

Desarrollo de sitios web con PHP y MySQL



Acceso a bases de datos MySQL
en PHP





Tema 4: Acceso a bases de datos MySQL en PHP

1. Bases de datos en la Web
2. Instalación y configuración de MySQL
3. MySQL
4. Herramientas de administración: phpMyAdmin
5. Lenguaje SQL
6. Funciones de PHP para el acceso a bases de datos MySQL
7. Ejercicios
8. Consulta avanzada de tablas

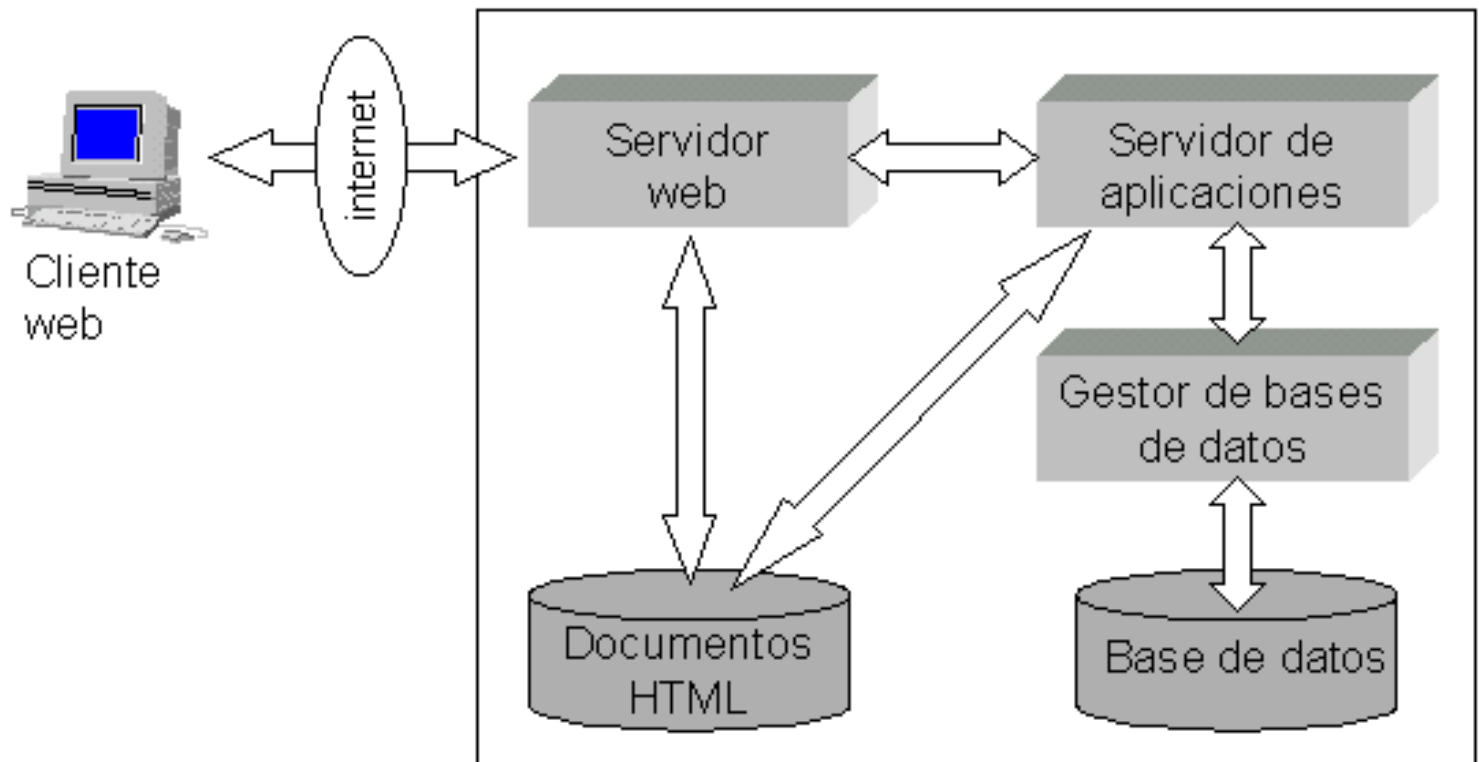


Bases de datos en la Web

- Las bases de datos permiten almacenar de una forma estructurada y eficiente toda la información de un sitio web
- Ventajas
 - Proporcionar información actualizada
 - Facilitar la realización de búsquedas
 - Disminuir los costes de mantenimiento
 - Implementar sistemas de control de acceso
 - Almacenar preferencias de los usuarios

