Introducción

En este laboratorio, aprenderemos a realizar una de las principales tareas de un administrador de bases de datos: la implantación de una política de copias de seguridad y el establecimiento de los mecanismos necesarios de recuperación. Para esto utilizaremos una base de datos de ejemplo disponible de la web de *MySQL*.

Sobre esta base de datos, primero realizaremos distintas copias de seguridad y posteriormente simularemos distintos fallos que nos obligarán a recuperar la base de datos. Finalmente, también veremos cómo se pueden hacer copias de seguridad de un servidor a otro.

Objetivos

Los objetivos de este laboratorio son:

- Averiguar la localización de los ficheros de datos de nuestras bases de datos.
- Aprender a realizar copias de seguridad de nuestras bases de datos.
- Aprender a recuperar nuestra base de datos a un estado consistente después de un fallo que provoca pérdida de información en disco (memoria secundaria).

Requisitos previos

Para realizar este laboratorio utilizaremos el servidor de bases de datos que obtuvimos como resultado del Laboratorio 1. El servidor tiene que estar correctamente configurado para ser accesible remotamente.

IMPORTANTE, ENTREGA

Para finalizar, en lo que respecta a la entrega del presente laboratorio, no se realizará un ejercicio específico con dicho fin. Pero sí se deben entregar los resultados de las actividades realizadas durante el laboratorio. Es decir:

- Copia de seguridad
- Copia de seguridad incremental
- Los comandos utilizados durante el punto 5

Procedimiento a seguir

A continuación, se describen los pasos a realizar suponiendo que el servidor de bases de datos está correctamente configurado.

1 Instalación de la base de datos Sakila

El primer paso del laboratorio consistirá en instalar la base de datos que utilizaremos a partir de este momento. La base de datos elegida es s*akila*, y en el siguiente enlace podréis encontrar toda la información relacionada con ella:

http://dev.mysql.com/doc/sakila/en/

Después de iniciar una sesión *SSH* con nuestro servidor, utilizaremos el siguiente enlace para descargarnos la base de datos:

http://downloads.mysql.com/docs/sakila-db.zip

```
ubuntu@ip-172-31-18-109:~$ pwd
/home/ubuntu
ubuntu@ip-172-31-18-109:~$ mkdir Sakila
ubuntu@ip-172-31-18-109:~$ cd Sakila/
ubuntu@ip-172-31-18-109:~/Sakila$ wget http://downloads.mysql.com/docs/sakila-db.zip
--2021-03-08 11:36:21--  http://downloads.mysql.com/docs/sakila-db.zip
Resolving downloads.mysql.com (downloads.mysql.com)... 137.254.60.14
Connecting to downloads.mysql.com (downloads.mysql.com)|137.254.60.14|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://downloads.mysql.com/docs/sakila-db.zip [following]
--2021-03-08 11:36:21--  https://downloads.mysql.com/docs/sakila-db.zip
Connecting to downloads.mysql.com (downloads.mysql.com)|137.254.60.14|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 729384 (712K) [application/octet-stream]
Saving to: 'sakila-db.zip'

sakila-db.zip           100%[=========================>] 712.29K   582KB/s    in 1.2s

2021-03-08 11:36:23 (582 KB/s) - 'sakila-db.zip' saved [729384/729384]

ubuntu@ip-172-31-18-109:~/Sakila$
```

A continuación, descomprimiremos el archivo descargado:

```
:~/Sakila$ unzip sakila-db.zip
```

```
ubuntu@ip-172-31-18-109:~/Sakila$ unzip sakila-db.zip

Command 'unzip' not found, but can be installed with:

sudo apt install unzip

ubuntu@ip-172-31-18-109:~/Sakila$
```

Si obtenéis un error como el que se muestra en la imagen superior, es muy probable que el programa *unzip* no se encuentre instalado y que tengáis que instalarlo. Para esto, utilizar el comando "apt-get". Después, utilizar "unzip" de nuevo para descomprimir el archivo.

```
ubuntu@ip-172-31-18-109:~/Sakila$ sudo apt-get install unzip
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  zip
The following NEW packages will be installed:
  unzip
0 upgraded, 1 newly installed, 0 to remove and 15 not upgraded.
Need to get 169 kB of archives.
After this operation, 593 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 unzip amd64 6.0-25ubun
tu1 [169 kB]
Fetched 169 kB in 0s (6207 kB/s)
Selecting previously unselected package unzip.
(Reading database ... 70766 files and directories currently installed.)
Preparing to unpack .../unzip_6.0-25ubuntu1_amd64.deb ...
Unpacking unzip (6.0-25ubuntu1) ...
Setting up unzip (6.0-25ubuntu1) ...
Processing triggers for mime-support (3.64ubuntu1) ...
Processing triggers for man-db (2.9.1-1) ...
ubuntu@ip-172-31-18-109:~/Sakila$ unzip sakila-db.zip
Archive:  sakila-db.zip
   creating: sakila-db/
  inflating: sakila-db/sakila-data.sql
  inflating: sakila-db/sakila-schema.sql
  inflating: sakila-db/sakila.mwb
ubuntu@ip-172-31-18-109:~/Sakila$
```

Una vez descomprimido el archivo, utilizaremos el cliente de *MySQL* con el usuario *root* para cargar la base de datos Sakila. El primer paso es cargar el fichero "sakila-schema.sql" que contiene la definición del esquema. Utilizar el siguiente comando para cargarlo:

```
:~/Sakila$ mysql -u root -p < sakila-db/sakila-schema.sql
```

```
ubuntu@ip-172-31-18-109:~/Sakila$ mysql -u root -p < sakila-db/sakila-schema.sql
Enter password:
ubuntu@ip-172-31-18-109:~/Sakila$
```

Después, cargar el fichero "sakila-data.sql" para cargar los datos de Sakila:

```
:~/Sakila$ mysql -u root -p < sakila-db/sakila-data.sql
```

```
ubuntu@ip-172-31-18-109:~/Sakila$ mysql -u root -p < sakila-db/sakila-data.sql
Enter password:
ubuntu@ip-172-31-18-109:~/Sakila$
```

Opcionalmente, podéis abrir los ficheros "sakila-schema.sql" y "sakila-data.sql" con un editor de texto (p.e. nano) y observar su contenido para entender cómo se crea la base de datos.

