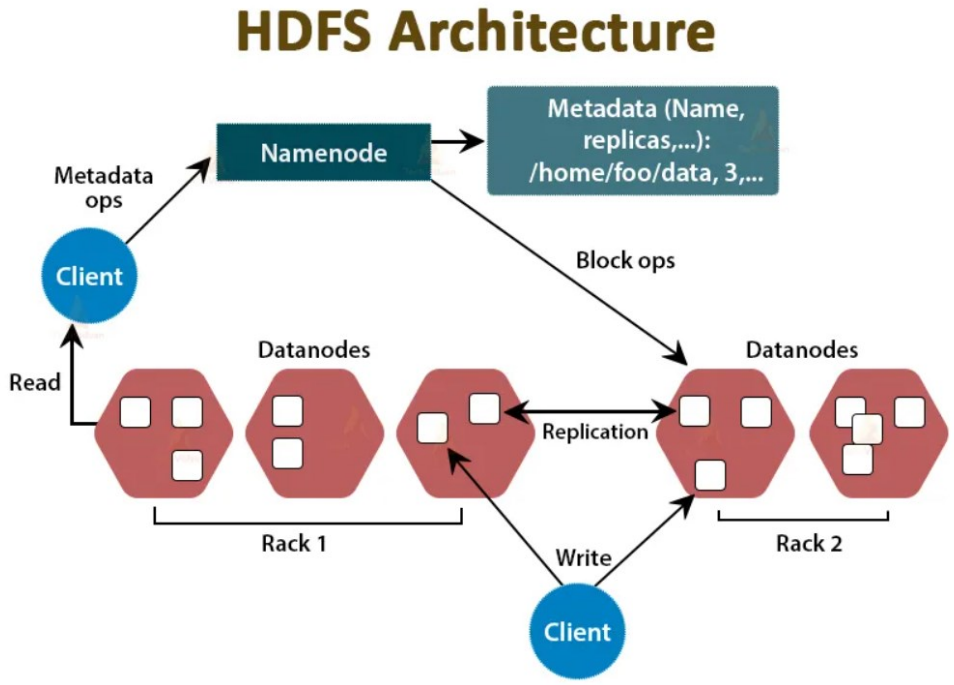
Hadoop و Spark هر دو فریمورکی متن باز یا open-source هستند که توسط شرکت Apache تولید و توسعه داده شده اند. این فریمورک ها در حوزه Big Data استفاده میشوند. اگر بخواهیم نگاهی به تعریف Big Data بیندازیم میتوانیم بگوییم Big Data اطلاعات یک مجموعه است که با سرعت بالایی تولید میشوند، حجم زیادی دارند و همچنین تنوع و گستردگی آن ها بالا است. ما برای کار با این داده ها و پردازش روی آن ها به ابزار های پیشرفته تری از پروداکت های معمول نیاز داریم که Hadopp و Spark را میتوان دو نمونه مهم از این ابزار دانست.

در سال 1999 توسعه Apache Lucene که به صورت متن باز منتشر شد، منجر به ایجاد Hadoop شد. در واقع Hadoop یک ابزار نرم افزاری open-source است که کاربرات میتوانند با استفاده از این ابزار Big Data را در گستره GB تا PB پشتیبانی میکند و با استفاده از شبکه ای که از node ها تشکیل شده است، این داده ها را مدیریت کند. در واقع میتوان گفت یک پردازش توزیع شده است که چند مولفه اصلی دارد: HDFS, YARN, MapReduce و ... که در ادامه به توضیح هر یک از این ها خواهیم پرداخت. Hadoop با زبان Java ساخته شده است که الگوریتم MapReduce آن را با استفاده از زبان های مختلفی به کمک یک کلاینت Thrift میتوان پیاده سازی کرد.

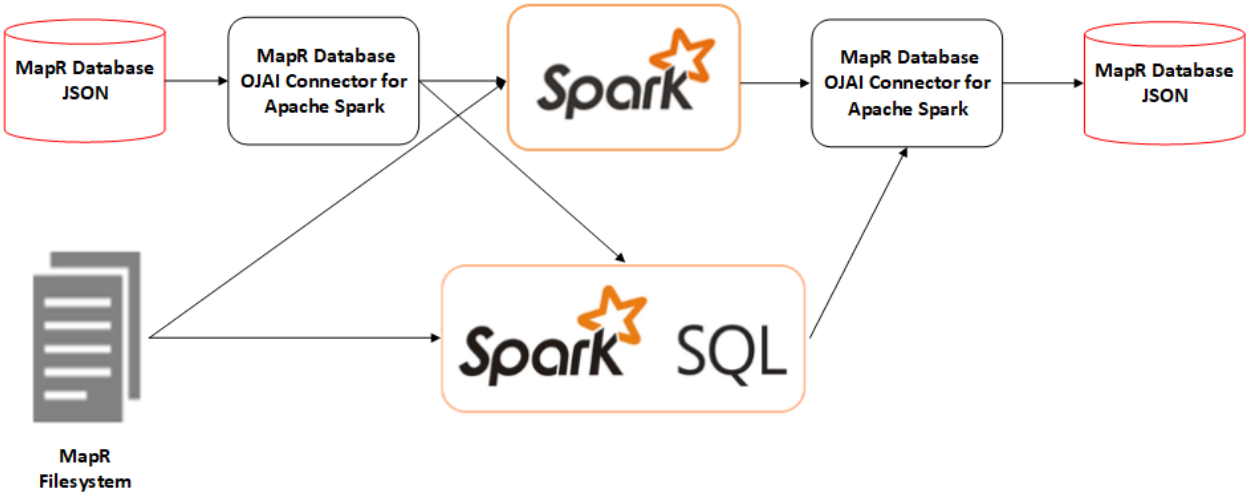
HDFS یک فایل سیستم Distributed و یا توزیع شده است که فایل ها را در قالب اصلی Hadoop ذخیره میکند. در این فایل سیستم از معماری Master-Slave استفاده میشود که داده ها را به چند block تقسیم میکند و بین data node ها به صورت تکراری که خطاپذیری را کم کند توزیع میشود. و نود اصلی از اطلاعات تمام بلاک ها و نودهای slave اگاهی دارد.



