Apache Spark Developer

Presenter

/arifmazumder

Mohammed Arif, PhD

Lead Data Scientist

Big Data | Machine Learning | AI

Mohammed Arif has more than eighteen (18) years of working experience in Information Communication and Technology (ICT) industry. The highlights of his career are more than nine (9) years of holding various senior management and/or C-Level and had six (6) years of international ICT consultancy exposure in various countries (APAC and Australia), specially on Big Data, Data Engineering, Machine Learning and AI arena.

He is also Certified Trainer for Microsoft & Cloudera.

Apache Spark 3.0 Developer Materials:

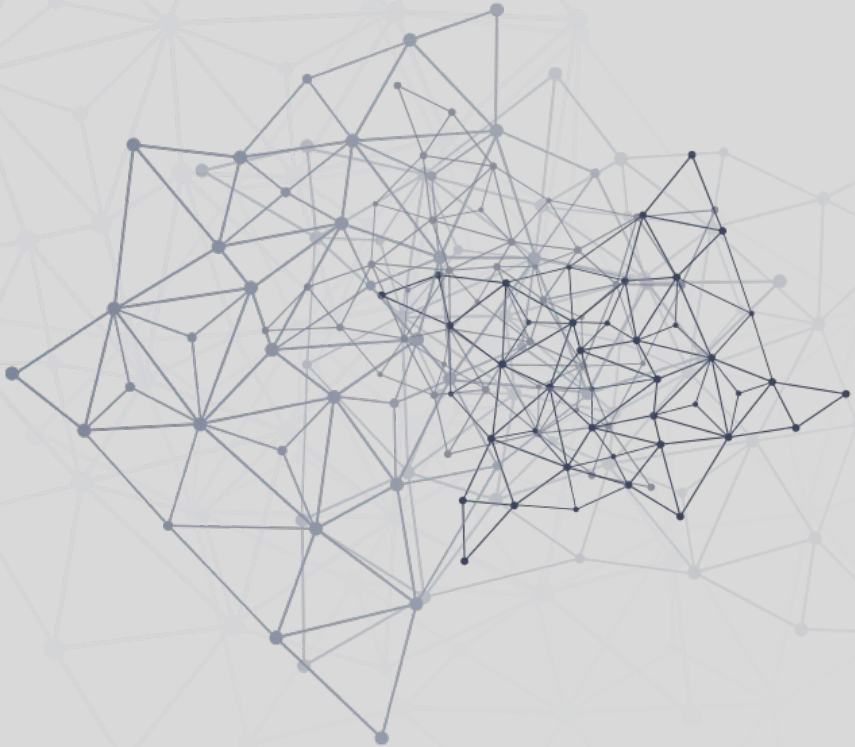
https://arif.works/shd/

Before we begin the setup and coding with Python and Spark, let's discuss what Spark is in the context of Big Data.

We'll begin with a general explanation of what Big Data is and related technologies.

Big Data Overview

- What is “Big Data”?
- Explanation of Hadoop, MapReduce
- Local versus Distributed Systems
- Overview of Hadoop Ecosystem
- Overview of Spark



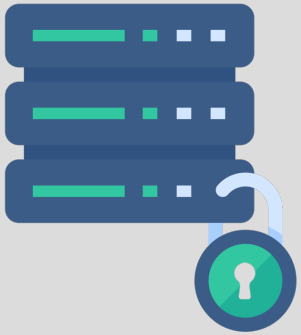
What is Big Data?

The 3 V's of big data

Big data is a collection of data from various sources, often characterized by what's become known as the 3 V's: *volume*, *velocity* and *variety*.

Volume	Velocity	Variety
The amount of data from myriad sources.	The speed at which big data is generated.	The types of data: structured, semistructured, unstructured.

Core challenges of Big Data



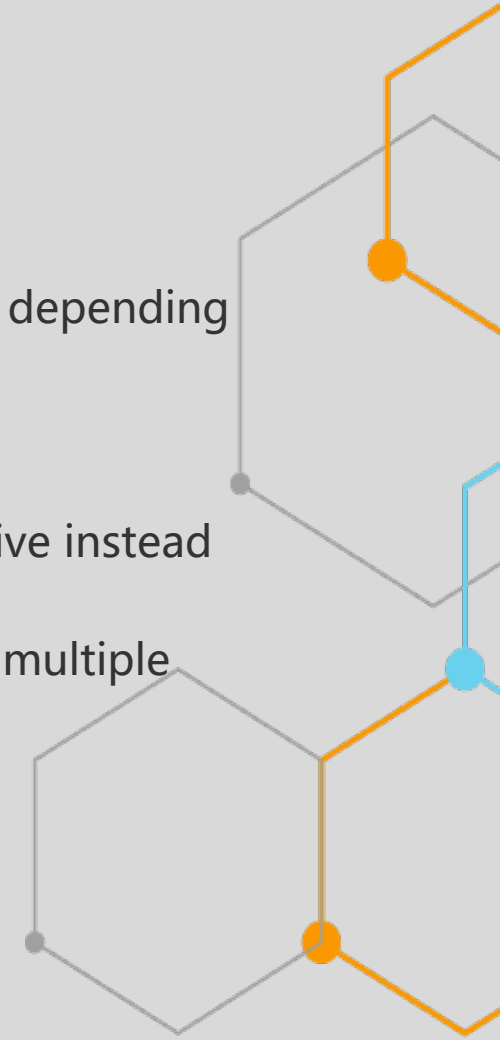
Massive Storage Requirements

Process and doing analysis of this massive data

Big Data Paradigm

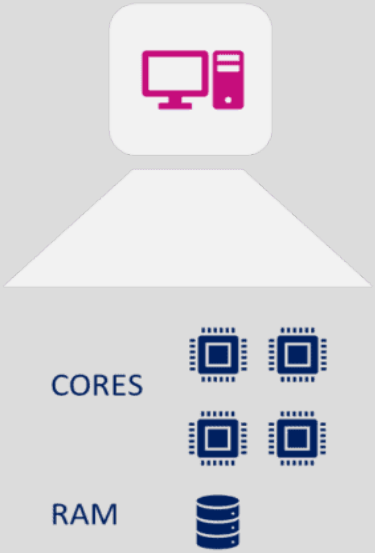
- Data that can fit on a local computer, in the scale of 0-32 GB depending on RAM.

- But what can we do if we have a larger set of data?
  - Try using a SQL database to move storage onto hard drive instead of RAM
  - Or use a distributed system, that distributes the data to multiple machines/computer.



