华中科技大学

大数据处理实验报告

实验四：大数据实时分析

专业班级：CS2005 班

学　　号：U202090063

姓　　名：董玲晶

指导教师：石宣化

报告日期：2022.04.10

计算机科学与技术学院

《大数据处理》课程实验报告

实验地点	南一楼 804	课程名称	大数据处理		
实验题目	大数据实时分析	成绩		指导教师	石宣化
教师评价	☐ 实验过程正确； ☐ 源程序/实验内容提交； ☐ 程序结构/实验步骤合理； ☐ 实验结果正确； ☐ 语法、语义/命令正确； ☐ 报告规范； 其他：				

一、实验目的

1. 了解大数据实时分析的用途
2. 掌握大数据实时分析的基本命令

二、实验内容

1. 实验环境配置
2. Python 脚本生成测试数据（20’）
3. 配置 Kafka（10’）
4. 安装 Flume 客户端（10’）
5. 配置 Flume 采集数据（20’）
6. MySQL 中准备结果表与维度表数据（10’）
7. 使用 DLI 中的 Flink 作业进行数据分析（20’）
8. 资源释放
9. 实验总结 （10’）

三、实验环境

1.软件：PuTTY 远程登录软件、云数据库 RDS 搭载 MySQL 引擎 5.7、DLI 的 SQL 队列和通用队列、Kafka、Flume 客户端、云数据迁移服务 CDM、数据可视化服务 DLV、Hadoop2.8.3、HBase1.3.1、Hive2.3.3，、Tez0.9.1，使用弹性公网 IP 访问 MRS。

2.硬件：使用 MRS1.9.2 分析集群。Master 节点和分析 Core 节点均搭载 4 个 Cortex 虚拟 CPU，16G 内存，以及高 IO/100G 数据盘和系统盘。

四、实验过程或步骤（源程序）

4.1 Python 脚本生成测试任务

4.1.1 打开 Putty，输入前面为 MRS 的 master 节点绑定的公网 IP，点击 open 后，输入用户名和密码登录到 MRS 服务的 master 节点，如图 4.1.a 所示。

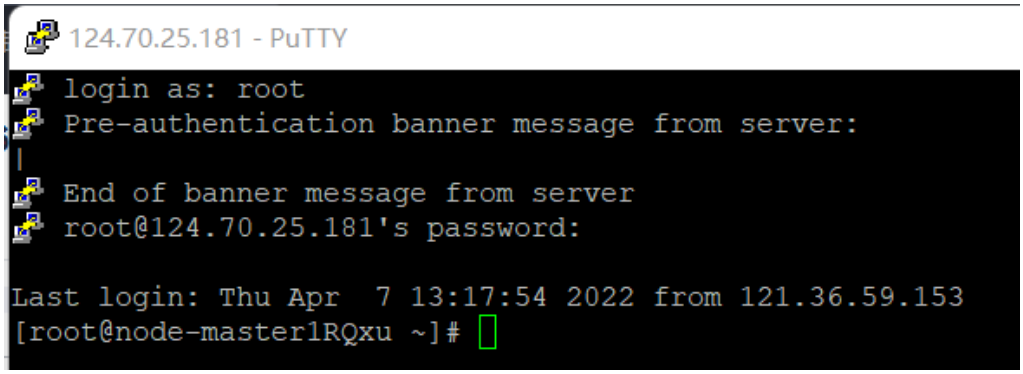

图 4.1.a

4.1.2 进入/opt/client/目录，使用 vi 命令编写 Python 脚本：autodatagen.py，复制所给的脚本代码（因为太长这里省略不写）到 vim 之中，如图 4.1.b 所示。

```
# cd /opt/client/
# vim autodatagen.py
```

```python
import random
import string
import sys
import time
alphabet_upper_list = string.ascii_uppercase
alphabet_lower_list = string.ascii_lowercase
def get_random(instr, length):
    res = random.sample(instr, length)
    result = ''.join(res)
    return result
rowkey_tmp_list = []
def get_random_rowkey():
    import time
    pre_rowkey = ""
    while True:
        num = random.randint(00, 99)
        timestamp = int(time.time())
        pre_rowkey = str(num).zfill(2) + str(timestamp)
        if pre_rowkey not in rowkey_tmp_list:
            rowkey_tmp_list.append(pre_rowkey)
            break
```

