Stream Processing in Modern Enterprise

Peter Nicaj

April 15, 2020

Abstract

Processing big data is one of the most valued departments to help make adjustments and drive a business. Big data processing is handling information that is too large to be handled by a traditional database. The ETL data integration of extract, transform and load was the first developed idea of processing big data information[1]. Batch processing used the ETL structure for processing records historically. the demand for information faster lead to the structure of stream processing or real time processing. Stream processing processes information one record at a time[2]. Stream processing is currently the most used implementation for modern enterprise companies.

1 Introduction

In today's digital economy, many companies are having higher demands for information. The most important information for companies to track is what are consumers most interested in looking at or what things are affecting the business in live time. Many modern companies want to get information received instantly so adjustments can be made to the company. Some examples of information that needs to be processed are bank statements, user interaction websites, and business reports. Hundreds to thousands of records are entered every minute (or seconds) of the day for these examples. These records are too big to be dealt with in a traditional database. When analyzing and processing these huge records it's called big data. Big data is the most important information for businesses to process. The way big data information is handled has changed over the years to bring the idea of stream processing. stream processing is currently the most commonly used processing for big data, but what made it come to be?

2 ETL

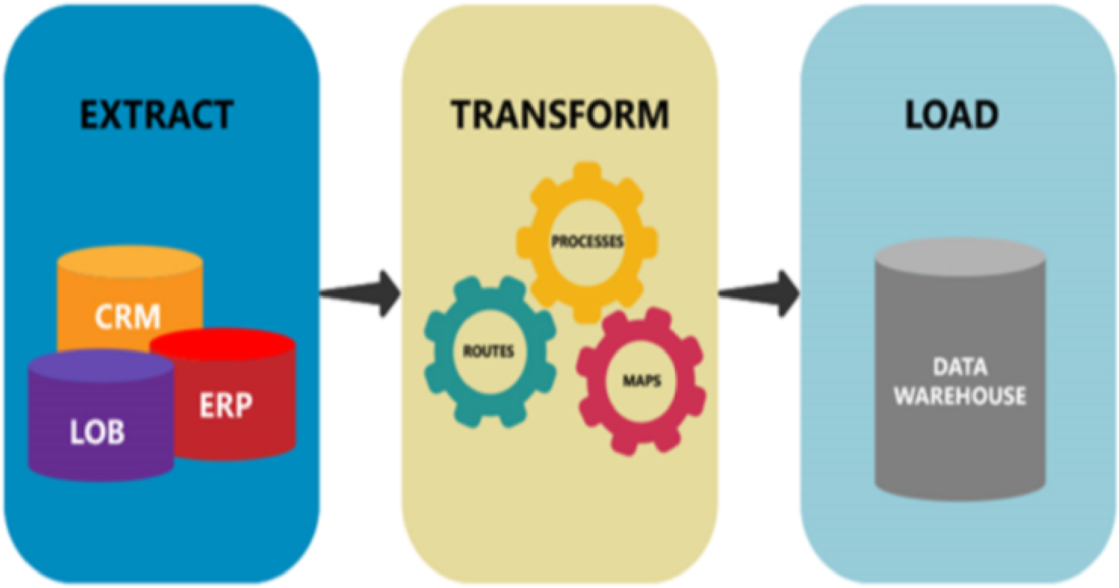


Figure 1: ETL Structure [3]

When big data information needs to be processed its basic data integration is ETL. ETL stands for extract, transform, and load. Extract is the first step where information is pulled from the data sources. Once the information is pulled it now needs to be transformed into a way for a system to understand its data. Once the information is transformed it can be loaded. Load is then loading the transformed information into a way it can be analyzed. The ETL structure is the thought process for processing big data. the important thing to change in the structure is the way that the information is transformed. Over the years the structure has been changed in the way that the information is handled in the transform phase.

