

ÉRICA PALOMINO

TEMA 10

# PHP Y MYSQL

**[erica.palomino@escuelaartegranada.com](mailto:erica.palomino@escuelaartegranada.com)**

ESCUELAARTEGRANADA

# INTRODUCCIÓN

Para nuestra aplicación web utilizaremos un SGBD **relacional**:

Los datos se guardan en tablas

- Cada tabla tiene filas
- Cada fila tiene columnas.

El acceso a SGBD relacionales se hace a través de **SQL**.

En este caso, vamos a utilizar el SGBD “**MariaDB**” que no es más que una versión extendida de MySQL

# USO DE MYSQL

# MYSQL

La elección de MySQL radica en que:

- Es gratuito.
- Está disponible para Windows, UNIX, OS/2, etc.
- Es capaz de trabajar con millones de registros.
- Es muy rápido
- No necesita grandes recursos de máquina.

Nosotros vamos a utilizar una herramienta gráfica (escrita en PHP ) para administrar MySQL vía Web: **phpMyAdmin**

Para acceder a esta herramienta: **localhost/phpmyadmin**

# MYSQL

Desde esta pantalla podemos realizar las acciones más básicas de administración:

## Crear una base de datos

### Bases de datos

 **Crear base de datos** 

## Comprobar las versiones de software utilizadas:

#### Servidor de base de datos

- Servidor: 127.0.0.1 via TCP/IP
- Tipo de servidor: MariaDB
- Versión del servidor: 10.1.9-MariaDB - mariadb.org binary distribution
- Versión del protocolo: 10
- Usuario: root@localhost
- Conjunto de caracteres del servidor: UTF-8 Unicode (utf8)

#### Servidor web

- Apache/2.4.18 (Win32) OpenSSL/1.0.2e PHP/7.0.0
- Versión del cliente de base de datos: libmysql - mysqlnd 5.0.12-dev - 20150407 - \$Id: 7e72f9690b1498a1bead7a637c33a831c0d2f65: \$
- extensión PHP: mysqli 
- Versión de PHP: 7.0.0

#### phpMyAdmin

- Acerca de esta versión: 4.5.1, versión estable más reciente: 4.9.5
- [Documentación](#)
- [Wiki](#)
- [Página oficial de phpMyAdmin](#)
- [Contribuir](#)
- [Obtener soporte](#)
- [Lista de cambios](#)

# MYSQL

Con esto ya estamos conectados a MySQL

También podemos conectarnos a través de clases de PHP y sus métodos

- MariaDBi::prepare
- MariaDBi::query
- MariaDBi\_stmt::bind\_param



# **CREAR UNA BASE DE DATOS**

# MYSQL

Vamos a empezar creando la siguiente base de datos llamada "CENTRO":

- Alumnos (dni, nombre, edad)
- Asignaturas (codigo, nombre, creditos, trimestre)
- Matriculas (dni, codigo, año, nota)

A screenshot of a web-based MySQL database creation interface. At the top, there is a header "Crear base de datos" with a small green plus icon and a blue help icon. Below this, there is a text input field containing the word "centro". To the right of the input field is a dropdown menu currently showing "Cotejamiento" with a downward arrow. To the right of the dropdown is a grey button with the text "Crear".

Crear base de datos ?


centro Cotejamiento ▼ Crear



# CREAR UNA BASE DE DATOS

En principio, aunque la base de datos está creada, aún no hay ninguna tabla en ella.

No se han encontrado tablas en la base de datos.

 Crear tabla

Nombre:  Número de columnas:

Continuar



# **CREACIÓN DE TABLAS**

# MÉTODOS MÁGICOS

Crear tablas en MySQL es tan sencillo como:

1. Elegir el nombre de la tabla
2. Elegir el número de campos (columnas) de la tabla.



Crear tabla

Nombre:  Número de columnas:

Continuar

# MÉTODOS MÁGICOS

Seleccionar para cada campo:

- Nombre
- Tipo
- Longitud
- Etc...

