

Adopting Data in Motion: The Modern-Event Driven Architecture



Alex Stuart

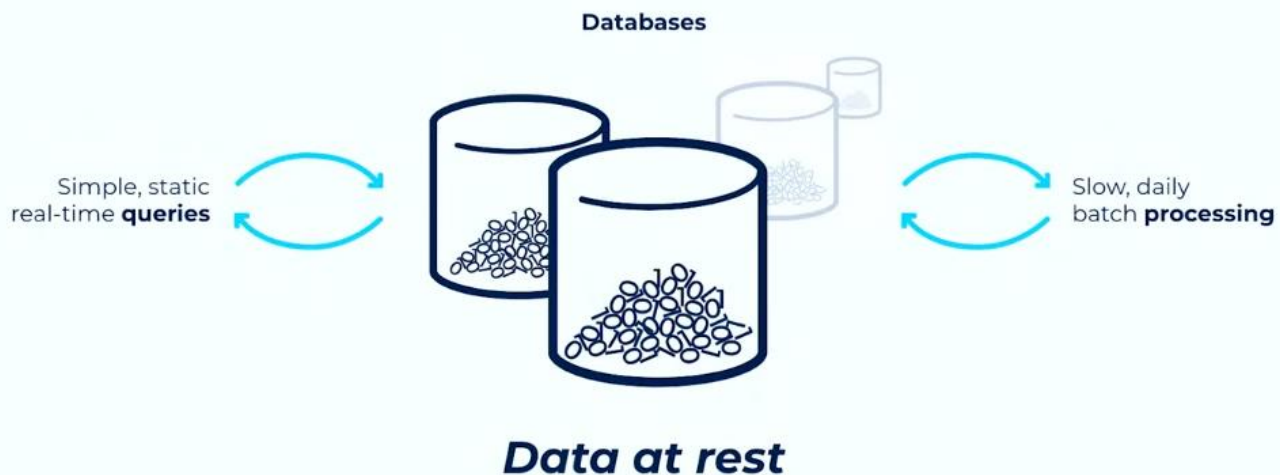
Advisory Solutions Engineer

astuart@confluent.io

[in](#) ajfstuart

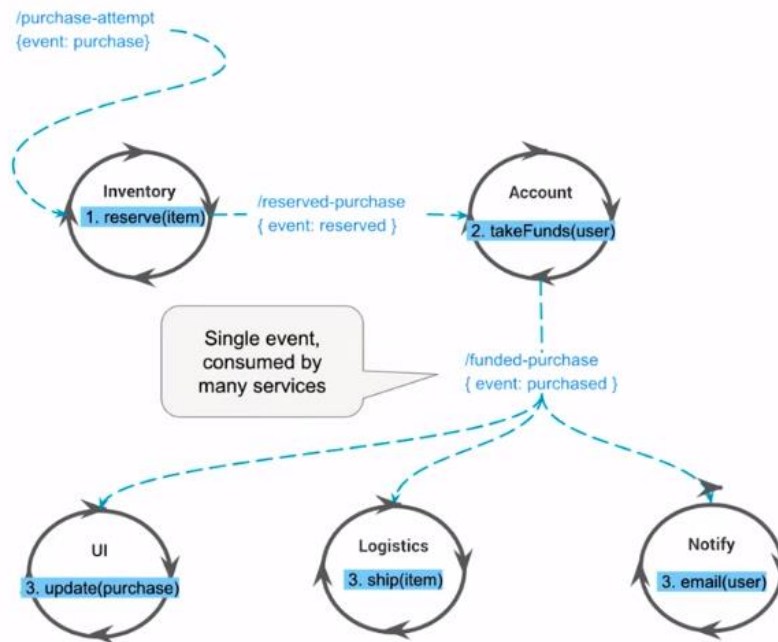
Far from a controversial choice, Apache Kafka® is technology developers and architects are enthusiastically adopting. And it's often not just a good choice but a technology enabling meaningful improvements in complex, evolvable systems that need to respond to the world in real-time. In this talk about event-driven architectures, Alex will start with our beloved database and the age-old way of handling data, then move on to the fundamental shift in the role of software and data - Data in Motion. Alex will share a comprehensive vision of an event-driven architecture suitable for the next generation of information technology deployments. You'll leave knowing where you need to go and how this new architectural paradigm will help you get there.

The Foundational Assumption of Every Database: Data at Rest

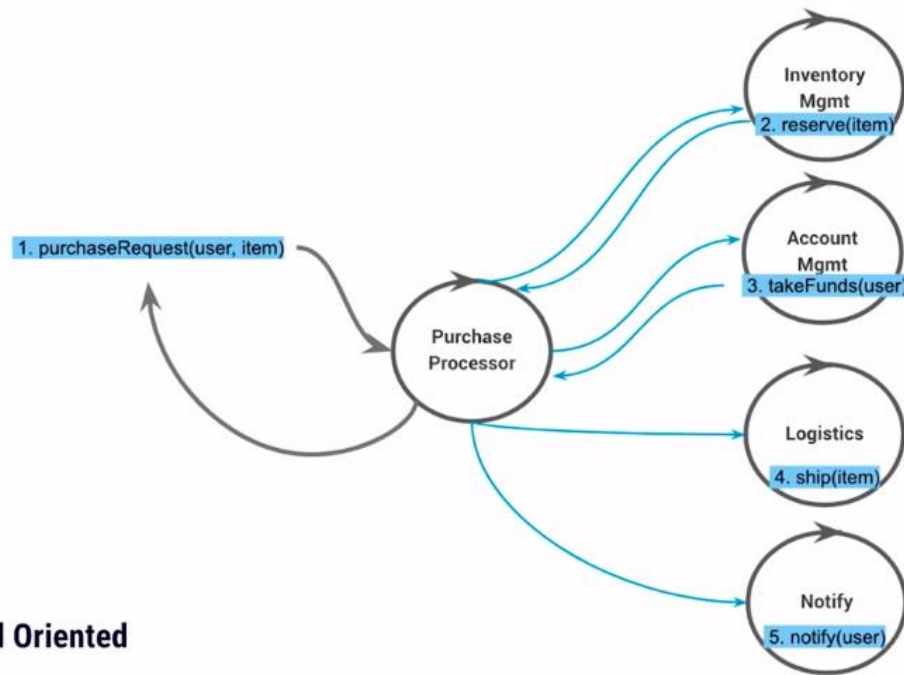


Data Is The New Engine of Business Success

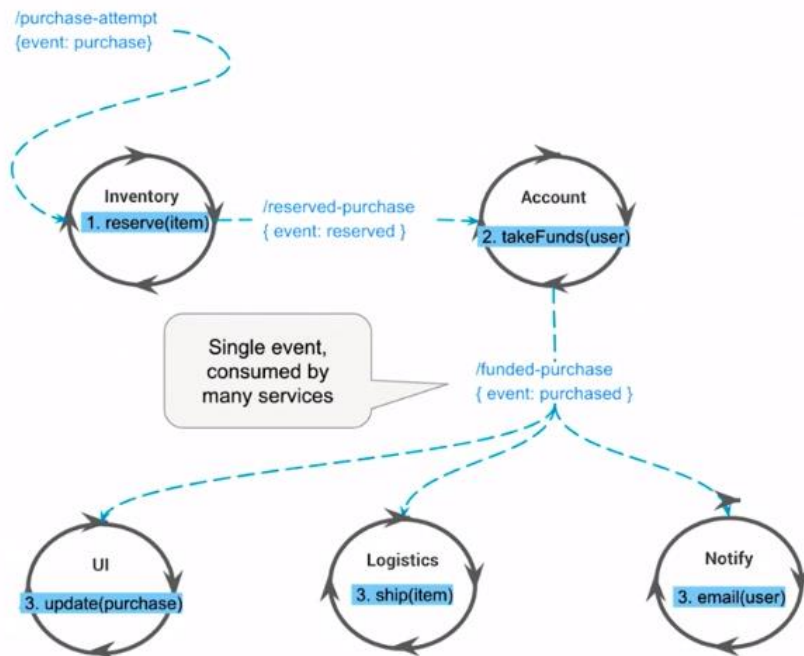
Paradigm for Data in Motion: Event Streams



