UNIVERSIDAD NACIONAL AUTONOMA DE MÉXICO

FACULTAD DE INGENIERÍA

INGENIERÍA EN COMPUTACIÓN

SISTEMA DE EVALUACIÓN AUTOMATIZADA EN ALMACENAJE POR BASE DE DATOS PARA PERFIL DEL ALUMNO

T E S I S

PARA OBTENER EL TÍTULO DE:
INGENIERO EN COMPUTACIÓN

P R E S E N T A:

JULIO CÉSAR ÁLVAREZ CASTILLO

DIRECTOR DE TESIS:
ING. JUAN FERNANDO SOLÓRZANO PALOMARES

MÉXICO D.F. **2013**

Agradecimientos y dedicatorias…

A mis papás. A mi mamá por siempre apoyarme con su enorme cariño y compresión, por cuidarme y alentarme cuando perdía el ánimo. A mi papá por siempre creer en mí, hacerme ver que tengo siempre mucho más potencial del que yo mismo podría darme cuenta.

A mi hermano porque siempre me demostró que la ingeniería no cualquiera la estudia, pero es una de las más grandes satisfacciones poder decir yo estudié ingeniería, yo soy ingeniero.

A mis profesores de la carrera, porque, los pocos malos, y la mayoría muy buenos que me encontré me demostraron como la soberbia, o la conformidad no se permiten en esta profesión, porque uno siempre tiene que buscar ser excelente en su trabajo y la vida, a darse a valorar mediante el trabajo realizado, que el ingeniero cuando hace algo mal se exhibe, y cuando se hace bien, a veces pasar desapercibido, pero siempre es muy valorado por el que realmente sabe el trabajo que implicó la solución.

A mi director de tesis que también fue mi profesor en la facultad, porque tuvo que ver más de lo que se imagina, para decidir trabajar en una de las áreas que actualmente laboro con muchísimo gusto.

A mis profesores en general porque me dieron una formación integral, la cual me ayudó a ser el ingeniero que soy el día de hoy, al profesor Alejandro Flores, Octavio Pérez, Teresa Nieves y Ernesto Olivares por mencionar a algunos de ellos.

A mis amigos Salatiel, Damián, Oscar, Jacobo, Joshua, Patricia y Anabel por ser como una segunda familia de hermanos siempre apoyándonos unos con otros aún en carreras y facultades distintas, viviendo lo bueno y lo malo siempre al pendiente de podernos ayudar.

A mis amigos de la facultad Fer, Carmina y Rafa, porque ya en la carrera encontrar un amigo verdadero y sincero comienza a complicarse muchas veces por competencia y rivalidad.

A mi primo porque sin saber que era mi primo, primero fue mi amigo en la carrera, me enseño a trabajar en mi profesión, y sigue hasta ahora siendo un gran apoyo y guía.

A mis jefes, el Dr. Víctor García Garduño y el Lic. Alejandro Cano porque de ellos aprendí muchísimo desde mediados de mi carrera e inicios de la vida profesional.

A Gennovy por hacerme dar cuenta que cuando uno desea realmente algo nunca es tarde para comenzar a perseguirlo y que siempre existe tiempo para todo, por animarme para cerrar este ciclo que dejé pendiente por un largo tiempo.

Dedico esta tesis a mis papás, hermano y prima, a mi abuela y abuelo, mis tías Patricia, Lilia, Luz e Irma y sus respectivas familias a mis amigos y amigas los cuales no pude mencionar a todos por espacio, a todos mis profesores y a esta institución que me ha cobijado desde los 12 años de edad cuando ingresé a ella.

SISTEMA DE EVALUACIÓN AUTOMATIZADA EN ALMACENAJE POR BASE DE DATOS PARA PERFIL DEL ALUMNO

INDICE DESGLOSADO

CAPÍTULO II
BASE DE DATOS RELACIONAL Y MANEJADOR MYSQL

CAPÍTULO III
IMPLEMENTACIÓN DE PROCEDIMIENTOS DE SEGURIDAD

CAPÍTULO IV
DESARROLLO DEL SISTEMA

INTRODUCCIÓN

Podemos definir la ingeniería como un conjunto de técnicas y conocimientos científicos que aplicándolos se puede crear, perfeccionar e implementar tanto física como teóricamente soluciones, para resolver problemas específicos de una manera optima.

Los sistemas computacionales a lo largo del tiempo han logrado optimizar tareas que en otros tiempos requerían de gran tiempo y esfuerzo humano, hasta hoy en día este tipo de factores propicia la posibilidad de errores en los procesos.

La evolución de estos sistemas también ha sido exponencial, ya que desde el tiempo en que se comenzaron a implementar, con el uso de tarjetas perforadas aproximadamente por los años 60's, pasando por las hojas de respuesta óptica, lectores de código de barras hasta el desarrollo de inteligencia artificial, hemos visto como han sufrido varios cambios y han empleado cada vez nuevas herramientas, modelos y lenguajes de programación para su desarrollo, optimización e implementación.

Los sistemas computacionales mediante la ingeniería, no solo ayudan a resolver problemas específicos de esta área, sino que tienen la posibilidad de ayudar en cualquier campo, como puede ser biología, medicina, psicología, derecho, administración, por mencionar algunos, donde se requiera el procesamiento de información mediante la recolección, desarrollo y presentación de la misma de la manera que el experto en el campo la necesite. Cabe destacar, que el ingeniero a cargo del procesamiento de la información no tiene que ser experto en cada campo para el que este aporte una solución con su trabajo, pero si tiene que familiarizarse con el tema para poder presentar la mejor solución posible.

Como es el caso de esta tesis se conjuntas los dos elementos mencionados en los párrafos anteriores, la ingeniería y los sistemas computacionales, para optimizar una tarea específica de la manera más óptima posible. La aplicación de una prueba para la evaluación de un perfil en el campo de la psicología fue la optimización primordial del sistema, anteriormente se tomaba una evaluación en papel y se aplicaba al sujeto en estudio, el cual iba llenado en su hoja de respuestas según sus decisiones. Una vez que se tenía la hoja de respuestas se comparaba contra una tabla de donde indicaba según la elección, un valor a sumar. Una vez con los valores obtenidos se comparaba contra otra tabla de "percentiles" dependiendo el sexo, y finalmente se ubicaba en una gráfica para obtener un resultado.

El primer punto a optimizar fue la recopilación de los datos que se recababan del sujeto de estudio de una manera electrónica, mediante un portal web, de manera que el sujeto puede realizar su evaluación desde cualquier lugar donde cuente con una conexión a internet, al momento de finalizarlo toda la información estará lista para ser procesada por el sistema. Optimiza tiempo e incluso el gasto innecesario de papel.

El segundo punto fue el procesamiento de los datos para presentar al experto de la materia, el psicólogo que interpreta los mismos, al tener los datos de manera electrónica y vía web, este puede contar con los mismos desde cualquier lugar donde cuente con una conexión a internet. Los datos al estar contenidos en una base de datos pueden ser manipulados de la manera que se desee, al tener sistematizada la correcta parametrización de estos datos, reduce la posibilidad de error en un porcentaje elevado.

Finalmente, los datos son procesados por el sistema de manera que sea lo más fácil para el psicólogo poder interpretar, así que se ahorra también la parte de ubicarlos sobre la tabla de "percentiles" que también está parametrizada por números específicos y únicamente se le presenta el resultado final en una gráfica casi idéntica a la que tiene en sus reactivos en papel para únicamente poder realizar su interpretación, esto optimiza el tiempo, facilitando su trabajo.

OBJETIVOS

Los objetivos principales son los siguientes:

- Crear un sistema de fácil uso para los alumnos que serán diagnosticados mediante la evaluación que hasta el momento se realiza por un cuestionario en papel, mediante una serie de preguntas, contra una hoja de respuestas.

- La plataforma debe tener fácil acceso para los alumnos ya que hasta el momento tiene que ser llevado a cabo en un lugar y horario específico, y atendido mediante el revisor calificado el cual da las instrucciones de los procedimientos.

- El fácil acceso a la información de los datos recopilados para el intérprete o experto, así como asegurarnos que tendrán la disponibilidad de los datos en todo momento, esto se buscará mediante la implementación de manera web, de esta manera también se reducirá el uso de papel pues todo se concentrará electrónicamente en una base de datos.

- Fiabilidad en los datos recopilados y los resultados buscando una reducción de los errores humanos tanto del alumno evaluado al llenar los datos de prueba, como de la lectura de los mismos por parte del experto encargado de la interpretación de los mismos.

- Reducción y optimización de tiempos para que el trabajo cuantitativo pase a ser más cualitativo sobre el análisis de los datos obtenidos.

- Por último el desarrollo del sistema que préndete optimizar todos estos procesos debe tener la capacidad de ser escalable, migrable y poder recibir mantenimiento, independientemente del desarrollador de sistema.

Capítulo I

LENGUAJE DE PROGRAMACIÓN INTERPRETADO PHP (HYPERTEXT PREPROCESSOR)

I.I Definición de Lenguaje de Programación Interpretado

Un lenguaje interpretado es un lenguaje de programación que está diseñado para que el código fuente sea ejecutado por medio de un "sistema traductor", en contraste con los lenguajes compilados que se genera del código fuente un archivo ejecutable para trabajar en un determinado sistema operativo.

I.II Lenguajes interpretados contra lenguajes compilados

Los lenguajes interpretados nacen en respuesta a la dificultad de manejo de los compilados. Un lenguaje compilado es sólo apto para un sistema operativo o formato de ejecutable en especifico, (Linux, Windows o Apple será diferente). Para comprobar **bugs** o errores la computadora debe:

- Compilar cada uno de los ficheros de código.
- Ensamblarlos en ficheros objeto.
- Enlazar los ficheros objeto.
- Volverlos a ensamblar.

Todo eso no es gran derroche de recursos para una computadora. En cambio, con un lenguaje interpretado, el programa intérprete analiza el fichero de código y lo va ejecutando en tiempo real, sin compilarlo ni ensamblarlo. Otra ventaja de los lenguajes interpretados es que son multiplataforma, es decir no debe ser compilado dos veces (para Linux y otra para Windows). Con que haya diferentes versiones del intérprete en cada uno de esos ordenadores, específicamente compilados para ellos basta.

Algunas desventajas son**:**

- Consumo de recursos de memoria, sobre todo RAM.
- Se depende del intérprete: si no tienes instalado el intérprete que corresponda, no podrás ejecutar el programa.

Ejemplos de lenguajes interpretados son PHP, Perl, Python, Tcl/Tk, BASIC, LISP (en algunas de sus versiones).

I.III Antecedentes de Lenguaje de Programación Interpretado

Inicialmente los lenguajes interpretados eran compilados línea por línea, cada línea fue compilada al ser ejecutada y un ciclo o subrutina podría requerir que ciertas líneas fueran ejecutadas varias veces, y recompilados. Esto en la actualidad ya es menos común. Otros lenguajes interpretados usan una representación intermedia, que combina compilación e interpretación, como puede ser compilar código binario en una salida, y el intérprete ejecuta ese código binario.

I.IV Características de Lenguaje de Programación Interpretado

Los lenguajes de programación interpretados tienen cierta flexibilidad sobre los compilados, características que son fáciles de implementar en intérpretes que en compiladores como:

- Independencia de plataforma.
- Uso reflector del evaluador, esto es, su primera orden es evaluar la función.
- Leer a tiempo de corrida.
- Fácil de depurar.
- Tamaño pequeño de los programas.
- Objetos polimórficos.
- Alcance dinámico.

I.V PHP

I.V.I ¿Qué es PHP?

El lenguaje PHP, es un lenguaje de programación interpretado creado originalmente para la construcción de páginas con contenido dinámico. Es usado principalmente en interpretación del lado del servidor, pero actualmente puede utilizarse desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas.

La manera en que funciona normalmente es la siguiente, el código fuente se lee desde un servidor, éste traduce el código fuente al momento de ser leído, sin necesidad de compilar propiamente, a código web que generalmente el resultado que obtenemos es un código que puede ser manipulado por un navegador.

I.V.II Evolución

La primera versión fue originalmente diseñada en Perl, con base en la escritura de un grupo de CGI binarios escritos en el lenguaje C por el programador danés-canadiense Rasmus Lerdorf en 1994, para mostrar la cantidad de tráfico que su página web recibía. El 8 de junio de 1995 fue publicado "Personal Home Page Tools" después de que Lerdorf lo combinara con su propio “Form Interpreter” para crear PHP/FI que daría paso de la versión **PHP 2.0** liberada para junio de 1997.

PHP 3

Dos programadores israelíes del Technion, Zeev Suraski y Andi Gutmans, reescribieron el analizador sintáctico en 1997 y crearon la base del PHP3, cambiando el nombre del lenguaje a la forma actual. Inmediatamente comenzaron experimentaciones públicas de PHP 3 y fue publicado oficialmente en junio del 1998. Las mejoras más notables fueron su gran extensibilidad por la sencillez para ampliar el lenguaje, así los desarrolladores pudieron hacerlo modular.

PHP 4

En mayo de 2000 PHP 4 fue lanzado bajo el poder del motor Zend Engine 1.0. El día 13 de julio de 2007 se anunció la suspensión del soporte y desarrollo de la versión 4 de PHP, a pesar de lo anunciado se ha liberado una nueva versión con mejoras de seguridad, la 4.4.8 publicada el 13 de enero del 2008 y posteriormente la versión 4.4.9 publicada el 7 de agosto de 2008.

PHP 5

El 13 de julio de 2004, fue lanzado PHP 5, utilizando el motor Zend Engine 2. La versión más reciente de PHP es la 5.5.0 (20 de junio de 2013), que incluye todas las ventajas que provee el nuevo Zend Engine 2 como:

- Mejor soporte para la Programación Orientada a Objetos, que en versiones anteriores era extremadamente rudimentario, con PHP Data Objects.
- Mejoras de rendimiento.
- Mejor soporte para MySQL con extensión completamente reescrita.
- Mejor soporte a XML (XPath, DOM, etc.).
- Soporte nativo para SQLite.
- Soporte integrado para SOAP.
- Iteradores de datos.
- Manejo de excepciones.
- Aceleradores de PHP que optimizan mediante almacenamiento de cache.

PHP 6

Está previsto el lanzamiento en breve de la rama 6 de PHP. Cuando se lance esta nueva versión quedarán solo dos ramas activas en desarrollo (PHP 5 y 6). Las diferencias que encontraremos frente a PHP 5 son:

- Soportará Unicode.
- Limpieza de funcionalidades obsoletas como register_globals, safe_mode, etc.
- PECL.
- Mejoras en orientación a objetos.

I.V.III Características

Ventajas

- Es un lenguaje multiplataforma.
- Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL.
- Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones).

- Posee una amplia documentación en su página oficial, 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 las técnicas de Programación Orientada a Objetos.
- Biblioteca nativa de funciones sumamente amplia e incluida.
- No requiere definición de tipos de variables.
- Tiene manejo de excepciones (desde PHP5).

Desventajas

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), aún 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.

I.V.IV Uso del lenguaje PHP

El código **PHP** puede incluirse, aunque no es necesario, dentro del código HTML de una página web. Para delimitar la sección de código PHP podemos hacerlo de las siguientes formas:

-Usando las etiquetas: **<?php** *Código PHP* **?>**

-Usando las etiquetas: **<?** *Código PHP* **?>**

-Mediante: **<script languaje="php">** *Código 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.

I.V.V Manejo de variables y tipos de dato

Una **variable** puede definirse como una posición de memoria creada para introducir o asignar cualquier valor o dato. Durante la ejecución del script el valor de la variable tiene la capacidad de cambiar tanto de tipo, como de valor. En PHP no hace falta declarar la variable ya que simplemente anteponiendo el caracter $ al nombre de la variable estamos indicando que es una variable.

Otro factor a destacar a la hora de programar en PHP y la declaración de variables es que PHP es un lenguaje "CASE SENSITIVE" es decir que diferencia entre mayúsculas y minúsculas y debido a esta razón no sería lo mismo:

```
$Variable=valor;
```

que

```
$VaRiABle=valor;
```

ya que PHP lo interpretaría como dos variables completamente diferentes.

Los tipos de datos posibles que puede almacenar una variable son los siguientes:

Integer Números enteros positivos y negativos
Double Números decimales o de coma flotante
String Cadenas de texto
Boolean Valores True o False
Array Tipo especial de colección de valores
Object Tipo especial de dato complejo

I.V.VI Uso de operadores

Los operadores son símbolos especiales que se utilizan para realizar tanto operaciones matemáticas, como de comparación.

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 negación. Si se tiene 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 igual

Operador ternario

El operador ternario evalúa una condición y retorna un valor dependiendo si la condición es verdadera (true) o falsa (false).

```
if (condición) {
variable = valor-cuando-es-verdadera;
} else {
variable = valor-cuando-es-falsa;
}
```

Utilizando el operador ternario se simplifica de esta manera:

```
variable = (condición) ? valor-cuando-es-verdadera : valor-cuando-es-
falsa;
```

I.V.VII Uso de funciones

Una de las herramientas más importantes en cualquier lenguaje de programación son las funciones.

Una función consiste en un conjunto de rutinas y acciones que a lo largo de un programa van a ser ejecutadas multitud de veces agrupados. La FUNCIÓN es usada desde cualquier punto del programa siendo llamada y ejecutada. A su vez, puede recibir parámetros externos de los cuales dependa el resultado de la misma.

Las funciones deben ser colocadas siempre antes de realizar las llamadas a estas. La sintaxis de una función es la siguiente:

```
function nombre(parámetros){
  instrucciones de la función
}
```

para llamar a la función sería de la siguiente forma:

```
nombre(parámetros)
```

Un hecho relevante que cabe destacar es que las variables que declaremos dentro de la función solo existirán o tendrán dicho valor dentro de esta.

Algunas funciones que se utilizaran son:

AddSlashes.- Escapa una cadena insertando barras "\". Devuelve una cadena con barras invertidas delante de los caracteres que necesitan escaparse en situaciones como consultas de bases de datos, etc. Los caracteres que se escapan son la comilla simple (’), comilla doble (”), barra invertida (\) y NULL (el byte **NULL**).

Trim.- Elimina espacios en blanco (u otros caracteres) del principio y final de una cadena:

- " " (ASCII 32 (0x20)), un espacio en blanco.
- "\t" (ASCII 9 (0x09)), un tabulador.
- "\n" (ASCII 10 (0x0A)), una nueva linea.
- "\r" (ASCII 13 (0x0D)), un retorno de carro.
- "\0" (ASCII 0 (0x00)), el byte NUL.
- "\x0B" (ASCII 11 (0x0B)), un tabulador vertical.

get_magic_quotes_gpc.- Devuelve el valor actual de configuración de magic_quotes_gpc (0 si está deshabilitado, 1 de lo contrario).

Isset.- Esta función nos permite comprobar si una variable se ha definido y en ese caso devuelve un True.

htmlentities.- En esta función todos los caracteres que tengan una entidad equivalente en HTML serán cambiados a esas entidades.

intval.- Devuelve el valor entero de una variable, usando la base de conversión especificada (por defecto es base 10).

doubleval .- Alias de **floatval().**

floatval .- Obtiene el valor real de una variable.

doubleval() se convirtió en un alias de **floatval()**. Anteriormente, sólo existía **doubleval()**.

$_SERVER.- Es una matriz que contiene información tal como cabeceras, rutas y ubicaciones de scripts.

'PHP_SELF'.- El nombre de archivo del script ejecutándose actualmente, relativo a la raíz de documentos.

'QUERY_STRING'.- Da la cadena de consulta, si existe, mediante la cual se accedió a la página.

date.- Devuelve una cadena de acuerdo al formato dado usando un entero o la hora local si no se da una marca de tiempo.

string date (string formato [, int marca_de_tiempo])

donde:

marca_de_tiempo es opcional y su valor predeterminado es el valor de **time()**.

Las opciones para dar el formato son las de la tabla 1.1.

Caracter de *formato*	**Descripción**	**Valores de ejemplo devueltos**
Día	---	---
D	Día del mes, 2 dígitos con ceros iniciales	*01* a *31*
D	Una representación textual de un día, tres letras	*Mon* a *Sun*
J	Día del mes sin ceros iniciales	*1* a *31*
l ('L' minúscula)	Una representación textual completa del día de la semana	*Sunday* a *Saturday*
N	Representación numérica ISO-8601 del día de la semana (agregado in PHP 5.1.0)	*1* (para Lunes) a *7* (para Domingo)
S	Sufijo ordinal en inglés del dia del mes, 2 caracteres	*st*, *nd*, *rd* o *th*. Funciona bien

Caracter de *formato*	Descripción	Valores de ejemplo devueltos
		con *j*
W	Representación numérica del día de la semana	*0* (para el Domingo) a *6* (para el Sábado)
Z	El día del año (comenzando en 0)	*0* a *365*
Semana	---	---
W	Número de la semana del año ISO-8601, las semanas comienzan en Lunes (agregado en PHP 4.1.0)	Ejemplo: *42* (la 42va semana del año)
Mes	---	---
F	Una representación textual completa de un mes, como January o March	*January* a *December*
M	Representación numérica de un mes, con ceros iniciales	*01* a *12*
M	Una representación textual corta de un mes, tres letras	*Jan* a *Dec*
N	Representación numérica de un mes, sin ceros iniciales	*1* a *12*
T	Número de días en el mes dado	*28* a *31*
Año	---	---
L	Indica si es un año bisiesto	*1* si es un año bisiesto, *0* de lo contrario.
o	Número de año ISO-8601. Este es el mismo valor que *Y*, excepto que si el número de semana ISO (*W*) pertenece al año previo o siguiente, ese año será usado en su lugar. (agregado en PHP 5.1.0)	Ejemplos: *1999* o *2003*
Y	Una representación numérica completa de un año, 4 dígitos	Ejemplos: *1999* o *2003*
y	Una representación de dos dígitos de un año	Ejemplos: *99* o *03*
Hora	---	---
a	Ante meridiano y Post meridiano en minúsculas	*am* o *pm*
A	Ante meridiano y Post meridiano en mayúsculas	*AM* o *PM*
B	Hora Swatch Internet	*000* a *999*
g	formato de 12-horas de una hora sin ceros iniciales	*1* a *12*
G	formato de 24-horas de una hora sin ceros iniciales	*0* a *23*
h	formato de 12-horas de una hora con ceros iniciales	*01* a *12*
H	formato de 24-horas de una hora con ceros iniciales	*00* a *23*

Caracter de *formato*	**Descripción**	**Valores de ejemplo devueltos**
i	Minutos con ceros iniciales	*00* a *59*
s	Segundos, con ceros iniciales	*00* a *59*
Zona horaria	---	---
e	Identificador de zona horaria (agregado en PHP 5.1.0)	Ejemplos: *UTC*, *GMT*, *Atlantic/Azores*
I (i mayúscula)	Indica si la fecha están en hora de ahorro de luz diurna	*1* si es Hora de Ahorro de Luz Diurna, *0* de lo contrario.
O	Diferencia con la hora Greenwich (GMT) en horas	Ejemplo: *+0200*
P	Diferencia con la hora Greenwich (GMT) con dos-puntos entre las horas y los minutos (agregada en PHP 5.1.3)	Ejemplo: *+02:00*
T	Configuración de zona horaria de esta máquina	Ejemplos: *EST*, *MDT* ...
Z	Desplazamiento de la zona horaria en segundos. El desplazamiento para zonas horarias al oeste de UTC es siempre negativo, y el de aquellas al este de UTC es siempre positivo.	*-43200* a *43200*
Fecha/Hora Completa	---	---
c	Fecha ISO 8601 (agregada en PHP 5)	2004-02-12T15:19:21+00:00
r	Fecha en formato RFC 2822	Ejemplo: *Thu, 21 Dec 2000 16:01:07 +0200*
U	Segundos desde el Epoch Unix (January 1 1970 00:00:00 GMT)	De función **time()**

Tabla 1.1 Parámetros para la obtención de fechas y tiempo.

Capítulo II

BASE DE DATOS RELACIONAL EN MANEJADOR MYSQL

II.I Base de Datos

II.I.I Definición

Es una colección integrada por datos persistentes, que contiene información relevante para una empresa, organización o proyecto.

II.I.II Tipos de Bases de Datos

Las bases de datos pueden clasificarse de varias formas, de acuerdo al contexto, la utilidad de las mismas o las necesidades que satisfagan.

II.I.II.I Por variabilidad de los datos almacenados

Bases de datos estáticas

Bases de datos de sólo lectura, utilizadas generalmente para almacenaje datos históricos que posteriormente se usan para estudiar el comportamiento de un conjunto de datos a través del tiempo, realizar proyecciones, tomar decisiones y realizar análisis de datos para toma de decisiones empresariales.

Bases de datos dinámicas

Bases de datos donde la información almacenada se modifica con el tiempo, permitiendo operaciones como actualización, borrado e inserción de datos, además de operaciones de consulta.

II.I.II.II De acuerdo a su modelo de administración

Un modelo de datos es básicamente una "descripción" de algo conocido como *contenedor de datos*, así como de los métodos para almacenar y recuperar información de esos contenedores. Los modelos de datos no son entidades físicas, son abstracciones que permiten la implementación de un sistema eficiente de *base de datos*; es decir algoritmos, y conceptos matemáticos.

Los modelos los podemos clasificar por:

Bases de datos jerárquicas

En este los datos se organizan en una forma similar a un árbol invertido, en donde un **nodo padre** de información puede tener varios **hijos**. El nodo que no tiene padres es llamado **raíz**, y a los nodos que no tienen hijos se los conoce como **hojas**. Son útiles en el caso de aplicaciones que manejan un gran volumen de información y datos compartidos. Su desventaja es la redundancia de información. No admite relaciones muchos a muchos **N:M**.

