UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

MANUAL TÉCNICO DEL SISTEMA DE INSCRIPCIONES DE LA FACULTAD DE FILOSOFÍA Y LETRAS (SIFYL)

REPORTE DE TRABAJO PROFESIONAL

QUE PARA OBTENER EL TÍTULO DE:

ACTUARIA

P R E S E N T A

MARGARITA HUERTA MARTÍNEZ

DIRECTORA DE TESIS
M. en D. MARÍA TERESA VELÁZQUEZ URIBE

FACULTAD DE FILOSOFÍA Y LETRAS
DEPARTAMENTO DE SISTEMAS

SISTEMA DE INSCRIPCIONES (SIFYL)

MANUAL TÉCNICO

INDICE Pág.

INTRODUCCIÓN

En el Departamento de Sistemas de la Secretaría Académica de Servicios Escolares de la Facultad de Filosofía y Letras, se ha desarrollado el "Sistema de Inscripciones para la Facultad de Filosofía y Letras (SIFYL)".

Este sistema permite que los estudiantes realicen la inscripción en línea a los cursos ordinarios y a los exámenes extraordinarios que semestre a semestre ofrece la Facultad, en el período lectivo próximo a iniciarse; validando el cumplimiento del Reglamento General de Inscripciones y del Reglamento General de Exámenes.
Este sistema además aplica filtros de validación de los requerimientos que para la administración del plan requieran las Coordinaciones de Carrera.
Algunos de los filtros implementados son:

- Alumnos que de acuerdo a su año de ingreso han rebasado en semestres el 100 % que determina el plan mas el 50 % adicional o que tienen un avance en créditos de 75% y al menos 8 de promedio, se pueden inscribir a 6 extraordinarios en un semestre. Todos los demás alumnos sólo pueden inscribir hasta 2.
- No se pueden inscribir materias de un plan anterior de su misma carrera.
- No se puede efectuar la inscripción de una materia aprobada.
- No se puede efectuar la inscripción de una materia cursada dos veces en ordinario.
- Un alumno no se puede inscribir al mismo curso/grupo con diferentes claves de asignatura.
- Reconocimiento de asignaturas propias y compartidas.

El acceso al sistema está restringido al uso de una clave individual por parte del estudiante (número de cuenta y contraseña la fecha de nacimiento), además de existir programación de la atención de acuerdo al plan de estudios y generación de los alumnos.

Este sistema permite a los alumnos:

- Inscribir la(s) asignatura(s) que desea y puede cursar de Cursos Ordinarios.
- Realizar ajustes a su inscripción (Altas, Bajas y Cambios) de los Cursos Ordinarios.
- .Imprimir tanto el "Comprobante Provisional de Inscripción", como el "Comprobante Definitivo de Inscripción".
- Registrar exámenes a extraordinarios.
- Realizar una inscripción en línea permite que los alumnos obtengan en ese momento la aceptación o el rechazo de la solicitud.

El sistema está desarrollado utilizando PHP, con bases de datos PostgreSQL.

El presente documento está estructurado en 7 capítulos:

El capítulo 1, contiene antecedentes del sistema de inscripciones, desde los procesos con uso de hojas de lectura óptica hasta el registro en línea de Internet.

El capítulo 2, detalla en los recursos tecnológicos utilizados, para la implementación de este sistema.

En el capítulo 3, se explica la forma en que se realizó la instalación y configuración de equipos.

En el capítulo 4, se explica el modelado del sistema.

En el capítulo 5, se muestra la programación y subsistemas.

El capítulo 6, contiene el diseño de la base de datos.

En el capítulo 7, se explica la forma de preparar el sistema para cada proceso de registro.

Finalmente están algunas conclusiones y recomendaciones.

1. ANTECEDENTES DEL SISTEMA DE INSCRIPCIONES

En la División de Estudios Profesionales, perteneciente a la Facultad de Filosofía y Letras, existen Coordinaciones de Carrera, que son las áreas encargadas de proveer a los estudiantes de toda la infraestructura que les permite cubrir el plan de estudios; algunas Coordinaciones tienen a su cargo una carrera y algunas hasta 4 carreras.
Para cumplir con sus objetivos las coordinaciones realizan actividades como; Planear horarios de clase: profesores, horarios, salones, cupos, contenidos temáticos; así como autorizar la inscripción a los alumnos que la soliciten.

En el año 1985, en la Universidad Nacional Autónoma de México, se tenían equipos Mainframes Burroughs (B6700), en la Dirección General de Administración Escolar (DGAE), que en esa época estaba ubicada en la Rectoría de la UNAM.

A través de esta infraestructura técnica, en la Secretaría de Asuntos Escolares de la Facultad de Filosofía y Letras, en el Departamento de Sistemas, utilizando en lenguaje Algol, se desarrollo un sistema con módulos : Inscripciones, Horarios, Historias Académicas.

La solicitud de las asignaturas grupo que el alumno quería registrar tanto para cursos ordinarios como para exámenes extraordinarios se realizaba a través del llenado de las hojas de lectura óptica, las cuales requerían del visto bueno de las Coordinaciones de Carrera para posteriormente ser entregadas en las ventanillas de Servicios Escolares, una vez que se verificaba el llenado de las hojas, por parte del Departamento de Sistemas, se enviaban para su lectura al centro de cómputo, después de lo cual se procesaba la información, para obtener

- Listas de grupo.
- Archivos de grupos, profesores e inscripción para enviar a la DGAE, para que se elaboraran las actas de calificación correspondiente.

Posteriormente en los años 1990 se generó una infraestructura de cómputo propia, proveyendo una red interna con un servidor Novell y desarrollo de programación en Clipper.

Este servidor era seguro, potente pero su tecnología no correspondía a los requerimientos de ese momento; también la estructura de la base de datos almacenada en dicho servidor, era obsoleta y fue necesario rehacerla.

Sin embargo, esto permitió dejar el uso de las hojas de lectura óptica, ya que los alumnos se presentaban en ventanillas a solicitar su inscripción, previo del visto bueno de la Coordinación.

En el año 1998, cuando ingrese a esta Facultad, se buscó una tecnología y plataforma que más se adecuara a las necesidades de la Secretaría, y fue en el año 2001 que logramos diseñar e implementar un sistema para el proceso de inscripción, con las siguientes características: base de datos con el manejador PostgreSQL, programado en PHP, con sistema operativo Linux, que se ejecuta en Internet

A partir del semestre 2002-2 se realizó el registro a cursos ordinarios (reinscripción) así como a exámenes extraordinarios en Internet.

El sistema se aplica a alumnos con trayectorias académicas en 16 carreras del sistema escolarizado y 7 del sistema abierto de la facultad.

Las reglas de los planes de estudio.- Son la aplicación de definiciones precisadas en el documento de aprobación del plan de estudios, por ejemplo la seriación de materias, la pertenencia de las materias a una área especifica del conocimiento así como el porcentaje que se puede cubrir dentro del plan de estudios y el carácter de los mismos; créditos obligatorios u optativos.

Los filtros de los planes de estudio.- Son las precisiones que realizan las Coordinaciones de carrera a fin de poder administrar el plan de estudios, por ejemplo, el seguimiento de grupo; que las asignaturas de dos semestres, requieran de ser cursadas con el mismo profesor.

Semestre a semestre se actualiza e incorpora código ya que ha sido paulatina la integración de los filtros y reglas de los planes de estudio.
Dentro de los cuales se puede mencionar:

a) Control de cupos. Las Coordinaciones nos definen la cantidad de alumnos que pueden atender y el sistema verifica si existe cupo y en caso de proceder le permite la inscripción en dicha asignatura grupo, con esto no se requiere que los alumnos se presenten en Coordinación para su visto bueno.
b) Seguimiento de grupo.- En lo referente al manejo de claves de asignaturas, en términos administrativos la DGAE asigna claves diferentes para las materias de todos y cada uno de los semestres, ya que en la definición de los planes de estudio de la Facultad se indica que son planes semestrales y por lo tanto se asignan claves por semestre con una duración de 8 hasta 10 semestres; sin embargo en el ámbito de lo académico, existen asignaturas con una duración de un semestre y algunas de dos, en el caso de asignaturas de 2 semestres, la academia exige que sean cursadas con el mismo profesor (ya que corresponden al mismo curso).
   El sistema contiene la información de las materias con seguimiento de grupo, esto es las materias de dos semestres, donde la materia antecedente es la primera asignatura y la materia a procesar, la segunda asignatura, y por lo tanto sólo permite que se inscriban a las asignaturas grupo en las que coincida el profesor en la asignatura antecedente.
c) Seriación de asignaturas.- De igual forma en algunas carreras se define la seriación obligatoria y el sistema verifica en la historia académica de los alumnos, si éstos tienen aprobada la materia antecedente para permitir registrar lo solicitado.
d) Integración de reglas de restricción indicadas por las Coordinaciones de Carrera.

2. RECURSOS TECNOLÓGICOS UTILIZADOS

EL sistema de inscripciones de la Facultad de Filosofía y Letras **(SIFYL)** fue desarrollado en el Departamento de Sistemas de la Secretaría Académica de Servicios Escolares.
Este sistema se desarrolló con software de licencia libre. Las herramientas y el equipo utilizado es el siguiente:

EQUIPO : GALILEO SERVIDOR WEB	
PLATAFORMA	Linux, sistema operativo Redhat 7.2
HARDWARE	Procesador :Pentium III á 450 Mhz. Memoria 530 MB D.D: cantidad 1, capacidad 18 GB
SOFTWARE	Linux Redhat 7.2 Apache version 1.3.24 PHP version 4.2

EQUIPO : SÓCRATES SERVIDOR DE BASES DE DATOS	
PLATAFORMA	Linux, sistema operativo Redhat 7.3
HARDWARE	HP WORKSTATION XW-6000 Procesador : XEON á 2.8 Mhz. Memoria: 2 GB D.D: cantidad 1, capacidad 60 GB
SOFTWARE	Linux Redhat 7.3 PostgreSQL versión 7.2 Perl versión 5 TeTeX

La instalación de Perl se encuentra en la máquina SISLINUX, desde la cual se realiza el mantenimiento a ambos equipos.

3. INSTALACIÓN Y CONFIGURACIÓN DE EQUIPOS

El esquema con el que se cuenta para la implementación del sistema es:

- Servidor web *Galileo*
- Servidor de bases de datos *Socrates*
- Equipo de soporte *Sislinux*

Para cada equipo, de acuerdo con su uso, se realizó una instalación y configuración.

3.1 SERVIDOR DE PÁGINAS WEB

Instalación de sistema operativo Redhat versión 7.3
Una vez instado Linux Redhat se inicia la configuración de los siguientes archivos:

- Se edita el archivo de definición de la interfaz de red, dicho archivo se encuentra en: ***/etc/pf.conf,*** en este archivo se le indican al servidor las conexiones internas de la red, por lo tanto se le agrega únicamente la IP del servidor de bases de datos, que es al que se va a conectar el servidor web. Para ejecutarlo se teclea: ***.#pfctl -f /etc/pf.conf*** (como súper usuario) (ANEXO 4).
- Se configura el servidor apache que edita el siguiente archivo (como súper usuario) y que se encuentra en ***var/www/conf/httpd.conf*** este archivo incluye declaración de directorios de trabajo, permisos, activa PHP entre otras cosas. Para activar el servicio se ejecuta ***/usr/sbin/apachectl restart*** (ANEXO 5)
- Para configurar PHP se edita el archivo PHP.INI que se encuentra en ***/var/www/conf/php.ini*** (ANEXO 6)

3.2 SERVIDOR DE BASES DE DATOS

- Instalación de sistema operativo *Redhat* versión 7.3

Una vez instado linux *Redhat* se empieza la configuración de los siguientes archivos:

- Configuración del *firewall*: la configuración se hace sobre el archivo ***/etc/sysconfig/ipchains*** (como súper usuario); este archivo contiene las configuraciones de las IP's que el servidor va a aceptar tanto para entrada como para salida para los puertos 22 y 5432, para realizar conexiones con *ssh* y *postgres* respectivamente (ANEXO 1). Para ejecutar este archivo se ejecuta:
  /etc/rc.d/init.d/ipchains restart
- Se configura postgres: el archivo de configuración es ***/etc/rc.d/init.d/postgresql*** dicho archivo contiene el script que levanta el servicio de postgres; este archivo se edita para indicarle la opción -i para que acepte conexiones (remotas) TCP/IP, el bloque que se edita es: #CHECK FOR POSTMASTER ALREADY RUNNING.... Una vez editado el archivo postgresql se levanta el servicio de la siguiente forma***: /etc/rc.d/init.d/postgresql start o /etc/init.d/postgresql o service postgres start*** (como súper usuario) (ANEXO 2)
- El punto anterior crea los siguientes archivos de configuración de postgres:
  1. initdb.i18
  2. base
  3. global
  4. pg_clog
  5. pg_hba.conf
  6. pg_ident.conf
  7. PG_VERSION
  8. pg_xlog
  9. postgresql.conf
  10. postmaster.pid
  11. postmaster.opts
- Se edita el archivo ***pg_hba.conf*** ; donde se indica las IP's que va a aceptar la (s) base (s) de datos que se encuentran alojadas en el servidor (como súper usuario) (ANEXO 3)
- Finalmente para levantar automáticamente al arrancar el servidor se teclea el siguiente comando (como súper usuario)
  chkconfig --level 5 postgres start ó setup
- Se agrega el servicio.

4. MODELADO DEL SIFYL

El diagrama de flujo de datos describe los procesos que cambian o transforman los datos en un sistema, las entidades externas que son fuente o destino de los datos y los almacenamientos o depósitos de datos a los cuales tiene acceso el sistema, permitiendo así describir el movimiento de los datos a través del sistema.

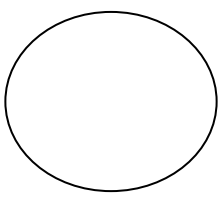
En el diagrama de flujo de datos (DFD), el Nivel 0 o Diagrama de Contexto es aquel que muestra una sola burbuja y las entidades con las que interactúa el sistema.

Simbología:

Entidad Externa:

Como el uso del sistema está básicamente dirigido a los alumnos nuestra entidad externa son los ALUMNOS.

Proceso:

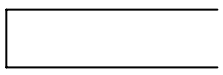


Sistema de Inscripciones de la Facultad de Filosofía y Letras (SIFYL).

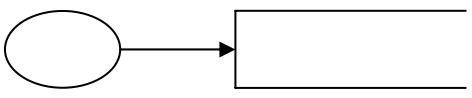
Flujo de datos:

Un flujo muestra las relaciones entre los elementos del DFD.

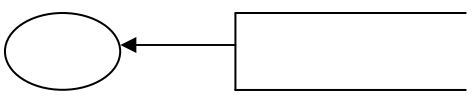
Almacén o archivo:



Representa un archivo lógico en donde se agregan o de donde se extraen datos. Es una estructura de datos, pero estática.



Implica escritura, actualización o borrado de datos.

