Week 12 exercises: Stream Processing

EPFL SaCS and DIAS

EPFL

May 19, 2025

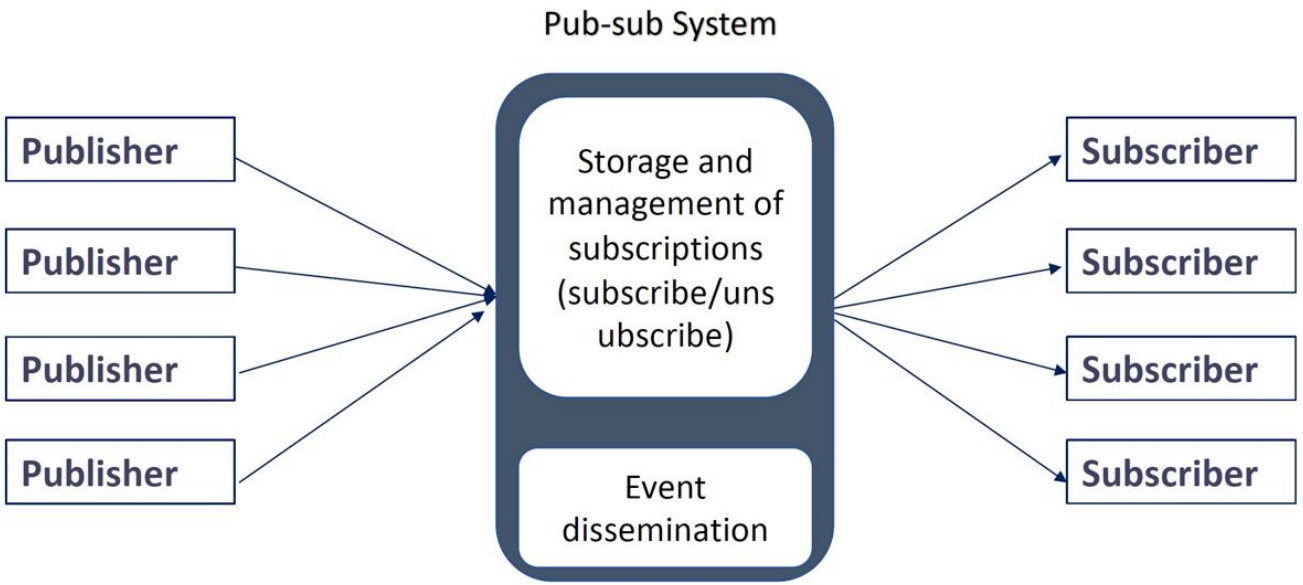
Exercise 1: Publish/Subscribe (Pub/Sub) Systems

- What is a Publish/Subscribe (pub/sub) system?

Exercise 1: Publish/Subscribe (Pub/Sub) Systems

- What is a Publish/Subscribe (pub/sub) system?

  Pub/Sub systems carry out a **communication paradigm** that decouples producers and consumers of data items in terms of **time**, **space** and **synchronization**.



