The American Journal of Interdisciplinary Innovations and Research
ISSN 2642-7478 | Open Access

Type Original Research
PAGE NO. 28-43
DOI 10.37547/tajiir/Volume07Issue07-04

A Five-Layer Framework for Cost Optimization in Snowflake: Applied to P&C Insurance Workloads

Shreekant Malviya
Tata Consultancy Services, Plano, Texas, USA

OPEN ACCESS

SUBMITED 28 May 2025
ACCEPTED 16 June 2025
PUBLISHED 07 July 2025
VOLUME Vol.07 Issue07 2025

CITATION

Shreekant Malviya. (2025). A Five-Layer Framework for Cost Optimization in Snowflake: Applied to P&C Insurance Workloads. The American Journal of Interdisciplinary Innovations and Research, 7(07), 28–43. https://doi.org/10.37547/tajiir/Volume07Issue07-04

Abstract:

The use of Snowflake as a cloud-native data warehouse has dramatically changed the management of analytics workload for Property and Casualty (P&C) insurers, while simultaneously presenting serious cost governance challenges. The heavy volume of searches, big data retention, and decentralized business intelligence operations are industry-standard procedures that tend to lead to uncontrolled credit usage and overspending on storage. This research introduces a modular five-layer optimization framework focused on property and casualty insurance data, combining workload segmentation, and compute sizing with Snowflake's account usage metadata. The framework is tested and validated using Kaggle's Insurance Agency Data, representing real-world P&C operations across 17 states. Benchmark queries simulating core insurance workloads were designed using modified TPC-H logic, a standard decision support benchmark that enables realistic performance evaluation under analytical query conditions, achieving up to 82% cost reduction and a 64% reduction in execution time without compromising the results. These results highlight the efficiency of the framework to facilitate proactive and elastic cost control. Future studies can investigate AI-driven query forecasting, scalable warehouse dynamics, and real-time anomaly detection to further advance cloud-native data ecosystem governance.

Keywords: Snowflake Cost Optimization, Property & Casualty Insurance Data Workloads, Metadata-Driven Cost Control, Query Performance Tuning

1. INTRODUCTION

The transition to cloud data warehousing has revolutionized big data analysis and management in the property and casualty insurance industry. Innovative cloud-native platforms like Snowflake provide elastic scaling and pay-as-you-go pricing, thereby enhancing the ease of analytical responsiveness [1]. This transition introduces a new operational risk of cost unpredictability to Property & Casualty (P&C) Insurance. Ad hoc queries, complete data scans, and the sheer amount of historical claims data can result in increased compute and storage consumption, especially when big data cost governance models are not aligned with workload behavior [2].

Despite Snowflake's wide array of features such as auto-suspend, materialized views, result caching, and query telemetry, these features are often applied ineffectively in insurance scenarios. Past research and industry practices reveal some structural and technical sources of insurance data platform cost inefficiency, some of these include over-provisioned warehouses, underutilized storage features, redundant business intelligence queries, and poor resource allocation [3] [4].

Further, FinOps adoption remains limited among most insurers [5], [6]. Traditional practices like monthly usage reviews or manual query monitoring are too slow and rigid to keep up with the fast-changing demands of insurance data. For example, during events like open enrollment or catastrophe modeling which requires real time responsiveness, these outdated governance methods can either drive up unnecessary costs or restrict essential workloads at critical moments.

This work presents a new, workload-aware, modular cost optimization method particularly tailored for Snowflake deployments within the property and casualty insurance business. The solution brings together query telemetry, warehouse isolation, dynamic scaling, and observability under an integrated method to manage cost more effectively while maintaining performance. This work used an insurance data set from Kaggle [5] and TPC-H-inspired query benchmarks [6] to simulate baseline and optimized workloads, quantifying computing efficiency and byte savings. The outcome provides a predictable approach to domain-aligned cost management in cloud-native insurance analytics environments. However, there is limited research on how Snowflake's cost control features can be systematically adapted to insurance-specific workloads. This study addresses that gap by exploring how a workload-aware strategy can improve cost efficiency and performance in Snowflake for P&C insurance use cases.

2. LITERATURE REVIEW

2.1 Overview of Snowflake Architecture and Pricing

Snowflake is a new-generation data warehousing cloud platform that has become popular due to its cloud-native architecture as well as simplicity. In contrast to legacy systems, Snowflake is designed for the cloud and keep storage and computation separate, resulting in higher flexibility and scalability **Figure 1.** It enables companies to manage resources autonomously, thus making it simpler to manage performance as well as costs [7].

For example, Snowflake architecture accommodates the ability to auto-suspend and resume so that compute resources can remain suspended when inactive and resumed as necessary. This prevents unnecessary expenses and enhances effectiveness compared to legacy data systems. It is also designed with a multi- cluster architecture, which allows customers not to encounter bottlenecks during peak periods, hence enhancing performance in teams [8], [9].

An important aspect is caching at varying levels—query results, metadata, and in-memory. Such caches help the system prevent re-computation of the same queries again and again, saving resources, and time. Such automation draws Snowflake to enterprises with lots of data but who desire to reduce infrastructure maintenance.

The Snowflake pricing is based on usage. Storage is priced per terabyte per month, and compute is priced per second based on the warehouse size. The companies have a number of warehouse sizes, ranging from X-Small to 6X-Large, to be able to accommodate their workload. The pricing is transparent with distinct prices for storage and compute that enable the teams to know what they are spending. Overall, the price offered by Snowflake is transparent, where companies can pick based on their needs [10], [11].

In general terms, Snowflake's architecture and pricing model provide flexibility, performance, and simplicity, and are extremely well-placed in sectors such as banking and insurance, where performance and price are paramount.