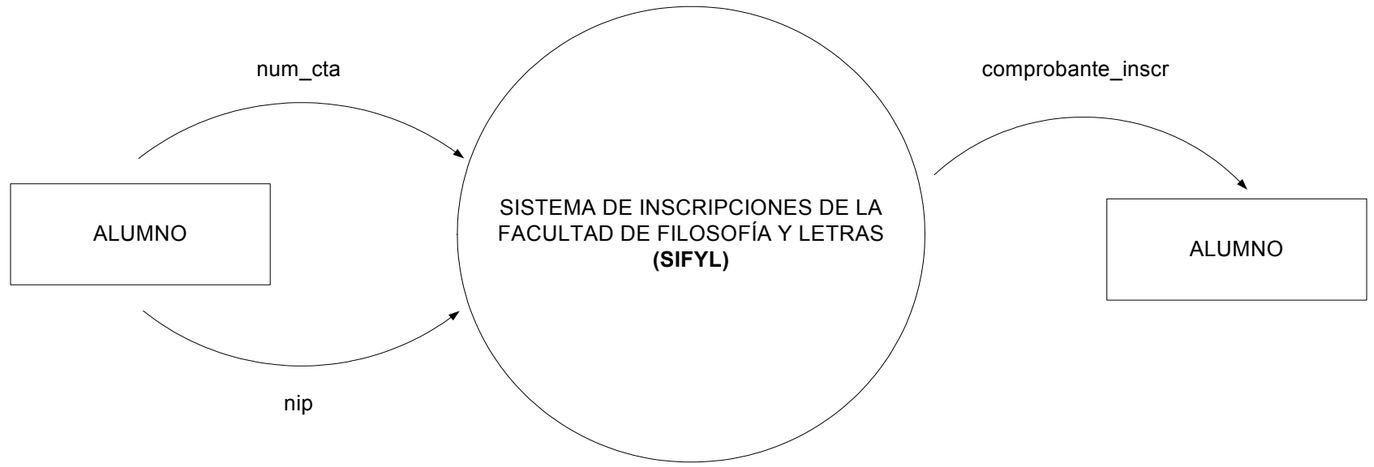


Implica lectura o recuperación de información almacenada.

4. 1 DFD NIVEL 0

DIAGRAMA DE CONTEXTO

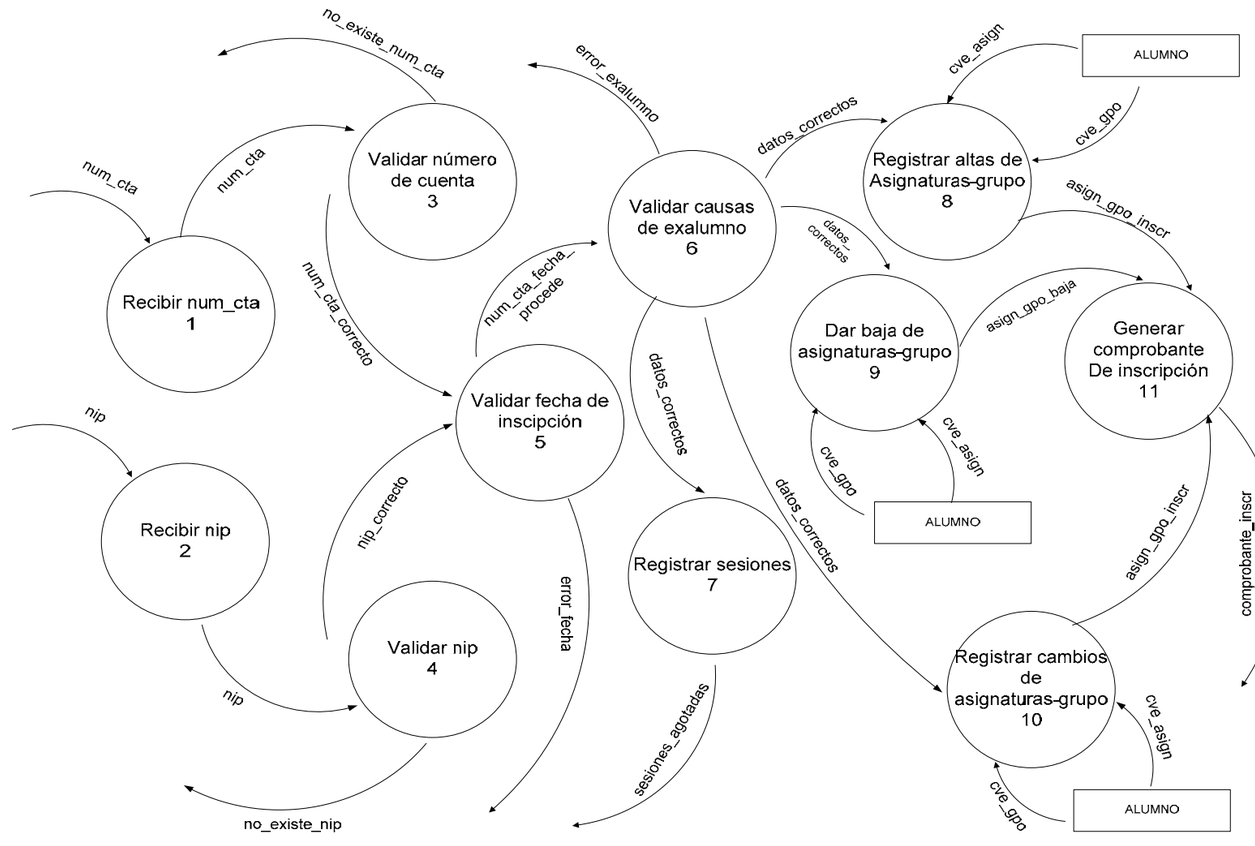
En términos generales el alumno proporciona su número de cuenta y su NIP (fecha de nacimiento) y el SIFYL genera un comprobante de inscripción al alumno.



4.2 DFD NIVEL 1

La secuencia de procesos que realiza el SIFYL es la siguiente:

1. Recibir Número de Cuenta
2. Recibir NIP
3. Validar Número de Cuenta
4. Validar NIP
5. Validar Fecha de Inscripción
6. Validar Causa De Exalumno
7. Registrar Sesiones
8. Registrar Altas de Asignatura-Grupo
9. Dar Baja de Asignatura-Grupo
10. Registrar Cambios de Asignaturas-Grupo
11. Generar Comprobante de Inscripción


num_cta
Recibir num_cta
1
num_cta
Validar número de cuenta
3
no_existe_num_cta
num_cta_correcto
nip
Recibir nip
2
nip_correcto
Validar nip
4
nip
no_existe_nip
Validar fecha de inscipción
5
num_cta_fecha_procede
error_fecha
Validar causas de exalumno
6
error_exalumno
datos_correctos
Registrar sesiones
7
sesiones_agotadas
datos_correctos
Registrar altas de Asignaturas-grupo
8
ALUMNO
cve_asign
cve_gpo
datos_correctos
Dar baja de asignaturas-grupo
9
ALUMNO
cve_asign
cve_gpo
asign_gpo_baja
asign_gpo_inscr
Generar comprobante De inscripción
11
comprobante_inscr
datos_correctos
Registrar cambios de asignaturas-grupo
10
ALUMNO
cve_asign
cve_gpo
asign_gpo_inscr

4.3 DFD NIVEL 2

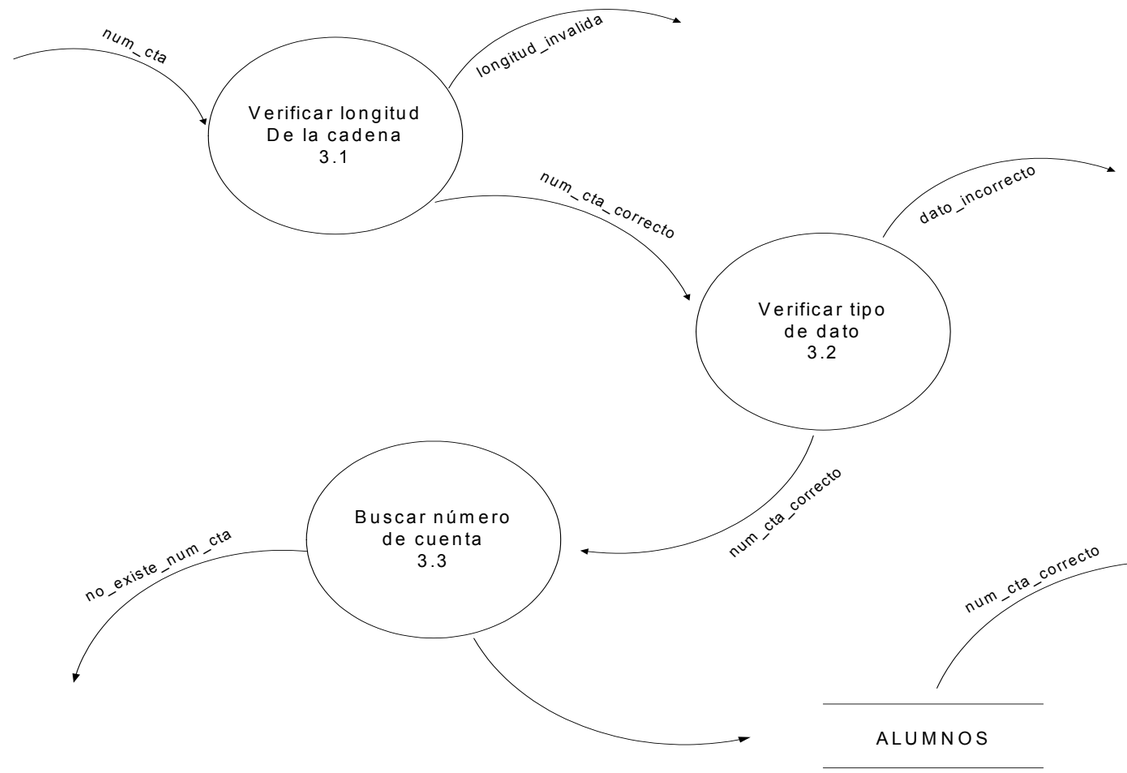
En NIVEL 2 se tienen los siguientes procesos:

4.3.1 Validar Número de Cuenta
4.3.2. Validar NIP
4.3.3 Validar Fecha de Inscripción
4.3.4 Validar Causas de Exalumno
4.3.5. Registrar Sesiones
4.3.6 Registrar Altas de Asignatura Grupo
4.3.7 Dar Baja de Asignatura Grupo
4.3.8 Registrar Cambios de Asignatura Grupo
4.3.9 Generar Comprobante de Inscripción

Los procesos 1 y 2 del NIVEL 1 (recibir número de cuenta y recibir NIP) no tienen seguimiento en NIVEL 2

