



Mastering Data Streaming Pipelines

Tim Spann
Principal Developer Advocate

09-May-2023



CLOUDERA



CLOUDERA

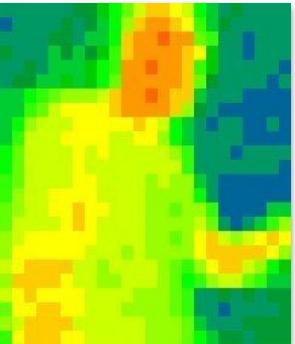


EDGE
2AI

CLOUDERA



FLaNK Stack



Tim Spann

@PaasDev // Blog: www.datainmotion.dev

Principal Developer Advocate.

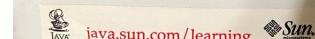
Princeton Future of Data Meetup.

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

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

<https://github.com/tspannhw>

Apache NiFi x Apache Kafka x Apache Flink x Java



FLiP Stack Weekly



<https://bit.ly/32dAJft>

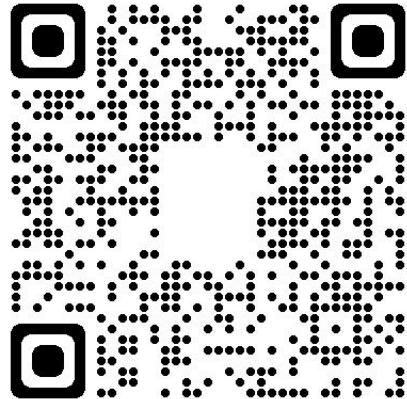


This week in Apache NiFi, Apache Flink, Apache Kafka, Apache Spark, Apache Iceberg, Python, Java and Open Source friends.



Meet the NiFi Committers

Wednesday, May 3, 2023
10am-11amPT/12pm-1pmCT/1pm-2pmET



Hosts



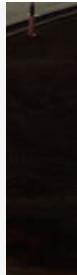
Joe Witt, VP of Engineering, Cloudera



Mark Payne, Principal Engineer, Cloudera



Matt Gilman, Principal Engineer, Cloudera



See details here: <https://lnkd.in/qCUVSvM6> (Passcode: a.7eZl5N).

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

FREE LEARNING ENVIRONMENT

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 under Applications

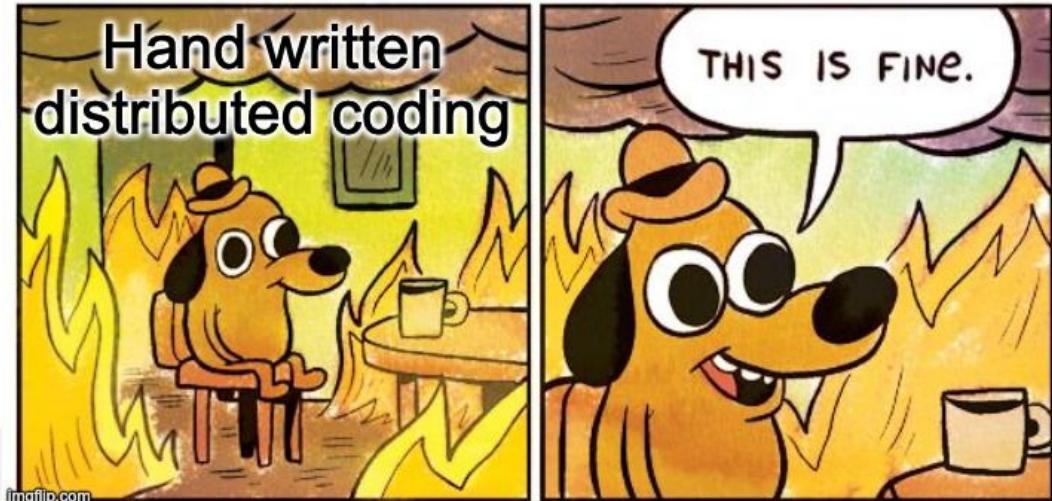


CSP Community Edition

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](#)

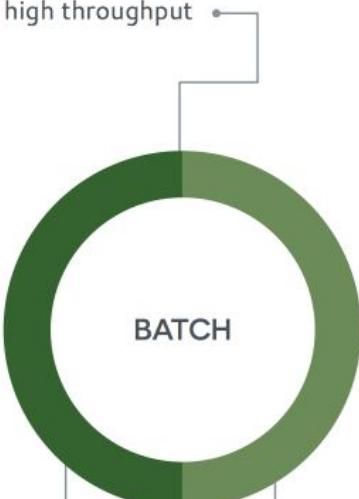
STREAMING



WHAT IS REAL-TIME?

> 1 HOUR

high throughput



10 MS – 1 SEC

approximate



< 500 MS

latency sensitive



< 1 MS

low latency



adhoc queries

monthly active users relevance for ads

ad impressions count hash tag trends

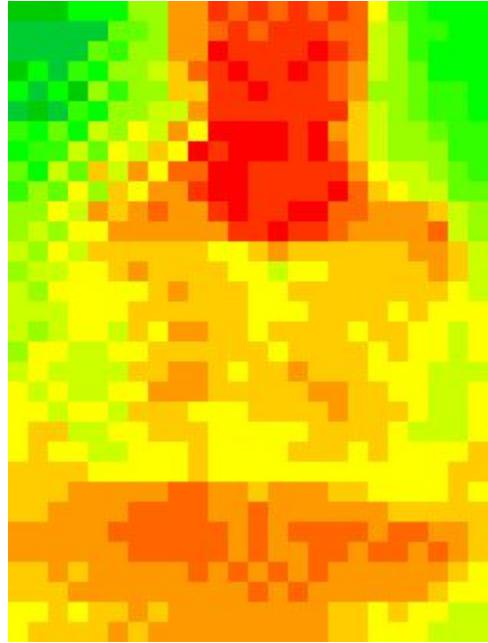
deterministic workflows

fanout Tweets search for Tweets

Financial Trading

BUILDING REAL-TIME REQUIRES A TEAM



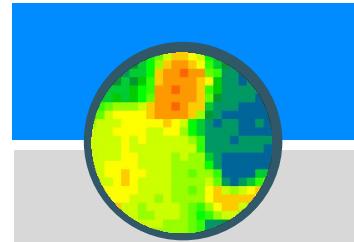


imgflip.com

JAKE-CLARK.TUMBLR

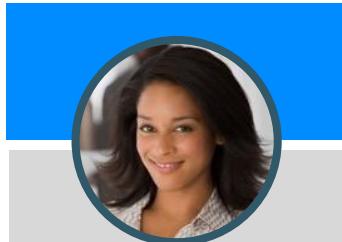
Streaming for Java Developers

Multiple users, frameworks, languages, devices, data sources & clusters



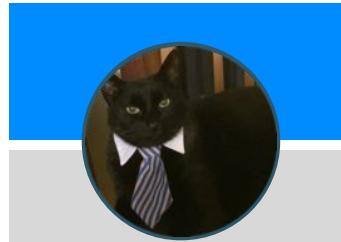
STREAMING ENGINEER

- Coding skills in Python, Java
- Experience with Apache Kafka or Pulsar
- Knowledge of database query languages such as SQL
- Knowledge of tools such as Apache Flink, Apache Spark and Apache NiFi



JAVA DEVELOPER

- Frameworks like Spring, Quarkus and micronaut
- Relational Databases, SQL
- Cloud
- Dev and Build Tools



CAT

- Expert in ETL (Eating, Ties and Laziness)
- Deep SME in Buzzwords
- No Coding Skills
- R&D into Lasers



AI

- Will Drive your Car?
- Will Fix Your Code?
- Will Beat You At Q-Bert
- Will Write my Next Talk

APACHE SPARK

APACHE SPARK

Data Engineering at Scale

- Multi-language support with Java, Scala, Python and R
- Batch and Microbatch
- Strong SQL support
- Machine Learning
- Jupyter and Apache Zeppelin notebook support



APACHE ICEBERG

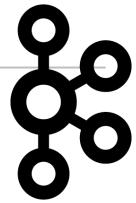
A Flexible, Performant & Scalable Table Format

- Donated by **Netflix** to the Apache Foundation in 2018
- Flexibility
 - Hidden partitioning
 - Full schema evolution
- Data Warehouse Operations
 - Atomic Consistent Isolated Durable (ACID) Transactions
 - Time travel and rollback
- Supports best in class SQL performance
 - High performance at Petabyte scale



APACHE KAFKA





Yes, Franz, It's Kafka

Let's do a metamorphosis on your data. Don't fear changing data.

You don't need to be a brilliant writer to stream data.



Franz Kafka was a German-speaking Bohemian novelist and short-story writer, widely regarded as one of the major figures of 20th-century literature. His work fuses elements of realism and the fantastic.

[Wikipedia](#)

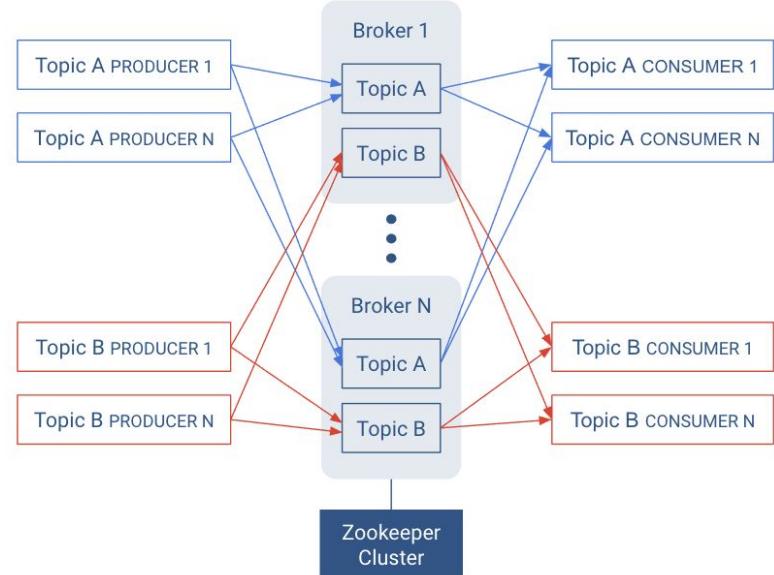


STREAMS MESSAGING WITH KAFKA



WriteToKafka		
PublishKafka2RecordCDP 1.0.0.2.2.2.0-127 com.cloudera - nifi-cdf-kafka-2-nar		
In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

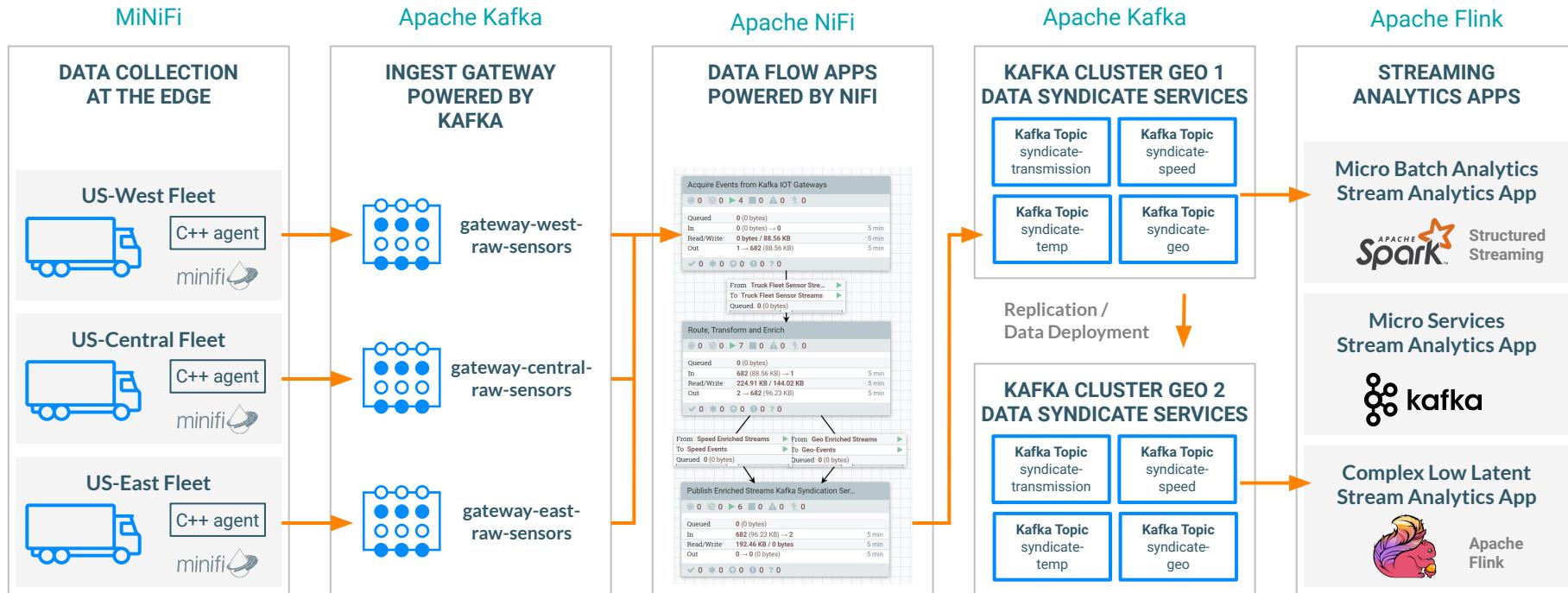
- Highly reliable distributed messaging system.
- Decouple applications, enables many-to-many patterns.
- Publish-Subscribe semantics.
- Horizontal scalability.
- Efficient implementation to operate at speed with big data volumes.
- Organized by topic to support several use cases.



What is Can You Do With Apache Kafka?

