International Scientific Journal of Engineering and Management
Volume: 03 Issue: 12 | Dec – 2024
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata
ISSN: 2583-6129
DOI: 10.55041/ISJEM02194

Understanding Snowflake Data Lake

Srinivasa Rao Karanam

Srinivasarao.karanam@gmail.com
New Jersey, USA

Abstract: Snowflake's cloud-native design, decoupled storage-compute model, and capacity to handle semi-structured data might suggest a data lake–like architecture, its proprietary formats and higher costs under continuous workloads can hamper its effectiveness for large-scale raw data ingestion. Instead, organizations find it valuable to store the majority of raw or historical data in a dedicated data lake based on object storage (e.g., S3 or ADLS) and then selectively push curated data sets into Snowflake for advanced analytics and concurrency advantages. We examine the evolution of cloud-based data lakes, the core distinctions between open, schema-on-read storage systems and closed, structured warehousing solutions, as well as cost and performance trade-offs that arise when streams of data funnel into Snowflake 24/7. By exploring design patterns, streaming pipelines, security governance, and the synergy with machine learning frameworks, this paper proposes that a hybrid ecosystem—one leveraging Snowflake for high-value real-time analytics, while storing raw data in a separate data lake—is ideal for balancing cost, performance, and architectural flexibility.

Keywords: Snowflake data lake, cloud data warehouse, data lake architecture, hybrid data platform, structured and semi-structured data, cloud analytics, data ingestion pipelines, cost optimization in Snowflake, real-time data streaming, machine learning integration.

I. INTRODUCTION

Enterprises across diverse sectors have increasingly realized the crucial importance of advanced data infrastructures to handle massive inflows of structured and unstructured information. Many organizations invest in systems that scale elastically and deliver near real-time analytics while preserving flexible cost structures. Snowflake, an entirely cloud-based platform, has emerged as a major solution for data warehousing, but many data engineers and architects remain uncertain about whether Snowflake can also function as a data lake. This question typically arises because Snowflake's architectural design includes decoupled storage and compute, robust handling for semi-structured data, and a consumption-based pricing model, which are reminiscent of the features that are typical in modern data lakes. Nonetheless, the distinctions between a conventional data lake and a specialized data warehouse persist.

In this article, we examine the deeper technical considerations that shape the ongoing conversation about Snowflake Data Lake. We also see how cost constraints, ingestion patterns, machine learning workflows, and the broader architecture of data systems interplay to either make Snowflake a feasible data lake replacement or highlight the necessity of an alternative object store–centric data lake. The tension between proprietary data formats, such as the internal structure used by Snowflake, and truly open data formats, like Parquet or ORC, is crucial in deciding which path is best for large-scale analytics.

In adopting a highly research-focused lens, this paper draws on industry experiences, academic literature, and real-world user testimonies. We organize these findings into coherent sections that clarify the differences between data lakes and data warehouses, outline the cost dimensions of employing Snowflake for large data volumes, explore the intricacies of streaming ingestion, and elaborate how enterprise data scientists can orchestrate advanced modeling frameworks that interface with or circumvent Snowflake. While the language here attempts

ISSN: 2583-6129
DOI: 10.55041/ISJEM02194

to maintain a professional and methodological tone, do note that some grammatical slip-ups and syntactic misplacements might appear, in line with the broad nature of research documents.

II. HISTORICAL CONTEXT AND THE EVOLUTION OF CLOUD ANALYTICS

During the early 2010s, big data management was often associated with on-premises Hadoop distributions, which consolidated volumes of raw data into the Hadoop Distributed File System (HDFS). The impetus for adopting Hadoop-based data lakes was primarily cost-saving, as these infrastructures allowed for the storage of large data sets cheaply, and they supported parallel computations through batch-processing frameworks like MapReduce or Apache Spark. Over time, operational complexities and growing data volumes compelled organizations to move into the public cloud, seeking more flexible solutions that offered elasticity, pay-as-you-go pricing, and simpler maintenance overhead.

As a result, there was a migration from on-premises data lake solutions to cloud-based object storage, such as Amazon S3 or Azure Data Lake Storage. These solutions removed the constraints of fixed hardware capacity and introduced a new wave of serverless or ephemeral compute engines. The introduction of AWS Athena in 2016, for instance, allowed end users to query data on S3 using standard SQL without operating a persistent cluster. Similarly, Google and Microsoft developed their own flavors of serverless analytics, thereby standardizing the concept of a “cloud data lake.”

