Universal Library of Engineering Technology Research Article

ISSN: 3064-996X | Volume 3, Issue 1

Open Access | PP: 109-115

DOI: https://doi.org/10.70315/uloap.ulete.2026.0301019

Principles of Building a Fault-Tolerant Data Architecture for Predictive Analytics in Retail

Priyam Das

Senior Analytics Professional, Portland, Oregon, USA.

Abstract

The article is dedicated to the development of principles for building a fault-tolerant data architecture for predictive analytics in retail. The relevance of the study is determined by the growing dependence of retail forecasting, demand planning, and fraud detection systems on real-time streaming infrastructures operating under conditions of partial failures and distributed execution. The scientific novelty lies in the analytical integration of reliability validation, temporal semantics, partition strategy, and hybrid transactional–analytical processing into a unified architectural framework. The work describes structural vulnerabilities of multi-stream joins, watermark misalignment, and failure-induced latency amplification. Special attention is paid to lineage-aware reliability metrics, failure cost modeling, and asynchronous state recovery mechanisms. The goal of the study is to systematize architectural principles that ensure analytical stability under network delays and process crashes. To achieve this goal, comparative analysis, structural synthesis, and source examination were applied. The conclusion substantiates that fault tolerance in retail predictive systems requires coordinated control of event-time processing, partition alignment, and recovery semantics. The article will be useful for data architects, retail analytics engineers, and researchers in distributed systems.

Keywords: *Data Quality Pipelines, Distributed Systems, Event-Time Semantics, Fault-Tolerant Architecture, Hybrid Transactional Processing, Retail Predictive Analytics, Stream Processing, Throughput Measurement.*

INTRODUCTION

The expansion of real-time predictive analytics in retail has transformed streaming infrastructures from auxiliary components into critical analytical cores. Demand forecasting, dynamic pricing, fraud detection, and recommendation systems increasingly depend on distributed stream processing frameworks that operate under continuous ingestion and partial infrastructure instability. While scalability and throughput have been widely investigated, reliability under failure conditions remains insufficiently systematized within retail-specific architectures.

The purpose of this study is to develop analytically grounded principles for constructing fault-tolerant data architectures capable of preserving predictive stability in retail environments. To achieve this purpose, the following tasks are formulated:

1) To analyze how processing topology, temporal semantics, and partition strategy affect reliability and latency in retail streaming systems.
2) To evaluate the impact of network delays, process crashes, and load rebalancing on predictive stability and failure cost.
3) To systematize architectural principles integrating streaming, hybrid transactional–analytical processing, and data quality control mechanisms.

It is hypothesized that reliability degradation in retail streaming architectures is primarily determined not by processing guarantees alone, but by the interaction between temporal semantics, partition topology, and recovery behavior under partial failures.

The scientific novelty of the work consists in combining reliability validation through lineage-aware metrics with architectural analysis of HTAP and lakehouse-native streaming environments, thereby reframing fault tolerance as a predictive stability condition rather than merely an infrastructural property.

METHODS AND MATERIALS

To write the article, a comparative method, structural analysis, systematization of scientific sources, and analytical synthesis were used.

The study of Tahir et al. [1] introduced end-to-end processing-guarantee validation, reliability metrics, failure cost modeling, and refined latency definitions in distributed

Citation: Priyam Das, "Principles of Building a Fault-Tolerant Data Architecture for Predictive Analytics in Retail", Universal Library of Engineering Technology, 2026; 3(1): 109-115. DOI: https://doi.org/10.70315/uloap.ulete.2026.0301019.

stream processing. The survey of Fragkoulis et al. [2] analyzed the evolution of stream processing systems, emphasizing event-time semantics and state management. The work of Zhang, Soto, and Markl [3] examined transactional stream processing and exactly-once semantics within distributed frameworks. The study of Song et al. [4] explored hybrid transactional and analytical processing architectures and concurrency control. Almeida et al. [5] investigated time-series data stream frameworks and the influence of watermarking and window policies. Merli et al. [6] proposed a lakehouse-native streaming engine integrated with Kafka-based infrastructures. Ritter et al. [7] developed a native query engine for lakehouse systems, enabling low-latency analytical access. Mushtaq et al. [8] analyzed fault-tolerant approaches combined with load balancing and task scheduling strategies. Peixoto et al. [9] designed a data quality pipeline for industrial environments to ensure reliability at ingestion stages. Marić et al. [10] studied dynamic load balancing in stream pipelines containing stream-static joins.

