©2025 Published in 11th International Symposium on Intelligent Technologies in Engineering and Science 16-18 May 2025 (ISITES2025 Diyarbakır - Turkey)

Comparison of Column-Based Database Performance for Varying Data Sizes

[1]Aykut GUVEN and [2]Muhammet Erol Yigin
[1]Beykent University, Faculty of Engineering, Ayazağa İstanbul, Türkiye
*[2]Beykent University, Faculty of Engineering, Ayazağa İstanbul, Türkiye

Abstract

Column-based databases generally stand out with their high-performance rates in big data environments where analytical queries are intensive. ClickHouse, as a column-based database system with open-source code and high-speed data query capacity, is among the solutions divided to meet many big data processing needs of the unit. Developments in open-source technologies in Turkey, ClickHouse's usage rate in corporate structures is increasing. It is claimed that the ClickHouse database can provide much faster solutions compared to relational databases in large change data sets.
In the study, an investigation was conducted to evaluate the performance of the ClickHouse database against varying data sizes. To obtain unbiased and repeatable performance metrics, datasets consisting of 1 million, 10 million, 50 million and 100 million records were prepared. Specific analytical queries and calculations were performed for each dataset, and the effect of the increase in data size on query times, resource usage and system efficiency were analyzed.
Automation scripts were used to ensure consistent execution of queries, elimination of manual errors and systematic measurements. Data was visualized with the help of tables and graphs. Trends in performance and factors affecting system efficiency were revealed. As a result of the study, ClickHouse's strengths and weaknesses under different workloads were determined. Findings were presented that will help researchers and system administrators choose the database system suitable for their usage scenarios.

Key words: Column-Based Databases, ClickHouse, Big Data, Data Analytics, Database Performance, Open Source Code, High Speed Queries

1. Introduction

In today's world where the need for big data processing and real-time analytics is rapidly increasing, databases with column-based and distributed architecture are becoming increasingly important in addition to relational databases. ClickHouse, one of the remarkable open-source technologies in this field, stands out with its high-performance, scalable architecture and data analytics feature. Unlike traditional row-based databases, ClickHouse, which stores data on a column basis, has begun to attract great attention in corporate and academic research environments, especially with the fast query response times it provides on big data. In Turkey, the public and private sectors are also turning to open source and high-performance solutions in line with the goal of reducing foreign technology dependency. For this reason, the need for technologies such as ClickHouse is increasing every day.

ClickHouse is used in various sectors with its highly scalable and distributed architecture, column-based compression methods, and query execution engine suitable for real-time analytical

*Corresponding author: Address: Faculty of Technology, Department of Civil Engineering Sakarya University of Applied Sciences, 54187, Sakarya TURKEY. E-mail address: mehmets@subu.edu.tr, Phone: +0264 616 0000-01-02

operations [1]. This technology, which allows efficient management of storage and query processes in large data sets, is especially preferred in areas such as finance, e-commerce, IOT and telecommunications [2].

In this study, an analysis was performed to understand the performance level of Clickhouse and obtain objective measurements. To create a scenario, different data sets consisting of 1 million, 10 million, 50 million and 100 million records were prepared, and standard queries were run on each data set and query times were systematically collected. The effects of changing data sizes on query types and functions were examined, and it was aimed to obtain performance indicators that could provide feedback for different sectoral applications [3].

Within the scope of the study, automatic scripts and test setups were used to ensure that the measurement results were consistent and repeatable, thus minimizing errors that could arise from manual intervention. The collected data was analyzed, and findings were obtained on ClickHouse's scalability capacity, bottlenecks, memory management strategies and query optimization opportunities.

The results obtained provide a better understanding of the high performance and real-time data analytics capabilities offered by ClickHouse in large-scale data management, as well as revealing the points to be considered in projects to be carried out at the institutional and research level. The results of this study are expected to guide institutions and the academic community to consider evaluating ClickHouse in the big data ecosystem.

