Journal of Engineering Research

ISSN 2764-1317 vol. 5, n. 9, 2025

ARTICLE 12
Acceptance date: 29/12/2025

PERFORMANCE, EFFICIENCY, AND SCALABILITY IN DATABASE MANAGEMENT SYSTEMS: A CRITICAL AND ANALYTICAL REVIEW OF SPECIALIZED RESEARCH

Marcos Borba Salomão
Master's Program in Computer Science (PMCC).
University of Campo Limpo Paulista (UNIFACCAMP), Campo Limpo Paulista/SP, Brazil.
https://lattes.cnpq.br/5849230040130429
https://orcid.org/0000-0002-1175-5922

ABSTRACT: Performance, operational efficiency, and scalability are fundamental pillars in contemporary studies of Database Management Systems (DBMSs), particularly in light of the increasing volume, heterogeneity, and velocity of data generation. This research conducts a comprehensive, critical, and analytical review of the specialized literature, examining in an integrated manner DBMS architectures, execution models, and the mechanisms employed to mitigate computational bottlenecks. The analysis delves into partitioning and fragmentation techniques, load-balancing strategies, fine-grained internal parameter optimizations, methods for data ingestion and increasing throughput of large-scale datasets, as well as solutions addressing concurrency, consistency, and transactional integrity. From an analytical perspective, the study goes beyond mapping existing approaches and investigates their behavior under different resource regimes, including memory, CPU, I/O subsystems, and storage architectures, while also considering the effects of deployment in heterogeneous, distributed, and virtualized environments. The works examined were categorized according to their methodological designs, encompassing detailed technical analyses, controlled experiments, benchmarks, and studies focused on the installation, configuration, operation, and maintenance lifecycle of DBMSs. The results of this review provide a critical synthesis of the most consolidated practices, identify recurring limitations in current solutions, and outline promising directions for future research, particularly in the context of efficient data management in diverse and large-scale computational ecosystems.

KEYWORDS: Databases; Performance; Efficiency; Scalability; Throughput; Data Management.

1. INTRODUCTION

The evolution of database systems has been driven by the growing need to manage large volumes of data with efficiency and high performance. Scalability is a fundamental aspect, as systems must be able to expand their capacity and performance in accordance with application demands. According to Stonebraker et al. (2010), scalability in database systems is directly related to a system's ability to increase its performance proportionally to the addition of resources. Performance, in turn, is essential to ensure adequate response times and the efficient execution of data query and manipulation operations (Abadi et al., 2013).

Efficiency in the use of computational resources such as memory, CPU, and I/O is a widely discussed topic in the literature. Hellerstein, Stonebraker, & Hamilton (2007) emphasize that careful optimization of database management system configurations is crucial for maximizing resource utilization and minimizing operational costs. Partitioning and load-balancing techniques are frequently employed to distribute data and queries evenly across servers, improving both scalability and performance (Das et al., 2013).

Another significant challenge is the ingestion and transfer of large volumes of data, which require efficient methods to avoid bottlenecks and ensure data integrity. According to Abadi et al. (2007), data throughput can be optimized through efficient materialization strategies, which often include advanced techniques for parallelism and data compression. Concurrency and data integrity are critical concerns, especially in high-performance environments and in workloads characterized by large volumes of simultaneous transactions (Garcia-Molina, Ullman, & Widom, 2011).

Management capability in heterogeneous platforms also emerges as a relevant aspect, since modern database systems often operate in diverse and distributed environments. Elmore et al. (2011) highlight that deployment and management efficiency in such environments depends on the flexibility and robustness of the implemented solutions.

In the systems domain, Casale (2024) examines performance engineering for in-memory databases, questioning the effectiveness of traditional performance engineering methods, such as analytical models, response surfaces, and queueing simulations, in describing such systems. The study discusses analytical models applied to performance evaluation and optimization of in-memory databases, including new response-time approximations under online analytical processing workloads. It also analyzes the relative merits of performance modeling in comparison with experimental design methods that generate response surfaces, while examining recent experiences in optimizing workload allocation in these systems.

Given the increasing complexity of the processing, distribution, and data management mechanisms observed in the reviewed literature, this research is guided by the following central question: **"How can strategies be established that promote significant advancements in performance, efficiency, scalability, and throughput in the ingestion, manipulation, and transfer of large volumes of data?".**

In addition to analyzing and classifying the technical approaches, the study seeks to synthesize the main contributions of the literature, offering a broad overview of recent advancements and identifying open research areas that may support the future development of database systems.

The central objectives of this study are: (i) to examine the techniques and theories applied, highlighting both the strengths and limitations of existing studies; and (ii) to identify gaps in the literature and the research challenges, both current and emerging.

The structure of this article follows a progressive analytical organization: Section 2 presents related reviews addressing the central research question; Section 3 describes the methodology employed in the selection and analysis of the works; Section 4 discusses the results obtained and their organization by technical categories; Section 5 deepens the discussion of these findings; and Section 6 presents the conclusions.

2. RELATED WORK

The reviews identified throughout the analysis of the collected studies are synthesized below.

Within the set of comparative analyses that inform the advancement of the field, the review conducted by Meyer et al. (2015) examines the use of database systems in mission-critical enterprise environments through a methodologically rigorous comparison (R1–R9) between a disk-based system (Alpha) and an in-memory system (Beta). Using the TPC-DS benchmark, the study highlights substantial advantages of the in-memory solution in analytical workloads, for both read operations and, to a large extent, write operations. Despite the expressive performance gains, the authors emphasize that the adoption of in-memory systems requires careful consideration of the inherent costs and limitations of memory capacity.

Among the approaches addressing the temporal dimension of data, the analysis conducted by Arora (2015) investigates temporal database models applied to the management of dynamic information in sensitive domains. The study distinguishes these systems from conventional databases by underscoring that temporal models preserve both transaction time and valid time. The field is organized according to the supported temporal dimensions (transactional, valid, and bi-temporal) and the applied timestamping techniques (tuple-based or attribute-based), discussing representative models that either extend the relational paradigm or operate through intermediate layers. Examples include TFORM, which adds temporal functionality to objects, and column-oriented temporal systems (CLTS), designed for efficient historical maintenance.

Within non-relational architectures, the approach presented by Khasawneh, Al-Sahlee, & Safia (2020) plays a structuring role by organizing the NoSQL ecosystem into four central classes: key–value, column-oriented, document-oriented, and graph-oriented. Their analysis compares these categories through fundamental criteria, notably the CAP theorem and BASE properties, articulating direct implications for consistency, availability, and partition tolerance. The study details characteristics, strengths, and limitations of each model while highlighting the need for more robust methodological instruments, emphasizing the importance of developing dedicated benchmarks capable of supporting empirical comparisons across heterogeneous NoSQL technologies.

Regarding the use of accelerated hardware in database systems, the empirical analysis by Suh et al. (2022) offers a systematic examination of the factors influencing the performance of GPU-based DBMSs. The study identifies critical determinants of query execution time and proposes a causal model capable of explaining approximately 77% of its variability, demonstrating the direct impact of reducing kernel time and data transfer operations. Their findings reveal structural limitations, including GPU memory dependency, weak operator expressiveness, scalability constraints, and recurring patterns of device underutilization, and articulate central research questions concerning the role of more advanced GPU architectures, multi-GPU configurations, and the applicability of similar analytical techniques to CPU-based DBMSs.

A comprehensive historical perspective on the evolution of database management systems is presented by Patel, Choudhary, & Patil (2023), who examine the development of the major architectures, from hierarchical and network models to RDBMS, OODBMS, ORDBMS, NoSQL, and NewSQL, over more than six decades of research. Their analysis reveals marked asymmetries in scientific production, with a strong concentration of studies on RDBMSs and limited contemporary investigation of hierarchical and network systems. The authors further discuss the recent growth of NoSQL and NewSQL technologies, driven by the need to handle large volumes of heterogeneous data, and argue that these systems still require substantial advances in areas such as concurrency control, recovery, and security. Although RDBMSs remain widely used, the study concludes that their traditional structure does not fully address the demands imposed by today's large-scale data environments.