Nombre	Tipo	Longitud/Valores	Predeterminado	Cotejamiento	Atributos	Nulo	Índice	A.I.
<input type="text"/>	INT	<input type="text"/>	Ninguno	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	---	<input type="checkbox"/>
Seleccionar desde las columnas centrales								
<input type="text"/>	INT	<input type="text"/>	Ninguno	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	---	<input type="checkbox"/>
Seleccionar desde las columnas centrales								
<input type="text"/>	INT	<input type="text"/>	Ninguno	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	---	<input type="checkbox"/>
Seleccionar desde las columnas centrales								



# **TIPOS DE DATOS DE MYSQL**

# TIPOS DE DATOS

## Char (longitud)

- Puede tener un tamaño máximo de 255 caracteres.

## Varchar(longitud)

- Igual que varchar2

## Int

- Números enteros entre -2.147.483.648 hasta 2.147.483.648
- Si ponemos unsigned (edad) serán desde 0 hasta 4.294.967.295

# TIPOS DE DATOS

## Tinynt

- Número pequeño
- Desde -127 hasta 128
- Con unsigned desde 0 a 255

## Smallint

- Número pequeño
- Desde -32.768 hasta 32.768
- Con unsigned desde 0 a 65.535

# TIPOS DE DATOS

## Float (e,d), double (e,d), decimal (e,d)

- Igual que number

## Date

- Almacena fechas
- Los distintos formatos aceptados son:
  - YYYY-MM-DD
  - YY-MM-DD
  - YYMMDD



# TIPOS DE DATOS

## Time

- Almacena valores de tipo hora
- Formatos:
  - HH:MM:SS
  - HHMMSS
  - HHMM

## Year

- Almacena valores de tipo año
  - Formato: YYYY

# TIPOS DE DATOS

## Timestamp

- Almacena los valores de marcas de tiempo

## Enum y set

- Especifica los valores que podrá contener una columna
- Igual que IN ('a'...)

## Serial

- Permite crear valores que se van a autorrellenar de forma incremental.



# **CARACTERÍSTICAS DE LAS COLUMNAS**

# CARACTERÍSTICAS DE LAS COLUMNAS

## Primary key

- MySQL indexará la tabla por esta columna de forma automática

## Autoincrement

- Sólo válido para campos de tipo entero.
- Si no se inserta valor para el campo, se le asigna el siguiente valor libre.
- No puede haber más de una columna de este tipo por tabla.

# CARACTERÍSTICAS DE LAS COLUMNAS

## Default

- Valor por defecto que se asignará a la columna en caso de que no se introduzca nada.


## Null ~ not null

- Indicamos si el campo podrá estar vacío.

# **CREACIÓN DE TABLAS**


# CREACION DE TABLAS


Una vez pulsado el botón “GRABAR” se nos muestra un resumen de la tabla creadas.


	#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predetermina
<input type="checkbox"/>	1	dni 	varchar(9)			No	Ninguna
<input type="checkbox"/>	2	nombre	varchar(50)			No	Ninguna
<input type="checkbox"/>	3	edad	int(2)			No	Ninguna


# CREACION DE TABLAS


Además nos permite modificar la tabla creada, añadiendo más campos:


 Vista de impresión

 Planteamiento de la estructura de tabla




 Hacer seguimiento a la tabla

 Mover columnas

 Agregar

columna(s)

