SERVIDORES WEB DE ALTAS PRESTACIONES

Práctica 5: Replicación de bases de datos MySQL

ugr | Universidad de Granada

Autor: Sergio Aguilera Ramírez

Curso 2019 - 2020

Índice

1. Creación de una base de datos (datos y registros)

En esta práctica, vamos a llevar a un nivel más avanzado la copia de 'archivos' entre máquinas a través del sistema de gestión de bases de datos MySQL. En primer lugar, vamos a crear una base de datos en la 'm1' junto con una tabla y varios registros. Para ello, entramos en la interfaz de comandos de MySQL mediante el comando '*sudo mysql −u root −p*' (siempre entraremos como usuario root). La base de datos que vamos a crear, como se indica en el guión, se llamará 'estudiante'. El comando necesario para crear dicha base de datos es '*create database estudiante*;', comprobamos su creación con '*show databases*;' (*Figura* 1).

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| estudiante         |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)
```

Figura 1: Base de datos

A su vez, dentro de esta base de datos, se ha creado una tabla , llamada *datos*, que contiene la información correspondiente a los estudiantes. Primero, accedemos a la base de datos 'estudiante' con el comando '*use estudiante*;', una vez seleccionada, creamos la tabla con el comando '*create table datos*(*nombre varchar*(100), *apellidos varchar*(100), *usuario varchar*(100), *email varchar*(100));', podemos comprobar que se ha creado esta tabla con el comando '*show tables*' (*Figura* 2). Asimismo, podemos ver la descripción de los atributos de esta tabla con el comando '*describe datos*;' (*Figura* 3).

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

Figura 2: Tabla

```
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)
```

Figura 3: Descripción atributos

Por último, en lo referente a la creación de la base de datos, he insertado varios registros a la tabla. Para ello, se utiliza el comando '*insert into datos*(*nombre*, *apellidos*, *usuario*, *email*) *values* ("*nombre..*","*apellido..*","*usuario..*","*email..*");' (*Figura* 4). Existen distintos tipos de atributos, algunos de ellos se pueden ver en la siguiente URL, Tipos de Atributos.

```
mysql> select * from datos;
+---------+-----------------+-----------+-----------------------------+
| nombre  | apellidos       | usuario   | email                       |
+---------+-----------------+-----------+-----------------------------+
| Sergio  | Aguilera Ramirez | aguilera4 | sergioaguilera@correo.ugr.es |
| Javier  | Hidalgo Hidalgo  | jhidalgo  | javierhidalgo@correo.ugr.es  |
| Geraldo | Benitez Benitez  | gBenitez  | gbenitez@correo.ugr.es       |
+---------+-----------------+-----------+-----------------------------+
3 rows in set (0.00 sec)
```

Figura 4: Registro tabla *datos*

2. Realizar copia de la base de datos con mysqldump y copiar el archivo de seguridad en m2

En esta parte de la práctica, vamos a realizar un copia de seguridad de la base de datos 'estudiante' en 'm1' y posteriormente vamos a copiarla en 'm2', creando la base de datos en m2 a partir de esta copia de seguridad.

En primer lugar, accedemos a la interfaz de comando de MySQL en 'm1'. Antes de realizar la copia de seguridad, debemos denegar el acceso a las tablas de la base de datos (para que nadie realice modificaciones durante la copia de seguridad) mediante el comando '$FLUSH\ TABLES\ WITH\ READ\ LOCK$;'. Después de esto, fuera de la interfaz, hacemos la copia de seguridad de la base de datos con el comando *sudo musqldump estudiante* $-u\ root\ -p\ >$ */tmp/estudiante.sql*', esto creará un archivo .sql en el directorio /tmp (*Figura* 5), que contendrá las sentencias necesarias para restaurar los datos de la base de datos. Seguidamente, desbloqueamos las tablas de la base de datos($UNLOCK\ TABLES$;).

```
aguilera4@m1:~$ sudo ls /tmp/
estudiante.sql
systemd-private-6e18b2eef8ad4d50a4a673ea74e0d0aa-apache2.service-LknkYE
systemd-private-6e18b2eef8ad4d50a4a673ea74e0d0aa-systemd-resolved.service-1EdBDq
systemd-private-6e18b2eef8ad4d50a4a673ea74e0d0aa-systemd-timesyncd.service-Ff0Xmr
```

Figura 5: Copia seguridad en m1

Una vez creada la copia de seguridad, la transferimos a 'm2' a través del comando *scp*, guardando la copia en el directorio /tmp (*Figura* 6).