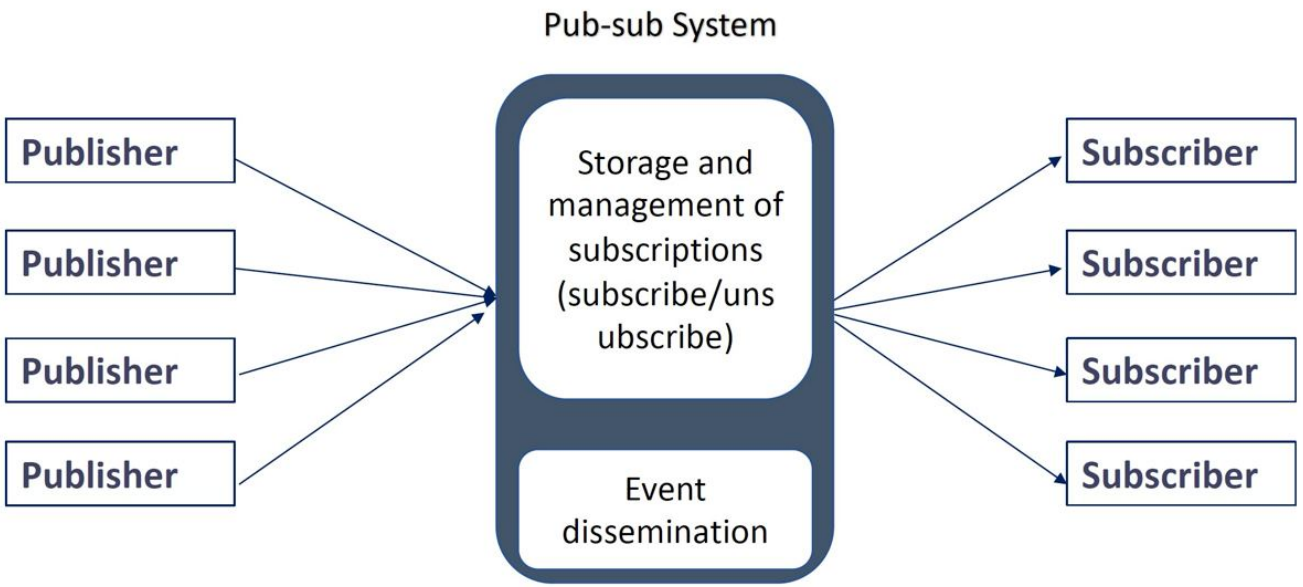
Exercise 1: Publish/Subscribe (Pub/Sub) Systems

- Explain its basic functioning.

Exercise 1: Publish/Subscribe (Pub/Sub) Systems

- Explain its basic functioning.

  Subscribers submit their interest in receiving certain kinds of events (a subscription). Publishers generate events and publish them. In between the two, the broker stores and routes the publications to matching subscribers, which finally consume the data.



Exercise 1: Publish/Subscribe (Pub/Sub) Systems

- What are the main advantages of pub/sub systems in comparison to direct messaging?

Exercise 1: Publish/Subscribe (Pub/Sub) Systems

- What are the main advantages of pub/sub systems in comparison to direct messaging?

  They do not require the endpoints of communication channels to be online at the same time. They can quickly adapt to dynamic environments: for example as new subscribers or publishers join or leave the system, the pub/sub architecture can adapt without requiring significant reconfiguration.

