Introducing Kafka Connect

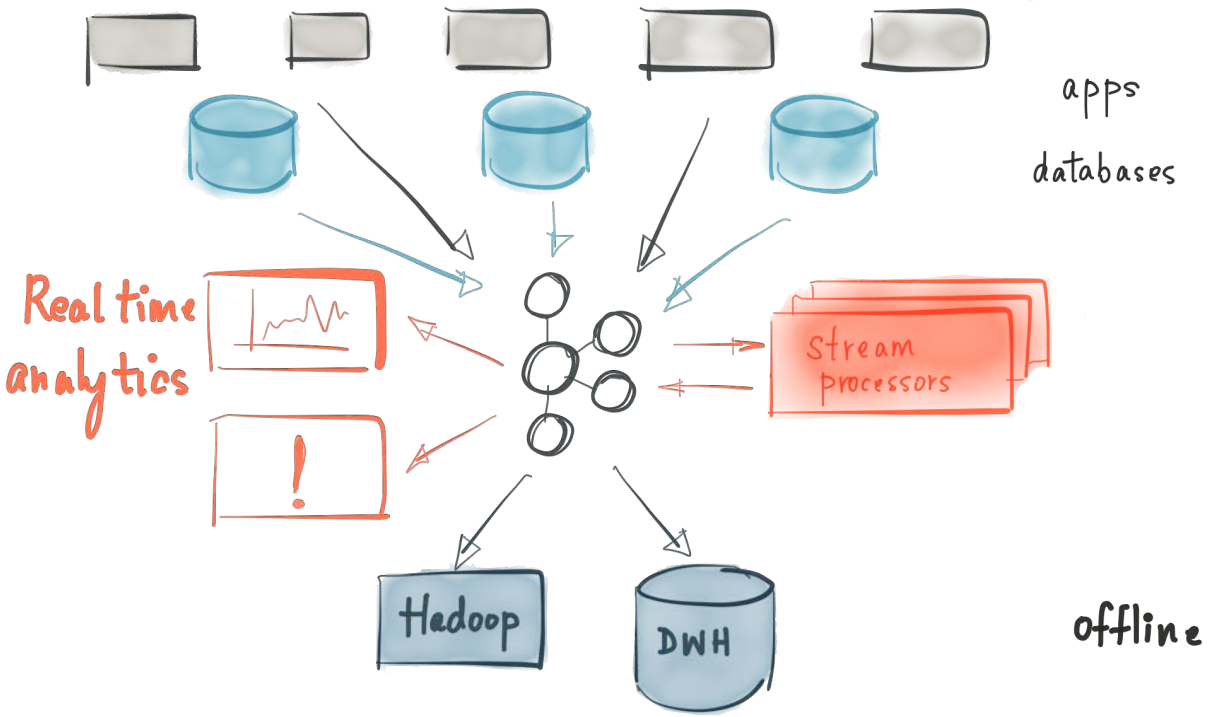
Large-scale streaming data import/export for Kafka

@tlberglund

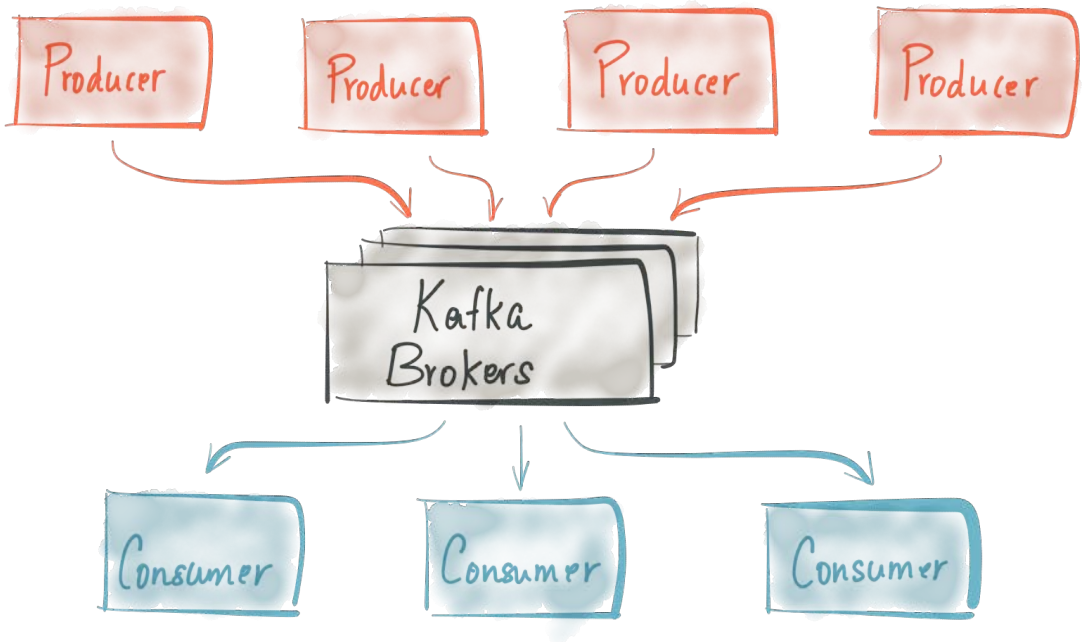
My Secret Agenda

1. Review of Kafka
2. Why do we need Connect?
3. How does Connect work?
4. Tell me about these "Connectors"
5. Single Message Transforms

