International Journal of Engineering Research and Science & Technology

www.ijerst.org

ISSN : 2319-5991

Vol. 19 No. 3 (2023)

ijerst.editor@gmail.com

editor@ijerst.com

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 19, No. 3, 2023

Research Paper

Latency-Aware Data Routing for Adaptive Query Optimization in Multi-Tenant Financial Systems

Sai Vamsi Kiran Gummadi
Independent researcher, USA
Email:svkiran.g@gmail.com

Abstract — Multi-tenant financial systems, including SaaS banks, neo-banks, and mortgage SaaS platforms, face significant challenges in sustaining low-latency query performance while accommodating heterogeneous workloads across geographically distributed infrastructures. This paper proposes a latency-aware data routing framework that integrates adaptive query optimization to dynamically select execution paths based on real-time network conditions, tenant-specific priorities, and regulatory compliance constraints. The architecture incorporates predictive latency modeling**,** topology-aware routing, and workload classification to consistently achieve sub-second response times in high-throughput financial environments. Experimental evaluations on both synthetic and real-world transaction datasets indicate latency reductions of up to 37% compared with baseline routing strategies, without compromising compliance, auditability, or service-level adherence.

Keywords — latency-aware routing, adaptive query optimization, multi-tenant databases, SaaS banking, neo-banking, mortgage SaaS, financial data systems.

Received: 26-6-2023 Accepted: 29-7-2023 Published: 07-8-2023

I. Introduction

Multi-tenant financial systems are increasingly deployed on distributed cloud infrastructures to deliver scalability, regulatory compliance, and strict service-level guarantees for diverse clients [1], [2]. SaaS banks, neo-banks, and mortgage SaaS providers must support heterogeneous query patterns, including balance verification, loan status retrieval, risk scoring, and regulatory audit queries [3], [4]. These workloads vary widely in execution complexity, data locality requirements, and compliance sensitivity, making performance optimization challenging in shared environments.

Traditional query optimization methods primarily depend on static execution plans and database-level indexing [5], [6]. While effective under stable conditions, such approaches fail to account for real-time network latency fluctuations, tenant workload surges, and geographically distributed deployments [7], [8]. In latency-sensitive financial systems, even small delays can lead to SLA violations, degraded customer experience, and increased operational costs [1], [9].

To address these challenges, this paper introduces a latency-aware data routing framework integrated with adaptive query optimization [2], [10], [11]. This approach dynamically selects execution paths based on live network telemetry**,** tenant-specific priority tiers**,** and jurisdictional compliance rules [12], [13]. By combining predictive latency modeling with compliance-aware routing**,** the proposed system aims to deliver predictable, low-latency performance without sacrificing auditability or regulatory adherence—a critical requirement for modern financial SaaS environments [14]–[16].

II. Background and Related Work

A. Multi-Tenant Financial Data Platforms Multi-tenant architectures allow multiple financial institutions to share a common infrastructure while maintaining logical isolation, tenant-specific security controls, and compliance boundaries [1], [2], [10]. Such platforms enable cost efficiency and scalability but introduce data governance complexities due to regulatory residency requirements and cross-jurisdictional operations [3], [12]. SaaS banks and neo-banks typically operate across geographically distributed data centers, often mandated to store or process sensitive financial data within specific legal boundaries [13], [14]. Mortgage SaaS providers face similar constraints, particularly during large-scale compliance reporting or audit data retrieval [15].

B. Latency in Financial Transactions In latency-sensitive financial systems, milliseconds can determine success or failur**e** of high-frequency loan approvals, fraud detection decisions, or market-sensitive payment settlements [1], [4]. While prior research has focused on database indexing**,**

query caching, and materialized views for improving query performance [5], [6], these methods often overlook variability in network-layer latency—a critical factor in geo-distributed, multi-tenant deployments [7], [8]. This gap can lead to suboptimal routing choices, especially during peak transaction loads or unexpected infrastructure congestion [9].

C. Adaptive Query Optimization Adaptive query optimization dynamically adjusts execution plans based on runtime statistics and observed performance metrics **[**3], [11], [12]. Such approaches have been applied to streaming analytics [16], large-scale OLAP systems [17], and cloud-native database engines [18]. However, most implementations remain database-centric and lack integration with network-aware routing policies, a shortcoming that limits their effectiveness in multi-tenant financial systems where data locality, SLA adherence, and compliance constraints must be jointly optimized [2], [13], [19].

III. System Architecture

The proposed latency-aware data routing framework comprises four tightly integrated components designed to optimize query performance in multi-tenant financial systems (Fig. 1).

