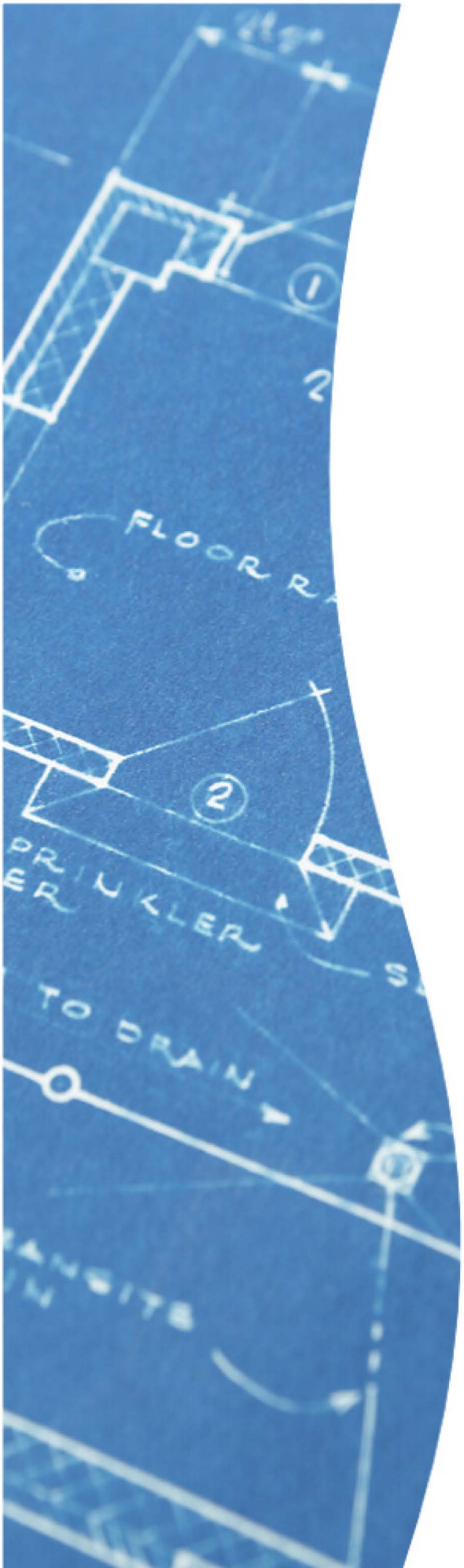


ESCALADO HORIZONTAL y REPLICACIÓN de BASES de DATOS MYSQL para **EMPRESAS de ELEVADO CRECIMIENTO**

Nick Kloski, Engineered Solutions Group

Sun BluePrints™ Online

Part No 820-6824-10
Revision 1.0, 11/4/08

Tabla de Contenidos

Escalado horizontal y Replicación de bases de datos MySQL

Está plenamente reconocido que MySQL es el software de base de datos más popular del mundo. Desde su nacimiento en 1.995, se han realizado cerca de 11 millones de i nstalaciones de este producto en todo el mundo y en una amplia variedad de sectores. Existen actualmente más instalaciones de MySQL en uso real que cualquier otra arquitectura de base de datos. Desde compañías de reciente creación, deseando convertirse en el próximo icono mediático de la Web2.0, hasta empresas globales de gran tamaño, la arquitectura de la base de datos MySQL ha demostrado se extremadamente flexible, extensible, escalable y más que capacitada para adaptarse a cualquier entorno de base de datos de alta capacidad en multitud de instalaciones diferentes.

El núcleo de la arquitectura MySQL está disponible gratuitamente para su descarga y utilización, lo que hace que MySQL sea un software enormemente popular en una amplia variedad de entornos. Sin embargo, la facilidad con la que se despliegan instancias de la base de datos a menudo conllevan también algunos inconvenientes: A medida que pasa el tiempo, algunas empresas observan cómo tienen que hacer frente a una creciente proliferación de instancias MySQL independientes, lo que les obliga a adoptar un planteamiento que les permita escalar su implementación de MySQL de forma inteligente, equilibrando la capacidad de expansión con la necesaria resistencia ante fallos.

Este documento trata las bases de una instalación MySQL para poder escalar de acuerdo con las demandas de los usuarios, pero proporcionando también la flexibilidad y facilidad de uso que ofrece una instalación individual.

Introducción al concepto de escalado horizontal

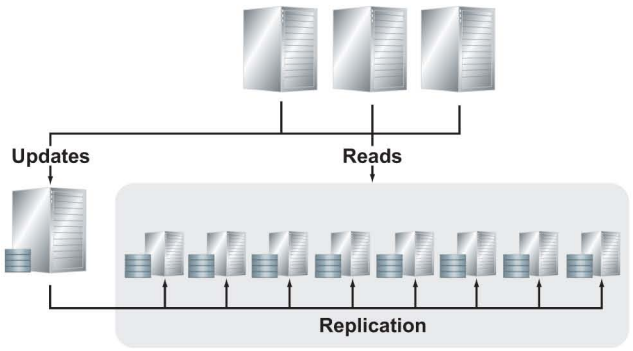
MySQL tiene un coste cero en el caso de ser utilizada como centro de un sistema de base de datos. Cualquier empresa, de cualquier tamaño, cuenta con todo el apoyo para descargar y poner en producción instancias MySQL en tantas configuraciones de tamaño y propósito como deseen implementar. Debido a esto, en muchas ocasiones MySQL es utilizada para pruebas de concepto iniciales y como back-end de los repositorios de datos de diversos proyectos. A medida que dichos proyectos se muestran factibles, la arquitectura MySQL se muestra habitualmente más que capacitada para abordar la demanda inicial generada por la aplicación asociada. En algún momento, sin embargo, las demandas de los usuarios requerirán que la base de datos MySQL escale para proporcionar tiempos de respuesta adecuados, y que posiblemente excederán la capacidad del hardware inicialmente dimensionado. Es precisamente en ese momento cuando la arquitectura MySQL necesita ampliarse más allá del servidor inicial para proporcionar una mayor capacidad que satisfaga la demanda. En ese momento una planificación inteligente para escalar horizontalmente no solamente resolverá estos problemas, manteniendo un tiempo de respuesta razonable para el usuario final, si no que además proporcionará la suficiente resistencia frente a fallos hardware, que de otra manera podrían hacer caer la base de datos.

Definición de escalado horizontal

cuando los administradores de base de datos (DBAs) hablan de escalado horizontal están haciendo referencia a la capacidad de servir un número cada vez mayor de peticiones contra la base de datos. El tiempo de ejecución específico de una petición, una vez que ha llegado a la base de datos, se ve afectado por determinados factores, como la lógica de aplicación y la potencia de la CPU, cantidad de memoria e interconexiones de E/S subyacentes.

