Large-Scale Health Data Pipelines with a Lakehouse Architecture: A SISVAN Case Study

Italo V. P. Guimaraes[1], **Marcelo Ladeira**[1], **Aleteia Araujo**[1]

[1]Computer Science Department (CIC), University of Brasília (UnB)
Brasília – DF – Brazil

italo.guimaraes@aluno.unb.br, mladeira@unb.br, aleteia@unb.br

Abstract. *National health surveillance systems such as Brazil's SISVAN accumulate tens of millions of records yet still rely on rigid data-warehouse pipelines ill-suited to heterogeneous, evolving data. This paper shows that a Lakehouse architecture — Delta Lake over MinIO, orchestrated by Apache Airflow and queried via Trino — processes 82 GB of SISVAN microdata through four medallion layers in under 100 minutes and returns Gold-layer analytical queries in under 2 seconds. A Superset dashboard co-designed with Ministry of Health analysts translates the curated data into actionable public-health indicators. The results validate that modern Lakehouse patterns deliver ACID guarantees, full data lineage, and sub-second BI latency on commodity hardware for national-scale health data governance.*

1. Introduction

Public health information systems are generating data at an unprecedented scale and complexity, posing new challenges for data management and analytics. Brazil's Food and Nutrition Surveillance System (SISVAN) is a prime example, collecting nationwide heterogeneous health data (e.g., anthropometric measurements, dietary intake records) from diverse sources over time. Traditional data warehouse architectures, while excellent for structured reporting, struggle to accommodate the variety of unstructured and semi-structured data prevalent in modern health surveillance (images, free-text, sensor readings, etc.). Data warehouses require upfront schema design ("schema-on-write") and cannot easily manage the *volume and diversity* of raw health data [Harby and Zulkernine 2022]. On the other hand, data lakes embrace raw data storage of any type ("schema-on-read"), but often lack the robust governance and data quality controls of warehouses-risking the emergence of disorganized "data swamps" if metadata management and governance are inadequate. In such uncontrolled lakes, the provenance, semantics, and quality of ingested health data may remain unknown, undermining their analytical utility [Hai et al. 2023].

These limitations motivate a new paradigm known as the **data lakehouse**, which seeks to *marry* the strengths of data warehouses and data lakes into a unified architecture. A lakehouse architecture provides the management flexibilities of a data lake for *disparate, large-scale datasets* while also enforcing the structured, reliable data engineering practices of a warehouse [Begoli et al. 2021]. In practice, this means supporting diverse data types and sources *with better governance, data cleansing, and ACID compliance* on the storage layer, so that data remains both highly scalable and analytics-ready. Notably, emerging lakehouse implementations add an **ACID transaction** layer over low-cost cloud

object storage, enabling reliable updates, deletes, and consistency guarantees that traditional data lakes lacked [Harby and Zulkernine 2022]. This ensures *improved data quality and integrity* for critical health datasets by preventing partial writes and preserving versioned histories of data changes. At the same time, lakehouses leverage open standards (e.g., columnar storage formats and SQL interfaces) to achieve high-performance querying comparable to data warehouses, thus making the data immediately available for epidemiological analysis, machine learning, and policy decision support.

Research gap and contribution. Despite growing interest in lakehouse architectures, no published study documents an end-to-end deployment on a Brazilian national-scale public-health surveillance system, nor reports the design trade-offs of co-designing the analytical layer with Ministry of Health practitioners. This paper addresses that gap with three concrete contributions: (i) a reference Lakehouse pipeline for SISVAN built entirely on open-source components (Airflow, Spark, Delta Lake, MinIO, Trino, Superset) and reproducible from a public repository; (ii) an empirical characterization of medallion-layer storage and end-to-end performance on 82 GB of national microdata, showing sub-2-second analytical latency on commodity hardware; (iii) a dashboard co-designed with Ministry of Health analysts, demonstrating that the curated layer translates directly into actionable nutritional-surveillance indicators.

