

BUILDING REALTIME DATA PIPELINES WITH KAFKA CONNECT AND SPARK STREAMING

Ewen Cheslack-Postava
Confluent



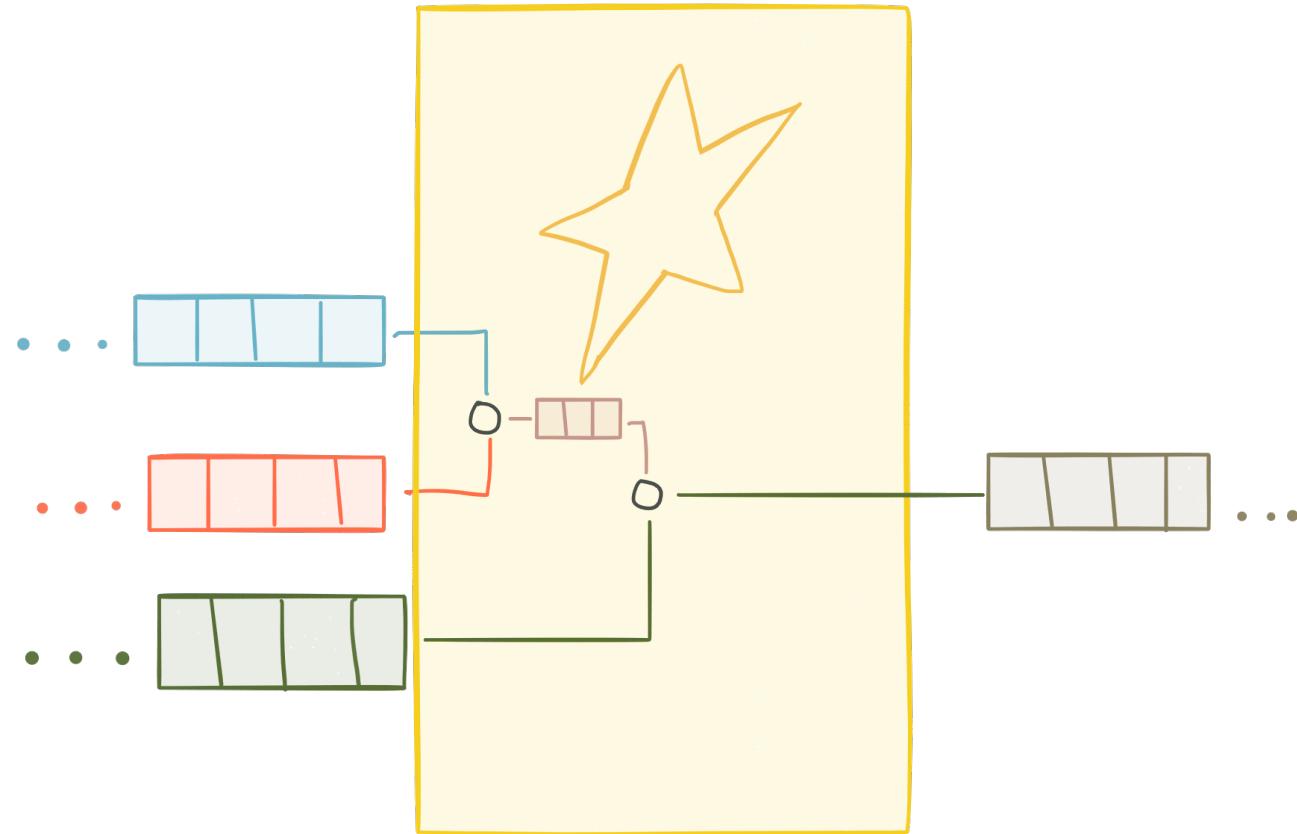
