PRÁCTICA 5:
REPLICACIÓN DE BASES DE DATOS MYSQL

Iñaki Melguizo Marcos

1. INTRODUCCIÓN

El objetivo de esta práctica va a ser seguir con la configuración de nuestra granja web. Concretamente, vamos a crear una base de datos y vamos a insertarle datos por la línea de comandos. Posteriormente vamos a hacer copias de seguridad de nuestras bases de datos, primero haciendo una copia de archivos de archivos de copia de seguridad de BD de la máquina m1-imm98 a m2-imm98 mediante ssh, más tarde configurando una estructura maestro-esclavo para tener así un clonado automático de m1-imm98 a m2-imm98 y por último una configuración maestro-maestro para que el clonado sea bidireccional.

Este clonado de las bases de datos añade fiabilidad en cuanto a fallos del sistema además de que no afecta al rendimiento del maestro.

2. Base datos MySQL comandos

Primeramente, vamos a desactivar todas las reglas del cortafuegos para permitir todo el tráfico. Esto lo hacemos ya que en caso de haber un fallo durante la realización de la práctica, ya sabemos que no es fallo del cortafuegos.

Para ello teníamos un script de la práctica anterior llamado *aceptar_todo.sh* que aceptaba todo el tráfico entrante y saliente (INPUT y OUTPUT). Lo ejecutamos con el siguiente comando:

sudo ./aceptar_todo.sh

```
imm98@m1-imm98:~$ sudo ./aceptar_todo.sh
[sudo] password for imm98:
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

Una vez desactivado el cortafuegos vamos a crear la base de datos. Como root, nos conectamos al servidor SQL con el comando:

sudo mysql -u root -p

```
imm98@m1-imm98:~$ sudo mysql -u root -p
[sudo] password for imm98:
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.33-0ubuntu0.18.04.1 (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>
```

Vamos a crear la que será nuestra base de datos para toda esta práctica. Se llamará *estudiante* y la crearemos con el siguiente comando:

create database estudiante;

```
mysql> create database estudiante
    -> ;
Query OK, 1 row affected (0.00 sec)
```

Seleccionamos la base de datos *estudiante* con el comando:

use estudiante;

```
mysql> use estudiante;
Database changed
```

Mostramos las tablas de la base de datos *estudiante*. Como la acabamos de crear, obviamente no hay ninguna tabla. Mostramos las tablas con el comando:

show tables;

```
mysql> show tables;
Empty set (0.00 sec)
```

Vamos a crear una tabla llamada *datos*, que será la tabla que utilizaremos durante la práctica. Esta tabla contendrá 4 atributos, todos de tipo varchar, que serán: nombre, apellidos, usuario y email. Para crearla utilizaremos el comando:

create table datos (nombre varchar(100), apellidos varchar(100), usuario varchar(100), email varchar(100));

```
mysql> create table datos (nombre varchar(100), apellidos varchar(100), usuario
varchar(100), email varchar(100));
Query OK, 0 rows affected (0.16 sec)
```

Volvemos a ejecutar el comando show tables, y se nos muestra la tabla que acabamos de crear, llamada *datos*.

```
mysql> show tables;
+----------------------+
| Tables_in_estudiante |
+----------------------+
| datos                |
+----------------------+
1 row in set (0.00 sec)
```

Vamos ahora a introducir una tupla en esa tabla, en la que voy a introducir mis datos personales. Para ello utilizaremos la sentencia:

***insert** into datos (nombre, apellidos, usuario, email) values (“Inaki”, “Melguizo Marcos”, “imm98”, “imm98@correo.ugr.es”);*

```
mysql> insert into datos (nombre, apellidos, usuario, email) values ("Inaki", "M
elguizo Marcos", "imm98", "imm98@correo.ugr.es");
Query OK, 1 row affected (0.04 sec)
```

Para ver las tuplas de la tabla datos ejecutaremos el comando:

*select * from datos;*

```
mysql> select * from datos;
+--------+-----------------+---------+---------------------+
| nombre | apellidos       | usuario | email               |
+--------+-----------------+---------+---------------------+
| Inaki  | Melguizo Marcos | imm98   | imm98@correo.ugr.es |
+--------+-----------------+---------+---------------------+
1 row in set (0.00 sec)
```

donde se nos mostrará la tupla que habíamos añadido previamente.

Para ver los diferentes campos de una tabla utilizaremos el comando:

describe datos;

