

DATA
STREAMING
WORLD TOUR

PARIS



CONFLUENT

Real time Data Lake avec Confluent et IBM watsonx.data



Eric Carlier

Staff Solutions
Engineer,
Confluent



Vincent Gautier

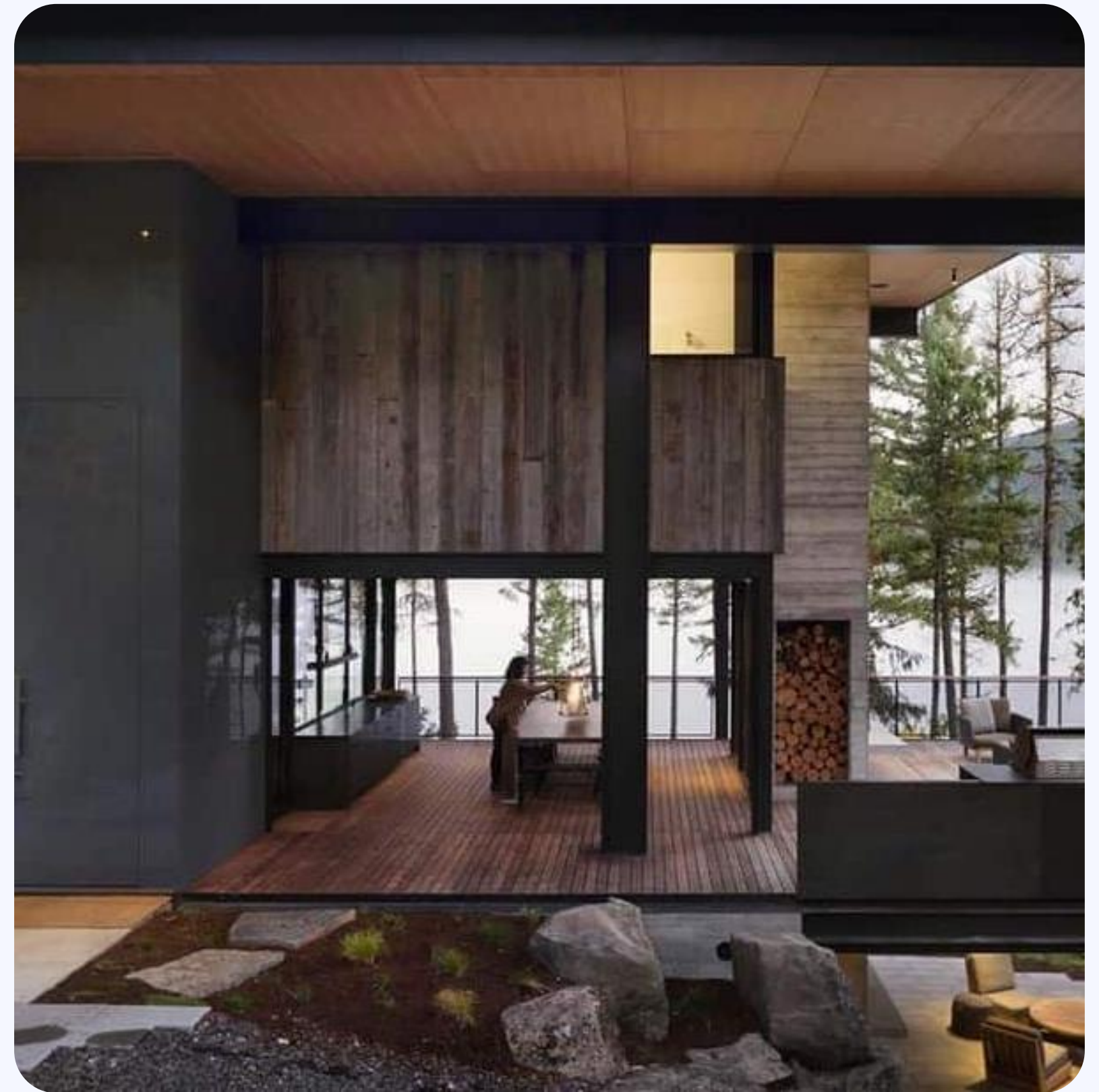
Customer Success
Engineer,
IBM



Confluent

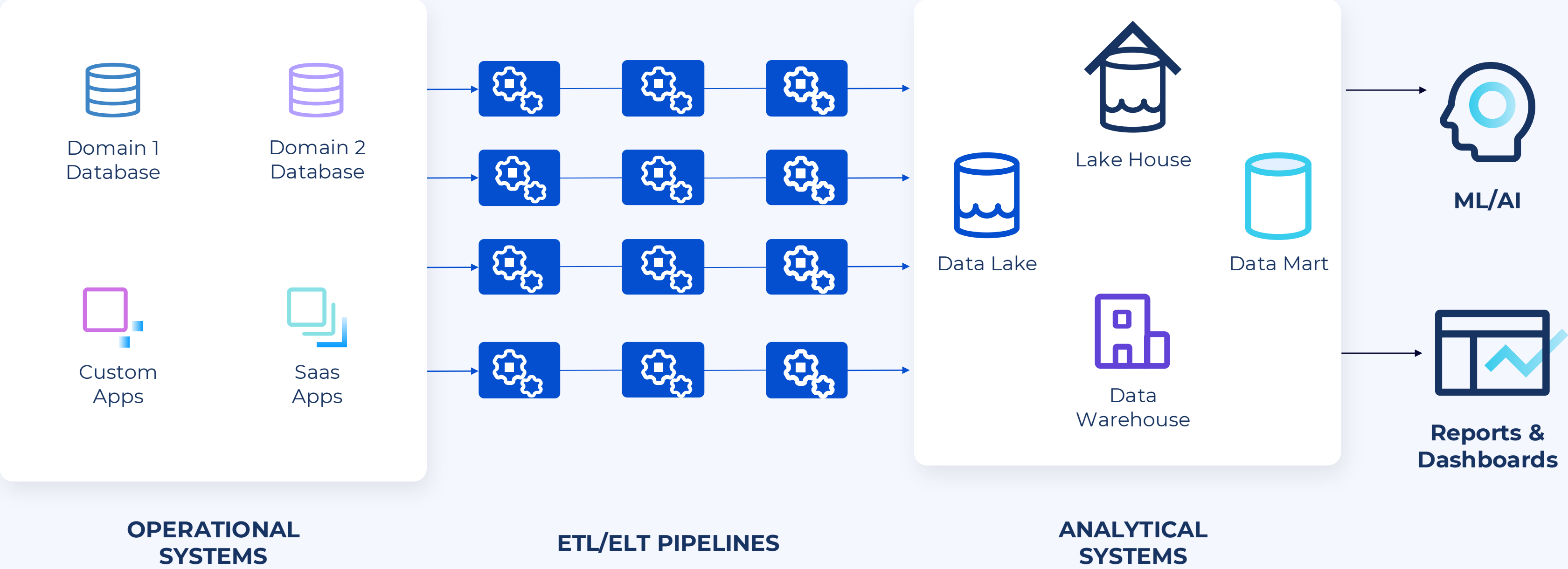


