Big Data Hadoop and Spark Developer

Big Data and the Need for Spark

Learning Objectives

By the end of this lesson, you will be able to:

- Work with Big data
- Differentiate between batch and real-time processing
- Work with Spark applications
- Identify the limitations of MapReduce
- Learn the advantages of Spark
- Describe Apache Storm and its limitations

Types of Big Data

Types of Big Data

Big data refers to a collection of large and complicated datasets which has the following types:

Unstructured data

Example: Images and videos data

Structured data

Example: RDBMS data

Quasi-structured data

Example: Web clickstream data

Semi-structured data

Example: XML data with schema

Big Data: Examples

Some examples of big data are given below:

Examples of big data

Twitter

Users send 400,000 tweets per minute.

Facebook

Users like five million posts every minute.

Instagram

Users like two million posts every minute.

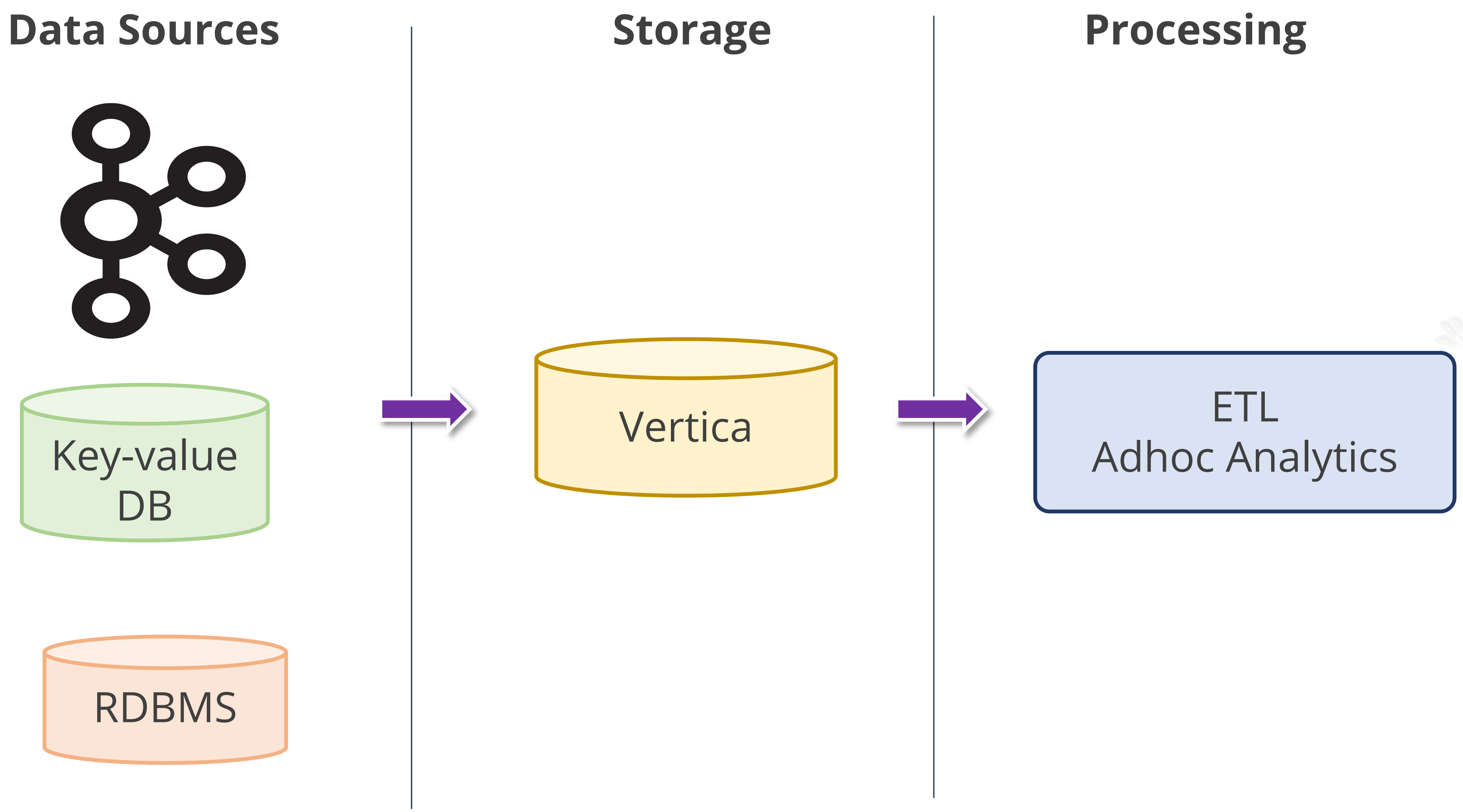
YouTube

Users upload 300hrs of new videos on an average per day.

Challenges in Traditional Data Solution

Uber Old Infrastructure

The following diagram illustrates the traditional infrastructure of Uber.



