Proyecto Final de Carrera

Raquel Boullón Garzón
Plácido Rodríguez Laredo
Profesor: Sebastià Vila
Ing. Téc. en Telecomunicaciones, esp en Sistemas Electrónicos

ÍNDICE

BLOQUE I : Objetivos y entorno de trabajo

1. Objetivo **1**

2. Elementos de trabajo **2**
   2.1. Servidor 2
      2.1.1. Sistema Operativo 2
      2.1.2. Servidor web 3
      2.1.3. Servidor telnet 4
      2.1.4. Servidor ftp 5
      2.1.5. Servidor de correo 6
   2.2. Conexión con CERVI, S.A. 8

3. Lenguajes para el desarrollo web **11**
   3.1. PHP 11
      3.1.1. Ventajas e inconvenientes de PHP 11
      3.1.2. PHP vs ASP 12
   3.2. Estandard SQL. 13
   3.3. Estandares para el diseño web 14
      3.3.1. Estandars de html. 14
         3.3.1.1.HTML 4 y HTML 4.01 15
      3.3.2. Estandars de css 15
   3.4. Estandares de Javacrip 16
      3.4.1. Problemas entre versiones 17

BLOQUE II : Creación de la base de datos

4. Conceptos básicos de bases de datos **18**

5. Características de los diferentes gestores de bases de datos existentes en el mercado **20**
   5.1. Mysql 3.XX.XX 20
   5.2. PostgreSql 7.3 21
   5.3. Interbase 6.0 22
   5.4. Sap DB 7.2 22
   5.5. Microsoft Sql Server 23
   5.6. Oracle 24
   5.7. Comparativas 25
      5.7.1. Comparativa de características 25
      5.7.2. Comparativa de rendimientos 27

5.8.Conclusion 31

6. Mysql: La opción escogida **33**
6.1. Claves Primarias 33
6.2. Claves Foráneas 33
6.3. Tipos de tablas en Mysql 35
6.4. Mysqladmin: El Administrador de Mysql 47

7. Diseño y creación de la base de datos **38**
7.1. Diseño Conceptual 40
7.1.1 Diagrama ER 45
7.2. Diseño lógico 47
7.3. Diseño Físico 50

BLOQUE III : Aplicación para la gestión de la base de datos

8. Sistema de gestión de la base de datos **52**
8.1. Arquitectura de la aplicación 52
8.2. Aplicaciones principales 53
8.2.1. Validación de usuarios 53
8.2.2. Gestión de artículos 54
8.2.3. Gestión de fabricantes 55
8.2.4. Gestión de categorias/tipos 56
8.2.5. Gestión de ofertas 56
8.2.6. Gestión de pedidos 57
8.2.7. Gestión de usuarios 58
8.2.8. Gestión de eventos 59
8.2.9. Gestión de descuentos 59
8.2.10. Gestión de clientes 60
8.3. Aplicaciones de soporte 61
8.4. Medidas de seguridad 62
8.4.1. La seguridad a nivel de usuarios 62
8.4.2. La seguridad a nivel de canal 64
8.4.3. La seguridad a nivel de programación 65
8.5. Diferencias con los estandars 67

BLOQUE IV : Creación del Web Site

9. WebSite **69**
9.1. La Portada 69
9.2. Quienes somos 71
9.3. Web Técnica 72

9.4. El buscador y el catálogo de artículos 73
9.5. Zona de acceso de Clientes 76

BLOQUE V : Conclusiones y Trabajo futuro

10. Conclusiones 77
11. Trabajo futuro 78

BLOQUE VI : Bibliografía

12. Bibliografía. 79

ANEXOS

I. Esquema de la base de datos i
II. Leyes y legislaciones. v

BLOQUE I

Objetivos y entorno de trabajo

MEMORIA

1. Objetivos.

El objetivo de este proyecto tiene como finalidad la creación de un web site para la empresa Cervi, S.A., una de las firmas más importantes del sector de la distribución de cables eléctricos y componentes para comunicaciones en nuestro país.

Somos conscientes de que la implementación de un proyecto de tal envergadura se tiene que llevar a cabo en diversas fases o etapas, pues hay que ver la aceptación por parte de los clientes y analizar los resultados de cada fase.

Las fases de implantación serían las siguientes:

- Primera fase: diseño de la base de datos, diseño y creación del sistema de gestión de la base de datos, diseño general del web site y creación de un catálogo digital accesible desde la web.

- Segunda fase: Implementación de un carrito de la compra únicamente para clientes de Cervi, S.A. No se realizarán transferencias bancarias y los pedidos generados se contabilizarán por las vías habituales de Cervi, S.A.

- Tercera fase: Apertura de la tienda virtual al resto de los usuarios de la red y pago mediante tarjeta electrónica o contra reembolso para los usuarios que no tengan código de cliente de Cervi, S.A.

Dado que el periodo para la realización del proyecto es de cuatro meses y que nosotros no somos profesionales del sector es lógico que nuestro proyecto únicamente abarque la primera fase.

El objetivo de nuestro proyecto pretende llevar a cabo la primera fase de la implementación, es decir, diseñaremos la base de datos y la crearemos, crearemos un sistema de gestión para mantener la base de datos y diseñaremos un website con un catálogo digital accesible desde la web.

2. Entorno de trabajo

En este punto describiremos las características básicas de nuestro servidor. Daremos detalles sobre el servidor web, el servidor telnet, servidor de ftp y el servidor de correo, también explicaremos como esta estructurada la red de cervi, así como las modificaciones realizadas en los ficheros de configuración.

2.1 El servidor

2.1.1 El sistema operativo.

EL sistema operativo que hemos escogido para nuestro servidor es un Caldera Open Linux de SCO, concretamente la versión 2.3, una configuración de linux para aplicaciones comerciales.

Este es el sistema que utiliza Cervi, S.A. en el resto de sus servidores por lo que nuestra decisión del sistema estaba ya condicionada. Además este sistema tiene otra característica que nos hizo decidirnos por esta configuración en vez de otras configuraciones más comerciales como Suse: es totalmente gratuito, no requiere de licencia y el numero de usuarios no está limitado.

Linux es un sistema operativo de código libre basado en unix. Que sea de código libre significa que es gratuito (sólo algunas configuraciones) y que es desarrollado simultáneamente por millones de usuarios alrededor del mundo.

Un sistema linux está basado en un kernel o nucleo que permite que el software y el hardware trabajen juntos. Las tres funciones más importantes, aunque no las únicas, son:

- Administración de la memoria para todos los programas en ejecución.
- Administración del tiempo de procesador que utilizan los programas en ejecución.
- Acceso cómodo a los periféricos

Es un sistema mutiusuario y multitrea. El usuario principal es root que es el usuario que tiene todos los permisos de ejecución, por lo tanto puede borrar, crear e instalar sin ningún tipo de restricción.

Nuestra distribución esta configurada como servidor web mediante APACHE, servidor de mysql y servidor de correo con el Sendmail. Además puede ejercer las funciones de administrador de ficheros compartidos en una red local aunque es una característica en la que nosotros no entramos ya que las características del proyecto no lo requieren.

El interfaz gráfico utilizado es Xwindows que es un sistema basado en dos aplicaciones:

- El servidor X que es el que se encarga realmente de dibujar en la pantalla.
- Gestor de ventanas: Kde o Gnome.

En nuestro caso el gestor de ventanas es Kde, concretamente la primera versión. Gnome es más utilizado en otras distribuciones como Red Hat.

2.1.2 El servidor Web.

Una vez más el software utilizado por Cervi, S.A. fuerza nuestra decisión sobre este aspecto. Lo que intentaremos explicar en esta sección es porque Cervi, S.A. escogió este servidor web: Apache 1.3.9.

Apache es el servidor web más usado en servidores Unix/Linux, aproximadamente el 40% de la cuota de mercado, básicamente por dos motivos:

- Es libre.
- Es robusto y sencillo de manejar.

Apache es realmente sencillo de instalar y el software esta disponible en dos formatos: precompilado y el código fuente para compilar. Además las versiones precompiladas están disponibles par la mayoría de sistemas operativos existentes en el mercado (FreeBSD, Linux, Solaris, Windows, Unix, etc...).

Es fácilmente configurable ya que solo requiere pequeñas modificaciones en tres archivos, y esto en el peor de los casos. Lo normal es añadir el directorio donde estarán alojadas todas las paginas disponibles y añadir algún modulo (en nuestro caso php3) que no viene configurado por defecto.

Para este proyecto sólo modificamos el archivo httpd.conf que se encuentra en el directorio etc /httpd/conf, a continuación detallamos las modificaciones.

```
##
## httpd.conf -- Apache HTTP server configuration file
##

[...]
```

Primero añadimos el modulo para la compilación de php.

```
<IfDefine PHP>
    Include /etc/httpd/conf/modules/mod_php.conf
</IfDefine>
```

Definimos el puerto por defecto.

```
# Port: The port to which the standalone server listens. For
# ports < 1023, you will need httpd to be run as root initially.
#
Port 80
```

Definimos el usuario. Al definirlo como nobody restringimos el acceso a otras característica del sistema, como crear o eliminar archivos, ya que este usuario tiene pocos privilegios.

```
# User/Group: The name (or #number) of the user/group to run httpd as.
#  . On SCO (ODT 3) use "User nouser" and "Group nogroup".
#  . On HPUX you may not be able to use shared memory as nobody, and the
#    suggested workaround is to create a user www and use that user.
#  NOTE that some kernels refuse to setgid(Group) or semctl(IPC_SET)
#  when the value of (unsigned)Group is above 60000;
#  don't use Group nobody on these systems!
#
User nobody
Group nobody
```

Definimos el directorio donde estarán almacenadas todas las paginas. De esta manera evitamos que el usuario tenga acceso al directorio raiz.

```
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot "/home/httpd/html"
```

Apache ofrece además todas las características de otros servidores web de pago como por ejemplo soporte para plug-and-play, soporte de las especificaciones estandars de html, soporte para servidores virtuales (esto permite alojar diferentes web sites dentro de la misma maquina), módulos ssl para la seguridad o soporte de cookies para la autentificación de usuarios.

Para ampliar la información sobre este servidor véase [APA].

2.1.3 Servidor Telnet.

Telnet es el programa que permite abrir una sesión (login) en un ordenador remoto. En la mayoría de los casos, se realiza la emulación de un terminal de tipo VT100 (estandard del terminal virtual creado por DEC cuya primera versión fue la 100). Proporciona en enlace bidireccional de 8 bits.

Las direcciones Telnet suelen tener el formato del nombre de dominio “maquina.remota.es” o de dirección de Ip “**192.168.1.2**” y pueden ir acompañadas de un numero final (el numero

de puerto). Por defecto y si no se especifica lo contrario el numero de puerto por defecto es el 23.

En nuestro caso el servidor que viene por defecto con la configuración linux es netkit-telnet, concretamente la versión 0.12-5S.

El comando de sentencias seria el siguiente:

```
[root@x220a conf]# telnet 192.168.10.2
  Trying 192.168.10.2...
  Connected to 192.168.10.2.
  Escape character is '^]'.
```

Una vez establecida la conexión, la maquina remota debemos autentificarnos como usuarios, por lo cual el sistema nos pedirá un login y un password

```
Caldera OpenLinux eServer
Versión 2.3
Copyright 1996-1999 Caldera Systems, Inc.

login: raquel
Password:
Last login: Mon Nov 10 19:39:49 2003 from 192.168.1.138 on
ttyp0

Welcome to your OpenLinux system!

[raquel@cervi raquel]$
```

Una vez echo esto la pantalla nos nuestra el símbolo del sistema por lo que solo nos queda convertirnos en superusuarios (root) si fuera necesario.

2.1.4 Servidor de ftp

Este programa establece una conexión entre dos ordenadores permitiendo en intercambio de ficheros entre ambos. Las siglas FTP significa ***file transfer protocol***, protocolo de transferencia de ficheros.

También es necesario conocer la dirección IP del servidor con el que queremos conectar.

La conexión se realizaría de la siguiente manera:

```
ftp 192.168.10.2
```

Si tuviéramos que especificar un puerto se escribiría a continuación de la dirección con la que queremos establecer conexión. Para una transferencia normal se utiliza el puerto 21,

aunque linux presenta una peculiaridad. Para hacer una transferencia ftp utiliza dos puertos, el 20 y el 21, por uno envía los datos o archivos y por el otro los comandos de control.

```
Connected to 192.168.10.2.
220 cervi.virtual.es FTP server (Versión wu-2.5.0(1) Thu
Nov 18 22:52:47 MST 1999)
ready.
```

Una vez establecida la conexión el servidor nos pide que nos autentifiquemos como usuarios del sistema.

```
Name (192.168.10.2:raquel): raquel
331 Password required for raquel.
Password:
230 User raquel logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
```

Una vez hecho esto, el sistema esta preparado para procesar peticiones y nos muestra el símbolo de ftp.

```
ftp>
```

Se puede establecer una conexión pasiva donde es el cliente quien inicia la conexión. Esta opción es útil cuando intentamos conectar una maquina windows con un linux, por la diferencia antes comentada.

En nuestra distribución tenemos dos aplicaciones de ftp :

- El servidor, contenido en el paquete wu-ftpd, versión 2.5.0-4.
- El cliente, contenido en le paquete netkit-ftp, versión 0.10-7, que permite establecer conexiones pasivas.

2.1.5 Servidor de correo.

Existe una gran variedad de programas de correo electrónico que deben ser divididos en dos grupos: los llamados Agentes de Usuario o MUA (Mail User Agent) que viene a ser la interfaz y los Agentes de Transporte o MTA (Mail Transport Agent), que son los encargados de transferir los mails a su correcto destino.

Sendmail es el agente de transporte de correo más común de Internet (en los sistemas UNIX). Aparece como servidor de correo principal en nuestra distribución, aunque disponemos de otro, el procmail, que va a ser pasado por alto en esta explicación simplemente porque no lo utilizamos.

Aunque sendmail actúa principalmente como MTA, también puede ser utilizado como MUA (aunque no posee interfaz de usuario, lo que significa que es controlado desde consola). Las misiones básicas de sendmail son las siguientes:

- Recogida de mails provenientes de un Mail User Agent (MUA) como pueden ser elm, Eudora o pine; o provenientes de un Mail Transport Agent (MTA) como puede ser el propio sendmail.
- Elección de la estrategia de reparto de los mails, basándose en la información de la dirección del destinatario contenida en la cabecera.
    - Si el mail es local en nuestro sistema, enviará el mail al programa de reparto local de mails.
    - Si el mail no es local, sendmail utilizará el DNS de nuestro sistema para determinar el host al que debe ser enviado el mail. Para transferir el mensaje, iniciará una sesión SMTP con el MTA de dicho host.
- Si no es posible mandar el mail a su destino (porque la maquina receptora esta desconectada, o va muy lenta), sendmail almacenará los mails en una cola de correo, y volverá a intentar el envío del mail un tiempo después. Si el mail no puede ser enviado tras un tiempo razonable, el mail será devuelto a su autor con un mensaje de error. Sendmail debe garantizar que cada mensaje llegue correctamente a su destino, o si hay error este debe ser notificado (ningún mail debe perderse completamente).
- Reformatear el mail antes de pasarlo a la siguiente máquina, según unas reglas de reescritura. Según el tipo de conexión que poseamos con una determinada máquina, o según el agente de transporte al que vaya dirigido el mail, necesitaremos cambiar los formatos de las direcciones del remitente y del destinatario, algunas líneas de la cabecera del mail, o incluso puede que necesitemos añadir alguna línea a la cabecera. Sendmail debe realizar todas estas tareas para conseguir la máxima compatibilidad entre usuarios distintos.
- Otra función muy importante de sendmail es permitir el uso de "alias" entre los usuarios del sistema; lo que nos permitirá (entre otras funciones) crear y mantener listas de correo entre grupos.
- Ejecución como agente de usuario (MUA). Aunque no posee interfaz de usuario, sendmail también permite el envío directo de mails a través de su ejecutable.

El funcionamiento del sendmail queda reflejado en la figura 2.

Todas estas características y muchas otras que posee el sendmail deben ser configuradas y variarán de unos sistemas a otros. Para configurarlas hacemos uso del fichero de configuración de sendmail. La revisión y modificación de este fichero es bastante complicada y necesita de una serie de conocimientos previos.

Una explicación más detallada sobre estos datos puede encontrarse en [SEN].

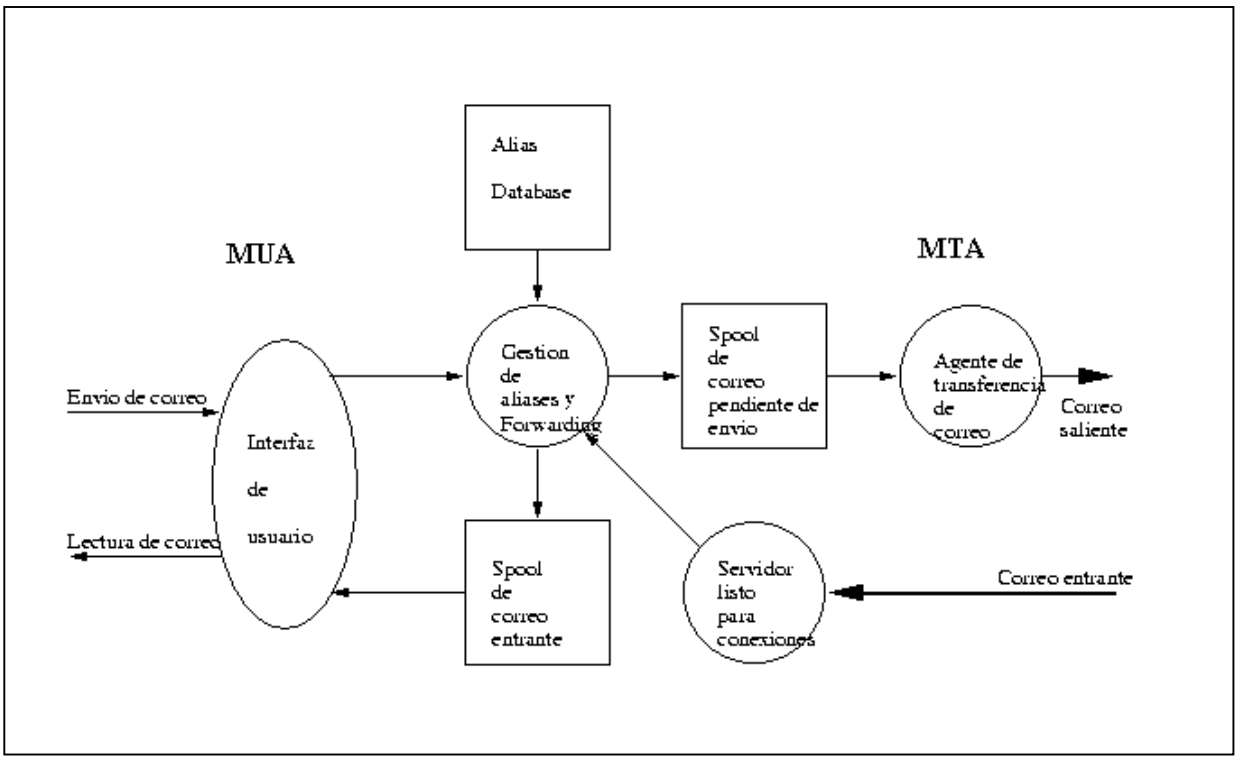


Figura 2.

2.2. La arquitectura de red en Cervi, S.A.

Para realizar la conexión con Cervi, S.A., utilizamos una IP pública seguida de unos puertos especiales, que mediante reglas de firewall nos redireccionan al servidor donde tenemos alojado el proyecto.

No se ha podido poner una limitación de IP ya que nosotros en casa no tenemos una IP fija, se ha optado por usar puertos con valores altos donde muchos rastreadores no entran y además el sistema pide autentificación antes de iniciar la sesión de trabajo. Esto implica que nuestro servidor de trabajo esta abierto al público por lo que se han tomado unas medidas especiales de seguridad.

Primero se han creado dos redes virtuales, como muestra la figura 3, que separan nuestro servidor del resto de la red interna de Cervi, S.A. Mediante instrucciones del firewall se ha restringuido el acceso de nuestro servidor a todas las máquinas excepto a dos:
el servidor de gestión de Cervi, S.A. y el ordenador personal que usamos para programar.
En este caso el sentido de la comunicación será siempre de Cervi, S.A. hacia fuera por lo que estos dos ordenadores no aceptaran peticiones de ftp o telnet que vengan de la subred 192.168.10.xxx por lo que se reducen los peligros si alguien no autorizado llega a conectarse.

