

# The Changing Face of ETL

## Event-Driven Architectures for Data Engineers

#OOPmuc @rmoff

**oop**  
software meets business





# The Changing Face of ETL

## Event-Driven Architectures for Data Engineers

#OOPmuc @rmoff

**oop**  
software meets business





























It used to be so simple





# More Sources



A high-angle, nighttime photograph of a complex multi-level highway interchange. The concrete structures of the overpasses and ramps are illuminated by streetlights, creating a series of overlapping curves and lines. Several cars with their headlights on are visible traveling along the various levels of the highway. In the background, the silhouettes of city buildings and a tall, white industrial tower are visible against the dark sky. The overall scene conveys a sense of constant movement and urban infrastructure.

# More Targets



The background is an abstract image featuring a dense, repeating pattern of hexagons. The pattern is composed of light gray hexagons with dark gray outlines, creating a textured, scale-like appearance. The pattern is slightly distorted, giving it a three-dimensional, undulating effect. The text "More Data" is centered over this pattern in a white, sans-serif font, enclosed within a semi-transparent gray rectangular box.

More Data





# Batches and Buckets





Applications  
**Respond**

→ an order was placed!

Analytics

**Tell Us What  
Happened**

→ how many orders  
were placed













# Events



An event is both:

- ★ Notification

- ★ State transfer



# A Customer Experience





# A Sensor Reading





# Databases





# The Stream/Table Duality

@rmoff #OOPmuc

## Stream

Time



Account ID	Amount
12345	+ €50
12345	+ €25
12345	-€60

## Table

Account ID	Balance
12345	€50

Account ID	Balance
12345	€75

Account ID	Balance
12345	€15



A night photograph of a campsite. In the foreground, a small, glowing yellow tent is pitched on a grassy area. Behind it, a large, dark evergreen tree stands prominently. The background is a dark, starry night sky with the Milky Way visible. The overall scene is serene and quiet.

**The truth is the log.**

**The database is a cache  
of a subset of the log.**

**—Pat Helland**

**Immutability Changes Everything**

[http://cidrdb.org/cidr2015/Papers/CIDR15\\_Paper16.pdf](http://cidrdb.org/cidr2015/Papers/CIDR15_Paper16.pdf)



# Events

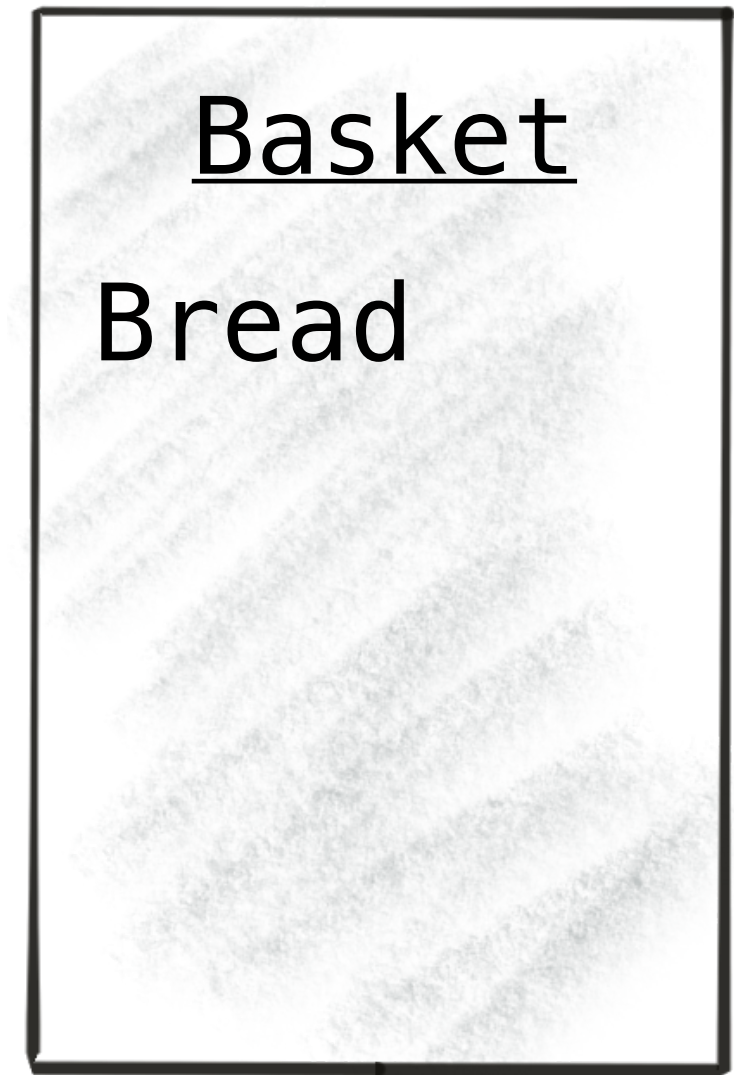
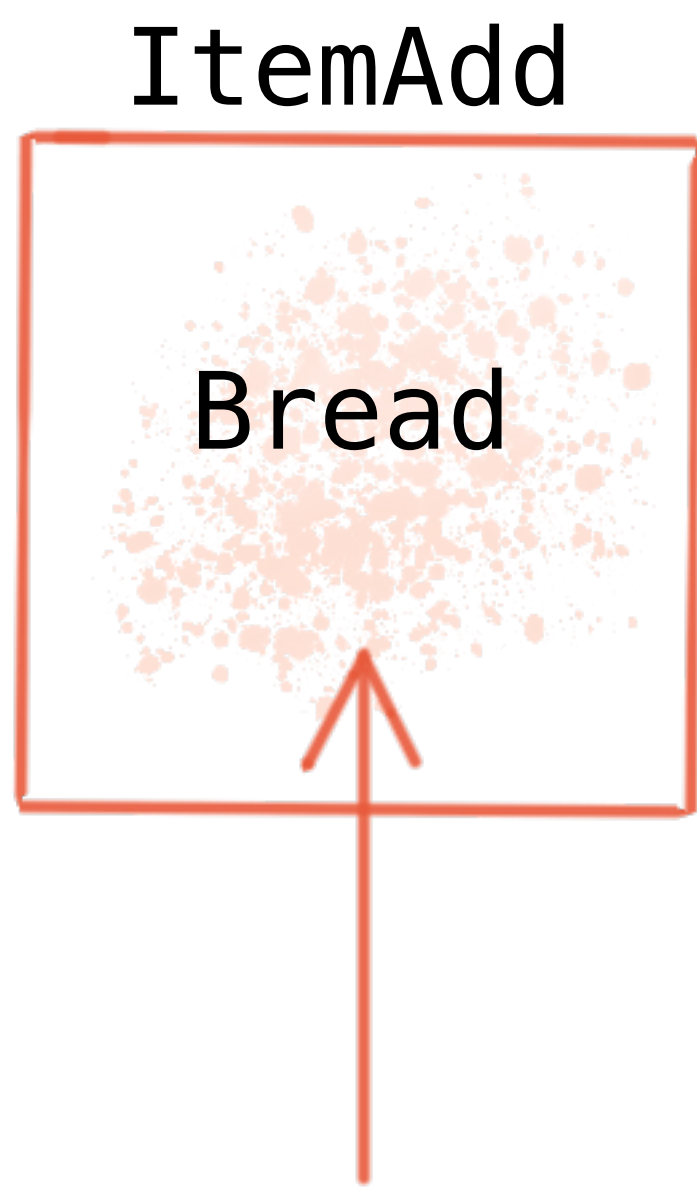
Basket

Bread

Tinned  
Spaghetti

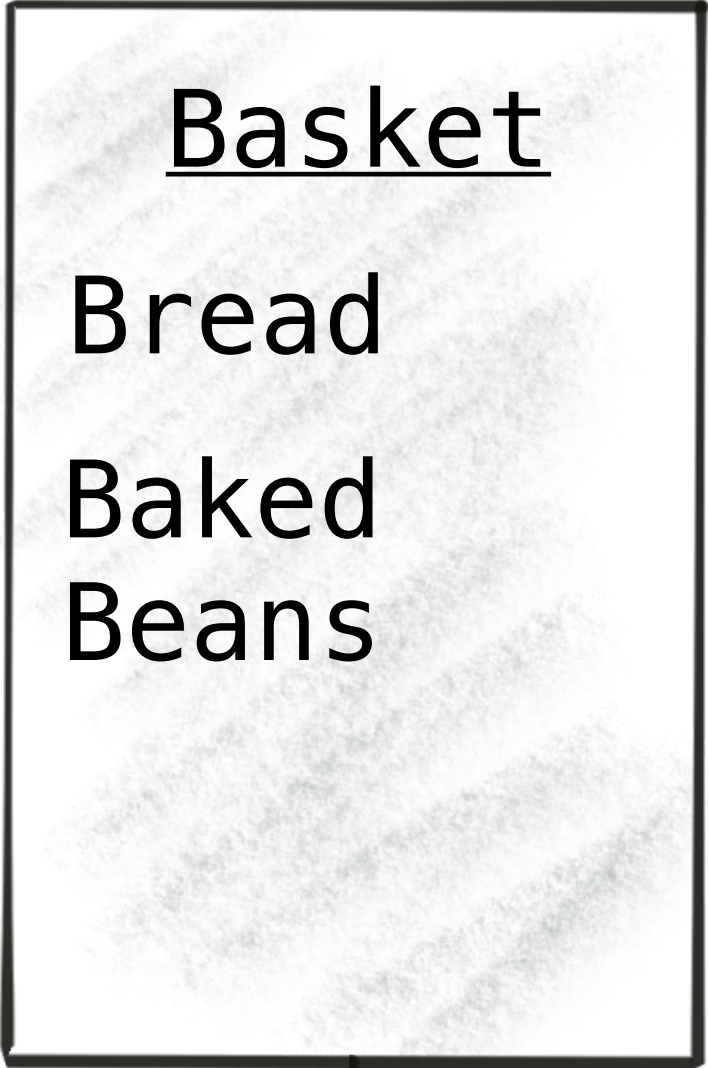
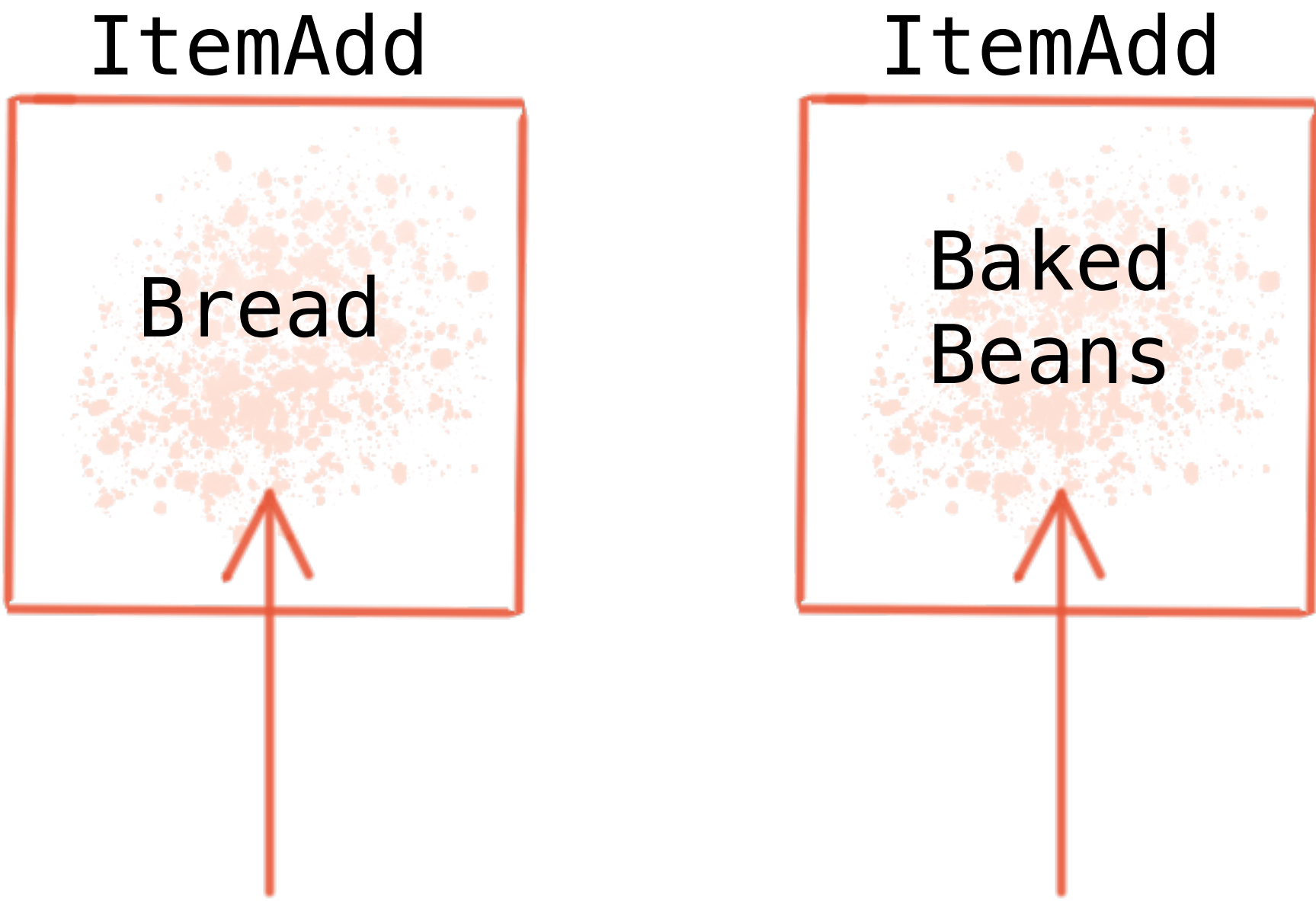


# Events



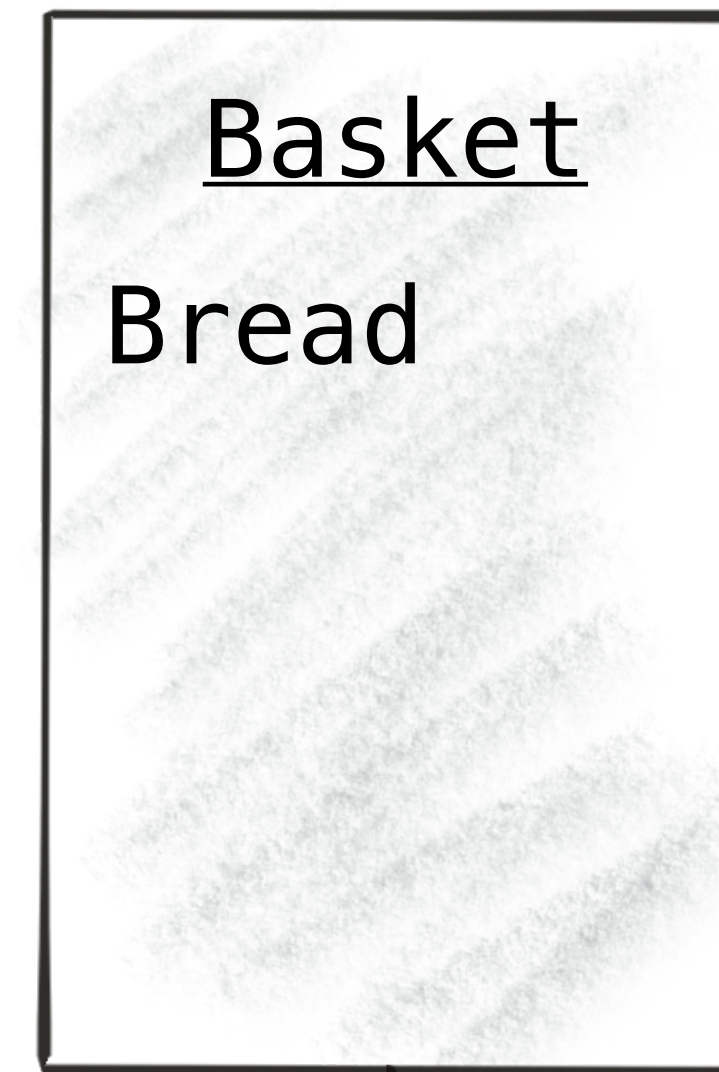
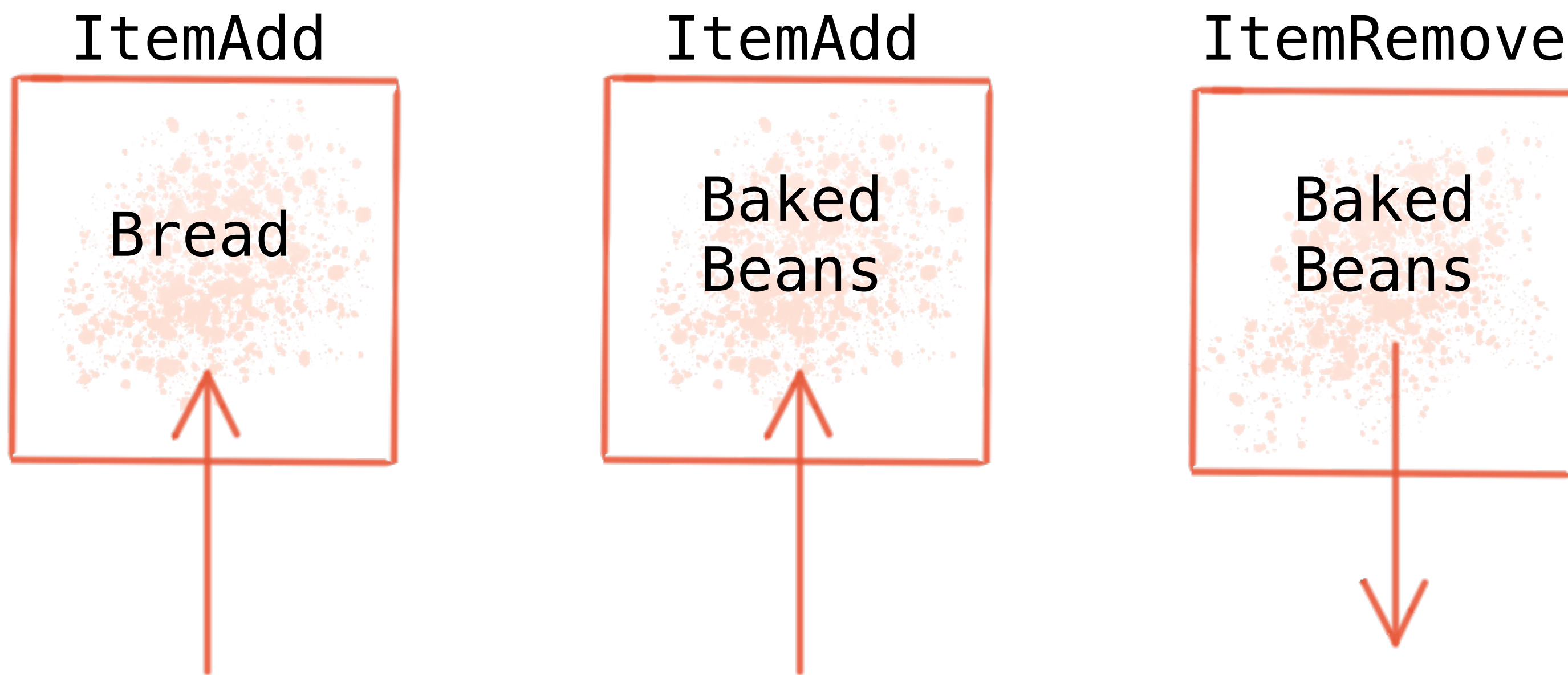


# Events



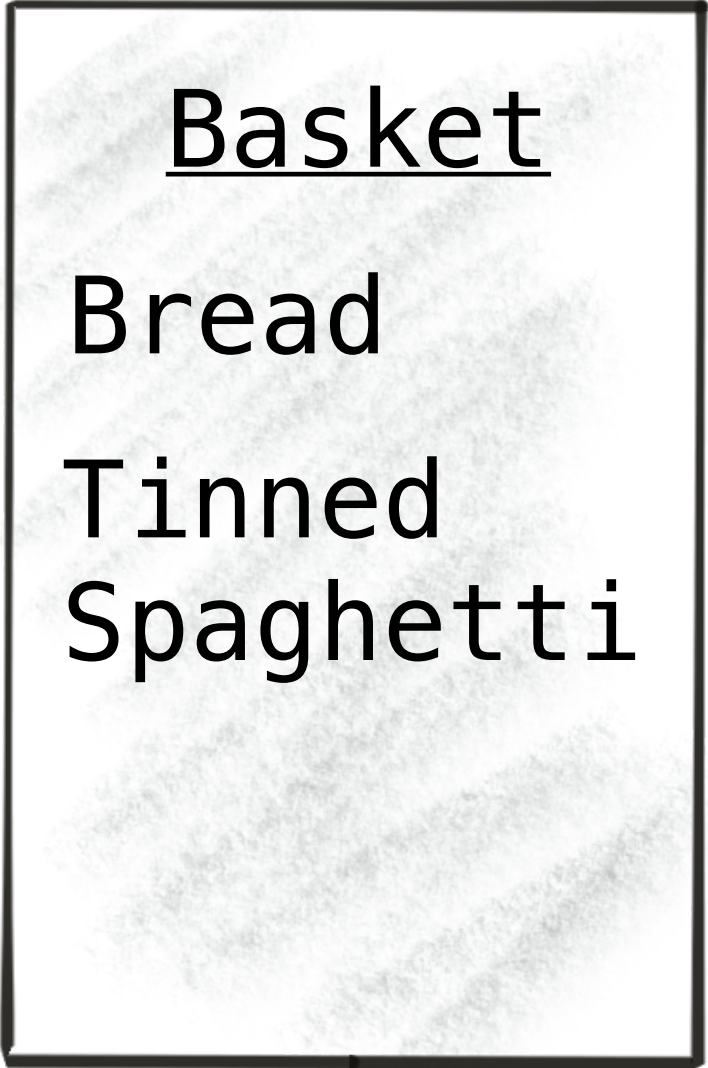
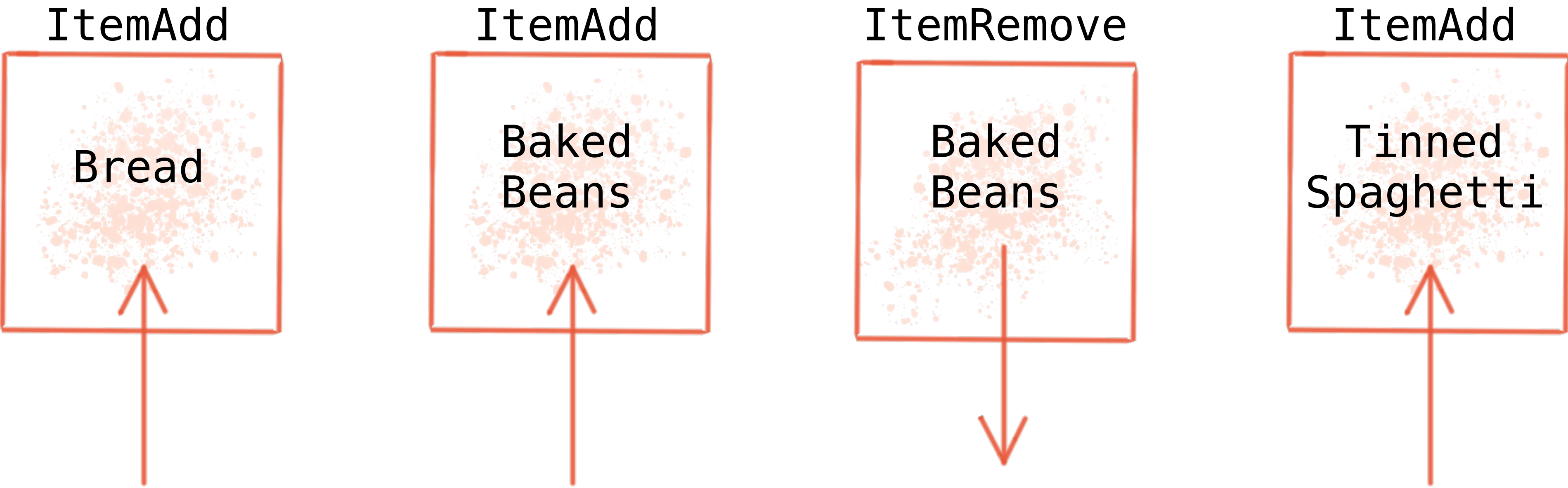