Bases de datos en la Web

- Esquema básico de un sitio web soportado por bases de datos:





¿Qué base de datos utilizaremos?

- PHP desde sus inicios permite trabajar con la BBDD MySQL.
- MySQL paso a manos de Oracle hace unos años y los desarrolladores (su creador) open source crearon un fork llamado MariaDB
- MySQL y MariaDB son similares y compatibles en la mayoría de las operaciones



Instalación y config. de MySQL/MariaDB

- Describiremos a continuación el proceso de instalación de la base de datos que utilizaremos. Sería similar para otro tipo de base de datos.
- En nuestro caso todo este proceso ya lo ha realizado el paquete XAMMP
- Pasos:
 - Descargar
 - Descomprimir e instalar
 - Configurar
 - Arrancar
 - Conectar con el servidor
 - Instalar la extensión para MySQL de PHP



Instalación y config. de MySQL/MariaDB

□ Instalación y configuración de MySQL. 1: descargar

- Conectarse a la página web de MySQL/MariaDB
 - <https://www.mysql.com/>
 - <https://mariadb.org/>
- Seleccionar la sección *Downloads*
- Elegir la versión estable más reciente.
- Seleccionar la plataforma Windows y descargar el archivo para instalar



Instalación y config. de MySQL/MariaDB

□ **Instalación y configuración de MySQL/MariaDB. 2: descomprimir e instalar**

- Descomprimir el archivo descargado en una carpeta temporal
- Ejecutar el archivo setup.exe y seguir las indicaciones:
 - Seleccionar instalación típica
 - Aceptar la carpeta de instalación por defecto
- Pulsar el botón *Finish* para terminar la instalación y pasar a la configuración del servidor

Instalación y config. de MySQL

□ **Instalación y configuración de MySQL. 3: configurar**

- Al finalizar la instalación se ejecuta el asistente para la configuración del servidor. Seguir sus indicaciones:
 - Elegir la configuración estándar
 - Seleccionar la instalación como servicio Windows y Marcar la casilla para lanzar el servidor automáticamente al arrancar el sistema
 - Establecer una contraseña para el administrador (root)
 - Pulsar el botón *Execute* para realizar la configuración
 - Pulsar el botón *Finish* para finalizar el asistente

Instalación y config. de MySQL

- **Instalación y configuración de MySQL. 4: arrancar**
 - Si se indicó el arranque automático en la configuración, el servidor se inicia de forma automática al arrancar el sistema
 - En caso contrario hay que iniciarlo manualmente con Inicio > Programas > MySQL > MySQL Server 5.1 > MySQL Server Instance Config Wizard
- Cuando se instala una BBDD en un servidor lo normal es que este funcione como un servicio

Instalación y config. de MySQL

□ Instalación y configuración de MySQL. 5: conectar

– Formas de establecer la conexión con el servidor:

- Desde la línea de órdenes con Inicio > Programas > MySQL > MySQL Server 5.1 > MySQL Command Line Client

```
Enter password: *****
```

```
mysql>
```

```
...
```

```
mysql> exit
```

- Mediante alguna herramienta que proporcione una interfaz gráfica como phpMyAdmin
- Desde una página web mediante la interfaz que proporciona MySQL. Es lo que haremos con la biblioteca de funciones de MySQL que posee PHP

