

@rmoff

#KafkaMeetup

From Zero to Hero with Kafka Connect

a.k.a.

A practical guide to becoming 133t with Kafka Connect

@rmoff

#KafkaMeetup

From Zero to Hero with Kafka Connect

a.k.a.

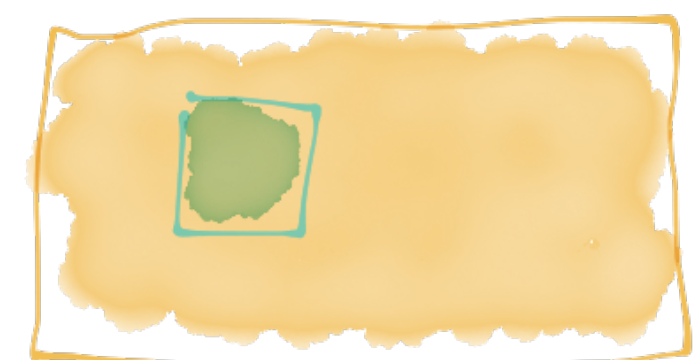
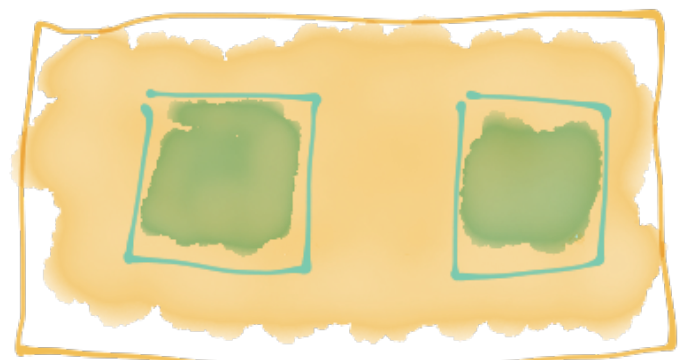
A practical guide to becoming 133t with Kafka Connect

What is
Kafka
Connect?

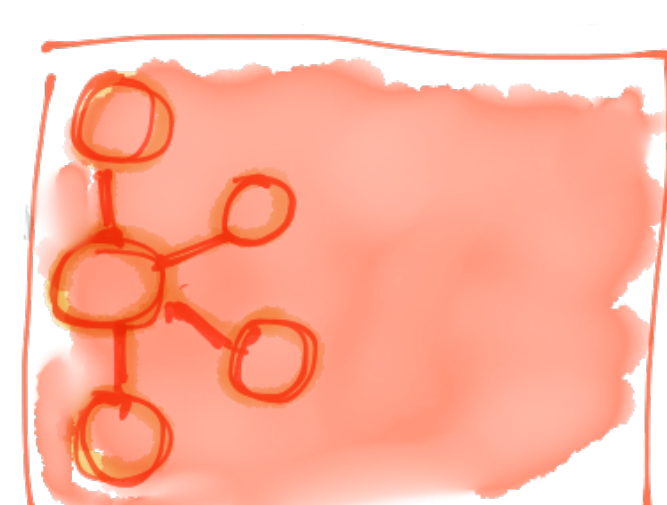
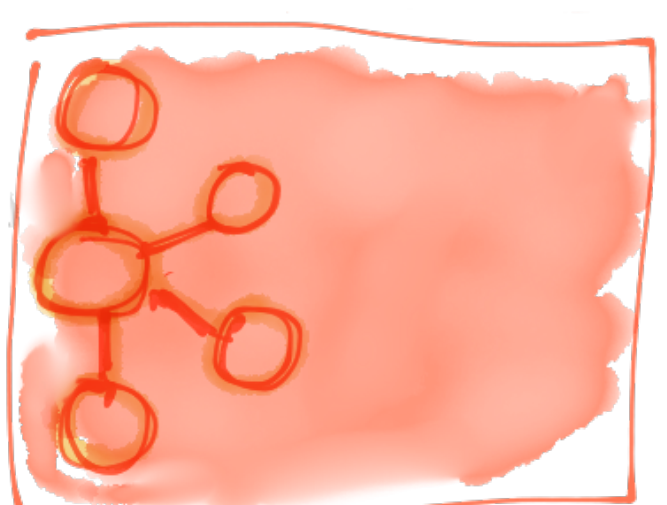
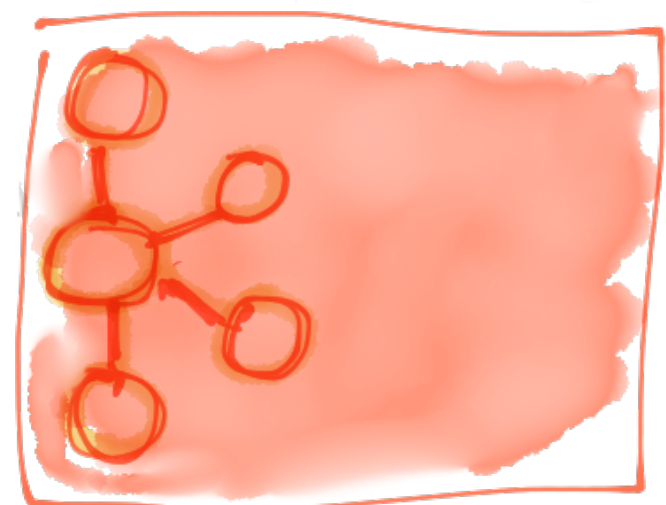
Streaming Integration with Kafka Connect



Sources



Kafka Connect



Kafka Brokers

Streaming Integration with Kafka Connect

Sinks



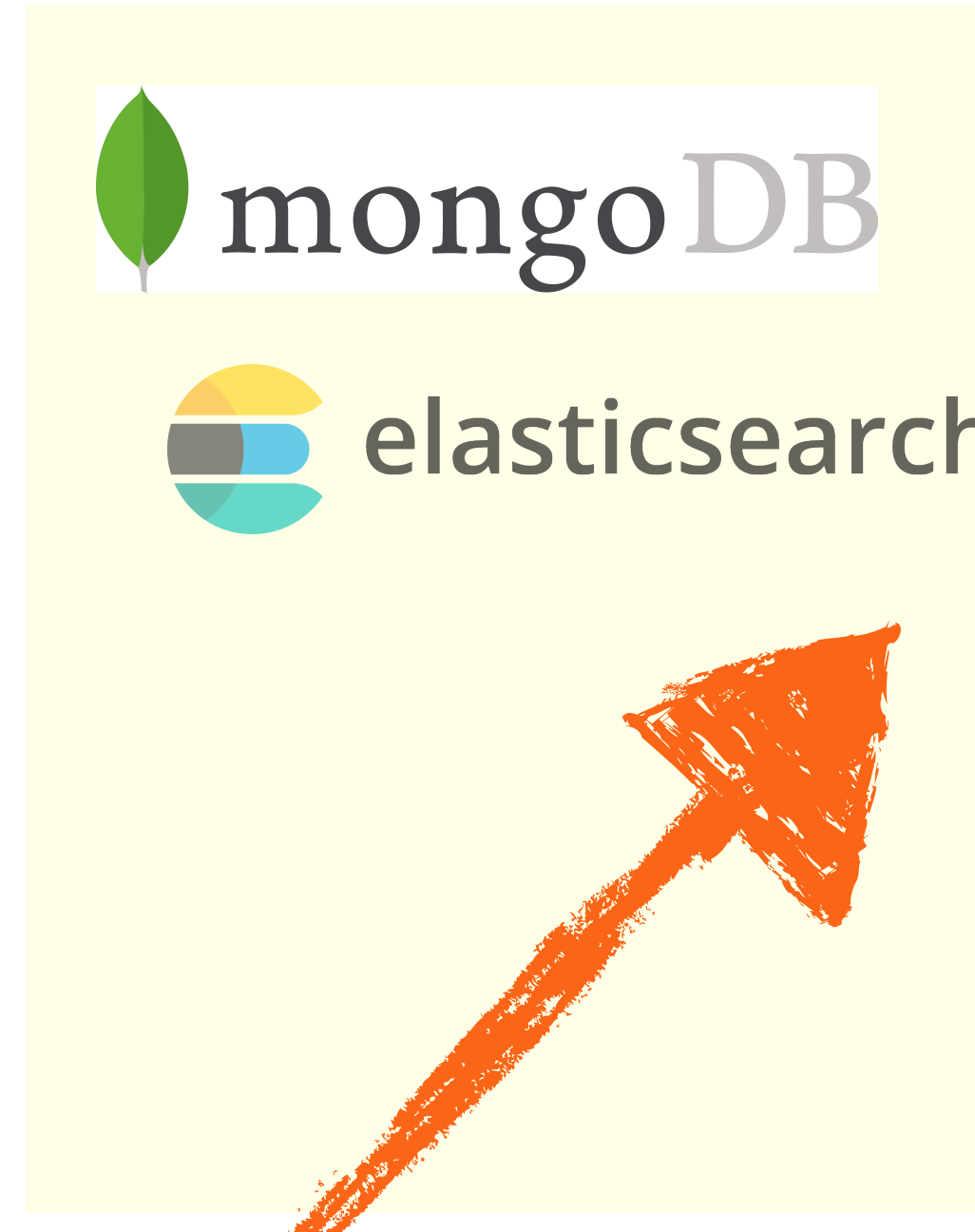
Kafka Connect

Kafka Brokers

Streaming Integration with Kafka Connect



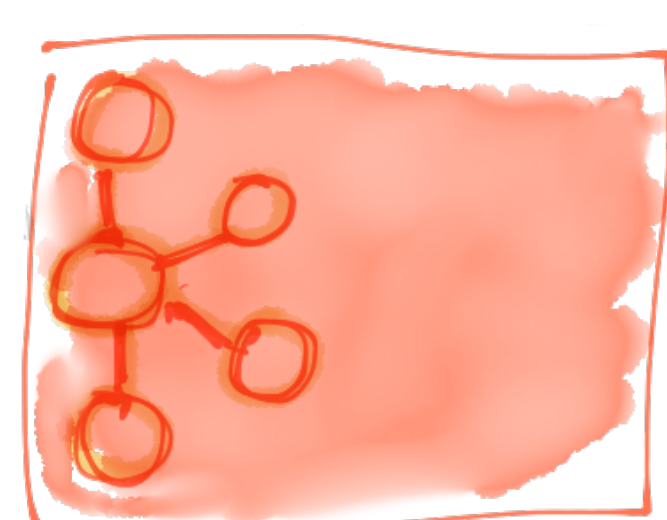
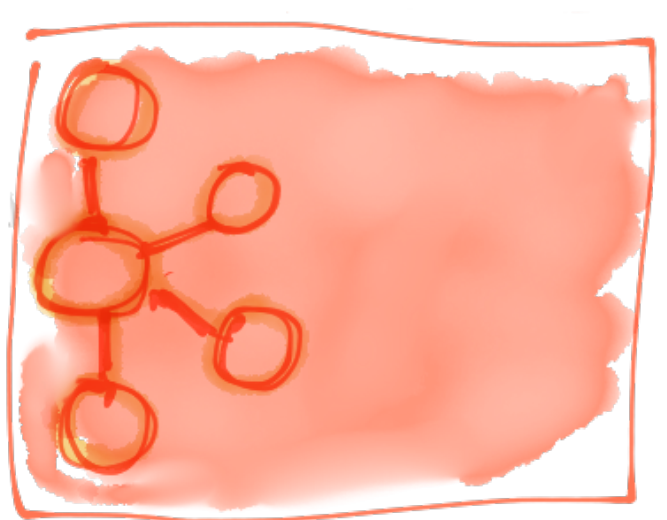
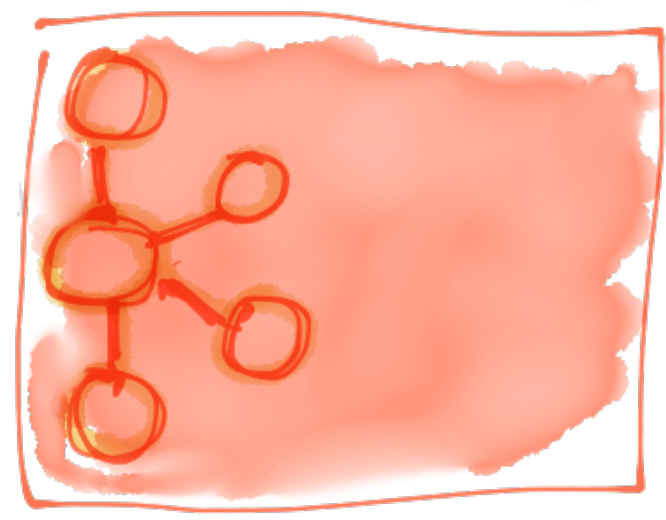
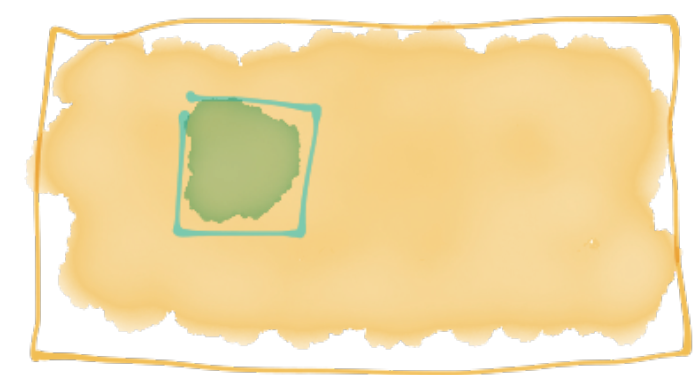
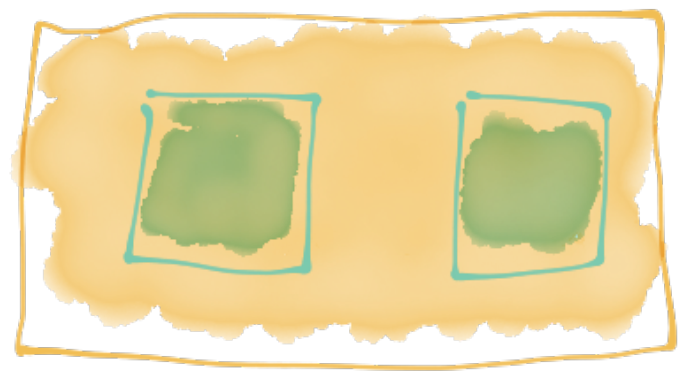
A yellow rectangular box containing various logos for data sources. The logos include: a purple square with white lines, a document icon with 'JSON', 'MQTT.ORG', 'MySQL', 'syslog', 'http://', 'CSV', 'salesforce', 'ORACLE', and 'Microsoft SQL Server'. A blue elephant icon is also present. An orange arrow points from this box towards the center.



A yellow rectangular box containing logos for data sinks. The logos include: 'mongoDB', 'elasticsearch', and 'Java JDBC'. An orange arrow points from this box towards the center.



A yellow rectangular box containing logos for data sinks. The logos include: 'Amazon S3', 'snowflake', 'IBM MQ', 'Google BigQuery', 'ORACLE', 'http://', 'salesforce', 'neo4j', 'splunk', 'hadoop HDFS', and 'MQTT.ORG'. A blue elephant icon is also present.