Para comprobar si la base de datos se ha cargado adecuadamente, abrir la consola de MySQL con el usuario “root” y ejecutar los siguientes comandos:

```
mysql> use sakila;
mysql> show tables;
```

```
mysql> use sakila;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+----------------------------+
| Tables_in_sakila           |
+----------------------------+
| actor                      |
| actor_info                 |
| address                    |
| category                   |
```

También podemos explorar el número de filas en un par de tablas de Sakila:

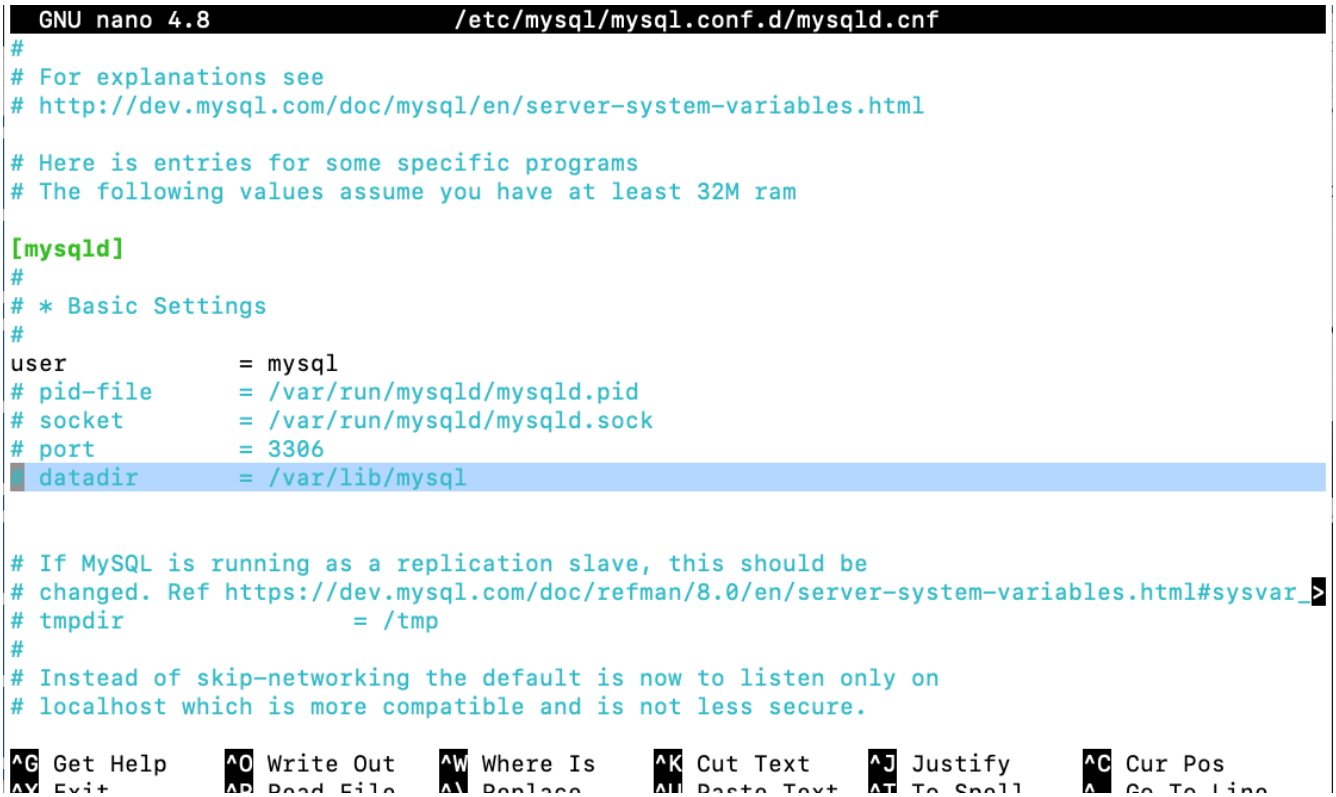
```
mysql> select count(*) from film;
mysql> select count(*) from film_text;
```

```
mysql> select count(*) from film;
+----------+
| count(*) |
+----------+
|     1000 |
+----------+
1 row in set (0.03 sec)

mysql> select count(*) from film_text;
+----------+
| count(*) |
+----------+
|     1000 |
+----------+
1 row in set (0.02 sec)

mysql>
```

2 Averiguar el directorio de almacenamiento de datos de MySQL

Antes de comenzar a realizar nuestra primera copia de seguridad, primero averiguaremos cuál es el directorio utilizado por *MySQL* para guardar la información de nuestras bases de datos.

Para esto, tendremos que consultar la configuración de *MySQL* a través de su archivo de configuración. En concreto, el directorio en el que *MySQL* almacena los archivos de datos se encuentra en la opción *datadir*. Este valor se encuentra comentado por defecto y no debemos modificarlo en este momento:



El directorio de datos de *MySQL* ha de estar convenientemente protegido para que no pueda haber accesos no autorizados a la información de nuestras bases de datos. Para comprobarlo, podemos mostrar los permisos del mismo desde la terminal de Linux:

```
:~$ ls -ld /var/lib/mysql
```

```
ubuntu@ip-172-31-18-109:~/Sakila$ ls -ld /var/lib/mysql
drwx------ 12 mysql mysql 12288 Mar  8 11:42 /var/lib/mysql
ubuntu@ip-172-31-18-109:~/Sakila$
```

Como se puede observar, tan solo el usuario *mysql*, que es el propietario del directorio, puede hacer uso del directorio. Esto implica que ni siquiera podemos acceder al directorio u obtener un listado sin utilizar nuestros privilegios de *root*:

```
:~$ ls -l /var/lib/mysql/
```

```
ubuntu@ip-172-31-18-109:~/Sakila$ ls -l /var/lib/mysql
ls: cannot open directory '/var/lib/mysql': Permission denied
ubuntu@ip-172-31-18-109:~/Sakila$ sudo ls -l /var/lib/mysql
total 471752
-rw-r----- 1 mysql mysql    196608 Mar  8 11:47 '#ib_16384_0.dblwr'
-rw-r----- 1 mysql mysql   8585216 Feb 24 12:08 '#ib_16384_1.dblwr'
drwxr-x--- 2 mysql mysql      4096 Mar  8 11:18 '#innodb_temp'
drwxr-x--- 2 mysql mysql      4096 Mar  4 11:30  DBCoin
drwxr-x--- 2 mysql mysql      4096 Mar  4 11:38  DBet
```

Para acceder a los ficheros de datos de la base de datos *sakila* tendremos que utilizar el mismo método:

```
:~$ sudo ls -l /var/lib/mysql/sakila
```

```
ubuntu@ip-172-31-18-109:~/Sakila$ sudo ls -l /var/lib/mysql/sakila
total 24256
-rw-r----- 1 mysql mysql   131072 Mar  8 11:45 actor.ibd
-rw-r----- 1 mysql mysql   262144 Mar  8 11:45 address.ibd
-rw-r----- 1 mysql mysql   114688 Mar  8 11:45 category.ibd
-rw-r----- 1 mysql mysql   147456 Mar  8 11:45 city.ibd
```

3 Creación de una copia de seguridad completa y recuperación

Para crear una copia de seguridad de nuestra base de datos existen diferentes opciones. Tanto *phpMyAdmin* como otras herramientas similares permiten realizar esta tarea de una forma muy sencilla, aunque la opción más extendida es la aplicación *mysqldump*, que se utiliza directamente desde la ventana de terminal.

