

Real-Time Intelligence: Accelerating Insights with RisingWave

Architectural Strategy & E-Commerce
Conversion Demo

Inter Savas Gioldasis | Principal Data Engineer



Moving from Retrospective to Reactive

Traditional batch analytics are too slow for modern conversion optimization. We are validating a stack that delivers immediate insights without the Big Data overhead.

The Solution



JetBrains Mono

RisingWave – A cloud-native streaming database compatible with PostgreSQL.

The Advantage



JetBrains Mono

High throughput with significantly lower TCO compared to legacy stacks like Flink.

The Proof



JetBrains Mono

A live E-Commerce Funnel demo showing real-time View-to-Cart and Cart-to-Buy conversion rates.



RisingWave: Streaming Made as Simple as SQL

Traditional Database



JetBrains Mono

Static Data. Batch Queries.

RisingWave



JetBrains Mono

Dynamic Streams. Continuous Queries.

- **Definition:** A cloud-native streaming database that uses SQL as the primary interface.

- **The Paradigm:** Treats data streams as tables. No complex Java/Scala; just standard SQL queries that update automatically.

- **Capabilities:** Powering real-time dashboards, ad-hoc queries, and alert systems immediately upon ingestion.

Strategic Comparison: RisingWave vs. Apache Flink

Apache Flink (The Incumbent)

JetBrains Mono



- **Pros:** Mature ecosystem, massive scale.
- **Cons:** High operational complexity, heavy infrastructure footprint, requires specialized Java/Scala talent.

RisingWave (The Agile Challenger)

JetBrains Mono



- **Pros:** PostgreSQL Compatible (leverages existing SQL talent), Cloud-native architecture (separation of compute/storage), significantly lower TCO.
- **Cons:** Newer ecosystem.

Verdict: RisingWave offers faster time-to-market for agile real-time applications.

The Business Case: E-Commerce Conversion Funnel

Viewers

Unique page views
(src_page.sql)

Carters

Users adding items to cart
(src_cart.sql)

Purchasers

Completed transactions
(src_purchase.sql)

The Goal:
Instantly calculate
`view_to_cart_rate` and
`cart_to_buy_rate` to
identify drop-offs in
real-time.