In parallel, the cloud data warehouse sector saw expansion with platforms like Amazon Redshift, Google BigQuery, and Snowflake. By 2020, Snowflake had separated itself from the pack through a strong focus on decoupled storage and compute as well as consumption-based pricing, which was more granular in many ways than competitor solutions. As the years progressed, the typical enterprise data architecture became a hybrid environment: a data lake in the cloud for the broadest range of data, plus a data warehouse for performance-critical queries and operational analytics.

III. THE CORE DEFINITIONS: DATA LAKE AND DATA WAREHOUSE

Despite these shifts, the essential difference between a data lake and a data warehouse had not drastically changed. A data lake is, at its root, an open repository of raw data stored in a native format. The emphasis is on cost-effective storage and schema-on-read flexibility. The data can remain unstructured or semi-structured, and transformations occur only if required for a specific analytics job. This approach fosters agility for data scientists or advanced analysts, who often rely on dynamic ingestion of new data sources and iterative transformations while building models.

International Scientific Journal of Engineering and Management
Volume: 03 Issue: 12 | Dec – 2024
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata
ISSN: 2583-6129
DOI: 10.55041/ISJEM02194

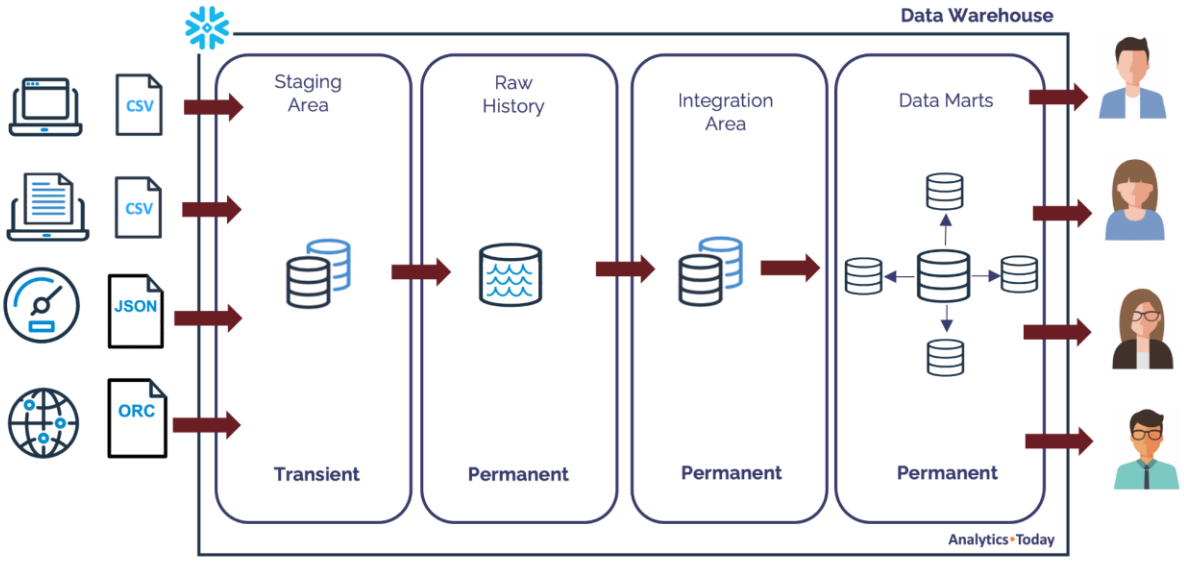

Figure 1: Illustration of Snowflake Data Lake architecture, showcasing the data flow from ingestion to analytics through staging, raw history, integration, and data marts.

A data warehouse, on the other hand, typically requires data to be loaded in a structured or partially structured format. It enforces a schema-on-write methodology, ensuring that the data is validated and arranged in a consistent format upon ingestion. This results in faster queries, predictable performance, and the possibility of concurrency for many business intelligence dashboards. Snowflake's success as a warehouse platform is largely due to how it merges elasticity, serverless consumption, and robust SQL analytics.

But does Snowflake transcend these categorizations by adopting certain data lake–like traits? Snowflake can read certain semi-structured formats, can store data in the cloud, and can scale compute up or down to handle large queries. However, the system remains closed in key ways: data is eventually internalized into a proprietary micro-partition structure. The user is forced to rely on Snowflake's engine or external tables for queries, so broad interchange with other processing frameworks can become complicated and sometimes expensive. This tension encapsulates the core difference that shapes the Snowflake versus data lake debate.