Data Sources

Key-value DB

RDBMS

Storage

Vertica

Processing

ETL
Adhoc Analytics

Uber Old Infrastructure

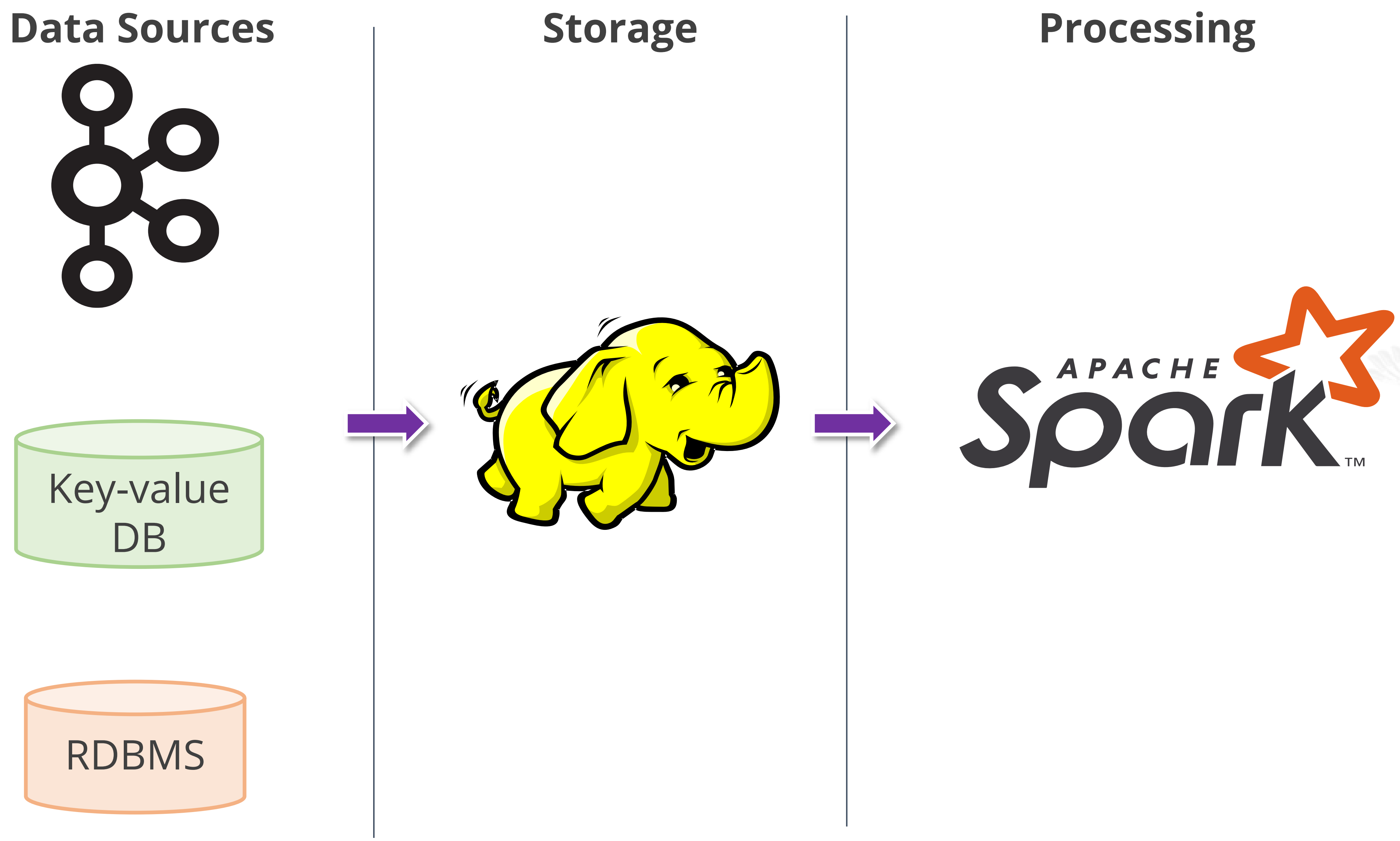
- The old Uber infrastructure consisted of an analytical data warehouse with the goal of centralizing all of Uber's data and facilitating data access.
- Uber chose Vertica as its data warehouse because of its fast, reliable, and column-oriented design.
- Multiple ETL (Extract, Transform, Load) jobs were created to copy data from multiple resources to Vertica.
- An online query service was used to accept user queries and submit them to the underlying query engine.

Uber Old Infrastructure: Problems

- The large number of files in the HDFS began to put additional strain on the NameNodes.
- The data was only accessible to users once every 24 hours, which was too slow to make real-time decisions.
- With data stores growing, these jobs could take over twenty hours with over 1,000 Spark executors to run.
- Hadoop analytics architecture was hitting scalability limitations and many services were affected by high data latency.

Uber New Infrastructure

The new architecture of Uber introduced Apache Spark which helped in the easier processing of data.



