

# 阿里云大学

## 毕业设计论文

数据可视化项目的案例分析、处理和展现

学 生 姓 名	张 程
专 业 名 称	大 数 据 开 发 专 业
指 导 教 师	陈 俊

2019 年 3 月 10 日

## 摘要

本文通过以 dx 通信公司某区域一天的流量使用日志作为原始数据进行分析。通过使用阿里云 MaxCompute 对数据进行提取，清洗，转换，并且进行数据的业务处理。具体表现为，DataWorks 的 sql 表建立，包括了原始表，清洗表，数据字段总表，以及四个业务表。最后将四个业务表通过 QuickBI 进行数据可视化处理,得到本文的四个业务图表。

本文通过这个案例，展示了数据可视化项目的案例分析、处理和展现的过程。

**关键词：**数据可视化 ， MaxCompute， QuickBI

## 目录

摘要.....	1
第一章 案例分析 .....	3
1.1 案例分析的意义 .....	3
1.2 DX 业务案例分析.....	3
1.2.1 业务分析 .....	3
1.2.2 数据分析 .....	3
第二章 数据的清洗处理 .....	5
2.1 实验源具体数据 .....	5
2.1.1 用到的应用字段信息详细释义 .....	5
2.2 使用 MaxCompute 处理数据 .....	7
2.2.1 建立 sql 表.....	7
2.2.2 清洗数据 .....	8
第三章 业务表的建立 .....	11
3.1 应用大类受欢迎程度表 .....	11
3.2 区域各网站的流量使用统计表 .....	12
3.3 区域各小区使用流量数量表 .....	13
3.4 某小区上网 APP 流量花费表.....	14
第四章 Quick BI 数据可视化.....	16
4.1 创建数据集 .....	16
4.2 创建数据可视化图表 .....	17
4.2.1 某区域 app 类型流量使用统计图 .....	17
4.2.2 某区域网站具体流量使用统计图 .....	17
4.2.2 某区域重点小区统计 .....	18
第五章 总结及展望 .....	19
参考文献.....	20

# 第一章 案例分析

## 1.1 案例分析的意义

数据可视化的过程中，案例分析是基础。案例分析是对具体要处理的具体业务进行分析，主要包括业务层面和数据支持两方面。业务层面的层面包括业务的面对对象，分析对象，分析的目的。数据层面的分析包含，有用字段的提取、转换、整合。

案例分析的作用还在于，对可视化结果的草拟，选取图表的种类，图表的维度和度量都在这一步确定。这样，也为之后数据的处理指定了方向。这个方向并不是最终的结果，在数据处理过程中，如果有更佳的选择，也可以对草拟的结果进行修改。

## 1.2 DX 业务案例分析

### 1.2.1 业务分析

本案例选取某通信公司对于某区域的用户流量使用情况进行统计。统计的目的主要包含两个方面，一个是针对企业的 App 的流量分析，一个是针对用户的小区流量使用分析。针对企业的 App 的流量使用分析，能够让公司针对受欢迎 App 在使用时，进行信号优化。针对小区流量使用分析能够筛选出重点客户群体。

### 1.2.2 数据分析

实验源数据，为某通信公司一天的用户流量使用日志。现在以，其中一行为例，说明原始数据中的字段含义。总共包含了 77 个字段，有些字段数据为空，实验字段以|分割。

下面是日志中的某一条数据：

```
533||11|93287887015245963|6|||1|100.82.254.88|100.82.98.100|2152|
2152|13849|147855076|||103|1409649427963|1409649428488|1|15|999||0
|10.83.124.18||60914|0|137.175.9.211||80|734|329|4|2|0|0|0|0|221|29|0|0|20
|221|12600|1260|1|0|1|3|6|200|221|221|255|559955.com|/tu/31322.JPG|55
9955.com|Mozilla/5.0 (Linux; Android 4.3; zh-cn; SAMSUNG-SM-
G7108V_TD Release/02.15.2014 Browser/AppleWebKit537.36
```

Build/JSS15J) AppleWebKit/537.36 (KHTML, like Gecko) Version/1.5  
MobileSafari/537.36||http://www.701111.com/||0|0|0|0||3|0|525|0|0|1:734/  
329

日志以“|”符号分割。实验所用到的信息有：

16 Cell ID byte UE 所在小区的 ECI

22 App Type byte 应用大类

23 App Sub-type byte 应用小类

根据集团定义的识别规则识别出来的小类，参见《中国移动数据流量 DPI 识别能力规范》。集团未定义的各厂家根据自己的 DPI 进行识别

26 USER\_IP byte 终端用户的 IPv4 地址

28 User Port byte 用户的四层端口号

30 App Server IP byte 访问服务器的 IPv4 地址

32 App Server Port byte 访问的服务器的端口

58 HOST char 访问域名

19 ProcudureStartTime long 请求起始时间

20 ProcudureEndTime long 请求结束时间

18 App Type Code byte 请求响应码，业务中只关注是否等于 103。

如果 App Type=103，表示成功发起了一次 Http 请求。

33 UL Data byte 上行流量

34 DL Data byte 下行流量

39 RetranUL byte 上行 TCP 重传报文数

40 RetranDL byte 下行 TCP 重传报文数

54 HTTP/WAP 事务状态 byte HTTP/WAP2.0 层的响应码

其中维度方面：小区信息从 Cell ID 获取，用户的具体信息通过 User\_IP 和 port 来确定，APP Type 则表示流量使用的 APP 种类，APP 的服务商通过 Host 域名。流量的使用度量则通过上行流量和下行流量来确定。

## 第二章 数据的清洗处理

### 2.1 实验源具体数据

实验源数据的格式如第一章所示。样本的案例为某天的日志。所以可以直接通过 sql 导入。由于本次的案例着重于数据可视化的分析，所以这块可以建立 MaxCompute 建立数据中心，然后定时上传数据，可以做到定时分析。在本次案例中，仅对此次数据进行处理，来展现数据可视化的过程。

