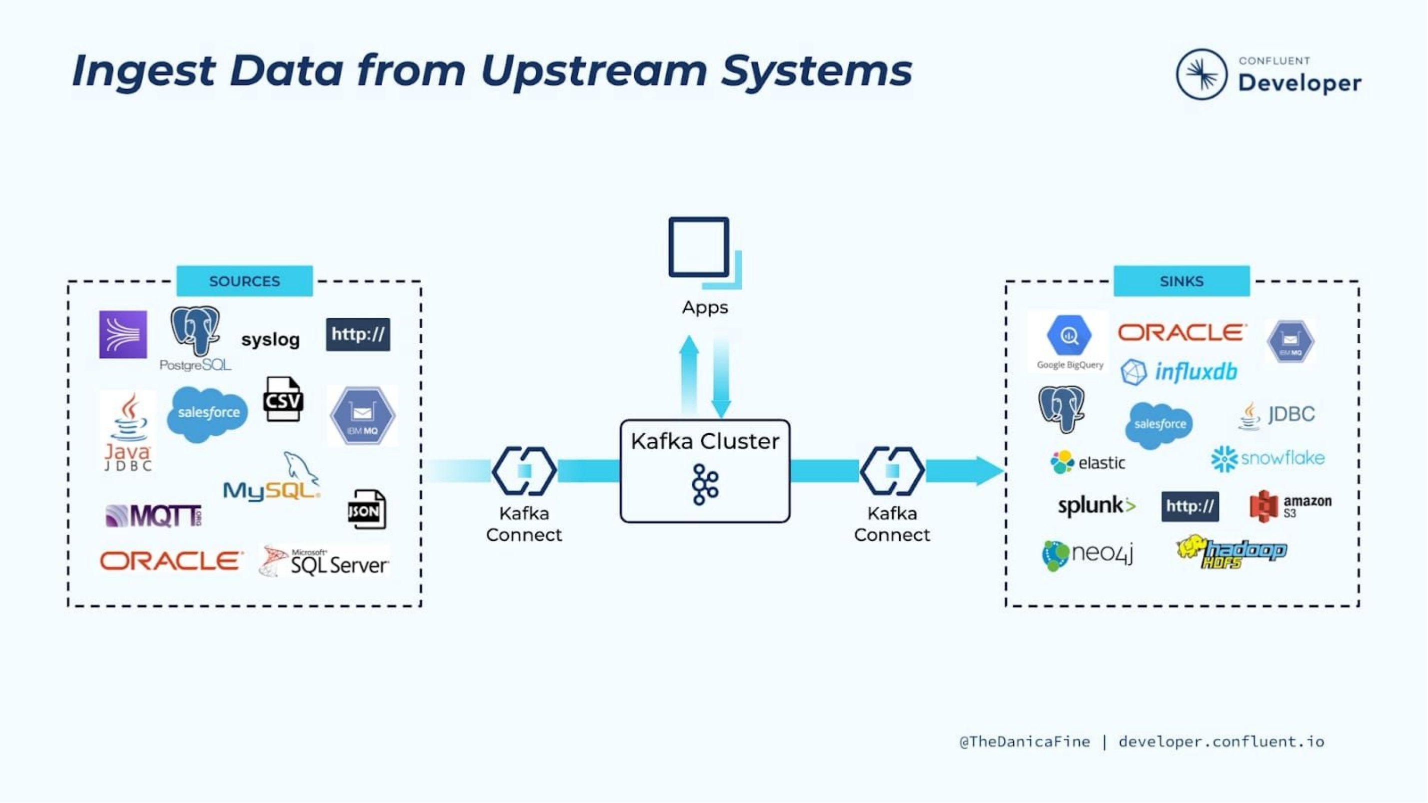
Kafka Connect API

Ingest Data from Upstream Systems

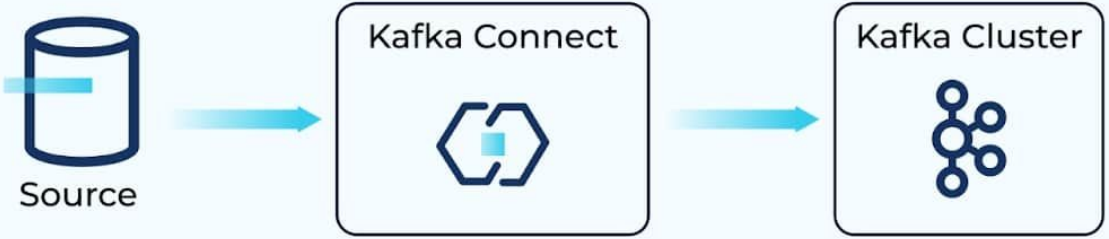


Ingest Data from Upstream Systems

CONFLUENT Developer

@TheDanicaFine | developer.confluent.io

How Kafka Connect Works



Source

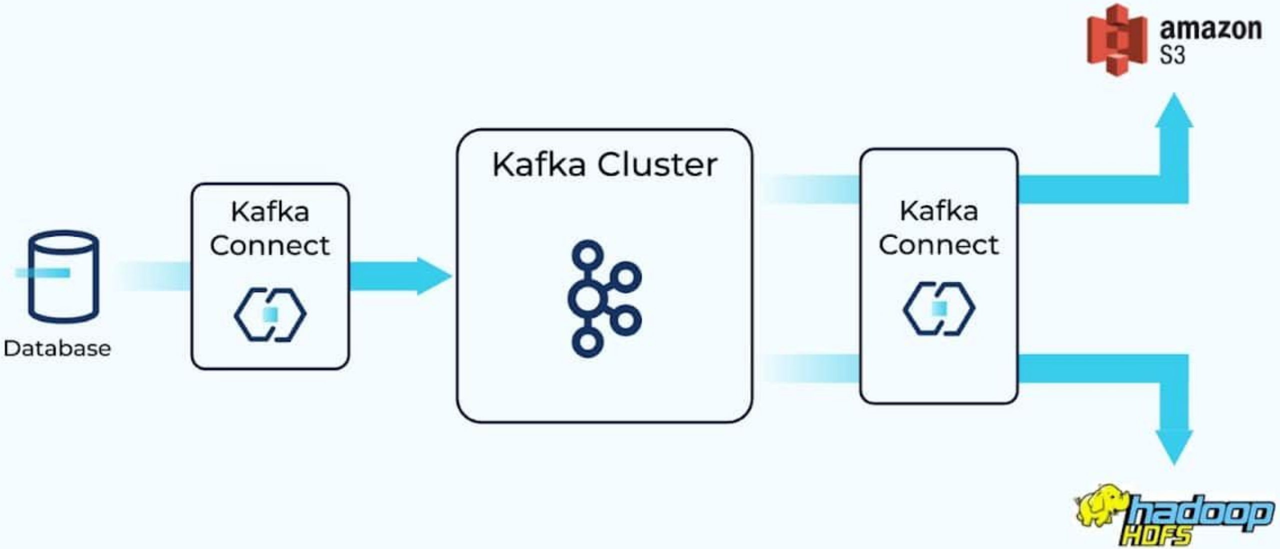
Kafka Connect

Kafka Cluster

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

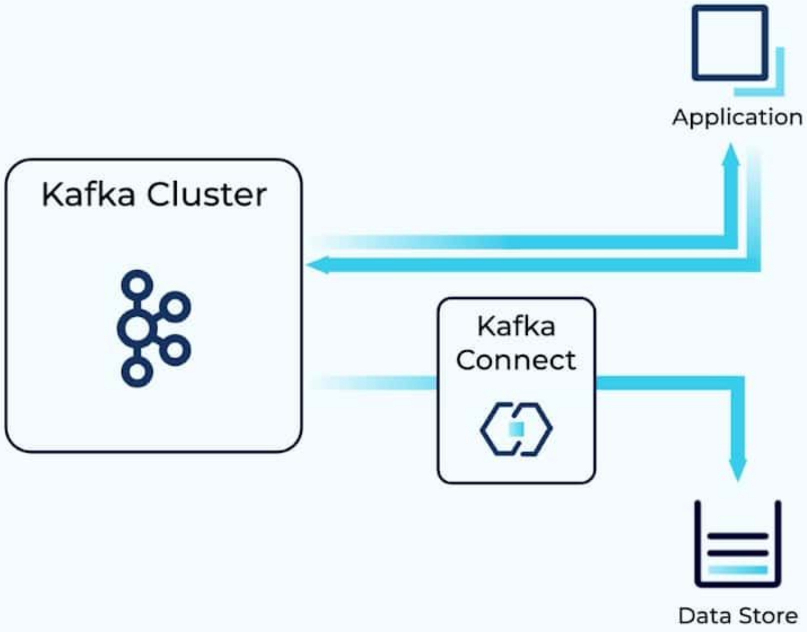
Streaming Pipelines

CONFLUENT
Developer



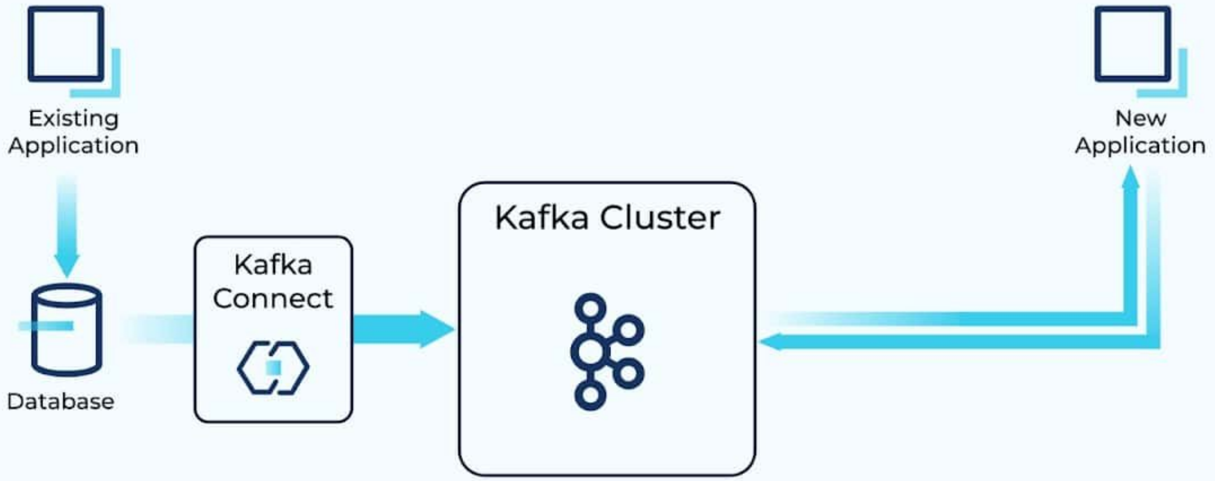
Writing to Datastores from Kafka

CONFLUENT Developer



Evolve Processing from Old Systems to New

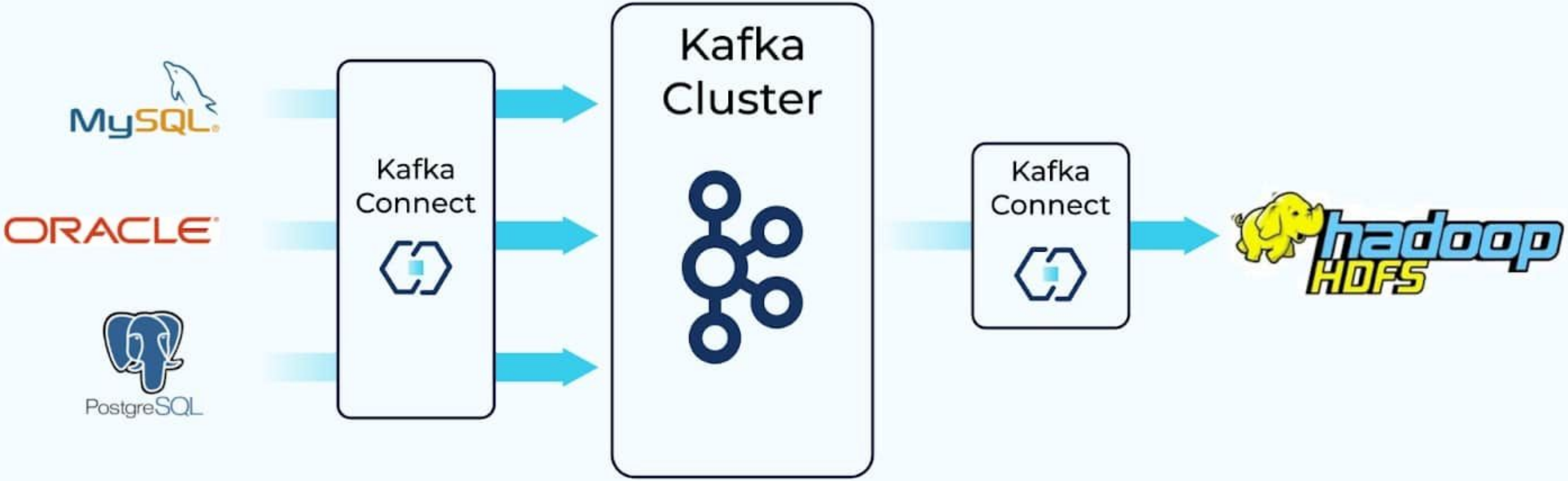
CONFLUENT Developer



@TheDanicaFine | developer.confluent.io

Make Systems Real Time

CONFLUENT Developer

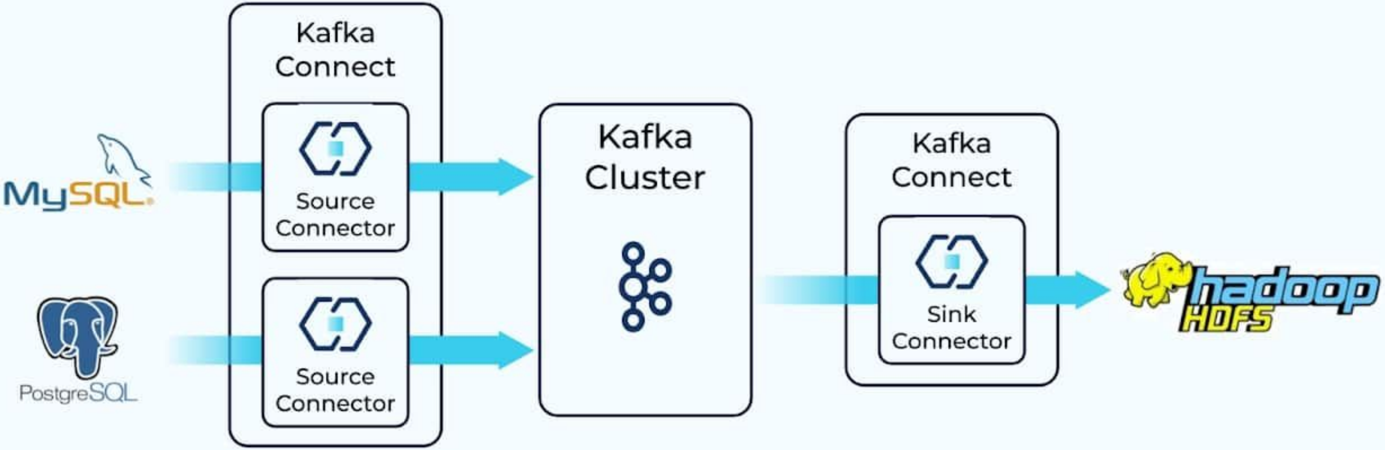


@TheDanicaFine | developer.confluent.io

Why Not Write Your Own Integrations?

Connectors

CONFLUENT Developer



Kafka Connect Workers

Ultimately, Kafka Connect workers are just JVM processes that you can deploy on bare metal or containers.

