# csvkitcat

*Release 1.6.1-alpha*

**unknown**

**Oct 03, 2020**

# CONTENTS

CHAPTER

ONE

# CHANGELOG



## 1.1 1.6.1-alpha (inprogress)

Likely the last update as "csvkitcat" before moving to csvmedkit

- csvwhere
- **csvsed, csvrgrep**
  - revamped required arguments and additional expressions
- csvsed
  - –like-grep now only filters by the first (required) expression, as chaining additional expressions likely results in a too-small result set for most use cases
- csvchart: killed, moved to csvviz library

## 1.2 1.5.6-alpha

- changed various flags for many of the tools
- added `--version` flag
- cleaned up internals and csvkit inheritance
- [NOT YET FIXED] csvsed is fixed

## 1.3 1.5.5-alpha

- *csvrgrep*: like `csvgrep`, but designed for when you want to concisely do a lot of regex filtering over a combination of columns
- csvsed: reworking its internals. Still being worked on, but current tests pass

## 1.4 1.5.1-alpha

- the `regex` module seems to be very slow (e.g. csvsed), so it's been replaced with standard `re` in all tools except `csvnorm`. Hopefully that's ok!

## 1.5 1.5.0-alpha

*csvgroupby*: for doing the equivalent of SQL `GROUP BY [columns]`

## 1.6 1.4.0-alpha

*csvpivot*: Pivot a table on rows and columns (TK better label)

## 1.7 1.3.7-alpha

*csvflatten* tweaked: - If a value contains newlines, then csvflatten will print newlines/rows; this is independent of whether the chop-length has been set.

## 1.8 1.3.5-alpha

*csvcount* has been overhauled:

- Basic functionality returns not just number of rows, but cells, empty rows+cells, and blank lines
- Restrict columns to search with``-c/–column``

## 1.9 1.3.0-alpha

- *csvxfind*: Findall regex matches in [COLUMN] and group concat them into a new column

## 1.10 1.2.0-alpha

- *csvxcap*: Extract captured group patterns from one column and create new columns

## 1.11 1.1.0-alpha

- *csvxplit*: create `n` new columns based on splitting a [COLUMN] by [PATTERN]

## 1.12 1.0.1-alpha

- *csvnorm*: renamed and refactored from `csvsqueeze`
  - csvnorm -C/–change-case for converting values to all upper or lower

## 1.13 1.0.0-alpha

Basic functionality and testing for these tools:

- *csvcount*
- *csvflatten*
- *csvsed*
- *csvslice*
- csvsqueeze (likely to be refactored)

CHAPTER

TWO

# UTILITIES

## 2.1 csvcount



### 2.1.1 Description

Count the number of rows, cells, empty rows and cells, and blank lines

Optionally, given a regex or list of regexes, count:

- the number of rows with at least 1 match
- the number of cells with at least 1 match
- the number of total matches (e.g. some cell values may match the patterns more than once)

### 2.1.2 Examples

#### Basic counting of file records

Basic example:

```
$ csvcount examples/dummy4.csv

rows,cells,empty_rows,empty_cells,blank_lines
4,12,0,0,0
```

On real world data:

```
$ csvcount examples/realdata/osha-violation.csv |  csvlook -I

| rows  | cells  | empty_rows | empty_cells | blank_lines |
| ----- | ------ | ---------- | ----------- | ----------- |
| 29999 | 299990 | 0          | 39035       | 0           |
```

On file with empty rows and cells:

```
$ csvcount examples/empties.csv

rows,cells,empty_rows,empty_cells,blank_lines
4,12,1,4,0
```

On file with blank lines:

```
$  csvcount examples/blankedlines.csv

rows,cells,empty_rows,empty_cells,blank_lines
3,9,0,0,4
```

### Counting of patterns

Counting words with at least 5 letters:

```
$ csvcount -P '[A-z]{5,}' examples/longvals.csv | csvlook

| pattern    | rows | cells | matches |
| ---------- | ---- | ----- | ------- |
| [A-z]{5,} |    3 |     9 |      43 |
```

Limiting the match searching to the `description` column:

```
$ csvcount -P '[A-z]{5,}' -c description examples/longvals.csv | csvlook

| pattern    | rows | cells | matches |
| ---------- | ---- | ----- | ------- |
| [A-z]{5,} |    3 |     3 |      31 |
```

