University of Salford MANCHESTER

SCHOOL OF SCIENCE, ENGINEERING & ENVIRONMENT

Big Data Tools and Techniques

Week 6

Real-Time Magic

Stream Processing & Structured Streaming in PySpark

2025

Expectations

1. Choose a quiet place to attend the class and please concentrate during the lecture.
2. Put your questions in Padlet and I will review them in the due time (Padlet link is in BB, week 6, Lecture folder for Q&A week6).
3. You can find a handout on BB.
4. We will have 5 mins break after the first hour of the lecture (please remind me).
5. Jisc code will be shared during the break time.

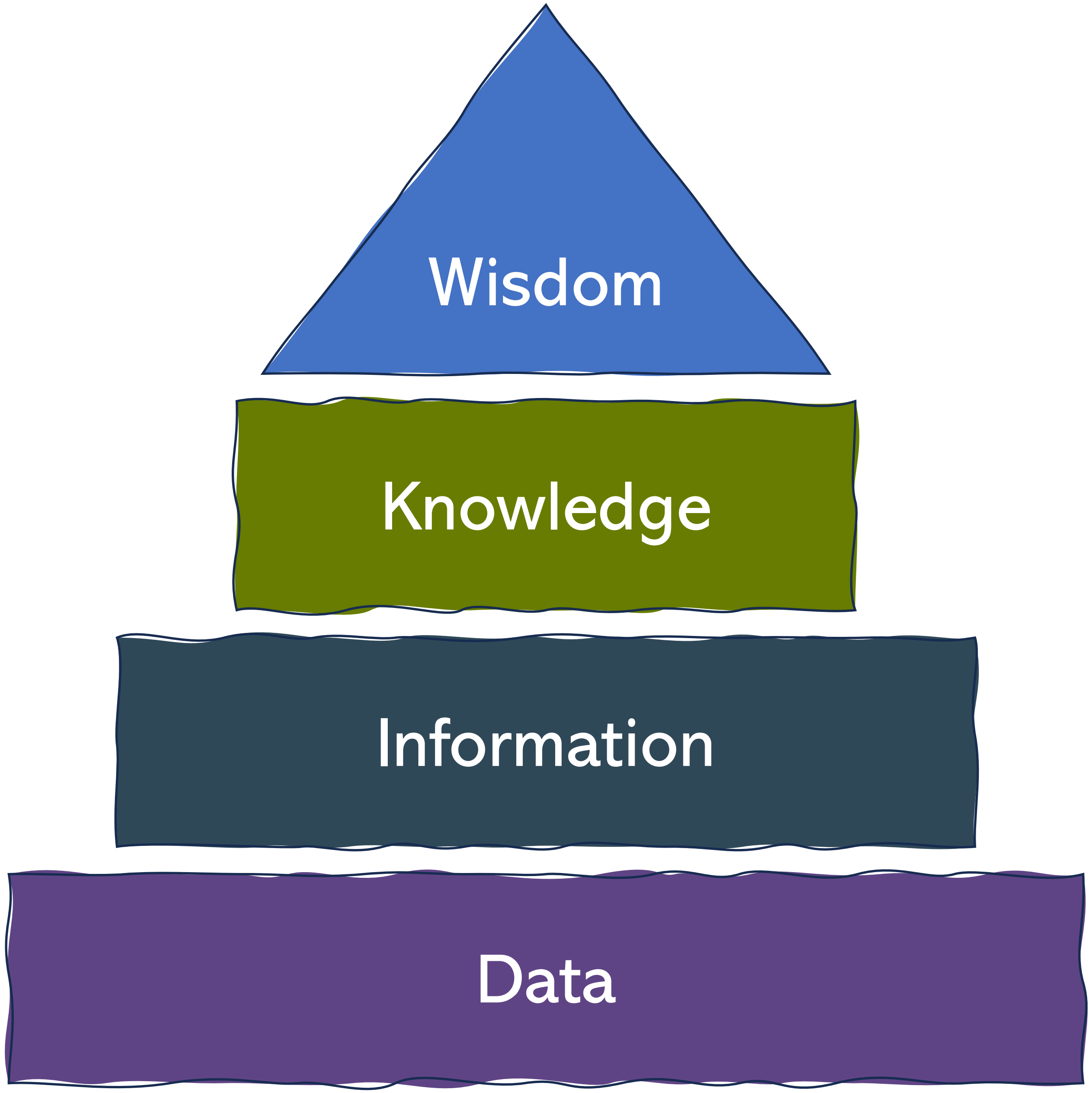
Learning Outcomes

1. To recognize the stream processing

2. To describe differences between batch and stream processing

3. To apply Apache Spark Structured Streaming in different use cases.

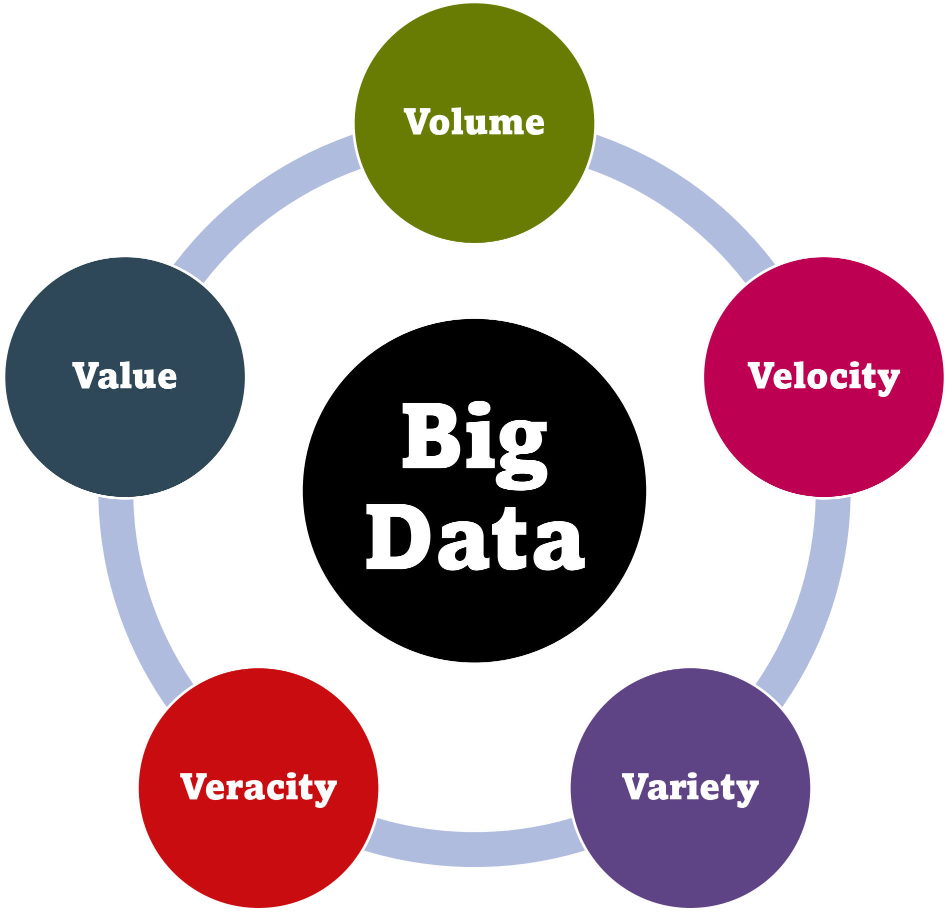
Recap


Wisdom
