TEMA 6. UTILIZACIÓN DE TÉCNICAS DE ACCESO A DATOS.

Objetivos

- Aprender a realizar una conexión con la base de datos.
- Estudiar la sintaxis para ejecutar sentencias SQL.
- Aprender a recuperar y utilizar información de la base de datos.
- Estudiar las posibles transacciones que se pueden realizar en una base de datos.
- Aprender los distintos niveles de aislamiento de una transacción.
- Aprender a obtener información de otros orígenes distintos, CSV, texto, XML.

6.1.- Establecimiento de conexiones.

A la hora de interactuar con la base de datos lo primero que debemos hacer es establecer una conexión desde la aplicación web.

```
$conexión = mysqli_connect($host, $user, $pass, $basedatos);
```

o

```
$conexión = mysqli_connect($host, $user, $pass);
mysql_select_db($basedatos);
```

Ejemplo:

```
$host = "localhost";
$user = "root";
$pass = "";
$basedatos = "videoclub";
$conexion = mysqli_connect($host, $user, $pass, $basedatos);
// UTF-8 conjunto de caracteres por defecto para conexión MySQL
mysqli_query ($conexion, "SET NAMES 'utf8'");
```

6.2.- Ejecución de sentencias SQL.

Una vez conectados a la base de datos podemos ejecutar sentencias SQL.

```
mysqli_query($conector, $sentencia);
```

Podemos asignar la ejecución de la sentencia SQL a una variable.

```
resultado = mysqli_query($conector, $sentencia);
```

Ejemplo:

```
$sentencia = "SELECT * FROM peliculas";
$resultado = mysqli_query($conexion, $sentencia);
```

6.3.- Utilización del conjunto de resultados.

Una vez ejecutadas las sentencias de la base de datos necesitamos procesar y utilizar dicha información, para ello almacenamos en una variable el resultado de la ejecución de la sentencia SQL. Disponemos de una serie de funciones y propiedades para procesar los datos obtenidos de la base de datos.

Método	Descripción
mysqli_num_rows($result)	Devuelve el número de filas del resultado
mysql_num_fields($result)	Devuelve el número de campos del resultado
mysqli_affected_rows($conexion)	Devuelve el número de filas afectadas por la sentencia SQL
Mysqli_free_result($result)	Libera la memoria reservada para almacenar el resultado
Mysql_fetch_array($result)	Devuelve un array con las filas obtenidas o false.
Mysqli_fetch_row($result)	Devuelve un array de resultados como un array asociativo.
Mysqli_fetch_field($result, [$pos_campo])	Devuelve el siguiente campo del conjunto de resultados.
Mysqli_fetch_assoc($result)	Devuelve la fila de resultados como un array asociativo.
Mysqli_fetcj_object($result)	Devuelve la fila actual del resultado en forma de un objeto.
Mysqli_fetch_lengths($result, $num)	Devuelve la longitud de las columnas de la fila actual en el conjunto de resultados.

Ejemplo:

```
while($registro=mysqli_fetch_row($resultado)){
      foreach($registro as $valor){
            echo $valor." - ";
      }
      echo "<br/>";
}
```

Ejemplo:

```
// Almacenamos el resultado de la consulta en un array
while($fila = mysqli_fetch_assoc($resultado)){
      $registros[] = $fila;
}
mysqli_free_result($resultado);
```

6.4.- Cierre de las conexiones.

Una vez hemos terminado de trabajar con la base de datos es necesario liberar la memoria reservada por el resultado de la ejecución de la sentencia SQL (sólo en la sentencia SELECT) y cerrar la base de datos pasándole como parámetro el conector que devolvió la conexión a la base de datos.

```
mysqli_free_result($resultado);
mysqli_close($conexion);
```

Ejemplo:

```
mysqli_free_result($resultado);
mysqli_close($conexion);
```

Ejemplo completo:

```
<style>
      .tabla{
            border: 1px solid black;
            border-collapse: collapse;
      }
      .cabecera{
            border: 1px solid black;
            background-color: red;
            width: 100px;
            font-weight: bold;
            color: white;
            text-align: center;
      }
      .campo{
            border: 1px solid black;
      }
</style>

<?php
    // Datos conexión a la base de datos
      $host = "localhost";
      $user = "root";
      $pass = "";
      $basedatos = "videoclub";
      $conexion = mysqli_connect($host, $user, $pass, $basedatos);
```

```
	// UTF-8 conjunto de caracteres por defecto para conexión MySQL
	mysqli_query ($conexion, "SET NAMES 'utf8'");

	// Sentencia SQL y almacenar en $resultado su ejecución
	$sentencia = "SELECT * FROM peliculas";
	$resultado = mysqli_query($conexion, $sentencia);

	// Muesta el número de registros del resultado de la consulta SQL
	echo "Registros: ".mysqli_num_rows($resultado);

	// Muestra resultado en forma de tabla
	echo "<table class='tabla'>";
	echo "<tr>";
	// Muestra cabecera de tabla
	$cabecera = array("Codigo","Titulo","Tema","Duracion","Precio");
	foreach($cabecera as $dato){
		echo "<td class='cabecera'>".$dato."</td>";
	}
	echo "</tr>";

	// Obtiene un registro del resultado de la consulta
	// Si no hay más registros sale del bucle while
	while($registro=mysqli_fetch_row($resultado)){
		echo "<tr>";
		// Muestra cada uno de los valores de los campos del resgistro
		foreach($registro as $valor){
			echo "<td class='campo'>".$valor."</td>";
		}
		echo "</tr>";
	}
	echo "</table>";
	mysqli_free_result($resultado);
	mysqli_close($conexion);
?>
```

