1º DAM/DAW Sistemas Informáticos

U2. Comandos en Linux.

Comandos. Contenido de ficheros II

Pascual Martínez

Comando AWK

- **AWK** es un comando muy versátil que sirve para **procesar patrones en líneas de texto**. Es decir, podemos usarlo, por ejemplo:
  - **Para acceder a datos con formato de tabla**, que estén en un fichero, o lleguen a través de una tubería.


```
root    0       0
daemon  1       1
bin             2       2
sys             3       3
sync    4       65534
games   5       60
```

  - **Para acceder a datos separados por un delimitador** que estén en un fichero, o lleguen a través de una tubería.

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
```

Comando AWK

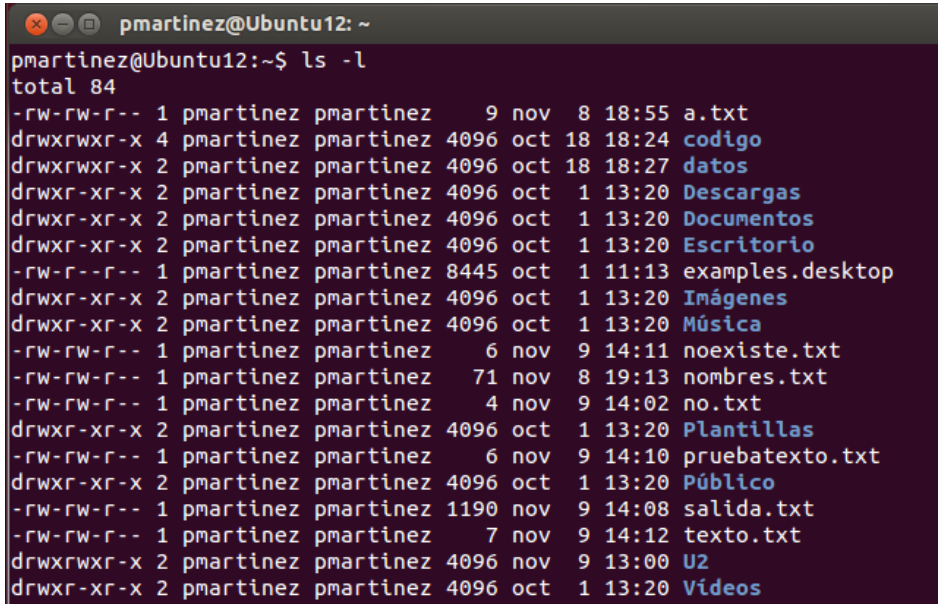
- **AWK sintaxis**: es compleja y amplia, de hecho, AWK se considera un lenguaje por su potencialidad y versatilidad.
- En nuestro caso, lo usaremos principalmente para: **dado un flujo o conjunto de información de entrada**, una tabla y/o fichero, poder **seleccionar o flitrar un subconjunto** en base a algún criterio.
- Por ejemplo, podemos leer ciertas columnas de una tabla, del siguiente modo:

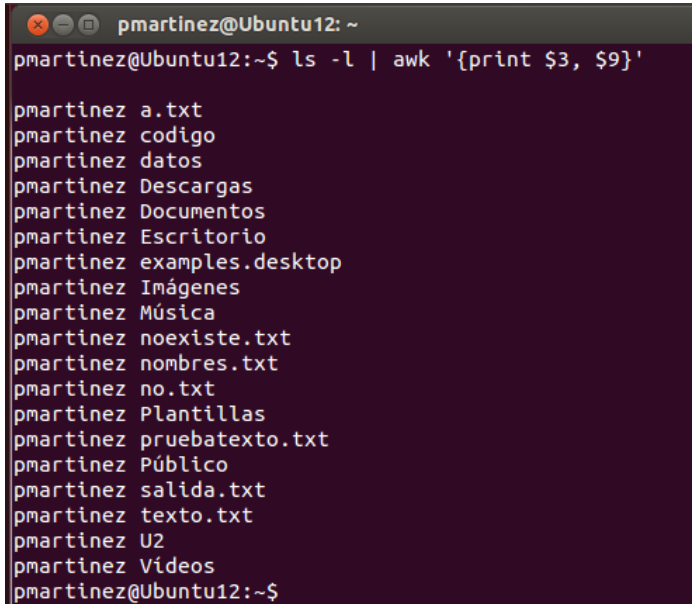
$ awk ‘{print $número_columna_1, …, $número_columna_n}’

Comando AWK

- **AWK Ejemplo**: ejecutamos ls -l en nuestra carpeta de usuario. El resultado lo redirigimos, mediante una tubería, para que el comando **awk** muestre el valor de las columnas **3** y **9** por pantalla:

$ ls -l | awk ‘{print $3, $9}’





Ampliación comando GREP

- Ampliando un poco la utilidad del **comando grep** (introducido en el anterior documento), podríamos decir que nos **permite localizar coincidencias en el contenido de ficheros, dado un patrón de búsqueda**.
- El patrón de búsqueda se denomina también **expresión regular**. Una expresión regular es una denominación más genérica utilizada también en otros ámbitos, y en detalle, es una **secuencia de caracteres que conforma un patrón de búsqueda**.

GREP. **Sintaxis**

- La sintaxis del comando grep es la siguiente:

  $ grep [opciones] [expresión regular] [archivos]

- Donde:
  - **[opciones]:** nos permite indicar parámetros para personalizar el uso del comando.
  - **[expresión regular]:** nos permite definir el patrón de búsqueda.
  - **[archivos]:** nos permite expresar los archivos (ruta) donde se realizará la búsqueda.

GREP. **Buscar en el contenido de un fichero**

- Vamos a comenzar con el caso más básico del comando grep, **sin usar opciones**, en el que queremos buscar una cadena de texto en un fichero conocido:

  $ grep “texto” fichero.txt

  ***(grep [expresión regular] [archivos])**

- Otra opción muy habitual para realizar lo mismo sería:

  $ cat fichero.txt | grep “texto”

  ***(grep [expresión regular])**

- El comando **devolverá las líneas del fichero que contengan alguna coincidencia** con el patrón de búsqueda, es decir, la cadena “**texto**”.

