

## Урок 2. Базовые Основы



# Цели

- Разобраться чем Кафка отличается от популярных систем обмена сообщениями
- Понять как Кафка хранит данные и обеспечивает гарантию сохранности
- Рассмотреть как записываются и читаются данные



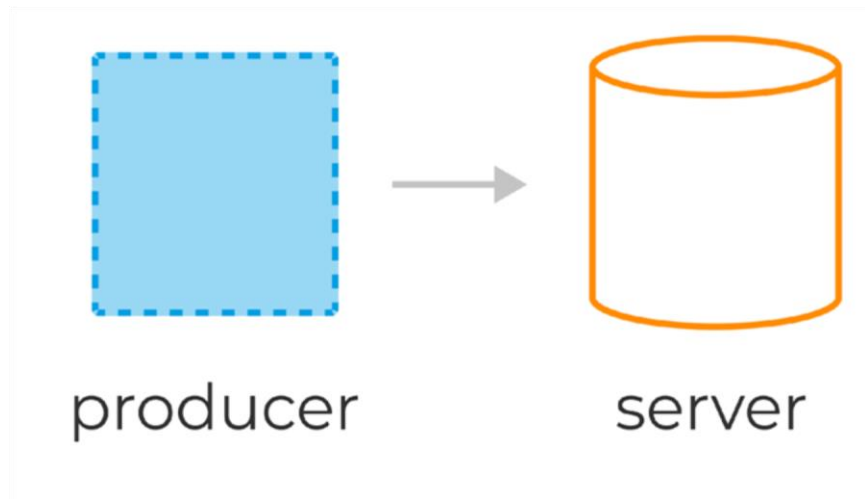
# Kafka vs. Queue

# Kafka vs. Queue

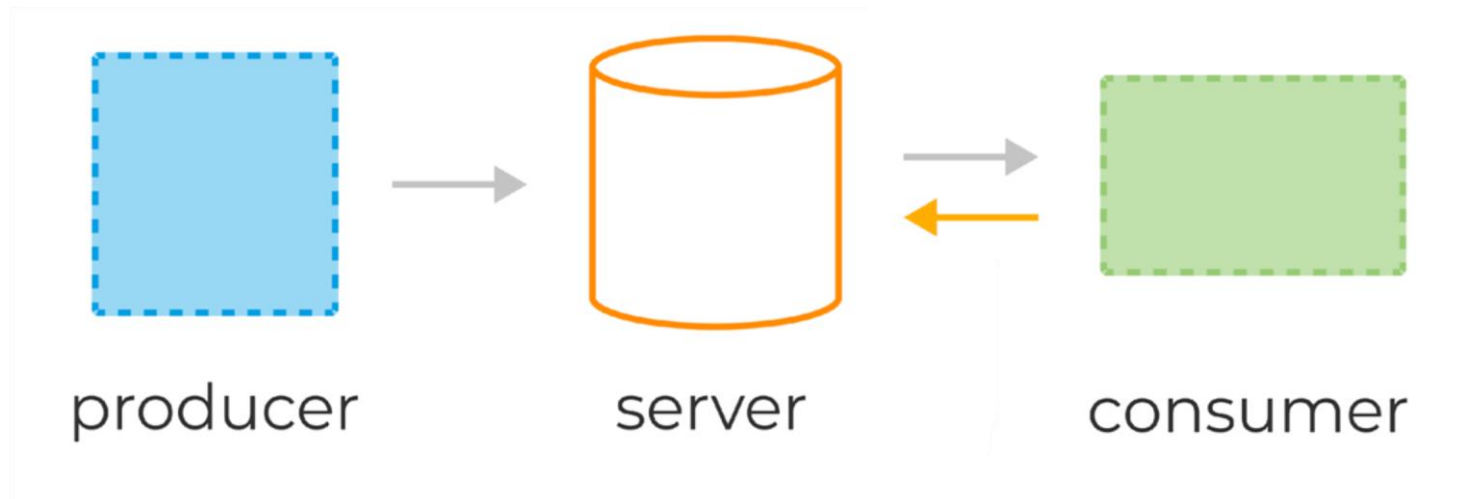


