TEMA 3. PROGRAMACIÓN DE APLICACIONES EN EL SERVIDOR

Curso 2016-2017

Programación Web

Bibliografía

- R. Nixon, «PHP, MySQL, & Javascript», O'Reilly, 2009
- J. N. Robbins, «Learning Web Design», O'Reilly, 4th Edition, 2012
- B.P. Hogan, et al. «Web Development Recipes», Pragmatic Programmers, 2012
- W. Steintmetz, B. Ward, «Wicked Cool PHP», No Starch Press, 2008

Contenido

- Aplicaciones y sitios web
- Lenguaje PHP
- Conexión con BD
- Autenticación, Cookies
- PEAR
- PHP Avanzado

APLICACIONES Y SITIOS WEB

Aplicación web

- Documento diseñado para su difusión a través de la web, accedido mediante navegadores
- Escrito en HTML o XHTML
- Contenido estático. Prefijado y almacenado en disco
- Contenido dinámico: se genera en el momento de la consulta
- Aplicación web: documento con capacidad interactiva

Sitio web

- Conjunto de páginas web relacionadas y albergadas bajo un mismo dominio de Internet. Suelen compartir esquema y autoridad
- Espacio documental estructurado habitualmente dedicado a un tema específico o propósito concreto.
- Las páginas web del sitio son accesibles a través de un URL raíz, denominado “portada”.
- Organización jerárquica (árbol o DAG)

Elementos de una aplicación web

- Interfaz
- Servidor web
- Base de datos
- Lenguaje de programación

LENGUAJE PHP

Lenguaje PHP

- Lenguaje de programación de aplicaciones web en el servidor de propósito general (*PHP Hypertext Pre-Processor*)
- Diseñado para la generación dinámica de contenidos
- Código incrustado en documentos html e interpretado por un módulo del servidor web
- Sintaxis cercana a C y Perl
- Fácil conectividad con SGBD

Script sencillo

hola.php

```
<? php
   echo "Hola. Hoy es".date("l").".";
?>
¿Cómo estás?
```

Elementos del lenguaje PHP

- Sentencias; Comentarios
- Variables
- Operadores
- Tipos de datos; *arrays*
- Concatenadores
- Funciones
- Expresiones y control de flujo
- PDO
- Formularios

Sentencias

- El intérprete se activa para bloques delimitados por:

  <?php …?>

- Todas las sentencias terminan con «;»

Comentarios

```
<? php
  // Esto es comentario de una línea
  /*
    Comentario de varias
    líneas
    */
?>
```

Variables en PHP

```
<? php
   $contador = $contador + 1;
   $nombre = "Luis";
   $vector = array("una", "cadena", "larga");
?>
```

No hay verificación de tipos. Una variable es un espacio para almacenar datos

Operadores

- Aritméticos: +, -, *, /, %, ++, --
- De asignación: =, +=, -=, *=, /=, %=, .=
- Lógicos: &&, ||, not, and, or, xor
- Comparación: ==, !=, <, >, <=, >=
- De bit: !,
- Concatenación de cadenas: . (punto)
- Casting: (int), (double), (string), (array), (object)

Tipos de datos

- Numéricos: enteros, coma flotante
- Lógicos: booleanos
- Cadenas de caracteres:
  - Literales: encerradas entre comillas simples
  - Interpretadas: encerradas entre comillas dobles

Arrays

- Tipo de dato estructurado (no necesariamente homogéneo)
- Se crean con «array()»
- Los componentes se acceden mediante índices, comenzando en 0
- Pueden ser multidimensionales:

```
$matriz = array(array(1, 2, 3), array(4, 5, 6));
```

Constantes

- `define(UBICACION, "/var/www/html");`
- `define(LONGITUD, 34);`
- Constantes predefinidas:
  - `__LINE__`
  - `__FILE__`
  - `__DIR__`
  - `__FUNCTION__`
  - `__CLASS__`
  - …

Funciones

```
<? php
function suma($a, $b)
{
      return $a + $b;
}
$a=3;
$b=4;
echo "La suma de $a y $b es".suma($a,$b);
?>
```

Extensa **biblioteca** de funciones en PHP o módulos

Ámbito de las variables

- Local: en el ámbito de la función en que se definen
- Estáticas: «static»
- Global: «global»
- Superglobal: variables globales predefinidas:

$GLOBALS	$_SERVER	$_GET	$_POST
$_FILES	$_COOKIE	$_SESSION	$_REQUEST
$_ENV			

Expresiones

- Sintaxis habitual de C/C++
- `$y = 2 * abs(5*$x)+4);`
- Asignaciones múltiples:
  - `$y = $x = $z +1`