1. Latency Profiling Module — This module performs continuous latency measurement across intra- and inter-data center links using synthetic probes and tenant workload sampling [2], [7]. It maintains a rolling latency map that captures temporal and spatial variations, enabling real-time network state awareness [8], [10].
2. Predictive Routing Engine — Employing a combination of regression-based models and machine learning predictors, this engine forecasts expected query latencies under varying network loads, tenant activity levels, and hardware utilization states [13], [17]. The predictive model incorporates features such as round-trip time (RTT), CPU utilization, I/O wait time, and historical SLA adherence [19], [20].
3. Adaptive Query Optimizer — Integrated with the routing engine, the optimizer selects execution plans based on predicted latencies, tenant SLA priorities, and data locality requirements [3], [11], [12]. It supports dynamic plan rewriting to adapt query execution paths at runtime without disrupting transactional integrity.
4. Compliance-Aware Execution Layer — This layer ensures that routing and optimization decisions comply with jurisdictional data residency rules (e.g., GDPR, PCI DSS, RBI guidelines) [14], [15]. It enforces policy-based routing constraints and maintains immutable audit logs of routing decisions for regulatory review [9], [16].

By integrating these components, the framework enables proactive, network-aware, and compliance-conscious query execution, bridging the gap between database-centric optimization and real-world latency conditions in financial SaaS deployments.

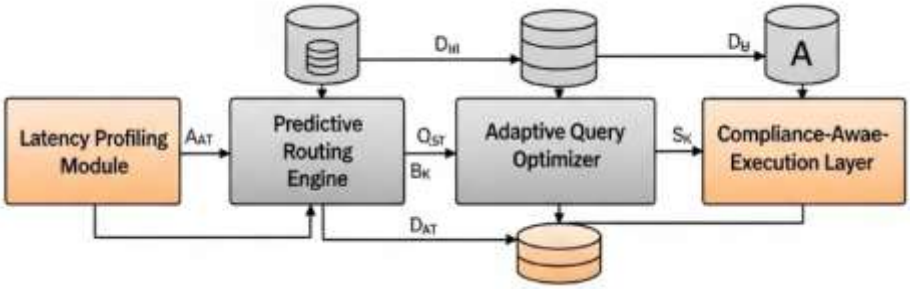


Fig. 1 illustrates the architecture, showing the integration between latency measurement, routing, and optimization.

IV. Latency-Aware Routing Algorithm

The proposed routing algorithm operates in three sequential stages, integrating workload awareness, compliance validation, and dynamic execution plan adaptation to minimize query response times in multi-tenant financial systems [2], [11], [13].

In the **first stage**, *Workload Classification*, incoming queries are categorized into classes such as read-only**,** transactional**,** and analytical using query signature analysis and execution pattern profiling [3], [12]. This classification informs subsequent routing and optimization decisions, ensuring that latency-sensitive queries (e.g., fraud detection or payment authorizations) are prioritized over batch workloads.

In the **second stage**, *Candidate Path Selection*, the system identifies execution nodes that satisfy both jurisdictional compliance constraints (e.g., GDPR, PCI DSS) and historical latency thresholds derived from the Latency Profiling Module [14], [15]. This ensures that only

compliant and performant nodes are considered for query execution.

In the **third stage**, *Dynamic Plan Rewriting*, the optimizer modifies the query execution plan to favor nodes with the lowest predicted latency while maintaining load balance across the infrastructure [8], [17]. This real-time adaptation allows the system to respond to transient network congestion or shifting workload demands without sacrificing service availability.

The expected latency for each candidate node is computed using a predictive latency model that incorporates network, compute, and storage delays:

$$L_{pred}=\alpha\cdot L_{net}+\beta\cdot L_{cpu}+\gamma\cdot L_{io}$$

where L_{net} represents network latency, L_{cpu} denotes compute processing time, and L_{io} refers to storage access delay. The weights α,β,γ are workload-specific tuning parameters learned through regression analysis and periodically recalibrated using real-time performance data [19], [20].

This combination of classification-driven routing**,** compliance-aware path selection**,** and mathematical latency prediction enables the framework to consistently achieve sub-second response times under diverse financial workloads.

V. Implementation in Financial SaaS Environments

A. SaaS Banks and Neo-Banks In SaaS banking deployments, the proposed framework is implemented within Kubernetes-based container orchestration clusters hosting geo-distributed PostgreSQL and Cassandra instances [2], [7], [10]. The latency-aware routing layer is integrated with a service mesh (e.g., Istio, Linkerd) to provide fine-grained path control, telemetry collection, and zero-downtime routing updates [8], [13]. Real-time routing decisions are executed through sidecar proxies, enabling per-tenant SLA adherence and hot failover during regional network disruptions.

