

[Home](#) > [Bash scripting](#) > [How To Parse CSV Files In Bash Scripts In Linux](#)[BASH SCRIPTING](#) ♦ [BASH](#) ♦ [BASH TIPS](#) ♦ [LINUX](#) ♦ [LINUX COMMANDS](#) ♦ [SCRIPTING](#) ♦ [SHELL SCRIPTS](#) ♦
[UNIX/LINUX BEGINNERS](#)

How To Parse CSV Files In Bash Scripts In Linux

Work With CSV Files In Bash Scripts

Written by Karthick | March 2, 2022 | 1369 Views

💬 2 comments

5 ❤️

[f](#) [t](#) [in](#) [g](#) [w](#) [e](#) [m](#)

Comma-separated values aka **CSV** is a semi-structured data that uses comma as the delimiter to separate the words. CSV file formats are very popular among data professionals since they have to deal with a lot of CSV files and process it to create insights. In this article, we will be focusing on how to parse CSV files in Bash shell scripts in Linux.

In most parts of this article, I will be using `awk` and `sed` tools for csv parsing instead of combining different commands like `grep`, `cut`, `tr`, etc.

The `awk` utility reduces the complexity of piping multiple commands or writing a loop with logic to grab the data. Instead, you can write a one-liner code in `awk` to do the job.



1. Preparing CSV File For Processing

Your CSV file may be generated from a database, an API, or you might have run some commands and converted the output to delimit in CSV format. In any of the cases, you have to first analyze the dataset before running your logic on top of it.

As a best practice, you should cleanse your dataset before using it. Why should we cleanse the dataset? There may be situations where there will be empty cell values or no proper formatting in headers, extra columns that are not required for processing, and many more.

I am using the below CSV data, which I grabbed from [Kaggle](#) for demonstration purposes.

```
Player_Id,Player_Name,DOB,Batting_Hand,Bowling_Skill,Country
1,SC Ganguly,8-Jul-72,Left_Hand,Right-arm medium,
2,BB McCullum,27-Sep-81,Right_Hand,Right-arm medium,
3,RT Ponting,19-Dec-74,Right_Hand,Right-arm medium,
4,DJ Hussey,15-Jul-77,Right_Hand,Right-arm offbreak,Australia
5,Mohammad Hafeez,17-Oct-80,,Right-arm offbreak,Pakistan
6,R Dravid,11-Jan-73,,Right-arm offbreak,India
7,W Jaffer,16-Feb-78,,Right-arm offbreak,India
8,V Kohli,5-Nov-88,,Right-arm medium,India
9,JH Kallis,16-Oct-75,,Right-arm fast-medium,South Africa
10,CL White,18-Aug-83,Right_Hand,Legbreak googly,Australia
11,MV Boucher,3-Dec-76,Right_Hand,Right-arm medium,South Africa
12,B Akhil,7-Oct-77,Right_Hand,Right-arm medium-fast,India
13,AA Noffke,30-Apr-77,Right_Hand,Right-arm fast-medium,Australia
14,P Kumar,2-Oct-86,Right_Hand,Right-arm medium,India
15,Z Khan,7-Oct-78,Right_Hand,Left-arm fast-medium,India
```

1.1. Replace Empty Cells

In some cases, the CSV file will not have any values in particular cells. Take a look at the below screenshot where there are some empty cells between the columns.



Player_Id	Player_Name	DOB	Batting_Hand	Bowling_Skill	Country
1	SC Ganguly	8-Jul-72	Left_Hand	Right-arm medium	
2	BB McCullum	27-Sep-81	Right_Hand	Right-arm medium	
3	RT Ponting	19-Dec-74	Right_Hand	Right-arm medium	
4	DJ Hussey	15-Jul-77	Right_Hand	Right-arm offbreak	Australia
5	Mohammad Hafeez	17-Oct-80		Right-arm offbreak	Pakistan
6	R Dravid	11-Jan-73		Right-arm offbreak	India
7	W Jaffer	16-Feb-78		Right-arm offbreak	India
8	V Kohli	5-Nov-88		Right-arm medium	India
9	JH Kallis	16-Oct-75		Right-arm fast-medium	South Africa
10	CL White	18-Aug-83	Right_Hand	Legbreak googly	Australia
11	MV Boucher	3-Dec-76	Right_Hand	Right-arm medium	South Africa
12	B Akhil	7-Oct-77	Right_Hand	Right-arm medium-fast	India
13	AA Noffke	30-Apr-77	Right_Hand	Right-arm fast-medium	Australia
14	P Kumar	2-Oct-86	Right_Hand	Right-arm medium	India
15	Z Khan	7-Oct-78	Right_Hand	Left-arm fast-medium	India