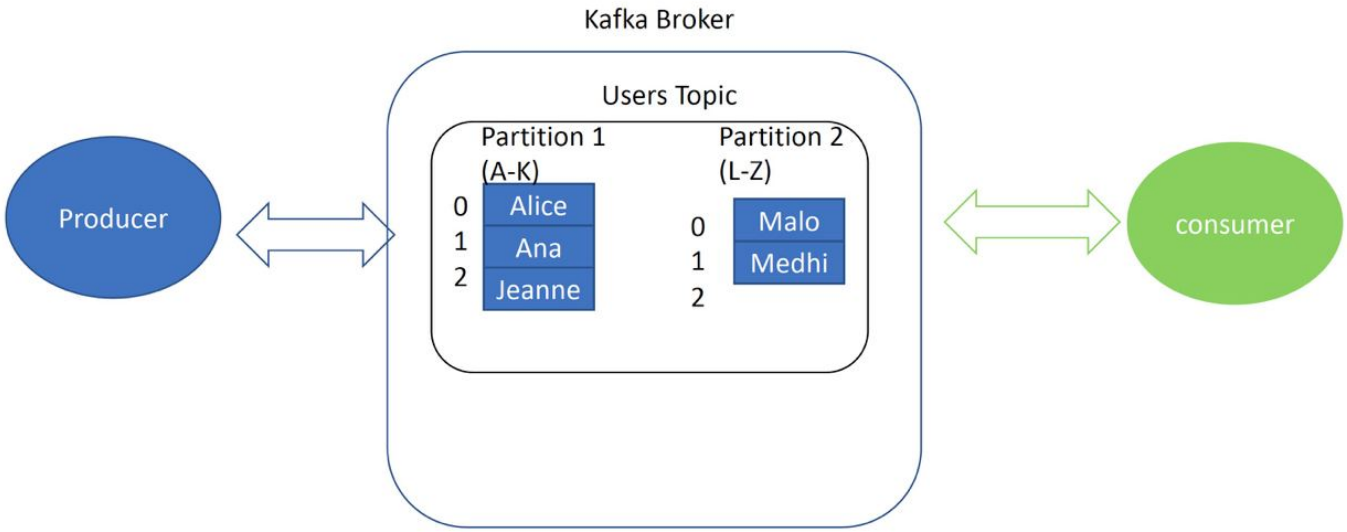
Exercise 2: Concerning Kafka

- List and briefly explain the main roles in Kafka.

Exercise 2: Concerning Kafka

- List and briefly explain the main roles in Kafka.

  Topics are streams of messages of a particular type.
  Producers publish messages to topics.
  Consumers subscribe to topics and read messages.
  Brokers store messages.



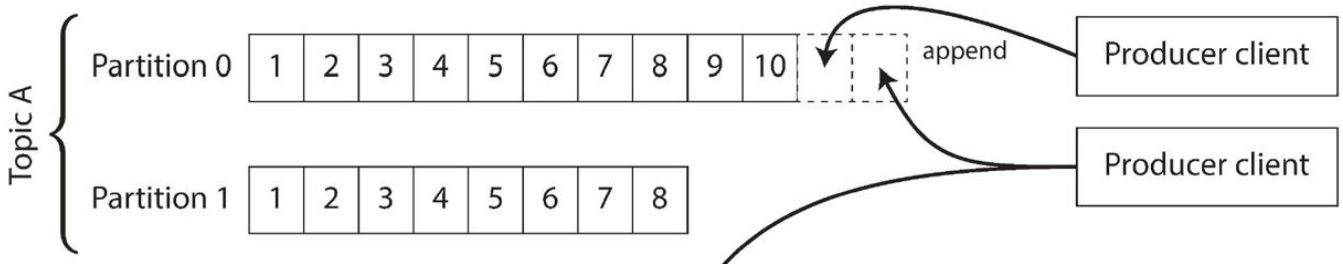
Exercise 2: Concerning Kafka

- What are topics and partitions? What ordering guarantees does Kafka provide?

Exercise 2: Concerning Kafka

- What are topics and partitions? What ordering guarantees does Kafka provide?

  Topics are streams of messages of a particular type. Topics are a logical collections of partitions that are ordered, append-only, and immutable.
  Messages sent by a producer to a particular topic partition will be appended in the order they are sent, so that Consumers see messages in this same order.
  Partitions of a topic may be stored across different brokers, but the order is only guaranteed within a single partition.



Exercise 2: Concerning Kafka

- What is the purpose of the offset?

Exercise 2: Concerning Kafka

- What is the purpose of the offset?

  Ordered messages within partitions are assigned a monotonically increasing number that is called offset. Its purpose is to uniquely identify every message within the partition.

Exercise 2: Concerning Kafka

- What is the role of ZooKeeper in Kafka?

Exercise 2: Concerning Kafka

- What is the role of ZooKeeper in Kafka?

  Kafka uses Zookeeper to detect the addition and removal of brokers and consumers.
  Additionally, it is used to keep track of the consumed offset in each partition.

Exercise 2: Concerning Kafka

- What are the responsibilities of Leaders and Followers?

Exercise 2: Concerning Kafka

- What are the responsibilities of Leaders and Followers?

  Every partition in Kafka has one server which plays the role of a Leader, and zero or more servers that act as Followers. The Leader performs all reads and writes for the partition. Followers, in turn, passively replicate the Leader for fault tolerance. In the event of a failing Leader, one of the Followers will assume the role of Leader.

Exercise 3: Windowing in Stream Processing

- What is Windowing?

Exercise 3: Windowing in Stream Processing

- What is Windowing?