Counting (naively) the number of @mentions, #hastags, and URLs in tweet texts:

```
$ csvcount -P '@\w+' -P '#\w+' -P 'https:' \
    -c text examples/realdata/tweets-whitehouse.csv \
    | csvlook


| pattern |  rows | cells | matches |
| ------- | ----- | ----- | ------- |
| @\w+    | 1,591 | 1,591 |   2,266 |
| #\w+    |   409 |   409 |     560 |
| https:  | 2,596 | 2,596 |   2,938 |
```

## 2.2 csvflatten

### 2.2.1 Description

Print records in column-per-line format. Best used in conjunction with csvlook

Similar in concept to xsv flatten, though the output is much different.

TK/TODO: copy text/rationale from original Github issue

### 2.2.2 Basic example and transformation

Given the file at `examples/statecodes.csv`, which looks like this:

| code | name |
|---|---|
| IA | Iowa |
| RI | Rhode Island |
| TN | Tennessee |

And then passing it into **`csvflatten`**:

```
$ csvflatten examples/statecodes.csv
```

Results in this output:

```
fieldname,value
code,IA
name,Iowa
~~~~~~~~~,
code,RI
name,Rhode Island
~~~~~~~~~,
code,TN
name,Tennessee
```

Which looks like this as a table:

| fieldname | value |
|---|---|
| code | IA |
| name | Iowa |
| | |
| code | RI |
| name | Rhode Island |
| | |
| code | TN |
| name | Tennessee |

### 2.2.3 More examples

Shakespeare and csvlook:

```
$ csvflatten examples/hamlet.csv | csvlook

| fieldname | value                                          |
| --------- | ---------------------------------------------- |
| act       | 1                                              |
| scene     | 5                                              |
| speaker   | Horatio                                        |
| lines     | Propose the oath, my lord.                     |
| ~~~~~~~~~ |                                                |
| act       | 1                                              |
| scene     | 5                                              |
| speaker   | Hamlet                                         |
| lines     | Never to speak of this that you have seen,     |
|           | Swear by my sword.                             |
| ~~~~~~~~~ |                                                |
| act       | 1                                              |
| scene     | 5                                              |
| speaker   | Ghost                                          |
| lines     | [Beneath] Swear.                               |
| ~~~~~~~~~ |                                                |
| act       | 3                                              |
| scene     | 4                                              |
| speaker   | Gertrude                                       |
| lines     | O, speak to me no more;                        |
|           | These words, like daggers, enter in mine ears; |
|           | No more, sweet Hamlet!                         |
| ~~~~~~~~~ |                                                |
| act       | 4                                              |
| scene     | 7                                              |
| speaker   | Laertes                                        |
| lines     | Know you the hand?                             |
```

Chopping the text length to no more than 20 characters per line:

```
$ csvflatten -L 20 examples/hamlet.csv | csvlook


fieldname | value |
——————— | ——————— |
act | 1 |
scene | 5 |
speaker | Horatio |
lines | Propose the oath, my |
      | lord. |
~~~~~~~~~ | |
act | 1 |
scene | 5 |
speaker | Hamlet |
lines | Never to speak of th |
      | is that you have see |
      | n, |
```

| Swear by my sword. |
~~~~~~~~~ | |
act | 1 |
scene | 5 |
speaker | Ghost |
lines | [Beneath] Swear. |
~~~~~~~~~ | |
act | 3 |
scene | 4 |
speaker | Gertrude |
lines | O, speak to me no mo |
| re; |
| These words, like da |
| ggers, enter in mine |
| ears; |
| No more, sweet Hamle |
| t! |
~~~~~~~~~ | |
act | 4 |
scene | 7 |
speaker | Laertes |
lines | Know you the hand? |