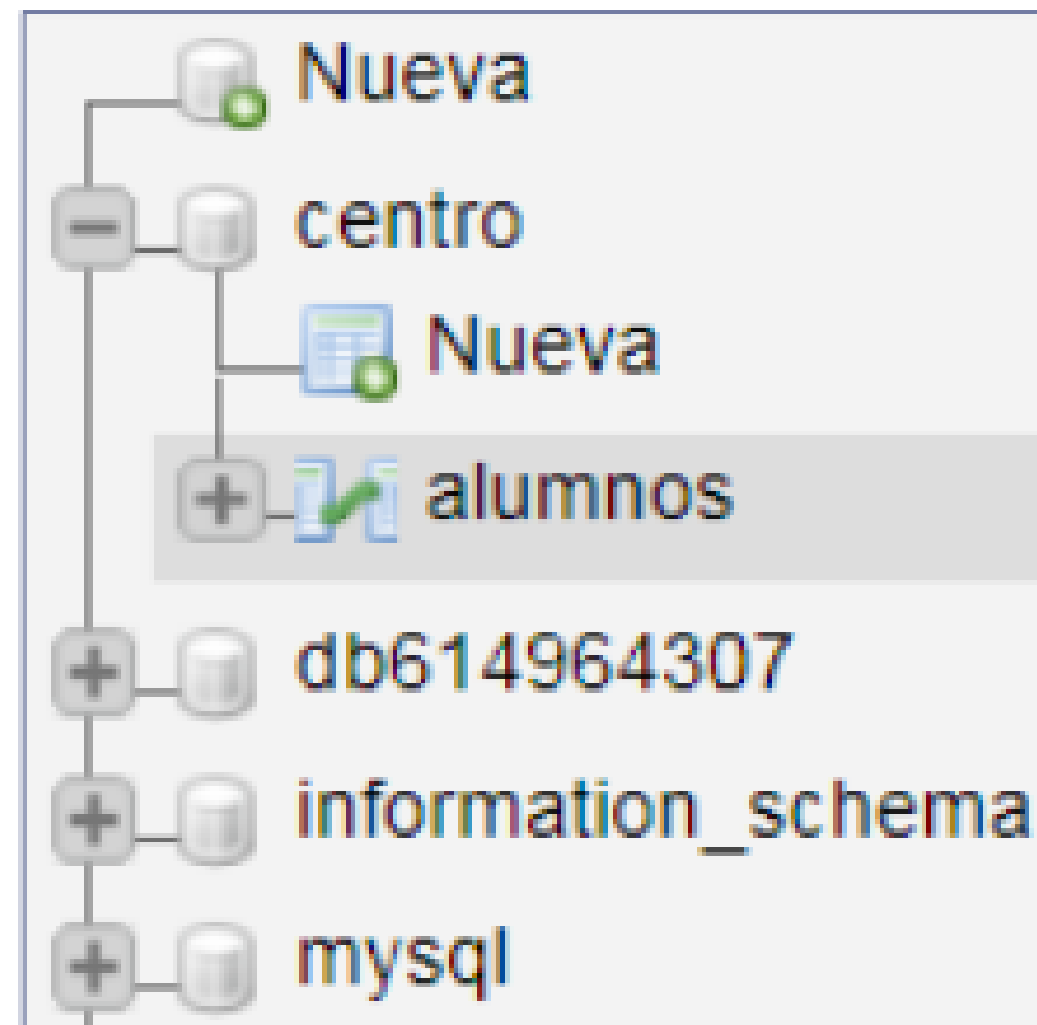


Continuar



# CREACION DE TABLAS

En la columna de la izquierda podemos ir viendo siempre la estructura de nuestra BD.



# CREACION DE TABLAS

Crearemos las siguientes 2 tablas de la siguiente forma:

Tabla asignaturas:

Nombre	Tipo	Longitud/Valores	Predeterminado	Cotejamiento	Atributos	Nulo	Índice
<input type="text" value="codigo"/> <small>Seleccionar desde las columnas centrales</small>	SERIAL	<input type="text"/>	Ninguno	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	PRIMARY
<input type="text" value="nombre"/> <small>Seleccionar desde las columnas centrales</small>	VARCHAR	50	Ninguno	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	---
<input type="text" value="creditos"/> <small>Seleccionar desde las columnas centrales</small>	INT	3	Ninguno	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	---
<input type="text" value="trimestre"/> <small>Seleccionar desde las columnas centrales</small>	SET	'1','2','3' <small>Editar valores ENUM/SET</small>	Ninguno	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	---

# CREACION DE TABLAS

Tabla matriculas:

Nombre	Tipo	Longitud/Valores	Predeterminado	Cotejamiento	Atributos	Nulo	Índice
<input type="text" value="dni"/> <small>Seleccionar desde las columnas centrales</small>	<input type="text" value="VARCHAR"/>	<input type="text" value="9"/>	<input type="text" value="Ninguno"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value="PRIMARY"/>
<input type="text" value="codigo"/> <small>Seleccionar desde las columnas centrales</small>	<input type="text" value="BIGINT"/>	<input type="text" value="20"/>	<input type="text" value="Ninguno"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value="PRIMARY"/>
<input type="text" value="año"/> <small>Seleccionar desde las columnas centrales</small>	<input type="text" value="INT"/>	<input type="text" value="4"/>	<input type="text" value="Ninguno"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value="PRIMARY"/>
<input type="text" value="nota"/> <small>Seleccionar desde las columnas centrales</small>	<input type="text" value="DECIMAL"/>	<input type="text" value="4,2"/>	<input type="text" value="Ninguno"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value=""/>