- Web site activity: track page views, searches, etc. in real time
- Events & log aggregation: particularly in distributed systems where messages come from multiple sources
- Monitoring and metrics: aggregate statistics from distributed applications and build a dashboard application
- Stream processing: process raw data, clean it up, and forward it on to another topic or messaging system
- Real-time data ingestion: fast processing of a very large volume of messages

Apache Kafka



APACHE FLINK

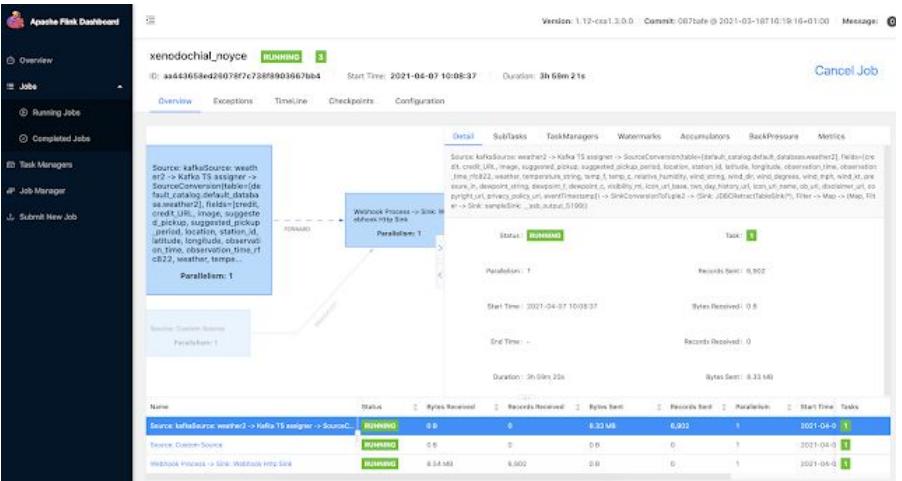


Am I streaming yet?

Flink SQL



- Streaming Analytics
 - Continuous SQL
 - Continuous ETL
 - Complex Event Processing
 - Standard SQL Powered by Apache Calcite



<https://www.datainmotion.dev/2021/04/cloudera-sql-stream-builder-ssb-updated.html>

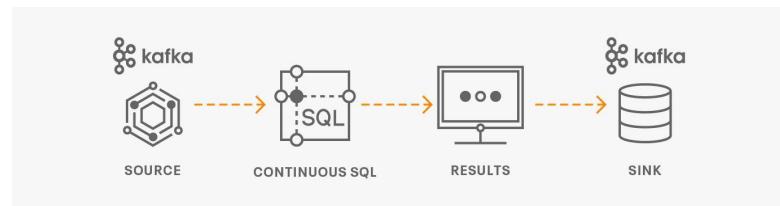
CONTINUOUS SQL

- SSB is a Continuous SQL engine
- It's SQL, but a slightly different mental model, but with big implications

Traditional Parse/Execute/Fetch model



Continuous SQL Model



Hint: The query is boundless and never finishes, and time matters

AKA: `SELECT * FROM foo WHERE 1=0 -- will run forever`

SQL STREAM BUILDER (SSB)

Democratize access to real-time data with just SQL

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

No Java or Scala code development required.

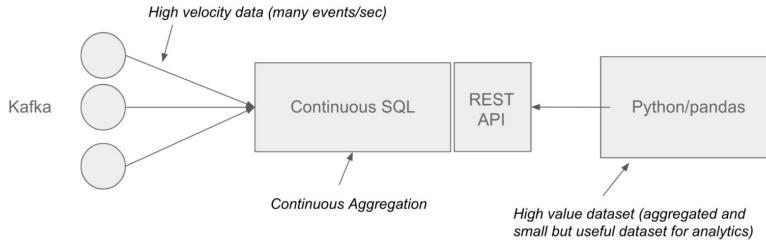
Simplifies access to data in Kafka & Flink. Connectors to batch data in HDFS, Kudu, Hive, S3, JDBC, CDC and more

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', -- Specify what connector to use, for Kafka it must use 'kafka'.
  'format' = 'json', -- Topic name to read from.
  'topic' = 'elegant_babbage', -- Comma separated list of Kafka brokers.
  'properties.bootstrap.servers' = '...', -- Optional flag to specify whether to encode all decimals as plain numbers instead of Note, only one of 'topic-pattern' and 'topic' can be specified for sources. It also supports topic list for source by separating topic by semicolon.
  'json.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 errors, false by default.
  'json.fail-on-missing-field' = 'false' -- Optional flag to fail if a field is missing or not, false by default.
  'json.ignore-error-on-malformed' = 'true' -- Optional flag to ignore errors on malformed JSON.
  '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.
  '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.
)
Note, only one of 'topic-pattern' and 'topic' can be specified for sinks.
-- 'json.codec.decimals-as-plain-number' = 'false' -- Optional flag to specify whether to encode all decimals as plain numbers instead of Note, only one of 'topic-pattern' and 'topic' can be specified for sinks.
-- 'json.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 errors, false by default.
-- 'json.fail-on-missing-field' = 'false' -- Optional flag to fail if a field is missing or not, false by default.
-- 'json.ignore-error-on-malformed' = 'true' -- Optional flag to ignore errors on malformed JSON.
-- '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.
-- 'map-null-key.mode' = 'FAIL' -- Optional flag to control the handling mode when serializing null key for map data, FAIL by default.
Option DROPP will drop null key entries for map data. Option LITERAL will use 'map-null-key.literal' as key literal.
[08/12/2022, 16:34:55] [INFO] Active job stopped
[08/12/2022, 16:35:00] [INFO] Inserted kafka (json)connector DDL template
[08/12/2022, 16:35:01] [INFO] Executing elegant_babbage
[08/12/2022, 16:35:02] [INFO] CREATE TABLE: (schema.waternark.v.strategy_exp.col_ts` - INTERVAL '5' SECOND, schema.0.data-type=VARCHAR(2147483647), schema.0.name=col_ts, format=json, schema.0.type=col_ts, schema.1.data-type=INT, properties.bootstrap.servers=..., schema.2.data-type=TIMESTAMP(3), connector=kafka, schema.waternark.v.routine=col_ts, schema.waternark.v.strategy,data-type=TIMESTAMP(3), topic<...>, schema.0.name=col_str), identifier: l(ssb(ssb_default).kafka_table_1670513700), ignoreIfExists: (false), isTemporary: (false)) command executed successfully.
[08/12/2022, 16:35:02] [INFO] Active job started
[08/12/2022, 16:35:12] [INFO] yogurt loaded into editor.
[08/12/2022, 16:35:13] [INFO] unruffled_thompson loaded into editor.
[08/12/2022, 16:35:17] [INFO] yogurt.loaded into editor.
[08/12/2022, 16:35:18] [INFO] yogurt.loaded into editor.
[08/12/2022, 16:35:24] [INFO] unruffled_thompson loaded into editor.
[08/12/2022, 16:35:27] [INFO] Inserted faker connector DDL template
[08/12/2022, 16:35:30] [INFO] unruffled_thompson loaded into editor.
[08/12/2022, 16:35:32] [INFO] elegant_babbage loaded into editor.
```

SSB MATERIALIZED VIEWS

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

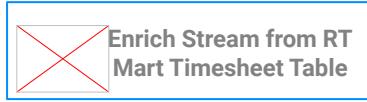
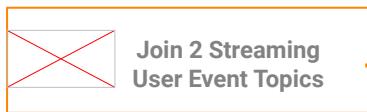


```
SELECT userid,
       max(amount) as max_amount,
       sum(amount) as sum_amount,
       count(*) as thecount,
       tumble_end(eventTimestamp, interval '5' second) as ts
  FROM authorizations
 GROUP BY userid, tumble(eventTimestamp, interval '5' second)
 HAVING count(*) > 1
```



```
[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      88761     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
```

Streaming ETL Data Pipeline Made Simple with SQL StreamBuilder

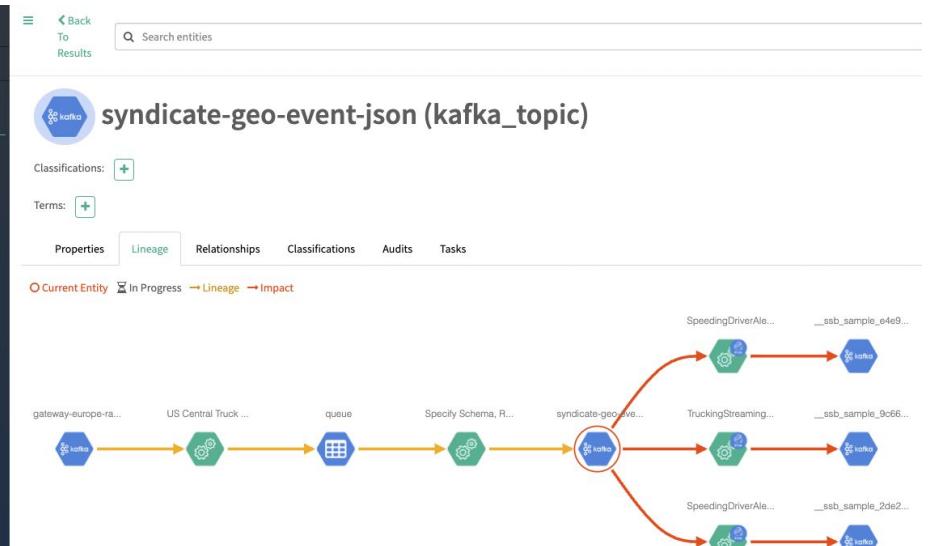
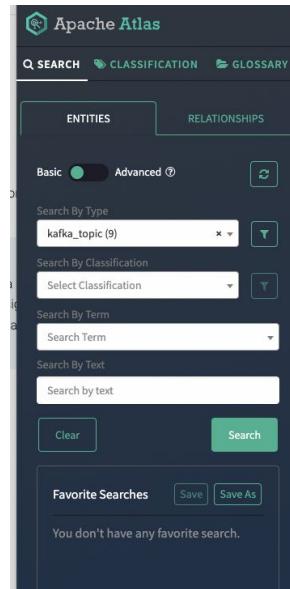


