# Práctica 5

## Replicación de bases de datos MySQL

***Duración: 2 sesiones***

A la hora de hacer copias de seguridad de nuestras bases de datos (BD) MySQL, una opción muy común suele ser la de usar una réplica maestro-esclavo, de manera que nuestro servidor en producción hace de maestro y otro servidor de backup hace de esclavo.

Tener una réplica en otro servidor también añade fiabilidad ante fallos totales del sistema en producción, los cuales, tarde o temprano, ocurrirán. Por ejemplo, podemos tener un pequeño servidor actuando como backup en nuestra oficina sincronizado mediante réplicas con nuestro sistema en producción.

Esta opción, además, añade fiabilidad ante posibles interrupciones de servicio permanentes del servidor maestro por cualquier escenario catastrófico que nos podamos imaginar. En ese caso, tendremos posiblemente decenas de clientes y servicios parados sin posibilidad de recuperar sus datos si no hemos preparado un buen plan de contingencias. Tener un servidor de backup con MySQL actuando como esclavo de replicación es una solución asequible y no consume demasiado ancho de banda en un sitio web de tráfico normal, además de que no afecta al rendimiento del maestro en el sistema en producción.

## 1. Objetivos de la práctica

Los objetivos concretos de esta práctica son:

- Crear BD e insertar datos por línea de comandos.
- Copiar archivos de copia de seguridad de BD mediante ssh.
- Clonar manualmente BD entre máquinas.
- Configurar una estructura maestro-esclavo entre dos máquinas para realizar el clonado automático de la información.

## 2. Crear un tar con ficheros locales y copiarlos en un equipo remoto

Ya vimos en la práctica 2 cómo crear un tar.gz con un directorio de un equipo y dejarlo en otro mediante ssh. Vimos que deberemos indicar al comando tar que queremos que use stdout como destino y mandar con una pipe la salida al servicio ssh. Éste debe coger la salida del tar y escribirla en un fichero. El comando quedaría:

```
tar czf - directorio | ssh equipodestino 'cat > ~/tar.tgz'
```

De esta forma en el equipo de destino tendremos creado el archivo tar.tgz

También vimos cómo ejecutar de una forma similar el comando scp. Sin embargo, esto que puede ser útil en un momento dado, no nos servirá para sincronizar grandes cantidades de información. En el caso de BD, hay mejores formas de hacerlo.

# 3. Crear una BD e insertar datos

Para el resto de la práctica debemos crearnos una BD en MySQL e insertar algunos datos. Así tendremos datos ejemplo con los cuales hacer las copias de seguridad. En todo momento, **como root**, usaremos la interfaz de línea de comandos del MySQL:

```
sudo mysql -u root -p
Enter password: [TECLEAR CONTRASEÑA SI NO ES VACÍA]
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7
Server version: 5.7.29-DISTRIBUCIÓN UBUNTU …
. . . .
Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.

mysql> create database estudiante;
Query OK, 1 row affected (0,00 sec)

mysql> use estudiante;
Database changed

mysql> show tables;
Empty set (0,00 sec)

mysql>  create table datos(nombre varchar(100), apellidos
varchar(100), usuario varchar(100), email varchar(100));
Query OK, 0 rows affected (0,01 sec)

mysql> show tables;
+---------------------+
| Tables_in_contactos |
+---------------------+
| datos               |
+---------------------+
1 row in set (0,00 sec)

mysql> insert into datos(nombre,apellidos,usuario,email) values
("Jose Manuel", "Soto Hidalgo", "jmsoto", "jmsoto@ugr.es");
Query OK, 1 row affected (0,00 sec)

mysql> select * from datos;
+-------------+--------------+---------+---------------+
| nombre      | apellidos    | usuario | email         |
+-------------+--------------+---------+---------------+
| Jose Manuel | Soto Hidalgo | jmsoto  | jmsoto@ugr.es |
+-------------+--------------+---------+---------------+
1 row in set (0.01 sec)
```

