Streaming Data Pipelines: How a Shift-Left Approach Cuts Latency and Cost

10/15/2025

Nathan Rayens, Charles Jekal

In collaboration with

Table of Contents

Introduction

The State of Data Pipelines

All data organizations manage some form of pipeline infrastructure to ensure that their data is regularized and actionable. A rich ecosystem of platforms, notably including Databricks and Confluent, supports this effort. In addition, many organizations are leveraging streaming solutions for fault tolerance and real-time processing; however, there is often a gap between streamed data and the batched or time-gated processes that drive state updates in analytical data products. Beyond this, any repeated transformation across multiple analytics processes is objectively wasted effort and compute dollars.

The Operational and Analytical Estates

To speak to this disparity between real-time streaming data and often batched analytical processing, Confluent defines these different data domains as distinct *Estates*: Operational and Analytical. The Operational Estate handles rapid processing, real-time updates, service-to-service communications, and substantial read/write actions. This includes state changes throughout applications and systems. Conversely, the Analytical Estate manages the decisions and insights derived from application and system data, including dashboarding and modeling.

An Opportunity to Shift Left

Shifting left refers to moving data processing tasks like cleansing, enrichment, and validation, closer to the point of data creation, rather than waiting until data reaches the analytical layer. For example, if a pipeline enables fulfillment of product orders for customers, it may have to extract, transform, and load (ETL) data for products, customers, and order transactions with distinct logic. There will likely be shared steps, especially around regularization and data quality, that each of those operations will do (e.g., pipelines may remove nulls or send non-conforming records to a dead letter queue (DLQ)). By shifting left, pipelines can group those preprocessing and enrichment steps so that the only data that reaches our Analytical Estate is the cleansed product orders per customer view.

Broadly, if the pipeline can identify non-conforming records earlier in the pipeline, such as at transaction time, those records will never advance forward and pollute action and insights downstream. Beyond these benefits, shifting left enables different business components to self-service off of clean, analytics-ready data throughout the platform. Hyper-specific data configurations do not need to be made each time a new initiative is proposed.

Motivation

Shifting left sounds compelling in theory. It makes sense that detecting nonconforming data and combining transformation earlier in a pipeline would be more efficient, as that nonconforming data will no longer need to be handled downstream and common transformations will not need to be repeated multiple times. Additionally, it is intuitive that democratizing data access by making clean and useful data available upstream can unleash innovation. Despite this, there is limited empirical evidence for the benefits of a Shift Left strategy.

This white paper describes a set of experiments to evaluate the compelling Shift Left theory. This work explores the harmonization of leading platforms Confluent and Databricks to evaluate how organizations might unlock new efficiencies and insights.

Experiment Design: Shifting an Existing Real-Time Pipeline Left

Background

Data Surge is a premier AI/ML services company with extensive experience in both the public and private sectors. One of our key areas of expertise is entity resolution, which empowers organizations to turn disparate and conflicting data sources into a clear, holistic view of individual people, companies, and products. Previously, Data Surge developed a state-of-the-art entity resolution solution called EntityStream, extensively leveraging Databricks as a comprehensive data platform. We encourage readers to further explore EntityStream by visiting our website or reaching out for a demo.

Unlike many alternatives, EntityStream was designed with real-time data streaming and resolution in mind. As the solution matured, it made sense to transfer streaming management to Confluent Cloud. Initially we migrated the ingest and final routing flows, and then we began to evaluate how shifting specific ETL tasks into Confluent Cloud Flink SQL statements could change the overall pipeline architecture.

Overall Architecture

The original architecture covered three principal areas: preprocessing, vectorization, and resolution. These latter two categories, vectorization and resolution, both leverage multiple custom models managed with MLflow in Databricks. It is logical to keep these components within the Databricks environment, as they form part of the broader Analytical Estate. In comparison, the transformation, quality control, and in-stream enrichment processes within the preprocessing stage are much more aligned with the Operational Estate. This alignment made it transferable directly to Confluent Cloud (Figure 1).