Sample CSV File

I would always replace it with "NA" or "No Value", so there will be no empty cells. You can use the following awk snippet to replace any empty cell with your desired value. In this case, I am replacing the empty cells with "No value".

```
awk 'BEGIN{FS=",";OFS=","}
{
  for(i=1;i≤NF;i++)
  {
    if($i == ""){
      $i="No Value"
    }
  }
  print
}' ~/Downloads/Player.csv > player_cleaned.csv
```

The way this snippet works is I am setting the field separator and output field separator to comma (FS=","; OFS=","). Using for loop, iterate through each cell in a line, and if a cell is found empty (\$i == "") then replace it with "No value" (\$i="No value"). You have to redirect the changes to a new file.

Suggested Read:

- [Bash Redirection Explained With Examples](#)

1.2. Capitalize The Header

CSV files may or may not have headers. But if there is a header I would always capitalize it for better readability. You can do it easily using awk or sed. I will show you both the ways.

```
awk 'BEGIN{FS=",";OFS=","}
{
  if(NR==1){
    print toupper($0)
  } else {
```



```

    print
}
}' player.csv > player_cleaned.csv

```

Here, we are checking if the line is first-line using `(NR==1)` and using the `toupper()` function to capitalize it. The same snippet can be written as a one-liner.

```
awk 'NR==1{ print toupper($0) }NR>1' player.csv > player_cleaned.csv
```

Using `awk`, you have to again redirect the changes to a new file. Instead, you can use `'sed'` to modify the changes directly into the file. Here `\U` converts the case to uppercase. If you want to do lowercase conversion, use `\L`.

```
$ sed -i -e '1 s/(.*)/\U\1/' player_cleaned.csv
$ cat player_cleaned.csv

```

1.3. Remove Trailing Comma

Your CSV file may have a comma at the end. To clean the trailing commas, you can follow the below method.

I have purposely added a trailing comma from lines **7** to **11** in my data file.

```
karthick@HP-NOTEBOOK:~/Documents$ bcat player_cleaned.csv
```

	File: player_cleaned.csv
1	PLAYER_ID,PLAYER_NAME,DOB,BATTING HAND,BOWLING SKILL,COUNTRY
2	1,SC Ganguly,8-Jul-72,Left_Hand,Right-arm medium,No Value
3	2,BB McCullum,27-Sep-81,Right_Hand,Right-arm medium,No Value
4	3,RT Ponting,19-Dec-74,Right_Hand,Right-arm medium,No Value
5	4,DJ Hussey,15-Jul-77,Right_Hand,Right-arm offbreak,Australia
6	5,Mohammad Hafeez,17-Oct-80,No Value,Right-arm offbreak,Pakistan
7	6,R Dravid,11-Jan-73,No Value,Right-arm offbreak,India,
8	7,W Jaffer,16-Feb-78,No Value,Right-arm offbreak,India,
9	8,V Kohli,5-Nov-88,No Value,Right-arm medium,India,
10	9,JH Kallis,16-Oct-75,No Value,Right-arm fast-medium,South Africa,
11	10,CL White,18-Aug-83,Right_Hand,Legbreak googly,Australia,
12	11,MV Boucher,3-Dec-76,Right_Hand,Right-arm medium,South Africa
13	12,B Akhil,7-Oct-77,Right_Hand,Right-arm medium-fast,India
14	13,AA Noffke,30-Apr-77,Right_Hand,Right-arm fast-medium,Australia
15	14,P Kumar,2-Oct-86,Right_Hand,Right-arm medium,India
16	15,Z Khan,7-Oct-78,Right_Hand,Left-arm fast-medium,India

CSV File With Trailing Commas

To remove all the trailing commas, run the following `sed` command:

```
$ sed -i 's/,,$//' ~/Documents/player_cleaned.csv
```



```
karthick@HP-NOTEBOOK:~/Documents$ sed -i 's/,,$//' ~/Documents/player_cleaned.csv
karthick@HP-NOTEBOOK:~/Documents$ bcat player_cleaned.csv
```