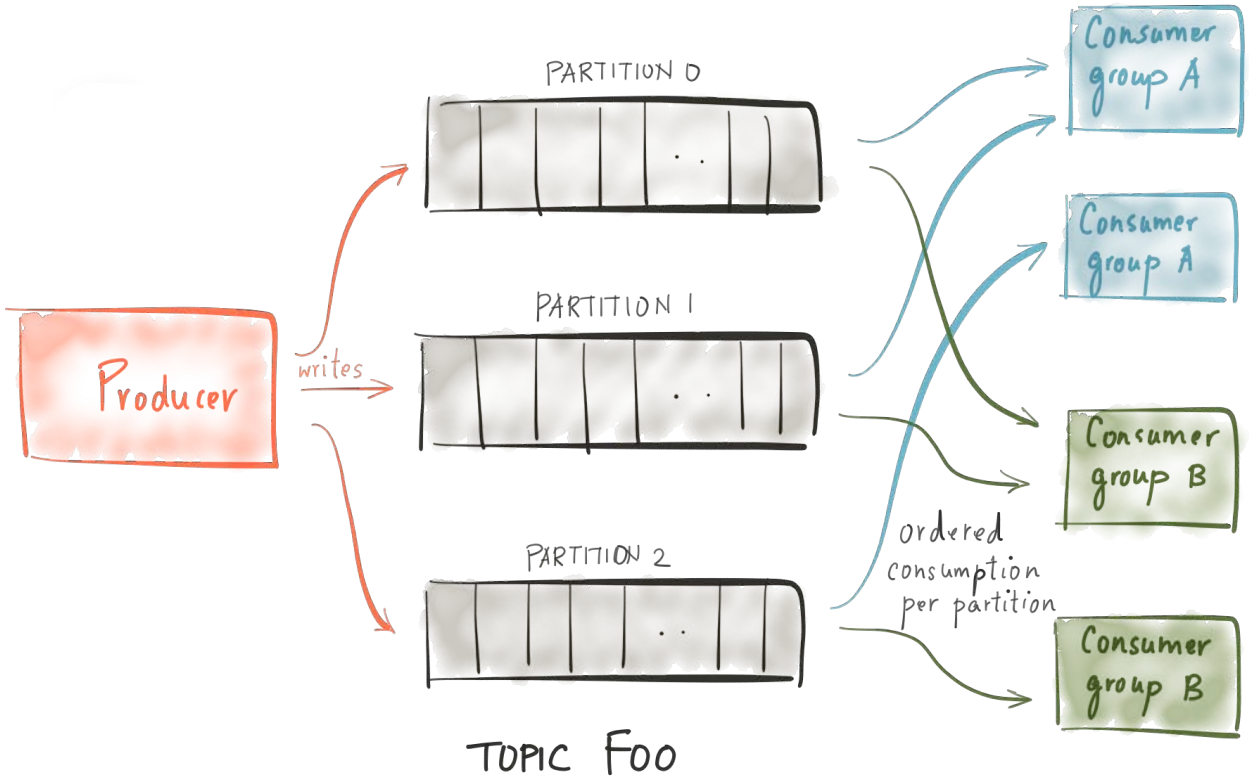
A Streaming Data Platform



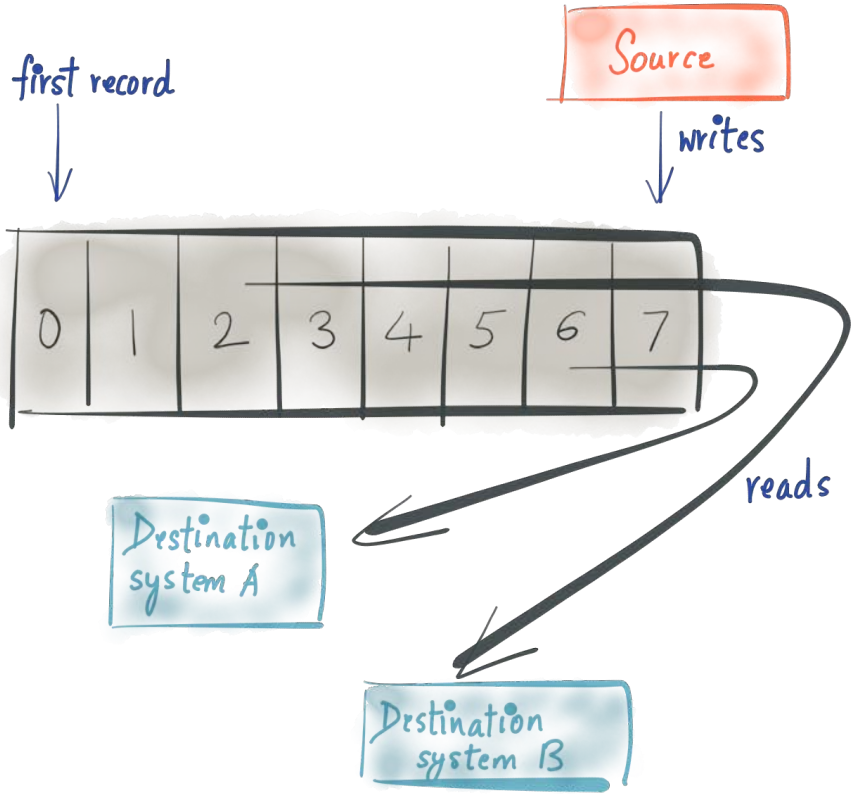
Apache Kafka: 10,000-ft view



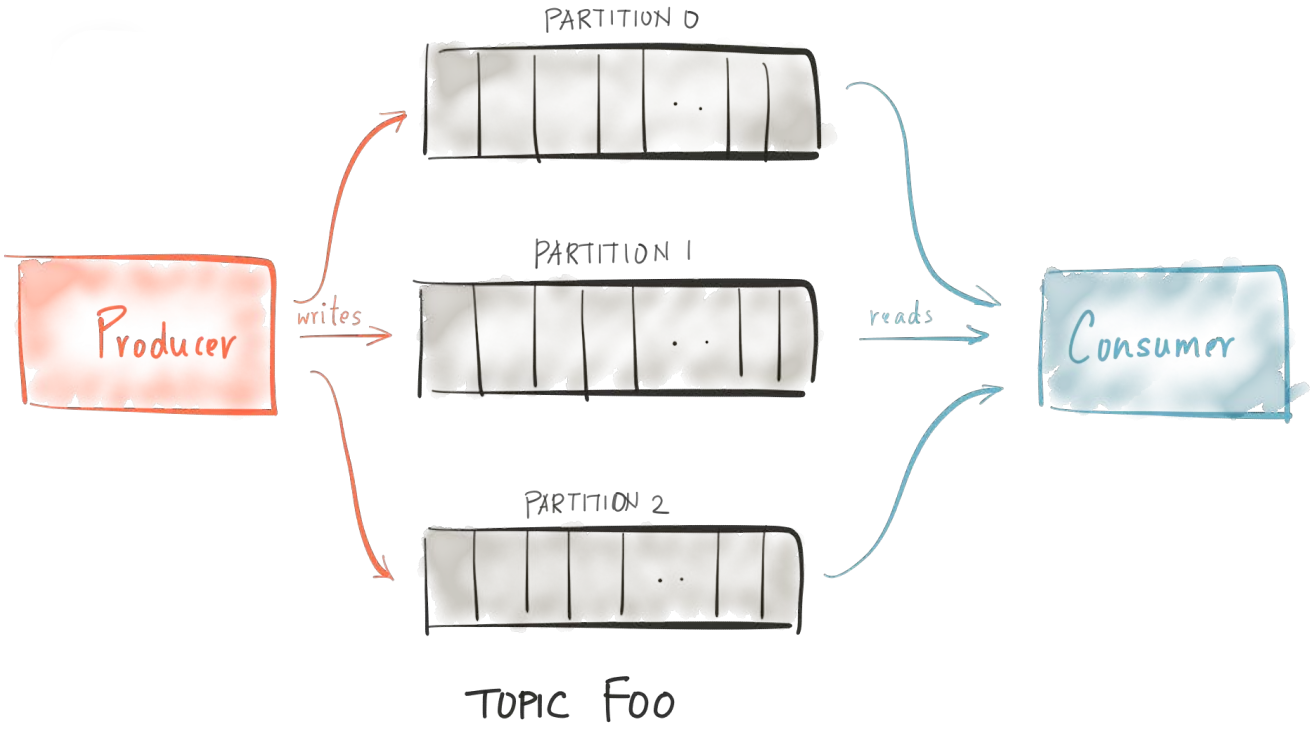
Scalable Consumption



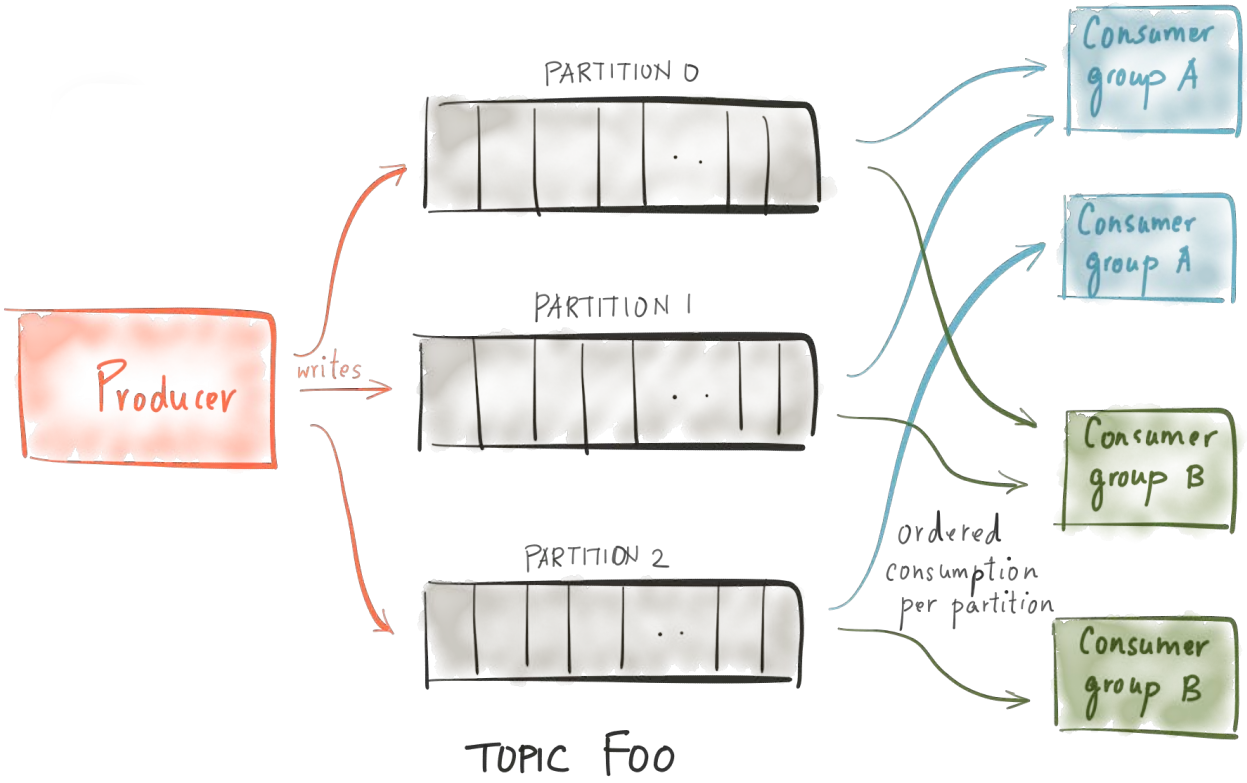
Logs and Pub-Sub



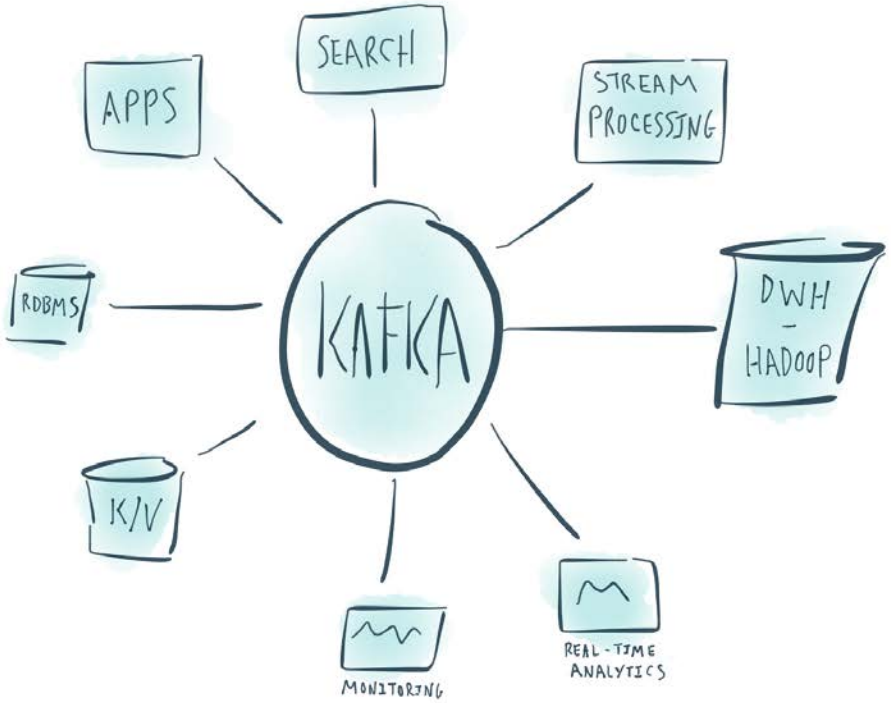
