



**Stream
Native**

Hail Hydrate! From Stream to Lake with Pulsar and Friends

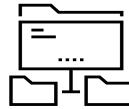
Tim Spann | Developer Advocate

<https://portotechhub.com/conference-2021>

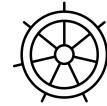
The Need For Real-Time Data



Hybrid and multi-cloud strategies with native geo-replication



Seamlessly build microservice architectures with support for streaming and messaging workloads



Built for Kubernetes
CloudNative migrations with tools



360 degree customer data
multi-tenancy, infinite retention, and extensive connector ecosystem



Tim Spann
Developer Advocate

DZone Zone Leader and Big Data
MVB Data DJay

- <https://www.datainmotion.dev/>
- <https://github.com/tspannhw/SpeakerProfile>
- <https://dev.to/tspannhw>
- <https://sessionize.com/tspann/>





- Founded the original developers of Apache Pulsar.
- Passionate and dedicated team.
- StreamNative helps teams to capture, manage, and leverage data using Pulsar's unified messaging and streaming platform.



Apache  **PULSAR** is an open source, cloud-native distributed messaging and streaming platform.

What are the Benefits of Pulsar?



Multi-Tenancy

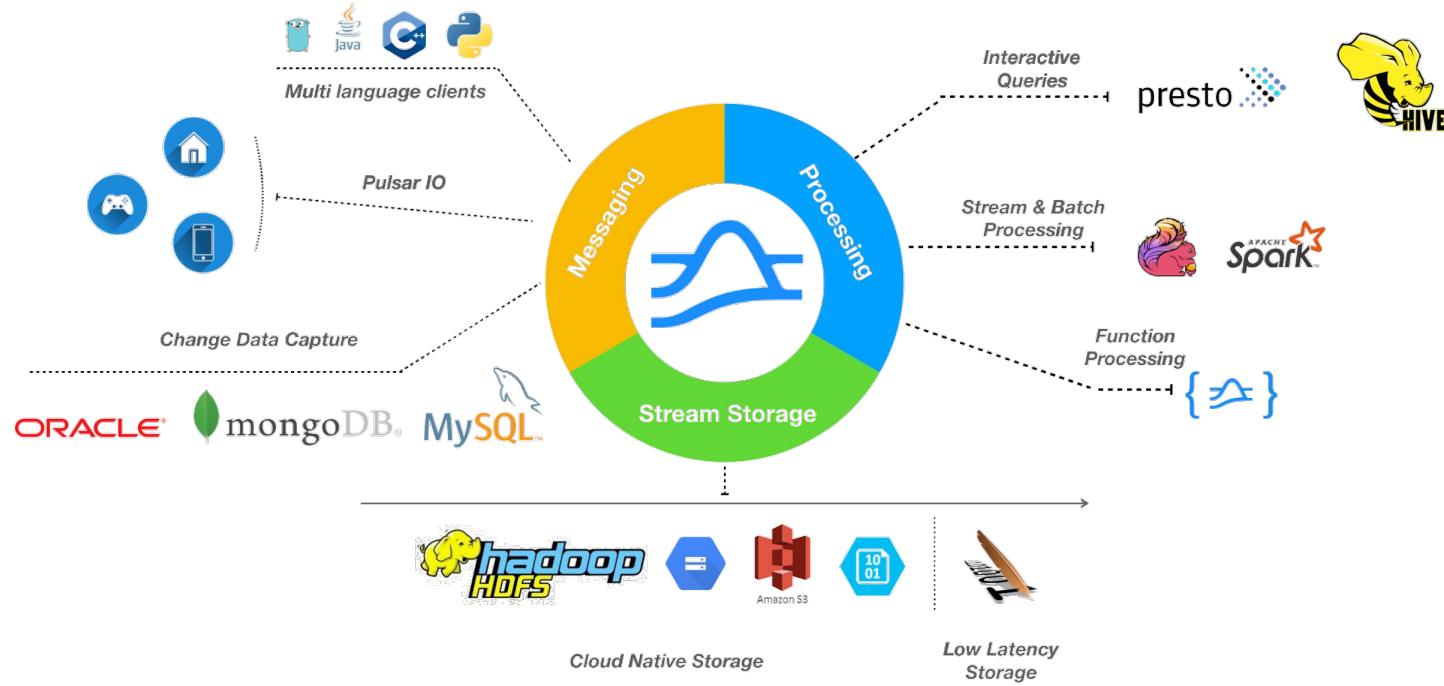
Scalability

Geo-Replication

Unified Messaging
Model

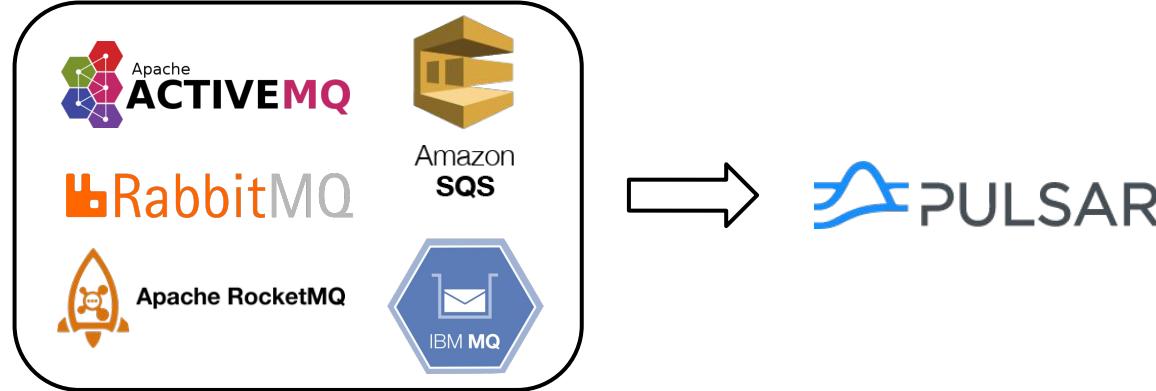
Data Durability

Apache Pulsar

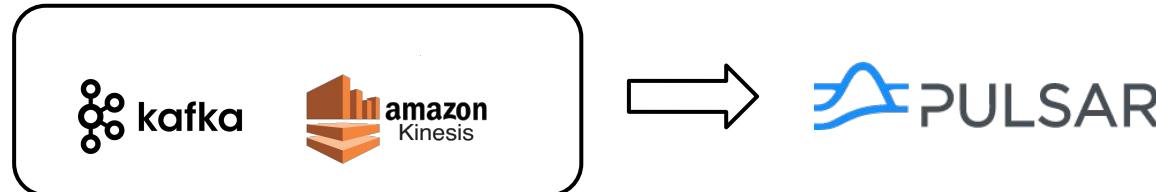


Top Pulsar Use Cases

#1 Message Queuing

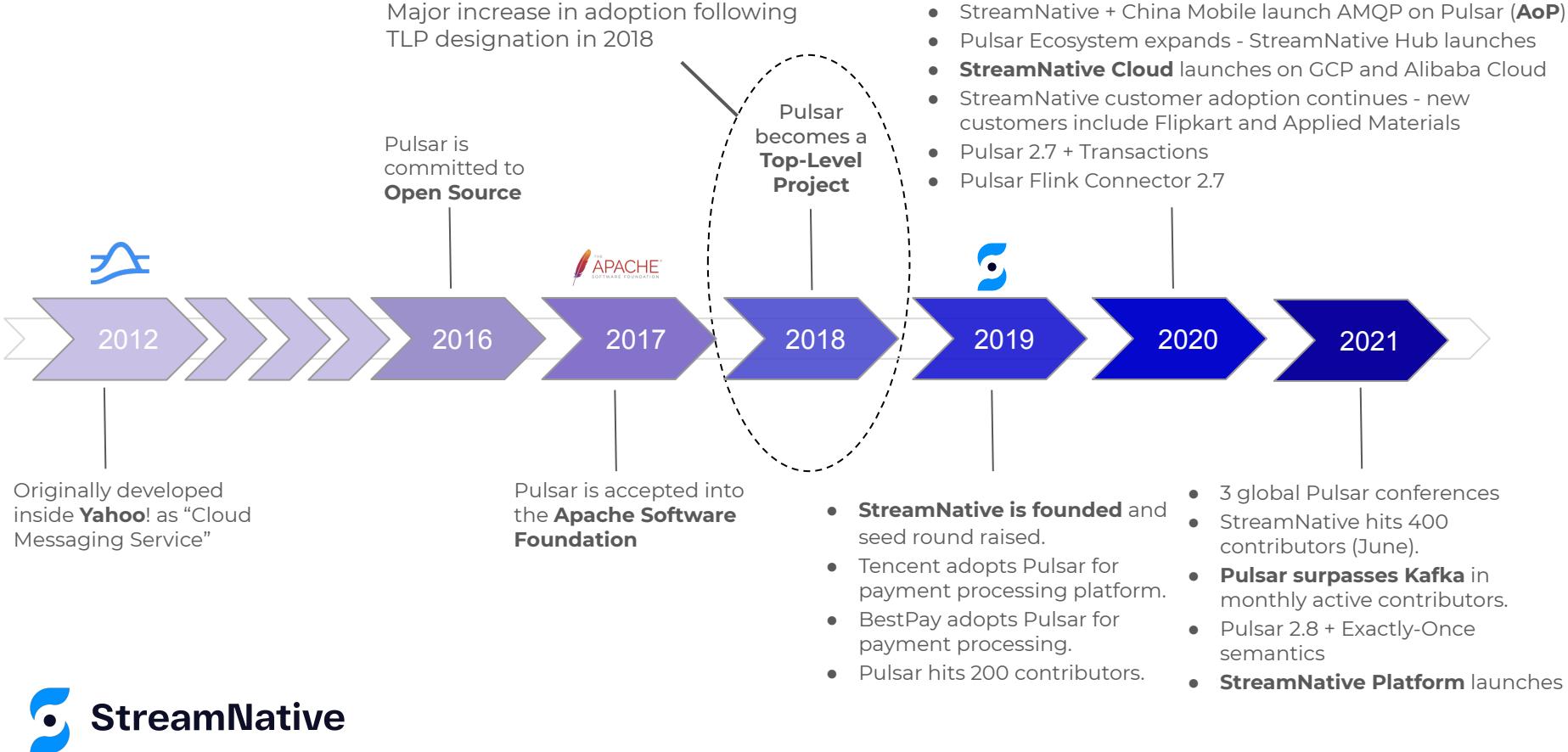


#2 Data Streaming



- Not built for the cloud
- Single tenant systems
- Monolithic architecture couples compute with storage
- Lack of geo replication support