# Events



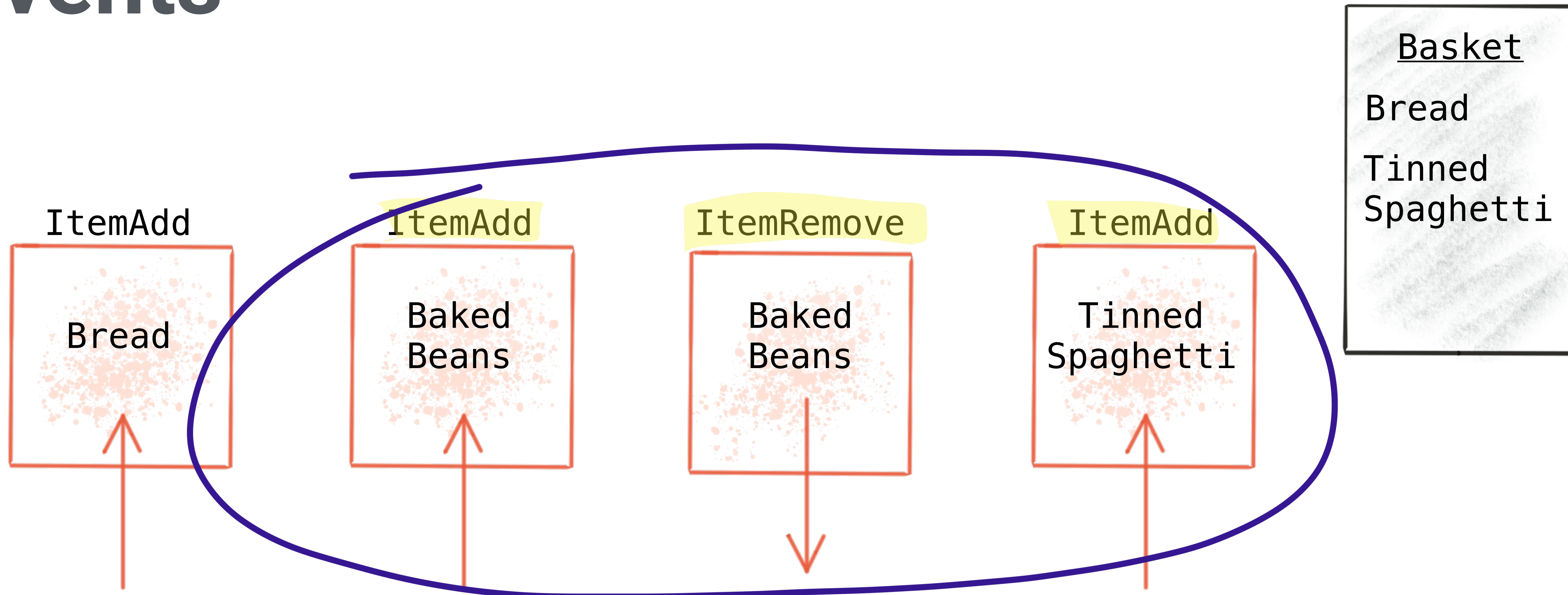


# Events





# Events

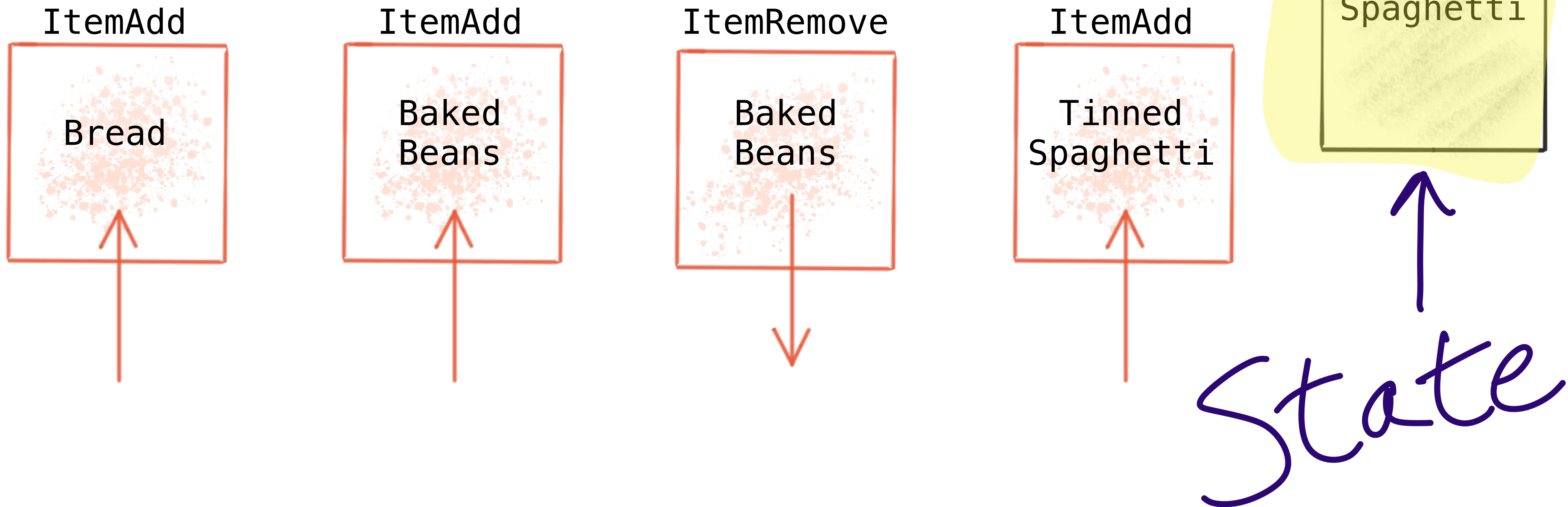


Behaviour



# Events

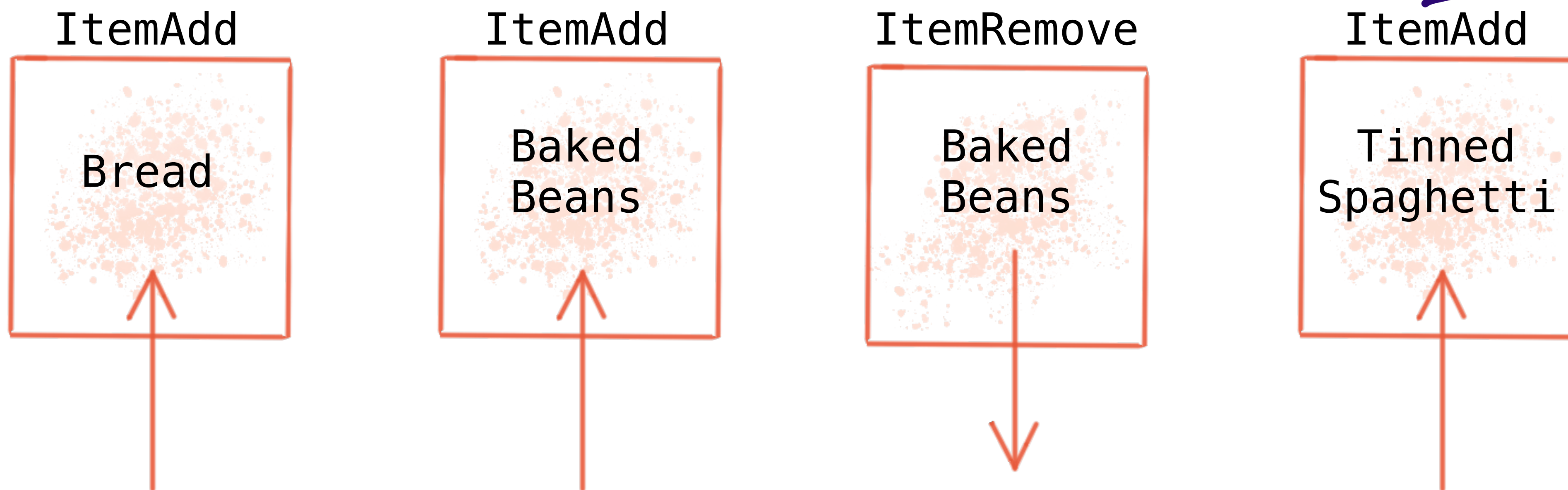
@rmoff #OOPmuc





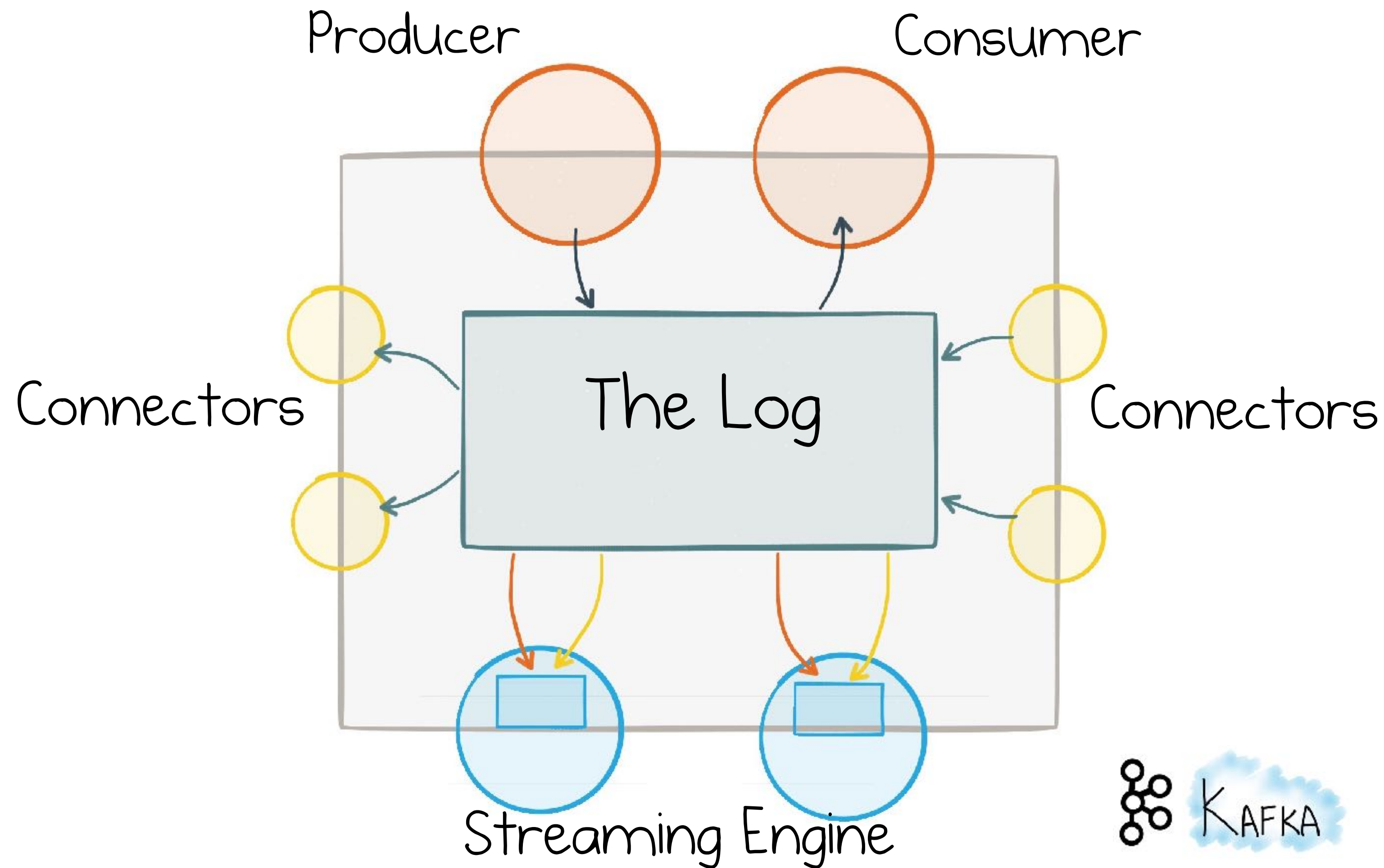
# Events

Event Stream



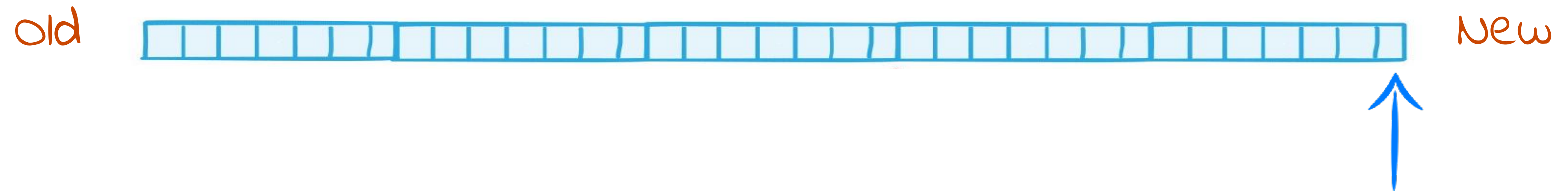


# What is an Event Streaming Platform?





# Immutable Event Log



Messages are added at the end of the log



# Topics

Clicks



orders



Customers

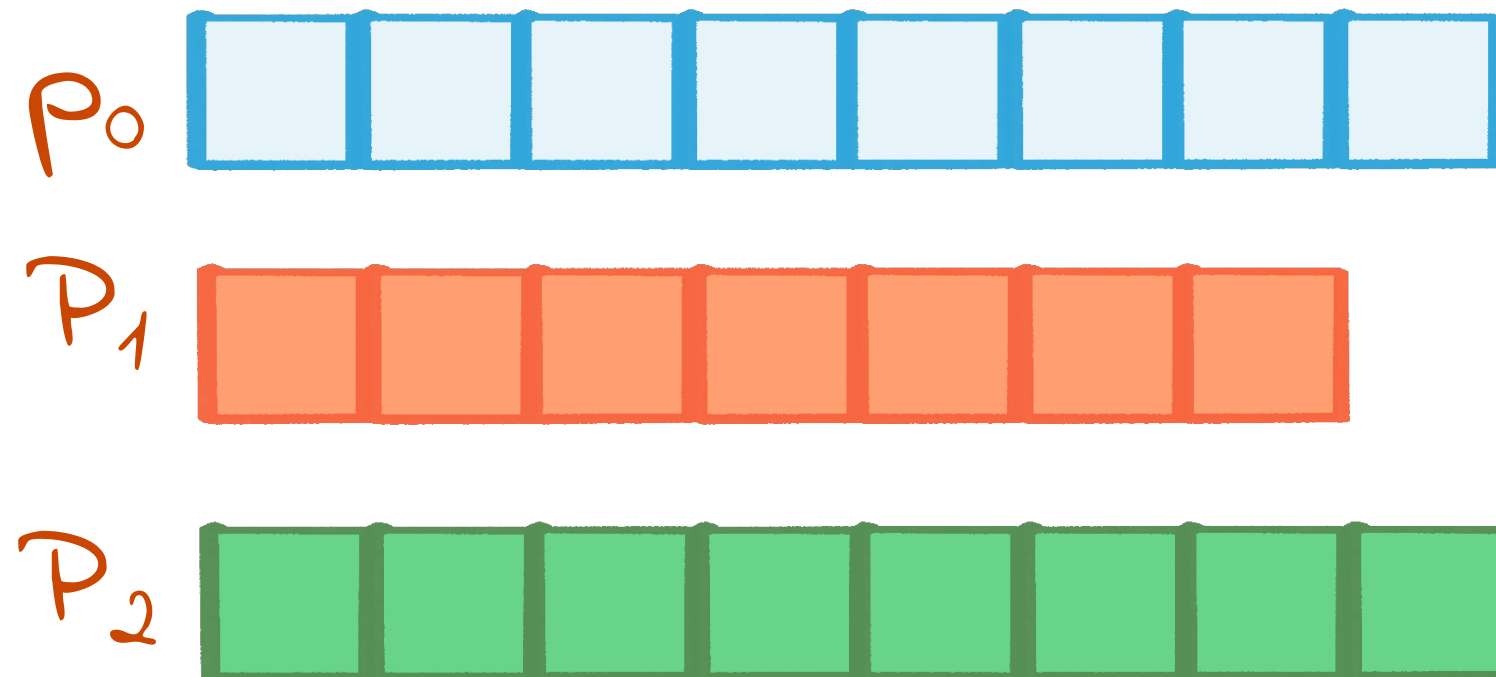


Topics are similar in concept to  
tables in a database



# Partitions

Clicks



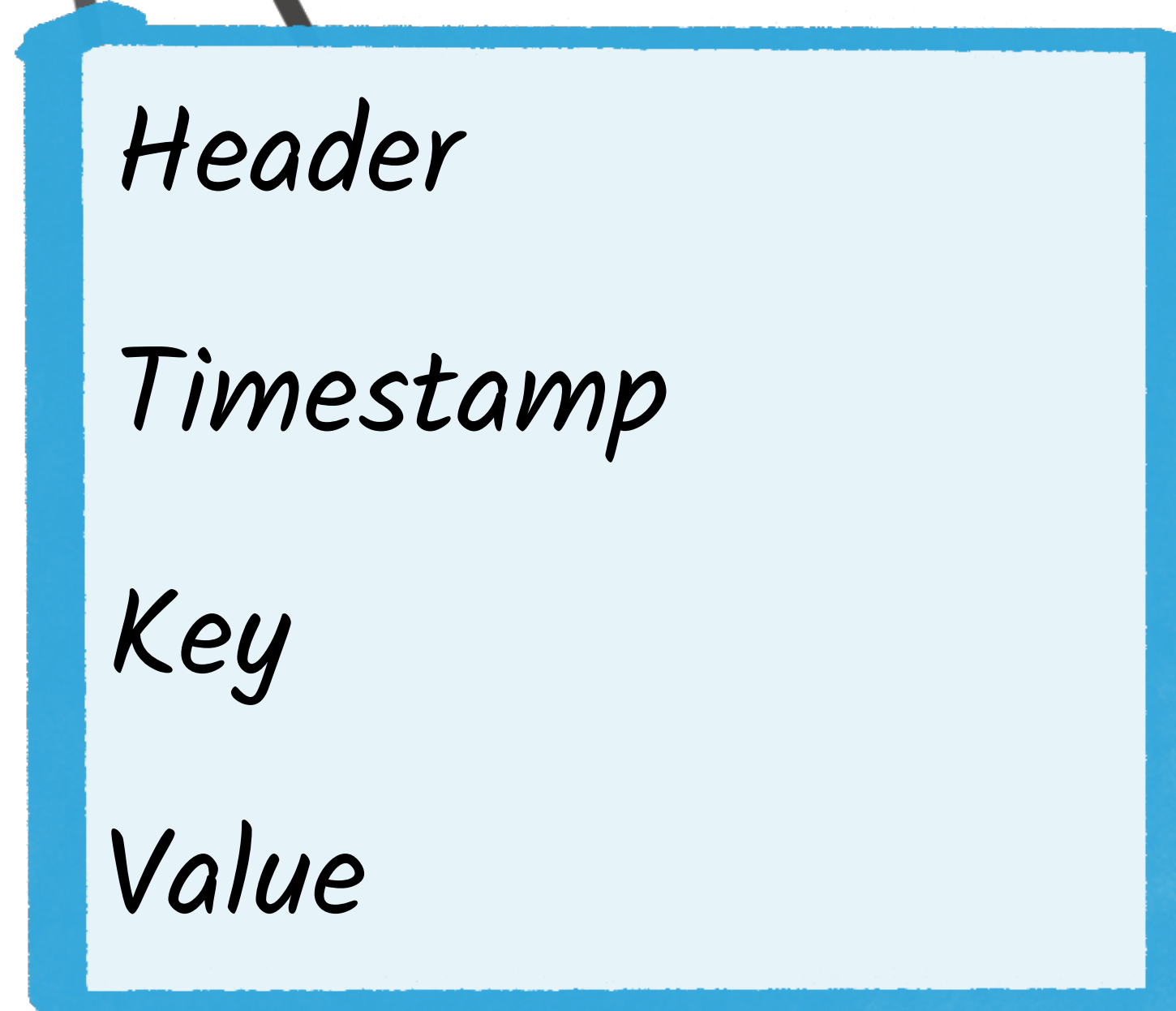
Messages are guaranteed to be  
strictly ordered within a partition



# Messages are just K/V bytes

*plus headers + timestamp*

Clicks





# Messages are just K/V bytes

With great power comes great responsibility

Avro

-> Confluent  
Schema Registry

Protobuf

JSON

CSV



**Gwen (Chen) Shapira**

@gwenshap

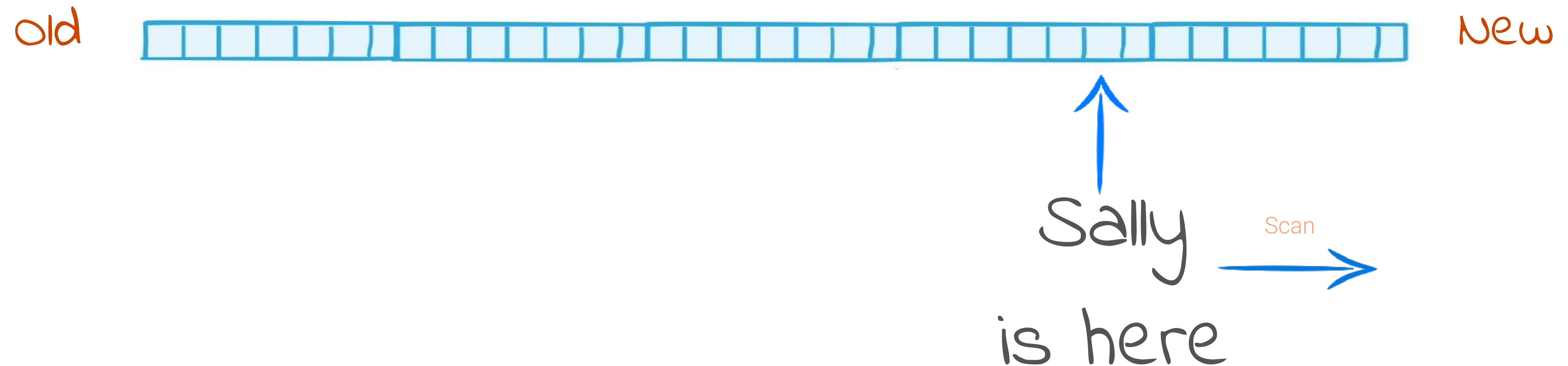
If your dev process doesn't validate schema compatibility somewhere between your IDE and production - you are screwed and don't know it.

5:50 AM - 5 Apr 2017

[https://qconnewyork.com/system/files/presentation-slides/qcon\\_17\\_-\\_schemas\\_and\\_apis.pdf](https://qconnewyork.com/system/files/presentation-slides/qcon_17_-_schemas_and_apis.pdf)