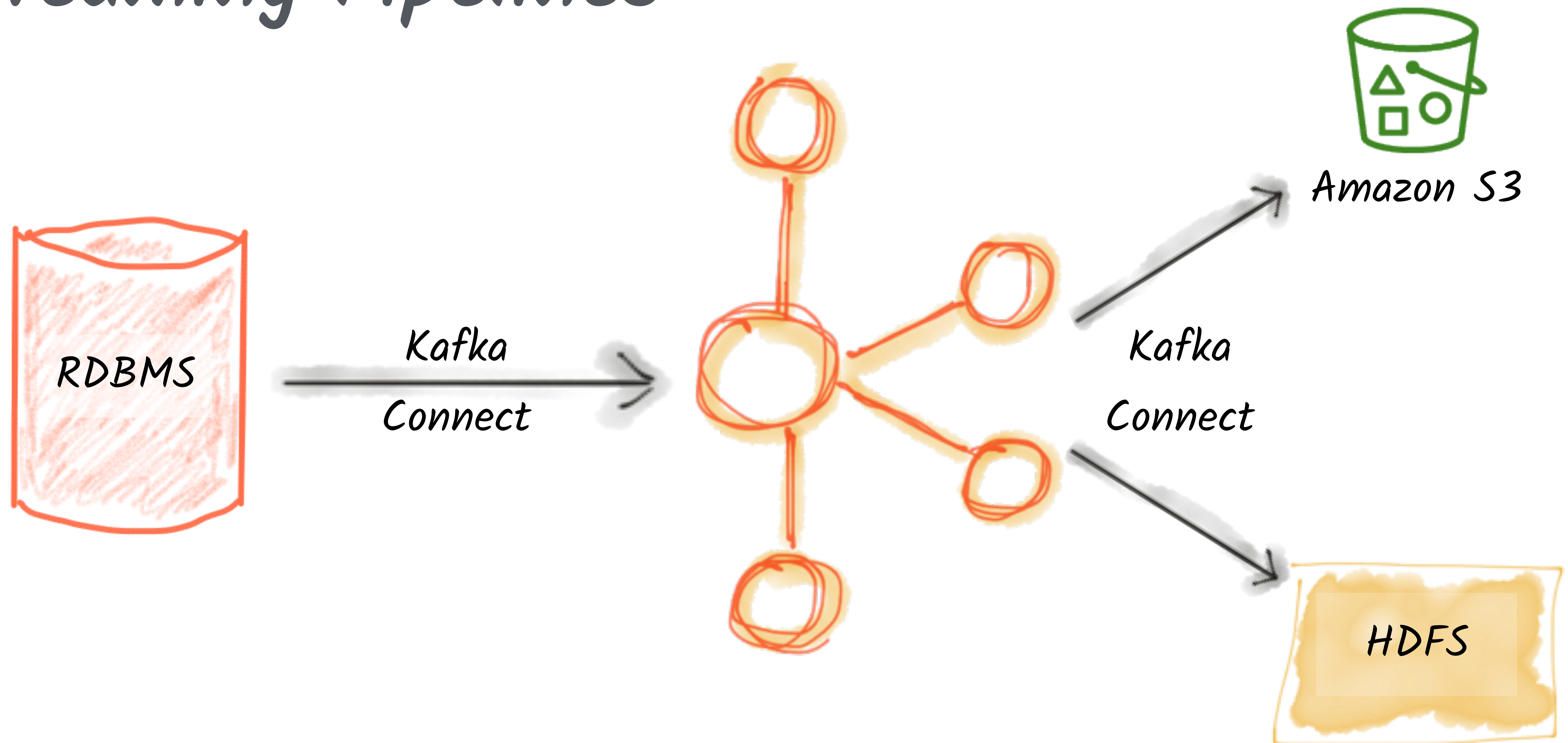
Kafka Connect

Kafka Brokers

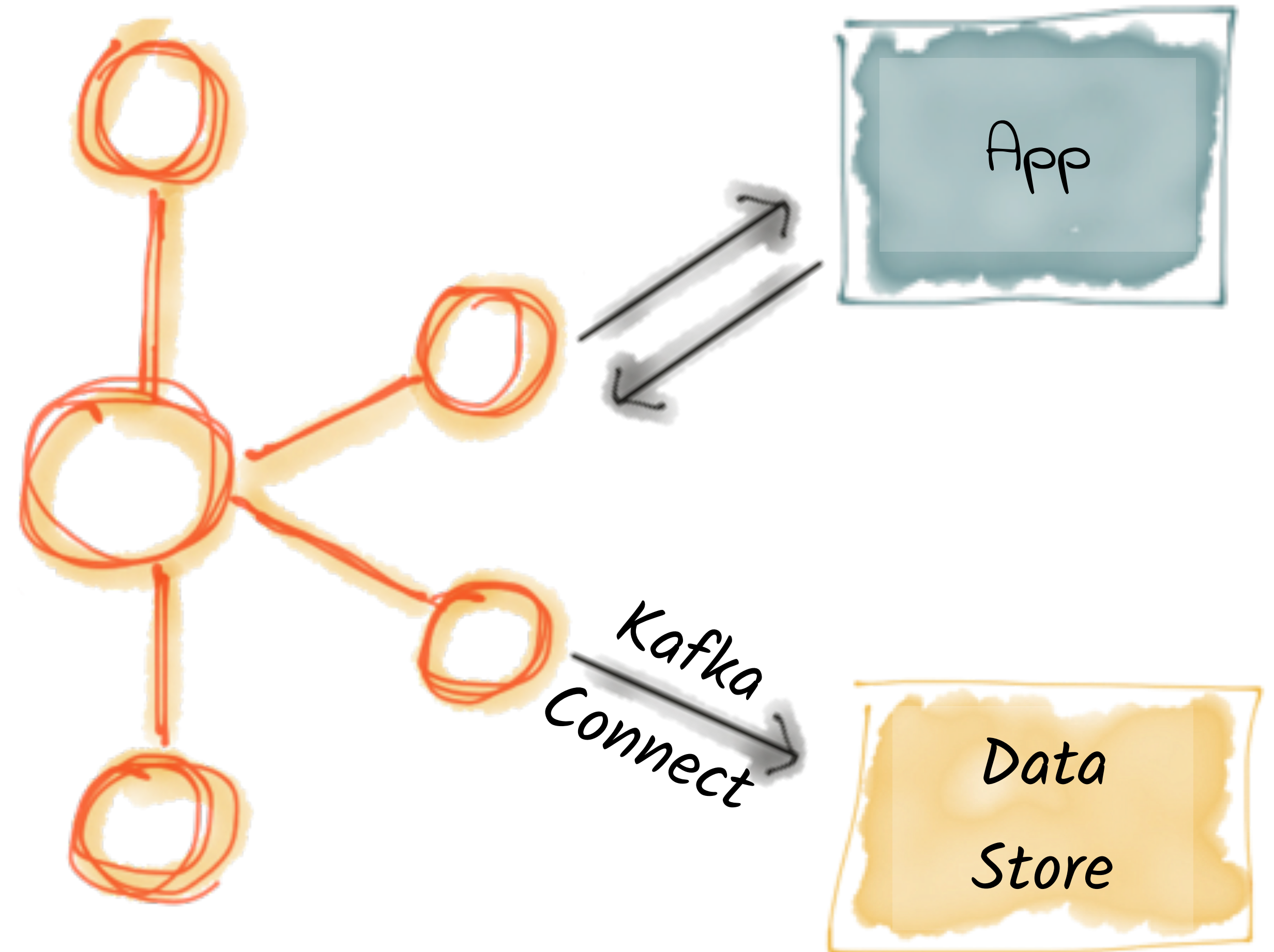
Look Ma, No Code!

```
{  
  "connector.class":  
    "io.confluent.connect.jdbc.JdbcSourceConnector",  
  "connection.url":  
    "jdbc:mysql://asgard:3306/demo",  
  "table.whitelist":  
    "sales,orders,customers"  
}
```

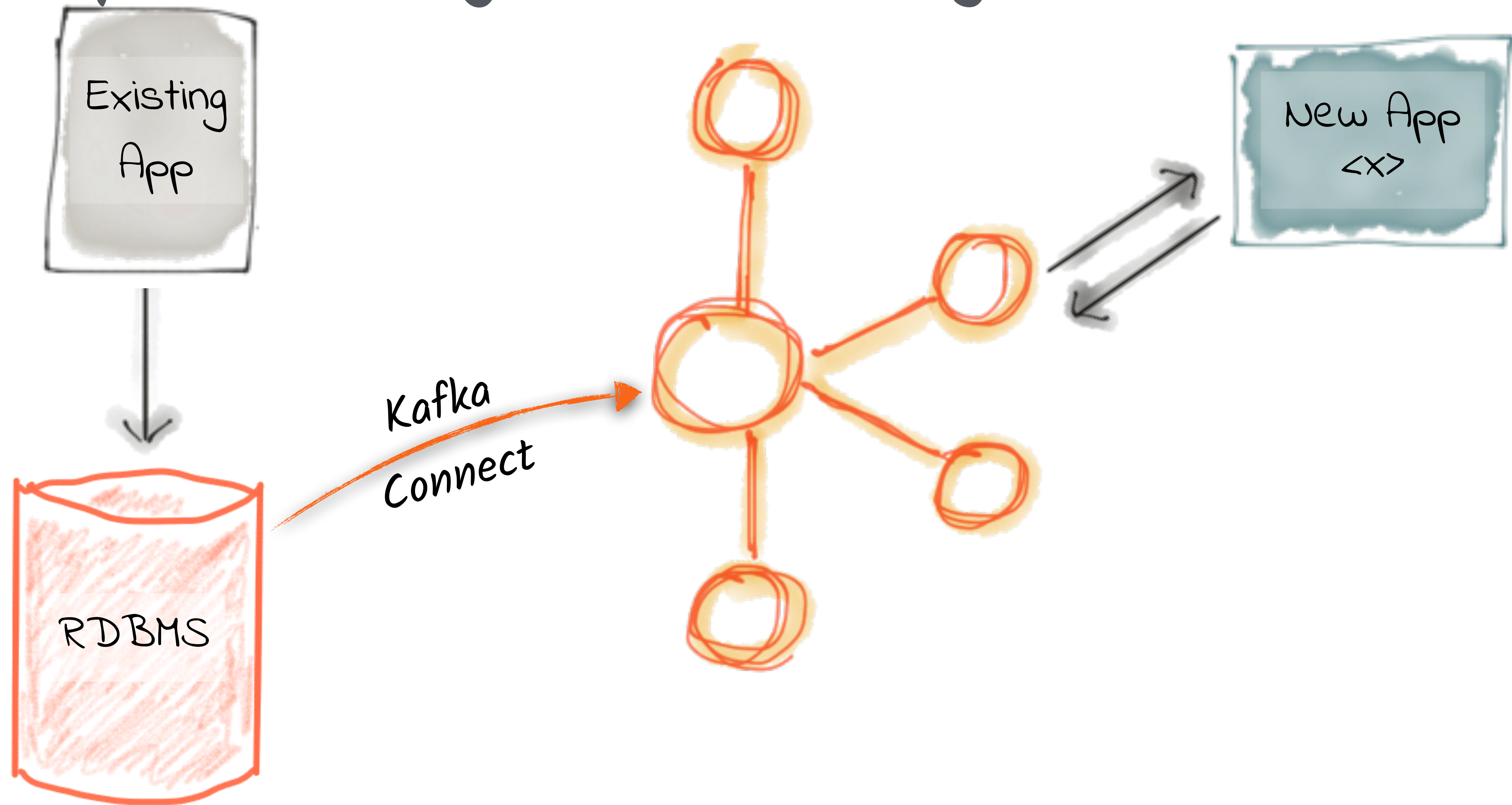

Streaming Pipelines



Writing to data stores from Kafka



Evolve processing from old systems to new



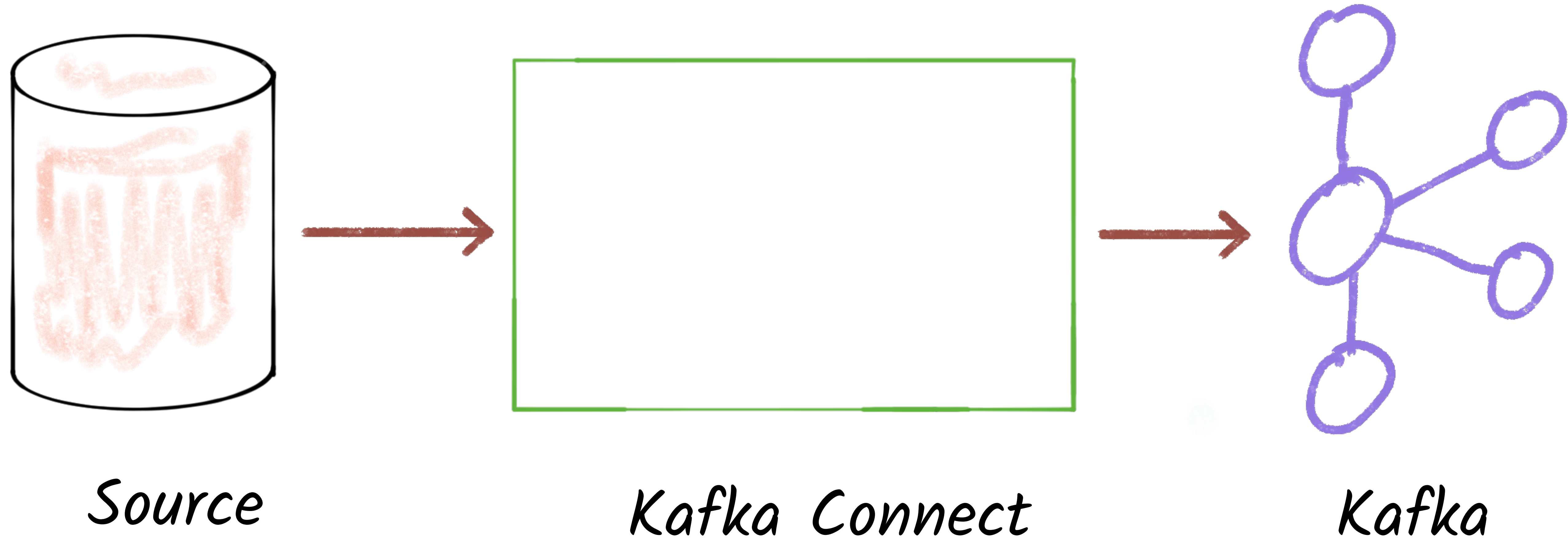
Demo

<http://rmoff.dev/kafka-connect-code>

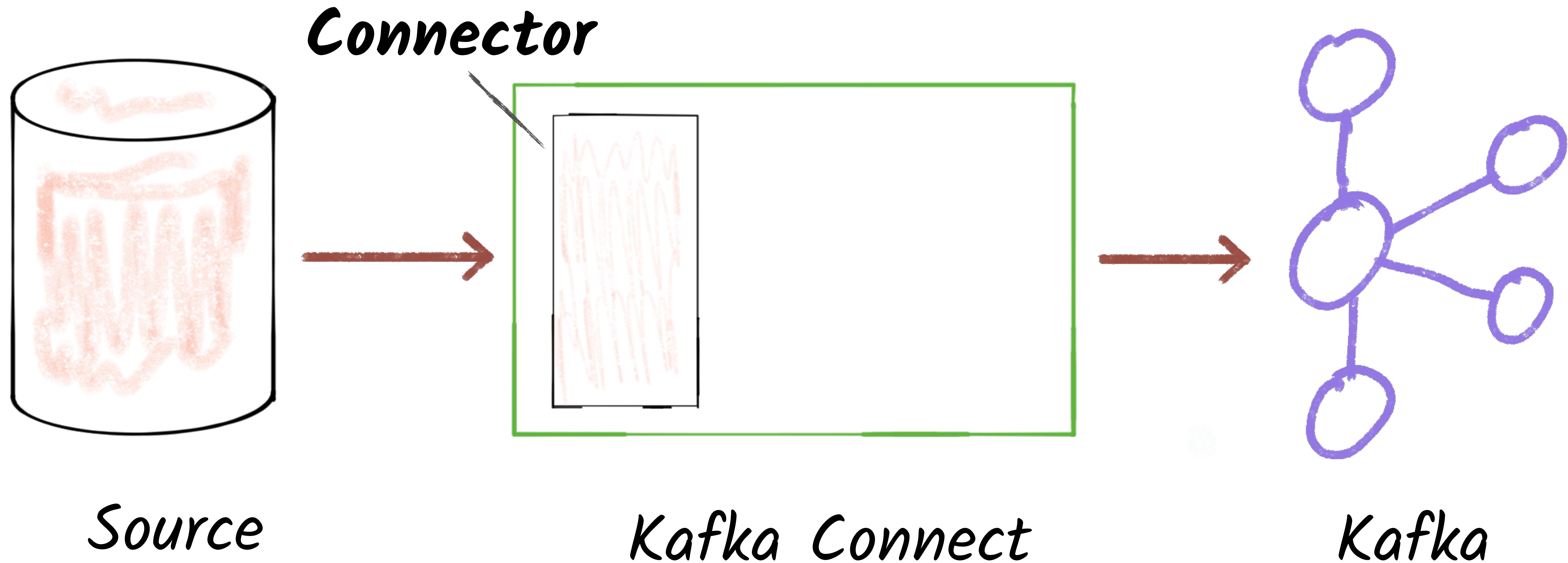
Configuring Kafka Connect

Inside the API - connectors, transforms, converters

Kafka Connect basics



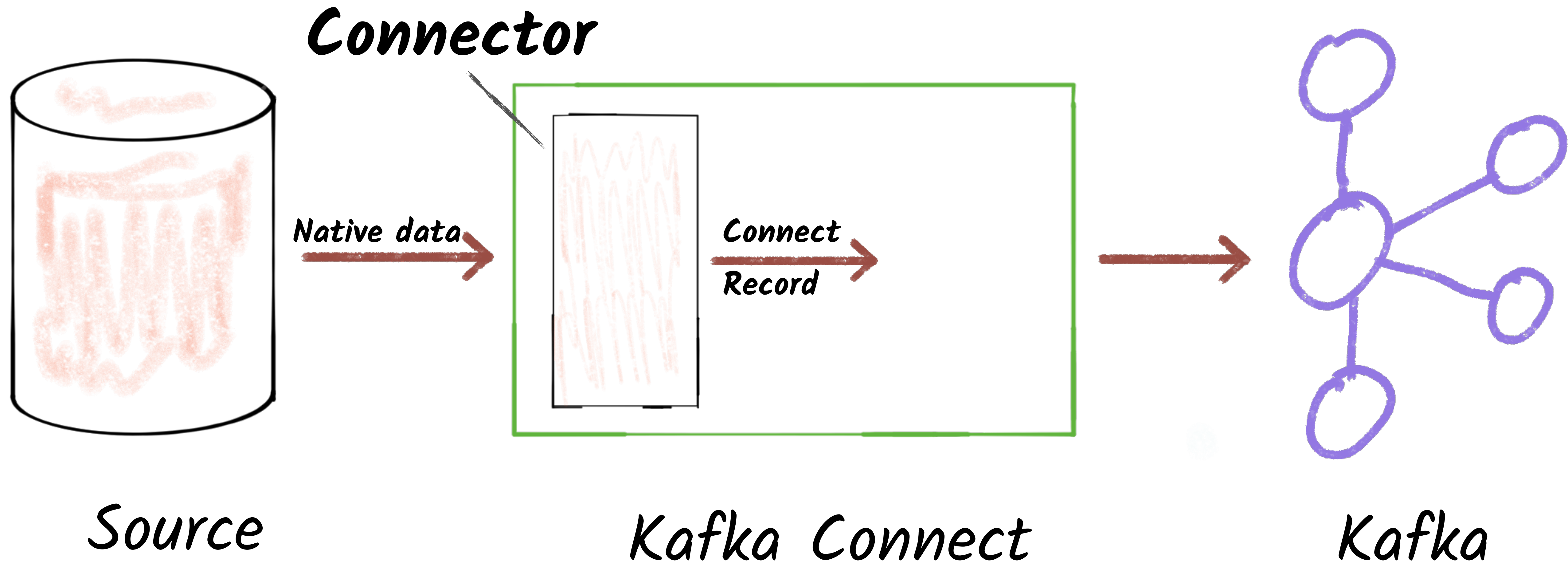
Connectors



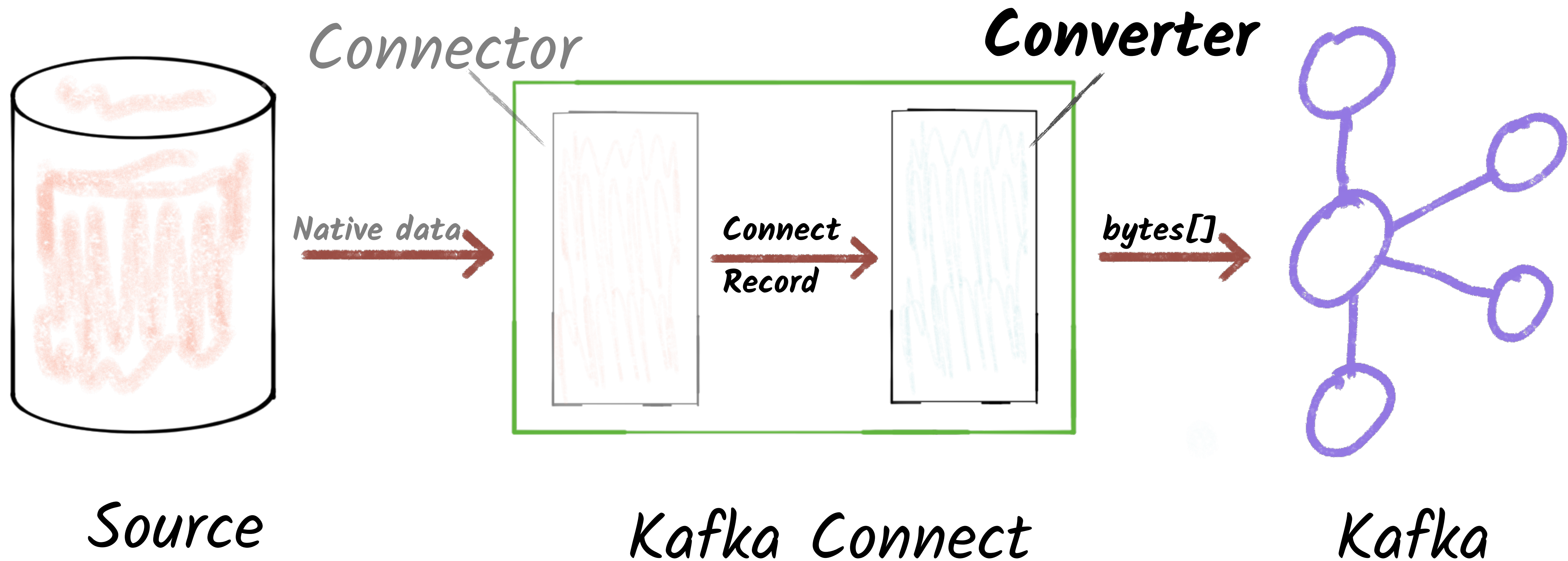
Connectors

```
"config": {  
  [...]  
  "connector.class": "io.confluent.connect.jdbc.JdbcSinkConnector"  
  "connection.url": "jdbc:postgresql://postgres:5432/",  
  "topics": "asgard.demo.orders",  
}
```


