

Servidores de bases de datos

Objetivos

Los objetivos son:

- Analizar diferentes motores de base de datos.
- Instalar el *driver* ODBC para acceder a la BD.
- Conocer herramientas para administrar las bases de datos.
- Conocer las operaciones básicas que se pueden hacer con SQL.

Contenidos

- 1** [Análisis de bases de datos.](#)
 - 2** [Drivers ODBC.](#)
 - 3** [Instalación y configuración de PhpMyadmin.](#)
 - 4** [Herramientas para administrar una BD.](#)
 - 5** [Lenguaje SQL.](#)
 - 6** [Conexión de PHP con MySQL.](#)
-

Análisis de bases de datos

El objetivo de este módulo es conocer diferentes motores de bases de datos.

Bases de datos Open Source

Debido a las constantes actualizaciones y mejoras que van apareciendo en los distintos productos deberíamos tomar este análisis como una introducción a las bases de datos. Analizaremos tres ejemplos de bases de datos:

- [MySQL](#)
- [PostgreSQL](#)
- [Interbase](#)
- [Conclusiones](#)



MySQL

Los orígenes de Mysql datan de 1979, como una herramienta de BD llamada UNIREG, creada por Michael Widenius para la empresa sueca TcX. El 1994, TcX necesitaba un servidor SQL para el desarrollo de aplicaciones web. Probaron varios servidores comerciales para las inmensas tablas de TcX, pero todo lo que encontraron fue demasiado lento. Incluso analizaron mSQL, pero no implementaba alguna de las características que TcX necesitaba. Se encargó a Widenius el desarrollo de un servidor de bases de datos, con una interface parecido a mSQL, ya que en aquel momento estaban disponibles algunas herramientas gratuitas para mSQL que se podían adaptar con poco esfuerzo al nuevo servidor.

En 1995, David Axmark de Detron HB animó a TcX para que publicase MySQL en Internet. MySQL no es un proyecto Open Source realmente, ya que bajo ciertas condiciones es necesaria una [licencia](#). Pese a todo, MySQL es uno de los preferidos de la comunidad Open Source.

MySQL no se limita a la comunidad Open Source, también es portable a otros sistemas operativos comerciales, como Windows (NT/2000/95/98/ME/XP), Solaris y otros. Uno de los éxitos de su expansión ha sido la inclusión de esta base de datos en casi todas las distribuciones de Linux, además de estar disponible en casi todas las plataformas *hardware* y sistemas operativos. El gran propulsor de esta BD ha sido el lenguaje interpretado para web PHP, que ha hecho que Mysql creciera al mismo tiempo que lo ha hecho la comunidad PHP.



Características de MySQL

Es posiblemente un de los motores de BD más rápidos que se pueden encontrar, además de consumir pocos recursos de la máquina en la que esta instalada.

Es fácil de instalar y administrar frente a otros productos del mercado. Dispone de infinidad de

utilidades, manuales y documentación que la comunidad de usuarios se ha encargado de realizar desinteresadamente.

La gestión de la base de datos utiliza SQL (Structured Query Language), además de utilizar protocolos de comunicación de bases de datos desarrollados por Microsoft, mediante ODBC o ADO.

Ofrece una gran capacidad de conectividad, ya que muchos clientes pueden conectar simultáneamente al servidor, y pueden utilizar diferentes bases de datos a la vez. Se puede acceder de forma interactiva utilizando diferentes interfaces que permiten introducir y visualizar las consultas. Además dispone de gran variedad de interfaces de programación para su acceso (C, Perl, Java, PHP, Python...).

Se trata de una BD preparada para el trabajo en red y se puede acceder desde cualquier sitio de internet como resultado de su gran habilidad en materia de control de permisos y seguridad de acceso.

La versión 3.23 no cumplía la integridad referencial, con posibles problemas de inconsistencia de datos, y tampoco tenía transacciones. En versiones actuales la mayoría de las deficiencias que tenía han sido subsanadas. Aunque todavía no cumple el 100 % de los requisitos de una verdadera RDBMS, ofrece otras características que hace que sea considerada como idónea para su uso.

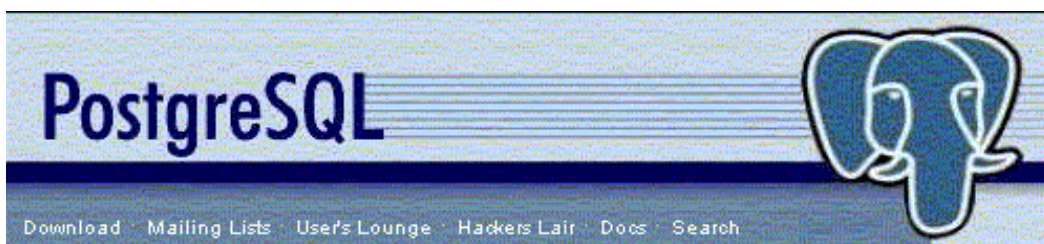


PostgreSQL

Se puede afirmar PostgreSQL es la segunda BD más popular implementada en el mercado Linux. PostgreSQL se diseñó como una base de datos orientada a objetos, es decir, una ORDBMS, con la cual el concepto de tabla es sustituido por objeto, y se permite crear nuevos tipos de datos, hacer herencias de objetos, etc.

PostgreSQL es, sin lugar a dudas, la base de datos más implementada por los expertos, dispone de lo que MySQL le falta o faltaba, pero también carece de algunas de las características que posee MySQL. PostgreSQL tiene transacciones, integridad referencial, vistas y multitud de funcionalidades, por contra es un motor más lento y más pesado.

La versión 7 incorpora MVCC (Multiversion Concurrency Control), gracias a la cual los bloqueos actúan solo en la sesión del cliente, y no en la de otros clientes.



PostgreSQL es una RDBMS poco viable para un servidor de internet, pero sí para un servidor con pocas conexiones que requieren mucha funcionalidad, como un comercio on-line. Para operaciones sencillas y muchas conexiones, el sistema resulta poco recomendable, al tener en cuenta la velocidad y recursos utilizados.



InterBase

Borland, siguiendo la política de liberar las herramientas de desarrollo, liberó InterBase. Se

trata de una buena BD, con 17 años de antigüedad dentro del sector de las bases de datos comerciales. InterBase ofrece herramientas de pago para el mundo comercial.

Es una de las primeras BD Open Source que es compatible con SQL92 en el nivel de entrada (MySQL, PostgreSQL y mSQL no lo son). Esto significa que podemos adaptarla a cualquier aplicación que funcione sobre una BD compatible con SQL92, como Oracle, DB2, o Informix.

InterBase es un producto muy profesional, preparado para cualquier proyecto que necesite una BD fiable. Tiene la mayoría de la funcionalidades de una base de datos comercial, no consume demasiados recursos, y tiene una velocidad próxima a MySQL, con algunas funcionalidades de PostgreSQL.

Pero, Borland no ha liberado las herramientas más avanzadas, y la comunidad de usuarios es muy pequeña, comparando con los otros motores de BD.



Conclusiones

Hemos visto una introducción de las posibles alternativas que tenemos, y según el proyecto que tengamos que realizar, la posible solución variará, ya que necesitaremos priorizar unas características sobre otras que deberemos sacrificar.

Dependiendo de las fuentes que busquemos obtendremos una respuesta u otra, hay verdaderas batallas por establecer cuál de ellas es mejor, pero la realidad es que no hay nada perfecto, y dependiendo de nuestras habilidades y necesidades deberemos elegir uno u otro.

Se trata de buscar un motor de base de datos que pueda integrar un servidor web Apache, y accesible desde PHP, para la creación de entornos web dinámicos. Ante esto, la respuesta es mayoritaria (MySQL, APACHE, PHP) es la combinación más acertada, fiable, estable, y documentada que tenemos en la actualidad.



Instalación y conexión a una BD.

Veremos la forma de obtener e instalar el *driver* ODBC, para acceder a la base de datos utilizando este tipo de conexión.

Una vez que el servidor web a solicitado al intérprete de php la ejecución de un determinado script, éste se puede encontrar con sentencias que pretenden manipular los datos de una base de datos. La forma en que el intérprete de PHP y el gestor de bases de datos pueden conectar es doble: ODBC (Open DataBase Connectivity) y funciones nativas. La primera consiste en una API estándar que oculta detalles de la conexión. Para poder utilizarla se hace necesario instalar o activar el módulo encargado de su gestión y realizar las llamadas correspondientes desde nuestro programa.

Obtención e instalación de MyODBC (Windows)

En la siguiente página de MySQL podemos encontrar el último enlace al *driver* ODBC estable para acceder a MySQL.

<http://www.mysql.com/products/>

Una vez tenemos el fichero de instalación *.exe, hacer doble clic para ejecutar el programa y se mostrará la siguiente pantalla:



Figura 5.2.1. Instalación de ODBC Windows.



A partir de aquí, tan solo debemos pulsar el botón **Siguiente**, si todo funciona correctamente, aparecerá la siguiente pantalla:



Figura 5.2.2. Instalación de ODBC Windows.

La conexión ODBC está lista para ser utilizada. Para comprobarlo se puede acceder al panel de control de los controladores ODBC instalados. En la siguiente tabla se muestra donde realizar estas comprobaciones dependiendo del sistema operativo utilizado.

Sistema	Ubicación de los controladores ODBC
Windows 95/98	Inicio / Configuración / panel de control / Controladores ODBC.
Windows 2000 Prof/Serv y XP	Inicio / Configuración / panel de control / Herramientas administrativas / Orígenes de datos (ODBC).