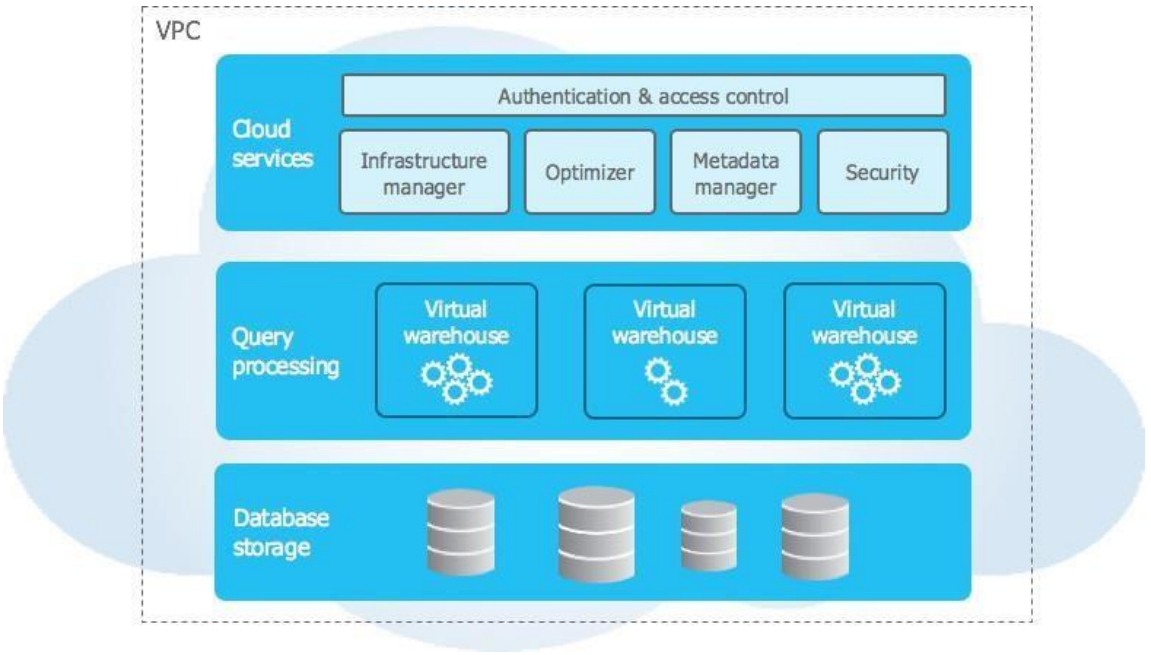

Figure 1. Snowflake Architecture

2.2 Cost Optimization Techniques in Cloud Data Warehouses

With more frequent utilization of cloud data platforms, cost management has become a fundamental requirement for most businesses. Snowflake, as much as other platforms such as Redshift and BigQuery, has mechanisms and processes in place to enable the mitigation of unwanted expenditures. Getting to know the proper use of the technologies is essential to achieving the maximum potential of the platform.

One of the initial measures is computation usage optimization. Snowflake's auto resume and suspend capabilities switch off compute resources while they are not in use. Several companies' de-isolate environments, such as different warehouses for production and development, to ensure that there's no overlap and resources are still allocated where they're needed [12]. This measure avoids the error of performing costly operations in unnecessary contexts and maintaining computer expenditure in check. Another category is storage. Snowflakes have columnar formats, such as Parquet, and have zero-copy cloning. These capabilities eliminate redundancy in storage as well as decrease costs in settings where teams need to test or create production-like data [13], [14].

Companies shift cold data into less expensive forms of storage, such as S3 Glacier, to lower the cost of long-term storage. This is particularly useful in industries such as insurance, where storage of data for several years is necessary to meet regulations. Query construction has a significant cost effect.

Badly constructed searches can traverse vast data sets and consume a lot of processing power. Snowflake, as well as other platforms, promotes the use of filters early on, the selection of columns required, and the utilization of materialized views or result caching. These actions minimize the level of data that is read and the time it takes for queries to execute [15], [16]. Monitoring wasteful or expensive queries regularly can result in significant savings overtime. This optimization on a technical level is also complemented by cost control. Resource monitors in Snowflake enable monitoring credit usage by a warehouse and can notify usage when usage rates exceed thresholds that are established in advance [12].

Teams utilize job schedulers, such as Airflow or dbt Cloud, to execute large jobs during off-peak periods when system loads are low. These measures introduce a non-technical level of control, which is very beneficial in

maintaining reasonable cloud expenses [17]. Lastly, developing cost-consciousness in teams is a long-term benefit. Increasingly, businesses are looking into FinOps practices that encourage shared responsibility between finance and engineering. Ascribing workloads or building dashboards that reflect usage by team or project raises visibility, of the effect of individual action on cloud expenditures [18]. Such a cultural shift makes a significant contribution to sustaining sustainable operations. Combined, these techniques offer a robust foundation for optimizing cloud expenditure while maintaining high performance. Uniformity and observability are critical in both tools and Teams.

2.3 Cost Challenges in Cloud-Based Data Warehouses for P&C Insurance

Although cloud solutions like Snowflake have numerous advantages, insurance businesses have specific challenges when it comes to keeping costs low. This is mostly because of the usage and storage of data in business. **Table 1** summarized the most significant cost drivers in the P&C workloads, Analysts and departments within insurance organizations, in most cases repeat similar queries repeatedly, in some cases uncoordinated. Redundant or inefficient queries can take up 30–40% of compute load [3]. This not only increases the cost but also makes the system slower, particularly when a large number of users are accessing data simultaneously.

The second problem pertains to storage. Insurance companies have to retain historical data for many years, sometimes decades. This encompasses policy documents, claims, and histories of transactions. Snowflakes incur a cost of approximately $23 per terabyte per month, which quickly adds up when retaining data for multiple years. Most businesses lack adequate procedures for archiving or purging data, so they end up with wasteful storage costs that may be sitting idle forever [19].

Table 1. Summary of Key Cost Drivers and Their Operational Impact in P&C Insurance Cloud Data Warehouses

Cost Issue	Quantitative Impact	Operational Effect
Redundant BI Queries	30–40% of compute spends	Increased costs, slower reporting, elevated risk
Large Historical Storage	$23/TB/month; $100k–$500k+ annually	Storage inflation, maintenance overhead
Data Redundancy	Up to 30% of IT budget	Complexity, inefficiency, and decision-making delays

Another challenge is that most firms don't segment or classify work by department. The calculations utilized by the underwriting employees might not be distinct from those which are used by the claim's employees. Without this demarcation, it is hard to recognize which department should manage higher costs [18]. The absence of visibility hinders proactive action and tends to result in a reaction too late, due to factors such as analyzing consumption by itself when the monthly invoice is too large. Real-time cost monitoring is not common. Most insurers continue to use quarterly check-ups or ad hoc audits. The insurance sector is highly reactive; unexpected events like natural disasters can trigger a massive peak in data activity. Without real-time notification and automation, these spikes can result in shocking overages. The intricacy of insurance data pipelines make it worse. Data tends to get pulled from various systems and undergoes numerous transformations. Without proper monitoring, minor inefficiencies can translate into major money issues. Despite increased awareness of FinOps, most insurance companies continue to act in silos, and it becomes challenging to have cost ownership that's collaborative.

Although there are recent studies that have examined methods to reduce cloud-based query expenses, the majority do not fulfill the specific needs of property and liability (P&C) insurance. One study proposed an approach to optimize cost and speed for cloud native queries; however, it is not customized to address the particular regulations and challenges within the insurance sector [20]. Another study proposed a

technique to minimize cloud query expenses, primarily concentrating on infrastructure configurations and neglecting platform-specific functionalities like Snowflake's caching and warehouse isolation [21]. Additional work examined the role of FinOps in cloud cost management [22]; however, a practical guide for its application in actual insurance data systems is lacking. This indicates the necessity for a comprehensive framework that aligns with the data and cost challenges in property and casualty insurance.