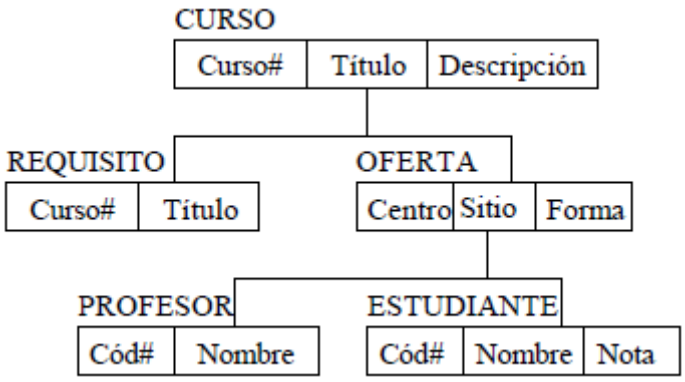


Figura 1. Ejemplo de diagrama de jerárquico.

Base de datos de red

Este se distingue del jerárquico con la diferencia de la modificación del concepto de **nodo**: se permite que un mismo nodo tenga **varios padres** (posibilidad no permitida en el modelo jerárquico), que es una solución eficiente al problema de redundancia de datos, es decir admite relaciones **N:M**.

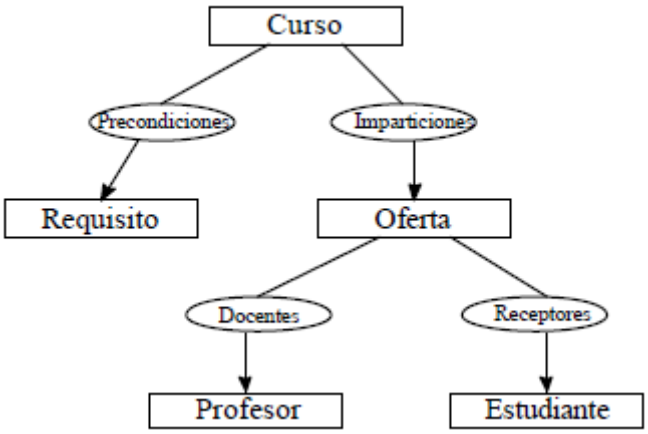


Figura 2. Modelo de Jerárquico mediante modelo de red.

Bases de datos transaccionales

Son bases de datos cuyo único fin es el envío y recepción de datos a grandes velocidades, no son tan comunes, dirigidas por lo general al entorno de análisis de calidad, datos de producción e industrial. Su fin único es recolectar y recuperar los datos a la mayor velocidad posible, la redundancia y duplicación de información no es problema como con las otras. Por lo general para poderlas aprovechar al máximo permiten algún tipo de conectividad a bases de datos relacionales.

Bases de datos relacionales

Éste es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. Su idea fundamental es el uso de "relaciones". Estas relaciones podrían considerarse en forma lógica como conjuntos de datos llamados "tuplas". Esto es pensada en cada relación como si fuese una tabla que está compuesta por registros (las filas de una tabla), que representarían las tuplas, y campos (las columnas de una tabla). En el punto **II.II** se explicará más a detalle este tipo de base.

Marca	Modelo	Color	Matrícula	Situación
Lamborghi.	Diablo 630	Amarillo	MA-2663-BC	En renta
Ferrari	F-40	Rojo	MA-8870-BC	Disponible
Sbärro R.	Decade	Blanco	VD-870-GTH	Disponible
De Tomaso	Pantera	Blanco	ML-7890-B	En renta
Pontiac	Trans-Am	Negro	KNIGHT	En taller
Austin M.	S3'40	Marrón	CA-5647-AB	Disponible
Jaguar	Destructor	Verde	AD-768-TTY	En renta

Apellidos	Nombre	D.N.I.	Edad
González Aranda	Javier	75836934	27
Beato Apóstol	Antonio	28836746	43
Campos Ortega	Adriano	82665358	36
Ruíz Rojo	Juan	83667228	35

Matrícula	D.N.I.
MA-2663-BC	75836934
ML-7890-B	83667228
AD-768-TTY	75836934
AD-768-TTY	82665358

Figura 3.Ej. de tablas de modelo de relacional.

Bases de datos multidimensionales

Bases de datos ideadas para desarrollar aplicaciones muy concretas, como creación de **Cubos OLAP**. Básicamente no se diferencian demasiado de las bases de datos relacionales (una tabla en una base de datos relacional podría serlo también en una base de datos multidimensional), la diferencia está más bien a nivel conceptual; en las bases de datos multidimensionales los campos o atributos de una tabla pueden ser de dos tipos, o bien representan dimensiones de la tabla, o bien representan métricas que se desean estudiar.

Bases de datos orientadas a objetos

Este modelo, bastante reciente, y propio de los modelos informáticos orientados a objetos, trata de almacenar en la base de datos los objetos completos es decir por estado y comportamiento. Incorpora los conceptos del paradigma de objetos como: encapsulación, herencia y polimorfismo.

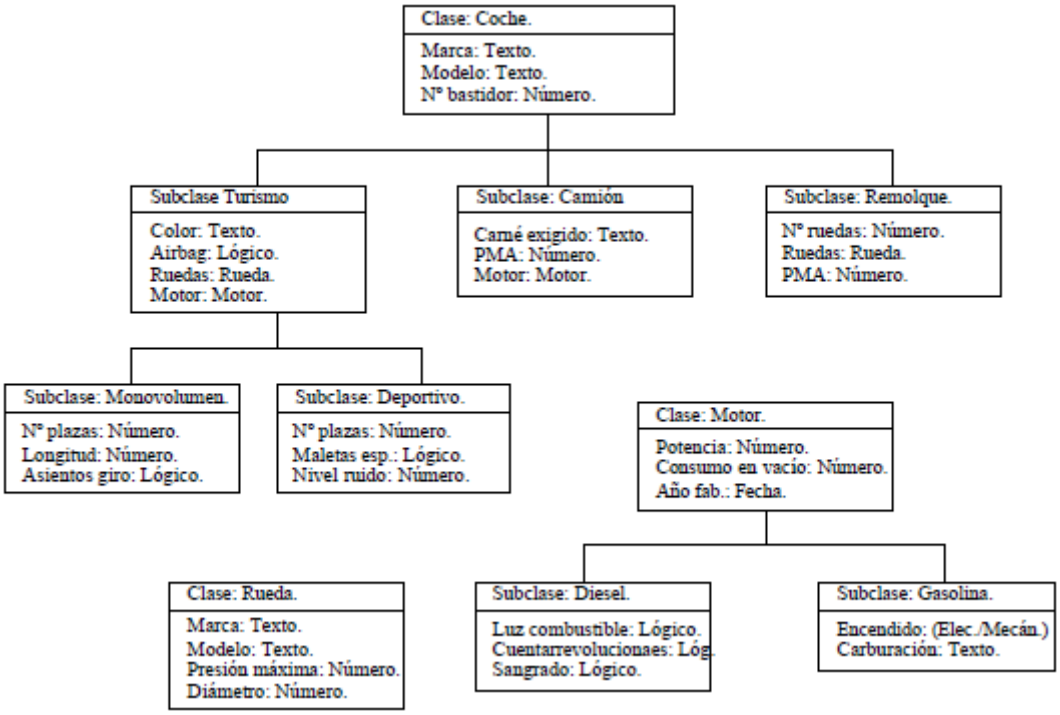


Figura 4. Diagrama de modelo orientado a objetos.

II.I.III Sistema Manejador de Base de Datos (DBMS)

Un Sistema Manejador de Base de Datos DBMS (por sus siglas en inglés) consiste en una colección de datos interrelacionados (base de datos) y un conjunto de programas para manipular administrar y acceder a dichos datos.

Su principal objetivo es proporcionar una forma de almacenar la información, así como la provisión de mecanismos para la manipulación de la información. Además el sistema debe proporcionar un alto porcentaje de confianza para información almacenada, a pesar de la caída del sistema o intentos de acceso no autorizador.

Generalmente los datos son compartidos por diversos usuarios, y el sistema debe evitar posibles resultados anómalos de la manipulación que le da cada uno.

II.I.IV Sistemas de Archivos

Un sistema de procesamiento de archivos, se mantiene mediante un sistema operativo convencional, los registros permanentes son almacenados en varios archivos y se escriben diferentes programas de aplicación para extraer registros y para añadir registros a los archivos adecuados.

Algunos de sus inconvenientes son:

- Redundancia e inconsistencia de datos.
- Dificultad de acceso a los datos.
- Aislamientos de datos.

- Problemas de integridad.
- Problemas de atomicidad.
- Anomalías en acceso concurrente.
- Problemas de seguridad.

II.I.V Visión de los datos

Uno de los propósitos de un SGBD es proporcionar a los usuarios una visión abstracta de los datos, es decir, el sistema esconde ciertos detalles de cómo se almacenan y mantiene los datos.

Los principales niveles de abstracción son los siguientes:

- Nivel físico. Es el nivel más bajo de abstracción, describe como se almacenan realmente los datos, el detalle de las estructuras de datos complejas de bajo nivel.
- Nivel lógico. Describe que datos se almacenan en la base de datos, y que relación existe entre ellos. Las estructuras a nivel lógico pueden ser muy simples aunque a nivel físico sean muy complejas.
- Nivel de vistas. El nivel más alto de abstracción describe solo parte de la base de datos completa. Muchas veces los usuarios no necesitan acceso completo a toda la base de datos, con base en esto, el sistema crea vistas para los usuarios dependiendo que necesidad tengan de acceso a los datos.

II.I.VI Modelos de los datos

Son una colección de herramientas conceptuales para describir los datos, las relaciones, la semántica y las restricciones de consistencia.

II.I.VI.I Modelo entidad-relación (E-R).

Esta basado en una percepción del mundo real que consta en una colección de objetos básicos, llamados *entidades,* y de *relaciones* entre estos objetos.

Una **entidad** es una “cosa” u “objeto” en el mundo real que es distinguible de otros objetos. Las entidades se describen en una base de datos mediante un conjunto de **atributos** que describen la entidad. Por lo general se utiliza atributos extra, un **identificador** que se usarán de manera única para discriminar un elemento de otro.

Una **relación** es una asociación entre varias entidades.

La forma de representación de una base de datos mediante un diagrama E-R consta de los siguientes componentes:

- Rectángulos. Representan conjuntos de **entidades**.
- Elipses. Que representan **atributos**.
- Rombos. Que representan **relaciones** entre conjuntos de entidades.
- Líneas. Que unen los atributos con los conjuntos de entidades y los conjuntos de entidades con las relaciones.

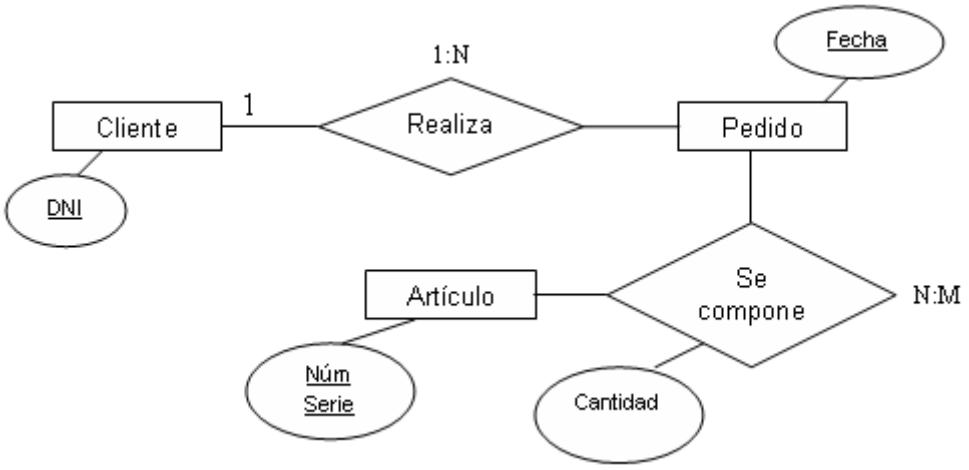


Figura 5. Ejemplo de diagrama Entidad - Relación

La forma de representación de una base de datos mediante un diagrama E-R.

II.II Bases de Datos Relacionales

II.II.I Definición

La forma de representación de una base de datos mediante un diagrama E-R consta de los componentes ya mencionados que son entidades, atributos y relaciones.

Una base de datos relacional es una base de datos que agrupa los datos que utiliza atributos comunes encontrados en un conjunto de datos. Los "grupos" resultantes de datos organizados son mucho más fáciles para comprender por las personas.

Tal colocación utiliza el modelo relacional (un término técnico para este esquema). De ahí tal base de datos es llamada una base de datos relacional.

El software usado para este tipo de bases de datos es un sistema administrador de base de datos relacionales. El termino base de datos relacionales es usualmente referido a este software.

Estrictamente, una base de datos relacional es una colección de relaciones (llamadas frecuentemente tablas). Otros elementos son considerados con frecuencia parte de la base de datos, como ellos ayudan a organizar y estructurar los datos, además de forzar la base de datos para conformarse a un conjunto de requerimientos.

II.II.II Relación Base y Derivadas

En una base de datos relacional, todos los datos se almacenan y se acceden a ellos por medio de relaciones. Las relaciones que almacenan datos son llamados "relaciones base" y su implementación es llamada "tabla". Otras relaciones no almacenan datos, pero que son calculadas al aplicar operaciones relacionales. Estas relaciones son llamadas "relaciones derivadas" y su implementación es llamada "vista" o "consulta". Las relaciones derivadas son convenientes ya que expresan información de varias relaciones actuando como si fuera una sola.

II.II.III Restricciones o Constraints

Una restricción es una condición que obliga el cumplimiento de ciertas condiciones en la base de datos. Algunas no son determinadas por los usuarios, sino que son inherentemente definidas por el simple hecho de que la base de datos sea relacional.

Las restricciones proveen un método de implementar reglas en la base de datos. Las restricciones restringen los datos que pueden ser almacenados en las tablas. Usualmente se definen usando expresiones que dan como resultado un valor booleano, indicando si los datos satisfacen la restricción o no.

Las restricciones no son parte formal del modelo relacional, pero son incluidas porque juegan el rol de organizar mejor los datos. Las restricciones son muy discutidas junto con los conceptos relacionales.

II.II.IV Dominios

Un dominio describe un conjunto de posibles valores para cierto atributo. Como un dominio restringe los valores del atributo, puede ser considerado como una restricción. Matemáticamente, atribuir un dominio a un atributo significa "todos los valores de este atributo deben de ser elementos del conjunto especificado".

II.II.V Llave Única

Cada tabla puede tener uno o más campos cuyos valores identifican de forma única cada registro de dicha tabla, es decir, no pueden existir dos o más registros diferentes cuyos valores en dichos campos sean idénticos. Este conjunto de campos se llama llave única.

Pueden existir varias llaves únicas en una determinada tabla, y a cada una de éstas suele llamársele candidata a llave primaria.

II.II.VI Llave Primaria

Una llave primaria puede ser una llave única elegida entre todas las candidatas, para especificar los datos que serán relacionados con las demás tablas. La forma de hacer esto es por medio de llaves foráneas. Otro tipo de llave primaria puede ser la combinación de 2 atributos que se vuelvan únicos e irrepetibles.

Sólo puede existir una llave primaria por tabla y ningún campo de dicha llave puede contener valores NULL.

II.II.VII Llave Foránea

Una llave foránea es una referencia a una llave en otra tabla. Las llaves foráneas no necesitan ser llaves únicas en la tabla donde están y si a donde están referenciadas.

Por ejemplo, el código de departamento puede ser una llave foránea en la tabla de empleados, pero obviamente se permite que haya varios empleados en un mismo departamento, pero existirá sólo un departamento.

II.II.VIII Estructura

La base de datos se organiza en dos marcadas secciones; el esquema y los datos (o instancia).

El esquema es la definición de la estructura de la base de datos y principalmente almacena los siguientes datos:

- El nombre de cada tabla
- El nombre de cada campo
- El tipo de dato de cada campo
- La tabla a la que pertenece cada campo

Las bases de datos relacionales pasan por un proceso al que se le conoce como normalización, el resultado de dicho proceso es un esquema que permite que la base de datos sea usada de manera óptima.

Los datos o instancia es el contenido de la base de datos en un momento dado. Es en si, el contenido de todos los registros.

II.II.IX Ventajas y Desventajas

Ventajas

- Provee herramientas que garantizan evitar la duplicidad de registros.
- Garantiza la integridad referencial, así, al eliminar un registro elimina todos los registros relacionados dependientes.
- Favorece la normalización por ser más comprensible y aplicable.

Desventajas

- Requieren un especialista que pueda manipular, administrar y dar, mantenimiento al DBMS que aloja la base de datos relacional.
- El lenguaje de manipulación SQL requiere un grado de preparación para realizar consultas de información a diferencia de otras bases de datos **no** relacionales como por ejemplo un archivo de Excel donde la manipulación es gráfica de origen.

II.II.X Modelo Relacional

El modelo relacional para la gestión de base de datos es un modelo de base de datos basado en la lógica de predicado de primero-orden, primero formulado y propuesto en 1969.

II.II.XI Lógica de Primer Orden

La lógica de primero-ordena es un sistema deductivo formal utilizado en matemáticas, en la filosofía, en la lingüística, e informática. Es un idioma formal enteramente sin ambigüedades, interpretado por estructuras matemáticas.

II.III SQL

El **lenguaje de consulta estructurado** o **SQL** (en inglés *structured query language*) es un lenguaje declarativo de "alto nivel" o "de no procedimiento", que gracias a su fuerte base teórica y su orientación al manejo de conjuntos de registros, y no a registros individuales, permite una alta productividad en codificación y la orientación a objetos. Esto permite acceso a bases de datos relacionales con diversos tipos de operaciones. Entre sus características es el manejo del álgebra y el cálculo relacional permitiendo efectuar consultas con el fin de obtener una manera sencilla información de interés en una base de datos, así como también hacer cambios sobre la misma.

II.III.I Lenguaje de definición de datos (DDL)

El lenguaje de definición de datos (en inglés *Data Definition Language*, o *DDL*), se encarga de la modificación de la estructura de los objetos de la base de datos. Contiene instrucciones para modificar, borrar o definir las tablas en las que se almacenan los datos de la base de datos. Ejemplos de operaciones básicas son: CREATE, ALTER, DROP y TRUNCATE.

II.III.II Lenguaje de manipulación de datos (DML)

Un lenguaje de manipulación de datos (*Data Manipulation Language*, *DML* en inglés) proporciona al sistema de gestión de base de datos que permite a los usuarios llevar a cabo tareas de consulta o manipulación de los datos, organizados por el modelo de datos adecuado. Ejemplo de instrucciones básicas son: INSERT INTO, UPDATE, DELETE.

II.IV Manejador de Base de Datos MySQL

Es un sistema de administración de bases de datos relacional, multihilo y multiusuario. El proyecto MySQL es patrocinado por una empresa privada, que posee el copyright de la mayor parte del código.

II.IV.I Características

Inicialmente, MySQL carecía de elementos esenciales en las bases de datos relacionales, como integridad referencial y transacciones. Aún así, para los desarrolladores de páginas web fue atractivo con contenido dinámico y su simplicidad.

En la actualidad los elementos de los que carecía MySQL están siendo incluidos tanto por internos, como por desarrolladores de software libre. Entre las características disponibles se puede destacar:

- Amplio subconjunto del lenguaje SQL. Algunas extensiones son incluidas igualmente.
- Disponibilidad en gran cantidad de plataformas y sistemas.
- Posibilidad de selección de mecanismos de almacenamiento que ofrecen diferente velocidad de operación, soporte físico, capacidad, distribución geográfica, transacciones...
- Transacciones de integridad y llaves foráneas.
- Conectividad segura.
- Replicación.
- Búsqueda e indexación de campos de texto.

Las siguientes características son implementadas únicamente por MySQL:

- Permite escoger entre múltiples motores de almacenamiento para cada tabla. Ejemplos de estos son:

    - Los hay nativos como MyISAM, Falcon, Merge, InnoDB, BDB, Memory/heap, MySQL Cluster, Federated, Archive, CSV, Blackhole y Example.
    - Desarrollados por *partners* como solidDB, NitroEDB, ScaleDB, TokuDB, Infobright (antes Brighthouse), Kickfire, XtraDB, IBM DB2). InnoDB Estuvo desarrollado así pero ahora pertenece también a Oracle.
    - Desarrollados por la comunidad como memcache, httpd, PBXT y Revision.

- Agrupación de transacciones, reuniendo múltiples transacciones de varias conexiones para incrementar el número de transacciones por segundo.

Capítulo III

IMPLEMENTACIÓN DE PROCEDIMIENTOS DE SEGURIDAD

Se puede definir como seguridad informática al seguimiento de las mejores prácticas, normas, metodologías y técnicas que permiten salvaguardar lo mejor posible un sistema tecnológico que administra información, tanto en software como hardware.

La seguridad que podemos implementar en un sistema de este tipo es diversa. Puede implementarse en la programación del sistema, o desde la configuración del servidor, la red dentro de la que se encuentra, y el control de acceso de usuarios en esta.

Para este desarrollo se implementará tanto, seguridad en el servidor, así como seguridad mediante programación, de esta manera sea cual sea el servidor, o la red, la seguridad seguirá operando aunque se cambie de infraestructura.

III.I Variables Globales

En muchos lenguajes deben crearse explícitamente variables para ser utilizadas. En PHP específicamente, la opción de “register_globals”, permite emplear las variables globales.

Consideremos el siguiente ejemplo que hace uso de las variables:

```
if ($password == "mi_password") {
   $autoriz = 1;
}
if ($autoriz == 1) {
   echo "Mis cosas importantes ";
}
```

En un servidor que tiene “register_globals” encendidos, simplemente agregando: **?autoriz=1** al URL dará acceso a cualquier persona a exactamente lo que no quisieras que todo el mundo viera. Éste es uno de los problemas más comunes de la seguridad de PHP.

Una de las soluciones es desactivar los “register_globals”. En ocasiones esto no es posible puesto que este tipo de variables son necesarias para trabajar, en este caso lo que se puede hacer es agregar líneas al código para asegurarse que aún, agregando líneas al URL, no dará acceso a menos que en verdad cumpla con las condiciones establecidas:

```
$autoriz = 0; //al principio del código

if ($password == "my_password") {

$autoriz = 1;

...
```

III.II Mensajes de Error

Los mensajes de error son una herramienta empleada tanto por los programadores como por hackers. Un desarrollador los necesita para detectar “bugs”, pero un hacker puede utilizarlos para descubrir todas las clases de información sobre un sitio, desde la estructura del directorio del servidor hasta la información de la conexión de la base de datos. En PHP se evita esto configurando los archivos de .htaccess o php.ini, fijando el “error_reporting” de manera que no aparezcan dichos mensajes.

III.III Cuidado de SQL

Una de las ventajas más grandes de PHP es la facilidad con la cual puede comunicarse con las bases de datos, lo más normal con MySQL. Muchas veces se hace un uso excesivo de éste acceso, y muchos sitios, confían en las bases de datos para funcionar. Sin embargo, con esa ventaja hay problemas suficientemente grandes en la seguridad que hay que solucionar.

III.III.I Inyección en SQL

Podemos definir **inyección SQL** como la vulnerabilidad en la validación de entradas a la base de datos en una aplicación. La aplicación puede ser cualquier tipo de software que precise del uso de una base de datos, o una aplicación web que interactúa con la base de datos para funcionar, ya bien sea en un sistema de búsqueda, extracción, inserción de información o creación de lugares restringidos dando acceso mediante manejo de usuarios con contraseñas estableciendo diferentes niveles de permisos. La **inyección SQL** es un fallo que puede resultar terrible, ya que **permite la ejecución de sentencias SQL no deseadas,** desde el borrado de las tablas hasta el cambio de permisos pasando por obtención de contraseñas y otros datos de tablas.

El peligro más común de seguridad con el que tenemos que tener cuidado, es cuando un usuario utiliza un fallo para poder atacar directamente al servidor de bases de datos con sentencias SQL.

Un ejemplo común es cuando se busca mediante el usuario y la contraseña, un fallo al hacer todas las combinaciones válidas del usuario y de su contraseña, por ejemplo para controlar el acceso a un área de administración:

```
$check = mysql_query("SELECT Username, Password, UserLevel FROM Users
WHERE Username = '".$_POST['username']."' and Password =
'".$_POST['password']."'");
```

Aquí introduciendo el siguiente “usuario” en el formulario:

```
'or '1'='1#
```

La sentencia que va a ser ejecutada sería esta:

```
SELECT Username, Password FROM Users WHERE Username = '' OR '1'='1'#
and Password = ''
```

La almohadilla (#) le dice a MySQL que todo que le sigue es un comentario y por lo tanto, que no procesará. Ejecutará SQL hasta ese punto. Después 1 es igual a 1, SQL devolverá todos los usuarios y contraseñas de la base de datos. Y como la primera combinación del usuario y de contraseña en la mayoría de las bases de datos es la del administrador, la persona que incorporó simplemente algunos símbolos en un formulario ahora entra como administrador, con los mismos privilegios que tendría si supiera realmente el usuario y la contraseña.

Con una poca de creatividad, este agujero de seguridad se puede explotar aún más lejos, permitiendo a un usuario crear su propia cuenta, leer y modificar datos importantes o simplemente borre la base de datos.

Afortunadamente, este tipo de vulnerabilidad es bastante fácil de solucionar. Comprobando si hay algún caracter raro cuando el usuario introduce los datos, y quitándolos o neutralizándolos, podemos evitar que cualquier persona utilice su propio código del SQL en nuestra base de datos. La función que sigue sería la adecuada:

```
function make_safe($variable) {
   $variable = addslashes(trim($variable));
   return $variable;
}
```

Ahora debemos modificar nuestra consulta. En vez de usar variables _POST como en la consulta de arriba, ahora utilizamos todos los datos del usuario con la función make_safe, dando por resultado el código siguiente:

```
$username = make_safe($_POST['username']);
$password = make_safe($_POST['password']);
$check = mysql_query("SELECT Username, Password, UserLevel FROM Users
WHERE Username = '".$username."' and Password = '".$password."'");
```

Si un usuario incorporó los datos anteriormente citados, la consulta será totalmente inofensiva. La función usada es la siguiente:

```
function make_safe($variable) {
$variable = addslashes(trim($variable));
return $variable;
}
```

Recibe una variable y la "limpia" de cualquier caracter no deseado mediante **addslashes y trim,** desechando cualquier posibilidad de inyección SQL. Su uso es recomendado cuando se quiere utilizar en una consulta contra la base de datos.