Methodological discussions concerning performance comparisons in DBMSs gain depth in Taipalus (2024a), whose systematic review exposes recurring weaknesses in the external validity of such analyses. The study identifies structural problems such as insufficient information for experiment reproduction and the neglect of critical contextual variables, including application domain, data scale, concurrency, hardware configuration, and read/write profile, all of which strongly condition the interpretation of performance outcomes. The selection of a DBMS cannot rely solely on response times or throughput, highlighting the importance of factors such as redundancy, storage costs, consistency, and the availability of qualified professionals. The study reinforces the need for rigorous benchmarking guidelines, advocating the adoption of recognized benchmark suites and transparent disclosure of testing environments as prerequisites for replicability. By emphasizing the complexity and limitations of inter-DBMS comparisons, the review argues that critical and contextualized analyses are indispensable for informed technical decision-making.

The review developed in this research distinguishes itself from existing syntheses in two central ways. First, it focuses on identifying and systematizing strategies capable of promoting substantive advances in maximizing performance, efficiency, and scalability in DBMSs, including the analysis of throughput in processes involving the import, manipulation, and movement of large data volumes. Previously published reviews, in contrast, tend to emphasize comparative evaluations, performance tests, or targeted analyses of specific architectures. Second, most existing reviews concentrate primarily on experimental results or performance-centered approaches, without expanding the discussion toward broader data management concerns or addressing the importance of application domain requirements in shaping the properties and constraints that guide the use of a DBMS.

3. REVIEW METHODOLOGY

The structured literature review was guided by a specific directive question formulated exclusively to support the identification, selection, and classification of relevant studies: *"What approaches have been employed to computationally analyze and evaluate performance, efficiency, scalability, and data transfer rate in database systems?"* To establish a solid methodological scope, an initial exploratory search was conducted to collect essential elements for defining the investigative parameters. This process enabled the delimitation of the temporal range, the selected scientific databases, the keywords used, and the specific search fields applied across the articles.

The temporal interval defined for the review, 2007 to 2025, reflects the significant evolution of research on DBMS performance, strongly driven by technological advances over the past two decades. The search string applied was: *(Database, performance) AND (Database OR performance OR scalability OR efficiency OR data transfer rate) AND (read latency OR write latency OR update latency OR execution time).* The search was conducted across IEEE Xplore, ACM Digital Library, ScienceDirect, and Springer, considering only studies within the database domain that directly addressed the themes of this research. The initial search returned 5,830 articles. To ensure rigor and relevance, filters based on specific relevance criteria were applied, resulting in a preliminary selection of the 200 most pertinent studies.

The inclusion and exclusion criteria were defined through the initial exploratory reading and later refined according to their adherence to the scope of this review. These criteria, originally presented in Table 1, were based on the article title, abstract, keywords, and a technical evaluation of the content, ensuring an exclusive focus on approaches related to the analysis of performance, efficiency, scalability, and data transfer rate in DBMSs.

Following the detailed analysis of the studies classified as potentially relevant, a final set of 55 articles was obtained, each fully aligned with the thematic requirements of this research.

Types	Actions	Specifications
Inclusion	I-1	Research addressing analyses and definitions with results related to data in general.
	I-2	Studies presenting technical analyses of Database Management System models.
	I-3	Studies providing definitions, knowledge, and computational strategies related to Databases.
	I-4	Studies describing definitions and knowledge representations focused on data management.
	I-5	Research that introduces new approaches or methodologies for optimizing database performance.
Exclusion	E-1	Research or studies lacking the presentation or definition of any research line relevant to the topic.
	E-2	Research or studies unrelated to scalability, performance, efficiency, data volume management, database configuration strategies, CPU metrics, I/O, read latency, write latency, update latency, execution time, usability, or database flexibility.
	E-3	Research or studies not belonging to the field of computing.
	E-4	Texts that do not constitute publications with scientific value.
	E-5	Abstracts lacking technical depth or relevant content for the topic.
	E-6	Reviews that do not address the computing domain or its relationship with databases.

Structured set of inclusion and exclusion criteria applied to ensure methodological rigor, thematic alignment, and scientific relevance in the selection of studies analyzed in this review.

Table 1. Inclusion and Exclusion Criteria for Research Studies

Source: Author (2025)

4. REVIEW RESULTS

This section presents a synthetic and categorized analysis of the selected studies. Subsection A compiles solutions grounded in comparative evaluations of database systems, focusing on the identification of structural and functional characteristics, operational capabilities, performance, efficiency, scalability, data transfer rate, and advanced strategies for data creation, insertion, updating, selection, and deletion, as well as for the optimization of complex queries. Subsection B groups solutions anchored in technical studies of computational evaluation, considering metrics such as resource utilization, execution time, read latency, write latency, update latency, CPU, memory, and I/O.

4.1 Solutions Based on Database Comparisons to Identify Performance, Efficiency, and Scalability Features

The studies comprising this axis of the review highlight how different architectures and processing models shape performance and scalability patterns in contemporary data environments.

In this landscape, Pavlo et al. (2009) empirically demonstrate that native parallelism mechanisms in database systems outperform MapReduce in most structured analytical tasks, even though the latter retains advantages in heterogeneous and unstructured workloads.

The analysis by Dean & Ghemawat (2010) shows that the MapReduce model has established itself as a particularly efficient mechanism for massive unstructured data workloads, although traditional database systems remain superior in highly structured scenarios. In a different direction, the investigation conducted by Chen, Chang, & Hou (2011) indicates that well-designed Java multithreading strategies, especially in multi-core and cloud environments, significantly enhance system resource utilization, offering valuable technical insights for configuring DBMSs aimed at maximizing operational efficiency.

In the domain of scalable architectures, Cattell (2011) synthesizes a broad set of mechanisms characteristic of modern distributed systems, including asynchronous replication, row-level atomic transactions, optimistic concurrency control, automatic partitioning and replication, and fault detection and recovery routines. He observes that in many scenarios, the pursuit of scalability and availability led to the abandonment of globally ACID transactions, noting that both NoSQL solutions and new relational DBMSs are progressively increasing in maturity and relevance within the technological ecosystem.

The comparison conducted by Kulshrestha & Sachdeva (2014) between the object-oriented Db4o and the relational MySQL demonstrates that structural differences in data models directly impact operational performance: while MySQL is more efficient in read and update operations, Db4o exhibits advantages in insertions and deletions, reflecting its object-oriented modeling. The study suggests that DBMS selection must consider not only the data model but also the dominant operation profile and specific efficiency requirements.

With respect to fundamental algorithms, the analysis by Malpani & Bassi (2014) examines external sorting techniques applied to large data volumes, assessing their scalability and efficiency under different experimental conditions. The study reinforces that choices internal to sorting mechanisms have a direct impact on throughput and the cost of operations executed by DBMSs, particularly in big-data environments, thus offering relevant technical input for configuration decisions.

The discussion on high-performance engines is expanded by Neumann (2014), who compares two contrasting architectures: the disk-oriented RDF-3X system, supported by compression and B+-trees to optimize I/O, and the main-memory system HyPer, designed to simultaneously handle OLTP and OLAP workloads via mechanisms such as virtual-memory-based transactional isolation and JIT compilation of queries. The analysis demonstrates how architectural decisions, from storage strategies to query-plan optimization, shape both the latency and throughput achieved by modern systems.

The minipage scheme proposed by Ash & Lin (2014) introduces a structural refinement designed to optimize operations on SSDs, particularly in scenarios with high selectivity or mixed search-and-scan workloads. The study demonstrates substantial throughput improvements on SSDs, although the effects are less pronounced on HDDs unless selectivity is extremely low. Despite the potential increase in fragmentation, the results indicate that minipages reduce execution time and more effectively exploit the access behavior of solid-state devices.