#### 2.1.1 用到的应用字段信息详细释义

22 App Type byte 应用大类：

序号 业务类型 业务说明

- 1 即时通信 互联网消息即时收发业务,如：QQ、飞信等
- 2 阅读 向用户提供在线或离线阅读服务的业务，如：手机阅读、熊猫阅读等
- 3 微博 微博业务，如：移动微博、新浪微博等
- 4 导航 提供浏览、查询、导航等功能的电子地图类业务，如：谷歌地图、高德导航等
- 5 视频 向用户提供音视频内容的直播、分享和下载服务的网站和应用（不包括传统意义上基于 P2P 技术的视频业务），如：优酷、手机电视等
- 6 音乐 提供音乐在线欣赏和下载服务的网站和应用，如：咪咕音乐、QQ 音乐等
- 7 应用商店 提供应用程序、音乐、图书等内容浏览、下载及购买服务的业务，如：Mobile Market、AppStore 等
- 8 游戏 基于客户端或者网页的游戏业务：QQ 游戏、开心农场等
- 9 支付 电子商务类业务，如：手机支付、支付宝、网银等
- 10 动漫 提供动漫在线欣赏和下载服务的网站和应用，如：手机动漫、爱看动漫等
- 11 邮箱 电子邮箱业务，如：139 邮箱、QQ 邮箱等

- 12 P2P 业务 基于 P2P 技术的资源共享业务，包括下载和视  
频两部分，前者如：迅雷、eMule 等，后者如：迅雷看看、  
PPLive 等
- 13 VoIP 业务 互联网语音通信业务，如：Skype、Uucall 等
- 14 彩信 彩信业务
- 15 浏览下载 基于 HTTP、WAP、FTP 等的普通浏览和下载  
业务
- 16 财经 金融资讯、股票证券类业务，如：手机商界、大智  
慧等
- 17 安全杀毒 提供网络安全服务的应用，如：360 安全卫  
士、麦咖啡等；以及网络恶意流量，如：病毒、攻击等
- 18 其他业务

## 23 App Sub-type byte 应用小类

序号 子业务名称 即时通信

- 1 飞聊
- 2 飞信
- 3 Gtalk
- 4 MSN
- 5 QQ
- 6 TM
- 7 阿里旺旺
- 8 米聊
- 9 微信
- 10 人人桌面
- 11 AOL AIM
- 12 Gadu\_Gadu
- 13 go 聊
- 14 ICQ
- 15 IMVU
- 16 Lava-Lava
- 17 NetChat

- 18 Paltalk
- 19 PowWow
- 20 TeamSpeak
- 21 Trillian
- 22 VZOchat
- 23 Xfire
- 24 百度 Hi
- 25 都秀
- 26 陌陌
- 27 天翼 Live
- 28 翼聊
- 29 网易泡泡
- 30 新浪 UC
- 31 新浪 UT
- 32 雅虎通

## 2.2 使用 MaxCompute 处理数据

### 2.2.1 建立 sql 表

```
create table dx_data (a1 string,a2 string,a3 string,a4 string,a5 string,a6 string,a7
string,a8 string,a9 string,a10 string,a11 string,a12 string,a13 string,a14 string,a15 string,a16
string,a17 string,a18 string,a19 string,a20 string,a21 string,a22 string,a23 string,a24
string,a25 string,a26 string,a27 string,a28 string,a29 string,a30 string,a31 string,a32
string,a33 string,a34 string,a35 string,a36 string,a37 string,a38 string,a39 string,a40
string,a41 string,a42 string,a43 string,a44 string,a45 string,a46 string,a47 string,a48
string,a49 string,a50 string,a51 string,a52 string,a53 string,a54 string,a55 string,a56
string,a57 string,a58 string,a59 string,a60 string,a61 string,a62 string,a63 string,a64
string,a65 string,a66 string,a67 string,a68 string,a69 string,a70 string,a71 string,a72
string,a73 string,a74 string,a75 string,a76 string,a77 string)
```

如下图：



```

5  -- *****
6
7  create table dx_data (a1 string,a2 string,a3 string,a4 string,
8      a5 string,a6 string,a7 string,a8 string,a9 string,
9      a10 string,a11 string,a12 string,a13 string,a14 string,a15 string,a16 string,a17 string,
10     a18 string,a19 string,a20 string,a21 string,a22 string,a23 string,a24 string,a25 string,
11     a26 string,a27 string,a28 string,a29 string,a30 string,a31 string,a32 string,a33 string,
12     a34 string,a35 string,a36 string,a37 string,a38 string,a39 string,a40 string,a41 string,
13     a42 string,a43 string,a44 string,a45 string,a46 string,a47 string,a48 string,a49 string,
14     a50 string,a51 string,a52 string,a53 string,a54 string,a55 string,a56 string,a57 string,
15     a58 string,a59 string,a60 string,a61 string,a62 string,a63 string,a64 string,a65 string,
16     a66 string,a67 string,a68 string,a69 string,a70 string,a71 string,a72 string,a73 string,
17     a74 string,a75 string,a76 string,a77 string)
18

```

运行日志

```

2019-03-07 15:06:15 get jobId:20190307070614830gqgtov21
ID = 20190307070614830gqgtov21
{
2019-03-07 15:06:15 INFO =====
2019-03-07 15:06:15 INFO Exit code of the Shell command 0
2019-03-07 15:06:15 INFO --- Invocation of Shell command completed ---
2019-03-07 15:06:15 INFO Shell run successfully!
2019-03-07 15:06:15 INFO Current task status: FINISH
2019-03-07 15:06:15 INFO Cost time is: 2.485s
/home/admin/alisa/tasknode/taskinfo//20190307/datastudio/15/06/07/2bit2grj0scv7dm711hd9da1/T3_0062042915.log-END-EOF

