

Урок 3. Первая Практика



Цели

- Получить фундаментальные практические знания



Цели

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



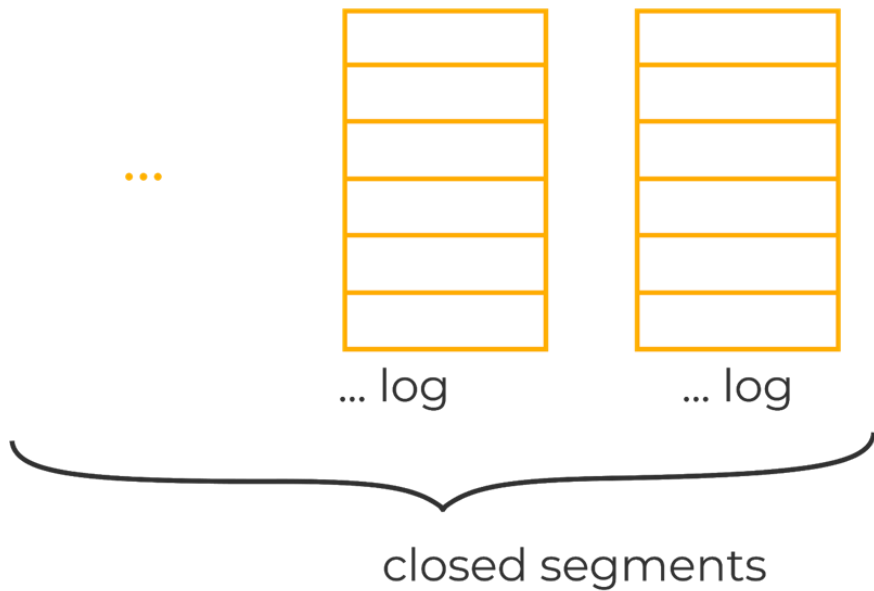
Цели

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

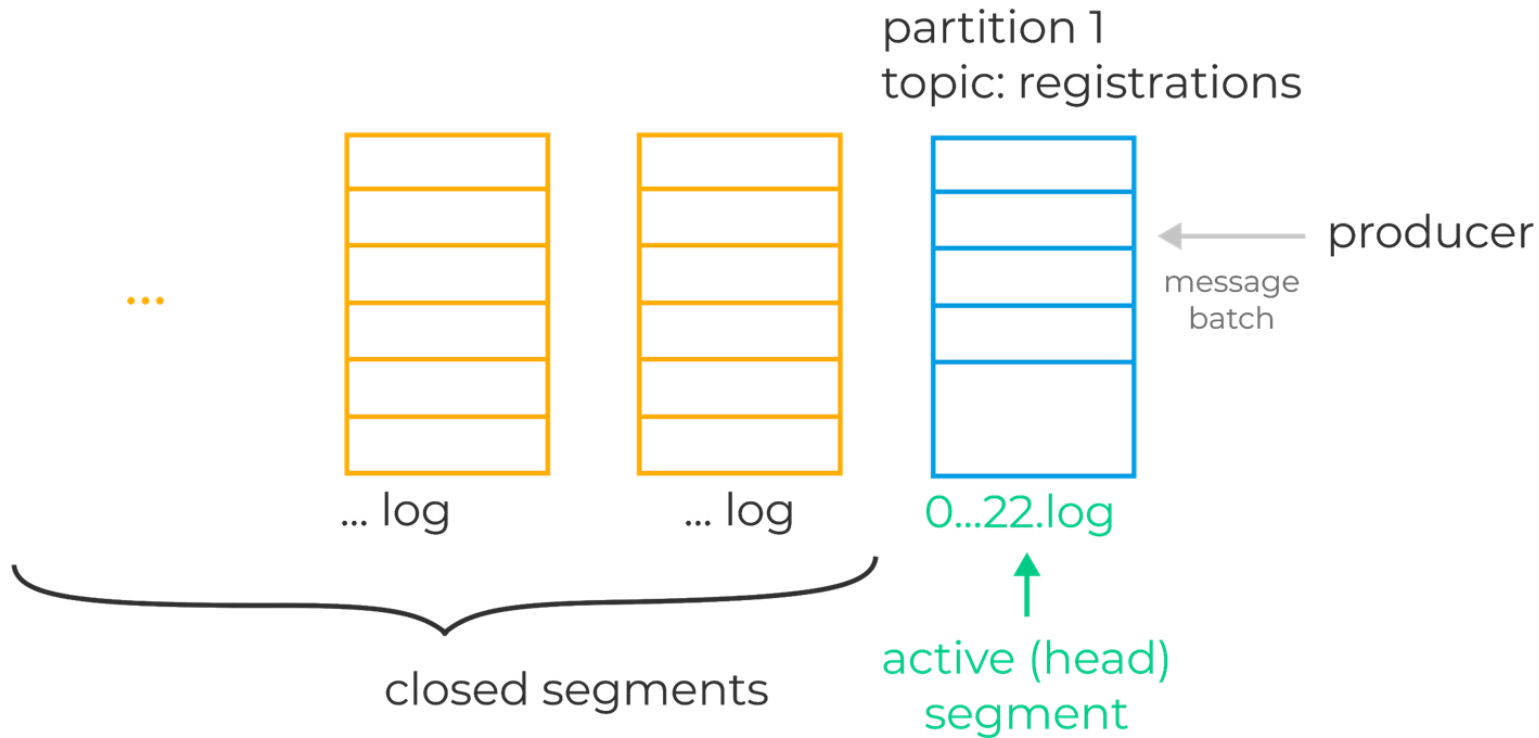


Структура Партиции

Структура Партиции



Структура Партиции



Структура Партии

■ Сегменты удаляет отдельный тред в брокере



Структура Партии



■ Сегменты удаляет отдельный тред в брокере

■ **retention.ms** — минимальное время хранения данных, после которого Кафка может их удалить

Структура Партии



- Сегменты удаляет отдельный тред в брокере

- **retention.ms** — минимальное время хранения данных, после которого Кафка может их удалить

- **retention.bytes** — максимальный размер партии 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 Партии

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

```
graph LR; subgraph "Before Compaction"; B1[1. Key: slurm, Value: io]; B2[2. Key: foo, Value: OldBar]; B3[3. Key: foo, Value: Baz]; end; subgraph "After Compaction"; A1[Key: slurm, Value: io]; A2[Key: foo, Value: Baz]; end; B2 -- compaction --> A2;
