# 1 Spark

## 1.1 SQLContext

### 1.1.1 SparkContext，SQLContext，SparkSession的区别

SparkContext是Spark 1.\*中的Spark SQL的入口，必须先创建SparkContext。再用SparkContext创建SQLContext。

图1（先创建SparkContext，再用SparkContext创建SQLContext）



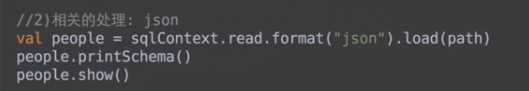
SparkSession是Spark2.\*中的入口。

图2



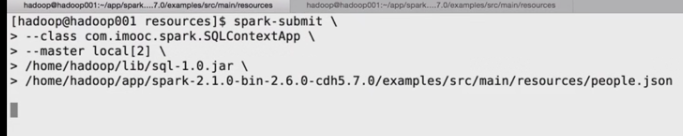
### 1.1.2 SQLContext读取JSON文件

图1



## 1.2 服务器上运行Spark SQL的jar包

图1

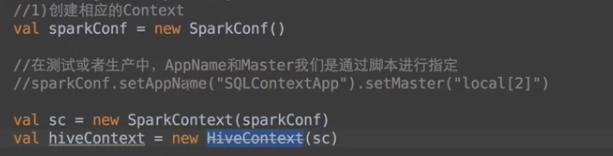


## 1.3 HiveContext

SparkContext是Spark 1.\*中的Spark SQL的入口，必须先创建SparkContext。再用SparkContext创建HiveContext。

（注意：使用HiveContext是不需要搭建Hive环境的）

图1



### 1.3.1 HiveContext显示table的数据

图1



## 1.4 SparkSession

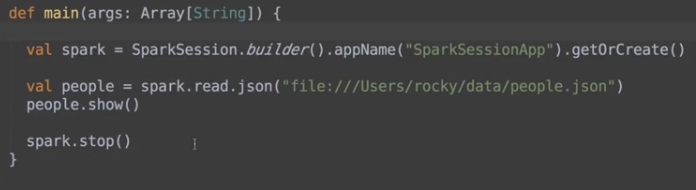
SparkSession是Spark2.\*中的入口。

图1



### 1.4.1 SparkSession读取JSON文件

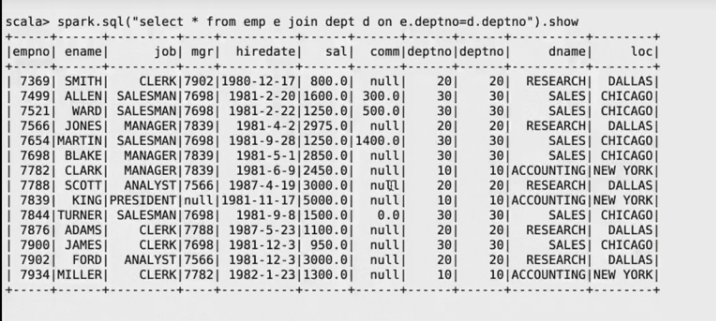
图1



## 1.5 Spark SQL相比Hive的速度

Spark SQL比Hive SQL的执行速度快很多。

图1



## 1.6 Spark thriftserver

启动thriftserver，然后通过客户端（beeline/code）去连接该thriftserver。

图1



图2



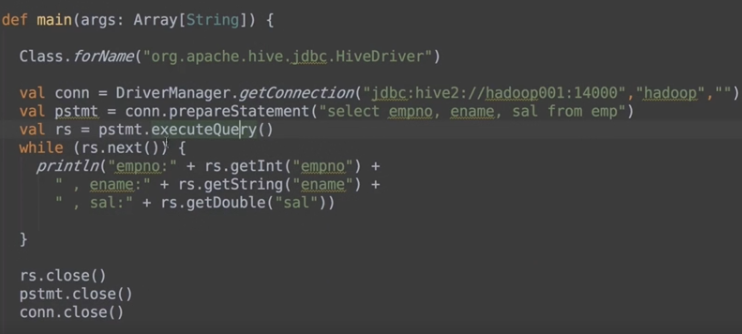
### 1.6.1 thriftserver和spark-shell/spark-sql的区别

