

# Índice: Tema 4.2

## **4.2 Conexión con Bases de Datos**

4.2.1 Envío y recepción de datos mediante formularios

4.2.2 Administración de MySQL con phpMyAdmin

4.2.3 Conexión con Bases de Datos desde PHP

# Índice: Tema 4.2

## **4.2 Conexión con Bases de Datos**

### **4.2.1 Envío y recepción de datos mediante formularios**

4.2.2 Administración de MySQL con phpMyAdmin

4.2.3 Conexión con Bases de Datos desde PHP



## 4.2 CONEXIÓN CON BASES DE DATOS

### 4.2.1 ENVÍO Y RECEPCIÓN DE DATOS MEDIANTE FORMULARIOS



#### ➤ Interacción con el usuario

- ✓ Los formularios HTML constituyen el primer mecanismo de interacción entre las páginas web y el usuario. El usuario rellena el formulario escribiendo datos o seleccionándolos de entre varios posibles.
- ✓ La información suministrada en cada campo se almacena en una variable con el mismo nombre del atributo **name** del elemento correspondiente, y puede ser:
  - Enviada a una dirección de correo electrónico.
  - Procesada directamente por la página (por ejemplo, con JavaScript).
  - Enviada al servidor para su procesamiento (por ejemplo, una página en PHP).
- ✓ Para indicar el destino al que serán enviados los datos obtenidos se utiliza el atributo **action** del elemento **<form>**:

```
<form action="http://localhost/pagina.php">
```



## 4.2 CONEXIÓN CON BASES DE DATOS

### 4.2.1 ENVÍO Y RECEPCIÓN DE DATOS MEDIANTE FORMULARIOS



#### ➤ Métodos de envío (1)

- ✓ Además, el atributo **method** especifica la forma en la que se deben transmitir los datos del formulario. Hay dos valores posibles para dicho atributo:
- ✓ **get** Es el método predeterminado.
  - Los datos se añaden al final del URL precedidos por un signo de interrogación (?).
  - Pares **clave=valor** separados por **&**.
  - <http://localhost/pagina.php?nombre=Alberto&apellido=Herran>
  - En la página procesadora se accede a los datos mediante el array **\$\_GET**.
- ✓ **post** Es el método habitual en envíos con formularios.
  - Los datos se incrustan en el encabezamiento de la solicitud HTTP, no quedando a la vista.
  - En la página procesadora se accede a los datos mediante el array **\$\_POST**.
- ✓ Además, en ambos casos se dispone del array **\$\_REQUEST**.

#### ➤ Métodos de envío (2)

- ✓ Prueba los métodos anteriores para el envío de los datos introducidos en un par de campos de texto:

The image shows two scenarios of data submission from a web form to a server page.

**Scenario 1 (GET):** The left browser window shows a form with 'Nombre: Alberto' and 'Apellido: Herrán'. A red arrow points from the 'Enviar' button to the right browser window. The right window's address bar shows the URL: `http://localhost/GIE/AW/Tema4/Ejercicio11/paginaget.php?nombre=Alberto&apellido=Herr%C3%A1n`. The page content displays: 'Mediante GET te llamas Alberto Herrán. Mediante REQUEST te llamas Alberto Herrán.'

**Scenario 2 (POST):** The left browser window is identical to the first. A red arrow points from the 'Enviar' button to the right browser window. The right window's address bar shows the URL: `http://localhost/GIE/AW/Tema4/Ejercicio11/paginapost.php`. The page content displays: 'Mediante POST te llamas Alberto Herrán. Mediante REQUEST te llamas Alberto Herrán.'



#### ➤ Métodos de envío (3)

- ✓ La propia página puede ser la encargada de procesar los datos:

```
<body>
```

```
<?php if(isset($_POST["nombre"])) { ?>
```

```
<p>Te llamas <?php echo $_POST["nombre"]; ?>.</p>
```

```
<?php } else { ?>
```

```
<form action="<?php echo($_SERVER["PHP_SELF"]); ?>" method="post">
```

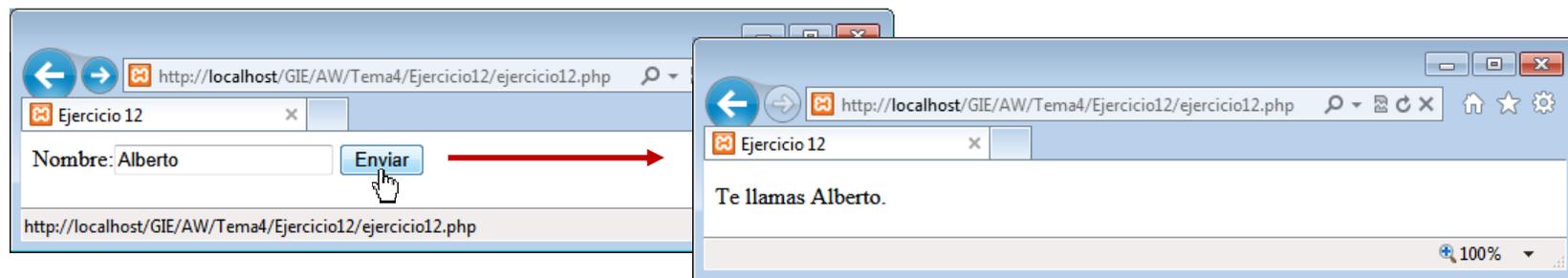
```
Nombre: <input type="text" name="nombre" />
```

```
<input type="submit" value="Enviar" />
```

```
</form>
```

```
<?php } ?>
```

```
</body>
```





## 4.2 CONEXIÓN CON BASES DE DATOS

### 4.2.1 ENVÍO Y RECEPCIÓN DE DATOS MEDIANTE FORMULARIOS



#### ➤ Otros campos (1)

- ✓ Como se acaba de ver, la información suministrada en un cuadro de texto (**text**) se almacena en una variable con el nombre especificado en su atributo **name**, lo mismo ocurre con un área de texto (**textarea**).
- ✓ Para las casillas de verificación (**checkbox**) se envía una variable con el nombre de su atributo **name** sólo si la casilla está activada. El valor de la variable es **on** a no ser que se haya especificado otro mediante su atributo **value**.
- ✓ En el caso de un grupo de botones de radio se asigna a la variable especificada en su atributo **name** el valor especificado en el atributo **value** del botón activado.
- ✓ Finalmente, en las listas de selección (**select**) el valor especificado en el atributo **value** de la opción seleccionada se asigna a la variable especificada en el atributo **name** de la lista. Si la lista es de selección múltiple los valores asociados a las opciones seleccionadas se almacenan en un array, por lo que el atributo **name** de la lista debe contener el nombre de la variable seguido por **[]**.



