Streaming data 이해하기

강사 : 윤성국

streaming?

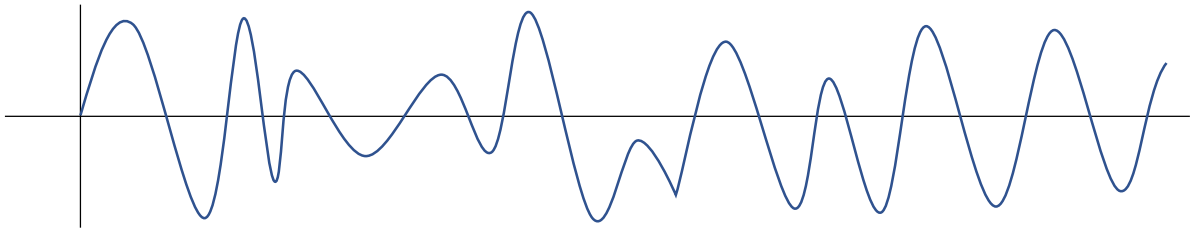
Receive

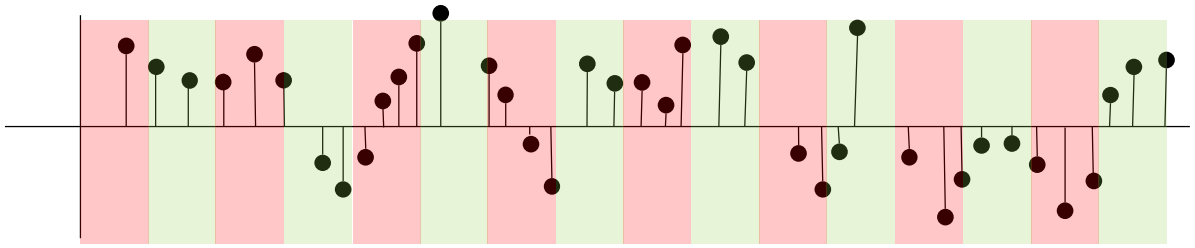
Processing in real time

React

Real time?

- 사실 진정한 의미의 실시간이란 건 없습니다.





Data processing

일단 저장

처리와 분석은
언젠가는(?) 하겠지

일단 리시브도 하고,

처리와 동작은
실시간으로

암튼 다 실시간으로

Batch processing

Stream processing

why?

Batch processing  Stream processing

과거로부터 배우는 방식

- 과거에 진행됐던 프로모션 / 제품

큰 그림을 결정하기 위해

- 새로운 스마트폰을 언제 어디서 런칭해야 잘 팔릴까?

why?

Batch processing

Stream processing

과거로부터 배우는 방식

- 과거에 진행됐던 프로모션 / 제품

큰 그림을 결정하기 위해

- 새로운 스마트폰을 언제 어디서 런칭해야 잘 팔릴까?

이벤트 모니터링과 추적

- 주식의 거래정보&가격 움직임
- 거래 실패 및 거래 사기 추적

빠르고 즉각적인 결정

- 언제 얼만큼의 주식을 사고 팔지
- 거래 게이트웨이 re-routing
- 거래 취소 처리

what?

Batch processing  Stream processing

Periodic Report

- day, week, month

Analyze trends

- WoW, MoM, QoQ

Compare Slices

- cities, categories

what?

Batch processing

Stream processing

Periodic Report

- day, week, month

Analyze trends

- WoW, MoM, QoQ

Compare Slices

- cities, categories

Process individual event

- tweets, stock quotes

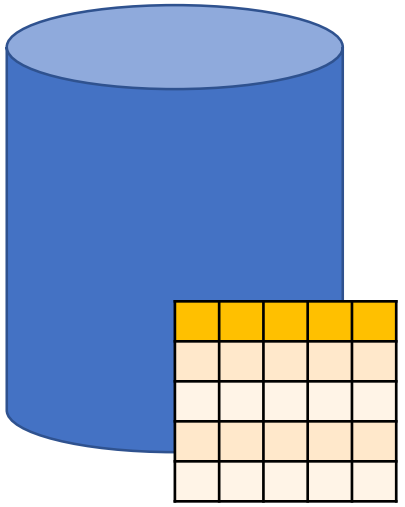
Batch processing

dataset

batch processing job

report, trends, insights

Batch processing



dataset

- 끝이 있는 바뀌지 않는 dataset
  (Week, Month, Year)

