Materialize

WHITEPAPER

A Reference Architecture for Real-Time Fraud Detection

According to the FTC, US consumers reported losing $10 billion to fraud in 2023, a 14% increase over 2022. With fraud attacks increasing yearly, companies must deploy real-time fraud detection systems to protect their customers and their assets.

But standard data architectures are not ideal for fraud detection. Traditional batch data architecture delays fraud determinations. And fraud detection needs to occur while fraud is happening. Anti-fraud measures that take hours, or even minutes, allow fraudsters to escape with their loot.

Most companies capture the data they need to detect fraud, including user behavior and account activity, in company databases. However, the challenge is transforming this data with split-second rapidity.

A traditional data warehouse, with SQL support and the ability to ingest diverse data sources, seems like a potential platform to power anti-fraud services.

But traditional data warehouses are designed around batch-loading and caching. They are optimized for analytics and historical reporting. Fraud detection requires the continuous transformation of real-time data, a task that is expensive and difficult for traditional data warehouses.

With the rise of operational data warehouses, companies can now build cost-effective, real-time fraud detection systems. Operational data warehouses combine streaming data, SQL support, and continuous data transformation to calculate fraud scores in real-time, stopping fraudsters in their tracks.

Materialize is an operational data warehouse that fuels real-time fraud detection systems for many of our customers in the financial services sector.

After working with several leading data teams to build these streaming anti-fraud systems from scratch, we want to share what we've learned about reference architectures for real-time fraud detection with you.

In this white paper, we'll detail best-practices for building fraud detection reference architectures. We'll also outline a real fraud detection use case from one of our customers, Ramp.

Fraud Detection: Accuracy vs. Latency

Effective fraud detection depends on two critical factors: accuracy and latency. Fraud detection workflows must predict fraud accurately in order to stop bad actors, without disrupting real customers. And fraud detection must achieve low latency in order to detect fraudulent activity in time to stop it.

Accuracy is essential not just to stop fraud, but to avoid disrupting legitimate customers. Both cut into company profit margins. Companies will never detect fraud with absolute accuracy. However, they can assign a well-refined, probabilistic fraud score to each transaction. They can apply automated deterrence actions when the score passes certain thresholds.

Companies can refine fraud score criteria over time, as more fraud data is verified. SQL remains a popular choice for programming fraud scores, due to its refined business logic, and its strength with manipulating data. That's why companies turn to data warehouses to power fraud detection.

Data warehouses can ingest, join, and transform large volumes of data. Teams use data warehouses to amalgamate fraud signals from different sources, including product sources and business systems. They restructure this data via SQL queries into business inputs for fraud workflows.

Typically, companies leverage traditional data warehouses — or 'analytical' data warehouses — to perform fraud detection. And when it comes to accuracy, analytical data warehouses are viable options.

Teams can analyze historical fraud data with analytical data warehouses. They can use this historical data to develop SQL logic that detects fraudulent activity. But because analytical data warehouses harness historical data, they can only detect fraud after it happens, rather than during the act.

In other words, you can use an analytical data warehouse to build SQL logic for detecting fraud. And this SQL logic can accurately identify fraudulent activity. But the data itself is hours or days old.

In terms of actually stopping fraud, analytical data warehouses have limited use. Fraud detection needs to occur within seconds in order to be effective. Otherwise, fraudsters can easily escape with their ill-gotten gains.

Thus, the problem with analytical data warehouses is not one of accuracy, but one of high latency.

The Cost of Latency for Analytical Data Warehouses

This problem of high latency is built into the way analytical data warehouses are designed.

Analytical data warehouses practice batch processing. Data is processed in batches, at set intervals, rather than in real-time. Queries are also run at intervals, perhaps a few times a day at most.

By the time the data is queried, it's out-of-date. The window for acting on the data has closed. For operational use cases such as fraud detection, this delay is unacceptable. Querying batched data every few hours is not sufficient, when the window for stopping fraudsters is measured in seconds.

However, cloud-native data warehouses are still in many ways ideal for the fraud detection use case. The ability to combine large volumes of disparate data sources, and utilize SQL for logic, is an attractive option for data teams. In fact, some teams are willing to push traditional data warehouses to their limits to keep this convenient architecture.