ABOUT ME

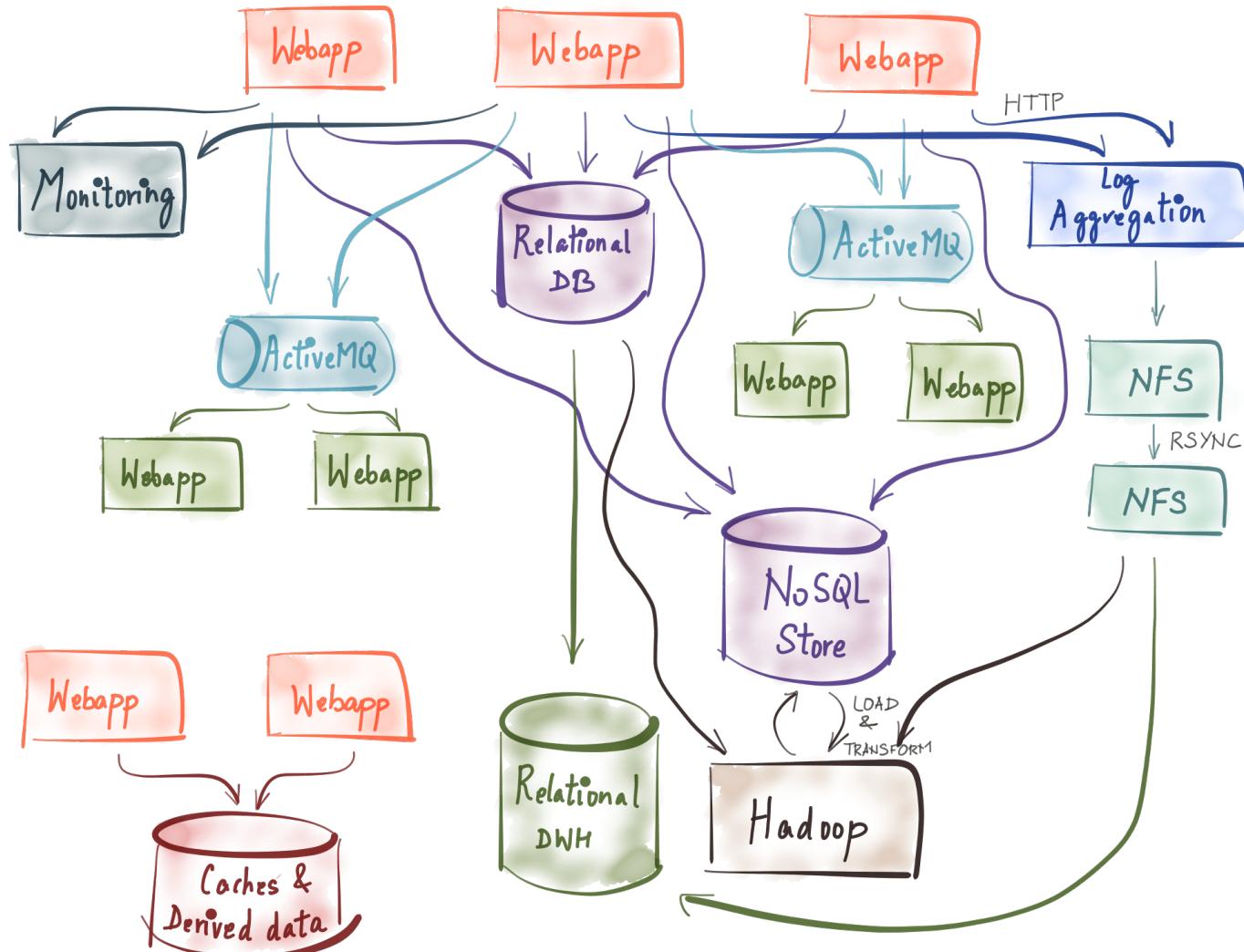
- Engineer @ Confluent
- Kafka Committer
- Kafka Connect Lead





