

Taller Replicación MySQL

STR Sistemas - Octubre 2012



Comenzamos

hola *

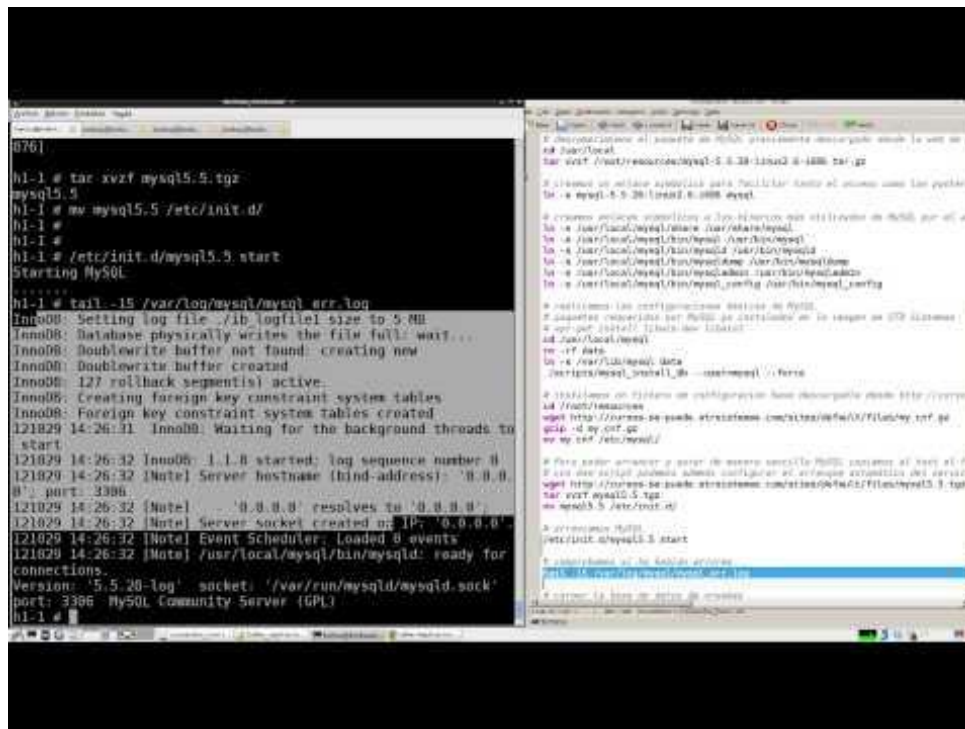
Entorno del taller (I)

Para seguir el taller necesitaremos:

- disponer de dos máquinas virtuales [Virtualbox](#) (host1 y host2) con Debian 6.0 Squeeze creadas a partir de la imagen descargable desde [aquí](#)
- disponer del paquete binario para Linux 32 bits de MySQL (la versión 5.5.28 está incluida en la imagen virtual anterior)
- [esta presentación](#) y [estos ficheros](#)
- ganas y un rato de dedicación

Entorno del taller (II)

Puedes realizar este taller siguiendo este vídeo (abrir el video en otra ventana):



```
h1-1 # tar xvfz mysql5.5.tar.gz
mysql5.5
h1-1 # mv mysql5.5 /etc/init.d/
h1-1 #
h1-1 #
h1-1 # /etc/init.d/mysql5.5 start
Starting MySQL
.....
h1-1 # tail -15 /var/log/mysql/mysql.err.log
InnoDB: Setting log file ./ib_logfile1 size to 5 MB
InnoDB: Database physically writes the file full: wait...
InnoDB: Doublewrite buffer not found: creating new
InnoDB: Doublewrite buffer created
InnoDB: 127 rollback segments active...
InnoDB: Creating foreign key constraint system tables
InnoDB: Foreign key constraint system tables created
121829 14:26:31 InnoDB: Waiting for the background threads to start
121829 14:26:32 InnoDB: 1.1.8 started; log sequence number 0
121829 14:26:32 [Note] Server hostname (bind-address): '0.0.0.0'; port: 3306
121829 14:26:32 [Note] --'0.0.0.0' resolves to '0.0.0.0'
121829 14:26:32 [Note] Server socket created on IP: '0.0.0.0'
121829 14:26:32 [Note] Event Scheduler: Loaded 0 events
121829 14:26:32 [Note] /usr/local/mysql/bin/mysqld: ready for connections.
Version: '5.5.28-log' socket: '/var/run/mysqld/mysqld.sock'
ports: 3306 MySQL Community Server (GPL)
h1-1 #
```