#### ➤ Otros campos (2)

- ✓ Repetir con PHP el ejercicio 37 del Tema 3. Seguir el patrón del Ejercicio 12.

El valor del cuadro de texto es TEXTO.

El valor del area de texto es ÁREA.

La opción seleccionada en la lista tiene un valor de 1.

El valor de la casilla 1 es casilla 1.

El valor de la casilla 2 es

**Notice:** Undefined index: casilla2 in C:\xampp\htdocs\GIE\AW\Tema4\Ejercicio13\ejercicio13.php on line 40

Ha seleccionado el botón de radio 1.



#### ➤ Funciones de utilidad (1)

##### ✓ Validar elementos obligatorios:

- Para tener la certeza de que se ha escrito algo en un elemento obligatorio, podemos usar la función `strlen()`.

```
if(strlen($_POST["nombre"])==0) {  
    $errores[]="Tiene que introducir su nombre";  
}
```

##### ✓ Validar elementos numéricos:

- Para asegurarnos que un valor enviado es un entero o un número en coma flotante, podemos usar las funciones `intval()` y `floatval()` que convierten de cadena a número. Una vez que hemos convertido a número, realizamos la conversión inversa con `strval()` y comparamos.

```
if($_POST["precio"]!=strval(floatval($_POST["precio"]))) {  
    $errores[]="Introduzca un precio válido";  
}
```



#### ➤ Funciones de utilidad (2)

##### ✓ Validar cadenas:

- Para comprobar que se ha introducido texto que no sean únicamente caracteres en blanco, podemos usar `trim()` y `strlen()`.

```
if(strlen(trim($_POST["apellido"]))==0) {  
    $errores[]="Tiene que introducir su apellido";  
}
```

##### ✓ Rangos numéricos:

- Comprobamos primero que se trata de un número y después si está entre el rango.

```
if($_POST["edad"]!=intval(intval($_POST["edad"]))) {  
    $errores[]="Introduzca una edad válida";  
}  
elseif(($_POST["edad"]<18)||($_POST["edad"]>65)) {  
    $errores[]="Su edad debe estar entre 18 y 65 años";  
}
```



### ➤ Funciones de utilidad (3)

✓ Utilice el código anterior para programar el siguiente formulario con validación:

Nombre   
Apellido   
Edad   
Precio

Tiene que introducir su nombre.  
Tiene que introducir su apellido.  
Introduzca una edad válida.  
Introduzca un precio válido.

Nombre   
Apellido   
Edad   
Precio

Tiene que introducir su nombre.  
Es necesario que introduzca su apellido.  
Introduzca una edad válida.  
Introduzca un precio válido.

Nombre   
Apellido   
Edad   
Precio

Es necesario que introduzca su apellido.  
Introduzca una edad válida.  
Introduzca un precio válido.

Nombre   
Apellido   
Edad   
Precio

Introduzca una edad válida.  
Introduzca un precio válido.

Nombre   
Apellido   
Edad   
Precio

Introduzca una edad válida.  
Introduzca un precio válido.

Nombre   
Apellido   
Edad   
Precio

Su edad debe estar entre 18 y 65 años.  
Introduzca un precio válido.

Nombre   
Apellido   
Edad   
Precio

Introduzca un precio válido.

Nombre   
Apellido   
Edad   
Precio

Introduzca un precio válido.

Nombre   
Apellido   
Edad   
Precio

Te llamas Alberto.  
Te apellidas Herrán.  
Tienes 19 años.  
Vales 25.3 euros.

# Índice: Tema 4.2

## **4.2 Conexión con Bases de Datos**

4.2.1 Envío y recepción de datos mediante formularios

**4.2.2 Administración de MySQL con phpMyAdmin**

4.2.3 Conexión con Bases de Datos desde PHP



## 4.2 CONEXIÓN CON BASES DE DATOS

### 4.2.1 ADMINISTRACIÓN DE MYSQL CON PHPMYADMIN



#### ➤ MySQL

- ✓ MySQL es un sistema de gestión de bases de datos relacionales rápido, sólido y flexible.
- ✓ Es ideal para crear bases de datos con acceso desde páginas web dinámicas, o para cualquier otra solución profesional que implique almacenar datos, teniendo la posibilidad de realizar múltiples y rápidas consultas.
- ✓ Características:
  - Sistema cliente/servidor.
  - Permite trabajar como servidor multiusuario y de procesamiento múltiple.
  - Para cada conexión con el servidor se crea un subproceso para manejar la solicitud del cliente, controlando el acceso simultáneo de varios clientes a los datos.
  - Se puede controlar el acceso a sólo usuarios autorizados.
  - Utiliza el lenguaje SQL (lenguaje de consulta más utilizado para acceder a bases de datos).