server

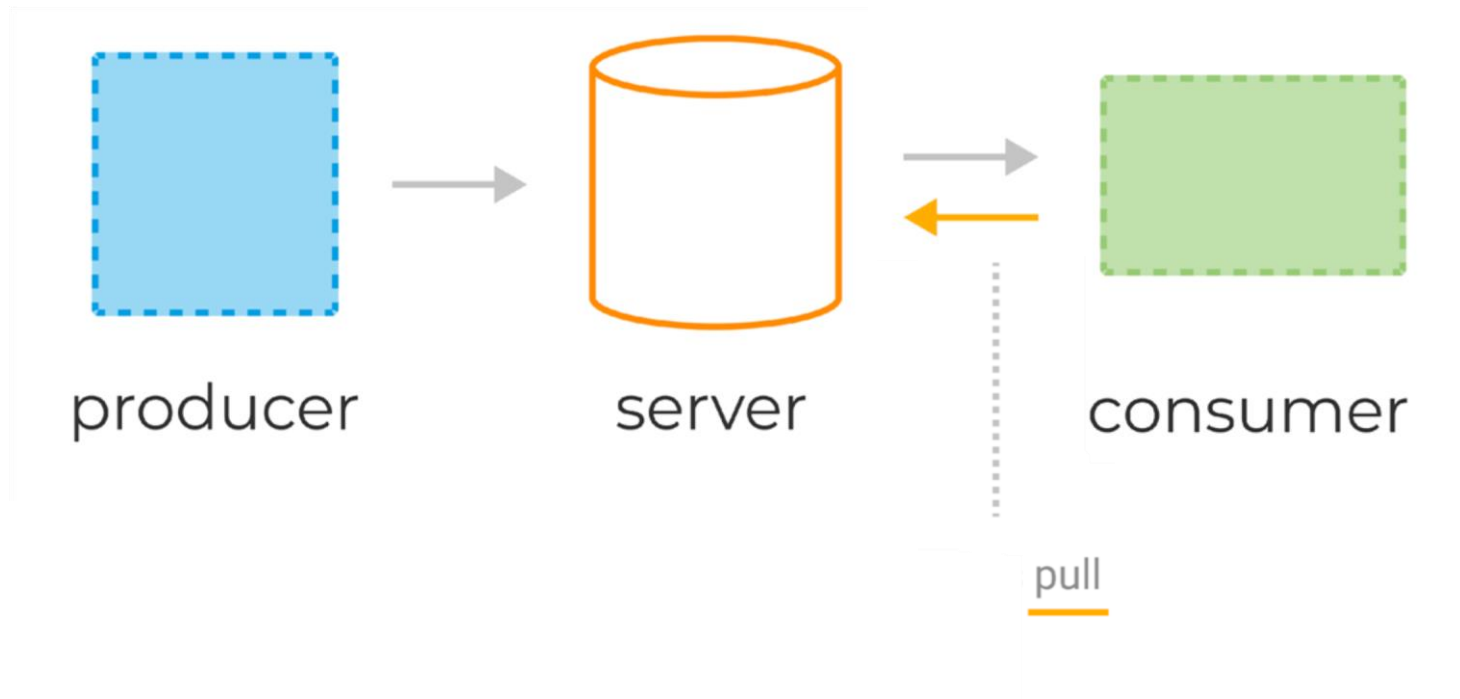
# Kafka vs. Queue



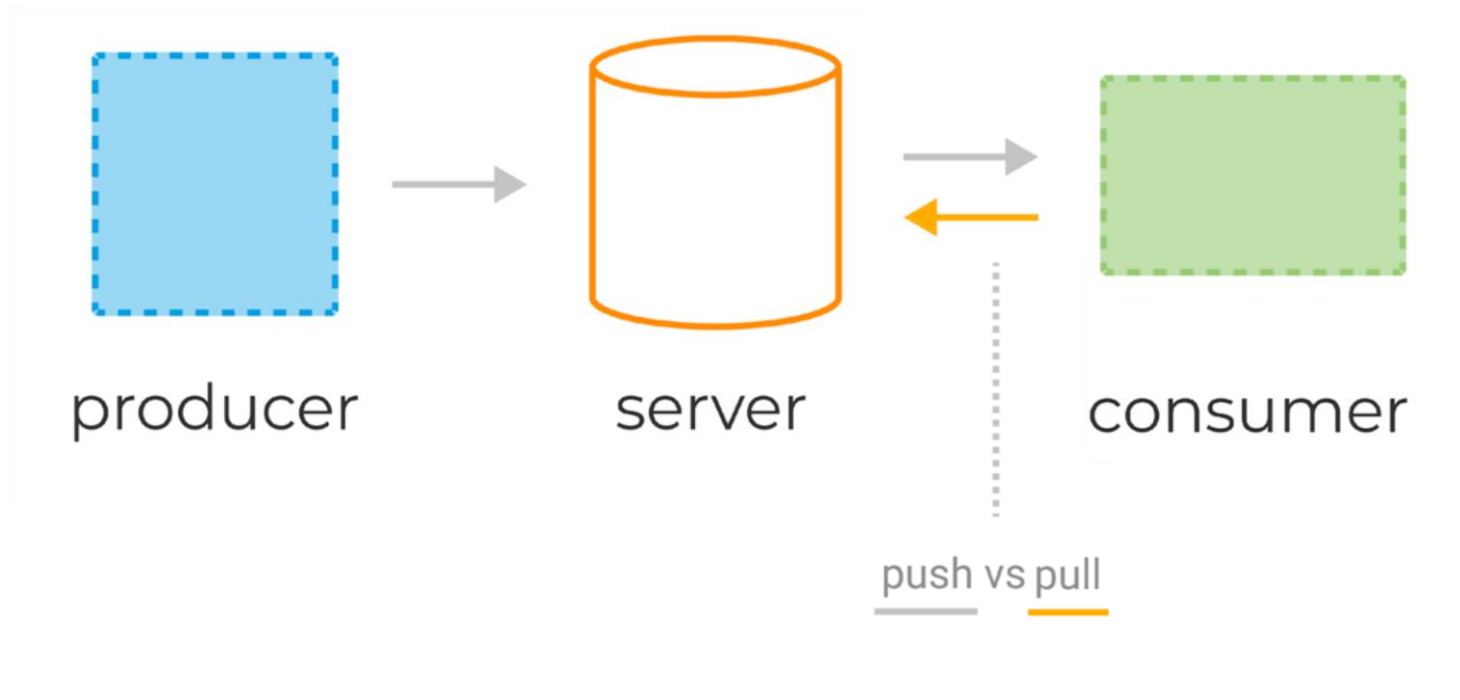
# Kafka vs. Queue



# Kafka vs. Queue

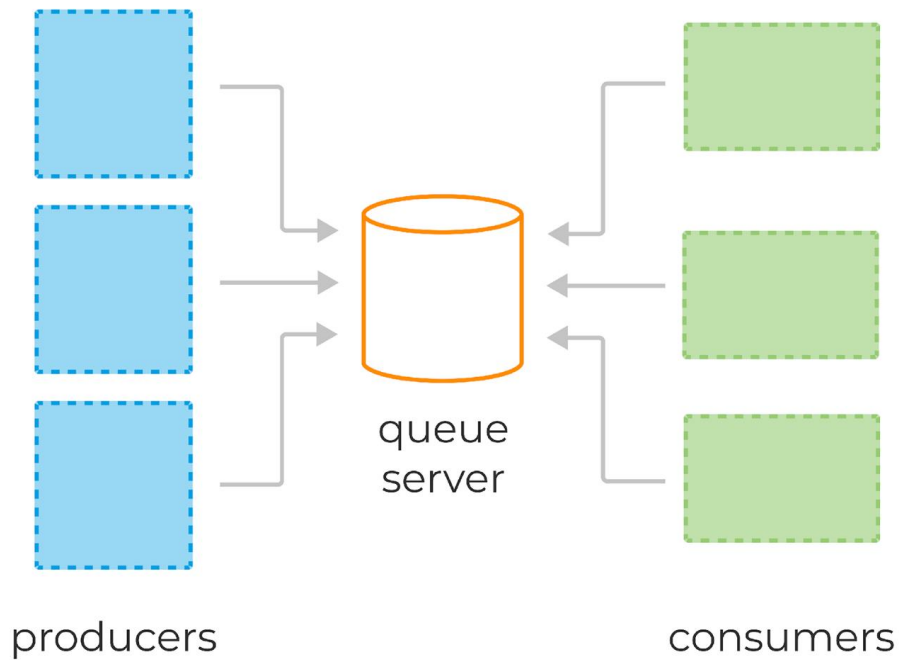


# Kafka vs. Queue

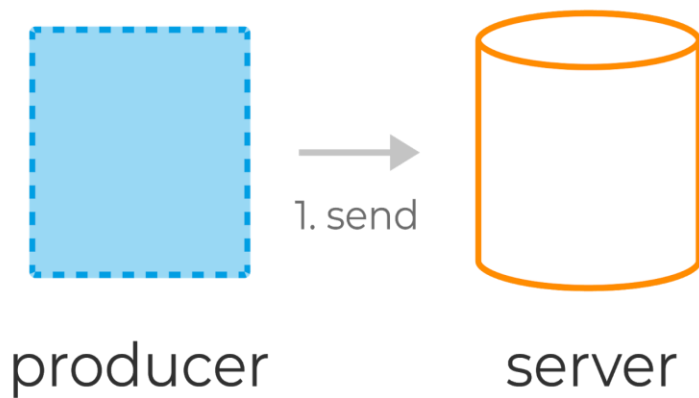