Planificación

Si da tiempo la planificación es:

- Instalar MySQL en host1
- Configurar host1 como master de MySQL
- Instalar MySQL en host2, será un slave
- Replicación master-slave de host1 -> host2
- Convertir host2 en master MySQL
- Replicación master-master host1 <-> host2
- Divagar sobre problemas, mejoras, soluciones

Replicación de bases de datos

Permite tener una copia de la base de datos en un segundo servidor de base de datos lo que sirve para:

- Alta disponibilidad de la base de datos
- Balanceo de base de datos
- Redundancia de los datos

Instalacion de MySQL host1 (I)

Vamos a instalar MySQL desde los paquetes binarios en host1:

- Ficheros a usar del taller:
 - comandos_host1.sh
 - my.cnf
 - mysql5.5
 - bd_pruebas.sql
- seguir las indicaciones:
 - del taller
 - del streaming/video (se actualizará con la URL)

Instalacion de MySQL host1 (II)

Directivas básicas mysqld a tener en cuenta:

- **bind-address**: debe permitir conexión TCP desde el otro servidor
- **port**: el puerto TCP de escucha de MySQL
- **socket**: path al fichero socket para conexiones locales
- **user**: el usuario del sistema que ejecuta MySQL
- **pid-file**: el fichero que almacena el PID del sistema del demonio de MySQL

Instalacion de MySQL host1 (III)

- **basedir:** path de instalación de MySQL
- **datadir:** path de datos de MySQL
- **tmpdir:** path donde se crearán las tablas temporales y ficheros de ejecución MySQL
- **log-error:** el path del log de errores
- **general_log:** activa o desactiva (0/1) el log general
- **general_log_file:** path del log general si éste está activado

