Master Data Science 22/11/19 Igor

Si se bloquea el terminal:Control Z o control C

# Tree

rse@LAPTOP-RISUJV9C:~$ tree DS\_CL\_files (para ver todos los directorios y subdirectorios desde donde me encuentre

DS\_CL\_files

├── README.md

├── opentraveldata

│   ├── 2columns.txt

│   ├── 5lines.csv.bz2

│   ├── numbers

│   ├── optd\_aircraft.csv.bz2

│   ├── optd\_airlines.csv.bz2

│   ├── optd\_por\_public.csv.bz2

│   └── uniq\_example.txt

└── shell

├── Finn.txt

├── **README.md.gz**

├── Text\_example.txt

└── Text\_example.txt.copy

Tree -L 2 Me devolveria los archivos hasta el subdirectorio 2.

# Sorting and Counting Utilities

## Secuencia

rscheze@LAPTOP-RISUJV9C:~$ seq 10

Tree es un comando genial para ver de forma clara los archivos debajo de los directorios.

**tree Documents/**

**du -h Documents/** me dice el tamaño debajo

1

2

3

4

5

6

7

8

9

10

rscheze@LAPTOP-RISUJV9C:~$ **seq 10 20**

10

11

12

13

14

15

16

17

18

19

20

**Empieza en 1 , a cada paso añade 2 hasta 10 (me devuelve impares)**

rscheze@LAPTOP-RISUJV9C:~$ seq 1 2 10

1

3

5

7

9

rscheze@LAPTOP-RISUJV9C:~$ seq 1 10 >> uniq\_example.txt (así guardo la serie de los nñumeros para duplicarlos después y ver un ejemplo)

rscheze@LAPTOP-RISUJV9C:~$ cat uniq\_example.txt

1

2

3

4

5

6

7

8

9

10

**Uniq compara líneas para que no sean iguales. Compara un numero con el siguiente por lo que es imprescindible ordenarlos antes**

### **Lo ordeno (numero -n, letras -d)**

rscheze@LAPTOP-RISUJV9C:~$ **sort -n uniq\_example.txt**

1

2

3

4

5

6

7

8

9

10

### **Ordeno y deduplico**

rscheze@LAPTOP-RISUJV9C:~$ sort -n uniq\_example.txt | uniq

1

2

3

4

5

6

7

8

9

10

#### **Esto sería lo mismo que el comando antes**

rscheze@LAPTOP-RISUJV9C:~$ sort -n -u uniq\_example.txt

### **Así me devolvería los duplicados**

rscheze@LAPTOP-RISUJV9C:~$ **sort -n uniq\_example.txt | uniq -d**

### **Así me devolvería las veces que está duplicado además del número**

rscheze@LAPTOP-RISUJV9C:~$ **sort -n uniq\_example.txt | uniq -c**

1 1

1 2

1 3

1 4

1 5

1 6

1 7

1 8

1 9

1 10

**Ordeno de forma numerica(n), inversa® y los valores únicos(u)**

rscheze@LAPTOP-RISUJV9C:~$ **sort -nru uniq\_example.txt**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **sort optd\_aircraft.csv | head**

**ordeno sobre delimitador “^” por columnas(k). desde la 6 a la última y cada columna me la ordena de forma inversa.el contenido de cada columna (reverse)**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **sort -t “^” -k 6r optd\_aircraft.csv | head**

**ordenar desde la primera hasta el final**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **sort -t “^” -k 1 optd\_aircraft.csv | head**

#### **Si quiero que empiece por la 6 columna y acabar en la 6 columna**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **sort -t “^” -k 6,6 optd\_aircraft.csv | head**

**puedo decirle que ordene la columna 6 y esa columna me la ordene de forma inversa y después lo mismo con la octava.**

### **Por la pantalla siempre me sale todo pero ordena lo que le pido que ordene**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **sort -t “^” -k 6,6 -k8,8r optd\_aircraft.csv | head**