Data Lakehouses³

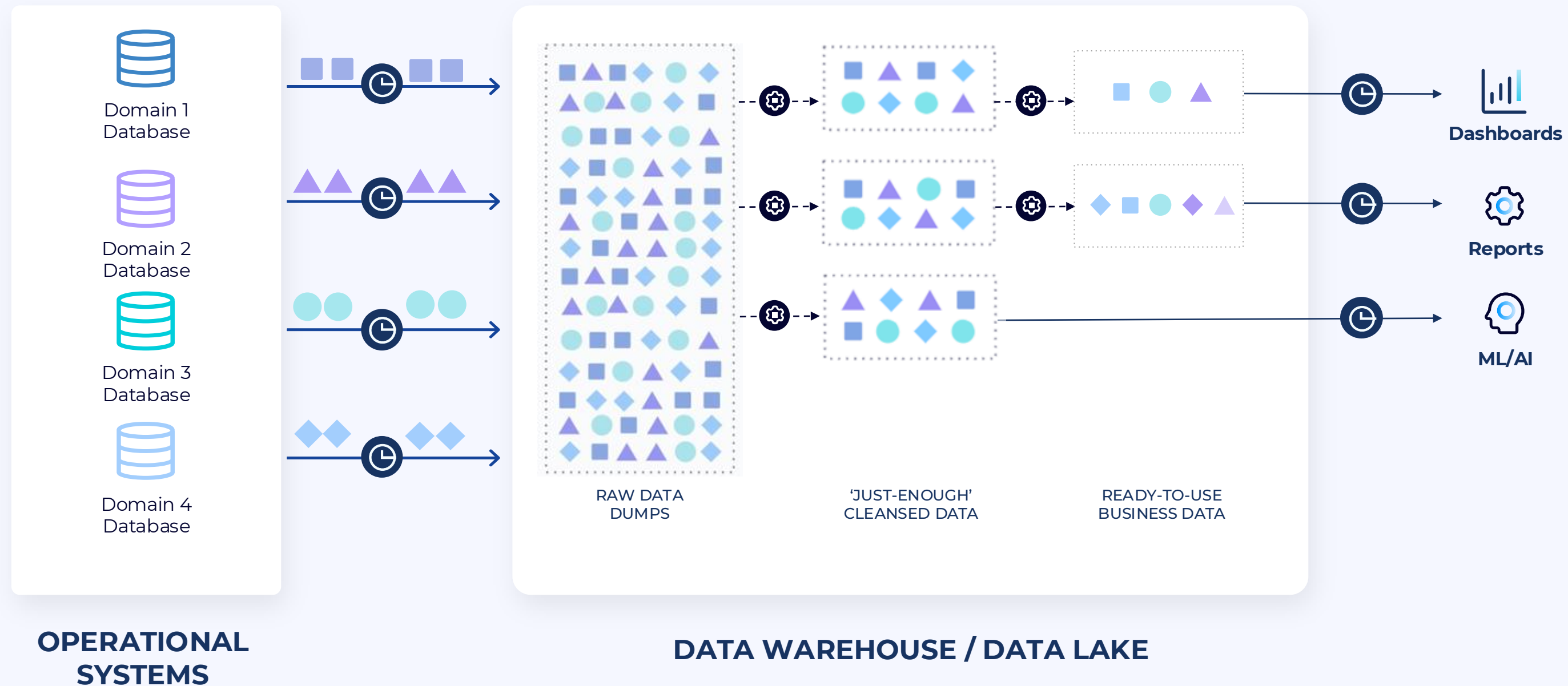




Today's Data Pipeline Approaches Are the Root of Your Data Problems

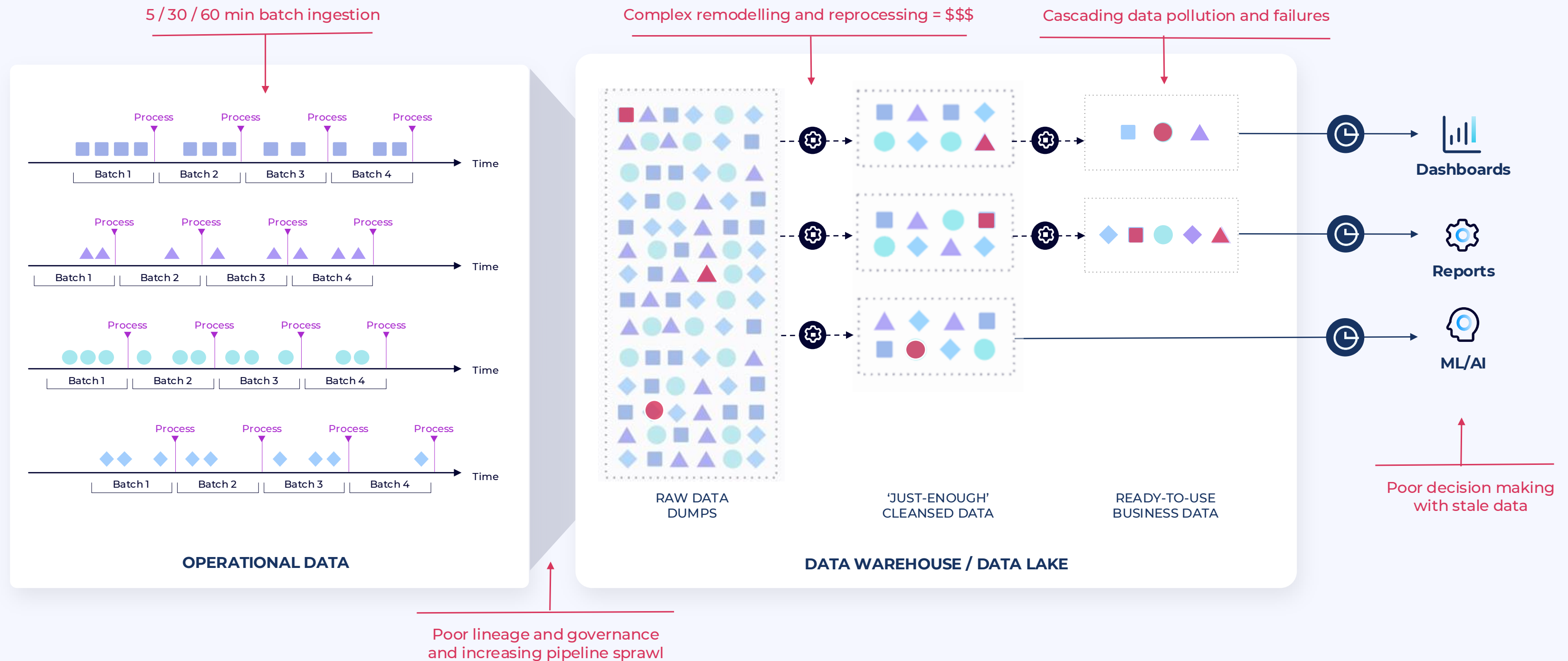


The conventional Extract, Load, Transform (ELT) architecture

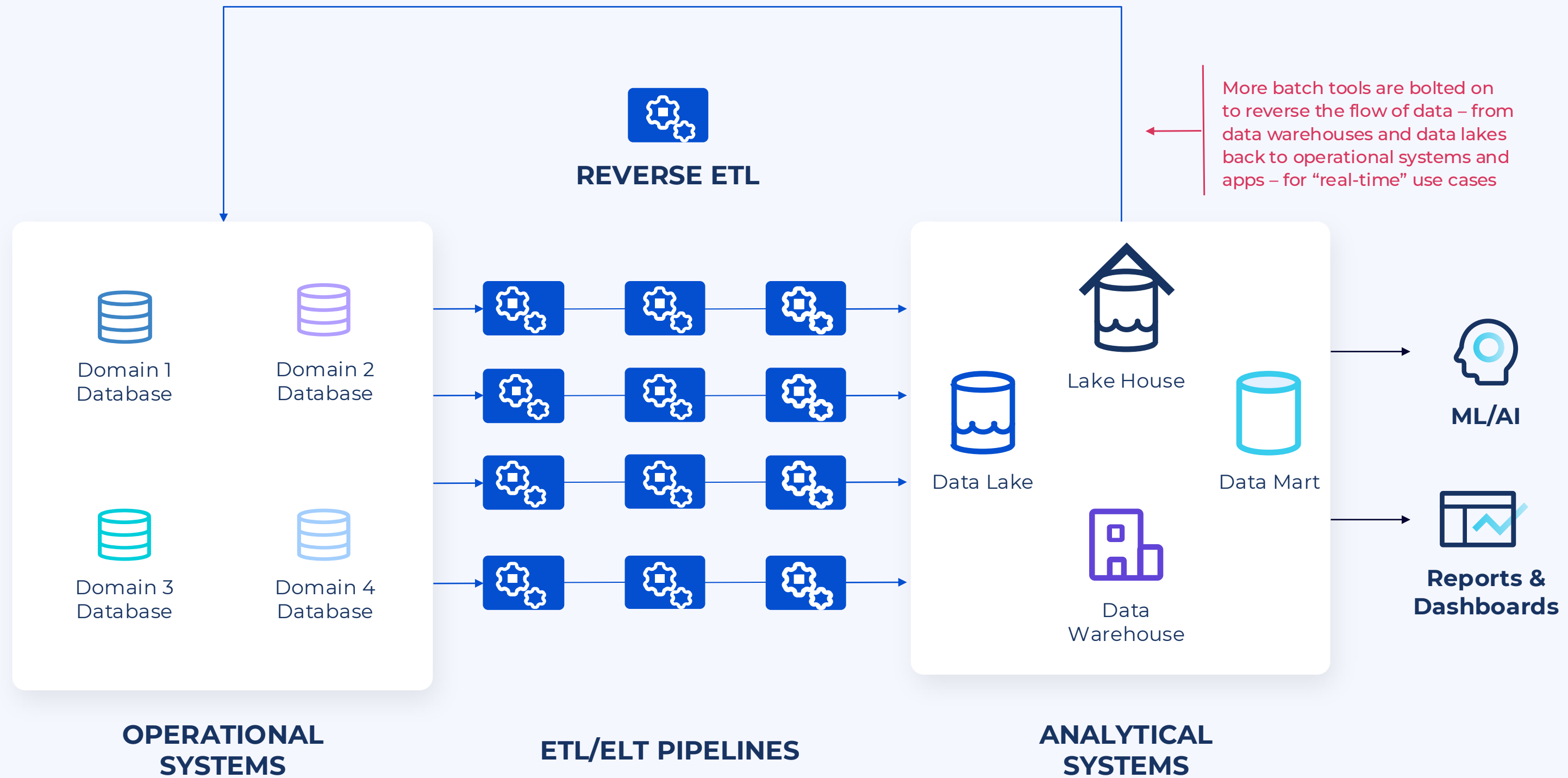