Ejemplo utilizando un array para almacenar el resultado de la consulta:

```
<style>
	.tabla{
		border: 1px solid black;
		border-collapse: collapse;
	}
	.cabecera{
		border: 1px solid black;
		background-color: red;
		width: 100px;
		font-weight: bold;
		color: white;
		text-align: center;
	}
	.campo{
		border: 1px solid black;
	}
</style>
```

```php
<?php
    // Datos conexión a la base de datos
       $host = "localhost";
       $user = "root";
       $pass = "";
       $basedatos = "videoclub";
       $conexion = mysqli_connect($host, $user, $pass, $basedatos);

       // UTF-8 conjunto de caracteres por defecto para conexión MySQL
       mysqli_query ($conexion, "SET NAMES 'utf8'");

       // Sentencia SQL y almacenar en $resultado su ejecución
       $sentencia = "SELECT * FROM peliculas";
       $resultado = mysqli_query($conexion, $sentencia);

       // Muesta el número de registros del resultado de la consulta SQL
       echo "Registros: ".mysqli_num_rows($resultado);

       // Almacenamos el resultado de la consulta en un array
       while($fila = mysqli_fetch_assoc($resultado)){
              $registros[] = $fila;
       }
       mysqli_free_result($resultado);

       // Muestra resultado en forma de tabla
       echo "<table class='tabla'>";
       echo "<tr>";
       // Muestra cabecera de tabla
       $cabecera = array("Codigo","Titulo","Tema","Duracion","Precio");
       foreach($cabecera as $dato){
              echo "<td class='cabecera'>".$dato."</td>";
       }
       echo "</tr>";

       // Obtiene un registro del array de resultados de la consulta
       foreach($registros as $registro){
              echo "<tr>";
              // Muestra cada uno de los valores de los campos del resgistro
              foreach($registro as $valor){
                     echo "<td class='campo'>".$valor."</td>";
              }
              echo "</tr>";
       }
       echo "</table>";
       mysqli_close($conexion);
?>
```

6.5 Sentencias del Lenguaje MySQL

Crear Base de Datos:

```
CREATE DATABASE basededatos;
```

Eliminar Base de Datos:

```
DROP DATABASE basededatos;
```

Mostrar Bases de Datos:

```
SHOW DATABASES;
```

Usar Base de Datos:

```
USE basededatos;
```

Crear Tabla:

```
CREATE TABLE nombretabla (
      campo1 tipo1,
      campo2 tipo2,
      :     :     :
      primary key (campo)
);

REQUERIDO:  NOT NULL
TAMAÑO:     (n)
VALOR POR DEFECTO:      DEFAULT valor
```

Muestra las tablas de la base de datos.

```
SHOW TABLES;
```

Muestra la estructuta de la tabla.

```
DESCRIBE nombretabla;
```

Muestra la estructura de la tabla.

```
SHOW CREATE TABLE nombretabla;
```

Añadir un campo a una tabla.

```
ALTER TABLE nombretabla ADD campo tipo AFTER|FIRST campo;
```

Modificar el nombre del campo de una tabla.

```
ALTER TABLE nombretabla CHANGE campo camponuevo tiponuevo;
```

Eliminar un campo de una tabla.

```
ALTER TABLE nombretabla DROP campo;
```

Modificar el tipo de campo de una tabla.

```
ALTER TABLE nombretabla MODIFY campo tipo;
```

Añadir una relación clave primaria-foranea a una tabla.

```
ALTER TABLE nombretabla_N ADD CONSTRAINT nombreconstraint
FOREIGN KEY (campotabla_N) REFERENCES tabla_1(campo_1)
ON DELETE CASCADE;
```

Mostrar las CONSTRAINTS de una tabla

```
SELECT * FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS WHERE
TABLE_NAME='nombretabla';
```

Borrar una relación clave primaria-foranea de una tabla.

```
ALTER TABLE nombretabla DROP FOREIGN KEY nombreconstraint;
```

Insertar un registro completo en una tabla.

```
INSERT INTO table VALUES (valor1, valor2, …);
```

Insertar algunos campos de un registro en una tabla.

```
INSERT INTO table (campo1, campo2,..) VALUES (valor1, valor2, …);
```

Duplicar la estructura de una tabla.

```
CREATE TABLE tablacopia LIKE tablaorigen;
```

Duplicar estructura y datos de una tabla.

```
CREATE TABLE tablacopia SELECT * FROM tablaorigen;
```