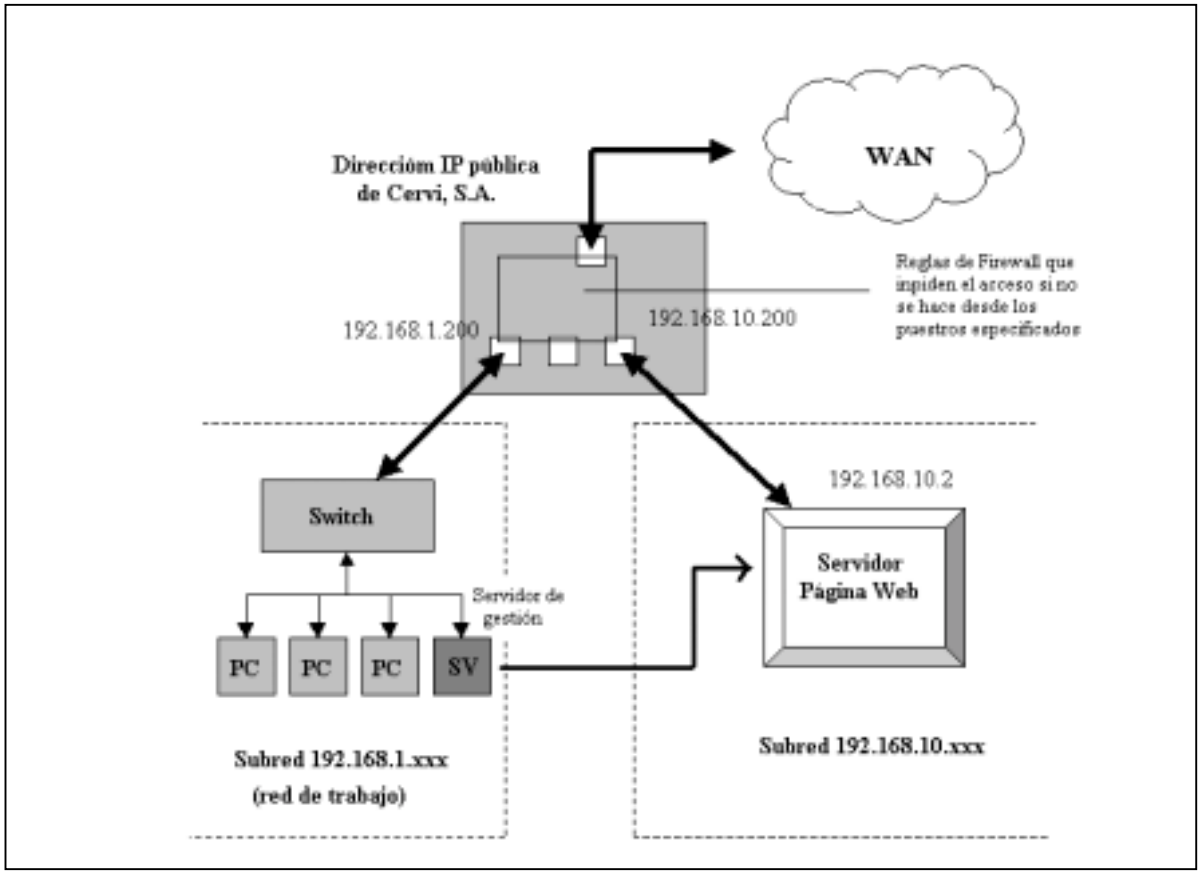


Figura 3

El hecho de aceptar o no peticiones se configura en un archivo especial donde se relacionan todos los permisos de acceso al sistema llamado hosts.allow (/etc/hosts.allow). En él se configuran los permisos para los diferentes protocolos. Nosotros únicamente hemos modificado los protocolos web, telnet y ftp.

Estas son las instrucciones de servidor de gestión de Cervi, S.A. para el protocolo telnet y ftp. Como se puede ver solamente están permitidas las direcciones de la subred 192.168.1.xxx.

```
# Telnet
in.telnetd: 80.33.181.18 80.59.217.79 localhost 192.168.1.0/255.255.255.0
: ALLOW
#in.telnetd: localhost 192.168.1.0/255.255.255.0 KNOWN: ALLOW
#in.telnetd: ALL: ALLOW

# FTP
in.ftpd: 80.33.181.18 80.59.217.79 localhost 192.168.1.0/255.255.255.0 :
ALLOW
#in.ftpd: localhost 192.168.1.0/255.255.255.0 KNOWN: ALLOW
#in.ftpd: ALL: ALLOW
```

(# en comentario, no son tenidas en cuenta por el sistema).

En el servidor donde alojamos la página web no hemos podido poner ninguna restricción a nivel de IP por lo que el archivo queda de la siguiente manera:

```
# Telnet
#in.telnetd: localhost 192.168.1.0/255.255.255.0 KNOWN: ALLOW
in.telnetd: ALL: ALLOW

# FTP
#in.ftpd: localhost 192.168.1.0/255.255.255.0 KNOWN: ALLOW
in.ftpd: ALL: ALLOW
```

(# en comentario, no son tenidas en cuenta por el sistema).

En este caso ALL hace referencias a todas las direcciones IP.

3. Lenguajes para el desarrollo web.

3.1 Php

PHP, acrónimo de Hipertext Preprocessor, es un leguaje scripting de lado servidor, es decir, que se ejecuta en el servidor, para crear paginas dinámicas (DHTML). El cliente solamente recibe una pagina con el código html resultante de la ejecución de PHP, por lo que es compatible con todos los navegadores.

3.1.1 Ventajas e inconvenientes de PHP.

Ventajas:

- Sencillo de aprender.
- Similar en sintaxis a C y a Perl.
- Soporta en cierta medida la orientación a objeto ya que contempla la creación de clases y herencias.
- Excelente soporte de acceso a bases de datos (MySQL, Access, Oracle, PostgreeSQL...).
- La comprobación de la validez de los parámetros para el funcionamiento de una pagina se hace en el servidor y no en el cliente (como se hace con javascript) de forma que se puedan evitar manipulaciones maliciosas.
- Es multiplataforma, funciona en todas las plataformas que soporten apache.
- Es software libre. Se puede obtener en la web y su código está disponible bajo la licencia GLP.

Desventajas:

- Todo el trabajo se realiza en servidor por lo que un numero elevado de petición puede relentizar el sistema.
- La legibilidad del código puede verse afectada al mezclar sentencias HTML y PHP.
- La orientación a objetos es aun muy deficiente para aplicaciones grandes aunque en versiones superiores se esta trabajando para eliminar este problema.

En [PHP] podemos encontrar un manual descargable de este lenguaje así como las últimas noticas y novedades.

3.1.2 PHP Vs ASP

ASP es el acrónimo de Active Server Pages y es una tecnología creada por Microsoft, lo que nos lleva a decir que no es gratuito (esta es la principal diferencia). Al igual que PHP, ASP es un lenguaje script encapsulado en html (consultar [ASP]).

Su funcionamiento es similar a PHP, se ejecuta en el servidor enviando al cliente solo código html.

Si hacemos una comparación de ambos lenguajes podemos establecer los siguientes puntos:

- ASP es más lento que PHP debido a que funciona con la arquitectura COM (Component Object Model, que es una especificación de programación orientada a objetos) lo que supone que deben establecerse conexiones entre diferentes objetos COM mientras que en PHP se ejecuta en el espacio de memoria
- La administración de la memoria que hace php es mas depurada, pongamos un ejemplo: si tenemos una cabecera .ASP que aparece en 20 paginas esta será almacenada 20 veces en memoria, mientras que con php sólo se cargan una vez todos los includes.
- En PHP no hay costes adicionales para nuevas utilidades puesto que es gratuito mientras que en ASP deben pagarse cada modulo por separado (ASPEncrypt para encriptar, QMail como servidor de correo ...), lo que encarece el coste del producto final.
- La sintaxis es similar a Java/C++ mientras que ASP ha seguido una evolución diferente.

Para resumir estos puntos podemos observar la siguiente tabla:

	PHP	**ASP**
Licencia	Libre distribución	Microsoft
Tipo de lenguaje	Scripting, embebido en HTML, ejecutado en el servidor	Scripting, embebido en HTML, ejecutado en el servidor
Plataformas en las que funciona	Windows, Linux/Unix, Solaris, (con el software servidor apropiado)	Windows 9x y Windows NT, y con el software también funciona en Linux/Unix (previo pago)
Arquitectura del lenguaje	Ejecución del programa en el mismo espacio de memoria	Programación orientada a objetos mediante la especificación COM
Gestión de memoria	Gestión más depurada que ASP	Algunos defectos en la gestión de memoria por solucionar
Servicios que soporta	Encriptado, ftp, e-mail,...	Los mismos, pero con módulos separados (que se pagan a parte)
Sintaxis	Sigue el modelo Java/C++	Sigue el modelo Visual Basic
Bases de datos que soporta	MySQL y otros SGBD con el estándar SQL, el controlador para conexión con las a incluido en el intérprete	Access y SGBD con el estándar SQL, requieren un controlador para la conexión. Son las bases de datos que se ha de instalar normalmente a mano (ODBC)

3.2 Standard SQL

Las siglas SQL significan *Standard Query Language*. Fue desarrollado por el departamento de investigación de IBM en los años 70 a partir de una interficie (*SEQUEL, Structured Engish QUEry Language*) para el sistema de bases de datos experimental SYSTEM dada la necesidad de un gestor standard para bases de datos.

El lenguaje ha sido aceptado por ANSI (*American Standard National Institute*) y la última versión standard es la del 1999. Aún así, existen variantes en función de los diferentes gestores de bases de datos.

Es un lenguaje declarativo, es decir, que no se establecen procedimientos, únicamente se ejecutan ordenes en el servidor, por este motivo también suelen calificarlo como un lenguaje imperativo.

El lenguaje SQL se divide en tres partes:

- **DML** (Data Manipulation Language): **lenguaje de manipulación** de datos que capacita a los usuarios a acceder o manipular datos según estén organizados por el modelo de datos.

  Por manipulación se entiende:
  - recuperar datos (query)
  - insertar datos (insert)
  - suprimir datos (delete)
  - modificar datos (update)

  Procedural - el usuario especifica los datos que necesita y cómo obtenerlos.
  No procedural - el usuario especifica los datos que necesita sin especificar cómo obtenerlos.

- **DDL** (Data Definition Language): **lenguaje de definición** de datos que por medio de un conjunto de definiciones permite especificar un esquema de base de datos, es decir, crear, modificar o borrar tablas y crear, modificar o borrar bases de datos.

  El DDL contiene definiciones de tipo especial respecto a la estructura de almacenamiento y los métodos de acceso usados por los sistemas de bases de datos.

  El resultado de la compilación de estas definiciones es un conjunto de instrucciones que especifican los detalles de implantación de los esquemas que se le esconden al usuario.

- **DCL** (Data Control Language), **lenguaje de control** de datos que contiene elementos útiles para trabajar en un entorno multiusuario, en el que es importante la protección de los datos, la seguridad de las tablas y el establecimiento de restricciones en el acceso, así como elementos para coordinar la compartición de datos por parte de usuarios concurrentes, asegurando que no interfieren unos con otros.

Los requerimientos de sintaxi y forma pueden consultarse en [SQL].

3.3. Estandars para el desarrollo web

3.3.1 Estandars de html

El lenguaje HTML fue desarrollado originalmente por Tim Berners-Lee mientras estaba en el CERN, y fue popularizado por el navegador Mosaic desarrollado en el NCSA. Durante los años 90 ha proliferado con el crecimiento explosivo de la Web. Durante este tiempo, el HTML se ha desarrollado de diferentes maneras. La navegabilidad depende de que los autores de páginas Web y las compañías que diseñan e implementan los navegadores compartan las mismas convenciones de HTML. Esto ha motivado el trabajo colectivo en las especificaciones del HTML.

La versión HTML 2.0 (aparecida en noviembre de 1995) fue desarrollada bajo los auspicios de la Internet Engineering Task Force (IETF) para codificar lo que era la práctica común a finales de 1994. Posteriormente se propusieron versiones mucho más ricas de HTML. A pesar de no haber logrado nunca el consenso en las discusiones sobre estándares, estos borradores llevaron a la adopción de un número de nuevas características.

La mayoría de las personas están de acuerdo en que los documentos HTML deberían funcionar bien en diferentes navegadores y plataformas. Gracias a la interoperabilidad los proveedores de contenidos reducen gastos, ya que sólo deben desarrollar una versión de cada documento. Si este esfuerzo no se realiza, hay un riesgo mucho mayor de que la Web se convierta en un mundo propietario de formatos incompatibles, que al final acabaría por reducir el potencial comercial de la Web para todos los que forman parte de ella.

Cada versión de HTML ha intentado reflejar un consenso cada vez mayor entre los interlocutores de la industria, de modo que no se desperdicien las inversiones hechas por los proveedores de contenidos y que sus documentos no dejen de ser legibles a corto plazo.

El lenguaje HTML ha sido desarrollado con la premisa de que cualquier tipo de dispositivo debería ser capaz de usar la información de la Web: PCs con pantallas gráficas con distintas resoluciones y colores, teléfonos móviles, dispositivos de mano, dispositivos de salida y entrada por voz, computadoras con anchos de banda grandes o pequeños, etc.

3.3.1.1. HTML 4 y HTML 4.01.

La versión HTML 4 desarrolla el lenguaje HTML con mecanismos para hojas de estilo, ejecución de scripts, marcos, objetos incluidos, soporte mejorado para texto de derecha a izquierda y direcciones mezcladas, tablas más ricas y mejoras en formularios, ofreciendo mejoras de accesibilidad para personas con discapacidades.

HTML 4.01 es una revisión de HTML 4.0 que corrige errores e introduce algunos cambios desde la versión anterior.

El estandard HTML 4.01 es el estandar al que pendemos ceñirnos durante la reacción de las páginas web, aunque habrá momentos donde deberemos dejar los estandars a un lado y optar por soluciones específicas para ciertos navegadores, generalmente para Internet Explorer de Microsoft.

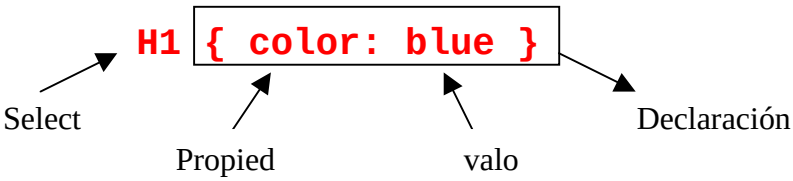
En [HTML4] pueden consultarse las especificaciones estandars sobre HTML 4.0 en diferentes formatos y idiomas.

3.3.2 Estandars de Css: CSS2

CSS2 es un lenguaje de hoja de estilo que permite que los autores y los usuarios asocien un estilo (por ejemplo, fuentes, espaciado y señales sonoras) a los documentos estructurados (por ejemplo, documentos HTML y aplicaciones XML). Separando el estilo de presentación del contenido de los documentos, CSS2 simplifica la creación y mantenimiento de los sitios Web.

CSS2 se cimenta en CSS1 (el estandard anterior) y, con muy pocas excepciones, todas las hojas de estilo CSS1 válidas son hojas de estilo CSS2 válidas. CSS2 brinda soporte a hojas de estilo específicas para cada medio, de modo que los autores puedan adaptar la presentación de sus documentos a los browsers visuales, a los dispositivos sonoros, a las impresoras, a los dispositivos braille, de mano, etc. Esta especificación también soporta el posicionamiento de contenidos, fuentes descargables, disposición de la página, aspectos para la internacionalización, contadores y numeradores automáticos, y algunas características relacionadas con la interfaz del usuario.

Una regla CSS consta de dos partes principales: un select (‘H1’) y una declaración ('color: blue'). La declaración tiene dos partes: una propiedad ('color') y un valor ('blue'). Con tal que el ejemplo anterior intente influir en una sola propiedad necesaria para el procesamiento de un documento HTML, ya lo califica como una hoja de estilo en sí mismo. Combinado con otras hojas de estilo (un rasgo fundamental de CSS es que las hojas de estilo se combinan) determinará la presentación final del documento.



La principal característica de las hojas de estilo es la posibilidad de heredar reglas de otras hojas. Mediante la regla @import permite importar hojas de estilo desde otras hojas de estilo como si de librerías se tratara.

En [CSS2] pueden consultarse las especificaciones establecidas por el World Web Consortium.

3.4. Estandares de Javascript

Los estandars actualmente reconocido son los estandares de ECMA, cuyo estandard Ecma-262 se basa en la primera versión que desarrollo Netscape para su Navigator desde la versión 2.0 (al la que llamó Javascript 1.0) y la que desarrollo posteriormente Microsoft para Internet Explorer 3.0 (a la que llamó Jscript), por lo tanto este estandard es originario de las dos tecnologías.

Su desarrollo comenzó en noviembre de 1996, y la primera versión fue aprobada por la asamblea general de Ecma en junio de 1997. En abril de 1998 fue reconocido por el estandar internacional ISO/IEC 16262.

Javascript es un lenguaje de programación orientado a objeto aunque el funcionamiento de estos objetos dista mucho de ser el funcionamiento de los objetos en JAVA.

Es un lenguaje interpretado, es decir, no requiere compilación. El navegador del usuario se encarga de interpretar las sentencias JavaScript contenidas en una página HTML y ejecutarlas adecuadamente.

Generalmente el lenguaje Script actúa como interficie entre el usuario y el sistema, ya que es capaz de obtener información del sistema (la hora actual, el nombre y la versión del navegador...) para poder manipularla y ofrecérsela al usuario.

Dentro del lenguaje script tenemos dos ramas bien diferenciadas:

- El código script que se ejecuta en el cliente (cliente-side), que es el que nosotros hemos usado para desarrollar la aplicación.
- El código script que se ejecuta en el servidor (server-side), que nosotros hemos cambiado por php debido a su complejidad.

Dentro de la primera rama podríamos incluir el web script que incluye los objetos y eventos necesarios para trabajar con formatos, inputs, textareas, documents, etc...

Un script, almacenados generalmente en archivos con la extensión js o incrustados en una pagina html, es un programa que puede acompañar un documento HTML o estar contenido en su interior. Las instrucciones del programa se ejecutan cuando se carga el documento o cuando se produce alguna circunstancia tal como la activación de un enlace por parte del usuario. Este tipo de acciones desencadenan lo que se conocen como eventos.

Los scripts permiten a los creadores de páginas web realzar el contenido de sus documentos. Algunas mejoras pueden ser las siguientes:

- Generación dinámica del documento en el momento de la carga.
- Validación de datos introducidos en un formulario, así como el rellenado automático de aquellos controles del mismo que lo precisen.
- Controlar los eventos que se producen en la página: un elemento toma el foco, un elemento es activado con el ratón, etc.

3.4.1. Problemas entre versiones

El hecho de que dos plataformas propietarias, es decir, que patentan su software y en el caso de Microsoft que es de pago, ha generado un sin fín de incompatibilidades entre ambos navegadores.

También se ha dado el hecho que a medida que se han ido desarrollando diferentes versiones de javascript (actualmente tanto Microsoft como Netscape usan Javascript 1.3 que se adhiere al estandard Ecma) se han generado incompatibilidades entre diferentes versiones del mismo navegador.

La siguiente tabla nos relaciona la versión de Javascript que utilizaba cada navegador según la versión de este:

Navegador	Versión de JavaScript
NAV 2.0	1.0
NAV 3.0	1.1
NAV 4.0 - 4.05	1.2
NAV 4.06 o superior	1.3 (aproximadamente sigue el estándar ECMA)
MIE 3.0	JScript (aproximadamente JavaScript 1.0, pero con muchos errores)
MIE 4.0 o superior	Estándar ECMA

BLOQUE II

Creación de la base de datos

4. CONCEPTOS BÁSICOS DE BASES DE DATOS.

El mercado de MBD (manejadores de bases de datos) es bastante grande y nos ofrece demasiadas alternativas a la hora de elegir un software en que confiar para desarrollar nuestro proyecto, ¿Por cual herramienta inclinarnos? ¿Cuál es la óptima en nuestro desarrollo específico? ¿Porqué? ¿Qué detalles de implementación debemos tener en cuenta para elegir nuestra herramienta MBD?

En un proyecto de ingeniería de software las herramientas de trabajo constituyen un aspecto de implementación fundamental, las características propias del proyecto hacen necesario que estas herramientas se ajusten adecuadamente a nuestro plan de desarrollo.

Antes de las bases de datos que hoy conocemos, los programadores utilizaban ficheros secuenciales como almacenes de datos. Estos daban un acceso muy rápido pero sólo de forma secuencial (para acceder a una posición, se tenía que recorrer el fichero entero). Más tarde aparecieron los ficheros indexados, donde el acceso ya podía ser aleatorio (acceder de una vez a la posición deseada del fichero).

El sistema de ficheros era el sistema más común de almacenamiento de datos. Para compartir los datos entre varias máquinas surgió el **NFS** (Network File System), y más tarde para evitar fallos en los sistemas de fichero aparecieron los sistemas **RAID**.

Pero los programas y datos cada vez eran más complejos y grandes por tal motivo se requería de un almacenamiento que garantizara un cierto número de condiciones y que permitiera operaciones complejas sin que se violaran estas restricciones. Además cada usuario que accediera a los datos debía tener su trabajo protegido de las operaciones que hicieran el resto de usuarios.

Respondiendo a estas necesidades, surgieron las bases de datos jerárquicas donde los datos se situaban siguiendo una jerarquía. Hoy en día, la mayoría de entidades bancarias aun funcionan con este tipo de base de datos, ya que no necesitan nada más: un cliente tiene X cuentas bancarias, una cuenta bancaria tiene X tarjetas, etc.

Las bases de datos jerárquicas tenían el problema que los accesos a los datos eran unidireccionales, y era más complicado hacer el camino inverso (pero posible, aunque el tiempo de cálculo era mayor). Por ejemplo, era fácil saber que cuentas tenía un cliente, pero no tanto saber de qué cliente era una cierta cuenta.

Para dar absoluta libertad a las relaciones entre tablas surgieron las bases de datos relacionales (**RDBMS**).

Las RDBMS trajeron dos cosas muy importantes: las propiedades ACID y un lenguaje común de acceso a los datos: SQL.

Mediante SQL, por primera vez, decías QUE datos querías y no COMO los tenías que sacar. Apareció el término de transacción: Agrupación de instrucciones SQL (por ejemplo varios Selects, Inserts y Updates).

Las propiedades **ACID** son:

- **Atomicidad:** Cada transacción del usuario debe tratarse de forma atómica. O se ejecuta todo o nada.

- **Consistencia:** Las transacciones han de cumplir las restricciones definidas dentro la base de datos. Si no las pueden cumplir, se evita su ejecución. De esta forma se conserva la integridad y coherencia de los datos.

- **Aislamiento:** (Isolation) Los resultados de una transacción son inaccesibles a las otras transacciones cuando no está terminada.

- **Durabilidad:** Una vez se ha completado la transacción, los resultados de la misma han de ser permanentes y sobrevivir a posibles caídas del sistema o la base de datos.

Debido a que las RDBMS tienen que soportar todas estas propiedades, nunca serán tan rápidas como trabajar directamente sobre ficheros, aunque internamente trabajen sobre ellos. La mayoría de desarrolladores prefieren hoy en día sacrificar la velocidad por las funcionalidades.