2. Related Work

Data warehouses provide structured storage and fast SQL but require fixed schemas, while data lakes embrace raw heterogeneous data at the cost of governance, degrading into "data swamps" without rigorous metadata management [Harby and Zulkernine 2022, Hai et al. 2023]. The lakehouse paradigm reconciles these extremes by combining warehouse reliability with lake flexibility [Harby and Zulkernine 2022]; ACID-compliant table formats such as Delta Lake provide the transactionality and time-travel needed to govern mutable data on object stores [Armbrust et al. 2020], and open columnar formats with SQL interfaces deliver performance comparable to warehouses. Zone or medallion reference models structure multi-stage pipelines and clarify the separation between raw, curated, and served data [Giebler et al. 2020].

Healthcare-specific lakehouses have been explored at the mega-biobank [Begoli et al. 2021] and FHIR ingestion [Patil and Belloum 2024] levels; HEALER targets clinical and research multi-tenancy with patient privacy and fine-grained access controls [Manco et al. 2023]; Zhao *et al.* report query-latency and storage efficiency gains for multi-modal medical data [Zhao et al. 2023]. Surveys trace the DW→DL→lakehouse evolution [Li et al. 2023, Janssen et al. 2024]. None of these works documents a deployment on a Brazilian national public-health surveillance system, nor reports the co-design of the analytical layer with Ministry of Health practitioners - the gap this paper addresses.

3. System Architecture

Figure 1 shows the end-to-end Lakehouse pipeline implemented for SISVAN. The diagram illustrates:

- **Orchestration:** Apache Airflow defines and schedules four DAGs (Landing, Bronze, Silver, Gold) that invoke PySpark jobs and manage dependencies.

- **Storage & Processing:** Raw and processed data are stored as Delta Lake tables on MinIO; Apache Spark performs the ETL transformations at each medallion layer.
- **Metadata Catalog:** A Hive Metastore (backed by MariaDB) holds table schemas for all layers, enabling Trino to discover and query the data.
- **Query & Visualization:** Trino provides a high-performance SQL interface over the Lakehouse; Apache Superset consumes Trino to render interactive dashboards on the Gold-layer outputs.