Nota – Un tratamiento en mayor profundidad del ajuste y optimización de la base de datos queda fuera del ámbito de este documento. En el mismo solo se tratan las consideraciones a nivel corporativo y la escalabilidad global del servidor.

Con el transcurso del tiempo, a medida que las aplicaciones se hacen más populares, l a capacidad de una instancia MySQL puede no ser suficiente para cubrir las necesidades. En el mundo de las bases de datos MySQL, el término escalado horizontal hace referencia a la mejora en el rendimiento y escalabilidad de la aplicación de una forma incremental, a medida que se genera la propia demanda, añadiendo múltiples servidores de base de datos replicados entre si, empleando hardware de bajo coste y amplia disponibilidad en el mercado



(Figura 1) Escalado horizontal básico para operaciones de lectura de la base de datos.

Este planteamiento es opuesto al otro método de escalado: el escalado vertical. En un escenario de escalado vertical, lo que se amplía es el hardware del servidor existente, permitiéndole servir mayor número de peticiones aumentando la potencia del back-end.

Los dos métodos tienen sus peculiaridades:

- Escalado vertical (también denominado Scale-up)
  - Debe realizarse con hardware SMP, que normalmente es más caro, y que facilita su propio escalado.
  - A veces necesita ejecutar software propietario para permitirle escalar.
  - Suele estar ligado en exclusiva a una plataforma hardware/software.
  - Una vez alcanzado el límite de una plataforma hardware específica, es necesario sustituir el servidor al completo.

- Escalado horizontal (también conocido como Scale-out)
  - Se puede conseguir fácilmente con hardware Intel/AMD estándar.
  - Se ejecuta sobre software/sistemas operativos de código abierto.
  - Puede aprovechar la independencia de plataforma para permitir instalaciones MySQL escaladas que se ejecuten sin problemas sobre una amplia variedad de servidores.

– Añadir servidores estándar permite disfrutar de una sostenible y mejor experiencia de usuario.

El servidor MySQL no fue originalmente diseñado para funcionar sobre servidores de alto coste y elevada complejidad. En lugar de ello, su diseño pretende interconectar fácilmente pequeños servidores estándar, incluyendo tecnologías de replicación que permitan un rápido escalado horizontal de las bases de datos sobre servidores de bajo coste y amplia disponibilidad en el mercado. El resto de este documento se centra en el escalado horizontal y la replicación de MySQL.

Componentes para el escalado horizontal de la base de datos MySQL

Los siguientes componentes pueden ser utilizados para implementar un escalado horizontal de una arquitectura de base de datos MySQL:

•*Replicación MySQL*
La definición de la replicación MySQL es simple: la duplicación de modificaciones o cambios en los datos en más de una ubicación. La replicación puede ser tanto síncrona como asíncrona. Con una replicación síncrona los nuevos datos que se incorporan a una arquitectura MySQL escalada pueden ser aceptados y escritos en todos los servidores de base de datos al mismo tiempo. Por el contrario, en una replicación asíncrona, tanto los cambios como los nuevos datos se graban sobre un servidor maestro, para ser enviados con posterioridad a los servidores secundarios.

•*Cluster MySQL*
El producto de cluster MySQL es una base de datos en memoria, sin compartición alguna, que permite a una empresa incrementar tanto la redundancia como la capacidad, rigiéndose por la premisa de más copias en más ubicaciones. Al añadir nodos, la base de datos subyacente se ejecuta sobre más servidores físicos y puede ser accedida por más clientes. Añadir replicación a un cluster MySQL puede ser adecuado para conseguir redundancia geográfica (con el objetivo de disponer de una solución de backup y recuperación ante desastres), tomando un cluster formado por múltiples servidores y replicándolo en otra ubicación geográficamente distante.

•*Linux Heartbeat*
El sistema operativo Linux dispone de una pequeña utilidad software que juega un importante papel en un cluster MySQL. A pesar de no ser requisito indispensable, añadir el componente heartbeat dentro del cluster MySQL hace que el sistema operativo trabaje mano a mano con la base de datos, proporcionando notificación a otro lado del cluster, cuando se produce una caída del otro extremo, debido a un fallo hardware o a otro tipo de suceso grave. Linux Heartbeat gestiona IP takeovers y alerta al framework MySQL de que se ha producido un fallo y que, por lo tanto, es necesario iniciar el procedimiento de recuperación correspondiente.

•*Balanceo de carga*
Existen varios métodos que un DBA puede emplear para dirigir inteligentemente las entrada de peticiones de usuario hacia una arquitectura MySQL escalada. Mediante el empleo de balanceadores de carga, la entrada de peticiones es dirigida a la base de datos MySQL apropiada. En un escenario MySQL escalado horizontalmente el balanceador de carga puede estar basado en hardware o software.

Un ejemplo de balanceo basado en hardware es un router inteligente que pueda interceptar las peticiones de entrada y sepa a qué servidor enviar dichas peticiones. El balanceo de carga basado en software también se denomina lógica de aplicación. La aplicación destino, en el momento de recibir una petición de lectura/escritura/actualización proveniente de un usuario, puede disponer de un mecanismo de balanceo incorporado que le capacita para redirigir la petición hacia el servidor MySQL adecuado.

•*Distributed Replicated Block Device*
Distributed Replicated Block Device (DRBD), como su nombre sugiere, crea una réplica a nivel bloque entre diferentes servidores físicos. El software DRBD es el componente principal que permite la replicación síncrona.
Existen además diversos dispositivos hardware de almacenamiento compartido y agentes de clustering para monitorizar clusters MySQL replicados, pero éstos no son objeto de este documento, más allá de su mera mención.

Nota – No todas estas tecnologías están incluidas como parte de la descarga de MySQL Community; algunos servicios pueden ser de pago y pueden requerir una suscripción a MySQL Enterprise. Consultar http://mysql.com/products/enterprise/features.html para comprobar las características incluidas.

El resto del documento trata acerca de dichos componentes y describe como éstos pueden ser empleados en arquitecturas MySQL con escalado horizontal.
Usos objetivo para la replicación MySQL

Para resolver los problemas comunes a muchas implementaciones MySQL

Es posible emplear una combinación de estos componentes. El caso más sencillo para utilizar la replicación es simplemente a efectos de backup en caso de presentarse algún fallo en el servidor principal. Utilizando una sencilla configuración maestro/esclavo, los datos que son enviados al servidor activo son también copiados asíncronamente sobre un servidor esclavo y, en caso de que el servidor activo principal alguna vez fallase, el servidor replicado asumiría sus funciones.