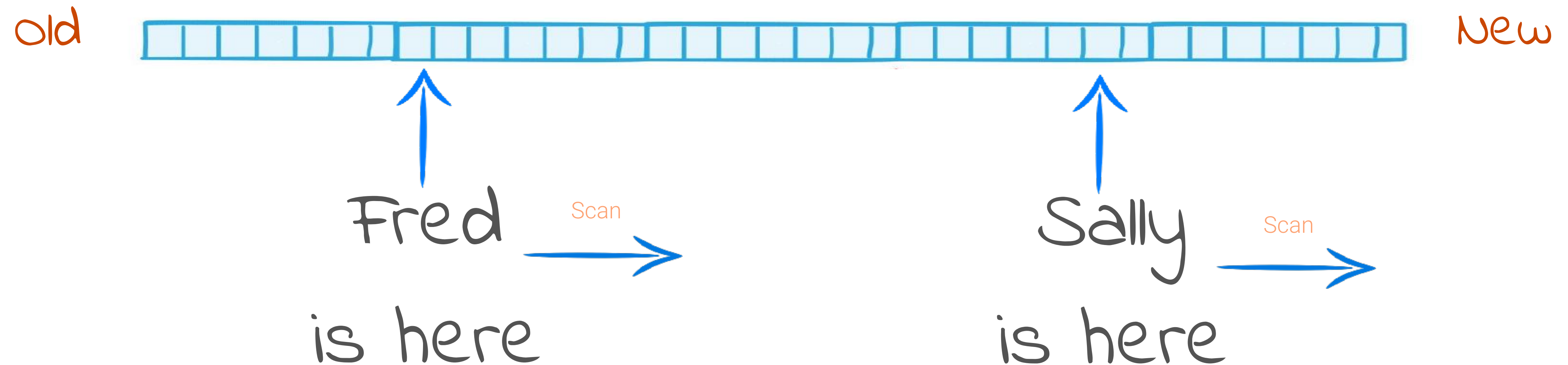


# Consumers have a position all of their own



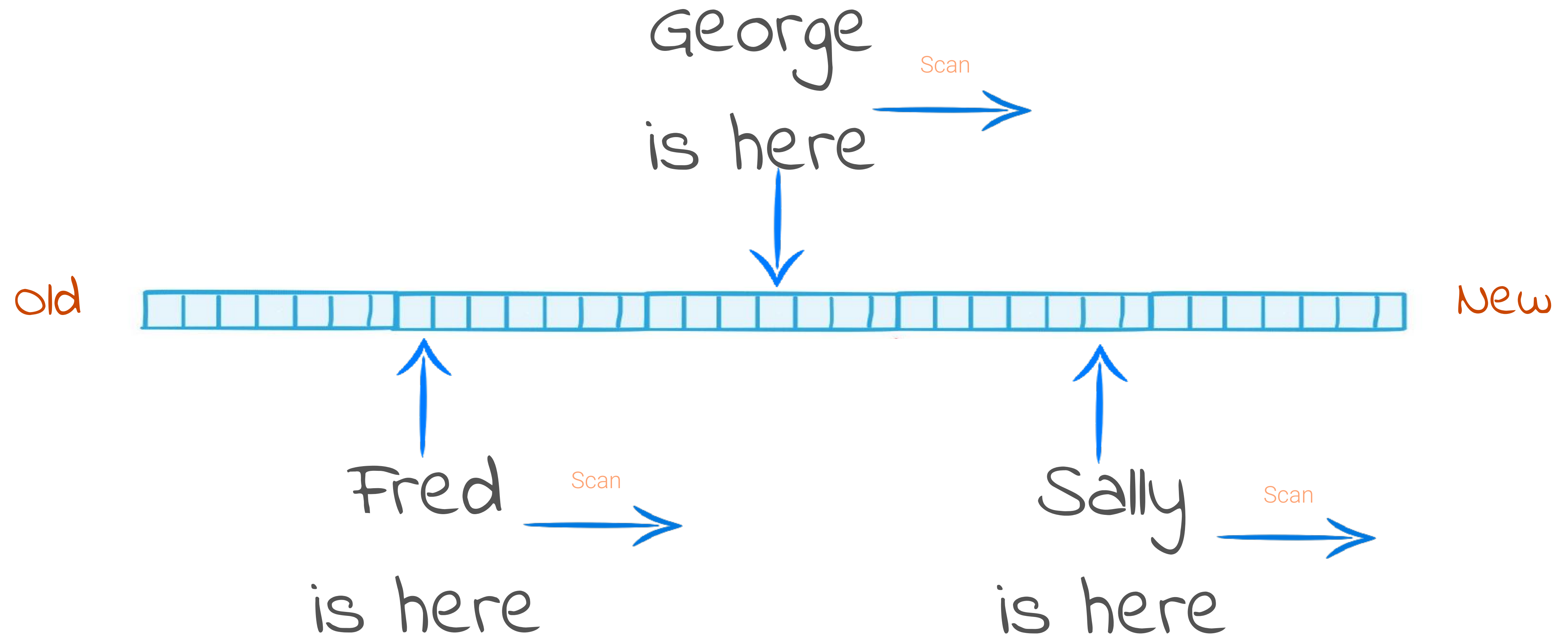


# Consumers have a position all of their own



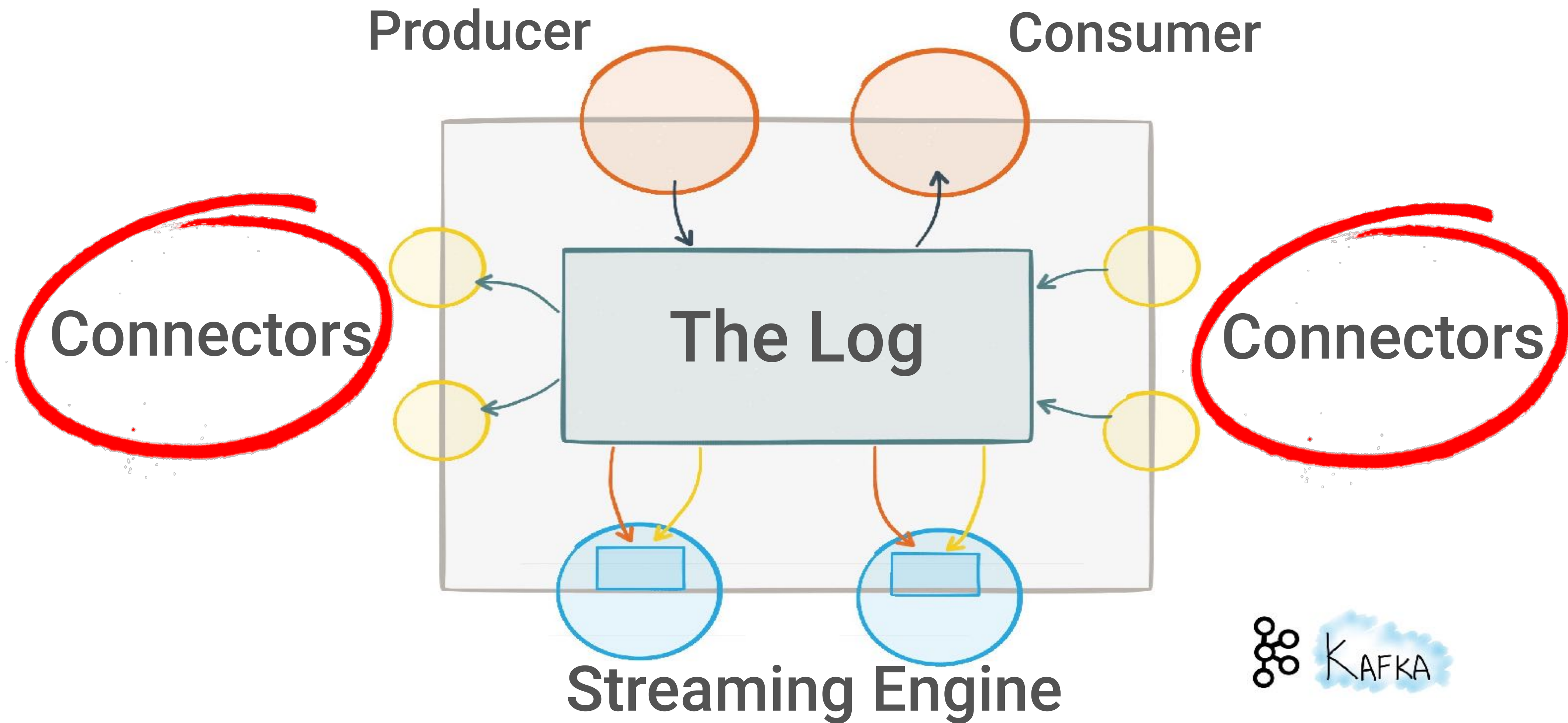


# Consumers have a position all of their own



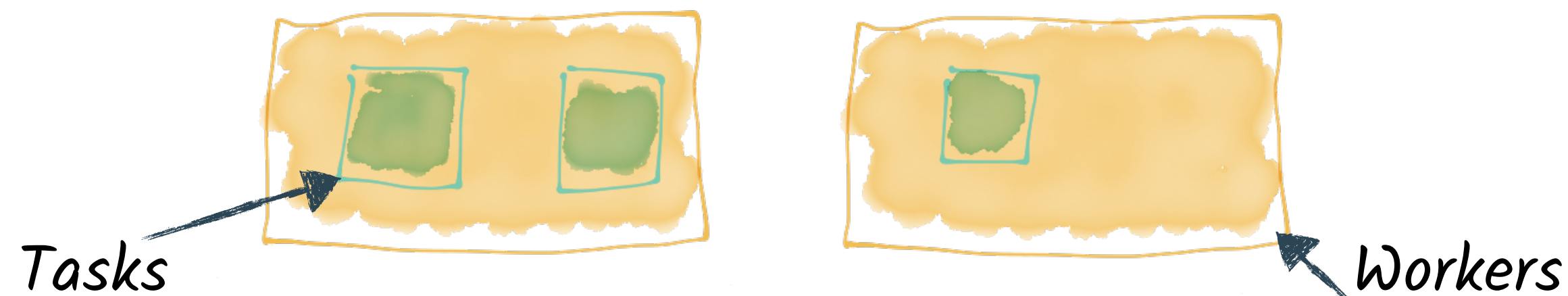


# The Connect API

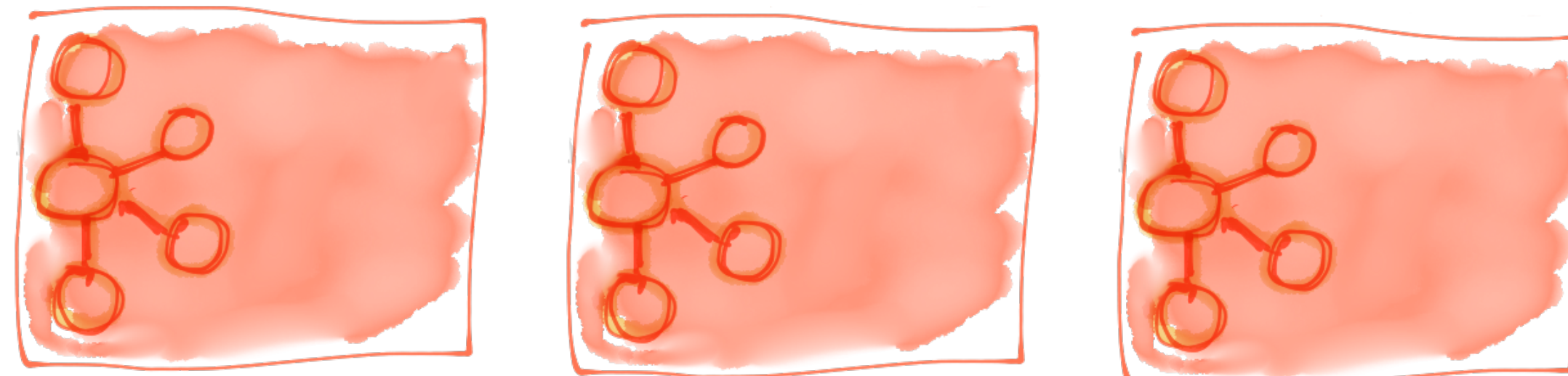




# Streaming Integration with Kafka Connect



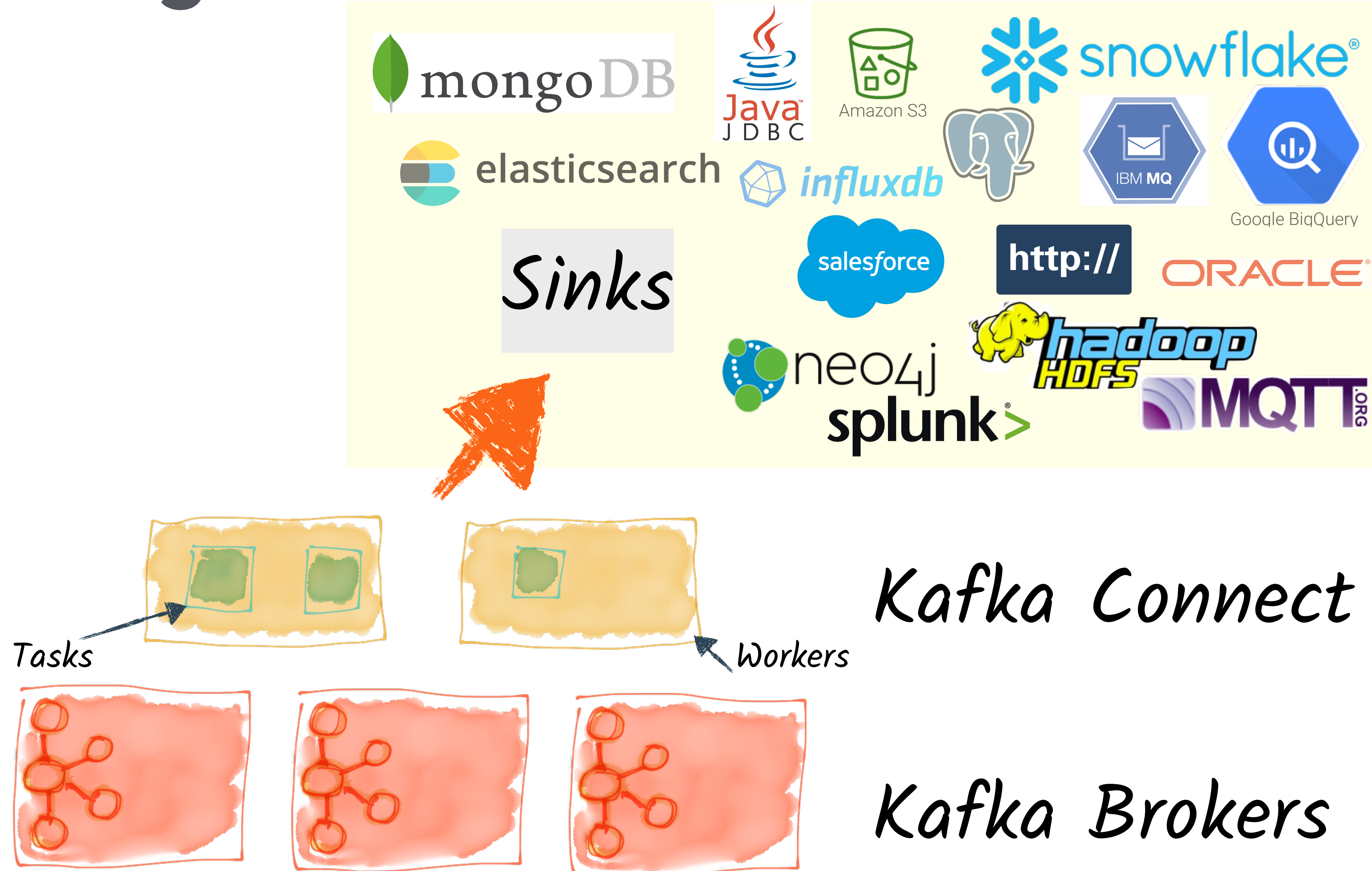
*Kafka Connect*



*Kafka Brokers*

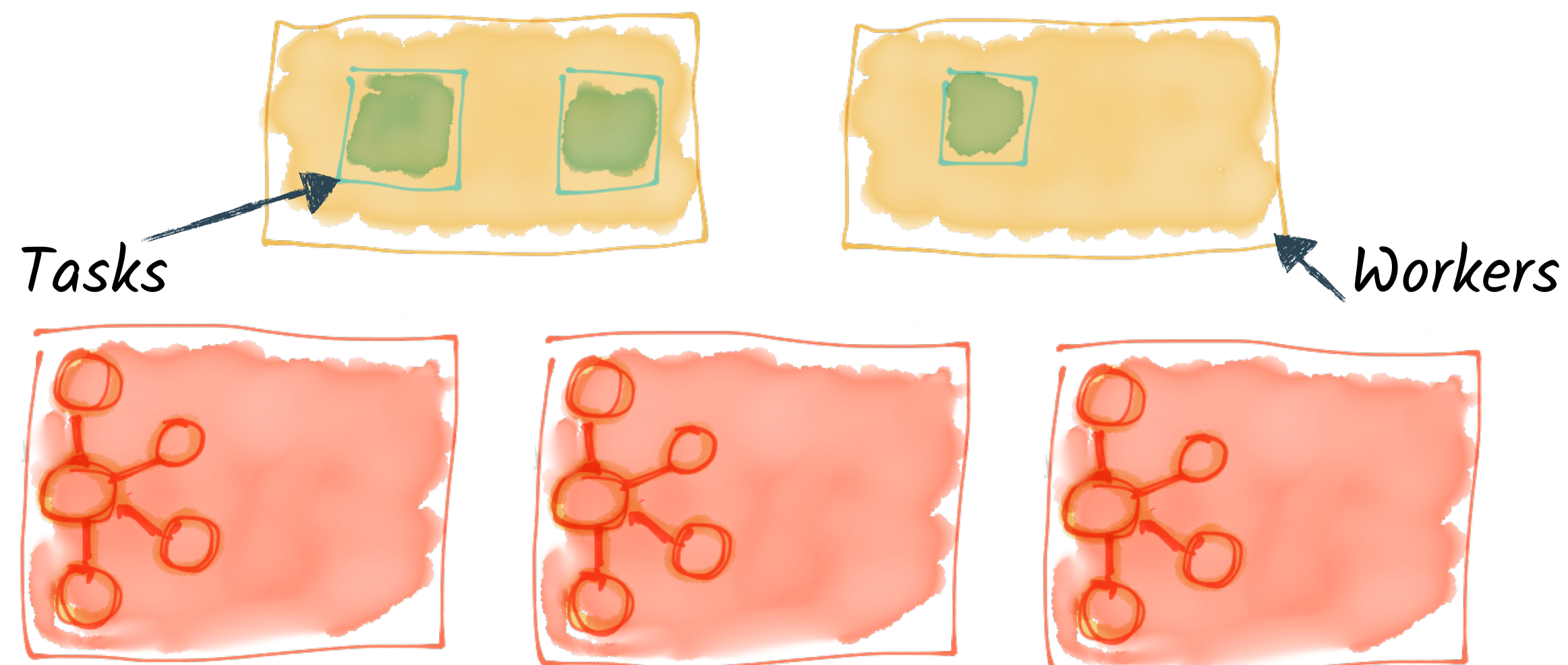


# Streaming Integration with Kafka Connect





# Streaming Integration with Kafka Connect

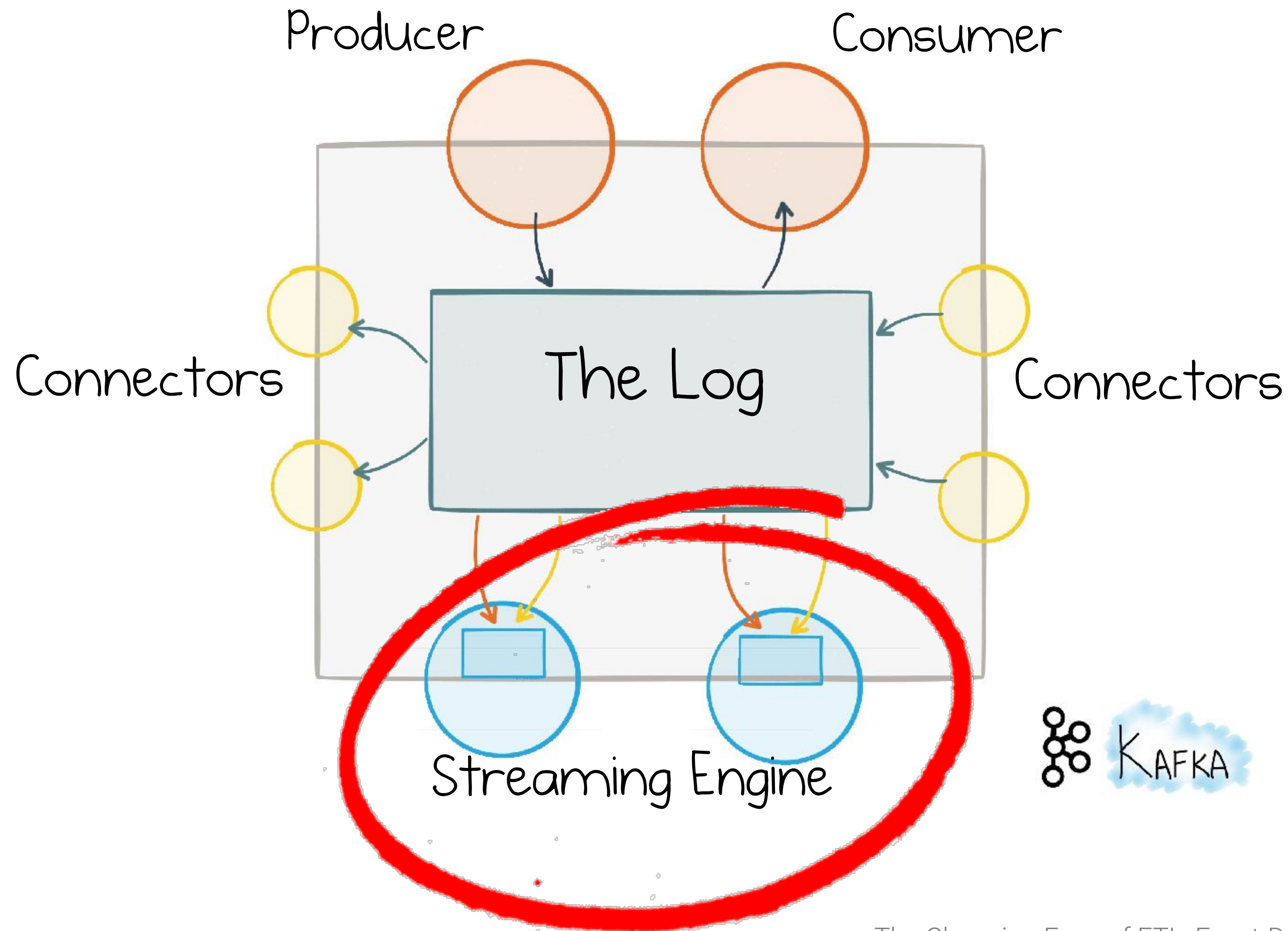


*Kafka Connect*

*Kafka Brokers*



# Stream Processing in Kafka





# Kafka Streams API



```
final StreamsBuilder builder = new StreamsBuilder()  
    .stream("orders", Consumed.with(stringSerde, ordersSerde))  
    .filter( (key, order) -> order.getStatus().equals("COMPLETE") )  
    .to("complete_orders", Produced.with(stringSerde, ordersSerde));
```





# Stream Processing with ksqlDB



```
CREATE STREAM completedOrders AS  
SELECT *  
FROM orders  
WHERE status='COMPLETE' ;
```



