Unidad V: Seguridad

5.1 Respaldo y Recuperación

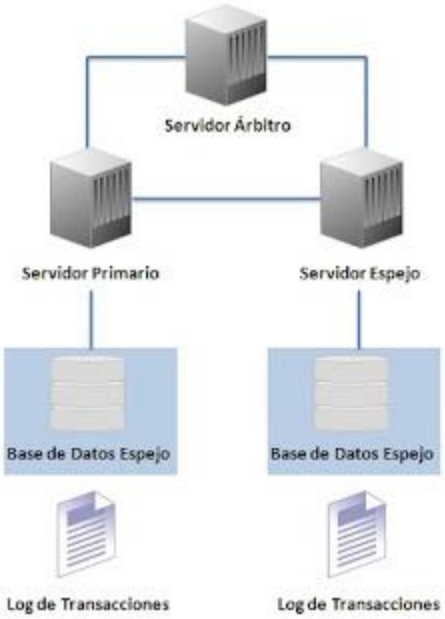
Las operaciones de backup y restore son actividade crítica y de orden crucial para cualquier organización, pues por motivos varios una base de datos puede llegar a fallar, los sistemas operativos, el hardware, crackers y hasta los mismos empleados pueden dañar la información. Es por eso que es importante definir políticas de backup en una organización o por lomenos calendarizar la realización de copias de seguridad para estar preparado ante cualquier eventualidad.

Dependiendo del gestor que se utilice y el tamaño de la base de datos, este puede ser una tarea fácil o relativamente compleja.

5.1.1 Espejeo (mirroring)

Base de Datos Espejo (Database Mirroring) es una configuración donde dos o tres servidores de base de datos, ejecutándose en equipos independientes, cooperan para mantener copias de la base de datos y archivo de registro de transacciones (log).

Tanto el **servidor primario** como el **servidor espejo** mantienen una copia de la base de datos y el registro de transacciones, mientras que el tercer servidor, llamado el**servidor árbitro**, es usado cuando es necesario determinar cuál de los los otros dos servidores puede tomar la propiedad de la base de datos. El árbitro no mantiene una copia de la base de datos. La configuración de los tres servidores de base de datos (el primario, el espejo y el árbitro) es llamado Sistema Espejo (Mirroring System), y el servidor primarioy espejo juntos son llamados Servidores Operacionales (Operational Servers) o Compañeros (Partners).



Para hacer el mirror, es necesario como mínimo 2 instancia y como máximo 3. Si utilizamos 2 instancias, una de ellas contiene la base de datos y la otra la espejo. La pega de esta configuración es que el failover no es automático y se necesita intervención humana. Si utilizamos 3 instancias, entonces utilizamos una de ellas como witness server y permite que el failover sea automático, osea que cuando una caiga, la otra se ponga en marcha. Para ello el witness server se encarga de "mirar" el estado de las 2 instancias y cuando una de ellas cae, pone la otra en marcha.

Hacer el mirror son dos pasos principales:

1. Copiar y restaurar la base de datos de la que queremos hacer el mirror desde una instancia a la otra
2. Configurar el asistente de configuración del mirror.

Vamos un ejemplo paso a paso.

Lo primero que tenemos que hacer es hacer un reflejo de nuestra base de datos en otra instancia. En nuestro ejemplo esta base de datos se denomina prueba.

5.1.1.1 Beneficios del espejeo de Datos en un DBMS

La creación de reflejo de la base de datos es una estrategia sencilla que ofrece las siguientes ventajas:

- **Incrementa la disponibilidad de una base de datos**. Si se produce un desastre en el modo de alta seguridad con conmutación automática por error, la conmutación por error pone en línea rápidamente la copia en espera de la base de datos, sin pérdida de datos. En los demás modos operativos, el administrador de bases de datos tiene la alternativa del servicio forzado (con una posible pérdida de datos) para la copia en espera de la base de datos. Para obtener más información, vea Conmutación de roles, más adelante en este tema.
- **Aumenta la protección de los datos**. La creación de reflejo de la base de datos proporciona una redundancia completa o casi completa de los datos, en función de si el modo de funcionamiento es el de alta seguridad o el de alto rendimiento. Para obtener más información, vea Modos de funcionamiento, más adelante en este tema.

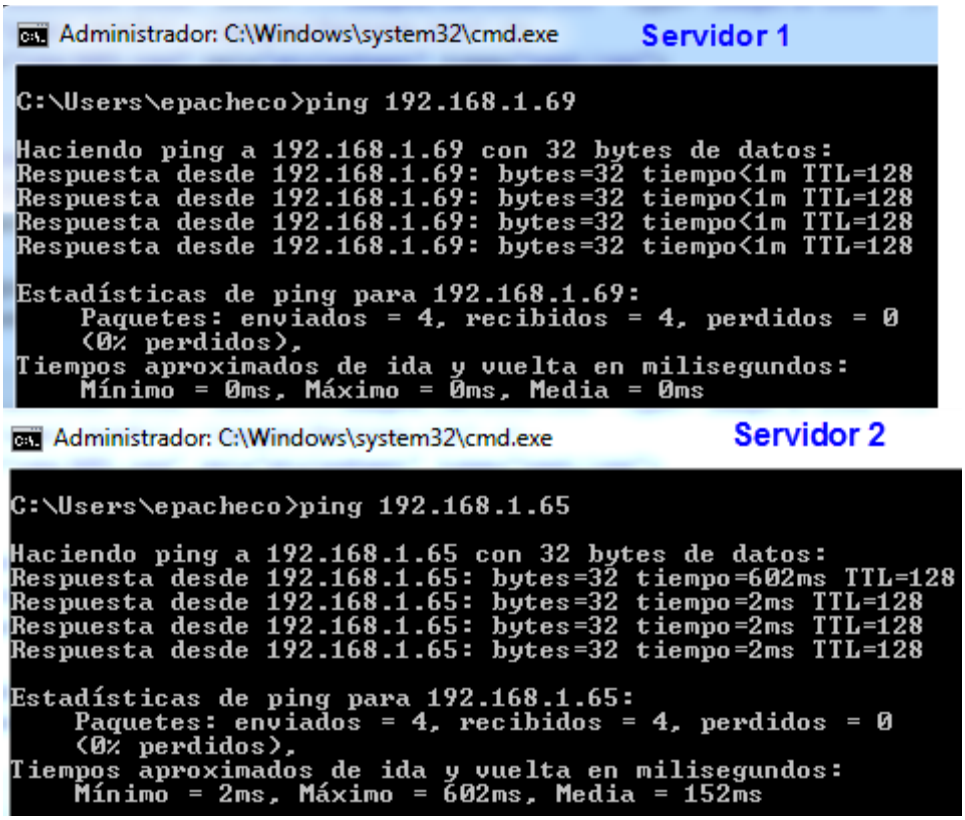
  Un asociado de creación de reflejo de la base de datos que se ejecute en SQL Server 2008 Enterprise o en versiones posteriores intentará resolver automáticamente cierto tipo de errores que impiden la lectura de una página de datos. El socio que no puede leer una página, solicita una copia nueva al otro socio. Si la solicitud se realiza correctamente, la copia sustituirá a la página que no se puede leer, de forma que se resuelve el error en la mayoría de los casos. Para obtener más información, vea Reparación de página automática (grupos de disponibilidad/creación de reflejo de base de datos).
- **Mejora la disponibilidad de la base de datos de producción durante las actualizaciones**. Para minimizar el tiempo de inactividad para una base de datos reflejada, puede actualizar secuencialmente las instancias de SQL Server que hospedan los asociados de creación de reflejo de la base de datos. Esto incurrirá en el tiempo de inactividad de solo una conmutación por error única. Esta forma de actualización se denomina actualización gradual. Para obtener más información, vea Instalar un Service Pack en un sistema con un tiempo de inactividad mínimo para bases de datos reflejadas.