### **Si quiero ordenar el avión con mas motores. (la columna donde esta esa info es la 7)**

Columnas: iata\_code^manufacturer^model^iata\_group^iata\_category^icao\_code^nb\_engines^aircraft\_type

### **El An-225 es el que mas motores tiene**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **sort -t “^” -k 7,7nr ./optd\_aircraft.csv | head**

A5F^Antonov^An-225^A5F^6J^A225^6^J

141^BAe^146-100^146^4J^B461^4^J

142^BAe^BAE Systems 146-200 Passenger^146^4J^B462^4^J

143^BAe^146-300^146^4J^B463^4^J

### **Cuántos fabricantes temenos? Necesito usar uniq para que se dedupliquen**

**La respuesta es 76 Ojo porque salen 77 pero el conteo incluye el cabecero del fichero por eso es una menos.**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **sort -t “^” -k2,2 -u optd\_aircraft.csv | wc**

**77 184 3185**

### **Así sería lo mismo**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **sort -ut “^” -k2,2 optd\_aircraft.csv | wc**

## **Como sacar una o varias columnas de un fichero**

f- field

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata**$ cut - d “^” -f 1,3-5 optd\_aircraft-csv | head**

**para guardarlo**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **cut - d “^” -f 1,3-5 optd\_aircraft-csv > optd\_aircraft-csv| head**

### **Paste pega los números (primero he creado seq 1 10 y lo he guardado como números)**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **paste -d “^” numbers numbers**

1^1

2^2

3^3

4^4

5^5

6^6

7^7

8^8

9^9

10^10

### En horizontal

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **paste -s numbers**

1 2 3 4 5 6 7 8 9 10

### **Paste donde le digo que quiero tres columnas (- - -). Separado cada raya por un espacio**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata**$ cat numbers | paste - - -**

1 2 3

4 5 6

7 8 9

10

### **En dos**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **cat numbers | paste - -**

1 2

3 4

5 6

7 8

9 10

**Paste concatena (en este caso las filas del fichero optd\_aircraft con numbers(creado antes que va del 1 al 10)**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **paste optd\_aircraft.csv numbers numbers | head**

iata\_code^manufacturer^model^iata\_group^iata\_category^icao\_code^nb\_engines^aircraft\_type 1 1

100^Fokker^100^100^2J^F100^2^J 2 2

141^BAe^146-100^146^4J^B461^4^J 3 3

142^BAe^BAE Systems 146-200 Passenger^146^4J^B462^4^J 4 4

143^BAe^146-300^146^4J^B463^4^J 5 5

146^BAe^146^^^^^ 6 6

14F^BAe^146 Freighter (-100/200/300QT & QC)^^^^^ 7 7

14X^BAe^146 Freighter (-100QT & QC)^14F^4J^B461^4^J 8 8

14Y^BAe^146 Freighter (-200QT & QC)^14F^4J^B462^4^J 9 9

14Z^BAe^146 Freighter (-300QT & QC)^14F^4J^B463^4^J 10 10

### **Si quiero me ejecute un comando( en este caso era una lista de números)**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **paste <(cat numbers) <(cat numbers)**

1 1

2 2

3 3

4 4

5 5

6 6

7 7

8 8

9 9

10 10

#### **Otro ejemplo**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **paste <(seq 1 10) <(head optd\_aircraft.csv)**

1 iata\_code^manufacturer^model^iata\_group^iata\_category^icao\_code^nb\_engines^aircraft\_type

2 100^Fokker^100^100^2J^F100^2^J

3 141^BAe^146-100^146^4J^B461^4^J

4 142^BAe^BAE Systems 146-200 Passenger^146^4J^B462^4^J

5 143^BAe^146-300^146^4J^B463^4^J

6 146^BAe^146^^^^^

7 14F^BAe^146 Freighter (-100/200/300QT & QC)^^^^^

8 14X^BAe^146 Freighter (-100QT & QC)^14F^4J^B461^4^J

9 14Y^BAe^146 Freighter (-200QT & QC)^14F^4J^B462^4^J

10 14Z^BAe^146 Freighter (-300QT & QC)^14F^4J^B463^4^J

**Si el header quiero ponerlo en horizontal y numerarlo**.

#### Tr= translate (me cambia el delimitador por el que yo quiera. En este caso ^por /

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **head -1 optd\_aircraft.csv | tr “^” “/n”**

iata\_code/manufacturer/model/iata\_group/iata\_category/icao\_code/nb\_engines/aircraft\_type