-> Bounded dataset

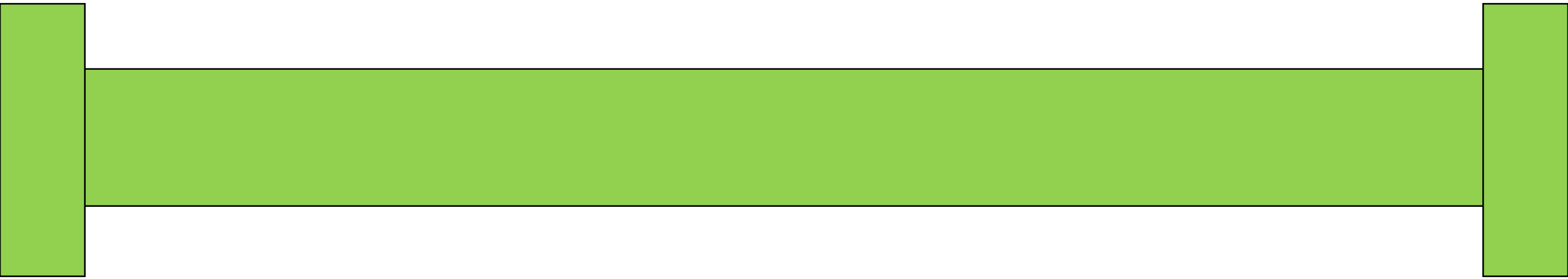
Batch processing

- 특정한 시점에 동작하는 데이터 처리 job

-> Batch jobs

batch processing job

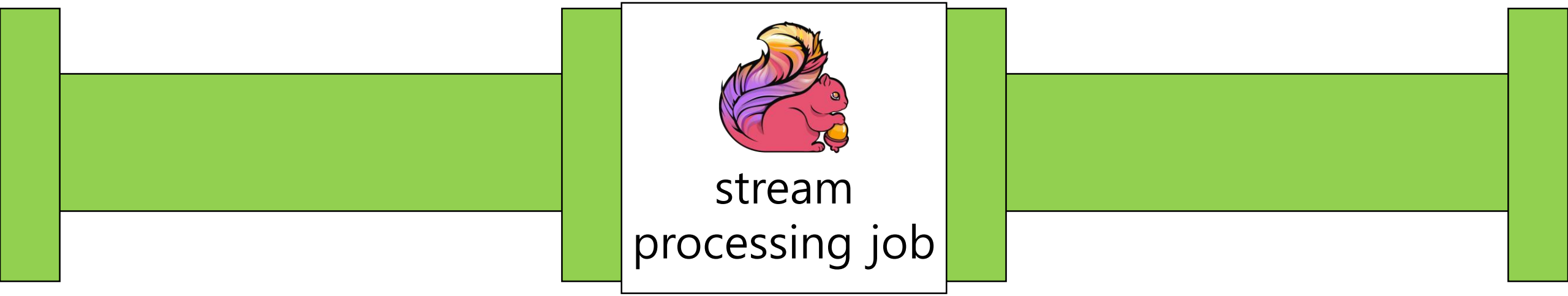
Stream processing



Event

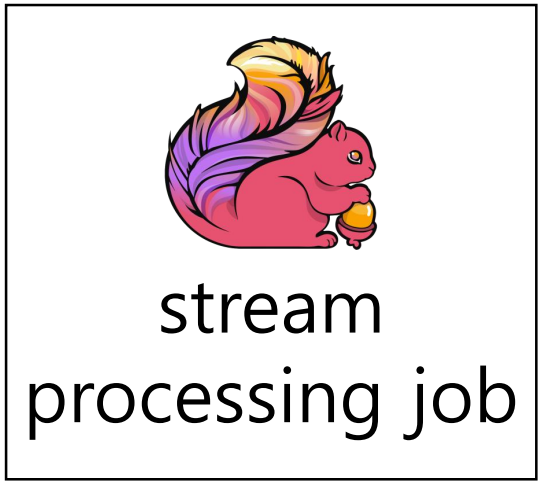
application event
news feed
stock quotes
Tweet