Renombrar una tabla.

```
RENAME TABLE nombreorigen TO nombredestino;
```

Eliminar una tabla.

```
DROP TABLE nombretabla;
```

Consultas.

```
SELECT nombrecampo1, nombrecampo2,..
FROM nombretabla
WHERE condicion
ORDER BY nombrecampo [ASC|DESC];

Nota: Podemos usar * para indicar que muestre todos los campos.
```

Consultas con tablas relacionadas.

```
SELECT nombretabla1.nombrecampo1, nombretabla2.nombrecampo2,..
FROM nombretabla1, nombretabla2
WHERE nombretabla1.campoclave1 = nombretabla2.campoclave2;
```

Consultas campos calculados.

```
SELECT nombrecampo, operación AS nombrecampocalculado,..
FROM nombretabla
WHERE condicion
ORDER BY nombrecampo [ASC|DESC];
```

Consulta de Creación de Tabla.

```
CREATE TABLE nombretabla as (consulta);
```

Consulta Grupos-Totales.

```
SELECT nombrecampogrupo, operacion(nombrecampooperacion), …
FROM nombretabla
WHERE condicion
GROUP BY nombrecampogrupo
HAVING condicioncampogrupo
ORDER BY nombrecampo;
```

Nota: La operación puede ser SUM, COUNT, MAX, MIN o AVG.

Actualización de datos.

```
UPDATE nombretabla
SET nombrecampo1 = valor1, nombrecampo2 = valor2, …
WHERE condicion;
```

Eliminar datos.

```
DELETE FROM nombretabla
WHERE condicion;
```

Eliminar todos los datos de una tabla.

```
TRUNCATE nombretabla;
```

Nota: SET FOREIGN_KEY_CHECKS = 1; para ignorer relaciones de tablas.

Tipos de Campos.

Tipo Texto (Char(x), Varchar(x), Text, TinyText, MediumText, LongText)

Char(x)	Tipo de datos que admite caracteres alfanuméricos. La longitud de este campo varía entre 1-255 y está delimitado a la longitud especificada entre paréntesis **(x)** en el momento de la creación del campo de la tabla. Al introducir datos en este campo siempre se solicitará el número de caracteres especificados. Si creamos un campo con **Char(5)** deberemos introducir cinco caracteres cada vez que incluyamos un dato en ese campo. Si incluimos menos, **mySQL** rellenará los caracteres que faltan hasta el número indicado con espacios.
Varchar(x)	Tipo de datos que admite caracteres alfanuméricos. Su uso es similar a **Char(x)**. A la hora de definir un campo de datos Varchar deberemos especificar el número máximo de caracteres que podrá aceptar en la entrada de datos, donde x es un número entre 1-255. A diferencia de **Char**, este tipo de datos es variable en su longitud, admitiendo entradas inferiores a la establecida.
Text, TinyText, MediumText, LongText	Mediante la declaración de este tipo de datos se admiten la inclusión de cadenas alfanuméricas "case-insensitive" de longitudes variables. **TinyText** admite un máximo de 255 caractéres, **Text** admite 65.535, **MediumText** permite introducir textos de hasta 16.777.215 caracteres, **LongText** nos ofrece la posibilidad de incluir un máximo de 4.294.967.295 caracteres. Estos campos no necesitan de especificaciones de longitud a la hora de ser declarados.

Enum: campo que puede tener un único valor de una lista que se especifica. El tipo Enum acepta hasta 65535 valores distintos.

```
tema ENUM('Aventuras', 'Drama', 'Comedia', 'Intriga')
```

Tipo numérico (TinyInt, SmallInt, MediumInt, Int, BigInt, Float, Double, Decimal)

Int	Este es un tipo de datos numéricos de tipo entero. Este tipo de datos guarda valores enteros (no decimales) entre -2.147.483.648 y 2.147.483.647.
TinyInt, SmallInt, MediumInt, BigInt	Son tipos de datos numéricos enteros (no decimal). **TinyInt** agrupa un rango de números entre -128 y 127. **SmallInt** alcanza desde -32.768 hasta 32.767. **MediumInt** tiene un rango comprendido entre -8.388.608 y 8.388.607. Finalmente el tipo de datos **BigInt** ocupa un rango numérico entre -9.223.372.036.854.775.808 hasta 9.223.372.036.854.775.807.
Float (M,D)	Número de coma flotante de precisión simple. El valor del argumento M nos indica el número de digitos decimales que se van a utilizar para representar el número. Así, un valor de 5 nos permitirá representar números comprendidos entre -99 y 99 (Numeros expresados en binario con 5 dígitos y signo). El valor del argumento D nos indica el número de posiciones decimales que se van a utilizar en la representación del número. Así, una representación tipo **Float (5,2)** nos permitirá incluir números entre -99,99 y 99,99. El rango de los números de coma flotante de precisión simple es de -3,402823466E+38 a -1,175494351E-38, 0, y 1,175494351E-38 hasta 3,402823466E+38.
Double (M,D)	Número de coma flotante de precisión doble. Es un tipo de datos igual al anterior cuya única diferencia es el rango numérico que abraca, siendo este el comprendido entre 1,7976931348623157E+308 hasta -2,.2250738585072014E-308, 0, y 2,2250738585072014E-308 to 1,7976931348623157E+308
Decimal (M,D)	Su uso es similar al de los anteriores, pero, en este caso, D puede tener valor 0. El rango de este número es el mismo que el de número con coma flotante de precisión doble.