Stream Processing





ANTI PATTERNS

1. One-off tools
2. Kitchen sink tools
3. Stream processing frameworks

Ad-hoc ← ? → E,T,&L

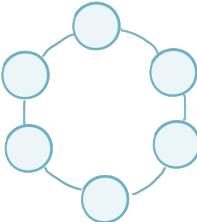


Introducing Kafka Connect

Large-scale streaming data import/export for Kafka

GOALS

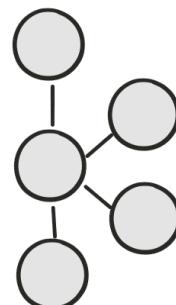
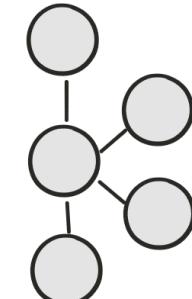
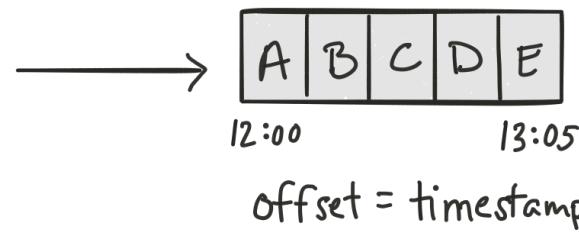
1. Focus on copying
2. Batteries included
3. Standardize
4. Parallelism
5. Scale