A few options present themselves:

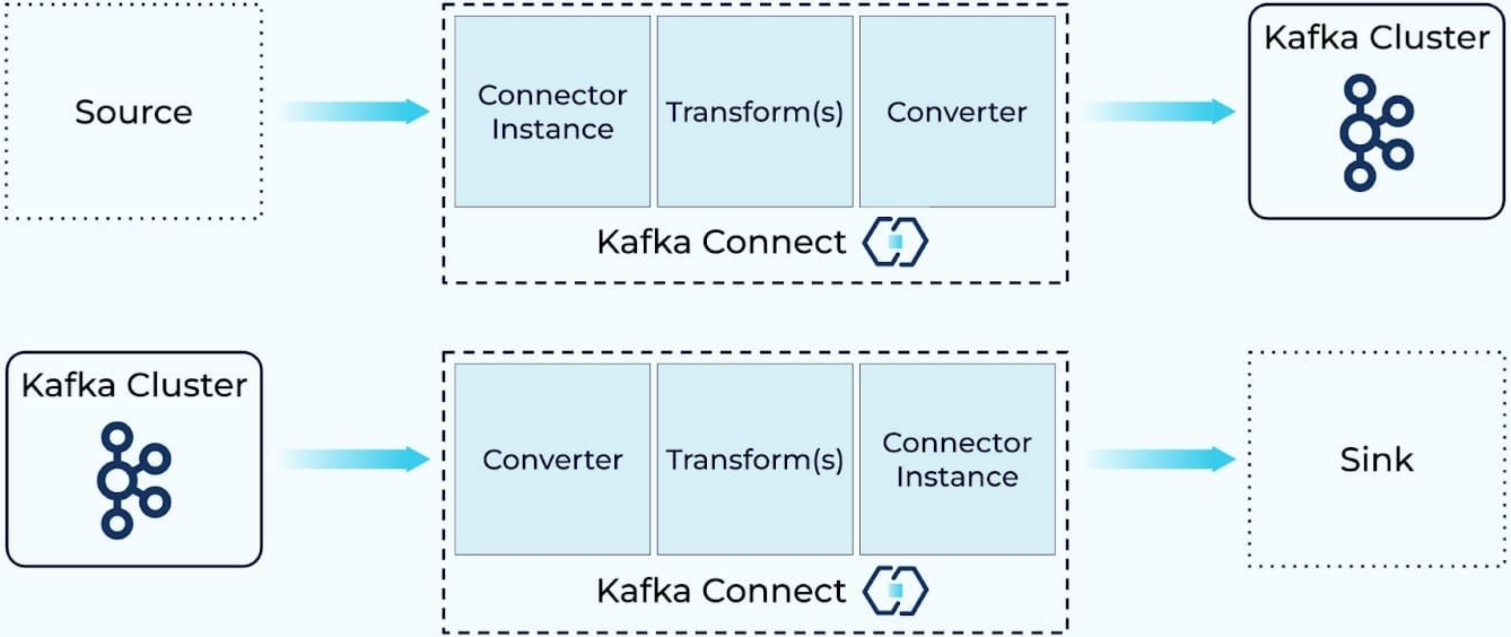
- You’re free to run a bare-metal, on-premises install of Confluent Platform
- For those leveraging infrastructure as a service, you may install Confluent Platform on those resources
- Terraform is an option on a couple cloud providers
- And of course, there’s Docker which you can use for both on-prem and cloud-based installations

Managing a Kafka Connect Cluster

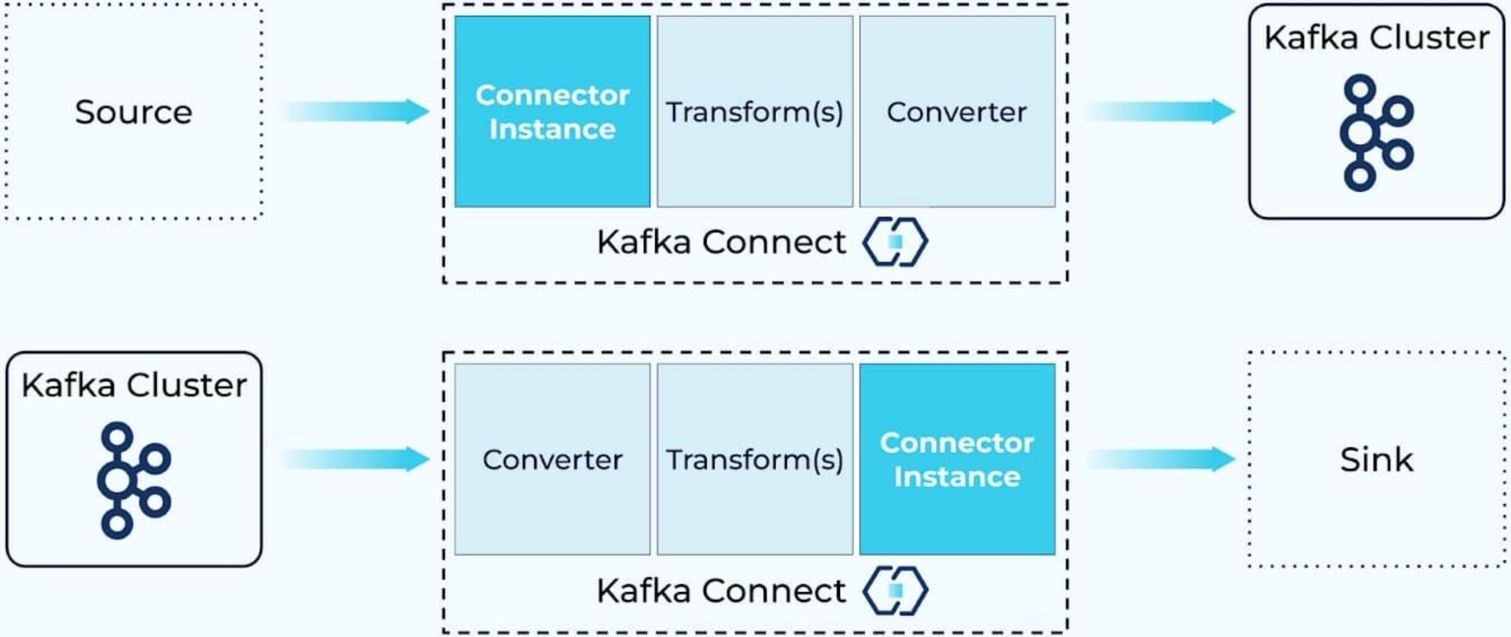
Once your Kafka Connect cluster is up and running, there’s a bit of management that needs to be done:

- Connect workers have a number of default configuration settings that you may need to alter
- Depending on the needs of your systems, you might need to scale the Connect cluster up or down to suit demand changes
- And of course, you’ll be monitoring for problems and fixing those that occur

Inside Kafka Connect



Connectors



Source

Connector Instance

Transform(s)

Converter

Kafka Connect

Kafka Cluster

Kafka Cluster

Converter

Transform(s)