## 4.2 CONEXIÓN CON BASES DE DATOS

### 4.2.1 ADMINISTRACIÓN DE MYSQL CON PHPMYADMIN



#### ➤ phpMyAdmin

- ✓ phpMyAdmin es una alternativa sencilla e intuitiva para administrar las bases de datos de MySQL a través de un servidor web.
- ✓ La aplicación phpMyAdmin no es más que un conjunto de páginas escritas en PHP y alojadas en directorio web del servidor.
- ✓ Mediante las diferentes páginas de la aplicación se puede:
  - Consultar las bases de datos disponibles.
  - Crear nuevas bases de datos así como tablas.
  - Realizar consultas, insertar y borrar registros.
  - Administrar usuarios y sus privilegios.
  - Hacer copias de seguridad de las bases de datos.
  - Etc...



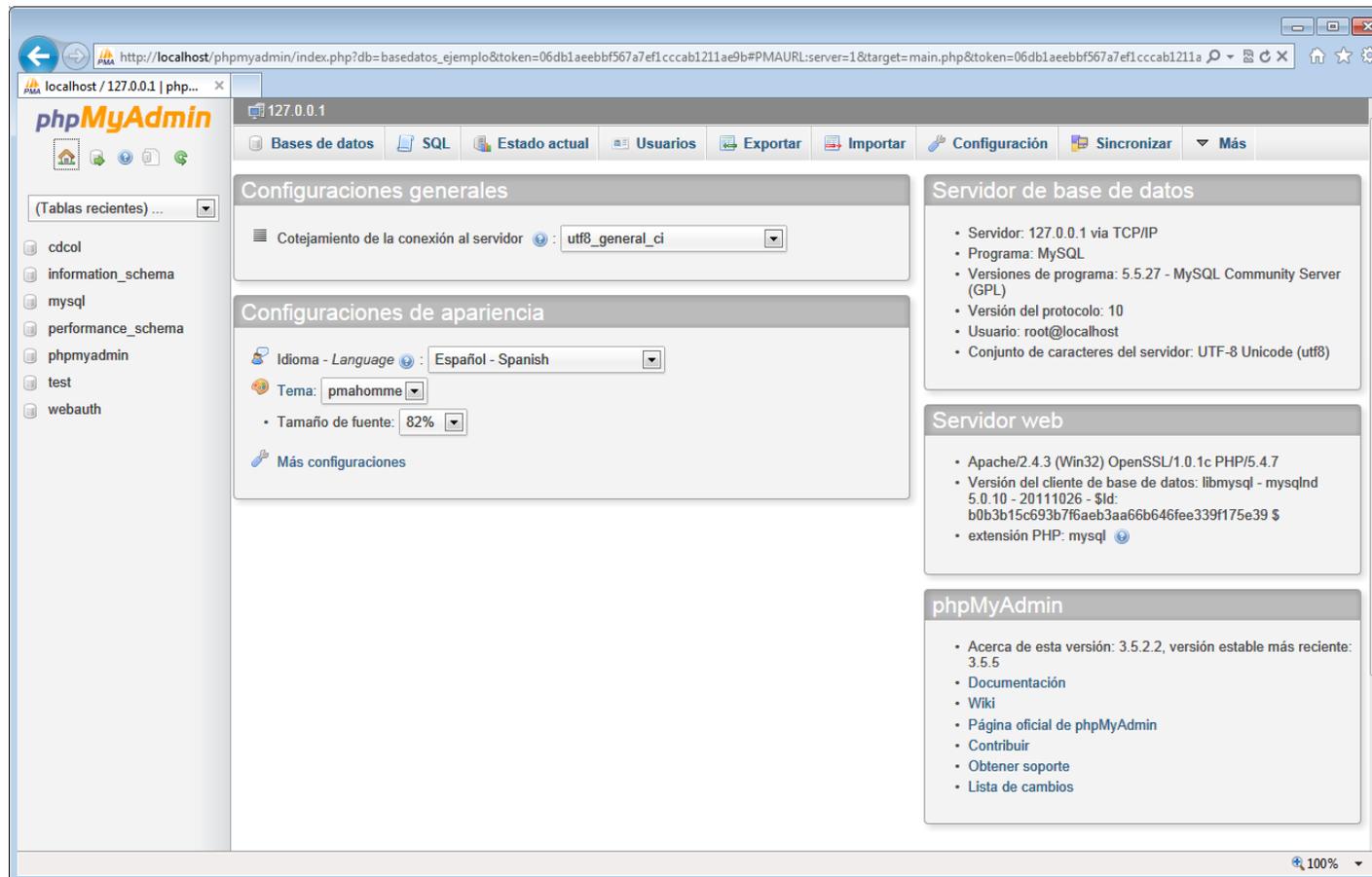
## 4.2 CONEXIÓN CON BASES DE DATOS

### 4.2.1 ADMINISTRACIÓN DE MYSQL CON PHPMYADMIN



#### ➤ phpMyAdmin (2)

- ✓ El acceso a la página principal de phpMyAdmin se hace desde el panel de control:





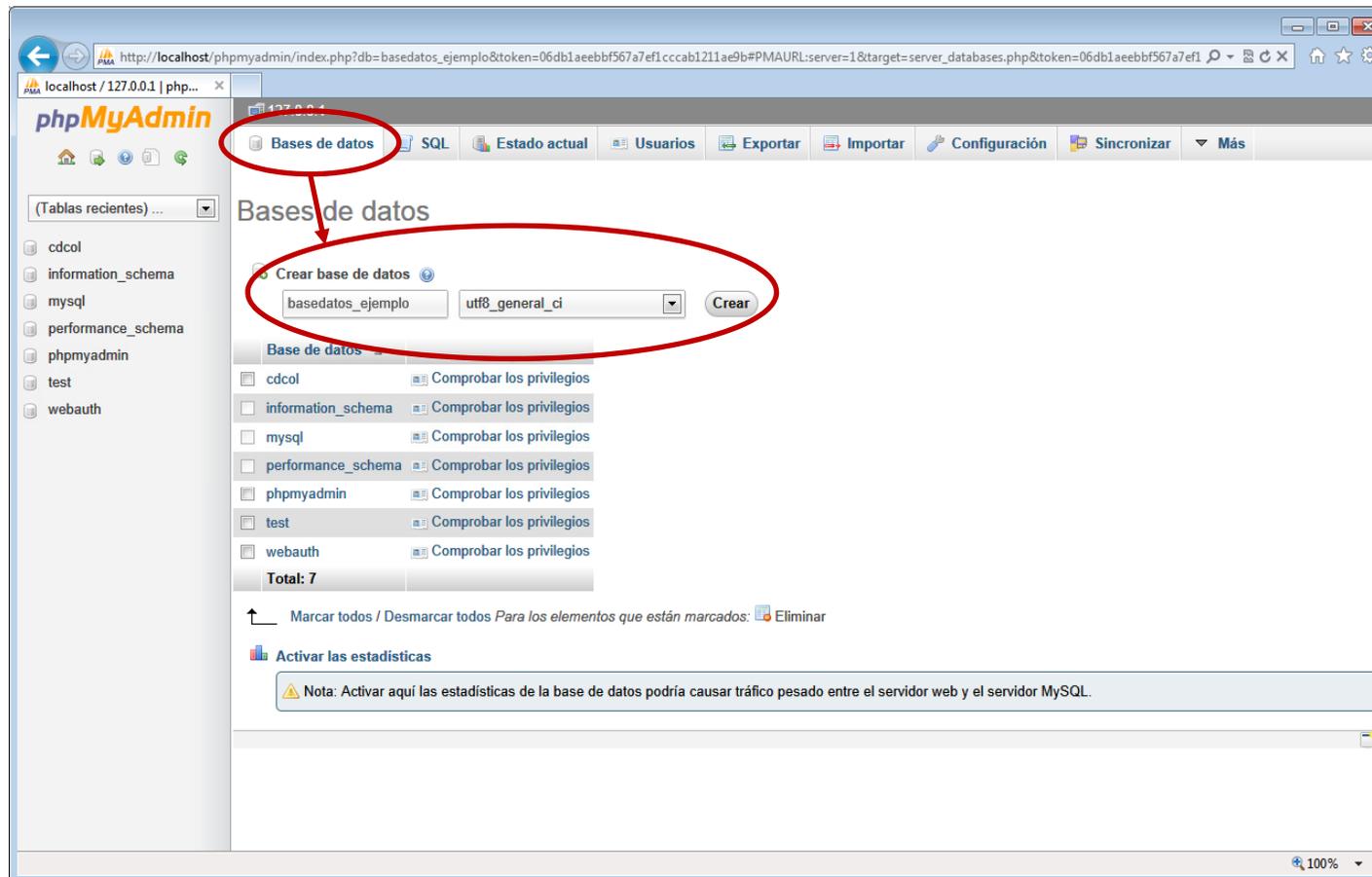
## 4.2 CONEXIÓN CON BASES DE DATOS

### 4.2.1 ADMINISTRACIÓN DE MYSQL CON PHPMYADMIN



#### ➤ Creación de una base de datos

- ✓ Crear una base de datos denominada **“basedatos\_ejemplo”** utf8:





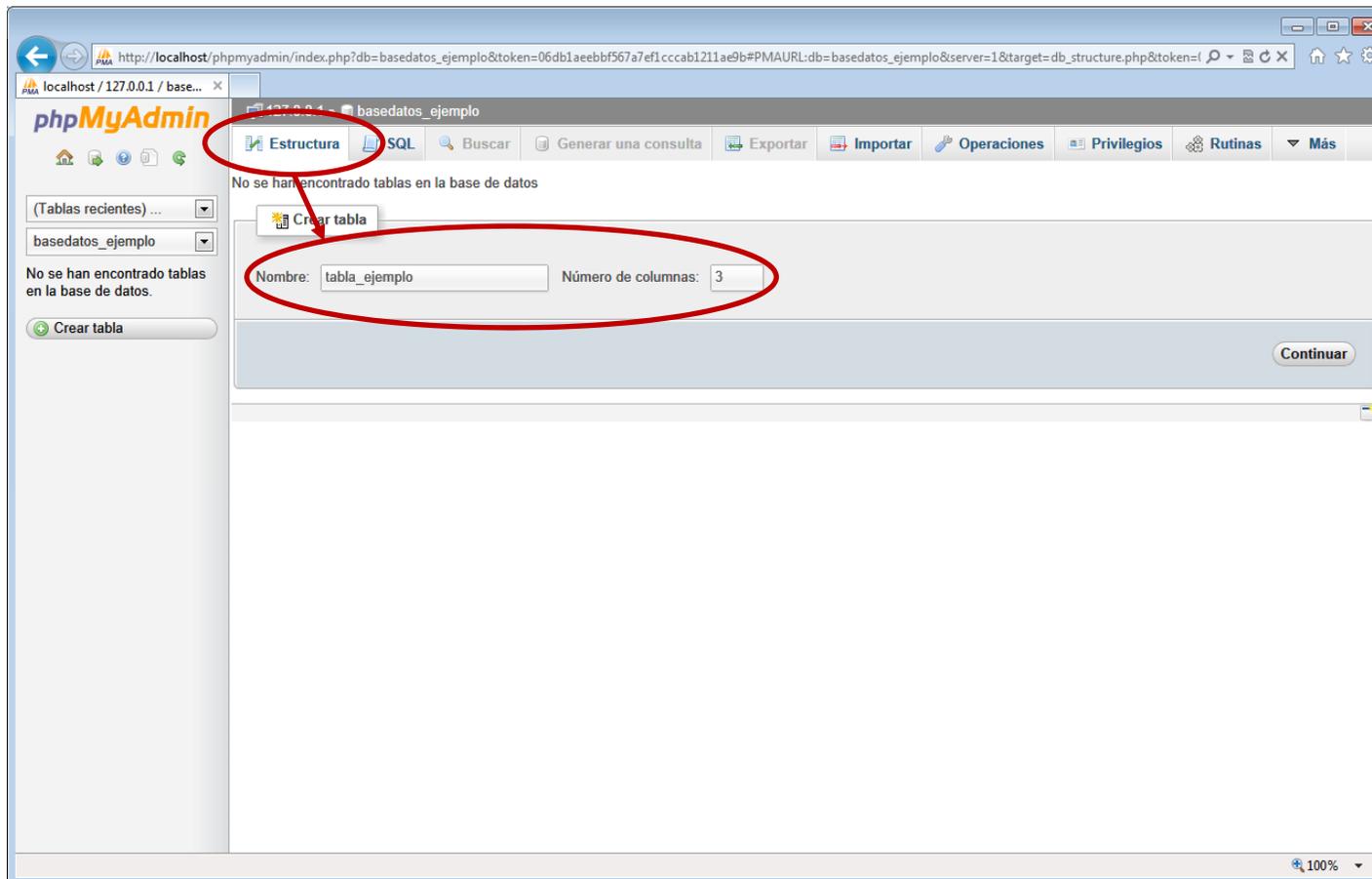
## 4.2 CONEXIÓN CON BASES DE DATOS

### 4.2.1 ADMINISTRACIÓN DE MYSQL CON PHPMYADMIN



#### ➤ Creación de una tabla (1)

- ✓ Pulsar sobre la BDB creada y crear una tabla “**tabla\_ejemplo**” con 4 columnas:





## 4.2 CONEXIÓN CON BASES DE DATOS

### 4.2.1 ADMINISTRACIÓN DE MYSQL CON PHPMYADMIN



#### ➤ Creación de una tabla (2)

- ✓ Añadir las siguientes columnas para cada registro de la tabla:

The screenshot shows the phpMyAdmin interface for creating a table named 'tabla\_ejemplo'. The table structure is defined as follows:

Nombre	Tipo	Longitud/Valores	Predeterminado	Cotejamiento	Atributos	Nulo	Índice	A_I
idPersona	INT		Ninguno			<input type="checkbox"/>	PRIMARY	<input checked="" type="checkbox"/>
dni	INT		Ninguno			<input type="checkbox"/>	---	<input type="checkbox"/>
nombre	VARCHAR	50	Ninguno			<input type="checkbox"/>	---	<input type="checkbox"/>
apellido	VARCHAR	50	Ninguno			<input type="checkbox"/>	---	<input type="checkbox"/>

Additional settings shown in the interface include: Motor de almacenamiento: InnoDB, Cotejamiento: (empty), and Comentarios de la tabla: (empty). A red circle highlights the 'idPersona' column and its 'PRIMARY' index setting.



## 4.2 CONEXIÓN CON BASES DE DATOS

### 4.2.1 ADMINISTRACIÓN DE MYSQL CON PHPMYADMIN



#### ➤ Añadir usuario y contraseña (1)

- ✓ Tras crear la tabla aparece la siguiente pantalla donde pulsamos en “Privilegios”:

The screenshot shows the phpMyAdmin interface for a database named 'basedatos\_ejemplo'. The 'Privilegios' tab is highlighted with a red circle. Below the navigation menu, there is a table with the following data:

Tabla	Acción	Filas	Tipo	Cotejamiento	Tamaño	Residuo a depurar
tabla_ejemplo	Examinar Estructura Buscar Insertar Vaciar Eliminar	0	InnoDB	utf8_general_ci	1.6 KB	-
1 tabla	Número de filas	0	InnoDB	utf8_general_ci	1.6 KB	0 B

Below the table, there are options to 'Marcar todos / Desmarcar todos' and 'Para los elementos que están marcados:'. At the bottom, there is a 'Crear tabla' section with input fields for 'Nombre:' and 'Número de columnas:', and a 'Continuar' button.



## 4.2 CONEXIÓN CON BASES DE DATOS

### 4.2.1 ADMINISTRACIÓN DE MYSQL CON PHPMYADMIN



#### ➤ Añadir usuario y contraseña (2)

- ✓ Dentro de la pestaña privilegios le damos a “Agregar usuario”:

The screenshot shows the phpMyAdmin interface for a database named 'basedatos\_ejemplo'. The 'Privilegios' tab is active, displaying a table of users with access to the database. The table has columns for 'Usuario', 'Servidor', 'Tipo', 'Privilegios', 'Conceder', and 'Acción'. Two users are listed: 'root' on 'linux' with 'global ALL PRIVILEGES' and 'root' on 'localhost' with 'global ALL PRIVILEGES'. Below the table, there is a 'Nuevo' section with a button labeled 'Agregar usuario', which is circled in red. The interface also includes a sidebar with navigation options and a top menu with various tools like 'Estructura', 'SQL', 'Buscar', etc.

Usuario	Servidor	Tipo	Privilegios	Conceder	Acción
root	linux	global	ALL PRIVILEGES	Sí	<a href="#">Editar los privilegios</a>
root	localhost	global	ALL PRIVILEGES	Sí	<a href="#">Editar los privilegios</a>



## 4.2 CONEXIÓN CON BASES DE DATOS

### 4.2.1 ADMINISTRACIÓN DE MYSQL CON PHPMYADMIN



#### ➤ Añadir usuario y contraseña (3)

- ✓ Escogemos un nombre de usuario, el servidor local (localhost) y una contraseña (en este caso se ha utilizado el botón “Generar”). Más abajo se otorgan todos los privilegios para la base de datos seleccionada y todos los privilegios globales:

**Información de la cuenta**

Nombre de usuario: Use el campo de te: aherranq

Servidor: Local localhost

Contraseña: Use el campo de te: .....

Debe volver a escribir: .....