Instalación y config. de MySQL

□ **Instalación y configuración de MySQL. 5: conectar**

- Para conectar con el servidor hay que crear antes una cuenta de usuario y asignarle los correspondientes permisos de acceso. En general para una base de datos es conveniente definir al menos dos usuarios:
 - Un usuario anónimo que tenga permisos de lectura sobre las tablas que se estime adecuado
 - Un usuario administrador que tenga permisos para insertar, modificar o eliminar elementos de las tablas de la base de datos
- En cada conexión hay que indicar el nombre del usuario, su contraseña y la máquina desde la que se realiza la conexión (localhost si es la propia máquina donde reside el servidor, que es lo habitual en el acceso desde la Web)

Instalación y config. de MySQL

□ Acceso a MySQL desde PHP

- Para acceder a un servidor de base de datos (MySQL, MariaDB, PostGres, ...) en PHP se precisa tener instalada la extensión correspondiente.
- Para MySQL o MariaDB, se hace de la siguiente manera:
 - Editar el fichero php.ini y habilitar la extensión:
`extension=mysqli`
 - En XAMMP ya viene configurada la extensión para acceder a la base de datos, no es preciso modificar nada.

MySQL

□ Características de MySQL

- Modelo relacional, multiusuario

□ Tipos de datos

- Numéricos
 - tinyint, smallint, mediumint, int, integer, bigint
 - decimal, float, numeric
- Fecha y hora
 - date, time, datetime, year, timestamp
- Cadena
 - char, varchar
 - tinytext, text, mediumtext, longtext
 - tinyblob, blob, mediumblob, longblob
 - enum, set
- Debe elegirse adecuadamente el tipo y el tamaño de cada campo

MySQL

□ Operadores

- Aritméticos
 - +, -, *, /
- Comparación
 - =, !=, <=, <, >=, >, IS NULL, IS NOT NULL
- Lógicos
 - not (!), and (&&), or (||), xor

□ Funciones

- Funciones de cadena
- Funciones de comparación de cadenas
- Funciones numéricas
- Funciones de fecha y hora
- Funciones de agregado

Véase el [manual de referencia de MySQL](#)



Herramientas de administración: phpMyAdmin

- **phpMyAdmin** es una herramienta para la administración del servidor de bases de datos MySQL
- Dispone de una interfaz gráfica y es de libre distribución
- Permite realizar todo tipo de operaciones sobre bases de datos:
 - crear, borrar y modificar tablas
 - consultar, insertar, modificar y eliminar datos
 - definir usuarios y asignar permisos
 - realizar copias de seguridad
 - etc
- Está escrita en php y se ejecuta desde el navegador
- Si está instalada en la carpeta phpmyadmin, se ejecuta escribiendo en la barra de direcciones del navegador la url
`http://localhost/phpmyadmin/`
- Puede administrar bases de datos locales y remotas



phpMyAdmin

- Pasos para su instalación:
 - Descargar
 - Descomprimir
 - Configurar
 - Ejecutar

- En XAMMP ya tenemos instalada la aplicación. Se mostrará automáticamente al pulsar el botón “Admin” en la utilidad XAMMP Control

phpMyAdmin

- **Instalación y configuración de phpMyAdmin. 1: descargar**
 - Conectarse a la dirección <http://www.phpmyadmin.net>
 - Seleccionar *Download*
 - Seleccionar la versión más reciente.

- **Instalación y configuración de phpMyAdmin. 2: descomprimir**
 - Descomprimir debajo de la carpeta raíz de la web
 - Cambiar el nombre de la carpeta creada a phpmyadmin



phpMyAdmin

□ Instalación y configuración de phpMyAdmin. 3: configurar

- El fichero de configuración se llama config.inc.php
- Este fichero no existe sino que hay que crearlo. Para ello se hace una copia del fichero config.sample.inc.php, situado en la carpeta donde se haya instalado phpMyAdmin, y se le pone como nombre config.inc.php
- Seguidamente se modifica el fichero config.inc.php
- Configuración típica para un servidor local:

```
...  
$cfg['Servers'][$i]['host']    = 'localhost'; //MySQL hostname  
$cfg['Servers'][$i]['user']    = 'root';      //MySQL user  
$cfg['Servers'][$i]['password'] = 'clave';    //MySQL password  
...
```

siendo 'clave' la contraseña asignada al administrador (root) de MySQL durante su instalación



phpMyAdmin

- **Instalación y configuración de phpMyAdmin. 4: ejecutar**
 - Ejecutar Apache
 - Abrir el navegador y teclear la url `http://localhost/phpmyadmin`