5. Características de los diferentes gestores de bases de datos existentes en el mercado.

El software de bases de datos ha experimentado un auge extraordinario a raíz de la progresiva informatización de casi la totalidad de las empresas de hoy día. Rapidez y efectividad en los procesos y los grandes flujos de información son las necesidades más apremiantes a la hora de optimizar servicios y productos. Ante esta notable demanda de soluciones informáticas han surgido multitud de gestores de bases de datos, estos son programas que permiten manejar la información de modo sencillo y que prestan servicios para el desarrollo y el manejo de bases de datos.

Por este motivo vamos a analizar los productos más destacados y de acuerdo a nuestras necesidades inclinarnos por la opción que mejor nos convenga.

Realizaremos una comparativa técnica de las diferentes alternativas existentes en el mercado de manejadores de bases de datos, daremos la descripción técnica y mostraremos las características de rendimiento que nos ofrecen las casas desarrolladoras de manejadores de bases de datos (MDB). Por último investigaremos las ventajas o desventajas que nos proporcionan los MBD propietarios y no propietarios.

La información que mostramos a continuación es el resultado de una intensa búsqueda en internet. Las diferentes páginas que visitamos aparecen relacionadas en la bibliografía.

Cabe destacar que nuestra decisión estará condicionada por dos factores:

a) Las características del proyecto nos aconsejan trabajar con software libre.

b) La empresa para la cual desarrollamos el WebSite ya trabaja con un servidor Mysql, por lo que condicionará de manera importante nuestra decisión. Además cabe la posibilidad de que el servidor que haga de hosting final para la página trabaje en una condiciones muy similares a las del nuestro, es decir, con una configuración linux, un servidor web Apache y un gestor de bases de datos Mysql.

Esto no descarta que las necesidades del proyecto nos obliguen a reconsiderar otros manejadores cuyas características se adapten mejor a las necesidades de la empresa.

5.1 MYSQL 3.XX.XX

Es seguramente la base de datos para Linux más popular de todas, por lo que viene incluida en casi todas las distribuciones de Linux. Además está disponible para casi todas las plataformas hardware y sistemas operativos (incluidos Windows NT/2000,98/95/ME).

MySQL es conocida sobretodo por su velocidad y escasos recursos que consume.

Es muy fácil de instalar y administrar y tiene una gran comunidad de usuarios. Buscar información para realizar cualquier cosa con MySQL es un muy fácil. Es muy indicada para iniciarse en el mundo de las bases de datos, puesto que dispone de una infinidad de utilidades, tutoriales y documentación que la inmensa comunidad de usuarios de MySQL se ha encargado de realizar desinteresadamente.

El gran propulsor de la base de datos MySQL ha sido sin duda el lenguaje interpretado para Web PHP. MySQL ha crecido al mismo paso que ha crecido la comunidad de PHP. Cuando llegó la popularización del Linux y de PHP como plataforma para aplicaciones Web de bajo coste se encontró que la Única base de datos gratuita robusta del momento era MySQL (PostgreSQL 6.X era lento y pesado). Hoy muchas Webs populares usan MySQL, como Slashdot o Freshmeat.

Tiene también contribuciones para añadir soporte de Full-text-indexing en las tablas, un valor añadido importante hoy en día.

Una desventaja de MySQL es que no cumple todas las propiedades ACID de las RDBMS. No tiene integridad referencial, con lo que la base de datos puede llegar a tener datos inconsistentes (por ejemplo empleados pertenecientes a departamentos inexistentes). En el CREATE TABLE aunque le pongas FOREIGN KEY, lo ignorará por completo. Carece también de transacciones incumpliendo otra vez las ACID.

MySQL guarda los Blobs (binarios) en la misma tabla que guarda los datos normales, o sea como un campo más en la misma fila. No es el mejor método para guardar un binario, ya que hace caer el rendimiento del acceso a los otros datos.

Según [MYS], en las próximas versiones quieren incluir estas propiedades y otras de las que carecen, como los triggers o los procedimientos almacenados.

5.2. PostgreSQL

PostgreSQL se diseñó como una base de datos orientada a objetos, es decir, una ORDBMS. Esto significa, que las tablas no son tablas, sino objetos, y las tuplas son instancias de ese objeto. Puedes crear nuevos tipos de datos y hacer herencias entre objetos.

PostgreSQL tiene todo de lo que carece MySQL, transacciones, integridad referencial (claves foráneas), vistas, y multitud de funcionalidades, pero es lento y pesado.

Hoy con la liberalización de las bases de datos comerciales y la entrada de los grandes en el mundo Linux el panorama va cambiando. Con la aparición de las versiones 7.X, los de PostgreSQL argumentaron que empezaba una nueva era: más rápido, más fiable, etc. En la práctica continúan más o menos igual.

Han incorporado la llamada **MVCC** (multiversión concurrency control) con lo que los bloqueos de escritura actúan sólo en la sesión del cliente, no en las de los demás clientes. También tiene soporte de Full-Text-indexing a través de un trigger incluido en la distribución.

También han arreglado el límite de 8k por fila. Por fin es de 32k (hasta la 7.1 no lo han arreglado).

Postgres usa un modelo cliente-servidor conocido como proceso por usuario una sesión de postgres consiste en los siguientes procesos cooperativos de Unix (programas):

- Un proceso demonio supervisor (postmaster).
- La aplicación sobre la que trabaja el usuario (frontend, ej: psl).
- Uno o más servidores de bases de datos en segundo plano (el mismo proceso Postgre).

5.3 INTERBASE 6.0

Borland siguiendo la política de moda de liberalizar las herramientas de desarrollo, liberalizó Interbase. Esta es una buena base de datos con 16 años de experiencia en el sector de las bases de datos comerciales.

Interbase ha sido la primera base de datos Open Source en ser compatible SQL92 a nivel de entrada (MySQL, PostgreSQL, mSQL, no lo son). Eso significa que puedes adaptar cualquier aplicación que funcione sobre otra base de datos compatible SQL92 como Oracle, DB2 o Informix y traspasarla fácilmente a Interbase, o al revés: empezar con Interbase al principio y pasarla si algún día hace falta a alguna RDBMS mayor. Eso tan simple e importante, es imposible hacerlo con MySQL o PostgreSQL.

Interbase tiene la mayoría de funcionalidades de una base de datos comercial: triggers, tratamiento especial de blobs, backup On-line, gran escalabilidad, bases de datos de solo lectura (para ponerlas en CD-Rom), integridad referencial en cascada o el autotunning.

La desventaja que tiene el Interbase frente a las dos anteriores es la poca comunidad de la que dispone de momento. Hay poca información aun en la red. Si Borland quiere que Interbase cuaje en el mundo Linux tendrá que abrir la Web a las comunidades de desarrolladores.

5.4 SAP DB 7.2

SAP DB es la base de datos desarrollada inicialmente para soportar el monstruo ERP llamado SAP R/3. Las Últimas versiones de SAP R/3 contienen alrededor de 10.000 tablas.

Con esto deducimos que es una base de datos preparada para mover grandes cantidades de datos.

Es sin duda la más difícil de administrar y instalar de las nombradas en este punto, aunque quizás la más profesional.

La instalación es muy complicada por la poca información que se consigue. No vienen ni scripts para arranque automático y es necesario generar scripts para crear nuevas bases de datos.

SAP DB es una muy buena base de datos, pero pesada y no muy moderna. Tiene integridad referencial, triggers, vistas, backup online (soportado por la mayoría de herramientas comerciales), pero carece de tratamiento de blobs o indexación de texto. Es una base de datos para guardar ingentes cantidades de texto y números, pero no multimedia.

Lo peor que tiene SAP DB es la complejidad de administración, unida con la poca comunidad que hay que la usa. En la web oficial, apenas pone información y instrucciones.

PHP no tiene soporte directo de SAP DB. En el sitio de SAP existe un patch y un micro HOWTO para dar soporte a SAP DB vía el Unified **ODBC** del PHP. El soporte ODBC nunca es tan rápido como el soporte directo ni soporta las características especiales de cada base de datos, como el tratamiento de blobs.

5.5 MICROSOFT SQL SERVER

Se podría decir que en cierta medida pretende ser el servidor de bases de datos genérico para Windows. No tanto por que la casa que desarrolla sea la misma, ni siquiera porque el SQL Server, a diferencia de otros servidores solo trabaja bajo Windows, sino porque Microsoft promete integración con todos los productos suyos (por ejemplo MsOffice 2000, ya que Access 2000 traerá consigo un nuevo MSDE-DATA-Engine, como alternativa al existente y compatible con SQL Server). También será posible llamar a SQL Server desde Ms-Access.

La escalabilidad es total. No es necesario decir que el producto puede funcionar en un Servidor NT multiprocesador de elevadas prestaciones, pero si lo es decir que también puede funcionar en un portátil con Windows 95/98. Las características de SQL Server son impresionantes. Tenemos soporte de transacciones OLPT, una maquinaria de búsqueda de textos completos que permite localizar información a lo largo de una tabla (lo que hace las delicias si el proyecto a considerar es para Internet). Posibilidad de ejecutar consultas en paralelo, así como homogéneas y distribuidas. Bloqueos dinámicos a nivel de filas, optimizador de consultas, estadísticas Automáticas, Unicode Nativo, Replicación Avanzada, Replicación dinámica de datos y un largo etcétera. Pero lo mejor de todo es la sencillez de comprensión y navegación entre procesos así como lo intuitivo de la herramienta.

Los protocolos de red soportados con garantía d funcionamiento son TCP/IP, Pipes con nombre, IPX/SPX, AppleTalk ADSP y Banyanvines.

5.6 ORACLE

A pesar de llevar ya algún tiempo en el mercado la versión 8 de Oracle, sigue junto con SQL Server, liderando el mercado NT. Además de Windows, el servidor de Oracle puede funcionar en una gran cantidad de sistemas operativos y diversidad de Hardware. Prácticamente tenemos a todas las familias de UNIX, MVS, VM, HPMPEXL, Siemens ICL, Novell Netware y OS/2.

Atendiendo a las características de manejabilidad escalabilidad rendimiento y soporte entre plataformas tenemos una arquitectura de servidor con ejecución multihilo y rendimiento de multiprocesadores simétricos (SMP). En cuanto a los bloqueos se refiere, los admite a nivel de filas sin restricciones. Las bases de datos pueden crecer hasta límites que en la practica pueden considerarse inalcanzables y gracias a un rediseño respecto alas versiones se han eliminado ciertos cuellos de botella.

Las características más relevantes en lo referente a la escalabilidad residen en las tablas particionadas que son aquellas que pueden dividirse en múltiples dispositivos de almacenamiento de acuerdo con valores preestablecidos y el multiplexado y *pooling* de conexiones. Precisamente la gestión de conexiones se realiza mediante cierto software específico. Concretamente el de red, que antes se llamaba SQL NET, ahora se denomina Net, que a parte de cambiar de nombre ha avanzado tanto en las características (interface de programación, navegador de Internet con soporte Java y soporte de red adicionales) y rendimiento, como en facilidad.

Los Protocolos de Red Soportados Por Oracle son los siguientes: Net 8, TCP/IP, IPX/SPX, Pipes con nombre, DECNET, DCE, NDS y LU 6.2 (APPC).

Entre las características de SQL podemos destacar una optimización independiente de la sintaxis, unas consultas y una generalización de estadísticas de tablas (ANALIZE). Pero lo que realmente incrementa drásticamente el rendimiento es el soporte de referencia a objetos que son rápidos punteros de tablas los cuales sustituyen a las letras y traicionares unidades relacionales.

5.7 Compativas

5.7.1 Comparativa de características

En la siguiente tabla podemos ver un resumen de las características técnicas que cumplen algunos de los RDBMS vistos. A continuación daremos una breve definición de las mas destacadas de las mismas.

Comparativa de Características Técnicas

	MySQL 3.23.37	**PostgreSQL 7.1**	**Interbase 6.0**	**SAPDB**
Cumple SQL 92	N	N	S	S
Integridad Referencial	S	S	S	S
Procedimientos Almacenados	N	S	S	S
Vistas (views)	N	S	S	S
Bases de datos de sólo lectura	N	N	S	N
Secuencias (generadores de números)	S	S	S	S
Subselects	N	S	S	S
Concurrencia Multi Versión	N	S	N	S (módulo aparte)
Soporte Perl, C, Python, TCL, Java, Delphi	S	S	S	S (módulo aparte)
Creación de nuevos tipos de datos	N	S	N	N (módulo aparte)
Triggers	N	S	S	S (módulo aparte)
Transacciones	N	S	S	S (módulo aparte)
Orientación a objetos	N	S	N	N (módulo aparte)
Limite tamaño registro	N	S	S	S (módulo aparte)
Soporte nativo PHP	S	S	S	N (módulo aparte)
Herencia	N	S	N	N (módulo aparte)
Multithread	S	N	S	S (módulo aparte)

- **SQL-92** Lenguaje SQL estandarizado por ANSI e ISO. Estandariza muchos de los puntos que anteriormente se dejaba al criterio de los fabricantes.
- **Una subconsulta** es una sentencia SELECT que aparece dentro de otra sentencia **SELECT** que llamaremos consulta principal. Se puede encontrar en la lista de selección, en la cláusula WHERE o en la cláusula HAVING de la consulta principal.
- La **integridad referencial** es un sistema de reglas que utilizan la mayoría de las bases de datos relacionales para asegurarse que los registros de tablas relacionadas son válidos y que no se borren o cambien datos relacionados de forma accidental produciendo errores de integridad.
- Los **Triggers,** también llamados "disparadores", permiten realizar acciones automáticas cuando se agregan/modifican/borran registros de una tabla.
- Los **Procedimientos,** también llamados **Procedimientos Almacenados**, o Stored Procedures, son conjuntos de instrucciones que pueden ejecutar acciones sobre los datos y también regresar resultados, y que pueden ser llamados desde un Trigger, o desde cualquier programa cliente.
- **Las secuencias** son **generadores de números** secuenciales utilizados como valor único para un determinado atributo de una relación. Sin secuencias, el proceso normal de generación de estos enteros conlleva un bloqueo manual en acceso para actualización de la tabla que los contiene. Con este mecanismo, las estructuras se bloquean justo en el momento de la actualización.
- **La Concurrencia multiversión** permite que mientras se consulta una base de datos, cada transacción vea una imagen de los datos (una *versión de la base de datos*) como si fuera tiempo atrás, sin tener en cuenta el estado actual de los datos que hay por debajo. Esto evita que la transacción vea datos inconsistentes que pueden ser causados por la actualización de otra transacción concurrente en la misma fila de datos, proporcionando *aislamiento transaccional* para cada sesión de la base de datos.
- **Multi-hebras de procesamientos (multithreaded):** Algunos motores de base de datos confían en múltiples aplicaciones para realizar su trabajo. En este tipo de arquitectura, cada vez que un usuario se conecta a la base de datos, ésta inicia una nueva instancia de la aplicación de base de datos. Con el fin de coordinar a muchos usuarios que accedan a los mismos conjuntos de datos estos ejecutables trabajan con un coordinador global de tareas que planifica operaciones para todos los usuarios.
- Completo soporte para **transacciones**. Una transacción está formada por un conjunto de acciones de forma que o se ejecutan todas ellas o bien ninguna. Utilizando transacciones aseguramos la consistencia de los datos.
- **Vista** es una relación *virtual*, que se construye a partir de tablas base o incluso otras vistas, formada por atributos de estas otras tablas de forma directa o como resultado de una consulta.
- **Base de datos orientada a objetos.-** Base de datos que contiene tipos de datos abstractos (objetos). Puede almacenar objetos directamente desde un lenguaje de programación orientado a objetos. Como no se puede almacenar ningún tipo de objeto (las reglas para almacenar los datos forman parte de un objeto), las bases de datos orientadas a objetos promete bases de datos completamente integradas

que contendrán datos, textos, imágenes y voz; esencialmente una variedad infinita de formatos en continuo cambio.

Hay otras características que no suelen aparecer en ninguna documentación y que sólo se aprenden mediante la experiencia. Son los temas referentes a la facilidad de administración, backups, instalación, etc. En cualquier caso, son difíciles de medir y varían ampliamente las opiniones entre los autores.

5.7.2.Comparativa de rendimientos

Los siguientes resultados han sido obtenidos de diversas páginas que aparecen en la bibliografía. Como es lógico nosotros no hemos realizado estas pruebas , ya que no disponemos de todos los RDBMS comparados.

Para comparar los rendimientos han preparado una serie de tests donde poder controlar los tiempos que tardaba cada base de datos en pasarlos.

La máquina de test es un Pentium III 800Mhz con 256Mb RAM y un disco IDE de 20Gb.

Para comprobar los rendimientos atacan las bases de datos desde dos puntos: desde una entrada de un fichero batch desde línea de comandos, y desde una apache con soporte PHP para generación de páginas dinámicas.

Las tareas preparatorias fueron las siguientes:

Crearon una base de datos llamada Empresas en todas las RDBMS. Crearon en todas las bases de datos las siguientes tablas:

```
Create table departamentos (
     Id integer primary key,
     Nombre varchar(10)
);

create table empleados (
     Id integer primary key,
     Nombre varchar(30),
     Nif varchar(9),
     Departamento integer,
     Foreign  key  FK_EMP_DEP  (ID)  references
departamentos(Id)
);
```

Instalaron PHP 4.0.5 como módulo en el Apache con soporte para las cuatro bases de datos.

Ataque desde línea de Comandos

Atacaron las RDBMS redireccionando 3 ficheros de entrada a los clientes de línea de comandos. Los ficheros son:

- Fichero con 100 inserts de departamentos y 1000 empleados por cada departamento. Total: 100.100 Inserts.
- Simularon un entorno normal de producción: 70% de Selects, 10% inserts, 10% updates, 10% deletes. Generaron un fichero con 100.000 instrucciones repartidas en esos porcentajes. Los selects son variados con Max(), Count(), joins, pero con resultados cortos.
- Simularon un entorno internet: 99% selects, 0.3% insert, s 0.3% updates, 0.3% deletes. Los selects son complejos y con resultados largos. Total de 1.000 instrucciones.

Pruebas de Acceso a Datos

	MySQL 3.23.37	**PostgreSQL 7.1**	**Interbase 6.2**	**SAP DB 7.2**
Inserción 100 mil tuplas	10 seg.	12min 38seg	42seg	2min 38seg
100.000 variados	17seg	6min 30seg	1min 07seg	8min
1000 selects complejos	55seg	1min 26seg	2min 23seg	1min 14seg

Debemos tener en cuenta a la hora de analizar estos resultados que Mysql no realiza comprobaciones para mantener la integridad referencial, por este motivo **MySQL** gana siempre en velocidad. Es la ventaja de no tener que comprobar que por cada empleado, existe el departamento.

PostgreSQL pierde mucho en las inserciones debido a que guarda básicamente cada escritura que hace. En las pruebas con PostgreSQL, se puede ver como el disco no para de escribir, mientras que con las otras, va escribiendo a ratos. PostgreSQL también tiene un modo Bflush(usar buffer) pero no recomiendan usarlo. En cuanto a los selects complejos los trata muy bien.

Interbase da tiempo moderados en todos los campos sin destacar en ninguno.

SAP DB como era de esperar pierde en los inserts, pero gana en los selects complejos, aunque se muestra lento en un entorno variado con selects cortos.

Ataque desde PHP

Con este test se pretende ver la capacidad para servir páginas dinámicas largas de una combinación Apache+PHP+base de datos. Las versiones usadas son: PHP 4.0.5 y Apache 1.3.12 y el PHP está compilado como módulo dinámico.

Generaron dos páginas para cada RDBMS a analizar. Una utilizando la apertura de conexión (MySQL_connect, pg_connect, ibase_connect, y odbc_connect) normal y otra con la apertura de conexión persistente (MySQLp_connect, ibasep_connect, y odbcp_connect). El propósito era ver si realmente incrementaba el rendimiento el hecho de usar conexiones persistentes o no.

Las páginas de pruebas hacen lo siguiente:

- Generar un número aleatorio entre 1 y 100.
- Obtener y mostrar el departamento que corresponde a ese número.
- Obtener y mostrar todos los empleados (1000) que correspondan a ese departamento, y los del departamento 10 (otros 1000).

Cada página resultante ocupa aproximadamente 240Kb y aprovecha los 100.000 empleados y 100 departamentos insertados previámente.

Las pruebas las realizaron siempre con 1000 peticiones y sólo variaron el número de usuarios concurrentes para poder ver la escalabilidad del RDBMS analizado. Hay que tener en cuenta que SAP DB tiene acceso por Unified ODBC mientras que los otros RDBMS tienen soporte nativo en el PHP, con la caída de rendimiento que supone para el SAP DB.

Segundos en completar las 1000 peticiones

	MySQL	**PostgreSQL**	**Interbase**	**SAP DB**
10p	312,66	752,74	225,30	305,31
10	325,22	943,40	212,42	300,03
20p	849,47	813,69	262,76	549,99
20	947,91	974,51	291,12	612,06
30p	1153,88	1620,00	377,75	763,76
30	1199,01	1468,79	408,23	806,18
40p	Timeout	Error	681,38	timeout
40	Timeout	Error	683,29	timeout
50p	Timeout	Error	timeout	timeout
50	Timeout	Error	timeout	timeout

Los tests empiezan con 10 clientes concurrentes hasta los 50, incrementando de 10 en 10. Los resultados se dan en tres variables: tiempo en resolver todas las peticiones, páginas por segundo y velocidad de transferencia en Kb/seg. En las tablas, un 20 significa un test con 20 peticiones concurrentes de páginas normales, un 20p significa un test con 20 peticiones con conexión persistente.

Cuando las bases de datos se colapsan (por falta de memoria, CPU, disco, etc.), MySQL, Interbase y SAP DB simplemente dejan de responder (o tardan demasiado en hacerlo) y el Apache da un timeout que recoge el Apache Benchmark. Pero el PostgreSQL cuando ya no puede absorber más conexiones da un error al generar la página de Too many clients. Veamos las siguiente tablas para ver los resultados:

Peticiones servidas por segundo

	MySQL	PostgreSQL	Interbase	SAP DB
10p	3,17	1,32	4,39	3,24
10	3,04	1,04	4,66	3,30
20p	1,17	1,21	3,77	1,80
20	1,04	1,01	3,40	1,62
30p	0,86	0,61	2,62	1,30
30	0,83	0,67	2,43	1,23
40p	Timeout	Error	1,45	timeout
40	Timeout	Error	1,45	timeout
50p	Timeout	Error	Timeout	timeout
50	Timeout	Error	Timeout	timeout

Mysql bajo pocas conexiones funcionó perfectamente, cumpliendo expectativas, pero no pudo con el Interbase en el tramo final.

