Building Advanced Streaming Pipelines Using Structured Streaming

Janani Ravi
CO-FOUNDER, LOONYCORN
www.loonycorn.com

Overview

Selections, projections and aggregations on streaming data

Adhoc SQL queries on streams

Windowing allows operating on a subset of streaming data

Work with Twitter streaming data

Lateness is the difference between event time and processing time

Watermarking helps deal with lateness

Demo

Using structured streaming in append mode

Demo

Using structured streaming in complete mode

Demo

Aggregations on streaming data

Demo

Running SQL queries on streaming data

Demo

Including timestamps to mimic event time

Demo

Grouping data on timestamps

Stateful Operations on Windows

Transformations

Stateless

Transformations which are applied on a single stream entity

Stateful

Transformations which accumulate across multiple stream entities

Window Transformations

Accumulate information across a window in a stream

80mph

45mph

Stateless Transformations

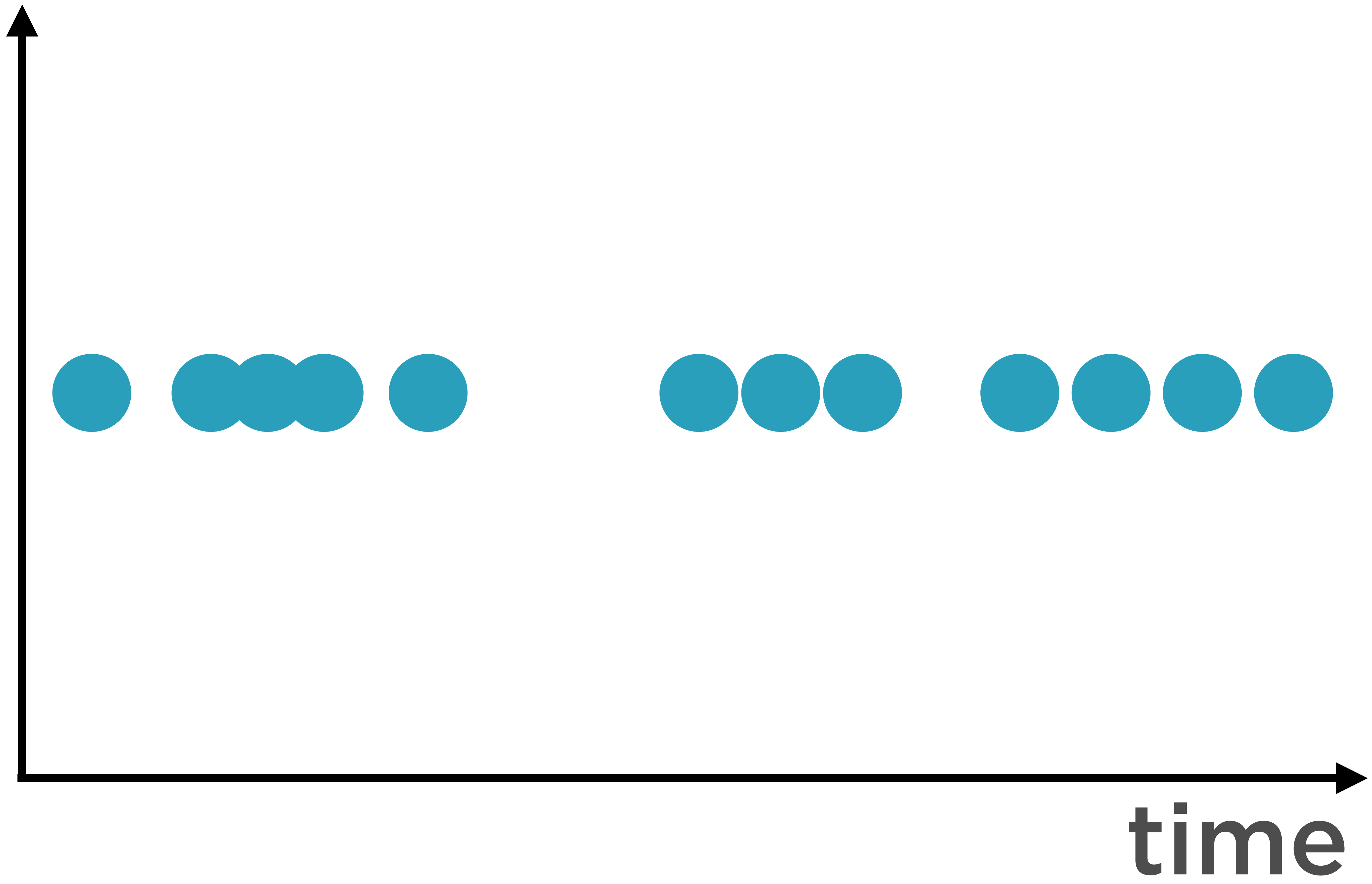
94 86 32 55 83 22 19 65 78 30

Each entity is operated on standalone

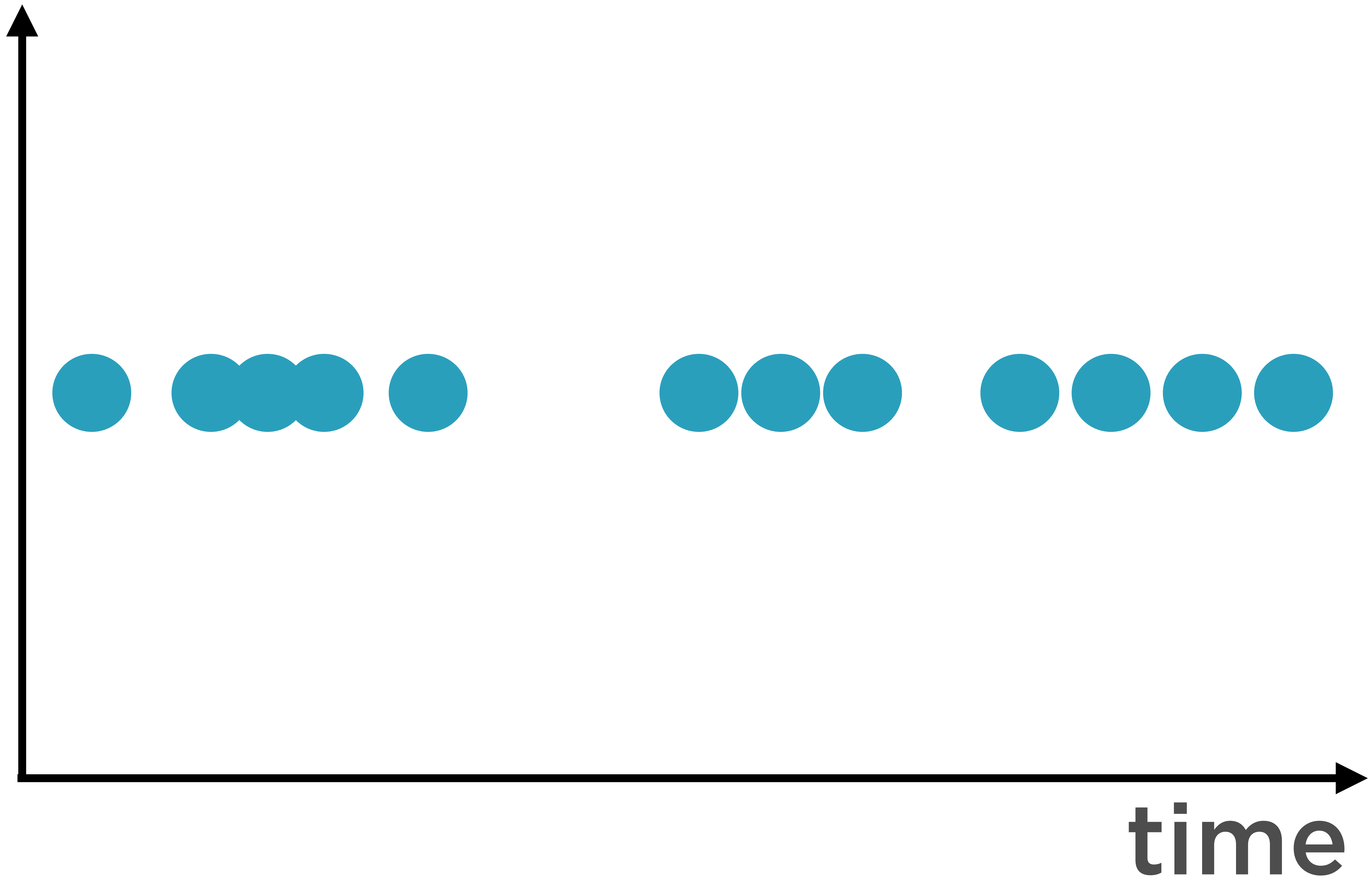
Speed exceeded? Alert triggered

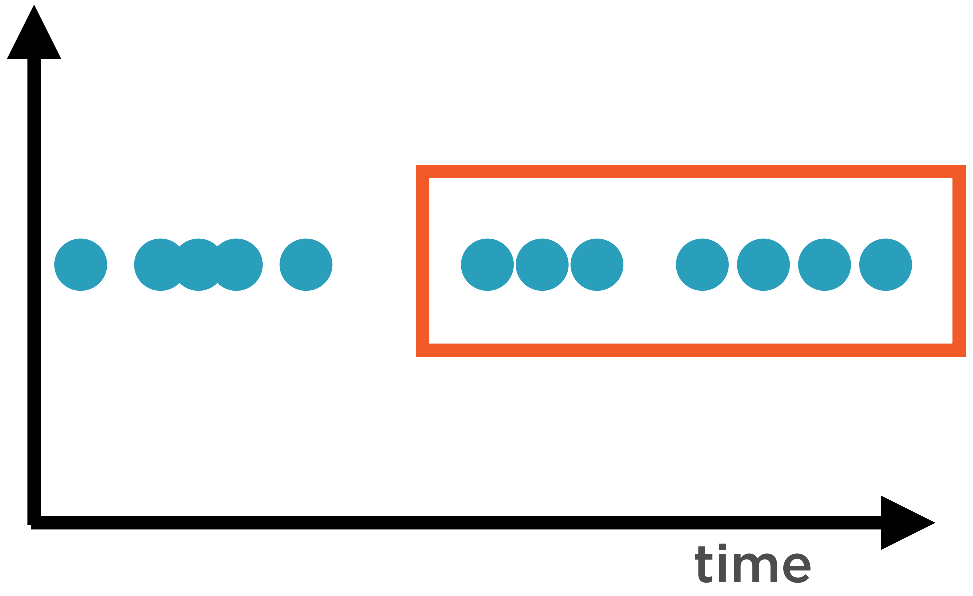
[illegible]

Streaming Data



Streaming Data

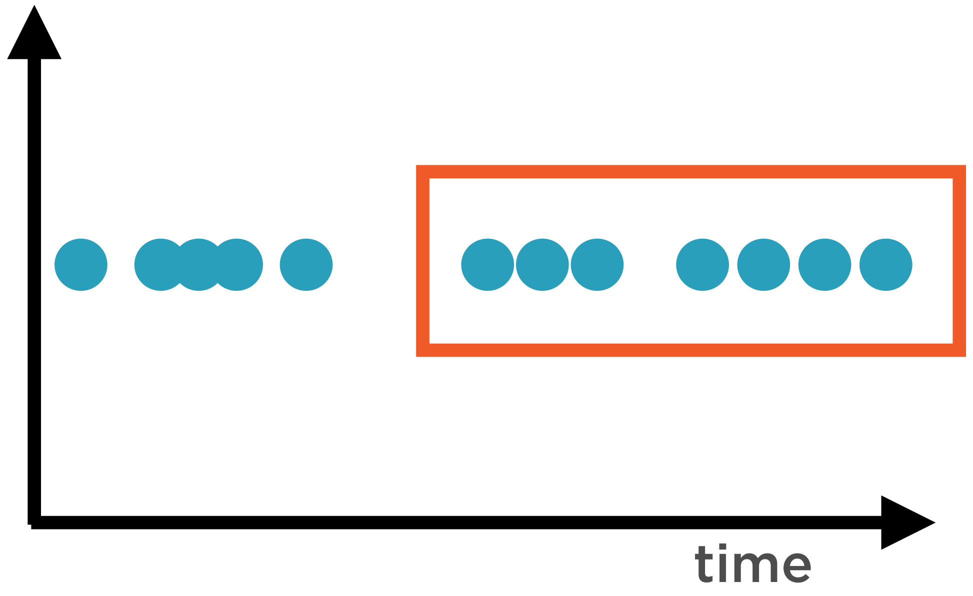




Window Transformations

A window is a subset of a stream based on

- Time interval
- Count of entities
- Interval between entities



Window Transformations

Transformations can be applied on all entities within a window

- sum, min, max, average

Types of Windows

Types of Windows

- Tumbling Window
- Sliding Window
- Count Window
- Session Window
- Global Window