Connector Instance

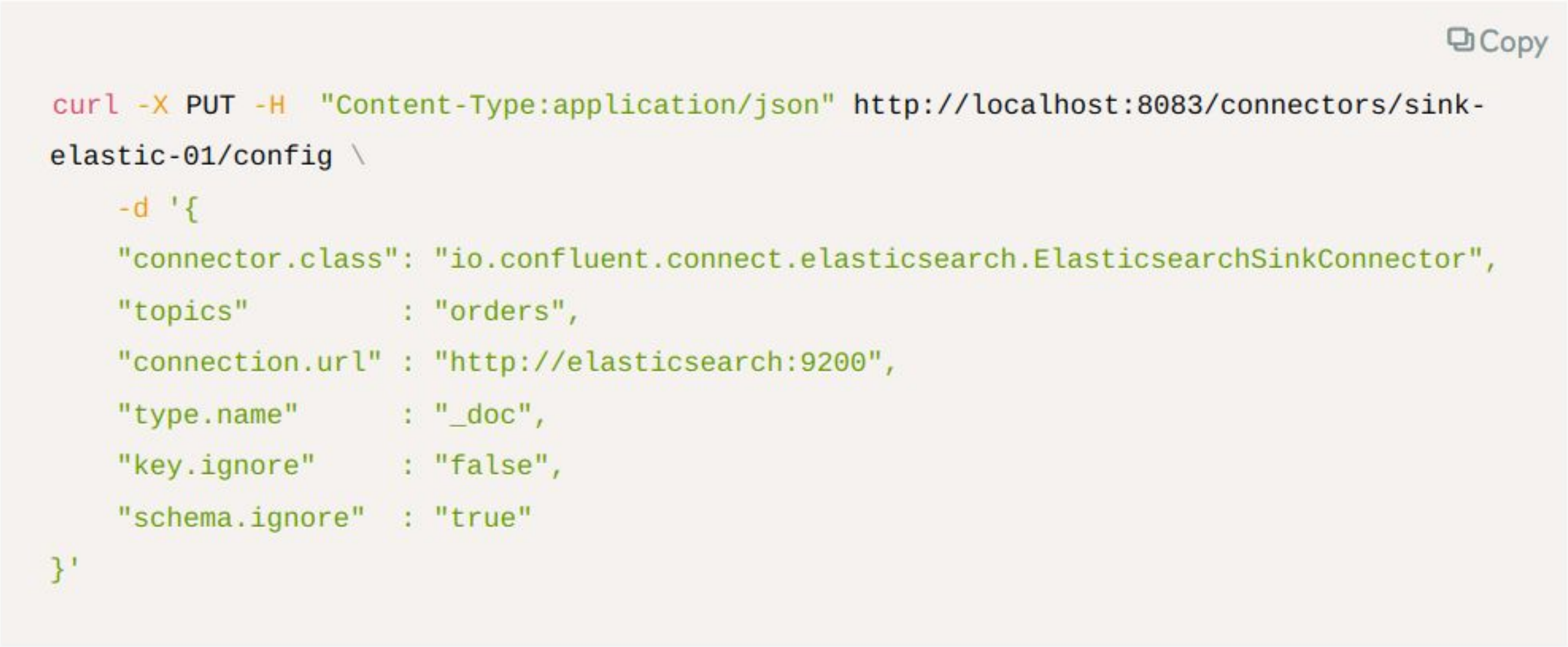
Kafka Connect

Sink

For Example

- The **Debezium MySQL source connector** uses the MySQL bin log to read events from the database and stream these to Kafka Connect
- The **Elasticsearch sink connector** takes data from Kafka Connect, and using the Elasticsearch APIs, writes the data to Elasticsearch
- The **S3 connector** from Confluent can act as both a source and sink connector, writing data to S3 or reading it back in

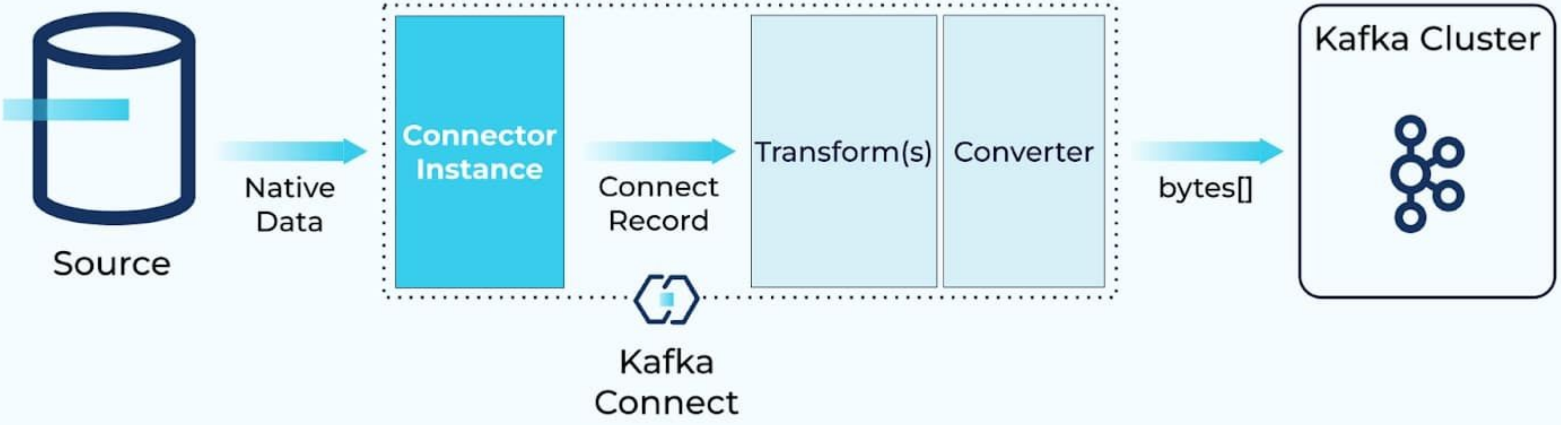
Add a Connector Instance with the REST API



```
curl -X PUT -H  "Content-Type:application/json" http://localhost:8083/connectors/sink-
elastic-01/config \
    -d '{
    "connector.class": "io.confluent.connect.elasticsearch.ElasticsearchSinkConnector",
    "topics"         : "orders",
    "connection.url" : "http://elasticsearch:9200",
    "type.name"      : "_doc",
    "key.ignore"     : "false",
    "schema.ignore"  : "true"
}'
```

What is the Role of the Connector?

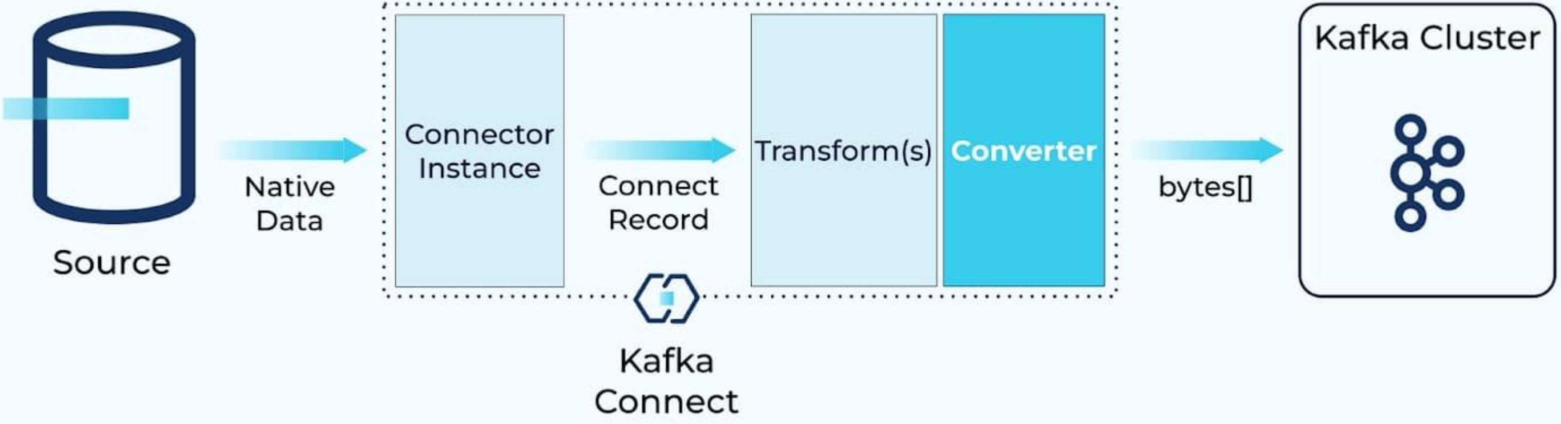
CONFLUENT Developer



@TheDanicaFine | developer.confluent.io

Converters Serialize/Deserialize the Data

CONFLUENT Developer

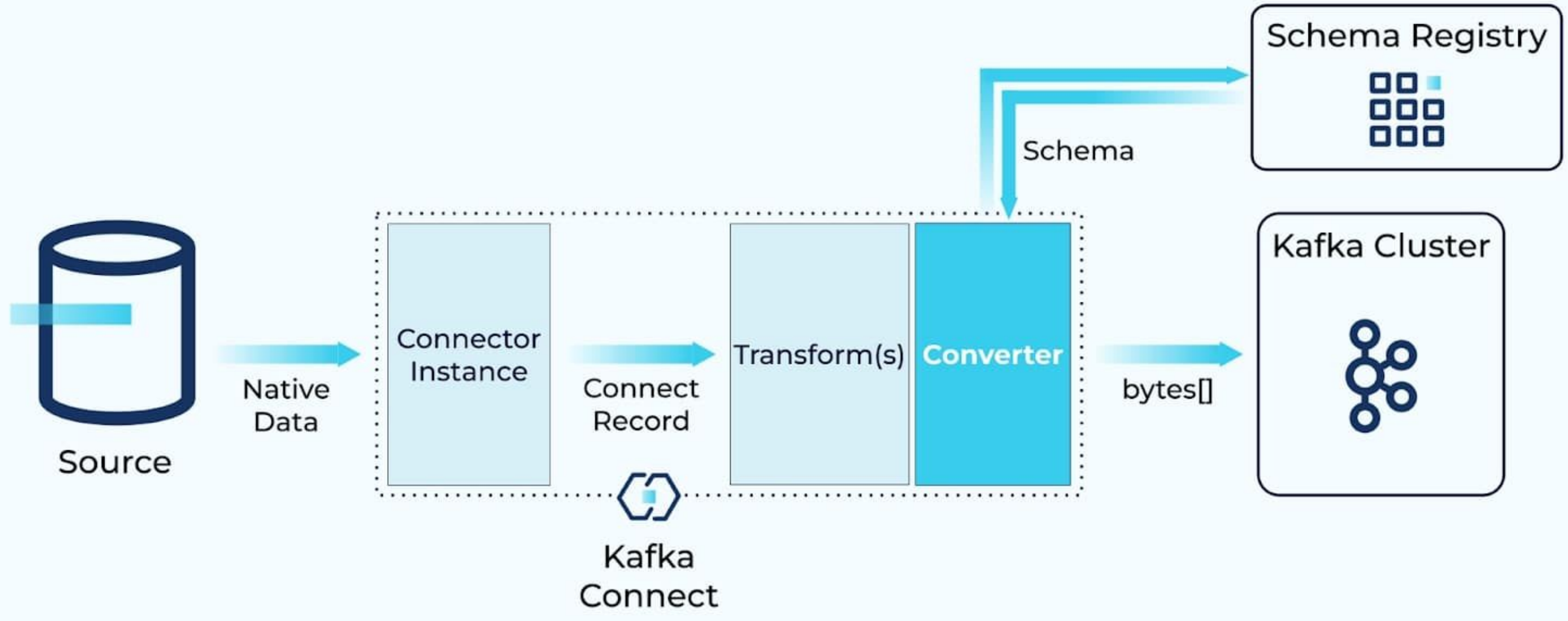


There are a ton of different converters available, but some common ones include:

- Avro – io.confluent.connect.avro.AvroConverter
- Protobuf – io.confluent.connect.protobuf.ProtobufConverter
- String – org.apache.kafka.connect.storage.StringConverter
- JSON – org.apache.kafka.connect.json.JsonConverter
- JSON Schema – io.confluent.connect.json.JsonSchemaConverter
- ByteArray – org.apache.kafka.connect.converters.ByteArrayConverter