```
mysql> describe datos;
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| nombre    | varchar(100) | YES  |     | NULL    |       |
| apellidos | varchar(100) | YES  |     | NULL    |       |
| usuario   | varchar(100) | YES  |     | NULL    |       |
| email     | varchar(100) | YES  |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

donde podemos ver los campos mencionados anteriormente con los que habíamos creado la tabla *datos*

Nos desconectamos de la base datos con el comando *quit*

```
mysql> quit
Bye
imm98@m1-imm98:~$
```

Como opciones avanzadas tenemos que podemos modificar las tuplas de la base de datos mediante la opción **update**. Por ejemplo para modificar el email del usuario cuyo email es "imm98@correo.ugr.es" por "pepe@ugr.es" se haría con el siguiente comando:

update datos set email="pepe@ugr.es" where email="imm98@correo.ugr.es"

```
mysql> update datos set email="pepe@ugr.es" where email="imm98@correo.ugr.es";
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Vemos que el cambio se ha realizado correctamente con el comando:

*select * from datos;*

```
mysql> select * from datos;
+--------+-----------------+---------+-------------+
| nombre | apellidos       | usuario | email       |
+--------+-----------------+---------+-------------+
| Inaki  | Melguizo Marcos | imm98   | pepe@ugr.es |
+--------+-----------------+---------+-------------+
1 row in set (0.00 sec)
```

Otra opción que tenemos es **delete** que borra las tuplas que cumplan las condiciones que establezcas. Por ejemplo, para borrar la tupla cuyo nombre es Inaki se haría con el siguiente comando:

delete from datos where nombre=”Inaki”;

```
mysql> delete from datos where nombre="Inaki";
Query OK, 1 row affected (0.04 sec)
```

Y vemos que se ha borrado correctamente con el comando:

*select * from datos;*

```
mysql> select * from datos;
Empty set (0.00 sec)
```

Otra opción es **drop table** que borra la tabla de base de datos que le indiquemos como parámetro. Para borrar la tabla *datos* debemos de ejecutar el comando:

drop table datos;

```
mysql> drop table datos;
Query OK, 0 rows affected (0.09 sec)
```

Y vemos que se ha borrado completamente con el comando *show tables*

```
mysql> show tables;
Empty set (0.00 sec)
```

3. Replicar una BD MySQL con mysqldump

La herramienta **mysqldump** nos permite clonar las BD que tenemos en nuestra propia máquina. Lo utilizaremos para realizar copias de seguridad de nuestra BD volcando en archivos de extensión .sql y posteriormente pasándoselos a la máquina m2-imm98. Dado que pueden estar modificándose constantemente los datos hay que evitar que se acceda a la BD en el momento de hacer la copia en el archivo .sql. Para realizar esto debemos bloquear las tablas.

Por tanto, accedemos al servidor mysql con el comando visto ya anteriormente:

sudo mysql -u root -p

```
imm98@m1-imm98:~$ sudo mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.7.33-0ubuntu0.18.04.1 (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.
```

Bloqueamos las tablas con la sentencia:

FLUSH TABLES WITH READ LOCK;

```
mysql> FLUSH TABLES WITH READ LOCK;
Query OK, 0 rows affected (0.00 sec)
```

Nos desconectamos del servidor SQL.

```
mysql> quit
Bye
```

Ahora que ya tenemos bloqueadas las tablas, podemos “copiar” la base de datos “estudiante” a un archivo sql que llamaremos en el directorio /tmp/ que llamaremos **estudiante.sql**. Para realizar esta acción ejecutaremos el comando:

sudo mysqldump estudiante -u root -p >/tmp/estudiante.sql

```
imm98@m1-imm98:/tmp$ sudo mysqldump estudiante -u root -p >/tmp/estudiante.sql
Enter password:
```

Una vez que hemos copiado la base de datos *estudiante,* podemos desbloquear las tablas de SQL. Para ello ejecutamos la sentencia:

UNLOCK TABLES;

```
imm98@m1-imm98:~$ sudo mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 6
Server version: 5.7.33-0ubuntu0.18.04.1 (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> UNLOCK TABLES;
Query OK, 0 rows affected (0.00 sec)

mysql> quit;
Bye
```

Vemos que se ha creado correctamente el fichero /tmp/estudiante.sql con el comando:

ls /tmp/

