

REFERENCIA RÁPIDA DE AWK

Patrones, campos, arreglos, funciones, procesamiento de texto

Fundamentos

Ejecutar AWK

```
awk '{ print }' file.txt # imprime cada línea
awk '{ print $1 }' file.txt # imprime el primer campo
awk -F '{ print $1 }' /etc/passwd # delimitador personalizado
awk -f script.awk file.txt # ejecutar desde archivo
cmd | awk '{ print $2 }' # entrada desde pipe
```

Estructura del Programa

awk 'pattezn { action }' Forma básica — la acción se ejecuta cuando el patrón coincide

BEGIN { ... } Se ejecuta una vez antes de procesar la entrada

END { ... } Se ejecuta una vez después de procesar toda la entrada

Sin patrón La acción se ejecuta para cada línea

Sin acción La acción por defecto es `{ print }`

Patrones y Acciones

Tipos de Patrones

```
awk '/error/' file.txt # coincidencia regex
awk '$3 > 100' file.txt # comparación
awk 'NR >= 5 && NR <= 10' file.txt # rango de líneas
awk '/start/,/end/' file.txt # patrón de rango
```

Referencia de Patrones

/regex/ Comparar línea con regex
\$1 ~ /pat/ Campo coincide con regex
\$1 !~ /pat/ Campo no coincide con regex
expr1, expr2 Rango: desde primera hasta segunda coincidencia
expr1 && expr2 AND lógico
expr1 || expr2 OR lógico
!expr NOT lógico

Variables

Variables Integradas

NR Número de registro (línea) actual
NF Número de campos en el registro actual
FS Separador de campo de entrada (por defecto: espacio)
OFS Separador de campo de salida (por defecto: espacio)
RS Separador de registro de entrada (por defecto: nueva línea)
ORS Separador de registro de salida (por defecto: nueva línea)
FILENAME Nombre del archivo de entrada actual
FNR Número de registro en el archivo actual

Variables de Usuario

```
awk '{ total += $1 } END { print total }' file.txt
awk -v threshold=50 '$1 > threshold' file.txt
awk 'BEGIN { count = 0 } /pat/ { count++ } END { print count }' file.txt
```

Campos

Acceso a Campos

\$0 Línea completa actual
\$1, \$2, ... Primer, segundo, ... campo
\$NF Último campo
\$(NF-1) Penúltimo campo

Separadores de Campo

```
awk -F '{ print $2 }' data.csv # coma
awk -F '\t' '{ print $1 }' data.tsv # tabulación
awk 'BEGIN { FS = "[,;]" } { print $1 }' f # multi-caracter
awk 'BEGIN { OFS = "," } { print $1, $3 }' f # sep salida
```

Control de Flujo

Condicionales y Bucles

```
awk '{ if ($1 > 50) print "high"; else print "low" }' f
awk 'for (i = 1; i <= NF; i++) print $i' f
awk '{ i = 1; while (i <= NF) { print $i; i++ } }' f
awk '/skip/ { next } { print }' f # omitir líneas coincidentes
```

Instrucciones de Control

if (cond) { ... } else { ... } Condicional
for (i = 0; i < n; i++) { ... } Bucle for estilo C
for (key in array) { ... } Iterar claves de arreglo
while (cond) { ... } Bucle while
do { ... } while (cond) Bucle do-while
next Saltar al siguiente registro de entrada
exit Detener procesamiento, ejecutar bloque END

Funciones

Funciones Definidas por el Usuario

```
awk 'function max(a, b) {
    return (a > b) ? a : b
}
{ print max($1, $2) }' file.txt
```

Funciones Numéricas

int(x) Truncar a entero
sqrt(x) Raíz cuadrada
sin(x), cos(x) Funciones trigonométricas
log(x), exp(x) Logaritmo natural y exponencial
rand() Flotante aleatorio entre 0 y 1
srand(seed) Inicializar generador de números aleatorios

Arreglos

Arreglos Asociativos

```
awk '{ count[$1]++ }
    END { for (k in count) print k, count[k] }' f
awk '{ arr[NR] = $0 }
    END { for (i = NR; i >= 1; i--) print arr[i] }' f
```

Operaciones con Arreglos

arr[key] = val Asignar elemento

arr[key] Obtener elemento (se crea automáticamente al acceder)

key in arr Comprobar si la clave existe

delete arr[key] Eliminar elemento individual

delete arr Eliminar arreglo completo

for (k in arr) Iterar sobre claves (sin orden)

length(arr) Número de elementos (gawk)

Funciones de Cadena

Referencia de Cadenas

length(s) Longitud de cadena
substr(s, start, len) Subcadena (indexada desde 1)
index(s, target) Posición de target en s (0 si no se encuentra)
split(s, arr, sep) Dividir cadena en arreglo
sub(pat, repl, s) Reemplazar primera coincidencia
gsub(pat, repl, s) Reemplazar todas las coincidencias
match(s, pat) Posición de coincidencia regex (establece RSTART, RLENGTH)
tolower(s) / toupper(s) Conversión de mayúsculas/minúsculas
sprintf(fmt, ...) Formatear cadena (como printf de C)

Ejemplos de Cadenas

```
awk '{ gsub(/old/, "new"); print }' f # reemplazo tipo sed
awk '{ print toupper($0) }' f # todo en mayúsculas
awk '{ print substr($0, 1, 40) }' f # truncar a 40
```

E/S

Salida

```
awk '{ print $1, $2 }' f # separado por espacio
awk '{ printf "%s,%d\n", $1, $2 }' f # salida formateada
awk '{ print $1 > "out.txt" }' f # redirigir a archivo
awk '{ print $1 >> "out.txt" }' f # añadir a archivo
```

Referencia de E/S

print Imprimir con ORS (nueva línea por defecto)
printf fmt, ... Impresión formateada (sin nueva línea al final)
print > file Redirigir salida a archivo
print >> file Añadir salida a archivo
print | cmd Enviar salida a un comando por pipe
getline < file Leer una línea del archivo
cmd | getline var Leer salida de comando en variable
close(file) Cerrar archivo o pipe

Patrones Comunes

One-Liners

```
awk '{ sum += $1 } END { print sum }' f # sumar columna
awk 'END { print NR }' f # contar líneas
awk '!seen[$0]++' f # eliminar duplicados
awk 'NF' f # eliminar líneas vacías
awk '{ print NF }' f # campos por línea
```

Recetas

CSV a TSV ``awk -F,'BEGIN{OFS="\t"} {$1=$1; print}'``
Sumar columna 2 ``awk '{s += $2} END {print s}'``
Primeras N líneas ``awk 'NR <= 10'` (como head)
Conteo de frecuencia `awk '{c[$1]++} END {for (k in c) print k, c[k]}'`
Entre marcadores `awk '/BEGIN/,/END/'`
Imprimir campo N `awk '{print $N}'` (reemplazar N)`