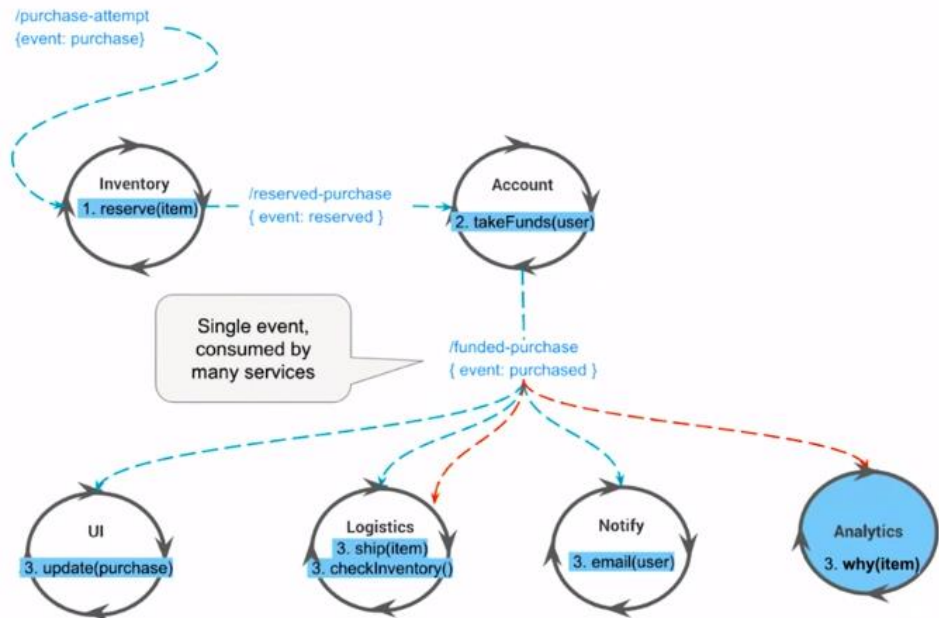
Basic Event Oriented



Event-Command Oriented



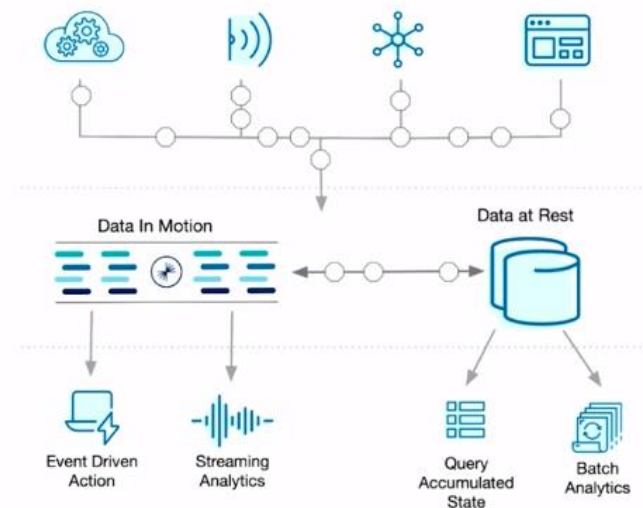
Event Oriented - Asynchronous



Event Oriented Decoupled

Modern Architecture

- Event Production
 - Everything is a source
 - Native Events
 - DB Transactions
- Events Flow Between
- Data in Motion
- Data at Rest



The Rise of Event Streaming

80%

Fortune 100
Companies
trust and use
Apache Kafka


created at LinkedIn by
Confluent founders

 CONFLUENT

2010

2014

2020

BLIZZARD

YAHOO!



