Kappa и Lambda

Обзор баззвордов и архитектур

Кто я?

- Chief Data Officer @ Qvant
- Преподаватель @ New Professions Lab
- Data Engineer @ остальное время

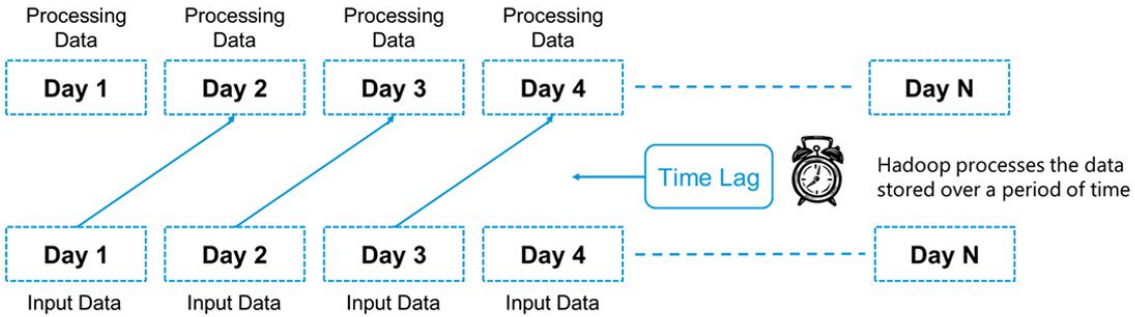
О чем сегодня поговорим

- Streaming vs Batch
- Lambda vs Kappa
- Нюансы реализации

Stream & Batch

Batch (what)

Batch processing - обработка данных, разбитых на непересекающиеся датасеты (чаще всего по времени).



Batch (why)

Плюсы

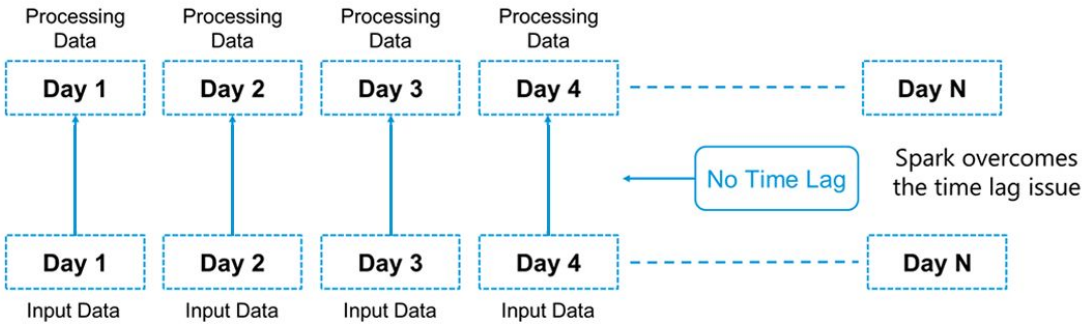
- ➕ Простой репроцессинг
- ➕ Высокая эффективность на больших объемах

Минусы

- ➖ Создает пиковые нагрузки
- ➖ Медленнее доставляет данные

Stream (what)

Stream processing - обработка данных как непрерывного потока в режиме реального времени.



Stream (why)

Плюсы

- ➕ Равномерная нагрузка
- ➕ Быстрая доставка данных

Минусы

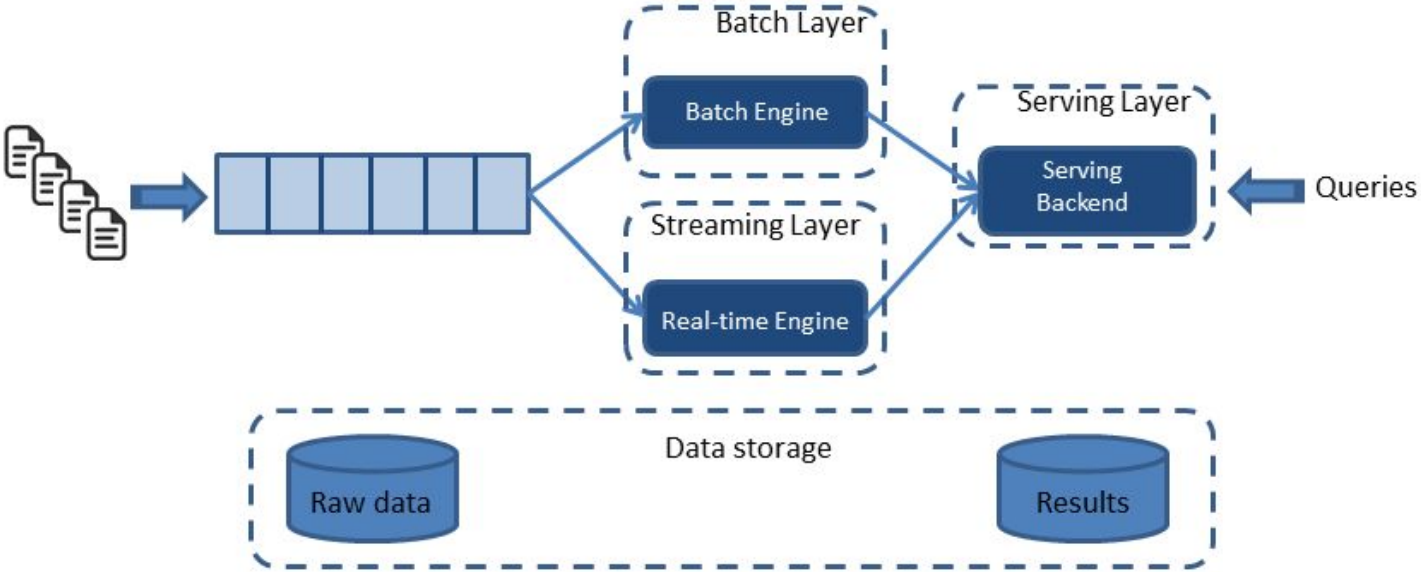
- ➖ Сложный репроцессинг
- ➖ Unbounded data