La aplicación *mysqldump* nos permite almacenar todas las sentencias *SQL* necesarias para recuperar la base de datos en un único archivo. Para utilizarlo, primero crearemos una nueva carpeta en nuestro directorio personal:

```
:~$ cd ~; mkdir Backup
:~$ cd Backup
```

```
ubuntu@ip-172-31-18-109:~/Sakila$ cd ~; mkdir Backup
ubuntu@ip-172-31-18-109:~$ cd Backup/
ubuntu@ip-172-31-18-109:~/Backup$ pwd
/home/ubuntu/Backup
ubuntu@ip-172-31-18-109:~/Backup$
```

A continuación, realizamos una copia de seguridad completa (todas las bases de datos del servidor) de la siguiente manera (reemplazar AAAA por el año actual, MM por el mes y DD por el día):

```
:~/Backup$ mysqldump -u root -p --all-databases --single-transaction
                          --events > myBackup.AAAA-MM-DD.sql
```

```
ubuntu@ip-172-31-18-109:~/Backup$ mysqldump -u root -p --all-databases --single-transaction
--events > myBackup.2021.03.08.sql
Enter password:
ubuntu@ip-172-31-18-109:~/Backup$ ls -lh myBackup.2021.03.08.sql
-rw-rw-r-- 1 ubuntu ubuntu 4.4M Mar  8 12:03 myBackup.2021.03.08.sql
```

El comando *mysqldump* nos proporciona muchas opciones. En particular, nosotros hemos utilizados las opciones --*all-databases* (todas las bases de datos), --*single-transaction* (solo transacciones confirmadas) y --*events* (incluir eventos). Para conocer el resto de opciones, podéis consultar el manual de *MySQL*:

http://dev.mysql.com/doc/refman/8.0/en/mysqldump.html

Además, también sería recomendable hacer una copia de seguridad de los archivos de configuración (*/etc/mysql/my.cnf y /etc/mysql/mysql.conf.d/mysqld.cnf*).

```
ubuntu@ip-172-31-18-109:~/Backup$ cp /etc/mysql/my.cnf .
ubuntu@ip-172-31-18-109:~/Backup$ cp /etc/mysql/mysql.conf.d/mysql.cnf .
ubuntu@ip-172-31-18-109:~/Backup$ ls -lh
total 4.4M
-rw-r--r-- 1 ubuntu ubuntu  682 Mar  8 12:09 my.cnf
-rw-rw-r-- 1 ubuntu ubuntu 4.4M Mar  8 12:03 myBackup.2021.03.08.sql
-rw-r--r-- 1 ubuntu ubuntu  132 Mar  8 12:10 mysql.cnf
ubuntu@ip-172-31-18-109:~/Backup$
```

A continuación, provocaremos un fallo en el sistema con pérdida de memoria no volátil (disco). Vamos a simular que el directorio de datos de *MySQL* se ha perdido o no se puede acceder. El fallo lo provocaremos borrando el directorio con el siguiente comando:

```
:~/Backup$ sudo rm -rf /var/lib/mysql
```

```
ubuntu@ip-172-31-18-109:~/Backup$ sudo rm -rf /var/lib/mysql
ubuntu@ip-172-31-18-109:~/Backup$ sudo ls -lh /var/lib
total 172K
drwxr-xr-x  4 root      root      4.0K Oct 26 23:27 AccountsService
drwxr-xr-x  2 root      root      4.0K Feb  2 10:52 PackageKit
drw-------  3 root      root      4.0K Feb  2 10:52 amazon
drwxr-xr-x  5 root      root      4.0K Feb  2 11:00 apache2
drwxr-xr-x  5 root      root      4.0K Mar  8 11:47 apt
                                    ...
drwxr-xr-x  2 root      root      4.0K Oct 26 23:29 man-db
drwxr-xr-x  3 root      root      4.0K Feb  2 10:56 mecab
drwxr-xr-x  2 root      root      4.0K Apr 15  2020 misc
drwx------  2 mysql     mysql     4.0K Feb  2 10:56 mysql-files
drwx------  2 mysql     mysql     4.0K Feb  2 10:56 mysql-keyring
drwxr-xr-x  2 root      root      4.0K Jan 27 14:25 mysql-upgrade
drwxr-xr-x  2 root      root      4.0K Feb 23 09:55 os-prober
drwxr-xr-x  2 root      root      4.0K Feb  2 10:55 pam
```

Tras hacer el comando “rm”, el directorio “mysql” ya no se encuentra en /var/lib.

Mediante este comando, hemos eliminado completamente el directorio de datos de nuestra base de datos, con lo cual estaríamos simulando una situación en la cual se nos ha estropeado el disco duro de nuestro servidor.

Para verificar que efectivamente se ha producido un fallo en el servidor, nos conectamos a *MySQL* y comprobamos si la base de datos *sakila* está disponible.

```
mysql> use sakila;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql>
```

Aunque hemos borrado el directorio de datos, Sakila sigue apareciendo como disponible mediante la consola. Podemos verificar que todas sus tablas aparecen listadas:

```
mysql> show tables;
+----------------------------+
| Tables_in_sakila           |
+----------------------------+
| actor                      |
| actor_info                 |
| address                    |
```

…

```
| staff                      |
| staff_list                 |
| store                      |
+----------------------------+
23 rows in set (0.00 sec)

mysql>
```

Nota: depende de la instalación, puede que estos comandos devuelvan respuestas diferentes. En ocasiones puede que no se liste el listado de tablas y en su lugar se devuelva un error. En este caso, pasar directamente a la página 13.

Incluso podemos insertar nuevas tuplas:

```
mysql> insert into sakila.actor(first_name, last_name) values ('Renee', 'Zellweger');
Query OK, 1 row affected (0.03 sec)

mysql>
```

Esto sucede porque el servicio de *MySQL* estaba en marcha desde antes de borrar el directorio de datos y tiene la información de sus datos cargada en memoria.

Probamos a reiniciar el servicio de *MySQL* para forzar que se carguen los datos del directorio. Nos encontramos con que el servicio no es capaz de reiniciarse.

```
ubuntu@ip-172-31-18-109:~/Backup$ sudo service mysql restart
Job for mysql.service failed because the control process exited with error code.
See "systemctl status mysql.service" and "journalctl -xe" for details.
ubuntu@ip-172-31-18-109:~/Backup$ sudo service mysql status
● mysql.service - MySQL Community Server
     Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: enable
     Active: failed (Result: exit-code) since Wed 2021-03-10 07:58:27 UTC; 7s ago
    Process: 1640 ExecStartPre=/usr/share/mysql/mysql-systemd-start pre (code=exited,

Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Control process exited, co
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Failed with result 'exit-c
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: Failed to start MySQL Community Server.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Scheduled restart job, res
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: Stopped MySQL Community Server.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Start request repeated too
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Failed with result 'exit-c
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: Failed to start MySQL Community Server.
```

Aunque conocemos el motivo del problema, en los siguientes pasos vamos a asumir que es desconocido para nosotros y utilizar diferentes herramientas del sistema para encontrar el fallo y solucionarlo.