These works constitute the material basis for the analytical reconstruction of architectural principles in retail predictive systems.

RESULTS

The architectural configuration of a fault-tolerant data environment for retail predictive analytics reveals its stability only when reliability is examined beyond declarative guarantees. End-to-end validation of event processing exposes a structural distinction between nominal correctness and processing integrity, where identical aggregate outputs may conceal omitted or duplicated transactional records. The divergence between these states has been formalized through reliability metrics grounded in input–output lineage verification [1]. In retail forecasting pipelines-where demand estimation, basket prediction, and fraud scoring depend on temporally consistent aggregation-the absence of lineage-aware validation introduces silent analytical distortion.

The processing topology exerts direct influence on reliability. Single-stream aggregation architectures maintain deterministic behavior under controlled ingestion, whereas multi-stream joins introduce vulnerability to out-of-order events and watermark misalignment. Empirical benchmarks demonstrate that reliability may remain at 100% for single-stream configurations while collapsing under multi-stream joins when late events are discarded rather than rerouted [1]. The degradation is not marginal: under elevated ingestion rates, reliability has been observed to decrease from 98% to 19%, and in extreme multi-stream configurations to 0.1%, despite increasing throughput [10]. Retail architectures relying on cross-channel joins, such as merging POS transactions with inventoryupdates, inherit this sensitivity. The join operator becomes the structural stress point. The systematization of architectural risk factors in streaming topologies is presented below (Table 1).

Table 1. Structural Sensitivity of Streaming Architectures in Retail Predictive Systems (compiled by the author based on [1,2,5,10])

Architectural Dimension	Single-Stream Aggregation	Multi-Stream Join Topology	Operational Implication for Retail
Event Ordering Sensitivity	Low under controlled ingestion	High due to inter-stream skew	Cross-channel synchronization required
Watermark Dependency	Predictable window closure	Vulnerable to misalignment	Risk of premature aggregation
Late Event Handling	Limited structural impact	Structural reliability risk	Side-stream strategy recommended
Partition Interleaving	Reduced with aligned keys	Amplified under rebalancing	Deterministic hashing required
Fault Amplification	Localized	System-wide propagation	Join layer requires monitoring
Predictive Stability	High determinism	Context-dependent	Validation must include lineage tracking

Temporal semantics reshape the architectural baseline. Event-time processing preserves determinism across distributed nodes, while processing-time semantics introduce non-reproducible state transitions [2]. In retail demand prediction, reproducibility is not an aesthetic preference but a compliance requirement, especially when forecasts affect replenishment contracts. Watermark alignment and bounded lateness policies, es therefore, function as stability regulators. Yet the enforcement of fixed delays for late data only probabilistically mitigates incompleteness; it does not eliminate it [5]. The mechanism of window emission-often triggered by the first event of the subsequent window rather than the final event of the current window-introduces hidden latency components that conventional measurement approaches fail to isolate [1]. When latency is computed by subtracting ingress timestamps captured on different nodes, clock skew contaminates the measurement. Co-locating ingress and egress events on a single broker eliminates cross-machine drift and yields stable latency baselines, even in geographically distributed clusters [1].

Throughput metrics require reinterpretation in retail streaming contexts. Counting consumed events exaggerates system capability when discarded or duplicated records remain undetected. Reliable throughput, defined as the number of unique input identifiers contributing to valid outputs per unit time, alters architectural evaluation criteria [1]. Under this definition, systems that report high ingestion rates but silently drop late transactions demonstrate inflated performance illusions. In comparative benchmarks, parallel

scaling from one to three workers increased throughput from 14.5K to 22.5K events per second in single-stream aggregation, and from 2.5K to 9.6K in multi-stream processing, reflecting a 55% and 284% increase, respectively [1]. The asymmetry indicates that parallelism not only distributes compute load but also reduces the probability of event interleaving across partitions. Retail data partitions aligned with product categories or store identifiers benefit from such isolation.