Teams can develop their SQL-powered fraud scores on analytical data warehouses. And by natural extension, they do try to use their analytical data warehouses for real-time fraud detection.

While it's not impossible to implement fraud detection on an analytical data warehouse, it's far from optimal. Analytical warehouses are designed around batch transfer and caching for existing queries.

This option makes sense if your data doesn't change very often. Results are stored in memory after a query and cached as long as possible so it can be re-accessed by a similar query. Since queries are infrequent, the database can maintain consistency with simple table-locking mechanisms.

However, this design is cumbersome for operational workloads such as fraud detection. Computational limits on large batches obstruct data freshness, and cached query results are not helpful when new data is constantly loaded.

This pushes the technical boundaries of analytical data warehouses. As more data is queried, computation times take longer. Anti-fraud workflows slow down, due to these technical limitations. And shaving a few seconds off response time can lead to thousands of dollars in losses.

This option is also much more expensive in terms of compute resources. Rapidly re-running queries demands excessive computation. With an analytical data warehouse, the pricing model is pay-per-query, and cost is linked to data freshness. Costs for operational use cases such as fraud detection, which require continuous query execution, skyrocket for analytical data warehouses.

With these limitations, teams soon realize that while traditional data warehouses can serve as testing grounds for SQL, they cannot operationalize real-time fraud detection. At least, not with the latency that the use case requires. And so, they're left with accurate but out-of-date fraud scores.

But what if teams could combine the ease and power of a data warehouse, along with this elusive low latency?

That would allow teams to engage in effective real-time fraud detection directly from their data warehouse. And this is not a thought experiment: teams are accomplishing this right now with operational data warehouses.

Operational Data Warehouse: Streaming Data + SQL Support + Continuous Transformation

Operational data warehouses combine streaming, real-time data with continuous data transformation to power essential business operations, including fraud detection.

Operational data warehouses leverage streaming data to enable use cases that require low latency. ODWs process data in a continuous, incremental way, so results are updated as they change, as opposed to all at once in a batch job.

To power real-time use cases, operational data warehouses continuously transform streams of raw data into actionable outputs. ODWs allow you to execute SQL queries on fresh data continuously.

This combination of streaming data and continuous transformation make operational data warehouses ideal for fraud detection use cases.

Real-time data ensures that the data is always up-to-date. ODWs receive fraud signals as they occur, so you can act on fraudulent activity in real-time.

Operational data warehouses also empower you to continuously transform this fresh data. You can reformat the data into usable inputs for your anti-fraud workflows every few seconds, rather than minutes, or hours.

While traditional data warehouses can detect fraud hours after it occurs, operational data warehouses combine streaming data and continuous transformation to detect fraud almost instantly. This enables operational data warehouses to operationalize SQL logic for fraud detection in real-time. With these new capabilities, companies can stop fraud as it occurs, rather than identify historical fraud.

The cost of operationalizing fraud detection is high for analytical data warehouses. Constantly re-running anti-fraud queries is expensive in a pay-per-query pricing model. But with operational data warehouses, price is not tied to query execution.

Instead, Materialize avoids constant query recomputation. By maintaining views incrementally, Materialize decouples the cost of compute and query execution. Materialize uses materialized views and indexes to provide up-to-date query outputs at a fraction of the cost.

Instead of re-running the query, Materialize only updates the results that have changed. This ensures the query output is fresh, while keeping costs down considerably. Materialize harnesses Timely Dataflow, a low-latency computation model, to perform efficient and correct incremental computation.

By updating queries rapidly, Materialize allows teams to constantly transform data for fraud workflows, so they can detect fraud in real-time. This enables them to stop fraudulent activity as it happens without the price or technical limitations of traditional data warehouses.

Now that we've identified the generalized components of a fraud detection system, let's take a look at an actual reference architecture from one of our customers.

Our customer Ramp, the spend management company revolutionizing corporate cards, recently built a real-time fraud detection system with Materialize.

We got a firsthand look at their reference architecture — and we wanted to share it with you. Here's how Ramp's fraud detection system works.