Algunos escenarios más complejos para la replicación son por ejemplo los de Business Intelligence, donde la base de datos de back-end puede ser replicada en otro servidor, sobre el que pueden aplicarse diferentes técnicas de investigación y análisis de datos. Las arquitecturas escaladas horizontalmente pueden aportar mayores niveles de rendimiento y crecimiento flexible. También las implementaciones en alta disponibilidad (HA) pueden permitir el acceso constante e ininterrumpido a la base de datos, convirtiendo a la implementación en resistente ante un único fallo hardware.

Replicación de la base de datos MySQL

MySQL 5.1, la última versión del software en el momento de elaborar este documento, proporciona como novedad soporte para réplicas basadas en filas. Las versiones previas de MySQL (5.0 y anteriores) soportaban

La principal diferencia entre una sentencia y una fila, a propósitos de replicación, es que una sentencia se genera mediante SQL (structured query language), mientras que en el caso de las filas, se basa en los datos en bruto almacenados en cada fila de la base de datos.

Antes de la aparición del soporte para la replicación basada en filas presente en MySQL 5.1, había que tener precaución al emplear la réplica basada en sentencias, con el objetivo de optimizar éstas para evitar cuellos de botella que afectasen negativamente al rendimiento. Dichos problemas de rendimiento podrían ser ocasionados por una ineficiente estructura de interrogación que solicitase múltiples datos de pequeño tamaño, haciendo que la query principal sea muy compleja. A veces era inevitable que el rendimiento se resintiese al intentar devolver el resultado de una query. Ahora, en MySQL 5.1, la réplicación puede realizarse no solamente mediante sentencias, sino que se puede hacer mediante filas, o una mezcla de ambos sistemas.

Replicación mediante sentencias versus filas

Comenzando en MySQL 5.1, los DBAs tienen la posibilidad de emplear tanto el sistema de filas como de sentencias, o una combinación de ambas, a la hora de articular una réplica. La réplica basada en sentencias es más adecuada para aplicaciones que no hacen un uso exhaustivo de funciones no determinísticas o llamadas al sistema como usuarios SELECT. La réplica basada en sentencias MySQL produce ficheros de log binarios más pequeños, y éstos pueden utilizarse para auditar la base de datos. Debido a la mayor eficiencia del logging binario, la réplica basada en sentencias puede, en muchos casos, procesar más transacciones por segundo.

Con la réplica basada en filas, todo (i.e., una fila al completo) puede ser replicado. Se emplean menor número de bloqueos para muchas de las sentencias DML, tanto en los servidores maestros como en los esclavos, al utilizar la réplica basada en filas. También la réplica basada en filas puede dar como resultado una mayor velocidad a la hora de aplicar cambios sobre los servidores esclavos, especialmente en objetos con claves principales.

Funcionamiento de la replicación MySQL

La Figura 2 ilustra el proceso de replicación en MySQL, asumiendo una configuración MySQL con un servidor maestro y otro esclavo. El demonio mysqld en el servidor maestro es el proceso que interactúa con las peticiones de entrada. Se ejecutan y sirven las operaciones READ y SELECT, mientras que las UPDATE son escritas en disco, pero también se escriben en lo que se denomina un log binario o binlog. Este binlog es indizado, o indexado, y se convierte en la base para la replicación de dichos UPDATES sobre el servidor esclavo. El binlog del servidor maestro es enviado al demonio mysqld del servidor esclavo, el cual sabrá en ese momento cuándo tiene que ejecutarse la réplica.

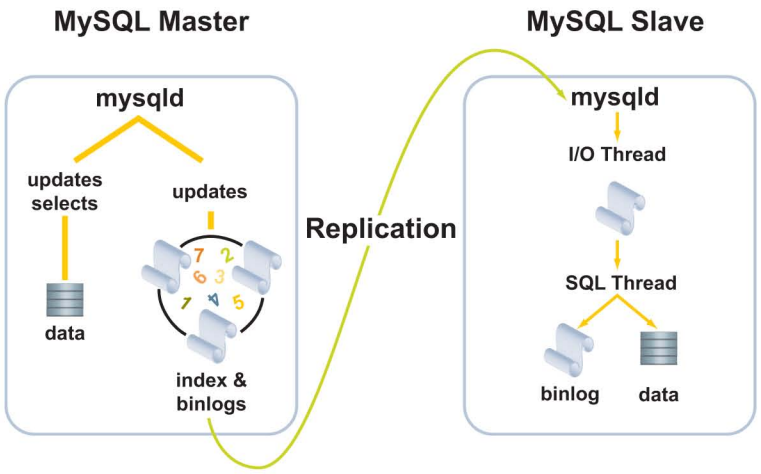


Figura 2. Funcionamiento de la replicación MySQL.

Para gestionar el flujo de entrada de binlogs, MySQL envía el binlog de replicación a un proceso especial, el I/O Thread, el cual es responsable de emular una operación de escritura real sobre el binlog del servidor esclavo. En este último, este nuevo thread de replicación (el binlog que proviene del maestro) es escrito en un log temporal en el servidor esclavo, denominado relay binlog. El relay binlog es entonces tratado como un thread SQL normal hacia la base de datos del servidor esclavo, el cual es responsable de dos cosas; el flujo de replicación, repleto de modificaciones en los datos, es almacenado en disco, pero al mismo tiempo es canalizado sobre el log binario del servidor esclavo.
Este binlog del servidor esclavo, si se desea, forma la base para la replicación en cascada hacia la otra línea de servidores. Una vez que exista un binlog en cualquier servidor MySQL, tanto si éste es maestro o esclavo, dicho binlog puede ser utilizado para una réplica posterior con otros servidores esclavos.

Debido a que los esclavos controlan el proceso de réplica, los esclavos individuales pueden ser conectados y desconectados del servidor sin afectar a las operaciones del maestro. También, ya que cada esclavo recuerda la posición dentro del log binario, es posible desconectarlos, volver a conectarlos y "ponerse al corriente" continuando desde la posición guardada. Advertir, sin embargo, que en escenarios donde se ha producido una recuperación después de un fallo, es recomendable comprobar con cuidado que todos los esclavos vuelven a formar parte del grupo de replicación sin problemas, así como que cualquier cambio realizado por los miembros online del grupo ha sido replicado, no solamente las operaciones de escritura en proceso.

Topologías de replicación MySQL

La replicación de bases de datos MySQL soporta diversas topologías, como puede apreciarse en la Figura 3. La topología más sencilla es la configuración de un maestro/esclavo. También están soportadas configuraciones de un maestro y múltiples esclavos.