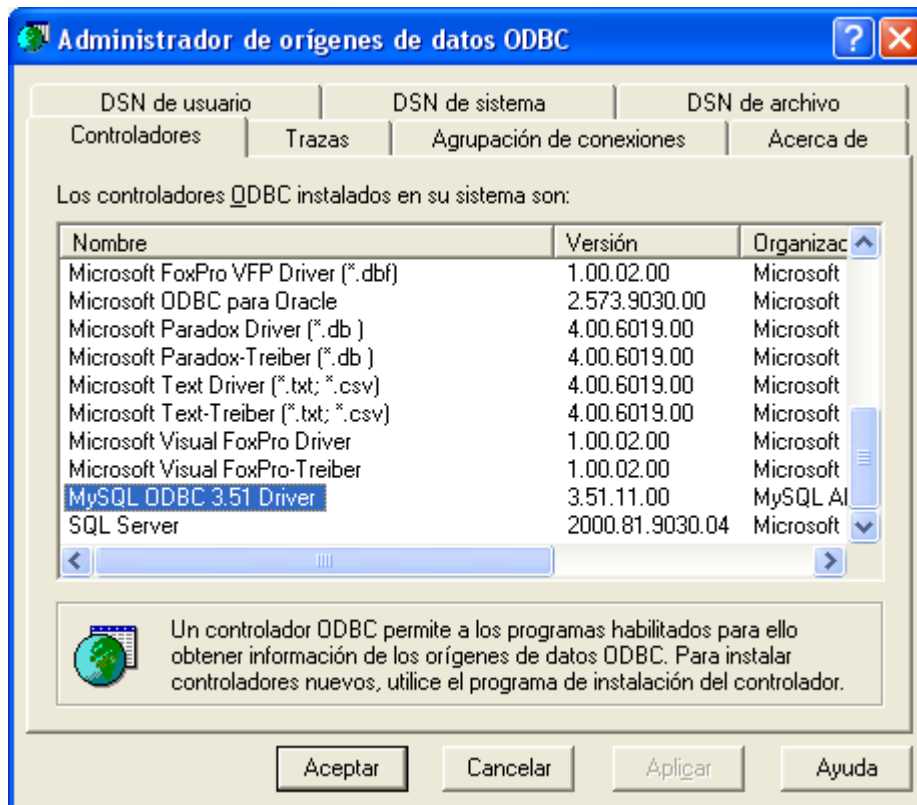


Figura 5.2.3. Administrador de controladores ODBC.

Debemos seleccionar la pestaña de DSN de usuario, y crear uno nuevo. Lo primero que se solicita es el controlador para establecer un origen de los datos.

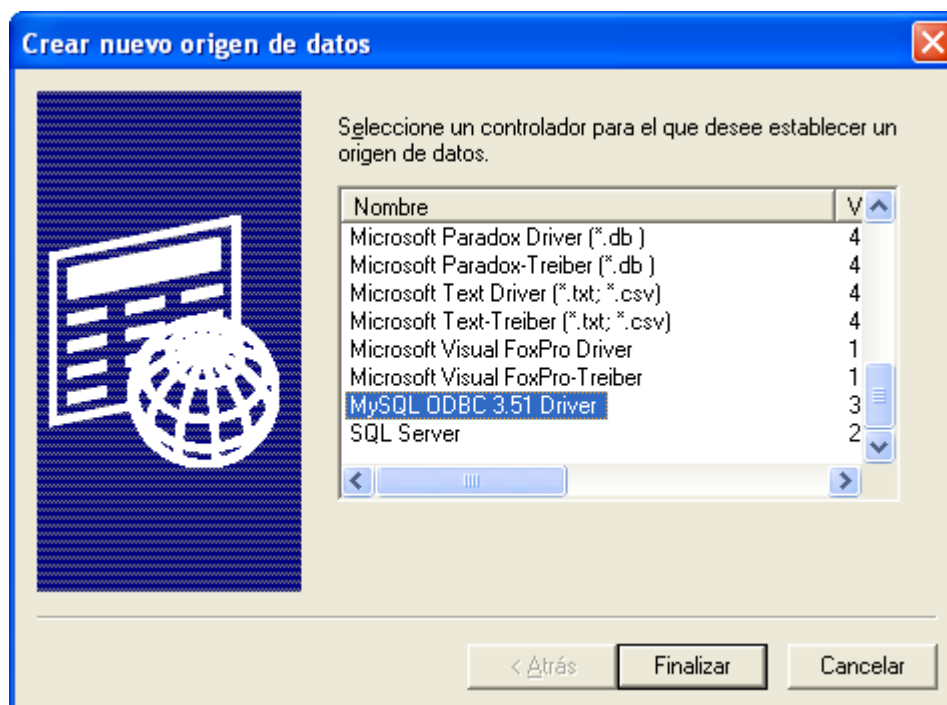


Figura 5.2.4. Nuevo DSN de usuario.

Data Source Name especifica el nombre que queremos dar al ODBC, en **Server** se indica el nombre del servidor de bases de datos (generalmente localhost). En el campo **Database** podemos seleccionar la base de datos a la que queremos establecer la conexión. Los



campos **User** y **Password** en principio no se deben modificar, a no ser que en la instalación se hayan modificado.

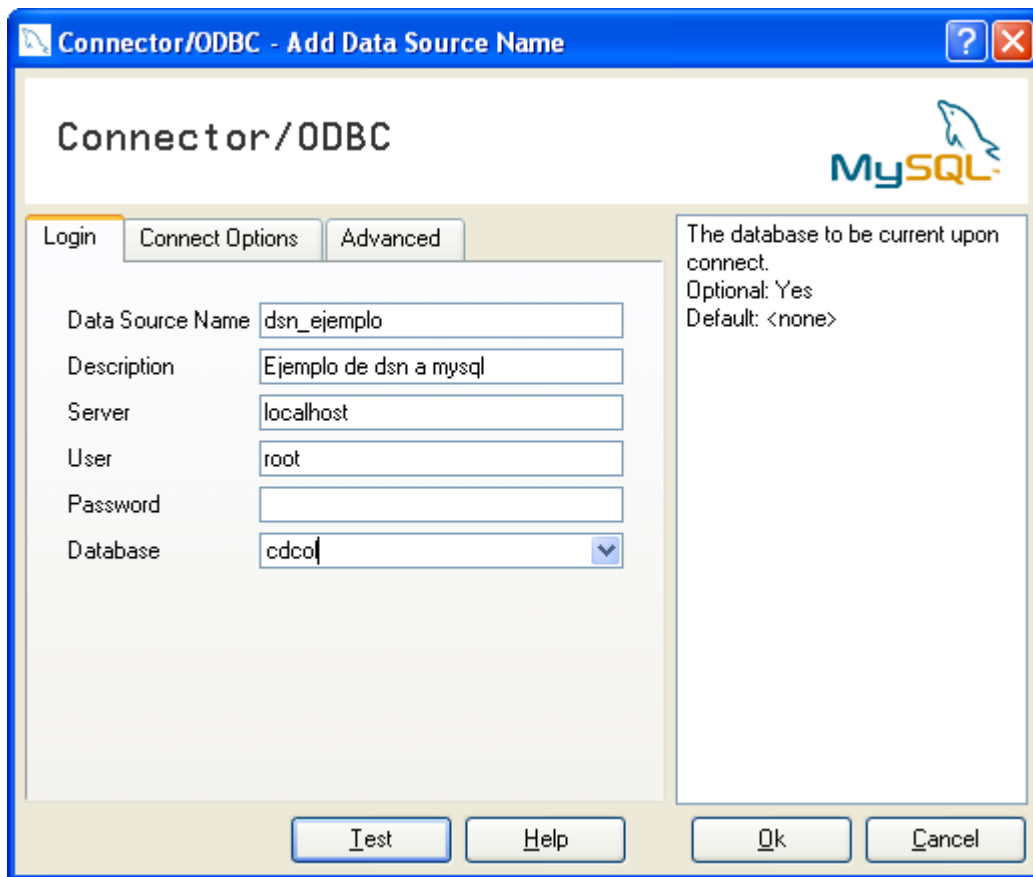


Figura 5.2.5. Parámetros del nuevo DSN.

Pulsar el botón **Test** y si la conexión es correcta, se abrirá una ventana como la siguiente:



Figura 5.2.6. Conexión correcta.

Una vez terminada la creación del nuevo DSN de usuario, utilizando el controlador ODBC para MySQL, aparecerá la siguiente pantalla:

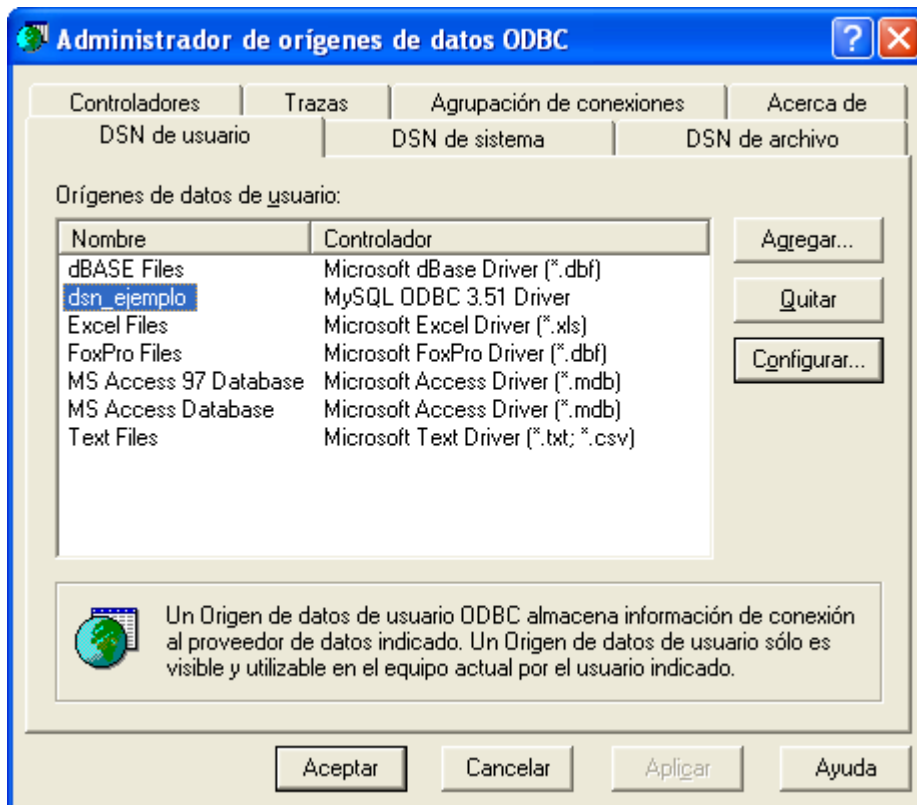


Figura 5.2.7. Administrador de orígenes de datos ODBC.



Hay que tener especial cuidado con la seguridad de las bases de datos, si tenemos un servidor en el que se puede ejecutar un administrador de bases de datos, o un usuario puede subir uno de ellos, probablemente conozca los usuarios por defecto que se instalan. Con el usuario por defecto 'root' podrá: ver, modificar o borrar todas nuestras bases de datos. Por lo que cada base de datos debería tener un usuario y contraseña diferente, estos datos se deberían cambiar en la definición del DSN.



Obtención e instalación de MyODBC (Linux)

En la siguiente página de MySQL podemos encontrar el último enlace al *driver* ODBC estable para acceder a MySQL. Acceder al apartado Linux, seleccionar la plataforma que nos interesa (debian) y bajar el fichero **MYODBC-n.nn.n_pc_linux_i386.tar.gz**

<http://www.mysql.com/products/>

Si descomprimos el paquete que hemos bajado, deberíamos encontrar un fichero llamado **README**, en el que se detallan los pasos que debemos seguir para la instalación del driver ODBC bajo linux. Pero en realidad nos recomienda que realicemos otro tipo de instalación o que busquemos más información en la página oficial.