Another example with csvlook:

```
$ csvflatten examples/longvals.csv -L 50 | csvlook | pbcopy
```

```
| fieldname    | value                                              |
| ------------ | -------------------------------------------------- |
| title        | Raising Arizona                                    |
| release_date | March 13, 1987                                     |
| length       | 94                                                 |
| box_office   | 292000000                                          |
| description  | Repeat convict "Hi" and police officer "Ed" meet i |
|              | n prison, get married, and hope to raise a family. |
| url          | https://en.wikipedia.org/wiki/Raising_Arizona      |
| ~~~~~~~~~    |                                                    |
| title        | Face/Off                                           |
| release_date | June 27, 1997                                      |
| length       | 139                                                |
| box_office   | 80000000                                           |
| description  | John Travolta plays an FBI agent and Nicolas Cage  |
|              | plays a terrorist, sworn enemies who assume each o |
|              | ther's physical appearance.                        |
| url          | https://en.wikipedia.org/wiki/Face/Off             |
| ~~~~~~~~~    |                                                    |
| title        | Adaptation                                         |
| release_date | Dec. 6, 2002                                       |
| length       | 114                                                |
| box_office   | 32800000                                           |
| description  | The self-loathing Charlie Kaufman is hired to writ |
```

(continues on next page)
~~~~~~~~~

(continued from previous page)

```
|              | e the screenplay adaptation of Susan Orlean's The  |
|              | Orchid Thief.                                      |
| url          | https://en.wikipedia.org/wiki/Adaptation_(film)    |
```

Label each line of a chopped field with its respective header:

```
$ csvflatten examples/longvals.csv -L 50 -B | csvlook


| fieldname     | value                                              |
| ------------- | -------------------------------------------------- |
| title         | Raising Arizona                                    |
| release_date  | March 13, 1987                                     |
| length        | 94                                                 |
| box_office    | 292000000                                          |
| description   | Repeat convict "Hi" and police officer "Ed" meet i |
| description~1 | n prison, get married, and hope to raise a family. |
| url           | https://en.wikipedia.org/wiki/Raising_Arizona      |
| ~~~~~~~~~     |                                                    |
| title         | Face/Off                                           |
| release_date  | June 27, 1997                                      |
| length        | 139                                                |
| box_office    | 80000000                                           |
| description   | John Travolta plays an FBI agent and Nicolas Cage  |
| description~1 | plays a terrorist, sworn enemies who assume each o |
| description~2 | ther's physical appearance.                        |
| url           | https://en.wikipedia.org/wiki/Face/Off             |
| ~~~~~~~~~     |                                                    |
| title         | Adaptation                                         |
| release_date  | Dec. 6, 2002                                       |
| length        | 114                                                |
| box_office    | 32800000                                           |
| description   | The self-loathing Charlie Kaufman is hired to writ |
| description~1 | e the screenplay adaptation of Susan Orlean's The  |
| description~2 | Orchid Thief.                                      |
| url           | https://en.wikipedia.org/wiki/Adaptation_(film)    |
```

## 2.3 csvgroupby

### 2.3.1 Description

Do SQL-like `GROUP BY` aggregations

Similar to *csvpivot*, except `csvgroupby` allows for multiple aggregation value columns. Call `-a/--agg` multiple times for multiple aggregations.

Example:

```
$ csvgroupby -c 'gender,race' -a count -a mean:age -a 'TOTAL age|sum:age' examples/
↪peeps.csv | csvlook

| gender | race   | Count | Mean_of_age | TOTAL age |
| ------ | ------ | ----- | ----------- | --------- |
| female | white  |     1 |        20.0 |        20 |
| female | black  |     2 |        22.5 |        45 |
| female | asian  |     1 |        25.0 |        25 |
| male   | asian  |     1 |        20.0 |        20 |
| male   | latino |     1 |        25.0 |        25 |
```

## 2.4 csvnorm



### 2.4.1 Description

Normalize whitespace, newlines, and character casing.

### 2.4.2 Examples

Basic examples:

```
$ csvnorm examples/consec_ws.csv

id,phrase
1,hello world
2,good bye
3,a ok



$ csvnorm examples/multi1.csv

id,text
1,hey
2,hello world
3,"to be, or not to be?"
```

## 2.5 csvpivot



### 2.5.1 Description

Do a simple pivot table, by row, column, or row and column

### 2.5.2 Examples

#### Basic counting

Pivot by rows:

```
$ csvpivot -r race examples/peeps.csv

race,Count
white,1
asian,2
black,2
latino,1
```

Pivot across several fields (TK?) of rows:

```
$ csvpivot -r race,gender examples/peeps.csv

race,gender,Count
white,female,1
asian,male,1
asian,female,1
black,female,2
latino,male,1
```

Pivot along columns:

```
$ csvpivot -c gender examples/peeps.csv

female,male
4,2
```

Pivot on rows and columns:

```
$ csvpivot -r race -c gender examples/peeps.csv

race,female,male
white,1,0
asian,1,1
black,2,0
latino,0,1
```

## 2.6 csvrgrep



### 2.6.1 Description

Basically like `csvgrep`, but designed for situations in which you want to do a variety of filters across a variety of column combinations.

Example:

```
$ csvrgrep -E '\w{5,}' 'name,address' \
           -E '\d{3}-\d{4}-XX'  \
           -E 'open\w*|close\w*' 'status,revised_status' \
           examples/SOMERANDODATA.csv
```

### 2.6.2 Examples

To mimic csvkit default behavior:

```
$ csvrgrep -c 1,2,3 -E 'pattern' --match-literal --all-match data.csv
```

By default, `csvrgrep` returns matches when any column matches the regex pattern

## 2.7 csvsed



### 2.7.1 Description

Like `sed`, but on a per-column basis

Example:

```
$ csvsed "Ab[bi].+" "Abby" -E "(B|R)ob.*" "\1ob" -E "(?:Jack|John).*" "John" ␣
↪examples/aliases.csv
```

(continues on next page)

(continued from previous page)

```
id,to,from
1,Abby,Bob
2,Bob,John
3,Abby,John
4,John,Abner
5,Rob,John
6,Jon,Abby
7,Rob,Abby
```

### 2.7.2 Examples

#### Multiple expressions

#### Cleaning up currency values

- remove commas and spaces
- replace negative notation, from `(42)` to `-42`
- add a decimal amount for all whole dollar values

```
$ csvsed '[$, ]' "" examples/ledger.csv \
        -c 'revenue,gross' \
        -E '\((.+?)\)' '\-\1' \
        -E '(?<=\d)(\d{2})$' '\1.00'
```

```
id,name,revenue,gross
001,apples,21456.00,3210.45
002,bananas,2442.00,-1234.00
003,cherries,9700.55,-7.90
004,dates,4102765.33,18765.00
005,eggplants,3987.00,501.00
006,figs,30333.00,-777.66
006,grapes,154321.98,-32654.00
```

#### Replace entire field with `-R/--replace`

```
$ cat examples/rolodex.csv |
    csvsed -c phone '(?P<area>\d{3}).*?(?P<x>\d{3}).*?(?P<y>\d{4})' \
                    '(\g<area>)-\g<x>-\g<y>'
```

Output:

```
name,zipcode,phone
Andie,10003,((555)-123-4567
Betty,23456,1-(800)-777-2222
Caren,33033,1((900)-333-1212
Denny,42742,(212)-867-5309
Ellie,90210,(555)-404-2020
```

Some of the phone numbers have superfluous characters, like country calling code (i.e. "1-") and unneeded parentheses. Use the `-R` option to specify that the *replacement* pattern should overwrite the entire field:

```
$ cat examples/rolodex.csv |
    csvsed -R -c phone '(?P<area>\d{3}).*?(?P<x>\d{3}).*?(?P<y>\d{4})' \
                    '(\g<area>)-\g<x>-\g<y>'


name,zipcode,phone
Andie,10003,(555)-123-4567
Betty,23456,(800)-777-2222
Caren,33033,(900)-333-1212
Denny,42742,(212)-867-5309
Ellie,90210,(555)-404-2020
```

## 2.8 csvslice



### 2.8.1 Description

Returns the header, plus rows in the specified 0-index range, half-open-interval

Similar to xsv slice

Example:

```
$ csvslice -B 2 -L 2 examples/yes.csv

code,value
3,Yes
4,Y
```

### 2.8.2 Examples

## 2.9 csvxcap

### 2.9.1 Description

Extract captured group patterns from one column and create new columns

### 2.9.2 Examples

Example with no captured group, just a pattern:

```
$ csvxcap 'name' '\w+' examples/honorifics-fem.csv | csvlook

| code | name       | name_xcap |
| ---- | ---------- | --------- |
|    1 | Mrs. Smith | Mrs       |
|    2 | Miss Daisy | Miss      |
|    3 | Ms. Doe    | Ms        |
|    4 | Mrs Miller | Mrs       |
|    5 | Ms Lee     | Ms        |
|    6 | miss maam  | miss      |
```

Example with captured groups:

```
$ csvxcap 'name' '(\w+)\.? (\w+)' examples/honorifics-fem.csv | csvlook

| code | name       | name_xcap1 | name_xcap2 |
| ---- | ---------- | ---------- | ---------- |
|    1 | Mrs. Smith | Mrs        | Smith      |
|    2 | Miss Daisy | Miss       | Daisy      |
|    3 | Ms. Doe    | Ms         | Doe        |
|    4 | Mrs Miller | Mrs        | Miller     |
|    5 | Ms Lee     | Ms         | Lee        |
|    6 | miss maam  | miss       | maam       |
```

Example with named captured groups (allows naming of headers):

```
$ csvxcap 'name' '(?P<prefix>\w+)\.? (?P<sur>\w+)' examples/honorifics-fem.csv |␣
↪csvlook


| code | name       | name_prefix | name_sur |
| ---- | ---------- | ----------- | -------- |
|    1 | Mrs. Smith | Mrs         | Smith    |
|    2 | Miss Daisy | Miss        | Daisy    |
|    3 | Ms. Doe    | Ms          | Doe      |
|    4 | Mrs Miller | Mrs         | Miller   |
|    5 | Ms Lee     | Ms          | Lee      |
|    6 | miss maam  | miss        | maam     |
```

## 2.10 csvxfind



### 2.10.1 Description

Find all regex [PATTERN] in [COLUMN], create new column with all matches

### 2.10.2 Examples

Basic:

```
$ csvxfind 'text' '@\w+' examples/mentions.csv | csvlook


| id | text                                                | text_xfind           ␣
↪|
| -- | --------------------------------------------------- | ---------------------␣
↪|
|  1 | hey                                                 |                      ␣
↪|
|  2 | hello @world                                        | @world               ␣
↪|
|  3 | Just like @a_prayer, your @Voice can take me @there! | @a_prayer;@Voice;@there␣
↪|
```

Specify delimiter with `-D`:

```
$ csvxfind -D ', '  'text' '@\w+' examples/mentions.csv | csvlook

| id | text                                                | text_xfind           ␣
↪  |
| -- | --------------------------------------------------- | ---------------------
↪- |
|  1 | hey                                                 |                      ␣
↪  |
|  2 | hello @world                                        | @world               ␣
↪  |
|  3 | Just like @a_prayer, your @Voice can take me @there! | @a_prayer, @Voice,␣
↪@there |
```

Limit matches with `-n`:

```
$ csvxfind -n 2 'text' '@\w+' examples/mentions.csv | csvlook

| id | text                                                | text_xfind       |
| -- | --------------------------------------------------- | ---------------- |
|  1 | hey                                                 |                  |
|  2 | hello @world                                        | @world           |
|  3 | Just like @a_prayer, your @Voice can take me @there! | @a_prayer;@Voice |
```

## 2.11 csvxplit



### 2.11.1 Description

Split a column by pattern into n-columns

### 2.11.2 Examples

Example with literal match:

```
$ csvxplit -n 2 items '|' examples/pipes.csv

code,items,items_xs_0,items_xs_1,items_xs_2
0001,hey,hey,,
0002,hello|world,hello,world,
0003,a|b|c|d|,a,b,c|d|
```

With regex:

```
$ csvxplit -n 2 items '\|' examples/pipes.csv
```

CHAPTER

THREE

# FAQ

Q. How is this related to wireservice/csvkit?

A. **csvkitcat** is an extension of *csvkit* (and thus has csvkit and agate as dependencies) that adds a bunch of new command-line utilities for data-wrangling convenience.

Q. What are the point of these new utilities?

A. As useful as core csvkit is, there are still a bunch of common data-wrangling tasks that are cumbersome to perform even when the data is in a spreadsheet or SQL database. "Cumbersome", in the sense that you'd basically have to write a custom Python script to do them.