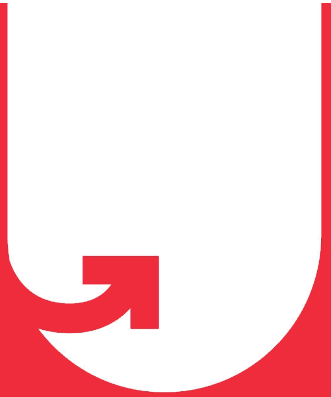
upGrad

Real-Time Data Streaming with Apache Kafka

About

upGrad



upGrad

Course: Data Engineering - II

Lecture On: Real-Time Data Streaming with Apache Kafka

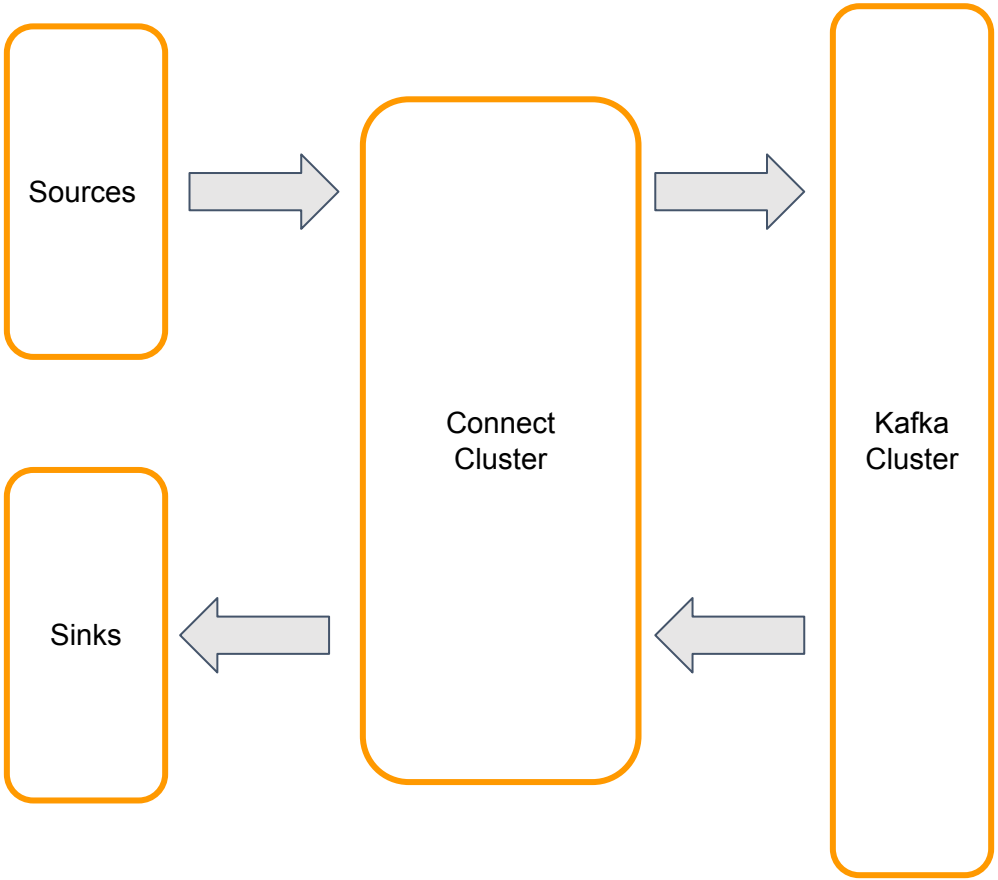
Instructor: Vishwa Mohan

KAFKA CONNECT

01 Framework to connect Kafka with external systems such as databases

02 Data can be moved from external systems into Kafka topics or from Kafka topics to external systems

KAFKA CONNECT



- Utilise Open-source community
- Connectors to known Sources
- Connectors to known Sinks
- Utilise the same cluster for Consumers and Producers

KEY CONCEPTS

01	Connectors - Coordinate data streaming by managing tasks
02	Tasks - Implement copying of data
03	Workers - Run the processes executing connectors and tasks

CONNECTORS

01	Connectors define where data should be copied to and from
02	Source Connectors - Collect data from a system and write to Kafka topics
03	Sink Connectors - Move data from Kafka topics to other systems
04	Several connectors have been open sourced and are available for use