De la misma forma que hicimos para la instalación de Apache, Mysql y Php, para el driver ODBC y unixODBC utilizaremos el gestor de paquetes synaptic. Seleccionaremos los siguientes paquetes:

- libmyodbc.
- php4-odbc.
- unixodbc.
- unixodbc-bin.
- unixodbc-dev.

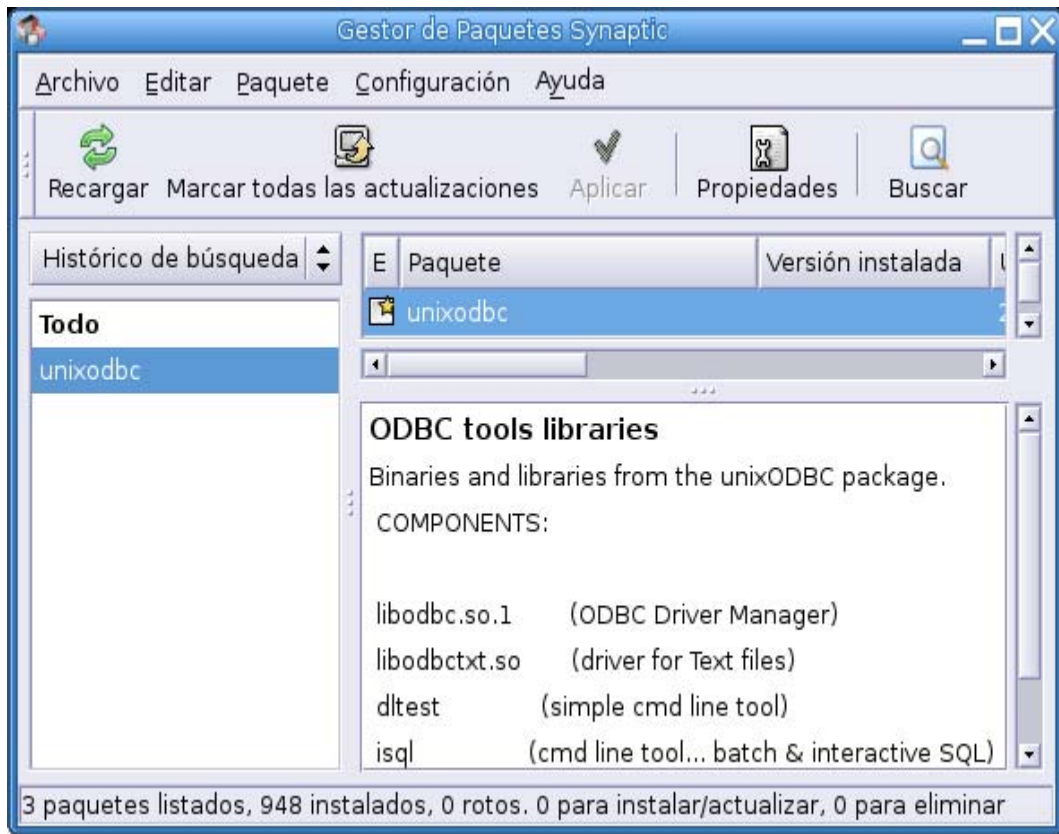


Figura 5.2.8 Instalación de UNIXODBC.



Con la herramienta gráfica de administración **unixODBC**, podemos acceder a las bases de datos utilizando una interface estándar de acceso. Muchas distribuciones Linux (RedHat, Suse, United Linux y plataformas como Solaris, AIX, HP/UX), incorporan esta herramienta.



Con este paquete viene una utilidad llamada ODBCConfig, muy parecida a la de Windows con la que podremos crear el DSN. Se encuentra en el directorio /usr/bin y se debe ejecutar como root, el comando es "**ODBCConfig**".

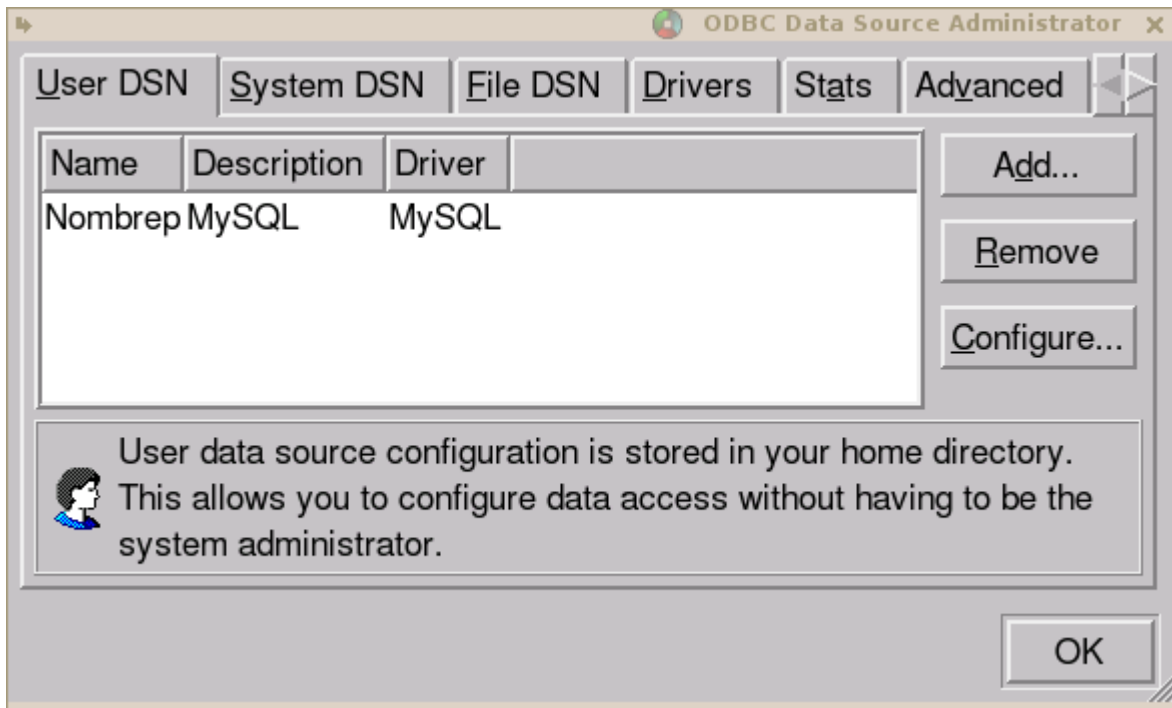


Figura 5.2.9. Administrador de controladores UnixODBC

Al igual que en Windows tan solo tenemos que añadir un DSN de usuario y rellenar los datos que se solicitan.

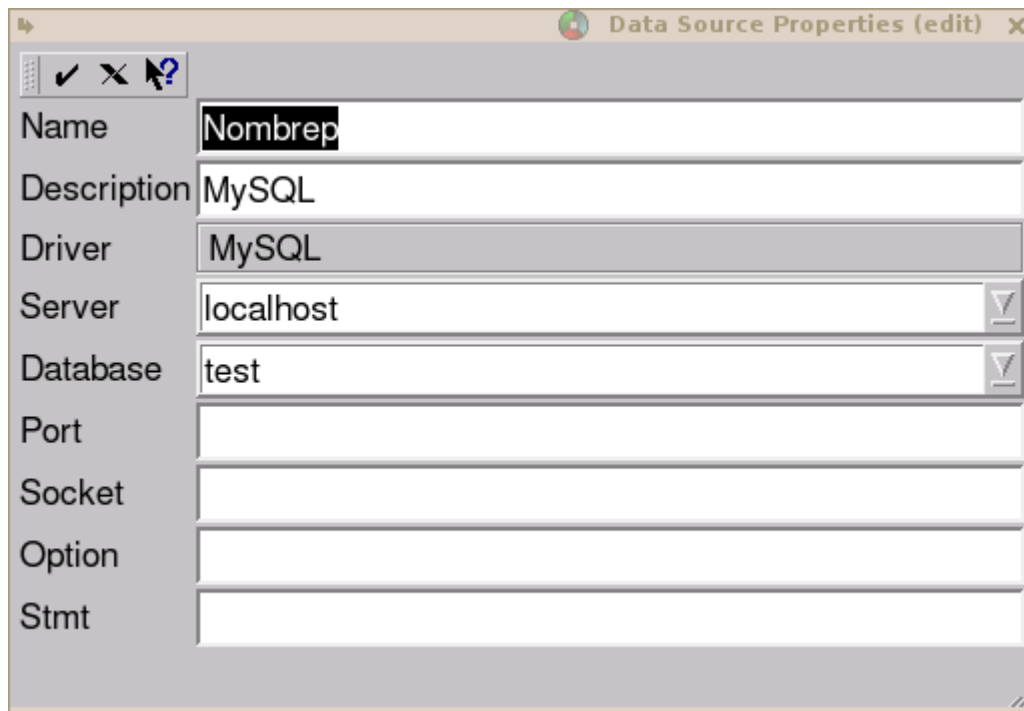


Figura 5.2.10 Nuevo DSN de usuario con UnixODBC.





Instalación y actualización de PhpMyAdmin



Si se quiere actualizar la versión de PhpMyAdmin, recordad realizar una copia del fichero 'config.inc.php o config.inc.php3 ' que se encuentra en el directorio donde esta instalado phpMyAdmin.

Podemos buscar la última actualización en:

<http://www.phpmyadmin.net/>

En función del sistema operativo con el que trabajemos, deberemos descargar un fichero ZIP o TGZ. Una vez descargado, debemos descomprimir el fichero en la carpeta 'phpMyAdmin' (aunque después podremos renombrarla).



Si no lo tenemos ya instalado, debemos descomprimir el fichero creando la carpeta 'phpmyadmin', en el directorio del servidor web que tengamos. En el caso de Internet Information Server (Microsoft) el directorio por defecto es: 'C:\inetpub\wwwroot', y en el caso de 'Apache', generalmente '...\Apache\http\ o '/var/www/html'.

Recordad que debemos dar permisos de ejecución al directorio donde hemos descomprimido phpMyAdmin, puesto que el motor de esta herramienta esta compuesto por ficheros del tipo *.php.



Fichero de configuración. (config.inc.php)



Este fichero contiene la configuración de phpMyAdmin, variables que son utilizadas en las diferentes funciones de la herramienta (colores, usuarios, contraseñas, nombres de los servidores...).