5.1.1.2 Activación de espejeo en un DBMS

MySQL

Lo primero que debemos hacer es checar si ambos servidores se encuentran en red

Caso Windows



Caso Linux

Cambie el comando ipconfig por ifconfig

Software

Verifque que el MySQL instalado en el maestro y en el esclavo son iguales. En este casp MySQL Server 5.6

Configuración del Maestro

Localizar el archivo My.ini -Windows- (My.cnf -Linux)

Buscar y comentar las siguientes lineas si es que se encuentran:

#skip-networking

#bind-address = 127.0.0.1

Agregar después de la línea [mysqld] lo siguiente:

log-bin =mysql-bin.log

binlog-do-db=dolar

server-id=1

Nota: El server-id en el servidor siempre será 1, y los esclavos serán 2, 3… n según sea el caso en binlog-do-db se pone el nombre de la base de datos que replicara después de signo =

Desde el panel de control entramos en Herramientas administrativas, Servicios y reanudamos MySQL. Este paso se omite en Linux

Ahora en el shell de mysql genere una cuenta para el esclavo con el privilegio REPLICATION SLAVE:

GRANT REPLICATION SLAVE ON *.* TO 'esclavo1'@'%' IDENTIFIED BY 'bingo';

FLUSH PRIVILEGES;

Nota: esclavo1 es el usuario identificado por el passwword bingo.Los posteriores replicadores deberán ser esclavo2, ...,esclavo-n.

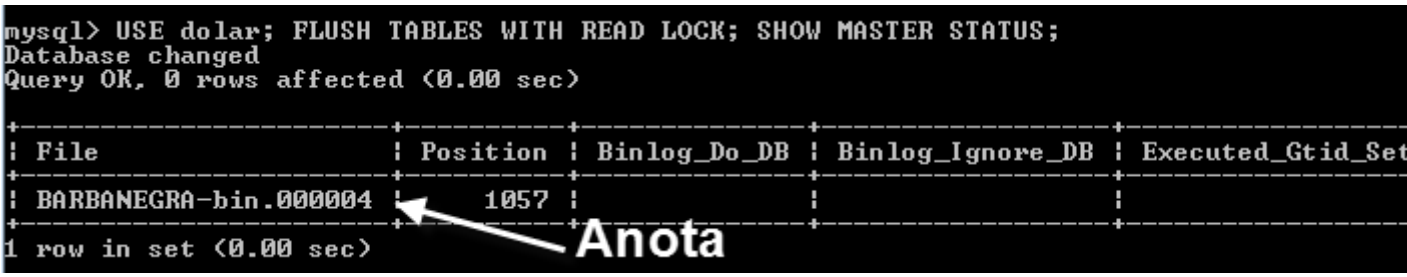
Seleccione la base de datos a replicar y realice lo siguiente:

USE dolar;

FLUSH TABLES WITH READ LOCK;

SHOW MASTER STATUS;

El resultado será algo similar a la figura



La columna File muestra el nombre del log, mientras que Position muestra el desplazamiento. En este ejemplo, el valor del log binario es BARBANEGRA-bin.000004 y el desplazamiento es 1057. Guarde los valores. Los necesitará más tarde cuando inicialice el servidor. Estos representan las coordenadas de la replicación en que el esclavo debe comenzar a procesar nuevas actualizaciones del maestro.

Salir de MySQL usando el comando exit o quit.

Ahora desde la terminal o en el cmd haremos un Backup de la Base de Datos que se encuentra en el Maestro para tener el mismo esquema y datos en los esclavos:

mysqldump -u root -p -dolar > dolar.sql

Por último desbloqueamos la base de datos

mysql -u root -p

UNLOCK TABLES;

quit;

Configuración del esclavo

Crear la base de datos que queremos replicar:

mysql -u root -p

CREATE DATABASE dolar;

quit;

Ejecutar desde la consola o a terminal el siguiente comando para copiar la base de datos del archivo que generamos:

mysql -u root -p dolar < dolar.sql

Localizar el archivo My.cnf (en caso de windows My.ini) y después del [mysqld] agregamos lo siguiente:

server-id=2

replicate-do-db=nombre_base_de_datos

En nuesto caso

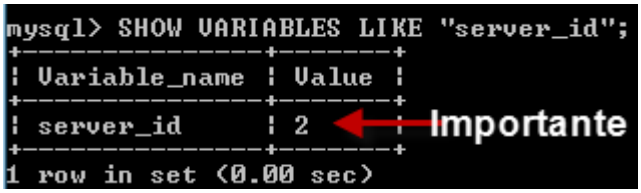
server-id=2

replicate-do-db=dolar

Reiniciamos el servicio de MySql y comprobamos el server-id,

mysql -u root -p

SHOW VARIABLES LIKE "server-id";



Ahora le indicaremos al esclavo la dirección del maestro, el usuario, password y directivas de control (master_log_file y master_log_pos)

CHANGE MASTER TO master_host = '192.168.1.65', master_user='esclavo1',

master_password='bingo', master_log_file='barbanegra-bin.000004', master_log_pos=1057;

Nota: Si olvido las directivas de control. Desde la consola del maestro use la sentencia SHOW MASTER STATUS;

Ahora iniciamos el esclavo y comprobamos su estado

START SLAVE; SHOW SLAVE STATUS\G;

```
*************************** 1. row ***************************
               Slave_IO_State: Connecting to master
                  Master_Host: 192.168.1.69
                  Master_User: esclavo1
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: barbanegra-bin.000012
          Read_Master_Log_Pos: 2692
               Relay_Log_File: pirata-relay-bin.000001
                Relay_Log_Pos: 4
        Relay_Master_Log_File: barbanegra-bin.000012
             Slave_IO_Running: Connecting
            Slave_SQL_Running: Yes
              Replicate_Do_DB: dolar
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 2692
              Relay_Log_Space: 120
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: NULL
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: 8e5cdc20-c969-11e2-9ee7-782bcbf1fd64
             Master_Info_File: C:\MySQL\data\master.info
                    SQL_Delay: 0
          SQL_Remaining_Delay: NULL
      Slave_SQL_Running_State: Slave has read all relay log; waiting for the slave I/O thread to update i
           Master_Retry_Count: 86400
```

5.1.1.3 Creación de espacios de disco con espejo

Discos espejo

Espejeado de disco significa que se conectan dos unidades de disco al mismo controlador de disco. Las dos unidades se mantienen idénticas cuando el servidor escribe en una unidad (la primaria), posteriormente se escribe en (la secundaria). Si durante la operación falla, la unidad primaria, en su lugar se utiliza la secundaria. Si la secundaria falla, no importa. En ambos casos los usuarios experimentan una breve pausa mientras el servidor se asegura que la unidad está muerta, y luego se regresa al servicio normal.

Como sucede con todas las cosas buenas, hay una desventaja. Para contar con este nivel de confiabilidad, se necesita un segundo disco duro, lo que duplica el costo del almacenamiento de datos. Pero en lo que concierne a su organización, tal vez valga la pena el costo relativamente pequeño de una unidad de disco, para evitar lo que de otra manera seria un desastre. Una de las desventajas de los discos espejos es la perdida de rendimiento. Dado que un controlador maneja dos unidades primarias para escribir los datos en la unidad secundaria. Esto provoca que las escrituras en disco se tarden el doble. En un servidor con carga ligera esto quizás no sea tan malo desde el punto de vista del usuario, ya que el caché de disco del servidor hace que el acceso a disco perezca extremadamente rápido. Sin embargo, la sobrecarga puede llegar a ser significativa en un sistema con carga pesada.

Otra de las desventajas del espejeado es que el controlador de disco duro o los cables de conexión llegan a fallar. Los datos se pueden leer desde la unidad o matriz duplicada sin que se produzcan interrupciones. Es una alternativa costosa para los grandes sistemas, ya que las unidades se deben añadir en pares para aumentar la capacidad de almacenamiento, para los disco espejos. Los discos espejos también llamado "duplicación" (creación de discos en espejo). Se basa en la utilización de discos adicionales sobre los que se realiza una copia en todo momento de los datos que se están modificando. El cual ofrece una excelente

disponibilidad de los datos mediante la redundancia total de los mismos.

Administración del espacio libre en un disco.

