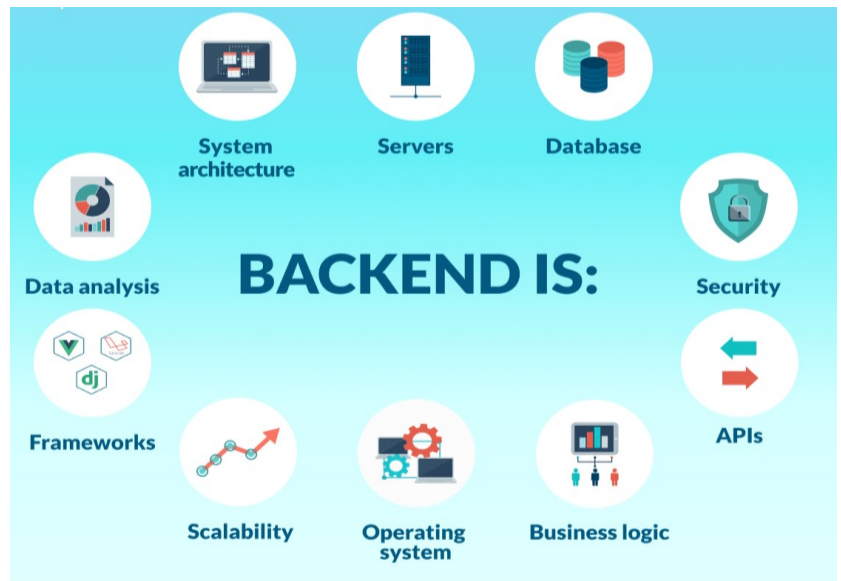


Desarrollo Web: Back End

El Backend developer es el responsable de la programación de un sitio en todos sus componentes funcionales.

Aunque la programación backend se encarga, principalmente, del intercambio de datos entre los servidores web y los usuarios, un desarrollador backend también debe tener conocimientos sobre frontend, porque debe ser responsable de la integración de los elementos frontend que desarrollaron otras personas.



¿Y cuál es ese perfil? **El del desarrollador back-end.**

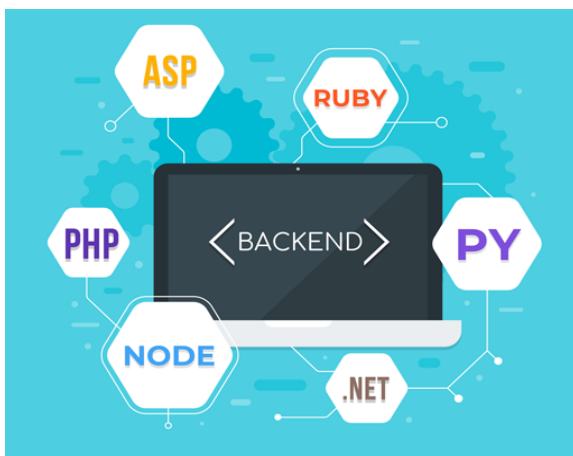
Su función es escribir el código de los servicios web y las APIs que se utilizan en el desarrollo frontend o en el desarrollo de aplicaciones móviles.

Un desarrollador back-end se encarga de que los programas del lado del servidor se ejecuten óptimamente. Una de las funciones principales es asegurar la conexión de la página con los servidores web y las bases de datos.

Debe tener conocimientos sobre:

- PHP → Lenguaje de código abierto para el desarrollo web con contenido dinámico.
- Lenguajes de scripting → Java, Javascript, Python, C++, C#, Ruby, entre otros.
- Base de datos → Oracle, SQL Server, MySQL...

PARA



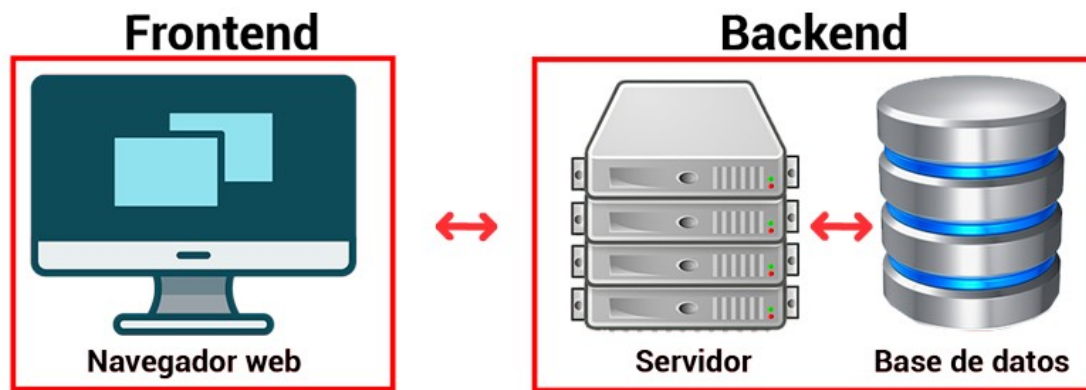
Gestionar la construcción de las funcionalidades para simplificar y/o automatizar el proceso de desarrollo de software.

Manejar, documentar y actualizar las librerías del servidor. Establecer conexiones entre las bases de datos y las soluciones operativas. Configurar y mantener los servidores y BD.

EN OTRAS PALABRAS:

Permite el acceso a la información desde las páginas al servidor de almacenamiento Web, el registro de datos, el procesamiento de solicitudes, ofrece soluciones y reportes, asegura un sistema de acceso a los datos a través de perfiles.

Organización de un sitio Web:



Servidor

Un servidor es un ordenador o equipo informático que se encarga de transmitir información a otros ordenadores que estén conectados a él.

Puede transmitir: datos de BD, archivos de texto, imágenes, videos, etc.

Según el uso que se le vaya a dar tenemos servidores para:

- Almacenamiento web
- Almacenamiento de bases de datos
- Recepción y clasificación de correo electrónico

De esta manera, **un servidor forma parte de una red de ordenadores y es quien responde a las peticiones que hacen estos ordenadores o personas.**

<https://hostingwebcloud.com/que-es-un-servidor/>

Bases de Datos – BD

Una base de datos es una herramienta que **recopila datos, los organiza y los relaciona** para que se pueda hacer una rápida búsqueda y recuperar con ayuda de un ordenador. Hoy en día, las bases de datos también sirven para desarrollar análisis. Las bases de datos más modernas tienen motores específicos para sacar informes de datos complejos.

<https://www.ticportal.es/glosario-tic/base-datos-database>

Conceptos para investigar:

1. ¿Qué tipos de BD, podemos utilizar?
2. ¿Qué es el sistema de gestión de base de datos?
3. Investigar en Internet diferentes aplicaciones para la gestión de BD. Indicar el nombre de las 5 más conocidas y sus costos.
4. ¿A qué hace referencia una base de datos distribuida?
5. ¿Qué son las BD NoSQL, en qué casos se utilizan?

Base de datos relacional

La **base de datos relacional** es una recopilación de la información empresarial organizada de tal forma que se puede consultar, actualizar, analizar y sacar los datos fácilmente. La información se encuentra en **tablas y campos** relacionados entre sí.

Estructura básica de una base de datos

	Nombre	CIF	Teléfono
Ejemplo 1		123456789	000000000
Ejemplo 2		987654321	999999999

Columna o atributo

Fila o registro

tic•PORTAL

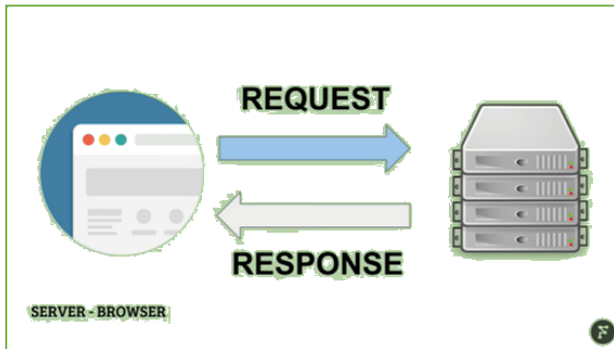
MODELO CLIENTE SERVIDOR

La arquitectura cliente-servidor es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, quien le da respuesta. Esta idea también se puede aplicar a programas que se ejecutan sobre una sola computadora, aunque es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras.