  Data stream is unbound data, which is broken into a sequence of individual tuples.
  A data tuple is the atomic data item in a data stream.
  Windowing is a technique used in stream processing to divide a continuous flow of data into discrete segments or windows, based on time or other criteria, for more manageable processing. A window is a buffer associated with a stream input that groups tuples for processing.

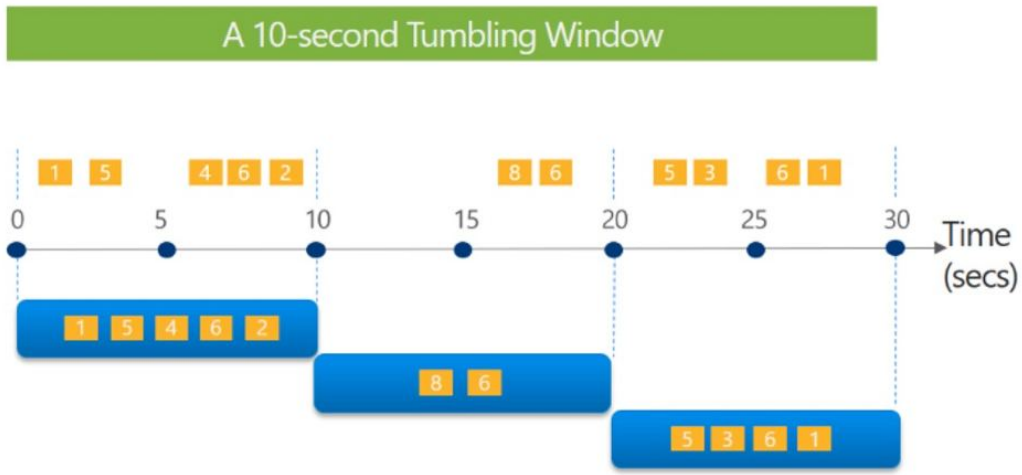
Exercise 3: Windowing in Stream Processing

Give a few examples on how to determine windows.

Exercise 3: Windowing in Stream Processing

- Tumbling window: fixed length, e.g., all events that happen in a timespan of 10 minutes are grouped in the same window. After that, a new non-overlapping window is created to group the events in the next period.

Tell me the count of tweets per time zone every 10 seconds



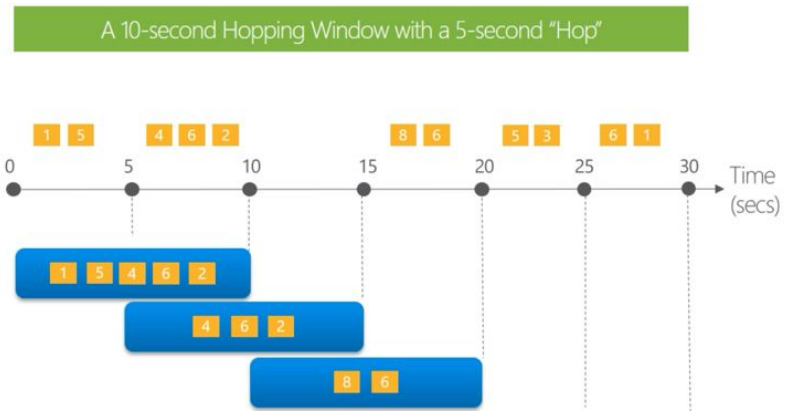
1

[1] Window diagrams adapted from Azure Stream Analytics Windowing

Exercise 3: Windowing in Stream Processing

- Hopping window: like tumbling window, but allows a fixed length overlap (the hop size).