TASKS

01	Implementation of how data is copied to and from Kafka
02	Connectors coordinate a set of tasks which does the actual work of copying/moving data
03	A single job can be broken into multiple tasks. This increases parallelism
04	Rebalancing of tasks happens when a worker fails

WORKERS

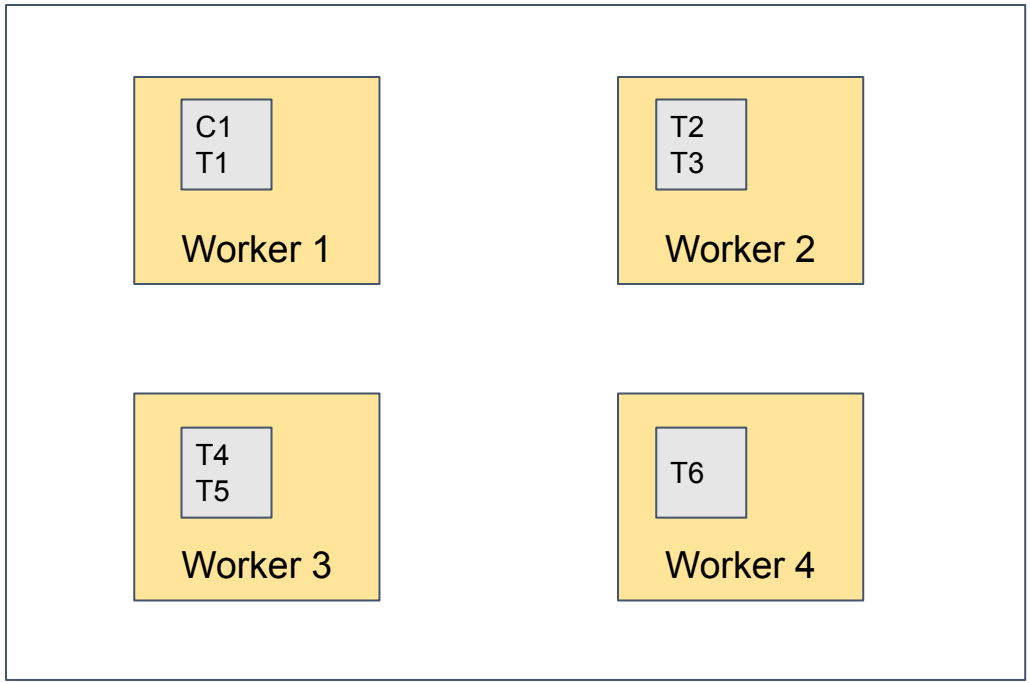
01	Running processes that are responsible for executing connectors and tasks
02	Connect Cluster is nothing but a group of workers
03	Two types - Standalone and Distributed

STANDALONE MODE

01	Single process responsible for executing all connectors and tasks
02	Minimal configuration required
03	Useful for the development and testing of Kafka Connect on local machines
04	No fault tolerance is possible

DISTRIBUTED MODE

01	Many worker processes start
02	Automatically coordinates to schedule the execution of connectors and tasks
03	Fault tolerant as more than one worker is present
04	Rebalancing happens if a new worker joins or a worker goes down similar to that seen in consumer groups



Kafka Connect Cluster

KAFKA STREAMS

01	Client library for building applications and microservices. Input and output data stored in Kafka Clusters
02	API of Apache Kafka
03	Available through a Java library
04	Used to build scalable and fault-tolerant applications

NEED FOR KAFKA STREAMS

01

You have data coming in from an e-commerce website. You need to filter and separate the mobile data and desktop data into two different topics in Kafka.

