Enhancing MongoDB query performance through index optimization

Marat Nuriev[1*], *Rimma* Zaripova[2], *Olga* Yanova[2], *Irina* Koshkina[3], and *Andrey* Chupaev[4]

[1]Kazan National Research Technical University named after A. N. Tupolev – KAI, Kazan, Russia
[2]Kazan State Power Engineering University, Kazan, Russia
[3]Kazan Federal University – Naberezhnye Chelny Institute, Naberezhnye Chelny, Russia
[4]Kazan National Research Technological University, Kazan, Russia

Abstract. This article delves into the critical aspect of enhancing query performance in MongoDB through meticulous index optimization. It begins with an introduction to MongoDB's unique document-oriented data storage approach and its inherent scalability, which sets the stage for understanding the importance of efficient query processing. The discussion progresses to highlight the pivotal role of indexes in MongoDB, emphasizing their function in expediting data retrieval and the necessity for their optimization to ensure peak database performance. A detailed exploration is provided on the methodologies for identifying fields suitable for indexing, considering factors such as query frequency and the specific use of fields in query operations. The article further elaborates on the selection of optimal index types, tailored to the diverse needs of varying data and query scenarios, thereby underscoring the versatility of MongoDB's indexing capabilities. Management of index size is discussed as a critical component of optimization, addressing the balance between index efficiency and resource consumption. The utilization of MongoDB's query planner is showcased as a powerful tool for achieving an in-depth understanding of query execution and identifying potential optimizations. In conclusion, the article encapsulates the essence of continuous index management and the strategic use of MongoDB's analytical tools to maintain and enhance database performance. It underscores the ongoing nature of optimization efforts required to keep pace with evolving data patterns and application demands, ultimately ensuring a responsive, efficient, and scalable database environment.

1 Introduction

MongoDB, a document-oriented database management system, introduces a distinct approach to data storage. Unlike traditional relational databases that organize data into tables, MongoDB stores information in document format, making it particularly suitable for complex or non-standard data schemas. These documents are represented in BSON (Binary JSON) format, accommodating a wide range of data types including strings, numbers, arrays,

---

[*] Corresponding author: marat_nu1@mail.ru

nested documents, and even binary data. This flexibility allows MongoDB to handle diverse and complex data structures efficiently [1,2].

The UML sequence diagram illustrates the analytical query processing workflow in a system utilizing MongoDB (see Fig.1). The diagram illustrates the following steps:

1) Client Application sends an analytic query to the Application Server.
2) Application Server processes the query and formulates a corresponding database query using the MongoDB Driver [3].
3) MongoDB Driver forwards the query to the MongoDB Server.
4) MongoDB Server accesses the MongoDB Database to fetch or manipulate data [4].
5) Data is returned from the database to the server, then to the driver, and finally, the application server sends the analytic query results back to the client application.

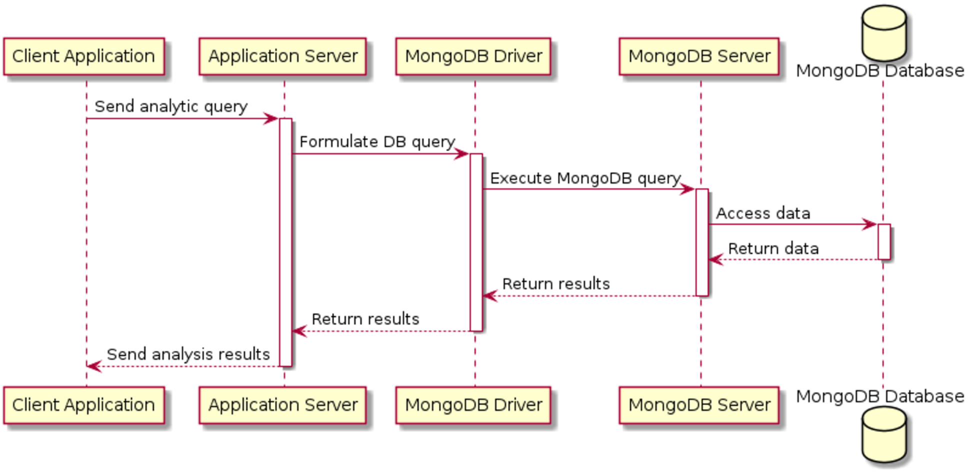

Fig. 1. Analytic query processing in a MongoDB system.

One of MongoDB's key advantages is its scalability. Designed to handle vast volumes of queries and data, it is an ideal choice for large-scale web applications and services that demand high performance and flexibility. MongoDB supports horizontal scaling through sharding, distributing data across multiple servers, thus facilitating expansion without significant alterations to the system's architecture [5,6].

