Hadoop

> Hadoop是一个由Apache基金会所开发的分布式系统基础架构。用户可以在不了解分布式底层细节的情况下，开发分布式程序。充分利用集群的威力进行高速运算和存储。Hadoop实现了一个分布式文件系统（Hadoop Distributed File System），其中一个组件是HDFS。HDFS有高容错性的特点，并且设计用来部署在低廉的（low-cost）硬件上；而且它提供高吞吐量（high throughput）来访问应用程序的数据，适合那些有着超大数据集（large data set）的应用程序。HDFS放宽了（relax）POSIX的要求，可以以流的形式访问（streaming access）文件系统中的数据。***Hadoop的框架最核心的设计就是：HDFS和MapReduce。HDFS为海量的数据提供了存储，而MapReduce则为海量的数据提供了计算***

Hadoop被公认是一套行业大数据标准开源软件，在分布式环境下提供了海量数据的处理能力。几乎所有主流厂商都围绕Hadoop开发工具、开源软件、商业化工具和技术服务。

可以把Hadoop理解为大数据存储与处理的规范或者说解决方案。它是多种技术的总称。

Hadoop核心包括三部分：

1.HDFS: Hadoop Distributed File System 分布式文件系统

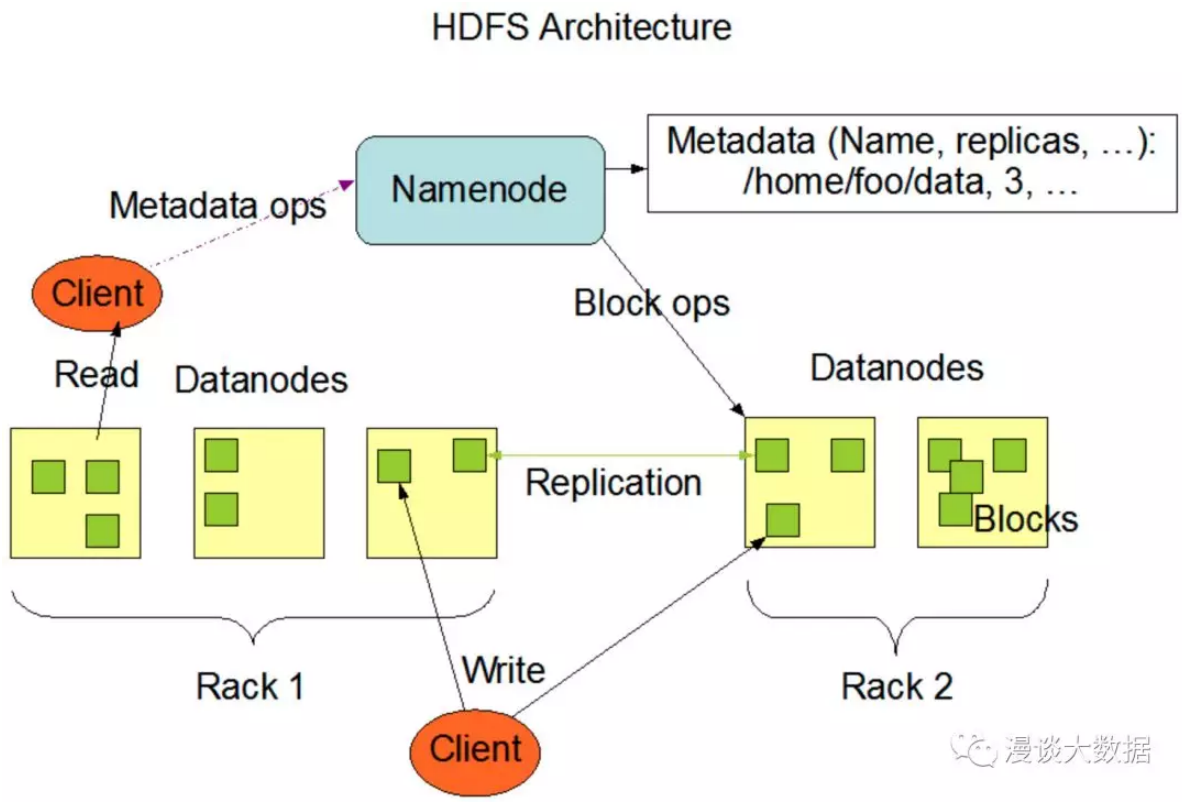
2.YARN: Yet Another Resource Negotiator 资源管理调度系统