```

## 2.2.2 清洗数据

从原始数据中挑选业务所需的 23 个字段，建立新表。

```
create table data_clear(appType bigint,appSubtype bigint,userIp
string,userPort bigint,appServerIP string,appServerPort bigint,host
string,cellid string,appTypeCodebigint,interruptType String,transStatus
bigint,trafficUL bigint,trafficDL bigint,retranUL bigint,retranDL
bigint,procudureStartTime bigint,procudureEndTime bigint)
```

```

4  --author:135/1808/41
5  --create time:2019-03-07 15:25:12
6  -- *****
7
8  create table data_clear(appType bigint,appSubtype bigint,
9      userIp string,userPort bigint,appServerIP string,appServerPort bigint,host string,cellid string,
10     appTypeCode bigint,interruptType String,transStatus bigint,trafficUL bigint,trafficDL bigint,
11     retranUL bigint,retranDL bigint,procudureStartTime bigint,procudureEndTime bigint)
12

```

运行日志

```

2019-03-07 15:26:07 get jobId:20190307072606300g9r0ussa
ID = 20190307072606300g9r0ussa
OK

```

insert overwrite table data\_clear select a23, a24, a27, a29, a31, a33, a59, a17, a19, a68, a55, a34, a35, a40, a41, a20, a21 from dx\_data;



```

1  --@extra_input=
2  --odps sql
3  --
4  --author:13571808741
5  --create time:2019-03-07 15:25:12
6  --
7
8  insert overwrite table data_clear select
9      a23, a24, a27, a29, a31, a33, a59, a17, a19, a68, a55, a34, a35, a40, a41, a20, a21 from dx_data;
10

```

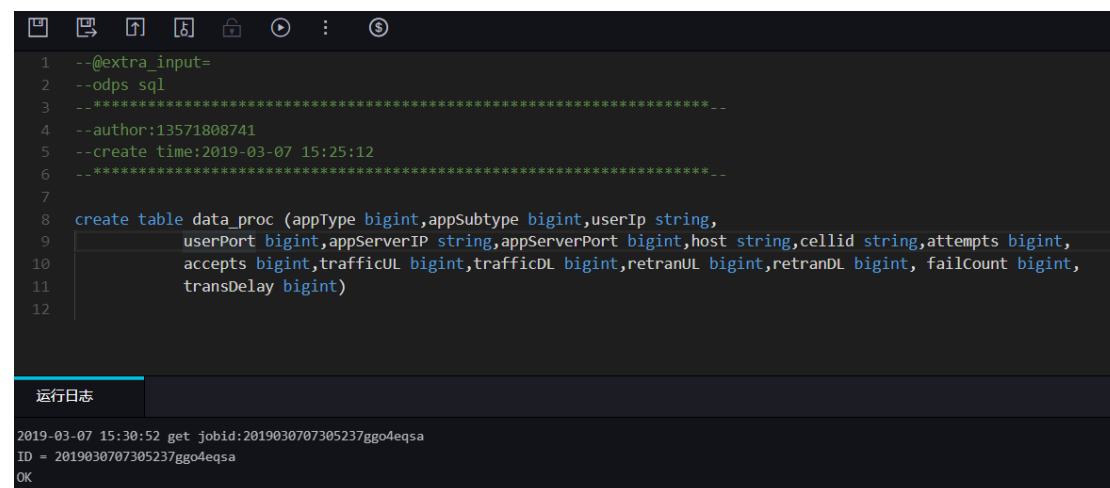
运行日志

output records:  
TableSink1: 19998 (min: 19998, max: 19998, avg: 19998)

OK

再进行业务的逻辑处理，建立 data\_proc 表：

create table data\_proc (appType bigint,appSubtype bigint,userIp string,userPort bigint,appServerIP string,appServerPort bigint,host string,cellid string,attempts bigint, accepts bigint,trafficUL bigint,trafficDL bigint,retranUL bigint,retranDL bigint, failCount bigint, transDelay bigint)



```

1  --@extra_input=
2  --odps sql
3  --
4  --author:13571808741
5  --create time:2019-03-07 15:25:12
6  --
7
8  create table data_proc (appType bigint,appSubtype bigint,userIp string,
9      userPort bigint,appServerIP string,appServerPort bigint,host string,cellid string,attempts bigint,
10     accepts bigint,trafficUL bigint,trafficDL bigint,retranUL bigint,retranDL bigint, failCount bigint,
11     transDelay bigint)
12

```

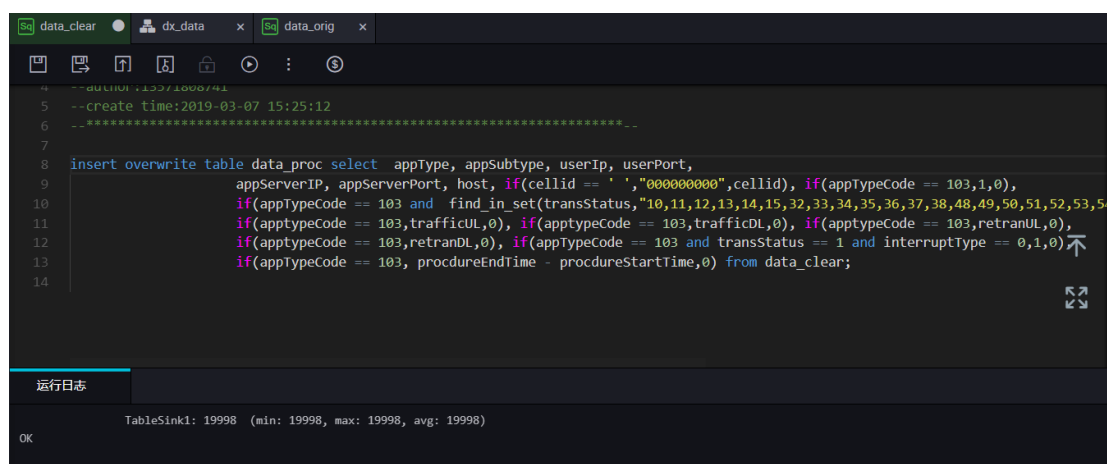
运行日志

2019-03-07 15:30:52 get jobId:2019030707305237ggo4eqsa  
ID = 2019030707305237ggo4eqsa  
OK