3. PROPOSED SOLUTION FRAMEWORK

This paper proposes a layered cost optimization framework divided into 5 layers tailored to the operational dynamics of Snowflake deployments within Property and Casualty (P&C) insurance environments. The aim is to provide a solution that is scalable and workload-aware, that aligns data platform efficiency with business imperatives such as claims processing, underwriting, fraud analytics, and regulatory compliance.

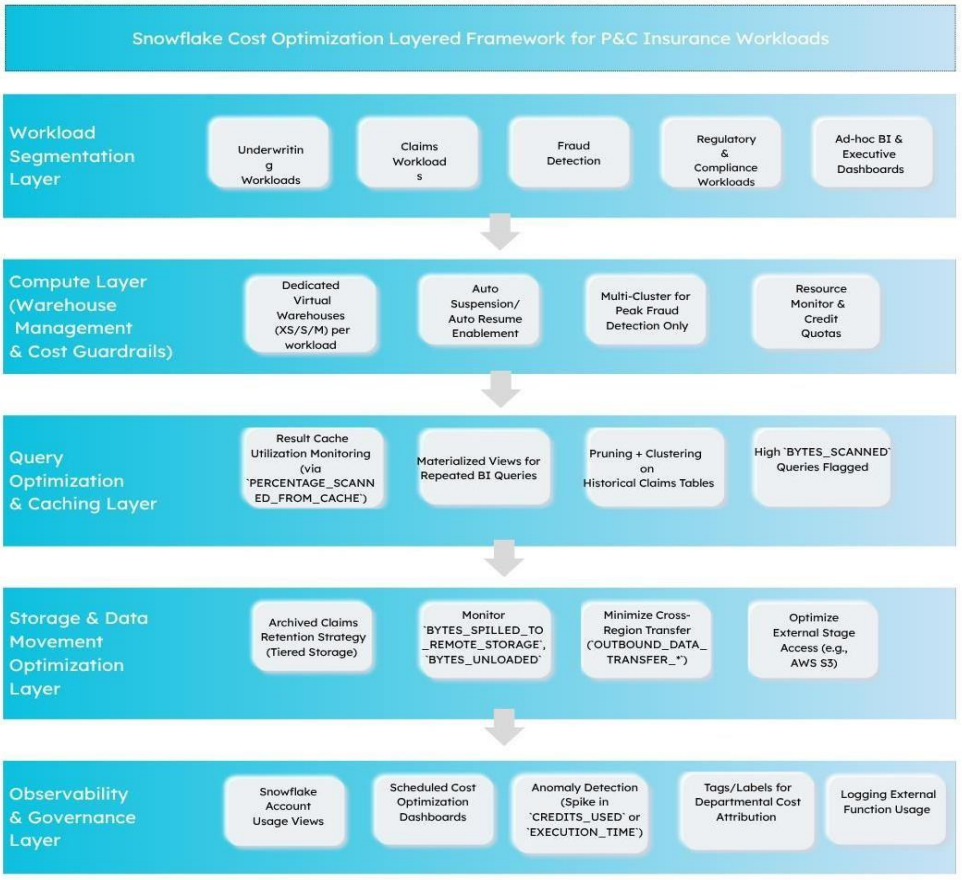

Figure 2. 5-layer high level architecture overview diagram

An architecture of a framework in **Figure 2** illustrates intended for Snowflake cost optimization for property and casualty insurance workloads. The framework begins with segmentation of P&C workloads, isolating important functionalities like claims, underwriting & pricing into dedicated compute layers. Subsequent layers ensure query efficiency, storage and movement, and system observability, each leveraging Snowflake metadata to drive automation and cost control. The layered design ensures scalability, governance, and alignment with insurance-specific analytics demands.

3.1 Five-layered proposed solutions

Layer 1: Workload Segmentation

This layer segments the workloads, and the loads are segmented by functional goals and time-based implementation patterns. Allocation of a virtual warehouse on a dedicated basis for a specific domain guarantees segregation and scalability. Segmentation is utilized with workload-specific controls such as auto-suspend/resume settings and resource monitors and subsequently lowers idle computing costs.

Layer 2: Compute Optimization

The compute optimization layer focuses on historical performance metrics from Snowflake metadata, like QUERY_LOAD_PERCENT, EXECUTION_TIME, and CREDITS_USED_COMPUTE, to help with decisions like expanding the warehouse and ensure that the computing resources are of the right size. After analyzing the metrics, multi-clustering is only turned on for loads that need a lot of parallel processing, like fraud detection, and single-threaded loads can be allocated to small compute.

Layer 3: Query Optimization and Caching

This layer solely emphasizes the performance tuning of queries using techniques that reduce BYTES_SCANNED and increase PERCENTAGE_SCANNED_FROM_CACHE. In this layer, recursive queries are tuned using materialized views and result caching. Query telemetry is monitored cautiously to detect performance bottlenecks, paying close attention to inefficient scanning and poor data trimming.

Layer 4: Storage and Data Movement Optimization

This layer focuses on optimization of storage and data migration. In this layer, long tail and historical data are relocated to low-cost storage tiers. Data egress and stage access fees are alleviated through the management of OUTBOUND_DATA_TRANSFER_BYTES and optimizing external file interactions. Memory-intensive operations like spill events and unloads are monitored and reorganized as needed.

Layer 5: Observability and Governance

An integrated observability layer consolidates telemetry for real-time performance and cost measurement. Departmental cost allocation is established with metadata tagging, and credit consumption anomalies are identified based on thresholds. External function calls and high-impact queries are constantly monitored to ensure ongoing optimization.