Herramientas Gestión MySQL

- MySQL Workbench
- TOrA:
- HeidiSQL
- ...
- **Más info en**
<https://tramullas.com/clientes-graficos-para-mysql/>

Lenguaje SQL

- SQL (*Structured Query Language*) es el lenguaje que se utiliza para comunicarse con la base de datos
- Procedimiento de comunicación con la base de datos:



Lenguaje SQL

- Las instrucciones más habituales son SELECT, INSERT, UPDATE, DELETE
- Veamos su sintaxis básica y algunos ejemplos de uso
- Para ello utilizaremos una tabla **noticias** con cinco campos: un identificador único de la noticia, el título de la noticia, el texto de la noticia, la categoría de la noticia y la fecha de publicación de la noticia

noticias
id
título
texto
categoría
fecha

Lenguaje SQL

□ SELECT

Sintaxis:

```
SELECT expresión FROM tabla  
[WHERE condición]  
[ORDER BY {unsigned_integer | col_name | formula} [ASC | DESC] ,...]  
[LIMIT [offset,] row_count | row_count OFFSET offset]
```

Ejemplo:

```
SELECT * from noticias WHERE fecha=CURDATE() LIMIT 10 ORDER BY fecha  
DESC
```

Obtiene las noticias del día con un tope máximo de 10, ordenadas de la más reciente a la más antigua

Lenguaje SQL

□ INSERT

Sintaxis:

```
INSERT [INTO] nombre_tabla [(nombre_columna,...)]  
VALUES ((expresión | DEFAULT),...), (...), ...  
INSERT [INTO] nombre_tabla  
SET nombre_columna=(expresión | DEFAULT), ...
```

Ejemplo:

```
INSERT INTO noticias (id, titulo, texto, categoria, fecha) VALUES  
    (37, "Nueva promoción en Nervión", "145 viviendas de lujo en  
    urbanización ajardinada situadas en un entorno privilegiado",  
    "promociones", CURDATE())
```

Inserta una noticia con los valores indicados

Lenguaje SQL

□ UPDATE

Sintaxis:

```
UPDATE nombre_tabla  
SET nombre_columna1=expr1 [, nombre_columna2=expr2 ...]  
[WHERE condición]  
[ORDER BY ...]  
[LIMIT row_count]
```

Ejemplo:

```
UPDATE noticias SET categoria = "ofertas" WHERE id=37
```

Modifica la categoría de la noticia con id=37 de la tabla

Lenguaje SQL

▣ DELETE

Sintaxis:

```
DELETE FROM nombre_tabla  
[WHERE condición]  
[ORDER BY ...]  
[LIMIT row_count]
```

Ejemplo:

```
DELETE FROM noticias WHERE fecha < CURDATE()-10
```

Borra las noticias con más de 10 días de antigüedad



Funciones de PHP para el acceso a bases de datos MySQL

- Los pasos para acceder desde PHP a una base de datos son los siguientes:
 - Conectar con el servidor de bases de datos
 - Seleccionar una base de datos
 - Enviar la instrucción SQL a la base de datos
 - Obtener y procesar los resultados
 - Cerrar la conexión con el servidor de bases de datos

Acceso a bases de datos MySQL

OBSOLETO con funciones

- Las funciones concretas de MySQL que realizan estas operaciones son:

- Conectar con el servidor de bases de datos:

[mysqli_connect\(\)](#) / [mysql_connect\(\)](#)

□ Nombre sin "i" esta obsoleto

- Seleccionar una
– base de datos:

[mysqli_select_db\(\)](#)

- Enviar la instrucción SQL a la base de datos:

[mysqli_query\(\)](#) / [mysql_query\(\)](#)

- Obtener y procesar los resultados (hay más ...):

