



Connecting Data to and from Kafka

Daniel Hinojosa

Introduction

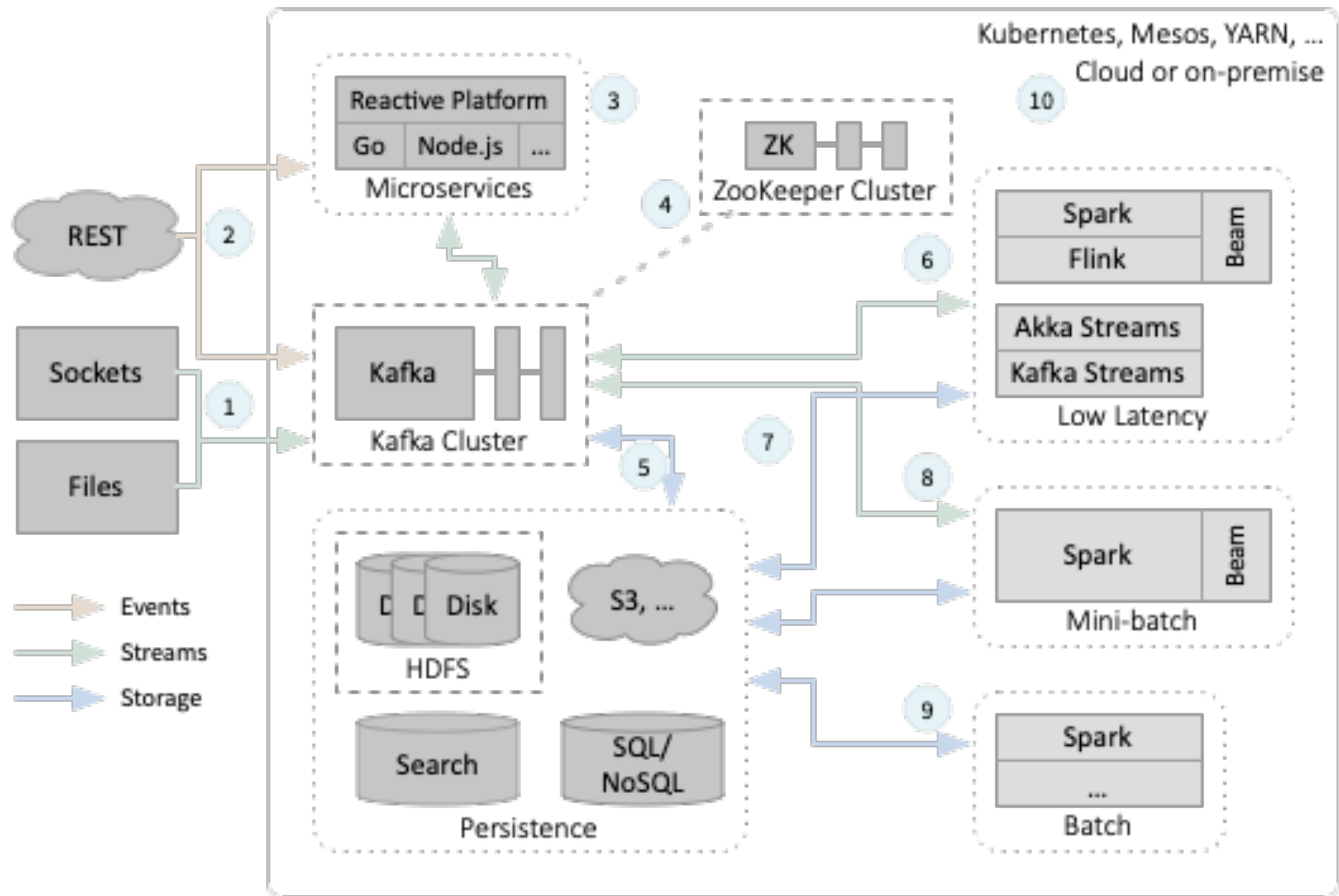
Daniel Hinojosa
Programmer, Consultant, Trainer

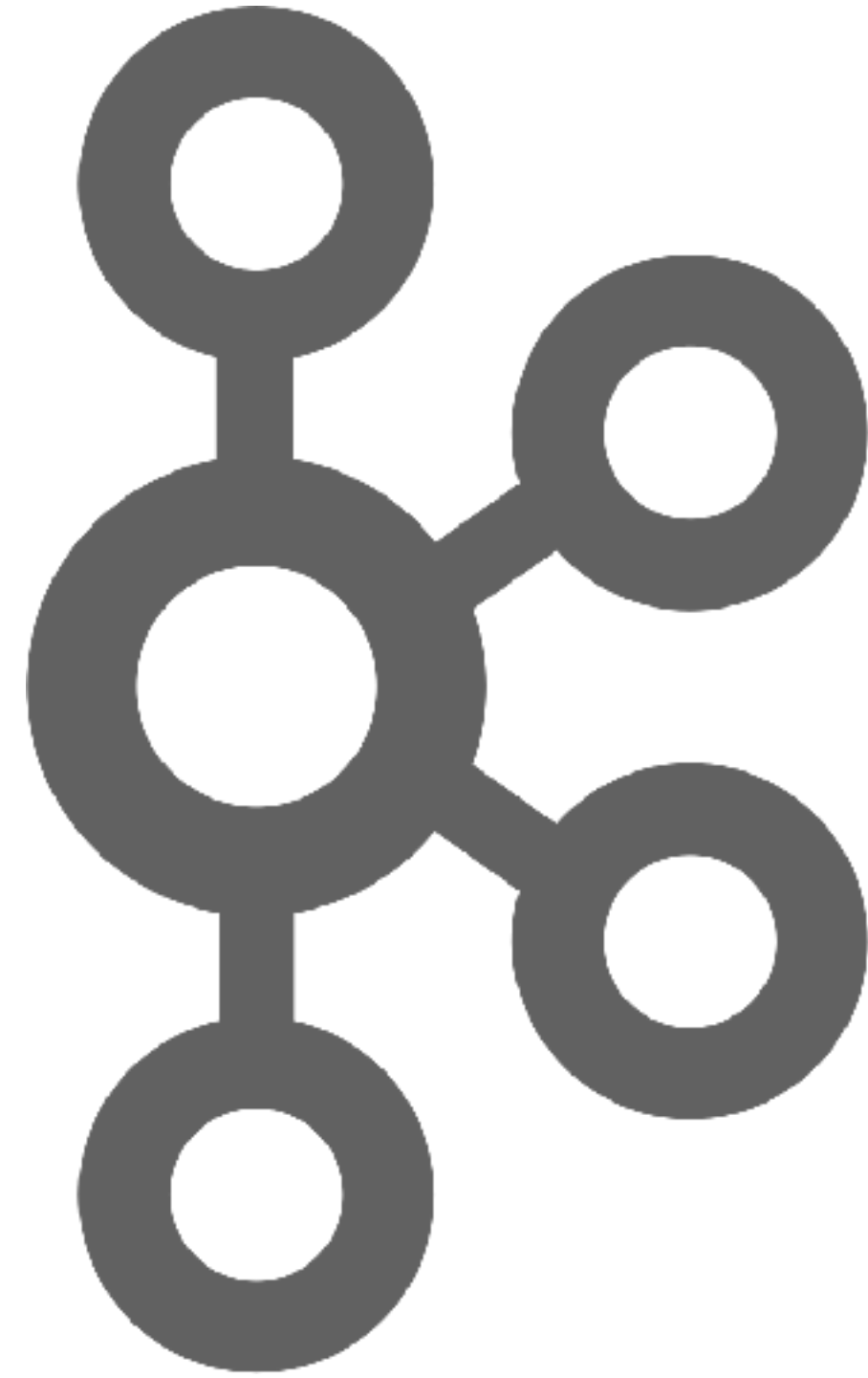
Testing in Scala (Book)
Beginning Scala Programming
(Video)
Scala Beyond the Basics (Video)

Contact:
dhinojosa@evolutionnext.com
@dhinojosa

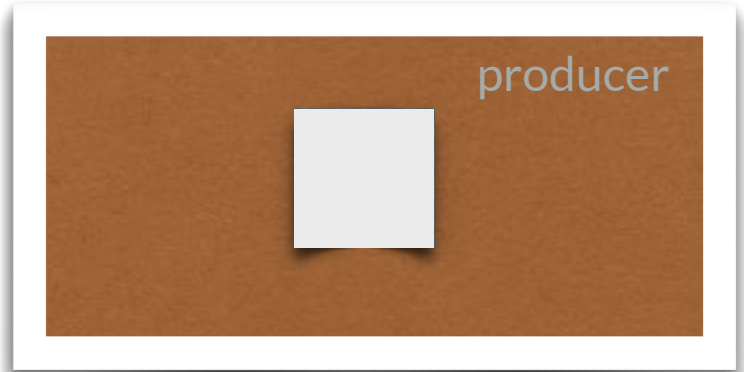


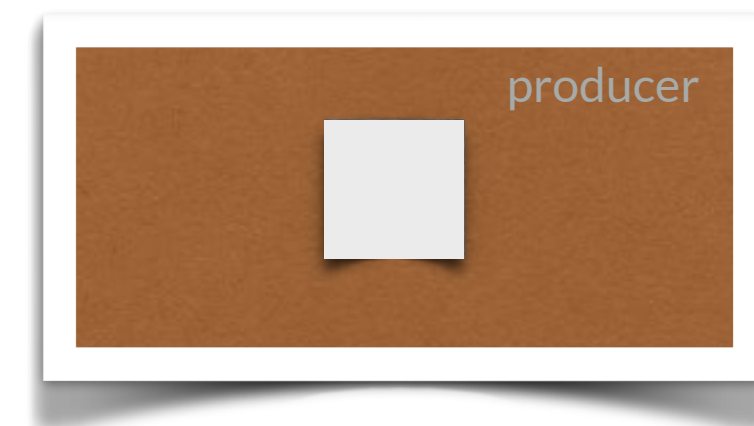
Source Code for this Presentation:
<https://github.com/dhinojosa/kafka-connect-study>