**Si cambio el delimitador de ^a \n entonces cambia de linea**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **head -1 optd\_aircraft.csv | tr “^” “\n”**

iata\_code

manufacturer

model

iata\_group

iata\_category

icao\_code

nb\_engines

aircraft\_type

**rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ paste <(seq 1 10) <(head -1 optd\_aircraft.csv | tr “^” “\n”)**

1 iata\_code

2 manufacturer

3 model

4 iata\_group

5 iata\_category

6 icao\_code

7 nb\_engines

8 aircraft\_type 9 10

**Ejemplo con tr donde me cambia minúsculas por mayúsculas(en este caso la a)**

**Translate buscar carácter por carácter. No busca la palabra entera.**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **echo “master data science” | tr a A**

mAster dAtA science

**o cambios de letras**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **echo “master data science” | tr ma AM**

##### AMster dMtM science

**Comandos de squeeze. Se queda con el primer espacio y borra el resto**.

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **echo “master data science” | tr -s “ “**

master data science

**otros ejemplos con otros comandos de tr**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **echo “master data science” | tr -d “ a”**

msterdtscience

### **Para cambiar el delimitador siempre lo haría** así

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **cat optd\_aircraft.csv | tr “^” “\t” | head**

iata\_code manufacturer model iata\_group iata\_category icao\_code nb\_engines aircraft\_type

100 Fokker 100 100 2J F100 2 J

141 BAe 146-100 146 4J B461 4 J

142 BAe BAE Systems 146-200 Passenger 146 4J B462 4 J

143 BAe 146-300 146 4J B463 4 J

146 BAe 146

14F BAe 146 Freighter (-100/200/300QT & QC)

14X BAe 146 Freighter (-100QT & QC) 14F 4J B461 4 J

14Y BAe 146 Freighter (-200QT & QC) 14F 4J B462 4 J

14Z BAe 146 Freighter (-300QT & QC) 14F 4J B463 4 J

#### **Cambiar todas a mayusculas**

**rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ echo “master data science” | tr “[:lower:]” “[:upper:]”**

##### MASTER DATA SCIENCE

### **Comando grep**

### **Busca un texto dentro de un archivo. Devuelve filas**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell**$ grep this Text\_example.txt**

**con la n me añade el numero de línea**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **grep -n this Text\_example.txt**

2:this line is the 1st lower case line in this file.

6:Two lines above this are empty.

7:And this is the last line.

#### **Puedo buscar en dos ficheros a la vez y me dirá donde lo ha encontrado**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ grep -n this Text\_example.txt Text\_example.txt

Text\_example.txt:2:this line is the 1st lower case line in this file.

Text\_example.txt:6:Two lines above this are empty.

Text\_example.txt:7:And this is the last line.

Text\_example.txt:2:this line is the 1st lower case line in this file.

Text\_example.txt:6:Two lines above this are empty.

Text\_example.txt:7:And this is the last line.

### **Para que me diga donde no estaba**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **grep -nv this Text\_example.txt** Text\_example.txt

Text\_example.txt:1:THIS LINE IS THE 1ST UPPER CASE LINE IN THIS FILE.

Text\_example.txt:3:This Line Has All Its First Character Of The Word With Upper Case.

Text\_example.txt:4:

Text\_example.txt:5:

Text\_example.txt:1:THIS LINE IS THE 1ST UPPER CASE LINE IN THIS FILE.

Text\_example.txt:3:This Line Has All Its First Character Of The Word With Upper Case.

Text\_example.txt:4:

Text\_example.txt:5:

### **Si quiero que me busque This con o sin mayúsculas**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **grep -ni This Text\_example.txt**

1:THIS LINE IS THE 1ST UPPER CASE LINE IN THIS FILE.

