# Урок 3. Первая Практика


# Цели

- Получить фундаментальные практические знания

# Цели

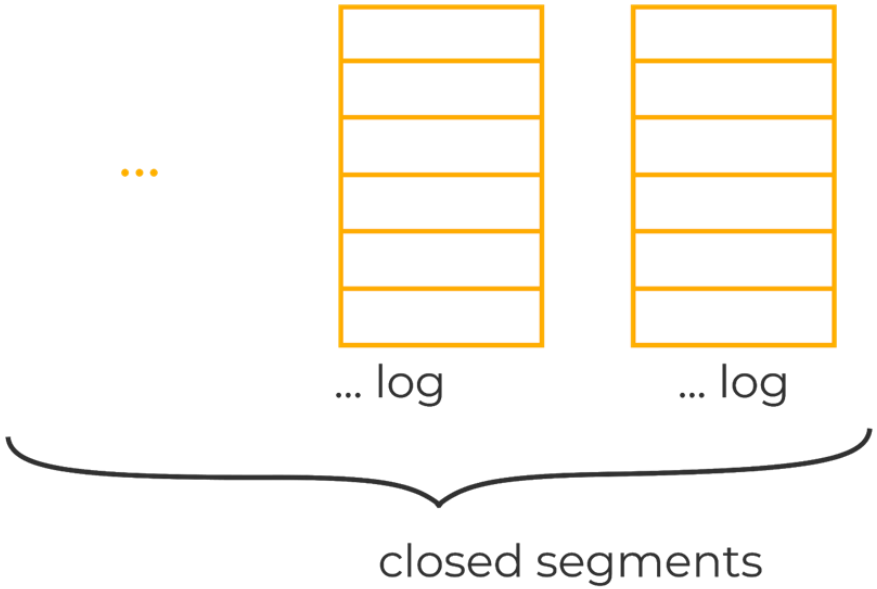
- Получить фундаментальные практические знания
- Установить и запустить кластер кафки состоящий из одного брокера

# Цели

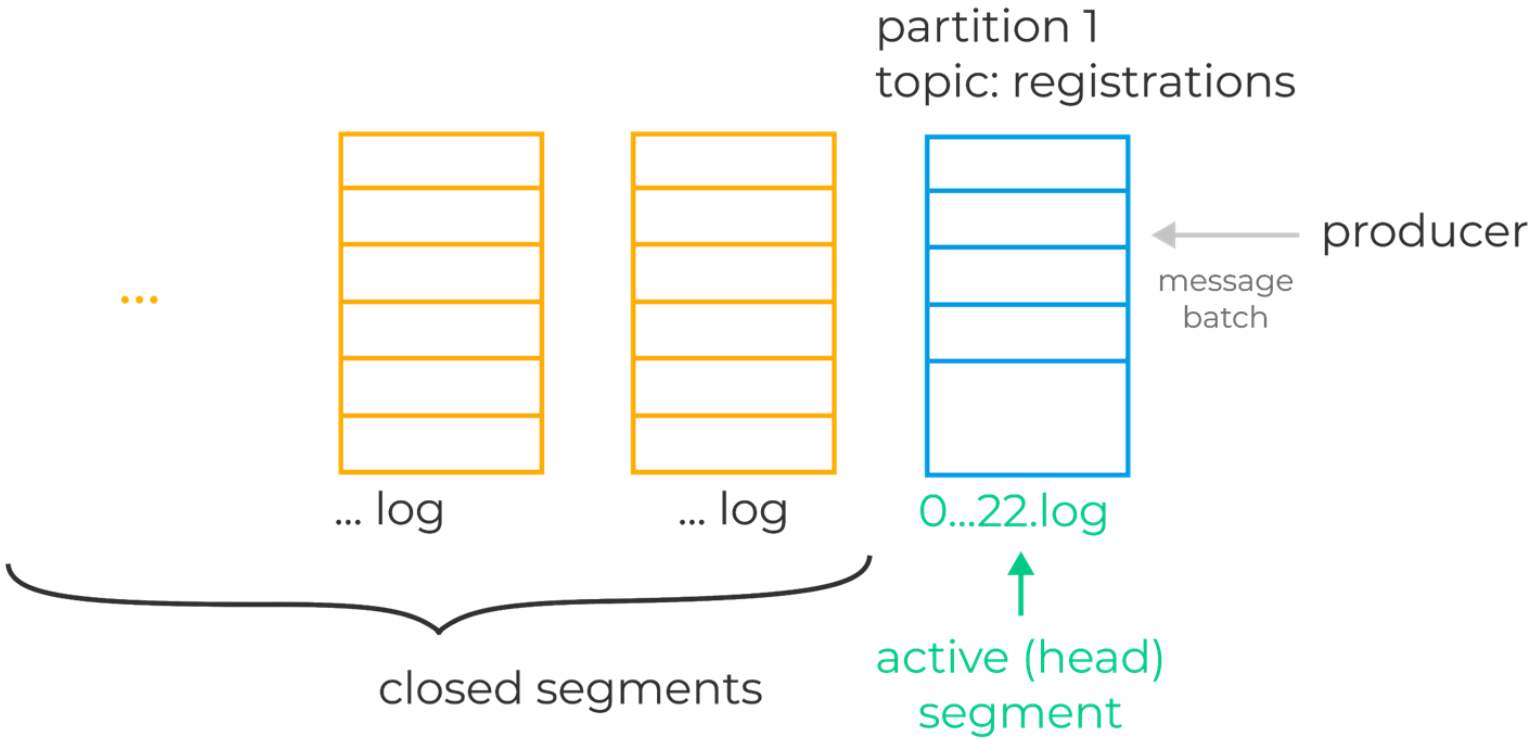
- Получить фундаментальные практические знания
- Установить и запустить кластер кафки состоящий из одного брокера
- Записать и прочитать данные используя встроенный функционал