Serialization and Schemas

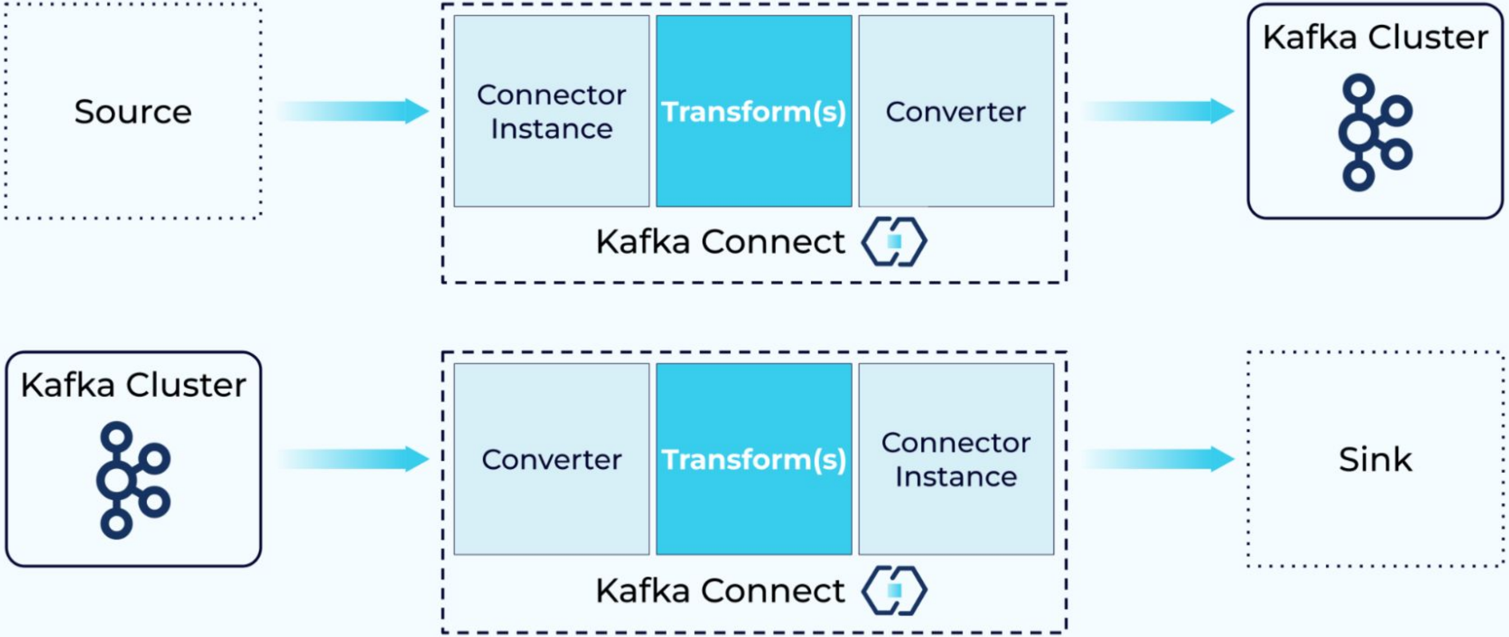
CONFLUENT Developer



@TheDanicaFine | developer.confluent.io

Single Message Transforms

CONFLUENT Developer



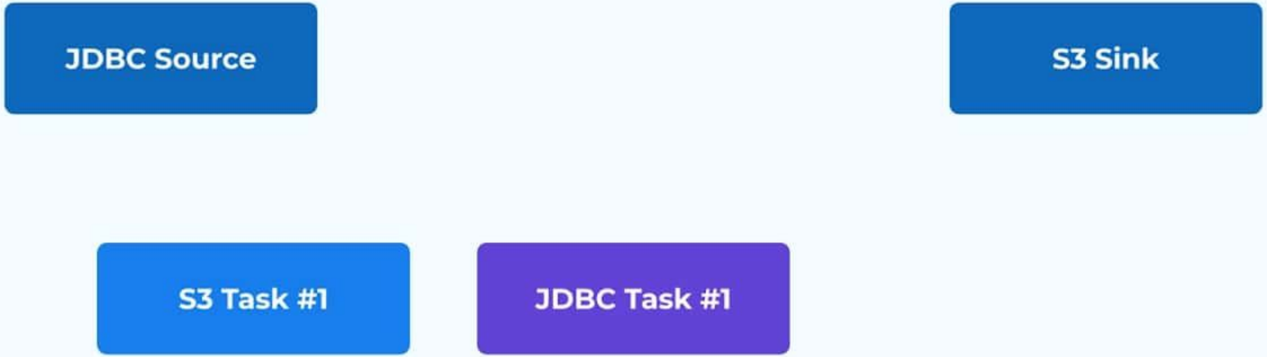
@TheDanicaFine | developer.confluent.io

Common uses for SMTs include:

- Dropping fields from data at ingest, such as personally identifiable information (PII) if specified by the system requirements
- Adding metadata information such as lineage to data ingested through Kafka Connect
- Changing field data types
- Modifying the topic name to include a timestamp
- Renaming fields

