1. 有关 hadoop 补充知识点 1

MapReduce、HDFS 和 YARN 是 Hadoop 生态系统的三个核心组件，它们共同协作，实现了大规模数据处理的分布式计算和存储。

（1）HDFS（分布式文件系统）提供**分布式存储服务**。HDFS 将大文件分割成小块，每个小块都被分散存储在多台服务器上。HDFS 采用主/从(Master/Slave)架构，一般一个 HDFS 集群由一个 NameNode、一个 SecondaryNameNode 和多个 DataNode 组成。NameNode 是 HDFS 集群的主节点，负责存储和管理文件系统的**元数据(节点信息)**。SecondaryNameNode 辅助 NameNode，分担其工作量，用于同步元数据信息。DataNode 是 HDFS 集群从节点，存储实际的数据，并汇报存储信息给 NameNode。

元数据是数据的“说明书”，完善的元数据有利于数据使用者了解企业有什么数据，它们分布在哪里，数据的业务含义是什么，数据口径及颗粒度是怎样的，需要使用数据时应该向谁提出申请，以及如何获取数据。

HDFS 的运行机制可以分为以下几个步骤：

- 客户端向 NameNode 请求访问文件，NameNode 返回文件的位置信息。
- 客户端根据位置信息向 DataNode 请求数据块。
- DataNode 将数据块返回给客户端。
- 客户端将数据块拼接成完整的文件。

HDFS 还提供了数据的冗余备份机制，可以在某些数据节点失效时保证数据不丢失。

（2）MapReduce 是 Hadoop 的**分布式计算框架**，它将计算任务分成两个阶段：Map 阶段和 Reduce 阶段。运行机制可以分为以下几个步骤：

- 客户端提交 MapReduce 作业，作业被分配到一个计算节点上的 JobTracker。
- JobTracker 将作业分成多个任务，并将任务分配给相应的计算节点上的 TaskTracker。
- TaskTracker 执行 Map 和 Reduce 任务，并将结果存储到 HDFS 中。
- JobTracker 将 Map 和 Reduce 的输出结果进行合并，最终输出结果。

（3）YARN 是 Hadoop 的**资源管理系统**，负责管理计算资源和任务的调度。运行机制可以分为以下几个步骤：

- 应用程序通过客户端向 ResourceManager 提交任务。
- ResourceManager 为应用程序分配一个 ApplicationMaster，ApplicationMaster 负责管理任务的执行。
- ApplicationMaster 向 ResourceManager 请求计算资源，ResourceManager 为 ApplicationMaster 分配资源，并将资源信息返回给 ApplicationMaster。
- ApplicationMaster 将任务分配给相应的计算节点上的 NodeManager 进行计算。
- NodeManager 执行任务并将结果返回给 ApplicationMaster。
- ApplicationMaster 将结果存储到 HDFS 中。

YARN 可以支持多种类型的计算框架，如 MapReduce、Spark 等，这些框架可以共享 YARN 提供的资源管理能力。

2. 有关 hadoop 补充知识点 2

Hadoop 中，每个服务器可以运行一个 NameNode 节点，多个 DataNode 节点和一个 SecondaryNameNode 节点。

（1）NameNode 节点：负责管理 HDFS（Hadoop 分布式文件系统）的命名空间，存储了所有文件和目录的元数据。它是 Hadoop 集群的主节点之一，通常是单点故障。

（2）DataNode 节点：负责实际存储数据块，并响应客户端或其他节点的读写请求。集

群中可以有多个 DataNode 节点。

（3）SecondaryNameNode 节点：负责定期合并和压缩 NameNode 的编辑日志，以避免其变得过大，并生成检查点以用于恢复。SecondaryNameNode 并不是 NameNode 的备份，它只是协助 NameNode 执行某些操作。

对于单台物理服务器，通常会在同一台机器上运行多个 Hadoop 节点，以实现分布式计算和存储。通常情况，可以将 NameNode 节点和 SecondaryNameNode 节点分配到同一台服务器上，而将 DataNode 节点分配到另一台或多台服务器上。这是因为 NameNode 节点和 SecondaryNameNode 节点通常需要更多的计算和存储资源，而 DataNode 节点则需要更多的磁盘存储空间。将它们分开部署可以更好地利用服务器资源，提高整个集群的性能和可靠性。

当然，这只是一种常见的节点组合方式，并不是唯一的选择。具体的节点组合方式需要根据实际需求和资源情况进行调整和优化。

3. 有关 hadoop 补充知识点 3

ResourceManager 和 NodeManager 是 YARN 中两个重要组件，与 NameNode 节点、DataNode 节点和 SecondaryNameNode 节点分别具有不同的功能和关系。

ResourceManager 负责整个集群的资源管理和调度。ResourceManager 会接收客户端或应用程序提交的请求，分配可用的资源给它们，并监控和控制整个集群的运行状态。ResourceManager 通常运行在 Hadoop 集群中的一个节点上，可以和 NameNode 节点以及 SecondaryNameNode 节点等组合在一起。