Knowledge
Information
Data

Different Types of Analysis

Types of Analysis	Questions	Value	Complexity	Example
Descriptive	What happened?	Summarizes past performance.	Low	Monthly sales report.
Diagnostic	Why did it happen?	Identifies causes of past events.	Moderate	Analysing drop in sales for a region.
Predictive	What will happen?	Forecasts future outcomes.	High	Predicting product demand for the next quarter.
Prescriptive	What should we do about it?	Recommends actions to achieve goals.	Very high	Optimizing inventory levels for the predicted demand to avoid stockouts.

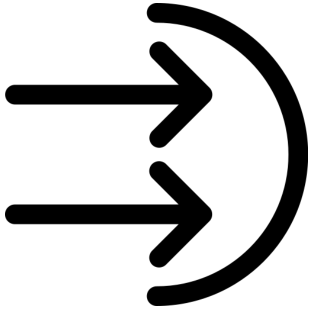
Characteristics of Big Data (The 5 Vs)



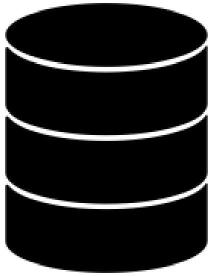
Data Analytics

Static Batch Data

Allows comprehensive processing at scheduled intervals



Input data



Historical datasets

Modelling

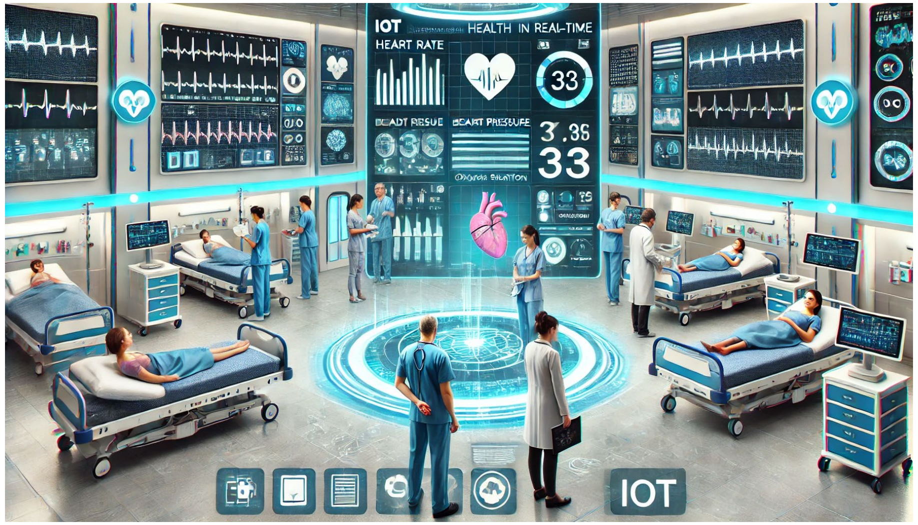
Insight

Question?