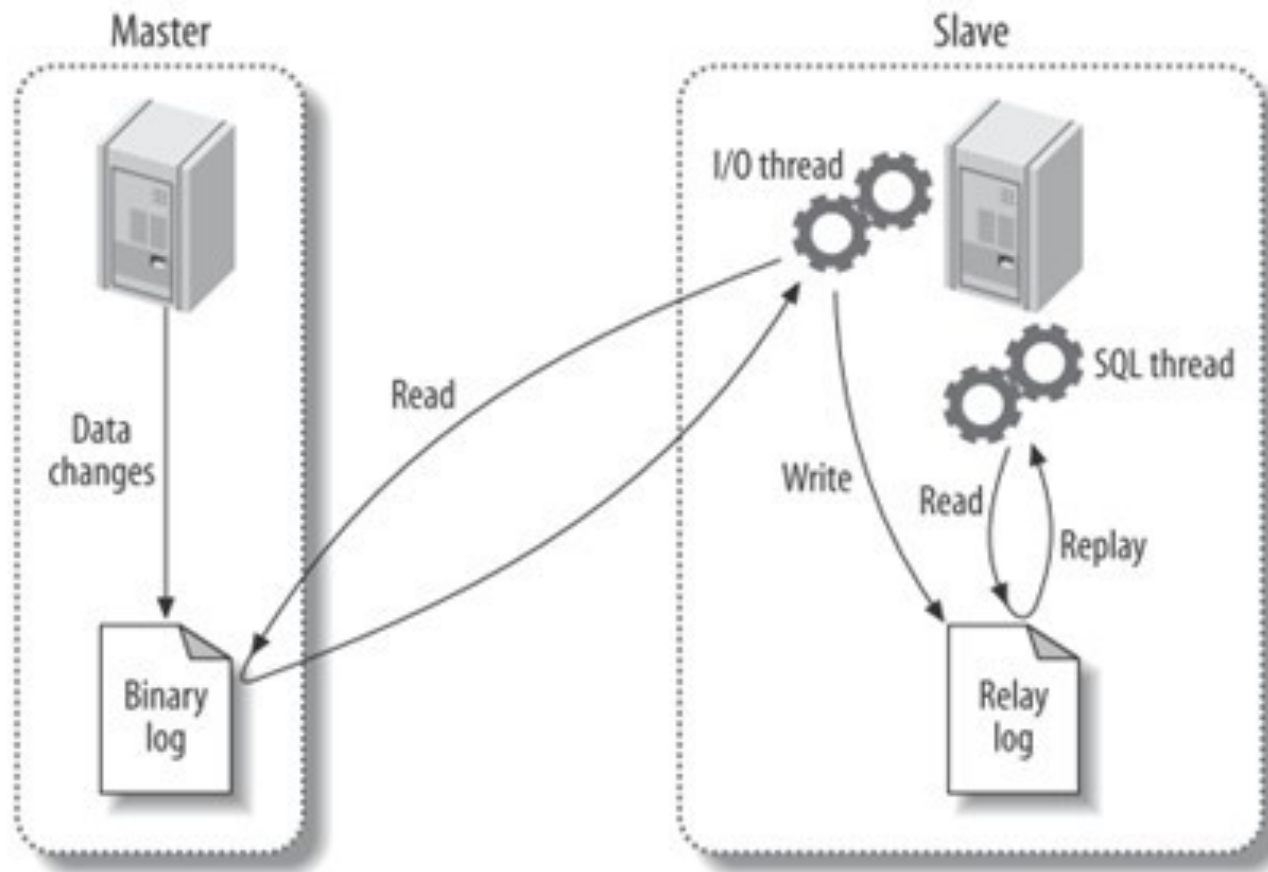
Instalacion de MySQL host1 (IV)

- **slow_query_log**: activa/desactiva (0/1) log de consultas lentas
- **slow_query_log_file**: path del log de consultas lentas
- **long_query_time**: tiempo en segundos para considerar una consulta lenta
- **log-queries-not-using-indexes**: incluir en log de consultas lentas aquellas consultas que aún durando menos de long_query_time segundos no hacen uso de índices.

Instalacion de MySQL host1 (V)

- **max_connections**: número máximo de conexiones TCP que admite MySQL
- **table_cache**: indica el número de descriptores de ficheros de tablas que guarda MySQL en caché

Replicación en MySQL (I)



Replicación en MySQL (II)