The efficiency of query processing in MongoDB is closely linked to the effective use of indexes. MongoDB allows for the customization of indexes for different data types, significantly accelerating data retrieval and access [7]. Properly optimized indexes enhance performance by reducing the need for full document scans and cutting down response times.

As an open-source product, MongoDB offers developers the liberty to create specialized solutions and integrate with various systems. Thanks to its active community and an extensive range of available plugins and extensions, MongoDB provides flexible options for database functionality expansion [8,9].

However, MongoDB is not a one-size-fits-all solution for every data type. It excels with data that is complex in structure and requires schema flexibility. For projects with simple data structures and rigid schemas, traditional relational databases might be a better fit.

This article focuses on index optimization as a key method to improve query performance in MongoDB. Index optimization involves selecting the appropriate index types for specific queries, managing the order of fields in indexes [10], and balancing between data write speed and read speed. We will explore best practices, techniques, and strategies that can help you fully leverage the potential of indexing in MongoDB to enhance data handling efficiency.

2 The importance of index optimization in MongoDB

In the realm of MongoDB, indexes serve a pivotal role in ensuring query execution efficiency. Fundamentally, MongoDB indexes function akin to pointers in a book, facilitating rapid data search processes. Absent these indexes, MongoDB would be compelled to undertake a complete scan of all documents within a collection for every query – a procedure that becomes exceedingly resource-intensive and sluggish with large data volumes [11,12].

Optimizing indexes within MongoDB can lead to a substantial acceleration in query execution. Well-planned and configured indexes guarantee swift access to the requisite data, minimizing the time needed for data retrieval. This means that instead of perusing every document in a collection, a query can swiftly locate the pertinent data via an index.

Effective index optimization results in a significant reduction in response times for queries, which is crucial in applications where response speed is critical. This is particularly relevant in real-time interactive systems, such as online gaming [13], high-traffic websites, and data-intensive applications. Optimized indexes contribute to enhanced performance, improve user experience, and help avert delays and performance issues as the system scales.

Beyond query acceleration, index optimization also facilitates more efficient utilization of system resources, such as memory and CPU power. Poorly planned or excessive indexes can lead to unnecessary resource consumption and increased query processing times. Consequently, index optimization encompasses not only the selection of appropriate fields for indexing but also the regular review and updating of indexes to align with evolving data and query requirements. This strategic approach ensures that indexes remain effective and resource-efficient, supporting the overarching goal of maintaining high-performance, scalable database systems [14,15].

3 Identifying fields for indexing in MongoDB

The initial step in index optimization within MongoDB is a meticulous analysis of the queries frequently executed in your database. Understanding which fields are utilized in "where" clauses, which are often involved in sorting operations ("sort"), and which participate in query constraints ("limit") is crucial [16]. This analysis helps in identifying the key fields that are candidates for indexing.

Assess the frequency and contexts in which different fields are used in your queries. For instance, a field frequently used in "where" conditions is a prime candidate for indexing. Also, pay attention to fields commonly used for sorting data, as indexing these can significantly expedite response times [17,18].

In instances where queries are complex and encompass numerous fields, index optimization becomes a more intricate task. It's essential to thoroughly examine these queries to determine which combinations of fields are most frequently used together. Composite indexes, which incorporate multiple fields, can be particularly beneficial in such scenarios.

When selecting fields for indexing, it's important to consider the balance between enhancing query performance [19] and the costs associated with storing and maintaining these indexes. Indexes consume additional memory and disk space and require time to update when data is added, removed, or modified. Therefore, indiscriminate indexing of every field may be impractical.

To facilitate the process of determining the most crucial fields for indexing, specialized analytical tools and MongoDB functionalities, such as the `explain()` function, can be utilized. These tools help understand how queries are executed and how indexes can improve their performance. By leveraging these insights, developers can make informed decisions about which fields to index, ultimately leading to optimized query performance while managing the overhead associated with index maintenance [20,21]. This strategic approach ensures that the benefits of indexing are maximized without unnecessarily burdening the system resources.

4 Selecting the appropriate index type in MongoDB

MongoDB offers a variety of index types, each optimized for specific kinds of queries and data. Understanding the distinctions among these index types and selecting the most suitable one for particular use cases can significantly enhance database performance [22].

Standard indexes, often referred to as B-tree indexes, are the most common and versatile, suitable for a broad range of query operations. They are efficient for search, sort, and aggregation operations [23,24]. These indexes are particularly valuable when operations need to be performed on multiple fields, offering a balance between speed and versatility.