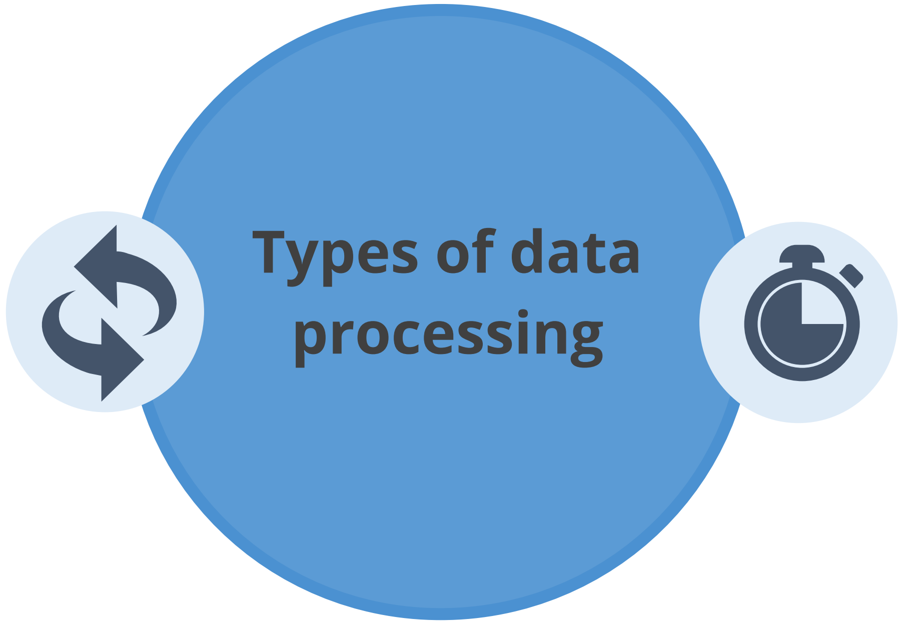
Uber Infrastructure: Solution

- More data warehouse options such as Hive, Presto, and Spark became available, which enabled the users to easily access data.
- Hadoop Upserts and Incremental (Hudi) was introduced, an open-source Spark library.
- Hudi provides an abstraction layer on top of HDFS and Parquet to support the required update and delete operations.
- Hudi enables users to take out just modified data incrementally, boosting query efficiency and enabling incremental changes.

Data Processing in Big Data

Data Processing

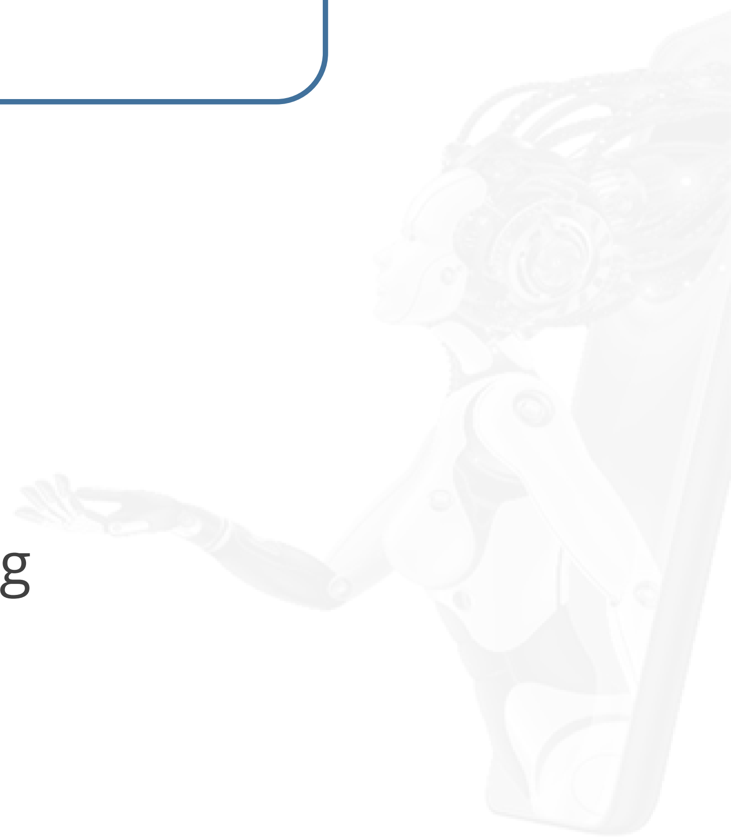
Data processing is a technique of manipulating information. It refers to the transformation of unstructured data into meaningful and readable information.



