

Gestión de la seguridad de los datos.

Caso práctico



[Jonny Goldstein \(CC BY\)](#)

Juan ha estado supervisando el trabajo de **Vindio, Noiba y Naroba**, y parece que ya está a punto de concluir. La base de datos del taller mecánico está totalmente implantada y las expectativas que los socios del taller tenían puestas en su trabajo se han cumplido. En este momento sería imposible prescindir de la operatividad que proporciona el **SGBD**, no sólo con relación a la actividad cotidiana: registro de ingresos y salidas, reparaciones, facturación, control de recambios, personal, etc.; sino que les permite analizar los resultados obtenidos para obtener estadísticas y con ello hacer previsiones y planificar estrategias futuras.

Precisamente por la importancia que tiene la base de datos en la gestión del taller, **Juan les plantea ahora una cuestión muy importante ¿Qué pasaría si por alguna causa imprevista la base de datos se daña o se destruye?** En este momento eso sería un desastre para la gestión del negocio.

Por ello, **Juan** les propone que para completar su trabajo deben establecer una **política de copias de seguridad** de la base de datos, así como procedimientos adecuados de recuperación de los datos ante posibles fallos tanto de hardware como de software, o ante cualquier causa externa que se pueda producir.

¿Cómo podríamos saber a qué clientes tenemos que llamar, qué facturas no se han cobrado, cuántas horas ha trabajado cada empleado, si un fallo en el disco duro que contiene la base de datos nos impide acceder a éstos?

En **BK Sistemas Informáticos** lo tienen claro, es **importante que los datos estén protegidos y que puedan ser reconstruidos** ante cualquier situación imprevista que se pueda presentar.

En esta unidad aprenderás a gestionar la seguridad de los datos almacenados en tu base de datos. Éste es un tema muy importante, pues imagina que por alguna causa imprevista la base de datos se daña o se destruye. ¿Qué podemos hacer ante este hecho? Lo ideal sería poder restaurarla sin ninguna pérdida de datos, ¿verdad?

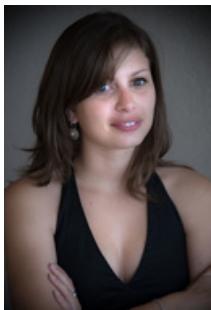
Esto es precisamente lo que vamos a tratar en esta unidad, cómo establecer una política de copias de seguridad de la base de datos, así como procedimientos adecuados de recuperación de los datos ante posibles fallos tanto de hardware como de software, o ante cualquier causa externa que se pueda producir. Verás cómo utilizar las herramientas y comandos que permiten realizar el mantenimiento de las tablas de una base de datos, y la importancia de los ficheros de registro o de log.



[Ministerio de Educación y Formación Profesional \(Dominio público\)](#)

1.- Principales fallos en una base de datos.

Caso práctico



[Alain Bachellier \(CC BY-NC-SA\)](#)

Antes de establecer las medidas y políticas de seguridad que debe tomar para evitar cualquier daño en la base de datos, **Noiba, Vindio y Naroba empiezan por considerar cuáles son los posibles fallos que se pueden producir**. Eso les ayudará a establecer una estrategia para evitar que se produzcan y, si eso no es posible, al menos poder garantizar que los datos de TalleresFaber puedan ser recuperados en su totalidad.

Noiba no puede evitar que la base de datos sea destruida si ocurre, por ejemplo, un incendio en la oficina, pero podrá restaurar los datos si ha tenido la precaución de mantener una copia de seguridad ubicada en otro lugar.

Del mismo modo que se han tomado medidas de seguridad en las instalaciones del taller con relación a la existencia de maquinaria eléctrica, almacenamiento y manipulación materiales inflamables, etc. también es necesario proteger un bien muy valioso para la empresa: los datos.

Las funciones de mantenimiento, respaldo y recuperación constituyen un componente muy importante de los SGBD actuales. Algunos SGBD proporcionan herramientas para que se puedan programar respaldos automáticos en dispositivos de almacenamiento secundario que eviten desastres ante problemas como:

Falta de suministro eléctrico.

Algún problema de hardware: con el disco duro, sectores dañados, chips de memoria, procesador, etc.

Fallos de software: imputables al sistema operativo, al software del SGBD, a los programas de aplicación o virus.

Problemas con el sistema de ficheros.

Factores externos como terremotos, inundaciones, incendios, etc.

El SGBD debe garantizar que los datos puedan ser recuperados en su totalidad en caso de una pérdida física o de una pérdida de la integridad de la base de datos.

La pérdida de datos puede ser total o parcial.

Una pérdida **parcial** puede ser provocada por la pérdida física de una parte de la base de datos, o por la pérdida de integridad de una parte de la misma.

Una pérdida **total** puede ocurrir por una pérdida de integridad de toda la base de datos, pero que ésta físicamente continúe existiendo o que se haya perdido por completo.

Cuando en un servidor con una base de datos en funcionamiento se produce alguno de los problemas anteriores será necesario disponer de una copia de seguridad que nos permita recuperar los datos.

El respaldo de la base de datos debe guardarse en lugar seguro, generalmente otro edificio para que esté protegido contra catástrofes como incendios, robos, inundaciones y cualquier otro peligro potencial.

La pérdida de datos puede ser total o parcial, del mismo modo la copia de seguridad puede ser completa o incremental, es decir, a partir de los datos desde la última copia de seguridad.

En algunos casos podremos recuperar los datos de las tablas mediante el uso de utilidades disponibles en cualquier SGBD, así como desde sentencias SQL para el mantenimiento de tablas.

Reflexiona

Los datos tienen un valor y deben ser tratados como cualquier otro elemento de la empresa. Cuando los datos no están disponibles la empresa puede sufrir pérdidas y dificultades de gestión muy importantes. Por eso es muy importante planificar, organizar y probar procedimientos de recuperación que garanticen la seguridad y la integridad de la base de datos.

1.1.- Elementos para la recuperación de fallos.

Cuando administramos una base de datos siempre es deseable no encontrarnos con tablas corruptas o destruidas, pero si esto ocurre es necesario minimizar los riesgos estableciendo una estrategia para:

Recuperar datos ante caídas del sistema: debemos conocer cómo **restaurar y reparar tablas, recuperar archivos de respaldo y utilizar los registros de actualización** para recuperar los cambios que se hayan hecho desde la última copia de seguridad.

Minimizar esas caídas con un mantenimiento preventivo: debemos **establecer un programa de mantenimiento** para prevenir las posibilidades de daños en la base de datos así como **realizar copias de seguridad**.

El conjunto de actividades de administración que pretenden garantizar la disponibilidad en caso de desastres incluyen: la planificación, organización, pruebas y procedimientos de recuperación y deben incluir:

Respaldo periódico de datos y aplicaciones: el SGBD incorpora herramientas para realizar diferentes tipos de recuperación: total o incremental.

Respaldo **total**: produce una copia completa de toda la base de datos.

Respaldo **incremental**: produce un respaldo de todos los datos desde la última copia.

Respaldo **concurrente**: ocurre mientras el usuario está trabajando en la base de datos.

Identificar el respaldo de forma apropiada: descripción detallada y fecha. También es importante evitar que ocupen demasiado mediante técnicas de comprensión. El medio más común es la cinta.

Almacenamiento de la copia de respaldo en sitio seguro: Cada copia de respaldo debe guardarse en un lugar diferente, dentro y fuera de la empresa, convenientemente preparados, a prueba de incendios y controlando la humedad y temperatura.

Existen empresas que ofrecen el servicio de copia de seguridad remota, periódica, automática y segura.

Protección física de hardware y software: uso de instalaciones adecuadas, con aire acondicionado, protección contra incendios, un SGBD de respaldo, etc.

Control del acceso del personal al software mediante contraseñas y sistemas de privilegios multinivel.

Cobertura de seguro para los datos en la base de datos para contar con protección financiera.



[Everaldo Coelho \(YellowIcon\)](#)
[\(GNU/GPL\)](#)

Autoevaluación

Una forma de evitar completamente la pérdida de datos en un SGBD es establecer un sistema de discos espejo en el mismo servidor. ¿Verdadero o falso?

- Verdadero Falso

Falso

De esa forma podemos evitar la pérdida por problemas en el almacenamiento secundario pero es necesario almacenar una copia de seguridad en un lugar distinto y seguro.

1.2.- Herramientas del SGBD para la recuperación de fallos.

En los siguientes apartados desarrollaremos las distintas herramientas que todo SGBD tiene disponibles para recuperar los datos ante cualquier fallo del sistema. En nuestro caso nos referiremos a MySQL, pero todas las bases de datos presentan este tipo de utilidades.

En el caso de MySQL tenemos:



Everaldo Coelho and
YellowIcon (GNU/GPL)

1.- **Herramientas relativas al mantenimiento y reparación de tablas.** Cuando se produce una posible pérdida de datos en una tabla, la primera medida que debemos tomar es intentar reparar esa tabla. Para ello en cada SGBD disponemos de utilidades y en el propio lenguaje SQL existen también sentencias con esa finalidad.

En esta fase es muy importante diferenciar de qué tipo de tablas hablamos (InnoDB, MyISAM, BDB, etc.), ya que no todas las utilidades funcionan con todos los tipos de tabla. Para la recuperación de tablas disponemos de:

1.1.- Utilidades como myisamchk.

1.2.- Sentencias SQL como ANALYZE TABLE, REPAIR TABLE, CHECK TABLE.

2.- **Ficheros de registro de operaciones:** ficheros de log. Si activamos esta opción dispondremos de:

2.1.- Un fichero que registra cada consulta que se haga a la base de datos.

2.2.- Un fichero que registra cada error que se produzca en el servidor.

2.3.- Un fichero que registra cada consulta que actualice datos.

Estos ficheros nos van a resultar muy útiles cuando queramos saber qué ha pasado, cuando se produce un error, y sobre todo para recuperar todas las acciones que se lleven a cabo a partir de la última copia de seguridad. Para ver el contenido del log binario disponemos de la herramienta mysqlbinlog.

3.- **Herramientas para realizar y restaurar copias de seguridad.** Bien porque el problema que se nos presenta ha causado un pérdida de datos irrecuperable, o porque no ha sido posible la recuperación con las herramientas de reparación de tablas, necesitaremos recurrir a la copia de respaldo. Establecer una estrategia adecuada para elaborar respaldos periódicos de la base de datos es la principal y más barata medida de seguridad para los datos. Algunas de estas herramientas son:

3.1.- Mysqldump.

3.2.- Mysqlhotcopy.

3.3.- Herramientas gráficas disponibles en aplicaciones como **PhpMyAdmin**, **MySQL Workbench**, entre otras.

4.- **Herramientas para migración de bases de datos.**

Existen múltiples aplicaciones gráficas para exportar bases de datos de un servidor a otro, tanto a MySQL como a otros SGBD. En versiones anteriores de MySQL encontrábamos la herramienta **MySQL Migration toolkit**, pero ya no se distribuye con MySQL. Actualmente se incluye la herramienta de migración en MySQL Workbench.

Autoevaluación

Empareja las herramientas con su funcionalidad:

Ejercicio de relacionar

Herramienta	Relación	Funcionalidad
Mysqldump.	O	1. Analizar y reparar una tabla MyISAM.
MySQL Migration toolkit.	O	2. Realizar copias de seguridad.
Mysqlbinlog.	O	3. Editar el registro binario.
Myisamchk.	O	4. Importar una tabla de Access a MySQL.

Enviar

MySQL dispone de múltiples herramientas para garantizar la seguridad de los datos.

2.- Los ficheros de registro log.

Caso práctico



Naroba ha observado que hay algunas consultas que ralentizan algo el sistema y le pregunta a **Noiba y Vindio** cual sería una forma sencilla de poder conocer aquellas consultas que tardan en ejecutarse más tiempo del esperado y que por tanto pueden crear un cuello de botella en el sistema. Noiba le recuerda, que en el módulo de Gestión de Bases de Datos estudiaron el fichero de log denominado de consultas lentas. Este fichero se puede activar en el SGBD MySQL y así registrar todas las consultas que ralentizan el sistema. De hecho, si activan ese fichero de log, ya tendrán activados todos ficheros de log que proporciona MySQL en la versión actual con la que están trabajando.