# Структура Партиции

# Структура Партиции



# Структура Партиции



# Структура Партиции

- Сегменты удаляет отдельный тред в брокере

# Структура Партиции

- Сегменты удаляет отдельный тред в брокере
- **retention.ms** — минимальное время хранения данных, после которого Кафка может их удалить

# Структура Партиции

- Сегменты удаляет отдельный тред в брокере
- **retention.ms** — минимальное время хранения данных, после которого Кафка может их удалить
- **retention.bytes** — максимальный размер партиции на диске

# Структура Партиции

- Сегменты удаляет отдельный тред в брокере
- **retention.ms** — минимальное время хранения данных, после которого Кафка может их удалить
- **retention.bytes** — максимальный размер партиции на диске
- **segment.ms** — период роллапа сегмента после открытия *(по-умолчанию 1 неделя)*

# Структура Партиции

- Сегменты удаляет отдельный тред в брокере
- **retention.ms** — минимальное время хранения данных, после которого Кафка может их удалить
- **retention.bytes** — максимальный размер партиции на диске
- **segment.ms** — период роллапа сегмента после открытия *(по-умолчанию 1 неделя)*
- **segment.bytes** — максимальный размер сегмента *(по-умолчанию 1ГБ)*

# Структура Партиции

Большая часть настроек Кафки может быть определена на 2-х уровнях:

1. **broker-level config** — уровень сервера, используется по-умолчанию *(часто имеют префикс log.*)*

# Структура Партиции

Большая часть настроек Кафки может быть определена на 2-х уровнях:

1. **broker-level config** — уровень сервера, используется по-умолчанию *(часто имеют префикс log.*)*

2. **topic-level config** — оверрайды для отдельных топиков, имеют более высокий приоритет

# Структура Партиции

Примеры:

1. **log.retention.ms** — глобальный дефолт ретеншена для всех топиков, задается в server.properties

# Структура Партиции

Примеры:

1. **log.retention.ms** — глобальный дефолт ретеншена для всех топиков, задается в server.properties

2. **retention.ms** — ретеншен для отдельно взятого топика, задается через kafka-configs.sh или AdminClient

# Структура Партиции

Примеры:

1. **log.retention.ms** — глобальный дефолт ретеншена для всех топиков, задается в server.properties

2. **retention.ms** — ретеншен для отдельно взятого топика, задается через kafka-configs.sh или AdminClient

Полный перечень настроек здесь: https://kafka.apache.org/documentation/#configuration