Параллельный расчет стрима и батча

Батч - лишь частный случай стрима

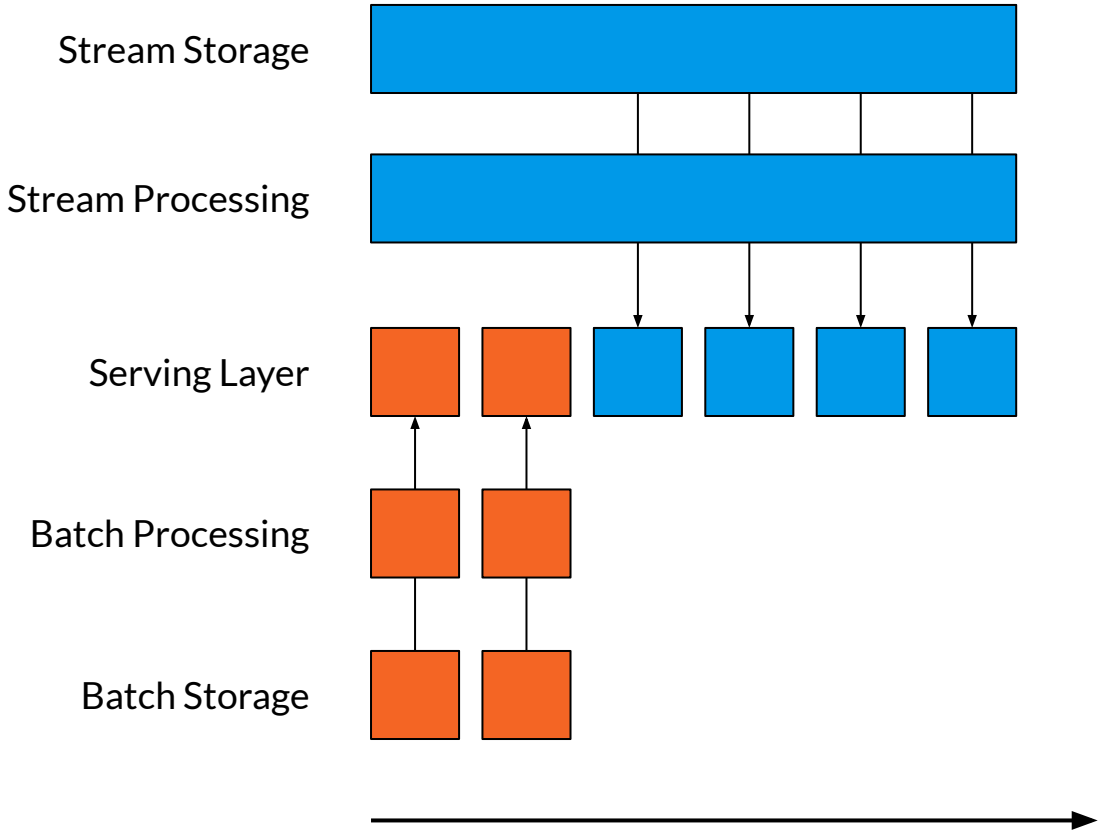
Lambda

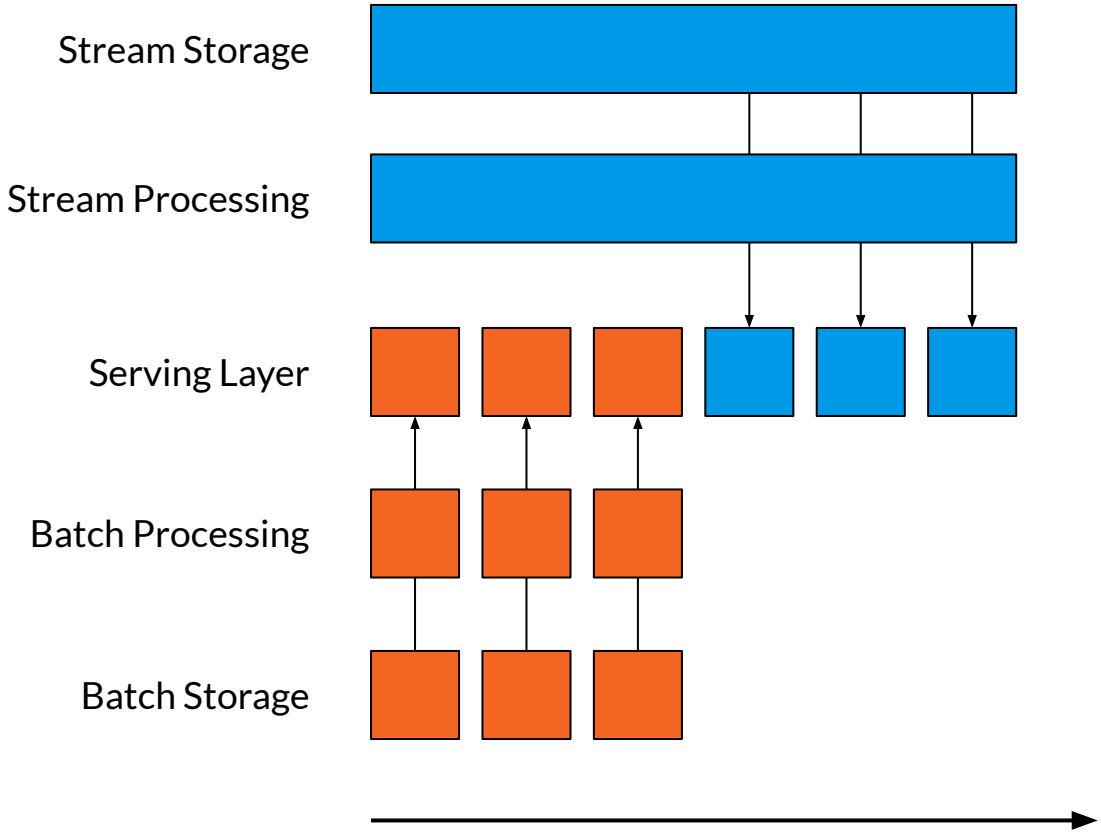
Lambda (what)

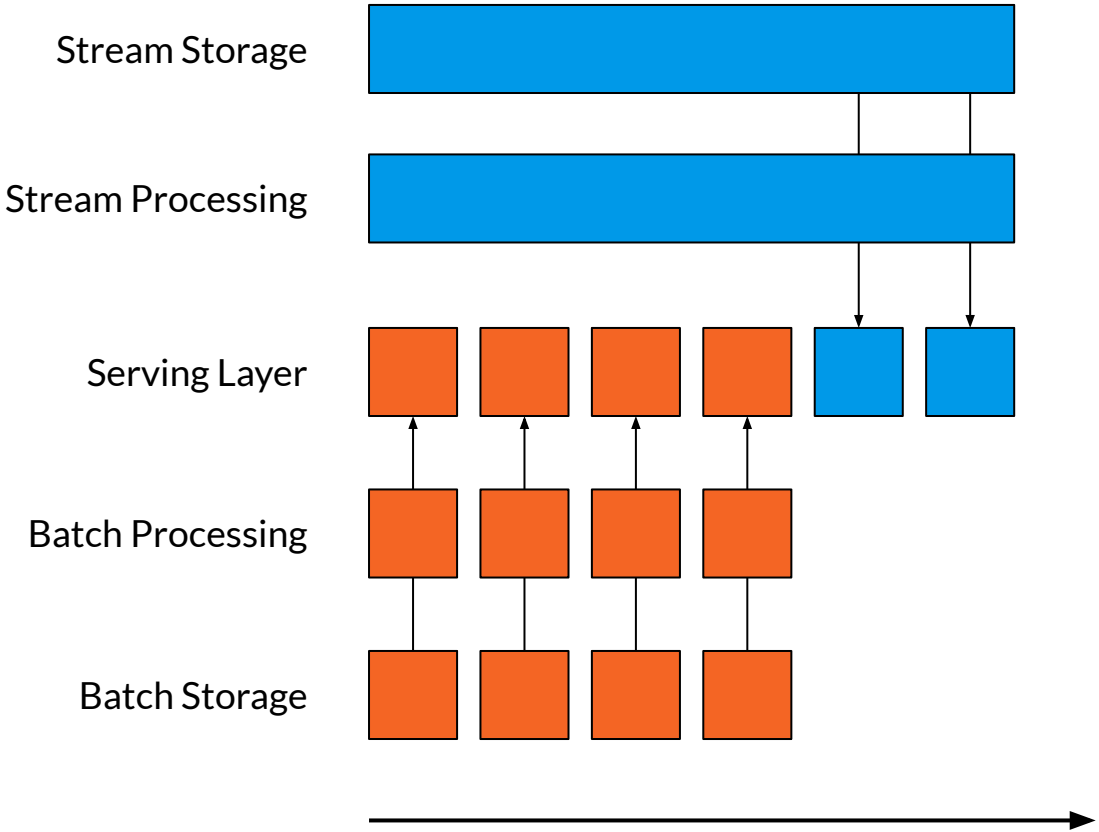

Batch Layer
Batch Engine
Serving Layer
Serving Backend
Queries
Streaming Layer
Real-time Engine
Data storage
Raw data
Results