Bit ó Bool: un número entero que puede ser 0 ó 1.

Tipo Fecha-Hora (Date, DateTime, TimeStamp, Time, Year)

Date	Formato de Fecha. Su representación es en formato de fecha numérica del tipo '**YYYY-MM-DD**' (Año con cuatro dígitos, Mes con dos dígitos, día con dos dígitos). Su rango es '1000-01-01' (1 de enero del año 1000, en el cual yo era aún muy pequeño) hasta '9999-12-31' (31 de diciembre del 9999, que ya veremos que pasa después de las uvas)
DateTime	Es una combinación de formato de fecha y hora conjuntamente. Su representación es '**YYYY-MM-DD HH:MM:SS**' (Año con cuatro dígitos, Mes con dos dígitos, día con dos dígitos, hora con dos dígitos, minutos con dos dígitos, segundos con dos dígitos). El rango que soporta este formato es de '1000-01-01 00:00:00' (las 00 horas, 00 minutos, 00 segundos del 1 de enero del año 1000, que no se yo con que reloj podían medir esto) hasta '9999-12-31 23:59:59' (las 23 horas, 59 minutos, 59 segundos del 31 de diciembre del año 9999, es decir, justo antes de las campanadas y una vez que han acabado los cuartos).

TimeStamp(N)	Este es un tipo de datos muy particular. Necesita de un argumento N que puede ser uno de estos números: 14, 12, 10, 8, 6, 4, 2. N representa el número de dígitos que se utilizarán para repsentar un valor de fecha y hora comprendido desde el inicio del año 1970 hasta algún momento del año 2037. Así: TimeStamp(14): YYYYMMDDHHMMSS (Año 4 dígitos + mes + día + hora + minutos + segundos 2 dígitos) TimeStamp(12): YYMMDDHHMMSS (Año 2 dígitos + mes + día + hora + minutos + segundos 2 dígitos) TimeStamp(10): YYMMDDHHMM (Año + mes + día + hora + minutos 2 dígitos) TimeStamp(8): YYMMDDHH (Año + mes + día + hora 2 dígitos) TimeStamp(6): YYMMDD (Año + mes + día 2 dígitos) TimeStamp(4): YYMM (Año + mes 2 dígitos) TimeStamp(2): YY (Año 2 dígitos)
Time	Tipo de datos con formato de Hora. mySQL muestra valores de hora con formato '**HH:MM:SS**'
Year(D)	Tipo de datos con formato de año. Su representación puede ser '**YYYY**' (año con formato de 4 dígitos) o '**YY**' (año con formato de 2 dígitos) donde el valor del argumento D puede ser 4 o 2 respectivamente.

Operadores.

Operador	**Significado**
<	Menor
>	Mayor
<=	Menor o Igual
>=	Mayor o igual
=	Igual
<> , !=	Distinto
BETWEEN a AND b	Entre dos valores a y b
IS NULL	No contiene dato
IS NOT NULL	Contiene Dato
IN(valor1, valor2,….)	Es igual a alguno de los valores
LIKE %	Cero o Varios caracteres LIKE A% Comienza por A LIKE %N Termina en N LIKE %S% Contiene S NOT LIKE A% No comienza por A NOT LIKE %N No termina en N NOT LIKE %S% No contiene S
LIKE _	Un carácter Mari_ (Mario, Maria, Marín,…)
AND	TRUE si todas las condiciones se cumplen.
OR	TRUE si alguna de las condiciones se cumplen.
NOT, !	NEGACIÓN

Metacaracteres en Expresiones Regulares

Carácter	Descripción
*	Cero o más coincidencias de de la cadena a la que preceden.
+	Una o más coincidencias de de la cadena a la que preceden.
?	Cero o una coincidencias de de la cadena a la que preceden.
.	Cualquier caracter individual
[abc]	Coincida con cualquiera de los caracteres incluidos.
[^abc]	No coincida con cualquiera de los caracteres incluidos.
[A-Z]	Coincida con cualquier letra mayúscula.
[a-z]	Coincida con cualquier letra minúscula.
[0-9]	Coincida con cualquier dígito del 0 al 9.
[^A-Z]	No coincida con cualquier letra mayúscula.
[^a-z]	No coincida con cualquier letra minúscula.
[^0-9]	No coincida con cualquier dígito del 0 al 9.
^	Se utiliza para indicar el comienzo.
$	Se utiliza para indicar el final.
|	Se utiliza para indicar diferentes alternativas.
[[:<:]]	Coincide con el comienzo de las palabras.
[[:>:]]	Coincide con el final de las palabras
BINARY	Hacer una busque case sesitive (diferencia mayúsculas de minúsculas).
(|)	Permite crear grupos de coincidencias.
{}	Indica la cantidad de repeticiones {n} n repeticiones {n,} n o más repeticiones {n,m} entre n y m repeticiones * puede escribirse como {0,} + puede escribirse como {1,} ? puede escribirse como {0,1}

La barra invertida (\) se utiliza como un carácter de escape. Si queremos usarlo como parte del patrón de una expresión regular, debemos utilizar dobles barras invertidas (\\).