Cliente es un Host (computadora) que puede recibir información o utilizar el servicio de los Servidores.

Servidor es la computadora remota que ofrece información o acceso a servicios.

Entonces, es básicamente un Cliente que solicita algo y un Servidor que lo atiende brindando su presencia en una base de datos.



<https://www.youtube.com/watch?v=49zdlyLSwhQ>

<https://blog.infranetworking.com/modelo-cliente-servidor/>

Ejercitación:

Indicar al menos 3 componentes necesarios para conformar un modelo cliente-servidor.

Investigar aplicaciones para utilizar la pc como un servidor local.

Lenguaje SQL

SQL es un acrónimo en inglés para **Structured Query Language - Lenguaje de Consulta Estructurado**.

Un tipo de **lenguaje de programación** que te permite manipular y descargar datos de una base de datos. Tiene capacidad de hacer cálculos avanzados y álgebra. Es utilizado en la mayoría de empresas que almacenan datos en una base de datos. Ha sido y sigue siendo el lenguaje de programación más usado para bases de datos relacionales.

Comandos básicos de SQL:

SELECT: Permite seleccionar los datos para descargar

WHERE: Permite seleccionar qué filtro aplicar a los a datos descargar

INSERT: Permite insertar datos

DELETE: Permite borrar datos

UPDATE: Permite actualizar los datos

<https://datademia.es/blog/que-es-sql>

CRUD*

El concepto **CRUD** está estrechamente vinculado a la gestión de datos digitales. Hace referencia a **un acrónimo** en el que se reúnen las primeras letras de las cuatro operaciones fundamentales de aplicaciones persistentes en sistemas de BD: **crear, leer, modificar/actualizar y borrar**.



*Para realizar estas operaciones desde **php**, vamos a utilizar **mysqli**, que nos permite conectarnos a la BD y trabajar con cada una de las funciones en particular.

Para conocer las referencias básicas antes de programar con la función `mysqli`, podemos ingresar a la siguiente página e ir a la tabla de contenidos:

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

MySQL:

MySQL es un **sistema de gestión de base de datos** (SGBD) de código abierto. Pertenece actualmente a Oracle. Funciona con un modelo cliente-servidor. Eso quiere decir que los ordenadores que instalan y ejecutan el software de gestión de base de datos se denominan clientes. Cada vez que necesitan acceder a los datos, los clientes se conectan al servidor del sistema de gestión de base de datos y le solicitan la información que necesitan. El servidor se la brinda siempre y cuando tenga los derechos de acceso.



Aparte de su uso como sistema de gestión de base de datos, también es bastante frecuente encontrarse **MySQL** funcionando con los sistemas operativos, servidores y lenguajes de programación de Linux, Apache y PHP/Perl/Python para desarrollar aplicaciones web, por ejemplo, webs dinámicas.

<https://www.hostinger.com.ar/tutoriales/que-es-mysql>
<https://www.oracle.com/ar/mysql/>

phpMyAdmin



Es una herramienta escrita en **PHP** que permite manejar la administración de MySQL, el gestor de bases de datos, desde las páginas Web, de licencia pública.

Podemos crear, añadir, modificar y eliminar tablas, campos, bases de datos. Ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios, exportar BD en diferentes formatos.

PHP

PHP- acrónimo recursivo de **PHP: Hypertext Preprocessor**- es un lenguaje de código abierto utilizado especialmente para el desarrollo web. Favoreciendo la conexión entre los servidores y la interfaz de usuario.



PHP está enfocado principalmente a la programación de scripts del lado del servidor, como, por ejemplo recopilar datos de formularios, generar páginas con contenidos dinámicos, o enviar y recibir cookies. Entre otras cosas.

Es de código abierto, no hay restricciones de uso vinculadas a los derechos. El usuario puede usar PHP para programar en cualquier proyecto y comercializarlo sin problemas. Este lenguaje está en constante perfeccionamiento, gracias a una comunidad de desarrolladores proactiva y comprometida.

<https://www.php.net>

El PHP es definido como un **lenguaje del lado del servidor**. Esto significa que se aplica en la programación que tiene lugar en el servidor web responsable de ejecutar la aplicación o, más a menudo, en un sitio web. por lo tanto debemos ejecutar la programación desde un servidor web o un servidor local.



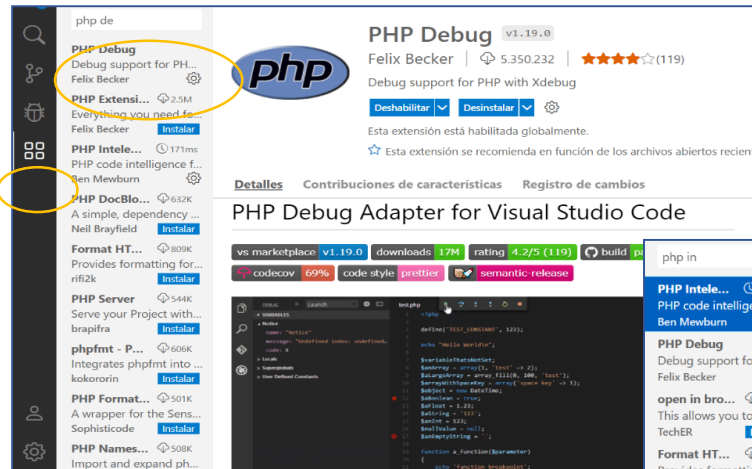
<https://desarrolloweb.com/home/php>

<https://rockcontent.com/es/blog/php/>

PHP es un lenguaje interpretado, es decir, no es necesario compilar el código para traducirlo a código de máquina. Se ejecuta en cada petición (nace, se ejecuta y muere, una y otra vez).

Teniendo en cuenta las extensiones y módulos que lo adaptan para una innumerable cantidad de tareas, no es un lenguaje para crear videojuegos, animaciones, ni tampoco aplicaciones de escritorio.

Para facilitar la programación en php, desde Visual Code, podemos instalar las extensiones PHP Debug, PHP Intelephense:



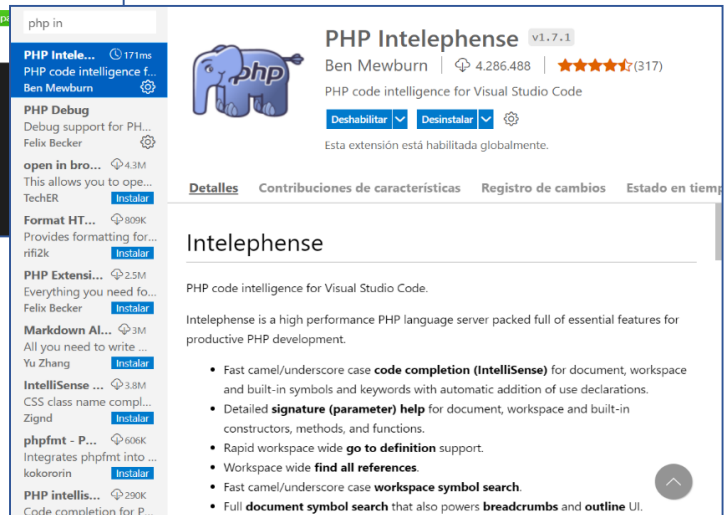
Herramientas de debugging PHP

Es un complemento que permite, entre otras cosas: Ejecutar el código paso por paso. Evaluar el contenido de las variables en tiempo real. Establecer puntos de corte.

PHP Intelephense

Ayuda detallada de los parámetros para documentos, constructores, métodos y funciones.

Analiza todos los archivos de tu proyecto y proporciona información necesaria para el auto-completado.