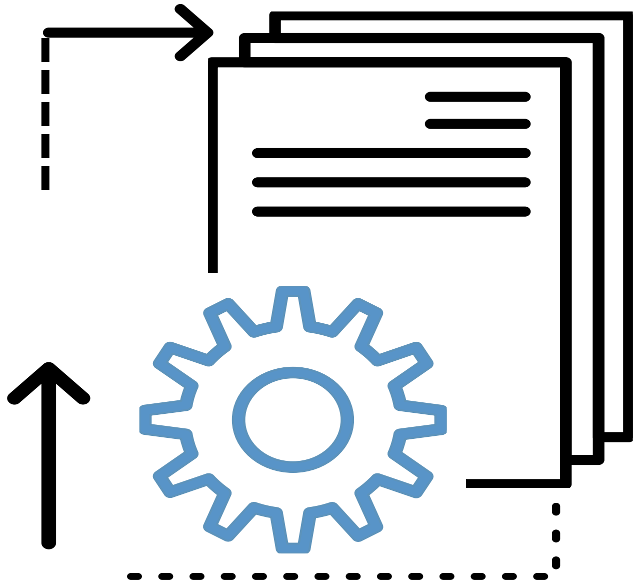
Types of data processing

Batch processing

Real-time processing



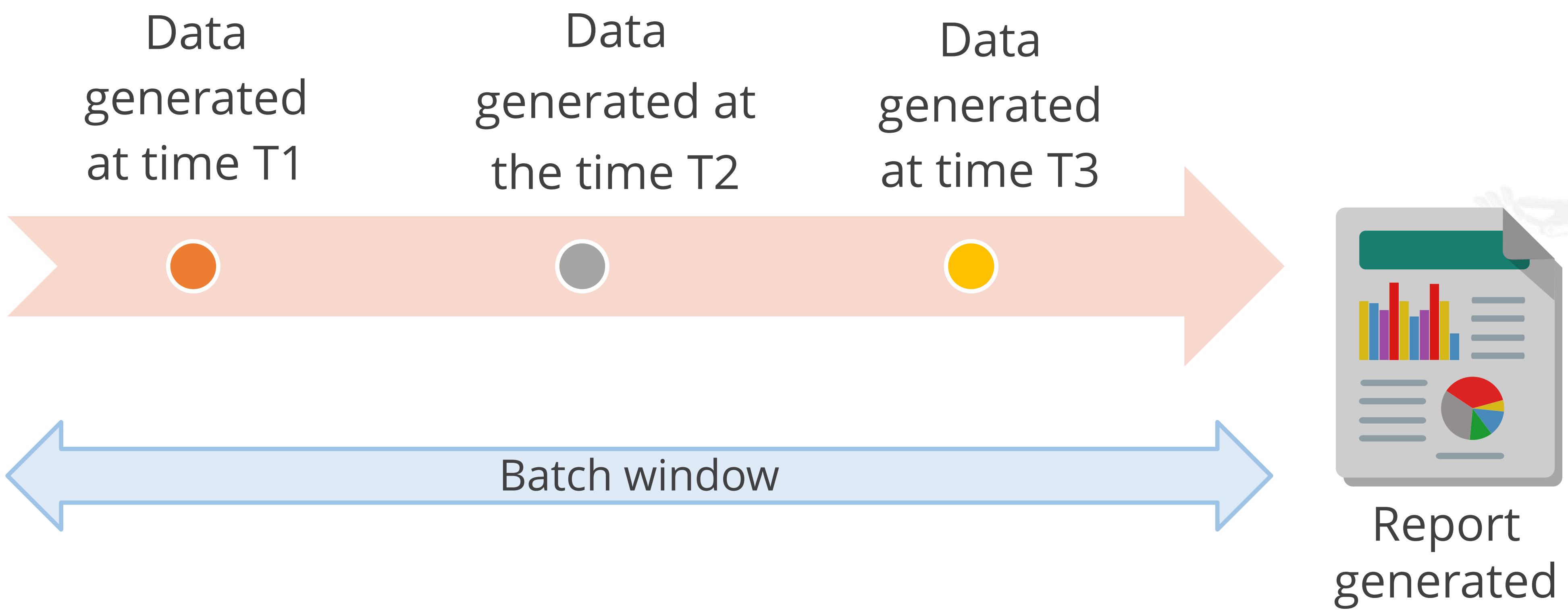
Batch Processing



- Batch processing is a technique where a large volume of data is processed in a single batch.
- The data is collected over a certain period and then processed in every batch window.

Batch Processing: Example

A telecom company generates a report in a batch window every 24 hours.

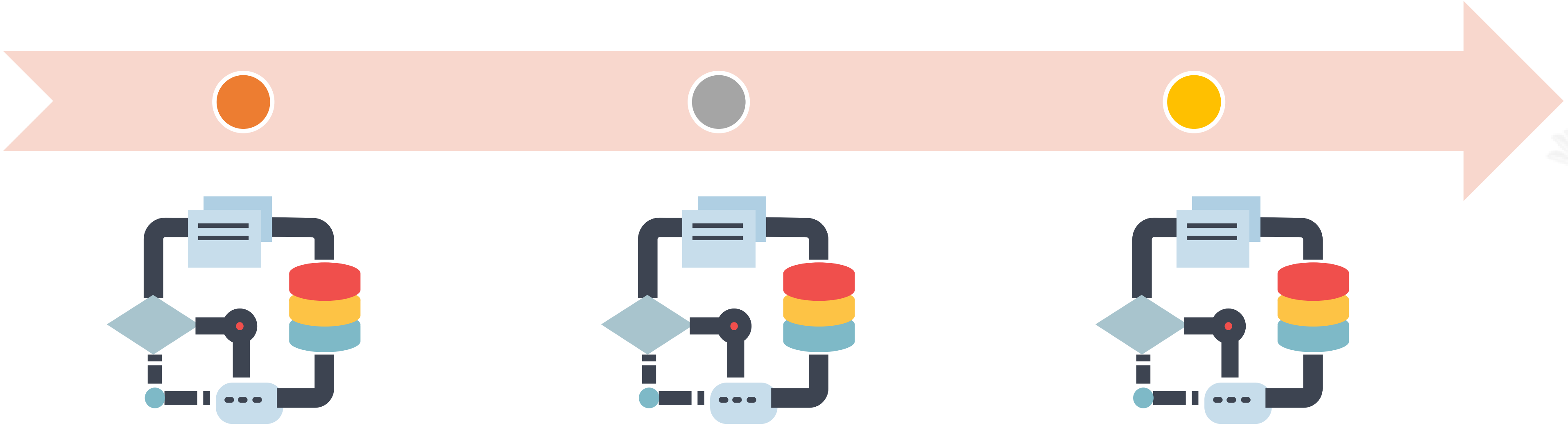

Data generated at time T1
Data generated at the time T2
Data generated at time T3
Batch window
Report generated

Real-Time Processing

- The data is processed as soon as it is generated.
- The data processing time is much smaller.

Real-Time Processing: Example

Banks use real-time processing to detect credit card fraud.



