基于生产设备数据的实时大数据分析

（1）实验环境

Linux Ubuntu 16.04、java 环境、Spark 单机环境、Flume 环境、Kafka 环境、Redis环境、Python（Django、pyecharts）、IDEA、Maven 环境

（2）

数据说明

示例

7361,22,vm-914,20,5,52.0,4.5,3.7,4.6,6.0,9.0,2.2,3.6,9.4,4.5,0.6,2.4，8.9,4.9,2019-09-20 00:21:00

数据以“,”分割， 每列分别表示设备的唯一ID、设备所属虚拟中心ID、虚拟设备ID、处理的物品数量、处理异常数、运行时的温度、网络传输速度、CPU使用率、内存利用率、时间等信息。

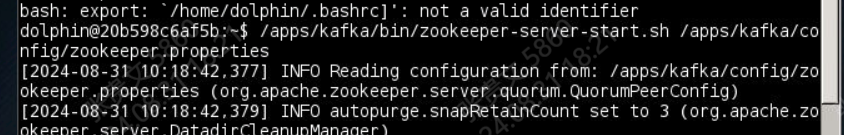
目的：对各个设备处理的物品数、处理异常数、运行时的温度做统计分析并可视化。

（3）

启动Zookeeper服务

这里使用的是Kafka里自带的Zookeeper服务，所以无需另外单独安装Zookeeper。

Zookeeper在本实验中不仅为Kakfa消息队列服务，还用来存储Spark Streaming在消费Kakfa中数据时的偏移量。



（4）启动Kafka服务

双击桌面打开另一个命令行终端，输入下面的命令后再按回车：

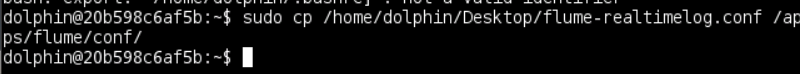
/apps/kafka/bin/kafka-server-start.sh /apps/kafka/config/server.properties



(5)配置Flume

使用Flume之前需要配置，例如需要监控的目录、数据落地到哪里等。

使用命令配置文件flume-realtimelog.conf复制到Flume的配置文件下

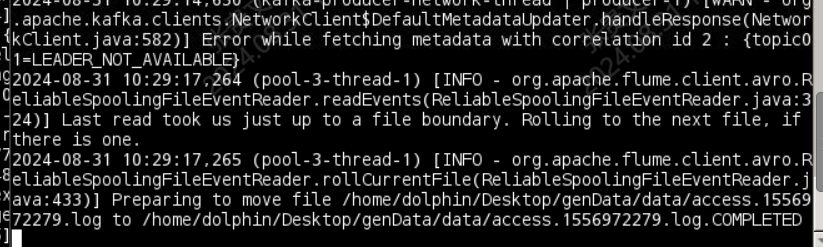


(6)启动Flume

在刚才运行命令的终端中输入下面的命令后再按回车：

/apps/flume/bin/flume-ng agent -c /apps/flume/conf -n a1 -f /apps/flume/conf/flume-realtimelog.conf -Dflume.root.logger=INFO,console