图 4.1.b

4.1.3 使用 mkdir 命令在/tmp 下创建目录 flume_spooldir，把 Python 脚本模拟生成的数据放到此目录下，后面 Flume 就监控这个文件下的目录，以读取数据。

```
# mkdir /tmp/flume_spooldir
```

4.1.4 执行 Python 命令，测试生成 100 条数据，再使用 more 命令查看生成的数据。命令如图 4.1.c，生成数据如图 4.1.d 所示。

```
# Python autodatagen.py “/tmp/flume_spooldir/test.txt” 100
# more /tmo/flume_spooldir/test.txt
```

```
[root@node-master1RQxu client]# vim autodatagen.py
[root@node-master1RQxu client]# python autodatagen.py "/tmp/flume_spooldir/t
est.txt" 100
[root@node-master1RQxu client]# more /tmp/flume_spooldir/test.txt
```

图 4.1.c

```
281649311082,Yadru,20,man,230121,940.87,313020,scan,13709376241,iKQUCwXAbF@1
63.com,2019 08 07 n251649311082,Wljuy,53,man,152121,648.65,313014,scan,15290
214657,rcGpxhHgnJ@qq.com,2019 08 04 n221649311082,Ogcxq,56,man,550012,194.89
,313021,scan,15651089362,dXnPhMloWq@gmail.com,2019 08 05 n521649311082,Lqstq
,55,woman,950013,5.85,313020,scan,13497206584,ngfXrMDuhP@huawei.com,2019 08
07 n121649311082,Ksfrp,53,man,230121,312.36,313019,cart,15671238094,cgqoSDHI
Gh@gmail.com,2019 08 04 n511649311082,Xispt,37,woman,230121,43.88,313012,sca
n,158 71904568,FORodTCQme@gmail.com,2019 08 06 n891649311082,Vpdyt,57,man,65
0012,776.25,313014,scan,15335279684,JLAHbcTaVm@qq.com,2019 08 07 n66164931
82,Yqakf,33,man,550012,359.54,313018,pv,15983967540,tKTGoFaOci@126.com,20 19
 08 03 n061649311082,Yaqzl,54,man,550012,733.76,313021,pv,18650429368,FXYfoB
QRkS@huawei.com,2019 08 04 n631649311082,Ueonr,55,woman,950013,146.43,31302
,scan,15978132459,sxBRZjiqmo@126.com,2019 08 02 n971649311082,Wbnqc,44,woman
,480071,294.41,313020,scan,15312589370,ScArJRiQCV@163.com,2019 08 06 n261649
311082,Raskj,51,man,230121,298.88,313019,fav,13306598741,uBPojcvtOa@126.com,
2019 08 07 n671649311082,Fahxg,35,man,250983,324.36,313017,fav,13174896021,z
PHlmMiZfA@gmail.com,2019 08 01 n411649311082,Uhxgi,28,man,250983,414.84,3130
```

图 4.1.d

4.2 配置 Kafka

4.2.1 使用 Putty 登录 MRS 的 master 节点服务器后，首先使用 source 命令进行环境变量的设置使得相关命令可用，如图 4.2.a 所示。

```
# source /opt/client/bigdata_env
```

```
[root@node-master1RQxu ~]# source /opt/client/bigdata_env
```

图 4.2.a

4.2.2 cd 到如下指定目录，执行如下命令创建 topic，如图 4.2.b-图 4.2.c 所示.创建成功后界面中会显示 Created topic “fludesc”.

cd /opt/client/Kafka/kafka/bin

kafka-topics.sh --create --zookeeper 192.168.0.117:2181/kafka -partitions 1 -replication-factor 1 -topic fludesc

```
[root@node-master1RQxu ~]# cd /opt/client/Kafka/kafka/bin
```

图 4.2.b

```
[root@node-master1RQxu bin]# kafka-topics.sh --create --zookeeper 192.168.0.117:
2181/kafka --partitions 1 --replication-factor 1 --topic fludesc
Created topic "fludesc".
```