a Kafka Topic is a Partitioned Log



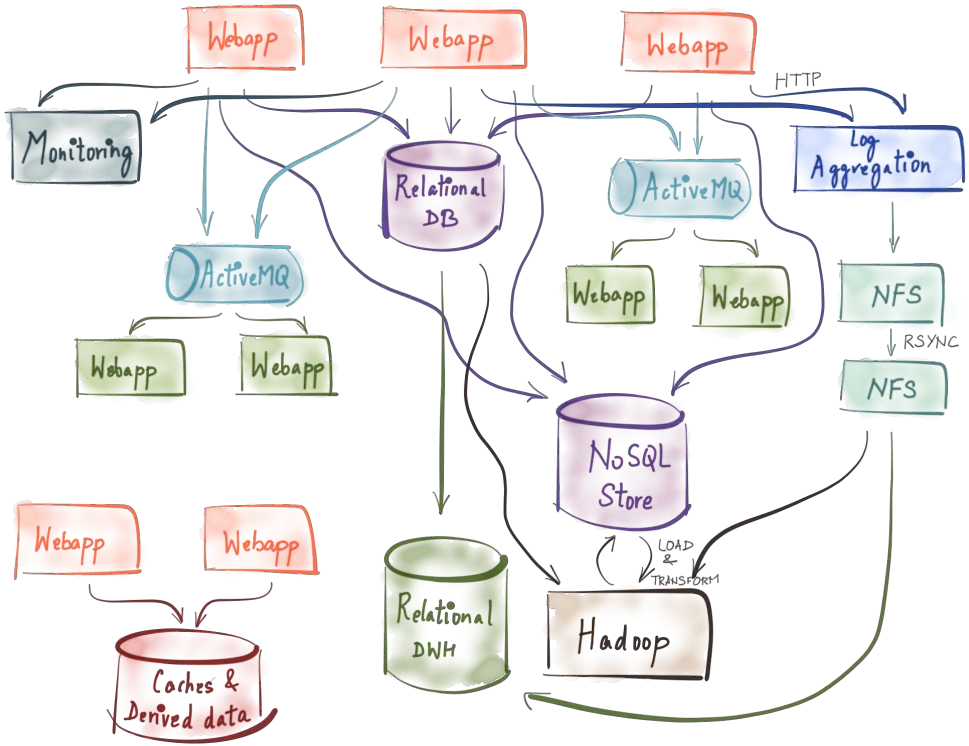
Scalable Consumption



STREAMING PLATFORM

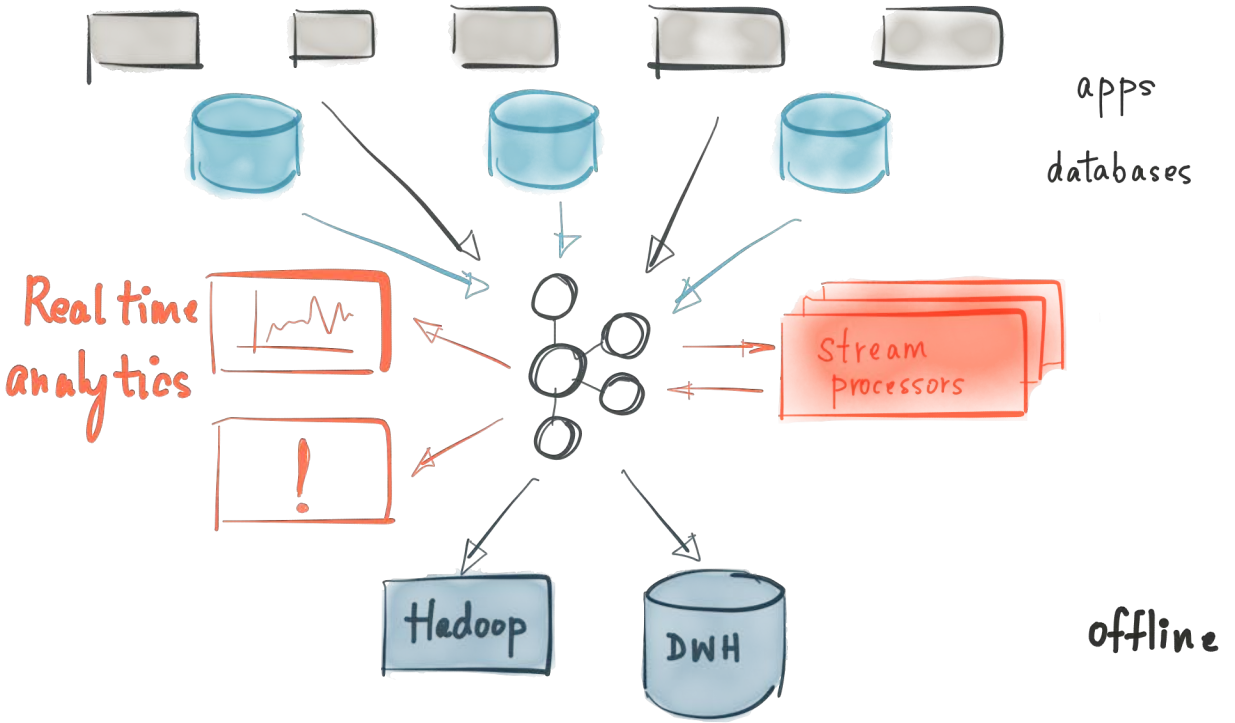


When Streaming Data Pipelines Attack

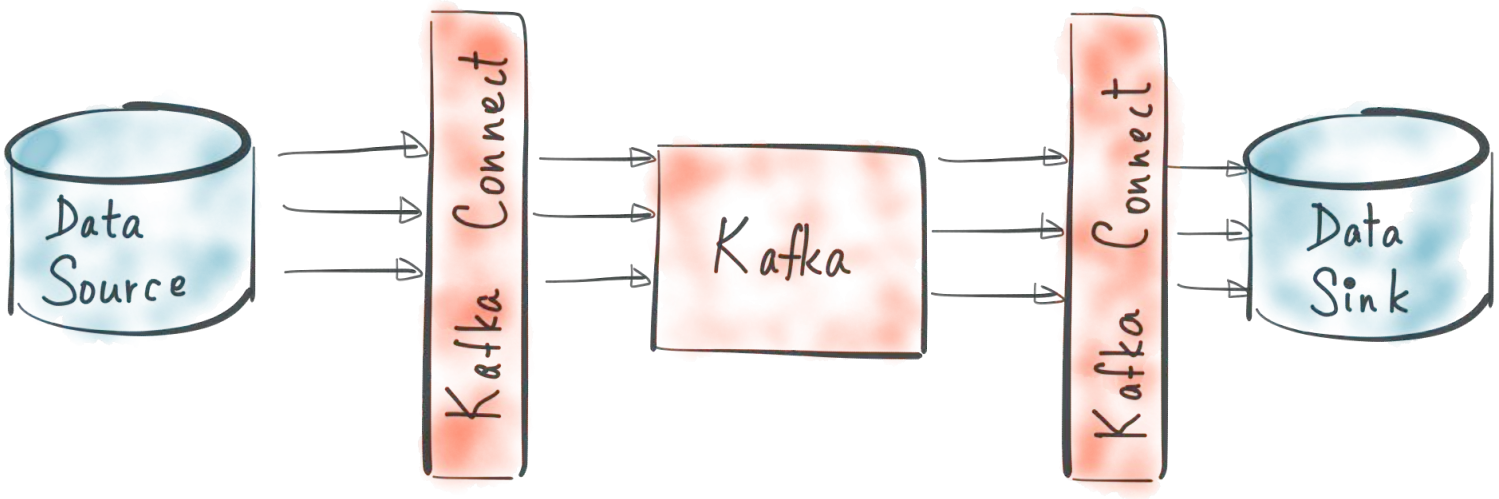


confluent

When Kafka Fights Back



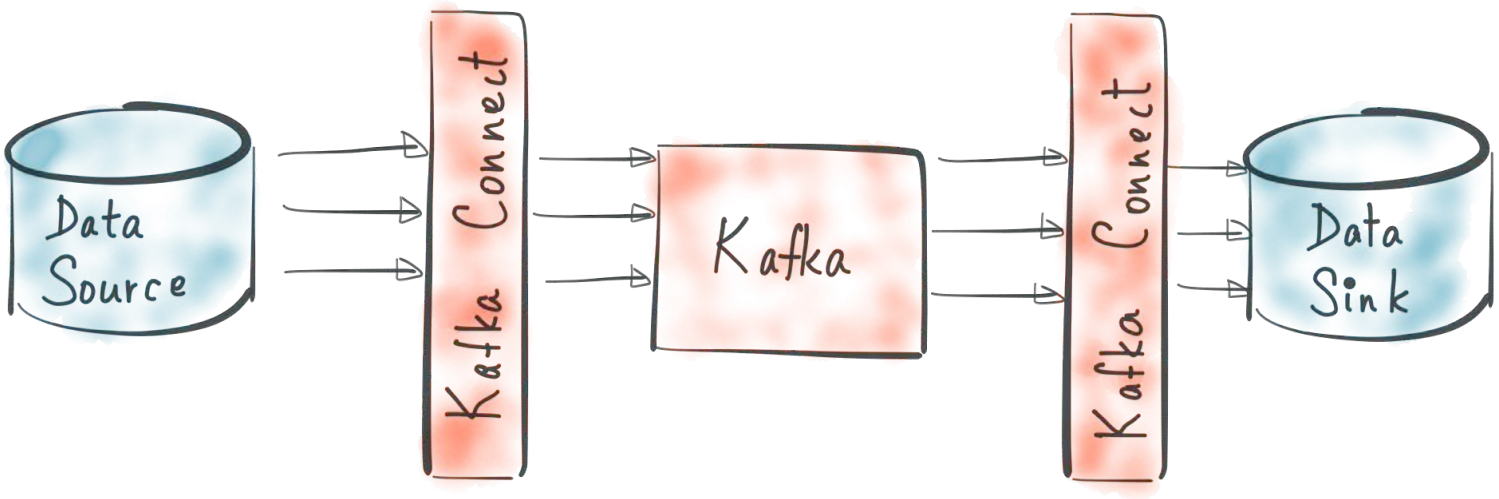
Kafka Connect



