Formularios POO e introducción a manejo de Bases de Datos.

> La ciencia moderna aún no ha producido un medicamento tranquilizador tan eficaz como lo son unas pocas palabras bondadosas. **Sigmund Freud**

PDO.

La extensión *Objetos de Datos de PHP* (PDO por sus siglás en inglés) define una interfaz ligera para poder acceder a bases de datos en PHP. Cada controlador de bases de datos que implemente la interfaz PDO puede exponer características específicas de la base de datos, como las funciones habituales de la extensión. Obsérvese que no se puede realizar ninguna de las funciones de la bases de datos utilizando la extensión PDO por sí misma; se debe utilizar un controlador de PDO específico de la base de datos para tener acceso a un servidor de bases de datos.

PDO proporciona una capa de abstracción de *acceso a datos*, lo que significa que, independientemente de la base de datos que se esté utilizando, se usan las mismas funciones para realizar consultas y obtener datos. PDO *no* proporciona una abstracción de *bases de datos*; no reescribe SQL ni emula características ausentes. Se debería usar una capa de abstracción totalmente desarrollada si fuera necesaria tal capacidad.

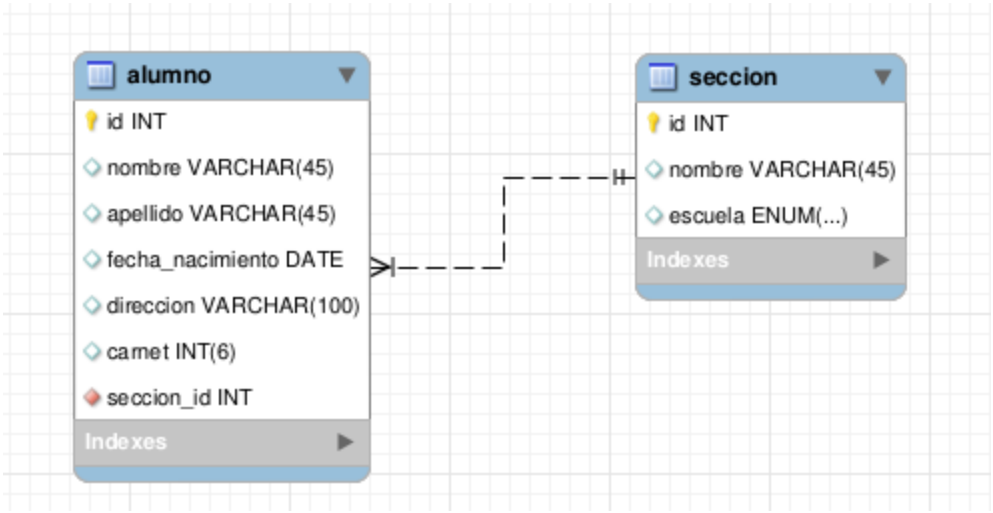
PDO viene con PHP 5.1, y está disponible como una extensión PECL para PHP 5.0; PDO requiere las características nuevas de OO del núcleo de PHP 5, por lo que no se ejecutará con versiones anteriores de PHP.

más información

http://php.net/manual/es/pdo.connections.php

Práctica Desarrollo de app-- Continuación

Vamos a desarrollar una aplicación que permita el poder almacenar en una base de datos la información básica de un alumno tomaremos como base el siguiente diagrama.



Actividad 1.

Modificar archivo de Coneccion

pasos.

1. En el archivo de coneccion que tenemos cambiaremos la coneccion de clase por medio del uso de un objeto de coneccion PDO asi.

```php
$dsn = 'mysql:dbname=alumnos;host=127.0.0.1';
$usuario = 'root';
$clave = '';
    try {
        $con = new PDO($dsn,$usuario, $clave);
    } catch (PDOException $e) {
        print $e->getTraceAsString();
    }
```

2. creamos una función para el manejo de datos

```php
function consultaA($coneccion, $sql){

}
```

3. Dentro de la función agregaremos las siguientes variables que nos servirán para el control de las salidas y devolución de datos.

```php
$ejecutor = $coneccion;
$msgok = NULL;
$msgerror = NULL;
```