运行后终端上会显示大量日志，表明Flume服务正在运行。



(7) .启动Redis服务

双击桌面打开另一个命令行终端，输入下面的命令后再按回车：

redis-server

运行后终端上会显示大量日志，表明Redis服务正在运行。



(8使用客户端配置Redis

使用redis-cli命令来登陆Redis，双击桌面打开另一个命令行终端，输入下面的命令后再按回车：

redis-cli -h 127.0.0.1

显示如下：

$ redis-cli -h 127.0.0.1

127.0.0.1:6379>

此时已经进入了Redis命令行

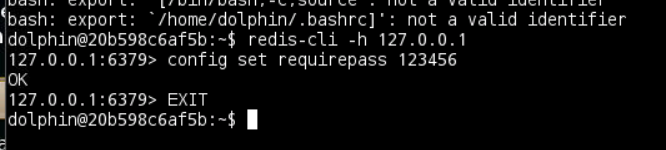
在Redis命令行中运行下面的命令，用于设置密码

config set requirepass 123456

显示如下：

127.0.0.1:6379> config set requirepass 123456

OK

再在Redis命令行中运行 exit 命令，退出Reids命令行。

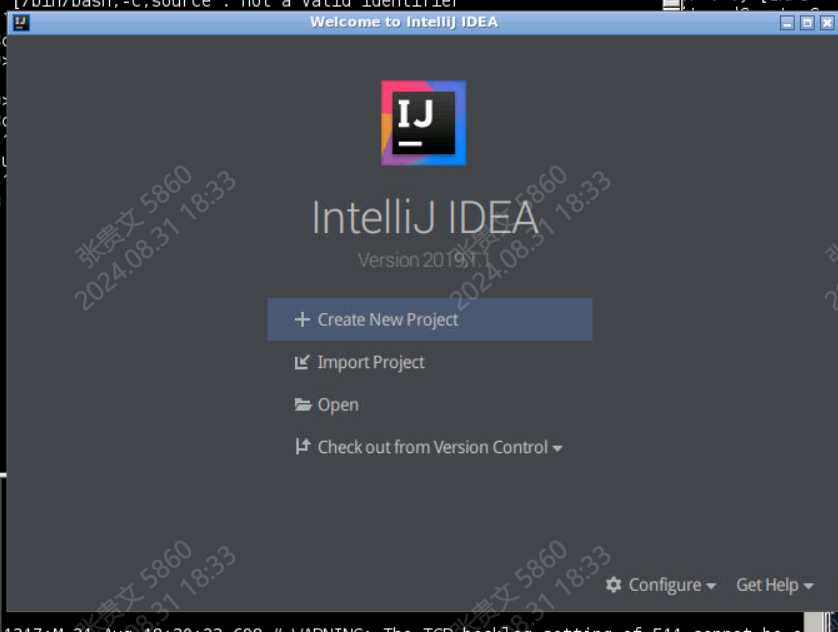
(9) 启动IDEA

在刚才运行命令的终端中输入下面的命令后再按回车：

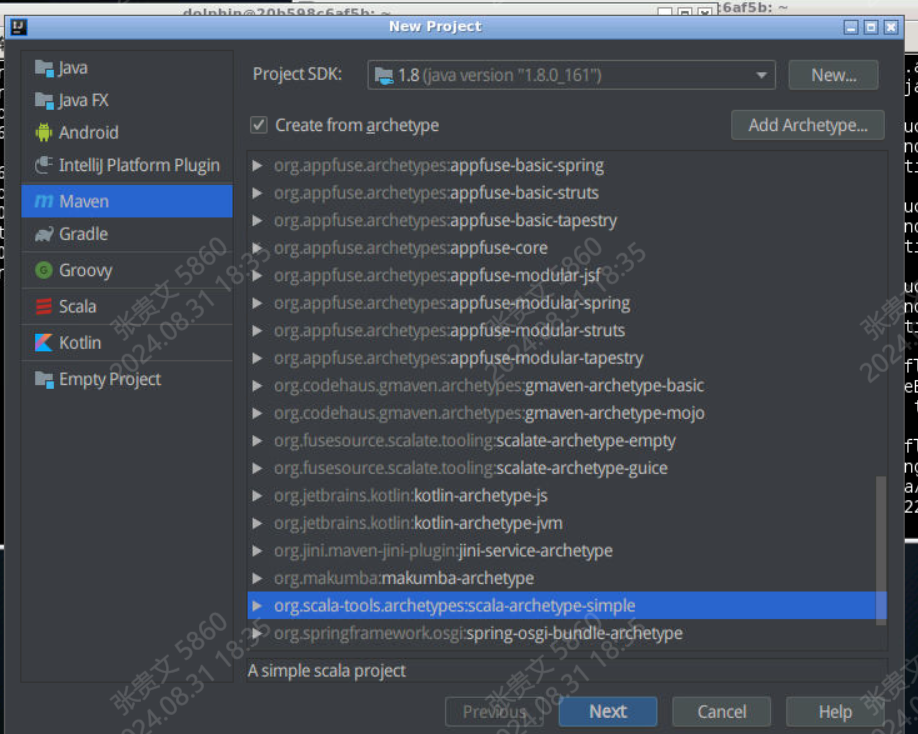
/home/dolphin/idea-IC-191.6707.61/bin/idea.sh