Data generated at time T1

Data generated at time T2

Data generated at time T3

Data processed at T1

Data processed at T2

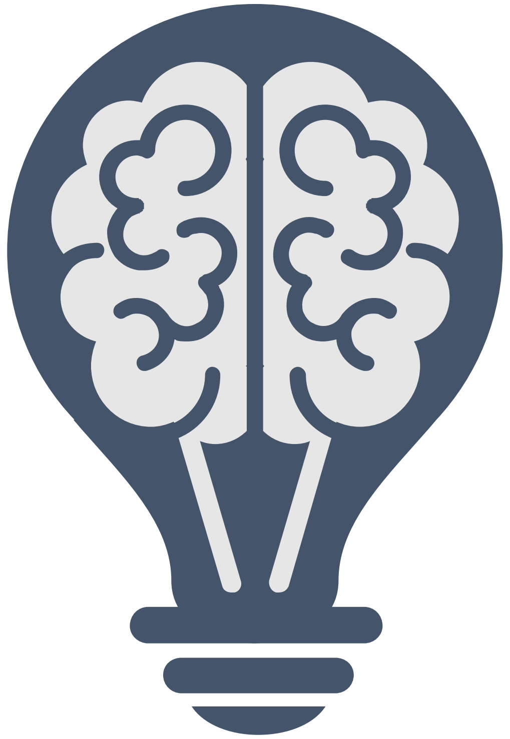
Data processed at T3

Batch vs. Real-Time Processing

Batch processing	Real-time processing
Data processing takes place after an amount of time without any manual intervention.	Data processing takes place instantaneously upon data entry.
A large group of data or transactions is processed in a single run.	It is executed in real-time within stringent constraints.
Batch-time processing is also referred to as traditional processing.	Real-time processing is also referred to as stream processing.
Example: Regular reports that require decision-making	Example: Fraud detection

Distributed Computing and Its Challenges

Distributed Computing



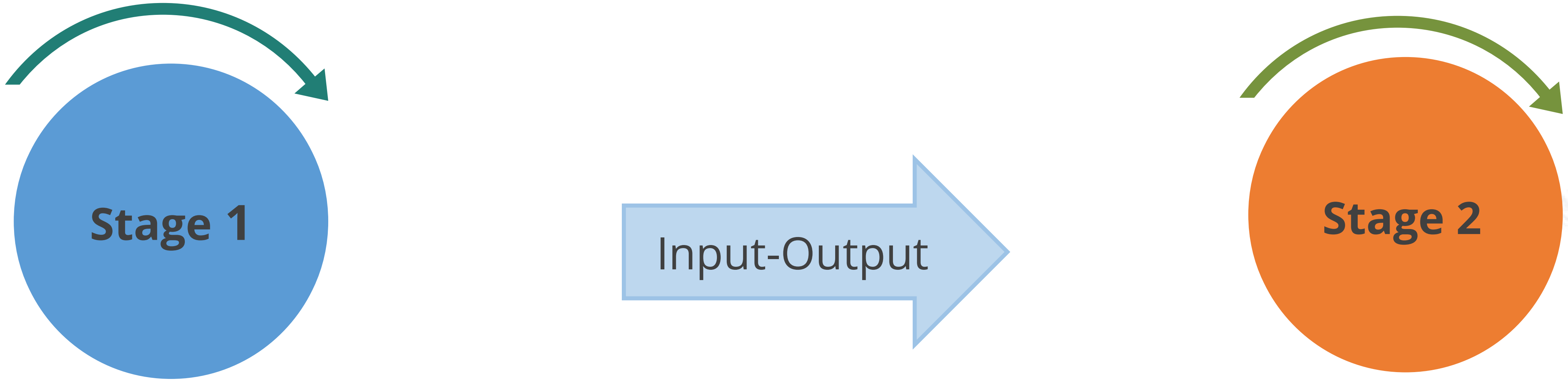
01 Distributed computing is a form of file-based computing.

02 Distributed computing is where the processing of data happens at multiple stages.

03 The output of the first stage is the input to the second stage.

Challenges with Distributed Computing

The intermediate output of each stage is stored in file-based storage. This kind of processing involves a lot of input or output overhead and results in slower computation.