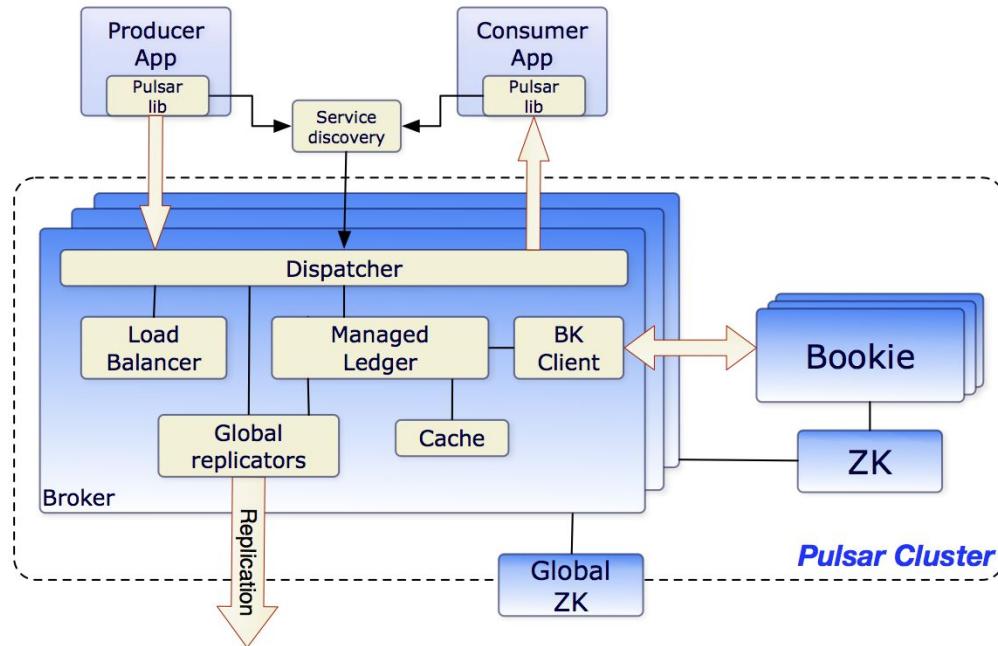
Key Milestones



Apache Pulsar Overview

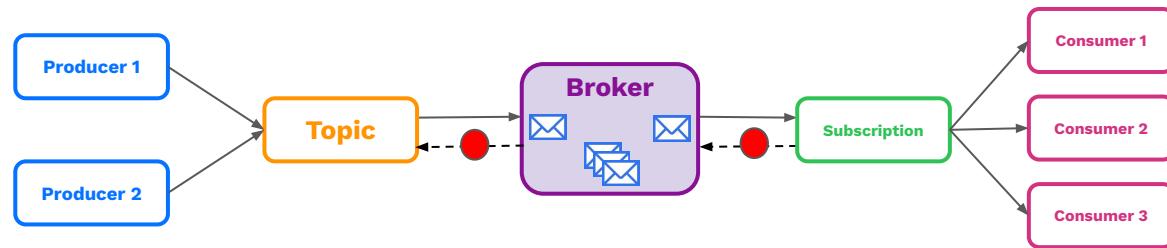
Enable Geo-Replicated Messaging

- Pub-Sub
- Geo-Replication
- Pulsar Functions
- Horizontal Scalability
- Multi-tenancy
- Tiered Persistent Storage
- Pulsar Connectors
- REST API
- CLI
- Many clients available
- Four Different Subscription Types
- Multi-Protocol Support
 - MQTT
 - AMQP
 - JMS
 - Kafka
 - ...



Pulsar's Publish-Subscribe model

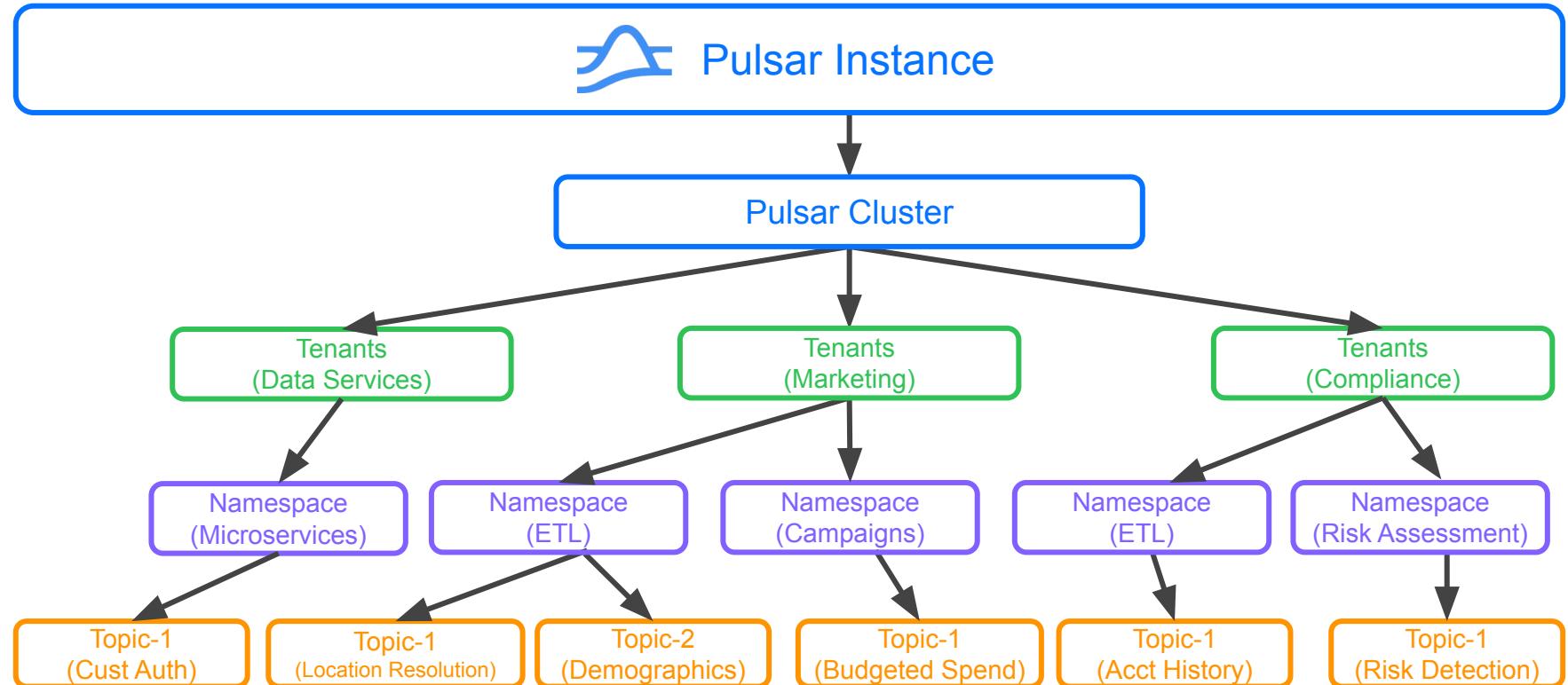
- Producers send messages.
- Topics are an ordered, named channel that producers use to transmit messages to subscribed consumers.
- Messages belong to a topic and contain an arbitrary payload.
- Brokers handle connections and routes messages between producers / consumers.
- Subscriptions are named configuration rules that determine how messages are delivered to consumers.
- Consumers receive messages.



What is the Pulsar Ecosystem?

- Functions and Connectors
 - Functions: Lightweight stream processing
 - Connectors: Part of “Pulsar IO”, includes “Source” and “Sink” APIs
 - Files, Databases, Data tools, Cloud Services, etc
- Protocol Handlers
 - Allows Pulsar to handle additional protocols by an extendable API running in the broker
 - AoP (AMQP), KoP (Kafka), MoP (MQTT)

Topics



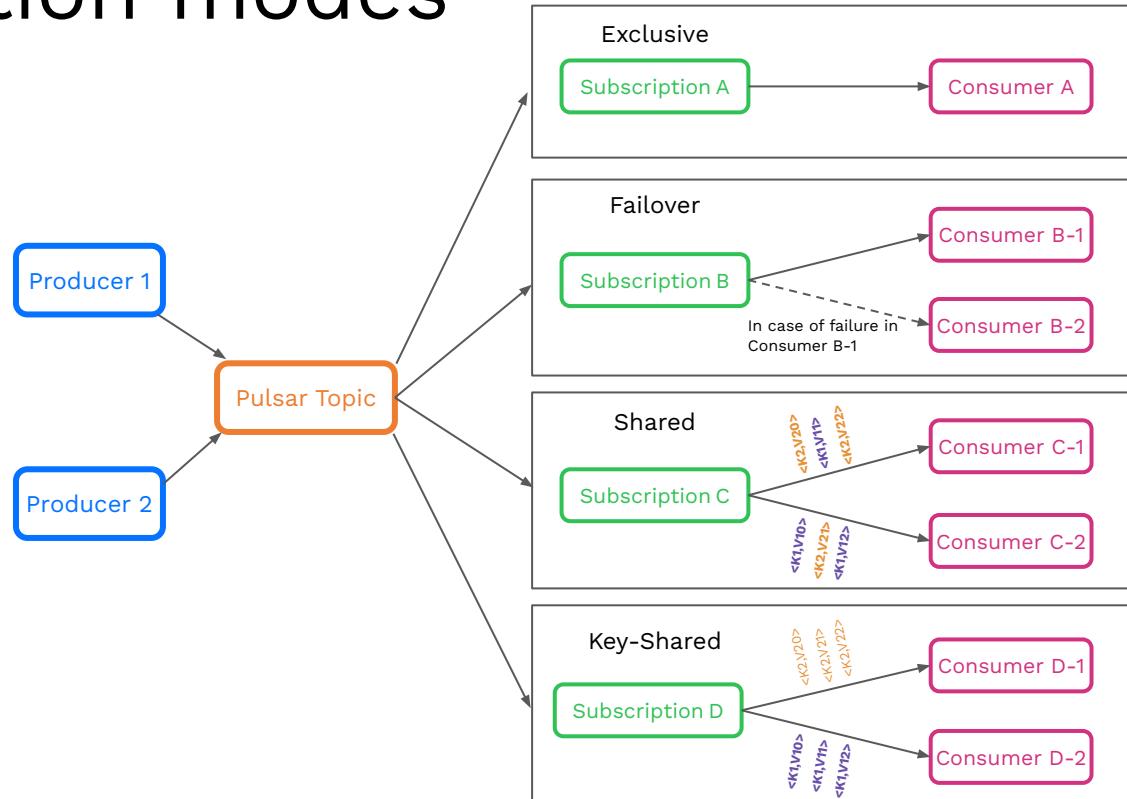
Pulsar subscription modes

Different subscription modes have different semantics:

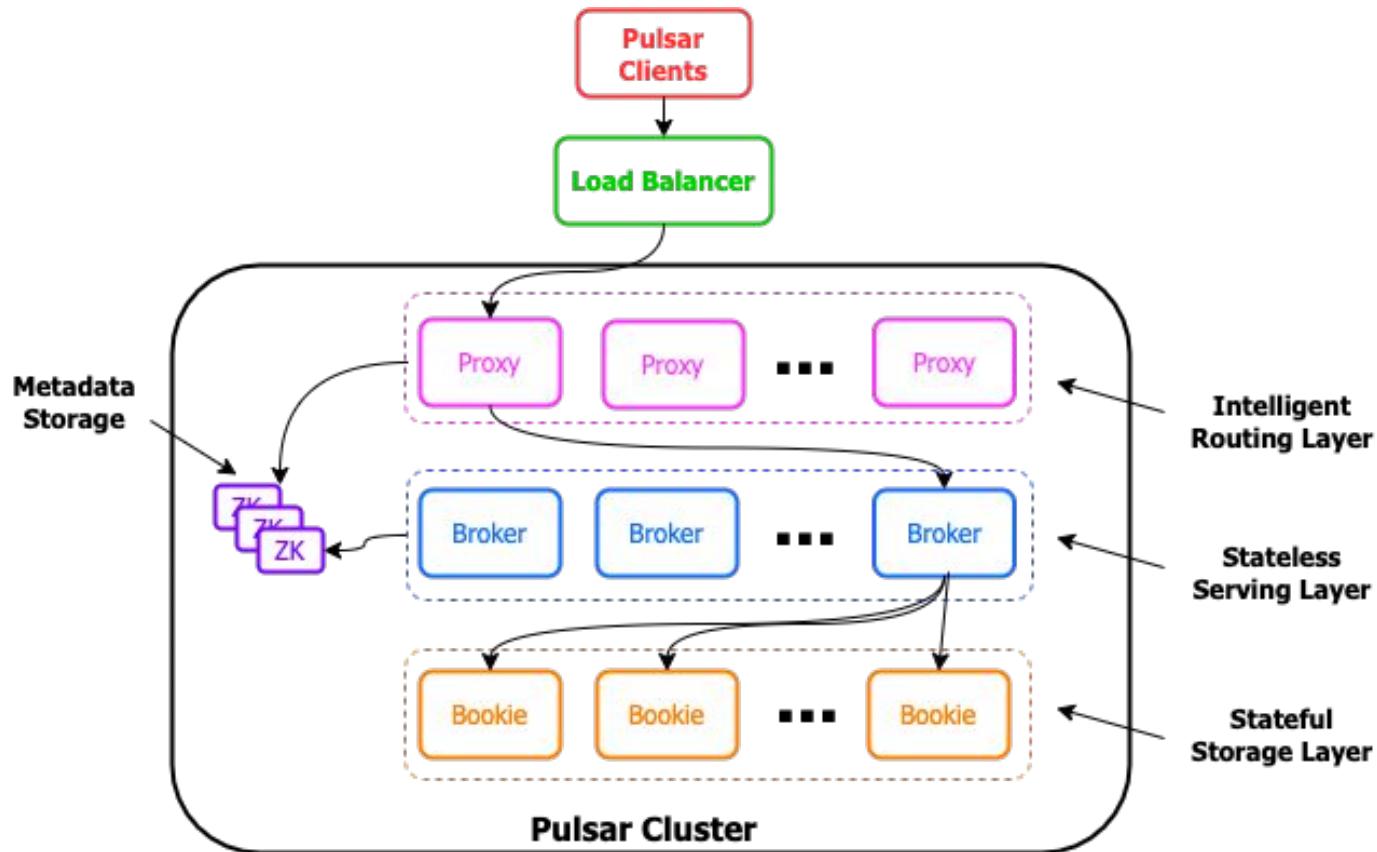
Exclusive/Failover - guaranteed order, single active consumer

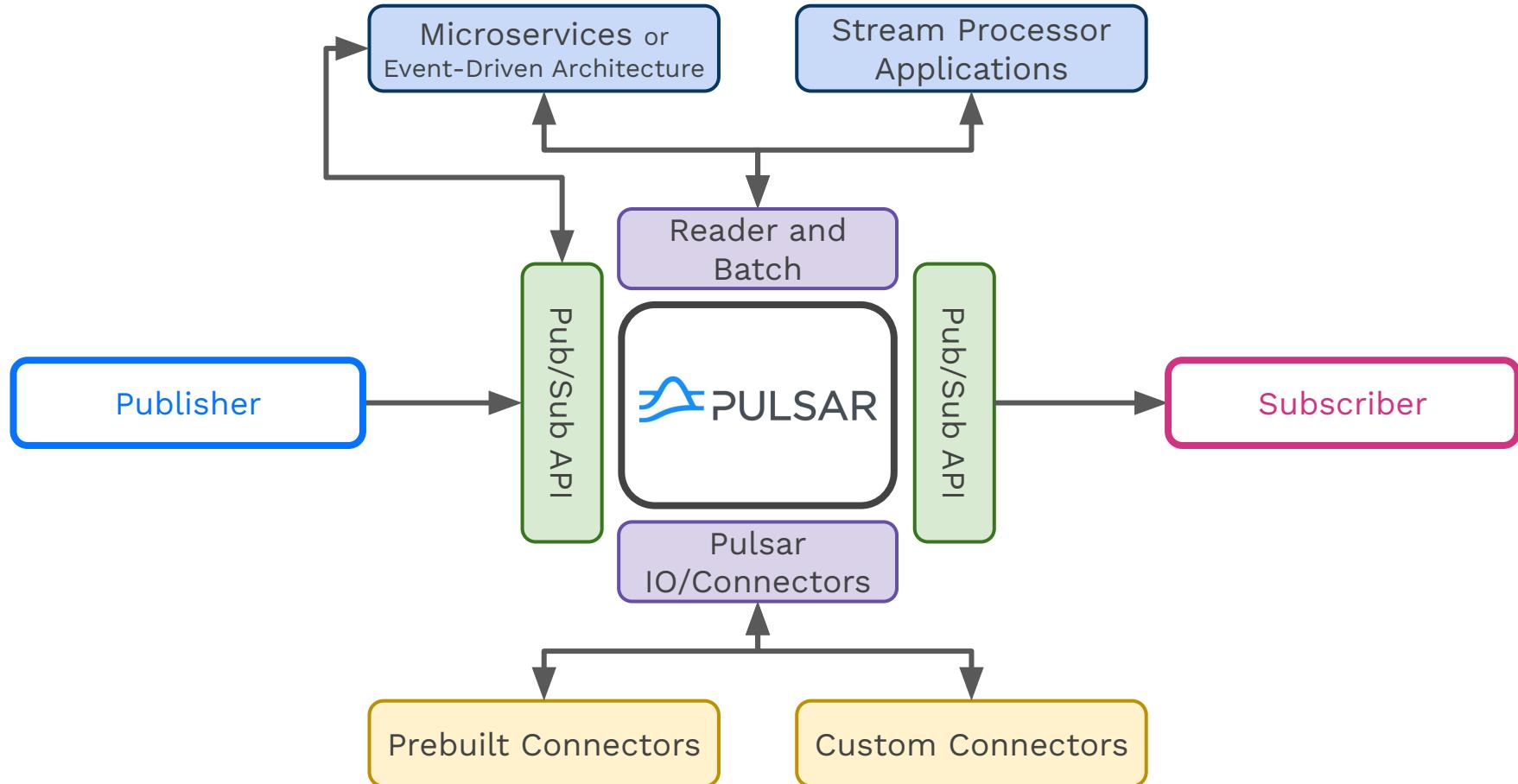
Shared - multiple active consumers, no order

Key_Shared - multiple active consumers, order for given key



Multi-Tiered Architecture





Moving Data In and Out of Pulsar

IO/Connectors are a simple way to integrate with external systems and move data in and out of Pulsar. <https://pulsar.apache.org/docs/en/io-jdbc-sink/>

- Built on top of Pulsar Functions
- Built-in connectors - hub.streamnative.io



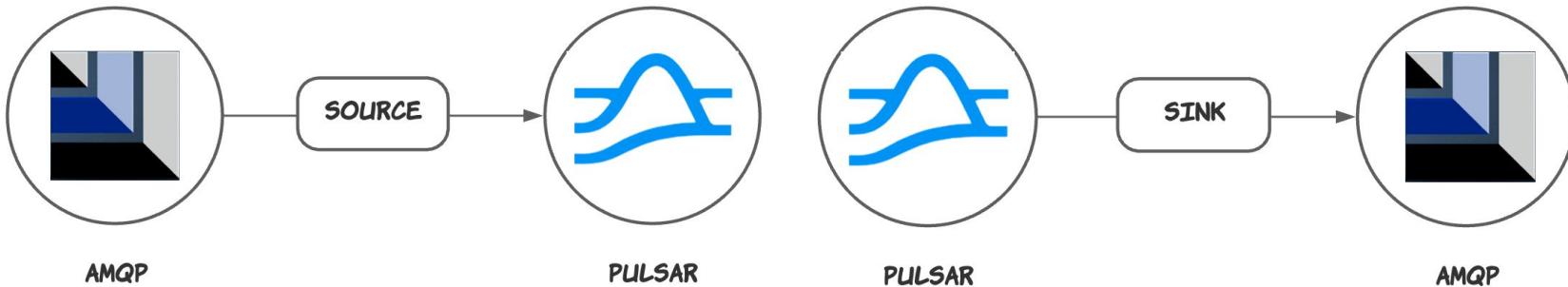
AMQP / RabbitMQ Protocol

AMQP on Pulsar (AoP)