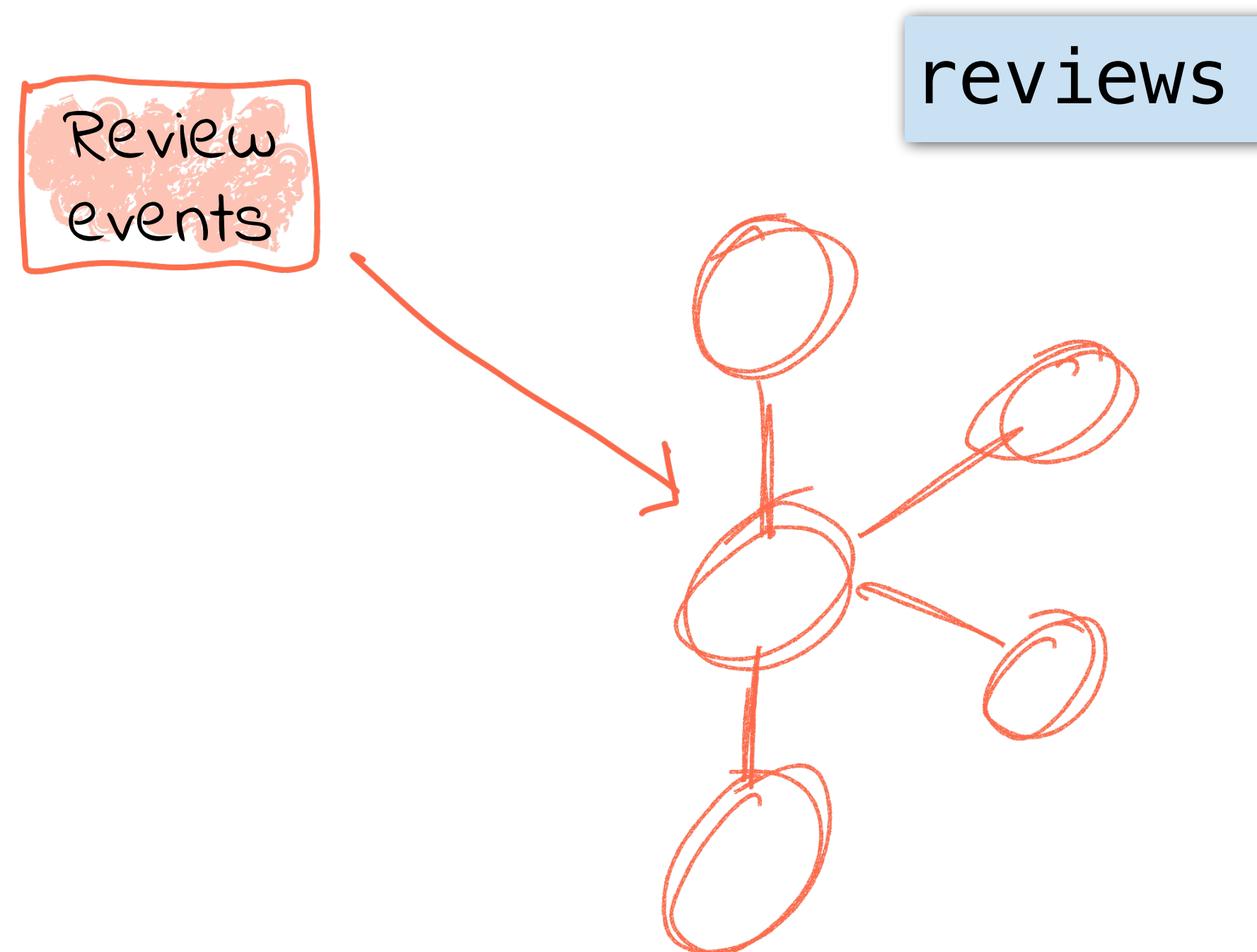




**This is  
Something  
New**

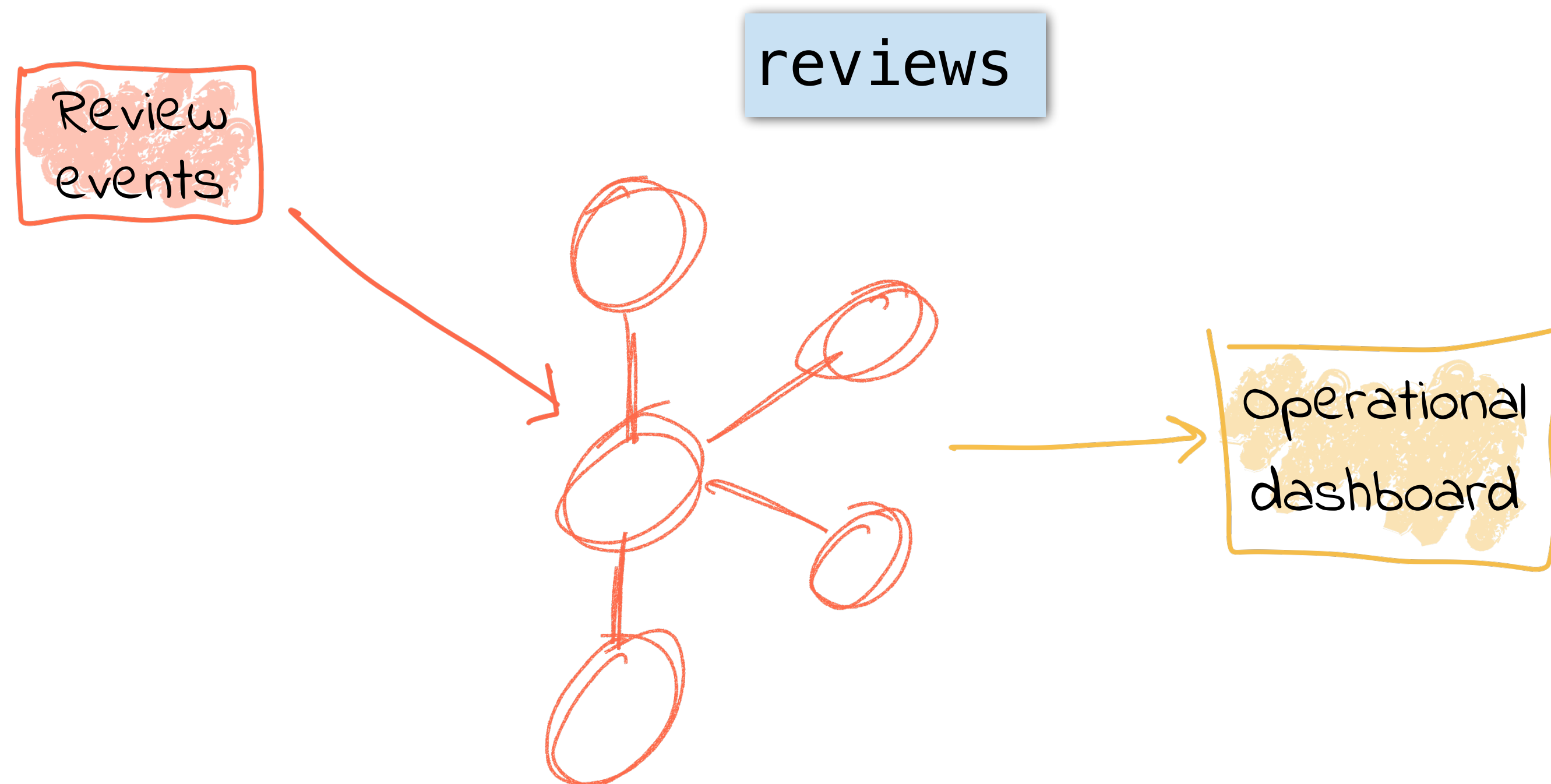


# Events in Action



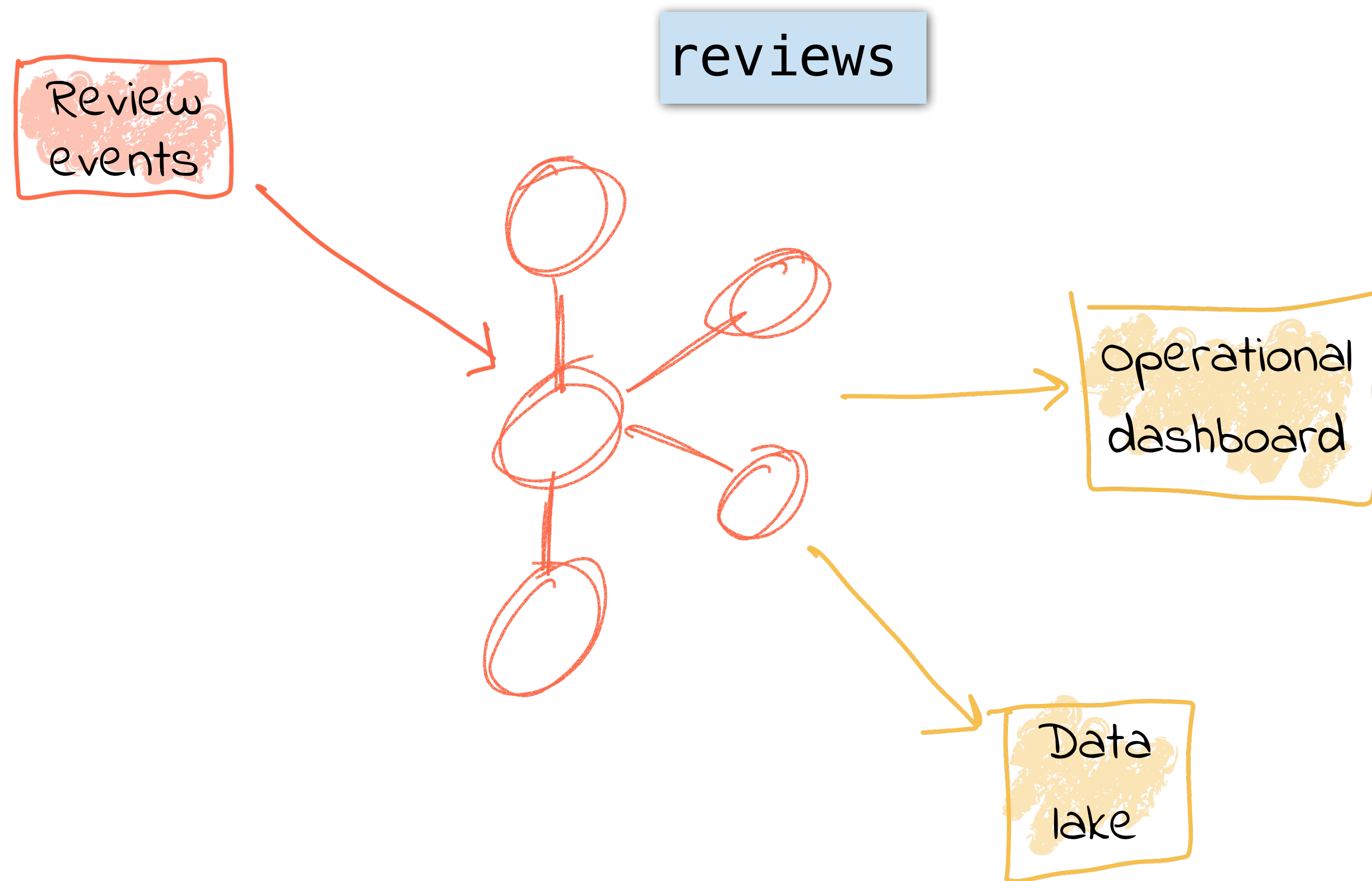


# Events in Action



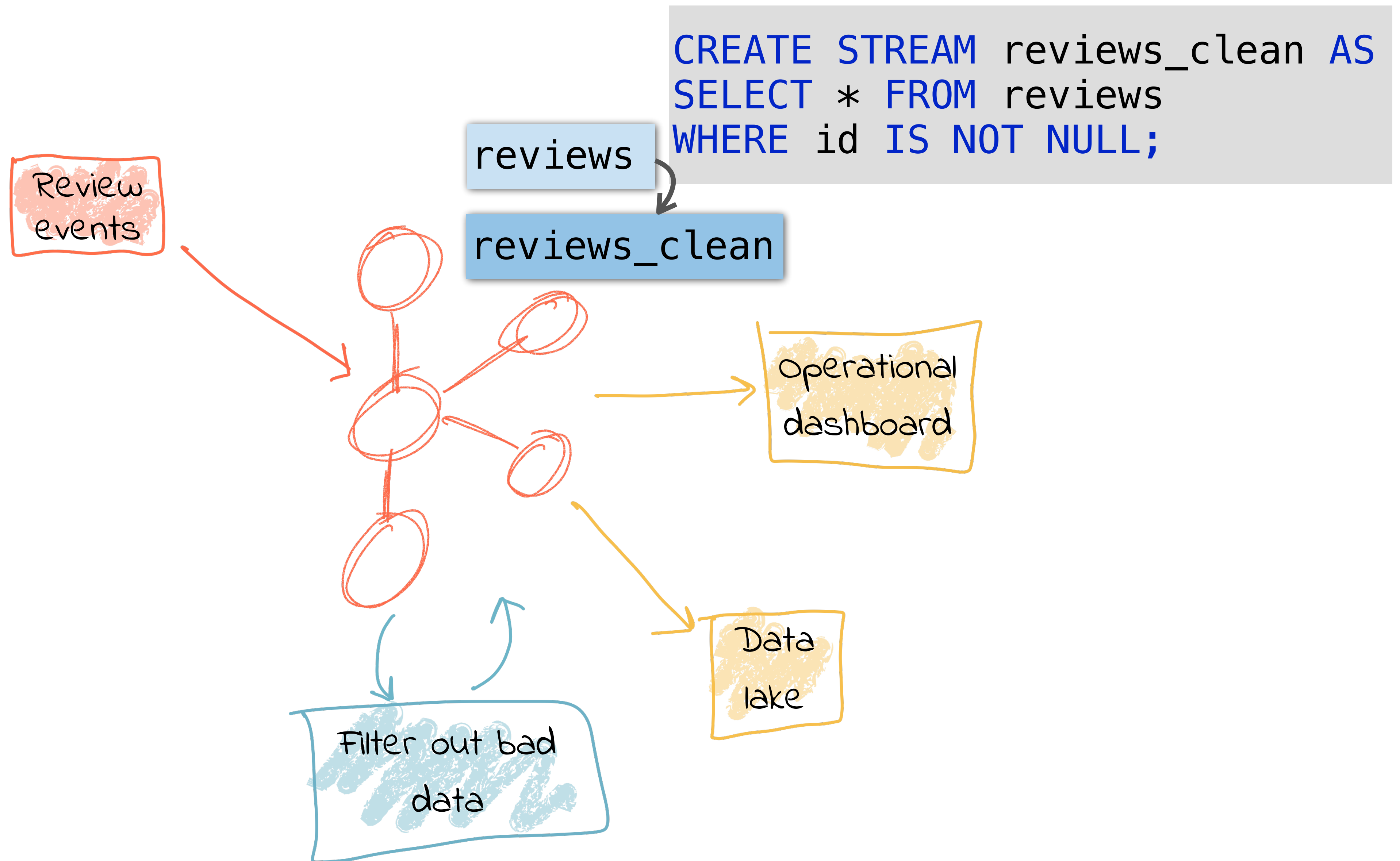


# Events in Action



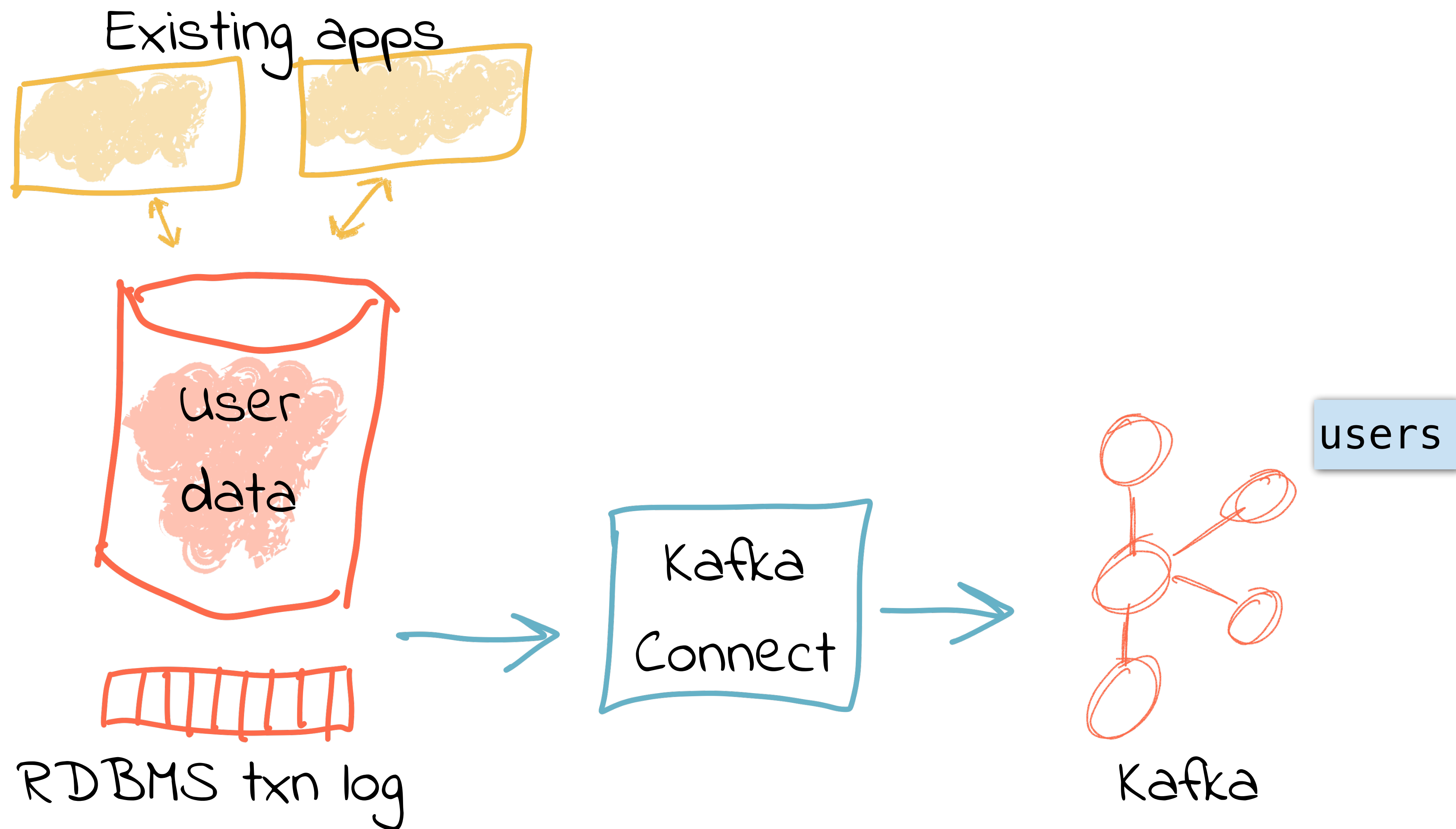


# Events in Action



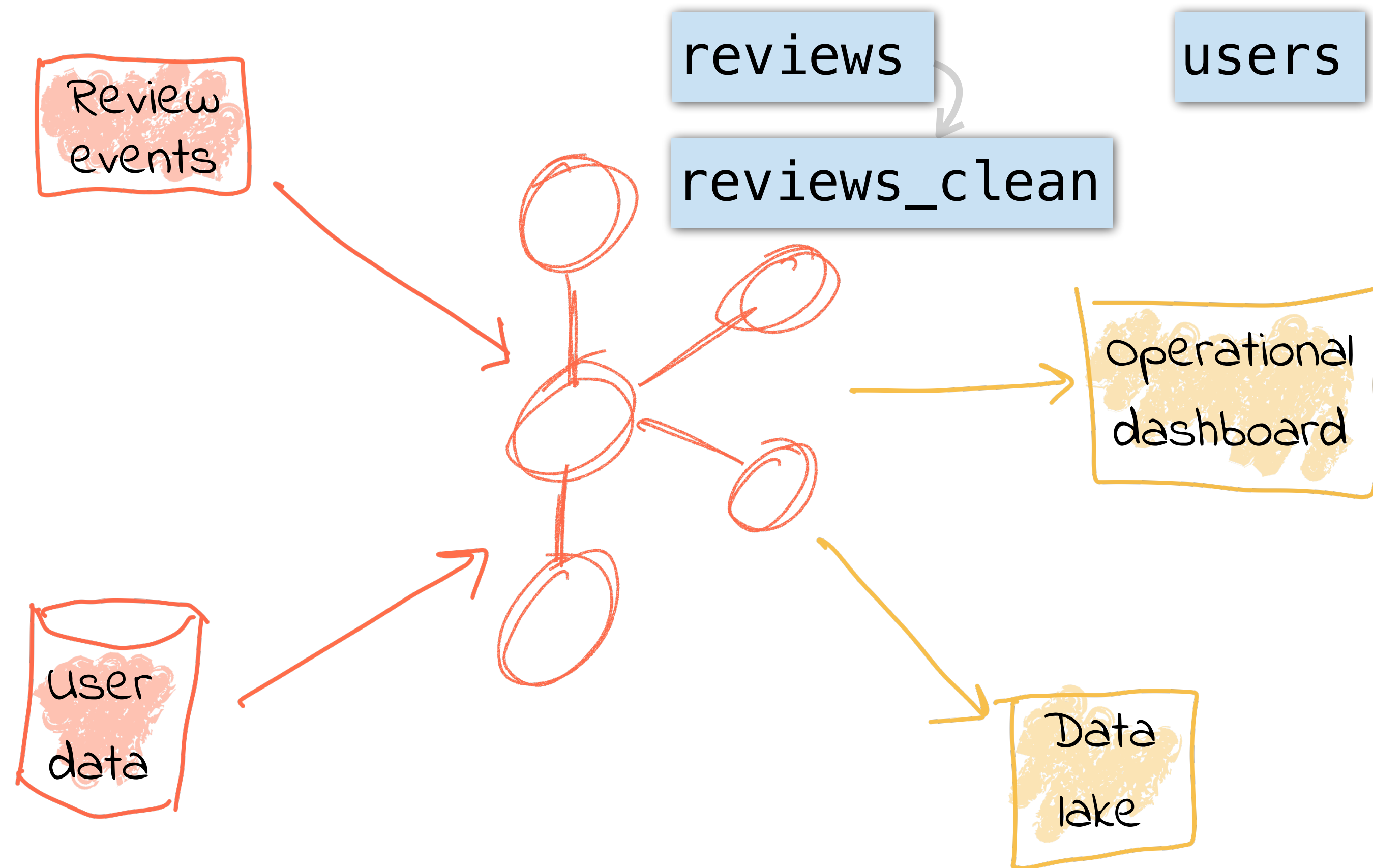


# Events in Action





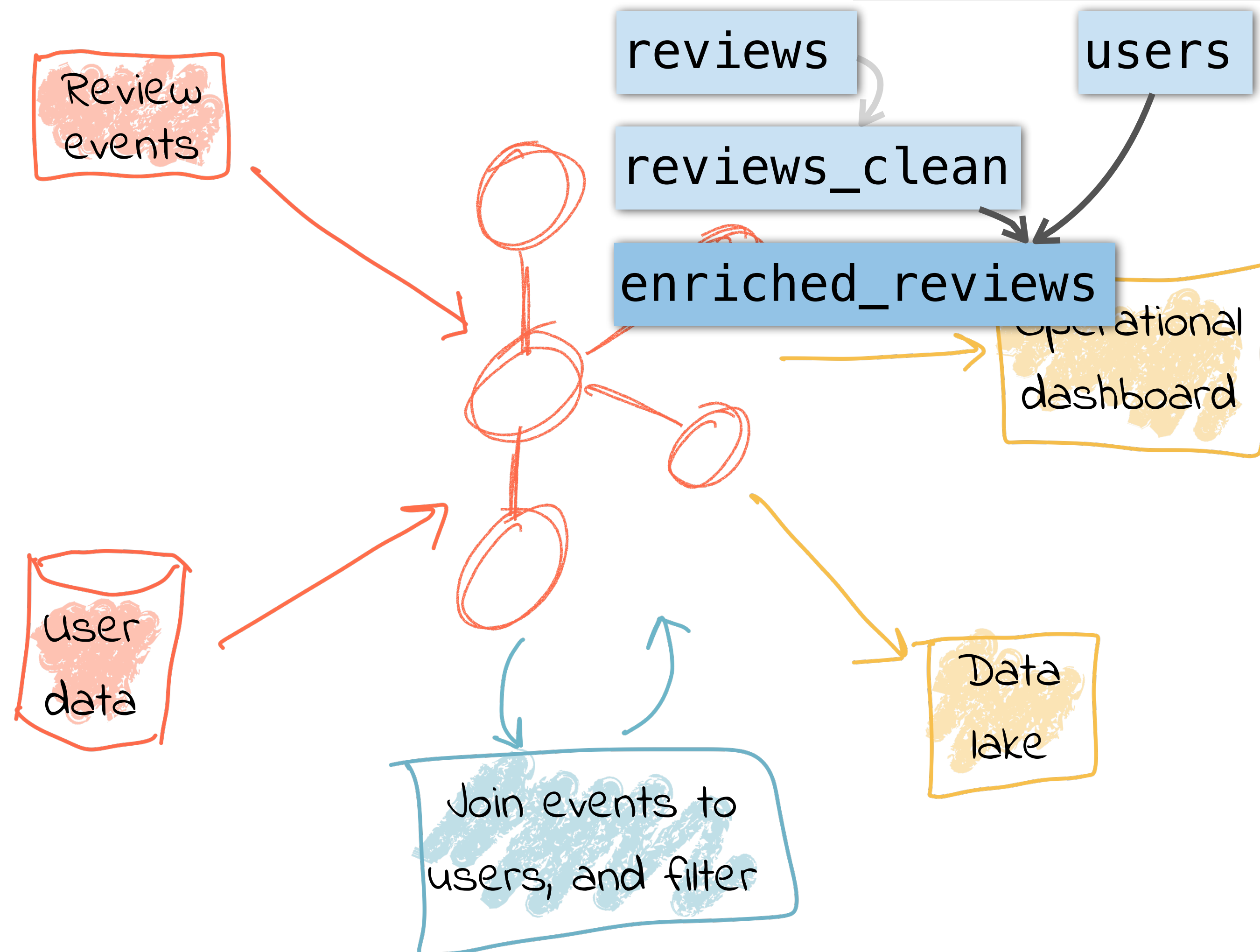
# Events in Action





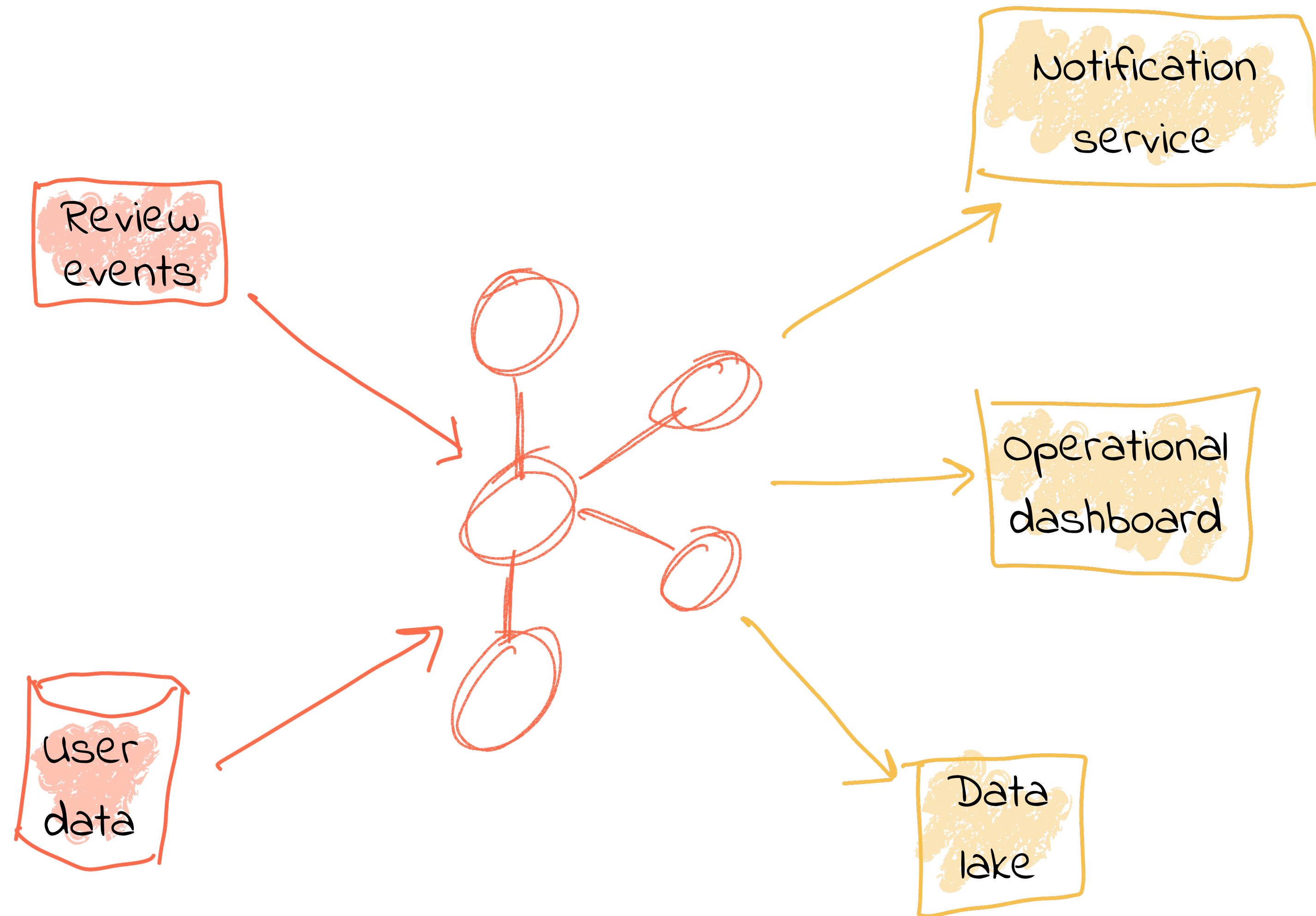
# Events in Action

```
CREATE STREAM enriched_reviews AS  
SELECT * FROM reviews_clean r  
      INNER JOIN users u  
      ON r.userid=u.userid;
```





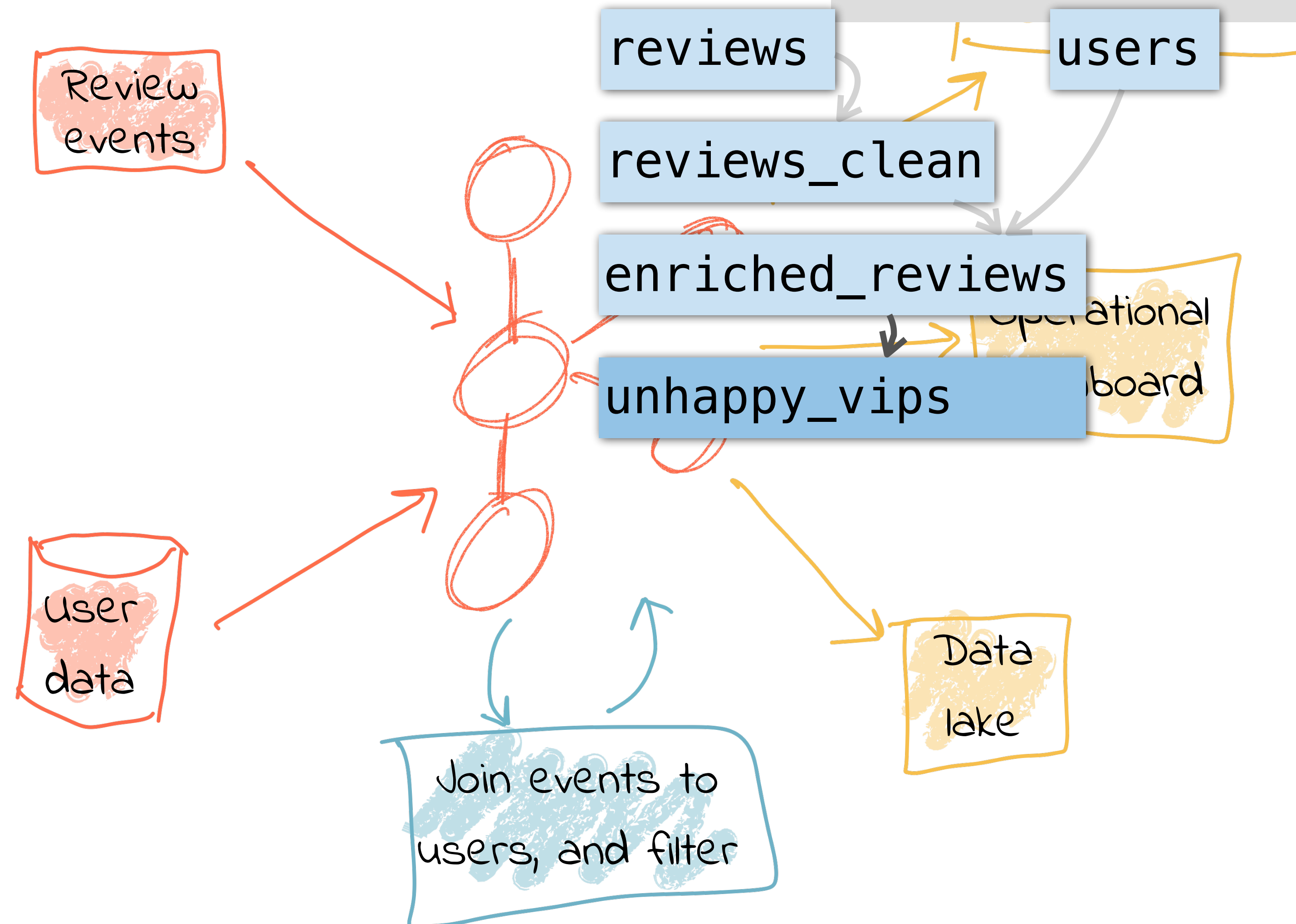
# Events in Action





# Events in Action

```
CREATE STREAM unhappy_vips AS
SELECT * FROM enriched_reviews
WHERE rating < 3
AND status = 'Platinum';
```