PostgreSQL 7.X realmente ha mejorado la velocidad respecto las versiones 6.X. Lo normal en las 6.X es que fueran 3 veces más lentas que el MySQL. Ahora bajo pocas conexiones apenas es la mitad, y la iguala en un entorno cargado.

SAP DB empieza con resultados cercanos al MySQL, y después se aleja para mejor debido a su mejor escalabilidad.

Interbase es el mejor en todos los campos en este terreno (ver Imagen 12). Le pertenecen los mejores tiempos en cada test. Es el Único que llega a 50 peticiones concurrentes sin dar errores y además el que mejor escala.

Si MySQL se llevó el premio del test por fichero, el Interbase se lleva el de base de datos ideal para servidor Internet.

Estos son los resultados en los tests.

Ratio de transferencia en Kb/seg

	MySQL	PostgreSQL	Interbase	SAP DB
10p	763,40	317,09	1059,45	781,79
10	733,98	253,01	1208,55	795,54
20p	280,97	293,34	962,69	433,98
20	251,80	244,93	940,92	389,97
30p	206,85	147,34	664,13	312,52
30	199,07	162,50	658,69	296,07
40p	Timeout	Error	363,22	timeout
40	Timeout	Error	402,41	timeout
50p	Timeout	Error	timeout	timeout
50	Timeout	Error	timeout	timeout

En la siguiente tabla quedan reflejadas otros datos que nos dan una idea de las limitaciones a nivel de tamaño.

	MySQL 4.1.0	**PostgreSQL 7.1**	**Oracle**	**Sybase**	**Access**	**Informix**
Conexiones simultáneas	101	32	41	25	64	269
Columnas por Tabla	2.819	1.600	1.000	250	255	994
Tamaño de fila por Columna	65.534	103.275	255.000	1.960	2.025	32.356
Tamaño de fila por Columna sin Null	65.502	103.275	255	1.941	2.025	32.356
Tamaño de Consulta	1.048.574	16.777.216	16.777.216	65.535	16.777.216	32.766

5.8.CONCLUSIONES

Como habíamos comentado al principio del punto 5, nuestra decisión estaba condicionada por diversos factores ya mencionados. Las siguientes características de Mysql nos reafirman en nuestra decisión:

- El principal objetivo de MySQL es velocidad y robustez.
- Clientes C, C++, JAVA, Perl, TCL, PHP.
- Puede trabajar en distintas plataformas y S.O. distintos.
- Sistema de contraseñas y privilegios muy flexible y segura.
- Todas la palabras de paso viajan encriptadas en la red.
- Registros de longitud fija y variable.
- 16 índices por tabla, cada índice puede estar compuesto de 1 a 15 columnas o partes de ellas con una longitud máxima de 127 bytes.
- Todas las columnas pueden tener valores por defecto.
- Los clientes usan TCP o UNIX Socket para conectarse al servidor.
- El servidor soporta mensajes de error en distintas lenguas.
- Todos los comandos tienen -help o -? Para las ayudas.
- Diversos tipos de columnas como enteros de 1, 2, 3, 4, y 8 bytes, coma flotante, doble precisión, carácter, fechas, enumerados, etc.
- Su bajo consumo lo hacen apto para ser ejecutado en una máquina con escasos recursos sin ningún problema.
- Las utilidades de administración de este gestor son envidiables para muchos de los gestores comerciales existentes, debido a su gran facilidad de configuración e instalación.
- Tiene una probabilidad muy reducida de corromper los datos, incluso en los casos en los que los errores no se produzcan en el propio gestor, sino en el sistema en el que está.

- El conjunto de aplicaciones Apache-PHP-MySQL es uno de los más utilizados en Internet en servicios de foro (Barrapunto.com) y de buscadores de aplicaciones (Freshmeat.net).

No obstante nos encontramos con ciertas limitaciones:

- Carece de soporte para transacciones, rollback's y subconsultas, así como triggers y soporte para escabilidad.
- No ofrece integridad referencial.

Lo que queremos es sencillez de manejo e instalación, por lo tanto, nada es mejor que MySQL. Además existe una infinidad de documentación en la red sobre esta base de datos.

6. Mysql: La opción escogida.

Para diseñar una base de datos relacional es necesario tener claro algunos conceptos básicos como por ejemplo las claves primarias, las claves foráneas, los tipos de tablas disponibles etc…

También es conveniente tener algunas nociones sobre como funciona el administrador de la base de datos.

En los siguientes puntos pretendemos aclarar estos conceptos.

6.1 Claves Primarias

En una base de datos relacional cada tabla tiene una columna o combinación de columnas cuyos valores identifican unívocamente cada fila en la tabla. Esta columna (o columnas) se denomina clave primaria, **PRIMARY KEY**, de la tabla.

La clave primaria tiene un valor único y diferente para cada registro de una tabla, de modo que no hay dos registros de una tabla con una clave primaria que sean duplicados exactos el uno del otro. Una tabla en donde cada registro es diferente de todas las demás se llama una *relación* en términos matemáticos. El nombre de *base de datos relacional* proviene de este término, ya que las relaciones (la tabla con registros distintos) son el corazón de una base de datos relacional.

Cabe destacar que dadas las propiedades de una clave primaria un campo denominado como tal no admitirá valores nulos.

Según [MYS], la sintaxis de una clave primaria es la siguiente (esquema extraido de una de nuestras tablas mediante la instrucción *mysqldump –d ecommerce familias*) :

```
Create table familias (
      familia smallint(6) default ‘0’ NOT NULL,
      descripcion char(50) default ‘’ NOT NULL,
      fabricante smallint(6 ) default ‘0’ NOT NULL,
      PRIMARY KEY (familia)
)
```

6.2. Claves foráneas

Una columna de una tabla cuyo valor coincide con la clave primaria de alguna otra tabla se denomina s denomina una clave foránea, FOREING KEY.

Juntas, una clave primaria y una clave foránea crean una relación padre/hijo entre las tablas que las contienen, del mismo modo que las relaciones padre/hijo de una base de datos jerárquicos.

Lo mismo que una combinación de columnas puede servir como clave primaria de una tabla, una clave foránea puede ser también una combinación de columnas. De hecho, la clave foránea será siempre una clave compuesta (multicolumna) cuando referencia a una tabla con una clave primaria compuesta. Obviamente el numero de columnas y los tipos de datos de las columnas de la clave foránea y en la clave primaria deben ser idénticos unos a otros.

Una tabla puede contener más de una clave foránea si está relacionada con más de una tabla adicional.

Las clave foráneas son parte fundamental del modelo relacional ya que crean relaciones entre tablas en la base de datos.

A diferencia de las claves primarias, las claves foráneas admiten valores nulos. Los estandares de SQL consideran que una clave foránea que contienen un valor null satisface la restricción de integridad referencial, aun cuando el valor de la clave foránea no coincida con ninguna fila en la tabla padre.

Esto puede producir un comportamiento inesperado y poco intuitivo en claves foráneas compuestas.

Como hemos visto en el punto 5, Mysql no soporta la clave foránea hasta su versión 4.0.1. Aunque se las incluya en el esquema de creación de la tabla, el gestor no almacena esta información.

No cabe duda que a la hora de asegurar la integridad referencial de los datos son de gran ayuda y simplifican la tarea del programador. Aún así, hay que tener cuidado a la hora de definirla ya que se pueden producir errores como por ejemplo reglas circulares o cascadas de DELETE's erroneas.

En la versión 4 de Mysql únicamente las tablas del tipo Inno DB soportan claves foráneas a parte de otras características como transacciones o recuperaciones de datos.

La sintaxis para una clave foránea sería la siguiente:

```
        Create table familias (
              familia smallint(6) default '0' NOT NULL,
              descripcion char(50) default '' NOT NULL,
              fabricante smallint(6 ) default '0' NOT NULL
REFERENCES fabricantes(fabricante) ON DELETE CASCADE, ON UPDATE
CASCADE,
              PRIMARY KEY (familia)
        )
```

6.3 Tipos de tablas en MySQL

Una característica más de Mysql es que puedes variar el tipo de tabla después de creada, a continuación detallamos los tipos de tabla de los que disponemos en Mysql.

ISAM: es el formato de almacenaje mas antiguo, y posiblemente pronto desaparecerá. Presentaba limitaciones (los ficheros no eran transportables entre máquinas con distinta arquitectura, no podía manejar ficheros de tablas superiores a 4 gigas).

MYISAM: es el tipo de tabla por defecto en MySQL desde la versión 3.23 y el tipo de tabla que nosotros utilizamos Optimizada para sistemas operativos de 64 bits, permite ficheros de mayor tamaño que ISAM. Además los datos se almacenan en un formato independiente, con lo que se pueden copiar tablas de una máquina a otra de distinta plataforma. Los índices se almacenan en un archivo con la extensión ".MYI" y los datos en otro archivo con extensión ".MYD". Ofrece la posibilidad de indexar campos BLOB y TEXT. Además este tipo de tablas soportan el tipo de dato *VARCHAR.*

Estas tablas presentan algunos inconvenientes, uno de ellos es que pueden llegar a corromperse, es decir, almacenar datos incorrectos. Las principales causas són:

- El proceso mysqld haya sido eliminado en el transcurso de una escritura.
- Una caida del sistema.
- Un error de hardware.
- Un gusano en el código Mysql o MyISAM..

HEAP: Crea tablas en memoria. Son temporales y desaparecen cuando el servidor se cierra; a diferencia de una tabla TEMPORARY, que solo puede ser accedida por el usuario que la crea, una tabla HEAP puede ser utilizada por diversos usuarios. No soportan columnas de autoincremento ni que haya valores nulos en los índices. Los datos son almacenados en pequeños bloques.

BDB: Base de datos Berkeley. TST(Transactions safe tables). Solo en MySQL MAX. Este tipo de tablas permita la realización de transacciones (a partir de la versión 3.23.34), por que es posible la recuperación de datos (COMMIT y ROLLBACK). Estas tablas requieren una clave primaria para cada tabla, clave que ha de crear el administrador de lo contrario Mysql creará una oculta. Otra de sus características es que pueden ser bloqueadas con el comando LOCK. Estas tablas se almacenan en archivos con extensión ".DB".

INNODB: TST (Transactions safe tables), ACID, con posibilidad de commit, rollback, recuperación de errores y bloqueo a nivel de fila. La referencia entre tablas y su mantenimiento, ya puede ser gestionada de forma automática por MySQL, gracias a InnoDB, y así definir tablas con la opción "ON DELETE CASCADE", que nos quitará mucho trabajo de administración a los programadores de aplicaciones basadas en este potente gestor de bases de datos.

Además, InnoDB ofrece unos rendimientos superiores a la anterior tecnología de tablas de MySQL, MyISAM, que podemos comprobar a continuación:

	InnoDB	**MyISAM**
100.000 insert	25s.	40s.
100.000 selects on primary Key	57s.	58s.
100.000 selects on secondary Key	68s.	95s.

Se puede utilizar la sentencia estándar BEGIN WORK seguida de varias consultas y finalizar con un COMMIT o ROLLBACK para completar la transacción. O, se pueden correr en modo AUTOCOMMIT, así que cada consulta es efectivamente una transacción separada.

Pero esto es sólo el comienzo. InnoDB es un motor de bases de datos muy completo que ha sido embebido dentro de MySQL. InnoDB también proporciona lo siguiente:

Recuperación automática ante fallas. Si MySQL se da de baja de una forma anormal, InnoDB automáticamente completará las transacciones que quedaron incompletas..

Integridad referencial. Ahora se pueden definir llaves foráneas entre tablas InnoDB relacionadas para asegurarse de que un registro no puede ser eliminado de una tabla si aún está siendo referenciado por otra tabla.

Bloqueo a nivel de filas. Al usar tablas MyISAM, y tener consultas muy grandes que requieren de mucho tiempo, simplemente no se podían ejecutar más consultas hasta que terminarán las consultas que estaban en ejecución. En cambio, las tablas InnoDB usan bloqueo a nivel de filas para mejorar de manera impresionante el rendimiento.

SELECTs sin bloqueo. Como si el bloqueo a nivel de filas no fuera suficiente, el motor InnoDB usa una técnica conocida como *multi-versioning* (similar a PostgreSQL) que elimina la necesidad de hacer bloqueos en consultas SELECT muy simples. Ya no será necesario molestarse porque una simple consulta de sólo lectura está siendo bloqueada por otra consulta que está haciendo cambios en una misma tabla.

MERGE mas que un tipo de tabla es la posibilidad de dividir tablas MYISAM de gran tamaño (solo útil si son verdaderamente de GRAN tamaño) y hacer consultas sobre todas ellas con mayor rapidez. Las tablas deben ser myisam e idénticas en su estructura.

Cuando hablamos de TST nos referimos a 'Transactions safe tables', o tablas para transacciones seguras. Son menos rápidas y ocupan mas memoria, pero a cambio ofrecen mayor seguridad frente a fallos durante la consulta. Las tablas TST permiten ir introduciendo consultas y finalizar con un COMMIT (que las ejecuta) o ROLLBACK (que ignora los cambios)

Estas tablas sólo están disponibles a partir de la versión 4 de Mysql por lo que nosotros no tendremos la posibilidad de utilizarlas.

Toda esta información puede ampliarse consultado [MYS]

6.4 MysqlAdmin: El Administrador de la base de datos

La principal herramienta de MySQL es **mysqladmin**, que como indica su nombre administra la base de datos. Generalmente el usuario que tiene permisos para trabajar con esta aplicación suele ser root, ya que por defecto es el usuario que administra toso el sistema.

La sintaxis básica de mysqladmin es la siguiente:

```
shell>mysqladmin [OPTIONS] command [command-options] command ...
```

Tal y como se describe en [MYS], los comandos que mysqladmin soporta actualmente son:

- *create databasename*: crea una base de datos.
- *drop databasename*: borra una base de datos.
- *extended-status*: devuelve un mensaje sobre el estado del sistema.
- *kill id:* Mata un hilo de mysql.
- *password:* cambia el password nuevo por el password antiguo.
- *ping:* comprueba si el proceso mysqld(demonio de mysql, para que funcione el sistema debe estar siempre corriendo) esta activo.
- *processlist:* muestra la lista de hilos activos en el servidor.
- *reload:* recargar.
- *refresh:* refresca las tabas y los archivos de log (donde se anotan las consultas constructivas y los posible errores que se hayan producido).
- *shutdown*: apaga el servidor.
- *Slave-star*: crea un hilo esclavo.
- *slave-stop*: detiene el hilo esclavo.
- *status*: genera un mensaje corto sobre es estado del sistema.
- *variables*: muestra las variables.
- *version*: nos da la version del servidor.

7. Diseño y creación de la base de datos

En el momento de diseñar una base de datos debemos tener en cuenta algunos conceptos básicos, así como establecer una metodología concreta. En 1985, Ted Codd definió 12 reglas que toda base de datos debería seguir para ser realmente relacional. Estas reglas no son de obligado cumplimiento pero sientan las bases de como definir una base de datos, así como la forma que deben tener sus tablas.

En la siguiente tabla podemos ver las doce reglas del Codd (esta información ha sido extraida de [Groff 1990]).

La regla de información Toda la información de una base de datos relacional está representada explícitamente a nivel lógico y exactamente de modo (mediante valores en tablas).
Regla de acceso garantizado Todos y cada uno de los datos (valor atómico) de una base de datos relacionales garantizan que sean lógicamente accesibles recurriendo a una combinación de nombre de tabla, valor de clave primaria y nombre de columna. *Explicación: Esta regla refuerza la inportancia de las claves primarias para localizar datos en la base de datos.*
Tratamiento sistemático de valores nulos Los valores nulos (distinto de la cadena de caracteres vacía o de una cadena de caracteres en blanco y distinta del cero o de cualquier otro numero) se soportan en las DBMS completamente relacionales para representar la falta de información y la información inaplicable de un modo sistemático independiente del tipo de datos.
Catalogo en línea dinámico basado en el modelo relacional La descripción de la base de datos se representa a nivel lógico del mismo modo que los datos ordinarios, de modo que los usuarios autorizados puedan aplicar a su interrogación el mismo lenguaje relacional que aplican a los datos regulares. *Explicación: Esta regla requiere que una base de datos sea autodescriptiva, en otras palabras, la base de datos debe contener ciertas tablas de sistema cuyas columnas describan la estructura de la propia base de datos.*
Regla de sublenguaje completo de datos Un sistema relacional puede soportar varios lenguajes y varios modos de uso terminal (por ejemplo, el modo rellenar con blancos). Sin embargo debe haber al menos un lenguaje cuyas sentencias sean expresables, mediante alguna sintaxis bien definida, como cadenas de caracteres, y que sea completa en cuanto al soporte de los siguientes puntos: - Definición de datos - Definición de vista - Manipulación de datos (interactiva y por programa) - Restricciones de integridad

- Autorización
- Fronteras de transacciones (comienzo, cumplimentación, y vuelta atrás)

Explicación: Básicamente recomienda usar algún lenguaje como SQL

Regla de actualización de vista
Todas las vistas que sean teóricamente actualizables son también actualizables por el sistema

Explicación: Esta regla trata de vistas que no son más que tablas virtuales utilizadas para dar a diferentes usuarios diferentes vistas de su estructura

Inserción, actualización y supresión de alto nivel.
La capacidad de manejar una relación de base de datos o una relación derivada como único operando se aplica no solamente a la recuperación de datos, sino también a la inserción actualización y supresión de datos.

Explicación: Requiere que las filas sean tratadas como conjuntos en operaciones de inserción, supresión y actualización

Independencia física de los datos
Los programas de aplicación y las actividades terminales permanecen lógicamente inalterados cualquiera que sean los cambios efectuados ya sea a las representaciones de almacenamiento o a los métodos de acceso.

Explicación: Aislar los datos del usuario.

Independencia lógica de los datos
Los programas de aplicación y las actividades terminales permanecen lógicamente inalterados cuando se efectúen sobre las tablas de base cambios preservadores de la información de cualquier tipo que teóricamente permita alteraciones.

Explicación: Aislar los datos del programa

Independencia de integridad
Las restricciones de integridad especificas para una base de datos relacional particular deben ser definibles en el sublenguaje de datos relacional y almacenables en el catalogo, no en los programas de aplicación.

Independencia de distribución
Un DBMS relacional tiene independencia de distribución

Regla de no subversión
Si un sistema relacional tiene un lenguaje de bajo nivel (un solo registro cada vez), ese bajo nivel no puede ser utilizado para subvertir o suprimir las reglas de integridad y las restricciones expresadas en el lenguaje relacional de nivel superior (múltiples registros a la vez)-

A la hora de diseñar nuestra base de datos hemos seguido una metodología concreta que está dividida en tres etapas concretas:

- **Diseño Conceptual**: En esta etapa deben responderse a las preguntas ¿qué representar?, es decir, qué necesidades debemos contemplar dentro del modelo lógico, y ¿como representarlas?, es decir, qué entidades, relaciones y atributos debemos definir para satisfacer dichas necesidades.
  Para poder responder a estas preguntas desarrollaremos el Diagrama ER (Diagrama de Entidades y Relaciones).

- **Diseño Lógico**: Esta etapa consiste en la transformación de nuestro Diagrama ER en un esquema relacional (con tablas) que usara el gestor de bases de datos para crear nuestra base de datos.

- **Diseño Físico**: Consistirá en la implementación física de la base de datos, es decir, una vez establecidas las tablas y los campos que éstas contienen, expresarlas en el lenguaje adecuado para que el gestor de bases de datos lo entienda, en nuestro caso, en lenguaje SQL.

Entenderemos entidad como un objeto de datos que puede hacer referencia a algo concreto (un articulo, un cliente, etc...) o a algo abstracto (una oferta, un descuento, etc..).

Entenderemos atributo como un objeto de datos perteneciente a una entidad al que se le puede asociar un valor y ser utilizado como operando en una operación aritmética, boleana, etc...

Entenderemos interrelación un objeto de datos que hace posible la selección de entidad por medio de referencia a un atributo de otra entidad (selecciona un pedido según un determinado cliente).

7.1. Diseño Conceptual

Las necesidades que debemos contemplar son:

Artículos

Deben recogerse las tanto los datos explicativos (datasheets, descripciones, imagen del articulo) como los datos comerciales (precios de coste, precios de venta, plazos de entrega etc...). Por lo tanto hemos definido la entidad articulo que contendrá todos los datos referentes a un artículo.

Sus atributos son código, nombre o descripción breve, path_imagen, PVP, precio_coste, descripción completa, tipo_artículo, unidades, plazo_entrega, palabra_clave1,

palabra_clave2, palabra_clave3, path_ficha_técnica, path_ficha_técnica2, subfamilia, categoría y dispo_tienda.

En la siguiente tabla están explicados estos atributos con más claridad.