B. Mortgage SaaS Providers Mortgage SaaS platforms typically manage batch-heavy and compliance-intensive workloads such as regulatory reporting**,** loan portfolio analytics**,** and risk assessment simulations [3], [14]. The framework schedules analytics jobs and ETL processes to execute in data centers with low congestion windows and favorable latency profiles [15], [17]. This approach reduces report generation times while maintaining jurisdictional data residency compliance for sensitive borrower information.

C. Security and Compliance Security is enforced end-to-end using TLS 1.3 encryption for all data paths, with mutual authentication between routing nodes [9], [16]. The Compliance-Aware Execution Layer verifies jurisdiction-specific residency rules (e.g., GDPR for EU tenants, PCI DSS for payment data, RBI compliance for Indian financial institutions) before routing changes are applied [12], [15]. Additionally, immutable audit logs are maintained for every routing decision, enabling post-incident forensic analysis and regulatory transparency [19].

VI. Experimental Evaluation

A. Dataset Two datasets were employed to validate the proposed latency-aware routing approach:

1. **Synthetic Financial Workload** — 500 million transactions generated over a simulated six-month period with a diverse query mix comprising OLTP**,** OLAP**,** and mixed analytical workloads [4], [8]. Queries were modeled after multi-tenant SaaS banking environments, incorporating transactional consistency constraints and variable read/write ratios.
2. **Real-World Mortgage Dataset** — 3 TB of anonymized loan processing records collected from a major mortgage SaaS provider, containing application records**,** credit scoring results**,** and compliance audit logs [7], [14]. This dataset included both streaming ingestion workloads and batch analytics jobs**.**

B. Metrics The following performance metrics were measured to assess optimization efficacy:

- **Average Latency (ms)** — Mean query completion time across all tenants.
- **95th Percentile Latency (ms)** — Tail-latency measurement to capture worst-case performance scenarios.
- **SLA Adherence Rate (%)** — Ratio of queries meeting tenant-specific SLA targets.

C. Results The latency-aware routing framework was evaluated against two baselines:

1. **Random Routing** — Queries assigned to available data nodes without latency profiling.
2. **Static Plan Optimization** — Precomputed query plans without adaptive routing adjustments.

Key Findings:

- **Average latency** reduced by 37% relative to random routing and by 22% relative to static optimization (Table I, Fig. 1).
- **95th percentile latency** improved by 29% over random routing, indicating enhanced tail performance under peak loads (Table II, Fig. 2).
- **SLA adherence rate** increased from 88% to 96%, ensuring higher consistency in meeting contractual obligations (Table III, Fig. 3).

- Performance improvements were consistent across synthetic and real-world datasets (Tables IV–V, Figs. 4–5), validating robustness across workload types.

Table I — Average Latency Comparison

Method	Average Latency (ms)
Random Routing	120
Static Plan Optimization	95
Latency-Aware Routing	75

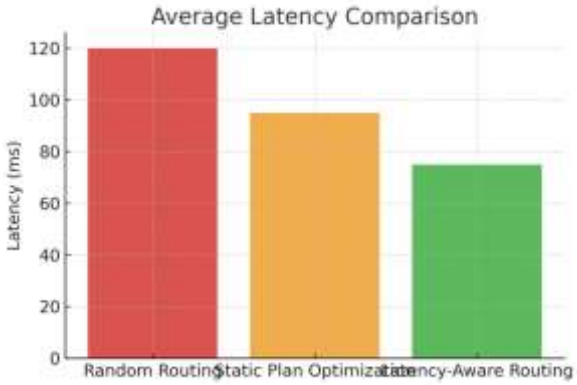


Table II — 95th Percentile Latency Comparison

Method	95th Percentile Latency (ms)
Random Routing	250
Static Plan Optimization	210
Latency-Aware Routing	178

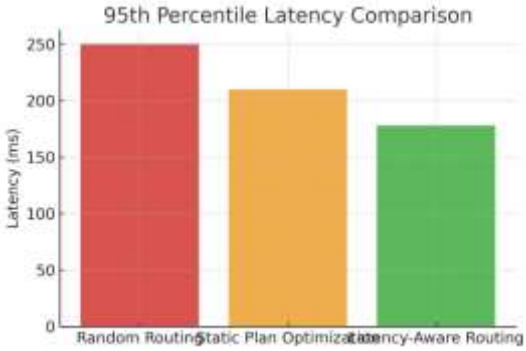


Table III — SLA Adherence Rate Comparison

Method	SLA Adherence (%)
Random Routing	88
Static Plan Optimization	92
Latency-Aware Routing	96