Types of Windows

Tumbling Window

Sliding Window

Count Window

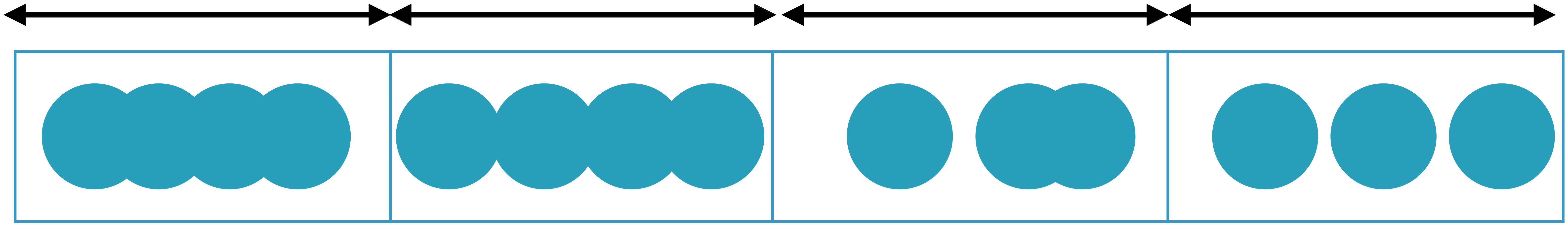
Session Window

Global Window

Types of Windows

A stream of data

Tumbling Window

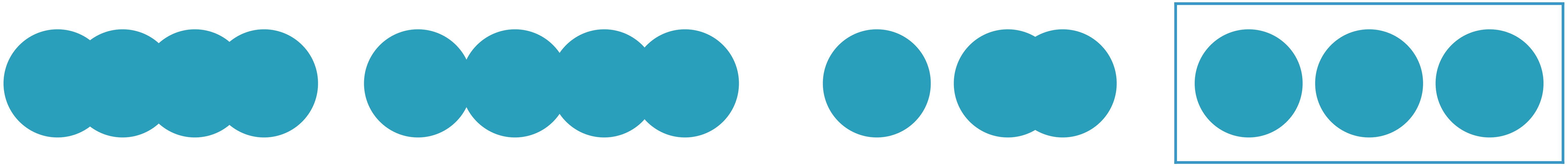


Fixed window size

Non-overlapping time

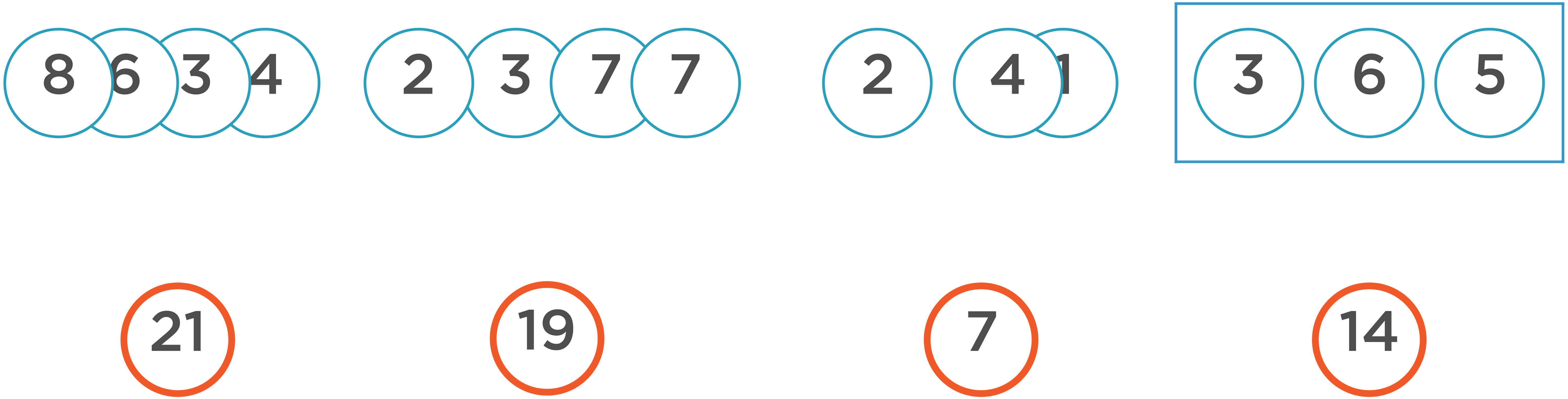
Number of entities differ within a window

Tumbling Window



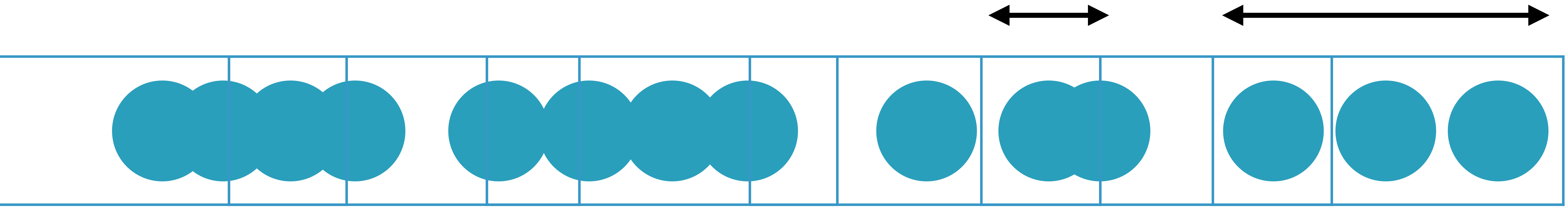
The window tumbles over the data, in a non-overlapping manner

