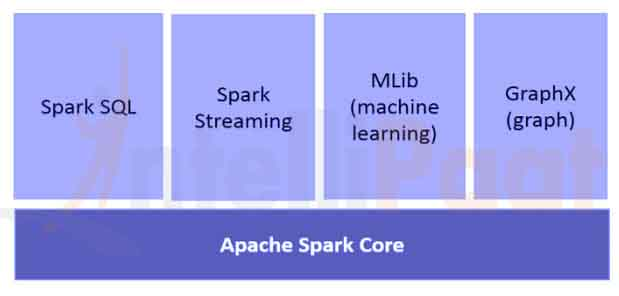
**Day 17 Assignment - Vamsi Viswanadham**

**Date: 22/12/2023**

Apache Spark is an open-source, distributed computing system that provides an interface for programming entire clusters with implicit data parallelism and fault tolerance. It was originally developed at UC Berkeley in 2009, and since then, it has become one of the key big data processing frameworks in the industry.

Here's an overview of its components and architecture:

**Components of Apache Spark**



**Spark Core:**

* The foundation of the platform, providing basic functionality like task scheduling, memory management, fault recovery, interacting with storage systems, and more.
* Contains the API for resilient distributed datasets (RDDs), which is a fundamental data structure of Spark.

**Spark SQL:**

* Allows users to execute SQL queries on structured data using either SQL or the DataFrame API.
* Supports various data sources and can be used to read and write data in a variety of structured formats.

**Spark Streaming:**

* Enables high-throughput, fault-tolerant stream processing of live data streams.
* Can ingest data in mini-batches and perform RDD transformations on those mini-batches of data.

**MLlib (Machine Learning Library):**

* Provides a variety of machine learning algorithms for classification, regression, clustering, and collaborative filtering, as well as underlying optimization primitives.
* Designed to be scalable and to work with big data.

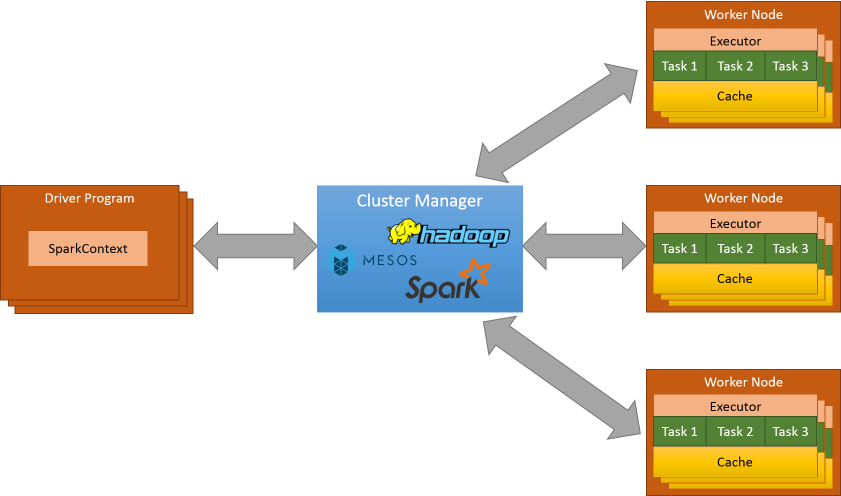
**GraphX:**

* A library for manipulating graphs (like social networks) and performing graph-parallel computations.
* Provides the Pregel API, a scalable message-passing API inspired by the Pregel graph computing model.

**Cluster Managers:**

* Spark is designed to efficiently scale up from one to many thousands of compute nodes.
* To achieve this while maximizing flexibility, Spark can run over a variety of cluster managers, including Hadoop YARN, Apache Mesos, and Kubernetes.

**Architecture of Apache Spark**



**Driver Program:**

The process running the main() function of the application and creating the SparkContext.

Converts a user program into tasks and schedules them on executors.

**SparkContext:**

Connects to the cluster manager (like YARN or Mesos) and negotiates resources.

Responsible for converting an application to tasks and distributing data across the cluster.

**Cluster Manager:**

Responsible for allocating resources to each application and managing executors on nodes in the cluster.

**Executors:**

* Run on the worker nodes of the cluster.
* Responsible for executing the tasks assigned to them by the driver program.
* Each executor runs multiple tasks in separate threads.
* RDDs (Resilient Distributed Datasets):

The fundamental data structure of Spark: They are immutable, distributed collections of objects that can be processed in parallel.

**DAG Scheduler & Task Scheduler:**

The DAG scheduler divides the operator graph into stages, and the Task Scheduler launches tasks via cluster manager.

**Storage Layer:**

Spark can interface with a wide variety of data sources, including HDFS, Cassandra, HBase, S3, etc.

This architecture allows Spark to handle a wide variety of data processing tasks, ranging from batch processing, stream processing, iterative algorithms, and interactive queries, all within the same application.