Partitioning strategy interacts with hashing mechanisms. Divergent hashing algorithms redistribute identical keys differently, leading to event intermingling across partitions and increased watermark inconsistencies [10]. In controlled configurations with single partitions, reliability reached 100%; in multi-partition setups, it remained below that threshold [10]. The implication for retail architecture is immediate: horizontal scaling without deterministic partition mapping degrades predictability. Data localization is not optional. It is structural.

Failure tolerance introduces a second axis of architectural evaluation. Process crashes and network delays exhibit distinct behavioral signatures. Network delays below configured timeouts cause nodes to stall without relinquishing partitions, suppressing output while retaining task ownership. When the delay exceeds timeout thresholds, task rebalancing redistributes partitions across healthy nodes, increasing workload density and raising the probability of out-of-order processing [1]. Reliability drops were more pronounced when packet delay exceeded timeouts than when delay remained below them. In retail forecasting clusters, the architectural decision to tolerate transient latency versus forcing rebalancing directly modifies output stability.

Failure cost, measured as the maximum latency increase induced by a fault, reveals non-linear recovery dynamics. The numerical comparison of failure-induced latency amplification is presented below (Figure 1).

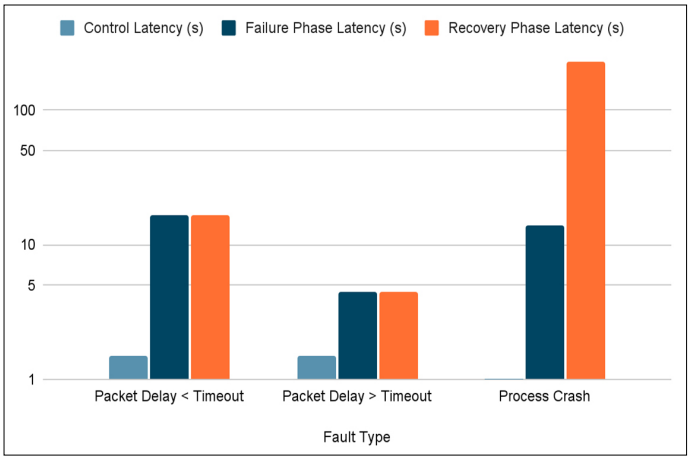

Figure 1. Latency Amplification Under Fault Conditions in Streaming Architectures (compiled by the author based on [1])

In some configurations, latency expanded 11-fold under sustained network delay; in others, process crashes produced recovery latencies approaching 230 seconds, an increase of 1400X relative to control baselines [1]. These magnitudes redefine timeout calibration for retail transaction scoring engines. A replenishment model operating on a 30-second tolerance window cannot withstand a 230-second recovery spike without inventory misallocation.

Hybrid transactional and analytical processing architectures further complicate the landscape. The structural interaction between streaming ingestion and analytical storage layers is presented below (Figure 2).