Es necesario saber qué bloques están libres. Las opciones son parecidas a las que se pueden usar para administrar espacio en memoria. Mapa de bits. Un bit por bloque. Es eficiente si se puede mantener el mapa entero en memoria. Disco de 1 GB, con bloques de 512 KB requiere un mapa de 256 KB. Usado en los MACS. Lista ligada. En un bloque reservado (fijo) del disco se registran las direcciones de los bloques desocupados. La última dirección apunta no a un bloque libre, sino a otro bloque con más direcciones de bloques libres... En MS-DOS se usa la misma FAT para administrar el espacio libre.

Cachés de disco

Ya que el disco es tan lento comparado con la memoria (unas 10000 veces) resulta rentable usar un caché para mantener en memoria física parte de la información que hay en el disco, de manera que, si en el futuro se requiere un bloque que ya está en memoria, se ahorra el acceso al disco.

Igual que en el caso de memoria virtual, hay que tratar de adivinar qué bloques se van a acceder en el futuro cercano, para mantener esos bloques en el caché. Pero al contrario de lo que ocurre con memoria virtual, no se requiere ningún apoyo especial del hardware para implementar LRU. Ya que todos los accesos a disco pasan por las manos del sistema operativo. Paradójicamente, LRU no es necesariamente la mejor alternativa tratándose de bloques de disco. ¿Qué pasa, por ejemplo, en el caso del acceso secuencial a un archivo? Por otra parte, algunos de los bloques contienen información crítica respecto del sistema de archivos (por ejemplo, un bloque que contiene información del directorio raíz o de un i-node o de los bloques libres). Si este bloque es modificado y puesto al final de la cola LRU, puede pasar un buen tiempo antes de que llegue a ser el menos

recientemente usado, y sea escrito en el disco para ser reemplazado. Si el sistema se cae antes que eso, esa información crítica se perderá, y el sistema de archivos quedará en un estado inconsistente. Se puede modificar un poco LRU, considerando dos factores:

A qué tan probable es que el bloque se necesite de nuevo. Bloques de directorios se suelen usar bastante. El último bloque de un archivo que se está escribiendo, también es probable que se vuelva a necesitar.

A Qué tan esencial es el bloque para la consistencia del sistema de archivos. Básicamente todos los bloques, excepto los de datos, que han sido modificados. Estos deben grabarse en disco lo más rápidamente posible.

Planificación de disco

Un disco, mirado desde más bajo nivel, no es simplemente una secuencia de bloques. Están compuestos de platos, cada uno de los cuales contiene una serie de pistas o tracks concéntricos. A su vez, las pistas se dividen en sectores. Las pistas exteriores, que son más grandes, pueden contener más sectores que las interiores. (En un CD, en realidad hay una espiral de sectores.) Existe un brazo mecánico con un cabezal lector/escritor para cada plato. El brazo mueve todos los cabezales juntos. Un cilindro se conforma por las pistas que los cabezales pueden leer cuando el brazo está en una posición determinada. Los bloques lógicos (secuenciales) que ve el sistema de archivos deben traducirse a un trío (cilindro, plato, sector). El tiempo requerido para leer un sector depende de:

1. El tiempo de búsqueda (seek time), es decir, el tiempo requerido para mover el brazo al cilindro apropiado.
2. El retardo rotacional, o sea, el tiempo que hay que esperar hasta que el sector requerido pase por debajo del cabezal.
3. El tiempo de transferencia de los datos.

El primero es el que predomina, de manera que conviene reducirlo para aumentar la eficiencia del sistema. El sistema de archivo puede ayudar (por ejemplo, con asignación contigua). Obviamente, bloques en el mismo cilindro deben considerarse contiguos. Pero hay otra cosa que se puede hacer, considerando que en un sistema con muchos procesos la cola de solicitudes pendientes de un dispositivo suele no estar vacía: atenderlas en un orden que reduzca los movimientos del brazo.

Algoritmos de planificación de disco *Fifo.*

Es simple, pero no estamos haciendo nada por la eficiencia. Es malo si las solicitudes se alternan entre cilindros exteriores e interiores. Por ejemplo, si, mientras se lee el cilindro 11 llegan solicitudes para los cilindros 1, 36, 16, 34, 9, 12, y se atienden en ese orden, el brazo recorrerá 111 cilindros.

SSTF (shortest seek-time first).

Se trata de atender primero las solicitudes más cercanas a la posición actual del brazo. La atención sería en el orden 11, 12, 9, 16, 1, 34,36, para un total de 61 cilindros de desplazamiento. El problema es que, cuando hay muchas solicitudes, es posible que sólo se atiendan las cercanas al centro. Puede haber inanición para los procesos que solicitan cilindros de los extremos.

Algoritmo del ascensor

Para evitar inanición, se mantiene la dirección de movimiento del brazo hasta que no queden solicitudes pendientes en esa dirección. Es lo mismo que hacen los ascensores. En el ejemplo, suponiendo que el brazo iba hacia las direcciones altas, las solicitudes se atenderían en el orden 11, 12, 16,34,36,9,1, lo que da un total de 60 cilindros de recorrido del brazo. O sea, en este caso en particular es un poco mejor que SSTF, pero en general es peor. Una propiedad interesante es que para cualquier conjunto de solicitudes, el movimiento del brazo está acotado: 2 veces el ancho del disco. Un pequeño problema es que las solicitudes en los

extremos tienen, en promedio, un tiempo de espera mayor. Esto se puede resolver si las solicitudes siempre se atienden en un solo sentido. En el otro sentido, el cabezal se devuelve, pero sin atender solicitudes a su paso. También podríamos pensar en un algoritmo óptimo, pero su complejidad no justifica usarlo. Si la carga es muy poca (la cola nunca tiene más de una solicitud pendiente) todos los algoritmos tienen el mismo rendimiento. Para cargas pesadas, se usa el del ascensor.

Discos RAM

Gracias a la estructuración en capas, podemos usar el mismo sistema de archivos en cualquier dispositivo de bloques con un driver adecuado, que implemente la interfaz para el software independiente del dispositivo. Por ejemplo, en los primeros computadores personales, que tenían sólo una disquetera como medio de almacenamiento, era habitual crear un disco RAM, es decir reservar un trozo de la memoria para usarlo como un disco virtual, para almacenar archivos. Un driver de disco RAM es extremadamente simple.

Dado un tamaño de bloque B, leer o escribir el bloque i es simplemente accesar B bytes a partir de la posición B*i del área reservada para el disco.

Bloques dañados

Los discos, en cuanto dispositivo mecánico, son propensos a fallas. A veces la falla es transitoria: el controlador no puede leer un sector debido a que se interpuso una partícula de polvo entre el cabezal y la superficie del disco. El controlador siempre reintenta varias veces una operación que fracasa por si la falla es transitoria; muchas veces se resuelve, sin que el driver siquiera se entere. En los casos en que el sector está permanentemente dañado, el error se informa al driver, y el driver informa al sistema de archivos, quien registra el bloque como dañado, para no volver a usarlo. ¿Cómo se pueden registrar los bloques dañados? Igual hay bloques críticos: en todo sistema de archivo, debe haber al menos un bloque en una dirección fija. Si ese bloque se daña, el disco entero se hace

inusable. Algunos controladores inteligentes reservan de antemano algunas pistas, que no son visibles para el driver. Cuando se daña un sector, el propio controlador lo reemplaza por uno de los reservados. (en forma transparente, si la operación era de escritura, pero no tan transparente si era de lectura). Muchos discos vienen con sectores dañados ya marcados desde la fábrica. Pero ¿dónde se guarda la información de los bloques malos? Así, si el bloque 5 se daña, entonces el controlador usa, digamos, el 999 cada vez que el driver le solicita el 5. Pero ¿que pasaría entonces con los algoritmos de scheduling de disco? Un esquema que a veces se usa para no perjudicarlos, es que el controlador reserva bloques esparcidos en el disco, y cuando se daña un sector, trata de sustituirlo por uno de los de reserva que se encuentre en el mismo cilindro, o por lo menos cerca.