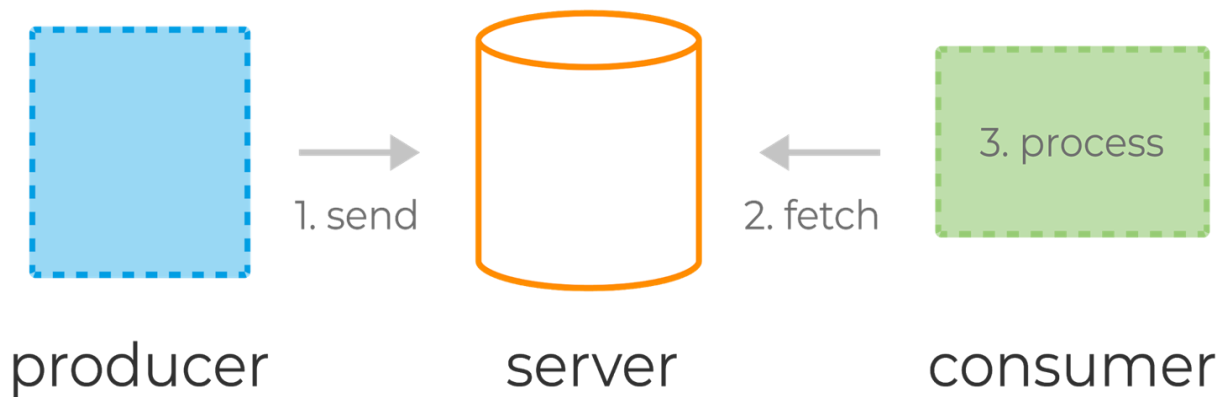
# Kafka vs. Queue



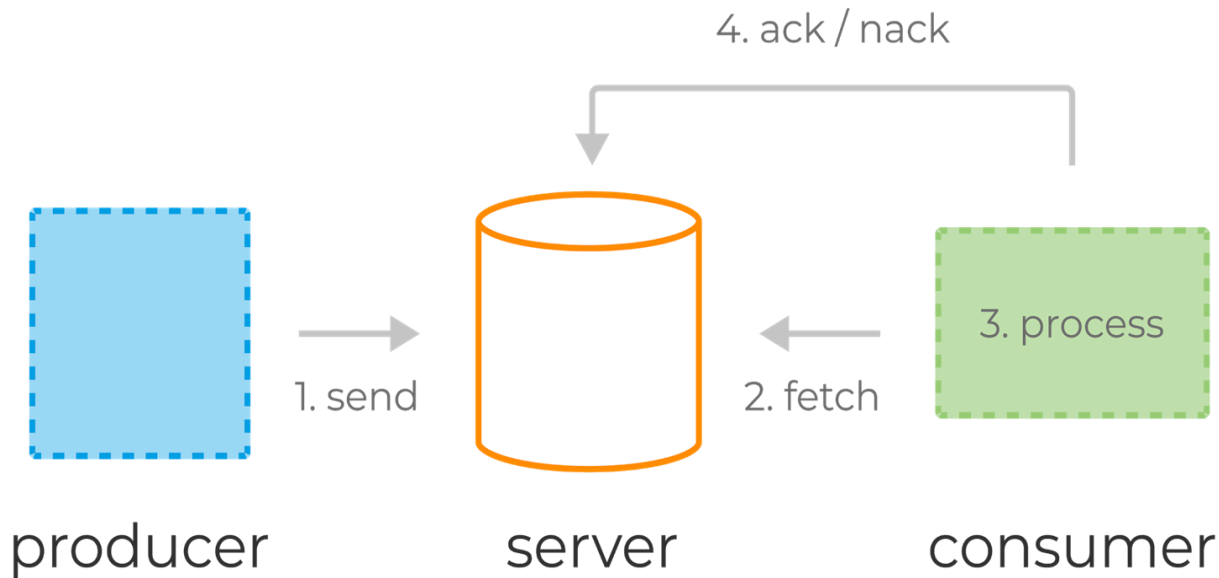
# Kafka vs. Queue



# Kafka vs. Queue



# Kafka vs. Queue



# Kafka vs. Queue



VS.

The RabbitMQ logo, featuring an orange icon of a rabbit head to the left of the text "RabbitMQ".

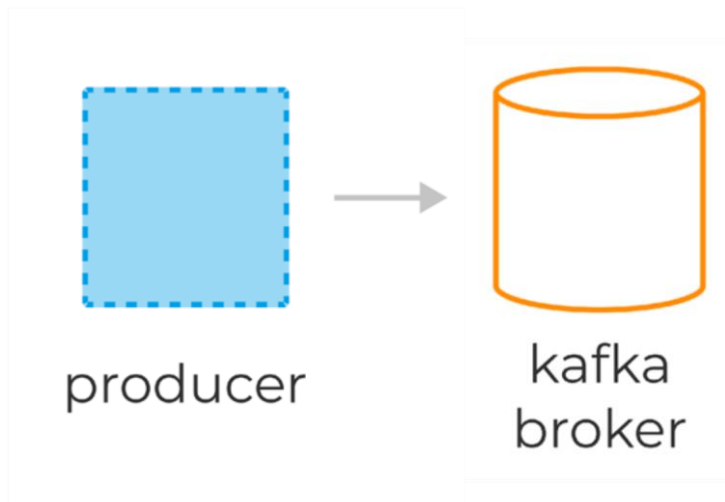
...