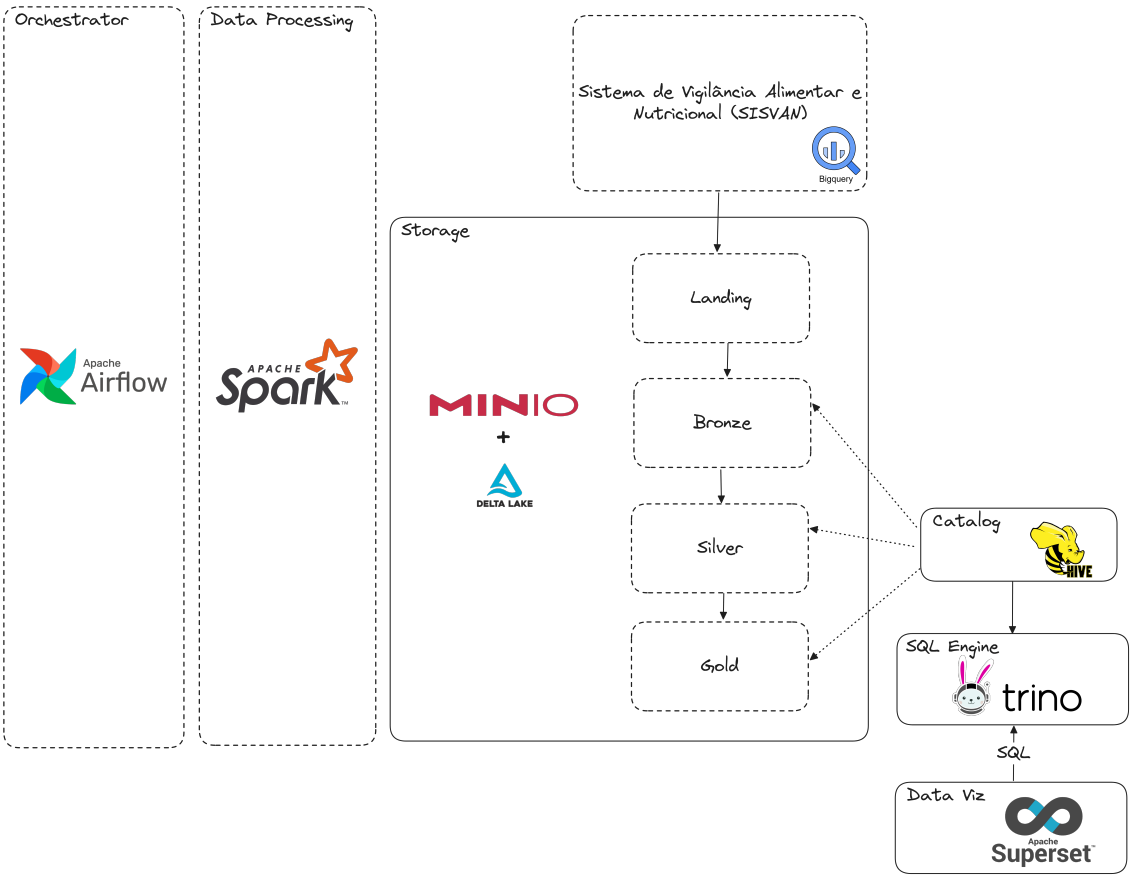

Figure 1. SISVAN Lakehouse Pipeline Architecture.

4. Methodology

The pipeline is implemented as four Airflow DAGs that materialize the medallion layers as Delta Lake tables on MinIO. DAG 1 (*bigquery_to_landing*) uses the Spark BigQuery Connector to extract the pre-cleaned SISVAN microdata (*basedosdados.br_ms_sisvan.microdados*) into the *Landing* table, partitioned by *year*, *month*, and *sigla_uf*. DAG 2 (*landing_to_bronze*) writes the raw snapshot to the *Bronze* table with idempotent ACID writes and minimal type parsing. DAG 3 (*bronze_to_silver*) applies business rules: null/outlier filtering, code-value translation (race, life stage, education), and unit standardization, leveraging Delta Lake's automatic schema evolution. DAG 4 (*silver_to_gold*) computes SISVAN indicators (malnutrition rates by region, monthly follow-ups, unique individuals per year) into separate *Gold* tables optimized through partitioning and Z-ordering.

After each write the Spark jobs register the tables in a Hive Metastore (MariaDB-backed); bootstrap scripts populate schemas at startup, so Trino and Superset discover them immediately and enforce access control through the unified catalog. The entire stack (Airflow, Spark, MinIO, Trino, Superset, Hive Metastore) runs as Docker containers orchestrated by Docker Compose and the Astronomer CLI, and scales horizontally by adjusting service replicas without code changes.

5. Results

The SISVAN Lakehouse pipeline was executed on a modest two-worker Spark cluster (6 vCPU and 1 GiB RAM per worker). Trino was tuned with a 9 GiB JVM heap, `query.max-memory`=8 GiB and 4 GiB per node. Table 1 summarises the wall-clock time of each Airflow DAG.

Table 1. End-to-end pipeline performance.

Pipeline stage	Duration
bigquery_to_landing	38 min 27 s
landing_to_bronze	29 min
bronze_to_silver	19 min
silver_to_gold	13 min

5.1. Storage, Volume, and Query Workload

Table 2 reports per-layer storage on Delta Lake/MinIO and approximate row counts ($\sim$230 M follow-ups; 2023 peak of $>$50 M unique individuals). Columnar Parquet plus Delta compression keeps Landing/Bronze/Silver within $\sim$34 GiB each, while aggregation collapses Gold to under 1 MiB. A full scan of every Gold table via Trino took 1.747 s and a complete read of `silver.sisvan` finished in 2 s. Representative Gold-layer SQL queries (e.g., obesity-rate-by-state for the dashboard's choropleth) are available in the public repository (Sect. 8).

Table 2. Medallion-layer storage and approximate row counts.