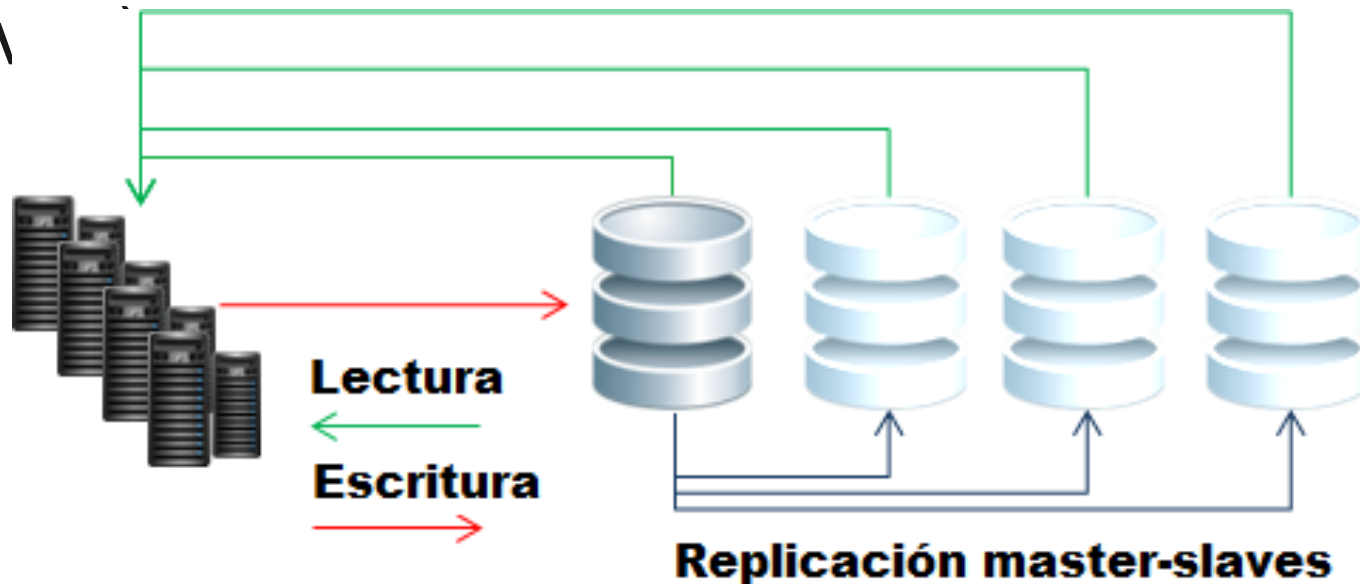
- En MySQL la replicación se basa en servidores maestros (master) y esclavos (slave)
- Los servidores master generan un log binario que contiene aquellas operaciones que producen cambios en la base de datos
- Los servidores slave deben leer los logs binarios del master, guardarlas en el *relay log* y posteriormente ejecutar las operaciones en él indicadas

Replicación en MySQL (III)

- La replicación es asíncrona, es decir, el master escribe en el log binario sin esperar a que el slave lo lea y ejecute
- Un master puede tener varios slaves pero un slave sólo puede leer de un master
- MySQL 5.5 permite replicación semisíncrona, esto quiere decir que el master no valida una transacción hasta que al menos un slave la haya leído del log binario

Replicación en MySQL (IV)

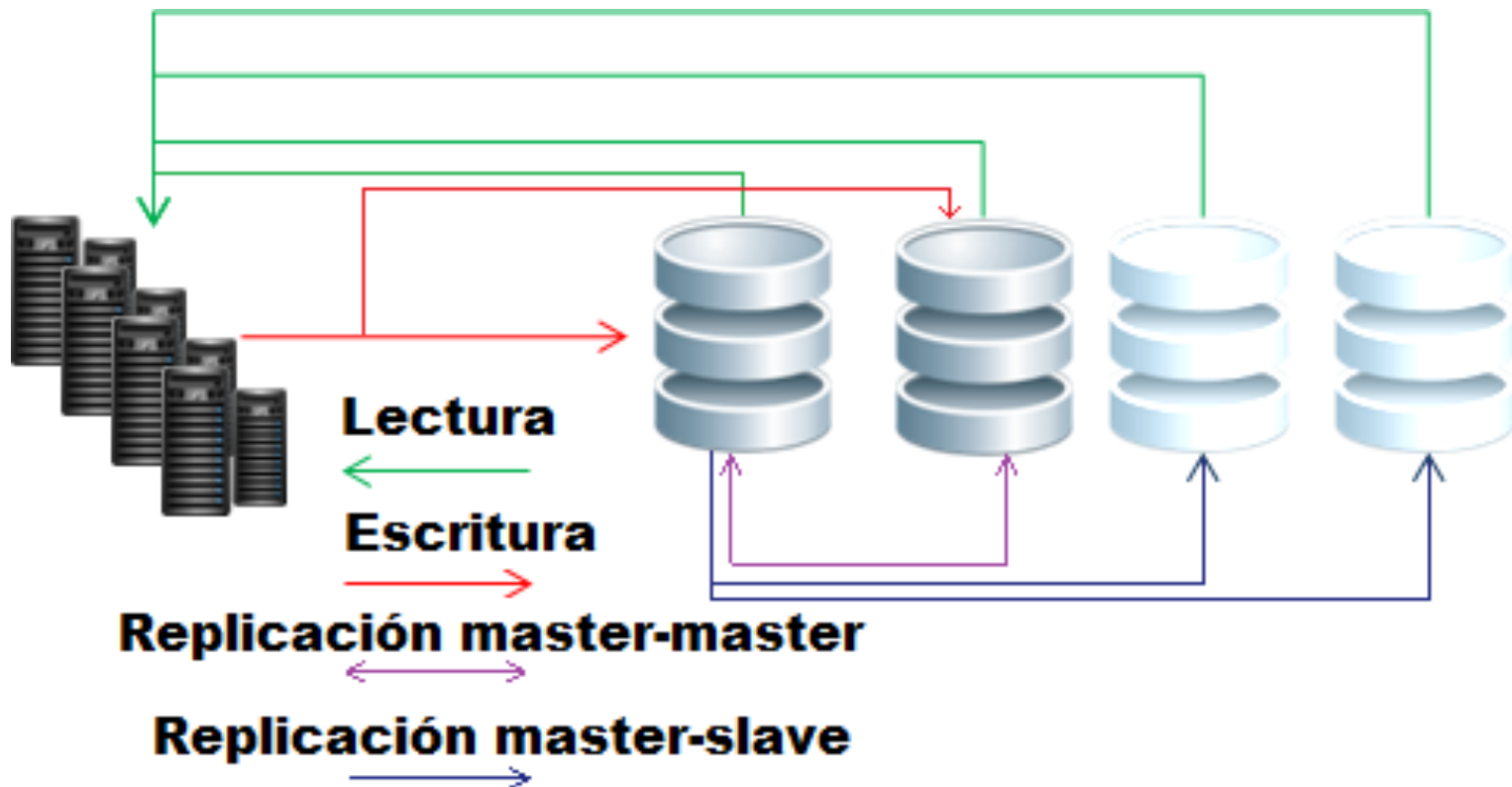
Balanceo lectura en master-slave (varios slaves)