# **CREACIÓN DE RELACIONES**

## CREACION DE RELACIONES

Una vez creadas las tablas, hay que crear las restricciones de integridad referencial.

- Foreign key.

Como siempre, las relaciones se crearán en la tabla que hace referencia a la otra.

Abrimos la tabla para ver su estructura y pulsamos en "Vista de relaciones"

## CREACION DE RELACIONES

Vamos a crear las relaciones entre matriculas y alumnos, asignaturas.

Seleccionamos la tabla "matriculas" puesto que es ahí dónde están las claves externas.



# **INSERCIÓN DE DATOS**

# INSERCIÓN DE DATOS

PhpMyAdmin nos permite realizar varias acciones sobre las tablas, entre ellas:

- Ver su estructura o contenido
- Insertar filas
- Vaciar su contenido
- Borrar la tabla



# INSERCIÓN DE DATOS

Para insertar datos, sólo tenemos que seleccionar dicha opción.



# INSERCIÓN DE DATOS

Una vez introducidos los datos podemos elegir qué hacer después de insertar la fila. Nosotros, seguiremos insertando:

# INSERCIÓN DE DATOS

Vamos a insertar los siguientes datos:

Tabla Alumnos

- 55555555Z, Ramón Torres, 19
- 22222222B, María López, 21
- 33333333C, Paloma Ruiz, 24
- 44444444R, Isabel Perea, 25

# INSERCIÓN DE DATOS

## Tabla Asignaturas

- 1, Bases de datos, 15, 1
- 2, Programación, 18, 2
- 3, Lenguajes de marcas, 23, 1

## Tabla Matriculas

- 55555555Z, 1, 2020, 8
- 55555555Z, 2, 2020, 4
- 22222222B, 1, 2019, 4
- 22222222B, 1, 2020, 6
- 33333333C, 2, 2019, 7



# **CONEXIÓN DESDE PHP**

# CONEXIÓN DESDE PHP

PHP cuenta con una serie de clases que nos permiten conectarnos a las bases de datos que se encuentran en nuestro servidor:

- `mysqli`
- `mysqli_stmt`
- `mysqli_result`

# CLASE MYSQLI

# CONEXIÓN DESDE PHP

PHP cuenta con una serie de clases que nos permiten conectarnos a las bases de datos que se encuentran en nuestro servidor:

- `mysqli`
- `mysqli_stmt`
- `mysqli_result`



# MYSQLI

La clase **mysqli** representa una conexión entre PHP y una base de datos MariaDB o MySQL

Nos ofrece una serie de propiedades y métodos que nos van a facilitar la vida mucho.

Podemos consultarlos todos en:

<https://www.php.net/manual/es/class.mysqli.php>



Veremos aquí como siempre los más importantes

# MYSQLI

**mysqli::\_construct(servidor, usuario, contraseña, base de datos)**

- Abre una nueva conexión con el servidor de MariaDB
- Recibe como parámetros:
- Devuelve un objeto de la clase **mysqli**

**mysqli::close():** cierra la conexión abierta



# **EJECUCIÓN DE CONSULTAS**

# EJECUCIÓN DE CONSULTAS

La consulta que queremos enviar a la base de datos será una **cadena de texto**.

Puede estar almacenada en una variable o no.

Podemos realizar **cualquier operación** que conocemos de bases de datos.

Para ejecutar una consulta utilizamos el método:

# EJECUCIÓN DE CONSULTAS

**mysqli::query(consulta):** Permite realizar una consulta a la base de datos

- Insert
- Update
- Delete
- Drop
- Select
- Etc... a base de datos

# EJECUCIÓN DE CONSULTAS

Este método devuelve distintos valores dependiendo de la sentencia ejecutada.