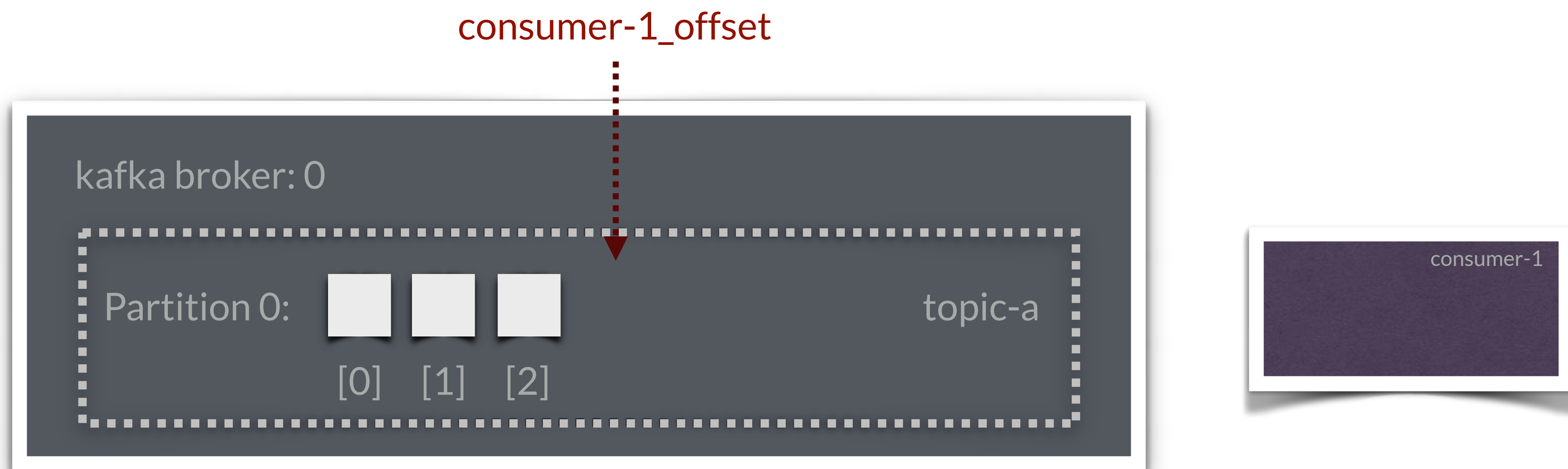
Quick Review of Kafka

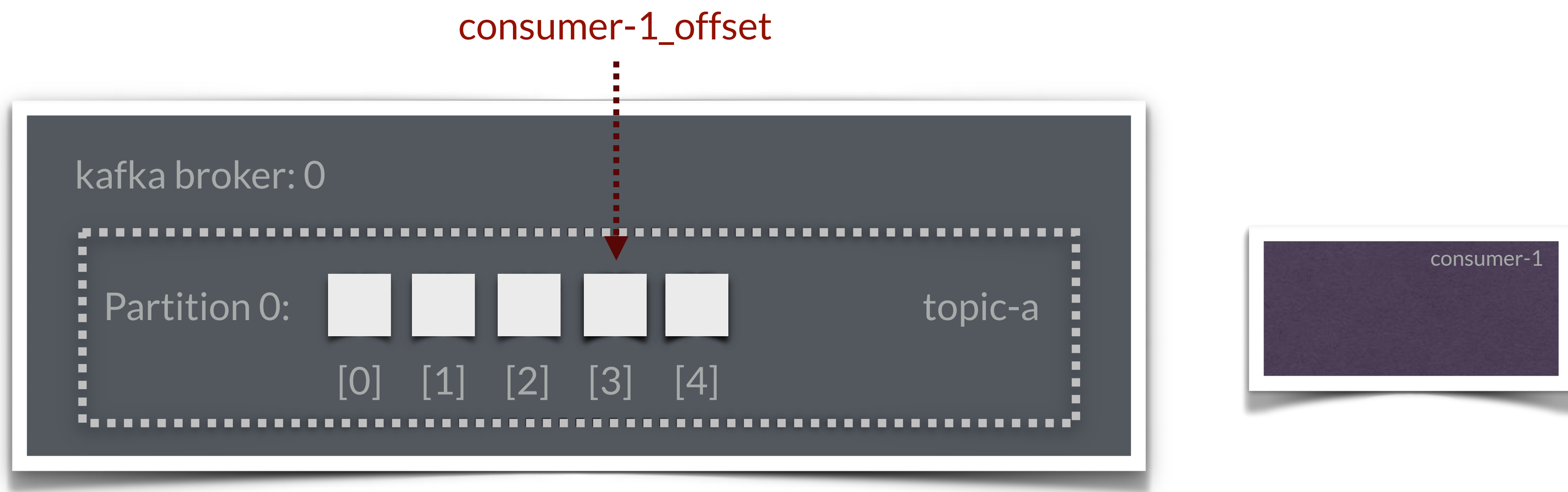




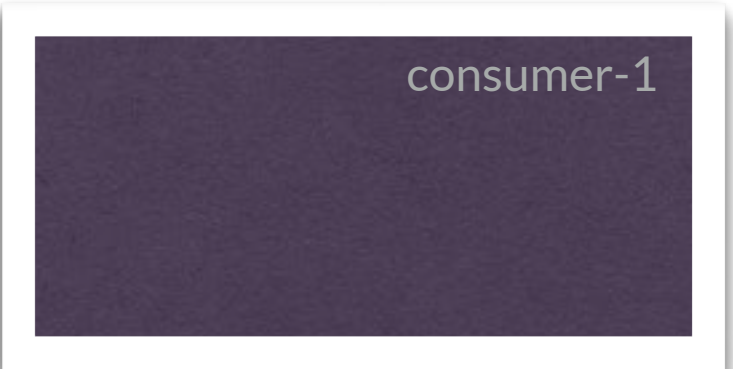
Retention: The data is temporary

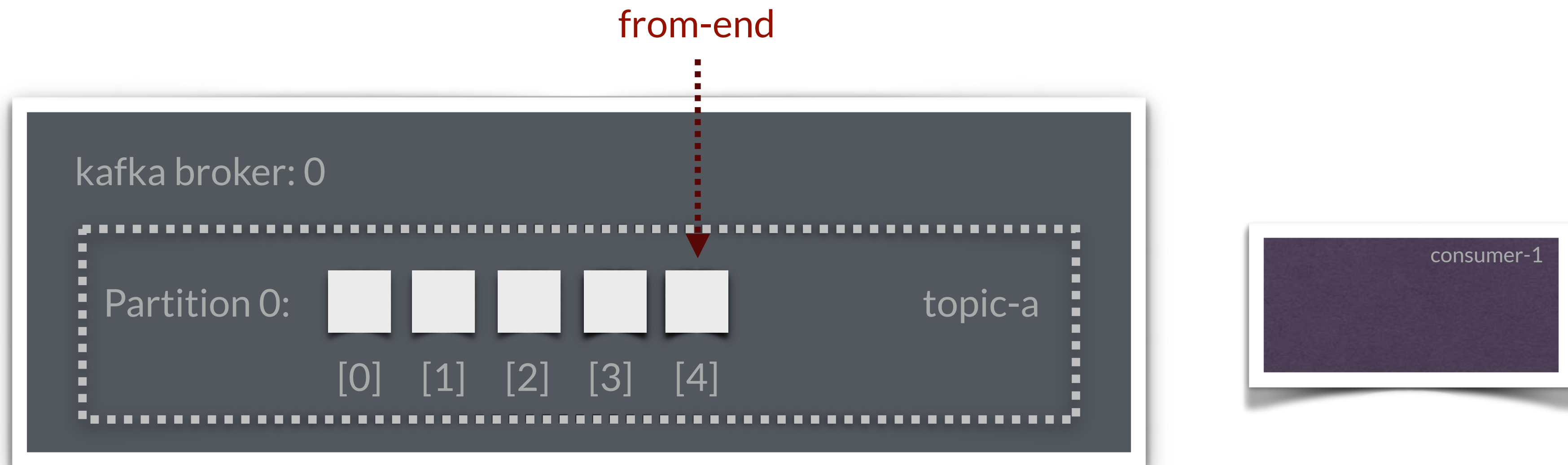
**How messages are
consumed?**

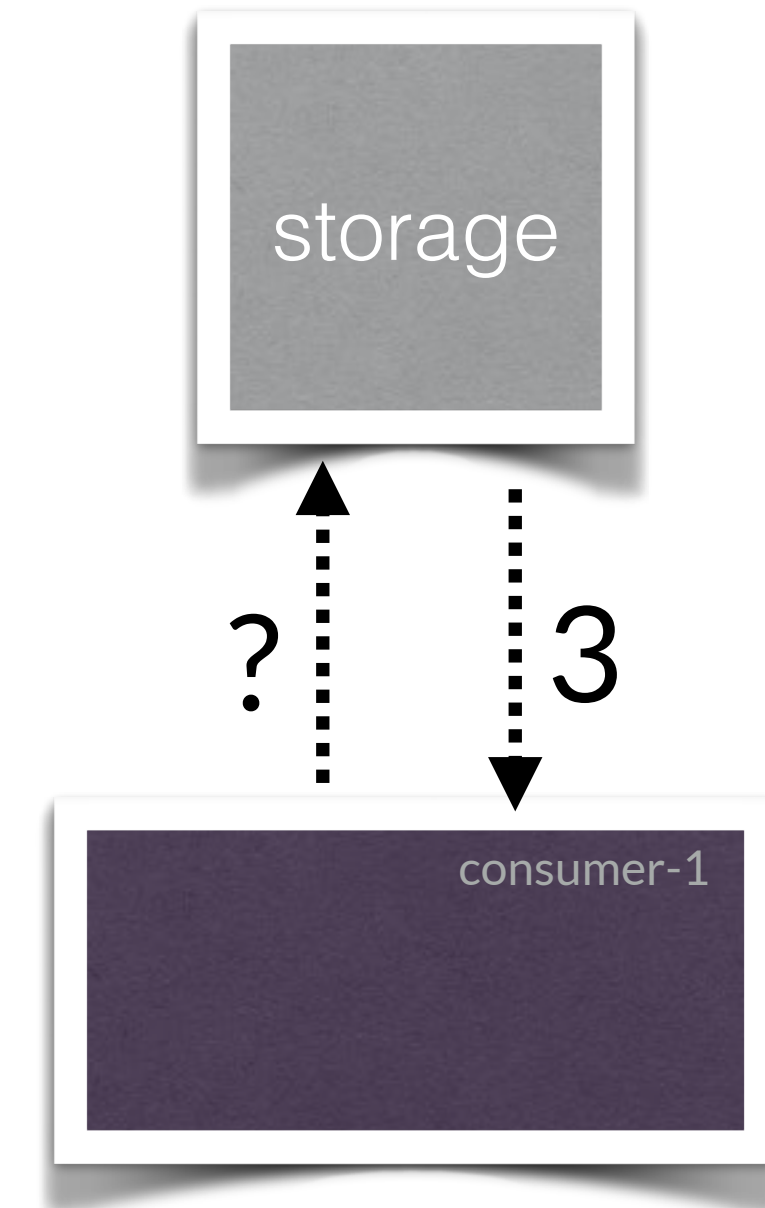
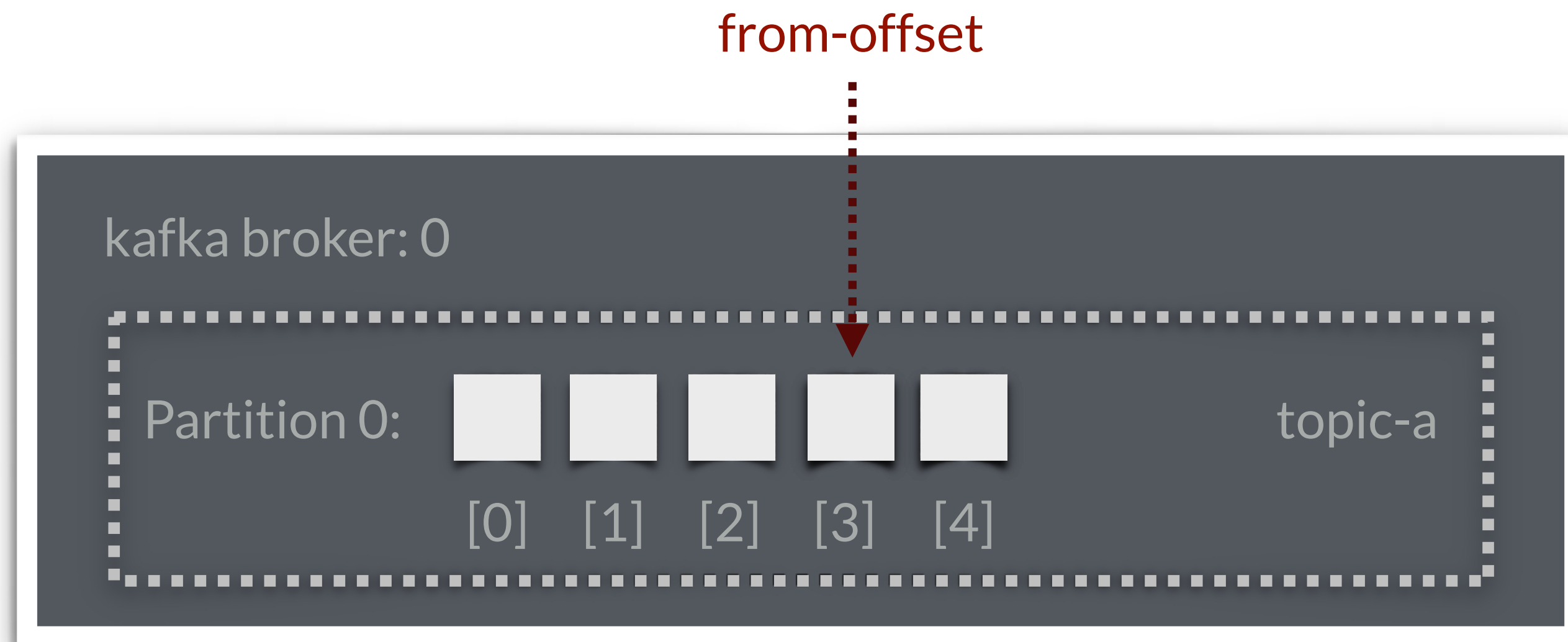


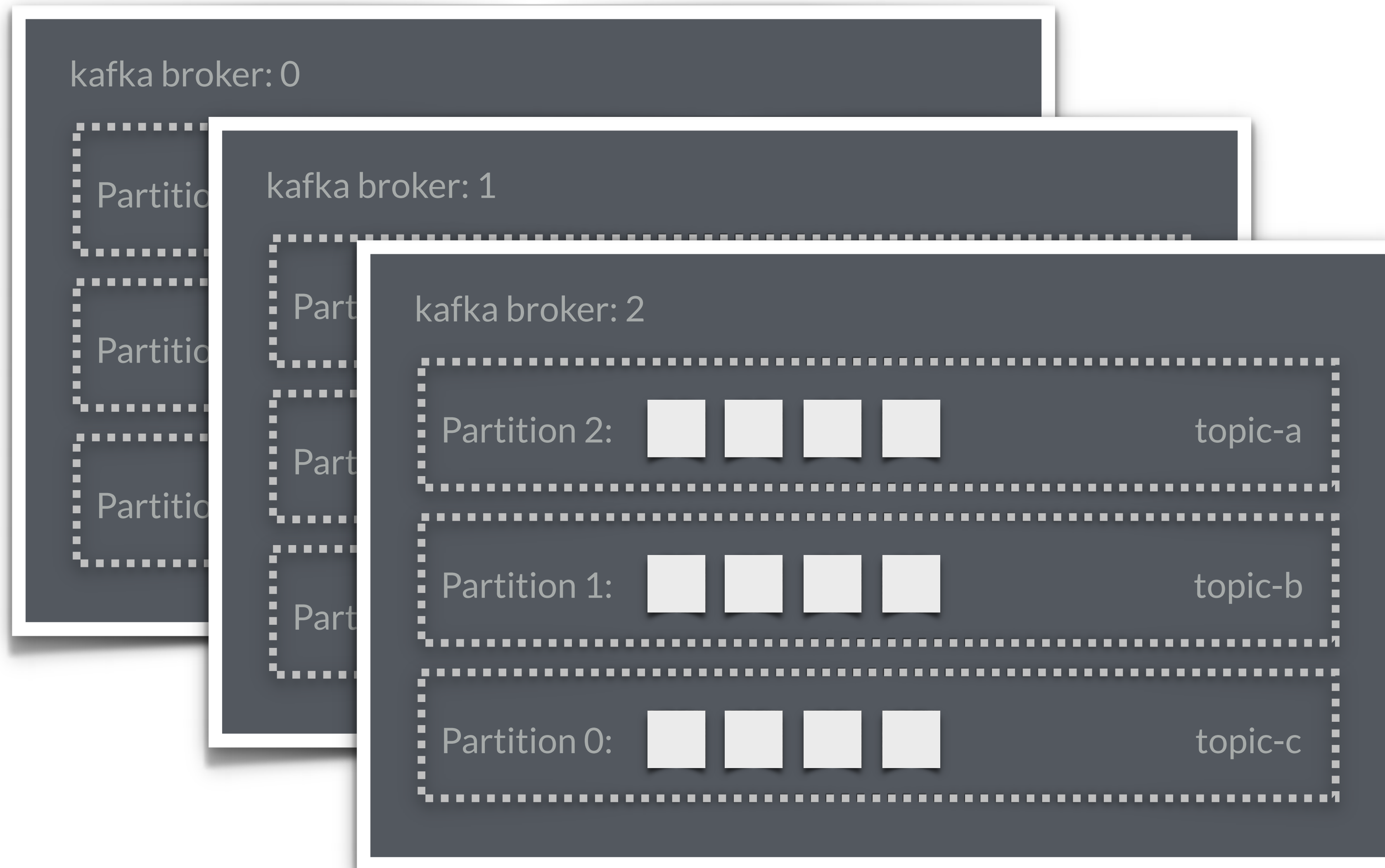


from-beginning









Each partition is on a different broker,
therefore a single topic is scaled

kafka bro

Partition

Partition

Partition

Partition

Partition

kafka brok

Partition

Partition

Partition

Partition

Partition


kafka broker: 2

Partition 0:  topic-a

Partition 2:  topic-a

Partition 1:  topic-b

Partition 0:  topic-c

Partition 2:  topic-c

Partition 1:  topic-a

Partition 0:  topic-b

Partition 2:  topic-b

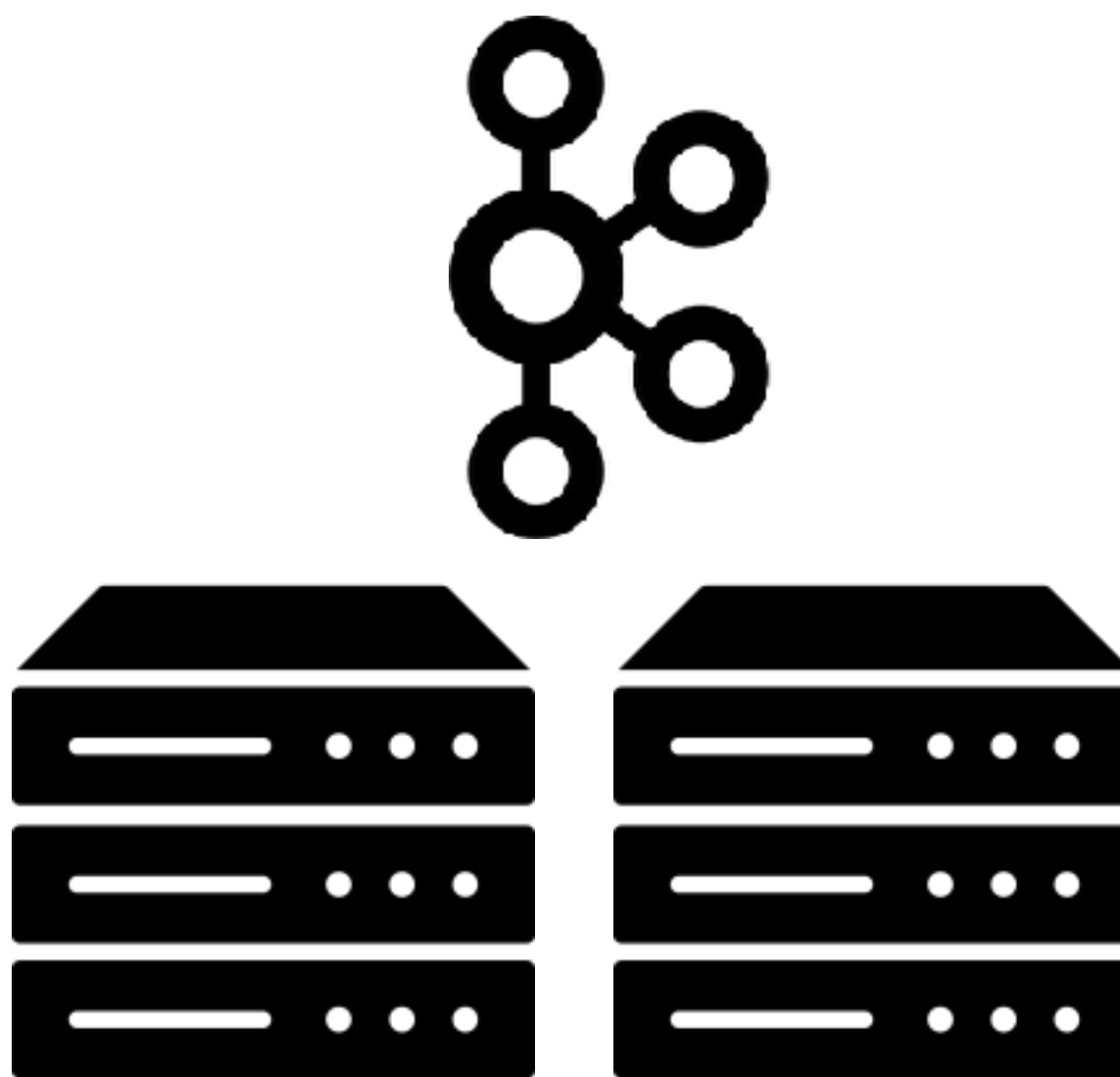
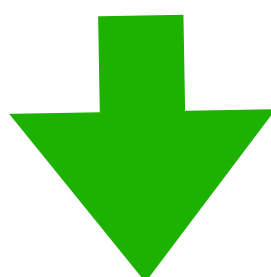
Partition 1:  topic-c