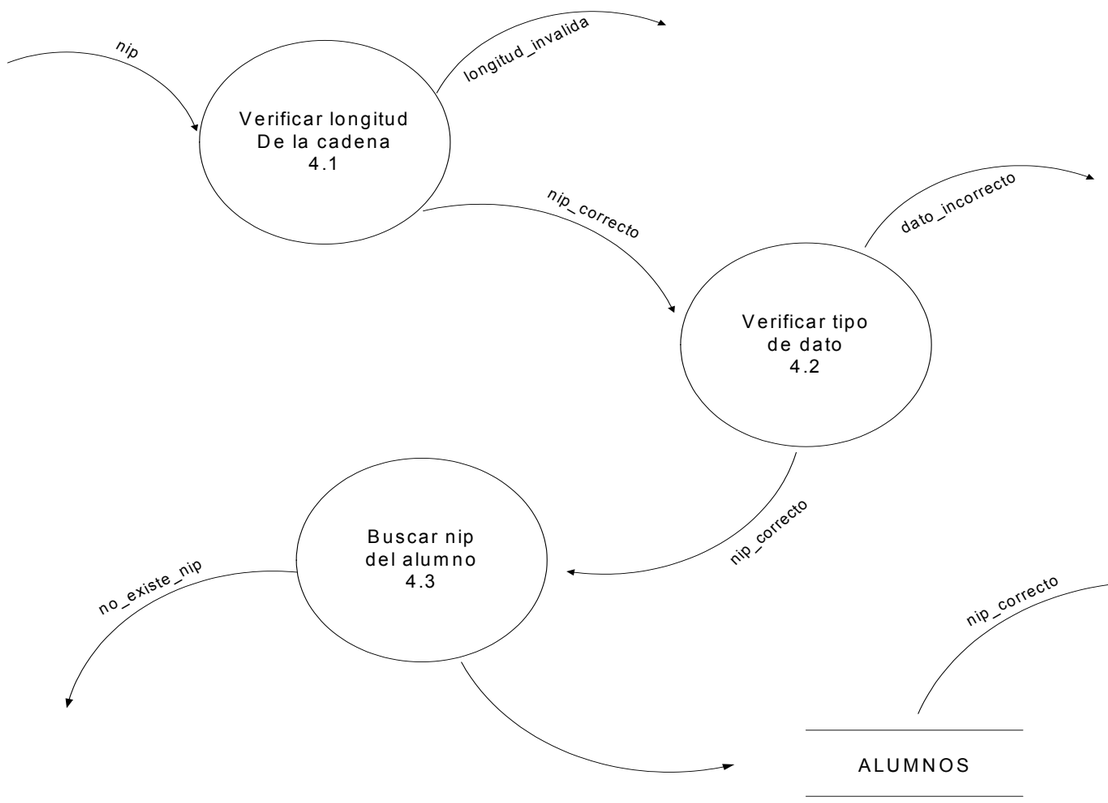
4.3 DFD NIVEL 2

4.3.1 Validar Número de Cuenta



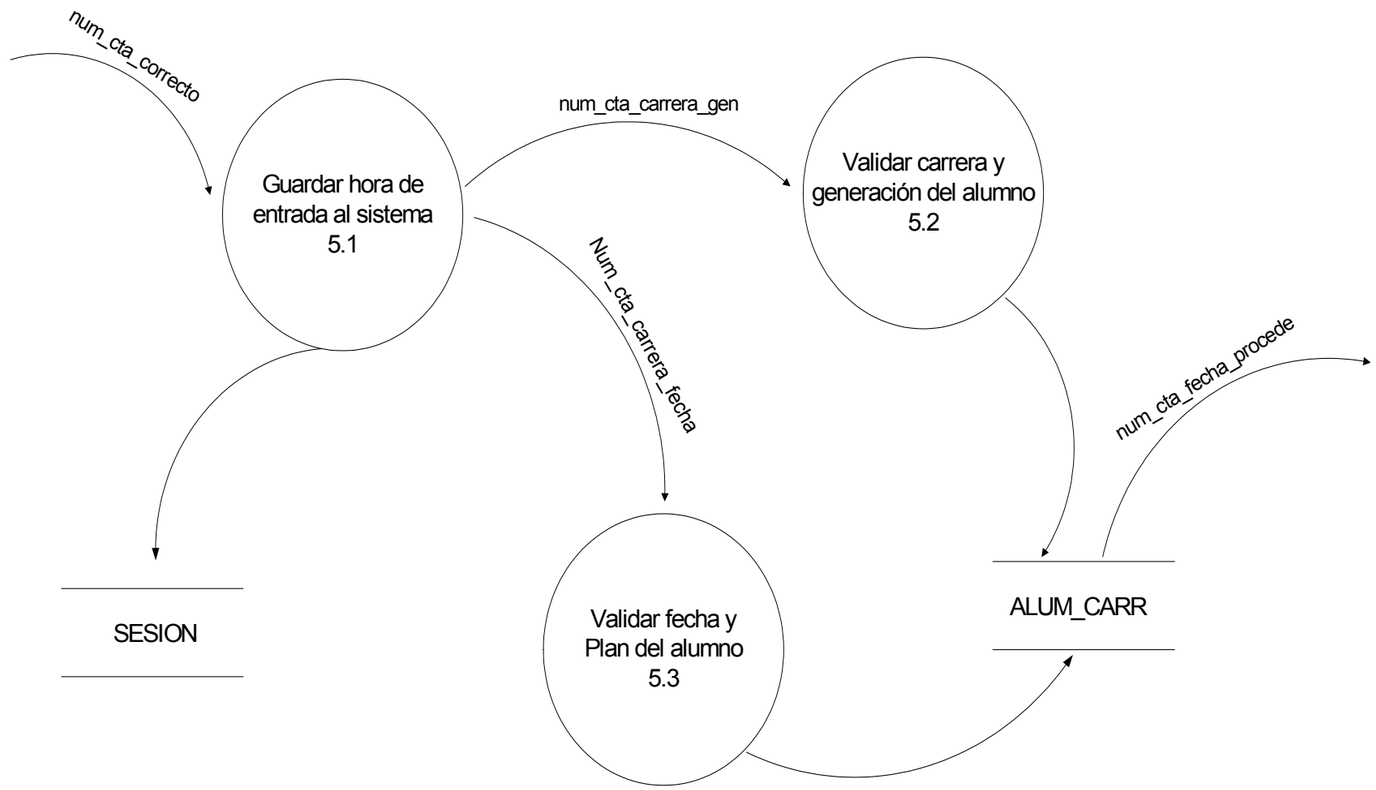
4.3 DFD NIVEL 2

4.3.2 Validar NIP



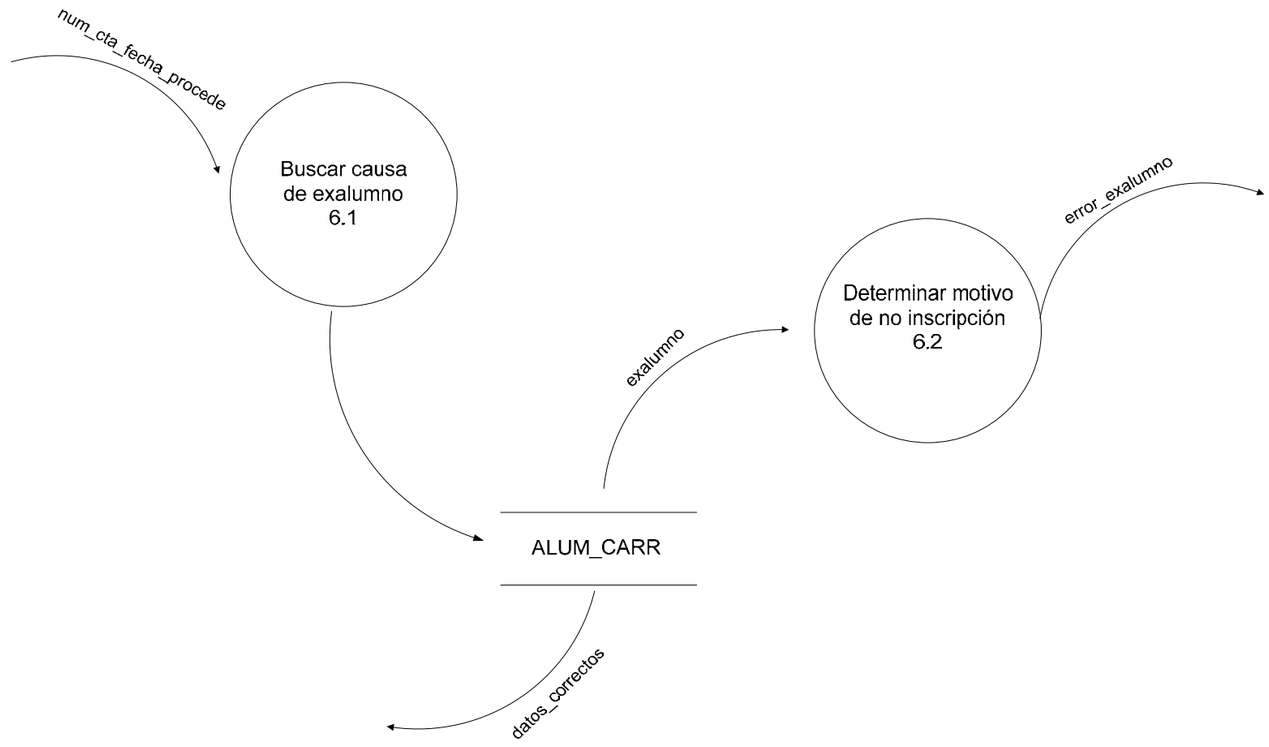
4.3 DFD NIVEL 2

4.3.3 Validar Fecha de Inscripción



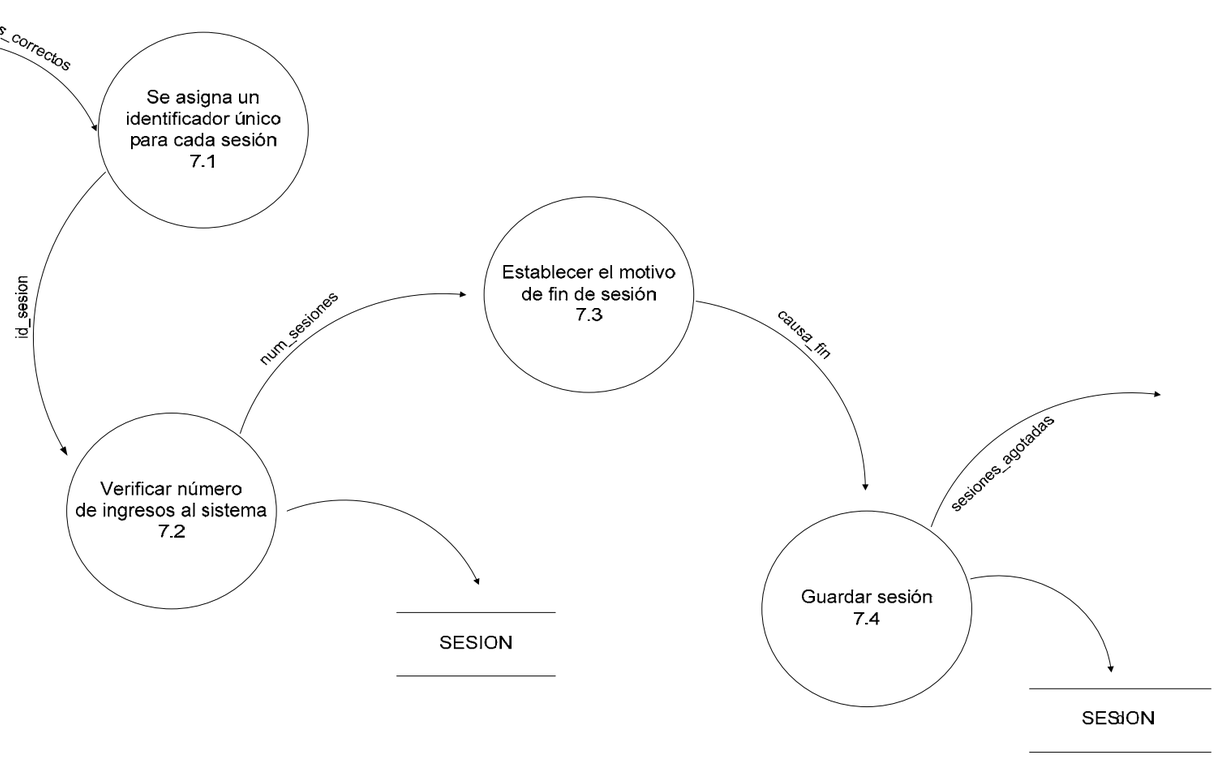
4.3 DFD NIVEL 2

4.3.4 Validar Causas de Exalumno



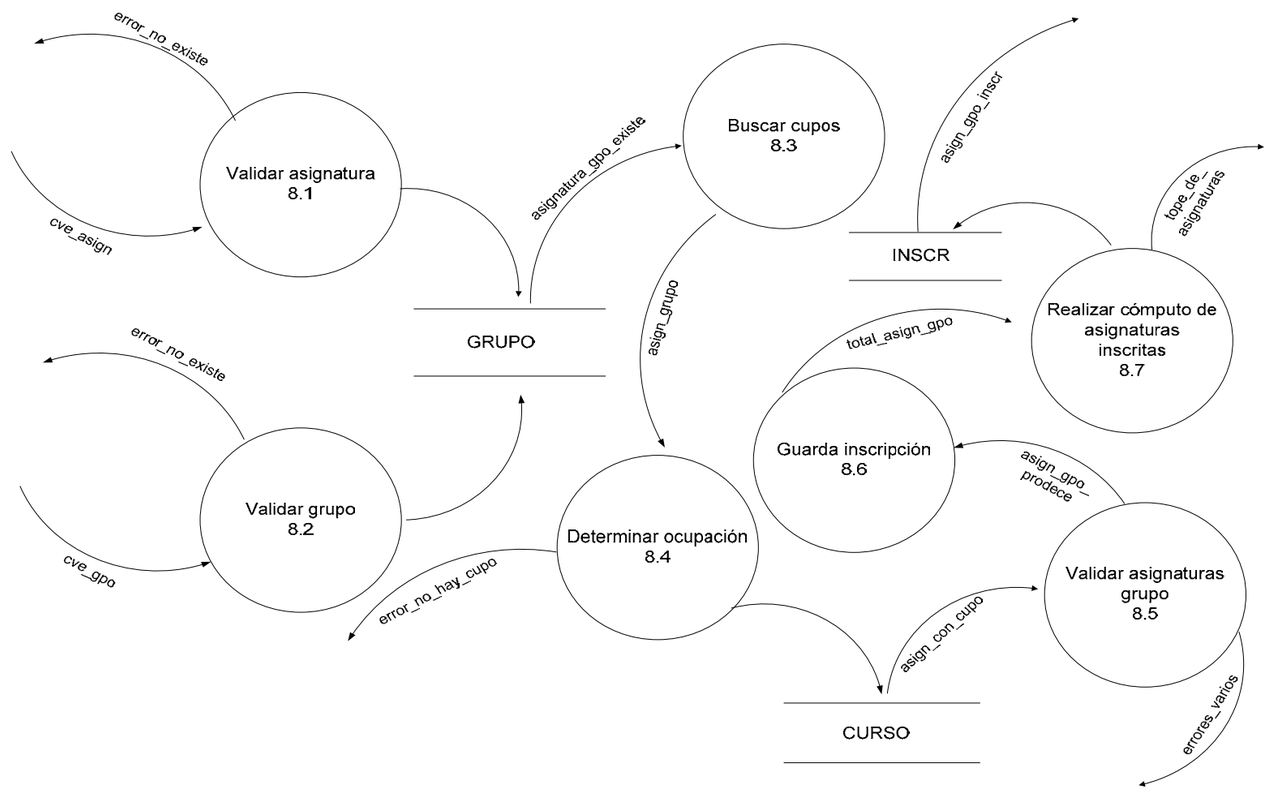
4.3 DFD NIVEL 2

4.3.5 Registrar Sesiones



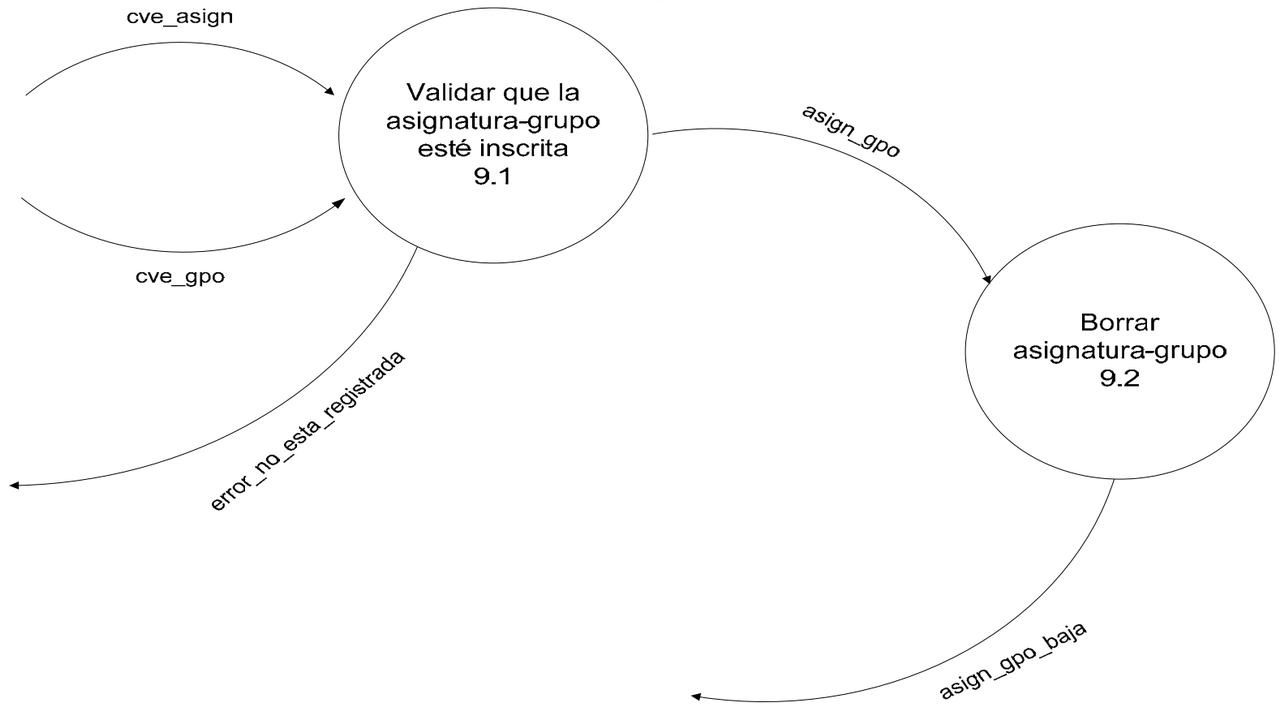
4.3 DFD NIVEL 2

4.3.6 Registrar Altas de Asignatura-Grupo



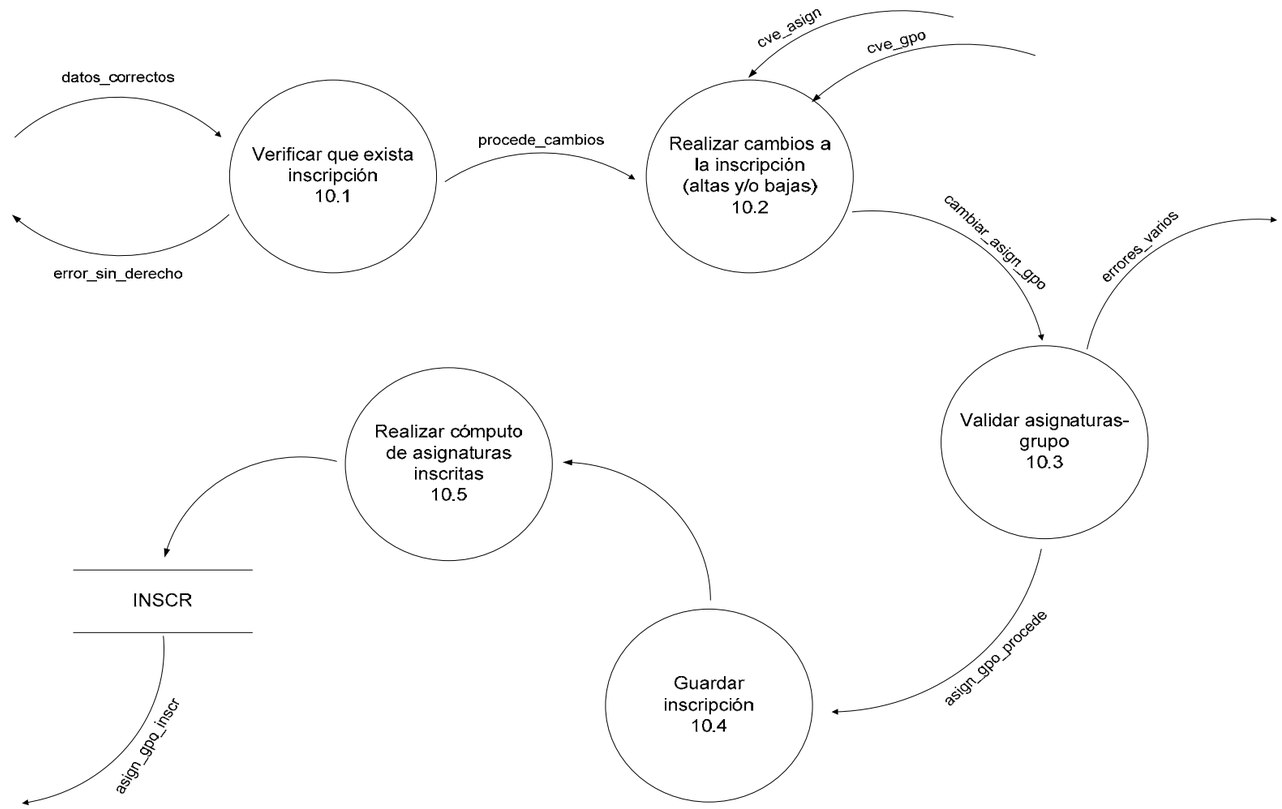
4.3 DFD NIVEL 2

4.3.7 Dar Baja de Asignatura Grupo



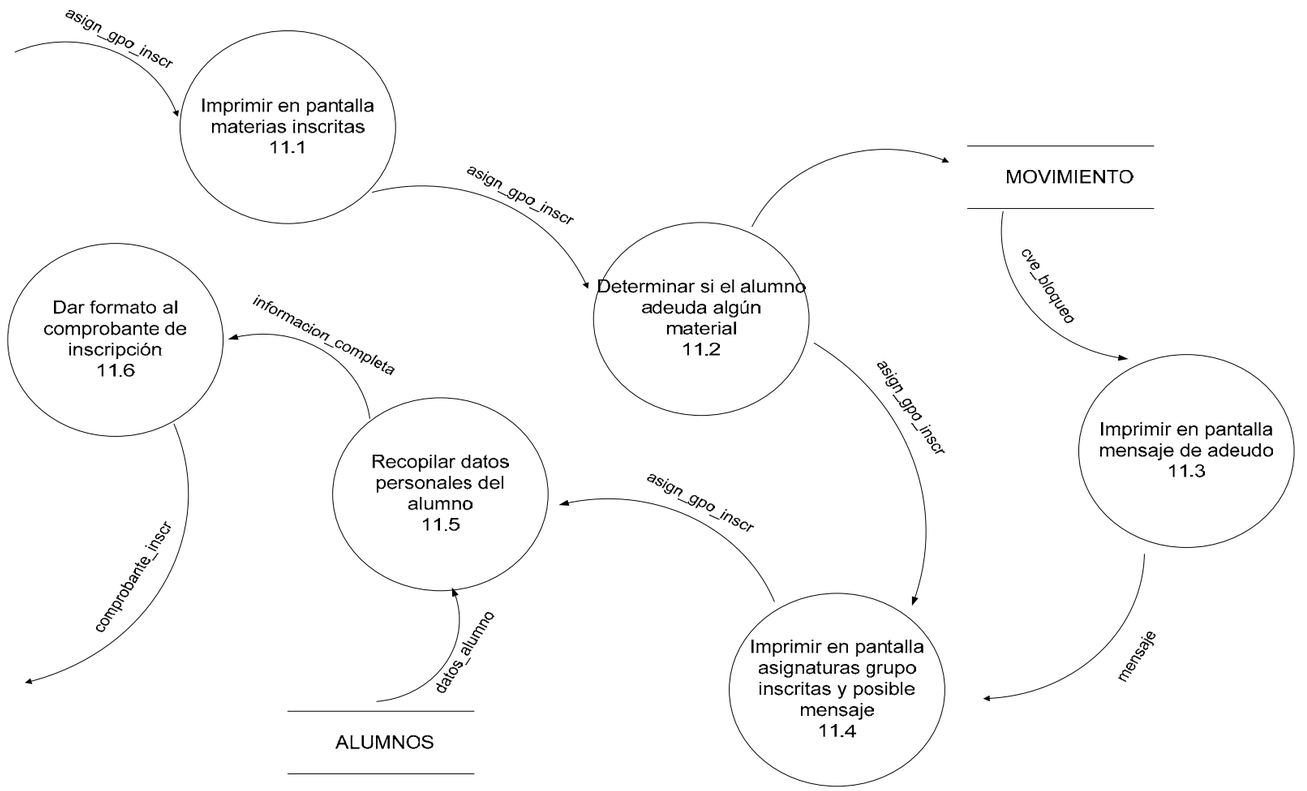
4.3 DFD NIVEL 2

