Complete AWK Scripting

Intro

AWK is a powerful text processing tool in Linux used for pattern scanning and data extraction.

Linux AWK Syntax

Syntax:
awk 'pattern {action}' file
Example:
awk '{print $1}' file.txt

How AWK command works?

AWK reads input line by line, splits into columns (fields), and performs actions.
Example:
awk '{print NR, $0}' file.txt

Examples of AWK command

Print column:
awk '{print $2}' file.txt
Print last column:
awk '{print $NF}' file.txt

AWK working with CSV

Definition: AWK uses -F to define delimiter.
Script:
awk -F',' '{print $1,$3}' emp.csv

AWK with multiple delimiter

Definition: AWK supports multiple delimiters.
Script:
awk -F'[:,|]' '{print $1}' file.txt

AWK working with other commands

Definition: AWK can combine with pipes.
Script:
ls -l | awk '{print $9}'

Log analysis in given range of time

Definition: Filter logs based on time.
Script:
awk '$3>="10:00" && $3<="11:00"' log.txt

Replace a word using AWK

Definition: Substitute function replaces text.
Script:
awk '{gsub("error","warning"); print}' file.txt

AWK Scripting BEGIN, END

Definition:
BEGIN executes before input
END executes after input
Script:
awk 'BEGIN{print "Start"} {print $0} END{print "End"}' file.txt

AWK conditions using if-else

Definition: Apply conditions.
Script:
awk '{if($3>50000) print "High"; else print "Low"}' emp.txt

Using AWK patterns in a file

Definition: Pattern matches specific text.
Script:
awk '/error/' log.txt

AWK Real Industry Commands Guide

Print only a given column

awk '{print $2}' file.txt

Print last column

awk '{print $NF}' file.txt

Search a word

awk '/error/' file.txt

Print only line no. 5

awk 'NR==5' file.txt

Print line number at start

awk '{print NR, $0}' file.txt

Print range of lines (3 to 6)
```
awk 'NR>=3 && NR<=6' file.txt
```

Get line no. of empty lines
```
awk 'NF==0 {print NR}' file.txt
```

Search multiple words
```
awk '/error|fail/' file.txt
```

Ignore case while searching
```
awk 'BEGIN{IGNORECASE=1} /error/' file.txt
```

Check if char present in column
```
awk '$2 ~ /a/' file.txt
```

Work with CSV file
```
awk -F',' '{print $1,$3}' file.csv
```

Salary more than 50k
```
awk '$3>50000' emp.txt
```

Multiple delimiter
```
awk -F'[:|]' '{print $1}' file.txt
```

Get service status
```
systemctl status sshd | awk '/Active/ {print $2}'
```

Get list of files
```
ls -l | awk '{print $9}'
```

Read logs in time range
```
awk '$3>="10:00" && $3<="11:00"' log.txt
```

Files modified in Oct
```
ls -l | awk '$6=="Oct"'
```

Replace a word
```
sed 's/error/warning/g' file.txt
```

Length of line
```
awk '{print length($0)}' file.txt
```

Index of word
```
awk '{print index($0,"error")}' file.txt
```

Upper case

```
awk '{print toupper($0)}' file.txt
```

Lower case

```
awk '{print tolower($0)}' file.txt
```

Total salary

```
awk '{sum+=$3} END {print sum}' emp.txt
```

Average salary

```
awk '{sum+=$3} END {print sum/NR}' emp.txt
```

Count lines

```
awk 'END {print NR}' file.txt
```

Ignore header count users

```
awk 'NR>1' file.txt | wc -l
```

Longest line length

```
awk '{ if(length>max) max=length } END {print max}' file.txt
```

Print HIGH or LOW salary

```
awk '{if($3>50000) print "HIGH"; else print "LOW"}' emp.txt
```

Total salary in Loan dept

```
awk '$2=="Loan" {sum+=$3} END {print sum}' emp.txt
```