En los servidores slave sólo se pueden hacer operaciones de lectura

Replicación en MySQL (V)

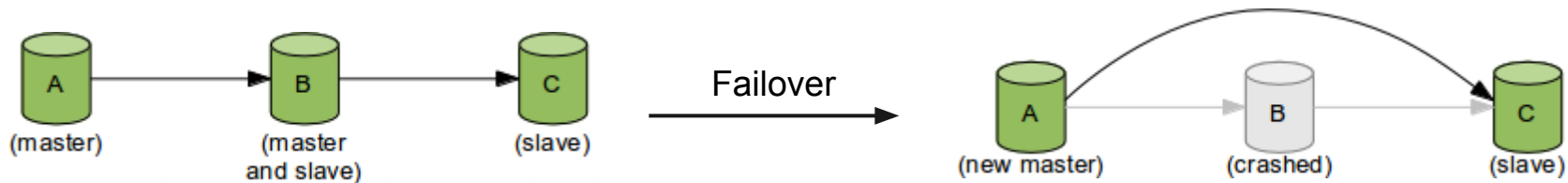
Balanceo lectura-escritura en master-master



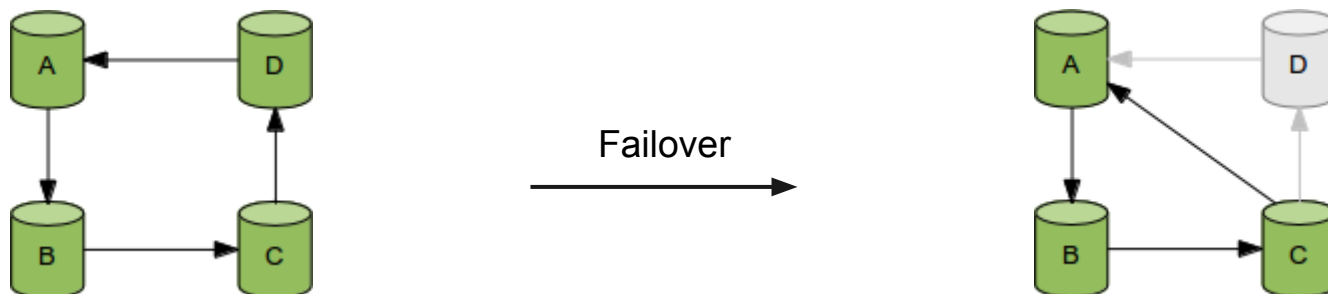
Replicación en MySQL (VI)