A continuación detallaremos la función de algunas de estas definiciones de variables, y en concreto las relacionadas con la configuración del servidor.

```
$cfg['PmaAbsoluteUri'] = 'http://localhost/pma/';
```

Este parámetro determina, la localización de la aplicación PhpMyAdmin, en principio el sistema lo autodetecta, pero si fallara podríamos colocar la localización. Además, cabe recordar que 'pma' es el nombre de la carpeta relativa al PhpMyAdmin (se renombró la carpeta).

```
$cfg['Servers'][$i]['host'] = 'localhost';
```

En este parametro se define quien será el [identificativo de host], por defecto será 'localhost' ya que PhpMyAdmin actua siempre desde el propio servidor de bases de datos.

```
$cfg['Servers'][$i]['port'] = '';
```

Indica el puerto de acceso a MySQL, si no ponemos nada el puerto por defecto es el 3306, definido por defecto por MySQL.

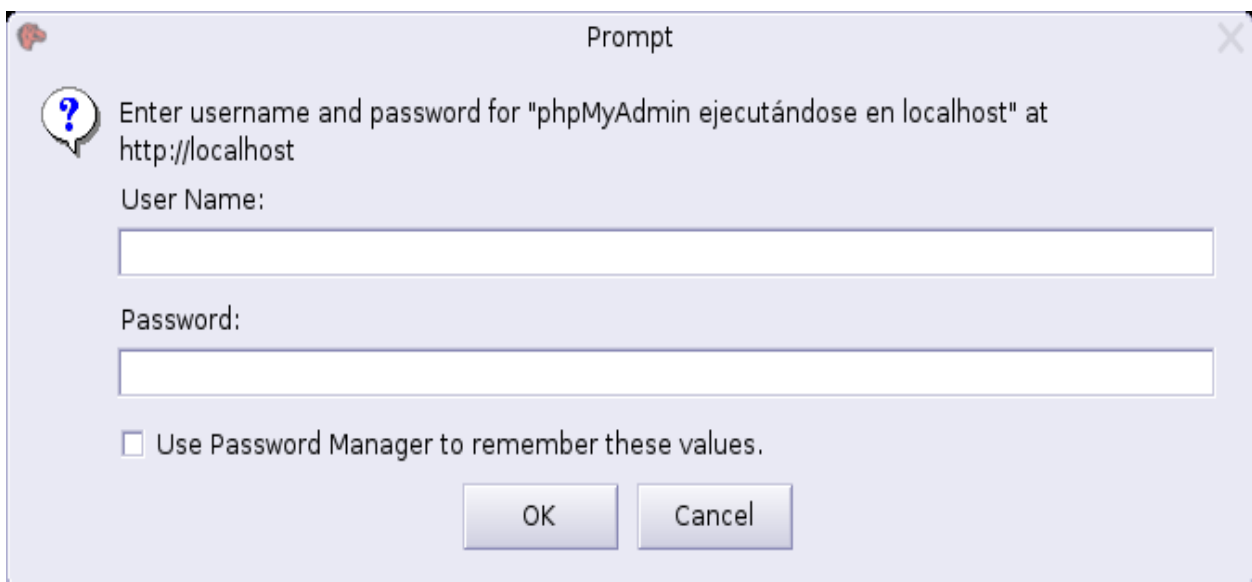
```
$cfg['Servers'][$i]['connect_type'] = 'tcp';
```

Definire el tipo de protocolo utilizado en la conexión, por defecto se utiliza 'TCP'.

```
$cfg['Servers'][$i]['auth_type'] = 'config';
```

Con este parametro definimos el método de autenticación que utilizarán los usuarios para acceder a phpMyAdmin. Si utilizamos el método 'config', tendremos que establecer los parámetros de usuario en este fichero, en los apartados 'user' y 'password' . Este es el método más utilizado, pero no el más recomendable por razones de seguridad.

Si utilizamos el método ' http', se solicitará para la autenticación del usuario, el usuario y la contraseña desde una pantalla del propio sistema, se consultarán los permisos que tiene el usuario a partir de los privilegios definidos en el propio servidor de MySQL. En la siguiente figura se puede observar la interface de identificación de usuario. Si se entra como 'root' en phpMyAdmin, hay un apartado de 'Privilegios', donde se definen los usuarios y los privilegios que tienen sobre las bases de datos.



The image shows a 'Prompt' dialog box with a question mark icon. The text inside reads: 'Enter username and password for "phpMyAdmin ejecutándose en localhost" at http://localhost'. Below this, there are two input fields: 'User Name:' and 'Password:'. At the bottom, there is a checkbox labeled 'Use Password Manager to remember these values.' and two buttons: 'OK' and 'Cancel'.

Figura 5.3.1. Autenticación PhpMyAdmin.



El sistema de seguridad y privilegios son implementados por el servidor MySQL, no por phpMyAdmin.

```
$cfg['Servers'][$i]['user'] = 'root';
```

Indica el nombre del usuario con el que se accederá desde phpMyAdmin sobre el servidor MySQL. Es muy importante recordar que hay un usuario por defecto llamado 'root' y sin contraseña. Este parámetro, por lo tanto solo tiene sentido si el método de autenticación es 'config'.

Nunca se debe permitir que se acceda a una base de datos con el usuario 'root', debemos crear otros usuarios con sus permisos correspondientes. (Opción 'Privilegios' de PHPMyAdmin).

```
$cfg['Servers'][$i]['password'] = '';
```

Esta claro que este parámetro establece la contraseña, por defecto, la contraseña de 'root', esta vacía.

Al igual que el parámetro anterior, sólo tiene sentido si el método de autenticación es 'config'.

```
$cfg['Servers'][$i]['only_db'] = '';
```

En esta variable se establece la base de datos a la que se accede, o puede tratarse de un array de bases de datos que se podrán visualizar desde PhpMyAdmin. Si se deja vacía esta variable se verán todas las bases de datos.

Existen muchas más variables, que establecen filtros, caminos, etc. En caso de querer profundizar en la configuración de PhpMyAdmin, sería recomendable consultar la información asociada a la herramienta.





Herramientas para administrar una BD.

En este módulo se pretende conocer herramientas para la administración de bases de datos, como ejemplo se creará una base de datos sobre la que trabajaremos en módulos posteriores, veremos tres:

- [PhpMyadmin](#)
- [MySQL Control Center](#)
- [MySQL Administrator](#)

PhpMyAdmin

Se trata de una herramienta de gestión exclusiva de MySQL creada inicialmente por Tobias Ratschiller con PHP y continuada por muchos colaboradores. Consiste en un conjunto de scripts PHP para administrar las bases de datos de MySQL bajo un entorno web. Esta particularidad de funcionar bajo entorno web permite administrar bases de datos en remoto, simplemente necesitamos una conexión a Internet y un navegador. En muchísimas empresas en las que se da servicio web, se ofrece PhpMyadmin como herramienta de administración de nuestras bases de datos.

Con cualquier navegador podemos acceder directamente a <http://localhost/pma/> (donde pma es el directorio virtual donde hemos instalado PhpMyadmin) para ver la pantalla principal de administración.

En el caso de querer actualizar la herramienta a una versión más moderna o que se quiera instalar de nuevo, en el punto anterior se explican los pasos a seguir.

En la pantalla inicial de PhpMyAdmin, se puede observar que aparece en la parte inferior (texto en rojo) información de que en el fichero de configuración, existen parámetros por defecto (usuario: *root* y sin contraseña). Esta información intenta avisarnos de que podemos tener en peligro la seguridad del sistema, ya que cualquiera podría entrar.

Es muy recomendable que los usuarios se identifiquen con su usuario y contraseña, para ello debemos modificar el parámetro correspondiente (ver en el punto anterior). Debemos modificar la contraseña del usuario 'root' y resulta muy recomendable crear otro usuario con los mismos privilegios que root.

Pantalla principal de PhpMyAdmin

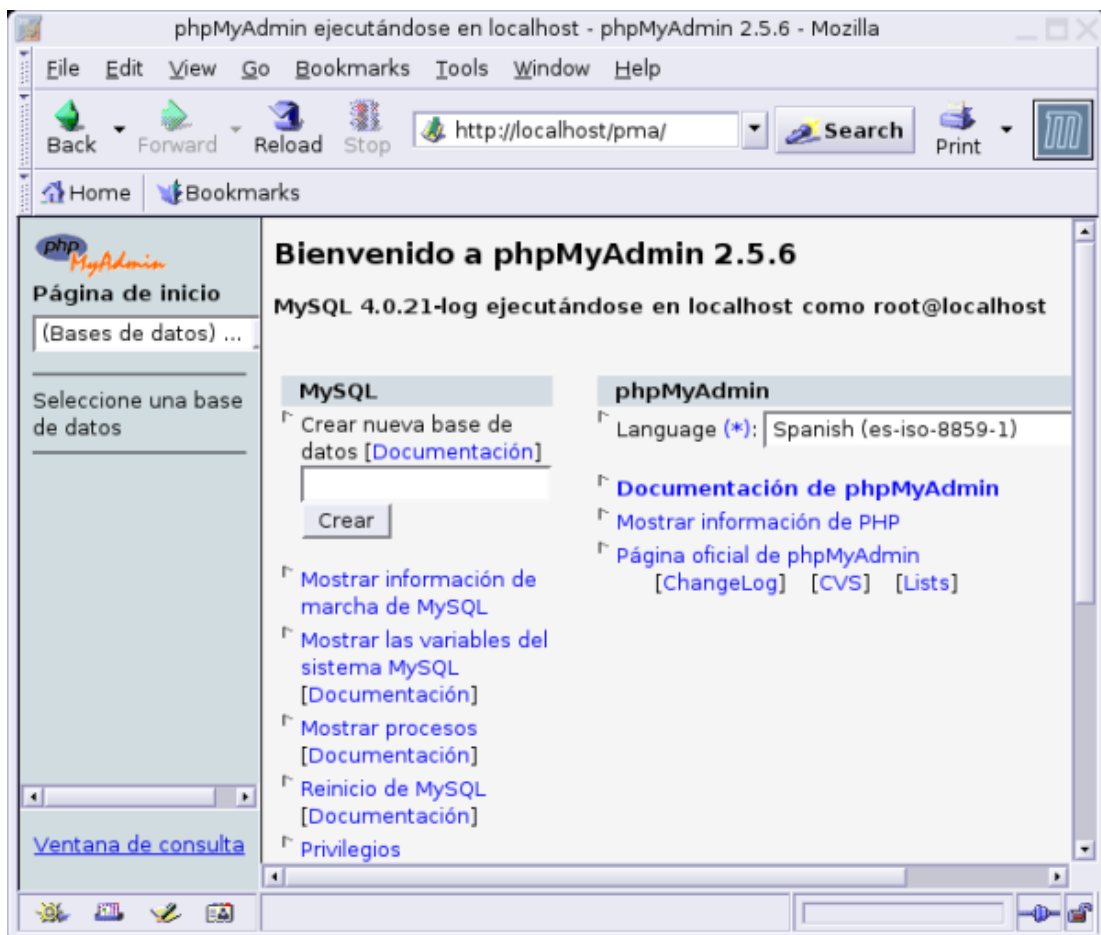


Figura 5.4.1. Pantalla principal de phpMyAdmin.