Control de flujo

- Sentencias condicionales:
  - `if/else`
  - `elseif`
  - `switch`
- Bucles:
  - `while`
  - `do while`
  - `for`
  - Interrupción del flujo: `continue, break`

Control de flujo: condicionales

```
<? php
   if (condicion)
   {
      sentencias true;
   }
   else
   {
      sentencias false;
   }
?>
```

```
<? php
   if (condicion)
   {
      sentencias primera;
   }
   elseif (condicion
segunda)
   {
      sentencias segunda;
   }
?>
```

switch

```
<? php
switch ($pagina)
{
      case "Principal":
            echo "Página principal";
            break;
      case "Noticias":
            echo "Página de noticias";
            break;
      default: echo "Página por defecto";

}
?>
```

Operador ?:

```
<? php
echo $espacio > 0  ? "Hay espacio" :
     "No hay espacio"
?>
```

Bucles while y do-while

```
while (condicion)
{
    cuerpo;
}
```

```
do {
    cuerpo;
} while (condicion);
```

Bucle for

```
<? php
for ($i = 1; $i <= 10; ++$i)
{
     echo "valor: $i";
     echo "<br/>"
}
?>
```

Ruptura de flujo en bucles

- Concluir iteración actual y volver al inicio: `continue;`
- Concluir ejecución del bucle: `break;`

Estructuración del código

- Diseño modular: `varios ficheros .php`
- Sentencias:
  - `include`
  - `include_once`
  - `require_once`

PDO

- PHP soporta el paradigma PDO
- El lenguaje fue actualizado para permitir la definición de clases, objetos, variables de instancia, métodos, herencia y polimorfismo
- También soporta sobrecarga de operadores
- Incluye definición de constructores y destructores

Ejemplo de clase

```
<?php
class phpClass{
   var $var1;
   var $var2 = "constant string";
   function myfunc ($arg1, $arg2) {
     [..]
   }
  [..]
}
?>
```

ARRAYS

Arrays

- Contenedores heterogéneos
- Contenido accesible mediante índices numéricos o contenedor asociativo
- Se pueden usar iteradores para recorrerlos

Arrays indizados con enteros

- Índices a partir de 0
- Se pueden crear con el constructor (array)
- Se pueden añadir elementos:

```
$impresora[] = "laser";
$impresora[] = "de inyección de tinta";
$impresora[] = "matricial";
$impresora[] = "de margarita";
```

Mostrar contenido

```
<? php
   print_r($paper);

   for ($j = 0; $j < 4; ++$j)
     echo "$j: $paper[$j]<br>";
 ?>
```

Arrays asociativos

- Indizados mediante clave:

```
<? php
$fruta['naranja'] = "navel";
$fruta['melon'] = "piel de sapo";
$fruta['manzana'] = "fuji";

echo $paper['melon'];
?>
```

Iterando sobre arrays

- Con la estructura «for each»:

```
<? php
$fruta = array ("naranja", "melon",
"manzana");

foreach ($fruta as $pieza)
     echo "$item<br>";
?>
```

Iterando sobre arrays asociativos

```
<? php
$fruta = array ("naranja" => "navel,
                "melon" => "piel de sapo",
                "manzana" => "fuji");

foreach ($fruta as $pieza => $variedad)
     echo "$item: $variedad<br>";
?>
```

Arrays multidimensionales

```
<? php
$chessboard = array(
array('r', 'n', 'b', 'q', 'k', 'b', 'n', 'r'),
array('p', 'p', 'p', 'p', 'p', 'p', 'p', 'p'),
array(' ', ' ', ' ', ' ', ' ', ' ', ' ', ' '),
array(' ', ' ', ' ', ' ', ' ', ' ', ' ', ' '),
array(' ', ' ', ' ', ' ', ' ', ' ', ' ', ' '),
array(' ', ' ', ' ', ' ', ' ', ' ', ' ', ' '),
array(' ', ' ', ' ', ' ', ' ', ' ', ' ', ' '),
array(' ', ' ', ' ', ' ', ' ', ' ', ' ', ' '),
array('P', 'P', 'P', 'P', 'P', 'P', 'P', 'P'),
array('R', 'N', 'B', 'Q', 'K', 'B', 'N', 'R'));
echo "<pre>";
foreach ($chessboard as $row)
{
        foreach ($row as $piece)
                echo "$piece ";
        echo "<br />";
}
echo "</pre>";
```