```

1. Key: slurm, Value: io
2. Key: foo, Value: OldBar
3. Key: foo, Value: Baz

→ compaction →

Key: slurm, Value: io
Key: foo, Value: Baz

Log Compaction

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

1. Key: slurm, Value: io
2. Key: foo, Value: OldBar → compaction → Key: slurm, Value: io
3. Key: foo, Value: **NULL**

Log Compaction

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

- 
1. Key: slurm, Value: io
 2. Key: foo, Value: OldBar → compaction → Key: slurm, Value: io
 3. Key: foo, Value: **NULL** → Key: slurm, Value: io

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

Log Compaction

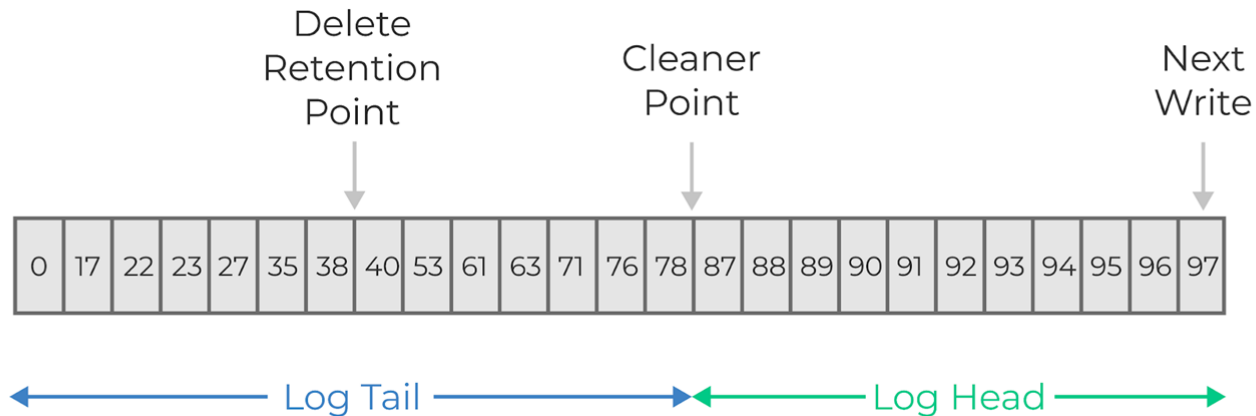
| | | | | | | | | | | | | | | | | | | | | | | | | |
|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 0 | 17 | 22 | 23 | 27 | 35 | 38 | 40 | 53 | 61 | 63 | 71 | 76 | 78 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 |
|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|