业务逻辑处理，插入数据到总表中：

insert overwrite table dataproc select appType, appSubtype, userIp, userPort, appServerIP, appServerPort, host, if(cellid == '','0000000000',cellid), if(appTypeCode == 103,1,0),if(appTypeCode == 103 and

```
find_in_set(transStatus,"10,11,12,13,14,15,32,33,34,35,36,37,38,48,49,50
,51,52,53,54,55,199,200,201,202,203,204,205,206,302,304,306")!=0 and
interruptType == 0,1,0),if(apptypeCode == 103,trafficUL,0),
if(apptypeCode == 103,trafficDL,0), if(apptypeCode == 103,retranUL,0),
if(apptypeCode == 103,retranDL,0), if(appTypeCode == 103 and
transStatus == 1 and interruptType == 0,1,0),if(appTypeCode == 103,
procedureEndTime - procedureStartTime,0) from dataclear;
```



The screenshot shows a SQL IDE interface with a query execution window. The query is an SQL INSERT statement that inserts data into a table named 'data\_proc'. The query includes a complex WHERE clause using the 'find\_in\_set' function to filter data based on 'transStatus' and 'interruptType'. The execution window shows the result of the query, indicating that 19998 rows were inserted into the 'TableSink1' table. The window also displays the minimum, maximum, and average values for the 'TableSink1' table.

```
--dataclear:133/1000/44
5 --create time:2019-03-07 15:25:12
6
7
8 insert overwrite table data_proc select appType, appSubtype, userIp, userPort,
9 appServerIP, appServerPort, host, if(cellid == ' ', "00000000", cellid), if(apptypeCode == 103,1,0),
10 if(apptypeCode == 103 and find_in_set(transStatus,"10,11,12,13,14,15,32,33,34,35,36,37,38,48,49,50,51,52,53,5
11 if(apptypeCode == 103,trafficUL,0), if(apptypeCode == 103,trafficDL,0), if(apptypeCode == 103,retranUL,0),
12 if(apptypeCode == 103,retranDL,0), if(apptypeCode == 103 and transStatus == 1 and interruptType == 0,1,0),
13 if(apptypeCode == 103, procedureEndTime - procedureStartTime,0) from data_clear;
14
```

运行日志

TableSink1: 19998 (min: 19998, max: 19998, avg: 19998)

OK

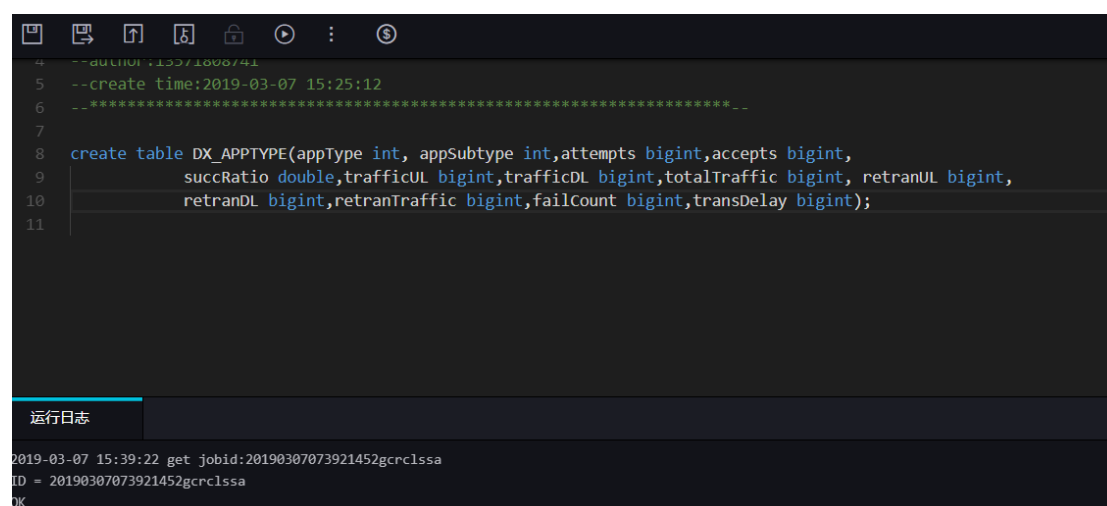
## 第三章 业务表的建立

上一章将源数据进行处理。本章，将从已经处理之后的数据表中，选取合适的字段，进行分析。分析数据，得到自己想要的结果。并且在业务上，建立相符合的表，以辅助实现数据可视化。

### 3.1 应用大类受欢迎程度表

首先，先创建表。然后从上一章的数据中获取，相应的数据信息。下方受欢迎 APP 分析，先创建表，然后处理数据。

```
create table DX_APPTYPE(appType int, appSubtype int, attempts  
bigint, accepts bigint, succRatio double, trafficUL bigint, trafficDL  
bigint, totalTraffic bigint, retranUL bigint, retranDL bigint, retranTraffic  
bigint, failCount bigint, transDelay bigint);
```



The screenshot shows a SQL execution window with a dark theme. The top toolbar contains icons for file operations, execution, and settings. The main text area displays the SQL command to create the DX\_APPTYPE table. Below the code, there is a '运行日志' (Execution Log) tab. The log shows the execution time as 2019-03-07 15:39:22 and the job ID as 20190307073921452gcrc1ssa. The status is 'OK'.

```
--duid:15571008741  
--create time:2019-03-07 15:25:12  
--  
create table DX_APPTYPE(appType int, appSubtype int, attempts bigint, accepts bigint,  
succRatio double, trafficUL bigint, trafficDL bigint, totalTraffic bigint, retranUL bigint,  
retranDL bigint, retranTraffic bigint, failCount bigint, transDelay bigint);
```