III.IV Otras fallas de seguridad

Algunos sitios tienen URL's parecidas a esto:

index.php?page=contacto.html

El archivo "index.php" incluye simplemente el archivo de "contacto.html", y el sitio parece funcionar. Sin embargo, el usuario puede cambiar muy fácilmente el archivo de "contacto.html" por cualquier otro que le apetezca. Por ejemplo, si estamos utilizando el mod_auth de Apache para proteger archivos y para guardar tu contraseña en el archivo llamado ".htpasswd" (el nombre convencional), si un usuario quisiera visitar esa URL, la respuesta haría salir el nombre de usuario y contraseña con:

index.php?page=.htpasswd

Este hueco de seguridad es razonablemente fácil de solucionar. Primero hay que cerciorase de haber fijado correctamente el "open_basedir" en el archivo de php.ini, y fijar "allow_url_fopen" a "OFF". Eso prevendrá la mayor parte de estas clases de ataques evitando la inclusión de archivos alejados y de ficheros del sistema. Después se debe comprobar el archivo solicitado contra una lista de archivos válidos, limitando así los archivos que se pueden alcanzar usando esta salida.

III.IV Manejo de Cookies

PHP permite manejar cookies con gran sencillez. Las cookies son pequeños archivos de texto que nuestra página puede almacenar en el disco duro de los visitantes, y recuperar cuando vuelvan a visitarla.

La función para colocar una cookie es setcookie(), y su sintaxis:

```
setcookie(Nombre, Valor, Tiempo_Vida, Path, Dominio, Seguro);
```

El primer parámetro es, pues, el **nombre** de la cookie. Es el único valor estrictamente necesario, los demás son opcionales.

Automáticamente se establecen los parámetros por defecto:

tiempo de vida = lo que dure la sesión;

path = el directorio actual (dependiendo del navegador);

dominio = el dominio de la página;

seguro = indica si la cookie está obligada a usar conexiones seguras.

Debemos tener en cuenta que la instrucción para colocar la cookie ha de ir junto a las cabeceras http de la página, por tanto hay que incluirla al comienzo del script, antes de que comience el volcado de html al usuario; de lo contrario dará error.

El tercer parámetro de la cookie es su tiempo de vida. Se especifica en segundos, si no especificamos nada solo dura la sesión, es decir, hasta que cerremos el navegador.

La función setcookie devuelve un 1 o un 0 según se haya ejecutado con éxito, aunque que el hecho de que la función se haya ejecutado correctamente, no implica que la cookie haya sido aceptada por el navegador del usuario, esto solo puede comprobarse leyendo la cookie.

Capítulo IV

DESARROLLO DEL SISTEMA

IV.I Diseño del Sistema

IV.I.I Análisis de los Parámetros del Sistema

El sistema que se pretende realizar, es un sistema de captura por hoja electrónica para sustituir la captura en hoja de papel, automatizar el conteo y la valoración de las preguntas para su posterior interpretación.

Cada hoja tiene su propio orden y número de preguntas con valoración, abarcando distintos aspectos.

Son siete tipos de hojas que describirán diferentes aspectos del perfil del alumno.

Una vez almacenados los datos en la base, se podrá extraer la información para su análisis.

Se creará un manejo de usuarios de modo que los alumnos podrán acceder para poder contestar su test, mientras existirá otro tipo de usuario que podrá hacer la consulta de los test respondidos para poder realizar la interpretación y finalmente existirá un usuario que tendrá la capacidad de administrar el sistema.

Este tipo de hoja se realizará en un formulario de HTML con conexión a la base de datos mediante PHP. El sistema estará montado sobre un servidor para poder acceder de vía web.

IV.I.II Análisis de los Requerimientos

Se requiere un servidor sobre el que este montado el sistema, por compatibilidad se puede usar un servidor web **Apache**.

De igual modo se necesita una base de datos, el montaje de esta se hará con el **DBMS MySQL**. Para el diseño del Modelo E - R haremos uso de un software llamado **MySqlWorkbench** que es una herramienta **CASE**, el cual nos permite trabajar de forma gráfica los elementos de la base de datos y posteriormente exportar el código de creación de la base al manejador de base de datos.

Al final podemos emplear diferentes programas que ya traen el montaje de ambos con su respectivo sistema operativo y el interprete del lenguaje de programación, tanto el servidor web como el DBMS, del tipo **LAMP** (Linux, Apache, MySQL y PHP) o **WAMP** (Windows, Apache, MySQL y PHP) para el desarrollo aunque es aconsejable dejar en el servidor final instalado todo de manera independiente para incrementar un poco la seguridad.

Necesitamos por último el lenguaje de programación para realizar el sistema, se eligió **PHP**, lo único que se necesita para programarlo es un **editor de texto plano o código** (ya que al ser interpretado no se compila), si se quiere tener un ambiente gráfico más amigable podremos utilizar **Dreamweaver** catalogado como una herramienta **CASE,** aunque este requiere de licencia para su uso.

IV.I.III Justificación de Elementos Seleccionados

La justificación se basa principalmente en dos puntos importantes.

La primera es que todos los elementos seleccionados para la realización del sistema, son de uso libre, es decir, no se necesita licencias para su uso.

La segunda justificación sobre el lenguaje de programación seleccionado, que es PHP, es que el mantenimiento y cambios a este tipo de sistemas, desarrollado en este lenguaje de programación, es sencillo puesto la forma de programar está basada en programación estructurada, y a la vez, se puede implementar programación orientada a objetos si se tiene el conocimiento, además de no requerir algún complemento extra para correrlo de lado del cliente más que su navegador web. Un desarrollo sobre Java en necesitará tenerlo instalado, así como .NET necesita el framework instalado.

El manejador de base de datos, MySQL, tiene también alta compatibilidad con el lenguaje de programación, aunque pudiera utilizarse otro como es PostgreSql, pero dado que la instalación nos proporciona todos los elementos sobre MySQL y la base de datos no necesita ser tan robusta, es el que se implementará.

IV.I.IV Casos de Uso y modelo de desarrollo usado

El modelo de desarrollo usado será el de cascada o clásico-tradicional con los pasos de: análisis de requisitos, diseño del sistema, diseño del programa, codificación o programación, pruebas, verificación y mantenimiento. Se ajusto apropiadamente ya que los requisitos fueron fijos desde un inicio por lo que la linealidad no se perdió. El desarrollo estará preparado para convertirse si es necesario en un modelo incremental ya que quedará la documentación necesaria y el sistema está diseñado de manera escalable. El modelo del tipo Desarrollo Rápido de Aplicaciones (DRA) no es tan apto pues llega a tener problemas en la documentación por ser tan ágil.

<u>Administrador - Alta de Administradores</u>

Un administrador principal podrá dar de alta administradores secundarios con un cierto grado de privilegio para manipular los resultados obtenidos por el sistema sobre los formularios contestados por los alumnos. Podrán de la misma forma dar de alta alumnos al sistema una vez registrados.

Se tiene planeado tener dos niveles de administradores, uno que tenga dominio total sobre el sistema y otros que solo tengan acceso a datos o alta de alumnos.

Administrador - Baja de Administradores

Podrá existir la opción para dar de baja administradores, cambiando la bandera del registro de activo a inactivo puesto que es importante no borrar estos registros.

Administrador - Modificación de Administradores

Se podrá modificar los datos o privilegios de los administradores. Solo un administrador primario podrá dar privilegios a los secundarios.

Administrador - Alta de Alumnos

Los alumnos podrán ser dados de alta por los administradores para que posteriormente estos puedan llenar los formularios.

Administrador - Baja de Alumnos

Si algún alumno se ya no requiere ser evaluado o abandona la carrera, podrá ser dado de baja en el sistema cambiando la bandera de estado de activo a inactivo pero conservando su registro.

Administrador - Modificación de Alumnos

Se podrá modificar los datos de los alumnos, si es que es requerido o existió un error en captura, esto solo por los administradores.

Alumno - Resolución de Cuestionario

Una vez registrado el alumno, podrá accesar al sistema mediante su nombre de usuario y contraseña, para resolver los seis formularios que se le presentaran, una vez que acceso al sistema tendrá que terminar por completo, pero ya que así lo requiere el análisis del perfil, más aún si llegase a existir algún problema como un falla eléctrica, por ejemplo, el sistema tendrá la capacidad de no perder por completo el avance y poder ubicar al alumno en la parte donde se quedó.

Administrador - Consulta de Resultados

El evaluador del perfil podrá, una vez que el alumno haya contestado en su totalidad los seis formularios, consultar los resultados sin tener que recurrir a la hoja de interpretación, ya que esto lo realizará el sistema automáticamente.

Existirá una sección donde se graficaran los resultados.

Finalmente es importante mencionar que el sistema no es un administrador de contenido y como se mencionó anteriormente los requerimientos como el cuestionario fue definido desde un inicio por lo que no requiere ser modificado o crear uno nuevo.

IV.I.V Creación del Modelo E – R

El modelo Entidad-Relación, es una representación gráfica de los elementos que percibimos en el mundo real, se basa bajo objetos definidos como entidades y las relaciones que pueden existir entre ellos. Al tener este diagrama nos será mucho más sencillo realizar el modelo Relacional.

Podemos observar 13 entidades en nuestro modelo.

Relacionadas entre sí, existe la entidad **administrador** con la entidad **alumno** la cual seguirá la lógica de que un administrador puede dar de alta a muchos alumnos (1:M). La llave primaria de **administrador** podremos encontrarla como foránea de **alumno**.

La entidad **alumno** a su vez está relacionada con las entidades formulario **fa, fb, fc, fd, fe, ff, fg, fh, fi** de manera que la relación existente es uno a uno (1:1), el atributo número de cuenta (*noCuenta*) es la llave primaria tanto de la entidad **alumno** como de las entidades **formulario** y a la vez una llave foránea de este último que permitirá hacer la relación entre ellos.

Existen dos entidades que no tienen relación con las demás que son Normas Percentiles para los sexos masculino y femenino **nPercentilM** y **nPercentilF**, pero se están proyectando de manera que estas contengan información que posteriormente pueda ser consultada varias veces en distintas aplicaciones, la razón es que si esta información es programada en alguna sección de código, y si es necesaria posteriormente, se puede volver redundante. Pudiese extraerse desde código mediante una función pero el manejo de la información es mucho más sencillo desde la base de datos.

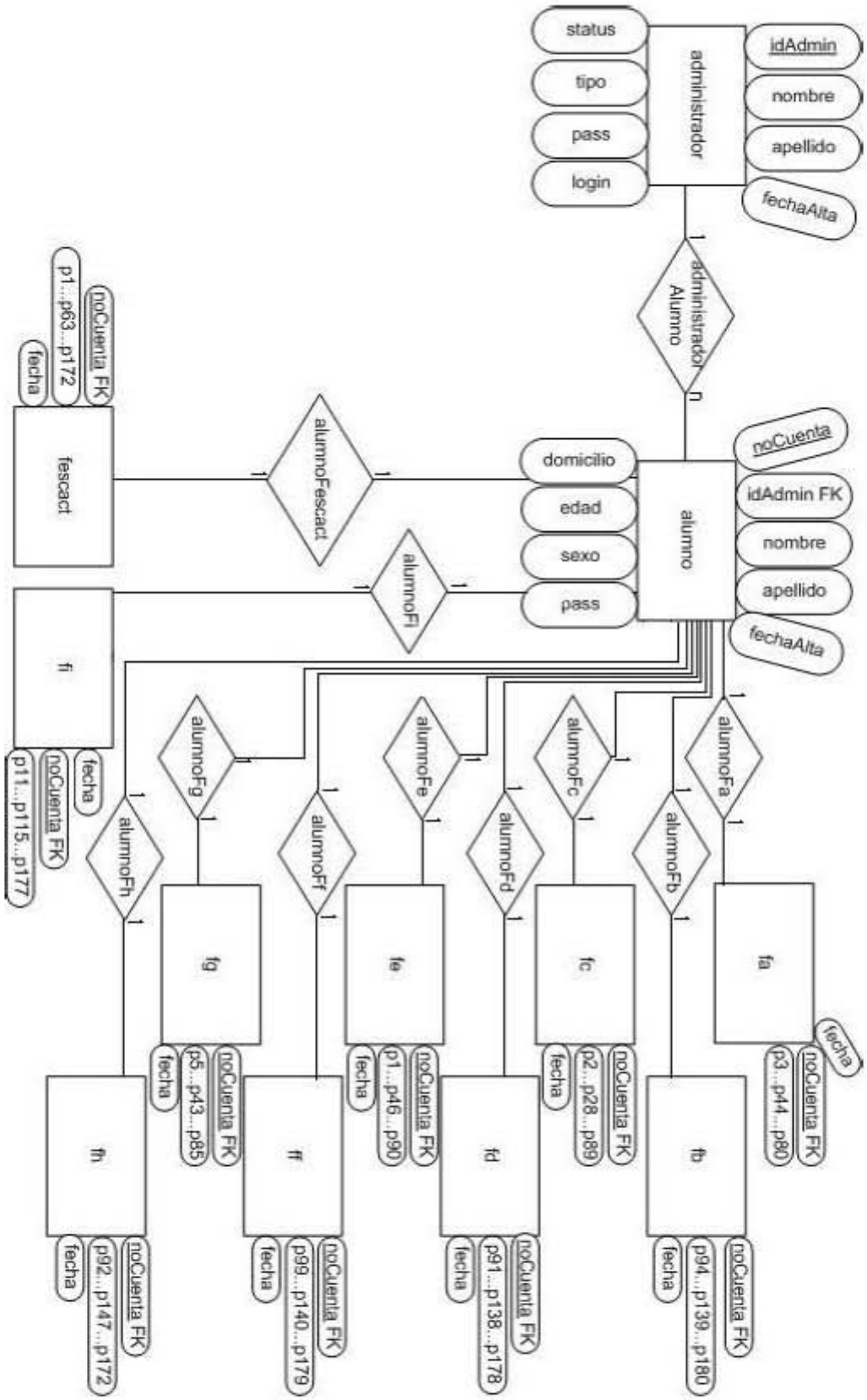


Figura 4.1 Diagrama E - R

IV.I.VI Creación del Modelo Relacional

La creación del Modelo Relacional nos ayuda a planear que tablas integraran nuestra base de datos más en forma, los elementos que la integraran con su respectivo tipo de dato y las relaciones que existirán entre ellas.

Este modelo es extraído con mayor facilidad si es que ya tenemos construido el modelo E-R, puesto que gráficamente tenemos la ubicación de las tablas y sus campos.

Como podemos observar tendremos una tabla **alumno** la cual contendrá los datos de este, a parte su contraseña de acceso y una llave foránea que muestra quien le dio de alta, con el ID de Administrador. Si es necesario se puede crear una tabla de logs sobre cambios en los alumnos si estos se volvieran muy recurrentes aunque para este desarrollo no fueron tan necesarios.

Tenemos también una tabla de **administrador** la cual no se integró como una sola a la anterior en una tabla de usuarios puesto que los datos son distintos y de esta forma asegura mayor seguridad de acceso.

Existe además una relación entre los alumnos y administrador puesto que los primeros serán dados de alta por los segundos ya que será una de las aplicaciones del sistema. Como la relación que existe es 1:M no hay necesidad de crear una tabla intermedia.

Por último tenemos las tablas de los formularios con alguna relación, en total son 9 y están relacionados a la tabla de **alumno** puesto que cada registro del formulario pertenece a una alumno.

Y finalmente las tablas independientes de **percentil** que contendrán información de consulta, sin relación alguna con alguno alumno en específico o administrador.

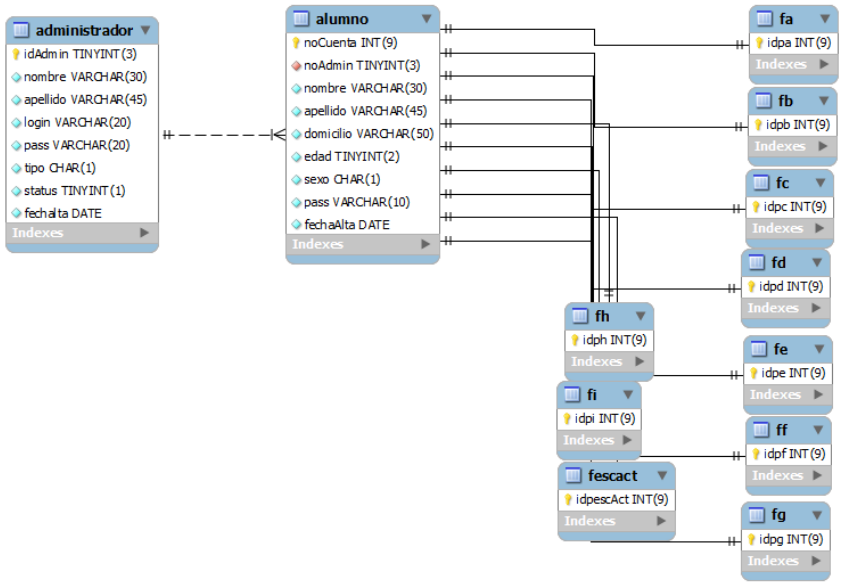


Figura 4.2 Diagrama Relacional Compacto

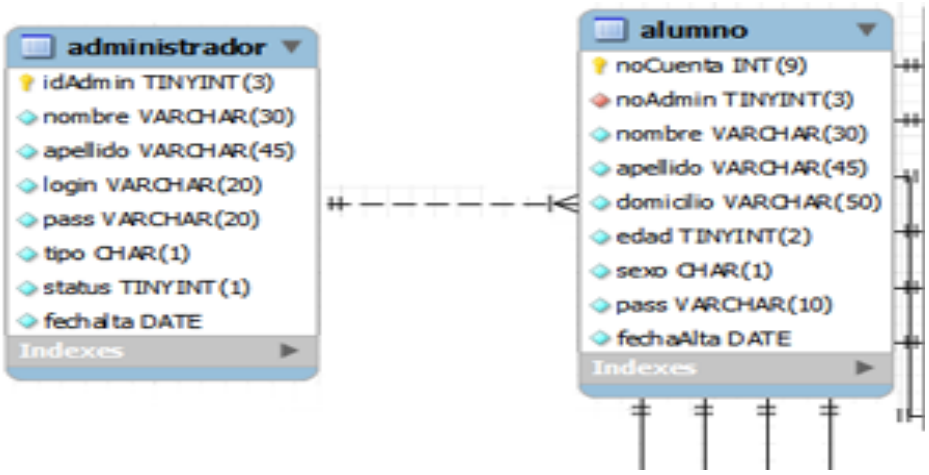


Figura 4.3 Tablas de administrador y alumno

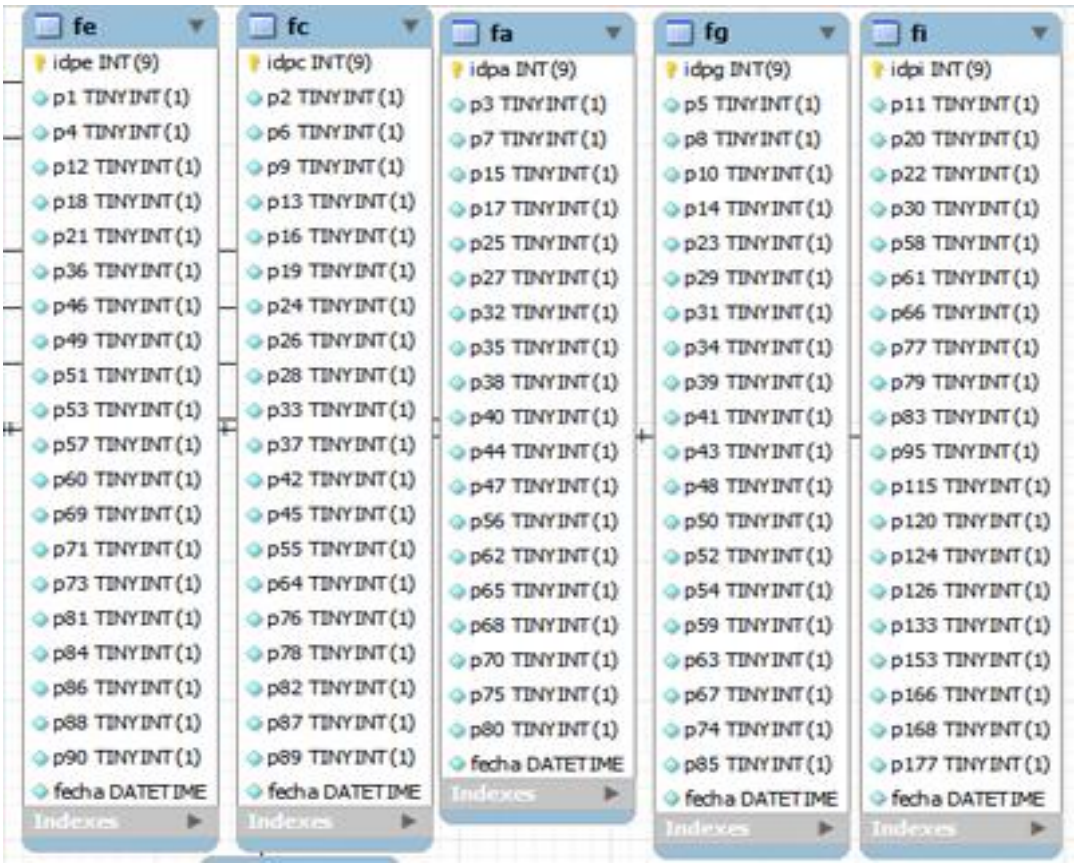


Figura 4.4 Tablas formulario

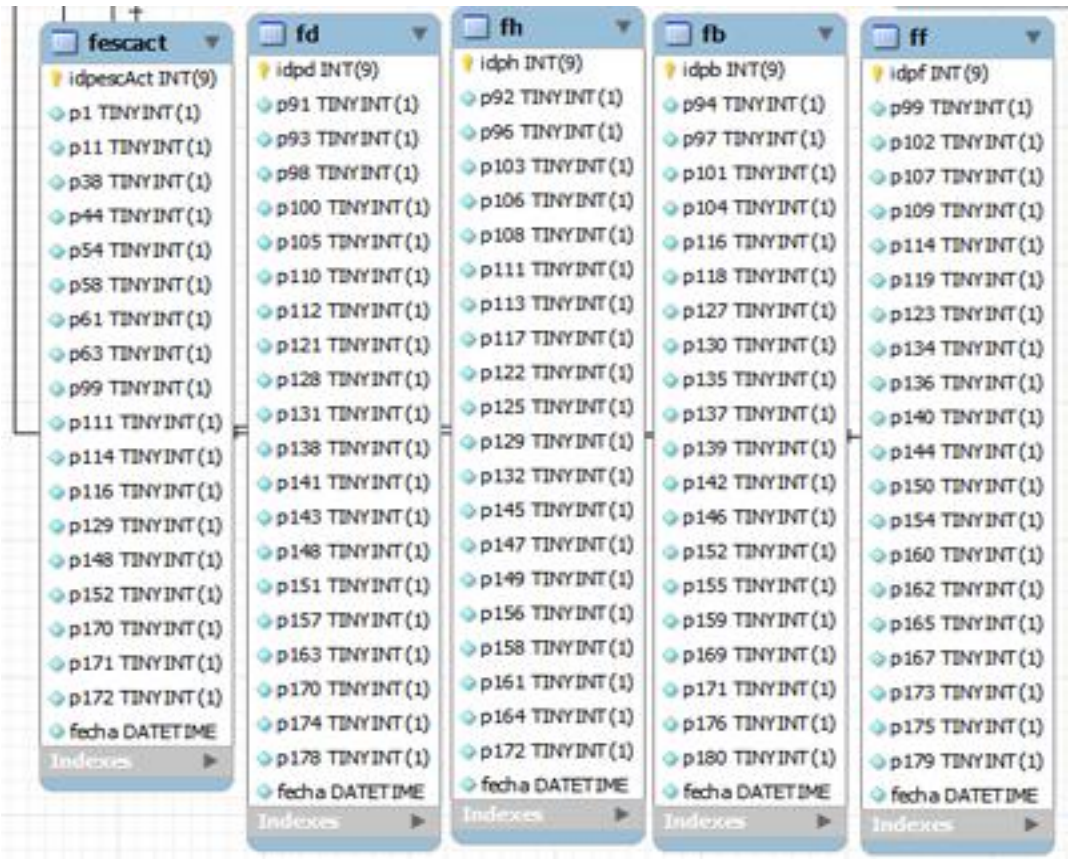


Figura 4.4 Tablas formulario

IV.II Desarrollo del Sistema

IV.II.I Montaje del Servidor

El montaje del servidor de en ambiente de desarrollo se trabajo mediante paquetería que contiene al servidor web, el manejador de base de datos y los módulos ya instalados y configurados de PHP.

El paquete utilizado se llama XAMPP y la instalación básica es realmente sencilla, puesto que al descargarse este servidor cuenta con su propio descompresor e instalador. Contiene un servidor web Apache, el manejador de base de datos MySQL y como ya se mencionó los módulos de PHP ya configurados por defecto.

Una de las observaciones que hay que tener al manejar el módulo, es tener cuidado con la versión de instalación ya que muchas veces suele pasar que al migrar de una plataforma a otra, la versión de PHP puede ser una más actual y esto conlleva a actualizaciones de seguridad que muchas veces al utilizar librerías viejas puede llegar a marcar este tipo de errores, de seguridad.

Otro error muy recurrente al utilizar librerías o código antiguo es el de como interpretación de llaves de la etiqueta (o tag) para insertar dentro del html el código PHP del tipo:

<? **<?php**

por

?> **?>**

lo cual provoca que marque un error de interpretación.

Una solución a este problema es modificar en el archivo de configuración php.ini la línea short_open_tag a "On":

short_open_tag = On

a partir de de PHP5 fue que se introdujo este cambio en la llave de la etiqueta pero no es recomendable ya que si la programación llega a migrar a otro servidor donde no esté activada esta opción de configuración llegara a marcar error en la interpretación.

