**Hadoop与MapReduce**

***Hadoop大数据开源软件生态-数据处理***

**Map-Reduce：**作为一种编程模型，用于大规模数据集（大于1TB）的并行运算。"Map（映射）"和"Reduce（归约）"，都是从函数式编程语言里借来的，还有从矢量编程语言里借来的特性。它极大地方便了编程人员在不会分布式并行编程的情况下，将自己的程序运行在分布式系统上。当前相关软件通常指定一个Map（映射）函数，用来把一组键值对映射成一组新的键值对，指定并发的Reduce（归约）函数，用来保证所有映射的键值对中的每一个共享相同的键组。

**Spark：**是加州大学伯克利分校的AMP实验室所开源的类Hadoop MapReduce的通用并行框架。具体内容以后讲解。

***Hadoop大数据开源软件生态-数据存储***

**HDFS：**即Hadoop分布式文件系统（Hadoop Distributed Filesystem），以流式数据访问模式来存储超大文件，运行于商用硬件集群上，是适合运行在通用硬件(commodity hardware)来管理网络中跨多台计算机存储的分布式文件系统。Hadoop分布式文件系统和现有的分布式文件系统有很多共同点。但同时，它和其他的分布式文件系统的区别也是很明显的。HDFS是一个高度容错性的系统，适合部署在廉价的机器上。HDFS能提供高吞吐量的数据访问，非常适合大规模数据集上的应用。HDFS放宽了一部分POSIX约束，从而实现流式读取文件系统数据。HDFS在最开始是作为Apache Nutch搜索引擎项目的基础架构而开发的。

***Hadoop大数据开源软件生态-运行管理***

**YARN：**Apache Hadoop YARN （Yet Another Resource Negotiator，另一种资源协调者）是一种新的 Hadoop 资源管理器，它是一个通用资源管理系统，可为上层应用提供统一的资源管理和调度，为集群在利用率、资源统一管理和数据共享等方面带来了巨大支持。

***MapReduce-做什么***

　　  MapReduce擅长处理大数据，这可由MapReduce的设计思想发觉，即“分而治之”。

**（1）**负责“分”，即把复杂的任务分解为若干个“简单的任务” - 数据或计算的规模相对原任务要大大缩小；并且要秉持就近计算原则，即任务会分配到存放着所需数据的节点上进行计算；这些小任务可以并行计算，彼此间几乎没有依赖关系。

**（2）**Reducer负责对map阶段的结果进行汇总。至于需要多少个Reducer，用户可以根据具体问题进行配置

***Map-Reduce具体介绍-三层含义***

        MapReduce是面向大数据并行处理的计算平台、框架和模型，它隐含了以下三层含义：

        1）MapReduce是一个基于集群的高性能并行计算平台（Cluster Infrastructure）。它允许用市场上普通的商用服务器构成一个包含数十、数百至数千个节点的分布和并行计算集群。

        2）MapReduce是一个并行计算与运行软件框架（Software Framework）。它提供了一个庞大但设计精良的并行计算软件框架，能自动完成计算任务的并行化处理，自动划分计算数据和计算任务，在集群节点上自动分配和执行任务以及收集计算结果，将数据分布存储、数据通信、容错处理等并行计算涉及到的很多系统底层的复杂细节交由系统负责处理，大大减少了软件开发人员的负担。

        3）MapReduce是一个并行程序设计模型与方法（Programming Model & Methodology）。它借助于函数式程序设计语言Lisp的设计思想，提供了一种简便的并行程序设计方法，用Map和Reduce两个函数编程实现基本的并行计算任务，提供了抽象的操作和并行编程接口，以简单方便地完成大规模数据的编程和计算处理。

***MapReduce具体介绍-主要功能***

**数据划分和计算任务调度**：系统自动将一个作业（Job）待处理的大数据划分为很多个数据块，每个数据块对应于一个计算任务（Task），并自动调度计算节点来处理相应的数据块。作业和任务调度功能主要负责分配和调度计算节点（Map节点或Reduce节点），同时负责监控这些节点的执行状态，并负责Map节点执行的同步控制。

**数据/代码互定位：**为了减少数据通信，一个基本原则是本地化数据处理，即一个计算节点尽可能处理其本地磁盘上所分布存储的数据，这实现了代码向数据的迁移；当无法进行这种本地化数据处理时，再寻找其他可用节点并将数据从网络上传送给该节点（数据向代码迁移），但将尽可能从数据所在的本地机架上寻找可用节点以减少通信延迟。

**系统优化：**为了减少数据通信开销，中间结果数据进入Reduce节点前会进行一定的合并处理；一个Reduce节点所处理的数据可能会来自多个Map节点，为了避免Reduce计算阶段发生数据相关性，Map节点输出的中间结果需使用一定的策略进行适当的划分处理，保证相关性数据发送到同一个 Reduce节点；此外，系统还进行一些计算性能优化处理，如对最慢的计算任务采用多备份执行、选最快完成者作为结果。

**出错检测和恢复：**以低端商用服务器构成的大规模MapReduce计算集群中，节点硬件（主机、磁盘、内存等）出错和软件出错是常态，因此 MapReduce需要能检测并隔离出错节点，并调度分配新的节点接管出错节点的计算任务。同时，系统还将维护数据存储的可靠性，用多备份冗余存储机制提高数据存储的可靠性，并能及时检测和恢复出错的数据。