Funciones para arrays

- is_array
- count
- sort
- shuffle
- explode
- reset
- end

- extract:
  - $_GET
  - $_POST
- compact:

FORMULARIOS

Formularios

- Principal medio de interacción con el usuario en PHP
- Proceso:
  - Creación del formulario
  - Envío de datos al servidor
  - Interpretación y filtrado en el servidor
  - Acciones
  - Generación de contenido

Construcción de formularios

- Encerrados entre `<form> … </form>`
- Tipo de envío: GET o POST
- Campos de entrada
- URL de envío

Ejemplo de formulario

```
<? php
<html>
<head>
  <title>Formulario de ejemplo</title>
</head>
<body>
<form method="post" action="formtest.php">
   Introduzca su nombre:
   <input type="text" name="nombre" />
   <input type="submit" />
 </form>
 </body>
 </html
?>
```

```
<? php
if (isset($_POST['name'])) $name = $_POST['name'];
else $name = "(Not entered)";

<html>
<head>
  <title>Formulario de ejemplo</title>
</head>
<body>
Tu nombre es: $name<br />
<form method="post" action="formtest.php">
   Introduzca su nombre:
   <input type="text" name="nombre" />
   <input type="submit" />
 </form>
 </body>
 </html
?>
```

Componente «form»

- Atributo «action»: script para procesar el formulario (php, js, python, perl, …)
- Atributo «method»:
  - POST: El navegador envía una petición independiente con la información. Sólo la ve el servidor
  - GET: La información enviada se incluye en el URL usando los caracteres ? y %.

Asignación a variables

- Antes $_POST y $_GET se asignaban a variables PHP
- Problema de seguridad por lo que se deshabilita el uso de «register_global»

Valores por defecto

```
<form method="post" action="calc.php">
<pre>
Loan Amount <input type="text"
name="principle" />
Monthly Repayment <input type="text"
name="monthly" />
Number of Years <input type="text"
name="years" value="25" />
Interest Rate <input type="text" name="rate"
value="6" />
</pre>
```

Elementos en un formulario

- Cajas de texto (*Text boxes*)
- Cajas de marca (*Checkboxes*)
- Botones de radio (Radio buttons)
- Campos ocultos
- Etiquetas

Campos de texto

- **Atributos**: name, maxlength, size, value

- **text**: una sola línea de texto

```
<label>City <input type="text" name="city"
id="form-city‟ value="Your Hometown"
maxlength="50"></label>
```

- **textarea**:

```
<textarea name="comment" rows="4" cols="45"
placeholder="Leave us a comment."></textarea>
```

- **password**:

```
<input type="password" name="pswd"
maxlength="8" id="form-pswd">
```

Campos de texto (HTML5)

- `<input type="search">`
- `<input type="email">`
- `<input type="tel">`
- `<input type="url">`

Campos de fecha y hora (HTML5)

- `<input type="date">`
- `<input type="time">`
- `<input type="datetime">`
- `<input type="datetime-local">`
- `<input type="month">`
- `<input type="week">`

Campos numéricos (HTML5)

- `<input type="number">`
- `<input type="range">`

Botones de envío y reinicio

- `<input type="submit">`
- `<input type="reset">`

Botones de radio

```
<p>¿Cual es tu edad?</p>
<ol>
<li><input type="radio" name="age"
value="Menor de 24" checked> under
24</li>
<li><input type="radio" name="age"
value="25-34"> 25 to 34</li>
<li><input type="radio" name="age"
value="35-44"> 35 to 44</li>
<li><input type="radio" name="age"
value="Mayor de 45"> 45+</li>
</ol>
```

Botones de marcado

```
<p>¿Qué tipo de música prefieres?</p>
<ul>
<li><input type="checkbox" name="genre"
value="punk" checked> Pop rock</li>
<li><input type="checkbox" name="genre"
value="indie" checked> Folclórica rock</li>
<li><input type="checkbox" name="genre"
value="hiphop"> Rap</li>
<li><input type="checkbox" name="genre"
value="rockabilly"> Rock</li>
</ul>
```

Menúes

```
<p>¿Cuál es tu banda favorita de los 80?
<select name=«Favorito80">
<option>The Cure</option>
<option>Cocteau Twins</option>
<option>Tears for Fears</option>
<option>Dire Straits</option>
<option value="EBTG">Everything But the
Girl</option>
<option>Tue Queen</option>
</select>
```