Lambda (what)

- Batch Layer - надежная пакетная обработка
- Streaming Layer - быстрая потоковая обработка
- Serving Layer - абстракция для агрегатов из двух предыдущих слоев


Stream Storage
Stream Processing
Serving Layer
Batch Processing
Batch Storage


Stream Storage
Stream Processing
Serving Layer
Batch Processing
Batch Storage


Stream Storage
Stream Processing
Serving Layer
Batch Processing
Batch Storage

Lambda (why)

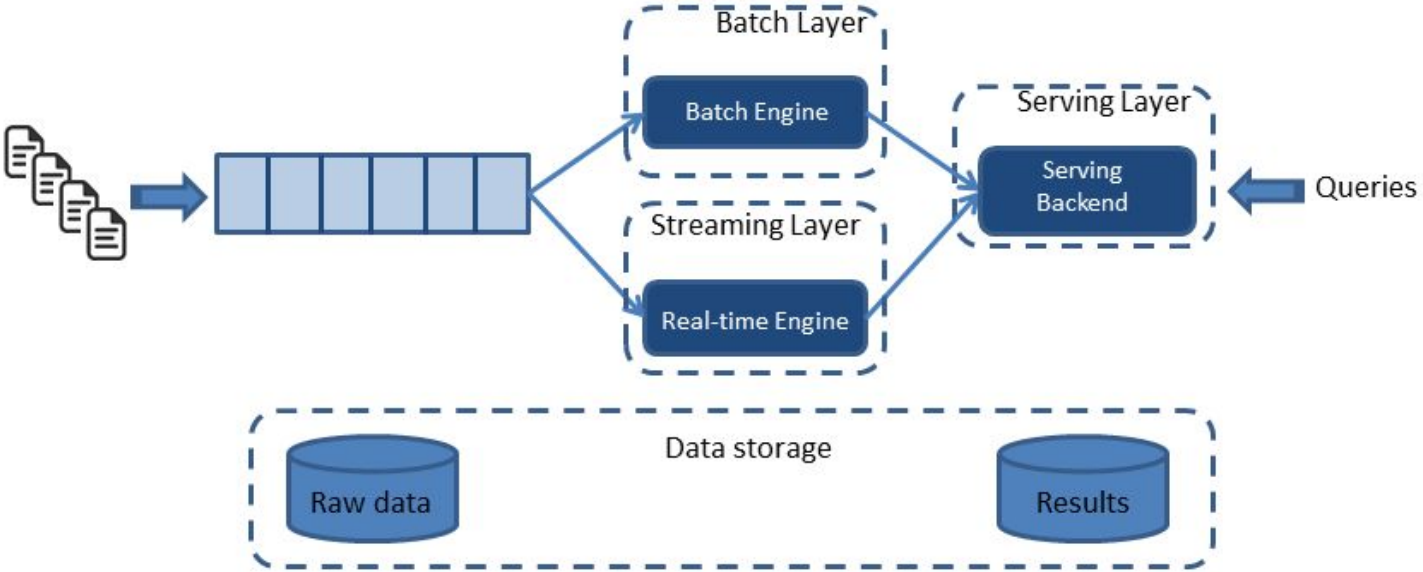
Плюсы

- ➕ Скорость стрима
- ➕ Надежность батча
- ➕ Простой репроцессинг

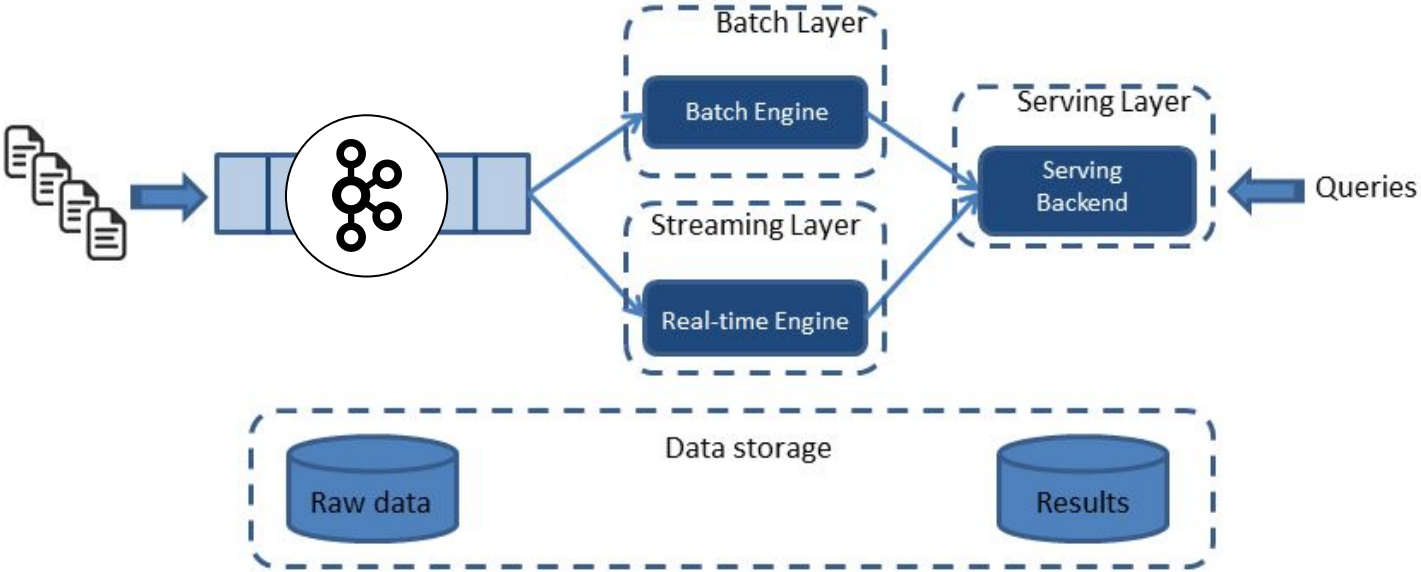
Минусы

- ➖ Дублирование логики
- ➖ Дублирование сервисов

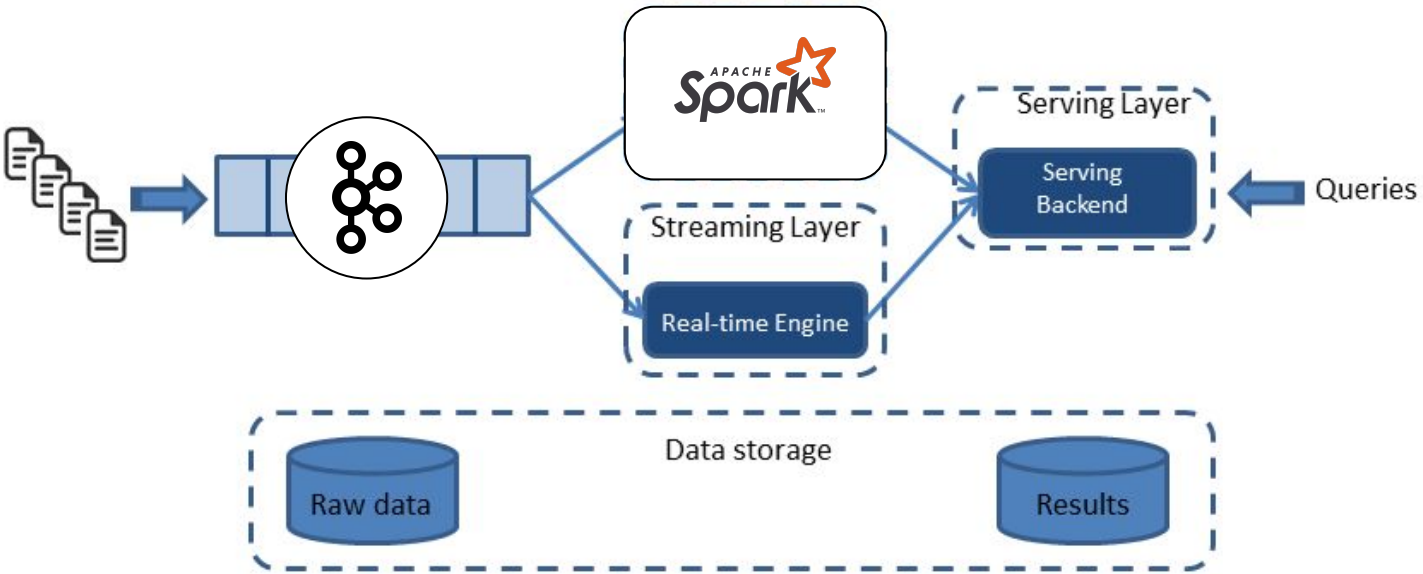
Lambda (example)



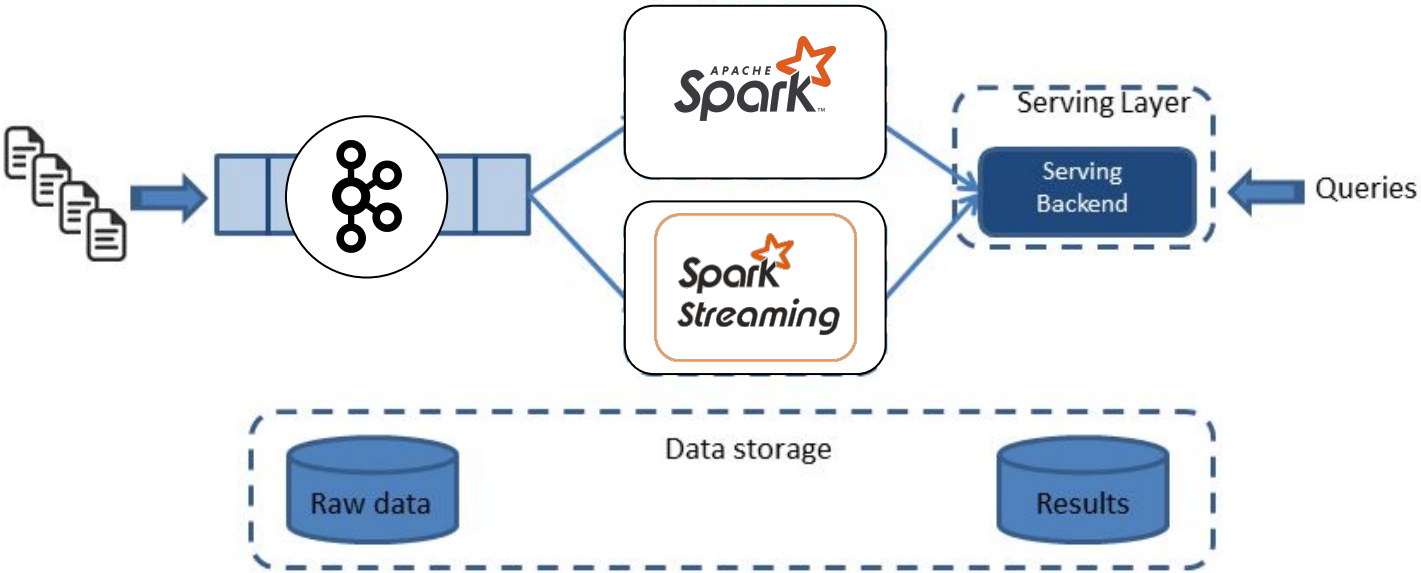
Lambda (example)



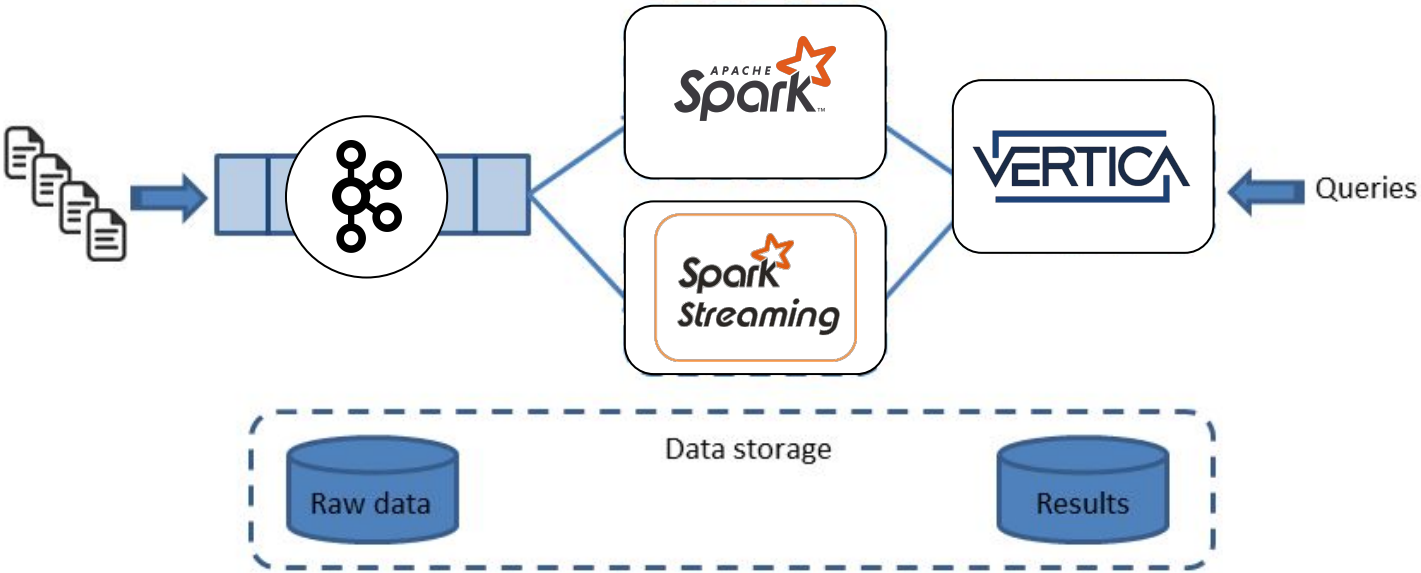
Lambda (example)