Ya tenemos datos (un registro) insertados en nuestra BD “estudiante” en la tabla llamada “datos”. Para ver los tipos de dato de la tabla “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)
mysql> quit
```

# 4. Replicar una BD MySQL con mysqldump

MySQL ofrece la una herramienta para clonar las BD que tenemos en nuestra maquina. Esta herramienta es ***mysqldump***.

*mysqldump* es parte de los programas de cliente de MySQL, que puede ser utilizado para generar copias de seguridad de BD. Puede utilizarse para volcar una o varias BD para copia de seguridad o para transferir datos a otro servidor SQL (no necesariamente un servidor MySQL). EL volcado contiene comandos SQL para crear la BD, las tablas y rellenarlas.

Esta herramienta soporta una cantidad considerable de opciones. Ejecuta como root el siguiente comando:

```
mysqldump --help
```

para obtener la lista completa. En la siguiente URL se explican con detalle todas las opciones posibles:
http://dev.mysql.com/doc/refman/5.0/es/mysqldump.html

Concretamente, las opciones --quick o --opt hacen que MySQL cargue el resultado entero en memoria antes de volcarlo a fichero, lo que puede ser un problema si se trata de una BD grande. Sin embargo, la opción --opt está activada por defecto.

La sintaxis de uso es:

```
mysqldump ejemplodb -u root -p > /root/ejemplodb.sql
```

Esto puede ser suficiente, pero tenemos que tener en cuenta que los datos pueden estar actualizándose constantemente en el servidor de BD principal. En este caso, antes de hacer la copia de seguridad en el archivo *.sql* debemos evitar que se acceda a la BD para cambiar nada.

Así, en el servidor de BD principal (M1) hacemos:

```
sudo mysql -u root –p

mysql> FLUSH TABLES WITH READ LOCK;
mysql> quit
```

Ahora ya sí podemos hacer el mysqldump para guardar los datos. En el servidor principal (M1) hacemos:

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

Como habíamos bloqueado las tablas, debemos desbloquearlas (quitar el “LOCK”):

```
mysql -u root –p

mysql> UNLOCK TABLES;
mysql> quit
```

Ya podemos ir a la máquina esclavo (M2, configurada como secundaria) para copiar el archivo *.sql* con todos los datos salvados desde la máquina principal (M1):

```
sudo scp /tmp/estudiante.sql usuario@M2:/tmp/estudiante.sql
```

y habremos copiado desde la máquina principal (M1) a la máquina secundaria (M2) los datos que hay almacenados en la BD.

Es importante destacar que el archivo *.sql* de copia de seguridad tiene formato de texto plano, e incluye las sentencias SQL para restaurar los datos contenidos en la BD en otra máquina. Sin embargo, la orden mysqldump no incluye en ese archivo la sentencia para crear la BD (es necesario que nosotros la creemos en la máquina secundaria en un primer paso, antes de restaurar las tablas de esa BD y los datos contenidos en éstas).

Con el archivo de copia de seguridad en el esclavo ya podemos importar la BD completa en el MySQL. Para ello, en un primer paso creamos la BD:

```
sudo mysql -u root -p

mysql> create database estudiante;

mysql> quit
```

Y en un segundo paso restauramos los datos contenidos en la BD (se crearán las tablas en el proceso):

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

Por supuesto, también podemos hacer la orden directamente usando un "pipe" a un ssh para exportar los datos al mismo tiempo (siempre y cuando en la máquina secundaria M2 ya hubiéramos creado la BD):

```
mysqldump estudiante -u root -p | ssh M2 mysql
```

# 5. Replicación de BD mediante una configuración maestro-esclavo

La opción anterior funciona perfectamente, pero es algo que realiza un operador a mano. Sin embargo, MySQL tiene la opción de configurar el demonio para hacer replicación de las BD sobre un esclavo a partir de los datos que almacena el maestro.

Se trata de un proceso automático que resulta muy adecuado en un entorno de producción real. Implica realizar algunas configuraciones, tanto en el servidor principal como en el secundario.

A continuación, se detalla el proceso a realizar en ambas máquinas, para lo cual, supondremos que partimos teniendo clonadas las bases de datos en ambas máquinas. Lo primero que debemos hacer es la configuración de mysql del maestro.

En **M1 (**configuración de mysql del maestro)**:**

Para ello, editamos, como root, el `/etc/mysql/my.cnf` (aunque según la versión de mysql puede que la configuración esté en el archivo `/etc/mysql/mysql.conf.d/mysqld.cnf`) para realizar las modificaciones que se describen a continuación.

Comentamos el parámetro bind-address que sirve para que escuche a un servidor:

```
#bind-address 127.0.0.1
```

Le indicamos el archivo donde almacenar el log de errores. De esta forma, si por ejemplo al reiniciar el servicio cometemos algún error en el archivo de configuración, en el archivo de log nos mostrará con detalle lo sucedido:

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

Establecemos el identificador del servidor.