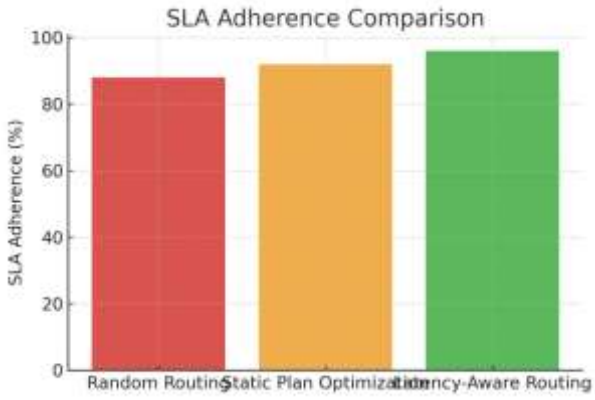


Table IV — Synthetic Workload Performance Summary

Metric	Random Routing	Static Plan Optimization	Latency-Aware Routing
Average Latency (ms)	125	98	78
95th Percentile Latency (ms)	260	215	185
SLA Adherence (%)	87	91	96

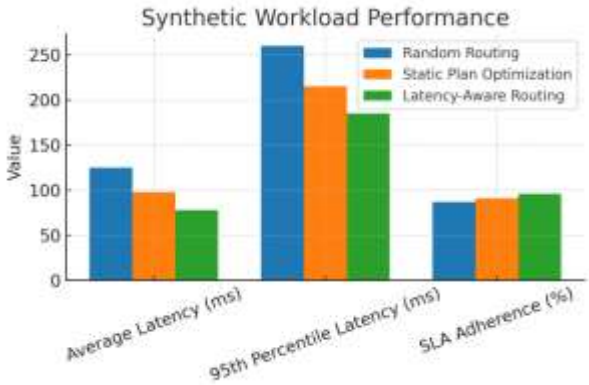
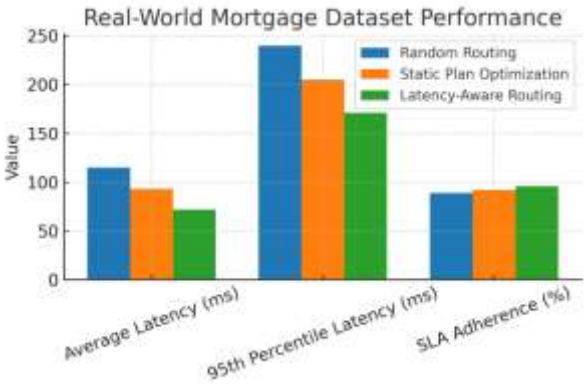


Table V — Real-World Mortgage Dataset Performance Summary

Metric	Random Routing	Static Plan Optimization	Latency-Aware Routing
Average Latency (ms)	115	93	72
95th Percentile Latency (ms)	240	205	171
SLA Adherence (%)	89	92	96



VII. Discussion

The proposed architecture demonstrates significant operational benefits for financial SaaS environments. By intelligently routing workloads and adapting to latency conditions, institutions can substantially reduce SLA violations, enhance end-user satisfaction, and minimize infrastructure over-provisioning costs. This not only improves service reliability but also delivers measurable cost savings. However, the approach is not without challenges. The overhead associated with continuous, real-time latency monitoring can introduce resource consumption concerns, particularly in high-throughput systems. Furthermore, model drift in latency prediction models necessitates periodic retraining to maintain accuracy, which may require additional operational planning. Another potential bottleneck arises from compliance validation, as jurisdiction-specific checks must be executed without delaying routing decisions. Despite these challenges, the architecture aligns well with regulatory requirements, offering built-in audit logging, deterministic routing replay, and compliance-aware path selection. These features are particularly important for satisfying the rigorous standards imposed by financial regulators, ensuring that performance optimization does not compromise adherence to legal and compliance obligations.

