Большие данные

613x-010402D x={1,2,3} *осень 2025*

Лекция 2: Kafka – распределенная платформа потоковой передачи сообщений

Сергей Борисович Попов
sepo@ssau.ru

Материалы лекций:
https://1drv.ms/f/c/5ed33c8b23e26391/EpMzlOhQgt5Oq-cqHF_ChjwB0NP5AovVgqYTfBokEY1zhA

Современное программное обеспечение – это мир ***сервис-ориентированных*** архитектур: идея, возникшая в начале века, основанная на том, что компании перестраиваются вокруг использования разделяемых сервисов, которые делают простые и полезные вещи.

Организационные изменения, которые привнёс полностью сервис-ориентированный подход, означали не просто переход от монолитных приложений, но вывод на передний план организацию внутрисистемных взаимодействий. Приложения стали платформами, системами взаимодействующих сервисов.

На пути такой эволюции (проблемы стабильности сложных зависимостей, версионирования) в компании LinkedIn важнейшей вехой стало использование системы обмена сообщениями собственной разработки – ***Kafka***. Kafka добавила в общую архитектуру асинхронную модель ***издатель-подписчик***, позволившую передавать внутри организации триллионы сообщений в день. Это было важно для компании в стадии активного роста, т. к. позволяло подключать новые приложения, не затрагивая хрупкую сеть синхронных взаимодействий, на которых держался пользовательский интерфейс.

Событийно-ориентированные архитектуры уменьшают связность системы: события изолируют сервисы, позволяют гораздо проще подключать к потокам событий реального времени новые системы.
Варианты реализации: системы обмена сообщениями, сервисные шины предприятия (enterprise service bus – ESB).
Проблемы: хронология, процессы передачи данных.

Альтернатива: воспроизводимые журналы событий, которые также отделяют сервисы друг от друга, но кроме этого они предоставляют единое хранилище, отказоустойчивое и масштабируемое – ***общий источник истины***, на который может сослаться любое приложение.

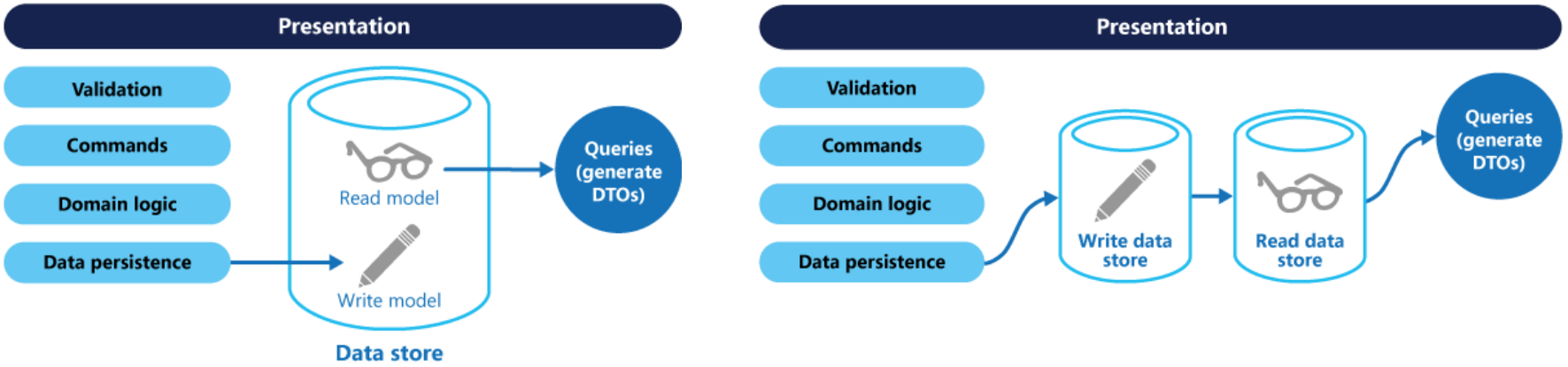
Стопфорд, Бен

Проектирование событийно-ориентированных систем: Концепции и шаблоны проектирования сервисов потоковой обработки данных с использованием Apache Kafka / Бен Стопфорд ; Пер. с англ. — 2-е изд., испр. — Иркутск : ITSumma Press, 2019. – 175 с.

ISBN 978-5-6042412-1-9

CQRS означает *Command and Query Responsibility Segregation* (разделение ответственности команд и запросов), шаблон, который разделяет операции чтения и обновления для хранилища данных.
CQRS разделяет чтение и запись на разные модели, используя команды для обновления данных и запросы для чтения данных.

- Команды должны быть основаны на задачах, а не на данных.
- Команды могут быть помещены в очередь для асинхронной обработки, а не обрабатываться синхронно.
- Запросы никогда не изменяют базу данных. Запрос возвращает DTO, который не инкапсулирует никаких знаний о домене.



Сообщения. Хотя CQRS не требует сообщений, сообщения обычно используются для обработки команд и публикации событий обновления. В этом случае приложение должно обрабатывать сбои сообщений или дублирующие сообщения.

The Art of Immutable Architecture

Theory and Practice of Data Management in Distributed Systems

qedcode.com

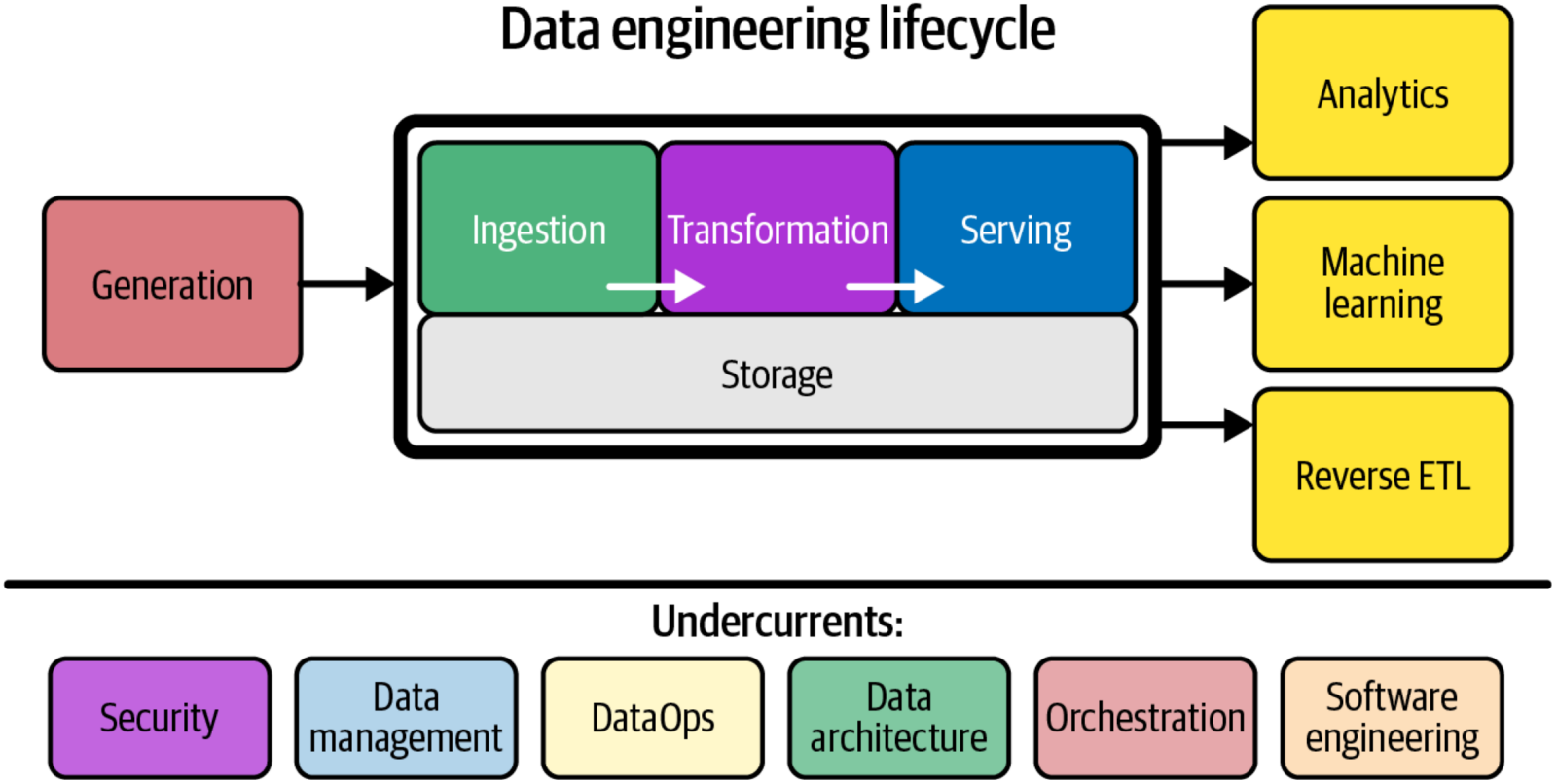
Искусство архитектуры неизменяемых объектов

Майкл Л. Перри

П26 Искусство неизменяемой архитектуры: теория и практика управления данными в распределенных системах / пер. с анг. С. В. Минца; науч. ред. В. С. Яценков. – М.: ДМК Пресс, 2022. – 388 с.: ил.

ISBN 978-5-93700-111-5

Жизненный цикл инженерии данных



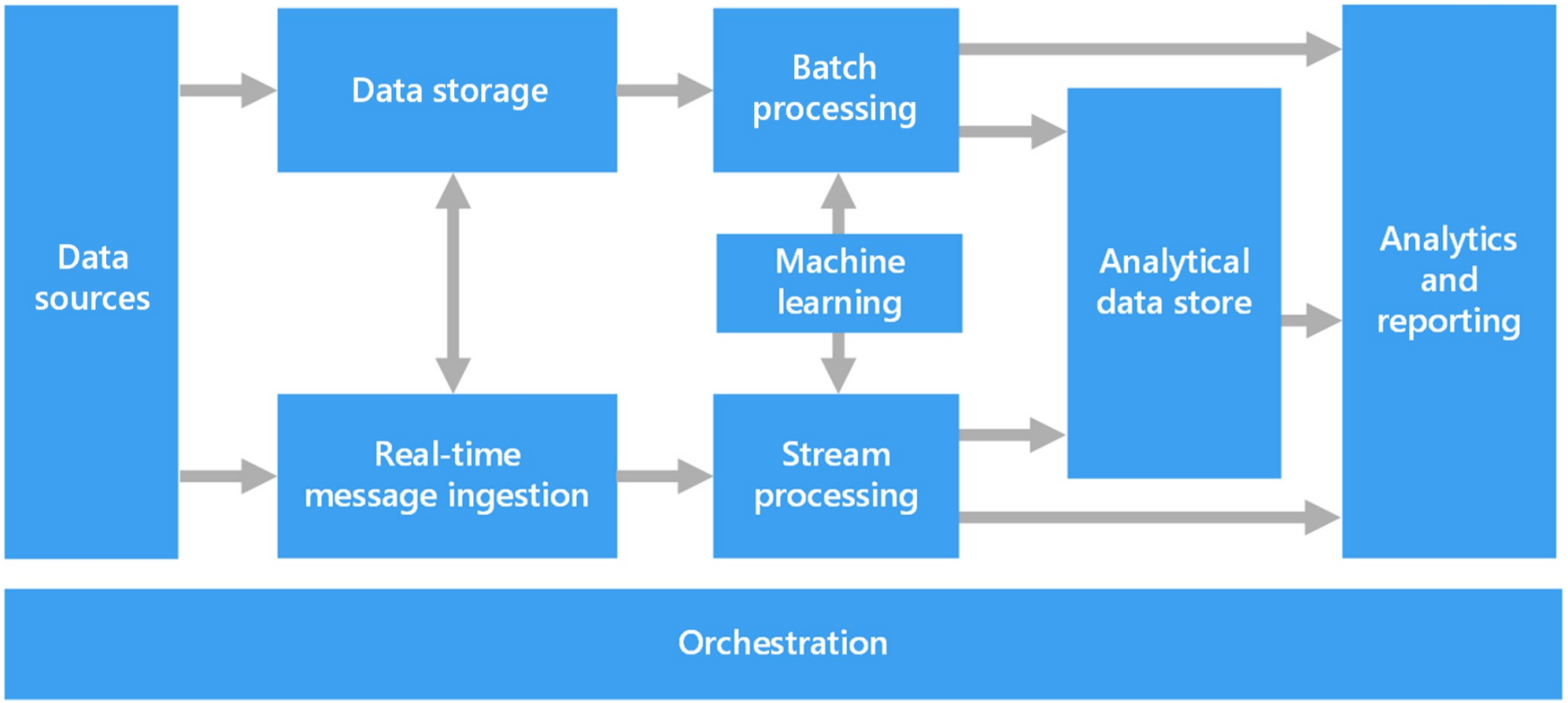
- Генерация
- Хранение
- Приём/Сбор
- Преобразование
- Предоставление

- Аналитика
- Машинное обучение
- Обратное ETL

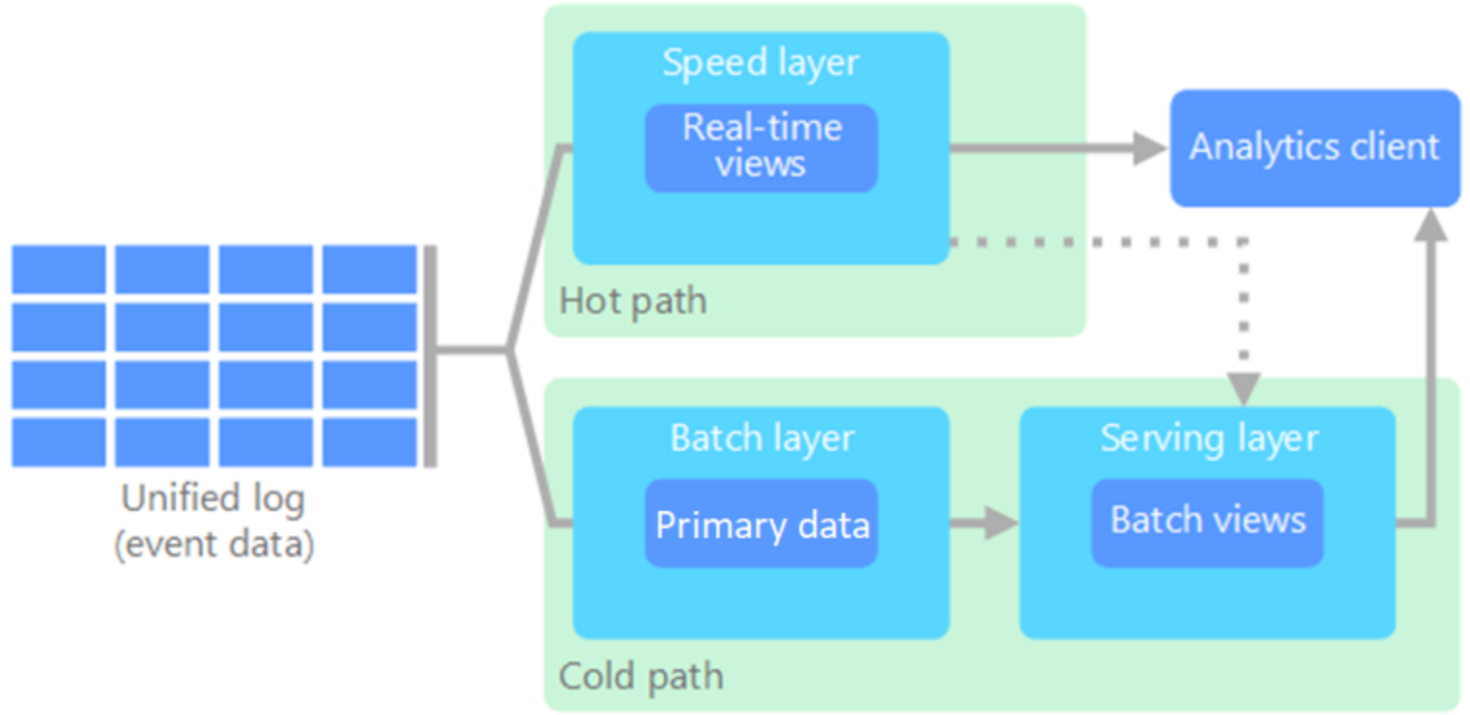
Фоновые процессы:

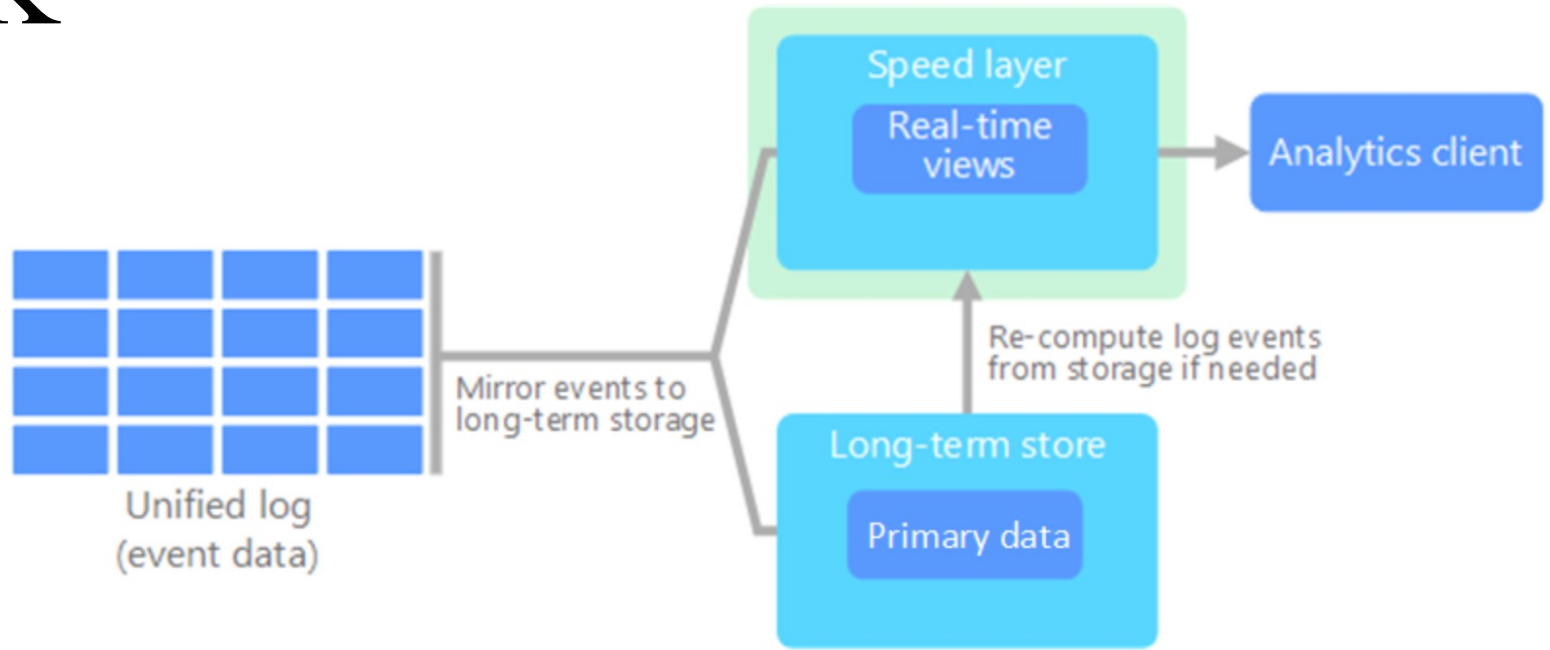
- Безопасность
- Управление данными
- DataOps
- Архитектура данных
- Оркестровка
- Программная инженерия

Логические компоненты архитектуры больших данных



Λ vs. K


Speed layer
Real-time views
Analytics client
Hot path
Unified log
(event data)
Batch layer
Primary data
Serving layer
Batch views
Cold path

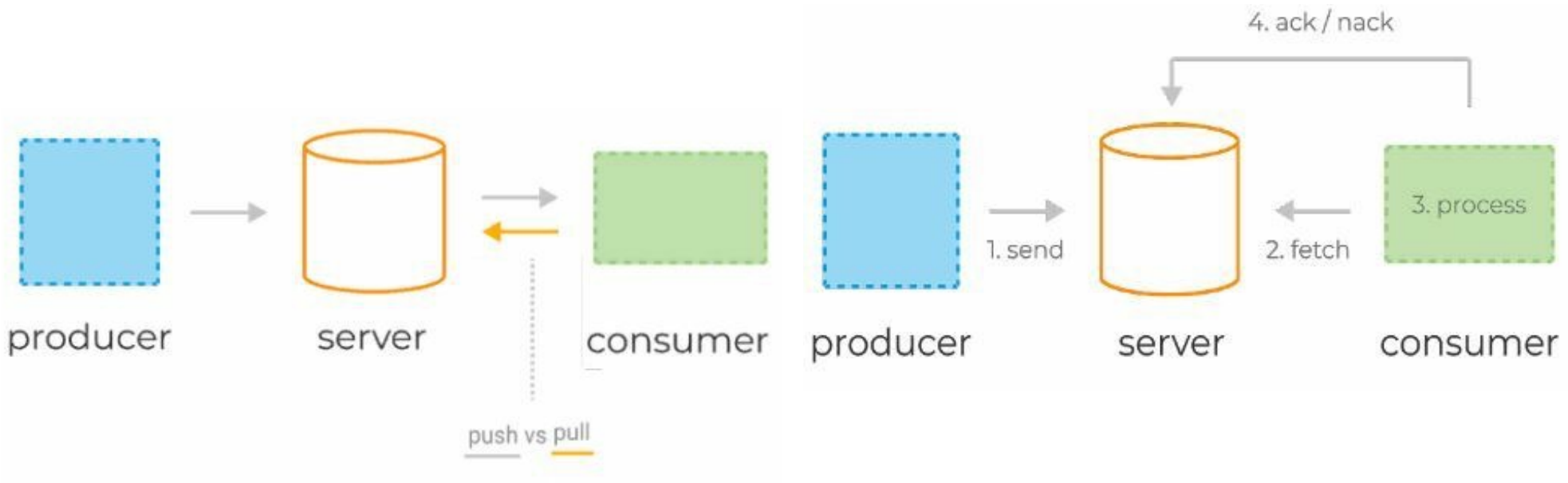

Speed layer
Real-time views
Analytics client
Mirror events to long-term storage
Re-compute log events from storage if needed
Long-term store
Primary data
Unified log
(event data)

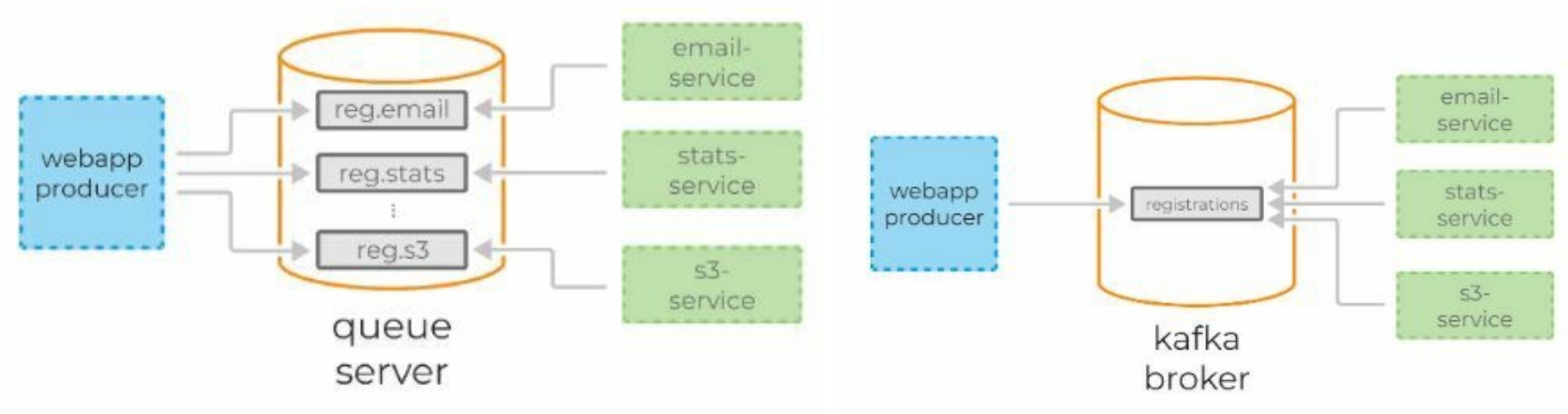
Apache Kafka is an open-source distributed event streaming platform

Kafka — это:

- масштабируемая платформа сообщений реального времени, обрабатывающая миллионы сообщений в секунду;
- платформа потоковой передачи сообщений для аналитики больших данных и маленьких объемов транзакционной обработки;
- распределенное хранилище, обеспечивающее настоящее разделение для обработки замедленной обратной связи, с поддержкой различных протоколов соединений и воспроизведением событий в нужном порядке;
- фреймворк интеграции данных для потоковой передачи ETL;
- фреймворк обработки данных для непрерывной обработки stateless- или stateful-потоков.

Kafka vs. классические сервисы очередей





Источник:

Kafka за 20 минут. Ментальная модель и как с ней работать

https://habr.com/ru/companies/kuper/articles/738634/

Kafka — это архитектурный слой распределенного хранения, который отделяет производителей от потребителей.
Кроме того, Kafka-нативные обработчики вроде Kafka Streams and ksqlDB обеспечивают ***обработку в реальном времени***.

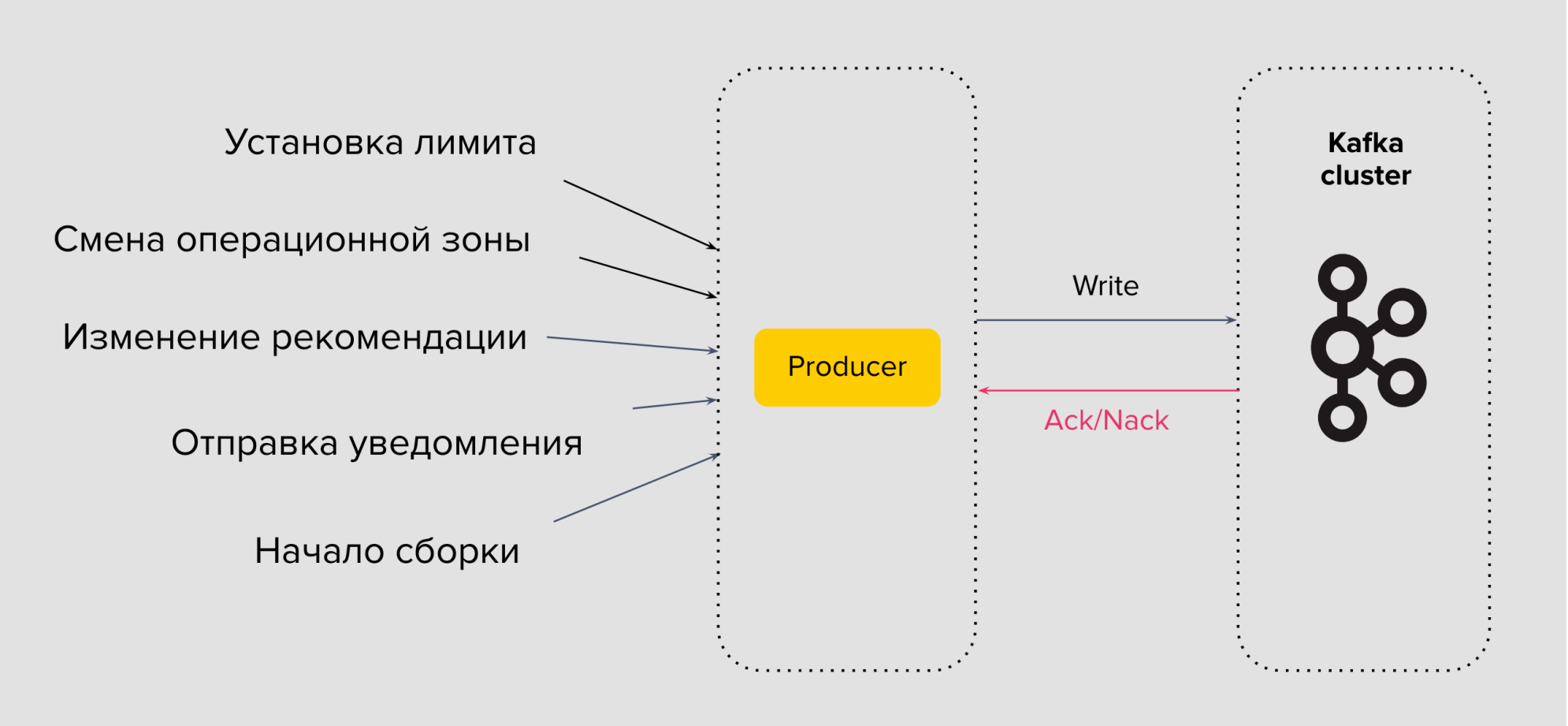
Основы кластера Kafka – это ***продюсер***, ***брокер*** и ***консьюмер***.