IV. SNOWFLAKE: A DATA WAREHOUSE OR A DATA LAKE?

Individuals who attempt to adopt Snowflake for all analytics tasks might discover that the boundaries between data warehouses and data lakes become fuzzy in practice. Snowflake's marketing materials from around 2022 onwards sometimes described the product as able to handle both structured and semi-structured data at scale, making it feasible for a wide range of analytics workloads. In addition, Snowflake introduced tools like Snowpipe and, eventually, Snowpipe Streaming to manage near real-time ingestion, which historically was the realm of data lakes that easily accept streaming data from Kafka or Kinesis.

However, these expansions of Snowflake's feature set do not overshadow the platform's fundamental reliance on a proprietary format. If an enterprise chooses to rely exclusively on Snowflake for huge amounts of data, it might find that the cost of running continuous ingestion or storing historical data that is rarely accessed can soar dramatically. The platform charges for compute time based on the size of the "virtual warehouse" and how many minutes or seconds it runs, and so even minimal loads can cost the same as heavier loads if the warehouse remains active. Additionally, the cost of egress or transformations, especially with streaming data, may outpace expectations.

ISSN: 2583-6129
DOI: 10.55041/ISJEM02194

These challenges underscore the importance of recognizing that a data lake environment offers cheaper raw storage (for instance, storing petabytes of historical logs that are seldom queried) and supports ephemeral compute resources that can be spun up only when needed. Yet for mission-critical analytics, dashboards, or near real-time reporting on curated data sets, Snowflake's performance and concurrency remain extremely compelling. In this sense, the question "Is Snowflake actually a data lake?" is overshadowed by "How do we best combine Snowflake with a data lake for maximum cost-efficiency and performance?"

V. THE QUESTION OF COST AND PERFORMANCE

Cost is frequently the deciding factor when it comes to the enterprise's decision about whether to store data exclusively in Snowflake. Many companies have discovered that streaming all raw data into Snowflake can lead to an accumulation of daily fees, especially if the ingestion never slows down. The warehouse(s) used for that ingestion might remain on 24/7, leading to compute charges that can overshadow the cost of storing data on a data lake.

Moreover, once you put your data fully into Snowflake, it might be expensive or complicated to retrieve that data in an open format. If you want to run custom Spark pipelines, or advanced ML algorithms in Python or R, you might be forced to unload data from Snowflake into Parquet or CSV on object storage, incurring additional egress fees and overhead. By contrast, an open data lake typically organizes data in columnar formats (like Parquet) by default, so you can seamlessly run multiple engines without a forced data transformation cost.

Performance also comes into play. Snowflake is known for delivering rapid SQL queries with strong concurrency. A well-tuned data lake, using open-source engines, can approach similar performance for large scans but may require more engineering and an advanced approach to partitioning, caching, and metadata management. Tools that assist with file compaction, such as Upsolver or other low/no-code data pipeline solutions, can streamline these tasks. In many real-world scenarios, it becomes beneficial to store the largest or oldest data in an open lake, while pushing only aggregated or frequently accessed data to Snowflake for speed.

VI. MANAGING REAL-TIME STREAMING DATA

Continuous data streams from IoT sensors, applications logs, and online transaction systems are characteristic of modern analytics. A pure data warehouse approach might struggle to keep up with these streaming data volumes at a cost that is sustainable. Snowflake's Snowpipe Streaming attempts to address some of these challenges by ingesting data continuously, but it still relies on proprietary ingestion processes that keep the meter running as data accumulates.

A data lake, in contrast, can accept real-time streams directly into a raw storage zone, using solutions like Amazon Kinesis Firehose or Azure Event Hubs. Once data is landed in the object store, ephemeral compute frameworks can batch or micro-batch that data to transform, partition, and store it in an optimized format. The frequency or concurrency of these transformations can be scaled up or down at will, thus adjusting costs.