	File: player_cleaned.csv
1	PLAYER_ID,PLAYER_NAME,DOB,BATTING HAND,BOWLING SKILL,COUNTRY
2	1,SC Ganguly,8-Jul-72,Left_Hand,Right-arm medium,No Value
3	2,BB McCullum,27-Sep-81,Right_Hand,Right-arm medium,No Value
4	3,RT Ponting,19-Dec-74,Right_Hand,Right-arm medium,No Value
5	4,DJ Hussey,15-Jul-77,Right_Hand,Right-arm offbreak,Australia
6	5,Mohammad Hafeez,17-Oct-80,No Value,Right-arm offbreak,Pakistan
7	6,R Dravid,11-Jan-73,No Value,Right-arm offbreak,India
8	7,W Jaffer,16-Feb-78,No Value,Right-arm offbreak,India
9	8,V Kohli,5-Nov-88,No Value,Right-arm medium,India
10	9,JH Kallis,16-Oct-75,No Value,Right-arm fast-medium,South Africa
11	10,CL White,18-Aug-83,Right_Hand,Legbreak googly,Australia
12	11,MV Boucher,3-Dec-76,Right_Hand,Right-arm medium,South Africa
13	12,B Akhil,7-Oct-77,Right_Hand,Right-arm medium-fast,India
14	13,AA Noffke,30-Apr-77,Right_Hand,Right-arm fast-medium,Australia
15	14,P Kumar,2-Oct-86,Right_Hand,Right-arm medium,India
16	15,Z Khan,7-Oct-78,Right_Hand,Left-arm fast-medium,India

```
karthick@HP-NOTEBOOK:~/Documents$ _
```

Remove Trailing Commas In CSV File

Now we are done with the cleaning part. There may be a few more steps required for you but that depends on how your CSV file is structured and what needs to be cleaned.

2. Pretty Print CSV File In Terminal

If you are trying to display the CSV files in the terminal, then there are a few options where you can print the file in tabular format which will give you better readability.

2.1. Column Command

The first approach is to use the `column` command. Column command accepts a separator which is set to comma and a delimiter to split the column which is set to tab in the below command. You can also set your own custom delimiters.

```
$ cat player_cleaned.csv | column -s, -t
$ column -s, -t player_cleaned.csv
```

```
karthick@HP-NOTEBOOK:~/Documents$ column -s, -t player_cleaned.csv
PLAYER_ID  PLAYER_NAME  DOB      BATTING HAND  BOWLING SKILL  COUNTRY
1          SC Ganguly   8-Jul-72   Left_Hand     Right-arm medium  No Value
2          BB McCullum  27-Sep-81  Right_Hand    Right-arm medium  No Value
3          RT Ponting   19-Dec-74  Right_Hand    Right-arm medium  No Value
4          DJ Hussey    15-Jul-77  Right_Hand    Right-arm offbreak  Australia
5          Mohammad Hafeez  17-Oct-80  No Value      Right-arm offbreak  Pakistan
6          R Dravid     11-Jan-73  No Value      Right-arm offbreak  India
7          W Jaffer     16-Feb-78  No Value      Right-arm offbreak  India
8          V Kohli      5-Nov-88   No Value      Right-arm medium    India
9          JH Kallis    16-Oct-75  No Value      Right-arm fast-medium  South Africa
10         CL White     18-Aug-83  Right_Hand    Legbreak googly    Australia
11         MV Boucher   3-Dec-76   Right_Hand    Right-arm medium    South Africa
12         B Akhil      7-Oct-77   Right_Hand    Right-arm medium-fast  India
13         AA Noffke    30-Apr-77  Right_Hand    Right-arm fast-medium  Australia
14         P Kumar      2-Oct-86   Right_Hand    Left-arm medium     India
15         Z Khan       7-Oct-78   Right_Hand    Left-arm fast-medium  India
karthick@HP-NOTEBOOK:~/Documents$ _
```