Selección de archivos

```
<form action="/client.php" method="POST"
enctype="multipart/form-data">
<label>Envía una foto como icono <em>(optional)</
em><br>
<input type="file" name="photo" size="28"><label>
</form>
```

Campo oculto

- Enviar información que no proporciona el usuario directamente:

```
<input type="hidden" name="success-link"
value="http://www.example.com/
littlechair_thankyou.html">
```

- Ejemplo de uso: crawler-trap

MANEJO DE ARCHIVOS Y OTRAS FUNCIONES

Funciones de fecha y hora

```
time()
date($format, $timestamp)
checkdate($month, $day, $year)
```

Manejo de archivos

- Gestión de archivos en local (servidor)
- `file_exists:`

```
if (file_exists("testfile.txt"))
    echo "El fichero existe";
```

- Creación de ficheros:
  - Apertura
  - Escribir
  - Cerrar

Ejemplo de creación

```
<?php
  $fh = fopen("fichero.txt«, ‘w’) or die("No se puede
crear");
  $text = <<<_END
  Linea 1
  Linea 2
  Linea 3


  _END
  fwrite($fh, $text) or die("No puedo escribir en el
fichero");
  fclose($fh);
  echo "Fichero 'fichero.txt' creado correctamente";
  ?>
```

Ejemplo de lectura

```
<?php
$fh = fopen(“fichero.txt", 'r') or
die(“El fichero no existe o no tienes
permisos");
$line = fgets($fh);
fclose($fh);
echo $line;
?>
```

Lectura del fichero completo

```
<?php
echo "<pre>"; // Enables display of line feeds
echo file_get_contents("testfile.txt");
echo "</pre>"; // Terminates pre tag
?>
```

Copiar, renombrar, borrar

```
<?php // copyfile.php
copy('testfile.txt', 'testfile2.txt') or die("Could not copy
file");
echo "File successfully copied to 'testfile2.txt'";
?>

<?php // movefile.php
if (!rename('testfile2.txt', 'testfile2.new'))
echo "Could not rename file";
else echo "File successfully renamed to 'testfile2.new'";
?>

<?php // deletefile.php
if (!unlink('testfile2.new')) echo "Could not delete file";
else echo "File 'testfile2.new' successfully deleted";
?>
```

Subir ficheros

```
<?php // upload.php
echo <<<_END
<html><head><title>PHP Form Upload</title></head><body>
<form method='post' action='upload.php' enctype='multipart/
form-data'>
  Select File: <input type='file' name='filename' size='10' />
  <input type='submit' value='Upload' />
</form>
_END;
if ($_FILES)
{
  $name = $_FILES['filename']['name'];
  move_uploaded_file($_FILES['filename']['tmp_name'], $name);
  echo "Uploaded image '$name'<br /><img src='$name' />";
}
echo "</body></html>";
?>
```

Llamadas a sistema

```
<?php // exec.php
$cmd = "dir"; // Windows
// $cmd = "ls"; // Linux, Unix & Mac
exec(escapeshellcmd($cmd), $output,
$status);
if ($status)
 echo "Exec command failed";
else
{
  echo "<pre>";
  foreach($output as $line) echo "$line\n";
}
?>
```

Función printf

- La misma funcionalidad de la función `printf` de C

```
printf("My name is %s. I'm %d years old,
which is %X in hexadecimal",'Simon', 33,
33);
```

ACCESO A BASES DE DATOS

Bases de Datos

- Muchas aplicaciones web se apoyan en una BD para datos
- El diseño e implementación de la BD es una **tarea esencial** en estas aplicaciones
- Manejo de las BD: operaciones CRUD
- Consultas

Recordando conceptos

- Esquema
- Base de datos
- Tabla
- Fila
- Columna
- Consultas

Acceso a bases de datos

- Uso de BD para almacenamiento de datos
- Diseño e implementación adecuadas de las BDs
- SGBD: relacionales, jerárquicos o NoSQL
- Lenguaje de consulta SQL
- SGBD *open-source*:
  - MySQL (MaríaDB)
  - PostgreSQL
  - …

MySQL

- SGBD relacional, multihebra y multiusuario
- Soporta integridad referencial, transacciones, replicación
- Simple, ágil, versátil
- Multiplataforma: Linux, Windows, Mac OS X, …
- Lenguajes de programación: PHP, C, C++, Pascal Delphi, Python, Java, Perl, Eiffel, Smalltalk, Lisp, …
- La más utilizada para aplicaciones web
- Distintos motores de almacenamiento e indización