Продюсер пишет сообщения в лог брокера, а консьюмер его читает.

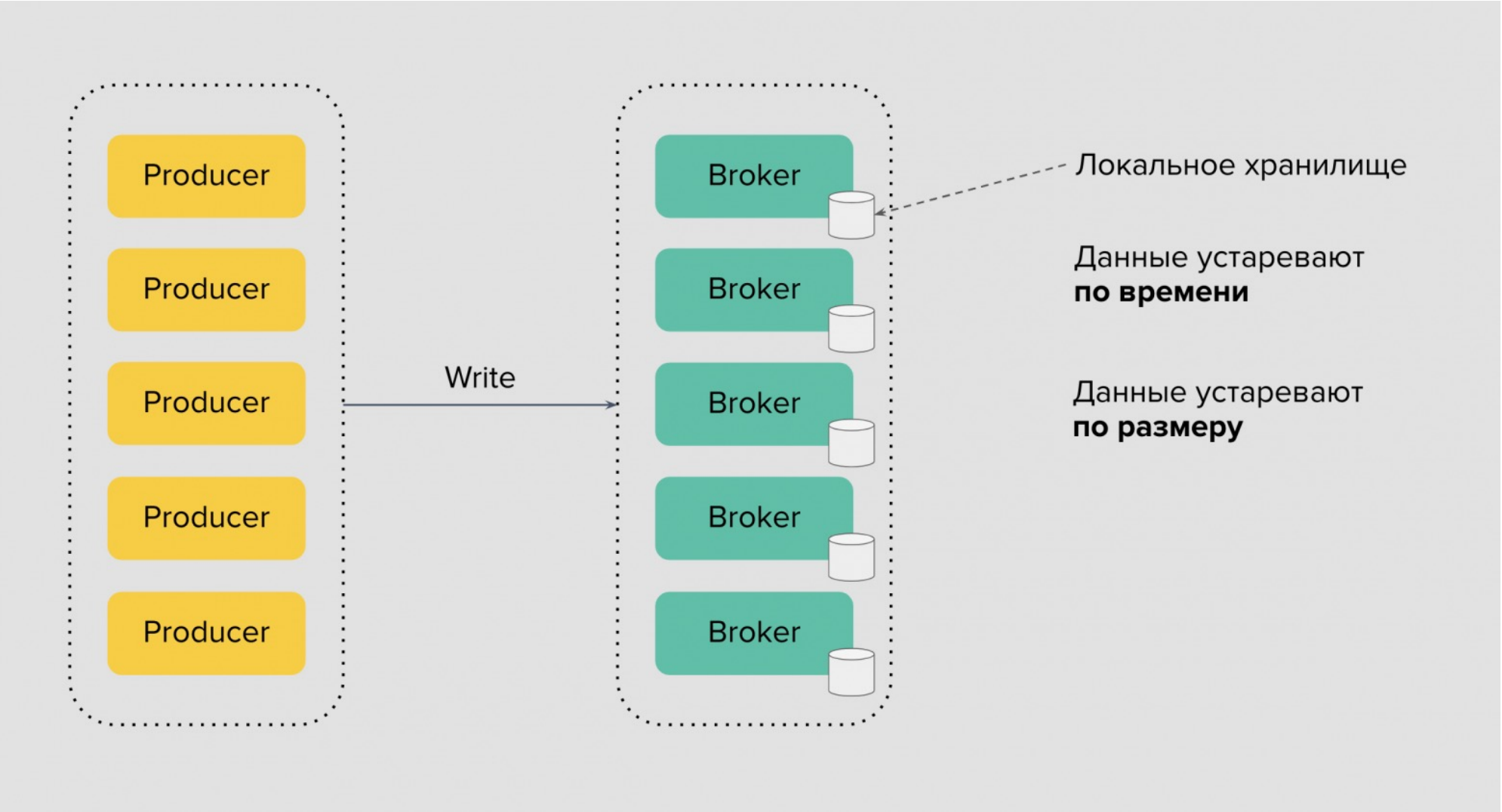
Лог — это упорядоченный поток событий во времени.
Событие происходит, попадает в конец лога и остаётся там неизменным.

Apache Kafka управляет логами и организует платформу, которая соединяет поставщиков данных с потребителями и даёт возможность получать упорядоченный поток событий в реальном времени.

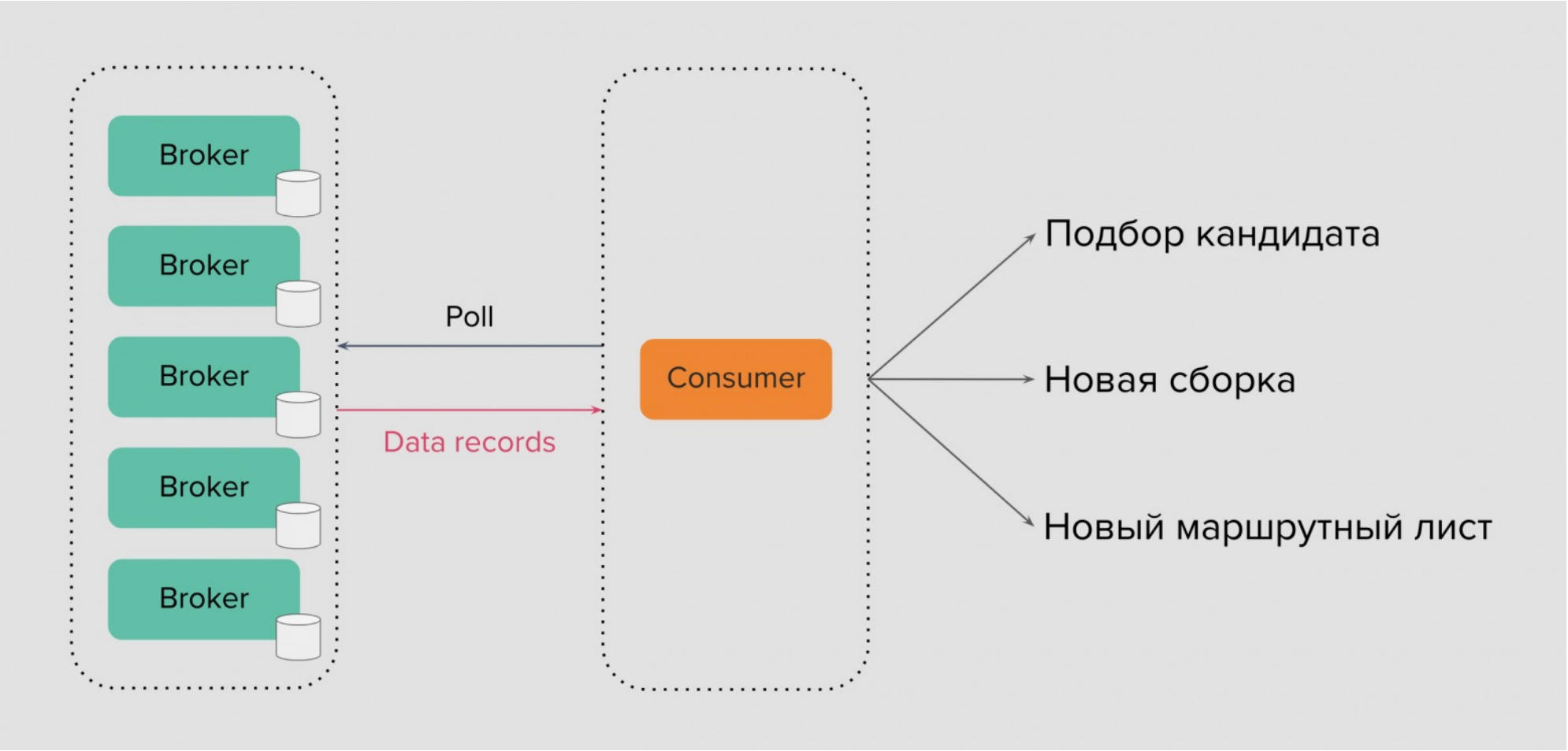
Продюсеры


Установка лимита
Смена операционной зоны
Изменение рекомендации
Отправка уведомления
Начало сборки
Producer
Write
Ack/Nack
Kafka cluster

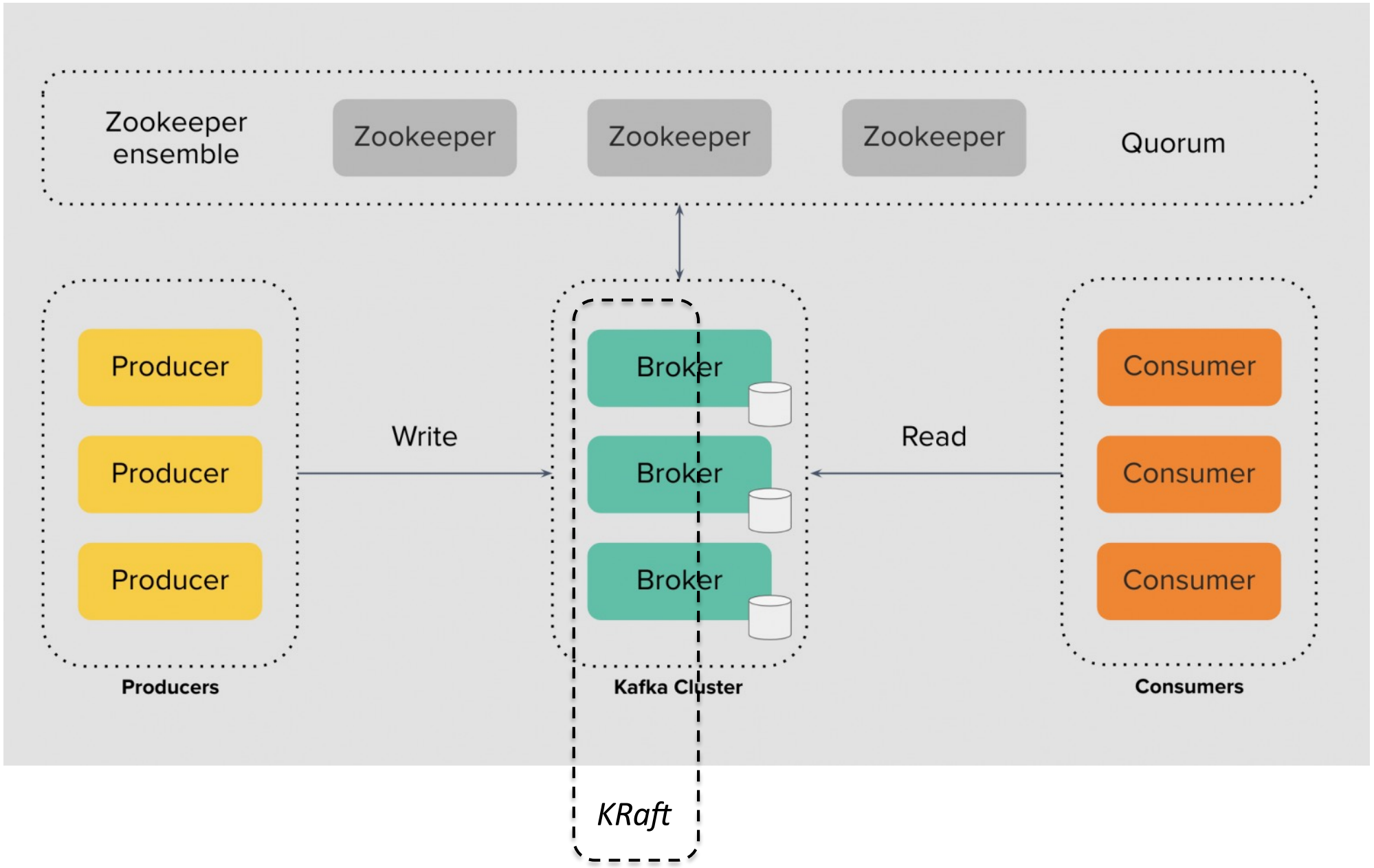
Брокеры Kafka-кластер



Консьюмеры

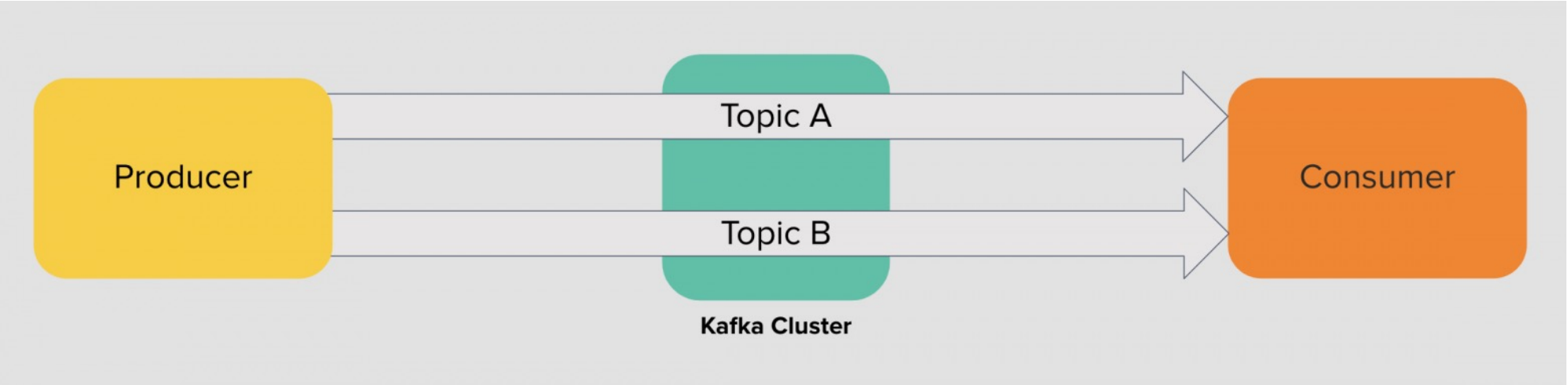

Broker
Broker
Broker
Broker
Broker
Poll
Data records
Consumer
Подбор кандидата
Новая сборка
Новый маршрутный лист

Архитектура Kafka



Данные в брокерах

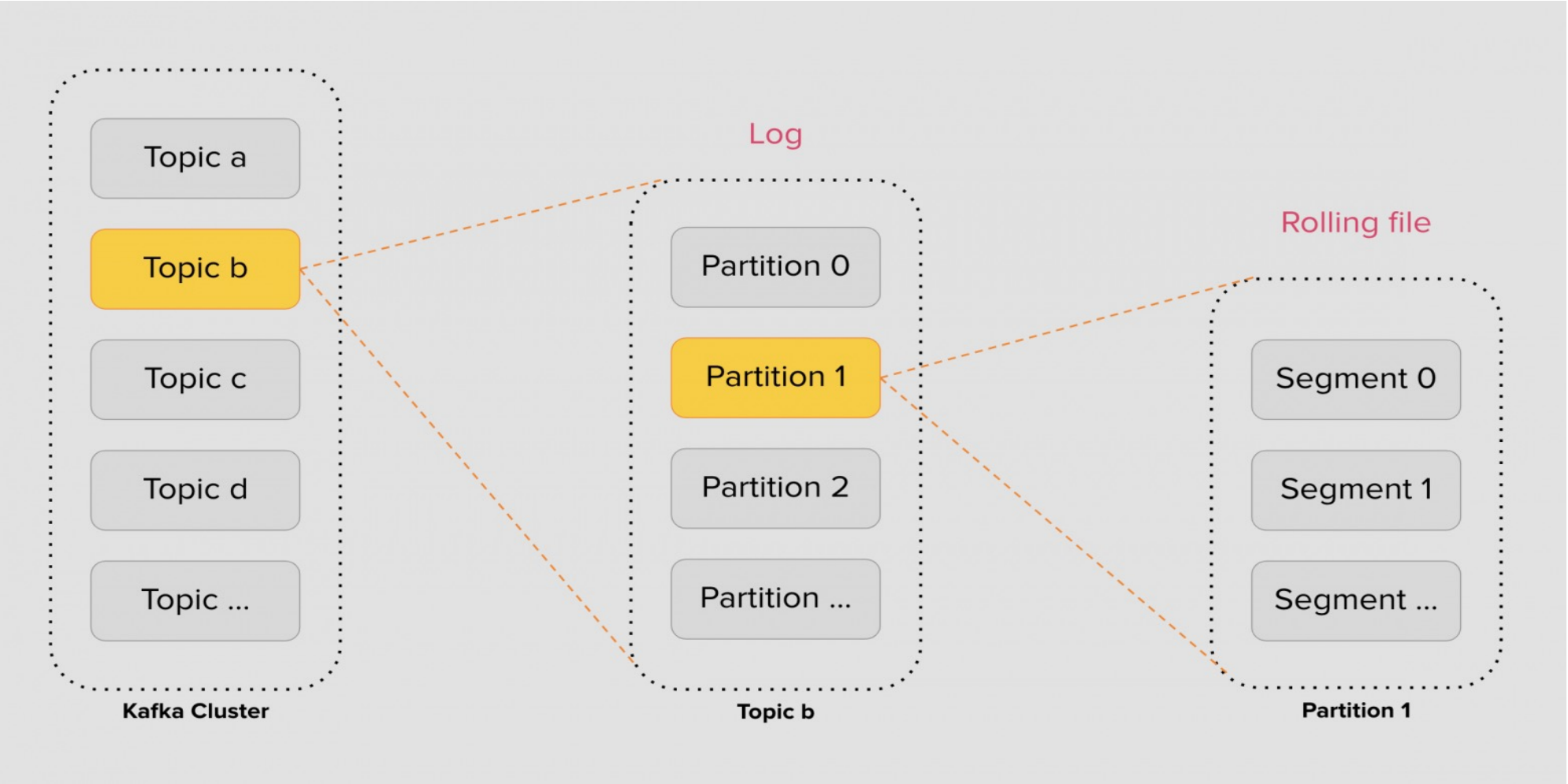
Топик — это логическое разделение категорий сообщений на группы.
Топики в Kafka разделены на разделы (партиции, partition).
Партиции находятся на одном или нескольких брокерах, что позволяет кластеру масштабироваться. Партиции хранятся на локальных дисках брокеров и представлены набором лог-файлов – ***сегментов***.
Запись в сегменты идёт в конец, а уже сохранённые события неизменны.
Каждое сообщение в таком логе определяется порядковым номером — ***оффсетом***. Этот номер монотонно увеличивается при записи для каждой партиции.



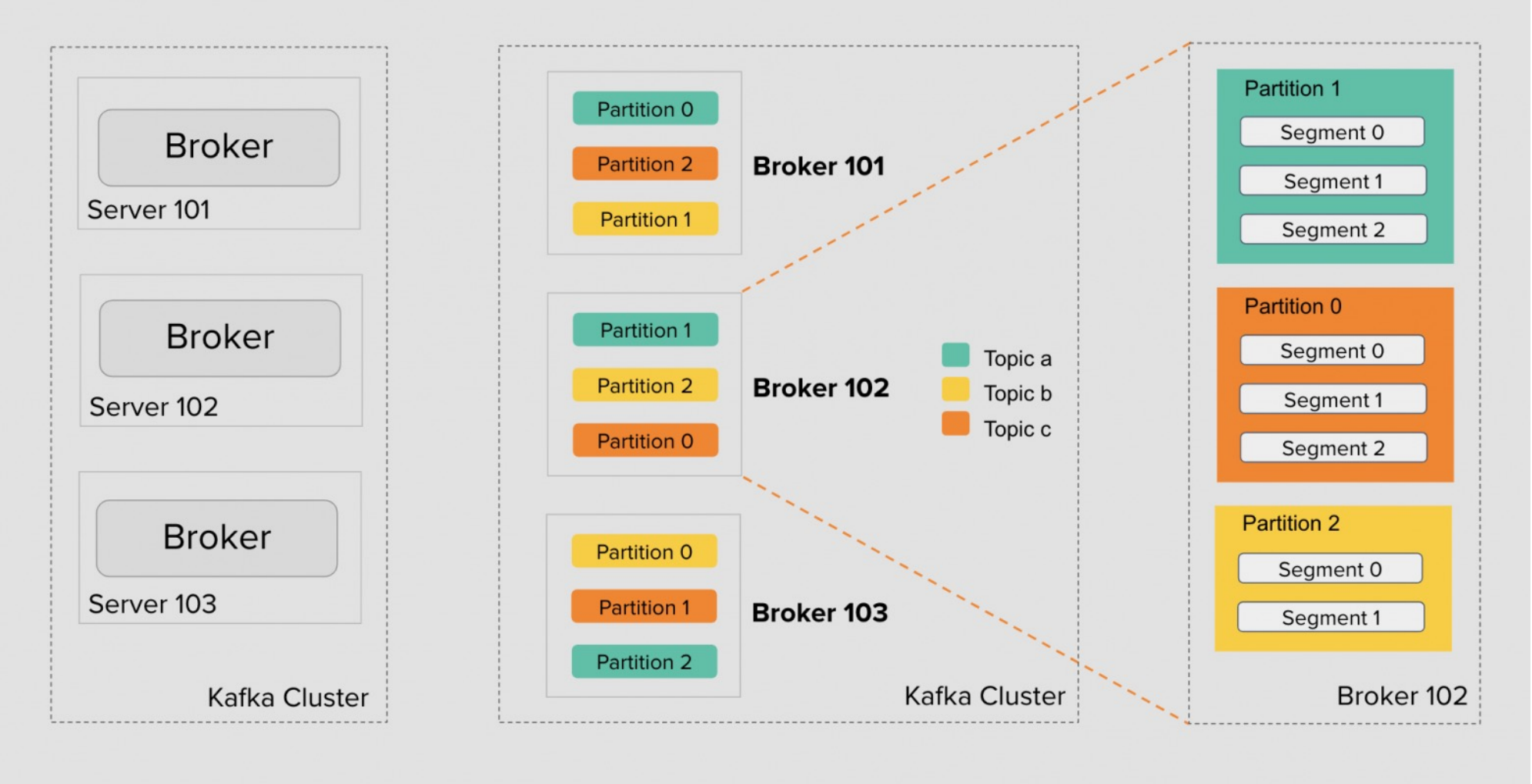
- Один продюсер может писать в один или несколько топиков
- Один консьюмер может читать один или несколько топиков
- В один топик могут писать один или более продюсеров
- Из одного топика могут читать один или более консьюмеров

Данные в брокерах

Формально партиция – строго упорядоченный лог сообщений.
Каждое сообщение добавляется в конец без возможности изменить его в будущем.
При этом сам топик в целом не имеет никакого порядка, но порядок сообщений всегда соблюдается в каждой партиции.

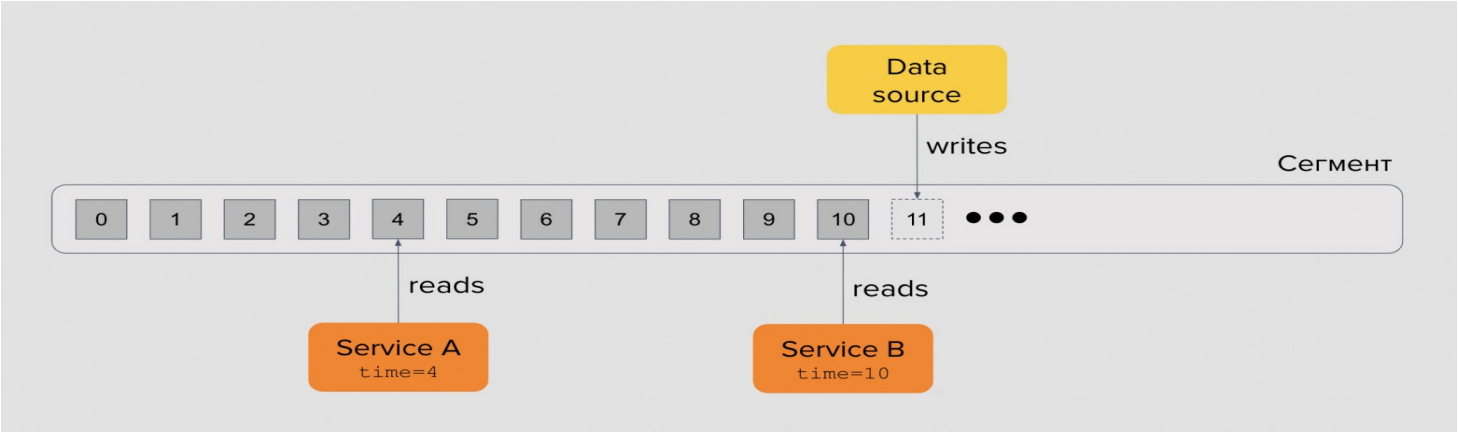


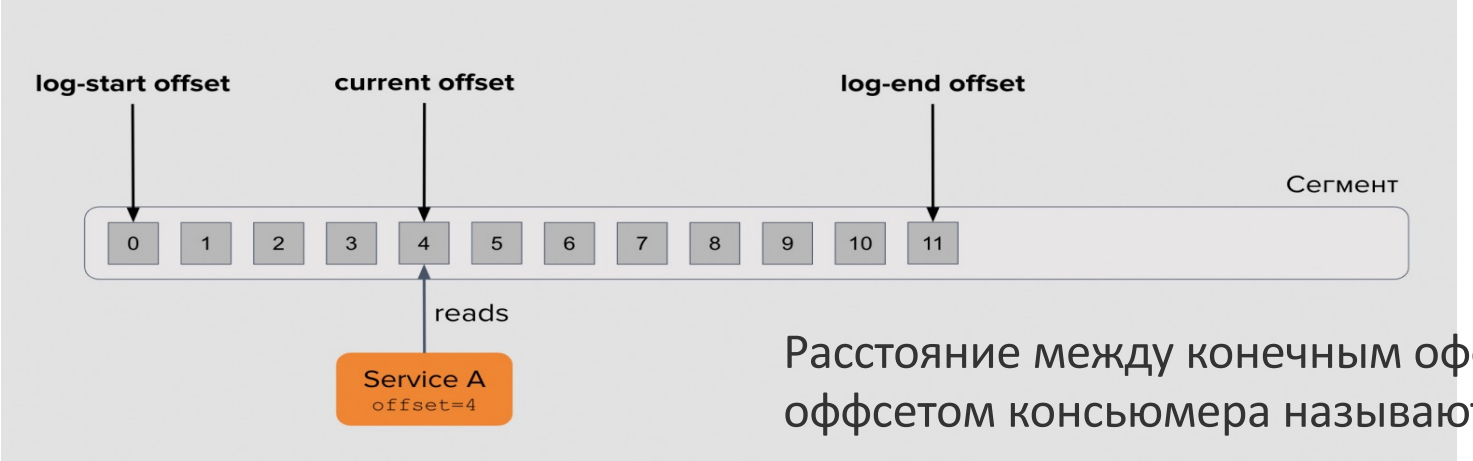
Данные в брокерах



Сами партиции физически представлены на дисках в виде ***сегментов***. Это отдельные файлы, которые можно создать, ротировать или удалить в соответствии с настройкой устаревания данных в них.

