UNIVERSIDAD TECNICA DE BABAHOYO

FACULTAD DE ADMINISTRACIÓN, FINANZAS E INFORMÁTICA

ESCUELA DE SISTEMAS

ESPECIALIZACION INGENIERIA EN SISTEMAS

"DESARROLLO DE UN SISTEMA INFORMATICO PARA EL CONTROL DE TESORERIA PARA LA ESCUELA ALFA & OMEGA DE LA CIUDAD DE BABAHOYO"

TESIS DE GRADO

Previa a la obtención del Título de:

ANALISTA DE SISTEMAS

INTEGRANTE:

KARINA GEOMARY BASILIO SILVA

DIRECTOR: ING. ANGEL RAFAEL ESPAÑA LEÓN

LECTOR: ING. MARIA ISABEL GONZÁLEZ VALERO

BABAHOYO-LOS RIOS-ECUADOR

2012

TEMA

“DESARROLLO DE UN SISTEMA INFORMATICO PARA EL CONTROL DE TESORERIA PARA LA ESCUELA ALFA & OMEGA DE LA CIUDAD DE BABAHOYO”

AGRADECIMIENTO

A Jehová Dios por ser el guiador principal de mi vida, otorgándome salud, fortaleza y sabiduría, para poder desarrollar con desempeño y dedicación el presente trabajo.

A mí Familia que fueron mi principal motor en esta etapa, llenándome cada día de optimismo, y en momentos difíciles jamás dejaron de creer que tenía la certeza de culminar con éxito mi carrera universitaria.

Y a aquellas personas que de una u otra manera me ayudaron a seguir adelante.

Karina Geomary Basilio Silva

DEDICATORIA

A Jehová Dios por ser el protector de mi vida y darme la sabiduría y serenidad necesaria para enfrentar cualquier adversidad de la vida.

A mí familia por el apoyo incondicional que no se fijaron en mis tropiezos sino en cada logro obtenido.

A mis Maestros un infinito agradecimiento por todas las enseñanzas que me brindaron para si mejorar mis conocimientos.

Karina Geomary Basilio Silva

INDICE

CAPITULO I

CAPITULO II

CAPITULO III

CAPITULO IV

CAPITULO V

CAPITULO I

1. EL PROBLEMA

1.1 PLANTEAMIENTO DEL PROBLEMA

Es un rechazo a los inconvenientes encontrados en el sistema manual, en el cual la información para el control de tesorería no es exacta, esto representa un problema al no estar actos estos informes como información verídica.

El jefe de tesorería de la escuela debe de tener información constante y clara para poder administrar de forma adecuada los bienes a su cargo, y esto no sucede principalmente porque la información no es actualizada a tiempo, también porque estos datos están archivados en libros y en hojas sueltas, muchas de las veces los datos requeridos es muy variado y eso hace que se torne difícil encontrar informes específicos.

El control se hace difícil ya que poseer datos de bienes de todo tipo, archivarlos se vuelve complicado para el administrador y más al momento de buscar información de los mismos.

La responsabilidad de un apropiado control interno de los recursos, radica en una buena administración, el sistema manual implementado allí no le permite a la institución mantener un adecuado manejo y control de los bienes, así como también no conocer la totalidad actual.

La falta de recursos en la actualidad también se ha vuelto un problema a la hora de administrar los bienes ya que el personal con el que se cuenta no es suficiente ni está preparado de la mejor manera para desarrollar esta función.

1.2. FORMULACIÓN DEL PROBLEMA

¿Cómo se podría realizar de manera más eficiente el control de tesorería de la Escuela Alfa & Omega?

1.3. DELIMITACIÓN DEL PROBLEMA

Esta investigación se realizara en la Escuela Alfa & Omega en la ciudad de Babahoyo, ubicado en las calles Rocafuerte y Juan x marcos, para el periodo 2012-2013.

Campo	Educativo
Área	Financiera
Aspecto	Control de Tesorería

1.4. OBJETIVOS

1.4.1 Objetivos General

Desarrollar un sistema informático para el control de Tesorería de la Escuela Alfa & Omega de la ciudad de Babahoyo.

1.4.2 Objetivos Específicos

- Sustentar las bases teóricas en cuanto al manejo de tesorería.
- Analizar la información más importante que ayude al establecimiento de una base de datos de acorde a la necesidad de tesorería.
- Implementar un sistema informático que reemplace el sistema manual en el control de tesorería.
- Implementar una Base de Datos.

1.5. JUSTIFICACIÓN

El siguiente proyecto es necesario para mejorar en forma ordenada y segura el control de tesorería de la Escuela Alfa & Omega ya que al implementarse el sistema podrá mejorar varios aspectos como por ejemplo: llevar un control de cuánto dinero se recauda a diario, mensual, etc.

Este sistema permitirá realizar de una manera rápida la administración del dinero logrando la satisfacción de la directiva, alumnos, padres de familia de este lugar. permitiendo así llevar un mejor control de cada uno de los procesos a realizarse en el sistema y de forma segura.

El sistema que se va a realizar permitirá realizar ingresos de nuevos contribuyentes, consultas, eliminación, auditoria, registros de usuario, reportes, ingresos, egresos.

Además el proyecto se realizara utilizado los lenguajes de programación php y como base de datos se utilizara MySql ya que estos no necesitan licencia por ser software libre y además se aplicara a una institución privada.

El proyecto también tratara de impulsar el uso de programas si licencia utilizado todo recurso que os permita realizar un excelente sistema sin necesidad de realizar grades gastos en licencias ya que estas son caras y da el mismo resultado.

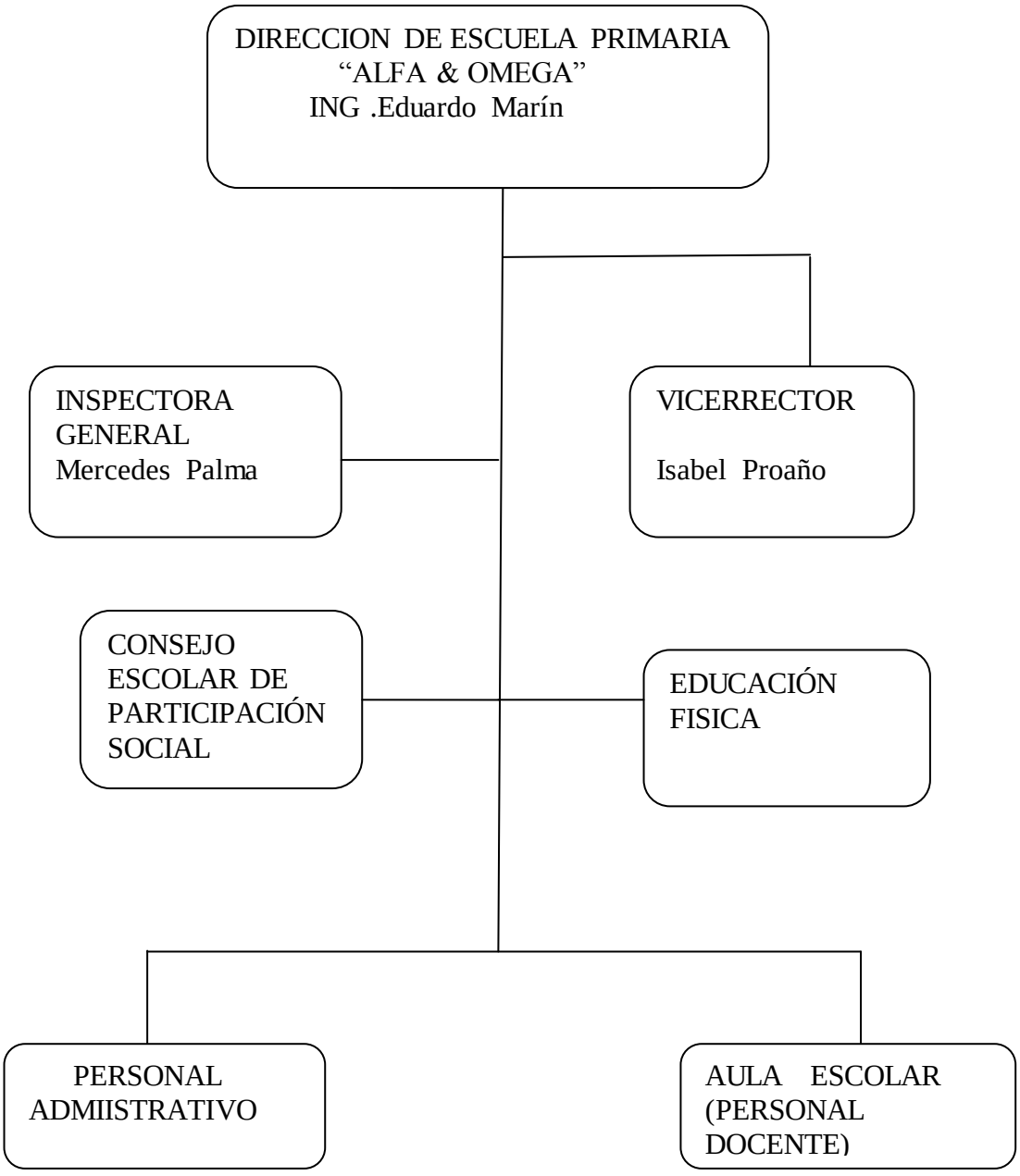
CAPITULO II

2. MARCO TEORICO

2.1 Escuela Alfa & Omega de la ciudad de Babahoyo..

2.1.1 Organización del Escuela Alfa & Omega.

Organigrama general



2.1.2 MISION Y VISION

MISION

Ofrecer u servicio educativo que asegure a los alumnos una educación suficiente y de calidad que contribuya como factor estratégico de justicia social, que los forme como sujetos competentes en donde se favorezca el desarrollo de sus habilidades para acceder a mejores condiciones de vida, aprendan a vivir en forma solidaria y democrática y sean capaces de transformar su entorno.

VISION.

Alcanzar niveles de excelencia, conjuntando con responsabilidad los esfuerzos de autoridades y sociedad para brindar un servicio eficiente y eficaz que satisfaga plenamente las necesidades y expectativas de los educandos, logrando su desarrollo amónico e integral.

2.1.3 Ubicación:

Rocafuerte 521 entre Juan x Marcos y García Moreno

2.1.4 Personal

DIRECTOR:	Ing. Eduardo Marín
VICERRECTORA:	Lcda. Isabel Proaño
INSPECTORA GENERAL:	Lcda. Mercedes Palma
PERSONAL ADMIISTRATIVO:	Sra. mercedes Arreaga
	Sra. Diana Chocho
PERSONAL DOCENTE:	Lcda. Cindi Montenegro
	Lcda. Katty Roldan
	Ing. Karla Huacon
	Lcda. Luisa García

Lcda. Tatiana Fiallos

Lcdo. Marlon Montoya

Lcdo. Héctor Chic

2.1.5 Gestión Académica

E. Prebásica

Matemáticas

Lenguaje

Ingles

Educación Artística

- dibujo
- música
- manualidades

E. Básica

Matemáticas
Ciencias Naturales
Educación Cívica
Educación Artística
Educación Física
Lengua y literatura
Entorno natural y social

Inglés

Estudios Sociales

Actividades Prácticas

Modalidad Horas pedagógicas propuestas (45 min. c/u)

2.2 PHP

2.2.1 PHP Introducción.

PHP es un lenguaje de scripting que permite la generación dinámica de contenidos en un servidor web. El significado de sus siglas es HyperText Preprocessor. Entre sus principales características cabe destacar su potencia, su alto rendimiento, su facilidad de aprendizaje y su escasez de consumo de recursos.

El código PHP puede incluirse dentro del código html de la pagina. Para delimitar la seccion de codigo PHP podemos hacerlo de varias formas:

-Usando las etiquetas <?php y <?
-Usando las etiquetas <? y ?>
-Mediante <script languaje="php"> </script>

El funcionamiento de las páginas en PHP alojadas en un servidor es el siguiente:
-El navegador del cliente solicita el documento PHP.
-Llega la solicitud del servidor y el servidor localiza el documento, lanza el intérprete de PHP y ejecuta todo su código.
-Una vez ejecutado el código se genera el resultado en HTML y lo devuelve al servidor para que lo transfiera al cliente.
-El servidor transfiere el resultado en HTML y es mostrado en el navegador del cliente.

2.2.2 ¿Qué es PHP?

PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es usado principalmente en interpretación del lado del servidor (server-side scripting) pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas Qt o GTK+.

2.2.3 Característica

PHP en cada nueva versión soporta nuevas funcionalidades por lo que el mejor método para ver que nuevas librerías incluye es cuando compilamos. Si dentro del directorio con las fuentes de PHP ejecutamos el comando "configure --help" nos dará información de todas las posibles opciones que tiene PHP al ser compilado y, entre esta información, la de todos los módulos que podemos añadir a PHP si disponemos de las librerías adecuadas. Dentro del manual de PHP disponemos también de una referencia a todas las funciones disponibles. Ojo que muchas veces es necesario compilar PHP de forma especial para incluir soporte para una determinada funcionalidad. Por ejemplo, si queremos acceder desde PHP al gestor de base de datos MySQL, es necesario que a la hora de compilar PHP incluyamos ests soporte. Para ello es necesario tener las librerías de MySQL instaladas en el sistema, normalmente en el paquete "dev" de MySQL, y ejecutar "./configure --with-mysql" antes de compilar PHP. De esta forma PHP incluirá en el módulo generado dicho soporte. En la información que obtenemos con la llamada a la función "phpInfo()" se incluye información sobre los módulos incluidos. En la figura 3 se puede observar la información referente al módulo MySQL y al GD, que se utiliza para la creación de gráficos "al vuelo".

2.2.4 Ventajas y Desventajas

Ventajas

- Es un lenguaje multiplataforma.
- Completamente orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una Base de Datos.
- El código fuente escrito en PHP es invisible al navegador y al cliente ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable.
- Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.

- Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones).
- Posee una amplia documentación en su página oficial ([4]), entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite aplicar técnicas de programación orientada a objetos.
- Biblioteca nativa de funciones sumamente amplia e incluida.
- No requiere definición de tipos de variables aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.
- Tiene manejo de excepciones (desde PHP5).
- Si bien PHP no obliga a quien lo usa a seguir una determinada metodología a la hora de programar (muchos otros lenguajes tampoco lo hacen), aun estando dirigido a alguna en particular, el programador puede aplicar en su trabajo cualquier técnica de programación y/o desarrollo que le permita escribir código ordenado, estructurado y manejable. Un ejemplo de esto son los desarrollos que en PHP se han hecho del patrón de diseño Modelo Vista Controlador (o MVC), que permiten separar el tratamiento y acceso a los datos, la lógica de control y la interfaz de usuario en tres componentes independientes (ver más abajo Frameworks en PHP).

Desventajas

- Como es un lenguaje que se interpreta en ejecución para ciertos usos puede resultar un inconveniente que el código fuente no pueda ser ocultado. La ofuscación es una técnica que puede dificultar la lectura del código pero no la impide y aparte en ciertos casos representa un costo en tiempos de ejecución.

2.2.5 Formularios

Los Formularios no forman parte de PHP, sino del lenguaje estándar de Internet, HTML. Vamos a dedicar en este capítulo algunas líneas al HTML, para entrar posteriormente a tratarlos con PHP.

Todo formulario comienza con la etiqueta <FORM ACTION="lo_que_sea.php" METHOD="post/get"> . Con . Con ACTION indicamos el script que va procesar la información que recogemos en el formulario, mientras que METHOD nos indica si el usuario del formulario va ha enviar datos (post) o recogerlos (get) La etiqueta <FORM> indica el final del formulario.

A partir de la etiqueta <FORM> vienen los campos de entrada de datos que pueden ser:

Cuadro de texto:
<input type="text" name="nombre" size="20" value="jose">
Cuadro de texto con barras de desplazamiento:
<textarea rows="5" name="descripcion" cols="20">Es de color rojo</textarea>

Casilla de verificación:
<input type="checkbox" name="cambiar" value="ON">

Botón de opción:
<input type="radio" value="azul" checked name="color">

Menú desplegable:
<select size="1" name="dia">
<option selected value="lunes">lunes</option>
<option>martes</option>
<option value="miercoles">miércoles</option>
</select>

Boton de comando:
<input type="submit" value="enviar" name="enviar">

Campo oculto:

<input type="hidden" name="edad" value="55">

2.2.6 Métodos “ Get” y “ Post”

Métodos de envió de datos GET y POST

Son los métodos de los cuales los datos de un formulario son enviados, el método es indicado en el atributo method y los dos métodos posibles son GET y POST. La diferencia entre estos dos métodos radica en la forma de enviar los datos a la página, mientras que el método GET envía los datos usando la URL, el método POST los envía por la entrada estándar STDIO, en otras palabras se pasa los datos codificados en el flujo de datos HTTP y los datos no son visibles para los usuarios.

Tipos de contenido enviado por un formulario

En los tipos de contenidos posibles:

* multipart/form-data: cuando se envían archivos.
* text/plain: solo texto legible.

2.2.7 Aplicaciones desarrolladas en PHP

- OsCommerce
- PrestaShop
- WordPress
- Burning Board
- CMSformE
- Dokuwiki
- Drupal
- Gallery Project
- Mambo Open Source
- MediaWiki (desarrollado para Wikipedia)
- Moodle
- Phorum

- phpMyAdmin
- PHP-Nuke
- phpPgAdmin
- PhpWiki
- PmWiki
- Zikula (anteriormente llamado PostNuke)
- Smarty
- SPIP
- SugarCRM
- vBulletin
- Xaraya
- Xoops
- Joomla
- MODx
- SMF
- phpBB
- UVG SCADA
- PhpCollab
- Facebook

2.2.7.1Comprimir página PHP

Para aligerar el tiempo de carga de nuestras páginas generadas con PHP, podemos enviarlas al navegador comprimidas con GZip
Para aligerar el tiempo de carga de nuestras páginas generadas con PHP, podemos enviarlas al navegador comprimidas con GZip utilizando las funciones de control de salida, para ello, llamaremos a la funcion predefinida ob_gzhandler cómo tratante de la funcion ob_start, veamos un ejemplo:

```
<?
ob_start("ob_gzhandler");

// Contenido de la página, puede contener
// tanto HTML cómo PHP

ob_end_flush();
?>
```

Tener en cuenta que todo el contenido debe estar en el lugar indicado por lo que los primeros carácteres del documento deben ser <? y los dos últimos ?> y no se debe añadir nada excepto donde se indica, si no vamos con cuidado recibiremos un error parecido al siguiente:

Warning: Cannot add header information...

Otra forma más completa todavía de compresion, consiste en aplicar la misma funcion, pero eliminando a su vez los espacios y saltos de linea de la fuente del documento, lo que no tendra ningun efecto visual pero disminuira el tiempo de descarga, veamos cómo hacerlo:

```
<?
ob_start();

// Contenido de la página, puede contener
// tanto HTML cómo PHP

$cntACmp =ob_get_contents();
ob_end_clean();
$cntACmp=str_replace("\n",' ',$cntACmp);
$cntACmp=ereg_replace('[[:space:]]+',' ',$cntACmp);
ob_start("ob_gzhandler");
echo $cntACmp;
ob_end_flush();
```

2.2.7.2 Contador simple para páginas PHP

Creamos un contador, programado en PHP, que lleva la cuenta de lasimpresiones que se han realizado en una página web, utilizando un archivo de texto como apoyo.

Hice una modificación al Script publicado en el artículo Escritura de archivos con PHP, en el que se enseña a escribir archivos de texto mediante PHP, tocando los temas de lectura y escritura. El objetivo es llevar un conteo de las veces que se ha visitado una página.

Puse el siguiente script PHP al final de la página, se entenderá bien si se lee el articulo señalado antes.

```
<?
$archivo = "contador.txt";
$contador = 0;

$fp = fopen($archivo,"r");
$contador = fgets($fp, 26);
```

```
fclose($fp);

++$contador;

$fp=fopen($archivo,"w+");
fwrite($fp,$contador,26);
fclose($fp);

echo "Esta página ha sido visitada $contador veces";
?>
```

Además, creé un archivo llamado "contador.txt" que lo guardé en el mismo directorio que la página. Dicho

archivo fue inicializado con un cero (0) como único texto.

2.2.7.3 Editar y Borrar datos con MySQL y PHP

Editar

La edición de datos en mysql, combina opciones de Insertar datos a MySQL y de Consultas MySQL , tendremos que hacer una consulta cómo la siguiente:

UPDATE tabla SET campo = 'valor' WHERE condicion

Como veis, volvemos a utilizar la clausula **WHERE** para escojer las entradas que hay que editar, podemos actualizar varios campos de la siguiente manera:

UPDATE tabla SET campo = 'valor', campo2 = 'valor2' WHERE condicion

El metodo no tiene mas secretos que esto, veamos un ejemplo real para ver cómo funciona exactamente desde PHP:

```
<?php
$sql = "UPDATE agenda SET telefono = 555405181 WHERE nombre = 'eloi'" ;
mysql_query ( $sql , $db );
?>
```

Recordar que $db contiene un identificador de la conexión.

