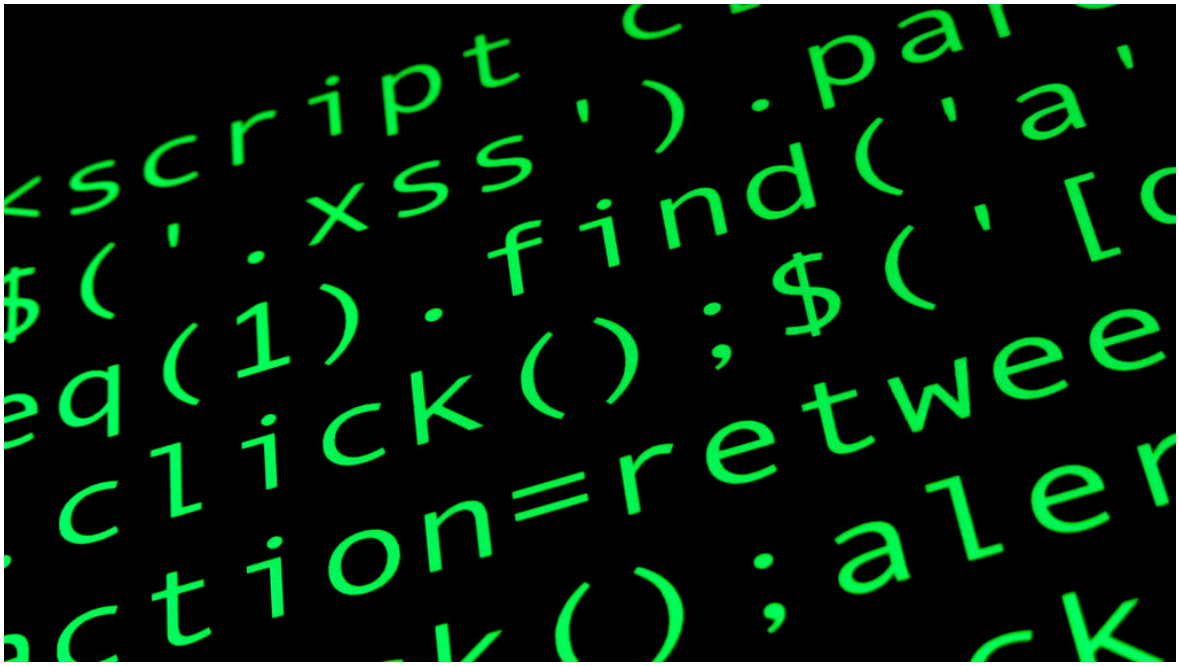
PRÁCTICA DE SCRIPTING



Enunciado:

Como administradores de sistemas gestores de bases de datos necesitaremos, en múltiples ocasiones, realizar tareas que deban ejecutarse a horas en las que no haya nadie en la empresa, que tengan que hacer múltiples ediciones en una base de datos determinada, se tengan que repetir muchas veces, debamos obtener un informe tipo para un grupo de usuarios del sistema gestor,… Este tipo de situaciones es el apropiado para el uso de lenguajes de scripting que nos ayuden en la tarea. Lenguajes que pueden ser el nativo de la gestión de la información en las bases de datos por parte del sistema gestor de bases de datos, como SQL, uno de los que utiliza el sistema operativo en su línea de comandos, como BASH, o un lenguaje de alto nivel distinto, como pueda ser Python. En esta práctica vamos a practicar el uso de un lenguaje representativo de cada una de las posibilidades: Lenguaje de manipulación de datos del sistema gestor de bases de datos, lenguaje de comandos del sistema operativo y lenguaje de alto nivel. El DML (lenguaje de manipulación de datos) del sistema gestor de bases de datos deberá usarse para preparar el script que se ejecutará en el sistema gestor, que será montado con la ayuda de BASH o Python, según corresponda.

Ejercicios:

1. Al dar de alta a los usuarios en el sistema, el administrador del servidor insertó en los mismos la información del nombre completo del usuario, con el nombre y los dos apellidos, del número de despacho, del número de teléfono del trabajo, del número de teléfono de casa y del puesto de trabajo en la empresa. Sin embargo estos datos no los tienen guardados en recursos humanos. Para facilitar la tarea a recursos humanos se deberá crear una tabla que permita almacenar estos datos, y realizar un script que los obtenga del sistema y los inserte en la tabla automáticamente.

Para la realización se tendrá en cuenta que los usuarios se han dado de alta en el sistema usando “adduser”, y que existe la siguiente correspondencia entre los parámetros que hay que insertar en el sistema y los que se guardarán la tabla correspondiente:

Nombre completo:	nombre	primer_apellido	segundo_apellido
Número de habitación:	despacho		
Teléfono del trabajo:	tlf_trabajo		
Teléfono de casa:	tlf_casa		
Otro:	puesto	(Ej.: Director de Finanzas)	

También deberá haber un campo que sea el nombre_usuario, que será el nombre de usuario que tenga el usuario correspondiente en el sistema, y el ID, que será la clave primaria, y corresponderá al ID que tenga el usuario en el sistema.