Ramp's Use Case: Fraud Detection System for Billion-Dollar Payments Platform

Ramp is a 5-in-1 payments platform that combines corporate cards, expense management, billing, accounting integrations, and real-time reporting in a single interface. The company processes over $13 billion in payments annually, and has a valuation of $5.8 billion.

Ramp's use case with Materialize is centered around their core offering: their corporate credit card. Fraud is a critical threat to any financial card product. A top-notch fraud detection system is a core requirement for any corporate card.

For corporate cards, there are two main types of fraud:

- **Account takeover** is when an attacker accesses the login credentials of an account, usually by phishing. The attacker logs in to the account and purchases goods that are easy to resell.
- **Transaction fraud** is similar to account takeover, but instead of phishing credentials, the attacker steals the card details or the physical card itself and makes fraudulent purchases.

Ramp needed to respond to these attacks very quickly in order to thwart them. Using exploits and hacking tools, attackers can spend thousands of dollars in minutes. While an individual card is limited by its credit limit, there's no upper bound on the potential losses for Ramp across all cards.

Ramp absorbed the cost of all fraudulent activity on their platform. The financial damages fall solely on them, not their customers. Fraud also harms their reputation as a platform. These are some of the key challenges Ramp faced as they built their fraud detection system.

At first, Ramp believed their current tech stack could support a fraud detection system.
So they built the first version on an analytical data warehouse.

Fraud Detection System V1: Analytical Data Warehouse Leads to Cost and Technical Barriers

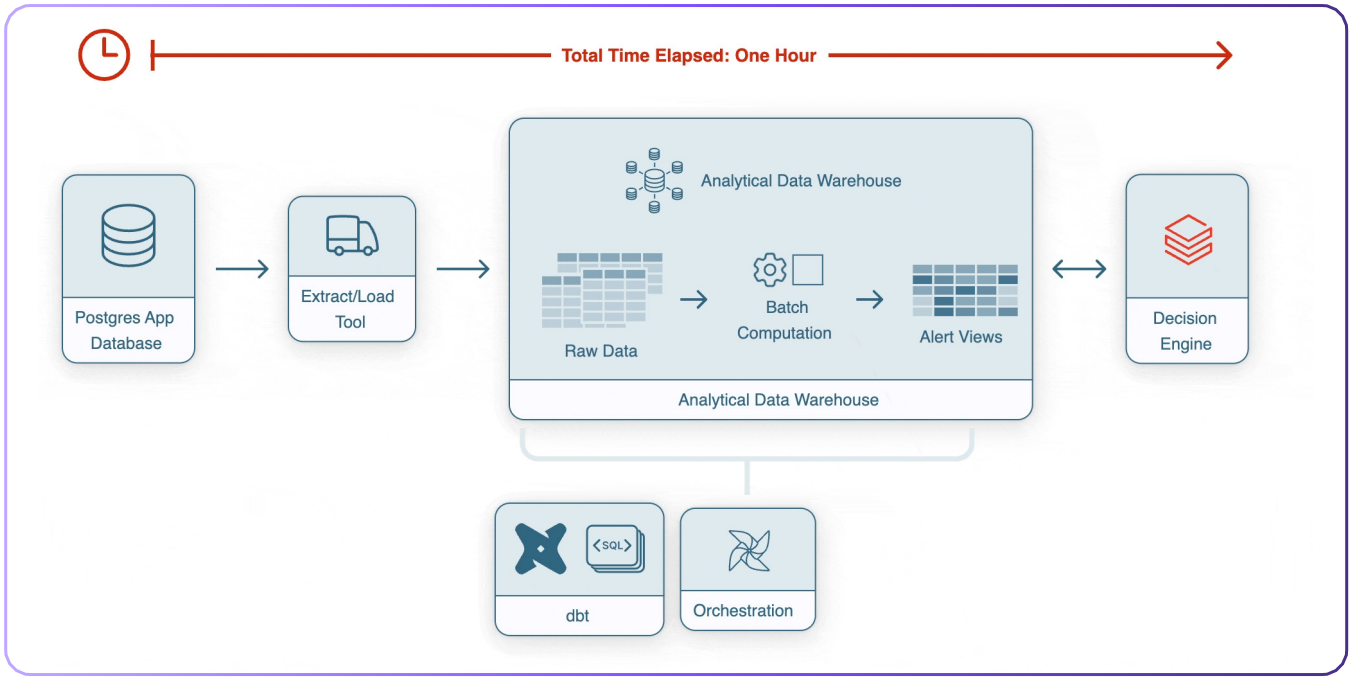
Ramp's development team was a PEDD pod — product, engineering, design, and data — consisting of technical and non-technical users. As a first step, Ramp needed to create a real-time fraud detection system that employees with different technical skill sets could easily work on.