Live Visualization: The Modern React Dashboard

Viewer Count:
12,450

Cart Rate:
45%

Purchase Rate:
12%

Producer Controls

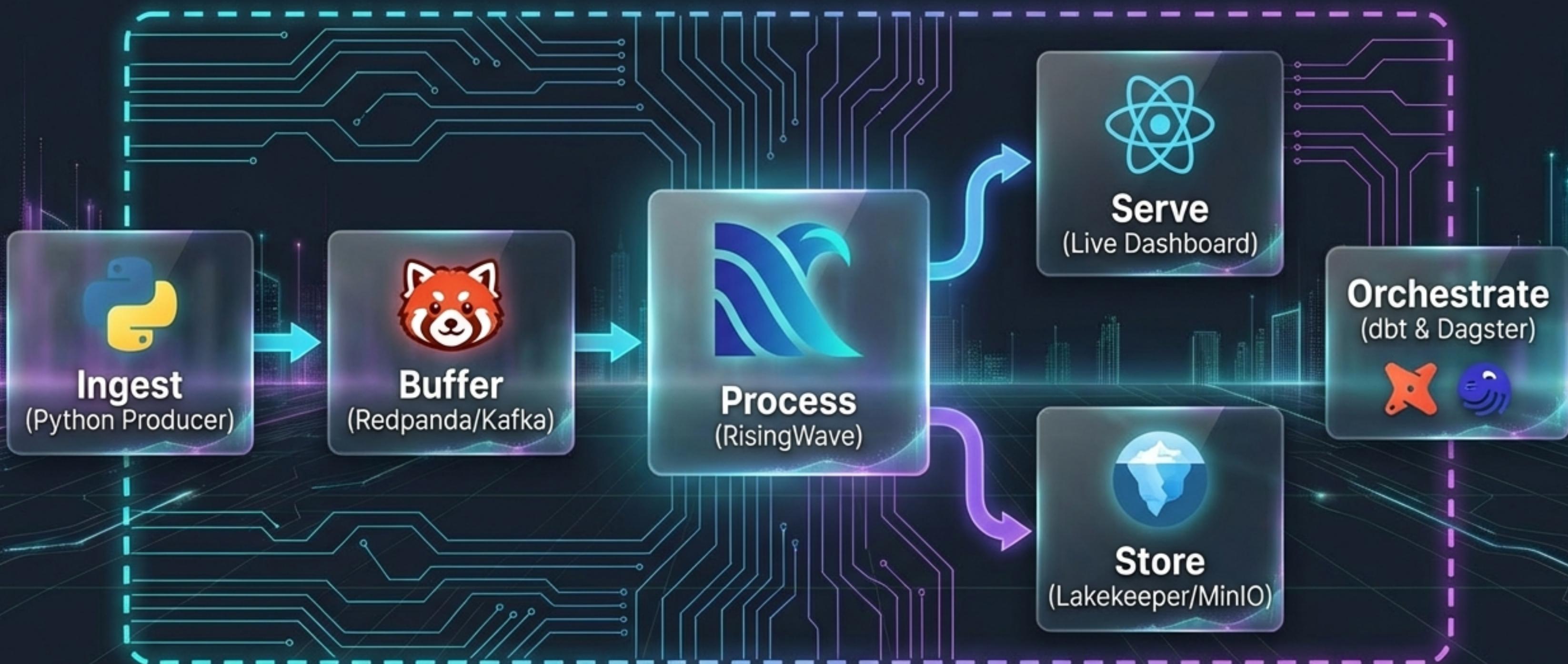
▶ Start

■ Stop

Real-time Updates:
Millisecond-level latency.

Tech Stack:
React Frontend + FastAPI
Backend + RisingWave.

High-Level Architecture

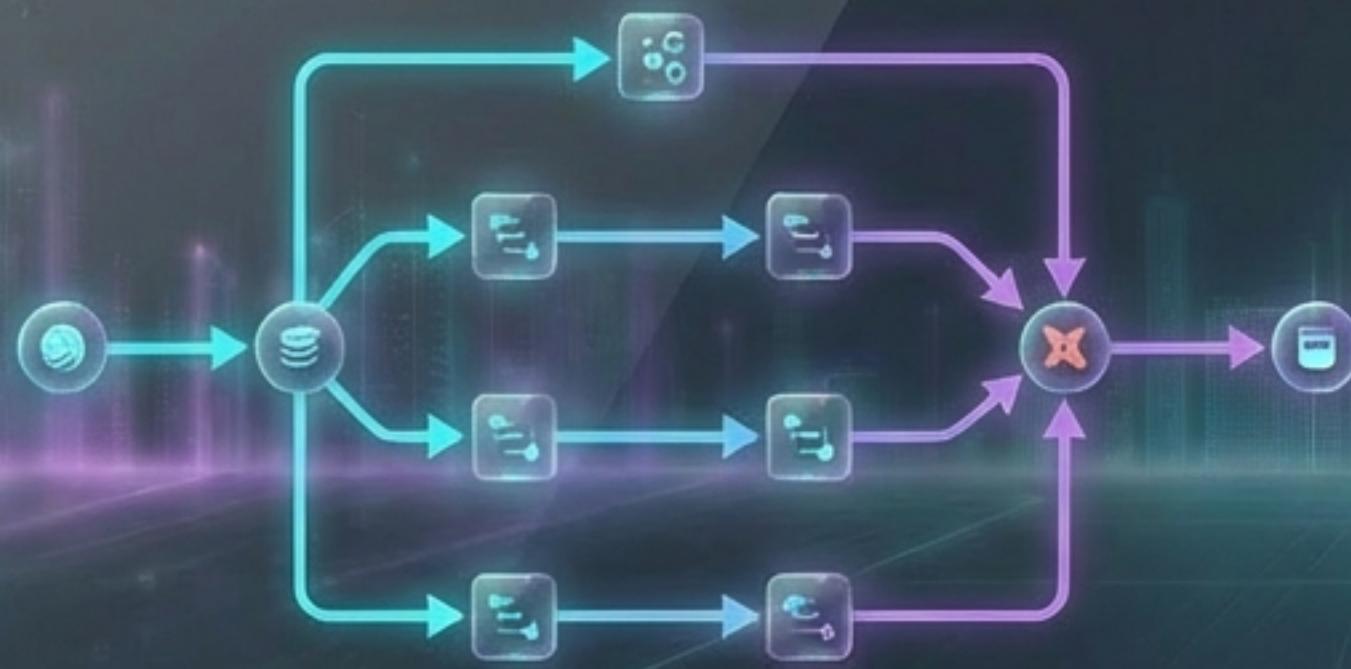


The Data Journey: From Event to Materialized View



Orchestration & Quality Assurance

Orchestration (Dagster)



Manages pipeline lifecycle
and ensures reliability.

Transformation (dbt)

```
SELECT *  
FROM {{ source('risingwave', 'page_views') }}
```

Applies version control and
testing to streaming logic.

