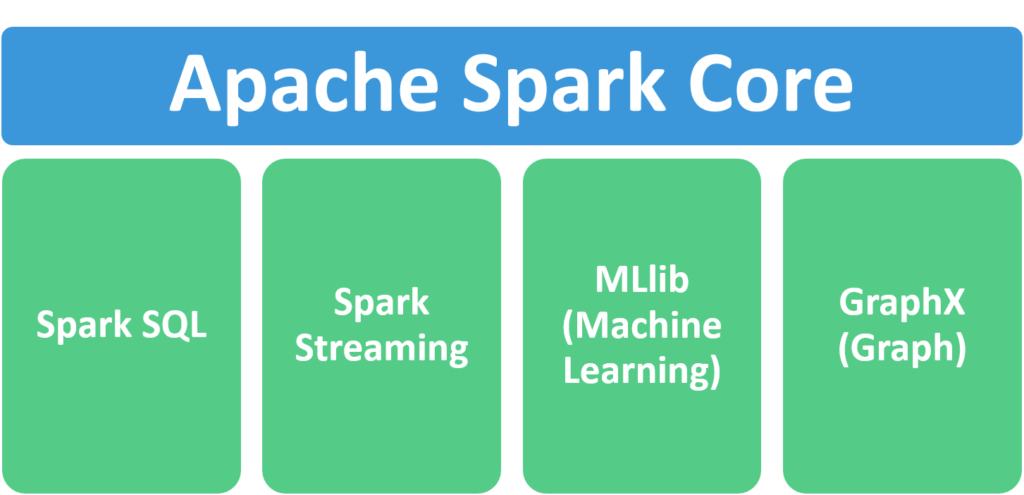
**What is Spark?**

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 queries against data of any size. Simply put, Spark is a **fast and general engine for large-scale data processing**.

The **fast** part means that it’s faster than previous approaches to work with Big Data like classical [MapReduce](https://www.ibm.com/analytics/hadoop/mapreduce). The secret for being faster is that Spark runs on memory (RAM), and that makes the processing much faster than on disk drives.



1. **Apache Spark Core** – Spark Core is the underlying general execution engine for the Spark platform that all other functionality is built upon. It provides in-memory computing and referencing datasets in external storage systems.
2. **Spark SQL** – Spark SQL is Apache Spark’s module for working with structured data. The interfaces offered by Spark SQL provides Spark with more information about the structure of both the data and the computation being performed.
3. **Spark Streaming** – This component allows Spark to process real-time streaming data. Data can be ingested from many sources like Kafka, Flume, and HDFS (Hadoop Distributed File System). Then the data can be processed using complex algorithms and pushed out to file systems, databases, and live dashboards.
4. **MLlib (Machine Learning Library)** – Apache Spark is equipped with a rich library known as MLlib. This library contains a wide array of machine learning algorithms- classification, regression, clustering, and collaborative filtering. It also includes other tools for constructing, evaluating, and tuning ML Pipelines. All these functionalities help Spark scale out across a cluster.
5. **GraphX** – Spark also comes with a library to manipulate graph databases and perform computations called GraphX. GraphX unifies ETL (Extract, Transform, and Load) process, exploratory analysis, and iterative graph computation within a single system.

**[Features](https://chartio.com/learn/data-analytics/what-is-spark/" \l "features)**

1. **Fast processing** – The most important feature of Apache Spark that has made the big data world choose this technology over others is its speed. Big data is characterized by volume, variety, velocity, and veracity which needs to be processed at a higher speed. Spark contains [Resilient Distributed Dataset (RDD)](https://intellipaat.com/blog/tutorial/spark-tutorial/programming-with-rdds/) which saves time in reading and writing operations, allowing it to run almost **ten to one hundred times faster than Hadoop**.
2. **Flexibility** – Apache Spark supports multiple languages and allows the developers to write applications in Java, Scala, R, or Python.
3. **In-memory computing** – Spark stores the data in the RAM of servers which allows quick access and in turn accelerates the speed of analytics.
4. **Real-time processing** – Spark is able to process real-time streaming data. Unlike MapReduce which processes only stored data, Spark is able to process real-time data and is, therefore, able to produce instant outcomes.
5. **Better analytics** – In contrast to [MapReduce](https://www.ibm.com/analytics/hadoop/mapreduce) that includes Map and Reduce functions, Spark includes much more than that. Apache Spark consists of a rich set of SQL queries, machine learning algorithms, complex analytics, etc. With all these functionalities, analytics can be performed in a better fashion with the help of Spark.

**What is PySpark?**

Apache Spark is written in Scala programming language. PySpark has been released in order to support the collaboration of Apache Spark and Python, it actually is a Python API for Spark. In addition, PySpark, helps you interface with Resilient Distributed Datasets (RDDs) in Apache Spark and Python programming language. This has been achieved by taking advantage of the Py4j library. Py4J is a popular library which is integrated within PySpark and allows python to dynamically interface with JVM objects. PySpark features quite a few libraries for writing efficient programs. Furthermore, there are various external libraries that are also compatible. Here are some of them:

**PySparkSQL**

A PySpark library to apply SQL-like analysis on a huge amount of structured or semi-structured data. We can also use SQL queries with PySparkSQL. It can also be connected to [**Apache Hive**](https://databricks.com/glossary/apache-hive). HiveQL can be also be applied. PySparkSQL is a wrapper over the PySpark core. PySparkSQL introduced the DataFrame, a tabular representation of structured data that is similar to that of a table from a relational database management system.

**MLlib**

MLlib is a wrapper over the PySpark and it is Spark’s machine learning (ML) library. This library uses the data parallelism technique to store and work with data. The machine-learning API provided by the MLlib library is quite easy to use. MLlib supports many machine-learning algorithms for classification, regression, clustering, collaborative filtering, dimensionality reduction, and underlying optimization primitives.

**GraphFrames**

The GraphFrames is a purpose graph processing library that provides a set of APIs for performing graph analysis efficiently, using the PySpark core and PySparkSQL. It is optimized for fast distributed computing. Advantages of using PySpark: • Python is very easy to learn and implement. • It provides simple and comprehensive API. • With Python, the readability of code, maintenance, and familiarity is far better. • It features various options for data visualization, which is difficult using Scala or Java.