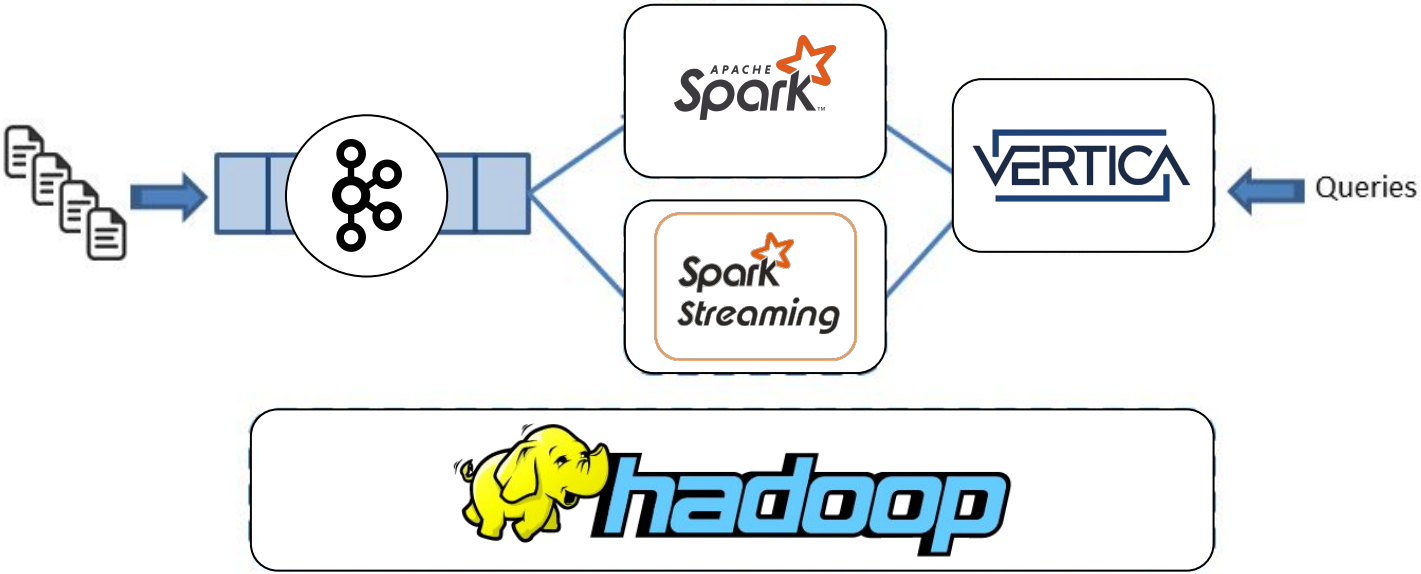
Lambda (example)



Lambda (example)



Lambda (example)

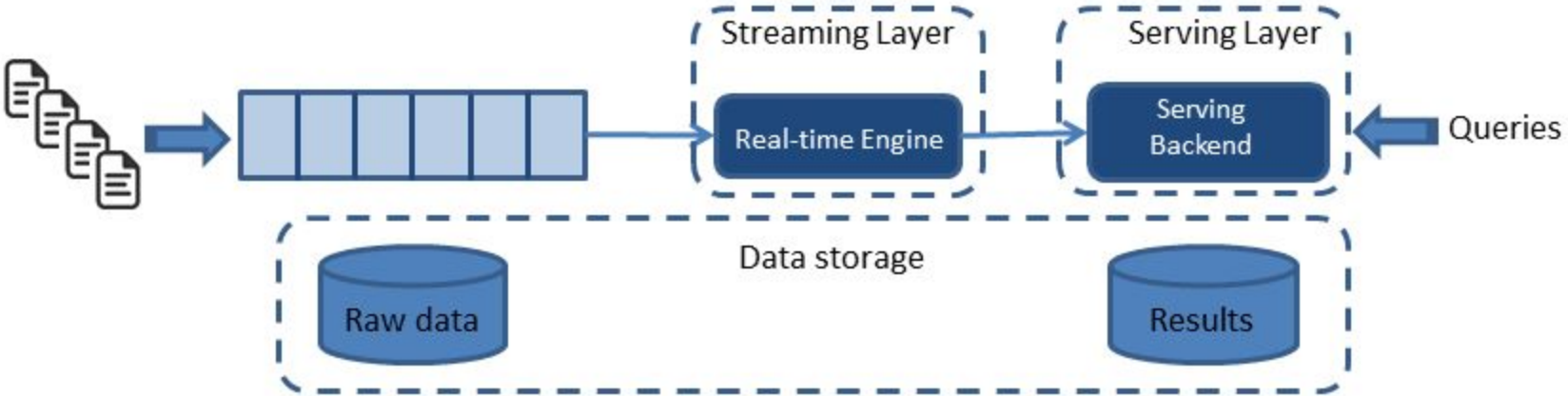


Lambda (fix)

- Apache Spark - схожий API для stream и batch
- Apache Beam - унифицированный API для stream и batch

Kappa

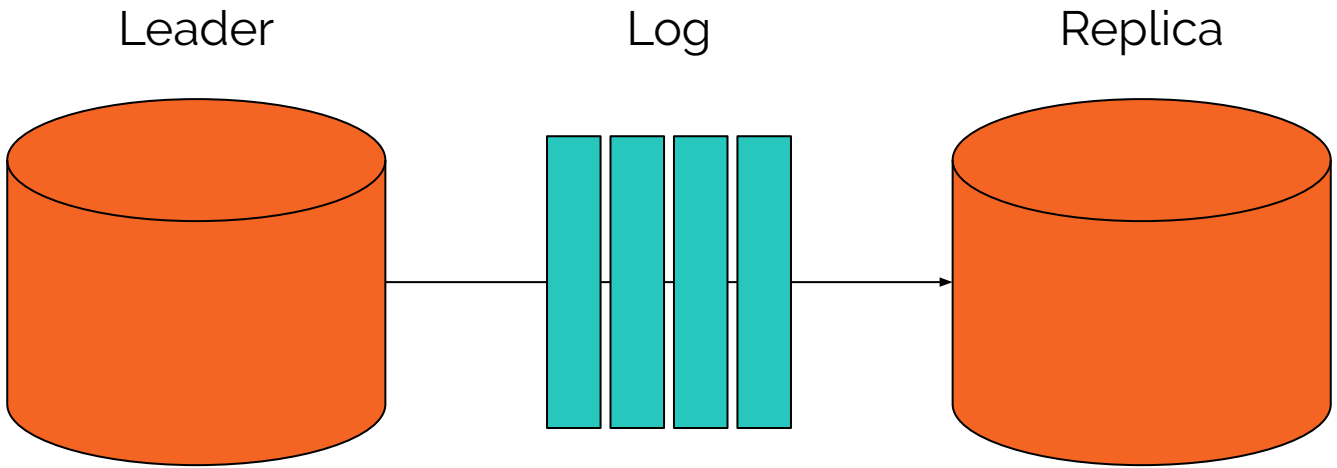
Kappa (what)


Streaming Layer
Serving Layer
Real-time Engine
Serving Backend
Queries
Data storage
Raw data
Results