Stream processing



이벤트가 도착하자마자 처리

Stream processing



- 지속적으로 계속 추가되는 끝이 없는 dataset

  -> Unbounded dataset

- 데이터를 받는 한 지속적으로 실행

  -> Continuous processing

System

Batch processing

Stream processing

bounded datasets 처리

특정 기간마다 처리 및 갱신

데이터를 받는 순서는 중요하지 않음

언제 어디든 글로벌한 단일 상태

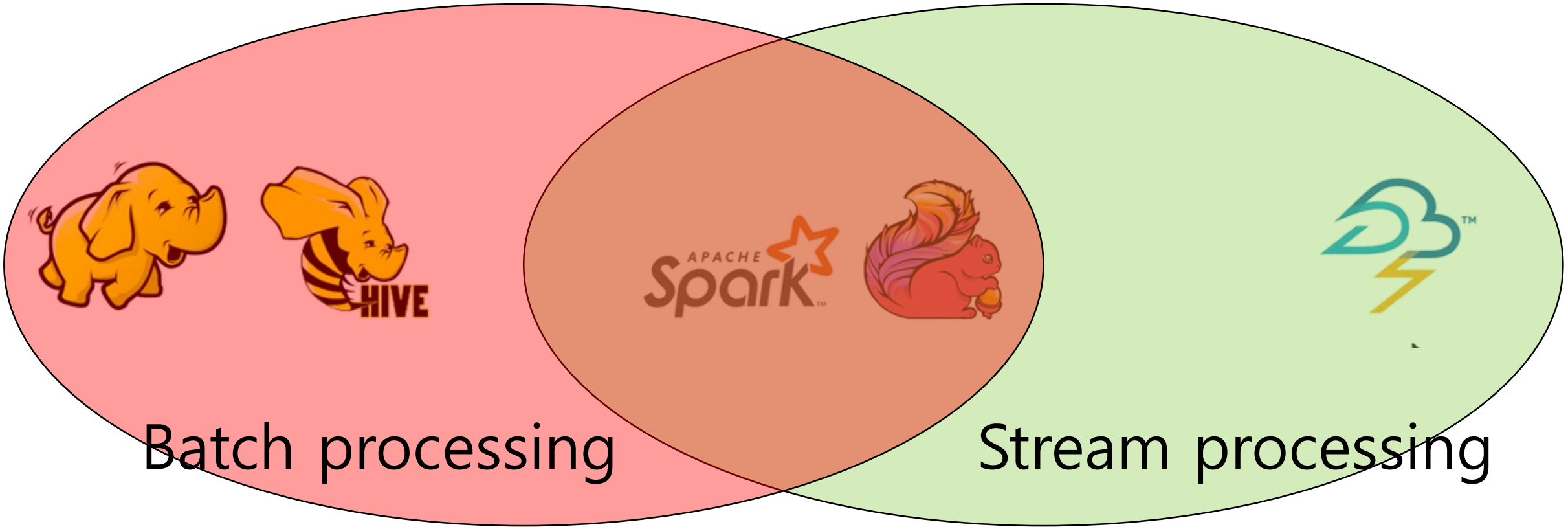
unbounded datasets 처리

지속적인 실시간 업데이트

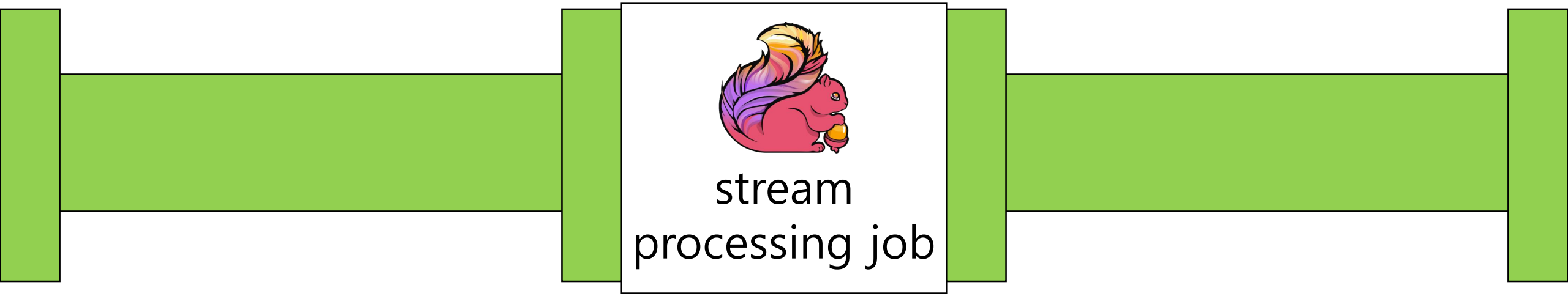
순서가 상당히 중요함, 도착순서 오류 추적

단일상태 아님,받은 이벤트의 윈도우 기반

processing tool



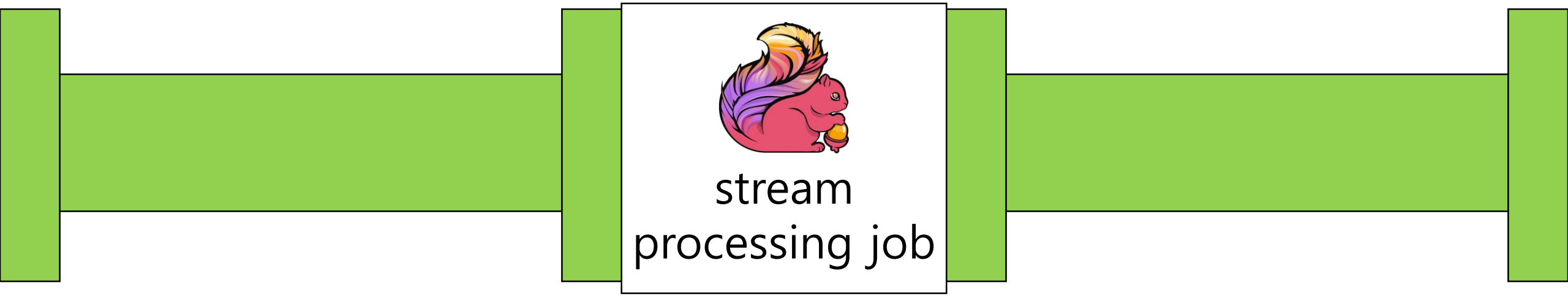
Stream processing



이벤트가 도착하자마자 처리

Filter, transform, combine
정보의 손실없이 빠르게 처리해야 한다

Stream processing



Message Transport

streaming data의 버퍼
성능과 지속성 유지
다중 데이터 소스와 다중 처리기의 결합도를 낮춤 -> async

Source of Truth

Tranditional Architecture

persistant storage
-> databases / files

Stream processing Architecture

Stream
-> message queue
message broker
pipeline

streaming architecture에서 데이터 일관성 유지

- Checkpointing / periodic backups(snap shot) to manage failures
- 잘못된 이벤트와 예외 처리
- 필요시 stream replay – roll back / fail back