[mysqli_fetch_array\(\)](#) / [mysql_fetch_array\(\)](#)

- Cerrar la conexión con el servidor de bases de datos:

[mysqli_close\(\)](#) / [mysql_close\(\)](#)

Acceso a bases de datos MySQL con Clase `mysqli`

- Actualmente el acceso a la base de datos se realiza utilizando objetos de la clase `mysqli`, la cual contiene los métodos vistos anteriormente y muchos más
- Extensión MySQL mejorada – php.net
- Operaciones:
 - Conexión: Constructor [Conexiones](#)
 - [Ejecutar sentencias](#)

Acceso a bases de datos MySQL

- Conectar con el servidor de bases de datos: **mysqli_connect()**
 - Devuelve un identificador de la conexión en caso de éxito y false en caso contrario

- Sintaxis:

```
$conexion = mysqli_connect (
                        servidor, username, password, db);
```

- Ejemplo:

```
$conexion = mysqli_connect (
    "localhost", "cursophp-ad", "php.hph", "bd")
or die ("No se puede conectar con el servidor");
```

Este esquema simplificado no se suele utilizar. Se muestra por lo curioso de la expresión, pues utiliza lógica de cortocircuito. Cuando al evaluar una parte de una expresión ya conocemos el resultado no se evalúa la otra. Si hay conexión una parte será TRUE (conexión), por tanto no se ejecutará la función die()

Acceso a bases de datos MySQL

- Enviar la instrucción SQL a la base de datos: **mysqli_query()**
 - Devuelve un identificador o true (dependiendo de la instrucción) si la instrucción se ejecuta correctamente y false en caso contrario

- Sintaxis:

```
$consulta = mysqli_query ($conexión, instrucción);
```

- Ejemplo:

```
$consulta = mysqli_query ($conexión, "select * from noticias")  
or die ("Fallo en la consulta");
```


Acceso a bases de datos MySQL

- Obtener y procesar los resultados: **mysqli_fetch_array()**
 - En el caso de que la instrucción enviada produzca unos resultados, `mysqli_query()` devuelve las filas de la tabla afectadas por la instrucción
 - `mysqli_num_rows()` devuelve el número de filas afectadas
 - Para obtener las distintas filas del resultado se utiliza la función `mysqli_fetch_array()`, que obtiene una fila del resultado en un array asociativo cada vez que se invoca
 - Las funciones `mysqli_fetch_row()`, `mysqli_fetch_assoc()`, `mysqli_fetch_object()` son iguales pero devuelven el registro en otro formato.

- Sintaxis:

```
$nfilas = mysqli_num_rows ($consulta);  
$reg= mysqli_fetch_array ($consulta);
```

- El nº de filas normalmente no se usa. Leeremos hasta que el registro esté vacío (sea NULL)

Acceso a bases de datos MySQL

□ Ejemplo:

noticias				
1	Título 1	Texto 1	ofertas	05/02/2004
2	Título 2	Texto 2	promociones	05/02/2004
3	Título 3	Texto 3	promociones	04/02/2004
4	Título 4	Texto 4	costas	01/02/2004
5	Título 5	Texto 5	promociones	31/01/2004

Instrucción:

```
select * from noticias where categoria="promociones"
```

Acceso a bases de datos MySQL

□ Ejemplo:

noticias				
1	Título 1	Texto 1	ofertas	05/02/2004
2	Título 2	Texto 2	promociones	05/02/2004
3	Título 3	Texto 3	promociones	04/02/2004
4	Título 4	Texto 4	costas	01/02/2004
5	Título 5	Texto 5	promociones	31/01/2004

← \$consulta
← \$nfilas=3
←

Instrucción:

```
select * from noticias where categoria="promociones"
```

Acceso a bases de datos MySQL

▣ Obtención de los registros:

```
$nfilas = mysqli_num_rows ($consulta);  
echo "<p>$nfilas encontradas</p>";  
  
while($reg = mysqli_fetch_array ($consulta)) {  
    procesar registro ($reg)  
    fila i-ésima de los resultados  
}
```