1.每一次启动spark-shell或spark-sql都会启动一个Spark Application。

2.启动thriftserver时会启动一个Spark Application，但后面无论多少个客户端连接，永远都是一个Spark Application。

### 1.6.2 JDBC访问thriftserver

图1



## 1.7 DataFrame和DataSet

DataSet是一个分布式数据集，DataFrame是一个以列（列名，列类型，列值）的形式构成的分布式数据集。

图1



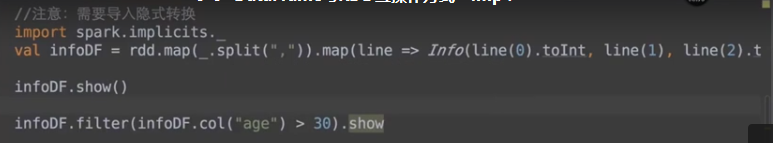
图2（DataFrame的基本API）



### 1.7.1 反射的方式根据RDD获取DataFrame

这种方式必须预先知道schema。

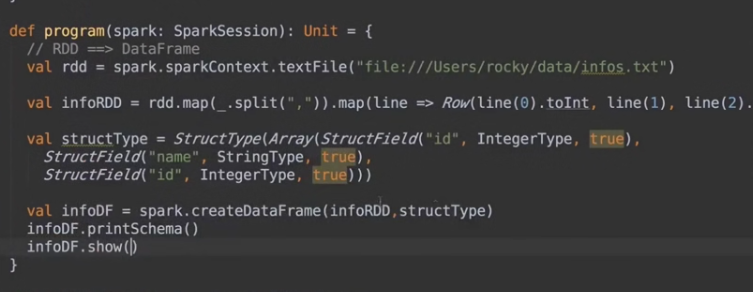
图1



### 1.7.2 structType的方式根据RDD获取DataFrame

这种方式不需要预先知道schema。

图1



# 2 Spark Streaming

Spark Streaming是一个实时流处理框架。

## 2.1 实时流处理

### 2.1.1 为什么需要实时流处理

比如对于需求：统计慕课网主站上每个课程的地域信息（由IP可以得到地域信息）。

对于1小时，10分钟，1分钟级别需求或许都可以采用离线计算（mapreduce）的方式，但对于秒级别的需求只能用实时流处理，因为离线计算可能光启动进程就已经超过1秒了。

### 2.1.2 什么时候需要进行实时流处理

1.时效性高：比如，秒级别的需求

2.数据量大

### 2.1.3 实时流处理的特点

1.实时：时效性高

2.流式：数据连绵不断（像滔滔江水连绵不绝），7\*24小时连续运作

### 2.1.4 实时流计算和离线计算的区别

数据来源：

1. 离线计算的数据来源于HDFS上的历史数据，数据量很大。
2. 实时流计算的数据来源于消息队列（比如kafka），数据量一般是实时新增/修改的某一笔增量数据，增量数据很小。

处理过程：

1. 离线计算用mapreduce处理数据。
2. Spark实时流计算用Spark（DStream/SS）处理数据。

处理速度：

1. 离线慢，实时块。

进程：

1. 离线计算map进程和reduce进程用完销毁。
2. Spark Streaming 7\*24小时不停运作。

### 2.1.5 流行的实时流框架

1.Apache Storm：完全实时，来一笔数据就计算。

2.Apache Spark Streaming：Spark Streaming是Spark的一个子模块，不是完全实时，定时间（比如每2分钟，每5分钟）处理N笔数据。