Tumbling Window



Apply the sum() operation on each window

Sliding Window

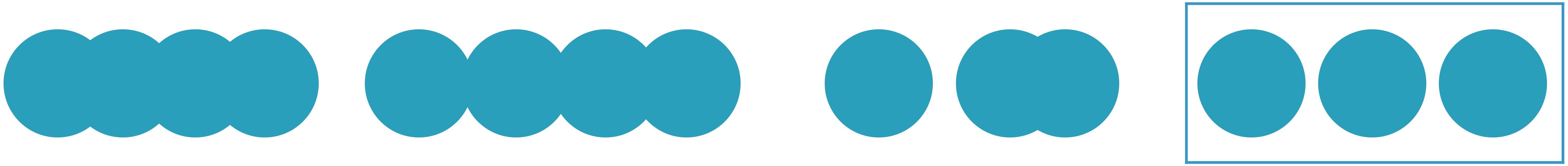


Fixed window size

Overlapping time - sliding interval

Number of entities differ within a window

Sliding Window

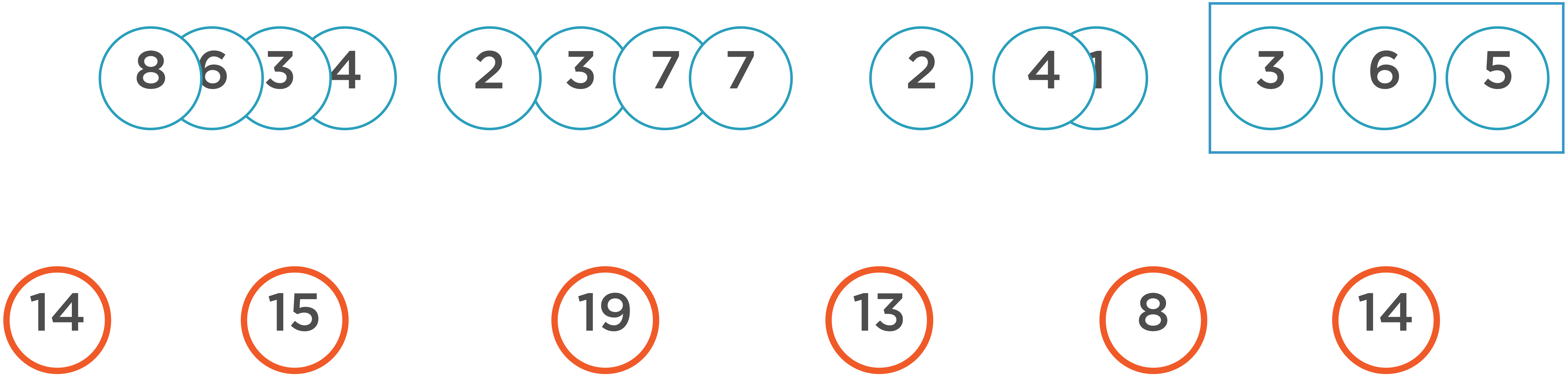


Fixed window size

Overlapping time - sliding interval

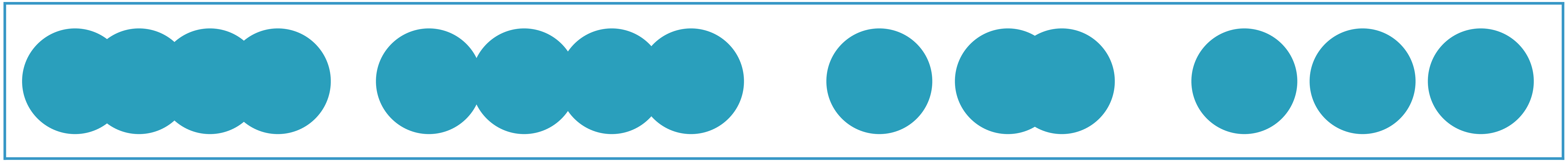
Number of entities differ within a window