Para crear nuestro primer programa en php, debemos tener en cuenta que alojaremos el mismo en el servidor local – localhost, al igual que la BD creada, podemos crear una carpeta de trabajo nueva para organizar nuestros archivos y desde el explorador de Visual Code nos ubicamos en la carpeta creada para comenzar a realizar nuestros archivos php.

Sintaxis Básica Un script php se puede colocar en cualquier parte del documento HTML, encerrado en la siguiente etiqueta: `<?php.....;?>`

Un archivo php es una concatenación de código HTML y código propio de php. Las instrucciones php terminan con un punto y coma (;). En las instrucciones no diferencia entre mayúsculas y minúsculas (echo = Echo = ECHO) pero si en las variables (cantidad ≠ CANTIDAD). Para comentarios usar // o # si es una sola línea, si es un bloque /* comentario */. Para mostrar en pantalla se usa la instrucción echo o print.

<https://www.php.net/manual/es/tutorial.firstpage.php>

Actividades de repaso:

- Ver el siguiente link sobre características básicas de BD:

<https://www.youtube.com/watch?v=6S8A-1jBD5Y>

Indicar como se ordena la información y ejemplificar el uso de BD. Indicar que es un DBMS. ¿En qué se basa el funcionamiento de estas BD, o banco de datos? El almacenamiento en tabla facilita la eficacia en 4 aspectos fundamentales ¿cuáles son? ¿Qué es SQL? ¿Qué tipo de modelos de consulta conocen? ¿Qué características tiene la clasificación por variabilidad? ¿Qué característica tiene la clasificación por contenidos?

- Ver el siguiente link para repasar los conceptos básicos de BD:

<https://www.youtube.com/watch?v=yoeV4Ex8C8U>

XAMPP

Es un servidor independiente de plataforma de código libre. Permite instalar Apache y otras aplicaciones de uso práctico para del desarrollo backend. Se instala en cualquier S.O. de forma gratuita.



Esta herramienta de desarrollo permite probar el trabajo (páginas web o programación por ejemplo) en tu propio ordenador sin necesidad de tener que acceder a internet.

Es una distribución de Apache que incluye diferentes softwares libres. El nombre es un acrónimo compuesto por las iniciales de los programas que lo constituyen:

Linux: Es el sistema operativo donde estará instalado nuestra aplicación. A diferencia de Windows, Linux es una distribución libre que es segura, no requiere pago de licencias y tiene alto rendimiento.

Apache: el servidor web de código abierto es la aplicación usado globalmente para la entrega de contenidos web. Las aplicaciones del servidor son ofrecidas como software libre por la Apache Software Foundation.

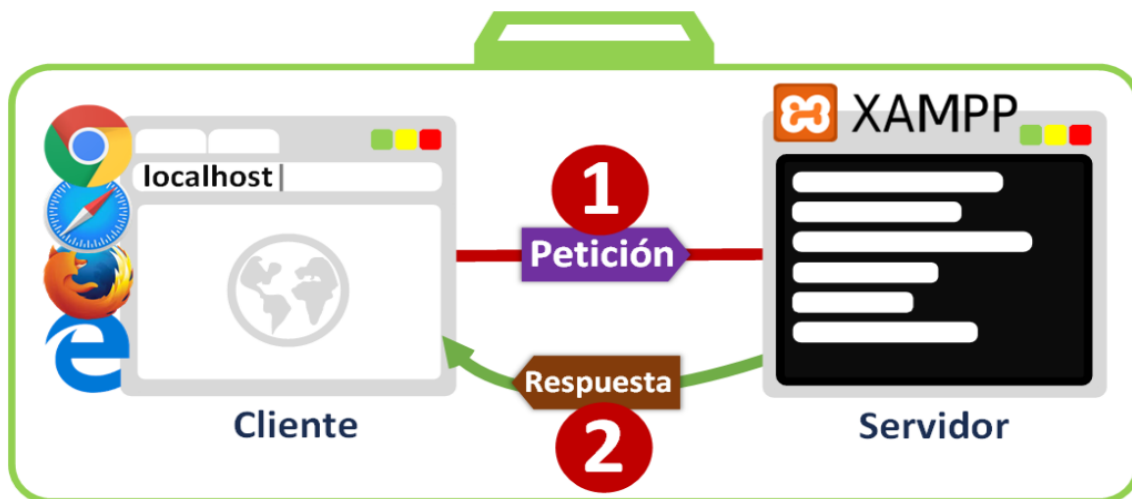
MySQL/MariaDB: sistemas relacionales de gestión de bases de datos más populares del mundo. Sirve para el almacenamiento de datos para servicios web.

PHP: es un lenguaje de programación de código de lado del servidor que permite crear páginas web o aplicaciones dinámicas. Es independiente de plataforma y soporta varios sistemas de bases de datos.

Perl: este lenguaje de programación se usa en la administración del sistema, en el desarrollo web y en la programación de red. También permite programar aplicaciones web dinámicas.

Además de estos componentes principales, esta distribución gratuita también incluye, según el sistema operativo, otras herramientas como el servidor de correo Mercury, el programa de administración de bases de datos **phpMyAdmin**, el software de analítica web Webalizer, OpenSSL, Apache Tomcat y los servidores FTP FileZilla o ProFTPd.

<https://www.nettix.com.pe/blog/web-blog/que-es-xampp-y-como-puedo-usarlo>



El paquete integrado XAMPP nos va a facilitar el Desarrollo Web desde nuestra computadora:

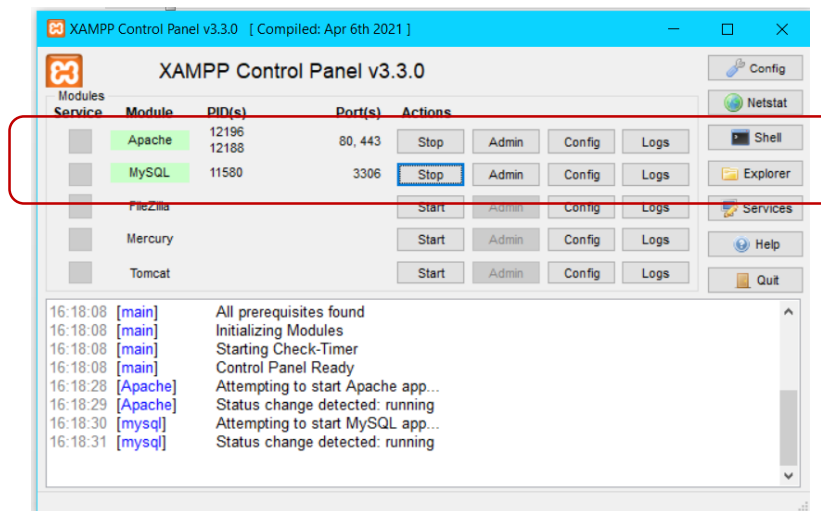
- Instala el servidor Web Apache
- Instala MySQL (SGBD) y phpMyAdmin
- Instala el intérprete de programación de PHP

Trabajo Práctico:

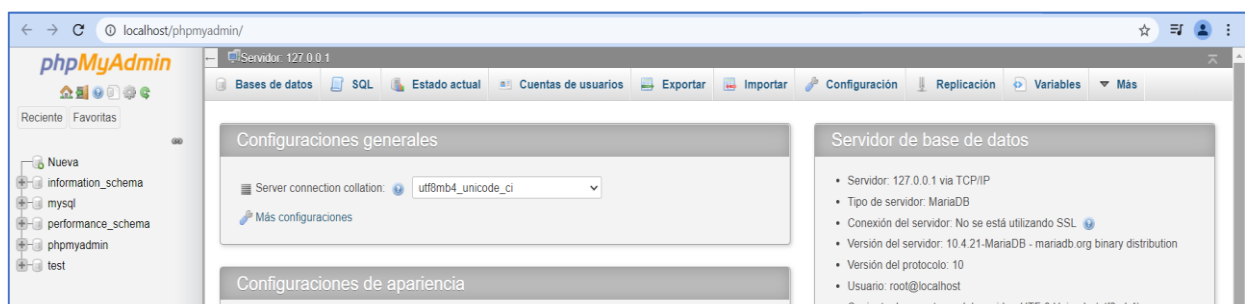
1 - Crear y agregar datos en una BD desde phpMyAdmin