Connectors



Converters



Serialisation & Schemas

Avro

-> Confluent
Schema Registry

Protobuf

JSON

CSV



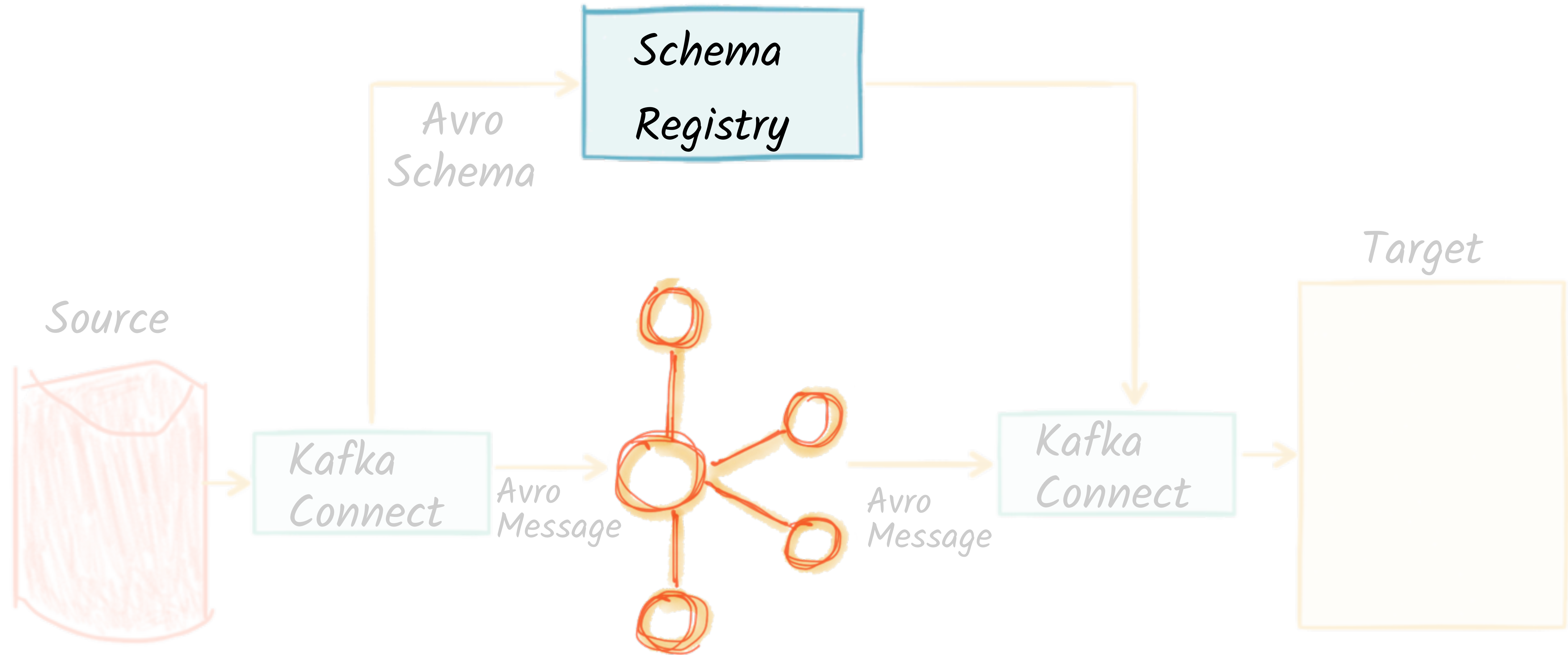
Gwen (Chen) Shapira
@gwenshap

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

The Confluent Schema Registry



Converters

key.converter=`io.confluent.connect.avro.AvroConverter`

key.converter.schema.registry.url=`http://localhost:8081`

value.converter=`io.confluent.connect.avro.AvroConverter`

value.converter.schema.registry.url=`http://localhost:8081`

Set as a global default per-worker; optionally can be overridden per-connector

What about internal converters?

value.converter=org.apache.kafka.connect.json.JsonConverter

internal.value.converter=org.apache.kafka.connect.json.JsonConverter

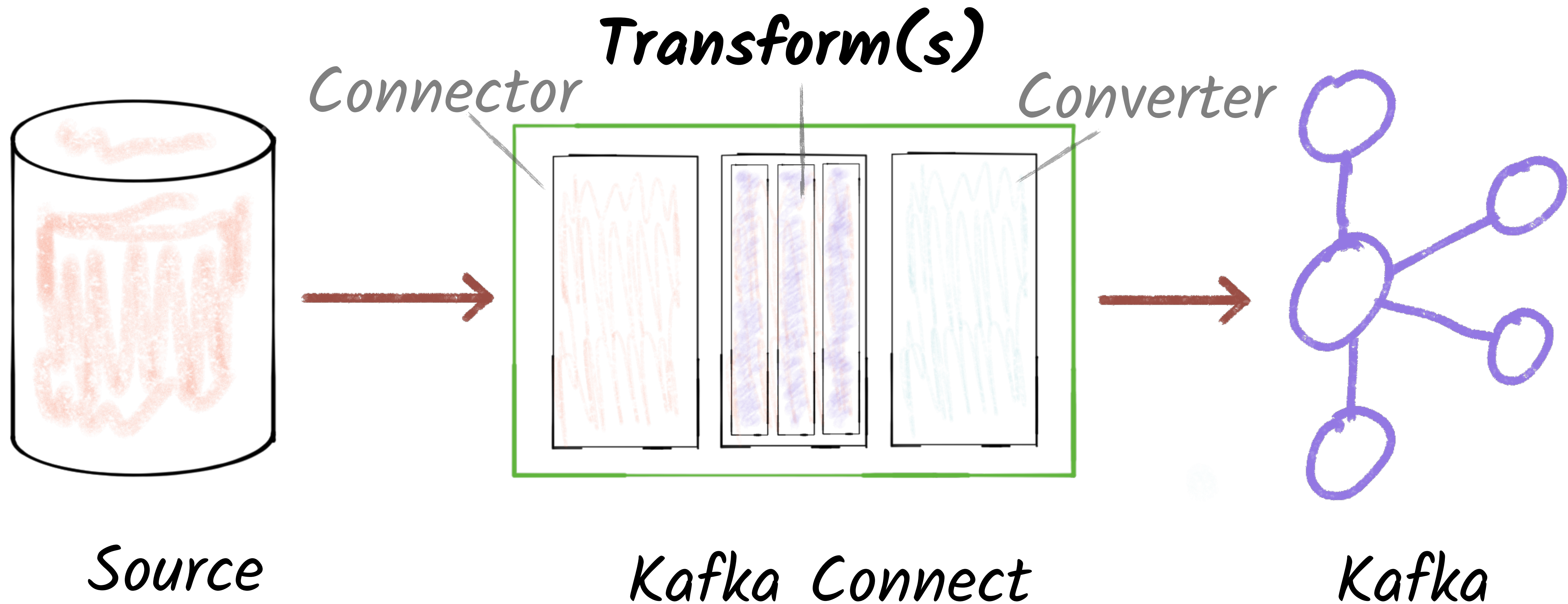
key.internal.value.converter=org.apache.kafka.connect.json.JsonConverter

value.internal.value.converter=org.apache.kafka.connect.json.JsonConverter

key.internal.value.converter.bork.bork.bork=org.apache.kafka.connect.json.JsonConverter

key.internal.value.please.just.work.converter=org.apache.kafka.connect.json.JsonConverter

Single Message Transforms



Single Message Transforms

```
"config": {
```

```
  [...]
```

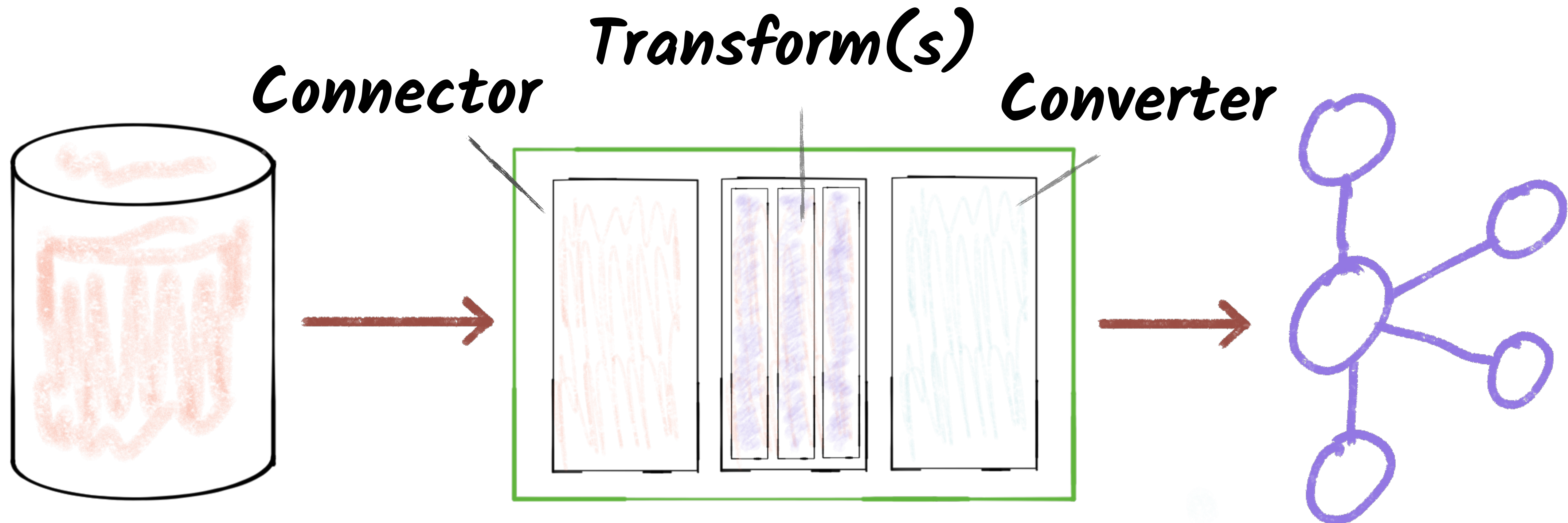
```
  "transforms": "addDateToTopic,labelFooBar",
  "transforms.addDateToTopic.type": "org.apache.kafka.connect.transforms.TimestampRouter",
  "transforms.addDateToTopic.topic.format": "${topic}-${timestamp}",
  "transforms.addDateToTopic.timestamp.format": "YYYYMM",
  "transforms.labelFooBar.type": "org.apache.kafka.connect.transforms.ReplaceField$Value",
  "transforms.labelFooBar.renames": "delivery_address:shipping_address",
}
```