<https://github.com/streamnative/aop>

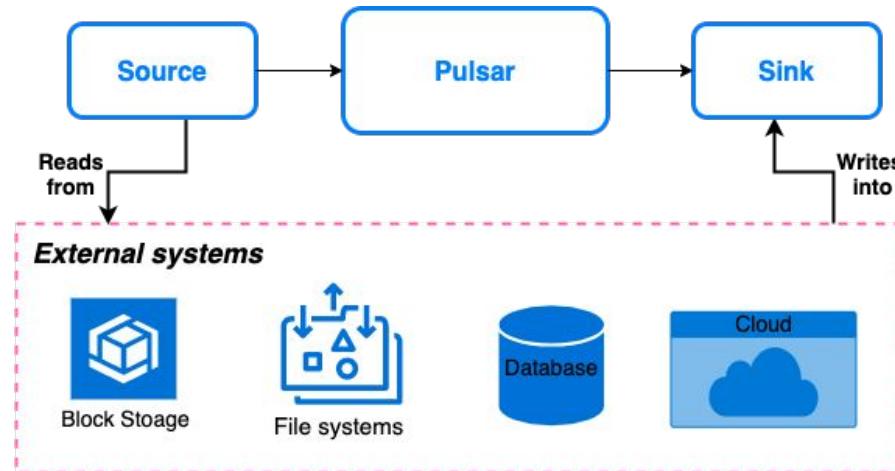
<https://hub.streamnative.io/connectors/amqp-1-0-sink/>

<https://hub.streamnative.io/connectors/amqp-1-0-source>



Use Azure BlobStore offloader with Pulsar

<https://pulsar.apache.org/docs/en/tiered-storage-azure/>



Apache Pulsar - Other Sinks



mongoDB



AWS Lambda



redis



AWS S3

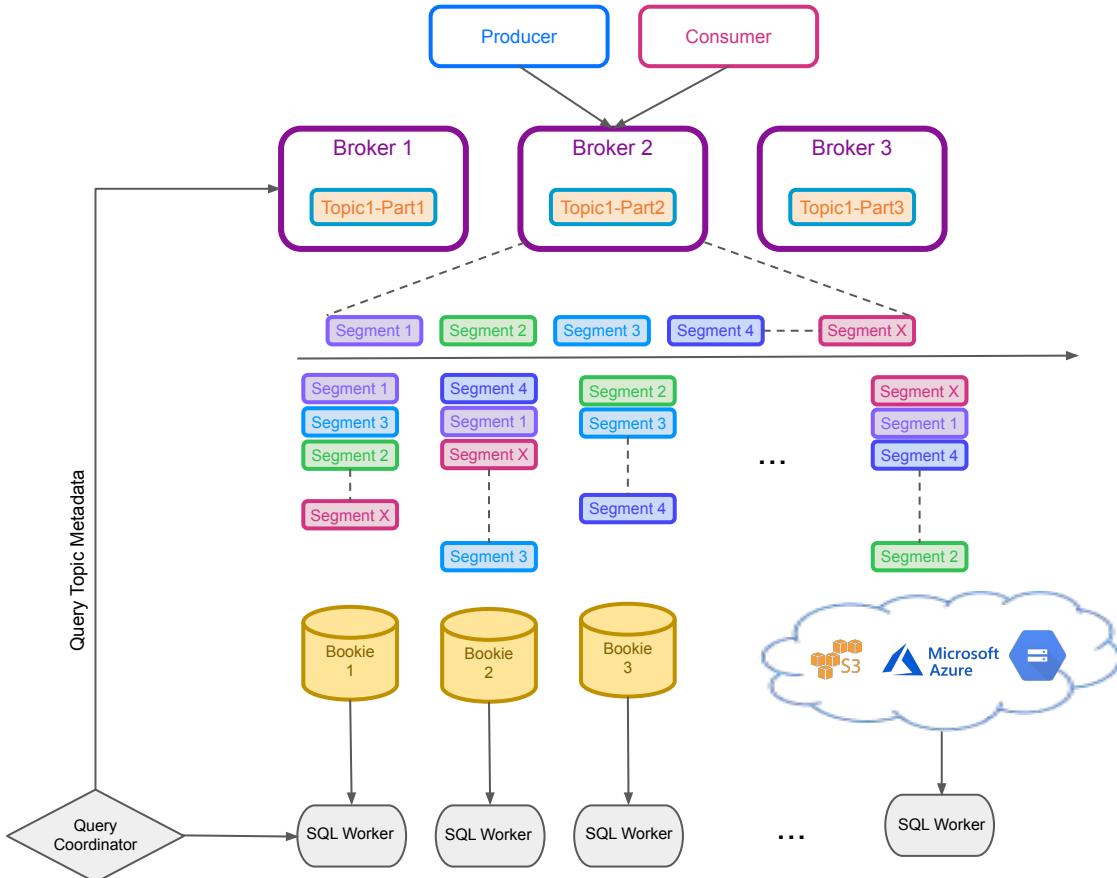


GCS

<https://hub.streamnative.io/connectors/cloud-storage-sink/2.5.1/>

Pulsar SQL

Presto/Trino workers can read segments directly from bookies (or offloaded storage) in parallel.



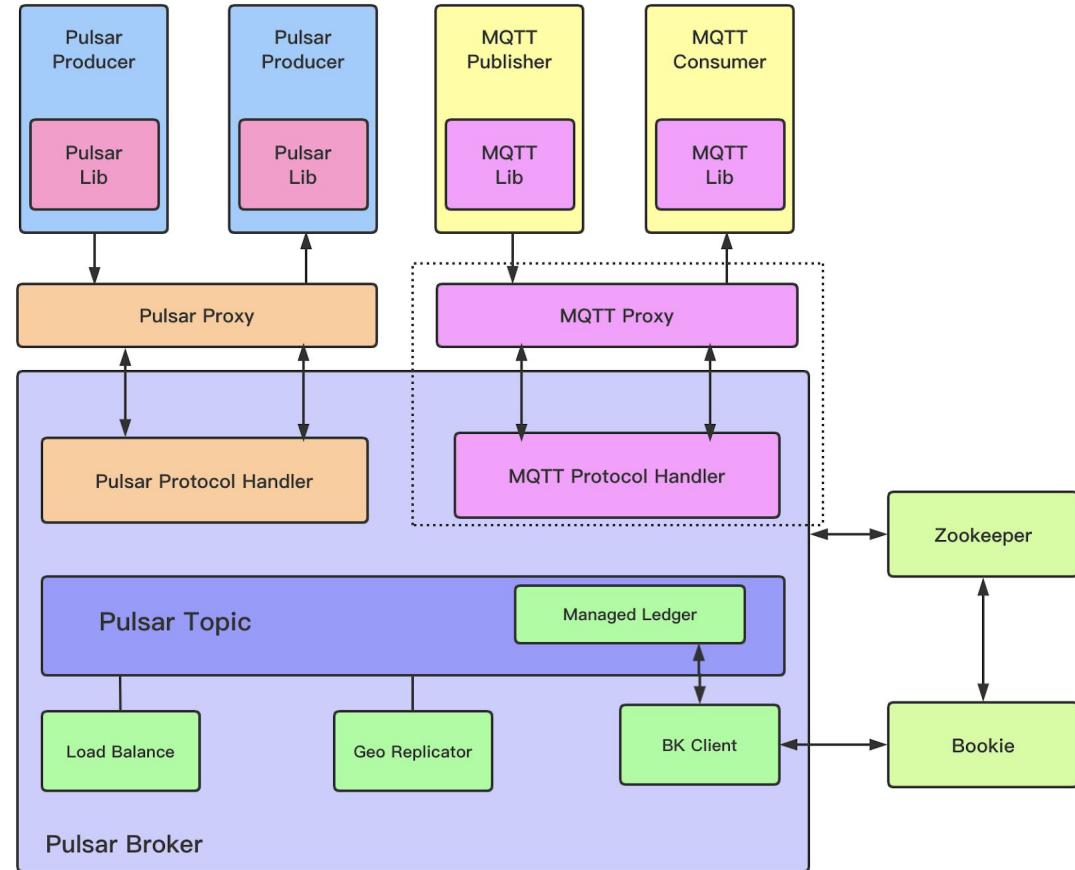
Query Your Topics with Pulsar SQL (Trino)

```
presto> select camera, cpu, cputempf, gputempf, memory, top1, top1pct, uuid, __publish_time__, __message_id__, __key__ from pulsar."public/default".iotjetsonjson;
      camera |   cpu | cputempf | gputempf |   memory |      top1 |    top1pct |          uuid | __publish_time__ | __message_id__ | __key__
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
/dev/video0 |  8.7 |   82 |  82 |  33.5 | microphone, mike | 18.85986328125 | xav_uuid_video0_lgl_20211001183019 | 2021-10-01 14:30:30.657 | (564,3,0) |
/dev/video0 |  8.7 |   82 |  82 |  33.6 | microphone, mike | 19.22607421875 | xav_uuid_video0_kpt_20211001183033 | 2021-10-01 14:30:44.380 | (564,4,0) |
/dev/video0 | 12.0 |   80 |  81 |  33.5 | microphone, mike | 12.53662109375 | xav_uuid_video0_gzd_20211001182930 | 2021-10-01 14:29:48.756 | (564,0,0) |
/dev/video0 |  8.5 |   82 |  82 |  33.6 | microphone, mike | 14.0625 | xav_uuid_video0_wlw_20211001182951 | 2021-10-01 14:30:02.919 | (564,1,0) |
/dev/video0 |  8.5 |   82 |  82 |  33.5 | microphone, mike | 29.8828125 | xav_uuid_video0_ulq_20211001183005 | 2021-10-01 14:30:16.787 | (564,2,0)
5 rows
[END]
```

