Uno de los empleos principales de PHP es el acceso a una base de datos en el servidor. Las operaciones básicas se hacen empleando como lenguaje el SQL.

PHP implementa distintas funciones según la base de datos a emplear. Existen funciones actualmente para acceder a las siguientes servidores de bases de datos:

- MySQL
- MariaDB
- Microsoft SQL Server
- Oracle
- PostgreSQL
- SysBase
- FrontBase
- Informix
- InterBase
- Ingres
- mSQL
- dBase
- SQLlite

El más empleado en la actualidad en la web es el gestor de base de datos MySQL.

Cuando instaló el entorno de XAMPP en un principio para trabajar con PHP, se instaló el MySQL o MariaDB.

Para crear una base de datos el XAMPP instala también un programa codificado en PHP que nos permite interactuar con el MySQL o MariaDB.
Este programa se llama PHPMyAdmin (como veremos nos permite crear las bases de datos, tablas, índices, usuarios etc.)

Para iniciar el PHPMyAdmin debemos presionar el botón "Admin" de MySQL en el panel de control de XAMPP:

Arrancar el PHPMyAdmin con XAMPP
Como podemos ver la interfaz del PHPMyAdmin es un programa que se ejecuta en la web:

PHPMyAdmin
Para crear una base de datos procedemos a seleccionar la pestaña "Base de datos" e ingresamos como nombre "base1" y presionamos el botón crear:
PHPMyAdmin creación de una base de datos
Luego de crear la base de datos podemos ver que aparece en el lado izquierdo:
PHPMyAdmin creación de una base de datos

En el lado derecho de la pantalla podemos ahora ingresar el nombre de una tabla y la cantidad de campos que tendrá (crearemos una tabla llamada alumnos con 4 campos):
PHPMyAdmin creación de una tabla

La estructura de la tabla es:

codigo int auto_increment primary key
nombre varchar(50)
mail varchar(70)
codigocurso int

En el PHPMyAdmin ingresamos:

PHPMyAdmin creación de una tabla

Es importante también hacer notar que en el campo codigo debemos marcar en Indice el valor "primary" y tildar la columna A_I (Auto_Increment):

PHPMyAdmin creación de una tabla

Por último presionamos el botón guardar y ya tenemos la tabla "alumnos" creada en la base de datos "base1":

PHPMyAdmin creación de una tabla

La tabla almacenará datos de alumnos que desarrollarán cursos de programación en PHP, ASP y JSP.

El código del alumno es de tipo numérico (int) y al indicar que es auto_increment se generará automáticamente por el gestor de base de datos.
Los campos nombre y mail son de tipo varchar (podemos almacenar cualquier caracter) y por último el campo codigocurso representa el curso a tomar por el alumno (1=PHP, 2=ASP y 3=JSP)
El campo clave de esta tabla es el código de alumno (es decir no podemos tener dos alumnos con el mismo código, no así el nombre del alumno que puede eventualmente repetirse)

En los próximos conceptos comenzaremos a ver como desde PHP podemos comunicarnos con la base de datos "base1" y acceder a la tabla "alumnos" para ejecutar los comandos SQL más comunes como pueden ser: select, insert, delete, update etc.

Luego de crear una base de datos y sus tablas (Vamos a trabajar con la base de datos ya creada: base1, que contiene la tabla alumnos), veremos como agregar registros.

Para añadir datos en la tabla empleamos el comando SQL llamado insert.

Necesitamos dos páginas para este proceso, una será el formulario de carga de datos y la siguiente será la que efectúe la inserción en la tabla.

Formulario de carga de datos:

```
 <select name="codigocurso">
     <option value="1">PHP</option>
     <option value="2">ASP</option>
     <option value="3">JSP</option>
   </select>
```

Cada opción tiene su respectivo valor (en este caso los números 1,2 y 3) y los textos a mostrar PHP, ASP y JSP. El dato que se envía a la otra página es el código de curso (esto debido a que definimos la propiedad value).