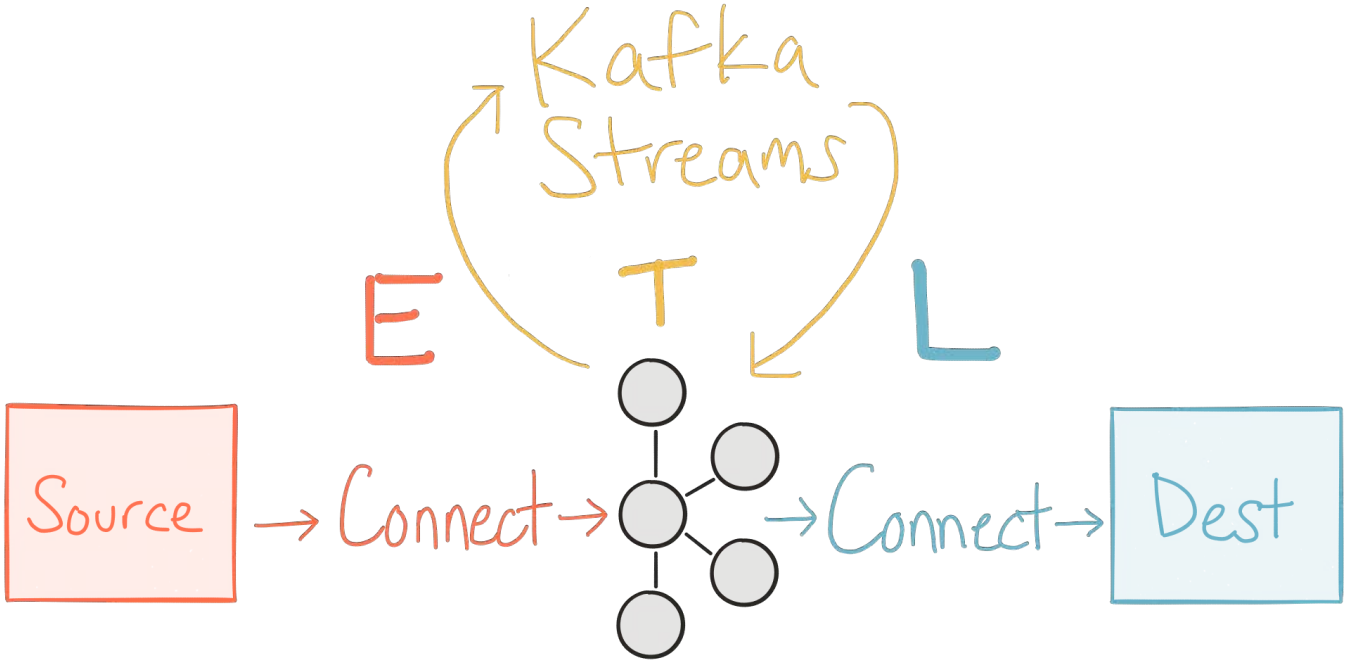
Kafka Connect

- a data integration framework
- scalable and fault-tolerant
- exactly-once delivery in many cases
- integrates Kafka with other data systems
- library of existing "connectors" for common data sources and sinks
- necessary component of modern streaming ETL systems