4. Insertaremos un bloque try catch que nos servirá para el manejo de excepciones.

```php
try {


  } catch (Exception $exc) {

  }
```

5. En el bloque try agregaremos una variable para identificar que instruccion de sql de accion vamos a utilizar

```php
$condicion = strstr(trim($sql)," ", TRUE);
```

6. Agregamos una condicional switch para definir los mensajes que enviamos con relación al valor devuelto por la variable $condicion para cada sentencia sql de acción.

```php
switch ($condicion)
{
    case "INSERT":
        $msgerror = "No se ha Podido Insertar los Datos";
        $msgok = "Datos Insertados";

        break;
    case "DELETE":
        $msgerror = "No se ha Podido Eliminar los Datos";
        $msgok = "Datos Eliminados";
        break;
    case "UPDATE":
        $msgerror = "No se ha Podido Actualizar los Datos";
        $msgok = "Datos Actualizados";
        break;
    default:
        $msgerror = "Es posible que no use un estandar de consulta SQL";
        break;
}
```

.

7. Con los elemento de PDO empezamos la transacción, ejecutamos la sentencia y realizamos el commit para su ejecución completa en la base de datos..

```php
$ejecutor->beginTransaction();
$fila = $ejecutor->exec($sql);
$ejecutor->commit();
```

8. Agregamos instrucciones para saber cuántas filas fueron afectadas por la acción y si no se afectaron definimos un mensaje de error alertando que no han habido filas afectadas por la ejecución..

```
if($fila == 0){
    $msgok = $msgerror."<br> No existe coincidencia para realizar la accion sobre los
}
```

9. Retornamos el valor del mensaje y numero de filas afectadas.

```
return $msgok. " Filas Afectadas ".$fila ;
```

10. En el bloque cacht agregamos el siguiente código para controlar un mensaje de error por si no se pudo ejecutar las acciones de forma correcta.

```
$ejecutor->rollBack();
return $msgerror. ":(<br>".$exc->getMessage();
```

11. Crear una funcion para manejar las consultas de datos.

```
function consultaD($coneccion, $sql,$modo=2,$presentacion=FALSE)
{


}
```

12. Crear variables para el control de mensajes y salida de datos.

```
$ejecutor = $coneccion;
$manejador = trim($sql);
$devolucion = "";
```

13. Agregar un bloque try/catch

```
try {

} catch (Exception $exc) {

}
```

14. en el bloque try agregar el código que ejecuta la sentencia sql y la asocia a un arreglo para su manipulación.

```
$datos = $ejecutor->query($manejador);
$datos->setFetchMode($modo);
```

15. Agregar una condicional if/else para comprobar el parámetro presentación.

```
if($presentacion == TRUE){

}else{

}
```

16. En el bloque if se creará una estructura para la presentación de los datos en una tabla de html para ello agregar la siguiente definición.

```
$devolucion .="<table border=1>";
```

17. Agregamos una estructura foreach para recorrer los elementos que devolvera la instruccion fetchAll.

```
foreach ($datos->fetchAll() as $registros) {

}
```

18. Agregamos las siguientes instrucciones para definir las filas de la tabla

```
$devolucion.="<tr>";

$devolucion.="</tr>";
```

19. Entre el código que agregamos definiremos un segundo foreach que se encarga de sacar los datos para cada celda de la tabla.

```
foreach ($registros as $valor) {
    $devolucion.="<td>".$valor."</td>";
}
```

20. Fuera de los dos bloques foreach que acabamos de definir agregamos la instrucción que cierra la tabla.

```
$devolucion .="</table>";
```

21. Ahora en el bloque else agregamos el código siguiente.

```
$contenedor = $datos->fetchAll();
$devolucion=$contenedor;
```

22. En el bloque catch agregamos la siguiente sentencia de código.

```
return  "No se pudieron Consultar los Datos<br />".$exc->getMessage();
```

23. Fuera del bloque catch agregamos la siguiente linea de código.

```
return $devolucion;
```

24. Guardamos el archivo.

Actividad 2.

Modificar el Archivo AlumnoControlador.php.

Pasos.