ELT pipelines are brittle, slow and inefficient

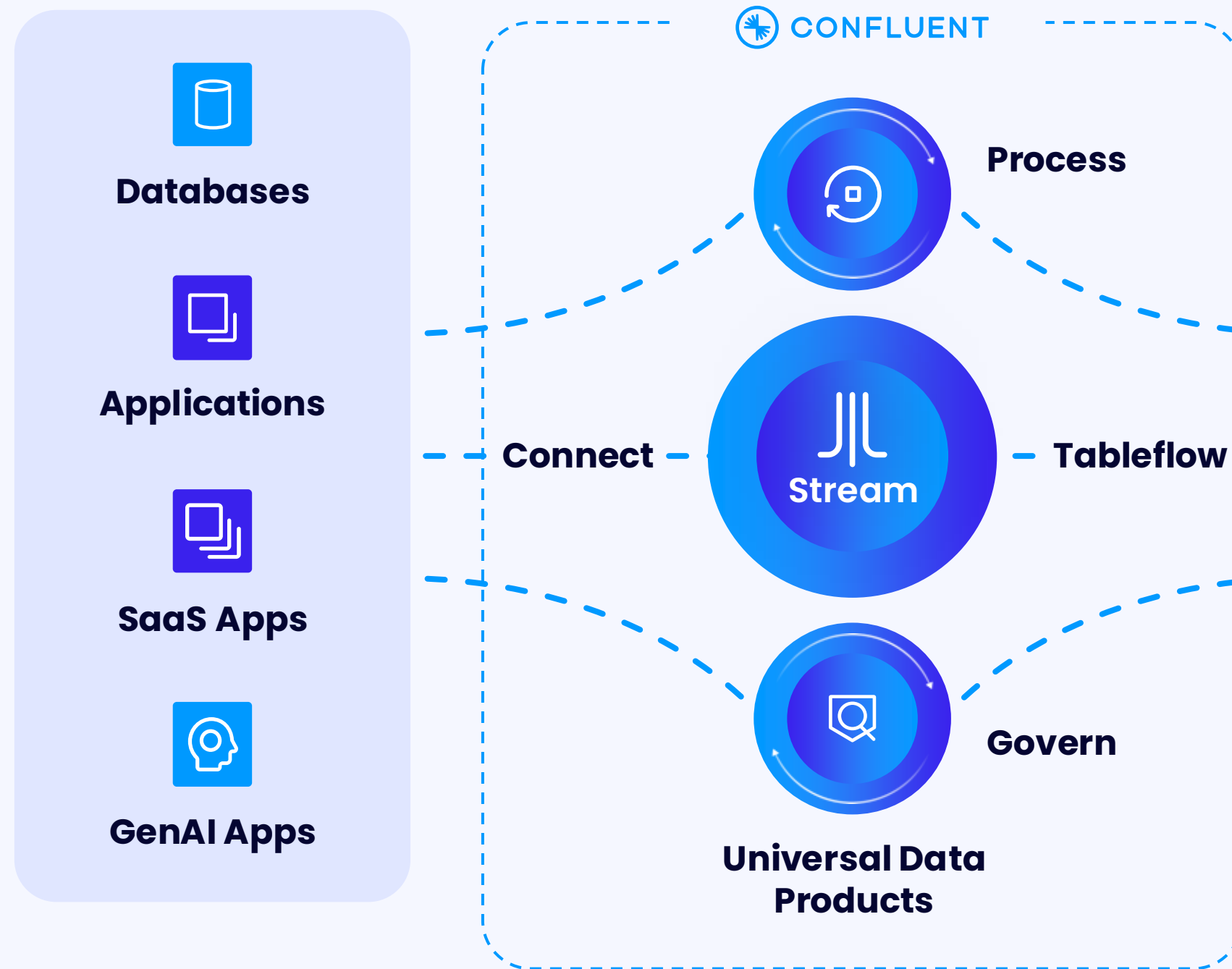


Modern applications need data to flow ‘upstream’ too

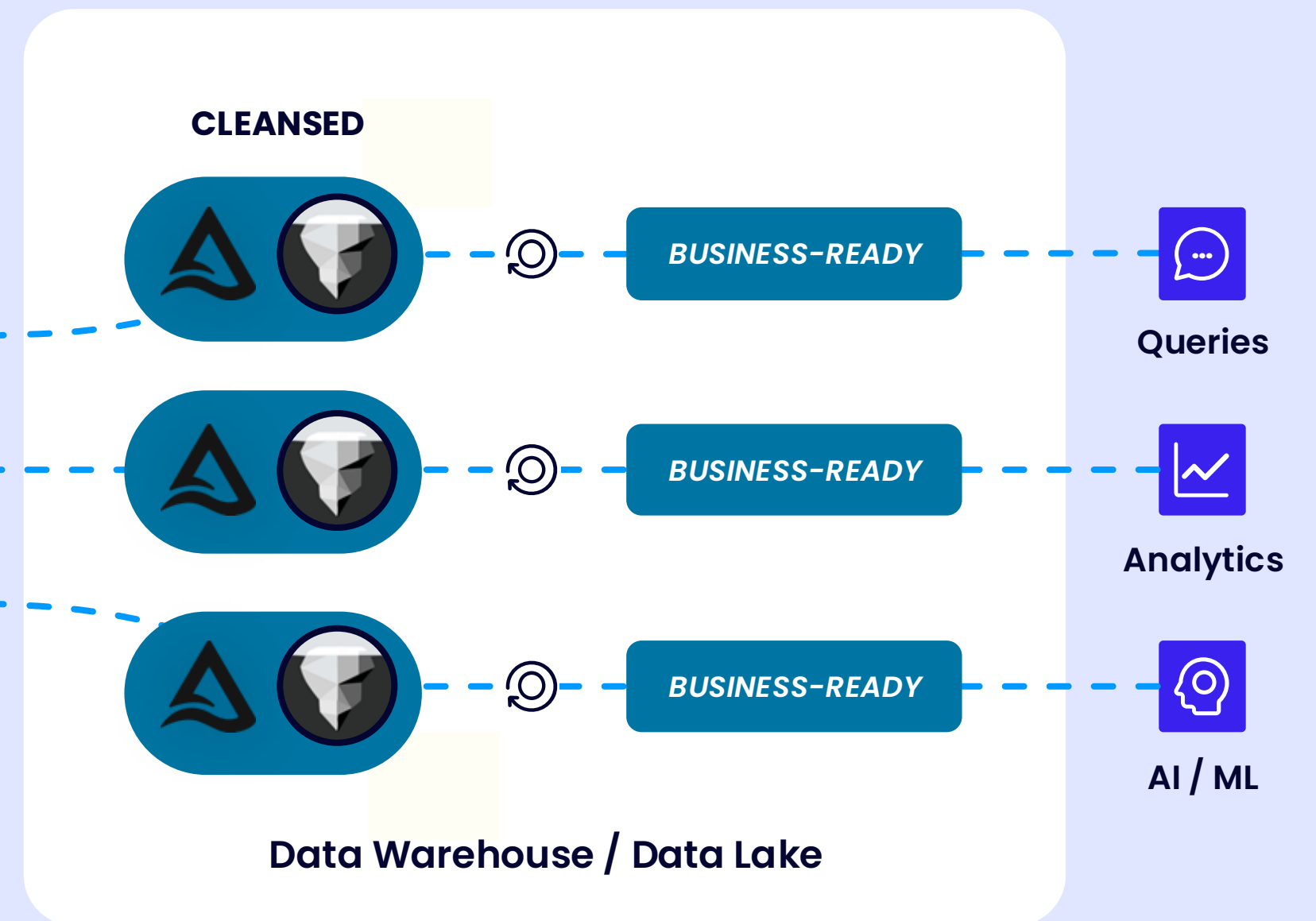


Power Applications, Analytics, and AI Use Cases With Real-time, Trustworthy Universal Data Products