运行日志  
2019-03-07 15:39:22 get jobid:20190307073921452gcrc1ssa  
ID = 20190307073921452gcrc1ssa  
OK

```
insert overwrite table DX_APPTYPE select apptype, appsubtype,  
sum(attempts), sum(accepts), round(sum(accepts)/sum(attempts),2) ,  
sum(trafficUL), sum(trafficDL), sum(trafficUL)+sum(trafficDL),  
sum(retranUL), sum(retranDL), sum(retranUL)+sum(retranDL),  
sum(failCount) , sum(transDelay)from data_proc group by apptype,  
appsubtype;
```

```

4  --author:13571808741
5  --create time:2019-03-07 15:25:12
6  --*****
7
8  insert overwrite table DX_APPTYPE select apptype, appsubtype, sum(attempts), sum(accepts),
9      round(sum(accepts)/sum(attempts),2) , sum(trafficUL), sum(trafficDL), sum(trafficUL)+sum(trafficDL),
10     sum(retranUL), sum(retranDL), sum(retranUL)+sum(retranDL), sum(failcount) , sum(transDelay)
11     from data_proc group by apptype, appsubtype;
12

```

运行日志

output records:  
TableSink1: 226 (min: 226, max: 226, avg: 226)

OK

## 3.2 区域各网站的流量使用统计表

操作步骤同 2.1 表:

create table DX\_HOST(host string, appServerIP string, attempts  
bigint, accepts bigint, succRatio double ,trafficUL bigint, trafficDL  
bigint, totalTraffic bigint, retranUL bigint, retranDL bigint, retranTraffic  
bigint, failCount bigint, transDelay bigint);

```

1  --@extra_input=
2  --odps sql
3  --*****
4  --author:13571808741
5  --create time:2019-03-07 15:25:12
6  --*****
7
8  create table DX_HOST(host string, appServerIP string, attempts bigint,
9      accepts bigint, succRatio double ,trafficUL bigint, trafficDL bigint, totalTraffic bigint,
10     retranUL bigint, retranDL bigint, retranTraffic bigint, failCount bigint, transDelay bigint);
11

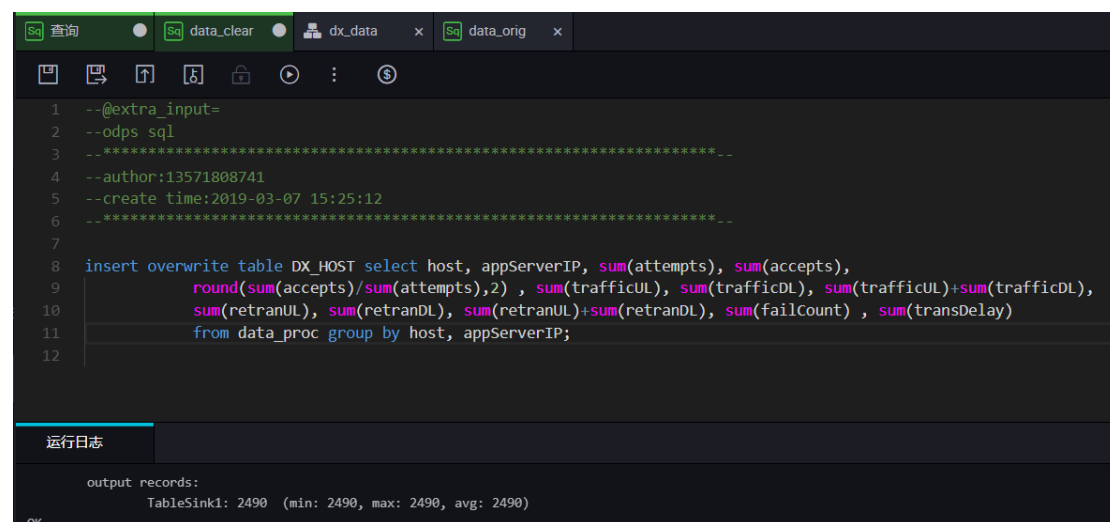
```

运行日志

2019-03-07 15:43:36 get jobid:20190307074335874g91z66pr2  
ID = 20190307074335874g91z66pr2  
OK

insert overwrite table DX\_HOST select host, appServerIP,  
sum(attempts), sum(accepts), round(sum(accepts)/sum(attempts),2) ,  
sum(trafficUL), sum(trafficDL), sum(trafficUL)+sum(trafficDL),  
sum(retranUL), sum(retranDL), sum(retranUL)+sum(retranDL),  
sum(failCount) , sum(transDelay) from data\_proc group by host,

appServerIP;



```

1  --@extra_input=
2  --odps sql
3  --_*****_
4  --author:13571808741
5  --create time:2019-03-07 15:25:12
6  --_*****_
7
8  insert overwrite table DX_HOST select host, appServerIP, sum(attempts), sum(accepts),
9      round(sum(accepts)/sum(attempts),2) , sum(trafficUL), sum(trafficDL), sum(trafficUL)+sum(trafficDL),
10     sum(retranUL), sum(retranDL), sum(retranUL)+sum(retranDL), sum(failCount) , sum(transDelay)
11     from data_proc group by host, appServerIP;
12

```

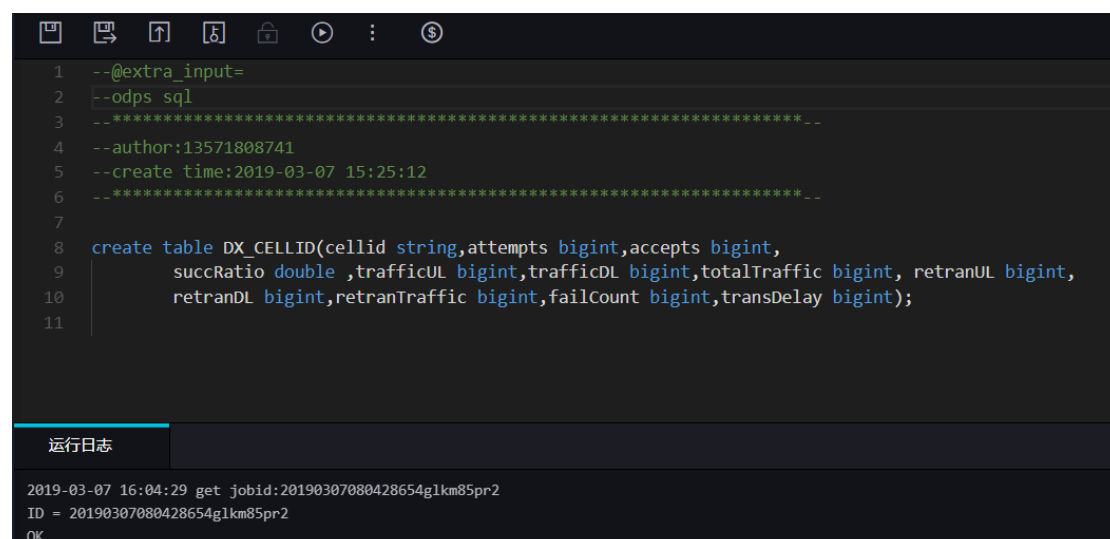
运行日志

output records:  
TableSink1: 2490 (min: 2490, max: 2490, avg: 2490)