Alain Bachellier (CC BY-NC-SA)

Los archivos de registro de MySQL son muy importantes para la recuperación de datos en caso de fallos. En un SGBD es importante conocer:

La localización del directorio de datos (**datadir**): para conocer dónde está situado, recuerda que podemos ejecutar el siguiente comando:

```
SHOW VARIABLES LIKE 'DATADIR';
```

Dicho directorio aparece indicado en el fichero de configuración de MySQL (**my.ini** o **my.cnf**). En el directorio de datos se encuentran todas las bases de datos y tablas organizadas en forma de árbol. Cada base de datos corresponde a un directorio dentro de data. Las tablas se almacenan como archivos dentro de ese directorio de la base de datos, tal como vimos en una unidad anterior.

El registro de errores (**Error Log**): contiene información sobre cuándo se inicia y para el servidor, y si ha ocurrido algún error crítico mientras se estaba ejecutando. Por defecto este archivo se encuentra en el directorio **data** y se denomina **NombreHost.err.** Por defecto, el registro de errores está activado.

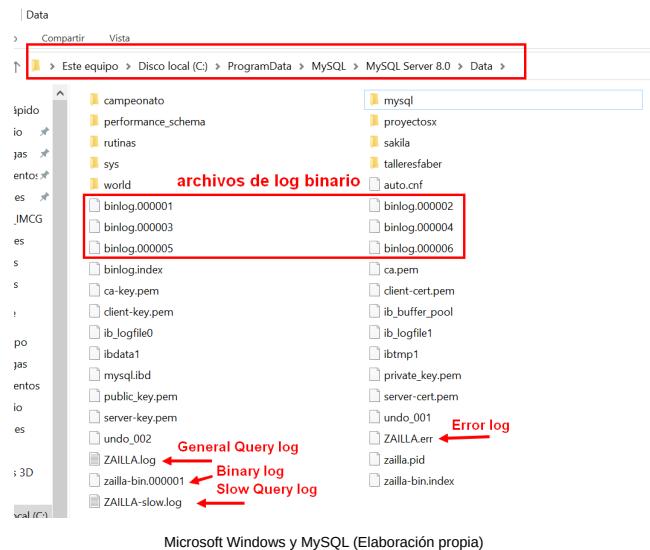
El registro general de consultas (**General Query Log**): registra todas las conexiones y sentencias (incluidas las **SELECT**) en un archivo en el orden en que las recibe. El archivo general de consultas por defecto se denomina **NombreHost.log**. Por defecto, este fichero de log está deshabilitado. Para activarlo debemos acceder al fichero de configuración **my.ini** y establecer el parámetro **general-log** a 1. También es posible activarlo desde los distintos clientes gráficos de una manera mas sencilla. Posteriormente, como siempre que cambiamos algún parámetro en **my.ini**, es necesario reiniciar el servidor

El registro binario (**Binary Log**): contiene todos los eventos que describen los cambios que se producen en la base de datos. Es decir, contiene todas las sentencias que han actualizado datos o podrían haberlo hecho y cuánto han tardado. **Sólo recoge las sentencias que han actualizado datos (INSERT, UPDATE, DELETE)**, si queremos registrar todas tendremos que consultar el registro general de consultas (**General Log**). Dependiendo de la versión de MySQL, puede estar activado por defecto o no. En la versión actual, MySQL 8 está activado por defecto.

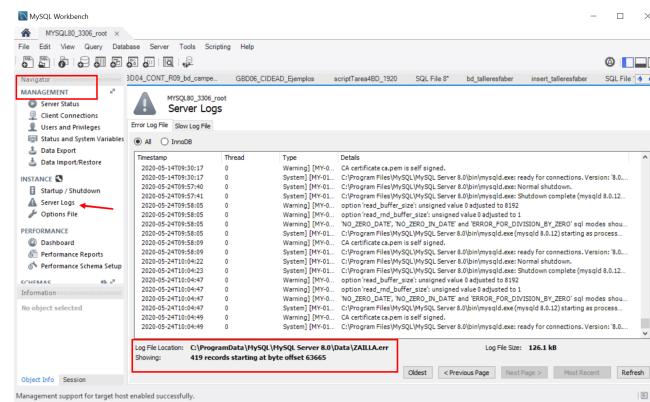
Este registro es muy importante para recuperar todas las actualizaciones que se hayan hecho tras la última copia de seguridad. Si no especificamos lo contrario, este registro se almacena en el directorio **data**, en el archivo **NombreHost-bin**. MySQL agrega un número consecutivo al nombre anterior cada vez que se vuelcan los registros. Ya que este registro se almacena en formato binario, para leerlo es necesario utilizar la herramienta **mysqlbinlog**, incluida en el directorio **bin** de MySQL

El registro de consultas lentas (**Slow Query Log**): Registra todas las consultas que han llevado más tiempo (en ms) que el establecido por defecto en la variable **long_query_time**. Por defecto este archivo se denomina **NombreHost-slow.log** y se guarda en el directorio de datos. Se utiliza para examinar qué consultas llevan demasiado tiempo para su optimización. Dependiendo de la versión de MySQL, puede estar activado o no por defecto. Para activar este fichero de log basta con activar la línea **slow-query-log** del fichero **my.ini**. Para examinar este registro se utiliza **mysqldumpslow**.

En las siguientes imágenes puedes ver estos archivos en un equipo cuyo NombreHost es ZAILLA, imagen izquierda, y en la derecha los archivos de log en Workbench.



Microsoft Windows y MySQL (Elaboración propia)



Workbench (Elaboración propia)

El servidor MySQL puede crear estos archivos de registro para facilitar el ver lo que está pasando pero es necesario limpiar estos archivos para que no ocupen demasiado cuando se hacen copias de seguridad y volver a registrar en archivos nuevos. Esto puede hacerse con la sentencia: FLUSH LOGS.

Para saber más

Si necesitas más información sobre los ficheros de registro (log) de MySQL pincha en el siguiente enlace:

[Ficheros de registro de MySQL](#)

Autoevaluación

Relaciona cada archivo con su función:

Ejercicio de relacionar

Archivo	Relación	Función
NombreHost.err	0	1.- Contiene todas las consultas ejecutadas por los clientes
NombreHost-slow.log	0	2.- Contiene todas las consultas que modifican la base de datos
NombreHost.log	0	3.- Registra los errores en el servidor
NombreHost-bin	0	4.- Contiene las consultas consideradas lentas

Enviar

Los ficheros log ayudan a recuperar la información relativa al estado de la base de datos.

Recomendación

Realiza las siguientes operaciones sobre tu servidor MySQL:

Localiza el directorio en el que tu servidor MySQL almacena todas las bases de datos. Para ello, ejecuta un comando que devuelva el directorio de datos. Además, localiza en el fichero de configuración my.ini o my.cnf el directorio de datos. Por último, accede al directorio en cuestión y visualiza su contenido.

Localiza en tu sistema el fichero de log que almacena el registro de errores.

Localiza en el fichero de configuración la sección en la que se configuran los ficheros de registro **Slow Query Log** y **General Query Log**. Por ahora no es necesario que realices ningún cambio.

2.1.- Utilidad Mysqlbinlog.

Como hemos dicho los ficheros de log binario se generan en formato binario. Este registro será especialmente útil cuando queramos recuperar sentencias almacenadas en él.

Para poder leer los eventos almacenados en el log binario tenemos dos alternativas:

La utilidad mysqlbinlog.

La sentencia: SHOW BINLOG EVENTS.

Utilidad **Mysqlbinlog.**

Everaldo Coelho and
YellowIcon (GNU/GPL)

La forma de lanzar esta herramienta es:

```
mysqlbinlog [opciones] FicheroLog
```

Ejemplo 1.

Para visualizar el contenido del fichero de log binario infoalisa-bin.000001, escribiríamos en la shell del sistema el siguiente comando:

```
mysqlbinlog infoalisa-bin.000001
```

Algunas de las opciones más importantes de esta herramienta son:

Opciones para Mysqlbinlog

OPCIONES DE mysqlbinlog	Significado
--database=<i>NombreBaseDatos</i>
 -d <i>NombreBaseDatos</i>	Sólo obtenemos las entradas para la base de datos seleccionada.
--local-load=<i>ruta</i>,
 -l <i>ruta</i>	Prepara los ficheros temporales para hacer LOAD DATA INFILE en el directorio que se especifique como ruta.
--offset=<i>N, -o N</i>	Ignora las N primeras entradas.
--password[=<i>contraseña</i>],
 -p[<i>contraseña</i>]	Contraseña para conectarse al servidor.
--result-file=<i>nombre</i>,
 -r <i>nombre</i>	Envía la salida a un fichero cuyo nombre se especifica.
--short-form, -s	Sólo muestra los comandos, sin la información adicional.
--start-datetime=<i>datetime</i>	Comienza a leer los eventos a partir de la fecha dada.
--stop-datetime=<i>datetime</i>	Lee los eventos hasta la fecha dada.
--start position=<i>N</i>	Lee el registro binario desde la posición N.
--stop position=<i>N</i>	Lee el registro binario hasta la posición N.
--disable-log-bin, -D	Desactiva el log binario.
--user=<i>NombreUsuario</i>,
 -u <i>NombreUsuario</i>	Nombre del usuario que se conecta al servidor remoto.

Ejemplo 2. Recuperar un fallo ejecutando las sentencias del log binario enviando la salida a un cliente mysql:

```
mysqlbinlog infoalisa-bin.000001 | mysql
```

Ejemplo 3. Para enviar la salida a un fichero de texto por si necesitamos hacer alguna modificación:

```
mysqlbinlog infoalisa-bin.000001 > /tmp/fichero.sql
```

Ejemplo 4. Actualizar un servidor MySQL desde el registro binario:

```
mysqlbinlog FicheroLog | mysql -h NombreServidor.
```

Sentencia SHOW BINLOG EVENTS.

```
SHOW BINLOG EVENTS [IN 'NombreFicheroLog'] [FROM pos] [LIMIT [offset, ] row_count]
```

NombreFicheroLog: escribimos el nombre del fichero log que queremos ver. Si no se escribe ninguno mostrará el primero de ellos.

LIMIT: puede tomar los mismos valores que en el caso de la sentencia SELECT. Es conveniente utilizar un valor en esta cláusula para evitar que el proceso sea largo y que consuma demasiados recursos.

Para saber más

Para ver todas las opciones disponibles para la herramienta mysqlbinlog consulta el siguiente enlace:

[La herramienta mysqlbinlog.](#)

Autoevaluación

Indica si la siguiente afirmación es verdadera o falsa:

'Para leer los eventos de log binario a partir de una fecha dada, se debe utilizar la siguiente opción **--start-datetime=<i>datetime </i>** al usar la utilidad mysqlbinlog"

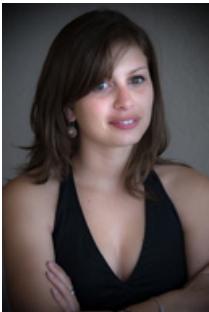
- Verdadero Falso

Verdadero

De las muchas opciones que permite mysqlbinlog, esa es la opción que comienza a leer los eventos a partir de la fecha dada.

3.- Mantenimiento de tablas.

Caso práctico



[Alain Bachellier \(CC BY-NC-SA\)](#)

A **Noiba** se le han quedado grabados dos conceptos que considera muy importantes '**mantenimiento**' y '**preventivo**'. Sin dejar de lado la importancia de las copias de seguridad, pretende ahora revisar las sentencias que maneja el **SGBD** para los casos en que una tabla pierda su integridad poder recuperar los datos sin sufrir pérdidas y si es posible evitar que esas situaciones se produzcan.

Sabe que, al igual que para nuestros mecánicos en el taller, para el correcto funcionamiento de la base de datos el mantenimiento y la prevención son esenciales.