Architectural integration across multiple data models is examined in depth by Elmore et al. (2015), who demonstrate the increasing maturity of polystore systems and highlight their utility in environments requiring coherent integration of heterogeneous data models. The analysis details internal components, memory management, transactions, and query optimization, and discusses partitioning mechanisms, load balancing, and system configuration, indicating that the effectiveness of such systems depends on carefully calibrated architectural combinations to maximize performance and flexibility.

The contrast between relational and document-oriented models is systematically analyzed in the study by Jung et al. (2015), who compare the behavior of PostgreSQL and MongoDB under increasing data volume and heterogeneity. The results show consistent advantages for MongoDB in basic insert, select, update, and delete operations, a direct consequence of its unstructured and document-oriented model. However, they also demonstrate that appropriate indexing can reduce the gap observed in PostgreSQL selection operations. As a synthesis, the authors suggest that applications with strict structural requirements benefit from the relational model, while workloads characterized by large volumes of unstructured data better exploit MongoDB's storage mechanisms.

The comparative evaluation of transactional mechanisms in OLTP environments conducted by Tongkaw & Tongkaw (2016), contrasting MariaDB and MySQL under transactional loads, reinforces the relevance of optimization and load-balancing strategies in transactional performance. The experiments show significant variations in latency and efficiency depending on configurations and techniques applied, contributing to a practical understanding of how to tune transactional systems in high-concurrency scenarios.

The NoSQL comparison by Tang & Fan (2016) broadens the landscape by examining Redis, MongoDB, Couchbase, Cassandra, and HBase in four-node clusters, evaluating throughput and execution time under multiple workload patterns. Their results show that although NoSQL systems offer superior horizontal scalability compared to traditional DBMSs, their efficiency depends heavily on alignment between the data model, required consistency, and application usage patterns. The analysis emphasizes that selecting the appropriate NoSQL system is a functional and architectural decision, not merely one based on raw performance.

Turning to the NewSQL domain, Kaur & Sachdeva (2017) characterize read, write, and update latencies as well as performance under different partitioning and load-balancing strategies. The findings show that NewSQL solutions mitigate typical limitations of traditional models but still face challenges in consolidating flexibility, scalability, and usability in high-performance environments.

The experimental analysis of MariaDB clusters by Widiono (2019) focuses on availability and integrity in distributed scenarios. The study demonstrates how different replication and partitioning strategies impact resilience and throughput, reinforcing the importance of proper configuration to ensure continuous and consistent operation in critical environments.

The work of Yang (2019) shifts the analytical focus toward the interface between web applications and DBMSs, examining inherent limitations of ORM frameworks under high-volume and low-latency workloads. The proposed methodology, grounded in static analysis and automated detection tools, identifies and corrects more than one thousand performance issues in real-world applications, demonstrating that database bottlenecks often emerge as deficiencies in application design and in the way the ORM layer interacts with the DBMS.

Performance optimization effects in relational systems are further expanded by Hairah & Budiman (2020), whose comparative study of MariaDB and PostgreSQL in inner join queries shows how join-optimization strategies can significantly reshape response times over large datasets, providing practical guidance for index selection and configuration.

The automated testing infrastructure introduced by Ingo & Daly (2020) for performance evaluation in MongoDB highlights the rising importance of continuous-integration pipelines applied to distributed systems. Automated benchmarking ensures higher experimental consistency and reproducibility, while demonstrating that architectural decisions in distributed environments must be validated through controlled and repeatable testing routines.

Relational and graph-oriented models are contrasted in the study of Do et al. (2022), which evaluates query performance in Neo4J and MySQL across four classes, selection/search, recursion, aggregation, and pattern matching, using the real-world Career Village dataset. The results show substantial advantages for Neo4J in recursive queries and in complex patterns requiring multiple joins, while the Cypher language tends to be more concise than SQL. The authors emphasize the need to expand the experimental scope across other sectors and incorporate broader performance metrics.

Advances in techniques for detecting performance degradations in DBMSs are examined by Liu et al. (2022), who demonstrate the effectiveness of AMOEBA in generating and validating semantically equivalent query pairs to reveal performance bugs in systems such as PostgreSQL and CockroachDB. The investigation spans four research questions, effectiveness (RQ1), efficiency (RQ2), the contribution of mutation rules (RQ3), and comparison with alternative approaches (RQ4), showing that mutation-driven diversity increases query complexity and improves the capacity to expose anomalies undetectable through traditional methods.

The consolidation of vector data management systems is outlined by Taipalus (2024b), who examines foundational principles, use cases, and emerging challenges associated with VDBMSs in scenarios where text, images, and videos are represented as high-dimensional vectors. The study discusses structural difficulties such as sparsity, computational costs of similarity estimation, and specialized requirements for storage and indexing, offering an organized perspective on current limitations and evolving trajectories of these systems.

A systematic overview of processing, indexing, and querying techniques in VDBMSs is presented by Pan, Wang, & Li (2024), who analyze why traditional DBMSs are inadequate for handling vector attributes characterized by imprecise semantic similarity, high dimensionality, and hybrid query requirements. The review covers similarity-scoring models, vector query types, query interfaces, in-memory and on-disk index structures, as well as approaches for compression, optimization, hardware-accelerated execution, and distributed processing.

4.2 Solutions Grounded in Technical Resource Analyses

Recent investigations into architectures and resource-management mechanisms in data management systems reveal a substantive shift in how different platforms leverage their execution and optimization strategies. The longstanding SQL–NoSQL dichotomy becomes less decisive when analytical focus moves away from languages and logical abstractions toward the structural costs that fundamentally shape performance.

This perspective is reinforced by Stonebraker (2010), whose influential analysis demonstrates that many advantages historically attributed to the NoSQL ecosystem stem not from abandoning SQL itself but from removing four major sources of structural overhead: logging, locking, latching, and buffer management. His argument shows that high performance in modern systems largely results from eliminating these traditional control mechanisms, rather than from rejecting the relational model. The study further demonstrates that significant scalability can indeed be achieved while preserving ACID guarantees, provided that the system is reorganized to avoid these systemic costs.

Efficiency debates evolve further with the integration of CPU and GPU resources for data processing. Zidan, Bonny, & Salama (2011) propose a hybrid architecture that distributes workloads between GPU and CPU according to sequence length, mitigating the underutilization of GPUs in very short executions. Their strategy delivers consistent throughput gains by enabling cooperative processing between both hardware components, constituting a low-cost approach with immediate impact on applications sensitive to intensive sequential computation.

In cloud environments, performance concerns shift toward the effects of geographically distributed replication on latency and consistency. Islam (2012) compares three consistency-maintenance mechanisms, classical models, quorum-based schemes, and a hierarchical structure built upon a variant of Dijkstra's algorithm, and shows that each performs more efficiently under specific combinations of bandwidth, packet loss, and traffic load. The study highlights that selecting an appropriate consistency mechanism is a decisive factor for sustaining real scalability in distributed database systems.

Infrastructure considerations also become critical when examining I/O patterns in NoSQL systems. Schindler (2013) shows that although these platforms typically employ local storage devices with software protocols to ensure availability, their access patterns increasingly resemble those of relational systems, largely due to a shared reliance on write-ahead logging. The essential distinction lies in the predominance of high-throughput inserts, contrasting with the multi-valued update operations characteristic of traditional RDBMS workloads.

Empirical comparisons also expose relevant nuances across systems. Parker, Poe, & Vrbsky (2013) demonstrate that MongoDB achieves superior performance in horizontally scalable scenarios, particularly for insert and read operations, whereas SQL-based systems maintain advantages when applications require strict transactional integrity.

