我试了试用 SQL查 Linux日志，好用到飞起

架构师精选 今天

以下文章来源于程序员内点事，作者小富

程序员内点事

专注于系统架构实战，面试干货分享，Java小白的入门布道，程序员内点事这都有

> **来自公众号：程序员内点事**

大家好，我是小富~

最近发现点好玩的工具，迫不及待的想跟大家分享一下。

大家平时都怎么查`Linux`日志呢？像我平时会用`tail`、`head`、`cat`、`sed`、`more`、`less`这些经典系统命令，或者`awk`这类三方数据过滤工具，配合起来查询效率很高。但在使用过程中有一点让我比较头疼，那就是命令参数规则太多了，记的人脑壳疼。

那查日志有没有一种通用的方式，比如用SQL查询，毕竟这是程序员都比较熟悉的表达式。

今天分享的工具**q**，就实现了以写SQL的方式来查询、统计文本内容，一起看看这货到底有什么神奇之处。

搭个环境

q是一个命令行工具，允许我们在**任意文件**或者**查询结果**，比如可以在 `ps -ef` 查询进程命令的结果集上，直接执行SQL语句查询。

宗旨就是文本即数据库表，额~，当然这句话是我自己理解的，哈哈哈

它将普通文件或者结果集当作数据库表，几乎支持所有的SQL结构，如 `WHERE`、`GROUP BY`、`JOINS` 等，**支持自动列名和列类型检测**，**支持跨文件连接查询**，这两个后边详细介绍，支持多种编码。

安装比较简单，在 `Linux CentOS` 环境，只要如下三步搞定，`Windows` 环境更是只需安装个 `exe` 就可以用了。

```
wget https://github.com/harelba/q/releases/download/1.7.1/q-text-as-data-1.7.1-1.noarch.rpm #下载版本

sudo rpm -ivh q-text-as-data-1.7.1-1.noarch.rpm # 安装

q --version  #查看安装版本
```

> 官方文档：https://harelba.github.io/q

语法

q支持所有 `SQLite` SQL语法，标准命令行格式 `q + 参数命令 + "SQL"`

```
q <命令> "<SQL>"
```

我要查询 `myfile.log` 文件的内容，直接 `q "SELECT * FROM myfile.log"` 。

```
q "SELECT * FROM myfile.log"
```

q不附加参数使用是完全没有问题的，但利用参数会让显示结果更加美观，所以这里简单了解一下，它的参数分为 2种。

`input` 输入命令：指的是对要查询的文件或结果集进行操作，比如：`-H` 命令，表示输入的数据包含标题行。

```
q -H "SELECT * FROM myfile.log"
```