Borrar

Borrar unas determinadas de una tabla de MySQL es incluso más sencillo que editarlas, pues solo es nescesario indicar que entradas

queremos borrar con una clausula **WHERE** y en que tabla lo haremos, y esto junto con la palabra **DELETE FROM**nos darán el resultado que esperamos:

DELETE FROM tabla WHERE condicion

No creo que sea nescesario poner el ejemplo de este caso, si ahún asi salen dudas consultar en el foro .

Autor: Eloi de San Martín

2.2.7.4 Mostrar la fecha en Español con PHP

imprimimos con normalidad la fecha en nuestra página, mediante PHP, seguramente esta aparecerá en inglés. Si ponemos algo así:

<?php echo strftime("%A %d de %B del %Y"); ?>

Aparecerá por ejemplo "Thursday 17 de February del 2005", cosa que noquedará muy acorde con nuestra página, si esta está en idioma Español.

La mejor solución es configurar las locales y PHP hará el resto por nosotros. Las locales son traducciones de cosas básicas, como la fecha, que suelen venir en el sistema operativo. Veamos como configurar las locales para el idioma Español:

set_locale(LC_ALL,"es_ES@euro","es_ES","esp");

Ponemos varias opciones por si la primera no está disponible, saltará a la siguiente y así sucesivamente. Si ahora probamos el mismo código:

```
<?php

set_locale(LC_ALL,"es_ES@euro","es_ES","esp");
echo strftime("%A %d de %B del %Y");

?>
```

Aparecerá "Jueves 17 de Febrero del 2005".

Si no disponemos de locales, podemos hacer la traducción nosotros mismos. Por ejemplo, para el día de la semana haríamos algo así:

```
<?php
```

```
$dias =

array("Domingo","Lunes","Martes","Miercoles","Jueves","Viernes","Sábad
o");
echo "Hoy es ".$dias[date('w')];

?>
```

date('w') devuelve el día de la semana, entre 0 y 6. 0 es el domingo y 6 el sábado. Para el mes podemos hacer algo parecido, sabiendo que date('n') devuelve el mes, entre 1 y 12. Hay que tener en cuenta que en este caso no empieza desde 0.

2.2.7.5 Operadores en PHP

Al desarrollar cualquier programa empleamos normalmente operadores que nos sirven para realizar diversas operaciones que le otorgan un cierto grado de complejidad a nuestros programas, ya que, de otro modo el programa realizaría siempre lo mismo y por tanto no seria un programa útil.

Operadores aritméticos

+ Suma dos valores
- Resta dos valores (o pasa a negativo un valor)
***** Multiplica dos valores
/ Divide dos valores
% Resto de dividir dos valores
++ Incremento en una unidad
-- Decremento en una unidad

Operadores de asignación

= Asigna a la parte derecha el valor izquierdo
+= Realiza la suma de la derecha con la izquierda y la asigna a la derecha
-= Realiza la resta de la derecha con la izquierda y la asigna a la derecha
***=** Realiza la multiplicación de la derecha con la izquierda y la asigna a la derecha
/= Realiza la división de la derecha con la izquierda y la asigna a la derecha
%= Se obtiene el resto y se asigna
.= Concatena el valor de la izquierda con la derecha y lo asigna a la derecha

Operadores lógicos

! Operador NO o negacion.Si era true pasa a false y viceversa

and Operador Y, si ambos son verdaderos vale verdadero

or Operador O, vale verdadero si alguno de los dos es verdadero

xor Verdadero si alguno de los dos es true pero nunca ambos

&& True si ambos lo son

|| True si alguno lo es

Operadores condicionales

== Comprueba si dos números son iguales

!= Comprueba si dos números son distintos

> Mayor que, devuelve true en caso afirmativo

< Menor que, devuelve true en caso afirmativo

>= Mayor o igual

<= Menor o igua

2.2.7.6 Matrices (array)

En la realización de un script en PHP en múltiples ocasiones existen variables que tienen información similar y se procesan de forma semejante. Para ello PHP (y otros lenguajes) poseen un elemento denominado array. Un array es un conjunto de variables agrupadas bajo un único nombre. Cada variable dentro de la matriz se denomina elemento. Dentro de la misma matriz pueden existir variables de diferentes tipos y no es necesario que sean todas del mismo tipo.

Hay que diferenciar entre los dos tipos de matrices existentes:
- **Indexada**: Aquella cuyo acceso a los elementos se realiza por la posición que ocupan dentro de la estructura (se inician siempre desde la posición 0). Ejemplo: $amigos[0]
- **Asociativa**: Es aquella en la que los elementos están formados por pares clave-valor y el acceso se realiza proporcionando una determinada clave. Ejemplo: $amigos['edad']

Para crear matrices en PHP existen dos formas:

- **De forma implícita**, que consistiría en indicarle el elemento(ya sea proporcionando su posición o su clave).Ejemplo: $nombres[0]='Javier';

En caso de no indicarle una posición el array tomara el valor siguiente al ultimo valor introducido.Ejemplo: $nombres[]='Lucas' // tomaría como valor 1 ya que lo ultimo introducido era 0.

- **Mediante array()** en el cual le pasamos los elementos como parámetros. En caso de matriz indexada toman la posición que ocupan en la creación de la matriz, mientras que los de la matriz asociativa se les asigna su valor mediante "=>".Ejemplo: $amigo=array('Nombre'=>'Jose','Direccion'=>'Neopatria 21');

Cabe destacar que PHP no solo se limita a la existencia de matrices por sí solo sino que existen matrices de matrices, o lo que es lo mismo, matrices multidimensionales. Ejemplo: $amigos[2]['Pedro']

Recorrido de una matriz

Disponemos de diversas herramientas para poder acceder a los elementos de una matriz. En cada momento se mantiene una referencia del elemento de la matriz al que se tiene acceso, por tanto, para recorrer una matriz bastará con modificar dicha referencia. En caso de una matriz indexada el recorrido se realizara mediante un bucle y para ello debemos saber el número de elementos totales que posee la matriz. Para ello nos basamos de la función **count(variable)** donde variable representa la variable de la que se quiere obtener el número de elementos. Si**variable** es una matriz devuelve el número de elementos que tiene, devuelve 1 si solo tiene un elemento (aunque no sea matriz) y 0 si no tiene ningún valor.

Otra función que nos permite saber el número de elementos es **sizeof(matriz)** .
Para acceder a los elementos de una matriz asociativa debemos usar la función**ecah()** que recupera el par formado por clave y valor y además avanza una posición de puntero. Su sintaxis es each(matriz) y los valores que devuelve la matriz asociativa son los siguientes:

Clave	Significado
0	Nombre de la clave
1	Valor asociativo de la clave
key	Nombre de la clave
value	Valor asociado a la clave

La función que realiza el constructor **list(variable1,variable2...variableN)** es asignar los valores del elemento actual de una matriz a las variables indicadas como parámetro.

Navegación sobre matrices

Cuando se trata de matrices indexadas la navegación es sencilla ya que tan solo basta acceder al elemento que queremos mostrar, pero al tratarse de alguna matriz asociativa no se puede aplicar el mismo tratamiento. Para ello existen un conjunto de funciones prefabricadas que nos permiten realizar multitud de acciones:

Sintaxis	Acción
reset(matriz);	El puntero interno vuelve a la primera posición
end(matriz);	El puntero interno va a la ultima posición
next(matriz);	El puntero va al elemento siguiente
prev(matriz);	Accede al elemento anterior
current(matriz);	Devuelve el contenido del elemento actual

Inserción de elementos

Para la inserción de elementos dentro de un array existen una serie de funciones que nos permiten añadir elementos. Entre ellas destacamos:

array_push(matriz,variable1,variableN);
Añade elementos al final de la función y su longitud se incrementara tantos elementos como se hayan añadido

array_unshift(matriz,variable1,variableN);
Añade elementos al principio de la función desplazando a los otros tantas posiciones como elementos haya.

array_pad(matriz,nuevo_tamaño,valor_relleno);
Aumenta el tamaño de la matriz empleando un valor proporcionado como relleno.

Eliminación de elementos

array_shift(matriz); Elimina el primer elemento de la matriz

array_pop(matriz);
Elimina el último elemento de la matriz

array_splice(entrada,pos_ini,[tamaño],[sustitutos]);
Se usa para reemplazar o borrar el contenido de una porción de matriz, para ello debemos especificar la posición desde la cual queremos iniciar el borrado o sustitución, el tamaño o número de elementos que se verán afectados y los sustitutos (en caso que deseemos sustituirlo por algún elemento).

array_keys(matriz,[valor buscado]);
Se emplea cuando deseamos eliminar un elemento cuya posición desconocemos.

array_values(matriz);
Devuelve una matriz indexada con todos los valores almacenados en la matriz pasada como parámetro.

Manipulación masiva de matrices

array_walk(matriz,nombre_de_la_funcion,lista_parametros);
Se emplea para realizar el mismo proceso definido en la función en todos los elementos incluidos.

Obtención de submatrices

array_slice(matriz,posicion,tamaño);
Permite extraer una secuencia de elementos de una matriz. Los parámetros a pasarle son la matriz en la cual queremos extraer dichos elementos, la posición desde la que se inicia la extracción y el tamaño de la extracción (posiciones que abarcamos a partir de la inicial).

Ordenación de matrices

Criterio	**Función**
Orden ascendente(matriz indexada)	sort(matriz)
Orden descendente(matriz indexada)	rsort(matriz)
Orden ascendente por valor(matriz asociativa)	asort(matriz)
Orden descendente por valor(matriz asociativa)	arsort(matriz)
Orden ascendente por clave(matriz asociativa)	ksort(matriz)
Orden descendente por clave(matriz asociativa)	krsort(matriz)

Otras funciones

En este apartado se comentaran una serie de funciones (no todas porque seria imposible) que nos pueden servir en cierto momento.

compact() Devuelve una matriz asociativa a partir de un numero indeterminado de parámetros
extract() Crea variables desde matriz asociativa
array_unique() Devuelve matriz sin datos repetidos ya que algunos se eliminan
array_reverse() Devuelve matriz con mismos elementos pero en orden inverso
shuffle() Modifica el orden de elementos de forma aleatoria
array_count_values() Devuelve una matriz asociativa que contiene

frecuencias de repetición de los valores de la matriz **in_array()** Permite comprobar si un valor esta en la matriz **array_merge()** Combina elementos de dos matrices en 1

2.2.7.7 Clases

Las **Clases** son máximo exponente de la Programación Orientada a Objetos (POO). PHP no es un lenguaje orientado a objeto, pero implementa las características que permiten definir las clases.

Pero, ¿qué son las Clases y para que sirven?. Empecemos por los segundo, sirven hacer el código más legible, y lo que es más importante, reutilizable. Escribir una Clase es sin duda más largo que escribir el código directamente, pero a la larga es más rentable por su portabilidad a otras aplicaciones y su mantenimiento.

Las Clases no son más que una serie de variables y funciones que describen y actúan sobre algo. Por ejemplo, vamos a crear la clase automóvil , la cual tendrá diversas variables, $color , $modelo , $marca , $potencia , $matricula y habrá una serie de funciones que actuarán sobre la clase automóvil como Precio() , Acelerar() , Frenar() , Girar() y Reparar() .

Como ejemplo vamos a crear la clase mysql , que nos servirá para realizar consultas a las bases de datos MySQL.

```
<?php

class DB_mysql {

/* variables de conexión */

var $BaseDatos;

var $Servidor;

var $Usuario;

var $Clave;

/* identificador de conexión y consulta */

var $Conexion_ID = 0;

var $Consulta_ID = 0;

* número de error y texto error */
```

```
var $Errno = 0;

var $Error = "";

/* Método Constructor: Cada vez que creemos una variable

de esta clase, se ejecutará esta función */

function DB_mysql($bd = "", $host = "localhost", $user = "nobody", $pass = "") {

$this->BaseDatos = $bd;

$this->Servidor = $host;

$this->Usuario = $user;

$this->Clave = $pass;

}

/*Conexión a la base de datos*/

function conectar($bd, $host, $user, $pass){

if ($bd != "") $this->BaseDatos = $bd;

if ($host != "") $this->Servidor = $host;

if ($user != "") $this->Usuario = $user;

if ($pass != "") $this->Clave = $pass;

// Conectamos al servidor

$this->Conexion_ID = mysql_connect($this->Servidor, $this->Usuario, $this-
>Clave);

if (!$this->Conexion_ID) {

$this->Error = "Ha fallado la conexión.";

return 0;

}

//seleccionamos la base de datos
```

```
- if (!@mysql_select_db($this->BaseDatos, $this->Conexion_ID)) {
$this->Error = "Imposible abrir ".$this->BaseDatos ;
return 0;
}
/* Si hemos tenido éxito conectando devuelve
el identificador de la conexión, sino devuelve 0 */
eturn $this->Conexion_ID;
}
/* Ejecuta un consulta */
function consulta($sql = ""){
if ($sql == "") {
$this->Error = "No ha especificado una consulta SQL";
eturn 0;
}
//ejecutamos la consulta
$this->Consulta_ID = @mysql_query($sql, $this->Conexion_ID);
if (!$this->Consulta_ID) {
$this->Errno = mysql_errno();
$this->Error = mysql_error();
3       }
/* Si hemos tenido éxito en la consulta devuelve
el identificador de la conexión, sino devuelve 0 */
return $this->Consulta_ID;
}
```

```
/* Devuelve el número de campos de una consulta */
function numcampos() {
return mysql_num_fields($this->Consulta_ID);
}
/* Devuelve el número de registros de una consulta */
function numregistros(){
return mysql_num_rows($this->Consulta_ID);
}
4       /* Devuelve el nombre de un campo de una consulta */
function nombrecampo($numcampo) {
return mysql_field_name($this->Consulta_ID, $numcampo);
}
/* Muestra los datos de una consulta */
function verconsulta() {
echo "<table border=1>n";
// mostramos los nombres de los campos
for ($i = 0; $i < $this->numcampos(); $i++){
echo "<td><b>".$this->nombrecampo($i)."</b></td>n";
}
echo "</tr>n";
// mostrarmos los registros
while ($row = mysql_fetch_row($this->Consulta_ID)) {
echo "<tr> n";
for ($i = 0; $i < $this->numcampos(); $i++){
```

```
echo "<td>".$row[$i]."</td>n";

}

echo "</tr>n";

}

}

} //fin de la Clse DB_mysql

?>
```

Como habreis observado, para crear una clase utilizamos la sentencia class , y además hemos creado una función con el mismo nombre que la clase, a esa función se le llama **constructor** y se ejecutará cada vez que definamos una variable de esa clase. No es obligatorina variable de esa clase. No es obligatorio crear un**constructor** en una definición de clase.

Otra cosa importante en las clases es el operador -> , con el que indicamos una variable o método (parte derecha del operador) de una clase (parte izquierda del operador). Para hacer referencia a la clase que estamos creando dentro de su definición, debemos utilizar this .

Y ahora veamos un ejemplo de la clase que hemos creado, y supongamos que el código anterior lo hemos guardado en un fichero llamado **clase_mysql.inc.php** .

```
<body>

html>

<?php

require ("clase_mysql.inc.php");

$miconexion  = new DB_mysql ;

$miconexion->conectar("mydb",  "localhost", "nobody", "");

$miconexion->consulta("SELECT  * FROM agenda");

$miconexion->verconsulta();

?>

</body>
```

</html>

2.2.7.8 Operaciones con Ficheros

En el desarrollo y administración de sitios webs resulta bastante habitual tener que acceder a ficheros del servidor para manipularlos. Por esta razón en este capítulo vamos a describir las funciones creadas en PHP para realizar dichas operaciones.

Cambio, creación y borrado de directorios

chdir(ruta_al_directorio); Nos permite cambiar el directorio activo a la ruta establecida como parámetro.
mkdir(ruta_al_directorio,permisos); Esta función crea un nuevo directorio en la ruta que hemos indicado, el segundo parámetro debe ser un numero octal y es por el que vienen determinados los permisos.
rmdir(ruta_directorio); Borra el directorio pasado como parámetro.

Procesamiento de los elementos de un directorio

Supongamos que queremos realizar una operación determinada como una búsqueda, visualización, etc sobre todos los ficheros de un directorio. PHP nos proporciona una solución a este problema: el manejador de directorios (representa una conexión lógica con un directorio determinado que permite leer la lista con los nombres de los elementos contenidos en el directorio actual).

La función empleada para abrir un directorio es **opendir(ruta);** cuya función como ya se ha comentado es abrir el directorio de la ruta especificada. Una vez se ha ejecutado opendir() podemos realizar tres operaciones:

La función **readdir(manejador);** nos devuelve una cadena con el nombre del siguiente elemento del directorio, ya sea un subdirectorio o un fichero.
La función **rewinddir(manejador);** procesa un directorio y sitúa el puntero interno en el primer directorio.
La función **closedir(manejador);** finaliza el tratamiento de entradas de directorio.

La clase dir

PHP nos proporciona una pseudoclase predefinida para el manejo de ficheros. Esta clase no aporta ninguna funcionalidad que no hayamos visto hasta este punto pero recopila todas las funciones a partir de una sola. Para poder trabajar con un directorio primero hay que crear una instancia de clase dir por medio de su constructor.**$directorio=dir(ruta_directorio);**

Este objeto cuenta con 3 métodos y 2 propiedades(las propiedades sólo de consulta por lo que no pueden ser modificadas. Los métodos empleados son **read(),rewind() y close()**

Copiado,borrado y renombrado de ficheros

copy(fichero_origen,fichero_destino); Realiza una copia de un fichero.
unlink(nombre_fichero); Elimina el fichero.
rename(nombre_antiguo,nombre_nuevo); Renombra el fichero pasado como parámetro.

Atributos de ficheros y directorios

Los ficheros y directorios poseen una serie de características propias denominadas atributos. PHP pone a nuestra disposición un conjunto de funciones que nos permitirán obtener información sobre los archivos o carpetas.

La función **file_exists(elemento);** Comprueba que el elemento pasado como parámetro exista.

filesize(nombre_fichero); nos informa sobre el tamaño del fichero en bytes.

La función fileatime(fichero); nos informa sobre el ultimo acceso al fichero.

La función filemtime(fichero); nos informa sobre la ultima modificación del fichero.
La función filectime(fichero); nos informa sobre el último cambio al fichero.
La función filetype(fichero); nos devuelve el tipo de elemento que estamos tratando. Los resultados posibles que puede devolver son:

Resultado	Significado
block	Dispositivo de bloques
char	Caracteres
dir	Directorio
fifo	FIFO
file	Fichero
link	Enlace
unknown	Desconocido

Chmod(elemento_directorio,permisos); recibe como parámetro el elemento y los permisos que deseamos otorgarle a dichos elementos

2.2.8 Conexión a Base de Datos

Las variables del código son:

$host: Nombre del host en el que está la base de datos
$user : Un nombre de usuario de la base de datos.
$password: Clave para ese usuario

El comando or die salta cuando hay un error y nos permite mostrar un mensaje por pantalla (como en el ejemplo) o llamar a otra función. La función mysql_connect() nos conecta con una base de datos de mysql en el puerto por defecto de mysql. Si devuelve null, es que no se ha podido conectar. Nos devuelve el "enlace" con la base de datos en $link.

Nota: Hay que tener en cuenta el trato de las claves, usuarios, etc. Ya que cualquiera que esté en posesión de un usuario de la BD con suficientes permisos puede hacer mucho daño.

Una optimización sería añadir al inicio [include('ficheroDatos.php')] un fichero con las claves encriptadas (en md5 por ejemplo) y utilizarlas con más tranquilidad en el código.
Hemos supuesto que hay dos campos nada más en la tabla y vamos a sacar el resultado en un normalito de HTML. Las variables son
$dbname: Nombre de la base de datos a usar.
$tablename: Nombre de la tabla a usar
$query: La consulta que queremos realizar
mysql_db_query realiza la consulta y devuelve los resultados
mysql_fetch_array () vamos obteniendo cada uno de las filas resultado de la consulta.
$row como un array en el que el indice es el nombre del campo en la base de datos.
mysql_free_result() liberamos la memoria que hemos reservado para gestionar el conjunto de campos devueltos por la base de datos (que puede ser muy grande según lo que saquemos con la query).

2.3 ¿Qué es una DB?