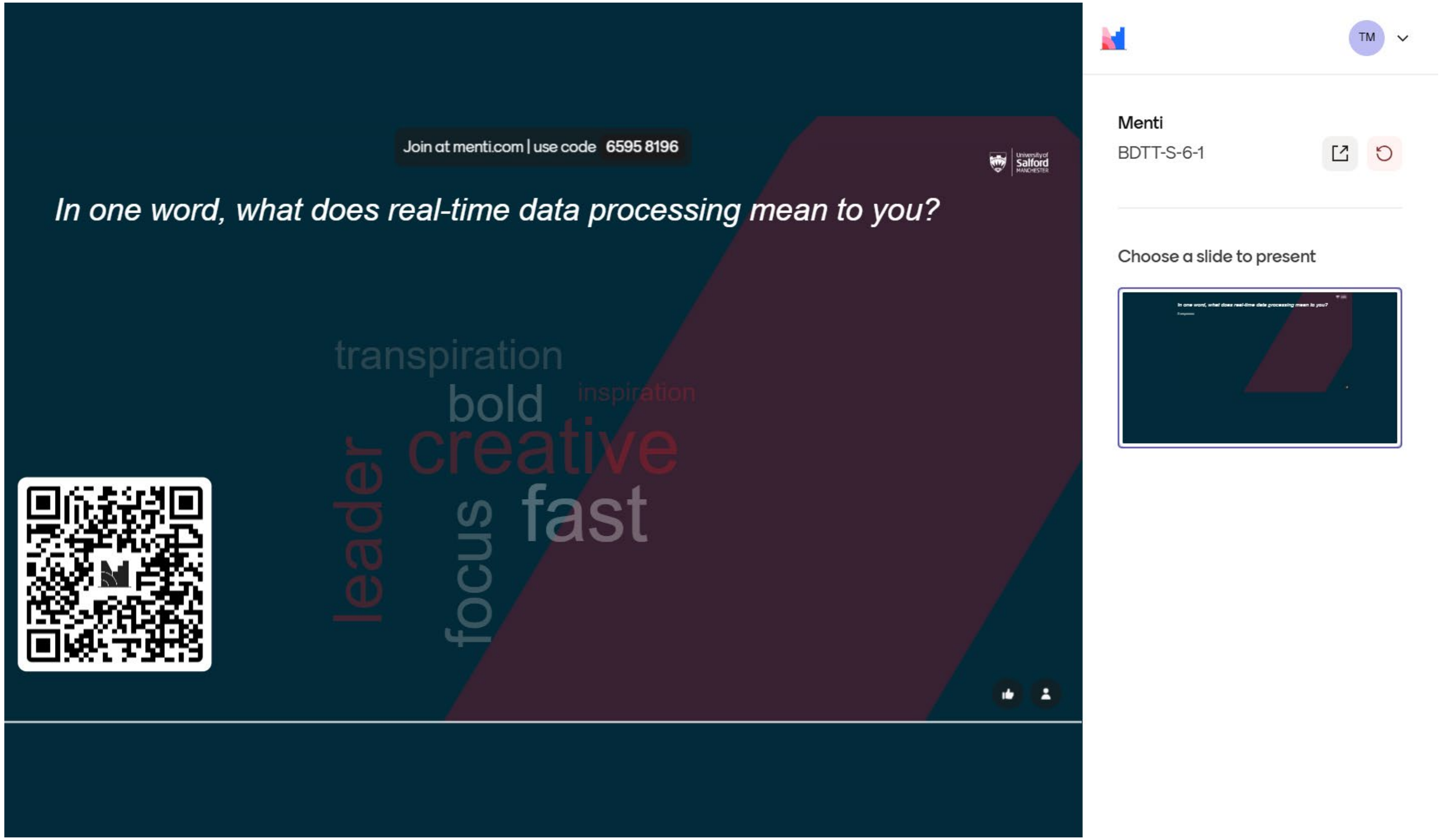
Does this approach work for every applications?