# Kafka vs. Queue

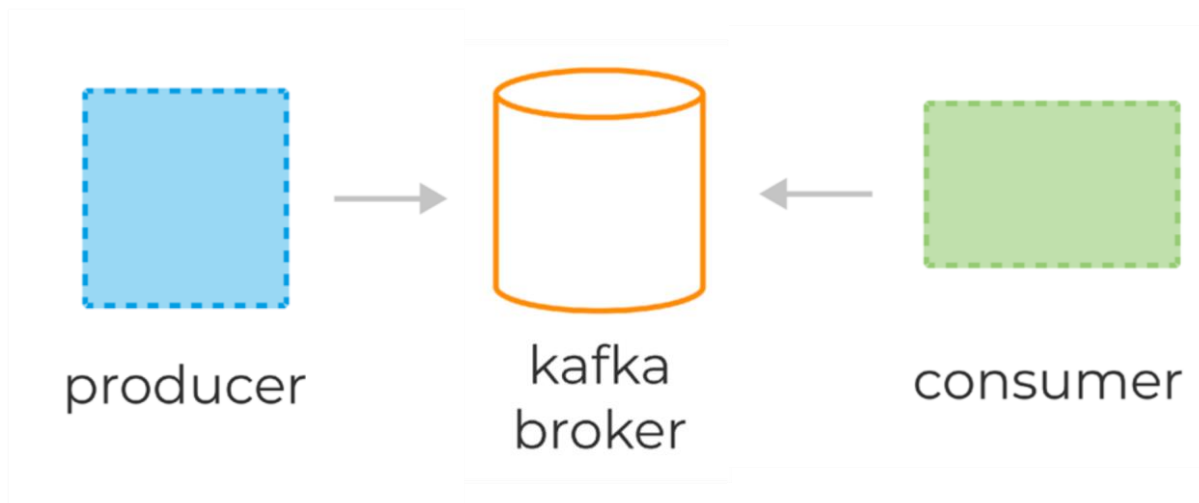


kafka  
broker

# Kafka vs. Queue



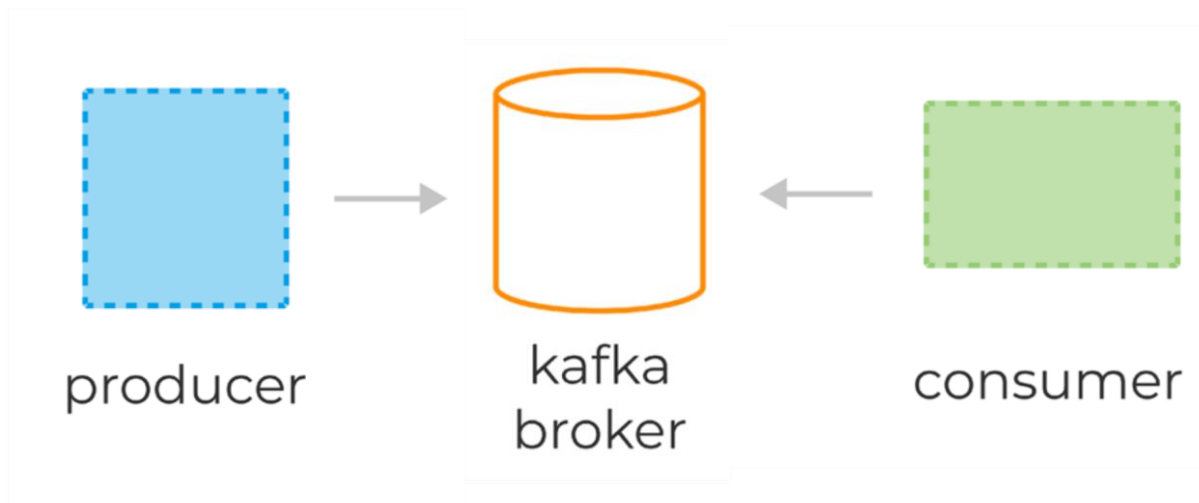
# Kafka vs. Queue





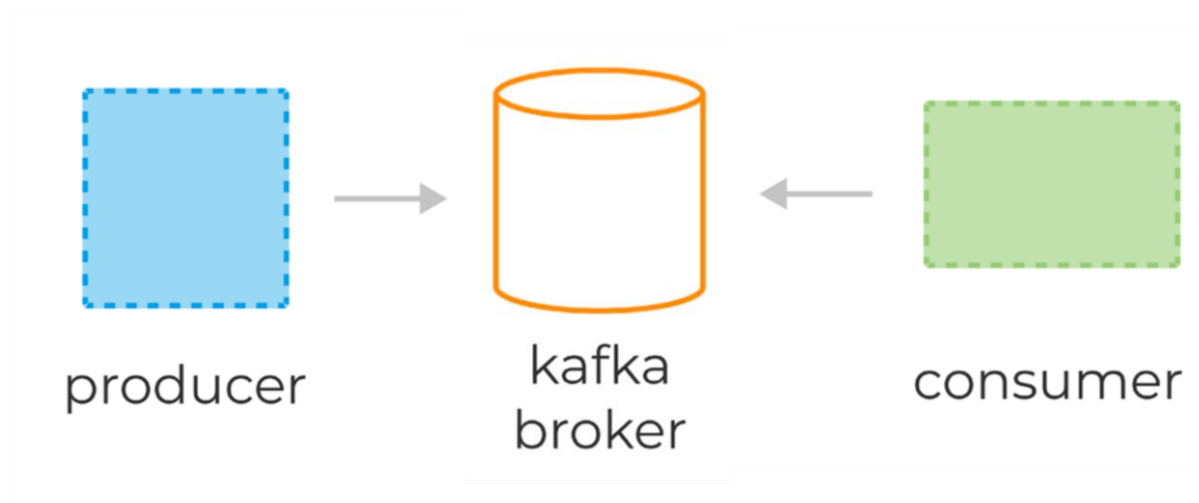
# Kafka vs. Queue

- Сообщения в Кафке **не удаляются** по мере их обработки консюмерами

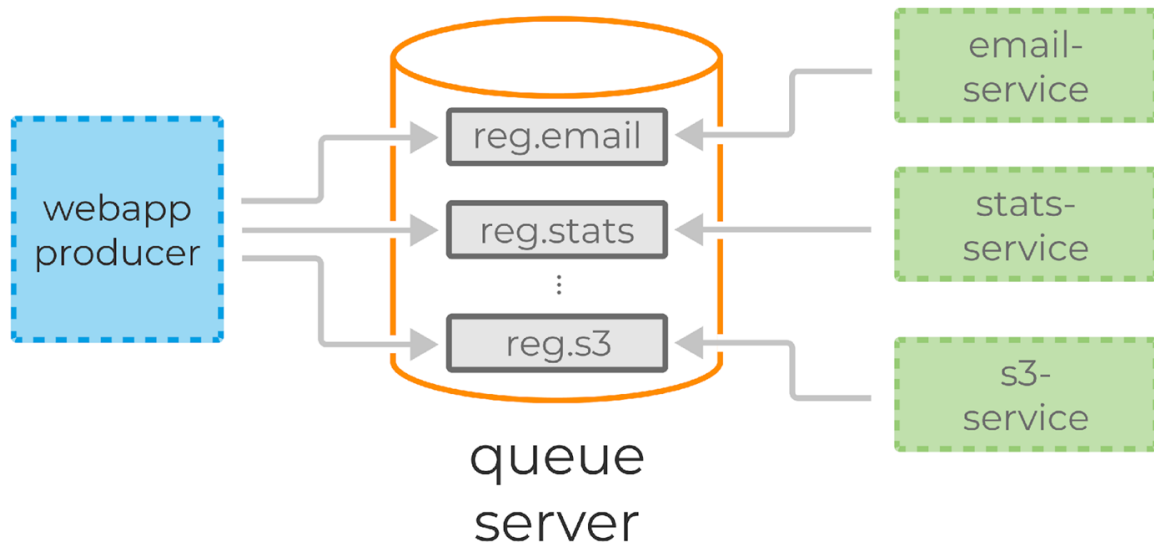


# Kafka vs. Queue

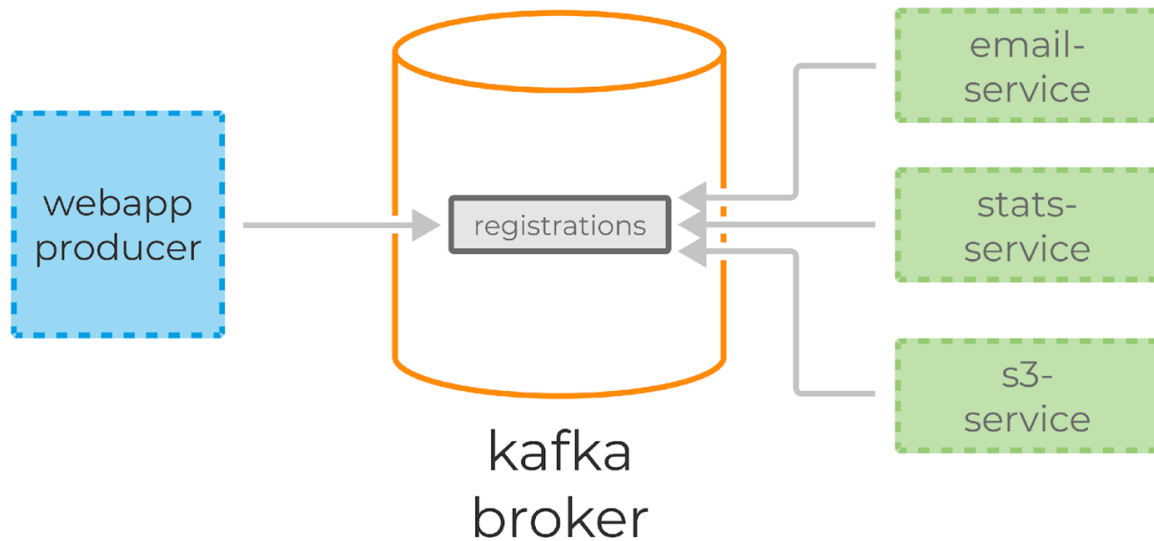
- Сообщения в Кафке **не удаляются** по мере их обработки консюмерами
- Одни и те же сообщения могут быть обработаны **сколько угодно раз**