## Para Select:

- Devuelve un objeto de tipo `mysqli_result`
- `FALSE` si hay algún error.

## Para Insert, Update, Delete:

- `TRUE` si todo ha ido bien
- `FALSE` si ha habido algún error

# EJECUCIÓN DE CONSULTAS

## **mysqli::affected\_rows**

- Número de filas afectadas por la última consulta realizada de tipo insert, update o delete
- Número de filas devueltas por la última consulta realizada de tipo select

En caso de que una consulta de error, podemos consultar varias **propiedades** del objeto conexión para saber qué ha pasado:

- **mysqli::errno**: contiene el código del error que ha sucedido
- **mysqli::error**: contiene una cadena de texto con la descripción del error que ha sucedido.



# **MYSQLI\_RESULT**



# MYSQLI\_RESULT

Representa el conjunto de datos devueltos por una consulta SELECT

Esta clase tiene varias ***propiedades***. Las más importantes:

- **mysqli::num\_rows**: contiene el número de filas que ha devuelto el select
- **mysqli::field\_count**: contiene el número de columnas que ha devuelto el select

# MYSQLI\_RESULT

Además, tiene muchos **métodos** que nos facilitan la vida para trabajar con los datos.

Veamos algunos:

# MYSQLI\_RESULT

## `mysqli_result::fetch_array ([tipo])`

Devuelve un array en el que hay una de las filas del conjunto de datos.

El array tiene **dos posiciones** por cada campo devuelto por el select. Una **numérica** y una **posicional**.

Devuelve **Null** cuando no hay más datos para mostrar

# MYSQLI\_RESULT

Como parámetro opcional recibe el tipo de array que queremos obtener.

Posibles tipos:

- MYSQLI\_ASSOC: para obtener el array asociativo
- MYSQLI\_NUM: para obtener el array posicional
- MYSQLI\_BOTH: para obtener el array doble

Por defecto será **BOTH**

# MYSQLI\_RESULT

## `mysqli_result::fetch_all ([tipo])`

Devuelve en este caso una **matriz** que tendrá:

- Tantas filas como filas devuelva la consulta select
- Tantas columnas como campos le pidamos a la consulta select.



# MYSQLI\_RESULT

Por defecto la matriz será **posicional**.

Se puede elegir el tipo de matriz utilizando:

- MYSQLI\_ASSOC: para obtener el array asociativo
- MYSQLI\_NUM: para obtener el array posicional
- MYSQLI\_BOTH: para obtener el array doble

Solo cambiarán las columnas, las filas siguen siendo posicionales



# **SENTENCIAS PREPARADAS**

# SENTENCIAS PREPARADAS

Una de las ventajas que aporta el uso de la clase **mysqli** frente al estilo procedimental son las **sentencias preparadas**.

Una sentencia preparada es aquella en la que el código de la sentencia incluirá ciertos **parámetros** que se dejan para rellenar más adelante.

De esta forma, la sentencia se podrá ejecutar de la misma forma o muy similar de manera muy rápida y eficiente.



## SENTENCIAS PREPARADAS - VENTAJAS

**Reducen** el tiempo de análisis ya que la preparación de la consulta se hace una sola vez.

Son muy útiles **contra inyecciones** ya que los valores de los parámetros se transmiten después usando un protocolo diferente y no necesitan ser escapados.

La ventaja que tienen las sentencias preparadas es que podemos incluir en ellas **parámetros** que se rellenarán al momento de ejecutarlas.

Los parámetros se indican con ?

# SENTENCIAS PREPARADAS

La ejecución de una sentencia preparada tiene dos fases:

- **Preparación y comprobación:** se crea la plantilla de la sentencia y el servidor comprueba que no hay errores de sintaxis.
- **Vinculación y ejecución:** se rellenan los parámetros y se ejecuta la sentencia.

# SENTENCIAS PREPARADAS

**mysqli::prepare(sentencia):**

- Prepara una sentencia para su posterior ejecución.
- La llamada a «prepare» envía la plantilla de la consulta a la base de datos.
- La base de datos analiza, compila y optimiza la consulta y la guarda **sin ejecutar**.