CAMPO	DESCRIPCION
Código_artículo	Código con el que se hará referencia al artículo dentro de la aplicación
Nombre	Descripción breve del articulo
Path_imagen	Path para poder visualizar la imagen
Precio	Precio del artículo
Precio_coste	Precio de coste del articulo.
Descripción_completa	Definición completa del artículo
Tipo_artículo	Para definir si es artículo de corte (bobinas) o de unidades
Unidades	Para definir si son metros (cuando sean bobinas) o unidades
Categoría	Agrupación en función de las características del artículo
Subfamilia	Agrupación en función del fabricante
Plazo_entrega	Número de días de espera
Palabra_clave1	Palabra clave para la búsqueda en el catálogo
Palabra_clave2	Palabra clave para la búsqueda en el catálogo
Palabra_clave3	Palabra clave para la búsqueda en el catálogo
path_ficha_tecnica	Directorio donde esta almacenada el DataSheet
path_ficha_técnica2	Directorio donde esta almacenada el DataSheet 2
Dispo_tienda	Da el permiso necesario para que el articulo pueda ser incluido en el carrito de la compra

Pedidos

Debemos almacenar los datos que definen un pedido, es decir, el cliente al que esta vinculado, y los artículos que lo forman. Para esto hemos asignar una valor numérico entero a cada pedido y dividir esta necesidad en dos entidades:

- Cabecera: contendrá los datos del cliente al que está vinculado el pedido así como la fecha de creación. Sus atributos más importantes son pedido, cliente, fecha e importe total. La totalidad de los atributos queda reflejada en la siguiente tabla:

CAMPO	DESCRIPCION
Cliente	Código del cliente
Pedido	Código del pedido
Domicilio	Domicilio del cliente
Población	Población del cliente
Provincia	Provincia del cliente
Fecha	Fecha de introducción
Total_pedido	Importe total del pedido
Aceptado	Pedido aceptado por Cervi

- Líneas: habrá una entidad línea para cada articulo, por lo que entenderemos un pedido como una cabecera y un conjunto de líneas. Sus atributos son pedido, artículo, cantidad, precio y descuento. En la siguiente tabla podemos ver todos los atributos de esta entidad:

CAMPO	DESCRIPCION
Pedido	Código del pedido
Artículo	Código del artículo
Descripción	Descripción del articulo
Cantidad	Cantidad pedida
Precio	Precio del artículo
Dto	Descuento
Precio_línea	Precio neto de la línea
Almacén	Almacén de donde sale el material
Plazo_entrega	Días hasta la fecha de entrega

Queda claro que en este caso ambas entidades están interrelacionadas por el atributo pedido que ambas comparten.

Clientes

Esta entidad almacenará todos los datos pertenecientes al cliente (nombre, domicilio, login y password, etc...). Sus atributos son código del cliente, login, password, domicilio, nombre, población, C.P., provincia, país, cif, num_visitas, acep_prop, forma_loc, fecha_alta, e_mail y web. En la siguiente tabla parece una explicación de cada uno de estos atributos:

CAMPO	DESCRIPCION
Código_cliente	Código con el que se hará referencia al cliente dentro de la aplicación
Login	Nombre para identificar al usuario
Password	Contraseña
Web	Dirección web del cliente
E_mail	Dirección de e_mail de contacto
Domicilio	Domicilio del cliente
Nombre	Nombre completo del cliente
Código_postal	Código Postal del cliente.
Población	Población del cliente.
Provincia	Provincia del cliente.
País	País del cliente.
Cif	Código de identificación fiscal
Num_visitas*	Número de visitas realizadas a la página web
Acepta_prop	Consentimiento de envío de información y publicidad
Forma_localización	Como localizó nuestra web
Fecha_alta	Fecha de registro como usuario

Descuentos

Esta entidad relaciona el precio del artículo con el cliente que compra dicho artículo y la familia a la que pertenece asociándole un descuento específico. Sus atributos serán, por tanto, cliente, familia y descuento. En la siguiente tabla se relacionan estos atributos:

CAMPO	DESCRIPCION
Dto	Descuento aplicable.
Familia	Familia a la que se refiere el descuento.
Cliente	Cliente al que se le asocia el descuento.

Nótese que se relaciona el descuento únicamente con la jerarquía asociada a los fabricantes y no a la jerarquía asociada con las características del artículo. Esto es así, porque únicamente los artículos de electrónica (routers, switchos y otros equipos) estarán comprendidos dentro de esta categoría y serán los únicos que podrán ser accesibles desde el servicio de venta electrónico. Todos los artículos de cables estarán comprendidos dentro del catálogo virtual, pero no podrán comprarse en la red debido a la complejidad de su facturación, que no sólo esta en función del cliente, sino también de la cantidad que se compre.

Si en esta tabla tuviésemos una entrada no por familia, sino por articulo, nuestro sistema de descuentos sería mucho más exacto aunque aumentaría el tamaño de la tabla y también la complejidad a la hora de tratarla. También sería más costoso en tiempo hacer un adecuado mantenimiento.

Actos

Sería interesante mostrar en la web una relación de los eventos más importantes relacionados con el sector (ferias o convenciones). Sus atributos son acto (código númerico), nombre, fecha_inicio, fecha_fin, provincia, poblacion, dirección, codigo_postal, url, mail y descripción. En la siguiente tabla tenemos una descripción más detallada de estos atributos:

CAMPO	DESCRIPCION
Acto	Código del acto
Nombre	Nombre del acto
Fecha_inicio	Fecha de inicio del acto
Fecha_fin	Fecha del final del acto
Provincia	Provincia donde se celebra el acto
Población	Población donde se celebra el acto
Dirección	Dirección donde se celebra el acto
Código Postal	Código Postal
Url	Dirección de la página web del acto
Mail	Dirección de e_mail de contacto
Descripción	Descripción del acto

Ofertas

En esta estancia almacenaremos los datos de los artículos sujetos a algún tipo de promoción especial. Sus atributos son oferta (código numérico), articulo, precio, fecha_fin, imagen_publicidad y cond_esp. En la siguiente tabla aclaramos el significado de cada uno de ellos:

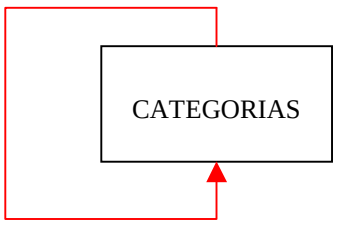
CAMPO	DESCRIPCION
Oferta	Código numérico que identifica la oferta
Artículo	Código con el que se hará referencia al artículo dentro de la aplicación
Imagen_publicidad	Imagen promocional del articulo
Precio_promoción	Precio especial
Fecha_fin_promoción	Fecha en la que la promoción deja de ser válida
Condiciones_especiales	Para definir condiciones especiales de pago o cualquier dato que se pudiera necesitar

Categorías

Esta entidad relaciona la organización jerárquica de los artículos en función de sus características. Sus atributos son categoría_padre, categoría_hijo y descripción de categoría_hijo.

Esta entidad presenta una peculiaridad. Podríamos decir que es recursiva ya que siempre debe recurrir a ella misma para conocer la categoría_padre.

Las categorías superiores, es decir, las que se encuentran en el primer nivel de la estructura jerárquica tendrán por lo tanto el campo de categoría_padre vacío y las categoriás_hijas no aparecerán en ninguna entrada como categoría_padre.



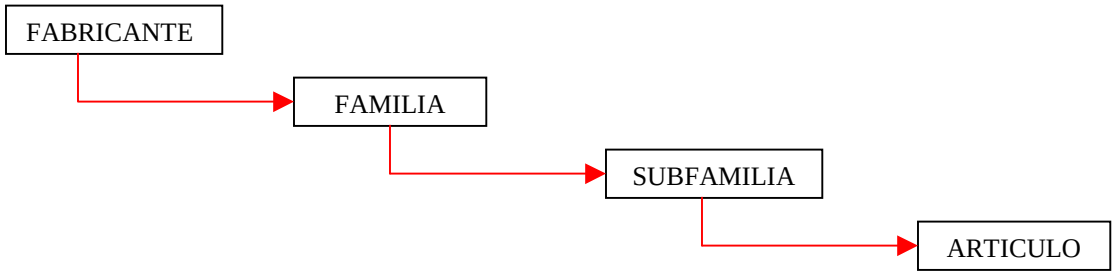
En la siguiente tabla podemos ver sus atributos relacionados:

CAMPO	DESCRIPCION
Categoría Padre	Código numérico de la categoría superior
Categoría Hijo	Código numérico de la categoría
Descripción	Descripción de la categoría.

Fabricantes, familias y subfamilias

Estas tres entidades relacionan la jerarquía de los artículos en función del fabricante.

La estructura seria la siguiente:



Las tablas quedarían así:

CAMPO	DESCRIPCION
Fabricante	Código numérico que identifica al fabricante.
Descripción	Nombre del fabricante.

CAMPO	DESCRIPCION
Familia	Código numérico que identifica a la familia
Descripción	Nombre de la familia
Fabricante	Fabricante al que pertenece

CAMPO	DESCRIPCION
Subfamilia	Código numérico que identifica a la subfamilia.
Descripción	Nombre de la subfamilia.
Familia	Familia a la que pertenece.

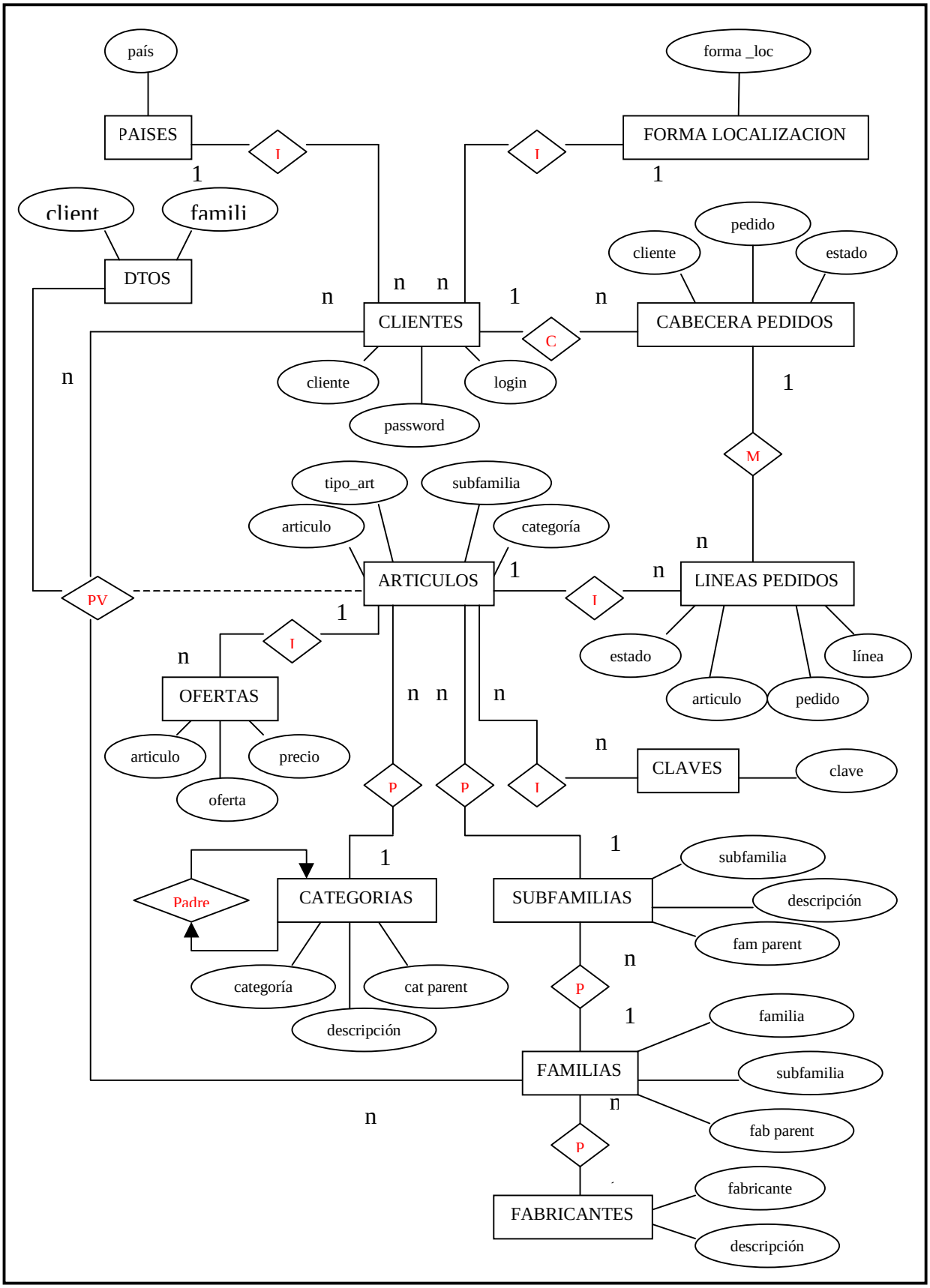
7.1.1 Diagrama E-R

En la siguiente página podemos ver como quedaría estructurado el diagrama E-R.

Las relaciones aparecen representadas con rombos, cuyo contenido significa el tipo de relación establecida:

- I: Incluye.
- C: Compra.
- M: Material
- P: Pertenece
- PV: Precio de venta
- Padre: Relación de jerarquía padre e hijo.

DIAGRAMA E-R


país
PAISES
I
1
forma _loc
FORMA LOCALIZACION
I
1
client
famili
DTOS
n
n n
CLIENTES
1
n
C
CABECERA PEDIDOS
pedido
cliente
estado
cliente
login
password
n
1
M
tipo_art
subfamilia
articulo
categoría
n
n
1
ARTICULOS
LINEAS PEDIDOS
PV
I
1
I
n
OFERTAS
estado
línea
articulo
pedido
n n
n
articulo
precio
oferta
n
CLAVES
clave
P
P
I
1
1
subfamilia
Padre
CATEGORIAS
SUBFAMILIAS
descripción
fam parent
n
categoría
cat parent
descripción
P
1
familia
FAMILIAS
subfamilia
n
n
fab parent
P
fabricante
FABRICANTES
descripción

Las entidades están representadas por rectángulos mientras que sus atributos son los ovoides que están ligados a ellas.

Este esquema muestra claramente la relación entre la cabecera y las líneas del pedido y a su vez la relación de estas con los artículos. La relación entre cabecera y líneas es de 1 a n lo que significa que una cabecera puede tener mas de una línea pero una línea no puede tener más de una cabecera. A su vez un articulo puede aparecer en varia línea pero una línea sólo puede contener una línea.

La relación que hay entre los clientes y las cabeceras también es de 1 a n , lo que significa que un cliente puede aparecer en más de una cabecera pero una cabecera sólo puede tener un cliente.

Como podemos apreciar en el diagrama la mayoría de la interrelaciones son de este estilo, excepto la que seda entre el precio del articulo por un lado y el cliente y la familia por otro. Esta relación es del tipo m a n, es decir que un familia (que no olvidemos es una agrupación de diferentes artículos) puede tener diferentes descuentos (lo que equivale a decir que puede tener diferentes precios) y que a la vez un mismo descuento puede pertenecer a diferentes familias. Es por este motivo que tiene sentido la existencia de la entidad descuento, y a que refleja esta complejidad.

También debemos comentar la relación recurrente que se le ah añadido a la entidad categoría que como anteriormente hemos comentado, recurre a ella misma para ver cual es la categoría jerárquicamente superior a la que pertenece.

7.2 Diseño Lógico

En este punto definiremos las tablas necesarias, que generalmente se corresponderán con cada una de las entidades definidas en el punto 7.1 y otra para la interrelación de precios, clientes y familias, que hemos denominado descuentos.

También definiremos otras tablas de soporte, como por ejemplo una que contenga una lista de países para mostrarse en un select o la que contenga los numeradores que se consultara cada vez que queramos crear un acto, un pedido o una oferta.

Definiremos también los campos que actuarán como índices (PRIMARY KEY, definida en el punto 6.1), que evitan que puedan existir dos filas exactamente iguales. Como hemos comentado anteriormente un índice puede ser un único banco o una combinación de varios campos, en cuyo caso reciben el nombre de índices múltiples. Serán índices todos los campos marcados de color rojo.

Nuestras tablas serán, por lo tanto, las siguientes:

Actos	
Campo	**Tipo de datos**
acto	int(4)
nombre	char(50)
fecha_inicio	date
fecha_fin	date
provincia	char(50)
poblacion	char(50)
dirección	char(50)
codigo_postal	char(6)
url	char(50)
mail	char(50)
descripción	char(100)

Descuentos	
Campo	**Tipo de datos**
cliente	int(9)
familia	int(6)
descuento	int(4)

Categorías	
Campo	**Tipo de datos**
parent	char(5)
descripción	char(15)
nombre	char(5)

Familias	
Campo	**Tipos de datos**
familia	int(6)
descripción	char(50)
fabricante	int(6)

Clientes	
Campo	**Tipo de datos**
cliente	int(9)
login	char(8)
password	char(40)
web	char(50)
domicilio	char(50)
nombre	char(50)
poblacion	char(50)
cos_pas	char(6)
país	char(40)
cif	char(11)
num_visitas	int(6)
acep_prop	int(4)
forma_loc	char(40)
fecha_alta	date
e_mail	char(50)
provincia	char(50)

Artículos	
Campo	**Tipo de datos**
articulo	char(8)
nombre	char(30)
path_imagen	char(50)
precio	decimal(8,2)
precio_coste	decimal(8,2)
descrip_comp	char(250)
tipo_artículo	char(1)
unidades	char(2)
plazo_entrega	int(2)
palabra_clave1	char(40)
palabra_clave2	char(40)
palabra_clave3	char(40)
path_fic_tec	char(50)
path_fic_tec2	char(50)
dispo_tienda	char(1)
subfamilia	int(6)
categoría	int(6)

Fabricantes	
Campo	**Tipo de datos**
Fabricante	int(6)
descripción	char(50)

Subfamilias	
Campo	**Tipo de datos**
subfamilia	int(6)
descripción	char(50)
familia	int(6)

Usuarios_gestión	
Campo	**Tipo de datos**
usuario	int(3)
login	char(15)
db user	char(15)
password	char(32)
nombre	char(60)
e_mail	char(60)
cargo	char(30)
fecha_alta	date
fecha_baja	date

Países	
Campo	**Tipo de datos**
país	char(100)

Localización	
Campo	**Tipo de datos**
forma	char(50)

Cab_pedidos	
Campo	**Tipo de datos**
pedido	int(9)
cliente	int(9)
domicilio	char(50)
nombre	char(50)
poblacion	char(50)
codpos	char(6)
provincia	char(50)
país	char(11)
cif	char(11)
fecha	date
total_pedido	decimal(11,2)
total_coste	decimal(11,2)
aceptado	int(4)

Palabras_claves	
Campo	**Tipo de datos**
palabra_clave	char(40)

Numeradores	
Campo	**Tipo de datos**
código	int(3)
descripción	char(50)
valor	int(8)

Lin_pedidos	
Campo	**Tipo de datos**
pedido	int(6)
articulo	char(8)
descripción	char(50)
cantidad	int(6)
precio	decimal(8,2)
precio_coste	decimal(8,2)
descuento	int(4)
precio_línea	decimal(8,2)
precio_c_línea	decimal(8,2)
almacén	char(2)
plazo_entrega	int(4)
línea	int(4)

Ofertas	
Campo	**Tipo de datos**
Oferta	int(6)
articulo	char(8)
precio	decimal(8,2)
fecha_fin	date
imagen	char(75)
cond_esp	char(50)

Menús	
Campo	**Tipo de datos**
path	char(50)
path_imagen	char(50)

El numero que acompaña al tipo int es el numero de bytes que puede ocupar por lo que un campo int(3) podrá almacenar valores comprendidos entre –8.388.608y 8.388.607 si utiliza signo o entre 0 entre 16777215 si no lo utiliza.

En los tipos char(n) se pueden almacenar hasta 255 caracteres. Cada carácter estará almacenado en un byte.

Los tipos decimal (n, m) ocuparán (n+2) bytes su m>0 o (n+1) bytes si m=0.

Aparecen otras tablas, aparte de las ya definidas como entidades, que actúan como soporte para la aplicación de gestión. Por ejemplo Menús, que almacena los paths de las imágenes que actúan como botones así como el path de la pagina web que debe descargarse en cada uno de los casos. Esto nos permite añadir programas a la gestión sin tener que modificar la pagina web principal.

7.3 Diseño Físico

En este punto pasaremos el esquema relacional del punto 7.2 a sentencias SQL que se ejecutaran en el motor.

La primera sentencia será:

```
create database ecommerce
```

Esta sentencia crea un directorio con el nombre ecommerce en /var/lib/mysql, que es el directorio donde nuestra configuración linux crea todas las bases de datos. Tanto el usuario como el grupo al que pertenece este directorio es mysql (por defecto). Una vez creado este directorio, MySQl, almacenará dentro tanto las estructura como los datos de las tablas que vayamos creando.

Para cada tabla que nosotros creamos MySQL crea tres archivos:

- Un archivo donde almacena los datos, con la extensión .ISD.
- Un archivo donde almacena los índices, con la extensión . ISM.
- Un archivo donde almacena la estructura, con la extensión .frm

Una vez creada la base de datos debemos seleccionarla para poder trabajar con ella.

```
use  ecommerce
```

Ahora ya podemos definir las tablas. Pero antes de esto nos gustaría comentar ciertas dificultades con las que nos topamos cuando intentamos crear tanto el motor de la base de datos como la propia base de datos.

En un primer momento intentamos trabajar con la versión 4.0.1 de MySQL, la cual, en teoría (porque realmente no es así...), incluye el soporte para claves foráneas, explicadas más extensamente en le punto 6.2. Cuando después de un largo y entretenido proceso de instalación de esta versión (ya que hemos descubierto que instalar algo en linux no es ni mucho menos trivial) conseguimos un motor de MySQL para una configuración de un sistema de pequeñas dimensiones (no más de 64Mb de RAM), descubrimos que realmente esta versión aún no tiene integrada dicha característica (cabe remarcar que la documentación que consultamos en [MYS] afirma que esta característica está implementada desde su versión 4.0).

Nuestro esquema era del tipo:

```
      CREATE TABLE subfamilias (
            subfamilia SMALLINT(6) DEFAULT '0' NOT NULL ,
            descripción char(50) DEFAULT '' NOT NULL,
            familia   SMALLINT(6)   DEFAULT   '0'   NOT   NULL   REFERENCES
familias(familia) ON DELETE CASCADE, ON UPDATE  CASCADE,
            PRIMARY KEY (subfamilia)
```

```
) TYPE=InnoDB;
```

donde el texto coloreado es una clave foránea que hace referencia a la clave primaria de la tabla familias. El motor no devolvió ningún error cuando ejecutamos esta sentencia pero al pedirle que nos devolviera la estructura con la instrucción *mysqldump –d ecommerce subfamilias* nos devolvió el siguiente esquema:

```
CREATE TABLE subfamilias (
      subfamilia SMALLINT(6) DEFAULT '0' NOT NULL ,
      descripción char(50) DEFAULT '' NOT NULL,
      familia SMALLINT(6) DEFAULT '0' NOT NULL,
      PRIMARY KEY (subfamilia)
) TYPE=InnoDB;
```

Como podemos observar no aparece la referencia de la clave foránea y pruebas posteriores realizadas sobre la base de datos ya funcionando nos han demostrado que realmente no está implementada esta característica. En este punto decidimos volver a la versión inicial que viene instalada por defecto con nuestra configuración linux, por resultarnos más conocida y ofrecernos casi las mismas características.