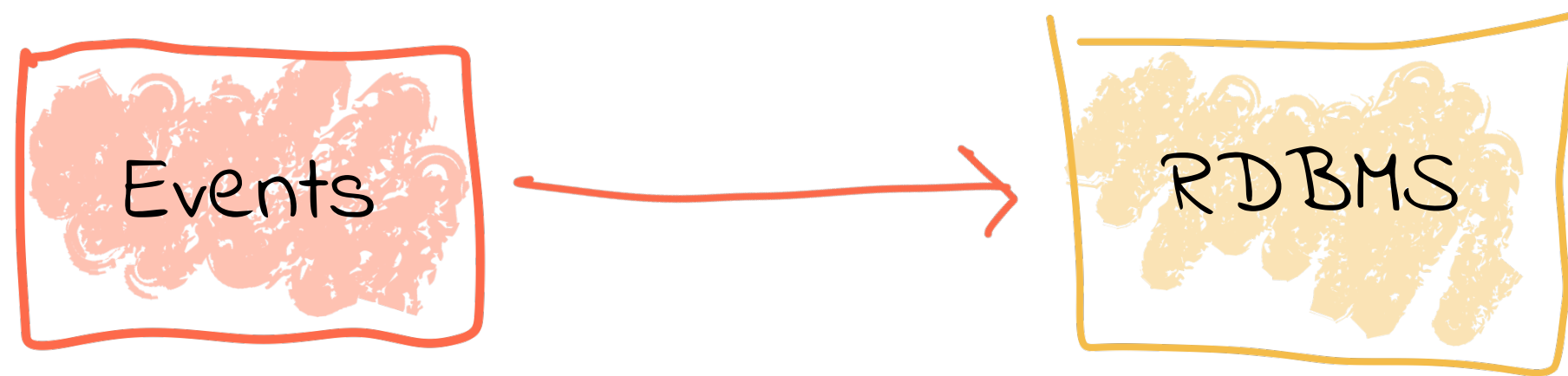


# The Power of an Event-Driven Architecture



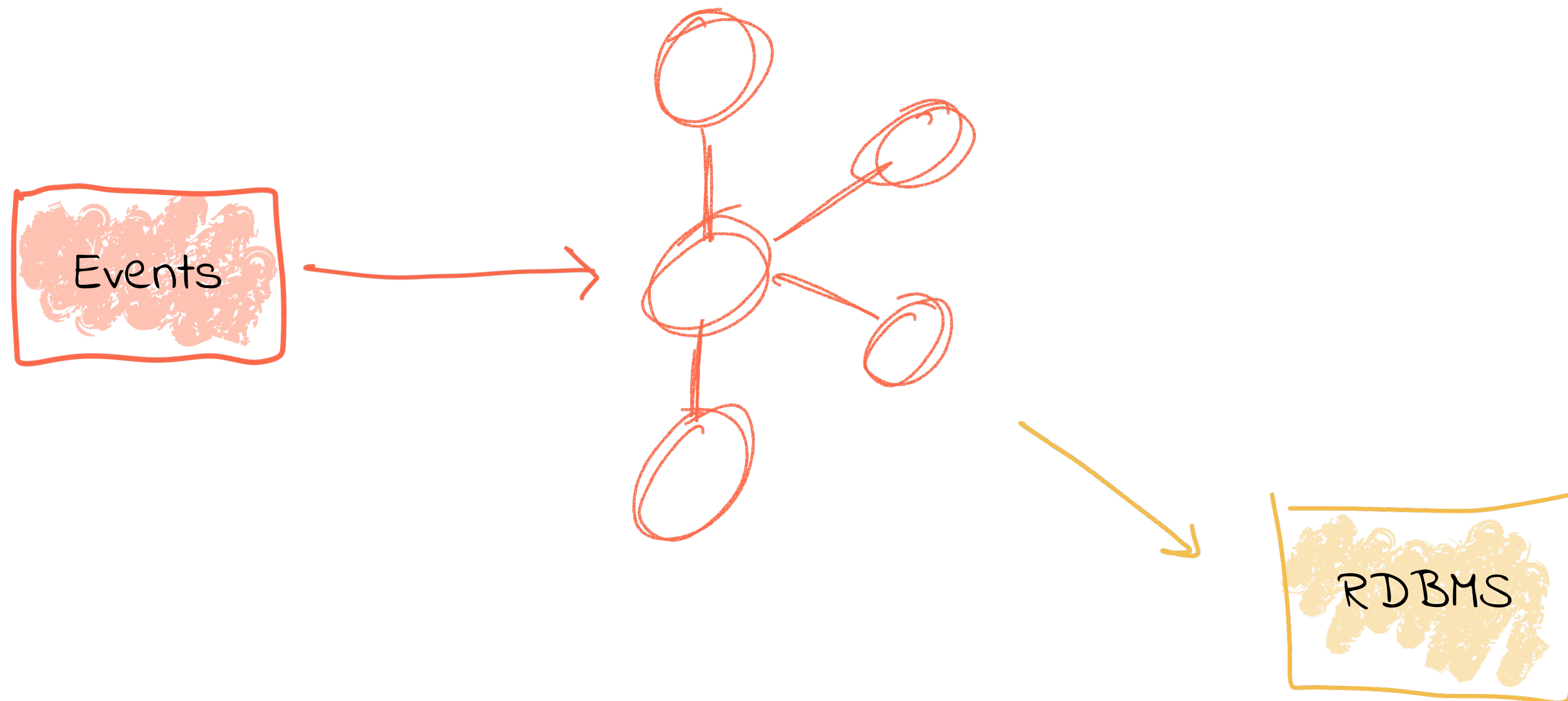


# Not Everything is a Nail



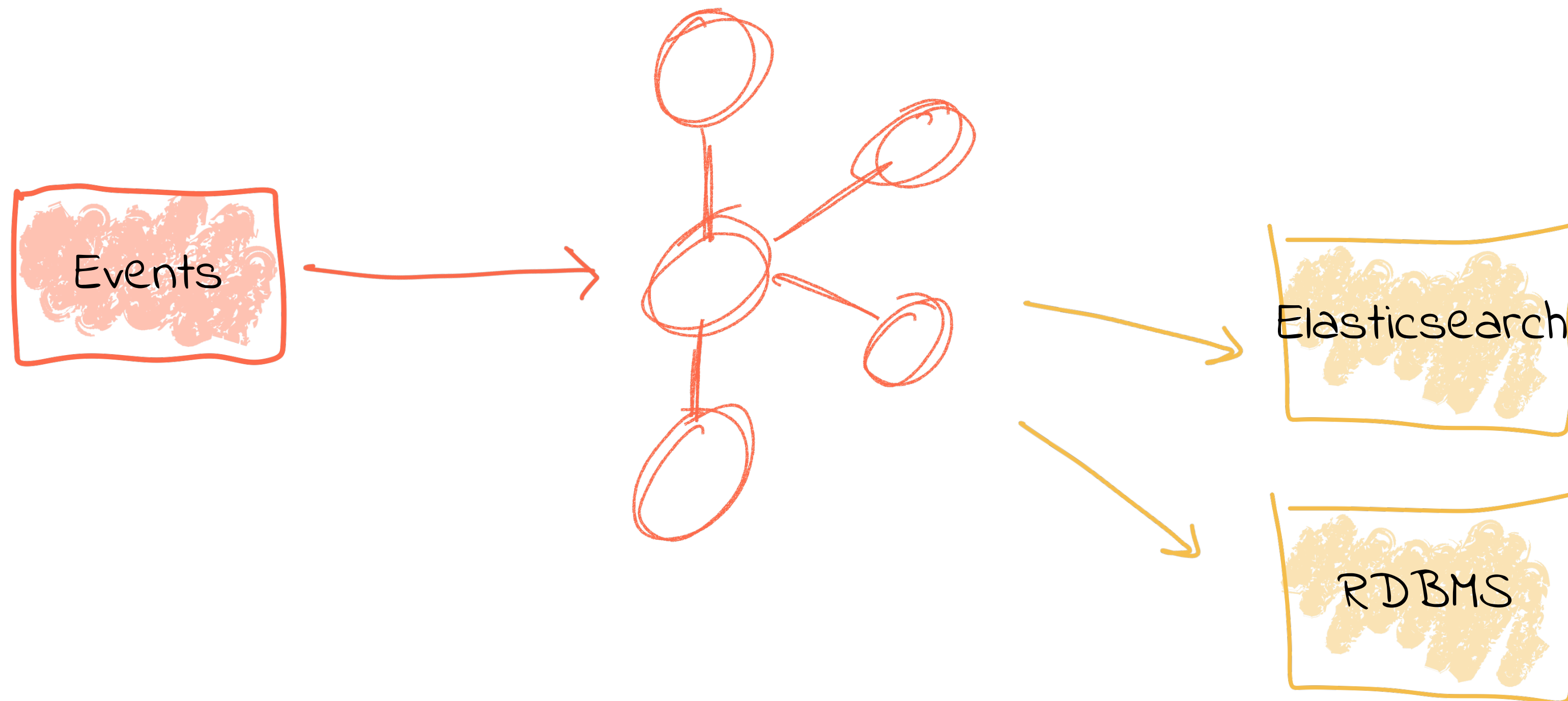


# Not Everything is a Nail



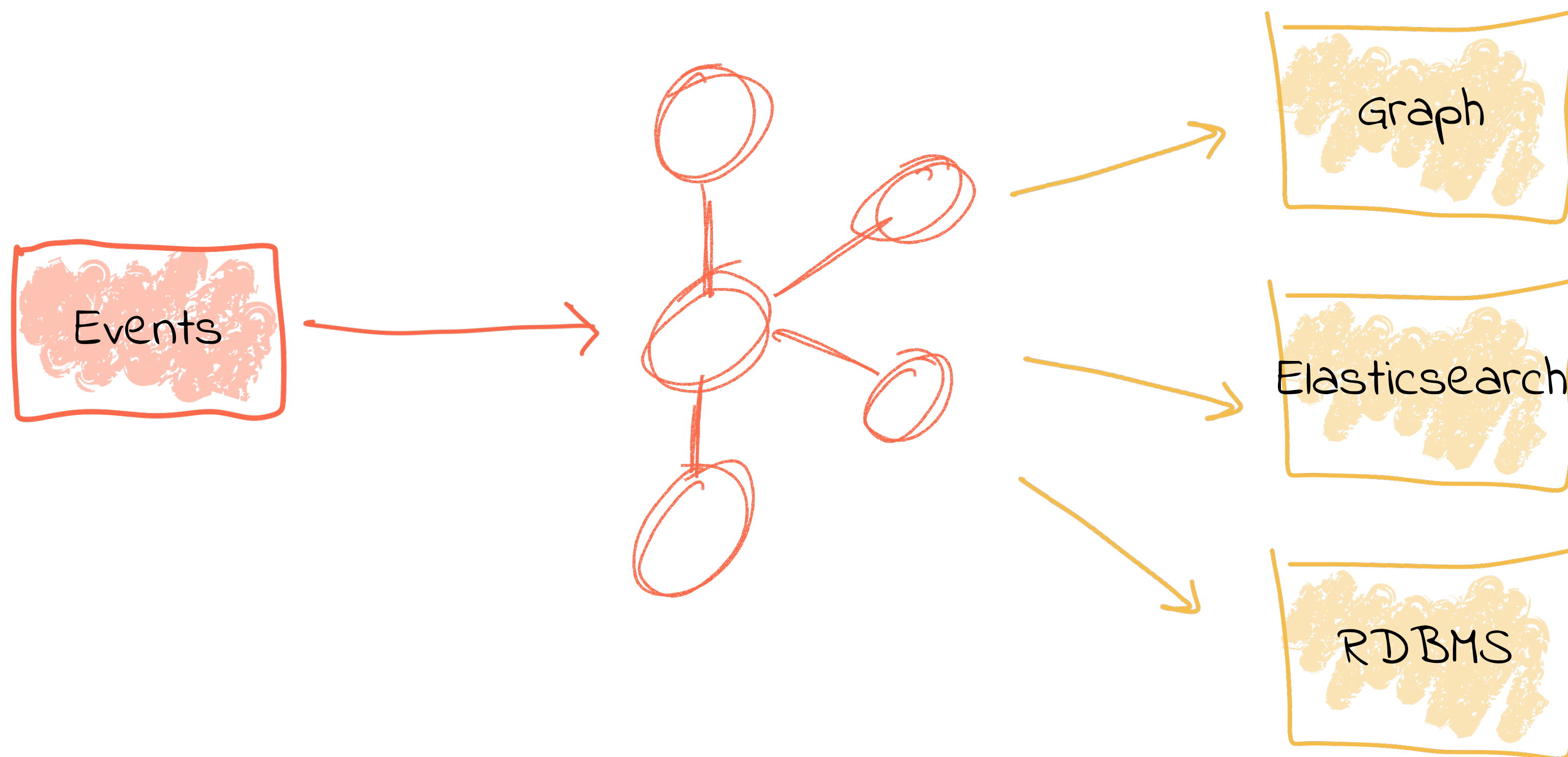


# Not Everything is a Nail



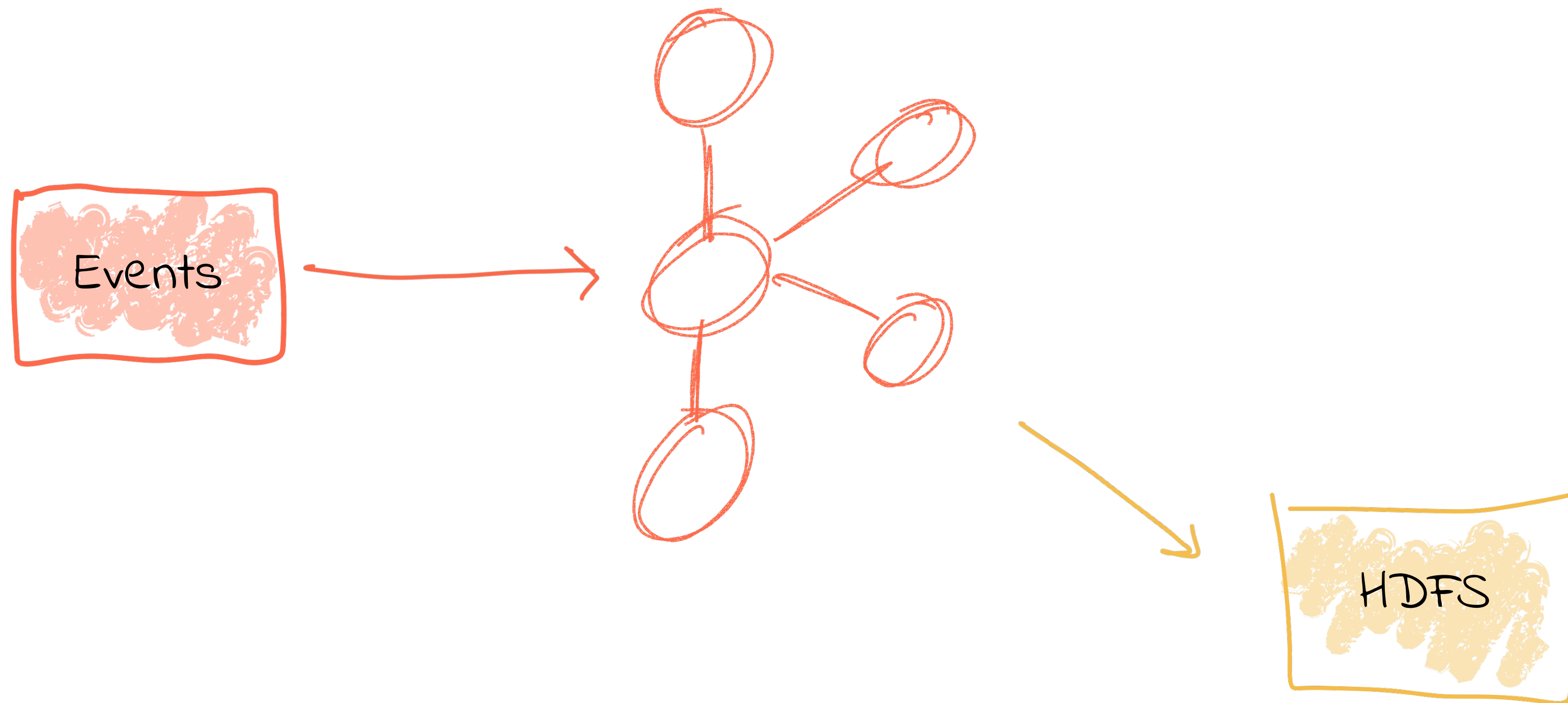


# Not Everything is a Nail



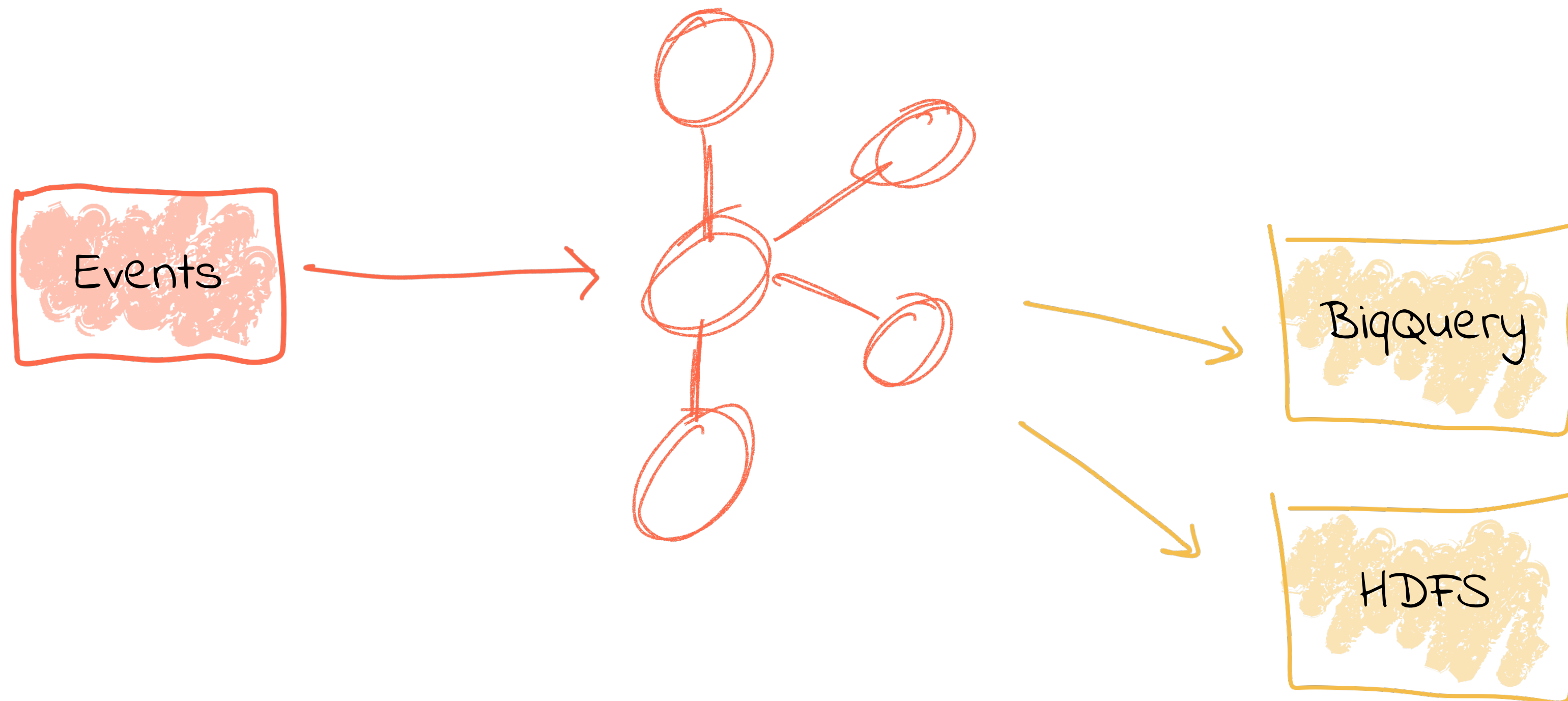


# Side-by-Side Tech Evaluation



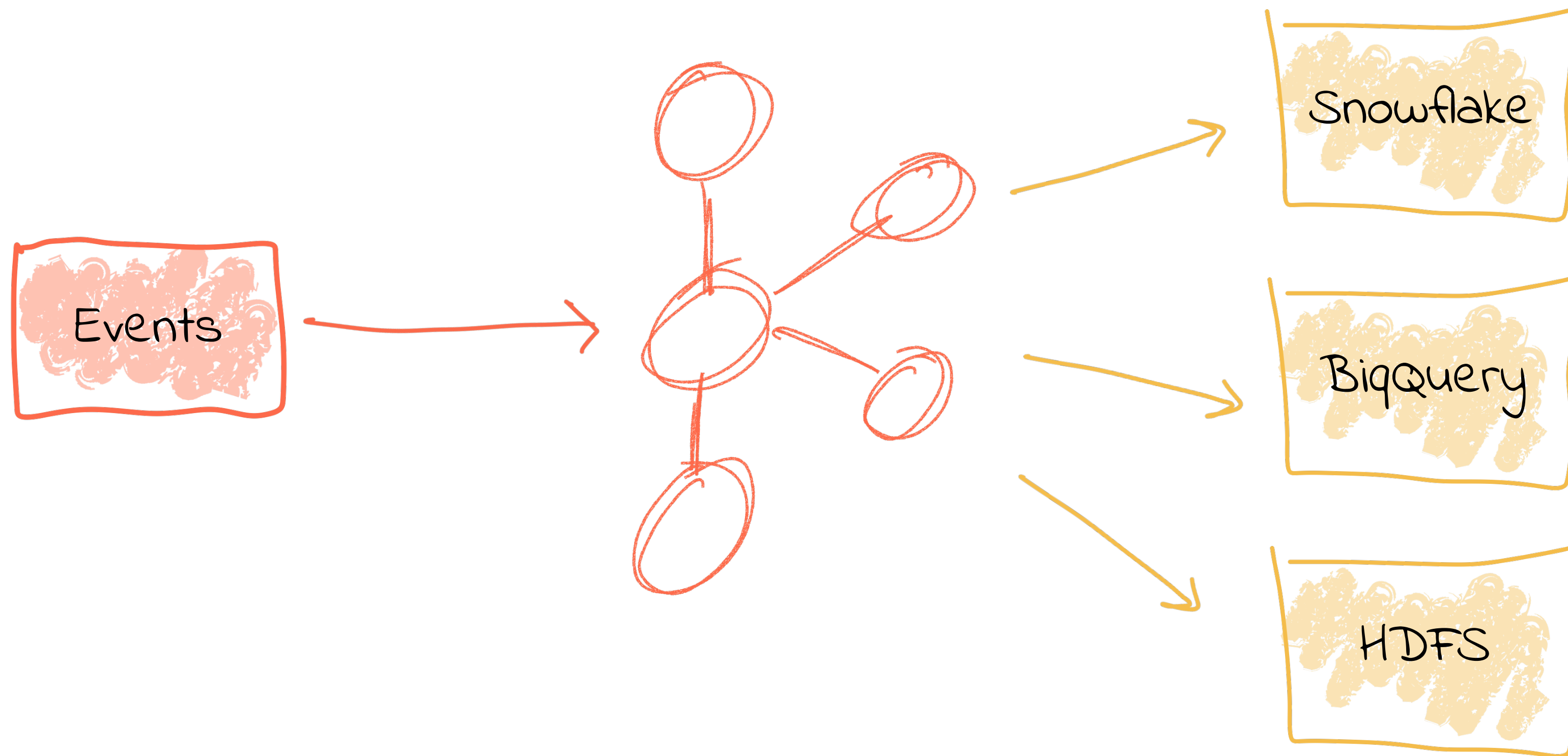


# Side-by-Side Tech Evaluation



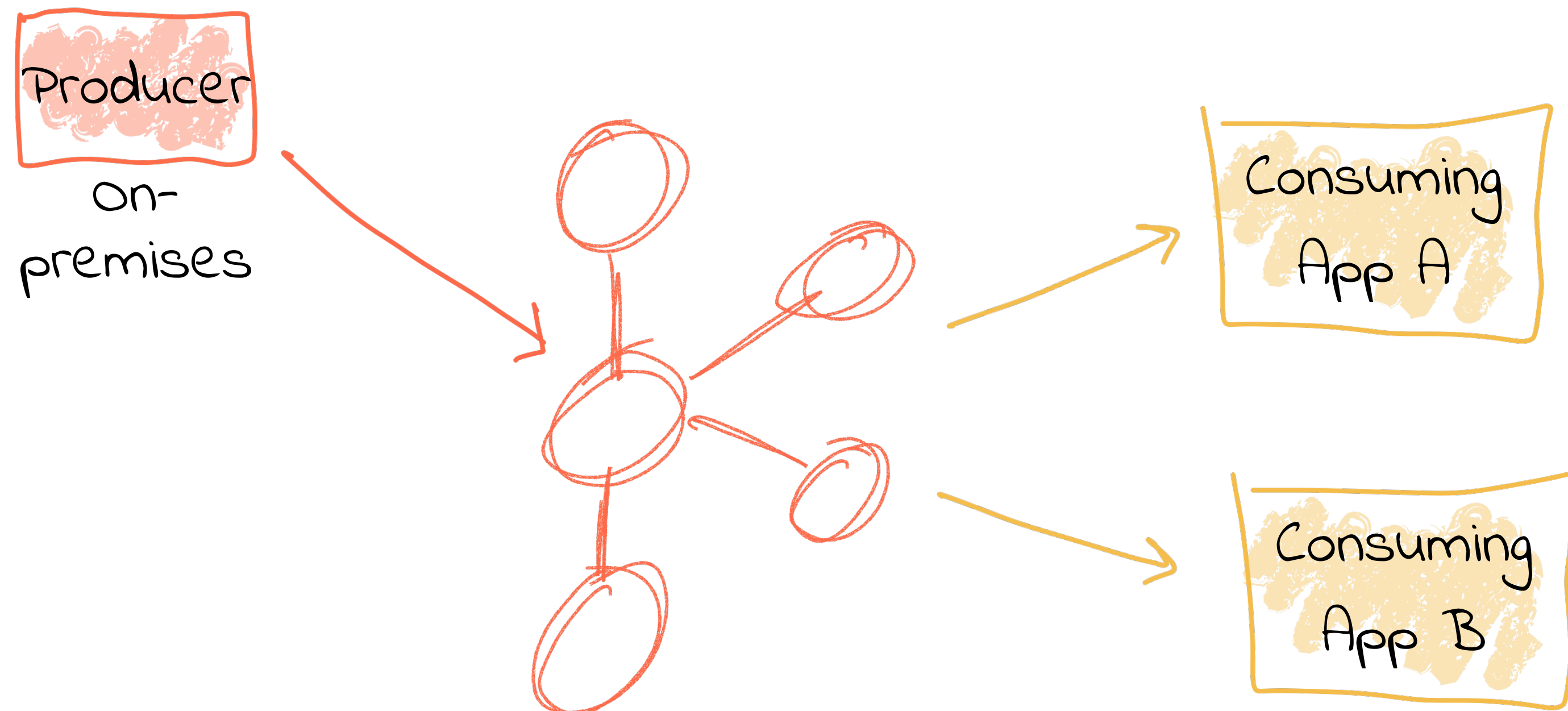


# Side-by-Side Tech Evaluation



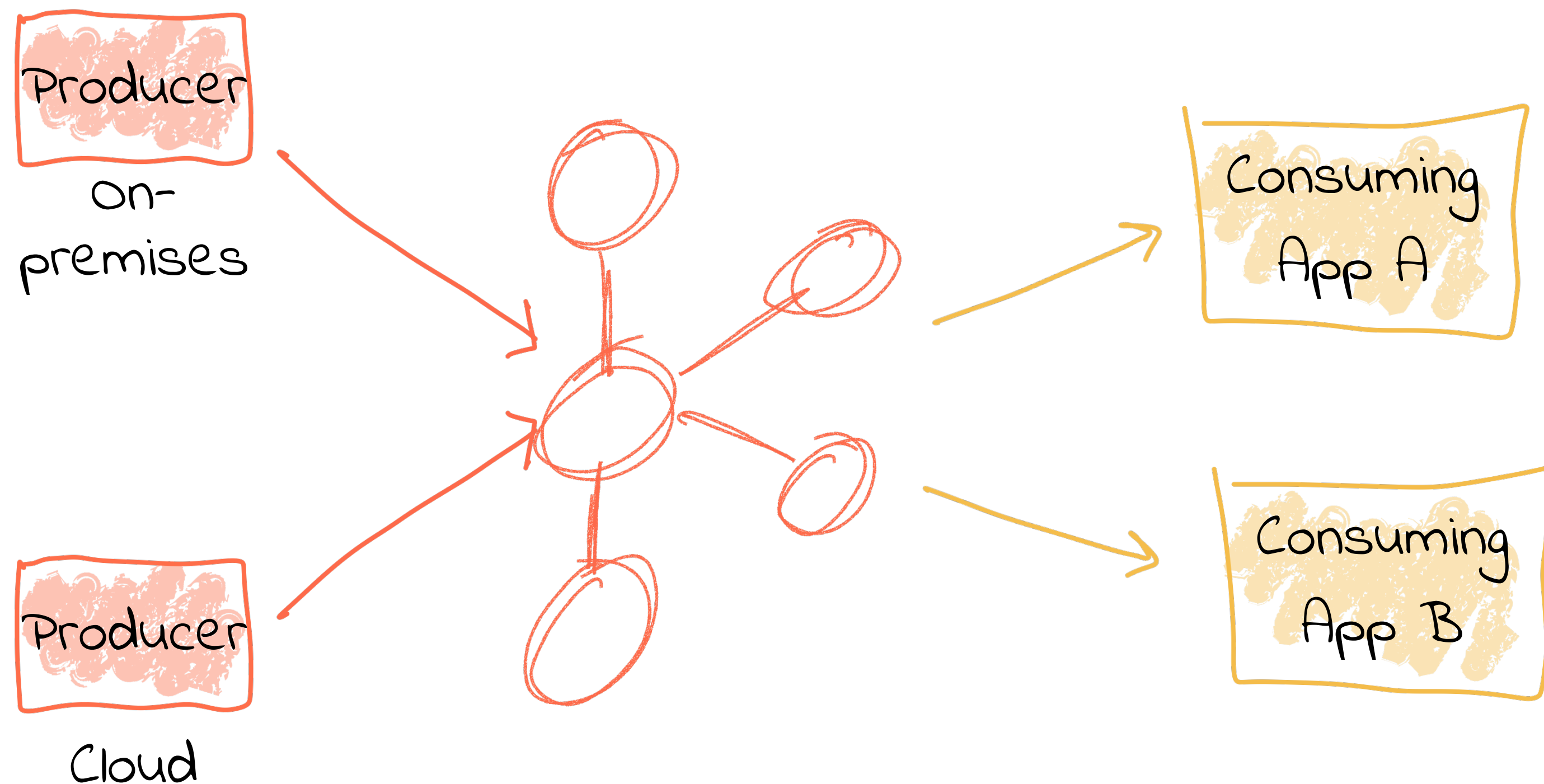


# Evolve Data Sources



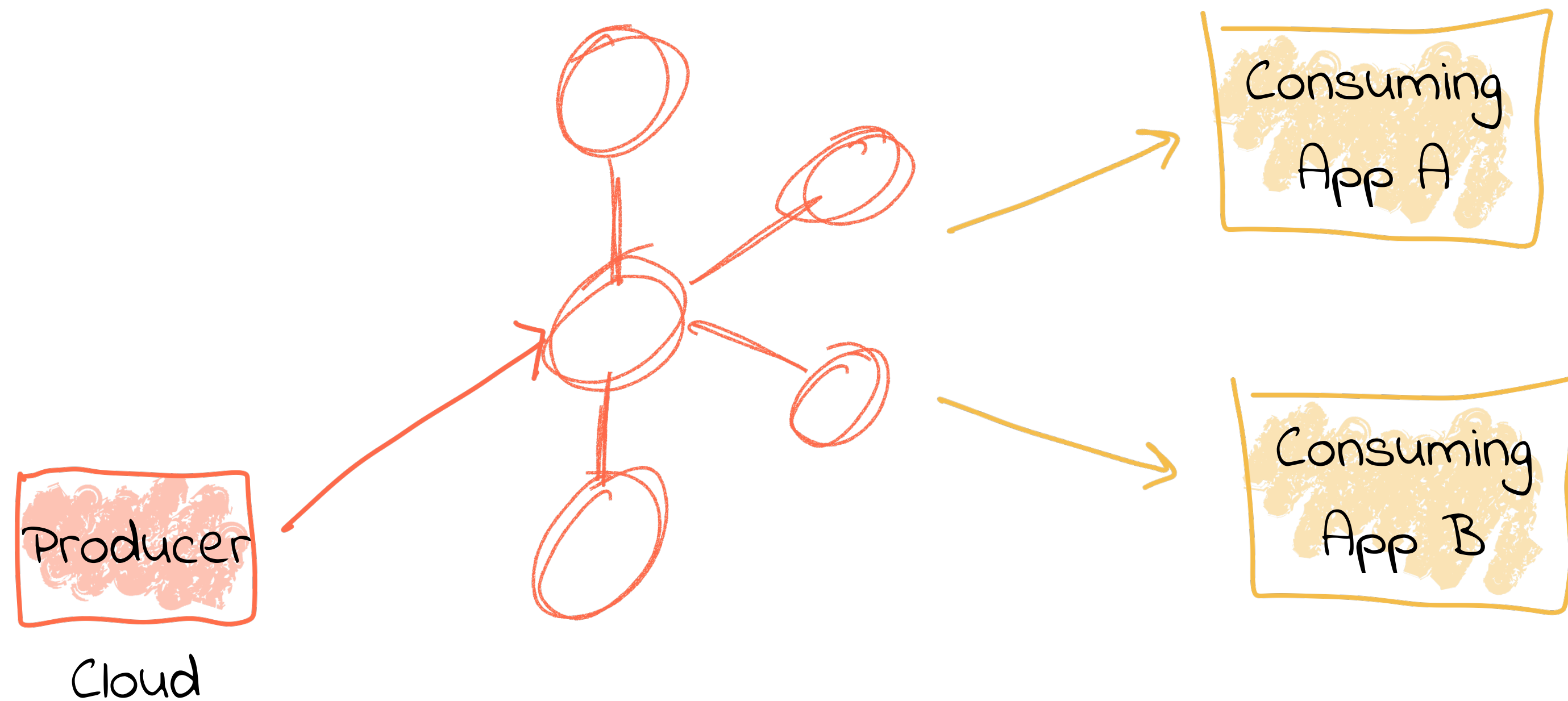


# Evolve Data Sources



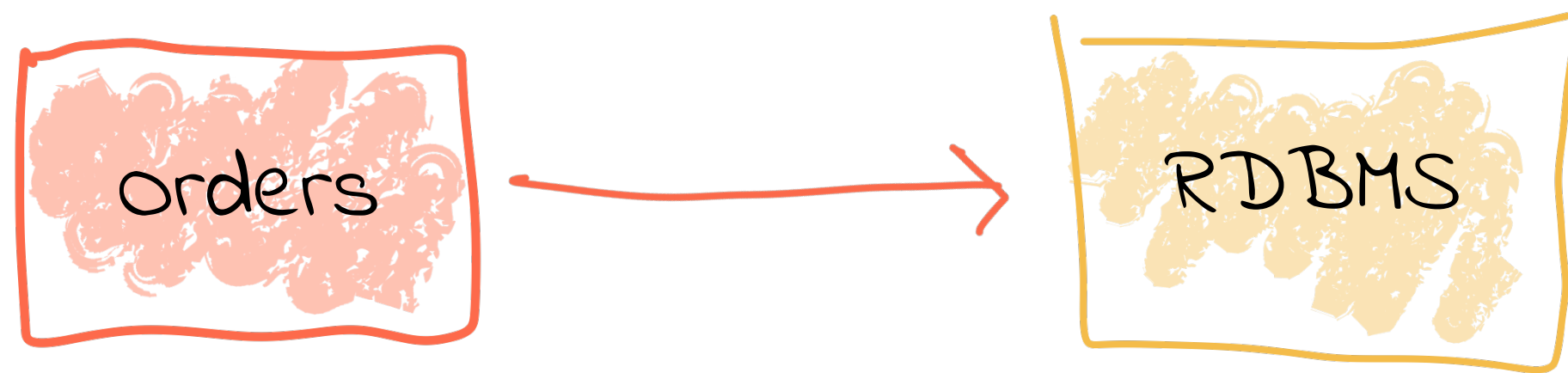


# Evolve Data Sources





# Tight Coupling != Flexible



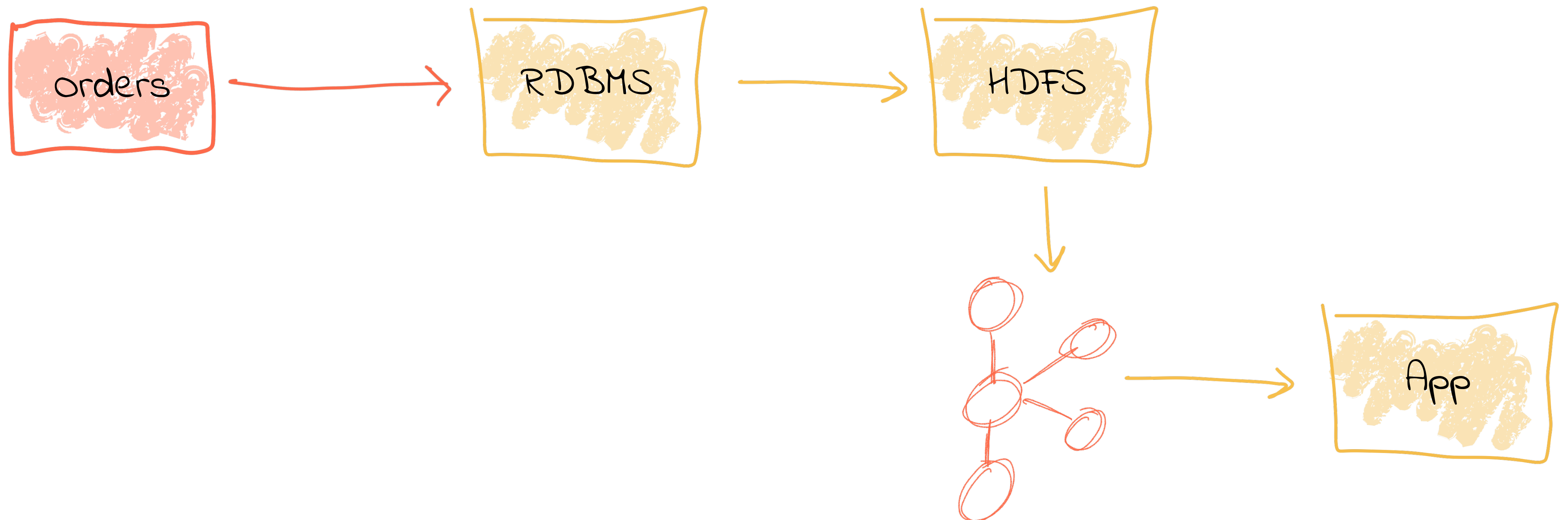