Las topologías más complejas incluyen configuraciones multi-maestro, que constan de dos o más servidores maestro. Aunque dichas configuraciones están soportadas, se debe prestar especial cuidado para garantizar que los espacios de datos no son compartidos entre los servidores maestros. Si el maestro no es correctamente configurado en un entorno multi-master, puede producirse la sobre-escritura de algunos datos.
Una topología que no está soportada es la configuración multi-origen, con varios maestros actualizando un único servidor esclavo.

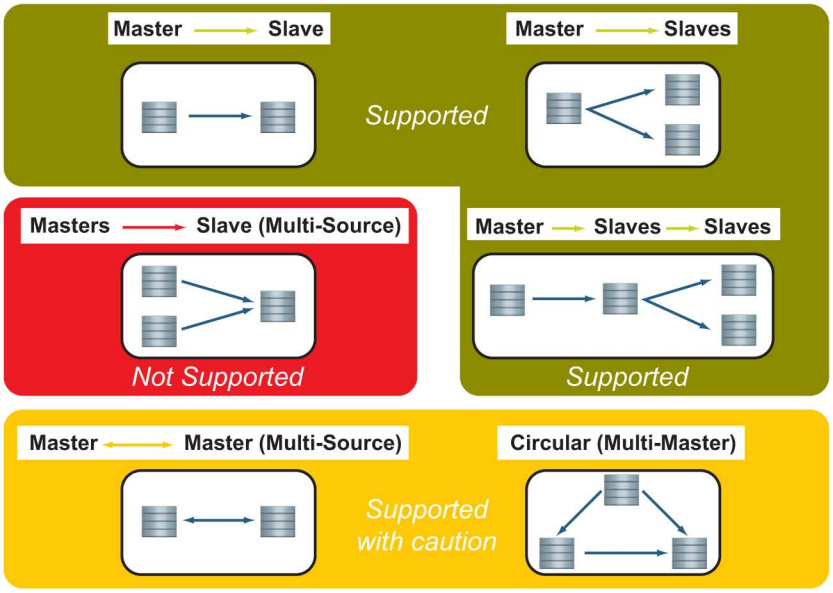


Figura 3. Topologías de replicación MySQL.

Arquitecturas básicas de escalado horizontal

Con el transcurso del tiempo, muchas bases de datos necesitan ampliar su capacidad de respuesta ante aumentos en el número de peticiones de los usuarios. Otras pueden necesitar ser capaces de resistir fallos hardware sin parar el servicio y lo que buscan son soluciones de alta disponibilidad. La replicación MySQL es un medio para ayudar a resolver estos problemas tan comunes.

Escalado horizontal para operaciones de lectura

El uso más habitual de las bases de datos es para servir información almacenada en ellas en respuesta a las peticiones de los usuarios. Un problema muy común en las bases de datos, a medida que crece su volumen, es que deben ser capaces de realizar muchas más operaciones de lectura que de escritura. Esta situación se presenta cuando la aplicación que accede a la base de datos se convierte en popular y sobrepasa la capacidad del hardware que se aprovisionó originalmente. El problema, por lo tanto, es el de ser capaz de soportar cada vez más operaciones de lectura, con solo un pequeño aumento en las operaciones que modifiquen el contenido de la base de datos. Este planteamiento es conocido como read scaling, o escalado de lectura, y es la más común de las formas de replicación.

En la Figura 4 se muestra una arquitectura para el escalado horizontal de operaciones de lectura. Esta arquitectura añade múltiples servidores asignados a operaciones READ y es bastante adecuada para aplicaciones con un uso intensivo de operaciones de lectura. Todas las operaciones de escritura son efectuadas por un servidor maestro. La base de datos es replicada sobre múltiples servidores esclavos, que son los que gestionan las peticiones de lectura.

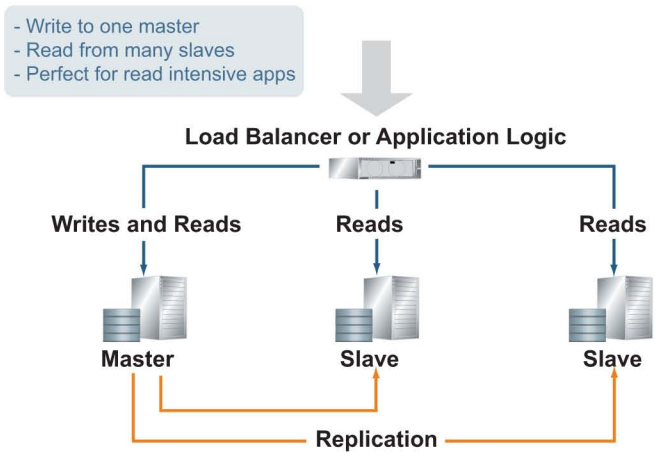


Figura 4. Arquitectura básica para el escalado horizontal de las operaciones de lectura.

Replicación asíncrona

Es realmente costoso, en términos de proceso de la base de datos, realizar una modificación o adición a una base de datos. Es mucho más sencillo dimensionar múltiples servidores para las operaciones de lectura y dejar la operativa de modificación en un solo servidor maestro.

Al contrario que en el clustering, la replicación en MySQL envía todas las modificaciones a la base de datos al servidor maestro, mientras que las operaciones de lectura recaen sobre los servidores esclavos escalados a tal efecto. Las modificaciones sobre el servidor maestro son realizadas y comprobadas en la base de datos de la manera habitual. En la replicación asíncrona (a partir de ahora la llamaremos simplemente replicación, a no ser que se especifique lo contrario) permite que dichas modificaciones sean enviadas a cualquier cantidad de servidores esclavos, con posterioridad a la escritura y comprobación original sobre el servidor maestro. Por lo tanto, las modificaciones realizadas sobre la base de datos pueden no estar disponibles inmediatamente en los servidores esclavos que gestionan el grueso de las operaciones de lectura. Este efecto es apreciable en muchas grandes Webs sociales en donde las actualizaciones realizadas por un usuario pueden ser consultadas por el mismo usuario, pero lleva algo de tiempo que el resto de usuarios del mismo site puedan acceder a ellas.

Balanceo de carga

El servidor MySQL incluye tecnologías de replicación que permiten el escalado horizontal de las bases de datos muy rápidamente, empleando hardware de bajo coste y amplia presencia en el mercado. La capacidad para conectar fácilmente varios servidores pequeños entre si proporciona una gran flexibilidad, lo que permite que una organización pueda emplear diversas configuraciones de replicación para resolver un determinado problema. Sin embargo, debido a esta flexibilidad, no existe una tecnología unificada que actúe como un servidor proxy MySQL general para la replicación. En este momento, aunque existen varios proyectos en curso para el desarrollo de aplicaciones proxy MySQL, ninguna ha trascendido al nivel de producto completamente funcional. Propiamente dicho, el principal problema que una organización

necesita considerar cuando implanta una arquitectura escalada horizontalmente es la separación real de las peticiones SELECT/READ/WRITE hacia servidores MySQL distintos.