Pasos:

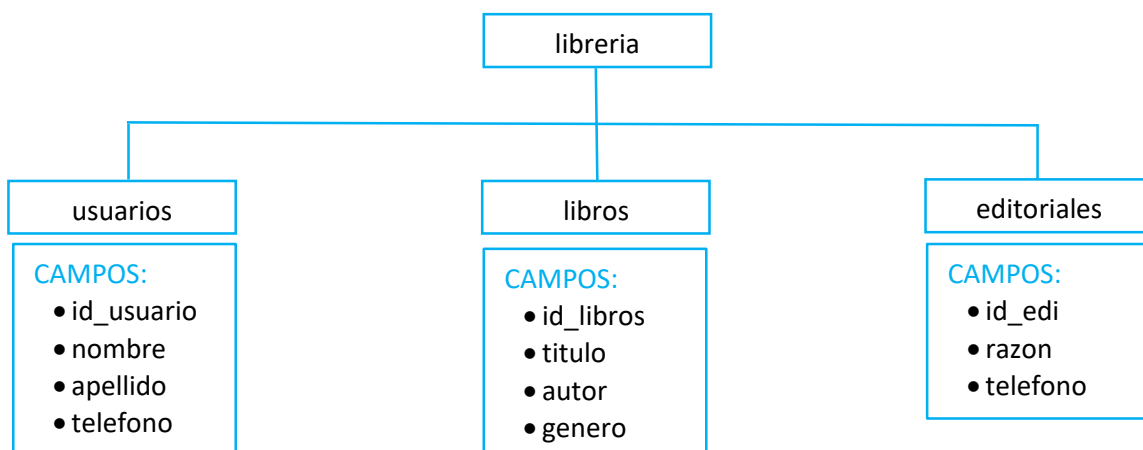
- Instalar **XAMPP** y ejecutar el servidor **Apache** y el SGBD **MySQL** que nos permite trabajar con el administrador **phpMyAdmin**



Al presionar el botón <ADMIN> de MySQL, nos carga en el navegador la interfaz de phpMyAdmin para que podamos trabajar de manera más sencilla con las BD SQL:



- Crear una DB nueva, con phpMyAdmin, teniendo en cuenta la siguiente estructura:



Recordar que en la definición de los nombres de la BD, las tablas y los campos, la utilización de mayúsculas y/o minúsculas es muy importante, no se pueden utilizar espacios o caracteres especiales, sugiero no utilizar acentos, y en caso de querer usar más de una palabra unir las con el guion bajo “_”.

2 – Crear un programa en PHP, utilizar Visual Code como editor de código:

Definimos el bloque de programación php con:

```
<?php
.....instrucciones php;
?>
```

localhost/basededatos1/prueba1.php

primer programa en phpCódigo de ingreso45

En las líneas 2 y 3, se asignan las variables de manera directa, quedando definido el tipo de variables por su asignación, en este caso: **\$nro** es variable de tipo numérica y **\$nombre** variable de tipo alfanumérica. Todas las variables de php comienzan con el signo \$.

En la línea 4 y 5 utilizamos la instrucción **echo** que muestra cadena de caracteres o variables.

Si queremos mejorar la visualización de nuestras líneas, tenemos que intercalar programación HTML y PHP.

Ejemplo #3 Mezcla de los modos HTML y PHP

```
<?php
if (strpos($_SERVER['HTTP_USER_AGENT'], 'MSIE') !== FALSE) {
?>
<h3>strpos() debe haber devuelto no falso</h3>
<p>Está usando Internet Explorer</p>
<?php
} else {
?>
<h3>strpos() debe haber devuelto falso</h3>
<p>No está usando Internet Explorer</p>
<?php
}
?>
```

Ejemplo #2 Mostrar información de nuestro formulario

```
Hola <?php echo htmlspecialchars($_POST['nombre']); ?>.
Usted tiene <?php echo (int)$_POST['edad']; ?> años.
```

➔ Métodos para manipular información entre el formulario y el servidor:

GET y POST son métodos de envío de la información de los formularios. Cada método tiene sus ventajas y sus inconvenientes, elegir entre un método y otro depende de la aplicación concreta que se esté desarrollando.

MÉTODO	CONCEPTO	OBSERVACIONES
GET	GET lleva los datos de forma "visible" al cliente (navegador web). El medio de envío es la URL. Los datos los puede ver cualquiera.	Los datos son visibles por la URL, por ejemplo: www.aprenderaprogramar.com/action.php?nombre=pedro&apellidos1= gomez
POST	POST consiste en datos "ocultos" (porque el cliente no los ve) enviados por un formulario cuyo método de envío es post. Es adecuado para formularios. Los datos no son visibles.	La ventaja de usar POST es que estos datos no son visibles al usuario de la web. En el caso de usar get, el propio usuario podría modificar la URL escribiendo diferentes parámetros a los reales en su navegador, dando lugar a que la información tratada no sea la prevista.

POST: El método HTTP POST envía datos al servidor. Una solicitud POST es enviada por un [formulario HTML](#) y resulta en un cambio en el servidor.

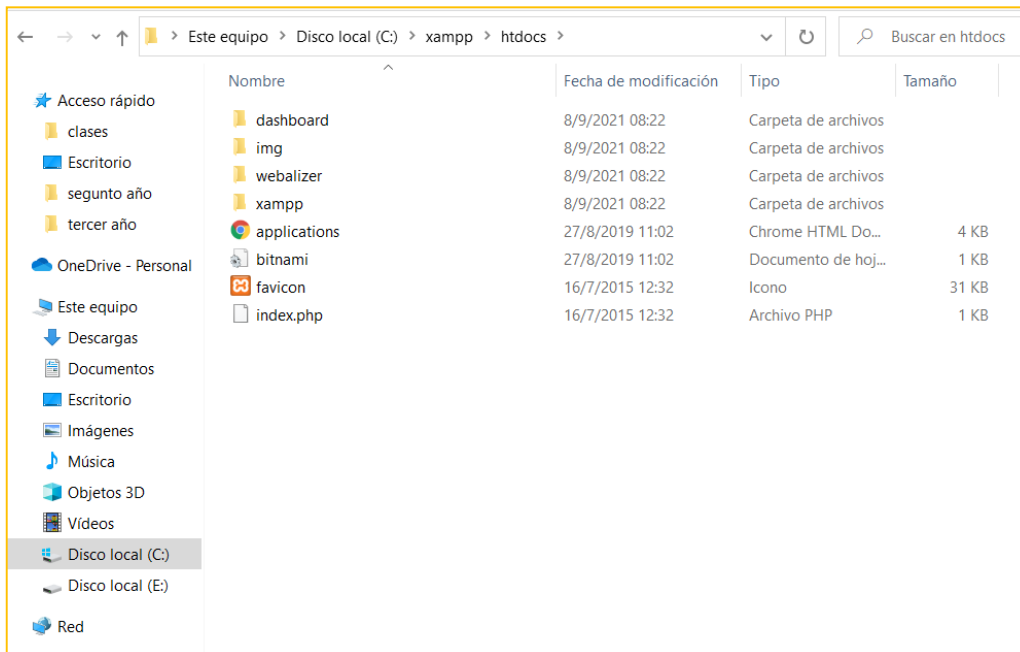
<https://www.php.net/manual/es/refs.basic.vartype.php>

Desarrollar un “Sitio Web Dinámico”, que permita crear, modificar y borrar registros de una BD del servidor desde una página a través de un formulario de usuario:

Debemos tener en cuenta que XAMPP permite que **nuestra PC** se convierta en **un servidor local**. Al instalar XAMPP en la computadora, esta aplicación genera en la carpeta “XAMPP” un lugar de trabajo para el material que vamos a desarrollar y/o utilizar desde el navegador, esta carpeta tiene el nombre de **“htdocs”** este va a ser el lugar desde donde nuestra PC (como SERVIDOR LOCAL: **LOCALHOST**) va a cargar los archivos al navegador.

