EDEM
Escuela de Empresarios

Captura de Datos

Basado en Eventos

RUBÉN SANCHIS

rusaso@edem.es

Curso 1 - Edición 2ª

Fecha 12/11/2020

Índice de contenidos

EDEM
Escuela de Empresarios

- Patrón Public/Subscribe
- Apache Kafka
  - Características y Arquitectura
  - Principales conceptos

EDEM
Escuela de Empresarios

1.Patrón Publish/Subscribe

Patrón Publish/Subscribe

EDEM
Escuela de Empresarios

- Es un patrón de arquitectura que utiliza una cola de mensajería para la comunicación entre aplicaciones, servicios o componentes de manera desacoplada y asíncrona
- Se enfoca más en el intercambio de información final procesada, aunque también puede cumplir aspectos de preparación o modificación
- Tiene varias denominaciones:
  - Publish/Subscribe - > Publicador/Suscriptor
  - **Producer/Consumer** -> Productor/Consumidor

Patrón Publish/Subscribe

EDEM
Escuela de Empresarios

¿Cómo funciona?

- Un servicio tipo **Publisher** (emisor) genera un dato o mensaje pero sin dirigirlo o referenciar a ningún **Subscriber** (receptor) en concreto
  - Es decir, no lo envía de forma directa a la "dirección" de ningún subscriber
- Para ello dispone de listas de topics (temas) publicados específicos, el productor clasifica el mensaje en base a una tipología, lo pone en la lista de un tema específico y el receptor se suscribe a la listas para recibir ese tipo de mensajes
  - Lo que conecta a un Publisher con los Subscriber es el nombre del Topic

Patrón Publish/Subscribe
Captura de datos: Requerimientos a considerar

EDEM
Escuela de Empresarios

Topología: Hace referencia a las distintas estrategias (con/sin duplicados, con/sin orden y con/sin perdida de datos) para establecer garantías de entrega y de ordenación

- **Estrategia de entrega**
  - **AMO: At Most Once** (como máximo una vez)
    - El mensaje se envía sólo una vez y se aunque se garantiza que no hay duplicados pueden perderse datos
  - **ALO: At Least Once** (al menos una vez)
    - Idem que AMO pero se garantiza que no hay perdida de datos aunque pueden haber duplicados
  - **EO: Exactly Once** (exactamente una vez)
    - Se garantiza que no hay duplicados en la escritura, que no hay pérdida de datos y que sólo se va a consumidor una vez

Patrón Publish/Subscribe

Captura de datos: Requerimientos a considerar

EDEM
Escuela de Empresarios

- **Estrategia de ordenación**
  - **No ordering** (sin orden)
    - No importa el orden de recepción y por lo tanto se puede incrementar el rendimiento
  - **Partition ordering** (ordenación por partición)
    - Asegurar un orden por partición y por lo tanto tiene un coste extra en la escritura
  - **Global Order (ordenación global)**
    - Se requiere un orden en los datos por lo tanto necesita un mayor coste extra con implicaciones en el rendimiento

Patrón Publish/Subscribe

Captura de datos: Requerimientos a considerar

EDEM
Escuela de Empresarios

- **Disponibilidad (Availability)**
  - Capacidad para estar el mayor tiempo posible trabajando o activo. En este punto hay que tener en cuenta los mantenimientos y la aparición de errores (detección + reparación).
- **Transaccionalidad (Transaction)**
  - Capacidad para agrupar acciones o elementos en unidades atómicas
    - O se ejecutan todas como una única operación o bien no se ejecuta ninguna
- **Escalabilidad (Scalability)**
  - Capacidad de un elemento para evolucionar en un aspecto con el objetivo de poder dar soporte a una mayor cantidad de acciones o elementos
    - Pueden existir diferentes dimensiones: mensajes, topics, productores o consumidores

Patrón Publish/Subscribe

Captura de datos: Requerimientos a considerar

EDEM
Escuela de Empresarios

- **Rendimiento (Throughput)**
  - Mide la capacidad en términos de la eficiencia en lo referente a la cantidad (nº de bytes) de elementos por unidad de tiempo con los que puede procesar
    - En algunos casos también se denomina ancho de banda (**bandwidth**)
- **Latencia (Latency)**
  - Mide la eficiencia basada en el pipeline requerido para su procesamiento, es decir, se suele considerar el tiempo requerido para procesar un elemento
    - En algunos casos también se denomina **tiempo de respuesta**
- **Desacoplamiento (Decoupling)**
  - Capacidad que hace referencia a diferentes aspectos:
    - **Entidad**: Productores y consumidores no necesitan conocerse entre si
    - **Tiempo**: Productores y consumidores no necesitan participar activamente
    - **Sincronización**: Si los hilos de ejecución o los clientes requieren algún tipo de bloqueo síncrono

EDEM
Escuela de Empresarios

Características y Arquitectura

EDEM
Escuela de Empresarios

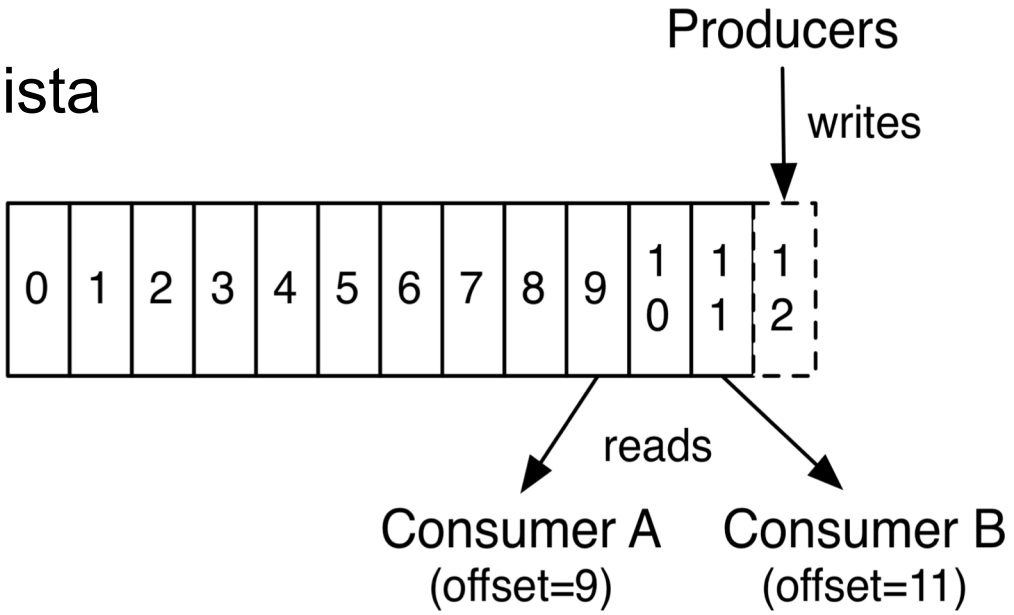
- Apache Kafka es un sistema de mensajes distribuido **publish/subscribe** open source basado en una arquitectura Peer to Peer (P2P)
- Kafka de autodefine como un **commit log *distribuido***

  ➢ ***No confundir con los logs de los mensajes de error de las aplicaciones***

Características de un "commit log"

EDEM
Escuela de Empresarios

- Es una **estructura de datos ordenada y persistente**
  - El orden permite un procesamiento determinista
- El commit log constituye el núcleo principal de Kafka
  - Sólo se permite añadir mensajes (append)
  - No se pueden modificar ni borrar registros
    - No se existen comandos tipo UPDATE o DELETE



Características de un "commit log"

EDEM
Escuela de Empresarios

Distribuido

- El commit log se distribuye en varios nodos, los cuales trabajan de forma conjunta al trabajar dentro de un clúster, aunque para el usuario final se ve como un único elemento
- Cada nodo del clúster se llama **Broker**
- El ser distribuido proporciona
  - Escalado horizontal
  - Tolerancia a fallos

Características de un "commit log"

EDEM
Escuela de Empresarios

Escalado Horizontal

- Se puede escalar a muchos nodos en relativo poco tiempo
  - Facilita agregar un nuevo nodo sin tiempos de inactividad
  - No suele tener "límites" en la cantidad de nodos
  - Uso de sharding
  - Se puede fragmentar las listas / temas / topics en particiones y repartirlas por los nodos

Características de un "commit log"

EDEM
Escuela de Empresarios

Durabilidad / Persistencia

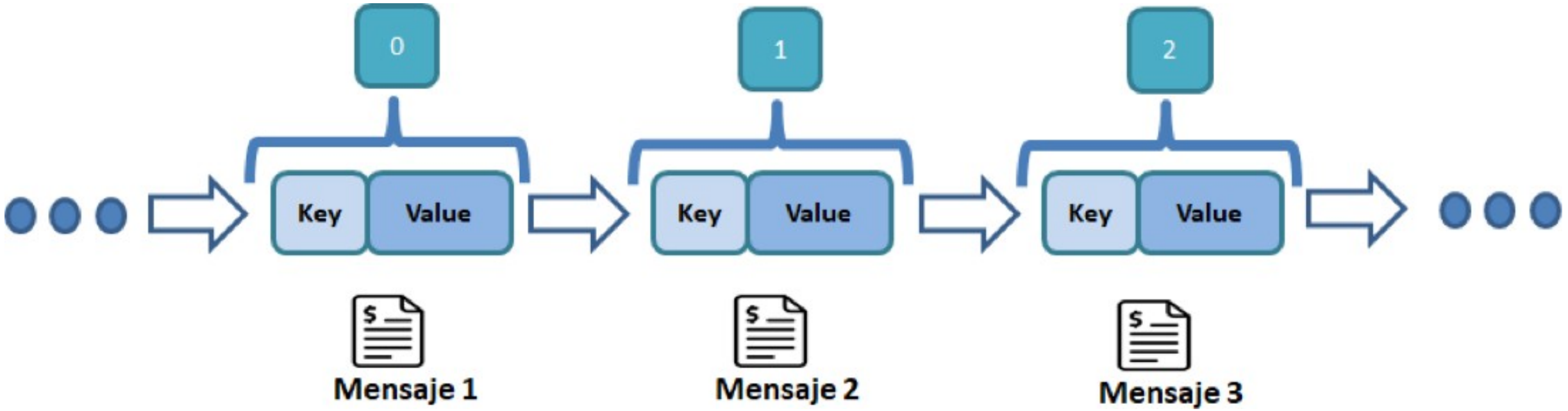
- Los mensajes son persistidos (guardados) en el sistema de ficheros del sistema operativo y son replicados entre los nodos del clusters