4.3.8 Registrar Cambios de Asignatura Grupo



4.3 DFD NIVEL 2

4.3.9 Generar Comprobante de Inscripción



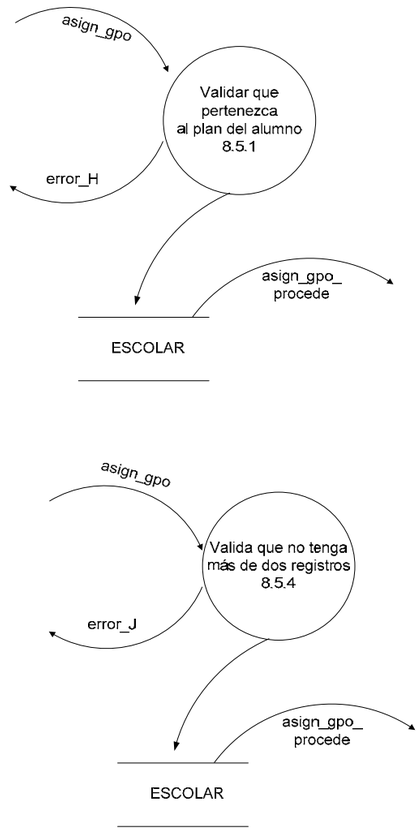
4.4 DFD NIVEL 3

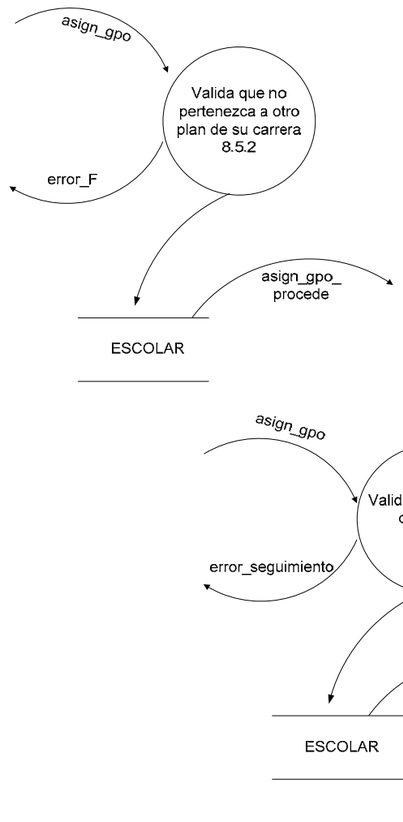
En NIVEL 3 se tienen los siguientes procesos:

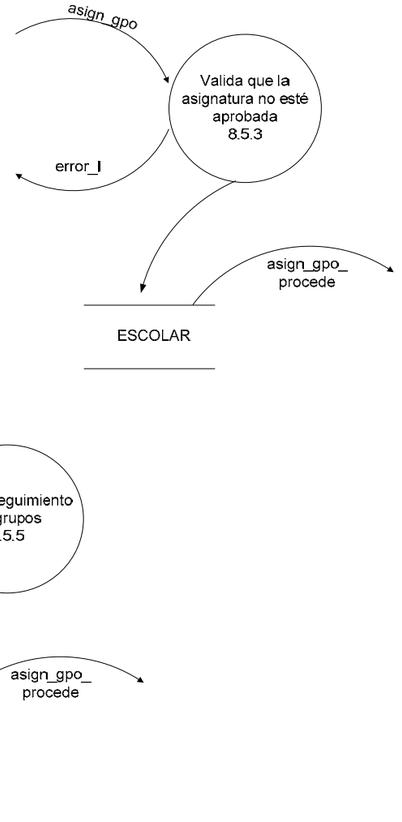
4.4.1 Validar Asignaturas Grupo
4.4.2. Validar Seriación Asignaturas Grupo

4.4 DFD NIVEL 3

4.4.1 Validar Asignaturas Grupo

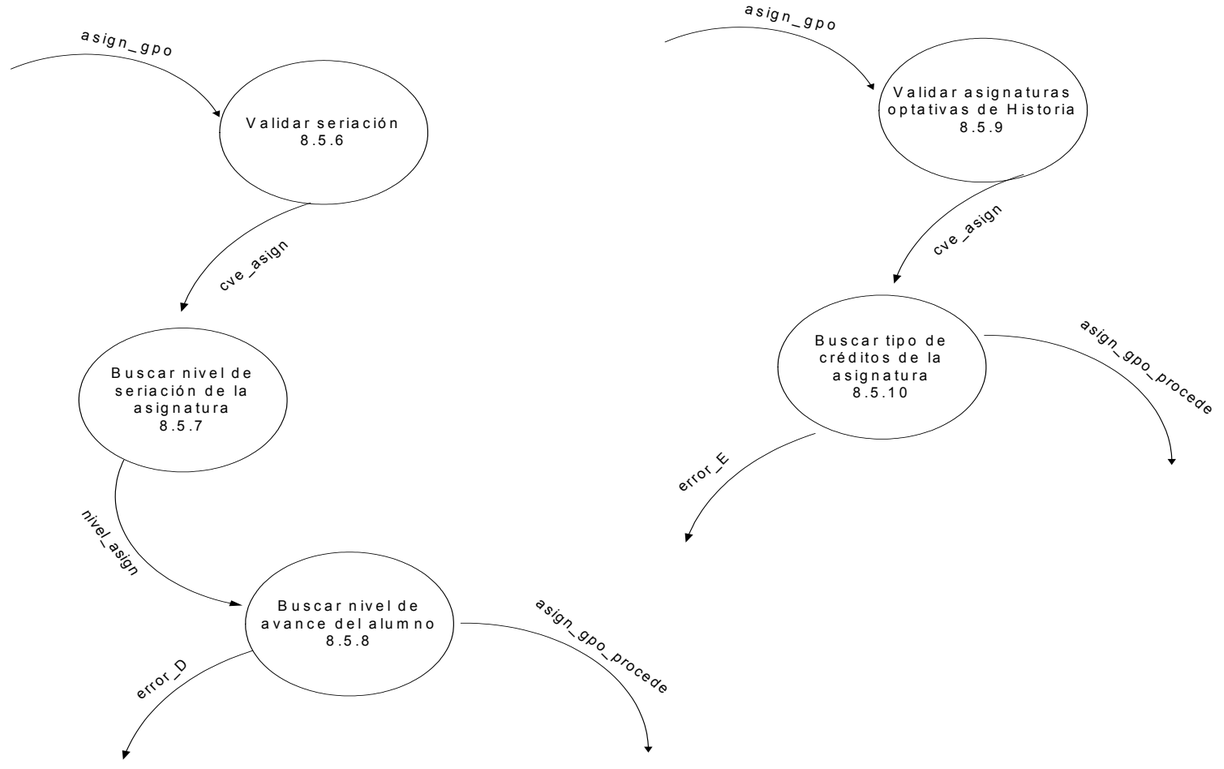






4.4 DFD NIVEL 3

4.4.2 Validar Seriación Asignaturas Grupo



4.5 DICCIONARIO DE DATOS

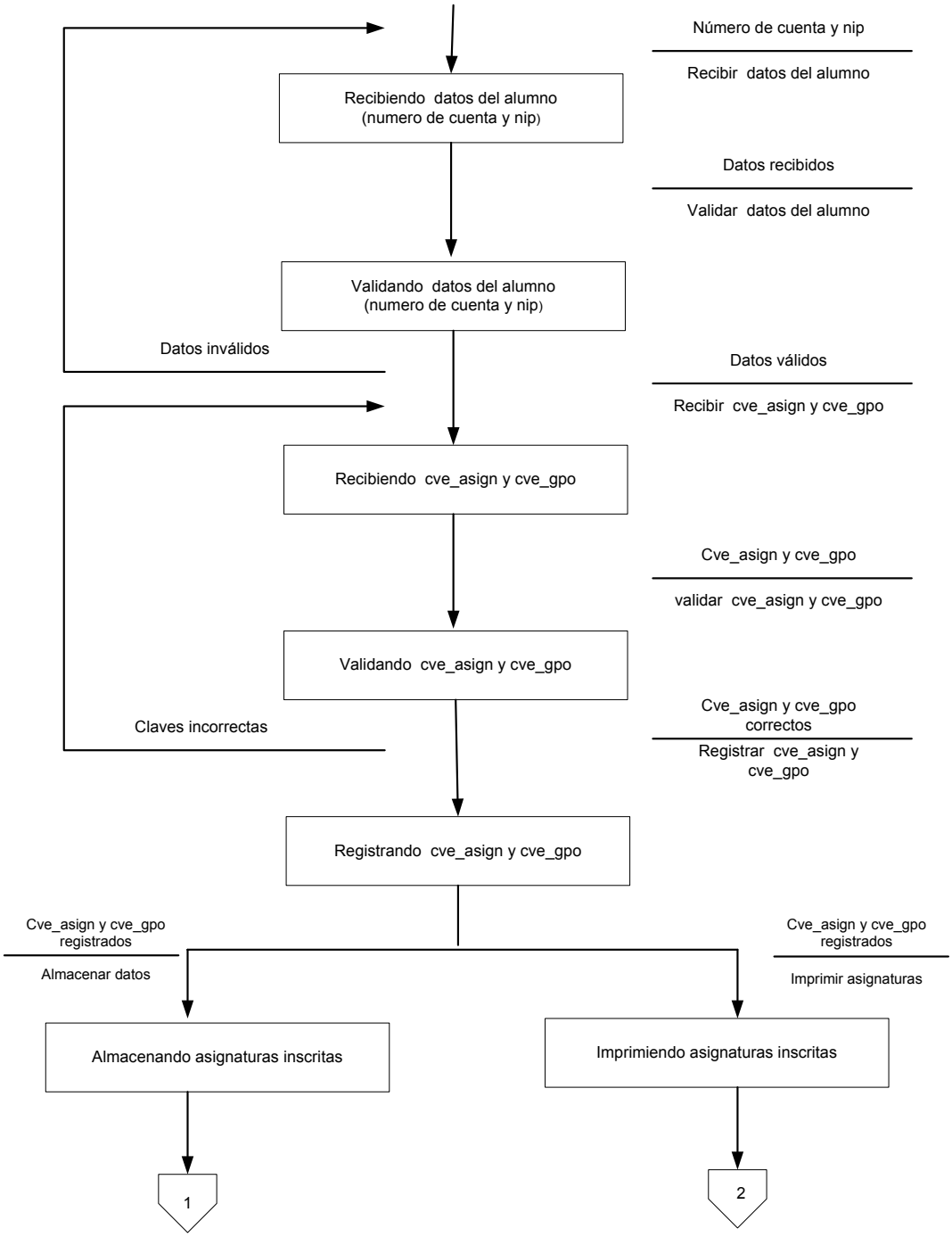
SÍMBOLO	*SIGNIFICADO*
=	Compuesto de, se define como, significa
+	Concatenación de datos
{}	Iteración
**	Comentario
| |	Opción