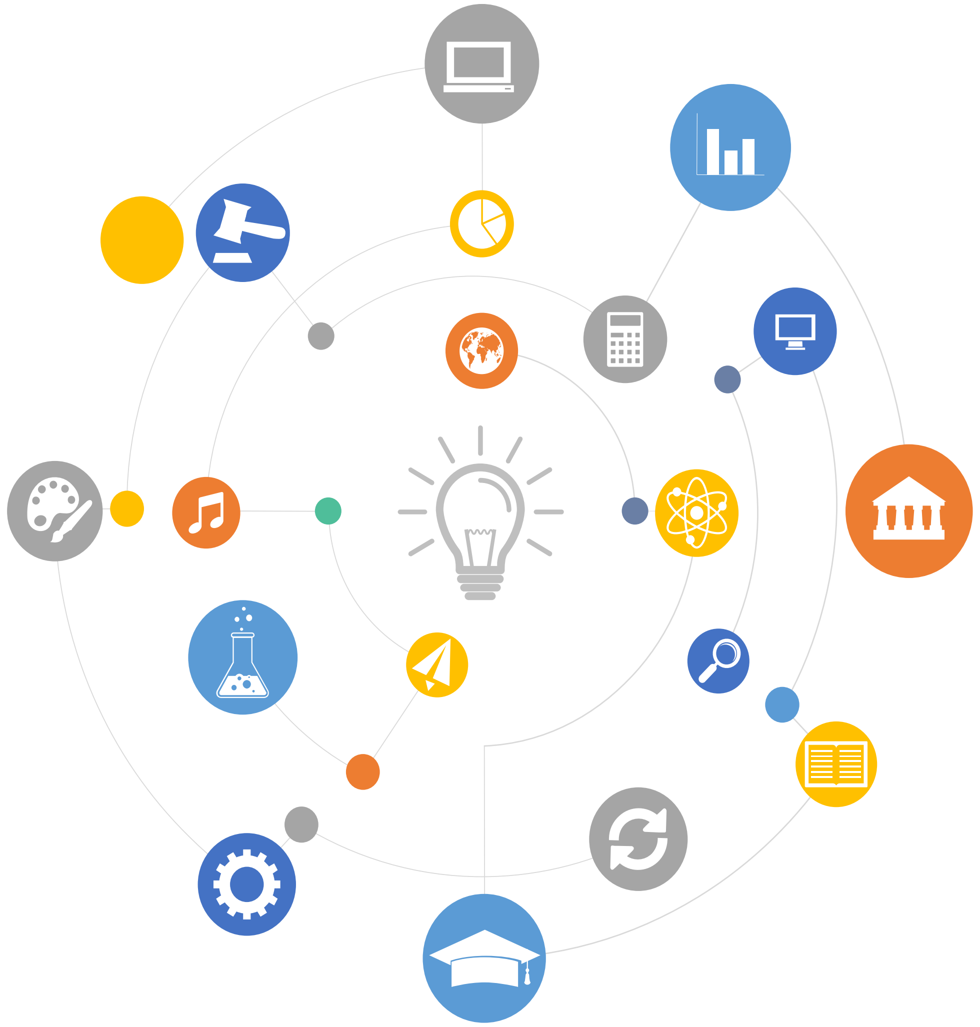


The first stage saves the data in a disk.

The saved data is read in the second stage for further processing.

MapReduce and Its Limitations

MapReduce



- MapReduce is a framework that allows one to create applications that reliably process enormous volumes of data parallelly on vast clusters of commodity hardware.
- A typical MapReduce framework can be used to reduce the input-output overhead.
- The input-output movement of data is time-consuming.

Limitations of MapReduce in Hadoop

Unsuitable for Real-time processing

As it is batch-oriented, it takes minutes to execute jobs depending on the amount of data and the number of nodes in the cluster.

Unsuitable for trivial operations

For operations like filters and joins, one might need to rewrite the jobs, which can be complex due to the key-value pattern.

Unsuitable for large data on the network

As it works on the data locality principle, it cannot process a lot of data that requires shuffling over the network.

Limitations of MapReduce in Hadoop

Unsuitable with OLTP

OLTP requires many short transactions as it works on the batch-oriented framework.

Unsuitable for processing graphs

The Apache graph library processes graphs, which adds additional complexity to the top of MapReduce.

Unsuitable for iterative execution

As it is a stateless execution, MapReduce isn’t useful in cases like k-means that need iterative execution.

Spark over MapReduce

All data processing done through Hadoop MapReduce can also be executed by using Spark.

- **Batch processing:**
  Spark batch can be used instead of Hadoop MapReduce.
- **Structured data analysis:**
  Spark DataFrames are simple and can be used for quick analysis of data.
- **Machine learning analysis:**
  MLlib can be used for clustering, recommendations, and classification.
- **Interactive SQL analysis:**
  Spark SQL can be used over Stringer, Tez, or Impala.
- **Real-time streaming data analysis:**
  Spark Streaming can be used instead of specialized libraries, such as Storm.

Apache Storm and Its Limitations

Apache Storm

- Apache Storm is a free and open-source distributed real-time computation system.

- It is a streaming data framework with high ingestion rates.

- Apache Storm ensures that every message is processed at least once across the topology.

Apache Storm: Limitations

Apache Storm has the following limitations:

There is no framework-level support.

Storm deployment and installation via several tools are challenging.

The Solution Is Apache Spark

Apache Spark can be used to overcome the limitations of Apache Storm

Apache Storm is an excellent solution for real-time stream processing but can be complex for developers hence Apache Spark is preferable.