Data Base = Base de datos en inglés

¿Qué es una BBDD o BD?
Base de datos

¿Para qué sirve una DB?
Almacenar datos

¿Cómo se aprende a programar en mySQL?
No se aprende, mySQL es una base de datos, no un lenguaje

¿SQL es una base de datos?
SQL server sí, pero SQL es un lenguaje

¿Cuántos tipos hay?
Muchos, pero los habituales en web son Access, mySQL y SQL Server 2000

¿Necesito algún lenguaje especial para conectarme?
Sí, SQL es el lenguaje para interactuar con las bases de datos

¿Qué es un Query?
Una consulta a la base de datos

¿Qué es ABM?
Alta, Baja y Modificación de cualquier cosa. Usualmente de una base de datos que contiene datos.

¿Qué es CMS?
Control Managment System : Un panel de control que administra un web site, y a menudo una base de datos.

Ahora, aclaradas estas dudas básicas, vamos al grano. Separaré en varios items la explicación a fin de un mejor orden.

2.3.1 Funcionalidad de un DB

Una base de datos (sea cual sea) es un soporte digital que tiene como fin el almacenamiento masivo de información en formato texto plano. No es capaz de almacenar imágenes como se cree, sino que almacena las rutas (path) de dichas fotos; ni almacena otro tipo de datos; sino que almacena sus rutas de acceso de ser necesario.

Las bases de datos, son utilizadas en sistemas que requieren una interacción

fluida con la aplicación; estas se encargan muchas veces de administrar, editar, y dar de alta. Usualmente la base de datos, esta ligada a la programación directa del site, causando que una edición en ella cause una modificación directa en lo que ve el usuario.

Ejemplos de aplicación de una base de datos (entiéndase que están ligadas a un lenguaje dinámico como PHP o ASP):
E – comerce, Agendas, Libros de visitas, foros, portales, etc.

2.3.2 Estructura normal de una DB

Una base de datos, a fin de ordenar la información de manera lógica, posee un orden que debe ser cumplido para acceder la información de manera coherente.

Cada base de datos tiene una o más tablas, las cuales cumplen la función de contener los campos. Un ejemplo de tabla sería “contactos”. Para entender mejor esto, sería como un libro en el excel. Mientras que los campos serían las columnas del excel donde se ordena cada datos insertado al libro. Ejemplo “id, nombres, apellidos, teléfono”. Y luego finalmente tenemos las filas (row), que son la información propiamente dicha.

Por consiguiente una base de datos posee el siguiente orden jerárquico:

- Tablas
- Campos
- Registros

2.3.3Tipos usuales de bases de datos en la Web

En la web, se suelen usar 3 tipos de bases de datos:

Access: Es una base de datos desarrollada por Microsoft comúnmente utilizada bajo el lenguaje ASP (Active Server Pages). Esta base de datos, debe ser creada bajo el programa access, el cual crea archivo .mdb con la estructura ya explicada. El programa usa un entorno gráfico normal, y es muy parecido a usar excel.

MySQL: Es una base de datos con licencia GPL basada en un servidor, puede ser sólo creada por código. Usualmente se utiliza el programa phpMyAdmin

como soporte para administrar la base de datos en el nivel de programación (a un usuario normal le resultaría complicado utilizarla desde linea de comandos).

SQL Server: Es una base de datos más potente que access desarrollada por Microsoft también, que se supone es recomendable arriba del millon datos.

PostgreSQL / Oracle: Son realmente mucho más poderosas que todas las antes mencionadas, aunque también se duplican los problemas. Administra muy bien grandes cantidades de datos, y suelen ser utilizadasen intranets y sistemas de gran calibre.

2.3.4 Crear y seleccionar una base de datos

Si el administrador crea su base de datos en el mismo momento que le otorga privilegios, puede comenzar a utilizarla, de lo contrario necesitará crearla:

```
mysql> CREATE DATABASE menagerie;
```

En ambientes Unix, los nombres de las bases de datos son case sensitive (al contrario que las palabras clave), de modo que siempre debe referirse a su base de datos como **menagerie**, y no **Menagerie**, **MENAGERIE**, o una variante similar. Esto también se aplica a los nombres de tablas. Esta restricción no existe en Windows, aunque puede utilizar el mismo esquema de mayúsculas cuando se refiera a bases de datos y tablas en una consulta dada.

Al crear una base de datos, ésta no se selecciona para su uso, debe hacerlo explicitamente. Para convertir a**menagerie** en la base de datos actual, use este comando:

```
mysql> USE menagerie
Database changed
```

Las bases de datos sólo necesitan ser creadas una sola vez, pero deben ser seleccionadas cada vez que se inicia una sesión de **mysql**. Puede hacerse a través del comando **USE** como se muestra en el ejemplo, o puede indicar la base de datos en la linea de comandos al ejecutar **mysql**. Simplemente debe indicar el nombre de la base de datos a continuación de los parámetros que necesite ingresar. Por ejemplo:

```
shell> mysql -h host -u user -p menagerie
Enter password: ********
```

Advierta en el comando anterior que **menagerie** ***no*** es la contraseña. Si se quisiera suministrar la contraseña en la linea de comandos, después de la opción **-p**, debe hacerse sin dejar espacios en blanco (por ejemplo, **-**

pmypassword, no **-p mypassword**). De todos modos, colocar la contraseña en la linea de comandos no es recomendable porque lo expone a la vista de otros usuarios.

En algunos casos, podría querer ejecutar múltiples servidores **mysqld** en la misma máquina. Quizá quiera probar una nueva versión de MySQL dejando la configuración de producción sin cambios. O quizá quiera dar acceso para diferentes usuarios a diferentes servidores **mysqld** que ellos mismos puedan administrar. (Por ejemplo, podría ser un proeedor de servicios de Internet que proporcione instalaciones de MySQL independientes para cada cliente.)

Para ejecutar múltiples servidores en una única máquina, cada servidor tiene que tener valores únicos para los diferentes parámetros de operación. Estos se pueden establecer en la línea de comandos o en archivos de opciones. Consulte Sección 4.3, “Especificar opciones de programa”.

Al menos las siguientes opciones deben ser diferentes para cada servidor:

- **--port=*port_num***

  --port controla el número de puerto de las conexiones TCP/IP.

- **--socket=*path***

  --socket controla la ruta del archivo socket de Unix y el nombre de la named pipe en Windows. En Windows, es necesario especificar diferentes nombres de pipe solo para los servidores que soporten conexiones mediante named pipe.

- **--shared-memory-base-name=*name***

  Esta opción, actualmente, se utiliza sólo en Windows. Designa el nombre de memoria compartida utilizado por un servidor Windows para permitir a los clientes conectarse mediante memoria compartida.

- **--pid-file=*path***

  Esta opción se utiliza únicamente en Unix. Indica el nombre del archivo en el que el servidor escribe su ID de proceso.

Si utiliza las siguientes opciones de archivo de registro, deben ser diferentes para cada servidor:

- **--log=*path***
- **--log-bin=*path***
- **--log-update=*path***
- **--log-error=*path***

- `--bdb-logdir=path`

Las opciones de archivo de registro se explican en Sección 5.10.5, "Mantenimiento de ficheros de registro (log)".

Para un mejor rendimiento, puede especificar las siguientes opciones a cada servidor, de manera que se distribuya la carga entre discos físicos:

- `--tmpdir=path`
- `--bdb-tmpdir=path`

Tener diferentes directorios temporales también es una buena práctica, para hacer más fácil determinar qué servidor MySQL creó cualquier archivo temporal.

Generalmente, cada servidor debería utilizar un directorio de datos diferente, lo que se especifica con la opción `--datadir=path`.

Atención: Normalmente no debería tener dos servidores que actualicen los datos de la misma base de datos. Esto podría llevar a obtener sorpresas indeseadas si su sistema operativo no tiene soporte para bloqueos sin posibilidad de fallo. Si (a pesar de este aviso) ejecuta múltiples servidores que utilicen el mismo directorio de datos y tienen el registro activado, debería utilizar las opciones adecuadas para especificar nombres de archivos de registro que sean únicos para cada servidor. De otra manera, lo servidores intentan registrar en los mismos archivos. Por favor, tengan cuenta que este tipo de configuración sólo funciona con tablas `MyISAM` y `MERGE`, y no con ningún otro de los motores de almacenamiento.

Este aviso en contra de compartir un directorio de datos entre servidores también se aplica en entornos NFS. Permitir que múltiples servidores MySQL accedan a un directorio común es una *muy mala idea*.

- El principal problema es que NFS es un cuello de botella de velocidad. No está diseñado para un uso tal.

- Otro riesgo con NFS es que tiene que encontrar una manera de asegurarse que dos o más servidores no se interfieran unos con otros. Usualmente, el bloqueo de archivos de NFS está gestionado por el demonio `lockd`, pero de momento no hay ninguna plataforma que realice bloqueos 100% seguros en todas las situaciones.

Facilítese las cosas: Olvídese de compartir un directorio de datos entre servidores sobre NFS. Una solución mejor es tener una máquina que tenga

diferentes CPUs y utilizar un sistema operativo que gestione los hilos de ejecución eficientemente.

Si tiene múltiples instalaciones de MySQL en diferentes lugares, normalmente puede especificar el directorio base de instalación para cada uno con la opción `--basedir=path`, de manera que cada servidor utilice un directorio de datos, archivo de registro, y archivo de PID diferentes. (Los valores por defecto de todos ellos son determinados en relación al directorio base). En ese caso, las únicas opciones adicionales que necesita especificar son las opciones `--socket` y `--port`. Por ejemplo, suponga que debe instalar diferentes versiones de MySQL utilizando distribuciones binarias en archivos `tar`. Se instalan en diferentes lugares, así que puede iniciar el servidor de cada instalación utilizando el comando **bin/mysqld_safe** bajo su correspondiente directorio base. ¡**mysqld_safe** determina la opción `--basedir` apropiada para pasarle a **mysqld**, y usted sólo necesita especificar las opciones `--socket` y `--port` a **mysqld_safe**.
Tal como se explica en las siguientes secciones, es posible iniciar servidores adicionales mediante el establecimiento de sus variables de entorno, o especificando opciones de línea de comandos apropiadas. Aún así, si necesita ejecutar servidores múltiples de manera permanente, es más conveniente utilizar archivos de opciones para especificar a cada servidor las opciones que deben ser únicas para él.

Conexiones Persistentes a Bases de Datos

Las conexiones persistentes son enlaces que no se cierran cuando la ejecución del script termina. Cuando una conexión persistente es solicitada, PHP chequea si ya existe una conexión persistente idéntica (que fuera abierta antes) - y si existe, la usa. Si no existe, crea el enlace. Una conexión "Idéntica" es una conexión que fue abierta por el mismo host, con el mismo usuario y el mismo password (donde sea aplicable).

Las personas que no estan completamente familiarizadas con como trabajan los web servers y la distribución de carga podrían equivocarse en usar conexiones persistentes para lo que no son. En

particular, ***no*** le da la habilidad de abrir "sesiones de usuario" en el mismo enlace, ***no*** le da la habilidad de construir una transacción eficiente, pero no hacen muchas otras cosas más. De hecho, para ser extremadamente claros acerca de esto, las conexiones persistentes no dan ***cualquier*** otra funcionalidad que no fuera posible hacerse con sus hermanas no-persistentes.

¿Por qué?

Esto tiene que ver con la forma en que los web servers trabajan. Hay 3 formas en las cuales el web server puede generar paginas web usando PHP.

El primer método es usar PHP como una "capa" CGI. Cuando se ejecuta de esta forma, una instancia del interprete PHP es creado y destruido por cada solicitud de la página (para una página PHP) al web server. Porque es destruido después de cada solicitud, cualquier recurso que se necesite (como un enlace a un servidor de base de datos SQL) son cerradas cuando son destruidas. En este caso, no se gana nada intentando usar conexiones persistentes -- simplemente no persisten.

La segunda, y mas popular, es ejecutar PHP como un modulo en un servidor multiproceso, el cual actualmente solo incluye a Apache. Un servidor multiproceso normalmente tiene un proceso (el padre) el cual coordina un grupo de procesos (los hijos) los cuales no trabajan sirviendo páginas web. Cuando una solicitud viene desde el cliente, es manejada por uno de los hijos el cual no este sirviendo a otro cliente. Esto significa que cuando el mismo cliente hace una segunda solicitud al servidor, podría ser servido por un proceso hijo diferente a la primera vez. Cuando se abre una conexión persistente, cada petición siguiente de servicios SQL puede reusar la misma conexión establecida al servidor SQL.

El último método es usar PHP como un plug-in para un servidor web multihilos. Actualmente PHP 4 tiene soporte para ISAPI, WSAPI, y NSAPI (en Windows), los cuales permiten usar PHP como un plug-in multihilo en servidores como Nestcape FastTrack (iPlanet), Microsoft Internet Information Server (IIS), y O'Reilly's WebSite Pro. El comportamiento es esencialmente el mismo para el modelo multiproceso descrito antes.

Si las conexiones persistentes no tienen ninguna funcionalidad adicional, ¿Para que son buenas?

La respuesta es extremadamente simple -- eficiencia. Las conexiones persistentes son buenas si la sobrecarga para crear enlaces al servidor SQL son altas. Que hallan o no sobrecargas depende de muchos factores. Como, cual base de datos se usa, que sea o no la misma computadora en que esta el servidor web, así como la carga de la maquina que tiene el servidor SQL y así por el estilo. Lo esencial es que si la sobrecarga de conexiones es alta, las conexiones persistentes ayudan considerablemente. Podrían causar que los procesos hijos únicamente se conecten una vez en todo lo que duran, en lugar de que se haga cada vez que se procese una página que se conecte a un servidor SQL. Esto significa que por cada hijo que abrió una conexión persistente mantendrá una conexión persistente al servidor. Por ejemplo, si se tienen 20 procesos hijos diferentes que ejecutaran un script que hace una conexión persistente al servidor SQL, se tendrían 20 conexiones diferentes al servidor SQL, una por cada hijo.

Nótese, sin embargo, que esto puede tener algunos inconvenientes si se esta usando una base de datos con un limite de conexiones que sean excedidas por las conexiones persistentes hijas. Si la base de datos tiene un limite de 16 conexiones simultaneas, y en el curso de una sesión ocupada del servidor, 17 hilos intentan conectarse, uno no sera posible de hacerse. Si hay bugs en los scripts los cuales no contemplen los cierres de las conexiones (como un loop infinito), la base de datos con los 16 conexiones podría rápidamente hundida. Chequear la documentación de la base de datos para obtener información de como manejar conexiones abandonadas u ociosas.

2.3.5 Descripción de las tablas que conforman la base de datos

Tabla: I_Internacionales

Contenido: Instrumentos Internacionales

Campo – Descripción- Tipo de datos - Tamaño

ID Código que identifica al instrumento Autonumérico -

Instrumento Código correspondiente a la clasificación general del

instrumento (ver tabla lista_instrumentos)

Numérico Entero largo

Tipo Código del tipo de instrumento (ver tabla lista_tipos) Numérico Entero largo

Nombre Nombre del instrumento Texto 255

Numero Número de instrumento Texto 30

Pais Código del país donde se firmó el instrumento (ver

tabla lista_paises)

Texto 2

Lugar Lugar donde se firmó el instrumento (localidad,

provincia, estado)

Texto 50

Fecha_Sancion Fecha de sanción del instrumento Fecha / Hora Fecha

Organismo_Sancion Organismo que sancionó el instrumento Texto 255

En_Vigor Si se encuentra en vigor el instrumento Sí/No -

Fecha_Vigor Fecha desde la entrada en vigor del instrumento Fecha / Hora Fecha

Alcance Alcance del instrumento Texto 1

Firma_AR Si Argentina firmó el instrumento Sí/No -

Fecha_Firma_AR Fecha en que Argentina firmó el instrumento Fecha / Hora Fecha

Firma_UY Si Uruguay firmó el instrumento Sí/No -

Fecha_Firma_UY Fecha en que Uruguay firmó el instrumento Fecha / Hora Fecha

Firma_Otro Código de otro país que hubiera firmado el

instrumento (ver tabla lista_paises)

Texto 2

Ratificado_AR Si Argentina ratificó el instrumento Sí/No -

Ratificado_AR_Norma Código de la norma utilizada por Argentina para ratificar el instrumento (ver tabla I_Nacionales)

Numérico Entero largo

Ratificado_UY Si Uruguay ratificó el instrumento Sí/No -

Ratificado_UY_Norma Código de la norma utilizada por Uruguay para ratificar el instrumento (ver tabla I_Nacionales)

Numérico Entero largo

Terminado_AR Si Argentina terminó el instrumento Sí/No -

Terminado_AR_Norma Código de la norma utilizada por Argentina para terminó el instrumento (ver tabla I_Nacionales)

Numérico Entero largo

Terminado_UY Si Uruguay terminó el instrumento Sí/No -

Terminado_UY_Norma Código de la norma utilizada por Uruguay para terminó el instrumento (ver tabla I_Nacionales)

Numérico Entero largo Claves Lista de palabras que permiten describir al instrumento

(separadas por punto y coma)

Memo -

TextoCompleto1 Texto del instrumento Memo -

TextoCompleto2 Continuación del texto del instrumento Memo -

TextoCompleto3 Continuación del texto del instrumento Memo –

TextoCompleto4 Continuación del texto del instrumento Memo -

TextoCompleto5 Continuación del texto del instrumento Memo -

TextoCompleto6 Continuación del texto del instrumento Memo -

TextoCompleto7 Continuación del texto del instrumento Memo -

TextoCompleto8 Continuación del texto del instrumento Memo -

TextoCompleto9 Continuación del texto del instrumento Memo -

TextoCompleto10 Continuación del texto del instrumento Memo -

Archivo Nombre del archivo que contiene el texto completo del

instrumento.

Texto 255

Observaciones Cualquier información de interés adicional Memo -

Fecha_Actualizacion Fecha de actualización del registro Fecha / Hora Fecha

Tabla: I_Nacionales

Contenido: Instrumentos Nacionales

Campo Descripción Tipo de datos Tamaño

ID Código que identifica al instrumento Autonumérico -

Instrumento Código correspondiente a la clasificación general del

instrumento (ver tabla lista_instrumentos)

Numérico Entero largo

Tipo Código del tipo de instrumento (ver tabla lista_tipos) Numérico Entero largo

Nombre Nombre del instrumento Texto 255

Numero Número de instrumento Texto 30

Pais Código del país donde se firmó el instrumento (ver

tabla lista_paises)

Texto 2

Lugar Lugar donde se firmó el instrumento (localidad,

provincia, estado)

Texto 50

Fecha_Sancion Fecha de sanción del instrumento Fecha / Hora Fecha

Organismo_Sancion Organismo que sancionó el instrumento Texto 255

Organismo_Aplicacion Organismo de aplicación del instrumento Texto 255

Fecha_Publicacion Fecha de publicación del instrumento Fecha / Hora Fecha

Fuente_Publicacion Fuente donde se publicó el instrumento Texto 100

Claves Lista de palabras que permiten describir al instrumento

(separadas por punto y coma)

Memo -

TextoCompleto1 Texto del instrumento Memo - TextoCompleto2 Continuación del texto del instrumento Memo -

TextoCompleto3 Continuación del texto del instrumento Memo -

TextoCompleto4 Continuación del texto del instrumento Memo -

TextoCompleto5 Continuación del texto del instrumento Memo -

TextoCompleto6 Continuación del texto del instrumento Memo -

TextoCompleto7 Continuación del texto del instrumento Memo -

TextoCompleto8 Continuación del texto del instrumento Memo -

TextoCompleto9 Continuación del texto del instrumento Memo –

TextoCompleto10 Continuación del texto del instrumento Memo -

Archivo Nombre del archivo que contiene el texto completo del

instrumento.

