

UNIDAD 1: INTRODUCCIÓN A LOS SISTEMAS GESTORES DE BASES DE DATOS. INSTALACIÓN Y CONFIGURACIÓN

Resultados de Aprendizaje a trabajar en esta Unidad:
RA1: Implanta sistemas gestores de bases de datos analizando sus características y ajustándose a los requerimientos del sistema. **RA2:** Configura el sistema gestor de bases de datos interpretando las especificaciones técnicas y los requisitos de explotación.
Contenidos a trabajar en esta Unidad:
- Funciones del sistema gestor de base de datos (SGBD). Componentes. Tipos. - Arquitectura del sistema gestor de base de datos. - Sistemas gestores de base de datos comerciales y libres. - Instalación y configuración de un SGBD. Parámetros relevantes. - Estructura del diccionario de datos. - Ficheros log.

Contenidos de la Unidad:

1. SISTEMAS GESTORES DE BASES DE DATOS 2
2. ARQUITECTURA DE MySQL 4
3. INSTALACIÓN DE MySQL 4
4. CONFIGURACIÓN DEL SERVIDOR 5
5. TIPOS DE SERVIDOR 6
6. MOTORES DE ALMACENAMIENTO 7
7. FICHEROS Y CONFIGURACIÓN DEL SERVIDOR 8
8. VARIABLES DEL SERVIDOR 9
9. LOGS 11
10. BIBLIOGRAFÍA 14

1. SISTEMAS GESTORES DE BASES DE DATOS

Una base de datos es un conjunto organizado de datos que permite almacenar y consultar información.

Un sistema gestor de base de datos (SGBD) es el software encargado de crear, almacenar, organizar, gestionar y proporcionar acceso a las bases de datos.

Un SGBD no sirve únicamente para ejecutar consultas SQL. Entre sus principales funciones encontramos:

- Almacenamiento de datos: se encarga de guardar los datos de forma organizada y gestionar cómo se almacenan físicamente.
- Consulta y modificación de datos: permite realizar operaciones mediante SQL.
- Gestión de transacciones: permite agrupar operaciones que deben realizarse de forma coherente.
- Control de acceso: permite crear usuarios y establecer qué puede hacer cada uno.
- Concurrencia: coordinar accesos simultáneos para evitar inconsistencias en los datos.
- Seguridad: incluye aspectos como autenticación, autorización, privilegios, cifrado, auditoría, registro de actividad…
- Monitorización: el administrador debe poder conocer cómo está funcionado el servidor (utilización de recursos, conexiones, consultas, errores, rendimiento…).

1.1. Tipos y ejemplos de SGBD

Existen muchos SGBD y no todos tienen las mismas características. Algunos ejemplos conocidos son:

SGBD	Fabricante/proyecto	Características
MySQL	Oracle	Relacional, muy utilizado en aplicaciones web
PostgreSQL	Comunidad	Relacional/objeto-relacional, muy completo
Oracle Database	Oracle	Empresarial, amplio conjunto de funcionalidades
Microsoft SQL Server	Microsoft	Relacional, muy integrado con el ecosistema Microsoft
MariaDB	Comunidad	Derivado de MySQL
SQLite	Proyecto SQLite	SGBD embebido, sin servidor independiente

No existe un SGBD universalmente adecuado para todas las situaciones. La elección dependerá de factores como:

- Características necesarias
- Tamaño de la base de datos
- Número de usuarios
- Rendimiento
- Sistema operativo
- Compatibilidad con las aplicaciones
- Disponibilidad
- Seguridad
- Facilidad de administración
- Conocimientos del equipo
- Licencia y coste
- Soporte técnico

1.2. ¿Por qué vamos a trabajar con MySQL?

A lo largo del curso, vamos a trabajar con MySQL como SGBD de referencia.