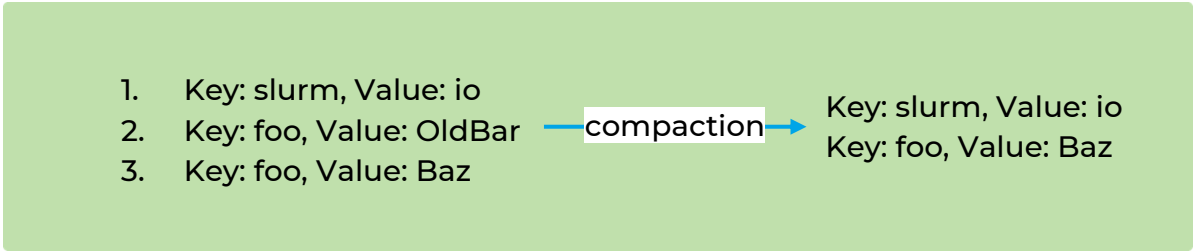
# Log Compaction

# Log Compaction

**cleanup.policy** — *delete* для ретеншена по времени/размеру (включен по-умолчанию), *compact* для включения compaction

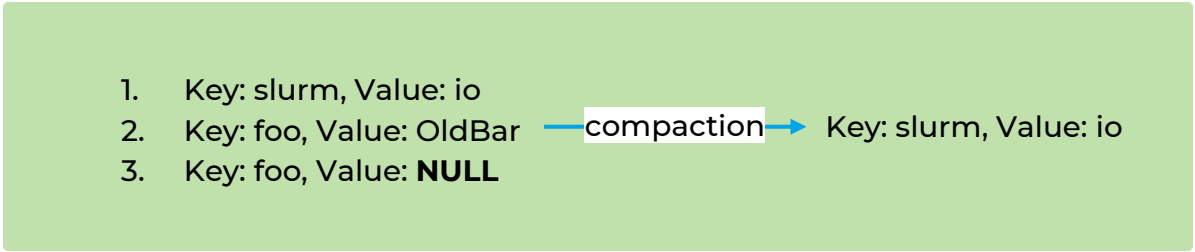
# Log Compaction

**cleanup.policy** — *delete* для ретеншена по времени/размеру (включен по-умолчанию), *compact* для включения compaction



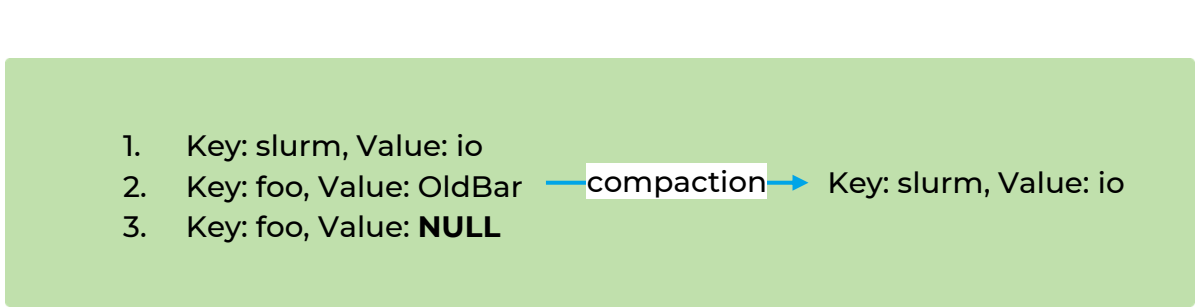
# Log Compaction

**cleanup.policy** — *delete* для ретеншена по времени/размеру (включен по-умолчанию), *compact* для включения compaction



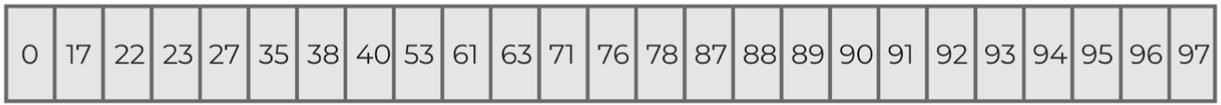
# Log Compaction

**cleanup.policy** — *delete* для ретеншена по времени/размеру (включен по-умолчанию), *compact* для включения compaction