En la parte central y a la derecha, vemos que podemos seleccionar el idioma, ver documentación y llamar a la función `phpinfo()` para obtener información del sistema y su configuración.

En el centro aparecen las operaciones relacionadas con las bases de datos MySQL. Estadísticas de acceso, entorno de variables, información de procesos, privilegios y detalles de las bases de datos.

En la parte superior izquierda aparece un desplegable donde podemos seleccionar la base de datos que queremos utilizar.

En la parte inferior izquierda podemos abrir una nueva ventana de consulta, donde utilizando instrucciones SQL podemos atacar la base de datos.

Existen problemas a la hora de exportar una base de datos de una plataforma a otra (linux - windows), aunque hemos detectado que alguna versión, o alguna otra herramienta, mejor dicho, ya resuelve el problema, es preferible escribir siempre los nombres en minúsculas, de esta manera, no tendremos problemas para que nuestra base de datos sea portable entre plataformas.



Crear una base de datos (academia).

Hacer clic sobre el campo crear base de datos y teclear el nombre que tendrá la base de datos (en nuestro caso **academia**). Hay que tener en cuenta que es sensible a la letras mayúsculas y minúsculas, por lo que a la hora de utilizar la base de datos desde PHP tendremos que escribir su nombre como se creó.

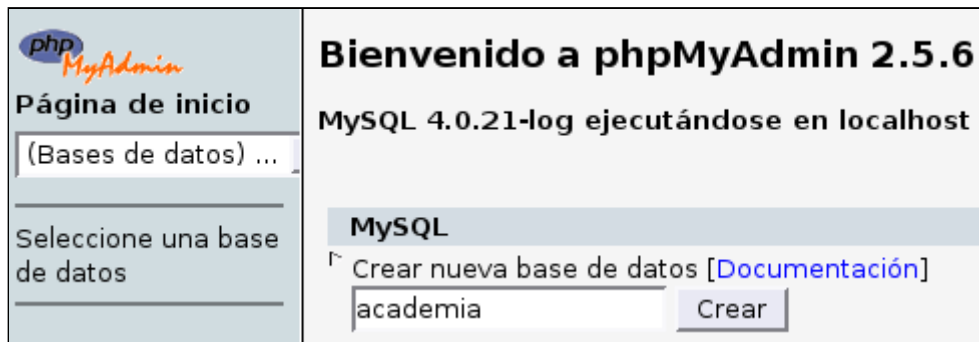


Figura 5.4.2. Crear una nueva base de datos.

Después de pulsar el botón Crear, se pide el nombre de la primera tabla y el número de campos que tendrá.

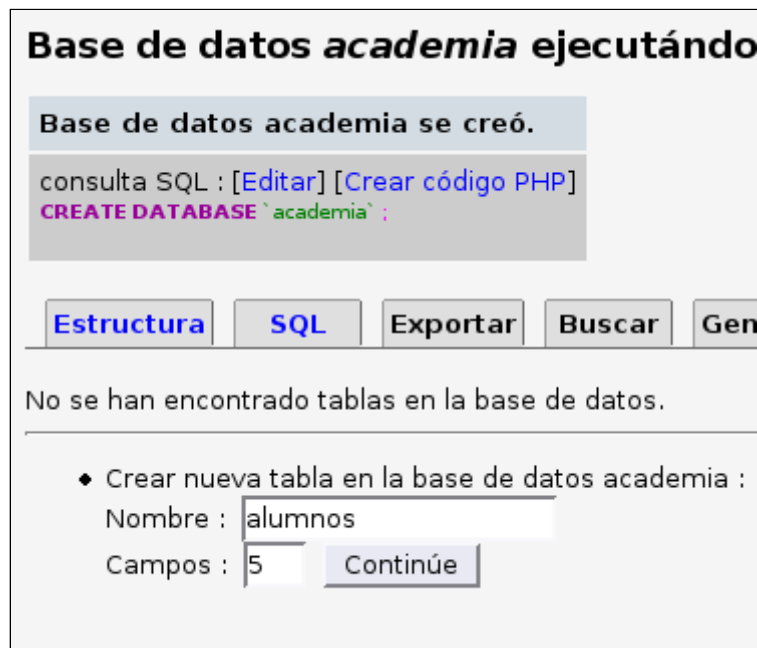


Figura 5.4.3. Crear una tabla de datos.

Podemos ver los tipos de variables que soporta Mysql desde el siguiente enlace [tipos de variable](#).

Modificar los tipos de campos, como se muestra en la siguiente tabla.

Tabla: alumnos				
Campo	Tipo	Null	Defecto	Otros
dni	varchar(8)	No	0	clave primaria
nombre	varchar(30)	No	null	
apellidos	varchar(30)	No	null	
telefono	varchar(30)	Sí	null	
poblacion	tinyint(1)	No	null	

La tabla de la base de datos quedaría:

The screenshot shows the phpMyAdmin interface for the 'academia' database. On the left, there is a sidebar with 'Página de inicio' and a search box containing 'academia (3)'. Below this, the 'academia' database is listed with its tables: 'alumnos', 'cursos', and 'matrículas'. At the bottom of the sidebar is a 'Ventana de consulta' button. The main area displays a table list with columns: 'Tabla', 'Acción', 'Registros', and 'Tipo'. The tables listed are 'alumnos', 'cursos', and 'matrículas', each with 0 records and MyISAM engine. A summary row shows '3 tabla(s)' and 'Número de filas' as '0'. Below the table list, there are links for 'Revisar todos/as / Desmarcar todos' and a 'Con marca:' input field. At the bottom, there are links for 'Vista de impresión', 'Diccionario Datos', and a form to 'Crear nueva tabla en la base de datos academia' with fields for 'Nombre' and 'Campos', and a 'Continúe' button.

Figura 5.4.4. Tabla de alumnos.

Crear las otras dos tablas siguiendo las siguientes instrucciones.

Tabla: cursos

Campo	Tipo	Null	Defecto	Otros
codigo	int(11)	No	autoincrement	clave primaria
titulo	varchar(30)	No	null	
n_plazas	tinyint(4)	No	0	
precio	int(11)	No	0	
lugar_realizacion	varchar(30)	No	null	

Tabla: matriculas

Campo	Tipo	Null	Defecto	Otros
dni	varchar(8)	No		clave primaria
codigo	int(11)	No		clave primaria



Otras operaciones con PhpMyAdmin.

Una vez seleccionada la base de datos ejemplo (**academia**) podemos ver las operaciones que podemos ejecutar sobre ella.

A la izquierda se muestra la base de datos en uso y las tablas que la componen. A la derecha se muestran cada una de las tablas con sus características (número de registros, tipo de tabla, tamaño) y operaciones asociadas a cada una de ellas.

Las operaciones que podemos realizar:



Examinar: permite ver y actualizar los registros de la tabla.



Seleccionar: permite hacer una selección de registros a visualizar.



Insertar: permite insertar datos en la tabla seleccionada.



Propiedades: muestra información de los campos de la tabla.



Eliminar: permite eliminar la tabla, la estructura y los datos.



Vaciar: borra los datos y deja la estructura.



Exportar tablas de una BD.

Una de las pestañas superiores '**Exportar**' permite hacer copias de seguridad de la BD o intercambiar datos con otros sistemas, el resultado se almacena en un fichero que se puede enviar a otra máquina.

Existen varios formatos para realizar la exportación (SQL, Latex, CSV, XML) pero el más común es SQL que crea un fichero con extensión SQL que incorpora junto a las sentencias SQL necesarias, el contenido y la estructura de las tablas seleccionadas.

Lo primero que debemos seleccionar son las tablas a exportar y el formato deseado. Si hemos seleccionado el formato SQL, a la derecha dispondremos de dos recuadros: 'Estructura' y 'Datos'. Desde el recuadro de estructura podemos modificar las características de los campos de las tablas y en el recuadro de datos decidimos si se crean las sentencias necesarias para exportar los datos.

En el apartado estructura existe la opción '*añadir drop table*', que añade al fichero .sql las instrucciones necesarias para borrar las tablas que se exportan antes de ejecutar las sentencias de inserción.

Para poder guardar el fichero generado debemos seleccionar la opción 'enviar', donde podremos escribir el nombre del fichero resultante o permitir que la herramienta asigne el nombre de la base de datos. Cuando pulsemos el botón continuar, aparecerá un cuadro de diálogo que nos permitirá seleccionar la ubicación de dicho fichero.



Importar tablas de una BD.

Seleccionar la pestaña **SQL**, y pulsar el botón **Browse**, buscar y seleccionar el fichero creado con la opción exportar.

Base de datos *academia* ejecutándose en *localhost*

Ejecute la/s consulta/s SQL en la base de datos academia
[\[Documentación\]](#) :

Mostrar esta consulta otra vez

O Localización del archivo de texto :

(Tamaño máximo: 30,720KB)

Compresión: Autodetecte Ninguna "Comprimido con gzip" "Comprimido con bzip"

Figura 5.4.7. Exportar tablas.

Después de pulsar el botón continuar se ejecutará el fichero de ordenes SQL e informará que la consulta se ejecutó con éxito.

Este sistema es ideal para hacer copias de seguridad de las bases de datos, es aconsejable hacer las exportaciones con ficheros ***.sql**.

Para practicar estos conceptos podemos exportar la base de datos academia, crear una nueva base de datos con un nombre nuevo, y recuperar la estructura y los datos de la anterior.



MySQL Control Center

La misma empresa de MySQL lanzó una aplicación cliente de administración para el servidor de bases de datos MySQL, que podemos descargar de la web de [MySQL](#). Incorpora opciones adicionales a PhpMyAdmin. Hay versiones para Windows y Linux.

La filosofía de la herramienta es totalmente diferente: mientras que MySQLCC es una aplicación que se ejecuta en la maquina cliente, PhpMyAdmin reside en el servidor, por lo que solo tiene sentido para los administradores de los servidores. No son excluyentes y muy recomendable tener ambos accesibles.