Tolerancia a fallos

- No existe un único punto de fallo (SPoF) al replicar las particiones en múltiples brokers
  - Cuanto mayor es la replicación menor es el rendimiento de escritura (acks)

“Stream” de Mensajes

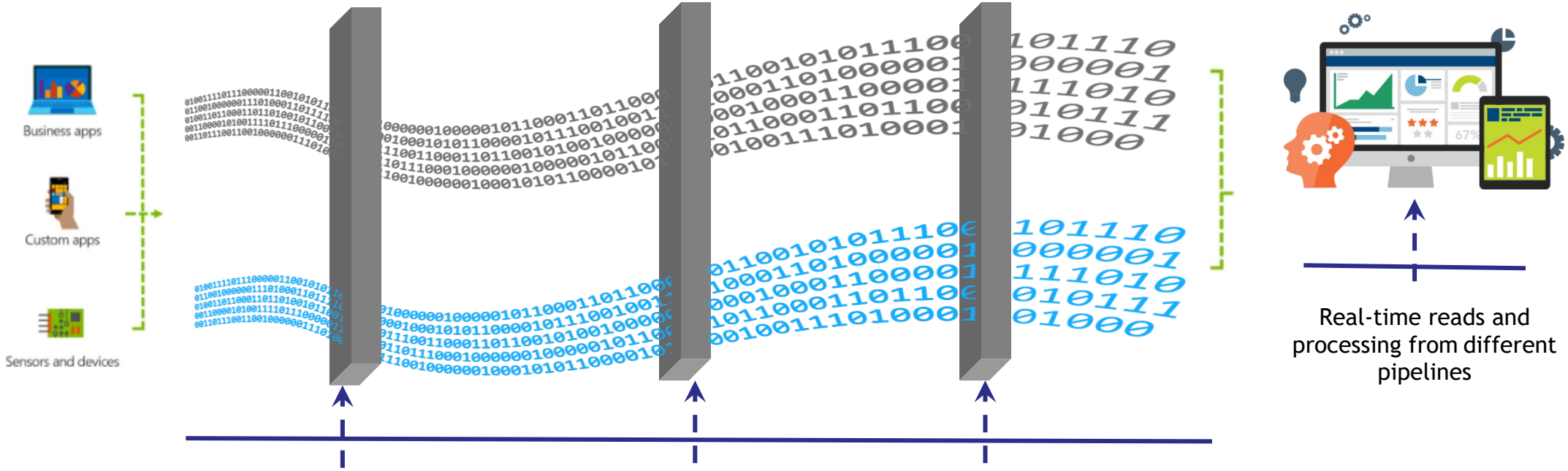
EDEM
Escuela de Empresarios



¿Qué es un Data Stream?

EDEM
Escuela de Empresarios

Building real-time **streaming applications** that transform or react to the streams of data

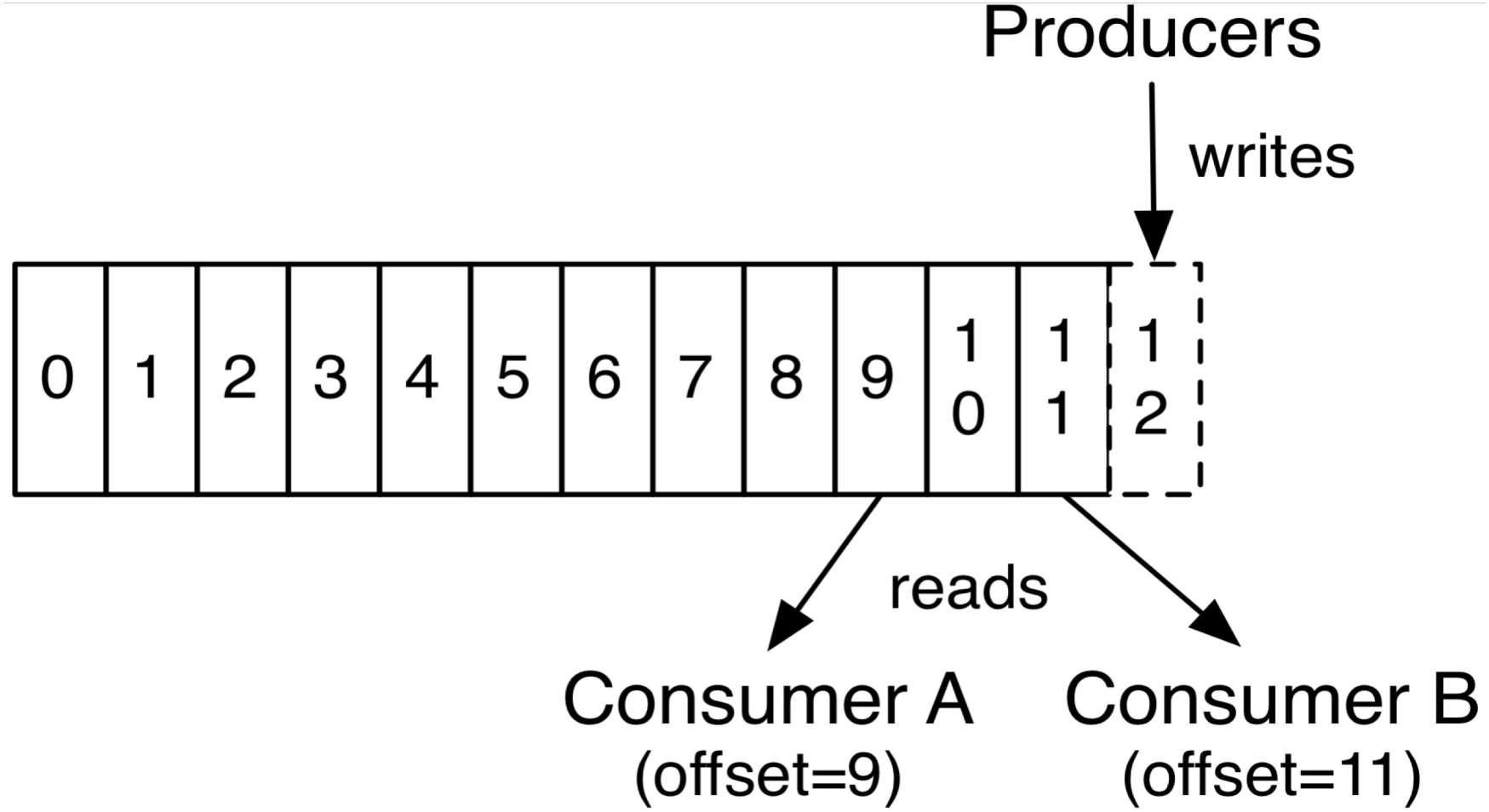


Operations: Enrichment, routing, tagging, etc

Características de un "commit log"

EDEM
Escuela de Empresarios

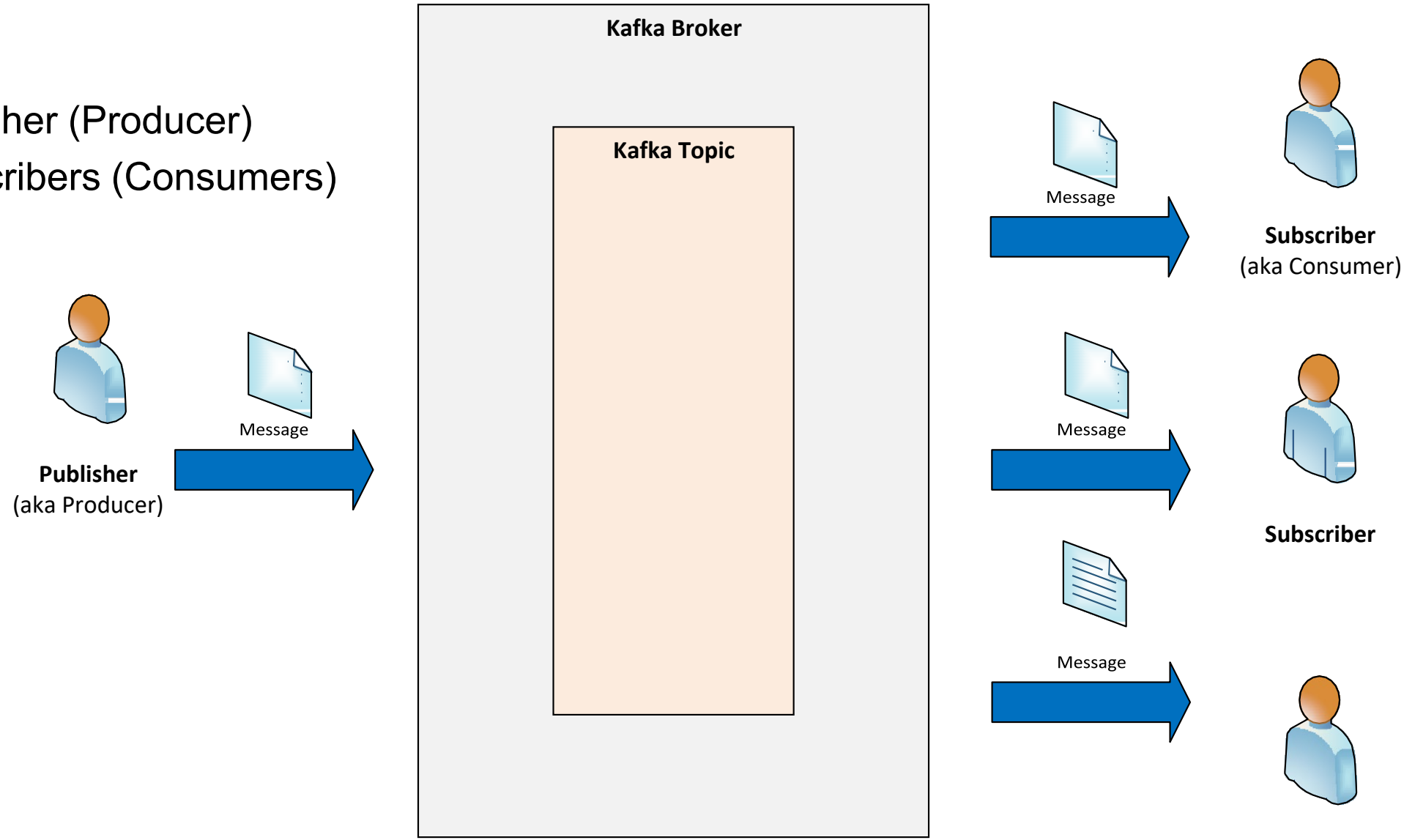
- Es una **estructura de datos ordenada y persistente**