YARN یک scheduler است که زمان های اجرای برنامه را مدیریت و هماهنگ میکند.

MapReduce الگوریتمی است که داده ها را به صورت موازی پردازش میکند.

Apache Spark یک پروژه جدیدتر نسبت به Hadoop است که در سال 2012 در دانشگاه Berkley توسعه پیدا کرد. این پروژه نیز بر روی پردازش داده ها به صورت موازی بر روی یک کلاستر کار میکند اما مهمترین تفاوت آن این است که in-memory است و به جای اینکه مانند Hadoop فایل ها را در HDFS بنویسد و بخواند از RAM استفاده میکند و از مفهوم Resilient Distributed Dataset استفاده میکند. در توضیح این مفهوم میتوانیم بگوییم یک دیتا استراکچر اصلی در Spark است که از آن به نام مجموعه داده های توزیع شده انعطاف پذیر یاد میکنند. هر مجموعه داده در این RDD به پارتیشن های منطقی تقسیم میشوند که ممکن است بر روی node های مختلف cluster محاسبه شوند. در واقع مجموعه ای از رکوردهای فقط خواندنی و پارتیشن بندی شده است که Spark از آن برای دسترسی به عملیات MapReduce سریع تر و کارآمدتر استفاده میکند.



اگر بخواهیم به صورت کلی به تفاوت های Hadoop و Spark بپردازیم میتوانیم جدول زیر را مطالعه کنیم:

Fields	Hadoop	Spark
Category	یک موتور پردازش داده ساده و ابتدایی است.	یک موتور انالیز داده است.
Scalability	وقتی که حجم داده ها به سرعت زیاد میشود Hadoop با استفاده از فایل سیستم توزیع شده خودش به سرعت scale میکند.	Spark به دلیل اینکه فایل سیستم خودش را ندارد باید به HDFS متکی باشد اما باز هم از Hadoop ضعیف تر است.
Latency (Performance)	Latency در هدوپ بیشتر است زیرا از خواندن و نوشتن از طریق HDFS بر روی دیسک و روی چندین منبع ذخیره	Spark به دلیل اینکه از RAM استفاده میکند با سرعت بالاتری اجرا میشود و زمان کمتری میبرد.

	میکند و همین باعث افزایش زمان و کاهش پرفورمنس میشود.	
Spark برای پردازش های Real time و بدون ساختار کارایی دارد.	**Hadoop برای پردازش دسته ای و خطی داده ها مناسب است.**	Usage
در Spark چون امنیت و احراز هویت از طریق Shared secret و لاگ event ها تامین میشود از سطح امنیت کمتری نسبت به Hadoop برخودار است.	**Hadoop به دلیل اینکه از چندین روش تایید هویت و کنترل دسترسی ها استفاده میکند امنیت بالاتری دارد.**	Security
در Spark به دلیل انتزاع سازی و Abstraction ای که انجام شده است با عملگر های سطح بالاتری ارتباط داریم و برای استفاده راحت تر است.	**Hadoop مدل MapReduce پیچیده تری دارد و نیاز داریم که API سطح پایین تری را هندل کنیم.**	Easy in use
به دلیل محاسبات داخل مموری به scheduler خارجی نیاز ندارد.	**به یک scheduler خارجی نیاز دارد.**	Scheduler
اما در Spark ما نیاز به حافظه RAM در مقیاس بالا داریم که هزینه بیشتری نسبت به دیسک دارد.	**در هدوپ هزینه ما در کل کمتر است زیرا با هر نوع ذخیره سازی روی هر دیسکی میتواند کار خودش را انجام دهد.**	Cost

در کل موارد استفاده هر کدام به صورت کاربردی در زیر بیان کرده ایم:

Spark :

- **پردازش موازی گراف برای مدل کردن داده های**
- **دسترسی سریع به نتایج با محاسبات درون حافظه ای**
- **استفاده از الگوریتم های تکرار شونده به جای عملیات موازی**
- **تجزیه و تحلیل جریان داده های** real time
- **همه برنامه های یادگیری ماشین**

Hadoop:

- **انجام کارهایی که به زمان حساس نیستند**
- **تجزیه و تحلیل داده های آرشیو شده و تاریخی**
- **پردازش مجموعه داده های بزرگ زمانی که اندازه داده ها از حافظه موجود بیشتر است.**
- **پردازش به صورت** batch
- **اجرای یک زیرساخت تجزیه و تحلیل داده ها با بودجه کم**