```
1 SELECT
2     geo_event.eventTimestamp, geo_event.driverId, geo_event.eventTime, geo_event.eventSource,
3     geo_event.truckId, geo_event.driverName, geo_event.routeId, geo_event.route, geo_event.eventType,
4     geo_event.latitude, geo_event.longitude, geo_event.correlationId, geo_event.geoAddress,
5     speed_event.speed,
6     driver.certified, driver.wage_plan,
7     timesheet.hours_logged, timesheet.miles_logged
8
9 FROM
10    geo_events_json AS geo_event
11    JOIN speed_events_json AS speed_event
12    ON (geo_event.driverId = speed_event.driverId)
13    LEFT JOIN CDP_Hive_Catalog.employees_hr_hive_db.driver
14        FOR SYSTEM_TIME AS OF PROCTIME() driver
15        ON driver.driverid = geo_event.driverId
16    LEFT JOIN `CDP_Kudu_Catalog`.`default_database`.`impala::employees_hr_kudu_impala_db.timesheet`
17        FOR SYSTEM_TIME AS OF PROCTIME() timesheet
18        ON (timesheet.driverid = geo_event.driverId AND timesheet.week = 1)
19    WHERE
20        geo_event.eventTimestamp BETWEEN
21            speed_event.eventTimestamp - INTERVAL '1' SECOND AND
22            speed_event.eventTimestamp + INTERVAL '1' SECOND
23        AND geo_event.eventType != 'Normal'
24        AND driver.wage_plan = 'hours'
25        AND timesheet.hours_logged > .45
```

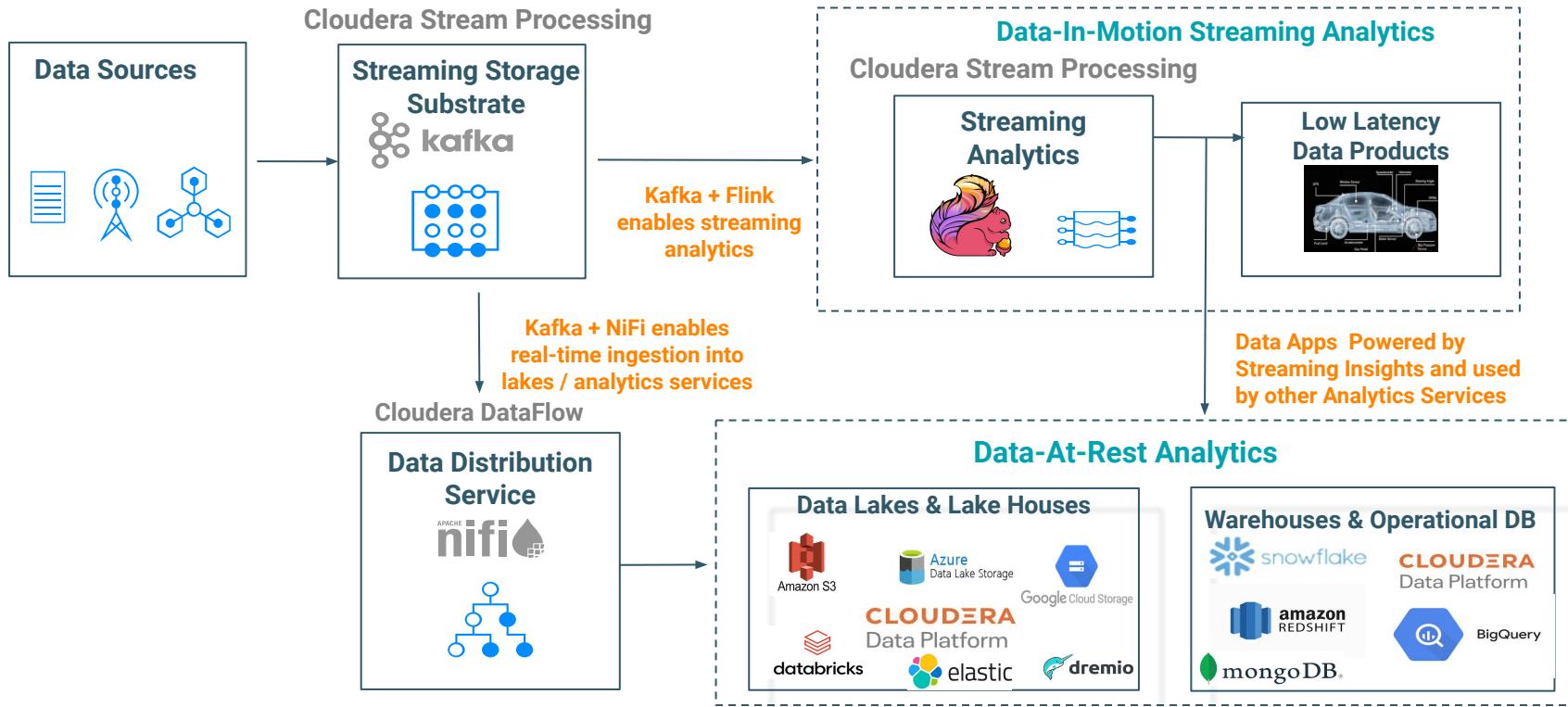
DATA GOVERNANCE FOR THE ENTIRE STREAMING PIPELINE

Streaming Data Lineage with SDX

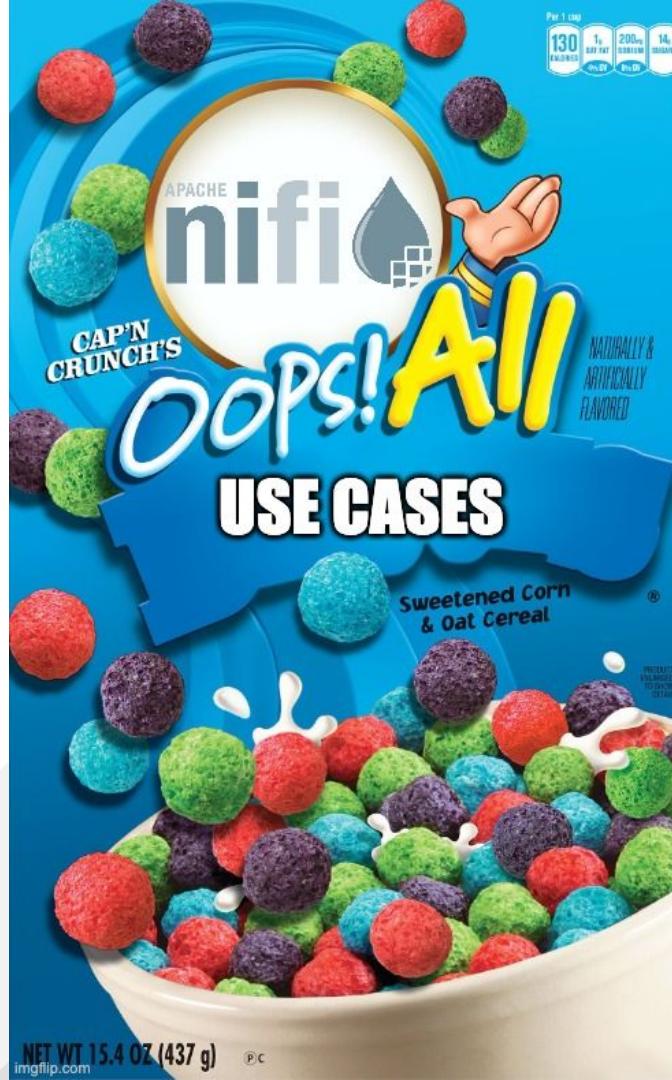
- Track Consumer, Producer, Topics and Consumer Group Lineage
- No changes required to Consumers or Producers
- End-To-End lineage from consumer to producer



Moving Beyond Draining of Streams Into Lakes: Analytics-in-Stream

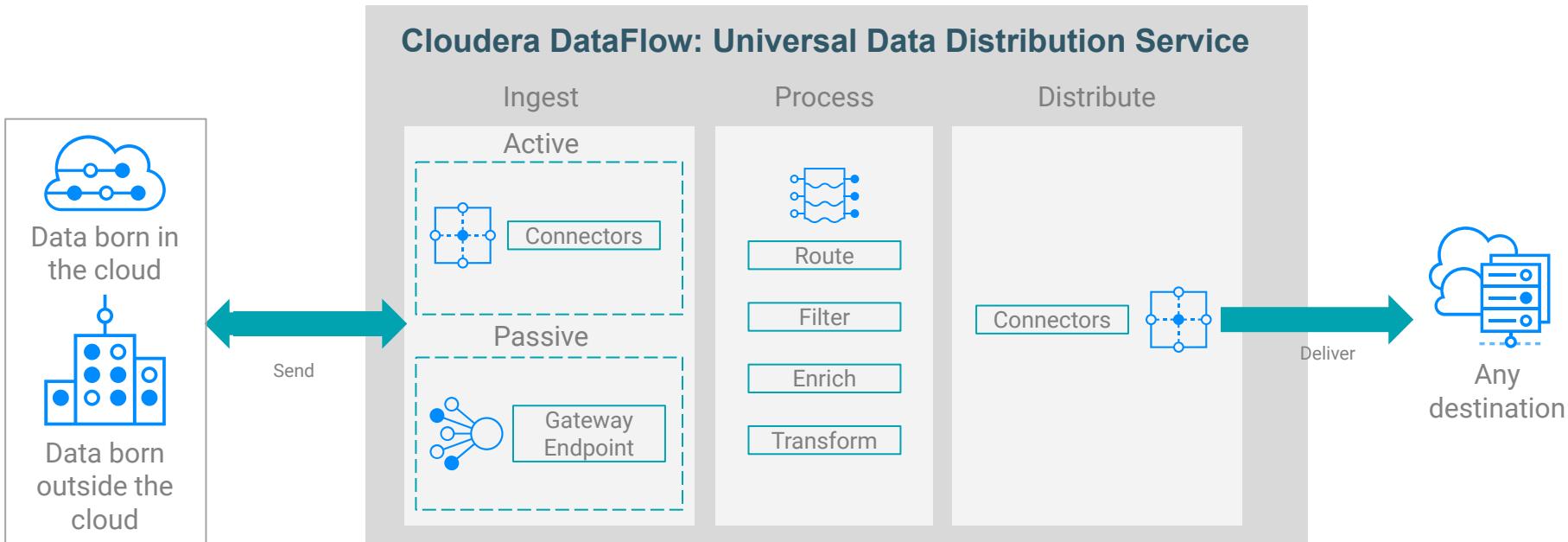


DATAFLOW APACHE NIFI



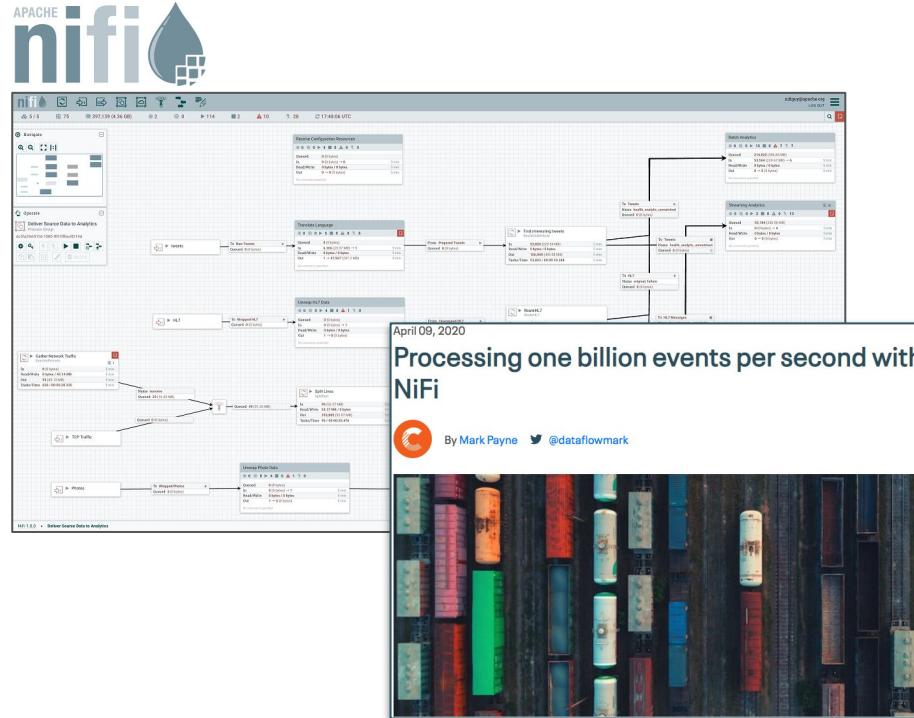
UNIVERSAL DATA DISTRIBUTION WITH CLOUDERA DATAFLOW (CDF)

Connect to Any Data Source Anywhere then Process and Deliver to Any Destination



CLOUDERA DATAFLOW - POWERED BY APACHE NiFi

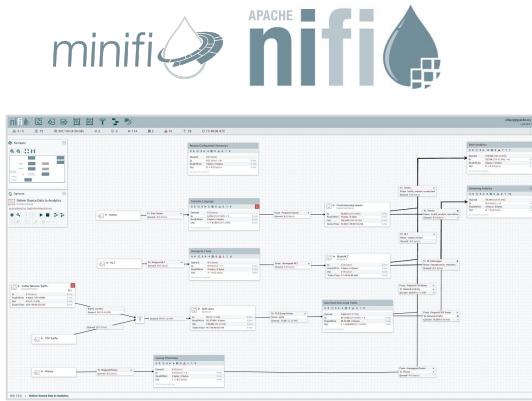
Ingest and manage data from edge-to-cloud using a no-code interface



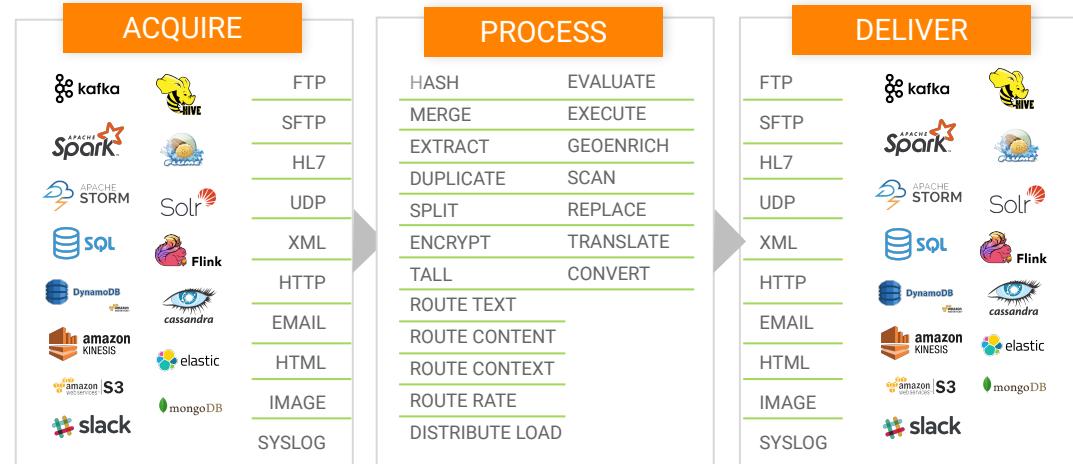
- #1 data ingestion/movement engine
- Strong community
- Product maturity over 11 years
- Deploy on-premises or in the cloud
- Over 400+ pre-built processors
- Built-in data provenance
- Guaranteed delivery
- Throttling and Back pressure

CLOUDERA FLOW AND EDGE MANAGEMENT

Enable easy ingestion, routing, management and delivery of any data anywhere (*Edge, cloud, data center*) to any downstream system with built in end-to-end security and provenance



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

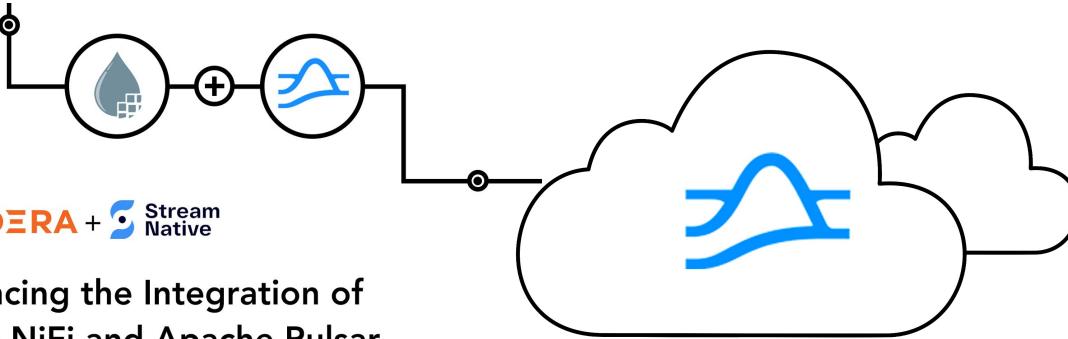


- Over 300 Prebuilt Processors
- Easy to build your own
- Parse, Enrich & Apply Schema
- Filter, Split, Merger & Route
- Throttle & Backpressure

- Guaranteed Delivery
- Full data provenance from acquisition to delivery
- Diverse, Non-Traditional Sources
- Eco-system integration



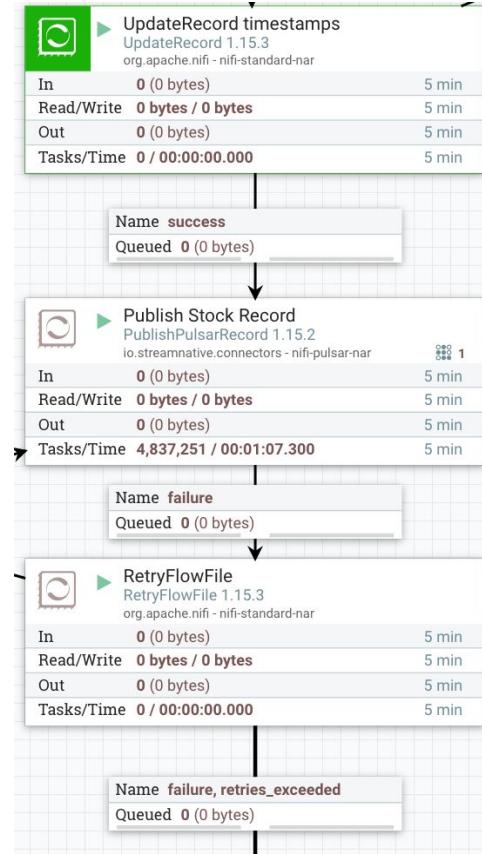
Apache NiFi Pulsar Connector

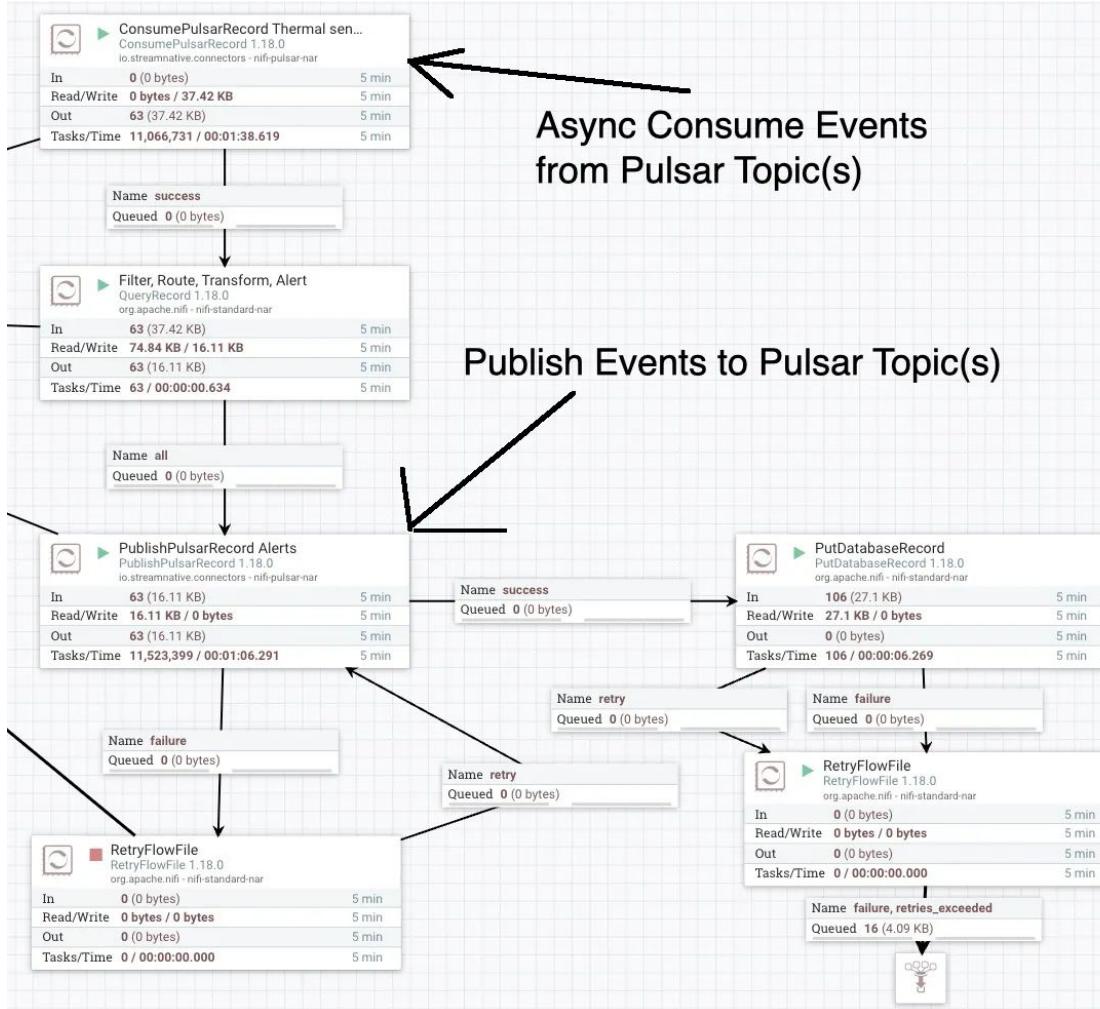


CLOUDERA + Stream Native

Announcing the Integration of
Apache NiFi and Apache Pulsar

<https://streamnative.io/apache-nifi-connector/>





PROVENANCE

Displaying 13 of 104
Oldest event available: 11/15/2016 13:34:50 EST

Showing the most recent events.

ConsumeKafka by component name

Date/Time	Type	FlowFile Uuid	Size	Component Name	Component Type
11/15/2016 13:35:03.8...	RECEIVE	379fc4f6-60e0-4151-9743-28...	44 bytes	ConsumeKafka	ConsumeKafka
11/15/2016 13:35:02.7...	RECEIVE	78f8c38b-89fc-4d00-a8d8-51...	44 bytes	ConsumeKafka	ConsumeKafka
11/15/2016 13:35:01.6...	RECEIVE	2bcd5124-bb78-489f-ad8a-7...	44 bytes	ConsumeKafka	ConsumeKafka

• Tracks data at each point as it flows through the system

• Records, indexes, and makes events available for display

• Handles fan-in/fan-out, i.e. merging and splitting data

• View attributes and content at given points in time

The diagram illustrates a data flow process. It starts with a red circle labeled "RECEIVE", which has an arrow pointing down to a grey circle labeled "JOIN". From the "JOIN" circle, an arrow points down to a grey circle labeled "DROP". Two green arrows originate from the "RECEIVE" and "JOIN" circles and point to a separate window titled "Provenance Event".

Provenance Event

DETAILS ATTRIBUTES CONTENT

Attribute Values

filename	328717796819631
kafka.offset	44815
kafka.partition	6
kafka.topic	nifi-testing
path	/
uuid	328717796819631-0000-0000-0000-000000000000

EXTENSIBILITY

- Built from the ground up with extensions in mind
- Service-loader pattern for...
 - Processors
 - Controller Services
 - Reporting Tasks
 - Prioritizers
- Extensions packaged as NiFi Archives (NARs)
 - Deploy NiFi lib directory and restart
 - Same model as standard components

The screenshot shows the IntelliJ IDEA interface with the project 'nifi-mxnetinference-processor' open. The left sidebar displays the project structure, including the 'nifi-mxnetinference-nar' directory and its contents. The right side shows the code editor with a Java test class 'InferenceProcessorTest.java'. Below the code editor is a 'Run' tool window showing a test run named 'InferenceProcessorTest.testProcessor'. The run details show three sequential steps: 1. A 'LinkProcessor' step with metrics: In 0 bytes, Read/Write 0 bytes / 31.45 KB, Out 2 (31.45 KB), Tasks/Time 2 / 0:00:04.808. 2. An 'UpdateAttribute' step with metrics: In 2 (31.45 KB), Read/Write 0 bytes / 0 bytes, Out 2 (31.45 KB), Tasks/Time 2 / 0:00:00.005. 3. A 'PutHDFS' step with metrics: In 2 (31.45 KB), Read/Write 31.45 KB / 0 bytes, Out 0 (0 bytes), Tasks/Time 2 / 0:00:00.603.

