本期内容：

1. Spark Streaming于Kafka链接的问题
2. KafkaReceiver

无故的数据链接，不正常的数据波动，数据来源一般都来自于kafka

1. Receiver
2. Direct

Zookeeper一般都是集群，Zookeeper对数据的管理是一个树状结构，跟zookeeper建立链接的话，就会被认为是树上的一个节点

// Create threads for each topic/message Stream we are listening  
 **val** topicMessageStreams = *consumerConnector*.createMessageStreams(  
 topics, keyDecoder, valueDecoder)  
  
 **val** executorPool =ThreadUtils.newDaemonFixedThreadPool(topics.values.sum, "KafkaMessageHandler")  
 **try** {  
 // Start the messages handler for each partition  
 topicMessageStreams.values.foreach { streams =>  
 streams.foreach { stream => executorPool.submit(**new** MessageHandler(stream)) }  
 }  
 } **finally** {  
 executorPool.shutdown() // Just causes threads to terminate after work is done  
 }  
}

Topic下的message,用线程池并发的读取获取的数据

Spark 2.x支持静态功能检查，而python不支持静态功能检查，所以最好不要用Python

Kafka Receiver是被zookeeper来管理的，*/\*\* zookeeper session timeout \*/***val** *zkSessionTimeoutMs* = props.getInt("zookeeper.session.timeout.ms", 6000)

进行ACK确认的超时时间是六秒钟，如果有一个Receiver在六秒钟没有响应的话，zookeeper会对所有的Kafka Receiver的客户端来重新分配资源，再重新分配的时间段里抓不到数据，网络部稳定更加麻烦了：会直接造成数据的极大的波动（这个值是真正影响receiver状态的值，zookeeper就会根据注册的客户端进行资源的管理，kafka的receiver是被zookeeper管理的，这时候需要进行ack的确认）

**class** ZKConfig(props: VerifiableProperties) {  
 */\*\* ZK host string \*/* **val** *zkConnect* = props.getString("zookeeper.connect")  
 */\*\* zookeeper session timeout \*/* **val** *zkSessionTimeoutMs* = props.getInt("zookeeper.session.timeout.ms", 6000)  
 */\*\* the max time that the client waits to establish a connection to zookeeper \*/* **val** *zkConnectionTimeoutMs* = props.getInt("zookeeper.connection.timeout.ms",*zkSessionTimeoutMs*)  
 */\*\* how far a ZK follower can be behind a ZK leader \*/* **val** *zkSyncTimeMs* = props.getInt("zookeeper.sync.time.ms", 2000)  
}

可以将props.getInt的时间设置超过30秒

ZkUtils.scala:zookeeper.session.timeout.ms系统默认6000 6s,这个超时zk就资源重新分配，所有数据都不能接收。考虑GC等因素，zookeeper.session.timeout.ms这个值在生产环境中设置为30s.30000 ACK确认

Session.timeout是sync.timeout的若干倍。

connection.timeout

1. Spark Streaming不能接到kafka的数据
2. 数据的重新分配
3. 网络和GC都会有影响

relible模式保存了很多元数据的信息，崩溃后会进行恢复

"zookeeper.connect" -> zkQuorum, "group.id" -> groupId,  
"zookeeper.connection.timeout.ms" -> "10000")

这个timeout的作用，我们的receiver和zookeeper链接的超时时间。这个时间的价值是没什么价值，对于zookeeper是有价值，作为客户端链接，必须保证客户端是连上去了，才能发数据。峰值非常夸张的时候，做fullgc很可能出现问题超时。

connection是链接的时候，相当于是注册的时间，session-time-out是心跳的时间