```
server-id = 1
```

El registro binario contiene toda la información que está disponible en el registro de actualizaciones, en un formato más eficiente y de una manera que es segura para las transacciones:

```
log_bin = /var/log/mysql/bin.log
```

Guardamos el documento y reiniciamos el servicio:

```
/etc/init.d/mysql restart o sudo service mysql restart
```

Si no nos ha dado ningún error la configuración del maestro, podemos pasar a hacer la configuración del mysql del esclavo, en este caso M2 (editar como root su archivo de configuración).

**En M2 (**configuración de mysql del esclavo)**:**
La configuración es **similar** a la del maestro, con la diferencia de que el server-id en esta ocasión será 2. Es decir, haremos las mismas configuraciones que en M1 pero poniendo el server-id a 2:

```
server-id = 2
```

Reiniciamos el servicio en el esclavo:

```
/etc/init.d/mysql restart o sudo service mysql restart
```

y una vez más, si no da ningún error, habremos tenido éxito. Podemos volver al maestro para crear un usuario y darle permisos de acceso para la replicación.

En **M1 (**configuración de mysql del maestro)**:**

Entramos en mysql y ejecutamos las siguientes sentencias para crear un usuario "esclavo" para realizar la replicación, con los permisos necesarios:

```
sudo mysql -u root -p

mysql> CREATE USER esclavo_usuarioUGR IDENTIFIED BY 'esclavo_usuarioUGR';

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

mysql> FLUSH PRIVILEGES;

mysql> FLUSH TABLES;

mysql> FLUSH TABLES WITH READ LOCK;
```

Para finalizar con la configuración en el maestro, comprobamos la configuración y obtenemos los datos de la BD que vamos a replicar para posteriormente usarlos en la configuración del esclavo:

```
mysql> SHOW MASTER STATUS;
```

```
+------------------+----------+--------------+------------------+-------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set |
+------------------+----------+--------------+------------------+-------------------+
| mysql-bin.000001 |      980 |              |                  |                   |
+------------------+----------+--------------+------------------+-------------------+
```

En **M2 (**configuración de mysql del esclavo)**:**

Volvemos a la máquina esclava, entramos en mysql y le damos los datos del maestro. Ejecutamos la siguiente sentencia (ojo con la IP del maestro, con el valor de "master_log_file" y del "master_log_pos". Datos que obtuvimos del maestro con el comando show master status):

```
sudo mysql -u root -p
```

```
mysql> CHANGE MASTER TO MASTER_HOST='DIRECCION_IP_M1',
MASTER_USER='esclavo_usuarioUGR',
MASTER_PASSWORD='esclavo_usuarioUGR', MASTER_LOG_FILE='mysql-
bin.000001', MASTER_LOG_POS=980, MASTER_PORT=3306;
```

Por último, arrancamos el esclavo y ya está todo listo para que los demonios de MySQL de las dos máquinas repliquen automáticamente los datos que se introduzcan/modifiquen/borren en el servidor maestro:

```
mysql> START SLAVE;
```

Ahora, podemos hacer pruebas en el maestro y deberían replicarse en el esclavo automáticamente.

En **M1 (**configuración de mysql del maestro)**:**

Por último, volvemos al maestro y volvemos a activar las tablas para que puedan meterse nuevos datos en el maestro:

```
mysql> UNLOCK TABLES;
```

En **M2 (**configuración de mysql del esclavo)**:**

Ahora, si queremos asegurarnos de que todo funciona perfectamente y que el esclavo no tiene ningún problema para replicar la información, nos vamos al esclavo y con la siguiente orden:

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

revisamos si el valor de la variable "*Seconds_Behind_Master*" es distinto de "*null*" y que no aparece ningún error. En ese caso, todo estará funcionando perfectamente. Si no es así, es que hay algún error y los valores de las demás variables indicarán cuál es el problema y cómo arreglarlo para que todo funcione bien (*un error común se debe a la falta de conexión entre máquinas porque el puerto 3306 esté bloqueado en una de las máquinas*).

Por ejemplo, al ejecutar el comando `show slave status\g` vemos que el valor de la variable "*Seconds_Behind_Master*" es 0, indica que no hay ningún error y todo funciona.

Para comprobar que todo funciona, debemos ir al maestro e introducir nuevos datos a la base de datos. A continuación vamos al esclavo para revisar si la modificación se ha reflejado en la tabla modificada en el maestro.

## Cuestiones a resolver

En esta práctica el objetivo es configurar las máquinas virtuales para trabajar de forma que se mantenga actualizada la información en una BD entre dos servidores (la máquina secundaria mantendrá siempre actualizada la información que hay en la máquina servidora principal).

En esta práctica **se llevarán a cabo, como tareas básicas**:

1. Crear una BD con al menos una tabla y algunos datos.
2. Realizar la copia de seguridad de la BD completa usando mysqldump en la máquina principal y copiar el archivo de copia de seguridad a la máquina secundaria.
3. Restaurar dicha copia de seguridad en la segunda máquina (clonado manual de la BD), de forma que en ambas máquinas esté esa BD de forma idéntica.
4. Realizar la configuración maestro-esclavo de los servidores MySQL en M1 y M2 para que la replicación de datos se realice automáticamente. M1 (maestro) – M2 (esclavo)
5. Añadir regla IPTABLES para permitir tráfico al puerto 3306

**Como tareas avanzadas**:

1. Además de las tareas básicas
2. Realizar la configuración maestro-maestro entre las dos máquinas de bases de datos.

## Normas de entrega

Se elaborará un documento con el funcionamiento del proceso de clonado automático de la información entre bases de datos MySQL en las máquinas principal y secundaria (configuración maestro-esclavo y/o maestro-maestro, en su caso). En el documento a entregar se describirá en detalle cómo se ha realizado la configuración de ambos servidores (configuraciones y comandos de terminal ejecutados en cada momento) y se **ilustrará con capturas de pantalla**. Los datos a insertar en la base de **datos deben de ser los del estudiante**. Por ejemplo, el estudiante Luis Pérez Pérez, con nombre de usuario “luisp” e email ugr, luisp@ugr.es deberá crear una base de datos donde los datos a insertar sean:

```
+-------------+--------------+---------+---------------+
| nombre      | apellidos    | usuario | email         |
+-------------+--------------+---------+---------------+
| Luis.       | Perez Perez  | luisp   | luisp@ugr.es  |
+-------------+--------------+---------+---------------+
```

La práctica se realizará de manera individual.

Se entregará un documento .pdf con el desarrollo de la práctica según el guion detallando, en su caso, los aspectos básicos y avanzados realizados. Se deja a libre elección la estructura del documento el cual reflejará el correcto desarrollo de la práctica a modo de diario/tutorial. En el documento de texto a entregar se describirá cómo se han realizado las diferentes configuraciones (así como comandos de terminal a ejecutar en cada momento).

Para la entrega se habilitará una tarea en PRADO donde se entregará el documento desarrollado siguiendo OBLIGATORIAMENTE el formato **ApellidosNombreP4.pdf**

## Evaluación

La práctica se evaluará mediante el uso de rúbrica específica (accesible por el estudiante en la tarea de entrega) y una defensa final de prácticas.

Tiene un peso del 20% del total de prácticas

La detección de prácticas copiadas implicará el suspenso inmediato de todos los implicados en la copia (tanto del autor del original como de quien las copió).
OBLIGATORIO ACEPTAR LICENCIA EULA DE TURNITIN
Si la memoria supera un 40% de copia Turnitin —> suspenso
del 1-10% -> 0
del 11-20% -> -1
del 20-30% —> -2
del 30-40% —> -3
40% —> suspenso

Las faltas de ortografía se penalizarán con hasta 1 punto de la nota de la práctica.

# Referencias


https://support.rackspace.com/how-to/configuring-mysql-server-on-ubuntu/
https://support.rackspace.com/how-to/set-up-mysql-master-slave-replication/
https://dev.mysql.com/doc/refman/5.7/en/server-configuration-defaults.html
https://dev.mysql.com/doc/refman/5.7/en/server-configuration.html
http://stackoverflow.com/questions/38490785/where-is-mysql-5-7-my-cnf-file
http://blog.programster.org/ubuntu-16-04-default-mysql-5-7-configuration/
https://www.percona.com/doc/percona-xtrabackup/2.3/howtos/setting_up_replication.html#step-3-configure-the-master-s-mysql-server
https://en.wikibooks.org/wiki/MySQL/Replication


# Problemas habituales al realizar la práctica

Entre los problemas más comunes que podemos encontrarnos tras realizar el procedimiento de configuración completo y ver el estado del esclavo están los siguientes:

- Nos da un error sobre el **UUID** de los mysql: Esto se suele dar cuando hemos generado una máquina clonando la otra. Para resolverlo, simplemente tenemos que borrar el archivo `/var/lib/mysql/auto.cnf` y a continuación reiniciar el servicio de mysql.

- Nos da un error sobre la imposibilidad de hacer la **conexión** con el maestro: Esto suele producirse cuando no hay conectividad entre las máquinas. Es importante revisar que todas las políticas por defecto del iptables estén a ACCEPT o se haya añadido regla correspondiente para tráfico tcp al puerto correspondiente. Además, hay que asegurarse de que ambas máquinas “se ven” (se puede hacer ping entre ellas).

- Nos da un error sobre la **IP o el nombre de usuario** en el maestro: Esto se produce cuando nos equivocamos y no le damos los datos correctos en la sentencia SQL de configuración ejecutada en el esclavo.

- Nos da un error indicando que **no encuentra cierta tabla o la propia base de datos**: Esto se suele producir cuando en el maestro y en el esclavo no tenemos exactamente la misma estructura de base de datos. Es importante que tengan los mismos nombres, mismas tablas y mismos campos en éstas (de ahí que se muestre en la sección 4 de este guion cómo hacer la replicación a mano justo antes de intentar hacer la configuración maestro-esclavo).

- Si en un momento dado no estamos seguros de **qué valor tiene alguna variable** definida en el archivo .cnf (p.ej. el identificador de servidor), podemos comprobarla ejecutando:

```
SHOW VARIABLES LIKE 'server_id';
```

- En el momento de **crear el usuario** "esclavo" en el maestro, la sentencia devuelve un "**warning**" (algo falla). En ese caso, hay que revisar muy bien la sintaxis. Dependiendo incluso de la versión de mysql se usan sintaxis diferentes para este paso. Así, intentaremos hacer esta parte de configuración del maestro como sigue (usar la sintaxis adecuada a la versión de mysql que tengamos):

MySQL 5.7

```
GRANT REPLICATION SLAVE ON *.* TO 'usuarioesclavo'@'ipEsclavo'
IDENTIFIED BY 'clave_usuario_esclavo';
```

MySQL 8.0:

```
CREATE USER 'usuarioesclavo'@'ipEsclavo' IDENTIFIED BY
'clave_usuario_esclavo';
GRANT REPLICATION SLAVE ON *.* TO 'usuarioesclavo'@'ipEsclavo' ;
```

Y seguido:

```
FLUSH PRIVILEGES;
FLUSH TABLES;
FLUSH TABLES WITH READ LOCK;
SHOW MASTER STATUS;
```

Otra opción para crear el usuario es la siguiente:

```
CREATE USER 'usuarioesclavo';
SET PASSWORD FOR 'usuarioesclavo' = PASSWORD('laclave');
GRANT REPLICATION SLAVE ON *.* to 'usuarioesclavo';
FLUSH PRIVILEGES;
FLUSH TABLES;
FLUSH TABLES WITH READ LOCK;
SHOW MASTER STATUS;
```