2:this line is the 1st lower case line in this file.

3:This Line Has All Its First Character Of The Word With Upper Case.

6:Two lines above this are empty.

7:And this is the last line.

### **Para que me cuenta las veces que aparece la palabra que busco**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **grep -n -o this Text\_example.txt**

2:this

2:this

6:this

7:this

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **grep -n -o this Text\_example.txt | wc**

4 4 28

### **Si busco algo en el primer símbolo de la línea.**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **grep -n “^T” Finn.txt | less**

**El capuchón antes de la T es fundamental para que me busque solo las líneas que empiezan por T**

### **Comando sed (String editor)**

Se usa para cambiar de un string a otra cosa.

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **echo Sunday | sed ssdaysnights**

Sunnight

**Ssdaysnights- la primera s es substitute y la segunda el delimitador. También se podría poner así**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ echo Sunday | sed s/day/night/

Sunnight

**Antes pusimos una s como delimitador y ahora /. Se puede poner lo que queramos como delimitador.**

**En los anteriores casos se paraba cuando lo encontraba. Si pongo g al final seguirá para encontrar más casos**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **echo Sunday | sed s/day/night/g**

**otro ejemplo**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **sed s/this/WHAAT/g Text\_example.txt**

THIS LINE IS THE 1ST UPPER CASE LINE IN THIS FILE.

WHAAT line is the 1st lower case line in WHAAT file.

This Line Has All Its First Character Of The Word With Upper Case.

Two lines above WHAAT are empty.

And WHAAT is the last line.

**Si ponemos -I significa inplace y lo que hace es cambiar el fichero. Mejor hacer una copia del fichero antes. (La I después de g significa case insensitive)**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **sed -i s/this/WHAAT/gI Text\_example.txt.copy**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **cat Text\_example.txt.copy** WHAAT LINE IS THE 1ST UPPER CASE LINE IN WHAAT FILE. WHAAT line is the 1st lower case line in WHAAT file.

WHAAT Line Has All Its First Character Of The Word With Upper Case.

Two lines above WHAAT are empty.

And WHAAT is the last line.

Si me he equivocado, para volver al a versión anterior en nuestro ordenador local. Justo el último commit.

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **git reset—hard HEAD** Unstaged changes after reset:

M shell/Text\_example.txt.copy

**Como puedo seleccionar una línea de un fichero y eliminarla o sacar sin tener que partir el fichero en dos.**

### **De las 2 a la 4**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **seq 5 | sed ‘2,4d’**

1

5

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **seq 5 | sed ‘2d,4d’**

sed: -e expression #1, char 3: extra characters after command

### **Borrar la 2 y la 4**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **seq 5 | sed ‘2d;4d’**

1

3

5

### **Borrar la línea 3 que es el numero 12**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **seq 10 15 | sed ‘3d’**

10

11

13

14

15

### **Si lo que quiero es que me borre el numero 14**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell**$ seq 10 15 | sed ‘/14/d’**

10

11

12

13

15

### **Si quiero señalar solo el numero 14**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **seq 10 15 | sed ‘/14/!d’**

### **O la linea 3 que es el numero 12 en este caso**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **seq 10 15 | sed ‘3!d’**

12

### **Si quiero quitar parte de una variable en una columna ( ej 150lbs para que me devuelve solo 150)**

Fifa.weight = [int(x.strip(‘lbs’)) if type(x) == str else x for x in fifa.weight]

He hecho un bucle y de paso lo he transformado en entero.