En las siguientes ventanas podemos observar la versión que quedó montada en el servidor, versión del servidor web, versión de manejador de base de datos y versión de PHP así como su configuración.

PHP 5.3.5

XAMPP for Windows English / Deutsch / Francais / Nederlands / Polski / Italiano / Norwegia / Português (Brasil) / 日本語

PHP Version 5.3.5

System	Windows NT MAXWELL 5.1 build 2600 (Windows XP Professional Service Pack 3) i586
Build Date	Jan 6 2011 17:50:45
Compiler	MSVC6 (Visual C++ 6.0)
Architecture	x86
Configure Command	cscript /nologo configure.js "--enable-snapshot-build" "--disable-isapi" "--enable-debug-pack" "--disable-isapi" "--without-mssql" "--without-pdo-mssql" "--without-pi3web" "--with-pdo-oci=D:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8=D:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8-11g=D:\php-sdk\oracle\instantclient11\sdk,shared" "--enable-object-out-dir=../obj/" "--enable-com-dotnet" "--with-mcrypt=static"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	C:\WINDOWS
Loaded Configuration File	C:\xampp\php\php.ini

Figura 4.5 Configuración PHP

Apache 2.2.17

apache2handler

Apache Version	Apache/2.2.17 (Win32) mod_ssl/2.2.17 OpenSSL/0.9.8o PHP/5.3.4 mod_perl/2.0.4 Perl/v5.10.1
Apache API Version	20051115
Server Administrator	postmaster@localhost
Hostname:Port	localhost:80
Max Requests	Per Child: 0 - Keep Alive: on - Max Per Connection: 100
Timeouts	Connection: 300 - Keep-Alive: 5
Virtual Server	No
Server Root	C:/xampp/apache
Loaded Modules	core mod_win32 mpm_winnt http_core mod_so mod_actions mod_alias mod_asis mod_auth_basic mod_auth_digest mod_authn_default mod_authn_file mod_authz_default mod_authz_groupfile mod_authz_host mod_authz_user mod_autoindex mod_cgi mod_dav_lock mod_dir mod_env mod_headers mod_include mod_info mod_isapi mod_log_config mod_mime mod_negotiation mod_proxy mod_proxy_ajp mod_rewrite mod_setenvif mod_ssl mod_status mod_php5 mod_perl

Figura 4.6 Configuración Apache

MySQL 5.0.7

mysql

MySQL Support	enabled
Active Persistent Links	0
Active Links	0
Client API version	mysqlnd 5.0.7-dev - 091210 - $Revision: 304625 $

Directive	Local Value	Master Value
mysql.allow_local_infile	On	On
mysql.allow_persistent	On	On
mysql.connect_timeout	3	3
mysql.default_host	*no value*	*no value*
mysql.default_password	*no value*	*no value*
mysql.default_port	3306	3306
mysql.default_socket	MySQL	MySQL
mysql.default_user	*no value*	*no value*
mysql.max_links	Unlimited	Unlimited
mysql.max_persistent	Unlimited	Unlimited
mysql.trace_mode	Off	Off

Figura 4.7 Configuración MySql

La configuración inicial del paquete establece el uso de las variables globales desactivadas, esto por razones de seguridad, pero para el manejo de sistema en algunas aplicaciones, se utilizarán, por lo que abra que activarlas.

La manera de activarlas es en el archivo de configuración **php.ini**

En la línea de código

register_globals = Off

hay que cambiar por

register_globals = On

de esta manera observaremos lo siguiente en la propiedades de configuración:

register_argc_argv	Off	Off
register_globals	On	On
register_long_arrays	Off	Off

Figura 4.8 Configuración Registros Globales

La manera de ejecutar el servidor web y el administrador de la base de datos es muy sencilla, basta con ejecutar la aplicación una vez instalada y aparecerá una ventana similar a esta:

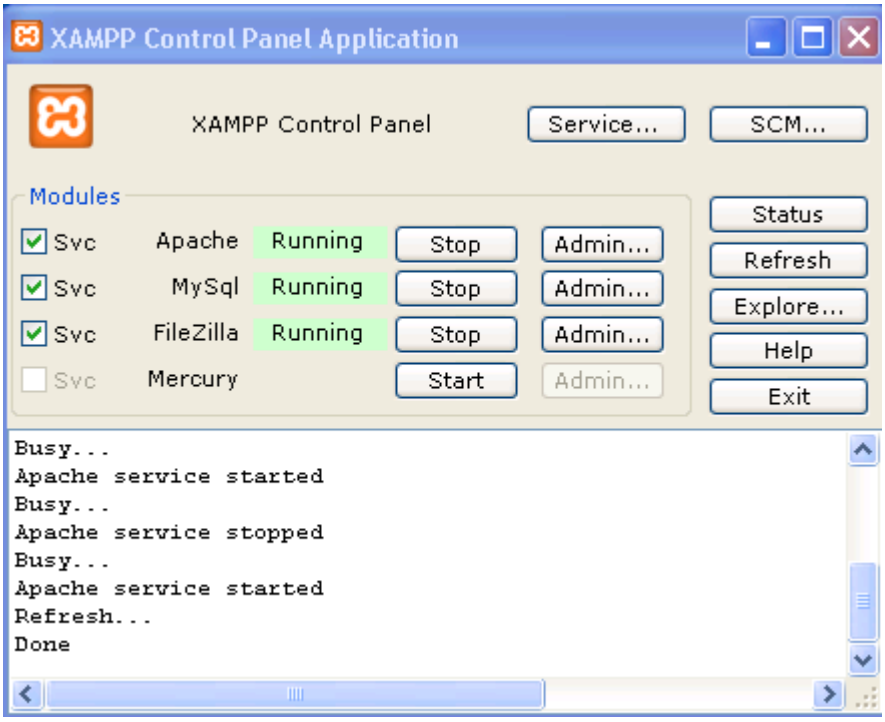


Figura 4.9 Panel de servicios

Si se instaló correctamente la aplicación las banderas que se observan en la figura deberán estar en verde, avisando que los servicios están levantados.

La carpeta **htdocs** es nuestra ruta del servidor local o localhost donde podremos administrar diferentes carpetas para su publicación web, por defecto viene vacía. La forma de accesar es mediante un navegador web como se muestra en la siguiente imagen:

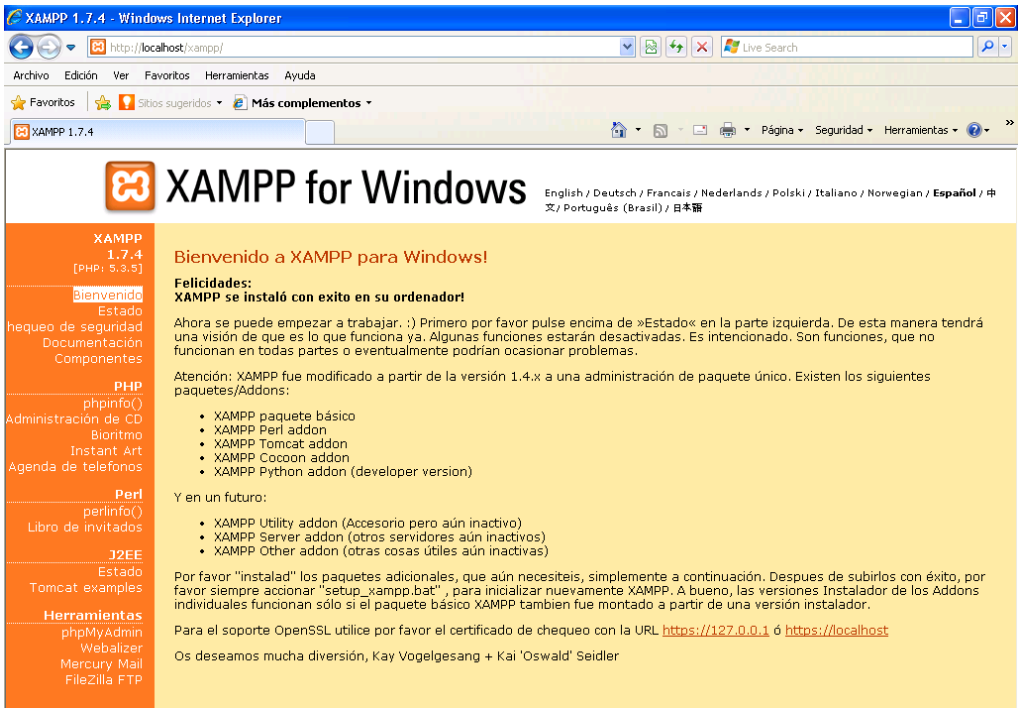


Figura 4.10 Pantalla de confirmación de servicio Apache encendido

El montaje del servidor real es otro proceso totalmente diferente ya se recomienda por razones de seguridad no se instale mediante paquetería, sino de manera separada el servidor web Apache, el manejador de base de datos y los módulos de manera manual pero para el alcance de la tesis nos bastará con un servidor de este tipo.

IV.II.II Exportación de la Base de Datos

Desde nuestro modelado de la base de datos hecho en **MySql Workbench** nosotros exportaremos el código que generará la base de datos. Una vez abierto el archivo donde se realizó el modelado y revisado a la perfección, en la barra de menú iremos a File -> Export ->Fordward Engineer SQL CREATE Script.. como se muestra en la siguente imagen:

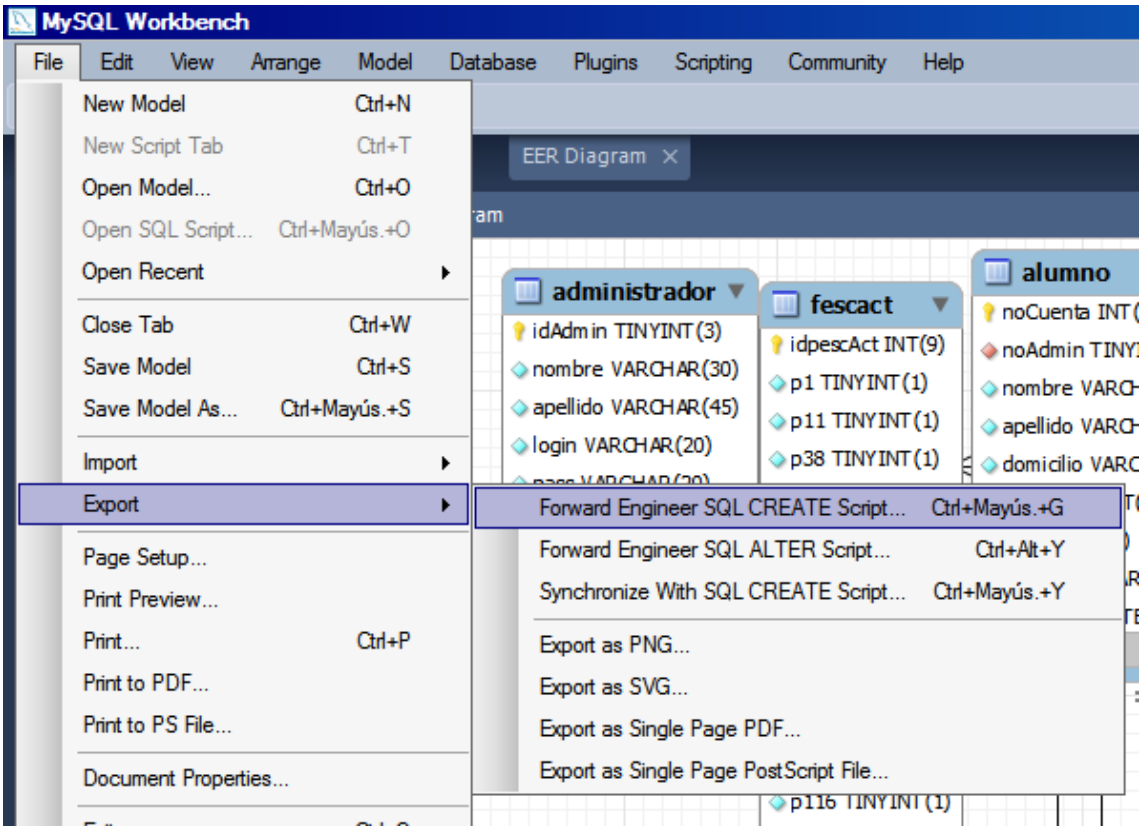


Figura 4.11 Método de exportación mediante MySql Workbench

Elegimos la ruta donde se guardará el archivo:

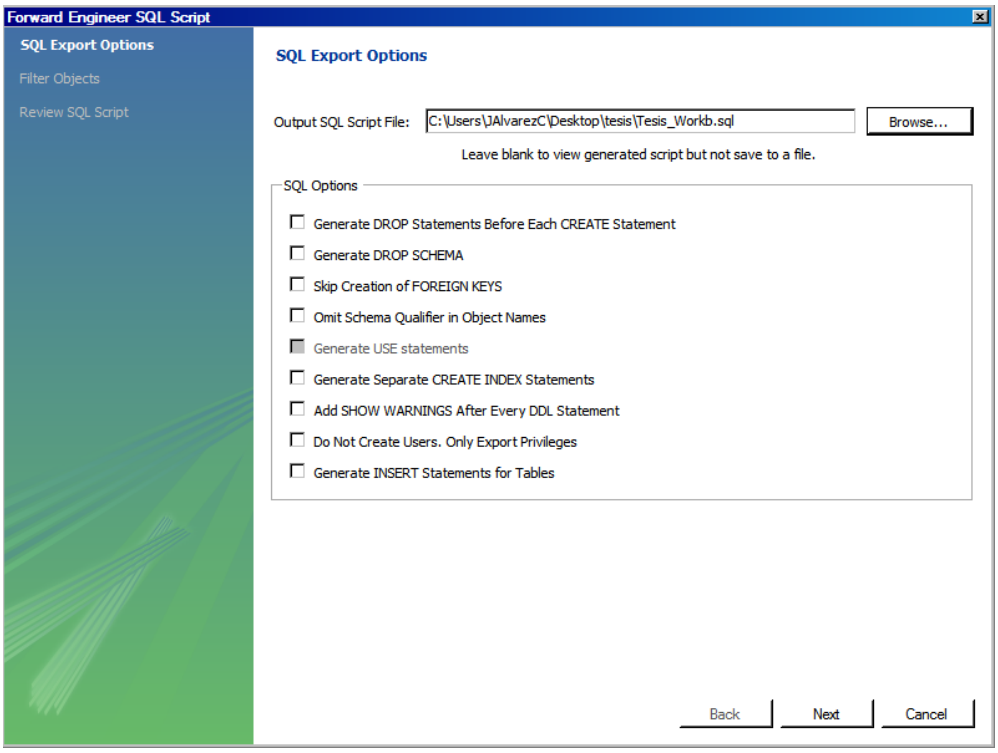


Figura 4.12 Ruta de archivo .sql a exportar

Elegimos de las tablas:

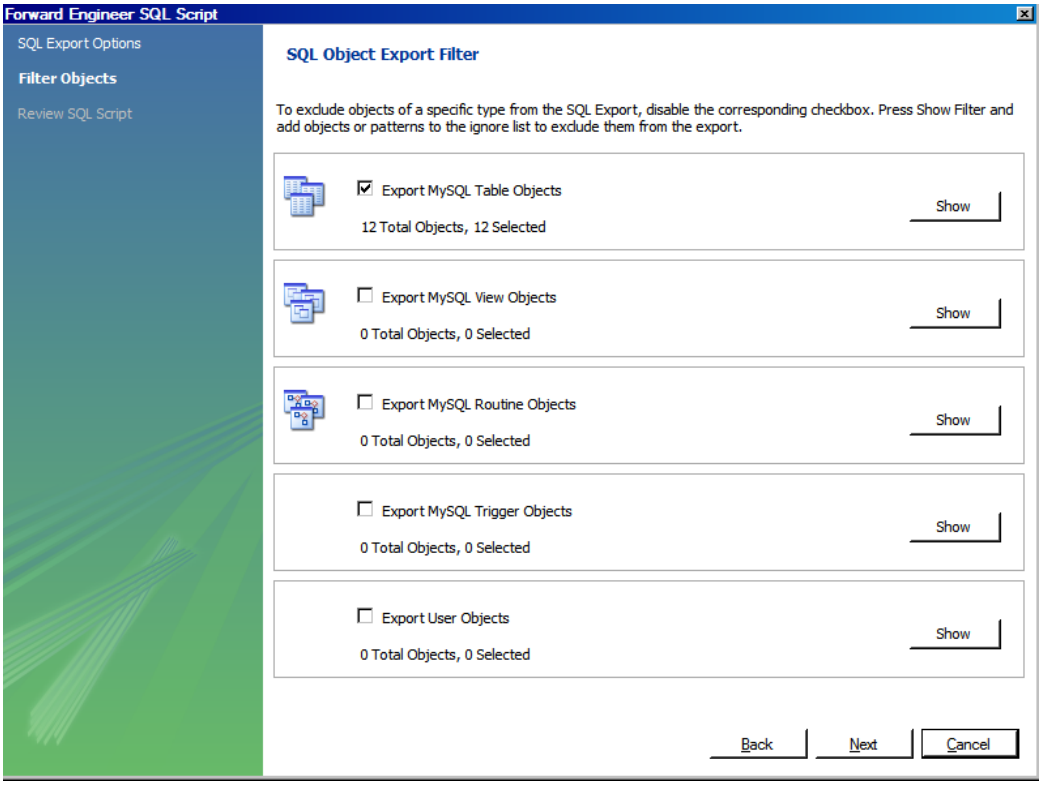


Figura 4.13 Opciones para la exportación del archivo

Y finalmente podremos observar el código SQL generado a partir de estos pasos:

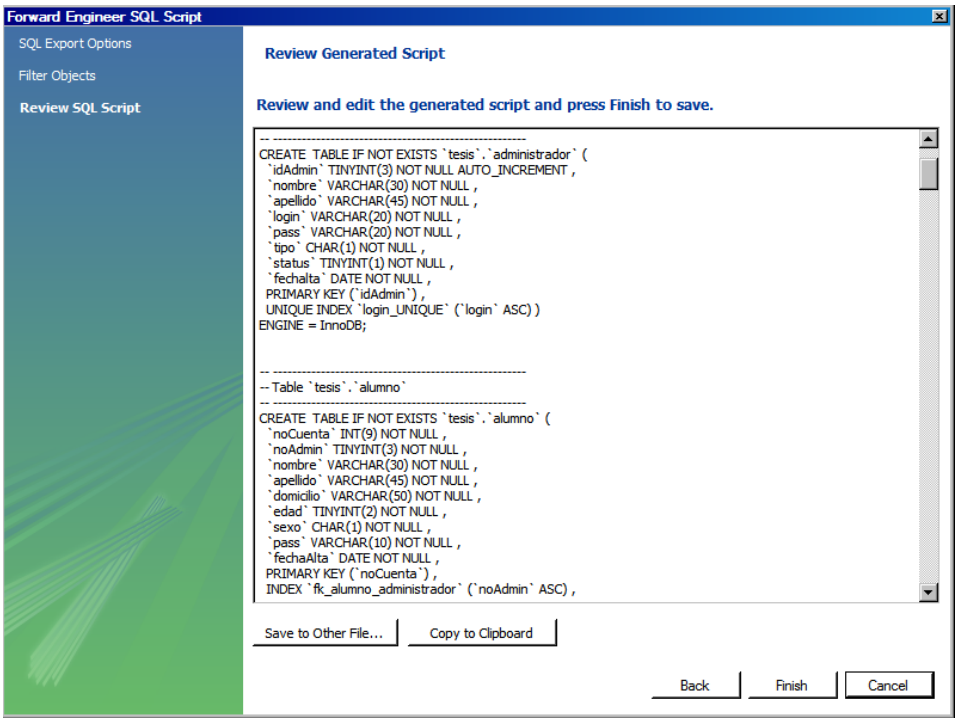


Figura 4.14 Ventana de confirmación para finalizar exportación

IV.II.II.I Herramienta Gráfica del DBMS

La base de datos en el servidor local viene integrada y configurada para su uso. La paquetería implementa un administrador gráfico de esta base llamada **phpMyAdmin**, el cual facilita el uso, administración y consulta de la base de datos.

Por defecto esta opción viene en las opciones del menú de inicio como se muestra en la siguiente figura:

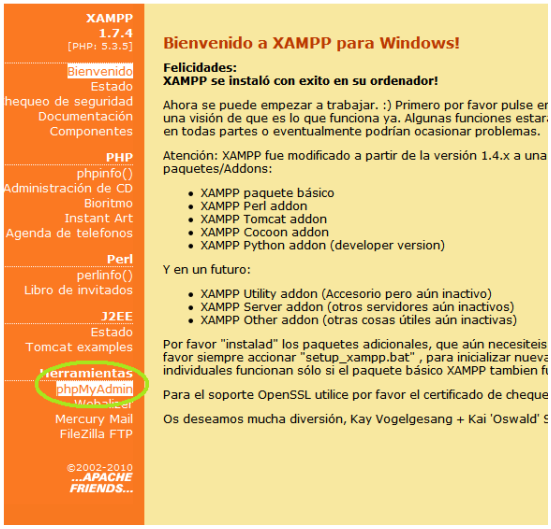


Figura 4.15 Opción de ingreso a PhpMyAdmin

o entrando directamente mediante la ruta http://localhost/phpmyadmin.

Al dar clic sobre este hipervínculo que apareció tendremos acceso a una herramienta gráfica del Administrador de Base de Datos, donde podremos crear usuarios con distintos privilegios, bases de datos, etc.

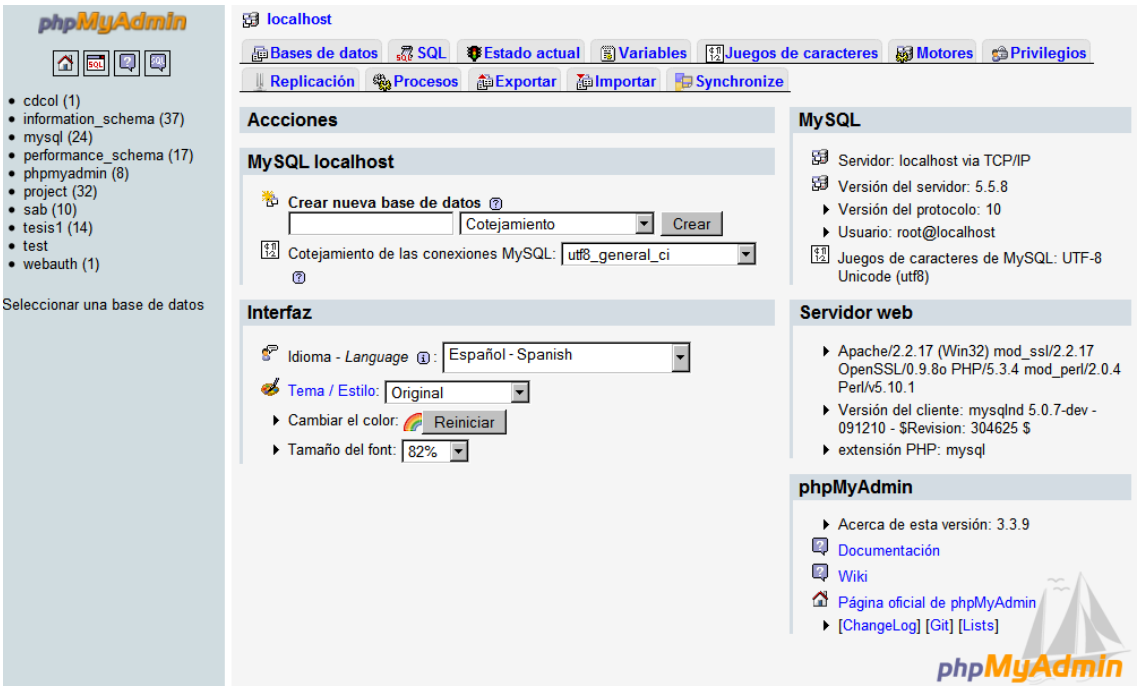


Figura 4.16 Ventana de panel de administración de PhpMyAdmin

IV.II.II.I Creación de la Base de Datos

Una vez que ya tenemos funcionando la herramienta gráfica el siguiente punto es crear la base de datos para en un siguiente paso, comenzar a trabajar sobre la programación del sistema.

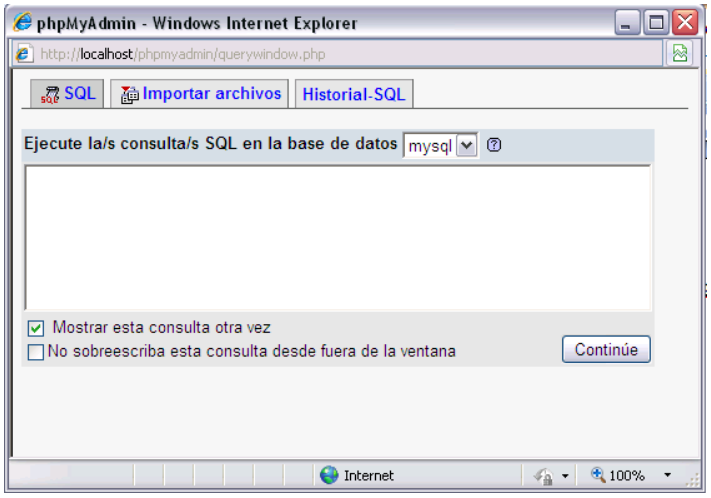


Figura 4.17 Ventana para ejecución e importación de lenguaje SQL

Se mencionó anteriormente que para proyectar el diseño de la base datos nos apoyamos en el software llamado **MySql Workbench** que permite la exportación de la base de datos.

En el capítulo **IV.I.VI Creación del Modelo Relacional**, podemos observar cómo fue diseñada la base de datos, tomando en cuenta sus relaciones, los tipos de datos de los campos entre otros. Una vez que tenemos guardado el proyecto, podemos exportar la base de datos mediante un script de SQL.