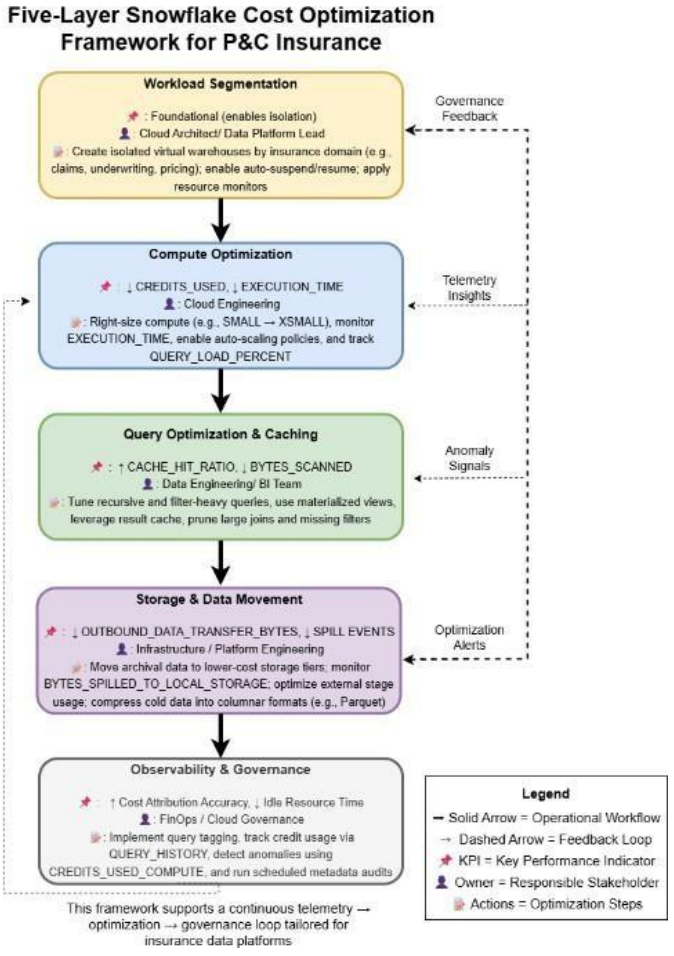

Figure 3. Detailed 5-layer Snowflake Cost Optimization Framework for P&C Insurance

The layered framework in **Figure 3**. visually maps out how compute, query, and storage efficiencies align with governance in Snowflake for insurance workloads. Feedback loops enable continuous cost control through telemetry-driven insights.

Snowflake offers access to the key metadata fields for monitoring and performance evaluation. **Table 2.** Maps key Snowflake metadata fields to their respective roles in the proposed five-layer optimization framework. Each field provides granular insight into query behavior, compute usage, storage patterns, or network activity. These metrics inform targeted actions such as warehouse right-sizing, query tuning, cache optimization, and data lifecycle management. By aligning field-level telemetry with each optimization layer, the framework enables evidence-based cost control. This structured mapping also supports automation and ongoing performance monitoring across P&C insurance workloads.

Table 2. Mapping of Snowflake Metadata Fields to Optimization Layers

Snowflake Field	**Used In Layer**	**Purpose in Optimization**
CREDITS_USED	Layer 2 , Layer 5	Overall credit consumption per query/workload, basis for cost attribution and monitoring.
CREDITS_USED_COMPUTE	Layer 2	Measures compute usage for right-sizing and scaling warehouses.
CREDITS_USED_CLOUD_SERVICES	Layer 5	Tracks cloud service charges; useful in optimizing metadata operations and automation.
EXECUTION_TIME	Layer 2 , Layer 5	Identifies slow queries; helps size warehouses appropriately.
QUERY_LOAD_PERCENT	Layer 2	Assesses resource utilization for concurrency tuning and load balancing.
QUEUED_PROVISIONING_TIME	Layer 2	Indicates warehouse provisioning delays; informs warehouse scaling decisions.
QUEUED_REPAIR_TIME	Layer 2	Flags infrastructure recovery delays may indicate system-level inefficiencies.
QUEUED_OVERLOAD_TIME	Layer 2	Captures overload bottlenecks; suggests need for multi-cluster or query optimization.
BYTES_SCANNED	Layer 3	Core metric for evaluating query scan efficiency; high values trigger pruning strategies.
PERCENTAGE_SCANNED_FROM_CACHE	Layer 3	Measures cache effectiveness; used to assess result reuse opportunities.
BYTES_WRITTEN	Layer 3	Identifies heavy write operations; relevant for assessing data movement cost.
BYTES_WRITTEN_TO_RESULT	Layer 3	Indicates result set size; can inform result set optimization or compression.
BYTES_READ_FROM_RESULT	Layer 3	Reflects cache re-use behavior; high values = optimized performance.

ROWS_PRODUCED	Layer 3	Helps evaluate result yield vs. scan size; inefficient queries can be restructured.
BYTES_SPILLED_TO_LOCAL_STORAGE	Layer 4	Indicates local disk spill; signals under-provisioned compute or inefficient query.
BYTES_SPILLED_TO_REMOTE_STORAGE	Layer 4	Highlights of remote spill events can slow queries and increase costs.
BYTES_SENT_OVER_THE_NETWORK	Layer 4	Captures inter-node communication cost; reduced by optimizing joins/distributions.
ROWS_INSERTED, ROWS_UPDATED, ROWS_DELETED	Layer 4	Flags frequent DML operations; supports lifecycle management and data tiering.
BYTES_UNLOADED	Layer 4	Used to monitor unload activity; excessive unloading may need redesign.
OUTBOUND_DATA_TRANSFER_BYTES	Layer 4	Tracks cross-region/cloud data transfer; high values prompt architecture review.
INBOUND_DATA_TRANSFER_BYTES	Layer 4	Similar to the above, informs ingress costs and data loading efficiency.
LIST_EXTERNAL_FILES_TIME	Layer 4	Measures time listing files in external stages; optimized by partitioned/exact paths.
IS_CLIENT_GENERATED_STATEMENT	Layer 5	Identifies BI tool-generated queries; used to analyze/report tool inefficiencies.
TRANSACTION_BLOCKED_TYPE	Layer 5	Reveals lock/contention issues; helps tune concurrency and transaction design.
EXTERNAL_FUNCTION_TOTAL_* (all fields)	Layer 5	Used to monitor usage, latency, and data volume of external UDFs; cost driver if misused.
TOTAL_ELAPSED_TIME	Layer 2 , Layer 3	Combined metric to flag long-running queries and analyze performance trends.
PARTITIONS_SCANNED, PARTITIONS_TOTAL	Layer 3	Indicates pruning effectiveness; informs clustering and filtering strategies.
BYTES_DELETED	Layer 4	Tracks data removal efficiency; supports data lifecycle and tiered storage planning.
RELEASE_VERSION	Layer 5	Used for compatibility and audit tracking; less directly linked to cost but relevant.

4. IMPLEMENTATION CONSIDERATIONS

The envisioned five-layer Snowflake cost optimization architecture has to be implemented by way of strategic planning, metadata instrumentation, automation, and P&C insurance workload-specific performance monitoring. Each of the layers of optimization relies on the level to which Snowflake capabilities are mapped to insurance industry-specific business activities, including underwriting, claims handling, and regulatory reporting.

Tools and Instrumentation. The first stage in

implementation begins with workload segmentation, enabled through analysis of metadata fields such as CREDITS_USED, IS_CLIENT_GENERATED_STATEMENT, and QUERY_LOAD_PERCENT, which are available in the Snowflake Account Usage schema, facilitating isolation and classification of workloads. Dedicated Virtual Warehouses (VWHs) are provisioned for core functions, each configured with auto-suspend and auto-resume policies to eliminate idle compute costs. Resource monitoring is essential for setting credit thresholds and alerting stakeholders to anomalous spend behavior.