International Scientific Journal of Engineering and Management
Volume: 03 Issue: 12 | Dec – 2024
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata
ISSN: 2583-6129
DOI: 10.55041/ISJEM02194

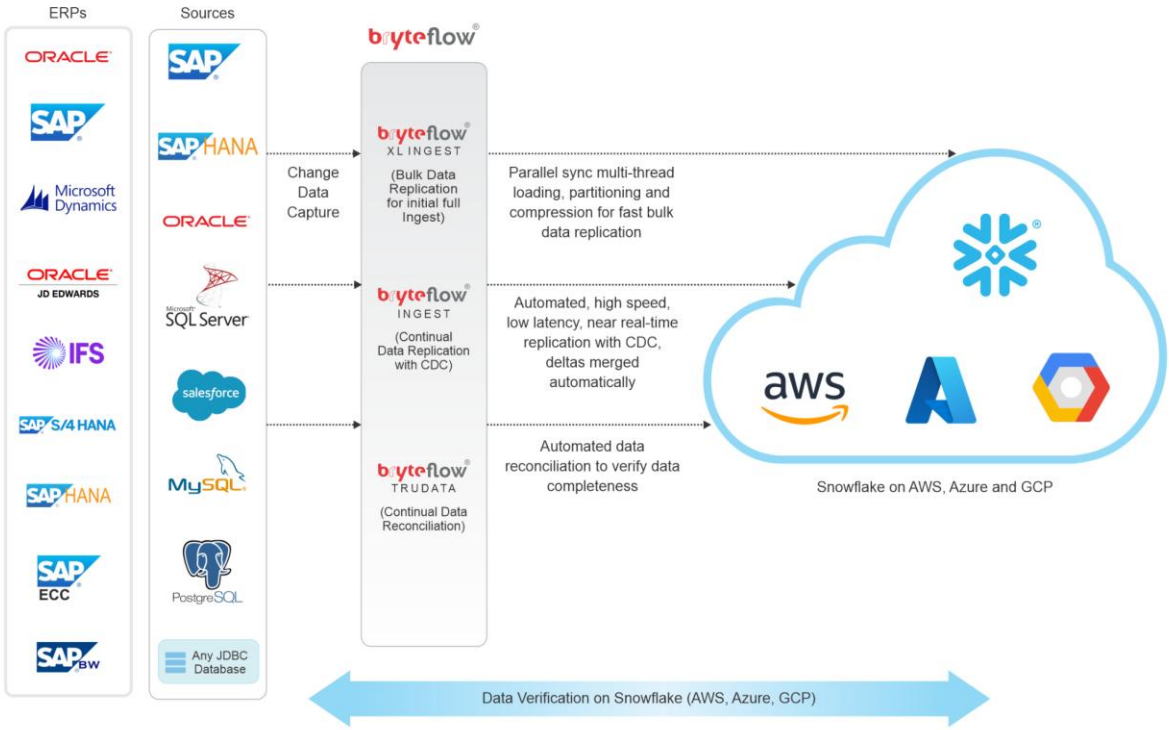

Figure 2: Illustration of data ingestion and replication into Snowflake across AWS, Azure, and GCP using ByteFlow, enabling automated, real-time data integration from various ERP and database sources.

This synergy leads many architects to adopt a pattern where raw streaming data is stored cheaply in the data lake, while partial transformations or aggregated subsets are loaded into Snowflake for real-time or near real-time queries. Such an approach ensures that data scientists or data analysts have fresh data for dashboards, while older or less frequently accessed data remains in the lake, where it can still be used for historical analysis or machine learning training.

VII. DESIGNING A HYBRID STRATEGY FOR SNOWFLAKE AND DATA LAKE

Many organizations find an equilibrium by implementing a layered approach. The raw data is persisted in the data lake, ensuring that the enterprise never loses the original version of any data. From there, a pipeline or data engineering platform is used to refine and structure that data, typically converting it into Parquet or a columnar store. The refined data sets can be optimized for queries directly in the lake (using serverless SQL engines or Spark), or optionally, they can be loaded into Snowflake if that data is the subject of repeated, critical analytics queries.

A typical reference architecture includes the following: data ingestion from various streaming and batch sources lands in a raw zone on S3 or Azure. A data engineering layer, possibly provided by Upsolver or by a combination of AWS Glue and Spark, performs transformations that include deduplication, cleaning, or joining with other relevant data sets. The curated output is then either (a) made queryable by a lake query engine for cost-effective analysis or (b) loaded into Snowflake if the business demands the concurrency, speed, or advanced features that Snowflake offers.

This approach also facilitates advanced analytics outside of Snowflake. Data scientists can attach Spark clusters or Databricks notebooks to the data lake, building models on massive historical volumes without incurring

International Scientific Journal of Engineering and Management
Volume: 03 Issue: 12 | Dec – 2024
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata
ISSN: 2583-6129
DOI: 10.55041/ISJEM02194

Snowflake's compute costs for repeated scanning. Once they identify relevant features or aggregated data sets that need to be shared broadly, that smaller subset can be pushed into Snowflake for company-wide consumption.