The system needed to use tools and languages that these users were already familiar with.

That's why Ramp built the first version of their fraud detection system on their analytical data warehouse. Everyone was used to using it, and everyone had varying degrees of familiarity with SQL.

In the first version of this architecture, Ramp employed a PostgreSQL database to collect transaction data from its front-end applications and systems. Every hour, a scheduled dbt pipeline loaded a batch of new transaction data from PostgreSQL into the analytical data warehouse.

Once the transaction data was loaded, the analytical data warehouse ran SQL logic to determine if fraudulent activity had occurred. Scheduled notebooks queried the DAG results to produce fraud alerts.



This early version of Ramp's fraud detection system ran passably at first. But as time went on, and data volume scaled, the problems mounted. In Ramp's analytical data warehouse, computation was not continuous, so significant lag developed between scheduled processes.

This gap caused at least an hour of delay between data ingestion and fraud alerts. By the time the fraud alerts arrived, fraudsters already had over an hour to steal their loot and get away unscathed.

Even when Ramp shortened the schedule between ingestion and alerting, the size of the data batches continued to grow, causing runtimes to extend. So fraud alerting still took an hour, even when Ramp tried to make technical adjustments to improve response times. In other words, this was not a real-time fraud detection system.

Computation costs also ballooned. With this early version of the system, Ramp had to run queries very often in order to transform raw fraud signals into actionable inputs. This would normally be fine for a daily use case, such as an analytics workflow that runs once every 24 hours.

But with the fraud detection use case, Ramp saw its compute costs skyrocket, since queries were executed constantly. The result: Ramp ended up with **yearly computation costs exceeding $120,000** just for fraud detection.

With these glaring cost and technical inefficiencies, Ramp needed a new path forward.
And that's how they found Materialize.

Materialize: Operational Data Warehouse Drives Real-Time Fraud Detection for Ramp

> "By moving SQL models for fraud detection from an analytics warehouse to Materialize, Ramp cut lag from hours to seconds, stopped 60% more fraud and reduced the infra costs by 10x."
>
> **RYAN DELGADO, STAFF SOFTWARE ENGINEER, DATA PLATFORM - RAMP**

Before adopting Materialize, the team had looked at several stream processor solutions. All of them required learning a specialized language and syntax. Since the Ramp team wanted flexibility and ease of access, the solutions were non-starters.

After research and testing, Ramp settled on Materialize. Materialize is an operational data warehouse that combines streaming data with SQL support, so teams can transform data in real-time and power critical business processes.