El primer paso es obtener información del fallo. Para ello, ejecutamos los comandos que sugiere el sistema tras fallar en el reinicio de *MySQL* (utilizar la tecla "q" para salir de ambos).

```
:~/Backup$ systemctl status mysql.service
```

```
ubuntu@ip-172-31-18-109:~/Backup$ systemctl status mysql.service
● mysql.service - MySQL Community Server
     Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: enabled)
     Active: failed (Result: exit-code) since Wed 2021-03-10 07:58:27 UTC; 9min ago
    Process: 1640 ExecStartPre=/usr/share/mysql/mysql-systemd-start pre (code=exited, status=1/FA

Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Control process exited, code=exited,
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Failed with result 'exit-code'.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: Failed to start MySQL Community Server.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Scheduled restart job, restart counte
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: Stopped MySQL Community Server.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Start request repeated too quickly.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Failed with result 'exit-code'.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: Failed to start MySQL Community Server.
```

```
:~/Backup$ journalctl -xe
```

```
ubuntu@ip-172-31-18-109:~/Backup$ journalctl -xe
-- Support: http://www.ubuntu.com/support
--
-- A stop job for unit mysql.service has finished.
--
-- The job identifier is 1478 and the job result is done.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Start request repeated too quickly.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Failed with result 'exit-code'.
-- Subject: Unit failed
-- Defined-By: systemd
-- Support: http://www.ubuntu.com/support
--
-- The unit mysql.service has entered the 'failed' state with result 'exit-code'.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: Failed to start MySQL Community Server.
```

A primera vista no se obtiene gran información de estos comandos.

Como se vio en el laboratorio anterior de la asignatura, *MySQL* incluye un *log* de errores. El siguiente paso es leer sus últimas líneas para buscar más información sobre el problema.

```
ubuntu@ip-172-31-18-109:~/Backup$ tail -n 55 /var/log/mysql/error.log
2021-03-10T07:58:26.344885Z 0 [ERROR] [MY-012593] [InnoDB] The error means the system cannot find the pa
th specified.
2021-03-10T07:58:26.344892Z 0 [Warning] [MY-012093] [InnoDB] Cannot open './sakila/fts_00000000000005bb_
config.ibd'. Have you deleted .ibd files under a running mysqld server?
2021-03-10T07:58:26.344903Z 0 [ERROR] [MY-012157] [InnoDB] Trying to do I/O to a tablespace which exists
 without an .ibd data file. I/O type: read, page: [page id: space=299, page number=4], I/O length: 16384
 bytes
2021-03-10T07:58:26.344911Z 0 [ERROR] [MY-011966] [InnoDB] trying to read page [page id: space=299, page
 number=4] in nonexisting or being-dropped tablespace
2021-03-10T07:58:26.344922Z 0 [ERROR] [MY-011899] [InnoDB] [FATAL] Unable to read page [page id: space=2
99, page number=4] into the buffer pool after 100 attempts. The most probable cause of this error may be
 that the table has been corrupted. Or, the table was compressed with with an algorithm that is not supp
orted by this instance. If it is not a decompress failure, you can try to fix this problem by using inno
```

Entre sus últimas líneas podemos encontrar que el propio *MySQL* ha detectado cuál es el problema: no encuentra los datos asociados a una de sus bases de datos e incluso menciona que ha sucedido mientras el servicio estaba en marcha.

Nuestro servicio *MySQL* está instalado en un sistema Linux y este último también tiene su propio sistema de *logs* para registrar los problemas que sucedan con los procesos en marcha en el sistema. Utilizar el siguiente comando para consultar las últimas líneas del mismo:

```
:~/Backup$ tail -n 15 /var/log/syslog
```

```
ubuntu@ip-172-31-18-109:~/Backup$ tail -n 15 /var/log/syslog
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: Starting MySQL Community Server...
Mar 10 07:58:27 ip-172-31-18-109 mysql-systemd-start[1640]: MySQL data dir not found at /var/lib/mysql.
Please create one.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Control process exited, code=exited, status=
1/FAILURE
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Failed with result 'exit-code'.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: Failed to start MySQL Community Server.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Scheduled restart job, restart counter is at
 5.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: Stopped MySQL Community Server.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Start request repeated too quickly.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: mysql.service: Failed with result 'exit-code'.
Mar 10 07:58:27 ip-172-31-18-109 systemd[1]: Failed to start MySQL Community Server.
```

Podemos leer que el problema de *MySQL* también se ha registrado y en este caso, además, se nos indica una sugerencia de cómo proceder.

Vamos a comenzar la recuperación del sistema usando la información que hemos obtenido hasta ahora. Ya que el log de Linux indica que *MySQL* no puede arrancar por la falta de un directorio de datos, comenzamos creando uno. Los directorios de *MySQL* son gestionados por el usuario del sistema "mysql", así que lo asignamos como su propietario.

```
:~/Backup$ sudo mkdir /var/lib/mysql
:~/Backup$ sudo chown mysql:mysql /var/lib/mysql
```

```
ubuntu@ip-172-31-18-109:~/Backup$ sudo mkdir /var/lib/mysql
ubuntu@ip-172-31-18-109:~/Backup$ sudo chown mysql:mysql /var/lib/mysql
ubuntu@ip-172-31-18-109:~/Backup$ ls -ld /var/lib/mysql
drwxr-xr-x 2 mysql mysql 4096 Mar 10 08:34 /var/lib/mysql
```

Seguidamente, intentamos reiniciar el servicio de *MySQL* de nuevo.

```
ubuntu@ip-172-31-18-109:~/Backup$ sudo service mysql restart
Job for mysql.service failed because the control process exited with error code.
See "systemctl status mysql.service" and "journalctl -xe" for details.
```

El servicio no se inicia y volvemos a obtener un error. En este caso consultaremos directamente el *log* del sistema Linux, ya que es el que más información proporcionó de todos.

```
ubuntu@ip-172-31-18-109:~/Backup$ tail -n 10 /var/log/syslog
Mar 10 08:36:22 ip-172-31-18-109 systemd[1]: Starting MySQL Community Server...
Mar 10 08:36:22 ip-172-31-18-109 mysql-systemd-start[1904]: MySQL system database not found in
/var/lib/mysql. Please run mysqld --initialize.
Mar 10 08:36:22 ip-172-31-18-109 systemd[1]: mysql.service: Control process exited, code=exited
, status=1/FAILURE
Mar 10 08:36:22 ip-172-31-18-109 systemd[1]: mysql.service: Failed with result 'exit-code'.
Mar 10 08:36:22 ip-172-31-18-109 systemd[1]: Failed to start MySQL Community Server.
Mar 10 08:36:23 ip-172-31-18-109 systemd[1]: mysql.service: Scheduled restart job, restart coun
ter is at 5.
Mar 10 08:36:23 ip-172-31-18-109 systemd[1]: Stopped MySQL Community Server.
Mar 10 08:36:23 ip-172-31-18-109 systemd[1]: mysql.service: Start request repeated too quickly.
Mar 10 08:36:23 ip-172-31-18-109 systemd[1]: mysql.service: Failed with result 'exit-code'.
Mar 10 08:36:23 ip-172-31-18-109 systemd[1]: Failed to start MySQL Community Server.
```

Obtenemos un error diferente: *MySQL* no es capaz de encontrar la base de datos que utiliza internamente para gestionar las otras bases que contiene. Sin embargo, se sugiere una solución.