2. Materials and Method

In this study, the performance of the ClickHouse database on large-scale data sets and various query types is examined in detail. The advantages offered by modern column-based databases become especially evident in real-time analytical workloads. However, the scalability of these systems and their behavior on distributed architectures may vary in different scenarios [4], [5]. Therefore, the following methodological steps were followed in order to reveal the extent to which ClickHouse is successful under which conditions.

2.1. Data Sets and Dimensions

2.1.1. 1 Million Records: Used to measure the basic performance response on small database queries. All queries were validated at this stage [6].

2.1.2. 10 Million Records: Performance response times of medium-sized dataset scenarios were measured. All queries were run on this dataset.

2.1.3. 50 Million Records: The performance response times of large-scale dataset scenarios were measured. All queries were run on this dataset [7].

2.1.4. 100 Million Records: Performance response times of very large-scale data set scenarios were measured.

2.2. Query Types

Queries used in performance measurements are diversified to suit real-world scenarios.

2.2.1. Simple SELECT / Filtering Queries: Specific columns are selected. These queries provide a direct demonstration of the performance advantage provided by column-based storage when reading data.

2.2.2. Statistical Queries: Queries that provide statistical calculations were used. These are the most frequently used functions in real-time analytics processes, and the response time of ClickHouse to different data volumes was observed [8].

2.2.3. Grouping Queries: Queries that group the data were used. The effect of sorting operations on memory consumption and query time is different in column-based databases, especially in such aggregate operations [9].

Each query type is adapted to all dataset sizes. For example, a query for 1 million rows is run using the same logic for 100 million, but with a different number of rows.

2.3. Automation and Performance Monitoring

To ensure repeatability and consistency, automated Python scripts were used. These scripts reported the average of 3 replications of predefined queries on each dataset. Each query was run 3 times on each database. These processes were performed automatically.

2.4. Test Environment and Hardware Conditions

By using the same software and hardware conditions throughout the study, the measurement results were aimed to reflect only the performance differences of ClickHouse [10]. The test environment is defined as follows.

- **Processor (vCPU):** 8vCPU (Intel Xeon Platinum 8000 series)
- **Memory (RAM):** 32 GB
- **Storage:** 300 GB SSD
- **Operating System:** Linux Distribution

Additionally, no non-test applications were installed on the system and services that could create “noise” in resource usage during the tests were disabled [6]. The test environment was set up as an EC2 machine over AWS. The t3.2xlarge type of the EC2 machine was selected [11].

2.5. Data Analysis and Visualization

In the data analysis step, the execution times obtained for each query type were examined statistically.

2.5.1. Preprocessing

Incorrect measurements or incomplete queries were detected and removed from the data set. Queries with high variance were re-run to obtain data that would represent the average.

2.5.2. Statistical Comparison

The consistency of the system was observed by calculating the mean, median and standard deviation values for each query type and data size. An attempt was made to determine at which point ClickHouse became a bottleneck in complex operations.

2.5.3. Graphs and Tables

As the data size increases, the increase rates in query times are monitored. It is observed how CPU/Memory/Disk usage changes in different query types.

These representations allow decision makers to quickly identify which query types consume the most resources in big data scenarios.

2.6. Interpretation of Results

The performance efficiency and scalability of ClickHouse were evaluated on the obtained results. Evaluation of the results on all queries and all data sets shows the speed of ClickHouse. The evaluation was evaluated under the following headings.

2.6.1. Performance Curve of Large Scales

Simple SELECT and statistical queries generally show linearly increasing query times even for large data sizes, with the advantage of column-based architecture. With growing data size, the memory consumption of queries increases rapidly, and query times increase more steeply.

2.6.2. Resource Usage

CPU utilization can become a hotspot for high volume datasets and complex queries. Disk IO has become a limited bottleneck due to fewer unnecessary reads from column-based storage.

2.6.3. Analytical Potential

Even in large sizes such as 50 million and 100 million rows, it provided speed advantages in aggregation and grouped queries. This is an important indicator for sectors where instant data processing is critical, such as IOT and e-commerce. When complex queries increase, the memory load rate of the system increases. Even when sufficient memory allocation is made, larger increases in query times were observed compared to other queries.