Deploying Kafka Connect

Deploying Kafka Connect



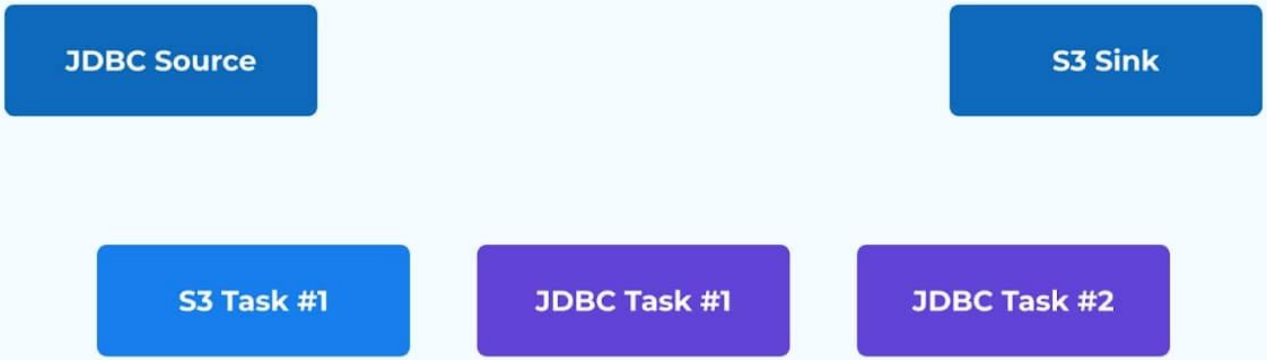
JDBC Source

S3 Sink

S3 Task #1

JDBC Task #1

Tasks Are the Unit of Parallelism and Scale



JDBC Source

S3 Sink

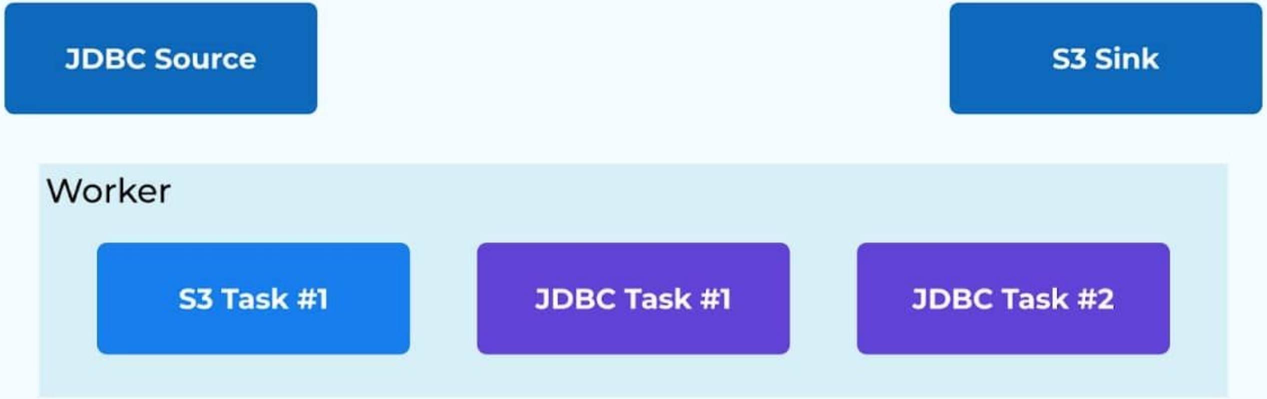
S3 Task #1

JDBC Task #1

JDBC Task #2

Connect Worker

CONFLUENT Developer



JDBC Source

S3 Sink

Worker

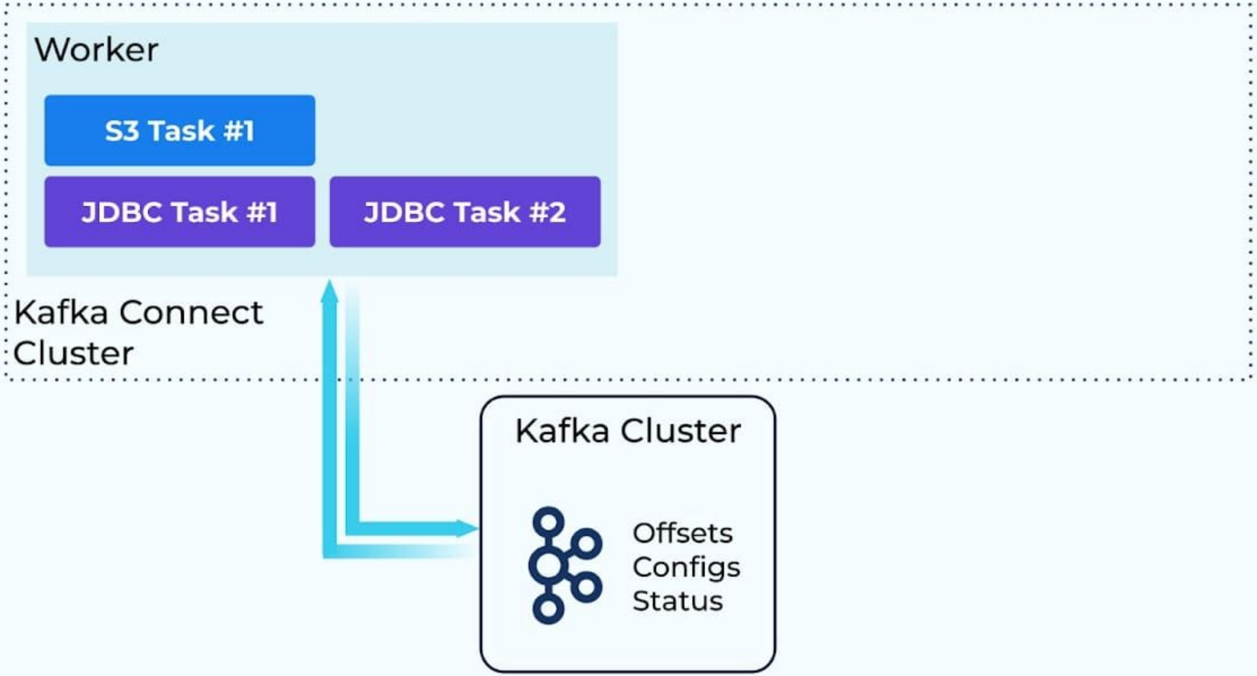
S3 Task #1

JDBC Task #1

JDBC Task #2

Kafka Connect Distributed Mode

CONFLUENT Developer



Worker

S3 Task #1

JDBC Task #1

JDBC Task #2

Kafka Connect Cluster

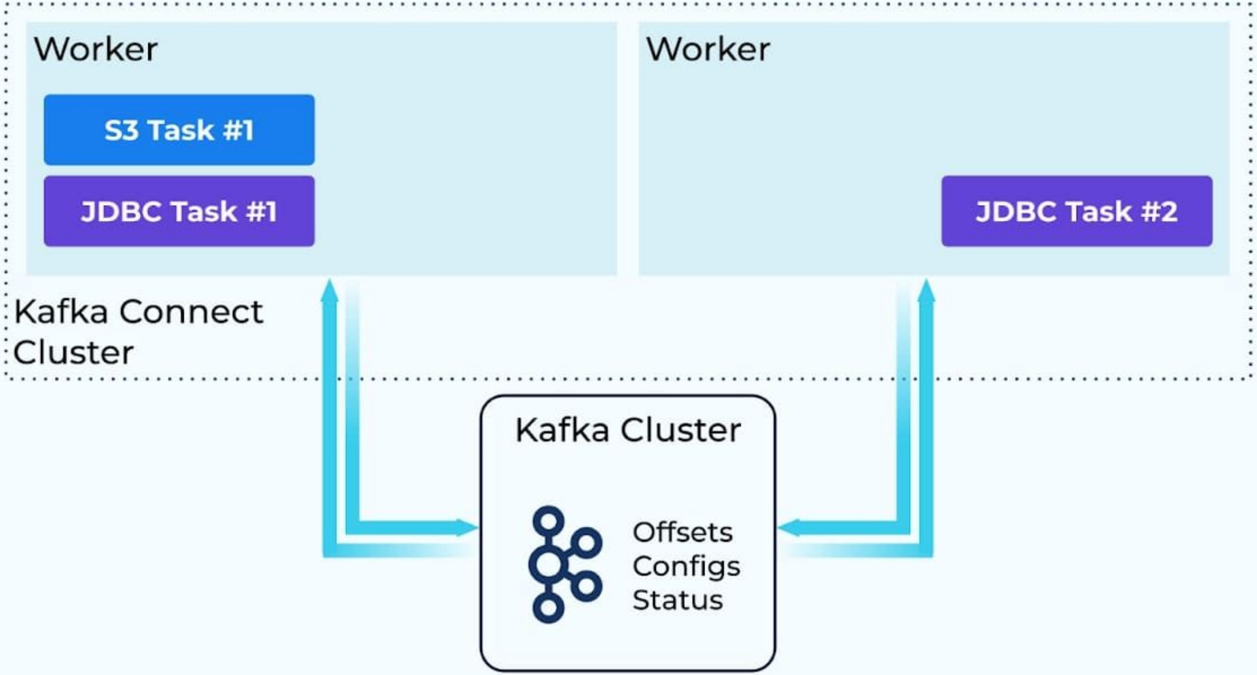
Kafka Cluster

Offsets
Configs
Status

@TheDanicaFine | developer.confluent.io

Kafka Connect Scalability

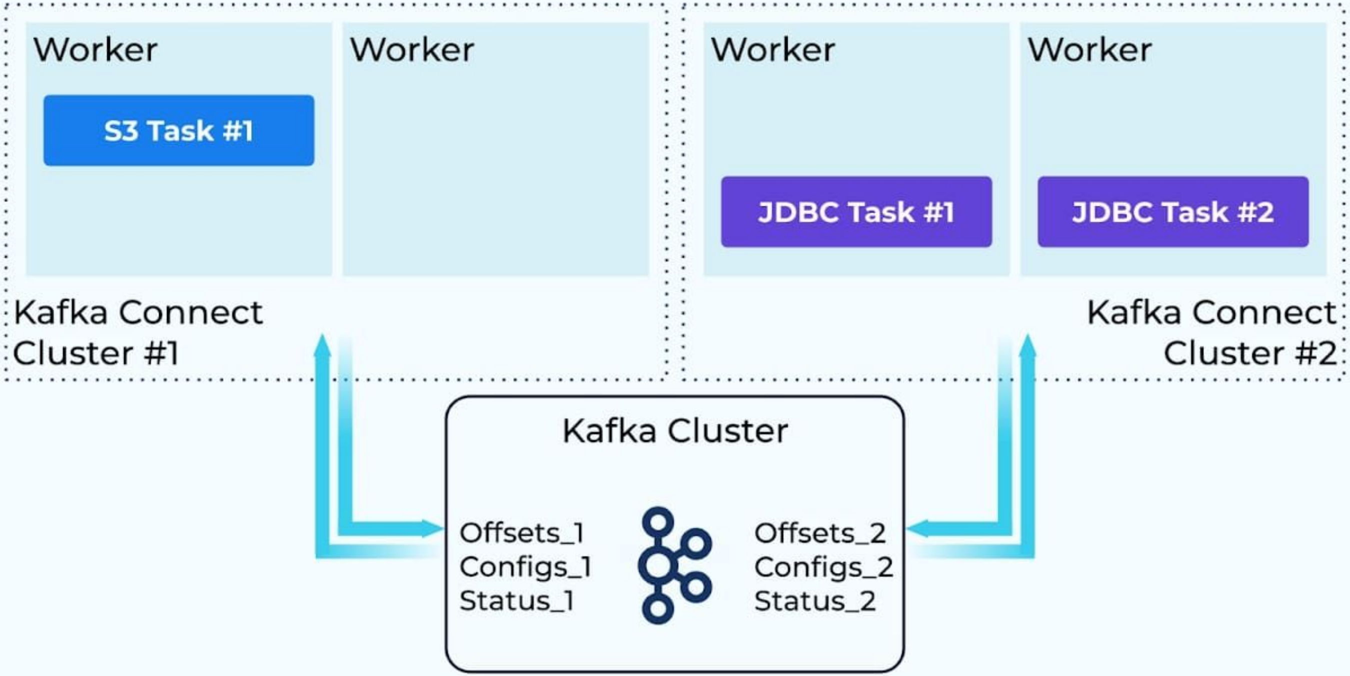
CONFLUENT Developer



@TheDanicaFine | developer.confluent.io

Multiple Workers vs Multiple Clusters

CONFLUENT Developer



Worker

S3 Task #1

Worker

Kafka Connect Cluster #1

Worker

JDBC Task #1

Worker

JDBC Task #2

Kafka Connect Cluster #2

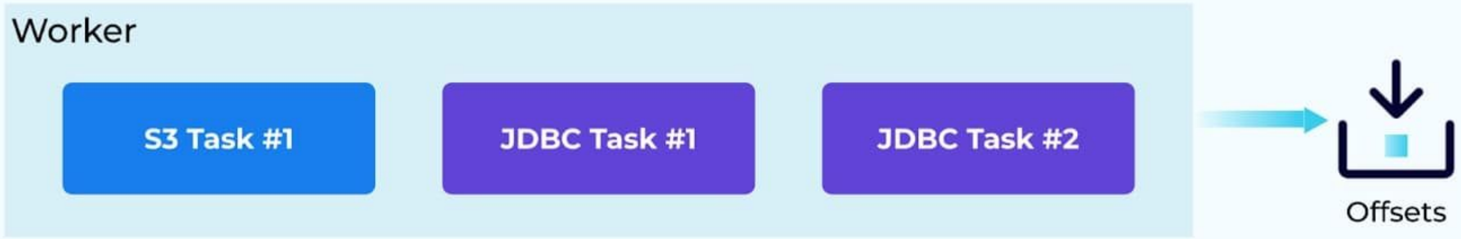
Kafka Cluster

Offsets_1
Configs_1
Status_1

Offsets_2
Configs_2
Status_2

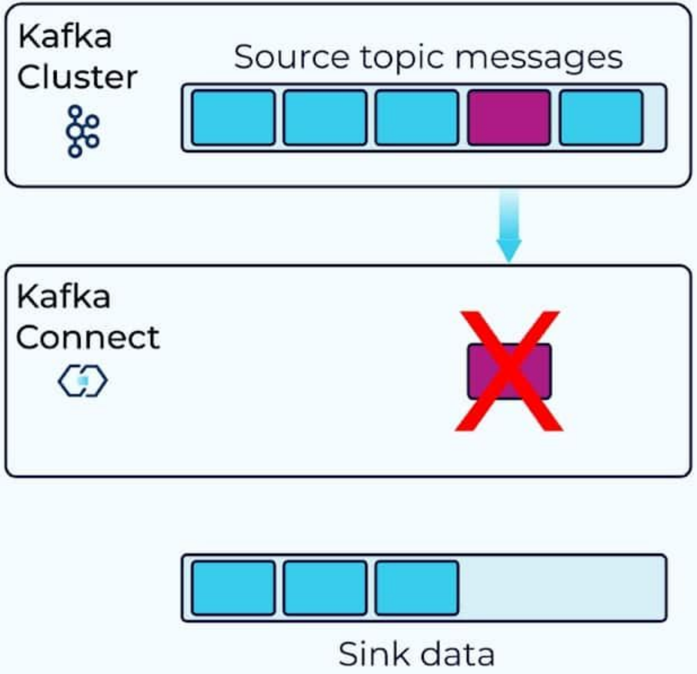
@TheDanicaFine | developer.confluent.io

Kafka Connect Standalone Mode



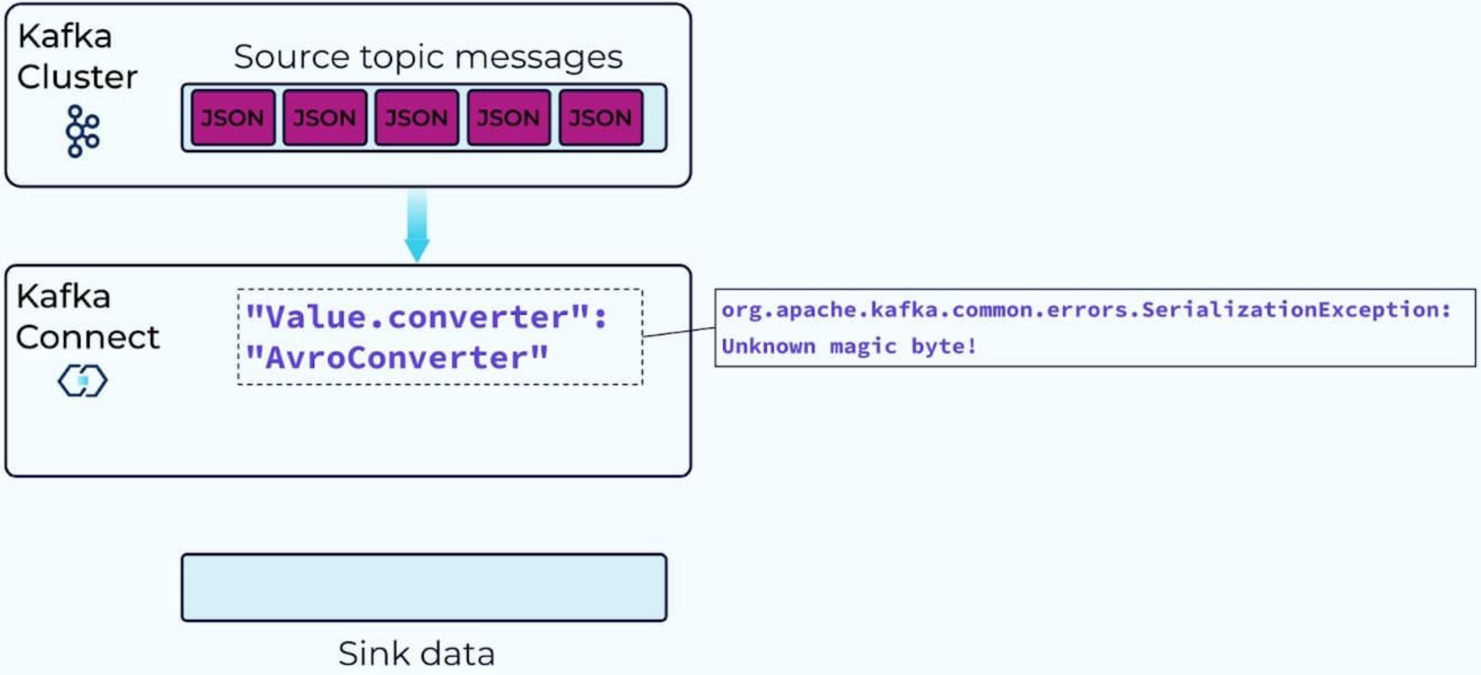
Error Handling in Kafka Connect

CONFLUENT Developer



Serialization Challenges - Wrong Converter

CONFLUENT Developer

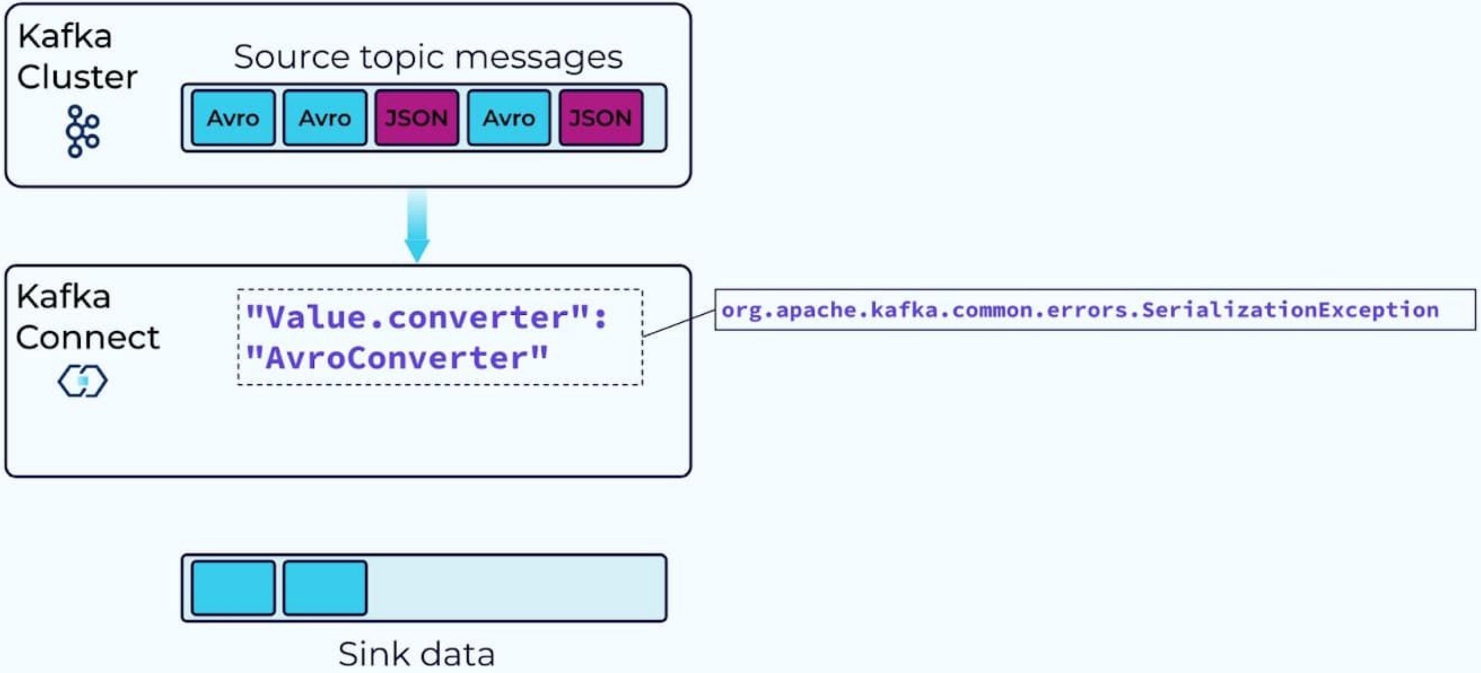


```
"Value.converter":
"AvroConverter"
```