```
aguilera4@m2:~$ sudo ls /tmp/
estudiante.sql
systemd-private-fce9e52a322c408bbc1c47c357caf8c1-apache2.service-RUjcTL
systemd-private-fce9e52a322c408bbc1c47c357caf8c1-systemd-resolved.service-TWNwZx
systemd-private-fce9e52a322c408bbc1c47c357caf8c1-systemd-timesyncd.service-3mEXlz
```

Figura 6: *scp* copia seguridad

3. Restaurar copia de seguridad en m2

Para generar la copia completa de la base de datos en la 'm2', accedemos a la interfaz mysql y creamos una nueva base de datos con el mismo nombre con el que se denomina la base de datos a restaurar. Una vez hecho esto, salimos de la interfaz mysql y a través del comando *mysql* restauramos la copia de la base de datos, introduciendo sus datos en la base de datos que hemos creado en m2, donde el comando completo sería '$sudo\ mysql\ -u\ root\ -p\ estudiante\ <\ /tmp/estudiante.sql$'. Tras realizar todos los pasos comentados anteriormente ya tendríamos nuestra base de datos restaurada (*Figura* 7).

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| estudiante         |
| ligaFutbol         |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
6 rows in set (0.00 sec)

mysql> use estudiante
Database changed
mysql> select * from datos;
+---------+-----------------+-----------+-----------------------------+
| nombre  | apellidos       | usuario   | email                       |
+---------+-----------------+-----------+-----------------------------+
| Sergio  | Aguilera Ramirez | aguilera4 | sergioaguilera@correo.ugr.es |
| Javier  | Hidalgo Hidalgo  | jhidalgo  | javierhidalgo@correo.ugr.es  |
| Geraldo | Benitez Benitez  | gBenitez  | gbenitez@correo.ugr.es       |
+---------+-----------------+-----------+-----------------------------+
3 rows in set (0.00 sec)
```

Figura 7: Restauración copia seguridad

4. Configuración maestro-esclavo de los servidores

En esta sección, vamos a configurar ambas máquinas para que realicen el proceso anterior de forma automática, estableciendo una configuración maestro-esclavo. En primer lugar, configuramos el archivo $mysqld.cnf$ de la máquina que actuará como maestro (situado en la ruta $/etc/mysql/mysql.conf.d/$). En este archivo, comentamos la línea $bind-address$ 127,0,0,1 y indicamos donde almacenaremos los errores estableciendo $log_error = /var/log/mysql/error.log$ (la línea estaba por defecto). Además, estableceremos el id del servidor como 1 y cambiamos la línea a $log_bin = /var/log/mysql/bin.log$. Todo estos cambios podemos verlos en las figuras 8 y 9.