Ejemplo Mantenimiento tabla: Altas, bajas, modificaciones, consultas y listado.

conexion.php

```
<?php
	$host = "localhost";
	$user = "root";
	$pass = "";
	$basedatos = "videoclub";
	$conexion = mysqli_connect($host, $user, $pass, $basedatos)
		or die("Error al conectar a la base de datos");
	// UTF-8 conjunto de caracteres por defecto para conexión MySQL
	mysqli_query ($conexion, "SET NAMES 'utf8'");
?>
```

desconexion.php

```
<?php
	mysqli_close($conexion);
?>
```

insertar.php

```
<form action="" name="formulario" method="post">
	Codigo Socio <input type="text" name="codsocio"><br>
	Nombre <input type="text" name="nombre"><br>
	Apellidos <input type="text" name="apellidos"><br>
	Direccion <input type="text" name="direccion"><br>
	Telefono <input type="text" name="telefono"><br>
	Poblacion <input type="text" name="poblacion"><br>
	<input type="submit" value="Insertar" name="boton">
</form>
<?php
if (isset($_POST['boton'])){
	include("conexion.php");
	$codsocio = $_POST['codsocio'];
	$nombre = $_POST['nombre'];
	$apellidos = $_POST['apellidos'];
	$direccion = $_POST['direccion'];
	$telefono = $_POST['telefono'];
	$poblacion = $_POST['poblacion'];
	$sentencia = "insert into socios values($codsocio,'$nombre',
		'$apellidos','$direccion','$telefono','$poblacion')";
	$consulta = mysqli_query($conexion, $sentencia)
		or die("Error de Consulta");
	if (mysqli_affected_rows($conexion)!=0) {
		echo "El registro ha sido insertado con exito<br>";
	}
	include("desconexion.php");
}
?>
```

consulta.php

```
<form action="" name="formulario" method="post">
	Codigo Socio <input type="text" name="codsocio"><br>
	<input type="submit" value="Mostrar">
</form>
<?php
	if (isset($_POST["codsocio"])){
		include("conexion.php");
		$codsocio = $_POST['codsocio'];
		$sentencia = "select * from socios where CODSOCIO=$codsocio";
		$consulta = mysqli_query($conexion, $sentencia)
			or die("Error de Consulta");
		if ($fila = mysqli_fetch_array($consulta)){
			$codsocio = $fila['codsocio'];
			$nombre = $fila['nombre'];
			$apellidos = $fila['apellidos'];
			$direccion = $fila['direccion'];
			$telefono = $fila['telefono'];
			$poblacion = $fila['poblacion'];
			echo "Codigo: $codsocio<br>";
			echo "Nombre: $nombre<br>";
			echo "Apellidos: $apellidos<br>";
			echo "Direccion: $direccion<br>";
			echo "Telefono: $telefono<br>";
			echo "Poblacion: $poblacion<br>";
			mysqli_free_result($consulta);
		}
		else
		{
			echo "Registro no encontrado.<br>";
		}
		include("desconexion.php");
	}
	else
	{
		echo "Introduzca codigo de socio que queremos mostrar.<br>";
	}
?>
```

actualizar.php

```
<form id="form1" method="post" action="">
    Codigo Socio: <input type="text" name="codsocio"><br>
    <input type="submit" name="boton" value="Consultar"><br>
</form>
<?php
if ((isset($_POST['boton'])) && ($_POST['boton'] == 'Consultar')){
    if (isset($_POST["codsocio"])){
        include("conexion.php");
        $codsocio = $_POST['codsocio'];
        $sentencia = "select * from socios where CODSOCIO=$codsocio";
        $consulta = mysqli_query($conexion, $sentencia)
            or die("Error de Consulta");
        if ($fila = mysqli_fetch_array($consulta)){
            $codsocio = $fila['codsocio'];
            $nombre = $fila['nombre'];
            $apellidos = $fila['apellidos'];
```

```
            $direccion = $fila['direccion'];
            $telefono = $fila['telefono'];
            $poblacion = $fila['poblacion'];
            echo "<form id='form2' action='' method='post'>";
            echo "Codigo Socio:
            <input type='text' value='$codsocio' name='codsocio'
readonly><br>";
            echo "Nombre:
            <input type='text' value='$nombre' name='nombre'><br>";
            echo "Apellidos:
            <input type='text' value='$apellidos' name='apellidos'><br>";
            echo "Direccion:
            <input type='text' value='$direccion' name='direccion'><br>";
            echo "Telefono:
            <input type='text' value='$telefono' name='telefono'><br>";
            echo "Poblacion:
            <input type='text' value='$poblacion' name='poblacion'><br>";
            echo "<input type='submit' name='boton' value='Actualizar'><br>";
            echo "</form>";
        }
        mysqli_free_result($consulta);
        include("desconexion.php");
    }
    else
    {
        echo "Registro no encontrado.<br>";
    }
}
if ((isset($_POST['boton'])) && ($_POST['boton'] == 'Actualizar')){
    $codsocio = $_POST['codsocio'];
    $nombre = $_POST['nombre'];
    $apellidos = $_POST['apellidos'];
    $direccion = $_POST['direccion'];
    $telefono = $_POST['telefono'];
    $poblacion = $_POST['poblacion'];

    include("conexion.php");
    $sentencia = "UPDATE socios SET nombre='$nombre',apellidos='$apellidos',
direccion='$direccion', telefono='$telefono', poblacion='$poblacion' where
codsocio=$codsocio";
    $consulta = mysqli_query($conexion, $sentencia)
        or die("Error de Consulta");
    echo "Registro ha sido actualizado con exito<br>";
    include("desconexion.php");
}
?>
```