```
/*
 */
public class InferenceProcessorTest {

    private TestRunner testRunner;

    @Before
    public void init() {
        testRunner = TestRunners.newTestRunner(InferenceProcessor.class);
    }

    private String pathForResource
        URL r = this.getClass()
        URL uri = r.toURI();
        return Paths.get(uri)
    }

    private void runAndAssert(
        testRunner.setValidation();
        testRunner.assertValue();
        testRunner.assertAll();
        List<MockFlowFile> success;
        for (MockFlowFile mockFlowFile : success) {
            assertEquals(expect, mockFlowFile.getAttribute("key_1"));
        }
    }

    @Test
    public void testProcessor() {
        runAndAssert();
    }
}
```

NiFi Load Balancing

- Improve NiFi cluster throughput
- Defined at connection level
- Configurable balancing strategies
- Critical for scale up paradigm in Kubernetes
- Alleviates S2S balancing “hack” customers use

The screenshot shows the NiFi interface with a flow editor. At the top, there's a 'DETAILS' tab and a 'SETTINGS' tab. Under 'DETAILS', you can see the flowfile's Name, Id (88cbd631-0166-1000-0000-00002af80f96), FlowFile Expiration (0 sec), and Back Pressure settings (Object Threshold: 10000, Size Threshold: 1 GB). The 'Available Prioritizers' list includes FirstInFirstOutPrioritizer, NewestFlowFileFirstPrioritizer, OldestFlowFileFirstPrioritizer, and PriorityAttributePrioritizer. The 'Selected Prioritizers' list is currently empty. Below these, the 'Load Balance Strategy' dropdown is set to 'Do not load balance'. The flow editor shows two parallel connections. The first connection starts with a 'GenerateFlowFile' processor (version 1.8.0.3.3.0.0-165) with the following configuration:

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 42 KB	5 min
Out	42 (42 KB)	5 min
Tasks/Time	42 / 00:00:00.117	5 min

The second connection starts with a 'LogAttribute' processor (version 1.8.0.3.3.0.0-165) with the following configuration:

In	41 (41 KB)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	41 / 00:00:00.141	5 min

Both processors have a 'success' relationship pointing to a 'LogAttribute' processor. The final 'LogAttribute' processor has three outgoing connections, each with a target host 'hdf-pm-fe1' and port 'field.hortonworks.com:9...'. The last two connections are highlighted with red boxes.

QUEUE CONFIGURATION

- **FlowFile Expiration** - Data that cannot be processed in a timely fashion can be automatically removed from the flow.
- **Back Pressure Thresholds** - Thresholds indicate how much data should be allowed to exist in the queue before the component that is the source of the Connection is no longer scheduled to run. This allows the system to avoid being overrun with data.
- **Load Balance Strategy** – Strategy to distribute the data in a flow across the nodes in the cluster. When enabled, compression can be configured on FlowFile contents and attributes.
- **Prioritization** – Determines the order in which flow files are processed.

Generate Syslog RFC5424	
ExecuteScript 1.13.2.2.2.0-127 org.apache.nifi - nifi-scripting-nar	
In	0 (0 bytes)
Read/Write	0 bytes / 0 bytes
Out	0 (0 bytes)
Tasks/Time	0 / 00:00:00.000
	5 min

Configure Connection

DETAILS SETTINGS

Name: success_Generate-FilterEvents

Id: 64146cca-d197-3c27-9c47-015dd7b7a6c6

FlowFile Expiration: 0 sec

Back Pressure Object Threshold: 10000

Size Threshold: 1 GB

Available Prioritizers:

- FirstInFirstOutPrioritizer
- NewestFlowFileFirstPrioritizer
- OldestFlowFileFirstPrioritizer
- PriorityAttributePrioritizer

Selected Prioritizers:

Load Balance Strategy: Round robin

Load Balance Compression: Do not compress

RECORD-ORIENTED DATA WITH NIFI

- **Record Readers** - Avro, CSV, Grok, IPFIX, JSON1, JSON, Parquet, Scripted, Syslog5424, Syslog, WindowsEvent, XML
- **Record Writers** - Avro, CSV, FreeFromText, Json, Parquet, Scripted, XML
- Record Reader and Writer support referencing a schema registry for retrieving schemas when necessary.
- Enable processors that accept any data format without having to worry about the parsing and serialization logic.
- Allows us to keep FlowFiles larger, each consisting of multiple records, which results in far better performance.

Filter Events	
QueryRecord 1.13.2.2.2.2.0-127 org.apache.nifi - nifi-standard-nar	
In	0 (0 bytes) 5 min
Read/Write	0 bytes / 0 bytes 5 min
Out	0 (0 bytes) 5 min
Tasks/Time	0 / 00:00:00.000 5 min

Configure Processor

SETTINGS SCHEDULING PROPERTIES COMMENTS

Required field

Property	Value
Record Reader	CSVReader
Record Writer	JsonRecordSetWriter

+

RUNNING SQL ON FLOWFILES

- Evaluates one or more SQL queries against the contents of a FlowFile.
- This can be used, for example, for field-specific filtering, transformation, and row-level filtering.
- Columns can be renamed, simple calculations and aggregations performed.
- The SQL statement must be valid ANSI SQL and is powered by Apache Calcite.

Filter Events		
QueryRecord 1.13.2.2.2.2.0-127 org.apache.nifi - nifi-standard-nar		
In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

Configure Processor | QueryRecord 1.13.2.2.2.2.0-127

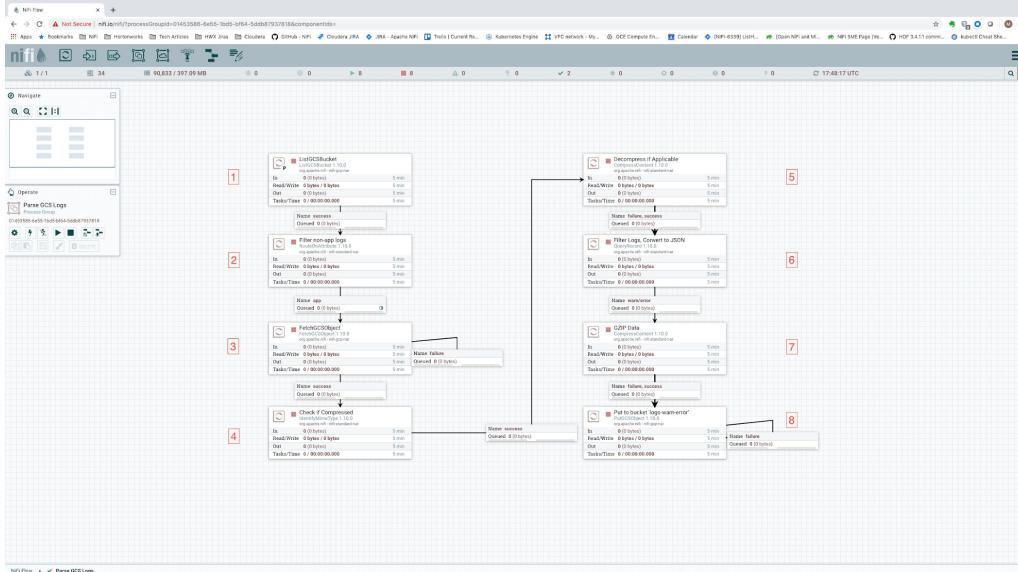
Stopped

SETTINGS SCHEDULING PROPERTIES COMMENTS

Required field

Property	Value
Record Reader	Syslog_5424_Reader
Record Writer	JSON_Syslog_5424_Writer
Include Zero Record FlowFiles	false
Cache Schema	false
Default Decimal Precision	10
Default Decimal Scale	0
filtered_events	#(Filter Rule)

Processing one million events per second with NiFi



Nodes	Data rate/sec	Events/sec	Data rate/day	Events/day
1	192.5 MB	946,000	16.6 TB	81.7 Billion
5	881 MB	4.97 Million	76 TB	429.4 Billion
25	5.8 GB	26 Million	501 TB	2.25 Trillion
100	22 GB	90 Million	1.9 PB	7.8 Trillion
150	32.6 GB	141.3 Million	2.75 PB	12.2 Trillion

Apache NiFi with Python Custom Processors

Python as a 1st class citizen