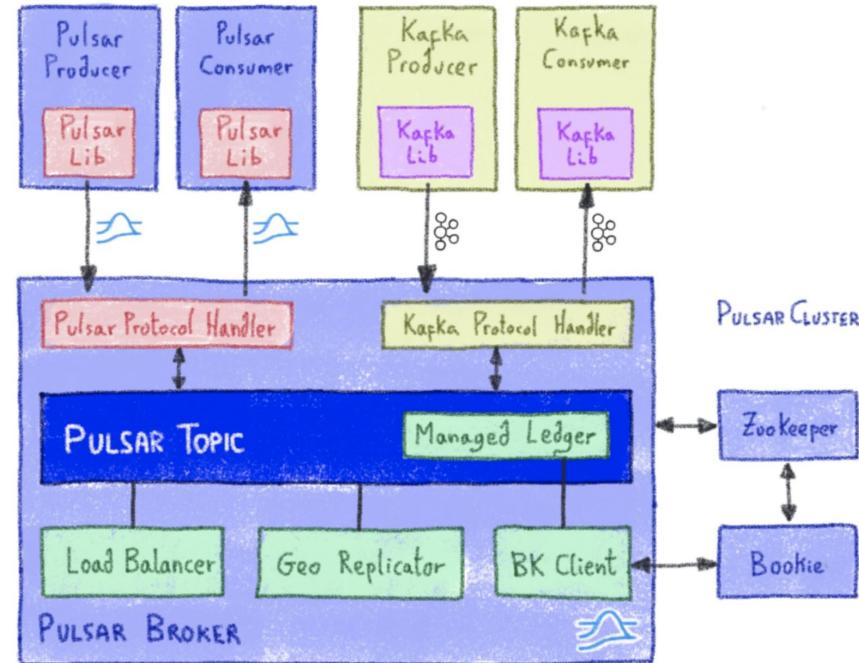
```
presto> show tables in pulsar."public/default";
      Table
-----
generator_test
iotjetsonjson
mqtt-2
(3 rows)

Query 20211001_054538_00008_s8x23, FINISHED, 1 node
Splits: 19 total, 19 done (100.00%)
0:00 [3 rows, 105B] [14 rows/s, 493B/s]
```

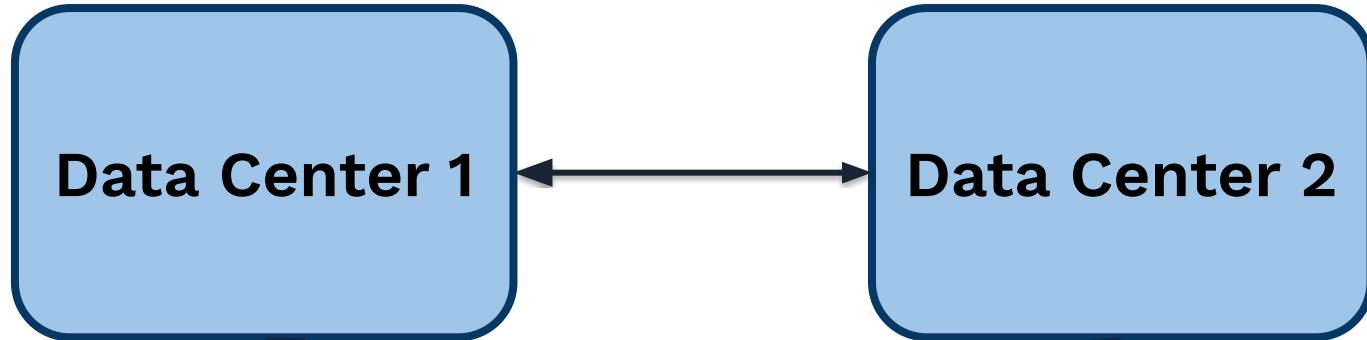
MQTT on Pulsar (MoP)



Kafka-on-Pulsar (Kop)

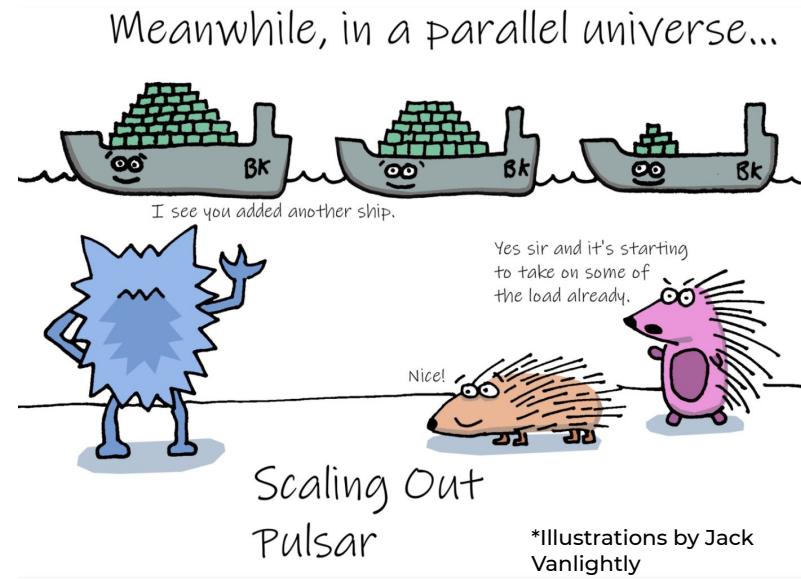
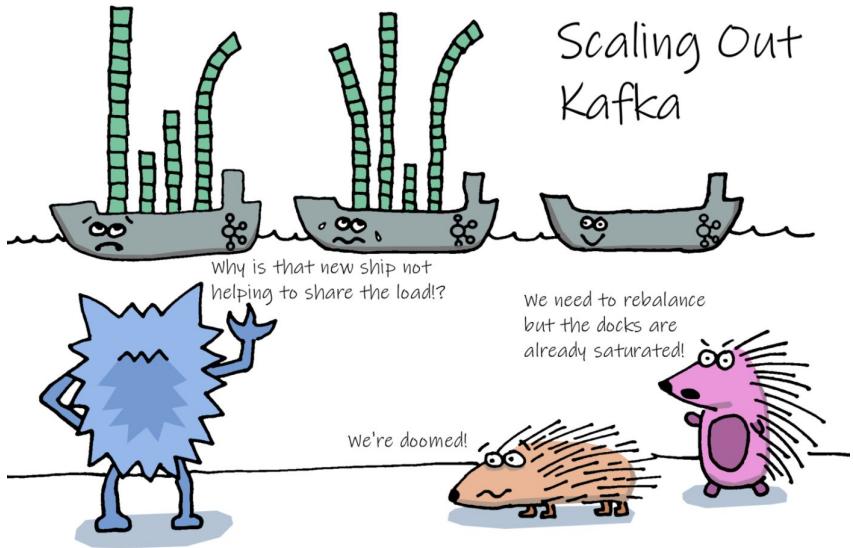


Geo Replication



Replication is done asynchronously.

Pulsar has built-in cross data center replication that is used in production already.



Pulsar is built for easy scale-out.

StreamNative
Cloud



**Stream
Native
Cloud**

StreamNative Cloud

Powered by Apache Pulsar, StreamNative provides a cloud-native, real-time messaging and streaming platform to support multi-cloud and hybrid cloud strategies.



Cloud Native



kubernetes

Built for Containers



Flink

Flink SQL

The unified messaging and streaming platform made by the creators of Apache Pulsar.

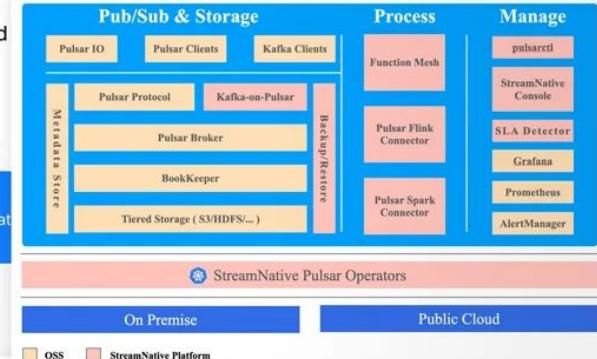
Built for Kubernetes. Made for the cloud. Enables multi-cloud and hybrid

Take A Tour

Contact Sales

Product Update

StreamNative Cloud on AWS Marketplace. Leverage Pulsar on the largest cloud provider with StreamNative



StreamNative, Powered by Apache Pulsar

StreamNative
Cloud

Apache Pulsar as a service, StreamNative Cloud delivers a resilient and scalable messaging and event streaming service deployable in minutes.

StreamNative
Platform

StreamNative Platform is a cloud-native messaging and event streaming platform built by the original creators for Apache Pulsar.

StreamNative
Pro Services

Accelerate your messaging and streaming platform development and drive business results with help from StreamNative's Pulsar experts.

InfoWorld

BOSSIE
2021 AWARDS



StreamNative

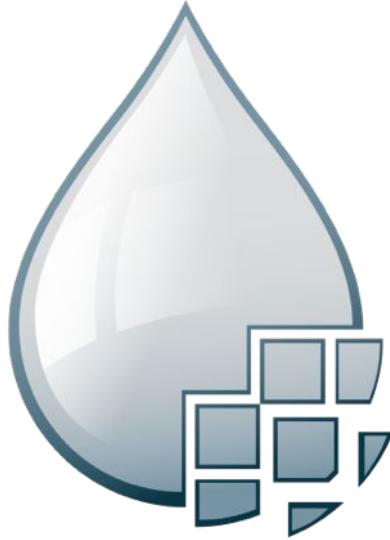
Apache NiFi



Don't Be Afraid of Open Source

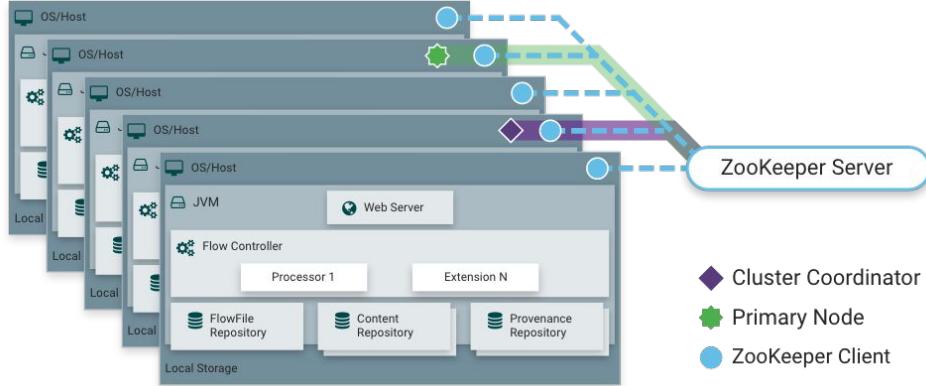
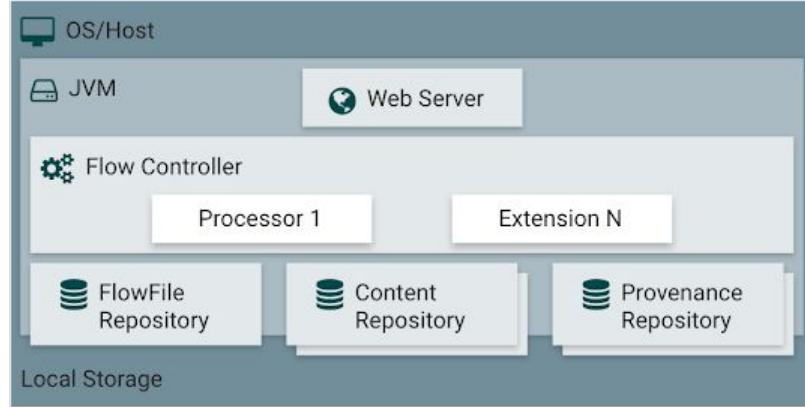


Why Apache NiFi?