图 4.2.c

4.2.3 输入如下指令查看 topic 的信息，如图 4.2.d 所示。

kafka-topic.sh -list -zookeeper 192.168.0.117:2181/kafka

```
[root@node-master1RQxu bin]# kafka-topics.sh --list --zookeeper 192.168.0.117:21
81/kafka
__consumer_offsets
fludesc
```

图 4.2.d

4.3 安装 Flume 客户端

4.3.1 进入 mrs_rta 集群页面，点击“前往 manager”，输入用户名和登录密码点击“登录”，进入到 MRS Manager 界面。再 MRS Manager 集群管理页面，点击“服务管理”，点击“Flume”，进入 Flume 服务，点击“下载客户端”按钮，如图 4.3.a 所示。

保存弹窗里的下载路径。

图 4.3.a

4.3.2 使用 PuTTY 登录到 master 节点服务器，进入/tmp/MRS-client 目录，如图 4.3.b 所示。

```
# cd /tmp/MRS-client/
# ll
```

```
[root@node-master1RQxu ~]# cd /tmp/MRS-client/
[root@node-master1RQxu MRS-client]# ll
total 546180
-rw-------. 1 omm wheel 559288320 Apr  7 14:31 MRS_Flume_Client.tar
[root@node-master1RQxu MRS-client]#
```

图 4.3.b

4.3.3 执行以下命令，解压压缩包获取校验文件与客户端配置包，如图 4.3.c 所示。

```
# tar -xvf MRS_Flume_Client.tar
```

```
[root@node-master1RQxu MRS-client]# tar -xvf MRS_Flume_Client.tar
MRS_Flume_ClientConfig.tar.sha256
MRS_Flume_ClientConfig.tar
```

图 4.3.c

4.3.4 执行如下命令，校验文件包。回车后出现 OK 表明文件包校验成功，如图 4.3.d 所示。

```
# sha256sum -c MRS_Flume_ClientConfig.tar.sha256
```

```
[root@node-master1RQxu MRS-client]# sha256sum -c MRS_Flume_ClientConfig.tar.sha2
56
MRS_Flume_ClientConfig.tar: OK
[root@node-master1RQxu MRS-client]#
```

图 4.3.d

4.3.5 解压“MRS_Flume_ClientConfig.tar”文件，然后查看解压后的文件，如图 4.3.e 和 4.3.f 所示。

```
# tar -xvf MRS_Flume_ClientConfig.tar
# ll
```

```
[root@node-master1RQxu MRS-client]# tar -xvf MRS_Flume_ClientConfig.tar
MRS_Flume_ClientConfig/
MRS_Flume_ClientConfig/Flume/
MRS_Flume_ClientConfig/ca.crt
MRS_Flume_ClientConfig/JDK/
```

图 4.3.e

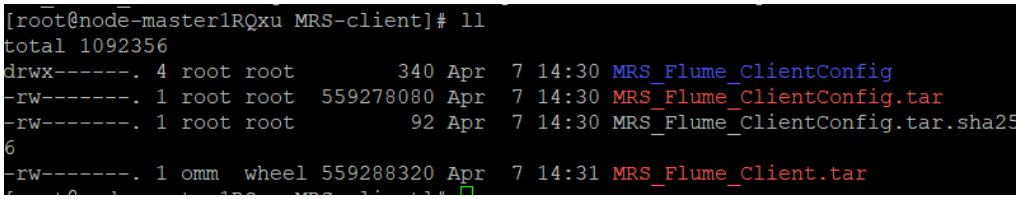

图 4.3.f

4.3.6 安装客户端运行环境到目录“/opt/Flume_env”，当出现“Components client installation is complete.”时说明客户端运行环境安装成功，如图 4.3.g 所示。

```
# sh /tmp/MRS-client/MRS_Flume_ClientConfig/install.sh /opt/Flume_env
```