创建Maven项目

点击 Create New Project 按钮，如下图



选择maven项目，再选中 Create from archetype 左边的单选框，再选择下面的 scala-archetype-simple后，点击Next按钮，如图：

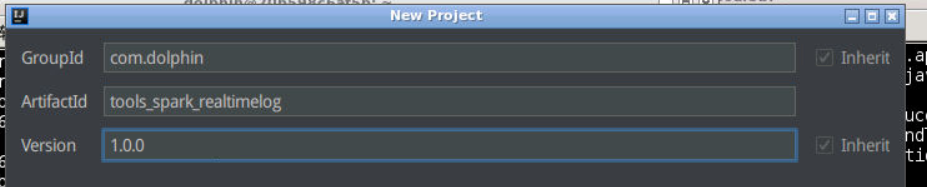


GroupID填写：com.dolphin

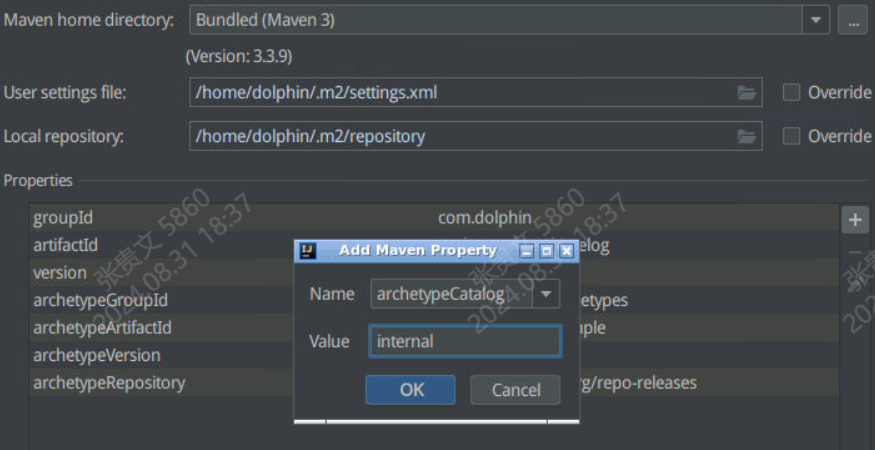
ArtifactId填写： tools\_spark\_realtimelog

Version改为：1.0.0

填写好后点击Next按钮，如图：



点击Properites框的右边的+号，Name填写：archetypeCatalog， Value填写：internal，填写好后点击OK，再点击Next，如图