En este tipo de arquitecturas es necesario utilizar balanceo de carga para separar las peticiones de los usuarios, antes de que éstas lleguen a los servidores de back-end. Cuando se configura un entorno replicado no existe un intermediario que haga posible esta separación. Por lo tanto, es necesario utilizar un balanceador de carga, tanto si es hardware como software.

Un balanceador hardware es un equipo diseñado especialmente para leer las peticiones de entrada y utilizar su lógica de proceso para dirigir éstas hacia los servidores MySQL adecuados situados en el back end. El mismo resultado es posible conseguirlo por software. Pero ello precisa modificar la propia aplicación, de tal manera que ésta sepa qué topología de replicación existente en el back end y redirija las peticiones READ de acuerdo con la disposición de los servidores de solo lectura añadidos en el escalado horizontal.

Particionado de aplicaciones

El particionado de aplicaciones, o sharding, es otra técnica que puede ser utilizada para escalar horizontalmente una implementación MySQL. El término shard significa “un trozo de” y es empleado en este contexto para hacer referencia a la distribución de las operaciones de escritura, no de las de lectura.

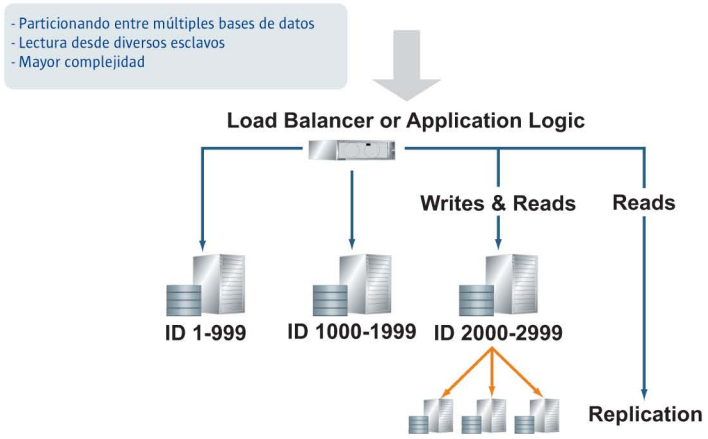


Figura 5. Particionado de aplicaciones, o sharding.

Dado que la replicación en lectura dispone de múltiples copias de solo-lectura de la base de datos, el particionado de aplicaciones o sharding todavía dispone de múltiples servidores físicos, pero entre ellos lo que se articula son secciones de la base de datos principal (Figura 5). A modo de ejemplo, asumamos que los usuarios finales de la base de datos están categorizados por su userid, o identificación de usuario. En este caso, la implementación puede ser escalada en tres servidores, con un tercio del rango de userid residiendo en cada uno de los servidores.

Asumiendo la carga del total de los usuarios en un momento dado y distribuyéndola equitativamente, los servidores, por lo tanto, unicamnete estarán ocupados 1/3 respecto a un solo servidor asumiendo todas las funciones de la base de datos.

En una configuración sharding pura, como se muestra en este ejemplo, la caída de cualquier servidor puede ocasionar el corte en el servicio a los usuarios de dicho servidor. En una configuración puramente de replicación en lectura, la pérdida de un servidor esclavo solamente ocasionará una caída temporal para los usuarios finales, unicamnete hasta que el mecanismo de balanceo de carga redirija la carga de trabajo de los usuarios afectados hacia el resto de los servidores en funcionamiento.

Layering Techniques

A pesar de que las técnicas descritas en este documento pueden ser aplicadas individualmente, también pueden emplearse en conjunto para disfrutar de las ventajas de todas ellas. Por ejemplo:

- Replicación en lectura mediante esclavos de réplica que permiten escalar las operaciones de lectura, pero que es vulnerable a un posible fallo en el servidor maestro (en cuyo caso solo podrían realizarse operaciones de lectura).
- Particionado de aplicaciones/sharding que ayuda a escalar las operaciones de escritura, pero es todavía más vulnerable a un fallo en el servidor, puesto que existen más servidores físicos (en cuyo caso no se podrían realizar operaciones de escritura ni lectura en el servidor caído).
- Utilizar sharding combinado con esclavos de réplica ofrecería escalabilidad tanto el lectura como en escritura (en donde un fallo en un servidor maestro todavía impediría actualizar dicho servidor, pero permitiría todas las operaciones de lectura).

Cuando pensamos en combinar o superponer en capas una técnica encima de la otra, no consideremos los servidores como el elemento básico para la replicación. Incluso para instalaciones pequeñas, a menudo suele necesitarse crear una copia de la configuración completa de la base de datos en otro lugar para disponer de recuperación ante desastres. Si algo grave sucediera en la sede principal, un centro de datos alternativo pasaría a restablecer el servicio.

La replicación puede ser utilizada para obtener redundancia geográfica. Una arquitectura MySQL al completo puede, mediante replicación, ser recreada en otro centro de datos independiente. Este proceso es asíncrono, como una réplica normal, pero el centro de datos de respaldo, o failover, dispondrá de una copia de la base de datos razonablemente actualizada en caso de producirse un fallo.

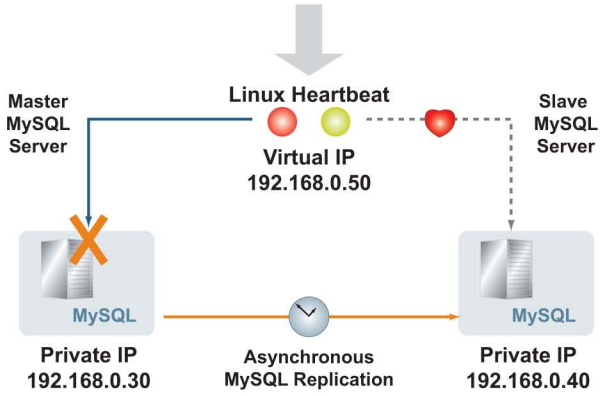
Precaución – ¡La replicación geográfica desde un centro de datos a otro normalmente no dispone de failover automático, tampoco de resincronización automática una vez que se ha restablecido el servicio en el centro de datos principal!

Linux Heartbeat

El sistema operativo Linux dispone de una utilidad denominada heartbeat (latido del corazón) que permite a dos servidores (o grupos de servidores) emplear una conexión entre ellos para determinar si alguno ha sufrido algún problema y no funciona con normalidad. Esto permite que un grupo de servidores actúe como conjunto MySQL principal y el otro como grupo en espera, listo para tomar el control si el principal sufriera problemas.