eliminar.php

```
<form action="" name="formulario" method="post">
    Codigo Socio <input type="text" name="codsocio"><br>
    <input type="submit" value="Mostrar"name="boton">
</form>
<?php
if (isset($_POST["boton"])){
    if ($_POST["boton"]=="Mostrar"){
        include("conexion.php");
        $codsocio = $_POST['codsocio'];
        $sentencia = "select * from socios where CODSOCIO=$codsocio";
```

```
        $consulta = mysqli_query($conexion, $sentencia)
            or die("Error de Consulta");
        if ($fila = mysqli_fetch_array($consulta)){
            $codsocio = $fila['codsocio'];
            $nombre = $fila['nombre'];
            $apellidos = $fila['apellidos'];
            $direccion = $fila['direccion'];
            $telefono = $fila['telefono'];
            $poblacion = $fila['poblacion'];
            echo "Codigo: $codsocio<br>";
            echo "Nombre: $nombre<br>";
            echo "Apellidos: $apellidos<br>";
            echo "Direccion: $direccion<br>";
            echo "Telefono: $telefono<br>";
            echo "Poblacion: $poblacion<br>";
            echo "<br>";
            echo "<form action='' name='formulario2' method='post'>";
            echo "<input type='hidden' name='codsocio' value='$codsocio'>";
            echo "<input type='submit' value='Eliminar'name='boton'>";
            echo "</form>";
            include("desconexion.php");
        }
        else
        {
            echo "Registro no encontrado.<br>";
        }
    }

    if ((isset($_POST["boton"])) && ($_POST["boton"]=="Eliminar")){
        include("conexion.php");
        $codsocio = $_POST['codsocio'];
        $sentencia = "delete from socios where codsocio=$codsocio";
        $consulta = mysqli_query($conexion, $sentencia)
            or die("Error de Consulta");
        if (mysqli_affected_rows($conexion)!=0) {
            echo "El registro ha sido eliminado con exito<br>";
        }
        else
        {
            echo "El registro NO ha sido eliminado con exito<br>";
        }
        include("desconexion.php");
    }
}
?>
```

listar.php

```
<?php
    include("conexion.php");
    $sentencia = "select * from socios";
    $consulta = mysqli_query($conexion, $sentencia)
        or die("Error de Consulta");
    echo "El numero de registros es ".mysqli_num_rows($consulta)."<br>";
    while($fila = mysqli_fetch_array($consulta)){
        $codsocio = $fila['codsocio'];
        $nombre = $fila['nombre'];
        $apellidos = $fila['apellidos'];
        $direccion = $fila['direccion'];
```

```
        $telefono = $fila['telefono'];
        $poblacion = $fila['poblacion'];
        echo "$codsocio.- $nombre $apellidos - $direccion - $telefono -
$poblacion<br>";
    }
    mysqli_free_result($consulta);
    include("desconexion.php");
?>
```

6.6.- Otros orígenes de datos.

Operaciones con Ficheros.

Podemos abrir ficheros utilizando la sentencia:

```
fopen (nombre, modoApertura);
```

El parámetro modoApertura puede tomar los siguientes valores:

Valor	Significado
r	Modo de sólo lectura. El puntero se coloca al inicio del fichero.
r+	Modo de lectura y escritura. El puntero se coloca al inicio del fichero.
w	Modo de sólo escritura. Si no existe el fichero, se crea. Si ya existe, se borra todo el contenido.
w+	Modo de lectura y escritura. Si no existe el fichero, se crea. Si ya existe, se borra todo su contenido.
a	Modo de sólo escritura. Si no existe el fichero, se crea. Si ya existe, el puntero se coloca al final del fichero para añadir datos.
a+	Modo de lectura y escritura. Si no existe el fichero, se crea. Si ya existe, el puntero se coloca al final del fichero para añadir datos.

La función fopen () devuelve un descriptor de fichero, esto es, el canal por el que vamos a poder acceder a él a la hora de realizar operaciones sobre ficheros tales como lectura, escritura, cerrado, etc.

En el caso de que el fichero no se pudiera abrir por la causa que sea, fopen() devolverá FALSE.