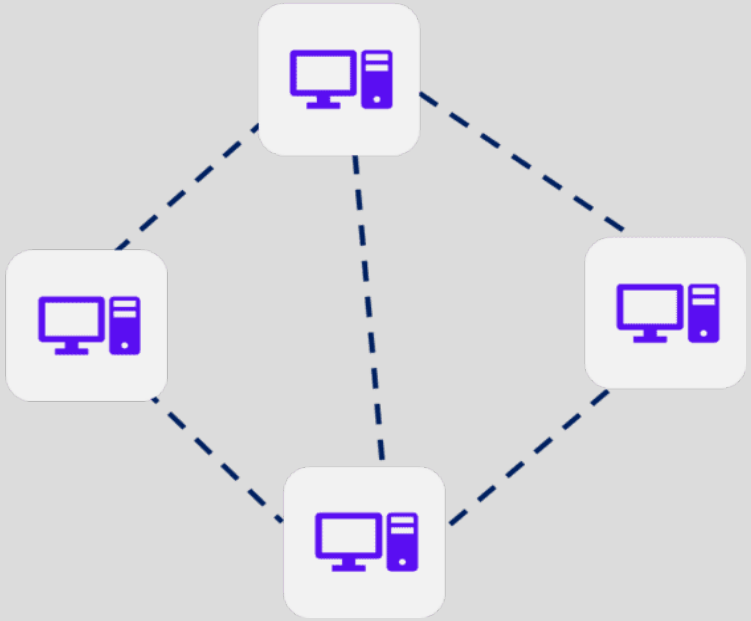
Local versus Distributed System

A SINGLE MACHINE



A local process will use the computation resources of a single machine.

DISTRIBUTED COMPUTING CLUSTER



A distributed process has access to the computational resources across a number of machines connected through a network

What is Apache Hadoop?

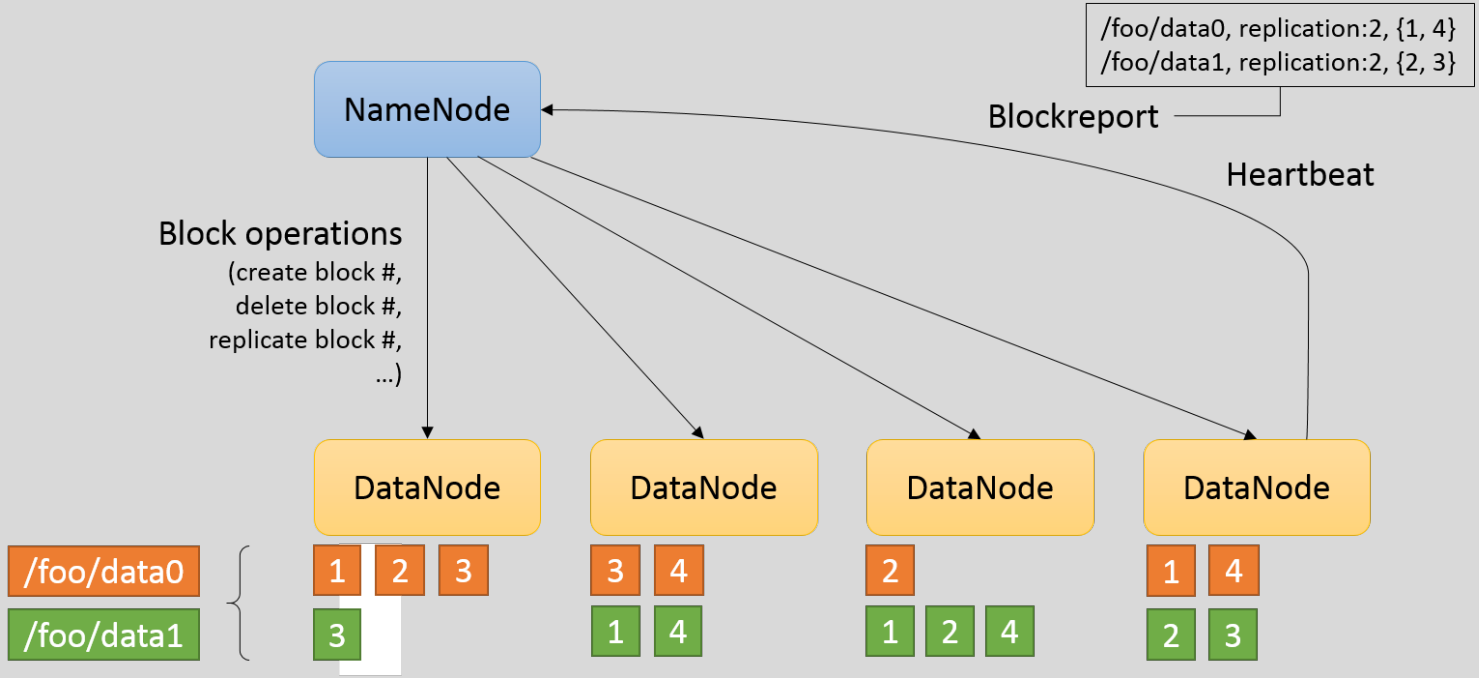
Apache Hadoop software is an open source framework that allows for the distributed storage and processing of large datasets across clusters of computers using simple programming models. Hadoop is designed to scale up from a single computer to thousands of clustered computers, with each machine offering local computation and storage. In this way, Hadoop can efficiently store and process large datasets ranging in size from gigabytes to petabytes of data.