Uno de los problemas que nos podemos encontrar en el mantenimiento de una base de datos, es que las tablas puedan resultar dañadas en algún momento con el desarrollo de la actividad normal de la empresa.

Es necesario tener en cuenta que además de fallos de hardware, interrupciones de corriente, cierre indebido del servidor, y otras circunstancias que ya hemos mencionado; distintas personas van a acceder a los datos y se pueden producir fallos por descuidos o porque estos usuarios tengan distintos niveles de conocimientos con relación al tratamiento de los datos.

Vamos a tratar ahora la detección y resolución de problemas con las tablas, independientemente de cómo surjan esos problemas.

Para verificar y reparar las tablas de nuestra base de datos con MySQL, podemos utilizar distintas herramientas:

Las utilidades myisamchk y mysqlcheck, para tablas de tipo MyISAM.

Mediante sentencias SQL que podemos ejecutar directamente o mediante una interfaz denominada mysqlcheck.

La ventaja de utilizar las sentencias directamente es que el trabajo lo hace el servidor y con mysqlcheck tendremos que asegurarnos que el servidor no utiliza las tablas a la vez, provocando interferencias.

Las sentencias para el mantenimiento de tablas en MySQL las clasificamos en función del tipo de tablas a las que se pueden aplicar; ya que nos vamos a encontrar con que en algunos casos sólo pueden utilizarse con tablas de tipo MyISAM y en otros casos con varios tipos de tablas como MyISAM e InnoDB (y BDB), que son las más utilizadas:

Sentencias de mantenimiento para tablas MyISAM.

CHECKSUM
REPAIR TABLE

Para otros tipos de tablas (MyISAM, InnoDB, BDB), etc.

ANALYZE TABLE
CHECK TABLE
OPTIMIZE TABLE

El procedimiento general para detectar los daños de la tabla y corregirlos es el siguiente:

- 1.- Verificar la tabla en busca de errores. Si no tiene errores el proceso habrá terminado.
- 2.- Hacer copias de los archivos de la tabla antes de empezar a repararla, por si algo sale mal.
- 3.- Intentar reparar la tabla.
- 4.- Si la reparación falla, restaurar la tabla desde las copias de seguridad y los ficheros log.

Partimos de que hemos realizado **copia de seguridad** de la base de datos y que está **habilitado el uso del log**.

Reflexiona

Hacemos hincapié en las tablas de tipo MyISAM e InnoDB porque, tal como se dijo en una unidad anterior, son las más utilizadas en función de la utilidad que vayamos a dar a nuestra base de datos.

Si nuestra intención es publicar en la web: las tablas MyISAM son más rápidas
Para poder trabajar con transacciones y claves ajenas: necesitamos tablas InnoDB.

Por este motivo es fundamental conocer estas herramientas en función del tipo de tablas con las que trabajemos.

Autoevaluación

Selecciona las sentencias que podemos aplicar a una tabla de tipo MyISAM:

CHECK TABLE.

CHECKSUM.

REPAIR TABLE.

OPTIMIZE TABLE.

[Mostrar retroalimentación](#)

Solución

1. Correcto
2. Correcto
3. Correcto
4. Correcto

3.1.- Utilidades para tablas MyISAM: Myisamchk.



Esta utilidad sirve para comprobar, reparar y optimizar tablas MyISAM.

Antes de ejecutar **myisamchk** debemos comprobar que ningún otro programa esté utilizando las tablas. Si esto ocurre es necesario cerrar ese programa y ejecutar FLUSH TABLES para actualizar los cambios que se hayan podido hacer.

La forma de ejecutar este programa es la siguiente:

Everaldo Coelho and
YellowIcon (GNU/GPL)

myisamchk [opciones] NombreTabla1, ...

NombreTabla: Si la tabla no se encuentra en el directorio actual habrá que indicar la ruta. Pueden comprobarse una o varias tablas o indicar todas las tablas de un directorio refiriéndonos a ellas por los archivos de índices como *.MYI.

Opciones: Son muchas las opciones que se pueden añadir a **myisamchk**, haremos un resumen de las más utilizadas:

Opciones para Myisamchk

OPCIONES DE myisamchk	SIGNIFICADO DE LA OPCIÓN
--silent, -s	Activa el modo silencioso que sólo escribe cuando ocurre algún error.
--verbose, -v	Imprime más información.
--version, -V	Informa únicamente sobre la versión.
--wait, -w	Si la tabla está bloqueada, espera a que se desbloquee y continúa.

OPCIONES PARA COMPROBAR TABLAS

--check, -c	Es la opción por defecto. Comprueba si hay errores en la tabla.
--check-only-changed, -C	Sólo se comprueban las tablas que han cambiado desde la comprobación anterior.
--extended-check, -e	Comprueba la tabla de forma minuciosa.
--fast, -F	Comprueba las tablas cerradas de forma inadecuada.
--force, -f	Repara automáticamente la tabla si encuentra algún error.
--medium-check, -m	Comprueba más rápido que extend-check.
--update-state, -U	Indica que se ha comprobado la tabla y guarda información en el archivo .MYI.

OPCIONES PARA REPARAR TABLAS

--backup, -B	Hace una copia de seguridad del archivo .MYD.
--correct-checksum	Corrige el checksum de la tabla.
--extend-check, -e	Intenta reparar todos los registros del archivo de datos. No es conveniente.

OPCIONES DE myisamchk	SIGNIFICADO DE LA OPCIÓN
--force, -f	Sobreescribe archivos temporales (.TMD) sin interrumpir la reparación.
--keys-used=#, -k #	Indica qué índices tiene que actualizar.
--quick, -q	No modifica el archivo de datos por lo que la reparación es más rápida.
--recover, -r	Es la opción más común para la reparación de tablas.
--safe-recover, -o	Repara algunos casos que no se pueden reparar con -r.
--sort-recover, -n	Utiliza ordenación para establecer las claves.
OTRAS OPCIONES	
--analyze, -a	Analiza la distribución de las claves, ayudando a escoger el orden y las claves al unir las tablas con join.
--description, -d	Imprime una descripción de la tabla.
--set-auto-increment[<i>=value</i>], -A[<i>value</i>]	Sirve para especificar un valor para la numeración autoincremental.
--sort-index, -S	Ordena los bloques de índices para que las búsquedas sean más rápidas.
--sort-records=#, -R #	Ordena los registros con un índice que hace los datos más localizables.

Para saber más

Las opciones de myisamchk referenciadas son sólo un resumen de todas las que dispone esa herramienta. Para conocer el resto de opciones, junto con ejemplos de reparaciones, visita el siguiente enlace:

[La utilidad myisamchk para el mantenimiento de tablas MyISAM.](#)

Ejercicio resuelto

Escribe la sentencia para comprobar todas las tablas de la base de datos pruebas usando myisamchk.

NOTA: Puedes descargar el script de creación de la base de datos pruebas.

[CreatePruebas](#) (zip - 535 B)

[Mostrar retroalimentación](#)

```
C:\ProgramData\MySQL\MySQL Server 8.0\data\pruebas>myisamchk *.MYI
```

3.1.1.- Utilidad Mysqlcheck.



Esta aplicación comprueba, analiza, optimiza y repara las tablas de tipo MyISAM. Presenta muchas similitudes con **myisamchk**, la diferencia es que **mysqlcheck** no necesita que el servidor esté parado y myisamchk sí.

La forma de ejecutar este programa es la siguiente:

[Everaldo Coelho and YellowIcon \(GNU/GPL\)](#)

`mysqlcheck [opciones] NombreBaseDatos [NombreTabla1, ...]`

Para más de una base de datos:

```
mysqlcheck [opciones] --databases NombreBaseDatos1 [NombreBaseDatos2 NombreBaseDatos3 ...]  
mysqlcheck [opciones] --all-databases
```

Esta herramienta usa los comandos SQL, CHECK TABLE, REPAIR TABLE, ANALYZE TABLE, OPTIMIZE TABLE, según convenga a la operación que queramos realizar.

Por defecto utiliza la opción -c. Si en lugar de mysqlcheck lo sustituimos por:

Mysqlrepair: repara tablas. Como la opción –repair.
Mysqlanalyze: analiza las tablas. Como la opción –analyze.
Mysqloptimize: optimiza las tablas. Como la opción –optimize.

Algunas de las opciones que admite mysqlcheck son:

Opciones para Mysqlcheck

OPCIONES DE mysqlcheck	SIGNIFICADO DE LA OPCIÓN
..all databases, A	Para comprobar todas las tablas de todas las bases de datos.
--analyze, -a	Analiza las tablas.
--auto-repair	Si al comprobar una tabla encuentra fallos, la repara automáticamente.
--check, -c	Comprueba las tablas en busca de errores.
--check-only-changed, -C	Comprueba únicamente las tablas que han sufrido cambios o se han cerrado mal.
--extended, -e	Comprueba las tablas al 100%, pero tarda mucho y genera registros basura.
--medium-check, -m	Chequeo más rápido que –extend y comprueba las tablas al 99,99%.
--optimize, -o	Optimiza las tablas.
--quick, -q	Es el chequeo más rápido.
--repair, -r	Repara prácticamente todo tipo de errores en las tablas.
--silent, -s	Modo silencios en el que sólo se muestran los mensajes de error.

Para saber más

Para ampliar la información sobre mysqlcheck consulta el siguiente enlace:

[Sentencia mysqlcheck.](#)

Ejercicio resuelto

Partiendo de la base de datos denominada **pruebas** y dentro de ella una tabla **agenda** de tipo MyISAM.

Utilizando la base de datos del apartado 3.1.- chequea la tabla con **mysqlcheck** optimizándola.

[Mostrar retroalimentación](#)

```
C:\ProgramData\MySQL\MySQL Server 8.0\data>mysqlcheck -uroot -pusuario -hlocalhost -o pruebas. agenda
```

3.2.- Sentencias para el mantenimiento de tablas I: ANALYZE TABLE y OPTIMIZE TABLE.

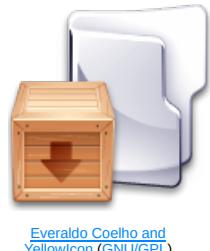
Para analizar tablas tanto MyISAM como InnoDB podemos usar las siguientes sentencias: ANALYZE TABLE, OPTIMIZE TABLE y CHECK TABLE.

ANALYZE TABLE

Se utiliza para analizar y almacenar los índices de una tabla, para ello bloquea las tablas para lectura.

Esta información almacenada se utilizará para establecer en qué orden se hacen los joins en las tablas. La distribución de claves almacenada se puede ver con el comando SHOW INDEX.

Funciona de forma similar a myisamchk -a, pero en este caso se puede utilizar con tablas **BDB** e **InnoDB**, además de **MyISAM**.



```
ANALYZE [LOCAL | NO_WRITE_TO_BINLOG] TABLE NombreTabla1 [, NombreTabla2, ...]
```

El resultado de este comando devuelve una tabla con 4 columnas:

Nombre: con el nombre de la tabla.
Op: la opción analyze (análisis).
Msg_type: que puede contener los valores: status, error, info o warning (estado, error, información o aviso).
Msg_text: contiene el mensaje de texto.

El resultado de este comando se escribe en el log binario excepto si incluimos NO_WRITE_TO_BINLOG o LOCAL. Si la tabla no ha cambiado desde el último ANALYZE, no se vuelve a analizar.

Una de las razones para emitir ANALYZE TABLE es que cuando una tabla sufre muchas modificaciones (INSERT o DELETE, por ejemplo), con el tiempo el optimizador no puede hacer la mejor elección cuando se trata de decidir si debe utilizar un índice específico o no. ANALYZE ayuda al optimizador de consultas a tomar decisiones correctas mediante un análisis detallado de los datos, a diferencia del optimizador de consultas que hace un análisis rápido.

OPTIMIZE TABLE

Se utiliza cuando queremos recuperar el espacio que ocupaban los datos borrados en tablas con registros de longitud variables, (son las que usan columnas de tipo VARCHAR, TEXT, BLOB, etc.).

Por defecto ese espacio no está disponible cuando insertamos nuevos datos. OPTIMIZE TABLE desfragmenta el fichero que contiene los datos resultando una tabla más pequeña.

