

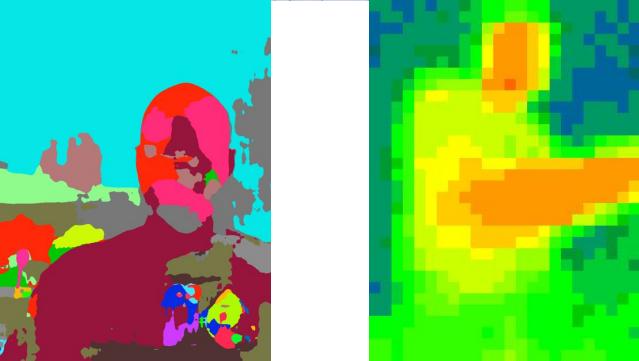
# The Never Landing Stream with HTAP and Streaming

**Timothy Spann**

Principal Developer Advocate

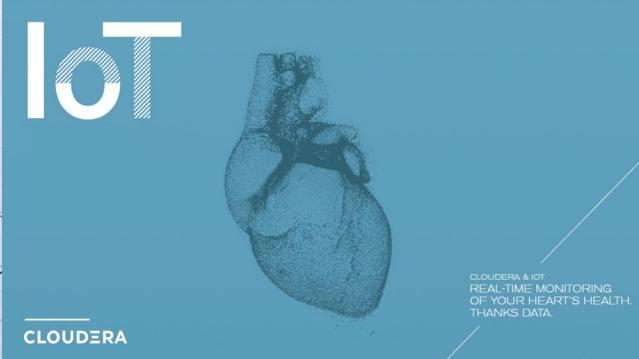
# Introduction





**ENTERPRISE  
DATA CLOUD**

CLOUDERA



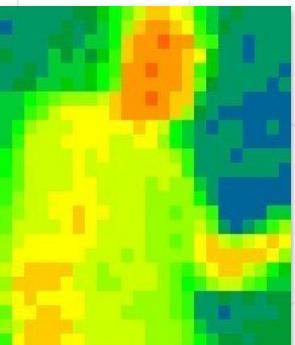
**CLOUDERA**



**EDGE  
2AI**



# FLaNK Stack



**Tim Spann**

@PaasDev // Blog: [www.datainmotion.dev](http://www.datainmotion.dev)

Principal Developer Advocate.

Princeton Future of Data Meetup.

ex-Pivotal, ex-Hortonworks, ex-StreamNative, ex-PwC

<https://medium.com/@tspann>

<https://github.com/tspannhw>

Apache NiFi x Apache Kafka x Apache Flink



DZone REFCARDS TREND REPORTS EXPERTS

## Top IoT Experts

**Tim Spann**  
Principal Developer Advocate, Cloudera  
<https://github.com/tspannhw/SpeakerProfile>

Tim Spann is a Principal Developer Advocate in Data In Motion for Cloudera. He works with Apache NiFi, Apache Pulsar, Apache Flink, and Apache Beam. He has experience in big data processing, stream processing, and data integration. He is also involved in the Apache NiFi community, contributing to its development and documentation. He has presented at various conferences and meetups, sharing his knowledge and expertise in these fields.



# Future of Data - Princeton + Virtual



<https://www.meetup.com/futureofdata-princeton/>

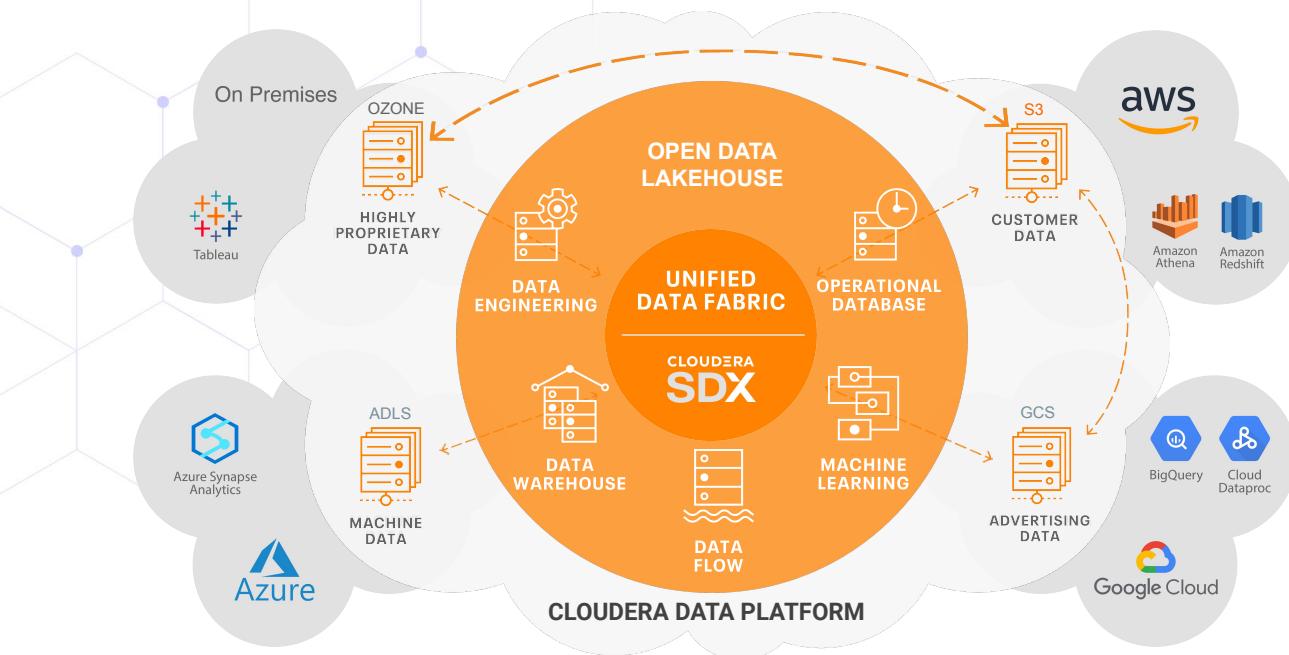
From Big Data to AI to Streaming to Containers to Cloud to Analytics to Cloud Storage to Fast Data to Machine Learning to Microservices to ...



@PaasDev

# CDP IS THE ONLY HYBRID DATA PLATFORM

Hybrid. Open. Portable. Secure.