Sliding Window



Apply the sum() operation on each window

Global Window

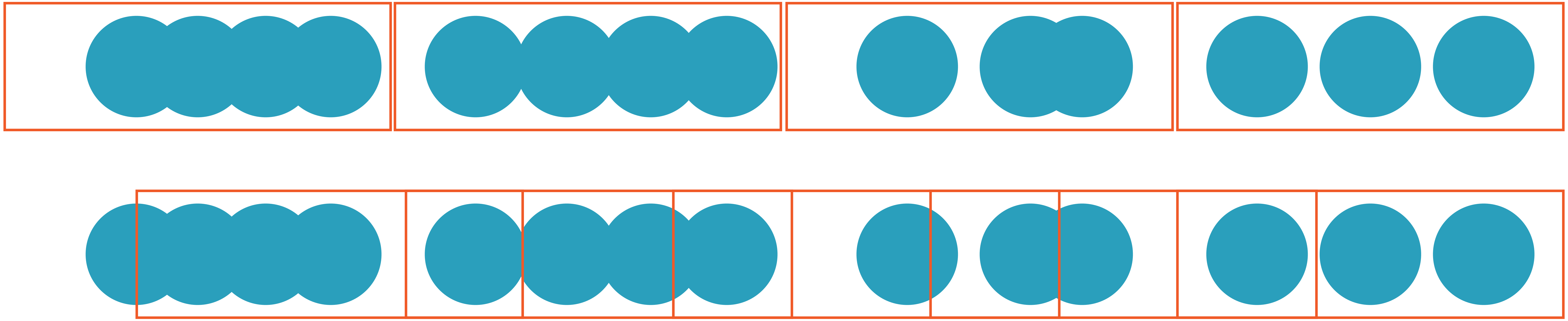


All data in the stream in one window

The Notion of Time

Time-based Windows

Tumbling and sliding windows consider entities in a fixed interval of time



Time-based Windows

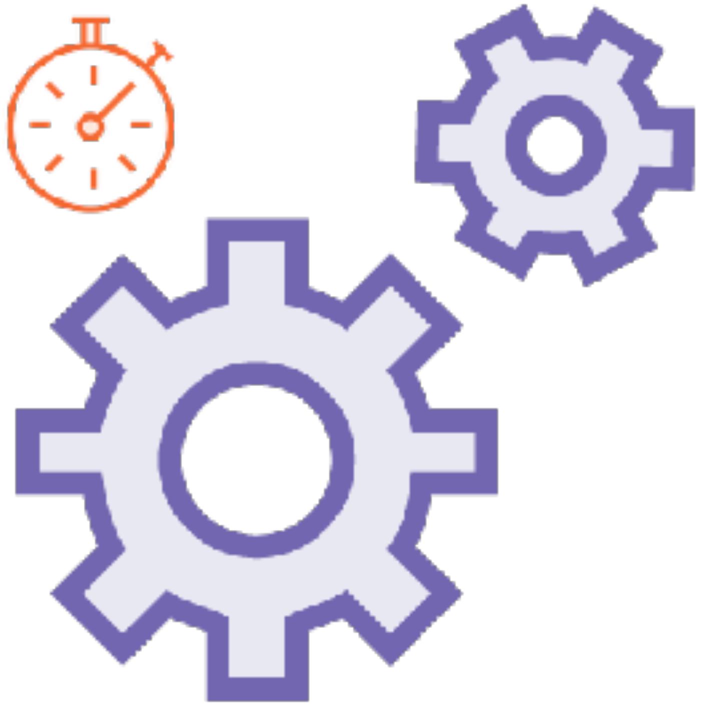
Tumbling and sliding windows consider entities in a fixed interval of time

There are different notions of time that can apply to entities in a stream

Time

Event Time

Ingestion Time



Processing Time

Event Time

The time at which the event occurred at its original source

- Mobile phone, sensor, website

Usually embedded within records

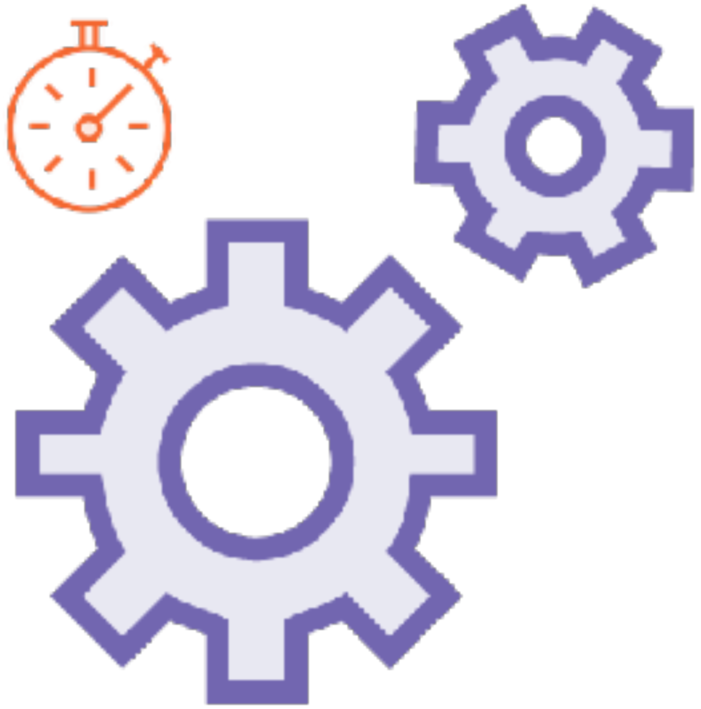
Gives correct results in case of out of order or late events

Ingestion Time

The time at which the event enters Spark via a source

Timestamp given by system chronologically after the event time

Cannot handle out of order events



Processing Time

The system time of the machine processing entities

Chronologically after event time and ingestion time

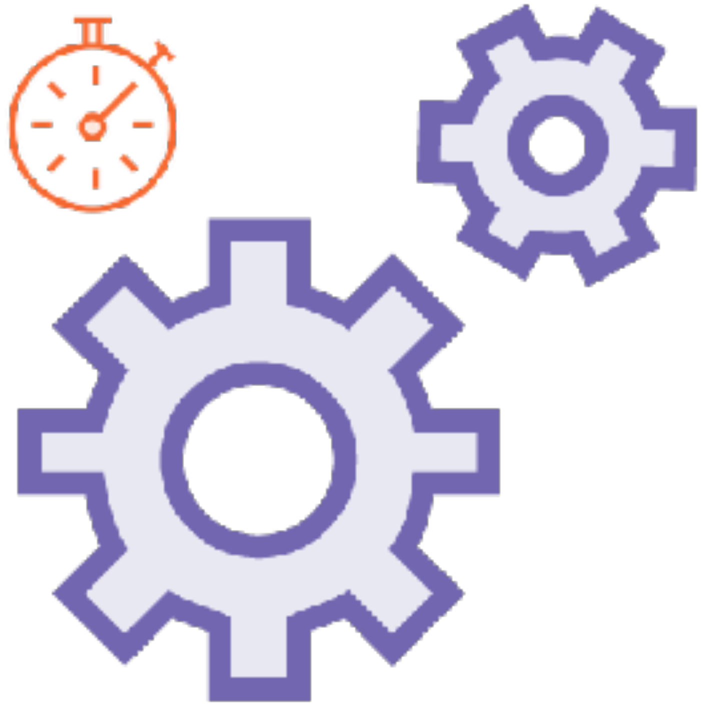
Non-deterministic, depends on when data arrives, how long operations take