OPERATIONAL ESTATE

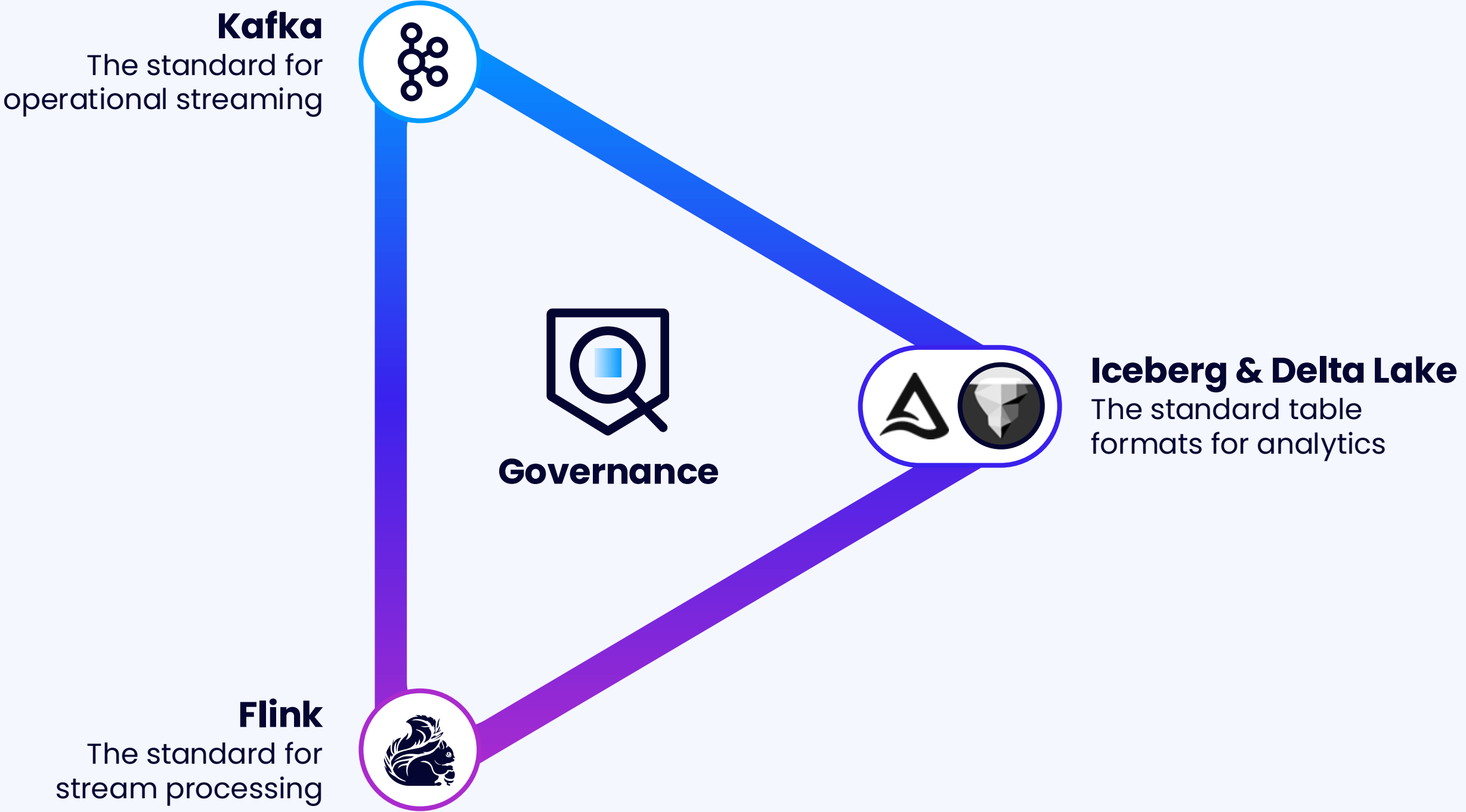


ANALYTICAL ESTATE

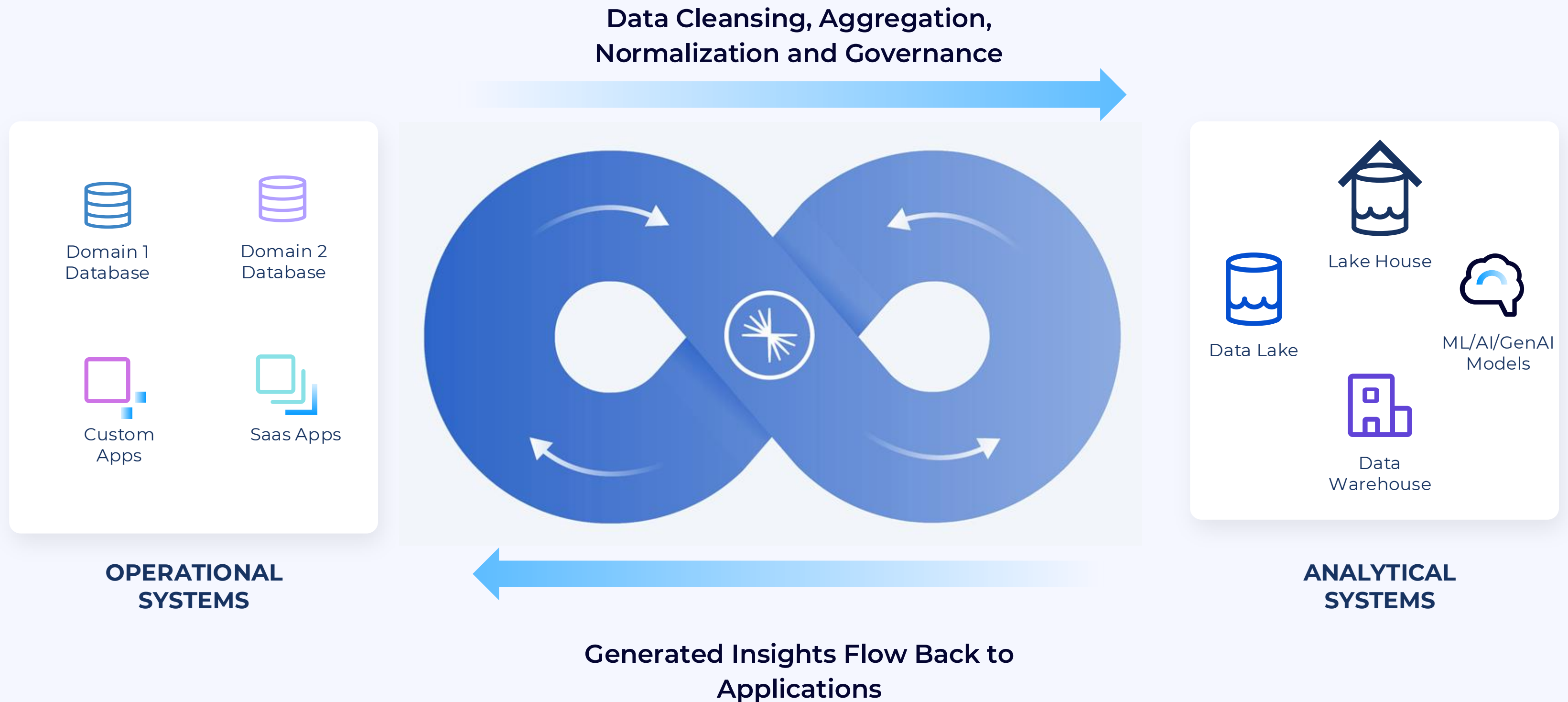




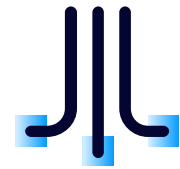
We Are Unifying Important Standards



This Unification Ensures High Value Data for Analytics and AI Is Always Fresh

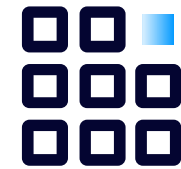


The Confluent Data Streaming Platform Advantage



Streaming

Continuously capture and share real-time data everywhere - to your data warehouse, data lake and operational systems and apps



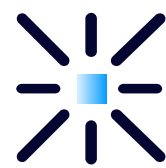
Schema Management

Reduce faulty data downstream by enforcing quality checks and controls in the pipeline with data contracts



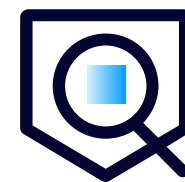
Flink Stream Processing

Continuously optimize the treatment of data, the moment it's created, for well-curated reusable data products



Data Portal

Enable anyone with the right access controls to effortlessly explore and use real-time data products for greater data autonomy



Stream Lineage

Understand the complex data relationships and the data journey to ensure trustworthiness