# Tight Coupling != Flexible



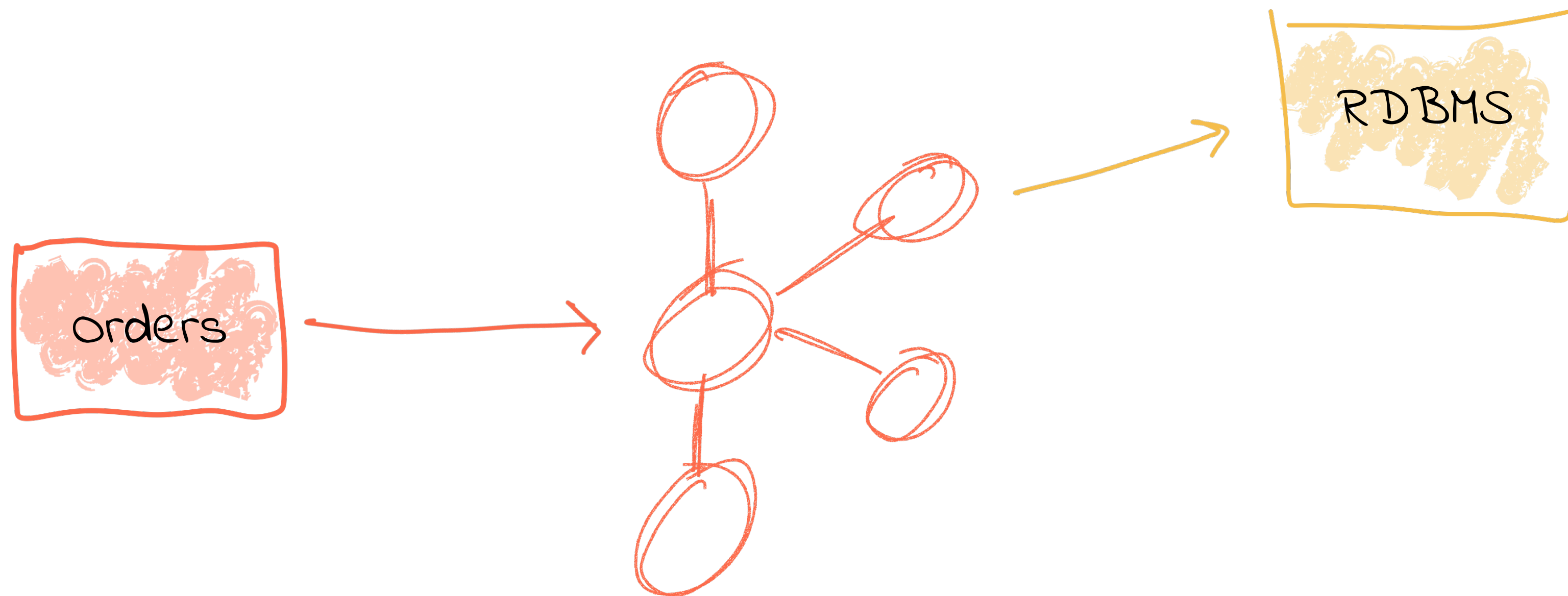


# Tight Coupling != Flexible



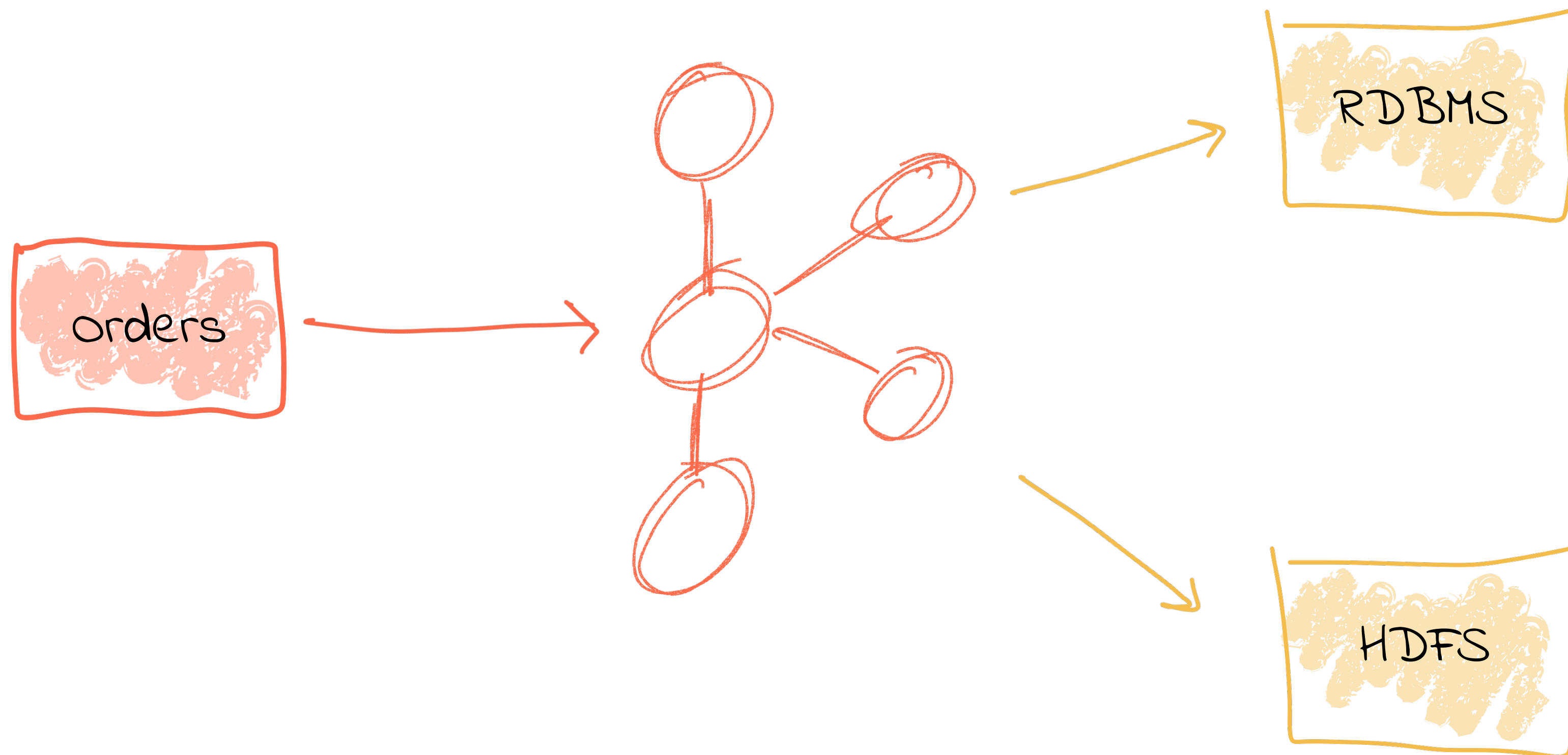


# Loose Coupling == Freedom to Evolve



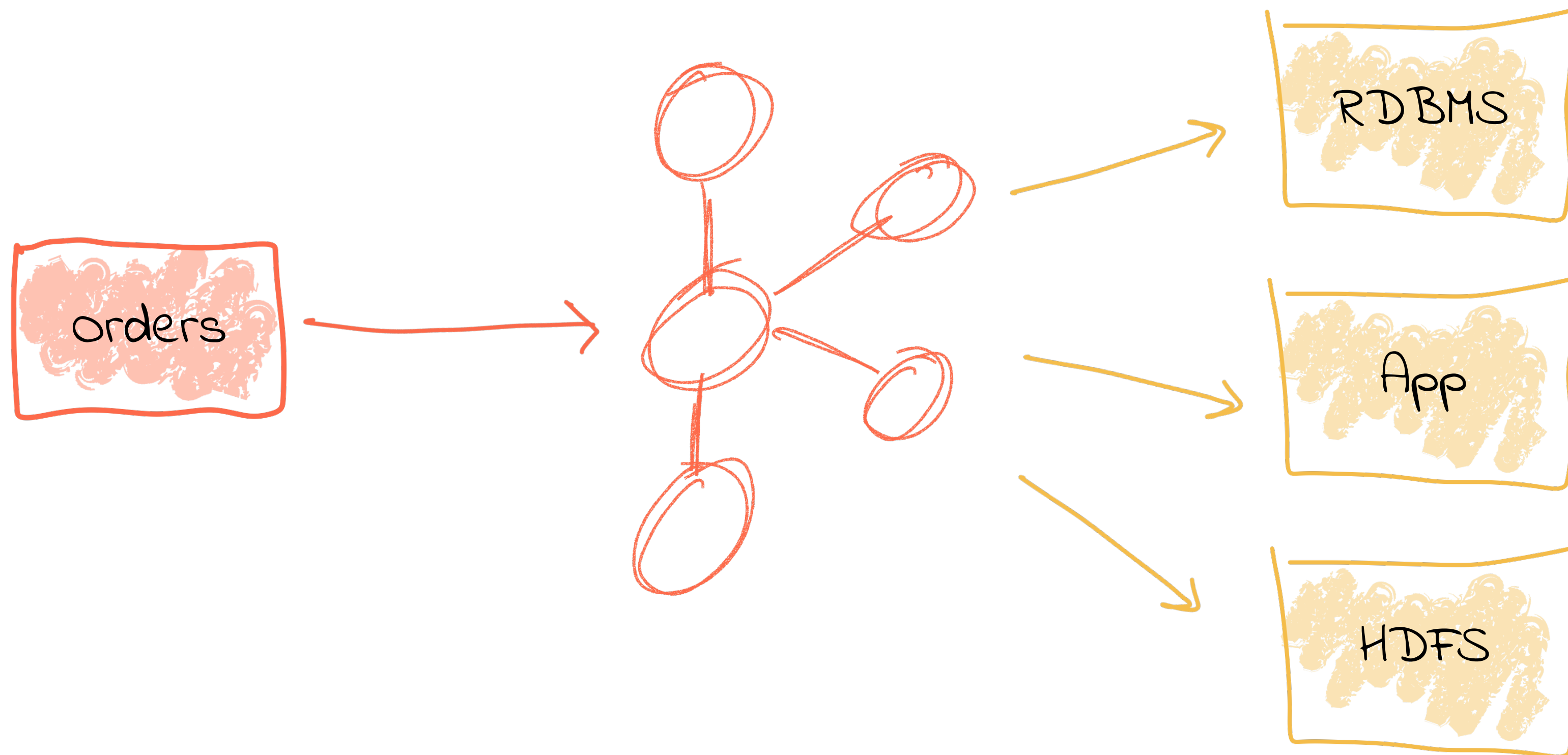


# Loose Coupling == Freedom to Evolve



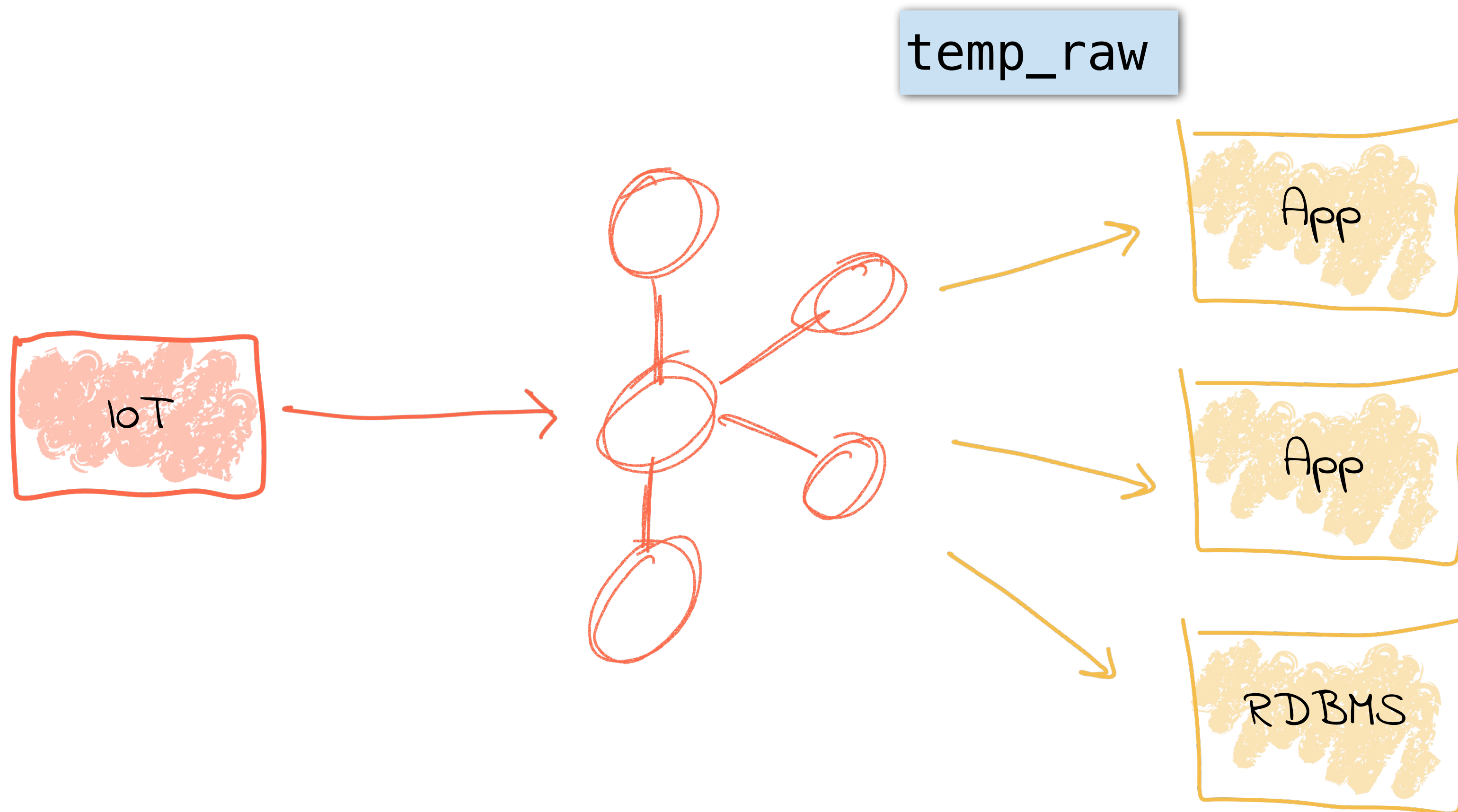


# Loose Coupling == Freedom to Evolve





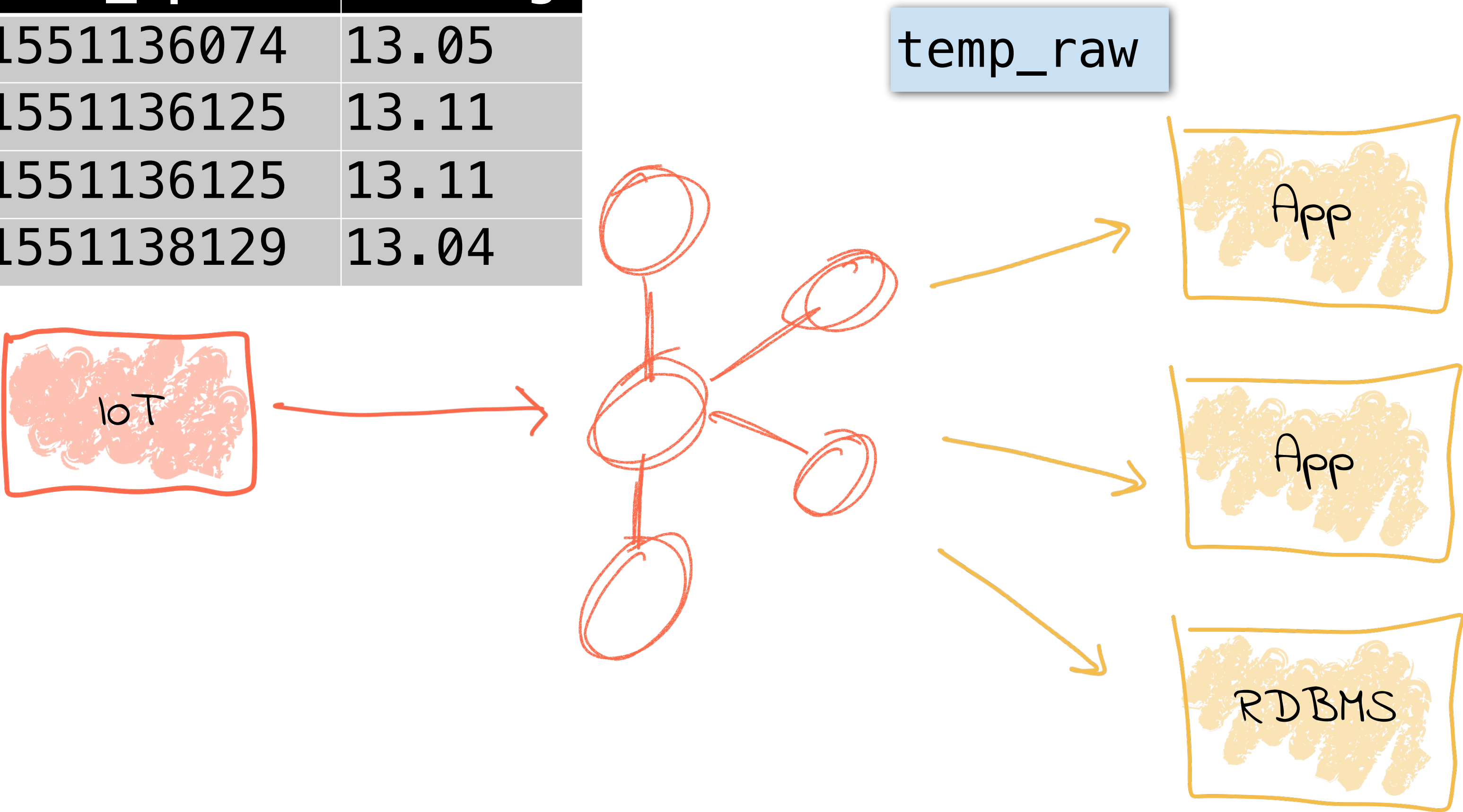
# Transform Once, Use Many: Data Cleansing





# Transform Once, Use Many: Data Cleansing

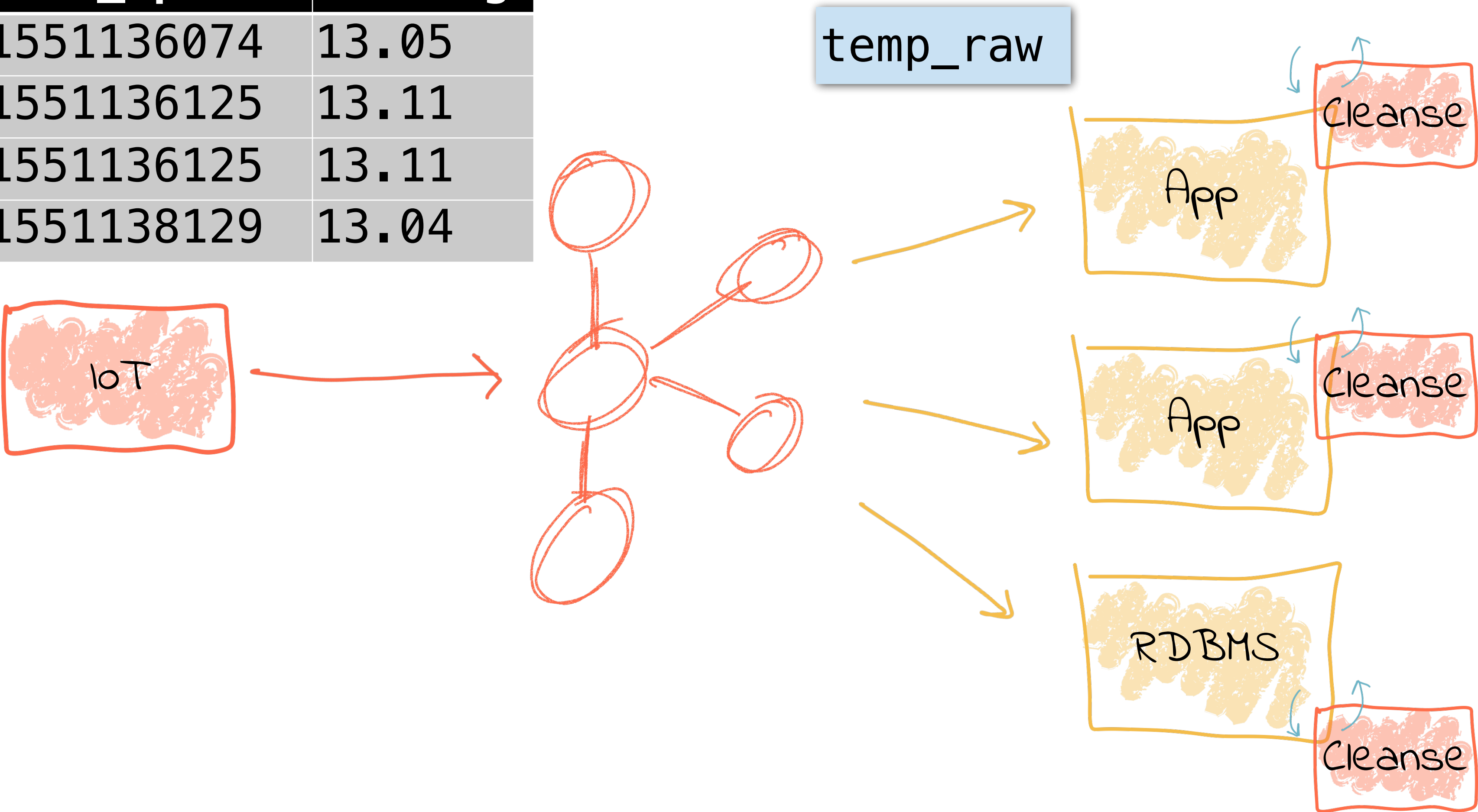
sensor_id	time_epoch	reading
42	1551136074	13.05
42	1551136125	13.11
	1551136125	13.11
42	1551138129	13.04





# Transform Once, Use Many: Data Cleansing

sensor_id	time_epoch	reading
42	1551136074	13.05
42	1551136125	13.11
	1551136125	13.11
42	1551138129	13.04