Funciona con tablas InnoDB, **BDB** y MyISAM. Se puede hacer funcionar con otros tipos de tabla modificando las opciones de arranque.

En el caso de tablas MyISAM: repara la tabla, ordena los índices y actualiza las estadísticas.

Para tablas **BDB** : se mapea como ANALYZE TABLE.

Para tablas InnoDB se mapea como ALTER TABLE: reconstruye la tabla, las estadísticas, los índices y libera el espacio no usado.

La sintaxis de este comando es:

```
OPTIMIZE [LOCAL | NO_WRITE_TO_BINLOG] TABLE NombreTabla1 [NombreTabla2, ...] ...
```

Este comando se escribe en el log binario excepto que se establezca NO_WRITE_TO_BINLOG o LOCAL

Mientras se ejecuta, esta sentencia bloquea la tabla que está optimizando.

Ejercicio resuelto

Partiendo de la tabla **pruebas.agenda** del ejercicio anterior, analiza la tabla con la sentencia ANALYZE y comprueba que el resultado se ha guardado en el log binario con la sentencia SHOW BINLOG EVENTS.

[Mostrar retroalimentación](#)

```
ANALYZE TABLE agenda;
SHOW BINLOG EVENTS IN 'distancia-bin.000004';
```

Para ver el resultado de la sentencia observa la siguiente captura de pantalla:

Workbench (Elaboración propia)

3.2.1.- Sentencias para el mantenimiento de tablas II: CHECK TABLE.

CHECK TABLE

Es un comando que chequea una o más tablas en busca de errores. También puede comprobar errores en las vistas como por ejemplo tablas a las que se hace referencia en una vista y ya no existen.



Everaldo Coelho and
YellowIcon (GNU/GPL)

CHECK TABLE NombreTabla1 [NombreTabla2, ...] ... [Opciones]

El resultado de este comando devuelve una tabla con 4 columnas:

- Nombre: el nombre de la tabla.
- Op: La opción check (chequear).
- Msg_type: que puede contener los valores: status, error, info o warning (estado, error, información o aviso).
- Msg_text: contiene el mensaje de texto.

La salida de este comando puede ser de varios registros por cada tabla que chequeamos. Lo normal es que el último de esos registros tenga los valores de status en Msg_type y de Ok o un mensaje similar en Msg_text, eso significará que no es necesario chequear la tabla. Si no devuelve esos valores será necesario reparar la tabla.

Las opciones, sólo disponibles para tablas MyISAM, (en InnoDB no se tienen en cuenta) pueden ser:

- QUICK: No escanea los registros en busca de enlaces incorrectos.
- FAST: Sólo chequea las tablas que no se hayan cerrado correctamente.
- MEDIUM: Verifica que los enlaces borrados estén correctos y calcula el _____checksum de la clave para los registros comparándolo con el checksum calculado para las claves.
- EXTENDED: Asegura que la tabla es totalmente consistente ya que hace una búsqueda completa para todos los índices de cada registro. Es muy lento.
- CHANGED: Sólo verifica las tablas que no se han cerrado correctamente o se han modificado desde el chequeo anterior.

Algunos aspectos a tener en cuenta:

La opción por defecto es MEDIUM y es equivalente a ejecutar: myisamchk -medium-check NombreTabla. (Si hemos especificado CHANGED o FAST por defecto la opción será QUICK). Cuando una tabla está corrupta probablemente el problema esté en el índice y no en los datos. Si no se tiene prisa no se debe usar la opción QUICK. Cuando se chequea desde un script se suele utilizar FAST y CHANGED. Es más común el uso de FAST. EXTENDED se usa si después de un chequeo normal todavía quedan errores.

Algunos errores detectados con CHECK TABLE no pueden corregirse automáticamente, por ejemplo el valor 0 en un campo auto_increment que puede ocasionar problemas.

Ejercicio resuelto

Chequear la tabla pruebas.agenda, de los apartados anteriores, para ver si se ha cerrado correctamente. Utilizar un chequeo rápido.

[Mostrar retroalimentación](#)

```
CHECK TABLE Agenda FAST QUICK;
```

The screenshot shows the MySQL Workbench interface with a results grid. The grid has columns: Table, Op, Msg_type, and Msg_text. There is one row with the values: pruebas.agenda, check, status, and Table is already up to date.

	Table	Op	Msg_type	Msg_text
▶	pruebas.agenda	check	status	Table is already up to date

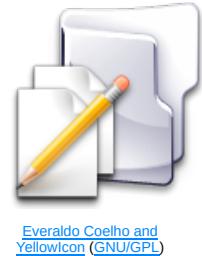
Workbench (Elaboración propia)

3.2.2.- Sentencias para el mantenimiento de tablas III: CHECKSUM TABLE, REPAIR TABLE.

Las sentencias de mantenimiento disponibles para tablas MyISAM son las siguientes: **CHECKSUM TABLE** y **REPAIR TABLE**.

CHECKSUM TABLE

Checksum o suma de comprobación de una tabla es una manera de controlar la redundancia, que verifica si los datos están o no corruptos garantizando así la integridad de los mismos. Consiste en almacenar cada byte y guardar el resultado. Posteriormente compara el checksum que se genera al leer los datos con el que resulta grabado al final. Si no coinciden se habrá producido algún error. Es un control muy simple pero no muy seguro.



CHECKSUM TABLE NombreTabla1 [, NombreTabla2, ...] ... [QUICK | EXTENDED]

Si se especifica QUICK el checksum es muy rápido. Si esta opción no está disponible devolverá NULL. Para poder usar esta opción tendremos que haber especificado **CHECKSUM=1** en la sentencia CREATE TABLE.

REPAIR TABLE

Se utiliza para reparar tablas que previsiblemente estén corruptas. Únicamente funciona en el caso de tablas MyISAM.

Es similar a utilizar myisamchk --recover

La sintaxis de este comando es:

**REPAIR [LOCAL | NO_WRITE_TO_BINLOG] TABLE NombreTabla1 [, NombreTabla2, ...] ...
 [QUICK] [EXTENDED] [USE_FRM]**

El resultado de este comando devuelve una tabla con 4 columnas:

Tabla: el nombre de la tabla.

Op: la opción repair (reparar).

Msg_type: que puede contener los valores: status, error, info o warning (estado, error, información o aviso).

Msg_text: contiene el mensaje de texto.

La salida de este comando puede ser de varios registros por cada tabla que repararemos. Lo normal es que el último de esos registros tenga los valores de status en Msg_type y de Ok en Msg_text, eso significará que no es necesario reparar la tabla. Si no devuelve esos valores será necesario reparar la tabla con myisamchk –safe—recover ya que este comando tiene más opciones que REPAIR TABLE.

QUICK: es equivalente a myisamchk –recover –quick. Repara sólo el índice.

EXTENDED: es equivalente a myisamchk –safe –recover. Crea el índice registro a registro.

USE_FRM: se usa cuando falta o está corrupto el fichero de índices (.MYI). Esta opción no está disponible con myisamchk, aunque no siempre es recomendable porque puede producir la pérdida de datos (metadatos), por ejemplo si la tabla está comprimida.

Este comando se escribe en el log binario excepto que se establezca **NO_WRITE_TO_BINLOG** o **LOCAL**.

Ejercicio resuelto

Intenta reparar la tabla pruebas.agenda de la base de datos de los apartados anteriores.

[Mostrar retroalimentación](#)

```
CHECKSUM TABLE Agenda EXTENDED;  
REPAIR TABLE Agenda QUICK;
```

Query 1 × |

```
1 • USE Pruebas;  
2 • REPAIR TABLE Agenda EXTENDED;  
3 • CHECKSUM TABLE Agenda EXTENDED;
```

Overview Output Snippets Query 1 Result

Table Op Msg_type Msg_text
pruebas.agenda repair status OK

Query 1 Result × |

Table Checksum
pruebas.agenda 2177763417

Workbench (Elaboración propia)

3.3.- Herramientas gráficas para el mantenimiento de tablas.

Además de las sentencias SQL y otros programas cliente que incorpora MySQL para analizar y reparar tablas, existen herramientas gráficas que incorporan interfaces para la administración de la base de datos e incluyen opciones para optimizar, chequear y reparar las tablas.

Entre esas herramientas gráficas disponibles de forma gratuita en el mercado, que incorporan opciones para llevar a cabo el mantenimiento de las tablas, se encuentran **MySQL Workbench** y **PhpMyAdmin**.

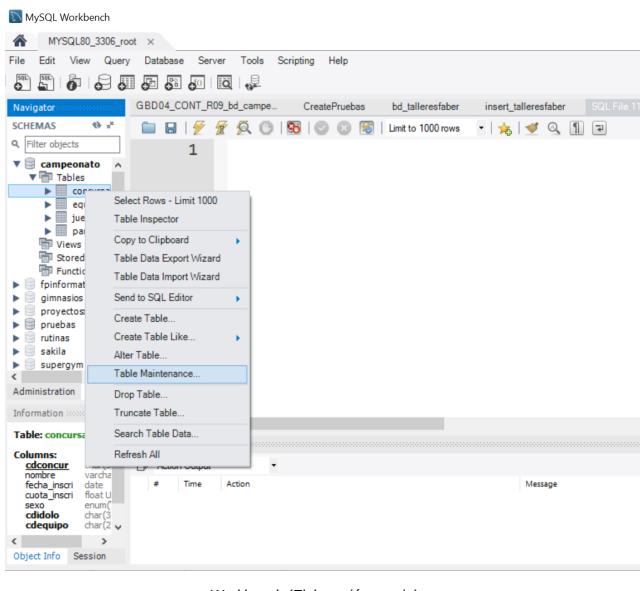
[Everaldo Coelho and Yellowico
\(GNU/GPL\)](#)

Veamos cómo hacer el mantenimiento de tablas con Workbench:

◀ 1 2 3 ▶

Mantenimiento de tablas en Workbench 1

- 1.- Ponemos en uso la base de datos con la que vamos a trabajar, haciendo doble clic sobre ella, en este caso, campeonato.
- 2.- Hacemos un clic derecho sobre una de las tablas de la base de datos, en este caso concursante.
- 3.- En el menú contextual seleccionamos la opción "**Table Maintenance...**"



Workbench (Elaboración propia)

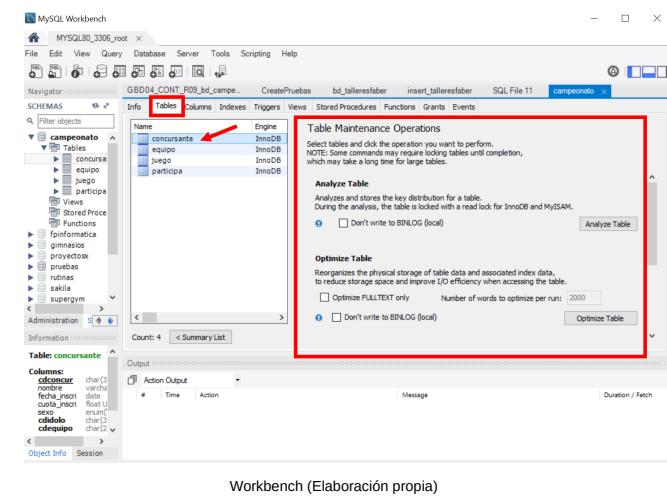
Mantenimiento de tablas en Workbench 2

En la nueva ventana:

- 1.- Selecciona la pestaña TABLES.
- 2.- Selecciona la tabla que te interese, en este caso concursante.
- 3.- Y verás en la sección derecha que las opciones disponibles son:
 - 3.1.- Analizar la tabla (ANALYZE TABLE).
 - 3.2.- Optimizar la tabla (OPTIMIZE TABLE).
 - 3.3.- Revisar la tabla (CHECK TABLE).

3.4.- Obtener el checksum de la tabla (CHECKSUM TABLE).

Existe además la posibilidad de indicar que no se escriba la información resultante en el log binario.

