International Journal of Latest Engineering and Management Research (IJLEMR)
ISSN: 2455-4847
www.ijlemr.com || Volume 10 – Issue 11 || November 2025 || PP. 01-04

Relational Databases: Assessing Potential in Modern Web Applications

Smirnov Andrei
Master's degree, Perm national research polytechnic university, Perm, Russia

Abstract: This article examines the role of relational databases in modern web applications, analyzing their advantages and limitations amid increasing computational demands and evolving architectural approaches. Special attention is given to scalability, performance, and integration with alternative solutions such as NoSQL storage. The paper explores contemporary optimization methods for relational DBMS, including distributed architectures, in-memory technologies, and machine learning applications. The study concludes that despite existing challenges, relational databases continue to evolve, remaining competitive in high-load and scalable systems.

Keywords: Relational Databases, Web Applications, Scalability, Optimization, SQL, NoSQL.

I. Introduction

Modern web applications place high demands on databases in terms of scalability, fault tolerance, and performance. For such applications, relational databases (RDB) are a key component of the infrastructure since they guarantee tight structuring and data integrity. Yet growing volumes of data, the advancement of distributed systems, and the emergence of alternative paradigms for data management question the long-term appropriateness of the relational model. In the last two years, graph databases and NoSQL solutions have seen growing adoption, making it necessary to critically assess the strengths and weaknesses of conventional database management system (DBMS) in the current scenario.

The topicality of this research is emphasized by the necessity to evaluate the potential of RDB in web technology development. Despite common opinions on their obsolescence, relational database systems are still developing and discovering new applications to counteract arising issues. The development of cloud computing, the introduction of hybrid data management paradigms, and the development of SQL query optimization algorithms allow relational DBMS to remain competitive. However, issues of scalability, support for unstructured data, and support for modern architectural styles, such as the microservice architecture, require additional research.

The objective of this paper is to evaluate the web development capabilities of RDB. Current techniques designed to improve the efficiency of RDB and how they can continue to grow to accommodate higher computation needs in the future are described.

II. Relational Databases in the Context of Modern Web Applications

The growth of web technologies and the volume of processed data has increased the demand for DBMS, which, due to their strict schema and data integrity, remain preferred in web programming. They form the basis for most enterprise software, social networks, online stores, and financial sites. Although the popularity of NoSQL solutions is on the rise, RDB remain important since they provide ACID transactions, which is crucial for the majority of the business processes [1].

Among the central reasons for the popularity of the relational model is a mature environment that surrounds DBMS such as PostgreSQL, MySQL, Microsoft SQL Server, and Oracle. All of these systems have a developed indexing system, support for analytics queries of great complexity, and built-in replication features. However, in the contemporary web application sphere, where high data processing speed and horizontal scaling are required, traditional databases have a number of limitations. Vertical scaling, in particular, meaning the increase in computing resources of a single server, is limited by physical constraints and leads to higher costs. In this connection, an intensively developed distributed architecture makes it possible to configure server clusters and balance the load between nodes [2].

The second issue of RDB is concerning the need to deal with unstructured and poorly structured data, which are most often used in modern web applications. For example, social network data, multimedia data, and user profiles do not typically adhere to rigid relational schemes. Here, programmers combine DBMS with NoSQL stores, such as MongoDB or Cassandra, via hybrid methods. This allows it to combine the advantages of structured storage with excellent flexibility when dealing with documents and data graphs.

Despite limitations, RDB continue to evolve. In the past couple of years, there has been increasing support for the JSON data format, improved caching, and integration of intelligent indexing algorithms. For

example, PostgreSQL natively supports JSONB, which allows you to work with semi-structured data efficiently without sacrificing query execution speed. The same can be observed in MySQL and other popular DBMS. Moreover, in heavily loaded web applications, techniques of microservice architecture optimization are widely adopted, e.g., database-level caching, load balancing, and asynchronous replication [3].

Thus, RDB remain a necessary tool for web application development, even if scalability and dealing with unstructured data are issues for them. Their adaptation to novel demands, like integration with cloud and NoSQL technologies, attests to their enormous potential for evolution.

III. Analysis of DBMS Performance and Scalability

The performance of a DBMS in web applications depends on various factors, including storage architecture, query optimization, and the indexing techniques applied. With increasing levels of information and high-performance requirements, the ability of a DBMS to perform queries efficiently in both a standalone server environment as well as with distributed load is becoming a vital consideration. Horizontal scaling, where data is divided into segments and the load shared among a sequence of servers, is also an emerging issue for ensuring fault tolerance and high availability [4].

One of the limiting factors in RDB is performing numerous concurrent queries. Under high loads, regular locking and transaction consistency algorithms will slow down performance. In response to these, modern DBMS implement adaptive competitive access control strategies such as Multi-Version Concurrency Control (MVCC), through which you can reduce conflicts when performing operations concurrently. Specifically, PostgreSQL employs MVCC to implement high-performance transaction processing without requiring hard locks, which is particularly significant in web applications with the highest level of user interaction [5].

