传统分布式文件系统架构：Database+Sharding+Cache

Database负责数据存储

Sharding负责数据分片，在多台机器间分担负载。有水平分片和垂直分片，水平分片将数据表中数据按行进行切分，即按照某种规则将某些行切分为一个数据库，垂直分片将数据按不同功能进行分片。

Cache用于提高响应速度

这种架构优点是简单，适用于数据量和用户量小的情形，缺点是扩展性/容错性差，维护成本高

Google文件系统GFS：GFS+MapReduce+BigTable

运行于廉价的普通硬件之上，提供容错功能，采用master/slave架构。

GFS由一个master和大量的chunkserver构成，master存在单点故障问题，主要解决措施为减少master和应用之间的交互和数据流量。master所有信息储存在内存中，当master启动时master信息从chunkserver中获取，有利于master宕掉后替换备用master。

GFS对外提供create，delete，open，close，read和write操作接口。

MapReduce中Map阶段将任务分发到多个节点进行处理，Reduce阶段对多个结点的结果进行合并。

BigTable用于存储大规模结构化数据，使用GFS来存储日志和数据文件，应用程序通常不会直接操作GFS文件系统，而是操作它的上一级存储结构BigTable，

Hadoop分布式文件系统HDFS：HDFS+MapReduce+HBase+ZooKeeper

类似于Google文件系统，文件一般有3个备份。

Hadoop是磁盘级计算，数据在磁盘上，需要读写磁盘。

Zookeeper用于管理集群，记录一些概要信息，记录当前哪些节点正在用，哪些已掉线，哪些还备用等。

MapReduce是批处理系统。

Hadoop常用于离线的复杂的大数据处理。

Hadoop存在以下不足：

1. MapReduce模型不适合描述复杂的数据处理过程。
2. Job只提供Map和Reduce两个操作，表达力欠佳，Reduce任务需要等到所有Map任务完成后才可以开始，并且复杂的计算将需要大量的Job来完成。
3. 处理逻辑隐藏在代码细节中，没有整体逻辑。
4. 时延高，对于交互式数据处理和实时数据处理支持不够。
5. 对于迭代式数据处理支持较差。

Spark：

Spark相对于Hadoop更加充分的利用了内存计算，Spark最大的改进是相较于MapReduce可支持更多的操作。

Spark提供了一个集群的分布式内存抽象，RDD(Resilient Distributed Dataset)编程模型。

说Spark代替了Hadoop更应该说是代替了MapReduce。

Spark常用于离线的快速的大数据处理。

Spark适合做需要反复迭代的机器学习。

Storm：

Storm是分布式环境下的实时数据计算模型，没有外存存储部分，Storm对内存中的数据进行操作，适应实时处理，超低延迟，无需大量存储数据。

Storm是内存级计算，数据通过网络直接导入内存，数据操作均在内存完成。

Storm也称为流式计算系统。

Strom常用于在线的实时的大数据处理。