En la Figura 6 se muestra un ejemplo de configuración empleando el mecanismo Linux heartbeat. Existe un flujo de mensajes entre el servidor maestro y el esclavo para determinar si alguno de ellos tiene problemas. Para ello se utiliza una dirección IP virtual, que sirve para redireccionar las peticiones de entrada hacia el servidor activo. En este ejemplo, el sistema opera con normalidad y todas las peticiones que son recibidas en la dirección IP virtual 192.168.0.50 son dirigidas a la dirección IP 192.168.0.30 privada del servidor.

En esta configuración, el grupo de servidores alternativo (esclavo) es actualizado mediante una réplica asíncrona y, por lo tanto, dispone de un conjunto de datos razonablemente actualizado en caso de failover. Sin embargo, sigue siendo una réplica asíncrona: cuando un fallo en el grupo de servidores principal inicia un procedimiento de failover, el grupo de servidores en espera puede disponer de algunas operaciones de escritura no actualizadas, ya que éstas operaciones, realizadas sobre el servidor maestro, puede que todavía no hayan sido replicadas sobre el servidor esclavo.



En esta configuración, el grupo de servidores alternativo (esclavo) es actualizado

mediante una réplica asíncrona y, por lo tanto, dispone de un conjunto de datos razonablemente actualizado en caso de failover. Sin embargo, sigue siendo una réplica asíncrona: cuando un fallo en el grupo de servidores principal inicia un procedimiento de failover, el grupo de servidores en espera puede disponer de algunas operaciones de escritura no actualizadas, ya que éstas operaciones, realizadas sobre el servidor maestro, puede que todavía no hayan sido replicadas sobre el servidor esclavo.

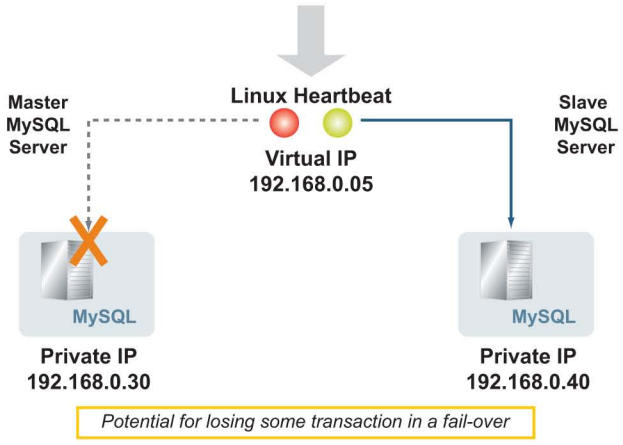


Figura 7. Linux heartbeat y replicación MySQL, después de producirse un fallo.

Después de producirse un fallo y que el servicio haya sido redirigido hacia el servidor esclavo, es necesario establecer procedimientos de fail back, o restauración del servicio sobre el servidor maestro, una vez resuelto el problema. Muchas veces, el mecanismo de failover desde los servidores principales hacia los de backup/replicados es bien conocido y probado, pero el procedimiento de restauración puede ser más largo y complejo de lo que sería de esperar

No existe un comando “ejecutar una replicación inversa” en MySQL y la recuperación hacia el lado contrario del habitual puede ser en ocasiones algo complicado.

La función heartbeat de Linux normalmente requiere una conexión serie entre los servidores a implementar. Esta solución es de código abierto y fácil de configurar, lo que la convierte en una opción muy económica de implementar y mantener. Una vez implementada, la gestión de las IPs virtuales es automática.

Configurar la funcionalidad heartbeat puede ser muy útil a la hora de desplegar grupos de servidores en centros de datos independientes. Aunque la replicación entre estos grupos de servidores no dispone de ninguna función en la arquitectura MySQL para realizar un failover automático, se puede utilizar heartbeat para implementar un auto-failover.

Distributed Replicated Block Device (DRBD)

El mecanismo de replicación MySQL puede emplearse para escalar las operaciones de lectura (READ) y escritura (UPDATE, INSERT, DELETE). La replicación geográfica también puede ser implementada para proporcionar cierta seguridad en caso de producirse un fallo en un grupo de servidores. Sin embargo, todavía existen dos principales inconvenientes en los planteamientos formulados hasta este momento:

- La replicación asíncrona aumenta la posibilidad de disponer de un versión de la base de datos algo desactualiza dando servicio a los usuarios en caso de ejecutar un failover.

- Puede llevar algo de tiempo recuperar el estado original de la topología de la base de datos.

Al contrario que con la replicación asíncrona, el servicio Distributed Replicated Block Device (DRBD) presente en MySQL ayuda a solventar estos problemas, ya que mantiene en todo momento actualizadas todas las copias de datos en todos los servidores. Por tanto, si un servidor maestro sufriera un fallo, todos los servidores esclavos dispondrán de copias actualizadas y disponibles de la base de datos.

El acrónimo DRBD puede definirse como:

- *Distribuido (D)* — Se ejecuta en múltiples servidores.
- *Replicado (R)* — Realiza las tareas de replicación de la base de datos.
- *Bloque (B)* — La replicación se realiza a nivel bloque/hardware, no a nivel de base de datos.
- *Dispositivo (D)* — Implementa un driver de dispositivo que realiza la sincroni zación, no a nivel de base de datos.

El servicio DRBD de MySQL permite la replicación síncrona de los bloques de datos que conforman la propia base de datos. DRBD funciona sobre redes IP estándar entre servidores (sin necesidad de hardware especial), con el failover y la direcciones IP virtuales gestionadas por la función heartbeat de Linux (Figura 8). DRBD puede ofrecer un gran rendimiento ya que la complejidad inherente a toda base de datos queda oculta bajo DRBD — DRBD se ocupa solamente de los bloques de datos, no de la configuración de la base de datos con todas sus complejidades. Sí es cierto que DRBD supone algo más de trabajo a la hora de la configuración inicial, pero las ventajas que luego se pueden disfrutar, con la replicación y recuperación ante fallos, hacen que este pequeño inconveniente merezca la pena.

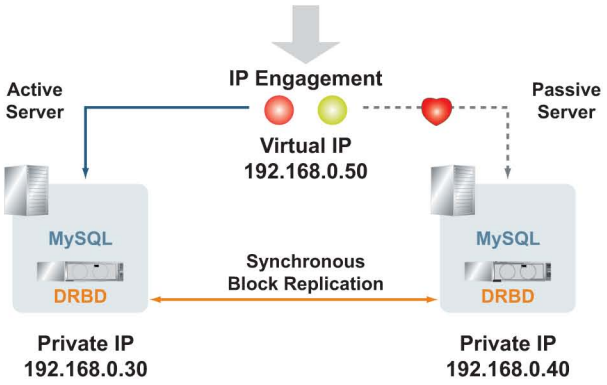


Figura 8. Linux heartbeat y distributed replicated block device (DRBD).