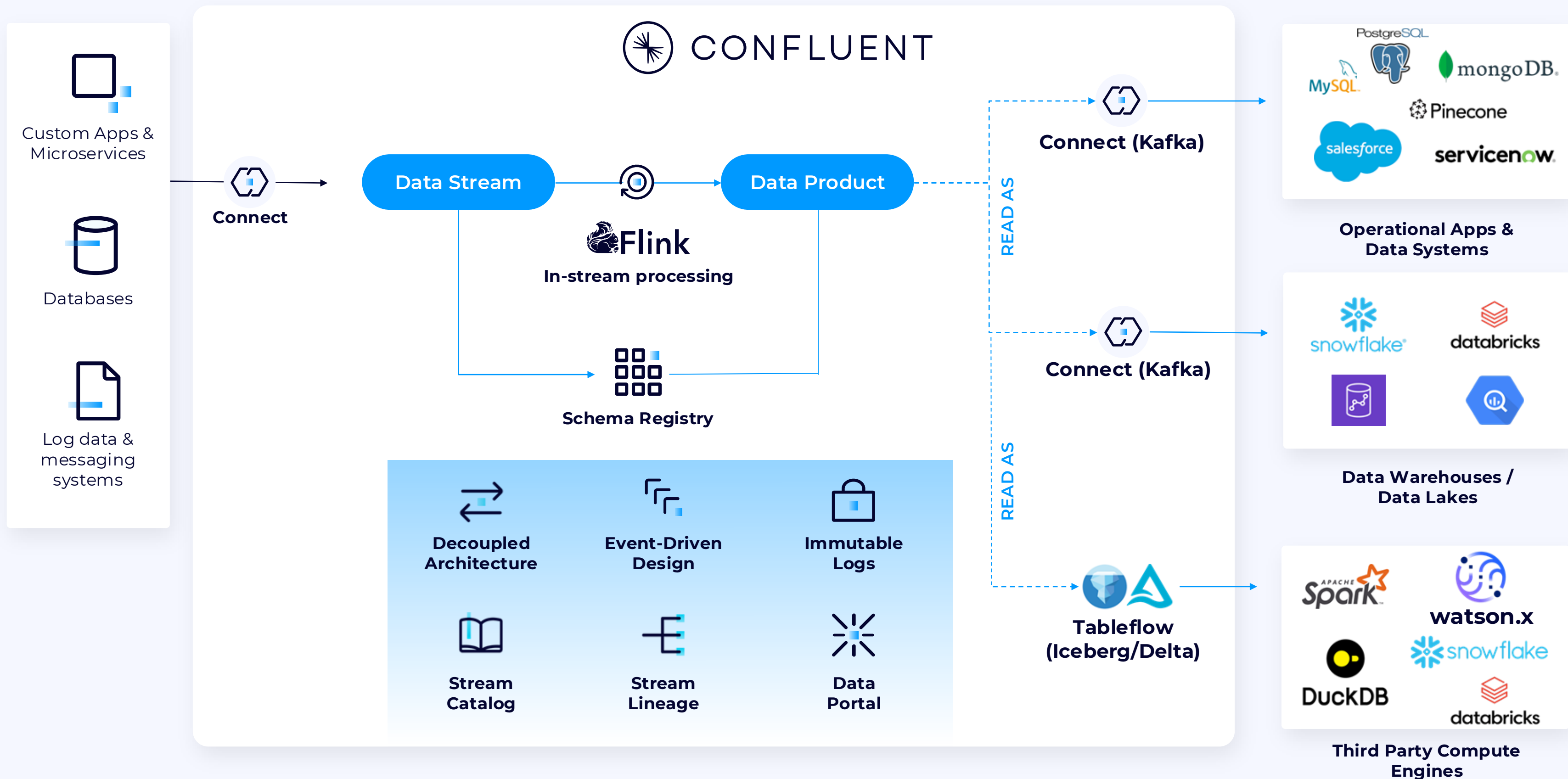
Tableflow

Simplify representing your operational data as a ready-to-use Iceberg table in just one-click



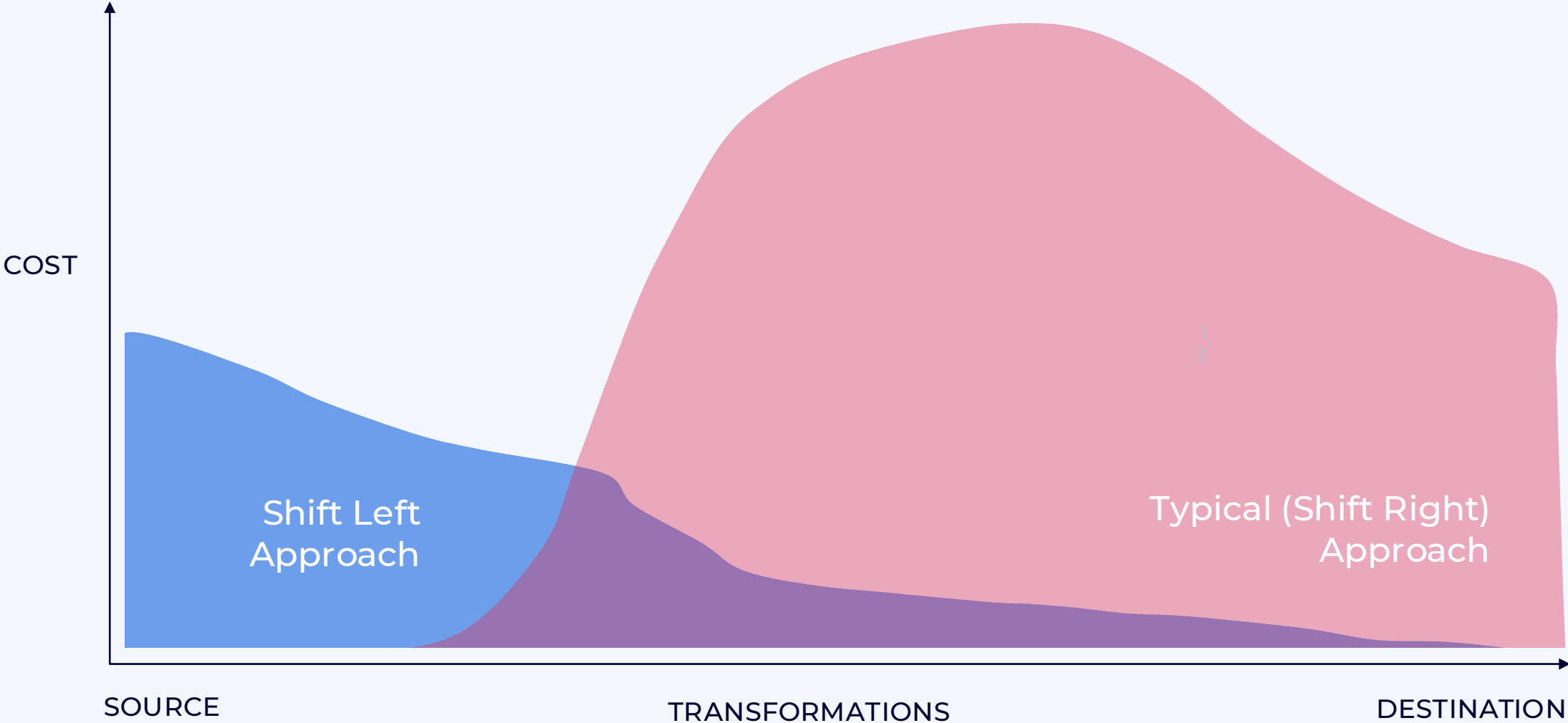
How we do it

Write Your Data Once, Read It as a Stream or Table



Maximize the ROI of Your Data Warehouses and Data Lakes

60% fewer data quality errors. 30% reduced compute costs. All with Shift Left.



We have **40,000+ ETL jobs** today. It's chaos...shifting data governance left would be **transformational** for our organization."

-Head of Data, Fortune 100 Financial Institution

Prevent Data Errors at the Source

Reduce data quality issues by **40-60%** by **shifting data governance left** to proactively keep bad data out

Cut Compute Costs in Data Warehouses

Reduce **30%** of your compute costs by **shifting data processing left** to feed multiple warehouses or lakes

Improve Data Engineer Productivity

Free up engineering bandwidth for new initiatives by **streamlining data consistency and access**



“Confluent helped us shift left on our data—giving teams full ownership of their data from source to output. Now it’s clean, validated, and production-ready upstream, reducing rework and accelerating delivery.”

Chris Kapp
SOFTWARE ARCHITECT





\$3M+

Cost savings

Replaced Azure Event Hubs and Ab Initio ETL pipelines with Confluent for streaming governed and structured Oracle CDC data to Iceberg for real-time replenishment stock data across 32 countries

35%

Cost savings

Created customer ID data products instead of through expensive joins within Snowflake for Manufacturing use cases

