ADScAI
Applied Data Science & Artificial Intelligence Symposium

Analysis of Performance, Integration, and Scalability Limitations of Data Virtualization Layers for Big Data Processing in Emerging Use Cases

1st Himani Perera
Software Engineering Teaching Unit
University of Kelaniya
Kelaniya, Sri Lanka
pereradinithi99@gmail.com

2nd Isuru Hewapathirana
Software Engineering Teaching Unit
University of Kelaniya, Sri Lanka
Kelaniya, Sri Lanka
ihewapathirana@kln.ac.lk

Keywords—Data Virtualization, Federated Queries, Data Integration, Multi-Source Data Aggregation, Performance Optimization

I. Introduction

In today's data-driven world, data virtualization has emerged as a transformative technology that addresses these needs by providing a unified, abstract view of data from heterogeneous sources. Unlike traditional data integration methods, Data virtualization eliminates the need for physical data movement and replication, enabling seamless, real-time data access while reducing complexity and enhancing agility. These features make Data virtualization a cornerstone of modern data management systems, particularly in applications like business intelligence, data science, big data analytics and cloud computing [1].

However, despite its potential, data virtualization faces significant performance, scalability, and integration challenges when applied to complex, different varieties of workloads and application scenarios. These limitations hinder its adoption in scenarios involving high volumes of data and federated queries across diverse systems. This research investigates these critical challenges to bridge the gap between Data virtualization's potential and its practical implementation. Additionally, it explores how Data virtualization can be optimized for emerging low-resource applications, such as personal IoT data management or small-scale analytics, extending its utility beyond enterprise environments.

II. Literature Review

Data Virtualization (DV) has gained traction as a modern approach to integrating heterogeneous data sources in real time. Unlike traditional Extract, Transform, Load (ETL) processes, which require physical data movement, DV enables on-demand data access, improving agility and reducing storage costs [1]. However, existing research highlights several challenges in DV implementation.

A. Challenges and Limitations in Data Virtualization

Data virtualization technologies introduce performance and scalability challenges, especially when handling large-scale datasets and federated queries. Studies indicate that real-time querying without intermediate storage results in significant latency, particularly in distributed environments [2]. Unlike Data Warehouses (DW), which leverage indexing and pre-aggregated storage, DV layers rely on on-demand query execution, leading to increased overhead.

Federated queries across multiple sources further exacerbate inefficiencies. Research on tools like Trino and Presto reveals bottlenecks related to limited caching mechanisms and weak MPP (Massively Parallel Processing) engine integration [3]. Additionally, DV struggles with real-time analytics, lacking mechanisms for Change Data Capture (CDC) and continuous data updates [4].

The integration of heterogeneous data sources remains a critical challenge for DV layers. Studies highlight schema incompatibility, proprietary query languages, and a lack of standardization as key barriers to seamless data integration. Moreover, integrating structured, semi-structured, and unstructured data formats requires extensive customization, reducing the efficiency of DV tools. Recent studies have also pointed to challenges in DV's compatibility with NoSQL databases and alternative data integration techniques, as schema flexibility and query translation mechanisms remain underdeveloped [5].

Cloud integration presents additional difficulties. While modern cloud platforms support distributed data storage, DV solutions exhibit limitations in managing hybrid environments due to network latencies and inadequate optimization techniques [6]. Furthermore, DV's reliance on distributed execution models introduces additional complexity when integrating cloud-native NoSQL systems, where eventual consistency and schema evolution pose further challenges [7].

B. Identified Research Gaps

Existing research has predominantly focused on enterprise-level implementations of DV. However, limited studies explore its utility for small-scale individual use cases, such as IoT data aggregation and personal data management. Additionally, most performance evaluations have been conducted using SQL-based systems like PostgreSQL and MySQL, leaving DV's compatibility with NoSQL and distributed file systems underexplored [7].

III. Methodology

This research involved a mixed-method approach to thoroughly analyze the limitations of data virtualization (DV)

https://doi.org/10.31705/ADScAI.2025.24

layers while demonstrating their utility for small-scale, multi-source data scenarios.

Federated Query Performance Experiment (Quantitative Component): Evaluated the performance of federated queries executed on Trino by measuring query execution time, CPU usage, and memory utilization across PostgreSQL and ClickHouse.

Smaller-Scale IoT Data Integration Case Study: A case study was conducted to evaluate the practical utility of DV layers in individual, small-scale scenarios under limited hardware resources.