Index optimization is another key area of performance improvement that allows you to speed up complex query processing. The use of B-trees, hash indexes, and bitmap indexes significantly reduces data sampling time, especially in analytic systems. However, their improper use accelerates their update overhead, and therefore, requires an equilibrium approach when designing them. New DBMS, including MySQL and SQL Server, also have built-in facilities for automatically analyzing and reconstructing indexes to let you modify their structure dynamically in response to workloads. Furthermore, application development for mobile platforms consciously adopts mechanisms for optimizing mobile client performance, e.g., offline cache utilization, keeping the number of server requests small, and the use of compact data representations [6].

Relational database scalability remains one of the principal problems, especially if distributed systems are taken into account. In contrast with NoSQL storage, built with support for clusters from the beginning, traditional DBSS require advanced sharding mechanisms to distribute the data among servers effectively. Sharding reduces node loads, but it makes it harder to preserve relationships between data. Alternatively, big cloud providers like Amazon RDS and Google Cloud SQL provide automated distributed data storage and processing solutions, which simplifies scaling without radical changes in the application architecture.

Relational database performance is solely reliant on query optimization algorithms, competitive access models, and scalability mechanisms. New advancements, such as MVCC, adaptive indexes, and cloud distributed storage, allow RDBMS to compete even in web applications with high traffic despite existing architectural limitations.

IV. Advantages and Disadvantages of Relational Databases

As mentioned earlier, the DBMS is one of the most widely used tools. Especially for managing structured data in web applications due to its stability, good data structuring, and support for transactional integrity. The strength of the relational model is the compliance with the ACID properties (Atomicity, Consistency, Isolation, Durability), which guarantee the correctness of transactions even in heavy loads. This makes RDB an important component in mission-critical uses such as banking systems, financial platforms, and business applications where data loss is not an option (table 1).

Table 1: The impact of database architecture on performance and scalability

Architectural aspect	Impact on performance	Impact on scalability
Indexes	Speed up searches but slow down inserts	Require monitoring and restructuring
Sharding	Can reduce latency	Complicates transaction management
Caching	Increases speed for repeated queries	Requires additional control
Replication	Improves fault tolerance	Increases synchronization overhead
Partitioning	Reduces load on individual data segments	Improves load balancing but requires configuration
In-memory storage	Significantly speeds up data processing	Limited by available RAM capacity

Another major advantage is that of the broad coverage of the standard SQL language that simplifies the generation of complex queries and table joins. All current DBMS like PostgreSQL, MySQL, and Oracle have extremely efficient SQL query optimization systems that can boost data processing speed through indexing, caching, and parallel processing. In addition, high standardization enables RDB to be implemented on various programming languages and platforms, which is important for large web applications.

Despite all its pros, RDB also have firm cons. Among its main limitations, the scalability problem stands out. Unlike NoSQL databases, which are initially designed to run in distributed scenarios, common databases require advanced replication and sharding tools to handle data distribution properly. This increases the cost of management and the consumption of resources for load balancing. For that reason, big businesses working with highly loaded systems use relational and non-RDB together through hybrid architectures.

Another issue is the inflexibility of the data schema, which constrains their dynamic adjustment. In web applications where the structure of information can be changed, RDB require expensive migration procedures, leading to additional time costs. In comparison, NoSQL databases such as MongoDB offer more flexibility, where you can store unstructured and semi-structured data with fewer restrictions on their shape.

Thus, RDB possess several merits, including reliability, transactional integrity, and sophisticated mechanisms for working with SQL queries. However, their flaws, like the inconvenience of horizontal growth and the inflexibility of the data schema, limit their usage in dynamically creating web applications. Nevertheless, RDB are continually evolving, suggesting novel optimizing mechanisms and integration with other technologies in order to become more competitive.

V. Prospects for the Development of Relational Databases

Maintaining RDB in accordance with contemporary challenges requires adaptation to increasing data volumes, distributed processing environments, and the need to support diverse data types. Among the notable areas of advancement is the advent of hybrid models of storage where the strengths of a relational model are combined with the ease of NoSQL solutions. Modern DBMS, such as PostgreSQL and MySQL, already possess JSON document support built-in, making them more flexible when dealing with semi-structured data.

Database resource automation is important. Improvements in artificial intelligence and machine learning algorithms have made it possible to implement intelligent mechanisms of query optimization, load forecast, and index auto-tuning. Such AI-driven systems with adaptive optimization drivers facilitate dynamic profiling of typical workload patterns and dynamically redesign SQL query execution plans at runtime for speeding up their run times without the DBA involvement.