stripe

intuit

Square

Tencent 腾讯

airbnb

GoPro

Wikipedia

Uber

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

Slack

Uber

Stripe

Intuit

Square

Tencent

Yahoo

Google

Blizzard

Wikipedia

Dropbox

PayPal

Netflix

Apple

Goldman Sachs

Twitter

LinkedIn

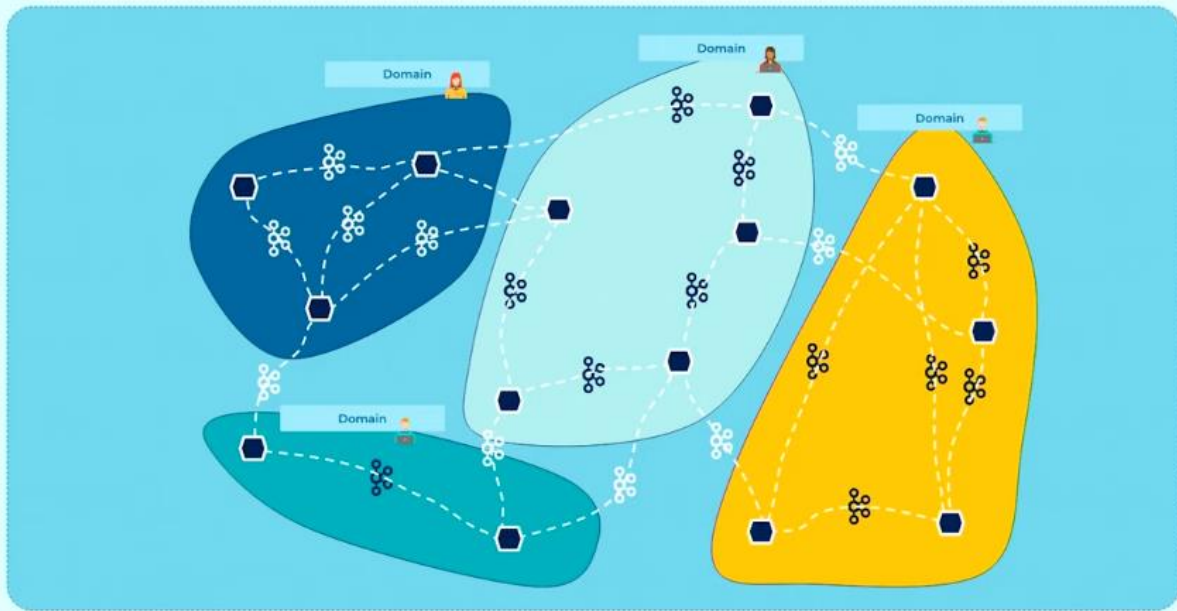
Slack

Uber

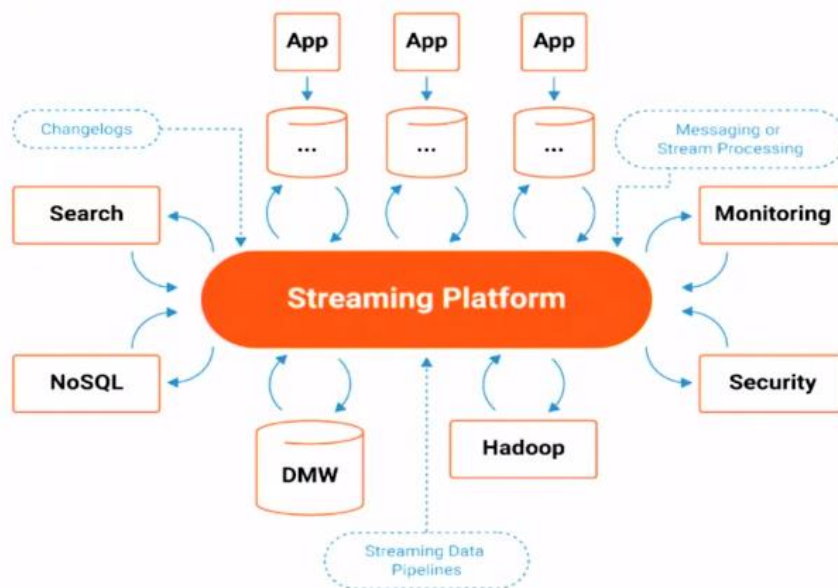
Stripe

Intuit

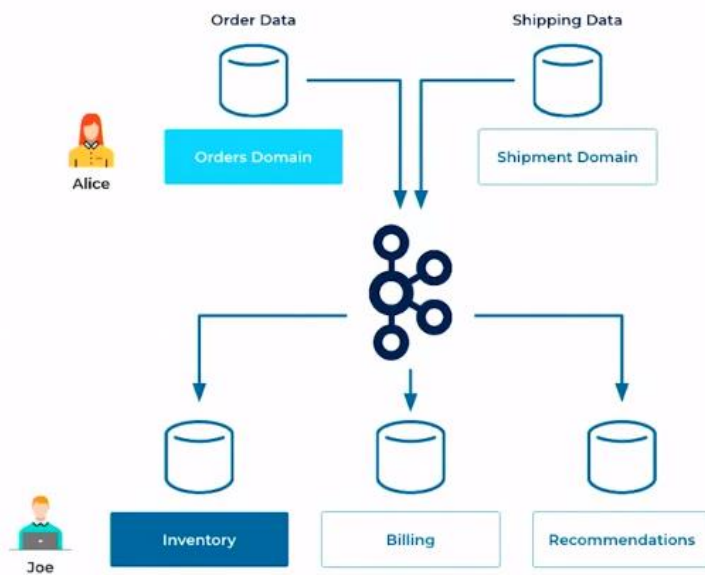
1. Treat Event Data as First Class Citizen




Data-Centric Approach to Computing

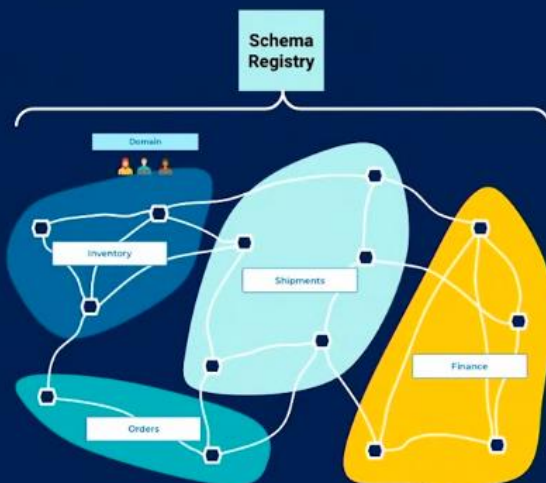


Practical example



- 
1. Joe in Inventory has a problem with Order data.
 2. Inventory items are going negative, because of bad Order data.
 3. He could fix the data up locally in the Inventory domain, and get on with his job.
 4. Or, better, he contacts Alice in Orders and get it fixed at the source. This is more reliable as Joe doesn't fully understand the Orders process.
 5. Ergo, Alice needs to be a responsible & responsive "Data Owner", so everyone benefits from the fix to Joe's problem.

2. Make Schemas the Contract for Event Streams



Confluent Schema Registry:

- Supports:
 - Avro
 - Protobuf
 - JSON Schema
- Can be used with Event Streams and other technologies

Schema Evolution - Compatible Changes

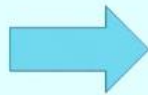


- Schemas will evolve!
- Think about backwards and forwards compatibility
- Compatible changes include (for example):
 - Adding new fields
 - Using default values (improves compatibility rules)
 - Removing fields with defaults

Schema Evolution



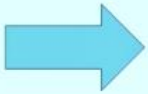
```
message User {  
  int32 id = 1;  
  string first_name = 2;  
  string last_name = 3;  
}
```



Adding fields

```
message User {  
  int32 id = 1;  
  string first_name = 2;  
  string last_name = 3;  
  string country_code = 4;  
}
```

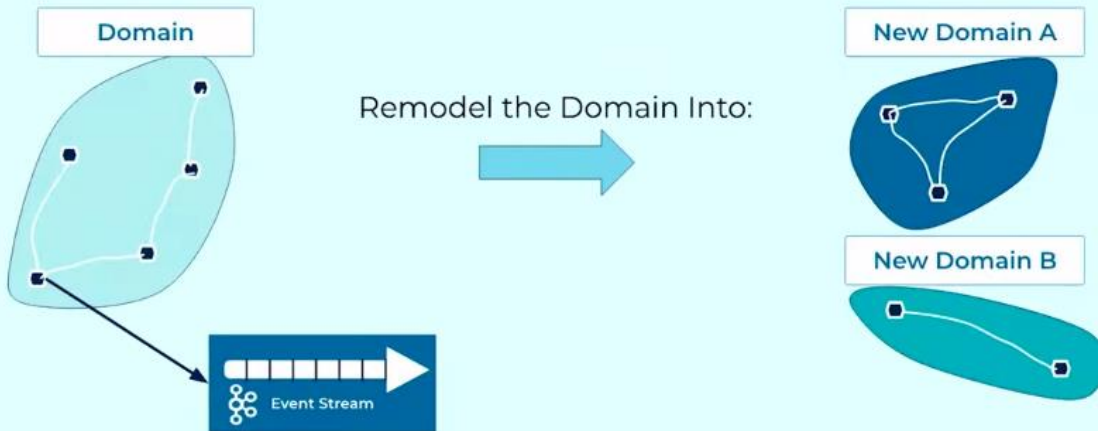
```
message User {  
  int32 id = 1;  
  string first_name = 2;  
  string last_name = 3;  
}
```



Removing fields
(eg: PII)

```
message User {  
  int32 id = 1;  
}
```


3. Plan for Breaking Changes and Failures



Collaborative Effort - Coordinate and Plan



Know your event-stream consumers

- Who is affected?
- What are their needs?