Log Compaction

| | | | | | | | | | | | | | | | | | | | | | | | | |
|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 0 | 17 | 22 | 23 | 27 | 35 | 38 | 40 | 53 | 61 | 63 | 71 | 76 | 78 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 |
|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|



Log Compaction

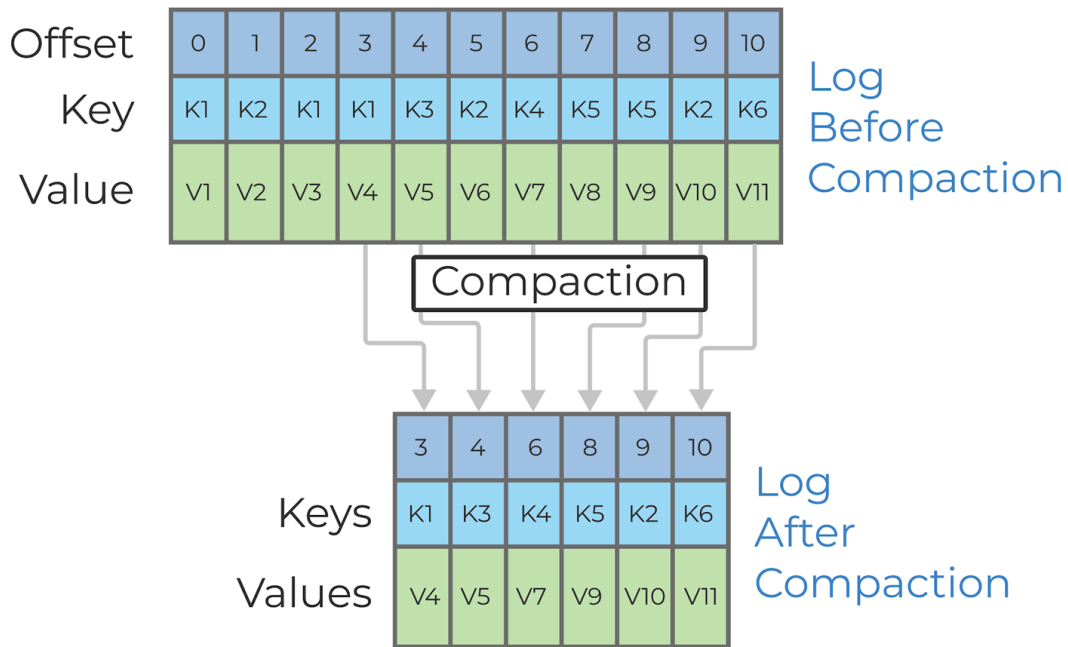


Log Compaction

| | | | | | | | | | | | |
|--------|----|----|----|----|----|----|----|----|----|-----|-----|
| Offset | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Key | K1 | K2 | K1 | K1 | K3 | K2 | K4 | K5 | K5 | K2 | K6 |
| Value | V1 | V2 | V3 | V4 | V5 | V6 | V7 | V8 | V9 | V10 | V11 |

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 партии 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 снапшоттинга и восстановления последнего состояния системы после падения/перезагрузки

Log Compaction

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

Не атомарен! Внутри партии 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 снапшоттинга 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