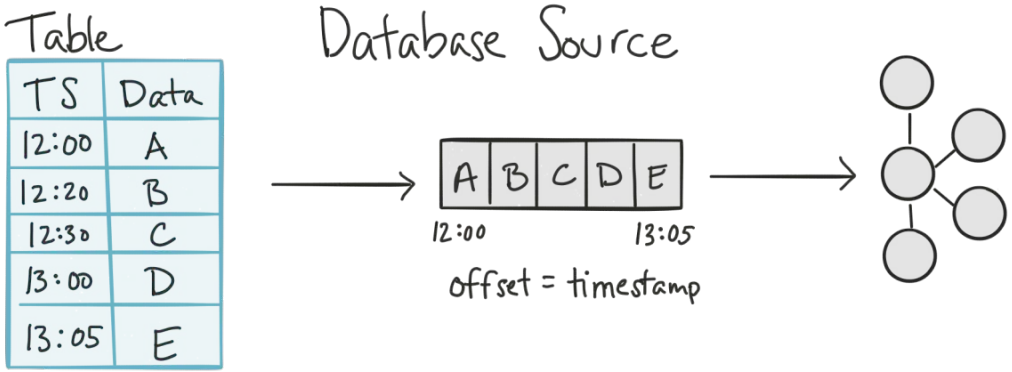
Kafka Connect



Streaming ETL

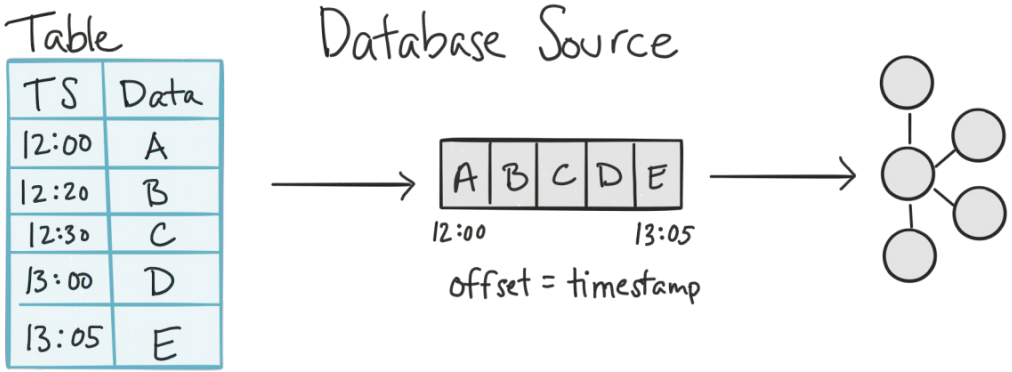


Database Source



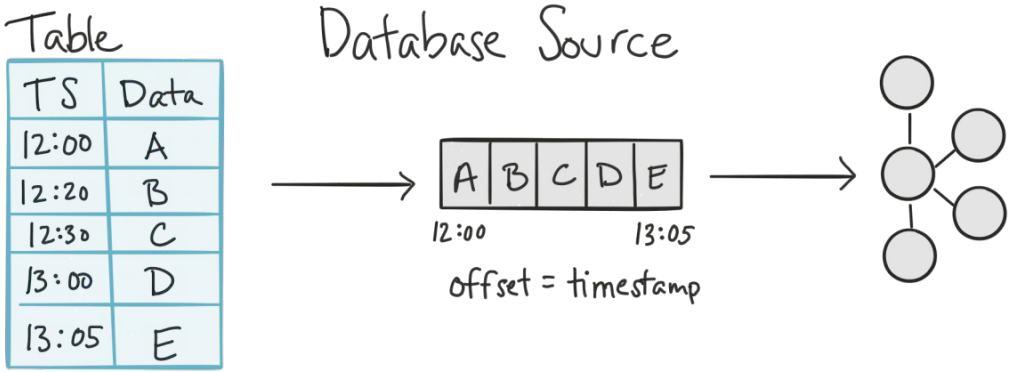
confluent

Database Source



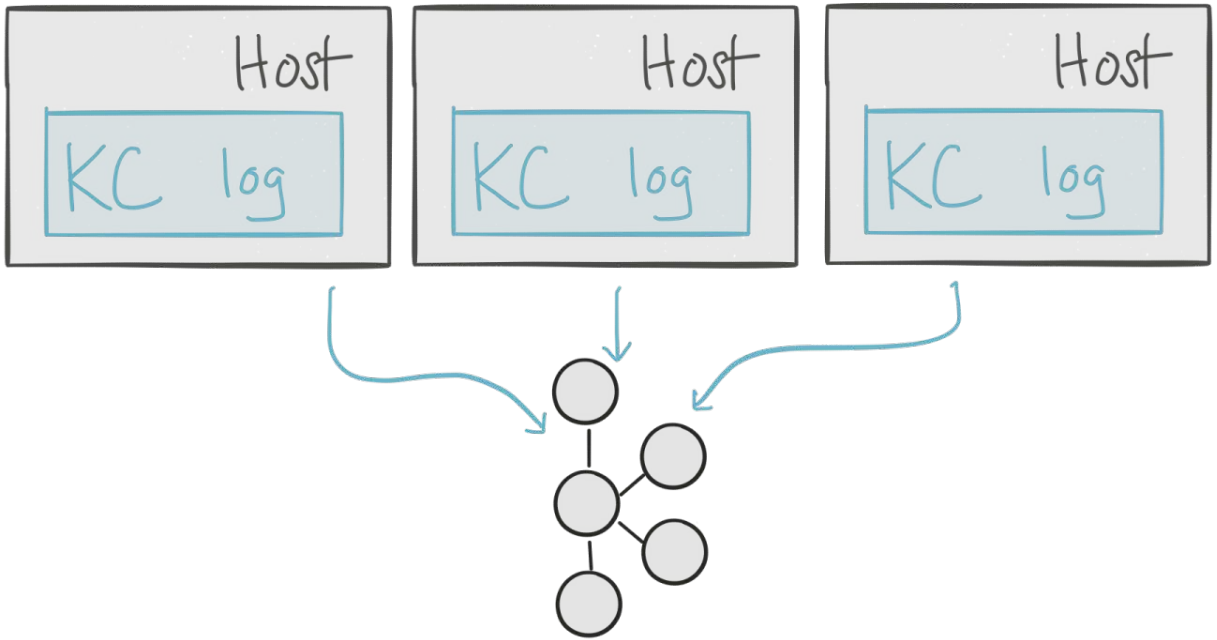
confluent

Database Source

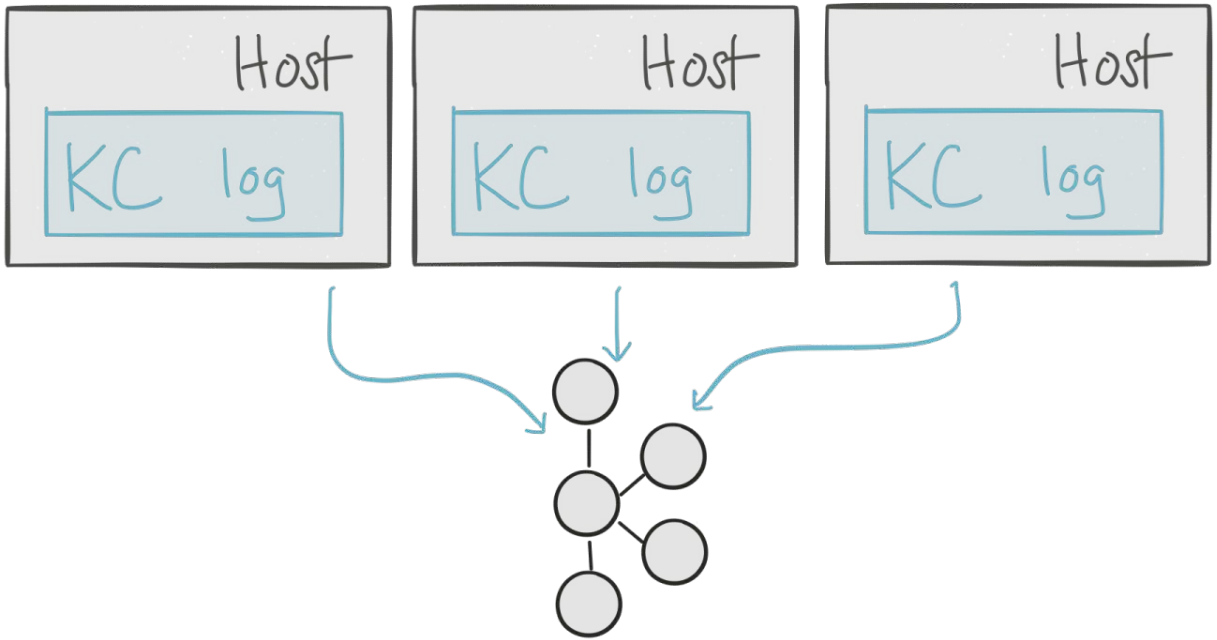


confluent

Standalone Mode



Distributed Mode



Delivery Guarantees

- Framework Managed Offsets
- At Least Once Default
- Exactly Once (w/connector support)

Architectural Components

- Connectors
- Tasks
- Workers
- Converters

Connectors

- a logical job that copies data in and out of Kafka
- maintains tasks
- provides lifecycle and configuration information
- ultimatey a JAR avaiable to the connect JVM
- stateless! uses Kafka topics for state

Tasks

- lifecycle-managed by a connector
- runs in a worker
- manages one or more topic partitions
- (that assignment is dynamic at runtime)
- does the actual copying and transforming of things

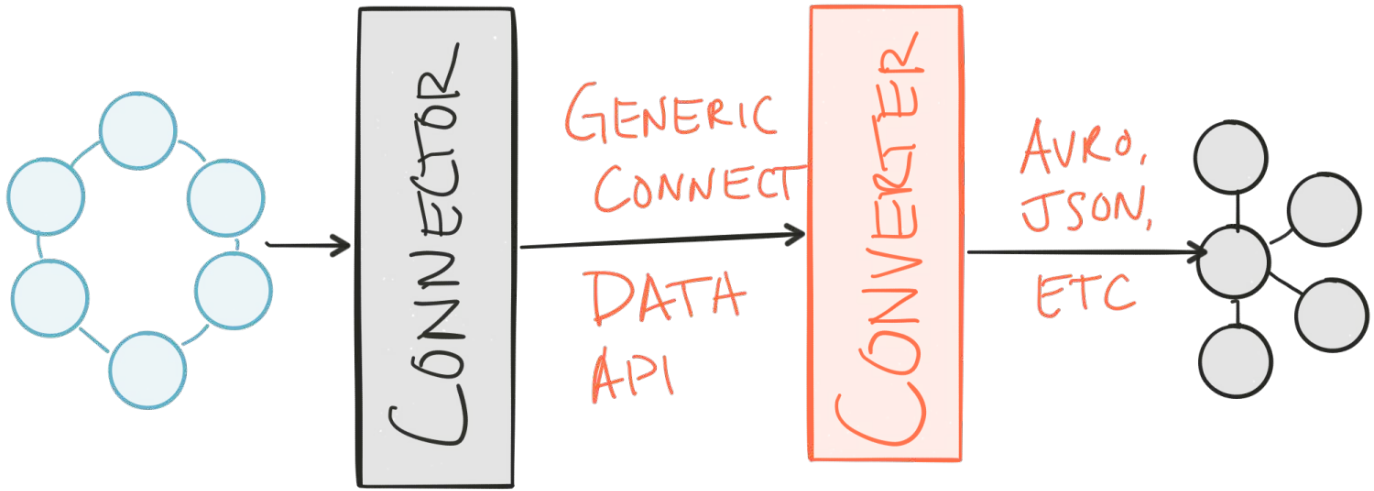
Workers

- actual JVM processes running on a computer of some kind
- tasks get allocated to workers
- can run in standalone or distributed mode
- standalone good for dev, one-offs, and conference demos
- distributed good for scale and fault-tolerance

Converters

- convert input data to bytes for consumption by Kafka
- or from bytes to output to somewhere else
- Sits between connector and Kafka in either direction

Converters


CONNECTOR
GENERIC CONNECT
DATA API
CONVERTER
AVRO, JSON, ETC

Use Cases

- Getting data to/from HDFS, Elastic, Cassandra, S3, MongoDB, SAP HANA, some dang relational database, some dang NoSQL database
- You're building a Kafka future, but have legacy systems lol
- Need to integrate with system X, but *cannot alter it*
- Space-age real-time systems need to know when a relational DB chagnes
- Actually building next-gen, real-time ETL like a boss

Certified Connectors

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

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?

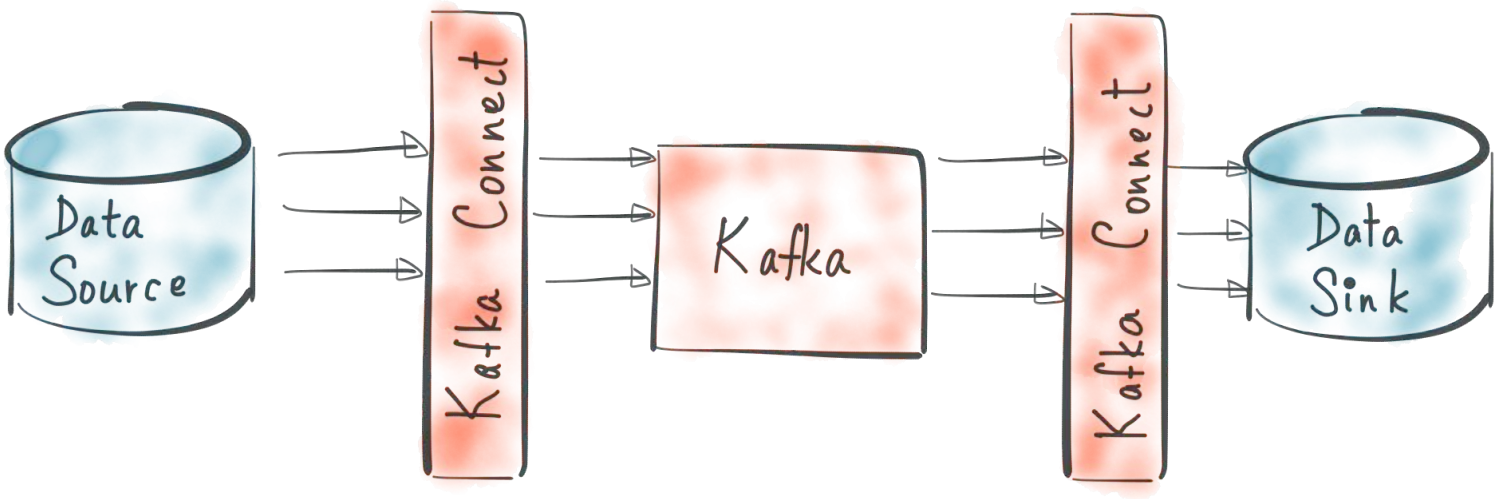
If you have a connector that you'd like to see added to our list, let us know at: confluent-platform@googlegroups.com