Arreglos de discos

Se puede decir que los discos son la componente menos confiable de un computador, la componente más complicada de sustituir, y la que frena el mejoramiento de la velocidad de procesamiento con los avances tecnológicos. En efecto, la velocidad de los procesadores se duplica más o menos cada 2 años, y la capacidad de los chips de memoria crece a un ritmo parecido. No obstante, el ancho de banda (velocidad de transferencia) del I/O ha variado muy poco. A este ritmo, en 7 años más los procesadores van a ser 10 veces más rápidos, pero en general las aplicaciones correrán menos de 5 veces más rápido, por las limitaciones de I/O. Una solución posible: en lugar de uno solo disco grande, usar muchos discos chicos y baratos, en paralelo, para mejorar el ancho de banda. Para garantizar paralelismo, se hace disk striping o división en franjas. Cada bloque lógico se compone de varios sectores físicos, cada uno en un disco distinto. Así, cada acceso a un bloque lógico se divide en accesos simultáneos a los discos. En 1991 la situación era la siguiente: _ IBM 3380: 7500 MB, 18 U$/MB, 30000 horas de MTTF (mean time to failure) _ Conner CP3100: 100 MB, 10 U$/MB, 30000 horas de MTTF El IBM 3380 tiene bastante más ancho de banda que un CP3100, pero si juntamos 75 de estos últimos tenemos la misma capacidad total, con menor costo, menor consumo de electricidad, y

potencialmente 12 veces más ancho de banda. El gran problema es la confiabilidad: si antes teníamos 30000 horas de funcionamiento sin fallas, ahora tendríamos 400 (30000/75) horas, o sea, sólo dos semanas. O sea, la tolerancia a fallas es crucial, y para obtenerla se usa redundancia, en una configuración conocida como RAID (Redundant Array of Inexpensive Disks), y que se puede implementar en varios niveles.

RAID 1: Se usan discos espejos, o sea, la información de cada disco se mantiene siempre duplicada en otro idéntico. O sea, MTTF aumenta notoriamente, pero duplicando el costo.
RAID 2: Se reduce la redundancia usando técnicas de detección y corrección de errores (códigos de Hamming). Por ejemplo, si un bloque se reparte entre 10 discos y suponemos que no va a haber más de una falla simultáneamente, entonces no necesitamos duplicar el bloque entero para reconstituirlo en caso de falla, puesto que ante una falla sólo se perderá un 10% de la información. El problema es que si no sabemos qué 10% se perdió, de todas maneras se necesita bastante redundancia (20 a 40%).
RAID 3: El punto es que, gracias a que cada controlador usa sumas de chequeo (y suponiendo que además podemos saber cuándo un controlador falla) sí podemos saber qué trozo de la información está errónea. Y sabiendo eso, basta con usar sólo un disco adicional para guardar información de paridad con la cual es posible reconstituir la información original. Hay otros niveles (RAID 4 y 5). Ahora (1996) la situación es:
1. **IBM 3390:** un disco de 102 GB, 3.9 MB/s, 22.8 ms de latencia.
2. **IBM RAMDAC 2:** 64 discos, 180 GB en total, 12.6 MB/s, 4.2 ms de latencia.

La ganancia en ancho de banda es menor que la teórica, entre otras cosas porque la tolerancia a fallas impone un overhead Ver figura 44. Por otra parte, con un RAID de 100 discos para datos y otros 10 para paridad, el MTDL (mean time to data loss) es de 90 años, comparado con 3 años de los discos estándares.

5.1.2 Replica (replication)

La replicación es un conjunto de tecnologías para copiar y distribuir datos y objetos de bases de datos de una base de datos a otra y, a continuación, sincronizar las diferentes bases de datos para mantener la coherencia. Mediante la replicación, podrá distribuir los datos a diferentes ubicaciones y usuarios remotos o móviles a través de redes de área local y extensa, conexiones de acceso telefónico, conexiones inalámbricas e Internet.

Por lo general, la **replicación de transacciones** se usa en escenarios de servidor a servidor, que requieren un rendimiento alto, donde se incluye: la mejora de la escalabilidad y disponibilidad; el almacenamiento datos y generación de informes; la integración de datos desde múltiples sitios; la integración de datos heterogéneos y la descarga de procesamiento por lotes.

La **replicación de mezcla** se ha diseñado principalmente para aplicaciones móviles que presentan posibles conflictos de datos. Los escenarios comunes incluyen: intercambio de datos con usuarios móviles; aplicaciones de puntos de venta (POS) para el consumidor e integración de datos desde varias ubicaciones.

La **replicación de instantáneas** se usa para proporcionar el conjunto de datos inicial para la réplica transaccional o de mezcla. También se puede usar cuando es necesaria una actualización completa de los datos. Con estos tres tipos de replicación, SQL Server ofrece un sistema eficaz y flexible para la sincronización de datos en toda la empresa.

5.1.2.1 Beneficios de la réplica de Datos en un DBMS

La replicación de base de datos es una herramienta muy potente en el mundo de las aplicaciones distribuidas. Sus aplicaciones en el mundo real son muy variadas. Sin embargo, para que se pueda utilizar de forma correcta y funcione como esperamos es importante conocer realmente cómo funciona y las diferentes opciones que nos ofrece.

Los beneficios o los entornos donde es aplicable la replicación de bases de datos son los siguientes:

- Usuarios trabajando en ubicaciones geográficamente alejados trabajando con sus propias copias locales de la base de datos.
- Entornos en los que se replica la base de datos principal en una secundaria como copia de seguridad. En el caso que la primaria caiga, la secundaria toma el control.
- En entornos en los que la carga de usuarios sea muy grande para un sólo gestor, se pueden replicar las bases de datos en varios servidores asignando a cada usuario un servidor. Balanceando de esta manera la carga podremos aliviar a los gestores.

Como observamos, los entornos son variados y comunes en muchos casos. El problema reside en la configuración y la elección correcta del tipo de replicación

Modelo de Replicación

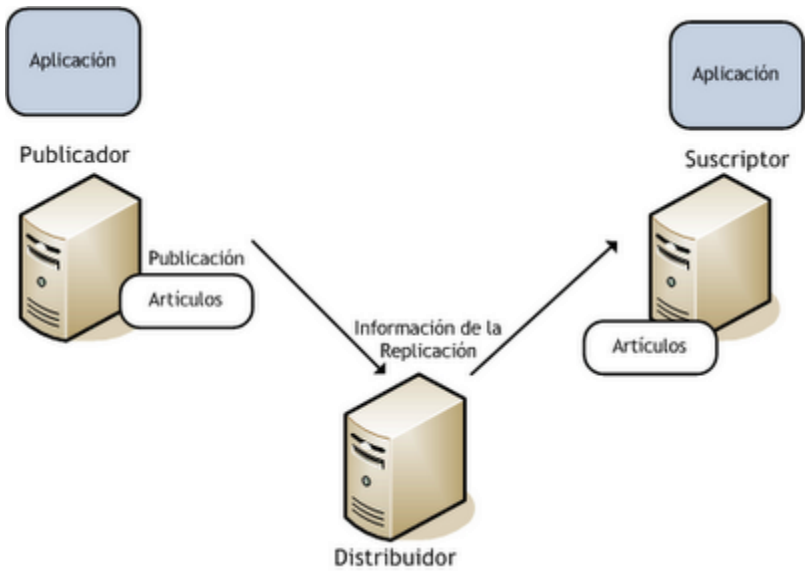
Antes de empezar, vamos a clarificar los conceptos y términos que se utilizan cuando hablamos de la replicación. Los elementos que componen la replicación son los siguientes:

- **Publicador**: es la instancia que pone sus datos a disposición de otras localizaciones mediante la replicación. El Publicador puede tener varias publicaciones configuradas cada una relacionada con un conjuntos lógico de objetos y datos.
- **Distribuidor**: es la base de datos destinada a almacenar la información específica asociada a la replicación de uno o más publicadores. Cada publicador es asociado con una base de datos (conocida como la base de datos de distribución) en el Distribuidor. La base de datos de distribución guarda el estado de la replicación, metadatos y en algunos casos hace de cola de distribución entre el publicador y el suscriptor. En la mayoría de los casos, la misma base de datos actúa como Publicador y Distribuidor. Cuando el

Publicador y el Distribuidor se encuentran en servidores separados, el Distribuidor es conocido como "*Distribuidor Remoto*".