Display CSV File With Column Command



2.2. CSV Look Command

Csvlook is a utility that comes with the [csvkit](#) package. There is no need to set a delimiter as we did with the `column` command.

```
$ cat player_cleaned.csv | csvlook
$ csvlook player_cleaned.csv
```

```
karthick@HP-NOTEBOOK:~/Documents$ csvlook player_cleaned.csv
| PLAYER_ID | PLAYER_NAME | DOB | BATTING HAND | BOWLING SKILL | COUNTRY |
|-----|-----|-----|-----|-----|-----|
| 1 | SC Ganguly | 8-Jul-72 | Left_Hand | Right-arm medium | No Value |
| 2 | BB McCullum | 27-Sep-81 | Right_Hand | Right-arm medium | No Value |
| 3 | RT Ponting | 19-Dec-74 | Right_Hand | Right-arm medium | No Value |
| 4 | DJ Hussey | 15-Jul-77 | Right_Hand | Right-arm offbreak | Australia |
| 5 | Mohammad Hafeez | 17-Oct-80 | No Value | Right-arm offbreak | Pakistan |
| 6 | R Dravid | 11-Jan-73 | No Value | Right-arm offbreak | India |
| 7 | W Jaffer | 16-Feb-78 | No Value | Right-arm offbreak | India |
| 8 | V Kohli | 5-Nov-88 | No Value | Right-arm medium | India |
| 9 | JH Kallis | 16-Oct-75 | No Value | Right-arm fast-medium | South Africa |
| 10 | CL White | 18-Aug-83 | Right_Hand | Legbreak googly | Australia |
| 11 | MV Boucher | 3-Dec-76 | Right_Hand | Right-arm medium | South Africa |
| 12 | B Akhil | 7-Oct-77 | Right_Hand | Right-arm medium-fast | India |
| 13 | AA Noffke | 30-Apr-77 | Right_Hand | Right-arm fast-medium | Australia |
| 14 | P Kumar | 2-Oct-86 | Right_Hand | Right-arm medium | India |
| 15 | Z Khan | 7-Oct-78 | Right_Hand | Left-arm fast-medium | India |
karthick@HP-NOTEBOOK:~/Documents$ _
```

Display CSV File With Csvlook Utility

2.3. Python Pretty Table

If you have the python [prettytable](#) module installed, then you can run the following one-liner and redirect the CSV file to generate the table.

```
python -c "import sys,prettytable; print(prettytable.from_csv(sys.stdin))" < player_cleaned.csv
```

You can also create an [alias](#) for the one-liner and pass the file name as an argument.

```
$ alias ptable='python -c "import sys,prettytable; print(prettytable.from_csv(sys.stdin))"'
```

```
$ ptable < player_cleaned.csv
```



```

karthick@HP-NOTEBOOK:~/Documents$ alias ptable='python -c "import sys,prettytable; print(prettytable.from_csv(sys.stdin))"'
karthick@HP-NOTEBOOK:~/Documents$ ptable < player_cleaned.csv
+-----+-----+-----+-----+-----+-----+
| PLAYER_ID | PLAYER_NAME | DOB | BATTING HAND | BOWLING SKILL | COUNTRY |
+-----+-----+-----+-----+-----+-----+
| 1 | SC Ganguly | 8-Jul-72 | Left_Hand | Right-arm medium | No Value |
| 2 | BB McCullum | 27-Sep-81 | Right_Hand | Right-arm medium | No Value |
| 3 | RT Ponting | 19-Dec-77 | Right_Hand | Right-arm medium | No Value |
| 4 | DJ Hussey | 15-Jul-74 | Right_Hand | Right-arm offbreak | Australia |
| 5 | Mohammad Hafeez | 17-Oct-80 | No Value | Right-arm offbreak | Pakistan |
| 6 | R Dravid | 11-Jan-73 | No Value | Right-arm offbreak | India |
| 7 | W Jaffer | 16-Feb-78 | No Value | Right-arm offbreak | India |
| 8 | V Kohli | 5-Nov-88 | No Value | Right-arm medium | India |
| 9 | JH Kallis | 16-Oct-75 | No Value | Right-arm fast-medium | South Africa |
| 10 | CL White | 18-Aug-83 | Right_Hand | Legbreak googly | Australia |
| 11 | MV Boucher | 3-Dec-76 | Right_Hand | Right-arm medium | South Africa |
| 12 | B Akhil | 7-Oct-77 | Right_Hand | Right-arm medium-fast | India |
| 13 | AA Noffke | 30-Apr-77 | Right_Hand | Right-arm fast-medium | Australia |
| 14 | P Kumar | 2-Oct-86 | Right_Hand | Right-arm medium | India |
| 15 | Z Khan | 7-Oct-78 | Right_Hand | Left-arm fast-medium | India |
+-----+-----+-----+-----+-----+-----+
karthick@HP-NOTEBOOK:~/Documents$ _