Apache Kafka

EDEM
Escuela de Empresarios

- 1 Topic
- 1 Publisher (Producer)
- 2 Subscribers (Consumers)



Apache Kafka

Subscriber

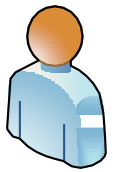
EDEM
Escuela de Empresarios

Apache Kafka

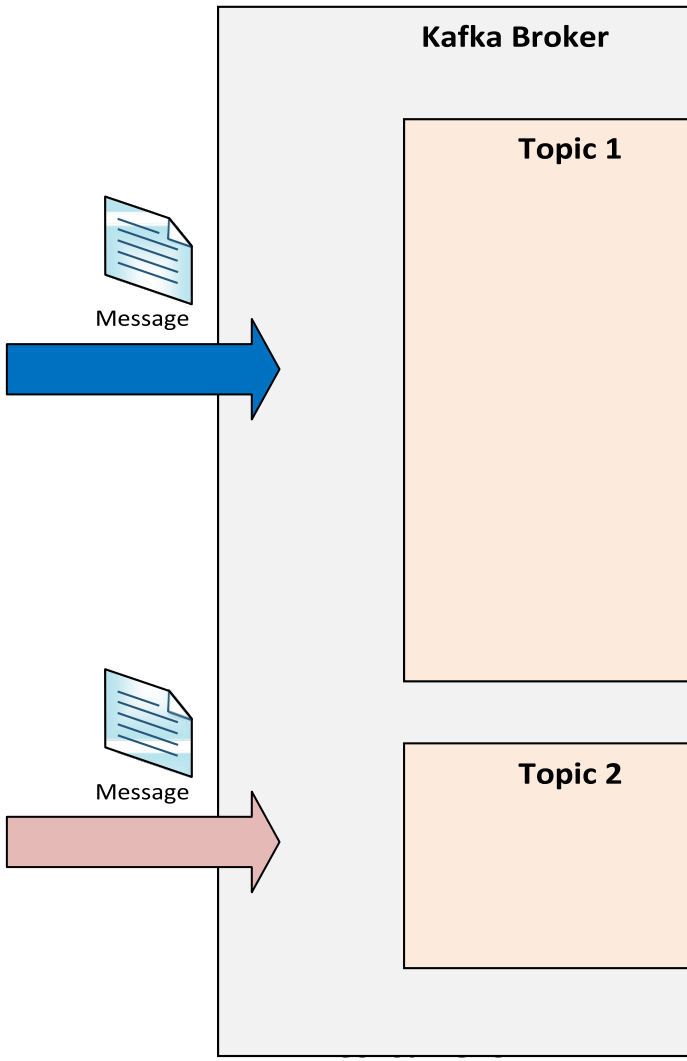
EDEM
Escuela de Empresarios

- 2 Topics
- 2 Publisher (Producer)
- 3 Subscribers (Consumers)