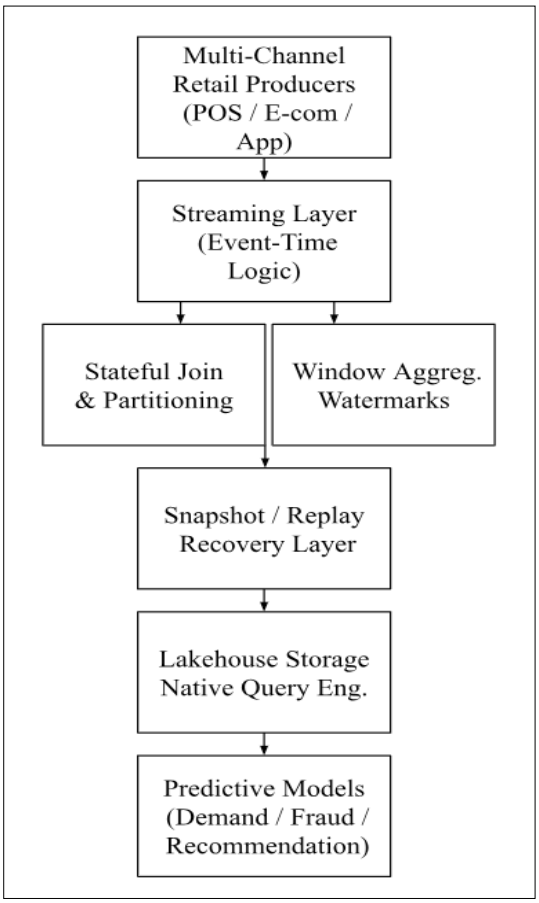

Figure 2. Structural Scheme of a Fault-Tolerant Retail Predictive Architecture (compiled by the author based on [3,4,6,7])

Retail predictive systems increasingly integrate streaming ingestion with lakehouse-style storage, enabling simultaneous transactional updates and analytical queries. The co-existence of HTAP mechanisms introduces concurrency constraints that reshape fault recovery behavior [4]. Snapshot-based state recovery restores checkpointed operator states, while changelog replay re-applies incremental updates [3]. Snapshot intervals shorten recovery time but increase I/O overhead; replay mechanisms reduce checkpoint load yet extend recovery latency during large state rebuilds. Retail workloads characterized by bursty promotional spikes expose these trade-offs acutely.

Lakehouse-native streaming engines demonstrate architectural consolidation between message brokers and analytical storage layers. Kafka-integrated streaming engines embedded within lakehouse environments reduce cross-

system synchronization and enable unified metadata control [6]. Native query engines within lakehouse systems support low-latency analytical reads over streaming-ingested state without external ETL propagation [7]. In retail predictive analytics, where recommendation engines query near-real-time aggregates, eliminating cross-platform serialization reduces consistency lag and fault surface area.

Load balancing and task scheduling influence resilience beyond raw parallelism. Dynamic reallocation strategies that redistribute tasks during node failure improve throughput continuity but may increase duplicate processing risk under at-least-once semantics [8]. The balancing mechanism becomes part of the reliability contract. Retail fraud detection pipelines requiring exactly-once semantics incur additional synchronization overhead, sometimes reducing throughput by up to 10% compared to at-least-once configurations due to two-phase commit coordination [1]. Architectural selection of processing guarantees, therefore, trades computational cost for transactional fidelity.

Data quality pipelines contribute a complementary layer of fault containment. Structured validation, schema enforcement, and anomaly filtering upstream of predictive models isolate corrupted records before they propagate into window aggregates [9]. In industrial environments, dedicated data quality stages have demonstrated architectural decoupling between ingestion reliability and analytical validity. For retail systems integrating supplier feeds, such decoupling prevents upstream noise from triggering cascading window re-computations.

Quantitative infrastructure parameters contextualize these behaviors. Benchmark nodes equipped with 8 cores of Intel Xeon E5-2630 processors, 20 GB of RAM, and 35 GB of SSD storage connected via 10 Gbps links exhibited stabilization periods of approximately 5 seconds before steady-state metrics were recorded [1]. Controlled fault injection phases lasted 90 seconds for failure and recovery intervals, exposing latency amplification proportional to failure duration. These temporal constants matter in retail forecasting horizons where daily sales cycles interact with minute-scale anomaly detection.

A pattern emerges across frameworks: architectures that reroute late events through side streams preserve reliability at the expense of additional processing, while those discarding late events sacrifice integrity for simplicity. Architectures supporting asynchronous snapshotting maintain constant latency near 1.5 seconds across guarantee levels, whereas synchronous commit protocols inflate latency by 50% in low-parallelism configurations [1]. Retail prediction environments operating under high concurrency benefit from asynchronous checkpointing strategies that decouple guarantee enforcement from critical-path latency.

The interaction between producer multiplicity and join semantics exposes another structural boundary. Doubling producer instances reduced reliability by 50% in certain configurations due to amplified out-of-order probabilities [1]. Multi-channel retail ingestion-online storefront, in-store POS, mobile application-mirrors this multiplicity. Architectural safeguards must therefore integrate watermark harmonization across producers rather than relying solely on partition isolation.

