FILTERS USING REGULAR EXPRESSIONS – grep and sed

CONTENT

grep to seach a file for a pattern and display

grep options to display, count, line numbers or filenames

Regular expressions

Basic regular expressions (BRE)

Extended regular expressions (ERE)

sed to edit / manipulate an input stream

substitution features

repeated and remembered patterns

grep

- It scans the file / input for a pattern and displays lines containing the pattern, the line numbers or filenames where the pattern occurs
- It’s a command from a special family in UNIX for handling search requirements

  grep *options pattern filename(s)*

```
grep “sales” emp.lst
```

- Patterns with and without quotes is possible
- Its generally safe to quote the pattern
- Quote is mandatory when pattern involves more than one word
- It returns the prompt in case the pattern can’t be located

```
grep president emp.lst
```

- When grep is used with multiple filenames, it displays the filenames along with the output

```
grep “director” emp1.lst emp2.lst
```

Where it shows filename followed by the contents

grep options

-i	ignores case for matching
-v	doesn’t display lines matching expression
-n	displays line numbers along with lines
-c	displays count of number of occurrences
-l	displays list of filenames only

-e exp	specifies expression with this option
-x	matches pattern with entire line
-f file	takes pattrens from file, one per line
-E	treats pattren as an extended RE
-F	matches multiple fixed strings

1. grep -i ‘agarwal’ emp.lst
2. grep -v ‘director’ emp.lst > otherlist

   wc -l otherlist will display 11 otherlist
3. grep –n ‘marketing’ emp.lst
4. grep –c ‘director’ emp.lst
5. grep –c ‘director’ emp*.lst

   will print filenames prefixed to the line count

6. grep –l ‘manager’ *.lst

   will display filenames *only*

7. grep –e ‘Agarwal’ –e ‘aggarwal’ –e ‘agrawal’ emp.lst

   will print matching multiple patterns

8. grep –f pattern.lst emp.lst

   all the above three patterns are stored in a separate file *pattern.lst*

BASIC REGULAR EXPRESSIONS

- It is tedious to specify each pattern separately with the -e option
- grep uses an expression of a different type to match a group of similar patterns
- if an expression uses meta characters, it is termed a regular expression
- Some of the characters used by regular expression are also meaningful to the shell

BASIC AND EXTENDED REGULAR EXPRESSIONS

(BRE & ERE)

BRE character subset

*****	Zero or more occurrences
g*	nothing or g, gg, ggg, etc.
.	A single character
.*	nothing or any number of characters
[pqr]	a single character p, q or r
[c1-c2]	a single character within the ASCII range represented by c1 and c2

The character class

- grep supports basic regular expressions (BRE) by default and extended regular expressions (ERE) with the –E option
- A regular expression allows a group of characters enclosed within a pair of [ ], in which the match is performed for a single character in the group

grep “[aA]g[ar][ar]wal” emp.lst

- A single pattern has matched two similar strings
- The pattern [a-zA-Z0-9] matches a single alphanumeric character. When we use range, make sure that the character on the left of the hyphen has a lower ASCII value than the one on the right

Negating a class (^) (caret)

THE *

- * Zero or more occurrences of the previous character
- g* nothing or g, gg, ggg, etc.
- grep “[aA]gg*[ar][ar]wal” emp.lst
- Notice that we don’t require to use –e option three times to get the same output!!!!!

THE DOT

A dot matches a single character

.*	signifies any number of characters or none

```
grep “j.*saxena” emp.lst
```

^ and $

Most of the regular expression characters are used for matching patterns, but there are two that can match a pattern at the beginning or end of a line

^	for matching at the beginning of a line
$	for matching at the end of a line

```
grep “^2” emp.lst
```

Selects lines where emp_id starting with 2

```
grep “7…$” emp.lst
```

Selects lines where emp_salary ranges between 7000 to 7999

```
grep “^[^2]” emp.lst
```

Selects lines where emp_id doesn’t start with 2

When metacharacters lose their meaning

- It is possible that some of these special characters actually exist as part of the text
- Sometimes, we need to escape these characters

Eg: when looking for a pattern g*, we have to use \

To look for [, we use \[

To look for .*, we use \.*

EXTENDED RE (ERE)

- If current version of grep doesn’t support ERE, then use egrep but without the –E option
- -E option treats pattern as an ERE

+ matches one or more occurrences of the previous character

? Matches zero or one occurrence of the previous character b+ matches b, bb, bbb, etc.

b? matches either a single instance of b or nothing

These characters restrict the scope of match as compared to the *

```
grep –E “[aA]gg?arwal” emp.lst
```

```
# ?include +<stdio.h>
```

The ERE set

ch+	matches one or more occurrences of character ch
ch?	Matches zero or one occurrence of character ch
exp1|exp2	matches exp1 or exp2
(x1|x2)x3	matches x1x3 or x2x3

Matching multiple patterns

```
grep –E ‘sengupta|dasgupta’ emp.lst
```

We can locate both without using –e option twice,

or

```
grep –E ‘(sen|das)gupta’ emp.lst
```

More Metacharcters

RE Metacharacter	Matches…
^	**beginning of line**
$	**end of line**
\char	**Escape the meaning of *char* following it**
[^]	**One character <u>not</u> in the set**
\<	**Beginning of word anchor**
\>	**End of word anchor**
( ) or \\(\\)	**Tags matched characters to be used later (max = 9)**
| or \\\|	**Or grouping**
x\\{m\\}	**Repetition of character x, m times (x,m = integer)**
x\\{m,\\}	**Repetition of character x, at least m times**
x\\{m,n\\}	**Repetition of character x between m and m times**

SUMMARY

- BRE [ ], *, ., ^, $, \
- ERE ?, +, |, (,)
- sed: the stream editor

- THANK YOU