Ref: https://cloud.google.com/learn/what-is-hadoop#section-4

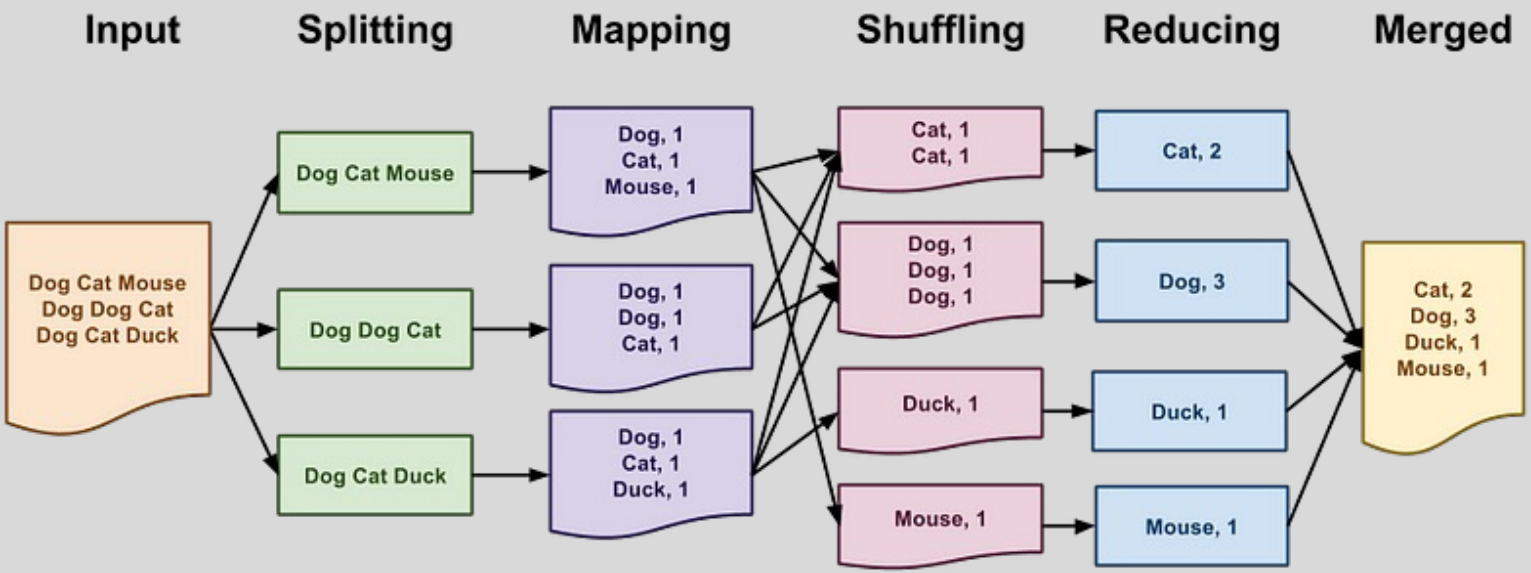
Hadoop Framework

- Hadoop is a way to distribute very large files across multiple machines.
- Hadoop Distributed File System (HDFS) use for store large amount of data
- HDFS allows a user to work with large data sets
- HDFS also duplicates blocks of data for fault tolerance
- It also then uses MapReduce for process massive amount of data

HDFS Architecture

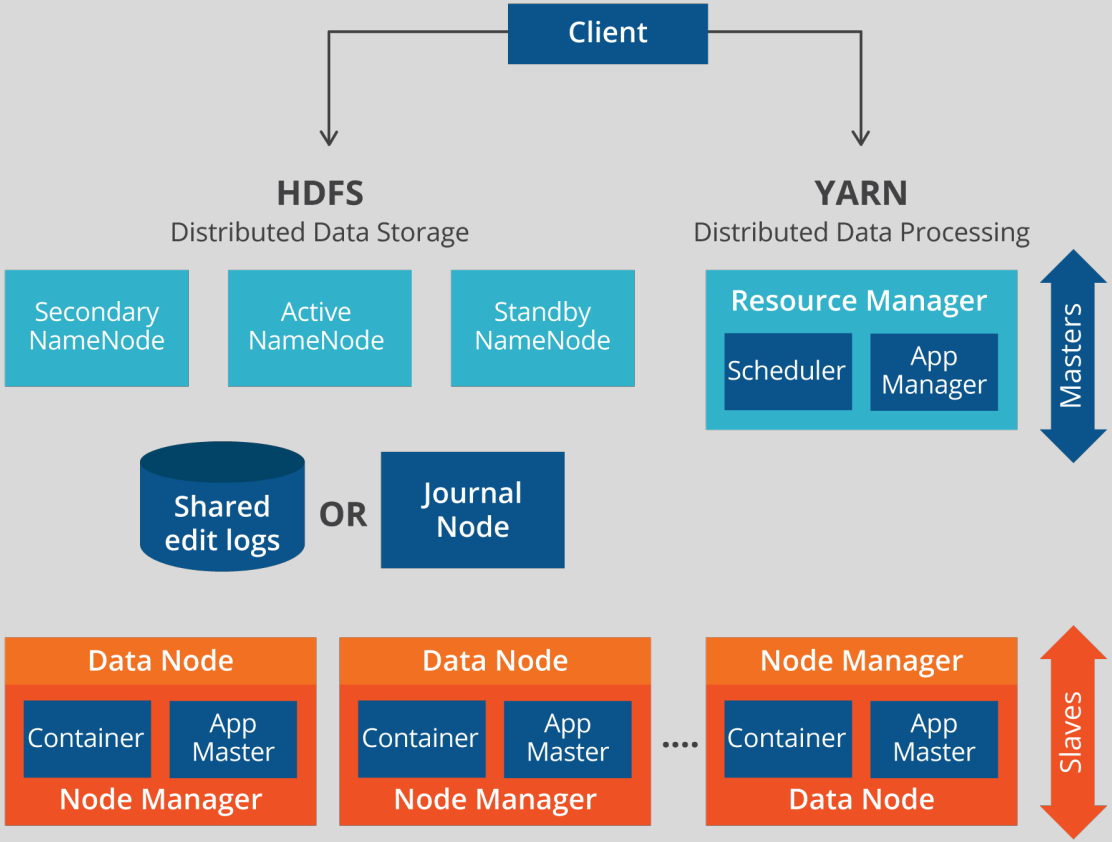


MapReduce


Input
Splitting
Mapping
Shuffling
Reducing
Merged
Dog Cat Mouse
Dog Dog Cat
Dog Cat Duck
Dog Cat Mouse
Dog Dog Cat
Dog Cat Duck
Dog, 1
Cat, 1
Mouse, 1
Dog, 1
Dog, 1
Cat, 1
Dog, 1
Cat, 1
Duck, 1
Cat, 1
Cat, 1
Dog, 1
Dog, 1
Dog, 1
Duck, 1
Mouse, 1
Cat, 2
Dog, 3
Duck, 1
Mouse, 1
Cat, 2
Dog, 3
Duck, 1
Mouse, 1