El esquema final de nuestra base de datos aparece en el anexo I.

BLOQUE III

Aplicación para la gestión de la base de datos

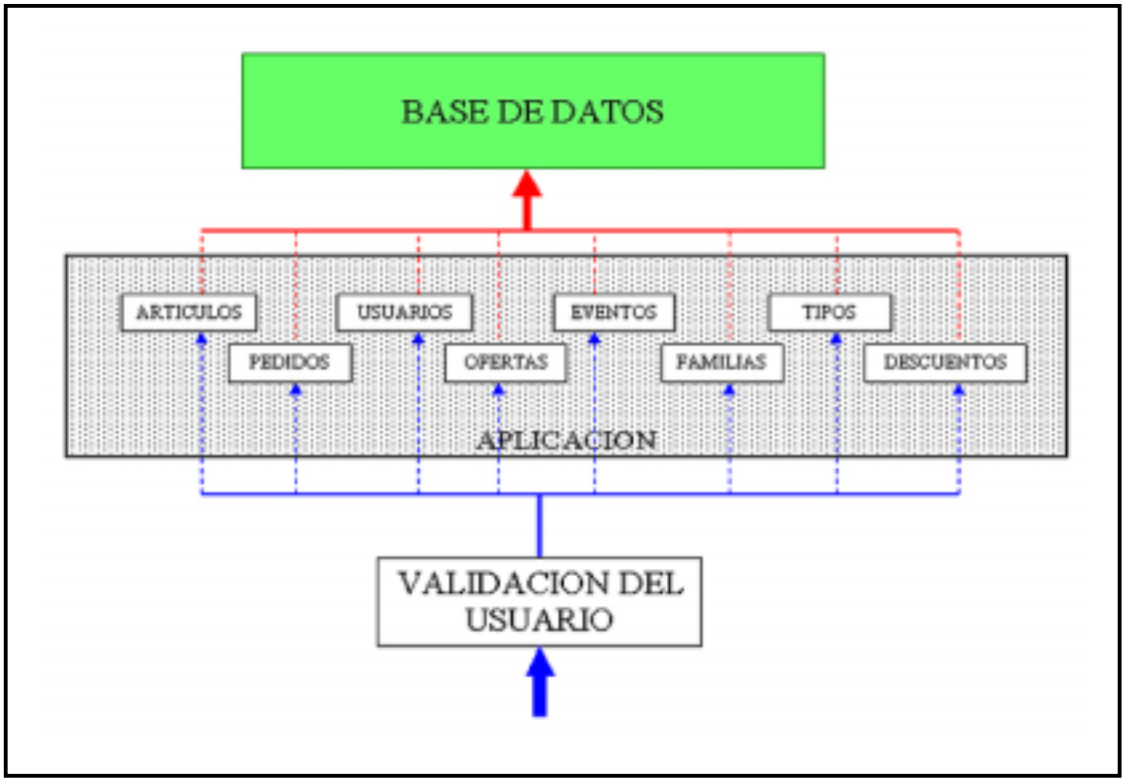
8. Sistema de gestión de la base de datos

Este punto abarca el segundo y más importante objetivo del proyecto y el que más trabajo ha supuesto: la creación de un sistema de gestión para la página web.

Para poder realizar un correcto mantenimiento de la base de datos hemos creado un sistema de gestión. Este sistema está desarrollado en entorno web, es decir, con paginas web. Para la creación de este sistema hemos usado los lenguajes html, javascript, php, SQL y Css. En los siguientes puntos explicaremos como ha sido estructurado, además de explicar algunos detalles que creemos interesantes a nivel de programación.

8.1. Arquitectura de la aplicación.

Esta aplicación actúa como interfície entre el usuario y la base de datos de manera que un usuario que no tenga conocimientos sobre bases de datos o no conozca el lenguaje SQL puede hacer un mantenimiento de los contenidos de la base datos. Sin embargo este sistema de gestión no permite que se altere la estructura de la base de datos, es decir, que no se pueden crear, eliminar o modificar tablas. La siguiente imagen muestra de manera gráfica la arquitectura de la aplicación.



La aplicación esta formada por un conjunto de programas con una función especifica.

En un primer paso el usuario se encuentra con una página de validación donde debe introducir su password y su contraseña. Una vez validado, la aplicación nos dirige a una página principal que esta divida en dos frames o marcos. En el frame de la izquierda aparece un menú con links a los diferentes programas creados. En el frame de la derecha es donde se despliegan los programas activados por los diferentes links. De esta manera el usuario puede "navegar" por la aplicación.

8.2. Aplicaciones principales.

En este apartado explicaremos las funcionalidades básicas de cada uno de los programas que hemos creado. Cada uno de estos programas se encarga de mantener una tabla en concreto aunque a veces este mantenimiento afecta a otras tablas como es el caso del programa encargado del mantenimiento de artículos, que actúa sobre la tabla artículos y sobre la tabla palabras_claves.

Comentaremos además algunas peculiaridades que nos gustaría destacar.

8.2.1. Validación de usuarios.

La validación de usuarios la realizan conjuntamente dos programas:

- ***ecommerce_ini.php*** que el formulario donde el usuario debe introducir su password y su login.
- ***ecommerce_aut.php*** que es el programa que se encarga de establecer una conexión con la base de datos para realizar la comprobación y generar una cookie que identifique al usuario y guarde el tiempo de sesión.

Una vez que el usuario ha sido validado pasa a la pagina ***ecommerce_main.php*** que contiene dos frames como hemos explicado anteriormente. En este punto debemos comentar que la conexión a ***ecommerce_aut.php*** se realizara con una conexión segura mediante SSL (ver 8.4.2), por lo tanto debemos llamar a la página del siguiente modo:

```
https://192.168.10.2/ecommerce_aut.php
```

De este modo cifraremos el canal para que no pueda ser escuchado. Además el password es almacenado en la base de datos después de aplicarle el algoritmo Hash que no es más que una forma de encriptación bastante conocida.

8.2.2. Gestión de Artículos

Esta aplicación se encarga de insertar, modificar y borrar artículos. Además se puede utilizar para realizar consultas sobre un articulo en concreto.

Este programa tiene dos particularidades que nos gustaría resaltar. En primer lugar se encarga de enviar al servidor los archivos necesarios para definir completamente un artículo, es decir, la foto y las 2 hojas técnicas contempladas. Para esto, dentro del formulario incluimos un tipo de objeto especial llamado ***file.*** Este objeto tiene la propiedad de desplegar una ventana para seleccionar cualquier archivo que tengamos almacenado en nuestro PC.

Para poder incluir este objeto debemos hacer algunas modificaciones en la etiqueta <FORM> (formulario) por lo que quedara como sigue:

```
<form name=Formato method=POST enctype="multipart/form-data"
                    action="articulos.php">
```

De esta forma le indicamos que le enviaremos información de diferentes tipos.

Una vez echo esto sólo debemos utilizar el comando ***copy*** de PHP pasándole como parámetro el path completo del archivo. Según nuestra configuración de PHP el tamaño máximo del archivo es de 2MB, tamaño más que suficiente para nuestras necesidades.

En caso de que no pueda copiar el archivo nos avisará generando un mensaje de error.

En segundo lugar nos gustaría comentar que este programa se hace de manera indirecta el mantenimiento de otra tabla: palabras_claves. Cada vez que insertamos un artículo el programa comprueba si existen las palabras claves que nosotros hemos designado, si no existen las crea para que podamos reutilizarlas con otros artículos. Del mismo modo, al eliminar un articulo comprueba si las palabras claves que lo definían aparecen en algún artículo más. Si no es así el programa eliminará estas palabras claves. De esta manera mantenemos, de forma transparente para el usuario la tabla palabras_claves.

De la misma manera también se hace un mantenimiento indirecto de los fabricantes, familias y subfamilias. Se le da al usuario la opción de eliminar si fuera necesario la subfamilia, la familia y el fabricante (siempre que haya ningún articulo más asociado a esta subfamilia). Para este propósito contamos con un checkbox que actúa como semáforo a la hora de eliminar.

En el punto 8.4 comentaremos algunos puntos del programa donde nos hemos visto obligados a alejarnos de los standards.

8.2.3. Gestión de fabricantes.

Este programa se encarga de mantener 3 tablas:

- Fabricantes: inserta, busca y borra fabricantes de esta tabla.
- Familias: inserta, busca y borra familias de esta tabla.
- Subfamilias: inserta, busca y borra subfamilias de esta tabla.

Hemos dividido esta aplicación en tres partes, donde el objeto principal en cada una de ellas es un select. Este select actúa como un listado.

Cuando la página se carga por primera vez el único select que tiene datos es el de fabricantes, ya que estos datos se cargan por defecto. En el select aparece únicamente la descripción de cada fabricante, pero internamente cada fabricante está definido por un código numérico de cuatro dígitos. Cuando seleccionamos una de las diferentes opciones se cargan en los inputs del lado derecho los datos relacionados con este fabricante: descripción y código numérico.

Cabe destacar que este código numérico es único, es decir, que identifica únicamente a un fabricante. Es por este motivo que en la estructura de esta tabla se ha añadido una clave primaria, ya que así se evitan las inserciones de registros repetidos.

En el select inferior aparecerán relacionadas todas las familias pertenecientes a este fabricante. De igual modo sólo vemos su descripción, aunque también tienen asignado un código numérico de 4 dígitos. Este código depende del código asignado al fabricante. Si por ejemplo, nuestro fabricante tiene el código 1000 todas las familias de este fabricante comenzaran por 1 a lo que añadiremos tres dígitos más que identificaran de forma inequívoca a cada familia. En este caso también se ha utilizado una clave primaria.

***1**000 – MOLEX* →
- ***1**010 – Cajas de superficie*
- ***1**020 – Módulos CAT 5 E*
- ***1**030 – Módulos CAT 6*
- ...

Cuando seleccionemos una familia de este select en el select inmediatamente inferior aparecerán las subfamilias pertenecientes a esta familia. Igualmente sólo podremos ver su descripción pero también tienen un código numérico. En este caso el código numérico será del tipo AAAX donde AAA son los primeros tres dígitos de la familia a la que pertenece.

***1**000 – MOLEX* → ***104**0 – Módulos Fibra Óptica* →
- ***104**1 – Conector ST*
- ***104**2 – Conector SC*
- ***104**3 – Conector MTRJ*

En la tabla de subfamilias también se ha añadido un clave primaria por los que un código identifica a una única subfamilia.

Al lado derecho del select de subfamilias tenemos un textarea donde aparecen relacionados los artículos vinculados a una subfamilia. Esta información es meramente informativa ya que no se permite ningún tipo de modificación sobre ella. Como se puede ver la estructura es totalmente jerárquica.

Para proteger la integridad referencial se ha implementado un conjunto de funciones que vienen a tener el comportamiento que tendría un clave foránea. Esto equivale a decir que cuando un fabricante es eliminado también se eliminan sus familias y sus subfamilias, de este modo no quedan familias o subfamilias sin estar vinculadas con ningún fabricante. Del mismo modo cuando se elimina una familia se eliminan también todas sus subfamilias.

8.2.4. Gestión de Categorías/Tipos

En este programa se realiza el mantenimiento de la tabla categorías, por lo tanto, se pueden insertar, buscar, modificar y borrar categorías y subcategorías.

El funcionamiento de este programa es similar al funcionamiento del programa que hemos explicado en el nivel anterior con la única diferencia de que aquí se aplican sólo dos niveles a la hora de diseñar la estructura del programa.

8.2.5. Gestión de Ofertas

Este programa se encarga del mantenimiento de la tabla oferta por lo que se podrán insertar, buscar, modificar y borrar ofertas. Como las ofertas hacen referencia a los artículos este programa necesita trabajar con datos proporcionados por esta tabla.

Nótese que cada oferta esta definida por un número que la identifica de forma única. Este número lo genera automáticamente la aplicación consultado una tabla donde hay almacenados algunos numeradores.

Al igual que en el programa de gestión de artículos, damos la posibilidad al usuario de almacenar en el servidor una imagen publicitaria, por lo que de nuevo volvemos a utilizar el objeto ***file*** y la instrucción de PHP ***copy.***

Al cargar por primera vez el programa de ofertas aparecen en el select de la derecha todas las ofertas seleccionadas cuya fecha final sea superior a la fecha final. Cuando seleccionamos una de las opciones los datos relacionados con esta oferta se cargan en los inputs de la derecha.

8.2.6. Gestión de pedidos

El funcionamiento de este programa es diferente a los comentados hasta ahora. Este programa se encarga del mantenimiento de dos tablas: lin_pedidos y cab_pedidos. Esto nos obliga a dividir el programa en dos bloques bien diferenciados. Por un lado tenemos el bloque que se encarga de generar cabeceras y por el otro el bloque que se encarga de generar líneas.

En el bloque de las cabeceras se pueden insertar, modificar, borrar o buscar diferentes cabeceras. Al igual que en la aplicación anterior, cada cabecera estará definida por un numero que genera la aplicación. Este número además relaciona la cabecera con las líneas del pedido.

Cuando se carga un pedido automáticamente aparecen debajo las líneas vinculadas a éste, en una especie de listado donde se muestra el artículo, la descripción, el precio, la cantidad y el descuento. La aplicación aún añade un valor más que el usuario no ve: el número de línea. Con el número de línea y el número de pedido se puede identificar de manera única a la línea.

La numeración de la línea empieza en la 100. Esto es así porque así nos aseguramos de que las líneas siempre tengan 3 dígitos. Ahora sólo nos queda explicar por que queremos que siempre sean 3 dígitos.

Para empezar dentro de aplicación diferenciamos entre el número de línea real y el número de línea pintada en la pantalla. Lo aclararemos con un ejemplo:

Imaginemos que hemos insertado dos líneas:

Artículo	Línea Real	Línea Pintada
00005400	100	100
00005410	101	101

Esta sería la numeración lógica. Pero se puede dar el caso de que eliminemos la línea real 100 y añadamos otra línea, que en este caso recibiría el valor 102 por lo que tendríamos lo siguiente:

Artículo	Línea Real	Línea Pintada
00005410	101	100
00005406	102	101

Todos los objetos de una línea tienen el nombre de la siguiente manera:

nombre = “articulo” + Línea_Pintada.
nombre = “descripcion” + Linea_Pintada.
....

Si únicamente tuviéramos una variable podríamos acceder de forma correcta a los datos del formulario, pero no a los almacenados dentro de la base de datos. Es por este motivo que para obtener los datos dentro del formulario usamos el valor de Línea_Pintada.

Ahora bien, cuando deseamos acceder a estos datos desde PHP debemos conocer exactamente el nombre del objeto que almacena el valor deseado. Por lo tanto de algún modo debemos conocer el valor de Línea_Pintada. No nos queda más remedio que leer que valor del nombre de los objetos de la línea y como esto no es más que una cadena debemos saber exactamente que posiciones de esta cadena contienen la información deseada. Es por este motivo que fijamos la longitud del valor en tres dígitos.

Una vez que estemos en PHP tendremos:

```
$articulo = “articulo” . $Linea_Pintada;
```

Como es evidente esta variable no contiene el valor de la variable que queremos, sino el nombre. Es evidente que en este caso nos vemos obligados a usar punteros. La sintaxis será entonces la siguiente:

```
$valor = ${$articulo};
```

Para obtener el valor de línea real (sabiendo que ha sido almacena en un objeto oculto para el usuario) tendremos lo siguiente:

```
$lin = ${$linea_real}:
```

Una vez obtenido esto las sentencias de SQL serían del tipo:

```
$sql="update lin_pedidos set articulo='${$articulo}' where
pedido='$p' and linea='${$lin}'";
```

El programa da la opción de insertar, modificar y borrar líneas.

8.2.7. Gestión de usuarios

Esta aplicación realiza el mantenimiento de la tabla usuarios_gestión, por lo tanto permite insertar, buscar, modificar y borrar.

Cada usuario está definido por un identificador, un login y un password. El identificador es un código numérico aunque en este caso no se ha generador un contador para darle libertad al administrador (de esta manera se podrían crear grupos en función del identificador). El identificador define a un usuario de manera única. Al igual que en otros casos hemos añadido a la tabla una clave primaria para asegurarnos.

La longitud máxima del login es de 15 caracteres mientras que el password no debe superar los 8. En este caso el password no es generado automáticamente por la aplicación, es un dato que debe introducir el administrador. Al igual que en la validación de usuarios, el password se almacena en la base de datos una vez que se le ha aplicado el algoritmo hash.

En esta aplicación es donde se decide si el usuario tendrá permisos como visitante o como administrador. El usuario que sea visitante únicamente podrá realizar consultas, pero no podrá alterar registros (no insertar, ni modificar, ni borrar).

8.2.8. Gestión de eventos.

Este programa se encarga de mantener actualizada la tabla de actos que almacena los datos de celebraciones y otros datos relacionados con el sector de la empresa. Su función principal es por lo tanto insertar, modificar, buscar o borrar actos de esta tabla.

Cada acto esta representado por un código numérico que lo define de manera única. Este código se genera automáticamente en la aplicación usando un contador, al igual que en otros programas.

La estructura es similar a la de otros de los programas ya comentados. Tenemos un select donde aparecen relacionados todos los actos y cuando seleccionamos una opción, los datos relacionados con este acto se cargan en los inputs del lado derecho.

8.2.9. Gestión de descuentos.

Esta aplicación se encarga del mantenimiento de la tabla de descuentos, por lo tanto se permite modificar, insertar, borrar y buscar descuentos. Debemos decir que la aplicación muestra los descuentos asociados a un cliente en particular por lo que el programa no mostrará todo el contenido de esta tabla.

Como ya dijimos cuando explicamos la entidad descuentos, los descuentos están asociados al cliente y las familias, no directamente con el articulo, por lo que el programa recibe información de la tabla familias.

8.2.10. Gestión de clientes.

Esta aplicación se encarga del mantenimiento de la tabla clientes, por lo tanto podemos insertar, modificar, borrar y buscar clientes. El código de cliente lo decide el usuario ya que esta en función de la localización geográfica de cliente. Así todos los clientes *comprendidos entre 0 y 19999 son de Cataluña, del 20000 al 29999 son de Valencia y Levante, etc...*

En un principio el login del cliente es su mismo código de cliente, y el password es una combinación de 8 caracteres generado automáticamente por la aplicación. La función que genera el password es la siguiente:

```
function generar_password()
{
        $i = 0;
        $password = "";
        $pw_largo = 8;
        $desde_ascii = 50; // "2"
        $hasta_ascii = 122; // "z"
        while ( $i < $pw_largo )
        {
                mt_srand ((double)microtime()*1000000);
                $numero_aleat=mt_rand($desde_ascii, $hasta_ascii);

                if(($numero_aleat<58||$numero_aleat>64) && $numero_aleat
!=73 && $numero_aleat!=79 && ($numero_aleat<91 || $numero_aleat>96) &&
$numero_aleat!=108 && $numero_aleat!=111)
                {
                        $password = $password . chr( $numero_aleat );
                        $i++;
                }
        }
        return $password;
}
```

Esta función genera una cadena de 8 caracteres aleatorios que contiene letras en mayúscula y en minúscula y números.

En la aplicación se incluyen algunos campos para obtener datos estadísticos como por ejemplo el número de veces que el cliente ha visitado la web o como la ha localizado. Con estos datos intentaremos crear un historial que nos ayude a enfocar la web del modo correcto.

En cuanto a los datos bancarios del cliente, o todos aquellos que tengan que ver con la facturación no están contemplados en esta tabla ya que estos datos no son necesarios para la aplicación y Cervi, S.A. los tiene almacenados en su sistema interno. Durante la primera y segunda fase de desarrollo de la web, la facturación del material adquirido a través de la web seguirá el mismo proceso que el material adquirido de forma “tradicional”.

Este programa presenta la peculiaridad de que contiene un acceso directo al programa de descuentos. Esto simplemente se realizó pensado en la comodidad de los usuarios, ya que de esta manera no deben abandonar la página en la que están trabajando.

8.3. Aplicaciones de soporte.

Para la realización del proyecto únicamente utilizamos dos aplicaciones de soporte. Ambas han sido utilizadas para rellenar la base de datos ecommerce. Una ha insertado diferentes familias de artículos (incluyendo su imagen y su hoja técnica) y la otra par insertar parte de los clientes de CERVI, S.A en la base de datos ecommerce.

Ambas aplicaciones han sido ejecutadas en servidor interno de Cervi, S.A., por lo que en un principio no teníamos acceso directo a la base de datos Ecommerce (ya hemos comentado en el punto 2.4 que el servidor donde se aloja la web está aislado en su propia red virtual). Para solucionar este problema lo que hemos hecho ha sido una base de datos en el servidor donde ejecutábamos el programa y dentro de ella crear una tabla exactamente igual (como el mismo nombre y los mismos campos) a la que tenemos en nuestra base de datos Ecommerce de nuestro servidor. Una vez hecho esto hemos ido rellenándola con los datos necesarios. Este proceso se ha repetido para clientes y artículos.

Una vez llena la tabla hemos generado un archivo que contenía tanto la estructura de la tabla como los datos contenidos en ella con la siguiente instrucción:

```
mysqldump –d ecommerce2 articulos>articulos.sql
mysqldump –d ecommerce2 clientes>clientes.sql
```

Mediante ftp (recordad que este protocolo acepta conexiones de Cervi, S.A. hacia fuera pero no de fuera hacia Cervi, S.A.) hemos transferido este archivo a nuestro servidor.

