Text Management

Introduction and/or Background

Believe it or not in computing there was a time before spreadsheets and SQL databases. Fact in the early days the printed phone book was published using these tools! Everything was a tabular text file. Unix developed a series of text manipulation tools used to process them. Linux adopted them for compatibility. Here’s an example of text management in Linux:

```
OrderDate,Region,Rep,Item,Units,Unit Cost,Total
1/6/18,East,Jones,Pencil,95, 1.99 , 189.05
1/23/18,Central,Kivell,Binder,50, 19.99 , 999.50
2/9/18,Central,Jardine,Pencil,36, 4.99 , 179.64
2/26/18,Central,Gill,Pen,27, 19.99 , 539.73
3/15/18,West,Sorvino,Pencil,56, 2.99 , 167.44
4/1/18,East,Jones,Binder,60, 4.99 , 299.40
4/18/18,Central,Andrews,Pencil,75, 1.99 , 149.25
5/5/18,Central,Jardine,Pencil,90, 4.99 , 449.10
5/22/18,West,Thompson,Pencil,32, 1.99 , 63.68
6/8/18,East,Jones,Binder,60, 8.99 , 539.40
6/25/18,Central,Morgan,Pencil,90, 4.99 , 449.10
7/12/18,East,Howard,Binder,29, 1.99 , 57.71
7/29/18,East,Parent,Binder,81, 19.99 ," 1,619.19 "
8/15/18,East,Jones,Pencil,35, 4.99 , 174.65
9/1/18,Central,Smith,Desk,2, 125.00 , 250.00
9/18/18,East,Jones,Pen Set,16, 15.99 , 255.84
10/5/18,Central,Morgan,Binder,28, 8.99 , 251.72
10/22/18,East,Jones,Pen,64, 8.99 , 575.36
11/8/18,East,Parent,Pen,15, 19.99 , 299.85
11/25/18,Central,Kivell,Pen Set,96, 4.99 , 479.04
12/12/18,Central,Smith,Pencil,67, 1.29 , 86.43
12/29/18,East,Parent,Pen Set,74, 15.99 ," 1,183.26 "
1/15/19,Central,Gill,Binder,46, 8.99 , 413.54
2/1/19,Central,Smith,Binder,87, 15.00 ," 1,305.00 "
2/18/19,East,Jones,Binder,4, 4.99 , 19.96
3/7/19,West,Sorvino,Binder,7, 19.99 , 139.93
3/24/19,Central,Jardine,Pen Set,50, 4.99 , 249.50
4/10/19,Central,Andrews,Pencil,66, 1.99 , 131.34
```

```
4/27/19,East,Howard,Pen,96, 4.99 , 479.04
5/14/19,Central,Gill,Pencil,53, 1.29 , 68.37
5/31/19,Central,Gill,Binder,80, 8.99 , 719.20
6/17/19,Central,Kivell,Desk,5, 125.00 , 625.00
7/4/19,East,Jones,Pen Set,62, 4.99 , 309.38
7/21/19,Central,Morgan,Pen Set,55, 12.49 , 686.95
8/7/19,Central,Kivell,Pen Set,42, 23.95 ," 1,005.90 "
8/24/19,West,Sorvino,Desk,3, 275.00 , 825.00
9/10/19,Central,Gill,Pencil,7, 1.29 , 9.03
9/27/19,West,Sorvino,Pen,76, 1.99 , 151.24
10/14/19,West,Thompson,Binder,57, 19.99 ," 1,139.43 "
10/31/19,Central,Andrews,Pencil,14, 1.29 , 18.06
11/17/19,Central,Jardine,Binder,11, 4.99 , 54.89
12/4/19,Central,Jardine,Binder,94, 19.99 ," 1,879.06 "
12/21/19,Central,Andrews,Binder,28, 4.99 , 139.72
```

Objectives
In this project/lab the student will:
- Use a text editor and text filtering tool.

Equipment/Supplies Needed
- As specified in Lab 0.0.1.
- Provided data file above.

Procedure
Perform the steps in this lab in the order they are presented to you. Answer all questions and record the requested information. Use the Linux Virtual Machine to perform lab activities as directed. Unless otherwise stated, all tasks done as a non-root user. If root access is needed use the sudo command.

Assignment

Housekeeping is in order before we complete this exercise. In the sample screen section is a text file. We need to turn that into a text file for the lab.

Open Terminal, execute

```
nano Data.txt
```

Nano should open blank. With your mouse select the text in the Example Screens section, copy it to memory. Do a ctrl+shift+v into nano. The text should fall nicely into the screen in rows in nano. Save the file with a ctrl+x then y. Check that is formatted correctly with, cat Data.txt Information should show up in format of value,value,value.... If not try again. Record a screenshot. Place that image in a Word or Writer document.

Launch Debian.

Sort

With Terminal still open, execute:

1. `sort -t$',' -k3 Data.txt`

Your original file has been sorted by the third field, Rep. That happens to be the third field in the file. So lets parse:

2. `-t$',' – -t`

what follows is the delimiter for the fields. $',' says use comma as the delimiter. -k3 – -k says which field to sort with. Sort uses a starting index of 1 so the date field is k1. Data.txt – File to sort. What is on the screen is the result, your original file is untouched.