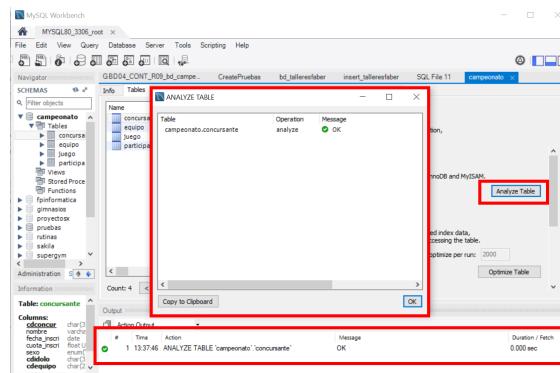


Workbench (Elaboración propia)

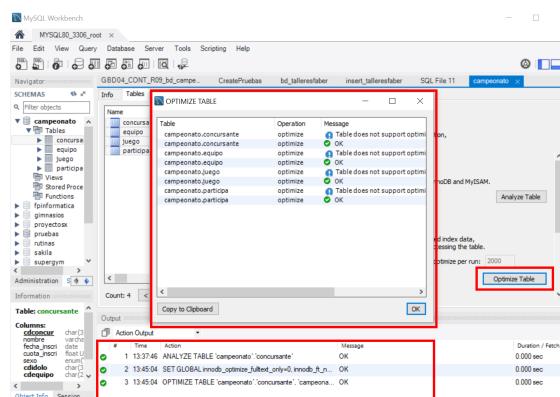
Mantenimiento de tablas en Workbench 3

Una vez seleccionada la tabla o tablas que vamos a mantener y la opción deseada, la operación se lleva a cabo automáticamente, tal y como vemos en las siguientes imágenes:

Ejemplo de ANALYZE de una sola tabla (concurante), en la imagen izquierda y Ejemplo de OPTIMIZE de cuatro tablas en la imagen derecha (concurante, equipo, juego, participa).



Workbench (Elaboración propia)



Workbench (Elaboración propia)

Debes conocer

Para ver cómo realizar el mantenimiento de las tablas de una base de datos desde PhpMyAdmin consulta en el enlace siguiente:

[Mantenimiento de tablas con PhpMyAdmin](#) (pdf - 0,69 MB)

Autoevaluación

MySQL Workbench y PhpMyAdmin incorporan una interfaz gráfica para el mantenimiento de tablas que implementa algunas sentencias SQL vistas en esta unidad. Selecciona de la siguiente lista, las sentencias a las que nos referimos:

REPAIR TABLE.

CHECK TABLE.

OPTIMIZE TABLE.

ANALYZE TABLE.

[Mostrar retroalimentación](#)

Solución

1. Correcto
2. Correcto
3. Correcto
4. Correcto

4.- Copias de seguridad.

Caso práctico



Alain Bachellier (CC BY-NC-SA)

En el apartado anterior **Noiba y Naroba** han considerado todas las situaciones que pueden provocar fallos en la base de datos y, han visto que muchas de ellas son imprevisibles.

Ellas saben que **la mejor solución de los problemas es la prevención**, pero en muchos casos esto no es posible. ¿Cómo podría evitar que el disco duro que guarda la base de datos sufra algún daño? Ahora que TalleresFaber está en pleno funcionamiento tiene claro que no debe correr riesgos. Si no puede evitar que en algún momento la base de datos pueda dañarse, si puede establecer una estrategia para poder recuperar la información, si eso ocurre.

Noiba y Naroba se disponen a estudiar cómo hacer **copias de seguridad de una base de datos** y poder restaurarlos después. Veamos cuáles son los pasos necesarios.

La información contenida en una base de datos debe estar siempre actualizada. Una vez implantado un **SGBD** es muy importante asegurar esa información, puesto que de ello va a depender el desarrollo de todas las aplicaciones posteriores. Una de las cuestiones más importantes a tener en cuenta, es establecer una política de copias de seguridad o **backup**. Las copias de seguridad de la base de datos tienen una importancia vital en caso de una caída grave del sistema.

Una **copia de seguridad o respaldo** se realiza con el objetivo de poder restaurar el original cuando se produzca una pérdida de datos garantizando la **integridad y la disponibilidad**.

A la hora de hacer una copia de seguridad hay que considerar:

- El **dispositivo** en que se van a almacenar los datos de la copia de seguridad.
- Procedimiento** de copia optimizado. Que permita trabajar con datos en uso.
- Procesos de **compresión y cifrado**.

Aunque parece un proceso sencillo en algunos casos es importante copiar únicamente los ficheros que se hayan modificado. Para ello se utilizan sistemas **incrementales** que permiten copiar sólo los bloques físicos que han sufrido cambios.

Cuando la copia de seguridad se hace sobre datos en uso es posible que haya ficheros abiertos y que este proceso tarde varios minutos, para ello se bloquea el fichero para evitar que se produzcan cambios, otra posibilidad es hacer una imagen del fichero y recoger un registro de los cambios que se han producido mientras dura este proceso.

La realización de copias de seguridad siempre tiene impacto en el sistema por eso es importante programar su ejecución para aumentar su efectividad y nivel de optimización.

Reflexiona

La copia de seguridad es el mejor sistema para proteger los datos de una empresa, por tanto es muy importante que ésta se haya realizado correctamente y que pueda ser restaurada a su ubicación original

4.1.- Tipos de copias de seguridad.

En MySQL podemos realizar copias de seguridad de diferentes formas. Vamos a verlas.

De tablas, a nivel de SQL:

Con SELECT... INTO OUTFILE.
Con BACKUP TABLE.

De la base de datos:

Con el programa mysqldump.
Con el script **mysqlhotcopy**.

Debes de tener en cuenta las siguientes consideraciones:

En el caso de tablas ISAM y MyISAM también podemos simplemente copiar los archivos que genera cada una de esas tablas (.frm, .MYD, .MYI), pero esto no es válido para tablas InnoDB.

Everaldo Coelho and
YellowIcon (GNU/GPL)

Antes de hacer una copia con **mysqldump** es recomendable activar el **registro binario** para poder recoger los cambios que se produzcan mientras ejecutamos **mysqldump**.

Podemos hacer una copia de seguridad incremental, para ello habrá que activar el registro binario. Para hacer la copia de seguridad incremental será necesario:

Ejecutar FLUSH LOGS, para volcar el contenido de todos los ficheros de log a disco.

Copiar al directorio de seguridad todos los registros binarios desde la última copia de seguridad.

Vamos a ver todos estos procedimientos más detenidamente en los siguientes apartados.



Autoevaluación

Para hacer una copia de seguridad de la base de datos es necesario parar el servidor para que no se modifiquen datos durante el proceso. ¿Verdadero o falso?

- Verdadero Falso

Falso

Puede hacerse una copia de seguridad de todo o parte de una base de datos con el servidor activo.

Debes conocer

En las últimas versiones de MySQL se ofrece una utilidad muy interesante para la realización de copias de seguridad. Se trata de la aplicación **mysqlbackup**, ejecutable desde la línea de comandos. Esta utilidad solo está disponible para los clientes de **MySQL Enterprise Edition**. Permite copiar y restaurar tablas basadas en cualquier motor, aunque está especialmente diseñado para trabajar con tablas InnoDB. La utilidad **mysqlbackup** es capaz de realizar las copias de seguridad usando los siguientes métodos:

Hot backups: es el mejor método para hacer copias de seguridad, ya que la base de datos y aplicaciones asociadas pueden seguir estando activas. De este modo, la copia de seguridad captura los cambios que se produzcan en las tablas durante el proceso de copia. Solo puede llevarse a cabo con tablas de tipo InnoDB.

Warm backups: se lleva a cabo sobre tablas que no sean InnoDB, por ejemplo tablas MyISAM. En este caso la base de datos permanece activa, pero en estado de solo lectura.

Cold backups: en este caso la base de datos no debe estar accesible. Suele llevarse a cabo sobre servidores esclavos en sistemas replicados.

Para saber más

Puedes leer más sobre la utilidad **mysqlbackup** en el siguiente enlace:

[Utilidad mysqlbackup en MySQL.](#)

4.1.1.- Copias de seguridad de tablas con SELECT... INTO OUTFILE.

Podemos hacer una copia de seguridad de los datos de una tabla o del resultado de una consulta, en un fichero de texto plano. (Exportación de datos a texto plano)

Para ello, se utiliza la sentencia SELECT ... INTO con el siguiente formato o sintaxis:

```
SELECT ... INTO OUTFILE 'NombreFichero' OpcionesDeExportación
```

El resultado es que escribe los registros seleccionados con SELECT en un fichero nuevo. Este fichero no debe existir previamente. El fichero de destino debe estar en la máquina servidor, no puede estar en una máquina distinta.

Su utilidad principal es la de permitir volcar rápidamente una tabla en el ordenador que hace de servidor.

Las **OpcionesDeExportación** pueden ser FIELDS y LINES, ambas son opcionales pero si se especifican ambas, FIELDS tiene que ir antes de LINES.



Everaldo Coelho (YellowIcon)
(GNU/GPL)

FIELDS

FIELDS TERMINATED BY 'Cadena'.

FIELDS [OPTIONALLY] ENCLOSED BY 'Carácter'.

FIELDS ESCAPED BY 'Carácter': Controla como escribir caracteres especiales.

LINES

STARTING BY 'Cadena': Ignora el carácter que se usa como prefijo común a todas las líneas.

 LINES TERMINATED BY 'Cadena'.

Si no se especifican, los **valores por defecto** son:

Para FIELDS

```
FIELDS TERMINATED BY '\t' ENCLOSED BY '' ESCAPED BY '\\'
```

Para LINES

```
LINES TERMINATED BY '\n' STARTING BY ''
```

Por defecto este comando funciona:

 Sin comillas en los campos.

 Con tabuladores para separar los campos.

 Cómo carácter de escape la barra (\).

 Al final de cada línea escribe una línea nueva.

Este comando complementa a LOAD DATA ... INFILE. (Realiza la importación del fichero en texto plano a una tabla)

```
LOAD DATA INFILE 'NombreFichero' [REPLACE | IGNORE] INTO TABLE NombreTabla  
OpcionesDelImportación
```

Las **OpcionesDelImportación** son las mismas que las **OpcionesDeExportación** vistas para SELECT ... INTO

Ejemplo: Hacer una copia de seguridad de archivos individuales.

Para volcar la tabla NombreTabla de una base de datos a un archivo en texto plano de nombre archivo, sería:

```
SELECT * INTO OUTFILE 'NombreArchivo' FROM NombreTabla;
```

Para restaurar la tabla (importar los datos del archivo a una tabla):

```
LOAD DATA INFILE 'NombreArchivo' INTO TABLE NombreTabla REPLACE ...;
```

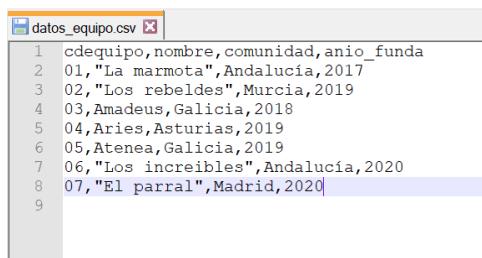
REPLACE se utiliza para evitar registros duplicados cuando un registro nuevo y uno viejo tengan la misma clave. La tabla tiene que tener un índice que no admita duplicados.

Debes conocer

Generalmente los archivos de texto plano que almacenan datos son archivos de extensión CSV.

Recuerda que un archivo de extensión CSV es un documento en formato abierto sencillo para representar datos en forma de tabla, en las que las columnas se separan por comas (o punto y coma).