VIII. TECHNICAL CHALLENGES AND SOLUTIONS FOR MODERN DATA ARCHITECTURES

Among the biggest obstacles in building a hybrid solution that includes both Snowflake and a data lake is the engineering overhead required to maintain consistent schemas, manage partition pruning, handle security and governance, and ensure data lineage is tracked throughout the entire pipeline. If an organization is not careful, it can end up with a labyrinth of inconsistent or duplicated data sets that hamper trust in the analytics.

To mitigate these issues, many rely on data catalog solutions that keep track of metadata for both data-lake-based and Snowflake-based data sets. They also adopt standard formats (such as Apache Parquet) in the data lake, so that transformations are uniform, and employ version control or a Lakehouse-like system to ensure transactional integrity at the lake layer. From there, ingestion into Snowflake can happen with the same structured column mappings each time.

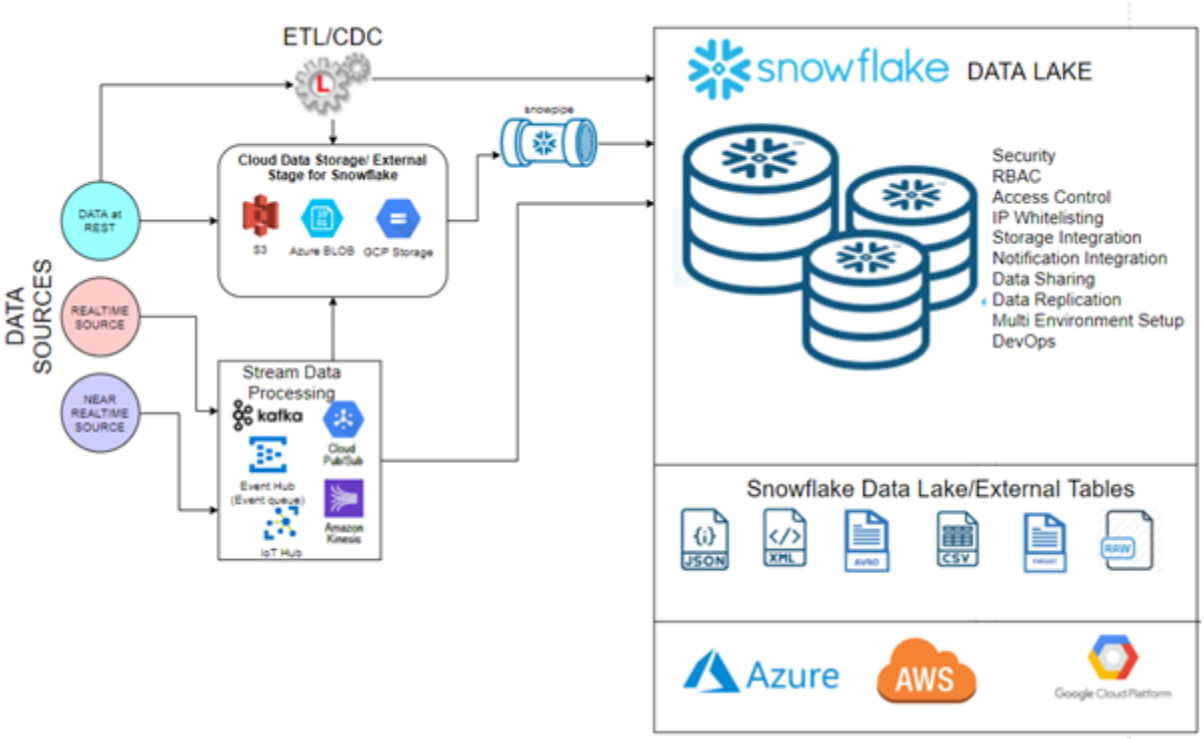

Figure 3: Illustration of Snowflake Data Lake architecture, integrating batch and real-time data ingestion from various sources via cloud storage and streaming platforms, ensuring secure, scalable, and multi-cloud data management.