3.IBM Stream。

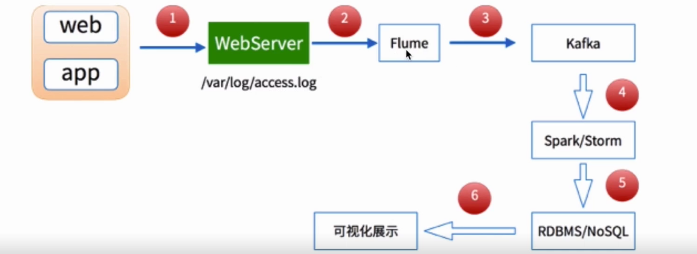
4.Kafka Streaming。

5.Yahoo! S4。

6.Apache Flink。

### 2.1.6 实时流架构

图1



1. 前端的请求到达web服务器（Web Server或Ngnix）后，会产生日志/var/log/access.log。
2. 通过分布式日志收集框架Flume收集web服务器的日志/var/log/access.log。
3. Flume收集日志后，丢到Kafka中。

（注意：其实Flume可以直接丢进Spark/Storm，但高峰期如果直接丢进Spark/Storm可以会导致Spark/Storm崩掉，所以Kakfa起到了缓冲的作用）

1. Spark/Storm处理完数据后，将结果写到关系型数据库（RDBMS）或NoSQL数据库中。
2. 通过可视化展示来观察统计数据是否有问题。

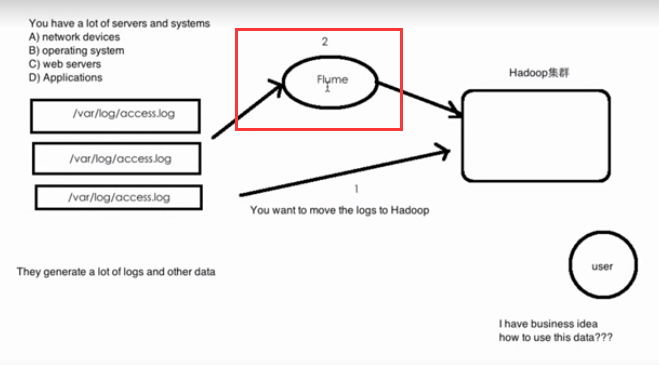
# 3 Flume

Flume是一个分布式日志收集框架。

## 3.1 为什么需要Flume

我们需要从日志中收集有价值的信息，因为日志数据量很大，所以需要将日志文件从web服务器收集到Hadoop集群中进行计算，这个收集过程Flume可以完成。

图1



## 3.2 Flume架构

每一个agent的组成包括：

Source：数据来源

Channel：数据缓存池，当Channel满了才将数据写到Sink中。

Sink：数据目的地

图1（单个agent）

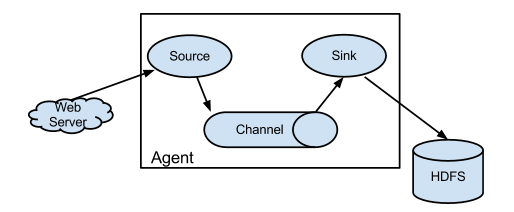


图2（多个agent）

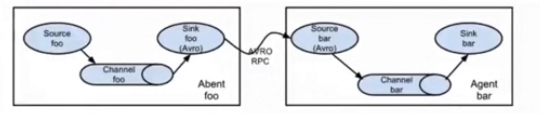


图3（多个agent）

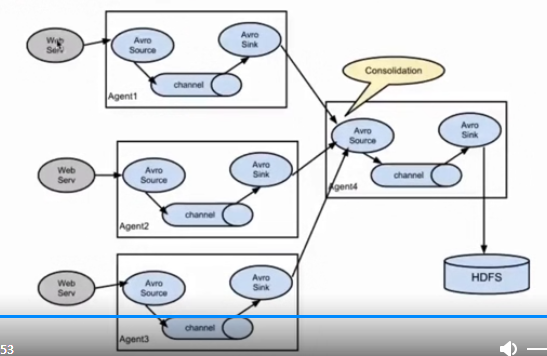


图4（多个Channel，多个Sink）