- Guaranteed delivery
- Data buffering
 - Backpressure
 - Pressure release
- Prioritized queuing
- Flow specific QoS
 - Latency vs. throughput
 - Loss tolerance
- Data provenance
- Supports push and pull models
- Hundreds of processors
- Visual command and control
- Over a sixty sources
- Flow templates
- Pluggable/multi-role security
- Designed for extension
- Clustering
- Version Control

Architecture



<https://nifi.apache.org/docs/nifi-docs/html/overview.html>

Provenance

NiFi Data Provenance

Displaying 165 of 165
Oldest event available: 12/21/2020 16:55:33 UTC

Filter by component name ▾

Date/Time	Type	Flowfile Uuid	Size
12/22/2020 16:54:17.193 UTC	ATTRIBUTES_MODIFIED	6fbbae84f6ba4-47c3-ba03-8830ec7cd3db	89 byte
12/22/2020 16:54:17.192 UTC	ATTRIBUTES_MODIFIED	1233e8d4e84d-4218-b3d0-2598e7f901e	87 byte
12/22/2020 16:54:14.194 UTC	ATTRIBUTES_MODIFIED	37fbca2153-4185-4460-b633-7eb974ad8718	81 byte
12/22/2020 16:54:05.297 UTC	ATTRIBUTES_MODIFIED	699d6fce9d71-4cf6-b733-b546e05d362	83 byte
12/22/2020 16:53:59.296 UTC	ATTRIBUTES_MODIFIED	d43a05c5-5aae-44c2-9edc-c20a8148604	84 byte
12/22/2020 16:53:59.295 UTC	ATTRIBUTES_MODIFIED	4b1db8b1c1f83-3-a09-8309-2da1277cd7c6	84 byte
12/22/2020 16:53:58.296 UTC	ATTRIBUTES_MODIFIED	45fe8edd-cx55-431e-82b9-436c4a4092e	81 byte
12/22/2020 16:53:57.298 UTC	ATTRIBUTES_MODIFIED	b07034b-6361-4c34-b	
12/22/2020 16:53:57.297 UTC	ATTRIBUTES_MODIFIED	d12601a-c793-4c16-b	
12/22/2020 16:53:57.297 UTC	ATTRIBUTES_MODIFIED	29966d0-4153-41bc-a	
12/22/2020 16:53:43.753 UTC	ATTRIBUTES_MODIFIED	1ca5c744-1cb4-4ff1-bb	
12/22/2020 16:53:37.747 UTC	ATTRIBUTES_MODIFIED	faf647db-9e65-48c0-a	
12/22/2020 16:53:21.646 UTC	ATTRIBUTES_MODIFIED	df1f60ff-6d65-460e-95	
12/22/2020 16:53:05.515 UTC	ATTRIBUTES_MODIFIED	964695fc-d953-44c0-b	
12/22/2020 16:52:43.374 UTC	ATTRIBUTES_MODIFIED	79fcfa90-b160-4fc4-ba	
12/22/2020 16:52:29.308 UTC	ATTRIBUTES_MODIFIED	3453eeb9-953c-4952-a	
12/22/2020 16:52:29.307 UTC	ATTRIBUTES_MODIFIED	a166e297-118a-4262-9	
12/22/2020 16:52:29.307 UTC	ATTRIBUTES_MODIFIED	bd2946fd-5a99-42d7-b	
12/22/2020 16:52:29.307 UTC	ATTRIBUTES_MODIFIED	a16841bc-2505-4c8c-b	
12/22/2020 16:52:29.306 UTC	ATTRIBUTES_MODIFIED	578540fe-e449-471f-a	
12/22/2020 16:52:29.306 UTC	ATTRIBUTES_MODIFIED	3d44c5fb-4737-4a9e-82	
12/22/2020 16:52:29.306 UTC	ATTRIBUTES_MODIFIED	4dc93a17-7059-424e-9	
12/22/2020 16:52:29.306 UTC	ATTRIBUTES_MODIFIED	9fb1d9c1-f304-4c11-93	

Provenance Event

DETAILS ATTRIBUTES

Attribute Values

lastprice	123.66
symbol	No value set
IBM	No value set
timestamp	1608654962884
volume	No value set

<https://www.datainmotion.dev/2021/01/automating-starting-services-in-apache.html>

Backpressure & Prioritizers

Configure Connection

DETAILS SETTINGS

Name:

Id: 3ca22430-cba4-3347-b45b-7bdc3530bd7e

FlowFile Expiration: 0 sec

Available Prioritizers:

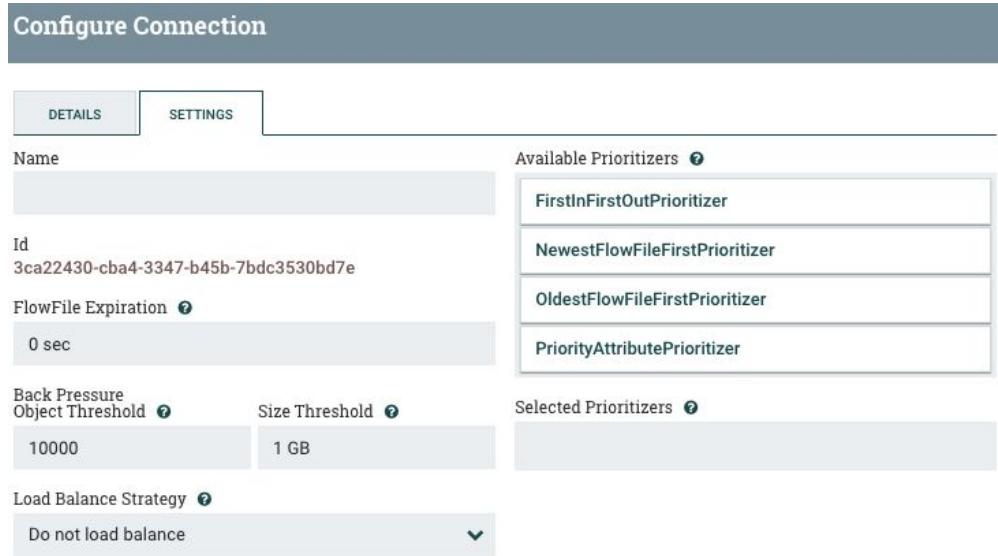
- FirstInFirstOutPrioritizer
- NewestFlowFileFirstPrioritizer
- OldestFlowFileFirstPrioritizer
- PriorityAttributePrioritizer

Selected Prioritizers:

Back Pressure Object Threshold: 10000

Size Threshold: 1 GB

Load Balance Strategy: Do not load balance



<https://www.datainmotion.dev/2019/11/exploring-apache-nifi-110-parameters.html>

Record Processors

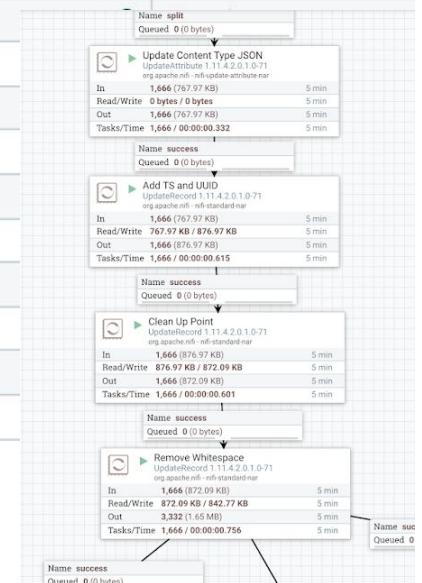
- XML, CSV, JSON, AVRO and more
- Schemas or Inferred Schemas
- Easily convert between them
- Support SQL with Apache Calcite

Property	Value
Record Reader	XMLReader
Record Writer	JsonRecordSetWriter
Include Zero Record FlowFiles	false
Cache Schema	true
query1	SELECT * FROM FLOWFILE

<https://www.datainmotion.dev/2019/03/advanced-xml-processing-with-apache.html>



Property	Value
Schema Access Strategy	Infer Schema
Schema Registry	AvroSchemaRegistry
Schema Name	\$(schema.name)
Schema Version	
Schema Branch	
Schema Text	
Schema Inference Cache	
Expect Records as Array	
Attribute Prefix	
Field Name for Content	
Date Format	
Time Format	
Timestamp Format	



The screenshot shows a complex Apache Nifi flow. It starts with an 'UpdateContent' processor (Type: JSON) which reads from a flowfile and writes to it. This is followed by an 'Add TS and UUID' processor, another 'UpdateContent' processor, a 'Clean Up Point' processor, and finally a 'Remove Whitespace' processor. Each processor has its own performance metrics displayed below it, such as read/write times and task counts.

Record Processors



Configure Processor

⚠ Invalid

SETTINGS SCHEMAS

Required field

Add Controller Service

Property

Record Reader

Record Destination S...

Include Zero Record I...

Requires Controller Service
RecordReaderFactory 1.13.0 from org.apache.nifi - nifi-standard-services-api-nar

Compatible Controller Services

- AvroReader 1.13.0
- CSVReader 1.13.0
- GrokReader 1.13.0
- JsonPathReader 1.13.0
- JsonTreeReader 1.13.0
- ParquetReader 1.13.0
- ReaderLookup 1.13.0
- ScriptedReader 1.13.0
- Syslog5424Reader 1.13.0
- SyslogReader 1.13.0
- WindowsEventLogReader 1.13.0
- XMLReader 1.13.0

Property

Record Reader

Record Destination S...

Include Zero Record I...