3 Batch Processing

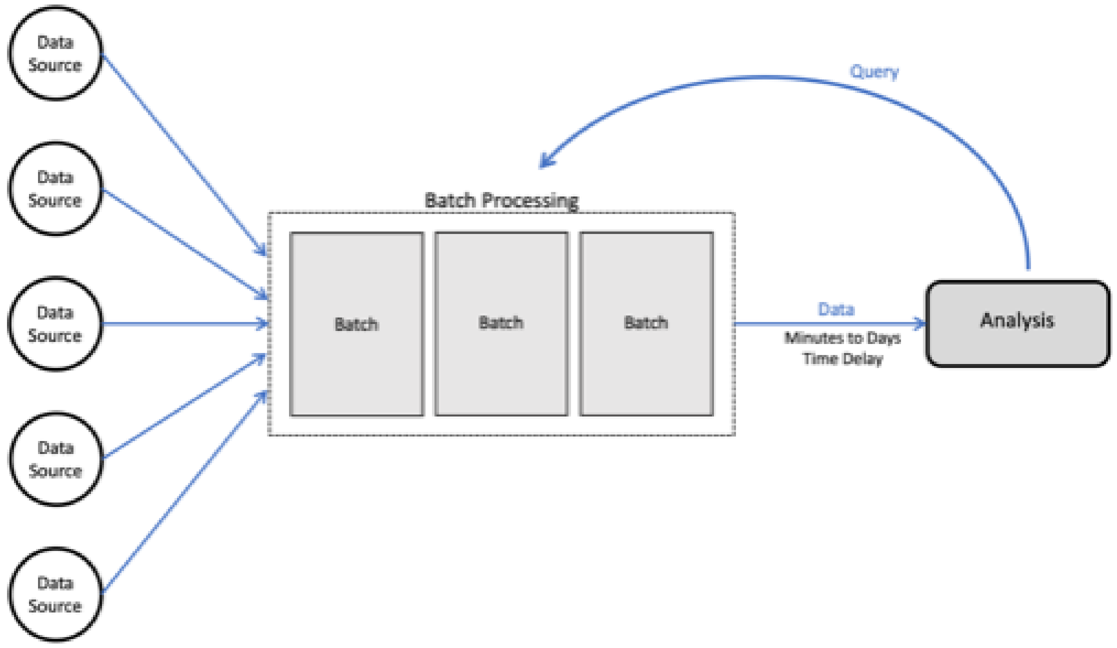


Figure 2: Batch processing structure [4]

The batch processing approach was the first developed structure for a system to handle the data. Batch processing is used to process data historically or periodically[1]. These processes execute on predetermined intervals (e.g, hourly, daily, weekly etc) every time new data wants to be received.

Figure 2 shows how information is passed by using batch processing. The information is pulled from the database, processed, and than sent out in minutes to days. The reason why the process takes time is because data is collected over time then processed in batches. an example of using batch is for business reports that need to be run to deliver at the end of the day.[5] Batch processing is best used when taking data from a static data set and transferring it without manipulating the data. The next issue rose of how do you analyze information while being transformed? In order to emulate live processing with this setup you would need to repeatably execute these jobs to stay up to date. Most batch processing platforms are not designed to handle this workload. Which then leads to the development of the stream processing structure.