Generar contraseña: Generar 5748fT8Ss9GCFExJ

**Base de datos para el usuario**

Ninguna  
 Crear base de datos con el mismo nombre y otorgar todos los privilegios  
 Otorgar todos los privilegios para la base de datos "basedatos\_ejemplo"

**Privilegios globales (Marcar todos / Desmarcar todos)**

*Nota: Los nombres de los privilegios de MySQL están expresados en inglés*

Datos	Estructura	Administración
<input checked="" type="checkbox"/> SELECT	<input checked="" type="checkbox"/> CREATE	<input checked="" type="checkbox"/> GRANT
<input checked="" type="checkbox"/> INSERT	<input checked="" type="checkbox"/> ALTER	<input checked="" type="checkbox"/> SUPER
<input checked="" type="checkbox"/> UPDATE	<input checked="" type="checkbox"/> INDEX	<input checked="" type="checkbox"/> PROCESS
<input checked="" type="checkbox"/> DELETE	<input checked="" type="checkbox"/> DROP	<input checked="" type="checkbox"/> RELOAD
<input checked="" type="checkbox"/> FILE	<input checked="" type="checkbox"/> CREATE TEMPORARY TABLES	<input checked="" type="checkbox"/> SHUTDOWN
	<input checked="" type="checkbox"/> SHOW VIEW	<input checked="" type="checkbox"/> SHOW DATABASES
	<input checked="" type="checkbox"/> CREATE ROUTINE	<input checked="" type="checkbox"/> LOCK TABLES
	<input checked="" type="checkbox"/> ALTER ROUTINE	<input checked="" type="checkbox"/> REFERENCES
	<input checked="" type="checkbox"/> EXECUTE	<input checked="" type="checkbox"/> REPLICATION CLIENT
	<input checked="" type="checkbox"/> CREATE VIEW	<input checked="" type="checkbox"/> REPLICATION SLAVE
	<input checked="" type="checkbox"/> EVENT	<input checked="" type="checkbox"/> CREATE USER

Agregar usuario Cancelar



## 4.2 CONEXIÓN CON BASES DE DATOS

### 4.2.1 ADMINISTRACIÓN DE MYSQL CON PHPMYADMIN



#### ➤ Gestión de la tabla desde phpMyAdmin

- ✓ Dentro de la tabla se pueden insertar, modificar y eliminar registros:

The screenshot shows the phpMyAdmin interface for a database named 'basedatos\_ejemplo' and a table named 'tabla\_ejemplo'. The 'Insertar' button in the top navigation bar is circled in red. Below the table structure, the 'Cambiar' and 'Eliminar' buttons for the 'apellido' column are also circled in red. The table structure is as follows:

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra	Acción
1	idPersona	int(11)			No	Ninguna	AUTO_INCREMENT	Cambiar Eliminar Primaria Único Índice Más
2	dni	int(11)			No	Ninguna		Cambiar Eliminar Primaria Único Índice Más
3	nombre	varchar(50)	utf8_spanish_ci		No	Ninguna		Cambiar Eliminar Primaria Único Índice Más
4	apellido	varchar(50)	utf8_epanic_ci		No	Ninguna		Cambiar Eliminar Primaria Único Índice Más

Below the table structure, there are options to 'Agregar' (Add) records, with a dropdown menu set to 'idPersona' and a 'Continuar' button. The 'Información' section shows the following statistics:

Espacio utilizado		Estadísticas de la fila	
Datos	16 KB	Formato	Compact
Índice	0 B	Cotejamiento	utf8_spanish_ci
Total	16 KB	Índice automático siguiente	1
		Creación	12-12-2013 a las 14:11:40

# Índice: Tema 4.2

## **4.2 Conexión con Bases de Datos**

4.2.1 Envío y recepción de datos mediante formularios

4.2.2 Administración de MySQL con phpMyAdmin

**4.2.3 Conexión con Bases de Datos desde PHP**



#### ➤ PHP y MySQL (1)

- ✓ PHP dispone de numerosas funciones para interactuar con las bases de datos de MySQL. Algunas de ellas son las siguientes:

- `identificador = mysql_connect(servidor, usuario, contraseña)`

Establece la conexión con el servidor de bases de datos indicado utilizando un usuario y contraseña registrado en MySQL.

Cuando el servidor MySQL y el servidor Web se encuentran en el mismo equipo el servidor es `localhost`. Este es el valor por defecto si no se indica nada.

La función devuelve un valor entero que se utiliza como el identificador de la conexión en posteriores operaciones. Si se produce un error devuelve 0.

- `mysql_close(identificador)`

La conexión se cierra automáticamente al terminar la ejecución del script en el que se ha establecido, o bien explícitamente con esta función.



#### ➤ PHP y MySQL (2)

- `identificador = mysql_pconnect(servidor, usuario, contraseña)`

Establece una conexión persistente, es decir, que no será cerrada al terminar la ejecución del script en el que se ha establecido.

De esta forma la conexión permanece abierta y puede ser utilizada en los scripts de otras páginas que se ejecuten posteriormente.

- `mysql_pclose(identificador)`

Se utiliza para cerrar de forma explícita una conexión persistente.

- `mysql_error()`

Devuelve un texto con el mensaje de error de la última operación MySQL.

- `die(mensaje)`

Muestra el mensaje indicado y aborta la ejecución del programa.



#### ➤ PHP y MySQL (3)

- `mysql_selectdb(nombre_base_datos, identificador)`

Se utiliza para seleccionar la base de datos sobre la que llevar a cabo consultas.

Además se pasa un segundo parámetro para identificar la conexión al servidor de bases de datos en el que se encuentra la base de datos a la que se desea conectar. Dicho parámetro es opcional y, si no se especifica, su valor es el de la última conexión abierta.

- `recurso = mysql_query(consulta, identificador)`

Ejecuta la consulta especificada como una sentencia SQL sobre la base de datos activa en la conexión al servidor de bases de datos especificado con el identificador (opcional).