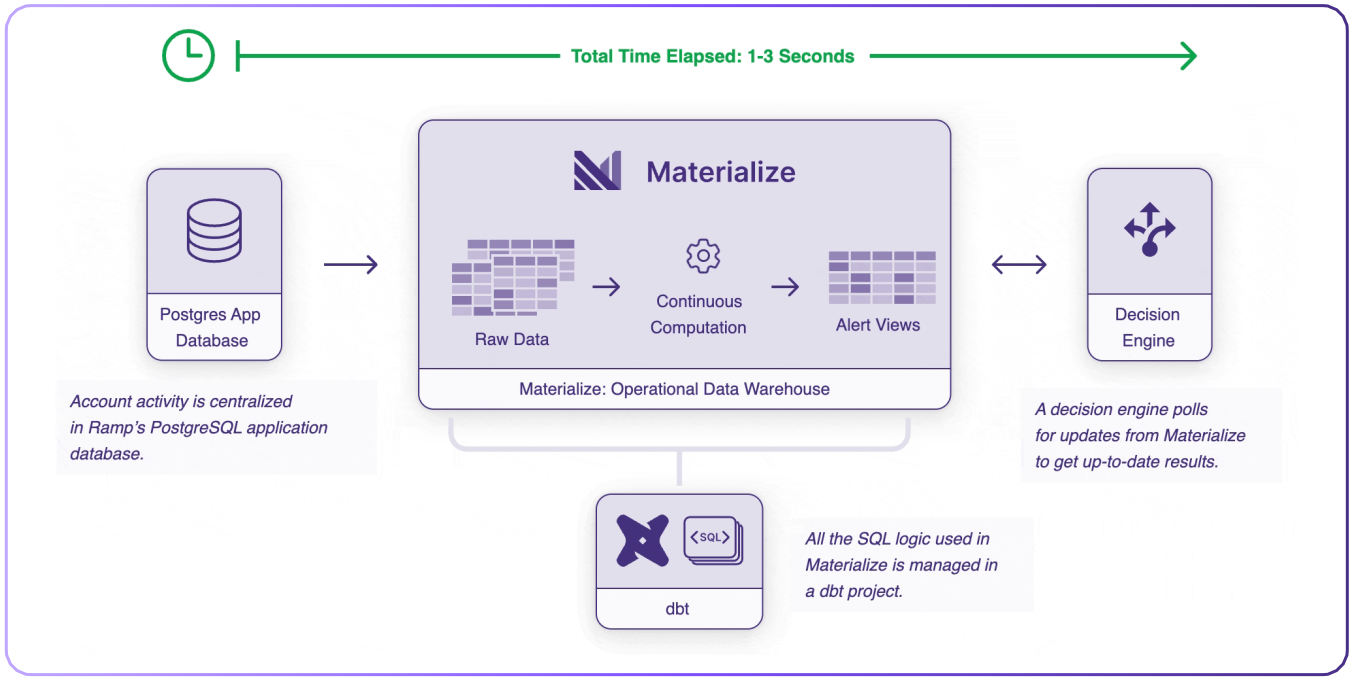
First, Ramp invested in establishing secure and performant data pipelines from production data sources. In this new architecture, the PostgreSQL database's write-ahead log (WAL) streamed directly to Materialize. This gave Materialize an up-to-the-second replica of the relevant PostgreSQL tables from the database. Fraud alert views were built from these replicated tables and polled by a decision engine.

This architecture scales in multiple dimensions. If data sources grow in size, Materialize can scale its computation power to match them, so there is no need to partition sources due to size constraints.

Ramp easily ported its rules-based fraud detection logic into Materialize. The Ramp team originally wrote the SQL logic on their analytical data warehouse, but they couldn't operationalize the underlying queries. Materialize operationalized the SQL logic, with streaming data and continuous transformation, so Ramp could detect fraud in real-time.

The porting process was simple. Ramp's development team "lifted and shifted " their query architectures onto Materialize from their analytical data warehouse. Now everyone on Ramp's team can work on real-time data products with basic SQL knowledge – no specialized training or expert consultation required.

After operationalizing the fraud detection system on Materialize, Ramp saw immediate improvements in performance. Instead of detecting fraud in over an hour, the system could now detect fraud in 1-3 seconds.



Since Ramp moved their real-time fraud detection system into Materialize:

- 50% of hacked accounts are flagged before any losses occur
- 60% of losses from account takeover have been eliminated
- 10x in infra cost reductions

The takeaway is clear. Operational data warehouses are the best option when low latency is key, especially for real-time fraud detection.

Operational Data Warehouses: The Future of Real-Time Fraud Detection

Analytical data warehouses can help create accurate fraud models, but to power real-time fraud detection systems, you need an operational data warehouse.

Operational data warehouses combine streaming data, SQL support, and continuous data transformation to trigger fraud alerts in real-time. This allows companies such as Ramp to detect fraud in seconds, rather than hours, and saves significant amounts of money by stopping fraudsters in their tracks.

To see if Materialize is right for you, sign up for a free sandbox account right now to get started at no cost.