```
imm98@m1-imm98:/tmp$ ls
estudiante.sql
systemd-private-9ca491baea084d53b387657853c7b0fe-apache2.service-Mzw2CJ
systemd-private-9ca491baea084d53b387657853c7b0fe-systemd-resolved.service-QTv4oC
systemd-private-9ca491baea084d53b387657853c7b0fe-systemd-timesyncd.service-OJsIKG
```

Una vez que hemos visto que el archivo se ha creado correctamente vamos a copiar el archivo *estudiante.sql de m1-imm98 a m2-imm98* mediante scp. Para ello utilizamos el comando:

sudo scp estudiante.sql imm98@192.168.56.104:/tmp/estudiante.sql

```
imm98@m1-imm98:/tmp$ sudo scp estudiante.sql imm98@192.168.56.104:/tmp/estudiante.sql
imm98@192.168.56.104's password:
estudiante.sql                                100% 1986     1.2MB/s   00:00
```

Vemos que se ha copiado bien a la máquina m2-imm98 con el comando:

ls /tmp/

```
imm98@m2-imm98:/tmp$ ls
estudiante.sql
systemd-private-abfe10e5840a46b28765a1b8fafc1037-apache2.service-3yUQeF
systemd-private-abfe10e5840a46b28765a1b8fafc1037-systemd-resolved.service-7UbVXy
systemd-private-abfe10e5840a46b28765a1b8fafc1037-systemd-timesyncd.service-VhkKOD
```

Ya que la orden mysqldump no ha introducido en el *archivo estudiante.sql* la sentencia para crear la Base de datos, es necesario que la creemos manualmente. Para ello, nos conectamos al servidor mysql en la máquina m2-imm98:

```
imm98@m2-imm98:/tmp$ sudo mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.33-0ubuntu0.18.04.1 (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>
```

Creamos la base de datos estudiante con el comando:

create database estudiante;

```
mysql> create database estudiante;
Query OK, 1 row affected (0.01 sec)
```

Nos desconectamos del servidor mysql con *quit*

```
mysql> quit;
Bye
imm98@m2-imm98:/tmp$
```

Y ya restauramos o volcamos los datos de la base de datos *estudiante* con el comando:

sudo mysql -u root -p estudiante < /tmp/estudiante.sql

```
imm98@m2-imm98:/tmp$ sudo mysql -u root -p estudiante < /tmp/estudiante.sql
Enter password:
imm98@m2-imm98:/tmp$
```

Vamos a comprobar que tanto las tablas como las tuplas de la base de datos *estudiante* se han volcado correctamente a la máquina m2-imm98:

```
mysql> use estudiante;
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_estudiante |
+----------------------+
| datos                |
+----------------------+
1 row in set (0.00 sec)

mysql> select * from datos;
+--------+-----------------+---------+---------------------+
| nombre | apellidos       | usuario | email               |
+--------+-----------------+---------+---------------------+
| Inaki  | Melguizo Marcos | imm98   | imm98@correo.ugr.es |
+--------+-----------------+---------+---------------------+
1 row in set (0.00 sec)
```

Y vemos que efectivamente el volcado se ha realizado satisfactoriamente. Es decir, tenemos las mismas tablas (tabla de *datos)* y la misma tupla de la tabla *datos* que teníamos en la máquina m1-imm98.

Como opciones avanzadas tenemos la opción **-p** que indica que bases de datos y tablas queremos almacenar en el archivo -sql. Por ejemplo el comando:

sudo mysqldump -u root -p estudiante datos > copia_avanzada.sql

```
imm98@m1-imm98:/tmp$ sudo mysqldump -u root -p estudiante datos > copia_avanzada.sql;
Enter password:
imm98@m1-imm98:/tmp$ ls
copia_avanzada.sql
```

Copia en el archivo copia_avanzada.sql tan sólo la tabla *datos* de la base de datos *estudiante*. Vamos a verificar que hace esto exáctamente viendo el contenido del archivo con el comando:

cat copia_avanzada.sql

```
imm98@m1-imm98:/tmp$ cat copia_avanzada.sql
```

```
DROP TABLE IF EXISTS `datos`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `datos` (
  `nombre` varchar(100) DEFAULT NULL,
  `apellidos` varchar(100) DEFAULT NULL,
  `usuario` varchar(100) DEFAULT NULL,
  `email` varchar(100) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
/*!40101 SET character_set_client = @saved_cs_client */;
```

Y vamos que efectivamente se ha copiado tan sólo la tabla *datos* de la base de datos *estudiante*.

Otra opción de mysqldump es **--ignore-table** que realiza la copia de una base de datos a un archivo .sql a excepción de la tabla que le indiquemos.

Por ejemplo, el comando:

sudo mysqldump -u root -p --ignore-table=estudiante.datos estudiante > copia_avanazada2.sql