Seguimos las instrucciones que proporciona el log y utilizamos el siguiente comando para inicializar *MySQL* en el directorio que hemos creado. Después, intentamos reiniciar el servicio.

```
:~/Backup$ sudo mysqld --initialize
```

```
ubuntu@ip-172-31-18-109:~/Backup$ sudo mysqld --initialize
ubuntu@ip-172-31-18-109:~/Backup$ sudo service mysql restart
Job for mysql.service failed because the control process exited with error code.
See "systemctl status mysql.service" and "journalctl -xe" for details.
```

El sistema sigue sin reiniciarse. Consultamos de nuevo el *log* de Linux:

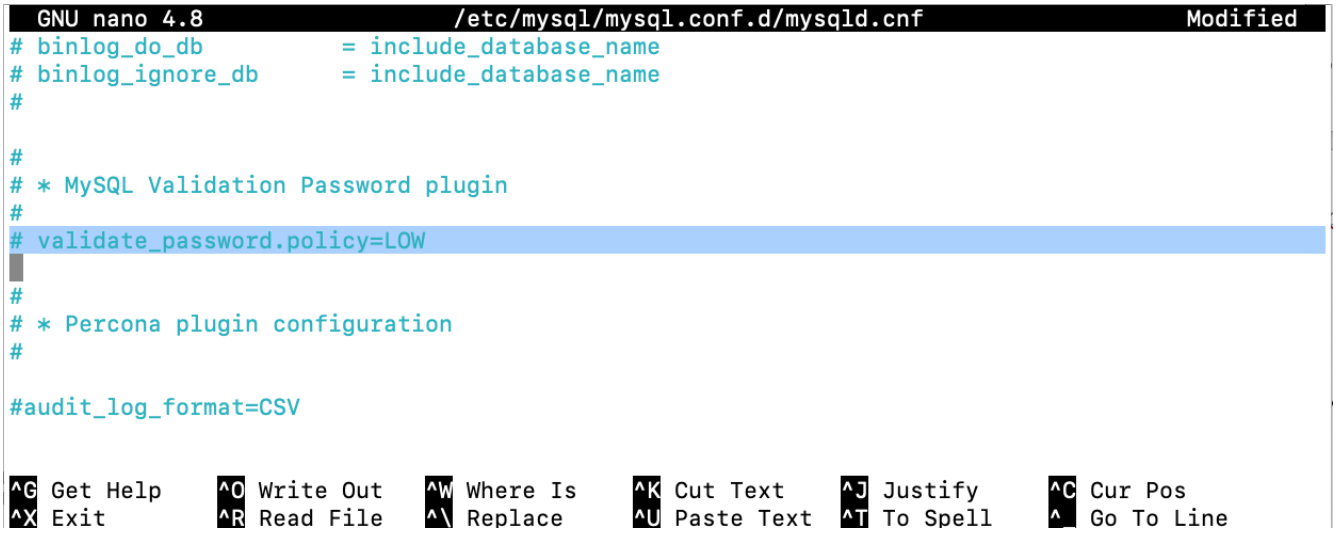
```
ubuntu@ip-172-31-18-109:~/Backup$ tail -n 10 /var/log/syslog
Mar 10 08:46:37 ip-172-31-18-109 systemd[1]: Failed to start MySQL Community Server.
Mar 10 08:46:37 ip-172-31-18-109 systemd[1]: mysql.service: Scheduled restart job, restart coun
ter is at 49.
Mar 10 08:46:37 ip-172-31-18-109 systemd[1]: Stopped MySQL Community Server.
Mar 10 08:46:37 ip-172-31-18-109 systemd[1]: Starting MySQL Community Server...
Mar 10 08:46:40 ip-172-31-18-109 systemd[1]: mysql.service: Main process exited, code=exited, s
tatus=1/FAILURE
Mar 10 08:46:40 ip-172-31-18-109 systemd[1]: mysql.service: Failed with result 'exit-code'.
Mar 10 08:46:40 ip-172-31-18-109 systemd[1]: Failed to start MySQL Community Server.
Mar 10 08:46:40 ip-172-31-18-109 systemd[1]: mysql.service: Scheduled restart job, restart coun
ter is at 50.
Mar 10 08:46:40 ip-172-31-18-109 systemd[1]: Stopped MySQL Community Server.
Mar 10 08:46:40 ip-172-31-18-109 systemd[1]: Starting MySQL Community Server...
```

En este caso no obtenemos ningún error concreto del *log* de Linux, es momento de consultar los otros *logs* que conocemos. Leemos las últimas líneas del log de error de *MySQL*.

```
ubuntu@ip-172-31-18-109:~/Backup$ tail -n 5 /var/log/mysql/error.log
2021-03-10T08:48:51.877220Z 0 [Warning] [MY-010441] [Server] Failed to open optimizer cost cons
tant tables
2021-03-10T08:48:51.877624Z 0 [ERROR] [MY-013129] [Server] A message intended for a client cann
ot be sent there as no client-session is attached. Therefore, we're sending the information to
the error-log instead: MY-001146 - Table 'mysql.component' doesn't exist
2021-03-10T08:48:51.877879Z 0 [Warning] [MY-013129] [Server] A message intended for a client ca
nnot be sent there as no client-session is attached. Therefore, we're sending the information t
o the error-log instead: MY-003543 - The mysql.component table is missing or has an incorrect d
efinition.
2021-03-10T08:48:51.878192Z 0 [ERROR] [MY-000067] [Server] unknown variable 'validate_password.
policy=LOW'.
2021-03-10T08:48:51.878723Z 0 [ERROR] [MY-010119] [Server] Aborting
ubuntu@ip-172-31-18-109:~/Backup$
```

En este *log* encontramos información más concreta: el sistema ha intentado cargar el fichero de configuración pero no reconoce la variable “validate_password.policy”. Esta variable fue configurada como parte de la instalación del *plug-in* Validate-Password para gestionar contraseñas (Laboratorio 2) y como este *plug-in* no se instala por defecto, el sistema no reconoce esta variable.

Para hacer posible el arranque del sistema, comentamos la variable "validate_password.policy" en el fichero de configuración de *MySQL*:



Para evitar problemas, borramos todo lo que el proceso de inicialización hubiera creado en el directorio de datos, repetimos el comando de inicialización de *MySQL* y seguido intentamos reiniciar el proceso de *MySQL*:

```
ubuntu@ip-172-31-18-109:~/Backup$ sudo rm -r /var/lib/mysql/*
ubuntu@ip-172-31-18-109:~/Backup$ sudo mysqld --initialize
ubuntu@ip-172-31-18-109:~/Backup$ sudo service mysql restart
ubuntu@ip-172-31-18-109:~/Backup$
```

Ahora el proceso de *MySQL* ya está en marcha. Accedemos a la consola de *MySQL* para verificar que el sistema funciona correctamente:

```
ubuntu@ip-172-31-18-109:~/Backup$ mysql -u root -p
Enter password:
ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: YES)
ubuntu@ip-172-31-18-109:~/Backup$
```

Sin embargo, no es posible. Si intentamos acceder a la consola con la contraseña de "root" que usábamos anteriormente no funciona porque el comando de inicialización genera una contraseña aleatoria temporal para el usuario "root".