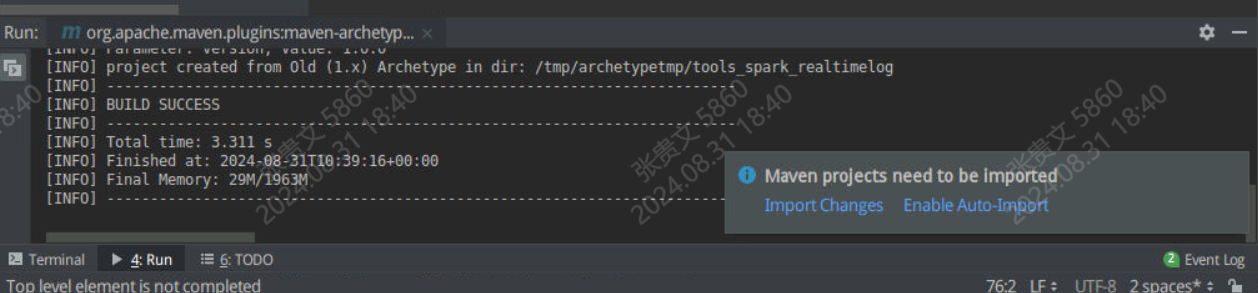


再次点击Next按钮后，再点击Finish后，完成项目创建。

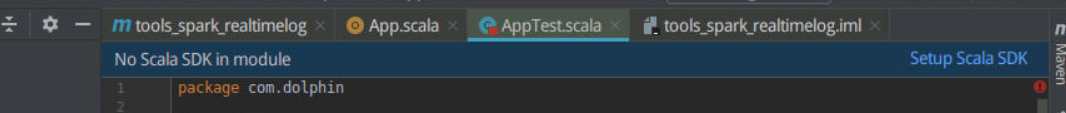
（10）配置Maven的pom.xml文件

配置文件在code文件夹里面的pom.xml文件

此时在点击右下角的 ImportChanges按钮



点击右上角的 Setup Scala SDK 按钮，再点击OK

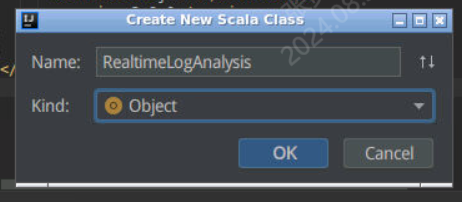


然后删除App文件、AppTest文件、MySpec文件。

（11）

编写Spark代码

点击com.dolphin包，右击选中New，再选中 Scala Class，再弹出的框中Name填写：RealtimeLogAnalysis，Kind更换为Object，再点击OK，如图：

点击com.dolphin包，右击选中New，再选中 Scala Class，再弹出的框中Name填写：CalculateUtils，Kind更换为Object，再点击OK

Spark Streaming代码解读

查看RealtimeLogAnalysis类中的main方法

前53行都是在配置和准备参数等，例如配置Kafka的ip和端口、Zookeeper的ip和端口、指定要消费的topic、设置好Zookeeper的路径用于存储偏移量等。

部分代码

var offsetRanges = Array[OffsetRange]()

kafkaStream.foreachRDD { kafkaRDD =>

  if (!kafkaRDD.isEmpty()) {

    // 只有KafkaRDD可以强转成HasOffsetRanges，并获取偏移量

    offsetRanges = kafkaRDD.asInstanceOf[HasOffsetRanges].offsetRanges

    val lines: RDD[String] = kafkaRDD.map(\_.\_2)

    // 业务逻辑

    // 统计各个设备生产的物品数量

    CalculateUtils.calculateTotalCount(lines)

    // 统计各个设备运行时的平均温度

    CalculateUtils.calculateAverageTemp(lines)

    // 统计各个设备生产物品时出现异常的次数

    CalculateUtils.calculateInvalidCount(lines)

    // 统计每个设备日志的条数

    CalculateUtils.calculateDeviceLogCount(lines)

    for (o <- offsetRanges) {

      val zkPath = s"${topicDirs.consumerOffsetDir}/${o.partition}"

      // 将该partition的offset保存到zookeeper

      ZkUtils.updatePersistentPath(zkClient, zkPath, o.untilOffset.toString)