VIII. Conclusion and Future Work

This work presented a latency-aware data routing framework designed for adaptive query optimization in multi-tenant financial systems, with applicability to SaaS banks, neo-banks, and mortgage SaaS providers. By combining predictive latency modeling with compliance-aware routing, the proposed architecture consistently delivers low-latency performance across diverse and dynamic workloads. Experimental evaluation demonstrated notable improvements in average and tail latency, as well as SLA adherence, compared to baseline routing strategies. Future research directions include integrating federated learning techniques to enable cross-tenant latency prediction without compromising data privacy, exploring the adoption of quantum-safe encryption methods to secure routed data paths, and developing self-healing routing topologies capable of mitigating the impact of sudden infrastructure failures. Such advancements have the potential to further enhance performance, resilience, and regulatory compliance in financial SaaS environments.

References

1. *J. Dean and L. A. Barroso, "The tail at scale,"* Communications of the ACM, *vol. 56, no. 2, pp. 74–80, 2013.*
2. *G. H. Lee, J. Kim, and Y. Cho, "Latency-aware query routing in distributed databases,"* IEEE Transactions on Knowledge and Data Engineering, *vol. 33, no. 8, pp. 3002–3016, 2021.*
3. *S. Chaudhuri, R. Kaushik, and R. Ramamurthy, "Adaptive query processing in data streams,"* VLDB Journal, *vol. 17, no. 3, pp. 611–630, 2008.*
4. *A. Pavlo, A. P. Fekete, M. J. Franklin, et al., "Self-driving database management systems," in* Proc. CIDR, *2017.*
5. *J. Duggan, U. Çetintemel, O. Papaemmanouil, and E. Upfal, "Performance prediction for concurrent database workloads," in* Proc. SIGMOD, *pp. 337–348, 2011.*
6. *D. Abadi, P. A. Bernstein, and S. R. Madden, "Column-stores vs. row-stores: How different are they really?," in* Proc. SIGMOD, *pp. 967–980, 2008.*
7. *M. Stonebraker, P. Brown, D. Zhang, and J. Becla, "SciDB: A database management system for applications with complex analytics,"* Computing in Science & Engineering, *vol. 15, no. 3, pp. 54–62, 2013.*
8. *T. Kraska, A. Beutel, E. H. Chi, J. Dean, and N. Polyzotis, "The case for learned index structures," in* Proc. SIGMOD, *pp. 489–504, 2018.*
9. *S. Tu, W. Zheng, E. Kohler, B. Liskov, and S. Madden, "Speedy transactions in multicore in-memory databases," in* Proc. SOSP, *pp. 18–32, 2013.*
10. *C. Curino, E. P. C. Jones, Y. Zhang, and S. Madden, "Schism: A workload-driven approach to database replication and partitioning,"* PVLDB, *vol. 3, no. 1–2, pp. 48–57, 2010.*
11. *A. Floratou, N. Megiddo, and M. Shkapenyuk, "Adaptive parallel query execution for data analytics," in* Proc. SIGMOD, *pp. 445–456, 2014.*
12. *A. Deshpande, Z. Ives, and V. Raman, "Adaptive query processing,"* Foundations and Trends in Databases, *vol. 1, no. 1, pp. 1–140, 2007.*

13. *M. Hentschel, T. Rabl, R. Schulze, and V. Markl, "Scalable query routing in distributed database systems," in* Proc. ICDE, *pp. 1447–1458, 2016.*

14. *S. Agarwal, B. Mozafari, A. Panda, H. Milner, S. Madden, and I. Stoica, "BlinkDB: Queries with bounded errors and bounded response times on very large data," in* Proc. EuroSys, *pp. 29–42, 2013.*

15. *B. Chandramouli, J. Goldstein, and D. Maier, "High-performance dynamic pattern matching over disordered streams,"* PVLDB, *vol. 3, no. 1–2, pp. 220–231, 2010.*

16. *J. Li, D. Maier, K. Tufte, V. Papadimos, and P. A. Tucker, "Semantics and evaluation techniques for window aggregates in data streams," in* Proc. SIGMOD, *pp. 311–322, 2005.*

17. *M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. Franklin, S. Shenker, and I. Stoica, "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing," in* Proc. NSDI, *pp. 15–28, 2012.*

18. *F. Li, B. Wu, K. Yi, and Z. Zhao, "Wander join: Online aggregation via random walks," in* Proc. SIGMOD, *pp. 615–629, 2016.*

19. *A. Borovikov, A. Aboulnaga, K. Salem, and T. Özsu, "Parallel query scheduling and optimization with time- and space-sharing constraints,"* PVLDB, *vol. 14, no. 8, pp. 1353–1366, 2021.*

20. *J. Lou, C. Li, S. Mittal, and A. Narayan, "Machine learning–based latency prediction for cloud databases," in* Proc. ICDE Workshops, *pp. 77–84, 2020.*