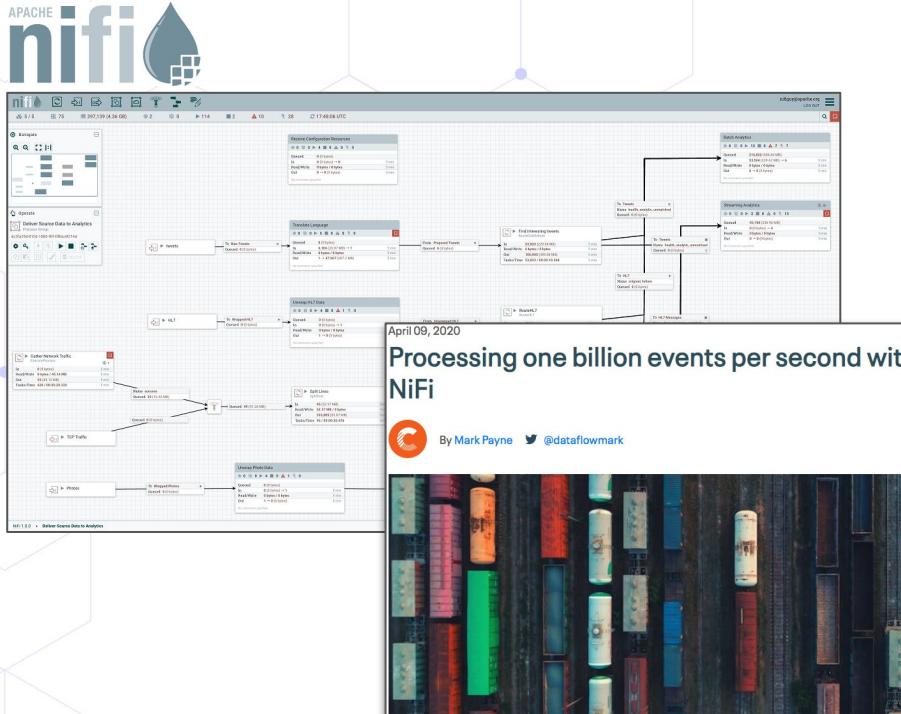


# Apache NiFi



# CLOUDERA FLOW MANAGEMENT – POWERED BY APACHE NiFi

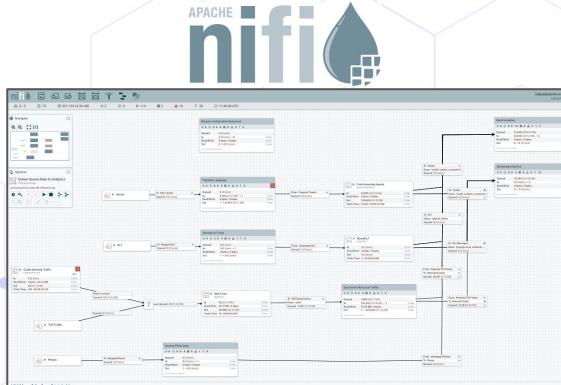
Ingest and manage data from edge-to-cloud using a no-code interface



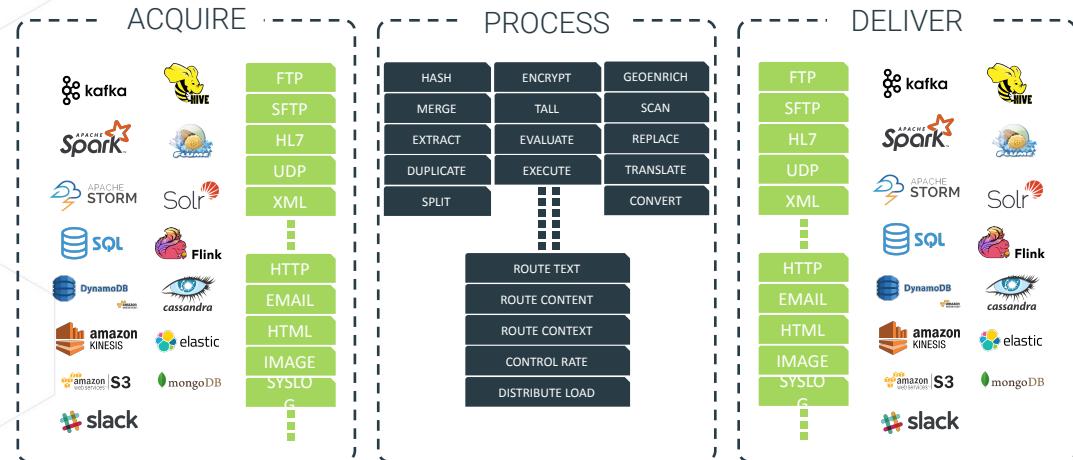
- #1 data ingestion/movement engine
- Strong community
- Product maturity over 11 years
- Deploy on-premises or in the cloud
- Over 400+ pre-built processors
- Built-in data provenance
- Guaranteed delivery
- Throttling and Back pressure

# Cloudera Flow Management

Ingest and manage data from edge-to-cloud using a no-code interface



Advanced tooling to industrialize flow development  
(Flow Development Life Cycle)

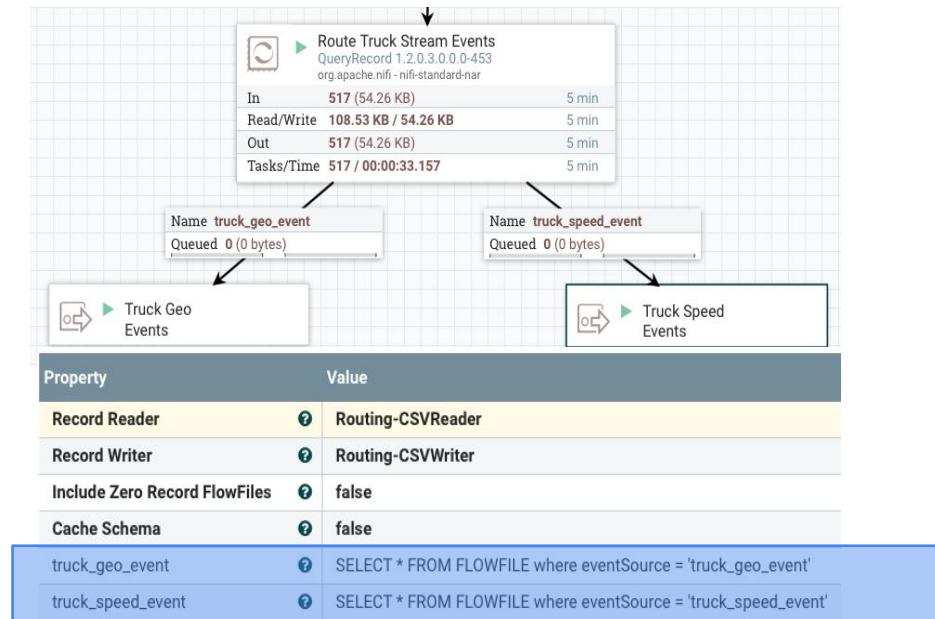


- Over 300 pre-built processors
- Easy to build your own processors
- Parse, enrich & apply schema
- Filter, Split, Merge & Route

- Throttle & Backpressure
- Guaranteed delivery
- Full data provenance
- Eco-system integration

# SQL BASED ROUTING WITH NiFi's QueryRecord Processor

- **QueryRecord Processor**- Executes a SQL statement against records and writes the results to the flow file content.
- **CSVReader**: Looking up schema from SR, it will converts CSV Records into ProcessRecords
- **SQL execution via Apache Calcite**: execute configured SQL against the ProcessRecords for routing
- **CSVRecordSetWriter**: Converts the result of the query from Process records into CSV for the for the flow file content



Do routing(routing geo and speed streams) using standard SQL as opposed to complex regular expressions.