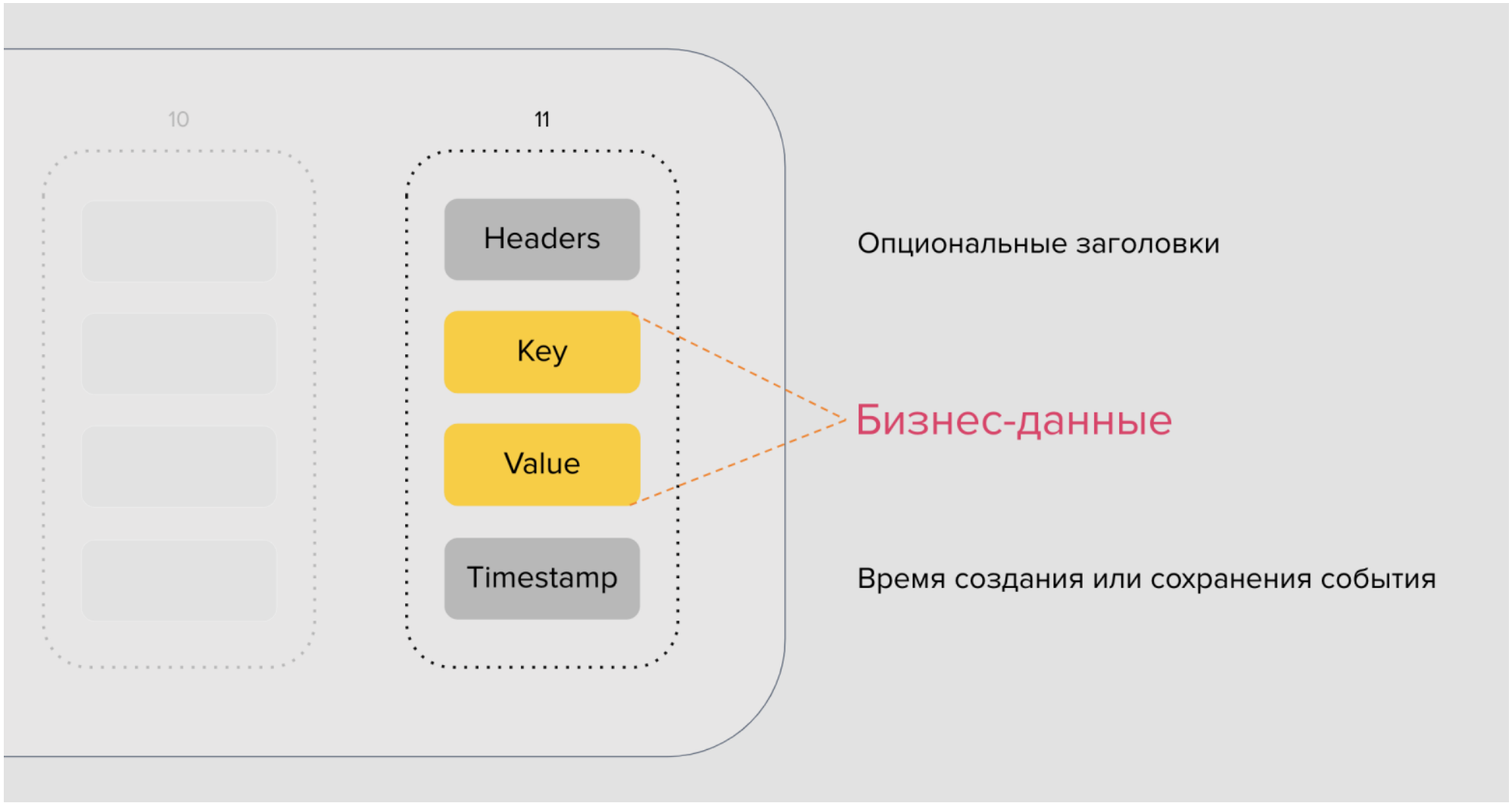
Поток данных как лог





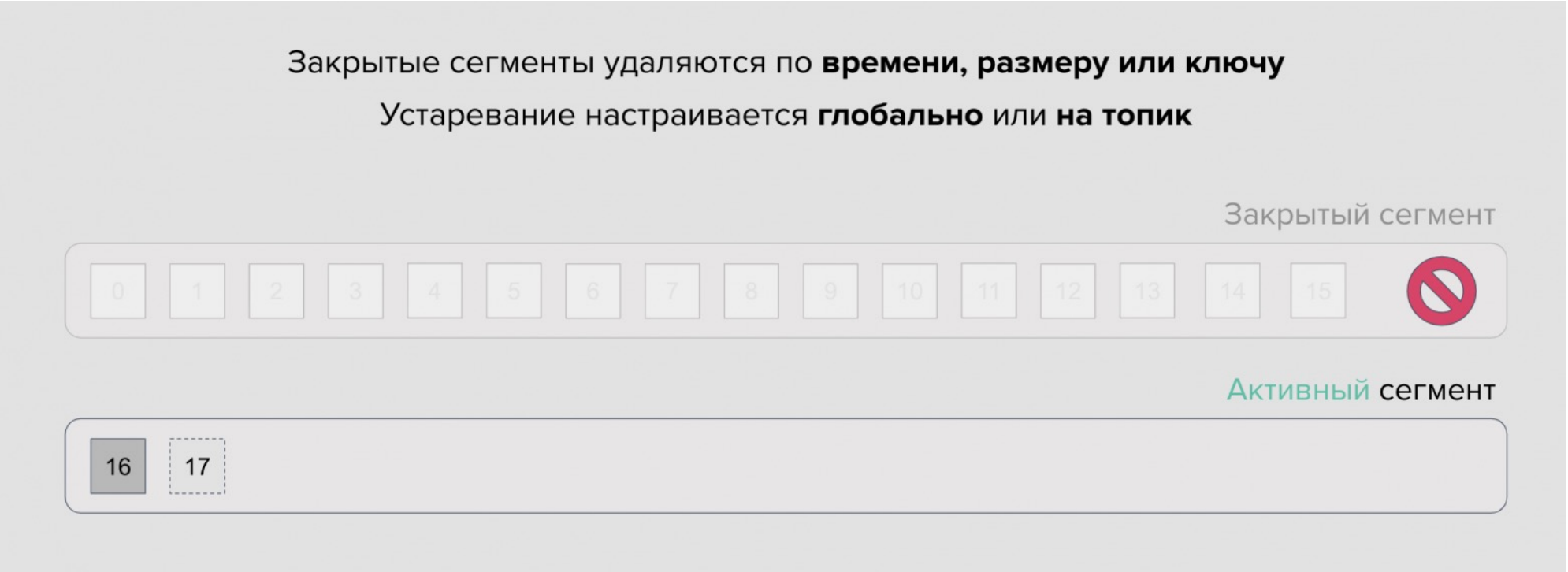
Расстояние между конечным оффсетом и текущим оффсетом консьюмера называют ***лагом***

Сообщение Kafka

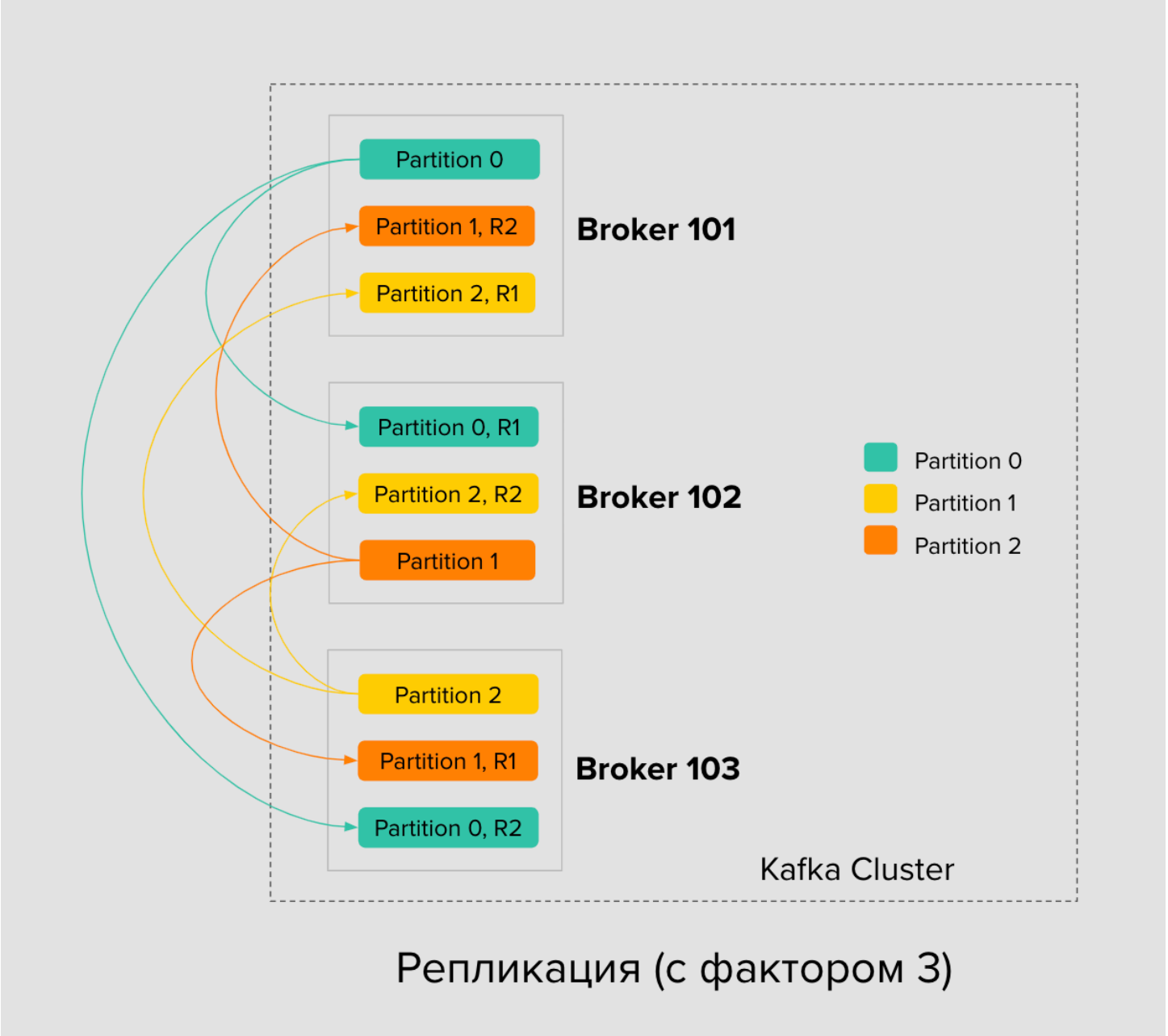


Устаревание данных

Когда сегмент достигает своего предела, он закрывается и вместо него открывается новый. Сегмент, в который сейчас записываются данные, называют ***активным сегментом*** — по сути это файл, открытый процессом брокеры. ***Закрытыми*** же называются те сегменты, в которые больше нет записи.

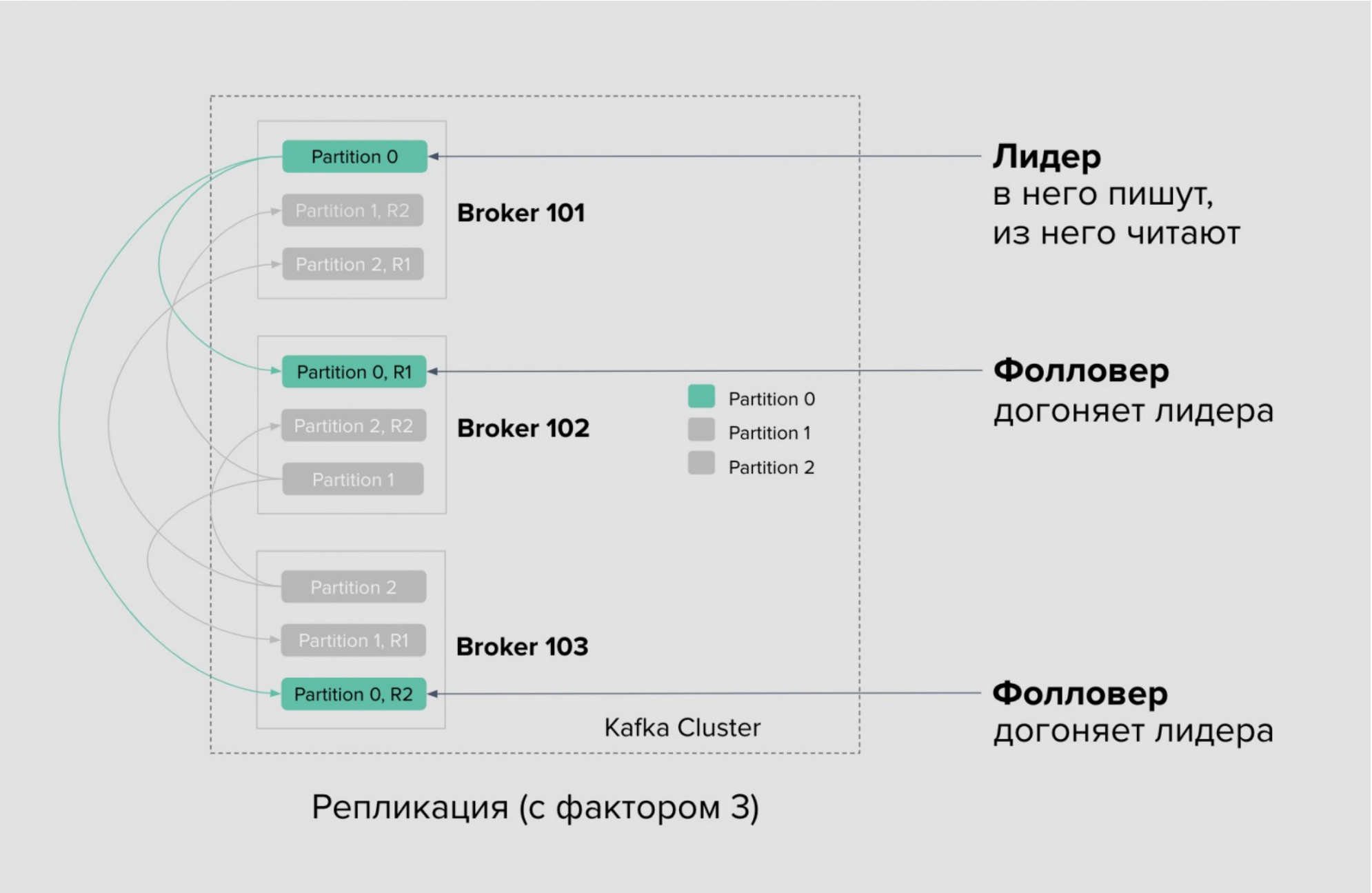


Репликация данных

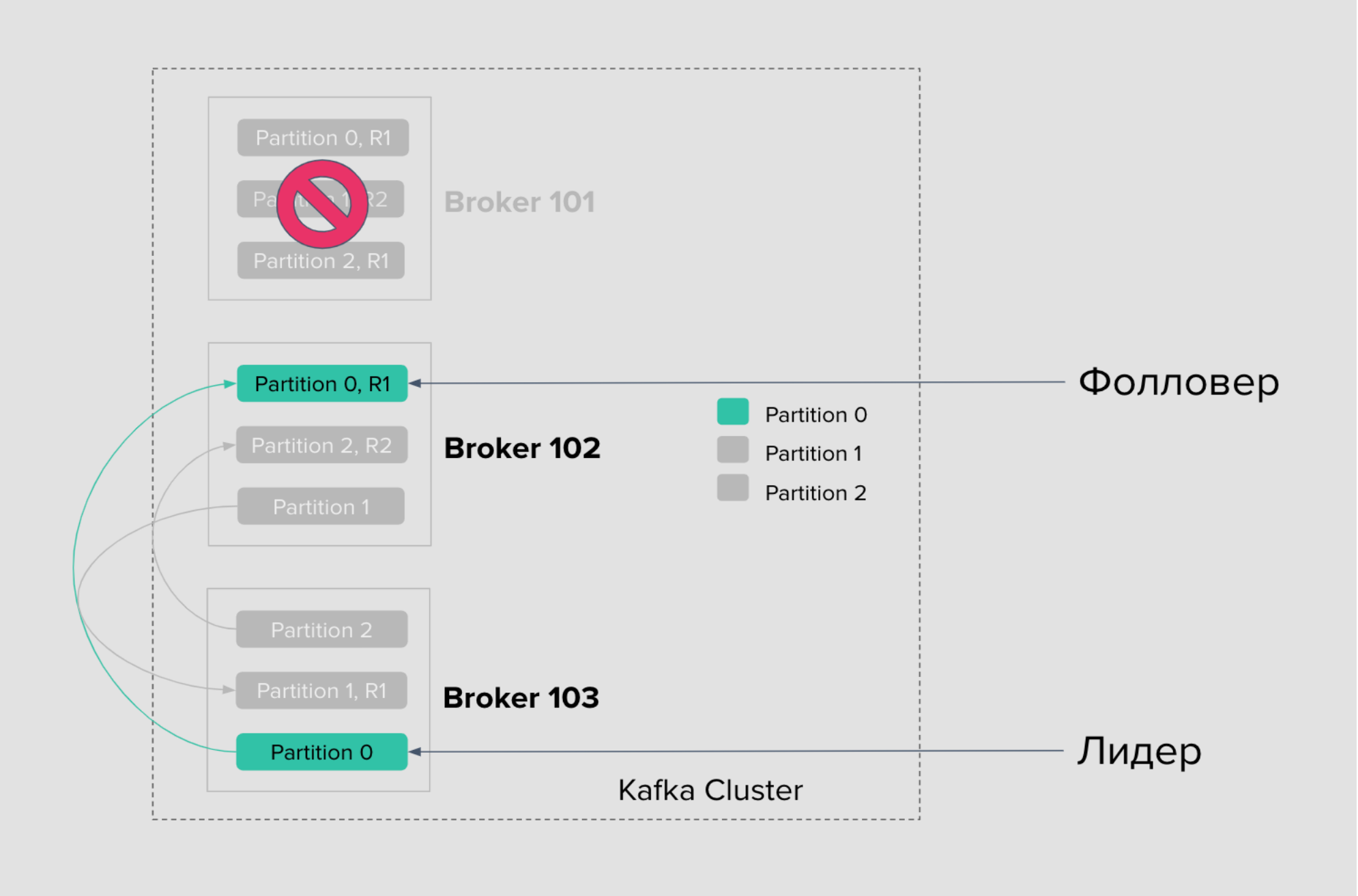


Репликация (с фактором 3)

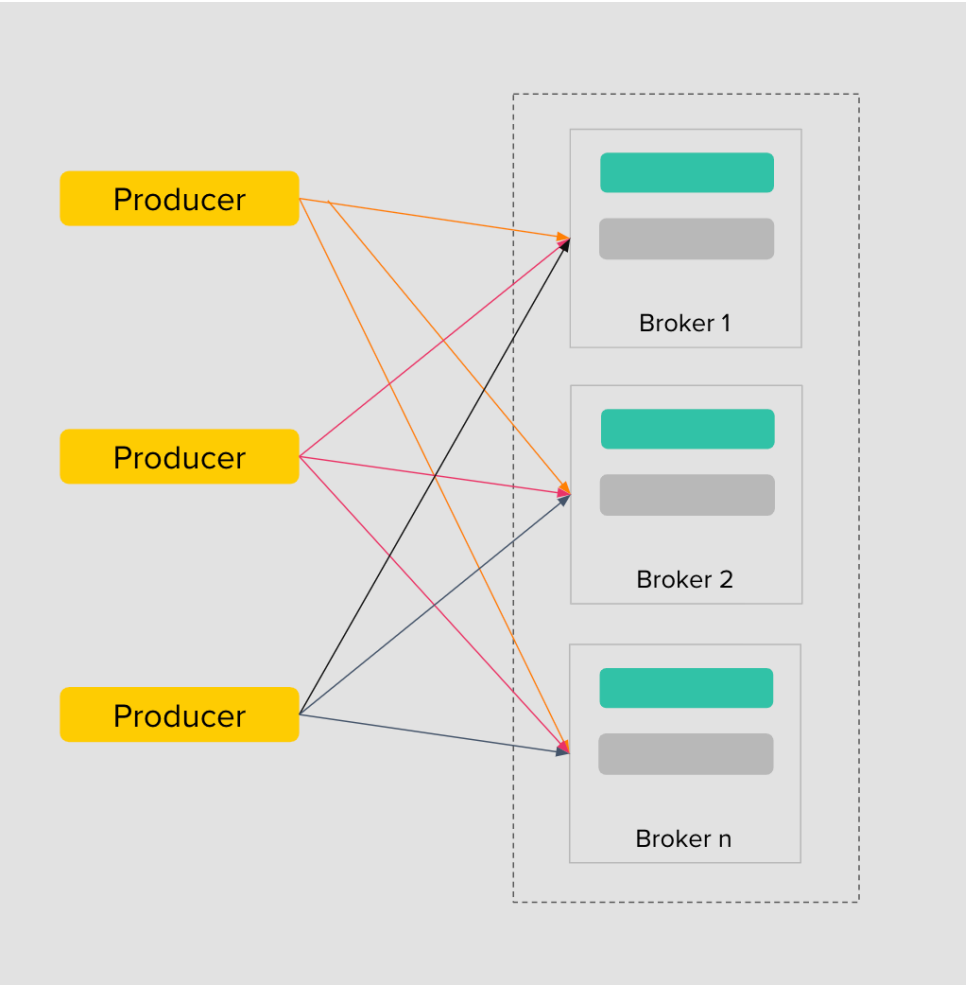
Репликация данных



Репликация данных



Балансировка и секционирование

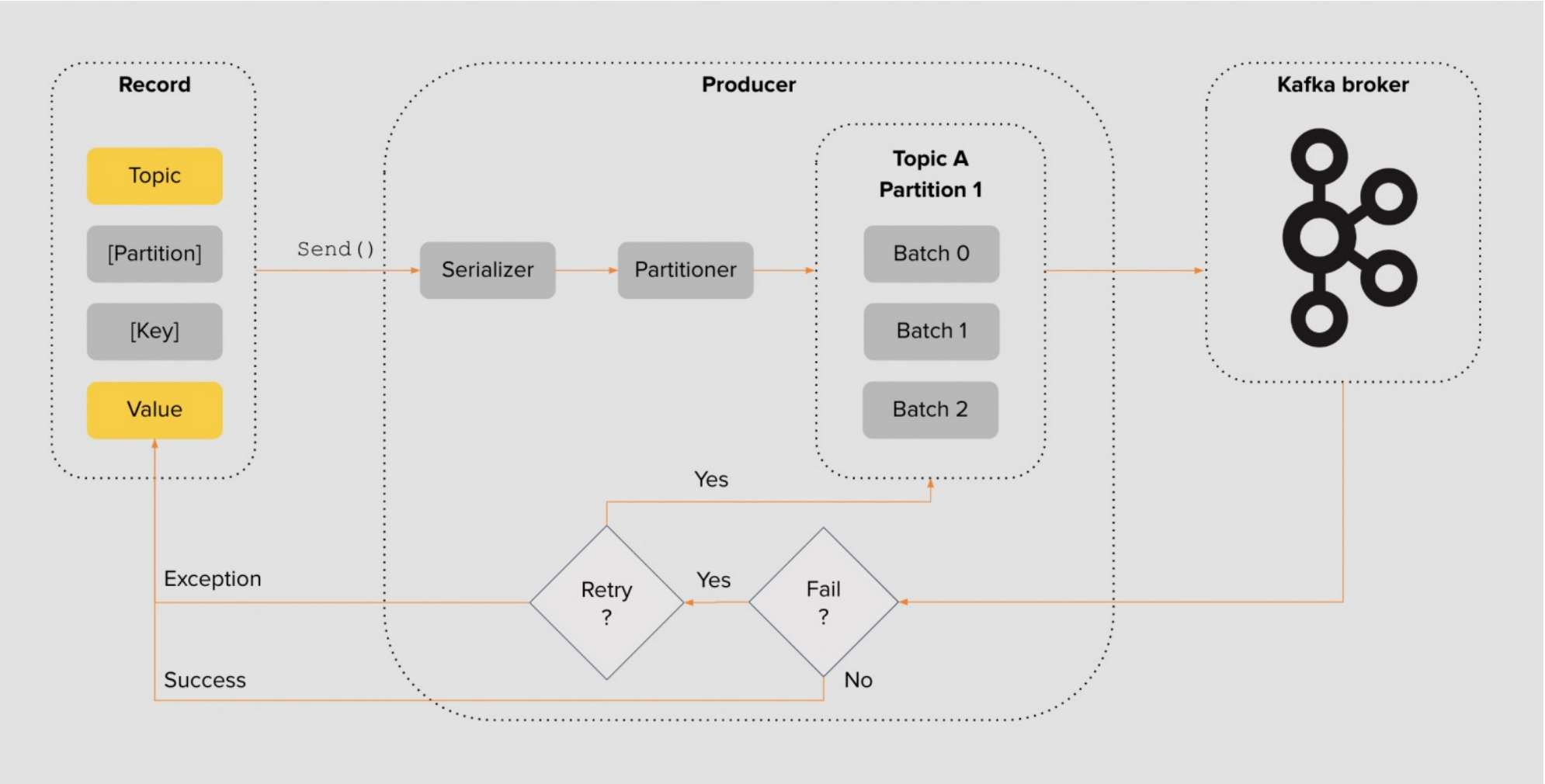


Стратегии секционирования:

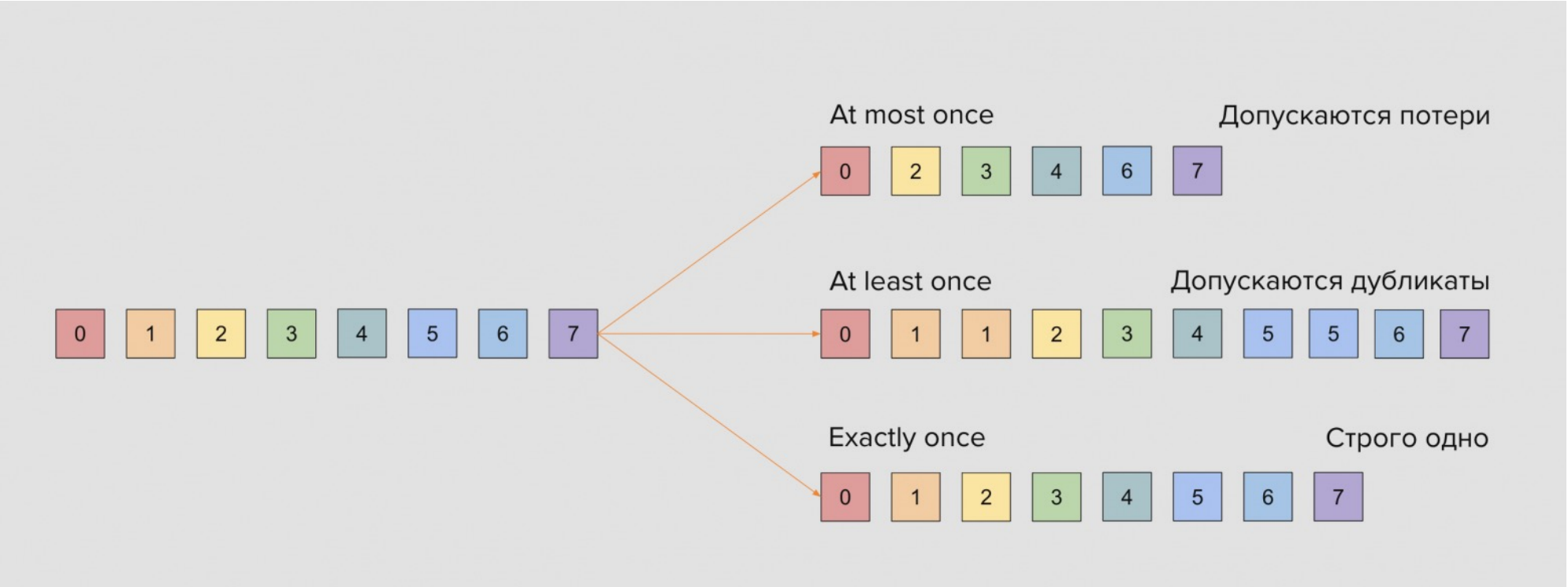
- По ключу (сообщения с одним ключом попадают в один раздел)
- По очереди (не указан ключ) стратегия секционирования по умолчанию называется round-robin — сообщения будут попадать в разделы по очереди
- Кастомная (реализуется продюсером)
- List-based partitioning,
- Composite partitioning,
- Range-based partitioning

Вся логика реализации секционирования данных реализуется на стороне продюсера.