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

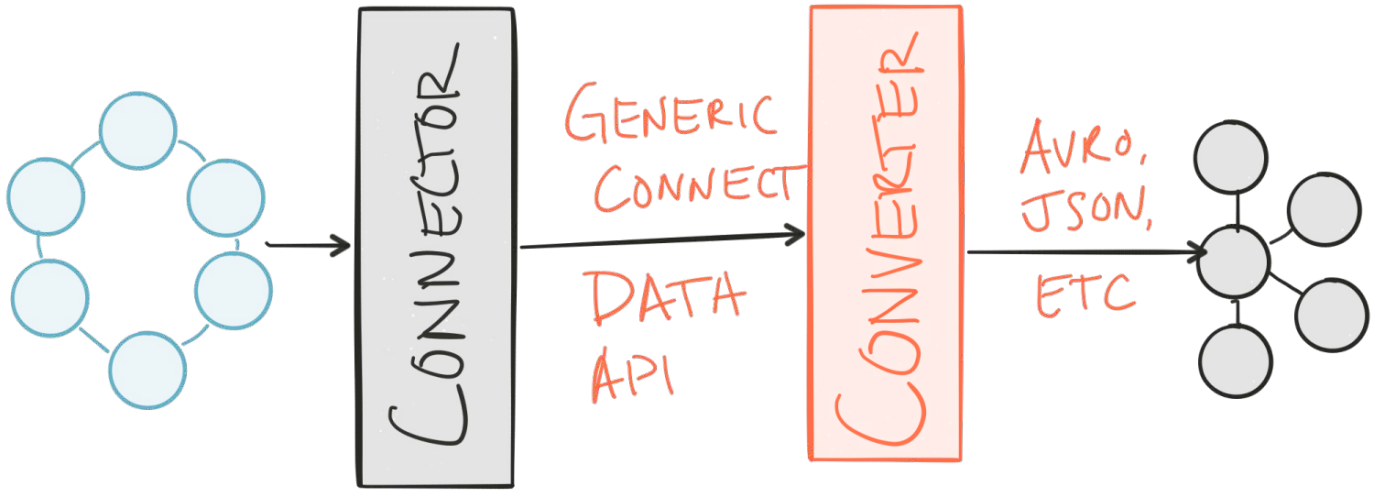
Contact Us

confluent

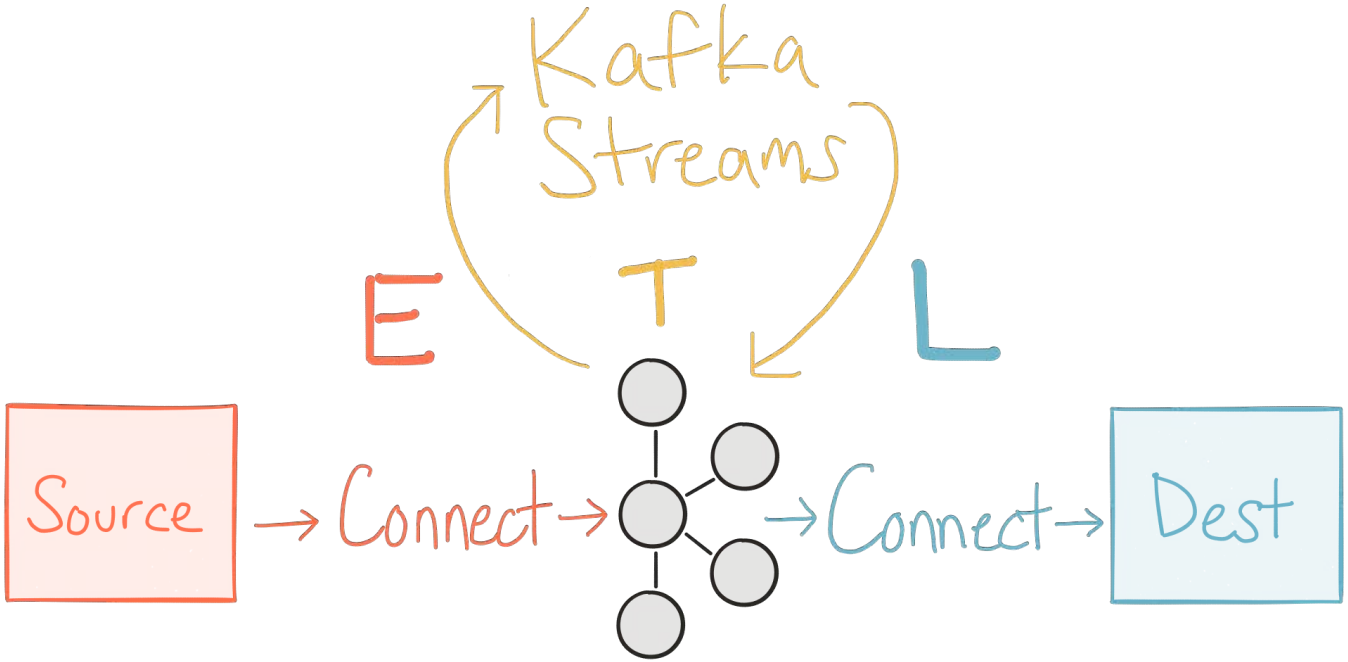
Kafka Connect



Converters


CONNECTOR
GENERIC CONNECT
DATA API
CONVERTER
AVRO, JSON, ETC

Streaming ETL



Single Message Transformations for Kafka Connect

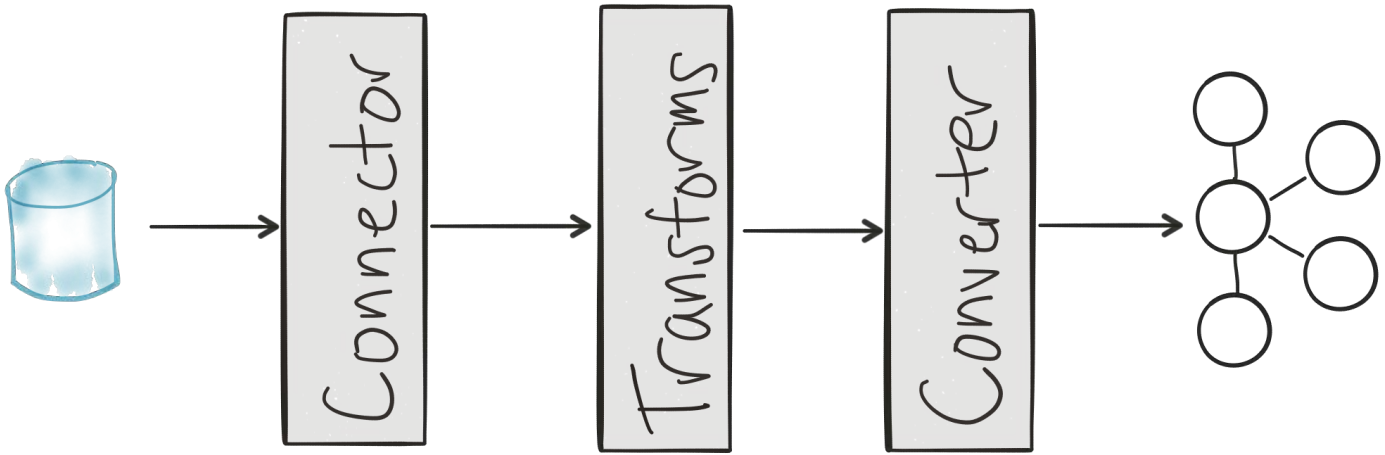
Modify events before storing in Kafka:

- Mask sensitive information
- Add identifiers
- Tag events
- Store lineage
- Remove unnecessary columns

Modify events going out of Kafka:

- Route high priority events to faster data stores
- Direct events to different Elasticsearch indexes
- Cast data types to match destination
- Remove unnecessary columns

Single Message Transformations for Kafka Connect



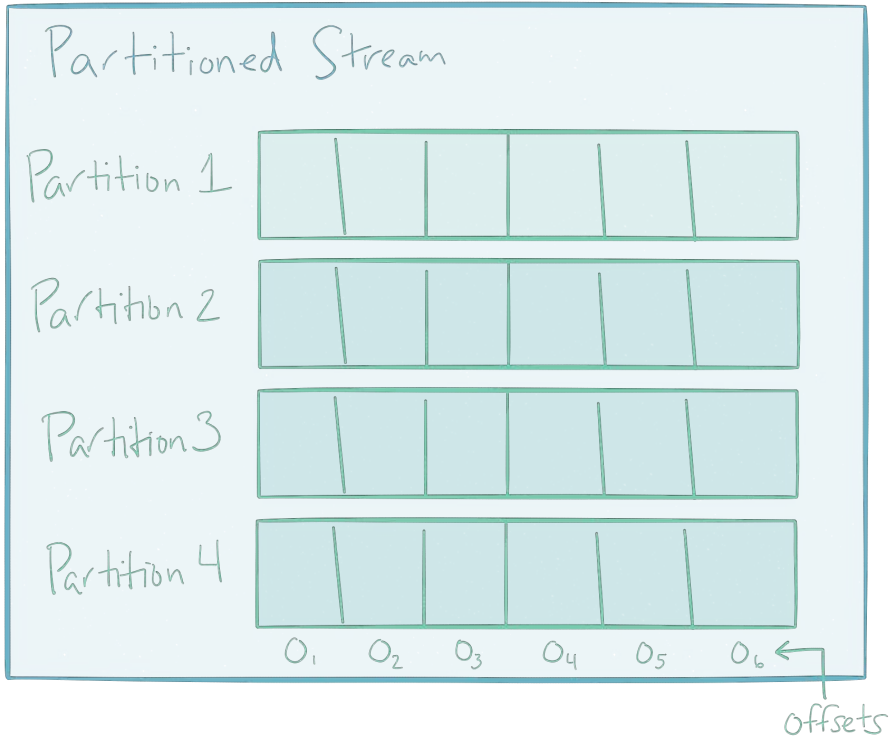
Built-in Transformations

- `InsertField` – Add a field using either static data or record metadata
- `ReplaceField` – Filter or rename fields
- `MaskField` – Replace field with valid null value for the type (0, empty string, etc)
- `ValueToKey` – Set the key to one of the value's fields
- `HoistField` – Wrap the entire event as a single field inside a Struct or a Map
- `ExtractField` – Extract a specific field from Struct and Map and include only this field in results
- `SetSchemaMetadata` – modify the schema name or version
- `TimestampRouter` – Modify the topic of a record based on original topic and timestamp. Useful when using a sink that needs to write to different tables or indexes based on timestamps
- `RegexpRouter` – modify the topic of a record based on original topic, replacement string and a regular expression