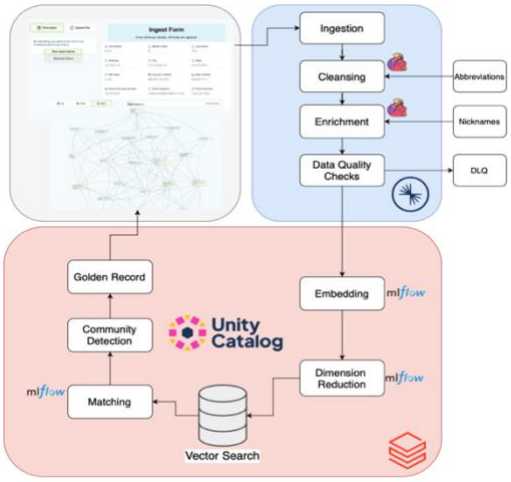

Figure 1: Simple overarching architecture diagram for EntityStream pipeline

Synthetic Data

EntityStream uses a custom synthetic dataset of 325 million data points that mimic a case management system. Each row represents an individual and may contain fields like name, address, and social security number. Additionally, each row may be linked to additional rows that describe the same person but may conflict with information for the same fields. This represents the complexity of systems that ingest data from different sources or track entities that naturally change over time, such as people.

firstName	middleInitial	lastName	localAddress	localCity	localState
John	K	null	null	null	null
Johnny	null	Woud	null	Weaverside	New Mexico
John	K	Gibson	919 Regina Mountains Suite 833	null	New Mexico
Jonnie	K	Barnett	534 Jonathan Underpass	null	NM
John	K	Wood	919 Regina Mountains Suite 833	Port Stacey	null
null	K	null	919 Regina Mountains Suite 833	Weaverside	West Virginia

Figure 2: Limited sample data for a person showing fields that are missing or conflicting.

Experiment Conditions and Measurement Points

To fairly assess the value of the shift-left strategy, we built two parallel pipelines. In one, the preprocessing tasks (the first core component of the architecture) were executed in Confluent Cloud. In the other, they were executed within Databricks. Because the architecture is shared for the rest of the components, we focused experiments to just this first preprocessing stage. Both pipelines perform typical data cleansing tasks as well as additional domain-specific cleansing with standardized state abbreviations and enrichment via added nicknames for the first name field where relevant. These steps help to pull similar records close together in vector space during the rest of the pipeline.

Baseline Pipeline

The Baseline Pipeline leverages the original approach and directly reads from a Confluent Cloud-managed topic using Spark structured streaming. Data is cleansed, transformed, and stored with the Spark DataFrame API. An AWS EC2 cluster remains active for the entire duration of the execution. This pipeline has three measurement points:

1. Initial load time – timestamp where first records enter the topic.
2. Final load time – timestamp where last records enter the topic.
3. Processed time – timestamp where records have cleaned and subsequently stored.

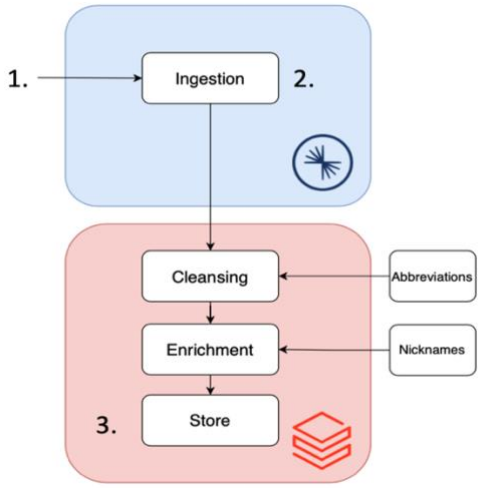

Figure 3: Baseline Pipeline steps with timestamp measurement points.