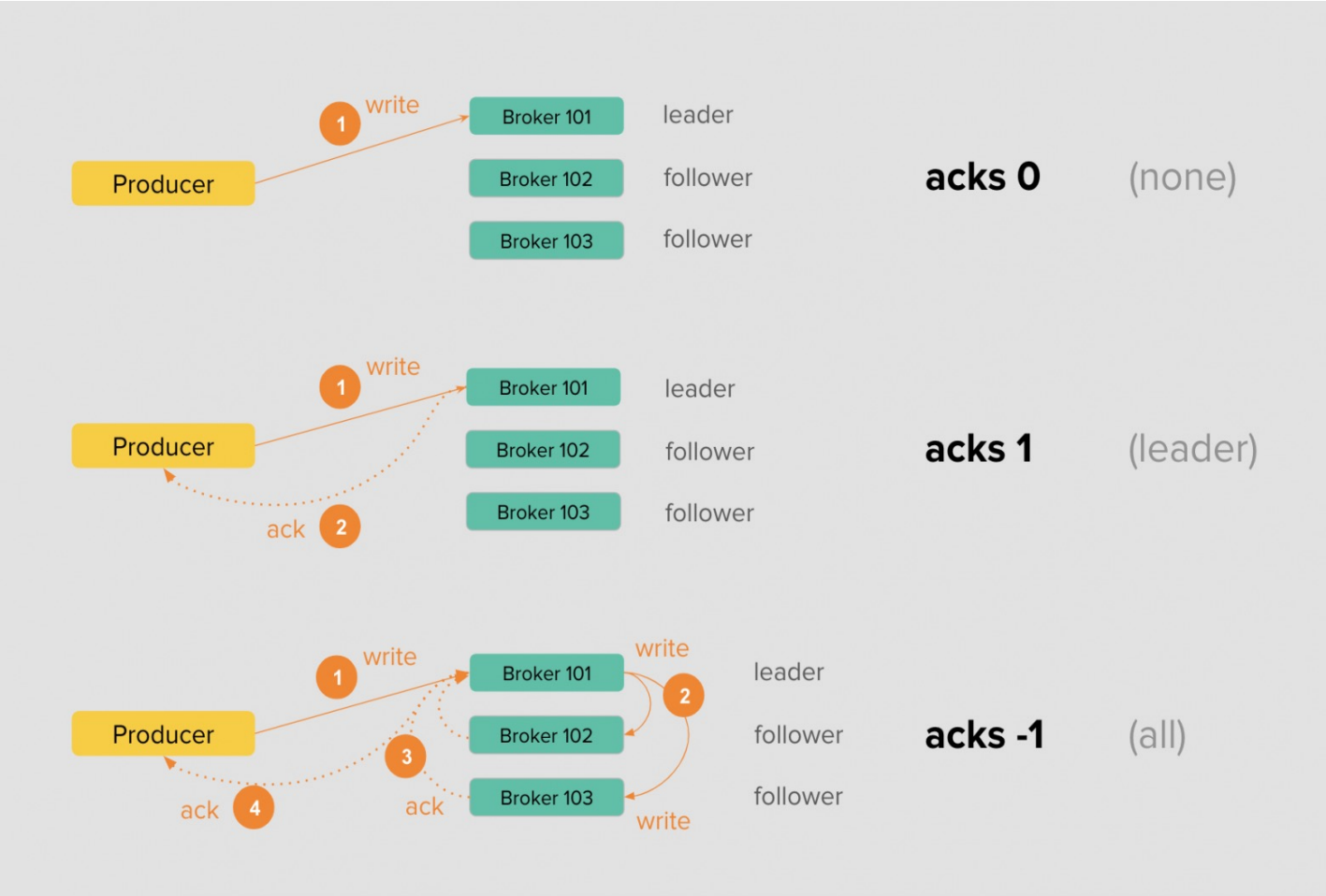
Дизайн продюсера



Семантики доставки



Надёжность доставки



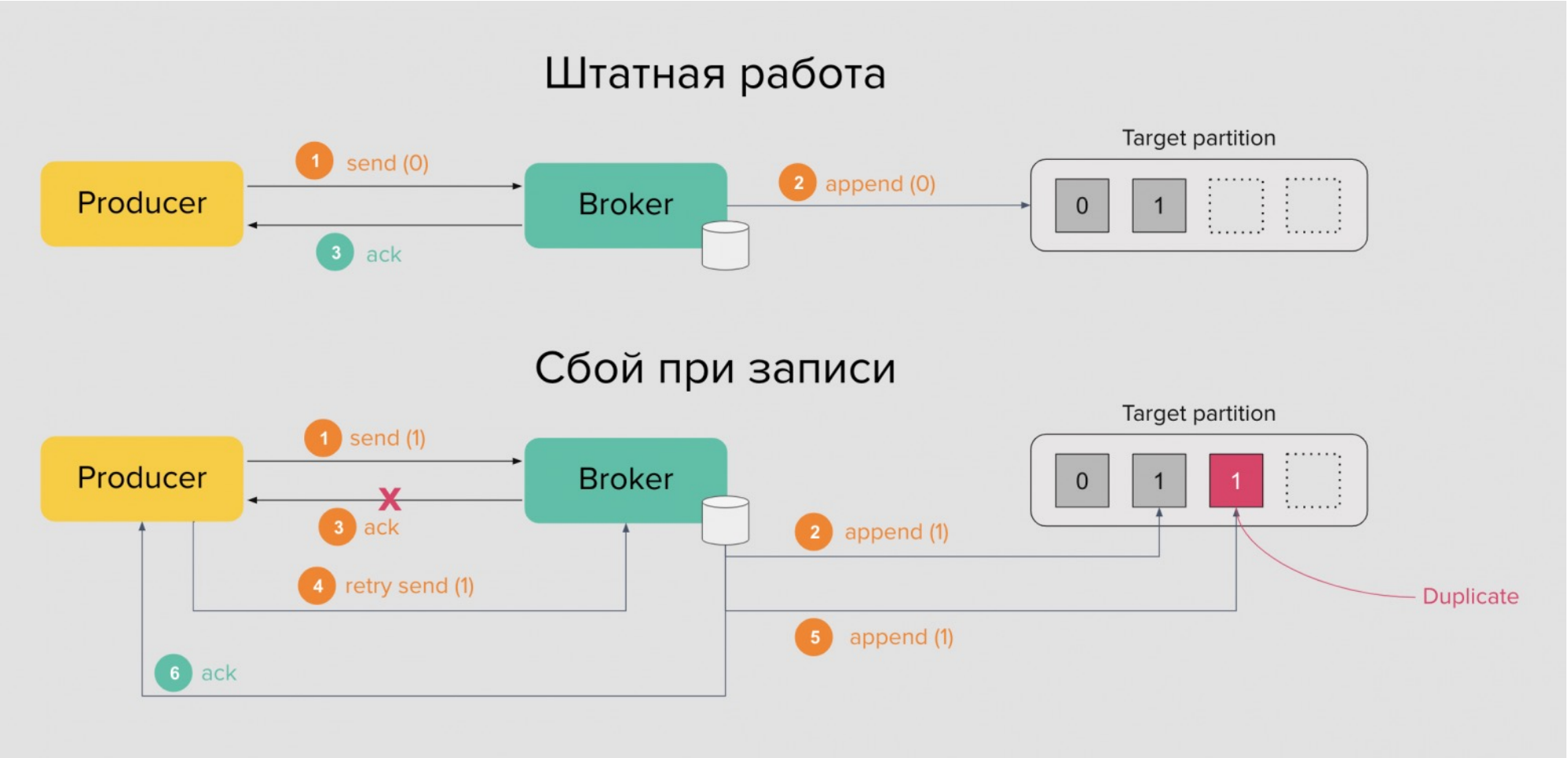
Сообщение сохранено на диск одного брокера

Число реплик устанавливает настройка ***min.insync.replicas***.

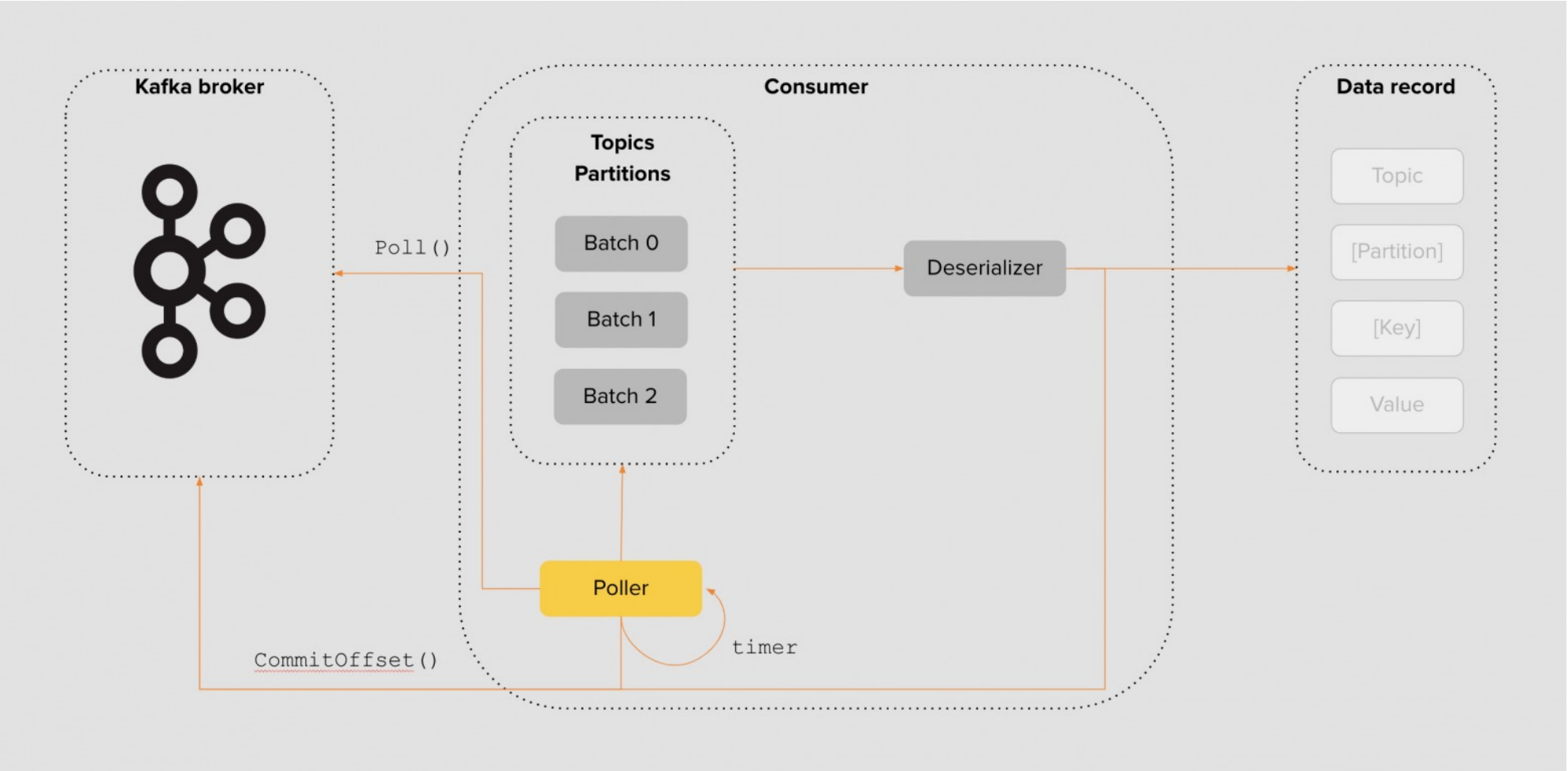
asks leader: Если в момент подтверждения брокер с лидерской партицией выходит из строя, а фолловеры не успели отреплицировать с него данные, то вы теряете сообщение и не узнаёте об этом.

asks all: выбор ***min.insync.replicas*** по числу реплик может привести к ошибке

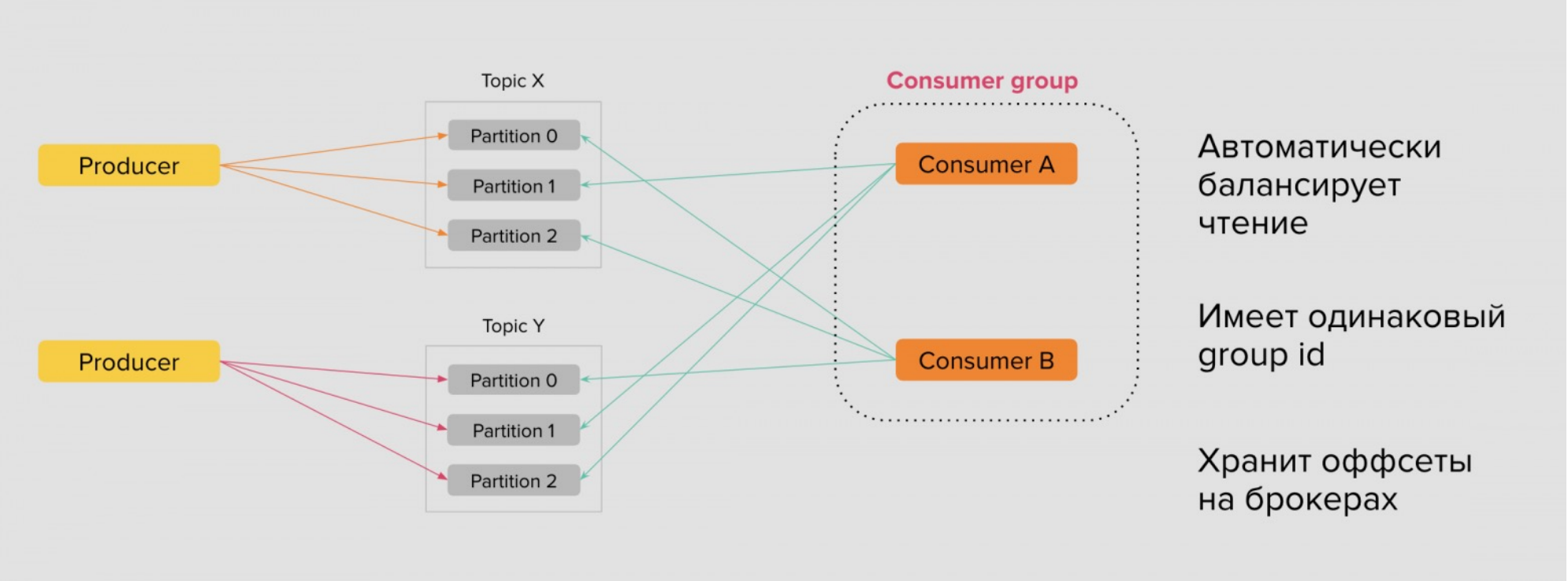
Идемпотентные продюсеры



Дизайн консьюмера

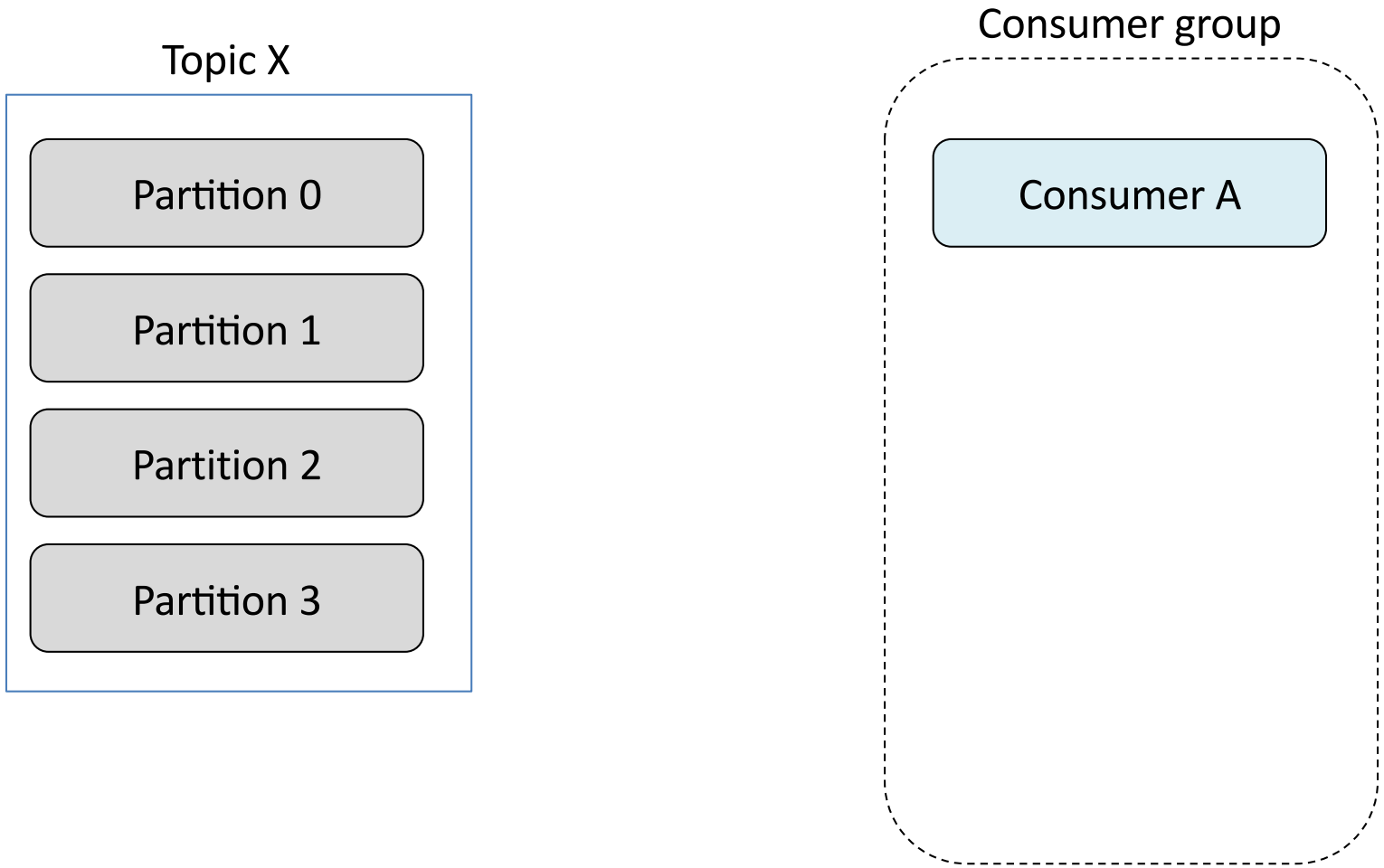


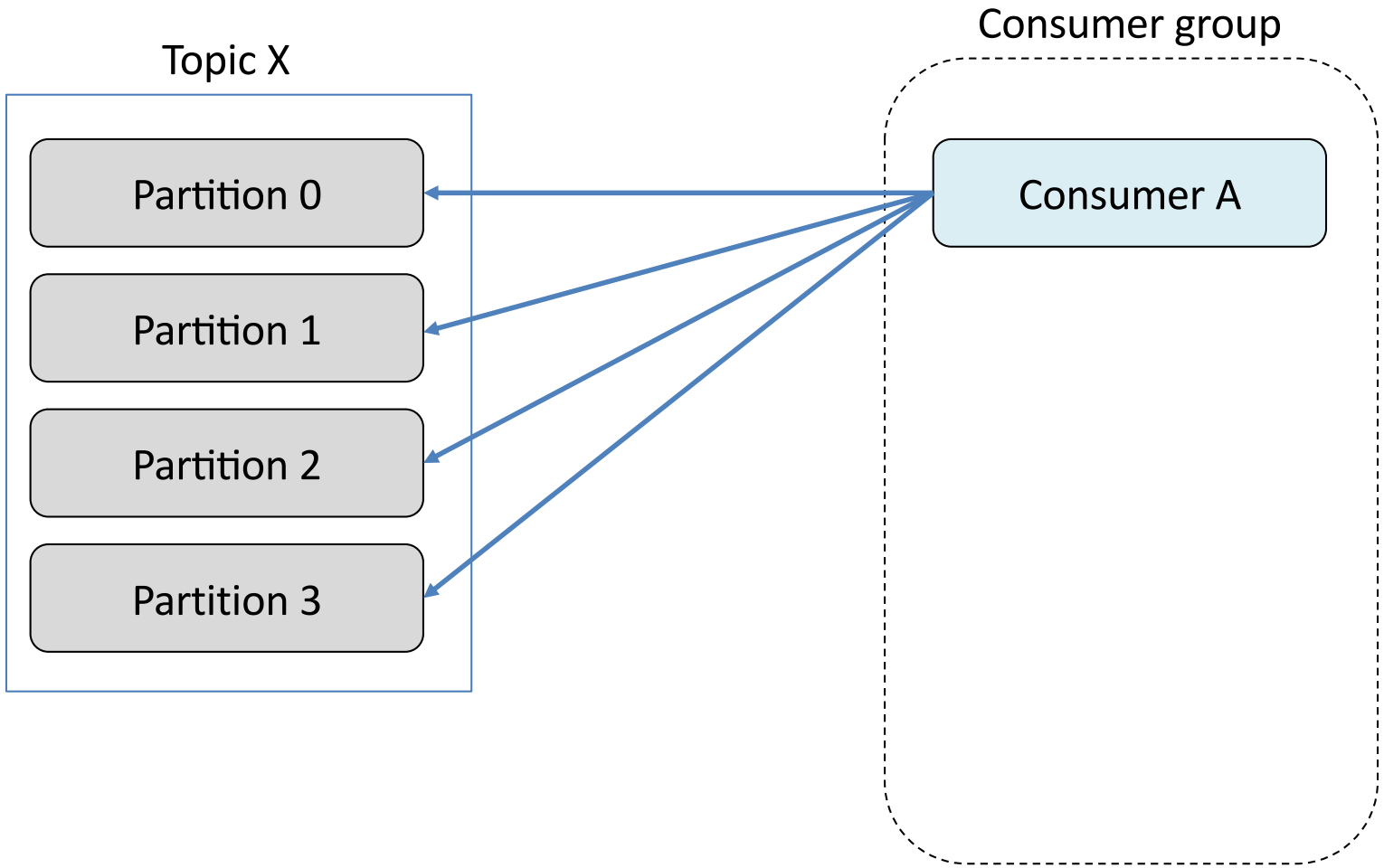
Консьюмер-группы

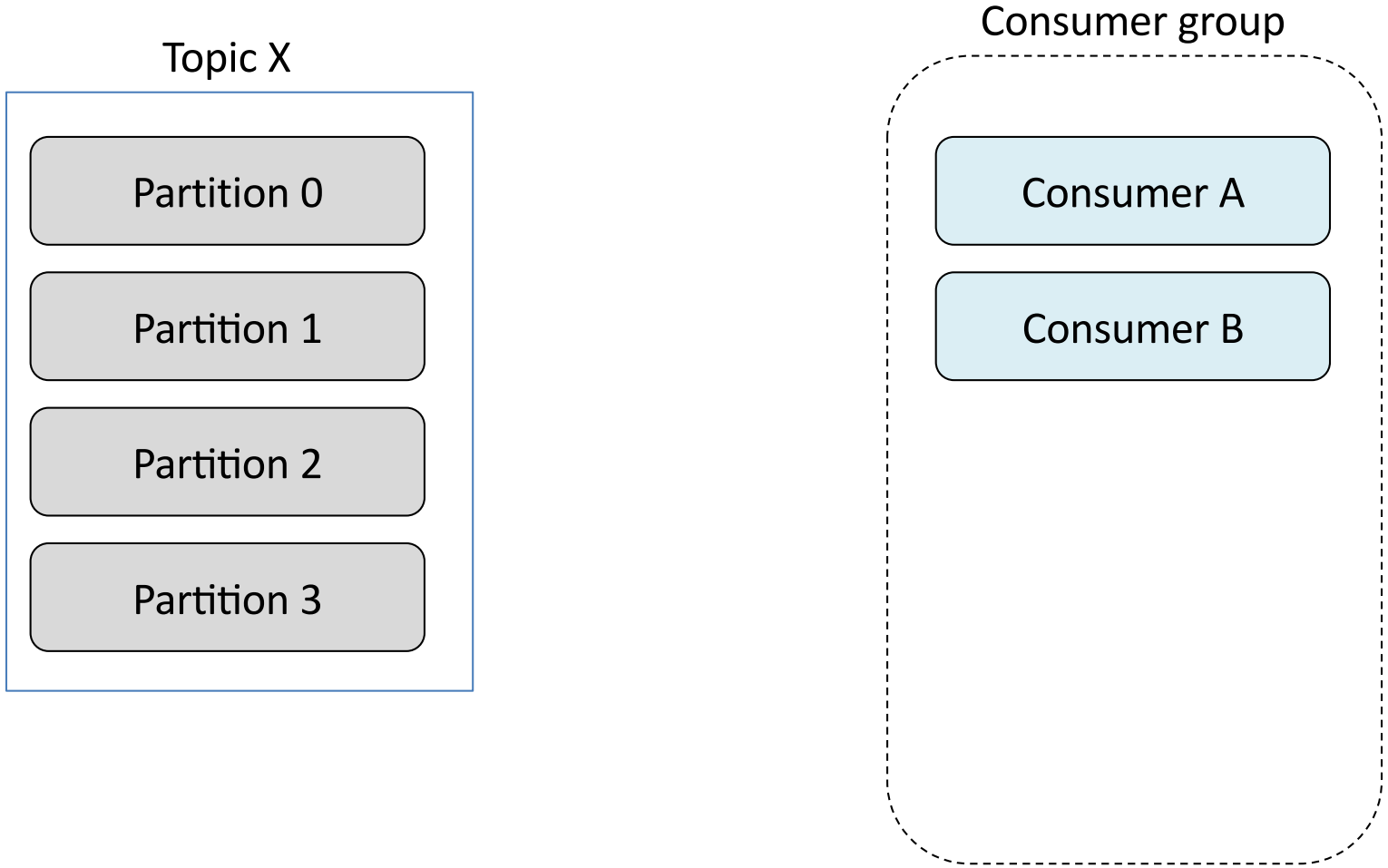


Партиции в консьюмер-группах распределяет автоматически ***Group Coordinator*** при помощи ***Group leader*** – первого участника в группе.

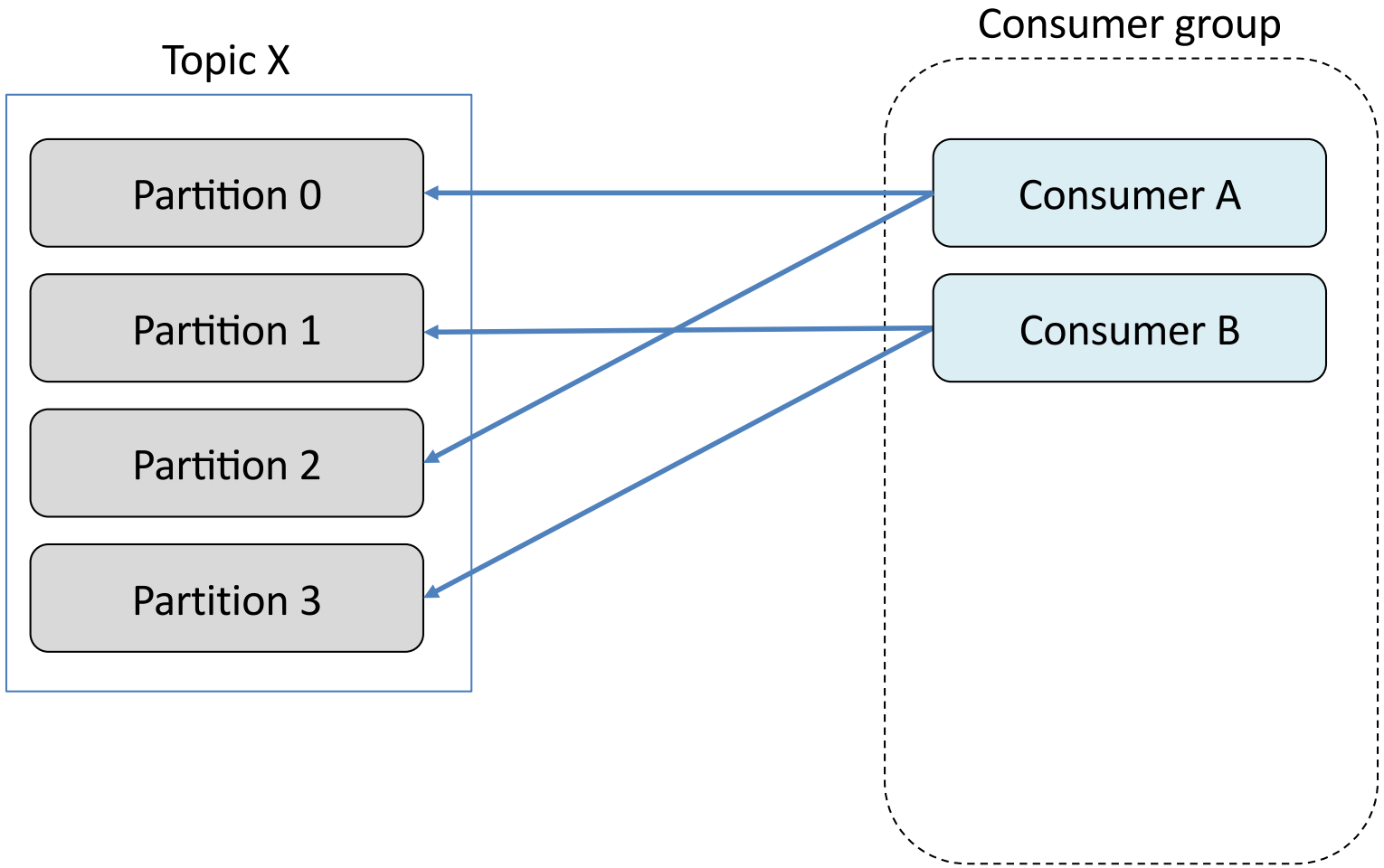
Два и более консьюмеров в группе не могут читать из одной и той же партиции.