```
imm98@m1-imm98:/tmp$ sudo mysqldump -u root -p --ignore-table=estudiante.datos estudiante > copia_avanzada2.sql;

Enter password:
```

No va a copiar ninguna tabla al archivo *copia_avanzada2.sql* ya que la única tabla que teníamos en la base de datos *estudiante* era *datos* y mediante el comando le hemos indicado que la ignore. Lo comprobamos viendo el contenido del archivo *copia_avanzada2.sql* con el comando:

cat copia_avanzada2.sql

```
imm98@m1-imm98:/tmp$ cat copia_avanzada2.sql
-- MySQL dump 10.13  Distrib 5.7.33, for Linux (x86_64)
--
-- Host: localhost    Database: estudiante
-- ------------------------------------------------------
-- Server version       5.7.33-0ubuntu0.18.04.1

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;

-- Dump completed on 2021-05-05  9:18:28
imm98@m1-imm98:/tmp$
```

4. Configuración Maestro-Esclavo

El problema del proceso de replicación anteriormente es que no es automático, es decir, cada vez que haces un cambio en m1-imm98 debes ejecutar las órdenes descritas con anterioridad para realizar la copia a m2-imm98. Ahora vamos a ver una opción automática provista por MySQL que puede configurar un demonio para replicar la BD sobre un esclavo.

Vamos a seguir una serie de pasos para realizar la configuración Maestro-Esclavo en la que la máquina m1-imm98 será el maestro y la máquina m2-imm98 será el esclavo. Para ello vamos a ir describiendo las tareas a realizar en cada una de las dos máquinas:

- **m1-imm98**

Realizamos la edición del archivo **/etc/mysql/mysql.conf.d/mysqld.cnf** con el comando:

sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf

```
imm98@m1-imm98:/tmp$ sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf
```

Comentamos el parámetro bind-address para que no escuche a ningún servidor

```
#bind-address           = 127.0.0.1
```

Hacemos que los errores los almacena en el fichero /var/log/mysql/error.log

```
log_error = /var/log/mysql/error.log
```

Establecemos que el identificador del servidor sea el 1 y que la dirección del log binario sea /var/log/mysql/bin.log

```
server-id               = 1
log_bin                 = /var/log/mysql/bin.log
```

Una vez que guardemos los cambios tenemos que reiniciar el servicio ya que hemos modificado el archivo de configuración de mysql. Para ello, ejecutamos el comando:

sudo service mysql restart

```
imm98@m1-imm98:~$ sudo service mysql restart
```

Vemos que el servicio se ha reiniciado correctamente:

```
imm98@m1-imm98:~$ sudo service mysql status
● mysql.service - MySQL Community Server
   Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2021-05-05 11:06:45 UTC; 20s ago
  Process: 3302 ExecStart=/usr/sbin/mysqld --daemonize --pid-file=/run/mysqld/mysqld.pid (code=exited, statu
  Process: 3284 ExecStartPre=/usr/share/mysql/mysql-systemd-start pre (code=exited, status=0/SUCCESS)
 Main PID: 3304 (mysqld)
    Tasks: 27 (limit: 1106)
   CGroup: /system.slice/mysql.service
           └─3304 /usr/sbin/mysqld --daemonize --pid-file=/run/mysqld/mysqld.pid

May 05 11:06:44 m1-imm98 systemd[1]: Stopped MySQL Community Server.
May 05 11:06:44 m1-imm98 systemd[1]: Starting MySQL Community Server...
May 05 11:06:45 m1-imm98 systemd[1]: Started MySQL Community Server.
```

- **m2-imm98**

Tenemos que hacer casi las mismas acciones que habíamos realizado en la máquina m1-imm98. Modificamos el archivo **/etc/mysql/mysql.conf.d/mysqld.cnf .**

```
imm98@m2-imm98:~$ sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf
```

La única diferencia será que en vez de ser el identificador del servidor el 1, será el 2. Por tanto establecemos **server-id=2**

```
server-id               = 2
log_bin                 = /var/log/mysql/bin.log
```

Reiniciamos el servicio de mysql y vemos que el reinicio se ha realizado correctamente:

```
imm98@m2-imm98:~$ sudo service mysql restart
imm98@m2-imm98:~$ sudo service mysql status
● mysql.service - MySQL Community Server
   Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: en
   Active: active (running) since Wed 2021-05-05 11:11:16 UTC; 6s ago
  Process: 2020 ExecStart=/usr/sbin/mysqld --daemonize --pid-file=/run/mysqld/my
  Process: 1999 ExecStartPre=/usr/share/mysql/mysql-systemd-start pre (code=exit
 Main PID: 2022 (mysqld)
    Tasks: 27 (limit: 1106)
   CGroup: /system.slice/mysql.service
           └─2022 /usr/sbin/mysqld --daemonize --pid-file=/run/mysqld/mysqld.pid

May 05 11:11:16 m2-imm98 systemd[1]: Starting MySQL Community Server...
May 05 11:11:16 m2-imm98 systemd[1]: Started MySQL Community Server.
```

- **m1-imm98**

Lo que vamos a realizar en la máquina m1-imm98 es crear un usuario esclavo. Primeramente, vamos a conectarnos al servidor de mysql con la orden:

sudo mysql -u root -p

```
imm98@m1-imm98:~$ sudo mysql -u root -p
```

Creamos el usuario esclavo al que llamaremos “esclavo_imm98” y su contraseña será también “esclavo_imm98” con la sentencia:

CREATE USER esclavo_imm98 IDENTIFIED BY ‘esclavo_imm98’;

```
mysql> CREATE USER esclavo_imm98 IDENTIFIED BY 'esclavo_imm98';
Query OK, 0 rows affected (0.03 sec)
```

Ejecutamos la siguiente sentencia para dar los permisos necesarios:

*GRANT REPLICATION SLAVE ON *.* TO ‘esclavo_imm98’@’%’ IDENTIFIED BY ‘esclavo_imm98’*

```
mysql> GRANT REPLICATION SLAVE ON *.* TO 'esclavo_imm98'@'%' IDENTIFIED BY 'esclavo_imm98';
```

Bloqueamos las tablas:

```
mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.02 sec)

mysql> FLUSH TABLES;
Query OK, 0 rows affected (0.03 sec)

mysql> FLUSH TABLES WITH READ LOCK;
Query OK, 0 rows affected (0.00 sec)
```

Ahora que ya tenemos nuestro usuario esclavo, vamos a ver el estado del maestro con la orden:

SHOW MASTER STATUS

```
mysql> SHOW MASTER STATUS;
+------------+----------+--------------+------------------+-------------------+
| File       | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set |
+------------+----------+--------------+------------------+-------------------+
| bin.000001 |     1197 |              |                  |                   |
+------------+----------+--------------+------------------+-------------------+
1 row in set (0.00 sec)
```

Estos campos, *File* y *Position* deberemos usarlos a continuación en la configuración del esclavo.

- **m2-imm98**

Ahora vamos a configurar la máquina que hará como esclava, la máquina m2-imm98. Le tenemos que dar los datos del maestro, es decir la IP de la máquina m1-imm98 que es 192.168.56.105, el valor de master_log_file y el de master_log_pos (ambos mostrados en la captura anterior). Nota : El master_log_pos debería de ser el mismo en las dos capturas

```
mysql> CHANGE MASTER TO
    -> MASTER_HOST='192.168.56.105',
    -> MASTER_USER='esclavo_imm98',
    -> MASTER_PASSWORD='esclavo_imm98',
    -> MASTER_LOG_FILE='bin.000001',
    -> MASTER_LOG_POS=1171, MASTER_PORT=3306;
```

Ya tenemos todo configurado, por lo tanto sólo nos queda iniciar el esclavo y los demonios de MYSQL podrían replicar en la máquina m2-imm98 (esclava) los datos de m1-imm98 (maestra). Ejecutamos la sentencia:

START SLAVE;

```
mysql> START SLAVE;
Query OK, 0 rows affected (0.02 sec)
```

- **m1-imm98**

Desbloqueamos de nuevo las tablas previamente bloqueadas para que puedan añadirse nuevas tablas/tuplas en el maestro. Para ello ejecutamos la sentencia:

UNLOCK TABLES;

```
mysql> UNLOCK TABLES;
Query OK, 0 rows affected (0.00 sec)
```

- **m2-imm98**

Para comprobar que todo funciona correctamente nos vamos a la máquina esclava (m2-imm98) y comprobamos que la variable “Second_Behind_Master” es distinta de NULL. Esto nos indica que no hay ningún error. Para mostrar el estado del esclavo ejecutaremos las sentencia:

SHOW SLAVE STATUS