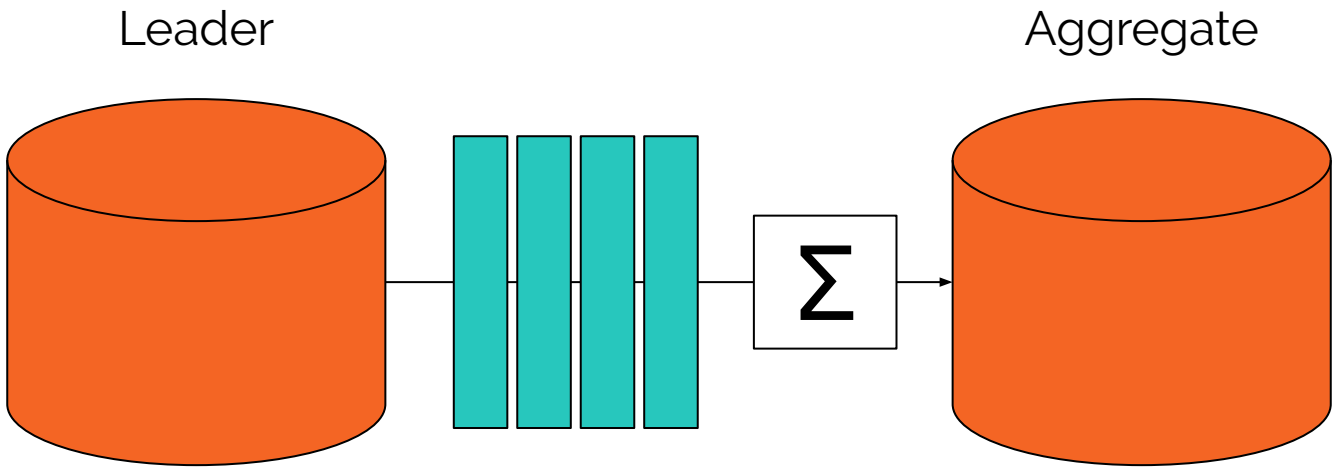
Kappa (what)

- Streaming Layer - потоковая обработка
- Serving Layer - хранилище агрегатов
- Raw data storage - хранилище сырых данных из потока

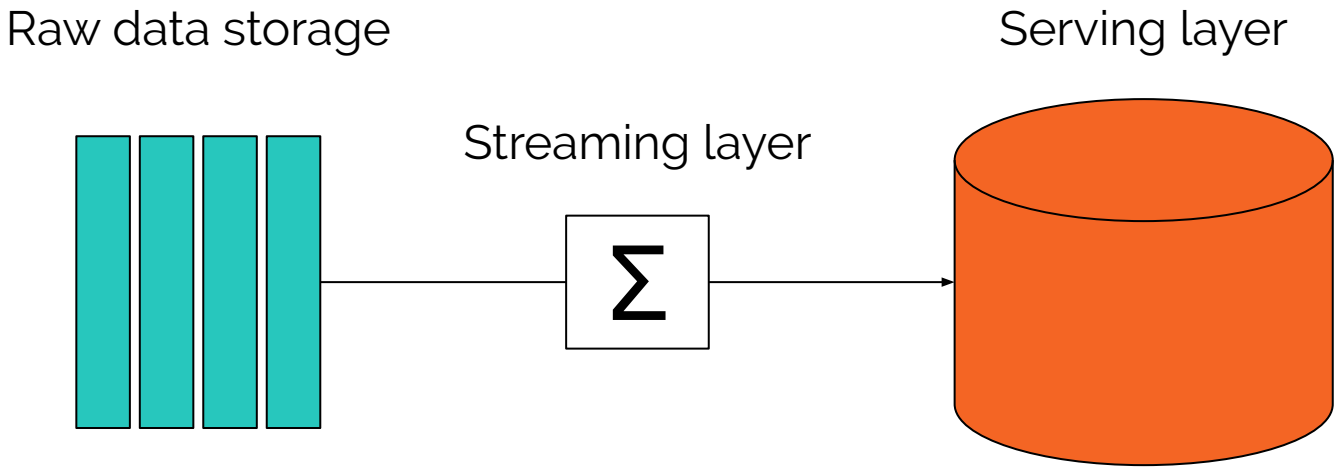
Kappa (what)



Kappa (what)



Kappa (what)



Kappa (why)