Acceso a bases de datos MySQL

□ Obtener los resultados: **mysqli_fetch_array()**

- Para acceder a un campo determinado de una fila se usa la siguiente sintaxis:

```
$reg["nombre_campo"] // por ser un array asociativo
$reg[$i]              // $i=índice del campo desde 0
** Se copia dos veces la información **
Si queremos solo el nombre de campo utilizaremos
mysqli_fetch_assoc()
```

□ Ejemplo:

```
while($reg = mysql_fetch_array ($consulta)){
    print "Título: " . $reg["titulo"];
    print "Fecha: " . $reg["fecha"];
}
```

□ Con *mysqli_fetch_object()*: \$fila->nombre_campo

Acceso a bases de datos MySQL

- ❑ Cerrar la conexión con el servidor de bases de datos:
mysqli_close()

- ❑ Sintaxis:

```
mysqli_close ($conexion);
```

- ❑ Ejemplo

```
mysqli_close ($conexion);
```

Utilizando objeto mysqli

- Hacemos lo mismo pero con notación O.O.
- Conexión: [Véase documentación php.net](#)

Se crea objeto desde el que operaremos. El objeto es la conexión

```
<?php
$mysqli=new mysqli(
    "ejemplo.com", "usuario", "contraseña", "basedatos");
if ($mysqli->connect_errno) {
    echo "Falló la conexión a MySQL: (" .
        $mysqli->connect_errno . ") " . $mysqli->connect_error;
}
```

Utilizando objeto mysqli

□ Ejecutar consultas de selección

```
$res=mysqli->query("SELECT id FROM test ORDER BY id ASC");
```

```
while ($fila=$res->fetch_assoc()) {  
    echo "id=".$fila['id']."\n";  
}
```


Utilizando objeto mysqli

- Ejecutar consultas de actualización
Se ejecutan igual

```
if (!$mysqli->query("DROP TABLE IF EXISTS test")) {  
    echo "Falló: (" . $mysqli->errno . ") " . $mysqli->error;  
}  
if (!$mysqli->query("CREATE TABLE test(id INT)")) {  
    echo "Falló: (" . $mysqli->errno . ") " . $mysqli->error;  
}  
if (!$mysqli->query("INSERT INTO test(id) VALUES (1)")) {  
    echo "Falló: (" . $mysqli->errno . ") " . $mysqli->error;  
}
```

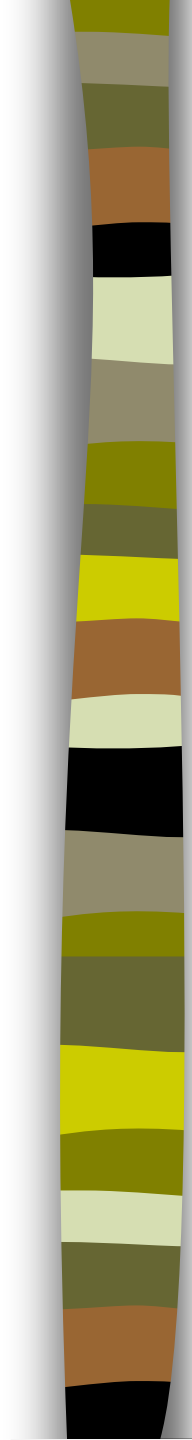
Acceso a bases de datos MySQL

```
<?php
$mysqli = new mysqli("ejemplo.com", "usuario", "contraseña", "basedatos")
;
if ($mysqli->connect_errno) {
    echo "Falló la conexión a MySQL: (" . $mysqli-
>connect_errno . ") " . $mysqli->connect_error;
}

$resultado = $mysqli->query("SELECT id FROM test ORDER BY id ASC");

echo "Mostramos los resultados...\n";
while ($fila = $resultado->fetch_assoc()) {
    echo " id = " . $fila['id'] . "\n";
}

// Aunque no pongmos mysqli->close() la conexión se cierra igual
```