BANKING SECURIT
REAL-TIME


IOT

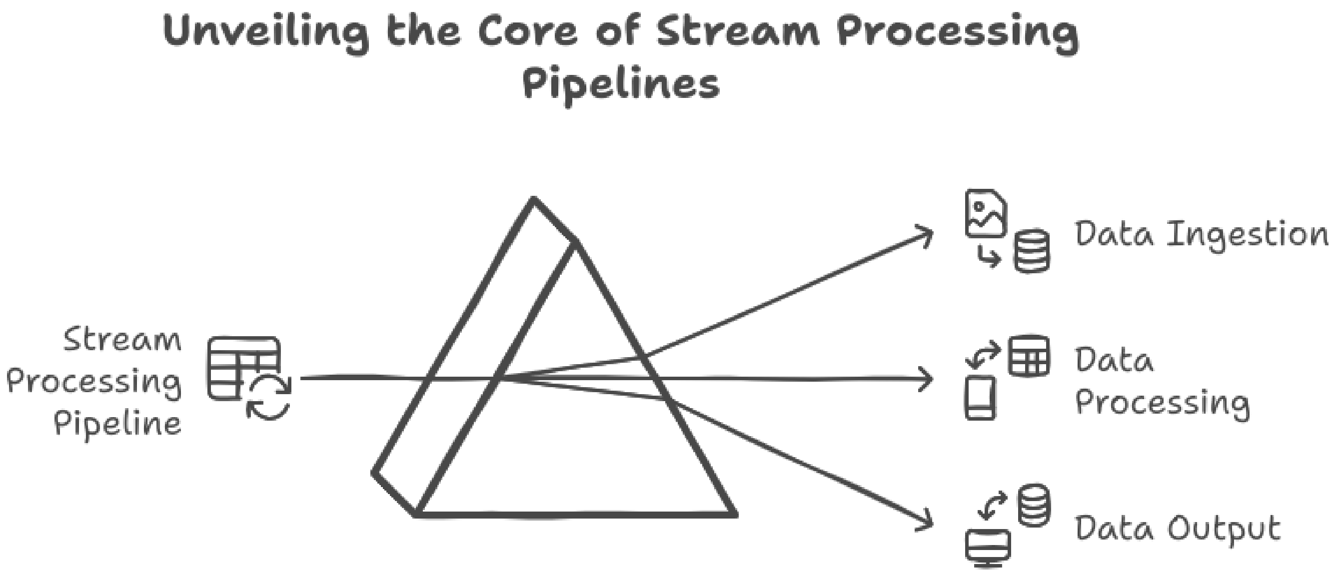

IOT
HEALTH IN REAL-TIME
HEART RATE
33
IOT

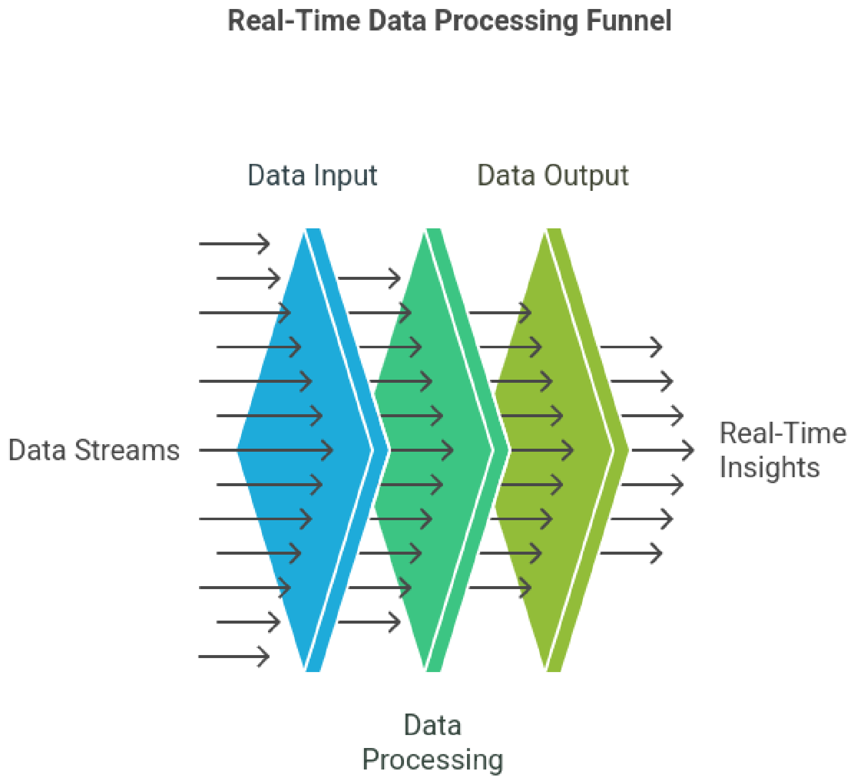
Activity


Join at menti.com | use code 6595 8196
In one word, what does real-time data processing mean to you?
transpiration
bold
inspiration
creative
leader
focus
fast
Menti
BDTT-S-6-1
Choose a slide to present