IV. Experiment

To evaluate the performance of data virtualization (DV) layers, a series of experiments were conducted using Trino as the DV layer and TPC-H SF100 as the benchmark dataset. The experiments compared two scenarios:

Unified Data Source (PostgreSQL): The entire dataset (1992–1998) was stored in a single PostgreSQL database to establish baseline performance metrics.

Federated Data Source (PostgreSQL + ClickHouse): The dataset was partitioned by date range—data from 1992–1994 in PostgreSQL and 1995–1998 in ClickHouse. Trino executed federated queries across both sources through a logical view created using Trino's memory connector.

The same standard 22 TPC-H benchmark queries were executed in both phases.

Fig. 1 and Fig. 2 present the performance comparisons between the two configurations.

Fig. 1. Comparison of Query Execution Times

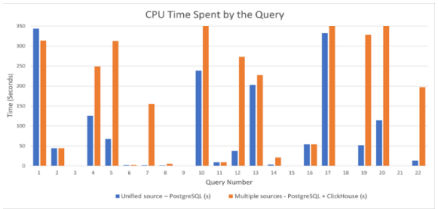

Fig. 2. Comparison of CPU Times: Unified vs. Federated Sources

Experiments showed that federated queries incurred significant latency compared to single-source queries due to the overhead of fetching and processing data from multiple sources. Specifically, the mean CPU time for queries executed on the Unified source – PostgreSQL was 102.44 seconds, while the median CPU time was 37.53 seconds. In contrast, when executing queries across Multiple sources - PostgreSQL + ClickHouse, the mean CPU time increased to 235.94 seconds, and the median CPU time was 249 seconds.

These results highlight the performance penalty introduced by federated queries, where the complexity of handling data from multiple sources leads to increased query times.

Additionally, the average peak total memory for queries executed with the Unified source – PostgreSQL was 1364.65 MB, with a median peak total memory of 300 MB. This is in stark contrast to the Multiple sources – PostgreSQL + ClickHouse setup, where the average peak total memory was 3883.57 MB, and the median peak total memory was much higher at 5335.04 MB. These figures further underline the scalability limitations, as handling multiple sources simultaneously demands significantly more memory resources.

V. Case Study

The case study showcased the effectiveness of DV layers for small-scale, multi-source data aggregation, such as in smart homes or small farms. Using Trino on low-spec hardware (Intel Core i7, 16 GB RAM), it integrated PostgreSQL, ClickHouse, and MongoDB to efficiently query and aggregate data across sources. As shown in Fig. 3, all queries executed within milliseconds, confirming that DV layers like Trino can deliver cost-effective, high-performance data integration even on limited resources. While not ideal for big data, the results demonstrate their strong potential for small-scale, individual use cases.

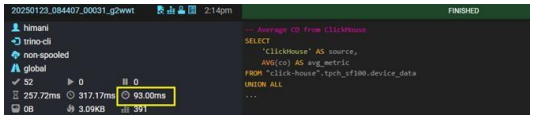

Fig. 3. Aggregation query output

VI. Conclusion

This research explored the performance, integration, and scalability challenges of data virtualization (DV) layers in big data processing. The findings highlighted significant performance bottlenecks, particularly in federated query scenarios with large datasets. However, the study also demonstrated that DV layers like Trino are effective for small-scale, multi-source data integration, offering seamless integration and low latency, even on resource-constrained hardware. The research contributes valuable insights to the broader field of big data processing and offers practical recommendations to improve DV layer performance, making data integration more accessible for smaller-scale applications.

References

[1] R. Van Der Lans, Data Virtualization for Business Intelligence Systems: Revolutionizing Data Integration for Data Warehouses, vol. 1. 2012.Integration for Data Warehouses, vol. 1. 2012.

[2] Trino Documentation, "Trino Query Engine: Architecture and Performance," 2024.

[3] Bologa, R. & Bologa, A., "Data Virtualization: A Modern Approach for Big Data Integration," Journal of Information Systems, 2023.

[4] Balakrishnan, K., "Performance Bottlenecks in Data Virtualization," IEEE Transactions on Data Engineering, 2024.

[5] R. Sethi et al., "Presto: SQL on Everything," 2019 IEEE 35th International Conference on Data Engineering (ICDE), Apr. 2019, doi: https://doi.org/10.1109/icde.2019.00196.

[6] Van Der Lans, R., "Data Virtualization: Integration Challenges in Cloud Environments," Springer, 2022.

[7] Lawrence, J., "Performance Evaluation of Data Virtualization Systems: A Case Study on MySQL and PostgreSQL," IEEE, 2018.