These findings objectively demonstrate the advantages and possible limitations of ClickHouse in

big data workloads. They guide organizations in their database preference planning processes.

3. Results

In the test results analysis carried out within the scope of the study, ClickHouse database performance was measured and created in the form of tables. The measurements were included in the analysis created in the form of a table.

Table 1: Query performance (ms) of small, medium, large and very large datasets Part 1

TABLE\Query	ALL	COUNT	AVG	SUM	MAX	MIN	ROUND
1M	116	53	53	54	55	54	54
10M	1864	802	823	842	829	823	833
50M	10082	5861	5822	5562	5579	5540	5707
100M	28251	10956	12215	12010	12361	12341	12485

Table 1 shows the average execution time in milliseconds of various SQL queries (ALL, COUNT, AVG, SUM, MAX, MIN, ROUND) executed on different dataset sizes (1M, 10M, 50M, 100M). The data reveals a significant increase in query execution times as the dataset size increases.

Table 2: Query performance (ms) of small, medium, large and very large datasets Part 2

TABLE\Query	GROUP BY	DISTINCT	HAVING AVG	TOMONTH	CASE	BETWEEN	UNIQEXACT	IN
1M	102	54	59	58	58	53	55	61
10M	1579	822	992	890	871	798	938	613
50M	9464	7351	7667	6477	6551	5024	7648	4270
100M	14143	11955	14516	11322	13521	11690	13380	9074

Table 2 shows the average execution times in milliseconds of various SQL queries (GROUP BY, DISTINCT, HAVING AVG, TOMONTH, CASE, BETWEEN, UNIQEXACT, IN) performed on different dataset sizes (1M, 10M, 50M, 100M).

