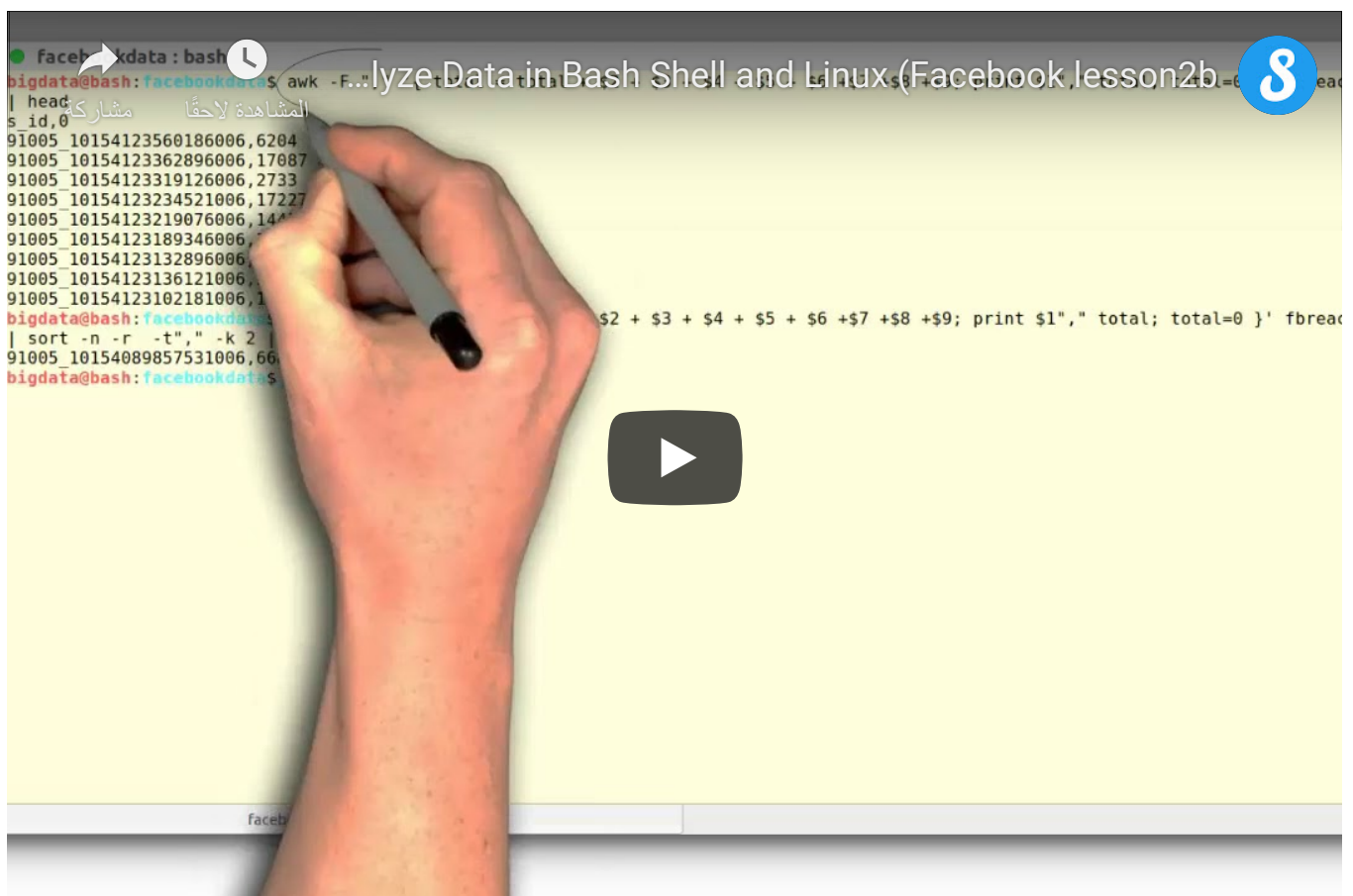


Find the most popular status entry (awk, sort, cat, csvcut, head)

WE'LL COVER THE FOLLOWING ^

- Do you want to know more?

To do this analysis efficiently, we'll use the command line language called **awk**, a tool that allows you to filter, extract and transform data files. awk is a very useful tool to put in your bag of tricks. But let's watch the following video lecture first!