Analyses applied to large-scale domains further deepen the assessment of technological trade-offs. Klein et al. (2015), examining MongoDB, Cassandra, and Riak across configurations ranging from single servers to nine-node clusters, report throughput levels between 225 and 3200 operations per second alongside substantial variations in read and write latency. Their results show that adopting strong consistency reduces throughput by 10–25%, highlighting the need for decisions grounded in careful balancing of consistency, availability, and partition tolerance.

Performance disparities across distributed analytical engines are examined by Wouw et al. (2015), whose micro-benchmarks on Hive, Impala, and Shark indicate that engine behavior varies significantly with workload characteristics. Impala excels in CPU-bound scenarios, while Hive exhibits constraints inherent to the Hadoop MapReduce model. Shark, by leveraging aggressive caching, delivers competitive performance and emerges as a viable option for small and medium-sized organizations seeking a balanced compromise between operational cost, scalability, and query efficiency.

The debate expands further when considering in-memory database architectures. A systematic analysis by Wang et al. (2015) of nineteen IMDB systems underscores their centrality for high-frequency transactional workloads, enabled by the elimination of I/O bottlenecks and the ability to operate with lightweight replicas and high availability. The study proposes the V3 model—Velocity, Volume, and Variety, as an evaluative framework and shows that NewSQL systems are particularly effective for time-sensitive financial operations. It also recommends a three-tiered optimization approach, memory access, kernel acceleration, and partitioning, supported by hardware-level parallelism. Empirical tests confirm the superiority of VoltDB in high-frequency scenarios, while systems such as SQLite encounter scalability limitations.

Network infrastructure emerges as a decisive factor for distributed performance as well. Binnig et al. (2016) show that high-speed networks reduce communication latency and significantly increase throughput, altering the traditional balance between local processing and inter-node communication and reshaping the design of contemporary distributed architectures.

In multi-tenant environments, in-memory systems exhibit even more complex behavior. Paluch, Kienegger, & Krcmar (2018) demonstrate that performance fluctuates according to workload intensity and heterogeneity, requiring finely tuned resource allocation to maintain adequate latency and throughput levels. Their analysis details the direct influence of CPU, memory, and I/O subsystems, emphasizing the need for adaptive management mechanisms to prevent progressive performance degradation.

Concerns surrounding data security also influence performance behavior. The experimentation conducted by Mitterer et al. (2018) with the cryptographic database ZeroDB demonstrates that the privacy-by-design model, although robust in protection, imposes significant overhead due to multiple communication rounds between client and server. Compared to traditional systems assessed through TPC-C metrics, ZeroDB experiences pronounced declines in write and update operations, especially under high-latency conditions, making it more suitable for scenarios in which security takes precedence over efficiency requirements.

The incorporation of persistent memory technologies further redefines strategies for handling commit-sensitive workloads. Tadakamadla et al. (2019) show that employing this memory class as a write cache substantially accelerates REDO log operations in Oracle systems, achieving gains of up to 22% in environments with I/O peaks and latency-sensitive conditions. The approach stands out for providing immediate improvements without requiring changes to existing applications, offering a pragmatic alternative for rapidly expanding OLTP systems.

Query acceleration and architectural optimization also reveal the expanding impact of specialized hardware on analytical workloads. Shanbhag, Madden, & Yu (2020) demonstrate through the Crystal model that GPUs produce substantial performance gains, although the magnitude of these gains varies according to operator characteristics and hardware configurations. Operations such as selection, projection, and sorting approach the memory bandwidth ratio, while joins exhibit more modest improvements. Still, the aggregate performance surpasses this ratio in typical analytical scenarios, reaching approximately a 25× improvement over CPUs, driven by vectorization limitations in the traditional operator pipeline.

The discussion on persistence and durability in in-memory databases (IMDBs) takes on new depth in the work of Lee, Kim, & Yeom (2021), who investigate the impact of validity tracking on checkpointing efficiency. By mitigating the requirement for additional memory and reducing the cost associated with periodically writing stable states, the VTC technique enhances the robustness of systems that rely on IMDBs for low-latency transactional performance.

The behavior of distributed storage systems under different network stacks is analyzed by Wang et al. (2022), who identify substantial throughput variations in workloads handling large volumes of key–value pairs. Systems operating with large KVs tend to be limited by network bandwidth, whereas workloads composed of smaller KVs become constrained by CPU capacities for packet processing. The interaction among network stack design, key–value distribution, and read/write patterns proves decisive for optimizing high-performance storage architectures.

Contemporary latency-mitigation strategies are further examined by Huang et al. (2023), whose analysis focuses on bottlenecks arising from memory access, I/O operations, synchronization, and scheduling policies. The proposed engine, MosaicDB, employs advanced latency-hiding mechanisms to reduce the impact of these factors in OLTP workloads. Experimental results demonstrate substantial improvements, especially in scenarios where storage latency and thread synchronization become dominant constraints.

Experimental approaches aimed at efficient data transfer gain new interpretation in the study by Pang & Wang (2024), which combines simulations and empirical testing to characterize the behavior of different strategies using metrics of latency, resource consumption (CPU, GPU, memory, and I/O), and scalability. The capacity to model distinct scenarios in a controlled environment, followed by practical validation, provides a precise understanding of the performance trade-offs inherent to large-scale data transfer processes.

Transactional benchmarks remain essential instruments for diagnosing bottlenecks in distributed architectures. The analysis conducted by Qu, Luyi et al. (2022) identifies significant gaps in existing benchmark suites, none of which integrate all critical factors associated with distributed transactional processing. Their evaluation of different architectures highlights recurrent chokepoints and reinforces the need for a comprehensive benchmark capable of simultaneously capturing coordination, replication, partitioning, and contention effects.

On a distinct yet equally fundamental research front, Carmeli et al. (2025) advance the theoretical foundations of Probability Databases (PDBs) in the infinite case, outlining the expressive power of representations that employ First-Order Logic (FO) views over tuple-independent PDBs (TI), a class referred to as FO(TI). Their analysis identifies deep probabilistic criteria for representability and demonstrates that the finite-moment property, although necessary, is insufficient for membership in FO(TI) – a result that exposes subtle expressive limits. Complementarily, they map the expressive power of FO fragments, showing that for infinite PDBs, hierarchies that collapse in the finite case remain strictly separated.

The complexity of distributed environments also extends to the development lifecycle, an operational gap examined by Carvalho et al. (2025), who address holistic approaches for Distributed DBMSs. They highlight the disconnection between conceptual modeling and physical deployment as a critical bottleneck. Although modeling tools (e.g., ER, UML) and deployment tools (e.g., Kubernetes, Terraform) exist, the lack of integration prevents effective automation of the transition from a unified model to a deployed system. The study consolidates this trajectory and advocates for the development of a unified modeling language and corresponding automation tools to bridge this gap.

5. DISCUSSION

Discussing the results of this review requires integrating, contrasting, and critically evaluating the approaches identified across nearly two decades of publications. The selected studies reveal an ecosystem of solutions that evolves in direct response to the continuous growth in the volume, heterogeneity, and complexity of data processed by Database Management Systems (DBMSs). These works converge toward four central analytical dimensions, data transfer strategies, methodological formalization, bottleneck identification, and performance prediction, and are distributed across twelve research domains that span from classical optimization techniques to emerging architectures oriented toward vector-based data. Table 2 synthesizes this distribution, enabling the identification of recurrent patterns, thematic gaps, and technological trends that shape the state of the art in DBMS performance. Building on this consolidation, the subsequent analysis is organized around conceptual convergences, methodological divergences, practical implications, and promising directions for future research.