在这种情况下，将自动检测列名，并可在查询语句中使用。如果未提供此选项，则列将自动命名为cX，以c1起始以此类推。

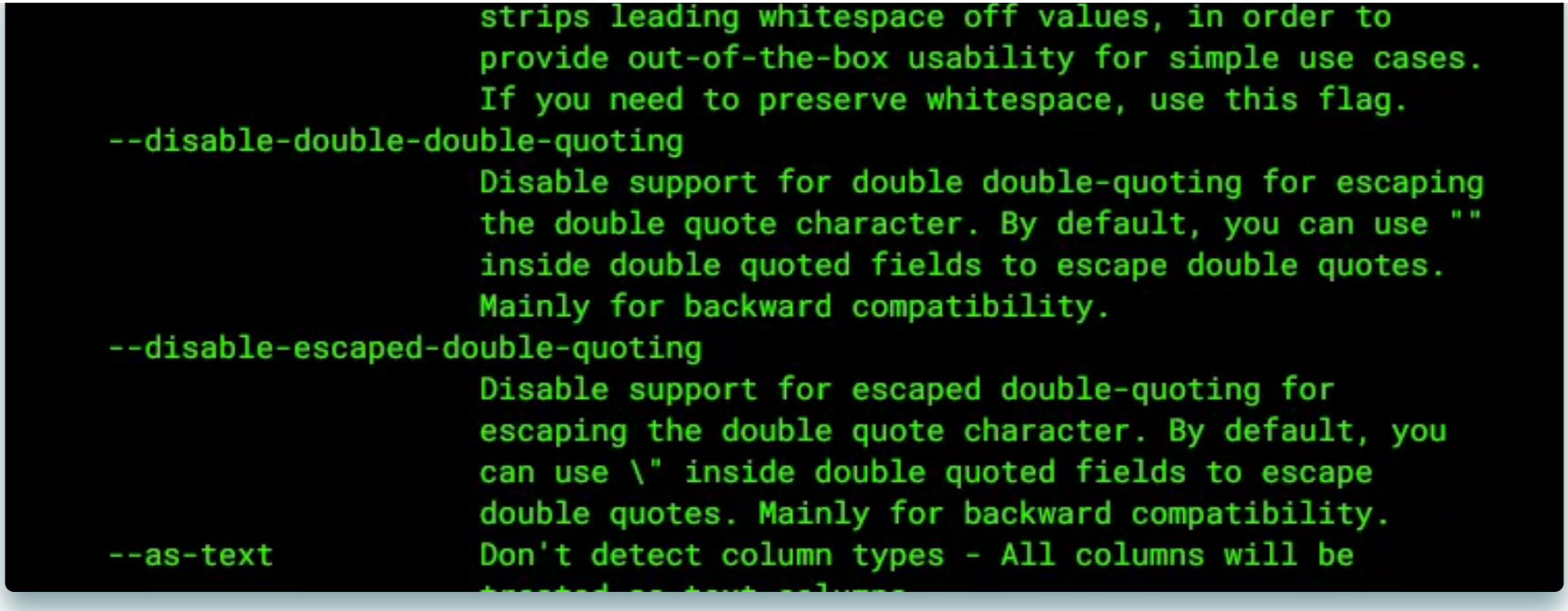
```
q  "select c1,c2 from ..."
```

- `output` 输出命令：作用在查询输出的结果集，比如：`-O` ，让查询出来的结果显示列名。

```
[root@iZ2zebfzaequ90bdlz820sZ software]# ps -ef | q -H "select count(UID) from - where UID='root'"
104
[root@iZ2zebfzaequ90bdlz820sZ software]# ps -ef | q -H -O "select count(UID) from - where UID='root'"
count(UID)
104
```

还有很多参数就不一一列举了，感兴趣的同学在官网上看下，接下来我们重点演示一下使用SQL如何应对各种查询日志的场景。

```
Input Data Options:
  -H, --skip-header   Skip header row. This has been changed from earlier
                      version - Only one header row is supported, and the
                      header row is used for column naming
  -d DELIMITER, --delimiter=DELIMITER
                      Field delimiter. If none specified, then space is used
                      as the delimiter.
  -p, --pipe-delimited
                      Same as -d '|'. Added for convenience and readability
  -t, --tab-delimited
                      Same as -d <tab>. Just a shorthand for handling
                      standard tab delimited file You can use $'\t' if you
                      want (this is how Linux expects to provide tabs in the
                      command line
  -e ENCODING, --encoding=ENCODING
                      Input file encoding. Defaults to UTF-8. set to none
                      for not setting any encoding - faster, but at your own
                      risk...
  -z, --gzipped       Data is gzipped. Useful for reading from stdin. For
                      files, .gz means automatic gunzipping
  -A, --analyze-only  Analyze sample input and provide information about
                      data types
  -m MODE, --mode=MODE
                      Data parsing mode. fluffy, relaxed and strict. In
                      strict mode, the -c column-count parameter must be
                      supplied as well
  -c COLUMN_COUNT, --column-count=COLUMN_COUNT
                      Specific column count when using relaxed or strict
                      mode
  -k, --keep-leading-whitespace
                      Keep leading whitespace in values. Default behavior
```



玩法贼多

下边咱们一起看几个查询日志的经常场景中，这个SQL该如何写。

1、关键字查询

关键字检索，应该是日常开发使用最频繁的操作，不过我个人认为这一点`q`并没有什么优势，因为它查询时必须指定某一列。

```
[root@iZ2zebfzaequ90bdlz820sZ software]# q "select * from douyin.log where c9 like '%待解析%'"
2021-06-11 14:46:49.323 INFO 22790 --- [nio-8888-exec-2] c.x.douyin.controller.ParserController : 待解析URL :url=https%3A%2F%2Fv.douyin.com%2Fe
2021-06-11 14:57:31.938 INFO 22790 --- [nio-8888-exec-5] c.x.douyin.controller.ParserController : 待解析URL :url=https%3A%2F%2Fv.douyin.com%2Fe
```

```
2021-06-11 15:23:48.004 INFO 22790 --- [nio-8888-exec-2] c.x.douyin.controller.ParserController : 待解析URL :url=https%3A%2F%2Fv.douyin.com%2Fe
2021-06-11 2
```

而用`grep`命令则是全文检索。

```
[root@iZ2zebfzaequ90bdlz820sZ software]# cat douyin.log | grep '待解析URL'
2021-06-11 14:46:49.323  INFO 22790 --- [nio-8888-exec-2] c.x.douyin.controller.ParserController   : 待解析URL :url=https%3A%2F%2Fv.douyin.com%
2021-06-11 14:57:31.938  INFO 22790 --- [nio-8888-exec-5] c.x.douyin.controller.ParserController   : 待解析URL :url=https%3A%2F%2Fv.douyin.com%
```

2、模糊查询

`like`模糊搜索，如果文本内容列有名字直接用列名检索，没有则直接根据列号c1、c2、cN。

```
[root@iZ2zebfzaequ90bdlz820sZ software]# cat test.log
abc
2
3
4
5
23
24
25
[root@iZ2zebfzaequ90bdlz820sZ software]# q -H -t "select * from test.log where abc like '%2%'"
Warning: column count is one - did you provide the correct delimiter?
2
23
24
25
```