```

Display CSV File With PrettyTable Module

3. Grabbing Data From CSV File

3.1. Print Row & Column Count

To get the number of columns in the CSV file, run the following command. Here the variable `NF` represents the number of fields split by a comma as the delimiter.

```

$ awk -F, 'END{print NF}' player_cleaned.csv
6

```

To get the number of rows, run the following command. Here the variable `NR` represents the current record (i.e) each line is considered as one record.

```

$ awk -F, 'END{print NR}' player_cleaned.csv
16

```

To skip the first line (header) and calculate the number of lines, run the following command.

```

$ awk -F, 'END{print NR-1}' player_cleaned.csv
15

```

3.2. Print Entire CSV File

This is pretty simple. You can use `cat` or `awk` to print the entire CSV file.

```

$ cat player_cleaned.csv

```

```

$ awk '{print}' player_cleaned.csv

```

3.3. Print Only Header From CSV File



Printing the header alone will give you a nice overview of what type of data your CSV file holds. You can use the `head` or `awk` command to grab the header alone.

```
$ head -n 1 player_cleaned.csv
```

```
$ awk 'NR==1' player_cleaned.csv
```

```
PLAYER_ID,PLAYER_NAME,DOB,BATTING HAND,BOWLING SKILL,COUNTRY
```

3.4. Exclude Header Line

To exclude the header line and print all other lines use the `awk` command. The `awk` variable `NR > 1` will make the first line to be skipped.

```
$ awk '(NR>1)' player_cleaned.csv
```

```
karthick@HP-NOTEBOOK:~/Documents$ awk '(NR>1)' player_cleaned.csv | column -t -s,
1  SC Ganguly      8-Jul-72   Left_Hand  Right-arm medium    No Value
2  BB McCullum     27-Sep-81  Right_Hand Right-arm medium    No Value
3  RT Ponting      19-Dec-74  Right_Hand Right-arm medium    No Value
4  DJ Hussey       15-Jul-77  Right_Hand Right-arm offbreak  Australia
5  Mohammad Hafeez 17-Oct-80  No Value   Right-arm offbreak  Pakistan
6  R Dravid        11-Jan-73  No Value   Right-arm offbreak  India
7  W Jaffer        16-Feb-78  No Value   Right-arm offbreak  India
8  V Kohli         5-Nov-88   No Value   Right-arm medium    India
9  JH Kallis       16-Oct-75  No Value   Right-arm fast-medium South Africa
10 CL White       18-Aug-83  Right_Hand Legbreak googly    Australia
11 MV Boucher     3-Dec-76   Right_Hand Right-arm medium    South Africa
12 B Akhil        7-Oct-77   Right_Hand Right-arm medium-fast India
13 AA Noffke      30-Apr-77  Right_Hand Right-arm fast-medium Australia
14 P Kumar        2-Oct-86   Right_Hand Right-arm medium    India
15 Z Khan         7-Oct-78   Right_Hand Left-arm fast-medium India
karthick@HP-NOTEBOOK:~/Documents$ _
```

Awk - Exclude Header Line

`Sed` can also be used to exclude the first line and print all other lines. The `1d` flag will delete the first line and print all other lines to stdout (Terminal).

```
$ sed 1d < player_cleaned.csv
```



```

karthick@HP-NOTEBOOK:~/Documents$ sed 1d < player_cleaned.csv | column -t -s,
1  SC Ganguly      8-Jul-72  Left_Hand  Right-arm medium      No Value
2  BB McCullum     27-Sep-81 Right_Hand  Right-arm medium      No Value
3  RT Ponting      19-Dec-74 Right_Hand  Right-arm medium      No Value
4  DJ Hussey       15-Jul-77 Right_Hand  Right-arm offbreak    Australia
5  Mohammad Hafeez 17-Oct-80 No Value   Right-arm offbreak    Pakistan
6  R Dravid        11-Jan-73 No Value   Right-arm offbreak    India
7  W Jaffer        16-Feb-78 No Value   Right-arm offbreak    India
8  V Kohli         5-Nov-88  No Value   Right-arm medium      India
9  JH Kallis       16-Oct-75 No Value   Right-arm fast-medium South Africa
10 CL White       18-Aug-83 Right_Hand  Legbreak googly      Australia
11 MV Boucher     3-Dec-76  Right_Hand  Right-arm medium      South Africa
12 B Akhil        7-Oct-77  Right_Hand  Right-arm medium-fast India
13 AA Noffke      30-Apr-77 Right_Hand  Right-arm fast-medium Australia
14 P Kumar        2-Oct-86  Right_Hand  Right-arm medium      India
15 Z Khan         7-Oct-78  Right_Hand  Left-arm fast-medium  India
karthick@HP-NOTEBOOK:~/Documents$ _