PASO A PASO

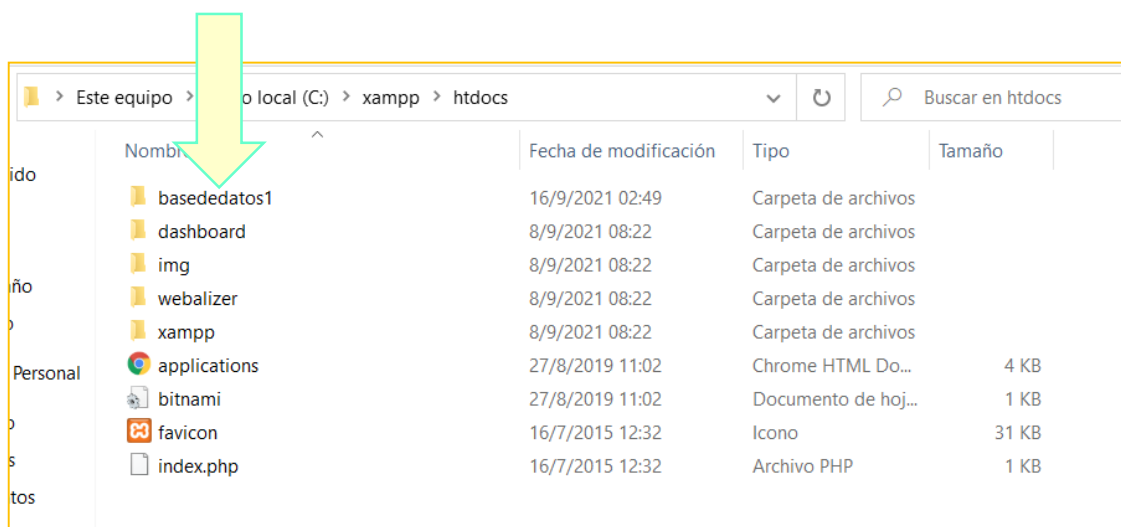
La carpeta **htdocs** es la raíz del servidor web local, es el lugar donde se deben almacenar los archivos .html, .csss, .js, .php, ect. que desarrollemos para nuestro sitio y para trabajar con BD:



Comenzamos:

- 1- Crear una carpeta de trabajo, en el ejemplo cree la carpeta “basededatos1”, pueden utilizar el nombre que les resulte más adecuado para su trabajo:

basededatos1



En la carpeta **basededatos1**, debemos vamos a crear una página html que contenga el diseño de un formulario:

Ej.:

Formulario Básico

Libros En Línea

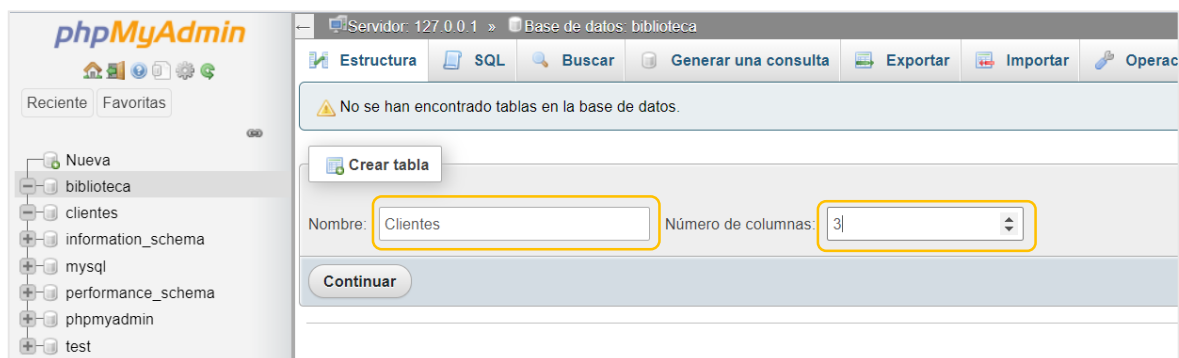
2- Ingresar a phpMyAdmin desde el panel de control de XAMPP

3- Crear una BD nueva, indicar **el nombre** (por ejemplo “Biblioteca”), en **el cotejamiento** dejar el UTF-8 sugerido y luego presionar el botón CREAR:

Base de datos	Cotejamiento	Acción
<input type="checkbox"/> clientes	utf8mb4_general_ci	Seleccionar privilegios
<input type="checkbox"/> information_schema	utf8_general_ci	Seleccionar privilegios
<input type="checkbox"/> mysql	utf8mb4_general_ci	Seleccionar privilegios
<input type="checkbox"/> performance_schema	utf8_general_ci	Seleccionar privilegios
<input type="checkbox"/> phpmyadmin	utf8_bin	Seleccionar privilegios
<input type="checkbox"/> test	latin1_swedish_ci	Seleccionar privilegios

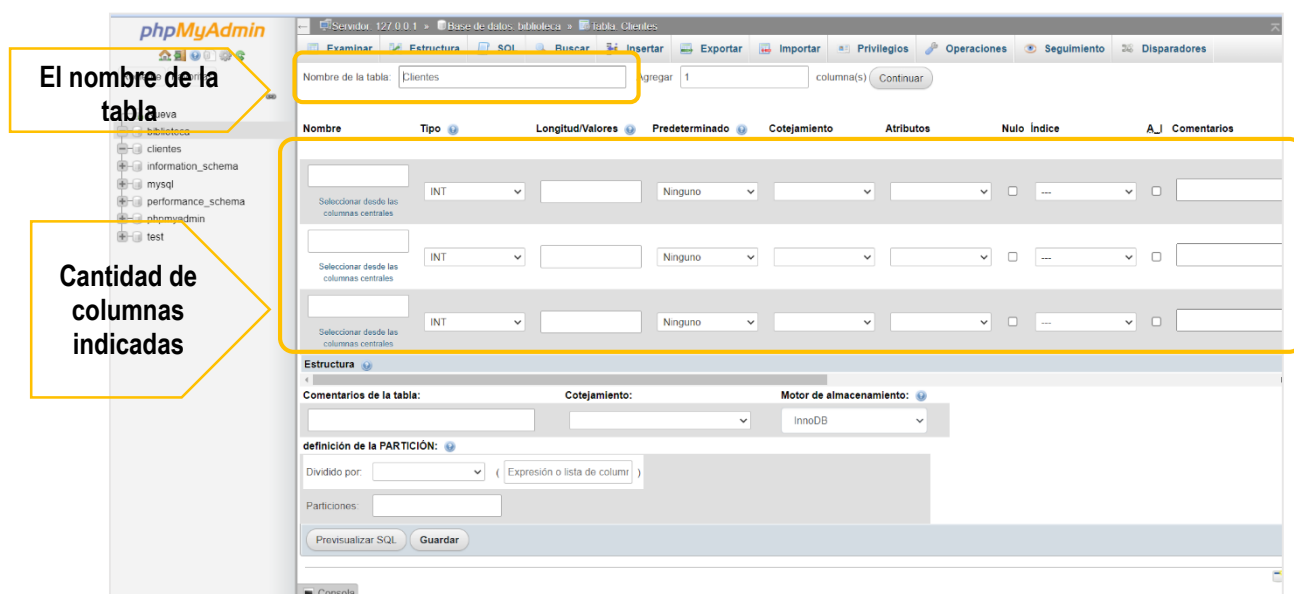
Total: 6

- 4- Creamos las tablas necesarias para nuestra BD, indicando su **nombre** y **cantidad de columnas**, luego presionamos el botón de CONTINUAR:

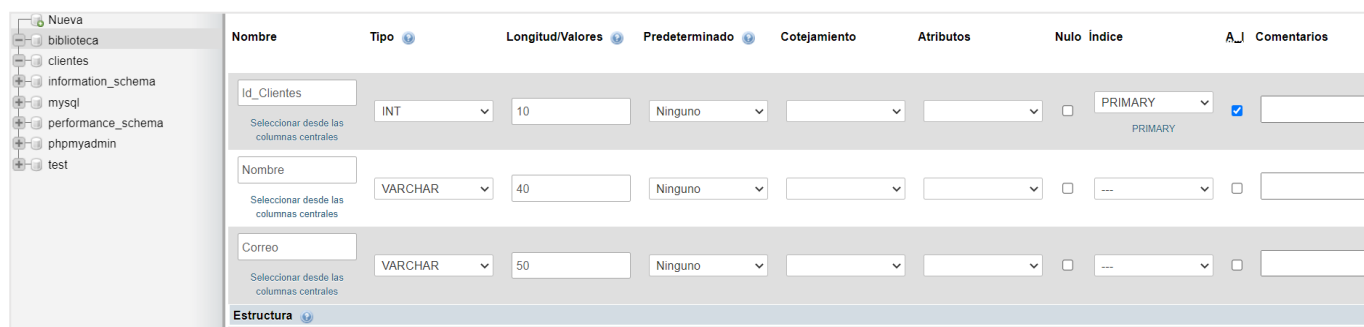


En el ejemplo, el nombre de la tabla es Clientes y el número de columnas 3, porque en este caso solo necesito almacenar: ID_Clientes, Nombre y Correo.

Al continuar, phpMyAdmin, nos presenta la tabla que creamos, teniendo en cuenta los siguientes:



- 5- Una vez creada la tabla de datos “Clientes”, asignamos las propiedades a los campos:



Tener en cuenta que

- **Tipo de datos** nos permite indicar si se va a almacenar números o textos, por lo general se usa **INT** para números enteros y **VARCHAR** para almacenar cadena de caracteres.
- **Longitud** nos permite indicar el máximo de caracteres que podemos escribir en ese campo.
- **Indice** nos permite identificar el campo que es la **llave primaria**.
- **A_I** significa **auto_incrementable** y se utiliza para los campos que se incrementan solos, es decir, cada vez que hago un registro se incrementa en uno.

Por ejemplo

En el primer campo indicamos:

Id_Clientes	INT	10	Ninguno				PRIMARY	<input checked="" type="checkbox"/>	
<small>Seleccionar desde las columnas centrales</small>									
<small>PRIMARY</small>									

nombre: Id_Clientes

Tipo de datos: INT

Longitud: 10

Indice: Primario

Autoincrementable

En el segundo campo indicamos:

Nombre	VARCHAR	40	Ninguno				---	<input type="checkbox"/>	
<small>Seleccionar desde las columnas centrales</small>									

nombre: Nombre

Tipo de datos: VARCHAR

Longitud: 40

Si presionamos en PREVISUALIZAR nos muestra, las instrucciones utilizadas desde el lenguaje de consulta SQL, para crear la tabla y los campos de la misma:

```
CREATE TABLE `biblioteca`.`clientes` ( `Id_Clientes` INT(10) NOT NULL AUTO_INCREMENT , `Nombre` VARCHAR(40) NOT NULL , `Correo` VARCHAR(50) NOT NULL , PRIMARY KEY (`Id_Clientes`)) ENGINE = InnoDB;
```

* Copiar en la carpeta la instrucción para crear registros en SQL

Al terminar de Asignar los campos, presionamos el botón GUARDAR:

Dividido por:	(Expresión o lista de columnas)
Particiones:	
Previsualizar SQL	Guardar

6- Al GUARDAR, nos muestra la tabla creada, la solapa ESTRUCTURA nos muestra los campos con los datos asignados:

The screenshot shows the phpMyAdmin interface. On the left, a sidebar lists databases: 'biblioteca', 'clientes', 'information_schema', 'mysql', 'performance_schema', 'phpmyadmin', and 'test'. The 'clientes' database is selected, and the 'clientes' table is highlighted. The main area shows the 'Estructura de tabla' (Table Structure) tab. It displays the table's schema with three columns:

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	Id_Clientes	int(10)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Más
2	Nombre	varchar(40)	utf8mb4_general_ci		No	Ninguna			Cambiar Eliminar Más
3	Correo	varchar(50)	utf8mb4_general_ci		No	Ninguna			Cambiar Eliminar Más

Below the table structure, there are options to 'Seleccionar todo' (Select all) and 'Para los elementos que están marcados' (For the marked elements), with actions like 'Examinar' (Examine), 'Cambiar' (Change), 'Eliminar' (Delete), 'Primaria' (Primary), 'Único' (Unique), and 'Índice' (Index). At the bottom, there is a 'Continuar' (Continue) button.

Corroborar que todos los datos asignados son los que corresponden a nuestro diseño de BD.

7- En la solapa INSERTAR podemos agregar registros a la BD:

The screenshot shows the phpMyAdmin interface with the 'Insertar' (Insert) tab selected for the 'clientes' table. The form displays the following fields:

Columna	Tipo	Función	Nulo	Valor
Id_Clientes	int(10)			
Nombre	varchar(40)			Cristian Plaza
Correo	varchar(50)			CristianPlaza_2@gmail.com

Below the form, there is a 'Continuar' (Continue) button. The interface also includes a sidebar with a database tree and a top navigation bar with options like 'Examinar', 'Estructura', 'SQL', 'Insertar', 'Exportar', 'Importar', 'Privilegios', 'Operaciones', 'Seguimiento', and 'Disparadores'.

Al completar los datos debemos tener en cuenta que los campos **auto_incrementable**, no debemos completarlos, ya que se asignan automáticamente.

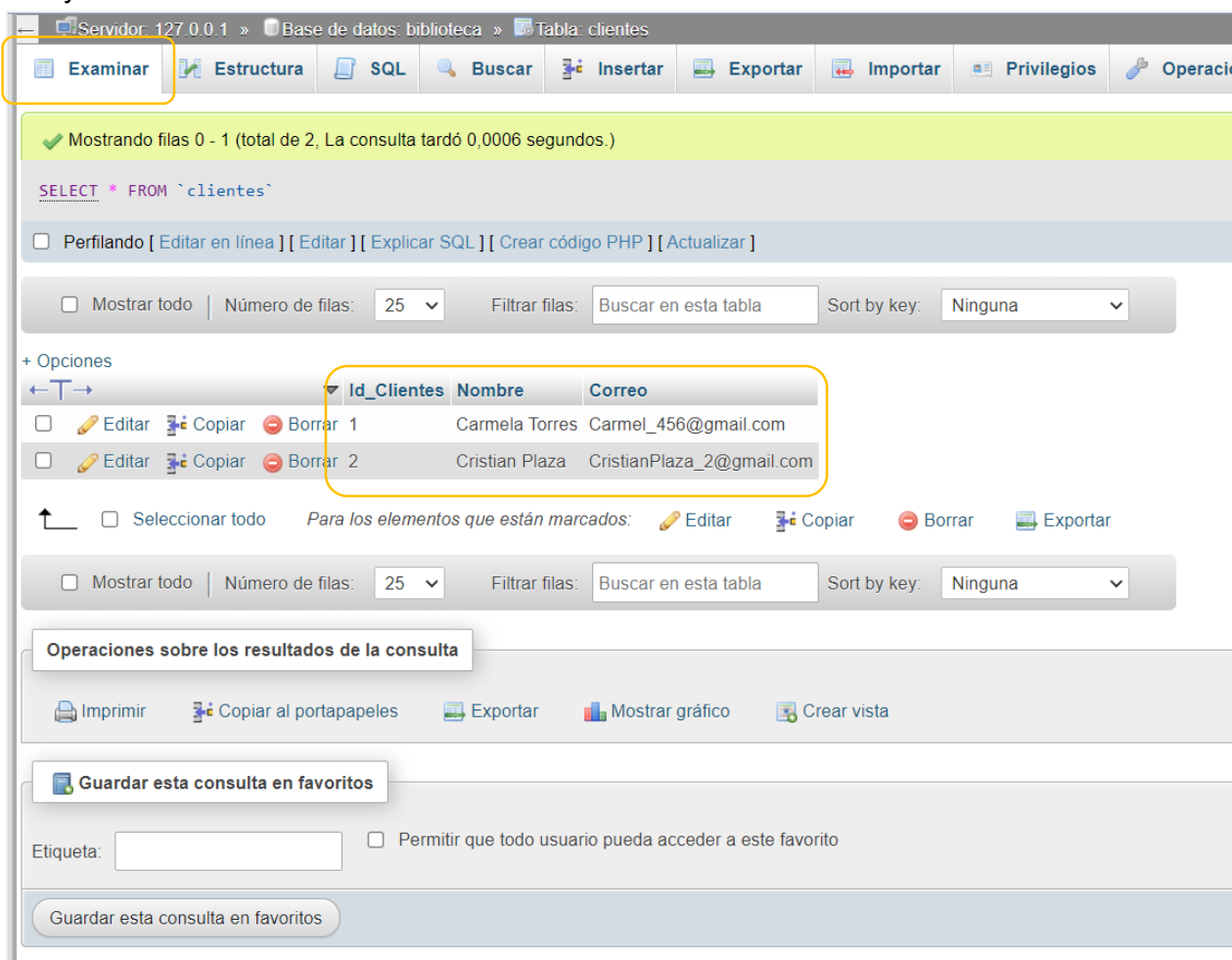
Al presionar el botón CONTINUAR, nos muestra la instrucción SQL para insertar registros:

The screenshot shows the phpMyAdmin interface displaying the SQL query generated for inserting a record into the 'clientes' table. The query is:

```
INSERT INTO `clientes` (`Id_Clientes`, `Nombre`, `Correo`) VALUES (NULL, 'Cristian Plaza', 'CristianPlaza_2@gmail.com');
```

Below the query, there is a 'Continuar' (Continue) button. The interface also includes a sidebar with a database tree and a top navigation bar with options like 'Examinar', 'Estructura', 'SQL', 'Insertar', 'Exportar', 'Importar', 'Privilegios', 'Operaciones', 'Seguimiento', and 'Disparadores'.

- 8- Como último paso, vamos a la solapa EXAMINAR, y comprobamos que nuestros registros se hayan realizado de manera correcta:



Tener en cuenta que la BD creada con MySQL, se guarda dentro de la carpeta:
Xampp/mysql/data/nombre_de_BD

Ejercitación adicional:

1. Transcribir las instrucciones correspondientes al CRUD de SQL en la carpeta.
2. Buscar los tipos de variables de PHP, formato de escritura, instrucciones repetitivas y condicionales.

MYSQLI extensión mejorada de MySQL

Las funciones de MySQLi le permiten acceder a servidores de bases de datos MySQL. Son funciones de PHP, que permiten hacer la conexión, gestionar consultas y manipular los registros de la BD alojada en el servidor.

Es decir que para poder trabajar con BD desde un sitio Web, debemos utilizar las instrucciones de la función `mysqli()`, por ejemplo:

```
7 $conn1=new mysqli("localhost","root","","biblioteca");
```

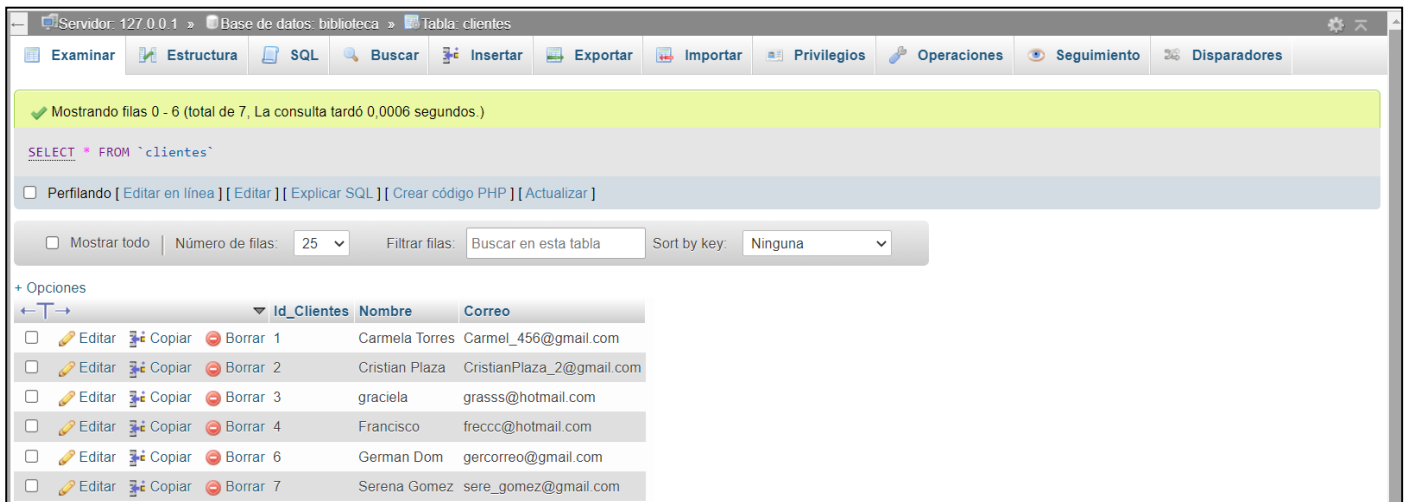


Indicar otras funciones de mysqli:

https://www.w3schools.com/php/php_ref_mysqli.asp

Agregar y leer registros de la BD con PHP

Corroborar los nombres de la BD, antes de programar:



Ejercicio:

- 1- Agregar un registro sin utilizar datos variables, solo asignación directa a través de la instrucción INSERT

\$agregar="INSERT INTO clientes(Nombre, Correo) VALUES ('Dario Lopez','daro3454 @hotmail.com')";

```
prueba1.php  agregardatos.php x
agregardatos.php > ...
1  <?php
2
3  $servername="localhost";
4  $username="root";
5  $password="";
6  $dbname="biblioteca";
7  $conn1= new mysqli($servername,$username, $password,$dbname);
8
9  $sql="SELECT * FROM clientes";
10
11 $agregar="INSERT INTO clientes( Nombre, Correo) VALUES ('Dario Lopez',
12 'daro3454@hotmail.com')";
13 $res=mysqli_query($conn1, $agregar);
```

Podemos hacer la asignación de los valores de conexión de manera directa:

```
7  $conn1=new mysqli("localhost","root","","biblioteca");
```

O utilizar variables donde asignamos los valores requeridos:

```
3  $servername="localhost";
4  $username="root";
5  $password="";
6  $dbname="biblioteca";
7  $conn1= new mysqli($servername,$username, $password,$dbname);
```

```

3  $servername="localhost";
4  $username="root";
5  $password="";
6  $dbname="biblioteca";
7  $conn1= new mysqli($servername,$username, $password,$dbname);
8
9  $sql="SELECT * FROM clientes";
10
11  $agregar="INSERT INTO clientes( Nombre, Correo) VALUES ('Dario Lopez
'daro3454@hotmail.com')";
12  $res=mysqli_query($conn1, $agregar);
13

```

Líneas 3 a 7: conexión a la BD.

Línea 9: instrucción para asignar la consulta a tabla determinada de la BD, en este caso se selecciona la tabla “clientes” y hacemos referencia a todos los campos.

Líneas 11: Guardamos en una variable de PHP el código SQL que utilizamos para agregar datos.

Línea 12: asignamos la consulta* a la BD conectada. (* La consulta en este caso corresponde a agregar campos).

Corroborar las asignaciones desde phpMyAdmin, antes de continuar.