Para quitar esa parte he utilizado el comando strip()

He tenido que meter if type(x)== str porque daba error y seguramente era porque habrá algún valor nulo en la columna

### **Comandos para trabajar con ficheros comprimidos, como descomprimirlos etc**

#### ZIP

## Buscar como se hace

#### Zip, zipinfo,zless,zgrep,unzip,zcat

**Gzip-** lo Bueno de este comando es que siempre tiene solo un fichero dentro. Sirve para hacer un zip del fichero

Gzip y el nombre del fichero

Gzip -d descomprimir archivo

Gzip -f forzar compresión

Gzip -k comprime y manteniene original sin comprimir

Gzip -r comprime todos los archivos del directorio y subdirectorios

Gzip -9 máximo nivel de compresión

**Gunzip** -para descomprimir.

2 directories, 12 files

rse@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ **gunzip README.md.gz**

rse@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$ ll

total 616

drwxrwxrwx 1 rse rse 4096 Nov 24 13:21 ./

drwxrwxrwx 1 rse rse 4096 Nov 23 17:56 ../

-rw-rw-rw- 1 rse rse 627563 Nov 23 17:56 Finn.txt

-rwxrwxrwx 1 rse rse 0 Nov 23 18:10 README.md\*

-rw-rw-rw- 1 rse rse 237 Nov 23 17:56 Text\_example.txt

-rw-rw-rw- 1 rse rse 272 Nov 24 12:29 Text\_example.txt.copy

rse@LAPTOP-RISUJV9C:~/DS\_CL\_files/shell$

**Gzip -k** para mantener el original.

#### Para descomprimir

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **bzip2 \*.csv**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ ll

total 1884

drwxrwxr-x 1 rscheze rscheze 4096 Nov 22 20:25 ./

drwxrwxr-x 1 rscheze rscheze 4096 Nov 22 17:38 ../

* rw-rw-r-- 1 rscheze rscheze 21 Nov 22 18:23 numbers
* rw-rw-r-- 1 rscheze rscheze 5906 Nov 22 17:38 optd\_aircraft.csv.bz2
* rw-rw-r-- 1 rscheze rscheze 29968 Nov 22 17:38 optd\_airlines.csv.bz2
* rw-rw-r-- 1 rscheze rscheze 1871775 Nov 22 17:38 optd\_por\_public.csv.bz2
* rw-rw-r-- 1 rscheze rscheze 6069 Nov 22 20:14 output.zip

### Para descomprimir ficheros grandes

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **bzcat optd\_aircraft.csv.bz2 | head**

### Los commandos de BZ son (bz y le doy dos veces a tab)

**rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ bz**

bzcat bzdiff bzexe bzgrep bzip2recover bzmore

bzcmp bzegrep bzfgrep bzip2 bzless

### Tar Podemos usar para comprimir o descomprimir varios ficheros a la vez.

comprimir

tar -cvzf

tar -czf

c-create

z-zip

f- file

r- para añadir más files a un archivo zip ya existente.

u- update existing files

v-show you the output (verbose)

ejemplo: **tar -czvf (aquí iría el nombre del archivo con .tar\* al final)**

Las extensión de los ficheros puede ser tar.gz o directamente tgz

descomprimir

tar -xf

x- extract

f-file

**tar -cjf para comprimir en bz2 . Con bz2 no puedo hacer agrupación de ficheros. En este caso uso tar para agrupar zip bajo un mismo fichero.**

**Script (<buscarlo entero)**

Con control r veo el comando que haya usado antes.Tengo que escribirlo.

Los archivos que se crean aquí suelen terminarse con sh por ser de Shell

**Así abro un editor de texto. Debe acabar con sh.**

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata**$ nano csv\_number\_column.sh**

**Esto es mejor hacerlo en otro terminal de Ubuntu**

* Copio los comando que quiero ejecutar (poniendo primero which bash)

**#!/bin/bash**

paste <(seq 1 10) <(head -1 optd\_aircraft.csv | tr "^" "\n") head -1 optd\_aircraft.csv | tr "^" "\n" | wc -l

* Así ejecuto lo que he creado en el script

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ ./csv\_number\_column.sh

Esto sirve para crear una variable y poder utilizarla cuando la necesite.