02

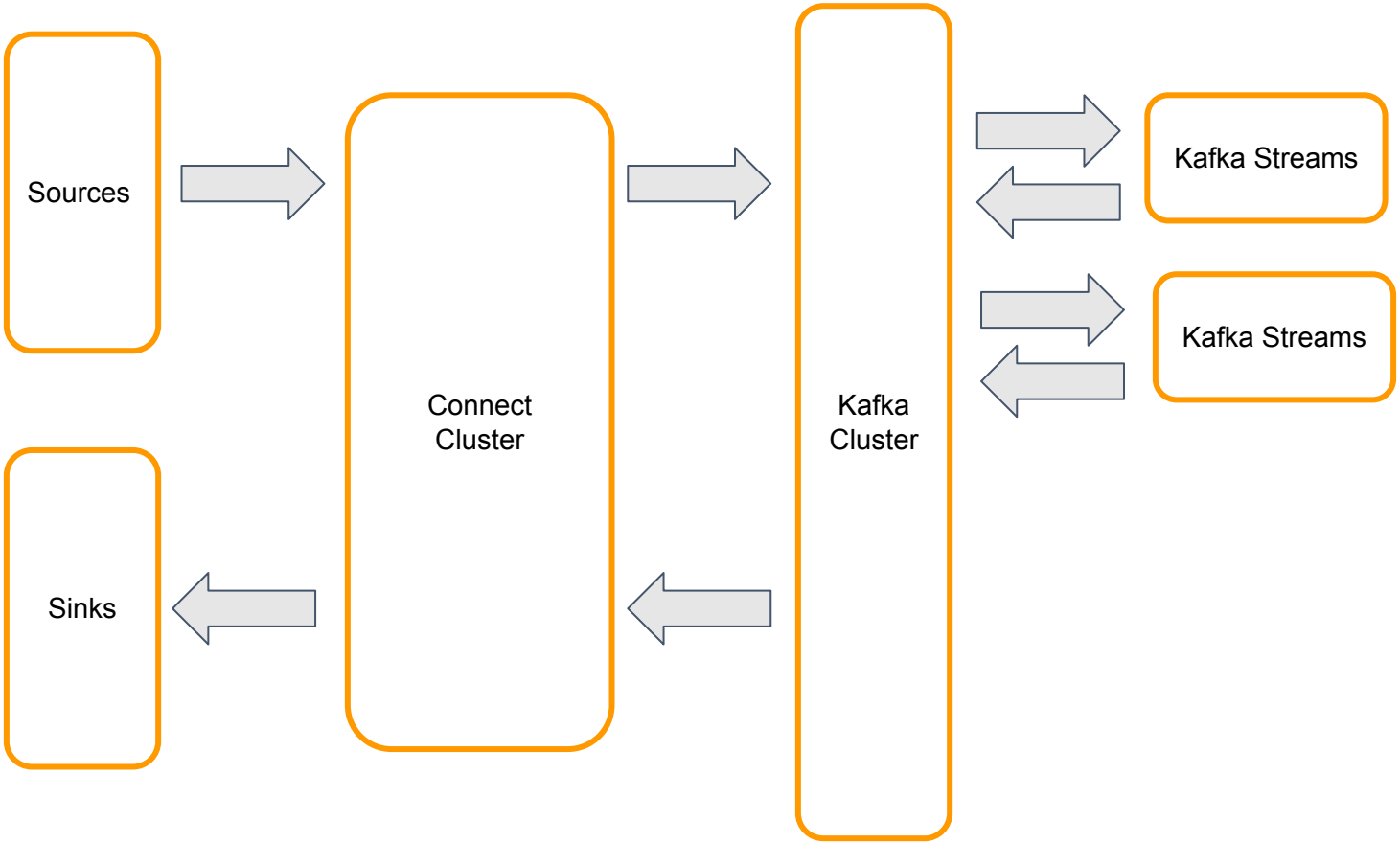
The solution is to have a consumer API read the full data and filter it to mobile and desktop data and then have a producer API that can write it back to Kafka.

03

Problems with this approach

- Excess use of coding
- Consumer and producer APIs are the essential APIs
- Not as friendly
- Difficult to perform