GREP. Buscar en el contenido de un fichero, **opciones**

- Respecto a las **opciones** (***(**$ grep [opciones] [expresión regular])***) que podemos indicar como parámetros, unas de las más relevantes son las siguientes:

-i : no tendrá en cuenta mayúsculas y minúsculas a la hora de buscar.

$ cat fichero.txt | grep -i “texto”

-v : nos devuelve el resultado inverso, es decir, las líneas que no cumplan el patrón.

$ cat fichero.txt | grep -v “texto”

GREP. Buscar en el contenido de un fichero, **opciones**

-n : muestra el número de línea donde sucede la coincidencia.

$ cat fichero.txt | grep -n “texto”

-w : encuentra coincidencia sólo si el patrón coincide con una palabra completa.

$ cat fichero.txt | grep -w “texto”

- **etc**.

- Las opciones **se pueden combinar**, por ejemplo:

$ cat fichero.txt | grep -iw “texto”

GREP. Buscar en el contenido de un fichero, **expresión regular**

- Hasta este momento, hemos visto cómo localizar cadenas de texto.
- Vamos a centrarnos ahora en cómo generar una **expresión regular** que nos sirva como **patrón** para realizar la búsqueda.

GREP. Buscar en el contenido de un fichero, **expresión regular**

- Las expresiones regulares son una **secuencia de caracteres que conforma un patrón de búsqueda.**
- Nos **permiten definir una serie de aspectos con mayor nivel de detalle**, para concretar mejor la búsqueda a realizar.
- Por ejemplo, si queremos buscar una cadena de texto, pero queremos que sea la primera palabra de una línea, o la última. O bien sabemos que la cadena está formada por 3 letras y 4 números, pero no sabemos qué cadena es…..

GREP. Buscar en el contenido de un fichero, **expresión regular**

- Las expresiones regulares utilizan una serie de caracteres específicos que permiten indicar este tipo de cuestiones:
  - **^** *(acento circunflejo)*: este carácter utilizado junto a una cadena de texto, indica que dicha cadena debe aparecer, al comienzo de una línea del fichero.

    $ cat fichero.txt | grep “^Texto”
  - **$** : lo mismo que el anterior, pero al final de una línea del fichero.

    $ cat fichero.txt | grep “texto$”
  - **.** *(punto)*: sustituye a cualquier carácter (similar al carácter comodín: ?).

    $ cat fichero.txt | grep “t...o”

GREP. Buscar en el contenido de un fichero, **expresión regular**

- **[]** *(corchetes)*: lo que va entre corchetes se entiende como un **conjunto de posibilidades que debe cumplir el resultado:**

  - **$ cat fichero.txt | grep “[a-z]texto”**

    *En este ejemplo, buscaría una cadena formada por una letra desde la **a** hasta la **z,** seguida de la palabra “texto”. Por ejemplo, encontraría una línea del fichero, si contiene la palabra “co**ntexto**” o “pr**etexto**”.*

  - **$ cat fichero.txt | grep “U[0-9]”**