Instalación y configuración de MySQL

- Elemento esencial de sistemas LAMP y WAMP
- Instalable desde código fuente y en distintos paquetes (.exe, .rpm, .deb, …)
- Componentes:
  - Servidor
  - Herramientas de cliente
  - Conectores

Paquetes en Fedora

- mysql
- mysql-server
- mysql-libs
- php-mysql
- mysql-test
- mysql-connector-java
- mysql-connector-odbc
- mysql-utilities

Configuración, directorios y binarios

- /etc/my.cnf
- /var/lib/mysql
- /var/run/mysqld
- /var/log/mysqld.log

- mysql
- mysqladmin
- mysqldump

Acceso a MySQL

- Por línea de órdenes: mysql
- GUI: phpMyAdmin, MySQL Workbench
- A través de aplicaciones:
  - Conectores: ODBC, JDBC
  - Java
  - Python
  - Perl
  - PHP

Usuarios

- DBA (root)
- Usuarios: CREATE USER
- Privilegios:

```
GRANT ALL PRIVILEGES ON DB.* TO
'usuario'@'localhost IDENTIFIED BY 'passwd';
update user set password=PASSWORD("clave-secreta")
where USER="usuario";
```

Tipos de datos

Categoría	Tipos
Numérico	INTEGER, INT, SMALLINT, TINYINT, MEDIMUINT, FLOAT, DOUBLE, BIT
Cadena de caracteres	CHAR, VARCHAR, VARBINARY
Fecha y hora	DATE, DATETIME, TIMESTAMP, TIME, YEAR
Texto	TEXT, TINYTEXT, MEDIUMTEXT, LONGTEXT
Binarios	TINYBLOG, BLOB, MEDIUMBLOB, LONGBLOB

Índices

- Permiten localizar y devolver registros de forma sencilla y rápida
- Implementación interna basada en B-trees y derivados
- PRIMARY KEY … UNIQUE INDEX …
- ALTER TABLE personas ADD INDEX (apellido);

Consulta SELECT

- Sentencia básica para recuperar información: SELECT
- `SELECT * FROM alumnos;`
- `SELECT nombre, apellido FROM alumnos WHERE apellido LIKE 'B%';`
- Cláusulas:
  - `GROUP BY, HAVING, ORDER BY, LIMIT`

INSERT

- Inserción de datos en tablas
- `INSERT INTO alumnos (nombre, apellido) VALUES ('Luis', 'Escobar');`

UPDATE

- Actualización de información
- `UPDATE alumnos SET nombre = 'Juan' WHERE dni='11223344';`