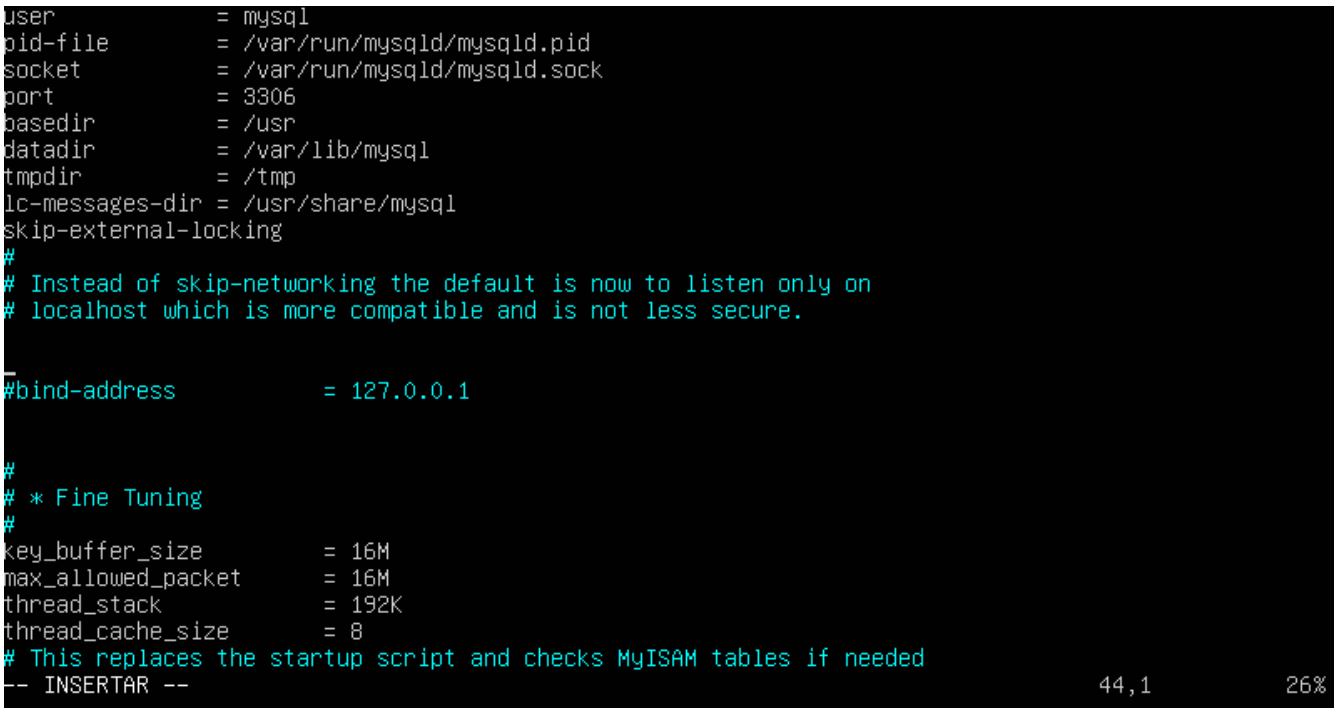

Figura 8: Configuración mysqld.cnf en maestro (1)

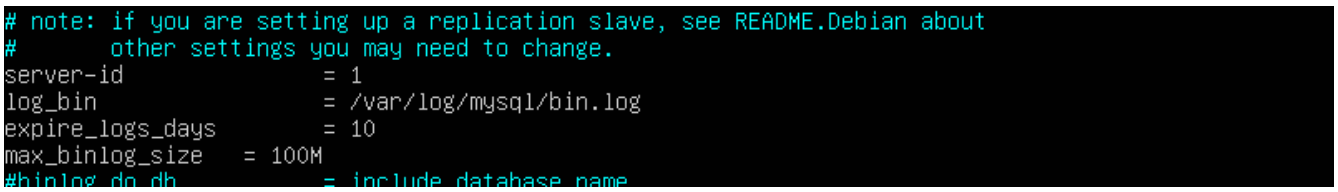

Figura 9: Configuración mysqld.cnf en maestro (2)

De igual forma se hace para el servidor que actúa como esclavo (m2), con la única diferencia de que el id de este servidor será 2.

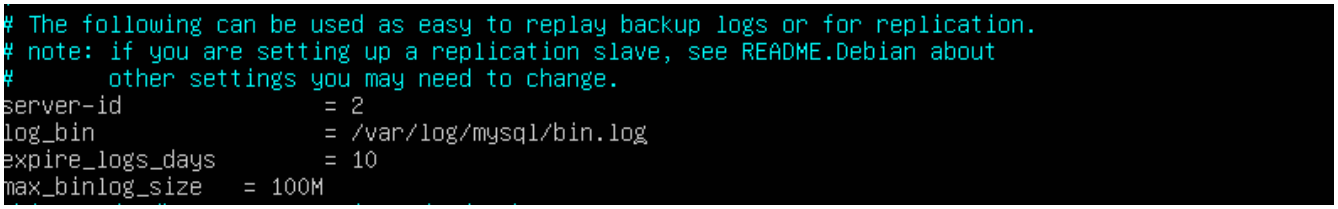

Figura 10: Configuración mysqld.cnf en esclavo

Una vez configurado estos archivos, procedemos a crear el usuario esclavo en el servidor maestro. Para ello, entramos en la interfaz de mySQL y ejecutamos los siguientes comandos de forma secuencial:

1. **CREATE USER esclavo IDENTIFIED BY 'esclavo';** (crearemos el usuario con la contraseña esclavo).
2. **GRANT REPLICATION SLAVE ON *.* TO 'esclavo'@'%' IDENTIFIED BY 'esclavo';** (replicamos todo del maestro a ese esclavo)
3. **FLUSH PRIVILEGES;**

4. **FLUSH TABLES;**

5. **FLUSH TABLES WITH READ LOCK;**

A través del comando *SHOW MASTER STATUS* obtenemos los datos de la BD que vamos a replicar (*Figura* 11).

```
mysql> show master status;
+------------+----------+--------------+------------------+-------------------+
| File       | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set |
+------------+----------+--------------+------------------+-------------------+
| bin.000005 |     5819 |              |                  |                   |
+------------+----------+--------------+------------------+-------------------+
1 row in set (0.00 sec)
```