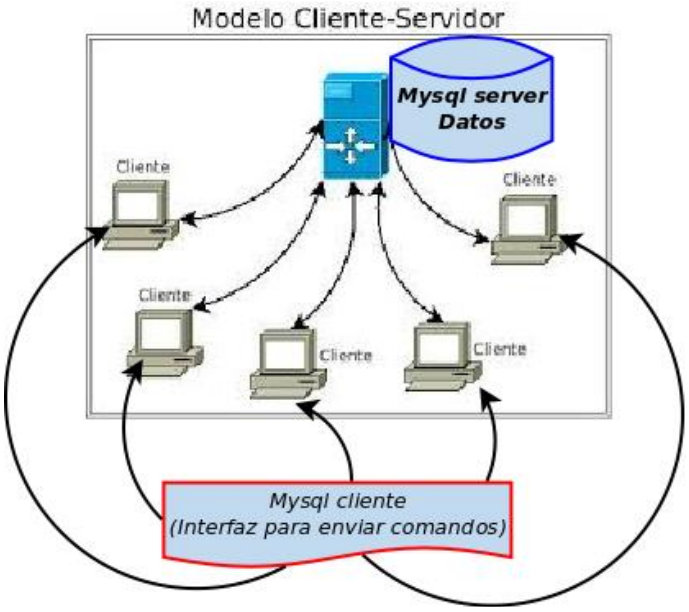
Esta elección no significa que MySQL sea el mejor SGBD en cualquier situación, pero resulta especialmente interesante para este módulo porque, no siendo el SGBD más complejo, permite trabajar con muchos conceptos que debe conocer un administrados como son:

- Instalación del servidor
- Configuración
- Usuarios y permisos
- Bases de datos
- Motores de almacenamiento
- Variables del servidor
- Logs
- Monitorización
- Importación y exportación
- Copias de seguridad
- Recuperación
- Rendimiento
- Herramientas gráficas
- Administración mediante consola

Como ya iremos viendo, MySQL nos permite trabajar tanto desde línea de comandos, como mediante herramientas gráficas como MySQL Workbench.

2. ARQUITECTURA DE MySQL

MySQL funciona siguiendo un modelo cliente – servidor. Por un lado, está la parte del servidor, donde realmente se guardan los datos, y por otro lado está la parte del cliente, que es un software que nos ofrece una interfaz de comandos o gráfica, y que nos permite conectarnos al servidor y gestionar las bases de datos.



En este caso, MySQL Server es el servidor que va a gestionar las bases de datos, y la línea de comandos, MySQL Workbench, o aplicaciones Java, PHP, Python, etc. que utilicen la base de datos, son los diferentes clientes que se conectan al servidor.

3. INSTALACIÓN DE MySQL

Para trabajar con MySQL necesitaremos instalar diferentes componentes, centrándonos principalmente en los siguientes:

3.1. MySQL Server

Es el componente principal de MySQL. Contiene el servicio que gestiona las bases de datos y atiende las peticiones de los clientes. El servidor es el encargado de:

- Gestionar las bases de datos
- Almacenar y recuperar los datos
- Procesar las consultas SQL
- Gestionar usuarios y permisos
- Controlar las conexiones de los clientes
- Gestionar las transacciones
- Utilizar los motores de almacenamiento
- Generar registros de actividad y errores

3.2. MySQL Workbench

Es la herramienta gráfica para trabajar con MySQL. Funciona como un cliente que se conecta al servidor, pudiendo así ejecutar consultas SQL, y diseñar modelos de bases de datos.

Entre sus múltiples posibilidades, nos va a permitir:

- Ejecutar sentencias SQL
- Consultar y modificar bases de datos
- Crear tablas
- Diseñar modelos de bases de datos
- Importar datos
- Consultar información del servidor

Sin embargo, es importante recordar que MySQL Workbench es únicamente el cliente gráfico, pero sigue siendo el servidor el encargado de ejecutar las operaciones.

3.3. MySQL Shell

Es el cliente que nos permite conectarnos al servidor desde una terminal. Es especialmente interesante en administración de sistemas porque en muchos servidores no se dispone de interfaz gráfica.

La utilización de la consola permite realizar tareas de administración y automatización mediante comandos y scripts. A lo largo del módulo utilizaremos tanto herramientas gráficas, como de línea de comandos.

3.4. MySQL Notifier

Es una herramienta específica para Windows que facilita la supervisión de determinados componentes y servicios de MySQL.

Permite, entre otras cosas, controlar el estado de los servicios de MySQL y acceder rápidamente a determinadas acciones relacionadas con ellos.

4. CONFIGURACIÓN DEL SERVIDOR

Una vez instalado MySQL Server, el servidor necesita una serie de parámetros que determinan cómo debe funcionar.