- **Artículo**: un artículo identifica un objeto de base de datos que es incluido en la publicación. Una publicación puede tener varios tipos de artículos: procedimientos almacenados, vistas, tablas y otro tipo de objetos. Cuando las tablas son publicadas, se pueden establecer filtros para restringir los datos y/o columnas que se envían al suscriptor.
- **Publicación**: es una colección de no o más artículos de una base de datos. La agrupación de artículos en una publicación hace más fácil especificar el conjunto de datos asociados en la replicación como una sola unidad
- **Suscripción**: es una petición para que una copia de la publicación sea enviada al suscriptor. La suscripción define qu´r publicación será recibida, cuando y donde. Hay dos tipos de suscripción: de inserción y de extracción
- **Agentes**: son los encargados de gestionar la comunicación y el envío de los datos entre los suscriptores y los publicadores

Una vez aclarados los conceptos, vemos un diagrama del flujo simplificado de los datos y de los elementos que intervienen en una replicación:



5.1.3 Métodos de respaldo de un DBMS

En mySQL existen varios métodos para la realización de un backup y esto se debe principalmente a que mySQL guarda las tablas como archivos y al tipo de tablas que se este manejando (***InnoDB***, ***MyISAM***, ***ISAM***). Así por ejemplo para la presente práctica se utilizó el tipo de tabla InnoDB y el método de backup utilizado es el que funciona con este tipo de tablas.

InnoDB es una de las tecnologías de almacenamiento que utiliza mySQL, es de codigo abierto. Entre sus características principales estan que soporta transacciones con características ACID (***Atomicidad, Consistencia, Aislamiento y Durabilidad***), tiene bloque de registros e integridad referencial (cosa que no maneja ***ISAM***, ni ***myISAM***). Esta última es una de sus características más importantes pues una base de datos sin integridad referencial, es nada mas un conjunto de datos que no denotan infomación.

Este tipo de almacenamiento también ofrece una alta fiabilidad y consistencia. El mismo gestiona el control de los datos y no se lo deja al sistema operativo, una de sus desventajas es que no tiene una buena compresión de datos, por lo que ocupa un poco mas de espacio que ***myISAM***.

5.1.3.1 Elementos y frecuencia de respaldo

Normalmente cuando uno plantea que va a respaldar los datos de su PC a una persona en una compañíauno tiene que definir muy bien cual es la información crítica para la empresa, por ejemplo la música que guarde un empleado en su PC no es crítica para las actividades de la empresa ni lo son las fotos de su última fiesta. En cambio su correo electrónico, proyectos, informes y papeles administrativos si lo suelen ser y tener un respaldo de estos es clave para el funcionamiento de la empresa en caso de cualquier eventualidad. Normalmente la data o información que es respaldada por las empresas es:

- Archivos creados por aplicaciones, como por ejemplo .doc, .odt, .xls, .mdb, .pdf, .ppt entre otros.
- Archivos de correo electrónico
- Directorios telefónicos y de contactos
- Favoritos de los navegadores como Firefox e Internet Explorer
- Base de datos
- Configuraciones de los equipos
- Archivos de CAD, PSD, XCF, etc.
- Imágenes y Fotografías de proyectos
- Configuraciones de servicios

5.1.3.2 Comandos para respaldo de datos

Para hacer un respaldo de una base de datos MySQL desde nuestro consola o mediante comandos shell podemos usar el comando mysqldump como lo ejemplificamos en la siguiente liga.

Comando: mysqldump -u "usuario" -p"contraseña" nombre-de-la-base-de-datos > nombre-del-respaldo.sql

NOTA: Las comillas deben omitirse tanto en el usuario como en la contraseña.

Para restaurar un respaldo de una base de datos MySQL usamos el siguiente comando

Comando: mysql -u "usuario" -p"contraseña" nombre-de-la-base-de-datos < nombre-del-respaldo.sql

NOTA: Al igual que en el ejemplo anterior las comillas deben omitirse tanto en el usuario como en la contraseña.

Para Respaldar o Restaurar una Base de datos remota usamos los mismos comandos que de manera local, con la única diferencia de agregar la opción "-h" con la cual especificaremos el nombre o dirección del host en donde se encuentra nuestra base.

Para Respaldar usamos:

Comando: mysqldump -u "usuario" -p"contraseña" -h"nombre-o-dirección-del-host" nombre-de-la-base-de-datos > nombre-del-respaldo.sql

Para restaurar usamos:

Comando: mysql -u "usuario" -p"contraseña" -h"nombre-o-dirección-del-host" nombre-de-la-base-de-datos < nombre-del-respaldo.sql

5.1.3.3 Métodos de recuperación de un DBMS

La recuperación consiste en tres pasos

principales:

Análisis:

Identifica las páginas sucias y el conjunto de transacciones activas en el momento de la caída y el punto del log apropiado para empezar la operación REHACER

Rehacer: se replican las operaciones del log.

Deshacer: Se recorre el log hacia atrás y se deshacen las transacciones activas en el momento de la caída, o iniciadas después, de las que no se ha encontrado confirmación.

Recuperación en Oracle

Red Log Files: dos o más archivos donde se registra cualquier modificación transaccional de una memoria intermedia de la BD.

Archivos de control: metadatos necesarios para operar en la base de datos, incluyendo información sobre copias de seguridad.

Segmento Rollback: guarda las últimas sentencias realizadas sobre la BD y sabe cuándo se ha confirmado o no una transacción.

En la Recuperación de un fallo: *Recupera los datos con REHACER (Desde Redo Log File). Deshace las transacciones no comprometidas con Deshacer (Desde el segmento de rollback).*

5.1.4 Comandos para recuperación

Cada vez que se ejecuta una tarea de copia de seguridad, CA ARCserve Backup registra la información en la base de datos sobre los equipos, directorios y archivos de los que se ha realizado copia de seguridad y los medios utilizados. Esto permite localizar archivos para cuando sea necesario restaurarlos. El comando de recuperación de base de datos (ca_recoverdb) es una opción de protección propia que permite recuperar una base de datos de CA ARCserve Backup si se ha perdido y se ha realizado copia de seguridad mediante el dominio de CA ARCserve Backup que está utilizando la base de datos.

La utilidad ca_recoverdb sólo se utiliza para recuperar una base de datos de ARCserve (ASDB) en el mismo equipo o dominio de ARCserve en el que se ha realizado la copia de seguridad de ASDB. Si desea realizar la copia de seguridad de ASDB en un equipo y recuperarla en otro (los dos equipos no se encuentran en el mismo dominio de ARCserve), no se puede utilizar este comando. Ante esta situación dispone de dos soluciones:

Solución 1:

Realizar una copia de seguridad de recuperación de desastres desde el equipo A y después recuperarla en el equipo B.

Esta solución necesita que esté instalada la opción de recuperación de desastres (DR, Disaster Recovery) .

Solución 2:

1. Instale CA ARCserve Backup en ambos equipos: equipo A y equipo B.
2. Realice una copia de seguridad de ASDB en el equipo A.
3. Mueva la cinta al equipo B y envíe una tarea combinada para combinar la información de la cinta en CA ARCserve Backup en el quipo B.
4. En el equipo B, abra el Gestor de restauración (opción Restaurar por árbol) y busque la "base de datos de CA ARCserve Backup".
5. Haga clic con el botón derecho del ratón en "base de datos de CA ARCserve Backup" y en el menú desplegable seleccione "Opciones del Agente".
6. Desde el cuadro de diálogo Opciones de restauración del Agente, seleccione las siguientes opciones:

Aplicación forzosa de la restauración en la base de datos o archivos existentes

Utilizar la base de datos de ARCserve actual como ubicación original

Conservar miembros del dominio de ARCServe actual

Para enviar la tarea de restauración.