Figura 11: Datos de la BD a replicar

Ahora los siguientes pasos se realizan sobre el servidor esclavo (m2). Una vez dentro de mySQL, le establecemos los datos del maestro. Para ello, ejecutamos el comando:
$CHANGE\ MASTER TO\ MASTER\_HOST =' 192{,}168{,}56{,}101', MASTER\_USER =' esclavo',$
$MASTER\_PASSWORD =' esclavo', MASTER\_LOG\_FILE =' bin{,}000005',$
$MASTER\_LOG\_POS = 5819, MASTER\_PORT = 3306;$

Tras finalizar el comando anterior, arrancamos el esclavo con $STARTSLAVE$; y activamos las tablas del maestro para permitir añadir nuevos datos ($UNLOCK\ TABLES$;). Por último, comprobamos que todo ha ido correctamente con el comando SHOW SLAVE STATUS\G (en el esclavo, *Figura* 12), donde si Seconds_Behind_Master es igual a 0 indica que todo ha ido bien.

```
           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
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 1
                  Master_UUID: 929b293f-76d3-11ea-8517-0800276aacf8
             Master_Info_File: /var/lib/mysql/master.info
                    SQL_Delay: 0
          SQL_Remaining_Delay: NULL
      Slave_SQL_Running_State: Slave has read all relay log; waiting for more updates
           Master_Retry_Count: 86400
                  Master_Bind:
      Last_IO_Error_Timestamp:
     Last_SQL_Error_Timestamp:
               Master_SSL_Crl:
           Master_SSL_Crlpath:
           Retrieved_Gtid_Set:
            Executed_Gtid_Set:
                Auto_Position: 0
         Replicate_Rewrite_DB:
                 Channel_Name:
           Master_TLS_Version:
1 row in set (0.00 sec)
```

Figura 12: SHOW SLAVE STATUS\G

Por último, podemos poner a prueba todo este proceso borrando e insertando diferentes registros en el maestro y observando si el esclavo realiza las mismas modificaciones(*Figura* 13), para entender bien esta imagen explicar que el primer select es antes de realizar ninguna modificación, el segundo select es el resultado en el esclavo tras insertar el registro ('Rodri') desde el maestro y el último select muestra el resultado en el esclavo tras eliminar el registro de Rodri desde el maestro..

```
mysql> select * from equipo1;
+--------+------------------+----------------+-------------+
| nombre | apellidos        | numeroCamiseta | posicion    |
+--------+------------------+----------------+-------------+
| Sergio | Aguilera Ramirez |             10 | MedioCentro |
| Javier | Muñoz Muñoz      |              7 | Portero     |
| Manuel | Lopez Lopez      |              2 | Defensa     |
+--------+------------------+----------------+-------------+
3 rows in set (0.00 sec)

mysql> select * from equipo1;
+--------+------------------+----------------+----------------+
| nombre | apellidos        | numeroCamiseta | posicion       |
+--------+------------------+----------------+----------------+
| Sergio | Aguilera Ramirez |             10 | MedioCentro    |
| Javier | Muñoz Muñoz      |              7 | Portero        |
| Manuel | Lopez Lopez      |              2 | Defensa        |
| Rodri  | Pelaez Pelaez    |              1 | ExtremoDerecho |
+--------+------------------+----------------+----------------+
4 rows in set (0.00 sec)

mysql> select * from equipo1;
+--------+------------------+----------------+-------------+
| nombre | apellidos        | numeroCamiseta | posicion    |
+--------+------------------+----------------+-------------+
| Sergio | Aguilera Ramirez |             10 | MedioCentro |
| Javier | Muñoz Muñoz      |              7 | Portero     |
| Manuel | Lopez Lopez      |              2 | Defensa     |
+--------+------------------+----------------+-------------+
3 rows in set (0.00 sec)
```

Figura 13: Comprobar replicación

5. Tarea opcional: Configuración Maestro maestro

En esta última sección, vamos a realizar una configuración maestro-maestro entre ambas máquinas. En los apartados anteriores hemos configurado las máquinas como maestro-esclavo (m1:maestro, m2:esclavo), donde las modificaciones realizadas en la m1 se realizaban a su vez en m2, pero de forma contraria no ocurría lo mismo, con esto me refiero a que si modificábamos la base de datos en m2 esta no se reflejaba en m1. El propósito de este apartado es que las modificaciones de ambas máquinas sobre la base de datos se proyectan en la otra máquina. Para ello, vamos a realizar el mismo procedimiento que el seguido en el apartado anterior pero de manera inversa. Creamos un nuevo usuario en m2 mediante los comandos:

1. **CREATE USER esclavo1 IDENTIFIED BY 'esclavo1';**
2. **GRANT REPLICATION SLAVE ON *.* TO 'esclavo1'@'%' IDENTIFIED BY 'esclavo1';**
3. **FLUSH PRIVILEGES;**
4. **FLUSH TABLES;**
5. **FLUSH TABLES WITH READ LOCK;**