Otras arquitecturas avanzadas

Replicación Lineal



Replicación Circular



Configurar host1 como master MySQL (I)

El servidor host1 será un servidor master por lo que debe generar log binario y tener un id único de servidor:

- **server-id**: identificador numérico **que debe ser único** por cada servidor que genere log binario
- **log_bin**: indica el path del log binario, controlar su rotado y tamaño con `expire_logs_days` y `max_binlog_size`

Configurar host1 como master MySQL (II)

A veces queremos replicar todo un servidor (opción por defecto) pero otras sólo queremos replicar alguna base de datos concretas.

- **binlog_do_db**: indica la(s) base(s) de datos a incluir en el log binario. Hay que indicar una directiva por cada BD a incluir
- **binlog_ignore_db**: indica la(s) base(s) de datos a excluir del lg binario- Hay que indicar una directiva por cada BD a excluir

Configurar host1 como master MySQL (III)

Tras reiniciar el servidor MySQL comprobamos que está funcionando de manera correcta como master (fichero comandos_master_slave.txt):

- `ls -lahrt /var/log/mysql`
- `SQL: SHOW MASTER STATUS`

Además debemos crear usuario para la replicación:

- `SQL: GRANT REPLICATION SLAVE ON *.* TO '<user>'@'<host1>' IDENTIFIED BY '<password>';`
- `SQL: GRANT REPLICATION SLAVE ON *.* TO`

Instalacion de MySQL host2 (I)

Instalar MySQL desde los paquetes binarios en host2 de manera similar a como lo hicimos en host1 teniendo en cuenta que debe ser un slave que replica a host1 (comandos_master_slave.txt y comandos_host2.sh):

- Bloquear master y ver su estado
- Obtener un dump de mysql de host1
- Cargar el dump de host1 en host2
- Comenzar la replicación: recordar que deben tener distinto server-id

Instalacion de MySQL host2 (II)

Para configurar un slave debemos partir de una versión de la base de datos estable y cuyo punto de replicación en el master conozcamos:

- master (host1):
 - SQL: FLUSH TABLES WITH READ LOCK; (se bloquea la base de datos para integridad)
 - SQL: SHOW MASTER STATUS; (copiar la salida en un fichero de texto)
 - Dump desde consola: mysqldump (obtener dump de todas las bases de datos)
 - SQL: UNLOCK TABLES;

Instalacion de MySQL host2 (III)

- slave (host2):
 - Cargar dump en host: mysql
 - SQL (datos del SHOW MASTER STATUS):
`CHANGE MASTER TO MASTER_HOST='<host1>',
MASTER_USER='<user>',
MASTER_PASSWORD='<password>',
MASTER_LOG_FILE='<bin_log>',
MASTER_LOG_POS=<bin_position>;`
 - SQL: `START SLAVE;`
 - SQL: `SHOW SLAVE STATUS\G`

Realicemos pruebas (seguir taller/video)

Replicación master-master (I)

Para configurar la replicación master-master (fichero comandos_master_master.txt):

- realmente tendremos dos master-slave, es decir, host1 -> host2 y host2->host1
- ya tenemos el primero: host1 -> host2
- luego se convierte en master-master creando el master-slave host2 -> host1:
 - master (host2 no bloqueamos tablas por que está sin uso de escritura todavía):
 - SQL: `SHOW MASTER STATUS;` (copiar salida)

Replicación master-master (II)

- slave (host1):
 - SQL (datos del SHOW MASTER STATUS):
CHANGE MASTER TO
MASTER_HOST='<host2>',
MASTER_USER='<user>',
MASTER_PASSWORD='<password>',
MASTER_LOG_FILE='<bin_log>',
MASTER_LOG_POS=<bin_position>;
 - SQL: START SLAVE;
 - SQL: SHOW SLAVE STATUS\G