Ahora veremos como realizar la registración de los datos cargados en el formulario, en la tabla alumnos de la base de datos base1:

pagina2.php

```
<html>
<html>

<head>
  <title>Problema</title>
</head>

<body>
  <?php
  $conexion = mysqli_connect("localhost", "root", "", "base1") or
    die("Problemas con la conexión");

  mysqli_query($conexion, "insert into alumnos(nombre,mail,codigocurso) values
                    ('$_REQUEST[nombre]','$_REQUEST[mail]',$_REQUEST[codigocurso])")
    or die("Problemas en el select" . mysqli_error($conexion));

  mysqli_close($conexion);

  echo "El alumno fue dado de alta.";
  ?>
</body>

</html>
```

Veamos los pasos para efectuar el alta en la tabla alumnos:

```
  $conexion = mysqli_connect("localhost", "root", "", "base1") or
    die("Problemas con la conexión");
```

La función mysqli_connect se conecta a una base de datos de tipo MySql, el primer parámetro es la dirección donde se encuentra el gestor de base de datos (en este caso en el mismo servidor por lo que indicamos esto con "localhost"), el segundo parámetro es el nombre de usuario de la base de datos ("root" en nuestro caso, que es el usuario por defecto que crea MySql para el administrador), seguidamente indicamos la clave del usuario root (por defecto al instalar el XAMPP se crea con clave vacía) y por último indicamos el nombre de la base de datos a conectarnos (en nuestro ejemplo ya creamos la base de datos llamada: base1 que tiene la tabla alumnos)
En caso de haber algún error en la llamada a la función la misma retorna false por lo que se ejecuta la instrucción seguida del operador or, en nuestro caso llamamos a la función die que detiene la ejecución del programa y muestra el mensaje por pantalla.

El paso más importante es la codificación del comando SQL insert(debemos llamar a la función mysqli_query pasando como primer parámetro la referencia a la conexion y el segundo parámetro es un string con el comando insert):

```
  mysqli_query($conexion, "insert into alumnos(nombre,mail,codigocurso) values
                    ('$_REQUEST[nombre]','$_REQUEST[mail]',$_REQUEST[codigocurso])")
    or die("Problemas en el select" . mysqli_error($conexion));
```

La sintaxis del comando insert es bastante sencilla, indicamos el nombre de la tabla y los campos de la tabla a cargar. Luego debemos indicar en el mismo orden los valores a cargar en cada campo (dichos valores los rescatamos del formulario anterior).

Los campos de tipo varchar SQL requiere que encerremos entre comillas simples, esto sucede para el nombre y el mail; en cambio, para el codigocurso esto no debe ser así.

Otra cosa a tener en cuenta es que los subíndices de los vectores asociativos no deben ir entre simples comillas ya que se encuentran dentro de un string, sino se producirá un error.

En caso que MySql detecte un error, retorna false la función mysqli_query, por lo que se ejecuta la instrucción posterior al or, es decir la función die que mostrará el error generado por MySql llamando a la función mysqli_error.

Por último cerramos la conexión con la base de datos y mostramos un mensaje indicando que la carga se efectuó en forma correcta.

Tener en cuenta que el campo código se generó en forma automática.

Si queremos controlar que el insert se efectuó en forma correcta podemos ingresar al PHPMyAdmin y seleccionar la tabla "alumnos", y en la pestaña "examinar" podremos ver los datos ingresados desde la página que creamos:

PHPMyAdmin insert en una tabla
pagina1.html

```
<html>

<head>
  <title>Problema</title>
</head>

<body>
  <h1>Alta de Alumnos</h1>
  <form action="pagina2.php" method="post">
    Ingrese nombre:
    <input type="text" name="nombre"><br>
    Ingrese mail:
    <input type="text" name="mail"><br>
    Seleccione el curso:
    <select name="codigocurso">
```

```
    <option value="1">PHP</option>
    <option value="2">ASP</option>
    <option value="3">JSP</option>
   </select>
   <br>
   <input type="submit" value="Registrar">
  </form>
</body>

</html>
```

El formulario es bastante similar a los que venimos desarrollando en puntos anteriores, tal vez lo distinto es cómo emplearemos el control "select" del curso a desarrollar:

Para recuperar datos desde MySQL o MariaDB debemos emplear el comando select:

```
 select codigo,nombre,mail,codigocurso from alumnos
```

Debemos pasar desde PHP un string con este comando para que MySQL lo ejecute y retorne todas las filas de la tabla alumnos.