3.MapReduce：分布式运算框架

分别用来存储数据，资源调度，处理数据

HDFS

> HDFS，是Hadoop Distributed File System的简称，是Hadoop抽象文件系统的一种实现。是Hadoop项目的核心子项目，承担存储数据的功能。可以运行在普通机器构成的集群，具有高容错，高可靠性，易扩展的特性。



- **Block数据块;**

1. 基本存储单位，一般大小为64M（配置大的块主要是因为：1）减少搜寻时间，一般硬盘传输速率比寻道时间要快，大的块可以减少寻道时间；2）减少管理块的数据开销，每个块都需要在NameNode上有对应的记录；3）对数据块进行读写，减少建立网络的连接成本）
2. 一个大文件会被拆分成一个个的块，然后存储于不同的机器。如果一个文件少于Block大小，那么实际占用的空间为其文件的大小
3. **基本的读写单**位，类似于磁盘的页，每次都是读写一个块
4. **每个块都会被复制到多台机器**，默认复制3份

- **NameNode**
  1. 存储文件的metadata，运行时所有数据都保存到内存，整个HDFS可存储的文件数受限于NameNode的内存大小
  2. **一个Block在NameNode中对应一条记录**（一般一个block占用150字节），如果是大量的小文件，会消耗大量内存。同时map task的数量是由splits来决定的，所以用MapReduce处理大量的小文件时，就会产生过多的map task，线程管理开销将会增加作业时间。处理大量小文件的速度远远小于处理同等大小的大文件的速度。因此Hadoop建议存储大文件
  3. **数据会定时保存到本地磁盘，但不保存block的位置信息，而是由DataNode注册时上报和运行时维护**（NameNode中与DataNode相关的信息并不保存到NameNode的文件系统中，而是NameNode每次重启后，动态重建）
  4. NameNode失效则整个HDFS都失效了，所以要保证NameNode的可用性
- **Secondary NameNode**
  1. 定时与NameNode进行同步（定期合并文件系统镜像和编辑日志，然后把合并后的传给NameNode，替换其镜像，并清空编辑日志，类似于CheckPoint机制），但NameNode失效后仍需要手工将其设置成主机
- **DataNode**
  1. 保存具体的block数据
  2. 负责数据的读写操作和复制操作
  3. DataNode启动时会向NameNode报告当前存储的数据块信息，后续也会定时报告修改信息
  4. DataNode之间会进行通信，复制数据块，保证数据的冗余性

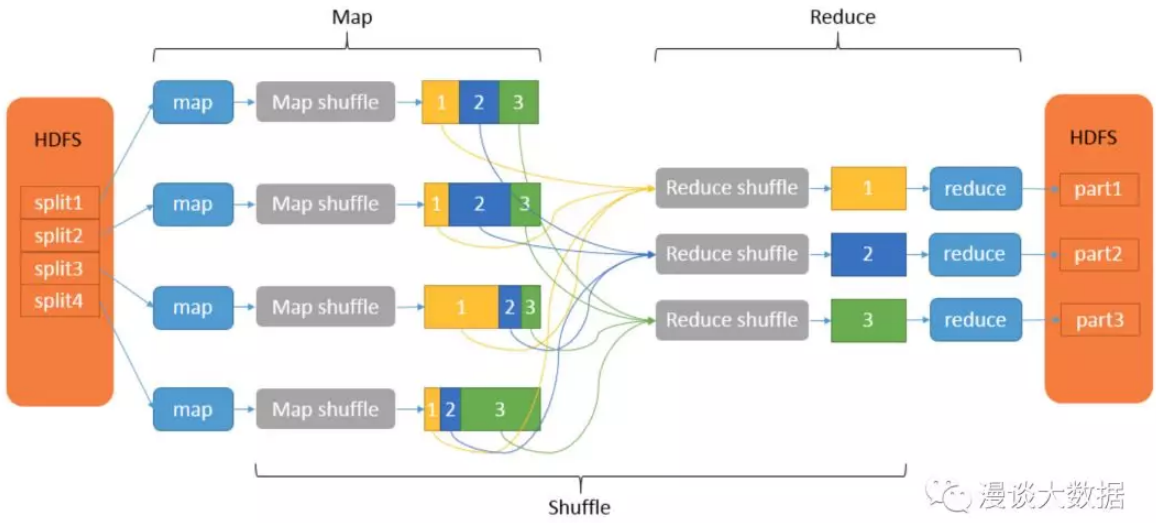
  client进行读/写请求时，首先都要和NameNode交互，由NameNode返回block以及block所在的DataNode的位置信息，然后client再和DataNode中对应的block交互。