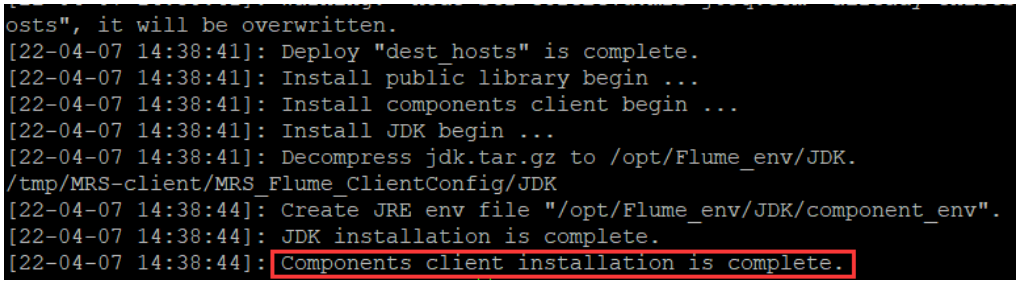

图 4.3.g

4.3.7 执行如下命令配置环境变量，如图 4.3.h 所示。

```
# source /opt/Flume_env/bigdata_env
```

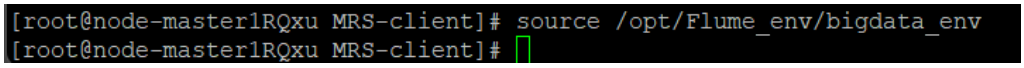

图 4.3.h

4.3.8 执行如下的解压命令解压 Flume 客户端文件，如图 4.3.i 所示。

```
# cd /tmp/MRS-client/MRS_Flume_ClientConfig/Flume
# tar -xvf FusionInsight-Flume-1.6.0.tar.gz
```

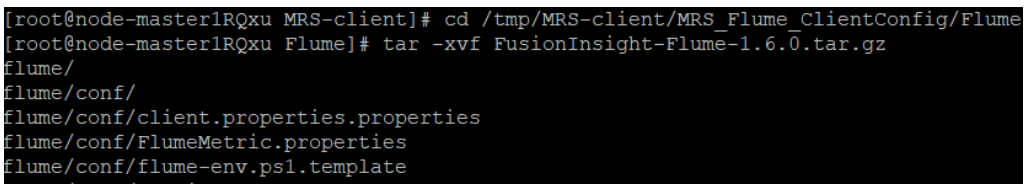

图 4.3.i

4.3.9 安装 Flume 到目录“/opt/FlumeClient”，当显示“install flume client successfully”时表示安装客户端运行环境成功，如图 4.3.j 所示。

```
# sh /tmp/MRS-client/MRS_Flume_ClientConfig/Flume/install.sh -d
```

/opt/FlumeClient

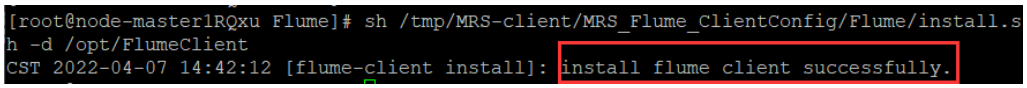

图 4.3.j

4.3.10 执行如下命令对 Flume 的服务进行重启，系统会先显示暂停了原来的进程，再显示启动 Flume 成功，如图 4.3.k 所示。

cd/opt/FlumeClient/fusioninsight-flume-1.6.0

sh bin/flume-manage.sh restart

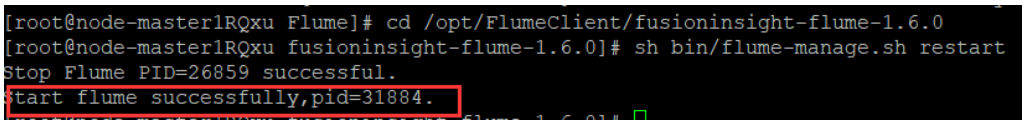

图 4.3.k

4.4 配置 Flume 采集数据

4.4.1 进入 Flume 安装目录，再 conf 目录下编辑文件 properties.properties，在该文件中加入相应的内容（在任务书中，此处省略），如图 4.4.a 所示。

cd /opt/FlumeClient/fusioninsight-flume-1.6.0/

vi conf/properties.properties

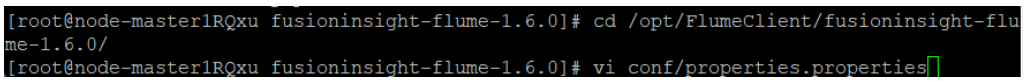

图 4.4.a

4.4.2 使用 PuTTY 登录 master 节点后，执行如下命令，如图 4.4.b 所示。

kafka-console-consumer.sh - topic fludesc - bootstrap-server 192.168.0.40:9092 - new-consumer - consumer.config /opt/client/Kafka/kafka/config/consumer.properties