5.1.4.1 Ventajas y Desventajas de cada método

Este artículo describe varias soluciones para recuperar datos de una base de datos de Microsoft SQL Server, si se produce un desastre. En este artículo también se describe las ventajas y las desventajas de cada solución.

Recuperación ante desastres es un proceso que puede utilizar para recuperar datos y sistemas de información si se produce un desastre.

Algunos ejemplos de desastres incluyen natural o un desastre artificial como un

incendio o un desastre técnico como una falla en dos discos de una matriz redundante matriz de independiente discos (RAID) 5.

Planificación de recuperación ante desastres es el trabajo que está dedicado a la preparación de todas las acciones que deben ocurrir en respuesta a un desastre. La planificación incluye la selección de una estrategia para ayudar a recuperar datos valiosos. La selección de la estrategia de recuperación ante desastres apropiada depende de los requerimientos del negocio.

Nota: Las soluciones que se describen en este artículo sólo proporcionan descripciones generales de las tecnologías que se pueden utilizar. Estas descripciones generales son para comparar los distintos métodos de recuperación de desastres y los planes de recuperación ante desastres. Antes de decidir en qué desastres es el mejor solución de recuperación, asegúrese de que examina cada una de las soluciones de recuperación ante desastres sugeridos con más detalle. Después de tratar cada solución de recuperación ante desastres, este artículo contiene vínculos donde encontrará información adicional acerca de esa solución.

5.1.4.2 Aplicación de cada método

Almacenan el estado de la estructura física de la BD.

• Contienen:

Nombre de la BD

Localización de los ficheros de datos y redo log

Nombre de los Tablespaces

Número de secuencia de log actual

Log histórico

Información de las copias de seguridad (backup) w etc.

• Es necesario para montar, abrir y mantener la BD

• Guían la recuperación. Sin este fichero la BD no podrá ser montada y la recuperación sería dificultosa.

• Se recomienda como mínimo dos ficheros de control en discos diferentes

5.2 Migración de la Base de Datos

La migración de bases de datos es generalmente una tarea compleja que no sólo supone transferir datos entre tipos de almacenaje y formatos de un servidor de base de datos a otro; sino que también supone reescribir sentencias SQL o incluso procedimientos (SPL) de lógica de negocio.

En comparación con los esquemas estándares de migración a mano, ofrecemos una potente gama de herramientas desarrolladas de probada eficacia en complejos módulos de bases de datos relacionales. Estas herramientas y nuestros especialistas pueden asegurar que las transiciones de las bases de datos se realicen perfectamente, independientemente de la naturaleza del sistema.

Desde la experiencia, estamos familiarizados con la complejidad, el coste que supone una larga migración de bases de datos y los problemas que aparecen durante el proceso cuando se emplean métodos inapropiados; ya que siempre comprobamos con los clientes potenciales que el uso de nuestras herramientas y métodos pueda ofrecer una ventaja significativa.

HERRAMIENTAS DE MIGRACIÓN

En comparación con la consultoría estándar de migraciones, la cual puede ofrecer poco más que soporte a la base de datos, nosotros tenemos gran experiencia en escribir grandes aplicaciones para empresas en sintaxis de la base de datos nativa y cross. Además, enseñamos a los equipos de las empresas una metodología y les proporcionamos una potente gama de herramientas para reducir costes y optimizar el proceso de migración.

5.3 Monitoreo y Auditoría de la Base de Datos

5.3.1 Monitoreo

Auditoria: Es el proceso que permite medir, asegurar, demostrar, monitorear y registrar los accesos a la información almacenada en las bases de datos incluyendo la capacidad de determinar:

- *Quién accede a los datos*
- *Cuándo se accedió a los datos*
- *Desde qué tipo de dispositivo/aplicación*
- *Desde que ubicación en la Red*
- *Cuál fue la sentencia SQL ejecutada*
- *Cuál fue el efecto del acceso a la base de datos*

Objetivos Generales de la Auditoría de BD

Disponer de mecanismos que permitan tener trazas de auditoría completas y automáticas relacionadas con el acceso a las bases de datos incluyendo la capacidad de generar alertas con el objetivo de:

- *Mitigar los riesgos asociados con el manejo inadecuado de los datos.*
- *Apoyar el cumplimiento regulatorio.*
- *Satisfacer los requerimientos de los auditores.*
- *Evitar acciones criminales.*
- *Evitar multas por incumplimiento.*

5.3.1.1 Monitoreo general de un DBMS

La elección de un buen manejador de base de datos es de vital importancia ya que puede llegar a ser una inversión tanto en hardware como en

software muy cuantioso pero no solo eso, además va a determinar el centro de información de la empresa.

Los sistemas orientados a los datos se caracterizan porque los datos no son de una aplicación sino de una Organización entera que los va a utilizar; se integran las aplicaciones, se diferencian las estructuras lógicas y físicas. El concepto de relación cobra importancia. Originalmente las aplicaciones cubrían necesidades muy específicas de procesamiento, se centraban en una tarea específica.

5.3.1.2 Monitoreo de espacio en disco

Uno de los principales indicadores que se tiene que tomar en cuenta como DBA es el espacio disponible en disco. No es problema cuando se tiene un server o 2 para monitorear, sin embargo cuando hay una cantidad considerable automatizar un proceso que lo haga es lo mejor. Dentro de SQL Server (7,2000,2005) hay un procedimiento no documentado que nos puede ayudar a cumplir este cometido.

El procedimiento es XP_FIXEDDRIVES, no lleva parámetros ni nada y nos regresan todos los discos a los que tiene acceso SQL Server y su espacio disponible en Megabytes.
Si esta en cluster mostrara todos los discos aunque los discos no estén en el mismo grupo que la instancia, lo que puede llegar a confundir.

Dejo a consideración de cada quien como utilizarlo, ya sea mandando un mail con el resultado u opciones más complejas como el revisar un porcentaje y en base a eso tomar una acción.

5.3.1.3 Monitoreo de logs

Las revisiones deben realizarse sobre el archivo de alerta de ORACLE (alert.log) y sobre los archivos de rastreo de procesos de background y de usuarios para identificar errores que se presenten a nivel de base de datos o de sistema operativo.

Los archivos de alerta útiles para el diagnóstico de información que contiene ORACLE y que se utilizan para la detección de errores en la base de datos son:

Archivo de Log´s de alerta (alert.log)

El Alert Log registra errores en forma cronológica, provenientes de la operación diaria de la Base de Datos. La ubicación actual del archivo es la ubicación por defecto establecida por ORACLE y se verifica mediante el parámetro BACKGROUND_DUMP_DEST del archivo init.ora:

BACKGROUND_DUMP_DEST = E:\U01\ORACLE\UCBL\ADMIN\bdump.

La revisión de este archivo en forma periódica permite detectar errores internos (ORA-600) y errores de corrupción de bloques (ORA-1578). Adicionalmente, permite monitorear las operaciones de la base de datos (CREATE DATABASE, STARTUP, SHITDOWN, ARCHIVE LOG y RECOVER) y ver los parámetros que no se muestran por defecto en la inicialización.

Archivos de rastreo de procesos de Background

Los archivos de rastreo de procesos de Background se generan cuando un proceso de background (SMON, PMON, DBWn, etc.) emite un error. Estos archivos se almacenan en BACKGROUND_DUMP_DEST = E:\U01\ORACLE\UCBL\ADMIN\bdump.

Archivos de rastreo de usuarios

Los archivos de rastreo de usuarios (user trace files) se crean a través de procesos de servidor cuando se generan errores o cuando se solicita el rastreo por el usuario o a nivel de parámetros de la base de datos.

Su ubicación actual definida por el parámetro USER_DUMP_DEST y actualmente es:

E:\U01\ORACLE\UCBL\ADMIN\udump.

Las normas de revisión de los archivos mencionados se definen en el documento Procedimientos de Administración de Base de Datos.