Por ejemplo:



```
datos_equipo.csv
1 cdequipo,nombre,comunidad,anio_funda
2 01,"La marmota",Andalucia,2017
3 02,"Los rebeldes",Murcia,2019
4 03,Amadeus,Galicia,2018
5 04,Aries,Asturias,2019
6 05,Atenea,Galicia,2019
7 06,"Los increibles",Andalucia,2020
8 07,"El parral",Madrid,2020
9
```

Notepad++ (Elaboración propia)

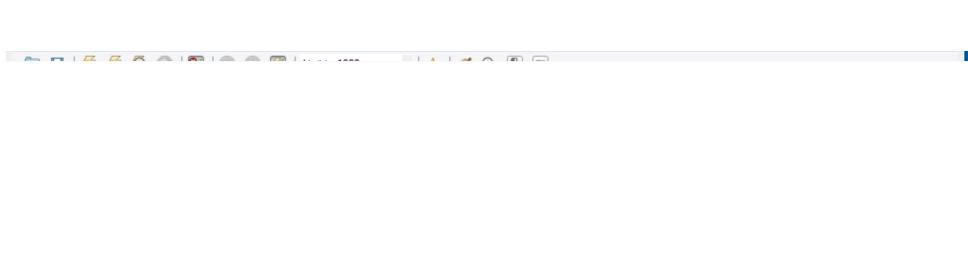
Ejercicio resuelto

Realiza una copia de seguridad de la tabla equipo de la base de datos compeonato. Guarda los datos en un archivo de nombre datos_equipo.csv

[Mostrar retroalimentación](#)

```
USE compeonato;
SELECT * INTO OUTFILE 'datos_equipo.csv' FROM equipo;
```

Observa que se muestra el siguiente error, en el caso de la versión 8 de MySQL:



Workbench (Elaboración propia)

Por lo que si no cambiamos en el archivo de configuración `my.ini` el directorio para los uploads (subidas), tendremos que indicar la ruta por defecto, que sería:

```
SELECT * INTO OUTFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/datos_equipo.csv'  
FROM equipo;
```

Para saber más

Para conocer el resto de opciones de las sentencias anteriores puedes acceder a los siguientes enlaces:

[La sentencia SELECT ... INTO OUTFILE](#)

[La sentencia LOAD DATA](#)

4.1.2.- Copias de seguridad de tablas con BACKUP TABLE.

Para copiar al directorio de destino el mínimo número de ficheros que se necesita para poder restaurar la tabla, se usa la siguiente sentencia.

```
BACKUP TABLE NombreTabla1, [, NombreTabla2..] ... TO ' / / NuevoDirectorio'
```

Este comando solo es válido con tablas MyISAM. Como sabemos las tablas MyISAM se almacenan en disco en 3 ficheros que se denominan igual que la tabla pero con las extensiones:

- frm: almacena la definición de la tabla.
- MYD: almacena los datos.
- MYI: almacena el índice.

Este comando copia los ficheros .frm y .MYD, el fichero de índice .MYI se reconstruye a partir de los otros dos.



Everaldo Coelho and
YellowIcon (GNU/GPL)

Antes de hacer la copia de seguridad se efectúa un bloqueo de lectura previo de cada tabla, si la copia de seguridad afecta a varias tablas, se debe utilizar LOCK TABLES para hacer un bloqueo de lectura de las mismas y así evitar que se hagan cambios en alguna mientras se está haciendo la copia de seguridad.

Para restaurar la copia de seguridad se utiliza el comando RESTORE TABLE cuya sintaxis es:

```
RESTORE TABLE NombreTabla1, [, NombreTabla2..] ... FROM ' / / NuevoDirectorio'
```

Debes conocer

Tanto BACKUP TABLE como RESTORE TABLE son comandos que están en desuso. En la versión de MySQL 5.5 y posteriores ya no se incluyen. En su lugar es mejor utilizar mysqldump.

4.1.3.- La utilidad mysqldump.

La utilidad o programa **mysqldump** permite hacer **copias de seguridad de una o varias bases de datos**, o bien de **algunas tablas** de una base de datos, **incluso de sólo la estructura**. Es un programa muy versátil, como verás, por la cantidad de opciones que incluye.

Esta utilidad permite transferir datos a cualquier otro servidor que puede no ser MySQL. Esto se hace mediante comandos **SQL** que facilitan la creación de la tabla y el volcado de los datos.

Everaldo Coelho and YellowIcon (GNU/GPL)

El archivo creado por mysqldump es un archivo de texto que contiene las sentencias **SQL** necesarias para crear las tablas e insertar los datos en las filas. Es un fichero de

salida que puede usarse después para generar de nuevo la base de datos, bien desde la línea de comandos de MySQL o desde la pantalla de creación de sentencias **SQL** de cualquier entorno gráfico.

Las limitaciones de la restauración dependerán de las opciones que se han especificado al hacer la copia de seguridad, por tanto es muy importante elegir estas opciones correctamente para no llevarnos sorpresas desagradables.

Para volcar una o más bases de datos en un fichero de texto pueden usarse distintas sintaxis:

Mysqldump [opciones] NombreBaseDatos [Tablas]

Mysqldump [opciones] --databases NombreBaseDatos1[NombreBaseDatos2 NombreBaseDatos3 ...]

Mysqldump [opciones] --all -databases

Este programa admite una gran variedad de opciones, de ellas vamos a destacar las más útiles:

Opciones para Mysqldump.

OPCIONES DE mysqldump	SIGNIFICADO DE LA OPCIÓN
--add-drop-table	Añade el comando drop table antes de create table al restaurar.
--add-locks	Añade bloqueos de tabla para que las inserciones sean más rápidas.
--all-databases, -A	Copia todas las bases de datos. Es lo mismo que utilizar --databases seleccionando todas.
--compatible=<i>Sistema</i>	La salida de mysqldump será compatible con el sistema elegido que puede ser: oracle, postgre, db2, etc.
--create-options	Incluye todas las opciones de tabla específicas de MySQL en CREATE TABLE.
--databases, -B	Se incluye el CREATE DATABASE y USE antes de cada base de datos.
--flush-logs, -F	Escribe en disco todos los logs antes de comenzar con la copia.
--force, -f	Continúa aunque se produzca un error de SQL durante la copia.
--lock-all-tables, -x	Bloquea todas las tablas de todas las bases de datos, con un bloqueo de lectura global.
--no-data, -d	Vuelca únicamente la estructura de la tabla.
--opt	Crea un volcado rápido. Actualmente está activado por defecto, para desactivarlo con --skip-opt.
--password [= <i>contraseña</i>], -	Sirve para introducir la contraseña al conectar con el servidor. Si se omite la contraseña la pide al ejecutar el comando.

OPCIONES DE mysqldump	SIGNIFICADO DE LA OPCIÓN
p[<i>contraseña</i>] 	
--quick, -q	Se utiliza para tablas grandes. Guarda los registros de uno en uno en la memoria previamente.
--routines, -R	Se incluyen en la copia los procedimientos y funciones almacenados.
--single-transaction	Se usa con tablas transaccionales. Ejecuta BEGIN antes de volcar los datos. No se puede usar con --lock-tables.
--skip-triggers	No se incluyen los triggers en la copia. Por defecto se incluyen.
--tables	Los argumentos que vienen a continuación se toman como nombres de tablas.
--triggers	Incluye en la copia los triggers. Por defecto está habilitada. Para saltar triggers usar --skip-triggers
--User=
<i>NombreUsuario</i>,
-u <i>NombreUsuario</i> 	Nombre del usuario para conectar con el servidor.
--where='<i>Condición</i>', -w '<i>Condición</i>' 	Vuelca sólo los registros que cumplen la condición WHERE.

Restauración de copias.

Para realizar la **restauración de una base de datos usando el archivo generado por mysqldump** haríamos lo siguiente:

```
shell>mysql -u usuario -p db_name < backup_file.sql
```

En este caso **db_name** es el nombre de la base de datos que queremos restaurar, y **backup_file.sql** es el archivo generado por mysqldump del siguiente modo:

```
shell>mysqldump -u usuario -p db_name > backup_file.sql
```

Ejemplos de copias y su restauración.

Ejemplo 1. Copia de seguridad de la base de datos campeonato incluyendo el CREATE DATABASE y con todas las rutinas que tenga.

Ejemplo 2. Copia de seguridad de la estructura la base de datos campeonato, solo las tablas y sin el CREATE DATABASE.

Ejemplo 3. Restaurar la copia de seguridad realizada en el Ejemplo 1.

Ejemplo 4. Restaurar la copia de seguridad realizada en el Ejemplo 2.

Para saber más

Para conocer el resto de opciones que admite el programa mysqldump accede al manual en el siguiente enlace:

[Utilidad mysqldump.](#)

Ejercicio resuelto

Crea una copia de seguridad con **mysqldump** de la base de datos TalleresFaber en el archivo **Copia20200501.sql** con los datos de usuario **Noiba** y password **Alisal2011**.

Escribe también el comando para restaurar la copia.

[Mostrar retroalimentación](#)

```
Mysqldump -opt -pAlisal2011 -uNoiba TalleresFaber > Copia20200501.sql  
Mysql -pAlisal2011 -uNoiba < Copia20200501.sql
```

4.1.4.- La utilidad mysqlhotcopy.

La utilidad mysqlhotcopy se utiliza para hacer copias de seguridad cuando todas las tablas que integran la base de datos son de tipo **MyISAM**, ya que tanto la copia, como la restauración son más rápidas.

Para hacer la copia de seguridad esta utilidad hace un bloque (lock) sobre las tablas y copia los archivos de la base de datos MyISAM.

Las características más destacadas de este programa son:

[Everaldo Coelho and YellowIcon \(GNU/GPL\)](#)

Está escrito en Perl.

Solo funciona en plataformas UNIX.

Utiliza las sentencias: LOCK TABLES, FLUSH TABLES, cp ó scp.

Necesariamente debe ejecutarse en el ordenador donde se encuentre el directorio de la base de datos que se va a copiar.

Puede hacerse una copia de toda la base de datos, de tablas o de varias bases de datos.

Sintaxis en el caso de una única base de datos:

Mysqlhotcopy NombreBaseDatos [/ / / NuevoDirectorio]

Para varias bases de datos:

Mysqlhotcopy NombreBaseDatos1, ... NombreBaseDatosN / / / NuevoDirectorio

Cuando queremos hacer una copia de seguridad de tablas que contengan una expresión, la sintaxis es.

Mysqlhotcopy NombreBaseDatos. / [-]Expresión/

Para seleccionar las tablas que no cumplen esa expresión anteponemos a la misma una tilde (~).

Las opciones que soporta **mysqlhotcopy** son las siguientes:

Opciones para Mysqlhotcopy script.

OPCIONES DE mysqlhotcopy	SIGNIFICADO DE LA OPCIÓN
--allowold	Si el destino ya existe lo renombra con _old.
--flushlog	Vuelca los ficheros log después de bloquear todas las tablas.
--keepold	Mantiene sin borrar los destinos renombrados.
--method=#	Para elegir el método de copia, que puede ser cp ó scp.
--noindices	No incluye los índices en la copia de seguridad. Posteriormente se pueden rehacer con <i>myisamchk -rq</i>.
--password=<i>contraseña</i>, -p<i>contraseña</i> 	Contraseña para conectar al servidor. La contraseña no es opcional.
--port=<i>número de puerto</i>, -p<i>número de puerto</i> 	Puerto TCP/IP por el que nos conectamos al servidor local.
--regexp=<i>expresión</i>	Copia las bases de datos cuyos nombres cumplen la

OPCIONES DE mysqlhotcopy	SIGNIFICADO DE LA OPCIÓN
	expresión dada.
--triggers	Vuelca los triggers asociados a las tablas. Está activada por defecto, para desativarla con --skip-triggers .
--user=<i>NombreUsuario</i>,
 -u<i>NomnreUsuario</i>	Nombre del usuario de MySQL para conectarse al servidor.

Para tener información adicional ejecutar: **perldoc mysqlhotcopy**.

Debes conocer

La utilidad mysqlhostcopy **fué deprecated en el versión MySQL 5.6.20 y eliminada en la versión 5.7.**

[Utilidad mysqlhotcopy](#)

Ejercicio resuelto

Haz una copia de seguridad de la base de datos **pruebas** con **mysqlhotcopy**.

[Mostrar retroalimentación](#)

Mysqlhotcopy pAlisal2011 -uNoiba pruebas

Con los datos de usuario **Noiba** y password **Alisal2011**.

4.1.5.- La utilidad mysqlimport.