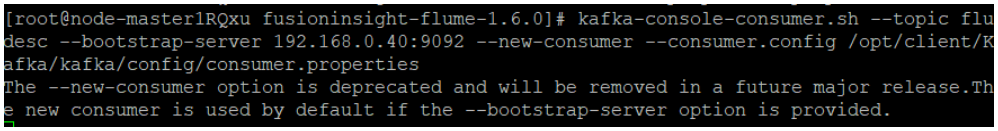

图 4.4.b

4.4.3 重新打开一个新的 PuTTY 会话窗口，在新的窗口中输入用户名和密码登录。进入 python 脚本所在的目录，执行 python 脚本，再生成一份数据。此时查看原窗口，可发现已经消费除了数据，有数据产生，表明此时 Flume 和 Kafka

是打通的，如图 4.4.c 所示。

```
# cd /opt/client/
# python autodatagen.py "/tmp/flume_spooldir/test.txt" 100
```

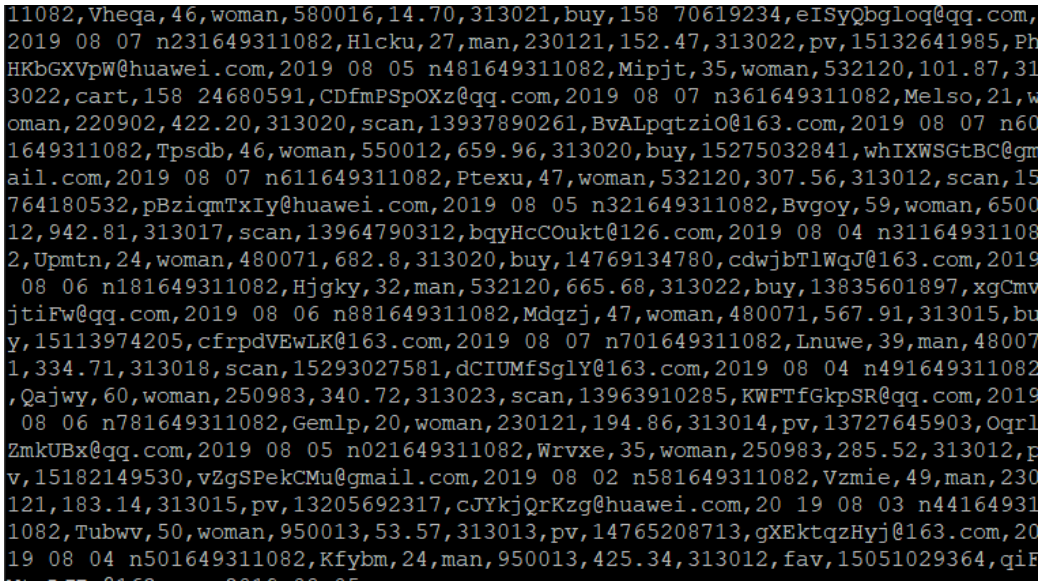

图 4.4.c

4.4.4 测试完毕，在新打开的窗口输入 eixt 关闭窗口，在原窗口输入 Ctrl+c 退出进程。

4.5 MySQL 中准备结果表与维度数据表

4.5.1 在控制台进入到云数据库 RDS 实例管理界面，点击实例后面的“登录”按钮。输入用户名 root 和密码，勾选“记住密码”，开启“定时采集”和“SQL 执行记录”， 然后点击“测试连接”，成功后点击“登录”按钮。