2 – Agregar una instrucción repetitiva que permita mostrar todos los registros de la BD en la página web:

```

3  $servername="localhost";
4  $username="root";
5  $password="";
6  $dbname="biblioteca";
7  $conn1= new mysqli($servername,$username, $password,$dbname);
8
9  $sql="SELECT * FROM clientes";
10
11  $agregar="INSERT INTO clientes( Nombre, Correo) VALUES ('Dario Lopez
'daro3454@hotmail.com')";
12  $res=mysqli_query($conn1, $agregar);
13
14  $res=mysqli_query($conn1,$sql);
15  while($row=mysqli_fetch_assoc($res)){
16      echo $row['Id_Clientes'];
17      echo $row['Nombre'];
18      ?> <br>
19      <?php
20  }
21  mysqli_free_result($res);
22  mysqli_close($conn1);
23  ?>

```

Línea 14 a 19: repetimos con WHILE la muestra de los registros de la tabla.

Línea 22: cerramos la consulta a la BD.

Los registros pueden mostrarse de manera informal usando la instrucción ECHO de PHP o podemos mejorar la presentación intercalando esta programación a un diseño realizado con HTML.

3 – Leer un registro determinado, de la BD y mostrarlo:

Asignamos el valor a buscar en una variable y luego realizamos la consulta teniendo en cuenta la variable, y el CAMPO de la TABLA asociado a la variable:

```
8   $id=23;
9   $sql="SELECT * FROM clientes WHERE Id_Clientes=$id";
10
```

En este caso el valor a buscar está asignado a la variable \$id, en la línea 8, a los efectos de corroborar el correcto funcionamiento de la instrucción se le asigna un numero; el 23.

Ejemplo:

```
8
9   $sql="SELECT * FROM clientes WHERE Id_Clientes=21";
10
11  $res=mysqli_query($conn1,$sql);
12  $row=mysqli_fetch_assoc($res);
13
14  echo $row['Id_Clientes'];
15  echo $row['Nombre'];
16
17  mysqli_free_result($res);
18  mysqli_close($conn1);
19  ?>
```

* En este caso, se asignó el valor a buscar directamente a la consulta, línea 9.

Al momento de realizar esta función lo correcto es ingresar el valor a buscar desde un formulario y luego asignarlo a una variable.

- Instrucción REQUIRE_ONCE:

Podemos hacer un código de conexión php y llamarlo desde otro archivo php con “requiere_once”. Se utiliza para códigos que se repiten en distintos archivos, como la conexión a BD:

```
1  <?php
2  require_once("conexionlocal1.php");
3  $sql="SELECT * FROM clientes";
4  //$res=$conn1->query($sql);
5
6  //.....siguiendo el ejemplo de lo otra forma de conectar.....
7  $res=mysqli_query($conn1,$sql);
8
```

- \$_POST:

Para asignar los valores ingresados en el formulario a una variable PHP, debemos utilizar el método POST:

```
36  <?php
37  if (isset($_POST['enviar'])){
38      $nom=trim($_POST['usuario']);
39      $corr=trim($_POST['correo'])
40  }
41  if (empty($nom)){
42      echo "ingrese un nombre de usuario";
43  }
```

Definir: “isset”, “trim”, “empty”

4 - Crear un formulario HTML, que nos permita ingresar datos y enviarlos a la tabla de la BD conectada:

```
1  <?php
2
3  $servername="localhost";
4  $username="root";
5  $password="";
6  $dbname="biblioteca";
7  $conn1= new mysqli($servername,$username, $password,$dbname);
8
9  $sql="SELECT * FROM clientes";
10
11 $res=mysqli_query($conn1,$sql);
12 ?>
13
14 <!DOCTYPE html>
15 <html lang="en">
16 <head>
17     <meta charset="UTF-8">
18     <meta http-equiv="X-UA-Compatible" content="IE=edge">
19     <meta name="viewport" content="width=device-width, initial-scale=1.0">
20     <title>Formulario1-Agregar</title>
21
22 </head>
23 <body>
24     <h1>Formulario Básico</h1>
25     <form method="post">
26         <h2>Libros En Línea</h2>
27         <input TYPE="text" maxlength="30" name= "usuario" placeholder="nombre_de_usuario">
28         <br>
29         <input type="text" maxlength="50" placeholder="correo"> <br>
30         <br>
31
32         <input type="submit" name="enviar" >
33
34     </form>
35
36 <?php
37 if (isset($_POST['enviar'])){
38     $nom=trim($_POST['usuario']);
39     $corr=trim($_POST['correo'])
40 }
41
42 if (!empty($nom) && !empty($corr)){
43     $agregar="INSERT INTO clientes( Nombre, Correo) VALUES ('$nom','$corr')";
44     $res=mysqli_query($conn1, $agregar);
45     echo "se ingresaron correctamente los datos";
46 }
47 mysqli_free_result($res);
48 mysqli_close($conn1);
49 ?>
50
51 </body>
```

Formulario Básico

Libros En Línea

Resumen: CRUD en PHP

Agregar:

```
prueba1.php  agregardatos.php x
agregardatos.php > ...
1  <?php
2
3  $servername="localhost";
4  $username="root";
5  $password="";
6  $dbname="biblioteca";
7  $conn1= new mysqli($servername,$username, $password,$dbname);
8
9  $sql="SELECT * FROM clientes";
10
11 $agregar="INSERT INTO clientes( Nombre, Correo) VALUES ('Dario Lopez',
12 'daro3454@hotmail.com')";
13
14 $res=mysqli_query($conn1, $agregar);
15
16 $res=mysqli_query($conn1,$sql);
17 while($row=mysqli_fetch_assoc($res)){
18     echo $row['Id_Clientes'];
19     echo $row['Nombre'];
20     ?> <br>
21 }
22 mysqli_free_result($res);
23 mysqli_close($conn1);
24 ?>
```

Leer:

```
prueba1.php  agregardatos.php x  leer.php  x
leer.php > ...
1  <?php
2
3  $servername="localhost";
4  $username="root";
5  $password="";
6  $dbname="biblioteca";
7  $conn1= new mysqli($servername,$username, $password,$dbname);
8
9  $sql="SELECT * FROM clientes WHERE Id_Clientes=21";
10
11 $res=mysqli_query($conn1,$sql);
12 $row=mysqli_fetch_assoc($res);
13
14 echo $row['Id_Clientes'];
15 echo $row['Nombre'];
16
17 mysqli_free_result($res);
18 mysqli_close($conn1);
19 ?>
```

```
8  $id=23;
9  $sql="SELECT * FROM clientes WHERE Id_Clientes=$id";
10
```

Borrar:

```
prueba1.php  agregardatos.php x  aa.php 1  leer.php  borrar.php x
borrar.php > ...
3  $servername="localhost";
4  $username="root";
5  $password="";
6  $dbname="biblioteca";
7  $conn1= new mysqli($servername,$username, $password,$dbname);
8  $id=23;
9  $sql="SELECT * FROM clientes WHERE Id_Clientes=$id";
10
11 $res=mysqli_query($conn1,$sql);
12 $row=mysqli_fetch_assoc($res);
13
14 echo $row['Id_Clientes'];
15 echo $row['Nombre'];
16 echo "*****";
17 echo "****hacer un cartel de precaución****";
18 echo "***agregar un boton de borrar antes**";
19 echo "*****";
20
21 $borrar= "DELETE FROM clientes WHERE Id_Clientes=$id";
22 $res2=mysqli_query($conn1, $borrar);
23 echo "YAAAA SE BORROOOOOO";
24 mysqli_free_result($res);
25
26 mysqli_close($conn1);
27 ?>
28
```

Actualizar:

```
1 UPDATE `clientes` SET `Id_Clientes`='[value-1]',`Nombre`='[value-2]',`Correo`='[value-3]' WHERE 1
```

Crear:

```
1 INSERT INTO `clientes`(`Id_Clientes`, `Nombre`, `Correo`) VALUES ('[value-1]','[value-2]','[value-3]')
```

Leer:

```
1 SELECT `Id_Clientes`, `Nombre`, `Correo` FROM `clientes` WHERE 1
```

Borrar:

```
1 DELETE FROM `clientes` WHERE 0
```