Kafka Connect

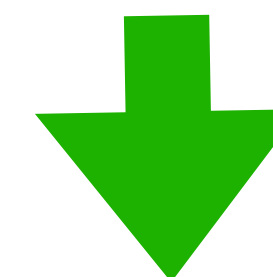
- Automatic Scaling
- Fault Tolerance
- All Configuration
- Pre-Engineered
- Transformations
- Confluent Community License

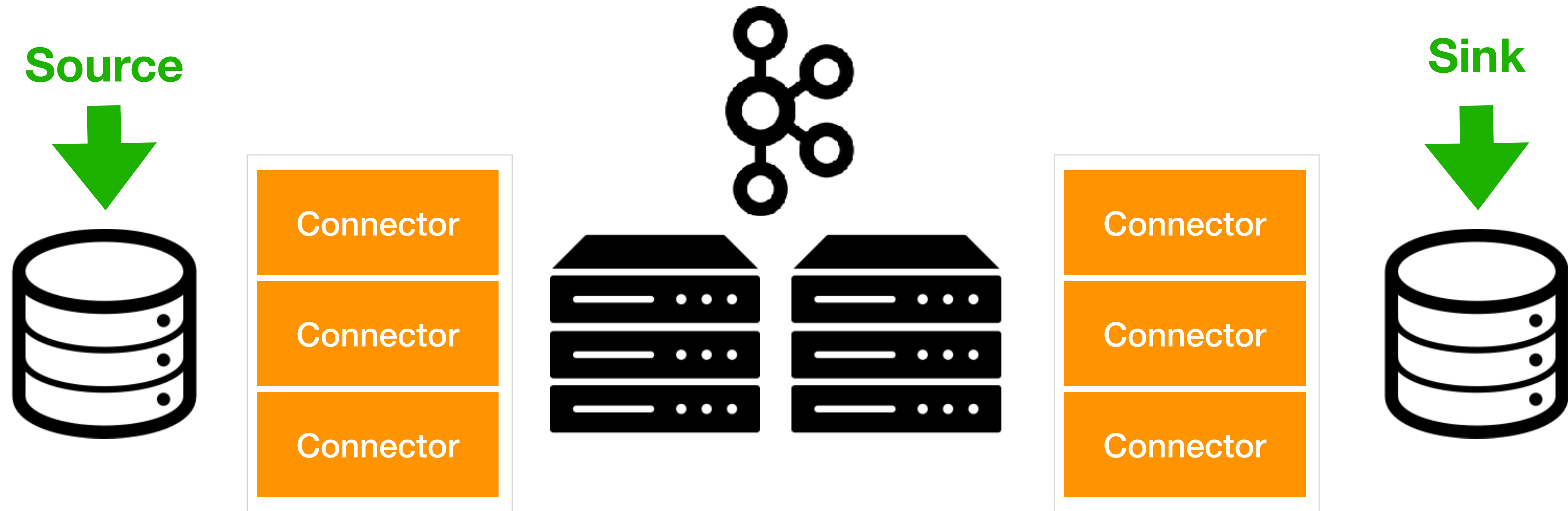


Source

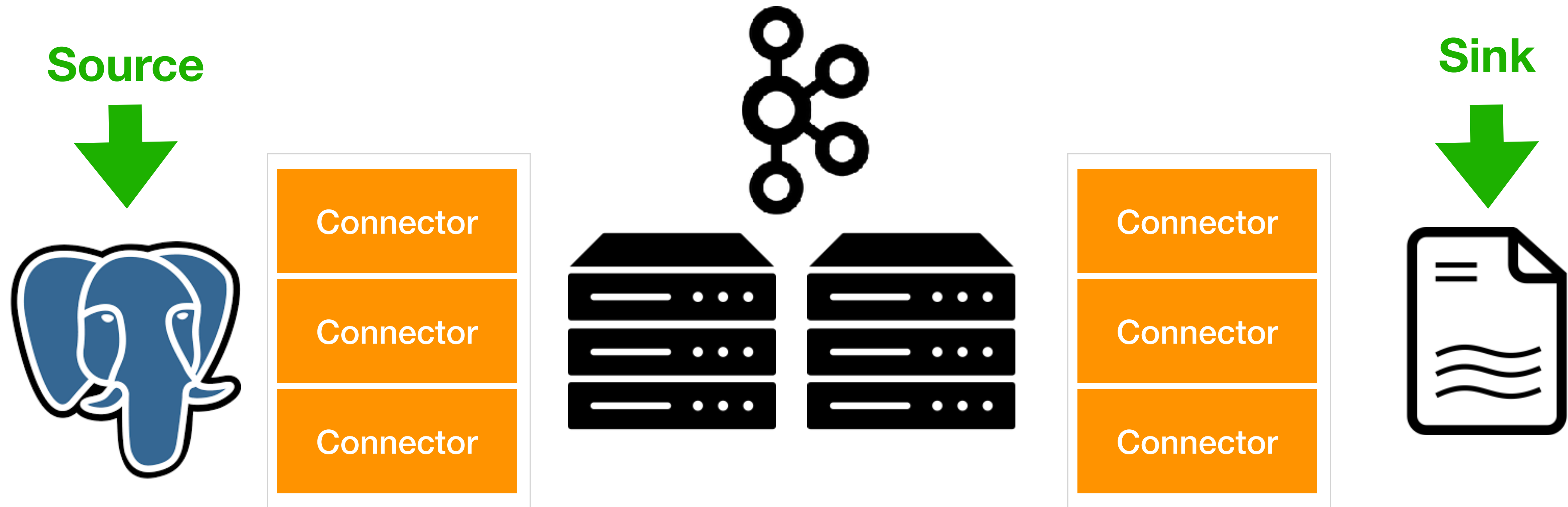


Sink

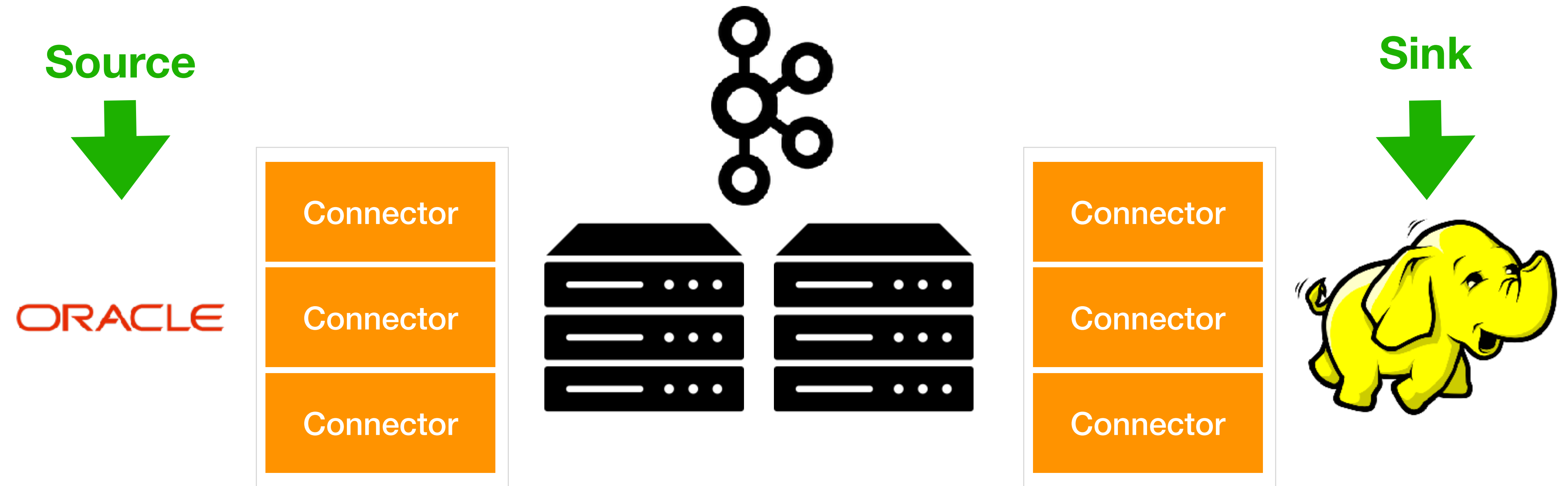





Kafka Connect is an Open Source Framework to stream data to and from Kafka











Connect a PostgreSQL database to a single file



Connect an Oracle database to Hadoop



ProductCloudDevelopersBlogDocsDownload

<div>Verified Standard</div> <div></div> <div>CockroachDB Change Data Capture Cockroach Labs Read More</div>	<div>Verified Standard</div> <div></div> <div>Vertica Analytics Platform Vertica Read More</div>	<div>Confluent Supported</div> <div></div> <div>Debezium MySQL CDC Connector Debezium Community Read More</div>	<div>Confluent Supported</div> <div></div> <div>Debezium PostgreSQL CDC Connector Debezium Community Read More</div>
<div>Confluent Supported</div> <div></div> <div>Debezium SQL Server CDC Connector Debezium Community Read More</div>	<div>Confluent Supported</div> <div></div> <div>Debezium MongoDB CDC Connector Debezium Community Read More</div>	<div>Confluent Supported</div> <div></div> <div>Kafka Connect ActiveMQ Source Confluent, Inc. Read More</div>	<div>Confluent Supported</div> <div></div> <div>Kafka Connect AWS CloudWatch Logs Source Connector Confluent, Inc. Read More</div>

<https://confluent.io/hub>

Connectors



Connector

Connector

Connector

- **Logical Jobs** that copy the data to and from databases
- Internally a *source connector* is a Producer
- Internally a *sink connector* is a Consumer



Kafka Connectors

Confluent Open Source Connectors

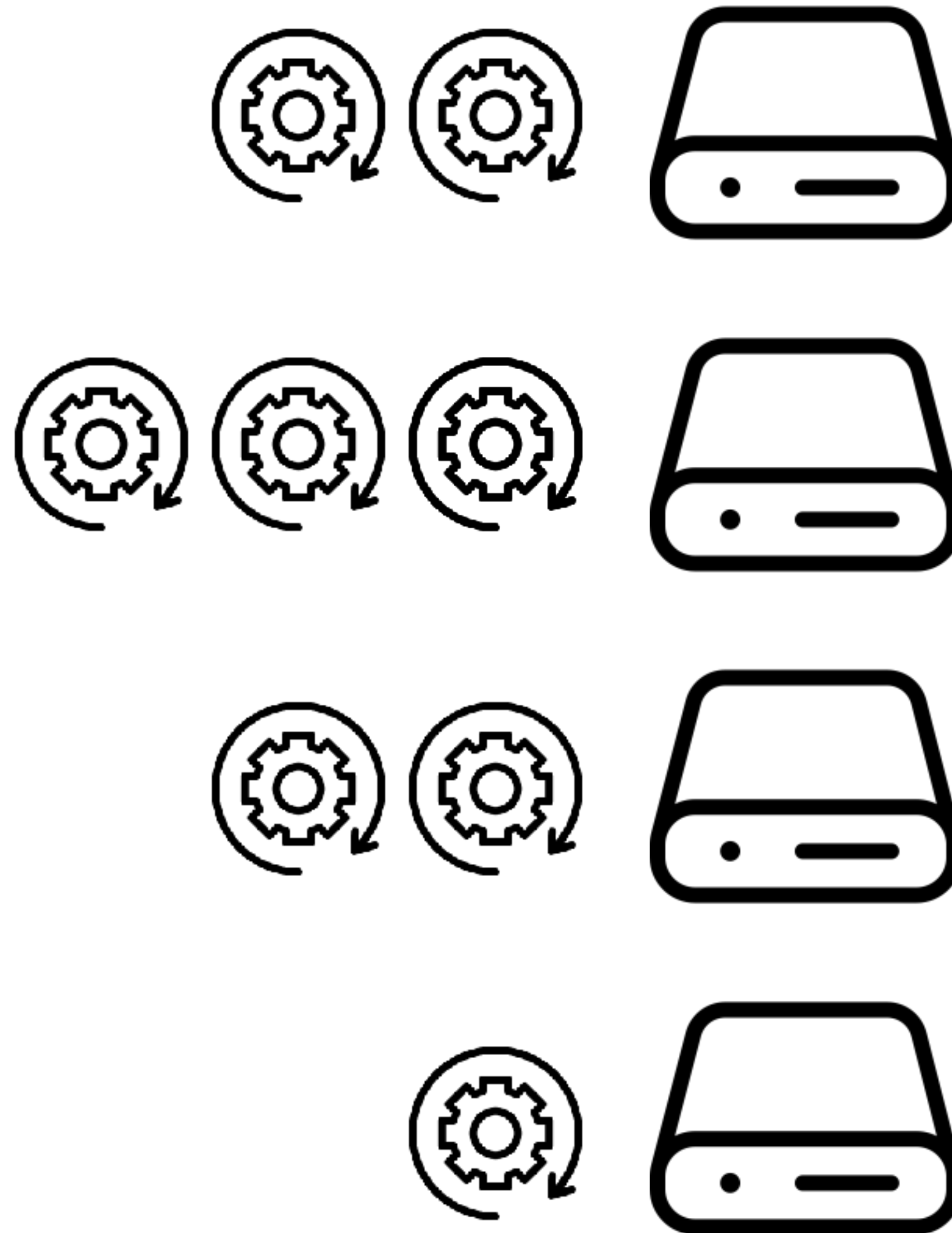
File Stream - JDBC - HDFS - S3 - Elastic Search