Do these transforms

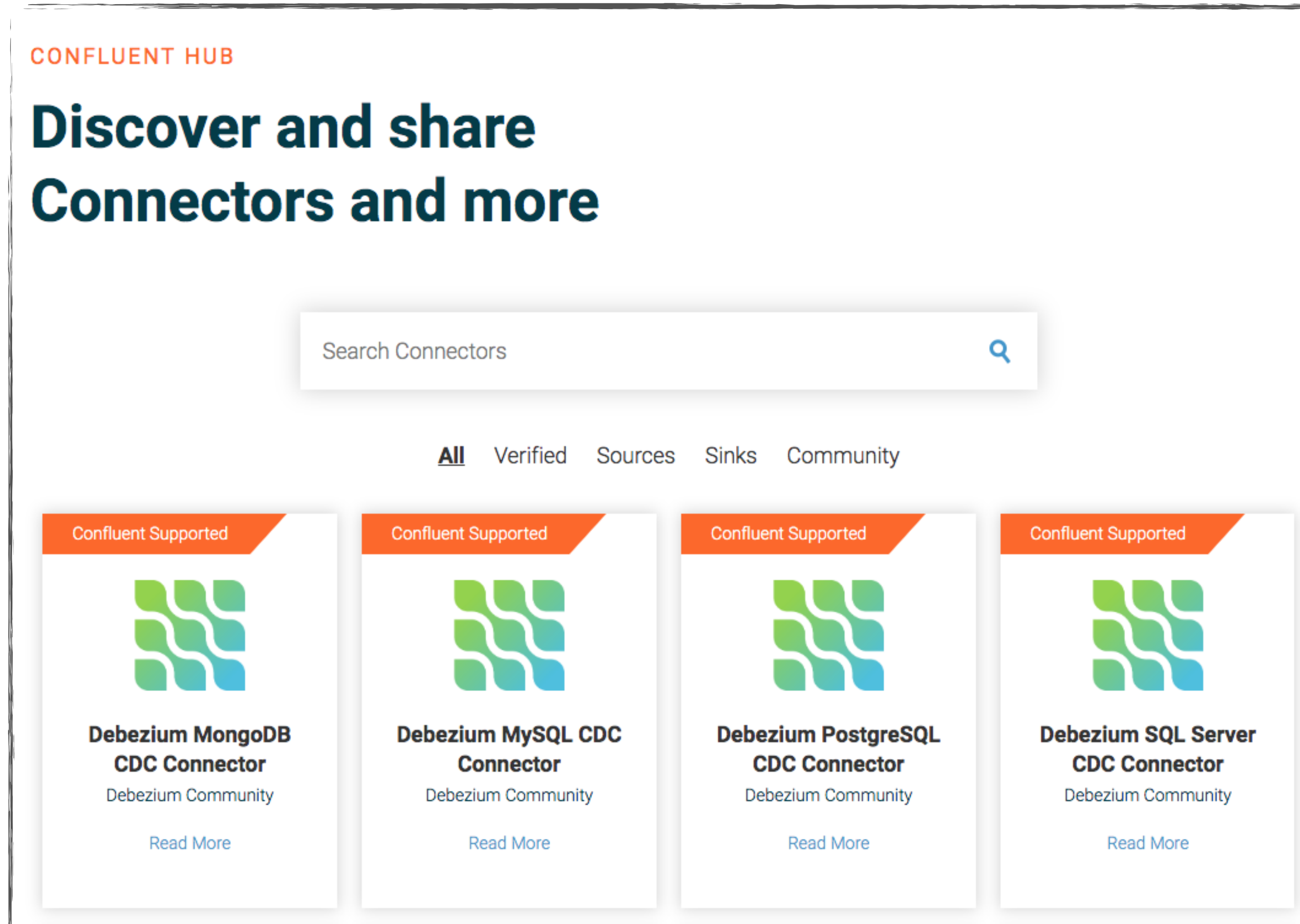
Transforms config

Config per transform

Extensible



Confluent Hub



hub.confluent.io

Deploying Kafka Connect

Connectors, Tasks, and Workers

Connectors and Tasks



S3 Sink

Connectors and Tasks

JDBC Source

S3 Sink

S3 Task #1

JDBC Task #1

Connectors and Tasks

JDBC Source

S3 Sink

S3 Task #1

JDBC Task #1

JDBC Task #2

Tasks and Workers

JDBC Source

S3 Sink

S3 Task #1

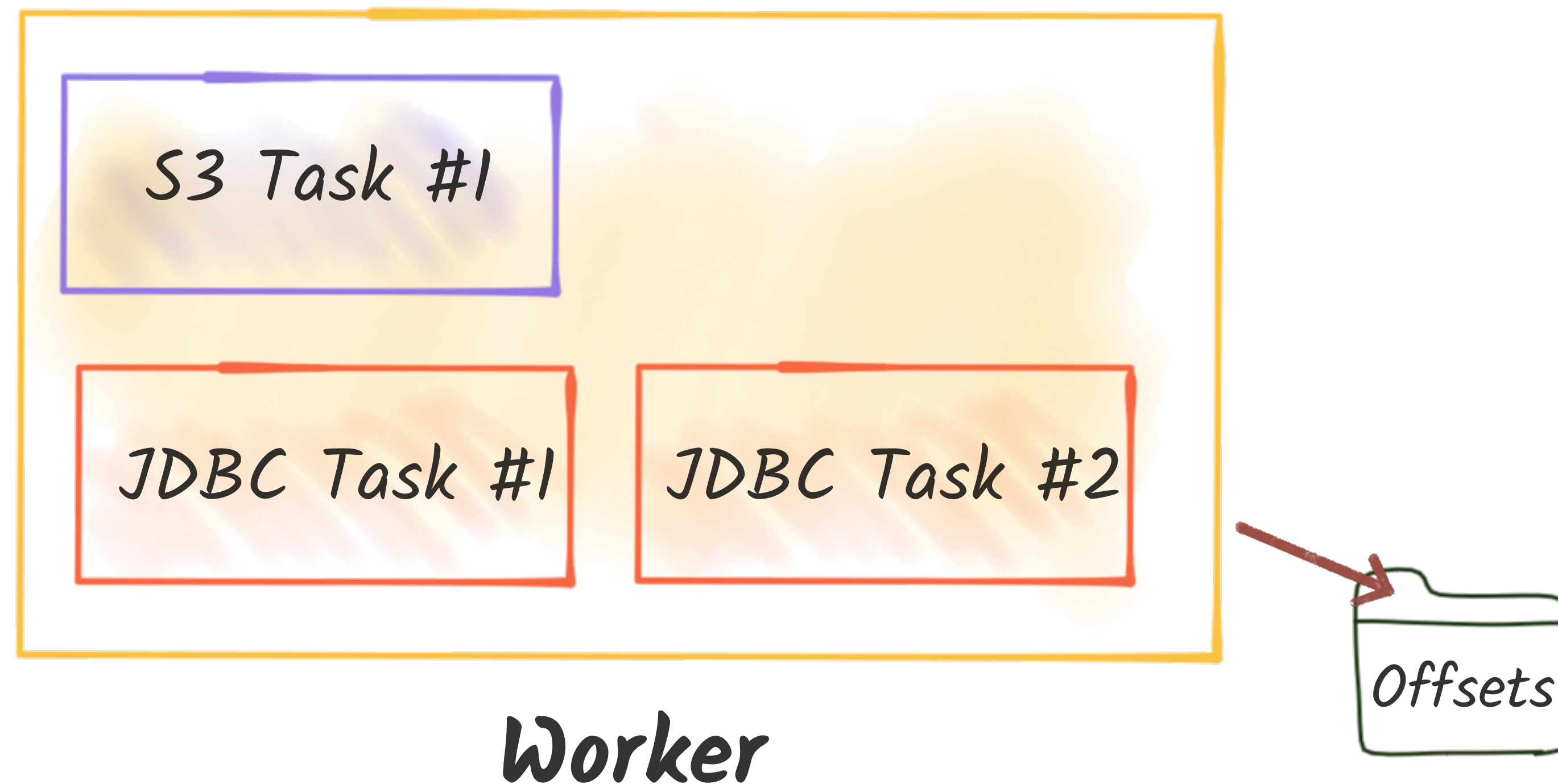
JDBC Task #1

JDBC Task #2

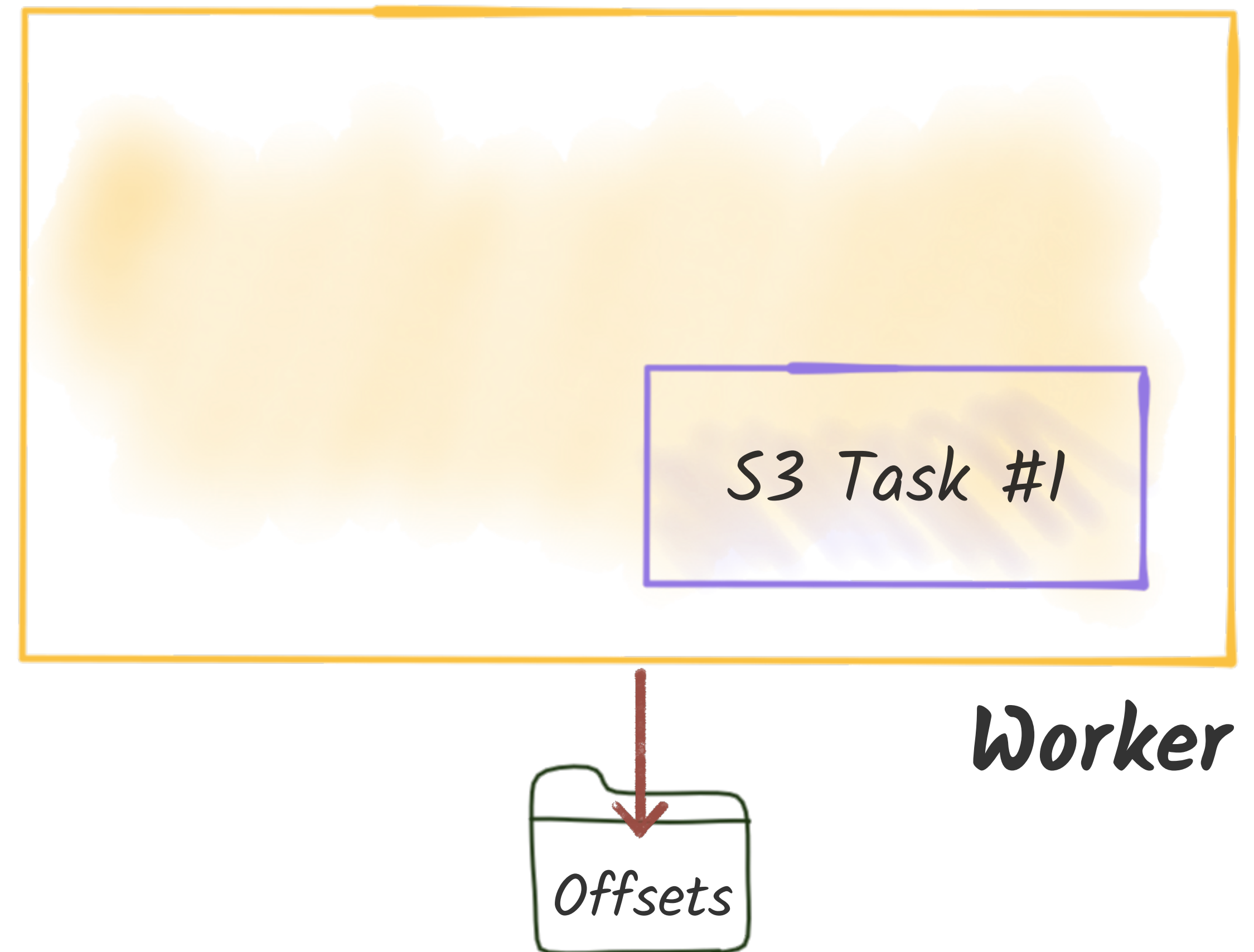
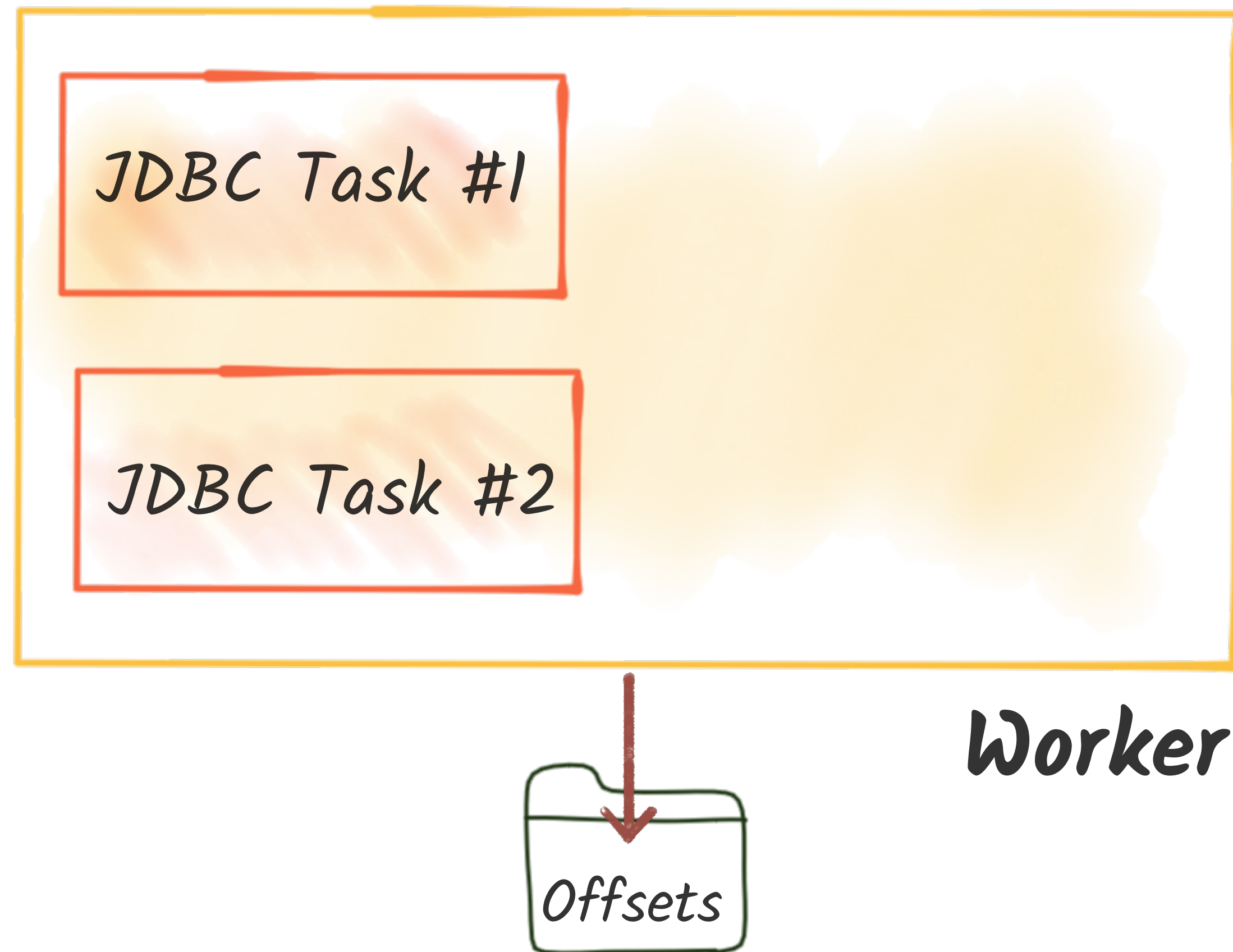
Worker



Kafka Connect Standalone Worker

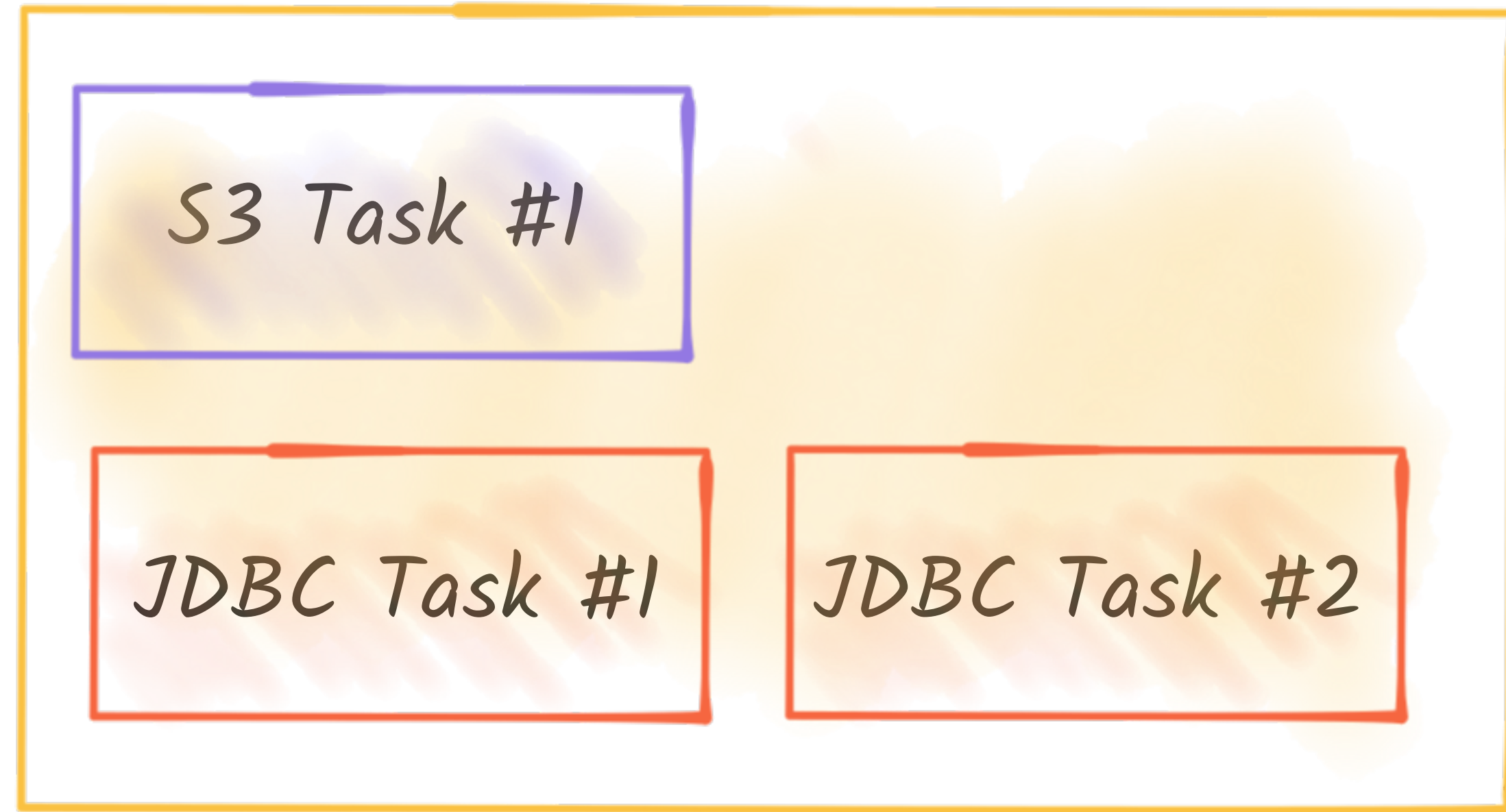


"Scaling" the Standalone Worker



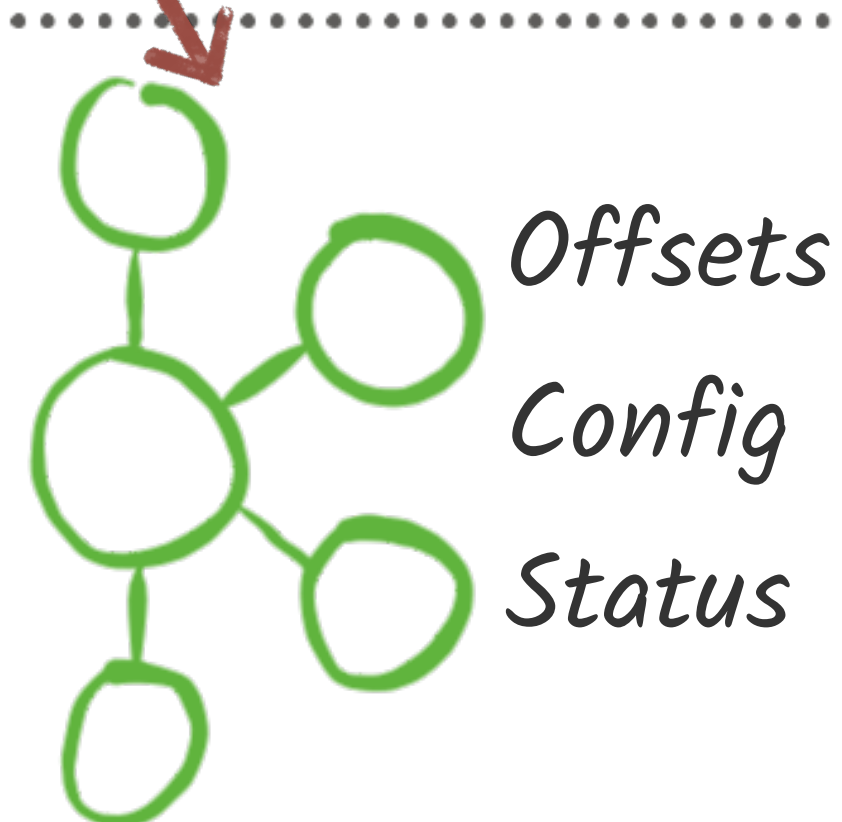
Fault-tolerant? Nope.

Kafka Connect Distributed Worker



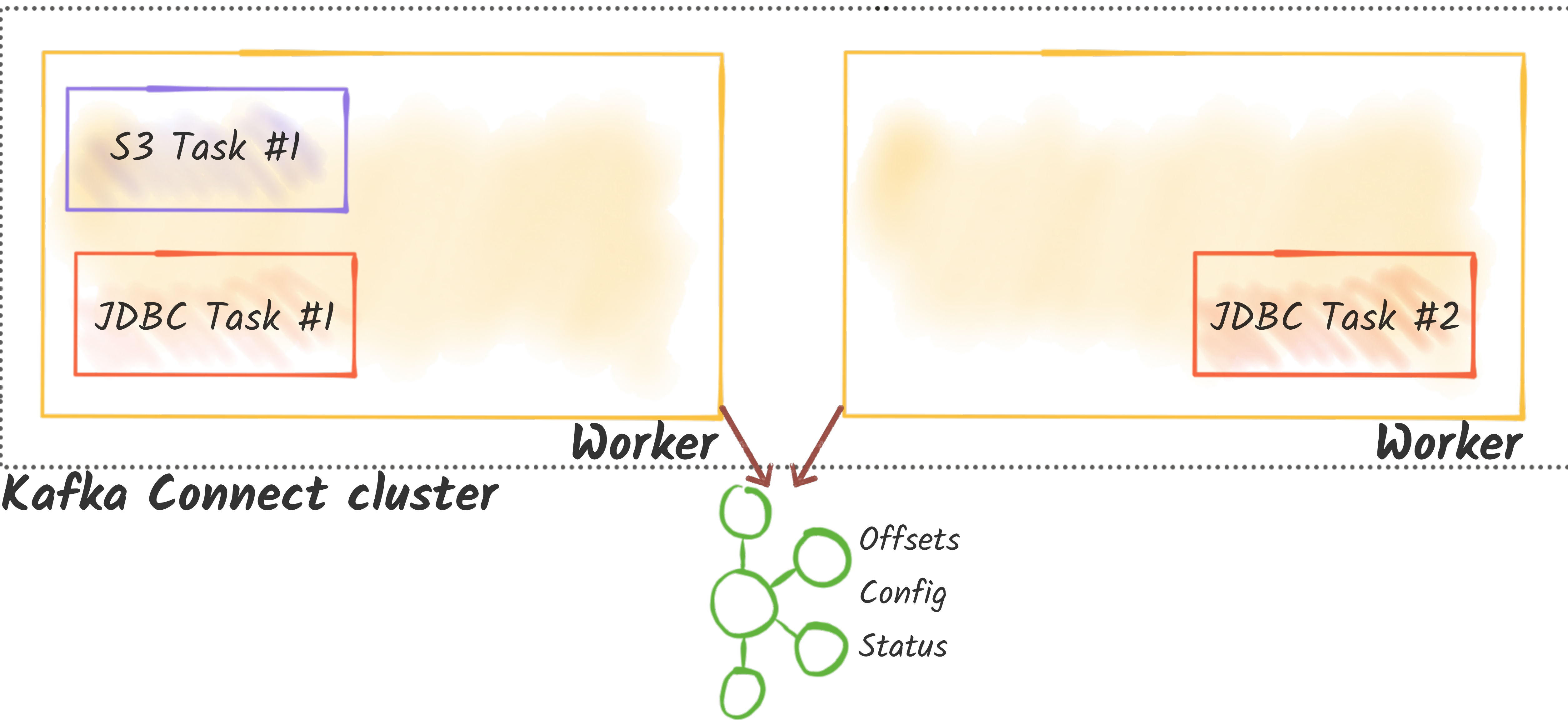
Worker

Kafka Connect cluster



Fault-tolerant? Yeah!

Scaling the Distributed Worker



Kafka Connect cluster

Worker

Worker

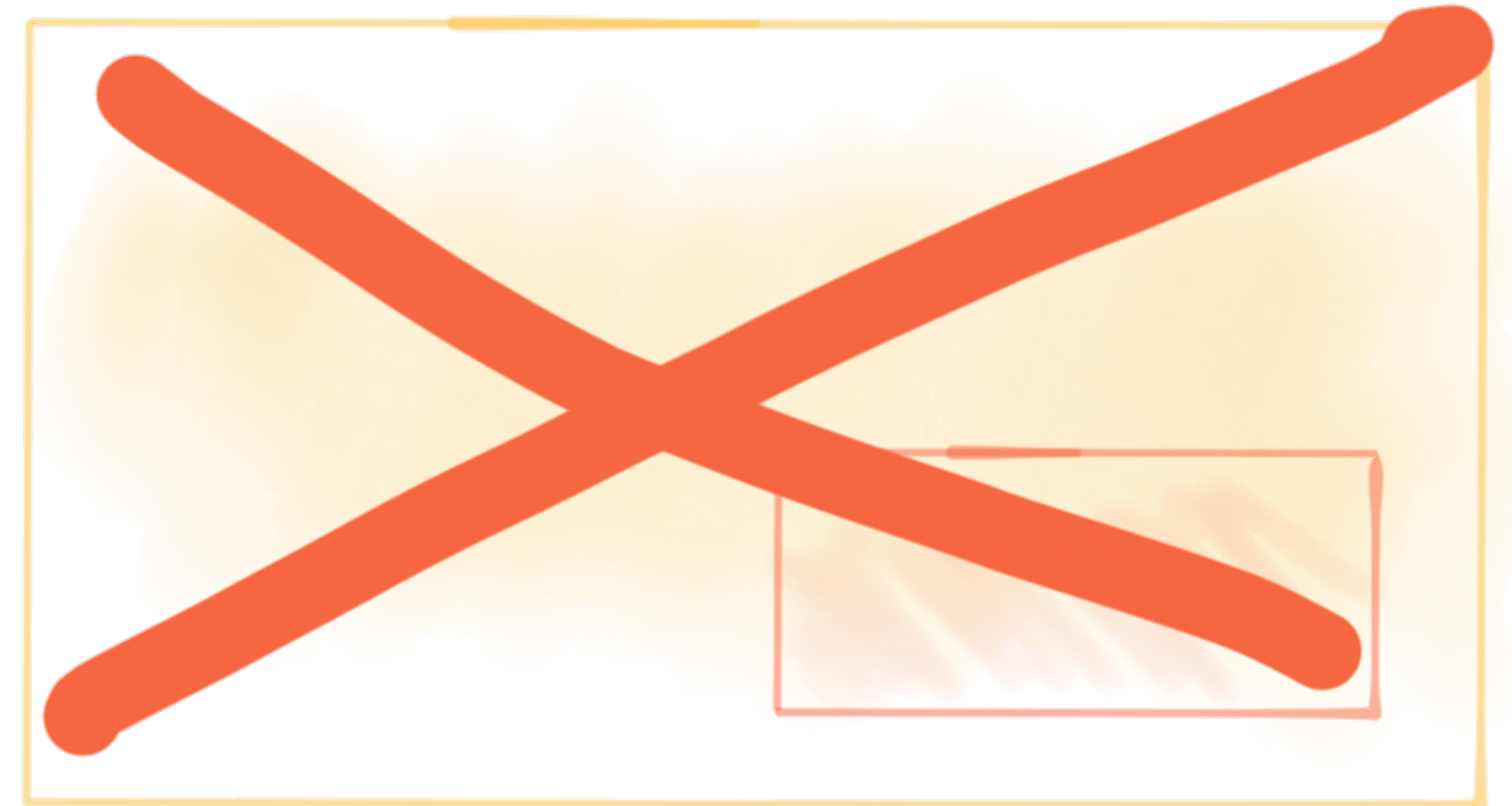
Offsets
Config
Status

Fault-tolerant? Yeah!