OK

### 3.3 区域各小区使用流量数量表

create table DX\_CELLID(cellid string,attempts bigint,accepts bigint, succRatio double ,trafficUL bigint,trafficDL bigint,totalTraffic bigint, retranUL bigint, retranDL bigint,retranTraffic bigint,failCount bigint,transDelay bigint);



```

1  --@extra_input=
2  --odps sql
3  --_*****_
4  --author:13571808741
5  --create time:2019-03-07 15:25:12
6  --_*****_
7
8  create table DX_CELLID(cellid string,attempts bigint,accepts bigint,
9      succRatio double ,trafficUL bigint,trafficDL bigint,totalTraffic bigint, retranUL bigint,
10     retranDL bigint,retranTraffic bigint,failCount bigint,transDelay bigint);
11

```

运行日志

2019-03-07 16:04:29 get jobid:20190307080428654g1km85pr2  
ID = 20190307080428654g1km85pr2  
OK

insert overwrite table DX\_CELLID select cellid, sum(attempts), sum(accepts), round(sum(accepts)/sum(attempts),2) , sum(trafficUL), sum(trafficDL), sum(trafficUL)+sum(trafficDL), sum(retranUL),

sum(retranDL), sum(retranUL)+sum(retranDL), sum(failCount) ,  
sum(transDelay)from data\_proc group by cellid;

```

1  --@extra_input=
2  --odps sql
3  --
4  --author:13571808741
5  --create time:2019-03-07 15:25:12
6  --
7
8  insert overwrite table DX_CELLID select cellid, sum(attempts), sum(accepts),
9      round(sum(accepts)/sum(attempts),2) , sum(trafficUL), sum(trafficDL), sum(trafficUL)+sum(trafficDL),
10     sum(retranUL), sum(retranDL), sum(retranUL)+sum(retranDL), sum(failCount) , sum(transDelay)
11     from data_proc group by cellid;
12

```

运行日志

output records:  
TableSink1: 1904 (min: 1904, max: 1904, avg: 1904)

OK

### 3.4 某小区上网 APP 流量花费表

create table DX\_HOST\_INTR(cellid string, host string, attempts  
bigint,accepts bigint, succRatio double ,trafficUL bigint,trafficDL  
bigint,totalTraffic bigint, retranUL bigint, retranDL bigint,retranTraffic  
bigint,failCount bigint,transDelay bigint);

```

1  --@extra_input=
2  --odps sql
3  --
4  --author:13571808741
5  --create time:2019-03-07 15:25:12
6  --
7
8  create table DX_HOST_INTR(cellid string, host string, attempts bigint,accepts bigint, succRatio double ,
9      trafficUL bigint,trafficDL bigint,totalTraffic bigint, retranUL bigint, retranDL bigint,
10     retranTraffic bigint,failCount bigint,transDelay bigint);
11

```

运行日志

2019-03-07 16:09:01 get jobid:20190307080900869gzc4eqsa  
ID = 20190307080900869gzc4eqsa  
OK

insert overwrite table DX\_HOST\_INTR select cellid, host,  
sum(attempts), sum(accepts), round(sum(accepts)/sum(attempts),2) ,  
sum(trafficUL), sum(trafficDL), sum(trafficUL)+sum(trafficDL),  
sum(retranUL), sum(retranDL), sum(retranUL)+sum(retranDL),  
sum(failCount) , sum(transDelay)from data\_proc group by cellid, host;

```
1  --@extra_input=
2  --odps sql
3  --_*****_
4  --author:13571808741
5  --create time:2019-03-07 15:25:12
6  --_*****_
7
8  insert overwrite table DX_HOST_INTR select cellid, host, sum(attempts), sum(accepts), round(sum(accepts)/sum(attempts),2) ,
9      sum(trafficUL), sum(trafficDL), sum(trafficUL)+sum(trafficDL), sum(retranUL), sum(retranDL),
10     sum(retranUL)+sum(retranDL), sum(failCount) , sum(transDelay
11     )from data_proc group by cellid, host;
12
```

运行日志

output records:  
TableSink1: 5976 (min: 5976, max: 5976, avg: 5976)