Confluent Enterprise

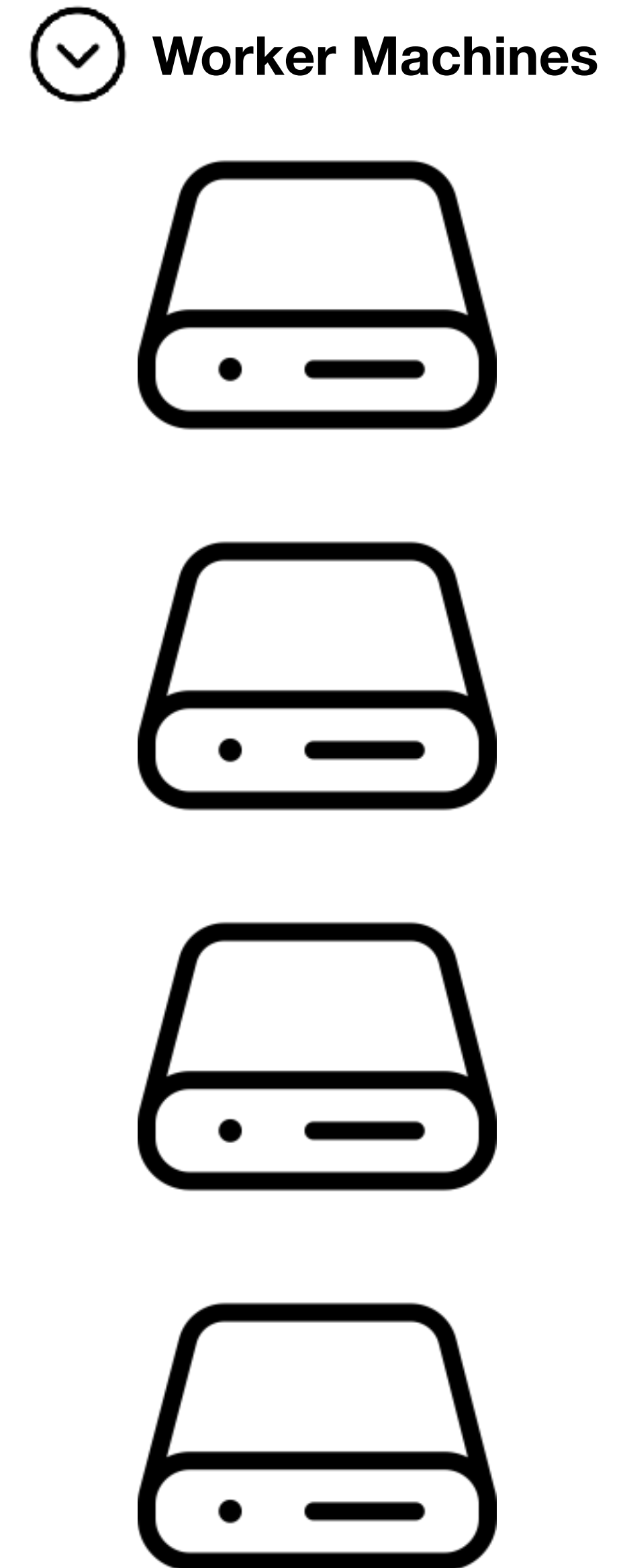
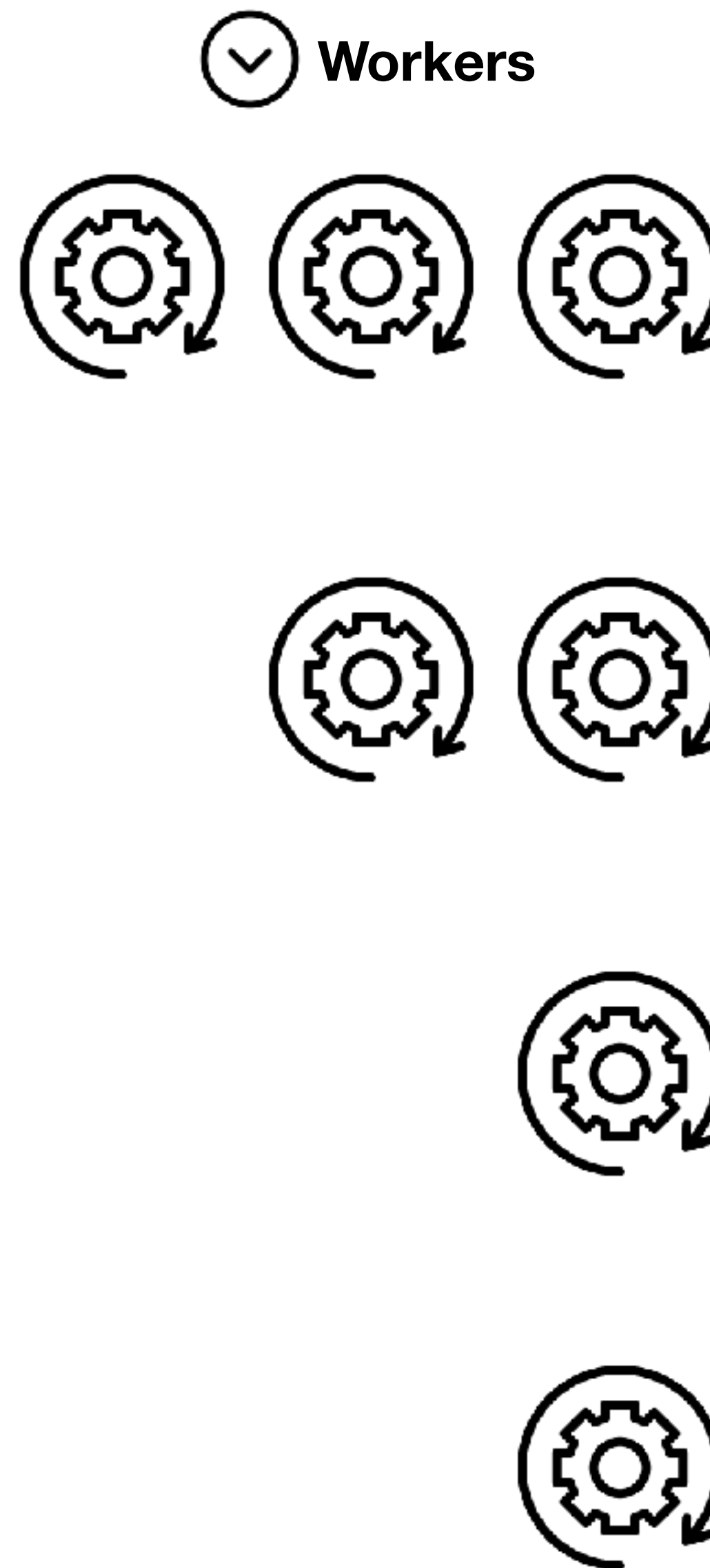
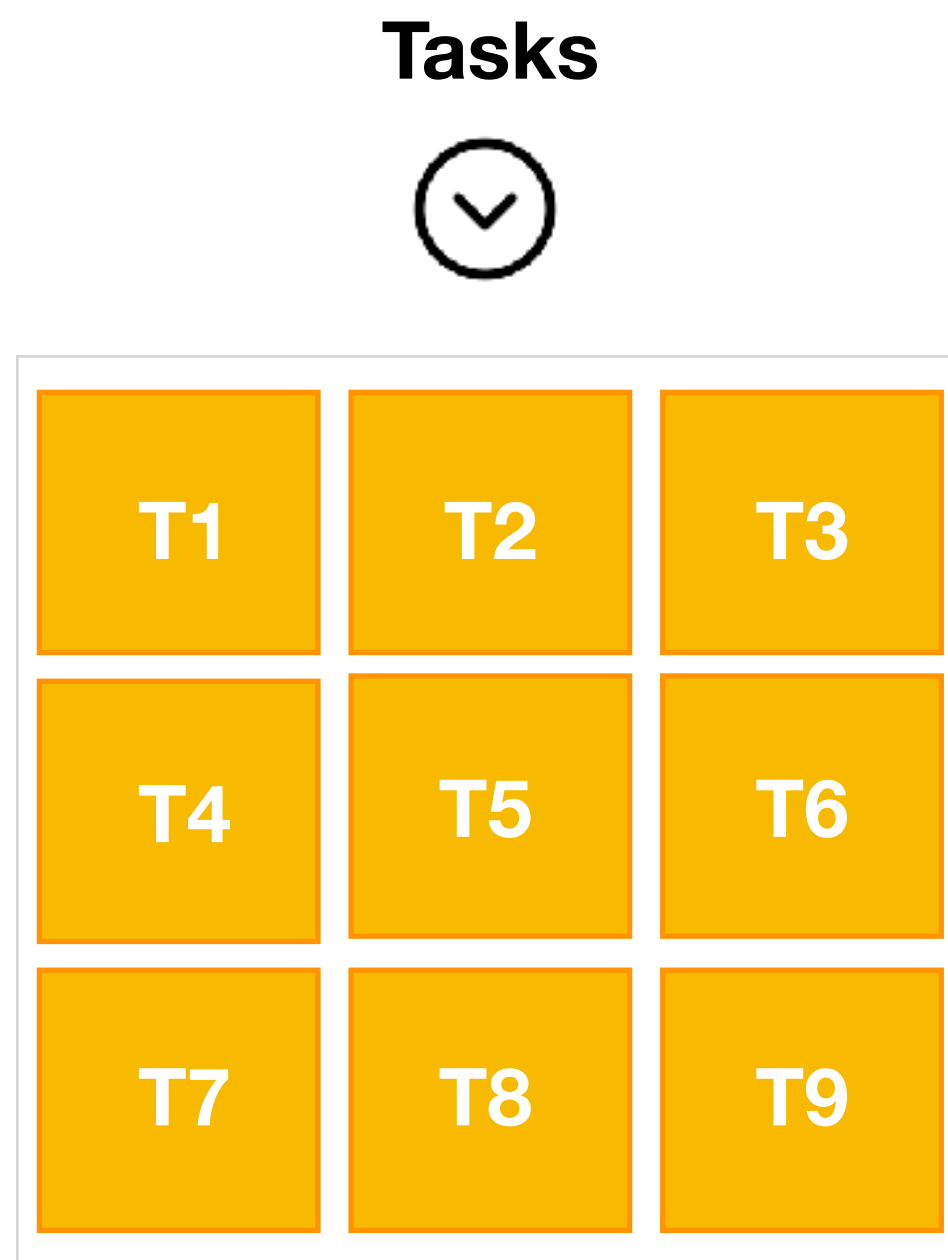
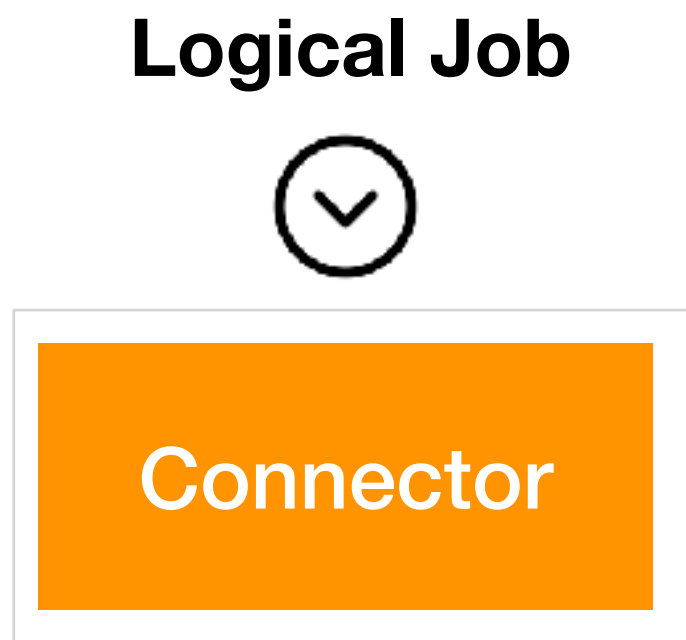
Replicator - GCS - JMS - IBM MQ - Active MQ - Cassandra

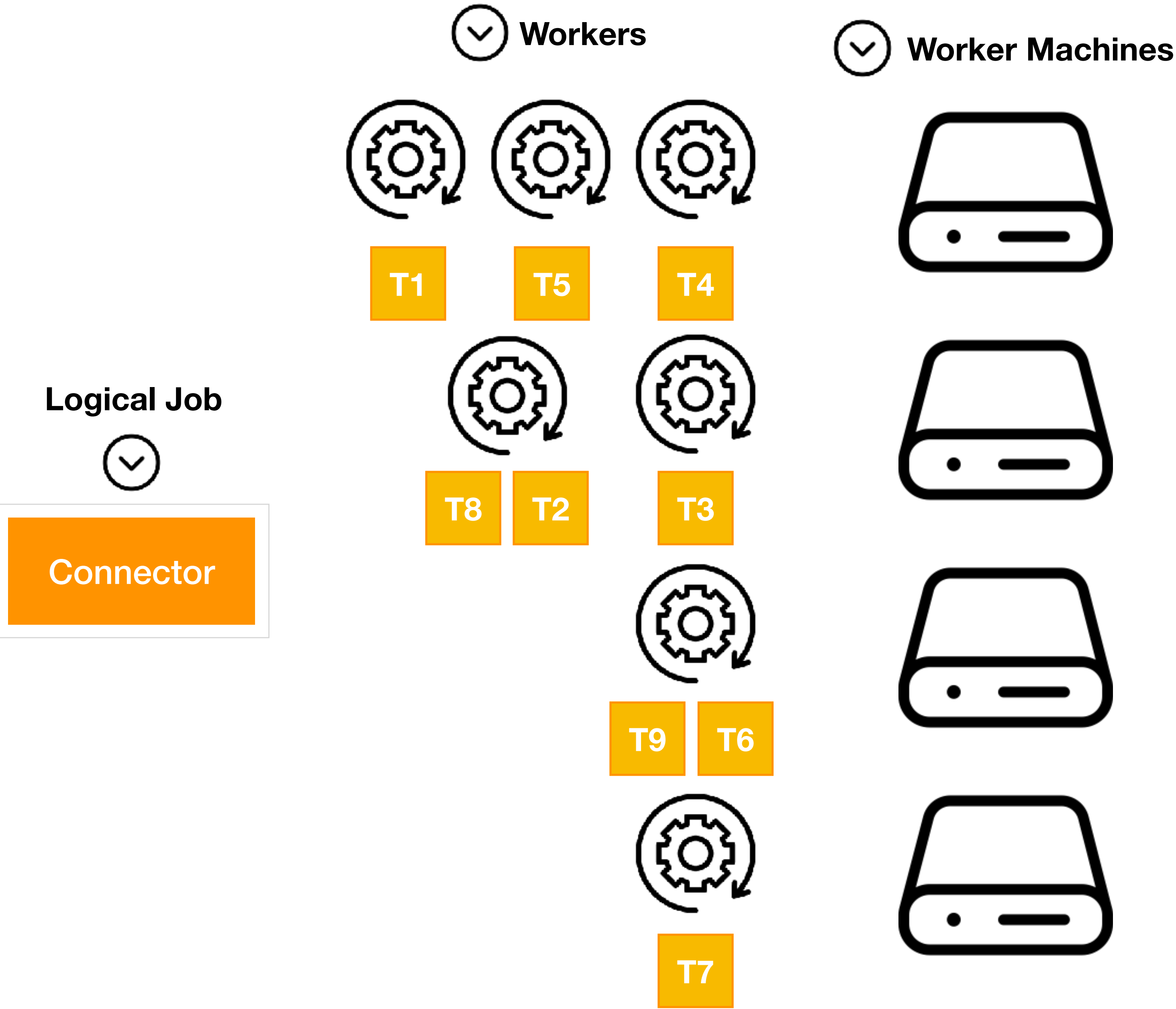
Architecture

Workers >



< Worker Machines

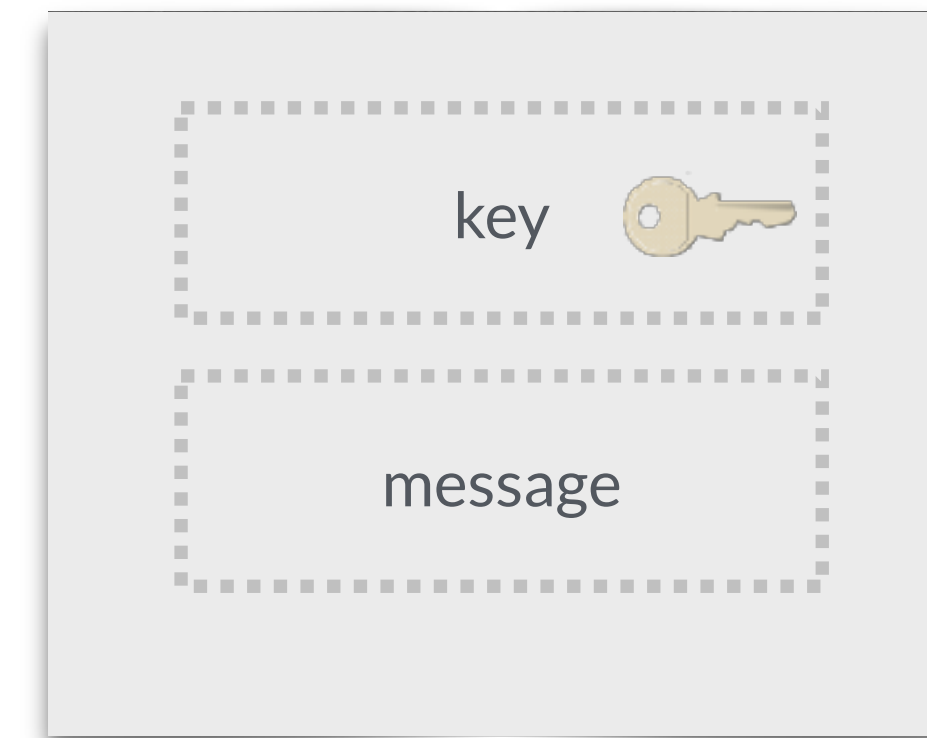




Source Architecture

Kafka Messages

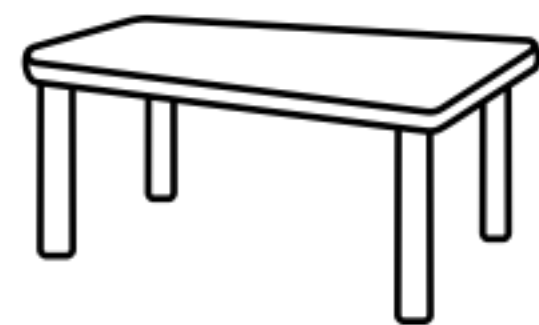
- Message may contain a *key* for better distribution to partitions
- The *key* is also an array of bytes
- If a *key* is provided, a partitioner will hash the key and map it to a single partition
- Therefore it is the only time that something is guaranteed to be in order