Distributed Worker - fault tolerance



Worker



Worker

Kafka Connect cluster



Distributed Worker - fault tolerance

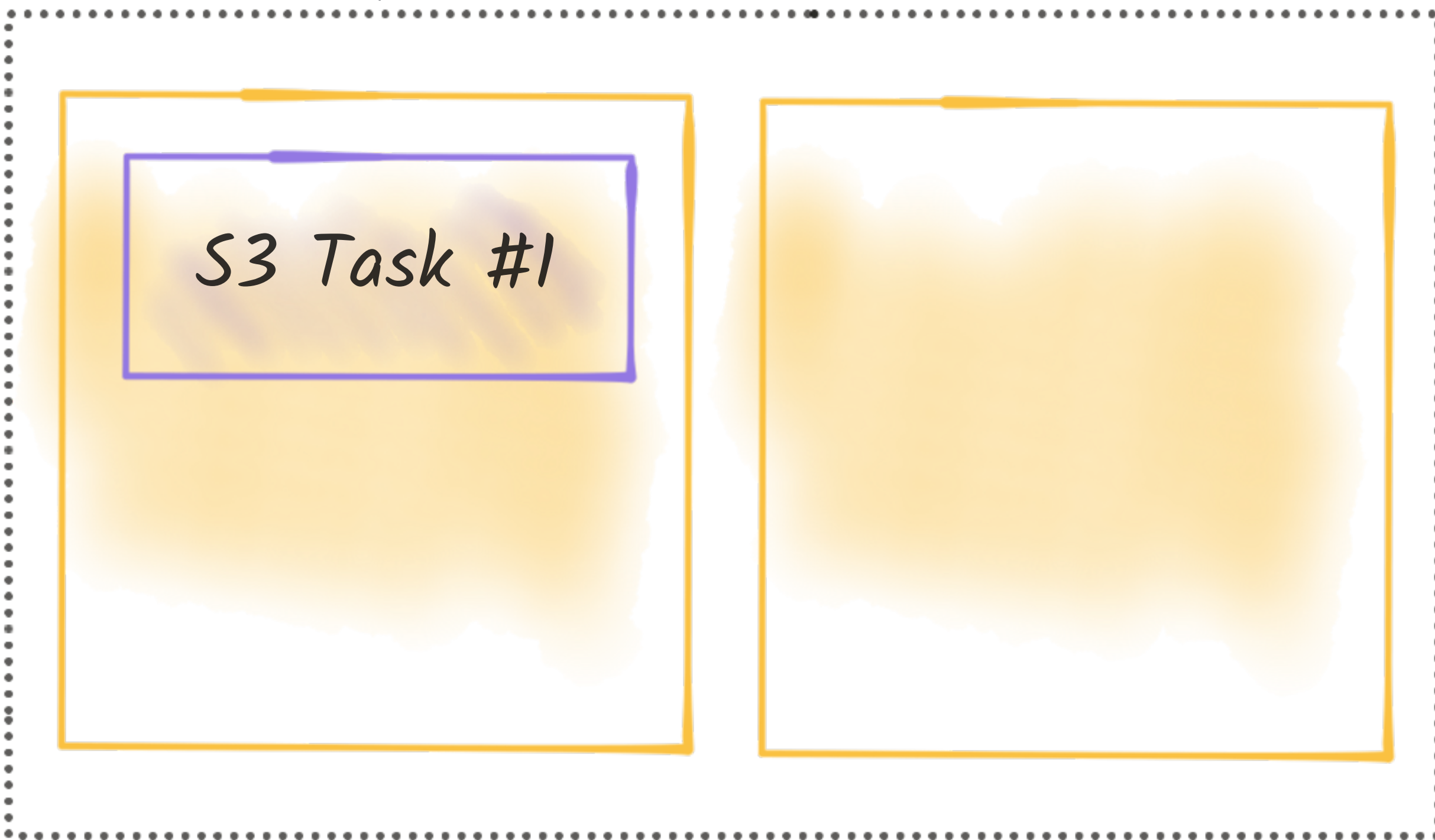


Worker

Kafka Connect cluster



Multiple Distributed Clusters



Kafka Connect cluster #1



Kafka Connect cluster #2



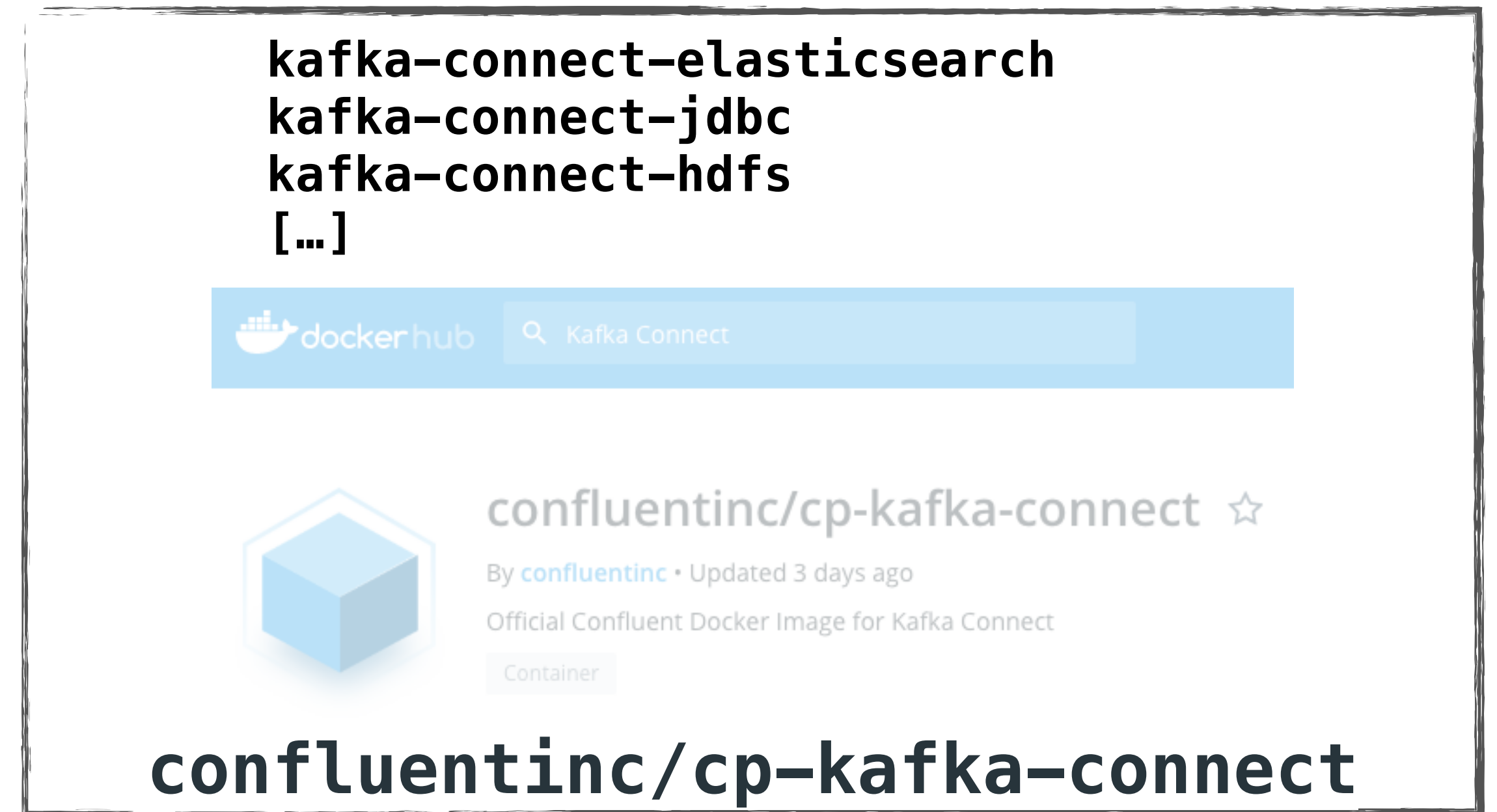
Containers

Kafka Connect images on Docker Hub



The screenshot shows the Docker Hub interface with a search bar at the top containing the text "Search for great content (e.g., mysql)". Below the search bar, the image for "confluentinc/cp-kafka-connect-base" is displayed. It features a blue cube icon, the repository name "confluentinc/cp-kafka-connect-base" with a star icon, the text "By confluentinc • Updated 3 days ago", the description "Confluent Docker Base Image for Kafka Connect", and a "Container" label. Below the screenshot, the repository name is written in large, bold, black text.

confluentinc/cp-kafka-connect-base



The screenshot shows the Docker Hub interface with a search bar at the top containing the text "Kafka Connect". Below the search bar, the image for "confluentinc/cp-kafka-connect" is displayed. It features a blue cube icon, the repository name "confluentinc/cp-kafka-connect" with a star icon, the text "By confluentinc • Updated 3 days ago", the description "Official Confluent Docker Image for Kafka Connect", and a "Container" label. Above the screenshot, a list of related repository names is shown: "kafka-connect-elasticsearch", "kafka-connect-jdbc", "kafka-connect-hdfs", and "[...]". Below the screenshot, the repository name is written in large, bold, black text.

kafka-connect-elasticsearch
kafka-connect-jdbc
kafka-connect-hdfs
[...]

confluentinc/cp-kafka-connect

Adding connectors to a container

Confluent Hub

JAR



At runtime

kafka-connect:

image: confluentinc/cp-kafka-connect:5.2.1

environment:

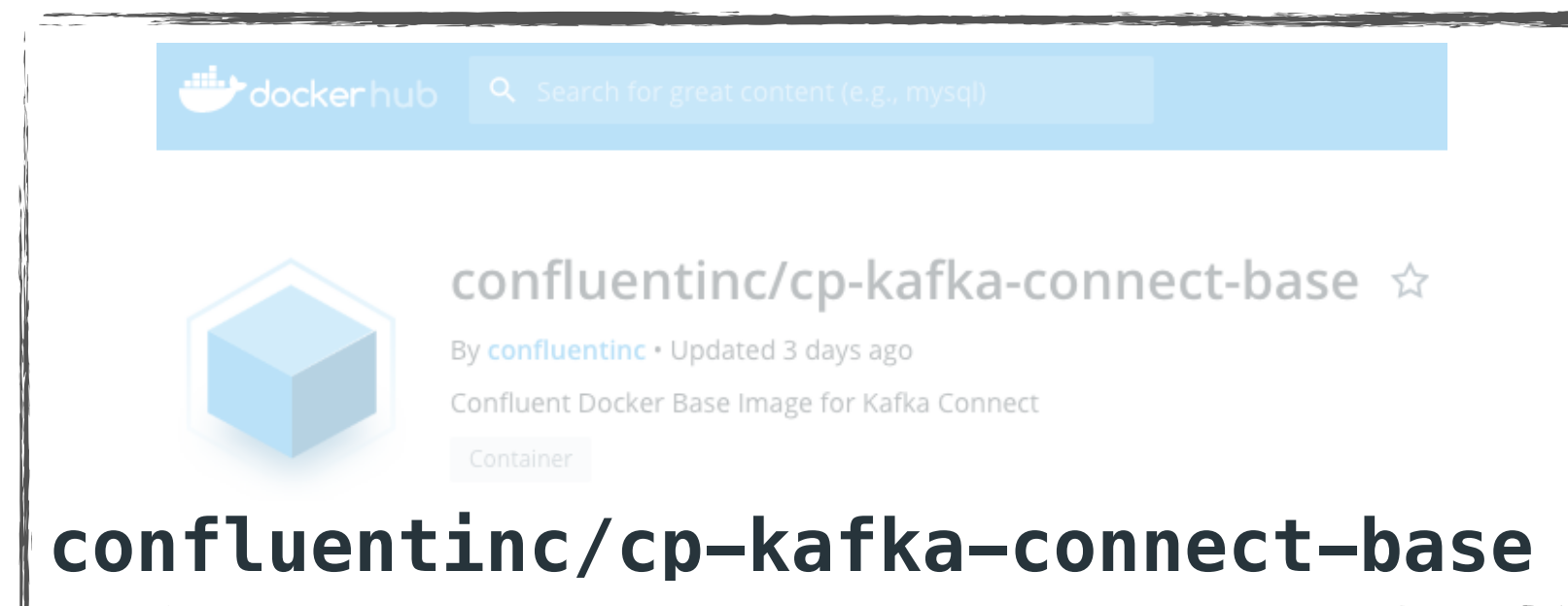
CONNECT_PLUGIN_PATH: '/usr/share/java,/usr/share/confluent-hub-components'

command:

- bash
- -C
- |

confluent-hub install --no-prompt neo4j/kafka-connect-neo4j:1.0.0
/etc/confluent/docker/run

JAR



<http://rmoff.dev/ksln19-connect-docker>

Build a new image

```
FROM confluentinc/cp-kafka-connect:5.2.1
ENV CONNECT_PLUGIN_PATH="/usr/share/java,/usr/share/confluent-hub-components"
RUN confluent-hub install --no-prompt neo4j/kafka-connect-neo4j:1.0.0
```

JAR





Automating connector creation

```
# Launch Kafka Connect
```

```
/etc/confluent/docker/run &
```

```
#
```

```
# Wait for Kafka Connect listener
```

```
while [ $$$(curl -s -o /dev/null -w %{http_code} http://$$CONNECTOR_HOST:8083/status) != 200 ]
do
    echo -e $$(date) " Kafka Connect listener HTTP state: " $$(curl -s -o /dev/null -w %{http_code} http://$$CONNECTOR_HOST:8083/status)
    sleep 5
done
```

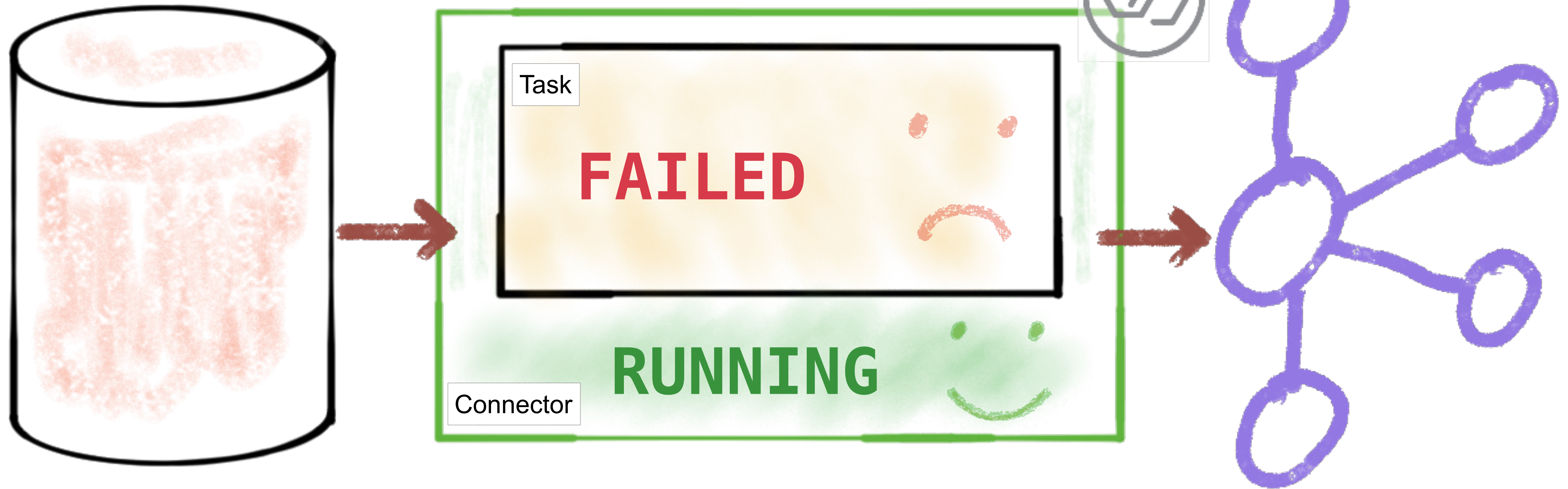
```
#
```

```
# Create JDBC Source connector
```

```
curl -X POST http://localhost:8083/connectors -H "Content-Type: application/json" -d '{
  "name": "jdbc_source_mysql_00",
  "config": {
    "connector.class": "io.confluent.connect.jdbc.JdbcSourceConnector",
    "connection.url": "jdbc:mysql://mysql:3306/default"
  }
}
```

Troubleshooting Kafka Connect

Troubleshooting Kafka Connect



```
$ curl -s "http://localhost:8083/connectors/source-debezium-orders/status" | \
jq '.connector.state'
```

"RUNNING"

```
$ curl -s "http://localhost:8083/connectors/source-debezium-orders/status" | \
jq '.tasks[0].state'
```

"FAILED"



Troubleshooting Kafka Connect

```
curl -s "http://localhost:8083/connectors/source-debezium-orders-00/status"  
| jq '.tasks[0].trace'
```

```
"org.apache.kafka.connect.errors.ConnectException\n\tat  
io.debezium.connector.mysql.AbstractReader.wrap(AbstractReader.java:230)\n\tat  
io.debezium.connector.mysql.AbstractReader.failed(AbstractReader.java:197)\n\tat  
io.debezium.connector.mysql.BinlogReader$ReaderThreadLifecycleListener.onCommunicationFailure(BinlogReader.java:  
1018)\n\tat com.github.shyiko.mysql.binlog.BinaryLogClient.listenForEventPackets(BinaryLogClient.java:950)\n\tat  
com.github.shyiko.mysql.binlog.BinaryLogClient.connect(BinaryLogClient.java:580)\n\tat  
com.github.shyiko.mysql.binlog.BinaryLogClient$7.run(BinaryLogClient.java:825)\n\tat java.lang.Thread.run(Thread.java:  
748)\nCaused by: java.io.EOFException\n\tat  
com.github.shyiko.mysql.binlog.io.ByteArrayInputStream.read(ByteArrayInputStream.java:190)\n\tat  
com.github.shyiko.mysql.binlog.io.ByteArrayInputStream.readInteger(ByteArrayInputStream.java:46)\n\tat  
com.github.shyiko.mysql.binlog.event.deserialization.EventHeaderV4Deserializer.deserialize(EventHeaderV4Deserializer.java  
:35)\n\tat  
com.github.shyiko.mysql.binlog.event.deserialization.EventHeaderV4Deserializer.deserialize(EventHeaderV4Deserializer.java  
:27)\n\tat com.github.shyiko.mysql.binlog.event.deserialization.EventDeserializer.nextEvent(EventDeserializer.java:  
212)\n\tat io.debezium.connector.mysql.BinlogReader$1.nextEvent(BinlogReader.java:224)\n\tat  
com.github.shyiko.mysql.binlog.BinaryLogClient.listenForEventPackets(BinaryLogClient.java:922)\n\t... 3 more\n"
```


The log is the source of truth

```
$ confluent log connect
```

```
$ docker-compose logs kafka-connect
```

