作业7

1. 简述为什么会有Spark

课件内容总结

人们开始关注大数据处理所需要的其他各种计算模式和系统，尤其以内存计算为核心、集诸多计算模式之大成的Spark生态系统为典型代表。

Spark是加州大学伯克利分校AMP实验室2009年开发的通用内存并行计算框架。Spark于2013年6月进入Apache成为孵化项目，8个月后成为Apache顶级项目。

Spark更快、支持更多语言、集成更多计算模式、跨平台性更好。

个人总结

直观而言，因为**UC Berkeley**的**AMPLab**把它开发出来了。

追根溯源到问什么要开发`Spark`，那应该有两方面的原因：

1. Hadoop有诸多缺点，如大量的文件IO、内存空间的浪费、仅有Map和Reduce的并行化操作、没有流式处理等等。
2. Hadoop并没有为这些缺点提供较好的解决方案，给予Spark差异化竞争的空间。

市场有需求，那资本自然会推动这方面的开发，于是一个意在补全Hadoop短板的Spark应运而生。

2. 对比Hadoop和Spark

课件内容

Hadoop	**Spark**
Distributed storage + Distributed compute	Distributed compute only
MapReduce framework	Generalized computation
Data stored on disk (HDFS)	Data stored on disk and in memory
Not ideal for iterative work	Great at iterative workloads
Batch process (good for large data volumes)	Faster on disk and memory
	Java, Python, Scala supported

个人总结

	Hadoop	Spark
Category	Basic Data processing engine	Data analytics engine
Usage	Batch processing with huge volume of data	Processing real time data, from real time events
Latency	High latency computing	Low latency computing
Data	Process data in batch mode	Can process interactively
Ease of use	Hadoop's MapReduce model is complex and programmers need to handle low-level APIs	Easier to use, abstraction enables a user to process data using high-level operators
Scheduler	External job scheduler is required	In-memory computation
Security	Highly secure	Less secure compared to Hadoop
Cost	Low	Costlier than Hadoop since it has in-memory solution

3. 简述Spark的技术特点

课件内容

- RDD: Spark提出的弹性分布式数据集(Resilient Distributed Datasets)，是Spark最核心的分布式数据抽象，Spark的很多特性都和RDD密不可分。
- Transformation & Action: Spark通过RDD的两种不同类型的运算实现了惰性计算，即在RDD的Transformation运算时，Spark并没有进行作业的提交；而在RDD的Action操作时才会触发SparkContext提交作业。
- Lineage: 为了保证RDD中数据的鲁棒性，Spark系统通过血统关系来记录一个RDD时如何通过其他一个或多个父类RDD转变过来的，当这个RDD的数据丢失时，Spark可以通过它父类的RDD重新计算。
- Spark调度：Spark采用了事件驱动的Scala库类Akka来完成任务的启动，通过复用线程池的方式来取代MapReduce进程或者线程启动和切换的开销。
- API: Spark使用Scala语言进行开发，并且默认Scala作为其编程语言。因此，编写Spark程序比MapReduce程序要简洁得多。同时，Spark系统也支持Java、Python语言进行开发。
- Spark生态: Spark SQL、Spark Streaming、GraphX等为Spark的应用提供了丰富的场景和模型，适合应用于不同的计算模式和计算任务。
- Spark部署: Spark拥有Standalone、Mesos、YARN等多种部署方式，可以部署在多种底层平台上。
- 适用于需要多次操作特定数据集的应用场合。需要反复此操作的次数越多，所需读取的数据量越大，受益越大，数据量小但是计算密度较大的场合，受益就相对较小。
- 由于RDD的特性，Spark不适用异步细粒度更新状态的应用。

- 数据量不是特别大，但是要求实时统计分析需求。
- 综上所述，Spark是一种基于内存的迭代式分布式计算框架，适合于完成一些迭代式、关系查询、流式处理等等计算密集型任务。