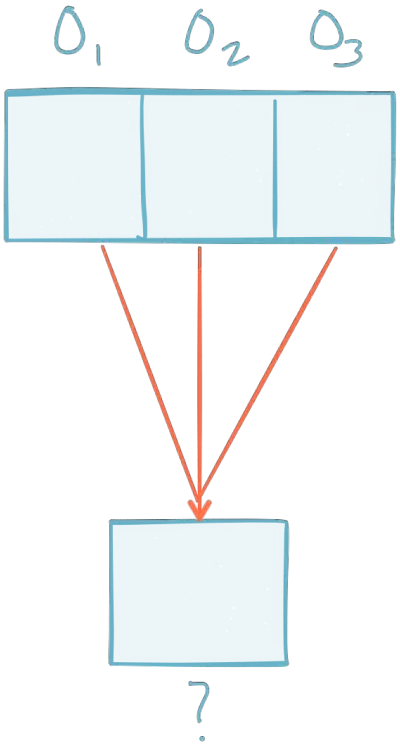
Configuring SMTs

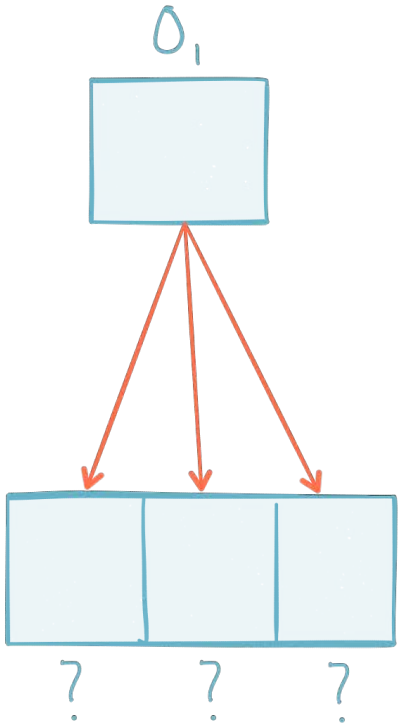
```
name=local-file-source
connector.class=FileStreamSource
tasks.max=1
file=test.txt
topic=connect-test
transforms=MakeMap,InsertSource
transforms.MakeMap.type=org.apache.kafka.connect.transforms.HoistField$Value
transforms.MakeMap.field=line
transforms.InsertSource.type=org.apache.kafka.connect.transforms.InsertField$Value
transforms.InsertSource.static.field=data_source
transforms.InsertSource.static.value=test-file-source
```

Why "Single?"



Why "Single?"





SMT Use Cases

- Data masking
- Mask sensitive information while sending it to Kafka
- Capture data from a relational database to Kafka,
- The data includes PCI / PII information and your Kafka cluster is not certified yet.

SMT Use Cases

- Event routing
- modify an event destination based on the contents of the event
- applies to events that need to get written to different database tables
- write events from Kafka to Elasticsearch, but each event needs to go to a different index - based on information in the event itself

SMT Use Cases

- Event enhancement
- Add additional fields to events while replicating
- e.g., Capture events from multiple data sources to Kafka, and want to include information about the source of the data in the event

SMT Use Cases

- Partitioning
- Set the key for the event based on event information before it gets written to Kafka
- e.g., reading records from a database table, partition the records in Kafka based on customer ID

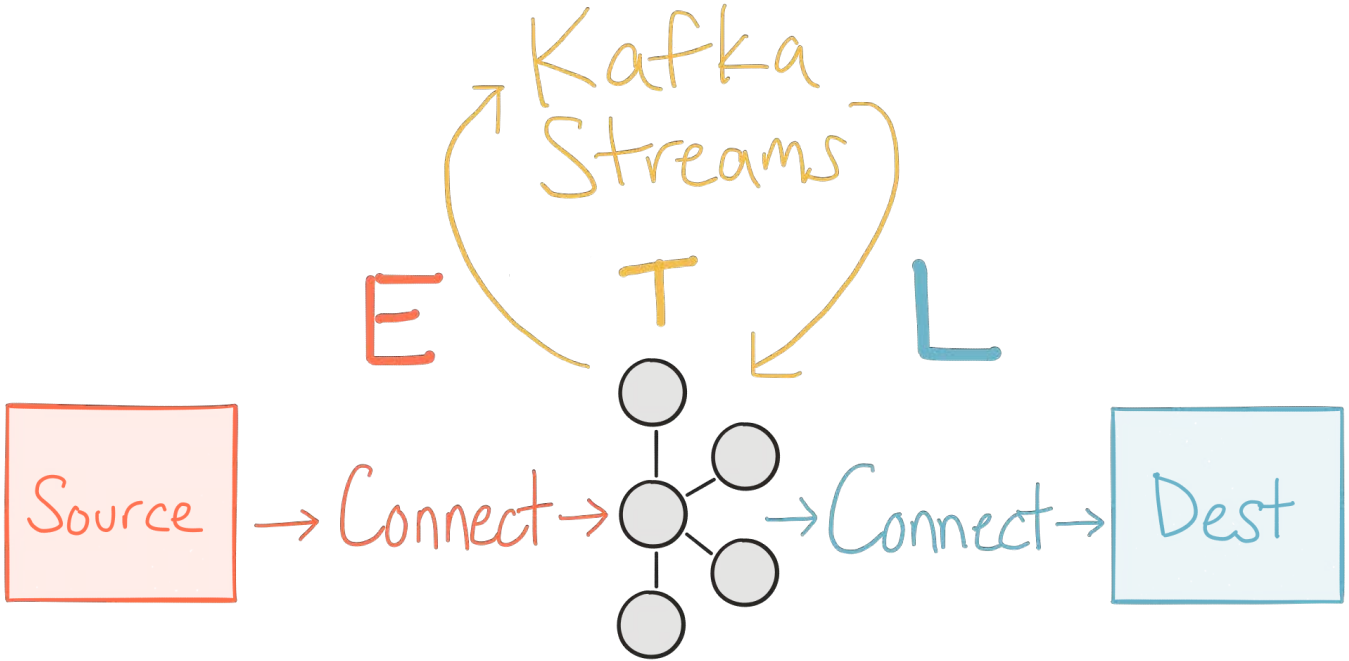
SMT Use Cases

- Timestamp conversion
- time-based data conversion standardization when integrating different systems
- e.g., there are many different ways to represent time. Often, Kafka events are read from logs, which use something like "[2017-01-31 05:21:00,298]" but the key-value store events are being written into prefer dates as "milliseconds since 1970"

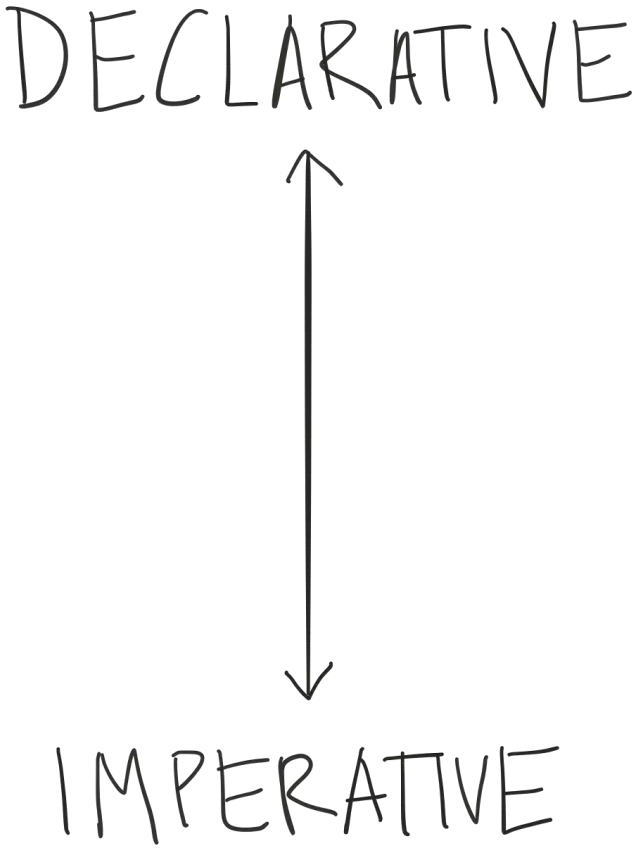
Unix Pipelines

```
cat <in | grep apache | tr a-z A-Z >out
```

Streaming Pipelines



Levels of Abstraction



Programming With Configuration

```
name=local-file-source
connector.class=FileStreamSource
tasks.max=1
file=test.txt
topic=connect-test
transforms=MakeMap,InsertSource
transforms.MakeMap.type=org.apache.kafka.connect.transforms.HoistField$Value
transforms.MakeMap.field=line
transforms.InsertSource.type=org.apache.kafka.connect.transforms.InsertField$Value
transforms.InsertSource.static.field=data_source
transforms.InsertSource.static.value=test-file-source
```