# Kafka vs. Queue




# Kafka vs. Queue



# Структура Данных

# Структура Данных

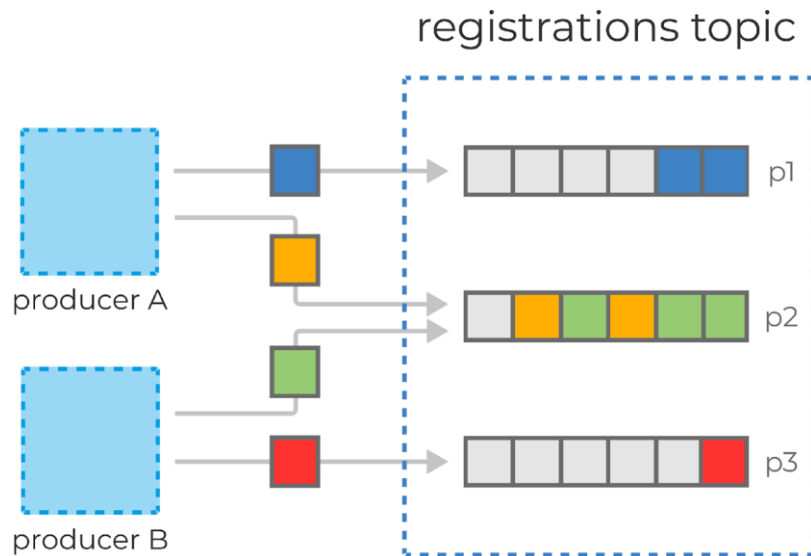
- **Key:** "Alice"
- **Value:** "Registered on our website"
- **Timestamp:** "Jun. 25, 2020 at 2:06 p.m." (*CreateTime, LogAppendTime*)
- **Headers:** [{"X-Generated-By": "web-host-12.eu-west2.slurm.io"}]



Крайне полезны в отладке,  
аудите и мониторинге!

# Структура Данных

- Сообщения отправляются в топики (*Topics*), каждый топик состоит из 1-й и более партиций (*Partition*)



# Структура Данных

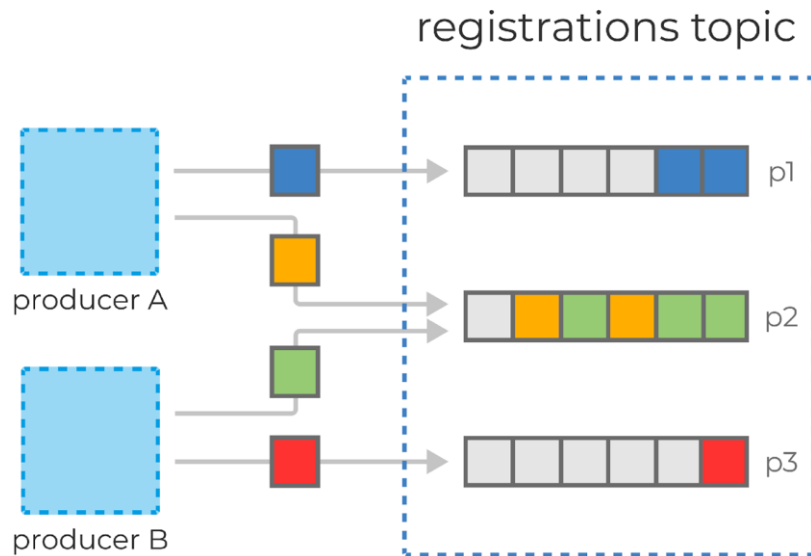
- Сообщения отправляются в топики (*Topics*), каждый топик состоит из 1-й и более партиций (*Partition*)
- Сообщения с одинаковыми ключами (*Key*) записываются в одну и ту же партицию — *MurmurHash*. Если ключ отсутствует — *RoundRobin*





# Структура Данных

- Сообщения отправляются в топики (*Topics*), каждый топик состоит из 1-й и более партиций (*Partition*)
- Сообщения с одинаковыми ключами (*Key*) записываются в одну и ту же партицию — *MurmurHash*. Если ключ отсутствует — *RoundRobin*
- Кафка гарантирует очередность записи и чтения в рамках **одной партиции**

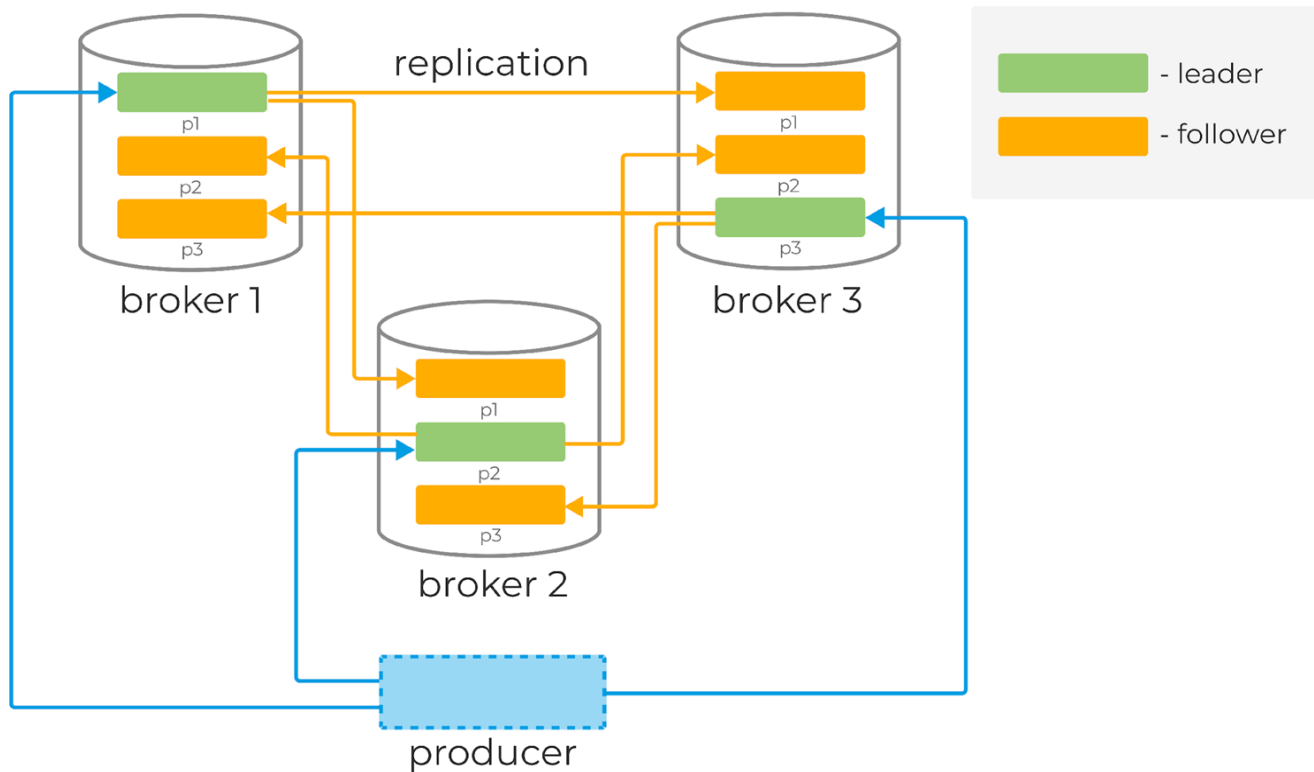


# Структура Данных

- Партиция — распределенный отказоустойчивый лог (*Log*)
- У каждой партиции есть 1 брокер лидер (*Leader*)
- У лидера может быть 0...N фолловеров (*Follower*)
- Сообщения всегда отправляются лидеру и, в общем случае, читаются с лидера