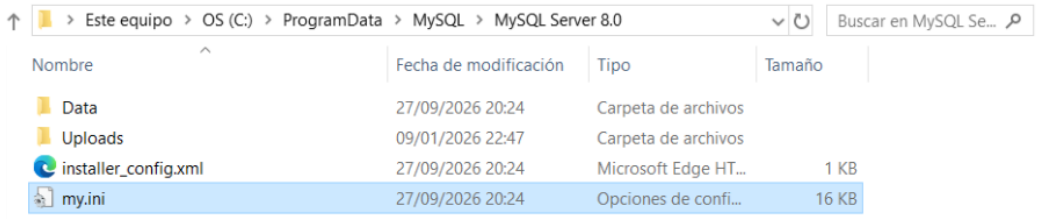
Entre otras cuestiones, podemos configurar aspectos relacionados con:

- Los recursos del equipo que utilizará MySQL
- Los motores de almacenamiento
- Las conexiones necesarias

- Los registros o logs
- Opciones de funcionamiento y parámetros de rendimiento

Una gran parte importante de esta configuración se encuentra en el fichero de configuración de MySQL.

En Windows, el fichero utilizado es **my.ini** mientras que en sistemas Linux es habitual **my.cnf**.



Si el instalador de MySQL detecta en el equipo una configuración ya existente porque ya hemos realizado una instalación anterior, el instalador nos ofrecerá diferentes opciones de mantenimiento:

- Instalar otra versión del servidor: en el caso de que queramos realizar una nueva instalación con una versión diferente.
- Reconfigurar el servidor: modifica la configuración existente. El fichero ya existente se conserva como coipa de seguridad mediante un nombre que incluye la fecha y hora de creación, de forma que podamos guardar también la configuración anterior.
- Eliminar la instancia: elimina la instancia de MYSQL ya existente, incluyendo la configuración asociada.

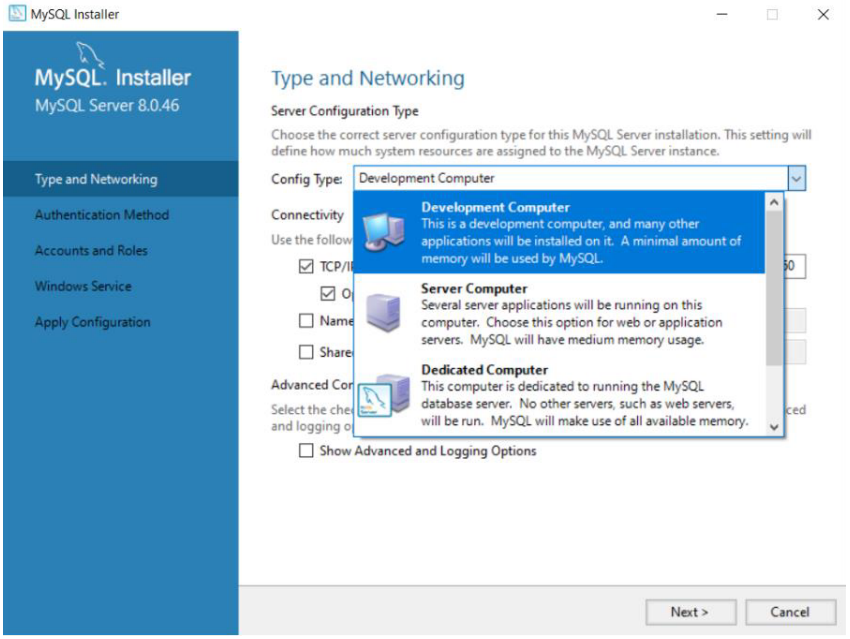
5. TIPOS DE SERVIDOR

La configuración del servidor debe adaptarse a los recursos disponibles y a la función que desempeña el equipo. El tipo de servidor que escojamos determina la configuración del servidor en los que respecta a memoria, disco y procesador.

Podemos encontrar diferentes tipos de servidor:

- **Developer machine (equipo de desarrollo)**: ordenadores de escritorio orientados a uso personal, en los que instalamos MySQL. Se asume que estarán ejecutando otras aplicaciones además del servidor, por lo que el servidor MySQL se configura para utilizar una cantidad mínima de recursos del sistema.
- **Server machine (servidor)**: es la opción para un equipo servidor que ejecuta MySQL junto con otros servicios como un servidor web, FTP... Se configura para utilizar una cantidad moderada de recursos ya que debe compartirlos con otros servicios.