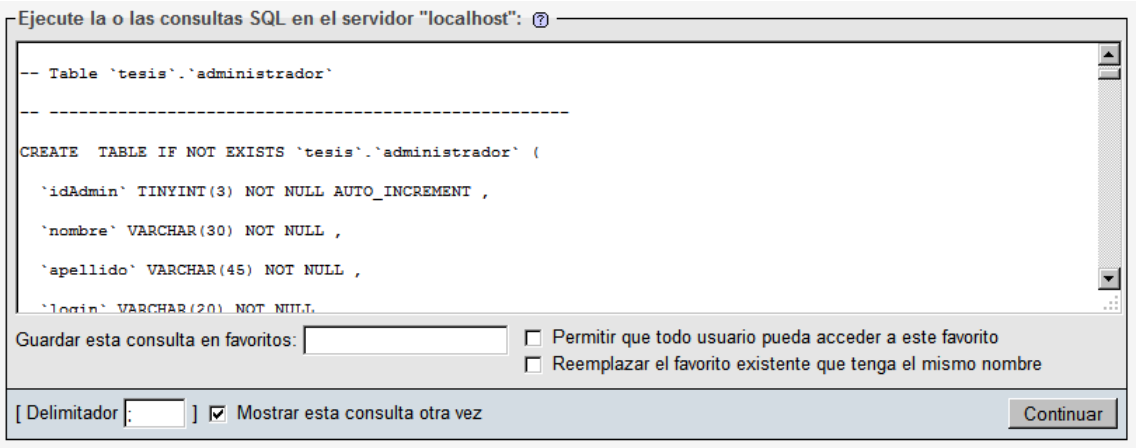


Figura 4.18 Importación de archivo .sql generado

Una vez que tenemos el script obtenido del **MySqlWorkbench** lo podemos cargar a la base de datos, ya sea importando todo el archivo que se creó, con extensión .sql o abriendo este archivo con un editor de código o texto plano y pegándolo en la casilla marcada SQL.

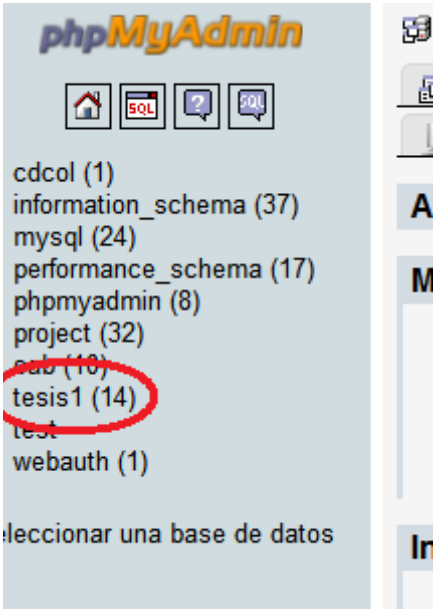


Figura 4.19 Base de datos generada a partir de la importación

Ya que hemos importado la base de datos podemos observar la creación de la misma de manera gráfica en la herramienta de phpMyAdmin.

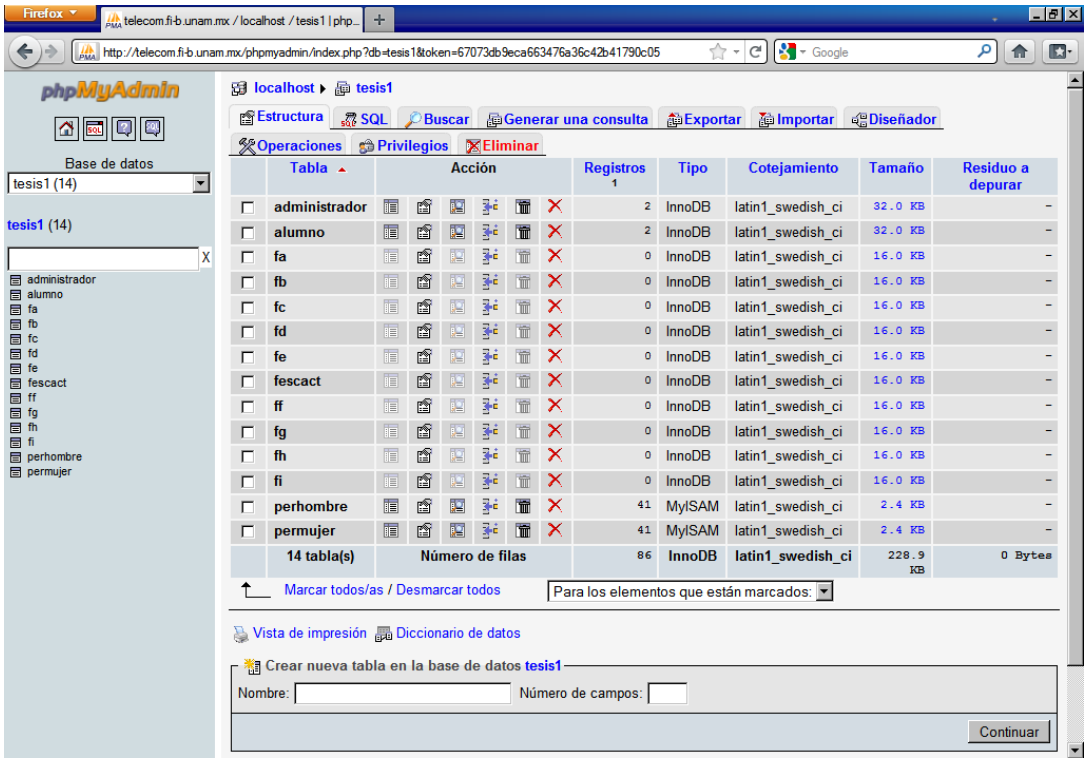


Figura 4.20 Listado de las tablas generadas

Cabe mencionar que con esto la base de datos ya está lista para usarse, puesto que ya se han fijado los tipos de datos para los campos, las distintas tablas, las relaciones y sus distintos tipos, llaves primarias y foráneas desde el modelador de la base de datos.

Podemos observar el diccionario de datos:

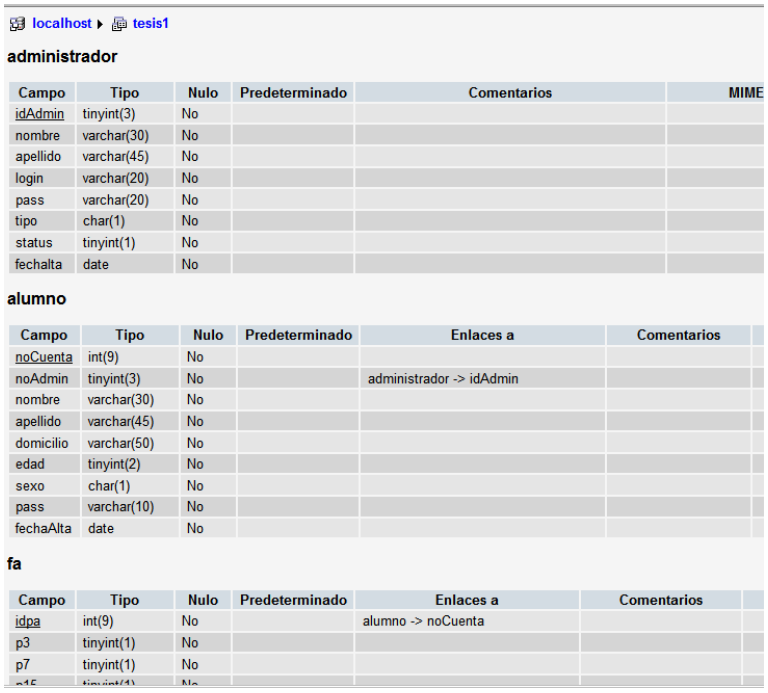


Figura 4.21 Vista de parte del diccionario de datos

IV.II.III Programación del Código

Una vez que ya tenemos diseñada la base de datos y exportada al manejador, podemos continuar con el diseño del sistema.

Lo primero que tenemos que crear es una carpeta de trabajo dentro de nuestra carpeta de publicación web **htdocs**, para este caso se ha creado con el nombre de **tesis**, y dentro de esta guardaremos el código del sistema para que pueda ser interpretado el servidor, esta al ser creada, aparecerá como se muestra en la siguiente figura:

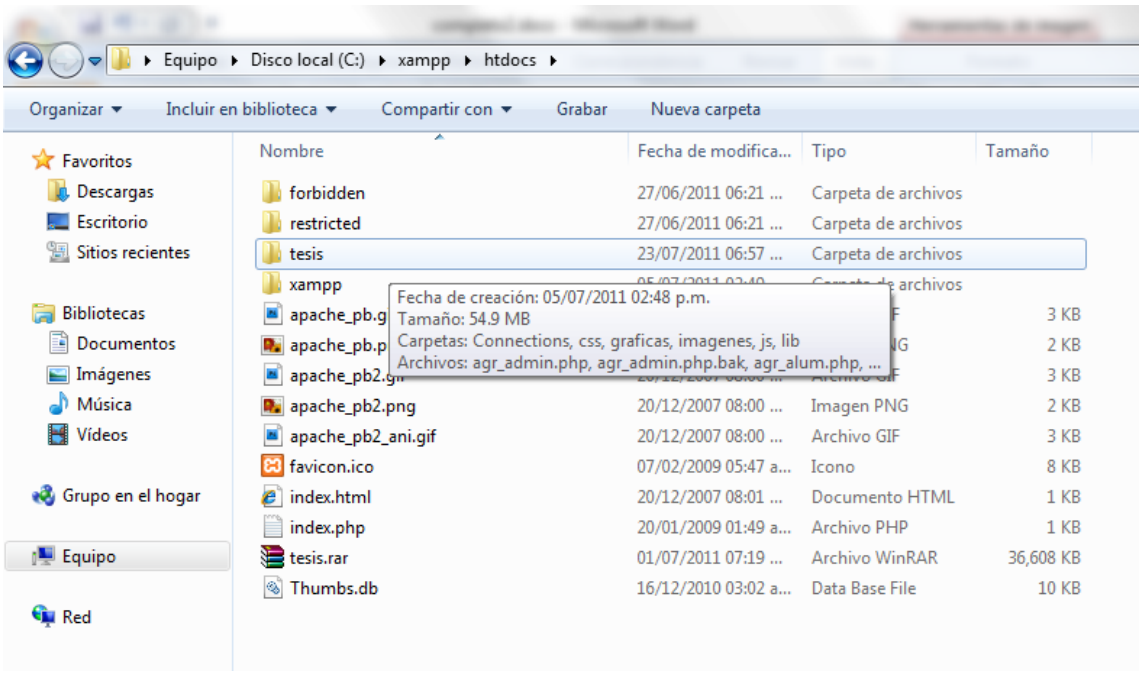


Figura 4.22 Carpeta de publicación del servidor local instalado

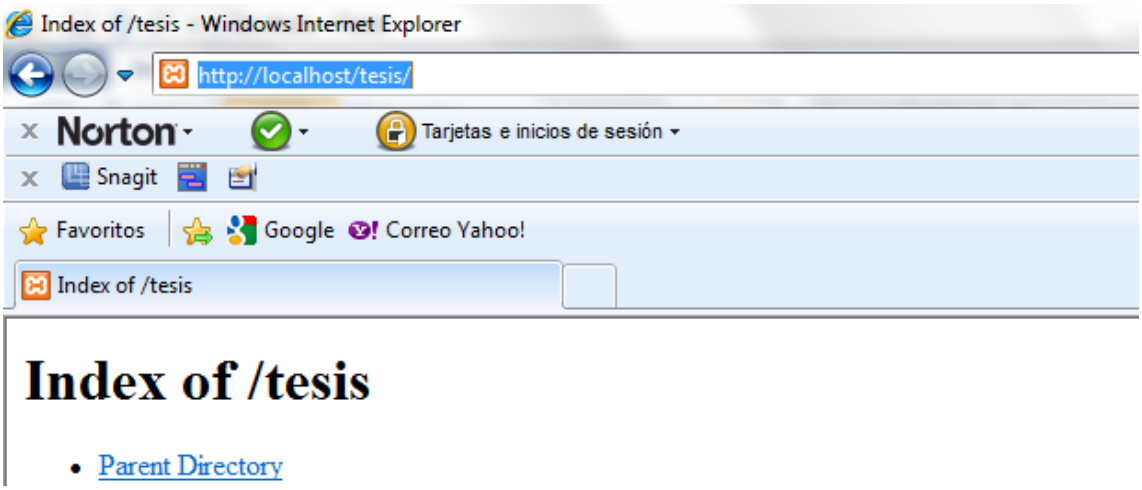


Figura 4.23 Vista de la carpeta generada para el código fuente

IV.II.III.I Administradores

El primer punto que tomaremos en cuenta es el manejo de administradores, para esto necesitamos una sección que nos ayude a su administración por lo que hemos creado un menú.

Para su acceso el administrador ingresará mediante un nombre de usuario y contraseña:

Figura 4.24 Ventana de inicio de sesión de administración

Menú Administrador

- ***Manejo de Seguridad***

La seguridad que se está implementado en el acceso al menú de administrador esta dada mediante el manejo de **setcookies** configuradas de manera que el tiempo de vida no se encuentra definido de esta manera la duración de la sesión terminará al cerrar el navegador.

La seguridad también esta implementada mediante el manejo de nivel de usuarios. Dentro de la base de datos en la tabla correspondiente a **administrador** existe un campo asignado para esta tarea, llamado **tipo** definiendo el tipo de usuario y en nivel de privilegios que tiene concedidos.

Otro campo en esta misma tabla que nos ayudará a tener un control de seguridad, es el campo **status** el cual será consultado al ingresar con una cuenta de usuario y se cerciorará que el usuario no haya sido dado de baja pretendiendo que ya no tenga privilegios de acceso.

La sentencia SQL asignada para esta tarea es la siguiente:

```
$sql = mysql_query("select * from administrador where login='$prime'
and pass='$segun' and status=1") or die ($Sql .mysql_error()."" ) ;
```

Figura 4.25 Sentencia de consulta para validación de administrador

donde:

login y pass son valores obtenidos del formulario de dos campos donde se está ingresando al menú de administrador, las variables son obtenidas mediante el método POST,

tipo = ‘A’ es el nivel más alto de privilegios en el sistema,

status = 1 marca que el usuario no ha sido dado de baja en el sistema.

Una vez que se comprueban los valores existen dos opciones: que el sistema conceda el acceso al usuario, o le sea negado en caso de que algún campo del formulario de acceso este incorrecto, no tenga los privilegios suficientes o haya sido dado de baja dentro del sistema apareciendo la siguiente ventana emergente:

Figura 4.26 Mensaje de error en usuario o contraseñas inválidos.

Si el acceso es correcto para el primer nivel tendremos este menú:

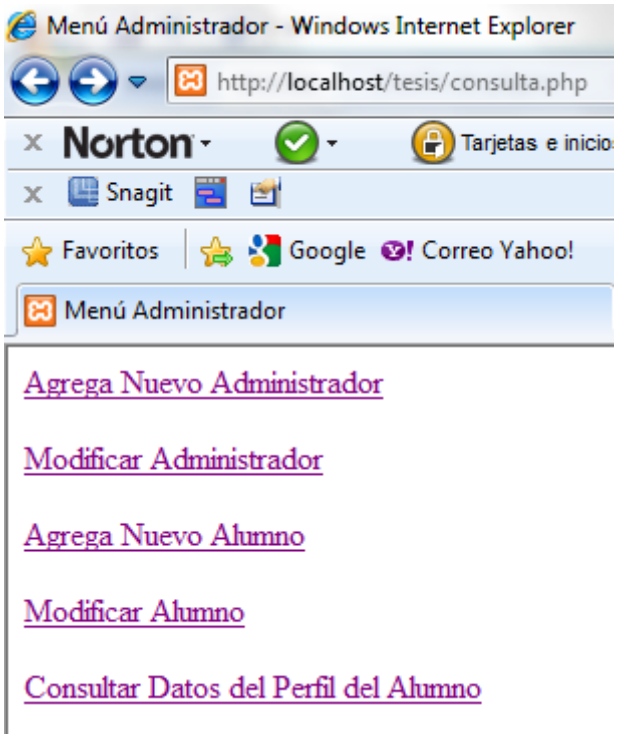


Figura 4.27 Menú de opciones de administrador principal

Se tiene pensado en 2 niveles de privilegios, el más alto, podrá dar de alta tanto nuevos administrador, como alumnos y podrá hacer consultas de la información.

Para nuestro segundo nivel de usuario tendremos solo la siguiente opción del menú:

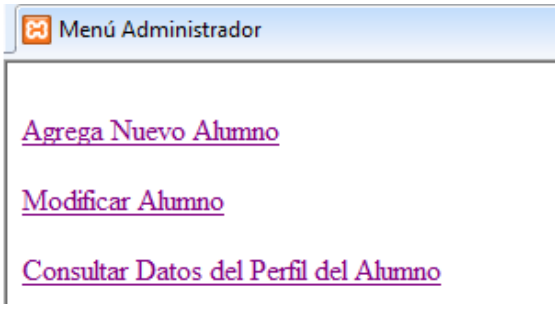


Figura 4.28 Menú de opciones de administrador secundario

El segundo solo podrá dar de alta alumnos y consultar información.

La manera de aislar las opciones es mediante la variable de “tipo” obtenida de nuestro archivo de consulta en la sección que envía los datos al archivo de menú:

```
setcookie ("nombre",trim($vnombre));

$login=$vlogin;
$tipo=$vtipo;

include('menuAdmin.php');

}
else
{
```

Figura 4.29 Restricción de perfil por tipo de administrador

Que finalmente en el archivo de menú lo usará con una condición de la siguiente manera:

```
<?php

if($tipo=='A'){
print'
    <a href="agr_admin.php">Agrega Nuevo Administrador</a><br>
    <br>
    <a href="lista_mod_admin.php">Modificar Administrador</a><br>
';
}
?>
<br>
<a href="agr_alum.php">Agrega Nuevo Alumno</a><br>
<br>
<a href="lista_mod_alum.php">Modificar Alumno</a><br>
<br>
<a href="perfil.php">Consultar Datos del Perfil del Alumno</a><br>
<br>
```

Figura 4.30 Condición de opciones en menú por tipo de administrador

En esta lógica existirá un hipervínculo en el menú de administración llamado Agrega Nuevo Administrador el cual se mostrará de la siguiente manera:

Agregar Nuevo Administrador

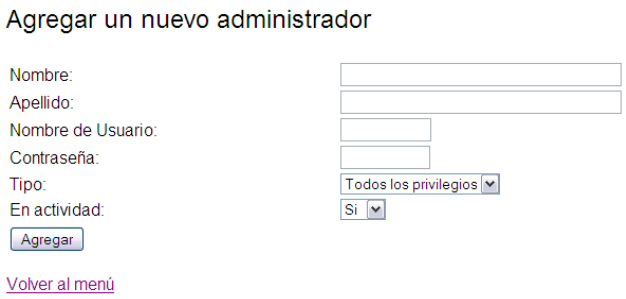


Figura 4.31 Formulario de alta para nuevo administrador

- ***Manejo de Seguridad en archivos***

La seguridad que se está implementando mediante programación en PHP, tomando al usuario y su contraseña obtenidos del acceso a menú mediante variables globales, esto de manera que si se intenta entrar directamente a la ruta del archivo y no cuenta con su usuario y contraseña mandará la ventana de error:

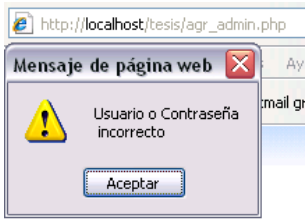


Figura 4.32 Menú de opciones de administrador secundario

El código que se usa para esto es el siguiente:

```
<?php
 global $prime,$segun,$login,$pass,$tipo;

 if (!empty($prime) && !empty($segun))
 {
 if ($login==trim($prime) && $pass==trim($segun))
  {
?>

        CODIGO USADO

<?php
  }
  else
  {
       print '<script language="Javascript">';
       print 'alert("Usuario o Contraseña \n incorrecto"); javascript:self.history.back();';
       print '</SCRIPT>';
  }
}
else
{
       print '<script language="Javascript">';
       print 'alert("Usuario o Contraseña \n incorrecto"); javascript:self.history.back();';
       print '</SCRIPT>';
}
?>
```

Figura 4.33 Estructura de bloque de seguridad

de manera todo lo que se encuentre en la línea **CODIGO USADO** no podrá ser usado y visto si no cumple con la condiciones de tener un usuario y una contraseña.

- ***Código de Validación***

Se compone de un filtro de validación:

Esta sección es donde se encuentran el código PHP referente a validación de datos, filtro de inyección SQL, y conexión e inserción de datos a la base de datos.

```php
function GetSQLValueString($theValue, $theType, $theDefinedValue = "", $theNotDefinedValue = "")
{
  $theValue = (!get_magic_quotes_gpc()) ? addslashes($theValue) : $theValue;

  switch ($theType) {
    case "text":
      $theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
      break;
    case "long":
    case "int":
      $theValue = ($theValue != "") ? intval($theValue) : "NULL";
      break;
    case "double":
      $theValue = ($theValue != "") ? "'" . doubleval($theValue) . "'" : "NULL";
      break;
    case "date":
      $theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
      break;
    case "defined":
      $theValue = ($theValue != "") ? $theDefinedValue : $theNotDefinedValue;
      break;
  }
  return $theValue;
}
```

Figura 4.34 Bloque de validación para filtrado de inyección SQL.

La función `GetSQLValueString` valida el **tipo de dato** enviado desde el formulario y "limpia" de caracteres extraños estos datos mediante las funciones `get_magic_quotes_gpc` y `addslashes`. Mediante el switch que se esta ejecutando se validará el tipo de dato de entrada.

```php
$editFormAction = $_SERVER['PHP_SELF'];
```

$_SERVER['PHP_SELF'] devuelve la ruta absoluta al script php en ejecución.

```php
$_SERVER['QUERY_STRING']
```

$_SERVER['QUERY_STRING'] marca si existe, la cadena de la consulta de la petición de la página.

```php
isset($VARIABLE)
```

isset determina si una variable está definida y no es **NULL.**

```
$insertSQL = sprintf("INSERT INTO administrador (nombre, apellido, login,
                     pass, tipo, status, fechalta)
                     VALUES (%s, %s, %s, %s, %s, %s, %s)",
                     GetSQLValueString($_POST['nombre'], "text"),
                     GetSQLValueString($_POST['apellido'], "text"),
                     GetSQLValueString($_POST['login'], "text"),
                     GetSQLValueString($_POST['pass'], "text"),
                     GetSQLValueString($_POST['tipo'], "text"),
                     GetSQLValueString($_POST['status'], "int"),
                     GetSQLValueString($fecha, "date"));
```

Figura 4.35 Estructura para inserción de un registro de administrador

La variable `$insertSQL` ejecutará una sentencia de inserción en la base de datos a la tabla `administrador`; el método **`$_POST[]`** trae los valores que fueron capturados desde el formulario **`"form1"`** pasando por la función `GetSQLValueString`.

```
mysql_select_db($database_conexion, $conexion);
$Result1 = mysql_query($insertSQL, $conexion) or die(mysql_error());
```

Figura 4.36 Bloque de conexión a base de datos

Se crea una conexión seleccionado la base de datos con la función `mysql_select_db();` mediante las variables `$database_conexion` y `$conexion`, una vez que se creo la conexión se pasa la variable `$insertSQL` donde creamos la consulta para poder ser ejecuta. La variable `$conexion`, es obtenida del archivo `conexion2.php` que se obtiene mediante un llamado a su ruta de la siguiente manera:

```
require_once('../tesis/Connections/conexion2.php');
```

Figura 4.37 Llamado de conexión a base de datos

Esto nos ayuda a optimizar el código de 2 maneras, la primera es dando seguridad la configuración de de conexión a la base de datos, teniéndole en un diferente directorio, y segundo, no teniendo que escribir los parámetros de la configuración cada vez que se necesite una acceso, sea una consulta, actualización o inserción, a la base de datos.

El archivo `conexion2.php` de configuración de parámetros a la base de datos se muestra a continuación

```
<?php

$conexion = mysql_connect ("localhost",
                           "NOMBRE DE USUARIO",
                           "CONTRASEÑA") or die (mysql_error());
mysql_select_db("BASE DE DATOS",$conexion) or die (mysql_error());

?>
```

Figura 4.38 Estructura general para conexiones a base de datos.

donde los parámetros:

"localhost": es el nombre del servidor de base de datos o dirección IP que la contiene; puede incluir el puerto de ser necesario especificarlo.

"NOMBRE DE USUARIO": es el nombre del usuario que tiene acceso a la base de datos.

"CONTRASEÑA": la contraseña del usuario a la base de datos.
"BASE DE DATOS": el nombre asignado a la base de datos.