Utilizando objeto mysqli

Sentencias preparadas

- Una sentencia preparada o una sentencia parametrizada se usa para ejecutar la misma sentencia repetidamente con gran eficiencia.
- Tareas a realizar:
 - Etapa 1: preparación
 - Etapa 2: vincular y ejecutar
 - Etapa 3: ejecución – Podremos volver a la etapa 2

Utilizando objeto mysqli

Sentencias preparadas – Preparación

```
/* Sentencia no preparada */
if (!$mysqli->query("DROP TABLE IF EXISTS test") || !$mysqli-
>query("CREATE TABLE test(id INT)")) {
    echo "Falló la creación de la tabla: (" . $mysqli-
>errno . ") " . $mysqli->error;
}

/* Sentencia preparada, etapa 1: preparación */
$sentencia = $mysqli->prepare("INSERT INTO test(id) VALUES (?)");
if (!$sentencia) {
    echo "Falló la preparación: (" . $mysqli-
>errno . ") " . $mysqli->error;
}
```

Los parámetros se ponen con ?
Representarán valores, y se pondrán tantos
como se precisen

Utilizando objeto mysqli

Sent. preparadas – Vincular y Ejecutar

```
/* Sentencia preparada, etapa 2: vincular y
$id = 1;
if (!$sentencia->bind_param("i", $id)) {
    echo "Falló la vinculación de parámetros: (" . $sentencia-
>errno . ") " . $sentencia->error;
}

// Ejecutar: Si repetimos la siguiente línea se toma el valor de $i,
que puede haber cambiado
if (!$sentencia->execute()) {
    echo "Falló la ejecución: (" . $sentencia-
>errno . ") " . $sentencia->error;
}
```

1º Tipo parámetro : i = entero
2º variable vinculada: \$id

Consulta avanzada de tablas

- Objetivo:
 - mostrar los resultados de la consulta **divididos en bloques de un número determinado de elementos** (por ejemplo, de 5 en 5)
- Requisitos:
 - Recuperar un número limitado de elementos de la tabla
 - Implementar un mecanismo de navegación que permita avanzar al siguiente o volver al anterior bloque de elementos

SQL específico de la base de datos MySQL, en otros entornos se puede hacer lo mismo pero de otra forma

Consulta avanzada de tablas

- Para recuperar un número fijo de elementos de una tabla se utiliza la opción **LIMIT** de la orden SELECT. Así, por ejemplo,

```
SELECT * from noticias LIMIT 0, 5
```

recupera los 5 primeros elementos de la tabla. Y en general,

```
SELECT * from noticias LIMIT $comienzo, $num
```

recupera \$num elementos a partir de la posición \$comienzo

Consulta avanzada de tablas

- La variable \$num tendrá un valor constante (en este caso 5), mientras que la variable \$comienzo se incrementará o decrementará en 5 unidades al pasar a la página siguiente o anterior
- Para ello se pasará la variable como parámetro en el enlace asociado al botón correspondiente
- Por ejemplo, el código para el botón siguiente será:

```
"<A HREF='" . $_SERVER['PHP_SELF'] . "?comienzo=" .  
($comienzo + $num) . "'>Sigüiente</A>"
```

- Habrá que comprobar previamente que el nuevo valor de comienzo es válido, es decir, que se encuentra dentro de los límites de la tabla devuelta por la consulta



Consulta avanzada de tablas

□ Objetivo:

- Mostrar los resultados de una consulta de manera que se puedan **filtrar en función del valor de una determinada columna** de la tabla

□ Requisitos:

- Recuperar de una tabla los elementos que cumplan una determinada condición
- Permitir seleccionar un valor de entre los valores posibles de una columna

Consulta avanzada de tablas

- Para recuperar los elementos de una tabla que cumplen una condición se utiliza la opción **WHERE** de la orden **SELECT**. Por ejemplo,

```
SELECT * from noticias WHERE categoria='ofertas'
```

recupera las noticias cuya categoría tiene el valor “ofertas”. Y en general,

```
SELECT * from noticias WHERE categoria ='$categoria'
```

recupera las noticias cuya categoría tiene el valor dado por la variable \$categoria

- La variable \$categoria se obtendrá de un elemento **SELECT**