Communicate with them

- Let them know ahead of time of upcoming changes
- Discuss how to fulfil their needs, such as:
 - Migrating old data to new format
 - Minimizing downtime

CONFLUENT

Stream Catalog

LEARN

HOME > ENVIRONMENTS > EDA > GOTO_EDA > KSQLDB >

Cluster overview

Topics

Data integration

Stream lineage

ksqlDB

Schema registry

EDA_KSQLDB

Editor Flow Streams Tables Persistent queries Performance Settings CLI instructions

```
1 select * from CREDIT_PAYMENT_HISTORY_TBL EMIT CHANGES;
```

Add query properties

auto.offset.reset = Earliest

+Add another field

Run query

All available streams and tables

- APPROVED_APPLICATIONS
- CREDIT_APPLICATIONS
- CREDIT_PAYMENT_HISTORY
- CREDIT_PAYMENT_HISTORY_TBL
- CREDIT_UTILIZATION
- CREDIT_UTILIZATION_REKEYED
- CREDIT_UTILIZATION_TBL
- KSQL_PROCESSING_LOG
- REJECTED_APPLICATIONS

New to stream processing and ksqlDB? Check out our [documentation](#) and [ksqlDB examples](#) <

EDA

Clusters Network management **Schema Registry** Environment settings

Schemas

Subject name	Schema type
credit_applications-value	JSON Schema
pksglc-y9mzpAPPROVED_APPLICATIONS-value	Avro
pksglc-y9mzpREJECTED_APPLICATIONS-value	Avro

View & manage schemas

Tags

Name	Description
Private	Private tag description
Public	Public tag description
Sensitive	Sensitive tag description
PII	Personally identifiable information

View & manage tags

Business metadata

Create and use business metadata to classify, organize, and improve discovery of entities in your environment.

View & manage business metadata



Schemas 9 schemas used

Search by schema subject name

+ Add schema

Subject name	Schema type	Current version
credit_applications-value	JSON Schema	1
pksglc-y9mzpAPPROVED_APPLICATIONS-value	Avro	2
pksglc-y9mzpREJECTED_APPLICATIONS-value	Avro	2

credit_applications-value

Type: JSON Schema Compatibility mode: Backward Used by topic: credit_applications

Version 1 [current] Schema ID: 100003

Compare versions Evolve schema ...

Search by keyword

- object (1) ⓘ
- type: object
 - additionalProperties: false
 - id: http://example.com/myURI.schema.json
 - schema: http://json-schema.org/draft-07/schema#
 - title: SampleRecord
 - description: Sample schema to help you get started.
 - properties (3)

Tags

Add tags to this version

Add business metadata

Schema doc

Sample schema to help you get started.

Date created

Aug. 31 2022 5:55 PM

CONFLUENT

Stream Catalog

LEARN

HOME > ENVIRONMENTS > EDA > SCHEMA REGISTRY > SCHEMAS >

credit_applications-value

Type: JSON Schema Compatibility mode: Backward Used by topic: credit_applications

Version 1 (current)

Schema ID: 100003

Compare versions

Evolve schema

Search by keyword

object

- type: object
- additionalProperties: false
- id: http://example.com/myURI:schema.json
- schema: http://json-schema.org/draft-07/schema#
- title: SampleRecord
- description: Sample schema to help you get started.
- properties
 - amount
 - Sensitive
 - credit_type
 - customer_id
 - PI
 - Private
 - type: integer
 - description: The integer type is used for integral numbers.

Tags

Add tags to this version

Add business metadata

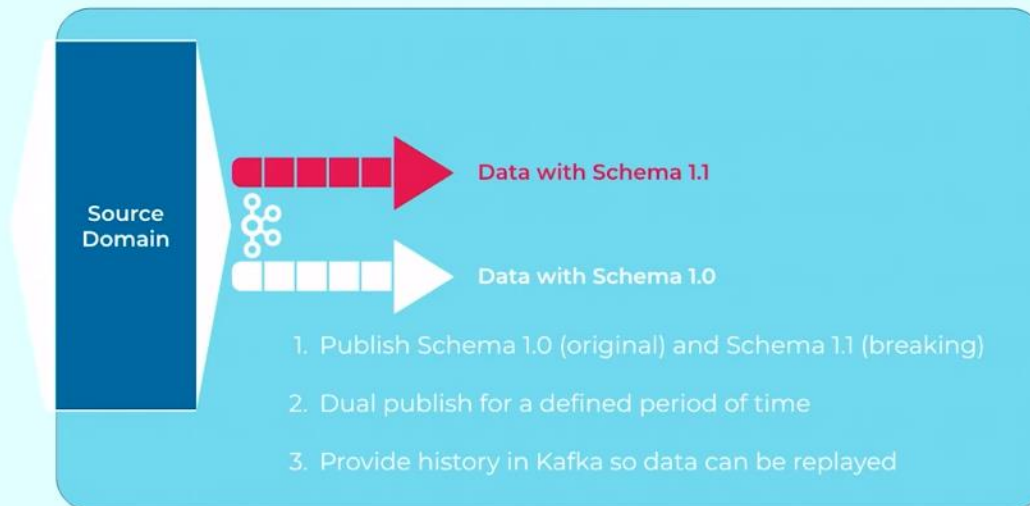
Schema doc

Sample schema to help you get started.

Date created

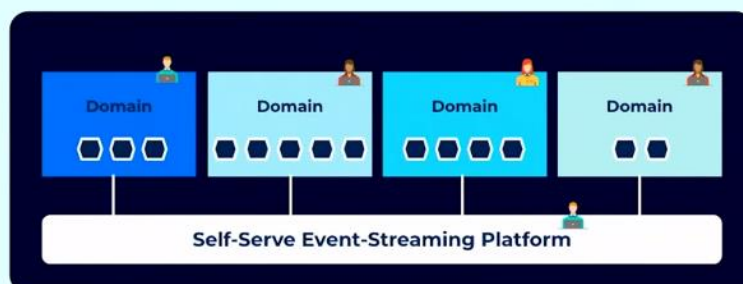
Aug. 31 2022 5:55 PM

Incompatible changes require a rolling upgrade window



4. Create and Empower a Governing Body

- Standards of Interoperability, Policies, and Support
- The minimum requirements for data in the org? (SLAs on changes, uptime, data, etc.)
- How schema changes are proposed, approved and communicated



Decisions can Include:



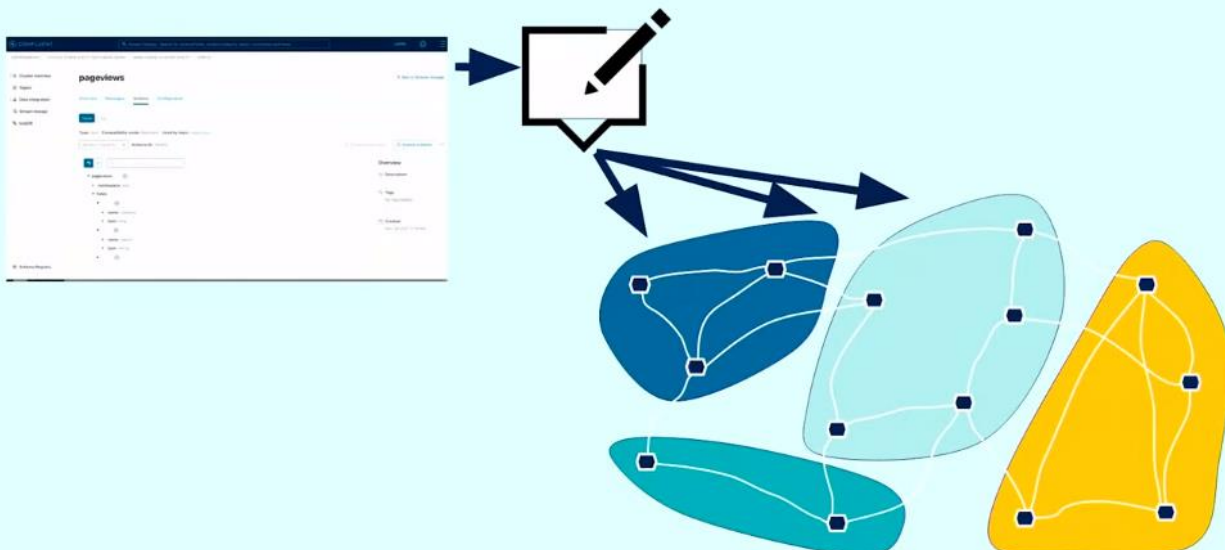
- What schema format do we use?
- How do we monitor our applications?
- How do we impose Access Controls?
- How do we adhere to our data policies?
- Etc: What else are the main data concerns of your business?

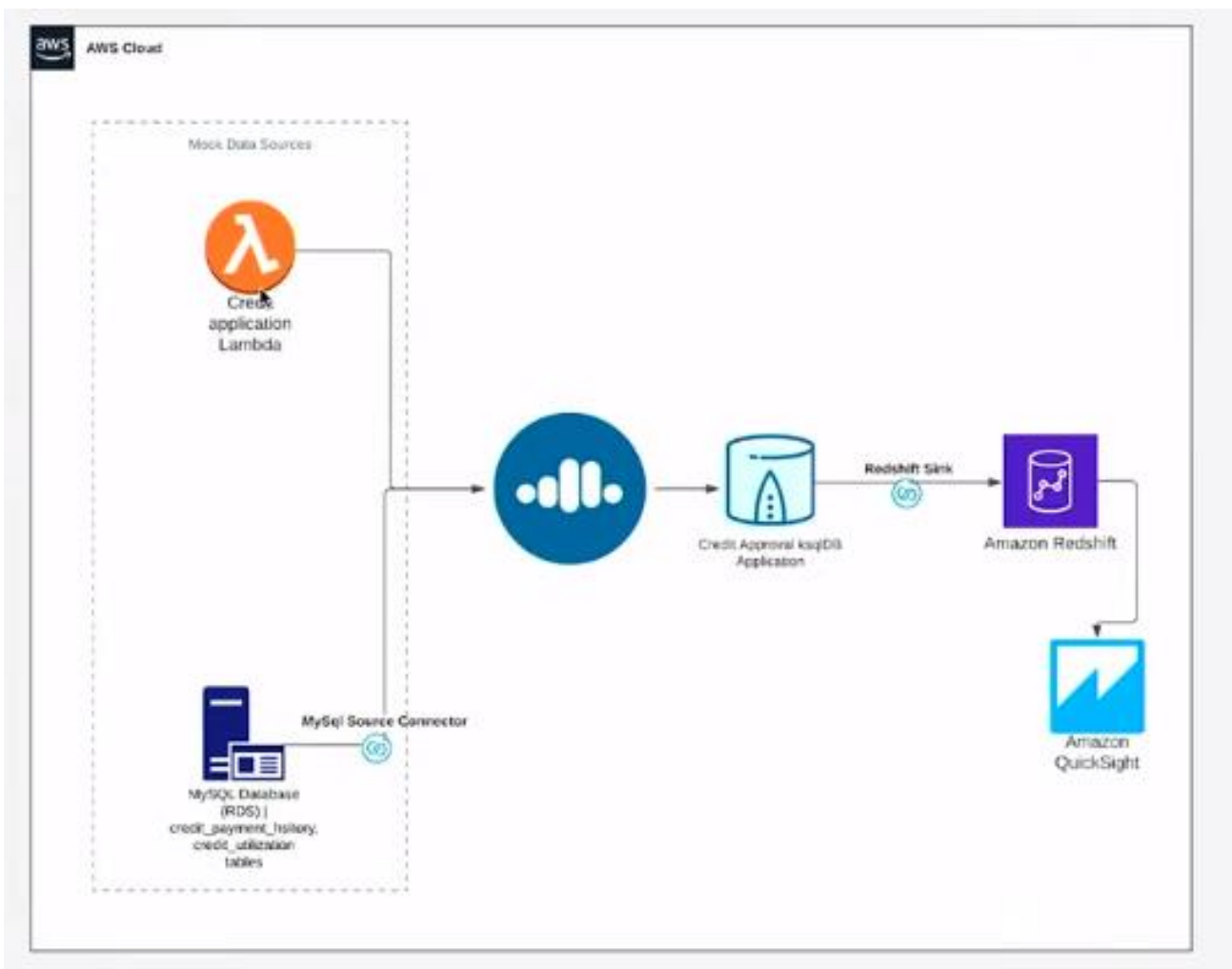
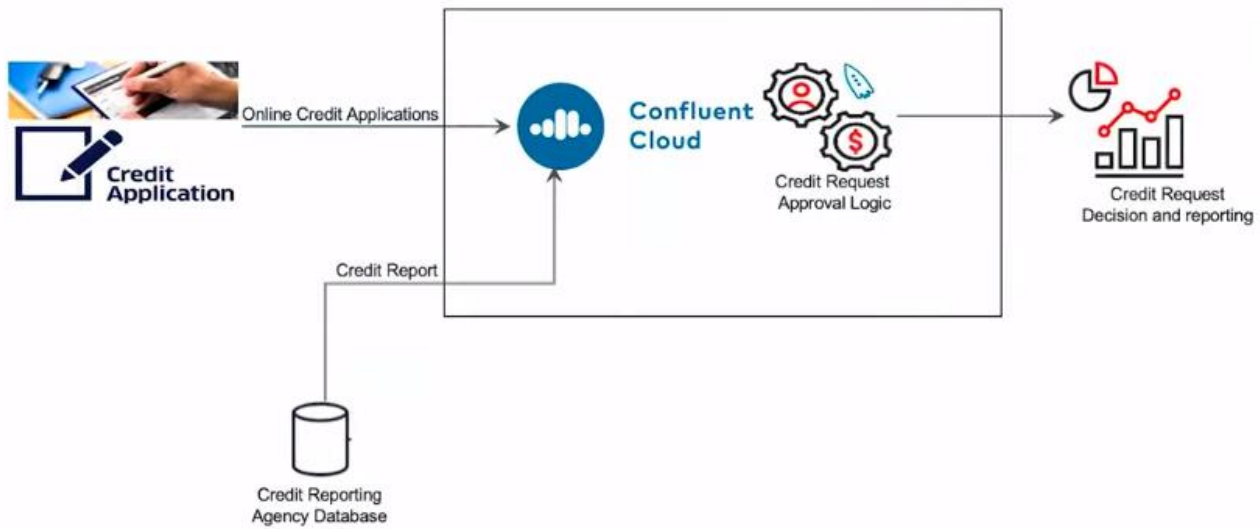
5. Provide Self-Service Functionality with First Class Support