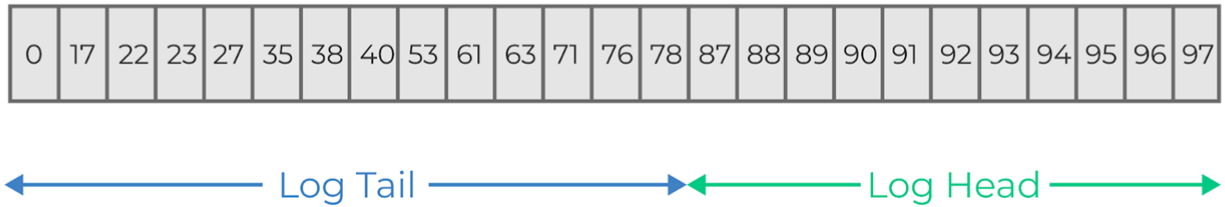



Могут быть включены одновременно: *cleanup.policy=compact,delete*

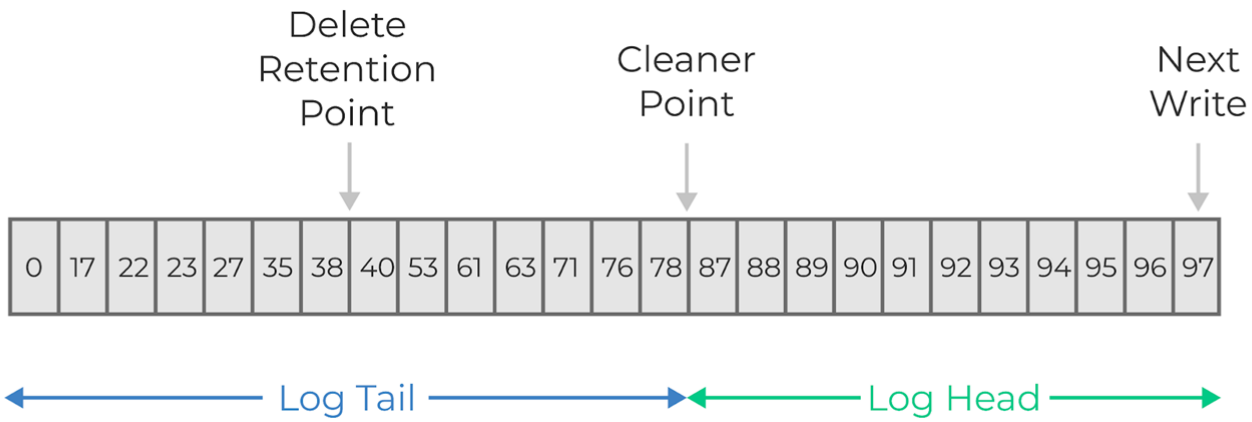
# Log Compaction



# Log Compaction



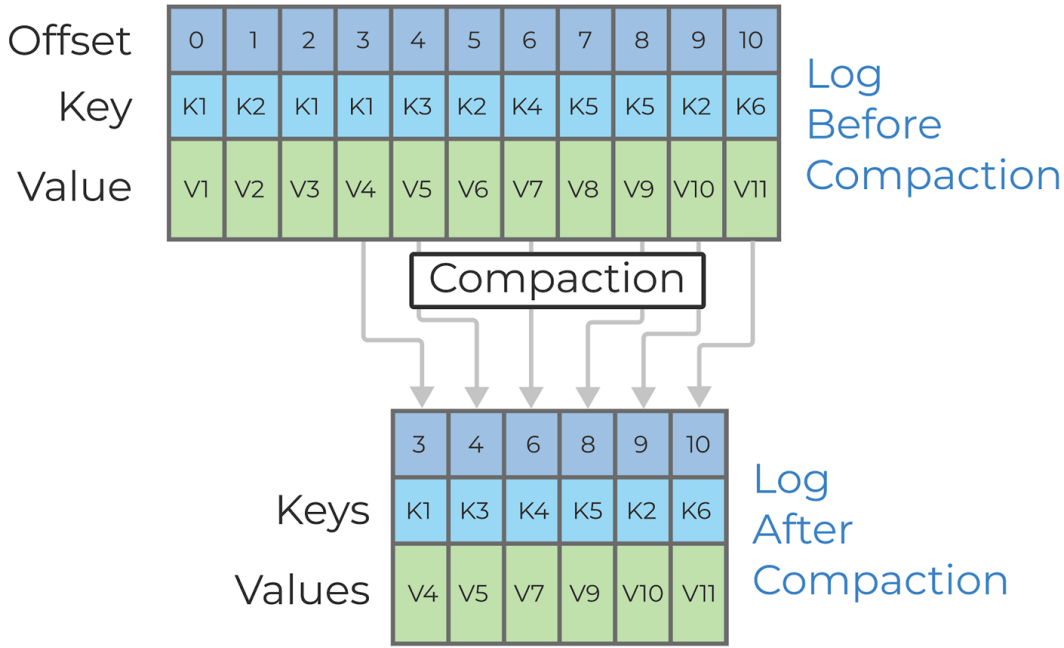
# Log Compaction



# Log Compaction


Log
Before
Compaction

# Log Compaction



# Log Compaction

- Довольно **трудоемкий** процесс, нагружает память, процессор и диск

# Log Compaction

- Довольно **трудоемкий** процесс, нагружает память, процессор и диск
- **Не атомарен**! Внутри партиции по-прежнему могут одновременно находиться несколько записей с одинаковым ключом

# Log Compaction

- Довольно **трудоемкий** процесс, нагружает память, процессор и диск
- **Не атомарен**! Внутри партиции по-прежнему могут одновременно находиться несколько записей с одинаковым ключом
- **Оффсеты не меняются**, порядок записей остается прежним

# Log Compaction

- Довольно **трудоемкий** процесс, нагружает память, процессор и диск
- **Не атомарен**! Внутри партиции по-прежнему могут одновременно находиться несколько записей с одинаковым ключом
- **Оффсеты не меняются**, порядок записей остается прежним