- **Dedicated MySQL Server Machine (Servidor MySQL dedicado)**: es la opción para un equipo en el que se ejecuta exclusivamente el servidor MySQL. Se configura para utilizar todos los recursos de sistema, al no tener que compartirlos con otras aplicaciones.



En el caso de las prácticas de aula, tendremos el primero de los tipos al estar trabajando sobre un ordenador personal.

6. MOTORES DE ALMACENAMIENTO

Un motor de almacenamiento determina cómo MySQL gestiona y almacena las tablas y determinadas operaciones sobre ellas. MySQL permite utilizar diferentes motores de almacenamiento.

6.1. InnoDB

En las versiones más actuales de MySQL, InnoDB es el motor predeterminado para las tablas nuevas.

Es un motor de base de datos transaccional ACID (Atomicidad, Coherencia, Aislamiento y Durabilidad), implementando también un sistema de control de concurrencia que permite tener lecturas que no bloquean a las escrituras y viceversa.

Incrementa la eficacia en entornos con muchas escrituras y lecturas.

6.2. MyISAM

Es otro motor de almacenamiento, muy utilizado en las versiones algo más antiguas de MySQL (anteriores a la 5.5.). Conviene conocerlo ya que aparece con frecuencia en la documentación del SGBD.

7. FICHEROS Y CONFIGURACIÓN DEL SERVIDOR

Al configurar un SGBD, si queremos que algunas opciones sean permanentes, lo normal es hacer que el servidor al iniciarse lea estas opciones desde ficheros de configuración. Estos proporcionan una forma conveniente de especificar opciones comúnmente usadas.

En Windows, MySQL utiliza un fichero de opciones denominado normalmente my.ini mientras que en Linux suele ser my.cnf.

Estos ficheros permiten incluir elementos como:

- Comentarios precedidos por #
- Opciones para programas o grupos, usando corchetes
- Opciones tipo opción = valor
- Opciones sin valor asociado

A continuación se encuentra un ejemplo de fichero de configuración de MySQL:

```
[client]
port=3306
password="guara"

[mysqld]
port=3306
key_buffer_size=16M
max_allowed_packet=8M

[mysqldump]
quick

[mysqladmin]
force
```

Podemos ver que en varias líneas aparecen elementos entre corchetes ([ ]). Estas líneas dividen el código en secciones, donde **cada sección configura un programa diferente**.

En este caso, la primera sección es **[client]** por lo que las opciones que aparecen debajo se aplican a los programas cliente de MySQL que utilicen este fichero de configuración. En el ejemplo vemos como el puerto asignado para realizar la conexión con el servidor es el 3306 (puerto por defecto), y la contraseña que utilizará el cliente al conectarse. Esta última línea se añade a modo de ejemplo de una mala práctica, ya que almacenar una contraseña directamente en un fichero de configuración puede suponer un riesgo de seguridad.

La segunda sección **[mysqld]** especifica la configuración del servidor MySQL. En ella se indica el puerto en el que debe escuchar las conexiones, el tamaño del búfer utilizado para los índices de las tablas y el tamaño máximo permitido para un paquete de comunicación cliente – servidor.

La sección **[mysqldump]** configura la herramienta con ese mismo nombre, que es la utilizada para realizar los volcados (dumps) de bases de datos. En este caso vemos que no aparece un parámetro clave – valor, sino simplemente un modo de trabajo: quick. Este modo hace que se recuperen los registros de las tablas de forma progresiva.

Por último, **[mysqladmin]** configura la herramienta de línea de comandos que permite realizar determinadas tareas de administración. En este caso la opción force hace que se continue ante determinados errores, en lugar de detener la operación que se está llevando a cabo.

8. VARIABLES DEL SERVIDOR

MySQL dispone de numerosas variables que permiten consultar y, en determinados casos, modificar diferentes aspectos de su funcionamiento.

Estas variables están relacionadas con la configuración general del servidor, la memoria, las caches, la red, características concretas del servidor, el estado y funcionamiento del propio servidor...

El administrador del sistema puede consultar estos valores y, cuando la variable lo permita, modificarlos para adaptar el servidor a las necesidades de las aplicaciones.

8.1. Variables del sistema del servidor

Las variables del sistema representan parámetros que determinan el funcionamiento del servidor MySQL.

El fichero de configuración my.ini contiene parámetros que determinan el funcionamiento del servidor. Por ese motivo, una de las tareas del administrador consistente en conocer qué variables existen y qué efecto tiene modificar sus valores.