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

Serialization Challenges - Multiple Formats

CONFLUENT Developer



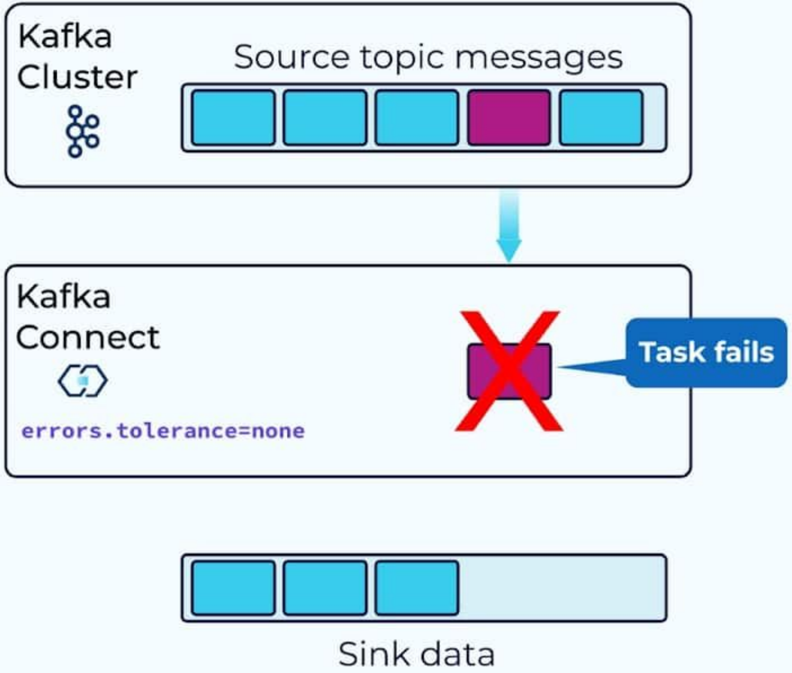
```
"Value.converter":
"AvroConverter"
```

```
org.apache.kafka.common.errors.SerializationException
```

@TheDanicaFine | developer.confluent.io

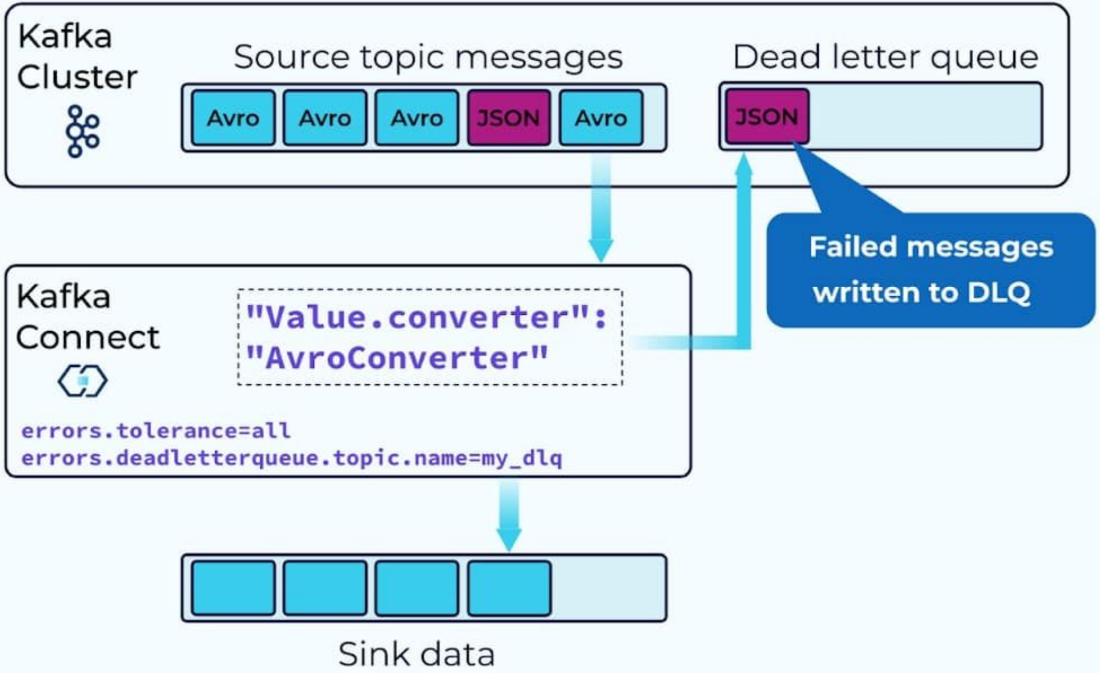
Error Tolerances - Fail Fast (Default)

CONFLUENT Developer



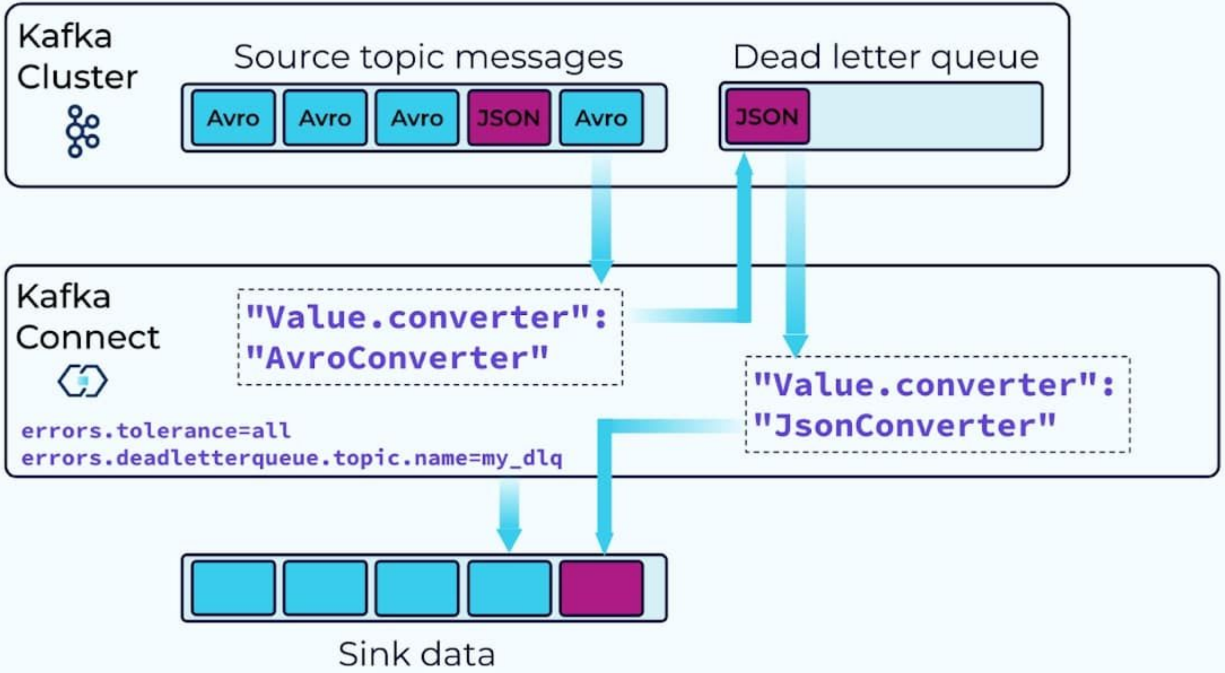
Error Tolerances - Dead Letter Queue

CONFLUENT Developer



Reprocessing the Dead Letter Queue

CONFLUENT Developer



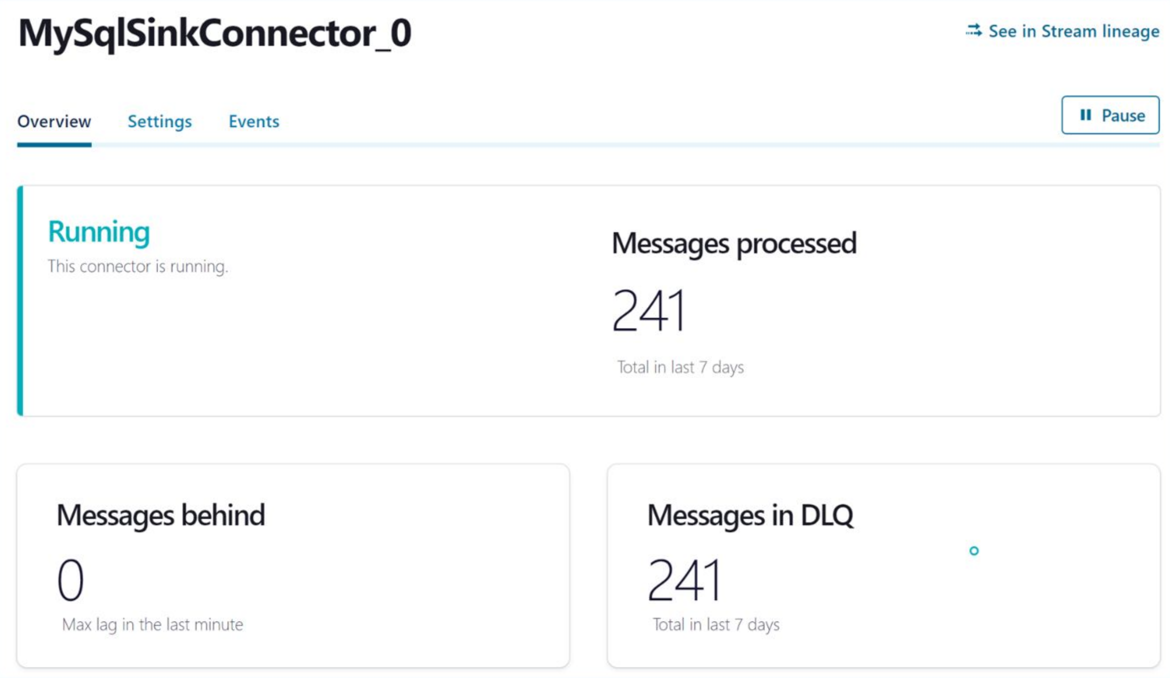
```
"Value.converter":
"AvroConverter"
```

```
"Value.converter":
"JsonConverter"
```

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

Troubleshoot the Dead Letter Queue

CONFLUENT Developer



MySqlSinkConnector_0

Running

This connector is running.