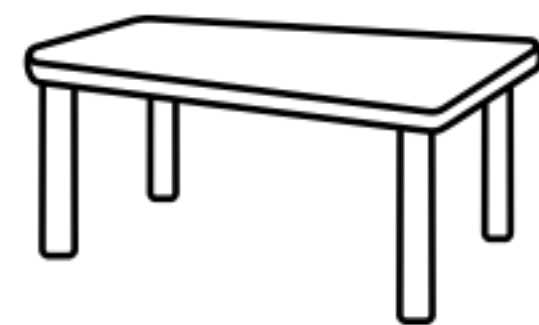
A-E



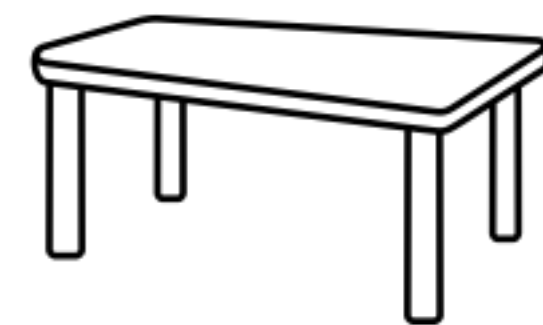
F-K




L-S



T-Z

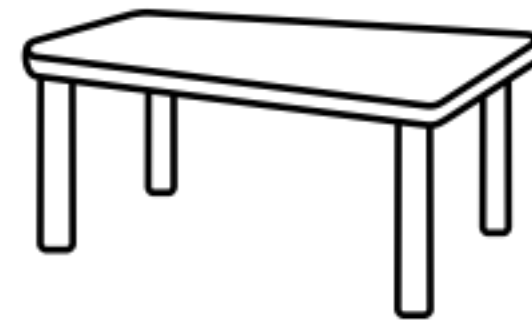


 **murmur2 % partitions**

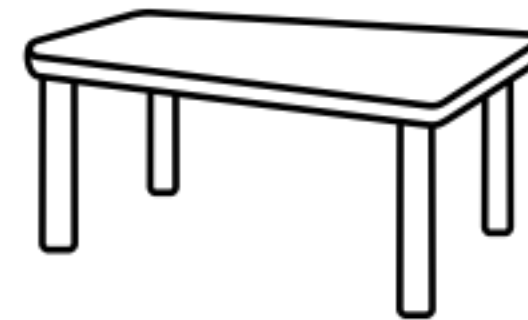
MSFT, AMZN



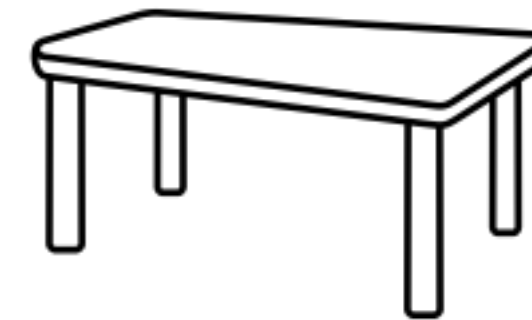
AAPL, F



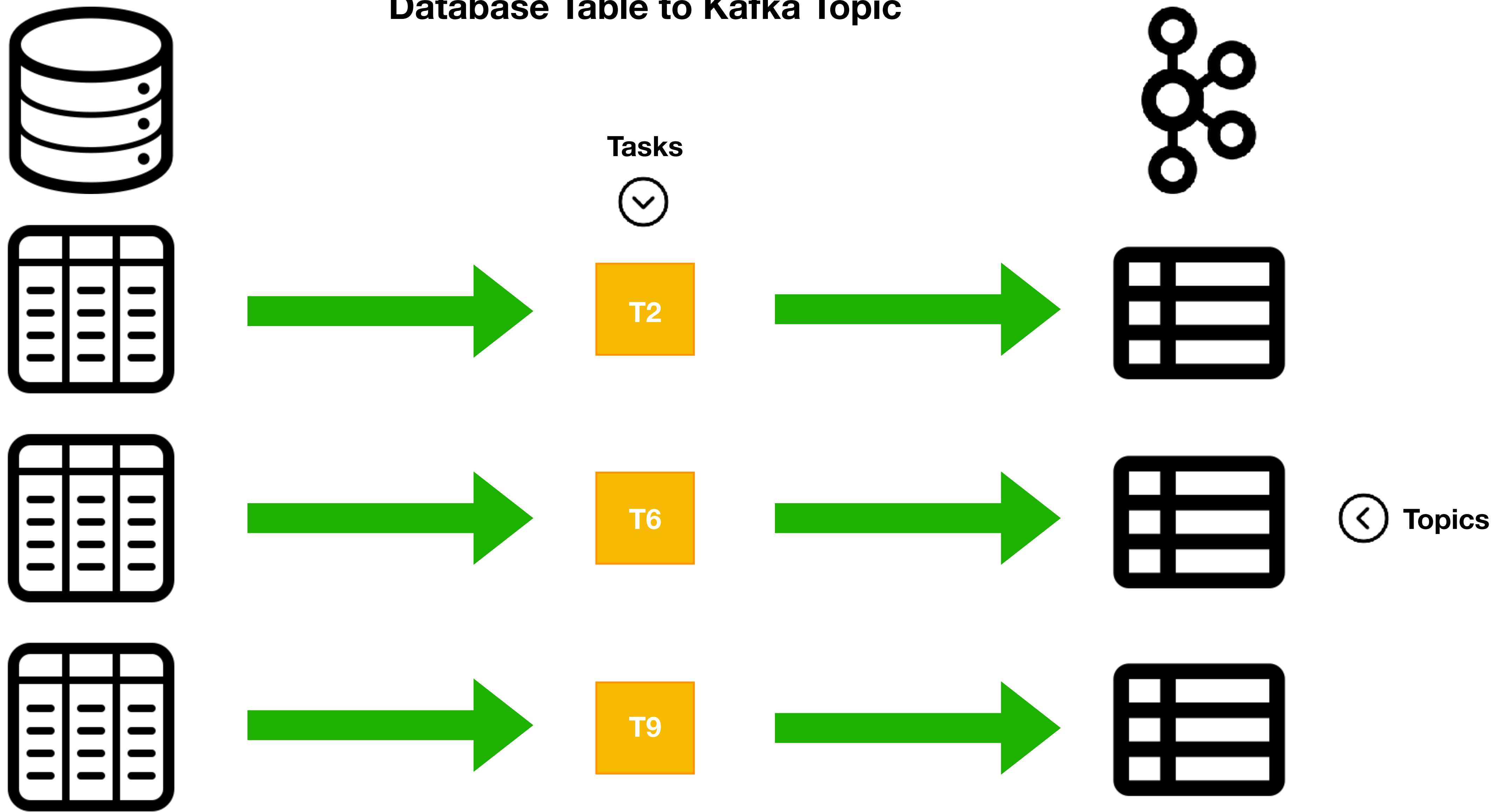
AXP, XOM,T



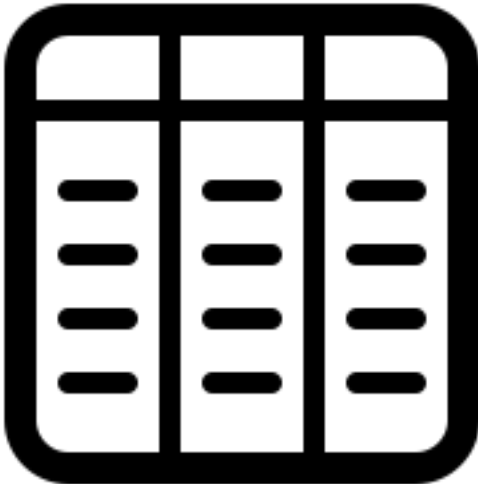
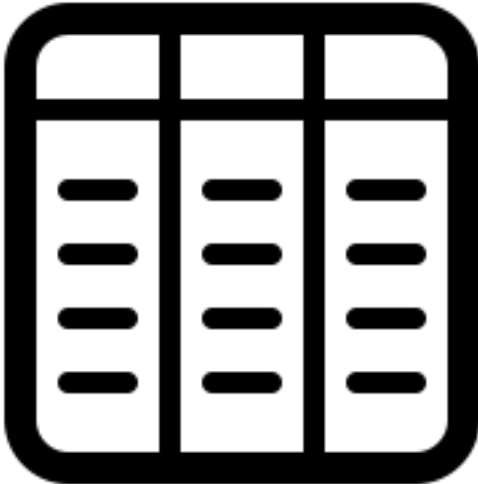
TWX, NOK, CSCO