Shift Left Pipeline

The Shift Left Pipeline takes the preprocessing steps and moves them completely into Confluent Cloud using Confluent Cloud Flink statements to perform transformations. The Shift Left Pipeline has largely similar measurement points to the Baseline Pipeline above. However, because the Baseline Pipeline includes writing to a Databricks streaming table, we have taken two sets of measurements, termed the Ideal and Conservative Latency Measures, to fairly evaluate shifting left across workflows.

Ideal Latency Measure

The Ideal Latency Measure (Figure 4) includes only Confluent Cloud processing steps. After the data has been cleansed and enriched, it is ready for use by any topic listener. This approach uses the following timestamps:

1. Initial load time – timestamp where first records enter the topic.
2. Final load time – timestamp where last records enter the topic.
3. Confluent processed time – timestamp where transformations have been applied and records are written to the cleaned topic.

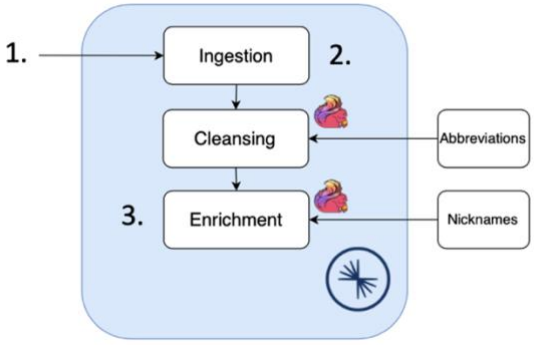

Figure 4: Shifted Left Pipeline steps with Ideal Latency Measure timestamp measurement points.

Conservative Latency Measure

The Conservative Latency Measure (Figure 5) includes a write step in Databricks for a more direct comparison to the Baseline Pipeline. This approach uses the following timestamps:

1. Initial load time – timestamp where first records enter the topic.
2. Final load time – timestamp where last records enter the topic.
3. Databricks storage time – timestamp where processed data is read from the cleaned topic and stored.

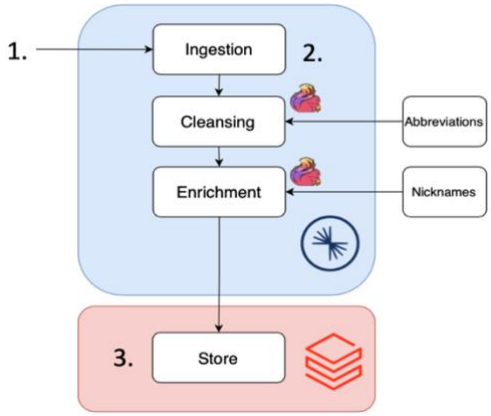

Figure 5: Shifted Left Pipeline steps with Conservative Latency Measure timestamp measurement points.

Tooling and Statistical Measures

In addition to the experimental conditions, we elected to go for minimal resources in both Databricks and Confluent Cloud because that choice simplifies fair comparisons between the pipelines and avoids leaning into individual strengths of either platform. We intend to scale the resource allocation design shown here in our next experiments.

Table 1: Summary of tooling and shared conditions across pipelines

Component	Baseline Pipeline	Shift Left Pipeline	Notes
Kafka Cluster	Basic	Basic	Same for parity
Flink Pool	N/A	Autoscale to 10 CFUs	Minimal Flink pool
Databricks Cluster	I3.2xlarge single node	N/A	Minimal cluster
Topic Partitions	1	1	Same for parity
Data Volume	Batches of 500 to 500k	Batches of 500 to 500k	Same for parity

Each condition was measured five times to capture variance and for each experiment condition, a two-way analysis of variance (ANOVA) was used to compare statistical differences between pipeline strategies. A two-way ANOVA is appropriate because it assesses contributions to overall variance from two categorical variables, pipeline choice and batch size, on a continuous variable, run time, enabling aggregate comparisons of overarching behavior. Additionally, this method allows for capture of interactions between batch size and pipeline choice that can speak to decision points for scales at which different pipeline strategies are appropriate.

