The Tenth International Conference on Advances in Databases, Knowledge, and Data Applications

Mai 20 - 24, 2018 - Nice/France

Powerful Unix-Tools - sed and awk

Andreas Schmidt

Department of Informatics and Business Information Systems University of Applied Sciences Karlsruhe Germany

Institute for Automation and Applied Informatics Karlsruhe Institute of Technologie Germany

String Substitution with sed

- sed - <u>S</u>tream <u>E</u>ditor
- non interactiv, controlled by a script
- line oriented text processing
- short scripts are typically given as parameter (-e option),
  longer scripts as files (-f option)
- Possible operations: Insert, Substitude, Delete, Append, Change, Print, Delete
- Commands in script can take an optional *address*, specifying the line(s) to be per-formed.
- *Address* can be a single line number or a regular expression
- *Address* can be an interval (start, stop)
- A loop executes script commands on each input line
- Default behavior: printing each processed line to STDOUT (suppress with: -n)

sed commands

- **s**: substitude
  - Replace all occurences of D with GER
    ```
    sed 's/\bD\b/GER/g' city.csv > city2.csv
    ```
  - Replace „Stuttgart" with „Stuttgart am Neckar" (extended regexp)
    ```
    sed -r '/Stuttgart/ s/^([A-Za-z]+)/\1 am Neckar/' city.csv
    ```
  - Replace all occurences of NULL in a line with \N (Inplace Substitution)
    ```
    sed -i 's/\bNULL\b/\\N/g' city.csv
    ```

- **p**: print (typically used with default printing behaviour off (-n option))
  - print from line 10 to 20 (resp.: 5-10, 23, 56-71)
    ```
    sed -n 10,20p city.csv
    sed -n '5,10p;23p;56,71p' city.csv
    ```
  - print lines starting from dataset about 'Sapporo' inclusive dataset about 'Seattle'
    ```
    sed -n '/^Sapporo/,/^Seattle/p' city.csv
    ```

sed commands

- **i**: insert
  - Insert dataset about Karlsruhe at line 2

    ```
    sed '2i Karlsruhe,D,"Baden Wuerttemberg",301452,49.0,6.8' city.csv
    ```

- **d**: delete
  - delete Aachen (inplace)

    ```
    sed -i '/Aachen/ d' city.csv
    ```
  - delete all empty lines

    ```
    sed '/^ *$/d' The-Adventures-of-Tom-Sawyer.txt
    ```
  - delete lines 2-10

    ```
    sed '2,10d' city.csv
    ```
  - delete all <script>..</script> sections in a file

    ```
    sed -Ei '/<script>/,/<\/script>/d' jaccard.html
    ```
  - delete from `<h2>Navigation menu</h2>` to end of file

    ```
    sed -i '/<h2>Navigation menu<\/h2>/,$d' jaccard.html
    ```

sed commands

- **c**: change
  - Replace entry of Biel

    ```
    sed '/^Biel\b/ c Biel,CH,BE,53308,47.8,7.14' city.csv
    ```

- **a**: append
  - Underline each CHAPTER

    ```
    sed '/^CHAPTER/ a ------------' The-Adventures-of-Tom-Sawyer.txt
    ```

- ...

awk

- like sed, but with powerful programming language
- filter and report writer
- ideal for processing rows and columns
- suport for associative arrays
- structure: pattern { action statements }
- special BEGIN, END pattern match **before** the first line is read and **after** the last line is read
- Access to column values via $1, $2, ... variables ($0: whole line)
- Examples:

```
awk -F, '$3=="Bayern" && $4 < 1000000 { print $1", "$4 }' city.csv
```

field delimiter

pattern

action statement

awk

- Calculating average population

see slide 13.

```
awk -F, -f average.awk  city.csv
```

```
# script: average.awk
BEGIN { sum = 0
        num = 0
}


$4!="NULL" {
    sum += $4
    num++
}


END { print "Average population: "sum/num }
```

pattern

awk

- Distribute entries in multiple files (all entries with same value in the first column, should go to the same file)

- awk-file:

append semantic

```
{
    printf "%06d\n", $2 >> "terms/"$1".dat"
}
```

output file

- Call:

```
awk -F':' -f file-redirection-example.awk data.txt'
```

input format (data.txt):
<string>:<integer>

- predefined variables
  - NF: number of fields
  - NR: number of records
  - FS: field separator (default: " ", same as -F from command line)
  - RS: record separator (default: \n)
  - ORS: output record separator
  - FPAT: Field pattern (alternative way to specify a field instead of use of FS)
  - FILENAME

awk example: multi-line input

- Input-file:

```
Andreas Schmidt
KIT
Germany

Petre Dini
IARIA
USA

Fritz Laux
University Reutlingen
Germany
```

- Output (STDOUT):

```
Andreas Schmidt,KIT,Germany
Petre Dini,IARIA,USA
Fritz Laux,University Reutlingen,Germany
```

awk -f demo2.awk address.txt

```
BEGIN {
    FS="\n"
    RS="\n\n"
    ORS=""
}
{
    for (i=1; i<NF; i++) {
        print $i","
    }
    print $NF"\n"
}
```

becomes $3

awk

- FPAT: Split a line by pattern, rather then by delimiter

- Example:
  - File:

    ```
    12,45,Test, 123.56
    13,21,"Test without comma", 345.2
    14,71,"Test, with comma", 0.7
    ```
  - Command:

    ```
    awk -F, '{print $1" : "$3}' fpat-demo.txt
    ```
  - Output:

    ```
    12 : Test
    13 : "Test without comma"
    14 : "Test
    ```
    WRONG!!!!! ←

- Example with FPAT:
  - command:

    strings, containing
    no comma

    matches "..."
    (more specific rule)

    ```
    awk 'BEGIN{FPAT = "([^,]*)|(\"[^\"]*\")"}  \
        {print $1" : "$3}' fpat-demo.txt
    ```

  - Output:

    ```
    12 : Test
    13 : "Test without comma"
    14 : "Test, with comma"
    ```

Example file: city.csv

```
Aachen,D,"Nordrhein Westfalen",247113,NULL,NULL
Aalborg,DK,Denmark,113865,10,57
Aarau,CH,AG,NULL,NULL,NULL
Aarhus,DK,Denmark,194345,10.1,56.1
Aarri,WAN,Nigeria,111000,NULL,NULL
Aba,WAN,Nigeria,264000,NULL,NULL
Abakan,R,"Rep. of Khakassiya",161000,NULL,NULL
Abancay,PE,Apurimac,NULL,NULL,NULL
Abeokuta,WAN,Nigeria,377000,NULL,NULL
Aberdeen,GB,Grampian,219100,NULL,NULL
Aberystwyth,GB,Ceredigion,NULL,NULL,NULL
Abidjan,CI,"Cote dIvoire",NULL,-3.6,5.3
```