# Key Differentiators

**Stream to Cloud** – Extend the same on-premises streaming capabilities to the cloud with full support for multi-cloud and hybrid cloud models



**400+ pre-built processors** – Only product to offer such comprehensive connectivity to a wide range of data sources from edge to cloud



**Democratize access to real-time data** – Enable data analysts and other personas to quickly build streaming applications with just SQL



**Enterprise-Grade Security & Governance** – Deploy your streaming applications with confidence and trust with Cloudera SDX offering unified security and governance across the entire platform

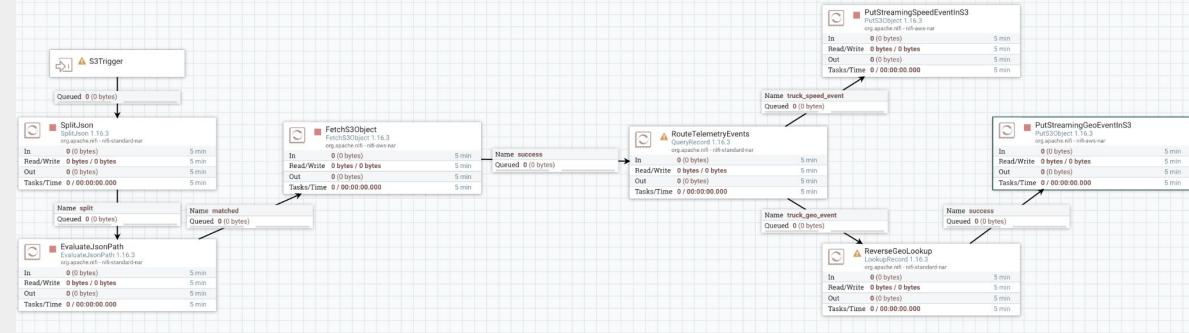


**Comprehensive streaming platform** – Only vendor to offer a open and comprehensive streaming platform for real-time data ingestion and processing to produce prescriptive and predictive analytics



# Development & Runtime of DataFlow Functions

**Step1. Develop** functions on local workstation or in CDP Public Cloud using **no-code**, UI designer



**Step 2. Run** functions on serverless compute services in AWS, Azure & GCP



AWS Lambda



Azure Functions



Google Cloud Functions

# DataFlow Functions Use Cases

Trigger Based, Batch, Scheduled and Microservice Use Cases

## Serverless Trigger-Based File Processing Pipeline

Develop & run data processing pipelines when files are created or updated in any of the cloud object stores

**Example:** When a photo is uploaded to object storage, a data flow is triggered which runs image resizing code and delivers resized image to different locations.

## Serverless Workflows / Orchestration

Chain different low-code functions to build complex workflows

**Example:** Automate the handling of support tickets in a call center or orchestrate data movement across different cloud services.

## Serverless Scheduled Tasks

Develop and run scheduled tasks without any code on pre-defined timed intervals

**Example:** Offload an external database running on-premises into the cloud once a day every morning at 4:00 a.m.

## Serverless Microservices

Build and deploy serverless independent modules that power your applications microservices architecture

**Example:** Event-driven functions for easy communication between thousands of decoupled services that power a ride-sharing application.

## Serverless Web APIs

Easily build endpoints for your web applications with HTTP APIs without any code using DFF and any of the cloud providers' function triggers

**Example:** Build high performant, scalable web applications across multiple data centers.

## Serverless Customized Triggers

With the DFF State feature, build flows to create customized triggers allowing access to on-premises or external services

**Example:** Near real time offloading of files from a remote SFTP server.

# Flow Catalog

- Central repository for flow definitions
- Import existing NiFi flows
- Manage flow definitions
- Initiate flow deployments

The screenshot shows the Cloudera DataFlow interface. On the left is a dark sidebar with a purple hexagonal icon, the text 'CLOUDERA DataFlow', and three menu items: 'Dashboard', 'Catalog' (which is selected and highlighted in purple), and 'Environments'. The main area is titled 'Flow Catalog' and contains a list of flow definitions. The list is ordered by name, with 'Covid Data Stream' at the top. Other listed flows include 'CovidIDBroker', 'drew\_kafka-hdfs-querydb-kudu', 'drew\_kafka\_to\_hdfs', 'Employees Data', 'Empty Dev Flow', and 'Generate Flow File Log'. To the right of the list is a detailed view for the 'Covid Data Stream' flow. It includes a 'FLOW DESCRIPTION' section stating 'This flow reads covid data from several sources and writes it to CDP', a checkbox for 'Only show deployed versions', and a table showing 'Version' (13) and 'Deployments' (0). Below this is a 'Deploy New Flow →' button. At the bottom of the detailed view is a 'LAST UPDATE' section with the date '2021-02-02 14:25 PST by Michael Kohs' and the note 'This version includes the latest fixes'. There are also sections for versions 12 and 11, each with deployment counts of 0.

# ReadyFlows

- Cloudera provided flow definitions
- Cover most common data flow use cases
- Can be deployed and adjusted as needed
- Made available through docs during Tech Preview

Cloudera Docs / DataFlow master ▾ (test • Technical Preview)

Search Document

**Cloudera DataFlow**

**Release Notes**

Release Notes

**Concepts**

Overview

**Planning**

AWS Resource Planning

NiFi Flow Limitations

**Getting Started**

Quick Start

Out of Box Flow Definitions

Import a flow definition

**Flow definition for ingesting data into a Kafka topic**

Flow definition for ingesting data into Amazon S3 Buckets

**How To: Environments**

Enabling a DataFlow Environment

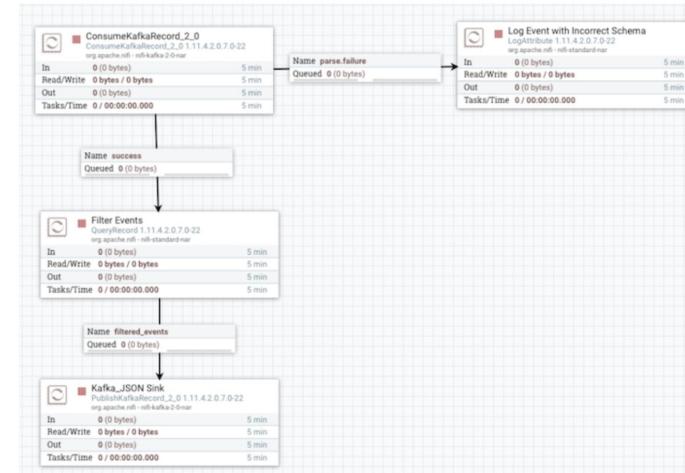
Managing a DataFlow Environment

## OUT OF BOX FLOW DEFINITIONS

Flow definition for ingesting data into a Kafka topic

### Example

The resulting flow will look similar to the following, on your NiFi canvas.



# Deployment Wizard

- Turns flow definitions into flow deployments
- Guides users through providing required configuration
- Pick from pre-defined NiFi node sizes
- Define KPIs for the deployment

## Start Deployment Wizard

### New Deployment

#### Select the target environment

ⓘ Sensitive data never leaves the environment. Changing the environment after this step requires restarting the deployment process.

#### Selected Flow Definition

NAME	VERSION
Machine Data To Warehouse	2

#### Target Environment

aws dataflow-demo	60% (3 of 5)
-------------------	--------------

## Configure Sizing & Scaling

- ✓ Overview
- ✓ Flow Parameters
- ✓ Sizing & Scaling
- ⓘ Key Performance Indicators
- ⓘ Review

### Sizing & Scaling

Select the NiFi node size and the number of nodes provisioned for your flow.