RecordSinkService 1.13.0 from org.apache.nifi - nifi-standard-services-api-nar

Compatible Controller Services

RecordSinkServiceLookup 1.13.0

Controller Service Name

RecordSinkServiceLookup

Bundle

org.apache.nifi - nifi-record-sink-service-nar

<https://www.datainmotion.dev/2019/03/advanced-xml-processing-with-apache.html>

Consume MQTT



This could read from Apache Pulsar - MoP (MQTT on Pulsar)

Property	ConsumeMQTT Processor	Value
Session state	?	Clean Session
MQTT Specification Version	?	AUTO
Connection Timeout (seconds)	?	30
Keep Alive Interval (seconds)	?	60
Group ID	?	No value set
Topic Filter	?	No value set
Quality of Service(QoS)	?	0 - At most once
Max Queue Size	?	No value set
Record Reader	?	No value set
Record Writer	?	No value set
Add attributes as fields	?	true
Message Demarcator	?	No value set

Apache MXNet Native Processor through DJL.AI for Apache NiFi



#workshop

Deep Learning Class Label: person

File: cc0a469f-c108-42c7-95c6-10e5fda95006.person.png

Probability: 0.96

UUID: 32ef65a3-0650-42cd-965c-ba25597eb1ad

Rank: 1

Bounding Box (Height/Width, X,Y)

0.74 / 0.69

0.27, 0.25

Image (Height/Width, X,Y)

480 / 640

0, 0

tspann 11:30 AM

371bdb8f-35bc-4a2a-919c-bdeb609b726c.person.png

```
private void runModelUserInput() {
    PathPaths.get(url).toAbsolutePath().getParent().toString());
}

private void runModelUserInput() {
    PathPaths.get(url).toAbsolutePath().getParent().toString());
    testRunner.setVariable("expressionUsage", false);
    testRunner.run();
    testRunner.assertValid();
}

testRunner.assertAllFlowFilesTransferred(DeepLearningProcessor.REL_SUCCESS);
List<MockFlowFile> successfuls = testRunner.getFlowFilesForRelationship(DeepLearningProcessor.REL_SUCCESS);

for (MockFlowFile mockFile : successfuls) {
    assertEquals("car", mockFile.getAttribute("category"));
    assertEquals("1.0", mockFile.getAttribute("probability"));

    System.out.println("MockFlowFile: " + mockFile);
    Map<String, String> attributes = mockFile.getAttributes();
    for (String attribute : attributes.keySet()) {
        System.out.println("Attribute: " + attribute);
    }
}

@test
public void testProcessor() throws Exception {
    MockFlowFile resourcesDirectory = new MockFlowFile();
    System.out.println(resourcesDirectory.getAbsolutePath());
    testRunner.setProperty(DeepLearningProcessor.MOCKDIR, resourcesDirectory.getAbsolutePath());
    testRunner.setProcessor(DeepLearningProcessor.DATASET, DeepLearningProcessorTest.testProcessor);
}

Run: DeepLearningProcessorTest.testProcessor
```

Tests passed: 1 of 1 test - 4 s 0.0 ms

Size:17632B

Attribute:boundingbox_height_1 = 0.35

Attribute:boundingbox_width_1 = 1.00

Attribute:probability_1 = 1.00

Attribute:rank_1 = 0

Attribute:image_min_x = 0

Attribute:class_1 = car

Attribute:rank_3 = 1

Attribute:uuid = e9993c52-f5ab-4849-8876-a25796714984

Attribute:boundingbox_width_3 = 0.24

Attribute Values

boundingbox_height_1

0.99

No value set

boundingbox_width_1

0.90

No value set

boundingbox_x_1

0.09

No value set

boundingbox_y_1

0.01

No value set

class_1

tmonitor

No value set

filename

2020-08-26_1330.jpg.tmonitor.png

2020-08-26_1330.jpg (previous)

This processor uses the DJL.AI Java Interface

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

<https://dev.to/tspannhw/easy-deep-learning-in-apache-nifi-with-djl-2d79>

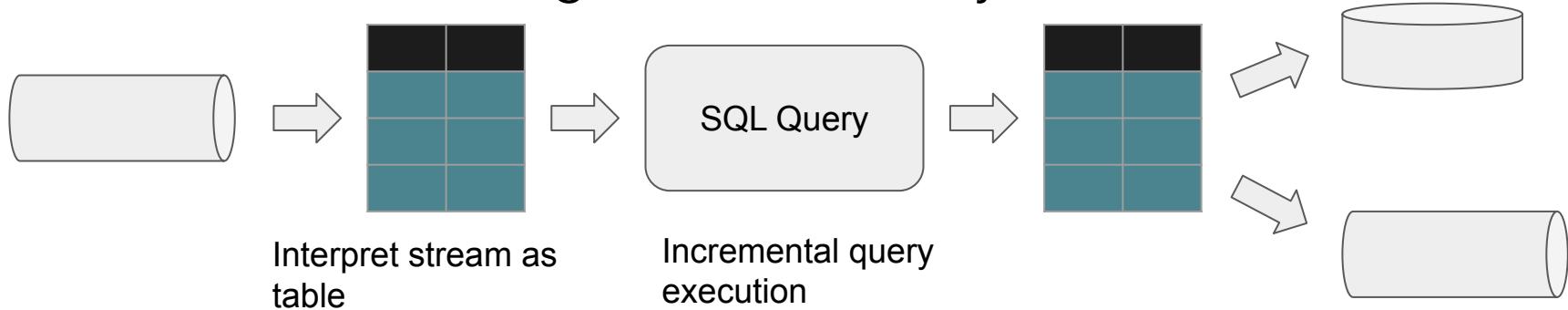
Apache Flink



Flink

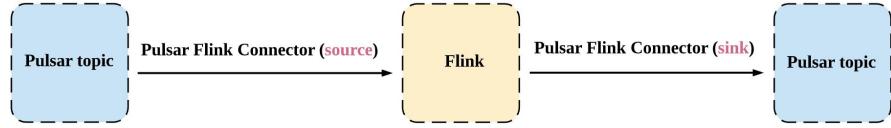
<https://flink.apache.org/2021/01/07/pulsar-flink-connector-270.html>

SQL / Table API: Running The Same Query On Streams

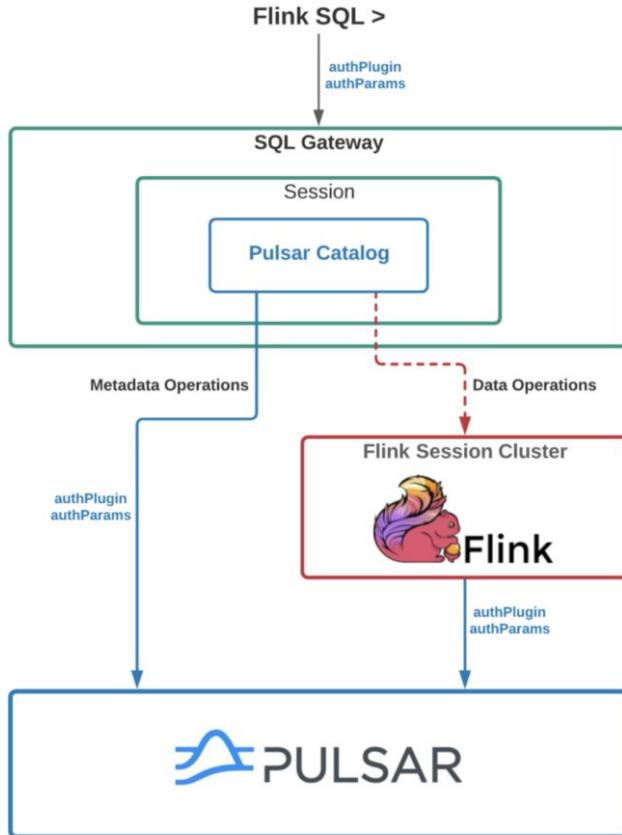


```
SELECT
    room,
    TUMBLE_END(rowtime, INTERVAL '1' HOUR),
    AVG(temperature)
FROM
    sensors
GROUP BY
    TUMBLE(rowtime, INTERVAL '1' HOUR), room
```

Flink SQL To Pulsar Catalog



Screenshot of the Apache Flink Dashboard showing a running job named 'xenodochial_noxye'. The job ID is aa643654ed280f8fc73bf9090467bb4. It started at 2021-04-07 10:08:37 and has a duration of 3h 6m 21s. The configuration shows a complex Flink pipeline involving KafkaSource, Weather API, and various Flink Process blocks. The dashboard provides real-time metrics like records sent, bytes sent, and task status.





StreamNative

StreamNative Cloud

International Organization Instance

Catalogs flink-test /

Create Flink Cluster

public/default

Tables

- data-gen-out
- jetsonjson
- jetsoniot
- jetsoniot2
- kinesis-input
- kinesis-output
- orders
- product
- test1
- test3
- [TENANT_NAMESPACE] ...

Execution Type streaming Run Stop

```
1 select cputempf, gputempf, diskusage, cpu, systemtime, uuid
2 from jetsoniot2
3 where cputempf > 105
4
```

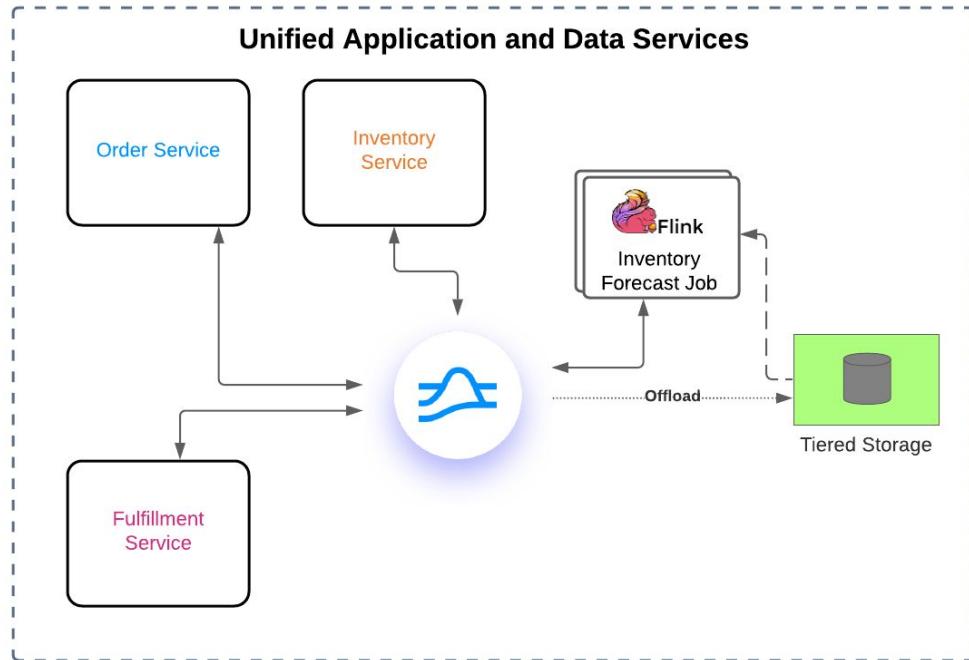
Result

cputempf	gputempf	diskusage	cpu	systemtime	uuid
106	106	33026.3 MB	14.8	09/27/2021 16:17:42	xai_21
106	106	33026.3 MB	15	09/27/2021 16:18:05	xai_21
106	106	33026.3 MB	15.2	09/27/2021 16:18:31	xai_20
106	106	33026.3 MB	15	09/27/2021 16:18:53	xai_21
106	106	33026.3 MB	15.4	09/27/2021 16:19:31	xai_20
106	106	33026.3 MB	14.8	09/27/2021 16:19:55	xai_20
106	107	33026.3 MB	14.7	09/27/2021 16:20:19	xai_21
106	107	33026.3 MB	15.3	09/27/2021 16:20:47	xai_21
107	107	33026.3 MB	13.9	09/27/2021 16:21:10	xai_20
107	107	33026.3 MB	15.3	09/27/2021 16:21:33	xai_21