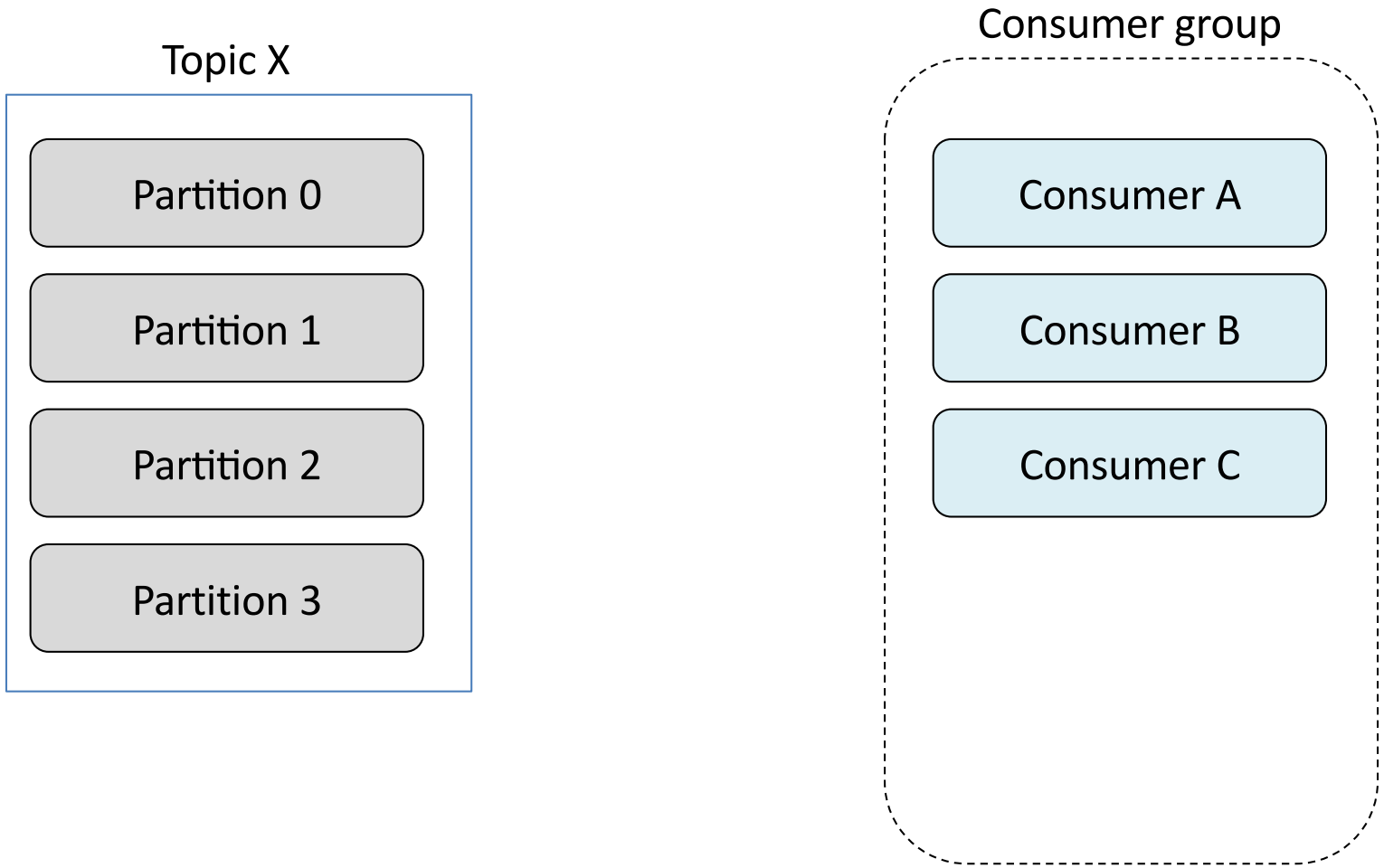

Topic X
Partition 0
Partition 1
Partition 2
Partition 3
Consumer group
Consumer A


Topic X
Partition 0
Partition 1
Partition 2
Partition 3
Consumer group
Consumer A

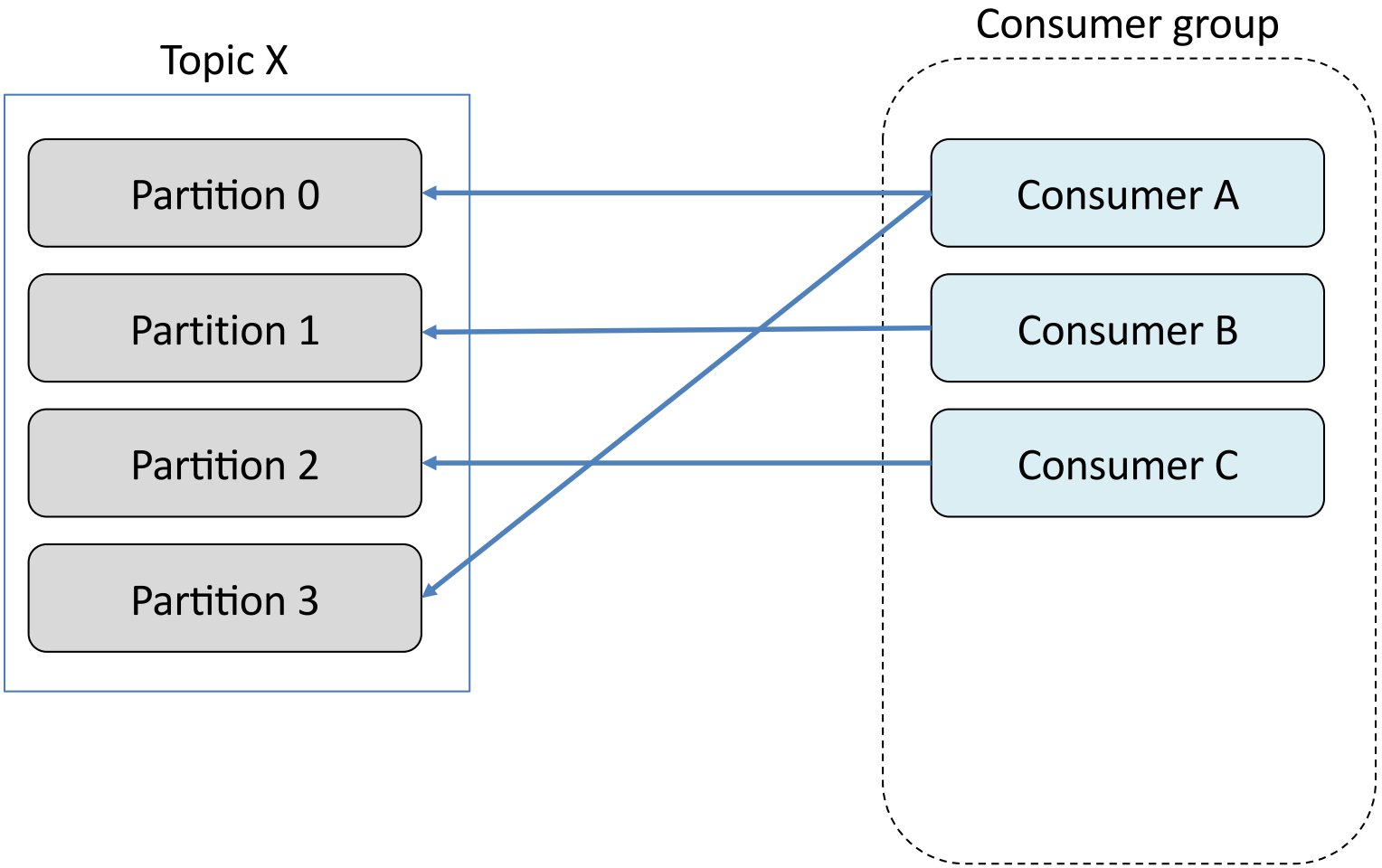


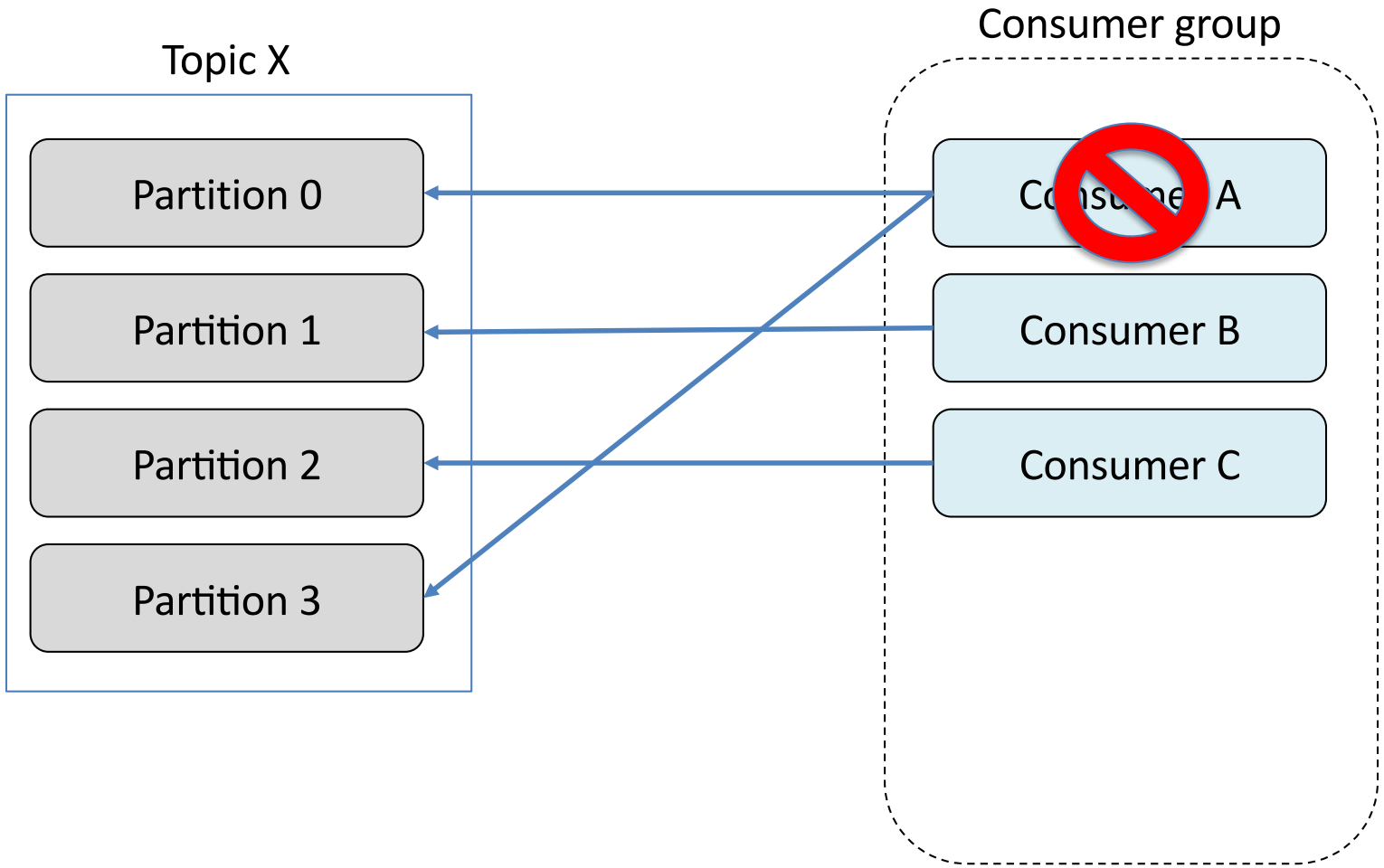
Rebalancing

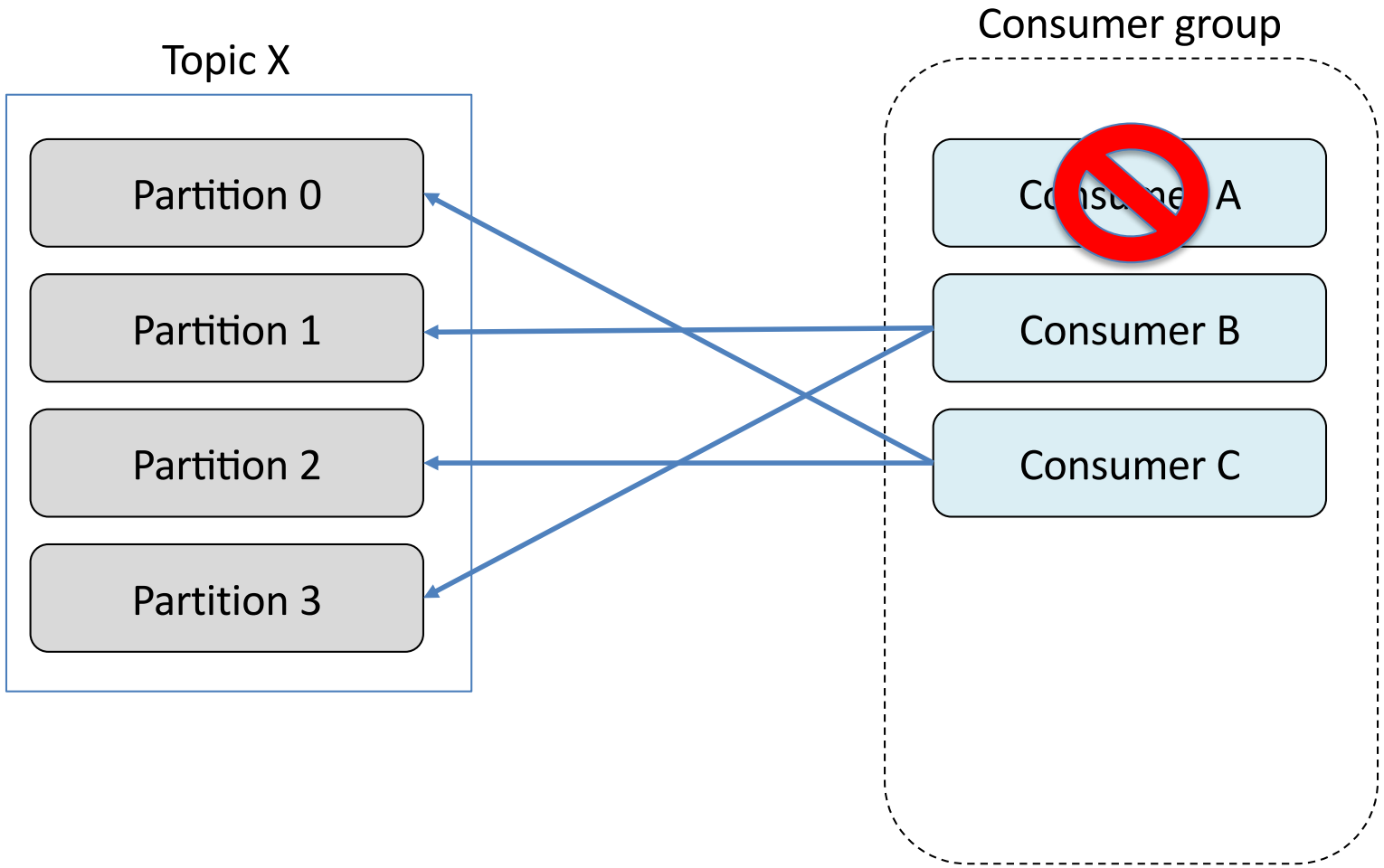

Topic X
Partition 0
Partition 1
Partition 2
Partition 3
Consumer group
Consumer A
Consumer B

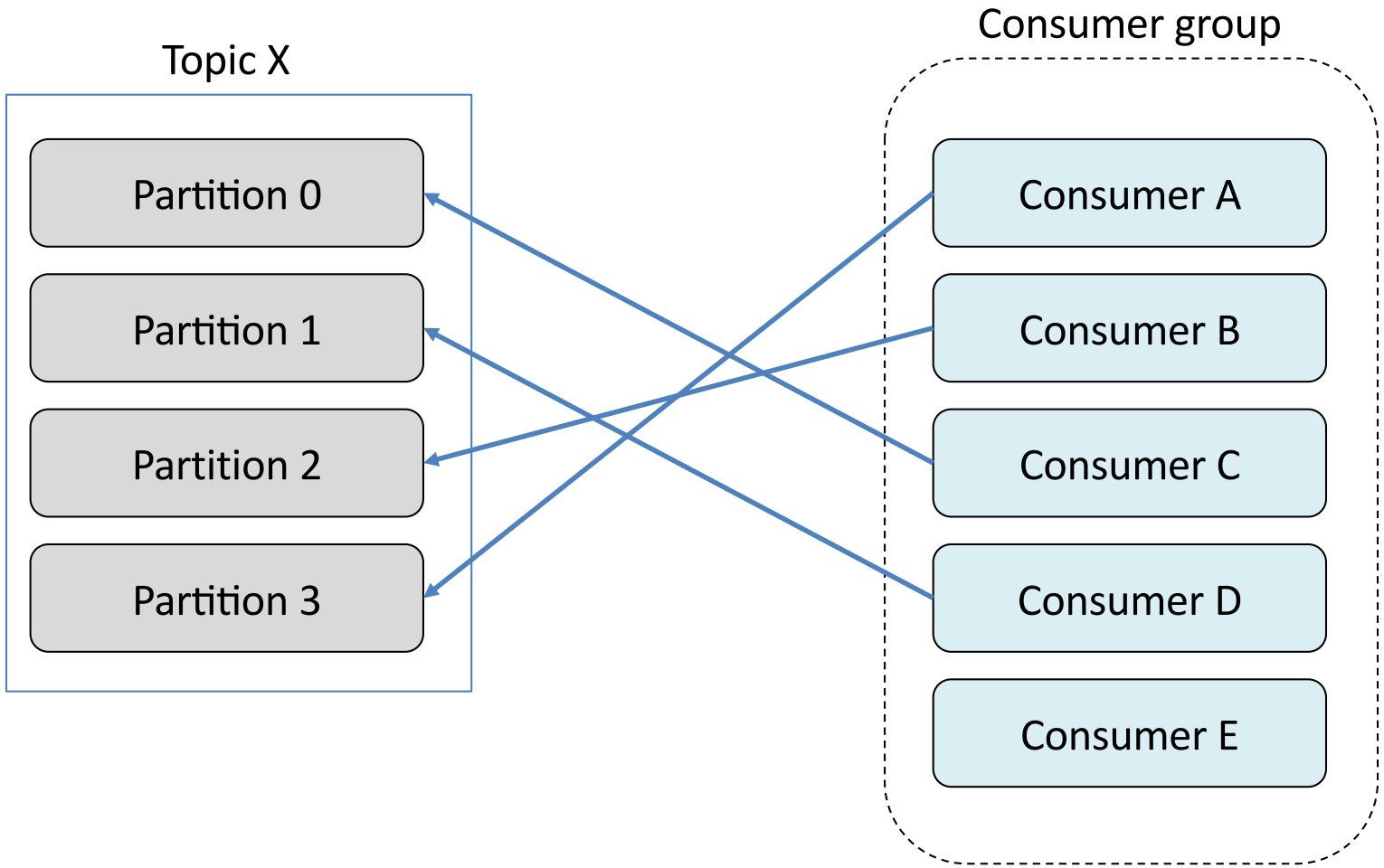


Rebalancing


Topic X
Partition 0
Partition 1
Partition 2
Partition 3
Consumer group
Consumer A
Consumer B
Consumer C


Topic X
Partition 0
Partition 1
Partition 2
Partition 3
Consumer group
Consumer A
Consumer B
Consumer C

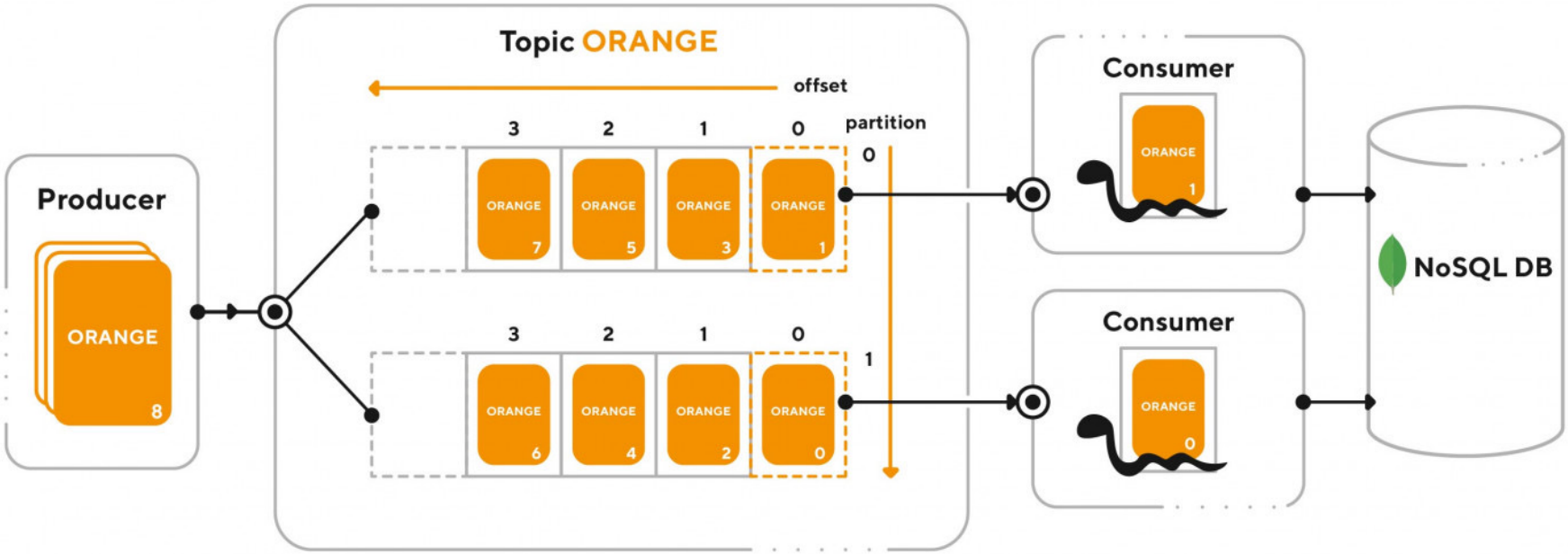

Topic X
Partition 0
Partition 1
Partition 2
Partition 3
Consumer group
Consumer A
Consumer B
Consumer C


Topic X
Partition 0
Partition 1
Partition 2
Partition 3
Consumer group
Consumer A
Consumer B
Consumer C
Consumer D
Consumer E

Обзор проблем ребалансировки разделов

при добавлении/отключении консьюмеров

https://habr.com/ru/articles/587592/



Установка Kafka и простейший сценарий работы

```
$ tar -xzf kafka_2.13-3.9.0.tgz
$ cd kafka_2.13-3.9.0
```

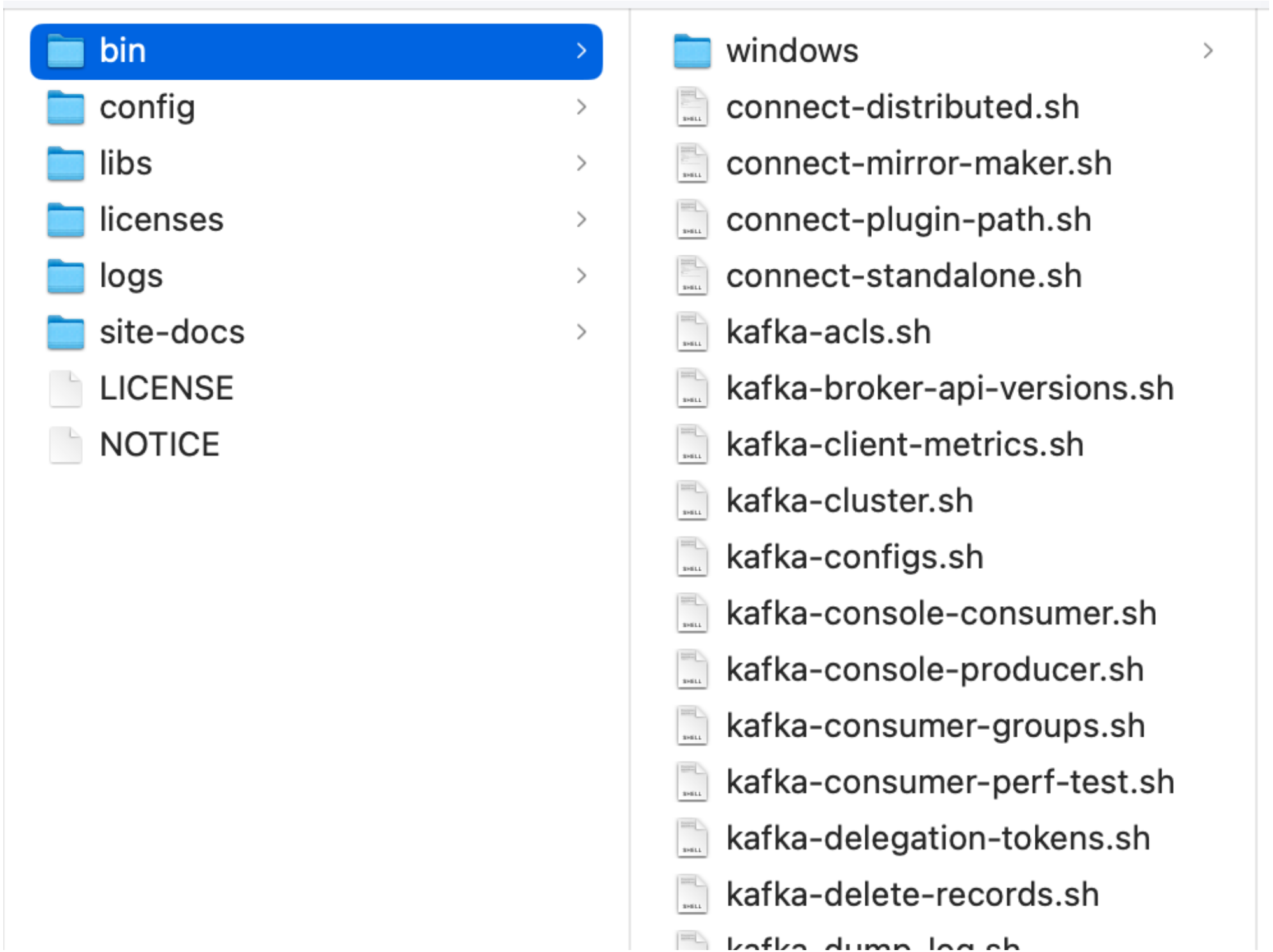
Kafka with KRaft

```
$ KAFKA_CLUSTER_ID="$(bin/kafka-storage.sh random-uuid)"

$ bin/kafka-storage.sh format --standalone -t $KAFKA_CLUSTER_ID
-c config/kraft/reconfig-server.properties

$ bin/kafka-server-start.sh config/kraft/reconfig-server.properties
```