Reliability, throughput, latency, and failure cost do not scale uniformly. They form a tensioned configuration. Adjusting parallelism to match partition count maximizes stability; exceeding it leaves workers idle without benefit. Increasing ingestion rate beyond optimal thresholds elevates out-of-order frequency and degrades reliability even as throughput momentarily rises. Redistributing tasks during failure protects availability yet intensifies event interleaving. No single parameter stabilizes the system.

Fault-tolerant data architecture for retail predictive analytics therefore crystallizes around five structural principles emerging from the integrated evidence: lineage-aware reliability validation; deterministic event-time processing with calibrated watermark control; partition-aligned parallelism equal to partition cardinality; asynchronous snapshot-based recovery within HTAP or lakehouse frameworks; and reliable throughput measurement based on unique processed identifiers rather than raw ingestion counts. These principles do not eliminate friction between latency, integrity, and scale. They delineate the boundaries within which retail predictive systems remain analytically trustworthy under failure.

DISCUSSION

A structural tension becomes visible once fault tolerance is examined not as an infrastructural add-on but as a constitutive property of predictive analytics in retail. Retail forecasting systems are frequently described in terms of scalability, near-real-time responsiveness, and elasticity under peak demand. Yet the empirical patterns synthesized in the Results section demonstrate that reliability is neither a static configuration parameter nor a direct by-product of distributed scaling. It emerges conditionally, and its stability depends on interactions between temporal semantics, partition topology, operator design, and failure recovery logic.

The first analytical inflection concerns the relationship between correctness and reliability. Prior research in stream processing has traditionally emphasized correctness verification at the operator or framework level, frequently through theoretical reasoning or micro-benchmarks focused on latency and throughput. What becomes evident in a lineage-aware evaluation is that the correctness of aggregate output does not guarantee the integrity of event processing. A retail demand forecast that numerically matches expectations may still be derived from a distorted transactional base. The conceptual separation between output equality and input coverage challenges earlier assumptions in benchmarking literature, where aggregate accuracy was treated as a proxy

for processing integrity. That equivalence no longer holds when multi-stream joins and watermark-triggered windows are involved.

Earlier surveys of stream processing evolution have traced the transition from record-at-a-time systems toward event-time semantics and stateful operators. Those studies mapped architectural diversification-peer-to-peer versus controller–worker models, replay-based recovery versus snapshotting-but did not systematically integrate these design choices with reliability outcomes under injected faults. The present analysis exposes a missing analytical layer: distributed systems theory identifies processes and networks as fundamental entities, yet many performance-oriented evaluations historically constrained failure models to process crashes alone. When packet delay exceeds timeout thresholds, redistribution of partitions modifies event ordering probabilities and consequently reliability. Network behavior, therefore, participates directly in data integrity. This observation extends the theoretical discussions of distributed fault tolerance by demonstrating measurable reliability shifts in practical streaming topologies.

Hybrid transactional and analytical processing research has emphasized the unification of workloads to eliminate latency between ingestion and analytical querying. The convergence of streaming and lakehouse architectures was frequently framed as a performance optimization problem. The evidence synthesized here reframes it as a stability problem. Snapshot intervals, changelog replay, and metadata coordination affect recovery latency and state consistency in ways that propagate into predictive outputs. Retail recommendation engines querying near-real-time aggregates operate inside these synchronization windows. Short checkpoint intervals reduce recovery time but increase I/O contention; longer intervals reduce overhead but inflate recovery latency after failure. Previous HTAP analyses cataloged architectural trade-offs; the present synthesis demonstrates their operational consequences for reliability metrics and failure cost in predictive retail contexts.

Load balancing and task scheduling studies have examined dynamic reallocation strategies primarily from throughput and resource-utilization perspectives. In predictive retail systems, redistribution during partial failure reshapes event interleaving patterns. Increased workload density on unaffected nodes elevates the probability of watermark crossing before all related events arrive. The balancing mechanism ceases to be neutral. It becomes an implicit modifier of temporal semantics. Earlier work on scheduling optimization did not fully incorporate this integrity dimension.