#### NiFi Node Sizing ⓘ



#### Number of NiFi Nodes

Auto Scaling ⓘ

Enabled

Min. Nodes: 1 - Max. Nodes: 32

## Provide Parameters

### Flow Parameters

Data entered here never leaves the environment in your cloud account. Provide parameter values directly in the text input or upload a file for parameters that expect a file.

#### MachineData

##### AWS Credential File

Enter parameter values.

Select File

Drop file or browse

#### CDP Truststore

Enter parameter values.

Select File

Drop file or browse

#### CDPSchemaRegistry

https://dataflow-streams-master0.dataflow.xcu2-8y8x.dev.cldr.work:7790/api/v1

## Define KPIs

### Key Performance Indicators

Set up KPIs to track specific performance metrics of a deployed flow. Click and drag to reorder how they are displayed.

#### Entire Flow

##### METRIC TO TRACK

##### Data In

ALERT SET  
Notify if less than 150 KB/sec, for at least 30 seconds.

#### Processor: Write to S3 using HDFS proc

##### METRIC TO TRACK

##### Bytes Sent

ALERT SET  
No alert set

Add New KPI

# Key Performance Indicators

- Visibility into flow deployments
- Track high level flow performance
- Track in-depth NiFi component metrics
- Defined in Deployment Wizard
- Monitoring & Alerts in Deployment Details

## KPI Definition in Deployment Wizard

Key Performance Indicators

Set up KPIs to track specific performance metrics of a deployed flow. Click and drag to reorder how they are displayed.

Entire Flow

METRIC TO TRACK  
Data In

ALERT SET  
Notify if less than 150 KB/sec, for at least 30 seconds.

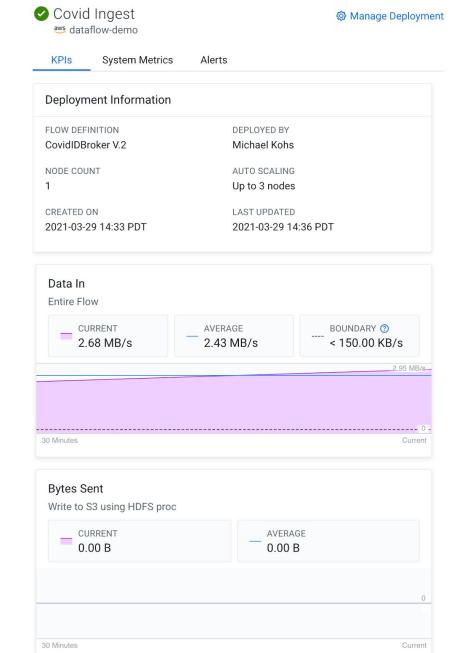
Processor: Write to S3 using HDFS proc

METRIC TO TRACK  
Bytes Sent

ALERT SET  
No alert set

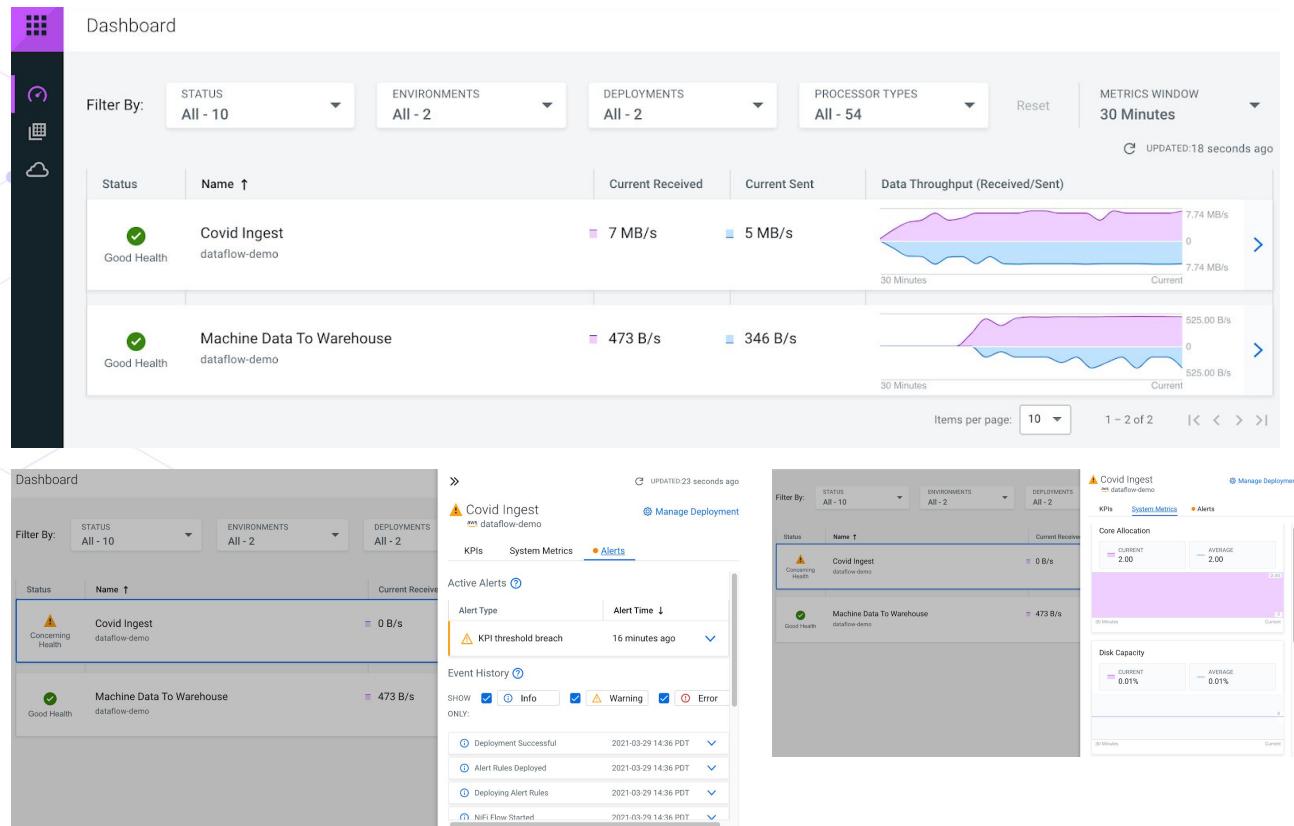
Add New KPI

## KPI Monitoring



# Dashboard

- Central Monitoring View
- Monitors flow deployments across CDP environments
- Monitors flow deployment health & performance
- Drill into flow deployment to monitor system metrics and deployment events



# Data Flow Design for Everyone

- Cloud-native data flow development
- Developers get their own sandbox
- Start developing flows without installing NiFi
- Redesigned visual canvas
- Optimized interaction patterns
- Integration into CDF-PC Catalog for versioning

The screenshot shows the Cloudera Data Flow interface. On the left is a dark sidebar with the Cloudera logo and navigation links: Dashboard, Catalog, ReadyFlow Gallery, Flow Design (which is selected), Functions, Environments, Get Started, Help, and Stephen Hawking. At the bottom of the sidebar are the version 'v2.0.0' and a back arrow icon. The main area is titled 'Flow Design / [WorkspaceName] / [FlowDefinitionName]'. It features a visual canvas with a single node highlighted in orange. A tooltip for this node displays the following information:

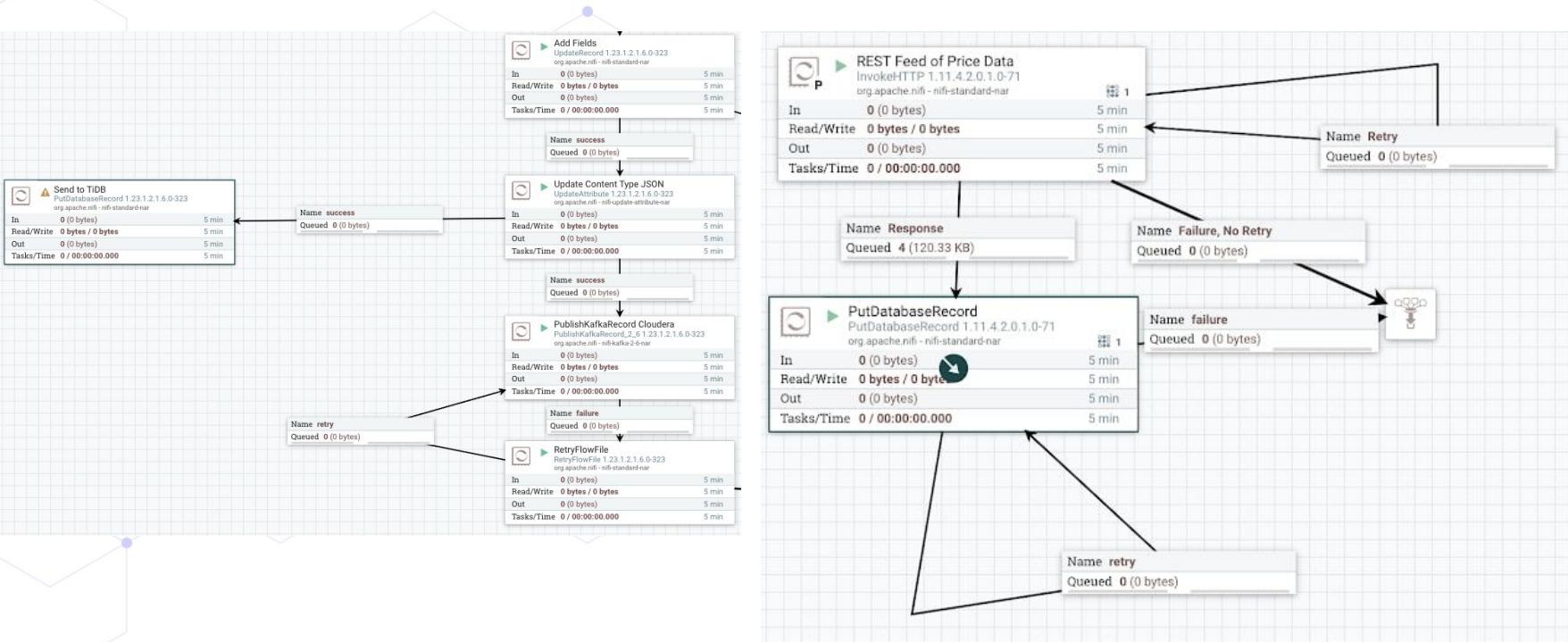
[ProcessorName]	[ProcessorType] [Version#]
IN	19 (14.16 MB)
READ/WRITE	4.88 MB/4.88 MB
OUT	0 (0 bytes)
TASKS	29/00:00.00.123

A metrics panel is overlaid on the canvas, showing a chart with two data series: 'IN' and 'OUT'. The right side of the screen shows a configuration panel with tabs for 'Configuration' (selected) and 'Metrics'. The configuration section lists various parameters with their current values:

* Region	US West (Oregon)
Access Key ID	No value set
Secret Access Key	No value set
Record Writer	No value set
* Minimum Object Age	0 sec
Listing Batch Size	100
* Write Object Tags	False
* Write User Metadata	No value set
Credentials File	Empty string set
AWS Credentials Provider	No value set
* Communications Timeout	30 sec
SSL Context Service	No value set
Endpoint Override URL	No value set

At the bottom of the configuration panel are 'Apply Changes' and 'Discard Changes' buttons.

# Data Distribution and Sharing with TiDB

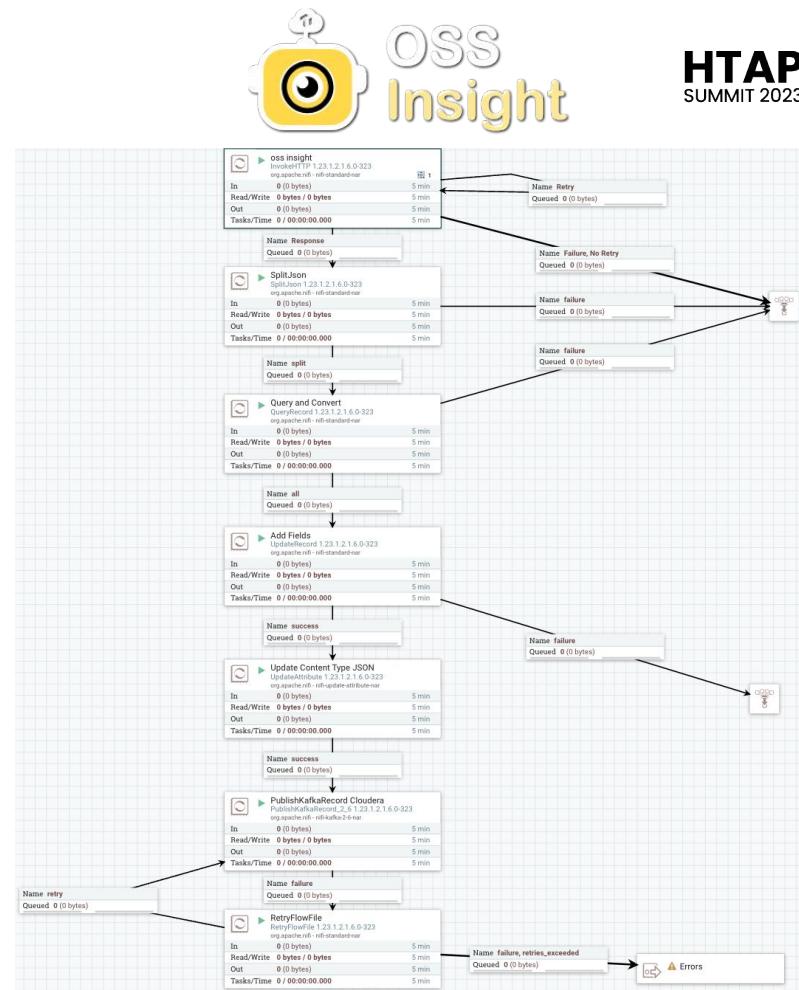


<https://docs.pingcap.com/tidb/dev/mysql-compatibility>

# NiFi Ingesting REST API

- NiFi consumes stream (cdc, rest, sensors)
- Distributes real-time to
- Kafka and MySQL at same time
- Flink SQL consumes from Kafka
- TiDB CDC → Kafka

<https://ossinsight.io/docs/api>



# Apache Kafka



## Data Distribution with Apache Kafka

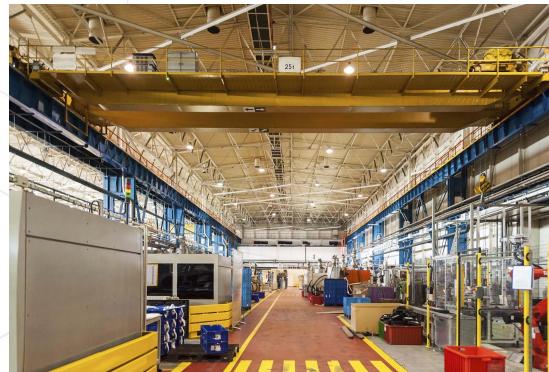
**HTAP**  
SUMMIT 2023

# Apache Kudu



# Why Kudu?

A simultaneous combination of sequential and random reads and writes



## Time Series Data

Can you insert time series data in real time? How long does it take to prepare it for analysis? Can you get results and act fast enough to change outcomes?