MapReduce

> Mapreduce 是一个分布式运算程序的编程框架，是用户开发“基于 hadoop 的数据分析 应用”的核心框架
> Mapreduce 核心功能是将用户编写的业务逻辑代码和自带默认组件整合成一个完整的 分布式运算程序，并发运行在一个 hadoop 集群上
>
> 如果说HDFS承担的是海量数据的存储功能，那么MapReduce就是对海量数据进行计算的框架。

MapReduce 作为一个分布式的计算框架，编程模型起源于函数式编程语言里的 map 和 reduce 函数。

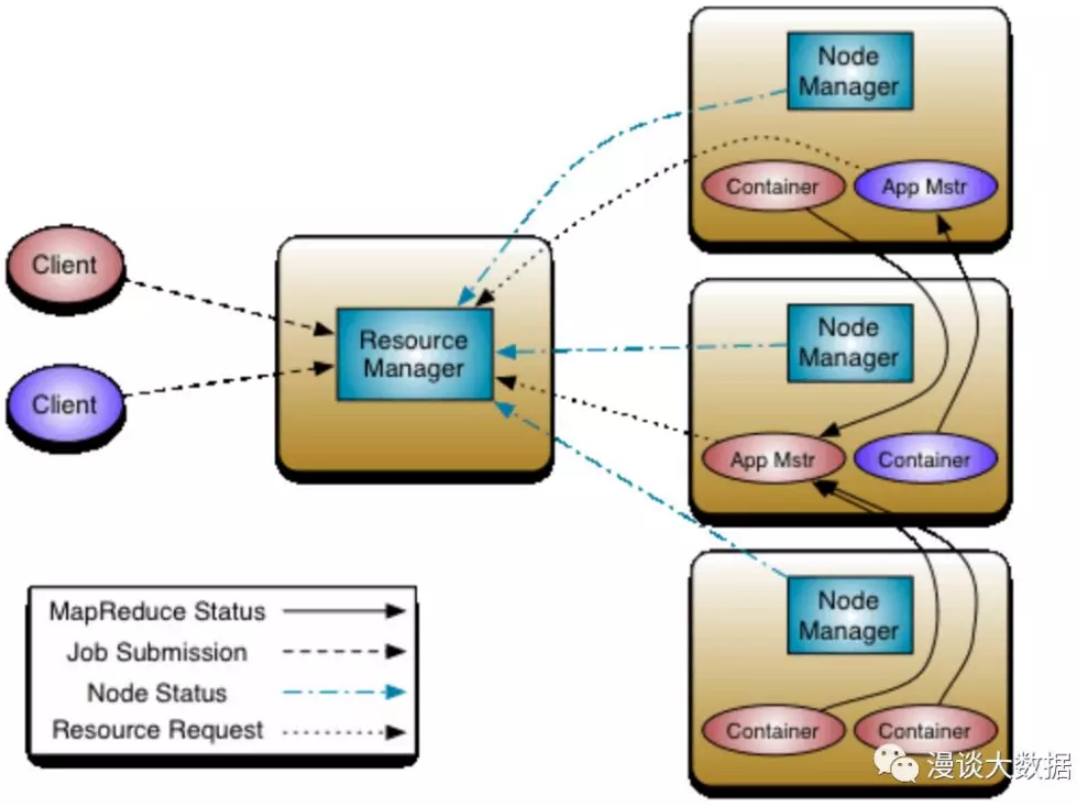
得益于 HDFS 已经将数据以 block 为单位切割，MapReduce 框架也就可以很轻松地将数据的初步处理以多个 map 函数的形式分发到不同的服务器上并行执行。map 阶段处理的结果又以同样的思路分布到不同的服务器上并行做第二阶段的 reduce 操作，以得到想要的结果。