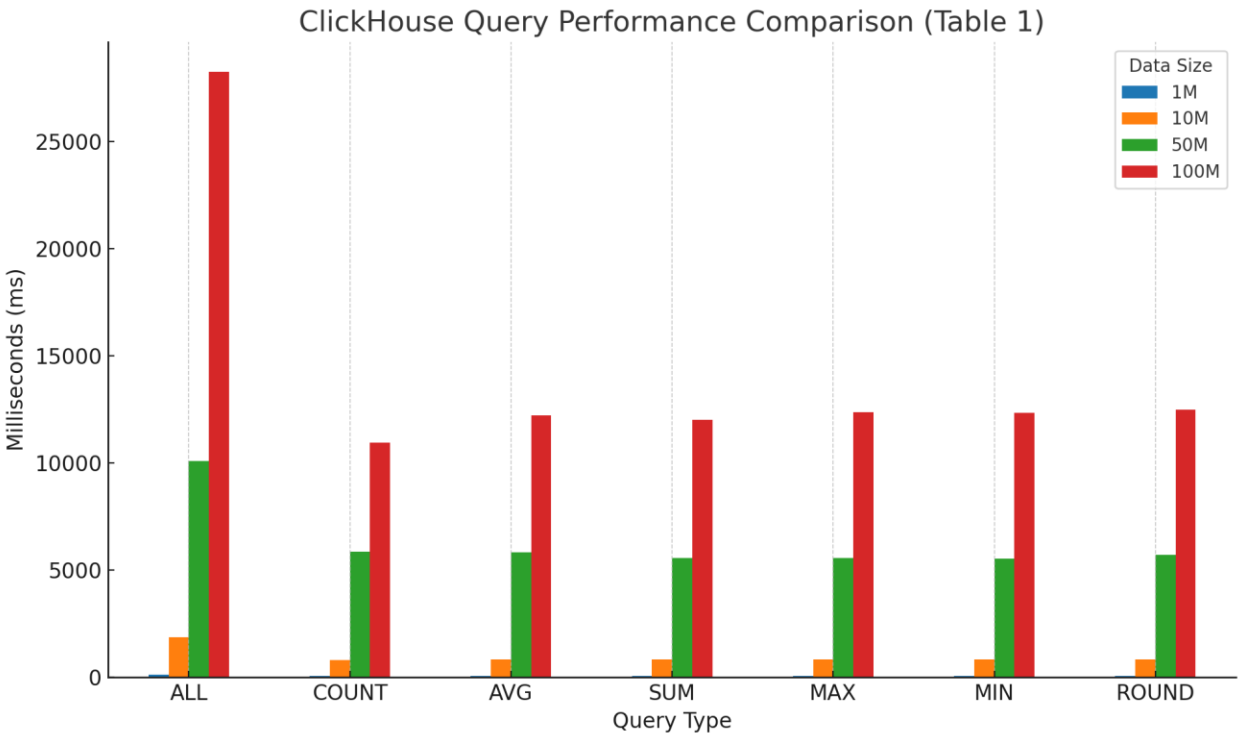

Figure 1: Column chart of queries in milliseconds Part 1

Figure 1 illustrates a detailed column chart that presents the execution times, measured in milliseconds, for a variety of SQL queries specifically “ALL, COUNT, AVG, SUM, MAX, MIN, and ROUND” as they are performed on datasets of varying sizes, including 1 million, 10 million, 50 million, and 100 million records. This visual comparison aims to highlight how query performance scales with increasing data volume, providing valuable insight into the efficiency and responsiveness of each query type under different load conditions.

In addition to showcasing the raw performance data, Figure 1 also enables the identification of trends and potential bottlenecks associated with specific query types. For example, aggregate functions such as AVG and SUM may demonstrate more consistent performance across varying data sizes, whereas queries like COUNT or ALL might exhibit significant increases in execution time as the dataset grows. This kind of analysis is crucial for database optimization, helping developers and database administrators make informed decisions when designing query strategies for large-scale data environments [15].