La documentación de MySQL proporciona información sobre las diferentes opciones y variables disponibles. Además, también se puede obtener información desde la propia línea de comandos (cmd):

```
mysqld --verbose --help
```

Podemos clasificar las variables en:

- Generales
- Relacionadas con la memoria y la caché
- Tipo have (indican si tiene una característica o no)
- Relacionadas con la red

8.2. Variables de estado del servidor

El servidor mantiene muchas variables de estado que proveen de información sobre su estado, por lo que conviene saber consultarlas.

Para mostrar las variables del servidor y sus valores:

```
SHOW VARIABLES;
```

Como la lista de variables a mostrar puede ser muy extensa, podemos filtrar los resultados:

```
SHOW VARIABLES LIKE 'patron';
```

Ejemplos:

```
SHOW VARIABLES LIKE 'port';

SHOW VARIABLES LIKE '%buffer%';
```

A continuación se enumeran diferentes comandos interesantes para obtener los valores de las variables. En casi todas las opciones se pueden emplear filtros con `LIKE`:

- `SHOW VARIABLES`, muestra las variables
- `SHOW STATUS`, muestra los valores de las variables de estado
- `SHOW BINARY LOGS`, muestra un listado de los ficheros de log binario en el servidor, así como su tamaño
- `SHOW BINLOG EVENTS`, muestra el contenido del fichero binario indicado. Debemos tener cuidado porque el fichero puede ser muy grande y saturar recursos del sistema
- `SHOW ENGINE`, muestra información sobre logs o estado del motor de almacenamiento en cuestión (BDB, INNODB, NDB y NDBCLUSTER)
- `SHOW ENGINES`, muestra información sobre el tipo de tablas o motores soportados por el servidor
- `SHOW ERRORS`, muestra los mensajes de error generados en el servidor
- `SHOW WARNINGS`, muestra los warnings (advertencias) del servidor
- `SHOW OPEN TABLES`, muestra tablas abiertas de la base de datos indicada
- `SHOW [FULL] PROCESSLIST`, muestra los procesos activos en el servidor. Si omitimos la opción FULL, sólo se muestran los 100 primeros caracteres del campo Info
- `SHOW STATUS`, muestra el estado de las variables de estado del servidor según sean globales o de sesión
- `SHOW VARIABLES`, muestra variables globales o de sesión del servidor

8.3. Modificación de variables

El comando SET nos permite modificar o crear nuevas variables en tiempo de ejecución. Sólo es válido para variables dinámicas o para crear nuevas variables de usuario.

La asignación de variables se asocia con un valor o expresión numérica o de tipo cadena especificando si es de tipo global (`@@global`) o de sesión (`@@session`).

Los cambios que se realicen sobre una variable ya sea global o de sesión, no afectarán a las correspondientes variables para usuarios que ya estén conectados. Solamente a sesiones de usuarios que se conecten después del cambio.

Los cambios de cualquier tipo de variable permanecen hasta que la sesión termina o se reinicia el servidor. De modo que si queremos un cambio permanente debemos incluir los valores en ficheros de opciones o como opciones de inicio del servidor.

A diferencia del fichero de configuración, con SET no podemos utilizar los sufijos, así si queremos asignarle 10 MB a la variable `sort_buffer_size` sería `SET sort_buffer_size = 10 * 1024 * 1024`

9. LOGS

Un fichero de registro o log, almacena información sobre los eventos producidos en el SGBD, tanto en su instalación, como en su explotación y administración. Por tanto, estos ficheros son esenciales para conocer el comportamiento del SGBD y ayudan al administrador a averiguar las causas de posibles malos funcionamientos, así como a auditar todas las operaciones que los clientes realicen.

Generalmente en los logs podemos distinguir tres tipos de anotaciones:

- **Información de eventos, alertas** y demás acciones que se lleven a cabo sobre las bases de datos existentes.
- **Estado general de los parámetros** de la instancia o sus bases de datos, así como posibles consejos de mejora de funcionamiento en caso de que alguno sobrepase un determinado umbral de funcionamiento programado.
- **Transacciones y operaciones que afecten a los datos o a la estructura** de los mismos, es decir, sentencias DMS y DDL respectivamente.

Existen diferentes tipos de ficheros de registros: registros de errores, registros generales de consultas, registros de consultas lentas y registros binarios.