Producer 1

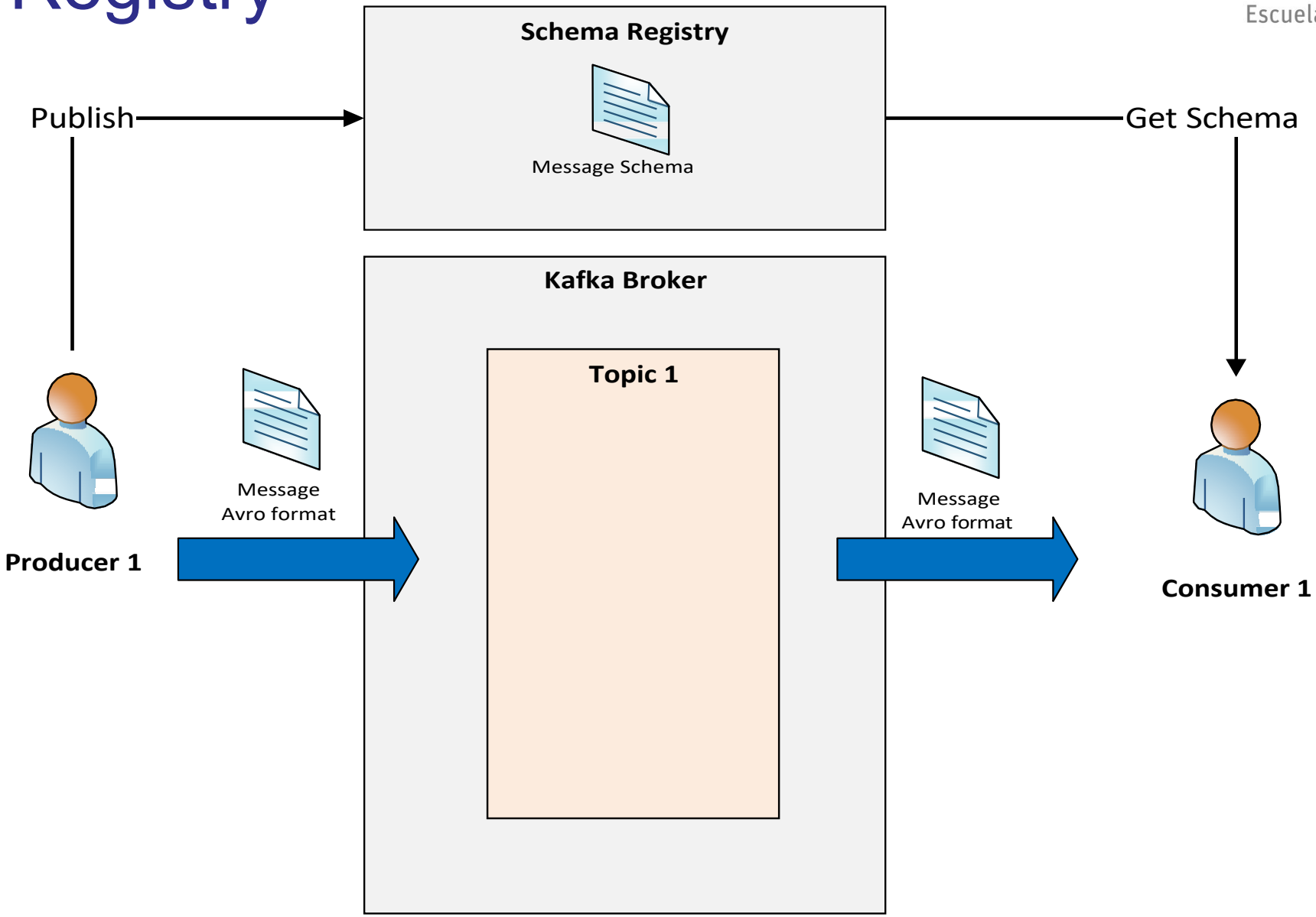


Producer 2



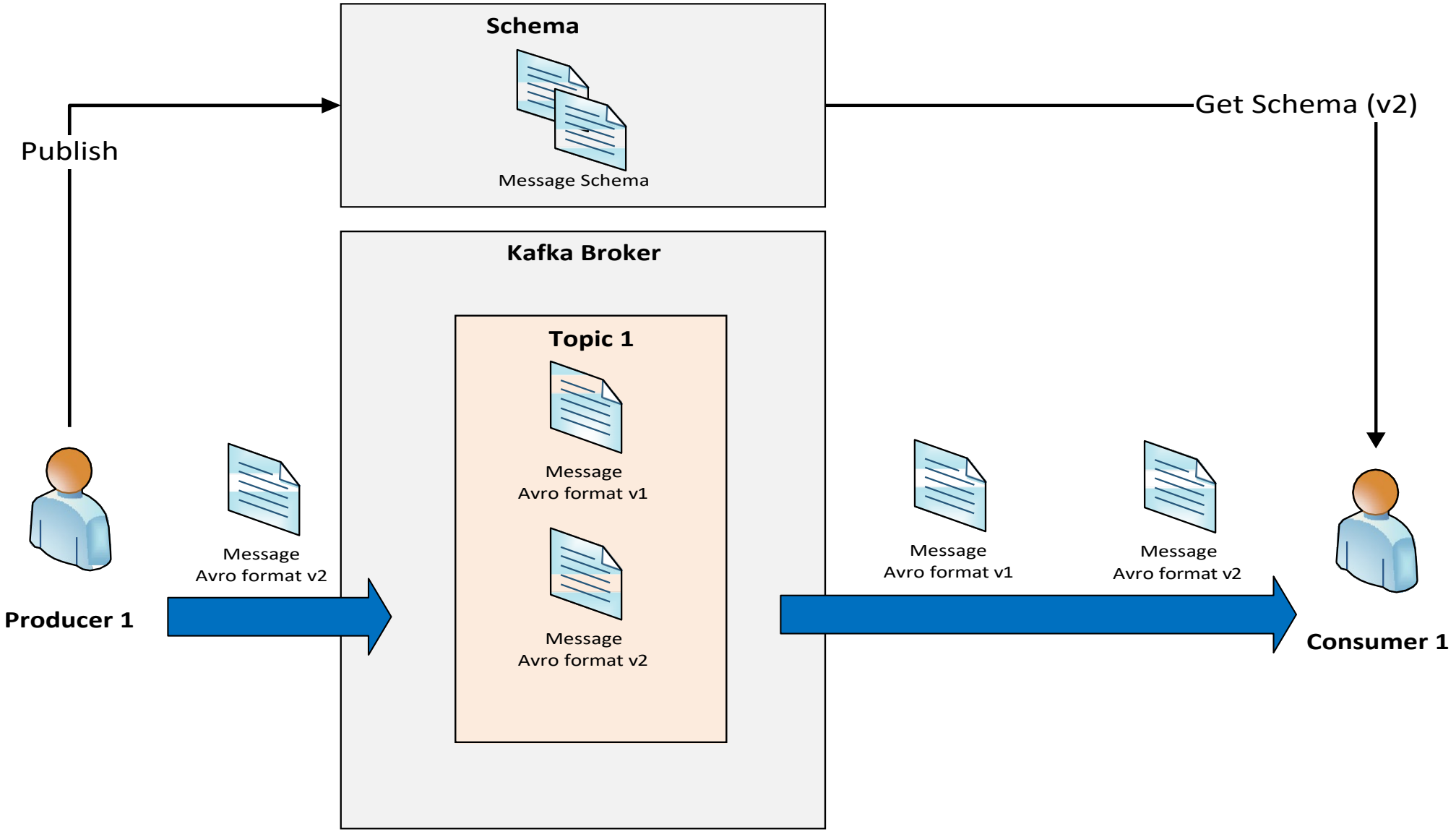
Schema Registry

EDEM
Escuela de Empresarios



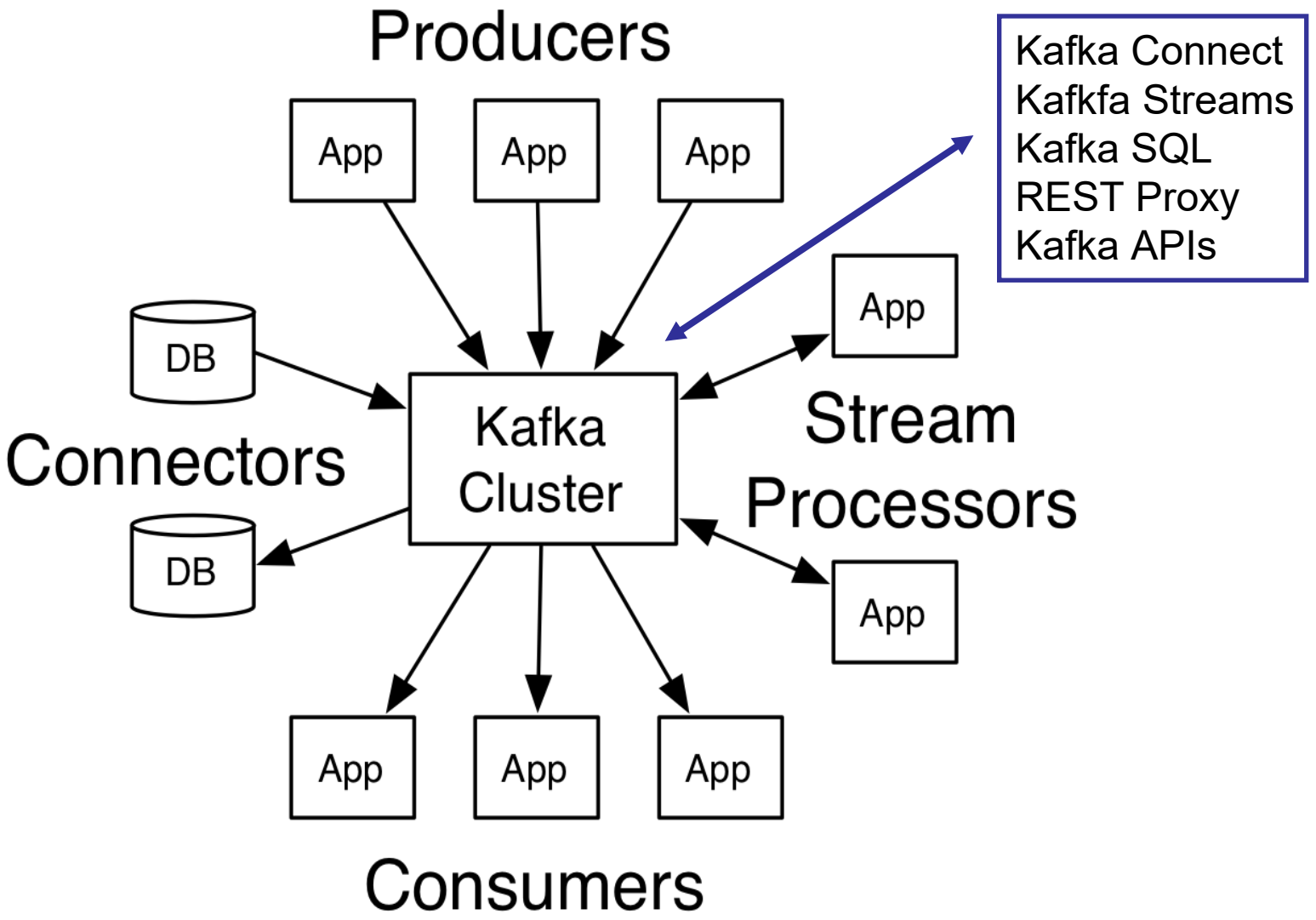
Schema Registry

EDEM
Escuela de Empresarios



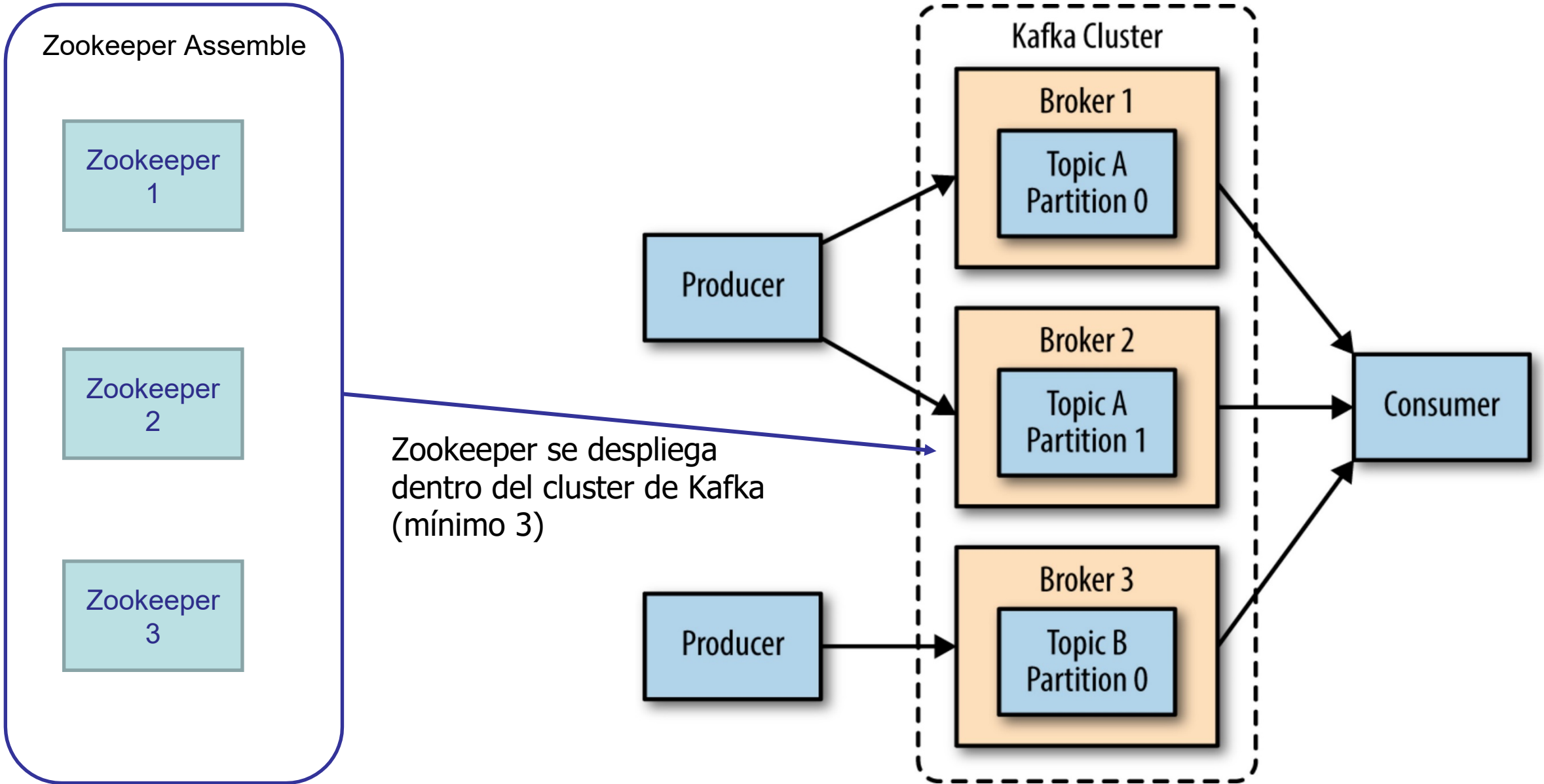
Ecosistema Kafka

EDEM
Escuela de Empresarios



Arquitectura de un cluster

EDEM
Escuela de Empresarios



Zookeeper

EDEM
Escuela de Empresarios

- Software que proporciona un servicio de coordinación de alto rendimiento para aplicaciones distribuidas
- En un clúster de Kafka
  - Coordina la topología de los brokers y su estado
  - Hace de almacén Clave/Valor distribuido con los aspectos de control de la plataforma
  - Proporciona una vista sincronizada de la configuración del clúster

Zookeeper

EDEM
Escuela de Empresarios

- Intercambia metadatos con: brokers, productores y consumidores
  - Direcciones IP de los brokers
  - Offsets de los mensajes consumidos
  - Detecta la carga de trabajo de cada componente y se encarga de realizar asignaciones de tareas bien por cercanía o bien por tener una menor ocupación
  - Realiza el **descubrimiento y control de los brokers** del clúster. Así, es el que se entera de:
    - Cuando se ha incorporado un nuevo bróker
    - Cuando falla un bróker
    - Cuando se crea un topic
    - Estado de salud de las particiones

Hands-on: Zookeeper & Kafka

Ejecuta Zookeeper y Kafaka con este ejercicio:

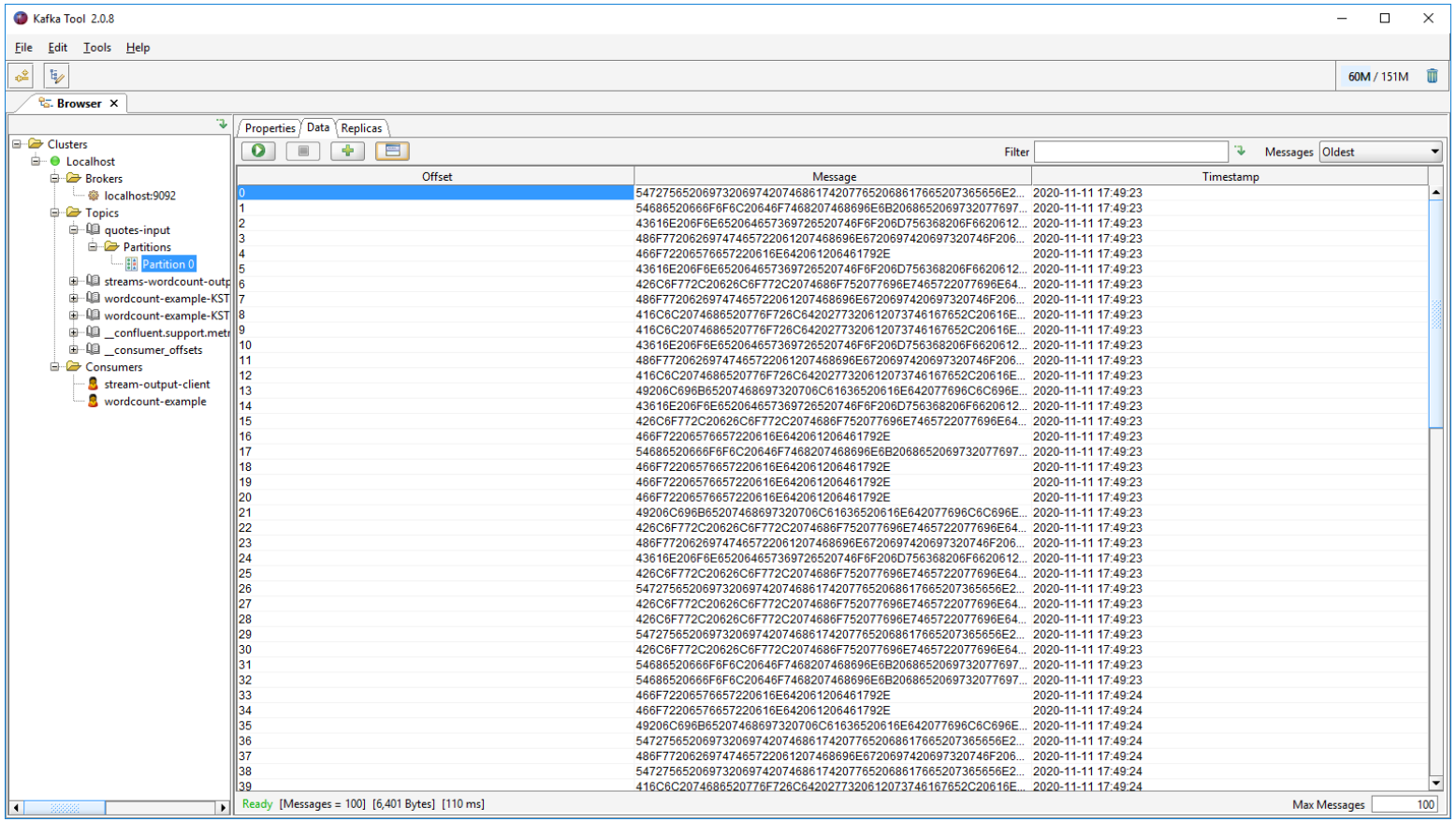
https://github.com/rusansor/edem2021/tree/main/Lab0_Inicio

Crea topics Kafka, produce mensajes y consúmelos:

https://github.com/rusansor/edem2021/tree/main/Lab1_Holla%20Mundo

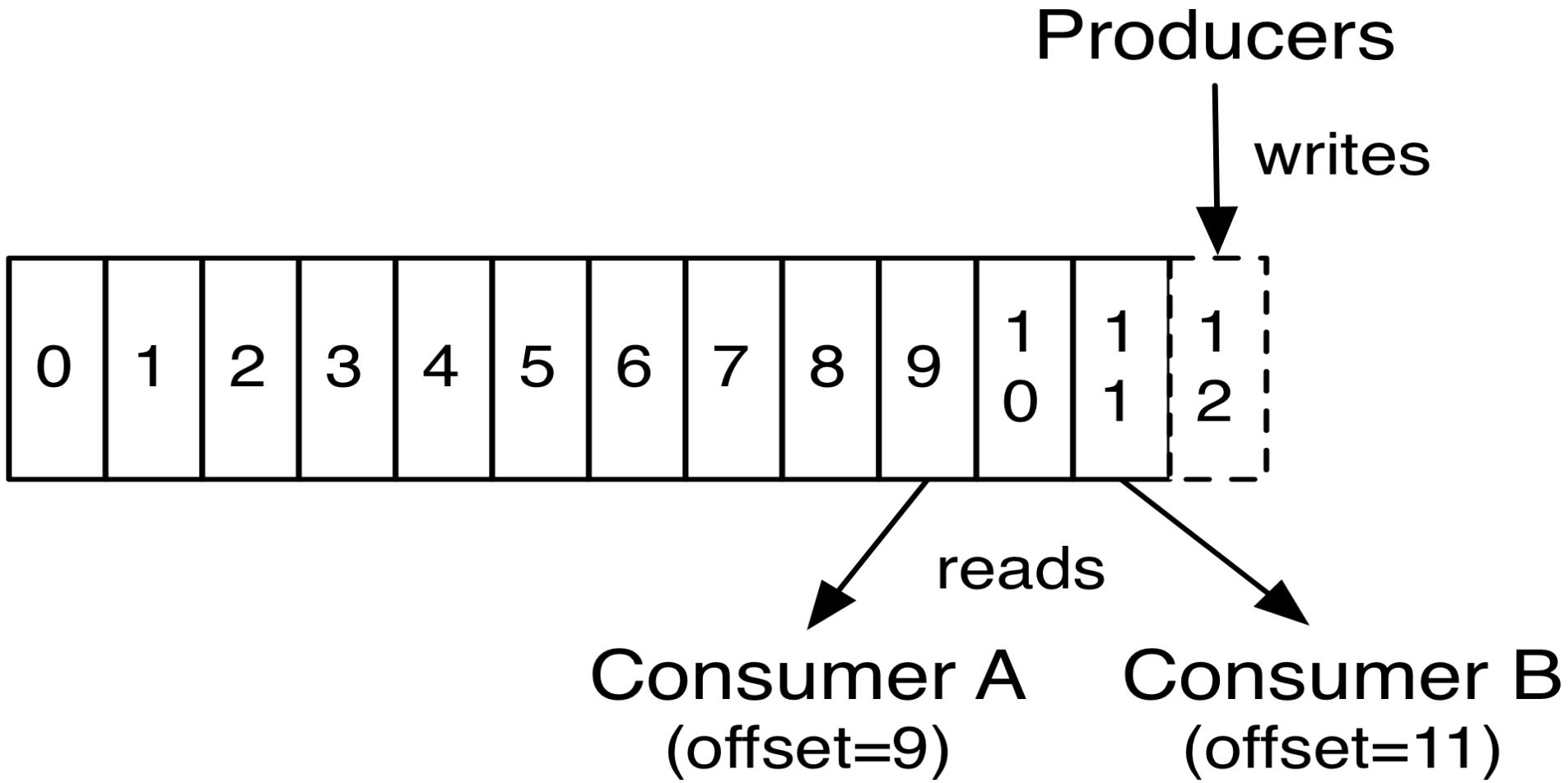
Hands-on: Kafka Tool

- Descárgate Kafka Tool e instálalo
  - https://www.kafkatool.com/download.html



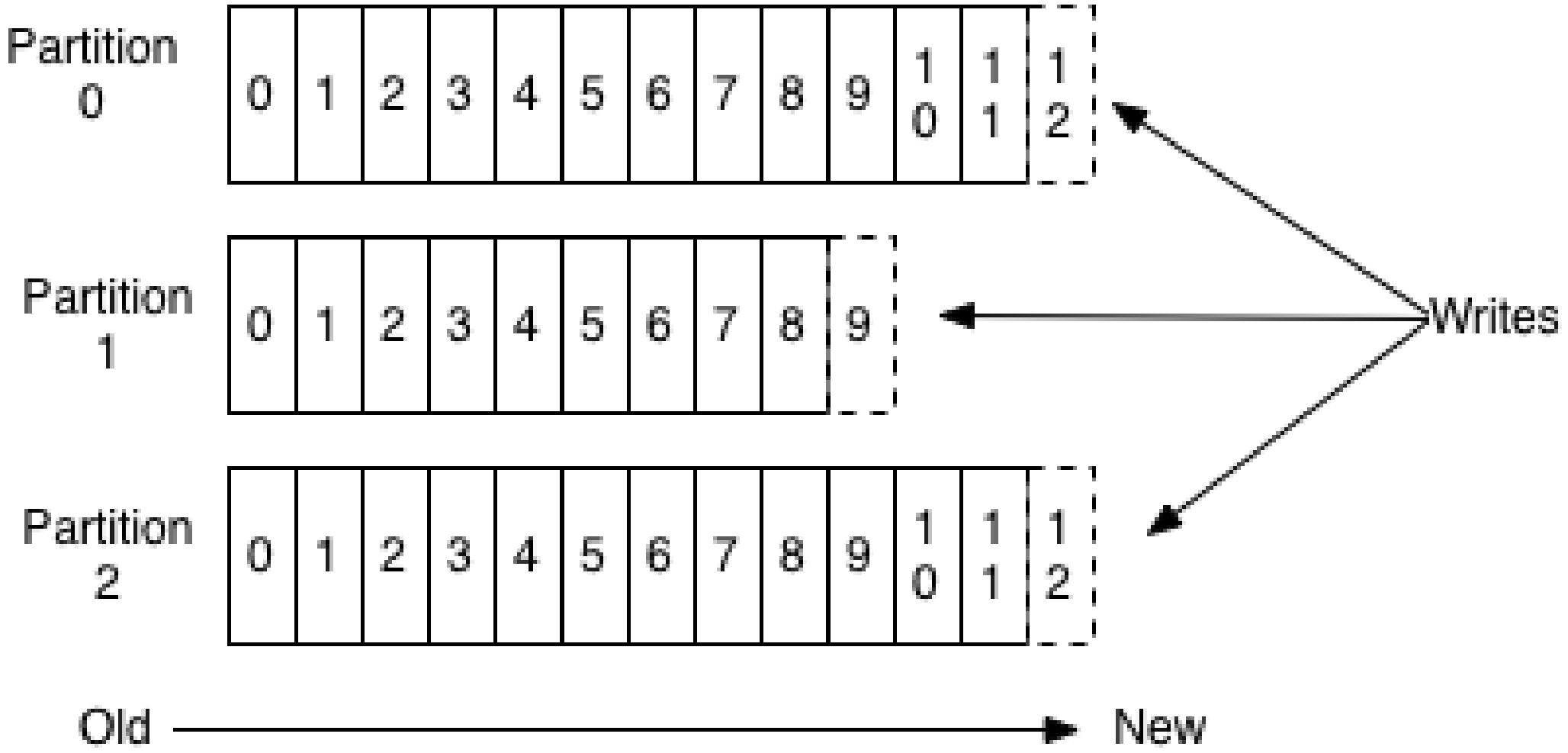
Estructura de un commit log

EDEM
Escuela de Empresarios



Estructura de un Topic

EDEM
Escuela de Empresarios



Estructura de un Topic

EDEM
Escuela de Empresarios

The logs directory is configured in the root at /logs.