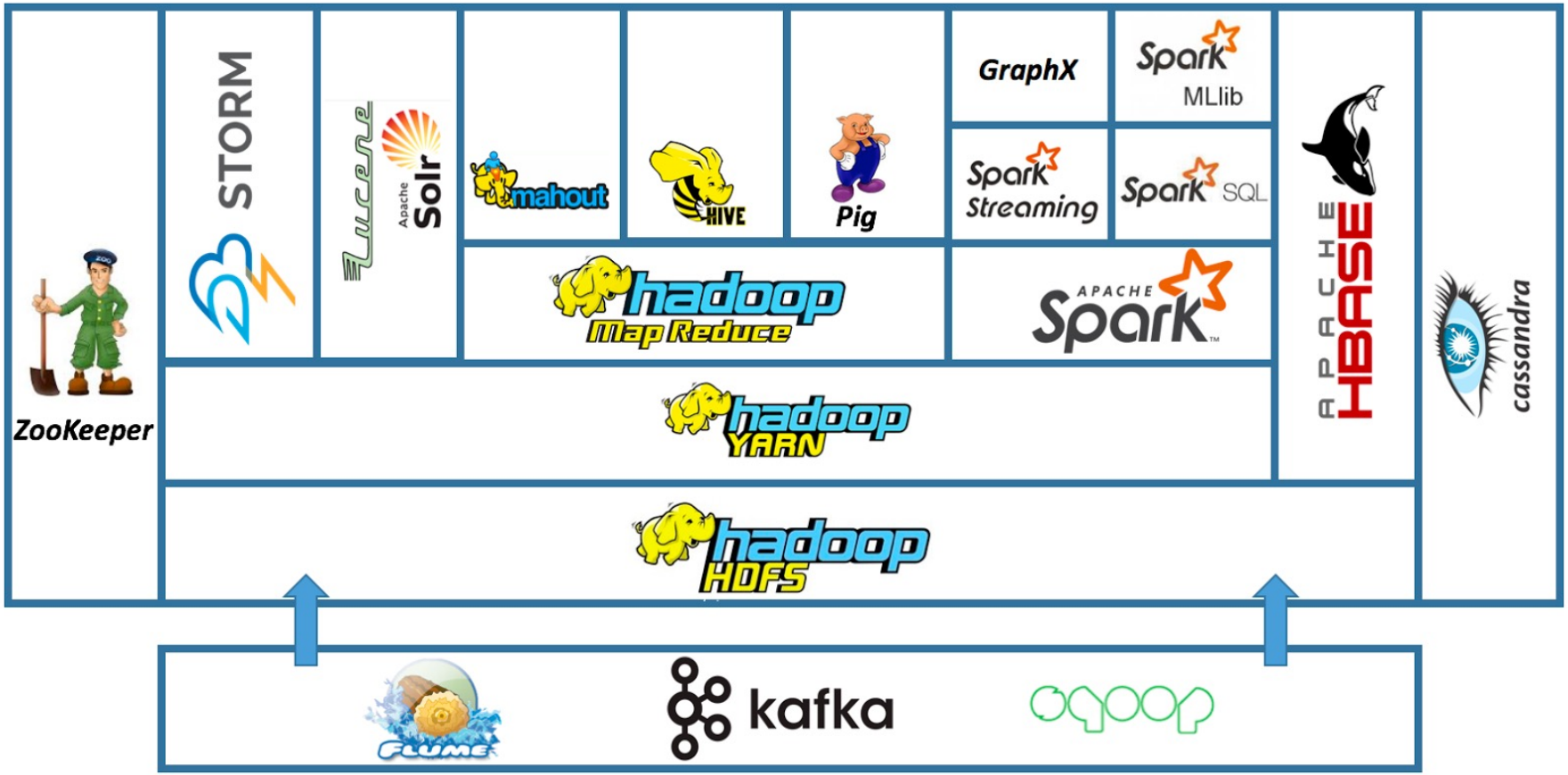

APACHE
hadoop

YARN



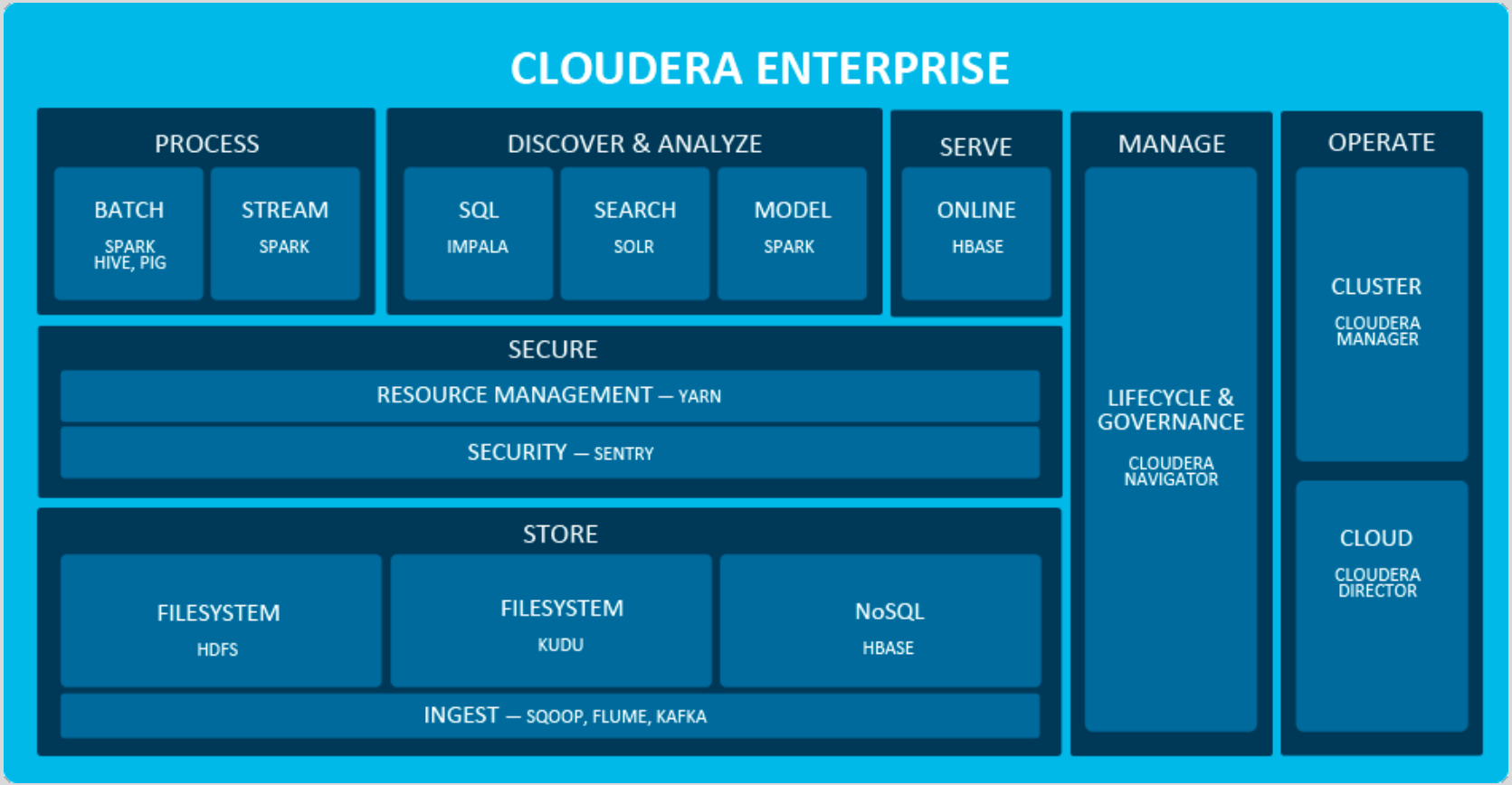
APACHE hadoop

Hadoop Ecosystem



Cloudera Hadoop Ecosystem

CLOUDERA



What we've covered can be thought of in two distinct parts:

- Using HDFS to distribute large data sets
- Using MapReduce to distribute a computational task to a distributed data set

Next we will learn about the latest technology in this space known as Spark. Spark improves on the concepts of using distribution to do any computational tasks.