Database Table to Kafka Topic



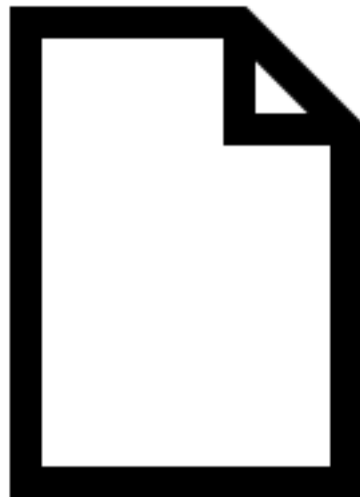
Partitions



Tasks



Partitions



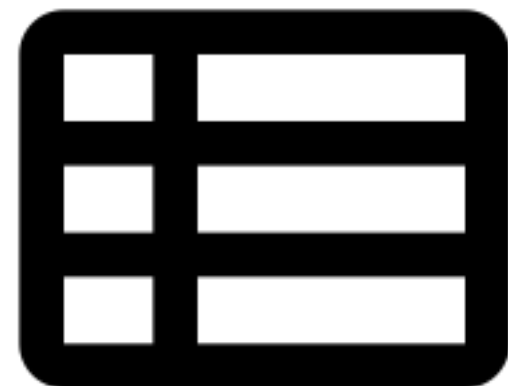
Tasks



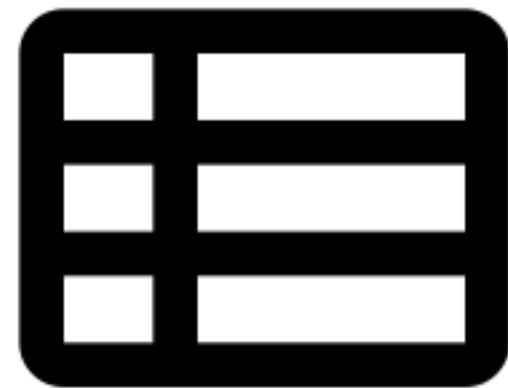
Tasks



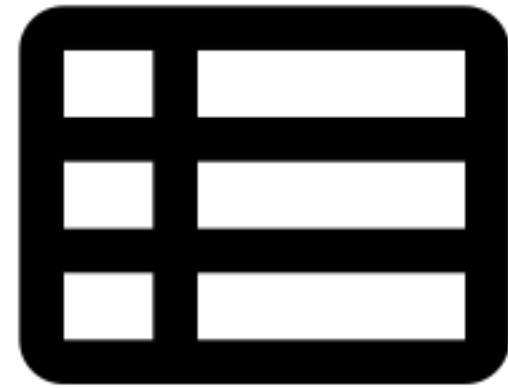
T2



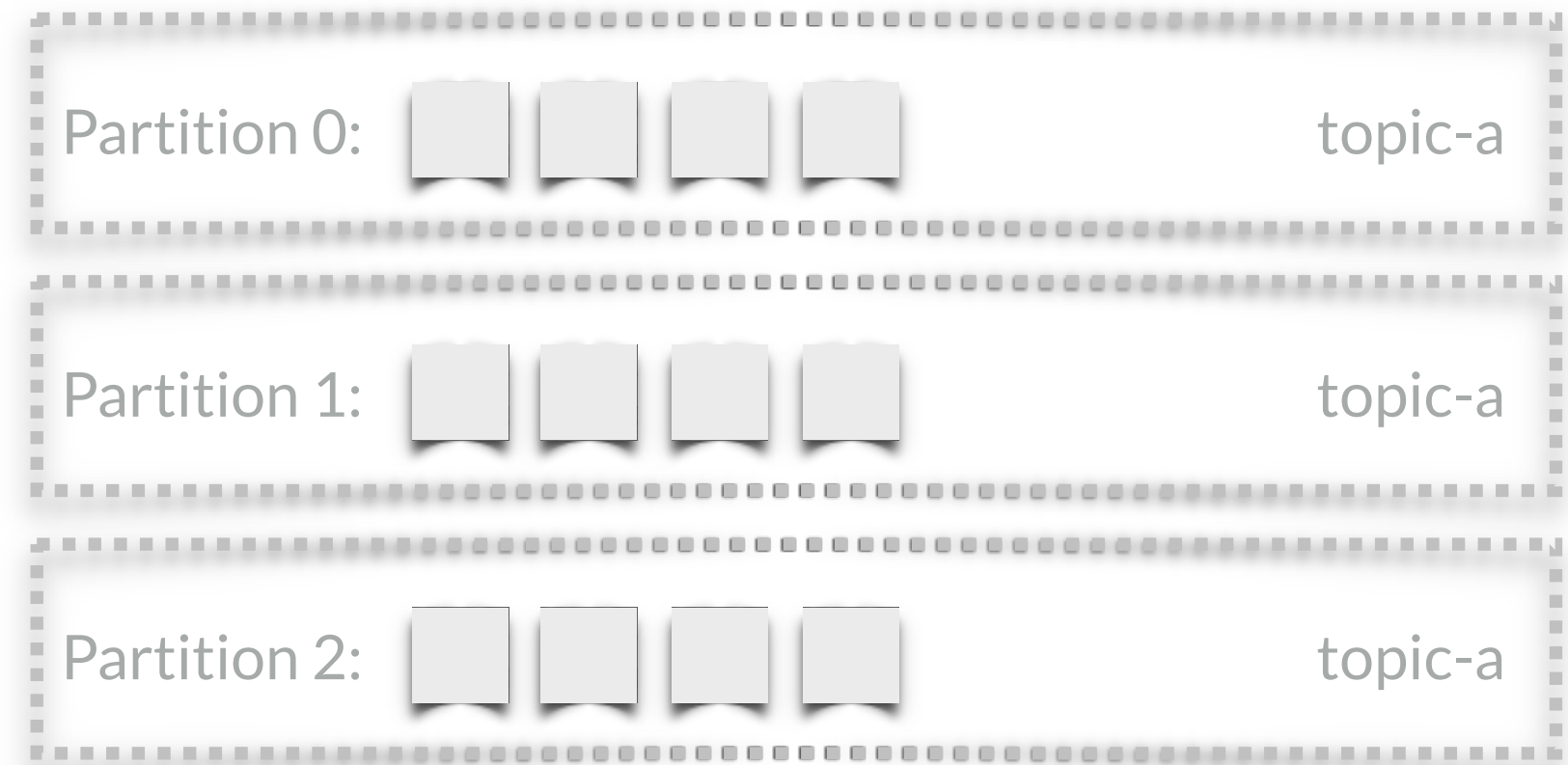
T6



T9



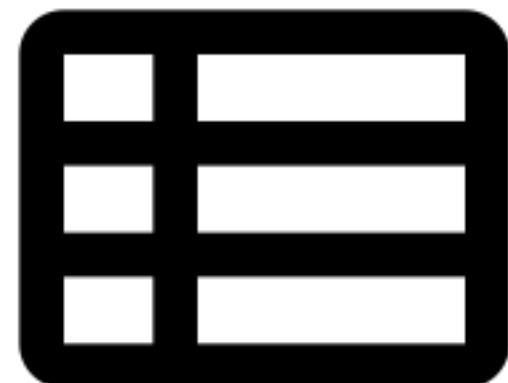
Partitions



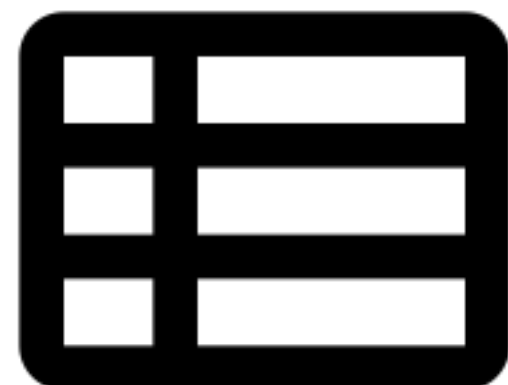
Tasks



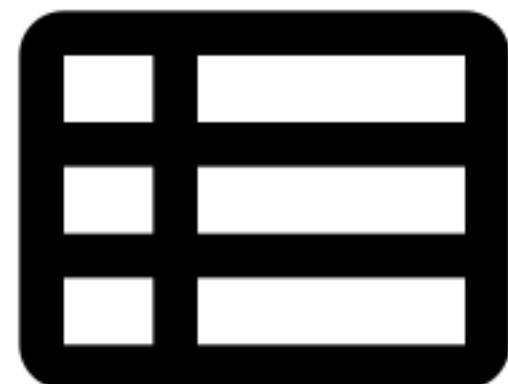
T2



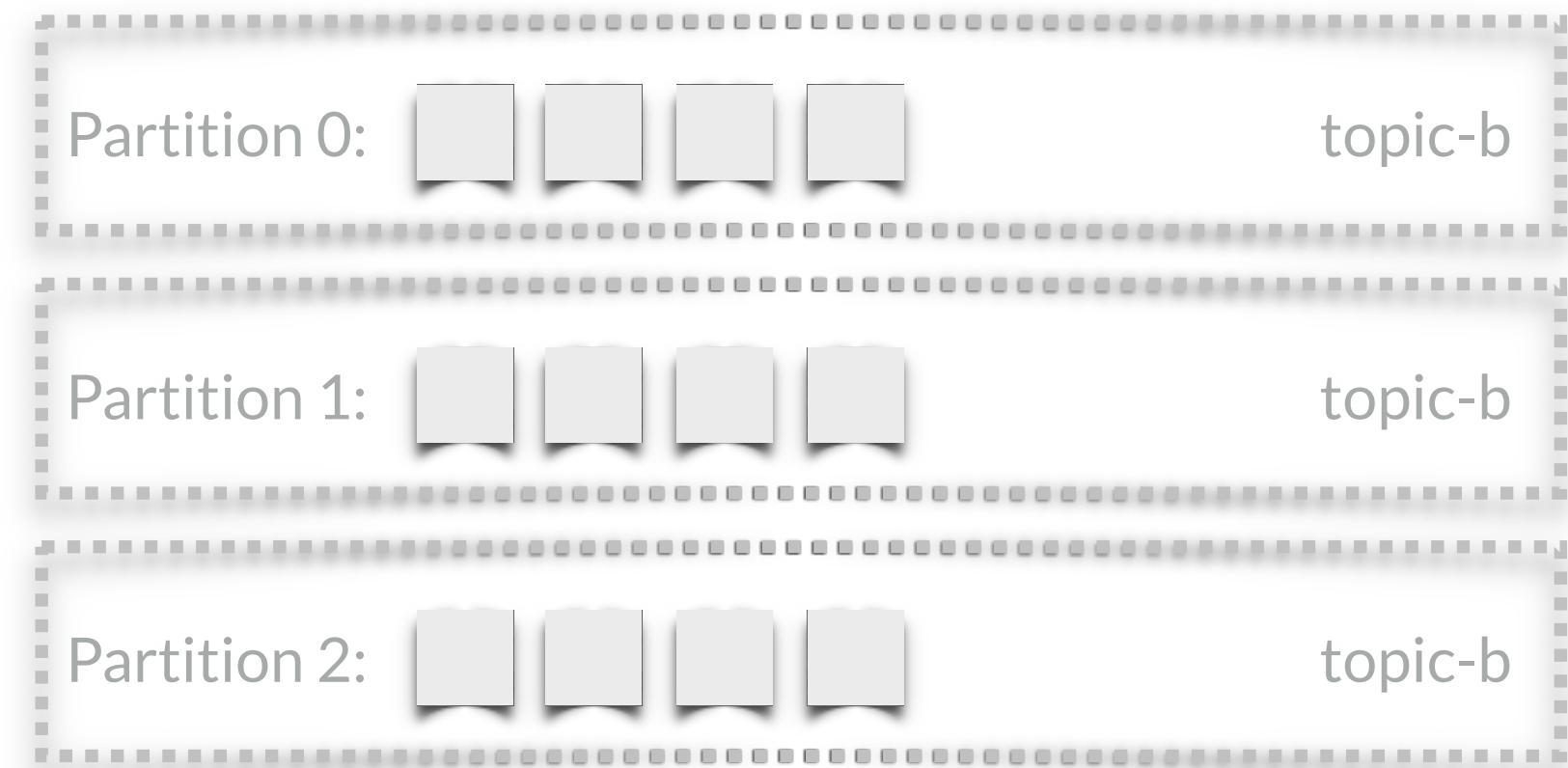
T6



T9



Partitions



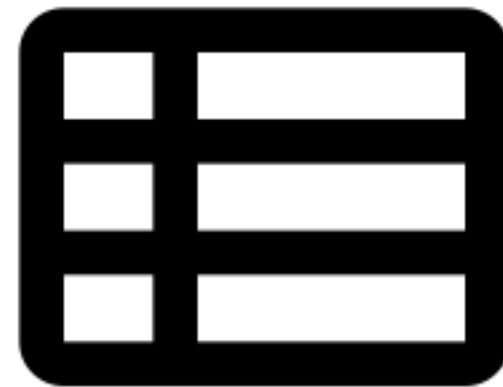
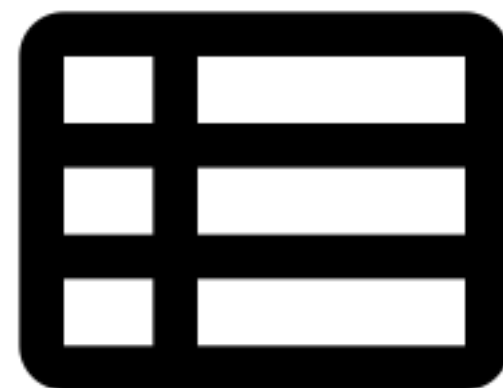
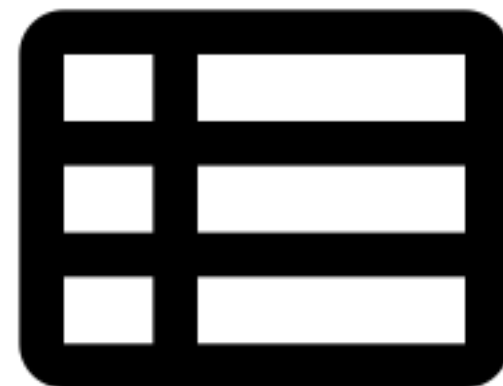
Tasks



T2

T6

T9



Partitions



Partition 0:



topic-c

Partition 1:



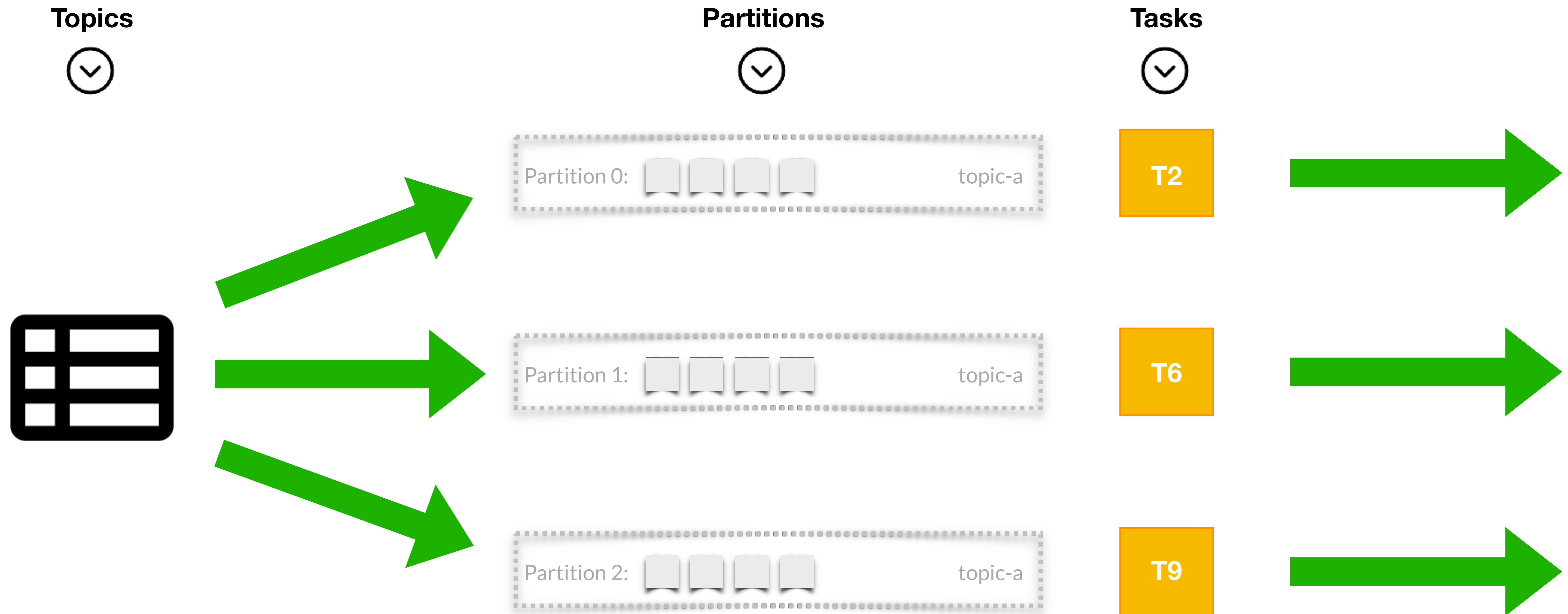
topic-c

Partition 2:



topic-c

Sink Architecture





T2

T8

T1



T6

T1

T2

T3



T3

T4

T7

T4



T9

T5

T6

T5

Offsets

Offsets

- Connect tracks the location of a file or database and marks its positions after each read
- This allows Connect to start at the right place
- Different Connectors do different things in regards to offsetting

Which Offsets?

- **File Source:** Position in that file
- **Database Source:** id or timestamp

Where do we track the offsets

- **JDBC:** A preconfigured Kafka Topic
- **HDFS Sink:** HDFS File
- **FileStream Source:** A Separate Local File

Connect Modes

Standalone Mode

Logical Jobs



Tasks



One Worker



One Machine

Common Configuration for All Connectors



```
% connect-standalone connect-standalone.properties \  
connector1.properties [connector2.properties connector3.properties ...]
```