Realicemos pruebas (seguir taller/video)

Replicación master-master (III)

¿Qué pasa si hacemos esto?:

host2: (consola) `/etc/init.d/mysql5.5 stop`

host1: (SQL BD pruebas) `INSERT INTO usuarios (IDUsuario, Nick) VALUES(NULL, 'strsistemas');`

host1: (consola) `/etc/init.d/mysql5.5 stop`

host2: (consola) `/etc/init.d/mysql5.5 start`

host2: (SQL BD pruebas) `INSERT INTO usuarios (IDUsuario, Nick) VALUES(NULL, 'strmailer');`

host1: (consola) `/etc/init.d/mysql5.5 start`

host1/host2: (SQL) `SHOW SLAVE STATUS\G`

Replicación master-master (IV)

SOLUCIÓN: Tenemos que controlar las claves autoincrement

(comandos_master_master_autoincrement.txt):

- **auto_increment_increment**: el incremento entre claves autoincrementemente en cada servidor (igual en todos los servidores, en nuestro caso 2 que equivale al número de servidores)
- **auto_increment_offset**: el offset inicial del servidor en cuestión (1 para host1 y 2 para host2)

Replicación master-master (IV)

Ahora hay que recuperar la replicación de manera estable en el punto que queríamos:

- host2: (SQL BD pruebas) `DELETE FROM usuarios WHERE Nick='strmailer';`
- host1: (SQL BD pruebas) `STOP SLAVE;`
- host1: (SQL BD pruebas) `SET GLOBAL SQL_SLAVE_SKIP_COUNTER=1;`
- host1: (SQL BD pruebas) `START SLAVE;`

Replicación master-master (V)

- host1: (SQL BD pruebas) SHOW SLAVE STATUS\G
- host2: (SQL BD pruebas) STOP SLAVE;
- host2: (SQL BD pruebas) START SLAVE;
- host2: (SQL BD pruebas) SHOW SLAVE STATUS\G
- host2: (SQL BD pruebas) INSERT INTO usuarios (IDUsuario, Nick) VALUES(NULL, 'strmailer');
- host2: (SQL BD pruebas) SHOW SLAVE STATUS\G
- host1: (SQL BD pruebas) SHOW SLAVE STATUS\G

SQL Replicación MySQL (I)

Ver estado slave:

```
SHOW SLAVE STATUS;
```

Ver estado master:

```
SHOW MASTER STATUS;
```

Pausar replicación:

```
STOP SLAVE;
```

Establecer permisos de replicación:

```
GRANT REPLICATION SLAVE ON *.* TO  
'<user>'@'<host_slave>' IDENTIFIED BY '<password>';
```

SQL Replicación MySQL (II)

Establecer replicación:

```
CHANGE MASTER TO MASTER_HOST='<host_master>',  
MASTER_USER='<user>',  
MASTER_PASSWORD='<password>',  
MASTER_LOG_FILE='<bin_log>',  
MASTER_LOG_POS=<bin_position>;
```

Eliminar replicación:

```
RESET SLAVE;
```

Saltar instrucción en la replicación:

```
SET GLOBAL SQL_SLAVE_SKIP_COUNTER=1;
```

Replicación MySQL 5.6

Se acaba de lanzar [la versión 5.6 de MySQL](#) que incluye las siguientes mejoras relativas a replicación:

- Replicación con retardo, muy útil para usar la replicación como backup de datos
- Replicación multithread, lo que permite una replicación más rápida
- Optimización de replicación basada en filas
- Nuevas herramientas de gestión de replicación para consola
- [Post interesante replicación MySQL 5.6](#)

HASTA PRONTO

GRACIAS



www.strsistemas.com



[@STRSistemas](https://twitter.com/STRSistemas)



info@strsistemas.com



902027609

Por favor, tanto si has hecho el taller presencial como si lo has seguido online rellena [esta encuesta](#).