```
mysql> SHOW SLAVE STATUS\G;
```

```
mysql> SHOW SLAVE STATUS\G;
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.56.105
                  Master_User: esclavo_imm98
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000001
          Read_Master_Log_Pos: 992
               Relay_Log_File: m2-imm98-relay-bin.000002
                Relay_Log_Pos: 320
        Relay_Master_Log_File: mysql-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB:
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 992
              Relay_Log_Space: 530
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
```

Y vemos que el valor de “Seconds_Behind_Master” es 0, por tanto, distinta de NULL.

- **m1-imm98**

Una vez que hemos comprobado que el esclavo se está ejecutando correctamente vamos a insertar una nueva tupla en la tabla *datos*. Para insertarla vamos a utilizar la sentencia SQL insert como se puede ver en la captura de abajo:

```
mysql> select * from datos;
+-------------+-----------------+---------+---------------------+
| nombre      | apellidos       | usuario | email               |
+-------------+-----------------+---------+---------------------+
| Inaki       | Melguizo Marcos | imm98   | imm98@correo.ugr.es |
| Jose Manuel | Soto Hidalgo    | jmsoto  | jmsoto@ugr.es       |
+-------------+-----------------+---------+---------------------+
2 rows in set (0.00 sec)

mysql> insert into datos(nombre,apellidos,usuario,email) values ("Pepe", "Sanchez Martinez", "pepe", "p
epe@ugr.es");
Query OK, 1 row affected (0.06 sec)
```

- **m2-imm98**

Vamos a comprobar que en la máquina m2-imm98 se ha duplicado la información de la máquina m1-imm98. Ejecutamos el comando:

*select * from datos;*

justo antes de añadir la tupla de la captura anterior en m1-imm98 y vemos que en la tabla *datos* solo hay dos tuplas. Volvemos a ejecutar la sentencia *select* anterior después de insertar la tupla en m1-imm98 y vemos que se ha actualizado también la tabla de m2-imm98:

```
mysql> select * from datos;
+-------------+-----------------+---------+---------------------+
| nombre      | apellidos       | usuario | email               |
+-------------+-----------------+---------+---------------------+
| Inaki       | Melguizo Marcos | imm98   | imm98@correo.ugr.es |
| Jose Manuel | Soto Hidalgo    | jmsoto  | jmsoto@ugr.es       |
+-------------+-----------------+---------+---------------------+
2 rows in set (0.00 sec)

mysql> select * from datos;
+-------------+-----------------+---------+---------------------+
| nombre      | apellidos       | usuario | email               |
+-------------+-----------------+---------+---------------------+
| Inaki       | Melguizo Marcos | imm98   | imm98@correo.ugr.es |
| Jose Manuel | Soto Hidalgo    | jmsoto  | jmsoto@ugr.es       |
| Pepe        | Sanchez Martinez | pepe    | pepe@ugr.es         |
+-------------+-----------------+---------+---------------------+
3 rows in set (0.00 sec)
```

5. Configuración Maestro-Maestro

Ahora vamos a realizar la configuración Maestro-Maestro, con la que cualquier cambio realizado en la base de datos de m1-imm98 debería realizarse en m2-imm98 y viceversa. Para ello, como ya teníamos creada la configuración en la que m1-imm98 era el maestro y m2-imm98 era el esclavo ahora vamos a realizar la configuración para que m2-imm98 sea el maestro y m1-imm98 sea el esclavo. De esta forma tendríamos una configuración maestro-maestro.

- **m2-imm98**

Creamos el usuario esclavo al que llamaremos “esclavo2_imm98” y su contraseña será también “esclavo2_imm98” con la sentencia:

CREATE USER esclavo2_imm98 IDENTIFIED BY ‘esclavo2_imm98’;

```
mysql> CREATE USER esclavo2_imm98 IDENTIFIED BY 'esclavo2_imm98';
Query OK, 0 rows affected (0.02 sec)
```

Ejecutamos la siguiente sentencia para dar los permisos necesarios:

*GRANT REPLICATION SLAVE ON *.* TO ‘esclavo2_imm98’@’%’ IDENTIFIED BY ‘esclavo_imm98’*

```
mysql> GRANT REPLICATION SLAVE ON *.* TO 'esclavo2_imm98'@'%' IDENTIFIED BY 'esclavo2_imm98';
Query OK, 0 rows affected, 1 warning (0.02 sec)
```

Mostramos el estado del maestro con la sentencia:

SHOW MASTER STATUS;

```
mysql> SHOW MASTER STATUS;
+------------------+----------+--------------+------------------+-------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set |
+------------------+----------+--------------+------------------+-------------------+
| mysql-bin.000002 |     1034 |              |                  |                   |
+------------------+----------+--------------+------------------+-------------------+
1 row in set (0.00 sec)
```

Y almacenamos los campos de *File* y *Position* ya que nos servirán para configurar la máquina esclava, en este caso m1-imm98.

- **m1-imm98**

En la máquina esclava (m1-imm98) le indicamos que la IP del maestro va a ser la 192.168.56.104 (es decir, la IP de m2-imm98) y que el valor de master_log_file y el de master_log_pos son los mostrados en la captura anterior.