## Machine Data Analytics

Can you handle large volumes of machine-generated data? Do you have the tools to identify problems or threats? Can your system do machine learning?

## Online Reporting

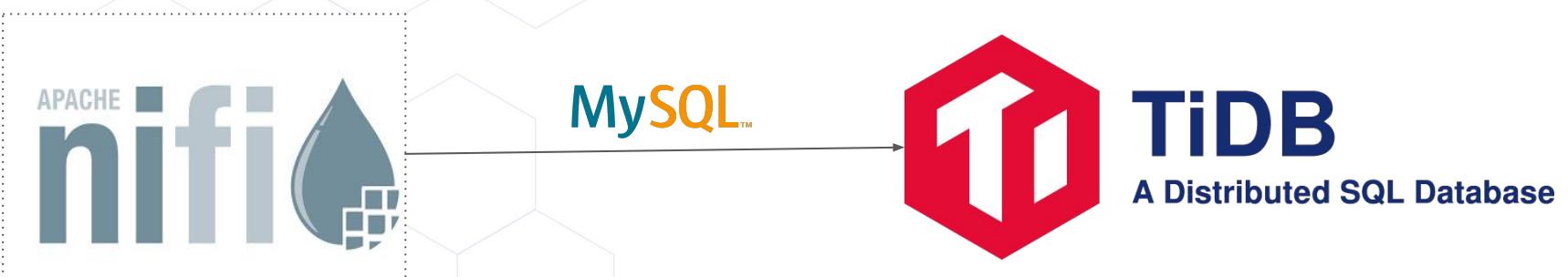
**How fast can you add data to your data store? Are you trading off the ability to do broad analytics for the ability to make updates? Are you retaining only part of your data?**

# HTAP Options - Apache Kudu

Online Transactional Processing (OLTP) and Online Analytical Processing (OLAP)

[https://hpi.de/fileadmin/user\\_upload/hpi/navigation/10\\_forschung/20\\_future\\_soc\\_lab/Poster/2019-1/Tozun\\_FSOC-Poster\\_2019\\_150443.pdf](https://hpi.de/fileadmin/user_upload/hpi/navigation/10_forschung/20_future_soc_lab/Poster/2019-1/Tozun_FSOC-Poster_2019_150443.pdf)

# HTAP Options - TiDB



# Apache Flink SQL



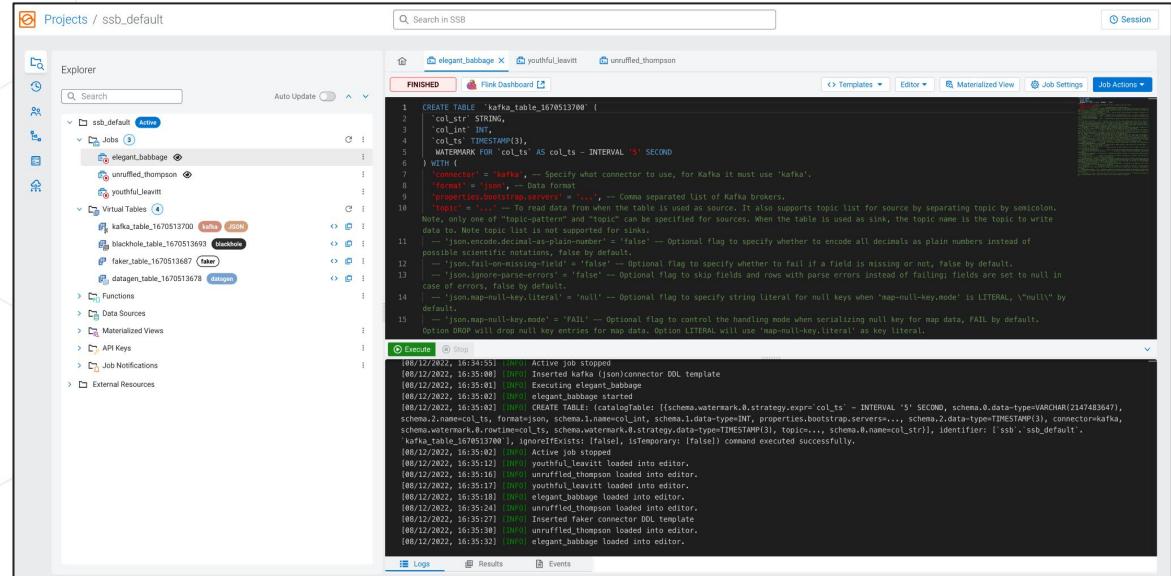
# SQL STREAM BUILDER (CLOUDERA SSB)

SQL STREAM BUILDER allows developers, analysts, and data scientists to write streaming applications with industry standard SQL.

No Java or Scala code development required.

Simplifies access to data in Kafka & Flink. Connectors to batch data in HDFS, Kudu, Hive, S3, JDBC, CDC and more