Results: Shifting Left Reduces Processing Times and Cost

Ingest Time Unaffected by Shifting Left

To precisely measure how quickly Confluent Cloud can ingest data and to validate experiment design, we recorded the offset in time between the first and last record arrival on a topic for logarithmically scaled batches from 500 to 500,000. Using a two-way ANOVA, we found no significant difference between ingest methods (Figure 6). This is unsurprising because both pipelines leverage Confluent Cloud and are configured the same way with respect to clusters, topics, and partitions.

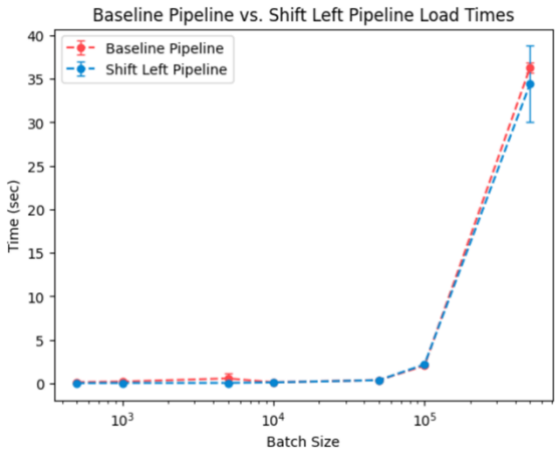

Figure 6: Ingest time does not vary by pipeline strategy. Statistical comparisons via two-way ANOVA.

Processing Time Significantly Reduced by Shifting Left

Next, we analyzed the difference in processing times for the two pipelines. For the Baseline Pipeline, processing time was measured as the difference in timestamps between when the last record was ingested to when the final storage operation completed. For the Shift Left Pipeline, two differences were considered. The first, the Ideal Latency Measure, subtracted the timestamp from when the last record was ingested from the timestamp where the last record was written to the cleaned topic (Figure 4). The second, the Conservative Latency Measure, instead subtracted the timestamp of the last ingested record from the timestamp of the final storage operation completed in Databricks (Figure 5). This enables isolation of effects by matching potential overhead of storage operations that the Baseline Pipeline likely experiences.

Using the Ideal Latency Measure (Figure 7A), we found that shifting left significantly reduced processing time by as much as 95% ($p < 0.0001$, indicating this result would be highly unlikely to occur if pipeline choice had no real effect). This pattern persisted with the Conservative Latency Measure (Figure 7B), where again, shifting left significantly reduced processing times ($p < 0.0001$, indicating this result would be highly unlikely to occur if pipeline choice had no real effect). These results strongly support the value of shifting streaming transformation and enrichment tasks into Confluent Cloud.