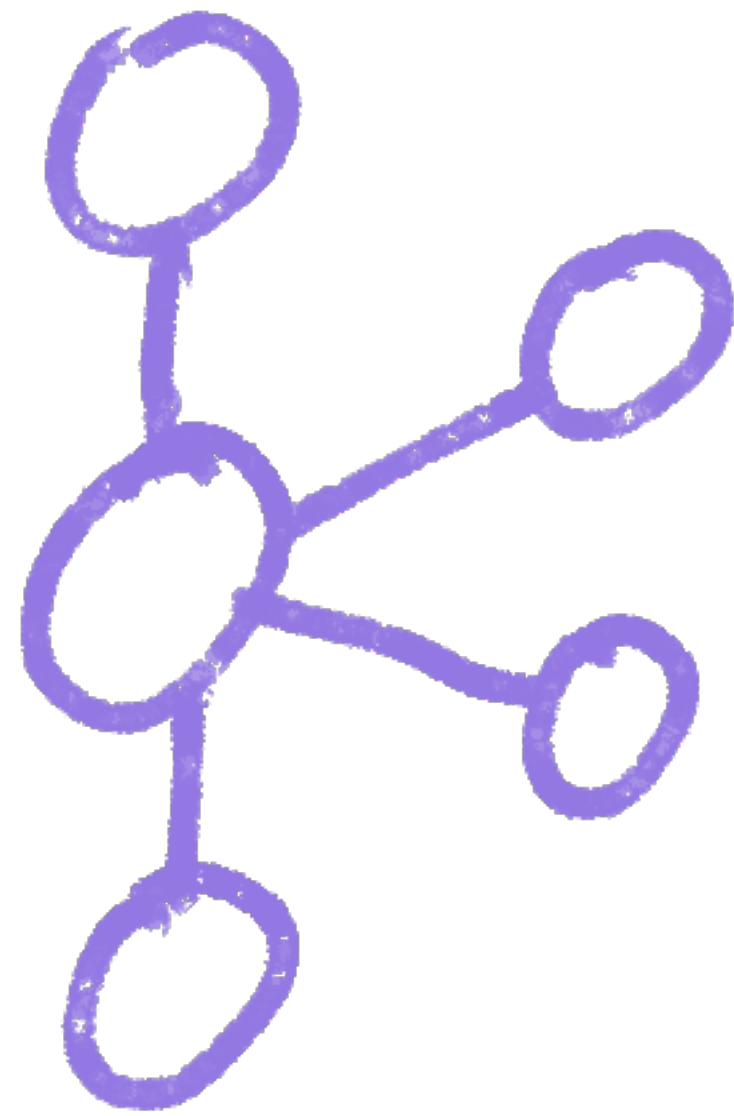
```
$ cat /var/log/kafka/connect.log
```