3、交集并集

支持 `UNION` 和 `UNION ALL` 操作符对多个文件取交集或者并集。

如下建了 `test.log` 和 `test1.log` 两个文件，里边的内容有重叠，用 `union` 进行去重。

```
q -H -t "select * from test.log union select * from test1.log"

[root@iZ2zebfzaequ90bdlz820sZ software]# cat test.log
abc
2
3
4
5
[root@iZ2zebfzaequ90bdlz820sZ software]# cat test1.log
abc
3
4
5
6
[root@iZ2zebfzaequ90bdlz820sZ software]# q -H -t "select * from test.log union select * from test1.log"
Warning: column count is one - did you provide the correct delimiter?
Warning: column count is one - did you provide the correct delimiter?
2
3
```

```
4
5
6
```

4、内容去重

比如统计某个路径下的`./clicks.csv`文件中，`uuid`字段去重后出现的总个数。

```
q -H -t "SELECT COUNT(DISTINCT(uuid)) FROM ./clicks.csv"
```

5、列类型自动检测

注意：q会理解每列是数字还是字符串，判断是根据实数值比较，还是字符串比较进行过滤，这里会用到`-t`命令。

```
q -H -t "SELECT request_id,score FROM ./clicks.csv WHERE score > 0.7 ORDER BY score DESC LIMIT 5"
```

6、字段运算

读取系统命令查询结果，计算`/tmp`目录中每个用户和组的总值。可以对字段进行运算处理。

```
sudo find /tmp -ls | q "SELECT c5,c6,sum(c7)/1024.0/1024 AS total FROM - GROUP BY c5,c6 ORDER BY total desc"
```

```
[root@iZ2zebfzaequ90bdlz820sZ software]# sudo find /tmp -ls | q "SELECT c5,c6,sum(c7)/1024.0/1024 AS total FROM - GROUP BY c5,c6 ORDER BY total
www www 8.86311340332
root root 0.207922935486
mysql mysql 4.76837158203e-06
```

7、数据统计

统计系统拥有最多进程数的前 3个用户ID，按降序排序，这就需要和系统命令配合使用了，先查询所有进程再利用SQL筛选，这里的q命令就相当 `grep` 命令。

```
ps -ef | q -H "SELECT UID,COUNT(*) cnt FROM - GROUP BY UID ORDER BY cnt DESC LIMIT 3"

[root@iZ2zebfzaequ90bdlz820sZ software]# ps -ef | q -H "SELECT UID,COUNT(*) cnt FROM - GROUP BY UID ORDER BY cnt DESC LIMIT 3"
root 104
www 16
rabbitmq 4
[root@iZ2zebfzaequ90bdlz820sZ software]# ps -ef | q -H -O "SELECT UID,COUNT(*) cnt FROM - GROUP BY UID ORDER BY cnt DESC LIMIT 3"
UID cnt
root 110
www 16
rabbitmq 4
```

我们看到加与不加 `-O` 命令的区别就是否显示查询结果的标题。

8，连文件查

一般情况下，我们的日志文件会按天分割成很多个固定容量的子文件，在没有统一的日志收集服务器的情况下，如果不给个报错时间区间去查一个关键词，那么无异于大海捞针。

```
-rw-r--r--  1 root root      118652 6月  18 07:11 douyin-2021-06-18.0.log
-rw-r--r--  1 root root      118652 6月  18 07:12 douyin-2021-06-18.1.log
-rw-r--r--  1 root root      118652 6月  18 07:12 douyin-2021-06-18.2.log
-rw-r--r--  1 root root      118652 6月  18 07:12 douyin-2021-06-18.3.log
-rw-r--r--  1 root root      118652 6月  18 00:23 douyin.log
```

如果可以将所有文件内容合并后在查就会省事很多，q支持将文件像数据库表那样联合查询。

```
q -H "select * from douyin.log a join douyin-2021-06-18.0.log b on (a.c2=b.c3) where b.c1='root'"
```

总结

看完可能会有人抬杠：`q`写这么多代码直接用`awk`不香吗？额～ 介绍这个工具的初衷并不是说要替换现有哪种工具，而是多提供一种更为便捷的查日志方法。

我也有在用`awk`确实很强大没得说，但这里边涉及到一个学习成本的问题，琳琅满目的命令、匹配规则想玩转还是要下点功夫的。而对于新手程序员稍微有点数据库经验，写SQL问题都不大，上手`q`则会容易的多。

--- EOF ---

推荐↓↓↓

数据分析专栏

分享数据分析相关技术文章、教程、工具，包括但不限制于R、Python、Spark、MySQL、Excel等在数据分析、数据挖掘、数据抓取…

公众号

喜欢此内容的人还喜欢

Java代码中，如何监控Mysql的binlog?

Hollis