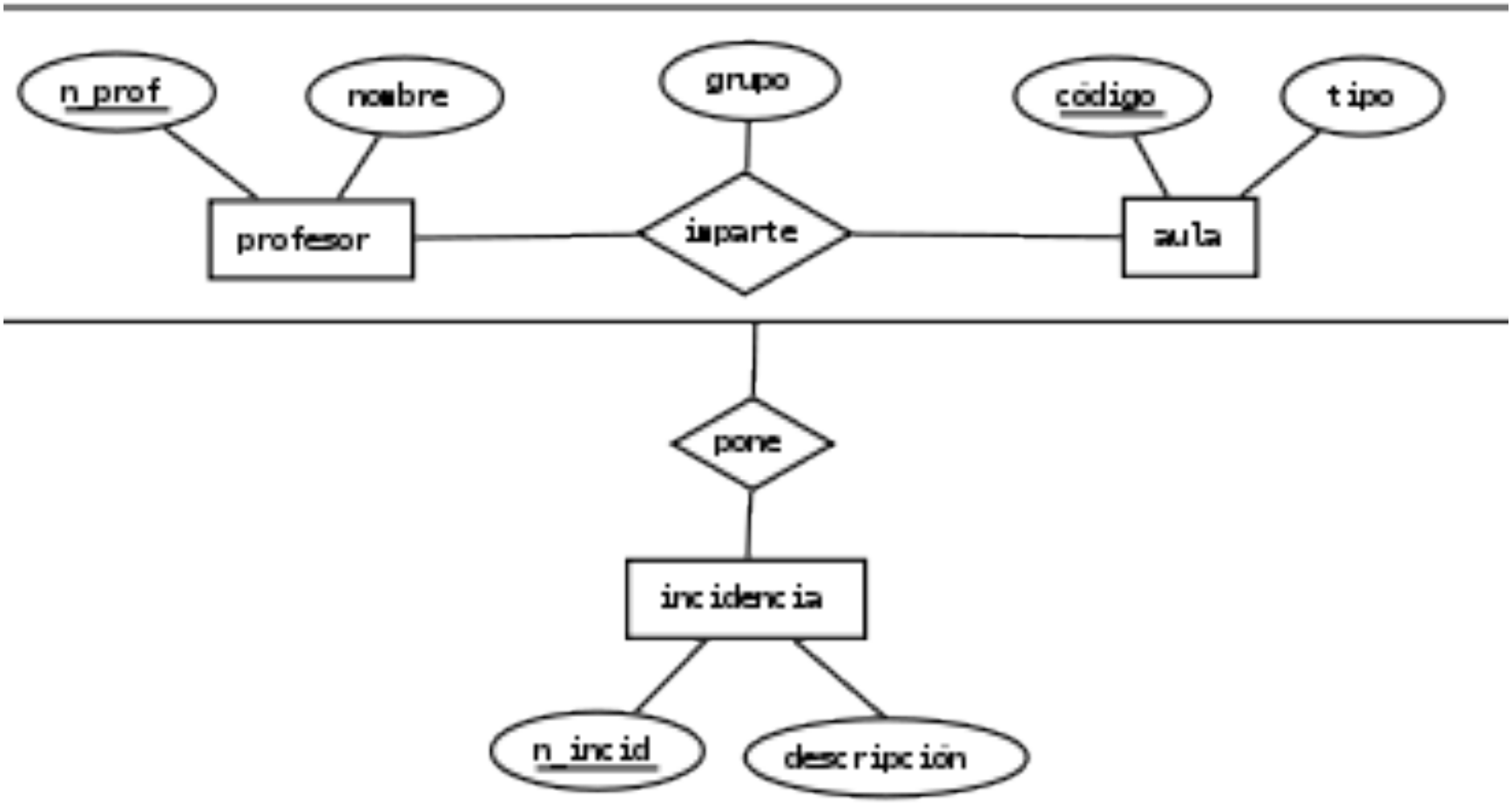
DELETE

- Eliminar filas de una tabla
- `DELETE FROM alumnos WERE dni = ‘11223344’;`

Diseño relacional

- Diagramas Entidad-Relación
- Esquemas
- Llaves primarias
- Llaves externas
- Reglas de integridad
- Formas normales

Diagramas entidad-relación



Relaciones

- Las BDs almacenan datos factuales y relaciones entre éstos
- Relaciones 1:1
- Relaciones 1:n
- Relaciones n:m

Transacciones

- Transacción: garantizar la ejecución íntegra y ordenada de una secuencia de consultas
- P.ej.: transferencia de fondos
- Soporte basado en el motor de almacenamiento (InnoDB)

```
BEGIN
UPDATE cuenta SET saldo = saldo+25 WHERE ID = 389;
COMMIT;
```

- Cancelación: ROLLBACK;

Copias de seguridad

- Volcado de datos en texto (también CSV)
  - mysqldump –u *user* –p*clave* *basedatos* > bd.sql

- Restauración de la BD
  - mysql –u *user* –p*clave* < bd.sql

Acceso a MySQL con PHP

1. Conexión
2. Seleccionar BD
3. Construir la cadena de consulta
4. Ejecutar la consulta
5. Recuperar los resultados y construir la página web
6. Repetir 3 a 5 las veces necesarias
7. Desconectar

1. Conexión

```
<?php
$db_hostname = 'localhost';
$db_database = 'basedatos';
$db_username = 'usuario';
$db_password = 'clave';

$db_conn = mysql_connect($db_hostname,
    $db_username, $db_password);
if (!$db_conn)
  die("No puedo conectar con MySQL: " .
  mysql_error());
?>
```

2. Seleccionar BD

```
<?php
mysql_select_db($db_database)
  or die("No puedo seleccionar la BD: " .
mysql_error());
?>
```

3. Construir la cadena de consulta
4 Ejecutar la consulta

```
<?php
   $consulta = "SELECT * FROM ALUMNOS";
   $result = mysql_query($consulta);

   if (!$result)
      die ("Error en el acceso a la BD: "
      . mysql_error());
?>
```

5. Recuperar los resultados y construir la página web

```
<?php
  $num_filas = mysql_num_rows($result);

  for ($j = 0; $j < $num_filas; ++$j)
  {
    echo "Nombre: " . mysql_result($result, $j,
       'nombre') . ' <br />';
    echo "Apellido: " . mysql_result($result, $j,
       'apellido') . ' <br />';
    echo "Curso: " . mysql_result($result, $j,
'curso')
       . ' <br /><br />';
  }
?>
```