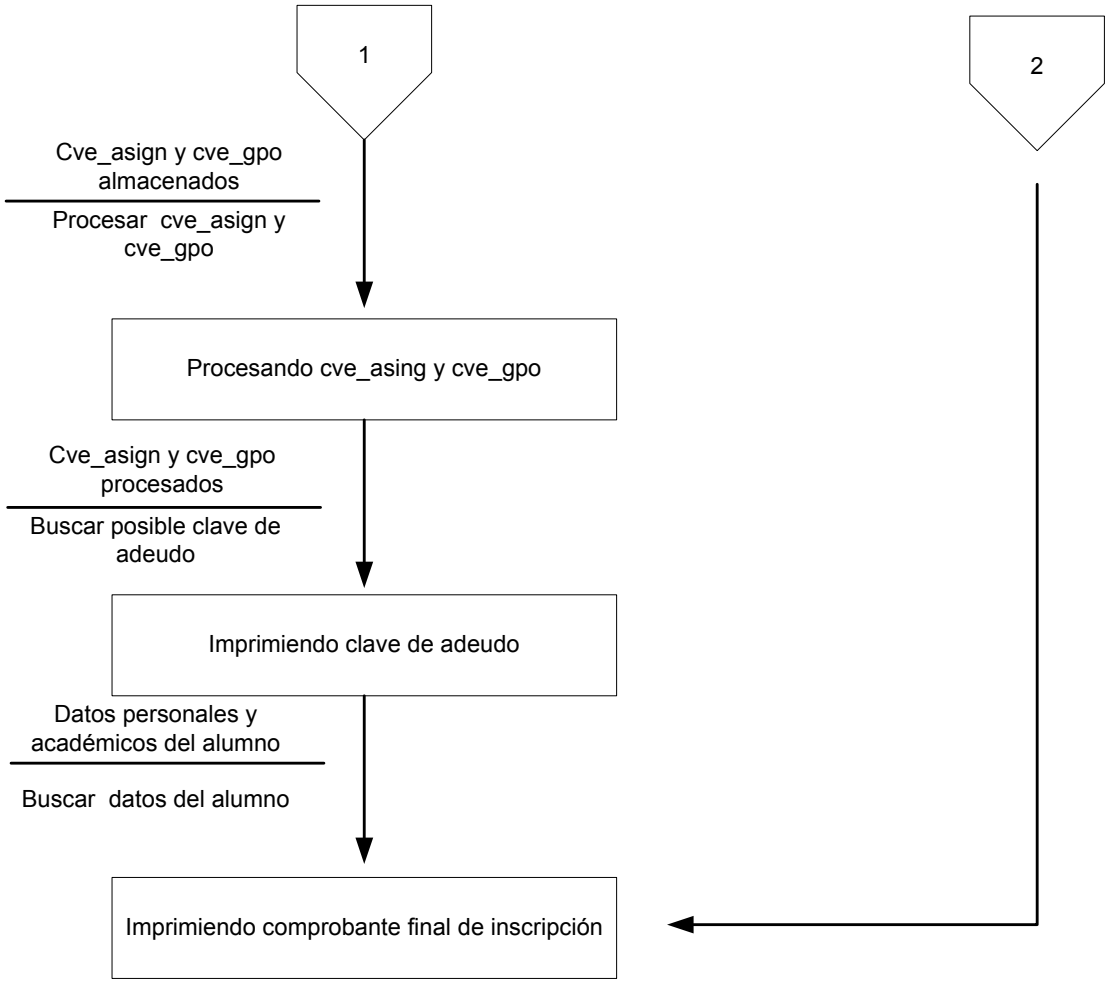
DATO	*DESCRIPCIÓN*	*EJEMPLOS*
num_cta	$=_1\{0\text{-}9\}_9$	096210753
nip	$=_1\{0\text{-}9\}_8$	30071982
comprobante_inscr	=**datos_alumno + asign_gpo_inscr + | mensaje |	
no_existe_num_cta	= {{a-z} | {A-Z} | {0-9}}	El Número de Cuenta 096210753 es incorrecto.
no_existe_nip	= {{a-z} | {A-Z} | {0-9}}	NIP Incorrecto
nip_correcto	$=_1\{0\text{-}9\}_8$	30071982
num_cta_correcto	$=_1\{0\text{-}9\}_9$	096210753
num_cta_fecha_procede	$=_1\{0\text{-}9\}_9$ + **fecha de inscripción correcta	096210753 + 26042004
error_fecha	== {{a-z} | {A-Z} | {0-9}}	EN ESTE MOMENTO NO PUEDES REALIZAR TRAMITES VERIFICA EL CALENDARIO DE SERVICIOS ESCOLARES
error_exalumno	= {{a-z} | {A-Z} | {0-9}}	ERROR. NO tienes derecho a inscripción NOMBRE DEL ALUMNO: JUAN TORRES MENDOZA
datos_correctos	$=_1\{0\text{-}9\}_9$ + $=_1\{0\text{-}9\}_8$	Número de cuenta y nip correctos
sesiones_agotadas	= {{a-z} | {A-Z} | {0-9}}	Sólo tienes derecho a ingresar tres veces al sistema de Registro.
cve_asign	$=_1\{0\text{-}9\}_4$	0315
cve_gpo	$=_1\{0\text{-}9\}_4$	0002
asign_gpo_inscr	$=_1\{0\text{-}9\}_4$ + $_1\{0\text{-}9\}_4$	0315 + 0002
asign_gpo_baja	$=_1\{0\text{-}9\}_4$ + $_1\{0\text{-}9\}_4$	0315 + 0002
longitud_invalida	= {{a-z} | {A-Z} | {0-9}}	El Número de cuenta debe estar formado por 9 dígitos, sin el guión.

		Si tu número es 9215329-7 debes teclear 092153297
dato_incorrecto	= {{a-z} | {A-Z} | {0-9}}	El nip equivale a tu fecha de nacimiento en el formato: ddmmaaa
num_cta_carrera_gen	$=_1\{0\text{-}9\}_9 + {}_1\{0\text{-}9\}_4$	097125985 + 0315
num_cta_carrera_fecha	$=_1\{0\text{-}9\}_9 + {}_1\{0\text{-}9\}_8$	097125985 + 12022004
exalumno	$= {}_1\{0\text{-}9\}_2$	11
error_exalumno	= {{a-z} | {A-Z} | {0-9}}	BAJA DEFINITIVA POR EL ALUMNO
id_sesion	= {{a-z} | {A-Z} | {0-9}}	e191c505bbc34e041f8a4566f7f83f4
num_sesiones	$=_1\{0\text{-}9\}_3$	16
causa_fin	$= {}_1\{A\text{-}Z\}_3$	FIN
sesiones_agotadas	= {{a-z} | {A-Z} | {0-9}}	Sólo tienes derecho a ingresar 3 veces al sistema de Registro.
error_no_existe	= {{a-z} | {A-Z} | {0-9}}	La asignatura 0001 no existe en horarios del sistema
asignatura_gpo_existe	$=_1\{0\text{-}9\}_4$	0312
error_no_hay_cupo	= {{a-z} | {A-Z}	No hay cupo
asign_grupo ó asign_gpo	$=_1\{0\text{-}9\}_4 + {}_1\{0\text{-}9\}_4$	0315 + 0002
asign_con_cupo	$=_1\{0\text{-}9\}_4$	0315
errores_varios	= {{a-z} | {A-Z} | {0-9}}	La asignatura no cumple con los requisitos establecidos por la coordinación
asign_gpo_procede	$=_1\{0\text{-}9\}_4 + {}_1\{0\text{-}9\}_4$	0315 + 0002
total_asign_gpo	$=_1\{0\text{-}9\}_1$	8
asign_gpo_inscr	$=_1\{0\text{-}9\}_4 + {}_1\{0\text{-}9\}_4$	0315 + 0002
tope_de_asignaturas	= {{a-z} | {A-Z} | {0-9}}	Solo puedes inscribir 6 asignaturas
error_no_esta_registrada	= {{a-z} | {A-Z} | {0-9}}	No puedes dar de baja la asignatura 0542 porque no la tienes registrada
asign_gpo_baja	$=_1\{0\text{-}9\}_4 + {}_1\{0\text{-}9\}_4$	0315 + 0002
error_sin_derecho	= {{a-z} | {A-Z} | {0-9}}	ERROR: No puedes realizar cambios porque no te reinscribiste al semestre 2004-1.
procede_cambio	$=_1\{0\text{-}9\}_9$	Se encontró el número de cuenta en la tabla entro
cambiar_asign_gpo	$=_1\{0\text{-}9\}_4 + {}_1\{0\text{-}9\}_4$	0315 + 0002
cve_bloqueo	$=_1\{0\text{-}9\}_2$	00
mensaje	= {{a-z} | {A-Z} | {0-9}}	ADEUDO EN BIBLIOTECA

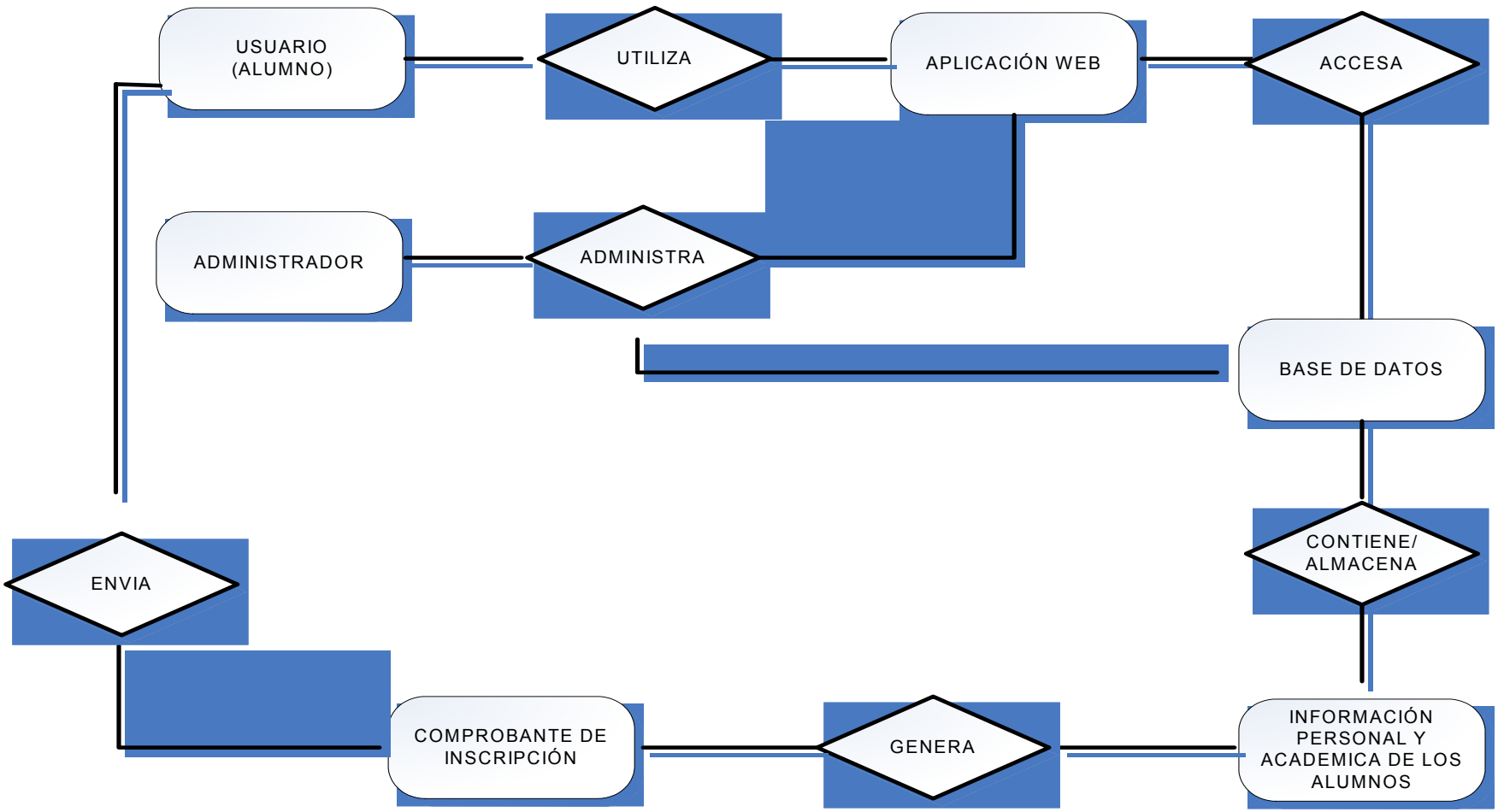
datos_alumno	= {{a-z} | {A-Z} | {0-9}}	JUAN PEREZ BONILLA 096210452 PLAN 0560 CLARRERA: LETRAS CLÁSICAS SEMESTRE 2004-1
información_completa	= {{a-z} | {A-Z} | {0-9}}	= Datos_alumno + Asign_gpo_inscr + mensaje
error_H	= {{a-z} | {A-Z} | {0-9}}	ERROR. La asignatura 3215 no pertenece a tu plan.
error_F	= {{a-z} | {A-Z} | {0-9}}	ERROR. La asignatura 2345 pertenece a otro Plan de tu misma Carrera.
error_J	= {{a-z} | {A-Z} | {0-9}}	ERROR. Art 33.- Ningún alumno podrá ser inscrito más de 2 veces en una misma asignatura. En caso de no acreditarla, solo podrá hacerlo en examen extraordinario de acuerdo con lo dispuesto en el capítulo III del reglamento general de exámenes.
error_D	= {{a-z} | {A-Z} | {0-9}}	ERROR. Asignatura seriada, no tienes calificación previa. Acude a la Coordinación de Modernas.
error_E	= {{a-z} | {A-Z} | {0-9}}	ERROR. No puedes registrar asignaturas optativas de Historia Plan 1999, tampoco podrás registrar asignaturas de otro Colegio si eres alumno de Historia. Acude a tu Coordinación.
error_I	= {{a-z} | {A-Z} | {0-9}}	ERROR. No puedes volver registrar asignaturas que ya aprobaste en periodos pasados.
error_seguimiento	= {{a-z} | {A-Z} | {0-9}}	ERROR. No cumples con los requisitos establecidos por la coordinación
nivel_asign	$=_1\{0\text{-}9\}_1$	8

4.6 DIAGRAMA DE TRANSICIÓN DE ESTADOS




1
2
Cve_asign y cve_gpo
almacenados
Procesar cve_asign y
cve_gpo
Procesando cve_asing y cve_gpo
Cve_asign y cve_gpo
procesados
Buscar posible clave de
adeudo
Imprimiendo clave de adeudo
Datos personales y
académicos del alumno
Buscar datos del alumno
Imprimiendo comprobante final de inscripción

4.7 DIAGRAMA ENTIDAD RELACIÓN LÓGICO



5. PROGRAMACIÒN Y SUBSISTEMAS

5.1 DESCRIPCIÓN DE LA PROGRAMACIÓN

El servidor web está montado sobre una PC con Linux, Apache y *PHP*; en este servidor se encuentran alojados todos los programas que interactúan directamente con la base de datos. Dichos programas y páginas están desarrollados en *PHP* y HTML.
El SIFYL está estructurado en siete programas

- registro.php
- movimientos.php
- comun.php
- entrada.php
- tpos.php
- materias.php
- plan_inc.php

Dichos programas están diseñados bajo una estructura de funciones.
A continuación se detalla cada uno de ellos.