The distribution of studies across the four analytical dimensions, Architecture (A), Computational Resources (R), Operational Integration (I), and Performance (P), reveals consistent patterns in how different methodological approaches investigate the mapped thematic domains. Although Table 2 provides a quantitative characterization of this relationship, the complementary graphical representation makes the structural variations between dimensions and domains explicit, enabling the observation of the relative concentration of contributions, thematic overlap among technological approaches, and the prevalence of particular investigative axes. The visualization in Figure 1 synthesizes how the studies are distributed across the twelve domains, highlighting not only the intensity of coverage in each segment but also potential asymmetries and emerging trends that shape a comprehensive understanding of the investigated landscape. This integrated perspective forms the basis for the more in-depth analyses developed in the following subsections.

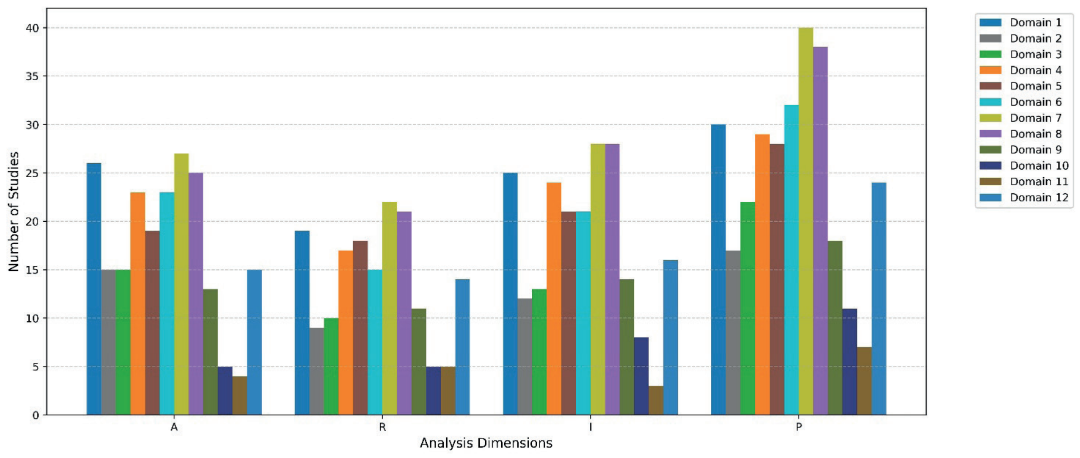
Studies	Analytical Dimensions				Research Domains												
	A	R	I	P	1	2	3	4	5	6	7	8	9	10	11	12	*
Hellerstein, Stonebraker, & Hamilton (2007)	■	■		■	■			■			■	■	■	■			
Abadi et al. (2007)	■	■		■	■			■		■	■	■			■		
Pavlo et al. (2009)	■			■				■			■	■	■				
Stonebraker (2010)	■	■	■	■	■			■	■		■	■	■		■		
Stonebraker et al. (2010)			■	■			■			■	■	■	■				
Dean & Ghemawat (2010)	■									■	■	■	■				
Garcia-Molina, Ullman, & Widom (2011)	■	■				■	■		■		■	■			■		
Elmore et al. (2011)	■	■			■	■	■	■		■	■	■	■		■		
Chen, Chang, & Hou (2011)	■	■			■						■		■	■			
Cattell (2011)	■		■	■	■		■	■	■	■	■	■	■	■			
Zidan, Bonny, & Salama (2011)	■		■	■	■			■			■	■	■		■		
Islam (2012)		■	■	■	■			■			■	■	■	■	■		
Abadi et al. (2013)	■	■	■	■	■			■	■		■	■	■				
Das et al. (2013)			■	■	■			■	■	■	■	■	■	■	■		
Schindler (2013)	■		■		■	■	■	■		■	■	■	■	■	■		
Parker, Poe, & Vrbsky (2013)	■	■	■	■	■			■	■	■	■	■	■		■		
Malpani & Bassi (2014)	■	■			■			■			■	■	■	■			
Kulshrestha & Sachdeva (2014)	■	■	■	■	■			■	■		■	■	■		■		
Neumann (2014)	■	■	■	■	■			■	■		■	■	■				
Ash & Lin (2014)	■	■		■	■						■	■					
Meyer et al. (2015)			■	■	■		■	■	■	■	■	■	■	■	■		■
Arora (2015)		■			■			■									■
Elmore et al. (2015)	■			■				■		■	■		■		■		
Klein et al. (2015)	■	■	■	■	■	■		■	■		■	■	■	■	■		
Wouw et al. (2015)			■	■	■						■	■	■	■			
Jung et al. (2015)	■			■	■		■	■	■		■	■					
Wang et al. (2015)			■	■	■		■	■	■		■	■	■	■	■		
Tongkaw & Tongkaw (2016)	■	■			■			■			■						
Binnig et al. (2016)			■	■							■	■	■				
Tang & Fan (2016)	■			■	■	■	■	■		■	■	■	■				
Kaur & Sachdeva (2017)	■			■	■			■	■		■	■			■		
Mitterer et al. (2018)	■		■	■	■	■					■	■	■		■		
Paluch, Kienegger, & Krcmar (2018)	■			■	■						■	■	■		■		
Widiono (2019)	■		■		■					■		■	■				
Yang (2019)	■	■	■	■	■	■		■	■	■							
Tadakamadla et al. (2019)	■	■	■	■	■	■	■	■	■	■	■	■	■		■		
Khasawneh, AL-Sahlee, & Safia (2020)					■		■	■	■	■	■	■	■	■	■		■
Hairah & Budiman (2020)	■		■		■	■	■	■	■		■	■	■				
Ingo & Daly (2020)		■	■	■	■		■	■	■	■	■	■	■				
Shanbhag, Madden, & Yu (2020)			■	■		■					■	■	■		■		
Lee, Kim, & Yeom (2021)		■	■	■		■			■		■	■	■				
Suh et al. (2022)	■	■	■	■	■						■	■					■

Do et al. (2022)			■	■	■	■	■				■	■					
Wang et al. (2022)		■	■	■		■				■	■	■	■		■		
Liu et al. (2022)	■		■	■	■			■			■	■					
QU, Luyi et al. (2022)			■	■	■			■	■		■		■	■			
Patel, Choudhary, & Patil (2023)					■		■	■	■	■	■	■	■	■			■
Huang et al. (2023)			■	■	■			■	■	■	■	■	■		■		
Casale (2024)			■	■	■			■		■	■	■	■	■	■		
Pan, Wang, & Li (2024)		■			■			■			■	■	■		■	■	
Taipalus (2024a)	■	■	■	■	■		■	■	■	■	■	■	■	■	■		■
Taipalus (2024b)	■			■	■	■	■				■	■	■		■	■	
Pang & Wang (2024)	■	■		■	■	■	■	■		■	■	■	■	■	■		
Carmeli et al. (2025)		■			■			■									
Carvalho et al. (2025)		■	■		■			■	■	■	■	■	■	■			
This Study	■	■	■	■	■	■	■	■	■	■	■	■	■		■		

Analytical Dimensions: (A) Analysis of data transfer strategies; (R) Representation/Formalization of the data transfer methodologies employed; (I) Identification of bottlenecks in data transfer processes; (P) Performance prediction based on different data transfer strategies. **Research Domains:** (1) Database analysis; (2) Data transfer; (3) Data volume; (4) Database techniques; (5) Database configurations; (6) Strategies; (7) Performance; (8) Efficiency; (9) Scalability; (10) Cloud and Big Data; (11) Resource usage, read latency, write latency, update latency, CPUs, GPUs, memory, and I/O; (12) Vector Database Management Systems, including metrics specific to vector data processing, execution time of vector queries, vector read and write latency, operation accuracy, and hardware/software resource utilization. **(*) Related Work:** Literature Review.

Table 2. Synthesis of the Analyzed Studies

Source: Author (2025)



The quantitative distribution of the analyzed studies, crossing the four Analytical Dimensions, Architecture (A), Resources (R), Implementation (I), and Properties (P), with the twelve Research Domains classified in the review. Each bar represents the frequency of studies simultaneously associated with a given dimension and a specific domain. This representation enables the visualization of structural patterns, the identification of thematic concentrations, and the exposure of relevant gaps in the contemporary literature on distributed database systems.