/logs

/logs/topicA_0	topicA has one partition.
/logs/topicB_0	topicB has three partitions.
/logs/topicB_1	
/logs/topicB_2	

Estructura de un Topic

EDEM
Escuela de Empresarios

- Topic (lógico)
  - Partition (lógico)
    - Partition Replica (físico, es un directorio)
      - Segment (lógico)
        - File (físico)
        - Segment Index (físico)

- El segment index se utiliza para localizar rápidamente un offset determinado

Hands-on: Topics

- Lista los tópicos existentes:

  $ kafka_2.12-2.3.0/bin/kafka-topics.sh --zookeeper localhost:2181 --list

  $ kafka_2.12-2.3.0/bin/kafka-topics.sh --zookeeper localhost:2181 --create --topic helloworld --partitions 3 --replication-factor 3

Los parámetros que se pasan a los comandos que vamos a ver pueden cambiar de una versión a otra, ya que la tendencia es quitar "responsabilidades" a Zookeeper

Hands-on: Consumer y Producer CLIs

- **Producer**
  - Abre una nueva consola, haz ssh a la VM y ejecuta :

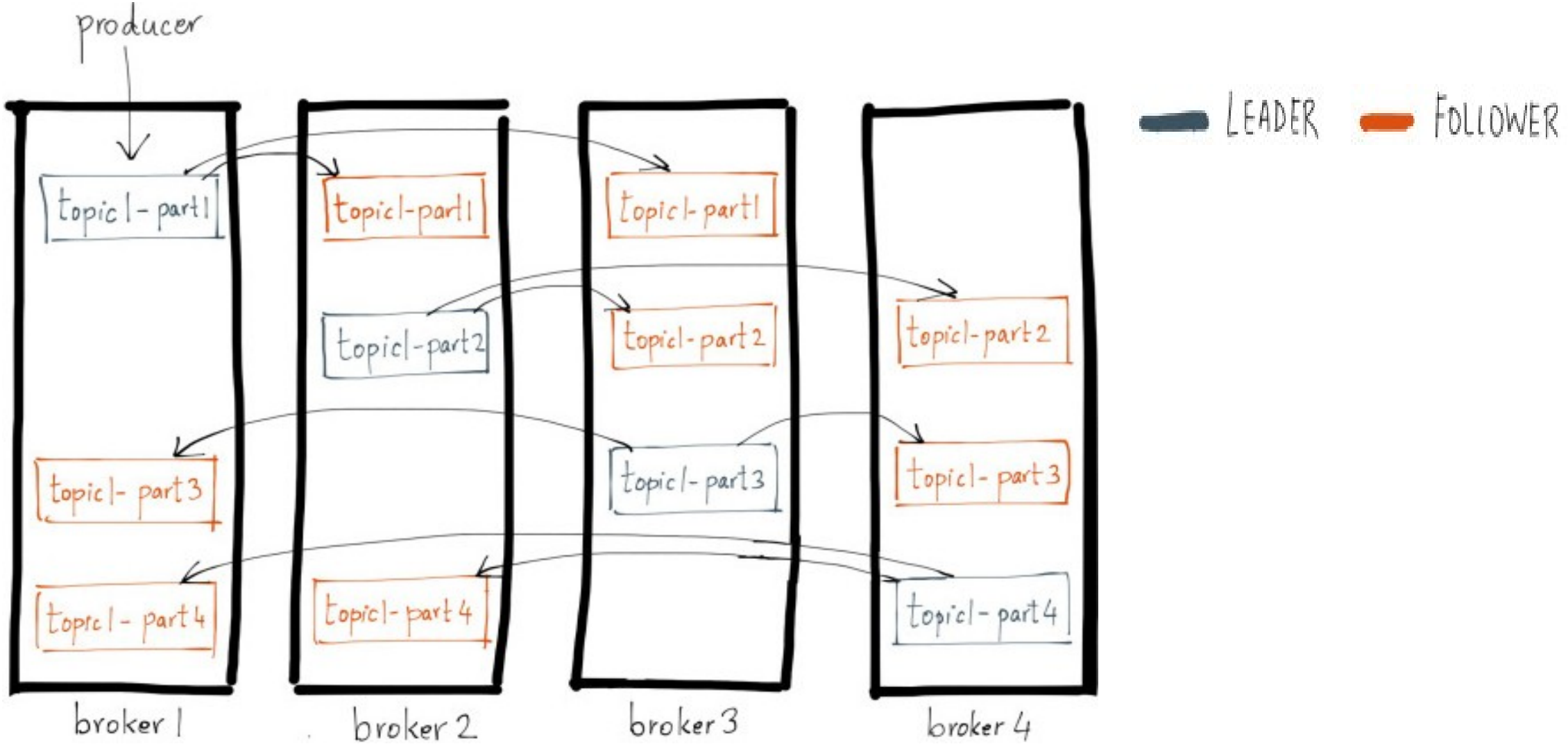
    $ docker-compose exec kafka kafka-console-producer --broker-list localhost:9092 --topic helloworld

- **Consumer**
  - Abre una nueva consola, haz ssh a la VM y ejecuta :

    $ docker-compose exec kafka kafka-console-consumer --bootstrap-server localhost:9092 --topic helloworld --from-beginning

Replicación de Particiones

EDEM
Escuela de Empresarios



Algunos comandos útiles

EDEM
Escuela de Empresarios

- Aumentar el nº de particiones a un tópico existente:

```
kafka-topics.sh --zookeeper localhost:2181 --alter --topic helloworld --partitions 10
```

Algunos comandos útiles

EDEM
Escuela de Empresarios

- Cambiar el factor de replicación
- Crea un fichero increase-replication-factor.json

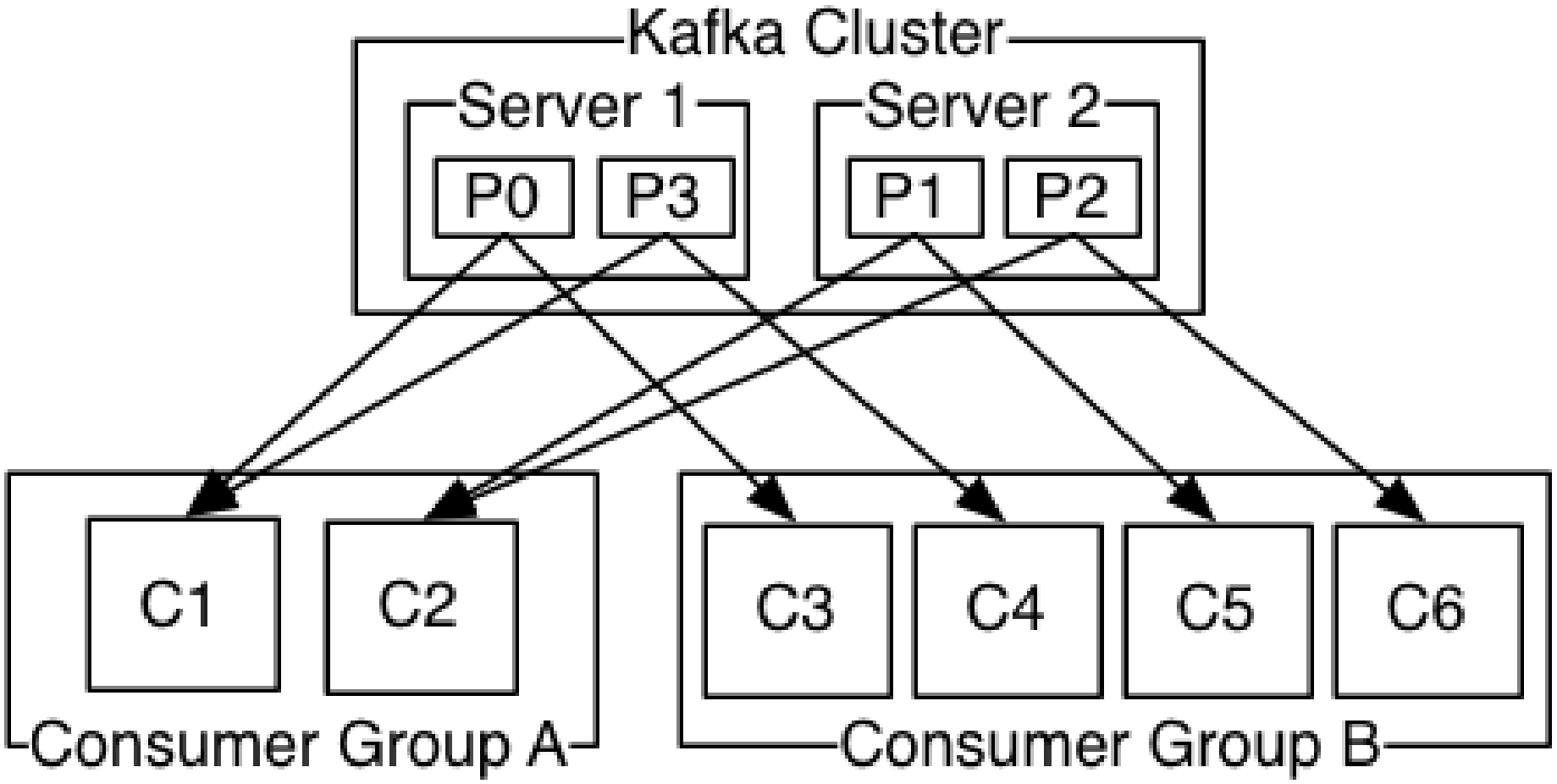
```
{"version":1,
  "partitions":[
     {"topic":"helloworld","partition":0,"replicas":[0,1,2]},
     {"topic":"helloworld","partition":1,"replicas":[0,1,2]},
     {"topic":"helloworld","partition":2,"replicas":[0,1,2]}
]}
```

kafka-reassign-partitions --zookeeper <zookeeper_external_ip>:2181 --reassignment-json-file increase-replication-factor.json --execute