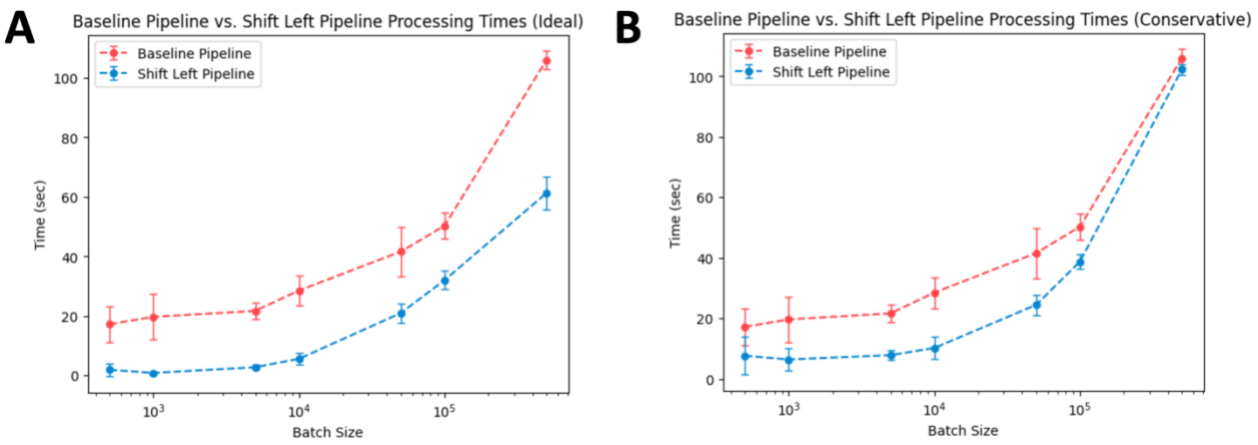

Figure 7: Processing time significantly reduced by shifting left. A. Comparison using the Ideal Latency Measure shows significant reduction in processing time ($p < 0.0001$). B. Comparison using the Conservative Latency Measure significant reduction in processing time ($p < 0.0001$). Statistical comparisons via two-way ANOVA.

Compute Costs Reduced by Shifting Left

Using similarly minimal compute for both strategies, we estimated an hourly differential rate comparing the Databricks cluster used in the Baseline Pipeline against the Confluent Cloud Flink pool used in the Shift Left Pipeline. Only these two compute resources were considered because all the other resources were the same between the pipelines.

Table 2: Comparison of differential compute costs when shifting left

Resource	Variable Costs	Variable Rate	Fixed Costs	Total Cost
i3.2xlarge	2 DBU/hr	$0.15/DBU	$0.62/hr	$0.92/hr
Flink pool	~4 CFU/hr	$0.21/CFU	N/A	$0.81/hr

Using cost estimations provided by Confluent and Databricks/AWS, we found that the Shift Left Pipeline had a roughly 11.5% reduction in differential compute costs compared to the Baseline Pipeline. This illustrates that the latency improvements shown above are not gained through cost increases, but rather, in parallel to a cheaper strategy.

Conclusions

These experiments show that shifting streamed data transformation tasks into Confluent Cloud is a practical and effective choice. The Shift Left strategy yields performance improvements and unlocks a more sustainable and scalable data architecture. Across data volumes, the Shifted Left Pipeline displayed a substantial reduction in processing time, even when including an additional storage step. This underscores how shifting left can reduce the time-to-value for an organization – the sooner uniformly cleansed and enriched data is available for downstream consumers, the sooner differential value can be realized. In our experiments we found time reductions of up to 95%. This means that decision timelines could be reduced from hours to minutes or for real-time customer interactions, seconds to milliseconds!

Further, this enhancement on processing time did not come at the cost of increased infrastructure spend. Rather, pivoting to stream-native transformation tools like Confluent Cloud led to reduced compute costs by over 11.5%. To contextualize this number, Data Surge has worked with public sector clients who spend over $800,000 per year on similar pipelines. An 11.5% reduction would translate to nearly $100,000 saved annually, without even considering the benefits of data democratization and minimizing repeated transformations in terms of total cost of ownership (TCO).

Beyond performance and cost metrics, organizations that shift transformation and enrichment tasks left gain additional value by optimally aligning the Analytical and Operational Estates. With this approach, non-conforming data can be easily removed before it impacts analytics or data products. Further, shifting left reduces the need for repeated processing on the analytical platform, streamlining workflows and reducing redundancy. Perhaps most importantly, having “analytics-ready” data in streams in the Operational Estate empowers teams across the organization to build custom use cases that meet their business unit needs in both the Operational and Analytical Estates. This democratization of “analytics-ready” data access can unleash substantial value and innovation.

Shifting left does not diminish the value of the Analytical Estate. In our experiments, the AI/ML modeling and visualization tasks are still best matched with analytical workload tooling like Databricks. The Shift Left strategy is not a replacement, but a critical complement in a broader data modernization journey. Apart from cost savings, imagine the compounded impact when you factor in less code to maintain, faster onboarding of new data sources and easier access to clean, analytics ready data. Data Surge is uniquely positioned to guide this transformation, with deep partnerships across Confluent and Databricks and a proven track record in both public sector and commercial domains. Be sure to watch out for our next white paper, which will explore shifting left at enterprise scales with substantially increased compute resources. We invite you to connect with us to explore how these results and many more strategies can translate into outsized impact for your data vision.