Every 5 seconds give me the count of tweets over the last 10 seconds



2

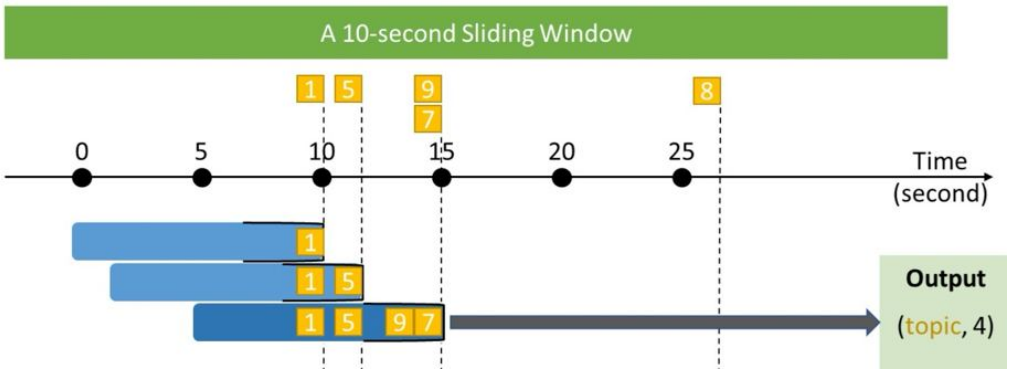
[2] Window diagrams adapted from Azure Stream Analytics Windowing

Exercise 3: Windowing in Stream Processing

- Sliding window: groups events within an interval that continually moves (i.e., slides), e.g., all events in the last 10 minutes.

Alert me whenever a topic is mentioned more than 3 times in under 10 seconds

Note:
- all tweets on the diagram belong to the same topic



3

[3] Window diagrams adapted from Azure Stream Analytics Windowing

Exercise 3: Windowing in Stream Processing

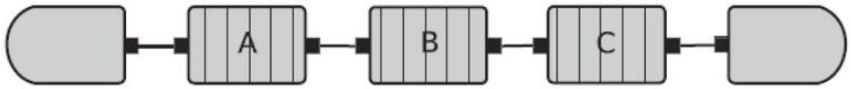
- Session window: groups together events that occur closely in time with no fixed duration, e.g., the clicks of a user in some Web page during a session. The sessions are separated by periods of activity and a specified gap of inactivity, for example a timeout of 30 minutes.

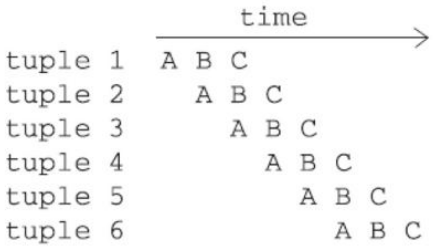
Exercise 4: Task and Data Parallelism

- What is the difference between task and data parallelism?
  This question is about how to scale with increasing the number queries and the rate of incoming events.
  There are different forms of parallelisms, e.g. Pipelined parallelism, Task parallelism, Data parallelism.

Exercise 4: Task and Data Parallelism

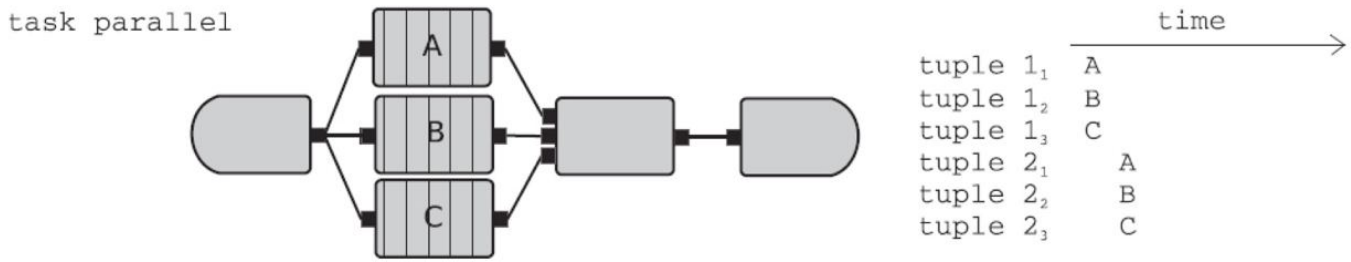
- Pipelined parallelism: Sequential stages of a computation execute concurrently for different data items.