    *En este ejemplo, buscaría una cadena formada por la letra “U” seguida de un número de **0** a **9**. Por ejemplo, encontraría la línea del fichero si contiene la palabra “**U1**” o “**U2**”.*

GREP. Buscar en el contenido de un fichero, **expresión regular**

- **[]** *(corchetes)*: podemos indicar el número de **repeticiones** de una letra o un número. Y también podemos **acotar el conjunto** de caracteres posible para una letra o un número:
  - **$ cat fichero.txt | grep "[a-t]*texto"**

    *En este ejemplo, el "*****" indica que la posible letra se repite de **0** a **n veces,** en este caso acotando el conjunto desde la **a** hasta la **t** (en orden alfabético)**,** y después va seguida la palabra "texto". Por ejemplo, encontraría una línea del fichero, si contiene la palabra "**contexto**" o "**pretexto**" o "**texto**" o "**htmltexto**"…*

  - **$ cat fichero.txt | grep "U[0-1]\{1,2\}"**

    *En este ejemplo, buscaría una cadena formada por la letra "U", en este caso seguida de **1** o **2** números acotados entre **0** y **1**. Por ejemplo, encontraría la línea del fichero si contiene la palabra "**U0**" o "**U1**" o "**U10**" o "**U01**".*

GREP. **Buscar en múltiples ficheros**

- Cuando la necesidad se amplía y en lugar de buscar coincidencias en 1 único fichero, lo que necesito es **localizar los ficheros de una determinada ruta** cuyo contenido contiene un patrón de búsqueda, lo podemos hacer del siguiente modo:

$ grep [opciones] [expresión regular] [ruta_archivos]

Donde **ruta_archivos** puede ser, por ejemplo:

- Todos los archivos de la carpeta actual (ruta relativa): **./***
- Los archivos que empiezan por lib, de la carpeta padre (ruta relativa): **../lib***
- Los archivos de texto de la carpeta de usuario (ruta absoluta): **/home/usuario/*.txt**
- Cualquier otra ruta absoluta o relativa…

GREP. Buscar en múltiples ficheros, **opciones**

Donde las **[opciones]** podrían ser:

-r : búsqueda **recursiva**, para que busque también en los subdirectorios.

$ grep -r “texto” ./*

Buscará la cadena “texto” en todos los fichero de la carpeta actual y en todos los subdirectorios de la carpeta actual también.

Nos dará como resultado la ruta de los ficheros en los que encuentre la cadena “texto” y, dentro de cada fichero, las líneas en las que ha localizado la coincidencia.

GREP. **Ejemplo combinado**

- En base al fichero “fichero.txt” que nos proporcionan y que guarda datos personales de soci@s de un club deportivo.
- Muestra el nombre y apellido (sabiendo que están en la columnas séptima y octava, respectivamente).
- De aquell@s soci@s, que contengan en sus datos el patrón siguiente: ***5 dígitos numéricos entre 0 y 9, seguidos de una de las letras “L” o “H”*** (esto podría ser el patrón de aquellos códigos de socio, que son socios de “Honor” o socios “Locales”):

$ cat fichero.txt | grep “[0-9]\{5\}[LH]” | awk ‘{print $7, $8}’