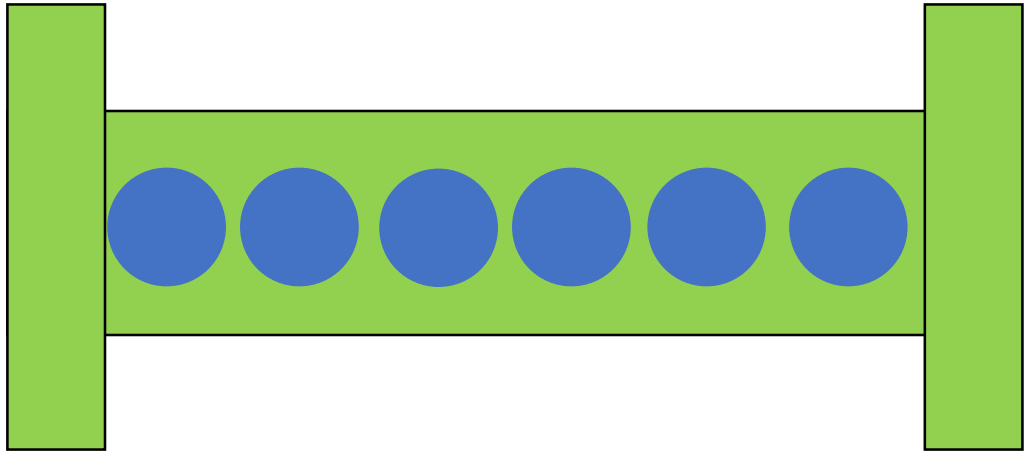
스트리밍 처리에 요구되는 사항

stream
processing job

- 높은 throughput, 낮은 latency
- 낮은 overhead의 Fault tolerance 기능
- 잘못된 event에 대한 관리 및 처리
- 쉬운 사용과 유지보수 (운영)
- Replay stream