Layer	Storage	Approx. rows
landing.sisvan	34.5 GiB	$\sim$230 M
bronze.sisvan	34.3 GiB	$\sim$230 M
silver.sisvan	34.3 GiB	$\sim$230 M (cleaned)
gold.indicadores_chave	505.9 KiB	$<$10 K (aggregated)

6. Interactive Dashboard

To ensure the lakehouse delivers value beyond batch analytics, an Apache Superset dashboard was *co-designed with analysts from the Brazilian Ministry of Health*, whose domain expertise guided the choice of metrics and visual encodings (Figure 2). Six panels combine *burden* (follow-ups per state, monthly follow-ups by source system), *coverage* (unique users per year, revealing a $\times 4$ growth since 2012 and a 2023 peak above 50 M individuals), and *outcomes* (obesity rate by state, nutritional status by sex, top status counts per state), letting decision-makers prioritize interventions by geography, period, and data-capture system.

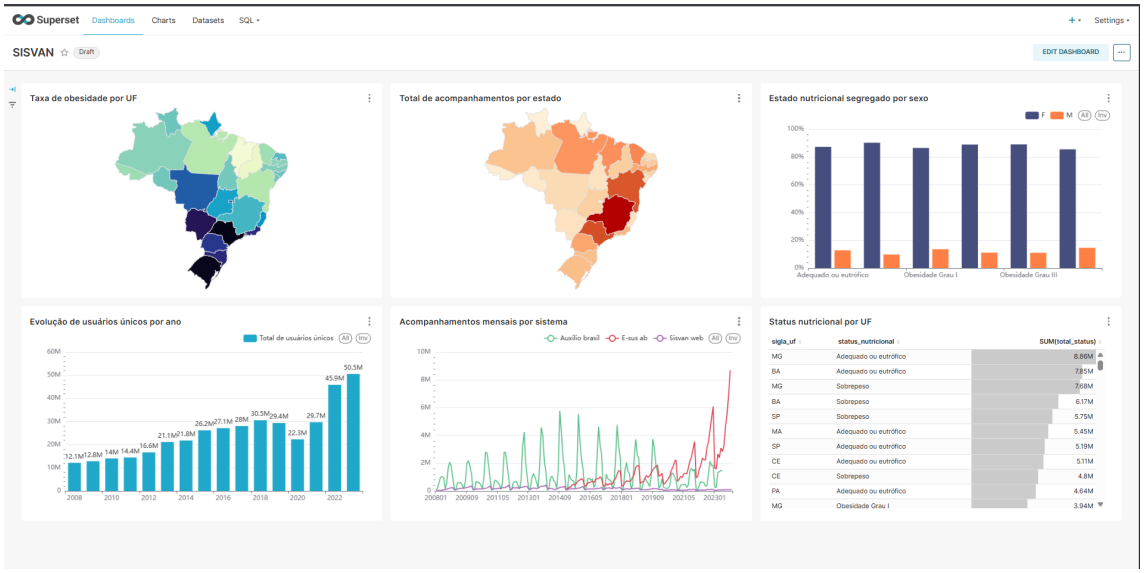

Figure 2. Superset dashboard generated from `gold` views: (a) obesity rate by state; (b) follow-ups per state; (c) nutritional status by sex; (d) unique users over time; (e) monthly follow-ups by source system; (f) top nutritional-status counts by state.

7. Discussion

Reliability and governance. Delta Lake's ACID semantics and transaction log over MinIO guarantee consistent snapshots and point-in-time recovery - indispensable for regulated health data - while a single Hive Metastore gives Spark, Trino, and Superset a unified governed catalog for lineage and access-control policies.

LGPD and security posture. Under Brazil's LGPD, a production-grade SISVAN deployment requires explicit data-protection controls. The proposed stack supports this by design: MinIO serves over HTTPS with server-side AES-256 encryption; Trino and Superset inherit single-sign-on via the ministry's LDAP/OIDC provider; and row-level filters on `sigla_uf` restrict regional access. Sensitive direct identifiers are absent from the published microdata [Harby and Zulkernine 2022]; pseudonymization at the Bronze boundary would extend the same guarantees to derived datasets.