Map阶段将任务分发到不同的服务器节点上并行处理，Reduce阶段将Map阶段处理后的多个节点的结果做合并处理。shuffle 操作处理 map 阶段的输出以作为 reduce 阶段的输入，是个承上启下的关键过程。

YARN（Yet-Another-Resource-Negotiator）

> 分布式存储的时候由NameNode承担了管理者的角色，分布式计算中也需要引入一个类似的管理者，负责计算资源的管理协调，计算任务的分配、任务进度和状态的监控、失败任务的重跑等职责。这就是YARN的功能。



YARN将资源管理和任务调度监控拆分成独立的进程：一个全局的资源管理和一个每个作业的管理（ApplicationMaster） ResourceManager 和 NodeManager 提供了计算资源的分配和管理，而 ApplicationMaster 则完成应用程序的运行

- **ResourceManager:** 全局资源管理和任务调度

- **NodeManager:** 单个节点的资源管理和监控
- **ApplicationMaster:** 单个作业的资源管理和任务监控
- **Container:** 资源申请的单位和任务运行的容器

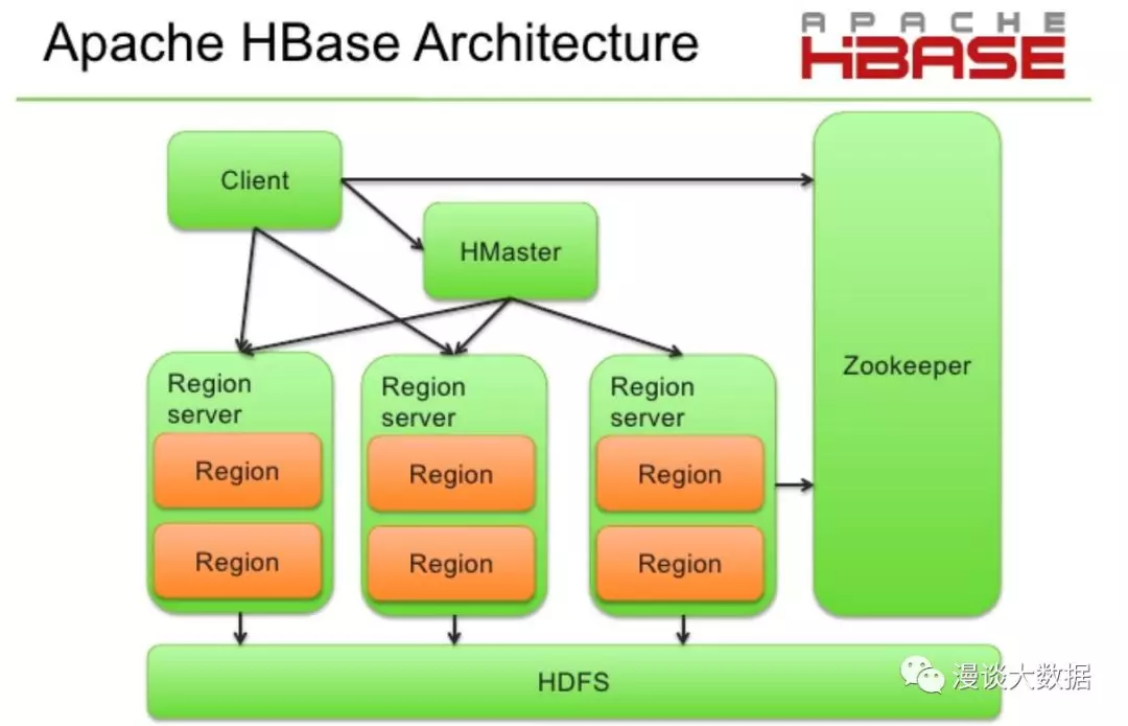
HBase

我们可以把计算分为两种类型：

- 离线（offline）计算，或者叫批量（batch）计算/处理
- 在线（online）计算，或者叫实时（realtime）计算、流（stream）计算/处理