- **Registro de errores (error log):**

El fichero de Registros de Errores de MySQL muestra información sobre los sucesos anormales que se han producido a nivel de sistema en el servidor de base de datos, por ejemplo, que al arrancar el servidor no encuentre el fichero de configuración.

La ubicación de este fichero se puede indicar mediante la opción `--log-file = nombre_fichero` al arrancar el servidor. También se puede especificar la ruta específica de este fichero desde el fichero de configuración. Si no se especifica un nombre, por defecto se escribe el fichero de errores en el directorio de datos con el nombre nombre_servidor.err o nombre_equipo.err.

En una instalación estándar, debería estar en C:\ProgramData\MySQL\MySQL Server 8.0\Data\.

Podemos averiguar la ubicación exacta del fichero en nuestro sistema desde la consola de MySQL:

```
SHOW VARIABLES LIKE 'log_error';
```

Que devolverá algo parecido a esto, con el nombre especifico del fichero en nuestro sistema.

```
mysql> show variables like 'log_error';
+---------------+----------------------+
| Variable_name | Value                |
+---------------+----------------------+
| log_error     | .\DESKTOP-7V49EEA.err |
+---------------+----------------------+
1 row in set (0.05 sec)
```

Con el comando `FLUSH LOGS`, el registro de errores es renombrado con el sufijo -old, y se crea un nuevo archivo de registro.

- **Registro general de consultas (general query log)**

Almacenan todas las consultas solicitadas a las bases de datos. Este modo por defecto no está activado puesto que activarlo repercute en el rendimiento del servidor de un modo muy negativo.

Si se desea activar hay que añadir al fichero de configuración las siguientes líneas, dentro de la configuración del servidor:

```
general_log=ON

general_log_file="C:/ProgramData/MySQL/MySQL Server 8.0/Data/general.log"
```

Otra opción de activarlo/desactivarlo es modificar el valor de las variables en tiempo de ejecución desde la consola MySQL:

```
SET GLOBAL general_log = 'ON';

SET GLOBAL general_log = 'OFF;
```

El servidor mysqld escribe las sentencias en el registro de consultas en el orden en que las recibe. Este orden puede ser diferente al de ejecución.

- **Registro de consultas lentas (low query log)**

Almacenan las consultas que pueden ser consideradas como pesadas en tiempo de ejecución. Este tipo de registro tampoco está activado por defecto y se puede configurar añadiendo los siguientes parámetros al fichero de configuración de MySQL:

```
slow_query_log=ON
slow_query_log_file="C:/ProgramData/MySQL/MySQL Server 8.0/Data/slow.log"
long_query_time=10
```

En este log solo se registrarán aquellas consultas que superen el umbral establecido en `long_query_time`.

- **Log binario (binary log)**

El registro binario o log binario registra eventos relacionados con cambios en los datos y en la estructura de las bases de datos, que pueden ser relevantes para la replicación y la recuperación. También contiene información sobre cuánto ha tardado cada sentencia que actualizó la base de datos.

El registro binario sirve, por ejemplo, para replicar bases de datos o para restaurar una copia de seguridad del sistema en cualquier punto anterior.

Para configurar el log binario, habrá que añadir la siguiente línea en el fichero de configuración:

```
log-bin="C:/ProgramData/MySQL/MySQL Server 8.0/Data/mysql-bin"
```

9.1. Mantenimiento de ficheros de registro (logs)

Estos archivos se deben limpiar regularmente para asegurarse de que no ocupan demasiado espacio. Al utilizar MySQL con los logs activados deberíamos hacer copias de seguridad de estos ficheros de vez en cuando, eliminarlos de la carpeta de datos del servidor, y decirme a MySQL que comience a registrar en archivos nuevos.

Podemos forzar al servidor para que comience a utilizar archivos de registro nuevos con el comando:

```
FLUSH LOGS;
```

Es recomendable almacenar las copias de seguridad de los logs ya que pueden ser una gran fuente de información diversa acerca del funcionamiento de nuestro servidor, el tipo de consultas, etc. que puede ser información muy útil para su optimización.

10. BIBLIOGRAFÍA

Documentación oficial de MySQL. Disponible en: https://dev.mysql.com/doc/

Documentación oficial de MySQL Server Administration. Disponible en: https://dev.mysql.com/doc/refman/8.0/en/server-administration.html