Error Handling and Dead Letter Queues

*org.apache.kafka.common.errors.SerializationException:
Unknown magic byte!*

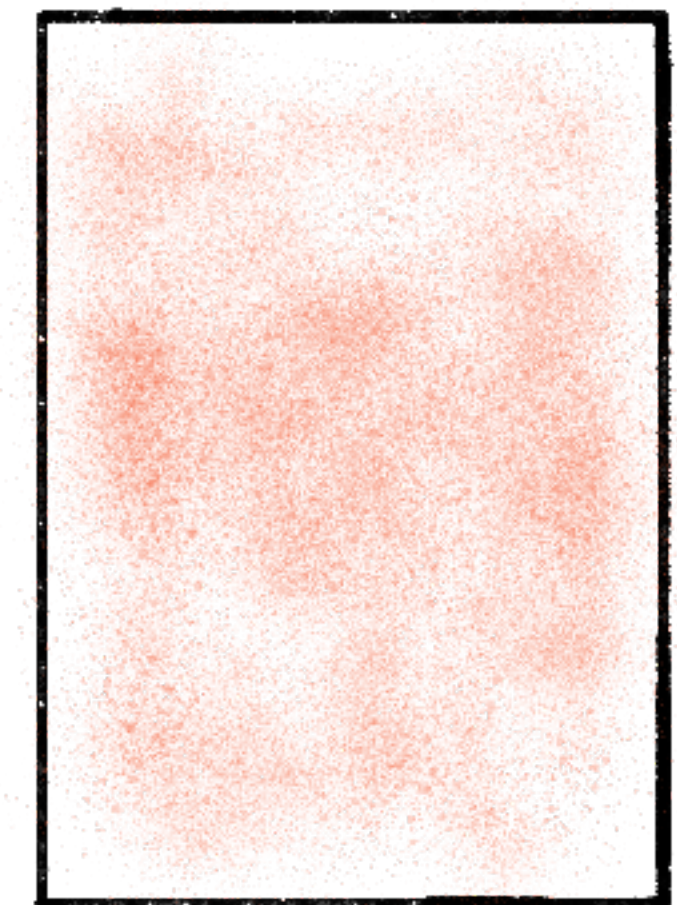
Mismatched converters

*org.apache.kafka.common.errors.SerializationException:
Unknown magic byte!*



Messages are not Avro

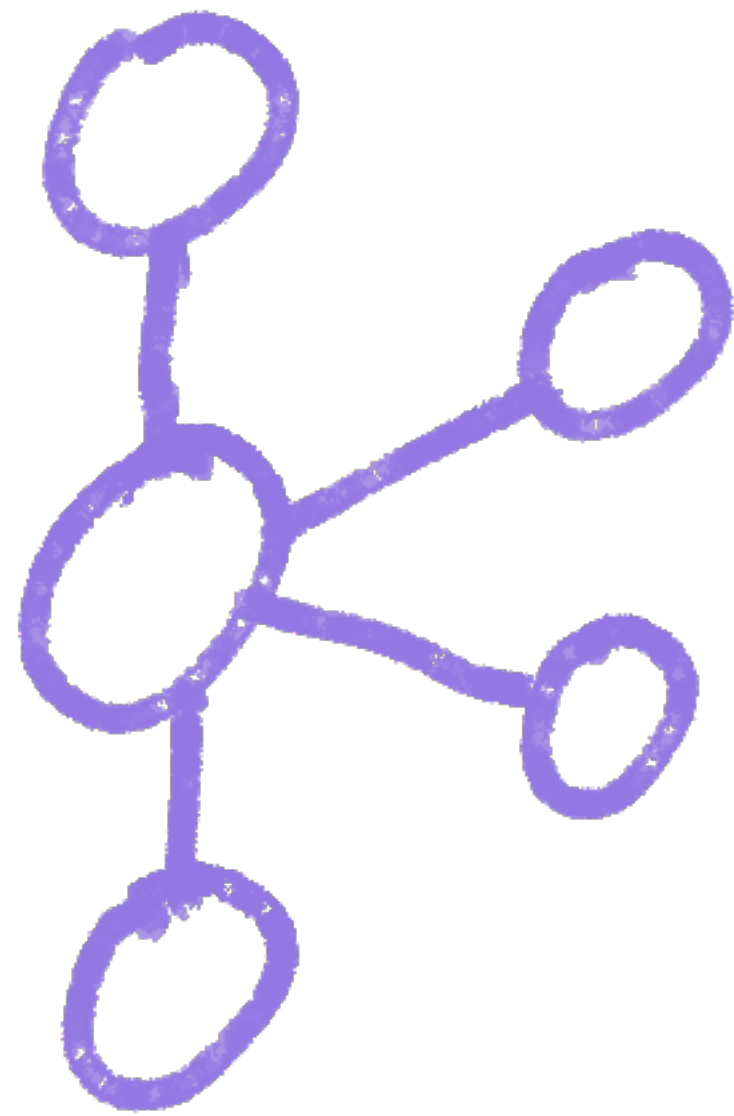
**"value.converter":
"AvroConverter"**



Use the correct Converter for the
source data

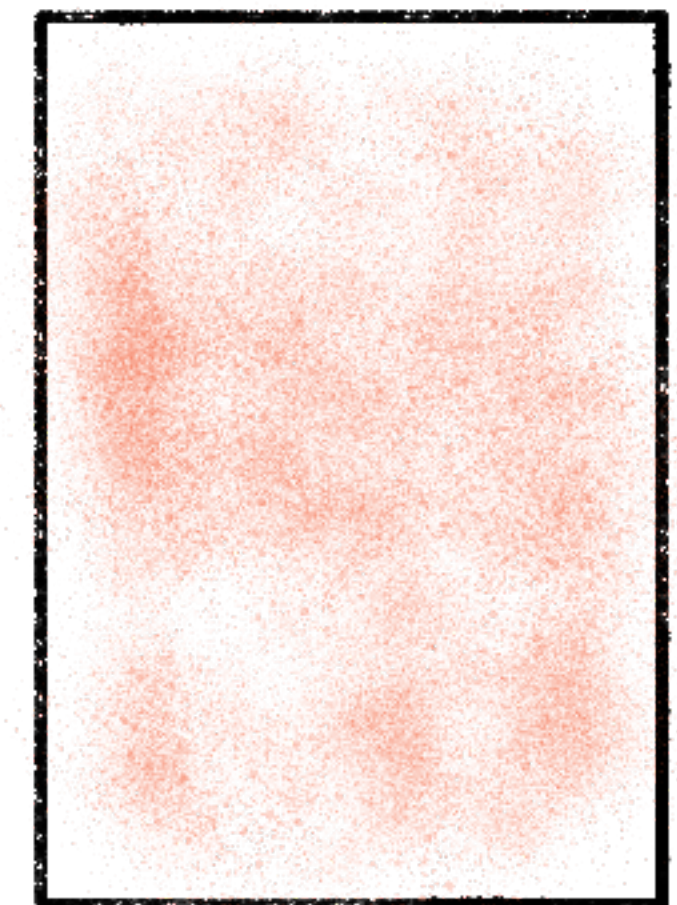
Mixed serialisation methods

*org.apache.kafka.common.errors.SerializationException:
Unknown magic byte!*



Some messages are not Avro

**"value.converter":
"AvroConverter"**



Use error handling to deal
with bad messages

Error Handling and DLQ

Handled

Convert

-> read/write from Kafka

-> [de]-serialisation

Transform

Not Handled

Start

-> Connections to a data store

Poll / Put

-> Read/Write from/to data store*

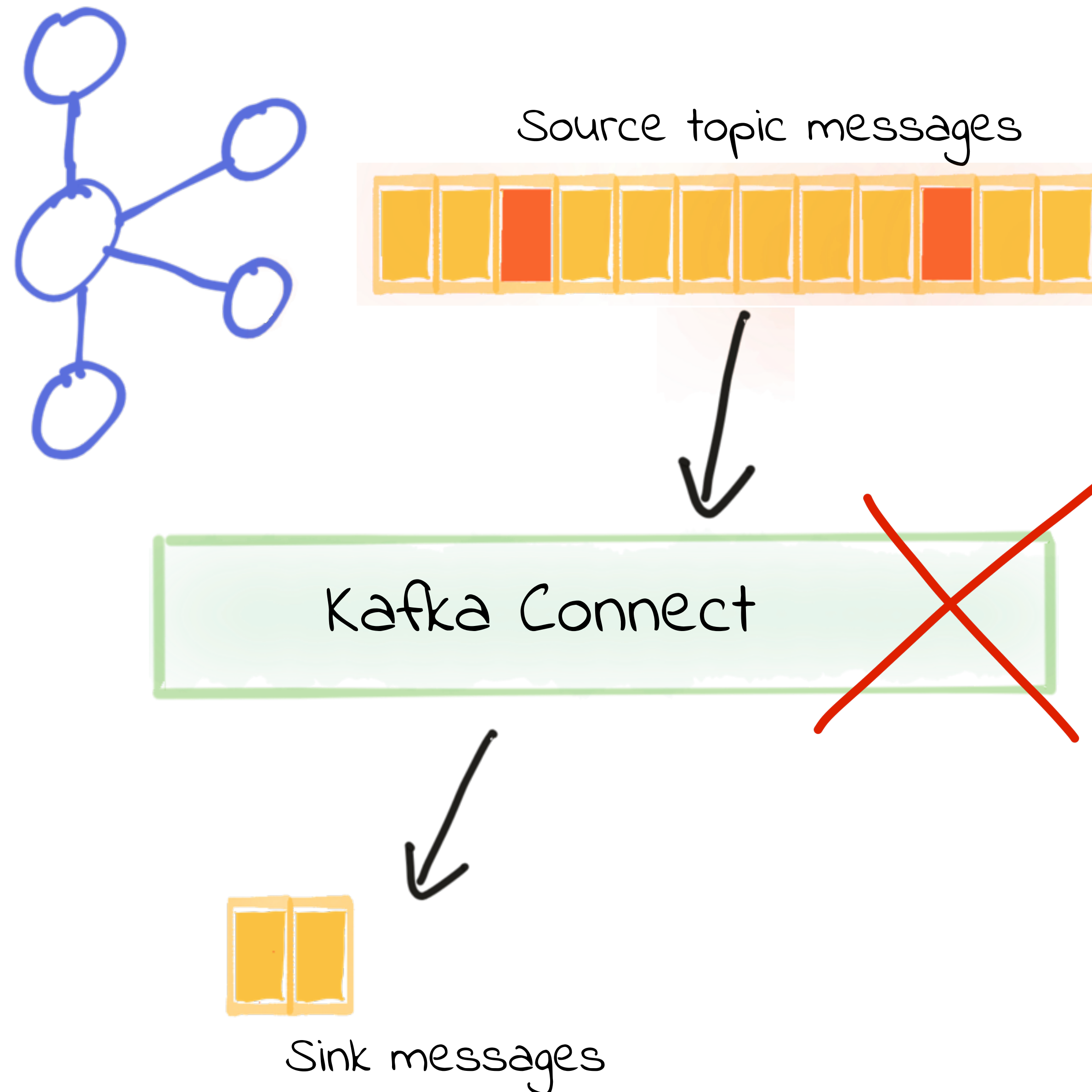
* can be retried by Connect



<https://cnfl.io/connect-dlq>

Fail Fast

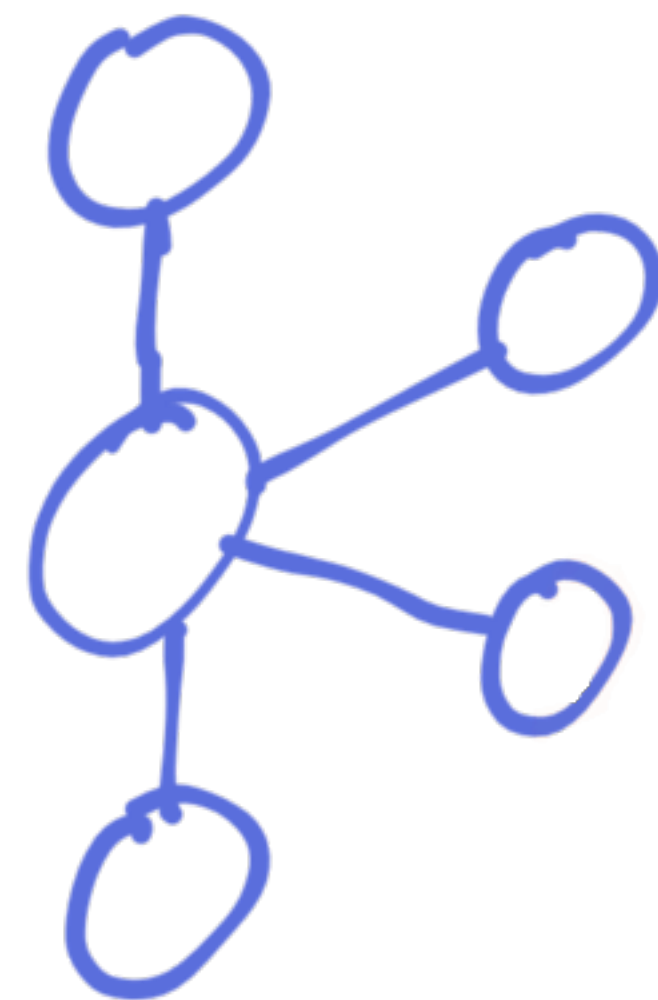
@rmoff #KafkaMeetup



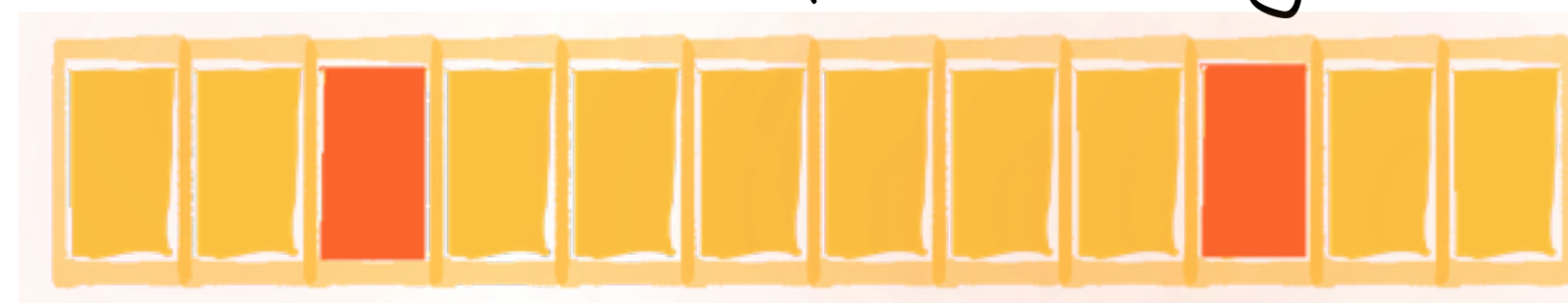
<https://cnfl.io/connect-dlq>



YOLO -_ (ツ) _/



Source topic messages



Kafka Connect



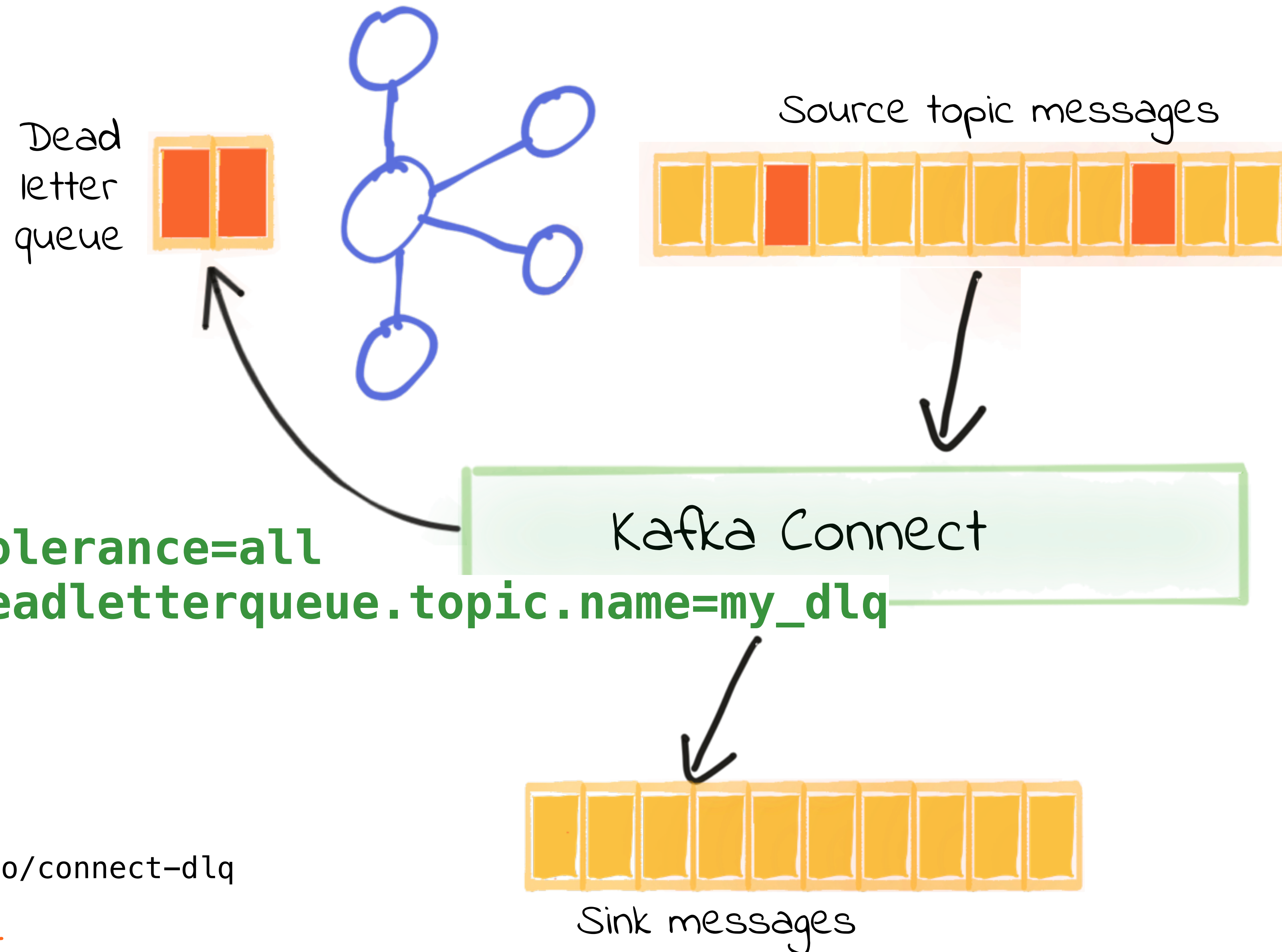
Sink messages

`errors.tolerance=all`



<https://cnfl.io/connect-dlq>

Dead Letter Queue

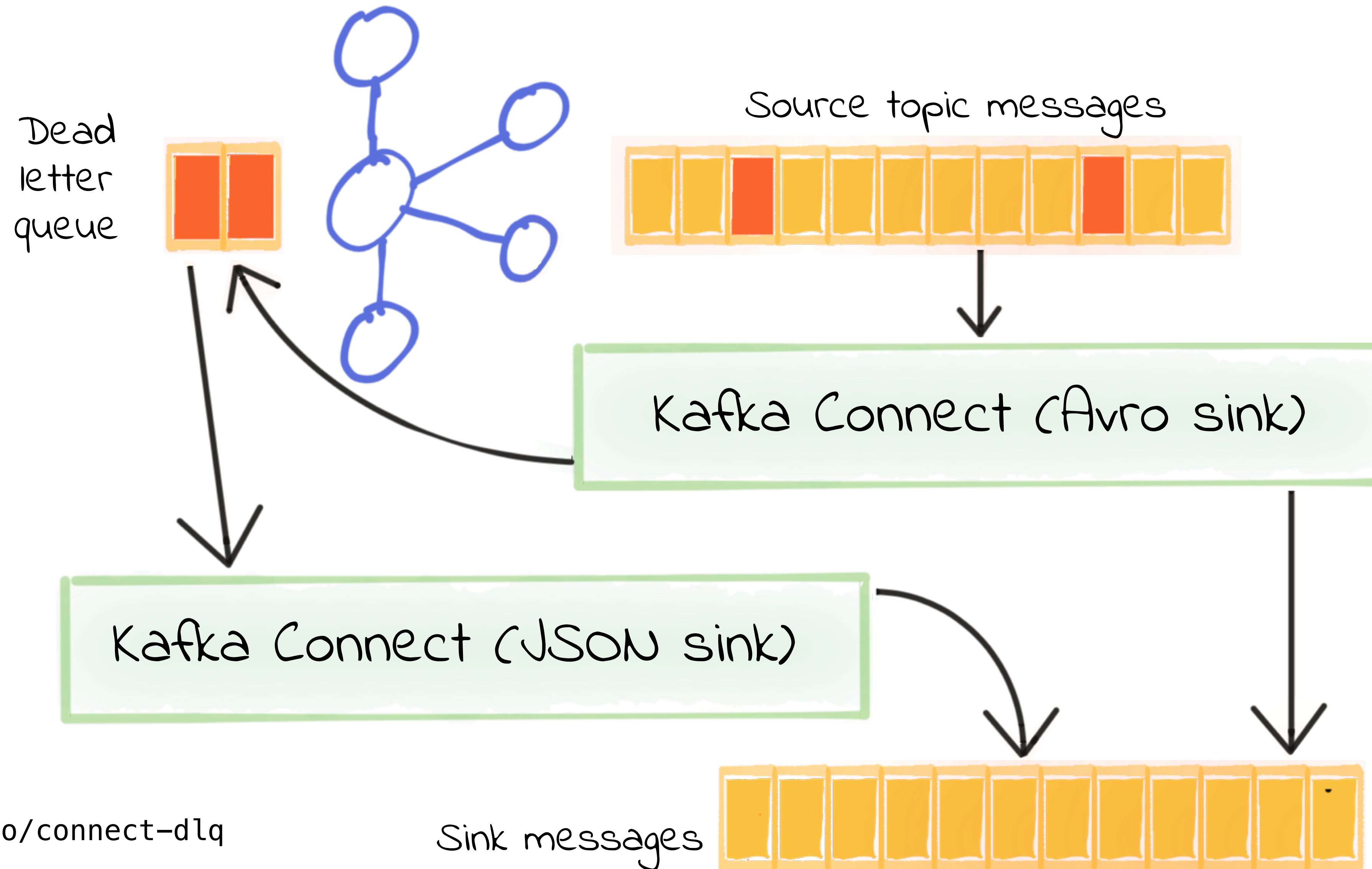


`errors.tolerance=all`
`errors.deadletterqueue.topic.name=my_dlq`



<https://cnfl.io/connect-dlq>

Re-processing the Dead Letter Queue



<https://cnfl.io/connect-dlq>

Metrics and Monitoring

REST API

```
[~] Robin@asgard02-2.local
$ curl -s "http://localhost:8083/connectors" | \
  jq '.[0]' | \
  xargs -I{connector_name} curl -s "http://localhost:8083/connectors/{connector_name}/status" | \
  jq -c -M '[.name,.connector.state,.tasks[].state]|join(":|: ")' | \
  column -s : -t | \
  sed 's/\\"//g' | \
  sort
```

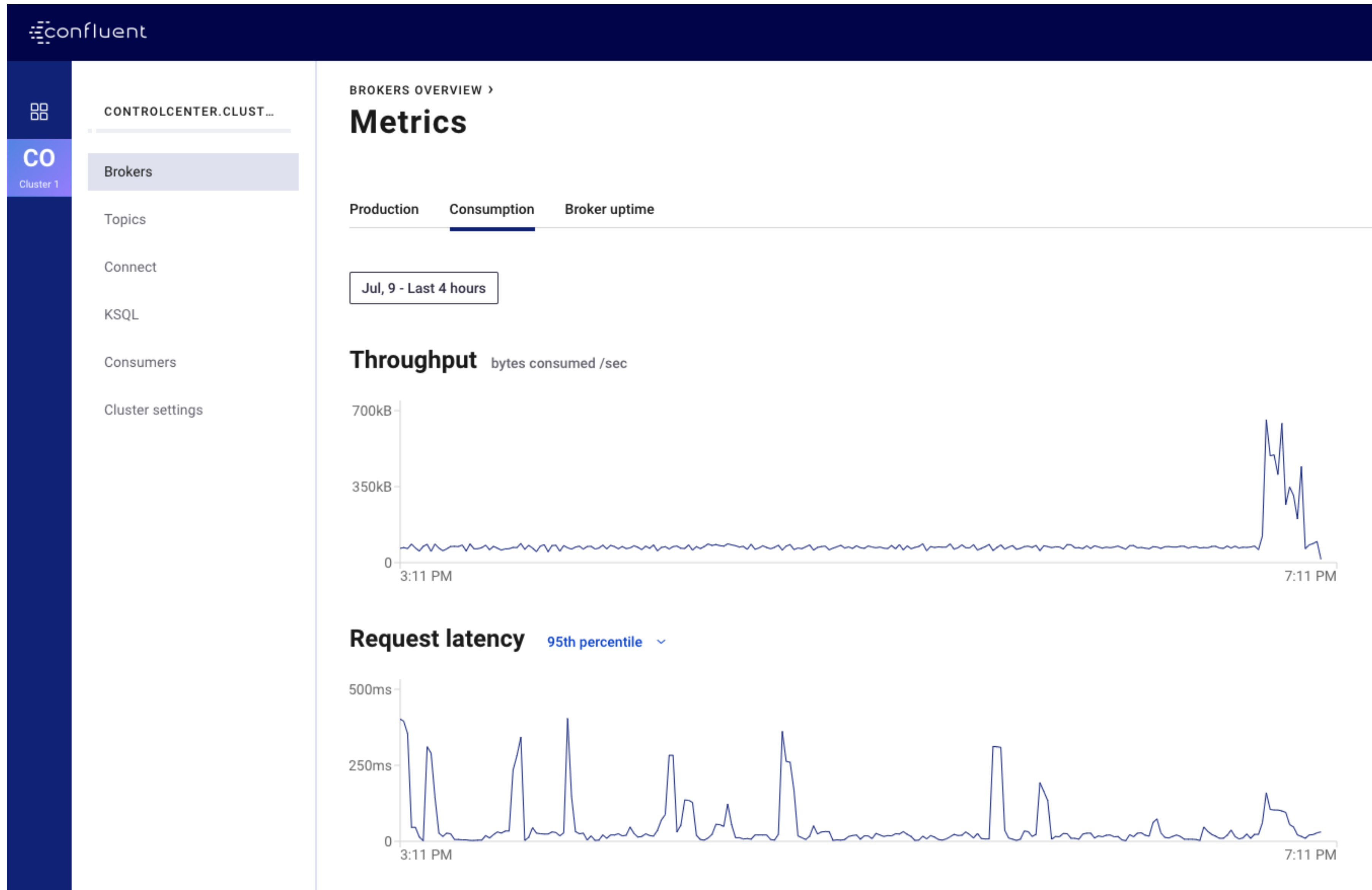
sink-elastic-orders-00		RUNNING		RUNNING
sink-elastic-orders-01		RUNNING		RUNNING
source-debezium-orders-00		RUNNING		RUNNING

```
[~] Robin@asgard02-2.local
$
```



<http://go.rmoff.net/connector-status>

Confluent Control Center



Consumer lag

confluent

MONITORING

System health

Data streams

Consumer lag

MANAGEMENT

Kafka Connect

Clusters

Topics

ALERTS

Overview

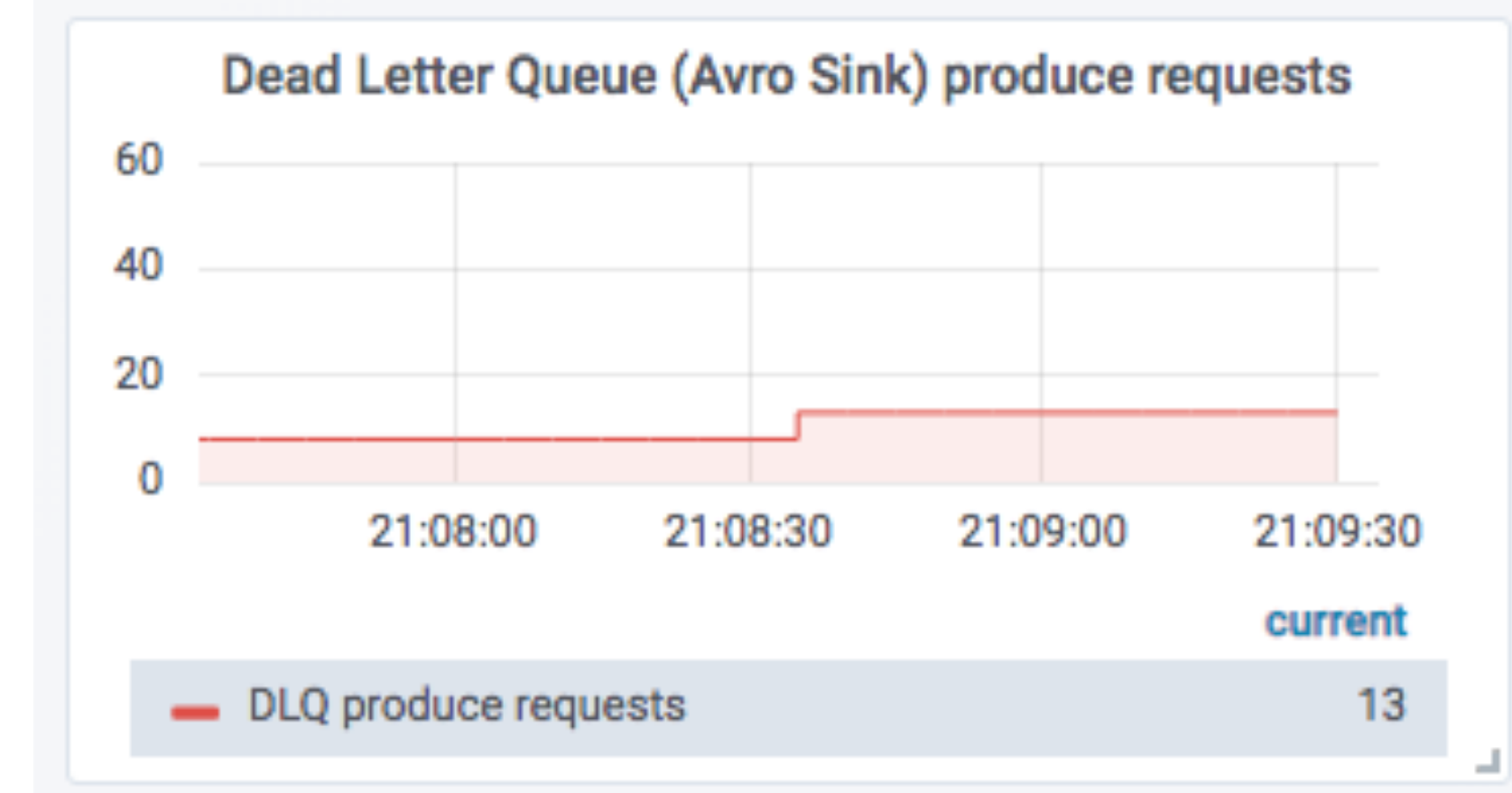
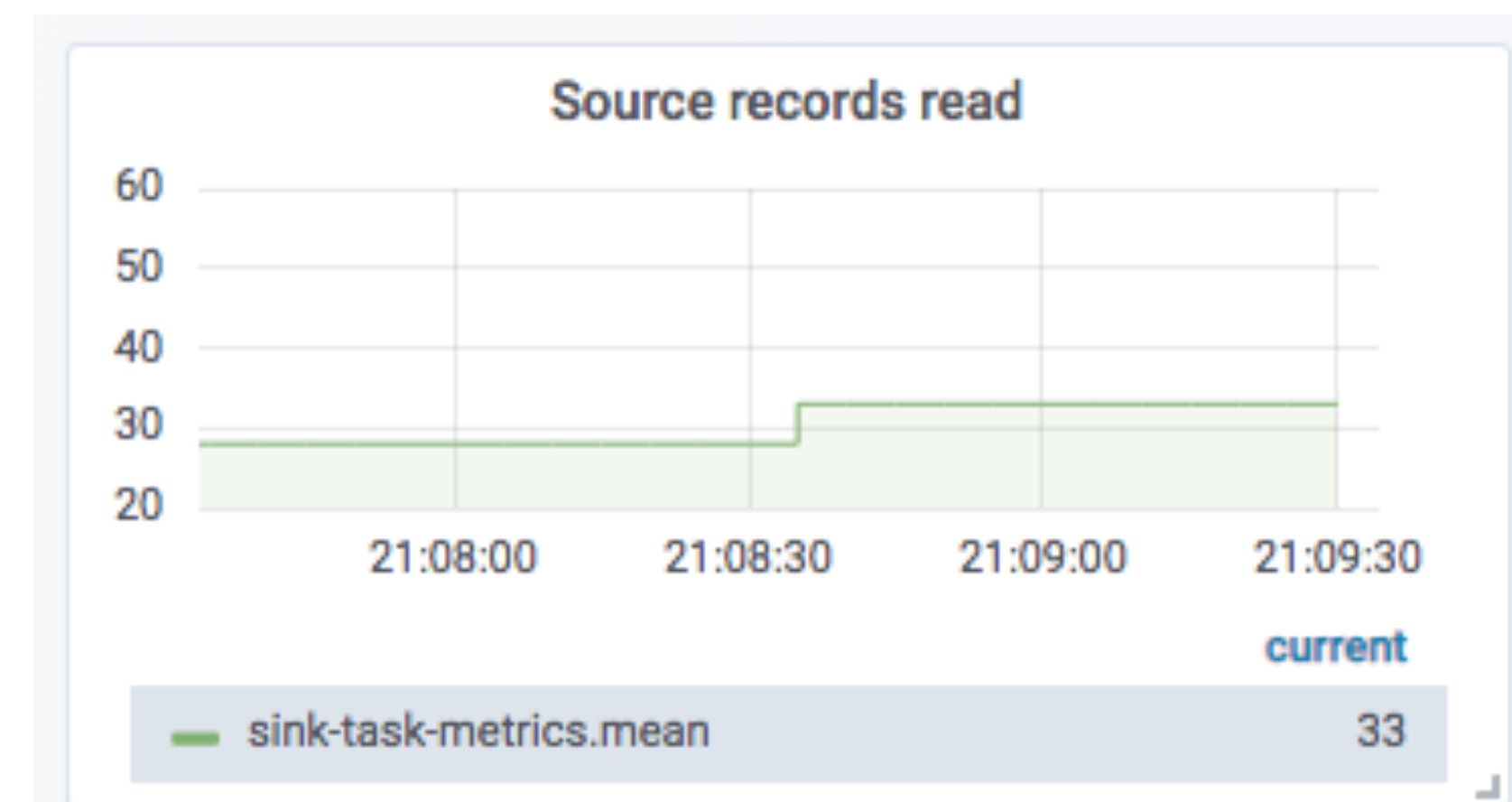
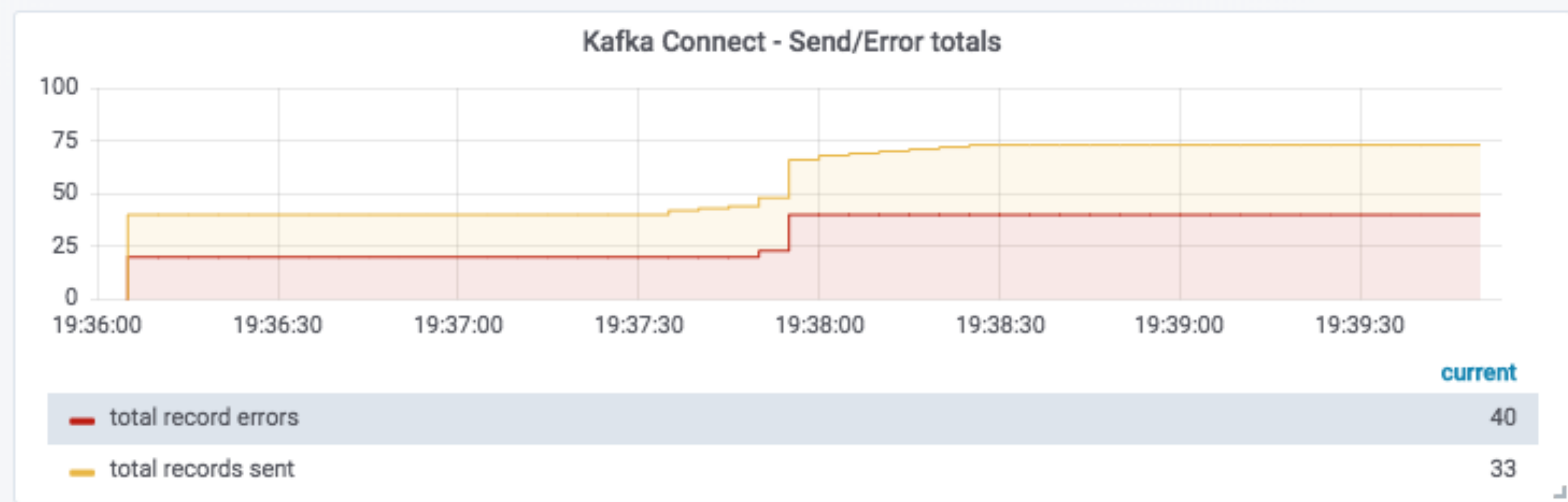
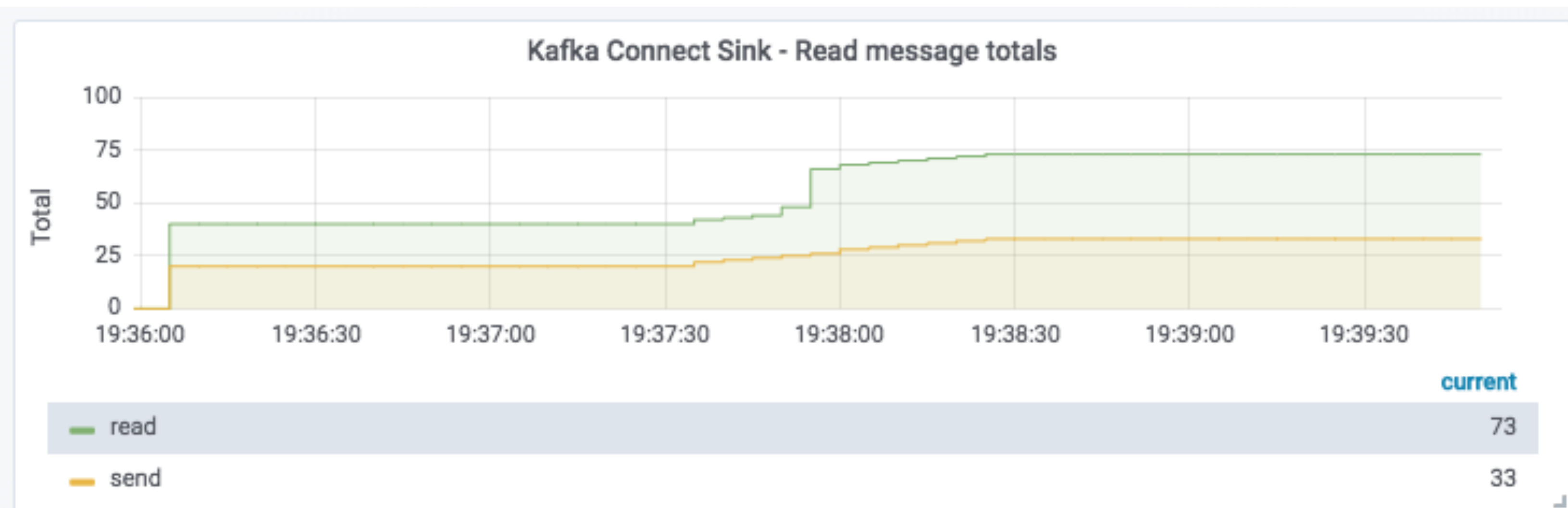
Integration