  在大数据领域，批量计算一般用于用于对响应时间和延迟不敏感的场景。有些是业务逻辑本身就不敏感，比如天级别的报表，有些则是由于数据量特别大等原因而被迫牺牲了响应时间和延迟。而相反，流计算则用于对响应时间和延迟敏感的场景，如实时的 PV 计算等。

  HDFS适合的是离线数据的存储，而HBase在线存储的常用方案。而Hbase也是基于HDFS的，所有的数据都以文件的形式保存在 HDFS 上。这样就天然拥有了横向扩展的能力，以及高可用等特性。



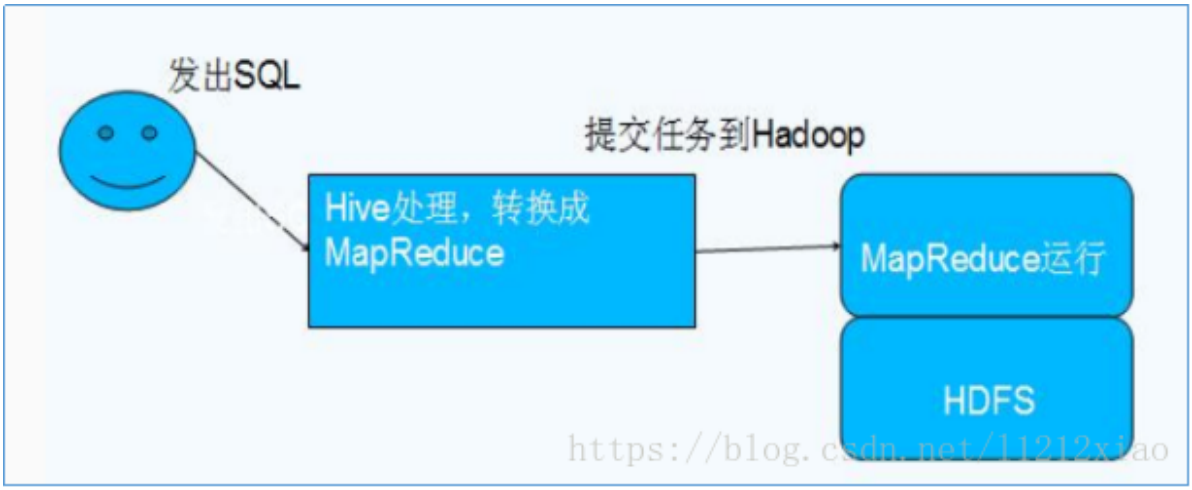
- 每个节点上都有一个叫 RegionServer 的程序来提供对数据的读写服务
- 每个表被拆分成很多个 Region，然后均衡地分散在各个 RegionServer 上
- 另外有个 HMaster 的服务，负责对表的创建、修改、删除已经 Region 相关的各种管理操作。

很容易看出，HBase 的分布式和 HDFS 非常像，RegionServer 类似于 DataNode，HMaster 类似于 NameNode，Region 类似于 Block。

同样的，MapReduce适合于批数据处理/计算，如果要实现流处理，目前常用的框架是Spark Streaming、Storm 和 Flink。

HIVE

> hive是基于Hadoop的一个数据仓库工具，用来进行数据提取、转化、加载，这是一种可以存储、查询和分析存储在Hadoop中的大规模数据的机制。hive数据仓库工具能将结构化的数据文件映射为一张数据库表，并提供SQL查询功能，能将SQL语句转变成MapReduce任务来执行。Hive的优点是学习成本低，可以通过类似SQL语句实现快速MapReduce统计，使MapReduce变得更加简单，而不必开发专门的MapReduce应用程序。hive十分适合对数据仓库进行统计分析。



Hive是基于Hadoop的一个数据仓库工具，可以将结构化的数据文件映射为一张数据库表，并提供类SQL查询功能。

HIve可以让开发者像使用SQL语法一样，直接从Hadoop中查询数据。其实是对Hadoop查询的一次封装，最终还是需要使用MapReduce完成真正的查询操作。