Compute and Query Optimization. The second phase in implementation is right-sizing and scaling of virtual warehouses driven by metrics such as EXECUTION_TIME, QUEUED_PROVISIONING_TIME, and CREDITS_USED_COMPUTE. Visualization tools such as Looker or Tableau can provide surface-level compute trends and usage patterns. For query optimization, dbt (data build tool) can be utilized for complex transformation logics to test for high-cost queries and enforce linting standards. SQL checks should detect excessive BYTES_SCANNED, missing filters, or ineffective joins, while automated alerts can flag regression in cache utilization (e.g., drops in PERCENTAGE_SCANNED_FROM_CACHE).

Storage and Data Movement. The third stage handles the Cold data, such as historical claims or policy records, which can be archived to cost-efficient tiers. Monitoring fields such as OUTBOUND_DATA_TRANSFER_BYTES, BYTES_UNLOADED, and LIST_EXTERNAL_FILES_TIME is critical for reducing inter-region egress charges and optimizing external stage interactions. Compression formats like Parquet can further reduce I/O costs.

Automation and Governance. The final phase involves an observability layer that should incorporate tagging logic for departmental attribution, scheduled cost reviews, and automated job audits. Integration with orchestration tools like Airflow or dbt Cloud enables the scheduling of metadata scans and optimization tests. Additionally, external function activity (e.g., EXTERNAL_FUNCTION_TOTAL_INVOCATIONS) must be monitored for cost and performance viability.

To evaluate effectiveness, organizations should track KPIs shown in **Figure 4**.

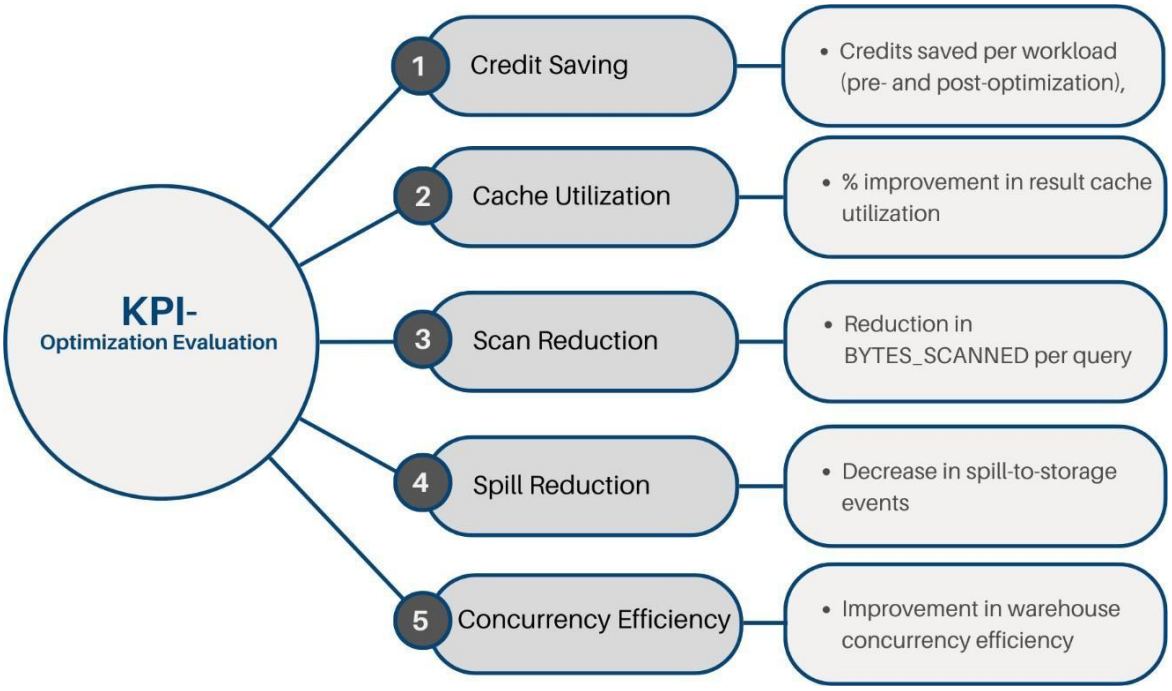

Figure 4. Key Performance Indicators (KPIs) for effective optimization evaluation

The KPIs were selected to directly represent Snowflake's cost determinants and were picked for their significance to essential P&C operations. They assess efficiency in computation, caching, storage, and concurrency, facilitating precise optimization. Each KPI is associated with a particular metadata field and corresponds to the priorities of real-world insurance analytic.

A dedicated FinOps or cloud analytics team should own this continuous improvement cycle, ensuring the framework evolves alongside the organization's data strategy. Establishing such a governance loop ensures both immediate cost savings and long-term platform sustainability.

5. EVALUATION

To assess the effectiveness of the proposed five-layer Snowflake cost optimization framework, we executed a series of benchmark queries **Table 3** modeled after TPC-H logic and tailored for Property and Casualty (P&C) insurance analytics. These queries were run on the *"Insurance Agency Dataset"* (Kaggle) [5], which contains over 213,000 records and 49 columns detailing agency-level insurance indicators across U.S. states from 2005 to 2015.

Table 3: Description of Benchmark Queries and Optimization Objectives in P&C Insurance Context

Query ID	Description	Key Fields Used	Layer(s) Optimized	Business Objective
Q1	Total Premium by State & Product Line	STATE_ABBR, PROD_LINE, WRTN_PREM_AMT	Layers 1, 2, 3	Market segmentation
Q2	Top 10 Agencies by Loss Ratio (2015)	AGENCY_ID, LOSS_RATIO, STAT_PROFILE_DATE_YEAR	Layers 2, 3, 4	Risk assessment
Q3	3-Year Average Loss Ratio by State	LOSS_RATIO_3YR, STATE_ABBR	Layers 1, 2, 3	Performance benchmarking
Q4	Policy Growth by Vendor	VENDOR, POLY_INFORCE_QTY	Layers 2, 3	Sales analysis
Q5	Retention Ratio by Agency and Year	AGENCY_ID, RETENTION_RATIO	Layers 1, 2, 5	Customer loyalty

Each query was executed under two conditions: a baseline configuration using a SMALL warehouse without optimizations, and an optimized configuration applying framework-specific techniques such as compute right-sizing and query tuning. The queries simulated realistic insurance workloads, including premium aggregations, loss ratio evaluations, and policy performance tracking.