KAFKA STREAMS



KAFKA STREAMS

01

Primarily used for Kafka-Kafka integrations

- Read
- Transform
- Write

02

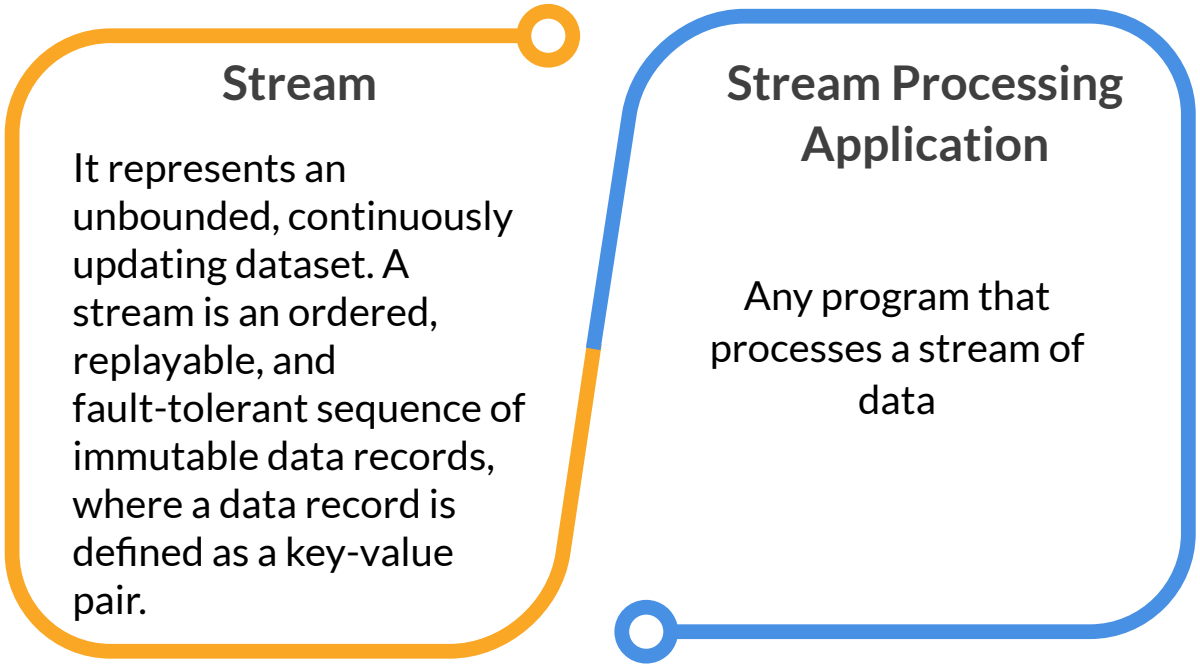
Used for

- Transformations
- Simple processing
- Anomaly detection
- Monitoring

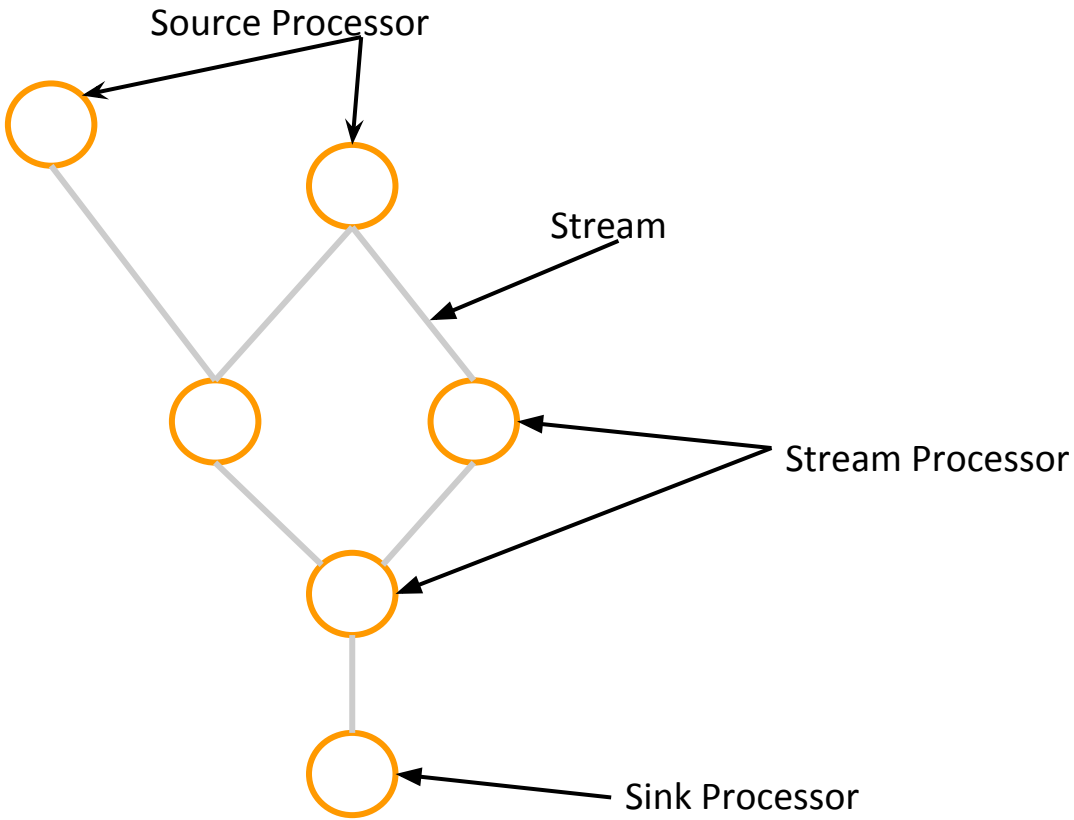