```
import cv2
import numpy as np
import json
from nifiapi.properties import PropertyDescriptor
from nifiapi.properties import ResourceDefinition
from nifiapi.flowfiletransform import FlowfiletransformResult

SCALE_FACTOR = 0.00392
NMS_THRESHOLD = 0.4 # non-maximum suppression threshold
CONFIDENCE_THRESHOLD = 0.5

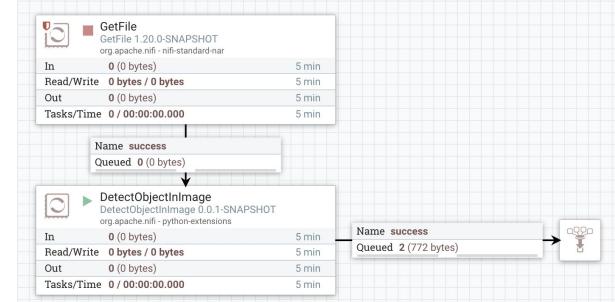
class DetectObjectInImage:
    class Java:
        implements = ['org.apache.nifi.python.processor.FlowfileTransform']
        class ProcessorDetails:
            version = '0.0.1-SNAPSHOT'
            dependencies = ['numpy >= 1.23.5', 'opencv-python >= 4.6']

    def __init__(self, jvm=None, **kwargs):
        self.jvm = jvm

        # Build Property Descriptors
        self.model_file = PropertyDescriptor(
            name = 'Model File',
            description = 'The binary file containing the trained Deep Neural Network weights. Supports Caffe (*.caffemodel), TensorFlow (*.pb), Torch (*.t7, *.net), Darknet (*.weights), ' +
                        'OLDY (*.bin), and ONNX (*.onnx)',
            required = True,
            resource_definition = ResourceDefinition(allow_file = True)
        )
        self.config_file = PropertyDescriptor(
            name = 'Network Config File',
            description = 'The text file containing the Network configuration. Supports Caffe (*.prototxt), TensorFlow (*.pbtxt), Darknet (*.cfg), and DLDT (*.xml)',
            required = False,
            resource_definition = ResourceDefinition(allow_file = True)
        )
        self.class_name_file = PropertyDescriptor(
            name = 'Class Names File',
            description = 'A text file containing the names of the classes that may be detected by the model. Expected format is one class name per line, new-line terminated.',
            required = True,
            resource_definition = ResourceDefinition(allow_file = True)
        )
        self.descriptors = [self.model_file, self.config_file, self.class_name_file]

    def getPropertyDescriptors(self):
        return self.descriptors

    def onScheduled(self, context):
        # read class names from text file
        class_name_file = context.getProperty(self.class_name_file.name).getValue()
        if class_name_file is None:
```



CUSTOM JAVA APPS

Custom Processors

<https://github.com/tspannhw/nifi-extracttext-processor>

<https://github.com/tspannhw/nifi-tensorflow-processor>

<https://github.com/tspannhw/nifi-nlp-processor>

<https://github.com/tspannhw/nifi-convertjsontodd1-processor>

<https://github.com/tspannhw/nifi-corenlp-processor>

<https://github.com/tspannhw/nifi-imageextractor-processor>

<https://github.com/tspannhw/nifi-attributecleaner-processor>

<https://github.com/tspannhw/linkextractorprocessor>

<https://github.com/tspannhw/GetWebCamera>

<https://github.com/tspannhw/nifi-langdetect-processor>

<https://github.com/tspannhw/nifi-postimage-processor>

Custom Processors in Java

Processor Details

Running STOP & CONFIGURE

SETTINGS SCHEDULING PROPERTIES RELATIONSHIPS COMMENTS

Required field

Property	Value
url	https://api.imgur.com/3/upload
fieldname	image
imagename	\${filename}
imagedtype	image/gif
headername	Authorization
headervalue	Client-ID x
basicusername	No value set
basicpassword	No value set

OK

Name

PostImageProcessor

Id

f62ebd85-3615-351b-6e72-812c68e68cf8

Type

PostImageProcessor 1.1

Bundle

com.dataflowdeveloper - nifi-postimage-nar

Penalty Duration ?

30 sec

Yield Duration ?

1 sec

Bulletin Level ?

WARN

	▶ PostImageProcessor	PostImageProcessor 1.1	com.dataflowdeveloper - nifi-postimage-nar
In	18 (15.85 MB)	5 min	
Read/Write	15.85 MB / 0 bytes	5 min	
Out	18 (15.85 MB)	5 min	
Tasks/Time	18 / 00:00:01.648	5 min	
	Name success		
	Queued 0 (0 bytes)		

<https://www.datainmotion.dev/2019/03/getting-started-with-custom-processor.html>

<https://www.datainmotion.dev/2019/03/posting-images-to-imgur-via-apache-nifi.html>

```
public class TensorFlowProcessor extends AbstractProcessor {
    public static final PropertyDescriptor MODEL_DIR = new
PropertyDescriptor.Builder().name(MODEL_DIR_NAME)
        .description("Model Directory").required(true).expressionLanguageSupported(true)
        .addValidator(StandardValidators.NON_EMPTY_VALIDATOR).build();