Spark

In this part of the lecture will be an abstract overview, we will discuss:

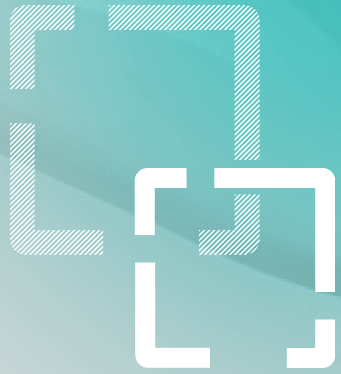
- Spark
- Spark vs MapReduce
- Spark RDDs
- Spark DataFrames

What is Apache Spark?

Apache Spark is an open-source, distributed processing system used for big data workloads. It utilizes in-memory caching, and optimized query execution for fast analytic queries against data of any size. It provides development APIs in Java, Scala, Python and R, and supports code reuse across multiple workloads—batch processing, interactive queries, real-time analytics, machine learning, and graph processing. You'll find it used by organizations from any industry, including at FINRA, Yelp, Zillow, DataXu, Urban Institute, and CrowdStrike.

Ref: https://aws.amazon.com/what-is/apache-spark/

Spark



- You can think of Spark as a flexible alternative to MapReduce
- Spark can use data stored in a variety of formats
  - Cassandra
  - AWS S3
  - HDFS
  - And more

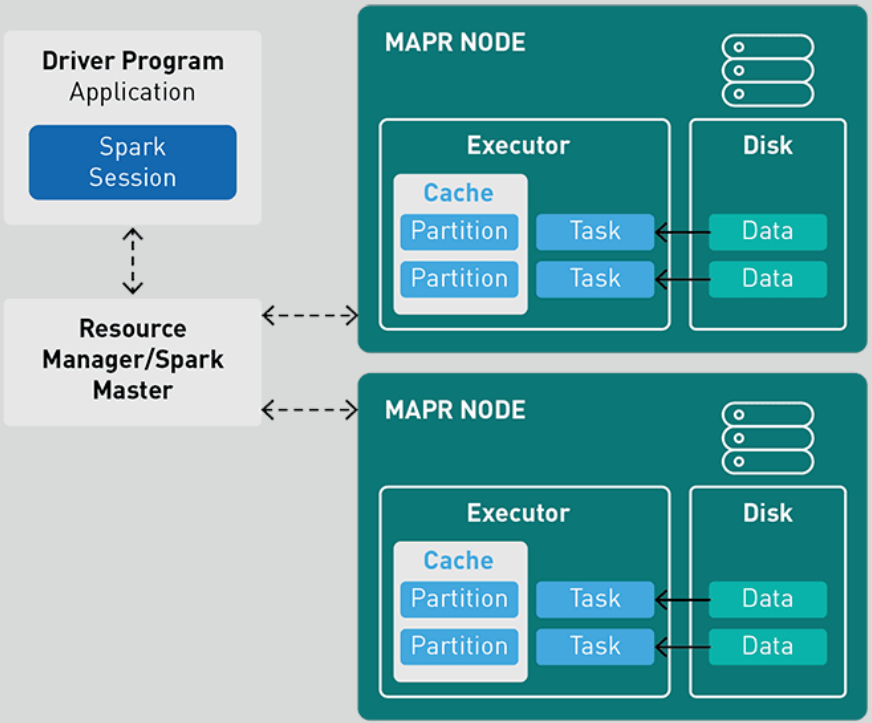
Spark

Spark vs MapReduce

- MapReduce requires files to be stored in HDFS, Spark does not!
- Spark also can perform operations up to 100x faster than MapReduce
- So how does it achieve this speed?
- MapReduce writes most data to disk after each map and reduce operation
- Spark keeps most of the data in memory after each transformation
- Spark can spill over to disk if the memory is filled

Spark

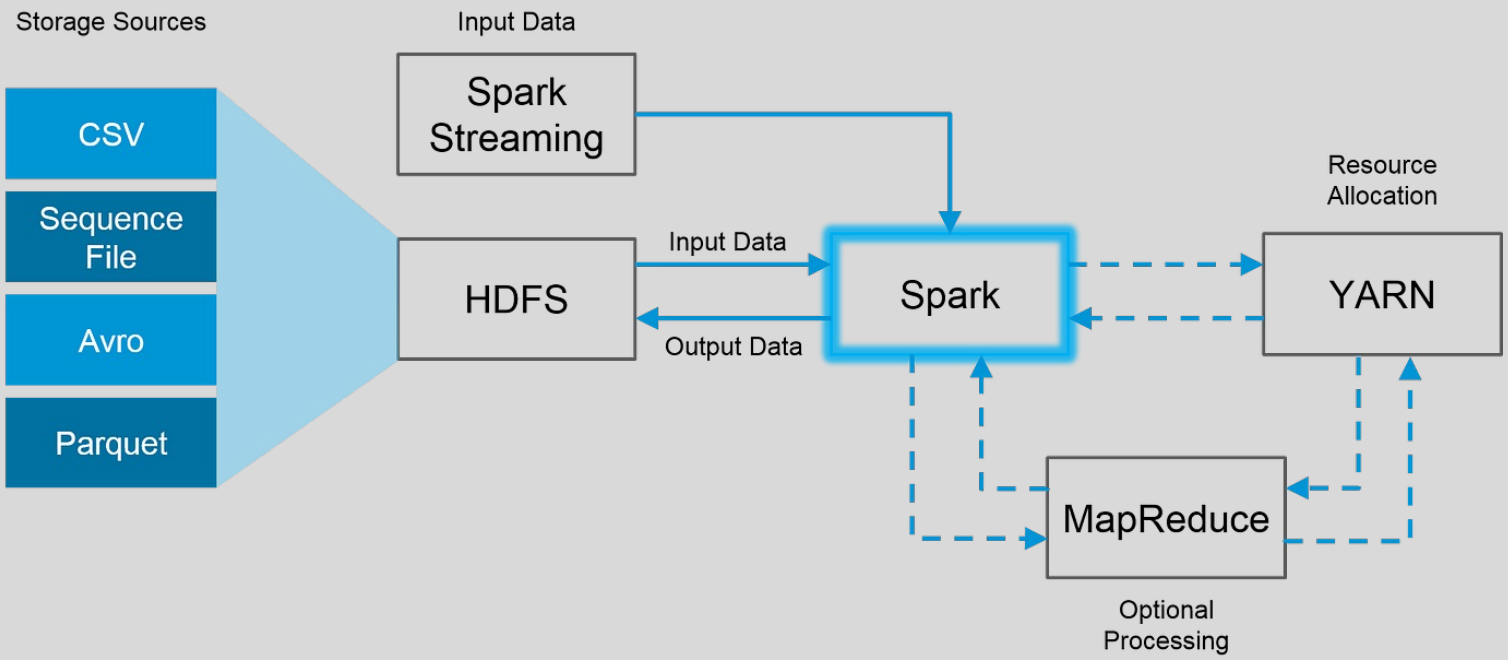
How Spark work in a Cluster



- A spark application runs as independent process, coordinated by SparkSession object in the driver program.
- The resource or cluster manager assign tasks to worker, one task per partition.
- A task applies its unit of work to the dataset in its partition and outputs a new partition dataset. Because iterative algorithms apply operations repeatedly to data, they benefit from caching dataset across iterations.
- Results are sent back to the driver application or can be save to disk.