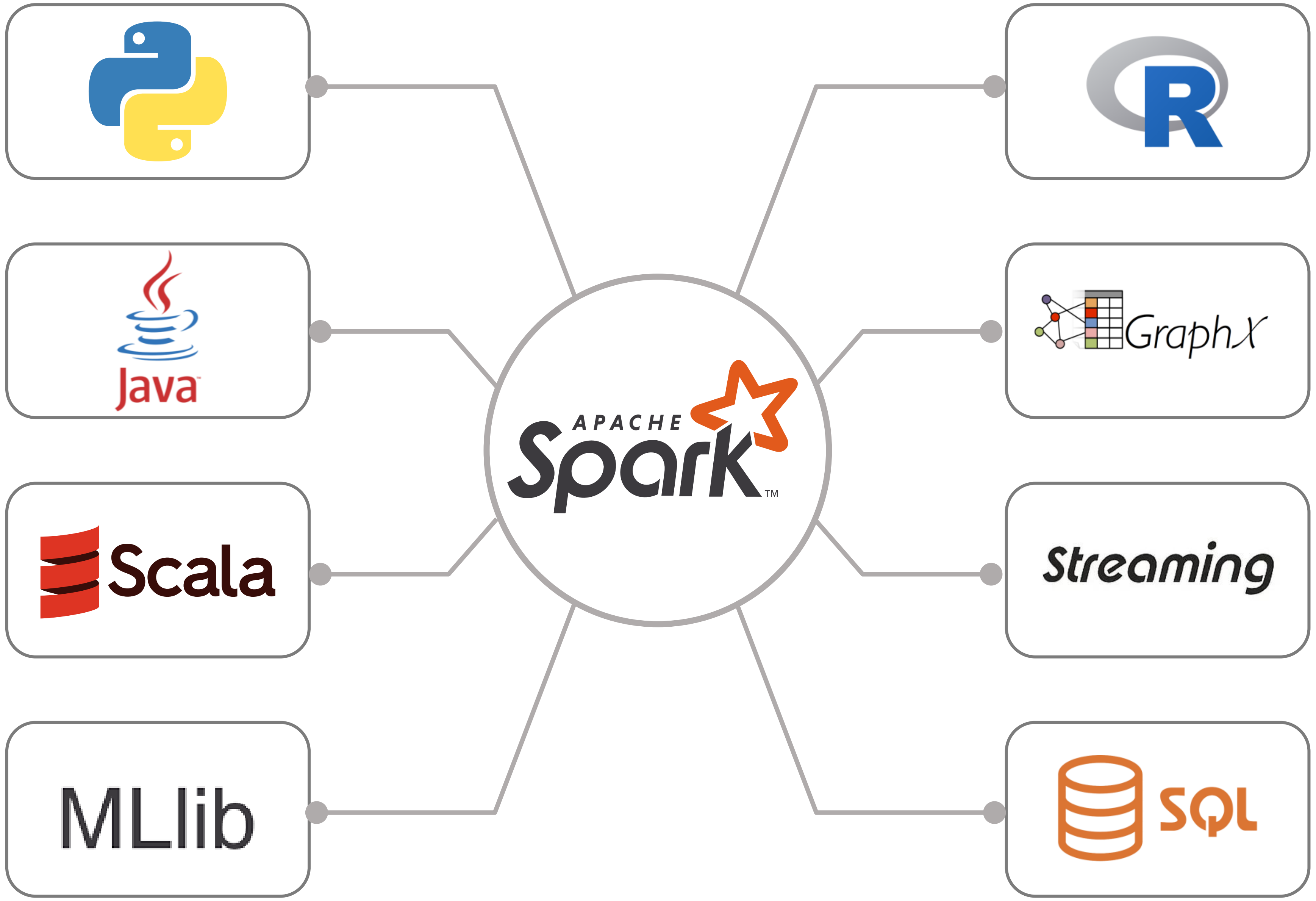
Apache Spark can help with multiple processing problems, such as batch processing, stream processing, and iterative processing.

Apache Spark provides a uniform model Machine learning library that runs on top of the Spark core engine.

General Purpose Solution: Apache Spark

Apache Spark

Apache Spark is an open-source cluster computing framework for real-time data processing.
It contains the following components:



Apache Spark: Features

The features of Apache Spark are given below:

Apache Spark is suitable for real-time trivial operations and to process larger data on a network.

Apache Spark is an open-source cluster computing framework.

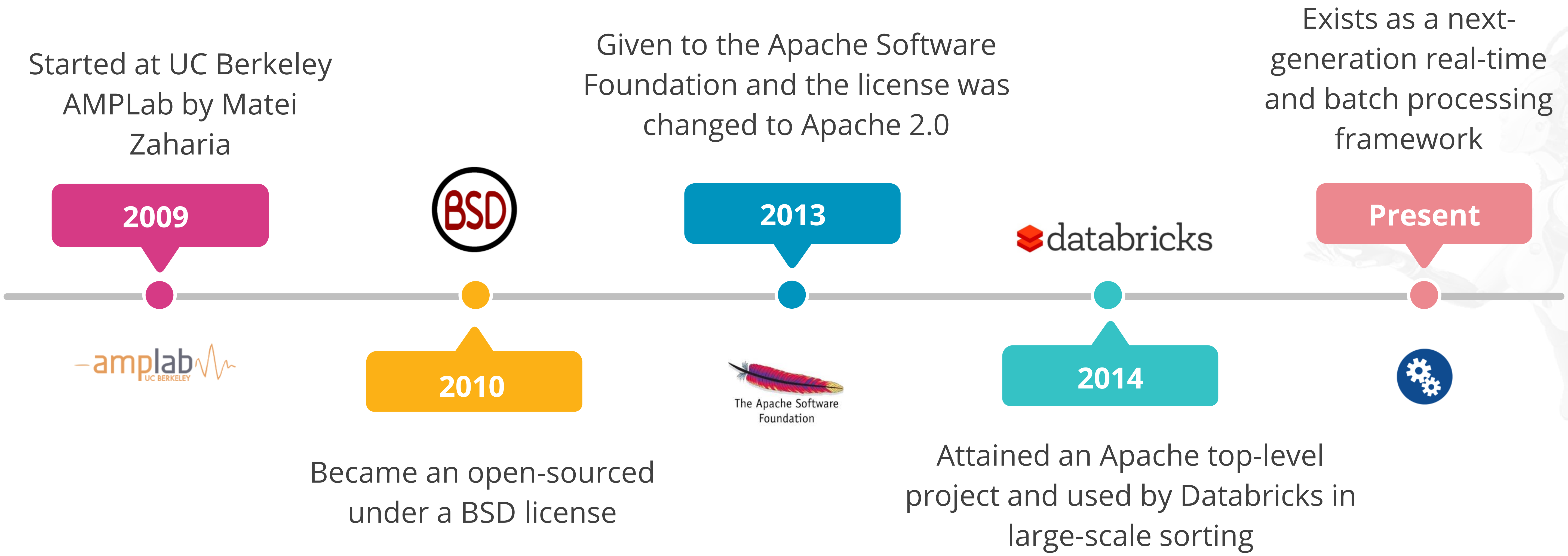
Apache Spark provides up to 100 times faster performance for a few applications.