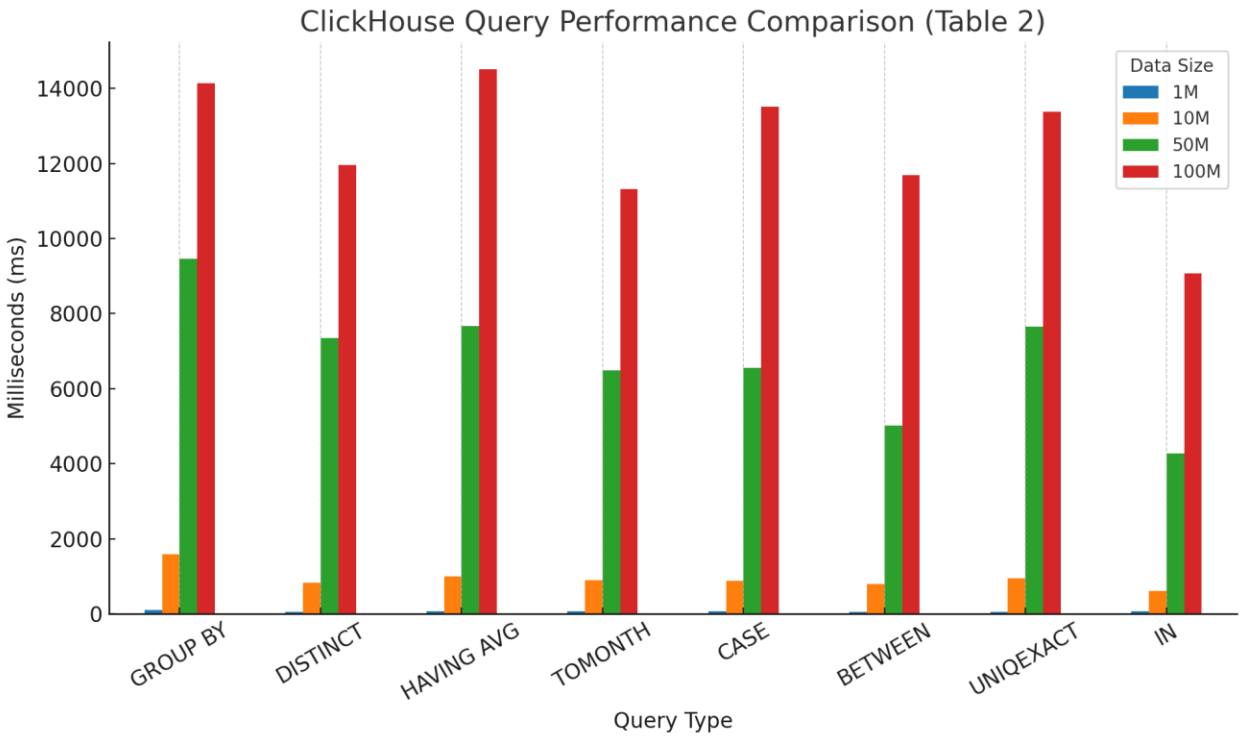

Figure 2: Column chart of queries in milliseconds Part 2

Figure 2 presents a column chart displaying the execution times, measured in milliseconds, of various SQL queries including "GROUP BY, DISTINCT, HAVING AVG, TO_MONTH, CASE, BETWEEN, UNIQEXACT, and IN" performed on datasets of different sizes (1 million, 10 million, 50 million, and 100 million records). This visual representation provides a comparative overview of how each query type performs as the dataset size increases, offering valuable insights into the scalability and efficiency of these queries under varying data loads.

The chart in Figure 2 also serves to identify which SQL operations are more sensitive to dataset size, particularly in terms of latency and computational cost. Queries such as GROUP BY and DISTINCT often involve sorting or grouping operations, which can significantly impact performance as the data volume grows. Conversely, conditions like IN or BETWEEN may show relatively stable performance across moderate data sizes but could become less efficient at scale. Understanding these performance behaviors is essential for optimizing complex queries and ensuring responsive data retrieval in large-scale database systems [15].