Using JVM Based Apache Kafka Docker Image

Kafka with ZooKeeper

```
kafka_2.13-3.9.0 — java -Xmx1G -Xms1G -server -XX:+UseG1GC -XX:MaxGCPauseMillis=20 -XX:InitiatingHeapOccupancyPercent=35 -XX:+ExplicitGCInvokesConcurrent -XX:MaxInlineLevel=15 -Djava.awt.headless=true -Xloggc:/Users/sepo/kafka_2.13-3.9.0/bin/../logs/k
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % KAFKA_CLUSTER_ID="$(bin/kafka-storage.sh random-uuid)"
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-storage.sh format --standalone -t $KAFKA_CLUSTER_ID -c config/kraft/reconfig-server.properties
Formatting dynamic metadata voter directory /tmp/kraft-combined-logs with metadata.version 3.9-IV0.
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-server-start.sh config/kraft/reconfig-server.properties
[2024-12-17 11:42:00,208] INFO Registered kafka:type=kafka.Log4jController MBean (kafka.utils.Log4jControllerRegistration$)
[2024-12-17 11:42:00,490] INFO Setting -D jdk.tls.rejectClientInitiatedRenegotiation=true to disable client-initiated TLS renegotiation (org.apache.zookeeper.common.X509Util)
[2024-12-17 11:42:00,699] INFO Registered signal handlers for TERM, INT, HUP (org.apache.kafka.common.utils.LoggingSignalHandler)
[2024-12-17 11:42:00,702] INFO [ControllerServer id=1] Starting controller (kafka.server.ControllerServer)
[2024-12-17 11:42:01,204] INFO Updated connection-accept-rate max connection creation rate to 2147483647 (kafka.network.ConnectionQuotas)
[2024-12-17 11:42:01,250] INFO [SocketServer listenerType=CONTROLLER, nodeId=1] Created data-plane acceptor and processors for endpoint : ListenerName(CONTROLLER) (kafka.network.SocketServer)
[2024-12-17 11:42:01,259] INFO authorizerStart completed for endpoint CONTROLLER. Endpoint is now READY. (org.apache.kafka.server.network.EndpointReadyFutures)
[2024-12-17 11:42:01,261] INFO [SharedServer id=1] Starting SharedServer (kafka.server.SharedServer)
[2024-12-17 11:42:01,337] INFO [LogLoader partition=__cluster_metadata-0, dir=/tmp/kraft-combined-logs] Loading producer state till offset 0 with message format version 2 (kafka.log.UnifiedLog$)
[2024-12-17 11:42:01,338] INFO [LogLoader partition=__cluster_metadata-0, dir=/tmp/kraft-combined-logs] Reloading from producer snapshot and rebuilding producer state from offset 0 (kafka.log.UnifiedLog$)
[2024-12-17 11:42:01,338] INFO [LogLoader partition=__cluster_metadata-0, dir=/tmp/kraft-combined-logs] Producer state recovery took 0ms for snapshot load and 0ms for segment recovery from offset 0 (kafka.log.UnifiedLog$)
[2024-12-17 11:42:01,386] INFO Initialized snapshots with IDs SortedSet(OffsetAndEpoch(offset=0, epoch=0)) from /tmp/kraft-combined-logs/__cluster_metadata-0 (kafka.raft.KafkaMetadataLog$)
[2024-12-17 11:42:01,409] INFO [raft-expiration-reaper]: Starting (kafka.raft.TimingWheelExpirationService$ExpiredOperationReaper)
[2024-12-17 11:42:01,419] INFO [RaftManager id=1] Starting request manager with bootstrap servers: [localhost:9093 (id: -2 rack: null)] (org.apache.kafka.raft.KafkaRaftClient)
[2024-12-17 11:42:01,430] INFO [RaftManager id=1] Reading KRaft snapshot and log as part of the initialization (org.apache.kafka.raft.KafkaRaftClient)
[2024-12-17 11:42:01,433] INFO [RaftManager id=1] Loading snapshot (OffsetAndEpoch(offset=0, epoch=0)) since log start offset (0) is greater than the internal listener's next offset (-1) (org.apache.kafka.raft.internals.KRaftControlRecordStateMachine)
[2024-12-17 11:42:01,442] INFO [RaftManager id=1] Latest kraft.version is KRAFT_VERSION_1 at offset -1 (org.apache.kafka.raft.internals.KRaftControlRecordStateMachine)
[2024-12-17 11:42:01,443] INFO [RaftManager id=1] Latest set of voters is VoterSet(voters={1=VoterNode(voterKey=ReplicaKey(id=1, directoryId=Optional[4YGRG27aQS-oKouus-IJrw]), listeners=Endpoints(endpoints={ListenerName(CONTROLLER)=localhost:9093}), suppo
:0, max_version:1])}) at offset -1 (org.apache.kafka.raft.internals.KRaftControlRecordStateMachine)
[2024-12-17 11:42:01,444] INFO [RaftManager id=1] Starting voters are VoterSet(voters={1=VoterNode(voterKey=ReplicaKey(id=1, directoryId=Optional[4YGRG27aQS-oKouus-IJrw]), listeners=Endpoints(endpoints={ListenerName(CONTROLLER)=localhost:9093}), supported
max_version:1])}) (org.apache.kafka.raft.KafkaRaftClient)
[2024-12-17 11:42:01,451] INFO [RaftManager id=1] Attempting durable transition to Unattached(epoch=0, votedKey=null, voters=[1], electionTimeoutMs=1725, highWatermark=Optional.empty) from null (org.apache.kafka.raft.QuorumState)
[2024-12-17 11:42:01,624] INFO [RaftManager id=1] Completed transition to Unattached(epoch=0, votedKey=null, voters=[1], electionTimeoutMs=1725, highWatermark=Optional.empty) from null (org.apache.kafka.raft.QuorumState)
[2024-12-17 11:42:01,627] INFO [RaftManager id=1] Attempting durable transition to CandidateState(localId=1, localDirectoryId=4YGRG27aQS-oKouus-IJrw,epoch=1, retries=1, voteStates={1=org.apache.kafka.raft.CandidateState$VoterState@664e5dee}, highWatermark
tached(epoch=0, votedKey=null, voters=[1], electionTimeoutMs=1725, highWatermark=Optional.empty) (org.apache.kafka.raft.QuorumState)
[2024-12-17 11:42:01,628] INFO [RaftManager id=1] Completed transition to CandidateState(localId=1, localDirectoryId=4YGRG27aQS-oKouus-IJrw,epoch=1, retries=1, voteStates={1=org.apache.kafka.raft.CandidateState$VoterState@664e5dee}, highWatermark=Optional
och=0, votedKey=null, voters=[1], electionTimeoutMs=1725, highWatermark=Optional.empty) (org.apache.kafka.raft.QuorumState)
[2024-12-17 11:42:01,635] INFO [RaftManager id=1] Attempting durable transition to Leader(localReplicaKey=ReplicaKey(id=1, directoryId=Optional[4YGRG27aQS-oKouus-IJrw]), epoch=1, epochStartOffset=0, highWatermark=Optional.empty, voterStates={1=ReplicaStat
[4YGRG27aQS-oKouus-IJrw]), endOffset=Optional.empty, lastFetchTimestamp=-1, lastCaughtUpTimestamp=-1, hasAcknowledgedLeader=true}}) from CandidateState(localId=1, localDirectoryId=4YGRG27aQS-oKouus-IJrw,epoch=1, retries=1, voteStates={1=org.apache.kafka.r
mark=Optional.empty, electionTimeoutMs=1874) (org.apache.kafka.raft.QuorumState)
[2024-12-17 11:42:01,636] INFO [RaftManager id=1] Completed transition to Leader(localReplicaKey=ReplicaKey(id=1, directoryId=Optional[4YGRG27aQS-oKouus-IJrw]), epoch=1, epochStartOffset=0, highWatermark=Optional.empty, voterStates={1=ReplicaState(replica
QS-oKouus-IJrw]), endOffset=Optional.empty, lastFetchTimestamp=-1, lastCaughtUpTimestamp=-1, hasAcknowledgedLeader=true}}) from CandidateState(localId=1, localDirectoryId=4YGRG27aQS-oKouus-IJrw,epoch=1, retries=1, voteStates={1=org.apache.kafka.raft.Candi
onal.empty, electionTimeoutMs=1874) (org.apache.kafka.raft.QuorumState)
[2024-12-17 11:42:01,651] INFO [kafka-1-raft-outbound-request-thread]: Starting (org.apache.kafka.raft.KafkaNetworkChannel$SendThread)
[2024-12-17 11:42:01,652] INFO [kafka-1-raft-io-thread]: Starting (org.apache.kafka.raft.KafkaRaftClientDriver)
[2024-12-17 11:42:01,676] INFO [MetadataLoader id=1] initializeNewPublishers: the loader is still catching up because we still don't know the high water mark yet. (org.apache.kafka.image.loader.MetadataLoader)
[2024-12-17 11:42:01,676] INFO [RaftManager id=1] Latest kraft.version is KRAFT_VERSION_1 at offset 1 (org.apache.kafka.raft.internals.KRaftControlRecordStateMachine)
[2024-12-17 11:42:01,677] INFO [RaftManager id=1] Latest set of voters is VoterSet(voters={1=VoterNode(voterKey=ReplicaKey(id=1, directoryId=Optional[4YGRG27aQS-oKouus-IJrw]), listeners=Endpoints(endpoints={ListenerName(CONTROLLER)=localhost:9093}), suppo
:0, max_version:1])}) at offset 2 (org.apache.kafka.raft.internals.KRaftControlRecordStateMachine)
[2024-12-17 11:42:01,678] INFO [ControllerServer id=1] Waiting for controller quorum voters future (kafka.server.ControllerServer)
[2024-12-17 11:42:01,678] INFO [ControllerServer id=1] Finished waiting for controller quorum voters future (kafka.server.ControllerServer)
[2024-12-17 11:42:01,680] INFO [RaftManager id=1] High watermark set to LogOffsetMetadata(offset=3, metadata=Optional[(segmentBaseOffset=0,relativePositionInSegment=174)]) for the first time for epoch 1 based on indexOfHw 0 and voters [ReplicaState(replic
aQS-oKouus-IJrw]), endOffset=Optional[LogOffsetMetadata(offset=3, metadata=Optional[(segmentBaseOffset=0,relativePositionInSegment=174)])], lastFetchTimestamp=-1, lastCaughtUpTimestamp=-1, hasAcknowledgedLeader=true)] (org.apache.kafka.raft.LeaderState)
[2024-12-17 11:42:01,688] INFO [RaftManager id=1] Registered the listener org.apache.kafka.image.loader.MetadataLoader@1799695096 (org.apache.kafka.raft.KafkaRaftClient)
[2024-12-17 11:42:01,689] INFO [MetadataLoader id=1] handleLoadSnapshot(00000000000000000000-0000000000): incrementing HandleLoadSnapshotCount to 1. (org.apache.kafka.image.loader.MetadataLoader)
[2024-12-17 11:42:01,694] INFO [MetadataLoader id=1] handleLoadSnapshot(00000000000000000000-0000000000): generated a metadata delta between offset -1 and this snapshot in 5128 us. (org.apache.kafka.image.loader.MetadataLoader)
[2024-12-17 11:42:01,695] INFO [MetadataLoader id=1] maybePublishMetadata(SNAPSHOT): The loader is still catching up because we have loaded up to offset -1, but the high water mark is 3 (org.apache.kafka.image.loader.MetadataLoader)
[2024-12-17 11:42:01,696] INFO [MetadataLoader id=1] maybePublishMetadata(LOG_DELTA): The loader finished catching up to the current high water mark of 3 (org.apache.kafka.image.loader.MetadataLoader)
[2024-12-17 11:42:01,697] INFO [MetadataLoader id=1] InitializeNewPublishers: initializing SnapshotGenerator with a snapshot at offset 2 (org.apache.kafka.image.loader.MetadataLoader)
[2024-12-17 11:42:01,727] INFO [RaftManager id=1] Registered the listener org.apache.kafka.controller.QuorumController$QuorumMetaLogListener@1422035566 (org.apache.kafka.raft.KafkaRaftClient)
[2024-12-17 11:42:01,736] INFO [controller-1-ThrottledChannelReaper-Fetch]: Starting (kafka.server.ClientQuotaManager$ThrottledChannelReaper)
[2024-12-17 11:42:01,737] INFO [controller-1-ThrottledChannelReaper-Produce]: Starting (kafka.server.ClientQuotaManager$ThrottledChannelReaper)
[2024-12-17 11:42:01,738] INFO [controller-1-ThrottledChannelReaper-Request]: Starting (kafka.server.ClientQuotaManager$ThrottledChannelReaper)
[2024-12-17 11:42:01,740] INFO [controller-1-ThrottledChannelReaper-ControllerMutation]: Starting (kafka.server.ClientQuotaManager$ThrottledChannelReaper)
[2024-12-17 11:42:01,766] INFO [ExpirationReaper-1-AlterAcls]: Starting (kafka.server.DelayedOperationPurgatory$ExpiredOperationReaper)
[2024-12-17 11:42:01,782] INFO [ControllerServer id=1] Waiting for the controller metadata publishers to be installed (kafka.server.ControllerServer)
[2024-12-17 11:42:01,782] INFO [ControllerServer id=1] Finished waiting for the controller metadata publishers to be installed (kafka.server.ControllerServer)
[2024-12-17 11:42:01,782] INFO [MetadataLoader id=1] InitializeNewPublishers: initializing KRaftMetadataCachePublisher with a snapshot at offset 6 (org.apache.kafka.image.loader.MetadataLoader)
[2024-12-17 11:42:01,782] INFO [MetadataLoader id=1] InitializeNewPublishers: initializing FeaturesPublisher with a snapshot at offset 6 (org.apache.kafka.image.loader.MetadataLoader)
[2024-12-17 11:42:01,782] INFO [SocketServer listenerType=CONTROLLER, nodeId=1] Enabling request processing. (kafka.network.SocketServer)
[2024-12-17 11:42:01,782] INFO [ControllerServer id=1] Loaded new metadata Features(metadataVersion=3.9-IV0, finalizedFeatures={metadata.version=21}, finalizedFeaturesEpoch=6). (org.apache.kafka.metadata.publisher.FeaturesPublisher)
[2024-12-17 11:42:01,782] INFO [MetadataLoader id=1] InitializeNewPublishers: initializing ControllerRegistrationsPublisher with a snapshot at offset 6 (org.apache.kafka.image.loader.MetadataLoader)
[2024-12-17 11:42:01,783] INFO [MetadataLoader id=1] InitializeNewPublishers: initializing ControllerRegistrationManager with a snapshot at offset 6 (org.apache.kafka.image.loader.MetadataLoader)
[2024-12-17 11:42:01,783] INFO [MetadataLoader id=1] InitializeNewPublishers: initializing DynamicConfigPublisher controller id=1 with a snapshot at offset 6 (org.apache.kafka.image.loader.MetadataLoader)
[2024-12-17 11:42:01,784] INFO [MetadataLoader id=1] InitializeNewPublishers: initializing DynamicClientQuotaPublisher controller id=1 with a snapshot at offset 6 (org.apache.kafka.image.loader.MetadataLoader)
[2024-12-17 11:42:01,786] INFO [MetadataLoader id=1] InitializeNewPublishers: initializing ScramPublisher controller id=1 with a snapshot at offset 6 (org.apache.kafka.image.loader.MetadataLoader)
[2024-12-17 11:42:01,788] INFO [MetadataLoader id=1] InitializeNewPublishers: initializing DelegationTokenPublisher controller id=1 with a snapshot at offset 6 (org.apache.kafka.image.loader.MetadataLoader)
[2024-12-17 11:42:01,790] INFO [MetadataLoader id=1] InitializeNewPublishers: initializing ControllerMetadataMetricsPublisher with a snapshot at offset 6 (org.apache.kafka.image.loader.MetadataLoader)
[2024-12-17 11:42:01,791] INFO [MetadataLoader id=1] InitializeNewPublishers: initializing AclPublisher controller id=1 with a snapshot at offset 6 (org.apache.kafka.image.loader.MetadataLoader)
[2024-12-17 11:42:01,796] INFO Awaiting socket connections on 0.0.0.0:9093. (kafka.network.DataPlaneAcceptor)
[2024-12-17 11:42:01,809] INFO [controller-1-to-controller-registration-channel-manager]: Starting (kafka.server.NodeToControllerRequestThread)
[2024-12-17 11:42:01,810] INFO [ControllerServer id=1] Waiting for all of the authorizer futures to be completed (kafka.server.ControllerServer)
[2024-12-17 11:42:01,810] INFO [ControllerServer id=1] Finished waiting for all of the authorizer futures to be completed (kafka.server.ControllerServer)
[2024-12-17 11:42:01,810] INFO [ControllerServer id=1] Waiting for all of the SocketServer Acceptors to be started (kafka.server.ControllerServer)
[2024-12-17 11:42:01,810] INFO [ControllerServer id=1] Finished waiting for all of the SocketServer Acceptors to be started (kafka.server.ControllerServer)
[2024-12-17 11:42:01,810] INFO [ControllerRegistrationManager id=1 incarnation=26NTWptBQiayjRc1azOK1Q] initialized channel manager. (kafka.server.ControllerRegistrationManager)
[2024-12-17 11:42:01,810] INFO [controller-1-to-controller-registration-channel-manager]: Recorded new KRaft controller, from now on will use node localhost:9093 (id: 1 rack: null) (kafka.server.NodeToControllerRequestThread)
[2024-12-17 11:42:01,811] INFO [BrokerServer id=1] Transition from SHUTDOWN to STARTING (kafka.server.BrokerServer)
[2024-12-17 11:42:01,812] INFO [BrokerServer id=1] Starting broker (kafka.server.BrokerServer)
[2024-12-17 11:42:01,812] INFO [ControllerRegistrationManager id=1 incarnation=26NTWptBQiayjRc1azOK1Q] sendControllerRegistration: attempting to send ControllerRegistrationRequestData(controllerId=1, incarnationId=26NTWptBQiayjRc1azOK1Q, zkMigrationReady=
='localhost', port=9093, securityProtocol=0)], features=[Feature(name='kraft.version', minSupportedVersion=0, maxSupportedVersion=1), Feature(name='metadata.version', minSupportedVersion=1, maxSupportedVersion=21)]) (kafka.server.ControllerRegistrationMar
[2024-12-17 11:42:01,819] INFO [broker-1-ThrottledChannelReaper-Fetch]: Starting (kafka.server.ClientQuotaManager$ThrottledChannelReaper)
[2024-12-17 11:42:01,819] INFO [broker-1-ThrottledChannelReaper-Produce]: Starting (kafka.server.ClientQuotaManager$ThrottledChannelReaper)
[2024-12-17 11:42:01,819] INFO [broker-1-ThrottledChannelReaper-Request]: Starting (kafka.server.ClientQuotaManager$ThrottledChannelReaper)
[2024-12-17 11:42:01,820] INFO [broker-1-ThrottledChannelReaper-ControllerMutation]: Starting (kafka.server.ClientQuotaManager$ThrottledChannelReaper)
[2024-12-17 11:42:01,854] INFO [BrokerServer id=1] Waiting for controller quorum voters future (kafka.server.BrokerServer)
[2024-12-17 11:42:01,854] INFO [BrokerServer id=1] Finished waiting for controller quorum voters future (kafka.server.BrokerServer)
[2024-12-17 11:42:01,856] INFO [broker-1-to-controller-forwarding-channel-manager]: Starting (kafka.server.NodeToControllerRequestThread)
[2024-12-17 11:42:01,856] INFO [broker-1-to-controller-forwarding-channel-manager]: Recorded new KRaft controller, from now on will use node localhost:9093 (id: 1 rack: null) (kafka.server.NodeToControllerRequestThread)
```


bin
config
libs
licenses
logs
site-docs
LICENSE
NOTICE
windows
connect-distributed.sh
connect-mirror-maker.sh
connect-plugin-path.sh
connect-standalone.sh
kafka-acls.sh
kafka-broker-api-versions.sh
kafka-client-metrics.sh
kafka-cluster.sh
kafka-configs.sh
kafka-console-consumer.sh
kafka-console-producer.sh
kafka-consumer-groups.sh
kafka-consumer-perf-test.sh
kafka-delegation-tokens.sh
kafka-delete-records.sh

```
 (kafka.server.KafkaConfig)
[2024-12-17 11:42:02,179] INFO [BrokerServer id=1] Waiting for the broker to be unfenced (kafka.server.BrokerServ
[2024-12-17 11:42:02,212] INFO [BrokerLifecycleManager id=1] The broker has been unfenced. Transitioning from REC
[2024-12-17 11:42:02,212] INFO [BrokerServer id=1] Finished waiting for the broker to be unfenced (kafka.server.B
[2024-12-17 11:42:02,213] INFO authorizerStart completed for endpoint PLAINTEXT. Endpoint is now READY. (org.apac
[2024-12-17 11:42:02,213] INFO [SocketServer listenerType=BROKER, nodeId=1] Enabling request processing. (kafka.n
[2024-12-17 11:42:02,213] INFO Awaiting socket connections on 0.0.0.0:9092. (kafka.network.DataPlaneAcceptor)
[2024-12-17 11:42:02,214] INFO [BrokerServer id=1] Waiting for all of the authorizer futures to be completed (kaf
[2024-12-17 11:42:02,214] INFO [BrokerServer id=1] Finished waiting for all of the authorizer futures to be compl
[2024-12-17 11:42:02,214] INFO [BrokerServer id=1] Waiting for all of the SocketServer Acceptors to be started (k
[2024-12-17 11:42:02,214] INFO [BrokerServer id=1] Finished waiting for all of the SocketServer Acceptors to be s
[2024-12-17 11:42:02,214] INFO [BrokerServer id=1] Transition from STARTING to STARTED (kafka.server.BrokerServer
[2024-12-17 11:42:02,214] INFO Kafka version: 3.9.0 (org.apache.kafka.common.utils.AppInfoParser)
[2024-12-17 11:42:02,215] INFO Kafka commitId: a60e31147e6b01ee (org.apache.kafka.common.utils.AppInfoParser)
[2024-12-17 11:42:02,215] INFO Kafka startTimeMs: 1734421322214 (org.apache.kafka.common.utils.AppInfoParser)
[2024-12-17 11:42:02,216] INFO [KafkaRaftServer nodeId=1] Kafka Server started (kafka.server.KafkaRaftServer)
```

Создаём топик:

```
$ bin/kafka-topics.sh --create --topic quickstart-events --bootstrap-server
localhost:9092
```

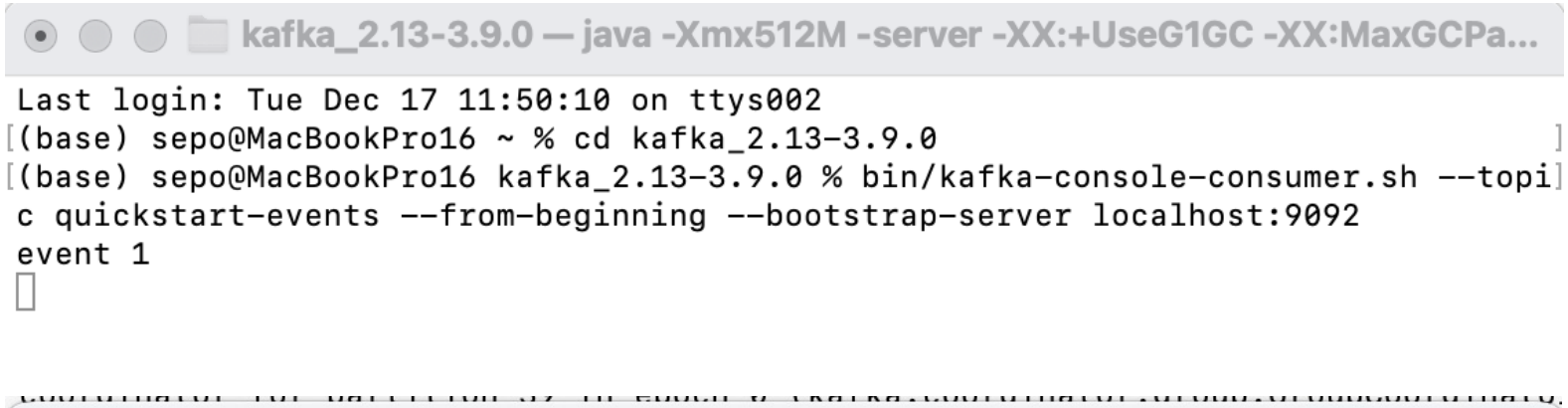
Пишем сообщения в топик:

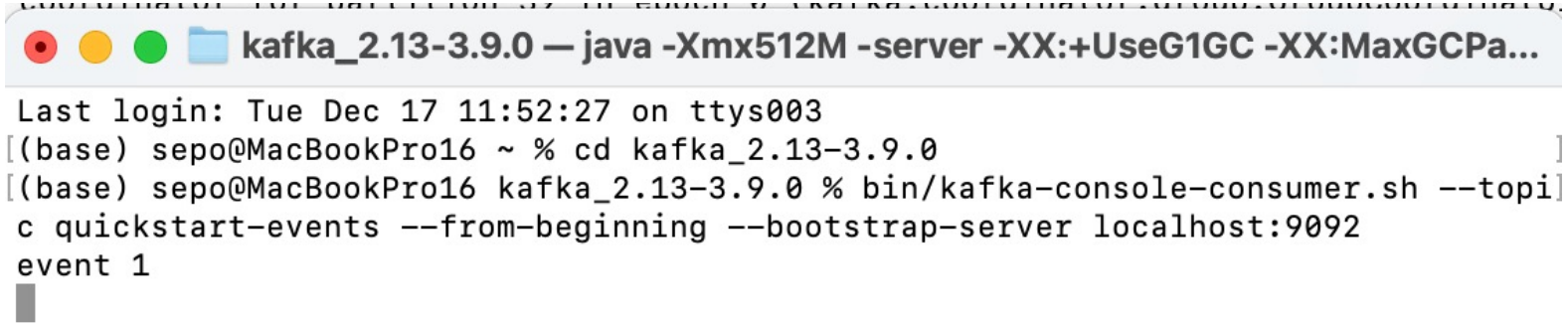
```
$ bin/kafka-console-producer.sh --topic quickstart-events --bootstrap-server
localhost:9092
>This is my first event
>This is my second event
```

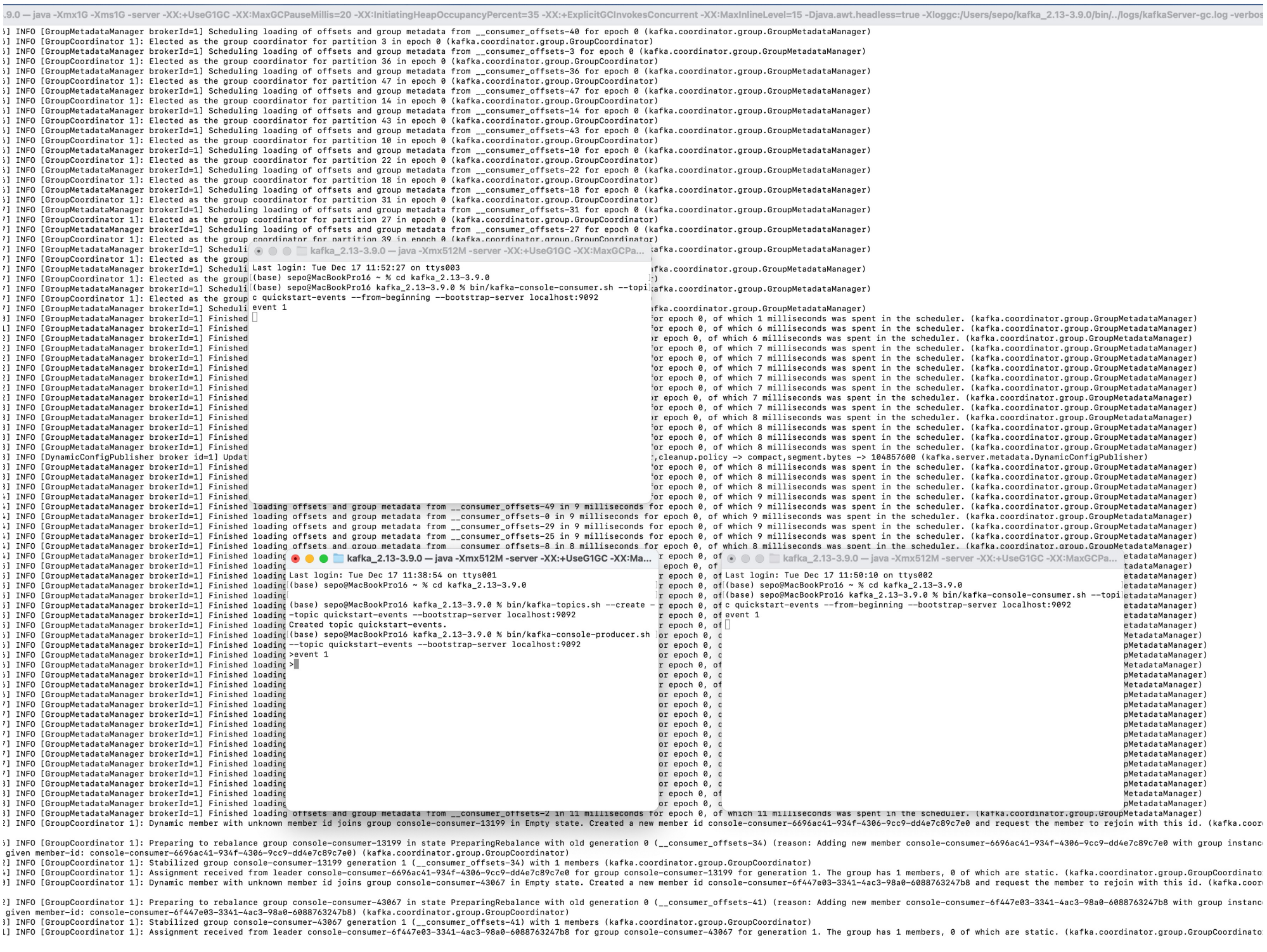
Читаем сообщения из топика:

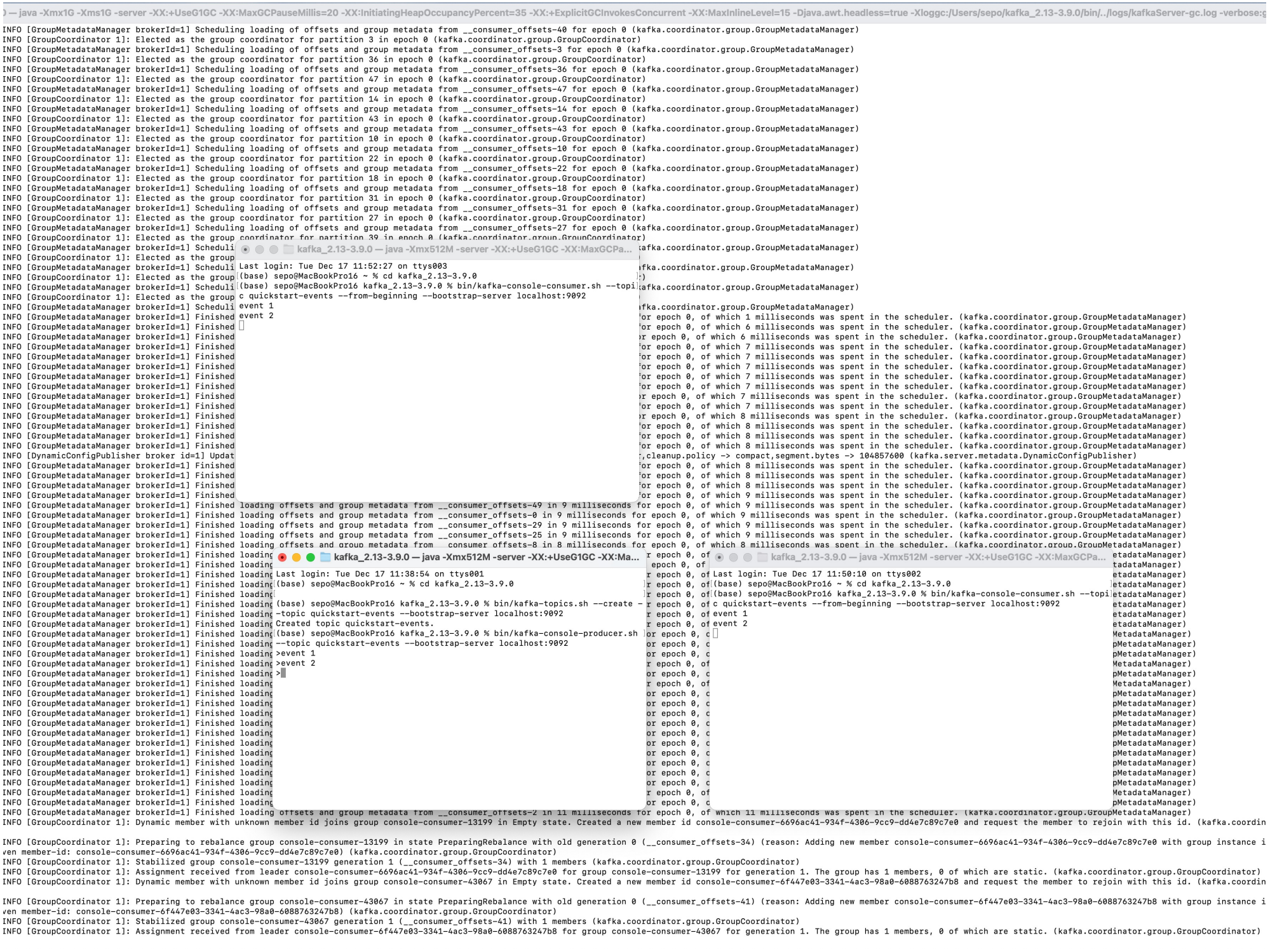
```
$ bin/kafka-console-consumer.sh --topic quickstart-events --from-beginning
--bootstrap-server localhost:9092
This is my first event
This is my second event
```


kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:Ma...
Last login: Tue Dec 17 11:38:54 on ttys001
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-topics.sh --create --topic quickstart-events --bootstrap-server localhost:9092
Created topic quickstart-events.
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-producer.sh --topic quickstart-events --bootstrap-server localhost:9092
>event 1
>


kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:MaxGCPa...
Last login: Tue Dec 17 11:50:10 on ttys002
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-consumer.sh --topic quickstart-events --from-beginning --bootstrap-server localhost:9092
event 1


kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:MaxGCPa...
Last login: Tue Dec 17 11:52:27 on ttys003
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-consumer.sh --topic quickstart-events --from-beginning --bootstrap-server localhost:9092
event 1


kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:MaxGCPa...
Last login: Tue Dec 17 11:52:27 on ttys003
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-consumer.sh --topi
c quickstart-events --from-beginning --bootstrap-server localhost:9092
event 1
kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:Ma...
Last login: Tue Dec 17 11:38:54 on ttys001
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-topics.sh --create -
-topic quickstart-events --bootstrap-server localhost:9092
Created topic quickstart-events.
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-producer.sh
--topic quickstart-events --bootstrap-server localhost:9092
>event 1
>
kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:MaxGCPa...
Last login: Tue Dec 17 11:50:10 on ttys002
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-consumer.sh --topi
c quickstart-events --from-beginning --bootstrap-server localhost:9092
event 1


kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:MaxGCPa...
Last login: Tue Dec 17 11:52:27 on ttys003
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-consumer.sh --topic quickstart-events --from-beginning --bootstrap-server localhost:9092
event 1
event 2
kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:Ma...
Last login: Tue Dec 17 11:38:54 on ttys001
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-topics.sh --create --topic quickstart-events --bootstrap-server localhost:9092
Created topic quickstart-events.
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-producer.sh --topic quickstart-events --bootstrap-server localhost:9092
>event 1
>event 2
>
kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:MaxGCPa...
Last login: Tue Dec 17 11:50:10 on ttys002
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-consumer.sh --topic quickstart-events --from-beginning --bootstrap-server localhost:9092
event 1
event 2

Библиотеки для работы с Kafka на Python:

1. confluent-kafka-python разработана Confluent, компанией, стоящей за коммерческим развитием Kafka. Библиотека обеспечивает высокую производительность и поддерживает последние функции Kafka, включая управление потреблением сообщений.
Она написана на C и Python

2. kafka-python полностью написана на питоне, что делает ее легко устанавливаемой и используемой. Идеально подходит для разработчиков, которые предпочитают работать исключительно в питон-экосистеме, не опираясь на внешние зависимости C-библиотек. Следует учесть, что она может быть менее производительной по сравнению с confluent-kafka-python.

https://kafka-python.readthedocs.io

kafka-python

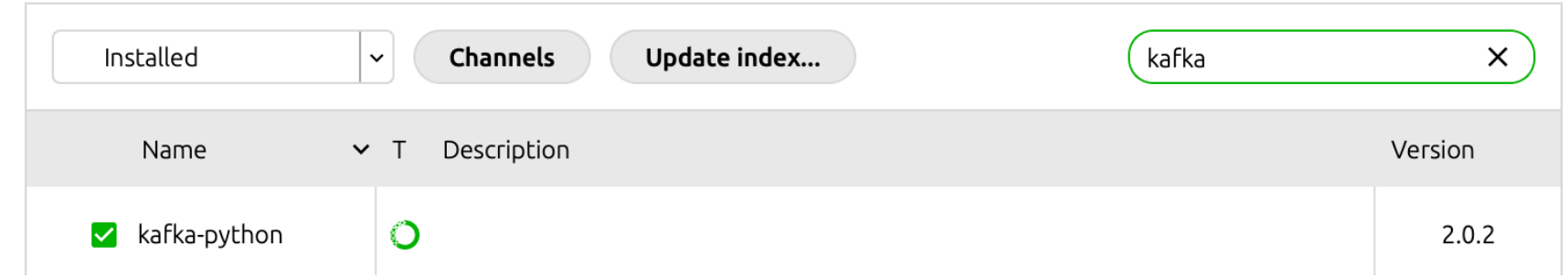
Python client for the Apache Kafka distributed stream processing system. kafka-python is designed to function much like the official java client, with a sprinkling of pythonic interfaces (e.g., consumer iterators).

kafka-python is best used with newer brokers (0.9+), but is backwards-compatible with older versions (to 0.8.0). Some features will only be enabled on newer brokers. For example, fully coordinated consumer groups – i.e., dynamic partition assignment to multiple consumers in the same group – requires use of 0.9 kafka brokers. Supporting this feature for earlier broker releases would require writing and maintaining custom leadership election and membership / health check code (perhaps using zookeeper or consul). For older brokers, you can achieve something similar by manually assigning different partitions to each consumer instance with config management tools like chef, ansible, etc. This approach will work fine, though it does not support rebalancing on failures. See Compatibility for more details.

Please note that the master branch may contain unreleased features. For release documentation, please see readthedocs and/or python's inline help.

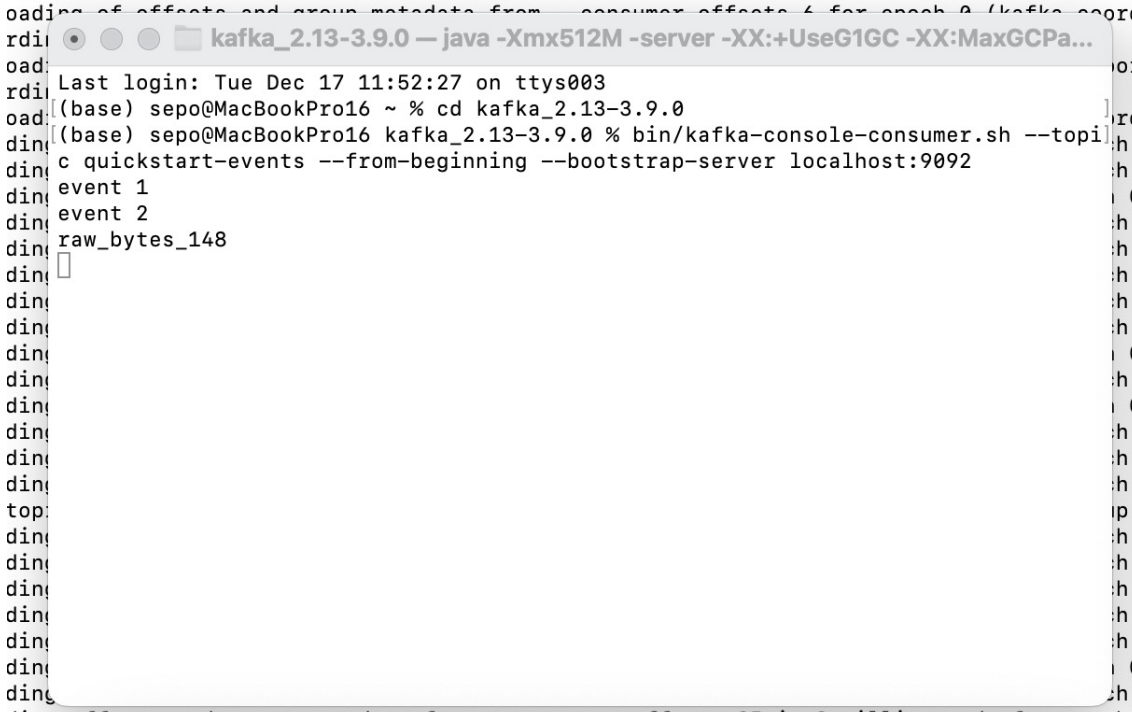
```
>>> pip install kafka-python
```

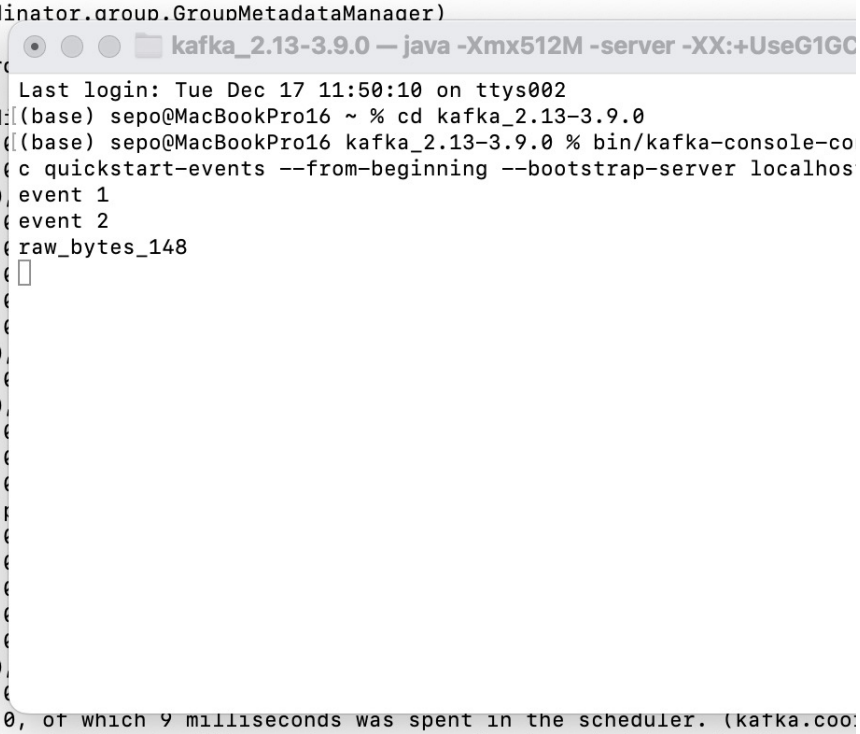
KafkaConsumer



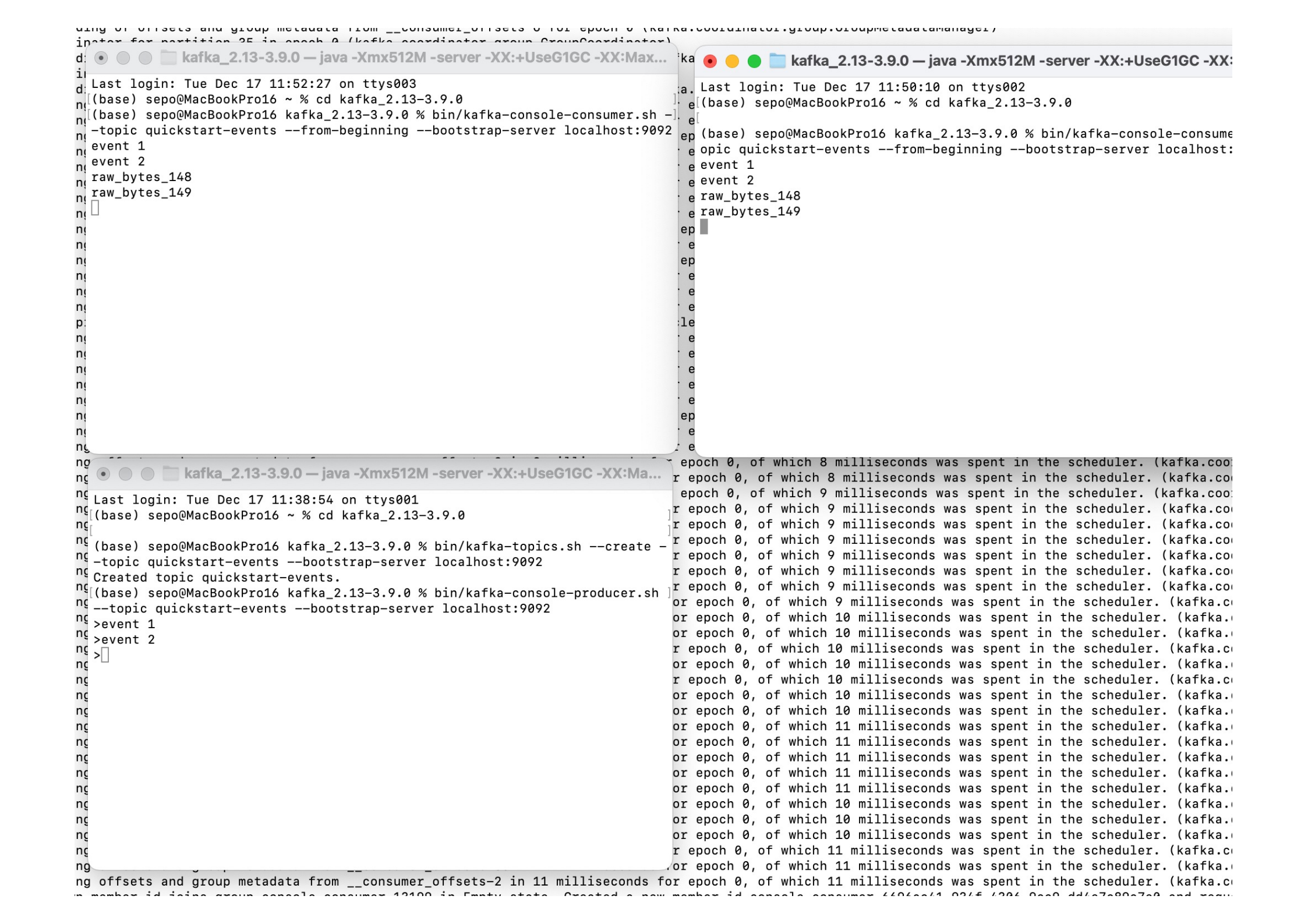

```
[2]: from kafka import KafkaProducer
     from kafka.errors import KafkaError

[4]: producer = KafkaProducer(bootstrap_servers=['192.168.1.144:9092'])

[6]: future = producer.send('quickstart-events', value=b'raw_bytes_148')

[ ]:
```

```
[2]: from kafka import KafkaProducer
     from kafka.errors import KafkaError

[4]: producer = KafkaProducer(bootstrap_servers=['192.168.1.144:9092'])

[6]: future = producer.send('quickstart-events', value=b'raw_bytes_148')

[8]: future = producer.send('quickstart-events', value=b'raw_bytes_149')

[ ]:
```


kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:MaxGCPa...
Last login: Tue Dec 17 11:52:27 on ttys003
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-consumer.sh --topic quickstart-events --from-beginning --bootstrap-server localhost:9092
event 1
event 2
raw_bytes_148


kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC
Last login: Tue Dec 17 11:50:10 on ttys002
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-co
c quickstart-events --from-beginning --bootstrap-server localhos
event 1
event 2
raw_bytes_148


kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:Ma...
Last login: Tue Dec 17 11:38:54 on ttys001
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-topics.sh --create --topic quickstart-events --bootstrap-server localhost:9092
Created topic quickstart-events.
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-producer.sh --topic quickstart-events --bootstrap-server localhost:9092
>event 1
>event 2
>


kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:Max...
Last login: Tue Dec 17 11:52:27 on ttys003
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-consumer.sh --topic quickstart-events --from-beginning --bootstrap-server localhost:9092
event 1
event 2
raw_bytes_148
raw_bytes_149
kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:
Last login: Tue Dec 17 11:50:10 on ttys002
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-consume
opic quickstart-events --from-beginning --bootstrap-server localhost:
event 1
event 2
raw_bytes_148
raw_bytes_149
kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:Ma...
Last login: Tue Dec 17 11:38:54 on ttys001
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-topics.sh --create --topic quickstart-events --bootstrap-server localhost:9092
Created topic quickstart-events.
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-producer.sh --topic quickstart-events --bootstrap-server localhost:9092
>event 1
>event 2
>
epoch 0, of which 8 milliseconds was spent in the scheduler. (kafka.coo
r epoch 0, of which 8 milliseconds was spent in the scheduler. (kafka.co
epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka.coo
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka.co
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka.co
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka.co
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka.co
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka.co
or epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka.c
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kafka.
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kafka.
r epoch 0, of which 10 milliseconds was spent in the scheduler. (kafka.c
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kafka.
r epoch 0, of which 10 milliseconds was spent in the scheduler. (kafka.c
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kafka.
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kafka.
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kafka.
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kafka.
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kafka.
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kafka.
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kafka.
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kafka.
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kafka.
r epoch 0, of which 11 milliseconds was spent in the scheduler. (kafka.c
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kafka.
ng offsets and group metadata from __consumer_offsets-2 in 11 milliseconds for epoch 0, of which 11 milliseconds was spent in the scheduler. (kafka.c

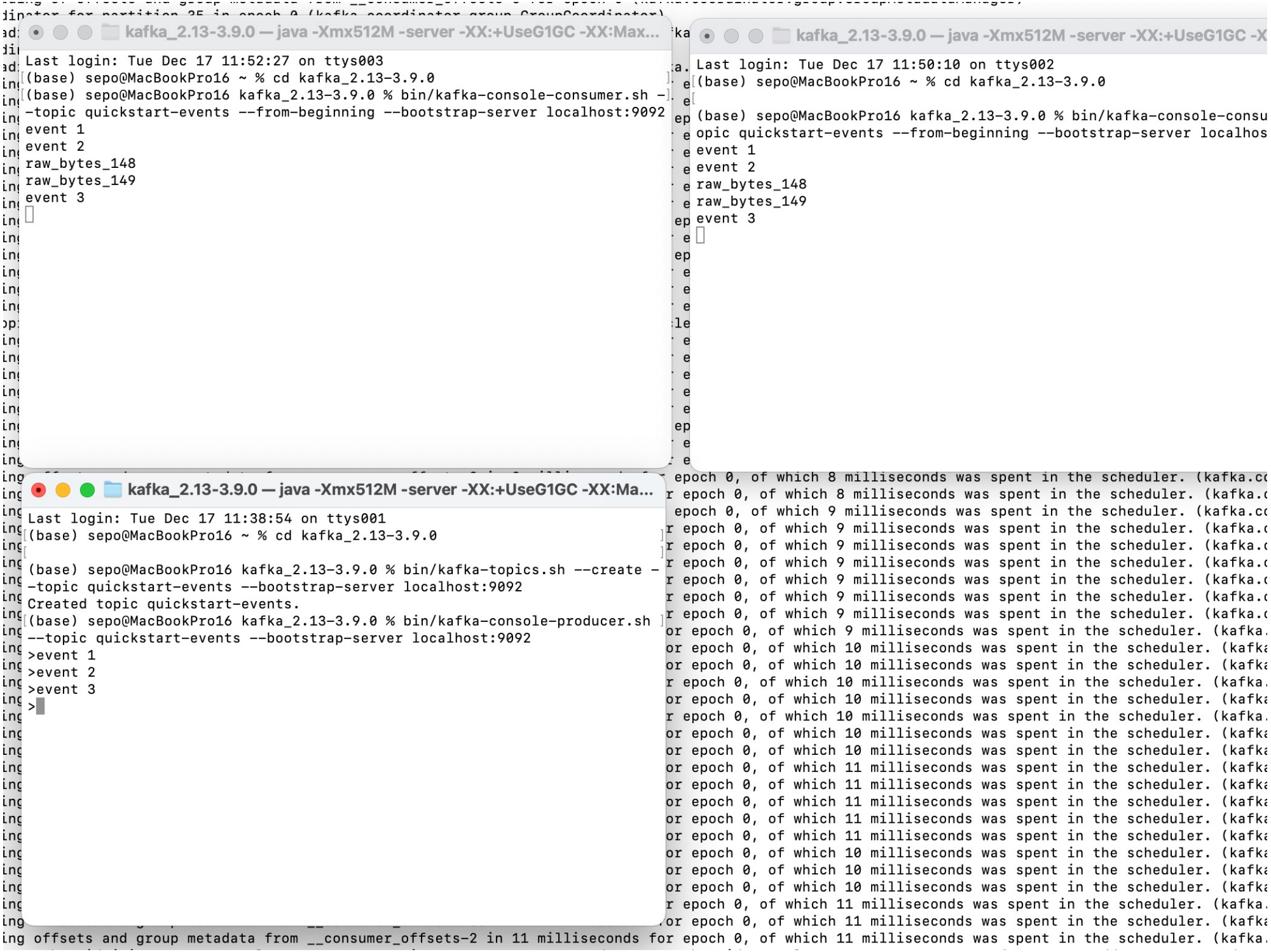
[2]:
```
from kafka import KafkaConsumer
```

[4]:
```
consumer = KafkaConsumer('quickstart-events',
                         auto_offset_reset='earliest',
                         bootstrap_servers=['localhost:9092'])
```

[*]:
```
for message in consumer:
    # message value and key are raw bytes -- decode if necessary!
    # e.g., for unicode: `message.value.decode('utf-8')`
    print ("%s:%d:%d: key=%s value=%s" % (message.topic, message.partition,
                                          message.offset, message.key,
                                          message.value.decode('utf-8')))
```

```
quickstart-events:0:0: key=None value=event 1
quickstart-events:0:1: key=None value=event 2
quickstart-events:0:2: key=None value=raw_bytes_148
quickstart-events:0:3: key=None value=raw_bytes_149
```

[ ]:


kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:Max...
Last login: Tue Dec 17 11:52:27 on ttys003
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-consumer.sh --topic quickstart-events --from-beginning --bootstrap-server localhost:9092
event 1
event 2
raw_bytes_148
raw_bytes_149
event 3
kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -X
Last login: Tue Dec 17 11:50:10 on ttys002
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-consu
opic quickstart-events --from-beginning --bootstrap-server localhos
event 1
event 2
raw_bytes_148
raw_bytes_149
event 3
kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:Ma...
Last login: Tue Dec 17 11:38:54 on ttys001
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-topics.sh --create --topic quickstart-events --bootstrap-server localhost:9092
Created topic quickstart-events.
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-producer.sh --topic quickstart-events --bootstrap-server localhost:9092
>event 1
>event 2
>event 3
>
ing offsets and group metadata from __consumer_offsets-2 in 11 milliseconds for epoch 0, of which 11 milliseconds was spent in the scheduler. (kafka.