```

Sed - Exclude Header Line

3.5. Print Particular Columns

We can use the column position to print the entire column. There are two approaches to achieve this. The first approach will be to use **awk** and the second approach will be to use **loops**. Awk will be much simpler to grab the column.

Awk by default splits the line based on the delimiter and stores the values in \$1, \$2, \$3, etc. The default delimiter for awk is **white space**.

Take a look at the below snippet where the field separator(FS=",") and output field separator(OFS=",") is set to comma. The print statement will print the first column, second column, and sixth column.

```

awk 'BEGIN{FS=",";OFS=","}
{
    print $1,$2,$6
}' player_cleansed.csv

```

You can write the above snippet in one-liner too.

```

awk 'BEGIN{FS=",";OFS=","}{print $1,$2,$6}' player_cleansed.csv

```

```

karthick@HP-NOTEBOOK:~/Documents$ awk 'BEGIN{FS=",";OFS=","}{print $1,$2,$6}' player_cleaned.csv | head -n 5
PLAYER_ID,PLAYER_NAME,COUNTRY
1,SC Ganguly,No Value
2,BB McCullum,No Value
3,RT Ponting,No Value
4,DJ Hussey,Australia
karthick@HP-NOTEBOOK:~/Documents$ _

```

Print Specific Columns

Now the second approach would be to use loops.

```

IFS=","
while read -r -a fields

```

^

```
do
    echo ${fields[0]},${fields[1]},${fields[5]}
done < player_cleaned.csv
```

Let me explain what exactly happens when you run the above snippet.

- We are setting the Internal field separator IFS to comma.
- Using the read command we are creating an array named "fields" and redirecting the input file to the while loop.
- For each iteration, it will read line by line and store the line as array elements in "fields" so you can use the array index position to grab the particular column alone.

Note: Index value starts from 0..N

3.6. Print Row That Matches The Condition

If you wish to print the rows that match a certain condition, then you can do it easily using `awk`. Let's go over a few scenarios.

To print all the rows that match a value in a column, run the following command. Here I am trying to print all rows that match the value "India" in column 6.

```
$ awk -F , '$6 == "India"' player_cleaned.csv
```

```
karthick@HP-NOTEBOOK:~/Documents$ awk -F , '$6 == "India"' player_cleaned.csv | column -t -s,
6  R Dravid  11-Jan-73  No Value  Right-arm offbreak  India
7  W Jaffer  16-Feb-78  No Value  Right-arm offbreak  India
8  V Kohli   5-Nov-88  No Value  Right-arm medium    India
12 B Akhil   7-Oct-77  Right_Hand Right-arm medium-fast India
14 P Kumar   2-Oct-86  Right_Hand Right-arm medium    India
15 Z Khan    7-Oct-78  Right_Hand Left-arm fast-medium India
karthick@HP-NOTEBOOK:~/Documents$ _
```

Conditional Match

To print all rows that do not match a certain value, run the following command. Instead of an **equality operator**, we are using **not equal operator**.

```
$ awk -F , '$6 != "India"' player_cleaned.csv
```

```
karthick@HP-NOTEBOOK:~/Documents$ awk -F , ',$6 != "India"' player_cleaned.csv | column -t -s,
PLAYER_ID  PLAYER_NAME  DOB      BATTING HAND  BOWLING SKILL  COUNTRY
1          SC Ganguly    8-Jul-72  Left_Hand     Right-arm medium  No Value
2          BB McCullum   27-Sep-81 Right_Hand     Right-arm medium  No Value
3          RT Ponting    19-Dec-74 Right_Hand     Right-arm medium  No Value
4          DJ Hussey     15-Jul-77 Right_Hand     Right-arm offbreak  Australia
5          Mohammad Hafeez 17-Oct-80 No Value      Right-arm offbreak  Pakistan
9          JH Kallis     16-Oct-75 No Value      Right-arm fast-medium  South Africa
10         CL White      18-Aug-83 Right_Hand     Legbreak googly  Australia
11         MV Boucher    3-Dec-76 Right_Hand     Right-arm medium  South Africa
13         AA Noffke     30-Apr-77 Right_Hand     Right-arm fast-medium  Australia
```