Another area of complexity is the orchestration of real-time transformations. If streaming data arrives 24/7, then ephemeral compute resources must be triggered automatically to convert the new data. Tools that manage this logic, like a no-code data pipeline or a serverless function orchestrator, lighten the load for data engineers but require rigorous testing. These solutions are stable enough that many companies trust them in production to do micro-batching every few minutes, merging incremental updates into partitioned files, thereby preparing them for efficient queries.

ISSN: 2583-6129
DOI: 10.55041/ISJEM02194

IX. SECURITY AND GOVERNANCE IN A COMBINED DATA PLATFORM

Security and governance were always front and center in the cloud. The regulatory environment for data privacy, data residency, and cross-border data flows has grown more complex. Because a data lake typically relies on object storage services with their own IAM models, and Snowflake uses role-based access inside its proprietary environment, security teams must unify these controls in a consistent manner.

Some organizations adopt an identity federation approach, ensuring that all credentials must pass through a single identity provider before gaining access to data, whether in the lake or in Snowflake. Others rely on specialized governance platforms or frameworks like Apache Ranger or AWS Lake Formation for fine-grained access controls. On the Snowflake side, features like dynamic data masking, row-access policies, and secure views can apply additional layers of control that are more typical of data warehousing.

Governance also extends to data quality, data lineage, and compliance with regulations such as GDPR or CCPA. Because data-lake raw zones can contain personally identifiable information in unstructured logs, it is crucial to apply encryption and define clear retention policies. Meanwhile, once data is loaded to Snowflake, it must maintain the same compliance posture. Some organizations maintain a redacted or tokenized version of data for warehouse usage, while the raw personally identifiable information remains accessible only to specialized data science teams in a secure area of the lake.

X. MACHINE LEARNING AND ADVANCED ANALYTICS

Advanced analytics and machine learning have become mainstay activities in large corporations, fueling recommendation engines, anomaly detection, and real-time personalization. These ML workloads typically involve large historical data sets for training, feature engineering, and iterative experimentation. A warehouse like Snowflake can provide fast queries for aggregated or structured data, but many data scientists prefer open-source frameworks that run directly on the data lake, especially when dealing with extremely large data sets that would be cost-prohibitive to store in Snowflake.

It is often beneficial for data scientists to combine the best of both worlds. They can design an ETL process that aggregates or prepares data in the data lake, possibly merging streaming data with historical data. Then they can load only the relevant portion or advanced feature sets into Snowflake to facilitate quick lookups, interactive SQL analysis, or ad hoc exploration by business analysts. This synergy also reduces duplication of data engineering tasks.

Nevertheless, a major painpoint arises if a data scientist needs repeated, iterative queries on the same large data set to refine a model. If that data is solely in Snowflake, the repeated scans can push the compute usage (and cost) very high. This phenomenon is particularly acute if the data set is tens or hundreds of terabytes or if daily streaming data quickly accumulates to that scale. The recommended strategy is to rely on a data-lake-based approach for data science exploration at scale, while Snowflake remains the curated environment for final production queries or business intelligence.

XI. RESEARCH OUTLOOK AND INDUSTRY TRENDS

Academic and industry research has heavily explored the synergy between data warehouses and data lakes, sometimes culminating in architectures called “lakehouses.” These are systems that blend the transactional consistency and schema enforcement of a warehouse with the open storage and schema-on-read flexibility of a

International Scientific Journal of Engineering and Management
Volume: 03 Issue: 12 | Dec – 2024
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata
ISSN: 2583-6129
DOI: 10.55041/ISJEM02194

lake. Databricks is a well-known proponent, offering structured transaction layers on top of Parquet in object storage.