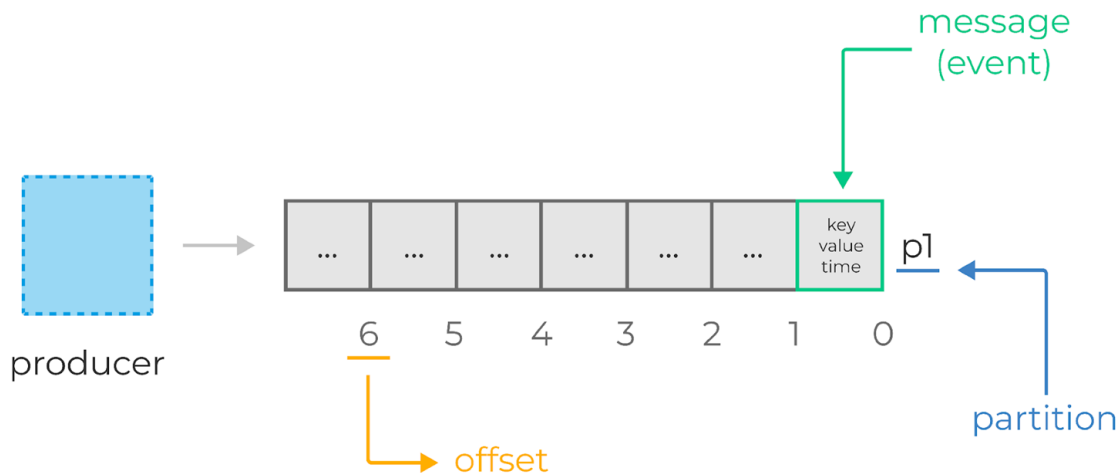


# Структура Данных



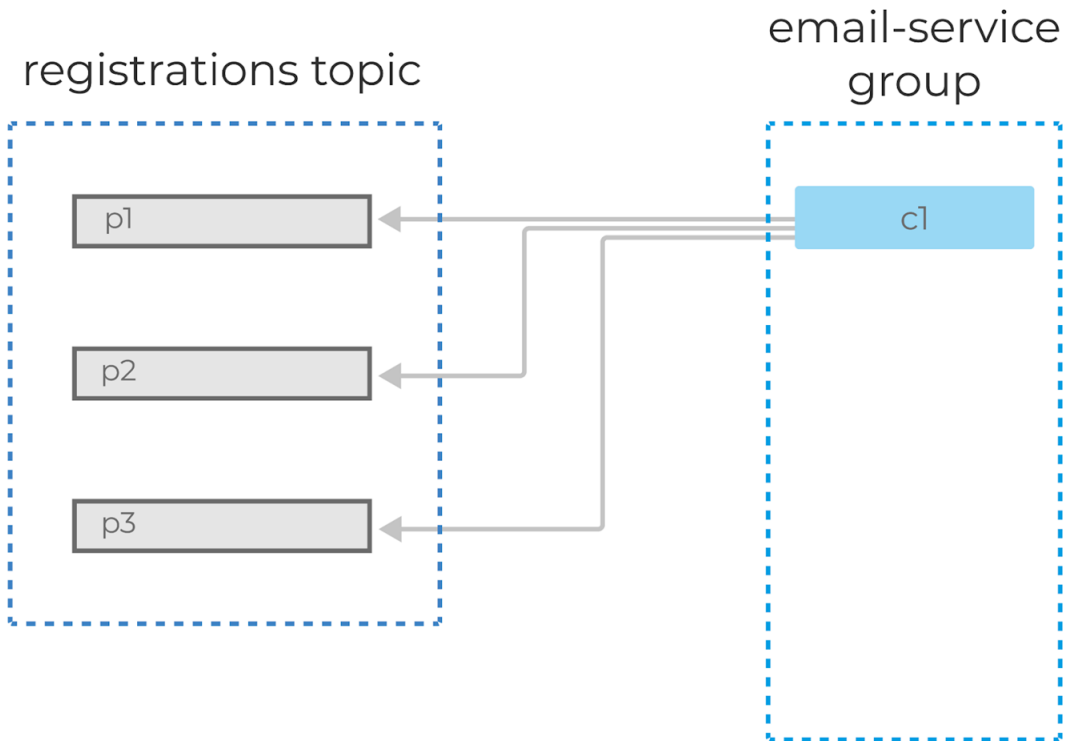
# Структура Данных

- Каждому записанному сообщению назначается **offset** — уникальный, монотонно возрастающий 64-bit unsigned int
- Данные удаляются согласно заданной конфигурации ретеншена (retention):
  - retention.ms — минимальное время хранения сообщений
  - retention.bytes — максимальный размер партии