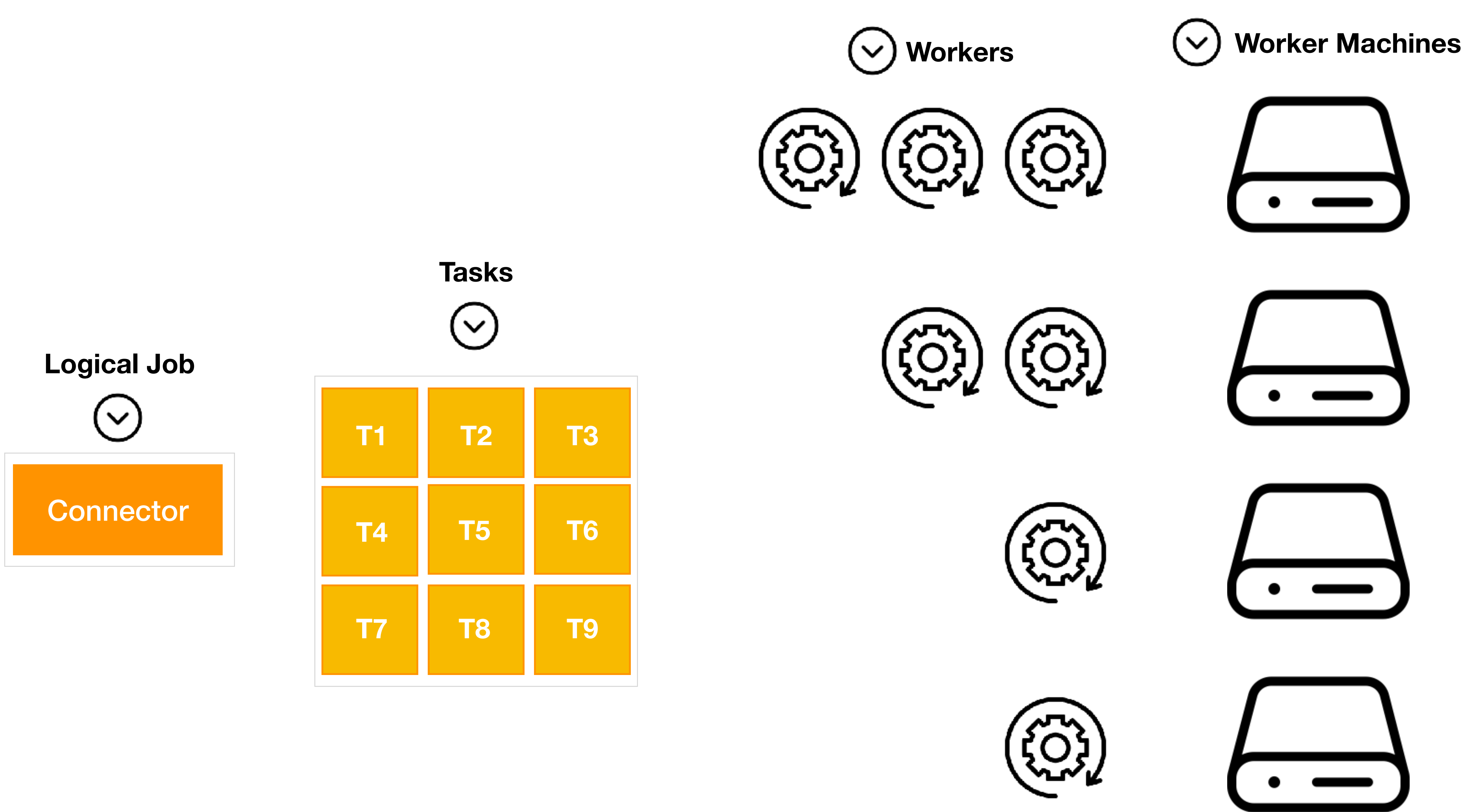


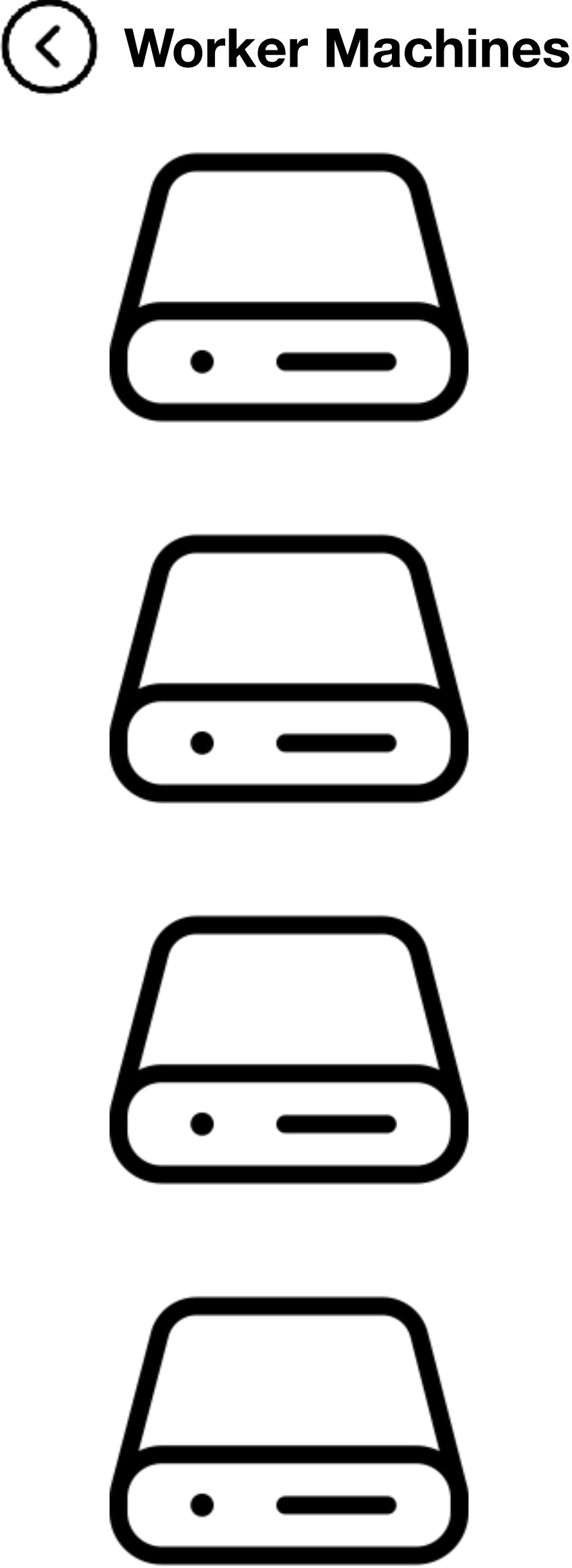
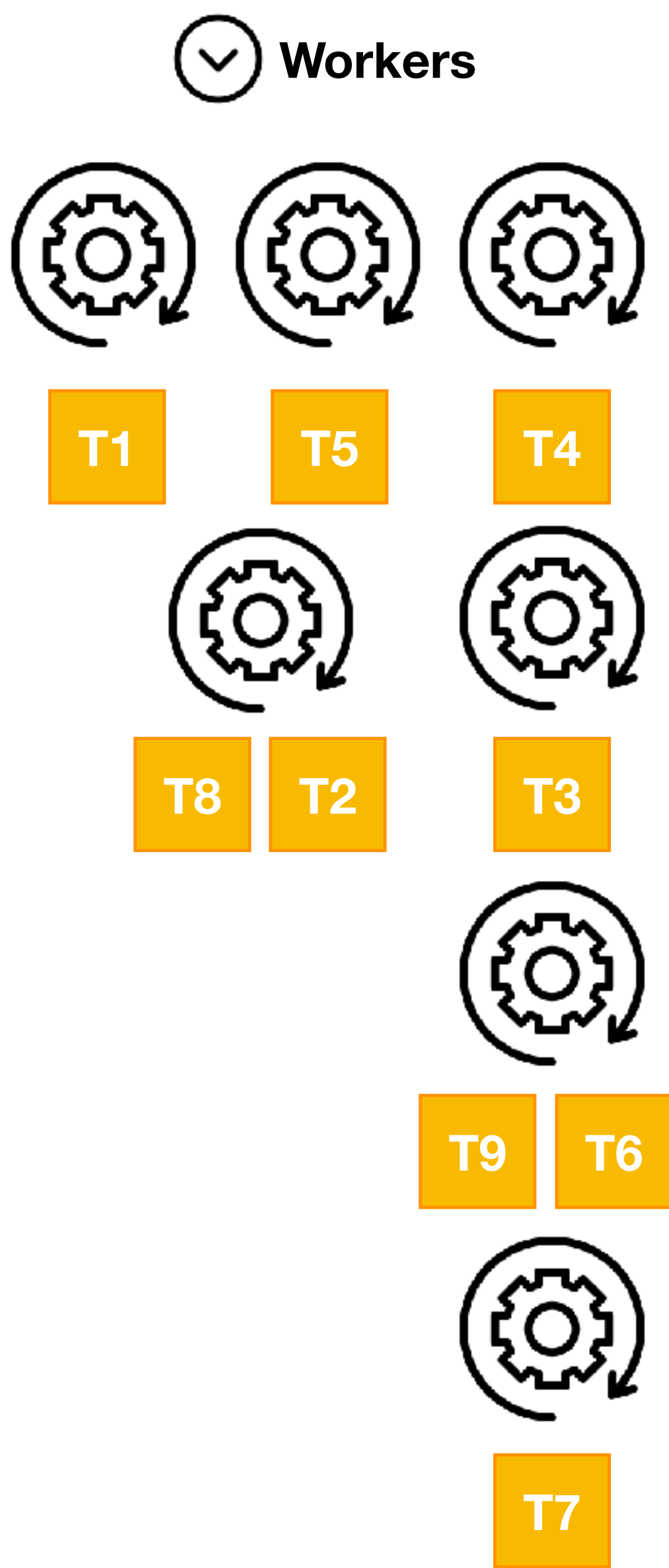
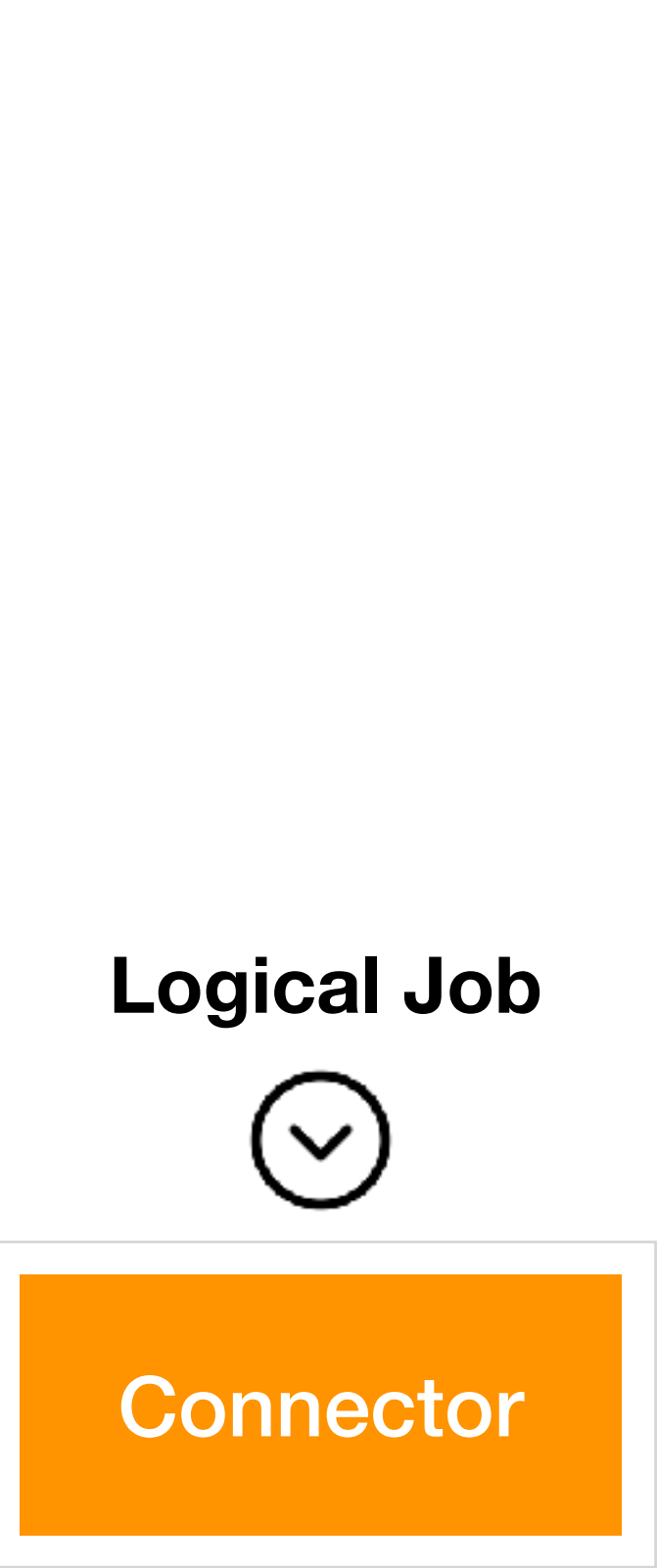
Configuration for Specific Connector



One Machine

Distributed Mode





Common Configuration for All Workers



```
% connect-distributed connect-distributed.properties
```

group.id=sales



group.id=sales



group.id=sales



group.id=sales



⤴ Worker Machines

**What is in a Common
Configuration?**

Common Configuration

<code>bootstrap.servers</code>	List of host/ports
<code>key.converter</code>	Converter Class for Key
<code>value.converter</code>	Converter Class for Value

Additional Configuration for Standalone

<code>offset.storage.file.filename</code>	The filename in which to store offset data for the Connectors (Default: ""). This enables a standalone process to be stopped and then resume where it left off.
---	---

Additional Configuration for Distributed

<code>group.id</code>	Group that the cluster belongs to
<code>session.timeout.ms</code>	Timeout used to detect failures when using Kafka's group management facilities fails
<code>heartbeat.interval.ms</code>	Expected time between heartbeats to the group coordinator when using Kafka's group management facilities. Must be smaller than <code>session.timeout.ms</code>
<code>config.storage.topic</code>	Topic in which to store Connector & task config. data
<code>offset.storage.topic</code>	Topic in which to store offset data for Connectors
<code>status.storage.topic</code>	Topic in which to store connector & task status

**What is in a Specific
Configuration?**

Specific Connector Configuration

name	Unique Name of Connector
connector.class	Connector class
tasks.max	Number of tasks to run; Connect decides if possible
key.converter	Converter for the key
value.converter	Value for the key
topics	Sink Topics

Single Message Transforms

Transformation Chains

- **transforms** - List of aliases for the transformation, specifying the order in which the transformations will be applied.
- **transforms.\$alias.type** - Fully qualified class name for the transformation.
- **transforms.\$alias.\$transformationSpecificConfig** - Configuration properties for the transformation

Alias Declarations

```
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
```

Alias Declarations →

Hoist Field Transformation

```
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
```

Make Map Configuration



org.apache.kafka.connect.transforms.HoistField

Wrap data using the specified field name in a Struct when schema present, or a Map in the case of schemaless data.

Use the concrete transformation type designed for the record key (`org.apache.kafka.connect.transforms.HoistField$Key`) or value (`org.apache.kafka.connect.transforms.HoistField$Value`).

NAME	DESCRIPTION	TYPE	DEFAULT	VALID VALUES	IMPORTANCE
field	Field name for the single field that will be created in the resulting Struct or Map.	string			medium

"Hello World"
"Kool And The Gang"
"Roger, Roger"
"Everyone Wang Chung Tonight"
"Get off my plane!"

```
{"line":"Hello World"}  
{"line":"Kool And The Gang"}  
{"line":"Roger, Roger"}  
{"line":"Everyone Wang Chung Tonight"}  
{"line":"Get off my plane!"}
```

Insert Source Transformation

```
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
```

← Insert Source Configuration

org.apache.kafka.connect.transforms.InsertField

Insert field(s) using attributes from the record metadata or a configured static value.

Use the concrete transformation type designed for the record key (`org.apache.kafka.connect.transforms.InsertField$Key`) or value (`org.apache.kafka.connect.transforms.InsertField$Value`).

NAME	DESCRIPTION	TYPE	DEFAULT	VALID VALUES	IMPORTANCE
static.field	Field name for static data field. Suffix with <code>!</code> to make this a required field, or <code>?</code> to keep it optional (the default).	string	null		medium
static.value	Static field value, if field name configured.	string	null		medium

```
{"line":"Hello World"}  
{"line":"Kool And The Gang"}  
{"line":"Roger, Roger"}  
{"line":"Everyone Wang Chung Tonight"}  
{"line":"Get off my plane!"}
```



```
{"line":"Hello World", "data-source":"test-file-source"}  
{"line":"Kool And The Gang", "data-source":"test-file-source"}  
{"line":"Roger, Roger", "data-source":"test-file-source:"}  
{"line":"Everyone Wang Chung Tonight", "data-source":"test-file-source:"}  
{"line":"Get off my plane!", "data-source":"test-file-source:"}
```

Available 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**
- **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
- **RegexRouter** - modify the topic of a record based on original topic, replacement string and a regular expression

Converters

Source

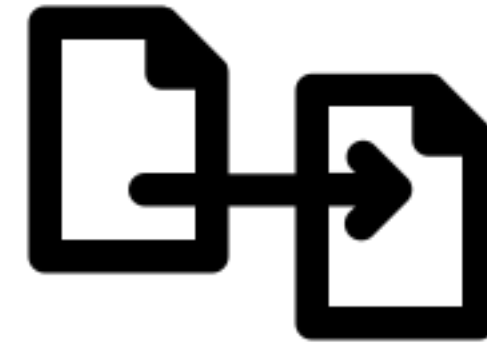


Custom Format



Connector

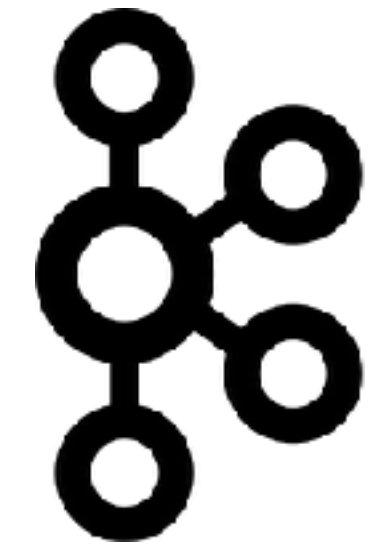
Converter



byte[]