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ VARIABLE=algo

rscheze@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ echo $VARIABLE

algo

Volviendo al ejemplo anterior. Puedo poner una variable en el terminal que estoy usando como repositorio

**Resumen script**

Muy útil para crear códigos que podré utilizar en varios de mis programas

Necesitaré crear tantas variables como campos de entrada vaya a tener. Si mañana cambian simplemente los cambiaré en un sitio.

En mi ejemplo :

rse@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **export FILE=optd\_aircraft.csv**

rse@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ echo $FILE

Después me creo el script. Para ello primero abro nano y le pongo un nombre *(csv\_number\_column.sh)*

rse@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **nano csv\_number\_column.sh**

**Una vez abierto incluyo el código. Siempre se empieza por la localización (#!/bin/bash)**

**Sustituyo el nombre del fichero donde tengo los datos a tratar por la variable. En este caso $FILE**

#!/bin/bash

paste <(seq 1 8) <(head -1 $FILE | tr "^" "\n")

Es conveniente poner la variable y su correspondiente dato dentro del script para acordarnos. Esto debe estar precedido de # almohadilla para que no nos de error.

**Para guardar control O después enter y después control X para salir**

**Después lo ejecuto poniendo el nombre desde donde lo quiero ejecutar.**

rse@LAPTOP-RISUJV9C:~/DS\_CL\_files/opentraveldata$ **./csv\_number\_column.sh**

**EJERCICIOS**

Find top 10 files by size in your home directory including the subdirectories. Sort them by size and print the result including the size and the name of the file (hint: use find with -size and -exec ls -s parameters)

Incluyo dos opciones para hacerlo:

**rse@LAPTOP-RISUJV9C:~$ du -a | sort -h -r | head -n 10**

**find /home/rse -exec du -Sh {} + | sort -nr | head -10**

2. Create a dummy file with this command : seq15> 20lines.txt; seq9 1 20 >> 20lines.txt; echo"20\n20" >> 20lines.txt; (check the content of file first)

a) Sort the lines of file based on alphanumeric characters

**sort -n 20lines.txt**

b) Sort the lines of file based on numeric values and eliminate the duplicates

**sort -n 20lines.txt | uniq**

c) Print all duplicated lines of the file

**sort -n 20lines.txt | uniq -d**

d) Print the line which has most repetitions

**sort -n 20lines.txt | uniq -cd**

3. Create another file with this command : seq 0 2 40 > 20lines2.txt

a) Create 3rd file from the first two but without duplicates

**rse@LAPTOP-RISUJV9C:~$ cat 20lines.txt 20lines2.txt | sort -u > 20lines3.txt**

b) Merge the first two files. Print unique lines together with the number of occurrences inside the merged file and sorted based on line content.

4. Go to ~/Data/opentraveldata. Get the line with the highest number of engines using sort.

Primero veo la primera fila para saber el numero de columna

**cat optd\_aircraft.csv | head -1**

**después selecciono la que quiero ver y la ordeno de forma inversa para que salgan las mayores primero. Importante el delimitador**

**sort -t '^' -k 7nr optd\_aircraft.csv | head**

**Go to ~/Data/opentraveldata**

1. Change the delimiter of optd\_aircraft.csv to “,”

**cat optd\_aircraft.csv | tr '^' ',' > optd\_aircraft\_commas.csv**

2. Check if optd\_por\_public.csv has repeated white spaces

Primero he tenido que descomprimirlo (-k para mantener original): **gzip -k optd\_por\_public.csv.bz2**

**Para quitar los espacios y me cuente palabras etc: cat optd\_por\_public.csv.bz2.gz | tr -s '[:blank]'| wc**

**7556 42483 1869054**

**Para ver como compara con el original: wc optd\_por\_public.csv.bz2.gz**

**7556 42483 1869344**

3. How many columns has optd\_por\_public.csv? (hint: use head and tr)

**head -1 optd\_por\_public.csv.bz2.gz | tr '^' '\n' | wc -l**

4. Print column names of optd\_por\_public.csv together with their column number. (hint: use paste)

5. Use optd\_airlines.csv to obtain the airline with the most flights?

6. Use optd\_airlines.csv to obtain number of airlines in each alliance?