El problema es que MySQL Control Center en estos momentos ha quedado obsoleto y no sigue desarrollándose. Se ha sustituido por el conjunto de programas MySQL Administrator y MySQL Query Browser.



MySQL Administrator

MySQL Administrator es el nuevo software de administración de servidores de Bases de Datos de MySQL que ha creado MySQL AB. Se trata de un software multiplataforma, que por el momento se encuentra disponible para Linux y Microsoft Windows y que cuenta con un entorno gráfico de usuario muy intuitivo.

Este nuevo producto suple las carencias que tiene MySQL Control Center en el área de

Administración de servidores.

La descarga del programa se hace desde la página web de MySQL, en "http://dev.mysql.com/downloads/administrator". Tan solo tenemos que descomprimir el fichero en un directorio y ejecutar el comando "**mysql-administrator**".

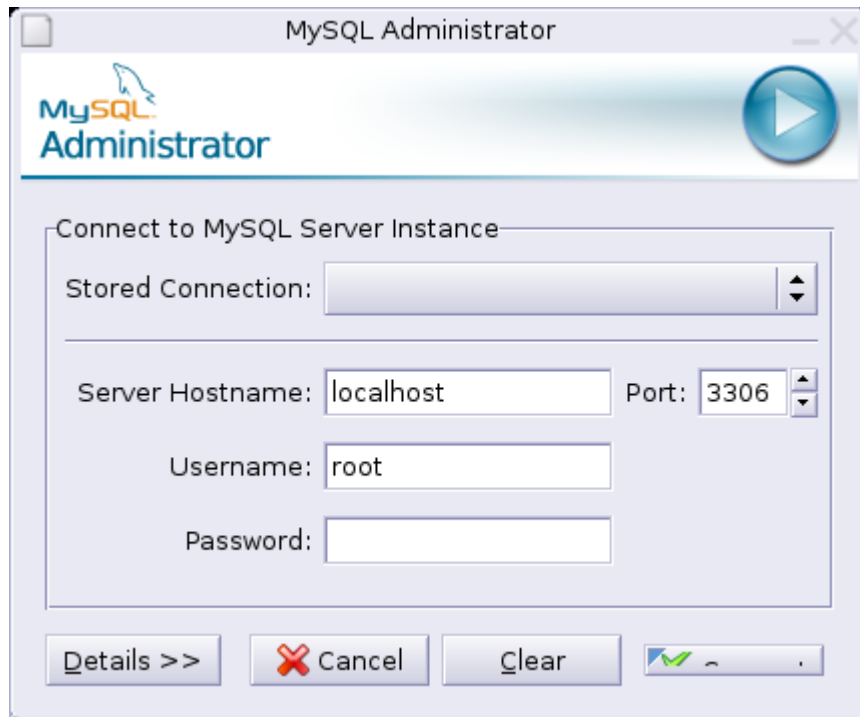


Figura 5.4.8. Mysql Administrator

MySQL Administrador es una herramienta que permite realizar tareas administrativas sobre servidores de MySQL incluyendo:

- la configuración de las opciones de inicio de los servidores.
- inicio y detención de servidores.
- monitorización de conexiones al servidor.
- administración de usuarios.
- monitorización del estado del servidor, incluyendo estadísticas de uso.
- visualización de los logs de servidor.
- gestión de copias de seguridad y recuperaciones.
- visualización de catálogos de datos.

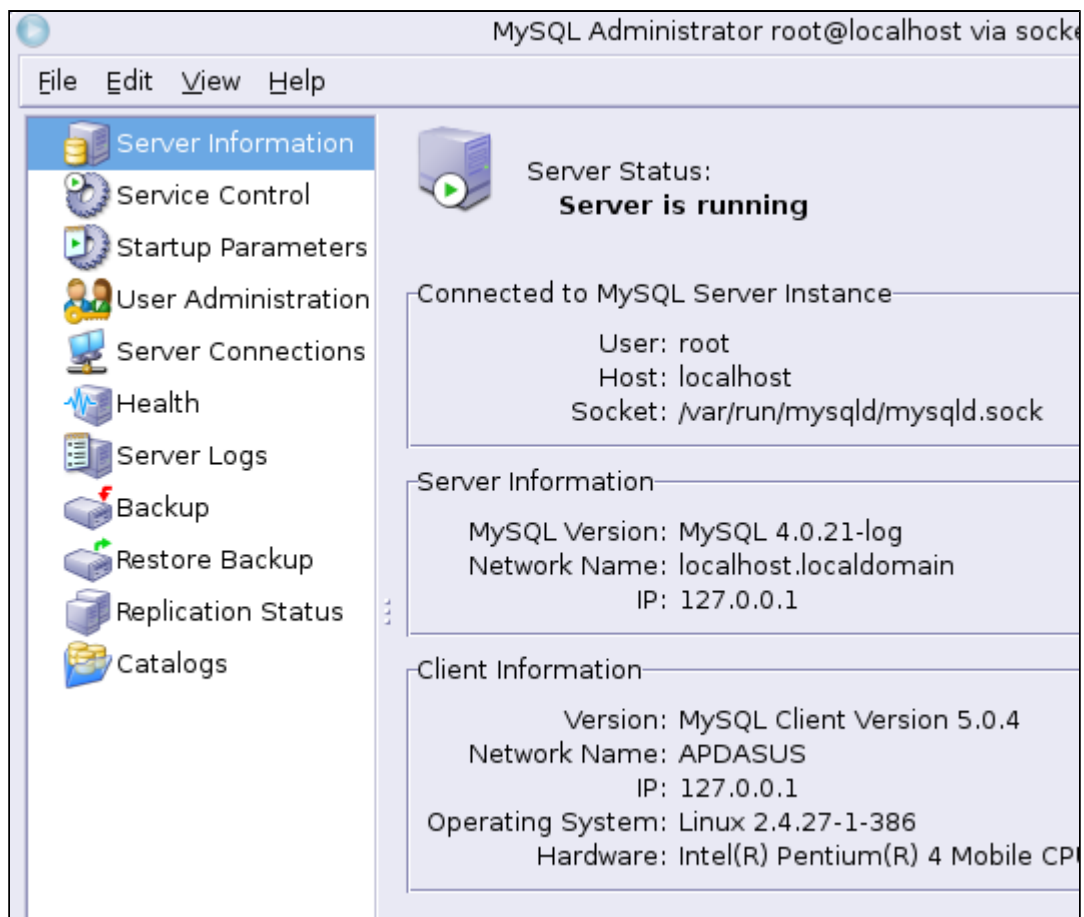


Figura 5.4.9. Mysql Administrator

También en la página de MySQL existen una serie de FAQ's que ayudan en la instalación, en caso de dudas, aunque la misma es bastante simple. Recordad que esta aplicación no es necesaria para el seguimiento de curso





Lenguaje SQL.

En este módulo se pretende dar a conocer las operaciones básicas que se pueden hacer con SQL y que tienen una aplicación directa con la creación de aplicaciones web.

SQL: lenguaje estructurado de consulta

A finales de los años setenta, IBM creó para su producto DB2 un lenguaje llamado SQL (Structured Query Language). Empresas de la competencia crearon paralelamente otros SQL a medida de sus necesidades. Pero fue cerca de los años ochenta cuando el comité ANSI definió un estándar para SQL.

SQL es un lenguaje estándar de comunicación con bases de datos. Un lenguaje normalizado que nos permite trabajar con cualquier tipo de lenguaje en combinación de cualquier tipo de bases de datos.

Que sea estándar no significa que las instrucciones de manipulación de distintas bases de datos sean idénticas, es más, existen determinadas operaciones que solo funcionan con determinadas bases de datos.



Componentes de SQL

El lenguaje SQL está distribuido en cuatro grandes bloques, que se combinan para crear, actualizar y manipular las bases de datos, estos bloques son:

- Comandos.
- Cláusulas.
- Operadores.
- Funciones.

Comandos

Hay dos tipos de comandos en SQL:

- Las **DDL**, que permiten crear y definir nuevas bases de datos, campos e índices.
- Las **DML**, que permiten generar consultas para ordenar, filtrar y extraer información de la base de datos.

Comandos DDL	Descripción
CREATE	Crear nuevas tablas, campos e índices.
ALTER	Modificación de tablas añadiendo campos o modificando la definición de los campos.
DROP	Instrucción para eliminar tablas, campos e índices.

Figura 5.5.1. SQL. Comandos DDL

Comandos DML	Descripción
SELECT	Consulta de registros de la base de datos que cumplen un criterio determinado.
INSERT	Insertar registros a la base de datos.

UPDATE	Instrucción que modifica los valores de los campos y registros especificados en los criterios.
DELETE	Eliminar registros de una tabla de la base de datos.

Figura 5.5.2. SQL. Comandos DML



Cláusulas

Las cláusulas son condiciones de modificación, utilizadas para definir los datos que se desean seleccionar o manipular.

Cláusula	Descripción
FROM	Utilizada para especificar la tabla de la que se seleccionarán los registros.
WHERE	Cláusula para detallar las condiciones que deben reunir los registros resultantes.
GROUP BY	Utilizado para separar registros seleccionados en grupos específicos.
HAVING	Utilizada para expresar la condición que ha de cumplir cada grupo.
ORDER BY	Utilizada para ordenar los registros seleccionados de acuerdo a una ordenación específica.

Figura 5.5.3. SQL. Cláusulas

Operadores

Operador lógico	Descripción
AND	Evalúa dos condiciones y devuelve el valor cierto, si ambas condiciones son ciertas.
OR	Evalúa dos condiciones y devuelve el valor cierto, si alguna de las dos condiciones es cierta.
NOT	Negación lógico. Devuelve el valor contrario a la expresión.

Figura 5.5.4. SQL . Operadores lógicos

Operadores de comparación	Descripción
<	[...] menor que [...]
>	[...] mayor que [...]
<>	[...] diferente a [...]
<=	[...] menor o igual que [...]
>=	[...] mayor o igual que [...]
=	[...] igual que [...]

BETWEEN	Especifica un intervalo de valores
LIKE	Compara un modelo
IN	Operadores para especificar registros de una tabla

Figura 5.5.5. SQL. Operadores de comparación



Consultas



Antes de comenzar a realizar algunas prácticas como ejemplo de las operaciones que hemos visto, debemos insertar algunos datos en nuestras tablas de la base de datos **academia**. Podemos hacerlo de varias formas:

- Directamente desde el servidor de bases de datos: abrimos una consola de 'root' y ejecutamos mysql, se abrirá una nueva línea de comandos donde podremos escribir sentencias (use database nombre,...).
- Con PhpMyadmin: lo hemos visto en el punto anterior, seleccionamos la base de datos, la tabla y pulsamos en la opción de insertar.
- Importando los datos: también lo hemos visto en el punto anterior, como podemos descargar el fichero [academia.sql](#).