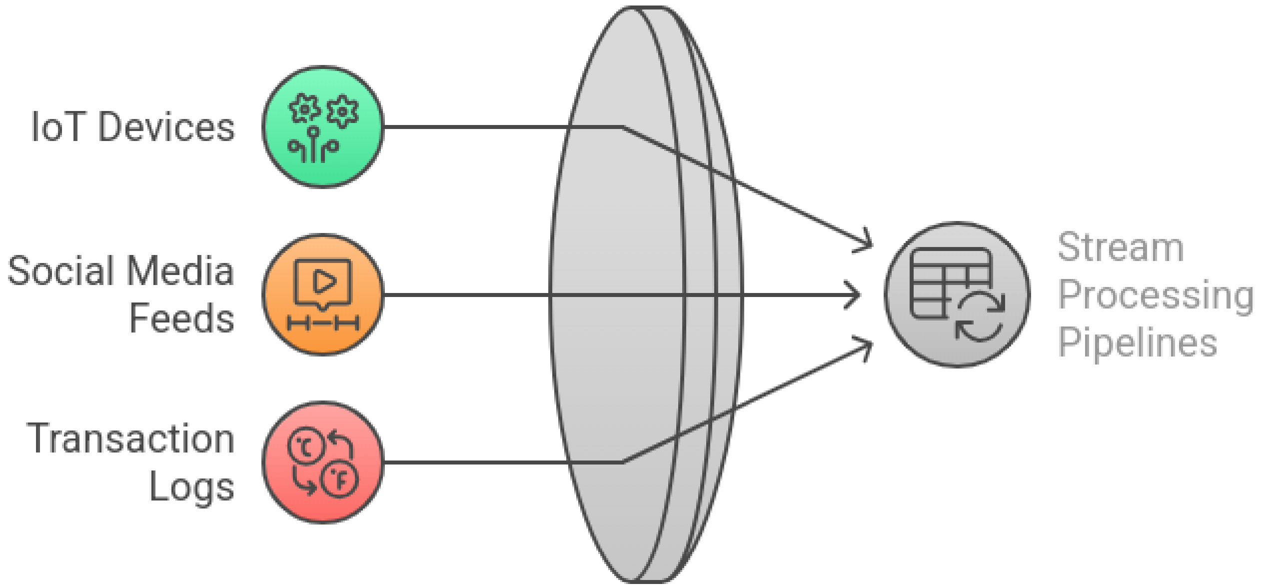
Stream Processing

Stream processing

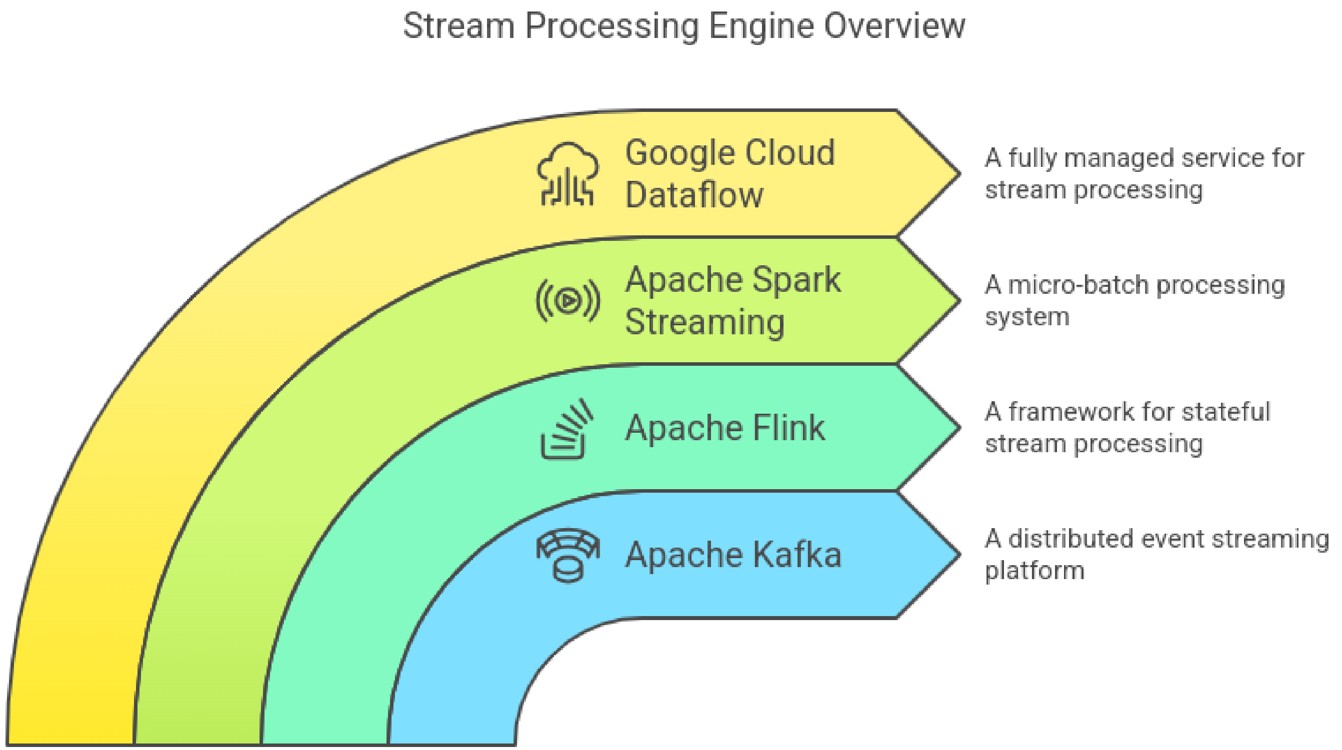




Data Sources



Engines


Stream Processing Engine Overview
Google Cloud Dataflow
A fully managed service for stream processing
Apache Spark Streaming
A micro-batch processing system
Apache Flink
A framework for stateful stream processing
Apache Kafka
A distributed event streaming platform

Data Storage

How should processed data be stored?

Databases

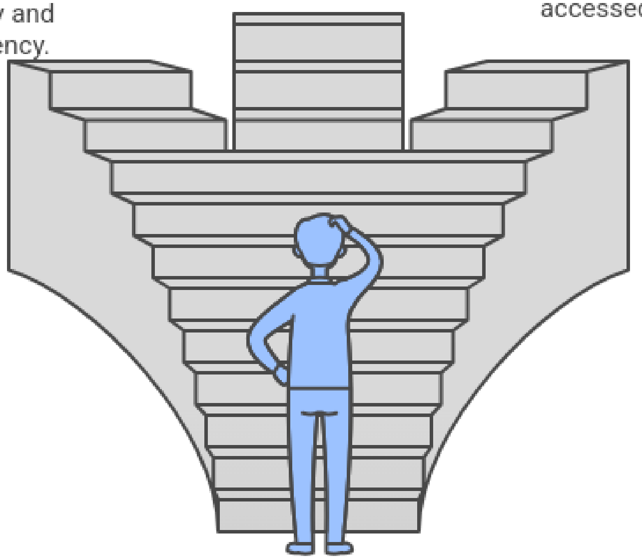
Suitable for structured data and transactional processes, offering reliability and consistency.

Data Lakes

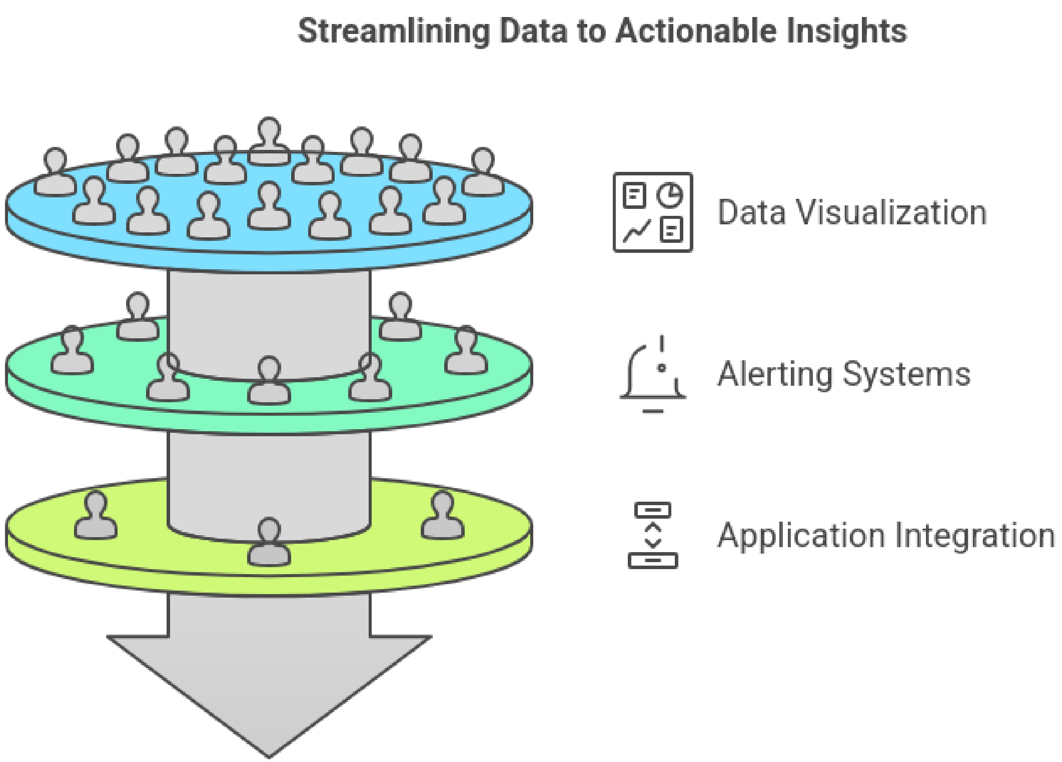
Ideal for storing large volumes of unstructured data, providing flexibility for future analysis.

