

Enhancing Legacy Relational Databases via NoSQL Integration

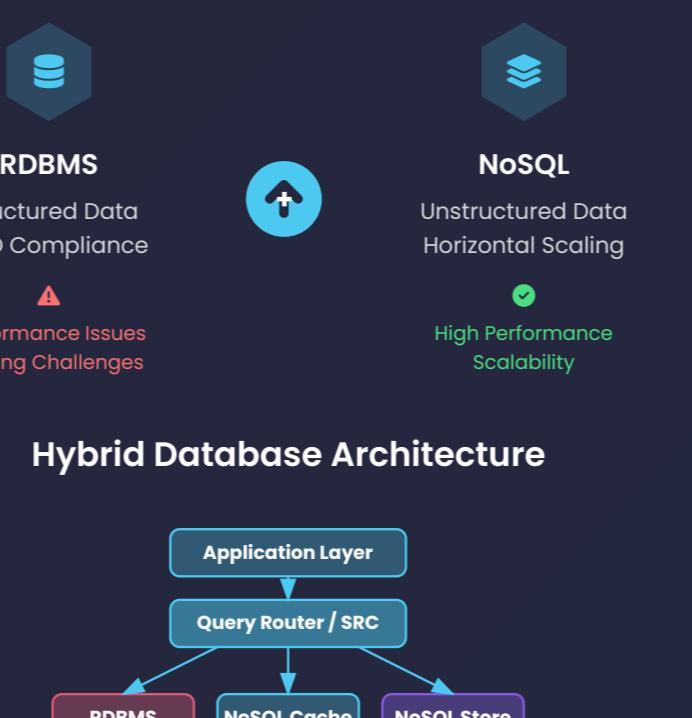
A Comparative Analysis of Hybrid Database Solutions

Research on Optimizing Query Performance and Scalability



Research Problem

Managing large or less structured data with legacy RDBMS is tough, but NoSQL systems are built to handle a large number of requests easily. This research studies how to use NoSQL approaches (caching, multi-codec strategy and mixed structures) to guarantee quick and effective work of relational databases.



Paper 1: Hybrid Cloud Integration

Ragunathan et al., 2025

Methodology

Created a hybrid database system by joining a relational store with a NoSQL system; used CloudSim to model workload and study the performance.

Using the simulation, we were able to run different experiments with workloads and scaling.

Focus

Evaluated query execution speed, latency, and scalability of the hybrid model versus standalone solutions (SQL-only or NoSQL-only).

Key Findings



Advantages

- Clear performance gains in simulated environments
- Insights on workload partitioning
- Detailed resource usage analysis

Limitations

- Results from simulations, not real systems
- Data consistency overhead not deeply evaluated
- High-level model may miss real-world issues

Comparison of Approaches

Aspect	Ragunathan et al. (Hybrid Cloud)	Bjeladinović (Statement Rewriting)
Scope	Architecture-level integration focusing on high-level metrics (latency, throughput, scalability)	Query-level optimization within a hybrid DB, focusing on execution time of specific operations
Method	Simulation (CloudSim) to model hybrid cloud interactions	Implementation and measurement on concrete system using real DBMS
Performance Gains	~30% gain in query speed	Multi-fold reductions in execution time for bulk DML operations
Integration Strategy	Emphasizes splitting workloads by data type	Emphasizes rewriting queries to the best underlying storage system
Trade-offs	Hybrid complexity and consistency challenges in theory	SRC may not always reduce time on small inputs and needs more work (e.g. support for SELECT)

Supporting Evidence

Caching Layers

Benefits are also present when using caching ideas. Privalov & Stupina (2024) discovered that a faster web app was achieved by using a Redis cache before a MySQL relational DB. This supports the earlier recommendation by holding common queries in NoSQL instead of relying on SQL.



Polyglot Principles

According to polyglot persistence, data should be kept in the best matching database. These papers describe how to send relational data to NoSQL when it is beneficial. Data from general surveys suggest that mixed server designs may achieve high growth and efficient performance with proper design.



Key Takeaways

- Hybrid Integration can accelerate relational queries: e.g. Ragunathan et al. found a hybrid SQL/NoSQL cloud model gave ~30% faster queries and higher scalability than SQL alone.
- Query Rewriting into NoSQL drastically speeds up bulk operations: e.g. Bjeladinović's SRC reduced large INSERT times by a factor of ~3–9.
- Complementary Strategies: Caching (Redis/Mongo), polyglot data placement, and hybrid architectures each address different bottlenecks.
- Drawbacks: Added system complexity and consistency management are key challenges. Small workloads may see less benefit.

Summary

The presented poster talks in detail about how using NoSQL approaches with legacy relational databases helps improve both speed and scalability when running common queries. It examines both hybrid cloud architectures and statement rewriting components, showing their positive aspects, drawbacks and important points. Other related works demonstrate that caching, polyglot persistence and combinations of these approaches are very beneficial. The research suggests that integrating different database traits, even with complexity, means databases can function more smoothly.

Conclusions & Future Work

Key Takeaways

- Both approaches demonstrate significant performance improvements over traditional RDBMS-only solutions
- Hybrid architectures can effectively leverage the strengths of both SQL and NoSQL systems
- The optimal approach depends on specific use cases and existing infrastructure
- Real-world implementation requires careful consideration of data consistency and integration complexity

Future Research Directions

- Develop automated tools for determining optimal data partitioning strategies
- Explore machine learning techniques to predict query performance across hybrid systems
- Investigate consistency guarantees in hybrid architectures under high concurrency
- Create standardized benchmarks for hybrid SQL/NoSQL systems
- Integrate these methods and validate on diverse real workloads

References

- Ragunathan, A., Bhavani, K., Sasi, A., & Kumar, S. (2025). Integration of NoSQL and Relational Databases for Efficient Data Management in Hybrid Cloud Architectures. *Journal of Machine and Computing*, 5(2), pp.1277–1287.
- Bjeladinović, S. (2025). Extending hybrid SQL/NoSQL database by introducing Statement Rewriting Component. *Computer Science and Information Systems*, 22(1), pp.1–32.
- Privalov, M.V. & Stupina, M.V. (2024). Improving web-oriented information systems efficiency using Redis caching mechanisms. *Indonesian Journal of Electrical Engineering and Computer Science*, 33(3), pp.1667–1675.
- Lajam, O. & Mohammed, S. (2022). Revisiting Polyglot Persistence: From Principles to Practice. *Int. J. Advanced Computer Science and Applications*, 13(5), pp.870–876.