kafka-topics --zookeeper localhost:2181 --topic vehicles --describe

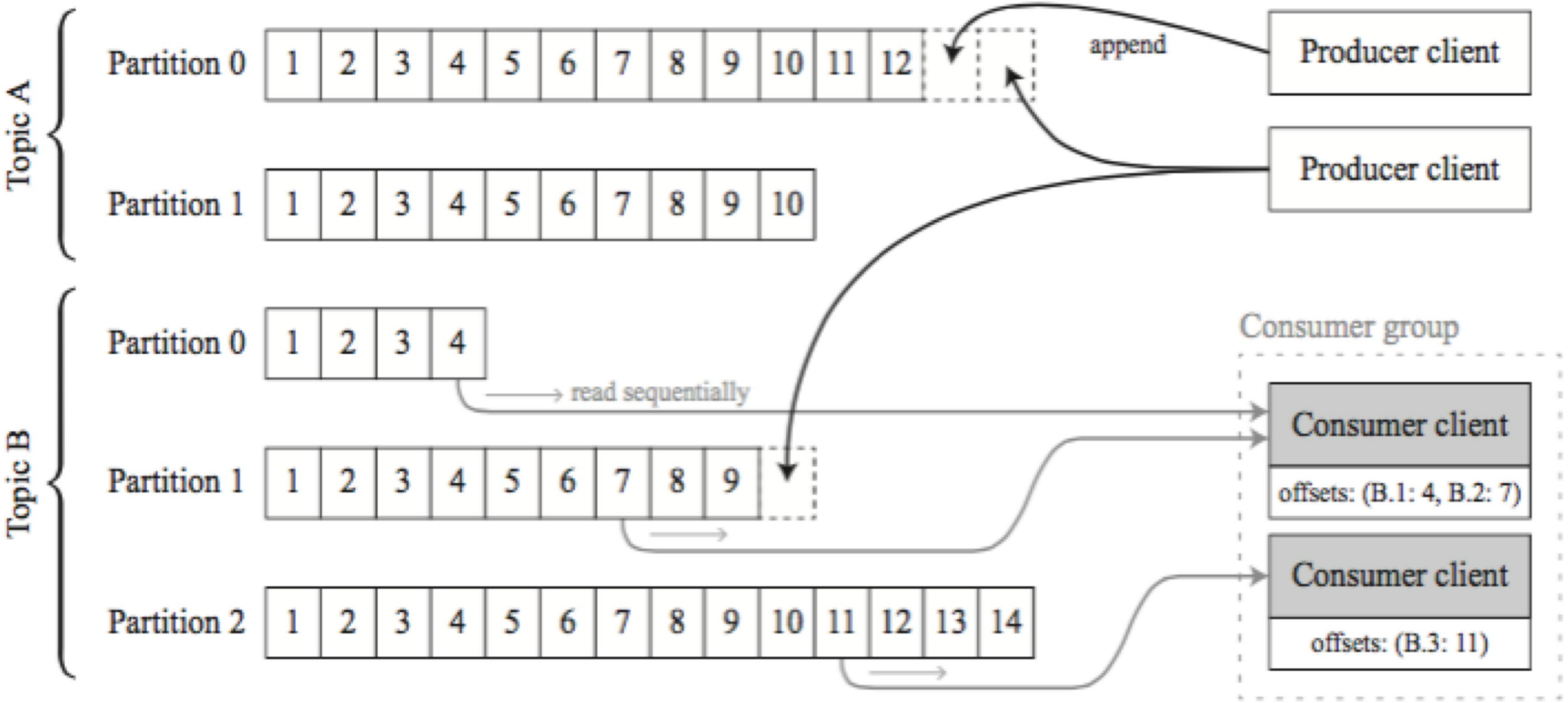
Grupos de Consumidores

EDEM
Escuela de Empresarios



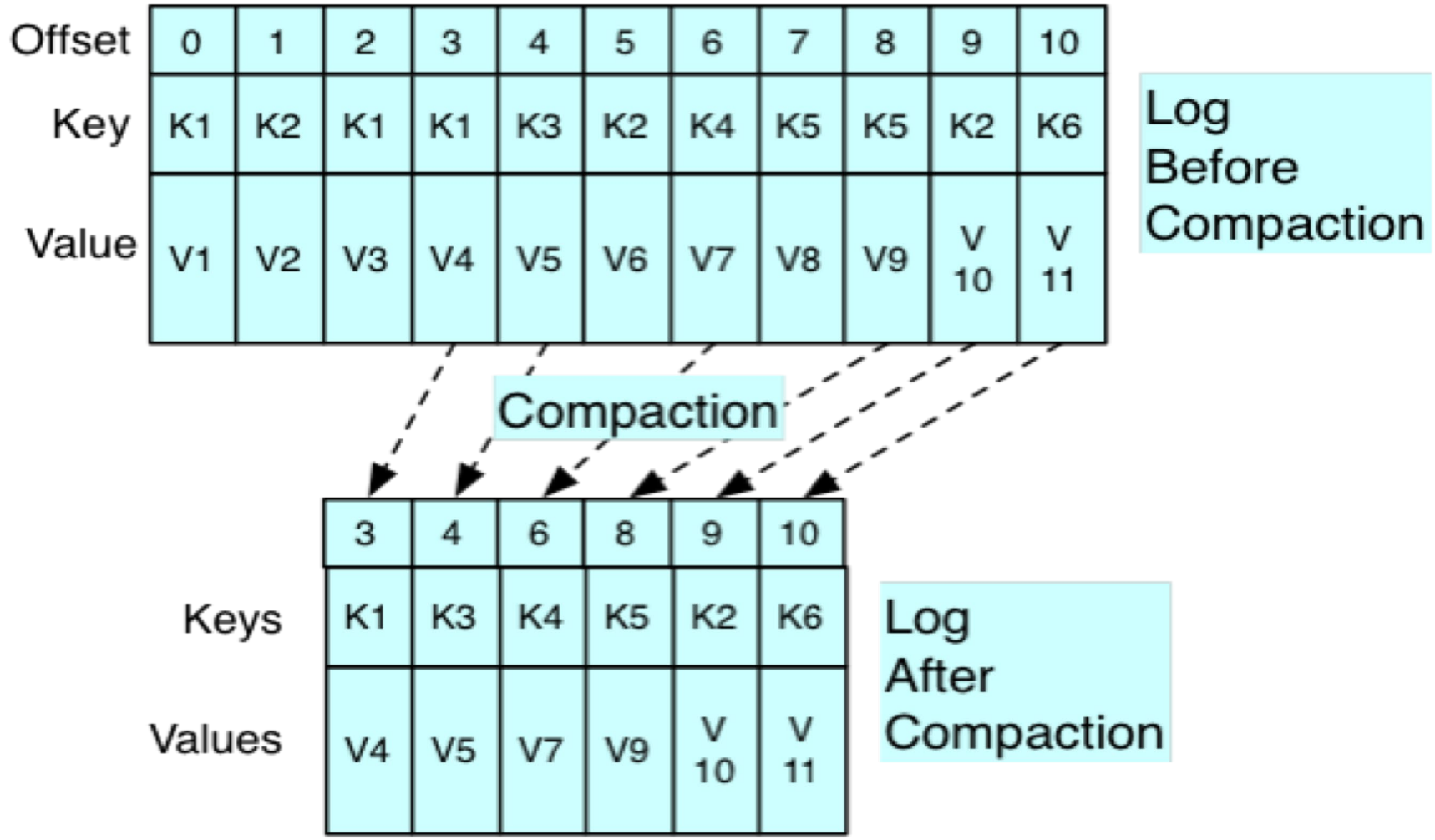
EDEM
Escuela de Empresarios

Consumer Groups



Log Compaction

EDEM
Escuela de Empresarios



Log Compaction

EDEM
Escuela de Empresarios

- Log compaction puede ser útil para:
  - Database change subscription
  - Event Sourcing (ES)
  - Journaling para alta disponibilidad (high-availability)

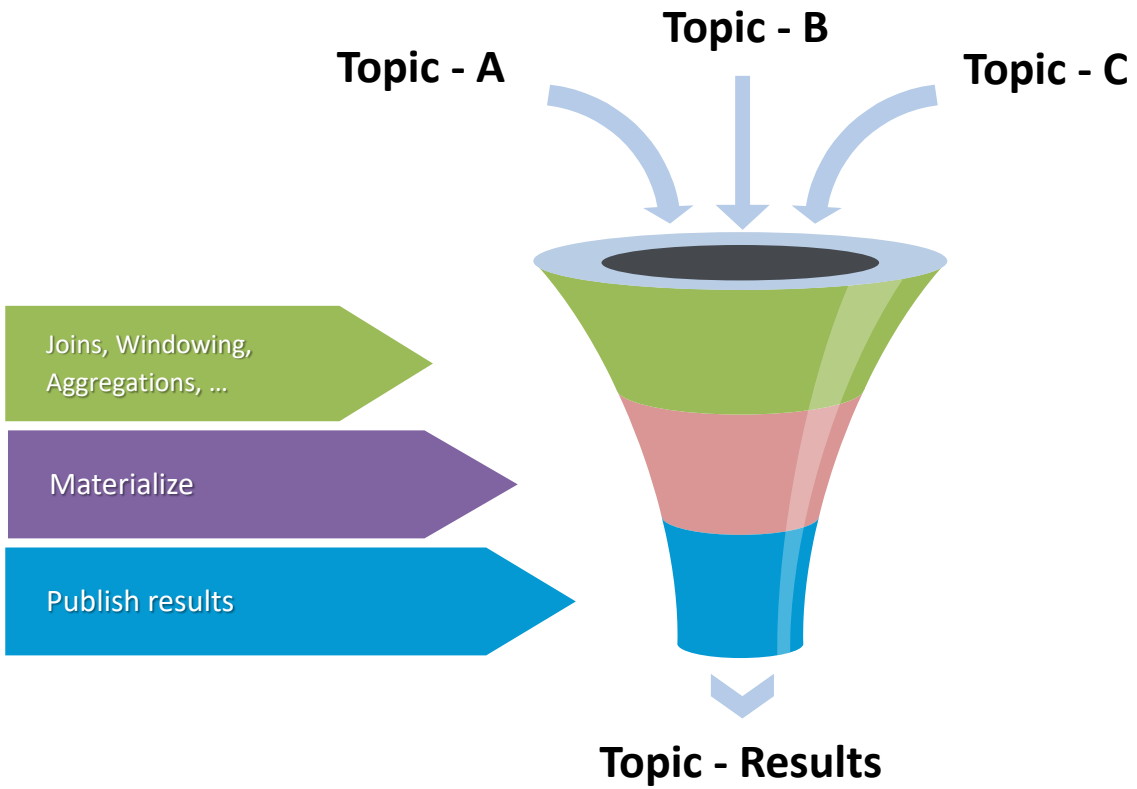
Kafka Streams y Kafka SQL

EDEM
Escuela de Empresarios

- **Stream**
  - Un **stream** es la abstracción más importante proporcionada por Kafka Streams: representa un conjunto de datos **ilimitado** que se actualiza continuamente
  - Una **partición de stream** es una secuencia ordenada, reproducible y tolerante a fallos de registros de datos inmutables, donde un registro de datos se define como un par clave-valor.
- **Aplicación de procesamiento de streaming**
  - Cualquier programa que utilice la biblioteca Kafka Streams. Puede definir su lógica computacional a través de una o más topologías de procesador.