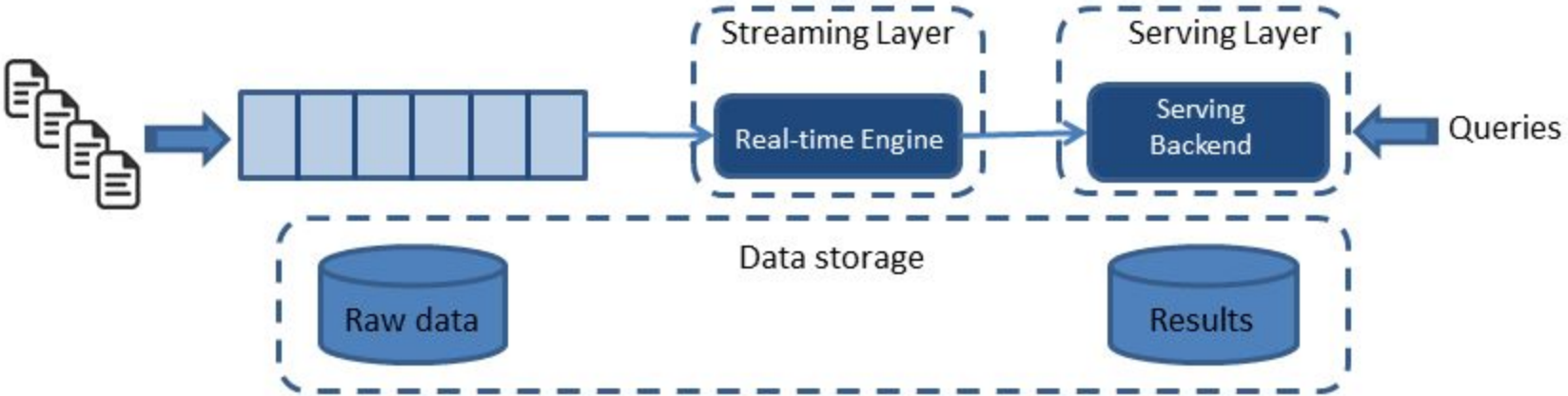
Плюсы

- ➕ Скорость стрима
- ➕ Без дублирования кода
- ➕ Простой репроцессинг

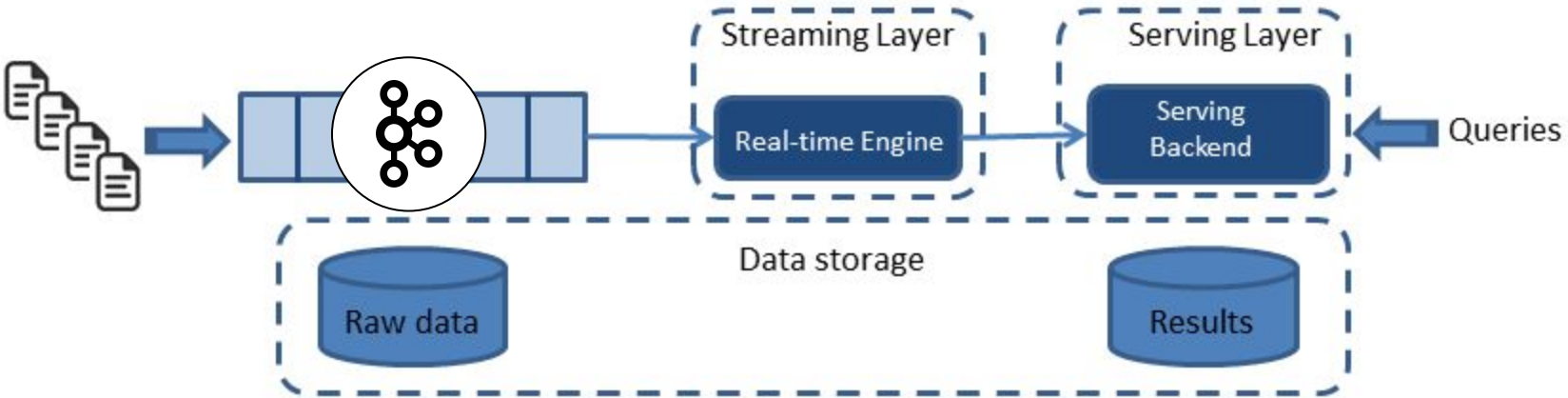
Минусы

- ➖ Ограничения реализации

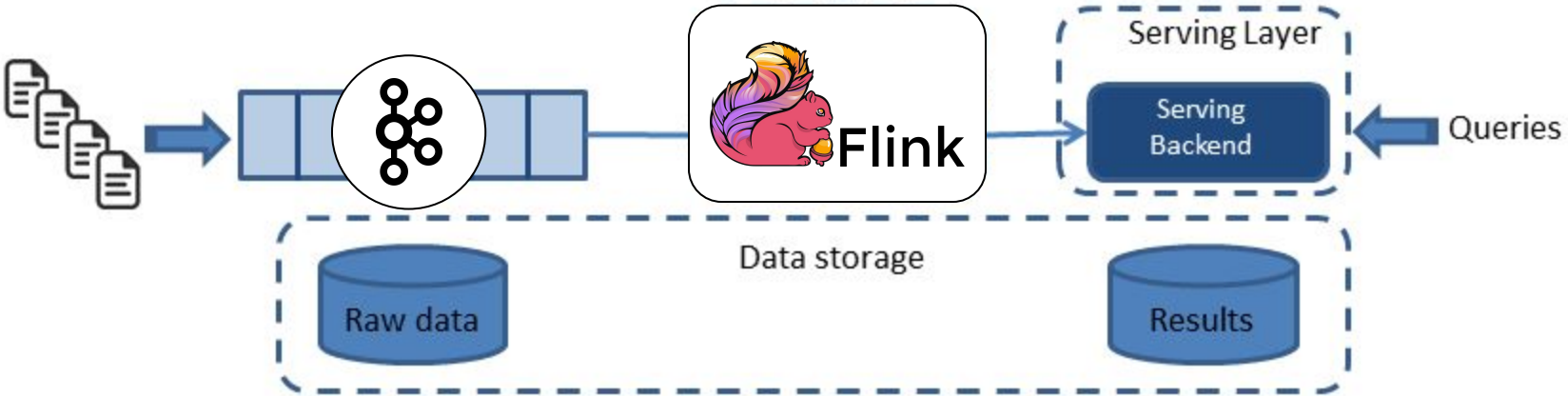
Kappa (example)



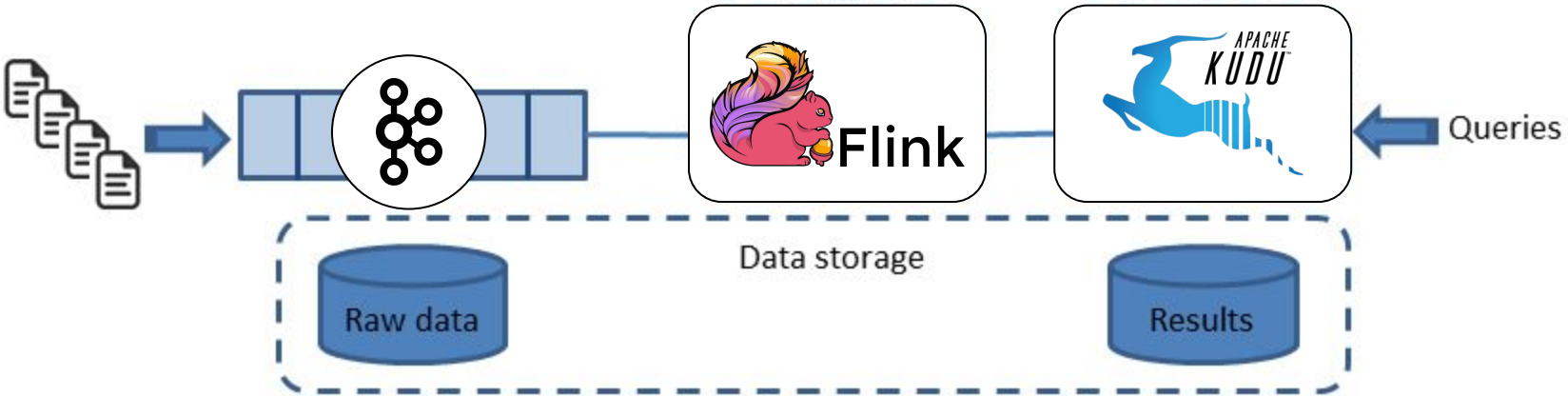
Kappa (example)



Kappa (example)



Kappa (example)



Наш опыт

Наш выбор

- Лямбду практически не используем (дорого)
- Каппу используем, но есть нюансы

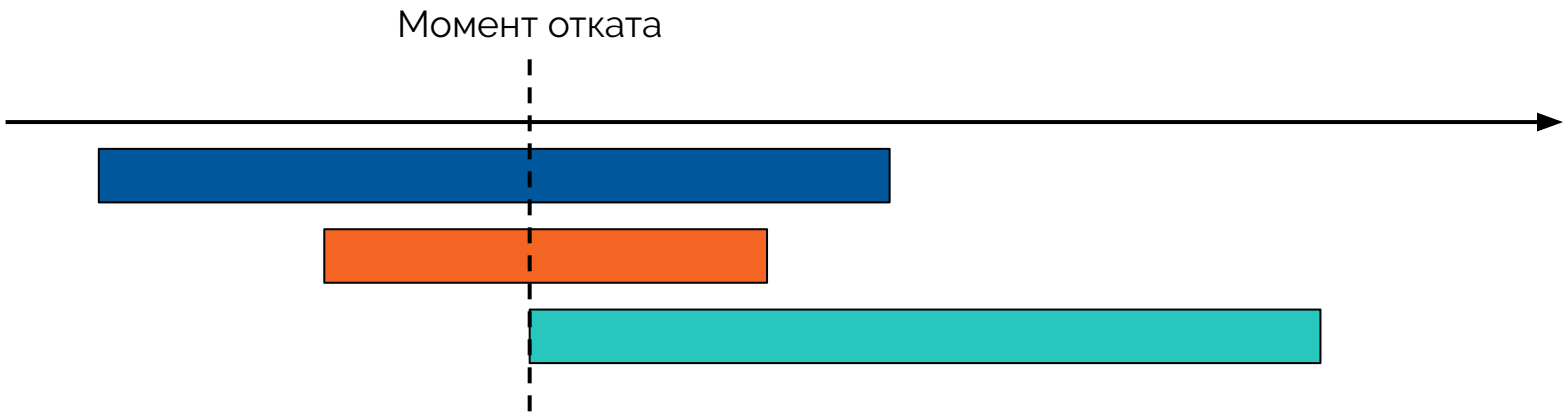
Проблема репроцессинга

- В оригинале предлагается откатывать чтение из потокового хранилища
- В нашем случае это не работает:
  - Мы не можем хранить бесконечно в Кафке
  - Сложно откатить на нужный момент