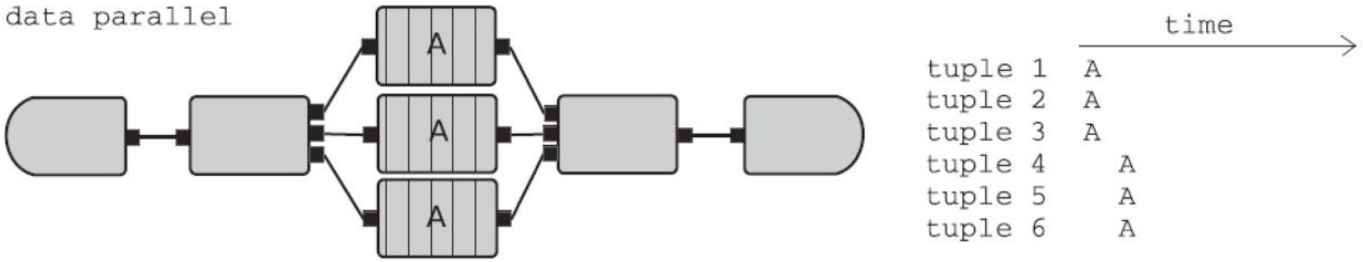
Exercise 4: Task and Data Parallelism

- Task parallelism: distributes different tasks or functions across multiple processing units.



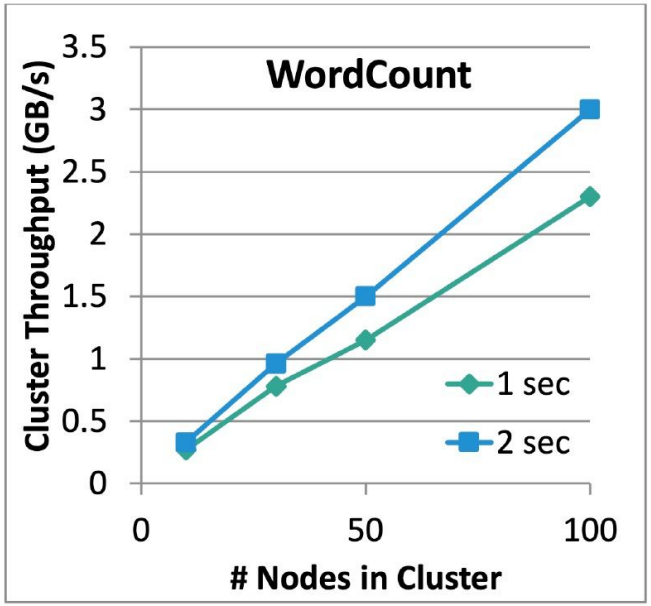
Exercise 4: Task and Data Parallelism

- Data parallelism: applies the same computation to different partitions of data.



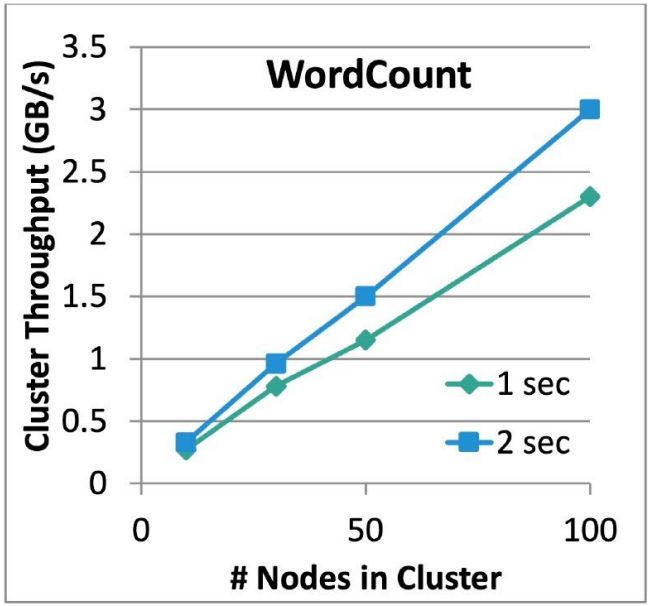
Exercise 5: Impact of Window Size on System Latency

- How does window size affect system latency?