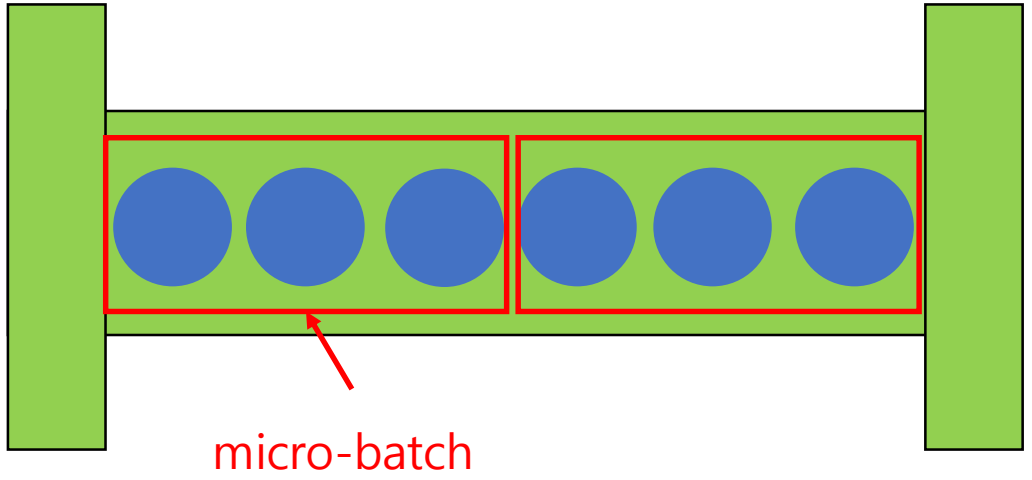
스트리밍 처리의 두가지 방식

Stream-first



stream을 stream 그대로 취급하여 처리

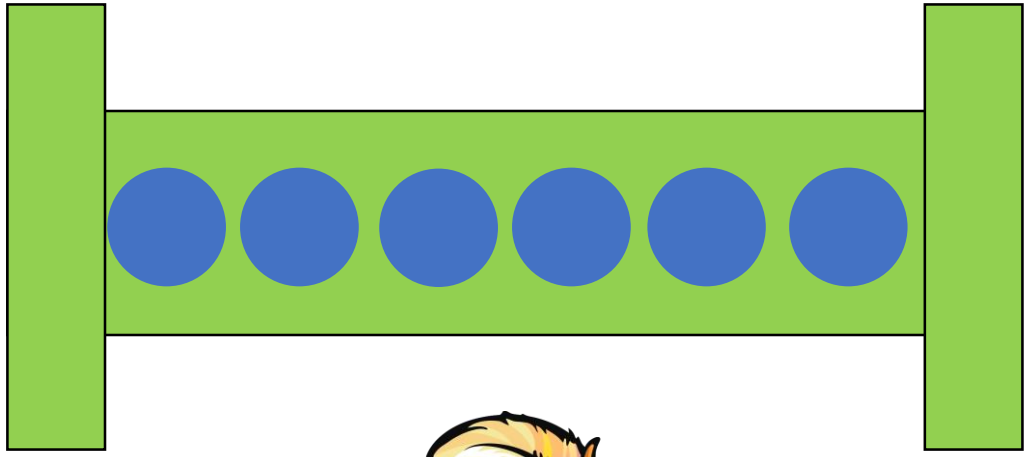
Micro-batching



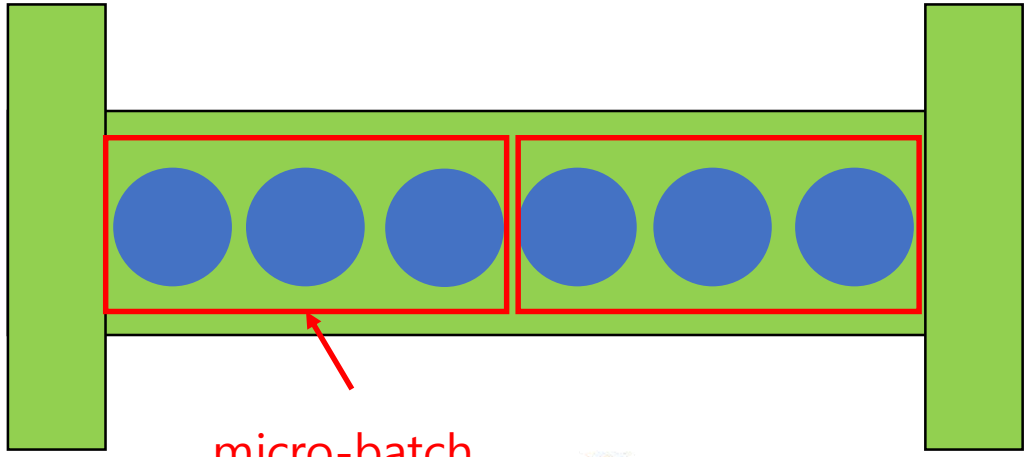
stream을 collection of batch로 취급하여 처리

스트리밍 처리의 두가지 방식

Stream-first