Cloud Storage

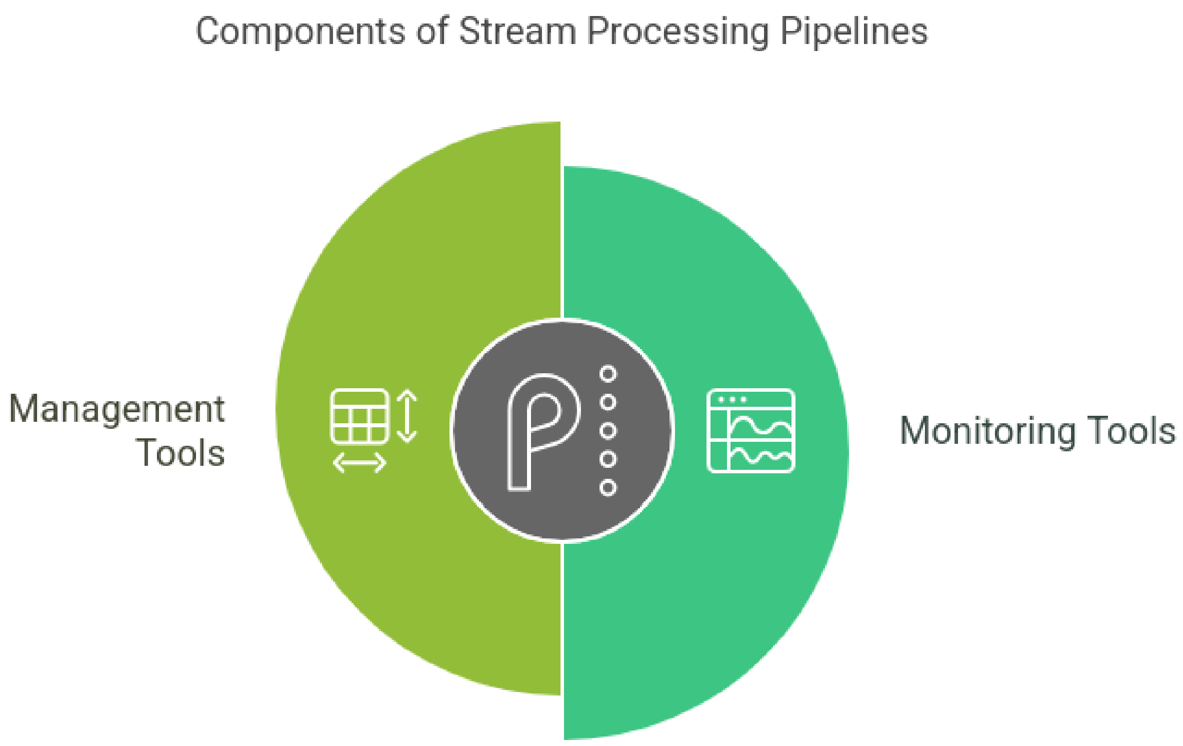
Offers scalability and accessibility, allowing data to be stored and accessed remotely.



Data Sinks


Streamlining Data to Actionable Insights
Data Visualization
Alerting Systems
Application Integration

Monitoring and Management



Key Features

- Real-time Processing
- Low Latency
- Data Streams
- Stateful Processing
- Event-Driven
- Windowing

Key Challenges

- Complexity of Handling Late Data
- State Management
- Resource & Performance Tuning
- Increased System Complexity
- Data Quality

News

- [Wes Streeting unveils plans for ‘patient passports’ to hold all medical records]


News Opinion Sport Culture Lifestyle

The Guardian

UK

UK US politics World Climate crisis Middle East Ukraine Football Newsletters Business Environment UK politics **Society** Science Tech Global development Obituaries

NHS

This article is more than **3 months old**

Wes Streeting unveils plans for ‘patient passports’ to hold all medical records

Health secretary launches consultation on government’s move to transform NHS in England from ‘analogue to digital’

Pippa Crerar *and* Denis Campbell

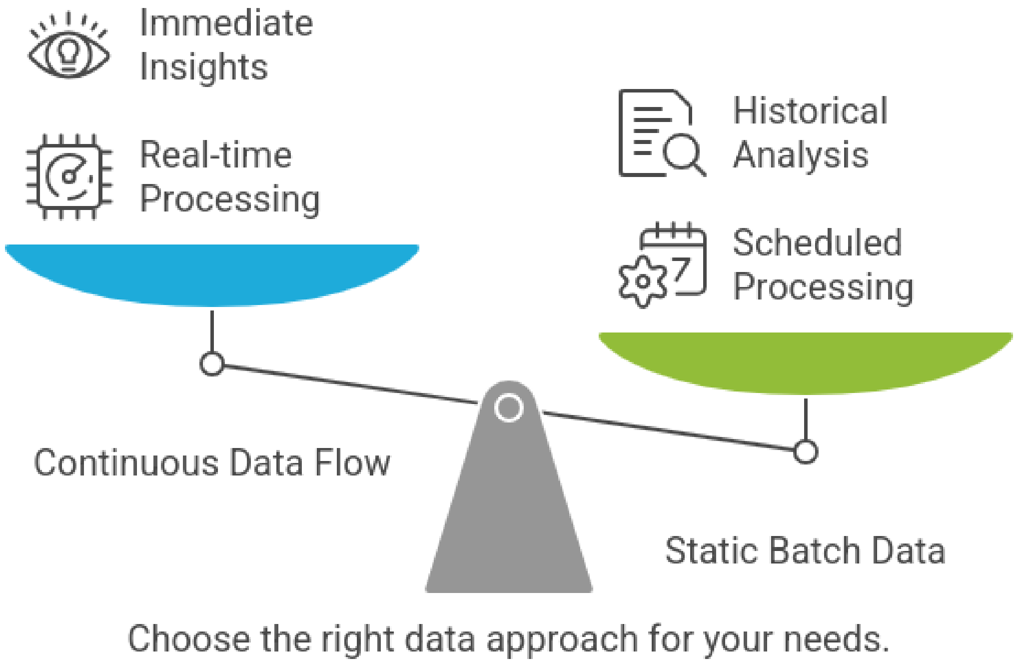
Use Case - Real-Time Health Monitoring for Chronic Disease Management