Veremos entonces como recuperar los datos almacenados en la tabla alumnos de la base de datos "base1".

El programa que muestra los registros en una página es:
pagina1.php

```
<html>

<head>
 <title>Problema</title>
</head>

<body>

 <?php
 $conexion = mysqli_connect("localhost", "root", "", "base1") or
  die("Problemas con la conexión");

 $registros = mysqli_query($conexion, "select codigo,nombre,mail,codigocurso
                  from alumnos") or
  die("Problemas en el select:" . mysqli_error($conexion));

 while ($reg = mysqli_fetch_array($registros)) {
  echo "Codigo:" . $reg['codigo'] . "<br>";
  echo "Nombre:" . $reg['nombre'] . "<br>";
  echo "Mail:" . $reg['mail'] . "<br>";
  echo "Curso:";
  switch ($reg['codigocurso']) {
   case 1:
```

```
      echo "PHP";
      break;
    case 2:
      echo "ASP";
      break;
    case 3:
      echo "JSP";
      break;
  }
  echo "<br>";
  echo "<hr>";
}

 mysqli_close($conexion);
 ?>

</body>

</html>
```

La primer parte es similar a lo visto hasta ahora, es decir nos conectamos con el servidor de base de datos y seleccionamos la base de datos base1:

```
 $conexion = mysqli_connect("localhost", "root", "", "base1") or
   die("Problemas con la conexión");
```

El comando SQL que nos permite recuperar datos de tablas se llama SELECT. Indicamos los campos a rescatar de la tabla y luego de la palabra clave from indicamos el nombre de la tabla:

```
 $registros = mysqli_query($conexion, "select codigo,nombre,mail,codigocurso
                  from alumnos") or
   die("Problemas en el select:" . mysqli_error($conexion));
```

En caso de haber codificado incorrectamente, el comando SQL select la función mysqli_query retorna false, por lo que se ejecuta el comando siguiente al operador or, es decir la función die.

Si el comando SQL es correcto, en la variable $registros se almacena una referencia a los datos rescatados de la tabla alumnos. Ahora debemos ir mostrando registro a registro los datos extraídos:

```
 while ($reg = mysqli_fetch_array($registros)) {
```

Para rescatar registro a registro los datos obtenidos por el select debemos llamar a la función mysqli_fetch_array. Esta función retorna un vector asociativo con los datos del registro rescatado, o false en caso de no haber más registros. Es decir que si retorna un registro se almacena en el vector $reg y la condición del while se valida como verdadero y pasa a ejecutarse el bloque del while:

```
   echo "Codigo:" . $reg['codigo'] . "<br>";
   echo "Nombre:" . $reg['nombre'] . "<br>";
```

```
   echo "Mail:" . $reg['mail'] . "<br>";
   echo "Curso:";
   switch ($reg['codigocurso']) {
     case 1:
       echo "PHP";
       break;
     case 2:
       echo "ASP";
       break;
     case 3:
       echo "JSP";
       break;
   }
   echo "<br>";
   echo "<hr>";
 }
```

El bloque del while muestra el contenido del registro rescatado por la función mysqli_fetch_array. Como vemos, para rescatar cada campo accedemos mediante el vector asociativo $reg indicando como subíndice un campo contenido en el select:$reg['codigo']

Cada vez que llamamos a la función mysqli_fetch_array nos retorna el siguiente registro.

Cuando debemos mostrar el curso mediante la instrucción switch, analizamos si tiene un 1,2 ó 3 y procedemos a mostrar el nombre del curso.

Para separar cada alumno en la página HTML disponemos el elemento "<hr>"