```
mysql> CHANGE MASTER TO MASTER_HOST='192.168.56.104', MASTER_USER='esclavo2_imm98', MASTER_PASSWORD='es
clavo2_imm98', MASTER_LOG_FILE='mysql-bin.000002', MASTER_LOG_POS=1034, MASTER_PORT=3306;
Query OK, 0 rows affected, 2 warnings (0.28 sec)
```

Iniciamos el esclavo con la sentencia:

START SLAVE;

```
mysql> START SLAVE;
Query OK, 0 rows affected (0.02 sec)
```

- **m2-imm98**

Desbloqueamos las tablas con la sentencia:

UNLOCK TABLES;

```
mysql> UNLOCK TABLES;
Query OK, 0 rows affected (0.00 sec)
```

- **m1-imm98**

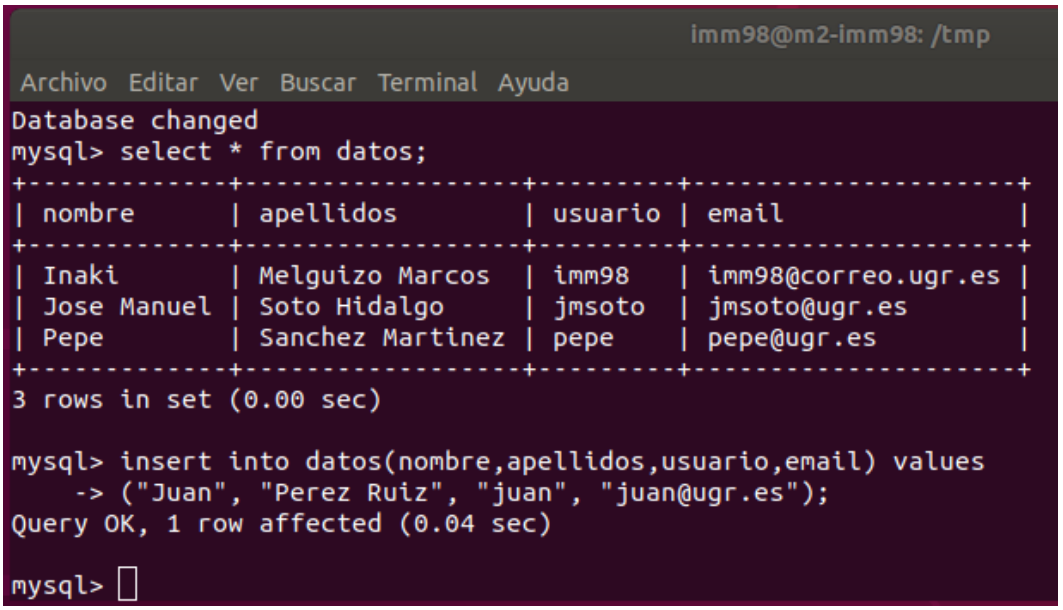
Vemos el estado del esclavo con la sentencia *SHOW SLAVE STATUS\G*