Texto 255

Observaciones Cualquier información de interés adicional Memo -

Fecha_Actualizacion Fecha de actualización del registro Fecha / Hora Fecha

Tabla: Lista_Alcances

Contenido: Lista de alcances de instrumentos internacionales

Campo Descripción Tipo de datos Tamaño

Codigo_Alcance Código que identifica al alcance del instrumento Texto 2

Alcance Nombre del alcance del instrumento Texto 15

Tabla: Lista_Instrumentos

2..3.6.C**ontenido: Clasificación general de los instrumentos**

Campo - Descripción -Tipo de datos -Tamaño

ID_Instrumento Código que identifica al tipo de instrumento (según

una clasificación general)

Numérico Entero largo

Instrumento Clasificación general del instrumento Texto 30

Plural Plural del campo instrumento Texto 35

Lista_Temas

Contenido: Lista de temas

Campo Descripción Tipo de datos Tamaño

ID Código que identifica al tema Autonumérico -

ID_padre Código que identifica al tema al cual este corresponde

(ver tabla lista_temas)

Numérico Entero largo

Nombre Nombre del tema Texto 80

Detalle Nombre detallado del tema Texto 50

Tabla: Lista_Tipos

Contenido: Lista de tipos de instrumentos

Campo Descripción Tipo de datos Tamaño

ID_Tipo Código que identifica al tipo de instrumento Autonumérico -

ID_Instrumento Código del tipo de instrumento (ver tabla

lista_instrumentos)

Numérico Entero largo

Tipo Nombre del tipo de instrumento Texto 60

Orden Orden jerárquico del tipo de instrumento Numérico Byte

Tabla: Temas_Internacionales

Contenido: Temas correspondientes a los instrumentos internacionales

Campo Descripción Tipo de datos Tamaño

ID Código que identifica al registro Autonumérico -

ID_Instrumento Código del instrumento internacional (ver tabla

I_Internacionales)

Numérico Entero largo

ID_Tema Código del tema (ver tabla lista_temas) Numérico Entero largo

Tabla: Temas_Nacionales

Contenido: Temas correspondientes a los instrumentos nacionales

Campo Descripción Tipo de datos Tamaño

ID Código que identifica al registro Autonumérico -

ID_Instrumento Código del instrumento nacional (ver tabla

I_Nacionales)

Numérico Entero largo

ID_Tema Código del tema (ver tabla lista_temas) Numérico Entero larg

Tabla: Biblio_Claves

Contenido: Temas correspondientes a las referencias bibliográficas

Campo- Descripción- Tipo de datos -Tamaño

id_bibliografia Código de la referencia bibliográfica (ver tabla bibliografia) Numérico Entero largo

id_clave Código del tema (ver tabla lista_claves) Numérico Entero largo

Tabla: Biblio_Ubicacion

Contenido: Ubicaciones correspondientes a las referencias bibliográficas

Campo Descripción Tipo de datos Tamaño

id Código que identifica al registro Autonumérico -

id_bibliografia Código de la referencia bibliográfica (ver tabla bibliografia)

Numérico Entero largo

id_ubicacion Código de la ubicación (ver tabla ubicacion) Numérico Entero largo

Tabla: Bibliografia

Contenido: Referencias bibliográficas

Campo Descripción Tipo de datos Tamaño

id Código que identifica a la referencia bibliográfica Autonumérico -

titulo1 Título de la bibliografía Texto 255

titulo2 Continuación del título de la bibliografía Texto 255

autor Autor de la bibliografía Texto 255

anio Año de publicación de la bibliografía Numérico Entero

id_tipo Código del tipo de bibliografía (ver tabla lista_tipos) Numérico Entero largo

id_publicacion Código de la publicación a la que pertenece la bibliografía

(ver tabla publicaciones)

Memo -

especificaciones Especificaciones de la bibliografía Memo -

resumen Resumen de la bibliografía Numérico Entero largo

fecha Fecha de actualización del registro Fecha / Hora Fecha

id_usuario Código del usuario que actualizó el registro (ver tabla

usuarios)

Numérico Entero largo Tabla: Lista_Claves

Contenido: Lista de temas

Campo Descripción Tipo de datos Tamaño

id Código que identifica al tema Autonumérico -

id_padre Código que identifica al tema al cual este corresponde (ver

tabla lista_claves)

Numérico Entero largo

nombre Nombre del tema Texto 80

descripcion Nombre detallado del tema Texto 50

Tabla: Lista_Tipos

Contenido: Lista de tipos de bibliografías y publicaciones

Campo Descripción Tipo de datos Tamaño

id Código que identifica al tipo de bibliografía y publicación Numérico Entero largo

tipo_bibliografia Nombre del tipo de bibliografía Texto 50

tipo_publicacion Nombre del tipo de publicación correspondiente al tipo de

bibliografía

Texto 50

Tabla: Publicaciones

Contenido: Publicaciones correspondientes a las referencias bibliográficas

Campo Descripción Tipo de datos Tamaño

id Código que identifica a la publicación Autonumérico -

id_tipo Código del tipo de publicación (ver tabla lista_tipos) Numérico Entero largo

nombre Nombre de la publicación Texto 255

fecha Fecha de actualización del registro Fecha / Hora Fecha

id_usuario Código del usuario que actualizó el registro (ver tabla

usuarios)

Numérico Entero largo

Tabla: Ubicacion

Contenido: Lista de ubicaciones

Campo Descripción Tipo de datos Tamaño

id Código que identifica a la ubicación Autonumérico -

nombre Nombre de la ubicación Texto 255

Tabla: Usuarios

Contenido: Lista de usuarios que pueden acceder a la base de datos

Campo Descripción Tipo de datos Tamaño

id Código que identifica al usuario Numérico Entero largo

nombre Nombre del usuario Texto 50 password Password del usuario Texto 50.

2.3.7 Cambios en el soporte a base de datos

- Sobre PDO
- Cambios de soporte en MySQL
- Cambios de soporte en SQLite

Sobre PDO

Los Objetos de Datos PHP (PDO) se introdujeron como extensión PECL en PHP 5.0, y comenzó a formar parte del núcleo de PHP en PHP 5.1.x. La extensión PDO ofrece una interfaz consistente de acceso a bases de datos, y funciona en conjunción con drivers PDO específicos a cada bases de datos. Cada driver puede además tener funciones específicas de su propia base de datos, pero las funcionalidades básicas de acceso, tal como hacer consultas o capturar dato, se realizan por las funciones de PDO, usando el driver que se haya indicado al llamar a PDO::__construct().

Tenga en cuenta que la extensión PDO y sus drivers están pensados para construirse como extensiones compartidas. Esto permite facilitar el proceso de actualización desde PECL, sin tener que reconstruir todo PHP.

Desde el punto de vista de la versión de PHP 5.1.x, PDO está completamente preparado para probarlo y puede utilizarse en la mayoría de escenarios. Sin embargo, es importante entender que tanto PDO como sus drivers no son una herramienta madura y pueden carecer de funcionalidades específicas de un motor de bases de datos; se debe analizar cuidadosamente PDO antes de usarse en nuevos proyectos.

Por norma general, el código fuente de versiones anteriores dependerá de las extensiones de base de datos preexistentes, las cuales seguirán siendo mantenidas.

Cambios de soporte en MySQL

En PHP 4, había soporte incorporado para MySQL 3. Con la llegada de PHP 5.0 había dos extensiones de MySQL, 'mysql' y 'mysqli', diseñadas para funcionar con MySQL < 4.1 y MySQL 4.1 o superior, respectivamente. Con la llegada de PDO, que proporciona una interfaz muy rápida a las APIs de las bases de datos soportadas por PHP, el driver PDO_MYSQL puede funcionar con cualquiera de las versiones (MySQL 3, 4 or 5) en códigos escritos para PDO, dependiendo de la versión de la biblioteca MySQL usada al compilarlo. La extensión de MySQL anterior continua disponible por retrocompatibilidad, pero no viene habilitada por omisión.

Cambios de soporte en SQLite

En PHP 5.0.x, el soporte para SQLite 2 venía incorporado por omisión con la extensión sqlite, también disponible como extensión PECL para PHP 4.3 y para PHP 4.4. Con la llegada de PDO, la extensión sqlite pasa a actuar como driver 'sqlite2' de PDO; es por esto que la extensión sql de PHP 5.1.x depende de la extensión PDO.

PHP 5.1.x incorpora numerosas interfaces alternativas para sqlite:

La extensión sqlite ofrece la "clásica" API procedural/OO a sqlite que se usaba en versiones anteriores de PHP. También contiene el driver PDO 'sqlite2', que permite acceder a bases de datos SQLite 2 usando la API de PDO.

PDO_SQLITE ofrece la versión 3 del driver 'sqlite'. La versión 3 de SQLite es muy superior a la 2, y el formato de ficheros de estas dos versiones no son compatibles.

Si un proyecto basado en SQLite estuviera escrito y funcionando con una versión de PHP anterior, se podrá seguir usando ext/sqlite sin ningún problema, pero se deberá habilitar explícitamente tanto PDO como sqlite. Los nuevos

proyectos deben usar PDO y el driver (versión 3) de 'sqlite', ya que es más rápido que SQLite 2, mejora la concurrencia, y soporta tanto sentencias preparadas, como columnas binarias, de forma nativa.

Se puede habilitar PDO usando la extensión SQLite. Si se deseara construir la extensión PDO como extensión compartida, se deberá construir SQLite también compartido, tal y como sucede con cualquier extensión que contenga un driver PDO.

2.3.8. Administración de Base de Datos

El administrador de base de datos (DBA) es la persona responsable de los aspectos ambientales de una base de datos. En general esto incluye lo siguiente:

- Recuperabilidad - Crear y probar Respaldos
- Integridad - Verificar o ayudar a la verificación en la integridad de datos
- Seguridad - Definir o implementar controles de acceso a los datos
- Disponibilidad - Asegurarse del mayor tiempo de encendido
- Desempeño - Asegurarse del máximo desempeño incluso con las limitaciones
- Desarrollo y soporte a pruebas - Ayudar a los programadores e ingenieros a utilizar eficientemente la base de datos.

El diseño lógico y físico de las bases de datos a pesar de no ser obligaciones de un administrador de bases de datos, es a veces parte del trabajo. Esas funciones por lo general están asignadas a los analistas de bases de datos o a los diseñadores de bases de datos.

2.3.9 MYSQL

MySQL es un sistema de gestión de base de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones.[1] MySQL AB —desde enero de 2008 una subsidiaria de Sun Microsystems y ésta a su vez de Oracle Corporación desde abril de 2009— desarrolla MySQL como software libre en un esquema de licenciamiento dual.

Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en ANSI C.

Al contrario de proyectos como Apache, donde el software es desarrollado por una comunidad pública y el copyright del código está en poder del autor individual, MySQL es patrocinado por una empresa privada, que posee el copyright de la mayor parte del código.

Esto es lo que posibilita el esquema de licenciamiento anteriormente mencionado. Además de la venta de licencias privativas, la compañía ofrece soporte y servicios. Para sus operaciones contratan trabajadores alrededor del mundo que colaboran vía Internet. MySQL AB fue fundado por David Axmark, Allan Larsson y Michael Widenius.

2.3.9.1 Motores de almacenamiento de MySQL y tipos de tablas

MySQL soporta varios motores de almacenamiento que tratan con distintos tipos de tabla. Los motores de almacenamiento de MySQL incluyen algunos que tratan con tablas transaccionales y otros que no lo hacen:

- MyISAM trata tablas no transaccionales. Proporciona almacenamiento y recuperación de datos rápida, así como posibilidad de búsquedas fulltext. MyISAM se soporta en todas las configuraciones MySQL, y es el motor de almacenamiento por defecto a no ser que tenga una configuración distinta a la que viene por defecto con MySQL.

- El motor de almacenamiento MEMORY proporciona tablas en memoria. El motor de almacenamiento MERGE permite una colección de tablas MyISAM idénticas ser tratadas como una simple tabla. Como MyISAM, los motores de almacenamiento MEMORY y MERGE tratan tablas no transaccionales y ambos se incluyen en MySQL por defecto.
  Nota: El motor de almacenamiento MEMORY anteriormente se conocía como HEAP.

- Los motores de almacenamiento InnoDB y BDB proporcionan tablas transaccionales. BDB se incluye en la distribución binaria MySQL-Max en aquellos sistemas operativos que la soportan. InnoDB también se incluye por defecto en todas las distribuciones binarias de MySQL 5.0 . En distribuciones fuente, puede activar o desactivar estos motores de almacenamiento configurando MySQL a su gusto.

- El motor de almacenamiento EXAMPLE es un motor de almacenamiento "tonto" que no hace nada. Puede crear tablas con este motor, pero no puede almacenar datos ni recuperarlos. El objetivo es que sirva como ejemplo en el código MySQL para ilustrar cómo escribir un motor de almacenamiento. Como tal, su interés primario es para desarrolladores.

- NDB Cluster es el motor de almacenamiento usado por MySQL Cluster para implementar tablas que se particionan en varias máquinas. Está disponible en distribuciones binarias MySQL-Max 5.0. Este motor de almacenamiento está disponible para Linux, Solaris, y Mac OS X . Añadiremos soporte para este motor de almacenamiento en otras plataformas, incluyendo Windows en próximas versiones.

- El motor de almacenamiento ARCHIVE se usa para guardar grandes cantidades de datos sin índices con una huella muy pequeña.

- El motor de almacenamiento CSV guarda datos en ficheros de texto usando formato de valores separados por comas.

- El motor de almacenamiento FEDERATED se añadió en MySQL 5.0.3. Este motor guarda datos en una base de datos remota. En esta versión sólo funciona con MySQL a través de la API MySQL C Client. En futuras versiones, será capaz de conectar con otras fuentes de datos usando otros drivers o métodos de conexión clientes.

Cada uno de los motores de almacenamiento MySQL excepto InnoDB y NDB Cluster, que se tratan de motor de almacenamiento InnoDB y Capítulo 16, MySQL Cluster.
Cuando crea una nueva tabla, puede decirle a MySQL qué tipo de tabla crear añadiendo la opción de tabla ENGINE o TYPE al comando CREATE TABLE :
CREATE TABLE t (i INT) ENGINE = INNODB;
CREATE TABLE t (i INT) TYPE = MEMORY;

Aunque se soporta TYPE en MySQL 5.0, ENGINE es el término preferido.

Si omite la opción ENGINE o TYPE, se usa el motor de almacenamiento por defecto, que es MyISAM. Puede cambiarlo usando las opciones de arranque --default-storage-engine o --default-table-type , o cambiando la variable de sistema storage_engine o table_type .

Cuando se instala MySQL en Windows usando el MySQL Configuration Wizard, InnoDB es el motor de almacenamiento por defecto en lugar de MyISAM. Consulte Sección 2.3.5.1, “Introducción”.

Para convertir una tabla de un tipo a otro, use un comando ALTER TABLE que indique el nuevo tipo:
ALTER TABLE t ENGINE = MYISAM;
ALTER TABLE t TYPE = BDB;

Consulte Sección 13.1.5, “Sintaxis de CREATE TABLE” y Sección 13.1.2, “Sintaxis de ALTER TABLE”.

Si trata de usar un motor de almacenamiento que no está compilado o que está desactivado, MySQL crea una tabla de tipo MyISAM. Este comportamiento es conveniente cuando quiere copiar tablas entre servidores MySQL que soportan distintos motores. (Por ejemplo, en una inicialización de

replicación, tal vez su maestro suporte un motor de almacenamiento transaccional para más seguridad, pero los esclavos usan un motor de almacenamiento no transaccional para mayor velocidad.)

La sustitución automática del tipo MyISAM cuando se especifica un tipo no especificado puede ser confuso para nuevos usuarios. En MySQL 5.0, se genera una advertencia cuando se cambia un tipo de tabla automáticamente.

MySQL siempre crea un fichero .frm para guardar la definición de tabla y columnas. El índice y datos de la tabla puede estar almacenado en uno o más ficheros, en función del tipo de tabla. El servidor crea el fichero .frm por encima del nivel de almacenamiento del motor. Los motores de almacenamiento individuales crean los ficheros adicionales necesarios para las tablas que administran.

Una base de datos puede contener tablas de distintos tipos.
Las tablas transaccionales (TSTs) tienen varias ventajas sobre las no transaccionales (NTSTs):

- Más seguras. Incluso si MySQL cae o tiene problemas de hardware, puede recuperar los datos, mediante recuperación automática o desde una copia de seguridad más el log de transacciones.

- Puede combinar varios comandos y aceptarlos todos al mismo tiempo con el comando COMMIT (si autocommit está desactivado).

- Puede ejecutar ROLLBACK para ignorar los cambios (si autocommit está desactivado).

- Si falla una actualización, todos los cambios se deshacen. (Con tablas no transaccionales, todos los cambios son permanentes.)

- Motores de almacenamiento transaccionales pueden proporcionar mejor concurrencia para tablas que tienen varias actualizaciones concurrentes con lecturas.

En MySQL 5.0, InnoDB usa valores de configuración por defecto si no los especifica. Consulte Sección 15.3, “Configuración de InnoDB”.
Tablas no transaccionales tienen varias ventajas al no tener una sobrecarga transaccional:

- Más rápidas
- Menor requerimiento de espacio.
- Menos memoria para actualizaciones

Puede combinar tablas transaccionales y no transaccionales en el mismo comando para obtener lo mejor de ambos mundos. Sin embargo, en una transaccion con autocommit desactivado, los cambios de tablas no transaccionales son permanentes inmediatamente y no pueden deshacerse.

WAMP es el acrónimo usado para describir un sistema de infraestructura de internet que usa las siguientes herramientas:

- Windows, como sistema operativo;
- Apache, como servidor web;
- MySQL, como gestor de bases de datos;

PHP (generalmente), Perl, o Python, como lenguajes de programación.
El uso de un WAMP permite servir páginas html a internet, además de poder gestionar datos en ellas, al mismo tiempo un WAMP, proporciona lenguajes de programación para desarrollar aplicaciones web.

LAMP es el sistema análogo que corre bajo ambiente Linux
MAMP es el sistema análogo que corre bajo ambiente Mac

2.4 HIPÓTESIS

2.4.1 Hipótesis General

La aplicación de un software permitirá gestionar información Financiera en el Escuela Alfa & Omega de la ciudad de Babahoyo.

2.4.2 Hipótesis Específicos

- Al utilizar el DBMS de MySql se podrá almacenar y administrar un alto volumen de información.

- Al estar disponible en una intranet permitirá acceso en los puntos de red conectados en el servidor.

- El diseño de las cuentas de usuarios garantiza la seguridad del sistema.

CAPITULO III

3. MARCO METODOLOGICO

3.1 Método

- Se utilizara el método científico para el desarrollo del conocimiento en relación al objeto de estudio.
- Además se empleara los métodos inductivos y deductivo porque se partirá de hechos conocidos para buscar soluciones y se tomara especial atención a los fúndamelos teóricos para formular una propuesta aplicable y valida.

3.2 Técnicas

Recopilar información para luego clasificarla, agruparla y finalmente presentar son:

- Entrevista por que es una técnica abierta para la obtención de información y de esta manera conseguir los datos más importantes para llevar a cabo esta investigación.
- Revisión documental para fundamentar la investigación y la propuesta del sistema.
- La experimentación al desarrollar del software para la propuesta.

3.3 Procedimientos

Análisis y Evaluación del Sistema

- Determinación de Requerimientos
- Definición de la frontera del Sistema

Diseño del Sistema

- Diseño de Base de Datos
- Diseño Procedimental
- Diseño Arquitectónico
- Diseño de las Entradas
- Diseño de salidas por pantalla
- Diseño de salidas por Impresora
- Diseño de Programas o Módulos
- Desarrollo e Implementación del Sistema
- Construcción y Prueba del Sistema
- Configuración del la Aplicación

3.4 Recursos Humanos

1 Investigadoras - 1 Director de Tesis. - 1 Profesor Asesor.

3.5 RECURSOS TÉCNICOS

HARDWARE

Cantidad	Característica
1 computador	procesador Pentium IV intel 3.2 ghz Disco duro 120gb Memoria ram DDR 512 mb, tarjeta de video 32 mb. Puerto paralelo, puerto serial, puerto usb, monitor Samsung "17", CD write 52x32x52 Fax modem 56 kpps Tarjeta red10/100 tarjeta de serial Mouse PS-2,teclado

3.6 Software

- WAMP5 - SQLyog - Macromedia Flash.8.0

3.7 Recursos Materiales

Memory Flash de 4GB

3.8 PRESUPUESTO Y COSTOS 60

1.1.1.1 Total	**1.1.1.2 Egresos**	
1.1.1.2.1 Autofinanciamiento (Recursos Propios) **Total $300**	**1.1.1.2.2 Viáticos** Movilizaciones 50	**50**
	1.1.1.2.3 Gastos de investigación Navegación en Internet 50 Compra de Libros 100 Compra de Revistas 50	**200**
	1.1.1.2.4 Papelería Resmas 4 25 Fotocopias 5	**30**
	Impresión: Valor de cartuchos 20	**20**
	Total	**$300**

3.9 Tabulación de Datos

INSTRUMENTO (ENCUESTA)

Dirigida a: Los usuarios de la escuela Alfa & Omega.

Objetivo: Identificar las necesidades de los usuarios, para mejorar el servicio que se brinda la escuela Alfa & Omega.

1) ¿En el departamento de tesorería existen personal que den excelentes reportes educativo y financiero?

☐ Si ☐ No

2) ¿Cree usted que se asignan horarios de atención y servicio al usuario de manera eficiente?

☐ Si ☐ No

3) ¿En el departamento de tesorería se guarda la información de manera manual?

☐ Si ☐ No

4) ¿En el departamento de tesorería existe material técnico disponible?

☐ Si ☐ No

5) Le gustaría que existiera un sistema Informático que facilite al usuario tener reporte educativo y financiero de manera rápida y verídica directamente al departamento de tesorería.