Seleccionar la tabla **alumnos** y ver que resultado muestre la siguiente instrucción SQL:

```
SELECT * FROM alumnos
```

Sentencia SQL 1

El signo '*' quiere decir que seleccionaremos todos los campos que tenga la tabla, en este caso, alumnos. El comando **SELECT** puede llevar adjunto un parámetro [ALL | DISTINCT | DISTINCT ROW]. El valor por defecto es 'ALL', que permite el retorno de registros duplicados, mientras que con las otras dos opciones los registros duplicados se omiten.

Figura 5.5.6. SQL. Creación de consulta1.

Si pulsamos el botón 'Continúe', se visualizarán todos los registros de la tabla **alumnos**.



Si pulsamos el botón Browse podremos guardar las instrucciones SQL de la ventana de texto en un fichero con extensión 'sql'.



Ahora para limitar la selección, filtraremos solo los alumnos que residen en la población de Sagunto:

```
SELECT * FROM alumnos WHERE poblacion="Sagunto"
```

Sentencia SQL 2



Si solo queremos extraer el *nombre* y el primer *apellido* de los alumnos que residen en Sagunto ordenados por *apellidos*, el código sería:

```
SELECT nombre, apellidos FROM alumnos WHERE poblacion="Sagunto" ORDER BY apellidos
```

Sentencia SQL 3

			nombre	apellidos
<input type="checkbox"/>			Andrea	Diaz-Alejo Leon
<input type="checkbox"/>			Stéphane	Diaz-Alejo Leon
Con marca:				

Figura 5.5.7. Resultado sentencia SQL 3



Pero por regla general necesitamos datos que residen en varias tablas. Queremos obtener el *nombre*, los *apellidos*, el *curso* en que estan matriculados y su *lugar de realización*. Las tablas 'alumnos' y 'cursos' estan relacionadas mediante la tabla 'matrícula'.

```
SELECT alumnos.nombre, alumnos.apellidos, cursos.título, cursos.lugar_realización
FROM alumnos, cursos, matrículas
WHERE alumnos.dni = matrículas.dni and cursos.codigo=matrículas.codigo
ORDER BY cursos.lugar_realización
```

Sentencia SQL 4



Se podría establecer los vinculos entre tablas utilizando la palabra reservada 'JOIN', pero deberemos asegurarnos que la versión de PHP de que disponemos las interpreta correctamente, algunas versiones dan problemas y si no somos nuestros propios administradores es muy posible que no tengamos la posibilidad de actualizar la versión correspondiente.

```
SELECT alumnos.nombre, alumnos.apellidos, cursos.título, cursos.lugar_realización
FROM alumnos, cursos
JOIN matrículas
WHERE alumnos.dni=matrículas.dni and cursos.codigo=matrículas.codigo
ORDER BY cursos.lugar_realización
```

Sentencia SQL 5

El resultado de esta consulta es:

nombre	apellidos	título	lugar_realización
María	Pérez Dasis	Linux	Alicante
Stéphane	Diaz-Alejo Leon	Oracle	Alicante
Andrea	Diaz-Alejo Leon	PHP	Castellón
Antonio	García Gómez	PHP	Castellón
Andrea	Diaz-Alejo Leon	Dreamweaver	Valencia

Mostrar : 30 filas empezando de 0
 en modo horizontal y repite
 encabezados cada 100 celdas

Figura 5.5.8. Resultado sentencia SQL 5



Crear una tabla



Crearemos una nueva tabla para almacenar datos del seguimiento del alumno en cada curso (horas de asistencia, realizado los ejercicios y aptitud). Para ello utilizaremos la sentencia **CREATE TABLE** seguida del nombre de la tabla. Entre paréntesis encontraremos las características de los campos (dni, código, horas_asist, ejercicios, apto); seguidamente se define el índice principal (dni+código).

```
CREATE TABLE `notas` (
  `dni` VARCHAR(8) NOT NULL,
  `codigo` INT(11) NOT NULL,
  `horas_asis` SMALLINT(3) NOT NULL,
  `ejercicios` VARCHAR(2) NOT NULL,
  `optitud` VARCHAR(7) NOT NULL,
  INDEX (`dni`, `codigo`)
)
```

Sentencia SQL 6



El parámetro 'NOT NULL', hace que este campo no pueda ser nulo cuando se realiza una inserción en la tabla.



Inserción de datos en una tabla

La sentencia para insertar registros en una tabla es la siguiente:

```
INSERT [INTO] nombre_tabla [(campo1,campo2...)] VALUES (valor1,valor2...), .....
```



Insertar un nuevo registro a la tabla 'alumnos', con los datos siguientes.

```
INSERT INTO `alumnos` (`dni`, `nombre`, `apellidos`, `teléfono`, `población`) VALUES ('66666666', 'Juan', 'Pérez García', '674832939', 'Castellón')
```

Sentencia SQL 7

		dni	nombre	apellidos	teléfono	población
<input type="checkbox"/>		11111111	Andrea	Díaz-Alejo Leon	890345321	Sagunto
<input type="checkbox"/>		22222222	Stéphane	Díaz-Alejo Leon	893437635	Sagunto
<input type="checkbox"/>		33333333	Antonio	García Gómez	896345861	Valencia
<input type="checkbox"/>		44444444	María	Pérez Dasis	678345092	Alicante
<input type="checkbox"/>		66666666	Juan	Pérez García	674832939	Castellón
Con marca:						

Figura 5.5.9. Resultado sentencia SQL 7



Modificación de estructura de la tabla

Para modificar la estructura de una tabla se utiliza la sentencia **ALTER TABLE**. La sintaxis es parecida a la de CREATE TABLE, pero con más opciones.

La sintaxis es: **ALTER TABLE** nombre_tabla seguida de alguna de las siguientes opciones:

Opciones disponibles	Descripción
ADD [COLUM] campo tipo_datos [NOT NULL [NULL] [DEFAULT valor_defecto] [AUTO_INCREMENT]	Añade nueva columna a la tabla.
ADD PRIMARY KEY (nom_index_primario)	Añade nueva clave primaria a la tabla.
ADD INDEX [nom_index,]	Añade nueva clave a la tabla
ALTER [COLUM] campo {SET DEFAULT DROP DEFAULT }	Introduce nuevo valor estándar.
CHANGE [COLUM] campo_viejo definición_campo	Cambia un elemento del campo y modifica el nombre de la columna.
MODIFY [COLUM] definición_camp	Cambia la definición del campo, sin modificar el nombre del campo.
DROP [COLUM]	Elimina una columna.
DROP [PRIMARY KEY]	Elimina una clave primaria.
DROP INDEX [nom_index,....]	Elimina una clave.
RENAME AS nom_nova taula	Renombra una tabla.

Figura 5.5.10. Opciones de sentencia ALTER TABLE



Añadir las columnas 'nivel_educativo' y 'dirección' a la tabla **alumnos**:

```
ALTER TABLE alumnos ADD COLUMN `dirección` varchar(40) NULL , ADD COLUMN
`nivel_educativo` varchar(10) NULL
```

Sentencia SQL 8



Modificar datos de una tabla

La sintaxis es:

```
UPDATE tabla SET campo1=valor1 , [campo2=valor2] ... [WHERE condición]
```

```
UPDATE alumnos SET teléfono= '666123456' WHERE nombre='Andrea' and apellidos='Diaz-
Alejo Leon'
```

Sentencia SQL 9

←T→			dni	nombre	apellidos	teléfono	población
<input type="checkbox"/>			11111111	Andrea	Diaz-Alejo Leon	666123456	Sagunto
<input type="checkbox"/>			22222222	Stéphane	Diaz-Alejo Leon	893437635	Sagunto
<input type="checkbox"/>			33333333	Antonio	García Gómez	896345861	Valencia
<input type="checkbox"/>			44444444	María	Pérez Dasis	678345092	Alicante
<input type="checkbox"/>			66666666	Juan	Pérez García	674832939	Castellón

Con marca:

Figura 4.4.13. Resultado de la sentencia SQL 9



Borrar datos de la tabla

La sintaxis es:

DELETE FROM nombre_tabla [WHERE condición]



Tener en cuenta que si borramos datos de una tabla no se podrán recuperar. Si no utilizamos la cláusula WHERE, se borrarán todos los registros de la tabla.



Borrar tablas

La sintaxis es:

DROP TABLE [IF EXISTS] nombre_de_la_tabla



Si utilizamos esta sentencia en un programa, debemos usar la cláusula IF EXIST, ya que si la tabla no existe obtendríamos un error de ejecución.



Hay que tener en cuenta que el poder ejecutar estas sentencias en un servidor dependerá de los permisos que tenga el usuario con el que nos conectamos al servidor de bases de datos. Es muy posible que no dispongamos de todos los permisos.





Conexión de PHP y MySQL.

En este último punto conseguiremos realizar una conexión sencilla con la base de datos **academia** desde el código PHP. Se introducirán las funciones PHP nativas para acceder a las bases de datos a un nivel elemental.

El protocolo para acceder a una base de datos no cambia de como lo hacemos en otros lenguajes de programación:

- Abrir la base de datos.
- Lanzar la sentencia SQL al servidor de base de datos.
- Recoger el resultado de la consulta.
- Cerrar la conexión de la base de datos.



Sentencias MySQL.

En la siguiente tabla se pueden ver algunas de las sentencias más utilizadas para acceder a una base de datos MySQL. Hay dos columnas, una para sentencias nativas de MySQL y otra equivalente para el acceso a bases de datos utilizando ODBC.