Spark

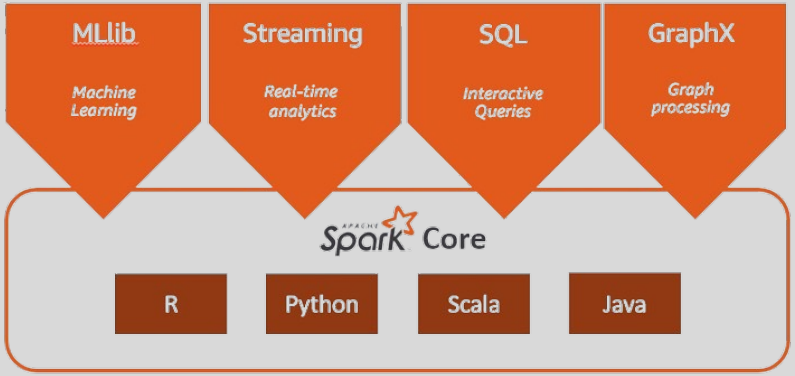
Spark


Storage Sources
CSV
Sequence File
Avro
Parquet
Input Data
Spark Streaming
HDFS
Input Data
Output Data
Spark
Resource Allocation
YARN
MapReduce
Optional Processing

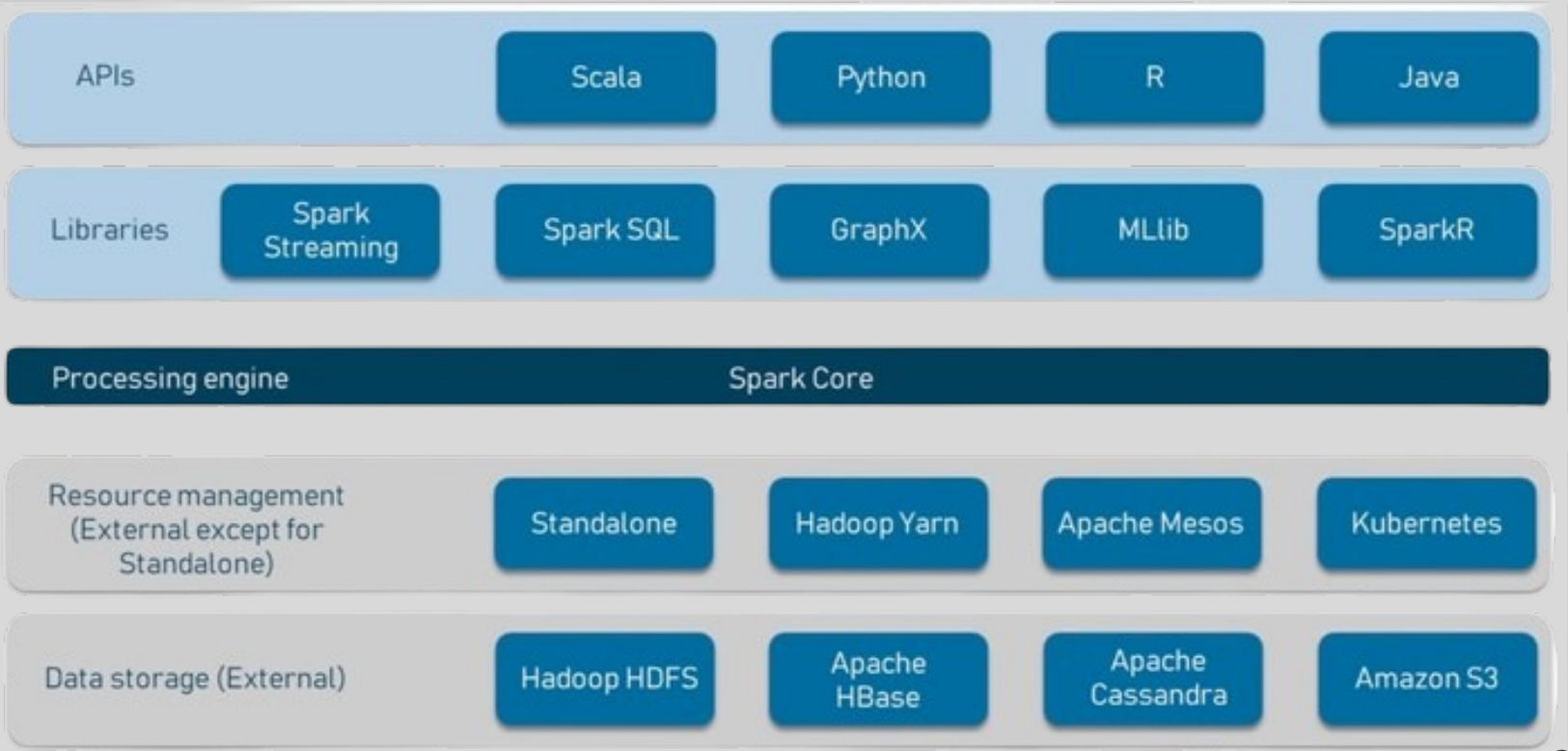
Apache Spark Workloads

The Spark framework includes:

- Spark Core as the foundation for the platform
- Spark SQL for interactive queries
- Spark Streaming for real-time analytics
- Spark MLlib for machine learning
- Spark GraphX for graph processing

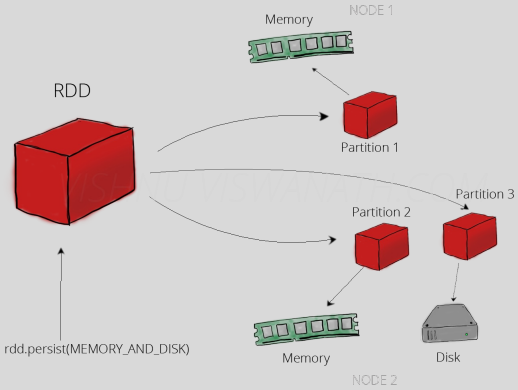


Apache Spark Components



Apache Spark – RDD vs Dataframe

We will discuss the difference in features of Apache Spark RDD vs Dataframe while we are developing Spark program. RDD is the read-only collection of different types of objects, while Dataframe is the distributed collection of a dataset.