Apache Spark is suitable for machine learning algorithms as it allows programs to load and query data repeatedly.

open source TM

History of Spark

The history of Apache Spark is explained below:



2009

Started at UC Berkeley AMPLab by Matei Zaharia

2010

Became an open-sourced under a BSD license

2013

Given to the Apache Software Foundation and the license was changed to Apache 2.0

2014

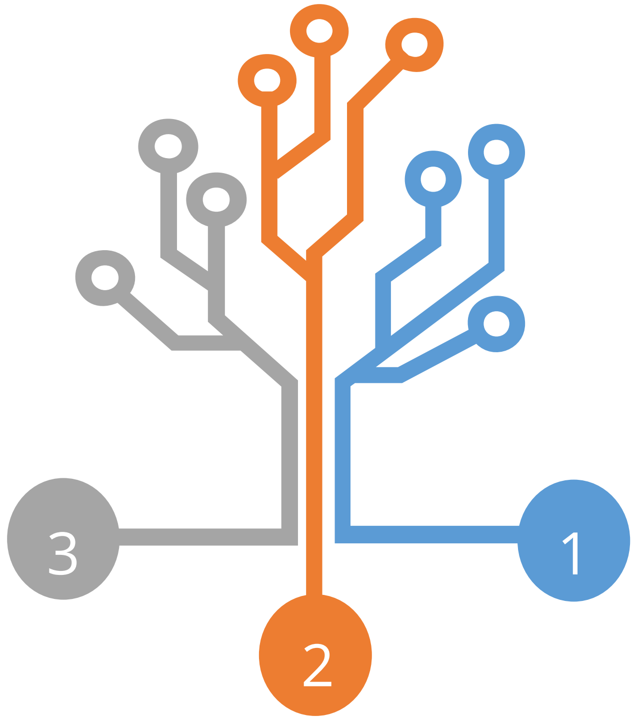
Attained an Apache top-level project and used by Databricks in large-scale sorting

Present

Exists as a next-generation real-time and batch processing framework

Advantages of Spark

Spark provides users with various benefits, such as:



Support for Hadoop:

Spark allows the creation of distributed datasets from files stored in the Hadoop Distributed File System (HDFS).

Speed:

Spark enhances the MapReduce architecture by allowing calculations like stream processing and interactive queries.

Combination:

Spark covers a wide range of workloads that require multiple distributed systems, making it simple to combine different processing types.

Key Takeaways

- Data processing is a technique of manipulating information.
- Data processing refers to the transformation of unstructured data into meaningful and machine-readable information.
- Batch processing is a technique where a large volume of data is processed in a single batch.
- Apache Spark is an open-source cluster computing framework for real-time data processing.

Knowledge Check

Knowledge Check 1

Which type of processing collects the data for a time period and then performs the processing?

A. Real-time processing

B. Batch processing

C. Window processing

D. None of the above

Knowledge Check 1

Which type of processing collects the data for a time period and then performs the processing?

A. Real-time processing

B. Batch processing

C. Window processing

D. None of the above

The correct answer is **B**

In batch processing, data is preselected and most of the time it is in GB, TB, PB, or EB and then jobs are run on top of that.

Knowledge Check 2

Which type of processing processes the data as soon as it is generated?

A. Real-time processing

B. Batch processing

C. Window processing

D. None of the above

Knowledge Check 2

Which type of processing processes the data as soon as it is generated?

A. Real-time processing

B. Batch processing

C. Window processing

D. None of the above

The correct answer is **A**

Real-time processing means that data needs to be processed as soon as it enters the system.

Knowledge Check 3

Apache MapReduce saves the intermediate data between stages in ________.

A. Memory

B. File System

C. Network backed Disk

D. Not required as it processes all data in memory

Knowledge Check 3

Apache MapReduce saves the intermediate data between stages in ________.

A. Memory

B. File System

C. Network backed Disk

D. Not required as it processes all data in memory

The correct answer is **B**

Map writes its intermediate data on temp location, then reduce reads from same and dump the output into /temp directory.

Knowledge Check 4

Spark was started in the year ______.

A. 2009

B. 2010

C. 2013

D. 2014

Knowledge Check 4

Spark was started in the year______.

A. 2009

B. 2010

C. 2013

D. 2014

The correct answer is **A**

Spark was started in the year 2009 at UC Berkeley AMPLab by Matei Zaharia.

Knowledge Check 5

What are the challenges in old big data solutions?

A. Storage

B. Slow processing

C. Data movement

D. All of the above

Knowledge Check 5

What are the challenges in old big data solutions?

A. Storage

B. Slow processing

C. Data movement

D. All of the above

The correct answer is **D**

Whether it is storing huge amounts of datasets without any schema or managing processing on large clusters with data moving across networks was the pain with old big data solutions.

Thank You