03

Processes one record at a time

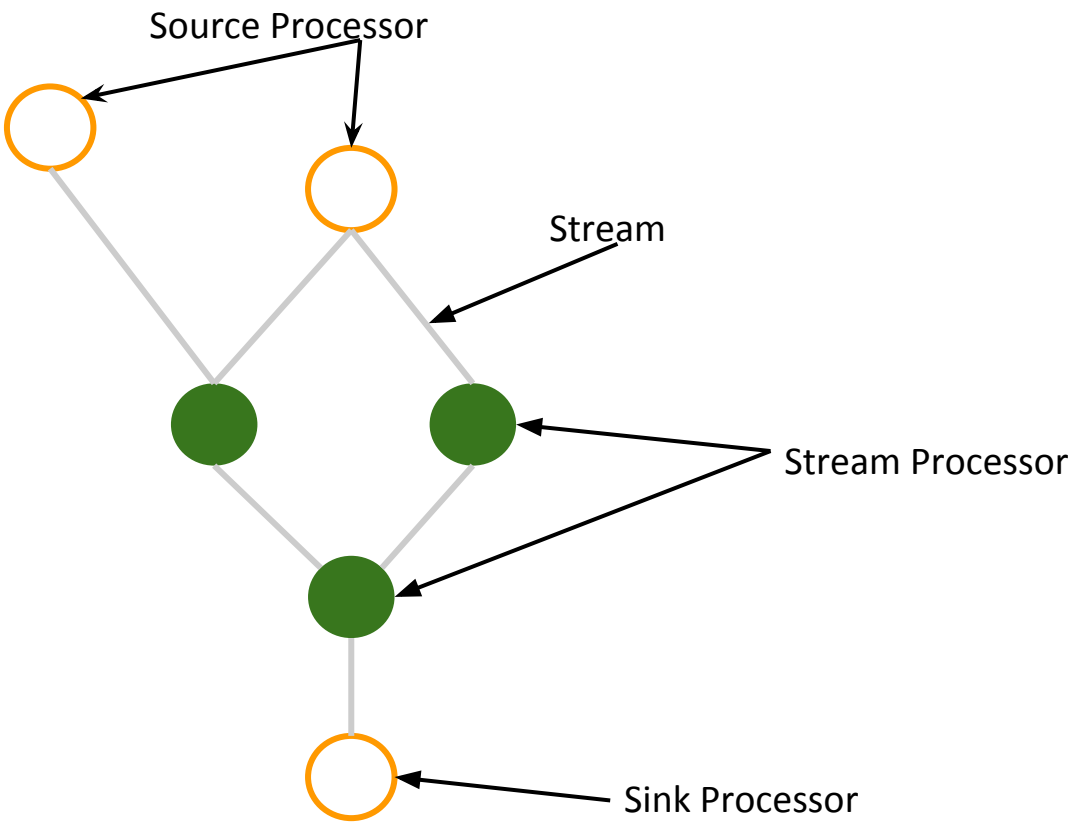
Stream & Stream Processing Application



STREAM PROCESSING TOPOLOGY

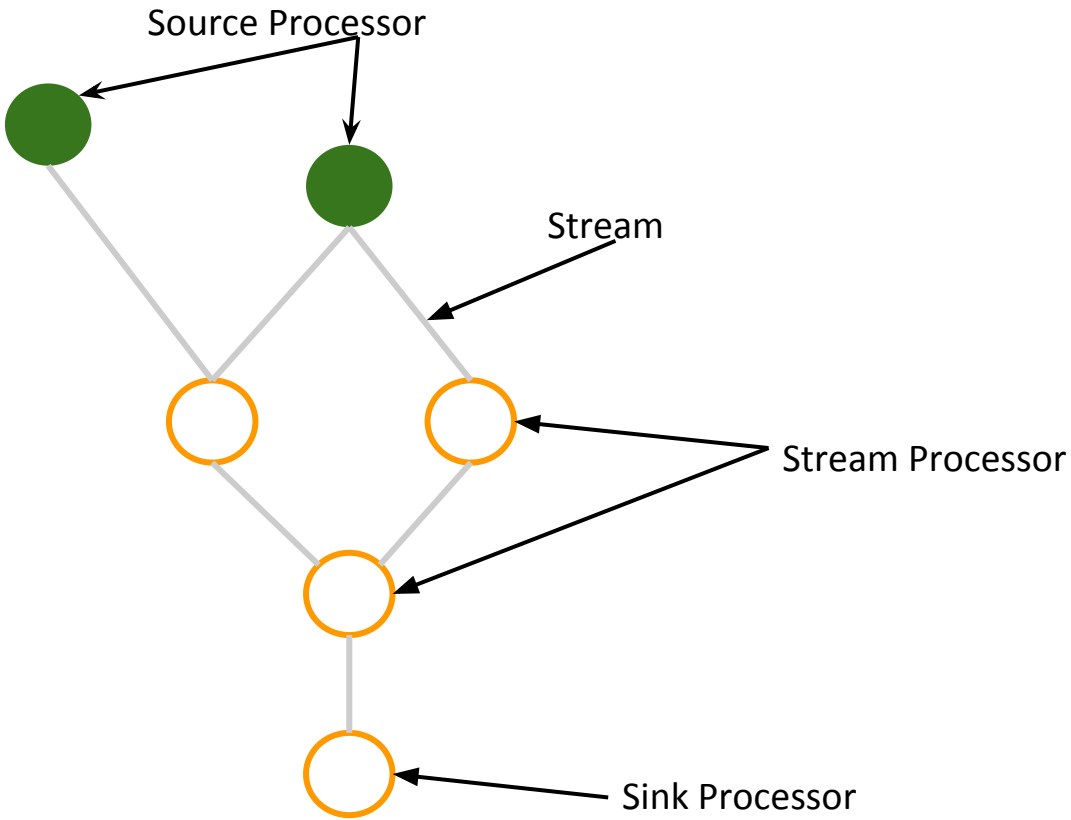


STREAM PROCESSOR



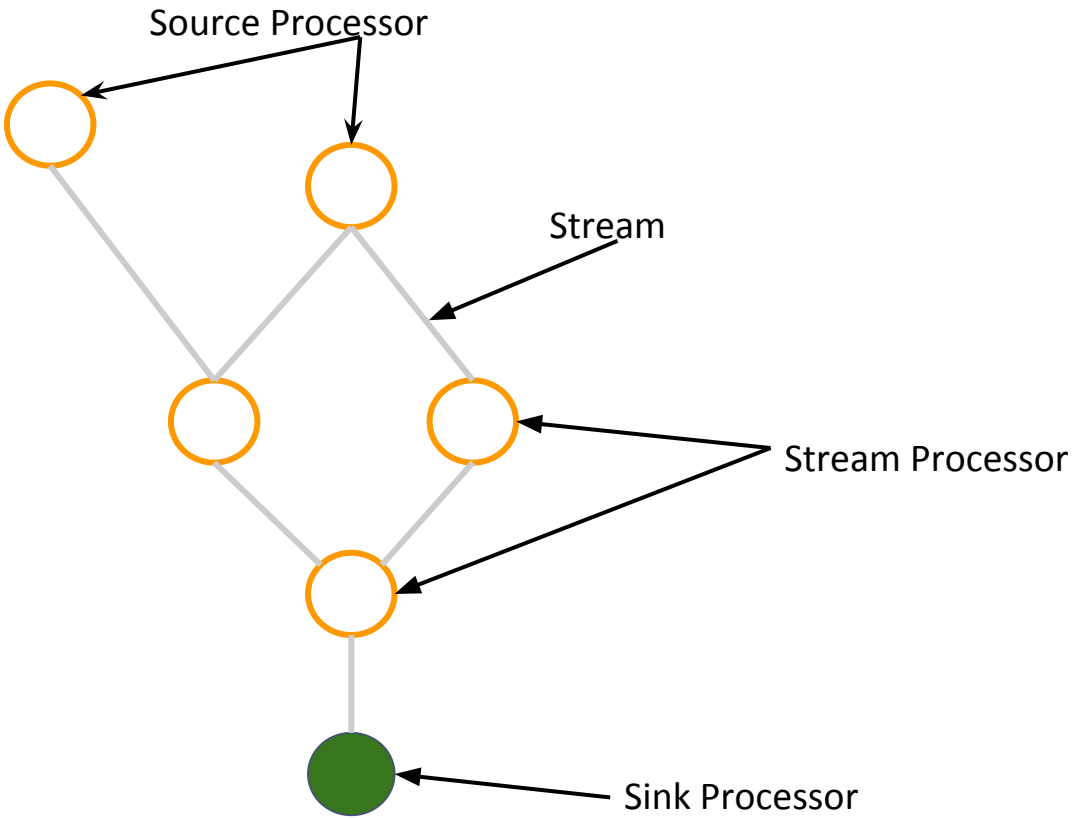
- Node in the processor topology
- Represents a processing step to transform data in streams
- Receives one input record at a time from its upstream processors in the topology
- Applies its operations to it
- Subsequently produces one or more output records to its downstream processors