@Override
    public void onTrigger(final ProcessContext context, final ProcessSession session) throws
ProcessException {
    FlowFile flowFile = session.get();
    if (flowFile == null) {
        flowFile = session.create();
    }
    try {
        flowFile.getAttributes();
    }
}
```

<https://github.com/tspannhw/nifi-tensorflow-processor>

CLOUDERA FLOW DESIGNER



ReadyFlow Gallery

Iceberg X

Added

Kafka to Iceberg

Version 1

Consumes JSON, CSV or Avro events from Kafka and writes them as Parquet files to a destination Iceberg table.

[View Added Flow Definition](#) [Create New Draft](#)

READYFLOW GALLERY

- Cloudera provided flow definitions
- Cover most common data flow use cases
- Optimized to work with CDP sources/destinations
- Can be deployed and adjusted as needed

ReadyFlow Gallery

Search by name

Added

Kafka filter to Kafka Version 1

Consumes JSON, CSV or Avro events from Kafka, filters them before writing them back to Kafka as JSON, CSV or Avro.

[View Added Flow Definition](#)

Added

Kafka to Cloudera Operational Database Version 1

Consumes JSON, CSV or Avro events from Kafka and ingests them into Cloudera Operational Database (COD).

[View Added Flow Definition](#)

Kafka to Kafka Version 1

Consumes events from Kafka and writes them to another Kafka topic.

[Add To Catalog](#)

Kafka to Kudu Version 1

Consumes JSON, CSV or Avro events from Kafka and ingests them into Kudu.

[Add To Catalog](#)

Kafka to S3 Avro Version 1

Consumes JSON, CSV or Avro events from Kafka and writes Avro files to S3.

[View Added Flow Definition](#)

S3 to S3 Avro Version 1

Consumes JSON, CSV or Avro files from source S3 location and writes Avro files to a destination S3 location.

[Add To Catalog](#)

FLOW CATALOG

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

The screenshot shows the Cloudera DataFlow Flow Catalog interface. On the left is a dark sidebar with navigation links: Dashboard, Catalog (which is selected), ReadyFlow Gallery, Environments, Help, and a user icon. At the bottom of the sidebar, it says "1.0.1-b570". The main area is titled "Flow Catalog" and contains a search bar and a refresh button indicating it was last refreshed 25 seconds ago. A blue "Import Flow Definition" button is located in the top right. Below these are two tabs: "Name ↑" and "Last Updated". The main content is a table listing ten flow definitions:

Name ↑	Type	Versions	Last Updated	Action
cc_fraud_template_int101run	Custom Flow Definition	2	a day ago	>
cc_fraud_template_int101run2	Custom Flow Definition	1	9 days ago	>
JSON_Kafka_To_Avro_S3	Custom Flow Definition	2	a day ago	>
Kafka filter to Kafka	ReadyFlow	1	2 days ago	>
Kafka to Cloudera Operational Database	ReadyFlow	1	2 days ago	>
Kafka to S3 Avro	ReadyFlow	1	14 hours ago	>
nifi_flows	Custom Flow Definition	1	2 months ago	>
Weather Data Flow	Custom Flow Definition	1	a day ago	>
Weather_Data	Custom Flow Definition	1	15 days ago	>
Weather_JSON_Kafka_To_Avro_S3	Custom Flow Definition	1	21 days ago	>

At the bottom right, there are buttons for "Items per page:" (set to 10), a page number "1 - 10 of 10", and navigation arrows.

DEPLOYMENT WIZARD

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

Start Deployment Wizard

dataflow-demo-new / New Flow Deployment

Overview

Selected Flow Definition

NAME	Machine Data To Warehouse	VERSION	3
------	---------------------------	---------	---

Target Environment

NAME	dataflow-demo-new
------	-------------------

NiFi Runtime Version

CURRENT VERSION	Latest Version (1.14.0.2.3.1.0-3)	Change Version
-----------------	-----------------------------------	----------------

Deployment Name

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

Select File

Drop file or browse

CDP Truststore

Select File

Drop file or browse

CDPSchemaRegistry

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

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

<input checked="" type="radio"/> Extra Small	<input type="radio"/> Small	<input type="radio"/> Medium	<input type="radio"/> Large
2 vCores Per Node 4 GB Per Node	4 vCores Per Node 8 GB Per Node	8 vCores Per Node 16 GB Per Node	16 vCores Per Node 32 GB Per Node

Number of NiFi Nodes

Auto Scaling

Enabled

Min. Nodes: 1

Max. Nodes: 3

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

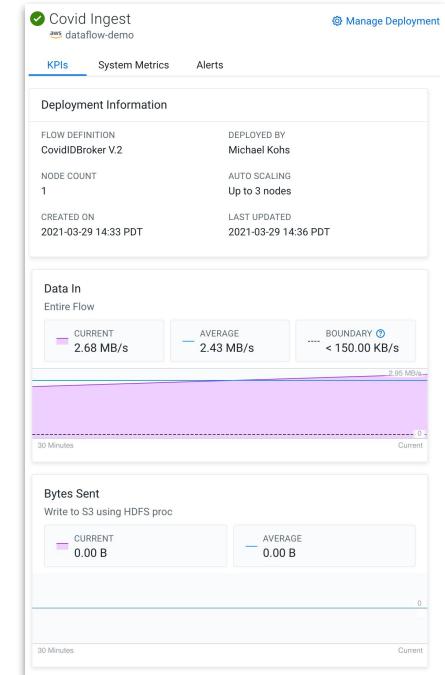
KPI Definition in Deployment Wizard

The screenshot shows the 'New Deployment' wizard step 5: 'Key Performance Indicators'. The left sidebar lists steps 1 through 6. The main area displays two sections for defining KPIs:

- Entire Flow**: Metric to track: Flow Files Queued. Alert set: No alert set.
- Entire Flow**: Metric to track: Data Out. Alert set: Notify if outside the range of 999 MB/sec - 1 MB/sec, for at least 5 minutes.

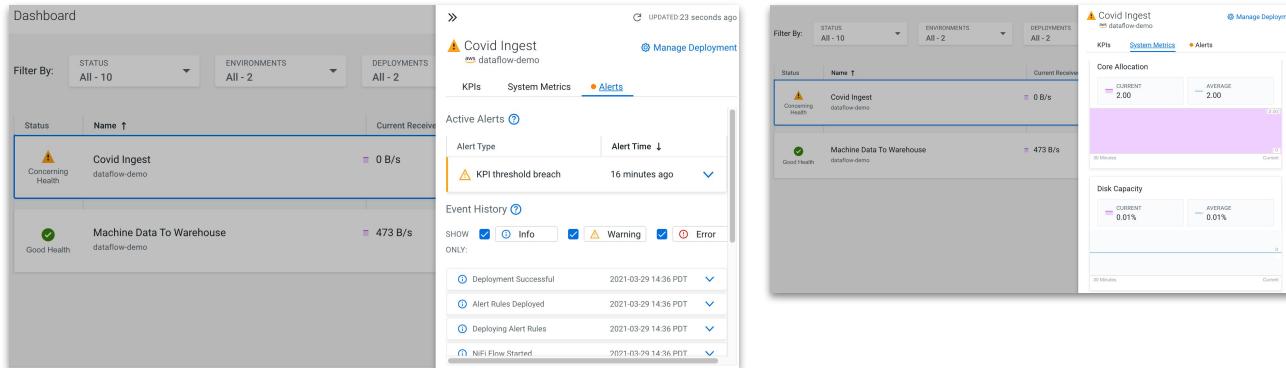
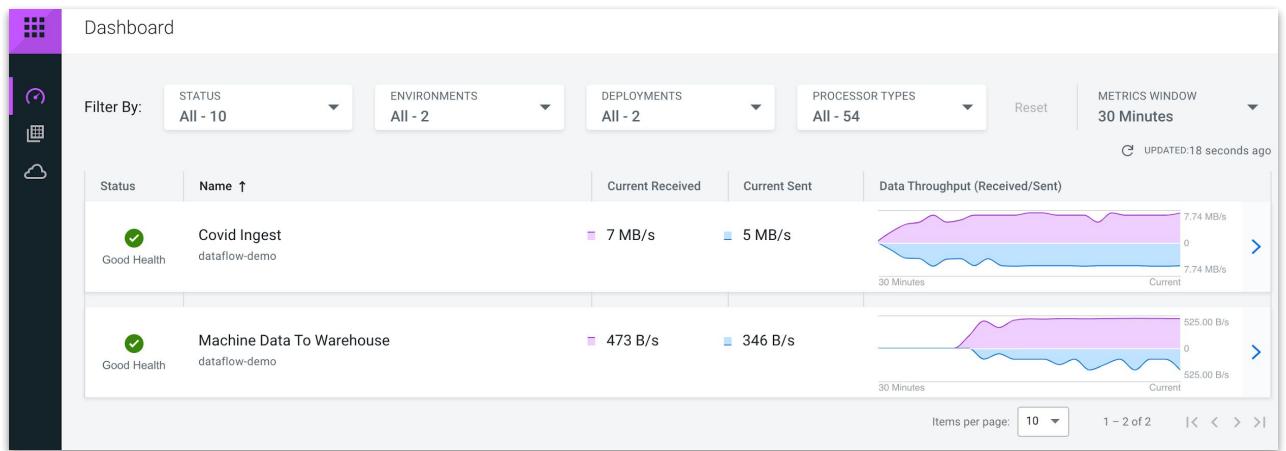
A button at the bottom right says '@ Add New KPI'.

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



DEPLOYMENT MANAGER

- Manage flow deployment lifecycle
(Suspend/Start/Terminate)
- Add/Edit KPIs
- Change sizing configuration
- Update parameters
- Change NiFi version of the deployment
- Gateway to NiFi canvas

Dashboard / dataflow-demo-new / Kafka to COD

REFRESHED 12 seconds ago

Actions ▾

Deployment Manager

Status: Good Health

Deployment Name: Kafka to COD

Flow Definition: Kafka to Cloudera Operational Database V.1

Node Count: 1

Auto Scaling: Disabled

Created On: 2021-07-26 17:05 PDT

Region: US West (Oregon)

NIFI Runtime Version: 1.14.0.2.3.0.0-89

Deployed By: Michael Kohs

Last Updated: 2021-07-26 17:07 PDT

CN# cmcdp:df:us-west-1:9d74eee4-1cad-45d7-b645-7ccf9edb73d deploy...

Deployment Settings

KPIs and Alerts Sizing and Scaling Parameters

Parameters

Running Processors that are affected by the Parameter changes will automatically be restarted.

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.

The selected flow definition references an external Default NiFi SSL Context Service. Hence, DataFlow will automatically create a matching SSL Context Service with a keystore and truststore generated from the target environment's FreeIPA certificate.

kafka-to-cod

CDP Workload User: srv_nifi-kafka-ingest

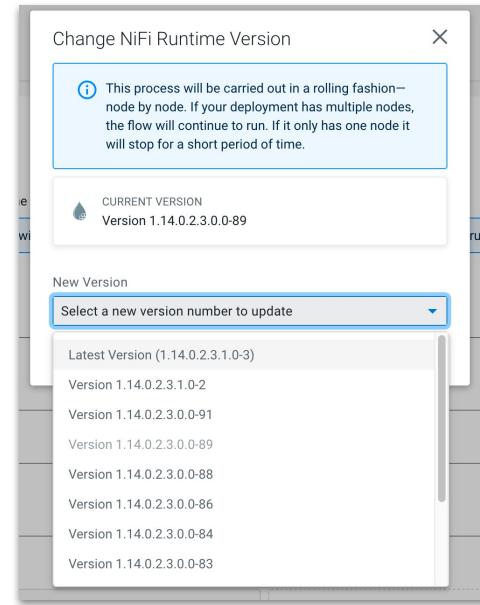
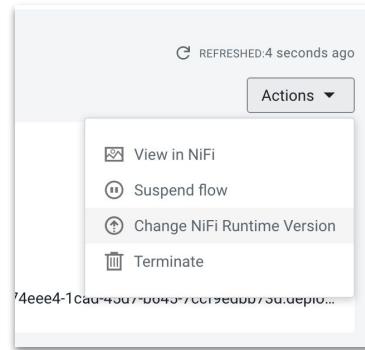
CDP Workload User Password: Sensitive value provided.

CDP Environment: hbase-site.xml core-site.xml

Discard Changes Apply Changes

NIFI VERSION UPGRADES

- Pick up NiFi hotfixes easily
- Upgrade (or downgrade) the hotfix version of existing deployments
- Rolling upgrade (if the deployment has >1 NiFi nodes)



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 and highlighted in purple), Functions, Environments, Get Started, Help, and Stephen Hawking. At the bottom of the sidebar, it says v2.0.0.

The main area is titled "Flow Design / [WorkspaceName] / [FlowDefinitionName]". It features a visual canvas with a single processor icon. A tooltip for the processor displays the following details:

[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

Below the canvas, there's a footer bar with "- + Apply Changes Discard Changes".

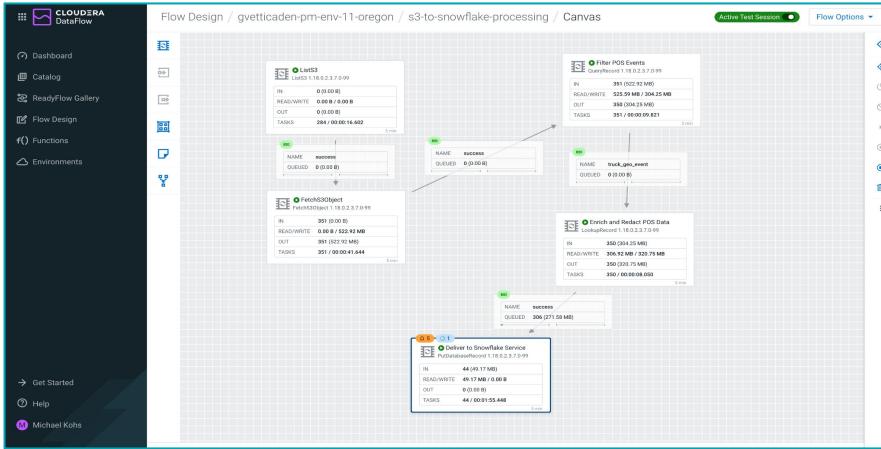
On the right side, there's a configuration panel with tabs for "Configuration" (which is selected) and "Metrics". The configuration section lists various parameters with their current values and edit icons:

- * 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.

No-Code/Low-Code User Experience

DataFlow Designer



visual canvas + automatic provisioning =
Self-service pipeline development

SQL Stream Builder

The screenshot shows the Data Pipeline UI interface. At the top, there's a navigation bar with 'Projects / sso_default' on the left and a search bar 'Search in SSB' on the right. Below the navigation bar is the 'Job Actions' dropdown menu.

The main area displays the 'elegant_babbage' job details. The job status is 'FINISHED'. It includes sections for 'Job Dashboard' and 'Job Settings'. The 'Job Settings' section shows the following configuration:

```
CREATE TABLE kafka_table_1670513700 (
    col_1$ STRING,
    col_2$ STRING,
    col_3$ TIMESTAMP(3),
    col_4$ TIMESTAMP(3)
) PARTITIONED BY col_1$ AS col_ts$ -> INTERVAL 10 SECOND
WITH CONNECTION FOR 'kafka' -- Specify what connector to use, for Kafka it must use 'kafka'.
FOR SOURCE 'kafka' -- Source for the data
USING CONNECTION 'kafka' -- Connection to Kafka brokers.
REPLICAS 1 -- Come separated list of Kafka brokers.
LOGICAL 'true' -- To read data from when the table is used as source. It also supports topic list for source by separating topic by commas.
NOCHECKPOINT -- No checkpoints will be created for sources. When the table is used as sink, the topic name is the table to write data to. Note: topic list is not supported for sinks.
ENCODING 'JSON' -- Optional flag to specify encoding for sinks.
PLAIN_NUMBERS 'false' -- Optional flag to specify whether to encode all decimals as plain numbers instead of scientific notation.
FAIL_ON_MISSING_FLD 'false' -- Optional flag to fail if a field is missing or null, false by default.
FAIL_ON_PARSE_ERROR 'true' -- Optional flag to skip fields and rows with parse errors instead of failing. Fields set to null in case of errors, false by default.
JSON_AS_NULLKEY_MODE 'null' -- Optional flag to convert to string literal for null keys when 'JSON_AS_NULLKEY_MODE' is LITERAL, 'null' by default.
JSON_AS_NULLKEY_MODE 'null' -- Optional flag to control the handling mode when serializing null key for exp data. FAIL by default.
```

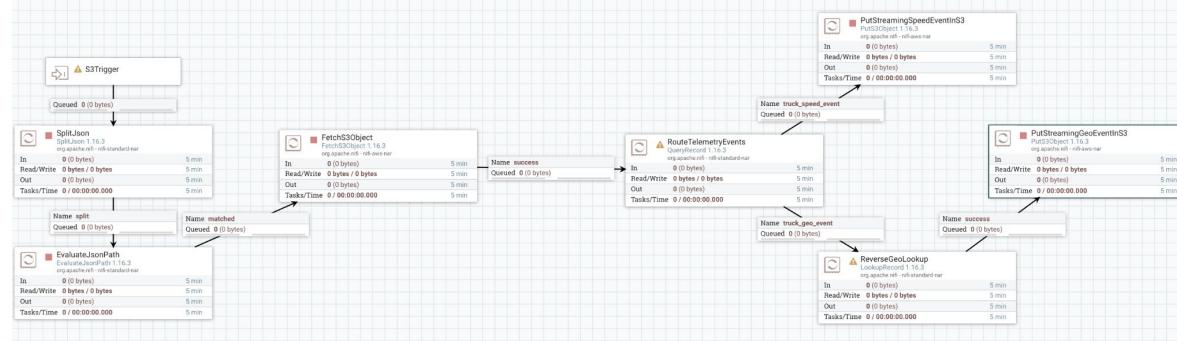
Below the job settings, there's a log window showing the execution history:

```
08/12/2022, 16:34:51 Active job started
08/12/2022, 16:35:08 Inserted kafka jsonconnector DDL template
08/12/2022, 16:35:08 elegant_babbage started
08/12/2022, 16:35:08 CREATE TABLE (catalogPath = '/etc/airflow/catalogs/elegant_babbage', database = 'kafka', interval = '10 second', schema = 'systematic((col1$))')
08/12/2022, 16:35:08 schemaMarking = 'auto', schemaVersion = 1, schemaName = 'ELEGANTBABBAGE', strategy = 'partitioned', type = 'table', schemaNameOrPath = 'elegant_babbage', schemaVersion = 1, schemaNameOrPath = 'elegant_babbage', identifier = 'sso_elegant_babbage'
08/12/2022, 16:35:10 Active job stopped
08/12/2022, 16:35:12 yourtable_babbage loaded into editor.
08/12/2022, 16:35:12 yourtable_leavest loaded into editor.
08/12/2022, 16:35:13 yourtable_babbage loaded into editor.
08/12/2022, 16:35:13 yourtable_leavest loaded into editor.
08/12/2022, 16:35:14 yourtable_babbage loaded into editor.
08/12/2022, 16:35:14 yourtable_leavest loaded into editor.
08/12/2022, 16:35:15 Inserted Faker connector DDL template
08/12/2022, 16:35:15 Inserted Faker connector DDL template
08/12/2022, 16:35:21 yourtable_babbage loaded into editor.
08/12/2022, 16:35:21 yourtable_leavest loaded into editor.
08/12/2022, 16:35:22 elegant_babbage loaded into editor.
```

Instant data access + unified SQL processing =
Self-service streaming event processing

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



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

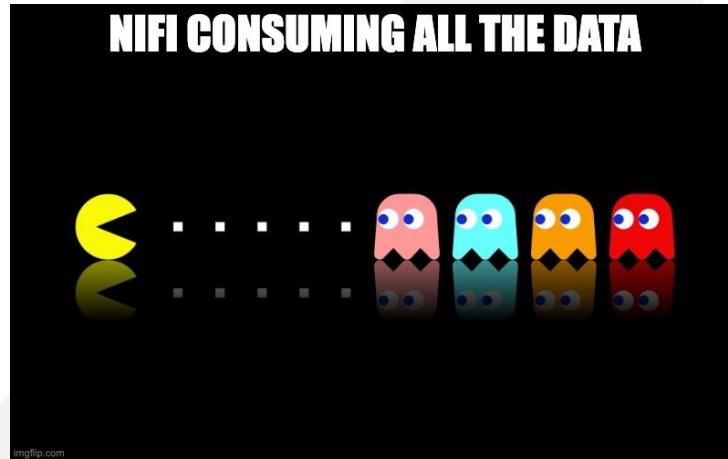
Example: Build high performant, scalable web applications across multiple data centers.

Serverless Customized Triggers

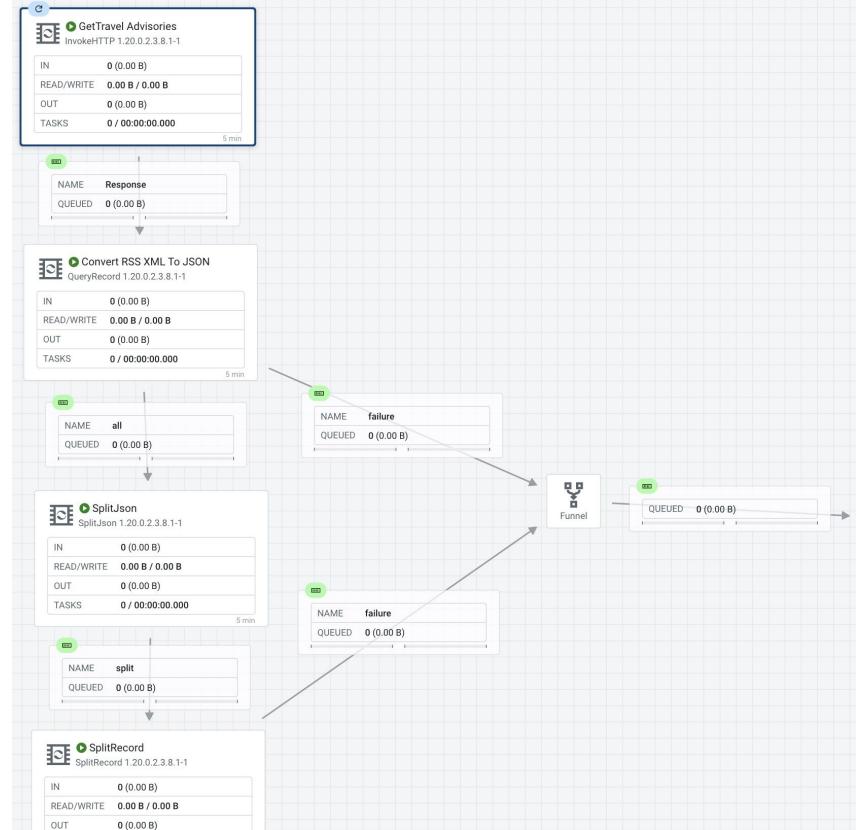
With the DFF State feature, build flows to create customized triggers allowing access to on-premises or external services

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

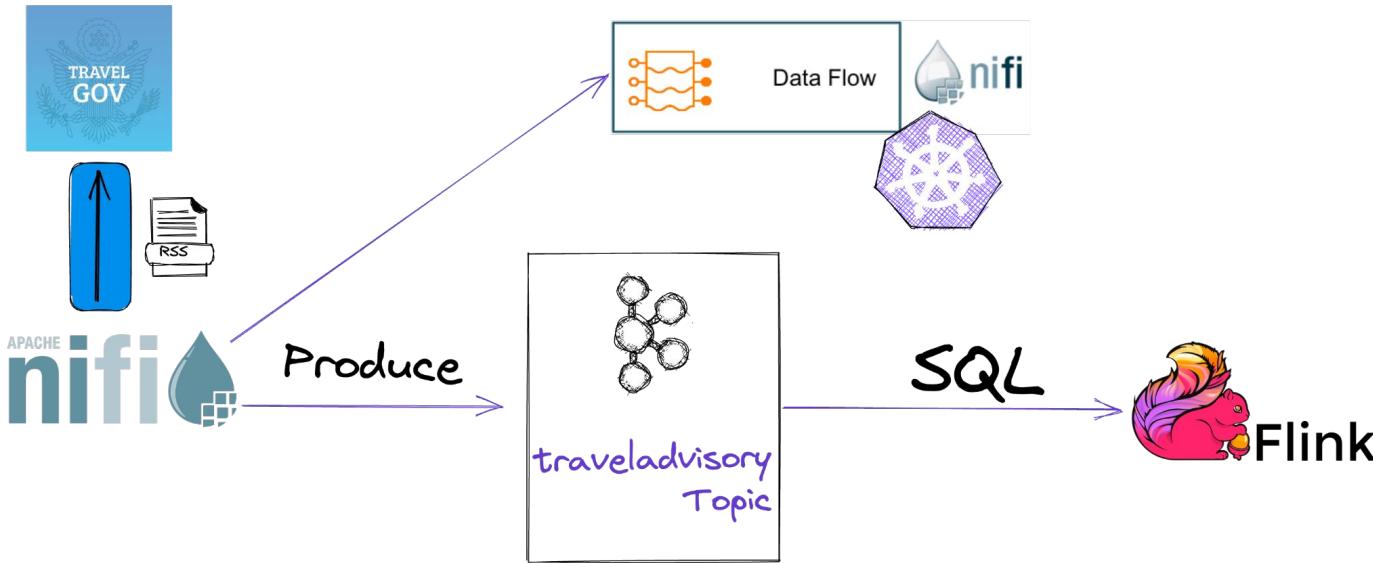
DEMOS



Cloudera Dataflow Demos



Examples



CLOUDERA

<https://github.com/tspannhw/FLaNK-TravelAdvisory>



traveladvisories X

searchplanes

RUNNING



Flink Dashboard

Templates

Editor

Materialized View

Job Settings

Job Actions