OK



## 第四章 Quick BI 数据可视化

这一章会对之前处理的表格数据进行可视化处理。主要体现在两个方面，小区上网和 APP 耗流量两个方面的四个表，分别选取饼图和柱状图来表示。

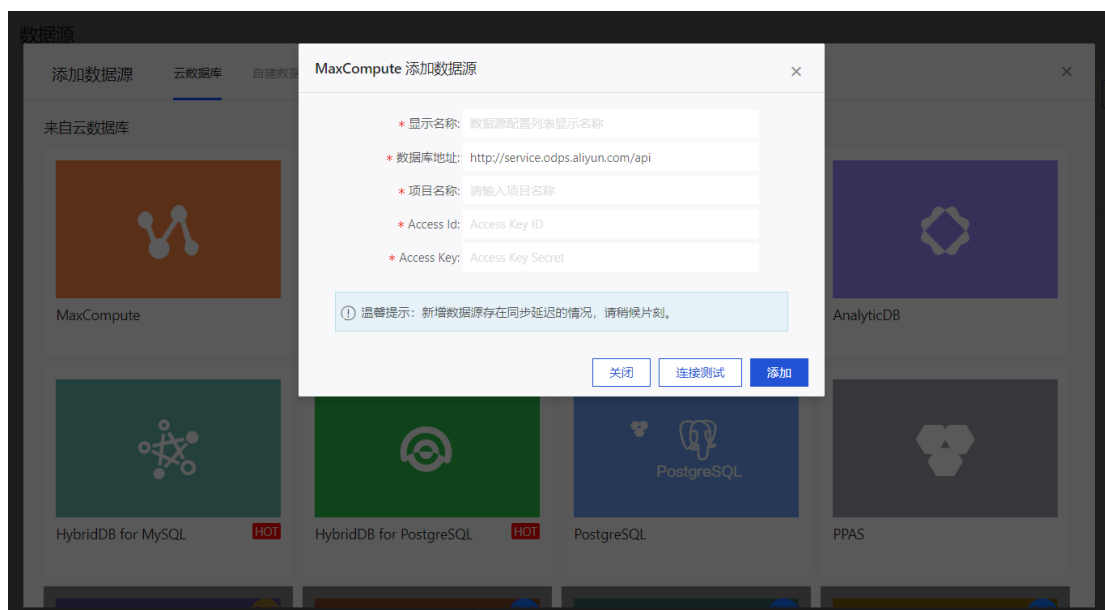
### 4.1 创建数据集

添加数据源，并创建数据集。这一步是，将 MaxCompute 的数据源，导入到 QuickBI 中。



名称	创建者	修改人/修改时间	数据源	操作
dx_host_intr dx_host_intr	13571808741	13571808741 2019/3/7 18:32:55	dx_data MaxCompute	🔍 ⚙️ 🗑️
dx_host dx_host	13571808741	13571808741 2019/3/7 18:20:55	dx_data MaxCompute	🔍 ⚙️ 🗑️
dx_cellid dx_cellid	13571808741	13571808741 2019/3/7 18:03:30	dx_data MaxCompute	🔍 ⚙️ 🗑️
dx_apptype dx_apptype	13571808741	13571808741 2019/3/7 16:49:17	dx_data MaxCompute	🔍 ⚙️ 🗑️