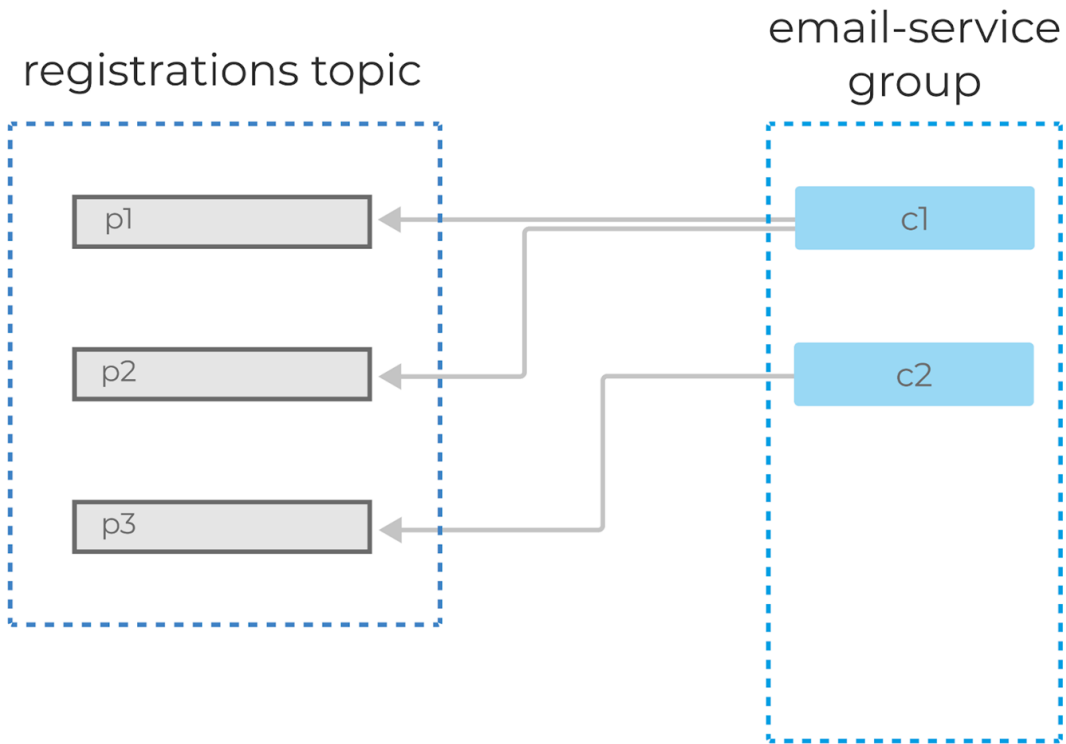


# Consumer Groups

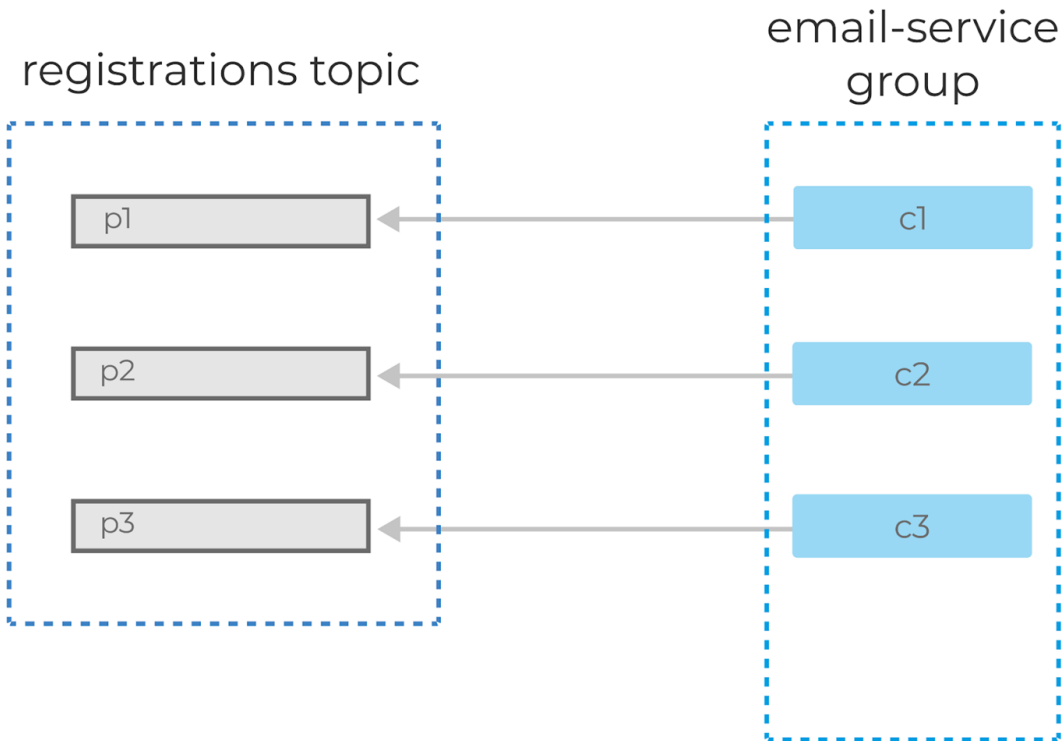
# Consumer Groups



# Consumer Groups

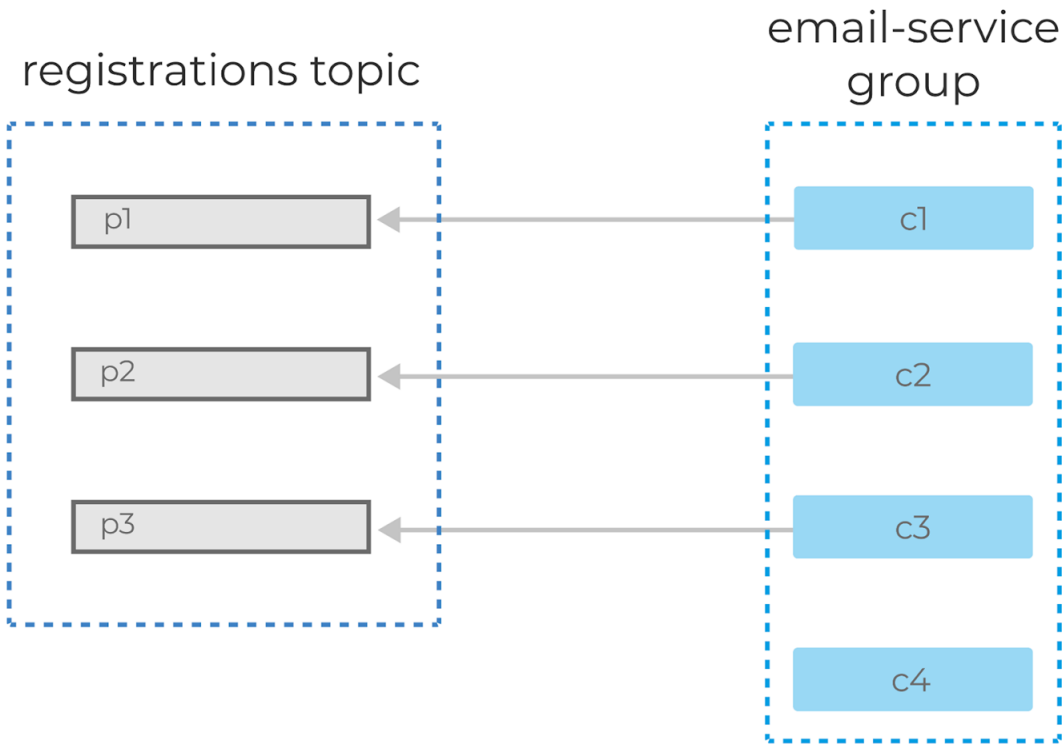


# Consumer Groups





# Consumer Groups



# Consumer Groups

- Партии внутри одной группы назначаются консюмерам **уникально**
- Партии — это наш **основной инструмент масштабирования**
- Если консюмеры не справляются с объемом данных, или необходимо распределить нагрузку между брокерами — **добавляем партии в топик**



# Consumer Groups

- Партии можно добавить в любой момент, **но**:
  - Нужно помнить про гарантию очередности в рамках одной партии
  - Индивидуальные партии нельзя удалить после создания

# Consumer Groups

- Партии можно добавить в любой момент, **но**:
  - Нужно помнить про гарантию очередности в рамках одной партии
  - Индивидуальные партии нельзя удалить после создания
- Нужно помнить про конфигурацию **auto.offset.reset** в консюмерах: при добавлении новой партии “на проде” вы наверняка захотите прочитать данные с начала лога (*auto.offset.reset=earliest*)!