DRBD actualiza de forma síncrona ambos servidores, por lo que si se produjese un failover, o recuperación después de un fallo, de la base de datos, no existirían copias desactualizadas de la misma. Además, cuando la causa que ha provocado un fallo en un servidor ha sido subsanada, DRBD copiará, a nivel bloque, la base de datos activa nuevamente sobre el servidor principal, poniendo al día todas las bases de datos. Esta función permite que una recuperación resulte mucho menos complicada y emplee menos tiempo que lo que sería normal en un escenario de replicación tradicional. Ya que la base de datos se encuentra actualizada en todo momento, los administradores no tienen que preocuparse de la problemática de fusionar dos bases de datos con datos antiguos y modificados.

DRBD y Replicación

A pesar de que DRBD puede utilizarse como función por si sola en la replicación MySQL, el poder real de esta tecnología se muestra en toda su extensión en combinación con otras funciones presentes en MySQL. Empleando DRBD conjuntamente con la replicación de lectura (Figura 9) permite que los servidores de base de datos (que suelen estar mejor equipados desde el punto de vista hardware) realicen de forma sincronizada las operaciones de escritura, dejando a los servidores esclavos el grueso de las operaciones de lectura. Al estar gestionado mediante la función heartbeat de Linux, el único servidor de base de datos activo puede sufrir un fallo, el servidor pasivo se convertiría en el activo, no se sufriría pérdida de datos alguna y los usuarios no advertirían ningún corte en el servicio.

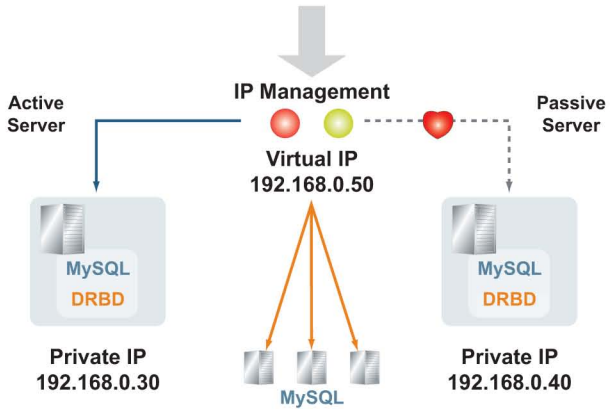


Figura 9. DRBD con replicación MySQL.

Clusters DRBD, particionado de aplicaciones y replicación

Un último ejemplo describe todas las posibilidades y utiliza los casos comentados con anterioridad para concentrarlos en una topología. La Figura 10 muestra una topología que incorpora DRBD, particionado de aplicaciones y replicación. Empleando DRBD para disponer de coherencia de servidores, particionado de aplicaciones para escalabilidad en escritura, así como múltiples servidores esclavos replicados para escalabilidad en lectura, una base de datos puede disfrutar de redundancia, además de poder manejar una elevada cantidad de peticiones. Si fuese necesario, esta arquitectura puede ser replicada geográficamente de forma asíncrona contra un centro de datos de failover, disponiendo en este caso de una magnífica solución de recuperación ante desastres.

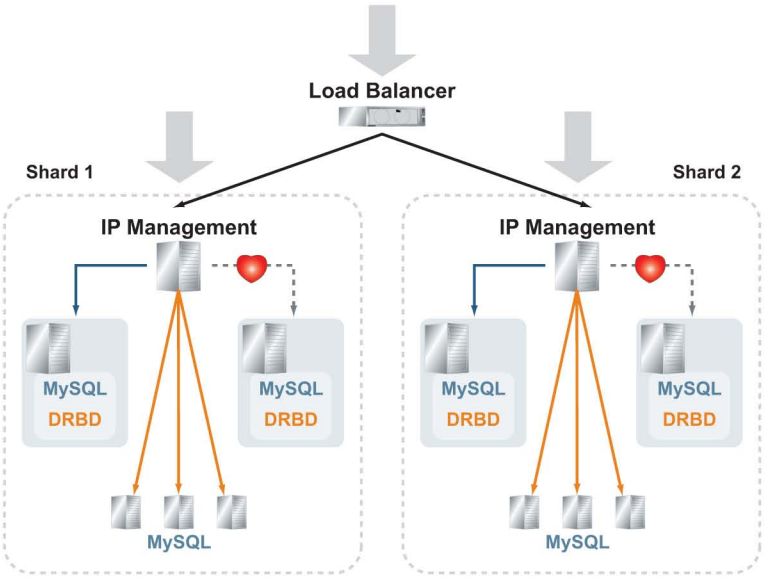


Figura 10. Clusters DRBD, particionado de aplicaciones y replicación.

Directrices para implementar un escalado horizontal

Las siguientes directrices son importantes a la hora de planificar el escalado horizontal y replicación de bases de datos MySQL:

- No pensar sincronizadamente
  No es necesario que cada acceso a la base de datos se realice al mismo tiempo.

- No pensar verticalmente
  Algunas veces el mejor rendimiento puede obtenerse añadiendo servidores estándar para dar servicio en determinados segmentos de la arquitectura.

- No mezclar transacciones y business intelligence
  Ya que es fácil configurar servidores esclavos de replicación, ejecutar aplicaciones de business intelligence contra estos servidores evita que el servidor principal realice las operaciones de escritura.

- Evitar mezclar datos estáticos y dinámicos
  Cuando existen servidores de replicación, asegurarse de almacenar más tablas estáticas en ellos y mantener las tablas más frecuentemente actualizadas en el servidor MySQL principal.

- No olvidar el poder de la memoria
  Se necesita disponer de suficiente memoria para gestionar adecuadamente la carga transaccional. Muchos de los problemas, que en ocasionas fuerzan a una empresa a adoptar una estrategia de replicación complicada, se podrían resolver configurando una adecuada cantidad de memoria.

En el Apéndice "Resumen de configuración de replicación en MySQL" en la página 18 se incluye la configuración básica de esta técnica.