Ejemplos de posibles consultas:

```
SELECT * FROM tabla
```

```
SELECT columna FROM tabla WHERE condicion
```

```
INSERT INTO tabla (columna1, columna2) VALUES (valor1, valor2)
```

```
UPDATE tabla SET columna2=valor3 WHERE columna1=valor1
```

```
DELETE FROM tabla WHERE columna1=valor1
```



#### ➤ PHP y MySQL (4)

En el caso de consultas tipo SELECT, la función `mysql_query()` devuelve un conjunto de registros en caso de éxito y FALSE en caso contrario.

Este conjunto de registros se almacena en una estructura de datos conocida como recurso, a partir de la cual se pueden extraer los registros correspondientes mediante las funciones siguientes:

- `registro = mysql_fetch_row(recurso)`

Devuelve un array numérico que corresponde al registro recuperado y mueve el puntero de datos interno hacia delante. Así, la primera columna del registro se encuentra en la posición 0 de dicho array, la segunda en la 1, etc...

- `registro = mysql_fetch_assoc(recurso)`

Devuelve un array asociativo que corresponde al registro recuperado y mueve el puntero de datos interno hacia adelante. En este caso el nombre de las columnas se puede utilizar como clave de dicho array para acceder a su valor.

#### ➤ Ejemplo de utilización

- ✓ Se va a diseñar una aplicación web desde la que poder insertar, consultar, modificar y eliminar registros en la tabla “**tabla\_ejemplo**” de la base de datos “**basedatos\_ejemplo**” creada en el apartado anterior con phpMyAdmin:

DNI	Nombre	Apellido		
73845574	Alberto	Herrán	<a href="#">Modificar</a>	<a href="#">Eliminar</a>
85156184	Francisco	Soltero	<a href="#">Modificar</a>	<a href="#">Eliminar</a>
95156184	Fulano	De Tal	<a href="#">Modificar</a>	<a href="#">Eliminar</a>

Pulse el botón para añadir registros [Insertar](#)



#### ➤ Conexión a la base de datos

- ✓ Es una práctica común crear un fichero “configuración.php” que contenga los datos de configuración como el servidor, usuario y contraseña.
- ✓ Además, se pueden colocar aquí también el nombre de la base de datos, de la tabla o tablas a utilizar, y el identificador de la conexión (abierta de forma persistente) para ser utilizado en el resto de las páginas de la aplicación:

```
<?php
// Datos de configuración
$servidorBD = "localhost";
$usuarioBD  = "aherrang";           // Si no se ha creado: "root"
$passwordBD = "5748fT8Ss9GCFExJ"; // Si no se ha creado: ""
// Datos para ser usados en el resto de las páginas
$identificadorBD = mysql_pconnect($servidorBD,$usuarioBD,$passwordBD)
                    or die("Error: " . mysql_error());
$nombreBD = "basedatos_ejemplo";
$tablaBD  = "tabla_ejemplo";
?>
```



#### ➤ Consulta de tablas (1)

- ✓ La página “**ejercicio15.php**” muestra una tabla con los registros de la tabla y un enlace para insertar nuevos registros. Además, al lado de cada registro se han añadido enlaces para modificar y eliminar el registro correspondiente:

```
<?php
    require_once("configuracion.php");
    mysql_select_db($nombreBD,$identificadorBD);
    $consultaBD = "SELECT * FROM $tablaBD";
    $resultadoBD = mysql_query($consultaBD,$identificadorBD) or die("Error: ".mysql_error());
?>
// Ahora, dentro de la tabla que ubiquemos en el cuerpo de la página...
<?php while($registroBD = mysql_fetch_assoc($resultadoBD)) { ?>
    <tr align="right">
        <td> <?php echo $registroBD['dni']; ?>      </td>
        <td> <?php echo $registroBD['nombre']; ?>   </td>
        <td> <?php echo $registroBD['apellido']; ?> </td>
        <td> <a href="modificar.php?idPersona=<?php echo $registroBD['idPersona']; ?>">Modificar</a>
            <a href="eliminar.php?idPersona=<?php echo $registroBD['idPersona']; ?>">Eliminar</a>
        </td>
    </tr>
<?php }; ?>
<p>Pulse el botón para añadir registros <a href="insertar.php">Insertar</a></p>
```



#### ➤ Consulta de tablas (2)

- ✓ Originalmente la tabla se encuentra vacía mostrándose únicamente las cabeceras de las columnas y el botón que nos lleva a la página “insertar.php”:

The image displays two browser windows showing the 'Acceso a Bases de Datos' page. The left window shows the page with an empty table header and an 'Insertar' button. The right window shows the page with a table containing three rows of data and 'Modificar' and 'Eliminar' buttons for each row.

DNI	Nombre	Apellido		
73845574	Alberto	Herrán	<a href="#">Modificar</a>	<a href="#">Eliminar</a>
85156184	Francisco	Soltero	<a href="#">Modificar</a>	<a href="#">Eliminar</a>
95156184	Fulano	De Tal	<a href="#">Modificar</a>	<a href="#">Eliminar</a>



#### ➤ Inserción de registros (1)

- ✓ La página “**insertar.php**” inserta nuevos registros en la tabla. Para ello, ofrece un formulario desde el que introducir los datos y conecta con la base de datos para añadir un nuevo registro con los mismos:

```
<?php
    if(isset($_POST['insertar'])) {
        require_once("configuracion.php");
        mysql_select_db($nombreBD,$identificadorBD);
        $consultaBD = "INSERT INTO $tablaBD (dni,nombre,apellido) "
            . "VALUES ('".$_POST['dni']."' , '".$_POST['nombre']."' , '".$_POST['apellido']."' )";
        mysql_query($consultaBD,$identificadorBD) or die("Error:" .mysql_error());
        header("Location: ejercicio15.php");
    }
?>

// Ahora, dentro del cuerpo de la página...

<form action="<?php echo($_SERVER["PHP_SELF"])?>" method="post">
    DNI:      <input type="text"   name="dni"   />
    Nombre:   <input type="text"   name="nombre"  maxlength="50" />
    Apellido: <input type="text"   name="apellido" maxlength="50" />
    <input type="submit" name="insertar" value="Insertar">
</form>
```

#### ➤ Inserción de registros (2)

- ✓ Al pulsar insertar, los datos son enviados mediante POST a la propia página para insertarlos en la base de datos. Tras insertar los datos se vuelve a la página “ejercicio15.php” que lista la tabla con el nuevo registro:

The screenshot shows two browser windows. The left window, titled 'Inserción', displays a form with the following fields: DNI (73845574), Nombre (Alberto), and Apellido (Herrán). Below the fields is an 'Insertar' button. A red arrow points from this button to the right window. The right window, titled 'Bases de Datos', shows the result of the insertion. It features a table with the following data:

DNI	Nombre	Apellido		
73845574	Alberto	Herrán	<a href="#">Modificar</a>	<a href="#">Eliminar</a>

Below the table, there is a prompt: 'Pulse el botón para añadir registros' followed by an [Insertar](#) link.