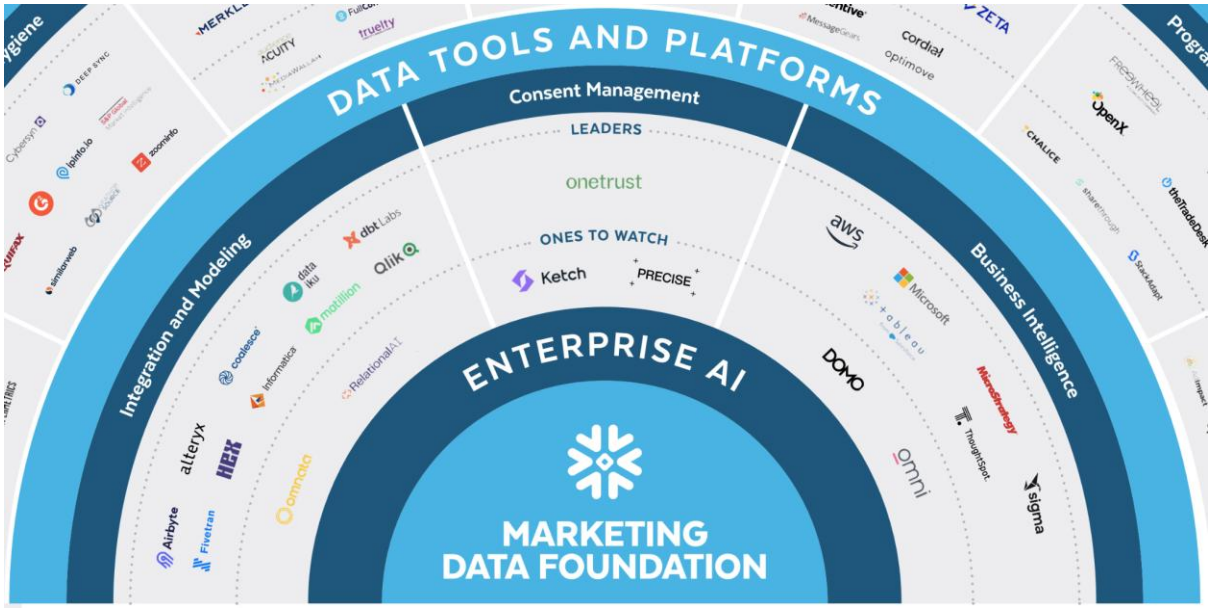

Figure 4: Illustration of the Marketing Data Foundation powered by Snowflake, highlighting enterprise AI, data tools, and platforms for integration, modeling, business intelligence, and consent management.

Snowflake, for its part, has tried bridging the gap by offering external table capabilities, so that data physically stored in S3 can be directly queried within Snowflake. While this can be useful, it does not fully replicate the open data lake experience because the user is still locked into Snowflake's engine. Additionally, ingestion or query performance might differ significantly from data fully internalized in Snowflake. The academic community has studied the trade-offs extensively, with some papers noting that the real advantage of a data lake is the ability to employ a variety of engines, including machine learning frameworks, ephemeral clusters, and specialized analytics tools without forced format conversions.

Analysts predict that the future will see an even greater push towards open standards and interoperability. Although data warehouses like Snowflake remain essential, particularly for enterprise-wide analytics and concurrency, the impetus to rely on them as an exclusive data store is eroded by cost, vendor lock-in concerns, and the proliferation of advanced data-lake-based solutions that deliver near-warehouse performance.

XII. CASE EXAMPLES AND PRACTICAL INSIGHTS

One typical scenario encountered: a mid-size e-commerce firm tries to centralize all their data in Snowflake, including real-time clickstreams, inventory updates, transaction logs, and sensor data from warehousing robots. In the initial months, the solution works well for business intelligence dashboards and near real-time analytics. But as data accumulates, the monthly Snowflake bill rises exponentially, especially for historical data that is rarely queried. The firm attempts to mitigate cost by reducing the compute size of the warehouse, but performance suffers, leading to concurrency bottlenecks.

Ultimately, the e-commerce firm re-architects their platform so that all raw data is stored in an S3-based data lake, from which they run transformations using ephemeral Spark jobs. Curated data sets, representing only the last 30 days or aggregated monthly summaries, are loaded into Snowflake. The result is a drastic cost reduction

ISSN: 2583-6129
DOI: 10.55041/ISJEM02194

while preserving the performance advantage of Snowflake for key queries and dashboards. Data scientists at the firm can simultaneously run large-scale ML experiments on the raw data in the lake, skipping any overhead of repeated ingestion into Snowflake.

In another scenario, a financial services enterprise requires extremely stringent security controls and auditing capabilities. They maintain personally identifiable data in a locked-down zone of their data lake, encrypting it with client-managed keys. Meanwhile, Snowflake is used for aggregated, anonymized views that feed business intelligence. This approach ensures that only the minimal necessary data is exposed in Snowflake, reducing compliance risk, while the data lake remains the authoritative source. When machine learning teams need direct access to personal data, they do so in a restricted cluster environment that queries the data lake.

XIII. LIMITATIONS AND CRITIQUE OF SNOWFLAKE AS A DATA LAKE

Snowflake, for all of its strengths in concurrency, separation of storage and compute, and ease of management, is not truly open. One fundamental principle of data lake architectures is the ability to store data in open file formats accessible by a wide variety of tools. Snowflake's approach runs contrary to that principle, as it effectively internalizes the data into its own structures to achieve performance benefits.