4.5.2 点击“新建数据库”， 输入名称“rds_desc”，字符集选择 utf8，点击“确定”，如图 4.5.a 所示。

图 4.5.a

4.5.3 点击数据库后面的“SQL 查询”，进入到 SQL 执行界面，清楚查询器中的原有内容，复制相应的 SQL 语句粘贴到 SQL 查询中，如图 4.5.b-4.5.e 所示。

```
DROP TABLE IF EXISTS `desc_goods_info`;
CREATE TABLE `desc_goods_info` (
 `goods_no` varchar(30) NOT NULL,
 `goods_name` varchar(30) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

图 4.5.b

```
INSERT INTO `desc_goods_info` VALUES ('220902', '杭州丝绸');
INSERT INTO `desc_goods_info` VALUES ('430031', '西湖龙井');
INSERT INTO `desc_goods_info` VALUES ('550012', '西湖莼菜');
INSERT INTO `desc_goods_info` VALUES ('650012', '张小泉剪刀');
INSERT INTO `desc_goods_info` VALUES ('532120', '塘栖枇杷');
INSERT INTO `desc_goods_info` VALUES ('230121', '临安山核桃');
INSERT INTO `desc_goods_info` VALUES ('250983', '西湖藕粉');
INSERT INTO `desc_goods_info` VALUES ('480071', '千岛湖鱼干');
INSERT INTO `desc_goods_info` VALUES ('580016', '天尊贡芽');
INSERT INTO `desc_goods_info` VALUES ('950013', '叫花童鸡');
INSERT INTO `desc_goods_info` VALUES ('152121', '火腿蚕豆');
INSERT INTO `desc_goods_info` VALUES ('230121', '杭州百鸟朝凤');
```

图 4.5.c

```
DROP TABLE IF EXISTS `desc_store_info`;
CREATE TABLE `desc_store_info`(
 `store_id` varchar(50) NOT NULL,
`store_name` varchar(50) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET =utf8;
```

图 4.5.d

```
INSERT INTO `desc_store_info` VALUES ('313012', '莫干山店');
INSERT INTO `desc_store_info` VALUES ('313013', '定安路店');
INSERT INTO `desc_store_info` VALUES ('313014', '西湖银泰店');
INSERT INTO `desc_store_info` VALUES ('313015', '天目山店');
INSERT INTO `desc_store_info` VALUES ('313016', '凤起路店');
INSERT INTO `desc_store_info` VALUES ('313017', '南山路店');
INSERT INTO `desc_store_info` VALUES ('313018', '西溪湿地店');
INSERT INTO `desc_store_info` VALUES ('313019', '传媒学院店');
INSERT INTO `desc_store_info` VALUES ('313020', '西湖断桥店');
INSERT INTO `desc_store_info` VALUES ('313021', '保淑塔店');
INSERT INTO `desc_store_info` VALUES ('313022', '南宋御街店');
INSERT INTO `desc_store_info` VALUES ('313 023', '河坊街店');
```

图 4.5.e

4.5.4 点击“执行 SQL”执行上面的语句，执行成功后可以在下面看到执行消息，如图 4.5.f 和 4.5.g 所示。

```
----------------开始执行----------------

【拆分SQL完成】：将执行SQL语句数量：（28条）

【执行SQL：(1)】
DROP TABLE IF EXISTS `desc_goods_info`
执行成功，耗时：[18ms.]
```

图 4.5.f

```
【执行SQL：(25)】
INSERT INTO `desc_store_info` VALUES ('313020', '西湖断桥店')
执行成功，当前返回：[1]行，耗时：[11ms.]

【执行SQL：(26)】
INSERT INTO `desc_store_info` VALUES ('313021', '保淑塔店')
执行成功，当前返回：[1]行，耗时：[12ms.]

【执行SQL：(27)】
INSERT INTO `desc_store_info` VALUES ('313022', '南宋御街店')
执行成功，当前返回：[1]行，耗时：[11ms.]