#### ➤ Modificación de registros (1)

- ✓ La página “**modificar.php**” carga los datos del registro a modificar en un formulario para poder modificarlos y reenviárselos a si misma. Una vez recibidos se utilizan para actualizar el registro correspondiente en la base de datos:

```
// Desde el enlace Modificar de la página ejercicio15.php
<a href="modificar.php?idPersona=<?php echo $registroBD['idPersona']; ?>">Modificar</a>

// Ahora, en la página modificar.php
<?php
    require_once("configuracion.php");
    mysql_select_db($nombreBD,$identificadorBD);

    if(!isset($_POST["modificar"])) { // Consulta para cargar los datos en un formulario
        $consultaBD = "SELECT * FROM $tablaBD WHERE idPersona='{$_GET['idPersona']}'";
        $resultadoBD = mysql_query($consultaBD,$identificadorBD) or die("Error: ".mysql_error());
        $registroBD = mysql_fetch_assoc($resultadoBD);
    } else { // Tras pulsar modificar se actualizan los datos con los valores del formulario
        $consultaBD = "UPDATE $tablaBD SET dni='{$_POST['dni']}', nombre='{$_POST['nombre']}', "
            . "apellido='{$_POST['apellido']}' WHERE idPersona='{$_GET['idPersona']}'";
        mysql_query($consultaBD,$identificadorBD) or die("Error:" .mysql_error());
        header("Location: ejercicio15.php");
    }
?>
```



#### ➤ Modificación de registros (2)

- ✓ A continuación se muestra el formulario desde el que actualizar los datos en la página “**modificar.php**”:

```
<form action="modificar.php?idPersona=<?php echo $_GET['idPersona']; ?>" method="post">
  DNI:
  <input type="text" name="dni" value="<?php echo $registroBD['dni']; ?>"/>
  Nombre:
  <input type="text" name="nombre" value="<?php echo $registroBD['nombre']; ?>"/>
  Apellido:
  <input type="text" name="apellido" value="<?php echo $registroBD['apellido']; ?>"/>
  <input type="submit" name="modificar" value="Modificar">
</form>
```

#### ➤ Modificación de registros (3)

- ✓ Al pulsar modificar, los datos son enviados mediante POST a la propia página para actualizarlos en la base de datos. Tras insertar los datos se vuelve a la página “ejercicio15.php” que lista la tabla con el nuevo registro modificado:

The image shows two browser windows. The left window, titled 'Modificación', displays a form with the following fields: DNI (73845574), Nombre (Alberto), and Apellido (González). A 'Modificar' button is visible, with a red arrow pointing from it to the right window. The right window, titled 'Bases de Datos', shows the result of the modification. It features a table with the following data:

DNI	Nombre	Apellido		
73845574	Alberto	González	<a href="#">Modificar</a>	<a href="#">Eliminar</a>

Below the table, there is a prompt: 'Pulse el botón para añadir registros' followed by an [Insertar](#) button.



#### ➤ Borrado de registros (1)

- ✓ La página “**eliminar.php**” elimina el registro cuyo dni es el recibido mediante GET de la base de datos:

```
// Desde el enlace Eliminar de la página ejercicio15.php
<a href="eliminar.php?idPersona=<?php echo $registroBD['idersona']; ?>">Eliminar</a>

// Ahora, en la página eliminar.php
<?php
    require_once("configuracion.php");
    mysql_select_db($nombreBD,$identificadorBD);
    $consultaBD = "DELETE FROM $tablaBD WHERE idPersona='{$_GET['idPersona']}'";
    mysql_query($consultaBD,$identificadorBD) or die("Error: ".mysql_error());
    header("Location: ejercicio15.php");
?>
```

#### ➤ Borrado de registros (2)

- ✓ Al pulsar eliminar, el dni del registro a eliminar es enviado mediante GET a la página “eliminar.php” que lo elimina de la base de datos y devuelve el control a la página “ejercicio15.php” que lista la tabla sin el nuevo registro:

The image displays two browser windows side-by-side, illustrating the deletion of a record from a database. Both windows show a page titled "Acceso a Bases de Datos" with a table of records.

**Left Screenshot:** The browser address bar shows `http://localhost/GIE/AW/Tema4/Ejercicio20/ejercicio20.php`. The table contains one record:

DNI	Nombre	Apellido	
73845574	Alberto	González	<a href="#">Modificar</a> <a href="#">Eliminar</a>

Below the table, there is a text prompt: "Pulse el botón para añadir registros" followed by an [Insertar](#) button. A mouse cursor is clicking on the [Eliminar](#) button, with a red arrow pointing to the right. The status bar at the bottom shows the URL: `http://localhost/GIE/AW/Tema4/Ejercicio20/eliminar.php?dni=73845574`.

**Right Screenshot:** The browser address bar shows the same URL. The table is now empty:

DNI	Nombre	Apellido	
-----	--------	----------	--

Below the table, there is a text prompt: "Pulse el botón para añadir registros" followed by an [Insertar](#) button. The status bar at the bottom shows the zoom level: 100%.