The Persistence Layer: Apache Iceberg & Lakekeeper

Real-Time
Serving

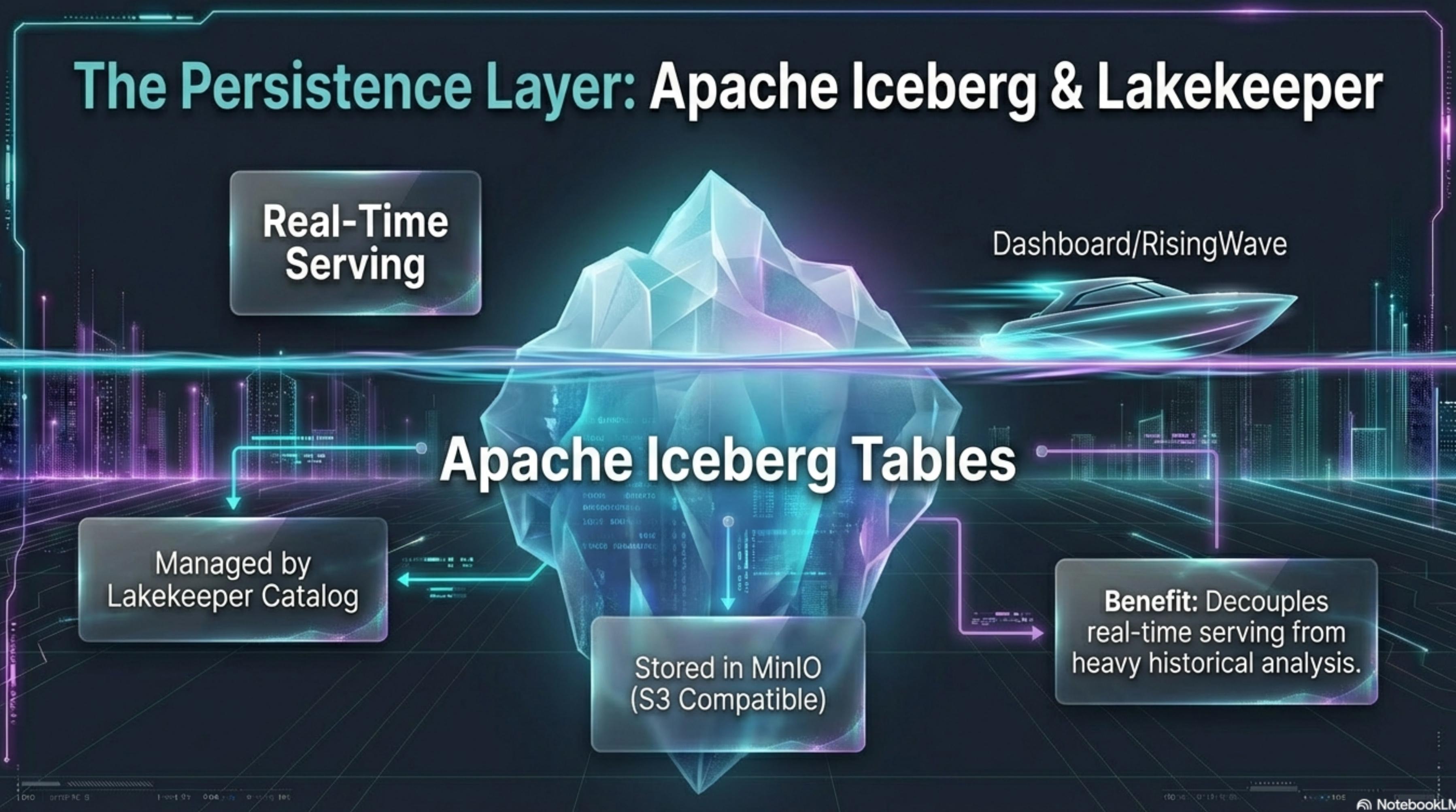
Dashboard/RisingWave

Managed by
Lakekeeper Catalog

Stored in MinIO
(S3 Compatible)

Benefit: Decouples
real-time serving from
heavy historical analysis.

Apache Iceberg Tables



Advanced Analytics: DuckDB & Spark Integration



DuckDB: Ad-hoc queries on Iceberg tables
(scripts/query_raw_iceberg.py)

Spark: Complex user behavior modeling and historical research.

Key Strategic Takeaways



Simplicity: Complex streaming logic implemented using standard SQL and dbt.



Performance: Achieved millisecond latency using Redpanda and RisingWave's compute-storage separation.



Scalability: Production-ready architecture separating speed (RisingWave) from storage (Iceberg/MinIO).

Ready to Scale Real-Time Intelligence

The architecture demonstrated is modular, scalable,
and ready for pilot deployment.



Savas Gioldasis
Principal Data Engineer