Exercise 5: Impact of Window Size on System Latency

- How does window size affect system latency?



Larger window sizes can increase latency because data is processed less frequently, but they can reduce processing overhead and increase throughput. Conversely, smaller window sizes can reduce latency but may increase processing overhead and reduce throughput.

Exercise 6: True or False?

In Topic-based pub/sub systems, a subscriber can specify filters on key/value attribute pairs of events.

Exercise 6: True or False?

In Topic-based pub/sub systems, a subscriber can specify filters on key/value attribute pairs of events.

> **False** — In Content-based pub/sub systems, subscribers can indeed specify filters based on key/value attribute pairs to receive only the events that match these filters.

Exercise 6: True or False?

Kafka Producers manifest their interest in some topics and pull the corresponding data from brokers.

Exercise 6: True or False?

Kafka Producers manifest their interest in some topics and pull the corresponding data from brokers.

> **False** — Producers publish data to topics, not manifest interest. Consumers subscribe to topics and pull data, as stated.

Exercise 6: True or False?

Kafka users need to check for duplicates, as the system only guarantees at-least-once delivery.

Exercise 6: True or False?

Kafka users need to check for duplicates, as the system only guarantees at-least-once delivery.

> **True** — Due to the at-least-once delivery mechanism, users must handle the possibility of duplicate messages.

Exercise 6: True or False?

Continuous processing-based systems collect data in small groups and take action on each of them.

Exercise 6: True or False?

Continuous processing-based systems collect data in small groups and take action on each of them.

> **False** — The statement describes micro-batch systems. In continuous processing-based systems, each node in the system continually listens to messages from other nodes.

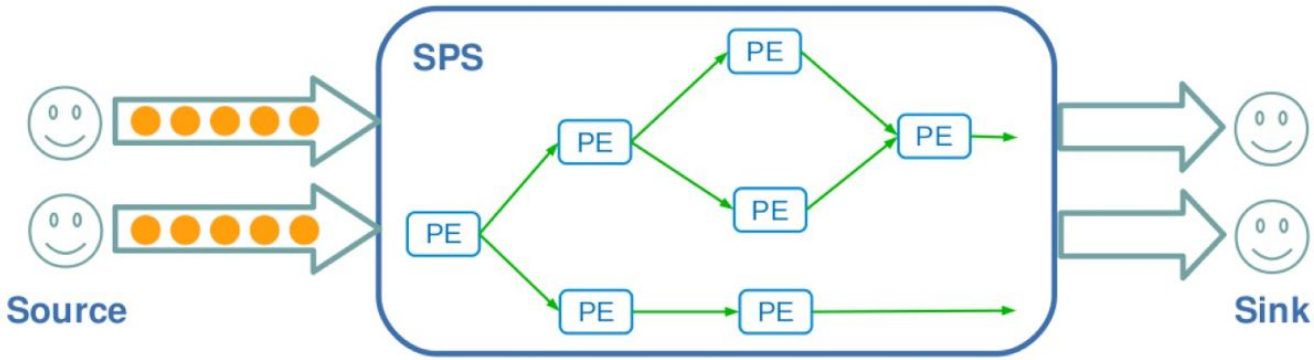
Exercise 6: True or False?

A processing element (PE) operates on input tuples by applying a function on them and outputting other tuples.

Exercise 6: True or False?

A processing element (PE) operates on input tuples by applying a function on them and outputting other tuples.

> **True** — This correctly describes the function of a processing element in stream processing systems. A PE is the basic functional unit in an application. A set of PEs and stream connections, organized into a data flow graph.



Exercise 6: True or False?

Pipelined parallelism consists of sequential stages that concurrently execute a computation on distinct data items.

Exercise 6: True or False?

Pipelined parallelism consists of sequential stages that concurrently execute a computation on distinct data items.

> **True** — Pipelined parallelism allows for the concurrent processing of different stages of a computation, each operating on its part of the data stream.