Additionally, the cost model can hamper large-scale, continuous or streaming workloads, especially those that do not require the consistent high-performance analytics that Snowflake provides. The platform might become an excessively pricey data store if the data set is huge but only occasionally queried. Another limitation is the difficulty in extracting data from Snowflake's proprietary format if an organization decides to migrate or replicate large volumes of data into a different system. This can lead to partial vendor lock-in.

Critiques from academic viewpoints also question the nature of multi-engine synergy. A data lake, in principle, fosters the usage of Spark, Presto, Flink, or even HPC frameworks for specialized computations. By centering all data in Snowflake, organizations might reduce their ability to adopt new analytics engines or hamper the advanced data processing tasks that are typical in cutting-edge AI research. A truly open data lake environment fosters more flexibility in these regards.

XIV. CONCLUSION

Snowflake is a powerful, cloud-native data warehouse solution that has integrated many features commonly associated with data lakes, including support for semi-structured data, near real-time ingestion, and consumption-based pricing. However, it cannot be considered a pure data lake because it does not store data in open formats accessible by multiple engines without overhead. Snowflake's proprietary data structure, while enabling advanced performance, can hamper cost optimization and architectural flexibility.

A more practical question to ask than "Is Snowflake a data lake?" is "How do we combine Snowflake with a data lake?" Indeed, the majority of organizations that rely on Snowflake have discovered that the optimal architecture is to keep a large portion of raw or historical data in an inexpensive object storage system, thus forming the data lake core, and use Snowflake strategically for high-value analytics, dashboards, or frequent queries that demands concurrency and speed.

The reasons behind this approach revolve around cost control, open data interoperability, and advanced analytics demands. When streaming data is ingested directly into Snowflake, cost can spiral if the warehouse runs at all hours. By storing raw data in a lake and selectively pushing curated subsets to Snowflake, a business can harness

ISSN: 2583-6129
DOI: 10.55041/ISJEM02194

the best of each platform. Data scientists who require advanced machine learning frameworks can operate directly on the data lake, while business users can rely on Snowflake for stable, secure, and speedy analytics.

XV. REFERENCES

[1] R. Hai, C. Koutras, Christoph Quix, and Matthias Jarke, “Data Lakes: A Survey of Functions and Systems,” IEEE Transactions on Knowledge and Data Engineering, vol. 35, no. 12, pp. 12571–12590, Apr. 2023.

[2] Seng, K.P.; Ang, L.; Liew, A.W.-C.; Gao, J. Multimodal Information Processing and Big Data Analytics in a Digital World. In Multimodal Analytics for Next-Generation Big Data Technologies and Applications; Springer: Cham, Switzerland, pp. 3–9, 2019;

[3] Shah RMukherjee KTyagi AKarnam SJoshi DBhosale SMitra S(2023)R2D2: Reducing Redundancy and Duplication in Data LakesProceedings of the ACM on Management of Data10.1145/36267621:4(1-25)Online publication date: 12-Dec-2023

[4] F. Bell, Raj Chirumamilla, B. B. Joshi, B. Lindstrom, R. Soni, and Sameer Videkar, “Data Sharing, Data Exchanges, and the Snowflake Data Marketplace,” Apress eBooks, pp. 299–328, Dec. 2021.

[5] A. Alserafi, A. Abelló, O. Romero and T. Calders, "Keeping the data lake in form: DS-KNN datasets categorization using proximity mining", Proc. Int. Conf. Model Data Eng., pp. 35-49, 2019.

[6] R. Kashyap, “Data Sharing, Disaster Management, and Security Capabilities of Snowflake a Cloud Datawarehouse,” Ijcttjournal.org, 2023.

[7] Ajay Rajadnye, “Datawarehouse Versus Datalake,” SSRN Electronic Journal, 2019

[8] R. Soni, “Snowflake Architecture and Overview,” Apress eBooks, pp. 17–30, Jan. 2023.

[9] K. R. Gade, “Data Lakehouses: Combining the Best of Data Lakes and Data Warehouses,” Journal of Computational Innovation, vol. 2, no. 1, 2022.

[10] P. Wieder and H. Nolte, “Toward data lakes as central building blocks for data management and analysis,” Frontiers in Big Data, vol. 5, Aug. 2022.