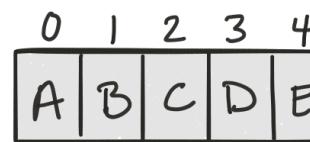
Table

TS	Data
12:00	A
12:20	B
12:30	C
13:00	D
13:05	E

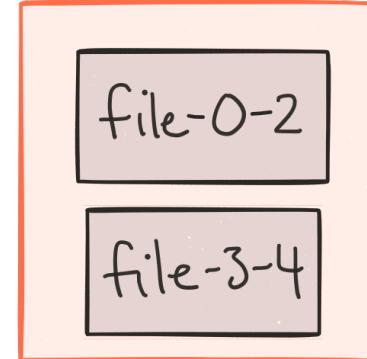
Database Source



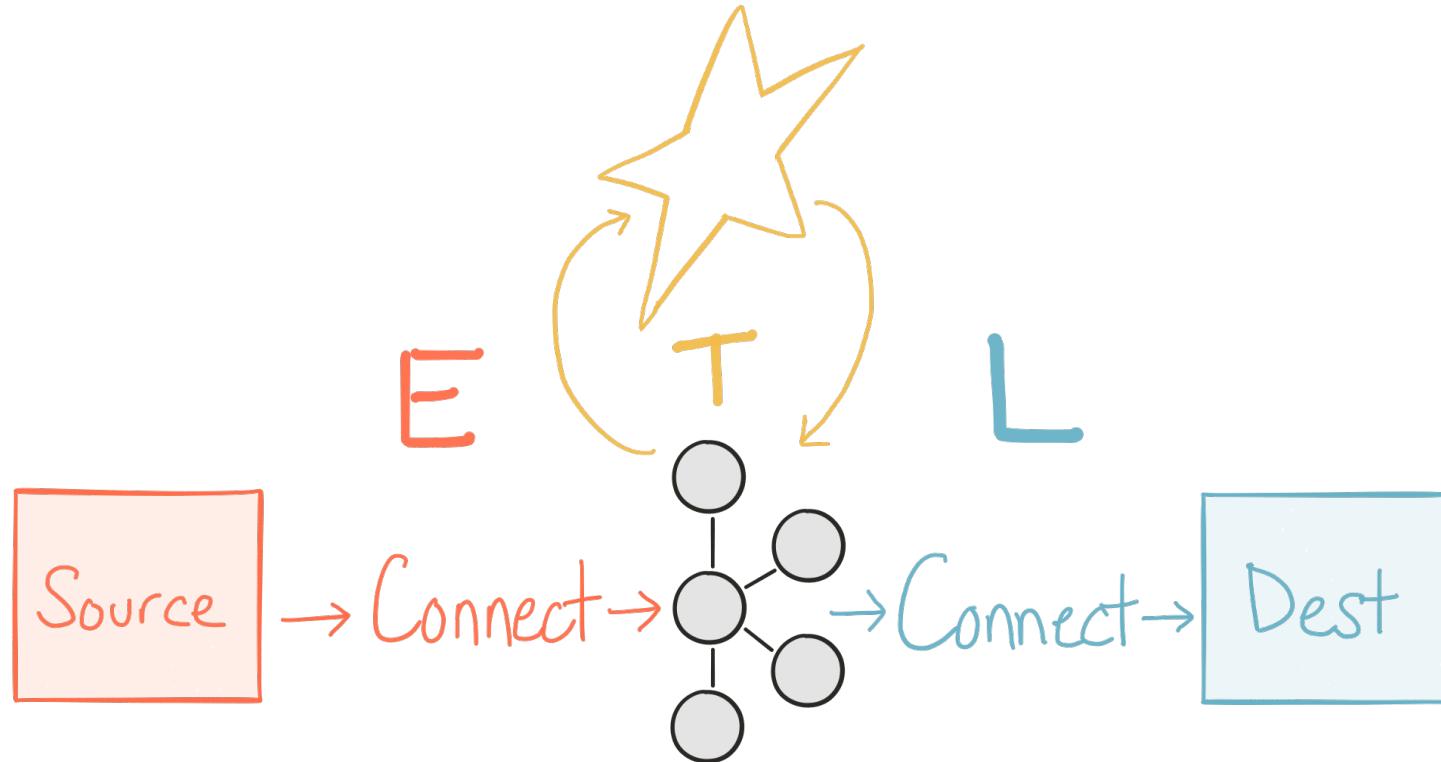
HDFS Sink



HDFS Directory



Separation of Concerns



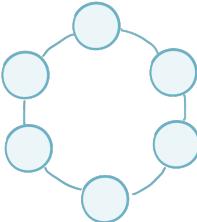
SPARK STREAMING & KAFKA

INPUT: DIRECT KAFKA STREAMS

- OLD CONSUMER (0.8.2.1+) - SPARK 2.0+

- NEW CONSUMER (0.10.0.0+) - EXPERIMENTAL

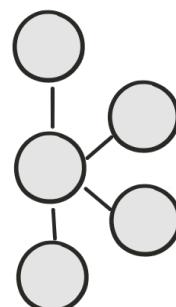
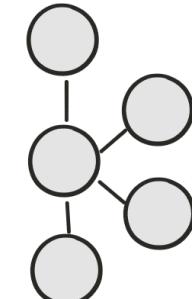
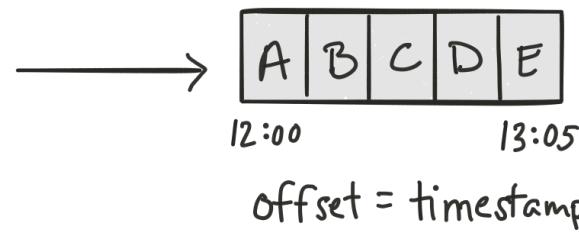
OUTPUT: SPARK KAFKA WRITER