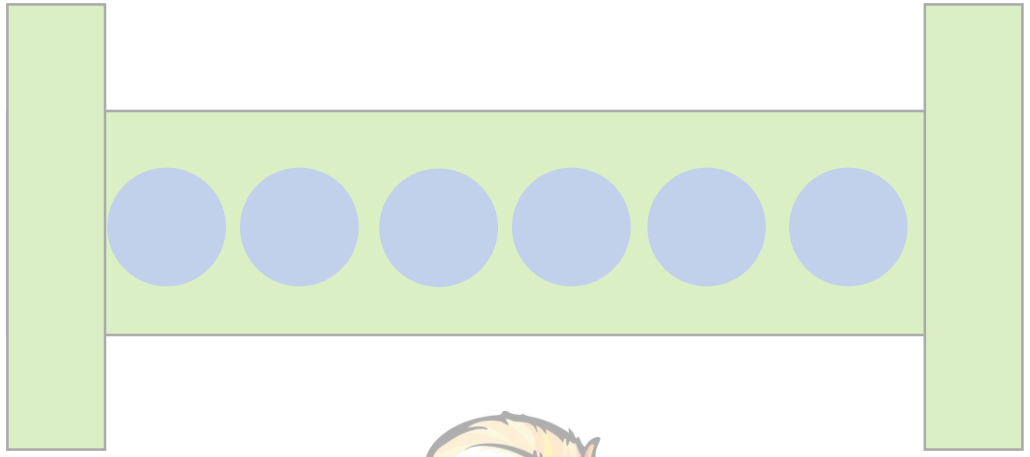
Micro-batching



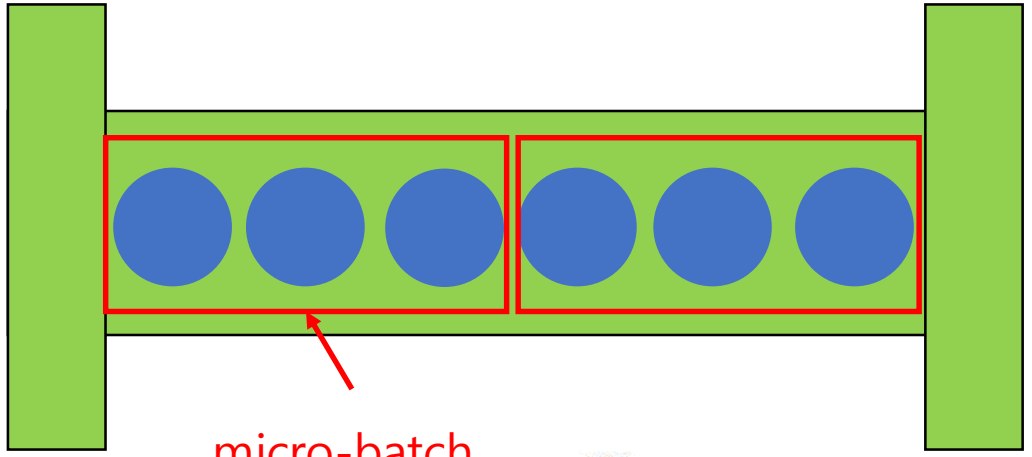
Spark Streaming

스트리밍 처리의 두가지 방식

Stream-first

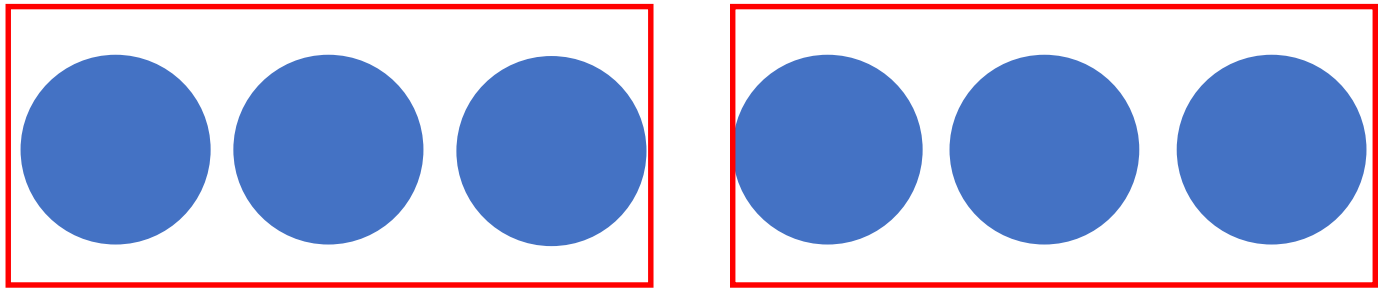


Micro-batching



Spark Streaming

Micro-batching

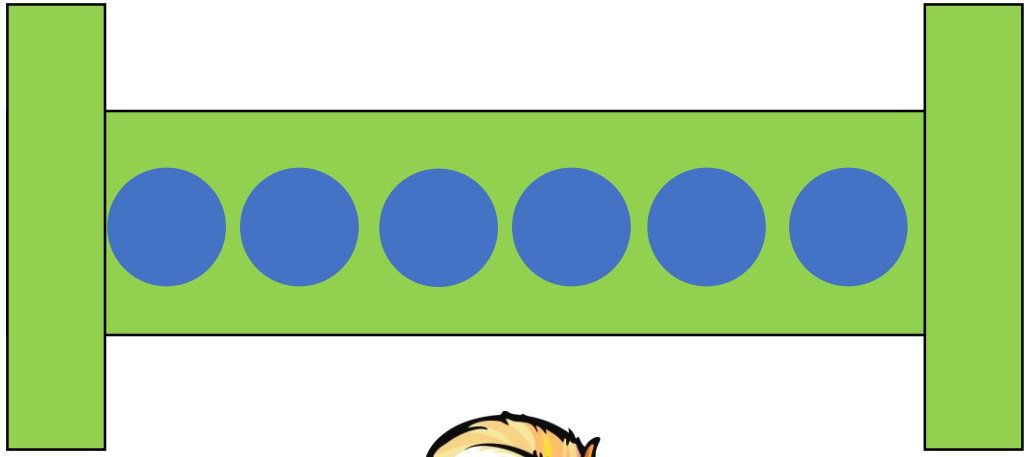


small batch