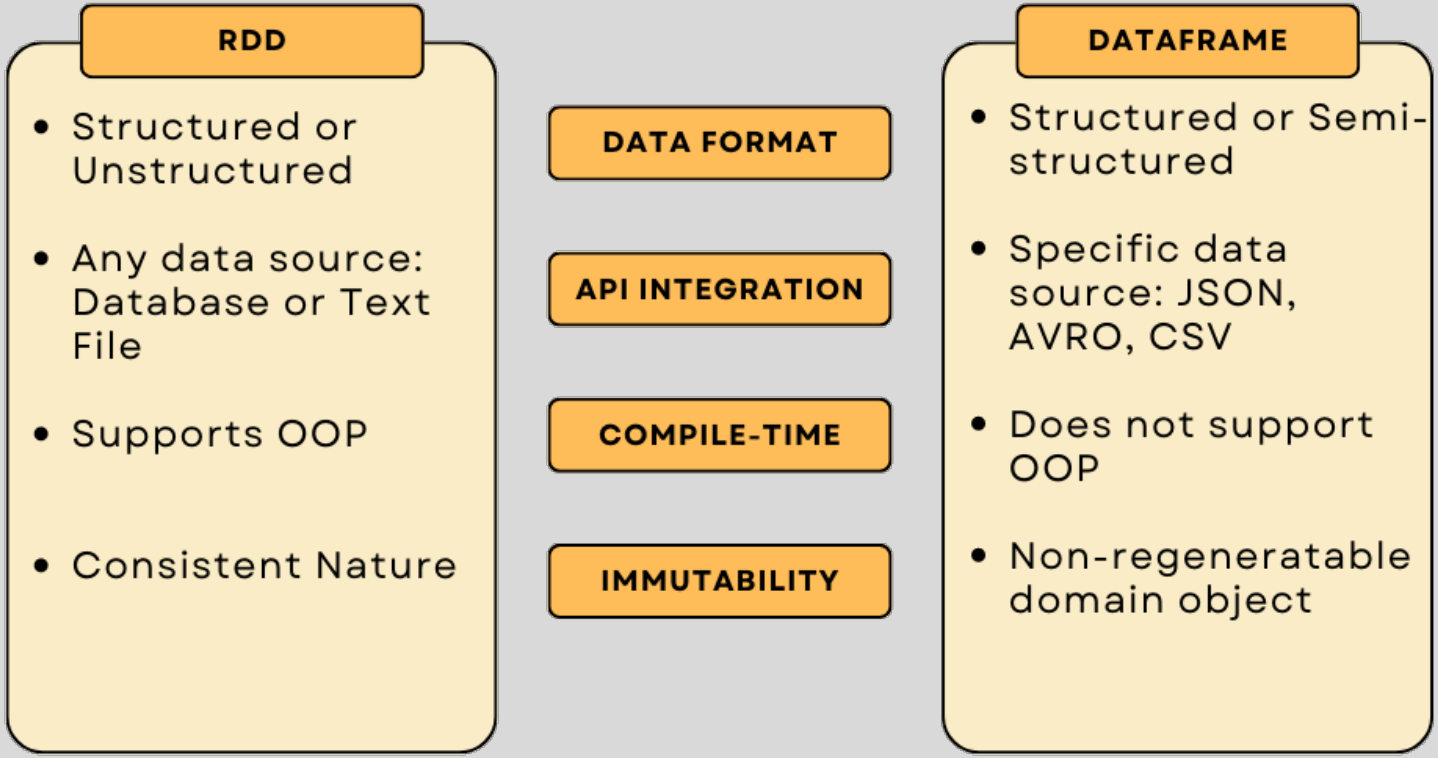
RDD works on the philosophy of "how to do" and
Dataframe and Dataset work on the philosophy of "what to do".

What does this mean?

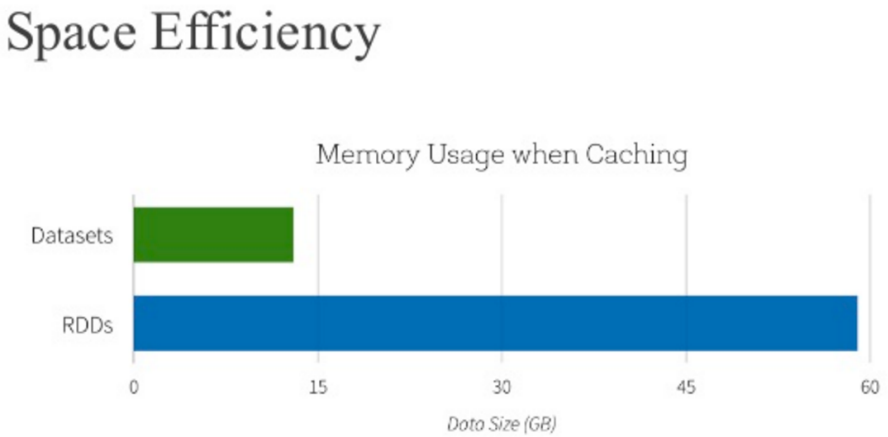
It means that in case of RDD we explicitly tell the program how our data needs to handled and how to process our data in short we tell program each step of execution, where as, in Dataframe and Dataset we just tell the engine what we want and the engine takes care under the hood of finding the most optimized way to execute our requirement.