Figure 1. Correlation Between Analytical Dimensions and Thematic Domains

Source: Author (2025)

5.1 Convergence of Approaches and Core Technological Patterns

The analysis of the reviewed studies reveals recurring patterns in the solutions proposed for optimizing performance, efficiency, and scalability in Database Management Systems (DBMSs). Classical techniques, such as partitioning, replication, query optimization strategies, and index tuning, coexist with modern hardware-oriented approaches, including the use of persistent memory, GPUs, and in-memory architectures (Dean & Ghemawat, 2010; Stonebraker et al., 2010; Tadakamadla et al., 2019; Shanbhag, Madden, & Yu, 2020). This convergence suggests that, regardless of technological generation or data model, the fundamental challenges of throughput, latency, and scalability remain central to DBMS engineering.

From an architectural and operational standpoint, different paradigms, traditional SQL, NoSQL, NewSQL, and VDBMSs, share the need to balance consistency, availability, and data partitioning, in alignment with the principles of the CAP theorem (Klein et al., 2015). Recent studies show that the effectiveness of NoSQL and NewSQL systems depends not only on the flexibility of their models but also on strategies for parallelism and workload distribution (Elmore et al., 2015; Taipalus, 2024a; Patel, Choudhary, & Patil, 2023). Thus, technological convergence emerges more strongly in the mechanisms of resource management and performance optimization than in purely conceptual characteristics of data modeling.

Trade-offs among performance, scalability, and consistency form a critical axis of analysis. Empirical evaluations in distributed clusters indicate that replication strategies and consistency levels significantly affect metrics such as throughput, read/write latency, and CPU/GPU usage (Widiono, 2019; Klein et al., 2015). For instance, the adoption of strong consistency can reduce throughput by 10–25% (Klein et al., 2015), demonstrating that configuration decisions must be tailored to specific operational scenarios, considering workload characteristics, fault tolerance, and data integrity requirements.

The use of specialized hardware constitutes another decisive vector for performance. Architectures that combine CPU and GPU for cooperative processing (Zidan, Bonny, & Salama, 2011), in-memory systems (Wang et al., 2015), and persistent memory employed as a write cache (Tadakamadla et al., 2019) exhibit substantial gains in throughput and latency reduction for both analytical and transactional workloads. However, such solutions also involve operational challenges, including high infrastructure costs, the need for efficient parallelism, and adjustments to execution pipelines. These factors reinforce the importance of decision-making grounded in rigorous experimental analysis.

Methodological integration between traditional and emerging solutions points to promising directions for future research. Persistent gaps in the convergence of graph and relational databases highlight opportunities for hybrid optimization (Do et al., 2022), as well as in the exploration of high-dimensional vector data, whose structural complexity demands advanced indexing and query-processing techniques (Pan, Wang, & Li, 2024; Taipalus, 2024b). These emerging fields underscore the need for holistic approaches that simultaneously consider data modeling, computational infrastructure, and operational metrics, thereby consolidating DBMS engineering as a highly multidimensional discipline.

5.2 Practical Implications

The evidence gathered in this review indicates that the adoption of specific database solutions directly affects the efficiency, integrity, and operational robustness of systems. Studies involving MariaDB clusters, for example, show that well-configured partitioning and replication strategies substantially increase throughput and availability (Widiono, 2019). Likewise, investigations of ORM frameworks in web applications reveal that performance bottlenecks often arise from application design rather than inherent limitations of the underlying DBMS, underscoring the need for refined adjustments in the interaction between application and database layers (Yang, 2019). These findings suggest that, to maximize practical benefits, technical configuration must be aligned with the predominant workload profile and the specific architecture of the system.

The use of specialized hardware, such as GPUs and persistent memory, demonstrates significant gains in high-volume transactional and analytical environments. Recent studies show that GPU acceleration can increase aggregate performance by up to 25× in typical analytical operations, while persistent memory optimizes commit-related operations in OLTP systems, reducing latency without requiring application-level changes (Shanbhag, Madden, & Yu, 2020; Tadakamadla et al., 2019). However, integrating these resources entails substantial costs and demands compatible infrastructure, reinforcing the importance of thorough cost–benefit analysis and efficient resource--allocation strategies.

The analyses also reveal gaps in the practical application of emerging technologies, such as vector databases (VDBMSs) and probabilistic systems, which still lack maturity and adequate management tools (Taipalus, 2024b; Carmeli et al., 2025). These gaps indicate that the transfer of technical knowledge into operational settings must account for current limitations, risks of computational overhead, and the need for integration with established architectures. Thus, adopting advanced solutions requires not only a deep understanding of each technology, but also controlled implementation practices and continuous experimentation to validate their impact across diverse usage contexts.

5.3 Future Research Directions

Despite consolidated advances in the evaluation of performance, efficiency, and scalability of Database Management Systems (DBMSs), this review reveals critical gaps that delineate the scope of future research. An emerging field concerns the integration of hybrid approaches across relational databases, NoSQL, NewSQL, and VDBMSs, whose potential to serve heterogeneous workloads still lacks systematic investigation (Patel, Choudhary, & Patil, 2023; Taipalus, 2024b). Studies indicate that convergence among different data models can maximize flexibility and throughput, but it requires sophisticated strategies for partitioning, load balancing, and transactional consistency, areas that remain insufficiently explored in production scenarios.

Another critical point is the need for more robust monitoring and automated performance-anomaly detection tools for distributed databases. The use of frameworks such as AMOEBA shows that automated mechanisms can expose bottlenecks undetectable by traditional techniques, yet generalizing these approaches across multiple DBMS types remains limited (Liu et al., 2022). Progress in this direction could support proactive optimization decisions, mitigating performance degradation in environments with high concurrency and variable load.

Vector and probabilistic databases represent promising frontiers for handling complex, high-dimensional data. Research on hybrid query support, indexing optimizations in memory and on disk, as well as compression and distributed parallelism, requires additional studies to validate scalability, accuracy, and efficiency in real-world applications (Pan, Wang, & Li, 2024; Carmeli et al., 2025). Complementarily, integrating VDBMSs with established relational and NoSQL architectures still poses significant technical challenges, including resource management, data coherence, and interoperability among diverse storage mechanisms.

The analysis suggests that future research should address not only raw performance but also reliability, resilience, security, and energy sustainability. Multi-tenant scenarios, massive workloads, and cloud environments demonstrate that DBMS optimization demands a holistic approach that jointly considers hardware, software, and distributed architecture (Paluch, Kienegger, & Krcmar, 2018; Wang et al., 2022). Consolidating these elements will enable the development of systems that are more adaptable, resilient, and prepared for the increasing complexity of contemporary data.

6. CONCLUSION

The consolidation of the results presented in this review demonstrates that performance and scalability in database management systems do not depend solely on isolated solutions, but rather on the synergistic articulation of storage mechanisms, distribution strategies, consistency models, and specialized computational resources. The comparative analyses revealed that traditional and emerging approaches converge toward common structural challenges, indicating that the evolution of DBMSs requires both technical innovations and conceptual revisions regarding how data is organized and processed at scale. Given this complexity, it becomes essential to synthesize the main findings, establish evidence-based recommendations, and outline implications for future research—objectives undertaken in this final section.

6.1 Summary of Key Findings

This review systematized techniques and solutions aimed at optimizing performance and efficiency in database management systems. Established strategies such as partitioning and replication (Widiono, 2019; Klein et al., 2015) proved effective in maintaining data integrity, availability, and scalability. In parallel, emerging technologies, including GPU acceleration (Shanbhag, Madden, & Yu, 2020) and persistent memory (Tadakamadla et al., 2019), demonstrated substantial performance gains, particularly in high-intensity analytical scenarios and distributed workloads.