☐ Si ☐ No

CAPITULO IV

4. REQUERIMIENTOS Y FUNCIONES DEL SOFTWARE

4.1 CARACTERISTICAS DEL SISTEMA PROPUESTO

4.1.1 Confiabilidad

Con el sistema propuesto la información que se obtendrá a través de consultas y reportes es 100% real y confiable, ya que la seguridad de los datos es uno de los puntos relevantes del presente proyecto.

4.1.2 Amigable

Las aplicaciones tienen un diseño sencillo, fácil, de manejar, con opciones claras y ordenadas.

4.1.3 Seguridad

Los datos solo podrán ser manipulados por personal autorizado, que cumpla con todas las exigencias de seguridad, evitando que los datos sean modificados por personas no autorizadas.

4.1.4 Efectividad

Se pondrán obtener datos reales, confiables, rápidamente y sin mayor esfuerzo.

4.1.5 ESTRATEGIAS DE DESARROLLO

La aplicación Web se desarrollará en el lenguaje de programación PHP y código de HTML, con un manejador de Base de Datos MySQL y un Servidor de tecnología Wamp Server.

El sistema contendrá básicamente 2 módulos:

- Control de Matriculas, que es el control de Estudiantes
- Control Financiero, que se refiere al ingreso de depósitos de los estudiantes, de manera que se lleve un control sobre sus pagos.

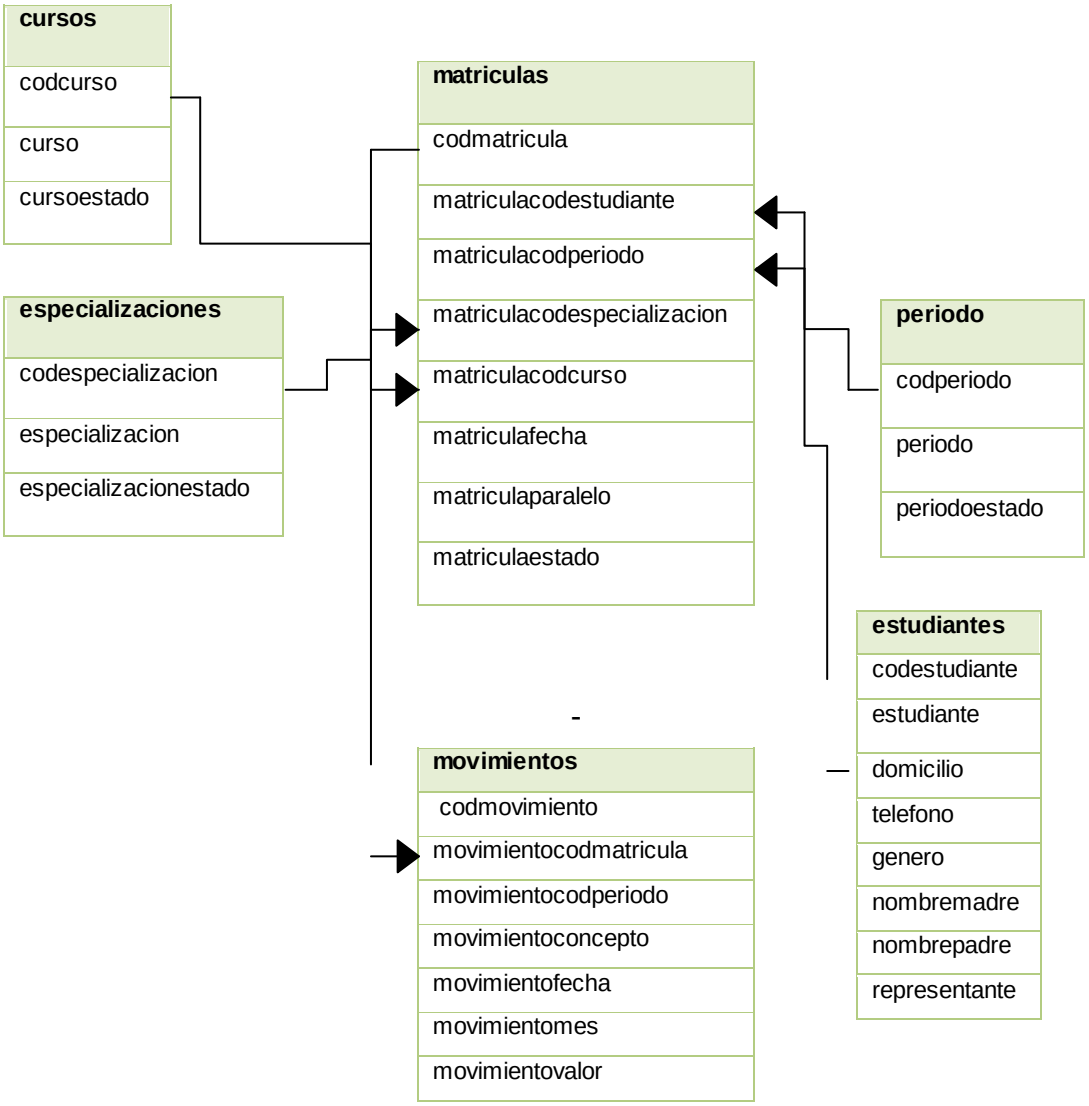
Mediante esta aplicación se creará una fácil interacción entre el programa y su operador, por las facilidades visuales y de adaptación con las que cuenta este programa. Un ambiente de trabajo amigable, de tal manera que cualquier persona con conocimientos básicos de computación pueda manipularlo.

Con una estructura de desarrollo pensada para expansiones futuras, de tal manera que en cualquier momento puede adicionársele nuevos módulos que permitan el control de otras áreas.

Existirá también el control de usuarios para garantizar la seguridad de la información del sistema, de tal manera que la Secretaria, el Director, tendrán su propia clave para ingresar al mismo.

4.1.5.1 BASE DE DATOS (MODELO CONCEPTUAL Y MODELO FÍSICO)

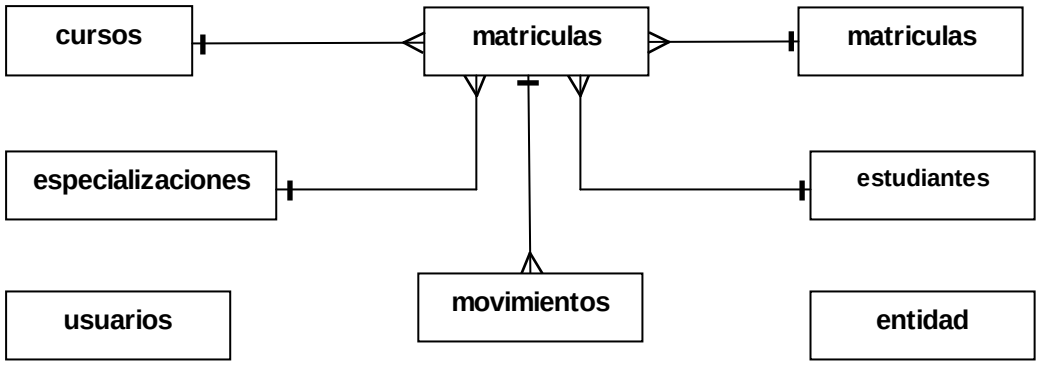
4.1.5.2 MODELO FÍSICO



usuarios
codusuario
Login
password
usuariotrato
usuario
usuariocargo
usuarioacceso
usuarioestado

entidad
entidad
siglas
titulo
lugar
dirección
teléfono
fax

4.1.5.3 MODELO CONCEPTUAL



4.1.5.4 DICCIONARIO DE DATOS

Tabla 1 curso:

Campo	Tipo	Null	Clave	Comentario
codcurso	varchar(2)	NO	PRI	Clave primaria
curso	varchar(50)	NO		Nombre de urso
cursoestado	varchar(8)	NO		Estado Activo/Inactivo

Tabla 2 entidad:

Campo	Tipo	Null	Clave	Comentario
entidad	varchar(200)	NO	PRI	Clave primaria
siglas	varchar(10)	NO		Iníciales de la Entidad
titulo	varchar(200)	NO		Titulo del Sistema
lugar	varchar(50)	NO		Lugar Ciudad/Provincia
dirección	varchar(150)	NO		Dirección
teléfono	varchar(20)	NO		N° de Teléfono
fax	varchar(20)	NO		N° de Fax

Tabla 3 especializaciones:

Campo	Tipo	Null	Clave	Comentario
codespecializacion	varchar(2)	NO	PRI	Clave primaria
especializacion	varchar(50)	NO		Nombre de la Especialización
especializacionestado	varchar(8)	NO		Estado Activo/Inactivo

Tabla 4 estudiantes:

Campo	Tipo	Null	Clave	Comentario
codestudiante	varchar(4)	NO	PRI	Clave primaria
estudiante	varchar(50)	NO		Nombre de estudiante
domicilio	varchar(50)	NO		Domicilio
telefono	varchar(30)	NO		N° Teléfono
genero	varchar(10)	NO		Genero
nombremadre	varchar(50)	NO		Nombre de la Madre
nombrepadre	varchar(50)	NO		Nombre del Padre
representante	varchar(50)	NO		Nombre del Representante

Tabla 5 matriculas:

Campo	Tipo	Null	Clave	Comentario
codmatricula	varchar(4)	NO	PRI	Clave primaria
matriculacodestudiante	varchar(50)	NO		Código de Estudiante
matriculacodperiodo	varchar(8)	NO		Código de Periodo
matriculacodespecializacion	varchar(2)	NO		Código de Especialización
matriculacodcurso	varchar(2)	NO		Código de Curso
matriculafecha	varchar(10)	NO		Fecha de Matriculación
matriculaparalelo	varchar(1)	NO		Letra de Paralelo
matriculaestado	varchar(8)	NO		Estado Activo/Inactivo

Tabla 6 movimientos:

Campo	Tipo	Null	Clave	Comentario
codmovimiento	varchar(7)	NO	PRI	Clave primaria
movimientocodmatricula	varchar(4)	NO		Numero de Matricula
movimientocodperiodo	varchar(3)	NO	MUL	Código de periodo
movimientoconcepto	varchar(2)	NO		Concepto de pago
movimientofecha	varchar(10)	NO		Fecha de ingreso
movimientomes	varchar(2)	NO		Mes del Pago
movimientovalor	varchar(8)	NO		Estado (Activo/Inactivo)

Tabla 7 Periodos:

Campo	Tipo	Null	Clave	Comentario
codperiodo	varchar(3)	NO	PRI	Clave primaria
periodo	varchar(50)	NO		Nombre de Periodos
periodoestado	varchar(8)	NO		Estado (Activo/Inactivo)

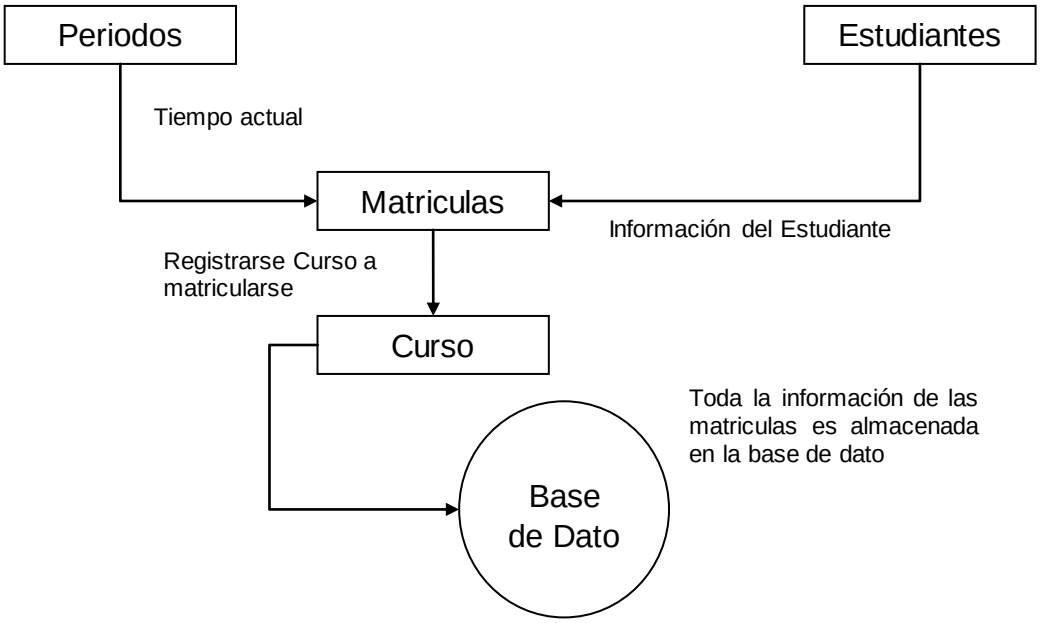
Tabla 8 Usuarios

Campo	Tipo	Null	Clave	Comentario
codusuario	Varchar(6)	NO	PRI	Clave primaria
Login	varchar(10)	NO		Alias del Usuario
password	varchar(50)	NO		Contraseña de Usuario
usuariotrato	varchar(20)	NO		Trato del Usuario
usuario	varchar(20)	NO		Nombre de Usuario
usuariocargo	varchar(20)	NO		Cargo del Usuario
usuarioacceso	varchar(20)	NO		Seguridad de Acceso
usuarioestado	varchar(8)	NO		Estado (Activo/Inactivo)

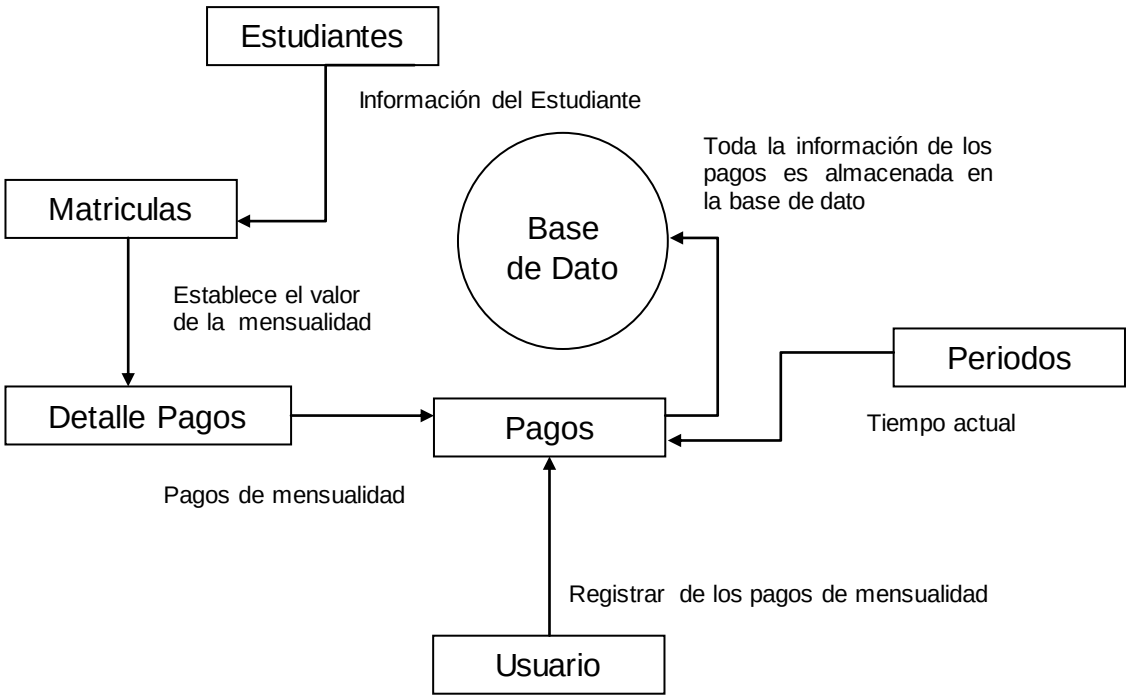
4.2 FLUJO DE INFORMACIÓN DEL SISTEMA PROPUESTO

A continuación se describirá como fluye la información entre las entidades del sistema usando los diagramas de contexto.

4.2.1 Diagrama de contexto nivel de Matriculación



4.2.2 Diagrama de contexto nivel Financiero



4.2.3 SCRIPT DE LA BASE DE DATOS

```
/*
SQLyog - Free MySQL GUI v5.11
Host - 5.0.18-nt : Database - acropolis
*********************************************************************
Server version : 5.0.18-nt
*/

SET NAMES utf8;

SET SQL_MODE='';

create database if not exists `acropolis`;

USE `acropolis`;

/*Table structure for table `cursos` */

DROP TABLE IF EXISTS `cursos`;

CREATE TABLE `cursos` (
  `codcurso` varchar(2) NOT NULL,
  `curso` varchar(30) NOT NULL,
  `cursovalor` varchar(8) NOT NULL,
  `cursoestado` varchar(8) NOT NULL,
  PRIMARY KEY (`codcurso`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

/*Data for the table `cursos` */

insert into `cursos` (`codcurso`,`curso`,`cursovalor`,`cursoestado`) values
('01','Octavo','30.00','Activo'),('02','Noveno','30.00','Activo'),('03','Décimo','30.00','Activo')
,('04','Primer','30.00','Activo'),('05','Segundo','30.00','Activo'),('06','Tercero','30.00','Activ
o');
/*Table structure for table `entidad` */
```

```
DROP TABLE IF EXISTS `entidad`;

CREATE TABLE `entidad` (
  `entidad` varchar(200) NOT NULL,
  `titulo` varchar(150) NOT NULL,
  `siglas` varchar(10) NOT NULL,
  `direccion` varchar(150) NOT NULL,
  `lugar` varchar(50) NOT NULL,
  `telefono` varchar(20) NOT NULL,
  `fax` varchar(20) NOT NULL,
  `email` varchar(50) NOT NULL,
  PRIMARY KEY (`entidad`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

/*Data for the table `entidad` */

insert into `entidad` (`entidad`,`titulo`,`siglas`,`direccion`,`lugar`,`telefono`,`fax`,`email`)
values ('CENTRO EDUCATIVO CRISTIANO \"ALFA Y OMEGA\"','Sistema de Control
Pagos de Mensualidades','A & O','Av. Universitaria Km. 1 ½. via Flores','Babahoyo -
Ecuador','052735269','','alfayomega@hotmail.com');

/*Table structure for table `especializaciones` */

DROP TABLE IF EXISTS `especializaciones`;

CREATE TABLE `especializaciones` (
  `codespecializacion` varchar(2) NOT NULL,
  `especializacion` varchar(100) NOT NULL,
  `especializacionestado` varchar(8) NOT NULL,
  PRIMARY KEY (`codespecializacion`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

/*Data for the table `especializaciones` */

insert into `especializaciones`
(`codespecializacion`,`especializacion`,`especializacionestado`) values ('01','Educación
```

```
General Básica','Activo');

/*Table structure for table `estudiantes` */

DROP TABLE IF EXISTS `estudiantes`;

CREATE TABLE `estudiantes` (
  `codestudiante` varchar(6) NOT NULL,
  `estudiante` varchar(50) NOT NULL,
  `genero` varchar(9) NOT NULL,
  `domicilio` varchar(50) NOT NULL,
  `telefono` varchar(20) NOT NULL,
  `nombremadre` varchar(50) NOT NULL,
  `nombrepadre` varchar(50) NOT NULL,
  `representante` varchar(50) NOT NULL,
  PRIMARY KEY (`codestudiante`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

/*Data for the table `estudiantes` */

/*Table structure for table `matriculas` */

DROP TABLE IF EXISTS `matriculas`;

CREATE TABLE `matriculas` (
  `codmatricula` varchar(4) NOT NULL,
  `matriculacodestudiante` varchar(6) NOT NULL,
  `matriculacodperiodo` varchar(3) NOT NULL,
  `matriculacodespecializacion` varchar(2) NOT NULL,
  `matriculacodcurso` varchar(2) NOT NULL,
  `matriculafecha` varchar(10) NOT NULL,
  `matriculaseccion` varchar(15) NOT NULL,
  `matriculaparalelo` varchar(1) NOT NULL,
  `matriculaestado` varchar(14) NOT NULL,
  KEY `FK_matriculas` (`matriculacodperiodo`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
/*Data for the table `matriculas` */

/*Table structure for table `movimientos` */

DROP TABLE IF EXISTS `movimientos`;

CREATE TABLE `movimientos` (
  `codmovimiento` varchar(7) NOT NULL,
  `movimientocodmatricula` varchar(4) NOT NULL,
  `movimientocodperiodo` varchar(3) NOT NULL,
  `movimientoconcepto` varchar(50) NOT NULL,
  `movimientofecha` varchar(10) NOT NULL,
  `movimientomes` varchar(2) NOT NULL,
  `movimientovalor` varchar(10) NOT NULL,
  PRIMARY KEY (`codmovimiento`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

/*Data for the table `movimientos` */

insert into `movimientos`
(`codmovimiento`,`movimientocodmatricula`,`movimientocodperiodo`,`movimientoconc
epto`,`movimientofecha`,`movimientomes`,`movimientovalor`) values
('0000001','0001','001','Mensualidad','28/06/2012','01','30'),('0000002','0001','001','Abon
o1','28/06/2012','02','15'),('0000003','0001','001','Abono2','28/06/2012','02','15'),('00000
04','0002','001','Mensualidad','28/06/2012','01','30.00'),('0000005','0002','001','Abono1','
28/06/2012','02','20'),('0000006','0002','001','Abono2','28/06/2012','02','5'),('0000007','0
002','001','Mensualidad','28/06/2012','03','35'),('0000008','0004','001','Mensualidad','28/
06/2012','01','30'),('0000009','0003','001','Abono1','28/06/2012','01','15'),('0000010','000
1','001','Mensualidad','28/06/2012','03','30'),('0000011','0560','001','Mensualidad','18/07/
2012','01','30'),('0000012','0560','001','Mensualidad','18/07/2012','02','25'),('0000013','0
560','001','Mensualidad','18/07/2012','03','35');

/*Table structure for table `periodos` */

DROP TABLE IF EXISTS `periodos`;
```

```
CREATE TABLE `periodos` (
  `codperiodo` varchar(3) NOT NULL,
  `periodo` varchar(50) NOT NULL,
  `periodoestado` varchar(8) NOT NULL,
  PRIMARY KEY (`codperiodo`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

/*Data for the table `periodos` */

insert into `periodos` (`codperiodo`,`periodo`,`periodoestado`) values ('001','2012 -
2013','Activo');

/*Table structure for table `usuarios` */

DROP TABLE IF EXISTS `usuarios`;

CREATE TABLE `usuarios` (
  `codusuario` varchar(4) NOT NULL,
  `usuariocedula` varchar(10) NOT NULL,
  `login` varchar(20) NOT NULL,
  `password` varchar(20) NOT NULL,
  `usuariotrato` varchar(8) NOT NULL,
  `usuario` varchar(50) NOT NULL,
  `usuariocargo` varchar(50) NOT NULL,
  `usuarioacceso` varchar(8) NOT NULL,
  `usuarioestado` varchar(8) NOT NULL,
  PRIMARY KEY (`codusuario`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

/*Data for the table `usuarios` */

insert into `usuarios`
(`codusuario`,`usuariocedula`,`login`,`password`,`usuariotrato`,`usuario`,`usuariocargo`
,`usuarioacceso`,`usuarioestado`) values
('01','1204632556','Karina','Karina','Sra.','Karina Basilio
Silva','Técnico','Inactivo','Activo'),('02','','admin','admin','','Administrador','Administrador',
```