- Позволяет **“удалять” записи по ключу**, хорошо подходит для снэпшоттинга и восстановления последнего состояния системы после падения/перезагрузки

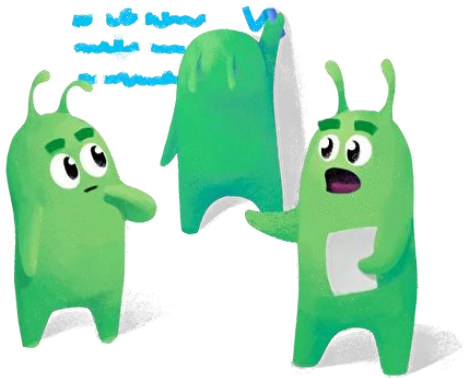
# Log Compaction

- Довольно **трудоемкий** процесс, нагружает память, процессор и диск
- **Не атомарен**! Внутри партиции по-прежнему могут одновременно находиться несколько записей с одинаковым ключом
- **Оффсеты не меняются**, порядок записей остается прежним
- Позволяет **“удалять” записи по ключу**, хорошо подходит для снэпшоттинга и восстановления последнего состояния системы после падения/перезагрузки

Живой пример такого приложения: Confluent Schema Registry (https://github.com/confluentinc/schema-registry)

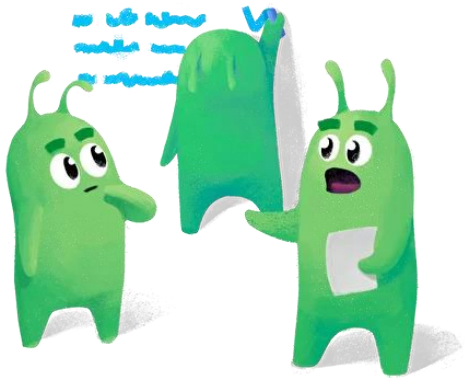
# Итоги

- Получили фундаментальные практические знания



# Итоги

- Получили фундаментальные практические знания
- Установили и запустили кластер кафки состоящий из одного брокера



# Итоги

- Получили фундаментальные практические знания
- Установили и запустили кластер кафки состоящий из одного брокера
- Записали и прочитали данные используя встроенный функционал