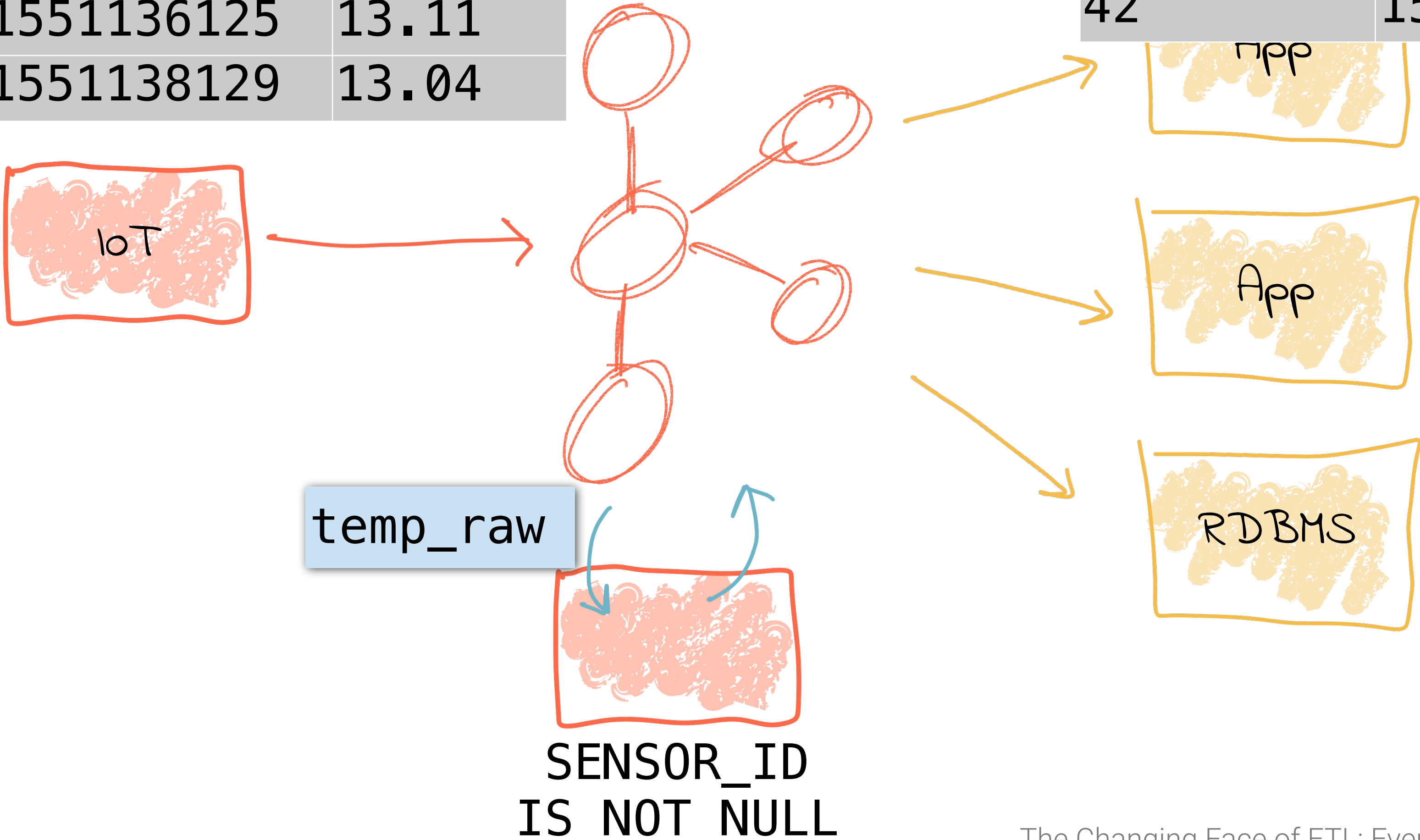


# Transform Once, Use Many: Data Cleansing

sensor_id	time_epoch	reading
42	1551136074	13.05
42	1551136125	13.11
	1551136125	13.11
42	1551138129	13.04

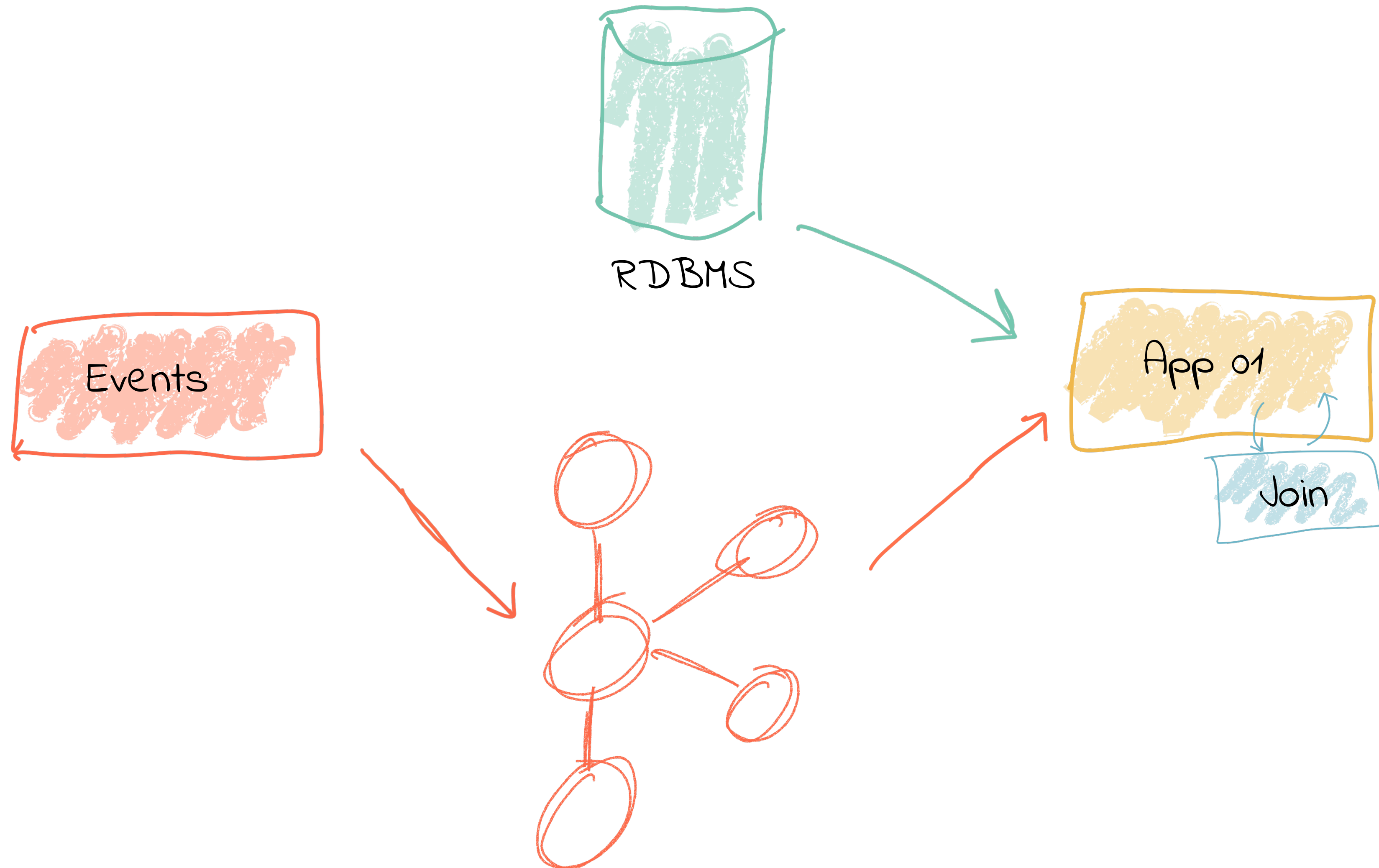
temp\_clean

sensor_id	time_epoch	reading
42	1551136074	13.05
42	1551136125	13.11
42	1551138129	13.04