Ejemplo:

```
$fichero = fopen("archivo.txt","w");
```

Podemos cerrar un fichero con la función fclose () cerramos el fichero que está referenciado por el descriptor que pasamos como argumento.

```
fclose (descriptor_fichero);
```

Ejemplo:

```
fclose($fichero);
```

Podemos leer en un fichero con las sentencias:

Sentencia	Descripción
fgetc (descriptor)	Devuelve un carácter del fichero referenciado por descriptor. Si se ha llegado al final del fichero, devuelve FALSE.
fread (descriptor, total_cars_a_leer)	Devuelve una cadena don el total de caracteres a leer.
feof (identificador)	No es una función de lectura propiamente dicha. pero es muy útil cuando leemos ficheros: indica si se ha llegado al final (no quedan más datos por leer) o no.

Podemos recorrer un fichero con las sentencias:

Sentencia	Descripción
rewind (descriptor)	Sitúa el puntero de lectura/escritura al principio del fichero.
fseek (descriptor, desplaz [,posicion])	Desplaza la posición del puntero de lectura / escritura desplaz posiciones. El tercer parámetro puede tomar los valores SEEK_SET, SEEK_CUR y SEEK_END, lo que significará que los desplazamientos son relativos al principio del fichero, la posición actual del puntero o al final del fichero (entonces desplaz será negativo).
ftell (descriptor)	Devuelve la posición del puntero.

Podemos escribir en un fichero con las sentencias:

fwrite(descriptor, cadena [, total_cars]);

fputs(descriptor, cadena [, total_cars]);

Ambas funciones escriben la cadena (completa) pasada como parámetro. Si se hace uso del tercer parámetro, sólo se escribirán los total_cars indicados. Devuelve el total de caracteres escritos o FALSE en caso de producirse algún error.

Ejemplo:

```
<?php
      //    Crea fichero e introduce la cadena.
      $fichero = fopen("archivo.txt","w");
      $cadena="Este es el texto que introduciremos en el fichero";
      fwrite($fichero, $cadena);
      fclose($fichero);
```

```
      // Abre el fichero y lee el contenido
      $fichero = fopen("archivo.txt","r");
      while($caracter=fgetc($fichero)){
            echo $caracter;
      }
      fclose($fichero);
?>
```

Información sobre ficheros.

Sentencia	Descripción
file_exists(nombre_fich)	Comprueba si el fichero nombre_fich existe. Devuelve TRUE si nombr_fich existe, en caso contrario, devuelve FALSE.
is_file (fichero)	Devuelve verdadero si el tipo del fichero es un fichero normal. En caso contrario, devuelve FALSE.
is_dir(fichero)	Devuelve verdadero si el nombre del fichero pasado como argumento es un directorio. En caso de que no exista o no lo sea, devuelve FALSE.
is_executable (fichero)	Devuelve verdadero si el nombre del fichero pasado como argumento tiene permisos de ejecución o es un fichero ejecutable (extensión exe, com o bat).
is_readable (fichero)	Devuelve verdadero si el fichero tiene permiso de lectura. Si no tiene dicho permiso o no existe, devuelve FALSE.
is_writeable (fichero)	Devuelve verdadero si el fichero tiene permiso de escritura. Si no tiene dicho permiso o no existe, devuelve FALSE.
filemtime (fichero)	Devuelve el instante ' timestamp' de la última modificación hecha sobre el contenido del fichero.
filesize (fichero)	Devuelve un número entero que indica el tamaño en bytes del fichero pasado como parámetro. En caso de error, devuelve FALSE.

Ejemplo:

```
<?php
      // fichero que comprobamos
      $fichero = "archivo.txt";

      // Zona horaria Madrid
      date_default_timezone_set('Europe/Madrid');

      if (file_exists($fichero)) {
            if (is_file ($fichero)) {
                  echo "Es un fichero<br>";
            } else {
                  echo "No es un fichero<br>";
            }

            if (is_dir ($fichero)) {
                  echo "Es un directorio<br>";
            } else {
                  echo "No es un directorio<br>";
            }
```

```
            if (is_executable ($fichero)) {
                  echo "Tiene permiso ejecutable<br>";
            } else {
                  echo "No tiene permiso ejecutable<br>";
            }

            if (is_readable ($fichero)) {
                  echo "Tiene permiso lectura<br>";
            } else {
                  echo "No tiene permiso lectura<br>";
            }

            if (is_writeable ($fichero)) {
                  echo "Tiene permiso escritura<br>";
            } else {
                  echo "No tiene permiso escritura<br>";
            }

            echo "Fecha: ".date("d/m/Y H:i:s", filemtime($fichero)).
                  "<br>";

            echo "Tamaño: ".filesize($fichero)."<br>";
      }
      else {
            echo "El fichero no existe";
      }
?>
```

Operaciones con Ficheros y Directorios.