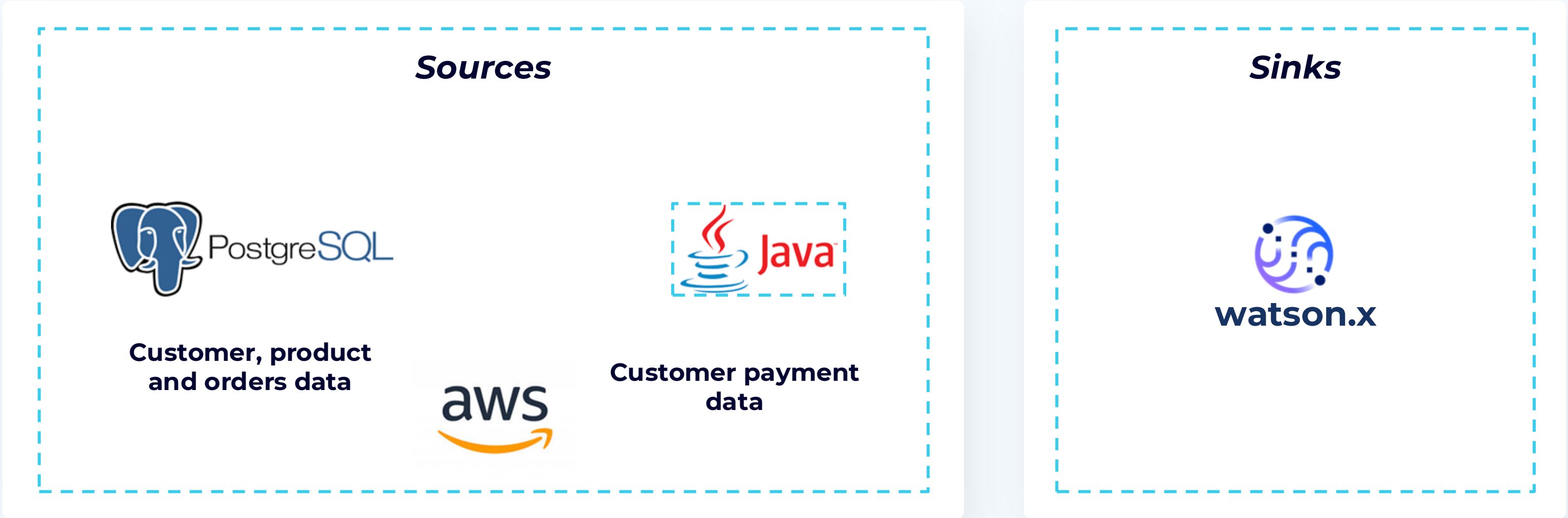
\$6M

Annual savings

Eliminated Glue/Lambda jobs, removed duplicative datasets from Matillion ELT pipelines and reduced warehouse costs for Customer 360 and Supply Chain use cases