Spark

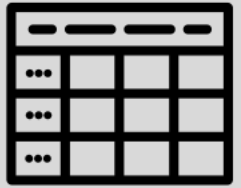
Apache Spark – RDD vs Dataframe



Apache Spark – RDD vs Dataframe



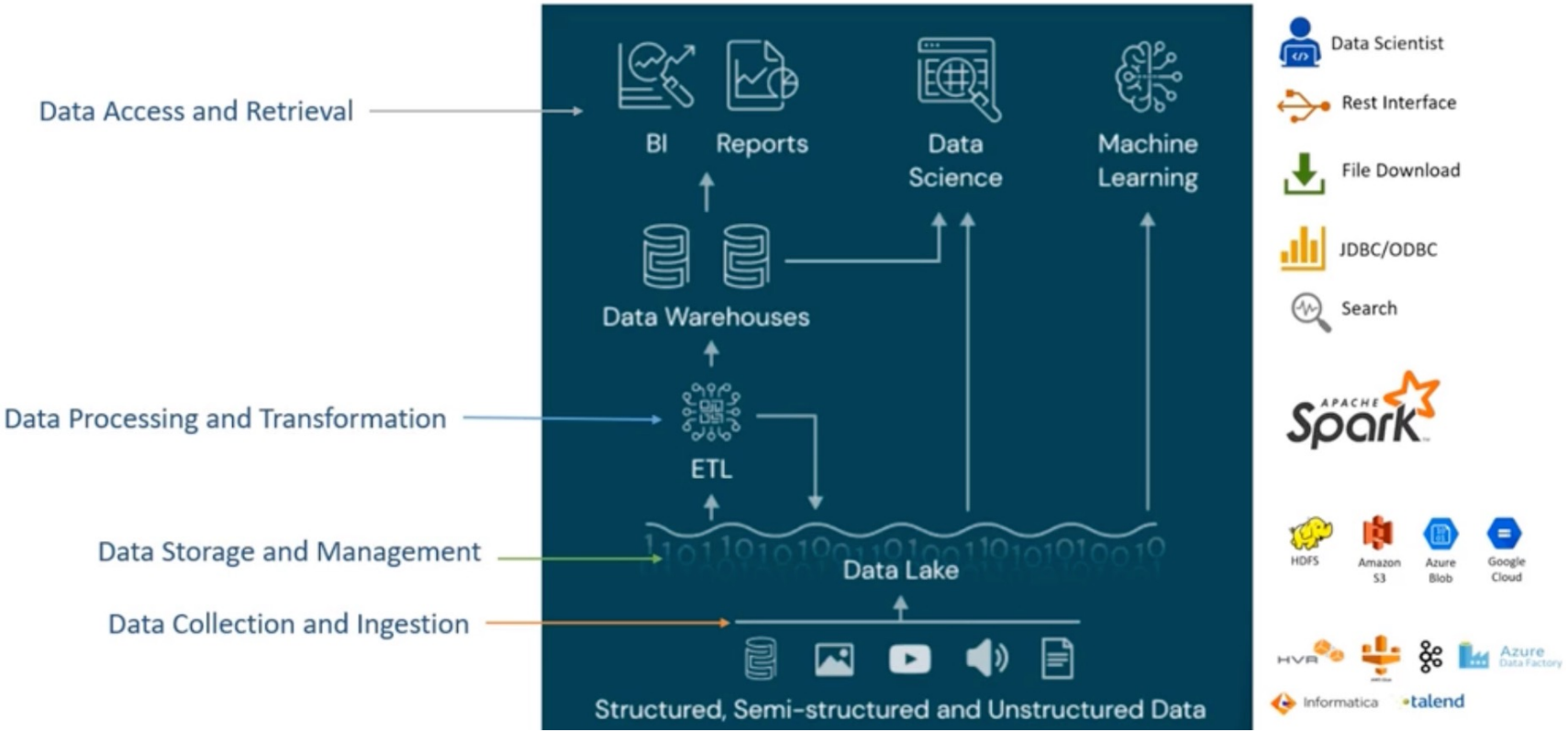
Apache Spark DataFrames



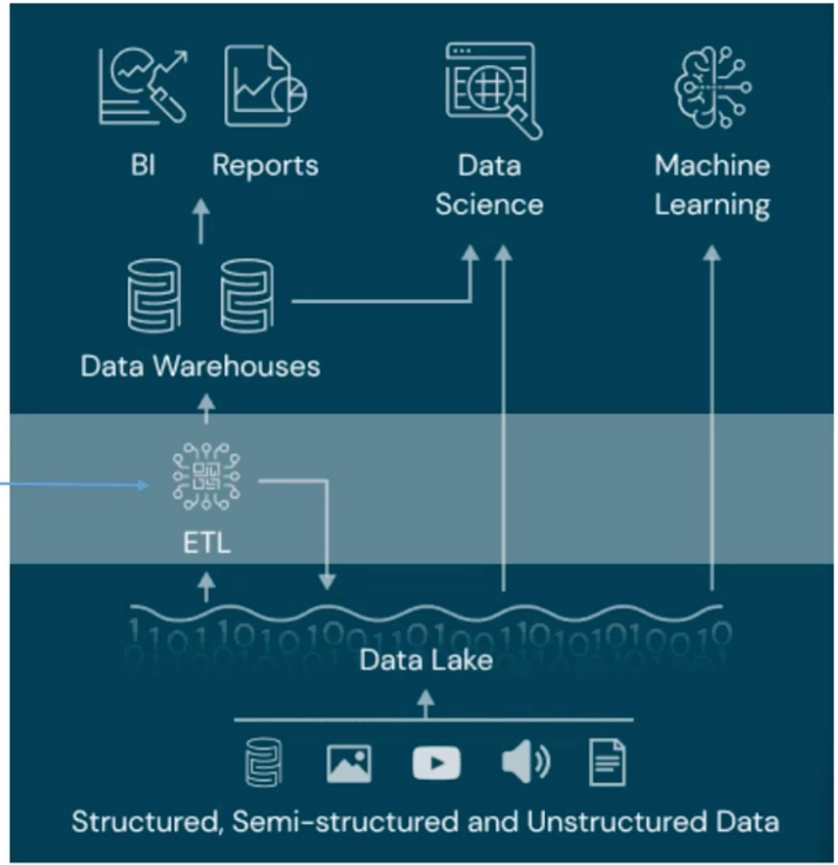
- Spark DataFrames are also now the standard way of using Spark's Machine Learning Capabilities.
- In this course the main way we will be working with Python and Spark is through the DataFrame Syntax.
- Spark DataFrames hold data in a column and row format.
- Each column represents some feature or variable.
- Each row represents an individual data point.
- Spark began with something known as the "RDD" syntax which was a little ugly and tricky to learn.
- Now Spark 2.0 and higher has shifted towards a DataFrame syntax which is much cleaner and easier to work with!

Spark

Data Lake with Apache Spark



Data Lake with Apache Spark



Data Processing and Transformation

1. Initial data quality check.
2. Transforming and preparing data.
3. Correlating, Aggregating, Analysing, and extracting business insights.
4. Applying machine learning model.

APACHE Spark

Databricks

1. Spark on the Cloud Platform
2. Spark Cluster Management
3. Notebooks and Workspace
4. Administration Controls
5. Optimized Spark
6. Databases/Tables and Catalog
7. Databricks SQL Analytics
8. Delta Lake Integration
9. ML Flow
10. Industry vertical accelerators

Spark

Apache Spark

- We've covered a lot!
- Don't worry if you didn't memorize all these details, a lot of this will be covered again as we learn about how to actually code out and utilize these ideas!

Let's get a brief tour of the documentation!

https://spark.apache.org/