Sentencia	Descripción
copy (origen, destino)	Copia un fichero destino. Si no ha habido ningún error, devuelve TRUE.
rename (nombre_original, nombre_final)	Cambia el nombre a un fichero o directorio.
unlink (nom_fichero)	Borra un fichero.
opendir (nombre)	Abre el directorio pasado como parámetro. Devuelve un descriptor de directorio.
readdir(descriptor)	Lee una entrada del directorio a partir del descriptor devuelto por opendir().
closedir(descriptor)	Cierra el directorio indicador por el descriptor.
rewinddir (descriptor)	Sitúa el puntero de lectura al principio del directorio.
getcwd ()	Devuelve el directorio actual de trabajo (también llamado activo).

chdir(nuevo_dir)	Establece nuevo_dir como nuevo directorio de trabajo. Devuelve FALSE en caso de error.
mkdir (nombre_dir, permisos)	Creamos el directorio nombre_dir con los permisos indicados en el segundo parámetro.
rmdir (nombre_dir)	Borra el directorio nombre_dir.

Ejemplo:

```
<?php
	// fichero origen y destino
	$fichero_origen = "archivo.txt";
	$fichero_destino = "copia_de_archivo.txt";
	// copia fichero
	copy ($fichero_origen, $fichero_destino);

	// renombra fichero
	rename ($fichero_destino, "archivo_datos.txt");

	// borra fichero
	unlink ("archivo_datos.txt");

	// Crea carpeta y muestra carpeta actual
	mkdir ("misdatos");
	echo "Carpeta Actual: ".getcwd()."<br>";

	// Accede a caropta y muestra carpeta actual
	chdir("misdatos");
	echo "Carpeta Actual: ".getcwd()."<br>";

	// Sale carpeta y muestra carpeta actual
	chdir("..");
	echo "Carpeta Actual: ".getcwd()."<br>";

	// Borra carpeta
	rmdir ("misdatos");

	// Abre el directorio actual para leer contenido
	$directorio = opendir (".");
	// Lee y muesra el contenido de la carpeta
	while($fichero=readdir($directorio)){
		echo $fichero."<br>";
	}
	echo "<br>";

	// Mueve puntero lectura al primer elemento y lo muestra
	rewinddir ($directorio);
	echo "Primer fichero: ".readdir($directorio)."<br>";

	// Cierra la lectura del directorio actual
	closedir($directorio);
?>
```

Acceso a ficheros XML.

Podemos acceder a ficheros XML utilizando la función

simplexml_load_file(fichero)

que interpreta un fichero XML como un array y podemos acceder a sus elementos de forma sencilla.

Ejemplo:

libros.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<libros>
	<libro>
		<codigo>1</codigo>
		<titulo>Internet para Empresas</titulo>
		<autor>Lorenzo Ros Medina</autor>
		<precio>45.20</precio>
	</libro>
	<libro>
		<codigo>2</codigo>
		<titulo>Windows 2016</titulo>
		<autor>Ana Romero Sanz</autor>
		<precio>19.50</precio>
	</libro>
</libros>
```

leerxml.php

```
<?php
	$xml=simplexml_load_file("libros.xml");
	foreach($xml as $dato){
		foreach($dato as $valor){
			echo $valor."<br>";
		}
		echo "<br>";
	}
?>
```

Acceso a ficheros CSV.

Podemos leer y escribir datos de un fichero CSV con las sentencias:

`fgetcsv($descriptor,[$tamaño], [$delimitador],[$cierre_campo], [$carácter_escape])`	Lee un fichero CSV. Devuelve en una matriz los valores de un campo.
`fputcsv($descriptor, $matriz_valores, [$delimitador],[$cierre_campo])`	Escribe en un fichero CSV. Devuelve la longitud de la cadena escrita.

Ejemlo:

```
<?php
	$fichero = fopen("contactos.csv","w");
	$alumnos = array(
			array("Sandra","sandra@terra.es"),
			array("Luis","luis@terra.es"),
			array("Ana","ana@terra.es"),
			array("Roberto","roberto@terra.es"),
			array("Maria","maria@terra.es")
			);
	foreach($alumnos as $alumno){
		fputcsv($fichero, $alumno, ";");
	}
	fclose($fichero);

	$fichero = fopen("contactos.csv","r");
	while($registro=fgetcsv($fichero,0,";")){
		echo "Nombre: ".$registro[0]."<br>";
		echo "Correo: ".$registro[1]."<br>";
		echo "<br>";
	}
	fclose($fichero);
?>
```

Creación de ficheros Excel desde PHP.

Podemos crear ficheros Excel desde PHP utilizando la siguiente cabecera:

```
<?php
	header('Content-type: application/vnd.ms-excel');
	header("Content-Disposition: attachment; filename=fichero.xls");
	header("Pragma: no-cache");
	header("Expires: 0");
?>
```

Ejemplo:

excel.php

```
<?php
	header('Content-type: application/vnd.ms-excel');
	header("Content-Disposition: attachment; filename=libros.xls");
	header("Pragma: no-cache");
	header("Expires: 0");
?>
<meta charset="UTF-8">
<table>
	<tr>
		<th>Código</th>
		<th>Titulo</th>
		<th>Aultor</th>
		<th>Precio</th>
	</tr>
```

```
      <tr>
            <td>1</td>
            <td>Internet para Empresas</td>
            <td>Luis Solis Romero</td>
            <td>36.50</td>
      </tr>
      <tr>
            <td>2</td>
            <td>Word 2016</td>
            <td>Ana Ros Sanz</td>
            <td>19.75</td>
      </tr>
</table>
```