Según se indica en el manual de *MySQL*[1], esta contraseña temporal la podemos encontrar en el fichero de *log* de error de *MySQL*:

```
ubuntu@ip-172-31-18-109:~/Backup$ tail -n 10 /var/log/mysql/error.log
2021-03-10T10:38:47.601356Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
2021-03-10T10:38:48.481634Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
2021-03-10T10:38:50.410230Z 6 [Note] [MY-010454] [Server] A temporary password is generated for root
@localhost: ,H9s_obw0dO?
2021-03-10T10:39:07.696620Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 8.0.23-0ubuntu0
.20.04.1) starting as process 25425
2021-03-10T10:39:07.708040Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
2021-03-10T10:39:08.027296Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
2021-03-10T10:39:08.224971Z 0 [System] [MY-011323] [Server] X Plugin ready for connections. Bind-add
ress: '::' port: 33060, socket: /var/run/mysqld/mysqlx.sock
2021-03-10T10:39:08.385555Z 0 [Warning] [MY-010068] [Server] CA certificate ca.pem is self signed.
2021-03-10T10:39:08.385761Z 0 [System] [MY-013602] [Server] Channel mysql_main configured to support
 TLS. Encrypted connections are now supported for this channel.
2021-03-10T10:39:08.431041Z 0 [System] [MY-010931] [Server] /usr/sbin/mysqld: ready for connections.
 Version: '8.0.23-0ubuntu0.20.04.1'  socket: '/var/run/mysqld/mysqld.sock'  port: 3306  (Ubuntu).
```

NOTA: La contraseña será diferente para cada caso, y puede aparecer en un número de línea diferente del fichero de *log* de error. En ejemplo de la imagen se encuentra entre las 10 últimas líneas, pero podría hacer falta visualizar más.

Utilizamos la contraseña temporal para verificar que tenemos acceso a la consola de *MySQL:*

```
ubuntu@ip-172-31-18-109:~/Backup$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 8.0.23-0ubuntu0.20.04.1

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

[1] Documentación de MySQL 8.0 - Inicialización de la BBDD: https://dev.mysql.com/doc/refman/8.0/en/data-directory-initialization.html#data-directory-initialization-password-assignment

Es momento de utilizar nuestra copia de seguridad para recuperar la base de datos. Utilizar el comando siguiente para recuperar la copia de seguridad que hemos hecho antes (reemplazar AAAA, MM y DD por el año, mes y día actual respectivamente).

```
:~/Backup$ mysql -u root -p < myBackup.AAAA-MM-DD.sql
```

```
ubuntu@ip-172-31-18-109:~/Backup$ mysql -u root -p < myBackup.2021.03.08.sql
Enter password:
Please use --connect-expired-password option or invoke mysql in interactive mode.
```

El proceso de recuperación da un error. Repetir el proceso de recuperación, pero esta vez añadir el parámetro que sugiere el anterior error:

```
ubuntu@ip-172-31-18-109:~/Backup$ mysql -u root -p --connect-expired-password < myBackup.2021.03.08.sql

Enter password:
ERROR 1820 (HY000) at line 24: You must reset your password using ALTER USER statement before executing
 this statement.
```

Estos dos últimos errores suceden porque el proceso de inicialización crea la contraseña temporal de "root" como "caducada" para evitar que se utilice para tareas administrativas. Sólo se permite su uso para iniciar sesión y cambiarla. Desde la consola *MySQL*, utilizamos el siguiente comando para cambiarla:

```
mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY '<contraseña>'
```

Una vez que se ha establecido una nueva contraseña de "root", utilizar el comando para recuperar la base de datos desde la copia de seguridad. Su ejecución puede tardar unos segundos.

```
ubuntu@ip-172-31-18-109:~/Backup$ mysql -u root -p < myBackup.2021.03.08.sql
Enter password:
ubuntu@ip-172-31-18-109:~/Backup$ _
```

Después, comprobar a través de la consola de *MySQL* que la base de datos Sakila está disponible y que

al utilizarla no se genera ningún error:

```
ubuntu@ip-172-31-18-109:~/Backup$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.23-0ubuntu0.20.04.1 (Ubuntu)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| DBCoin             |
| DBet               |
...
| phpmyadmin         |
| sakila             |
| sys                |
+--------------------+
10 rows in set (0.00 sec)

mysql> use sakila
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select count(*) from sakila.actor;
+----------+
| count(*) |
+----------+
|      200 |
+----------+
1 row in set (0.02 sec)
```

El último paso para completar la recuperación es reiniciar el servicio de *MySQL* y comprobar que se mantiene estable. Para ello, abrir la consola de *MySQL* y verificar que el sistema sigue funcionando con normalidad. A partir de ahora la contraseña de “root” a utilizar será la que teníamos inicialmente, antes de comenzar este apartado del laboratorio.

```
ubuntu@ip-172-31-18-109:~/Backup$ sudo service mysql restart
ubuntu@ip-172-31-18-109:~/Backup$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.23-0ubuntu0.20.04.1 (Ubuntu)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| DBCoin             |
| DBet               |
...
| performance_schema |
| phpmyadmin         |
| sakila             |
| sys                |
+--------------------+
10 rows in set (0.00 sec)

mysql>
```

4 Creación de una copia de seguridad incremental y recuperación

El tamaño de la base de datos que estamos usando no es muy grande, con lo cual es posible realizar copias de seguridad completas. Sin embargo, con otras bases de datos (de tamaño mayor), quizás fuese necesario realizar copias de seguridad incrementales.

MySQL permite la creación de copias de seguridad incrementales. Para utilizar esta característica, es necesario tener activado el *log* binario, que viene desactivado por defecto (porque consume recursos). La activación se realiza descomentando las siguientes 3 líneas en el fichero de configuración de MySQL:

A continuación, reiniciamos el servicio de *MySQL* y comprobamos que se han creado los siguientes ficheros (asociados al *log* binario) en la carpeta /var/log/mysql:

```
ubuntu@ip-172-31-18-109:~/Backup$ sudo service mysql restart
ubuntu@ip-172-31-18-109:~/Backup$ ls -l /var/log/mysql/
total 64
-rw-r----- 1 mysql adm   47027 Mar  8 15:14 error.log
-rw-r----- 1 mysql adm    1649 Mar  4 13:43 error.log.1
-rw-r----- 1 mysql adm     644 Feb 23 17:42 error.log.5.gz
-rw-r----- 1 mysql mysql   156 Mar  8 15:14 mysql-bin.000001
-rw-r----- 1 mysql mysql    32 Mar  8 15:14 mysql-bin.index
-rw-r----- 1 mysql adm       0 Mar  8 11:18 mysql-slow.log
-rw-r----- 1 mysql adm       0 Mar  4 08:05 mysql-slow.log.1
-rw-r----- 1 mysql adm       0 Mar  8 11:18 query.log
-rw-r----- 1 mysql adm       0 Mar  4 08:05 query.log.1
ubuntu@ip-172-31-18-109:~/Backup$
```

A partir de este momento, MySQL registra todos los cambios que realizamos en nuestra base de datos en el *log* binario.

Para verificar que es posible recuperar un estado consistente a partir de una copia completa (la que ya tenemos) y el *log* binario, primero necesitamos realizar algún cambio en la base de datos. Por ejemplo, podemos añadir una nueva entrada en la tabla *actor*:

```
mysql> use sakila;
mysql> insert into actor(first_name, last_name)
                    values ('Renee','Zellweger');
```

```
mysql> use sakila;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> insert into actor(first_name, last_name) values ('Renee', 'Zellweger');
Query OK, 1 row affected (0.00 sec)

mysql> select * from actor order by actor_id desc limit 5;
+----------+------------+-----------+---------------------+
| actor_id | first_name | last_name | last_update         |
+----------+------------+-----------+---------------------+
|      201 | Renee      | Zellweger | 2021-03-08 15:17:54 |
|      200 | THORA      | TEMPLE    | 2006-02-15 04:34:33 |
|      199 | JULIA      | FAWCETT   | 2006-02-15 04:34:33 |
|      198 | MARY       | KEITEL    | 2006-02-15 04:34:33 |
|      197 | REESE      | WEST      | 2006-02-15 04:34:33 |
+----------+------------+-----------+---------------------+
5 rows in set (0.00 sec)
```

Ahora, desde la terminal de Linux, podemos comprobar que la actualización ha sido registrada en el *log* binario.

```
:~/Backup$ sudo cat /var/log/mysql/mysql-bin.000001
```

```
ubuntu@ip-172-31-18-109:~/Backup$ sudo cat /var/log/mysql/mysql-bin.000001
abinT?F`y}8.0.23-0ubuntu0.20.04.1T?F`


**4
(??@T?F`#????"@F`"O? ????9?8??@F` ?Estd???
                                        ??sakilaBEGINs??m"@F`B|Xsakilaactor??????@?"@F`:?
X??Renee        Zellweger`F@"O?<?"@F`?Z
                                    W?ubuntu@ip-172-31-18-109:~/Backup$
ubuntu@ip-172-31-18-109:~/Backup$
```

Ya que los datos de este *log* están en un formato ininteligible, podemos utilizar la herramienta *mysqlbinlog* para mostrar el contenido en texto plano. El comando siguiente incluye "| tail -n 15" al final para mostrar las últimas 15 líneas del *log* binario, pero podéis utilizar un valor diferente a 15:

```
:~/Backup$ sudo mysqlbinlog -v /var/log/mysql/mysql-bin.000001
                            | tail -n 15
```

```
ubuntu@ip-172-31-18-109:~/Backup$ sudo mysqlbinlog -v /var/log/mysql/mysql-bin.000001 | tail -n 15
'/*!*/;
### INSERT INTO `sakila`.`actor`
### SET
###   @1=201
###   @2='Renee'
###   @3='Zellweger'
###   @4=1615216674
# at 438
#210308 15:17:54 server id 1  end_log_pos 469 CRC32 0xed570c5a  Xid = 30
COMMIT/*!*/;
SET @@SESSION.GTID_NEXT= 'AUTOMATIC' /* added by mysqlbinlog */ /*!*/;
DELIMITER ;
# End of log file
/*!50003 SET COMPLETION_TYPE=@OLD_COMPLETION_TYPE*/;
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=0*/;
ubuntu@ip-172-31-18-109:~/Backup$
```

Esta misma herramienta *mysqlbinlog* nos va a permitir recuperar el último estado consistente de la base de datos después de un error. Para ello, el primero paso es copiar los ficheros del *log* binario a nuestro directorio *Backup*:

```
:~/Backup$ sudo cp -v /var/log/mysql/mysql-bin.* .
```

```
ubuntu@ip-172-31-18-109:~/Backup$ sudo cp -v /var/log/mysql/mysql-bin.* .
'/var/log/mysql/mysql-bin.000001' -> './mysql-bin.000001'
'/var/log/mysql/mysql-bin.index' -> './mysql-bin.index'
ubuntu@ip-172-31-18-109:~/Backup$ ls -lh
total 4.5M
-rw-r--r-- 1 ubuntu ubuntu  682 Mar  8 12:09 my.cnf
-rw-rw-r-- 1 ubuntu ubuntu 4.4M Mar  8 12:03 myBackup.2021.03.08.sql
-rw-r----- 1 root   root    469 Mar  8 15:36 mysql-bin.000001
-rw-r----- 1 root   root     32 Mar  8 15:36 mysql-bin.index
-rw-r--r-- 1 ubuntu ubuntu  132 Mar  8 12:10 mysql.cnf
```

A continuación, vamos a provocar un fallo con pérdida de memoria permanente, pero esta vez sólo sobre la carpeta de *Sakila*.

```
ubuntu@ip-172-31-18-109:~/Backup$ sudo rm -rf /var/lib/mysql/sakila
ubuntu@ip-172-31-18-109:~/Backup$ sudo ls /var/lib/mysql/sakila
ls: cannot access '/var/lib/mysql/sakila': No such file or directory
ubuntu@ip-172-31-18-109:~/Backup$
```

Como hemos hecho anteriormente, recuperamos la base de datos utilizando la copia de seguridad completa (igual que en el apartado anterior de este laboratorio). En caso de que suceda un error (como en la imagen inferior), reiniciar el servicio MySQL.

Por último, recuperamos las actualizaciones realizadas después de crear la copia de seguridad completa utilizando la salida del proceso *mysqlbinlog* como entrada de *MySQL*. Si no hemos cambiado los permisos del fichero que almacena el *log* binario, tendremos que utilizar el comando *sudo*:

```
:~/Backup$ sudo mysqlbinlog mysql-bin.000001 | mysql -u root -p
```

```
ubuntu@ip-172-31-18-109:~/Backup$ mysql -u root -p < myBackup.2021.03.08.sql
Enter password:
ERROR 3664 (HY000) at line 1769: Failed to set SDI 'sakila.film_actor' in tablespace 'sakila/film_
actor'.
ubuntu@ip-172-31-18-109:~/Backup$ sudo service mysql restart
ubuntu@ip-172-31-18-109:~/Backup$ mysql -u root -p < myBackup.2021.03.08.sql
Enter password:
ubuntu@ip-172-31-18-109:~/Backup$ sudo mysqlbinlog mysql-bin.000001 | mysql -u root -p
Enter password:
ubuntu@ip-172-31-18-109:~/Backup$
```

Una vez completado, utilizar la consola de MySQL para verificar que la base de datos Sakila se ha recuperado correctamente y la última entrada que incluimos en la tabla “actor” está presente:

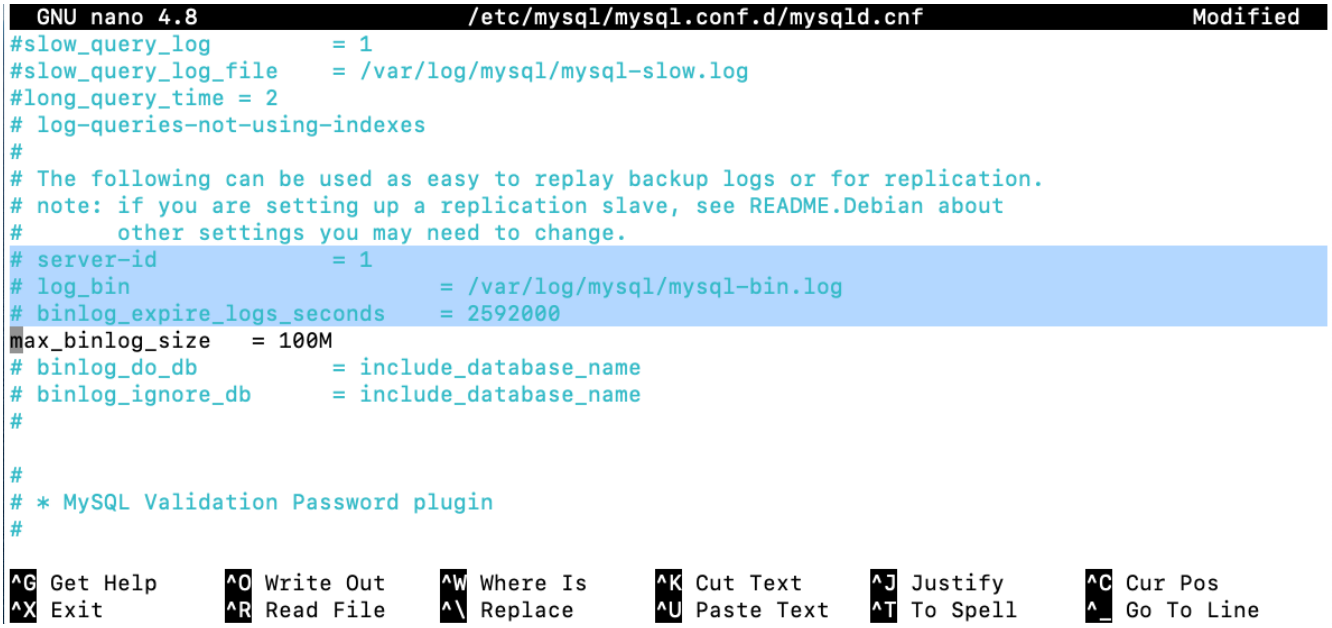
```
mysql> use sakila;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from actor order by actor_id desc limit 5;
+----------+------------+-----------+---------------------+
| actor_id | first_name | last_name | last_update         |
+----------+------------+-----------+---------------------+
|      201 | Renee      | Zellweger | 2021-03-08 15:17:54 |
|      200 | THORA      | TEMPLE    | 2006-02-15 04:34:33 |
|      199 | JULIA      | FAWCETT   | 2006-02-15 04:34:33 |
|      198 | MARY       | KEITEL    | 2006-02-15 04:34:33 |
|      197 | REESE      | WEST      | 2006-02-15 04:34:33 |
+----------+------------+-----------+---------------------+
5 rows in set (0.00 sec)

mysql>
```

La aplicación *mysqlbinlog* nos permite indicar el periodo del cual queremos recuperar las actualizaciones, facilitando de esta manera la recuperación incremental. Sin embargo, en este laboratorio no veremos todas las posibilidades que nos ofrece *mysqlbinlog*.

Si tenéis interés en conocer todas las opciones relaciones con esta cuestión, podéis consultar el manual de *MySQL*:

https://dev.mysql.com/doc/refman/8.0/en/point-in-time-recovery.html

IMPORTANTE: Al acabar de trabajar con el *log* binario en este laboratorio, es muy recomendable desactivarlo para evitar problemas de espacio en disco. Para ello, editar el fichero de configuración y volver a comentar las 3 líneas referidas al *log* binario.



Después de haber editado el fichero, reiniciar el servicio de MySQL:

```
ubuntu@ip-172-31-18-109:~/Backup$ sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf
ubuntu@ip-172-31-18-109:~/Backup$ sudo service mysql restart
ubuntu@ip-172-31-18-109:~/Backup$
```

5 Copias de seguridad entre servidores

Hoy en día ya no es tan común utilizar dispositivos físicos (cintas, discos ópticos, discos duros, …) para realizar copias de seguridad. En su lugar, cada vez es más frecuente realizar la copia de seguridad de un servidor en otro servidor.

En este último apartado se indica cómo hacer una copia de la base de datos Sakila entre 2 servidores que dispongan de MySQL. En las imágenes que se muestran a continuación, la terminal puede tener 2 colores diferentes:

- Fondo blanco y letra negra: servidor emisor que contiene la base de datos a respaldar.
- Fondo azul y letra blanca: servidor receptor de la copia.

El primer paso para realizar la copia entre servidores es asegurarse de que no existe la base de datos a copiar en el receptor. En la siguiente imagen se muestra cómo se borra la base de datos Sakila del servidor receptor.

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| auditingAB         |
| example            |
| information_schema |
| mysql              |
| performance_schema |
| phpmyadmin         |
| sakila             |
| sys                |
+--------------------+
8 rows in set (0.00 sec)

mysql> drop database sakila;
Query OK, 23 rows affected (0.28 sec)
```

En el servidor que queremos respaldar, podemos utilizar el siguiente comando para realizar un respaldo de MySQL y enviarlo a través de SSH al servidor receptor:

```
:~/Backup$ mysqldump -u root -p'<passwordSender>' --databases <database-name>
                          --single-transaction --events |  ssh          -i
<keys.pem> <user>@<backup-server-address> mysql -u root -p'<passwordRec>'
```

En este comando hay varias variables a reemplazar:

- <passwordSender>: Contraseña de "root" de MySQL del servidor a respaldar.
- <database-name>: Nombre de la BBDD a respaldar.
- <user>: Nombre de usuario a utilizar en el servidor receptor.
- <backup-server-address>: IP o nombre del servidor receptor.
- <passwordRec>: Contraseña "root" de MySQL del servidor receptor.

Los avisos "Warning" que aparecen en la imagen son debidos a que estamos escribiendo las contraseñas de los usuarios como parte del comando y estos quedarían registrados en el histórico de comandos de la terminal Linux por defecto.

En este laboratorio hacemos esto por simplicidad, pero en un entorno real las contraseñas deberían leerse directamente de un fichero o de variables de entorno para mayor seguridad.

Una vez realizada la copia, verificar en el servidor receptor que la base de datos Sakila existe y es accesible:

```
mysql> use sakila;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select count(*) from sakila.actor;
+----------+
| count(*) |
+----------+
|      201 |
+----------+
1 row in set (0.02 sec)

mysql> select * from actor order by actor_id desc limit 5;
+----------+------------+-----------+---------------------+
| actor_id | first_name | last_name | last_update         |
+----------+------------+-----------+---------------------+
|      201 | Renee      | Zellweger | 2021-03-08 15:17:54 |
|      200 | THORA      | TEMPLE    | 2006-02-15 04:34:33 |
|      199 | JULIA      | FAWCETT   | 2006-02-15 04:34:33 |
|      198 | MARY       | KEITEL    | 2006-02-15 04:34:33 |
|      197 | REESE      | WEST      | 2006-02-15 04:34:33 |
+----------+------------+-----------+---------------------+
5 rows in set (0.00 sec)
```