# SENTENCIAS PREPARADAS

Los pasos a seguir siempre serán:

1. Preparar la consulta con los parámetros necesarios.
2. Asignar variables para almacenar los datos devueltos por la consulta.
3. Asignar valores a los parámetros de la consulta
4. Ejecutar la consulta
5. Sacar los datos devueltos por la consulta

## SENTENCIAS PREPARADAS

El método `mysqli::prepare()` devuelve un objeto de la clase **`mysqli_stmt`** que representa a la sentencia preparada.



**MYSQLI\_STMT**

## MYSQLI\_STMT - PROPIEDADES

**mysqli\_stmt::affected\_rows**: número de filas afectadas por una sentencia **no select**.

**mysqli\_stmt::num\_rows**: número de filas devueltas por la sentencia **select**.

**mysqli\_stmt::field\_count**: número de columnas devueltas por la sentencia **select**.

**mysqli\_stmt::errno**: código del error ocurrido

**mysqli\_stmt::error**: texto descriptivo del error

**mysqli\_stmt::param\_count**: número de parámetros de la sentencia.

# MYSQLI\_STMT - MÉTODOS

## **mysqli\_stmt::bind\_param(tipos, list)**

Permite asignar valores a los parámetros de la sentencia.

Recibe dos elementos:

- El tipo de dato de los valores que va a recibir:
  - i => si el valor asignado es un número sin decimales
  - d => si el valor asignado es un número con decimales
  - s => si el valor asignado es una cadena de caracteres
- Listado con los valores que recibe para asignar a los parámetros

Si la sentencia preparada tiene más de un parámetro:

- El tipo se indica en una única cadena de texto
- Los valores se separan por comas, en orden correcto



# MYSQLI\_STMT - MÉTODOS

## `mysqli_stmt::bind_result(lista)`

- Permite asignar variables para recoger los valores devueltos por un select.
- Se asignará una variable por cada columna del select

# MYSQLI\_STMT - MÉTODOS

## `mysqli_stmt::execute()`

- Ejecuta una sentencia preparada anteriormente
- Buscará los valores que haya en las variables asociadas a la sentencia.

# MYSQLI\_STMT - MÉTODOS

## `mysqli_stmt::fetch()`

- Extrae una fila de los valores devueltos por la consulta
- Volcará el resultado en las variables que se asociaron.
- Esas variables ya se pueden usar para lo que sea necesario
- Devuelve FALSE si no hay datos que sacar

# MYSQLI\_STMT - MÉTODOS

## `mysqli_stmt::close()`

- Cierra la sentencia preparada y desvincula las variables usadas como parámetros o como resultado

# MYSQLI\_STMT - RESÚMEN

Atributo	Tipo	Descripción
\$affected_rows	int	Almacena el número de filas afectadas por la ultima sentencia ejecutada (no Select)
\$errno	int	Devuelve el código de error producido por la última sentencia ejecutada
\$error_list	array	Devuelve una lista de errores producidos por la última sentencia ejecutada
\$error	string	Devuelve una cadena con el error producido por la última sentencia ejecutada
\$field_count	int	Número de campos de la sentencia dada
\$insert_id	int	Devuelve el id generado en la última operación INSERT
\$num_rows	int	Devuelve el número de filas resultado de ejecutar un select
\$param_count	int	Número de parámetros de la sentencia dada

# MYSQLI\_STMT - RESÚMEN

Método	Devuelve	Descripción
attr_get (\$at)	int	Devuelve el valor actual de un atributo de la sentencia
bind_param(list)	bool	Agrega variables a la sentencia
bind_result(list)	bool	Vincula variables a la sentencia para almacenar resultados
data_seek(int \$n)	void	Busca una fila concreta dentro del resultado
execute()	Bool	Ejecuta la sentencia preparada
fetch()	Bool	Obtiene una fila del resultado
free_result()	Void	Libera la memoria de los resultados almacenadas
get_result()	mysqli_result	Obtiene el resultado de ejecutar una sentencia
prepare(\$consulta)	mixed	Prepara la sentencia
close()	bool	Cierra una sentencia preparada
reset()	bool	Reinicia una sentencia preparada