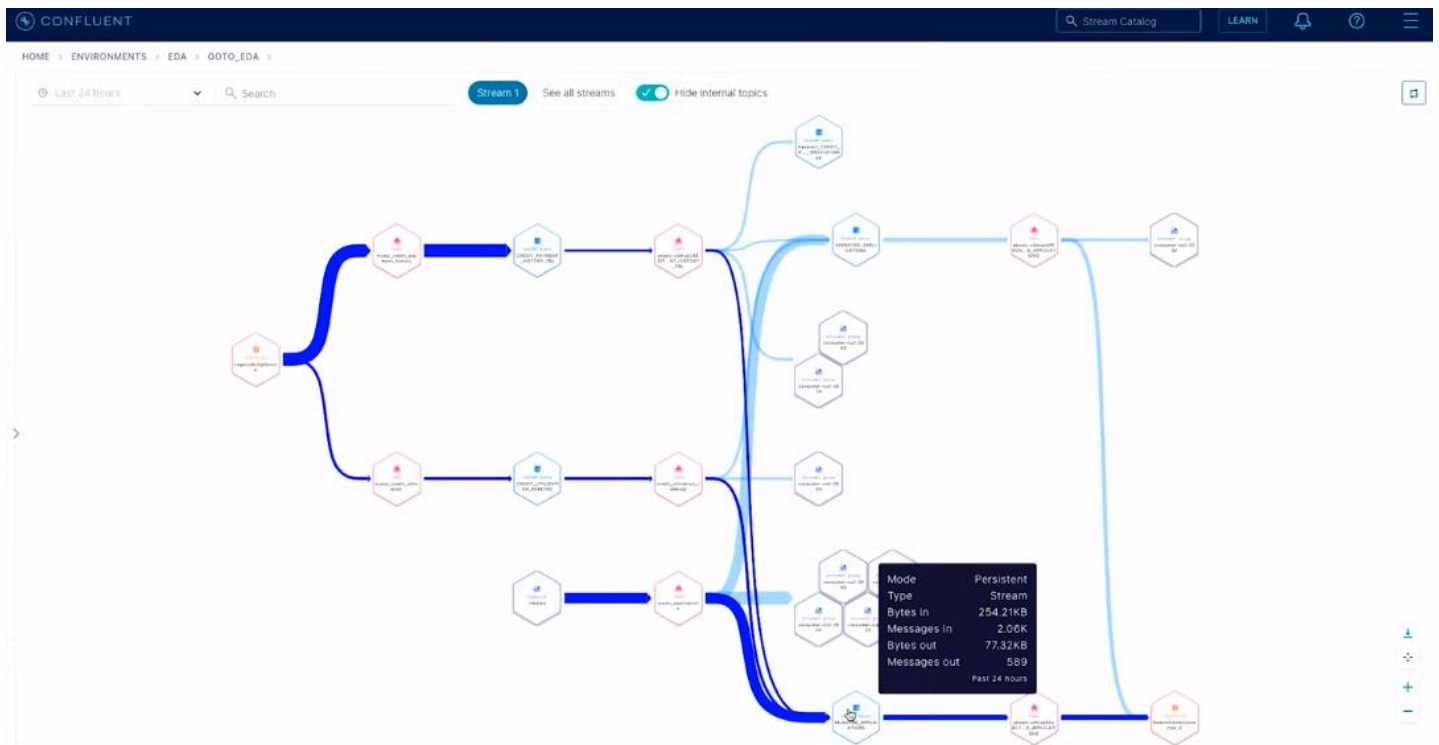
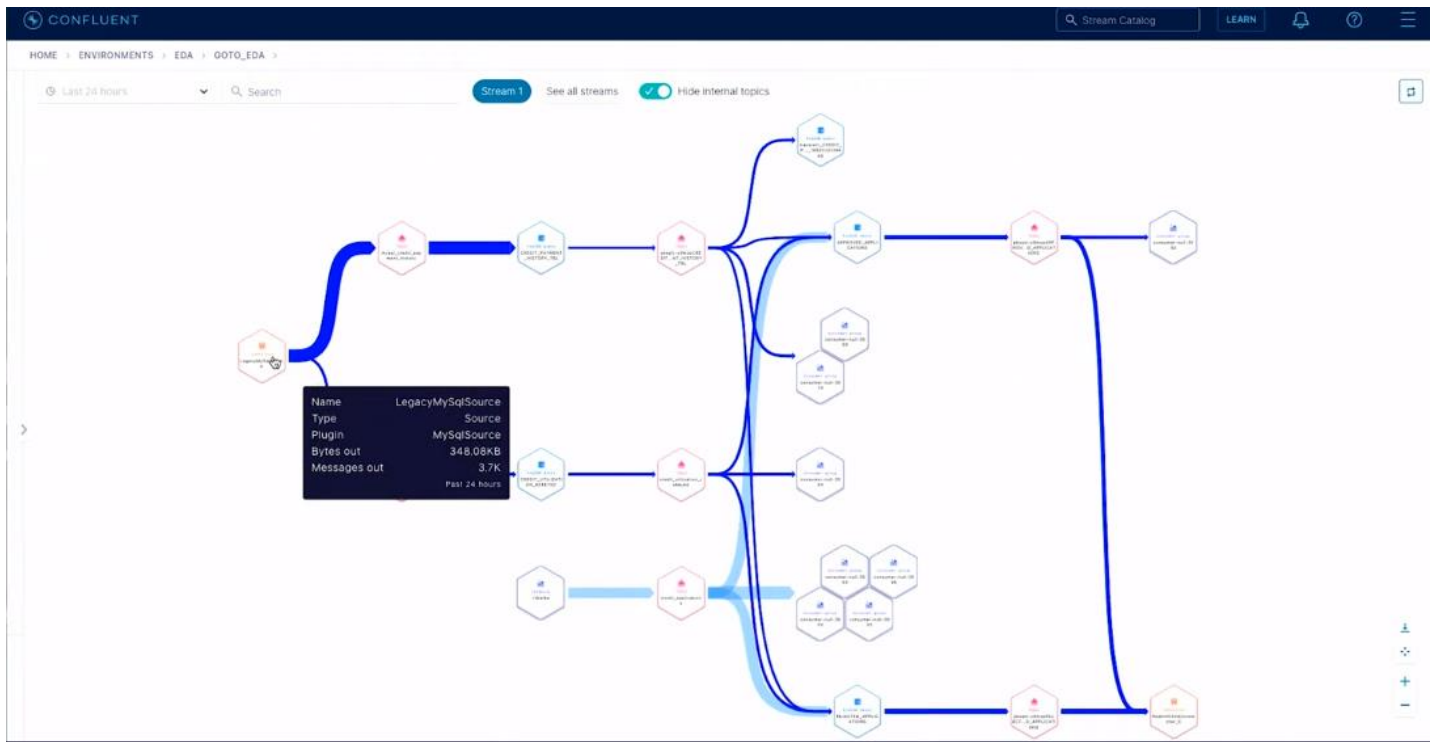