Monitorización MySQL y servicios profesionales

La organización MySQL ofrece muchas vías para facilitar la creación y evaluación de arquitecturas de replicación. Por lo que respecta al software, la aplicación MySQL Enterprise Monitor ayuda al administrador de la base de datos a comprobar el estado de los destinatarios de la réplica, determinar qué problemas pueden ocasionar que un servidor esclavo no reciba los datos precisos y también a comprobar el estado de todas las tareas de replicación en un servidor activo. Al adquirir una suscripción al servicio de soporte MySQL Enterprise Silver, se obtiene acceso al panel de control Enterprise Dashboard, mientras que el nivel de soporte MySQL Enterprise Gold disfruta de funciones más avanzadas, como las utilidades Replication Advisor y Memory Advisor.

Aunque el núcleo de la base de datos MySQL es completamente gratuito para su descarga y utilización, las empresas que deseen maximizar su experiencia MySQL disponen de la posibilidad de contratar soporte adicional. Por ejemplo, el producto MySQL Enterprise incluye los siguientes componentes, según el nivel de suscripción:

MySQL Enterprise Unlimited es otra opción que incluye soporte para entornos de producción corporativos por una cuota fija, que incluye soporte para un número ilimitado de servidores MySQL.

- Software MySQL Enterprise Server — la versión más reciente, fiable y segura de la base de datos MySQL
- Servicio de actualización del software MySQL — proporciona un servicio de alertas proactivo que informa de la aparición de nuevos productos, parches de seguridad y solución de errores
- Base de conocimiento MySQL — recursos de auto-ayuda que permite a los usuarios realizar búsquedas en una completa librería de artículos técnicos para resolver problemas difíciles dentro de los temas más populares en la base de datos

- Soporte en producción MySQL — soporte proporcionado por expertos de la organización MySQL, los cuales pueden aconsejar y prestar asistencia en aspectos como rendimiento, diseño y estrategias de implementación de la base de datos. También pueden ayudar a resolver problemas rápida y eficazmente
- Servicios de monitorización y asesoramiento MySQL — actúan con un asistente DBA virtual cuyo objetivo es ayudar a aplicar las mejores prácticas y recomendaciones en todos los servidores MySQL

Nota – Visitar http://www.mysql.com/products/enterprise/features.html para conocer más detalles acerca de las características de MySQL Enterprise.

Resumen

MySQL, la base de datos en uso más popular actualmente, dispone de funciones en la que muchas empresas confían para su operativa diaria, en sus servidores web y despliegues corporativos. A pesar de que existen otras bases de datos en el mercado de precios realmente elevados, MySQL a veces ofrece lo que otras no pueden: la capacidad de poder implementar una solución de base de datos de ámbito empresarial en prácticamente con cualquier hardware, sin coste.

A medida que aumenta la popularidad de una aplicación y crecen con ella las bases de datos de back-end, es necesario considerar cuidadosamente cómo escalar los despliegues MySQL. Una instalación MySQL inadecuadamente escalada puede afectar negativamente al rendimiento y desembocar en una excesiva complejidad en la infraestructura de la base de datos. Este documento proporciona información sobre métodos y estrategias para escalar horizontalmente una base de datos MySQL y evitar estos problemas. Planifique cuidadosamente su estrategia ayudándose de este documento como punto de partida. Puede obtener más ayuda y soporte recurriendo a la comunidad de usuario de MySQL y también mediante servicios de soporte profesionales, como el Soporte en producción MySQL. Con planificación, una implementación de bases de datos MySQL puede ser fácilmente gestionada para cubrir las demandas y crecimiento ahora y en el futuro.

Acerca del autor

Nick Kloski es un arquitecto de soluciones Web2.0 encuadrado en el grupo Web/HPC dentro del Systems Technical Marketing Group de Sun. Durante sus 10 años en Sun, Nick ha atesorado una gran experiencia, tanto en sistemas Sun como de otros fabricantes, ya que ha trabajado como administrador de sistemas durante más de seis años en soporte a clientes, equipos internos de testing y calidad, así como en su actual puesto como miembro del departamento de Marketing Técnico. En su papel de arquitecto de soluciones Web2.0, Nick es responsable de estar al corriente de las tendencias del mercado y de descubrir vías por las que la tecnología de Sun pueda ayudar a resolver problemas de nuestros clientes.

Agradecimientos

El autor quisiera agradecer a Jimmy Guerrero del MySQL Marketing Team su contribución a este documento.

Referencias

Base de datos MySQL: http://www.sun.com/software/products/mysql/index.jsp.

Documentación sobre MySQL: http://dev.mysql.com/doc/index.html MySQL

Enterprise: http://www.mysql.com/products/enterprise/features.html

Solicitar documentación de Sun

El programa SunDocsSM proporciona más de 250 manuales de Sun Microsystems, Inc. Si usted tiene su residencia en Estados Unidos, Canadá, Europa o Japón, puede adquirir paquetes de documentación o manuales sueltos a través de este programa.

Acceso online a documentación de Sun

La web docs.sun.com le permite acceder online a la documentación técnica de Sun. Puede navegar libremente por el archivo de docs.sun.com o buscar un título o tema concreto. La URL es

http://docs.sun.com/

Para consultar los artículos de Sun BluePrints Online, visite:
http://www.sun.com/blueprints/online.html

Apéndice: Resumen de configuración de replicación en MySQL

Los siguientes pasos proporcionan una configuración básica a alto nivel de la replicación en MySQL. Dichos pasos no muestran los comandos específicos necesarios para crear un esclavo de replicación MySQL, pero se ha incluido como herramienta conceptual para proporcionar un resumen o vista general del proceso de replicación.

Nota – Para disponer de información detallada acerca de la configuración de replicación, consultar la documentación de MySQL para la versión específica que se esté utilizando. Por ejemplo, el Manual de Referencia de MySQL 5.1, capítulo 15 “Replicación” y el Manual de Referencia de MySQL 5.0, en el mismo capítulo, ambos describen cómo establecer un servicio de replicación al completo de un servidor MySQL en las versiones 5.1 y 5.0 respectivamente.

---

1. Asegúrese de que los servidores maestro y esclavo están configurados con IDs únicas.

2. Otorgue privilegios de replicación al esclavo

3. Editar el fichero my.cnf (log binario, etc.) del maestro y esclavo

4. Reinicie el servidor maestro

5. Copie los datos iniciales del maestro en el(los) esclavo(s):
   -Backup/Restauración de datos sin modificar
   -Mysqldump — opción master-data
   -BLOQUEE las tablas y copie los ficheros

6. Ejecutar el comando MySQL show master status para registrar el fichero del índice y su posición

7. Ejecutar el comando MySQL stop slave

8. Ejecutar el comando MySQL change master
   -master_host=, master_user=, master_password,master_log_file=, master_log_pos

9. Ejecutar el comando MySQL start slave

Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 USA **Phone** 1-650-960-1300 or 1-800-555-9SUN (9786) **Web** sun.com

Printed in USA 11/08