【执行SQL：(28)】
INSERT INTO `desc_store_info` VALUES ('313 023', '河坊街店')
执行成功，当前返回：[1]行，耗时：[12ms.]
```

图 4.5.g

4.5.5 清楚 SQL 窗口中原有的代码，粘贴对应的代码并执行，执行成功后刷新按钮可以看到已创建的表，如图 4.5.h 所示。

图 4.5.h

4.5.6 进入数据湖探索服务（DLI）的控制台，点击左侧“作业管理”菜单，选择“Flink 作业”，然后点击右上角的“创建作业”。 类型选择“Flink SQL”，名称输入“desc_order_count”（可以自定义），模板、标签默认，点击“确定”。

4.5.7 编辑 Flink 作业的 SQL 脚本，复制对应的脚本到编辑框中，编辑完后点击“语义校验”，校验无错误则进行下一步骤操作，如图 4.5.i 所示。

图 4.5.i

4.5.8 测试 DLI 与 Kafka 网络是否连通，输入 kafka_bootstrap_servers 地址，测试连通性；同理测试 mysql 端口连通性，如图 4.5.j 和 4.5.k 所示。

图 4.5.j

图 4.5.k

4.5.9 进入 Flink 作业，选择“运行参数”，设置 CU 数量为 2，选择所属队列“queue_flink”， 点击右上角的“启动”。 在启动 Flink 作业页面点击右下角的“立即启动”， 回到 Flink 作业界面，状态变为“提交中”。 点击作业管理中的作业名称 desc_order_count 可以进入作业详情页，当作业变为“运行中”时进行下一步骤操作。

4.5.10 回到 Flink 作业管理界面，点击“作业监控”按钮，进入作业监控页面后，通过指标图表可以看到数据正常处理，如图 4.5.1 所示。

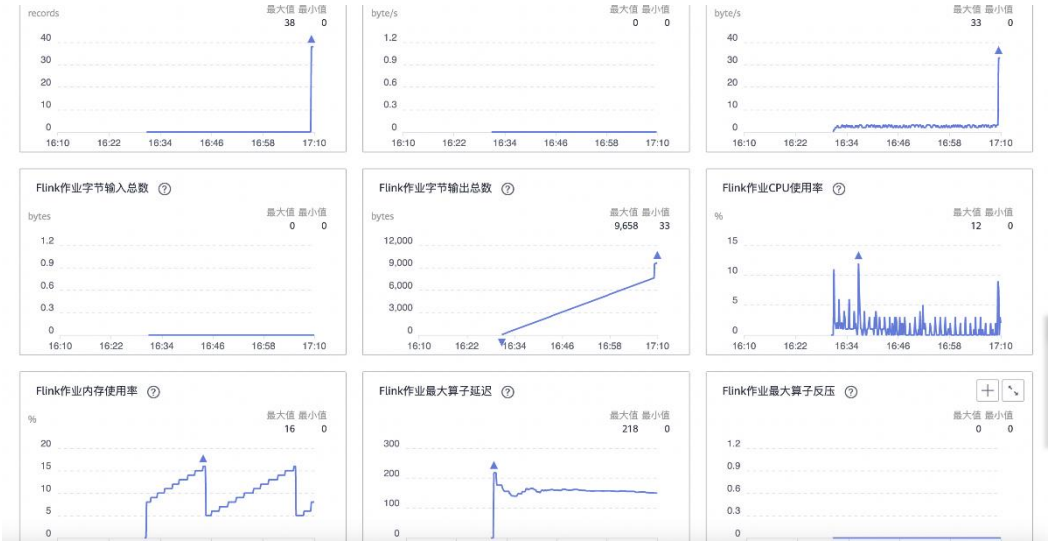

图 4.5.1

4.5.11 登录 MySQL 后点击数据库的名称或后面的“库管理”进入到库管理界面，可以看到结果表中有数据进来，如图 4.5.m 所示。

图 4.5.m

4.5.12 让生成测试数据的 Python 脚本每隔 10 秒钟运行一次，在可视化页面上可以看到统计数据在不断的变化，代码如图 4.5.n 和 4.5.o 所示，数据可视化如图 4.5.p 所示。

```
* * * * * python autodatagen.py "/tmp/flume_spooldir/test.txt" 100
* * * * * sleep 10; python autodatagen.py "/tmp/flume_spooldir/test.txt" 100
* * * * * sleep 20; python autodatagen.py "/tmp/flume_spooldir/test.txt" 100
* * * * * sleep 30; python autodatagen.py "/tmp/flume_spooldir/test.txt" 100
* * * * * sleep 40; python autodatagen.py "/tmp/flume_spooldir/test.txt" 100
* * * * * sleep 50; python autodatagen.py "/tmp/flume_spooldir/test.txt" 100
```

图 4.5.n