Más eficiente

```
<?php
  $num_filas = mysql_num_rows($result);

  for ($j = 0; $j < $num_filas; ++$j)
  {
    $fila = mysql_fetch_row($result);
    echo "Nombre: " . $fila[0] . ' <br />';
    echo "Apellido: " . $fila[1] . ' <br />';
    echo "Curso: " . $fila[2] . ' <br /><br />';
  }
?>
```

7. Desconectar

```
<?php
   mysql_close($db_server);
?>
```

Acceso a BD con clases

```
try {
   $conn = new
      PDO('mysql:host=localhost;dbname=myDatabase',
      $username, $password);
    $conn->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
} catch(PDOException $e) {
    echo 'ERROR: ' . $e->getMessage();
}
```

Consultas

```
$name = 'Joe'; # user-supplied data

try {
    $conn = new PDO('mysql:host=localhost;dbname=myDatabase', $username, $password);
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    $data = $conn->query('SELECT * FROM myTable WHERE name = ' . $conn->quote($name));

    foreach($data as $row) {
        print_r($row);
    }
} catch(PDOException $e) {
    echo 'ERROR: ' . $e->getMessage();
}
```

Sentencias “prepare”

```
$id = 5;
try {
    $conn = new PDO('mysql:host=localhost;dbname=myDatabase', $username, $password);
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    $stmt = $conn->prepare('SELECT * FROM myTable WHERE id = :id');
    $stmt->execute(array('id' => $id));

    while($row = $stmt->fetch()) {
        print_r($row);
    }
} catch(PDOException $e) {
    echo 'ERROR: ' . $e->getMessage();
}
```

Sitios con código de ejemplo

- http://www.php-scripts.com/php_diary/php_scripts.html
- http://www.java2s.com/Code/Php/CatalogPhp.htm
- http://php.happycodings.com/
- https://code.tutsplus.com/tutorials/php-database-access-are-you-doing-it-correctly--net-25338

Selección de la API

- Mysql (MySQL 4.1.3 y anteriores)
- Mysqli (necesaria para usar todas las características del servidor MySQL)
- PDO

http://www.php.net/manual/es/mysqlinfo.api.choosing.php

Ventajas de mysqli

http://php.net/manual/es/set.mysqlinfo.php

- Interfaz DO
- Soporte para sentencias múltiples
- Soporte para transacciones
- Mejoras en el soporte para depuración
- Soporte para servidores incrustados

API mysqli

```
<?php
$mysqli = new mysqli("ejemplo.com", "usuario",
  "contraseña", "basedatos");
$resultado = $mysqli->query("SELECT * FROM
   tabla");
$fila = $resultado->fetch_assoc();
echo htmlentities($fila['_message']);
?>
```

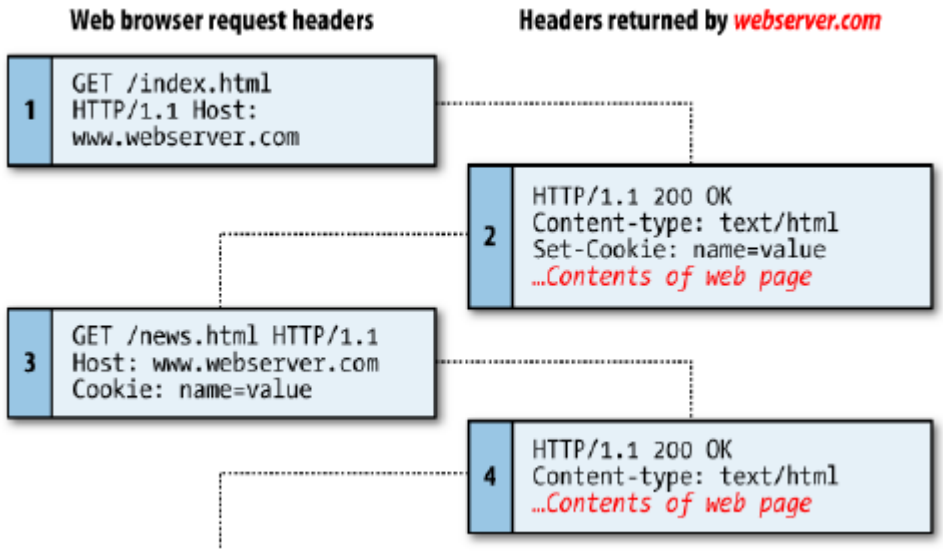
COOKIES, SESIONES, AUTENTICACIÓN

Cookies

- Mantener información sobre los usuarios
- No necesariamente nombres de usuario y claves; recuperar información para sus próximas conexiones
- Cookie: Dato que almacena un servidor web en el equipo local del cliente (de tamaño <= 4KB)
- Privacidad: sólo se pueden leer desde el servidor que las crea
- Autorización por parte del cliente