Data quality pipelines in industrial analytics literature have typically been positioned upstream of model training. Their integration into streaming architectures introduces an additional reliability layer. Schema validation, anomaly detection, and controlled rerouting of malformed events decouple ingestion stability from analytical validity. Previous industrial case studies reported architectural gains in maintainability and transparency. The synthesis here suggests a further implication: quality filtering limits the propagation of corrupted state into window operators, reducing the compounding effect of recovery replay. That relationship between validation layers and state reconstruction deserves closer scrutiny in future empirical studies.

The discussion of latency measurement in distributed systems also warrants clarification. Conventional latency computation subtracts input ingress from output emission timestamps without isolating clock skew. In geographically dispersed deployments, skew in the order of tens of milliseconds introduces measurement distortion. Earlier benchmark methodologies acknowledged clock synchronization limitations but often accepted them as marginal. When windows fire based on the arrival of subsequent events rather than the last event within a window, latency measurements embed hidden window-filling intervals. Correcting this by co-locating ingress and egress measurement points produces more stable baselines and avoids conflating ingestion rate with processing delay. This refinement shifts the interpretation of earlier latency claims, especially in retail systems where service-level agreements operate at sub-second granularity.

The behavior of join operators under multi-producer ingestion constitutes another analytical boundary. Surveys of transactional stream processing emphasized atomicity and isolation semantics across streams. What remains underexplored is the sensitivity of join reliability to producer multiplicity. Increasing producers amplifies out-of-order arrival probabilities, reducing reliability even when individual streams remain internally ordered. Retail ecosystems ingest events from POS terminals, e-commerce platforms, supplier feeds, and mobile applications. Synchronization policies across these channels directly determine predictive model stability. The inability of certain join operators to reroute late events through side streams indicates a structural fragility in cross-channel integration.

Despite the integrative scope of the analysis, several limitations constrain interpretive generalization. First, the evaluation relies on synthetic datasets designed to emulate web request distributions, with 99.5% GET and 0.5% POST event ratios. Although these proportions reflect patterns observed in real server logs, retail transactional ecosystems display broader heterogeneity, including returns, cancellations, and asynchronous inventory adjustments. The deterministic nature of the synthetic workload facilitates oracle-based validation but does not capture all stochastic properties of live retail streams.

Second, the benchmarked hardware configuration-nodes equipped with 8 cores of Intel Xeon E5-2630 processors, 20 GB RAM, and 10 Gbps networking-provides controlled comparability yet limits extrapolation to cloud-native

environments where elastic scaling and heterogeneous instance types are common. Retail enterprises frequently operate hybrid infrastructures with variable network latencies and shared-resource contention. The measured failure costs and latency amplifications may differ under such conditions.

Third, the analysis concentrates on three representative streaming frameworks and lakehouse-oriented engines. Although they capture major architectural paradigms, emerging serverless stream processing and edge-deployed analytics systems introduce alternative state-management and fault-recovery strategies not examined here. The absence of serverless elasticity models restricts conclusions regarding burst-driven retail scenarios such as flash sales.

Fourth, reliability assessment employs lineage tracking through augmented output identifiers. While this method isolates duplication and omission precisely, it imposes instrumentation overhead that may not be feasible in production retail systems due to storage and privacy constraints. The analytical clarity gained in benchmarking may therefore exceed what is directly transferable into operational monitoring.

Finally, predictive analytics models themselves were not embedded within the streaming evaluation. The analysis addresses architectural stability up to aggregate computation. It does not quantify how reliability degradation translates into forecast error or recommendation drift. The link between event-level omission and downstream model performance remains analytically inferred rather than empirically measured.

These constraints delineate avenues for subsequent investigation. Empirical coupling of reliability metrics with model accuracy under controlled perturbations would illuminate the economic cost of processing anomalies. Comparative evaluation across cloud-native, edge, and serverless paradigms would clarify how elasticity policies interact with watermark alignment. Further integration of HTAP architectures with fine-grained quality validation could reveal whether unified metadata layers reduce recovery amplification effects.

