[Apache Kafka](https://kafka.apache.org/" \t "_blank) is a distributed streaming platform designed to handle large volumes of [real-time data](https://builtin.com/data-science/real-time-analytics). It’s an [open-source system](https://builtin.com/founders-entrepreneurship/open-source-future) used for stream processing, real-time [data pipelines](https://builtin.com/learn/tech-dictionary/data-pipeline) and data integration. LinkedIn originally developed Kafka in 2011 to handle real-time data feeds. It was built on the concept of publish/subscribe model and provides high throughput, reliability and fault tolerance. It can handle over a million messages per second, or trillions of messages per day.

Kafka is a critical tool for modern data feeds. As data continues to grow every day, we need tools to handle massive amounts of data. This introduces two challenges: First, how to collect a large amount of data, and second, how to [analyze the collected data](https://builtin.com/data-science/types-of-data-analysis). To overcome these challenges, we need a messaging system.

**WHAT IS KAFKA?**

Apache Kafka is an open-sourced distributed streaming platform designed to handle large volumes of real-time data. It’s become a critical tool for modern data feeds as it helps them transfer data between applications and analyze the data to decide how to share it.

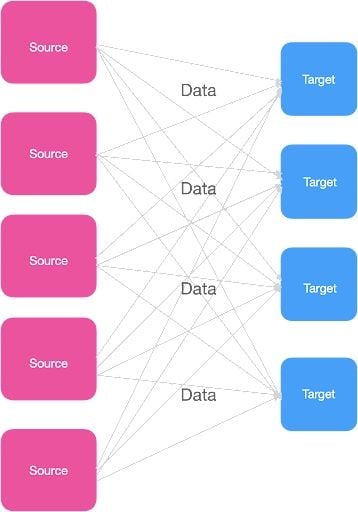
A messaging system helps to transfer data between applications. It helps applications to concentrate on data and the messaging system decides how to share the data.

Let’s take the data pipeline below. We have a source system and a target system, and we exchange the data between them. It looks pretty simple, right?

An example of a simple data pipeline. | Image: Dhilip Subramanian

The source system can be any system such as an app, email, financial data, streaming data etc. The target system can also be any system such as a database, email or analytics, etc. We’ll call them the source and target systems in this article for easy illustration.

What happens if we have multiple sources and target systems, and they all have to exchange data with one another? For example, let’s assume we have five sources and four target systems as below.

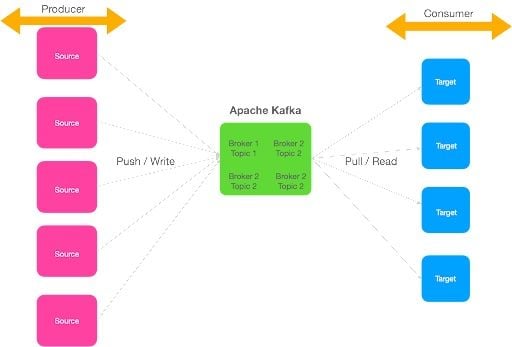
A complex data pipeline. | Image: Dhilip Subramanian

To exchange the data, each source system has to connect with the target system, which results in multiple integrations across the source and target systems. Each integration also comes with various difficulties, as follows:

1. How the data is transported.
2. How to format the data (parsing).
3. Data schema and evolution (shape of the data in the future).
4. Increasing load whenever we connect the source and target system.

It looks pretty messy, right? This is where Apache Kafka comes into the place.

## What Is Kafka?

An illustration of how Kafka can take data from multiple sources and distribute it to the proper targets. | Image: Dhilip Subramanian

Let’s take our earlier example and integrate it through Apache Kafka.

We can see from the image above that Apache Kafka helps us to decouple the source and target system. Source systems are called producers, which can send multiple streams of data to the Kafka brokers. Target systems are called consumers, where clients can read the data from the brokers and process it. Multiple consumers can read the same data; it’s not limited to one single destination. Source and target systems are completely decoupled, avoiding complex integrations.

There are two types of messaging systems companies can use: Point-to-point and publish-subscribe messaging systems. In a point-to-point system, producers persist data in a queue and only one application can read the data from the queue. The message gets removed from the queue once this system reads the data.

In the publish-subscribe messaging system, consumers can subscribe to multiple topics in the message queue and receive specific messages relevant to their application. Apache Kafka is based on a publish-subscribe messaging system.

MORE ON DATA SCIENCE[Differences Between SOQL and SQL Explained](https://builtin.com/data-science/soql)

## How Does Kafka Work?

Kafka is made up of six components. It’s important to understand how each of the components in Apache Kafka work together because they form the foundation of the system and also helps to effectively store and process data streams. These include:

1. **Producer:** A producer generates a large amount of data and writes this into Kafka.
2. **Consumer:** Consumers act as end-users that read data from Kafka that comes from producers.
3. **Topic:** Topic is a category or label on which records are stored and published. All Kafka records coming from producers are organized into topics. Consumer applications read from topics.
4. **Brokers:** These are the Kafka servers that handle the data. Kafka brokers receives message from producers and stores them on its data
5. **Partition:** This is a unit of data storage. It’s a sequence of messages that is stored in a log and is identified by a unique ID, known as the partition offset. Each partition is ordered and immutable, meaning that once a message has been written to a partition, it cannot be modified or deleted. A topic can have multiple partitions to handle a larger amount of data.
6. **Zookeeper:**This is a centralized service that is used to coordinate the activities of the brokers in a Kafka cluster. It is responsible for maintaining the list of brokers in the cluster and facilitating leader election for partitions.

The above six components provide a distributed, scalable and durable system for storing and processing streams of data in Apache Kafka.

================== Details ==================

Other details are available here: [Details](https://builtin.com/data-science/what-is-kafka)