- How stream processing can transform patient care by enabling real-time health monitoring, particularly for chronic diseases such as heart failure or diabetes.

- Source: [arxiv.org]() and [confluent.io]()

Batch Vs Stream

Aspect	Stream Processing	Batch Processing
Latency	Milliseconds to seconds	Minutes to hours
Data Nature	Unbounded, continuous	Bounded, discrete
Processing	Incremental (micro-batches) or event-by-event	Processed all at once
Use Cases	Real-time monitoring, IoT, fraud detection	Reporting, analytics, historical analysis



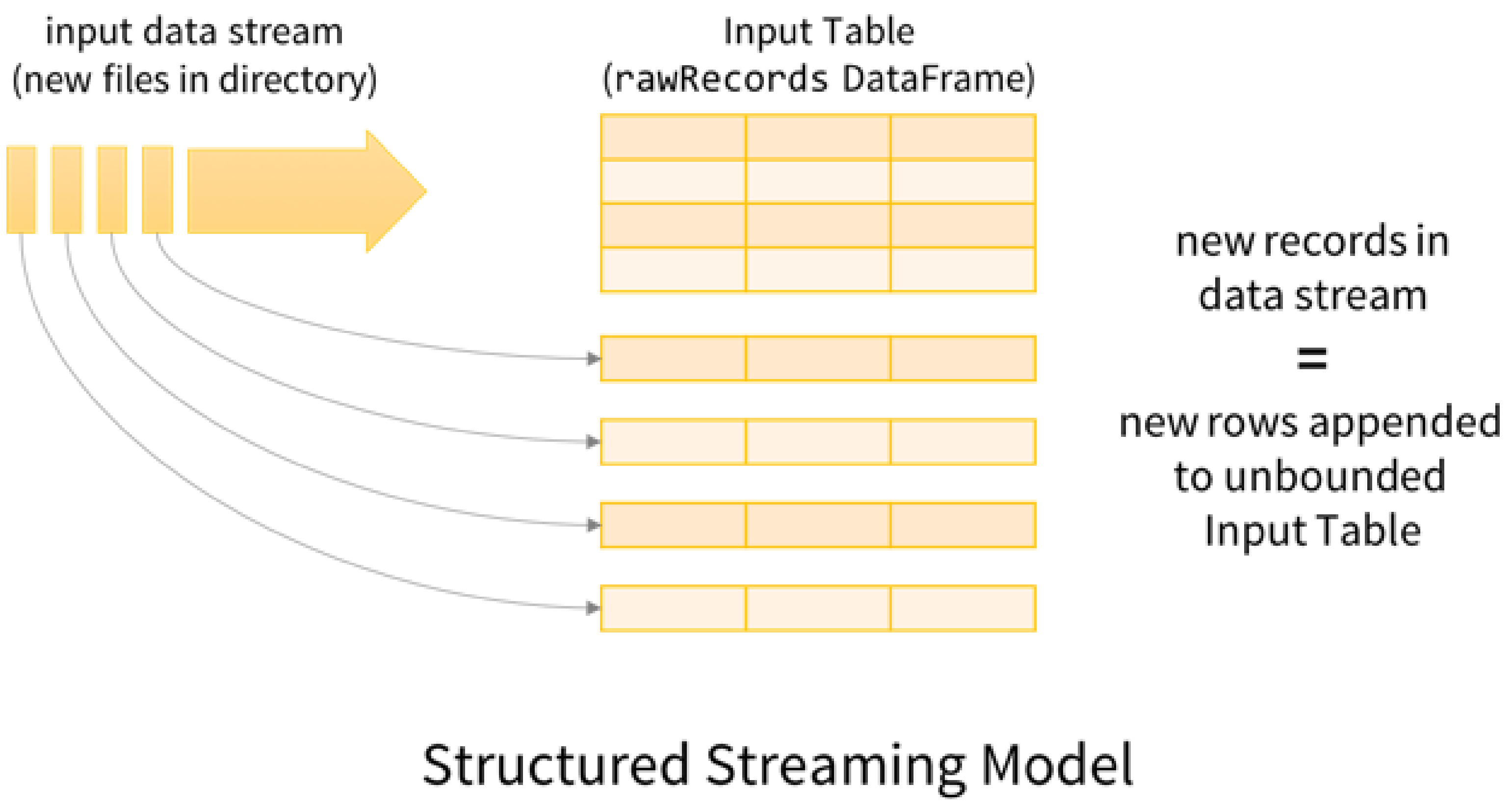
Apache Spark Structured Streaming

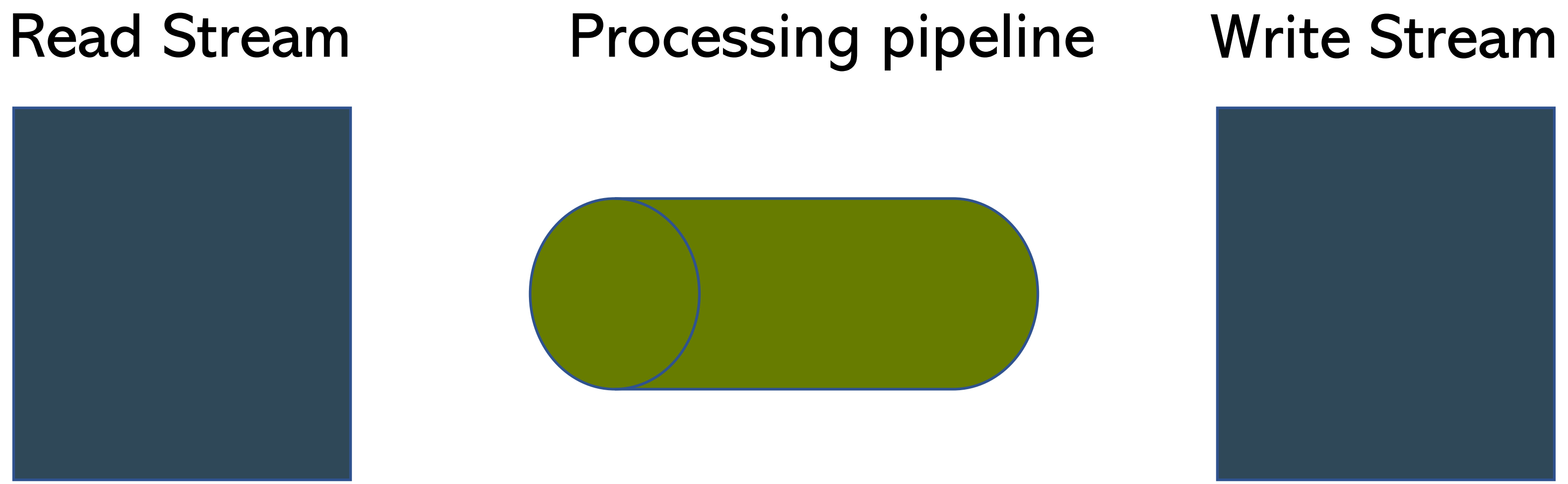
Apache Spark Streaming

SCHOOL OF SCIENCE, ENGINEERING & ENVIRONMENT

Structured Streaming

Every data item that is arriving on the stream is like a new row being appended to the input table.




Read Stream
Processing pipeline
Write Stream

The possible input sources

- File source
  - Reads files written in a directory as a stream of data.

File-0.json
File-1.json
File-2.json
File-3.json
File-4.json
File-n.json

Files will be processed in the order of file modification time.

If *latestFirst* is set, order will be reversed

Supported file formats are text, CSV, JSON, ORC, Parquet.

Input Stream (file source)

```
streamingInputDF = (
  spark
    .readStream                          # it can be "read" for defining static processing
    .schema(jsonSchema)                  # Set the schema of the JSON data
    .option("maxFilesPerTrigger", 1)  # Treat a sequence of files as a stream by picking one file at a time
    .json(inputPath)
)
```

Processing pipeline (examples)

You can do a wide variety of processing/transformation over the incoming data stream.

```
# Split the lines into words
words = streamingInputDF.select(
   explode(
       split(streamingInputDF.value, " ")
   ).alias("word") # name of the new field
)

# Generate running word count
wordCounts = words.groupBy("word").count()
```

Processing pipeline (examples)

```
streamingCountsDF = (
 streamingInputDF
    .groupBy(
      streamingInputDF.action,
      window(streamingInputDF.time, "1 hour"))
    .count()
)
```

Processing pipeline (examples)

```
result_df = streamingInputDF \
  .groupBy("name") \
  .agg({"price": "mean"})
```

The possible output modes

1

Complete Mode: The entire updated result table is written to external storage.