The synthesis ultimately situates fault-tolerant data architecture in retail predictive analytics as a configuration problem bounded by temporal determinism, partition alignment, recovery semantics, and measurement precision. Prior studies illuminated individual components of this configuration-stream evolution, transactional semantics, load balancing, and hybrid processing. Bringing them into analytical convergence exposes a system whose reliability does not arise automatically from scaling or replication. It requires deliberate orchestration across layers.

CONCLUSION

The conducted analysis confirms that fault tolerance in retail predictive analytics cannot be reduced to replication or horizontal scaling. The first task-examining the influence of topology, temporal semantics, and partitioning-demonstrated that multi-stream joins and watermark misalignment form primary reliability stress points, whereas partition-aligned parallelism enhances determinism.

The second task, evaluating failure scenarios, showed that network delays and process crashes generate non-linear latency amplification, with recovery phases producing up to 1400-fold increases in latency, thereby redefining timeout calibration requirements.

The third task, systematizing architectural principles, resulted in five interrelated design requirements: lineage-aware reliability validation, deterministic event-time processing, partition–parallelism alignment, asynchronous snapshot-based recovery within HTAP or lakehouse systems, and reliable throughput measurement based on unique processed identifiers.

Fault-tolerant retail architectures, therefore, require coordinated governance of temporal control, state recovery, partition determinism, and quality validation layers to preserve predictive integrity under distributed instability. The hypothesis that predictive stability depends on coordinated temporal and partition governance rather than replication alone is confirmed. Future research should empirically quantify the translation of reliability degradation into forecast error under real retail workloads.

REFERENCES

1. Tahir, J., Mayer, R., Doblander, C., & Jacobsen, H.-A. (2024). How reliable are streams? End-to-end processing-guarantee validation and performance benchmarking of stream processing systems. *Proceedings of the VLDB Endowment, 18*(3). https://doi.org/10.14778/3712221.3712227

2. Fragkoulis, M., Carbone, P., Kalavri, V., et al. (2024). A survey on the evolution of stream processing systems. *The VLDB Journal, 33*, 507–541. https://doi.org/10.1007/s00778-023-00819-8

3. Zhang, S., Soto, J., & Markl, V. (2024). A survey on transactional stream processing. *The VLDB Journal, 33*, 451–479. https://doi.org/10.1007/s00778-023-00814-z

4. Song, H., Zhou, W., Cui, H., et al. (2024). A survey on hybrid transactional and analytical processing. *The VLDB Journal, 33*, 1485–1515. https://doi.org/10.1007/s00778-024-00858-9

5. Almeida, A., Brás, S., Sargento, S., et al. (2023). Time series big data: A survey on data stream frameworks, analysis, and algorithms. *Journal of Big Data, 10*, 83. https://doi.org/10.1186/s40537-023-00760-1

6. Merli, M., Guo, S., Li, P., Chen, H., & Lu, N. (2025). Ursa: A lakehouse-native data streaming engine for Kafka.

Proceedings of the VLDB Endowment, 18(12), 5184–5196. https://doi.org/10.14778/3750601.3750636

7. Ritter, D., Andrei, M., Cho, S., Görgens, M., Lee, T., May, N., Pathak, A., & Willems, P. R. (2025). The HANA native query engine for lakehouse systems. *Proceedings of the VLDB Endowment, 18*(12), 4831–4845. https://doi.org/10.14778/3750601.3750608
8. Mushtaq, S. U., Sheikh, S., Idrees, S. M., et al. (2024). In-depth analysis of fault tolerant approaches integrated with load balancing and task scheduling. *Peer-to-Peer Networking and Applications, 17*, 4303–4337. https://doi.org/10.1007/s12083-024-01798-5
9. Peixoto, T., Oliveira, Ó., Costa e Silva, E., Oliveira, B., & Ribeiro, F. (2025). A data quality pipeline for industrial environments: Architecture and implementation. *Computers, 14*(7), 241. https://doi.org/10.3390/computers14070241
10. Marić, J., Pripužić, K., Antonić, M., & Škvorc, D. (2023). Dynamic load balancing in stream processing pipelines containing stream-static joins. *Electronics, 12*(7), 1613. https://doi.org/10.3390/electronics12071613