De igual forma que en apartado anterior, obtenemos los datos de la base de datos para utilizarlos en la configuración del esclavo en m1.

```
mysql> SHOW MASTER STATUS;
+------------+----------+--------------+------------------+-------------------+
| File       | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set |
+------------+----------+--------------+------------------+-------------------+
| bin.000007 |      982 |              |                  |                   |
+------------+----------+--------------+------------------+-------------------+
1 row in set (0.00 sec)
```

Figura 14: SHOW MASTER STATUS\G;

Una vez generado el esclavo en m2, procedemos a la configuración de este en m1. Al igual que con m2, ejecutamos el comando $CHANGE\ MASTER\ TO\ MASTER\_HOST ='$ 192,168,56,102$', MASTER_USER =' esclavo1', MASTER_PASSWORD =' esclavo1',$ $MASTER\_LOG\_FILE =' bin$,000007$', MASTER_LOG_POS = 982, MASTER_PORT =$ 3306;, pero ahora con la configuración del maestro m2. Tras ejecutar este comando, arrancamos el esclavo con $START\ SLAVE$. El resultado de esto es una configuración maestro-maestro.

La manera más sencilla que se me ha ocurrido para intentar demostrar esto es mediante la *Figura* 15, donde en primer lugar, en ambas máquinas se muestra la tabla en el estado actual, podemos ver que son la misma. Posteriormente insertamos en m1 un nuevo registro a la tabla de datos y mostramos los registros en la misma, de igual forma comprobamos mediante el segundo select de m2 que este registro también se a actualizado en esta máquina. Para comprobar la actuación recíproca, se elimina dicho registro desde m2 y mostramos la tabla actual en esta máquina, mediante el tercer select realizado en m1 comprobamos que esta acción se a producido de forma simultanea en m1. Por lo que tras estas comprobaciones, podemos concluir que la configuración maestro-maestro se ha realizado de forma correcta. (Espero que se haya entendido bien el método de verificación de la configuración maestro-maestro)

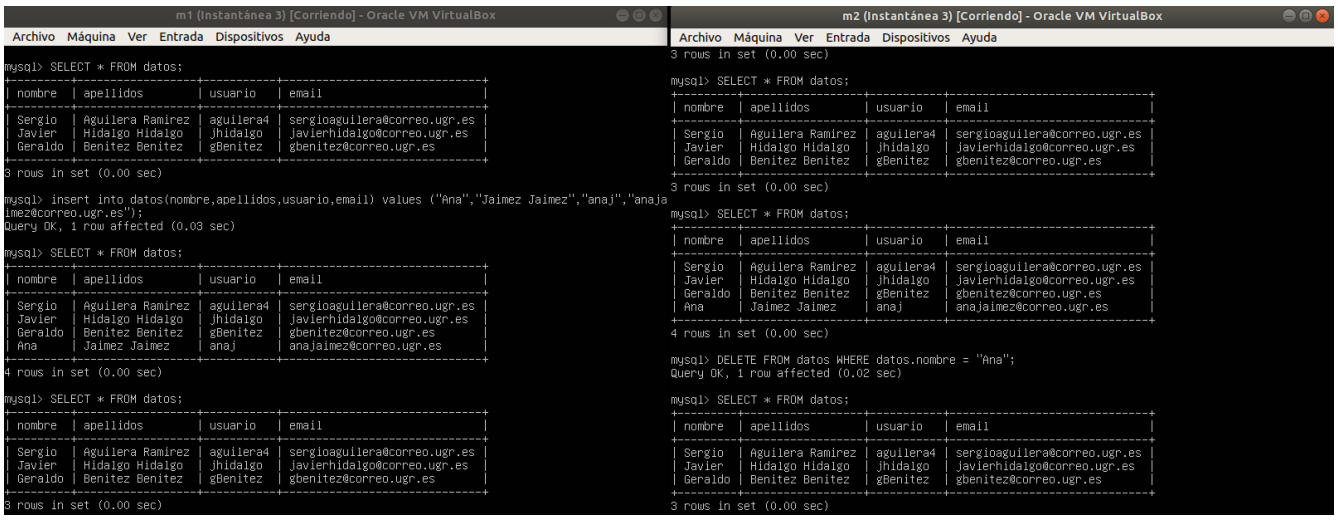

Figura 15: Configuración maestro-maestro

6. Bibliografía

https://support.rackspace.com/how-to/set-up-mysql-master-slave-replication/