Find the most popular status entry

To start, let's look at a very simple awk program to output every line of our **facebook.csv** file, where we specify the delimiter of the file (comma) using the

facebook.csv file, where we specify the delimiter of the file (comma) using the -F option:

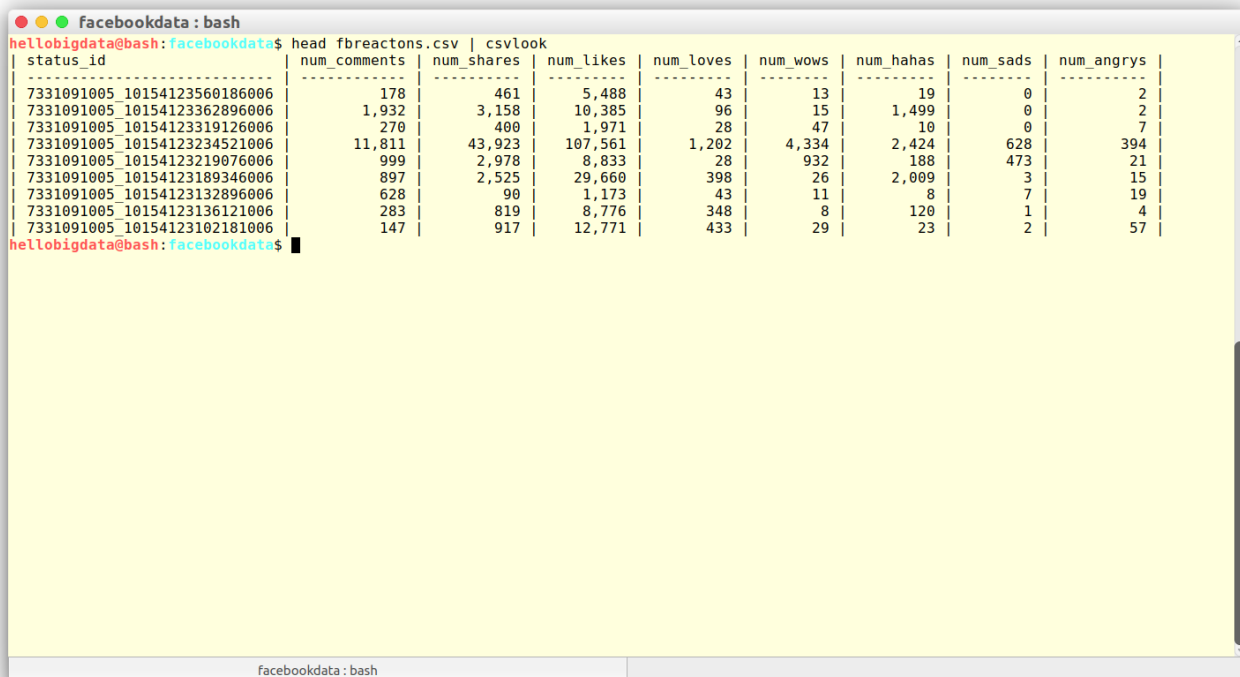
```
awk -F "," '{ print; }' facebookdata.csv
```

You should see the entire file being output to the screen. To only output the status ids (column 1), use the dollar sign (`$`) to denote columns as follows:

```
awk -F "," '{ print $1; }' facebookdata.csv | head
```

However, since the dataset has quoted ("text") cells we will use `csvcut` to extract the columns, e.g., we want to extract the column `1,8-15` into a file called `fbreactions.csv`. The idea is to sum-up all the reactions (columns `8 + ... + 15`) on each FB status and then find the status which had the maximum number of reactions.

```
csvcut -c 2,8-15 facebookdata.csv > fbreactions.csv
```



```
facebookdata : bash
hellobigdata@bash:facebookdata$ head fbreactions.csv | csvlook
status_id      num_comments  num_shares  num_likes  num_loves  num_wows  num_hahas  num_sads  num_angrys
-----
7331091005_10154123560186006 178          461         5,488      43          13         19         0         2
7331091005_10154123362896006 1,932        3,158       10,385     96          15        1,499      0         2
7331091005_10154123319126006 270          400         1,971      28          47         10         0         7
7331091005_10154123234521006 11,811       43,923      107,561    1,202       4,334     2,424     628       394
7331091005_10154123219076006 999          2,978       8,833      28          932        188       473       21
7331091005_10154123189346006 897          2,525       29,660     398         26        2,009      3         15
7331091005_10154123132896006 628          90          1,173      43          11         8         7         19
7331091005_10154123136121006 283          819         8,776     348         8         120       1         4
7331091005_10154123102181006 147          917         12,771     433         29         23        2         57
hellobigdata@bash:facebookdata$
```