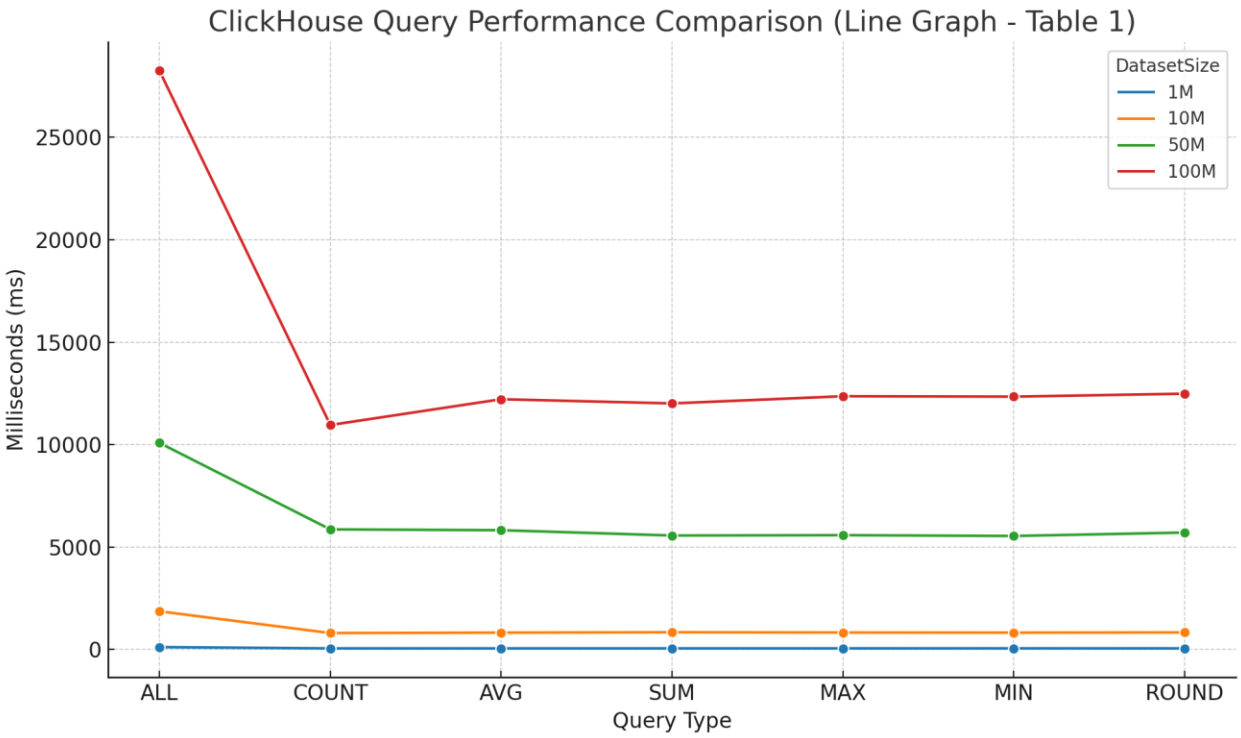

Figure 3: Line chart of queries in milliseconds Part 1

Figure 3 provides a comprehensive line chart that illustrates the execution times, in milliseconds, for a range of SQL queries namely ALL, COUNT, AVG, SUM, MAX, MIN, and ROUND across datasets of varying sizes, including 1 million, 10 million, 50 million, and 100 million records. The purpose of this visualization is to demonstrate how the computational cost of these commonly used operations scales with increasing data volume, offering a clear depiction of performance variation under different load conditions.

By analyzing the trend lines in this chart, it becomes possible to identify specific queries that exhibit linear, exponential, or irregular increases in latency as the dataset grows. This kind of performance insight is essential for database administrators and system architects, as it informs optimization decisions and supports the development of scalable, efficient data processing strategies in large-scale systems [15].