El principal riesgo que se menciona en las observaciones es la posibilidad que se realicen operaciones no autorizadas y que éstas no sean identificadas en la base de datos; sin embargo los archivos de log´s de usuarios no permiten identificar con facilidad estas operaciones y en todo caso requieren de una gran cantidad de tiempo de revisión y espacio de almacenamiento. Por este motivo, se utilizan tablas de auditoria para todos los sistemas y para aquellas tablas relevantes de cada uno, las tablas de auditoria tienen las siguientes características:

- Almacenan datos obligatorios (transacción, fecha y usuario) y datos relacionados con la tabla a la que hacen el monitoreo.
- Los registros a las tablas de auditoria se activan mediante un disparador cada vez que se realizan cambios en la tabla base.
- Se consultan estas tablas cuando se quiere identificar una transacción, un usuario o una fecha de transacción.

El contenido de estas tablas permite mantener un registro constante sobre las operaciones que se realizan en la base de datos y las mismas pueden ser consultadas en cualquier momento.

5.3.1.4 Monitoreo de Memoria compartida

PGA DE ORACLE (ÁREA GLOBAL DE PROGRAMA)

Un PGA es una región de memoria que contiene datos e información de control para un proceso de servidor. Es la memoria no compartida creada por la base de datos Oracle cuando un proceso de servidor se ha iniciado. El acceso a la PGA es exclusivo para el proceso del servidor. Hay un PGA para cada proceso de servidor. Procesos en segundo plano también se asignan sus propios PGA. La memoria total utilizada por todos los PGAs individuales se conoce como el ejemplo total de memoria PGA, y la recogida de PGAs individuales se refiere como el ejemplo total de la PGA, o simplemente instancia de la PGA. Puede utilizar los parámetros de inicialización de base de datos para definir el tamaño de la instancia de la PGA, no PGA individuales.

El PGA puede ser crítico para el rendimiento, especialmente si la aplicación está haciendo un gran número de clases. Operaciones de ordenación se produce si utiliza ORDER BY y GROUP BY comandos en las sentencias SQL.

SGA de oracle (Sistema de Área Global)

Es un conjunto de áreas de memoria compartida que se dedican a un Oráculo "instancia" (un ejemplo es los programas de bases de datos y la memoria RAM).

Sirve para facilitar la transferencia de información entre usuarios y también almacena la información estructural de la BD más frecuentemente requerida.

En los sistemas de bases de datos desarrollados por la Corporación Oracle , el área global del sistema (SGA) forma parte de la memoria RAM compartida por todos los procesos que pertenecen a una sola base de

datos Oracle ejemplo. El SGA contiene toda la información necesaria para la operación de la instancia.

La SGA se divide en varias partes:

Buffers de BD, Database Buffer Cache

Es el caché que almacena los bloques de datos leidos de los segmentos de datos de la BD, tales como tablas, índices y clusters. Los bloques modificados se llamas bloques sucios. El tamaño de buffer caché se fija por el parámetro DB_BLOCK_BUFFERS del fichero init.ora.

Plan de ejecución de la sentencia SQL.

Texto de la sentencia.

Lista de objetos referenciados.

Comprobar si la sentencia se encuentra en el área compartida.

Comprobar si los objetos referenciados son los mismos.

Comprobar si el usuario tiene acceso a los objetos referenciados.

Como el tamaño del buffer suele ser pequeño para almacenar todos los bloques de datos leidos, su gestión se hace mediante el algoritmo LRU.

2. Buffer Redo Log

Los registros Redo describen los cámbios realizados en la BD y son escritos en los ficheros redo log para que puedan ser utilizados en las operaciones de recuperación hacia adelante, roll-forward, durante las recuperaciones de la BD. Pero antes de ser escritos en los ficheros redo log son escritos en un caché de la SGA llamado redo log buffer. El servidor escribe periódicamente los registros redo log en los ficheros redo log. El tamaño del buffer redo log se fija por el parámetro LOG_BUFFER.

3. Área de SQL Compartido, Shared SQL Pool

En esta zona se encuentran las sentencias SQL que han sido analizadas. El analisis sintáctico de las sentencias SQL lleva su tiempo y Oracle mantiene las estructuras asociadas a cada sentencia SQL analizada durante el tiempo que pueda para ver si puede reutilizarlas. Antes de analizar una sentencia SQL, Oracle mira a ver si encuentra otra sentencia exactamente igual en la zona de SQL compartido. Si es así, no la analiza y pasa directamente a ejecutar la que mantinene en memoria. De esta manera se premia la uniformidad en la programación de las aplicaciones. La igualdad se entiende que es lexicografica, espacios en blanco y variables incluidas. El contenido de la zona de SQL compartido es:

Los pasos de procesamiento de cada petición de análisis de una sentencia SQL son:

Si no, la sentencia es nueva, se analiza y los datos de análisis se almacenan en la zona de SQL compartida.

También se almacena en la zona de SQL compartido el caché del diccionario. La información sobre los objetos de la BD se encuentra almacenada en las tablas del diccionario. Cuando esta información se necesita, se leen las tablas del diccionario y su información se guarda en el caché del diccionario de la SGA. Este caché también se administra mediante el algoritmo LRU. El tamaño del caché está gestionado internamente por el servidor, pero es parte del shared pool, cuyo manaño viene determinado por el parámetro SHARED_POOL_SIZE.

5.3.1.5 Monitoreo de Base de Datos

Mediante la auditoría de bases de datos se evaluará:

- Definición de estructuras físicas y lógicas de las bases de datos.

- Control de carga y mantenimiento de las bases de datos.
- Integridad de los datos y protección de accesos.
- Estándares para análisis y programación en el uso de bases de datos.
- Procedimientos de respaldo y de recuperación de datos.

Aspectos Claves

- No se debe comprometer el desempeño de las bases de datos
- Soportar diferentes esquemas de auditoría.
- Se debe tomar en cuenta el tamaño de las bases de datos a auditar y los posibles SLA establecidos.

Segregación de funciones

El sistema de auditoría de base de datos no puede ser administrado por los DBA del área de IT.

Proveer valor a la operación del negocio

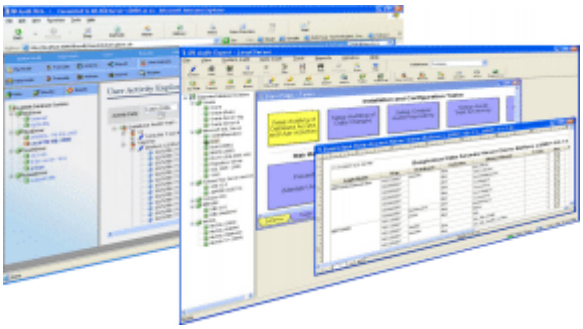
- Información para auditoría y seguridad.
- Información para apoyar la toma de decisiones de la organización.
- Información para mejorar el desempeño de la organización.

Auditoría completa y extensiva

- Cubrir gran cantidad de manejadores de bases de datos.
- Estandarizar los reportes y reglas de auditoría.

DB Audit Expert es un Auditor Multiplataforma y solución de Monitoreo Proactivo para Bases de Datos - Oracle, SQL Server, Sybase, MySQL y DB2

Vivimos en una economía de información; las empresas están hoy en día dependientes de las tecnologías de bases de datos que hacen funcionar su negocio. Con datos activos de misión crítica son almacenados en bases de datos de SQL, entre otras y es cuando surgen las preguntas: “Quien está teniendo acceso a nuestra base de datos, quien y cuando se hacen los cambios?” la mayor parte de la información financiera de una organización también se almacena y se mantiene en las bases de datos, el DBA trabaja para compañías públicas en donde se requiere que proporcione un rastro de intervención exacto, auditorias inmutable de todos los accesos de bases de datos y cambios en permisos de seguridad.



Componentes de DB Audit Expert Empresarial

5.3.1.6 Monitoreo de modos de operación

Definición y diseño de monitoreo.

Una vez identificados los riesgos, se puede realizar el modelo conceptual del monitoreo, a partir de las debilidades identificadas y contemplando los controles existentes en el proceso.

Posteriormente, se diseña el monitoreo en sí, analizando el procesamiento y la arquitectura de la información, así como el registro informático de las

operaciones. A partir de todo lo anterior, se desarrollan los procedimientos automatizados, determinando las consultas a realizar y los parámetros a cumplir.