6.2 Recommendations

It is recommended that consolidated techniques for partitioning and replication be adopted in database clusters to maximize availability and integrity, as evidenced by Widiono (2019) and Klein et al. (2015). The incorporation of specialized hardware, such as GPUs (Shanbhag, Madden, & Yu, 2020) and persistent memory (Tadakamadla et al., 2019), offers significant performance gains but requires careful evaluation of cost and operational complexity.

For future investigations, emphasis should be placed on integrating graph and relational databases (Do et al., 2022) and advancing robust tools for anomaly detection and performance-bug diagnosis (Liu et al., 2022). Within emerging perspectives, the study and enhancement of VDBMSs and vector-data management techniques (Taipalus, 2024b; Pan, Wang, & Li, 2024) represent promising areas to address the growing complexity of multimodal and high-dimensional data.

6.3 Final Considerations

The evolution of distributed databases and emerging systems such as NewSQL and VDBMSs represents a strategic response to contemporary demands for high availability, consistency, and performance (Patel, Choudhary, & Patil, 2023; Taipalus, 2024b). The solutions identified in this review illustrate technically viable paths for optimizing throughput, latency, and scalability. However, critical challenges persist, including the integration of heterogeneous architectures, operational complexity, and limitations in high-volume multimodal data environments. Continued empirical studies, controlled simulations, and the development of diagnostic and optimization tools are therefore essential to ensure that database systems remain resilient and adaptable to future needs.

REFERENCES

Abadi, D. J., Boncz, P., Harizopoulos, S., Idreos, S., & Madden, S. (2013). The design and implementation of modern column-oriented database systems. Now Publishers. https://doi.org/10.1561/1900000024

Abadi, D. J., Myers, D. S., DeWitt, D. J., & Madden, S. R. (2007). Materialization strategies in a column-oriented DBMS. In IEEE 23rd International Conference on Data Engineering (pp. 466–475). IEEE. https://doi.org/10.1109/ICDE.2007.367892

Arora, S. (2015). A comparative study on temporal database models: A survey. In International Symposium on Advanced Computing and Communication (pp. 161–167). IEEE. https://doi.org/10.1109/ISACC.2015.7377335

Ash, S. M., & Lin, K.-I. (2014). Optimizing database index performance for solid state drives. In Proceedings of the 18th International Database Engineering & Applications Symposium (IDEAS'14) (pp. 237–246). ACM. https://doi.org/10.1145/2628194.2628255

Binnig, C., Crotty, A., Galakatos, A., Kraska, T., & Zamanian, E. (2016). The end of slow networks: It's time for a redesign. Proceedings of the VLDB Endowment, 9(7), 528–539. https://doi.org/10.14778/2904483.2904485

Carvalho, G., Bernardino, J., Cabral, B., & Pereira, V. (2025). A survey of holistic approaches for distributed database systems: From conceptual model to deployment. IEEE Access, 13, 120830–120851. https://doi.org/10.1109/ACCESS.2025.3587670

Carmeli, N., Grohe, M., Lindner, P., & Standke, C. (2025). Tuple-independent representations of infinite probabilistic databases. ACM Transactions on Database Systems. https://doi.org/10.1145/3771733

Cattell, R. (2011). Scalable SQL and NoSQL data stores. ACM SIGMOD Record, 39(4), 12–27. https://doi.org/10.1145/1978915.1978919

Casale, G. (2024). Performance engineering for in-memory databases: Models, experiments and optimization. In Proceedings of the 15th ACM/SPEC International Conference on Performance Engineering (ICPE'24). https://doi.org/10.1145/3629526.3649131

Chen, K.-Y., Chang, J. M., & Hou, T.-W. (2011). Multithreading in Java: Performance and scalability on multicore systems. IEEE Transactions on Computers, 60(11), 1521–1534. https://doi.org/10.1109/TC.2010.232

Das, S., Agrawal, D., & El Abbadi, A. (2013). ElasTraS: An elastic, scalable, and self-managing transactional database for the cloud. ACM Transactions on Database Systems, 38(1), 5. https://doi.org/10.1145/2445583.2445588

Dean, J., & Ghemawat, S. (2010). MapReduce: A flexible data processing tool. Communications of the ACM, 53(1), 72–77. https://doi.org/10.1145/1629175.1629198

Do, T.-T.-T., Mai-Hoang, T.-B., Nguyen, V.-Q., & Huynh, Q.-T. (2022). Query-based performance comparison of graph database and relational database. In Proceedings of the 11th International Symposium on Information and Communication Technology (SoICT'22) (pp. 375–381). ACM. https://doi.org/10.1145/3568562.3568648

Elmore, A. J., Das, S., Agrawal, D., & El Abbadi, A. (2011). Zephyr: Live migration in shared-nothing databases for elastic cloud platforms. In Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data (pp. 301–312). ACM. https://doi.org/10.1145/1989323.1989356

Elmore, A. J., Duggan, J., Stonebraker, M. R., Balazinska, M., Çetintemel, U., Gadepally, V., ... & Zdonik, S. B. (2015). A demonstration of the BigDAWG polystore system. Proceedings of the VLDB Endowment, 8(12), 1908–1911. https://doi.org/10.14778/2824032.2824098

Garcia-Molina, H., Ullman, J. D., & Widom, J. (2011). Database systems: The complete book (2nd ed.). Pearson.

Hairah, U., Budiman, E. (2021). Inner join query performance: MariaDB vs PostgreSQL. In 2020 2nd International Conference on Science & Technology (ICOST). Journal of Physics: Conference Series, 1844(1), 012021. https://doi.org/10.1088/1742-6596/1844/1/012021

Hellerstein, J. M., Stonebraker, M., & Hamilton, J. (2007). Architecture of a database system. Foundations and Trends in Databases, 1(2), 141–259. https://doi.org/10.1561/1900000002

Huang, K., Wang, T., Zhou, Q., & Meng, Q. (2023). The art of latency hiding in modern database engines. Proceedings of the VLDB Endowment, 17(3), 577–590. https://doi.org/10.14778/3632093.3632117

Ingo, H., & Daly, D. (2020). Automated system performance testing at MongoDB. In Proceedings of the Workshop on Testing Database Systems (DBTest'20) (pp. 1–6). https://doi.org/10.1145/3395032.3395323

Islam, M. A. (2012). Performance comparison of consistency maintenance techniques for cloud database. In Proceedings of the 50th Annual Southeast Regional Conference (ACM-SE'12) (pp. 399–400). ACM. https://doi.org/10.1145/2184512.2184624

Jung, M.-G., Youn, S.-A., Bae, J., & Choi, Y.-L. (2015). A study on data input and output performance comparison of MongoDB and PostgreSQL in the big data environment. In 8th International Conference on Database Theory and Application (DTA) (pp. 14–17). IEEE. https://doi.org/10.1109/DTA.2015.14

Kaur, K., & Sachdeva, M. (2017). Performance evaluation of NewSQL databases. In International Conference on Inventive Systems and Control (ICISC) (pp. 1–5). IEEE. https://doi.org/10.1109/ICISC.2017.8068585

Khasawneh, T. N., Al-Sahlee, M. H., & Safia, A. A. (2020). SQL, NewSQL, and NoSQL databases: A comparative survey. In 11th International Conference on Information and Communication Systems (ICICS) (pp. 13–21). IEEE. https://doi.org/10.1109/ICICS49469.2020.239513

Klein, J., Gorton, I., Ernst, N., Donohoe, P., Pham, K., & Matser, C. (2015). Performance evaluation of NoSQL databases: A case study. In Proceedings of the 1st Workshop on Performance Analysis of Big Data Systems (PABS'15) (pp. 5–10). ACM. https://doi.org/10.1145/2694730.2694731