Unique indexes ensure that each value in the indexed field is distinct. They are ideal for fields containing identifiers, such as user logins or registration numbers. Unique indexes not only expedite queries but also maintain data integrity by preventing duplicate values.

Geospatial indexes are designed for handling geospatial data and are used for location-based queries, such as finding objects within a specific radius. These indexes are crucial for applications involving mapping [25], Geographic Information Systems, and mobile device location services, providing efficient querying of spatial data.

Text indexes in MongoDB are optimized for searching textual content within documents. They enable complex text search queries, including keyword and phrase searches. Text indexes are perfect for implementing search functionalities in blogs, news websites, and other applications where efficient text searching is required [26,27].

Composite indexes combine multiple fields and can be tailored to optimize specific queries that involve multiple conditions. These indexes are particularly beneficial for enhancing the performance of complex queries that include multiple criteria, allowing for more efficient data retrieval across several dimensions.

By carefully selecting the appropriate index type based on the nature of the queries and the data involved, developers can ensure that MongoDB operates with optimal efficiency [28]. This strategic approach to index selection helps in achieving faster query response times, reduced load on system resources, and an overall improvement in application performance.

5 Managing index size for optimal performance in MongoDB

Indexes, while crucial for accelerating queries in MongoDB, also contribute to the overall size of stored data. This becomes particularly noticeable with large data volumes, where indexes can occupy substantial disk space. Large indexes not only impact storage performance but also the speed of write operations, as data updates necessitate corresponding modifications in indexes [29,30].

Regular monitoring of index sizes is a key aspect of MongoDB performance optimization. By utilizing the `db.collection.stats()` command, developers and database administrators can access information about index sizes and assess their impact on overall system performance. This enables informed decisions regarding the deletion or rebuilding of indexes to be made [31].

As data characteristics and query patterns evolve over time, there arises a need for updating and rebuilding indexes. Changes in data schema, the introduction of new features, or shifts in data access patterns can trigger this necessity. Employing commands such as `db.collection.reIndex()` and `db.dropIndex()` allows for the adaptation of indexes to current needs, thereby enhancing data handling performance and efficiency.

The key to successful index management lies in finding the optimal balance between index size and query performance [32,33]. It is essential to evaluate to what extent reducing the size of an index may impact query execution times and, conversely, how increasing index size can improve performance but at the cost of additional resources.

Regular analysis of index usage is recommended to identify the most valuable indexes and those that could be removed or rebuilt to enhance overall system efficiency. This involves analyzing the frequency of index usage, their impact on query execution times, and the space they occupy. By continually reassessing and adjusting indexes, organizations can ensure that their MongoDB databases remain efficient [34], responsive, and capable of handling the demands of modern data-intensive applications. This strategic approach to index management not only optimizes resource utilization but also maintains high query performance, contributing to the overall effectiveness and scalability of the database system.

6 Leveraging the query planner in MongoDB for enhanced performance

The query planner in MongoDB is a robust tool designed to analyze and optimize the way queries are executed. A properly configured query planner can significantly enhance system performance by identifying the most efficient execution paths for queries, especially when dealing with large data volumes and complex indexes [35,36].

To harness the capabilities of the query planner, the first step involves enabling query profiling, which can be accomplished using the `setProfilingLevel(2)` command. This profiling level logs all query operations [37], allowing administrators and developers to conduct a thorough analysis of query performance.

Once profiling is activated, the `db.system.profile.find()` command can be employed to review the collected data. This offers the opportunity to examine which queries are running slowly, which indexes are being utilized (or not), and to identify potential performance bottlenecks. Understanding how queries interact with indexes is crucial for pinpointing areas for further optimization [38,39].

Based on the insights gathered, query execution can be optimized through various means such as rebuilding or adding indexes, modifying the structure of queries, or even reorganizing data. Optimization may also involve scheduling database maintenance tasks [40], like regular index rebuilding or purging outdated data, to maintain optimal performance.

For more advanced analysis, MongoDB provides additional tools and functions, such as the `explain()` command for queries [41,42]. This can offer deeper insights into the query execution plan, revealing how the database decides which index to use and how data retrieval operations are organized.

By strategically utilizing the query planner and associated analytical tools, organizations can fine-tune their MongoDB databases to achieve superior performance. This involves not only the technical adjustments to indexes and queries but also adopting a proactive approach to database maintenance and optimization [43,44]. Through continuous monitoring and adjustment, it's possible to ensure that the database remains responsive, efficient, and capable of handling the evolving demands of modern applications. This holistic approach to database management underscores the importance of the query planner as a critical component in the MongoDB optimization toolkit.