'Activo','Activo'),('03','','asistente','asistente','','Asistente','Asistente','Activo','Activo');

4.2.4 Diseño Arquitectónico.

Las necesidades actuales que tiene toda organización, institución para el logro de sus objetivos, demandan la construcción de sistemas de software que requieren de la combinación de diferentes tecnologías y plataformas de hardware y software para alcanzar un funcionamiento acorde con dichas necesidades, es por estas razones que hemos decidido realizar este sistema.

Todo software tiene una serie de requisitos comunes desde el punto de vista de las personas encargadas de su creación o mantenimiento

- Facilidad para gestionar su contenido.
- Facilidad para mantener una apariencia consistente

Nuestro software presenta las mejores facilidades de entendimiento en el diseño a presentar al usuario como por ejemplo: Ingresar los datos los estudiantes para su debido registro en la cual esta pantalla se encuentra validada para en que las cajas de texto que sean de solo número o letras solo dejen ingresar lo que corresponde.

Consultar y reportes de Estudiantes, Matriculas y Pagos realizados en el Escuela Alfa & Omega de la ciudad de Babahoyo.

Los eventos que se utilizan para realizar alguna consulta o ingreso en el sistema es a través del evento clic del mouse, El sitio web incluyendo el código PHP, también código HTML. Los contenidos dinámicos son los que se obtienen cada vez actualizamos o agregamos registros.

4.2.5 Descripción General las opciones y los módulos del Sistema

Sistema

- Unidad Educativa
- Perdidos
- Usuarios

- Cambiar usuario
- Cerrar Sesión

Administrar

- Cursos
- Especializaciones
- Estudiantes
- Matriculas
- Crear Pagos

Reportes

- Reporte de Estudiantes
- Reporte de Matriculas
- Reporte de Pagos por Mes

Recomendaciones

- Realizar actualizaciones periódicamente de la base de dato del servidor en la Web de las ofertas académicas que ofrece la Universidad Técnica de Babahoyo.

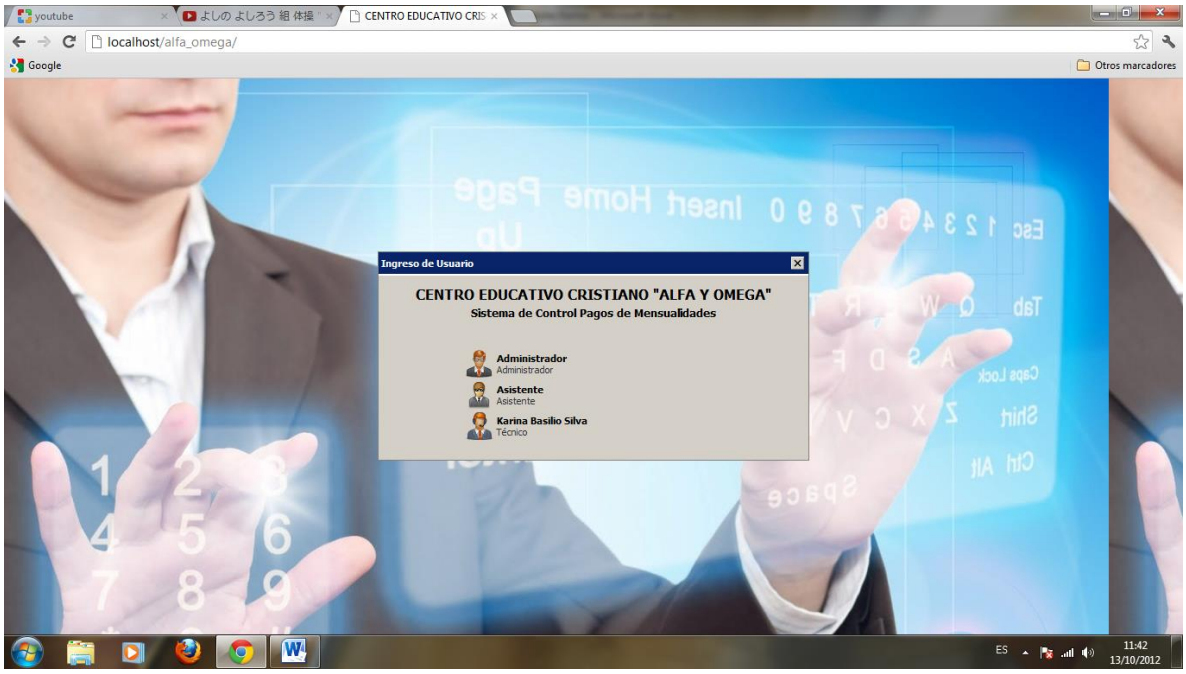
- Que el personal encargado de implementar el sistema tenga una completa capacitación del sistema para un óptimo manejo de todas las aplicaciones

- Para sistemas basadas en la Web se recomienda el uso de la plataforma que sean compatible con MySQL y PHP.

CAPITULO V

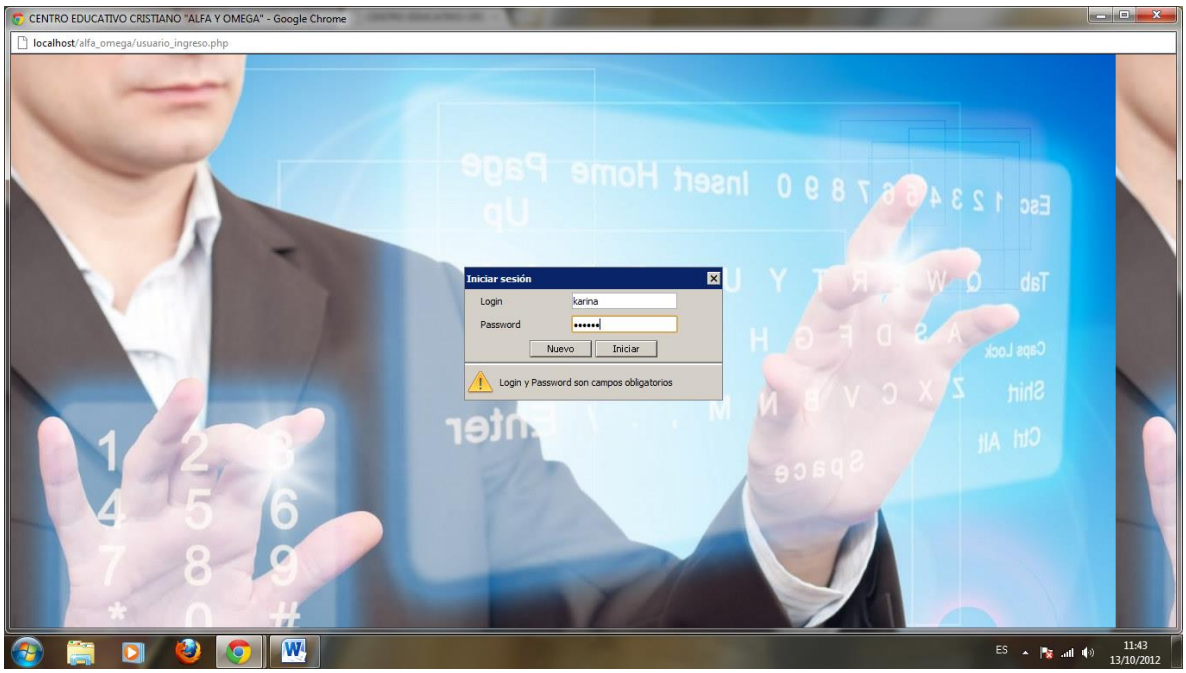
5 MANUAL DEL ADMINISTRADOR

5.1PANTALLA DE INICIO DEL SISTEMA



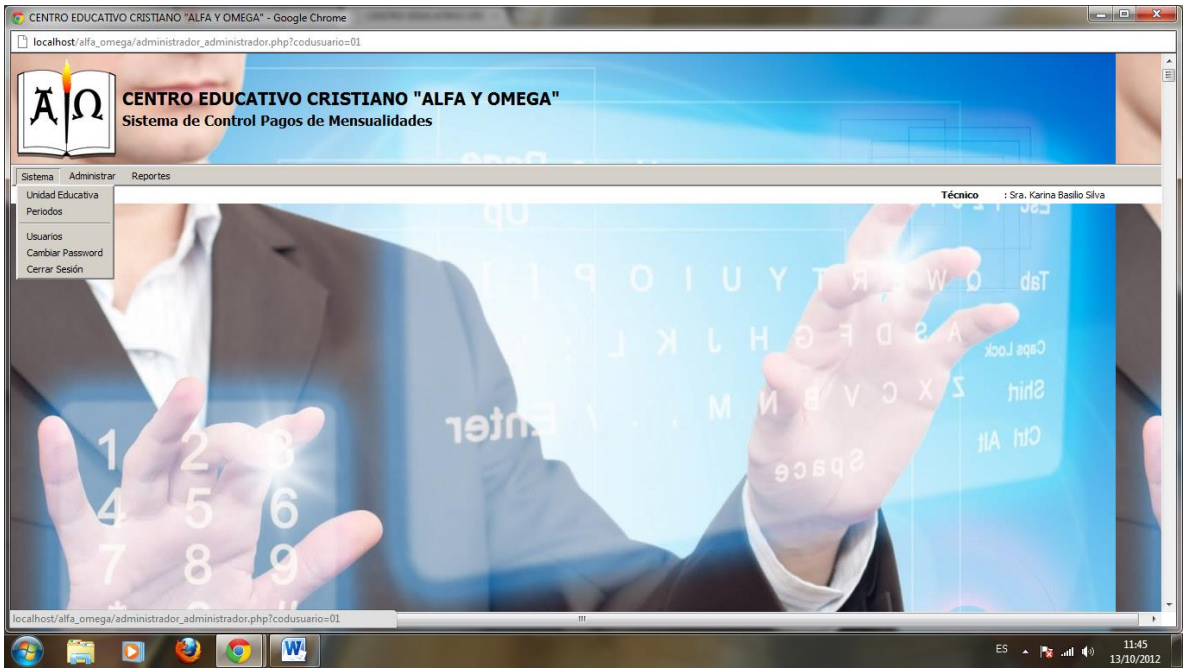
Permite escoger la opción para administrador, asistente y técnico.

5.2 PANTALLA DE INICIO DE SESION



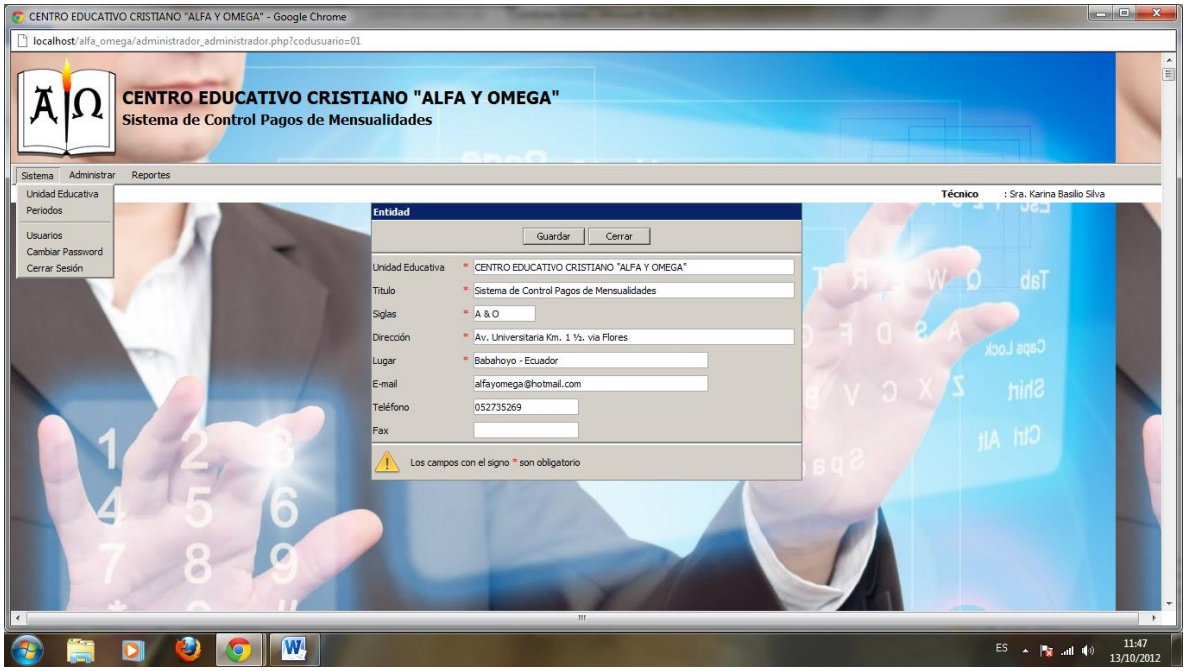
Permite ingresar el usuario y clave.

5.3 PANTALLA SISTEMA.



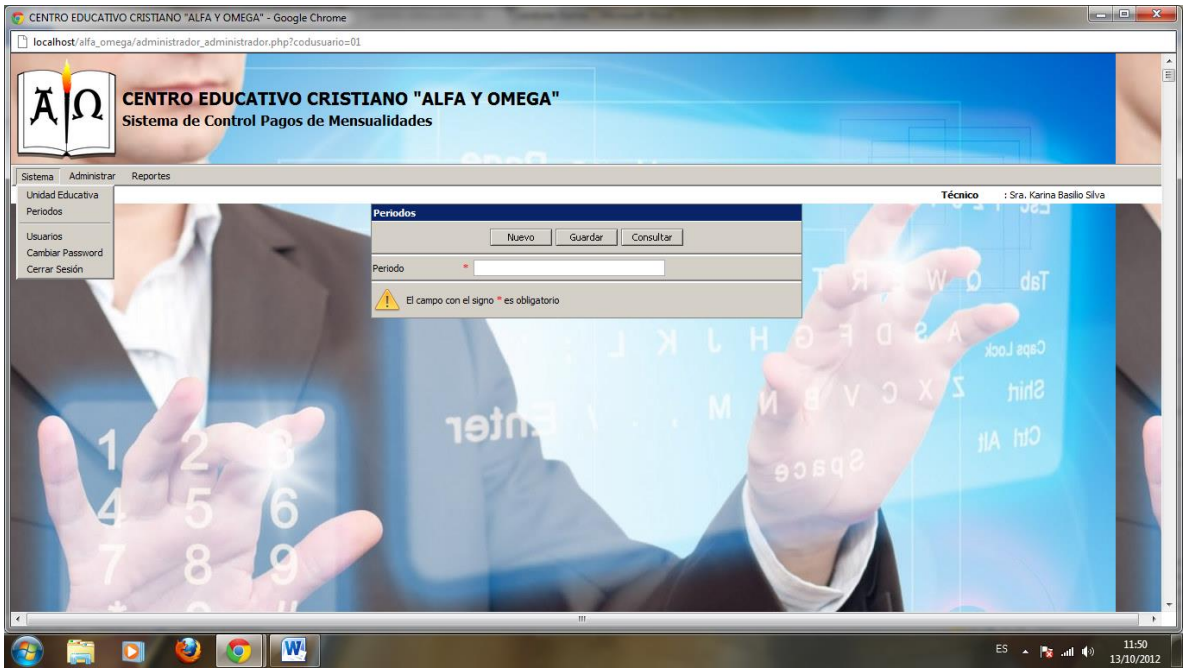
Permite escoger las opciones que se encuentran dentro del icono sistema del administrador.

5.3.1 Pantalla de Unidad Educativa.



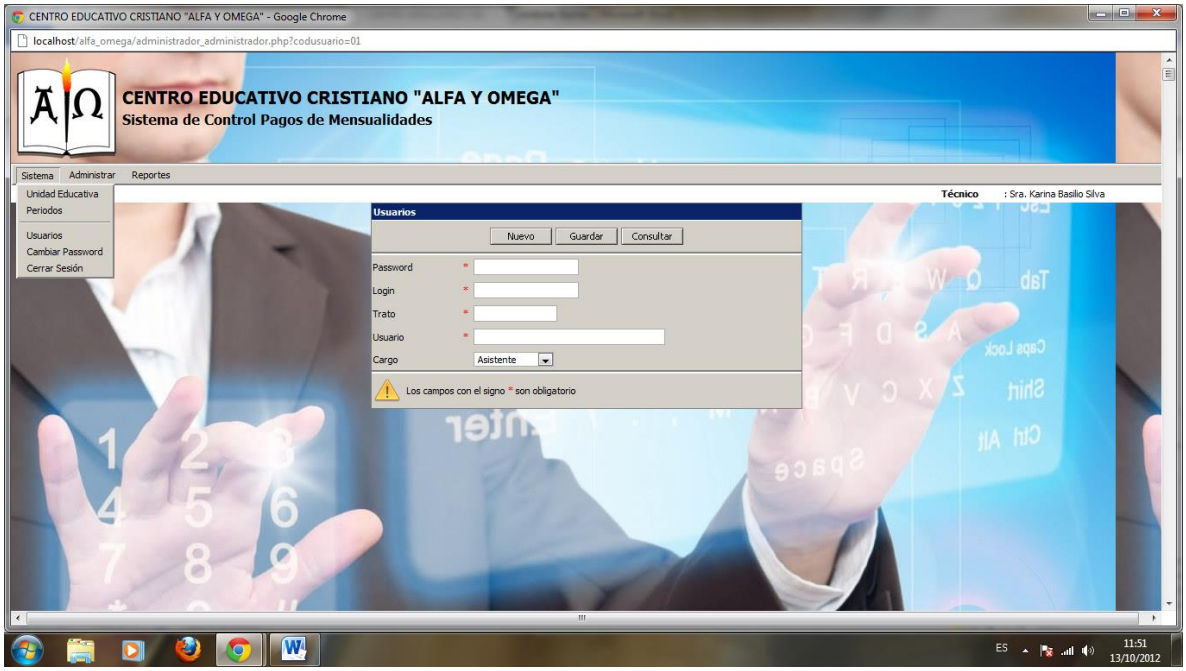
Permite ingresar los datos de la institución.