Comun.php

NOMBRE DE LA FUNCIÓN	DESCRIPCIÓN
function obten_nombre_x ($id_conex, $tabla, $cve)	Mediante una consulta a la BD obtiene como resultado el nombre de los alumnos o profesores según sea la elección de la condición.
function parametros_id ($conex, $num_id)	Únicamente verifica que el identificador de alguna sesión exista.
function plan_correcto ($planes, $selec)	Verifica que el plan de estudios exista.
function valida_sesion ($param)	Guarda la hora de entrada al sistema en la variable $param y verifica que la sesión esté dentro de los límites establecidos para ingresar al sistema, de no ser así, manda como salida mensajes de error.
function extrae_tuplas ($conex, $sql)	Esta función contiene la instrucción para ejecutar select's guardados en la variable $sql , realizados dentro de otras funciones y que al momento de ejecutarlos mandan llamar a esta función y que guarda en un arreglo el resultado del query.

function mat_inscritas ($conex, $numero, $plan)	Esta función manda un select a la BD para obtener las asignaturas grupo que el alumno va inscribiendo dependiendo del proceso (extraordinario u ordinario); manda a llamar a la función extrae_tuplas ($conex, $sql) , dicha función va a ejecutar el select y finalmente el resultado se guarda en un arreglo.
function imprime_deudores ($conex, $num_cta)	Esta función se conecta a la BD para hacer un select hacia la tabla "bloqueado" para detectar si el alumno debe libros en la biblioteca; se manda a llamar a la función extrae_tuplas ($conex. $sql) para que ejecute el query y el resultado lo guarda en una variable, en caso de que el valor guardado sea mayor a 0 entonces se manda como salida un mensaje avisando al alumno que "Adeuda libros en la biblioteca."
function imprime_inscritas ($conex, $num_cta, $cve_plan)	Esta función imprime en pantalla las asignaturas grupo que el alumno inscribió, para eso manda llamar a la función mat_inscritas ($conex, $num_cta, $cve_plan) y la asigna a una variable. Con código html se genera una tabla y se le da formato para que dentro de la misma se vayan imprimiendo las asignaturas grupo inscritas. También manda a imprimir el resultado de la función imprime_deudores ($conex, $num_cta).
function formulario_cambios ($action, $ident_sesion)	En esta función se genera un formulario con código html que contiene tres botones "ALTA" ,"BAJA" y "TERMINAR" , así como una pequeña descripción del uso de cada uno de los botones.
function formulario_mat_extra ($action, $ident_sesion)	Esta función es exclusiva del periodo de extraordinarios; genera con código html un formulario donde se ingresan la clave de la asignatura y del grupo que se van a dar de alta.
function leyenda_hist_99 ()	Esta función contiene un mensaje para los alumnos cuyo plan de estudios sea

	Historia.
function leyenda_clas_97 ()	Esta función contiene un mensaje para los alumnos cuyo plan de estudios es Letras Clásicas, esta función sólo se activa para los semestres nones.
function imprime_final ($conex, $id, $cta, $plan)	Esta función imprime en pantalla los datos personales del alumno (manda a llamar a la función obten_nombre_x ($conex, 'alumnos', $cta)), el tipo del sistema al que pertenece (escolarizado o abierto) , manda a llamar a la función imprime_inscritas ($conex, $cta, $plan) y finalmente muestra dos botones; uno de ellos para mandar a imprimir en papel el comprobante de inscripción y otro para cerrar la ventana del navegador y así terminar el proceso de registro.
function rompe_con ($con)	Esta función cierra la conexión hecha a la BD.

entrada.php

Este programa contiene al principio la calendarización del proceso de inscripción ya sea al periodo de inscripción a ordinario o al proceso de altas, bajas y cambios.

NOMBRE DE LA FUNCIÓN	DESCRIPCIÓN
function imprime_plan ($arr, $cve)	Esta función únicamente va a buscar el nombre de la carrera a la que corresponde una clave de plan determinada y la va a imprimir en pantalla.
function planes_alumno ($conex, $numero, $cve)	Esta función mediante un select a la BD busca el plan y la carrera del alumno y guarda el resultado en un arreglo.
function formulario_plan ($action, $nombre, $planes, $num_id)	Esta función se activa cuando el alumno tiene más de 1 trayectoria académica (carrera simultánea), se le presenta un combo para que elija a cuál de las carreras desea inscribirse.
function formato_correcto ($numero_cta, $fecha)	Esta función valida que los datos (número de cuenta y nip) sean los correctos; valida

	que los campos no queden vacíos, que tengan la longitud establecida y que sean datos de tipo entero; en caso de que no cumplan alguna condición se manda a pantalla un mensaje de error y la imposibilidad de continuar con el trámite de registro.
function carrera_motivo_noinscr ($id_conex, $numero)	Esta función mediante un select a la BD busca en determinadas tablas las causas por las que un alumno no tiene derecho a inscribirse, mandando como salida a pantalla un mensaje que le muestra al alumno el motivo por el cual no se puede inscribir.
function inicializa_sesion ($conex, $nombre, $numero)	Esta función asigna a cada sesión un identificar único haciendo uso de una función predefinida de php md5 y uniqid. Asigna un contador a cada sesión que el alumno va generando y en caso de que el alumno no haya cerrado sesiones anteriores se hace un update a la tabla "sesión" donde la causa término que se le va a asignar va a ser "UPD". Por lo tanto inserta en la tabla "sesion" de la BD los valores (num_cta, consec, id_sesion , hr_inicio, ip).
function guarda_sesion_plan ($conex, $id_sesion, $plan)	Esta función hace un update a la tabla sesión de la BD para que escriba el valor del campo cve_plan; ya que ese campo no se llena con la función anterior.
function datos_proceso ($conex)	Esta función verifica que el proceso que está corriendo en un momento determinado sea el correcto.En caso de no serlo manda a pantalla un mensaje avisando que el sistema no puede responder por el momento, de lo contrario asigna el valor del select a un arreglo y se retorna.
function cierra_sesion ($conex, $id_sesion, $causa)	Esta función hace un update a la tabla "sesion"; es decir llena los campos hr_termino de acuerdo a la hora real en que el alumno haya cerrado su sesión y causa_termino este campo lo llena de acuerdo a varias posibilidades de fin de sesión.
function mensaje_inscrito($claves)	Esta función únicamente manda a pantalla

	un mensaje al alumno, en proceso de ordinarios, siempre y cuando ya tenga más de una asignatura inscrita el alumno.
function formulario_materias ($action, $ident_sesion)	Esta función contiene un formulario generado en código html y contiene una caja de texto para dar de alta las asignaturas grupo que el alumno solicite.

materias.php

NOMBRE DE LA FUNCIÓN	DESCRIPCIÓN
function es_idioma($conex, $asgn ,$num ,$plan)	Esta función sirve para llevar a cabo la seriación de asignaturas lo cual sólo es aplicable para la carrera de Letras Modernas. Lo primero que se hace es un select a la tabla "materia" para saber si la asignatura existe, si pertenece a algún plan de Letras Modernas y que tenga un marcador que la identifique en algún nivel de la seriación. Si el resultado de este select es mayor a 0 entonces se hace otro select a la tabla "seriac_mod" y materia donde se va a verificar el nivel de avance que tiene el alumno y en caso que sea igual o mayor al nivel con que está marcada la asignatura entonces su inscripción sí procede; en caso contrario no procede y se manda a pantalla un mensaje de error.
function es_mat_opta_hist($conex, $asgn ,$gpo ,$plan)	Esta función sirve para validar que los alumnos pertenecientes al plan 0833 de Historia sólo puedan inscribir asignaturas obligatorias (menos optativas); mediante un select a la BD verifica que la asignatura tenga tipo de créditos obligatorios, sólo en ese caso procede el registro de la asignatura; en caso contrario se manda a pantalla un mensaje de error.
function asign_oblig_latinos ($conex, $asgn, $gpo, $num, $plan)	Esta función valida que un alumno del Colegio de Estudios Latinoamericanos

	generación 2004 y 2005 no pueda inscribir asignaturas optativas.
function curso_basic_clasicas ($conex, $asgn ,$num ,$plan)	Esta función valida que los alumnos del Colegio de Letras Clásicas no se puedan inscribir a los cursos básicos de su plan de estudios.
function cupo_grupo ($conex, $asgn, $gpo)	Esta función verifica el cupo que tiene declarado cada grupo ya sea cupo propio o compartido; mediante un select a la BD obtiene el cupo declarado para cada asignatura y el resultado lo guarda en un arreglo.
function ocupacion ($conex, $asgn, $gpo, $p_c)	Esta función hace una búsqueda a la BD para determinar el número de ocupación de cada asignatura grupo y así poder llevar un control de cupos.
function guarda_inscr ($conex, $numero, $plan, $asign, $grupo, $prop_comp)	Esta función guarda las asignaturas grupo en la inscripción del alumno, para esto hace uso de la función cupo_grupo ($conex, $asign, $grupo) y de la función ocupacion ($conex, $asign, $grupo, $prop_comp) en base a estas dos funciones determina si la inscripción procede o no. En el segundo caso se manda a pantalla un mensaje de error avisándole que ya no hay cupo en esa asignatura grupo
function ya_seleccionada ($conex, $numero, $plan, $asig)	Esta función verifica que el alumno no registre más de una vez la misma asignatura grupo.
function asign_ya_acreditada($conex, $asgn, $num)	Esta función va a validar que el alumno no pueda inscribir una asignatura que ya tenga aprobada en periodos anteriores (hace la validación contra historias académicas)
function asign_no_aprobada ($conex, $asgn, $num)	Esta función va a validar que el alumno ya haya registrado la misma asignatura 2 o más veces en periodo ordinario sin aprobarla. Si se cumple este caso el alumno no podrá registrar la asignatura grupo y se mandará a pantalla un mensaje de error; en caso contrario si procederá su registro.
function planes_mat_escol ($conex, $asign, $grupo)	Esta función verifica que la asignatura grupo que el alumno va a registrar

	pertenezca a un plan de estudios del sistema escolarizado.
function carrera_mat_escol ($conex, $asign, $grupo)	Esta función verifica que la asignatura grupo que el alumno va a dar de alta pertenezca al sistema escolarizado
function planes_mat_escol_cerr ($conex, $plan, $asign, $grupo)	Esta función valida que la asignatura grupo que el alumno va a dar de alta exista
function checa_carr($carreras, $seleccion)	Esta función verifica que una carrera exista
function cancela_inscripcion ($conex, $num_cta, $plan)	Esta función verifica que una asignatura grupo no esté dada de alta más de una vez; en caso de ya existir manda a pantalla un mensaje de error, en cuyo caso no procede el registro.
function cierra_sesion ($conex, $id_sesion, $causa)	Esta función hace un update dentro de la tabla sesión al campo hr_termino, en el momento en el que hay una causa de término del proceso.
function formulario_materias ($action, $ident_sesion, $maxim)	Esta función contiene un formulario escrito en código html; contiene dos cuadros de texto donde se ingresa la clave de la asignatura y la clave del grupo que se va a dar de alta; tiene un botón para enviar los datos, otro para cancelar las asignaturas grupo inscritas y uno para terminar el proceso.

plan_inc.php

Este programa contiene un catálogo de todas las carreras impartidas en la facultad con sus respectivos planes de estudio y cada uno está asignado a una variable, cuyo valor es constante y es usado en los otros programas.

tpos.php

Este programa es el que lleva el control en cuanto a los tiempos de acceso al sistema, ya que se hace mediante una calendarización por carrera, día y hora en periodo de reinscripción así como en periodo de altas, bajas y cambios, excepto para el periodo de registro a exámenes extraordinarios.
Los formatos que se manejan para las fechas son en "timestamp" y para cada carrera se asigna a variables la hora de inicio y fin, lapso en el cual se van a poder registrar los alumnos pertenecientes a la carrera que le toca inscribirse ese día.

movimientos.php
Este programa funciona con base en los programas antes descritos

1. Se establece una conexión a la BD; si la conexión falla me manda a pantalla un mensaje de error avisando que hay problemas para conectarse a la BD
2. Se asigna a una variable un identificador único de la sesión que abrió el alumno al sistema. Mediante la función valida_sesion asignada a una variable, valida que la hora en que ingresó el alumno esté dentro de los límites establecidos, en caso de no ser así se cierra esa sesión, pero en caso contrario se asignan los valores de número de cuenta y plan del alumno a variables.
3. Se definen las causas de término de sesión (cancelar, terminar, confirmado)
4. De acuerdo al proceso que se esté ejecutando (ordinario o extraordinario) se define el número de asignaturas grupo que el alumno puede cursar de acuerdo a determinados criterios de cada carrera
5. En caso de que se trate de un plan cerrado se verifica que la asignatura grupo que seleccionó el alumno sea de su plan.
6. Se mandan a llamar a las funciones $es_cerrado , $es_vjo_nvo , es_mat_opta_hist , es_idioma, asign_ya_acreditada, asign_no_aprobada, asign_oblig_latinos, curso_basic_clasicas. Se valida cada condición y en caso de que no se cumpla alguna se manda un mensaje, dependiendo de la condición que no se haya cumplido.
7. Se manda llamar la función asign_gpo_valido; dicha función es propia de la base de datos, ejecutándose esta función determina si un alumno puede o no inscribirse de acuerdo a determinados requisitos (Taller de Redacción de Letras Hispánicas, Griego y Latín de Letras Clásicas).
8. se crea la función formulario_ok($id, $indicador) ; dicha función se activa dependiendo del botón que se pulse en pantalla (alta o baja), si es alta las botones siguientes son para dar altas, y si el botón pulsado es para dar de baja, los siguientes botones que aparecerán en pantalla serán para dar de baja.

registro.php

Este programa hace uso de los programas antes descritos.

1. Manda llamar a la función formato_correcto
2. Hace la conexión a la base de datos
3. en caso de que los valores que trae la función formato_correcto sean incorrectos, manda a pantalla mensajes de error avisando el error.
4. En caso de que el alumno tenga una causa de exalumno diferente a 20, 11 o null, dicho alumno no se va a poder inscribir y se le manda un mensaje en pantalla avisándole el motivo. Esto se hace llamando a la función planes_alumno.
5. Se define la función formulario_entrada ($action, $num, $mensaje_error), la cual contiene dos cuadros de texto, uno para teclear el número de cuenta y el otro para teclear el NIP, un botón para enviar los datos y otro para limpiar los campos .
6. Se define la función function formulario_materias_inicial($conex, $id, $num_cta, $plan), que manda un mensaje para los alumnos que deben registrar movimientos en su coordinación, mediante la variable PLANES_OUT, ejecuta la función valida_fecha_plan y valida_fecha_gen
7. Se define la función function mensaje_de_pausa_en_sistema() ; esta función manda a pantalla un mensaje avisando que no se pueden realizar trámites porque está fuera de las fechas especificadas.
8. Se define : function verifica_ingresos ($conex, $num, $cve, $id) esta función va haciendo el computo del número de veces que un alumno ha ingresado al sistema y en caso de ser mayor a 3 veces el número de sesiones entonces manda a pantalla un mensaje avisándole al alumno que sólo tiene derecho a ingresar 3 veces.
9. Se define: function valida_fecha_gen($conex, $id, $num, $cve) , esta función valida por generación quién tiene derecho a ingresar al sistema.
10. Se define: function valida_fecha_plan($conex, $id, $cve), esta función valida por fecha (día) quién tiene derecho a ingresar al sistema.

En este programa es donde se define cada proceso (registro a reinscripción, registro de altas, bajas y cambios y finalmente el registro a exámenes extraordinarios) y de acuerdo a cada registro se sigue un proceso diferente

En caso de que el proceso sea cambios de grupo:

1. Verifica que el número de cuenta esté en la tabla ENTRO, en caso de no estar manda un mensaje a pantalla
2. Verifica el número de sesiones del alumno.
3. Imprime en pantalla las asignaturas inscritas.
4. Ejecuta la función formulario_cambios, para que el alumno dé altas o bajas a su inscripción.
5. Cierra la conexión a la base de datos.