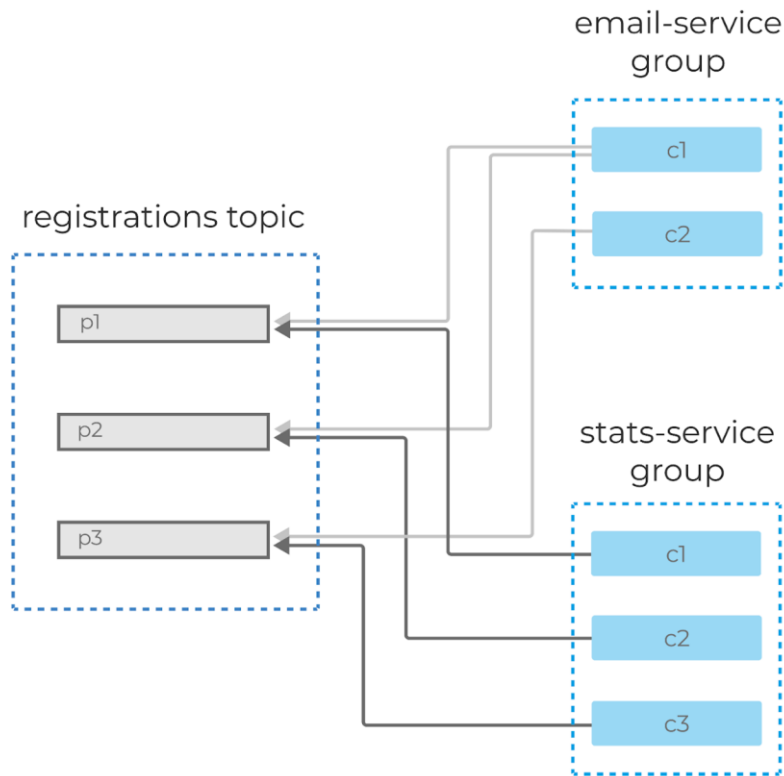
# Consumer Groups

- Партии можно добавить в любой момент, **но**:
  - Нужно помнить про гарантию очередности в рамках одной партии
  - Индивидуальные партии нельзя удалить после создания
- Нужно помнить про конфигурацию **auto.offset.reset** в консюмерах: при добавлении новой партии “на проде” вы наверняка захотите прочитать данные с начала лога (*auto.offset.reset=earliest*)!

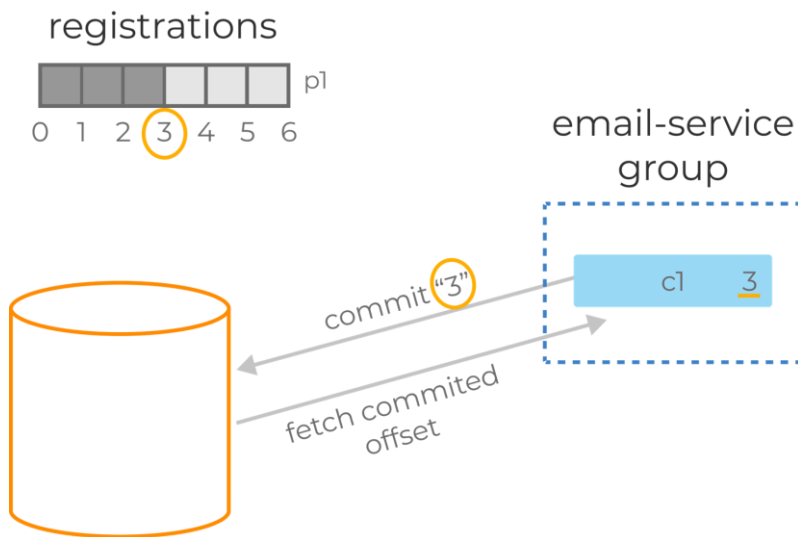
Партии не “бесплатны”! Каждая увеличивает время старта брокера и выбора лидеров после падения. Теоретический лимит на кластер:

**200.000 партий** для Кафки 2.0+

# Consumer Groups



# Consumer Groups

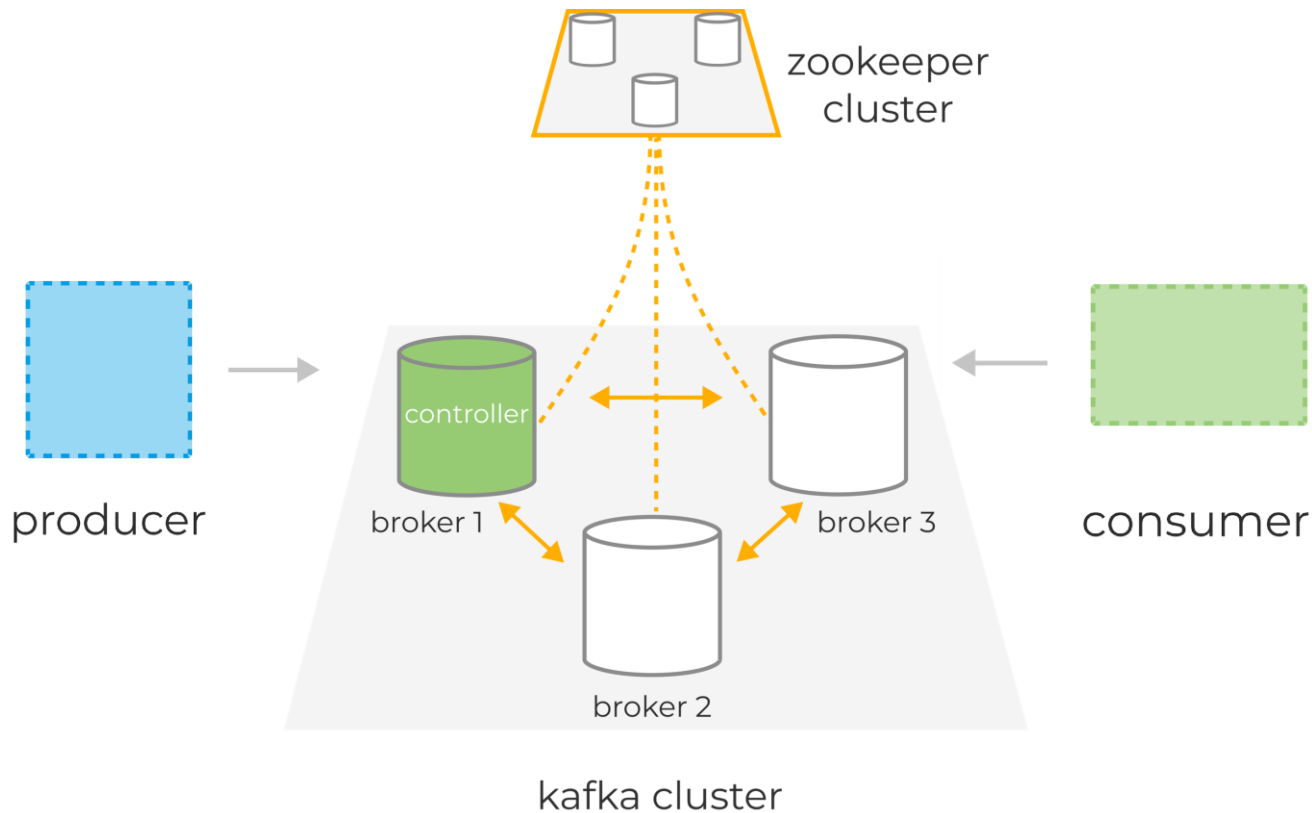


topic	partition	group	offset
registrations	p1	email-service	3
registrations	p1	stats-service	5
...			

# Apache ZooKeeper



# Apache ZooKeeper



# Урок 2. Базовые Основы

## Итог

- Разобрались чем Кафка отличается от популярных систем обмена сообщениями
  - Кафка не удаляет данные по мере их обработки
  - Позволяет читать одни и те же данные сколько угодно раз, в том числе несколькими сервисами одновременно
- Поняли как Кафка хранит данные и обеспечивает гарантию сохранности
  - Топики, партиции, оффсеты
  - Репликация
- Рассмотрели как записываются и читаются данные
  - Продюсеры
  - Консьюмеры и группы