5.3.2 PANTALLA PERIODO.



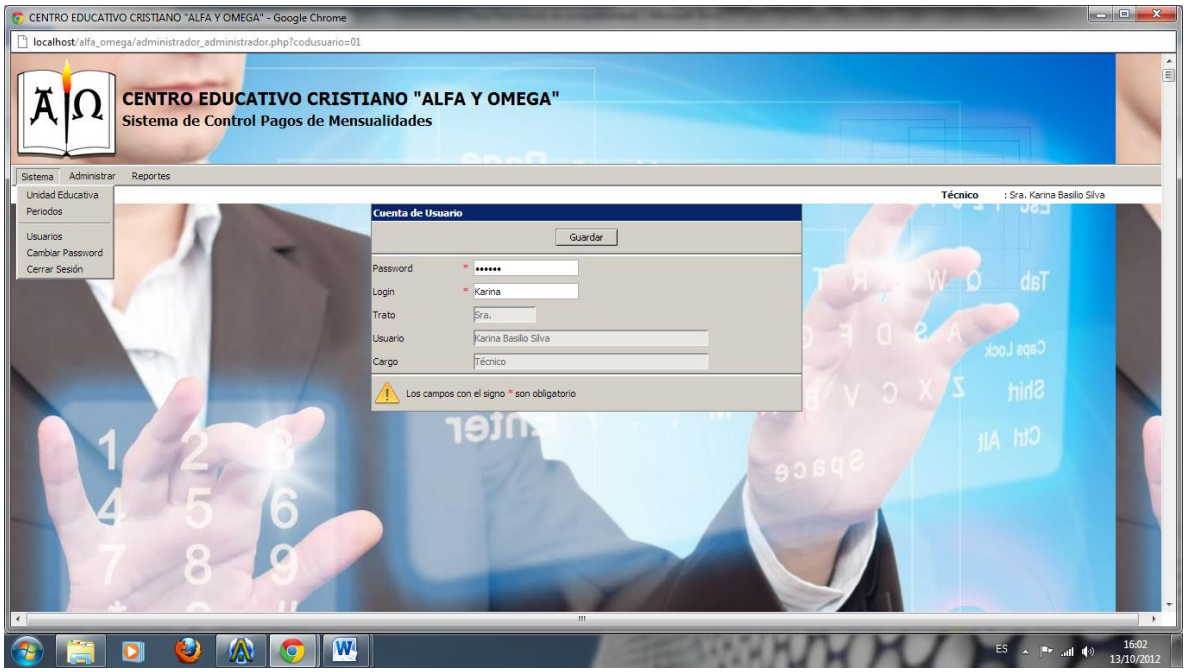
Permite ingresar el periodo lectivo.

5.3.3 PANTALLA USUARIOS.



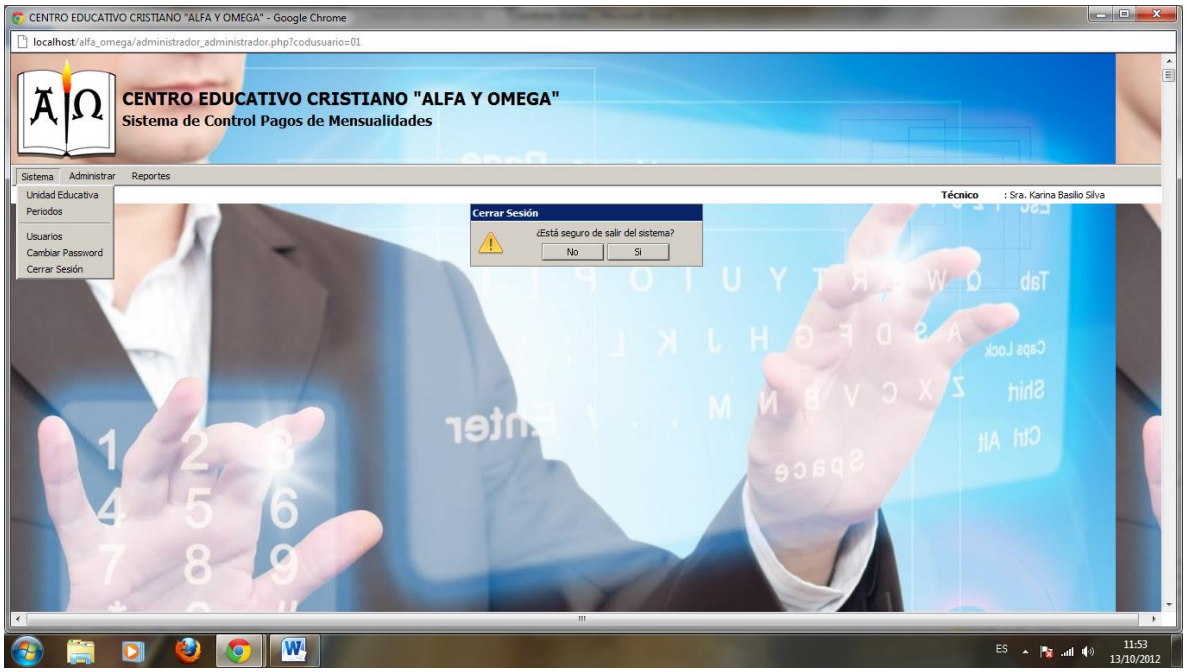
Permite ingresar password,login,cargo,usuario y titulo del administrador.

5.3.4 PANTALLA CAMBIAR EL PASSWORD.



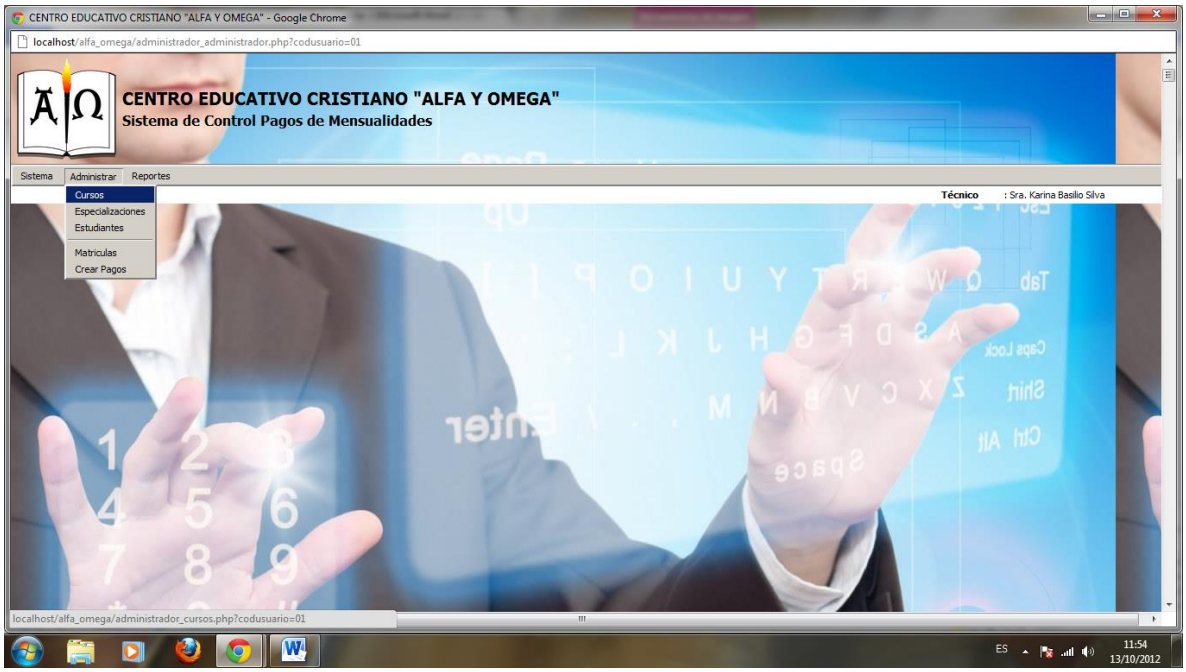
Permite cambiar el password del suario

5.3.5 PANTALLA CERRAR SECION.



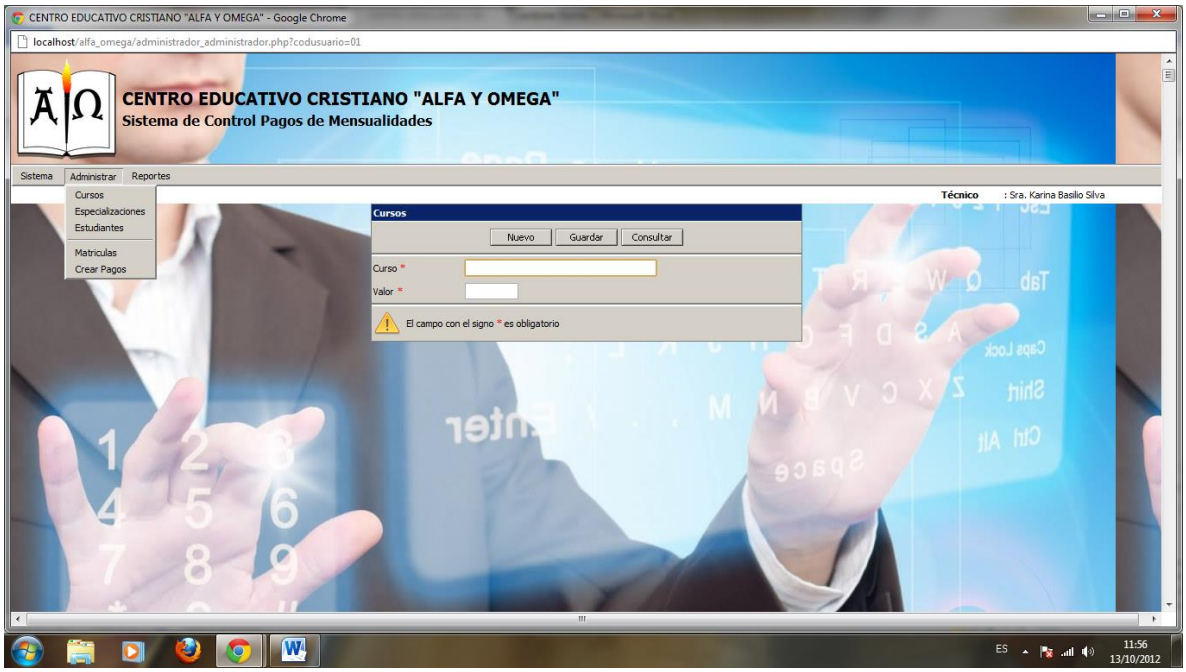
Permite salir del programa.

5.4 PANTALLA ADMINISTRAR



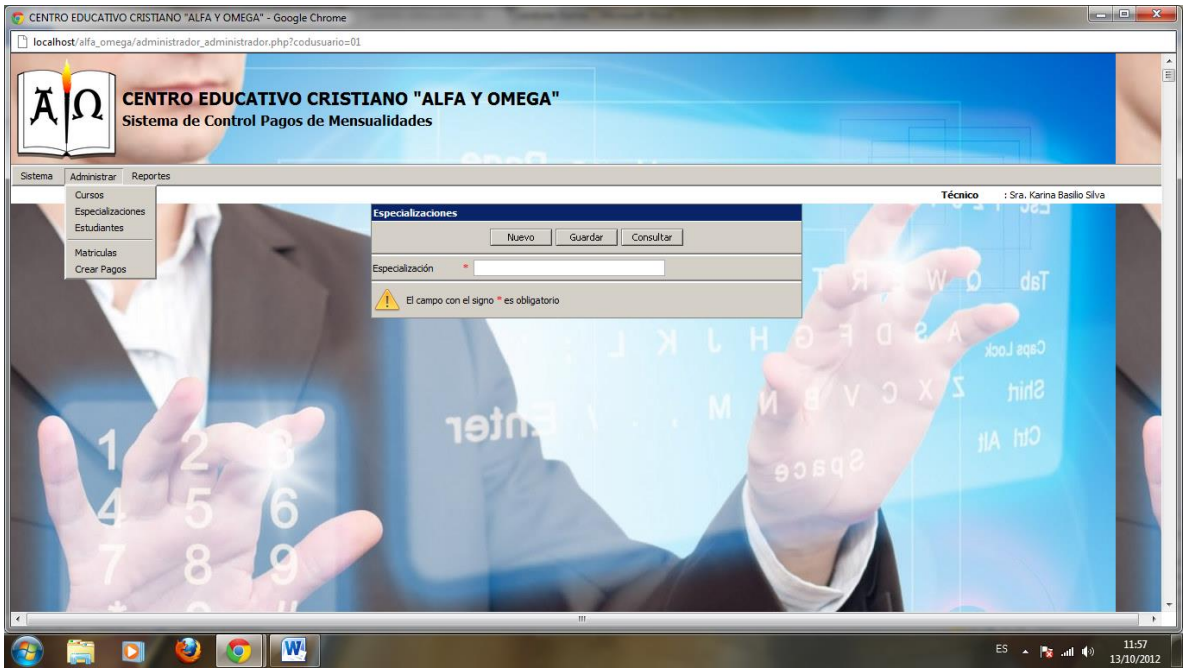
Permite escoger las opciones que se encuentran dentro del icono administrar del administrador.

5.4.1 PANTALLA CURSOS.



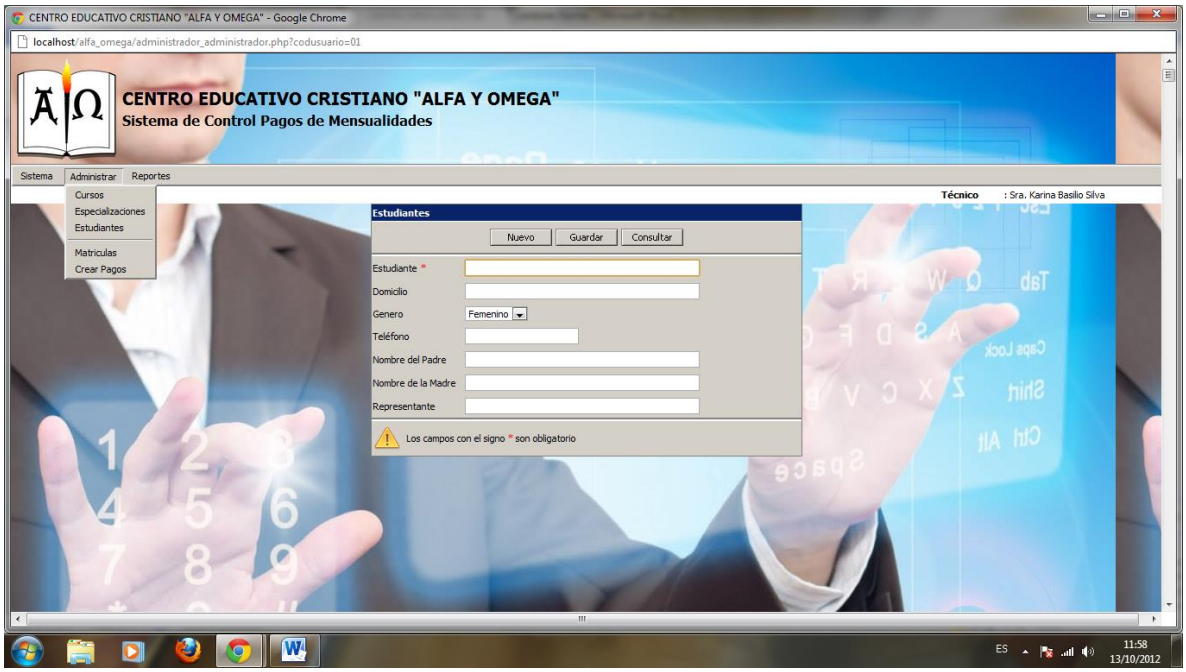
Permite ingresar al curso que va el alumno y el valor que le corresponde pagar.

5.4.2 PANTALLA ESPECIALIZACIONES.



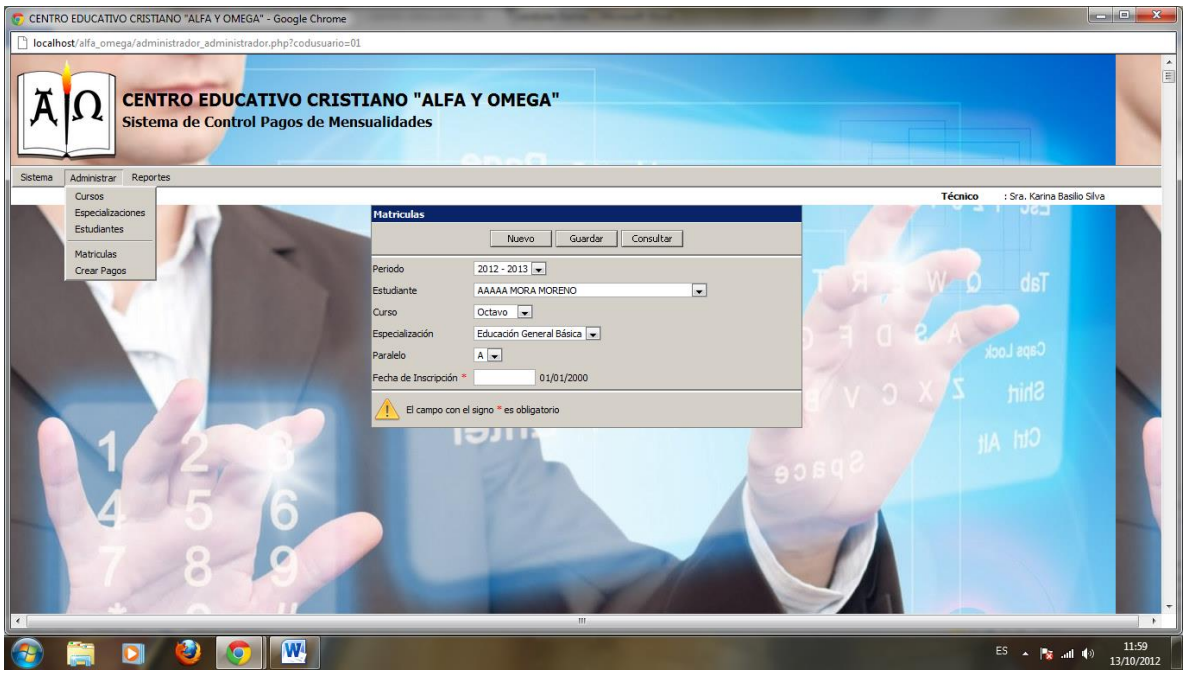
Permite ingresar la especializacion del alumno ingresado.

5.4.3 PANTALLA ESTUDIANTES.



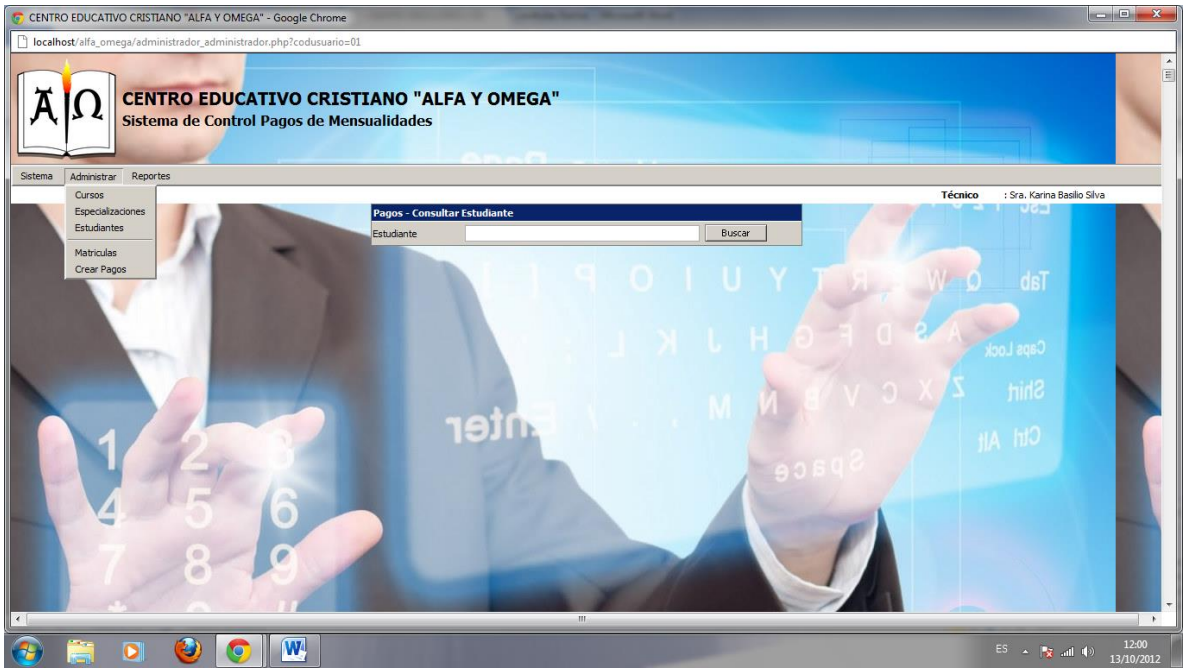
Permte ingresar los datos del estudiante.

