SOFTENGER
Strive to Excel

Use of Awk

Unix Fundamentals

WHAT CAN WE DO WITH AWK ?

1. AWK Operations:
   - (a) Scans a file line by line
   - (b) Splits each input line into fields
   - (c) Compares input line/fields to pattern
   - (d) Performs action(s) on matched lines
2. Useful For:
   - (a) Transform data files
   - (b) Produce formatted reports
3. Programming Constructs:
   - (a) Format output lines
   - (b) Arithmetic and string operations
   - (c) Conditionals and loops

Syntax and options

- Syntax:

awk options 'selection _criteria {action }' files

- Sample Commands
  - ❖Example:
    Consider the following text file as the input file for all cases below.
  - ❖$ cat > emp.txt

• Ajay	manager	account	45000
• sunil	clerk	account	25000
• varun	manager	sales	50000
• amit	manager	account	47000
• tarun	peon	sales	15000
• deepak	clerk	sales	23000

Default behaviour of AKW

- Default behavior of Awk : By default Awk prints every line of data from the specified file.

```
$ awk '{print}' emp.txt
```

Output:

```
ajay    manager     account     45000
sunil   clerk       account     25000
varun   manager     sales       50000
amit    manager     account     47000
tarun   peon        sales       15000
deepak  clerk       sales       23000
sunil   peon        sales       13000
satvik  director  purchase  80000
```

In the above example, no pattern is given. So the actions are applicable to all the lines. Action print without any argument prints the whole line by default, so it prints all the lines of the file without failure.

Awk example

To Print the lines which matches with the given pattern.

```
$ awk '/manager/ {print}' emp.txt Output:
ajay      manager       account      45000
varun     manager       sales        50000
amit      manager       account      47000
```

In the above example, the awk command prints all the line which matches with the ‘manager’.

Awk Example

Splitting a Line Into Fields :

For each record i.e line, the awk command splits the record delimited by whitespace character by default and stores it in the $n variables. If the line has 4 words, it will be stored in $1, $2, $3 and $4 respectively. Also, $0 represents the whole line.

```
$ awk '{print $1,$4}' emp.txt
```

Output:

```
ajay    45000
sunil   25000
varun   50000
amit    47000
tarun   15000
```

In the above example, $1 and $4 represents Name and Salary fields respectively.

Built In Variables In Awk

- Awk’s built-in variables include the field variables—$1, $2, $3, and so on ($0 is the entire line) — that break a line of text into individual words or pieces called fields.
- NR: NR command keeps a current count of the number of input records. Remember that records are usually lines. Awk command performs the pattern/action statements once for each record in a file.
- NF: NF command keeps a count of the number of fields within the current input record.

Use of NR

- NR: NR command keeps a current count of the number of input records. Remember that records are usually lines. Awk command performs the pattern/action statements once for each record in a file.
- Examples:

Use of NR built-in variables (Display Line Number)

```
$ awk '{print NR,$0}' emp.txt
```

Output:

```
1 ajay    manager      account      45000
2 sunil   clerk        account      25000
3 varun manager        sales        50000
4 amit    manager      account      47000
5 tarun  peon          sales        15000 s
6 deepak clerk         sales        23000
```

In the above example, the awk command with NR prints all the lines along with the line number.

Use of NR in Awk

Another use of NR built-in variables (Display Line From 3 to 6)

```
$ awk 'NR==3, NR==6 {print NR,$0}' emp.txt
```

Output:

```
3 varun         manager         sales     50000
4 amit          manager         account 47000
5 tarun         peon            sales     15000
6 deepak        clerk           sales     23000
```

Use of NF

Use of NF built-in variables (Display Last Field)

```
$ awk '{print $1,$NF}' emp.txt Output:
ajay      45000
sunil     25000
varun     50000
amit      47000
tarun     15000
deepak    23000
sunil     13000
satvik 80000
```

In the above example $1 represents Name and $NF represents Salary. We can get the Salary using $NF , where $NF represents last field.

More example of Awk

More Examples

For the given text file:

```
$cat > SIPL.txt
A  B        C
Tarun              A12               1
Man                B6                2
Praveen  M42               3
```

1) To print the first item along with the row number(NR) separated with ” – “ from each line in SIPL.txt:

```
$ awk '{print NR "- " $1 }' SIPL.txt
1 - Tarun
2 – Manav
3 - Praveen
```

More example of Awk

2) To return the second row/item from SIPL.txt:

```
$ awk '{print $2}' SIPL.txt
A12
B6
M42
```

3) To find the length of the longest line present in the file:

```
  $ awk '{ if (length($0) > max) max = length($0) } END { print max }' SIPL.txt
13
```

4) To count the lines in a file:

```
$ awk 'END { print NR }' SIPL.txt
3
```

More Examples of Awk

5) Printing lines with more than 10 characters:

```
$ awk 'length($0) > 10' SIPL.txt
Tarun           A12     1
Praveen  M42    3
```

6) To find/check for any string in any column:

```
$ awk '{ if($3 == "B6") print $0;}' SIPL.txt
```

7) To print the squares of first numbers from 1 to n say 5:

```
$ awk 'BEGIN { for(i=1;i<=5;i++) print "square of", i, "is",i*i; }' square of 1 is 1
   square of 2 is 4
   square of 3 is 9
   square of 4 is 16
   square of 5 is 25
```