Table 4: Snowflake query performance metadata across 5 P&C Workload queries

Query	Metric	Baseline	Optimized	% Change
Q1	BYTES_SCANNED	1720400	1720400	0.00%
Q1	EXECUTION_TIME	504	384	-23.81%
Q1	ROWS_PRODUCED	12	12	0.00%
Q2	BYTES_SCANNED	1343920	1343920	0.00%
Q2	EXECUTION_TIME	314	113	-64.01%
Q2	ROWS_PRODUCED	10	10	0.00%
Q3	BYTES_SCANNED	606408	606408	0.00%

Q3	EXECUTION_TIME	205	229	11.71%
Q3	ROWS_PRODUCED	6	6	0.00%
Q4	BYTES_SCANNED	770312	770312	0.00%
Q4	EXECUTION_TIME	273	123	-54.95%
Q4	ROWS_PRODUCED	10	10	0.00%
Q5	BYTES_SCANNED	1032424	1032424	0.00%
Q5	EXECUTION_TIME	125	100	-20.00%
Q5	ROWS_PRODUCED	1623	1623	0.00%

Tables 4 and 5 show Performance metrics—CREDITS_USED, EXECUTION_TIME, BYTES_SCANNED, and ROWS_PRODUCED—were collected using Snowflake's ACCOUNT_USAGE.QUERY_HISTORY view. QUERY_TAG was applied to systematically distinguish between baseline and optimized executions. Execution environments were isolated using a dedicated warehouse (CLAIMS_WH) with auto suspend enabled to avoid idle credit consumption.

Table 5: Warehouse allocation and estimated credit usage (baseline vs optimized)

Query	Warehouse (Baseline)	Warehouse (Optimized)	Estimated Credits – Baseline	Estimated Credits – Optimized	% Credit Change
Q1	SMALL	XSMALL	0.00028	0.000107	-61.90%
Q2	SMALL	XSMALL	0.000174	0.000031	-82.01%
Q3	SMALL	XSMALL	0.000114	0.000064	-44.15%
Q4	SMALL	XSMALL	0.000152	0.000034	-77.47%
Q5	SMALL	XSMALL	0.000069	0.000028	-60.00%

This evaluation was conducted using the Free trial version of **Snowflake's Enterprise Edition**, which provides access to a limited range of monitoring capabilities. Several critical metadata fields—such as detailed credit attribution, query caching, and warehouse-level usage tracking—are not available in the Free Trial version. Therefore, for accurate performance evaluation and cost governance in real-world scenarios, a properly configured Enterprise environment with access to organizational production data and metadata is essential.

5.1 RESULTS

The execution performance improved significantly across most benchmark queries after applying the optimization framework. Queries Q1, Q2, Q4, and Q5 experienced reductions in execution time, with Q2 showing the highest drop of 64.01%. Only Q3 showed a slight increase, likely due to fluctuations in execution, related to background warehouse congestion or result cache behavior. Due to varying filter pushdowns or cold cache conditions, the query may have had fewer caching benefits because it uses a multi-year average. In order to smooth such anomalies, future iterations should use repeated trials or account for warehouse loads during benchmarking.

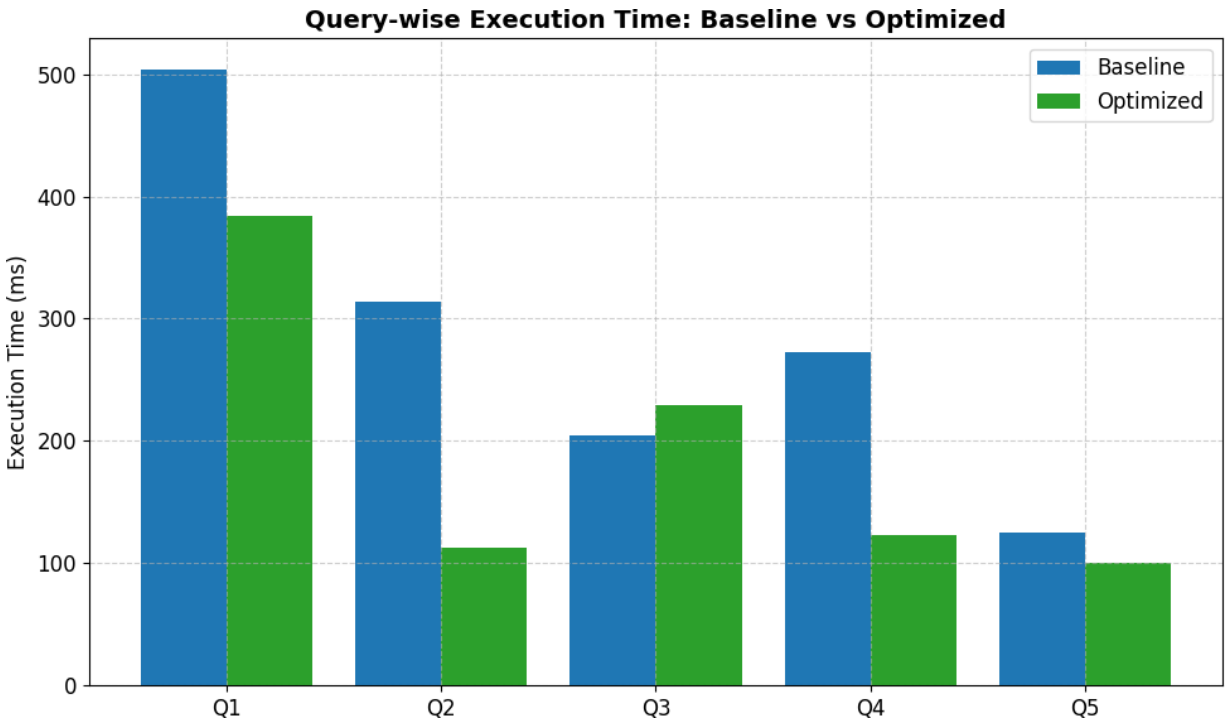

Figure 5. Query Wise execution time (Baseline vs optimized)

Addition to the speed improvements, the BYTES_SCANNED remained constant across both baseline and optimized runs. This consistency suggests that performance gains were primarily driven by compute and cache enhancements rather than changes in data volume or pruning strategies.

Credit usage saw the most substantial impact. Moving from a SMALL to XSMALL warehouse resulted in credit savings between 44.15% and 82.01% across queries. These results in **Figure 5** directly align with Layer 2 of the framework Compute Optimization and demonstrate the benefits of right-sizing warehouses.