Simple, no coordination between streams and processors

Time

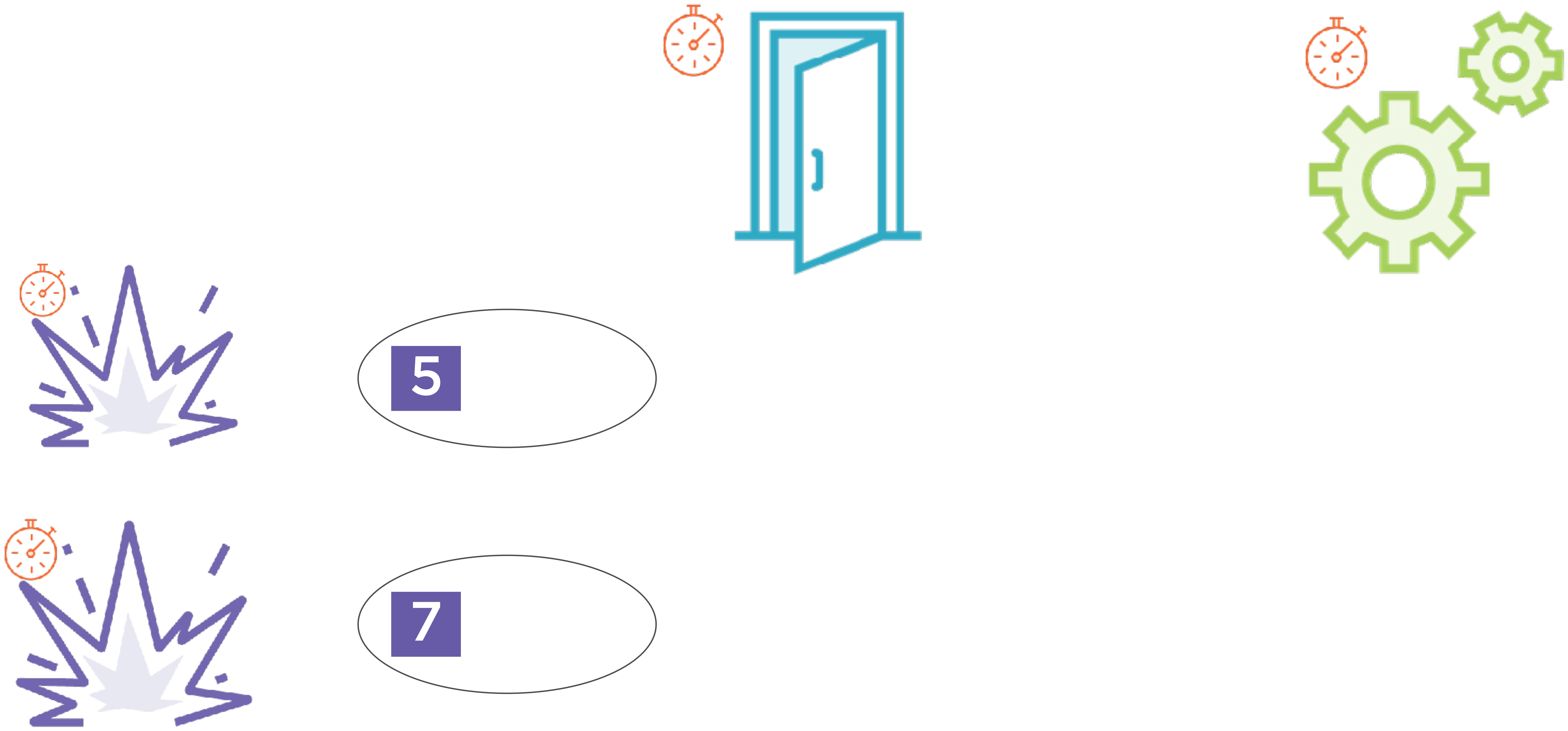
Event Time

Ingestion Time

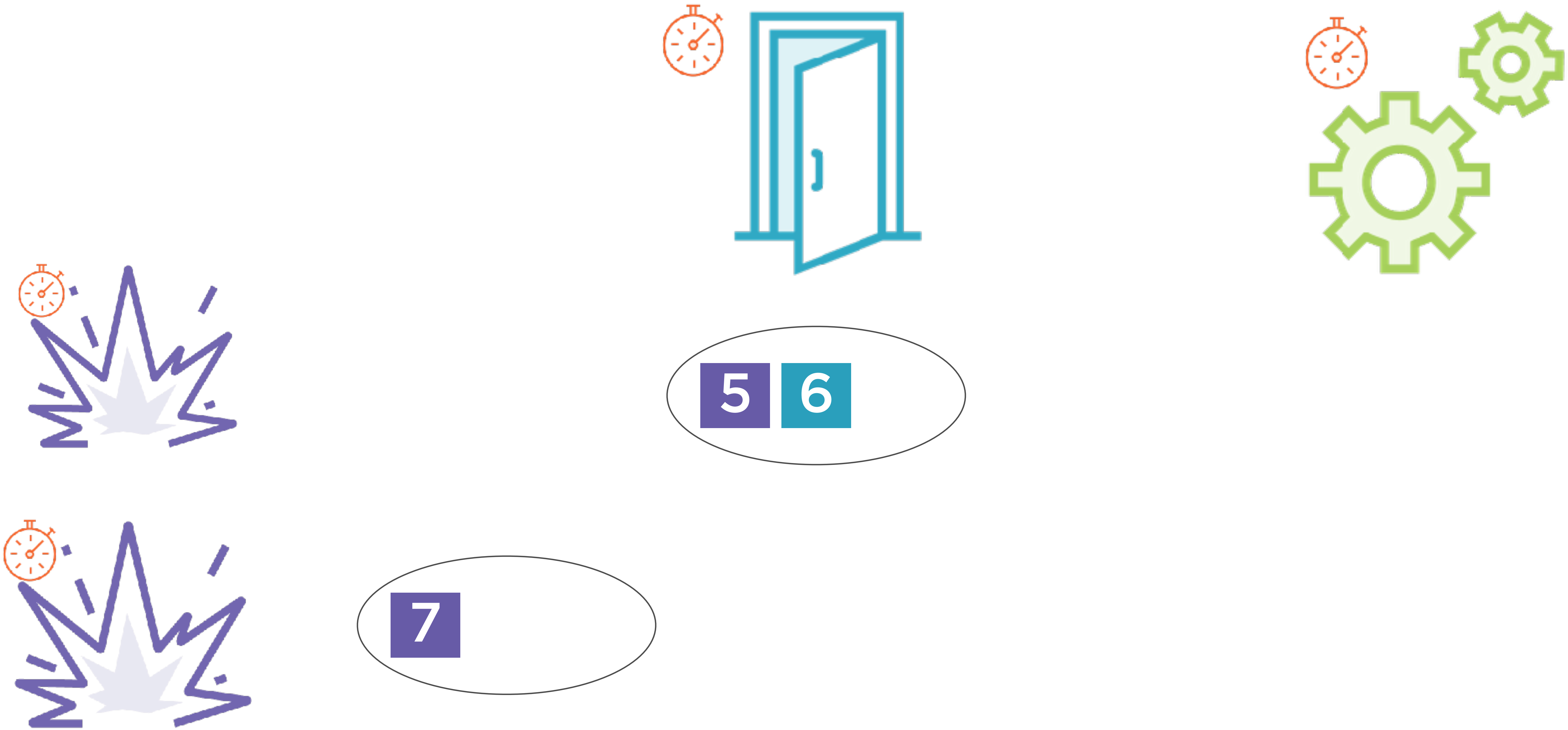


Processing Time

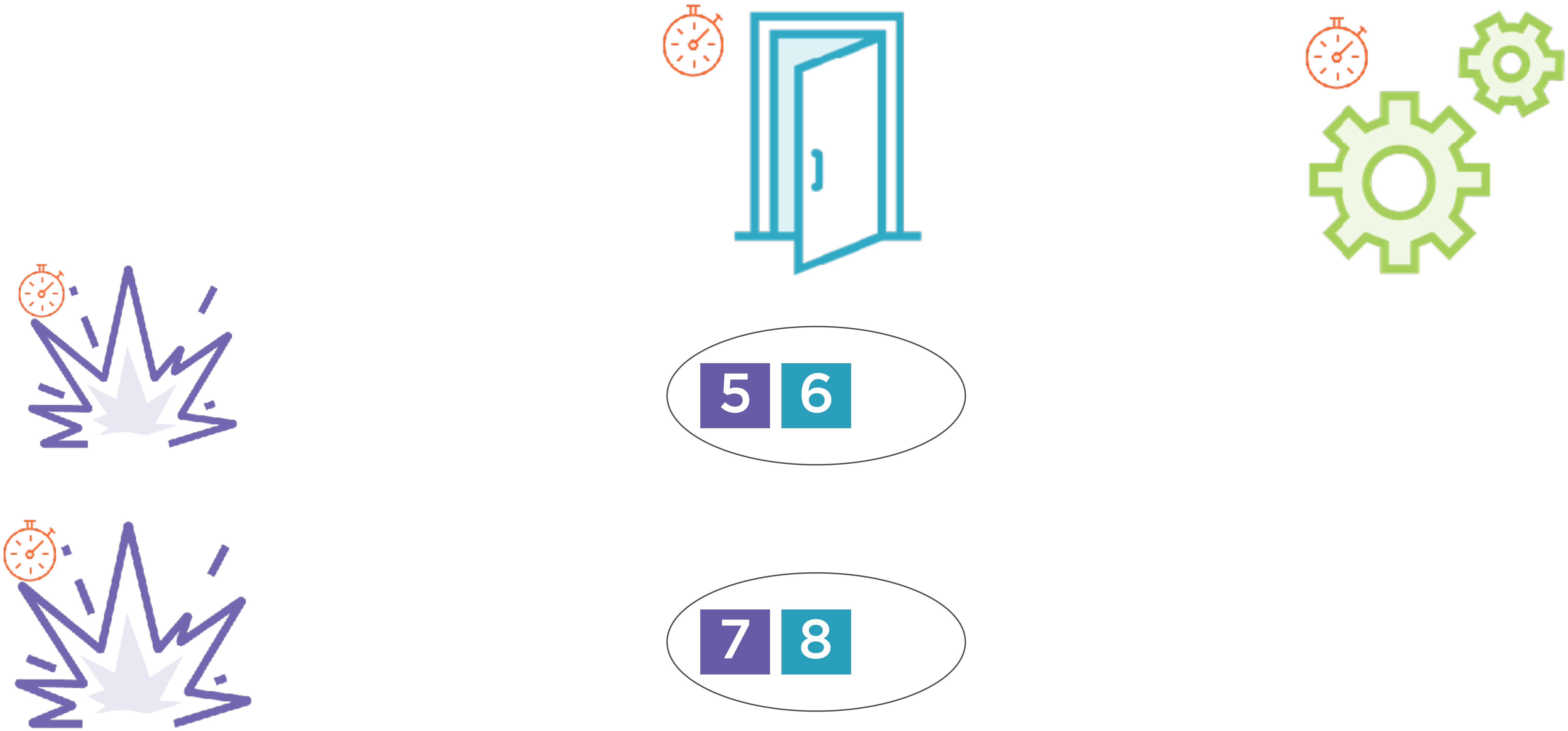
Time



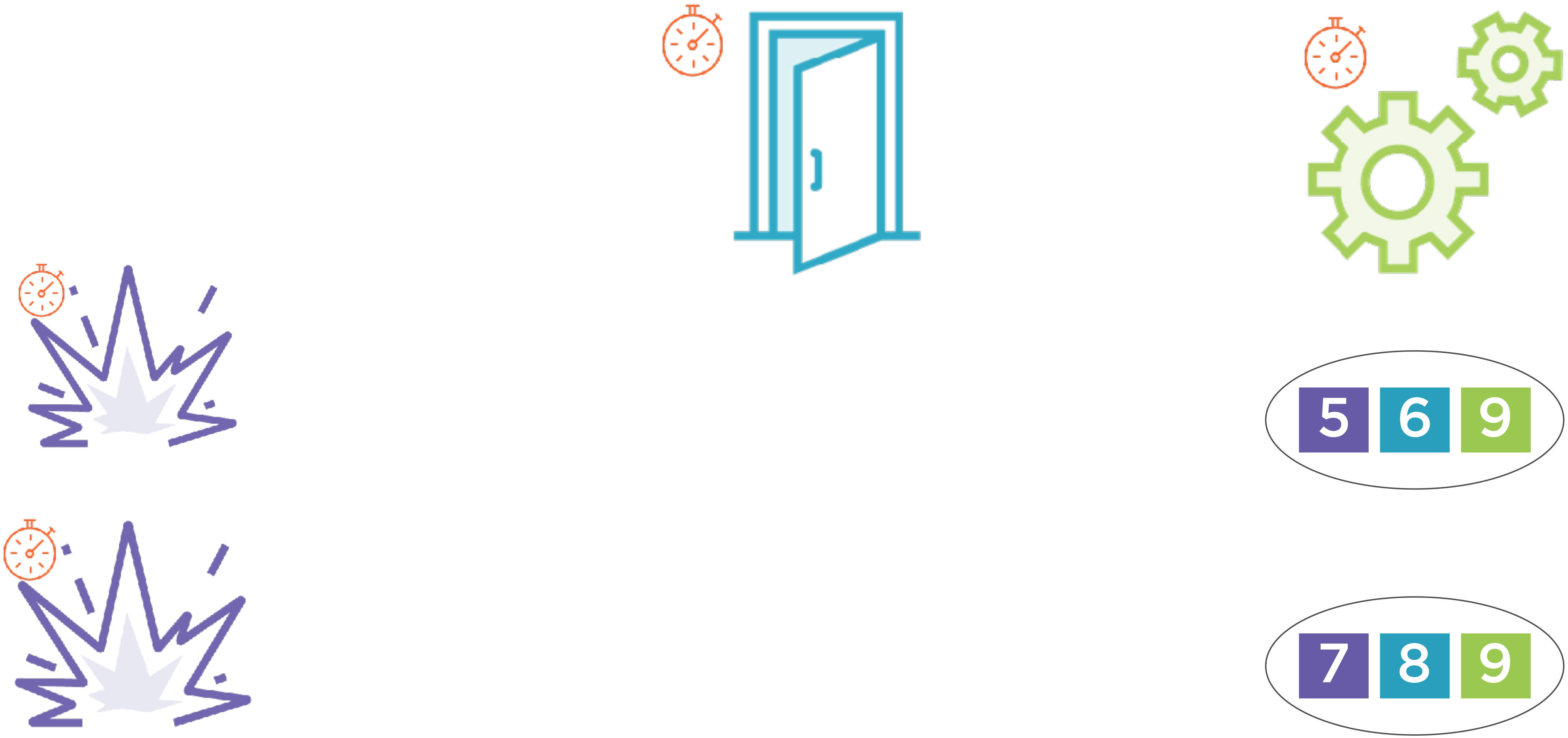
Time



Time



Time



Time

Window operations in structured streaming use event time

Demo

Using sliding windows on streaming data

Watermarks and Lateness

How Late is Late?

Class At 9 am

Class starts when clock strikes 9

Is 9:01 Late?

Realistically, at least some folks are going to be a minute late

Is 10:10 late?

A student is an hour late - allow in or send back?

Allowed Lateness

The professor “knows” what lateness is reasonable

Students entering within this reasonable lateness are late but OK

Students entering after this reasonable lateness are too late

Excessive Lateness

A student is too late

- Option 1: Send back home
- Option 2: Allow in, continue class
- Option 3: Allow in, restart class(!)

How Late is Late?

Trigger

Class starts when clock strikes 9

Allowed Lateness

Realistically, at least some folks are going to be a minute late

Unacceptable Lateness

A student is an hour late - allow in or send back?

Allowed Lateness

The system “knows” what lateness is reasonable

Data entering within this reasonable lateness is late but OK

Data entering after this reasonable lateness is too late