5.4.4 PANTALLA MATRICULA



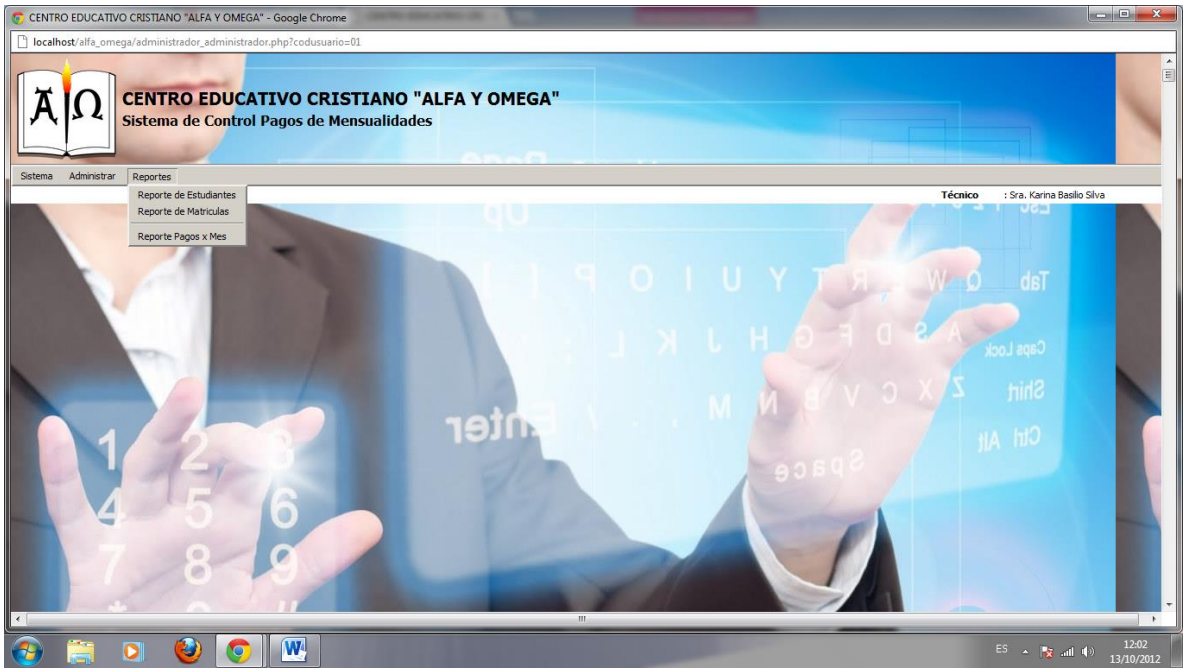
Permmite ingresar la fecha de la matricula del alumno y el paralelo que va.

5.4.5 PANTALLA CREAR PAGOS.



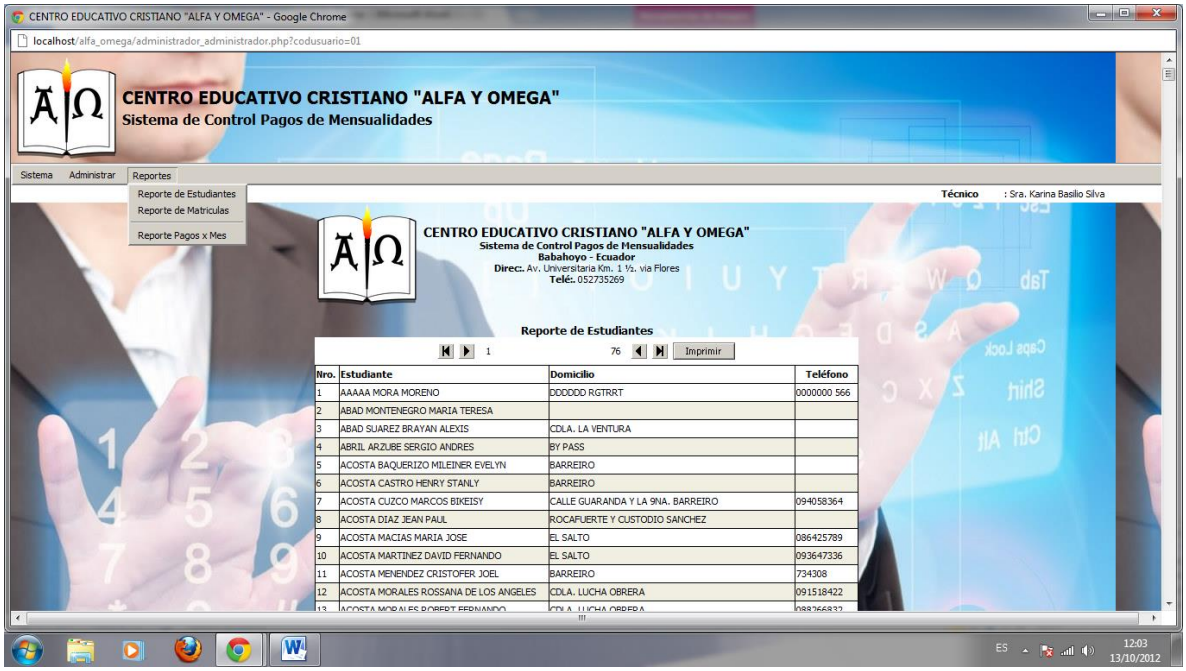
Permite ingresar los pagos mensual y abonos .

5.5 PANTALLA REPORTES.



Permite visualizar las clases de reporte que se pueden emitir.

5.5.1 PANTALLA REPORTE DE ESTUDIANTE.

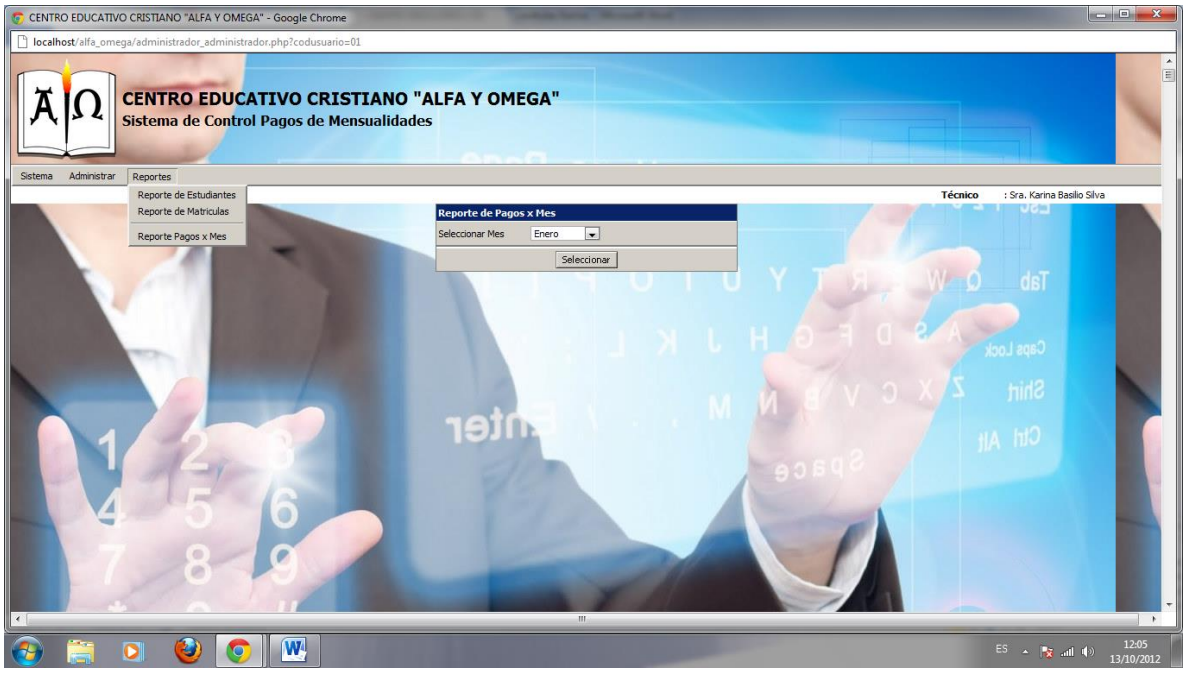


Permite emitir los listados por curso y paralelo de los alumnos.

5.5.2 PANTALLA REPORTE MATRICULA.

Permite emitir los listados con numero de matricula y paralelo

5.5.3 PANTALLA REPORTE PAGO POR MES



Permite

Emitir los listados de pagos por mes y si estan al dia o adeudan.

Conclusiones

Los datos implementados en esta tesis generan información de tipo académica y financiera permitiendo la elaboración de reportes detallados que establezcan un control interno del departamento de tesorería en la escuela Alfa & Omega y los pagos relacionados de cada estudiante.

La información de los reportes se presenta de una manera más eficientes, comprensible y amigable para que el usuario encargado de manejar el software obtenga un mejor resultado. El desarrollo de este tema concluimos que es posible presentar información académica y financiera el cual es posible integrar los resultados en un informe final.

RECOMENDACIONES

- Realizar respaldos mensuales de la base de dato, de tal manera de que siempre tenga los recursos necesarios para actualizaciones en caso de pérdida de información.

- Que el personal encargado de utilizar el software tenga una completa capacitación del sistema para un óptimo manejo de todas las aplicaciones.

- Utilizar sistemas operativos actuales como plataforma Windows 7 que es compatible a MySQL y PHP.

- Ingresar diariamente los movimientos de inventarios para que en todo momento este actualizado el stock de los artículos de tesoreria.

Cronograma

	Febrero	Marzo	Abril	Mayo	Junio	Julio
Proyecto						
Factibilidad						
Definir Entidades						
Creación Base de Datos						
Desarrollo del Sistema						
Diseño de Pantalla						
Pantallas de Entrada						
Pantallas de Salida						
Validaciones						
Pruebas y Depuración de Errores						
Defensa de tesis						

Bibliografía

BIBLIOGRAFÍA GENERAL

- Microsoft Conmerce Solutions
- Protocolos de Red (TCP/IP) Andrew Tanebaum
- Dynamic HTML Reference and Software Development Kit
- Análisis y Diseño de sistemas de Información. Autor (James A. Senn.)

BIBLIOGRAFIA REALACIONADA AL TEMA

- SQLyog
- PHP con base de datos

LINCOGRAFÍA

- http://www.saulo.net/pub/tcpip/
- http://html.rincondelvago.com/concepto-de-base-de-datos.html
- http://ola.icmyl.unam.mx/biblio/Tesis-Bus.php
- http://www.webestilo.com/foros/mensaje.php
- http://www.programacion.com/foros/5/msg/25774/
- http://www.fvet.uba.ar/biblioteca/como_hacer.htm
- http://www.programacion.com/php/
- http://www.br.uipr.edu/caiweb/default2.asp?tree=625
- http://www.asesoriatesis.com/cgi-bin/default.php

ANEXOS

Interpretación de los resultados de la aplicación de entrevista a los usuarios de la escuela Alfa & Omega.

A la muestra de 325 personas se le realizo el siguiente análisis detallado a continuación con tablas y gráficos para una mejor representación y análisis

1. **¿ En el departamento de tesorería existen personal que den excelentes reportes educativo y financiero?**

CUADRO 1

	Frecuencia	Porcentanje
SI	20	20%
NO	70	80%
TOTAL	90	100%

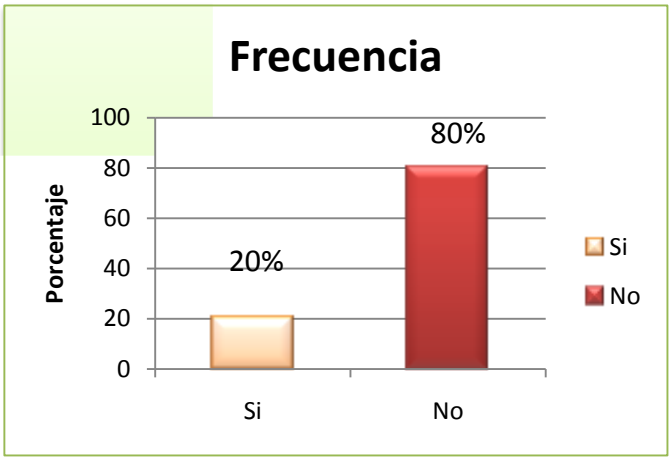


GRÁFICO 1

Con el resultado que se obtuvo podemos decir que un porcentaje mínimo de usuarios opinaron a favor, esto puede ser por apoyar a sus compañeros de trabajo o están de acuerdo con el servicio que ellos brindan, mientras que el otro grupo opina lo contrario, es decir el servicio de reportes no es el mejor, quizás no dependa del personal pues hay muchos factores que afectan la eficiencia del personal administrativo de tesorería para brindar un excelente reporte.

2. **¿Cree usted que se asignan horarios de atención y servicio al usuario de manera eficiente?**

CUADRO 2

	Frecuencia	Porcentaje
SI	35	30%
NO	55	70%
TOTAL	90	100%

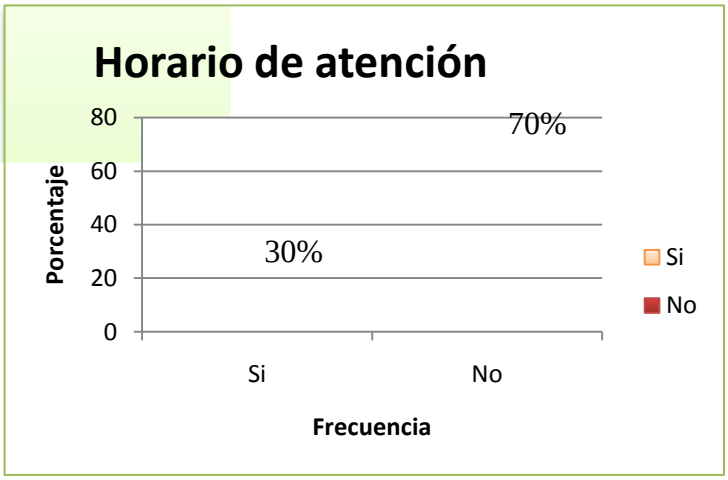


GRÁFICO 2

Con respecto al análisis de los porcentajes, un grupo mínimo opino a favor, por ellos creen que la asignación de horarios que se lleva hasta ahora es la mejor, mientras que por el otro 30 **%** podemos decir que no se designan horarios de atención de una manera adecuada, quizás porque el departamento de tesorería no cuenta con un sistema que le facilite realizar esto y es por eso que el personal no tienen una coordinación para brindar el servicio de manera eficiente.

3. **¿En el departamento de tesorería se guarda la información de manera manual?**

CUADRO 3

	Frecuencia	Porcentaje
SI	70	75%
NO	20	25%
TOTAL	90	100%

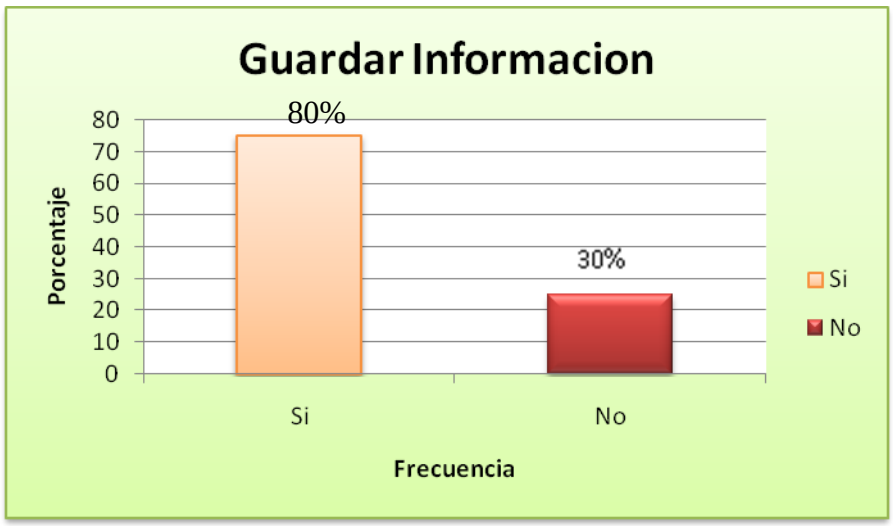


GRÁFICO 3

En el análisis que se realizó, se concluyo en que el grupo menor cree que no se guarda la información de forma manual, esto puede ser por apoyar al personal del departamento de tesorería. Pero con respecto al 70% de los encuestados nos dimos cuenta que dicho departamento aun no cuenta con la suficiente tecnología para almacenar sus registros de una manera actualizada.

4. ¿ En el departamento de tesorería existe material técnico disponible?

CUADRO 4

	Frecuencia	Porcentaje
SI	80	80%
NO	20	20%
TOTAL	100	100%

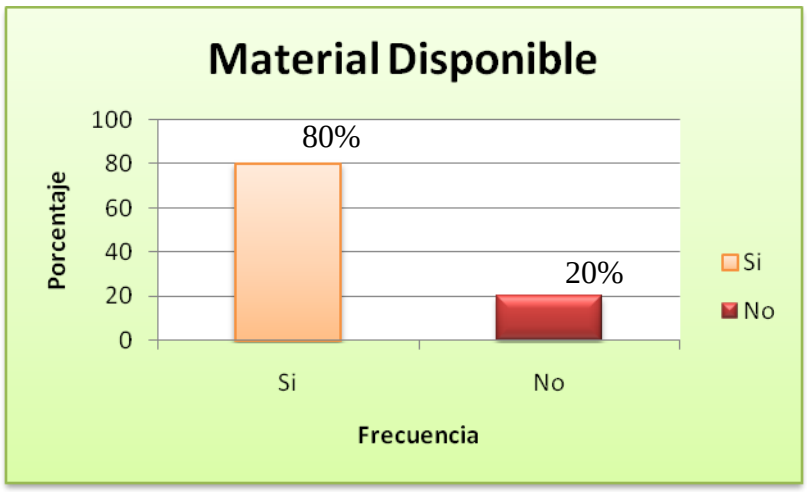


GRÁFICO 4

Con respecto a los resultados obtenidos, observamos que el 80% de los encuestados reconocen que si tienen materiales disponibles y esto nos permite decir que el departamento de tesorería si cuenta con material técnico disponible por lo cual se descarta que este sea un motivo por lo que el servicio de reportes no es eficiente.

5 ¿ Le gustaría que existiera un sistema Informático que facilite al usuario tener reporte educativo y financiero de manera rápida y verídica directamente al departamento de tesorería. ?

CUADRO 5

	Frecuencia	Porcentaje
SI	90	90%
NO	10	10%
TOTAL	100	100%

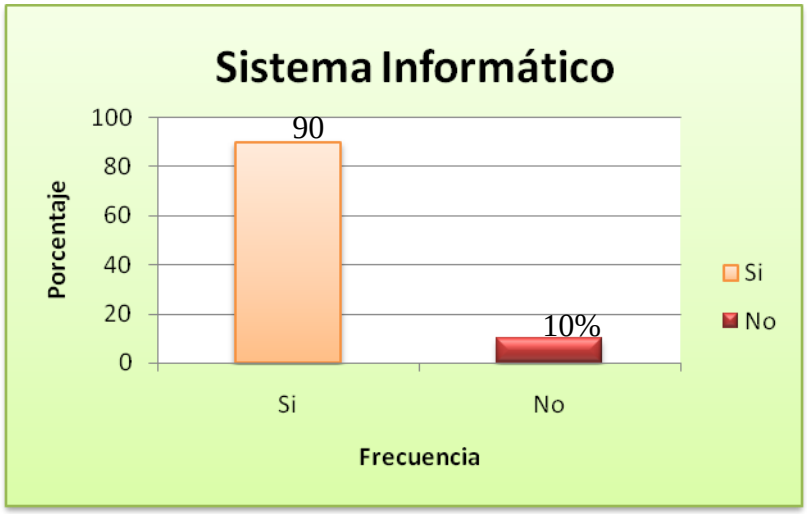


GRAFICO 5

Con respecto a los encuestados la mayor parte, considera como una alternativa que existiese un sistema Informático, donde se le facilite al usuario hacer su solicitud de mantenimiento directamente al departamento de tesorería, pero el otro grupo menor, considera que no es necesario, quizás porque ellos no necesita un contacto directo o están satisfechos con la eficiencia del personal de tesorería.