PHP / MySQL	PHP / ODBC
mysql_affected_rows (\$conex) Devuelve la cantidad de registros afectados en la última sentencia de actualización o eliminación.	
mysql_close (\$conex) Cierra la conexión con el servidor MySQL.	odbc_close(...)
mysql_connect (\$host, \$usuario, \$contraseña) Abre una conexión con el servidor MySQL.	odbc_connect(...)
mysql_query (\$db, \$sql, \$conex) Envía una sentencia SQL (\$sql) al servidor de base de datos (\$db) con la conexión (\$conex).	odbc_do(\$conex, \$sql) odbc_exec(\$conex, \$sql)
mysql_error (\$conex) Devuelve un mensaje de texto con el error producido.	
mysql_fetch_array (\$resul, \$tipo) Devuelve el resultado de la consulta realizada dentro de un <i>array</i> . En \$tipo se pueden definir tres constantes: MYSQL_ASSOC, MYSQL_NUM, MYSQL_BOTH.	
mysql_fetch_row (\$resul, \$num_fila) Avanza a la siguiente fila activa del resultado de la sentencia SQL generada. El parámetro \$num_fila es opcional, y si se pasa, se accederá a la fila especificada en este parámetro.	odbc_fetch_row (...)
mysql_field_name (\$resul, \$index) Devuelve el nombre del campo al que apunta la variable <i>index</i> .	odbc_fiel_name(...)

mysql_num_fields (\$resul)

Informa de la cantidad de campos que se han devuelto en el resultado.

odbc_num_fields(...)**mysql_num_rows (\$resul)**

Devuelve el número de filas que ha devuelto la sentencia SQL.

odbc_num_rows(...)**mysql_pconnect (\$host, \$usuario, \$contraseña)**

Abre una conexión persistente contra el servidor MySQL.

odbc_pconnect(\$dsn,\$usu,\$con)**mysql_query (\$sql, \$conex)**

Envía una instrucción SQL contra la conexión.

mysql_select_db (\$base_de_datos, \$conex)

Selecciona una de las bases de datos para establecer una conexión.

mysql_result (\$resul,\$fila,\$columna)

Devuelve el contenido parcial de una sentencia SQL correspondiente al objeto de la fila y columna seleccionadas por el usuario. A diferencia de *mysql_fetch_array*, sólo devuelve un valor.

odbc_result_all (&resul,\$fila,\$columna)**mysql_free_result (\$resul)**

Libera el espacio donde se ha almacenado el resultado de la sentencia SQL.

odbc_free_result(\$resul)**Otras funciones PHP sobre MySQL.****mysql_change_user (\$usuario, \$contraseña)**

Modifica el usuario actual por otro nuevo usuario.

mysql_create_db (\$base_de_datos)

Crea una nueva base de datos sobre el sistema.

mysql_drop_db (\$base_de_datos)

Elimina una base de datos y las tablas que la componen.

mysql_inser_id (\$resultado)

Devuelve el valor generado al insertar un registro que contiene un valor del tipo 'Auto increment' en alguno de sus campos.

mysql_get_host_info()

Devuelve información del estado de la conexión y del servidor.

mysql_get_proto_info()

Devuelve información relacionada con el protocolo utilizado en la conexión.



Visualizaremos los nombres de todos los alumnos de la academia. Primero estableceremos la

conexión con el servidor de BD, seleccionaremos la BD de trabajo y ejecutaremos la SQL. El resultado se almacenará en una variable de memoria de tipo array.



Se puede copiar el código siguiente o descargar el siguiente fichero [practica51.php](#).



```
<!doctype html public "-//W3C//DTD HTML 4.0 //EN">
<html>

<head>
<?php
/*-----
* Módulo: 5 Práctica: 5.1. Fichero: practica51.php
* Autor: Fecha:
* Descripción: Conexión MySQL con funciones nativas
* Pre condi.: Servidor MySQL, BD academia
* Post cond.:
-----*/
?>
<title>Práctica 5.1</title>
</head>
<body>
<?php
/* Abrir la conexión con el servidor(servidor, usuario, contraseña */
$conn=mysql_connect("localhost","root","");

/* Identificamos la base de datos a utilizar */
$db=mysql_select_db("academia");

/* Escribimos la sentencia SQL */
$sql="select nombre from alumnos" ;

/* Se ejecuta y si se obtienen resultados se almacenan en $resul */
if ( ! $resul=mysql_query($sql) ) {
    echo "No se ha podido realizar la consulta";
    echo mysql_error();
    exit;
}

echo "Conexión al Servidor MySQL. BD:academia tabla: alumnos <br>" ;
echo "La consulta ejecutada es: <b> $sql <b> <br>";
echo "El resultado es: <br> ";

/* Recorremos el array */
while ($arr_resul= mysql_fetch_array($resul)){
    print ($arr_resul[0]."<br>");
}

/* Liberamos espacio ocupado con el resultado de la consulta */
mysql_free_result($resul);

// Cerramos la conexión con el servidor de BD
mysql_close();
?>
</body>
</html>
```

Listado 5.5.1. Código del fichero **practica51.php**



En la figura siguiente podemos ver el resultado de la ejecución del códigoPHP:

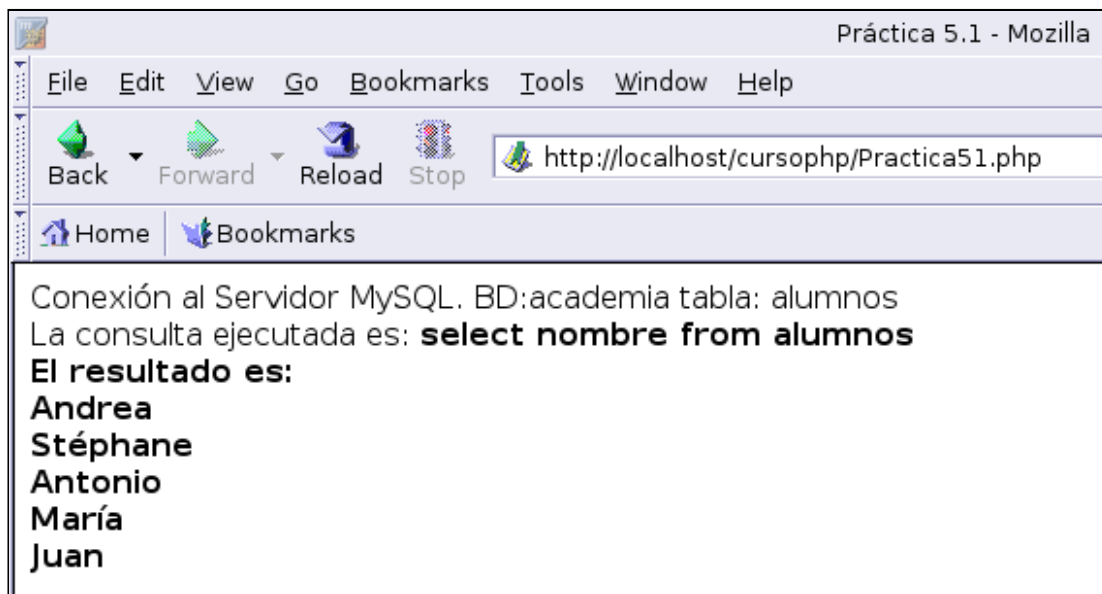



Figura 5.5.1. Resultado de la ejecución del fichero **practica51.php**

Como se puede ver, acceder a la información de una base de datos y mostrarla en pantalla es muy sencillo. En el siguiente tema se profundizará más sobre este tema.

Las sentencias nativas de PHP para acceder a MySQL las podemos encontrar ampliadas en [el apartado dedicado de la página oficial de PHP](#).

Si hacemos la misma operación que el ejemplo anterior pero utilizando el driver ODBC a una base de datos access. Debemos tener instalado MS-Access y creado un DSN a la base de datos, por ejemplo con el nombre **dsn_academia**.



```
<!doctype html public "-//W3C//DTD HTML 4.0 //EN">
<html>
<head>
<?php
/*-----
* Módulo: 4 Práctica: 5.2. Fichero: practica52.php
* Autor: Fecha:
* Descripción: Conexión a una base de datos Access por ODBC
* Pre condi.: ODBC DSN dsn_academia
* Post cond.:
-----*/
?>
<title>Práctica 5.2</title>
</head>
<body>
<?php

/* Abrir la conexión con el servidor(servidor, usuario, contraseña */$conn=odbc_connect
("dsn_academia","root","");

/* Identificamos la base de datos a utilizar */
$sql="select nombre from alumnos" ;

/* Si se obtienen resultados se almacenan en $resul */
if ( ! $resul=odbc_exec($conn,$sql)) {
    echo "No se ha podido realizar la consulta";
```

```
    echo odbc_error($conn);
    exit;
}

echo "Conexión al Servidor MySQL. BD:academia tabla: alumnos <br>" ;
echo "La consulta ejecutada es: <b> $sql <b> <br>";
echo "El resultado es: <br> ";

/* Recorremos el array */
while ($rc = odbc_fetch_into($resul,$arr_result)){
    print($arr_result[0]."<br>");
}

/* Liberamos espacio ocupado con el resultado de la consulta */
odbc_free_result($resul);

// Cerramos la conexión con el servidor de BD
odbc_close($conn);
?>
</body>
</html>
```

Listado 5.5.2. Código del fichero **practica52.php**

Observar que los cambios entre **practica51.php** y **practica52.php** son mínimos, sólo afectan a las sentencias que acceden a los orígenes de la base de datos.



En el caso de querer trabajar con el ODBC de Oracle, mSQL u otros motores de bases de datos, se puede consultar el apartado correspondiente del manual de PHP para obtener información de las sentencias concretas que se deben utilizar en cada caso.



Tipos de variables en MySQL

Tipo	Descripción	Rango
tinyint	entero muy pequeño	-128 ... 127
smallint	entero pequeño	-32768 ... 32767 sin signo 0 ... 65535
mediumint	entero mediano	-8388608 a 8388607 sin signo 0 ... 16777215
int	entero	-2147683648 a 2147683648 sin signo 0 ... 4294967295
bigint	entero largo	-2 E63 a 2 E63 sin signo 0 ... 2 E64
float	coma flotante simple precisión	± 1.175494351 E-38 (mínim) ± 3.402823466 E+38 (màxim)
double	coma flotante doble precisión	± 2.2250738585072014 E-308 ± 1.7976931348623157 E+308
decimal	coma flotante, representado como cadena	
char(M)	cadena caracteres longitud fija	Acepta entre 1 y 255 caracteres, la longitud máxima la determina M, que es obligatorio.
varchar(M)	cadena caracteres longitud variable	Acepta entre 1 y 255 caracteres, la longitud máxima la determina M, que es obligatorio.
tinyblob	blob (objeto binario grande) muy pequeño	255 bytes
blob	blob estándar	65 kilobytes
mediumblob	blob mediano	16 megabytes
longblob	blob grande	4 gigabytes
tinytext	cadena texto muy pequeña	255 bytes
text	cadena texto pequeña	65 kilobytes
mediumtext	cadena texto mediana	16 megabytes
longtext	cadena texto grande	4 gigabytes
enum	enumeración de elementos	64 K elementos
set	conjunto de elementos	64 elementos
date	fecha con formato: AAAA-MM-DD	1000-01-01 a 9999-12-31
time	hora con formato: hh:mm:ss	-838:59:59 a 838:59:59

datetime	fecha y hora	
timestamp	intervalo de tiempos de...	19700101000000 hasta cualquier fecha del año 2037
year	año con formato AAAA	de 1091 a 2155