Lee, K., Kim, H., & Yeom, H. Y. (2021). Validity tracking–based log management for in-memory databases. IEEE Access, 9, 111493–111504. https://doi.org/10.1109/ACCESS.2021.3103862

Liu, X., Zhou, Q., Arulraj, J., & Orso, A. (2022). Automatic detection of performance bugs in database systems using equivalent queries. In Proceedings of the 44th International Conference on Software Engineering (ICSE'22) (pp. 225–236). ACM. https://doi.org/10.1145/3510003.3510093

Malpani, P., & Bassi, P. (2014). Analytical & empirical analysis of external sorting algorithms. In International Conference on Data Mining and Intelligent Computing (ICDMIC) (pp. 1–6). IEEE. https://doi.org/10.1109/ICDMIC.2014.6954224

Meyer, R., Banova, V., Danciu, A., Prutscher, D., & Krcmar, H. (2015). Assessing the suitability of in-memory databases in an enterprise context. In International Conference on Enterprise Systems (ES) (pp. 78–89). IEEE. https://doi.org/10.1109/ES.2015.15

Mitterer, M., Niedermayer, H., von Maltitz, M., & Carle, G. (2018). An experimental performance analysis of the cryptographic database ZeroDB. In Proceedings of the 1st Workshop on Privacy by Design in Distributed Systems (W-P2DS'18) (Article 6, pp. 1–5). ACM. https://doi.org/10.1145/3195258.3195264

Neumann, T. (2014). Engineering high-performance database engines. Proceedings of the VLDB Endowment, 7(13), 1734–1740. https://doi.org/10.14778/2733004.2733076

Paluch, D., Kienegger, H., & Krcmar, H. (2018). A workload-dependent performance analysis of an in-memory database in a multi-tenant configuration. In Companion of the 2018 ACM/SPEC International Conference on Performance Engineering (pp. 131–134). ACM. https://doi.org/10.1145/3185768.3186290

Pan, J. J., Wang, J., & Li, G. (2024). Vector database management techniques and systems. In Companion of the 2024 International Conference on Management of Data (pp. 597–604). ACM. https://doi.org/10.1145/3626246.3654691

Pang, X., & Wang, J. (2024). Understanding the performance implications of the design principles in storage-disaggregated databases. Proceedings of the ACM on Management of Data, 2(3), Article 180. https://doi.org/10.1145/3654983

Parker, Z., Poe, S., & Vrbsky, S. V. (2013). Comparing NoSQL MongoDB to an SQL DB. In Proceedings of the 51st ACM Southeast Conference (ACMSE'13). https://doi.org/10.1145/2498328.2500047

Patel, S., Choudhary, J., & Patil, G. (2023). Revolution of database management system: A literature survey. International Journal of Engineering Trends and Technology, 71(7), 189–200. https://doi.org/10.14445/22315381/IJETT-V71I7P218

Pavlo, A., Paulson, E., Rasin, A., Abadi, D. J., DeWitt, D. J., Madden, S., & Stonebraker, M. (2009). A comparison of approaches to large-scale data analysis. In Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data (pp. 165–178). ACM. https://doi.org/10.1145/1559845.1559865

Qu, L., Wang, Q., Chen, T., Li, K., Zhang, R., Zhou, X., Xu, Q., Yang, Z., Yang, C., Qian, W., & Zhou, A. (2022). Are current benchmarks adequate to evaluate distributed transactional databases? BenchCouncil Transactions on Benchmarks, Standards and Evaluations, 2(1), Article 100031. https://doi.org/10.1016/j.tbench.2022.100031

Schindler, J. (2013). Profiling and analyzing the I/O performance of NoSQL DBs. In Proceedings of the ACM SIGMETRICS/International Conference on Measurement and Modeling of Computer Systems (pp. 389–390). ACM. https://doi.org/10.1145/2465529.2479782

Shanbhag, A., Madden, S., & Yu, X. (2020). A study of the fundamental performance characteristics of GPUs and CPUs for database analytics. In Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data (pp. 1617–1632). ACM. https://doi.org/10.1145/3318464.3380595

SIGMETRICS/PERFORMANCE '22. (2022). SIGMETRICS/IFIP Performance Joint International Conference on Measurement and Modeling of Computer Systems (pp. 51–52). ACM. https://doi.org/10.1145/3489048.3522644

Stonebraker, M. (2010). SQL databases v. NoSQL databases. Communications of the ACM, 53(4), 10–11. https://doi.org/10.1145/1721654.1721659

Stonebraker, M., Abadi, D., DeWitt, D. J., Madden, S., Paulson, E., Pavlo, A., & Rasin, A. (2010). MapReduce and parallel DBMSs: Friends or foes? Communications of the ACM, 53(1), 64–71. https://doi.org/10.1145/1629175.1629197

Suh, Y.-K., An, J., Tak, B., & Na, G.-J. (2022). A comprehensive empirical study of query performance across GPU DBMSes. In Abstract Proceedings of the 2022 ACM.

Taipalus, T. (2024a). Database management system performance comparisons: A systematic literature review. Journal of Systems and Software, 208, Article 111872. https://doi.org/10.1016/j.jss.2023.111872

Taipalus, T. (2024b). Vector database management systems: Fundamental concepts, use-cases, and current challenges. Cognitive Systems Research, 85. https://doi.org/10.1016/j.cogsys.2024.101216

Tadakamadla, R., Patocka, M., Kani, T., & Norton, S. J. (2019). Accelerating database workloads with DM-WriteCache and persistent memory. In Proceedings of the 2019 ACM/SPEC International Conference on Performance Engineering (ICPE'19) (pp. 255–263). ACM. https://doi.org/10.1145/3297663.3309669

Tang, E., & Fan, Y. (2016). Performance comparison between five NoSQL databases. In 7th International Conference on Cloud Computing and Big Data (CCBD) (pp. 105–109). IEEE. https://doi.org/10.1109/CCBD.2016.030

Tongkaw, S., & Tongkaw, A. (2016). A comparison of database performance of MariaDB and MySQL with OLTP workload. In IEEE Conference on Open Systems (ICOS) (pp. 117–119). IEEE. https://doi.org/10.1109/ICOS.2016.7881999

van Wouw, S., Viña, J., Iosup, A., & Epema, D. (2015). An empirical performance evaluation of distributed SQL query engines. In Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering (ICPE'15) (pp. 123–131). ACM. https://doi.org/10.1145/2668930.2688053

Wang, Y., Yu, M., Hui, Y., Zhou, F., Huang, Y., Zhu, R., Ren, X., Li, T., & Lu, X. (2022). A study of database performance sensitivity to experiment settings. Proceedings of the VLDB Endowment, 15(7), 1439–1452. https://doi.org/10.14778/3523210.3523221

Wang, Y., Zhong, G., Kun, L., Wang, L., Kai, H., Guo, F., Liu, C., & Dong, X. (2015). The performance survey of in-memory database. In 2015 IEEE 21st International Conference on Parallel and Distributed Systems (ICPADS) (pp. 815–820). IEEE. https://doi.org/10.1109/ICPADS.2015.109

Widiono, S. (2019). Experiments and descriptive analysis in the MariaDB database cluster system to prepare data availability. International Journal of Engineering, Technology and Natural Sciences, 1(1). https://doi.org/10.46923/ijets.v1i1.24

Yang, J. (2019). Improving performance and quality of database-backed software. In Proceedings Companion of the 2019 ACM SIGPLAN International Conference on Systems, Programming, Languages, and Applications: Software for Humanity (pp. 23–25). ACM. https://doi.org/10.1145/3359061.3361076

Zidan, M. A., Bonny, T., & Salama, K. N. (2011). High performance technique for database applications using a hybrid GPU/CPU platform. In GLSVLSI'11: Proceedings of the 21st Great Lakes Symposium on VLSI (pp. 85–90). ACM. https://doi.org/10.1145/1973009.1973027