4 Stream Processing

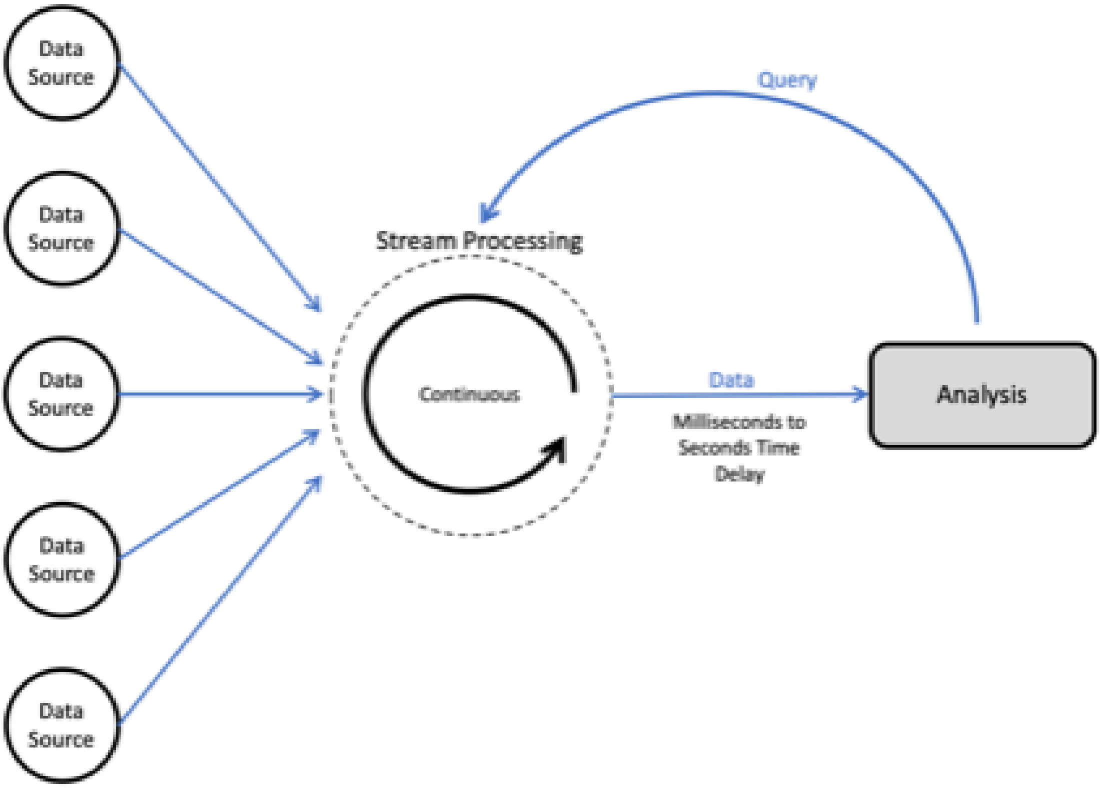


Figure 3: Stream processing structure [4]

Stream processing is handling data that is in motion[6]. As each record is extracted it is processed immediately. Information is being pulled from the data source then processed and loaded at the same time. Batch processing would require a wait time for the process to run to look at the information. in streaming information is able to be looked at while being updated.

Figure 3 shows the structure of how data is handled with stream processing. a record is collected then processed immediately and repeats. The processes are handling these data so fast the consumer is able to poll (which will be covered in more detail below) the data in seconds and even milliseconds. Whereas batch processing could take hours to days to process its data and make available to the consumer. Batch processing is different because it gives a time block to let data be collected.

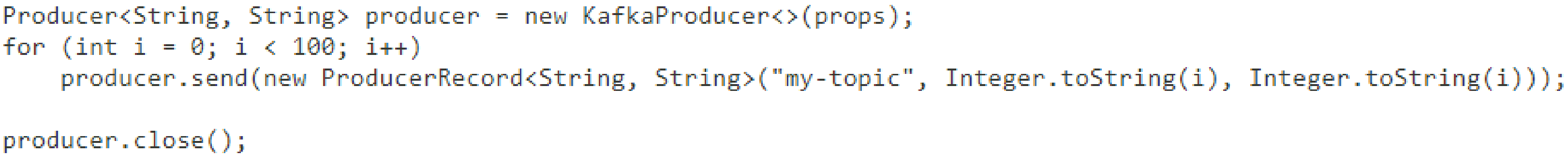
The reason of why modern enterprises are making the transition to real time is because the information can be received much faster. The faster the enterprise can receive their data and analyze it, the quicker the businesses are able to make adjustments or process the data. A useful case for streaming is fraudulent detection. Using real time data you are able to catch any abnormal events that are occurring with your data as it's being updated.

5 Example Platform

A great way to see how stream processing is handled is looking at a system using stream processing. one of the most common stream processing platforms is Apache Kafka. Kafka came to be from LinkedIn developers. The system is designed in Java and Scala for building real-time data pipelines and streaming apps. Kafka is designed by getting the data that needs to get analyzed then placing it into a Kafka cluster. When the data is placed into the cluster it is replicated across multiple nodes which form the Kafka cluster, these nodes are called partitions. The data that is fed into the cluster is the production (or producer) data. The production data is then processed and broken up into topics. Topics, in Kafka, are section names of where processed data is stored. When data is processed and distributed to the topics they are called messages (or records). On the other end, the data that is shared with the consumer after being analyzed is the consumption data.

Confluent a company that uses Kafka created a demo to show how stream processing is used.[7] the Kafka demo uses stream processing to monitor real time Wikipedia edits. The production data for the demo is all the edits for Wikipedia coming into the data source. When an edit is made on Wikipedia that information is sent to the Kafka cluster immediately. The information is then divided up and processed into different partitions. In the demo you are able to look at all the different topics that were gathered from the Wikipedia edits. After running the demo, it showed that live edits information on different topics can still be viewed while other information being processed. With the system handling large data sets, I was concerned of what an application would look like for processing the information.