En caso de que el proceso sea reinscripción:

1. Ejecuta la función mat_inscritas y cuenta el número de asignaturas inscritas, si el número es igual a cero ejecuta la función formulario_materias.
2. En caso de que el alumno ya tenga más de una asignatura registrada se ejecutan las funciones cierra_sesion, imprime_final.

En caso de que el proceso sea igual a extraordinario:

1. Se ejecuta la función verifica_ingresos
2. Se ejecuta la función mat_inscritas y se cuentan las asignaturas inscritas. Si tiene más de una asignatura inscrita se ejecuta la función imprime_inscritas, en caso contrario se ejecuta la función formulario_mat_extra.

Para cada semestre lo único que se tiene que reprogramar es la calendarización de fechas de inscripción por carrera y generación, en el programa **entrada.php** y se delimitan las fechas en el archivo **tpos.php**, se actualizan los datos del periodo en el archivo **comun.php** y se descomenta la pausa del sistema en el archivo **registro.php**.

5.2 ESTRUCTURA DE LA PÁGINA WEB DEL SIFYL

PÁGINA	REFERENCIA A	CONTIENE
index.html (frame)	encabezado.html	
	menu.html	
	principal.html	
encabezado.html	UNAM.swf	
	SISTEMAS.swf	
principal.html	ANIME.swf	
	17semanas.swf	
menu.html	Horarios_ord.html	0303.html 0311.html,0560.html 0304.html,0316.html ,833.html 0305.html,0317.html,0836.html 0306.html,0318.html,0837.html 0307.html,0319.html,1104.html 0308.html,0325.html,1114.html 0309.html,0326.html,letras.html
	Horarios_extra.html	0303_ext.html , 0311_ext.html 0560_ext.html , 0304_ext.html 0316_ext.html , 0833_ext.html 0305_ext.html , 0317_ext.html 0836_ext.html , 0306_ext.html 0318_ext.html , 0837_ext.html 0307_ext.html , 0319_ext.html 1104_ext.html , 0309_ext.html 0326_ext.html , 0308_ext.html 0325_ext.html , 1114_ext.html mod_ext.html
	inst-EA-042.htm	
	registro.php	
registro.php	movimientos.php	
	comun.php	
	entrada.php	
	materias.php	
	tpos.php	
	plan_inc.php	

5.3 SUBSISTEMAS ALTERNOS

El manejador de bases de datos que se utilizó para desarrollar la base de datos, es Postgres. Los subsistemas están desarrollados en Perl y son de vital importancia ya que con éstos se actualiza la información contenida en la base de datos y se generan varios reportes con la información contenida en la misma.

A continuación se detallan los subsistemas:

PROGRAMA (SUBSISTEMA)	DESCRIPCION
cambia_grado.pl	Este programa actualiza la tabla: • profesor para cambiar el grado que tenga.
cambios_gpoX_a_gpoY.pl	Este programa modifica las tablas: • inscr • grupo se ejecuta cuando se tienen que modificar alguna asignatura grupo y ya hay alumnos inscritos en la misma.
listado_noprocede.pl	Este programa genera un reporte de aquellos alumnos a los cuales no les procedió ninguna asignatura. Para generar el reporte toma un archivo en formato texto que contiene los números de cuenta de los alumnos a los cuales no les procedió alguna asignatura y con estos datos hace una búsqueda a la base de datos y genera el reporte , guardándolo en : /home/jcarlos/escolar/perl/datos
carga_clavenac.pl	Este programa actualiza la tabla: • alumnos modificando su estado de nacimiento, si es que se requiere. El archivo que contiene los datos debe estar ubicado en el directorio : /home/jcarlos/escolar/perl/datos
archivos_salida.pl	Este programa genera los archivos de la inscripción que se envían a DGAE divididos en asignaturas del sistema escolarizado compartidas y propias y del sistema SUA propias.

	Los archivos los genera el formato texto y en el directorio : /home/jcarlos/escolar/perl/datos
carga_datosper.pl	Este programa actualiza la tabla: • alumnos en caso que los datos del alumnos hayan cambiado; también sube los datos de los alumnos de primer ingreso. El archivo de los alumnos de primer ingreso debe estar en el directorio: /home/jcarlos/escolar/perl/datosper
carga_deudores.pl	Con este programa se cargan a la tabla: • deudores los datos de los alumnos que deben libros, documentos, etc.
rep_seg_sim.pl	Programa para realizar la carga y verificación de las tablas de “alumnos”, “alum_carr”, “alum_acad” para alumnos de segundas y simultaneas.
horarios_extra_html.pl	Este programa genera los archivos html de los horarios a exámenes extraordinarios, haciendo una conexión a la base de datos y tomando los datos de las tablas grupo, curso, horarios, plan, asignatura entre otras. Los archivos html los genera en el directorio : /home/jcarlos/escolar/perl/datos
carga_directorio.pl	Este programa se ejecuta cada vez que se va a actualizar el directorio de los alumnos inscritos en la facultad. Con este programa se modifican las tablas : • alumnos • alum_carr • alum_acad Si el número de cuenta ya está dado de alta en la tabla “alumnos” solo se modifican las tablas “alum_carr” y “alum_acad”. Los archivos a utilizar deben estar en formato txt y su ubicación debe ser en : */home/jcarlos/escolar/perl/datos/dir*
horarios_ord_html.pl	Este programa genera los archivos html de los horarios, haciendo una conexión a la base de datos y

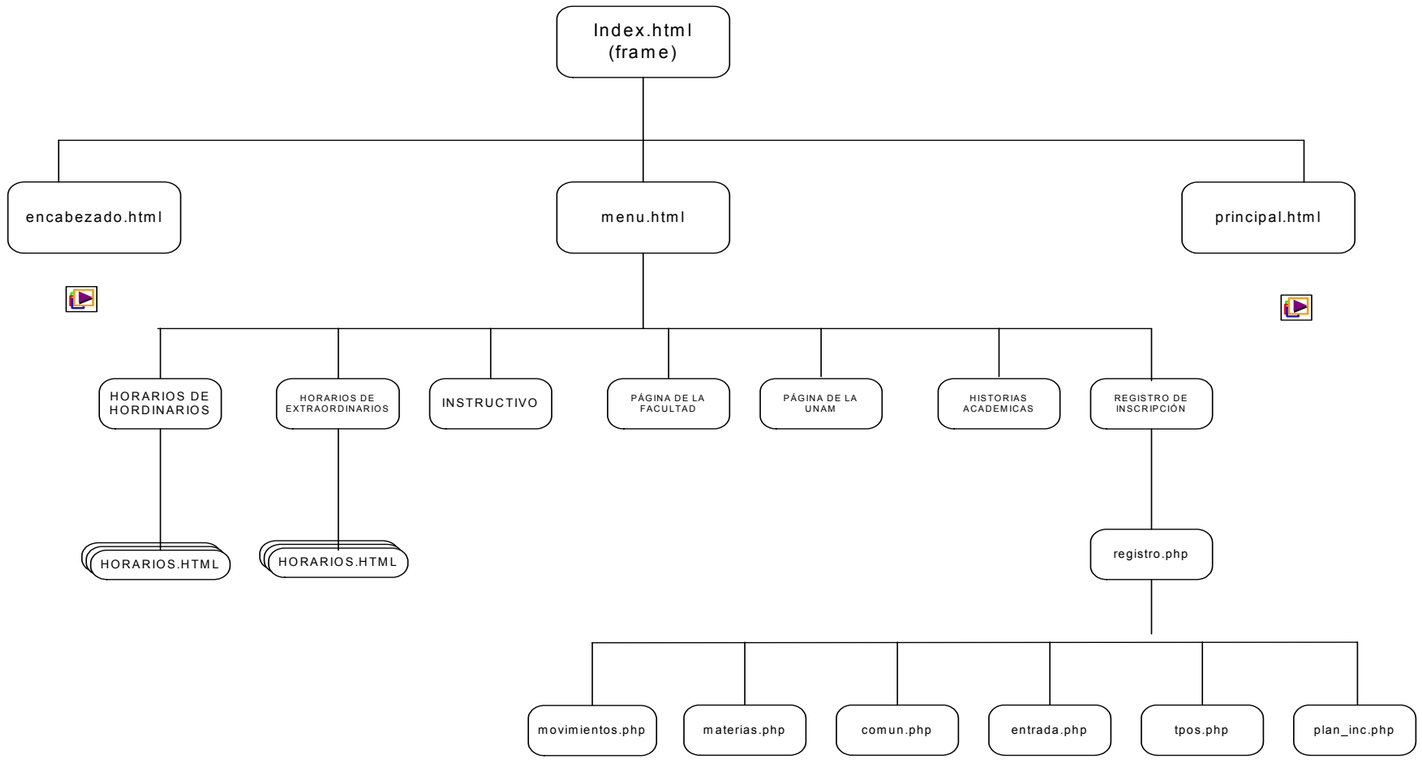
	tomando los datos de las tablas “grupo”, “curso”, “horarios”, “plan”, “asignatura” entre otras. Los archivos html los genera en el directorio : /home/jcarlos/escolar/perl/datos
carga_limpia.pl	Este programa se ejecuta para realizar el llenado de las tablas : • curso • grupo • maestr_gpo • horario (en ese estricto orden). Entre otras cosas también hace la actualización de la tabla “avance_re”, “seriac_mod”, ”num_hispan” (alumnos que ya aprobaron el Taller de Redacción) y la tabla deudores. Los archivos con la información que se va a subir a la base de datos deben estar en formato txt y su ubicación debe estar en el directorio: /home/jcarlos/escolar/perl/datos
carga_hist-acad.pl	Este programa se ejecuta en cada periodo, con la finalidad de tener las historias académicas actualizadas. La tabla que actualiza es: • hist_acad debe estar vacía, los archivos que contienen las historias académicas deben estar en formato txt y su ubicación debe ser : /home/jcarlos/escolar/perl/hist los registro que no se puedan subir a la tabla los guarda en un archivo ubicado en el mismo directorio.
carga_seguimientos.pl	Este programa actualiza las tablas : • candados • gpo_seg • grupo Sólo se ejecuta en periodos de inscripción par. En la primera tabla carga los números de cuenta y otros datos de aquellos alumnos que deben tener un seguimiento de grupo. En la tabla “gpo_seg” se cargan las asignaturas grupo que tienen seguimiento y en la tabla “grupo” solo se actualiza el campo cve_req. El archivo que contiene la información para hacer la actualización de las tablas debe tener formato texto y debe estar en el directorio:

	/home/jcarlos/escolar/perl/datos
borra_avance_re.pl	Este programa borra registro de la tabla: • avance_re mediante un archivo en formato texto ubicado en el directorio: /home/jcarlos/escolar/perl/datos
sube_tpo.pl	Este programa actualiza la tabla : • materia marcando las asignaturas que tienen determinado nivel de seriación. El archivo para hacer esta actualización debe estar en formato texto y debe estar ubicado en el directorio: /home/jcarlos/escolar/perl/datos
suma_cupos.pl	Este programa suma los cupos definidos en la tabla : • curso tomando los nuevos cupos de un archivo con formato texto que se encuentra en el directorio: /home/jcarlos/escolar/perl/datos
cambia_cupos.pl	Este programa cambia los cupos definidos en la tabla : • curso tomando los nuevos cupos de un archivo con formato texto que se encuentra en el directorio: /home/jcarlos/escolar/perl/datos
carga_prim-ing.pl	Este programa carga los datos de los alumnos de primer ingreso actualizando las tablas: • alumnos • alum_carr • alum_acad El archivo que contiene los datos debe tener formato texto y debe estar en el directorio: /home/jcarlos/escolar/perl/datos
imprime.pl	Este programa hace búsquedas a la base de datos para generar los comprobantes de inscripción definitivos. Los archivos de salida que genera son en formato dvips, es decir son archivos independientes de los dispositivos; por lo tanto se hace uso del paquete Tetex Latex para convertir esos archivos a formato postcript (ps) y así poder imprimirlos. Los archivos generados se guardan en el directorio: /home/jcarlos/escolar/perl/datos

sin_inscripcion.pl	Este programa se ejecuta para hacer una búsqueda de los alumnos a los que no les procedió ninguna asignatura grupo. Con los datos encontrados genera un archivo en formato texto y lo guarda en el directorio: /home/jcarlos/escolar/perl/datos
inscr_proced_2.pl	Este programa carga a la tabla: • inscr la inscripción que no procedió, para hacer esto el archivo debe estar en formato texto y su ubicación debe estar en el directorio: /home/jcarlos/escolar/perl/datos

NOTA: éstos subsistemas se ejecutan en la maquina PROMETEO haciendo conexiones remotas a la base de datos que está en el servidor SÓCRATES y/o BDSERV.

5.4 DIAGRAMA DE PÀGINA WEB



6 DISEÑO DE LA BASE DE DATOS

6.1 DICCIONARIO DE DATOS DE LA BASE DE DATOS "ESCOLAR"

La base de datos del **SIFYL** consta de 41 tablas, las cuales se describen a continuación:

TABLA: ***alum_acad***
Contiene los datos académicos anteriores (del bachillerato) al ingresar a la carrera del alumno así como el plan de la carrera a la que ingresó en la facultad de Filosofía y letras.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
num_cta	Char(9)	X	**Llave primaria:** Número de Cuenta del alumno	096180051
cve_plan	Char(4)	X	**Llave primaria**: Clave del plan del alumno	0560
cve_proced	Char(1)		Clave de la escuela de procedencia del alumno	8
cve_pl_proced	Char(3)		****	null
cve_plantel	Char(3)		Clave del plantel del alumno (únicamente perteneciente a la UNAM)	035
cursados	smallint			
promedio	numeric(2,2)		Promedio del alumno al terminar su bachillerato el rango va del 0 al 10	9.00
examen	numeric(2,2)			
cve_ingr	Char(2)		Clave de ingreso	56
cve_opingr	Char(1)		Número de opción de ingreso	

TABLA: ***alum_carr***
Contiene los datos de la (s) carrera (s) a la(s) que ingresó el alumno

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
num_cta	Char(9)	X	**Llave primaria**: número de Cuenta del alumno	096180051
cve_plan	Char(4)	X	**Llave Primaria:** clave del plan del alumno	0560
ingreso	Char(4)	X	Año en que ingresó el alumno a la carrera	1999
status	Char(1)		Estatus académico del alumno puede ser: • 'A' para alumnos activos • 'B' para alumnos bloqueados • 'S' para alumnos con suspensión temporal	A
cve_exalum	Char(2)		Contiene la clave de exalumno en caso de que ya no sea un alumno activo	06
art_19_22	Char(5)	X	Semestre a partir del cual el alumno es Art. 19.	20042

TABLA: ***alumnos***
Directorio de datos personales de los alumnos de Facultad de Filosofía y Letras.

Campo	Tipo	Nulo	Descripción	Ejemplos de instancias
num_cta	char(9)	X	**Llave primaria**: número de Cuenta del alumno	096180051
paterno	char(25)		Apellido paterno del alumno	GONZALES
materno	Char(25)		Apellido materno del alumno	HERNÁNDEZ
nombre	char(25)		Nombre del alumno	CLAUDIA
sexo	char(1)		Sexo del alumno, puede ser: • M masculino • F femenino • S sin sexo	F
calle	char(38)		Calle del domicilio del alumno	LAS ROSAS # 17
colonia	char(38)		Colonia en la que vive el alumno	SANTA MARTA
cve_deleg	char(2)		Clave de la delegación en la que vive el alumno	13
cp	char(5)		Código postal del alumno	13600
telef	char(10)		Teléfono del alumno	58471278
fech_nac	char(8)		Fecha de nacimiento del alumno (ddmmaaaa)	30061980
edo_nacim	char(2)		Estado de nacimiento del alumno	01
nom_padre	char(45)		Nombre del padre o tutor del alumno	Federico Ríos González
nacionalidad	char(1)		Nacionalidad del alumno	2
ciudad	char(25)		Ciudad en que vive actualmente el alumno	null
pais_origen	char(10)		País de origen del alumno	null

TABLA: ***asignatura***
Catálogo de asignaturas impartidas en la Facultad de Filosofía y Letras .