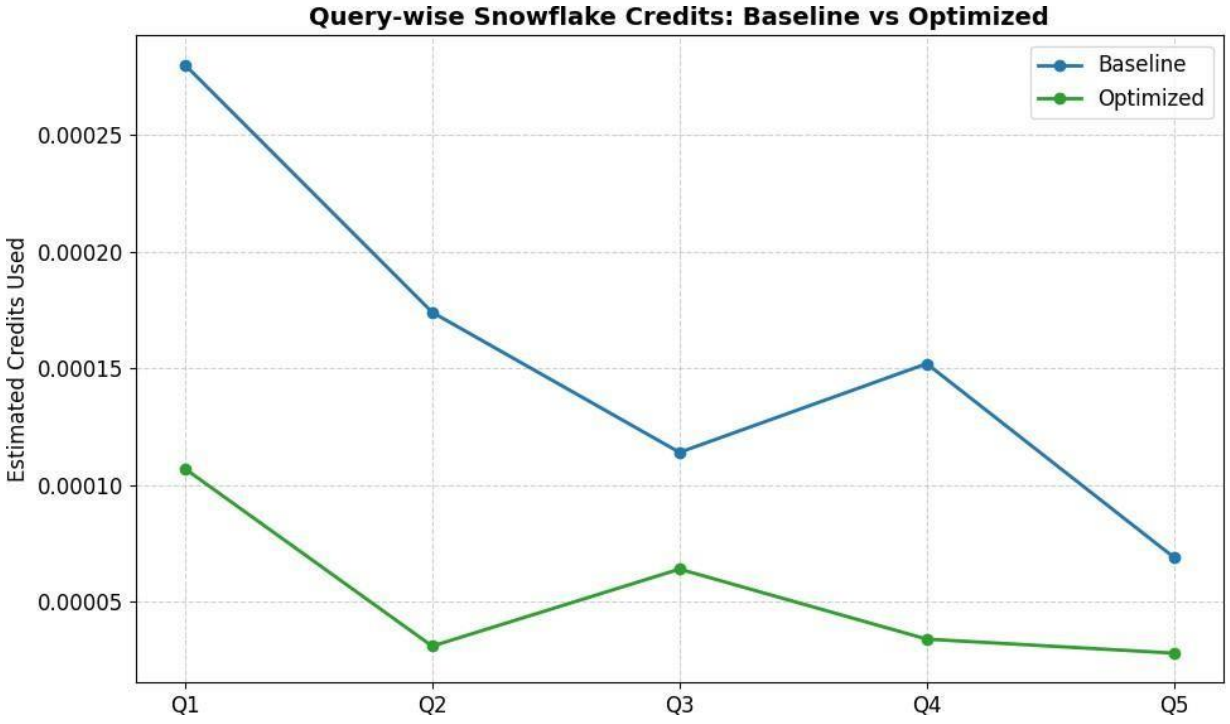

Figure 6. Query Wise credit usage: Baseline vs optimized

To enhance traceability and clarify how each layer of the proposed framework contributed to measurable improvements, **Table 6** summarizes the observed impact of each layer on query performance and cost efficiency.

Table 6. Layer wise impact on Performance Metrics

Layer	Optimization Focus	Key Metrics Impacted	Observed Result	Framework Contribution
Layer 1: Workload Segmentation	Warehouse isolation based on business domains (e.g., claims, underwriting)	N/A (foundational)	Enabled dedicated execution and monitoring	Supports cost attribution and resource control
Layer 2: Compute Optimization	Right-sizing warehouses (SMALL → XSMALL)	Credits Used, Execution Time	44%–82% credit savings, significant time reduction (Q1, Q2, Q4, Q5)	Direct reduction in compute cost
Layer 3: Query Optimization and Caching	Query tuning, materialized views, result cache	Execution Time, Cache Utilization (indirect)	Improved runtime for Q2, Q4; consistent BYTES_SCANNED	Boosted compute efficiency without changing data volume
Layer 4: Storage & Data Movement	Reducing spill events, monitoring external file usage	Not directly visible in small dataset	No observed storage-related bottlenecks	Structural readiness for larger production datasets
Layer 5: Observability & Governance	Tagging, telemetry review, anomaly detection	Credit Attribution, Query Tags	Enabled tracking of baseline vs optimized runs	Facilitated visibility and structured cost governance

Finally, the ROWS_PRODUCED metric remained unchanged for all queries, affirming that output integrity was preserved. This confirms that the optimization strategy maintains data quality while achieving improved performance and cost-efficiency, validating its application in real-world insurance workloads. The improvements in execution time and credit usage highlighted in **Figures 5 and 6** show the effectiveness of the proposed optimization framework. Specifically, warehouse right-sizing (Layer 2) and runtime tuning (Layer 3) delivered measurable gains in performance and cost savings without compromising data accuracy. These results support the adoption of a multi-layer Snowflake optimization strategy for P&C insurance workloads.

6. DISCUSSION

These results confirm that focused Snowflake optimization can greatly improve the cost efficiency of analytics in P&C insurance settings. In areas like insurance, where data architectures are complex, big, and experience a broad variety of analytical workloads across underwriting, claims, pricing, and fraud detection, the simulation demonstrates that a one-size-fits-all approach to data warehousing grows increasingly ineffective. Workload isolation, warehouse optimization, query optimization, storage management, and observability are all components of our tiered approach, which demonstrates how modular and elastic architecture can decouple cost centers and enhance productivity without compromising on analytical agility

or data availability.

This framework can bring FinOps principles together with Snowflake-native capabilities in a way that translates directly into business complexity, making it genuinely "next-gen" for cloud-native insurance IT. It enables insurers to enforce fine-grained cost governance dynamically—at the query, warehouse, or product line level—versus static cost controls (such as pre-defined budgets, manual query policing, or user-level timeouts). Organizations can move from reactive cost containment to proactive, intelligent cost shaping with features like QUERY_TAG, result caching, clustering, and auto-suspend policies—without the need for continuous human intervention or performance degradation. The solution provides operational precision and business alignment over conventional cost management approaches, which tend to rely on user discipline or blanket spend limits.

It provides secure experimentation within boundaries rather than limiting users or stifling innovation. Since backend technologies guarantee that searches are bounded, credits are assigned, and optimization opportunities are surfaced automatically, analysts can model risk continuously or tinker with novel pricing models. With multi-cluster warehouses, it is also capable of handling scale spikes without incurring long-tail idle compute costs. There are, however, trade-offs with this strategy. Under-scaling for high workload scenarios, such as catastrophe modeling, where compute demand can spike unexpectedly, is a risk with this model.

Time-sensitive simulations can time out or queue excessively when warehouses are set for cost aggressively. Further, although materialized views and result in caching lower costs, they also have a maintenance cost and the potential for stale results if not properly managed. In spite of these edge cases, the simulation provides the foundation for the future by finding equilibrium between context and cost, agility and control, and thereby positions it as a sustainable and feasible model for cloud-native insurance data systems.