Extract all the reactions into a file fbreactions.csv

To calculate the total number of reactions on each entry (status), all we need to do is horizontally add up all the numbers from the columns #8-15 and we do this easily with `awk`, as follows:

```
awk -F " " '{ total = total + $2 + $3 + $4 + $5 + $6 + $7 + $8 + $9; print $1 " " total; total=0 }
```

```
awk -F "," '{ total = total + $2 + $3 + $4 + $5 + $6 + $7 + $8 + $9; print $1, total; total=0 }' fbreactions.csv | \
head
```



Let's pay attention to the `awk` statement, which not only sums up the columns side by side, but also on each line prints two output (`status id` and `total` number of reaction on that row). Finally, at the end of each iteration, it nulls the `total=0`.

To get the status with `max` reactions, next, we sort the status ids, based on the number of reactions (column 2) using the `sort -n -r -t"," -k 2` function, which tells the system to sort out the piped (`|`) output numerically (`-n`), on the column 2 (`-k 2`) which is delimited by a comma (`,`):

```
awk -F "," '{ total = total + $2 + $3 + $4 + $5 + $6 + $7 + $8 + $9; print $1, total; total=0 }' fbreactions.csv | \
sort -n -r -t"," -k 2 | \
head -n 1
```



```
facebookdata : bash
hellobigdata@bash:facebookdata$ awk -F "," '{ total = total + $2 + $3 + $4 + $5 + $6 + $7 + $8 + $9; print $1, total; total=0 }' fbreactions.csv | head
status id,0
7331091005_10154123560186006,6204
7331091005_10154123362896006,17087
7331091005_10154123319126006,2733
7331091005_10154123234521006,172277
7331091005_10154123219076006,14452
7331091005_10154123189346006,35533
7331091005_10154123132896006,1979
7331091005_10154123136121006,10359
7331091005_10154123102181006,14379
hellobigdata@bash:facebookdata$ awk -F "," '{ total = total + $2 + $3 + $4 + $5 + $6 + $7 + $8 + $9; print $1, total; total=0 }' fbreactions.csv | sort -n -r -t"," -k 2 | head -n 1
7331091005_10154089857531006,668121
hellobigdata@bash:facebookdata$
```

The final output, tells us that the status id: `7331091005_10154089857531006` had the maximum number of reaction of total `668121`.

If we now use `grep`, we can easily find the message which had the largest number of reactions.

```
cat facebookdata.csv | grep 7331091005_10154089857531006
```

Let's make it little more beautiful using `csvcut`:

```
facebookdata: bash
hellobigdata@bash:facebookdata$ awk -F "," '{ total = total + $2 + $3 + $4 + $5 + $6 + $7 + $8 + $9; print $1," total; total=0 }' fbreactons .
.csv | sort -n -r -t"," -k 2 | head -n 1
7331091005_10154089857531006,668121
hellobigdata@bash:facebookdata$ cat facebookdata.csv | csvcut -c 1,2 | grep 7331091005_10154089857531006 | csvlook
| 7331091005_10154089857531006 | LeBron and the Cavs are tired of being bullied |
|-----|-----|
hellobigdata@bash:facebookdata$
```

FB Awk total find

However, we want to make it more interesting! let's efficiently pipe all the steps shown above into a single command and find the message as follows:

```
cat facebookdata.csv | \
csvcut -c 2,8-15 | \
awk -F "," '{ total = total + $2 + $3 + $4 + $5 + $6 + $7 + $8 + $9; print $1,"total; total=0 }' \
sort -n -r -t"," -k 2 | \
head -n 1
```

Do you want to know more?

['awk' man page](#) 