Sesión con cookies

- Las cookies se intercambian durante la transferencia de las cabeceras, antes del envío de la página html



Crear una *cookie*

```
setcookie(name, value, expire, path, domain,
secure, httponly);


setcookie('username', 'Hannah', time() + 60
* 60 * 24 * 7, '/');
```

Parámetros de la cookie

- *name*: nombre
- *value*: hasta 4KB de caracteres alfanuméricos
- *expires*: fecha (Unix timestamp); cierre de navegador
- *path*: trayectoria en el servidor
- *domain*: dominio del servidor
- *secure*: ¿requiere conexión segura?
- *httponly*: ¿usar protocolo http?

Uso y eliminación de cookies

- **Acceso** a una cookie:

```
if (isset($_COOKIE['username']))
$username = $_COOKIE['username'];
```

- **Eliminar** una cookie:

```
setcookie('username', 'Hannah', time()
- 2592000, '/');
```

Autenticación HTTP

- Autenticación de usuarios para el acceso al servidor web. Requiere soporte del servidor (apache: mod_authz_ldpa, _kerb, _mysql, _shadow, _pam,)

```
<?php
if (isset($_SERVER['PHP_AUTH_USER']) &&
     isset($_SERVER['PHP_AUTH_PW']))
{
  echo "Usuarior: " . $_SERVER['PHP_AUTH_USER'] .
   " Clave: " . $_SERVER['PHP_AUTH_PW'];
}
else
{
  header('WWW-Authenticate: Basic realm="Restricted Section"');
  header('HTTP/1.0 401 Unauthorized');
  die("Por favor, inserte su nombre de usuario y clave");
}
?>
```

Almacenamiento de logins y claves

- Uso de funciones de un único sentido para guardar información sensible:

```
$token = md5('mypassword');
```

Valor de `$token`:

`34819d7beeabb9260a5c854bc85b3e44`

- Salpimentar (añadir caracteres antes y después de la clave, previo a su cifrado):

```
$token = md5('hqb%$tmypasswordcg*l');
```

Sesiones

- Grupos de variables/valores almacenados en el servidor, relacionados con el usuario actual.
- La relación (usuario; sesión) se fija mediante cookies.
- Las sesiones se inician llamado a **session_start**.
- Las variables de sesión se guardan en el array $_SESSION.
- La sesión se finaliza llamando a **session_destroy**.

FRAMEWORKS

¿Qué es un *framework*?

- Marco de trabajo: conjunto estandarizado de conceptos y procedimientos para abordar problemas de un mismo tipo
- Estructura conceptual y tecnológica para desarrollo de módulos software reutilizables para soluciones problemas similares. Incluyen programas, bibliotecas y lenguaje
- Model-View-Controller

Ventajas

- Estandarización
- Reutilización de código
- Más eficiencia
- Rapidez en el desarrollo

Inconvenientes

- Limitado a un tipo de problemas. Sólo los contemplados en el framework
- A veces, difícil de adaptar a problemas distintos

PHP Frameworks

- www.phpframeworks.com:
  - Zend
  - CakePHP
  - QPHP
  - Symfony

Zend (framework.zend.com)

- Modular, con muy bajo acoplamiento
- Seguro
- Soporte para PHPUnit
- Extensible
- Alto rendimiento

MÁS SOBRE DESARROLLO EN PHP

PEAR: PHP Extension and Application Repository

- Entorno de desarrollo y sistema de distribución de componentes de código PHP
- Catálogo extenso de bibliotecas de PHP
- Desarrolladas con PDO
- pear.php.net

Desarrollo PHP basado en patrones

- Reutilizar un esquema definido, basado en el soporte natural del lenguaje de programación. Libro «Gang of the Four»
- Programación más efectiva y segura.
- W. Sanders, «Learning PHP Design Patterns», O'Reilly, 2013

Otros lenguajes y frameworks

- Python, Django
- Ruby, Ruby on Rails
- Java, Swing
- Scala, Lift
- Clojure, Luminus
- …