Obtención de herramientas informáticas. Una vez concluido el diseño, se puede identificar la herramienta a utilizar de manera individual o combinada.

5.3.1.7 Monitoreo de espacios espejeados

La optimización en MySQL pasa por tres componentes, a saber:

- Optimización del servidor MySQL
- Optimización de la base de datos
- Optimización de las consultas

Optimización de la configuración del servidor MySQL

La optimización del servidor puede incluir una multitud de enfoques y métodos, lo que intentaremos presentar en lo que sigue es una introducción a los enfoques de base, a saber:

- Compilación del servidor
- Afinamiento de los parámetros del servidor
- Afinamiento de otros parámetros

Para hacer una buena optimización, es necesario proceder con una metodología empírica a saber hacer las modificaciones una por una y probar cada vez la reacción del sistema para ver el resultado. Una medida

del rendimiento antes y después de haber efectuado la optimización permite ver si el sistema ha sido optimizado o no.

Compilación del servidor

Es recomendado utilizar la versión del código fuente del servidor MySQL y compilarla teniendo en cuenta los diferentes parámetros del sistema a saber el conjunto de caracteres a utilizar, el microprocesador sobre el que va a correr y utilizar un compilador adaptado (por ejemplo: pgcc para los microprocesadores Pentium).

Afinamiento de los parámetros del servidor

Es posible optimizar el funcionamiento de MySQL cambiando los valores de los parámetros del servidor.

Como recordarás para mostrar los parámetros se debe utilizar el comando: show variables;

Para ver el efecto de los parámetros sobre el servidor es necesario ejecutar el comando:

show status;

Existen numerosas herramientas de monitoreo que permiten ver los efectos de los cambios efectuados en los parámetros en el servidor MySQL, por ejemplo Mytop equivalente al comando **top** de Linux.
El fichero **my.cnf** contiene todos los parámetros que deben ser optimizados.

Inicialmente, es posible comenzar con los parámetros que gestionan la memoria. Se debe tener en cuenta que cuanta más memoria disponga el servidor, más rápido será, sin embargo, hay que asegurarse de que la memoria esté disponible.
MySQL contiene un conjunto de buffers y cachés internos, en el que es posible configurar el espacio asignado a cada uno a partir de las variables del fichero **my.cnf**. Las dos variables más importantes son **key_buffer_size** y **table_cache** ya que son compartidas por todos los threads que corren sobre el servidor e influyen de manera considerable en el rendimiento.
Un ejemplo de variables:

- key_buffer_size: memoria utilizada para las copias de seguridad de los índices MyISAM.
- table_cache: numero de tablas que pueden ser abiertas simultáneamente.
- read_buffer_size: memoria utilizada para la copia de respaldo de los datos salidos de los full scan de las tablas.
- sort_buffer: memoria utilizada para la copia de respaldo de los datos de las tablas que serán ordenadas con un **ORDER BY**

Afinamiento de otros parámetros

El servidor MySQL obtiene un funcionamiento óptimo en SOLARIS, sin embargo, es posible optimizarlo en otros SO para aproximarse a su rendimiento ideal.
El uso de RAID-RAID 0 es recomendado para la optimización de las operaciones de lectura escritura. Así como el uso de discos SCSI en vez de IDE.
El uso de redes rápidas optimiza el tiempo de respuesta y optimiza la comunicación entre cliente/servidor y amo/esclavo para la replicación.

Optimización de la base de datos

Generalmente para la optimización de las bases de datos lo recomendado es hacer uso de las buenas prácticas y las metodologías de concepción de base de datos que permitan implementar esquemas de bases de datos eficaces y normalizados. Sin embargo para ello es necesario:

- Saber lo que está lento en las bases de datos
- Elegir la metodología correcta
- Utilizar índices
- Utilizar **OPTIMIZE TABLE**

Qué es lo que ralentiza las bases de datos

Generalmente, un cierto número de factores son la causa de la lentitud de las bases de datos. Entre los más frecuentes:

- Insuficiente numero de índices: La primera causa de la lentitud es el uso de tablas sin índices o sin índices en las columnas relativas a las búsquedas. Esto no quiere decir que todas las tablas deben tener índices, sino que hay que estudiar bien las necesidades de indexación.
- Uso excesivo de índices: para optimizar las consultas y búsquedas, los índices son la solución, sin embargo, el aumento de índices afecta el rendimiento en lo relativo a las actualizaciones. En la actualización de una tabla, las operaciones de inserción, modificación y eliminación repercuten generalmente sobre los índices.

- Uso de privilegios en las tablas y columnas de las tablas: en cada acceso MySQL debe verificar los derechos sobre las tablas y las columnas de las tablas lo que ralentiza considerablemente el rendimiento.
- No hacer la elección correcta en la concepción de la base de datos.

Modelización de la base de datos

El uso de las buenas prácticas de modelización y concepción de bases de datos así como la elección de la metodología apropiada permite implementar bases de datos eficaces.
Es necesario tener en cuenta un cierto número de consideraciones:

- Apropiada elección de los tipos de campos: siempre procurar elegir las variables más adaptadas a las necesidades (por ejemplo para almacenar un numero con no más de 10 dígitos, lo mejor es utilizar un tipo **TINYINT**). El uso de campos de menor tamaño permite cargar en memoria más columnas.
- Uso de campos de longitud fija: el uso de longitudes predeterminadas permite optimizar el acceso a las columnas ya que sus posiciones son predefinidas. Esto implica disminuir el uso de VARCHAR, TEXT y BLOB (para TEXT y BLOB, se recomienda romper la normalización del esquema de la base de datos y hacer una copia de respaldo de estos campos en otras tablas).
- Aumentar el uso de las restricciones **NON NULL** cuando sea posible para optimizar el espacio de almacenamiento.
- Elegir el tipo correcto para las tablas: MySQL permite tener en un mismo esquema tablas de diferente tipo.
- Hacer una buena indexación de las tablas.

Utilizar los índices

Un índice es una tabla de búsqueda que nos permite encontrar rápidamente líneas en una tabla. El índice permite determinar la posición del registro buscado en una tabla.
Si una tabla no tiene índice, todos los registros serian recorridos durante la búsqueda.
Los índices en MySQL son almacenados como de b-trees (árboles binarios), que representa una estructura de datos fácil y rápida de recorrer.
El índice puede incluir una o varias columnas, el índice será llamado durante una búsqueda hecha sobre las columnas indexadas.
En MySQL, la indexación es automática en las tablas con campos teniendo las restricciones **PRIMARY, KEY, UNIQUE**.
La idea principal a tener en cuenta es que si una búsqueda es frecuente y ésta incluye una o varias columnas, será necesario crear el índice correspondiente para optimizar el tiempo de respuesta vía el comando CREATE INDEX

Uso del comando OPTIMIZE TABLE

Equivalente a la defragmentación del disco duro, el comando **OPTIMIZE TABLE** permite defragmentar las tablas.

Optimización de las consultas

MySQL permite analizar las consultas y conocer el tiempo y plan de ejecución. Esta información permite comprender lo que hace que las consultas sean lentas y optimizar la ejecución de éstas.

Detectar las consultas lentas

Para detectar las consultas lentas es posible:

- 1. observar las consultas lentas durante su ejecución y los tiempos de respuesta anormales.
- 2. hacer un benchmark: testear las aplicaciones para ver qué componentes son los más lentos.
- 3. verificar el **Slow query log**: es posible activar esta opción en MySQL configurando la variable **--log-slow-queries**

Una vez detectadas las consultas lentas, la ejecución del comando **EXPLAIN** permite comprender la ejecución y por lo tanto conocer o intervenir para optimizar.

5.3.2 Auditoría

5.3.2.1 Habilitación y deshabilitar el modo de auditoría

- Mediante la auditoría se intenta monitorizar y registrar acciones en la base de datos con el fin de:
- Investigar actividades maliciosas.
- Detectar privilegios incorrectamente otorgados a usuarios.
- Recoger datos sobre actividades concretas.
- Detectar problemas con la implementación de políticas de seguridad.