```
                                                    imm98@m1-imm98: /tmp
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
mysql> SHOW SLAVE STATUS\G;
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.56.104
                  Master_User: esclavo2_imm98
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000002
          Read_Master_Log_Pos: 1034
               Relay_Log_File: m1-imm98-relay-bin.000002
                Relay_Log_Pos: 320
        Relay_Master_Log_File: mysql-bin.000002
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB:
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 1034
              Relay_Log_Space: 530
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
```

Y vemos que, como habíamos explicado anteriormente, al ser el campo *Seconds_Behind_Master* distinto de NULL significa que el proceso de creación del esclavo se ha realizado satisfactoriamente. Ahora vamos a comprobar su funcionamiento práctico

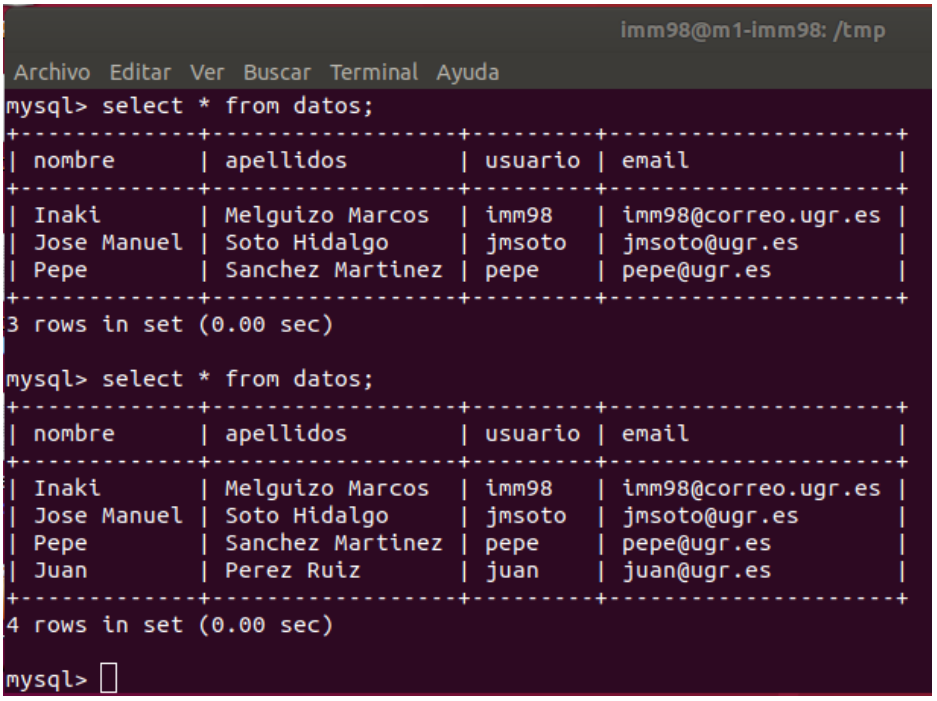
- **m2-imm98**

Tenemos en la tabla datos 3 tuplas y vamos a añadir una más con la sentencia *insert*:



- **m1-imm98**

Realizamos una sentencia *select* para obtener las tuplas de la tabla *datos* antes y después de ejecutar la sentencia *insert* en la máquina m2-imm98. Podemos ver que se ha actualizado la tabla *datos* por el buen funcionamiento de la configuración Maestro-Maestro:



6. Configuración IPTABLES

La última tarea que vamos a realizar en esta práctica es configurar el cortafuegos para que tanto m1-imm98 como m2-imm98 permitan el tráfico al puerto 3306, que es el puerto de MySQL. Para realizar esta labor, las reglas que vamos a tener que añadir al cortafuegos son las siguientes:

iptables -I INPUT -p tcp --dport 3306 -j ACCEPT
iptables -I OUTPUT -p tcp --dport 3306 -j ACCEPT

```
# (1) Eliminar toddas las reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

#(2) Denegar todo el trafico entrante
iptables -P INPUT DROP
iptables -P OUTPUT ACCEPT
iptables -P FORWARD DROP


# (3) Permitir solo peticiones de m3
iptables -I INPUT -s 192.168.56.107 -j ACCEPT
iptables -I OUTPUT -s 192.168.56.107 -j ACCEPT

# (4) Permitir tráfico al puerto 3306
iptables -I INPUT -p tcp --dport 3306 -j ACCEPT
iptables -I OUTPUT -p tcp --dport 3306 -j ACCEPT

iptables -L -n -v
```

Al ejecutar el script anterior vemos que se nos crean las reglas que queremos, es decir, que m1-imm98 acepte sólo el tráfico de m3-imm98 y que permita el tráfico al puerto 3306.

```
imm98@m1-imm98:~$ sudo ./cort_m3.sh
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:3306
    0     0 ACCEPT     all  --  *      *       192.168.56.107       0.0.0.0/0

Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:3306
    0     0 ACCEPT     all  --  *      *       192.168.56.107       0.0.0.0/0
```

Ejecutamos también el script en m2-imm98:

```
imm98@m2-imm98:~$ sudo ./cort_m3.sh
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:3306
    0     0 ACCEPT     all  --  *      *       192.168.56.107       0.0.0.0/0

Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:3306
    0     0 ACCEPT     all  --  *      *       192.168.56.107       0.0.0.0/0
```

Por tanto, ya hemos configurado el cortafuegos mediante IPTABLES en las máquinas m1-imm98 y m2-imm98.