7 Conclusion

Optimizing query performance in MongoDB through index optimization is a multifaceted process that involves a deep understanding of the database's structure, the nature of the data stored, and the types of queries most frequently executed [45]. The journey begins with a thorough analysis of query patterns to identify key fields for indexing, followed by selecting the appropriate index type tailored to specific data and query requirements. Managing index size is crucial to maintaining a balance between performance and resource utilization, ensuring that indexes do not become a burden on the system [46].

The MongoDB query planner plays an instrumental role in this optimization process, offering valuable insights into query execution and identifying opportunities for further refinement. By leveraging tools such as query profiling and the `explain()` command, developers and administrators can gain a deeper understanding of the database's operational dynamics [47], enabling them to make informed decisions about index management and query optimization.

Ultimately, the goal of index optimization in MongoDB is to enhance the efficiency and responsiveness of the database, ensuring that it can handle the demands of complex queries and large data volumes with ease. This requires a continuous effort to monitor, analyze, and adjust indexes and queries in response to evolving data patterns and application requirements. Through diligent management and strategic optimization of indexes, MongoDB can provide a robust, scalable, and high-performance database environment that supports the diverse needs of modern applications [48].

In conclusion, index optimization in MongoDB is not just a one-time task but an ongoing process of improvement and adjustment. By embracing a proactive approach to index management and leveraging the powerful tools and features MongoDB offers, organizations can ensure their databases remain agile, efficient, and capable of delivering the high performance required in today's data-driven world.

References

1. J. Kumar, V. Garg, 2017 International Conference on Computing and Communication Technologies for Smart Nation (IC3TSN), Gurgaon, India, 300-305 (2017)
2. H. Abbes, S. Boukettaya, F. Gargouri, 2015 IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA), Marrakech, Morocco, 1-7 (2015)
3. Z. Huang, L. Qian, D. Cai, 2022 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), Dalian, China, 680-684 (2022)
4. A. P. Mudrov, F. F. Khabibullin, G. V. Pikmullin, Z. D. Gurgenidze, BIO Web of Conferences **52**, 00046 (2023)
5. M. G. Yarullin, F. F. Khabibullin, Lecture Notes in Mechanical Engineering, 145-153 (2017)
6. Y. Smirnov, A. Kalyashina, R. Zaripova, International Russian Automation Conference (RusAutoCon), Sochi, Russian Federation, 913-917 (2022)
7. Z. Gizatullin, R. Gizatullin, 2023 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM), Sochi, Russian Federation, 261-265 (2023)
8. Z. M. Gizatullin, M. P. Shleimovich, Russ. Aeronaut **66**, 154-161 (2023)
9. S. Lyasheva, R. Safina, M. Shleymovich, 2023 International Conference on Industrial Engineering, Applications and Manufacturing, 797-802 (2023)
10. M. Shleymovich, R. Safina, 2022 International Russian Automation Conference, 289-293 (2022)
11. R. M. Shakirzyanov, A. A. Shakirzyanova, 2021 International Russian Automation Conference (RusAutoCon), 714-718 (2021)