Enrich streaming data with batch data in a single tool



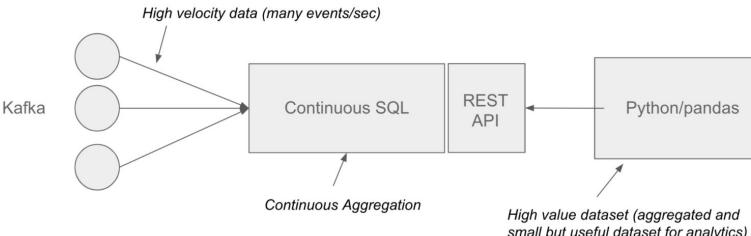
```

CREATE TABLE `kafka_table_1670513700` (
  `col_str` STRING,
  `col_int` INT,
  `col_ts` TIMESTAMP(3),
  WATERMARK FOR `col_ts` AS `col_ts` - INTERVAL '5' SECOND
) WITH (
  'connector' = 'kafka',
  'format' = 'json',
  'topic' = 'elegant_babbage',
  'properties.bootstrap.servers' = '... -- Comma separated list of Kafka brokers',
  'topic' = '... -- To read record from where the table is used as source. It also supports topic list for source by separating topic by semicolon,
  Note, only one of "topic-pattern" and "topic" can be specified for sources. When the table is used as sink, the topic name is the topic to write
  data to. Note topic list is not supported for sinks,
  'json.codecs' = 'org.apache.kafka.connect.json.JsonDecoder',
  'json.ignore-parse-errors' = 'true' -- Optional flag to specify whether to encode all decimals as plain numbers instead of
  primitive representations such as JSON,
  'value.converter' = 'org.apache.kafka.connect.json.JsonConverter',
  'key.converter' = 'org.apache.kafka.connect.json.JsonConverter',
  'value.serializer' = 'org.apache.kafka.common.serialization.StringSerializer',
  'key.serializer' = 'org.apache.kafka.common.serialization.StringSerializer',
  'partition.value.strategy' = 'random' -- Optional flag to specify whether to fail if a field is missing or not, false by default,
  'partition.key.strategy' = 'random' -- Optional flag to skip fields and rows with parse errors instead of failing; fields are set to null in
  case of error, false by default,
  'ignore.fail-on-missing-field' = 'false' -- Optional flag to specify whether to fail if a field is missing or not, false by default,
  'ignore.parse-errors' = 'false' -- Optional flag to skip fields and rows with parse errors instead of failing; fields are set to null in
  case of error, false by default,
  'json.map-null-key.literal' = 'null' -- Optional flag to specify string literal for null keys when 'map-null-key.mode' is LITERAL, '\"null\"'
  by default, 'null' -- 'json.map-null-key.mode' = 'FAIL' -- Optional flag to control the handling mode when serializing null key for map data, FAIL by default,
  Option DROP will drop null key entries for map data. Option LITERAL will use 'map-null-key.literal' as key literal.
)
  
```

The screenshot shows the Cloudera SQL Stream Builder interface. The left sidebar (Explorer) lists projects, jobs, virtual tables, functions, data sources, materialized views, API keys, job notifications, and external resources. The main area (Editor) displays a complex SQL query for creating a Kafka table named 'kafka\_table\_1670513700'. The query includes columns for 'col\_str', 'col\_int', and 'col\_ts' with a timestamp watermark. It specifies a Kafka connector with a topic of 'elegant\_babbage', setting 'topic' to 'elegant\_babbage' and 'topic' to '...'. Other properties include 'bootstrap.servers', 'key.converter', 'value.converter', and 'partition.value.strategy'. The right side of the interface shows a Flink Dashboard with tabs for 'FINISHED', 'Flink Dashboard', and 'Session'. The 'Session' tab is active, showing logs, results, and events. The logs tab displays a history of command executions, including the creation of the table and loading of connectors like 'elegant\_babbage', 'unruffled\_thompson', and 'youthful\_leavitt'.

# SSB MATERIALIZED VIEWS

Key Takeaway; MV's allow data scientist, analyst and developers consume data from the firehose



```
SELECT userid,
       max(amount) as max_amount,
       sum(amount) as sum_amount,
       count(*) as thecount,
       tumble_end(eventTimestamp, interval '5' second) as ts
  FROM authorizations
 GROUP BY userid, tumble(eventTimestamp, interval '5' second)
 HAVING count(*) > 1
```

```
[90]: import pandas as pd
[91]: mv = "https://xxxxxxxxxx"
[92]: df = pd.read_json(mv)
[93]: len(df.keys())
[93]: 5
[95]: df['ts'] = pd.to_datetime(df['ts'])
[97]: df.dtypes
[97]: max_amount          int64
sum_amount             int64
thecount              int64
ts                     datetime64[ns]
userid                 int64
dtype: object
[98]: df.set_index('userid').sort_values(by=['thecount'], ascending=False).head()
[98]:
      max_amount  sum_amount  thecount      ts
userid
    787      34911     57304     10 2020-06-16 19:52:15
    744      77407     95407      9 2020-06-16 19:52:15
    78      86761     330397      9 2020-06-16 19:52:15
    541      78762     282682      8 2020-06-16 19:52:15
    926      85636     129728      8 2020-06-16 19:52:15
```

## Infer Tables from Kafka Topics with JSON or Avro

### Kafka Table

Table Name \*

Kafka Cluster \*

Data Format \*

Topic Name \*

Schema Definition   Event Time   Data Transformation   Properties   Deserialization

```
1 {  
2   "type": "record",  
3   "name": "inferredSchema",  
4   "fields": [  
5     {  
6       "name": "bssid",  
7       "type": "string",  
8       "doc": "Type inferred from '\"\\\"'"  
9     },  
10    {  
11      "name": "channel",  
12      "type": "string",  
13      "doc": "Type inferred from '\"52\\\"'"  
14    },  
15    {  
16      "name": "channel_band",  
17      "type": "string",  
18      "doc": "Type inferred from '\"5\\\"'"  
19    },  
20    {  
21      "name": "channel_width",  
22      "type": "string",  
23      "doc": "Type inferred from '\"80\\\"'"  
24    },  
25    {  
26      "name": "country_code",  
27      "type": "string",  
28      "doc": "Type inferred from '\"\\\"'"  
29    },  
30    {  
31      "name": "interface",  
32      "type": "string",  
33      "doc": "Type inferred from '\"en0\\\"'"  
34    },  
35  }  
}
```



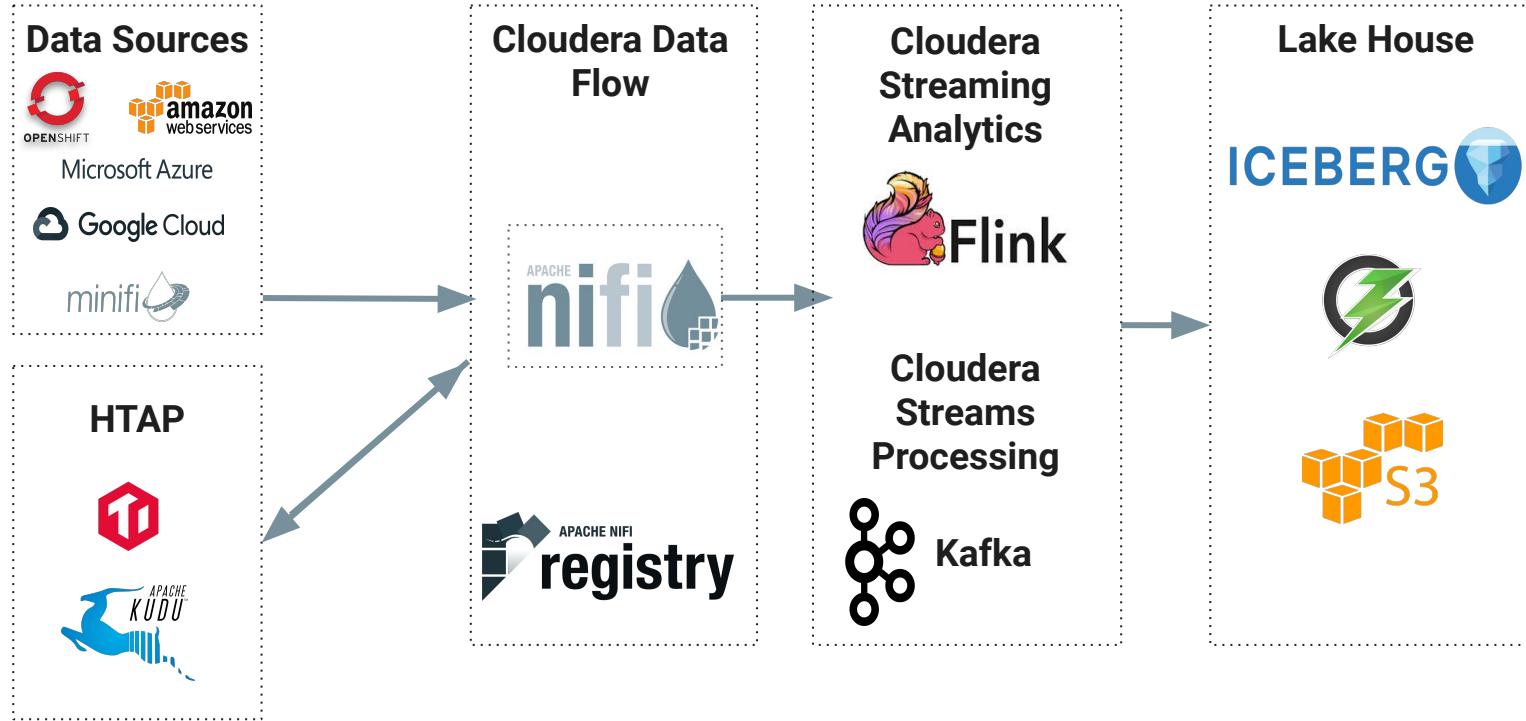
Schema is valid   [Detect Schema](#)