1. Modificamos el archivo agregando el código para que use la función consultaA y que se almacene el campo sección en la tabla alumno. Verificar que codigo tiene la siguientes instrucciones.

```
class AlumnoControlador extends Alumno{
    public function guardarDatos($con,$objAlumno) {
        $variableSql = "INSERT INTO alumno ";
        $variableSql .= "(id,nombre,apellido,fecha_nacimiento,";
        $variableSql .= "direccion,carnet,seccion) ";
        $variableSql .= "VALUES (";
        $variableSql.="'".$objAlumno[0]."',";
        $variableSql.="'".$objAlumno[1]."',";
        $variableSql.="'".$objAlumno[2]."',";
        $variableSql.="'".$objAlumno[3]."',";
        $variableSql.="'".$objAlumno[4]."',";
        $variableSql.="'".$objAlumno[5]."',";
        $variableSql.="'".$objAlumno[6]."');";
        return consultaA($con, $variableSql);

        }
}
```

2. Guardar Archivo.

Actividad 3.

Editar archivo manejadorAlumno.php

pasos.

1. Modificaremos el código del archivo manejador alumnos

```php
$alumnoA = new AlumnoControlador();
$accion= $_REQUEST['accion'];
switch ($accion) {
    case 'save':
        if(isset($_REQUEST['bot'])){
           $alumnoA->setId('NULL');
           $alumnoA->setNombre($_REQUEST['nombre']);
           $alumnoA->setApellido($_REQUEST['apellido']);
           $alumnoA->setCarnet($_REQUEST['carnet']);
           $alumnoA->setDir($_REQUEST['dir']);
           $alumnoA->setFechaNac($_REQUEST['fecha_nac']);
           $alumnoA->setSeccion($_REQUEST['seccion']);
           $datosObj=array($alumnoA->getId(),
                           $alumnoA->getNombre(),
                           $alumnoA->getApellido(),
                           $alumnoA->getFechaNac(),
                           $alumnoA->getDir(),
                           $alumnoA->getCarnet(),
                           $alumnoA->getSeccion());
           print $alumnoA->guardarDatos($con,$datosObj);
           print '<a href=\'manejadorAlumno.php?accion=con\'>Ver datos</a>';
       }else{
           print 'No se ha enviado datos ';
       }
        break;
    case 'con':
        $sql = 'SELECT * FROM alumno';
        print consultaD($con, $sql, 2, TRUE);
        break;
    case 'del':
        $sql = 'DELETE from alumno WHERE id='.$_REQUEST['id'].';';
        print consultaA($con, $sql);
        break;

    default:
        print 'No hay Accion que realizar';
        break;
}
```

2. Guardamos el archivo.

Actividad 4.

Modificar Archivo formulario.php.

pasos.

1. Primero cambiaremos el nombre del formulario a “formularioAlumno.php”.

2. Antes de la etiqueta <html> agregaremos el código siguiente.

```php
<?php include './clases/Coneccion.php';?>
```

3. Modificamos el atributo action del formulario nos quedara de la sugiente manera.

```
<form action="manejadorAlumno.php?accion=save" method="post">
```

4. Buscamos el el fragmento de código html donde se declara el control lista, en el cual se listan las secciones que se asignan a los alumnos, sustituimos el fragmento de html agregando el siguiente código.

```
<tr>
    <td>
        Sección:
    </td>
    <td>
        <select name='seccion'>
            <option value=""></option>
            <?php

        $sql = "SELECT id,nombre FROM seccion;";
        $datos = consultaD($con, $sql);
        foreach ($datos as $value) {
            print "<option value='";
            print $value['id'];
            print "'>";
            print $value['nombre'];
            print "</option>";
        }
    ?>
        </select>
    </td>
</tr>
```

5. Guardamos el archivo.

Actividad 6.

Crear la clase para el modelo de datos de la sección.

pasos.

1. Crear en la carpeta clases el archivo ‘Seccion.php’.

```php
<?php
class Seccion {
    private $id;
    private $nombre;
    private $escuela;

    public function getId() {
        return $this->id;
    }

    public function getNombre() {
        return $this->nombre;
    }

    public function getEscuela() {
        return $this->escuela;
    }

    public function setId($id) {
        $this->id = $id;
    }

    public function setNombre($nombre) {
        $this->nombre = $nombre;
    }

    public function setEscuela($escuela) {
        $this->escuela = $escuela;
    }


}
```

2. Guardar el archivo.

Actividad 7.