12. Y. I. Soluyanov, A. I. Fedotov, D. Y. Soluyanov, A. R. Akhmetshin, IOP Conference Series: Materials Science and Engineering **860(1)**, 012026 (2020)
13. A. V. Chupaev, R. S. Zaripova, R. R. Galyamov, A. Y. Sharifullina, E3S Web of Conferences **124**, 03013 (2019)
14. L. V. Plotnikova, R. R. Giniyatov, S. Y. Sitnikov, M. A. Fedorov, R. S. Zaripova, IOP Conference Series: Earth and Environmental Science **288**, 012069 (2019)
15. M. Tyurina, A. Porunov, A. Nikitin, R. Zaripova, G. Khamatgaleeva, Lecture Notes in Mechanical Engineering, 391-402 (2022)
16. E. I. Gracheva, O. V. Naumov, Journal of Pharmacy and Technology **8**, 4, 26763-26770 (2016)
17. D. D. Micu, I. V. Ivshin, E. I. Gracheva, O. V. Naumov, A. N. Gorlov, E3S Web of Conferences **124**, 02013 (2019)
18. O. Soloveva, S. Solovev, R. Zaripova, F. Khamidullina, M. Tyurina, E3S Web of Conferences **258**, 11010 (2021)
19. R. F. Gibadullin, N. S. Marushkai, 2021 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM), 404-409 (2021)
20. V. O. Kozelkova, G. A. Ovseenko, V. I. Karachin, T. Van Tung, N. C. Kien, R. S. Kashaev, 4th International Youth Conference on Radio Electronics, Electrical and Power Engineering (REEPE), 1-4 (2022)
21. R. F. Gibadullin, I. S. Vershinin, R. Sh. Minyazev, 2017 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM), 1-6 (2017)
22. M. Galimov, R. Burnashev, A. Gatiatullin, 8th International Conference on Computer Science and Engineering (UBMK), 382-386 (2023)
23. V. O. Kozelkova, G. A. Ovseenko, V. I. Karachin, N. C. Kien, T. Van Tung, O. V. Kozelkov, 4th International Youth Conference on Radio Electronics, Electrical and Power Engineering (REEPE), 1-5 (2022)
24. R. F. Gibadullin, G. A. Baimukhametova, M. Yu. Perukhin, 2019 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM), 1-7 (2019)
25. G. R. Rakhmatullina, E. A. Pankova, O. V. Fukina, M. Khayytov, L. V. Chapaeva, Journal of Physics: Conference Series **2270(1)**, 012056 (2022)
26. V. A. Gerasimov, M. G. Nuriev, D. A. Gashigullin, 2022 International Russian Automation Conference (RusAutoCon), 75-79 (2022)
27. S. R. Khasanov, E. I. Gracheva, M. I. Toshkhodzhaeva, S. T. Dadabaev, D. S. Mirkhalikova, E3S Web of Conferences **178**, 01051 (2020)
28. Z. M. Gizatullin, R. M. Gizatullin, M. G. Nuriev, 2020 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), 120-123 (2020)
29. S. Lyasheva, M. Shleymovich, R. Shakirzyanov, 2019 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon), 1-6 (2019)
30. E. Gracheva, M. Toshkhodzhaeva, O. Rahimov, S. Dadabaev, D. Mirkhalikova, S. Ilyashenko, V. Frolov, International Journal of Technology **11**, 8 (2020)
31. M. Shakirzyanov, R. Gibadullin, M. Nuriyev, E3S Web of Conferences **419**, 02029 (2023)
32. K. Kulagin, M. Salikhov, R. Burnashev, 2023 International Russian Smart Industry Conference (SmartIndustryCon), 690-694 (2023)

33. J. Yoqubjonov, R. Gibadullin, M. Nuriev, E3S Web of Conferences **431**, 07011 (2023)
34. I. Viktorov, R. Gibadullin, E3S Web of Conferences **431**, 05012 (2023)
35. R. F. Gibadullin, I. S. Vershinin, M. M. Volkova, 2020 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon), 1-7 (2020)
36. R. F. Gibadullin, M. Yu. Perukhin, B. I. Mullayanov, 2020 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon), 1-6 (2020)
37. S. N. Cherny, R. F. Gibadullin, 2022 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM), 965-970 (2022)
38. V. A. Raikhlin, I. S. Vershinin, R. F. Gibadullin, Journal of Physics: Conference Series **2096**, 012160 (2021)
39. R. F. Gibadullin, I. S. Vershinin, R. Sh. Minyazev, 2018 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM), 1-6 (2018)
40. V. A. Raikhlin, R. F. Gibadullin, I. S. Vershinin, Lobachevskii Journal of Mathematics **43**, 2, 455-462 (2022)
41. G. A. Ovseenko, R. S. Kashaev, O. V. Kozelkov, T. K. Filimonova, T. S. Evdokimova, A. M. Mardanova, 5th International Youth Conference on Radio Electronics, Electrical and Power Engineering (REEPE) **5**, 1-5 (2023)
42. R. Zaripova, A. Nikitin, Y. Hadiullina, E. Pokaninova, M. Kuznetsov, E3S Web of Conferences **288**, 01072 (2021)
43. I. N. Madyshev, V. V. Kharkov, N. Z. Dubkova, M. G. Kuznetsov, AIP Conference Proceedings **2647**, 1 (2022)
44. Z. M. Gizatullin, M. S. Shkinderov, R. R. Mubarakov, Proceedings of the 2022 Conference of Russian Young Researchers in Electrical and Electronic Engineering, 1350-1353 (2022)
45. Z. Gizatullin, M. Shkinderov, 2019 International Russian Automation Conference, 8867761 (2022)
46. A. G. Ilyin, A. S. Mahdi Khafaga, V. Yunusova, 2021 Systems of Signals Generating and Processing in the Field of on Board Communications, 1-4 (2021)
47. I. Barkov, C. S. Gabdrakhmanova, G. I. Gaptullazyanova, A. V. Kholkin, In Computer Applications for Management and Sustainable Development of Production and Industry (CMSD2021) **12251**, 26-35 (2021)
48. E. Kozlov, R. Gibadullin, E3S Web of Conferences **474**, 02031 (2024)