Demo

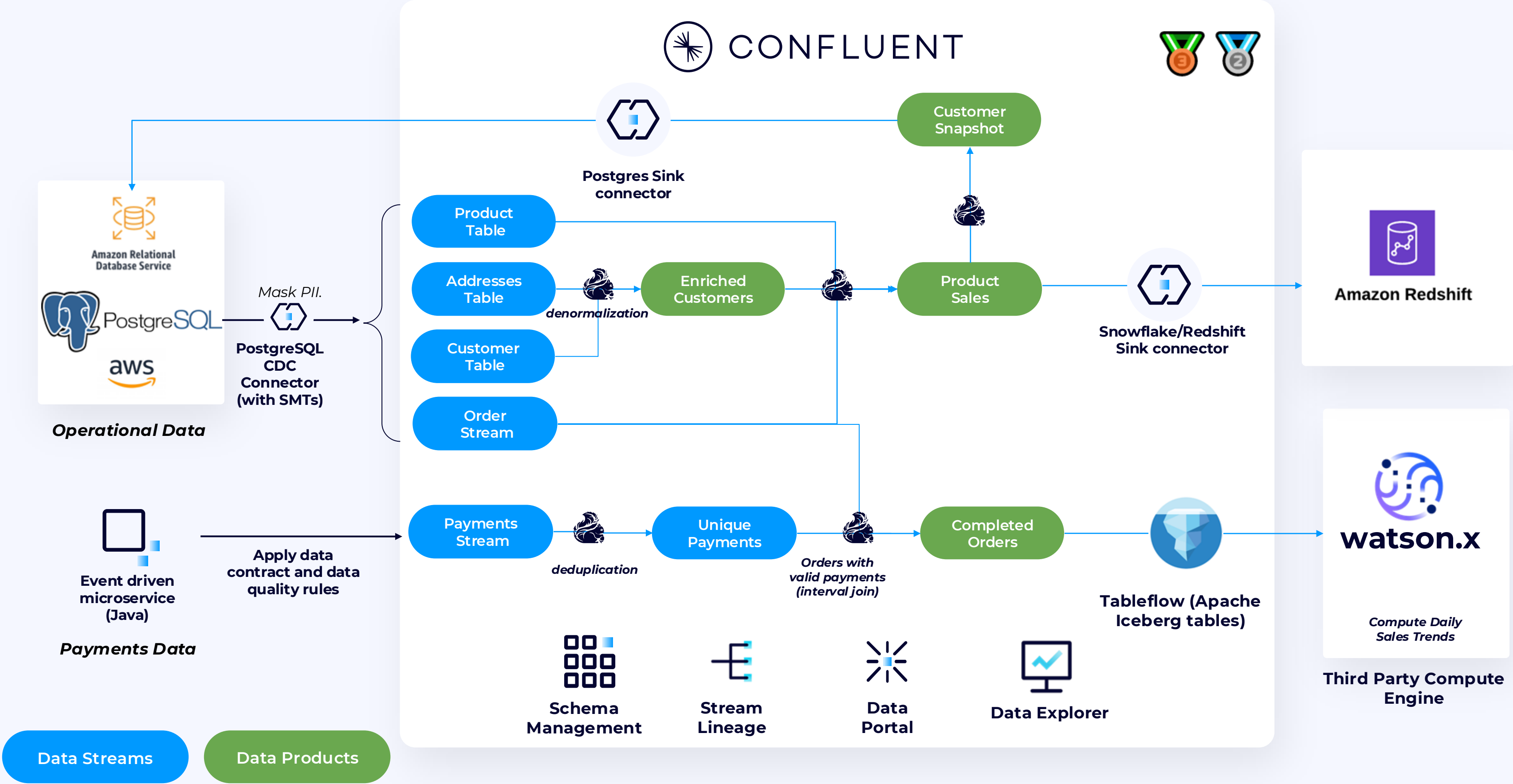


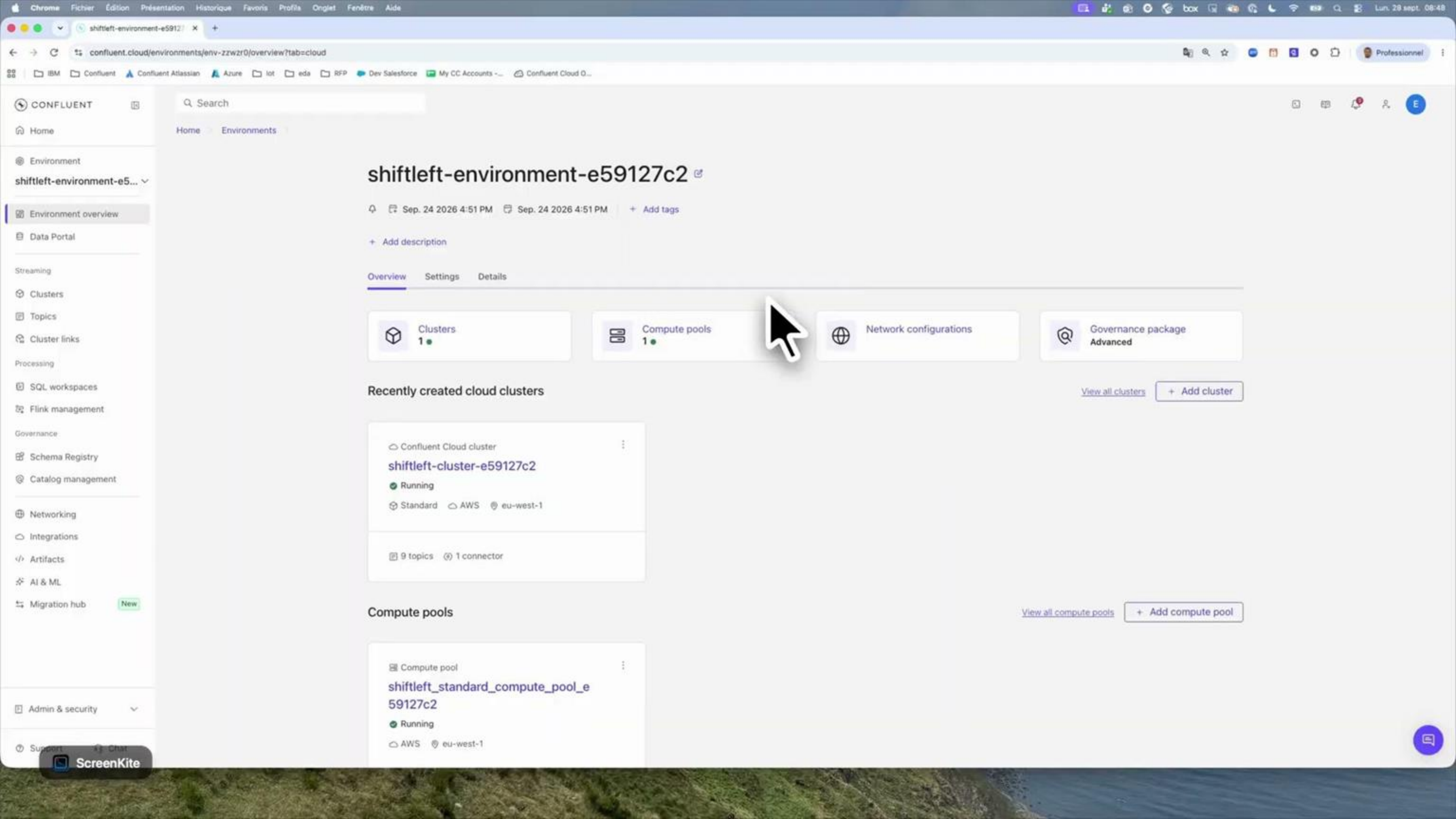


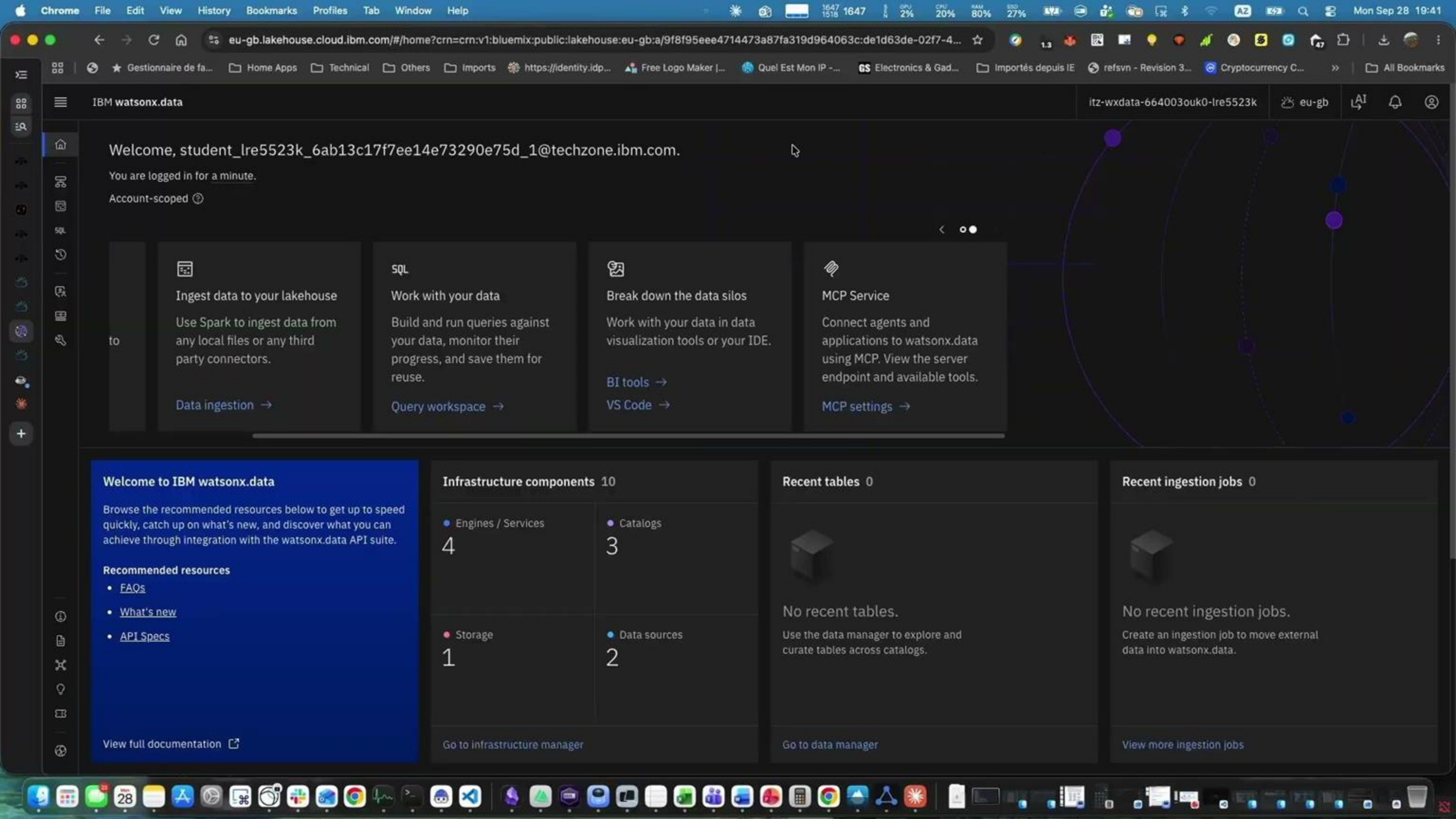
Challenges

- Data silos between 2 data teams using ELT pipelines in their Snowflake environment to implement 3 different yet related use case: inventory alerts, product sales, and sales trend analysis

Combined High-level Architecture Diagram







Materialized Tables will become *the* core object in Confluent Cloud for Apache Flink.

Materialized Tables represent the result of a continuous query (the table/topic/schema) as well as the process of materialization (the Flink Job).

```
CREATE MATERIALIZED TABLE
orders_by_product_id
AS SELECT
  product_id,
  COUNT(*) AS total,
  COUNT(DISTINCT user_id)
FROM orders
GROUP BY product_id;
```

Materialized Tables are the foundation for

- more flexible query evolution
- a Confluent **dbt adapter**

Before and After: Terraform / Hand-Rolled vs. dbt

	Before (Terraform / hand-rolled)	After (dbt-confluent)
Changing a live pipeline	Stop statement, record offsets by hand, recreate sink topic or dual-write, resubmit with offset overrides	Edit the model, dbt run (catches breaking changes safely), dbt run --full-refresh (clean one-command redeploy)
A statement crashes	Manual detection, manual resubmission, manual offset bookkeeping	Plain dbt run self-heals automatically -- no --full-refresh needed
Testing	Not really a concept	dbt test -- data + unit tests, real results, not just timeouts
Docs / lineage	Manual, if it exists at all	dbt docs generate, dependency DAG
Learning curve for a data engineer	Steep -- a new tool (HCL, state files, plan/apply)	None -- same dbt they already use
Where "before" still wins	Full infra control (networking, IAM, non-Flink resources)	Terraform remains right for that -- just not for pipeline logic



**DATA
STREAMING**
WORLD TOUR



Merci!

DATA
STREAMING
WORLD TOUR