Architectural trade-offs. The pipeline spends roughly 100 minutes to refresh every layer, in exchange for: (1) strict separation of raw, conformed, and analytic data; (2) ACID guarantees and versioned history on commodity object storage; and (3) sub-2-second BI latency on Gold views. Production hardening would integrate a rule engine such as *Great Expectations* after the Bronze write (e.g., biological plausibility on weight, height, BMI), emit OpenLineage events from Airflow for observability, and schedule nightly `OPTIMIZE ZORDER` / `VACUUM` jobs to preserve query latency while respecting the Delta time-travel window required for audit.

8. Code and Reproducibility

All artefacts required to reproduce the pipeline-Docker Compose orchestration, Airflow DAGs, PySpark transformation scripts, Trino/Hive catalog settings, Superset dashboard exports, and example queries-are publicly available at `https://github.com/`

`italovinicius18/sisvan-lakehouse-pipeline`.

9. Conclusion

The SISVAN Lakehouse validates that modern medallion patterns built on open-source components can operate on production-scale Brazilian government health data, delivering trustworthy ingestion, governed curation, and sub-2-second analytics on commodity hardware. Unlike conventional data-warehouse or pure data-lake solutions, the design unifies ACID transactions, open columnar storage, and a multi-zone model in a single Delta Lake repository accessed concurrently by Spark (batch) and Trino (interactive SQL)-an arrangement still uncommon in public-sector health systems. Future work covers (i) incremental stream ingestion for near real-time updates, (ii) automated data-quality alerts (Great Expectations) across layers, and (iii) federation with additional health repositories to enrich epidemiological insights.

References

Armbrust, M., Das, T., Sun, L., Yavuz, B., Zhu, S., Murthy, M., Torres, J., van Hovell, H., Ionescu, A., et al. (2020). Delta lake: high-performance acid table storage over cloud object stores. *Proc. VLDB Endow.*, 13(12):3411–3424.

Begoli, E., Goethert, I., and Knight, K. (2021). A lakehouse architecture for the management and analysis of heterogeneous data for biomedical research and mega-biobanks. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 4643–4651.

Giebler, C., Gröger, C., Hoos, E., Schwarz, H., and Mitschang, B. (2020). A zone reference model for enterprise-grade data lake management. In *2020 IEEE 24th International Enterprise Distributed Object Computing Conference (EDOC)*, pages 57–66.

Hai, R., Koutras, C., Quix, C., and Jarke, M. (2023). Data lakes: A survey of functions and systems. *IEEE Transactions on Knowledge and Data Engineering*, 35(12):12571–12590.

Harby, A. A. and Zulkernine, F. (2022). From data warehouse to lakehouse: A comparative review. In *2022 IEEE International Conference on Big Data (Big Data)*, pages 389–395.

Janssen, N., Ilayperuma, T., Jayasinghe, J., Bukhsh, F., and Daneva, M. (2024). The evolution of data storage architectures: examining the secure value of the data lakehouse. *Journal of Data, Information and Management*, 6(4):309–334.

Li, G., Hu, W., and You, T. (2023). Data lake development status and outlook. In *Third International Conference on Green Communication, Network, and Internet of Things (CNIoT 2023)*, volume 12814, page 128142G. SPIE.

Manco, C., Dolci, T., Azzalini, F., Barbierato, E., Gribaudo, M., and Tanca, L. (2023). *HEALER: A Data Lake Architecture for Healthcare*. DEU.

Patil, S. and Belloum, A. S. Z. A. (2024). Processing fhir in modern data lakehouse. In *2024 IEEE 20th International Conference on e-Science (e-Science)*, pages 1–7.

Zhao, T., Hai, N., Li, W., Zheng, W., Zhang, Y., Li, X., and Fei, G. (2023). Multi-modal medical data exploration based on data lake. In *Health Information Science*, pages 213–222. Springer Nature.