Se deberá realizar un script que cree la tabla (**0’25 puntos**), y otro que recoja los datos de cada usuario del sistema y los inserte en la tabla (**1’25 puntos**). Para facilitar la tarea sólo se guardarán aquellos usuarios que incluyan la palabra “director” en su nombre de usuario. Se deberán explicar los pasos seguidos. El script debe programarse en BASH, y ejecutarse en UBUNTU 18.04 LTS.

2. Para este ejercicio se debe instalar en el sistema "auditd", que nos sirve para auditar el uso del sistema por parte de los usuarios del mismo:

```
sudo apt install auditd
```

Se ha de comprobar que está activo, con "systemctl", y que existe el archivo "/var/log/audit/audit.log". Ese archivo genera entradas de tipo "USER_START" y "USER_END", cada vez que un usuario entra o sale del sistema, indicando el nombre del usuario en el campo "acct".

Se pide que se prepare un script que sea ejecutado todos los días a las doce de la noche, que compruebe todos los usuarios que en su nombre contengan la palabra "director" que hayan empezado a usar el sistema desde la última ejecución del script. Para esos usuarios se deberá crear una entrada por cada comienzo de utilización del sistema por parte de un usuario. La entrada deberá llevar un id, el nombre y la fecha de comienzo de utilización del sistema en una tabla creada al efecto. La tabla deberá estar previamente creada en la base de datos correspondiente.

Se deberá facilitar el script de creación de la tabla (**0'25 puntos**), el archivo de texto con la copia del "crontab" que permita la ejecución diaria del script, a las doce de la noche (**0'25 puntos**), el script que se deberá ejecutar diariamente y un script por cada paso por separado, como se detalla en los párrafos siguientes. También se deberá explicar cómo se generará el crontab, y las decisiones tomadas para realizar el script de ejecución diaria (**1 punto**).

Para facilitar la tarea se subdividirá el script principal en varios pasos:

2.1. En el primero se mostrarán todos los nombres de los usuarios que hayan comenzado a usar el sistema en ese día, uno debajo de otro, en pantalla. Los usuarios pueden haberse conectado varias veces a distintas horas. (**1 punto**)

2.2. Se debe obtener el tiempo UNIX (segundos desde el 1 de enero de 1970) por cada comienzo del sistema. Se debe mostrar exclusivamente el tiempo UNIX. Se sacará por pantalla uno debajo de otro, con todos los comienzos del sistema de todos los usuarios, independientemente de si el mismo usuario ha empezado varias veces. (**1 punto**)

2.3. Se debe obtener un hash MD5 que sirva como ID para cada una de las entradas. Para obtener el hash se debe usar la siguiente frase: "nombre_usuario tiempo_UNIX". Entre el nombre del usuario y el tiempo UNIX debe haber un espacio, y no deben existir más caracteres. La salida del hash debe ser exclusivamente el hash, sin espacios ni guiones. El script mostrará, exclusivamente, todos los hash obtenidos, uno debajo de otro. (**0'5 puntos**)

2.4. Script con todos los pasos, la preparación de las inserciones en la tabla de la base de datos, y la ejecución en el sistema gestor de bases de datos del script SQL creado en el script BASH. (**1'5 puntos**)

3. Se deberán realizar dos script en Python:

3.1. Uno que permita realizar un informe que muestre los tres usuarios que más veces se han conectado al sistema en el último mes, junto con el número de veces que se han conectado, ordenados de mayor a menor. El informe deberá llevar el mismo formato que el archivo "informe_3_1.txt" que se facilita junto con este enunciado. La salida debe mostrarse en pantalla y guardarse en un archivo de texto que se llame "informe_3_1.txt" (**1'75 puntos**)

3.2. Otro que permita realizar un informe que muestre todos los nombres y las fechas de los usuarios que se hayan conectado el día anterior, ordenados por nombre y por fecha de conexión. El informe deberá llevar el mismo formato que el archivo "informe_3_2.txt" que se facilita junto con este enunciado. (**1'25 puntos**)

Como ayuda para la realización de los scritps Python, para gestionar fechas en Python se utiliza la siguiente librería:

```
from datetime import datetime,timedelta

# Se recoge el dia actual y se le resta un día. Ojo, se resta un día a la hora en el momento de ser
# ejecutado.
dia = (datetime.today() - timedelta(days=1))

# Se recoge el día actual y se le resta un mes (desde el momento de ser ejecutado).
mes = (datetime.today() - timedelta(months=1))
```

Se entregarán los dos scripts y los dos archivos de texto con una prueba de los mismos. En el programa Python, en caso de usar Python2, no se tiene por qué gestionar los acentos, se puede escribir sin ellos. Se debe

Toda la entrega se hará en un sólo archivo comprimido, en el que figuren todos los archivos, tanto los scripts, como los archivos de texto, como un archivo pdf con las explicaciones oportunas. Para los scripts python se debe incluir en el pdf, junto con las explicaciones, capturas de pantalla con las ejecuciones y las salidas dadas. El archivo comprimido deberá llevar el siguiente formato: nombre_primerApellido_script

Ej.: jose_perez_script.zip