Try this:

3. `sort -t$',' -k5 Data.txt`

Hmmm. Not in the right order is it? OK, add a -n switch

4. `sort -t$',' -n -k5 Data.txt`

-n means to do a numeric sort on that field. Looks right now.

Record a screenshot. Place that image in a Word or Writer document.

5. `sort -t$',' -n -k7 Data.txt`

Why does that not work correctly? *Hint: strings.*

Record your answer in a Word or Writer document.

Unique

Unique is designed to eliminate duplicates. It is designed to scan whole lines and provide only the unique ones in order. Problem is for it to work properly the duplicates must be packed together. But unique has its uses.

6. `uniq -c Data.txt`

Will provide a listing with a count. In this case every line only has one entry due to the date field being unique.

Cut

cut extracts a particular field or group of fields as you need them. Try,

7. `cut -d ',' -f 3 Data.txt`

-d - defines the delimiter -f - the field

You can do multiple fields as well. Try,

8. `cut -d ',' -f 2-7 Data.txt`

Record a screenshot. Place that image in a Word or Writer document.

Say I wanted to find how many orders a salesman made out of this Data.txt

file. Try,

9. `cut -d ',' -f 3 Data.txt | sort -t$',' -k1 | uniq -c`

Not clear? Lets parse it,

cut -d ',' -f 3 Data.txt pulls down the field we need, the salesman's last name. It's piped to, sort -t$',' -k1 that arranges them in order by salesman. It’s piped to, uniq -c which provides only 1 unique name and the number of times it appears.

Linux provides tools that do one thing well. If each tool is doing its part in a string of tools one can develop unique answers to problems without doing a lick of programming.

Here is your challenge. How would I determine how many orders were made by region?

Record those command(s) in a Word or Writer document.

Paste

paste is a command used to combine two files together. It's best if both files are in row order and have the same number of rows Or maybe you need a number sequence to keep a file in order should it get jumbled.

Create these two files using cut from our Data.txt file,

10.
```
$ cut -d , -f 1,2,3 data.txt >file1.txt
$  cut -d , -f 4- data.txt >file2.txt
```

You should now have 2 files which contains 3 columns in file1.txt and the balance of the fields in file2.txt. Now try,

11. `paste file1.txt file2.txt`

Compare the resulting output to the format of your original Data.txt file. How is it different? Try this minor change --

12. `paste -d , file1.txt file2.txt`

Compare the resulting output to the format of your original `Data.txt` file.

Tr

tr does single character replacement of items in a file.

Try it:

12. `tr ',' ':' <Data.txt`

tr replaced all , with : . Not useful for replacing words but we will get to that in a moment.

Sed

sed is what is called a stream editor. It can take a file or even a live stream of data coming across the network and do whole word replacements. Say that Gill married and assumed the name of Smythe. I could make those changes rather quickly. Try

13. `sed 's/Gill/Smythe/g' <Data.txt`

Record a screenshot. Place that image in a Word or Writer document.

Parsing this sed command:

- the 's' indicates you wish to do a substitution with what follows. There is also 'd' for delete as well.
- The first word is the target item to be replaced.
- The second word is the replacement for the target.
- 'g' means to do this globally throughout the file. Without it only the first instance is acted upon in the file.

You are not limited to just words, you could use whole passages to be substituted. sed has buffer, redirects, file calls, etc. A whole academic course could be written on just the use of this command.

Awk

Have columnar data? awk is the tool to use. The linux version is called gawk. It is used quite a lot in parsing system logs, morphing data sets, acting much like a spreadsheet in function.

Let's find all the named accounts in the Linux box

14. `gawk -F: '{print $1}' /etc/passwd`

parsed out -

- gawk - awk command.
- -F : - the field separator, in this case ‘:’.

- '{print $1}' - action(s) to be taken.
- /etc/passwd - the file to be read.

Another example.

Try,

15. `gawk -F , '{print $3,$4,$5*$6 }' Data.txt`

Modify the example above and include the date and region in the output. Record a screenshot. Place that image in the Word or Writer document.

awk can get a lot more sophisticated. awk has BEGIN and END functions that permit the program to do pre and post processing of the data file. The code language is a subset of C. One can take actions based on a stack of filters as to which action(s) are to be executed. The data can then be summarized with the END block.

Lab Submissions Proof: Provide screenshots as indicated in the lab; upload your proof to Canvas for grading.

Rubric

Checklist/Single Point Mastery

Concerns Working Towards Proficiency	Criteria Standards for This Competency	Accomplished Evidence of Mastering Competency
	Criteria #1: Record screenshot of cat Data.txt (10 points)	
	Criteria #2: Record screenshot of sort -t$',' -n -k5 Data.txt (15 points)	
	Criteria #3: Why does sort -t$',' -n -k7 Data.txt not work correctly? (15 points)	
	Criteria #4: Record screenshot of cut -d ',' -f 2-7 Data.txt (15 points)	
	Criteria #5: How would I determine how many orders were made by region? Specify command (15 points)	
	Criteria #6: Record a screenshot of sed 's/Gill/Smythe/g' <Data.txt (15 points)	
	Criteria #7:Record a screenshot modifying gawk -F , '{print $3,$4,$5*$6 }' Data.txt to include the date and region in the output. (15 points)	