Messages processed

241

Total in last 7 days

Messages behind

0

Max lag in the last minute

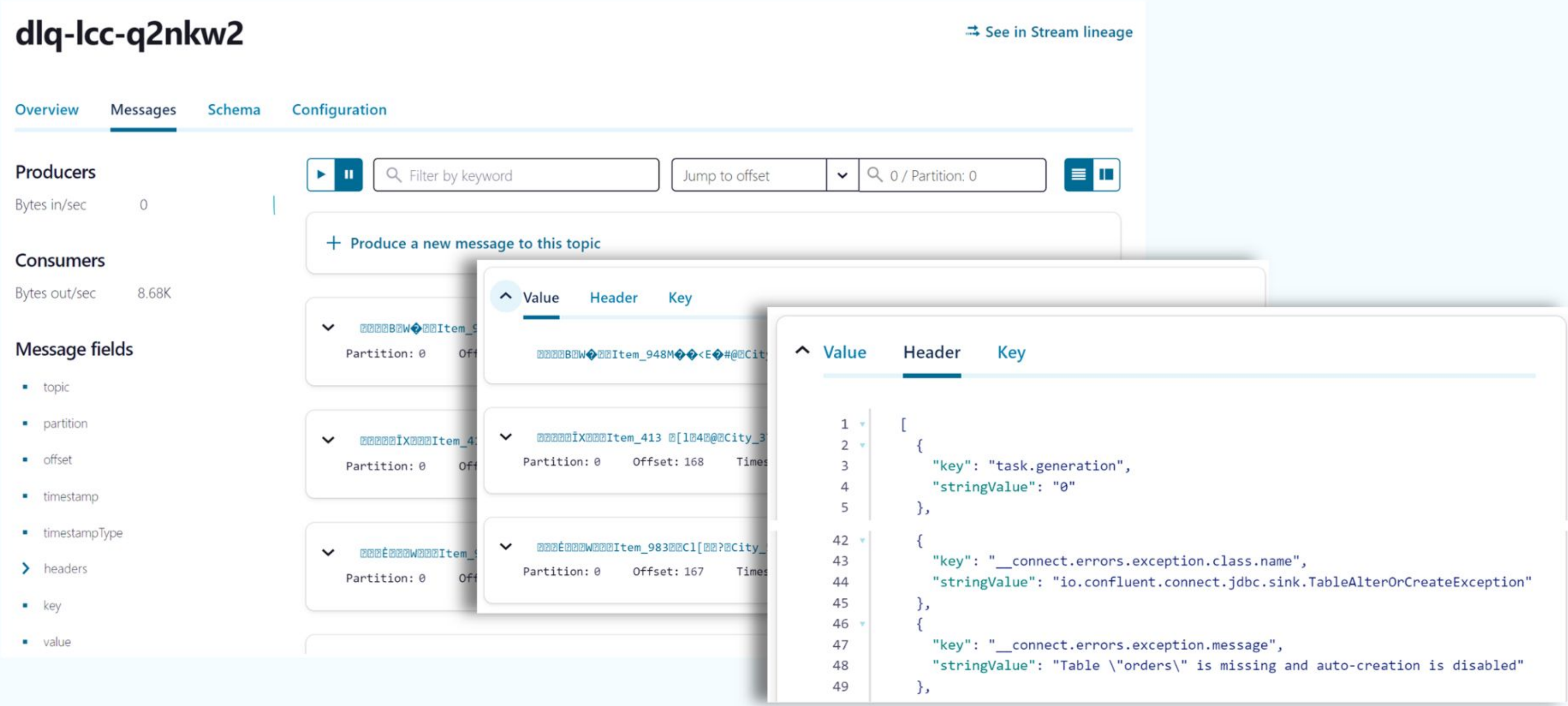
Messages in DLQ

241

Total in last 7 days

Dead Letter Queue Message Header

CONFLUENT Developer



dlq-lcc-q2nkw2

See in Stream lineage

Producers

Bytes in/sec 0

Consumers

Bytes out/sec 8.68K

Message fields

- topic
- partition
- offset
- timestamp
- timestampType
- headers
- key
- value

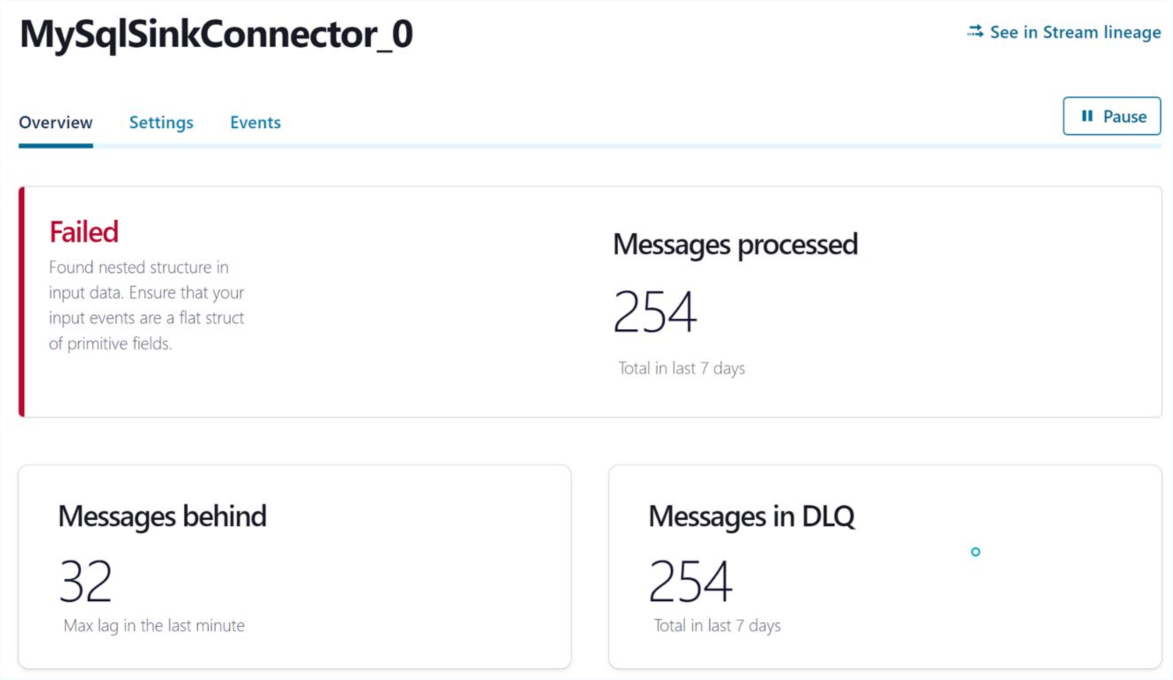
```
⍰⍰⍰⍰B⍰W�⍰⍰Item_9
Partition: 0     Off

⍰⍰⍰⍰⍰ÎX⍰⍰⍰Item_41
Partition: 0     Off

⍰⍰⍰Ė⍰⍰⍰W⍰⍰⍰Item_9
Partition: 0     Off
```

```
⍰⍰⍰⍰B⍰W�⍰⍰Item_948M��<E�#@⍰Cit

⍰⍰⍰⍰⍰ÎX⍰⍰⍰Item_413 ⍰[1⍰4⍰@⍰City_3
Partition: 0     Offset: 168     Times

⍰⍰⍰Ė⍰⍰⍰W⍰⍰⍰Item_983⍰⍰Cl[⍰⍰?⍰City_
Partition: 0     Offset: 167     Times
```

```
[
  {
    "key": "task.generation",
    "stringValue": "0"
  },
  {
    "key": "__connect.errors.exception.class.name",
    "stringValue": "io.confluent.connect.jdbc.sink.TableAlterOrCreateException"
  },
  {
    "key": "__connect.errors.exception.message",
    "stringValue": "Table \"orders\" is missing and auto-creation is disabled"
  },
```

Troubleshoot a Failed Connector

CONFLUENT Developer



MySqlSinkConnector_0

Failed

Found nested structure in input data. Ensure that your input events are a flat struct of primitive fields.

Messages processed

254

Total in last 7 days

Messages behind

32

Max lag in the last minute

Messages in DLQ

254

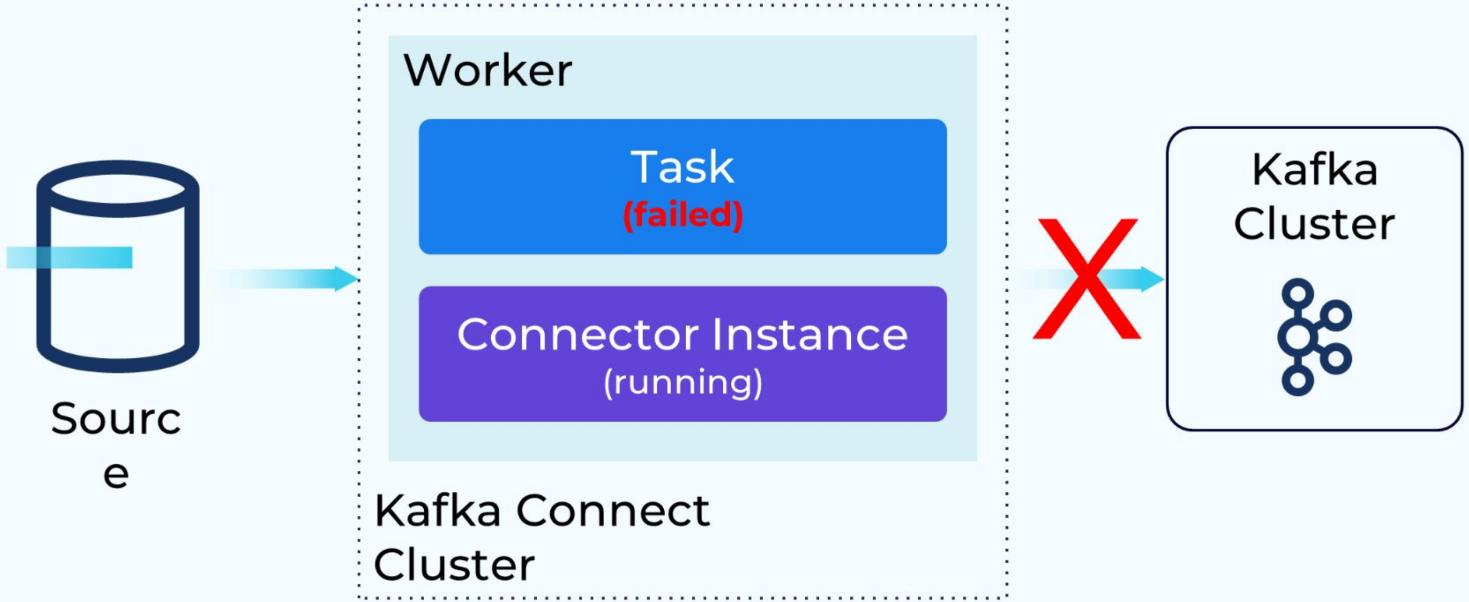
Total in last 7 days

Confluent Connect API

```
GET /connect/v1/environments/{environment_id}/clusters/{kafka_cluster_id}/connectors/{connector_name}/status

{
  "name": "MySqlSinkConnector_0",
  "connector": {
    "state": "FAILED",
    "worker_id": "MySqlSinkConnector_0",
    "trace": "Found nested structure in input data. Ensure that your input events are a flat struct of
primitive fields.\n"
  },
  "tasks": [
    {
      "id": 0,
      "state": "USER_ACTIONABLE_ERROR",
      "worker_id": "MySqlSinkConnector_0",
      "msg": ""
    }
  ],
  "type": "sink"
}
```

Troubleshooting Scenario

CONFLUENT Developer



Getting Connector and Task Status

```
$ curl -s "http://localhost:8083/connectors/jdbc-sink/status" | \
jq '.connector.state'
"RUNNING"
```

```
$ curl -s "http://localhost:8083/connectors/jdbc-sink/status" | \
jq '.tasks[0].state'
"FAILED"
```