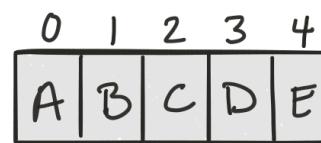
Table

TS	Data
12:00	A
12:20	B
12:30	C
13:00	D
13:05	E

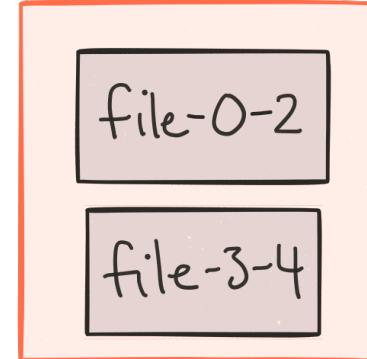
Database Source



HDFS Sink

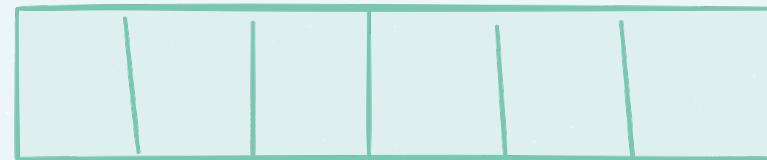


HDFS Directory



Partitioned Stream

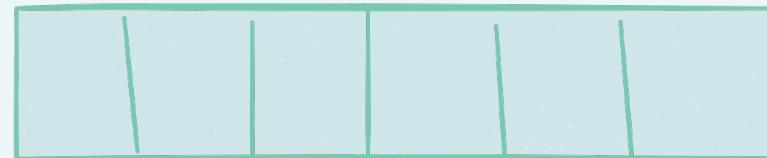
Partition 1



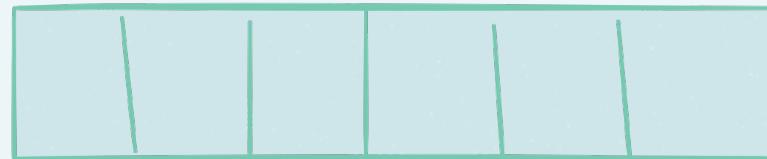
Partition 2



Partition 3



Partition 4



0₁ 0₂ 0₃ 0₄ 0₅ 0₆ ←
offsets

Partitioned Stream - Database

Table 1

--	--	--	--	--	--	--

Table 2

--	--	--	--	--	--	--

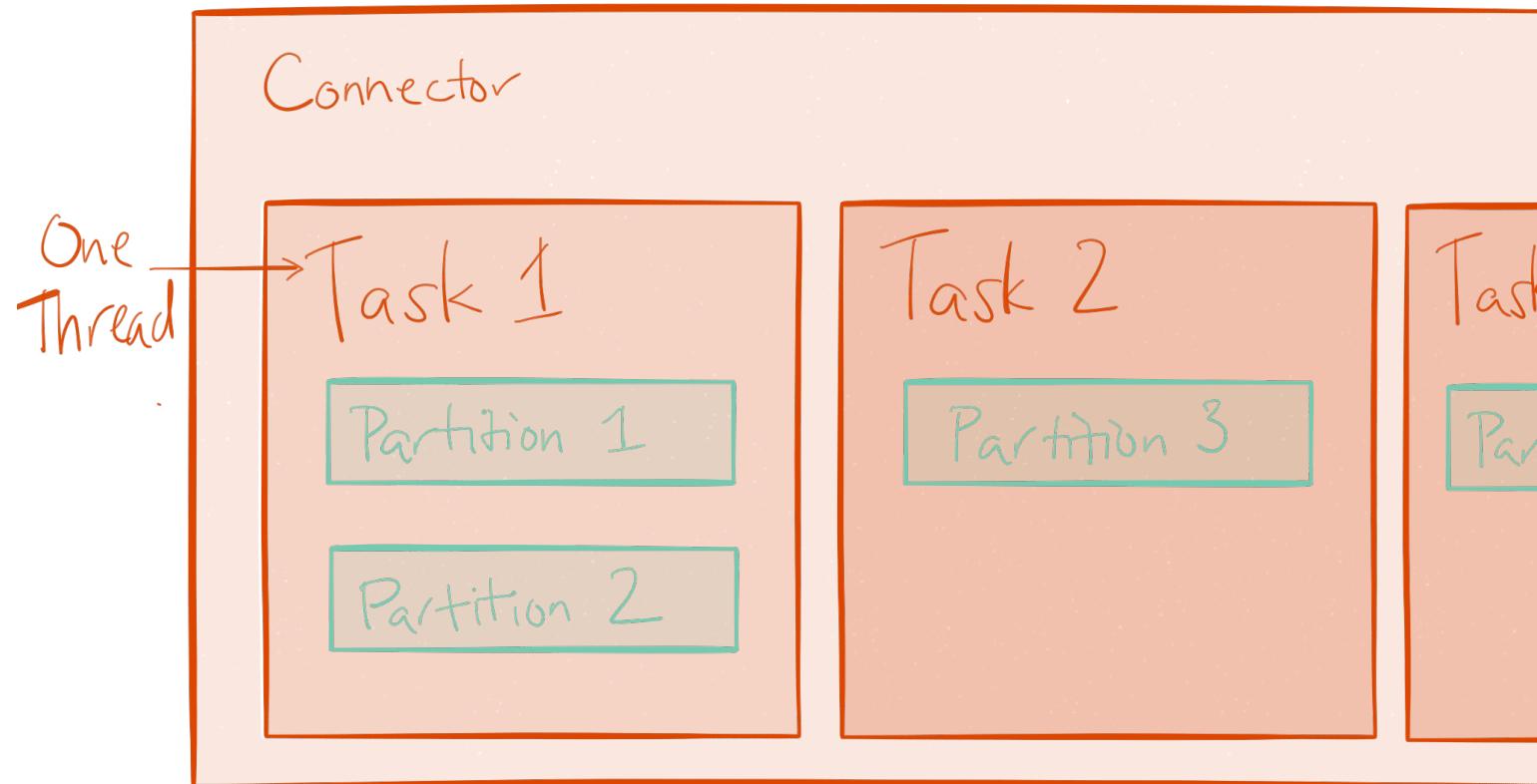
Table 3

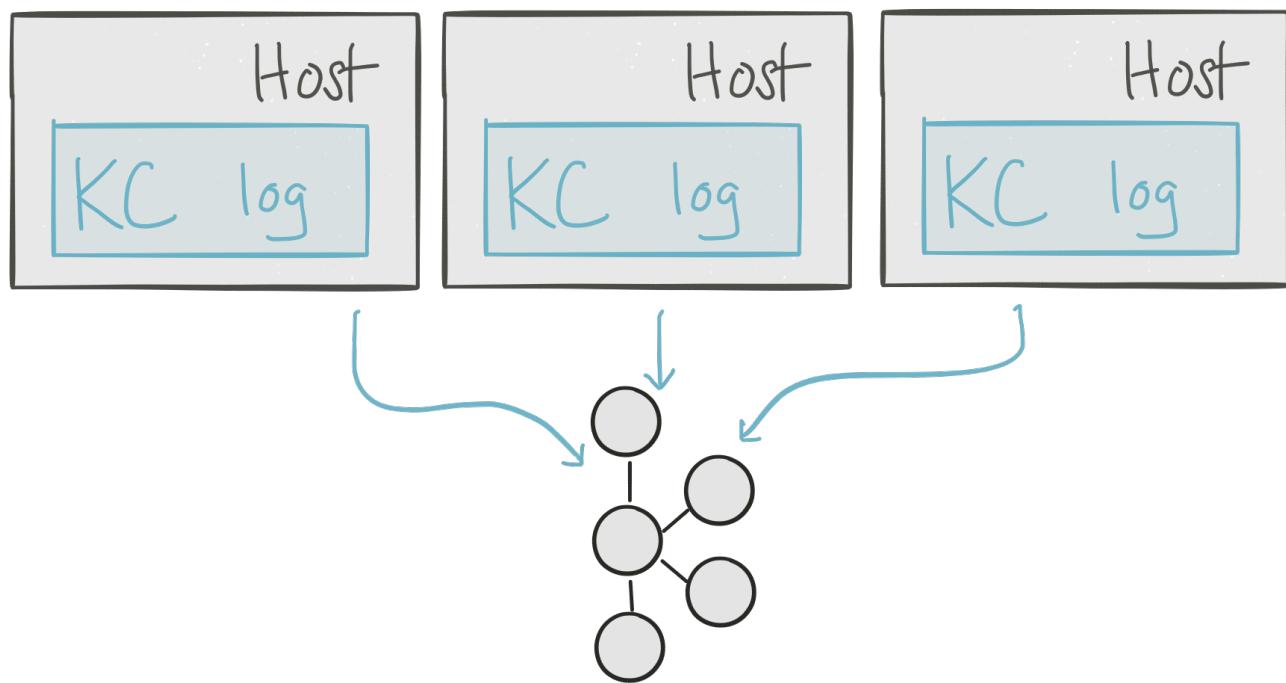
--	--	--	--	--	--	--

Table 4

--	--	--	--	--	--	--

$id=1$ $id=2$ $id=3$ $id=4$ $id=5$ $id=6$





Worker 1

Conn A

Task A1

Task A2

Worker 2

Task A3

