive index management and efficient data handling strategies in ensuring robust MongoDB performance.

VI. Conclusion and Future Work

A. Conclusion

In our comprehensive investigation into MongoDB's performance, a NoSQL document-based datastore, we placed a strong emphasis on comparing response times. We examined the impact of indexing on query execution times by implementing a series of diverse queries. To gauge performance accurately, we employed evaluation metrics such as ExecutionSuccess, executionTime, total keysExamined, and totalDocsExamined. To facilitate our testing process, we set up a replica set and a streaming replication cluster for both the local MongoDB and PostgreSQL servers. Notably, our findings revealed that with the implementation of indexing and sharding, we observed substantial speed improvements, particularly when dealing with larger datasets. This underscores the significant role that proper database optimization techniques can play in enhancing MongoDB's overall efficiency and responsiveness.

B. Future Work

To further enhance speed and optimize data management, we consider implementing a strategy to update indexes every time new data are inserted or existing data are updated. Additionally, for improved data quality assessment and accuracy, we explore the possibility of hosting MongoDB on the cloud while integrating it with Data Quality tools like Talend and MongoDB Compass. This integration can provide valuable insights into data accuracy and completeness, ultimately leading to better output results. Furthermore, we leverage MongoDB Atlas to implement sharding for databases hosted on servers or cloud infrastructure, ensuring scalability to efficiently manage expanding data volumes. It is also advisable to explore alternative deployment methods that utilize configuration files instead of command-line sets, offering greater deployment flexibility. Finally, to enhance overall database performance and reliability, we consider automating instance launch procedures and improving replication techniques. These forward-looking objectives can pave the way for more dependable and efficient data management and processing solutions, particularly in the realm of large-scale applications.

References

[1] Edouard Mathieu, Hannah Ritchie, Lucas Rode´s-Guirao, Cameron Appel, Charlie Giattino, Joe Hasell, Bobbie Macdonald, Saloni Dattani, Diana Beltekian, Esteban Ortiz-Ospina and Max Roser (2020) "Coronavirus Pandemic (COVID-19)". Published online at OurWorldInData.org. Retrieved from: https://ourworldindata.org/coronavirus' (Online Resource).

[2] Google. "COVID-19 Community Mobility Report." COVID-19 Community Mobility Report, 2020, www.google.com/covid19/mobility/.

[3] "Database Sharding: Concepts & Examples." MongoDB, 2022, www.mongodb.com/features/database-sharding-explained.

[4] R. Chopade and V. Pachghare, "MongoDB Indexing for Performance Improvement," Advances in Intelligent Systems and Computing, pp. 529–539, 2020, doi: https://doi.org/10.1007/978-981-15-0936-0 56.

[5] A. Gomes et al., "An Empirical Performance Comparison between MySQL and MongoDB on Analytical Queries in the COMEX Database,"

2021 16th Iberian Conference on Information Systems and Technologies (CISTI), Jun. 2021, doi: https://doi.org/10.23919/cisti52073.2021.9476623.

[6] J. Antas, R. Rocha Silva, and J. Bernardino, "Assessment of SQL and NoSQL Systems to Store and Mine COVID-19 Data," Computers, vol. 11, no. 2, p. 29, Feb. 2022, doi: https://doi.org/10.3390/computers11020029.

[7] "Login - University of Windsor," brightspace.uwindsor.ca. https://brightspace.uwindsor.ca/content/enforced/139690-COMP8157-4-R-2023S/csfiles/home dir/courses/COMP81572-R-2022S/COMP8157-2-R-2022S/COMP8157-1-R-2022S/ADT Project Final Report%20(1).pdf?&d2lSessionVal=VPAtNyL0RTd65CzrniVjWrKCK&ou=139690 (accessed Jun. 25, 2023).

[8] "Deploy Sharded Cluster Using Ranged Sharding — MongoDB Manual." Https://Github.com/Mongodb/Docs/Blob/V3.2/Source/Tutorial/ Deploy-Sharded-Cluster-Ranged-Sharding.txt, www.mongodb.com/docs/v3.2/tutorial/deploy-sharded-cluster-ranged-sharding/. Accessed 31 July 2023.

[9] Fan Zuo. Jingxing Wang, Jingqin Gao, Kaan Ozbay, Xuegang Jeff Ban, Yubin Shen, Hong Yang, Shri Iyer, "An Interactive Data Visualization and Analytics Tool to Evaluate Mobility and Sociability Trends During COVID-19." https://arxiv.org/pdf/2006.14882.pdf

[10] Indexes — MongoDB Manual," www.mongodb.com. https://www.mongodb.com/docs/manual/indexes/