Inverse Condition

You can also do a condition check on more than one column using logical AND, logical OR operator. Let's say I want to check all the rows that have the country as "India" and the batting hand as "Right_hand".

Here, \$4 points to the 4th column and \$6 points to the 6th column. The symbol && is used as a logical AND operator to evaluate two conditions.

```
$ awk -F , '$4 == "Right_Hand" && $6 == "India"' player_cleaned.csv
```

```
karthick@HP-NOTEBOOK:~/Documents$ awk -F , '$4 == "Right_Hand" && $6 == "India"' player_cleaned.csv | column -s, -t
12 B Akhil 7-Oct-77 Right_Hand Right-arm medium-fast India
14 P Kumar 2-Oct-86 Right_Hand Right-arm medium India
15 Z Khan 7-Oct-78 Right_Hand Left-arm fast-medium India
```

Multiple Conditional Check

If you wish to include the header along with the result from the conditional check, use the following command. First I am printing the first line using NR==1, then using the logical AND operator running the conditional check to print the results.

```
$ awk 'NR==1' player_cleaned.csv && awk -F , '$4 == "Right_Hand" && $6 == "India"' player_cleaned.csv
```

If you wish to print or redirect the output, then run the entire command inside a subshell by enclosing it with brackets.

```
$ (awk 'NR==1' player_cleaned.csv && awk -F , '$4 == "Right_Hand" && $6 == "India"' player_cleaned.csv)
```

```
karthick@HP-NOTEBOOK:~/Documents$ (awk 'NR==1' player_cleaned.csv && awk -F , '$4 == "Right_Hand" && $6 == "India"' player_cleaned.csv) | column -t -s,
PLAYER_ID  PLAYER_NAME  DOB      BATTING HAND  BOWLING SKILL  COUNTRY
12         B Akhil      7-Oct-77 Right_Hand     Right-arm medium-fast India
14         P Kumar      2-Oct-86 Right_Hand     Right-arm medium India
15         Z Khan       7-Oct-78 Right_Hand     Left-arm fast-medium India
```

Conditional Check - Header Included

A note about Csvkit

So far whatever we have seen in this article is simple and straightforward. But when your CSV file has a complex structure, then it becomes tedious to parse using the above approach. There is a utility called **CSVKIT**, which is an excellent utility to work with CSV files in bash.

The problem with the csvkit utility is it is installed by default in your distribution and you might have to manually install it. In your corporate environment, this may not be possible since there may be some restrictions to installing external packages. But this utility is worth the mention and we will create a separate detailed article for it.

Conclusion

In this guide, we have seen how to work with CSV files using awk, sed. You can also use other utilities like cut, grep, tr, etc to get the desired result but awk and sed will make your life simpler and reduce the complexity of writing a lot of codes. If you have any feedback do mention it in the comment section and we will be happy to hear it from you.

Similar Read:

- [Bash Scripting – Parse Arguments In Bash Scripts Using getopt](#)
- [How To Parse And Pretty Print JSON With Linux Commandline Tools](#)

AWK

BASH

BASH SCRIPTING

BASH TIPS

BASH TUTORIAL

CLI

COMMA-SEPARATED VALUES

COMMAND LINE

CSV

LEARN SHELL SCRIPTING

PARSE CSV FILES

SED

SHELL SCRIPTING

SHELL SCRIPTS

 2 comments5       

KARTHICK

Karthick is a passionate software engineer who loves to explore new technologies. He is a public speaker and loves writing about technology especially about Linux and opensource.



Previous post

How To Move Home Directory To New Partition Or Disk In Linux

YOU MAY ALSO LIKE

Bash Scripting – Conditional Statements

October 21, 2021

Bash Redirection Explained With Examples

August 27, 2021

Bash Echo Command Explained With Examples In Linux

September 1, 2021