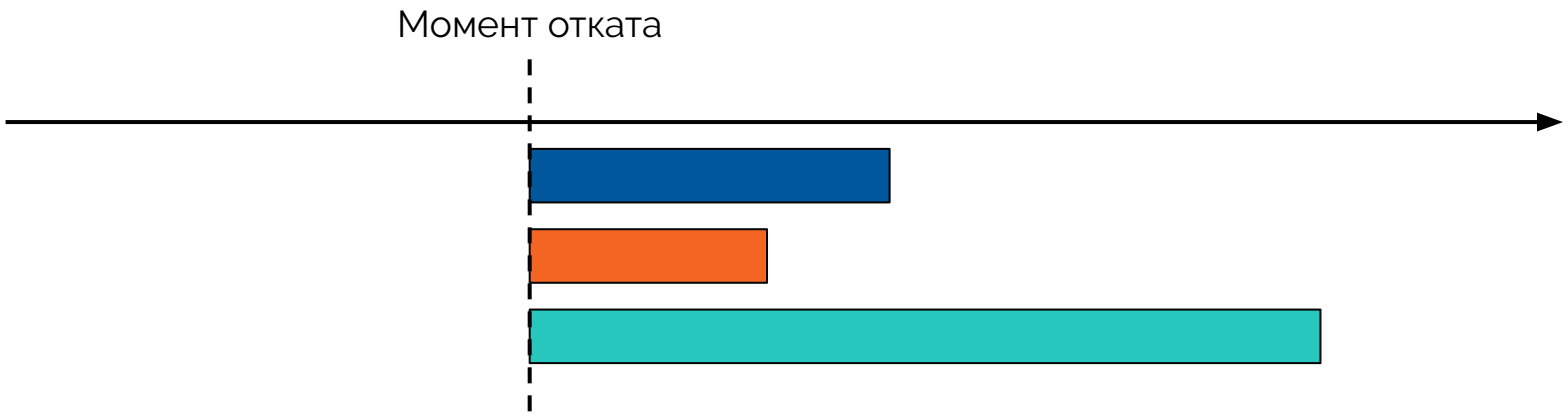
Kappa (fix)

- ~~Raw~~ Columnar data storage
- Flink Checkpointing

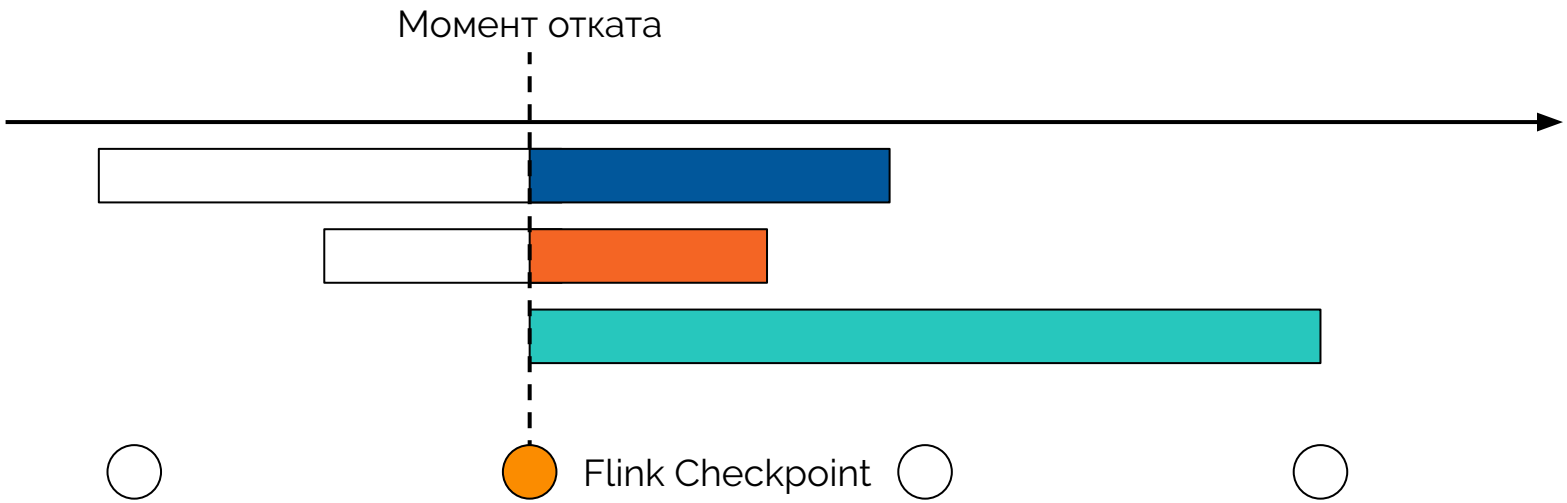
Пример: сессии



Пример: сессии

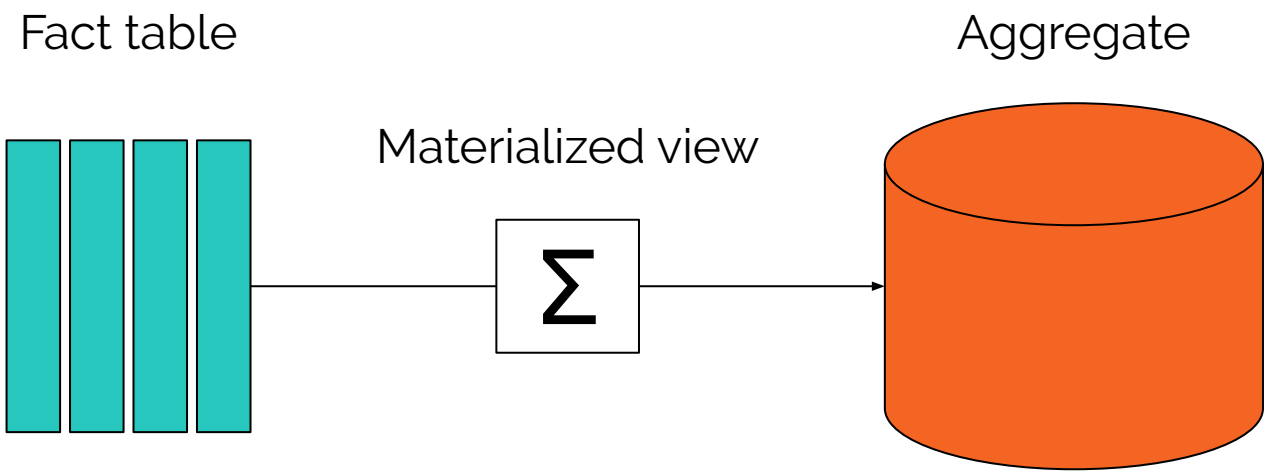


Пример: сессии



Еще одна идея

Использовать Materialized View в качестве Kappa



Главные идеи

Давайте выделим основные моменты, которые сегодня обсуждали

- ➔ **Stream vs Batch**
  Оба подхода - лишь “кирпичики” для более сложных архитектур.
- ➔ **Lambda**
  Надежно, но придется платить за оверхед.
- ➔ **Kappa**
  Красивая идея, но нужно аккуратно выбирать задачи и инструменты.

Спасибо за внимание!

Егор Матешук

egor@mateshuk.com