2

Append Mode: Only new rows appended in the result table since the last trigger are written to external storage.

3

Update Mode: Only the rows that were updated in the result table since the last trigger are written to external storage.

Output Sinks

File sink

```
writeStream
.format("parquet")        # can be "orc", "json", "csv", etc.
.option("path", "path/to/destination/dir")
.start()
```

Kafka sink

```
writeStream
 .format("kafka")
 .option("kafka.bootstrap.servers", "host1:port1,host2:port2")
.option("topic", "updates")
.start()
```

Output Sinks …

Console sink (for debugging) - Prints the output to the console/stdout every time there is a trigger. Both, Append and Complete output modes, are supported.

```
writeStream
.format("console")
.start()
```

Memory sink (for debugging) - The output is stored in memory as an in-memory table. Both, Append and Complete output modes, are supported.

```
writeStream
.format("memory")
.queryName("tableName")
.start()
```

Output Stream Example

Start running the query that prints the running counts to the console

```
query = wordCounts \
    .writeStream \
    .outputMode("complete") \
    .format("console") \
    .start()

query.awaitTermination() # keeps waiting for the termination of the query.
```

Output Stream Example …

```
query = result_df.writeStream \
  .format("console") \
  .outputMode("complete") \
  .start()
query.awaitTermination()
```

Output Stream Example …

```
query = (
  streamingCountsDF
    .writeStream
    .format("memory")         # memory = store in-memory table
    .queryName("counts")      # counts = name of the in-memory table
    .outputMode("complete")  # complete = all the counts should be in
the table
    .start()
)
```

References

- https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html
- https://docs.databricks.com/structured-streaming/examples.html
- https://www.databricks.com/spark/getting-started-with-apache-spark/streaming
- https://docs.databricks.com/getting-started/streaming.html#notebook-stream
- https://www.splunk.com/en_us/blog/learn/stream-processing.html
- https://arxiv.org/pdf/1707.04364
- https://www.confluent.io/en-gb/blog/single-patient-view/

Workshop

- Defining an appropriate schema for parsing incoming messages.
- Implementing input data stream.
- Designing a pipeline to process data.
- Creating a right stream writer