MONITORING >

Consumer lag

connect-sink

Consumer group Name	Messages behind	Consumers	Topics
connect-sink-jdbc-TRAIN_MOVEMENTS_ACTIVATIONS_SCHEDULE_00	23	1	1
connect-sink-file-delay-alerts	0	1	1
connect-sink-elastic-train_movements_01-v01	23	1	1
connect-sink-elastic-train_movements_activations_schedule_00-v01	23	1	1
connect-sink-s3-tiploc-v00	21283	1	1
connect-sink-s3-train-movements-v00	4613	1	1



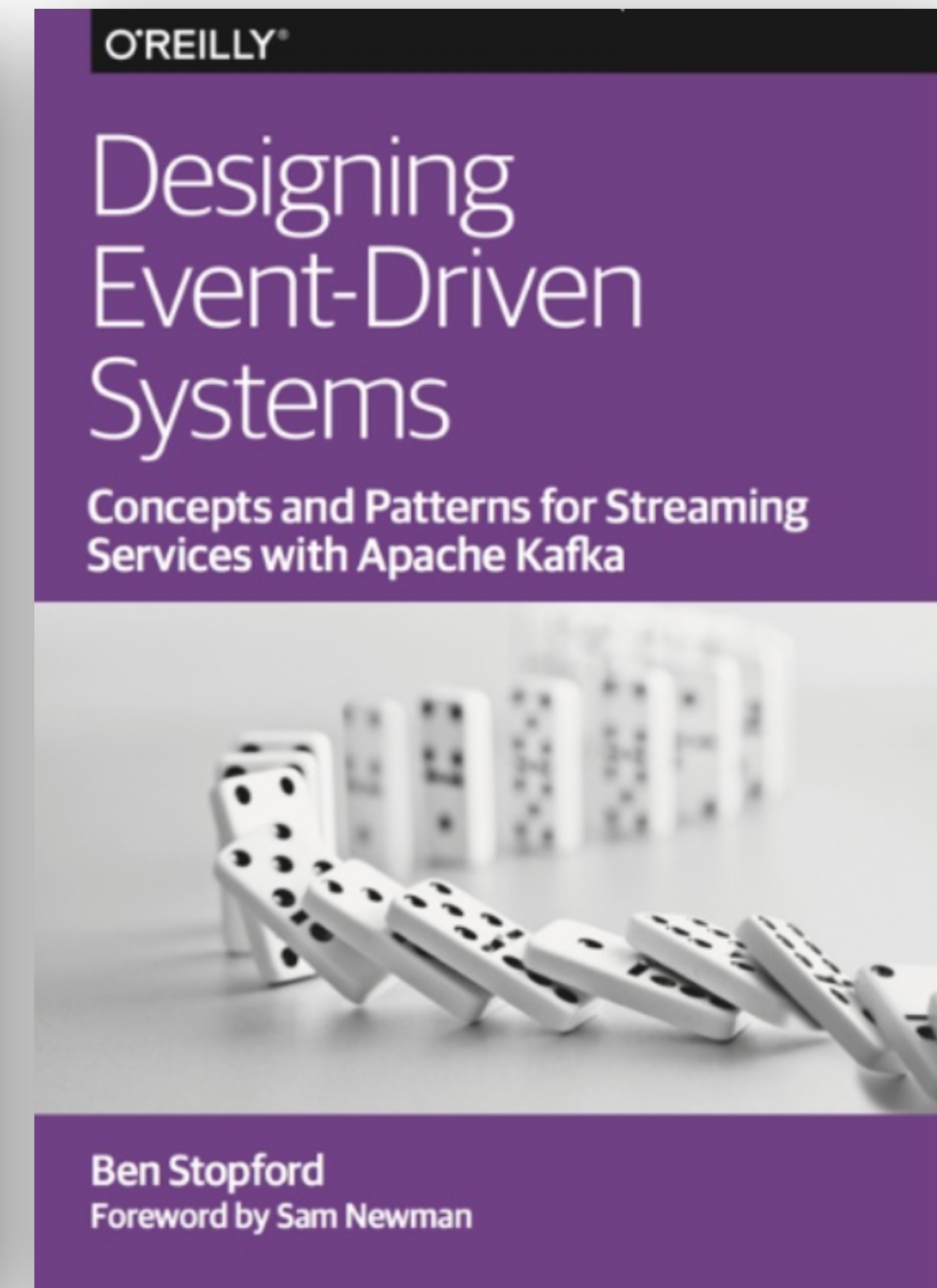
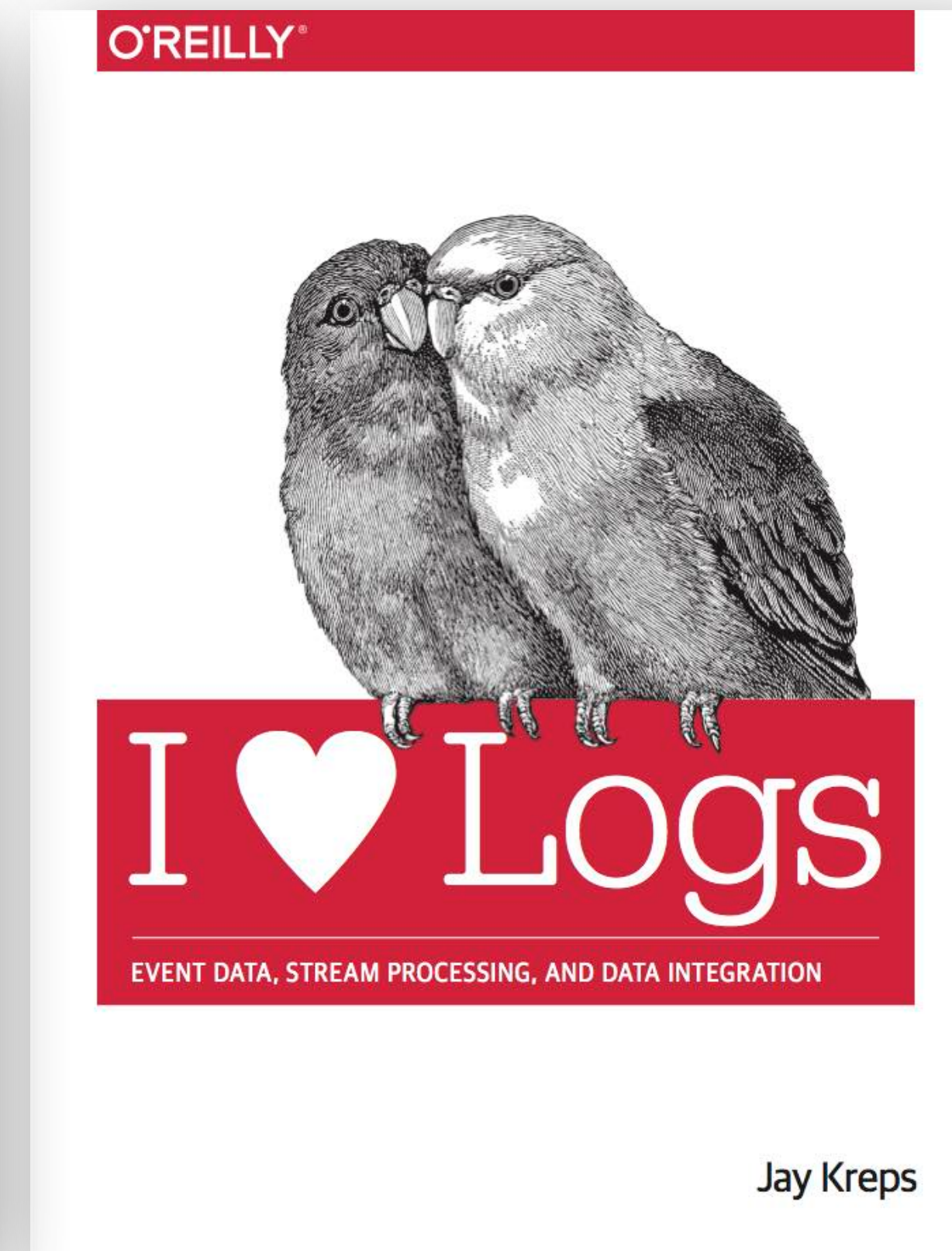
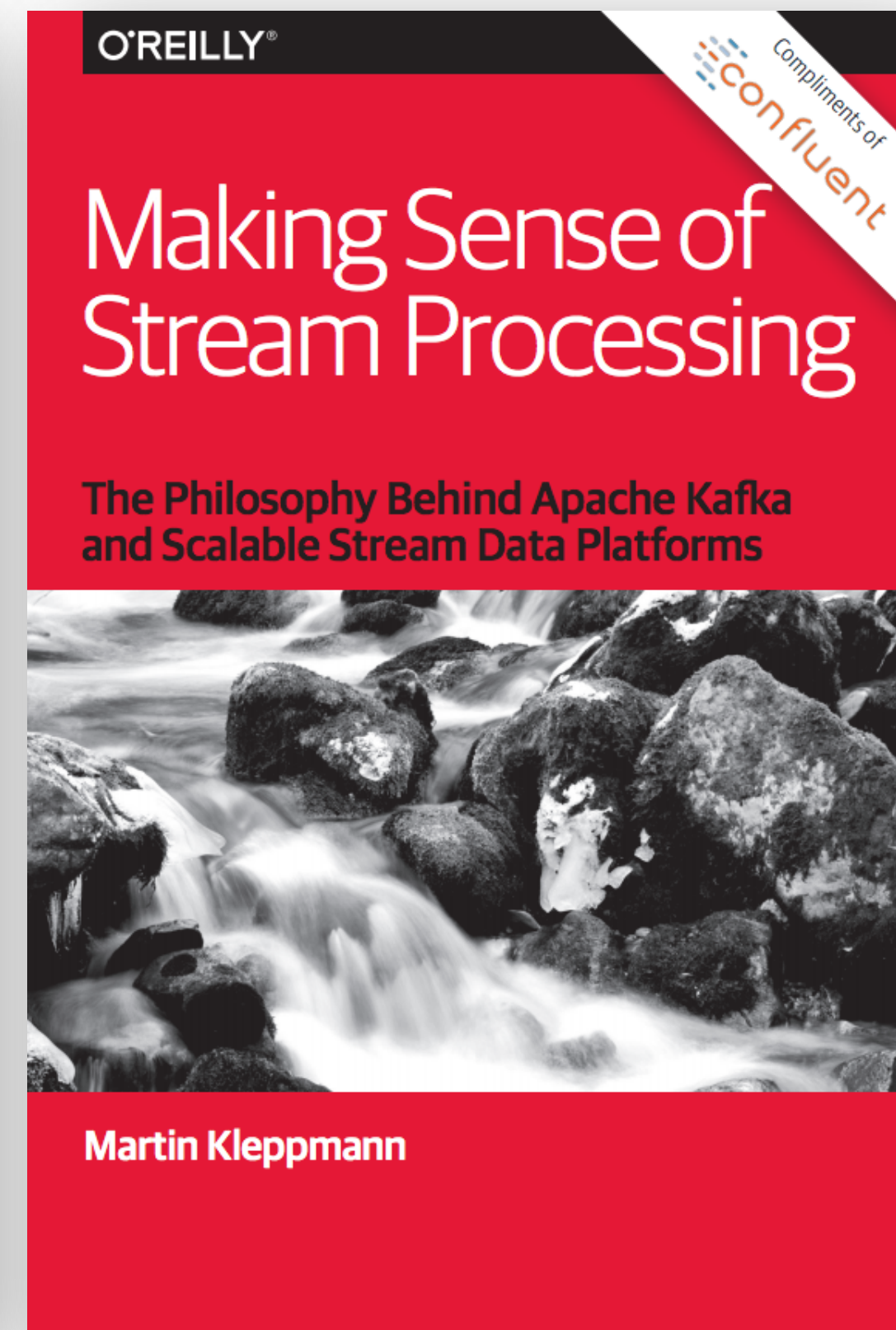
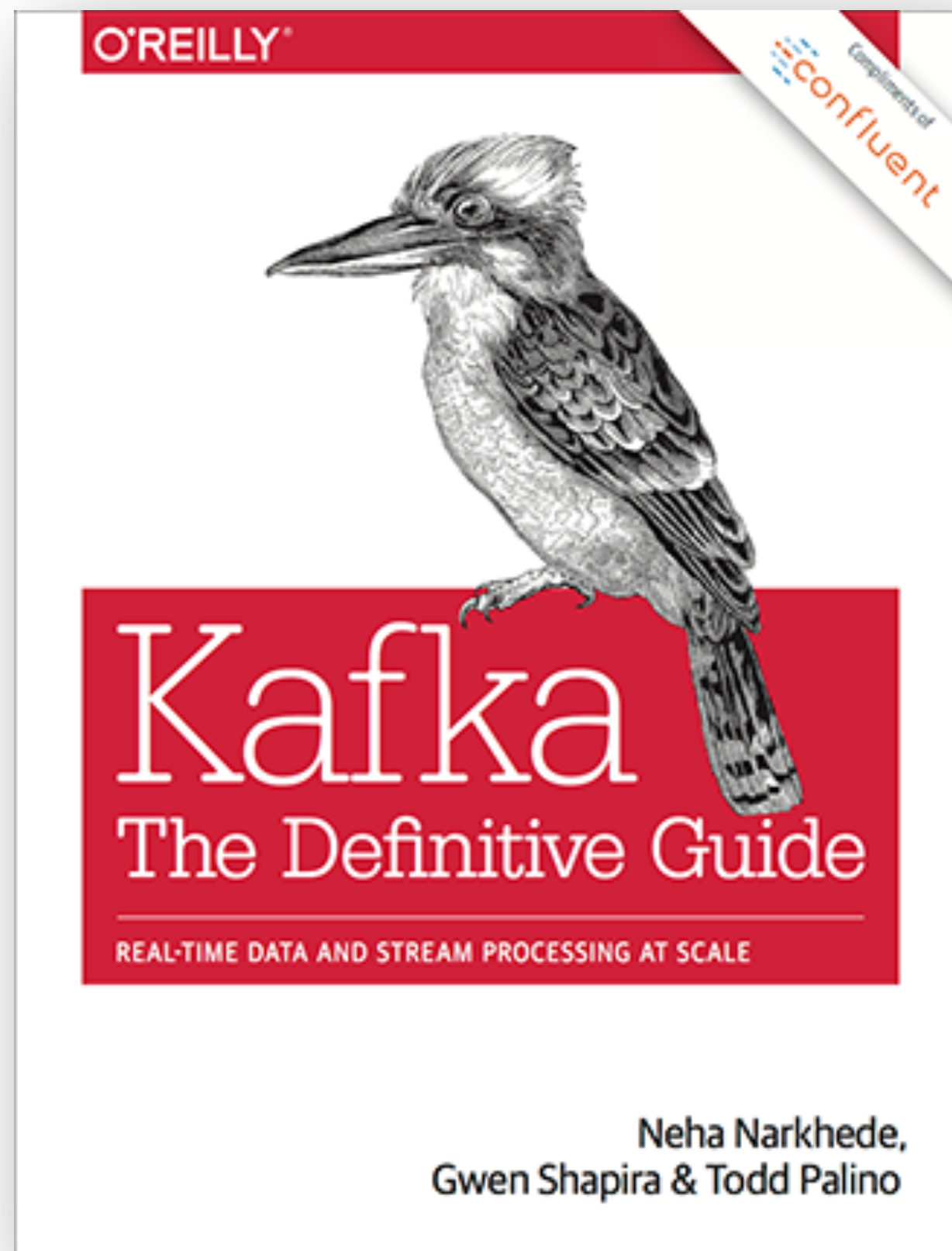


confluent cloud

Fully Managed Kafka as a Service

Free Books!
<http://cnfl.io/book-bundle>

@rmoff #KafkaMeetup



@rmoff

#KafkaMeetup

#EOF



Join the Confluent Community Slack group at <http://cnfl.io/slack>

<https://talks.rmoff.net>