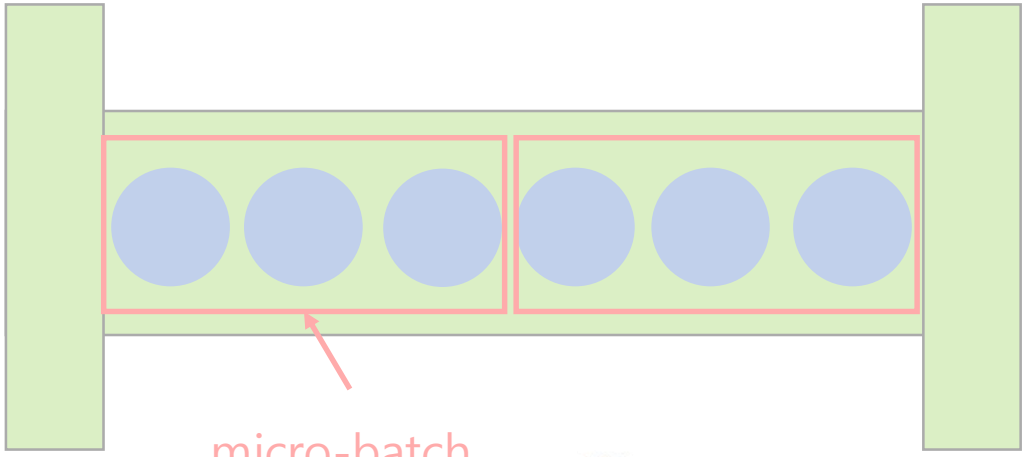
실시간 비스무리한(?) 스트리밍 처리
latency와 throughput은 batch size에 의해 결정

스트리밍 처리의 두가지 방식

Stream-first

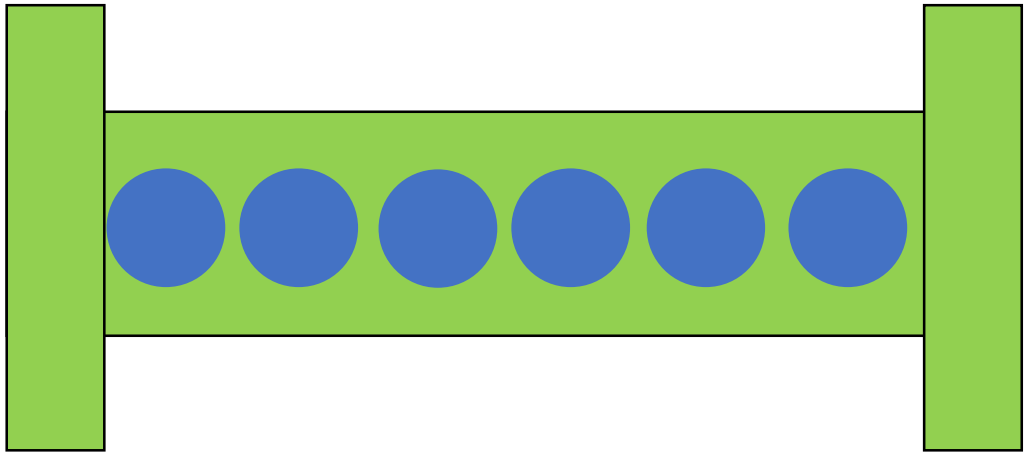


Micro-batching



Spark Streaming

Stream-first



한번에 한 이벤트에 대한 처리

batch processing은 stream processing 중에
하나의 특수 게이트로 취급