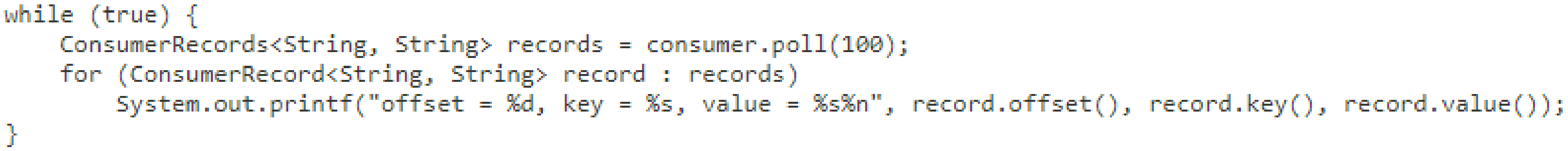
Kafka has a Javadoc that can be used for sending and receiving information for a Kafka cluster. The first important aspect to understand is how the information is sent to a Kafka cluster. Using the KafkaProducer class, a simple example is shown how records are sent to a Kafka cluster.[8] The `send()` method in the below section of code can be seen being used to send a list of records (or messages) to the Kafka cluster:



```
Producer<String, String> producer = new KafkaProducer<>(props);
for (int i = 0; i < 100; i++)
    producer.send(new ProducerRecord<String, String>("my-topic", Integer.toString(i), Integer.toString(i)));

producer.close();
```

Now that the messages are sent to the cluster the information in the Kafka cluster can be consumed. The consumer API shows how a application would consume the new data that was produced into the Kafka cluster.[9] According to the section of code below, the system reads a list of records:



```
while (true) {
    ConsumerRecords<String, String> records = consumer.poll(100);
    for (ConsumerRecord<String, String> record : records)
        System.out.printf("offset = %d, key = %s, value = %s%n", record.offset(), record.key(), record.value());
}
```

The `poll()` method is used to consume the list of records to the consumer. This is usually configured with an integer value (e.g, 100). The messages are usually polled and committed within a few hundred milliseconds. The issue rises for stream processing with efficiently storing your information.[10] The system is dealing with so many records that some information has to be deleted. A typical stream system would run a process to delete the oldest records so more room is made. Since the new information is more important than the old the older messages would be deleted. the oldest records are deleted to continuously have the system run so there isn't any overflow. Some of the reasons of why stream processing is so important is because it's what most modern companies are concerned of. Most modern companies are making the migration to streaming, and others are not. Some companies are not making the migration to streaming because its not beneficial to make the migration from batch. More people are required to monitor the processes to make sure all information is correct. Since streaming is continuously running you have to monitor it 24/7. Going to a real-time system depends on demand of information.

Any application or software that is having constant interactions with a user is a great example of using stream processing. Some applications that use stream processing are Facebook and Twitter. the information that is handled in real time are things that people are posting on social media. The type of information that would be of interest for processing are the trending topics that users are mentioning about on their respective social platform. When a company can get the information of what people are talking about on social media they are then able to make adjustments to their targeted audience immediately.

There are thousands of credit card companies and websites that have to worry about fraud. Companies like JP Morgan Bank and Google use stream processing for fraudulent detection because the information can be produced quickly. Stream processing has given the ability to track real time data for any unusual activity. Messages are sent when unusual actions are occurring with the topic. The data is constantly being analyzed and when there is a common change or suspicious behavior a message is received.

6 Conclusion

The ETL structure is still being manipulated till this day to meet companies demands of data process. Batch processing was the first developed structure, but isn't fast enough for most companies needs. Streaming allows for data to be collected and distributed at a very fast rate. The capabilities that streaming has given modern companies has increased the time of enterprise competition adjustments. Companies like Audi, JP Morgan Banks, and many other high level businesses are making the migration to real time analysis. The next solution to find for stream processing is efficiently storing the data, so the most amount of data could be held without servers crashing.

References

[1] Aljoscha Krettek. The data processing evolution: A potted history.

[2] SQLstream. Why is stream processing so fast?

[3] Nida Fatima. Data integration and etl: What it is and how to select the right etl tool.

[4] Eran Levy. Batch, stream, and micro-batch processing: A cheat sheet.

[5] John Spacey. 5 examples of batch processing.

[6] Ververica. What is stream processing?

[7] Confluent. Confluent demo.

[8] Kafka. Class kafkaproducer.

[9] Kafka. Class kafkaconsumer.

[10] Ivan Mushketyk. Introduction to stream processing.