Kafka



Converters Exist Independently

Schema Registry



Source

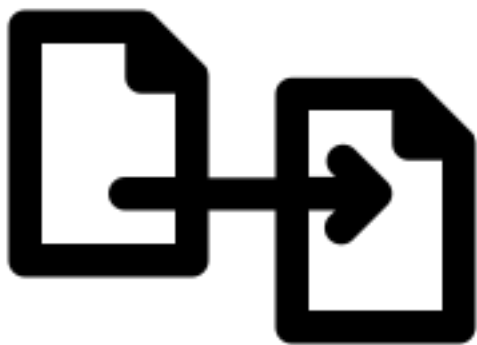


Custom Format



Connector

Converter



byte[]



Kafka



```
key.converter = org.apache.kafka.connect.storage.StringConverter
```

```
value.converter = org.apache.kafka.connect.storage.StringConverter
```

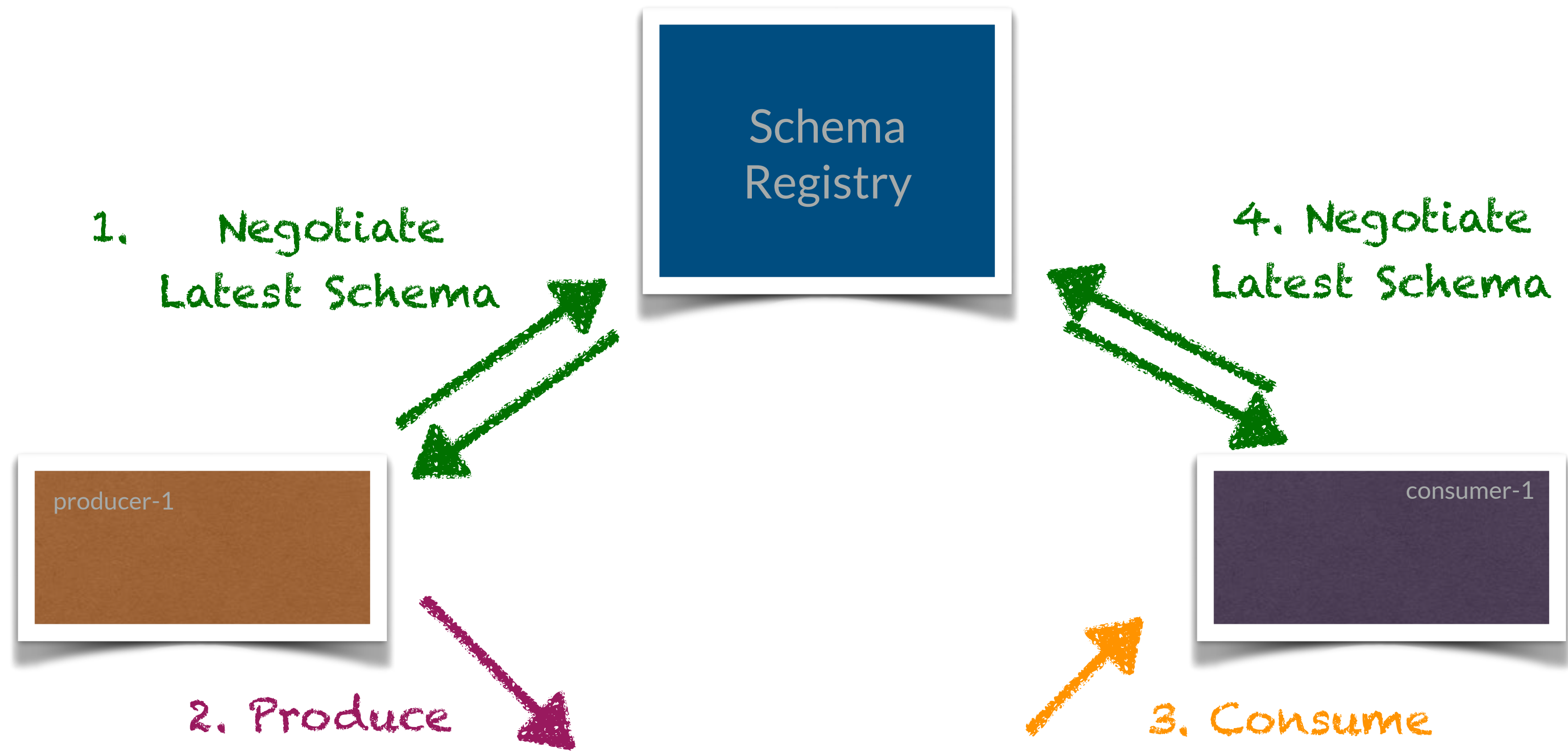
```
key.converter = org.apache.kafka.connect.json.JsonConverter
```

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

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

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

Uses Schema Registry



Configuring Connect to use Avro with Schema Registry

```
key.converter=io.confluent.connect.avro.AvroConverter  
key.converter.schema.registry.url=http://schemaregistry1:8081  
value.converter=io.confluent.connect.avro.AvroConverter  
value.converter.schema.registry.url=http://schemaregistry1:8081
```

JDBC Connector

JDBC Source Connector

- Polls a JDBC Connection to Database for any new and updated changes
- Creates a record in Kafka based on the changes
- Table to Topic Relationship
- New and Deleted Tables are propagated automatically

JDBC Connector Properties

<code>connection.url</code>	JDBC connection URL for the database to load.
<code>topic.prefix</code>	Prefix to prepend to table names to generate the name of the Kafka topic
<code>mode</code>	The mode for updating a table each time it is polled (See following slides).
<code>poll.interval.ms</code>	Frequency in ms to poll for new data in each table.
<code>timestamp.delay.interval.ms</code>	How long to wait after a row with certain timestamp appears before we include it in the result

Ensure that your database driver is in the classpath of your connector

JDBC Connector Properties

<code>incrementing.column.name</code>	The name of the strictly incrementing column to use to detect new rows
<code>query</code>	If specified, the query to perform to select new or updated row. Great for joins
<code>table.blacklist</code>	List of tables to exclude from copying
<code>table.whitelist</code>	List of tables to include in copying
<code>timestamp.column.name</code>	The name of the timestamp column to use to detect new or modified rows

JDBC Connector Properties

<code>batch.max.rows</code>	Maximum number of rows to include in a single batch when polling for new data
<code>table.poll.interval.ms</code>	Frequency in ms to poll for new or removed tables
<code>validate.non.null</code>	By default, the JDBC connector will validate that all incrementing and timestamp tables have NOT NULL

JDBC Connector Modes

JDBC Connector Modes

bulk	Perform a bulk load of the entire table each time it is polled
incrementing	Use a strictly incrementing column on each table to detect only new rows. Note that this will not detect modifications or deletions of existing rows
timestamp	Use a timestamp (or timestamp-like) column to detect new and modified rows.
timestamp+incrementing	Use two columns, a timestamp column that detects new and modified rows and a strictly incrementing column which provides a globally unique ID

RESTFul Configuration

REST API Configuration

- Connectors can be deployed via RestFul Configuration rather than file deployment
- For Distributed Mode
 - This is the only way to do so in distributed mode
 - Changes will persist even after restarts
 - REST Calls can be made to any worker
- For Standalone
 - REST Calls can be made to the single
 - Changes **will not persist**

REST Examples

GET /connectors	Get a list of active connectors
POST /connectors	Create a new connector, returning the current connector info if successful.
GET /connectors/(string:name)	Get information about the connector.
GET /connectors/(string:name)/config	Get the configuration for the connector
GET /connectors/(string:name)/status	Get current status of the connector, including whether it is running, failed or paused

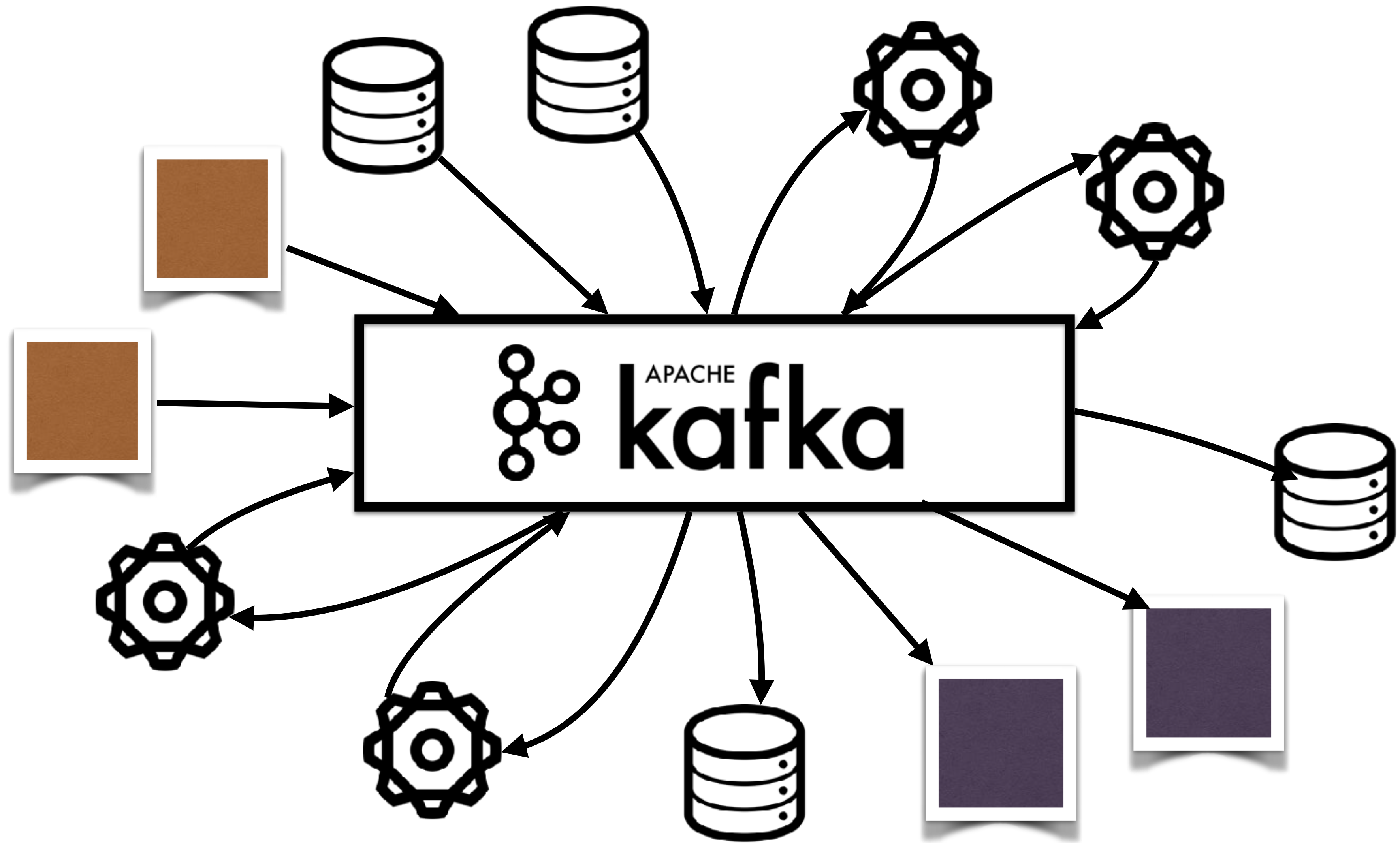
Creating an HDFS Sink Example

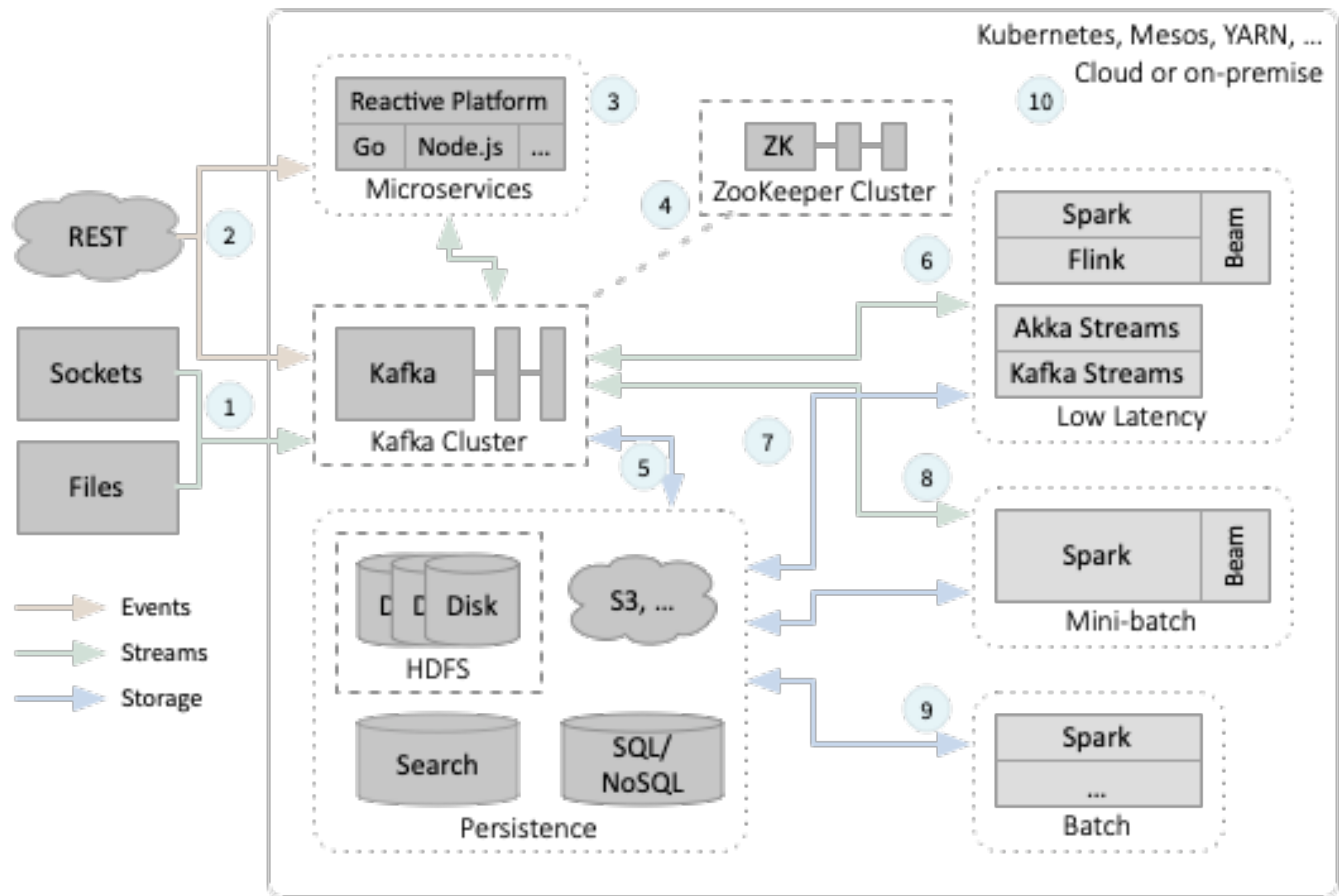
```
POST /connectors HTTP/1.1
Host: connect.example.com
Content-Type: application/json
Accept: application/json

{
  "name": "hdfs-sink-connector",
  "config": {
    "connector.class": "io.confluent.connect.hdfs.HdfsSinkConnector",
    "tasks.max": "10",
    "topics": "test-topic",
    "hdfs.url": "hdfs://fakehost:9000",
    "hadoop.conf.dir": "/opt/hadoop/conf",
    "hadoop.home": "/opt/hadoop",
    "flush.size": "100",
    "rotate.interval.ms": "1000"
  }
}
```

Demo

Conclusion





Thank You

Daniel Hinojosa
Programmer, Consultant,
Trainer

dhinojosa@evolutionnext.com
@dhinojosa

