为什么Kafka接入的数据，返回字段buy\_time并没有先后顺序？

在Kafka中，producer发送的数据会被分片，Kafka只确保每个partition中数据的有序。因此即使原始数据有序，分片后被消费者读取也会是无序的。

统计到达前五的城市

代码如下：

import java.util

import java.util.{Collections, Comparator, Properties, UUID}

import org.apache.flink.api.common.serialization.SimpleStringSchema

import scala.collection.immutable.Range

//import org.apache.flink.streaming.api.scala.\_

import org.apache.flink.streaming.connectors.kafka.FlinkKafkaConsumer010

import org.apache.flink.streaming.api.scala.StreamExecutionEnvironment

import org.apache.flink.streaming.api.scala.function.ProcessAllWindowFunction

import org.apache.flink.streaming.api.windowing.time.Time

import org.apache.flink.streaming.api.windowing.windows.TimeWindow

import org.apache.flink.util.Collector

import org.apache.flink.api.scala.\_

import scala.collection.mutable

object Main4 {

/\*\*

\* 输入的主题名称

\*/

val inputTopic = "mn\_buy\_ticket\_demo2"

/\*\*

\* kafkaT地址

\*/

val bootstrapServers = "bigdata35.depts.bingosoft.net:29035,bigdata36.depts.bingosoft.net:29036,bigdata37.depts.bingosoft.net:29037"

def getCity(string: String): String={

var result=new Array[String](4)

result=string.split(",")

var mid= result(3)

return result(3).split(":")(1)

}

class comparator extends util.Comparator[(String,Int)] {

override def compare(firstPair:(String,Int),secondPair:(String,Int)): Int ={

var firstInt=firstPair.\_2

var secondInt=secondPair.\_2

var fintVal = Integer.valueOf(firstInt)

var sintVal = Integer.valueOf(secondInt)

return -fintVal.compareTo(sintVal)

}

}

def main(args: Array[String]): Unit = {

val env = StreamExecutionEnvironment.getExecutionEnvironment

val kafkaProperties = new Properties()

kafkaProperties.put("bootstrap.servers", bootstrapServers)

kafkaProperties.put("group.id", UUID.randomUUID().toString)

kafkaProperties.put("auto.offset.reset", "earliest")

kafkaProperties.put("key.deserializer", "org.apache.kafka.common.serialization.StringDeserializer")

kafkaProperties.put("value.deserializer", "org.apache.kafka.common.serialization.StringDeserializer")

val kafkaConsumer = new FlinkKafkaConsumer010[String](inputTopic,

new SimpleStringSchema, kafkaProperties)

kafkaConsumer.setCommitOffsetsOnCheckpoints(true)

val inputKafkaStream = env.addSource(kafkaConsumer)

inputKafkaStream.map(x =>(getCity(x),1)).timeWindowAll(Time.seconds(5))

.process(new ProcessAllWindowFunction[(String, Int), mutable.Map[String, Int], TimeWindow] {

override def process(context: Context, elements: Iterable[(String, Int)], out: Collector[mutable.Map[String, Int]]): Unit = {

//定义一个hashMap统计每个城市的到达人数

var cityNumberMap=new util.HashMap[String,Int]()

elements.foreach(kv=>{

if(cityNumberMap.containsKey(kv.\_1)) {

var oldValue = cityNumberMap.get(kv.\_1)

var newValue = oldValue + 1

cityNumberMap.replace(kv.\_1, newValue)

}

else {

cityNumberMap.put(kv.\_1,1)

}

})

//将hashMap的每一个键值对导入一个ArrayList

var pairs=new util.ArrayList[(String,Int)]

var keys=cityNumberMap.keySet().iterator()

while(keys.hasNext){

var key=keys.next()

var value=cityNumberMap.get(key)

pairs.add((key,value))

}

//对arrayList进行排序

Collections.sort(pairs,new comparator())

//将前五个结果存到result

val result = mutable.Map[String, Int]()

var i=0

for(i<-0 to 4){

result.put(pairs.get(i).\_1, pairs.get(i).\_2)

}

out.collect(result)

}

}).print()

env.execute()

}

}

结果如下：