Best Practice Architectures

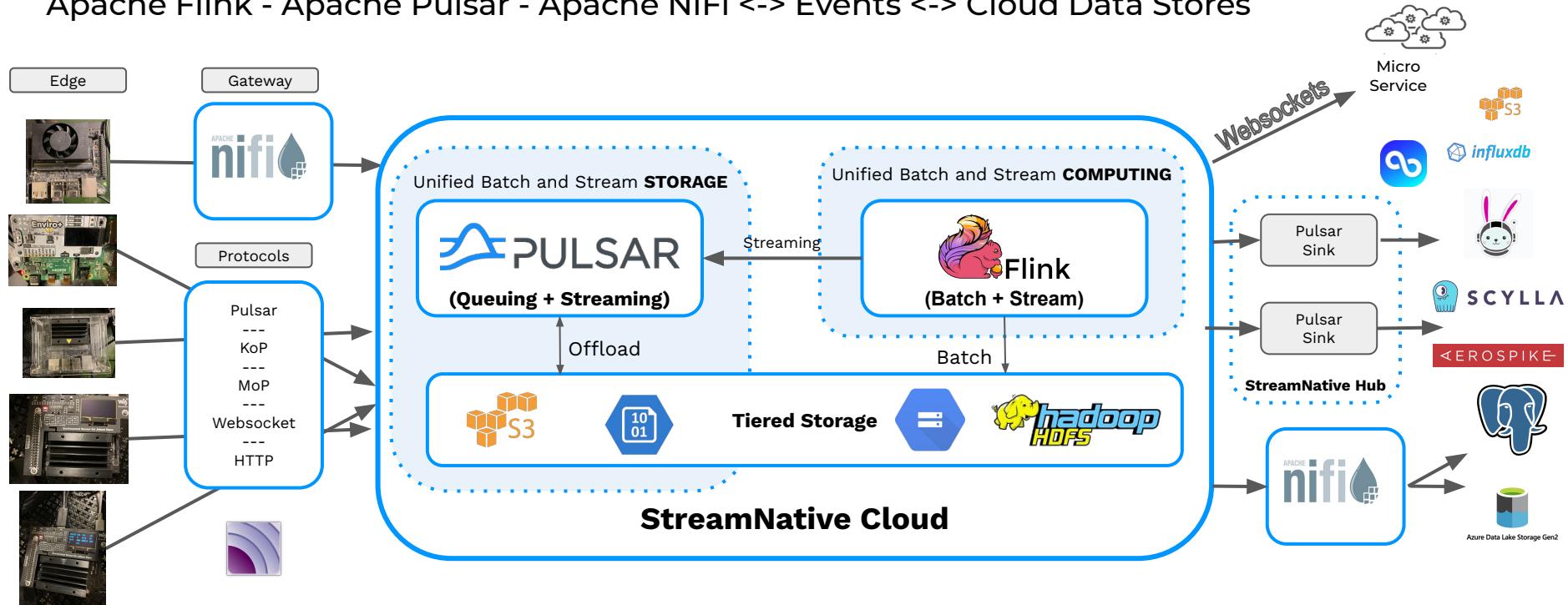
Example: E-Commerce with Pulsar

- Unified storage with access to underlying data
- Native tiered storage
- Single system to exchange data
- Teams share toolset



End-to-End Streaming FLiP(N) Apps

Apache Flink - Apache Pulsar - Apache NiFi <-> Events <-> Cloud Data Stores



Demo



StreamNative



Ingesting IoT Data via Java Pulsar

```
UUID uuidKey = UUID.randomUUID();
String pulsarKey = uuidKey.toString();
String OS = System.getProperty("os.name").toLowerCase();
String message = "" + jct.message;
IoTMessage iotMessage = parseMessage("" + jct.message);
String topic = DEFAULT_TOPIC;
if ( jct.topic != null && jct.topic.trim().length()>0 ) {
    topic = jct.topic.trim();
}
ProducerBuilder<IoTMessage> producerBuilder = client.newProducer(JSONSchema.of(IoTMessage.class))
    .topic(topic)
    .producerName("jetson")
    .sendTimeout(5, TimeUnit.SECONDS);

Producer<IoTMessage> producer = producerBuilder.create();

MessageId msgID = producer.newMessage()
    .key(iotMessage.getUuid())
    .value(iotMessage)
    .send();
```

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

Ingesting IoT Data via Java Pulsar

```
private static IoTMessage parseMessage(String message) {  
    IoTMessage iotMessage = null;  
  
    try {  
        if ( message != null && message.trim().length() > 0 ) {  
            ObjectMapper mapper = new ObjectMapper();  
            iotMessage = mapper.readValue(message, IoTMessage.class);  
            mapper = null;  
        }  
    }  
    catch(Throwable t) {  
        t.printStackTrace();  
    }  
  
    if (iotMessage == null) {  
        iotMessage = new IoTMessage();  
    }  
    return iotMessage;  
}
```

MQTT from Python

```
pip3 install paho-mqtt
```

```
import paho.mqtt.client as mqtt
client = mqtt.Client("rpi4-iot")

row = { }
row['gasKO'] = str(readings)
json_string = json.dumps(row)
json_string = json_string.strip()
client.connect("pulsar-server.com", 1883, 180)
client.publish("persistent://public/default/mqtt-2", payload=json_string,
               qos=0, retain=True)
```



Using NVIDIA Jetson Devices With Pulsar

<https://dev.to/tspannhw/unboxing-the-most-amazing-edge-ai-device-part-1-of-3-nvidia-jetson-xavier-nx-595k>

<https://github.com/tspannhw/minifi-xaviernx/>

<https://github.com/tspannhw/minifi-jetson-nano>

<https://github.com/tspannhw/Flip-iot>

<https://www.datainmotion.dev/2020/10/flank-streaming-edgeai-on-new-nvidia.html>

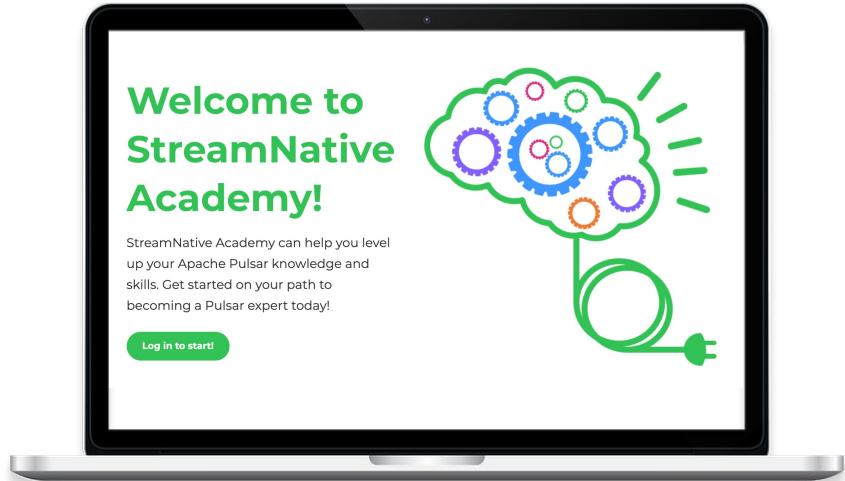
<https://github.com/tspannhw/FLiP-Mobile/blob/30bcc1ec98fc31e039b51a06180d98545c1e0542/python3/enviro.py>



Wrap-Up

Now Available On-Demand Pulsar Training

Academy.StreamNative.io



Platform Engineer [Remote]

San Francisco

Platform Engineer (Flink/Spark) [Remote]

San Francisco

Product Engineer - Cloud [Remote]

San Francisco

Platform Engineer (Flink/Spark) [Remote]

San Francisco

Product Engineer - Cloud [Remote]

San Francisco

Sr. Product Manager [Remote]

San Francisco

We're Hiring

streamnative.io/careers/



StreamNative

Connect with the Community & Stay Up-To-Date

- Join the Pulsar Slack channel - Apache-Pulsar.slack.com
- Follow [@streamnativeio](https://twitter.com/streamnativeio) and [@apache pulsar](https://twitter.com/apache_pulsar) on Twitter
- [Subscribe](#) to Monthly Pulsar Newsletter for major news, events, project updates, and resources in the Pulsar community

Interested In Learning More?



Resources

[Flink SQL Cookbook](#)

[The Github Source for Flink SQL Demo](#)

[The GitHub Source for Demo](#)



Free eBooks

[Manning's Apache Pulsar in Action](#)

[O'Reilly Book](#)



Upcoming Events

[11/8] [PASS Data Community](#)

[11/18] [Developer Week Austin](#)

[11/19] [Porto Tech Hub Con](#)

[12/3] [Data Science Camp](#)

Deeper Content

- <https://www.datainmotion.dev/2020/04/building-search-indexes-with-apache.html>
- <https://github.com/tspannhw/nifi-solr-example>
- <https://github.com/streamnative/pulsar-flink>
- <https://www.linkedin.com/pulse/2021-schedule-tim-spann/>
- https://github.com/tspannhw/SpeakerProfile/blob/main/2021/talks/20210729_HailHydrate!FromStreamtoLake_TimSpann.pdf
- <https://streamnative.io/en/blog/release/2021-04-20-flink-sql-on-streamnative-cloud>
- <https://docs.streamnative.io/cloud/stable/compute/flink-sql>



@PaasDev timothyspann

<https://www.pulsardeveloper.com/>

Let's Keep in Touch!



Tim Spann

Developer Advocate



@PassDev



<https://www.linkedin.com/in/timothyspann>



<https://github.com/tspannhw>

Questions