    }

  }

}

过从Zookeeper中获取上次消费kafka的偏移量，从上次结束的位置开始消费，从而保证不遗漏、不重复消费数据。

通过调用CalculateUtils类中的方法，来统计日志数据中的关键信息。最后将此次消费的偏移量记录到Zookeeper中，避免因故障而导致的偏移量丢失。

.Spark业务代码解读

业务代码都在CalculateUtils类中，该类中有很多方法用于统计不同的指标，下面我们选一个方法进行说明

/\*\*

  \* 统计各个设备生产的物品数量，结果保存在Redis中

  \* @param lines

  \*/

def calculateTotalCount(lines: RDD[String]): Unit = {

  val device01 = lines.filter(l => {l.split(",")(0) == "2400"})

  val device02 = lines.filter(l => {l.split(",")(0) == "7361"})

  val device03 = lines.filter(l => {l.split(",")(0) == "7363"})

  device01.foreachPartition(part => {

    val jedis = getConnection()

    var sum = 0.0

    part.foreach(l => {

      val count = l.split(",")(3).toFloat

      sum = sum + count

    })

    jedis.incrByFloat("DEVICE\_2400", sum)

    jedis.close()

  })

该方法用于统计各个设备生产的物品数量，结果保存在Redis中。其中参数lines是Spark Streaming一次消费的所有行数据。

想通过RDD提供的filter方法过滤出各个设备的数据，使用split函数来获取列数据，累加后存储到Redis中

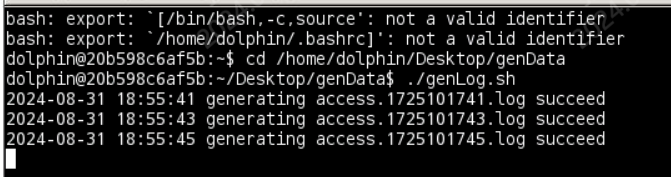
（12）生成数据

双击桌面打开另一个命令行终端，输入下面的命令后再按回车：

cd /home/dolphin/Desktop/genData

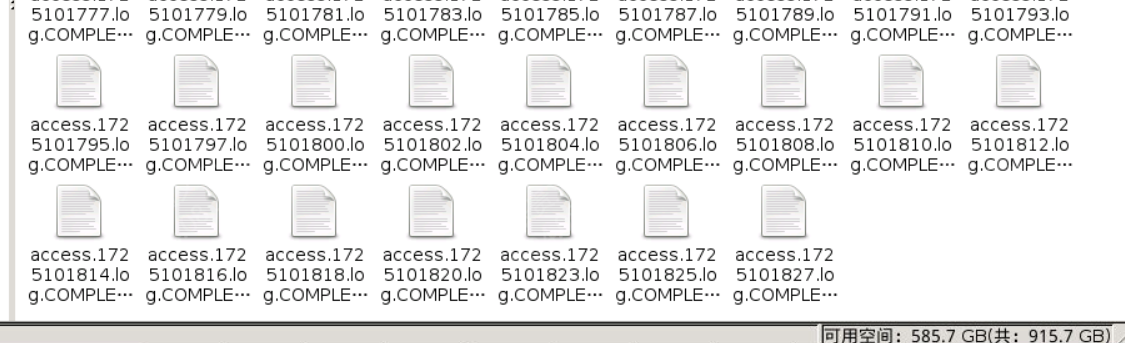
再运行下面的命令：

./genLog.sh



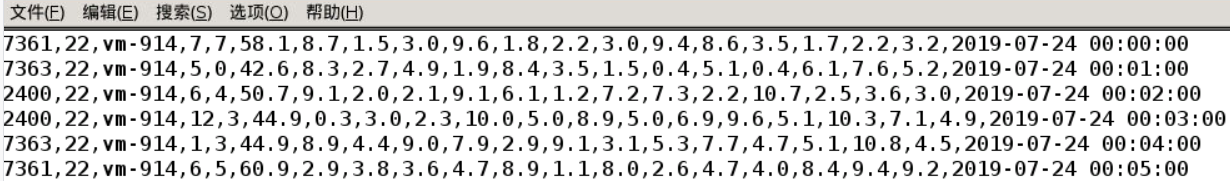
15查看生成的数据

在桌面上找到genData文件夹，再打开data文件夹，会看到很多日志，如下图：



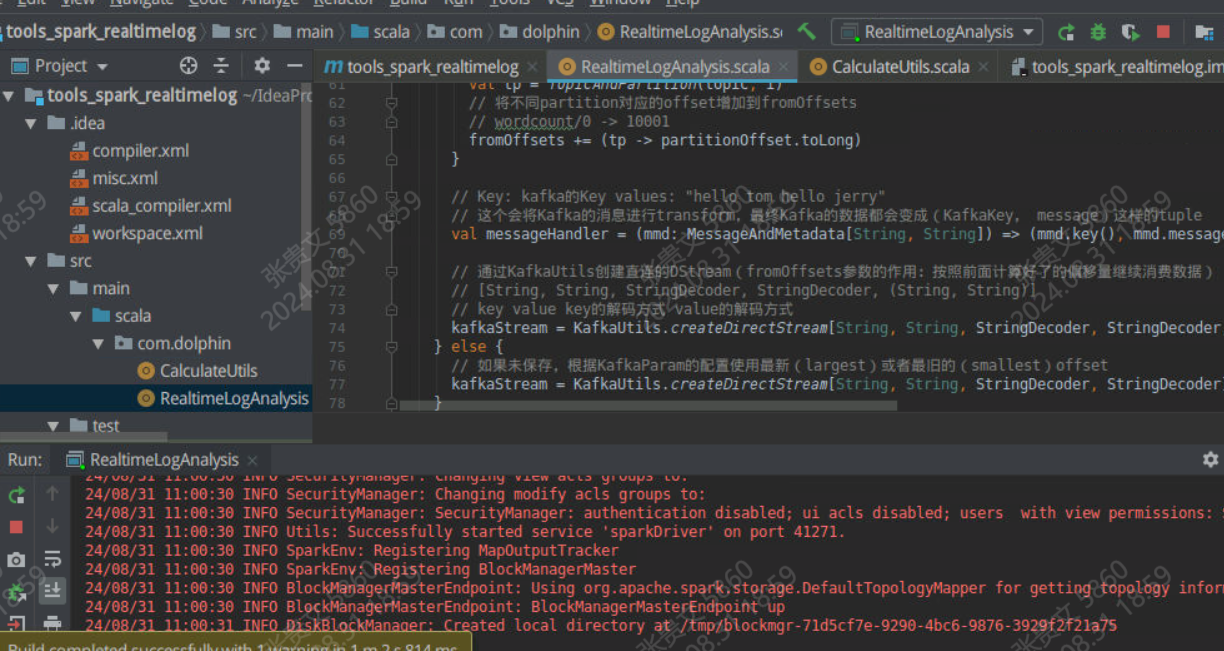
其中文件名称都以COMPLETED后缀结尾，表明此文件以及被Flume迁移到了Kakfa中。

随意右击一个文件，使用编辑器打开，如下图即生产设备生成的数据样式：



15.运行Spark Streaming程序

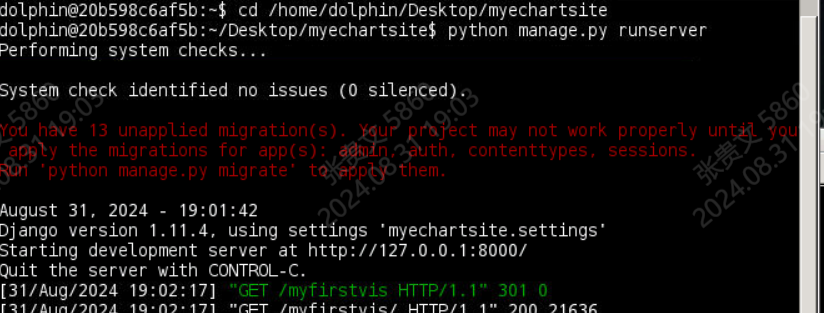
如下图所示：



（16）运行Django服务

cd /home/dolphin/Desktop/myechartsite

python manage.py runserver



查看可视化结果

打开浏览器输入127.0.0.1:8000/myfirstvis