Another thrilling area is distributed relational database architecture. Classical approaches to horizontal scaling, such as sharding, have significant configuration and maintenance costs. In response to this, large cloud vendors such as Amazon Aurora and Google Cloud Spanner are constructing natively distributed RDB with high availability and fault tolerance but without the hassle of infrastructure management [7]. These products enable web applications to scale well with lower transaction latency.

One more line of development is the performance improvement through the implementation of in-memory technologies allowing you to speed up the data processing by locating frequently accessed records in RAM (fig. 1).

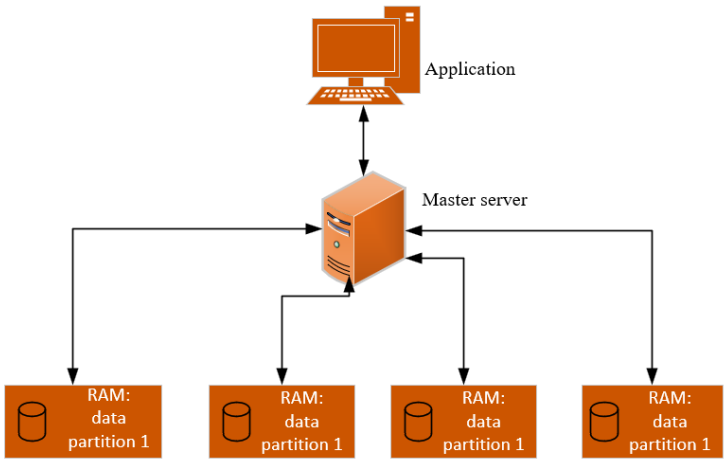

Fig. 1: In-memory database architecture [8]

Solutions like those implemented in Microsoft SQL Server (In-Memory OLTP) and Oracle Database (TimesTen) dramatically increase the efficiency of transaction and analytical query performance. In the future,

further refinement of these technologies is expected, enabling RDBMS to remain competitive with high-performance NoSQL storage systems.

Thus, RDB remain not just indispensable but also evolving in line with contemporary web development demands. Hybrid storage schemes, the emergence of machine learning technologies, distributed architectures, and the use of in-memory solutions ensure RDBMS to remain competitive and up-to-date even for loaded and scalable applications.

VI. Conclusion

RDB continue to be popular in web development due to their reliability, consistency of data, and support for complex transactions. Despite the threat posed by NoSQL alternatives, RDB remain relevant since they continue to evolve: improvement of indexing mechanisms, addition of in-memory technologies, and support for hybrid data storage model. Their good standardization and existence of a mature tool ecosystem enable RDB to be more easily embedded in modern solutions like cloud and microservice solutions. However, their limitations, like the fact that it is hard to scale them horizontally and they are very costly to implement unstructured data in, compel one to look for compromise solutions like hybrid deployment of different types of databases.

The future of RDB is associated with ongoing evolution of automated optimization capabilities, use of machine learning-based algorithms for load prediction and dynamic query rewriting. In addition, active deployment of distributed RDB, such as Google Cloud Spanner and Amazon Aurora, enables bypassing the constraints of the classic architecture. Thus, in spite of the difficulties, RDB are still evolving to meet the demands of heavily loaded systems, holding their own in the fast-paced era of technology.

References

[1] S. Sethi, S. Panda. SQL or NoSQL – Practical Aspect and Rational behind Choosing Data Stores, *Journal of Computer and Communications,12(8),* 2024, 155-174.

[2] F. Li, X. Zhou, P. Cai, R. Zhang, G. Huang, X. Liu. Integration of Cloud-Native and Distributed Architectures,*In Cloud Native Database: Principle and Practice*, 2025, 199-215.

[3] S. Bolgov. Optimizing microservices architecture performance in fintech projects, *Bulletin of the Voronezh Institute of High Technologies*, *19(1),* 2025. URL: https://vestnikvivt.ru/ru/journal/pdf?id=1401

[4] A. Uzzaman, M. Jim, N. Nishat, J. Nahar. Optimizing SQL databases for big data workloads: techniques and best practices, *Academic Journal on Business Administration, Innovation & Sustainability*, 4(3), 2024, 15-29.

[5] V. Vadlamani. MVCC, *In Postgre SQL Skills Development on Cloud*, 2024, 547-561.

[6] A. Blazhkovskii. Optimization of mobile application performance: modern approaches and methods, *ISJ Theoretical & Applied Science,140(12),* 2024, 290-294.

[7] Distributed Database System / Geeks for Geeks // URL: https://www.geeksforgeeks.org/distributed-database-system/ (date of application: 14.09.2025).

[8] M. Ahn, T. Willhalm, N. May, D. Lee, S. Desai, D. Booss, O. Rebholz. An Examination of CXL Memory Use Cases for In-Memory Database Management Systems using SAP HANA, *Proceedings of the VLDB Endowment,17(12),* 2024, 3827-3840.