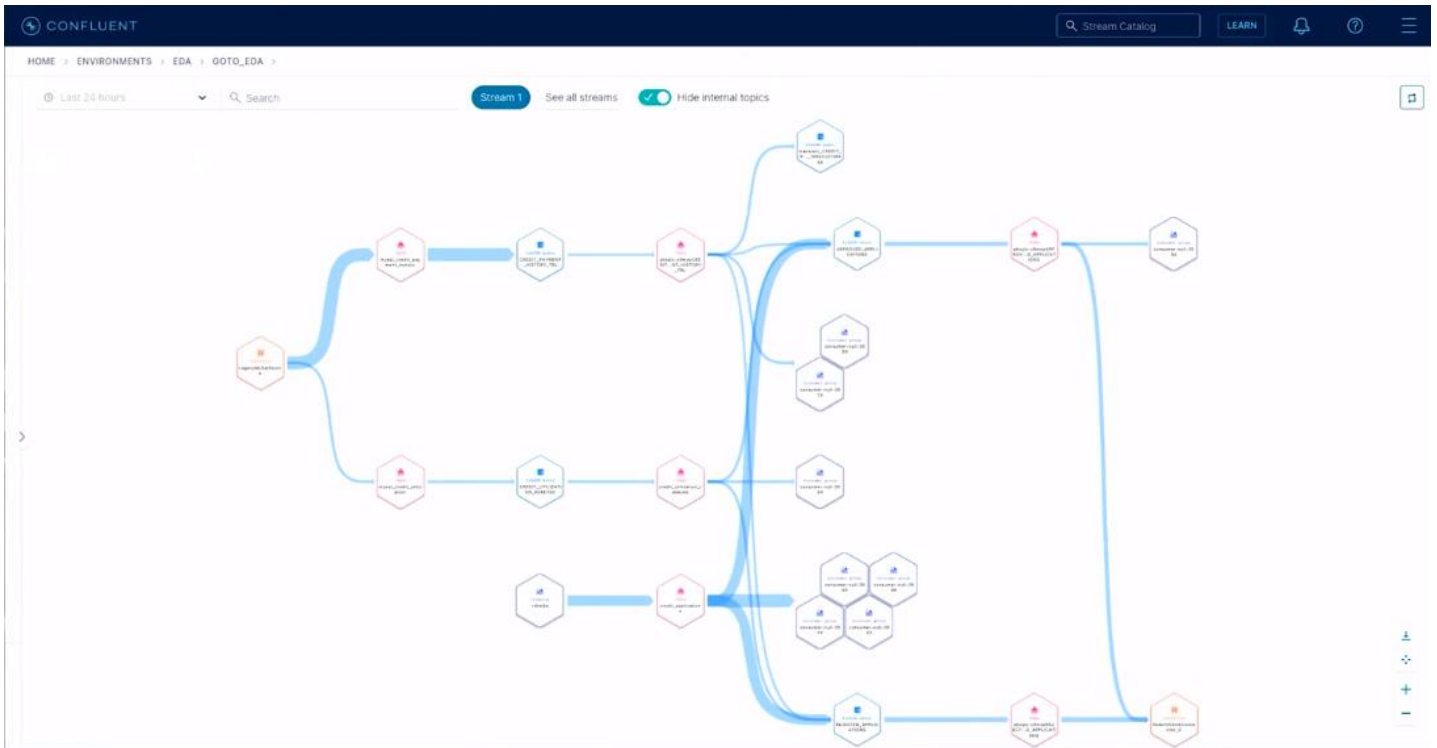
- **Global standards and streamlined support**
 - Eg: New applications must be in containers
 - Eg: New event-driven applications must use Confluent Cloud
- **“Paved Roads” - Make it Easy to Use**
 - Make it easy to build applications using event streams, in the languages and frameworks of your choice.
 - Proof-of-concept applications and pilot projects can lead the way
 - Focus on making it easy to get things done: reduce toil and overhead.