Task A4

Worker 3

Conn B

Task B1

Worker 4

Conn C

Task C1

Worker 1

Conn A

Task A1

Task A2

Worker 2

Task A3

Task A4

Worker 3

Conn B

Task B1

Worker 4

Conn C

Task C1

Worker 1

Conn A

Task A1

Task A2

Worker 2

Task A3

Task A4

Conn B

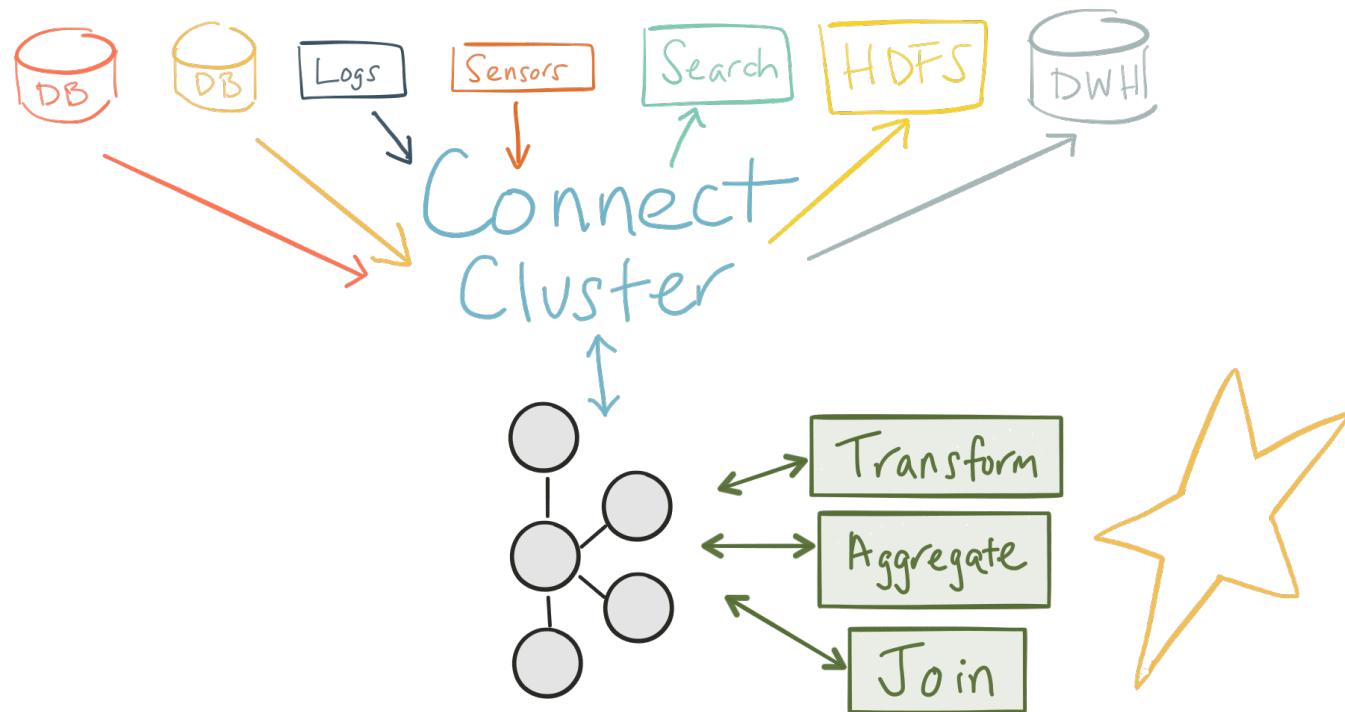
Worker 3

Task B1

Conn C

Conn C1

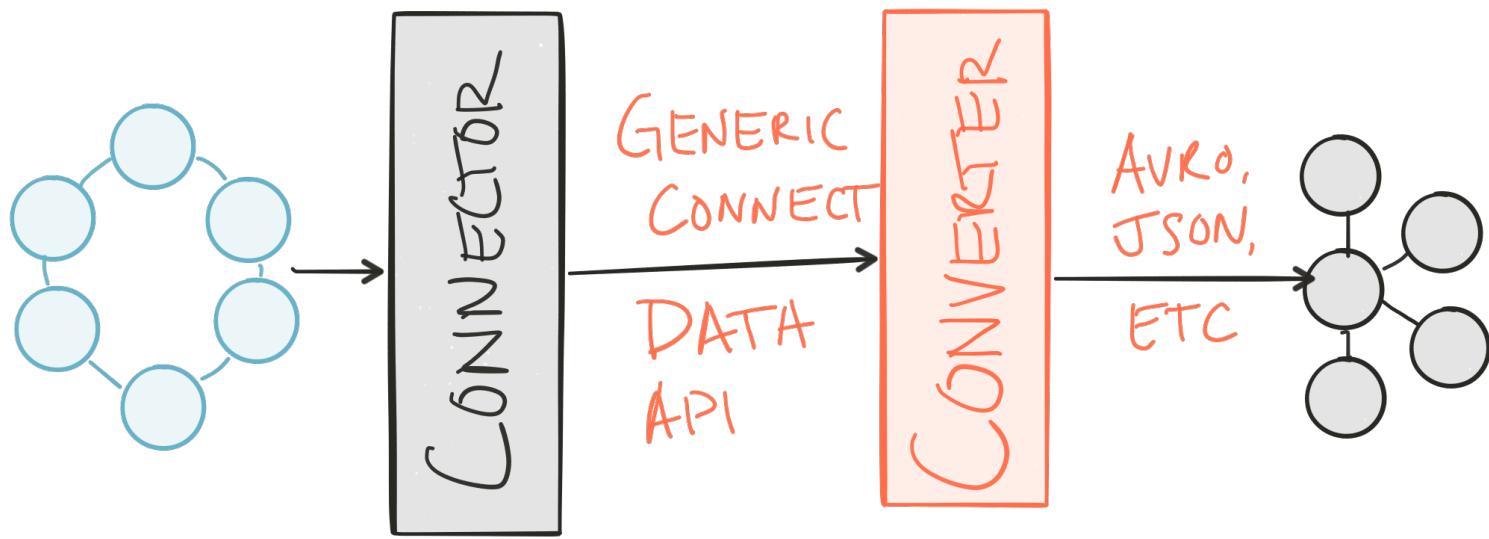
Data Integration as a Service



DELIVERY GUARANTEES

- FRAMEWORK MANAGED OFFSETS
- AT LEAST ONCE DEFAULT
- EXACTLY ONCE (w/ CONNECTOR SUPPORT)

CONVERTERS



SPARK & KAFKA CONNECT

MORE SPARK STREAMING SOURCES+SINKS

REDUCE ADOPTION FRICTION

LEVERAGE KAFKA AS DE FACTO STREAMING

STORAGE ENGINE



COMING SOON...

IMPROVED KAFKA CLIENT COMPATIBILITY

EXACTLY ONCE SEMANTICS