Crear en la clase para controlar acciones para la seccion.

pasos.

1. Crear en la carpeta clases archivo “SeccionControlador.php”.

```php
<?php

class SeccionControlador extends Seccion{
    public function guardarDatos($con,$objSeccion) {
        $variableSql = "INSERT INTO seccion ";
        $variableSql .= "(id,nombre,escuela) ";
        $variableSql .= "VALUES (";
        $variableSql.="'".$objSeccion[0]."',";
        $variableSql.="'".$objSeccion[1]."',";
        $variableSql.="'".$objSeccion[2]."');";
      return consultaA($con, $variableSql);
    }

}
```

2. guardar el archivo.

Actividad 8.

Crear archivo para el control de las acciones para los datos de la tabla sección.

pasos.

1. Crear en la carpeta raiz del proyecto el archivo manejadorSeccion.php y agregar los siguiente.

```php
<?php
include './clases/Coneccion.php';
include './clases/Seccion.php';
include './clases/SeccionControlador.php';
$seccionA = new SeccionControlador();
$accion= $_REQUEST['accion'];
switch ($accion) {
    case 'save':
        if(isset($_REQUEST['bot'])){
            $seccionA->setId('NULL');
            $seccionA->setNombre($_REQUEST['nombre']);
            $seccionA->setEscuela($_REQUEST['escuela']);

            $datosObj=array($seccionA->getId(),
                            $seccionA->getNombre(),
                            $seccionA->getEscuela()
                            );

            print $seccionA->guardarDatos($con,$datosObj);
            print '<a href=\'manejadorSeccion.php?accion=con\'>Ver datos</a>';
        }else{
            print 'No se ha enviado datos ';
        }
         break;
    case 'con':
        $sql = 'SELECT * FROM seccion';
        print consultaD($con, $sql, 2, TRUE);
        break;
    case 'del':
        $sql = 'DELETE from seccion WHERE id='.$_REQUEST['id'].';';
        print consultaA($con, $sql);
        break;

    default:
        print 'No hay Accion que realizar';
        break;
}
```

2. Guardar el archivo.

Actividad 9.

Crear en la formulario para capturar datos de las secciones.

pasos.

1. Crear en la carpeta raiz el archivo ‘formularioSeccion.php’.

```
<html>
    <head>
        <meta charset="utf-8">
        <title>Formulario de Captura de datos</title>
    </head>
    <body>
        <form action="manejadorSeccion.php?accion=save" method="post">
             <table>
            <tr>
                <td>
                    Nombre:
                </td>
                <td>
                    <input type="text" name="nombre">
                </td>
            </tr>
            <tr>
                <td>
                    Escuela:
                </td>
                <td>
                    <select name='escuela'>
                        <option value=""></option>
                        <option value="sistemas">Sistemas</option>
                        <option value="manto">Mantenimiento</option>
                    </select>
                </td>
            </tr>
            <tr>
                <td colspan="2">
                    <input type="submit" name='bot' value='Enviar'>
                </td>
            </tr>
        </table>
        </form>

    </body>
</html>
```

2. Guardar el archivo

Actividad 10.

Crear la base de datos completa con las tablas según esquema y relaciones.

pasos.

Crear la base de datos utilizando la herramienta que más le convenga para el proceso tome de base el siguiente código.

```
CREATE DATABASE alumnos;
USE alumnos;

CREATE TABLE alumno (
  id int(11) NOT NULL AUTO_INCREMENT,
  nombre varchar(45) NOT NULL,
  apellido varchar(45) NOT NULL,
  fecha_nacimiento date NOT NULL,
  direccion varchar(100) NOT NULL,
  carnet int(6) NOT NULL,
  seccion int(11) NOT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY carnet (carnet),
  KEY seccion (seccion)
)

CREATE TABLE  seccion (
  id int(11) NOT NULL AUTO_INCREMENT,
  nombre varchar(25) NOT NULL,
  escuela enum('sistemas','manto') NOT NULL,
  PRIMARY KEY (id)
)
```