Watermarks and Late Data

The system "knows" what lateness is reasonable

Data entering within this reasonable lateness is late but OK

Data entering after this reasonable lateness is too late

Watermarks and Late Data

Watermark

Threshold of allowed lateness (event time)

Late Data

Data within watermark is aggregated (used in processing Result Table)

Dropped Data

Data outside watermark is dropped (not used in processing Result Table)

```
windowedCounts = words.groupBy(
    window(words.timestamp, "10 minutes", "5 minutes"),
    words.word
).count()
```

Simple Group-by Without Watermark

Count words in each sliding window of width 10 minutes, sliding by 5 minutes

```
windowedCounts = words \
    .withWatermark("timestamp", "12 minutes") \
    .groupBy(
        window(words.timestamp, "10 minutes", "5 minutes"),
        words.word) \
    .count()
```

Simple Group-by With Watermark

We define the watermark i.e. lateness threshold to be 12 minutes

```
windowedCounts = words \
    .withWatermark("timestamp", "12 minutes") \
    .groupBy(
        window(words.timestamp, "10 minutes", "5 minutes"),
        words.word) \
    .count()
```

Simple Group-by With Watermark

Now window triggering will be delayed by 12 minutes

Watermark

System generated or user specified

If, say network speed drops, watermark can become more lenient

Lateness = Processing Time - Event time

Output Modes

Append Mode: Window not triggered at all until watermark elapses

- No partial updates

Update mode: Window will trigger even before watermark elapses

- Engine will keep partial counts

Complete mode: Can not be used with watermarks

Restrictions

No complete-mode queries

Aggregation must be event-time, or event-time window

`.withWatermark` must be called on same timestamp column as aggregate

`.withWatermark` must be called before the aggregation

One-way Guarantee

All data before watermark will definitely not be dropped

All data after watermark may or may not be dropped

Demo

Setting up a Twitter account

Getting keys and access tokens to access the Twitter streaming API

Demo

Using Tweepy to work with Twitter streaming data

Demo

Count hashtags on Twitter to find overall trends

Demo

Count hashtags to find trends using windows

Demo

Count hashtags to find trends using windows

Demo

Join operations using batch and streaming data

Demo

Join operations using aggregations

Demo

Find aggregate ratings using joins

Demo

Join operations on windowed streams

Summary

Selections, projections and aggregations on streaming data

Adhoc SQL queries on streams

Windowing allows operating on a subset of streaming data

Lateness is the difference between event time and processing time

Watermarking helps deal with lateness