[Product](#)[Services](#)[Resources](#)[About](#)[Blog](#)[Download](#)

Certified Connectors

Certified Connectors have been developed by vendors and/or Confluent utilizing the Kafka Connect framework. These Connectors have met criteria for code development best practices, schema registry integration, security, and documentation.

CONNECTOR	TAGS	DEVELOPER	SUPPORT
HDFS (Sink)	HDFS, Hadoop, Hive	Confluent	Confluent
JDBC (Source)	JDBC, MySQL	Confluent	Confluent
Elastic Search (Sink)	search, Elastic, log, analytics	Confluent	Confluent
DataStax (Sink)	Cassandra, DataStax	Data Mountaineer	Data Mountaineer
Attunity (Source)	CDC	Attunity	Attunity
Couchbase (Source)	Couchbase, NoSQL	Couchbase	Couchbase
GoldenGate (Source)	CDC, Oracle	Oracle	Community
JustOne (Sink)	Postgress	JustOne	JustOne
Striim (Source)	CDC, MS SQLServer	Striim	Striim
Syncsort DMX	DB2, IMS, VSAM, CICS	Syncsort	Syncsort

Want to build a connector?

Interested Open Source Developers and Vendors can get started with the following resources:

> [Kafka Connect Overview](#)

> [Kafka Connect Developers Guide](#)

Already built a connector?

If you have a connector that you'd like to see added to our list, let us know at:

confluent-platform@googlegroups.com

If your connector is well done, we'll send you a free T-shirt and may even list it here.

Contact Us

For software vendors seeking to build a connector, our Partner Engineering team is





THANK YOU

@ewencp

@confluentinc

<http://confluent.io/download/>

<http://connectors.confluent.io>