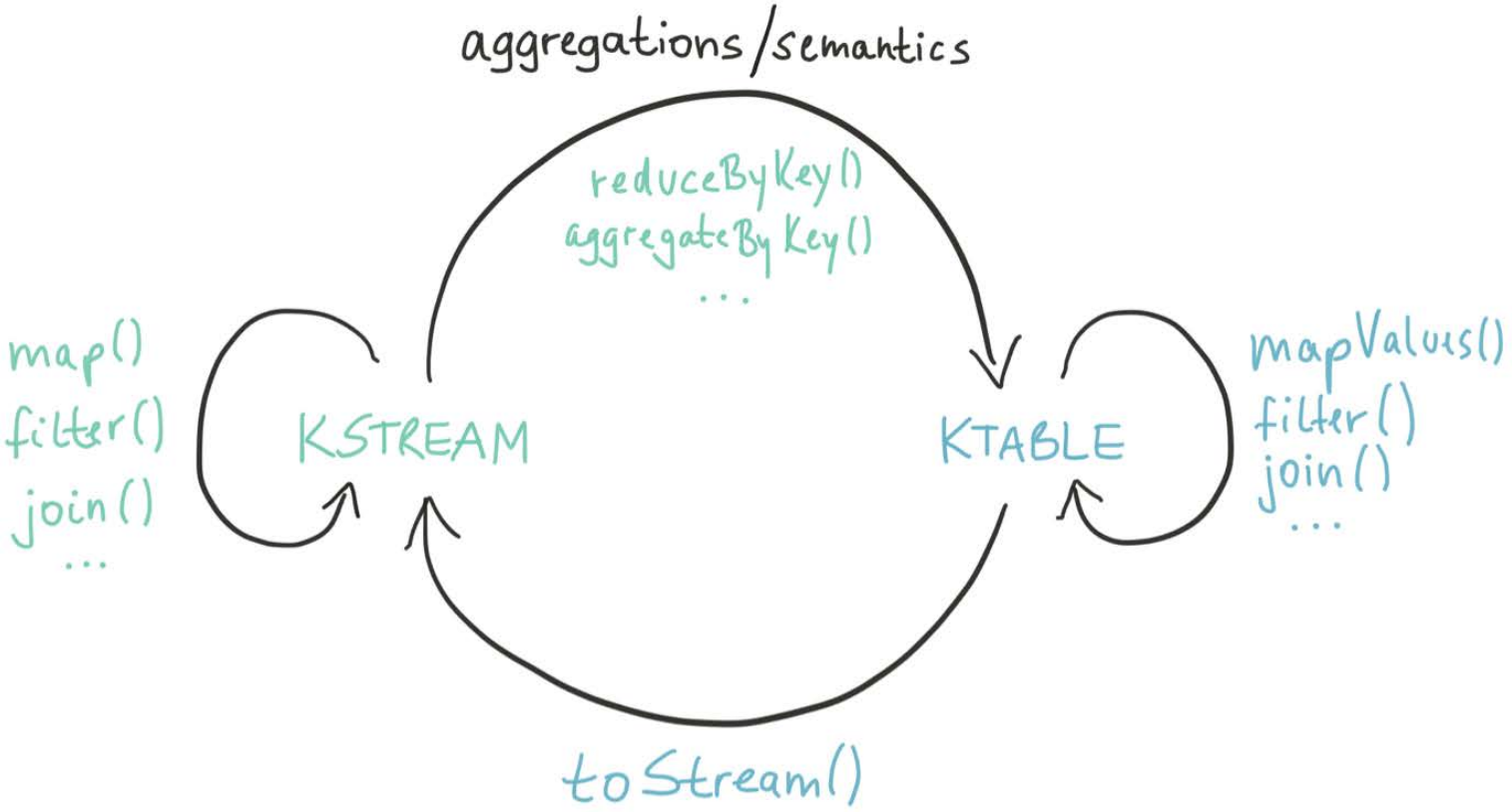
Programming With Configuration

```
name=local-file-source
connector.class=FileStreamSource
tasks.max=1
file=test.txt
topic=connect-test
transforms=MakeMap,InsertSource, InsertKey, ExtractStoreId
transforms.MakeMap.type=org.apache.kafka.connect.transforms.HoistField$Value
transforms.MakeMap.field=line
transforms.InsertSource.type=org.apache.kafka.connect.transforms.InsertField$Value
transforms.InsertSource.static.field=data_source
transforms.InsertSource.static.value=test-file-source
transforms.InsertKey.type=org.apache.kafka.connect.transforms.ValueToKey
transforms.InsertKey.fields=storeId
transforms.ExtractStoreId.type=org.apache.kafka.connect.transforms.ExtractField$Key
transforms.ExtractStoreId.field=storeId
```

Programming With Configuration

```
name=local-file-source
connector.class=FileStreamSource
tasks.max=1
file=test.txt
topic=connect-test
transforms=MakeMap,InsertSource, InsertKey, ExtractStoreId, MessageTypeRouter
transforms.MakeMap.type=org.apache.kafka.connect.transforms.HoistField$Value
transforms.MakeMap.field=line
transforms.InsertSource.type=org.apache.kafka.connect.transforms.InsertField$Value
transforms.InsertSource.static.field=data_source
transforms.InsertSource.static.value=test-file-source
transforms.InsertKey.type=org.apache.kafka.connect.transforms.ValueToKey
transforms.InsertKey.fields=storeId
transforms.ExtractStoreId.type=org.apache.kafka.connect.transforms.ExtractField$Key
transforms.ExtractStoreId.field=storeId
transforms.MessageTypeRouter.type=org.apache.kafka.connect.transforms.RegexRouter
transforms.MessageTypeRouter.regex=(foo|bar|baz)-.*
transforms.MessageTypeRouter.replacement=$1-logs
```

The Right Tool For The Job



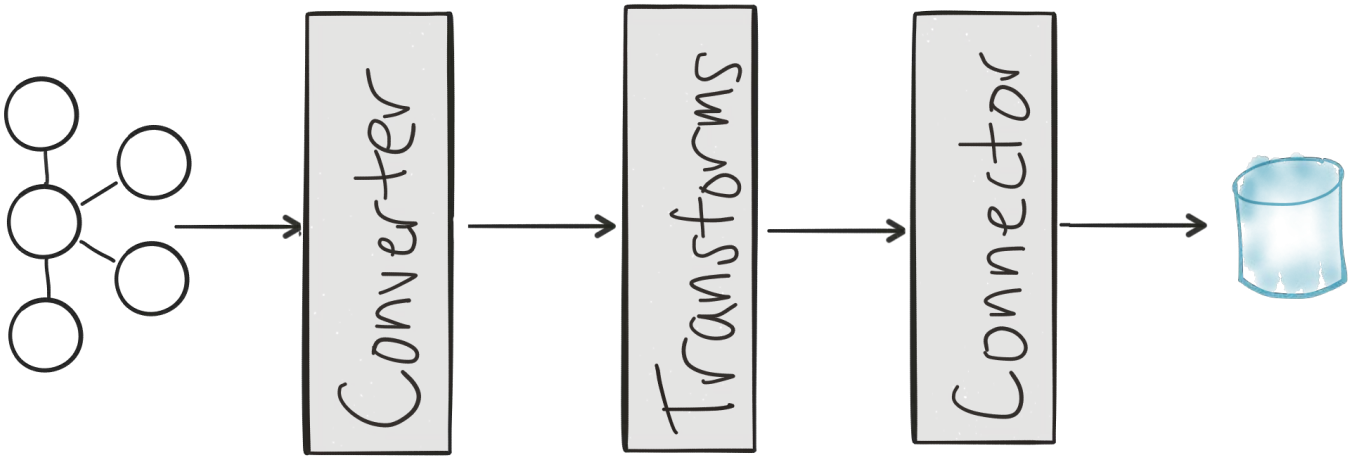
The Right Tool For The Job

```
KStream<Integer, Integer> input = builder.stream(“numbers-topic”);
KStream<Integer, Integer> doubled = input.mapValues(v -> v * 2);

KTable<Integer, Integer> sumOfOdds = input
    .filter((k,v) -> v % 2 != 0)
    .selectKey((k, v) -> 1)
    .reduceByKey((v1, v2) -> v1 + v2, ”sum-of-odds");
```

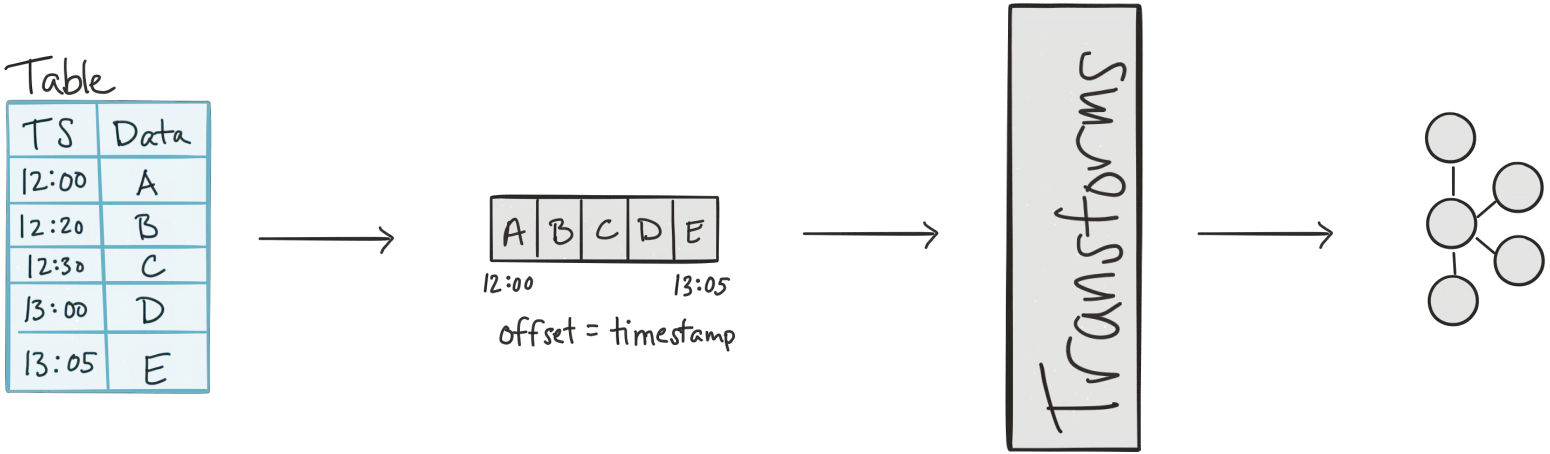
Order of Operations

```
name=my-sink
topics=foo-logs-jetty, foo-logs-app, bar-logs-jetty, bar-logs-app
topic.index.map=foo-logs-jetty:foo-logs,\
                foo-logs-app:foo-logs,\
                bar-logs-jetty:bar-logs,\
                bar-logs-app:bar-logs

transforms=Router
transforms.Router.type=org.apache.kafka.connect.transforms.TimestampRouter
transforms.Router.topic.format=${topic}-${timestamp}
transforms.Router.timestamp.format=yyyyMMddHH
```

Order of Operations



Schemas



confluent

Implementing a Transformation

```java
/**
 * Single message transformation for Kafka Connect record types.
 *
 * Connectors can be configured with transformations to make lightweight
 * message-at-a-time modifications.
 */
public interface Transformation<R extends ConnectRecord<R>> extends Configurable, Closeable {

    /**
     * Apply transformation to the {@code record} and return another record object.
     *
     * The implementation must be thread-safe.
     */
    R apply(R record);


    /** Configuration specification for this transformation. **/
    ConfigDef config();


    /** Signal that this transformation instance will no longer will be used. **/
    @Override
    void close();

}
```

Confluent Community

https://www.confluent.io/apache-kafka-meetups/

https://slackpass.io/confluentcommunity/