```
[root@node-master1punl client]# vim every.cron
[root@node-master1punl client]# crontab every.cron
[root@node-master1punl client]#
```

图 4.5.o

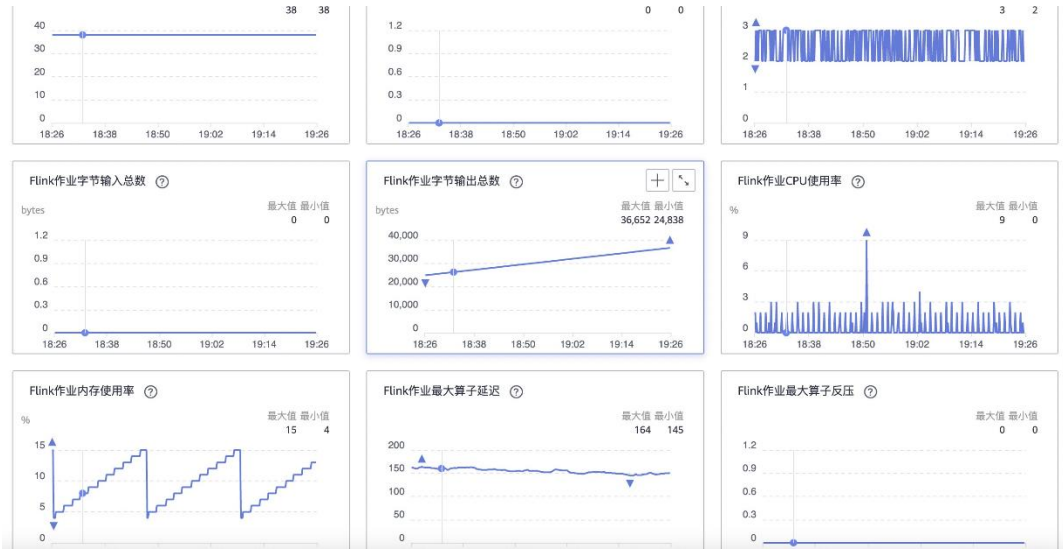

图 4.5.p

五、出现的问题与解决方案

1. 在配置 Kafka，创建 topic 时，一直创建 topic 失败，一开始是提示我 kafka 不存在，发现是路径不对，cd 到指定路径之下后，依然报错，提示参数缺失。后来在网上查了资料，“-xx yy”的形式表示前面是参数的 key，后面是对应 key 的 value，所以语句中某些地方应该要用空格隔开而不是连在一起，否则就会报参数缺失的错误。

2. Flume 和 Kafka 不连通，新开一个 PuTTY 窗口没法在原窗口消费出数据。发现是前面 Flume 客户配置有问题，重新从配置环境变量开始配置，重启后，问题解决。

六、实验总结

本次实验内容很多，特别考验人的耐心，虽然花了很多时间和精力，但我收获颇多。

首先，我认识并初次体验和使用了 PuTTY 这个远程登录工具，在 PuTTY 中远程管理 Linux，复习和进一步巩固 Linux 命令行的使用。其次，我通过动手自己配置了 Kafka，了解了 Kafka 的相关术语，如 broker、topic、partition、producer、consumer 这些参数与概念，接着自己通过 python 模拟生成消费数据后，在 Kafka 上对生成的消费数据进行处理，感受 Kafka 对消费者在网站中的动作流数据的处理。接着，我也体验了 Flume 的使用，通过 Flume 客户端对数据进行采集和局部的处理，并打通 Flume 和 Kafka，将两者结合使用，发挥各自所长。最后的 Flink 作业中，我不仅复习和巩固了 SQL 语句，最重要的是体验了 Flink 处理框架，动手进行了 Flink 作业，对并对数据进行可视化。

我认为这节课最大的收获是我尝试和体验了各种曾经在书上看到的系统或框架的使用，感觉很新奇且很有趣；同时本次实验也让我发现了华为云平台的强大之处，学习到了更多的使用平台，教会了我如何更好地利用现有的资源。