Al archivo le añadiremos en la cabecera:

```
use ecommerce
drop table articulos / drop table clientes
```

Hacemos esto porque sino el motor de mysql generaría un error porque estamos intentando crear una tabla que ya existe.

Ahora sólo nos queda hacer un volcado de los datos:

```
mysql<articulos.sql
mysql<clientes.sql
```

8.4 Medidas de seguridad

Durante todo el proceso de creación del sistema de gestión, la seguridad ha sido uno de los temas donde se ha hecho más incapié. Hemos intentado crear una aplicación robusta y

segura, que no permitiera manipulaciones maliciosas por parte de los usuarios, así como limitar la funcionalidad a la mínima necesaria.

Es por este motivo que hemos divido la seguridad en tres niveles:

- **La seguridad a nivel de usuario**: Para acceder a la aplicación es necesario validarse con un login y un password que están almacenados en una tabla de la base de datos. Una vez hecho esto creamos una cookie que controla únicamente el tiempo de sesión. Además hemos creado dos usuarios diferentes para la base de datos que condicionan los privilegios de cada usuario de la gestión a la hora de modificar la base de datos.
- **La seguridad a nivel de canal**: Encriptaremos el canal mediante SSL en aquellas aplicaciones susceptibles de manejar información confidencial.
- **La seguridad a nivel de programación**: Comprobaremos tanto a nivel de cliente (con Javascript) como a nivel de servidor (con PHP) que los contenidos de los campos de los formularios son correctos y no contienen cadena de caracteres que pudieran afectar al sistema de forma maliciosa (como por ejemplo sentencias que pudieran alterar la estructura de datos o comandos bash)

8.4.1 La seguridad a nivel de usuario

Cuando accedemos a la dirección del servidor (en nuestro caso 192.168.10.2) la primera página que aparece es un formulario donde se solicita al usuario que introduzca su login y su password.

El nombre de este archivo es ecommerce_ini.php que no es más que un pequeño formulario que nos redirecciona la pagina de validación y de ahí a la página principal si la validación ha sido correcta o de nuevo a este formulario si los datos no son correctos. El código fuente de esta página esta guardado en el CD que se adjunta con esta documentación.

Una vez que hemos introducido los datos en el formulario, la aplicación nos enviará a otra página llamada ecommerce_aut.php que se encargará de comprobar que somos quien decimos ser y generara las cookies que controlan el tiempo de sesión.

Las cookies son unas informaciones almacenadas por un sitio web en el disco duro del usuario. Esta información es almacenada en un archivo tipo texto que se guarda cuando el navegador accede al sitio web. Para un sistema windows suelen almacenarse en C:/Windows/Cookies, que es por defecto el directorio donde se almacenan las cookies.

Aunque las cookies pueden utilizarse para un sinfín de propósitos (estados de variables, fechas de caducidad, apariencia de la web en función del usuario, contadores de visitas, etc...) nosotros únicamente las utilizaremos para controlar el tiempo de sesión.

En el resto de páginas, una vez que se han cargado las librerías, la siguiente línea de código es la comprobación de que la cookie existe. Si no es así, el sistema nos pide que nos volvamos a autentificar. Cabe decir que la cookie sólo existe durante 20 minutos después de haber sido creado por lo que si han pasado más de 20 minutos sin que hayamos hecho nada la cookie desaparecerá y deberemos volver a validarnos. Si la cookie existe, vuelve a generarla de nuevo volviendo a iniciarse la cuenta de 20 minutos. La función que genera la cookie es la siguiente:

```
       function fy_resetcookie (  $cookie_user, $identificador,
$base_de_datos, $usuario_bd, $fy_cookietime )
      {
            setcookie("cookie_user",$cookie_user,time()+$fy_cookietime );
            setcookie("identificador",$identificador,time() +
$fy_cookietime );
            setcookie("base_de_datos",$base_de_datos,time()+$fy_cookietim
e);
            setcookie("usuario_bd", $usuario_bd, time()+$fy_cookietime );
      }
```

El tiempo de caducidad viene especificado por una variable global que hemos definido en la primera autentificación llamada *$fy_cookietime*. Esta variable almacena el tiempo de vida de la cookie.

Es evidente que si tenemos configurado nuestro navegador para que no acepte cookies, el sistema no podrá mostrarse, ya que nos pedirá constantemente que nos validemos porque no encuentra la cookie.

Aún debemos hablar de otro nivel de seguridad aplicado a usuarios. En este caso se trata de los usuarios con los que se ejecutan sentencias en el demonio de MySQL. Actualmente Mysql tiene definido tres usuarios en su tabla users (tabla que se crea por defecto). El primer usuario, **root**, se crea automáticamente cuando se crea el motor. Este usuario tiene todos los privilegios disponibles por lo que puede modificar tanto los datos como la estructura de la base de datos.

Los usuarios que hemos creado nosotros son:

- administrador: puede modificar datos pero no estructura por lo que los privilegios para borrar, crear o modificar tablas han sido suprimidos y también aquellos que modifican, crean o eliminan índices.
- visitante: Es cliente sólo tiene permisos a nivel de consulta, es decir, que únicamente puede ejecutar sentencias SELECT.

A cada usuario creado en la gestión se le ha asignado uno de estos dos usuarios de la base de datos y cuando un visitante intenta modificar algún registro el sistema le informa de que no tiene los permisos necesarios.

La identidad del usuario de la base de datos se controla en dos puntos del programa. Primero, en una función de javascript que advierte al usuario mediante un mensaje de que

no tiene permisos de ejecución y segundo, en los módulos php interrumpiendo la ejecución de programa.

A continuación aparece el código fuente de la validación que se hace en la función de javascript. En este punto del programa se comprueba el contenido de la cookie asociada al objeto document.

```
            if ( document.cookie.substring( document.cookie.indexOf( “=” ) + 1 ), document.cookie.indexOf( “;” ) ) <> ‘administrador’)
            {
                  alert( “No tiene los permisos necesarios para realizar esta operación” );
                  return;
            }
```

En el siguiente trozo de código fuente podemos ver esta misma validación en PHP.

```
            if ( $usuario_bd !=  “administrador” )
            {
                  echo “<br><center><b>NO TIENE PERMISOS DE EJECUCION, CONTACTE CON EL ADMINISTRADOR</b></center><BR>”;
            }
```

8.4.2. La seguridad a nivel de canal

La seguridad a nivel de canal será implementada utilizando el protocolo de encriptación SSL, en especial el que ofrece Apache-SSL, que es un servidor web seguro basado en apache y Open SSL. Es de código libre por lo que es gratuito. La encriptación del canal se hace con claves de 128 bits.

El protocolo SSL es un sistema diseñado y propuesto por Netscape Communications Corporation. Se encuentra en la pila OSI (por lo tanto esta estandarizado) entre los niveles de TCP/IP y de los protocolos HTTP, FTP, SMTP, etc. Proporciona sus servicios de seguridad cifrando los datos intercambiados entre el servidor y el cliente con un algoritmo de cifrado simétrico, típicamente el RC4 o IDEA, y cifrando la clave de sesión de RC4 o IDEA mediante un algoritmo de cifrado de clave pública, típicamente el RSA. La clave de sesión es la que se utiliza para cifrar los datos que vienen del y van al servidor seguro. Se genera una clave de sesión distinta para cada transacción, lo cual permite que aunque sea reventada por un atacante en una transacción dada, no sirva para descifrar futuras transacciones. MD5 se usa como algoritmo de hash.

Proporciona cifrado de datos, autenticación de servidores, integridad de mensajes y, opcionalmente, autenticación de cliente para conexiones TCP/IP.

Para que una página sea cifrada con SSL únicamente debemos llamarla de la siguiente manera:

https://192.168.10.2/ecommerce_aut.php

En esta primera fase no hemos creído conveniente instalar este servidor en concreto por lo que hemos seguido trabajando con un servidor normal. Cabe recordar que en esta primera fase todavía no se transmiten datos confidenciales.

Cuando se llegue a la fase final, donde será efectuar transacciones bancarias, no será suficiente con contar con un servidor seguro por lo que se hará necesario adquirir un certificado digital.

8.4.3. La seguridad a nivel de programación

En este apartado hemos intentado evitar que el usuario pudiera entrar a través de los formularios información maliciosa como por ejemplo palabras que pudieran alterar las sentencias de MySQL o que el sistema pudiera entender como comandos bash. Por este motivo se ha fijado de forma expresa que valores o caracteres puede contener un campo y cuales no. Esto está además en función del propósito del campo. Es evidente que el rango de caracteres permitidos es mayor para un campo debe almacenar una descripción que el de un campo que debe almacenar una fecha.

La primera medida que hemos aplicado ha sido comprobar la extensión y los nombres de los ficheros que eran almacenados en el servidor mediante la gestión de articulos, o la gestión de ofertas. Las limitaciones impuestas han sido las siguientes:

- Las extensiones permitidas son: jpg, JPG, gif, GIF, pdf, PDF.
- El nombre del archivo no puede contener valores como “/”, “\”, “*”, etc..

Para comprobar estos datos hemos utilizado las siguientes funciones de php:

```
function copiar_archivo ( $name, $archivo )
{
      global $cargar_body;
      if ( $name!=  ‘’ )
      {
            $extension = explode( “.”, $name );
```

explode divide $name en dos valores (nombre y extension) que almacena en el array $extension ($extension[1] y $extension[2])

```
                  if ( comprovacion_nombre ( $name ) )
                  {
                        $cargar_body= “onload = ‘alert( \”El nombre del
archivo contiene caracteres prohibidos, solo se permiten letras y
numeros.\”)’;”;
                  }
```

```
                else
                {
                        if ( $extension[1] != 'gif' && $extension[1] !=
'jpg' && $extension[1] != 'jpeg' && $extension[1] != 'GIF' &&
$extension[1] != 'JPG' && $extension[1] != 'JPEG' && $extension[1] |=
'jpg' && $extension[1]  !=  'pdf' && $extension[1] != 'PDF' )
                        {
                                $cargar_body = "onload='alert(\"La
extension del fichero es incorrecta, solo se permiten las extensiones
jpg, jpeg, gif y pdf\");'";
                        }
                        else
                        {
                                if ( !file_exist("../IMAGES/" . $name ) )
                                {
                                        if ( !copy( $archivo,  "../IMAGES/" .
$name ) )
                                        {
                                                $cargar_body =
"onload='alert(\"Ha sido imposible copiar el archivo\");'";
                                        }
                                }
                        }
                }
        }
```

```
        function comprovacion_nombre( $name )
        {
                $chOk=".0123456789abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑO
PQRSTUVWXYZ_";
                for ( $i = 0; $i < strlen( $name ); $i++ )
                {
                        $a = substr( $name, $i, 1 )
                        {
```

###Busca cada caracter que compone el nombre en la cadena de valores permitidos, si no lo encuentra devuelve un false.

```
                        for ( $j = 0; $j < strlen( $chOk ); $j++ )
                        {
                                $caracter = substr( $chOk, $j, 1 );
                                if ( $caracter == $a  )
                                {
                                        break;
                                }
                        }
                        if ( $j == strlen( $chOk ) )
                        {
                                return( false );
                        }
                }
```

```
        }
        return( true);
}
```

Esta función devuelve un true si el nombre es correcto y false si es incorrecto. Si esto sucede, se detiene la ejecución del programa.

Para el resto de los campos de todos los programas se ha utilizado una función similar que compara una cadena de caracteres permitidos con el valor del campo, si no encuentra ningún valor no permitido da el contenido por válido y continua con la ejecución del programa. Si encuentra algún valor permitido detiene la ejecución del programa.

Antes de utilizar esta medida tan drástica, se le avisa al usuario de que está utilizando caracteres prohibidos controlando el contenido de los campos mediante javascript. La estructura es muy similar a la función de php por lo que no la repetiremos de nuevo, solo hay que utilizar la sintaxis de este lenguaje.

Esto implica que tenemos dos barreras de programación para realizar el control:

- La primera en Javascript que un usuario experimentado puede burlar, pero más amigable para el usuario.
- La segunda en Php, que no es modificable ya que se ejecuta en el servidor.

8.5. Diferencias con los estandars

Aunque hemos intentado ajustarnos en todo lo posible a los estandars establecidos ha habido momentos en los que nos ha sido imposible. A continuación detallamos estos casos.

En primer lugar, dada la apariencia que hemos querido darle a la gestión de la base de datos, hemos usado un atributo no estandar para las celdas de las tablas: el background.

```
<td background="../IMAGES/tablita_02.gif">…
```

Al validar el código html en el validador que aparece en *www.w3.org* nos dice que este no es un atributo válido para esta etiqueta. Aún así, hemos visualizado la gestión en diferentes navegadores (IE6, Netscape, Opera, Mozilla) y el resultado ha sido en todos el esperado.

Otro punto donde no nos hemos ceñido a los estandars es al utilizar la etiqueta **<NOBR>,** , pero al igual que en caso anterior el resultado ha sido el esperado en todos los navegadores.

También hablaremos de las diferencias que hemos tenido que contemplar en la programación con Javascript, ya que en artículos hemos creado diversos trozos de código en función del navegador.

Internet Explorer tiene un tipo de ventana específico llamado ShowModalDialog, cuya principal peculiaridad es que devuelve un valor cuando se cierra, característica muy útil para ciertas aplicaciones. El problema radica en que este tipo de ventana es únicamente de Internet Explorer por lo que los demás navegadores no la abren. En los casos en los que hemos usado este tipo de ventana, primero hemos detectado el navegador y si era IE la usábamos y si era Nestcape, Mozilla o Opera abríamos una ventana normal.

BLOQUE IV

Creación del Web Site

9. Web Site

En este punto explicaremos como hemos estructurado la web de Cervi, S.A. así como los diferentes apartados que la componen. Hablaremos también sobre el procedimiento que queremos establecer en la segunda fase de desarrollo de la web.

Tanto el diseño como la apariencia de la página web están sujetos a la opinión final de Cervi, S.A. así como a la opinión de los clientes, por lo que pueden darse a lo largo del proceso diversas modificaciones.

9.1. La portada

La portada y algunas secciones de la web (Web técnica, Quienes somos) han sido desarrolladas con el programa DreamWeaver y las imágenes han sido tratadas con Photoshop 7 (vease [Nic 2003]). El buscador ha sido desarrollado directamente escribiendo el código ya que necesitábamos trabajar con el servidor PHP que está en la máquina linux.

Nuestra portada o página principal está estructurada en dos frames o marcos, es decir, dividida en dos partes horizontales. El marco superior contiene un pequeño menú desplegable desde donde se puede acceder a todas las secciones que forman nuestra web.

Este menú desplegable ha sido desarrollado con un software específico llamado Sothink, cuya principal utilidad es la de crear diferentes tipos de menús desplegables. Este hecho ha simplificado sobre todo la parte de diseño gráfico, ya que este programa te propone una serie de diferentes apariencias, de las cuales una de ellas hemos adaptado a nuestras necesidades.

Los menús principales que aparecen son :

- Productos.
- Web técnica
  - Reglamento ICT
  - Esquemas
- Cervi
  - Quienes Somos
  - Contacte con Nosotros
- Clientes
- Noticias
- Eventos
- Home

Estos menús tienen asociados unos vínculos que redireccionan al usuario a la sección deseada. Los archivos html de estas secciones se descargan en el marco inferior del que hablaremos a continuación.

En marco inferior encontramos una tabla con cuatro imágenes, cada una de ellas actúa de enlace a una de las secciones principales: Productos, Cervi, Clientes y Web técnica. Es, por lo tanto, un menú totalmente gráfico. Los archivos html de estas secciones se descargan en este mismo marco.

Estas imágenes presentan algunas peculiaridades, ya que han sido sometidas a algunos efectos especiales. Por ejemplo, al situar el ratón encima de una de estas imágenes, se cambia el valor del atributo src de la etiqueta img, esto significa que se carga otra imagen en la web. Este efecto es conocido como rollover, ya que se turnan dos imágenes en función de si tienen encima el ratón o no.

También se realiza el mismo efecto sobre otras imágenes que no se ven en un primer momento. Estas imágenes contienen la descripción de cada una de las secciones, por lo que funciona con el mismo efecto que las imágenes principales. Este concepto quedará mejor explicado en el siguiente gráfico.

onMouseOut
Raton fuera

onMouseOver
Ratón encima

onMouseOut
Ratón fuera

onMouseOver
Ratón encima

Para poder aplicar estos efectos sobre las imágenes hemos creado diferentes objetos Image donde hemos establecido un valor concreto para su atributo src, que almacena el path de la imagen. Para esto a cada imagen se le ha asignado un nombre (name) para poder diferenciarla del resto. Una vez hecho esto, lo único que queda por hacer es cambiar el valor del atributo src.

A continuación mostramos el código fuente de las funciones javascript que se ejecutan cuando se lanza el evento onMouseOver (cuando el ratón está encima).

```
function On(imgName,descripcion)
{
      if (document.images)
      {
            imgOn = eval (imgName + "_on.src");
            document.images[imgName].src = imgOn;
            deson = eval (descripcion + "_on.src");
            document.images[descripcion].src = deson;
       }
}

function Off(imgName,descripcion)
{
      if (document.images)
      {
            imgOff = eval (imgName + "_off.src");
            document.images[imgName].src = imgOff;
            desoff = eval (descripcion + "_off.src");
            document.images[descripcion].src = desoff;
      }
}
```

Por último en la portada aparecen, en cumplimiento de [LSS], los datos identificativos de Cervi, S.A., así como un enlace hacia una página donde explicamos nuestra política de protección de datos. En el Anexo II describimos más detalladamente que leyes debemos cumplir.

Todas las paginas pasan las especificaciones del Word Wide Web Consortium.

9.2. Quienes somos

Esta sección está formada por dos archivos html. En el primero aparece una breve explicación de la historia de Cervi, S.A., así como cual es el sector económico en el que realiza sus actividades.

En el segundo archivo aparecen todas las direcciones de las diferentes delegaciones de Cervi, S.A., así como enlaces a las respectivas direcciones de correo electrónico.

9.3. Web Técnica

La web técnica esta divida igualmente en dos subsecciones: Reglamento ICT (Infraestructuras Comunes de Telecomunicación) y Esquemas (esquemas de cables y latiguillos).

Dentro de la sección de ICT encontramos los siguientes 7 apartados :

- **Real Decreto 279/1999** :Real Decreto 279/1999 ,de 22 de Febrero, por el que se aprueba el reglamento regulador de las infraestructuras comunes de telecomunicaciones para el acceso a los servicios de telecomunicaciones en el interior de los edificios y de la actividad de instalación de equipos y sistemas de telecomunicaciones.
- **Reglamento ICT** : Reglamento regulador de las infraestructuras comunes de telecomunicaciones para el acceso a los servicios de telecomunicaciones en el interior de los edificios y de la actividad de instalación de equipos y sistemas de telecomunicaciones.
- **Anexo I :** Norma Técnica de infraestructura común de telecomunicaciones para la captación de señales de radiodifusión sonora y televisión ,procedentes de emisiones terrenales y de satélite.
- **Anexo II :** Norma técnica de infraestructura común de telecomunicaciones para el acceso al servicio de telefonía disponible al público.
- **Anexo III** : Norma técnica de la infraestructura común de telecomunicaciones para el acceso al servicio de telecomunicaciones por cable.
- **Anexo IV :** Especificaciones Técnicas mínimas de las edificaciones en materia de telecomunicaciones.
- **Decreto 172/1999 del 29 de junio de la Generalitat de Cataluña** : Documento en pdf sobre canalizaciones e infraestructuras de radiodifusión sonora, televisión, telefonía básica y otros servicios por cable en los edificios .

Estos documentos contienen planos y esquemas. Para mejorar la visualización de éstos, hemos añadido a cada plano y esquema un vínculo el cual nos lleva a otra página que contiene el plano o el esquema en cuestión pero de mayor tamaño. Estos planos y esquemas forman un total de 13 archivos html.

También se incluye en esta sección un apartado donde aparecen algunos enlaces a diferentes cursos de formación que Cervi, S.A. imparte. Estos enlaces no han sido activados porque el material recogido es de año anteriores y ha habido desde entonces algunas modificaciones en las respectivas legislaciones.

La segunda parte de la web técnica tiene como objetivo aclarar al visitante ciertos aspectos técnicos elementales de algunos productos básicos, en este caso hemos introducido dos enlaces a esquemas del cable cruzado CAT.5 UTP y al código de colores del UTP/FTP.

9.4. El buscador y el catálogo de artículos

El buscador está formado por dos archivos. El primero, ***productos_buscador.php***, contiene el formulario con las entradas para los tres tipos de búsqueda:

- Búsqueda por palabra clave: Aparece un listado de las diferentes palabras claves existentes. Podemos seleccionar tantas como queramos. Hay una pequeña función en javascript que las va concatenando en una cadena de caracteres que es pasada como parámetro para la consulta con MySQL.

  ```
  function Sel_Palabras( Obj, NomForm )
  {
          if ( Obj.value != '' )
          {
                NomForm.palabras_sel.value+='"'+Obj.value+'",';
          }
  }
  ```

  Esta función únicamente concatena palabra y los almacena en el atributo value de un input.
- Busqueda por Fabricante: Se debe seleccionar un fabricante de los que muestra la lista. Este el único campo obligatorio (si no se ha rellenado el campo artículo), a partir de aquí el resto de campos son optativos. Si se rellena el campo Artículo únicamente se tiene en cuenta este campo para la consulta a la base de datos, por lo que el programa ignorará el resto de los campos.
- Búsqueda por categoría: El funcionamiento es igual que el de la búsqueda anterior. El campo categoría es obligatorio si no se ha rellenado el campo articulo.