Getting Task Status

```
$curl -s "http://localhost:8083/connectors/jdbc-sink/status" | jq '.tasks[0].trace' | sed 's/\\n/\n/g; s/\\t/\t/g'

"org.apache.kafka.connect.errors.ConnectException: Exiting WorkerSinkTask due to unrecoverable exception.
     at org.apache.kafka.connect.runtime.WorkerSinkTask.deliverMessages(WorkerSinkTask.java:618)
     at org.apache.kafka.connect.runtime.WorkerSinkTask.poll(WorkerSinkTask.java:334)
     at org.apache.kafka.connect.runtime.WorkerSinkTask.iteration(WorkerSinkTask.java:235)
     at org.apache.kafka.connect.runtime.WorkerSinkTask.execute(WorkerSinkTask.java:204)
     at org.apache.kafka.connect.runtime.WorkerTask.doRun(WorkerTask.java:200)
     at org.apache.kafka.connect.runtime.WorkerTask.run(WorkerTask.java:255)
     at java.base/java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:515)
     at java.base/java.util.concurrent.FutureTask.run(FutureTask.java:264)
     at java.base/java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1128)
     at java.base/java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:628)
     at java.base/java.lang.Thread.run(Thread.java:829)
Caused by: org.apache.kafka.connect.errors.ConnectException: java.sql.SQLException: No suitable driver found for
jdbc:mysql://localhost/demo
     at io.confluent.connect.jdbc.util.CachedConnectionProvider.getConnection(CachedConnectionProvider.java:59)
     at io.confluent.connect.jdbc.sink.JdbcDbWriter.write(JdbcDbWriter.java:64)
     at io.confluent.connect.jdbc.sink.JdbcSinkTask.put(JdbcSinkTask.java:84)
     at org.apache.kafka.connect.runtime.WorkerSinkTask.deliverMessages(WorkerSinkTask.java:584)
     ... 10 more
Caused by: java.sql.SQLException: No suitable driver found for jdbc:mysql://localhost/demo
     at java.sql/java.sql.DriverManager.getConnection(DriverManager.java:702)
     at java.sql/java.sql.DriverManager.getConnection(DriverManager.java:189)
     at io.confluent.connect.jdbc.dialect.GenericDatabaseDialect.getConnection(GenericDatabaseDialect.java:247)
     at io.confluent.connect.jdbc.util.CachedConnectionProvider.newConnection(CachedConnectionProvider.java:80)
     at io.confluent.connect.jdbc.util.CachedConnectionProvider.getConnection(CachedConnectionProvider.java:52)
     ... 13 more
"
```

Kafka Connect Log4j Logging

- The log is the source of truth

```
$ confluent local services connect log

$ docker-compose logs kafka-connect

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

- The Log4j properties files controls what is logged, the log message layout, and where log files are stored

```
/etc/kafka/connect-log4j.properties (default location)
```

Identify the Problem Cause

```
[2022-07-19 23:57:28,600] ERROR [jdbc-sink|task-0] WorkerSinkTask{id=jdbc-sink-0} Task threw an uncaught and
unrecoverable exception. Task is being killed and will not recover until manually restarted
(org.apache.kafka.connect.runtime.WorkerTask:207)
org.apache.kafka.connect.errors.ConnectException: Exiting WorkerSinkTask due to unrecovera      ception.
    at org.apache.kafka.connect.runtime.WorkerSinkTask.deliverMessages(WorkerSinkTa
    at org.apache.kafka.connect.runtime.WorkerSinkTask.poll(WorkerSinkTask.java:33
    at org.apache.kafka.connect.runtime.WorkerSinkTask.iteration(WorkerSinkTask.ja      )
    at org.apache.kafka.connect.runtime.WorkerSinkTask.execute(WorkerSinkTask.java:204)
    at org.apache.kafka.connect.runtime.WorkerTask.doRun(WorkerTask.java:200)
    at org.apache.kafka.connect.runtime.WorkerTask.run(WorkerTask.java:255)
    at java.base/java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:515)
    at java.base/java.util.concurrent.FutureTask.run(FutureTask.java:264)
    at java.base/java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1128)
    at java.base/java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:628)
    at java.base/java.lang.Thread.run(Thread.java:829)
Caused by: org.apache.kafka.connect.errors.ConnectException: java.sql.SQLException: No suitable driver found for
jdbc:mysql://localhost/demo
    at io.confluent.connect.jdbc.util.CachedConnectionProvider.getConnection(CachedConne   ionProvider.java:59)
    at io.confluent.connect.jdbc.sink.JdbcDbWriter.write(JdbcDbWriter.java:64)
    at io.confluent.connect.jdbc.sink.JdbcSinkTask.put(JdbcSinkTask.java:84)
    at org.apache.kafka.connect.runtime.WorkerSinkTask.deliverMessages(WorkerSinkTa      4)
    ... 10 more
Caused by: java.sql.SQLException: No suitable driver found for jdbc:mysql://localhost/demo
    at java.sql/java.sql.DriverManager.getConnection(DriverManager.java:702)
    at java.sql/java.sql.DriverManager.getConnection(DriverManager.java:189)
    at io.confluent.connect.jdbc.dialect.GenericDatabaseDialect.getConnection(GenericDatabaseDialect.java:247)
    at io.confluent.connect.jdbc.util.CachedConnectionProvider.newConnection(CachedConnectionProvider.java:80)
    at io.confluent.connect.jdbc.util.CachedConnectionProvider.getConnection(CachedConnectionProvider.java:52)
    ... 13 more
```

Symptom, not the cause

Possible causes

Summary

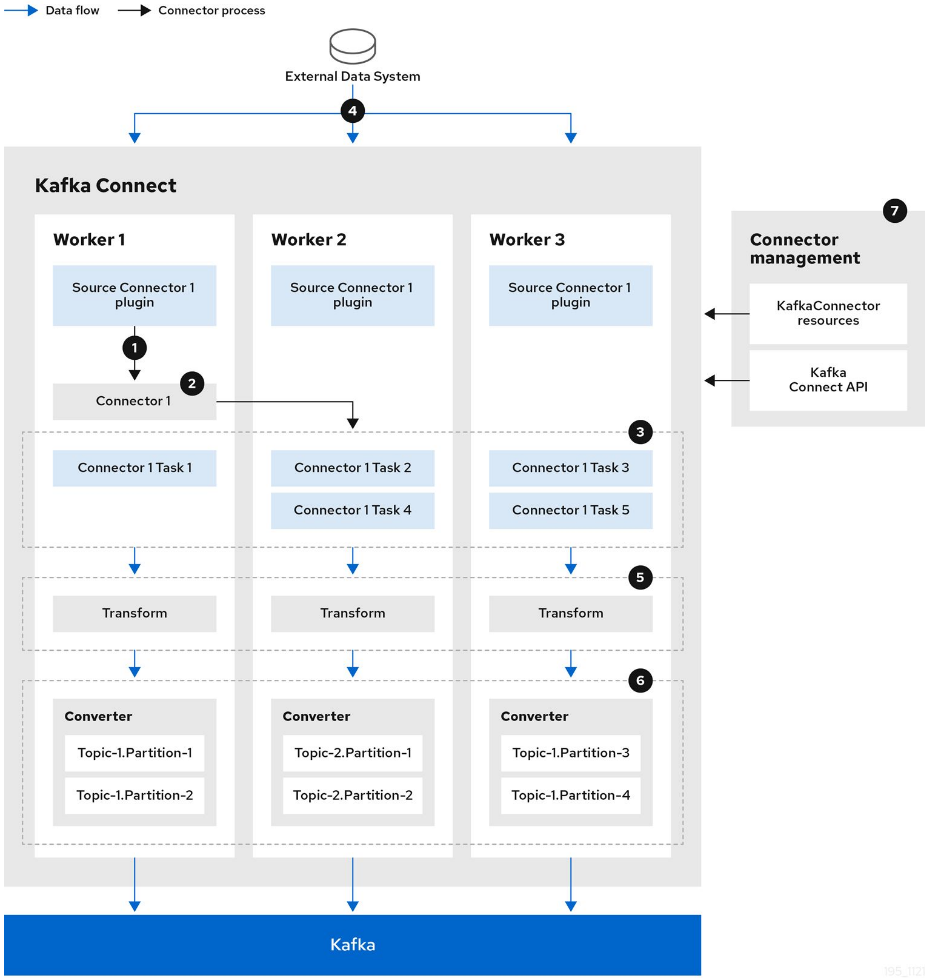
The main Kafka Connect components used in streaming data are as follows:

- Connectors to create tasks
- Tasks to move data
- Workers to run tasks
- Transforms to manipulate data
- Converters to convert data

Connectors can be one of the following type:

- Source connectors that push data into Kafka
- Sink connectors that extract data out of Kafka

Source connector streaming data to Kafka



Data flow

Connector process

External Data System

4

Kafka Connect

Worker 1

Source Connector 1 plugin

1

2

Connector 1

Connector 1 Task 1

Transform

Converter

Topic-1.Partition-1

Topic-1.Partition-2

Worker 2

Source Connector 1 plugin

Connector 1 Task 2

Connector 1 Task 4

Transform

Converter

Topic-2.Partition-1

Topic-2.Partition-2

Worker 3

Source Connector 1 plugin

3

Connector 1 Task 3

Connector 1 Task 5

5

Transform

6

Converter

Topic-1.Partition-3

Topic-1.Partition-4

7

Connector management

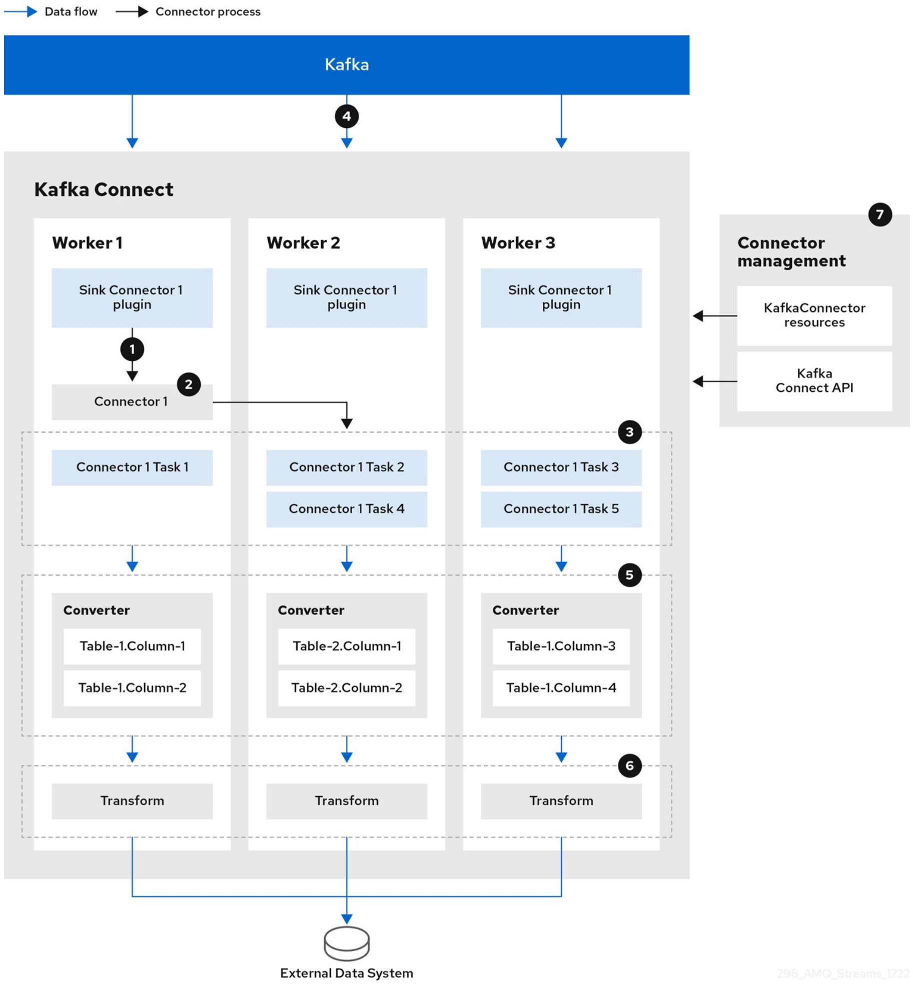
KafkaConnector resources

Kafka Connect API

Kafka

195_1121

1. A plugin provides the implementation artifacts for the source connector
2. A single worker initiates the source connector instance
3. The source connector creates the tasks to stream data
4. Tasks run in parallel to poll the external data system and return records
5. Transforms adjust the records, such as filtering or relabelling them
6. Converters put the records into a format suitable for Kafka
7. The source connector is managed using KafkaConnectors or the Kafka Connect API



Sink connector streaming data from Kafka

1. A plugin provides the implementation artifacts for the sink connector
2. A single worker initiates the sink connector instance
3. The sink connector creates the tasks to stream data
4. Tasks run in parallel to poll Kafka and return records
5. Converters put the records into a format suitable for the external data system
6. Transforms adjust the records, such as filtering or relabelling them
7. The sink connector is managed using KafkaConnectors or the Kafka Connect API