La segunda sección es el formulario que aparecerá en nuestro navegador web donde insertaremos los datos para agregar un nuevo usuario y mediante el método "POST" se enviaran a la primera sección donde serán validados e insertados.

```
<body>
<?php $fecha = date('Y-m-d'); ?>
<p class="style2">Agregar un nuevo administrador</p>
<form action="<?php echo $editFormAction; ?>"
      method="POST" name="form1" class="style1" id="form1">
<table width="90%" border="0">
    <tr>
          <td>Nombre:</td>
          <td><input name="nombre" type="text"
          id="nombre" size="40" maxlength="40" /></td>
          </tr>

                                .
                                .
                                .
    <input type="hidden" name="MM_insert" value="form1">
  </form>
```

Figura 4.39 Estructura del formulario de inserción

Modificar Administrador

Esta solo disponible para los administradores tipo 'A'.

```
if($tipo=='A'){
print'
    <a href="agr_admin.php">Agrega Nuevo Administrador</a><br>
    <br>
    <a href="lista_mod_admin.php">Modificar Administrador</a><br>
';
}
?>
<br>
```

Figura 4.40 Condición de opción para modificar Administrador

En el archivo lista_mod_admin.php encontraremos dos elementos importantes:

1) Mediante "javascript" y "mootools" creamos un filtrado de texto que nos ayuda a buscar mediante una caja de texto, esto lo hacemos desde el lado del cliente el cual interpreta esta información y en el navegador:

```
<head>
    <title>Administradores</title>

    <link rel="stylesheet" type="text/css" href="css/mtmultiselect__.css" />
    <script charset="UTF-8" type="text/javascript"
    src="js/mootools-1.2.2-core-nc_.js" ></script>
    <script charset="UTF-8" type="text/javascript"
    src="js/MTMultiSelect_____.js" ></script>
    <script charset="UTF-8" type="text/javascript"
    src="js/mtms__.js" ></script>

</head>
```

Figura 4.41 Bloque de cabeceras de javascript o hojas de estilo

y las líneas:

```
<select MULTIPLE name="select" class="multiselect">
   ...
  <option value=".$row['idAdmin']."><a href=''>".$row['nombre']." ".$row['apellido']."</a></option>
   ...
</select>
```

Figura 4.42 Fragmento de código para filtro de selección

2) Combinando con la consulta SQL que nos lista los administradores de la siguiente manera:

```
$sql = mysql_query("select idAdmin,nombre,apellido from administrador");
while ($row = mysql_fetch_array($sql))
{
  print "<option value=".$row['idAdmin']."><a href=''>".$row['nombre']."
        ".$row['apellido']."</a></option>";
}
mysql_free_result($sql);
```

Figura 4.43 Consulta a tabla de administradores

Ambos puntos en conjunto nos da como resultado la siguiente pantalla en el navegador:

Figura 4.44 Filtro de selección

que al insertar un texto dentro de la caja realizará el filtrado como se muestra en la siguiente pantalla:

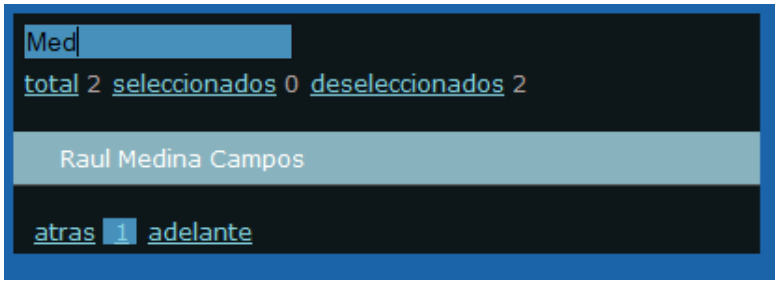


Figura 4.45 Ejemplo de uso del filtro

al seleccionar el administrador mandará la variable mediante el método Post al archivo mod_admin.php, el manejo de los formularios es prácticamente el mismo salvo 2 variantes, la seguridad de validación también es la misma.

La primera de las diferencias es la sentencia SQL que se está ejecutando de actualización:

```
$updateSQL = sprintf("UPDATE administrador SET nombre=%s, apellido=%s,
                     login=%s, pass=%s, tipo=%s, status=%s WHERE idAdmin=%s",
                      GetSQLValueString($_POST['vnombre'], "text"),
                      GetSQLValueString($_POST['vapellido'], "text"),
                      GetSQLValueString($_POST['vlogin'], "text"),
                      GetSQLValueString($_POST['vpass'], "text"),
                      GetSQLValueString($_POST['vtipo'], "text"),
                      GetSQLValueString($_POST['vstatus'], "int"),
                      GetSQLValueString($_POST['hiddenField'], "int")
                     );
```

Figura 4.46 Sentencia de actualización a administrador

La segunda es el llamado de los datos por una consulta para que aparezcan dentro del formulario ya que también hará uso de una consulta SQL para hacer el llamado:

```
$usr_modificar  =   $_POST["select"];

$query_Recordset1 = sprintf("SELECT * FROM administrador
                             WHERE idAdmin = $usr_modificar");
$Recordset1 = mysql_query($query_Recordset1, $conexion)
                          or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($Recordset1);
```

Figura 4.47 Llamado de datos de administrador seleccionado

De esta manera en las etiquetas **input** podemos introducir los valores en los campos **value** del la siguiente manera:

```
<td width="17%"><font face="arial" size="2">Nombre:</font></td>
<td width="83%"><input name="vnombre" type="text" id="vnombre"
value="<?php echo $row_Recordset1['nombre']; ?>" size="70" maxlength="60"/>
<input name="hiddenField" type="hidden"
value="<?php echo $row_Recordset1['idAdmin']; ?>" />
</td>
```

Figura 4.48 Ingreso de datos obtenidos en el formulario

Para que finalmente podamos obtener un formulario con los datos de la elección hecha:

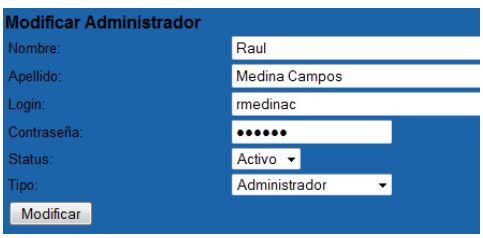


Figura 4.49 Apariencia del formulario con datos

IV.II.III.II Alumno

Agregar Nuevo Alumno

- ***Manejo de Seguridad***

El manejo de seguridad es el mismo que Agregar Nuevo Administrador usado mediante programación en PHP. Si no se cuenta con variables de usuario y contraseña mandará el mensaje de error:

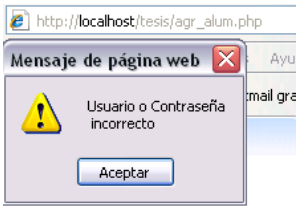


Figura 4.50 Mensaje de verificación de usuario y contraseña validos

La opción de Agregar Nuevo Alumno, como la de Modificar Alumno se encuentran para todo tipo de Administrador 'A' o 'B'.

```
if($tipo=='A'){
print'
    <a href="agr_admin.php">Agrega Nuevo Administrador</a><br>
    <br>
    <a href="lista_mod_admin.php">Modificar Administrador</a><br>
';
}
?>
<br>
<a href="agr_alum.php">Agrega Nuevo Alumno</a><br>
<br>
<a href="lista_mod_alum.php">Modificar Alumno</a><br>
```

Figura 4.51 Condición del menú administrador

El formulario de alta de Alta de Alumno maneja la misma estructura que el alta de un Administrador, variables de seguridad:

```
global $prime,$segun,$login,$pass;

if (!empty($prime) && !empty($segun))
{
if ($login==trim($prime) && $pass==trim($segun))
 {
```

Figura 4.52 Manejo del bloque de seguridad

Formulario con método post:

```
<form action="<?php echo $editFormAction; ?>"
method="POST" name="form1" class="style1" id="form1">
<table width="90%" border="0">
    <tr>
        <td>Número de Cuenta:</td>
        <td><input name="noCuenta" type="text"
            id="noCuenta" size="9" maxlength="9" /></td>
    </tr>

    ........

        <input name="Agregar" type="submit" id="Agregar" value="Agregar" />
```

Figura 4.53 Estructura de formulario de alta de alumno

Función de validaciones:

```
function GetSQLValueString($theValue, $theType, $theDefinedValue = "", $theNotDefinedValue = "")
{
  $theValue = (!get_magic_quotes_gpc()) ? addslashes($theValue) : $theValue;

  switch ($theType) {
    case "text":
      $theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
      break;
    case "long":
    case "int":
      $theValue = ($theValue != "") ? intval($theValue) : "NULL";
      break;
    case "double":
      $theValue = ($theValue != "") ? "'" . doubleval($theValue) . "'" : "NULL";
      break;
    case "date":
      $theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
      break;
    case "defined":
      $theValue = ($theValue != "") ? $theDefinedValue : $theNotDefinedValue;
      break;
  }
  return $theValue;
}
```

Figura 4.54 Función de validación de variables para evitar inyección SQL

Y recepción de variables **post**:

```
$subcuenta=substr($_POST['noCuenta'],5,1);
$subapell=substr($_POST['apellido'],2,1);
$subtime=substr(time(),2,6);
$uno=rand(1,500);
$dos=rand(1,500);
if($uno>255){$unos="*";}
elseif($uno<=255){$unos="/";}
if($dos>255){$doss="*";}
elseif($dos<=255){$doss="/";}
$pass=$unos.$subtime.$subapell.$subcuenta.$doss;

$insertSQL = sprintf("INSERT INTO alumno (noCuenta, noAdmin, nombre,
                     apellido, domicilio, edad, sexo, pass, fechaAlta)
                        VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s)",
                        GetSQLValueString($_POST['noCuenta'], "int"),
                        GetSQLValueString($noAdmin, "int"),
                        GetSQLValueString($_POST['nombre'], "text"),
                        GetSQLValueString($_POST['apellido'], "text"),
                        GetSQLValueString($_POST['domicilio'], "text"),
                        GetSQLValueString($_POST['edad'], "int"),
                        GetSQLValueString($_POST['sexo'], "text"),
                        GetSQLValueString($pass, "text"),
                        GetSQLValueString($fecha, "date"));

  $Result1 = mysql_query($insertSQL, $conexion) or die(mysql_error());
```

Figura 4.55 Sentencia de inserción de un registro de alumno

Finalmente la inserción de parámetros obtenidos a la base mediante la función mysql_query.

El algoritmo usado para la creación de la contraseña del alumno extrae valores de tiempo actual tomando del servidor, una letra del apellido del alumno y un número de su número de cuenta. Además se agregaron al inicio y final dos caracteres especiales (*I* y *****) obtenidos de manera aleatoria mediante la función **rand()**, los cuales también agregarán fortaleza a la contraseña.

```
$subcuenta=substr($_POST['noCuenta'],5,1);
$subapell=substr($_POST['apellido'],2,1);
$subtime=substr(time(),2,6);
$uno=rand(1,500);
$dos=rand(1,500);
if($uno>255){$unos="*";}
elseif($uno<=255){$unos="/";}
if($dos>255){$doss="*";}
elseif($dos<=255){$doss="/";}
$pass=$unos.$subtime.$subapell.$subcuenta.$doss;
```

Figura 4.56 Algoritmo para generación de contraseña aleatoria

Podría descifrarse las posiciones tanto del apellido como del número de cuenta pero el momento exacto en que se extrae el tiempo es casi imposible por lo que lo convierte en un método seguro en un alto porcentaje.

La modificación de los datos del alumno funciona exactamente igual que la del administrador se pasa a un listado de los alumnos los cuales pueden ser filtrados en la caja de texto mediante su número de cuenta, nombre o apellido:

Figura 4.57 Ejemplo de uso filtro de búsqueda de alumno por nombre

O por número de cuenta:

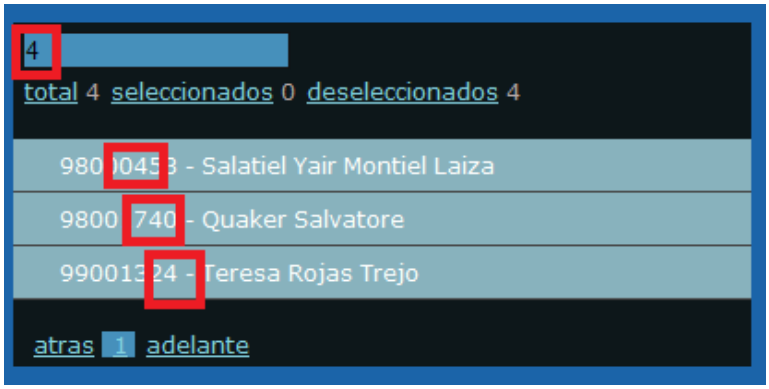


Figura 4.58 Ejemplo de uso filtro de búsqueda de alumno por número de cuenta

Y una vez elegido pasará al formulario de edición con las mismas características de la modificación de los administradores.

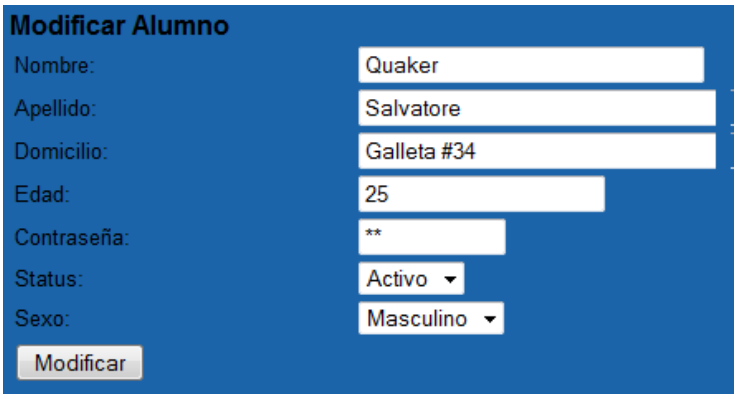


Figura 4.59 Formulario con datos de alumno a modificar

Consultar Datos del Perfil de Alumno

La última opción que encontramos en el menú de administradores tanto para tipo “A” como “B” es el de consultar los datos del perfil del alumno obtenidos mediante el procesamiento de la información del sistema.

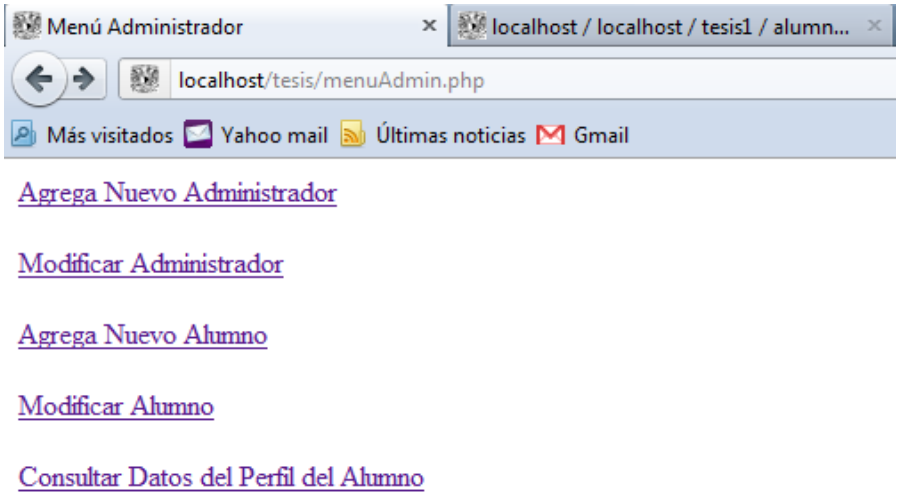


Figura 4.60 Menú con opción de consulta de datos de alumno

Código html:

```
<a href="perfil.php">Consultar Datos del Perfil del Alumno</a><br>
<br>
```

Esta opción nos llevará a listado que enviará mediante un formulario en una variable **post** y de los alumnos elegiremos al que se desea analizar el perfil.

Figura 4.61 Filtro de listado de alumnos

Código:

```
<div id="header">

    <h2>Consulta de Perfil de Alumnos</h2>
    </div>
    <div id="container"><div id="content">

<form ACTION='enviaNoC.php' METhOD=POST>
    <select MULTIPLE name="NoCuenta" class="multiselect">
<?php
require_once('../tesis/Connections/conexion2.php');

   $sql = mysql_query("select noCuenta,nombre,apellido from alumno");
   while ($row = mysql_fetch_array($sql))
   {
     print "<option value=".$row[0]."><a href=''>
            ".$row[0]." - ".$row[1]."
            ".$row[2]."</a></option>";
   }
   mysql_free_result($sql);

?>
    </select>
        <br><INPUT TYPE="SUBMIT" VALUE="Mostrar" id="enviar">

</form>
```

Figura 4.62 Consulta para generación de listado de alumnos

Al elegir al alumno nos desplegará la siguiente pantalla:

Datos:

Numero de Cuenta:98000453
Nombre:Salatiel Yair Montiel Laiza
Sexo:M
Se usa tabla de hombres para percentiles

Form A 27	Percentil A 95
Form B 31	Percentil B 92
Form C 26	Percentil C 53
Form D 18	Percentil D 26
Form E 32	Percentil E 74
Form F 28	Percentil F 94
Form G 26	Percentil G 69
Form H 26	Percentil H 95
Form I 22	Percentil I 45
Form EscAct 5	

Figura 4.63 Tabla general de información del Alumno

Obtenidos de la siguiente manera:

A través de la variable **post** que envía el número de cuenta se realiza una consulta para mostrar los datos elementales como son número de cuenta, nombre completo y sexo:

```
<?php

$NoCuenta = $_POST["NoCuenta"];
require_once('Connections/conexion2.php');

    $sqlSx = mysql_query("SELECT nombre,apellido,sexo FROM `alumno`
                          WHERE `noCuenta`= $NoCuenta ")
                          or die ($sqlSx .mysql_error()."Error en Sexo" ) ;
    while ($rowSx = mysql_fetch_array($sqlSx))
     {
     $nombre    = $rowSx['nombre'];
     $apellido  = $rowSx['apellido'];
     $sexoAlum  = $rowSx['sexo'];
     }
     mysql_free_result($sqlSx);
```

Figura 4.64 Consulta para generación de datos del Alumno

Para el cálculo de los percentiles primero necesitamos la suma de ciertas preguntas para cada letra como se muestra continuación:

```
$sql = mysql_query("SELECT (`p3`+ `p7`+ `p15`+ `p17`+ `p25`+ `p27`+
`p32`+ `p35`+ `p38`+ `p40`+ `p44`+ `p47`+ `p56`+ `p62`+ `p65`+
`p68`+ `p70`+ `p75`+ `p80`)
AS TOTAL FROM `fa` WHERE `idpa`= $NoCuenta ")
or die ($sql .mysql_error()."Errorsito" ) ;
while ($row = mysql_fetch_array($sql))
 {
 $TotA  = $row['TOTAL'];
 }
 mysql_free_result($sql);
```

...

```
$sql9 = mysql_query("SELECT (`p11`+ `p20`+ `p22`+ `p30`+ `p58`+
`p61`+ `p66`+ `p77`+ `p79`+ `p83`+ `p95`+ `p115`+ `p120`+ `p124`+
`p126`+ `p133`+ `p153`+ `p166`+ `p168`+ `p177`)
AS TOTAL FROM `fi` WHERE `idpi`= $NoCuenta ")
or die ($sql9 .mysql_error()."Errorsito" ) ;
while ($row9 = mysql_fetch_array($sql9))
 {
 $TotI  = $row9['TOTAL'];
 }
 mysql_free_result($sql9);
```

...

```
$sqlEA = mysql_query("SELECT (`p1`+ `p11`+ `p38`+ `p44`+ `p54`+
`p58`+ `p61`+ `p63`+ `p99`+ `p111`+ `p114`+ `p116`+ `p129`+
`p148`+ `p152`+ `p170`+ `p171`+ `p172`)
AS TOTAL FROM `fescact` WHERE `idpescAct`= $NoCuenta ")
or die ($sqlEA .mysql_error()."Errorsito" ) ;
while ($rowEA = mysql_fetch_array($sqlEA))
 {
 $TotEA = $rowEA['TOTAL'];
 }
 mysql_free_result($sqlEA);
```

Figura 4.65 Consultas para generación de percentiles

Y así sucesivamente de la “A” a la “I” y para la Escala de Actitud:

Con las sumas de estos datos y mediante el sexo de alumno elegiremos mediante la siguiente condición si se trabajará con la tabla de percentil hombre o percentil mujer como se muestra a continuación:

```
if($sexoAlum=='M')
{$tablsx='perhombre';
 print "<br>Se usa tabla de hombres para percentiles";
}
elseif($sexoAlum=='F')
{$tablsx='permujer';
 print "<br>Se usa tabla de mujeres para percentiles";
}
```

Figura 4.66 Selección de tabla de percentiles

Con la tabla podremos hacer la consulta final para obtener el cálculo de los percentiles de la siguiente manera:

```
$sqlPCa = mysql_query("SELECT `A` FROM ".$tablsx." WHERE `PC` = $TotA ")
or die ($sqlPCa .mysql_error()."Errorsito" ) ;
while ($rowPCa = mysql_fetch_array($sqlPCa))
 {
 $ValPCa    = $rowPCa['A'];
 }
 mysql_free_result($sqlPCa);
```

Figura 4.67 Consulta para cálculo de percentiles “A”

Con en el paso anterior desde la “TotA” hasta “TotI”.

```
$sqlPCi = mysql_query("SELECT `I` FROM ".$tablsx." WHERE `PC` = $TotI ")
or die ($sqlPCi .mysql_error()."Errorsito" ) ;
while ($rowPCi = mysql_fetch_array($sqlPCi))
 {
 $ValPCi    = $rowPCi['I'];
 }
 mysql_free_result($sqlPCi);
```

Figura 4.68 Consulta para cálculo de percentiles “I”

Finalmente usamos estos datos para mostrarlos en una tabla:

```
<table border="1">
  <tr>
    <td><?php
         print "Form A  ".$TotA;
         print"<br>";
         print "Form B  ".$TotB;
         print"<br>";
         print "Form C  ".$TotC;
         print"<br>";
         print "Form D  ".$TotD;
         print"<br>";
         print "Form E  ".$TotE;
         print"<br>";
         print "Form F  ".$TotF;
         print"<br>";
         print "Form G  ".$TotG;
         print"<br>";
         print "Form H  ".$TotH;
         print"<br>";
         print "Form I  ".$TotI;
         print"<br>";
         print "Form EscAct  ".$TotEA;

    ?></td>
    <td valign="top"><?php
         print "Percentil A  ".$ValPCa;
         print"<br>";
```

Figura 4.69 Generación del listado de información

Y en una gráfica para ubicar los percentiles mediante el archivo **graficaInt.php** al cual le pasaremos parámetro de la siguiente manera:

```
<?php
print '<br><img src="graficaInt.php?va='.$ValPCa.'&vb='.$ValPCb.'
&vc='.$ValPCc.'&vd='.$ValPCd.'&ve='.$ValPCe.'&vf='.$ValPCf.'&vg='
.$ValPCg.'&vh='.$ValPCh.'&vi='.$ValPCi.'">';
?>
```

Figura 4.70 Paso de variables para generación de gráfica

El archivo **graficaInt.php** usa las librerías especiales para generación de gráficos:

```
<?php
include ("../tesis/graficas/jpgraph.php");
include ("../tesis/graficas/jpgraph_line.php");
include ("../tesis/graficas/jpgraph_error.php");
include ("../tesis/graficas/jpgraph_bar.php");
```

Figura 4.71 Llamado de librerías para graficar

Y mediante arreglos fijamos los límites:

```
$l1datay  = array($va,$vb,$vc,$vd,$ve,$vf,$vg,$vh,$vi);
$datay    = array(100,100,100,100,100,100,100,100,100);
$l2datay  = array(85,80,100,100,100,80,98,74,100);
$l3datay  = array(50,45,100,100,100,63,95,53,90);
$l4datay  = array(25,22,100,100,100,35,90,27,85);
$l33datay = array(0,0,65,75,75,5,53,3,63);
$l22datay = array(0,0,35,50,50,3,35,1,37);
$l00datay = array(0,0,15,20,20,0,15,0,20);
$datax=array("Reposado-Calmado","Animoso","Tranquilo","Inhibido",
"Indiferente","Objetivo","Sumiso","Tolerante","Impulsivo");
```

Figura 4.72 Generación de arreglos y paso de variables

Instanciamos nuevos objetos declarados en las librerías, asignamos a los colores de verde como excelente, azul como aceptable, naranja como cambio deseable, magenta como cambio urgente y finalmente mediante la función **Add()** graficamos:

```
$l4plot = new LinePlot($l4datay);
$l4plot->SetFillColor("green");
$l4plot->SetLegend("Excelente");


$l33plot = new LinePlot($l33datay);
$l33plot->SetFillColor("blue");
$l33plot->SetLegend("Aceptable");


$l22plot = new LinePlot($l22datay);
$l22plot->SetFillColor("orange");
$l22plot->SetLegend("Es deseable un cambio");


$l00plot = new LinePlot($l00datay);
$l00plot->SetFillColor("magenta");
$l00plot->SetLegend("El cambio es urgente");


$graph->Add($plot);
$graph->Add($l2plot);
$graph->Add($l3plot);
$graph->Add($l4plot);
$graph->Add($l33plot);
$graph->Add($l22plot);
$graph->Add($l00plot);
$graph->Add($l1plot);
```

Figura 4.73 Parámetros y características de la gráfica

Para finalmente obtener la siguiente grafica que será la herramienta más importante para la persona que analice el perfil:

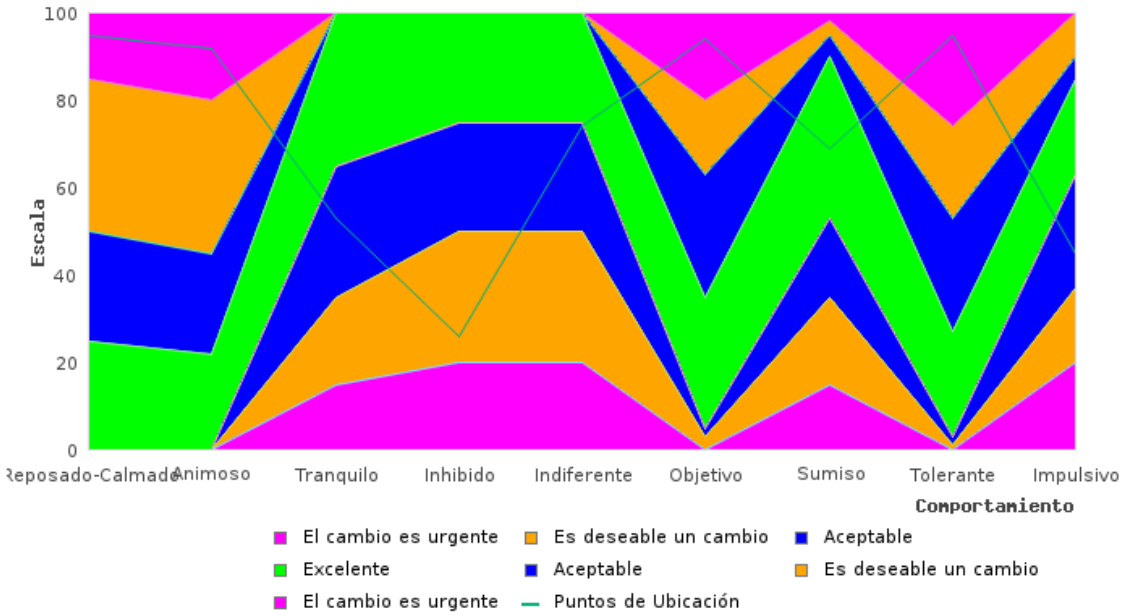


Figura 4.74 Generación de gráfica con parámetros enviados

IV.II.III.I Alumnos

Los alumnos también tendrán su acceso mediante una ventana de acceso en el cual tendrán que ingresar su número de cuenta y la contraseña asignada por el sistema.