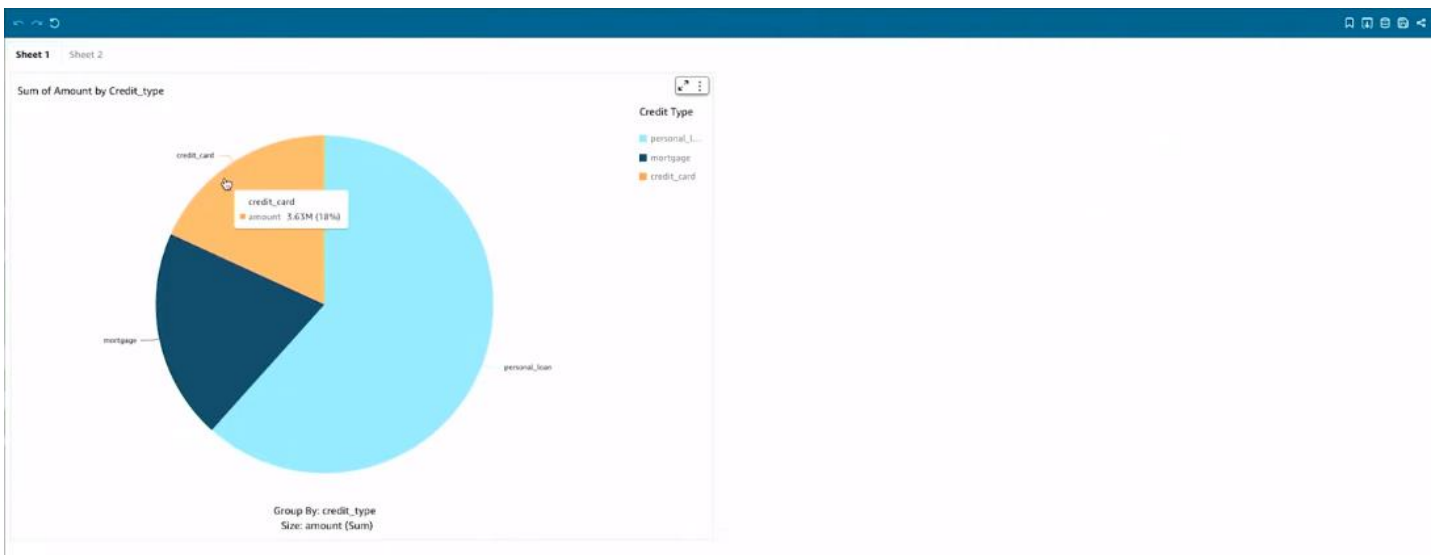
6. Make Data Easy to Discover and Use



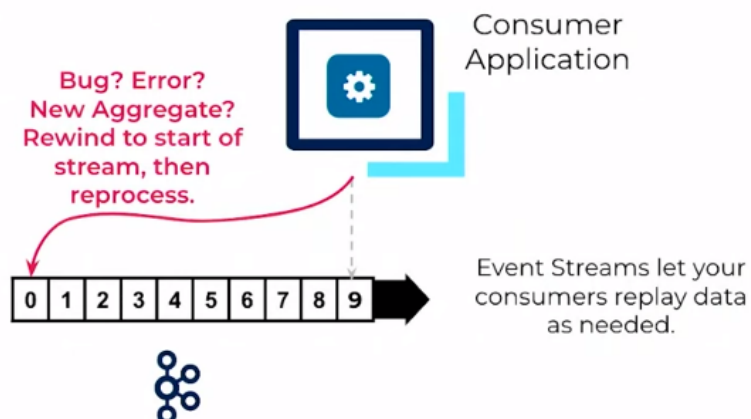








Rely on Event Streams for Historical Data



Store Event Data For As Long As Necessary



- Store all the data you need, for as long as you need it.
- Cheap disk! Compaction!
- Confluent Cloud's Infinite Storage
 - Fully transparent tiered-storage

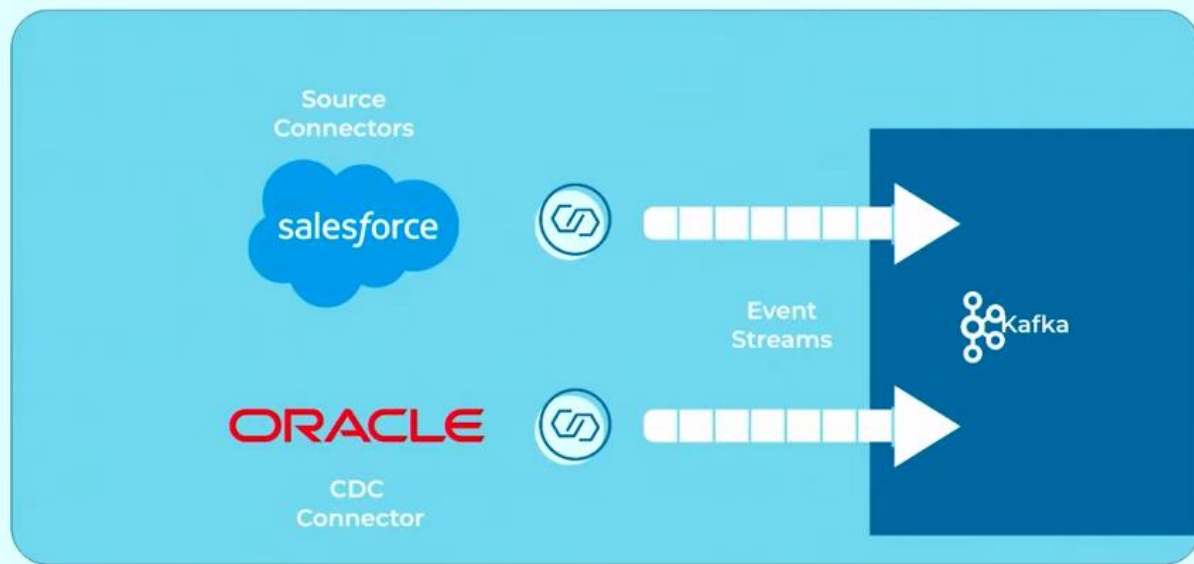
Self-Service ksqlDB: Process and Transform Event Streams



Self-Service ksqlDB: Query data however you need it



7. Rely on Connectors to Bridge Streams and Batch

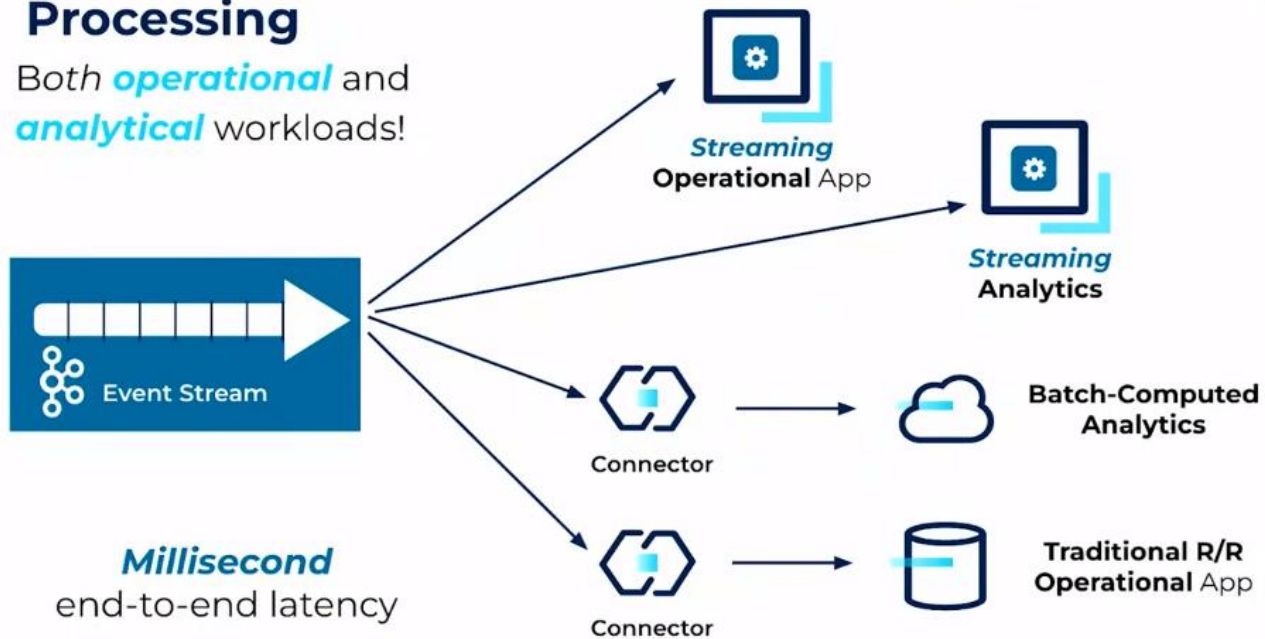


<https://www.confluent.io/hub/>

Event Streams Power Realtime & Batch Processing



Both **operational** and **analytical** workloads!



Principles for Building Event-Driven Architectures



1. Treat Data as a First-Class Citizen

So that it can be relied upon and reused by all users. Dedicated owners for all key datasets in the organization.

2. Make Schemas the Contract for Event Streams

Schema evolution. Confluent Schema Registry.

3. Plan for Breaking Changes and Failures

Some changes require remodeling existing streams. Plan for breakages.

4. Create and Empower a Governing Body

Standard. How to validate, publish, change, find, and use event streams.

5. Provide Self-Service Functionality

Compute and processing frameworks. Connectors. Create, build, and manage your own the data you publish for others.

6. Make Data Easy to Discover and Use

Search for schemas and event data. Preview event streams. Request access to data. Data lineage views.

7. Rely on Connectors to Bridge Streams and Batch

Connectors for sourcing and sinking data. Data is essential for getting EDAs started.

Thanks!
Check out *developer.confluent.io*