En estos dos tipos de búsqueda hay pequeñas funciones en javascript que se encargan de rellenar los selects inferiores. Por ejemplo, cuando seleccionamos un fabricante provocamos un evento que aprovechamos para recargar la página. En este momento el valor de la variable ***$fabricante*** no está vacío, por lo que cuando volvemos al cargar todo el código html y llegamos al select de las familias tenemos:

```
<select name=fam1 class=casillas onChange='SelModo("A",this,Formato);'>
  <option value=''>--------------------------------</option>
    <?
      if ( $fabricante != '' )
      {
            $sql="select * from familias where fabricante='$fabricante'
order by familia";
            $result_set=fy_execsql($fydb_type,$fydb_name,$conn,$sql);
            $rows=fy_resultrows( $fydb_type, $result_set );
            for ( $i = 0; $i < $rows; $i++ )
            {
                  $fami=fy_getfield($fydb_type,$result_set,$i,'familia');
```

```
                $desc=fy_getfield($fydb_type,$result_set,$i,descripcio)
                if ( $fami == $familia )
                {
                      echo "<option value='" . $fami . "' selected>" .
$desc . "</option>";
                }
                else
                {
                      echo "<option value='" . $fami . "'>" . $desc .
"</option>";
                }
            }
      }
   ?>
</select>
```

El proceso es idéntico para el resto de los selects.

Este archivo nos redirecciona al archivo que nos muestra el resultado de la búsqueda, ***lista_resultados.php***, donde aparece un listado de los diferentes artículos, con una foto y una pequeña descripción de cada uno. Además hay un enlace para cada articulo que despliega una ventana con información adicional sobre éste.

Este archivo contiene una paginación que muestra sólo 15 artículos por pantalla, por lo que podemos ir hacia adelante o hacia atrás.

El conjunto de estos dos archivos es lo hemos llamado CATÁLOGO DIGITAL.

A este programa se le ha añadido una paginación de resultados por lo que si el resultdao de la búsqueda es superior a 10 articulos, los mostrará en diferentes páginas permitiendonos navegar por ellas con enlaces de “anterior” y “siguiente”.

El código necesario es:

```
        if ( !isset( $pg ) )
        {
                $pg = 0;
        }
        $cantidad = 10;
        $inicial = $pg *  $cantidad;
```

##Inicial el numero de fila por donde queremos que empiece el siguiente result_set

```
        [...]
        echo "<p align=center>";
            if ( $pg != 0 )
            {
                  $url_ant = $pg - 1;
                  if ( $Modo == 'S' )
```

```
                {
                     $cadena = "lista_resultados.php? Modo=S
&fabricante=" . $fabricante . "&familia=". $familia . "&subfamilia=" .
$subfamilia . "&articulo=" . $articulo . "&pg=" . $url_ant;
                }
                if ( $Modo == 'B' )
                {
                     $cadena = “lista_resultados.php ?Modo=B &
palabras_sel=" . $palabras_sel . "&pg=" . $url_ant;
                }
                if ( $Modo == 'P' )
                {
                     $cadena = lista_resultados.php?Modo=P&categoria="
. $categoria . "&clase=" . $clase . "&subclase=" . $subclse .
"&articulo2=" . $articulo2 . "&pg=" . $url_ant;
                }

                echo "<a href='".$cadena."' target='_self'>
&laquo;Anterior</a>";
          }
          echo "   ";
          if ( $pg < $paginas )
          {
                $url_sig = $pg + 1;
                if ( $Modo == 'S' )
                {
                       $cadena = “lista_resultados.php?Modo=S
&fabricante=" . $fabricante . "&familia=". $familia . "&subfamilia=" .
$subfamilia . "&articulo=" . $articulo . "&pg=" . $url_sig;
                }
                if ( $Modo == 'B' )
                {
                       $cadena = “lista_resultados.php?Modo=B&
palabras_sel=" . $palabras_sel . "&pg=" . $url_sig;
                }
                if ( $Modo == 'P' )
                {
                       $cadena = “lista_resultados.php?Modo=P&
categoria=" . $categoria . "&clase=" . $clase . "&subclase=" . $subclse .
"&articulo2=" . $articulo2 . "&pg=" . $url_sig;
                }
                echo "<a href='" . $cadena . "' target='_self'>
Siguiente &raquo;</a>";
          }
          echo "</p>";
```

Creamos los links que nos llevarán a la página siguiente y a la página anterior.

9.5. Zona de acceso de clientes

Esta zona de acceso sigue la misma filosofia que la autentificación de usuarios de la aplicación de gestión. Cada cliente tiene un login (que inicialmente es el número de cliente) y un password generado de manera aleatoria de 8 caracteres que contiene números y letras mayúsculas y minúsculas.

Si el cliente es validado como tal, la aplicación lo envia directamente al catálogo de productos y si la validación es incorrecta le pide que vuelva a introducir los datos.

Este programa no trabaja con cookies aunque sería conveniente que el catálogo de artículos supiera cuando el cliente se ha validado como tal y cuando no. Para este propósito las cookies serían muy útiles.

BLOQUE V

Conclusiones y Trabajo futuro

10. Conclusiones.

Para que una página web dinámica ofrezca las máximas funcionalidades posibles todo lo que hay detrás de esta web, es decir, una base de datos potente y un buen sistema de gestión, debe estar bien diseñado. Es por este motivo por el que consideramos que el punto fuerte de este proyecto es el sistema de gestión. Por tanto, es la parte a la que hemos dedicado más tiempo y donde realmente hemos aplicado todo lo aprendido durante la realización del proyecto.

Esto nos ha llevado, no tan sólo a dedicarle muchas horas de programación en diferentes lenguajes, sino también a dedicarle muchas horas para establecer las funcionalidades más necesarias e útiles que debían ofrecer nuestras aplicaciones, de manera que además de eficaces fueran sencillas de utilizar

Es importante remarcar que aunque la programación con html no es complicada, la dificultad aumenta cuando debes ceñirte a los estandars y contemplar diferentes navegadores. En este sentido fue muy útil la documentación recogida en [W3C]. También dificulta la programación el hecho de usar javascript, pero es un esfuerzo que vale la pena realizar si con ello se consiguen webs más interactivas y por lo tanto más interesantes.

Nos gustaría comentar que aunque consideramos que Mysql es un motor bastante potente y se adecua perfectamente a las necesidades del proyecto es posible que en un futuro nos convenga migrar a otro tipo de gestor, sobre todo a medida que vaya creciendo la base de datos, ya que hemos comprobado que Mysql pierde mucha eficacia a la hora de manejar grandes archivos.

Queremos comentar, además, que hemos cumplido los objetivos establecidos en este proyecto, teniendo en cuenta que la creación completa de un Web Site es muy compleja y laboriosa creemos que hemos conseguido establecer los cimientos del proyecto global. Hemos diseñado y creado una base de datos, hemos diseñado y creado el sistema de gestión de la base de datos y hemos diseñado el web site así como un catálogo digital accesible desde la web.

A partir de este momento el proyecto de Cervi, S.A. debería centrarse en implementar el carro de la compra y en mejorar la presentación gráfica de la web. En este punto nos hemos esforzado todo lo que hemos podido, pero el tiempo no da para más.

Nos gustaría agradecer a Cervi, S.A. su paciencia y apoyo a la hora de realizar este proyecto.

11.Trabajo futuro

La continuación de este proyecto giraría entorno a dos grandes bloques:

- Implementación de un carrito de la compra:
  - Crear pedidos nuevos.
  - Añadir productos a pedidos ya existentes.
  - Consultar pedidos ya existentes.
  - Aceptación de los pedidos por parte de Cervi, S.A.: Quizá sería conveniente establecer unos límites dentro de los cuales el pedido puede ser aceptado de forma inmediata. Podría limitarse por ejemplo el importe económico del pedido de modo que cualquier pedido de importe superior al límite establecido requiera una autorización expresa por parte de Cervi, S.A.
  - Protocolos de comunicación entre Cervi, S.A. y sus clientes: Debería establecerse el canal que se utilizaría para que la comunicación fuera fluida. Nosotros recomendaríamos que se usara el correo electrónico ya que es un medio que esta al alcance de todo el mundo y suele ser el más utilizado para estos propósitos.

- Apertura de la web a todos los usuarios de la red.
  - Funcionamiento de transferencias bancarias: Aunque este punto no requiere mucha programación es de suma importancia. Debería seleccionarse un banco que ofreciera tranquilidad.
  - Nuevo estudio de seguridad: Una vez que se han puesto en funcionamiento las transacciones bancarias no es suficiente con utilizar un servidor seguro. Deben cumplirse algunas especificaciones técnicas y es recomendable (aunque no obligatorio) certificarse, es decir obtener una clave digital para encriptar estas transacciones.
  - Estudios estadísticos.

Aunque esto son solo ideas generales cada uno de estos bloques podría ser considerado como un proyecto, que a su vez se correspondería con cada una de las fases que definimos en el capítulo 1.

BLOQUE VI

Bibliografía

12. Bibliografía

[APA]	www.apache.org
[CSS2]	www.w3.org/TR/CSS2/
[DEW]	www.desarrolloweb.com
[DIN]	www.dynamicdrive.com (2003-11-05)
[Groff 1990]	Aplique SQL Ed Mc Graw Hill James R. Groff y Paul N. Weinberg, 1990
[HTML4]	www.w3.org/TR/html4/
[JSC1]	devedge.netscape.com/library/manuals/2000/javascript/1.3/guide/ ClientGuideJS13.pdf (2003-10-30)
[JSC2]	devedge.netscape.com/library/manuals/2000/javascript/1.4/reference / CoreReferenceJS14.pdf (2003-10-30)
[JSC3]	devedge.netscape.com/library/manuals/2000/javascript/1.4/guide/ CoreGuideJS14.pdf (2003-10-30)
[LOP]	www.lopd.com Ley Orgánica de Protección de Datos. (2003-11-05)
[LSS]	www.lssice.com/legislacion/lssice.html Ley de la servicios de la sociedad de información y comercio electrónico. (2004-01-07)
[MYS]	www.mysql.com Consultada repertidamente
[Nic 2003]	Photoshop 7 Ed Anaya Multimedia Nicolás Sanchez-Biezma y Luis Monje, 2003

[PHP]	www.php.net Consultada repetidamente
[Pow 2002]	Manual de Referencia Javascript Ed Mc Graw Hill Tomas Powell y Fritz Schneider, 2002.
[SEN]	www.sendmail.org (2003-11-05)
[SGC]	www.sgc.mform.es Ministerio de Ciencia y Tecnología
[SQL]	www.ncb.ernet.in/education/modules/dbms/SQL99 Standards SQL 1995 ANSI (2003-11-25)
[W3C]	www.w3.org A parte de consulta diferentes especificaciones se utilizó el validador de hojas html que puede encontrarse en la siguiente dirección: - validator.w3.org

ANEXOS

ANEXO I
ESQUEMA DE LA BASE DE DATOS ECOMMERCE

```
# MySQL dump 6.0
#
# Host: localhost    Database: ecommerce
#--------------------------------------------------------
# Server version        3.22.25-log
#
# Table structure for table 'actos'
#

CREATE TABLE actos (
  acto int(4) DEFAULT '0' NOT NULL,
  nombre char(50) DEFAULT '' NOT NULL,
  fecha_inicio date DEFAULT '0000-00-00' NOT NULL,
  fecha_fin date DEFAULT '0000-00-00' NOT NULL,
  provincia char(50) DEFAULT '' NOT NULL,
  poblacion char(50) DEFAULT '' NOT NULL,
  direccion char(50) DEFAULT '' NOT NULL,
  codigo_postal char(6) DEFAULT '' NOT NULL,
  url char(50) DEFAULT '' NOT NULL,
  mail char(50) DEFAULT '' NOT NULL,
  descripcion char(100) DEFAULT '' NOT NULL,
  PRIMARY KEY (nombre)
);

CREATE TABLE articulos (
  articulo char(8) DEFAULT '' NOT NULL,
  nombre char(30) DEFAULT '' NOT NULL,
  path_imagen char(50),
  precio decimal(8,2),
  precio_coste decimal(8,2),
  descrip_comp char(250),
  tipo_articulo char(1) DEFAULT '' NOT NULL,
  unidades char(2) DEFAULT '' NOT NULL,
  plazo_entrega int(2) DEFAULT '0' NOT NULL,
  palabra_clave1 char(40) DEFAULT '' NOT NULL,
  palabra_clave2 char(40) DEFAULT '' NOT NULL,
  palabra_clave3 char(40) DEFAULT '' NOT NULL,
  path_ficha_tecnica char(50),
  path_ficha_tecnica2 char(50),
  dispo_tienda char(1) DEFAULT '' NOT NULL,
  subfamilia int(6) DEFAULT '0' NOT NULL,
  categoria int(6) DEFAULT '0' NOT NULL,
  PRIMARY KEY (articulo)
);


CREATE TABLE cab_pedidos (
  pedido mediumint(9) DEFAULT '0' NOT NULL,
  cliente mediumint(9) DEFAULT '0' NOT NULL,
```

```
  domicilio char(50) DEFAULT '' NOT NULL,
  nombre char(50) DEFAULT '' NOT NULL,
  poblacion char(50) DEFAULT '' NOT NULL,
  codpos char(6) DEFAULT '' NOT NULL,
  provincia char(50) DEFAULT '' NOT NULL,
  pais char(50) DEFAULT '' NOT NULL,
  cif char(11) DEFAULT '' NOT NULL,
  fecha date DEFAULT '0000-00-00' NOT NULL,
  total_pedido decimal(11,2) DEFAULT '0.00' NOT NULL,
  total_coste decimal(11,2) DEFAULT '0.00' NOT NULL,
  aceptado tinyint(4) DEFAULT '0' NOT NULL,
  PRIMARY KEY (pedido)
);

CREATE TABLE categorias (
  parent char(5) DEFAULT '' NOT NULL,
  descripcion char(30) DEFAULT '' NOT NULL,
  nombre char(5) DEFAULT '' NOT NULL,
  PRIMARY KEY (nombre)
);


CREATE TABLE clientes (
  cliente mediumint(9) DEFAULT '0' NOT NULL,
  login char(8) DEFAULT '' NOT NULL,
  password char(40) DEFAULT '' NOT NULL,
  web char(50) DEFAULT '' NOT NULL,
  domicilio char(50) DEFAULT '' NOT NULL,
  nombre char(50) DEFAULT '' NOT NULL,
  poblacion char(50) DEFAULT '' NOT NULL,
  cos_pas char(6) DEFAULT '' NOT NULL,
  provincia char(50) DEFAULT '' NOT NULL,
  pais char(50) DEFAULT '' NOT NULL,
  cif char(11) DEFAULT '' NOT NULL,
  num_visitas smallint(6) DEFAULT '0' NOT NULL,
  acep_prop tinyint(4) DEFAULT '0' NOT NULL,
  forma_loc char(40) DEFAULT '' NOT NULL,
  fecha_alta date DEFAULT '0000-00-00' NOT NULL,
  e_mail char(50),
  PRIMARY KEY (cliente,login)
);

CREATE TABLE descuentos (
  cliente mediumint(9) DEFAULT '0' NOT NULL,
  familia smallint(6) DEFAULT '0' NOT NULL,
  descuento tinyint(4) DEFAULT '0' NOT NULL,
  PRIMARY KEY (familia,cliente)
);

CREATE TABLE fabricantes (
  fabricante smallint(6) DEFAULT '0' NOT NULL,
  descripcion char(50) DEFAULT '' NOT NULL,
  PRIMARY KEY (fabricante)
);
```

```
CREATE TABLE familias (
  familia smallint(6) DEFAULT '0' NOT NULL,
  descripcion char(50) DEFAULT '' NOT NULL,
  fabricante smallint(6) DEFAULT '0' NOT NULL,
  PRIMARY KEY (familia)
);

CREATE TABLE lin_pedidos (
  pedido mediumint(9) DEFAULT '0' NOT NULL,
  articulo char(8) DEFAULT '' NOT NULL,
  descripcion char(50) DEFAULT '' NOT NULL,
  cantidad smallint(6) DEFAULT '0' NOT NULL,
  precio decimal(8,2) DEFAULT '0.00' NOT NULL,
  precio_coste decimal(8,2) DEFAULT '0.00' NOT NULL,
  descuento tinyint(4) DEFAULT '0' NOT NULL,
  precio_linea decimal(8,2) DEFAULT '0.00' NOT NULL,
  precio_coste_linea decimal(8,2) DEFAULT '0.00' NOT NULL,
  almacen char(2) DEFAULT '' NOT NULL,
  plazo_entrega tinyint(4) DEFAULT '0' NOT NULL,
  linea tinyint(4) DEFAULT '0' NOT NULL
);

CREATE TABLE localizacion (
  forma char(50) DEFAULT '' NOT NULL
);


CREATE TABLE menus (
  path char(50),
  path_imagen char(50)
);

CREATE TABLE numeradores (
  codigo int(3) DEFAULT '0' NOT NULL,
  descripcion char(50) DEFAULT '' NOT NULL,
  valor int(7) DEFAULT '0' NOT NULL,
  KEY i_num (codigo)
);

CREATE TABLE ofertas (
  oferta int(6) DEFAULT '0' NOT NULL,
  articulo char(8) DEFAULT '' NOT NULL,
  precio decimal(11,2) DEFAULT '0.00' NOT NULL,
  fecha_fin date DEFAULT '0000-00-00' NOT NULL,
  imagen_publicidad char(75) DEFAULT '' NOT NULL,
  cond_esp char(50),
  PRIMARY KEY (oferta)
);


CREATE TABLE paises (
  pais char(100)
);
```

```
CREATE TABLE palabras_claves (
  palabra_clave char(40) DEFAULT '' NOT NULL
);


CREATE TABLE subfamilias (
  subfamilia smallint(6) DEFAULT '0' NOT NULL,
  descripcion char(50) DEFAULT '' NOT NULL,
  familia smallint(6) DEFAULT '0' NOT NULL,
  PRIMARY KEY (subfamilia)
);


CREATE TABLE usuarios_gestion (
  usuario int(3) DEFAULT '0' NOT NULL,
  login char(15) DEFAULT '' NOT NULL,
  password char(32) DEFAULT '' NOT NULL,
  nombre char(60) DEFAULT '' NOT NULL,
  email char(60) DEFAULT '' NOT NULL,
  cargo char(30) DEFAULT '' NOT NULL,
  fecha_alta date DEFAULT '0000-00-00' NOT NULL,
  fecha_baja date DEFAULT '0000-00-00' NOT NULL,
  bd_user char(15) DEFAULT '' NOT NULL,
  PRIMARY KEY (login,password),
  KEY usuarios_t1 (usuario)
);
```

ANEXO II
Leyes y legislaciones.

Dadas las características de este proyecto debemos ceñirnos a dos leyes:

- La LSSICE: Ley de Servicios de la Sociedad de Información y Correo Electrónico.
- La LOPD: Ley Orgánica de Protección de Datos.

LSSICE (Ley de Servicios de la Sociedad de Información y Correo Electrónico)

El objeto de la LSSICE es la regulación del régimen jurídico de los servicios de la sociedad de la información y la contratación por vía electrónica.

Los destinatarios de esta regulación son los prestadores de servicios de la sociedad de la información establecidos en España y los servicios prestados por ellos. Entendemos por prestador de servicios establecido en España aquel cuya residencia o domicilio social se encuentra en territorio español siempre y cuando desde este domicilio social se realice la gestión administrativa y la dirección de sus negocios.

En primer lugar esta ley establece un régimen de libre prestación de servicios en España, por lo tanto, no es necesaria la obtención de ninguna autorización previa.

Esta ley establece claramente, que tipo de información genérica debe mostrarse en la página web.

Esta ley puede consultarse en [LSS].

Obligaciones genéricas de información.

- Nombre o denominación social, domicilio, correo electrónico o cualquier otro dato necesario para poder establecer el contacto directo. En caso de no cumplimiento la sanción puede ser una multa de entre 30001 € y 150000 € al considerarse una falta grave.
- Datos de la inscripción registral, en nuestro caso en el Registre Mercantil.
- El número de identificación fiscal correspondiente.
- Identificación clara y exacta sobre el precio del producto indicando los impuestos correspondientes y los gastos de envío si fueran necesarios. En caso de no cumplimiento la sanción puede ser una multa de entre 30001 € y 150000 € al considerarse como una falta grave.

LOPD (Ley Orgánica de Protección de Datos)

Desde la entrada en vigor de la LOPD (para más información consultar [LOP]), puede afirmarse que todo fichero que almacene datos personales de personas físicas identificadas o identificables se encuentra dentro del ámbito de aplicación de la normativa, y sometido por tanto a los principios, requisitos y régimen que mediante ella se instaura.

La creación del fichero o base de datos

La creación de una base de datos o fichero que contenga datos de personas físicas debe responder a una finalidad concreta y legítima. No en vano esa finalidad deberá comunicarse a la Agencia de Protección de Datos.

De esta forma la Ley obliga, a quien pretenda la creación de una base de datos o fichero de estas características, que con anterioridad a su puesta en marcha prevea una serie de elementos que, básicamente, consisten en determinar la finalidad a que obedece el fichero y el nivel de medidas de seguridad que, desde el ámbito técnico, han de implantarse en función de la naturaleza de los datos.

La recogida de datos

Una vez que se ha notificado a la Agencia la creación del fichero, y determinadas las medidas de seguridad aplicables, el siguiente paso crítico es la recogida de datos en el que debe informarse al interesado sobre los derechos que le asisten, con las excepciones previstas en la Ley y las especialidades que igualmente marca la normativa según la finalidad del fichero, usos previstos para el mismo y naturaleza de los datos, fundamentalmente.

En virtud de estas disposiciones de la Ley resulta necesario revisar, actualizar y supervisar todos los formularios o cualesquiera otros procedimientos se utilicen para la recogida de datos susceptibles de ser tratados en la forma prevista en la Ley.

Conservación y cancelación de los datos

El almacenamiento y tratamiento de los datos ha de hacerse de forma que preserve los intereses del afectado: por un lado, los derechos de acceso, rectificación y demás reconocidos expresamente en la Ley y, de otro, la preservación de su intimidad y privacidad mediante el establecimiento, aplicación y cumplimiento de las medidas de seguridad que han de ser previstas con anterioridad a la creación del fichero. De esta manera, las comunicaciones y cesiones de los datos deberán respetar también los requisitos y las medidas de seguridad necesarias para garantizar la salvaguarda de los derechos dimanantes de la Ley.

Igualmente, si el fichero pierde la finalidad originaria no está permitido su reutilización en otras actividades debiendo, en consecuencia, destruirse los datos correspondientes.