SOURCE PROCESSOR



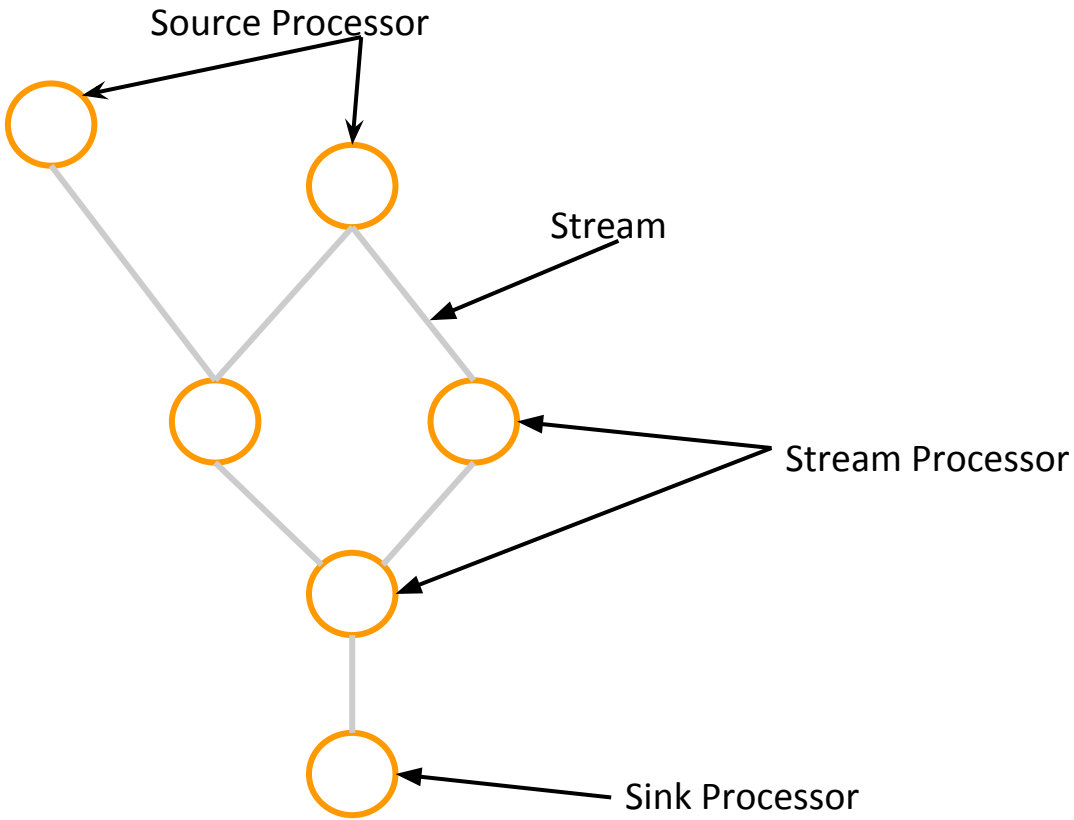
- Special type of stream processor
- Does not have any upstream processors
- Produces an input stream to its topology from one or multiple Kafka topics by consuming records from these topics and forwarding them to its downstream processors

SINK PROCESSOR



- Special type of stream processor
- Does not have any downstream processors
- Sends any received records from its upstream processors to a specified Kafka topic

STREAM PROCESSING TOPOLOGY



upGrad

Thank You