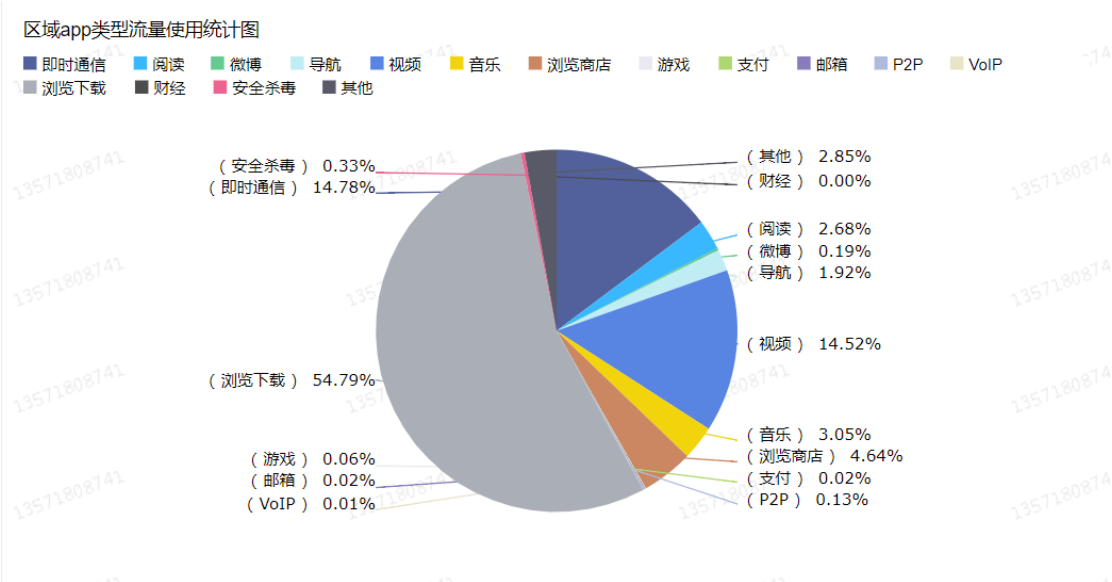
然后创建数据集。创建的数据集应该和导入的原数据对应。



## 4.2 创建数据可视化图表

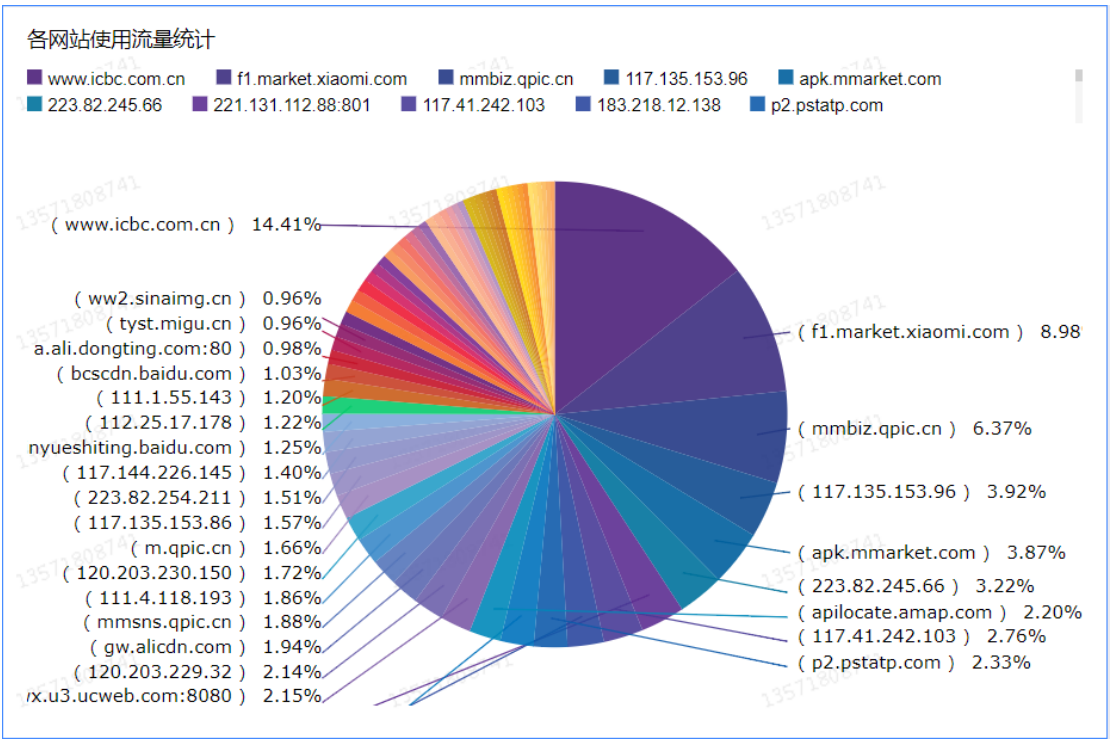
### 4.2.1 某区域 app 类型流量使用统计图

根据 DX\_APPTYPE 表，选取 APPTYPE 大类最为维度，流量总数作为度量。得到的可视化图如下图：



### 4.2.2 某区域网站具体流量使用统计图

由于没有获得具体域名与 IP 地址的对应，所以图没有显示具体的网站域名。



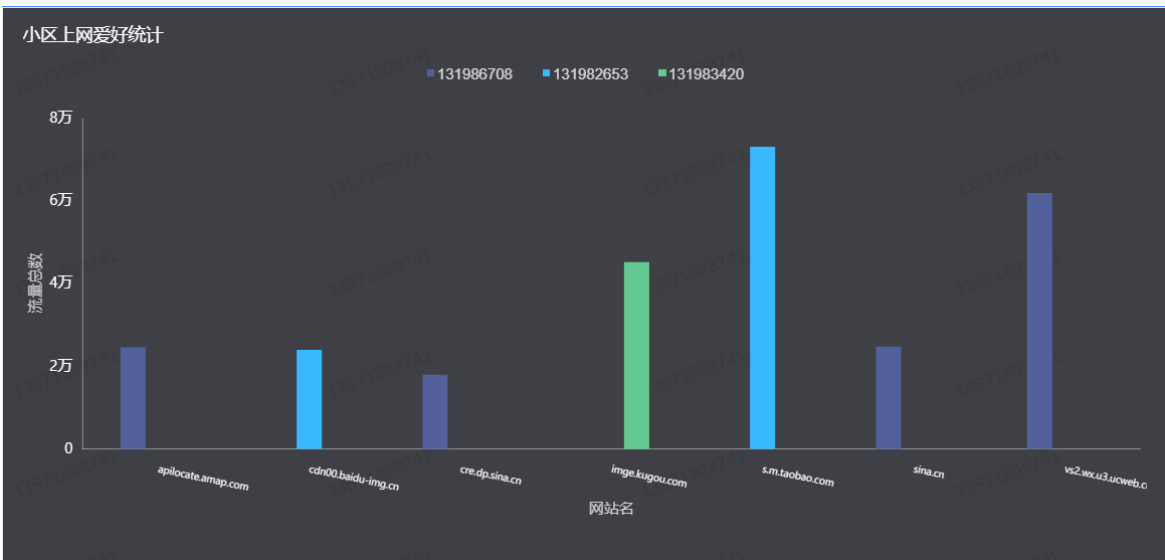
### 4.2.2 某区域重点小区统计

该区域小区数量非常庞大，分析数据之后，设置过滤器，将使用流量大于 200 万的小区设置重点小区。于是得到柱状图，如下图：



### 4.2.3 小区上网偏好图

选择了上方三个重点小区，然后网站的使用分析。得出下图：



## 第五章 总结及展望

本文根据 DX 的某一天流量使用日志进行数据分析，进行数据可视化展示。展现一天中，某区域的小区使用流量情况以及该区域访问 App 类型的比例。但是这仅仅是某一天的数据，并不具有普遍性。所以，对未来的展望有以下几点：

1. 建立数据仓库，每天按时收集日志数据；
2. 建表时分区，以时间为分区字段；
3. 每周，或每月对数据进行总的分析；
4. 可以根据数据的变化制作折线图，得到变化趋势；
5. 可以建立机器学习模型，预测未来流量使用情况。

以上是本文的全部内容。在此特别对陈俊老师和陈婉盈班助表示感谢。

## 参考文献

1. 阿里云帮助文档. Quick BI: 网站日志分析[EB/OL].  
[https://help.aliyun.com/document\\_detail/43097.html?spm=a2c4g.11186631.6.708.79ad595eKzaQrT](https://help.aliyun.com/document_detail/43097.html?spm=a2c4g.11186631.6.708.79ad595eKzaQrT)
2. 阿里云帮助文档. Quick BI: 如何之最多图关联的仪表盘[EB/OL].  
[https://help.aliyun.com/document\\_detail/43097.html?spm=a2c4g.11186631.6.708.79ad595eKzaQrT](https://help.aliyun.com/document_detail/43097.html?spm=a2c4g.11186631.6.708.79ad595eKzaQrT)
3. 阿里云帮助文档. MaxCompute: 使用 Java SDK 运行安全命令最佳实践[EB/OL].  
[https://help.aliyun.com/document\\_detail/27808.html?spm=a2c4g.11174283.6.564.69ea590ekLWWhe](https://help.aliyun.com/document_detail/27808.html?spm=a2c4g.11174283.6.564.69ea590ekLWWhe)
4. 阿里云帮助文档. MaxCompute: Hadoop 数据迁移 MaxCompute 最佳实践[EB/OL].  
[https://help.aliyun.com/document\\_detail/90315.html?spm=a2c4g.11186623.6.895.501d405e7MSvZM](https://help.aliyun.com/document_detail/90315.html?spm=a2c4g.11186623.6.895.501d405e7MSvZM)
5. ChinaTelecom 数据分析项目  
<https://github.com/LittleLawson/ChinaTelecom>