Figura 4.75 Ventana de ingreso para Alumnos

La seguridad está contenida en el archivo **consultaAlumno.php** comprobando los datos, primero se realiza la consulta:

```
$sql = mysql_query("select * from alumno where noCuenta='$prime'
and pass='$segun' ") or die ($Sql .mysql_error()."Errorsito MySql" ) ;
```

Figura 4.76 Consulta de seguridad en ingreso para Alumnos

Y con los datos obtenido se hace la comparación, si existe coincidencia en ambos valores concede el acceso, sino manda un mensaje de error.

```
if (!empty($prime) && !empty($segun))
{
  if ($prime==$vlogin && $segun==$vpass)
  {
   setcookie ("prime",$prime);
   setcookie ("segun",$segun);
   setcookie ("login",$vlogin);
   setcookie ("pass",$vpass);
   setcookie ("nomusutit",trim($vnombre));
   setcookie ("apellsutit",trim($vap));

   $login=$vlogin;
   $pass=$vpass;

   include('iniciaAlumno.php');

  }
  else
  {
   unset($prime);
   unset($segun);
   unset($login);
   unset($pass);
   unset($tipacc);
   unset($vcveusu);
   print '<script language="Javascript">';
   print 'alert("Usuario o Contraseña \n incorrecto");
   print '</SCRIPT>';
  }
```

Figura 4.77 Paso de parámetros de seguridad

Con los datos incorrectos genera una alerta emergente mediante javascript como la siguiente:

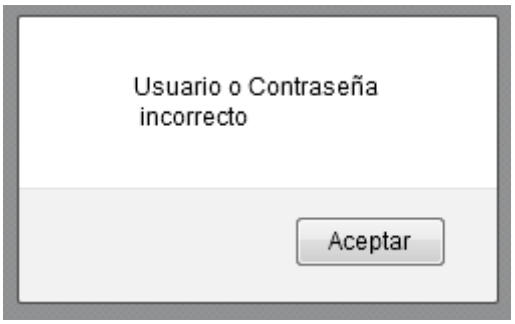
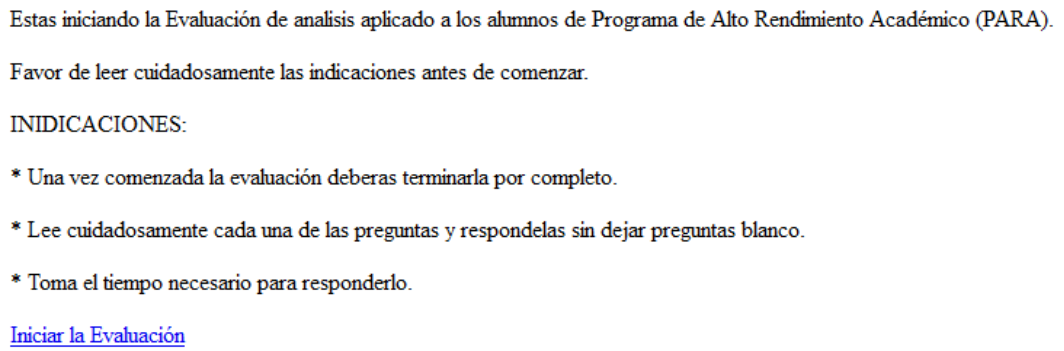


Figura 4.78 Mensaje generado por usuario o contraseña incorrectos.

Si los datos son correctos nos llevará a la siguiente ventana donde podrá iniciarse el respectiva Evaluación:


Estas iniciando la Evaluación de analisis aplicado a los alumnos de Programa de Alto Rendimiento Académico (PARA).

Favor de leer cuidadosamente las indicaciones antes de comenzar.

INIDICACIONES:

* Una vez comenzada la evaluación deberas terminarla por completo.

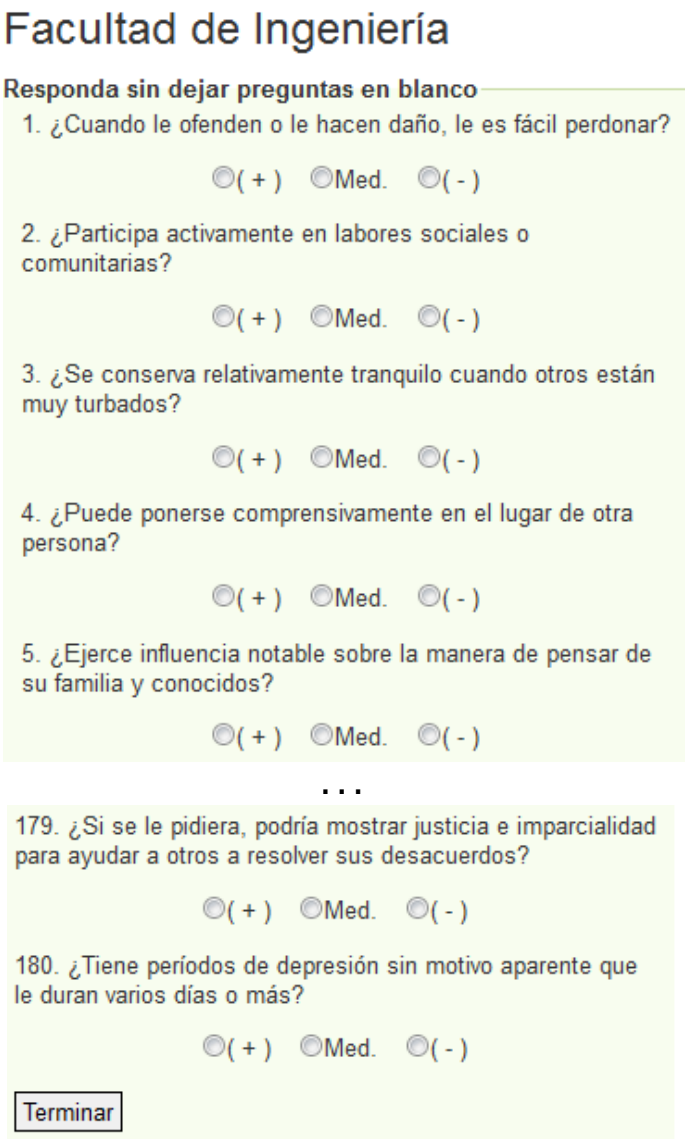
* Lee cuidadosamente cada una de las preguntas y respondelas sin dejar preguntas blanco.

* Toma el tiempo necesario para responderlo.

Iniciar la Evaluación

Figura 4.79 Indicaciones previas para el alumno

Al dar clic en Iniciar Evaluación nos llevar a una serie de preguntas que están contenidas en un formulario, son 180 en total y cada una de estas preguntas tiene asignado un valor específico que será guardado en la base de datos.


Facultad de Ingeniería

Responda sin dejar preguntas en blanco

1. ¿Cuando le ofenden o le hacen daño, le es fácil perdonar?

(+) Med. (-)

2. ¿Participa activamente en labores sociales o comunitarias?

(+) Med. (-)

3. ¿Se conserva relativamente tranquilo cuando otros están muy turbados?

(+) Med. (-)

4. ¿Puede ponerse comprensivamente en el lugar de otra persona?

(+) Med. (-)

5. ¿Ejerce influencia notable sobre la manera de pensar de su familia y conocidos?

(+) Med. (-)

. . .

179. ¿Si se le pidiera, podría mostrar justicia e imparcialidad para ayudar a otros a resolver sus desacuerdos?

(+) Med. (-)

180. ¿Tiene períodos de depresión sin motivo aparente que le duran varios días o más?

(+) Med. (-)

Terminar

Figura 4.80 Listado de preguntas para el Alumno

Están contenidas en el archivo **PrgComp.php** y la asignación de los valores está dada por las hojas anexas. La validación de los campos para asegurarse de que todas las preguntas sean respondidas se realizó mediante la función “valídate” de JQuery que ayudará a no permitir el envío de información si no se ha contestado la pregunta.

```
<p class="style2">Facultad de Ingeniería</p>
<form action="<?php echo $editFormAction; ?>" method="POST" name="form1" class="cmxform" id="form1">
    <fieldset>
        <legend>Responda sin dejar preguntas en blanco</legend>
        <fieldset>
            <legend>1. ¿Cuando le ofenden o le hacen daño, le es fácil perdonar?</legend>
            <label for="p1">
                <input name="p1" type="radio" value="2" validate="required:true" />( + )   
                <input type="radio" name="p1" value="1" />Med.   
                <input type="radio" name="p1" value="0" />( - )
            </label>
            <label for="p1" class="error">Por favor responda la pregunta 1</label>
        </fieldset>
        <fieldset>
            <legend>2. ¿Participa activamente en labores sociales o comunitarias?</legend>
            <label for="p2">
                <input name="p2" type="radio" value="2" validate="required:true" />( + )   
                <input type="radio" name="p2" value="1" />Med.   
                <input type="radio" name="p2" value="0" />( - )
            </label>
            <label for="p2" class="error">Por favor responda la pregunta 2</label>
        </fieldset>
```

Figura 4.81 Fragmento de código de preguntas

El script de validación lo encontramos en la cabecera de la siguiente manera:

```
<title>Cuestionario Tipo A</title>

    <link rel="stylesheet" type="text/css" media="screen" href="css/screen.css" />
    <script src="lib/jquery.js" type="text/javascript"></script>
    <script src="lib/jquery.validate.js" type="text/javascript"></script>
    <script src="lib/jquery.metadata.js" type="text/javascript"></script>

    <script type="text/javascript">

    $.metadata.setType("attr", "validate");

    $(document).ready(function() {
        $("#form1").validate();
        $("#selecttest").validate();
    });
    </script>

    <style type="text/css">
    .block { display: block; }
    form.cmxform label.error { display: none; }
    </style>

<style type="text/css">
<!--
.style1 {font-family: Arial}
.style2 {font-family: Arial; font-size: 24px; }
-->
</style>
</head>
```

Figura 4.82 Script de función de validación

Para que finalmente los datos al ser validados todos los campos sean introducidos en la base de datos así:

```
$fecha = date("Y-m-d H:i:s");
$idAlumno=$login;
if ((isset($_POST["MM_insert"])) && ($_POST["MM_insert"] == "form1")) {
  $insertSQL = sprintf("INSERT INTO fa (idpa, p3, p7, p15, p17, p25, p27,
  p32, p35, p38, p40, p44, p47, p56, p62, p65, p68, p70, p75, p80, fecha)
VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s,
%s, %s, %s, %s)",
                       GetSQLValueString($idAlumno, "int"),
                       GetSQLValueString($_POST['p3'], "text"),
                       GetSQLValueString($_POST['p7'], "text"),
                       GetSQLValueString($_POST['p15'], "text"),
                       GetSQLValueString($_POST['p17'], "text"),
                       GetSQLValueString($_POST['p25'], "text"),
                       GetSQLValueString($_POST['p27'], "text"),
                       GetSQLValueString($_POST['p32'], "text"),
                       GetSQLValueString($_POST['p35'], "text"),
                       GetSQLValueString($_POST['p38'], "text"),
                       GetSQLValueString($_POST['p40'], "text"),
                       GetSQLValueString($_POST['p44'], "text"),
                       GetSQLValueString($_POST['p47'], "text"),
                       GetSQLValueString($_POST['p56'], "text"),
                       GetSQLValueString($_POST['p62'], "text"),
                       GetSQLValueString($_POST['p65'], "text"),
                       GetSQLValueString($_POST['p68'], "text"),
                       GetSQLValueString($_POST['p70'], "text"),
                       GetSQLValueString($_POST['p75'], "text"),
                       GetSQLValueString($_POST['p75'], "text"),
                       GetSQLValueString($fecha, "text"));
```

Figura 4.83 Bloque de inserción de datos

IV.II.IV Incremento de seguridad a través de ocultar mensajes de tipo “error”, “notice”, “warning” o “deprecated”

Finalmente una vez que hemos probado la correcta función del código del sistema ocultaremos los mensajes de “**error**”, “**notice**”, “**deprecated**” y “**warning**” ya que alguno de ellos podría dar algún indicio a un atacante para accesar a los datos de sistema.

Los tanto los mensajes de tipo “**deprecated**”, como “**error**” deben ser nulos pero existe la posibilidad que en alguna actualización del servidor estos se produzcan, por otro lado los mensajes de “**warning**” o “**notice**” puede aparecer al ejecutar una sentencia SQL que no nos regrese información.

Para un ambiente de desarrollo son indispensables este tipo de mensajes pues nos guiaran en los errores que se puedan producir dentro del desarrollo pero una vez montado el sistema que estará en operación deben ser ocultados.

La manera es la siguiente. En el archivo **php.ini** tendremos que editar la siguiente línea:

error_reporting = E_ALL | E_STRICT

por esta:

error_reporting = ~E_ALL

lo cual indicara que no muestre ninguno de los mensajes o simplemente dejar comentada esa línea con un signo de punto y coma (**;**) al inicio, como se muestra:

;error_reporting = E_ALL | E_STRICT

Con esto obtendremos el siguiente resultado:

Figura 4.84 Pantalla con alertas de servidor encendidas

Ocultándolo así:

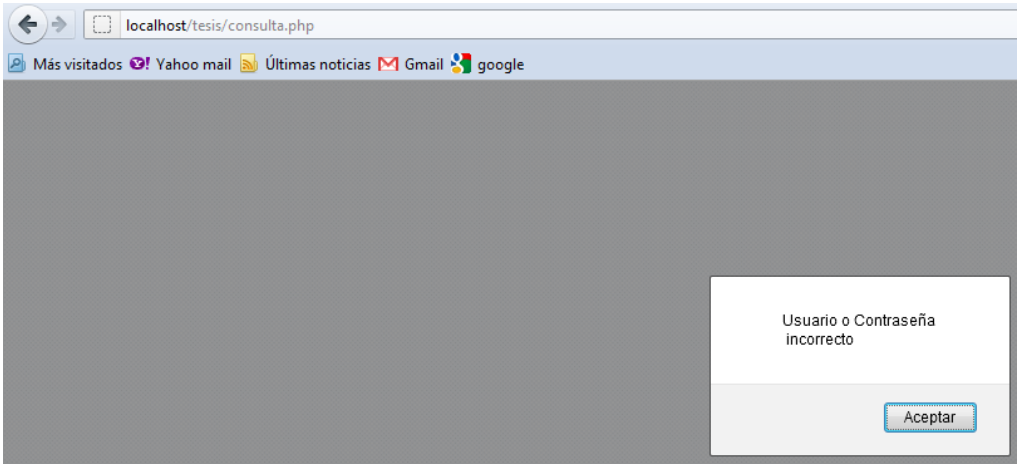


Figura 4.85 Pantalla con alertas de servidor ocultas

IV.III Pruebas e Implementación del Sistema

La implementación del sistema está preparada para ser utilizada por la persona responsable y con la capacidad para realizar aplicar el cuestionario a los alumnos y presentar los resultados pertinentes que en este caso sería con sujeto ideal un psicólogo o estudiante de la facultad de psicología de los últimos semestre el cual ya está familiarizado con este método de “Análisis de Temperamento de Taylor y Johnson” ya que nosotros nos limitamos exclusivamente a la realización técnica del

sistema el punto de la ingeniería, usando los parámetros e indicación aportadas por un profesional del área de psicología.

IV.III.I Prueba de Súper Administrador tipo “A”

a) Ingreso al sistema.

Se ingresa mediante una ventana donde se comprueba el nombre de usuario y su contraseña:

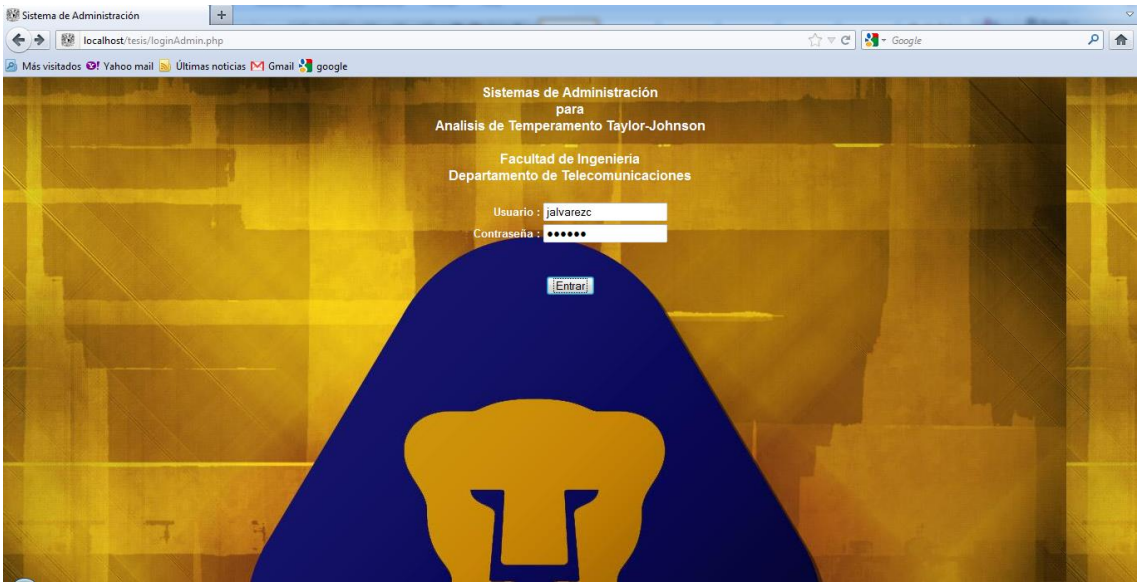


Figura 4.86 Pantalla de ingreso para súper administrador

Si la contraseña o el usuario son erróneos desplegará una ventaja con el siguiente mensaje:

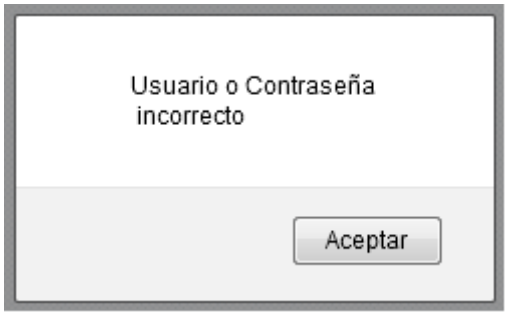


Figura 4.87 Mensaje de error por usuario o contraseña incorrecta

b) Si el nombre de usuario y contraseña son válidos accesamos a la siguiente página:

Figura 4.88 Menú de inicio de Administrador

La opción Agregar Nuevo Administrador nos mostrará:

Figura 4.89 Formulario de opción Agregar nuevo administrador

La opción Modificar Administrador no dejar elegir entre los administradores ya existentes:

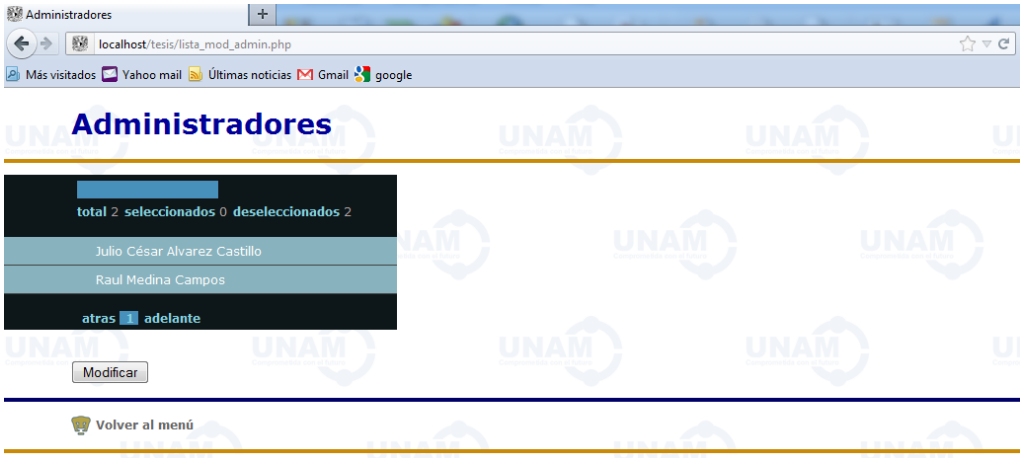


Figura 4.90 Filtro para selección de un administrador

Y al elegir uno de los administradores en el catálogo nos llevará al siguiente formulario:

Figura 4.91 Formulario de opción Modificar administrador

La opción Agregar Nuevo Alumno nos permitirá llenar el formulario:

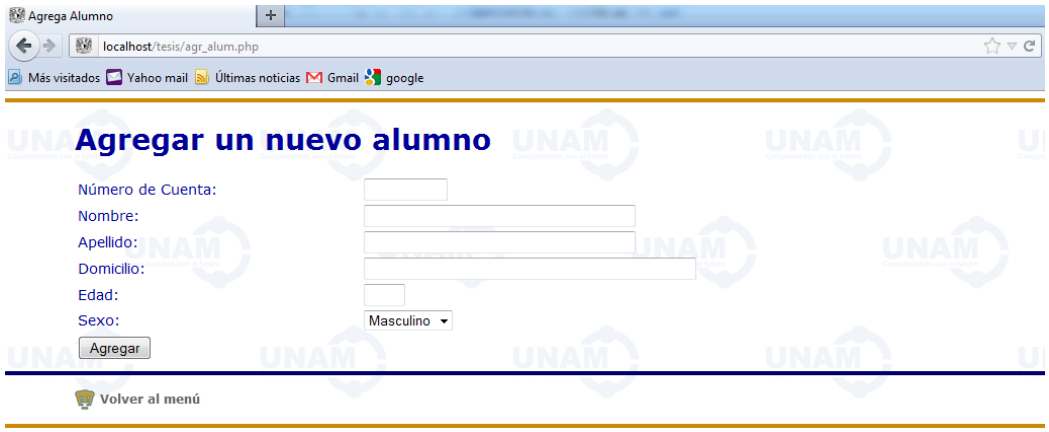


Figura 4.92 Formulario de opción Agregar nuevo alumno

La opción Modificar Nuevo Alumno no llevará al listado:

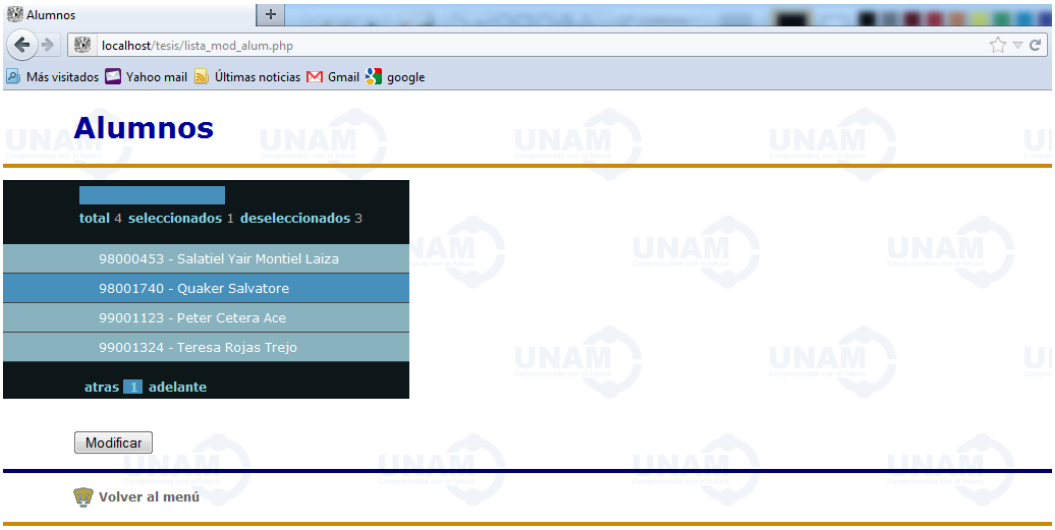


Figura 4.93 Filtro para selección de un alumno

Al elegir uno de los alumnos nos llevará al siguiente formulario:

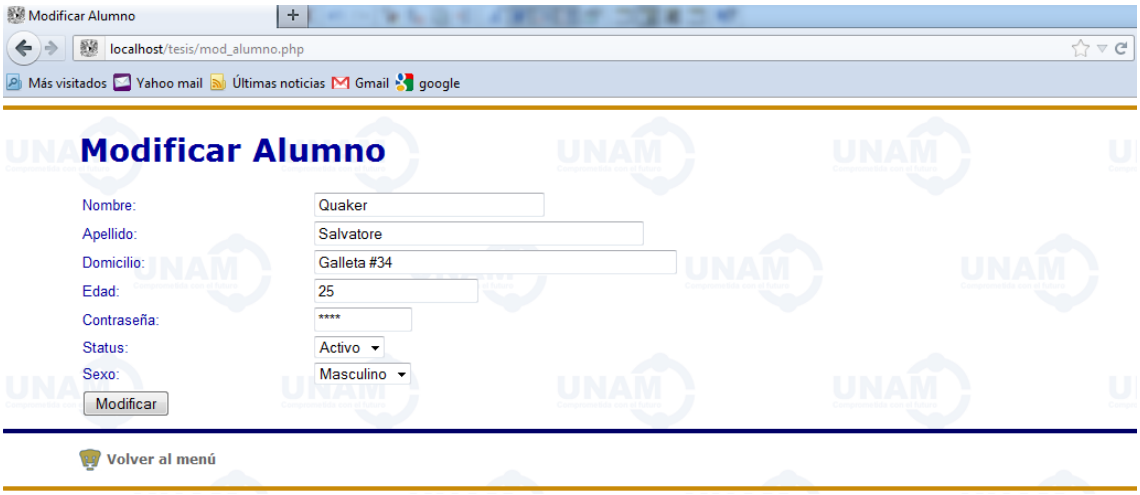


Figura 4.94 Formulario de opción Modificar alumno

Por último la opción Consultar Datos del Perfil del Alumno nos mostrará el siguiente listado:

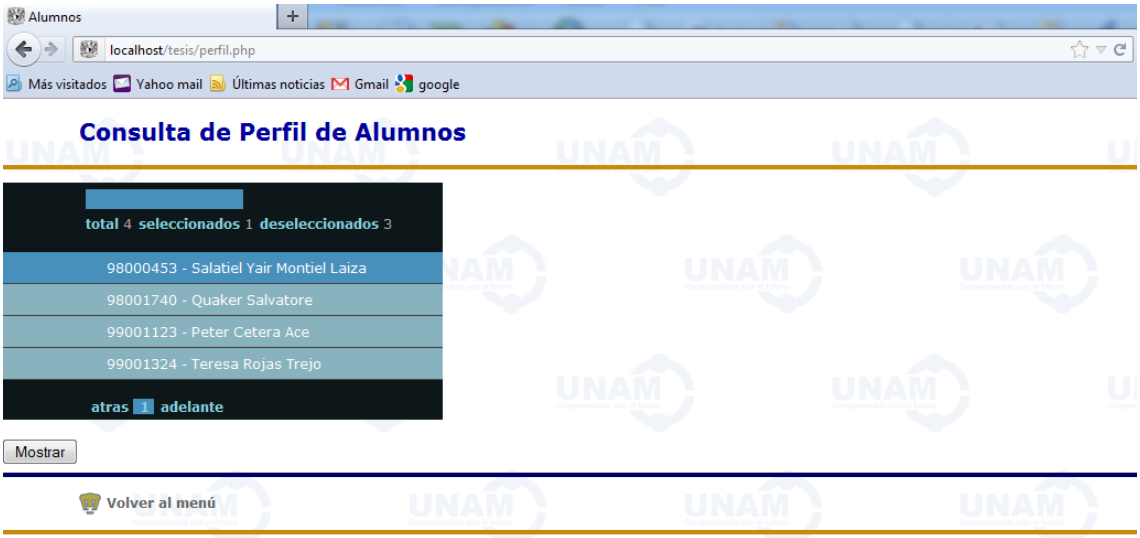


Figura 4.95 Filtro para selección de consulta de perfil de alumno

Y al elegirlo nos llevará la ventana donde desplegará todo el cálculo de los datos:

Figura 4.96 Ficha de Datos y perfil de alumno con posicionamiento en gráfica

IV.III.II Prueba de Administrador tipo "B"

La ventana de acceso es la misma de Super Administrador:

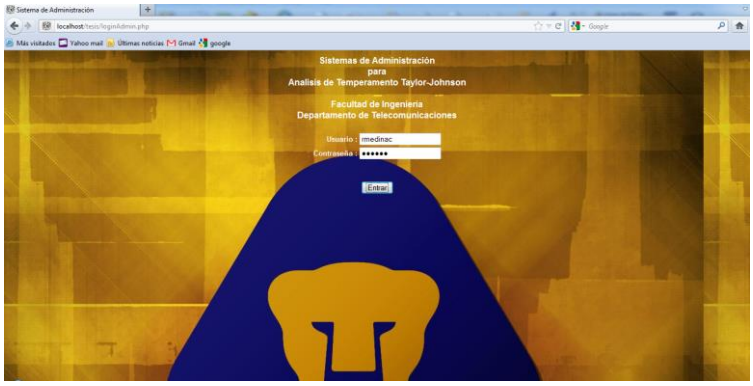


Figura 4.97 Pantalla de ingreso para Administrador

Pero las características de este solo permitirán mostrar las opciones de **Agregar Nuevo Alumno**, **Modificar Alumno** y **Consultar Datos del Alumno** como se muestra a continuación:

Figura 4.98 Menú de inicio de Administrador tipo B

IV.III.III Prueba de Alumno

Los alumnos ingresarán mediante el índice de la carpeta en la siguiente ventana con su número de cuenta y la contraseña proporcionada por el administrador:

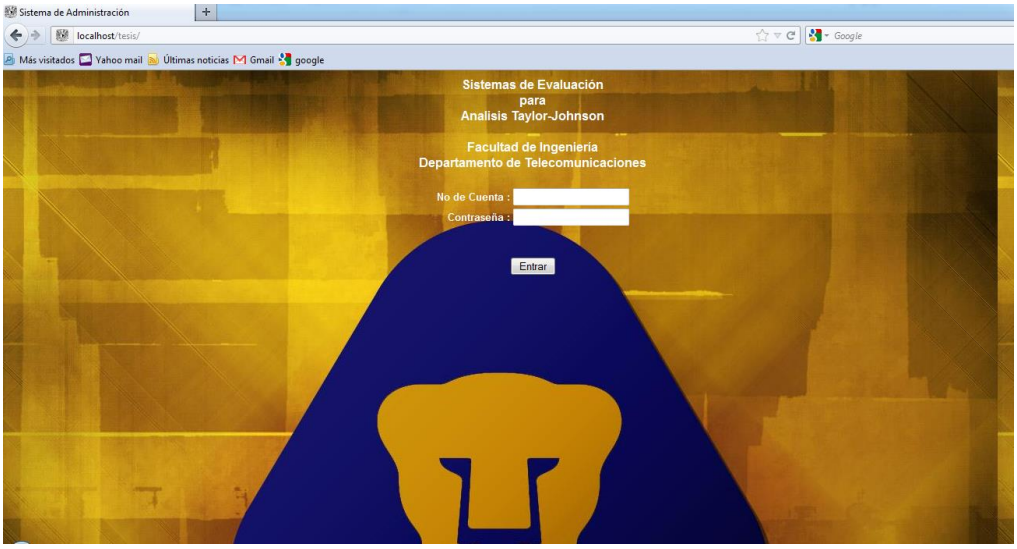


Figura 4.99 Pantalla de ingreso para Alumno

Si los datos son incorrectos nos mostrará el siguiente mensaje:

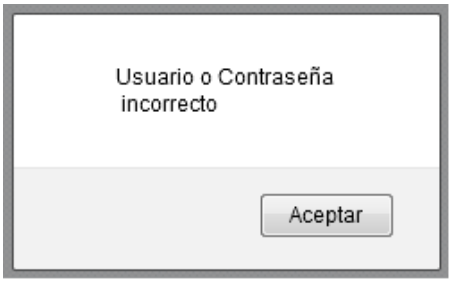


Figura 4.100 Mensaje de error por usuario o contraseña incorrecta

Si son correctos ingresaremos a la siguiente ventana con las instrucciones a seguir para contestar el formulario de la evaluación:

Figura 4.101 Instrucciones para realizar evaluación

Al **Iniciar la Evaluación** nos llevará a la siguiente página:

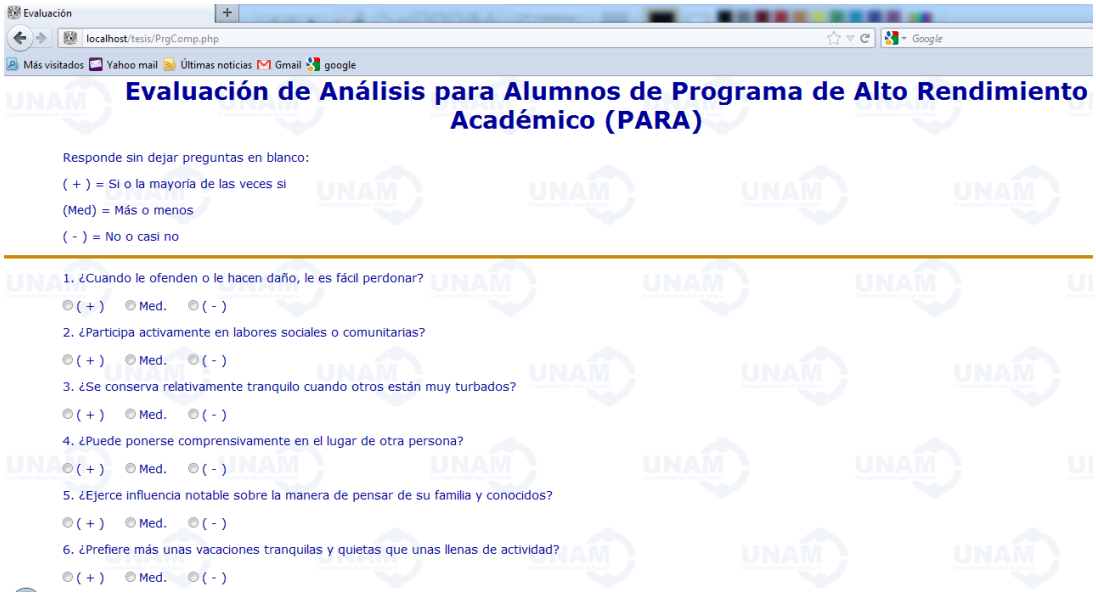


Figura 4.102 Preguntas iniciales de la evaluación

Si alguno de las preguntas no es contestada desplegará un mensaje en color rojo pidiendo que sea contestada como mostramos a continuación:

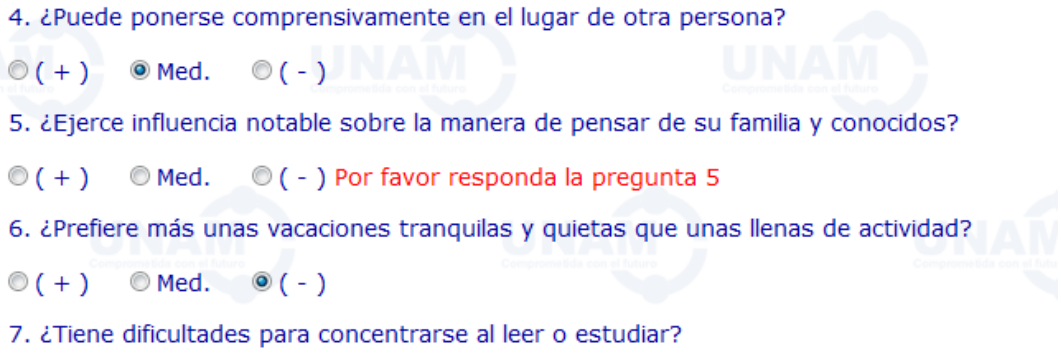


Figura 4.103 Mensaje de validación por pregunta sin contestar

Si todas las preguntas son contestadas nos llevará a una ventana de salida:

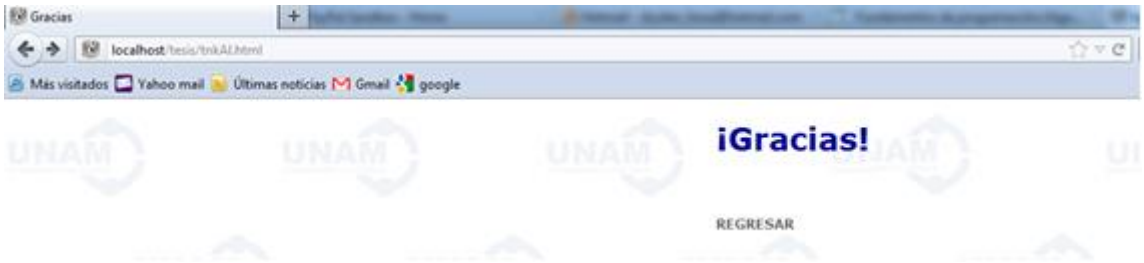


Figura 4.104 Mensaje de finalización de evaluación

Capítulo V

Conclusiones y trabajo a futuro

En base al trabajo realizado en esta tesis podemos puntualizar las siguientes conclusiones.

- La facilidad de resolución por parte del sujeto al que se aplica la evaluación.

Pudimos demostrar cómo se incremento la facilidad de resolución para el sujeto que realiza la prueba, pues no tiene que ir revisando hojas de preguntas y hojas de respuestas además de ir llenando los reactivos, reduciendo la posibilidad de error al contestar en un alto porcentaje mencionado en un punto posterior.

- La facilidad acceso al sujeto al que se aplica la evaluación.

Al tener esta prueba programada en una plataforma web, con un servidor en línea, nos abre la posibilidad de aplicarla en cualquier lugar y momento que se deseé mientras el sujeto cuente con una computadora y conexión a internet. Esto amplia el tiempo de disponibilidad para su aplicación.

- La facilidad de acceso al intérprete o experto.

También este punto es muy importante de mencionar ya que el experto en interpretación de los datos, con la forma tradicional que tenían de aplicar esta prueba, después tomar las respuestas, compararlas en papel contra una hoja de reactivos que daban un valor específico de acuerdo a la respuesta y una vez teniendo todos los valores de las respuestas, hacer una suma y ubicarlo dentro de una tabla de percentiles de acuerdo al sexo de sujeto, una vez ubicado el valor del percentil se buscaba dentro de una gráfica para poder tener una valor final y poder dar una interpretación. Con la programación sistematizada de este proceso el experto únicamente tiene que buscar el nombre del sujeto y el cálculo completo se realizará de manera automática mostrando el resultado final dentro de una gráfica hecha a medida, listo para ser interpretado. Un objetivo particular cumplido fue hacer fácil al interprete, de una manera que con los clics mínimos pudiera tener toda la información necesaria para él.

- Optimización de tiempo.

Como se mencionó en los dos primeros puntos se logro optimizar los tiempos, calculando una reduciéndolos en un 80% de original, usando todo en papel. De diez hrs. totales de trabajo desde resolución de sujeto, extracción de información y por último interpretación, se logró reducir a dos hrs. en promedio. Uno de los factores

resaltantes fue, no tener que tener físicamente presente al sujeto para aplicarle la prueba.

- Disponibilidad de los datos.

La disponibilidad de los datos puede tenerse 24 / 7 / 365, es decir prácticamente siempre mientras el servidor se encuentre en funcionamiento, y el interprete/experto con un punto de acceso a internet, los datos pueden exportarse, y respaldarse, sin tener que guardar grandes cantidades de papel, toda la base de datos de varios años puede concentrarse en un CD.

De ser necesario puede exportarse esta información otro manejador de base de datos más robusto de acuerdo a las necesidades del sistema, ya que el mismo fue diseñado para ser escalado de ser necesario.

Esta misma información al encontrarse en una base de datos puede ser explotada de diferentes formas, por ejemplo para realizar estadísticas cronológicas.

- Reducción de error humano en un 90 %.

Se reducen tanto en el sujeto como en el experto la posibilidad de error tanto, el sujeto al contestar, como el experto al tomar los datos, pasarlos por tablas y finalmente llevarlo a su grafica de interpretación. El sistema después de ser programado, se probó con varios reactivos en papel, teniendo resultados positivos.

El otro 10% de margen solo puede atribuirse a distractores externos.

- Reducción de tiempo de trabajo cuantitativo y mejor expectativa del cualitativo

Siguiendo un esquema de incremento de eficiencia este es uno de los objetivos que debe buscarse cumplir, mientras el trabajo del experto o interprete se menos desgastante, abre la oportunidad para que esta persona se enfoque en dar un trabajo de mayor calidad en cuanto al diagnóstico e interpretación de los datos, logrando tener un mayor impacto positivo sobre algún problema reflejado en el sujeto analizado y este a su vez logrará un mejor desempeño al ser detectado y trabajo el problema encontrado.

- Trabajo a futuro

El sistema fue programado de manera que tuviera posibilidad de ser modificado, escalado o migrado de acuerdo a las necesidades que vaya presentando con el paso del tiempo. A la finalización del mismo nos podemos encontrar que algunas funciones del lenguaje de programación van siendo obsoletas o se puede incrementar la seguridad del mismo o simplemente dar mantenimiento al mismo.

Así mismo, la base de datos aunque es funcional en este momento y cumple con su objetivo, puede ser optimizada en las tablas de formularios de manera que solo se maneje una sola y cada pregunta del formulario señalada por una columna extra. Esto se traduciría de finalmente como una optimización de espacio en disco y mejor rendimiento en las consultas.

Cabe destacar que desde el punto de vista del sistema, podemos presentar de manera numérica tantos datos como se nos permita, los nuevos sistemas que tienden a realizar una interpretación de los mismo aún no son 100% confiables, por lo que se debe resaltar que el sistema es solo una herramienta más para el experto que la usará.

Bibliografía

Capítulo 1

- Wikipedia*, Lenguaje de programación interpretado,* 2013, http://es.wikipedia.org/wiki/Lenguaje_de_programación_interpretado [Consulta, febrero 2013]
- Wikilibro, *Fundamentos de programación, Algoritmos y programas,*2013, http://es.wikibooks.org/wiki/Fundamentos_de_programación/Algoritmos_y_programas [Consulta, febrero 2013]
- Wikipedia, *PHP*, 2013, http://es.wikipedia.org/wiki/PHP [Consulta, febrero 2013]
- Kirk Brown, *Interpreted Language*, 2013, http://perl.about.com/od/programmingglossary/g/interpretedlang.htm [Consulta, febrero 2013]
- Indopedia, *Interpreted Language*, 2013, http://indopedia.org/Interpreted_language.html [Consulta, febrero 2013]
- Wikipedia, *PHP*, 2013, http://en.wikipedia.org/wiki/PHP [Consulta, febrero 2013]
- Manual de PHP, *Introducción*, *Variables*, 2013 http://www.manualdephp.com/manualphp/introduccion-php.html, [Consulta, marzo 2013]
- W3resourse, PHP : *intval() function*, 2013, http://www.hospedajeydominios.com/mambo/documentacion-manual_php-pagina-function_intval.html [Consulta, marzo 2013]
- Php.net, Function date, 2013, http://www.php-es.com/function.date.html [Consulta, marzo 2013]
- Universidad de Granada, *PHP*, 2013, http://atc.ugr.es/pedro/tutoriales/php/php.html [Consulta, marzo 2013]
- Wikipedia, Motor *Zend*, 2013, http://es.wikipedia.org/wiki/Motor_Zend [Consulta, marzo 2013]

Capítulo 2

- Codd, E.F. (1970). "*A Relational Model of Data for Large Shared Data Banks*". Communications of the ACM 13 (6): 377–387. doi:10.1145/362384.362685.
- Jon Barwise and John Etchemendy, 2000. *Language Proof and Logic*. Stanford, CA: CSLI Publications (Distributed by the University of Chicago Press).
- David Hilbert and Wilhelm Ackermann 1950. *Principles of Mathematical Logic* (English translation). Chelsea. The 1928 first German edition was titled Grundzüge der theoretischen Logik.
- Abraham Silberschatz, Henry F. Korth, S. Sudarshan. *Fundamentos de Bases de Datos*. Mc Graw Hill, Cuarta Edición.. España 2002.
- Dr. D. Sergio Gálvez Rojas , *TIPOS DE BASES DE DATOS*, 2010, http://www.lcc.uma.es/~galvez/ftp/bdst/Tema2.pdf [Consulta, marzo 2013]
- Wikipedia. *Base de Datos*., 2013, [Consulta, marzo 2013] http://es.wikipedia.org/wiki/Base_de_datos#Tipos_de_base_de_datos
- Wikipedia,*Cubo OLAP*, 2013, http://es.wikipedia.org/wiki/Cubo_OLAP [Consulta, marzo 2013]

Capítulo 3

- PhpMoot. 2013. Make Safe Input. http://www.phpmoot.com/php-make-safe-input/ [Consulta, marzo 2013]
- Php.net, *Magic Quotes*, 2013, http://php.net/manual/en/security.magicquotes.php [Consulta, marzo 2013]
- Php.net, *AddSlashes*, 2013, http://php.net/manual/es/function.addslashes.php [Consulta, marzo 2013]
- Php.net, *Reserved Variables,* 2013, http://php.net/manual/en/reserved.variables.php [Consulta, marzo 2013]
- Php.net, *Isset,* 2013, http://php.net/manual/es/function.isset.php [Consulta, marzo 2013]
- Php.net, Htmlentities*,* 2013, http://php.net/manual/es/function.htmlentities.php [Consulta, marzo 2013]

Capítulo 4

- Dream Host, *Php.ini*, 2013 http://wiki.dreamhost.com/PHP.ini [Consulta, marzo 2013]
- Php.net, *Arreglo $_SERVER*, 2013, http://php.net/manual/es/reserved.variables.server.php [Consulta, marzo 2013]
- Php.net, *Isset*, 2013, http://php.net/manual/es/function.isset.php [Consulta, marzo 2013]
- HTML Form Guide, *PHP_SELF*, 2013, http://www.html-form-guide.com/php-form/php-form-action-self.html [Consulta, marzo 2013]
- Php.net, *Substr*, 2013, http://php.net/manual/es/function.substr.php [Consulta, marzo 2013]
- Php.net, Rand, 2013, http://php.net/manual/es/function.rand.php [Consulta, marzo 2013]
- Steve George, Ultimate PHP error_reporting,2013, http://www.bx.com.au/tools/ultimate-php-error-reporting-wizard [Consulta, marzo 2013]

GLOSARIO

Motor Zend.- es un motor de procesamiento para la intepretación y cifrado del código php, desde la versión 4. Desarrollado por Zend Tecnologies para brindar un equipo de soporte y acelerar la carga de aplicaciones realizadas con php.

Cubo OLAP.- OnLine Analytical Processing o procesamiento Analítico En Línea, es una base de datos multidimensional, en la cual el almacenamiento físico de los datos se realiza en un vector multidimensional. Los cubos OLAP se pueden considerar como una ampliación de las dos dimensiones de una hoja de cálculo.

Herramienta CASE.- Las **herramientas CASE** (***C****omputer* ***A****ided* ***S****oftware* ***E****ngineering*, Ingeniería de Software Asistida por Ordenador) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero. Estas herramientas nos pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso de realizar un diseño del proyecto, calculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras.

Bugs.- Errores de programación.

Javascript.- es un lenguaje de programación interpretado, se utiliza principalmente en su forma del lado del cliente (client-side), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas.

Mootools.- (**M**y **o**riented **o**bject **tools**) es un Framework web orientado a objetos para JavaScript, de código abierto, compacto y modular. Su objetivo es aportar una manera de desarrollar JavaScript sin importar en qué navegador se ejecute de una manera elegante. Aporta una API documentada más enfocada a la orientación de objetos que la implementación estándar soportada por los navegadores web.

Jquery.- es una biblioteca de JavaScript, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.

Ajax.- (*A**synchronous*** ***J****avaScript* ***A****nd* ***X****ML)*, es una técnica de desarrollo web para crear aplicaciones interactivas. Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones.

Deprecated.- El nivel de error E_DEPRECATED se usa para indicar que una función o funcionalidad está obsoleto. El nivel de error **E_USER_DEPRECATED** tiene por objetivo indicar las funcionalidades obsoletas en el código de usuario.

Http.- **Hypertext Transfer Protocol** o **HTTP** (en español *protocolo de transferencia de hipertexto*) es el protocolo usado en cada transacción de la World Wide Web.

Html.- ***HyperText Markup Language*** (lenguaje de marcado de hipertexto), es el lenguaje de marcado predominante para la elaboración de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes.

Perl.- (*Practical Extraction and Reporting Language*) es s un lenguaje de programación que toma características del lenguaje C, del lenguaje interpretado bourne shell, Lisp y, en un grado inferior, de muchos otros lenguajes de programación. Estructuralmente está basado en un estilo de bloques como los del C y fue ampliamente adoptado por su destreza en el procesado de texto y no tener ninguna de las limitaciones de los otros lenguajes de script.

CGI.- Interfaz de entrada común (en inglés Common Gateway Interface, abreviado CGI) es una importante tecnología de la World Wide Web que permite a un cliente solicitar datos de un programa ejecutado en un servidor web. CGI especifica un estándar para transferir datos entre el cliente y el programa. Es un mecanismo de comunicación entre el servidor web y una aplicación externa cuyo resultado final de la ejecución son objetos MIME. Las aplicaciones que se ejecutan en el servidor reciben el nombre de CGIs.

MIME.- ***Multipurpose Internet Mail Extensions*** o **MIME** (en español "extensiones multipropósito de correo de internet") son una serie de convenciones o especificaciones dirigidas al intercambio a través de Internet de todo tipo de archivos (texto, audio, vídeo, etc.) de forma transparente para el usuario. Una parte importante del MIME está dedicada a mejorar las posibilidades de transferencia de texto en distintos idiomas y alfabetos.

XML.- En inglés *e**X**tensible **M**arkup **L**anguage* (“lenguaje de marcas extensible”), es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML). Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades, de ahí que se le denomine metalenguaje. Algunos de estos lenguajes que usan XML para su definición son XHTML, SVG, MathML.

CSS.- El nombre **hojas de estilo en cascada** viene del inglés *Cascading Style Sheets*, del que toma sus siglas. CSS es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). El W3C (World Wide Web Consortium) es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores.

W3C.- El **World Wide Web Consortium**, abreviado **W3C**, es un consorcio internacional que produce recomendaciones para la World Wide Web.

CASE.- Las herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Computadora) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas en términos de tiempo y de dinero.

SOAP.- **SOAP** (siglas de *Simple Object Access Protocol*) es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML. Este protocolo deriva de un protocolo creado por David Winer en 1998, llamado XML-RPC. SOAP fue creado por Microsoft, IBM y otros y está actualmente bajo el auspicio de la W3C. Es uno de los protocolos utilizados en los servicios Web.

PECL.- (PHP Extension Community Library) es un repositorio de extensiones para el lenguaje de programación PHP.

XPath.- (XML Path Language) lenguaje que permite construir expresiones que recorren y procesan un documento XML. Permite buscar y seleccionar teniendo en cuenta la estructura jerárquica del XML.

DOM.- (Document Object Model) es esencialmente una interfaz de programación de aplicaciones (API) que proporciona un conjunto estándar de objetos para representar documentos HTML y XML, un modelo estándar sobre cómo pueden combinarse dichos objetos, y una interfaz estándar para acceder a ellos y manipularlos. A través del DOM, los programas pueden acceder y modificar el contenido, estructura y estilo de los documentos HTML y XML, que es para lo que se diseñó principalmente.