La utilidad mysqlimport es un programa que se ejecuta desde la línea de comandos y proporciona todas las opciones de la sentencia LOAD DATA ... INFILE

La sintaxis de esta herramientas es la siguiente:

Everaldo Coelho and
YellowIcon (GNU/GPL)

Myslimport [opciones] NombreBaseDatos FicheroTexto1 [FicheroTexto2 ...]

FicheroTexto: El nombre de la tabla de destino lo toma del nombre del fichero de texto, sin tener en cuenta la extensión.

Algunas de las opciones que soporta son:

Opciones para Myslimport.

OPCIONES DE Myslimport	SIGNIFICADO DE LA OPCIÓN
--columns=<i>lista columnas</i>,
 -c <i>lista columnas</i> 	Lista de columnas separadas por comas. El orden de las columnas de datos indica el orden de las columnas de la tabla.
--delete, -D	Vacia la tabla antes de importar el fichero de texto.
--fields-terminated-by=...,
 --fields-enclosed-by=...,
 --fields-optionally-enclosed-by=...,
 --fields-escaped-by=...,
 --lines-terminated-by=... 	Son opciones equivalentes a las cláusulas de LOAD DATA INFILE.
--host=<i>NombreHost</i>,
 -h <i>NombreHost</i> 	Los datos se importan al servidor MySQL del equipo.
--local, -L	Lee los ficheros del equipo local.
--lock-tables, -l	Bloquea las tablas para escritura.
--password [=<i>contraseña</i>], -p[<i>contraseña</i>] 	Contraseña para conectarse al servidor.
--replace, -r
 --ignore 	Controlan cómo tratar los registros que duplican claves. --replace hace que los registros nuevos sustituyan a los anteriores, --ignore los ignora. Si no se pone opción esos valores provocan un error.
--silent, -s	Sólo muestra mensajes cuando hay error.
--user=<i>NombreUsuario</i>,
 -u <i>NombreUsuario</i> 	Usuario para conectarse al servidor MySQL.

Para saber más

Para conocer el resto de opciones que admite el programa mysqlimport y ver algunos ejemplos, accede al manual de MySQL en el siguiente enlace:

[Utilidad mysqlimport.](#)

Ejercicio resuelto

Partiendo del fichero CopiaClientes creado a partir de la tabla TalleresFaber.CLIENTES, con la instrucción:

```
mysql>SELECT * INTO OUTFILE 'CopiaClientes' FROM CLIENTES;
```

Importar la copia con **mysqlimport**:

[Mostrar retroalimentación](#)

```
shell>Mysqlimport -uusuario -p TalleresFaber CopiaClientes
```

4.2.- Planificación de copias de seguridad.

Debido a su importancia, no podemos dejar que la **realización de copias de seguridad** dependa de que las personas que gestionan la base de datos ejecuten este proceso cada cierto tiempo. Si queremos evitar muchos dolores de cabeza éstas **deben programarse para que se ejecuten periódicamente**, de forma **automática**.

Las copias pueden hacerse con diferentes herramientas:

InnoDB Hot Backup: copia de seguridad en línea, sin bloqueos, para archivos InnoDB, que no molesta a las lecturas y escrituras de las tablas. Actualmente ha cambiado su nombre por **MySQL Enterprise Backup**. Es propiedad de Oracle y no es una herramienta gratuita.

[Everaldo Coelho and YellowIcon \(GNU/GPL\)](#)

Mysqldump: copia de seguridad online que ya hemos visto en esta unidad.

Mysqlhotcopy exclusivamente para tablas MyISAM. (*Recuerda que fué eliminada en la versión MySQL 5.7*) Herramientas gráficas como **PhpMyAdmin**, **MySQL Workbench**, etc.

Te puedes estar planteando las siguientes cuestiones

1.- **¿Cuándo hacer la copia de seguridad?** Es importante programar estas copias cuando la carga de trabajo es baja, sobre todo si la copia de seguridad es completa.

2.- **¿Qué tipo de copia de seguridad se debe hacer?** Las copias de seguridad completas no siempre son convenientes porque incluyen también los datos que no han cambiado desde la última copia, llevan mucho tiempo y son muy grandes. Es conveniente realizar una copia de seguridad completa inicialmente y después hacer copias incrementales. Para recuperarlas se recarga la copia completa y las copias incrementales.

3.- **¿Cómo hacer las copias de seguridad?** Para hacer la **copia de seguridad completa** podemos utilizar mysqldump.

Ejemplo 1.

```
shell>mysqldump -u user -p --single-transaction TalleresFaber > Backup_20200531.sql
```

Si utilizamos --single-transaction suponemos que todas las tablas son InnoDB y nos aseguramos que la base de datos se vuelca en estado consistente. Obtenremos un archivo SQL con los comandos **INSERT** para recuperar las tablas volcadas. La copia se podría realizar un domingo (31/05/2020).

Para hacer una **copia de seguridad incremental**; necesitamos guardar los cambios que se hayan producido desde la anterior copia de seguridad, para eso es necesario activar el registro binario al iniciar el servidor para que todos los cambios que actualicen datos se guarden en un archivo de registro binario. Estos archivos se guardan en el directorio data y se crea uno nuevo cada vez que se arranca el servidor. Junto a estos archivos se guarda también un archivo .index. Para iniciar un nuevo registro binario cuando queramos podemos ejecutar **FLUSH LOGS** o **mysqladmin flush-logs**, pero **mysqldump** también tiene una opción para volcar los logs.

Ejemplo 2.

Supongamos que en nuestro taller se actualiza diariamente el registro binario para recoger todos los cambios, por tanto al hacer la copia de seguridad nos interesa volcar el registro binario e inicializar uno nuevo que recoja únicamente los cambios a partir de esa copia. Si tenemos esto en cuenta la copia de seguridad del domingo se hará:

```
shell>mysqldump -u user -p --single-transaction --flush-logs TalleresFaber > Backup_20200531.sql
```

Una vez hecha la copia de seguridad el domingo, en el directorio data habrá un nuevo archivo de registro binario que recogerá los cambios posteriores a la copia que hemos hecho.

Si la copia incremental es diaria, como hemos dicho, cada día ejecutaremos:

```
shell> mysqladmin -u user -p flush-logs  
o  
mysql>FLUSH LOGS;
```

para crear el siguiente registro binario.

4. - **¿ Cómo borrar los registros binarios que no se necesiten?** Los registros binarios ocupan espacio, de vez en cuando será necesario eliminar los que no se necesiten. Para ello al hacer la copia de seguridad se escribe:

Ejemplo 3.

```
shell>mysqldump -u user -p --single-transaction --flush-logs TalleresFaber --delete-master-logs > Backup_20200531.sql
```

5.- **¿Cómo restaurar la copia de seguridad?** Si la última copia de seguridad es la del domingo haremos:

a. Restaurar los datos al domingo:

```
shell>mysql -u user -p talleresfaber < Backup_20200531.sql
```

b. Restaurar los cambios hechos desde entonces, (dado que cada día se ejecuta **mysqladmin flush-logs** o FLUSH LOGS, habrá dos nuevos ficheros binarios, por ejemplo el 7 y el 8):

```
shell>mysqlbinlog hostname-bin.000007 hostname-bin.000008 | mysql -u user -p
```

Autoevaluación

Con relación a la aplicación del log binario al último respaldo de una base de datos, elige la opción correcta:

- Mysqldump NombreHost-bin.000001 | mysql
- Mysqlbinlog NombreHost -bin.000001 | mysql
- Mysqlhotcopy NombreHost -bin.000001 | mysql
- Mysqlcheck NombreHost -bin.000001 | mysql

Incorrecto. Esta herramienta realiza copias de seguridad.

Cierto. Esta utilidad aplica los datos del registro binario.

No es correcto. Esta herramienta realiza copias de seguridad de tablas MyISAM.

Respuesta incorrecta. Esta utilidad verifica errores en las tablas.

Solución

1. Incorrecto
2. Opción correcta
3. Incorrecto
4. Incorrecto

5.- Herramientas gráficas para la importación y exportación de datos y copias de seguridad.

Caso práctico

Los socios del taller mecánico están satisfechos con el funcionamiento del SGBD y su implantación en el negocio. Así se lo han hecho saber a **María**, jefa de la **BK Sistema Informáticos**.

Tanto es así que han decidido instalar un nuevo servidor, de mayores prestaciones, y colocarlo en un nuevo espacio que cumpla con todas las medidas necesarias para garantizar al máximo la seguridad de los datos.

Para **Noiba, Naroba y Vindio** esto plantea la necesidad de exportar la base de datos a un fichero, incluyendo en él la estructura de tablas e índices, así como los datos, procedimientos y funciones almacenados, e importarlo después en el servidor de destino.

[Nate Steiner \(Dominio público\)](#)

Deciden empezar por analizar todas las herramientas que le pueden ayudar en este trabajo para elegir la más conveniente.

En los apartados anteriores hemos visto algunas opciones para exportar los datos de una base de datos a un fichero y poder recuperarlos después de formas distintas:

Mediante el uso de sentencias **SQL**.

Con herramientas y programas cliente que incorpora MySQL.

En este apartado veremos cómo realizar esas funciones utilizando una interfaz gráfica.

Las herramientas gráficas que venimos utilizando hasta ahora incorporan funcionalidades relativas a la administración del servidor entre las que se incluye la posibilidad de exportar e importar datos.

Se analizan en esta unidad:

PhpMyAdmin.

MySQL Workbench.

Navicat for MySQL.

Las características que ofrecen son:

La mayoría de las herramientas ofrecen la posibilidad de volcar la base de datos completa o seleccionar tablas.

La exportación puede hacerse en distintos formatos: **SQL**, **CSV**, texto, **XML**, **JSON**, etc. así como la posterior recuperación.

En algunos casos se ofrece la posibilidad de programar las copias de seguridad para que se realicen automáticamente en un día y hora señalados.

MySQL permite copiar tablas en diferentes formatos de texto, así como importar datos a partir de ficheros de texto también en diferentes formatos.

Esto se puede usar para exportar los datos de nuestras bases de datos a otras aplicaciones, o bien para importar datos desde otras fuentes a nuestras tablas. También se puede usar para hacer copias de seguridad y restaurarlas posteriormente.

5.1.- PhpMyAdmin.

Como vimos en una unidad anterior, **PhpMyadmin** es una herramienta que ofrece la posibilidad de para manejar y administrar bases de datos de MySQL a través de una aplicación web.

Como vemos en la captura de abajo, cuando accedemos a PhpMyAdmin, en la parte superior de la página principal tenemos las opciones:

Importar.
Exportar.

[Everaldo Coelho and
YellowIcon \(GNU/GPL\)](#)

PhpMyAdmin (Elaboración propia)

Veamos algunos **ejemplos de importación y exportación** con PhpMyAdmin.

◀ 1 2 3 4 5 6 ▶

Exportar una base de datos 1

Una vez que pulsas sobre **Exportar**, puedes seleccionar varias opciones. (La opción Rápido realiza una copia de todo el servidor)

En este ejemplo, se va a exportar sólo la base de datos campeonato, por ello seleccionamos la opción **personalizada**, elegimos la base de datos y vamos seleccionando el resto de opciones indicadas...

PhpMyAdmin (Elaboración propia)

Exportar una base de datos 2

Entre las opciones a seleccionar indicamos que la **copia es de la estructura y los datos**.

PhpMyAdmin (Elaboración propia)

Exportar una base de datos 3

Otras opciones que seleccionamos son las que indican **que se copien los objetos programados** de la base datos (procedimientos, funciones, triggers y eventos) y que los datos se copien mediante sentencias INSERT.

PhpMyAdmin (Elaboración propia)

Exportar una base de datos 4

Pulsamos ahora en **continuar** para almacenar el fichero resultante .sql en el lugar de destino. Se le puede cambiar el nombre.

PhpMyAdmin (Elaboración propia)

Importar una base de datos 1

Se puede importar o restaurar una base de datos completa o ciertas tablas.

Vemos el ejemplo de cómo importar la base de datos campeonato cuya copia de seguridad se guarda en un script .sql y puede haber sido generado con cualquier herramienta o utilidad como: mysqldump, Workbench, Navicat, etc.