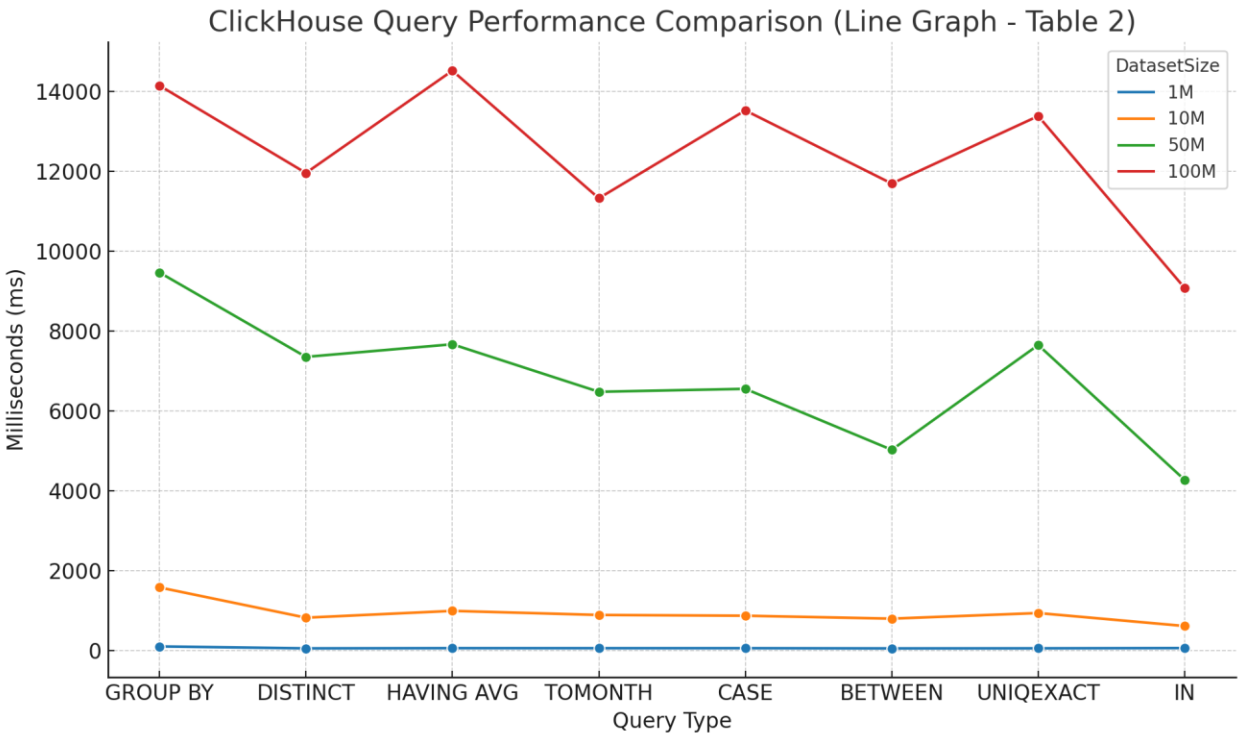

Figure 4: Line chart of queries in milliseconds Part 2

Figure 4 presents a line chart that illustrates the execution times, measured in milliseconds, for a set of SQL queries including GROUP BY, DISTINCT, HAVING AVG, TO_MONTH, CASE, BETWEEN, UNIQEXACT, and IN executed on datasets of varying sizes: 1 million, 10 million, 50 million, and 100 million records. This visualization offers a detailed perspective on how these complex query types perform under increasing data loads, providing essential insight into their scalability and responsiveness.

The line chart in Figure 4 enables the observation of performance trends specific to each query type, such as which operations are more sensitive to dataset size increases. While queries like GROUP BY and DISTINCT often involve computationally expensive grouping or sorting tasks, others like IN or BETWEEN may demonstrate more stable performance at moderate scales but deteriorate as the data volume becomes large. Such analysis is critical for designing efficient query strategies and ensuring optimal performance in high-volume database environments [15].

4. Discussion

This study evaluated the performance of ClickHouse database on 1M, 10M, 50M and 100M recorded datasets. The obtained results revealed that ClickHouse offers high performance and scalability especially in real-time data analysis scenarios. Simple SELECT and bulk operations such as AVG, SUM, MIN work linearly as the data size increases. A significant slowdown is observed in complex queries such as GROUP BY and HAVING as the data volume increases.
The performance of ClickHouse in statistical queries, even with high data volumes, shows that it

has a successful architecture in memory and disk I/O management. However, it has been observed that especially in GROUP BY and DISTINCT queries, the latencies increase significantly after 50M records, and the load on memory of sorting and grouping operations decreases performance [12].

Thanks to automation scripts, manual interventions have been minimized, and impartial and repeatable measurements have been made. The fixed hardware environment has been preserved in all tests, strengthening the comparability of the results. Some bottlenecks have been identified in CASE and nested condition queries [13].

The obtained data shows that ClickHouse provides significant superiority over traditional row-based relational databases in read-intensive workloads. However, its behaviour against write- and transaction-intensive applications is not addressed in this study, which may be an important area for future research. Non-linear increases in the time of some queries above 100M indicate that additional performance tuning and hardware optimizations may be required for very large data volumes [16].

It is of great importance for IoT, finance and e-commerce sectors that need instant data processing. This study only includes tests based on query type. Real-time system behaviours such as multi-user scenarios and instant data updates are not analysed.

Conclusions

This study demonstrated that ClickHouse delivers high performance on datasets containing millions of records, especially on read-heavy workloads. Thanks to the rigorous experimental design conducted on different data sizes and various analytical queries, it was concluded that ClickHouse performs consistently and efficiently in most scenarios.

Simple queries such as SELECT, COUNT, and other statistical operations were executed effectively even on large data sets. ClickHouse’s column-based structure increased query efficiency by reducing read latencies, validating it as a suitable solution for OLAP systems.

However, more complex queries such as GROUP BY, HAVING, and DISTINCT showed significant performance degradation when the data size exceeded 50 million, indicating bottlenecks in memory-based operations. Despite this, ClickHouse demonstrated satisfactory performance compared to traditional row-based systems.

The study also showed that performance gains diminish or begin to drop off as the data size reaches 100 million, indicating that horizontal scaling or configuration optimizations may be necessary at these scales.

The automation-based testbed provided reliable and repeatable results, increasing the real-world applicability of the study's findings.
ClickHouse is an open source and scalable solution for high-speed data processing that excels in industries that require instant data analysis, such as finance, IoT, and e-commerce [14].

Future research could examine ClickHouse's behaviour in write-intensive or mixed workloads, its performance in multi-tenant environments, and its integration with systems such as Apache Kafka or Spark.