Kafka Streams y Kafka SQL

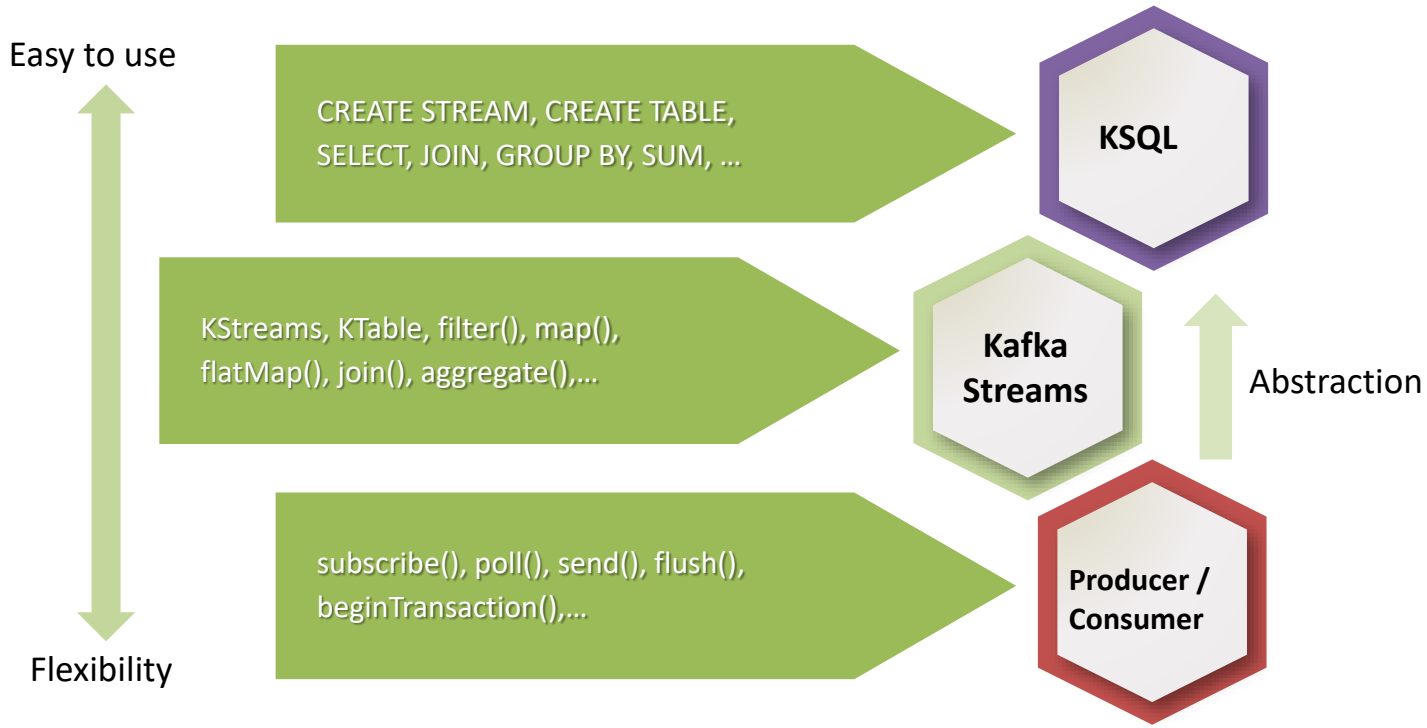
EDEM
Escuela de Empresarios



Kafka Streams y Kafka SQL

EDEM
Escuela de Empresarios

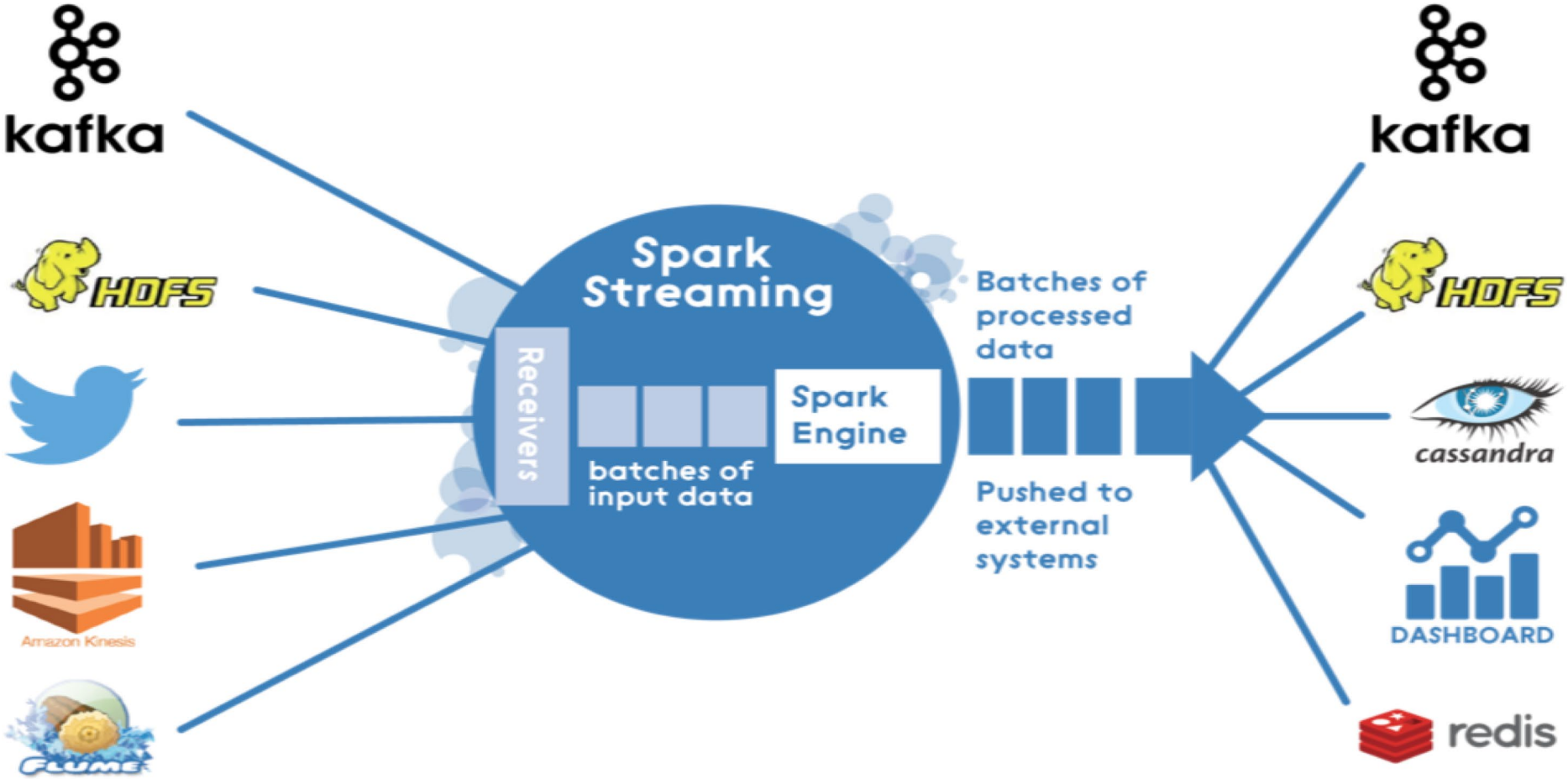
```
CREATE TABLE visitedLocationsPerUser AS SELECT username, COUNT(*)
FROM myStream GROUP BY username;
```



Casos de Uso

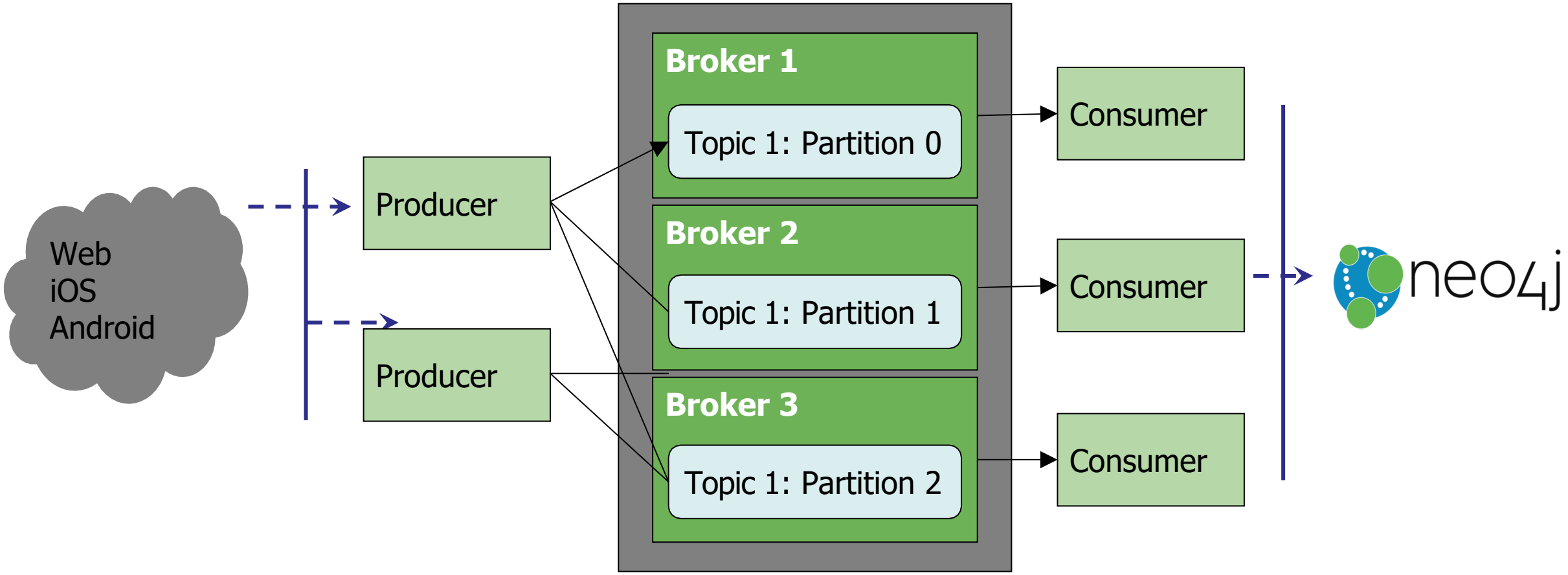
EDEM
Escuela de Empresarios

- Stream processing architectures



Casos de Uso

- Website activity tracking



Common use cases of Apache Kafka

EDEM
Escuela de Empresarios

- Agregación de logs y métricas