[Cancel](#) [Create and Review](#)

# Demos



# INGEST OF ALL DATA



# LLM USE CASE



Unstructured file types



Structured Sources



GURU

Other enterprise data

## Data in Motion on Cloudera Data Platform (CDP)

Capture, process &  
distribute any data,  
anywhere



AI Model



Vector DB



Applications/API's



Streams



Materialized Views

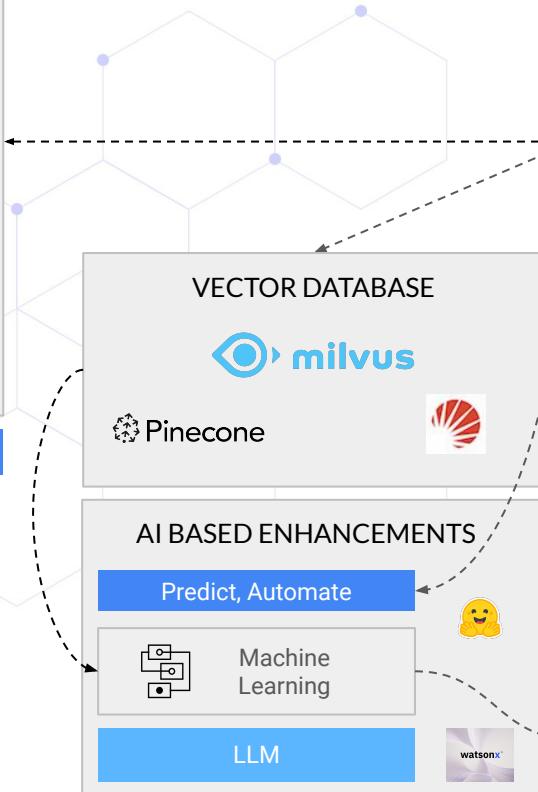


Open Data Lakehouse

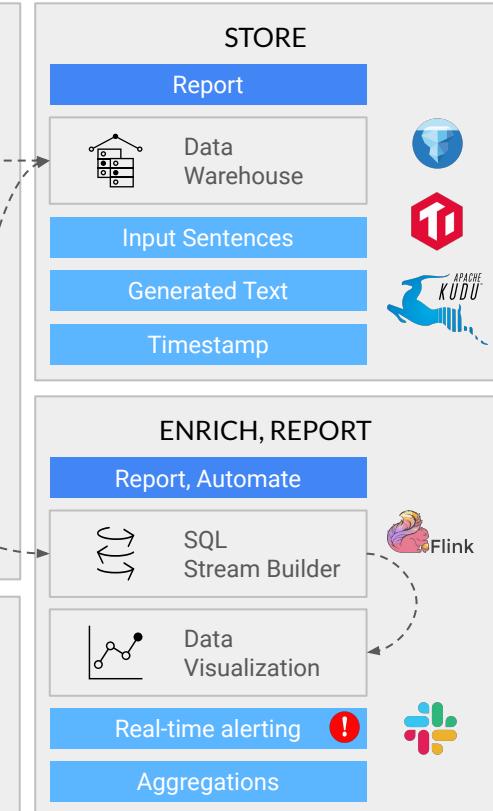
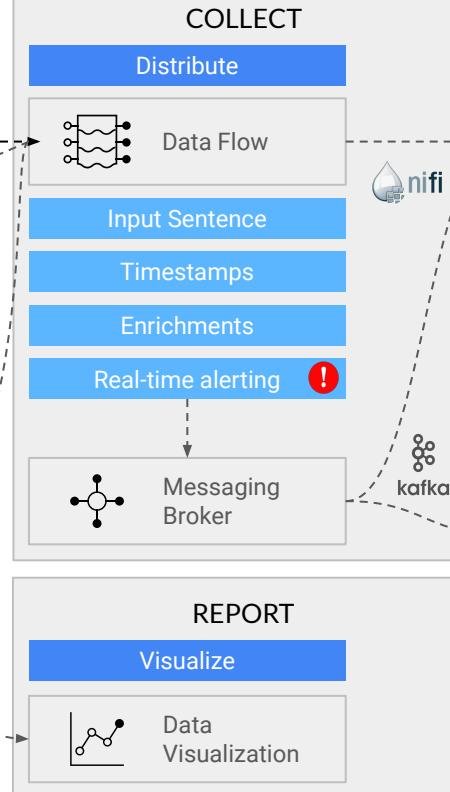
## INTERACT

- Live Q&A
- Travel Advisories
- Weather Reports
- Documents
- Social Media
- Internal Data
- Github Data
- REST API

Collect



## HYBRID CLOUD



RUN AT HOME



# CSP Community Edition

- Kafka, KConnect, SMM, SR, Flink, and SSB in Docker
- Runs in Docker
- Try new features quickly
- Develop applications locally



- Docker compose file of CSP to run from command line w/o any dependencies, including Flink, SQL Stream Builder, Kafka, Kafka Connect, Streams Messaging Manager and Schema Registry
  - \$> docker compose up
- Licensed under the Cloudera Community License
- **Unsupported**
- Community Group Hub for CSP
- Find it on [docs.cloudera.com](https://docs.cloudera.com) under Applications



CSP Community Edition

A readily available, dockerized deployment of Apache Kafka and Apache Flink that allows you to test the features and capabilities of Cloudera Stream Processing.

[Learn More](#)

# Open Source Edition



- Apache NiFi in Docker
- Runs in Docker
- Try new features quickly
- Develop applications locally

- Docker NiFi

- ```
docker run --name nifi -p 8443:8443 -d -e SINGLE_USER_CREDENTIALS_USERNAME=admin -e SINGLE_USER_CREDENTIALS_PASSWORD=ctsBtRBKHRAx69EqUghvvgEvjnaLjFEB apache/nifi:latest
```

- Licensed under the ASF License
- **Unsupported**



<https://hub.docker.com/r/apache/nifi>

# Thank You