Debes pulsar en **Importar**, y mediante el botón **Examinar** localizas el archivo que contiene la copia de seguridad, en este caso el archivo campeonato_phpmyadmin.sql, marcas el tipo de archivo que es, en este caso es un archivo SQL y **continuar**.

PhpMyAdmin (Elaboración propia)

Importar una base de datos 2

Si todo ha ido bien, veremos un mensaje de confirmación positiva y en la sección izquierda se mostrará la base de datos restaurada.

Autoevaluación

Señala la principal característica que diferencia a PhpMyAdmin de las otras herramientas para administrar bases de datos que hemos citado:

- Es una herramienta para MySQL.
- Permite la administración de la base de datos.
- La administración se hace desde una página Web.
- Se distribuye bajo licencia GNU.

Incorrecto. El resto de herramientas también se utilizan con MySQL.

No es correcto. El resto de herramientas vistas permiten la administración de MySQL.

Correcto. Con PhpMyAdmin permite la administración desde internet vía WebGNU.

No es cierto. Casi todas las herramientas vistas tienen licencia GNU.

Solución

1. Incorrecto
2. Incorrecto
3. Opción correcta
4. Incorrecto

Recomendación

Para practicar con PhpMyAdmin haz diferentes copias de seguridad de la base de datos campeonato, por ejemplo:

- 1.- Una copia de seguridad con estructura y datos, todos los objetos de la base de datos y el CREATE DATABASE.
- 2.- Una copia de seguridad de sólo la estructura de la base de datos.
- 3.- Una copia de seguridad de sólo la tabla concursante.
- 4.- Restaura las copias realizadas.

5.2.- MySQL Workbench.

La herramienta Workbench también dispone de herramientas que facilitan la exportación e importación de bases de datos.

Las opciones correspondientes pueden encontrarse en el panel de la izquierda, en la sección de Administración: **MANAGEMENT**. Estas opciones son:

Data Export.

Data Import/Restore.

La captura de abajo muestra la ventana que aparece al seleccionar la herramienta **Data Export**. Como puede observarse, la selección de los objetos que serán exportados resulta realmente simple. En la parte inferior de la ventana podemos seleccionar diferentes opciones, como la posibilidad de incluir en la exportación a los procedimientos almacenados o a los eventos.

[Everaldo Coelho \(YellowIcon\)](#)
[\(GNU/GPL\)](#)

Es posible exportar los datos de dos modos:

Export to Dump Project Folder: MySQL Workbench creará un fichero de backup independiente para cada tabla de la base de datos, en un directorio de backups. De este modo, será posible restaurar tablas independientes, sin necesidad de hacerlo con la base de datos completa.

Export to Self-Contained File: Todas las tablas seleccionadas se exportarán a un único fichero SQL.

En la captura de ejemplo se ha seleccionado a la base de datos campeonato, copia de estructura y datos, procedimientos y eventos, la copia se hace en un único archivo .sql y en él se incluye el CREATE DATABASE.

Workbench (Elaboración propia)

La imagen de abajo muestra la ventana que se abre al seleccionar la herramienta **Data Import/Restore**. Como vemos, tendremos que elegir entre **Import Dump Project Folder** o **Import from Self-Contained File**.

En este ejemplo seleccionamos que la copia está en un único archivo .sql, lo seleccionamos e indicamos que se restaure la estructura y los datos.

El archivo a restaurar puede haber sido creado manualmente, mediante la utilidad mysqldump, PhpMyAdmin, o cualquier otra herramienta

Workbench (Elaboración propia)

Autoevaluación

Señala si es verdadera o falsa la siguiente afirmación:

Desde MySQL Workbench podemos importar una copia de seguridad de una base de datos hecha con PhpMyAdmin.

- Verdadero Falso

Verdadero

Puesto que en ambas herramientas los datos se exportan a un fichero SQL la importación de datos es posible.

Recomendación

Para practicar con Workbench haz diferentes copias de seguridad de la base de datos campeonato, por ejemplo:

- 1.- Una copia de seguridad con estructura y datos, todos los objetos de la base de datos y el CREATE DATABASE.
- 2.- Una copia de seguridad de sólo la estructura de la base de datos.
- 3.- Una copia de seguridad de sólo la tabla concursante.
- 4.- Restaura desde Workbech las copias realizadas con PhpMyAdmin y viceversa.

5.3.- Navicat for MySQL.

La herramienta Navicat ofrece una utilidad muy potente para realizar copias de seguridad de las base de datos de un servidor MySQL.

Además de poder realizar copias de seguridad similares a las que hemos visto anteriormente, podemos realizar una programación temporal de las copias.

Observa las siguientes imágenes capturadas de esta aplicación y que son las que permitirán realizar esas tareas.

[Everaldo Coelho and YellowIcon \(GNU/GPL\)](#)

◀ 1 2 3 ▶

Navicat 1

Navicat (Elaboración propia)

Navicat 2

Navicat (Elaboración propia)

Navicat 3

Además, Navicat dispone de un **planificador** muy intuitivo, que permite establecer fácilmente en qué momento se realizará la copia de seguridad.

Navicat (Elaboración propia)

Autoevaluación

Una diferencia entre esta herramienta para hacer copias de seguridad y el resto de herramientas vistas en esta unidad es:

- Permite hacer copias de seguridad y restaurarlas.
- Permite elegir entre copia de tablas independientes o de toda la base de datos.
- Permite programar las copias de seguridad.
- Permite ejecutar el backup en una transacción.

Incorrecto. MySQL Workbench y PhpMyAdmin pueden efectuar copias de seguridad y restaurarlas.

No es correcto. MySQL Workbench y PhpMyAdmin permite también efectuar copias de seguridad de tablas independientes.

Correcto. Posee un planificador para programar copias de seguridad.

Respuesta incorrecta. MySQL Workbench, PhpMyAdmin pueden efectuar copias de seguridad en una transacción.

Solución

1. Incorrecto
2. Incorrecto
3. Opción correcta
4. Incorrecto

6.- Migración de datos entre S.G.B.D.

Caso práctico

Desde el taller mecánico, informan a **Juan**, jefe del departamento de Informática de **BK Sistemas Informáticos**, que antes del desarrollo de la actual base de datos, la gerente de administración, manejaba una pequeña base de datos en Access con algunos datos de los trabajadores de la empresa, y les gustaría integrarla con la nueva base de datos desarrollada en MySQL.

Por tanto, **Noiba, Naroba y Vindio** tendrán que trasladar esa base de datos a su actual SGBD que es MySQL.

[Alain Bachellier \(CC BY-NC-SA\)](#)

¿Sabemos cómo traspasar información de un SGBD a otro? Si no es así, será conveniente seguir los pasos de **Noiba, Vindio y Naroba** en este trabajo que puede resultar muy útil cuando, como en este caso, nuestro SGBD anterior se nos queda pequeño y necesitamos una herramienta más potente. Trataremos pues en este apartado de la migración de datos entre distintos SGBD.

Vamos a ver cómo podemos migrar bases de datos entre distintos SGBD con las herramientas que nos proporcionan las aplicaciones que venimos utilizando hasta ahora, para gestionar nuestras bases de datos.

Cuando tenemos una aplicación en un formato, por ejemplo **Access**, y necesitamos pasarl a otro, como en nuestro caso MySQL, será necesario convertir esos datos bien creando nuevas tablas o modificando las existentes; ya que algunos tipos de datos pueden diferir de unas bases de datos a otras, especialmente los datos de tipo fecha, numéricos, los de tipo memo (texto mayor de 256 caracteres), imágenes, etc. que pueden almacenarse de forma distinta.

Debido a que se trata de un proceso delicado no lo daremos por finalizado hasta que no estemos seguros de no haber perdido datos y nos ayudaremos para ello de herramientas fiables.

Las últimas versiones de *MySQL Workbench* incorporan el módulo **Migration Wizard**.

La utilidad **Migration Wizard** permite **migrar fácil y rápidamente**, bases de datos de Microsoft SQL Server, Microsoft Access, PostgreSQL, Sybase, SQLite, etc.

Además, podemos utilizar esta herramienta para **crear copias de bases de datos entre distintos servidores MySQL**, o **migrar datos entre diferentes versiones de MySQL**.

Puedes acceder a Migration Wizard según te indica la siguiente imagen.

Migration Wizard 1

Workbench (Elaboración propia)

Migration Wizard 2

Workbench (Elaboración propia)

Para saber más

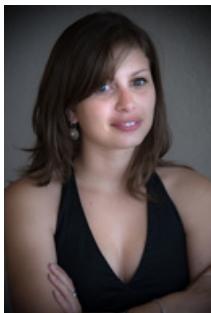
Puedes encontrar toda la información sobre **Workbench Migration Wizard** en los siguientes enlaces:

[MySQL Workbench Migration Wizard 1](#)

[MySQL Workbench Migration Wizard 2](#)

7.- Documentación de las medidas y políticas de seguridad.

Caso práctico



Noiba, Vindio y Naroba están a punto de finalizar la implantación de la base de datos para la gestión del taller mecánico. En esta última fase se ha encargado de los aspectos relativos a la integridad y a la seguridad de los datos. Una vez hecho todo este trabajo es necesario recoger las medidas y políticas de seguridad que ha diseñado en algún documento de forma que todas las personas que integran la empresa lo conozcan y lo cumplan.

Por otra parte, al tratar datos de carácter personal, todas las personas que manejan la información tendrán que tomar una serie de medidas exigidas por la **Ley de Protección de Datos**.

Alain Bachellier (CC BY-NC-SA)

Debido a que hoy en día las empresas admiten el acceso al sistema a través de Internet, es fundamental proteger los recursos y controlar el acceso al sistema. La seguridad informática consiste en garantizar que los recursos materiales, el software y los datos se utilicen para aquello para lo que se crearon.

La política de seguridad es el documento de referencia que define los objetivos de seguridad y las medidas que deben implementarse para tener la certeza de alcanzar estos objetivos.

En él se definen un número de reglas, procedimientos, controles y prácticas óptimas que aseguren un nivel de seguridad que esté a la altura de las necesidades de la organización. Este documento se debe presentar como un proyecto que incluya a todos. Una vez redactada la política de seguridad, se deben enviar a los empleados las cláusulas que los impliquen para que la política de seguridad tenga el mayor impacto posible.

Las políticas de seguridad incluyen los siguientes aspectos:

Concienciar a los usuarios sobre la importancia de la seguridad.

Seguridad lógica: a nivel de los datos de la empresa, de las aplicaciones, de los datos de los clientes, etc.

Seguridad en las telecomunicaciones, servidores, redes, etc.

Seguridad física: de infraestructuras materiales, alojamiento de servidores, estaciones de trabajo de los empleados, etc.

Seguridad informática: incluye procedimientos para administrar actualizaciones y una estrategia de copias de seguridad planificada.

Un **plan de recuperación** si se produce un accidente.

Un **sistema de documentación** actualizado.

Una vez establecidas las medidas de seguridad es necesario probar éstas para detectar posibles vulnerabilidades. Un método más eficaz para garantizar la seguridad es realizar una **auditoría de seguridad**, que consiste en que las medidas sean verificadas por un tercero, generalmente una empresa especializada.

De la misma forma en que los simulacros de incendio son fundamentales para repasar un plan de escape en caso de incendio, la práctica del plan contra desastres permite a una organización confirmar que el plan funciona, y garantizar que todas las personas involucradas sepan qué hacer.

Recomendación

En el siguiente enlace puedes consultar el reglamento de la Ley de protección de Datos.

[Ley de protección de datos](#)

Condiciones y términos de uso de los materiales

Materiales desarrollados inicialmente por el Ministerio de Educación, Cultura y Deporte y actualizados por el profesorado de la Junta de Andalucía bajo licencia Creative Commons BY-NC-SA.

Antes de cualquier uso leer detenidamente el siguiente [Aviso legal](#)

Historial de actualizaciones

Versión: 01.00.00	Fecha de actualización: 23/07/20
Versión inicial de los materiales.	