NodeManager 负责管理和监控单个节点上资源使用情况，运行在每个 DataNode 节点上，并与 DataNode 节点协同工作，负责管理节点上的容器（Container）以及处理来自 ResourceManager 的命令和任务。NodeManager 还负责将节点上的资源信息发送给 ResourceManager，以供全局资源调度。

ResourceManager 通常与 NameNode 节点和 SecondaryNameNode 节点组合在一起，负责整个集群的资源调度和管理；而 NodeManager 则与 DataNode 节点组合在一起，负责单个节点上的资源管理和监控。

4. 有关 hadoop 补充知识点 4

Hadoop 集群的主要运行流程：

（1）客户端提交作业请求：客户端通过向 ResourceManager 提交应用程序或作业请求，请求分配所需的计算资源和存储资源。

（2）ResourceManager 分配资源：ResourceManager 接收到客户端提交的请求后，会根据可用资源情况和调度策略为该作业分配计算和存储资源。

（3）NodeManager 启动容器：一旦 ResourceManager 为作业分配了所需的资源，它会将分配的资源信息发送给对应的 NodeManager，NodeManager 会根据资源分配信息启动一个或多个容器（Container）。

（4）容器中运行任务：容器启动后，作业中的任务（Task）会在容器内运行。任务的具体执行过程由 YARN 框架管理和控制。

（5）任务读写数据：任务在运行过程中需要读写 HDFS 中的数据，HDFS 会根据副本策略将数据块分布到多个 DataNode 节点上，并保证数据的可靠性和一致性。

（6）计算结果输出：任务执行完毕后，计算结果会被输出到 HDFS 中的指定目录，以供后续的计算和处理使用。

（7）故障检测与恢复：整个集群的运行过程中可能会出现各种故障，如节点宕机、网络中断等，Hadoop 集群通过各种机制来检测和恢复故障，保证集群的稳定和可靠性。

5. 有关 hadoop 补充知识点 5

Namenode 保存文件系统元数据镜像，namenode 在内存及磁盘（fsimage 和 editslog）上分别存在一份元数据镜像文件，内存中元数据镜像保证了 hdfs 文件系统文件访问效率，磁盘上的元数据镜像保证了 hdfs 文件系统的安全性。

namenode 在磁盘上的两类文件组成：

fsimage 文件：保存文件系统至上次 checkpoint 为止目录和文件元数据。

edits 文件：保存文件系统从上次 checkpoint 起对 hdfs 的所有操作记录日志信息。

hadoop 集群搭建完成后，在集群第一次启动时首先需要对 namenode 节点格式化，之后启动不需要格式化，因为 Hadoop 生态的文件系统 HDFS 类似一块磁盘，初次使用需要格式化。首次安装格式化（format）主要作用是在本地文件系统生成 fsimage 文件。格式化后启动 hdfs。

（1）首次启动 hdfs 过程：

启动 namenode：读取 Hadoop 的配置文件，包括 hdfs-site.xml、core-site.xml、mapred-site.xml、yarn-site.xml 等文件，确定各种配置参数的值。

读取 fsimage 生成内存中元数据镜像。

启动 datanode：

向 namenode 注册；

向 namenode 发送 blockreport。

启动成功后，client 可以对 HDFS 进行目录创建、文件上传、下载、查看、重命名等操作，更改 namespace 的操作将被记录在 edits 文件中。

（2）之后启动 HDFS 文件系统过程：

启动 namenode：

读取 fsimage 元数据镜像文件，加载到内存中。

读取 editlog 日志文件，加载到内存中，使当前内存中元数据信息与上次关闭系统时保持一致。然后在磁盘上生成一份同内存中元数据镜像相同的 fsimage 文件，同时生成一个新的 null 的 editlog 文件用于记录以后的 hdfs 文件系统的更改。

启动 datanode：

向 namenode 注册；

向 namenode 发送 blockreport。

启动成功后，client 可以对 HDFS 进行目录创建、文件上传、下载、查看、重命名等操作，更改 namespace 的操作将被记录在 editlog 文件中。

（3）SecondaryNameNode

辅助 namenode，不能代替 namenode。

SecondaryNameNode 的主要作用是用于合并 fsimage 和 editlog 文件。在没有 SecondaryNameNode 守护进程的情况下，从 namenode 启动开始至 namenode 关闭期间所有的 HDFS 更改操作都将记录到 editlog 文件，这样会造成巨大的 editlog 文件，所带来的直接危害就是下次启动 namenode 过程会非常漫长。

在启动 SecondaryNameNode 守护进程后，每当满足一定的触发条件（每 3600s、文件数量增加 100w 等），SecondaryNameNode 都会拷贝 namenode 的 fsimage 和 editlog 文件到自己的目录下，首先将 fsimage 加载到内存中，然后加载 editlog 文件到内存中合并 fsimage 和 editlog 文件为一个新的 fsimage 文件，然后将新的 fsimage 文件拷贝回 namenode 目录下。并且声称新的 editlog 文件用于记录 DFS 的更改。