```
1 select title, domain, category, link, pubdate, ts, uuid, advisoryId
2 FROM
3 `sr1`.`default_database`.traveladvisory
4
```

 Restart Stop Stop Polling Polling samples...

<input type="checkbox"/> title	domain	category	link	pubdate	ts	uuid
<input type="checkbox"/> Bhutan - Level 1: Exercise Normal Precautions	BT,advisory	Level 1: Exercise Normal ...	http://travel.state.gov/co...	Wed, 05 Oct 2022	1680277517680	0412509-ea00-4000-95...
<input type="checkbox"/> China - Level 3: Reconsider Travel	CH,advisory,MC,HK	CH	http://travel.state.gov/co...	Fri, 10 Mar 2023	1680277517682	79e7912a-5d40-4afb-96...
<input type="checkbox"/> China - Level 3: Reconsider Travel	CH,advisory,MC,HK	HK	http://travel.state.gov/co...	Fri, 10 Mar 2023	1680277517682	528c584a-e2cc-4119-ac...
<input type="checkbox"/> Tajikistan - Level 2: Exercise Increased Caution	TI,advisory	Level 2: Exercise Increas...	http://travel.state.gov/co...	Wed, 05 Oct 2022	1680277517683	24fef95e-42a9-4011-9f3...
<input type="checkbox"/> Zambia - Level 1: Exercise Normal Precautions	ZA,advisory	advisory	http://travel.state.gov/co...	Tue, 28 Mar 2023	1680277517684	a4e8106e-5f55-4ef9-a5e...
<input type="checkbox"/> Taiwan - Level 1: Exercise Normal Precautions	TW,advisory	advisory	http://travel.state.gov/co...	Mon, 24 Oct 2022	1680277517688	ed3bad9e-96a0-42ca-a6...
<input type="checkbox"/> Chad - Level 3: Reconsider Travel	CD,advisory	Level 3: Reconsider Travel	http://travel.state.gov/co...	Tue, 04 Oct 2022	1680277517690	1ac6673c-dd29-4186-b8...

Logs

Results

Events

1 to 7 of 7

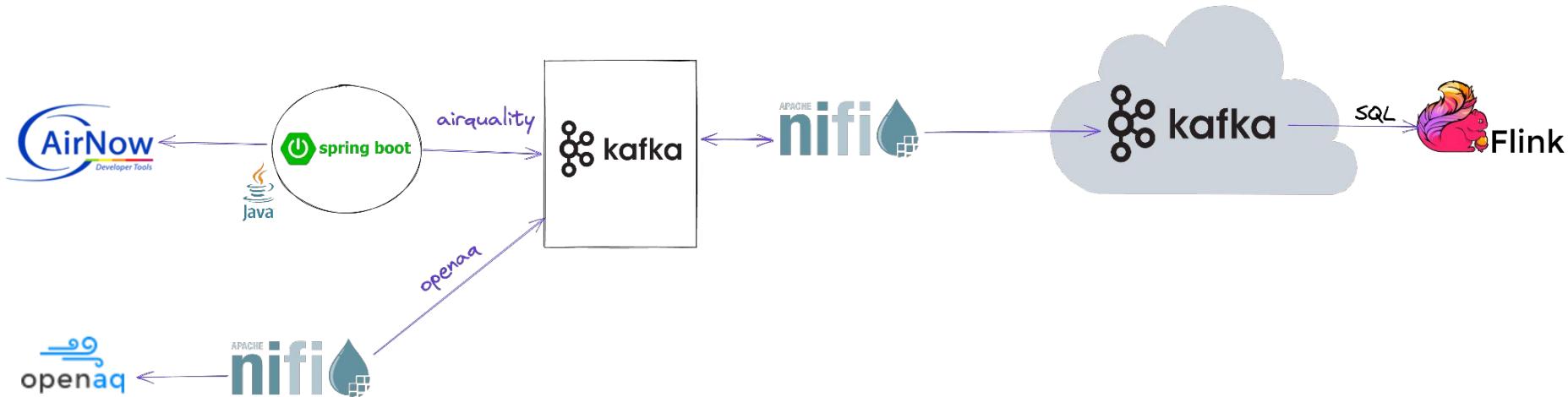
<

Page 1 of 1

>

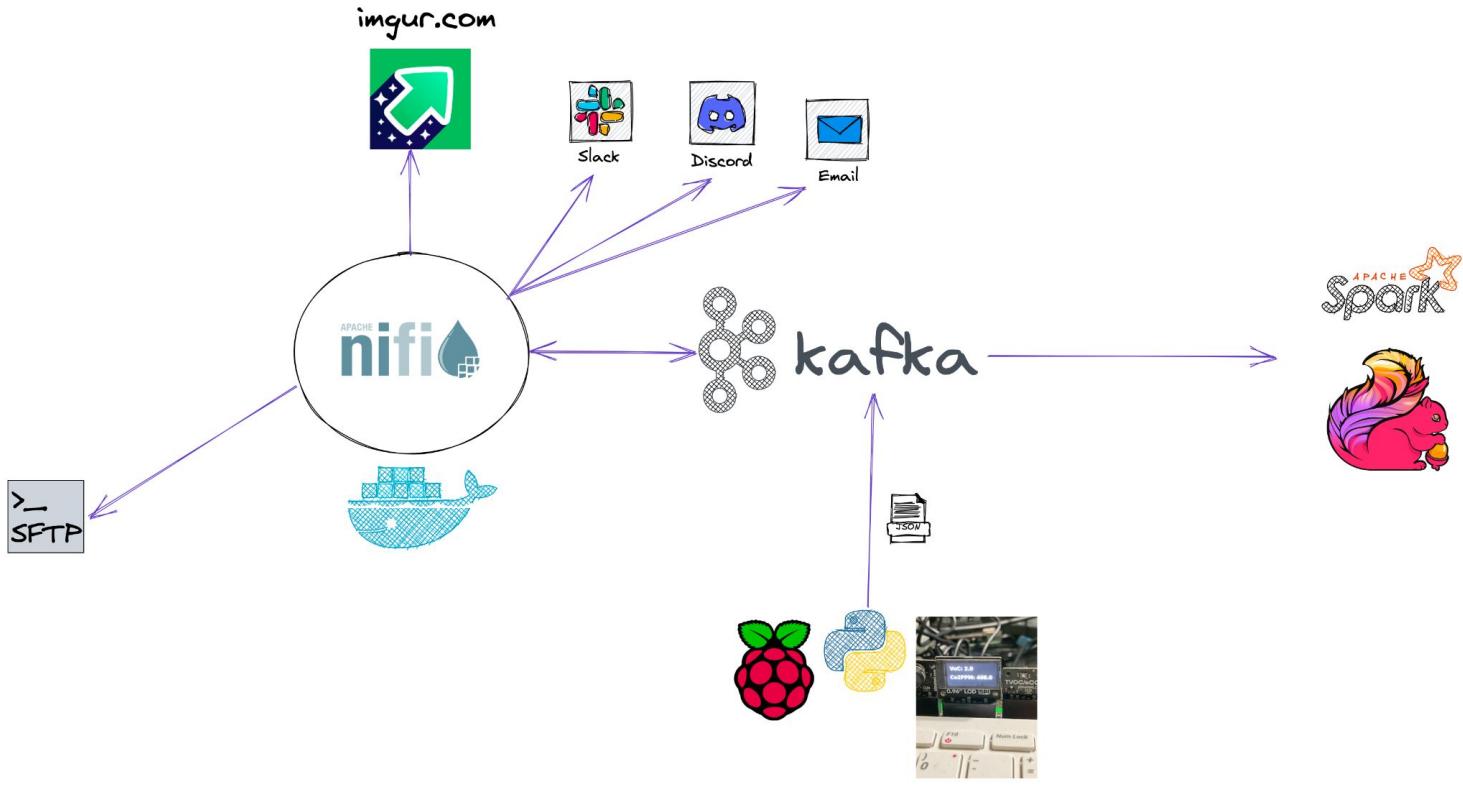
>>|

Examples - Spring Boot - Apache Kafka - Apache NiFi - Apache Flink

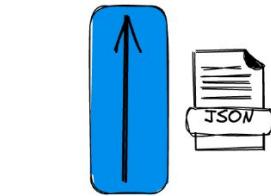


<https://github.com/tspannhw/flank-airquality>

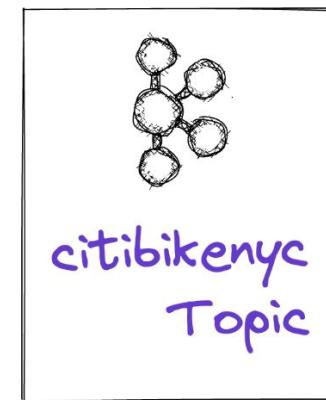
Examples - Python - Kafka - NiFi - Flink - Spark



Examples - NiFi - Kafka - Flink



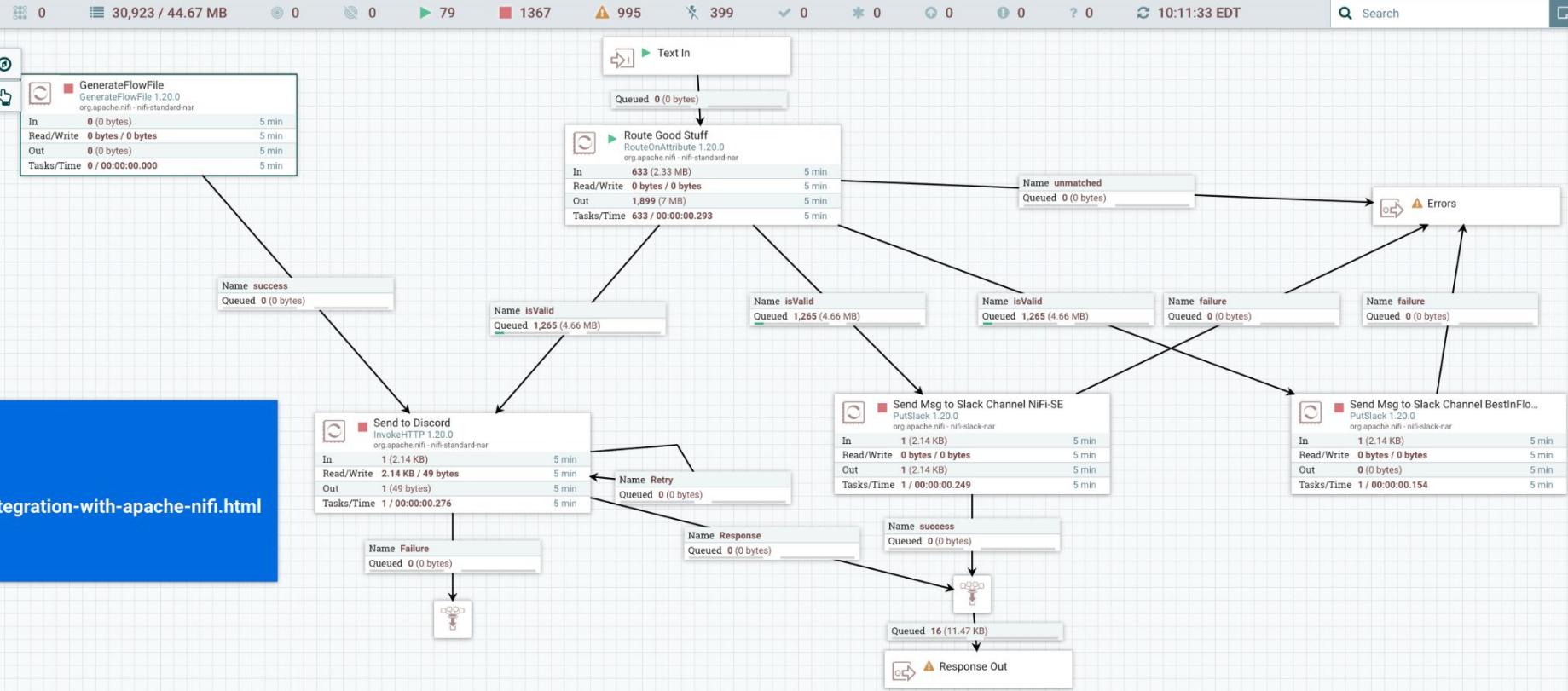
Produce



SQL



CLOUDERA



Weather Data For USA

Location: USA

City: Los Angeles

Country: United States

Timezone: UTC -07:00

Wind Speed: 10.0

Wind Direction: N

Cloud Cover: 50%

Humidity: 40%

Pressure: 1013 hPa

Temperature: 22.0 °C

UV Index: 5.0

Visibility: 10.0 km

Wind Gust: 15.0 m/s

Wind Gust Dir: NNE

Wind Gust Speed: 15.0

Wind Gust Time: 10:00

Live Transit Feeds

Location: USA

City: Los Angeles

Country: United States

Timezone: UTC -07:00

Wind Speed: 10.0

Wind Direction: N

Cloud Cover: 50%

Humidity: 40%

Pressure: 1013 hPa

Temperature: 22.0 °C

UV Index: 5.0

Visibility: 10.0 km

Wind Gust: 15.0 m/s

Wind Gust Dir: NNE

Wind Gust Speed: 15.0

Wind Gust Time: 10:00

Carried Data Fields

Location: USA

City: Los Angeles

Country: United States

Timezone: UTC -07:00

Wind Speed: 10.0

Wind Direction: N

Cloud Cover: 50%

Humidity: 40%

Pressure: 1013 hPa

Temperature: 22.0 °C

UV Index: 5.0

Visibility: 10.0 km

Wind Gust: 15.0 m/s

Wind Gust Dir: NNE

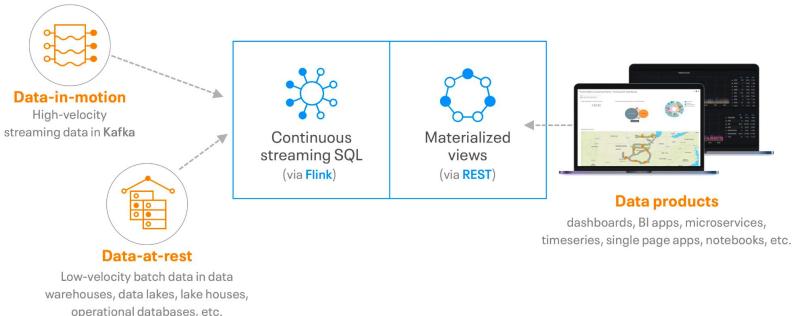
Wind Gust Speed: 15.0

Wind Gust Time: 10:00

BEST PRACTICES

STREAMING TECH DEBT TIPS

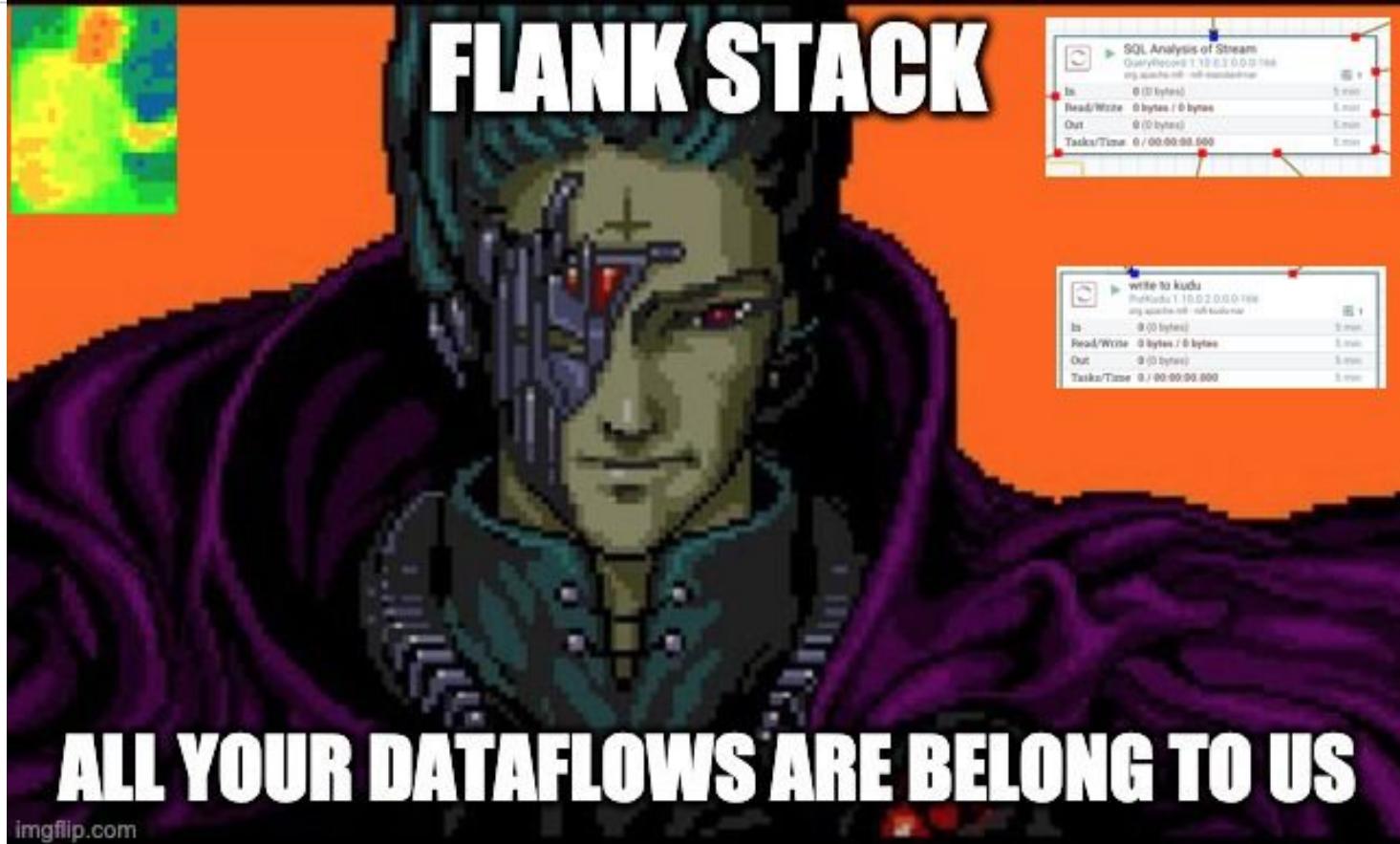
- Version Control All Assets
- Managed Public Cloud like Cloudera
- Use DevOps and APIs
- Latest Java and Python
- Stream Sizing (NiFi, Kafka, Flink)



RESOURCES AND WRAP-UP

Resources





ALL YOUR DATAFLOWS ARE BELONG TO US

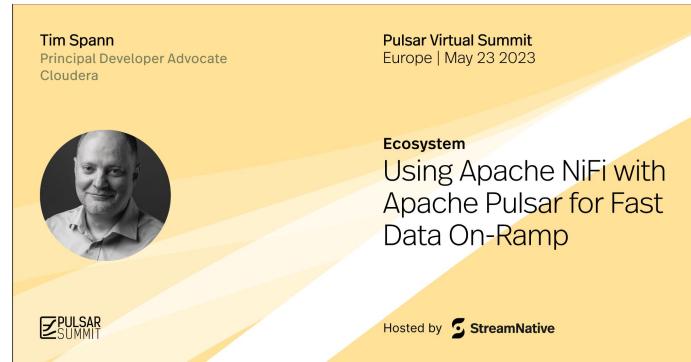
imgflip.com

Upcoming Events

May 10



May 23



May 25



Streaming Resources

- <https://dzone.com/articles/real-time-stream-processing-with-hazelcast-and-streamnative>
- <https://flipstackweekly.com/>
- <https://www.datainmotion.dev/>
- <https://www.flankstack.dev/>
- <https://github.com/tspannhw>
- <https://medium.com/@tspann>
- <https://medium.com/@tspann/predictions-for-streaming-in-2023-ad4d7395d714>
- https://www.apachecon.com/acna2022/slides/04_Spann_Tim_Citizen_Streaming_Engineer.pdf

Spring Things

- <https://spring.io/guides/gs/spring-boot/>
- <https://spring.io/projects/spring-amqp/>
- <https://spring.io/projects/spring-kafka/>
- <https://github.com/spring-projects/spring-integration-kafka>
- <https://github.com/spring-projects/spring-integration>
- <https://github.com/spring-projects/spring-data-relational>
- <https://github.com/spring-projects/spring-kafka>
- <https://github.com/spring-projects/spring-amqp>

DATA ENGINEER



CODER

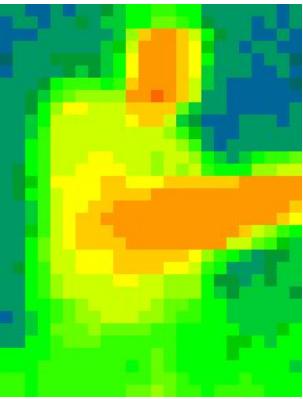


JAVA DEVELOPER



STREAMING ENGINEER

imgflip.com



TH^ON^G Y^OU[★]