As a result, ClickHouse is recommended as a powerful and effective alternative for applications that require fast query response on large data sets.

Acknowledgements

I would like to express my sincere gratitude to Dr. Aykut Güven for his invaluable guidance, encouragement, and academic support throughout this study. His insights have greatly contributed to the successful completion of this research.

I also extend my heartfelt thanks to all the esteemed faculty members of the Department of Computer Engineering at Beykent University for providing a rich academic environment and continuous support during my academic journey. Their contributions have played a significant role in shaping the foundation of this work.

References

[1] D. T. Montellano and S. T. D. Gravano, "High-Performance Columnar Databases for Real-Time Analytics: An Empirical Study," *Computers & Industrial Engineering*, vol. 153, 107039, 2021.
[2] Y. Liu and R. Chen, "Scaling Real-Time Analytics with ClickHouse: Storage, Query Execution, and Use Cases," *Future Generation Computer Systems*, vol. 119, pp. 556–565, 2021.
[3] L. Shaw and M. Olteanu, "Performance and Optimization Strategies in Modern Columnar Database Systems: A Comprehensive Overview," *Journal of Parallel and Distributed Computing*, vol. 157, pp. 98–108, 2021.
[4] A. Patel and A. Smith, "Columnar Data Store Performance Evaluation in Real-Time Analytics: A Case Study," *Journal of Big Data*, vol. 11, no. 3, pp. 25–44, 2023.
[5] R. Johnson and T. D. White, "Challenges and Strategies for Distributed ClickHouse Clusters," *Data & Knowledge Engineering*, vol. 141, 102898, 2023.
[6] H. Li and M. Xu, "Memory Management Optimization in Column-Oriented DBMS: Lessons from ClickHouse," *Information and Software Technology*, vol. 151, 107035, 2023.
[7] S. Gupta and M. Ahmed, "Scalable Architecture for Real-Time Analytics with Columnar DBs: A Performance Exploration," *Computers & Industrial Engineering*, vol. 187, 107934, 2024.
[8] J. Zhang, Q. Zhang, and C. Luo, "High-Speed Analytics with Columnar Databases for IoT Data Streams," Future Generation Computer Systems, vol. 134, pp. 145–158, 2023.
[9] E. Diaz, L. Herrera, and G. Ramirez, "Storage Optimization in Modern Columnar Systems: A Comparative Study," Information Systems, vol. 112, 102068, 2023.
[10] P. W. Sung, A. B. Kim, and J. Choi, "Evaluation of Real-Time Data Ingestion in Open-Source Column Stores: The Case of ClickHouse," *Journal of Parallel and Distributed Computing*, vol. 173, pp. 124–136, 2023.
[11] Amazon Web Services, "Amazon EC2: Getting Started," [Online]. Available:

https://aws.amazon.com/tr/ec2/getting-started/. [Accessed: 01-Feb-2025].
[12] R. Schulze, T. Schreiber, I. Yatsishin, R. Dahimene, and A. Milovidov, "ClickHouse - Lightning Fast Analytics for Everyone," Proc. VLDB Endow., vol. 17, pp. 3731–3744, 2024.
[13] A. K. Dwivedi, C.S. Lamba, S. Shukla, "Performance Analysis of Column Oriented Database Versus Row Oriented Database," *International Journal of Computer Applications,* vol. 50, pp. 31-34, July 2012.
[14] N. Jukic, B. Jukic, A. Sharma, S. Nestorov, and B. K. Arnold, "Expediting analytical databases with columnar approach," *Decision Support Systems,* vol. 95, pp. 61–81, Mar. 2017.
[15] ClickHouse, "Operations Overview," *ClickHouse Documentation*, 2025. [Online]. Available: https://clickhouse.com/docs/operations/overview. [Accessed: Mar. 03, 2025].
[16] M.-E. Vasile, G. Avolio, and I. Soloviev, "Evaluating InfluxDB and ClickHouse database technologies for improvements of the ATLAS operational monitoring data archiving," *ResearchGate,* 2020.