[2]:
```
from kafka import KafkaConsumer
```

[4]:
```
consumer = KafkaConsumer('quickstart-events',
                         auto_offset_reset='earliest',
                         bootstrap_servers=['localhost:9092'])
```

[*]:
```
for message in consumer:
    # message value and key are raw bytes -- decode if necessary!
    # e.g., for unicode: `message.value.decode('utf-8')`
    print ("%s:%d:%d: key=%s value=%s" % (message.topic, message.partition,
                                          message.offset, message.key,
                                          message.value.decode('utf-8')))
```

```
quickstart-events:0:0: key=None value=event 1
quickstart-events:0:1: key=None value=event 2
quickstart-events:0:2: key=None value=raw_bytes_148
quickstart-events:0:3: key=None value=raw_bytes_149
quickstart-events:0:4: key=None value=event 3
```

[ ]:

[2]:
```
from kafka import KafkaConsumer
```

[4]:
```
consumer = KafkaConsumer('quickstart-events',
                         bootstrap_servers=['localhost:9092'])
```

[*]:
```
for message in consumer:
    # message value and key are raw bytes -- decode if necessary!
    # e.g., for unicode: `message.value.decode('utf-8')`
    print ("%s:%d:%d: key=%s value=%s" % (message.topic, message.partition,
                                          message.offset, message.key,
                                          message.value.decode('utf-8')))
```

[ ]:

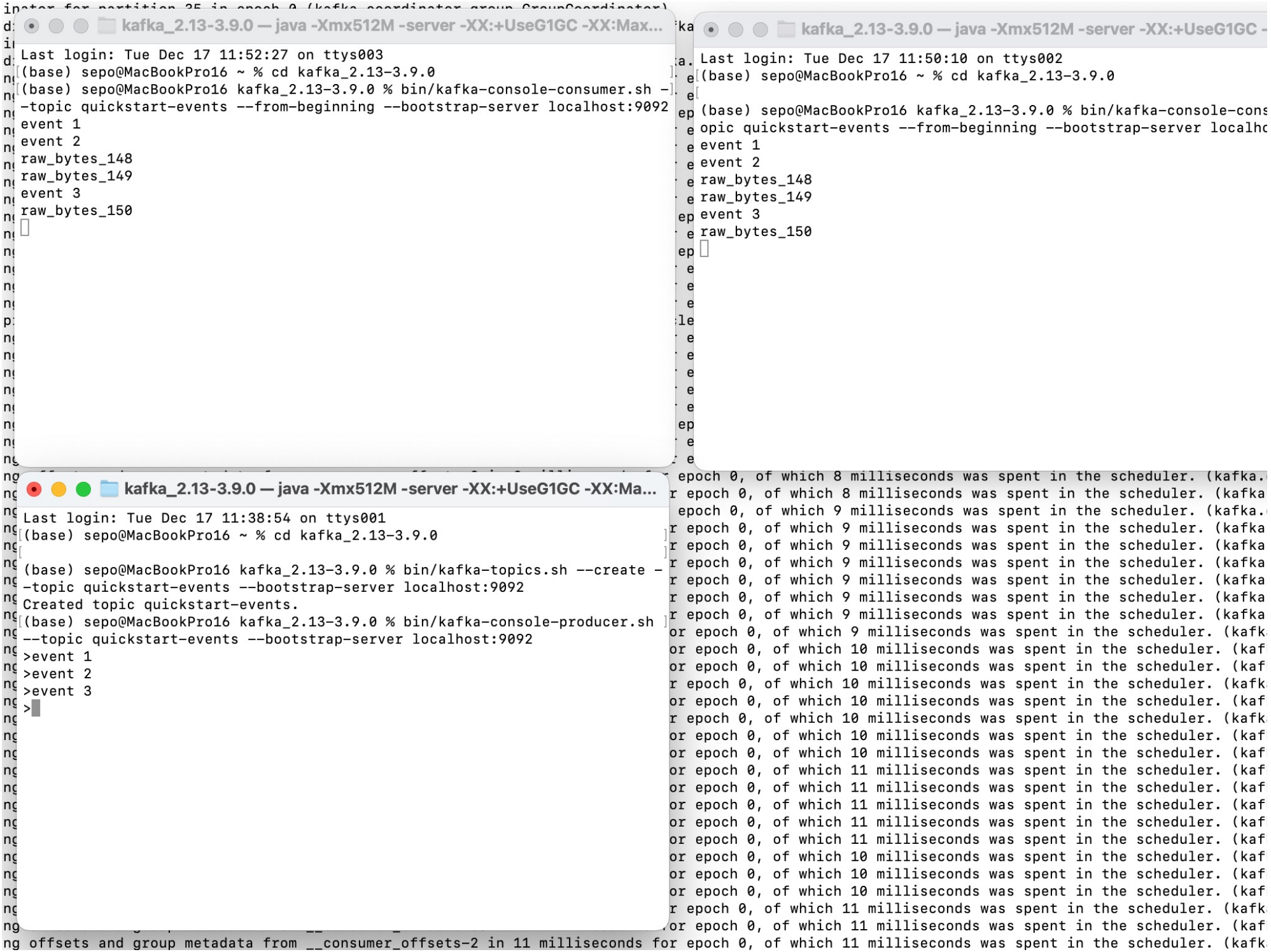
[2]:
```
from kafka import KafkaConsumer
```

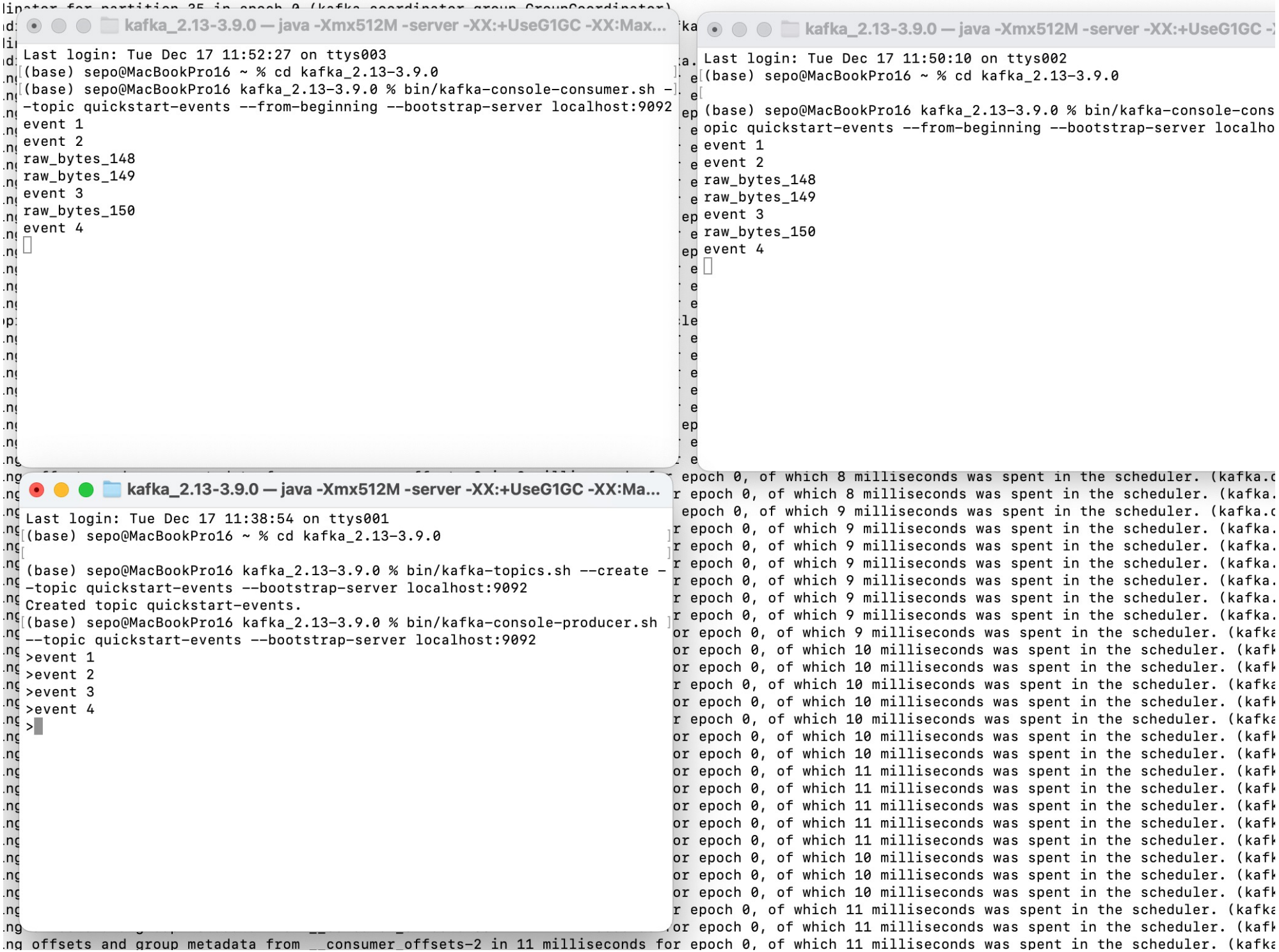
[4]:
```
consumer = KafkaConsumer('quickstart-events',
                         bootstrap_servers=['localhost:9092'])
```

[*]:
```
for message in consumer:
    # message value and key are raw bytes -- decode if necessary!
    # e.g., for unicode: `message.value.decode('utf-8')`
    print ("%s:%d:%d: key=%s value=%s" % (message.topic, message.partition,
                                          message.offset, message.key,
                                          message.value.decode('utf-8')))
```

```
quickstart-events:0:5: key=None value=raw_bytes_150
```

[ ]:


kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:Max...
Last login: Tue Dec 17 11:52:27 on ttys003
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-consumer.sh --topic quickstart-events --from-beginning --bootstrap-server localhost:9092
event 1
event 2
raw_bytes_148
raw_bytes_149
event 3
raw_bytes_150
kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -
Last login: Tue Dec 17 11:50:10 on ttys002
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-cons
opic quickstart-events --from-beginning --bootstrap-server localhc
event 1
event 2
raw_bytes_148
raw_bytes_149
event 3
raw_bytes_150
kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:Ma...
Last login: Tue Dec 17 11:38:54 on ttys001
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-topics.sh --create --topic quickstart-events --bootstrap-server localhost:9092
Created topic quickstart-events.
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-producer.sh --topic quickstart-events --bootstrap-server localhost:9092
>event 1
>event 2
>event 3
>
epoch 0, of which 8 milliseconds was spent in the scheduler. (kafka.
r epoch 0, of which 8 milliseconds was spent in the scheduler. (kafka
epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka.
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka
or epoch 0, of which 9 milliseconds was spent in the scheduler. (kafk
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kaf
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kaf
r epoch 0, of which 10 milliseconds was spent in the scheduler. (kafk
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kaf
r epoch 0, of which 10 milliseconds was spent in the scheduler. (kafk
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kaf
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kaf
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kaf
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kaf
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kaf
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kaf
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kaf
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kaf
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kaf
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kaf
r epoch 0, of which 11 milliseconds was spent in the scheduler. (kafk
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kaf
ng offsets and group metadata from __consumer_offsets-2 in 11 milliseconds for epoch 0, of which 11 milliseconds was spent in the scheduler. (kafk


kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:Max...
Last login: Tue Dec 17 11:52:27 on ttys003
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-consumer.sh --topic quickstart-events --from-beginning --bootstrap-server localhost:9092
event 1
event 2
raw_bytes_148
raw_bytes_149
event 3
raw_bytes_150
event 4
kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -X
Last login: Tue Dec 17 11:50:10 on ttys002
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-cons
opic quickstart-events --from-beginning --bootstrap-server localho
event 1
event 2
raw_bytes_148
raw_bytes_149
event 3
raw_bytes_150
event 4
kafka_2.13-3.9.0 — java -Xmx512M -server -XX:+UseG1GC -XX:Ma...
Last login: Tue Dec 17 11:38:54 on ttys001
(base) sepo@MacBookPro16 ~ % cd kafka_2.13-3.9.0
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-topics.sh --create --topic quickstart-events --bootstrap-server localhost:9092
Created topic quickstart-events.
(base) sepo@MacBookPro16 kafka_2.13-3.9.0 % bin/kafka-console-producer.sh --topic quickstart-events --bootstrap-server localhost:9092
>event 1
>event 2
>event 3
>event 4
>
epoch 0, of which 8 milliseconds was spent in the scheduler. (kafka.c
r epoch 0, of which 8 milliseconds was spent in the scheduler. (kafka.
epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka.c
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka.
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka.
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka.
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka.
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka.
r epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka.
or epoch 0, of which 9 milliseconds was spent in the scheduler. (kafka
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kafk
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kafk
r epoch 0, of which 10 milliseconds was spent in the scheduler. (kafka
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kafk
r epoch 0, of which 10 milliseconds was spent in the scheduler. (kafka
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kafk
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kafk
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kafk
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kafk
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kafk
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kafk
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kafk
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kafk
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kafk
or epoch 0, of which 10 milliseconds was spent in the scheduler. (kafk
r epoch 0, of which 11 milliseconds was spent in the scheduler. (kafka
or epoch 0, of which 11 milliseconds was spent in the scheduler. (kafk
ng offsets and group metadata from __consumer_offsets-2 in 11 milliseconds for epoch 0, of which 11 milliseconds was spent in the scheduler. (kafka

[2]:
```
from kafka import KafkaConsumer
```

[4]:
```
consumer = KafkaConsumer('quickstart-events',
                         bootstrap_servers=['localhost:9092'])
```

[*]:
```
for message in consumer:
    # message value and key are raw bytes -- decode if necessary!
    # e.g., for unicode: `message.value.decode('utf-8')`
    print ("%s:%d:%d: key=%s value=%s" % (message.topic, message.partition,
                                          message.offset, message.key,
                                          message.value.decode('utf-8')))
```

```
quickstart-events:0:5: key=None value=raw_bytes_150
quickstart-events:0:6: key=None value=event 4
```

[ ]:

[2]:
```
from kafka import KafkaConsumer
```

[4]:
```
consumer = KafkaConsumer('quickstart-events',
                         auto_offset_reset='earliest',
                         bootstrap_servers=['localhost:9092'])
```

[*]:
```
for message in consumer:
    # message value and key are raw bytes -- decode if necessary!
    # e.g., for unicode: `message.value.decode('utf-8')`
    print ("%s:%d:%d: key=%s value=%s" % (message.topic, message.partition,
                                          message.offset, message.key,
                                          message.value.decode('utf-8')))
```

```
quickstart-events:0:0: key=None value=event 1
quickstart-events:0:1: key=None value=event 2
quickstart-events:0:2: key=None value=raw_bytes_148
quickstart-events:0:3: key=None value=raw_bytes_149
quickstart-events:0:4: key=None value=event 3
quickstart-events:0:5: key=None value=raw_bytes_150
quickstart-events:0:6: key=None value=event 4
```

[ ]:

- **group_id** (*str or None*) – The name of the consumer group to join for dynamic partition assignment (if enabled), and to use for fetching and committing offsets. If None, auto-partition assignment (via group coordinator) and offset commits are disabled. Default: None

  Имя группы потребителей для присоединения к динамическому назначению разделов (если включено) и использования для извлечения и фиксации смещений. Если None, автоматическое назначение разделов (через координатора группы) и фиксации смещений отключены. По умолчанию: None

```
[4]: from kafka import KafkaConsumer

[6]: consumer = KafkaConsumer('quickstart-events',
                              group_id ='g_1',
                              auto_offset_reset='earliest',
                              bootstrap_servers=['localhost:9092'])

[*]: for message in consumer:
         # message value and key are raw bytes -- decode if necessary!
         # e.g., for unicode: `message.value.decode('utf-8')`
         print ("%s:%d:%d: key=%s value=%s" % (message.topic, message.partition,
                                               message.offset, message.key,
                                               message.value.decode('utf-8')))

quickstart-events:0:0: key=None value=event 1
quickstart-events:0:1: key=None value=event 2
quickstart-events:0:2: key=None value=raw_bytes_148
quickstart-events:0:3: key=None value=raw_bytes_149
quickstart-events:0:4: key=None value=event 3
quickstart-events:0:5: key=None value=raw_bytes_150
quickstart-events:0:6: key=None value=event 4

[ ]:
```

jupyter kafka-Copy2 Last Checkpoint: 5 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

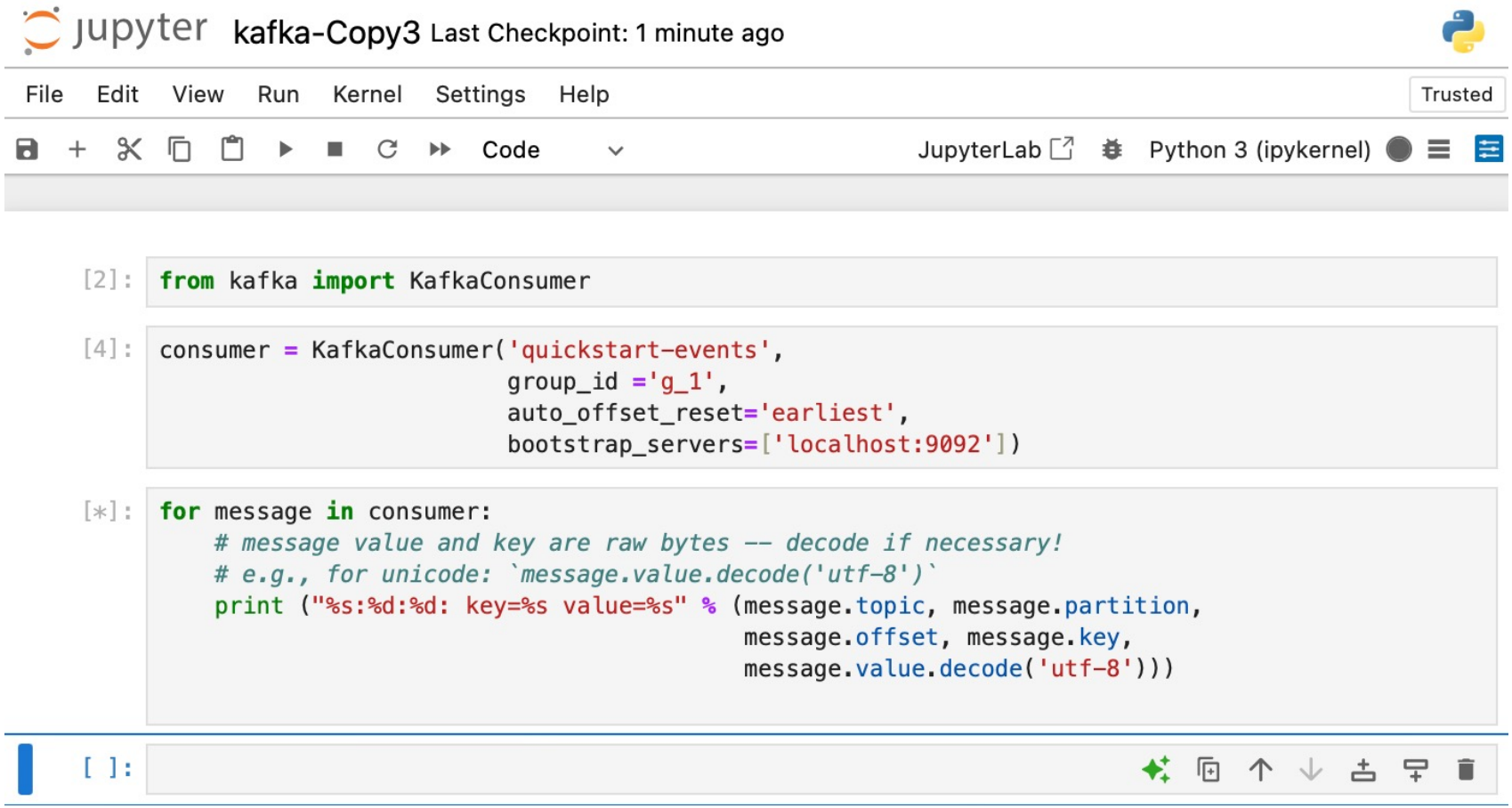
[4]:
```
from kafka import KafkaConsumer
```

[6]:
```
consumer = KafkaConsumer('quickstart-events',
                         group_id ='g_1',
                         auto_offset_reset='earliest',
                         bootstrap_servers=['localhost:9092'])
```

[*]:
```
for message in consumer:
    # message value and key are raw bytes -- decode if necessary!
    # e.g., for unicode: `message.value.decode('utf-8')`
    print ("%s:%d:%d: key=%s value=%s" % (message.topic, message.partition,
                                          message.offset, message.key,
                                          message.value.decode('utf-8')))
```

```
quickstart-events:0:0: key=None value=event 1
quickstart-events:0:1: key=None value=event 2
quickstart-events:0:2: key=None value=raw_bytes_148
quickstart-events:0:3: key=None value=raw_bytes_149
quickstart-events:0:4: key=None value=event 3
quickstart-events:0:5: key=None value=raw_bytes_150
quickstart-events:0:6: key=None value=event 4
```

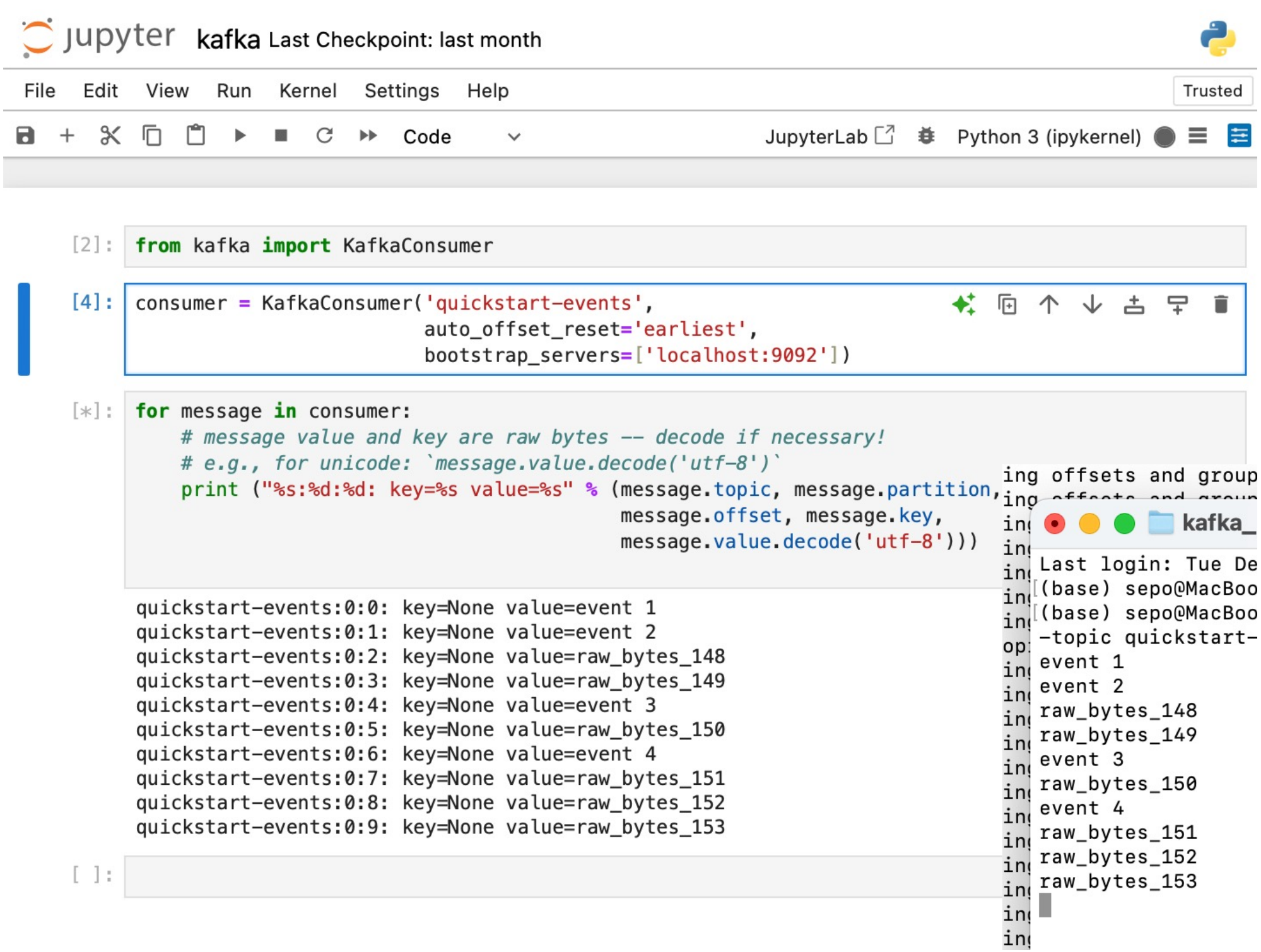
[ ]:


jupyter kafka-Copy3 Last Checkpoint: 1 minute ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
[2]:
from kafka import KafkaConsumer
[4]:
consumer = KafkaConsumer('quickstart-events',
group_id ='g_1',
auto_offset_reset='earliest',
bootstrap_servers=['localhost:9092'])
[*]:
for message in consumer:
# message value and key are raw bytes -- decode if necessary!
# e.g., for unicode: `message.value.decode('utf-8')`
print ("%s:%d:%d: key=%s value=%s" % (message.topic, message.partition,
message.offset, message.key,
message.value.decode('utf-8')))
[ ]:

```
[2]: from kafka import KafkaProducer
     from kafka.errors import KafkaError

[4]: producer = KafkaProducer(bootstrap_servers=['192.168.1.144:9092'])

[6]: future = producer.send('quickstart-events', value=b'raw_bytes_148')

[8]: future = producer.send('quickstart-events', value=b'raw_bytes_149')

[10]: future = producer.send('quickstart-events', value=b'raw_bytes_150')

[12]: future = producer.send('quickstart-events', value=b'raw_bytes_151')

[ ]:
```

```
[2]: from kafka import KafkaConsumer

[4]: consumer = KafkaConsumer('quickstart-events',
                              group_id ='g_1',
                              auto_offset_reset='earliest',
                              bootstrap_servers=['localhost:9092'])

[*]: for message in consumer:
         # message value and key are raw bytes -- decode if necessary!
         # e.g., for unicode: `message.value.decode('utf-8')`
         print ("%s:%d:%d: key=%s value=%s" % (message.topic, message.partition,
                                               message.offset, message.key,
                                               message.value.decode('utf-8')))

quickstart-events:0:7: key=None value=raw_bytes_151
quickstart-events:0:8: key=None value=raw_bytes_152

[ ]:
```

```
kafka_2.13
Last login: Tue Dec 1
(base) sepo@MacBookPr
(base) sepo@MacBookPr
-topic quickstart-eve
event 1
event 2
raw_bytes_148
raw_bytes_149
event 3
raw_bytes_150
event 4
raw_bytes_151
raw_bytes_152
```


jupyter kafka Last Checkpoint: last month
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
[2]:
from kafka import KafkaConsumer
[4]:
consumer = KafkaConsumer('quickstart-events',
auto_offset_reset='earliest',
bootstrap_servers=['localhost:9092'])
[*]:
for message in consumer:
# message value and key are raw bytes -- decode if necessary!
# e.g., for unicode: `message.value.decode('utf-8')`
print ("%s:%d:%d: key=%s value=%s" % (message.topic, message.partition,
message.offset, message.key,
message.value.decode('utf-8')))
quickstart-events:0:0: key=None value=event 1
quickstart-events:0:1: key=None value=event 2
quickstart-events:0:2: key=None value=raw_bytes_148
quickstart-events:0:3: key=None value=raw_bytes_149
quickstart-events:0:4: key=None value=event 3
quickstart-events:0:5: key=None value=raw_bytes_150
quickstart-events:0:6: key=None value=event 4
quickstart-events:0:7: key=None value=raw_bytes_151
quickstart-events:0:8: key=None value=raw_bytes_152
quickstart-events:0:9: key=None value=raw_bytes_153
[ ]:
ing offsets and group
kafka_
Last login: Tue De
(base) sepo@MacBoo
(base) sepo@MacBoo
-topic quickstart-
event 1
event 2
raw_bytes_148
raw_bytes_149
event 3
raw_bytes_150
event 4
raw_bytes_151
raw_bytes_152
raw_bytes_153

STEP 8: TERMINATE THE KAFKA ENVIRONMENT

Now that you reached the end of the quickstart, feel free to tear down the Kafka environment—or continue playing around.

1. Stop the producer and consumer clients with `Ctrl-C`, if you haven't done so already.
2. Stop the Kafka broker with `Ctrl-C`.
3. Lastly, if the Kafka with ZooKeeper section was followed, stop the ZooKeeper server with `Ctrl-C`.

If you also want to delete any data of your local Kafka environment including any events you have created along the way, run the command:

```
$ rm -rf /tmp/kafka-logs /tmp/zookeeper /tmp/kraft-combined-logs
```

Гвен Шапира, Тодд Палино, Раджини Сиварам, Крит Петти

А79 Apache Kafka. Потоковая обработка и анализ данных. 2-е изд. — СПб.: Питер, 2023. — 512 с.: ил. — (Серия «Бестселлеры O'Reilly»).

ISBN 978-5-4461-2288-2

Дилан Скотт, Виктор Гамов, Дейв Клейн

С44 Kafka в действии / пер. с англ. А. Н. Киселева. – М.: ДМК Пресс, 2022. – 310 с.: ил.

ISBN 978-5-93700-118-4

ISBN 978-5-6042412-1-9

Стопфорд, Бен

Проектирование событийно-ориентированных систем: Концепции и шаблоны проектирования сервисов потоковой обработки данных с использованием Apache Kafka / Бен Стопфорд ; Пер. с англ. — 2-е изд., испр. — Иркутск : ITSumma Press, 2019. – 175 с.

The Art of Immutable Architecture

Theory and Practice of Data Management in Distributed Systems

Искусство архитектуры неизменяемых объектов

Майкл Л. Перри

П26 Искусство неизменяемой архитектуры: теория и практика управления данными в распределенных системах / пер. с анг. С. В. Минца; науч. ред. В. С. Яценков. – М.: ДМК Пресс, 2022. – 388 с.: ил.

ISBN 978-5-93700-111-5