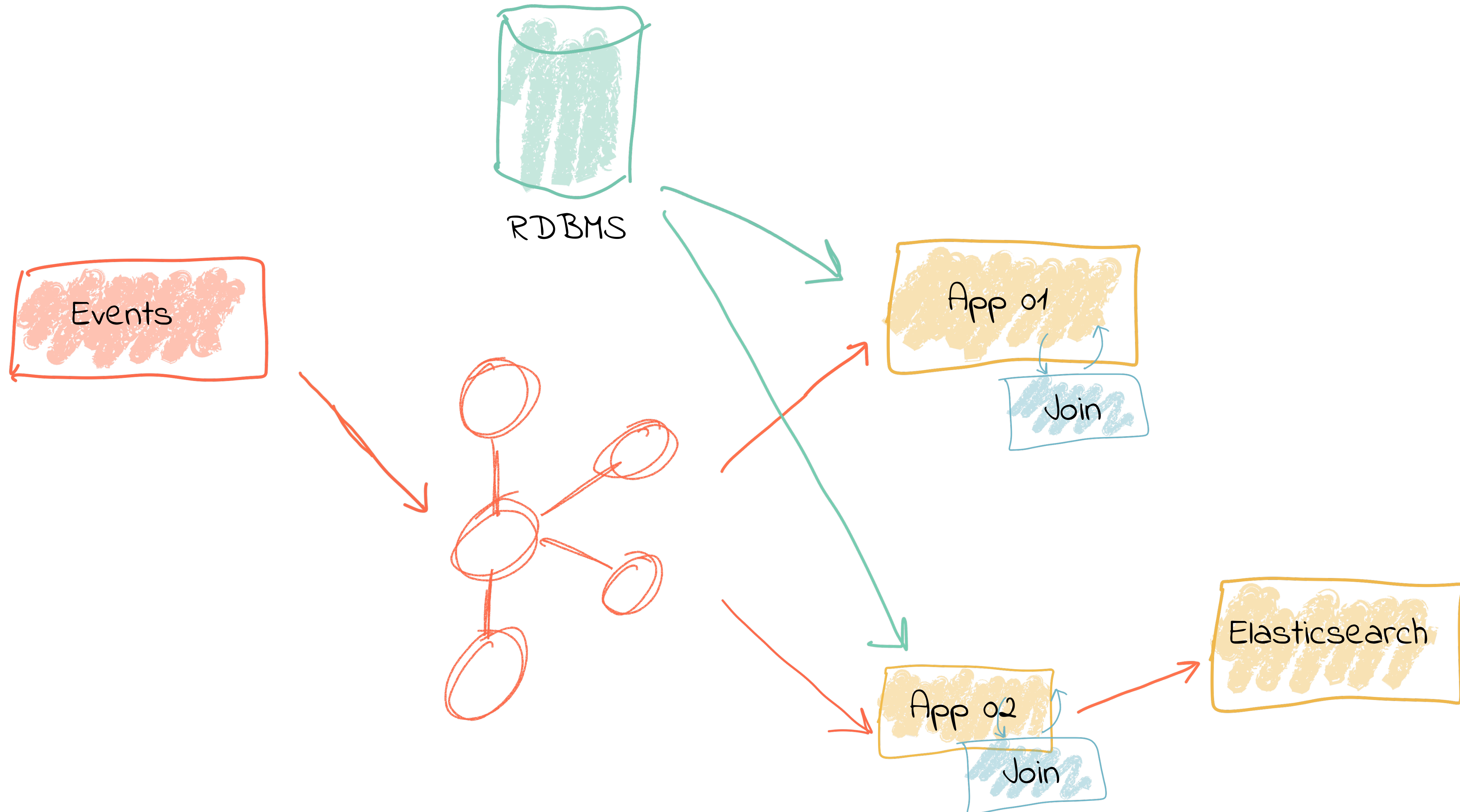


# Transform Once, Use Many: Data Enrichment



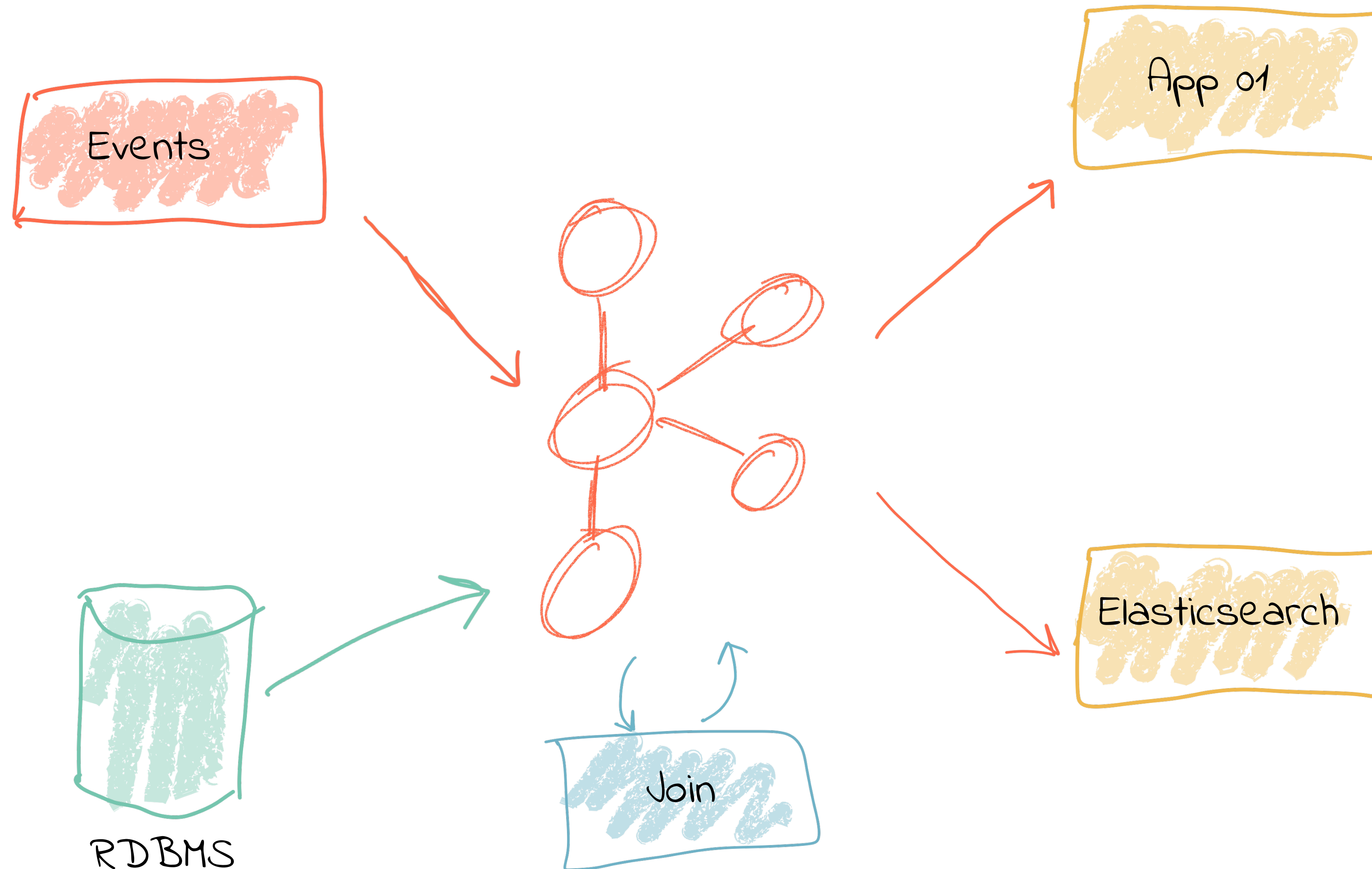


# Transform Once, Use Many: Data Enrichment



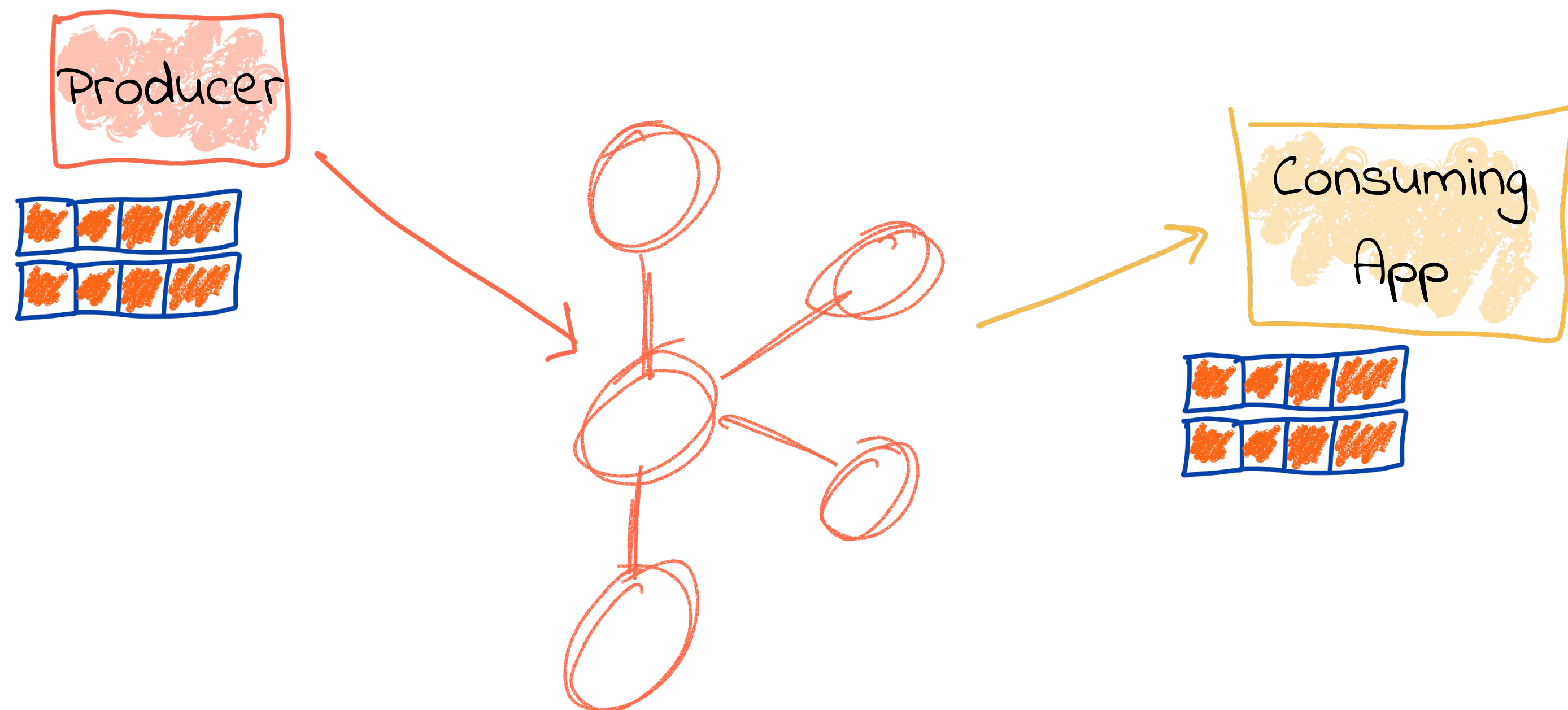


# Transform Once, Use Many: Data Enrichment



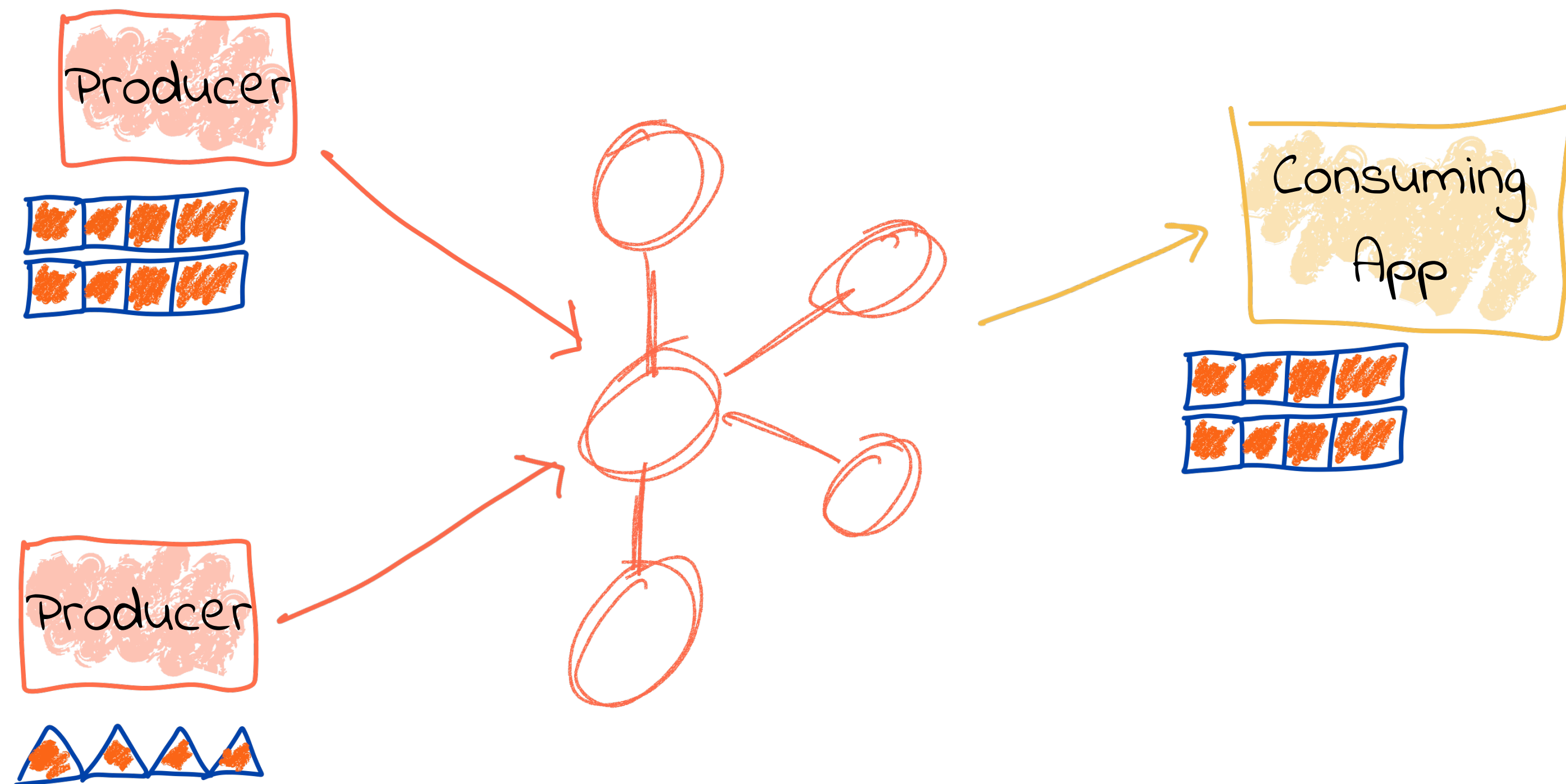


# Message Payload Compatibility



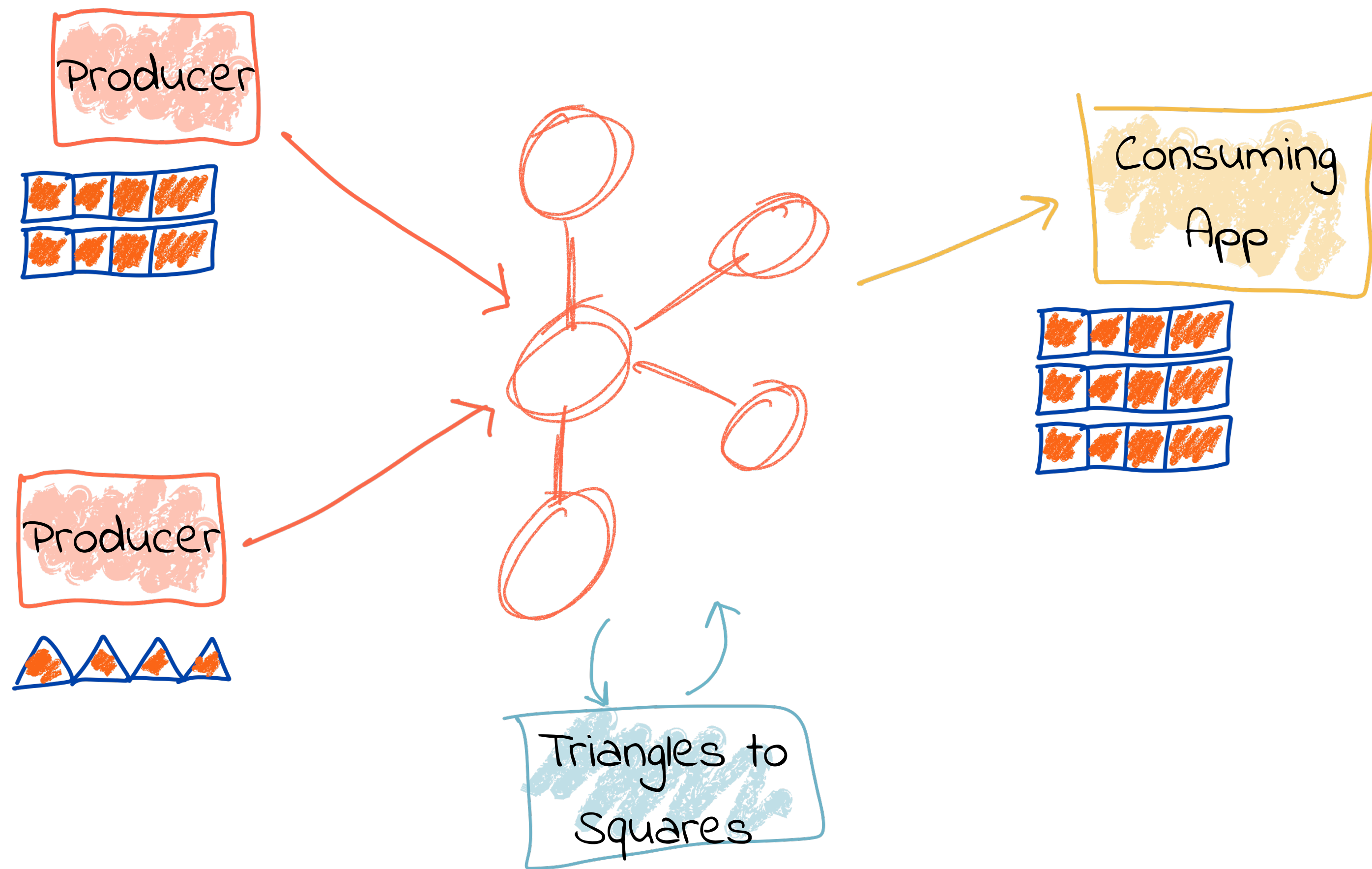


# Message Payload Compatibility



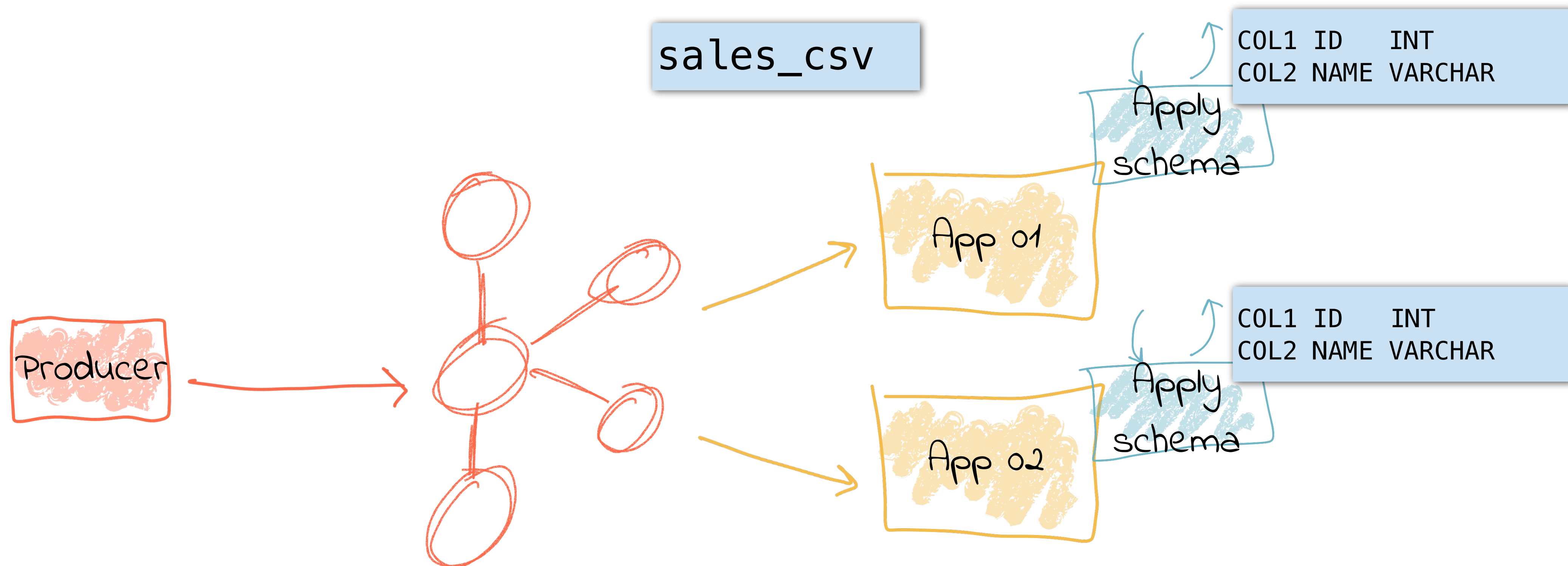


# Message Payload Compatibility



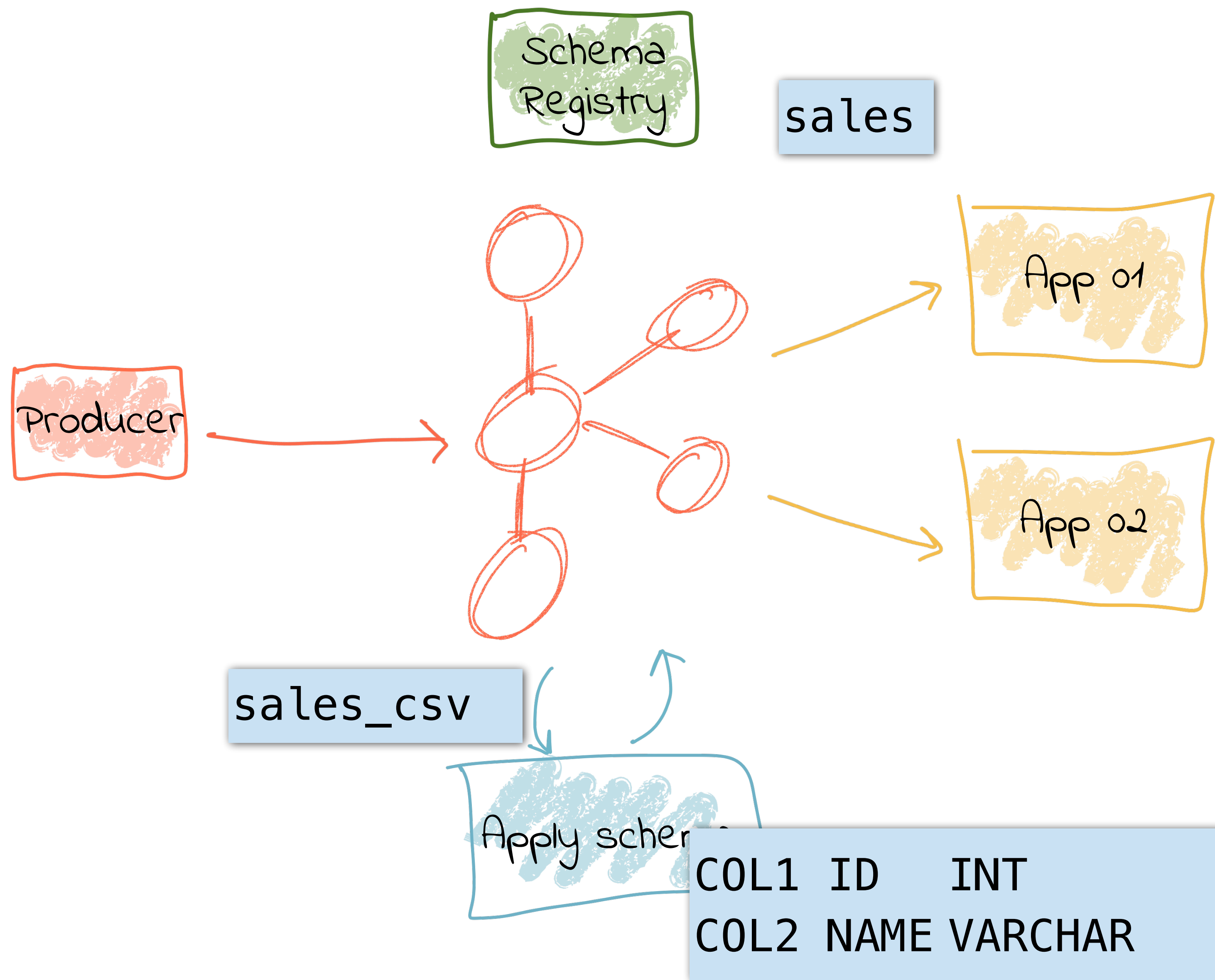


# Build Resilient Pipelines with Schemas





# Build Resilient Pipelines with Schemas

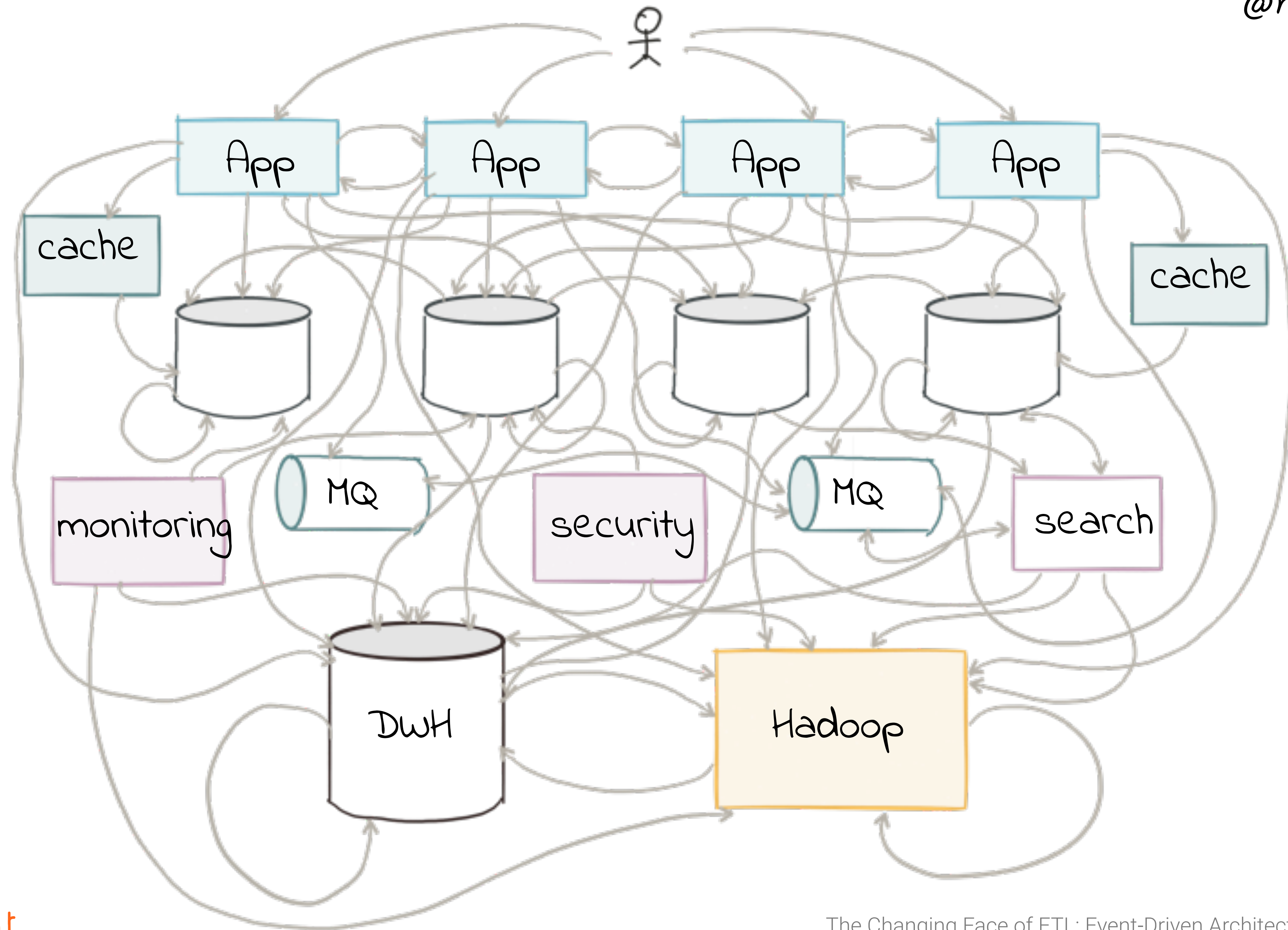




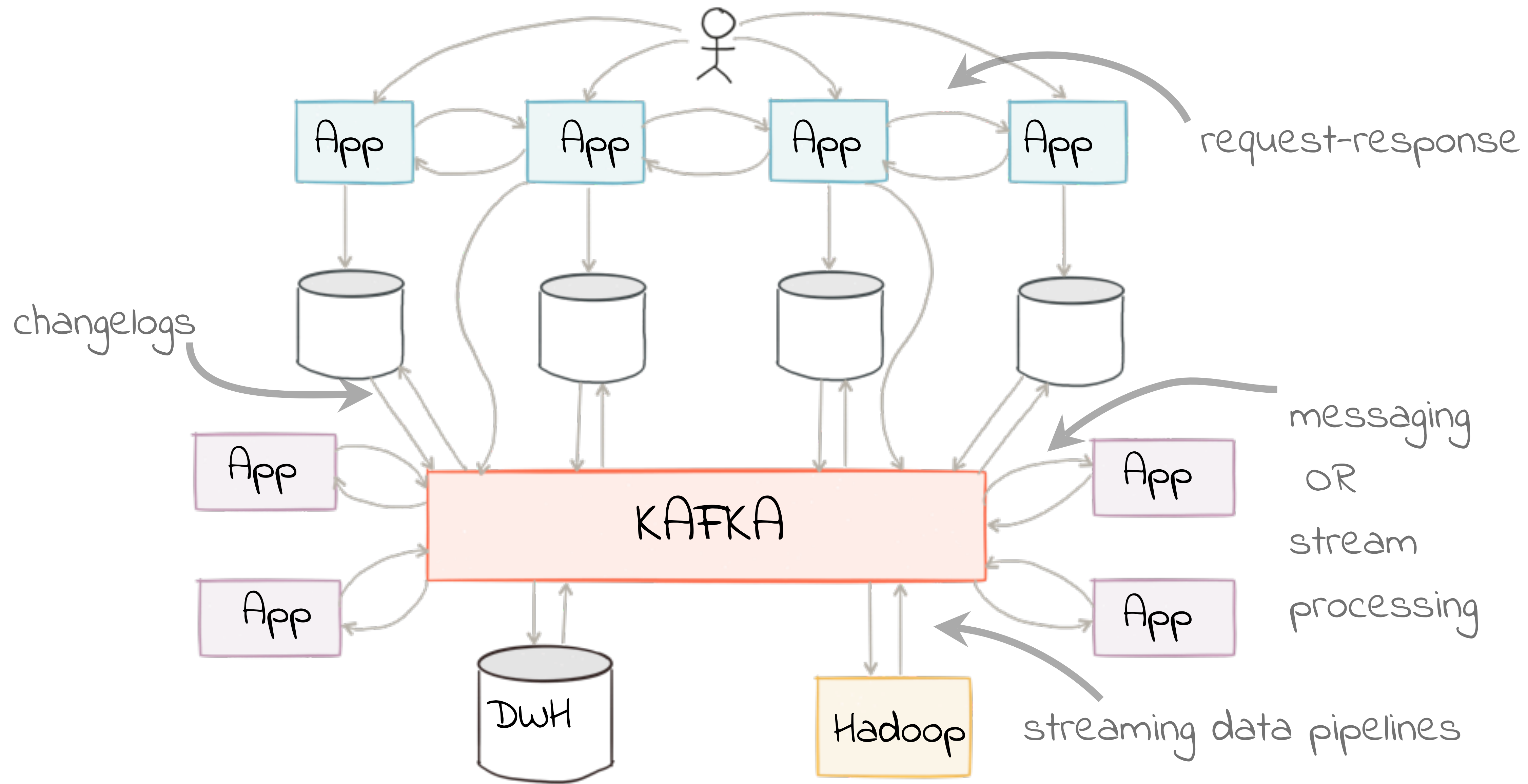
# Say NO to brittle pipelines













*Events model  
the real world*





# *Event streaming platform*

*Native stream  
processing*

*Data when  
you need it*

*Data persistence*

*Flexibility  
& scalability*





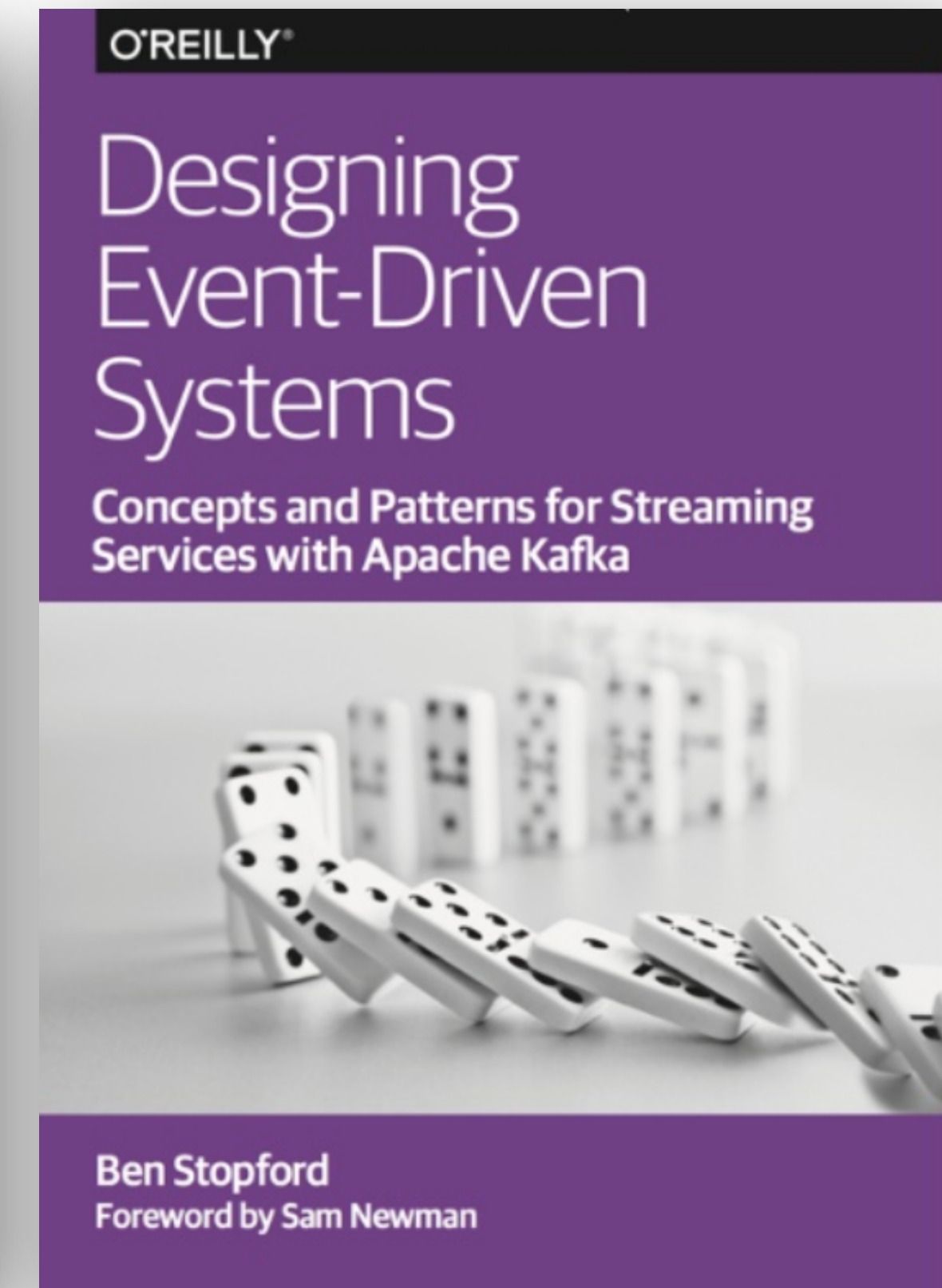
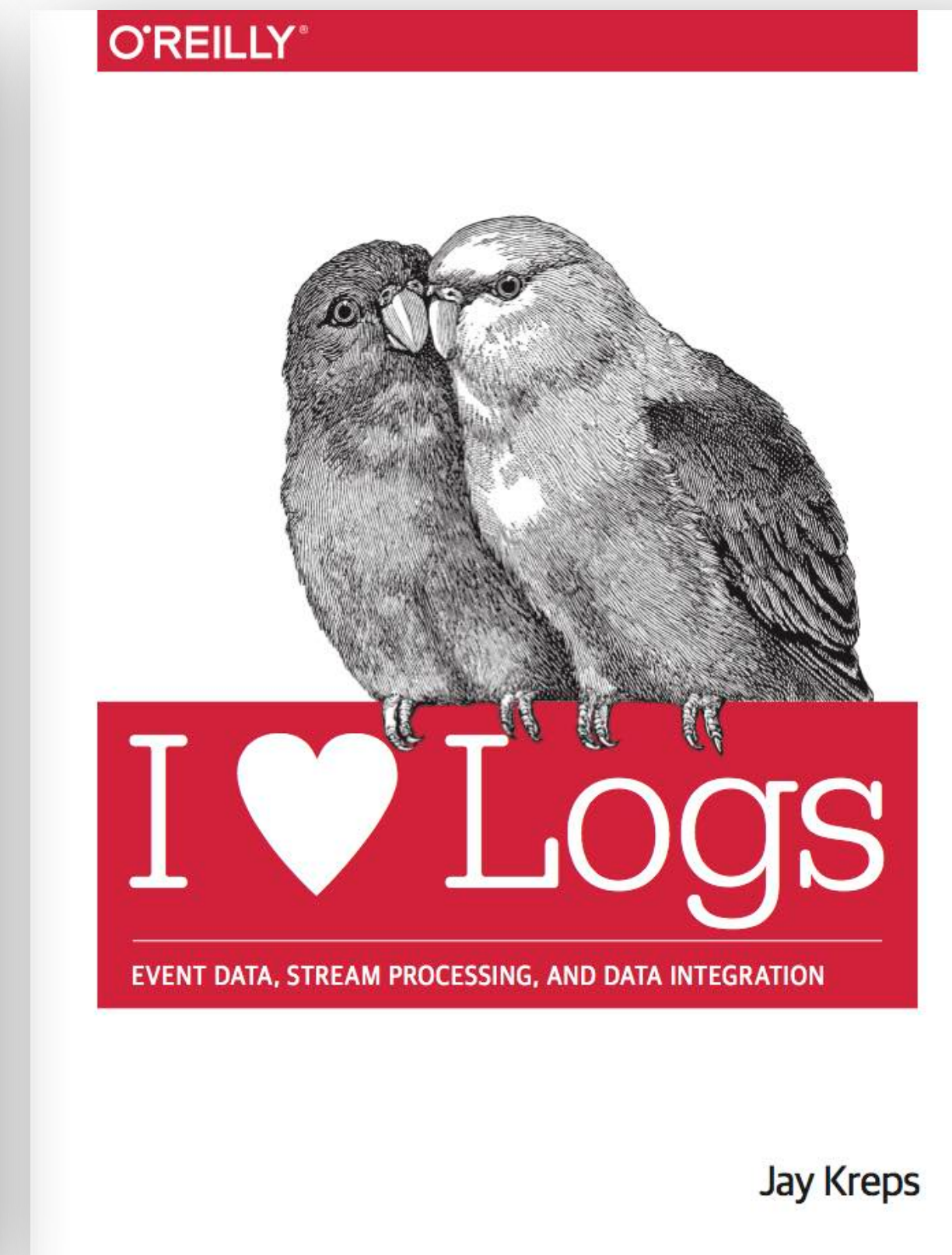
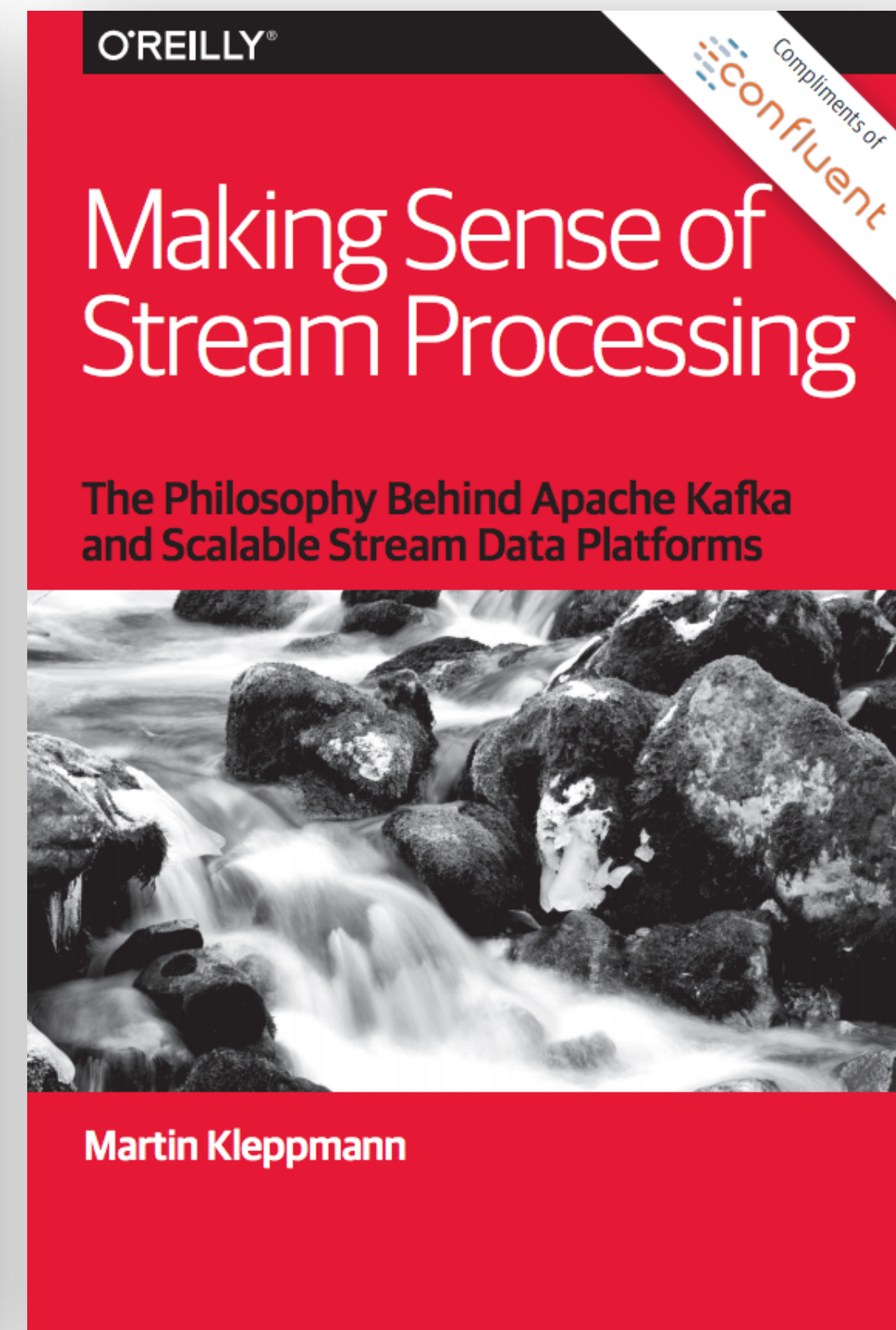
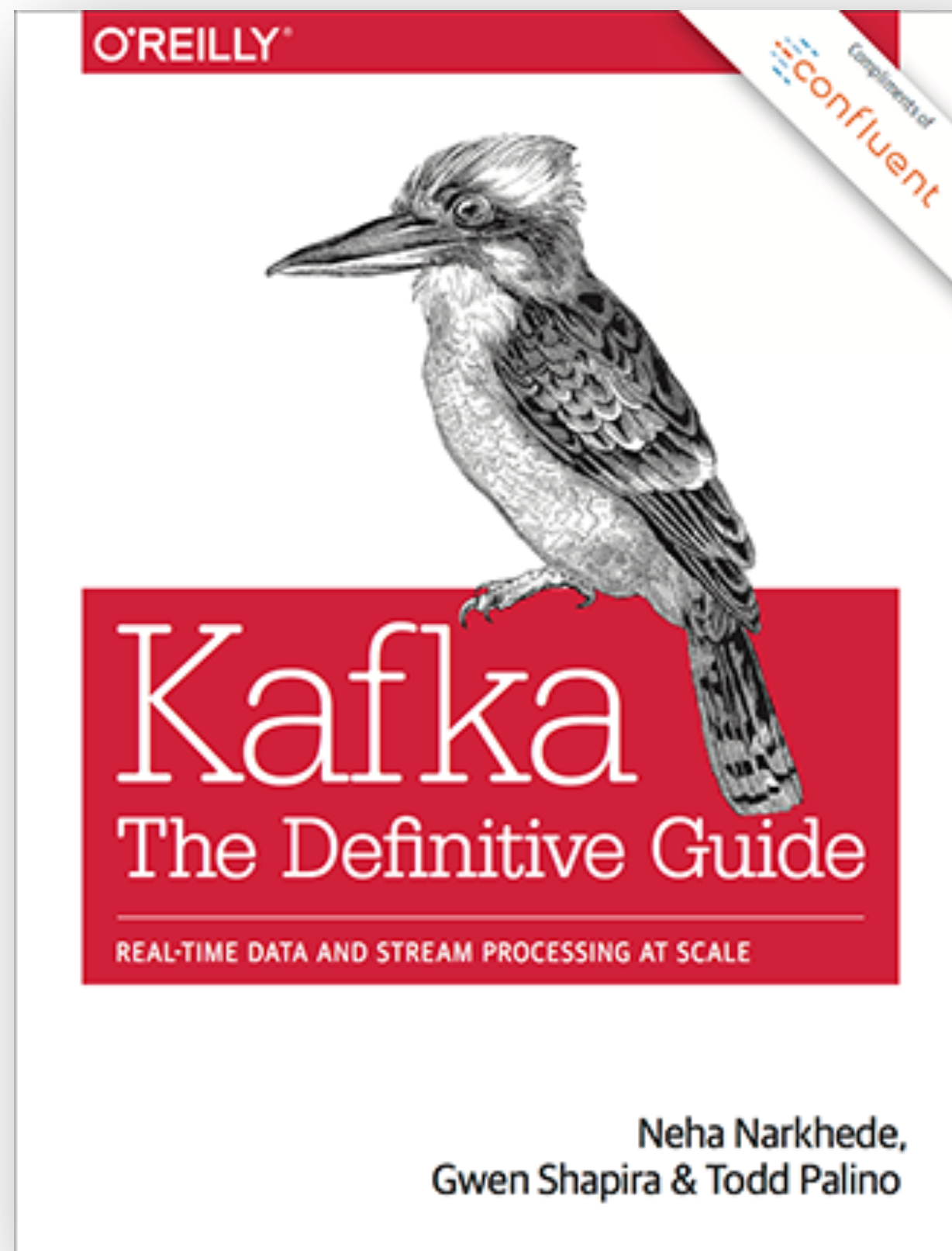


confluent cloud

# Fully Managed Kafka as a Service



# <https://rmoff.dev/oopmuc>





@rmoff

#00Pmuc

Photo by rmoff

<https://talks.rmoff.net>  
<http://cnfl.io/slack>