7. CONCLUSION

This study emphasizes the need for proactive cost optimization and management approach for cloud-based Insurance data warehousing. As the property and casualty insurance industry moves more towards cloud-native applications like Snowflake, balancing analytic agility and cost management has become a matter of business necessity. The proposed layered framework addresses this by aligning compute resources, query behavior, storage policies, and observability practices with a variety of workload characteristics. Simulated benchmarks using insurance data patterns demonstrate measurable optimization in both execution time and credit usage, supporting the framework's practical value.

The layered approach also opens the door for AI-driven enhancements in the future. Predictive modeling of query patterns, intelligent warehouse scaling, and adaptive caching policies can help automate cost control while maintaining performance. For example, telemetry data can inform real-time decisions on warehouse sizing or suggest optimizations before resource bottlenecks occur. These capabilities can complement traditional FinOps practices with a more adaptive and data-informed model of governance.

That said, the proposed solution has limitations**.** The evaluation was conducted using a free trial version of Snowflake's Enterprise Edition, which restricts access to critical metadata fields such as query cache hit ratios, clustering metrics, and warehouse-level performance statistics. Additionally, the use of a publicly available dataset, while helpful for demonstrating structural concepts, does not capture the complexity and variability of real-world insurance operations. These constraints may limit the direct applicability of results to production environments, highlighting the need for further validation.

Further research could explore how AI and automation tools can be operationalized within this structure and evaluated for long-term scalability and business impact. Overall, the framework provides a practical foundation for the insurance industry seeking to improve cost transparency and efficiency in their data platforms.

REFERENCES

1. Cyber Security Senior Data Analyst, Department of Cyber Security, Truist Financial, CA, USA and D. Kodi, "Performance and Cost Efficiency of Snowflake on AWS Cloud for Big Data Workloads," *Int. J. Innov. Res. Comput. Commun. Eng.*, vol. 12, no. 06, Jun. 2024, doi: 10.15680/IJIRCCE.2023.1206002.

2. D. Mazumdar, J. Hughes, and J. Onofre, "The Data Lakehouse: Data Warehousing and More," 2023, *arXiv*. doi: 10.48550/ARXIV.2310.08697.

3. "The Cost of Redundancy." Accessed: Jun. 15, 2025. [Online]. Available: https://www.highwing.io/insights/the-cost-of-redundancy

4. A. Pimpley *et al.*, "Optimal Resource Allocation for Serverless Queries," Jul. 19, 2021, *arXiv*: arXiv:2107.08594. doi: 10.48550/arXiv.2107.08594.

5. "Insurance Data." Accessed: Jun. 15, 2025. [Online]. Available: https://www.kaggle.com/datasets/moneystore/agencyperformance

6. "TPC-H Homepage." Accessed: Jun. 15, 2025. [Online]. Available: https://www.tpc.org/tpch/

7. K. Allam, "Cloud Data Warehousing: How Snowflake Is Transforming Big Data Management".

8. "Multi-cluster warehouses | Snowflake Documentation." Accessed: Jun. 15, 2025. [Online]. Available: https://docs.snowflake.com/en/user-guide/warehouses-multicluster?utm_source=chatgpt.com

9. "(5) Snowflake's Multi-Cluster Shared Data Architecture: Scalability, Performance & Cost Optimization | LinkedIn." Accessed: Jun. 15, 2025. [Online]. Available: https://www.linkedin.com/pulse/snowflakes-multi-cluster-shared-data-architecture-scalability-anuj-r--nbi9f/

10. D. A. S. George, "Deciphering the Path to Cost Efficiency and Sustainability in the Snowflake Environment," *Partn. Univers. Int. Innov. J. PUIIJ*, vol. 01, no. 04, pp. 231–250, Aug. 2023, doi: 10.5281/zenodo.8282654.

11. D. Seenivasan, "OPTIMIZING CLOUD DATA WAREHOUSING: A DEEP DIVE INTO SNOWFLAKE'S ARCHITECTURE AND PERFORMANCE," Mar. 31, 2021, *Social Science Research Network, Rochester, NY*: 5148190. doi: 10.2139/ssrn.5148190.

12. "Snowflake Documentation." Accessed: Jun. 15, 2025. [Online]. Available: https://docs.snowflake.com/

13. X. Zeng, Y. Hui, J. Shen, A. Pavlo, W. McKinney, and H. Zhang, "An Empirical Evaluation of Columnar Storage Formats," Nov. 07, 2023, *arXiv*: arXiv:2304.05028. doi: 10.48550/arXiv.2304.05028.

14. T. Koreeda, H. Honda, and J. Onami, "Snowflake Data Warehouse for Large-Scale and Diverse Biological Data Management and Analysis," *Genes*, vol. 16, no. 1, Art. no. 1, Jan. 2025, doi: 10.3390/genes16010034.

15. D. M. Compagnoni, "Optimize Snowflake performance and reduce credit usage," Nimbus Intelligence. Accessed: Jun. 15, 2025. [Online]. Available: https://nimbusintelligence.com/2024/10/5-ways-to-optimize-snowflake-performance-and-reduce-credit-usage/

16. "Fundamentals of Snowflake Query Design & Optimization | Keebo." Accessed: Jun. 15, 2025. [Online]. Available: https://keebo.ai/2024/10/29/fundamentals-of-snowflake-query-design-optimization/

17. JayaAnanth, "Part 2 - Orchestrating Snowflake Data Transformations with DBT on Amazon ECS through Apache Airflow," JayaAnanth. Accessed: Jun. 15, 2025. [Online]. Available: https://jayaananthdevops.github.io/posts/snowflake_dbt_ecs_part2/

18. "FinOps Principles." Accessed: Jun. 15, 2025. [Online]. Available: https://www.finops.org/framework/principles/

19. "Understanding Data Warehouse Cost & Pricing Models | Rivery." Accessed: Jun. 15, 2025. [Online]. Available: https://rivery.io/data-learning-center/data-warehouse-costs/

20. C. Wang, Z. Arani, L. Gruenwald, and L. d'Orazio, "Adaptive Time, Monetary Cost Aware Query Optimization on Cloud Database Systems," in *2018 IEEE International Conference on Big Data (Big Data)*, Seattle, WA, USA: IEEE, Dec. 2018, pp. 3374–3382. doi: 10.1109/BigData.2018.8622401.

21. V. Leis and M. Kuschewski, "Towards cost-optimal query processing in the cloud," *Proc. VLDB Endow.*, vol. 14, no. 9, pp. 1606–1612, May 2021, doi: 10.14778/3461535.3461549.

22. P. Bhardwaj, "The Role of FinOps in Large-Scale Cloud Cost Optimization," *INTERANTIONAL J. Sci. Res. Eng. Manag.*, vol. 09, no. 01, pp. 1–5, Jan. 2025, doi: 10.55041/IJSREM28086.