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_asign	char(4)	X	**Llave primaria**: contiene la clave de la asignatura	0156
asignatura	char(40)	X	Nombre de la asignatura	CURSO BASICO TRADUCCION LATIN II

TABLA: ***avance_hispan***
Contiene datos de los alumnos inscritos en la carrera de Letras Hispánicas que ya tienen determinado avance de créditos.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
num_cta	char(9)	X	**Llave primaria:** número de Cuenta del alumno	049734616
cve_plan	char(4)	X	**Llave Primaria:** clave del plan del alumno	0311
cve_plt	char(3)		Clave del plantel al que pertenece en este caso Facultad de Filosofía y Letras	010
inscrit	Boolean		Valor que determina si está o no inscrito en la carrera; puede ser ***F ó T***	T
aingr	char(4)		Año en que ingresó el alumno a la carrera	1983
cingr	char(2)		Clave de ingreso del alumno	56
cred_obl	Smallint		Créditos de asignaturas obligatorias	0
porc_obl	numeric(5, 2)		Porcentaje del avance de las asignaturas obligatorias	0.0
cred_opt	Smallint		Créditos de las asignaturas optativas	0
porc_opt	numeric(5, 2)		Porcentaje del avance de las asignaturas optativas	0.0
cred_tot	Smallint		Suma de los créditos totales	0
porc_tot	numeric(5, 2)		Porcentaje del total de avance	0.0
num_rep	Smallint		Número de asignaturas reprobadas	2
promedio	numeric(4, 2)		Promedio del alumno	0
Ha_pr	char(5)		**	19831
Ha_ul	char(5)		**	19832

TABLA: ***avance_re***
Esta tabla contiene los números de cuenta de los alumnos que ya tienen más del 75% de avance de créditos y un promedio mínimo de 8.0; se utiliza para permitir que se puedan registrar hasta 6 extraordinarios por semestre, ya que el reglamento indica que sólo pueden registrar 2.

Campo	Tipo	Nulo	Descripción	Ejemplos de instancias
num_cta	char(9)	X	**Llave primaria:** número de Cuenta del alumno	049734616
cve_plan	char(4)	X	**Llave Primaria:** clave del plan del alumno	0311
promedio	Smallint		Promedio del alumno	9.0
avance	Smallint		Avance de créditos del alumno	80

TABLA: ***bloqueado***
Datos de los alumnos bloqueados y/o con avisos.

Campo	Tipo	Nulo	Descripción	Ejemplos de instancias
num_cta	char(9)	X	**Llave primaria:** número de Cuenta del alumno	049734616
consec	smallint	X	**Llave primaria:** consecutivo del bloqueo	205
cve_mov	char(2)	X	Clave del movimiento	11
fecha	char(8)	X	Fecha de alta en la tabla	06101994
num_aviso	char(1)		Número de aviso	1

TABLA: ***calif***
Este es un catalogo que contiene todos los posibles valores de una calificación

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_calif	char(2)	X	**Llave primaria:** contiene el valor numérico o con letra de la calificación (2 dígitos)	MB
leyenda	char(20)		Contiene una descripción del significado de la cve_calif	Muy Bien
aprob	boolean	X	Determina si el valor de la cve_calif es aprobatorio (t) o reprobatorio (f)	T
promediar	boolean	X	Determina si el valor de la cve_calif se considera para obtener el promedio del alumno	T
valor	smallint		En caso de que el valor de la calificación sea numérico se estable con un dígito	7

TABLA: ***candados***
Esta tabla contiene información de los alumnos que deben tener un seguimiento de grupo, así como de los alumnos de Letras Clásicas que no pueden inscribirse a las asignaturas de Griego y Latín

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_asign	char(4)	X	**Llave primaria**: clave de la asignatura	1296
cve_gpo	char(4)	X	**Llave primaria:** clave del grupo	0021
num_cta	char(9)	X	**Llave primaria**: número de Cuenta del alumno	094214653
cve_plan	char(9)	X	**Llave Primaria:** clave del plan del alumno	0319
asign_req	char(4)		Contiene la clave de la asignatura que el alumno cursó un semestre anterior y que tiene seguimiento con la cve_asign	1288
gpo_req	char(4)		Contiene la clave del grupo que el alumno cursó un semestre anterior y que tiene seguimiento con la cve_gpo	0021

TABLA: ***carrera***
Catálogo de Carreras que se imparten en Facultad de Filosofía y Letras.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_siae	char(3)	X	**Llave primaria:** contiene la clave de cada carrera de acuerdo a la asignación que hace el SIAE	307
carrera	char(45)	X	Contiene el nombre de cada carrera	Geografía

TABLA: ***coordinación***
Esta tabla contiene el catálogo de las claves de las coordinaciones

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_coord	char(2)	X	**Llave primaria:** contiene la clave de la coordinación	01
coordinación	Char(45)	X	Nombre de la coordinación	BIBLIOTECOLOGIA
rfc	Char(10)		Contiene el RFC del coordinador	RAVC550105

TABLA: ***curso***
Esta tabla contiene la clave de curso a la que corresponde cada asignatura grupo, es la tabla principal de las tablas (horario, maestr_gpo y grupo)

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_curso	smallint	X	**Llave primaria:** es el número consecutivo del valor del último registro de la tabla	5810
cve_tem	char(3)			
cupo_prop	Smallint		Número de lugares disponibles para alumnos de la misma carrera	50
cupo_com	smallint		Número de lugares disponibles para alumnos de otras carreras	10

TABLA: ***delegación***
Catálogo de delegaciones.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_deleg	char(2)	X	**Llave primaria**: clave de la delegación a la que pertenece el alumno	13
delegación	char(38)	X	Nombre de la delegación	TLAHUAC

TABLA: ***descriptor***
Catálogo con información adicional de las materias con respecto a un plan.

Campo	Tipo	Nulo	Descriptor	Ejemplos de instancias
Cve_plan	Char(4)		Clave del plan	0833
cve_desc	Char(2)	X	**Llave primaria**, clave de la descripción	11
descripción	Char(40)		Nombre de la descripción	OPTATIVA AREA DE CONOCIMIENTO HISTORICO.

TABLA: ***entro***
Esta tabla contiene los números de cuenta de los alumnos que realizaron algún registro en su inscripción y que por lo tanto tienen derecho a realizar movimientos en el periodo de altas bajas y cambios.

Campo	Tipo	No nulo	Descriptor	Ejemplos de instancias
num_cta	char(9)	X	**Llave primaria:** número de cuenta del alumno	055167633

TABLA: ***exalumno***
Catálogo de claves de los tipos de bajas de alumnos

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_exalum	char(2)	X	**Llave primaria**: clave de exalumno	41
exalumno	char(80)	X	Descripción de la clave de exalumno	BAJA DEFINITIVA POR ASIGNATURAS FUERA DE SU PLANTEL Y/O CARRERA

TABLA: ***fallos***
Catálogo del tipo de error producido al intentar la inscripción de una asignatura cuando ésta no procede.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_error	char(5)	X	**Llave primaria:** clave del error	21301
error	char varying		Descripción del error	ASIGNATURA YA ACREDITADA

TABLA: ***gpos_seg***
Esta tabla contiene las asignaturas grupo que tienen seguimiento (sólo se utiliza en los semestres pares)

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
asign_req	char(4)	X	**Llave primaria**: clave de la asignatura que tiene seguimiento en el semestre par	1288
gpo_req	char(4)	X	**Llave primaria**: clave del grupo que tiene seguimiento en el semestre par	0021

TABLA: ***grupo***
Registro de las asignaturas-grupo de cada semestre (ya sea para ordinarios como para extraordinario)

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_asign	char(4)	X	**Llave primaria:** clave de la asignatura	1296
cve_gpo	char(4)	X	**Llave primaria:** clave del grupo	0021
cve_curso	Smallint		Es el número consecutivo del valor del último registro de la tabla	1435
cve_coord	char(2)	X	Clave de la coordinación	07
cve_req	char(4)		Clave del requisito	null

TABLA: ***hist_acad***
Esta tabla contiene las historias académicas de los alumnos de la Facultad de Filosofía y Letras.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
num_cta	char(9)	X	**Llave primaria**: número de cuenta del alumno	040293251
cve_plantel	char(3)	X	Clave del plantel	010
cve_siae	char(3)	X	Contiene la clave de cada carrera de acuerdo a la asignación que hace el SIAE	419
cve_plan	char(4)	X	**Llave primaria:** clave del plan del alumno	0310
cve_ingr	char(2)	X	Clave de ingreso del alumno	59
cve_asign	char(4)	X	**Llave primaria:** clave de la asignatura	2262
periodo	char(5)	X	**Llave primaria:** periodo en el que curso la asignatura el alumno	19911
cve_calif	char(2)	X	Calificación del la asignatura	B
cve_gpo	char(4)	X	**Llave primaria:** clave del grupo	0021
folio	char(7)	X	**Llave primaria:** número que identifica a cada asignatura grupo	1434702
tpo_exam	char(1)	X	**Llave primaria:** proceso en el que alumno cursó la asignatura , puede ser ordinario o extraordinario	O

TABLA: ***horario***
Datos de los horarios de los cursos del semestre en curso

Campo	Tipo	No nulo	Descripción	Ejemplos de instancia
cve_curso	smallint	X	**Llave primaria:** es el número consecutivo del valor del último registro de la tabla	5810
consec	smallint	X	**Llave primaria:** número consecutivo por grupo	1
cve_prof	Char(10)	X	RFC del profesor	PALF370916
cve_salon	Char(3)	X	Clave de salón	PDT
día	Char(1)	X	Día en que se imparte esa asignatura, en valor numérico del 1 al 6 (lunes ,....sábado respectivamente)	1
hora_ini	Char(2)		Hora de inicio del curso	10
hora_fin	Char(2)		Hora final del curso	12
fech_exam	Char(8)		Fecha en que se realizará el examen; sólo cuando el periodo es de exámenes extraordinarios	19042004

TABLA: ***ingreso***
Catálogo de tipos de ingreso.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_ingr	char(2)	X	**Llave primaria:** clave de ingreso del alumno	56
ingreso	char(60)		Descripción de la clave	INGRESO A NIVEL LICENCIATURA CONCURSO DE SELECCIÓN

TABLA: ***inscr***
Datos de las materias registradas por los alumnos en la solicitud de inscripción del semestre en curso, ya sea a ordinario o a extraordinario; en caso de no proceder se registra la causa.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
num_cta	char(9)	X	**Llave primaria:** número de cuenta del alumno	095584658
cve_plan	char(4)	X	**Llave primaria:** clave del plan del alumno	0304
cve_asign	char(4)	X	**Llave primaria: c**lave de la asignatura	2599
cve_gpo	char(4)	X	**Llave primaria** Clave del grupo	EA92
prop_comp	char(1)		'P' : asignatura propia al plan del Alumno 'C' : asignatura compartida	P
cve_error	char(5)		Clave del error por el cual no procedió la inscripción	null

TABLA: ***maestr_gpo***
Datos de los profesores que imparten lo cursos

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_curso	smallint	X	**Llave primaria:** es el número consecutivo del valor del último registro de la tabla	1
cve_prof	char(10)	X	**Llave primaria:** RFC del profesor	REAE341115
tit_sup	char(1)		Profesor titular o suplente. '1' titular '2' suplente	0

TABLA: ***materia***
Catálogo de materias de todos los planes de estudio.

Campo	Tipo	Nulo	Descripción	Ejemplos de instancias
cve_plan	char(4)	X	**Llave primaria:** clave del plan de la asignatura	0836
cve_asign	char(4)	X	**Llave primaria: c**lave de asignatura	0630
créditos	Smallint	X	Número de créditos	4
semestre	Smallint	X	Semestre en que se imparte la asign (((semestre >0) and (semestre <15)) or (semestre =40 : optativa))	40
cve_desc	char(2)		Clave de descripción	null
tpo_cred	char(1)	X	Créditos obligatorios u optativos: '0' : obligatoria ‘1' : optativa	1
tpo_asign	char(2)	X	Determina si la asignatura es obligatoria u optativa : '00' : obligatoria ‘01' : optativa	01
tpo_curso	char(1)	X	Determina el tipo de asignatura : ‘1’ : asignatura normal ‘2’ : seminario ‘3’ : idioma ‘4’ : taller ‘5’ : laboratorio	1
hrs_teo	Smallint		Horas teóricas a la semana	0
hrs_prac	Smallint		Horas prácticas a la semana	20
tpo_sub	char(1)		Tipo de sub área a la que pertenece la asignatura	null

TABLA: ***movimiento***
Catálogo de los tipos de adeudos de los alumnos

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_mov	char(2)	X	**Llave primaria**: clave del movimiento	00
movimiento	char(40)	X	Descripción del movimiento	ADEUDO DE LIBRO EN BIBLIOTECA

TABLA***: num_cta_seg***
Esta tabla contiene información respecto al seguimiento de grupos

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
num_cta	char(9)	X	**Llave primaria**: número de cuenta del alumno	096210758
cve_plan	char(4)	X	**Llave primaria**: clave del plan del alumno	0318
asign_req	char(4)	X	**Llave primaria**: clave de la asignatura que tiene seguimiento en el semestre par	0189
gpo_req	char(4)	X	**Llave primaria:** clave del grupo que tiene seguimiento en el semestre par	0021

TABLA: ***num_hispan***
Esta tabla contiene el listado de alumnos inscritos en la carrera de Letras Hispánicas que ya aprobaron el Taller de Redacción.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
num_cta	char(9)	X	**Llave primaria**: número de cuenta del alumno	098606283
cve_plan	char(4)	X	Clave del plan del alumno	0837

TABLA: ***plan***
Catálogo de planes de estudio impartidos en la Facultad de Filosofía y Letras.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_siae	char(3)	X	Contiene la clave de cada carrera de acuerdo a la asignación que hace el SIAE	402
cve_plan	char(4)	X	**Llave primaria:** clave del plan al que pertenece la carrera	1104
cve_sistema	Char(1)	X	Clave del sistema al que pertenece el plan de estudios, puede ser : • 'E': sistema escolarizado • 'A' :sistema abierto	E
cred_oblig	smallint	X	Número de créditos obligatorios que debe cubrir el plan de estudios	324
cred_opta	smallint	X	Número de créditos optativos que debe cubrir el plan de estudios	36
inic_vigen	Char(4)		Año en que entró en vigencia el plan	2003
vigente	Char(1)	X	Clave de vigencia	4
duración	smallint	X	Duración semestral del plan	8
abierto	Char(1)	X	Determina el tipo de plan: • '0': plan cerrado • '1': plan abierto	1
cve_coord	Char(2)	X	Clave de la coordinación	01

TABLA: ***plantel***
Catálogo de Planteles de la UNAM.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_plantel	char(3)	X	**Llave primaria**: clave del plantel	010
plantel	char(42)	X	Nombre del plantel	FACULTAD DE FILOSOFIA Y LETRAS

TABLA: ***procedencia***
Catálogo de escuelas de nivel medio superior de procedencia.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_proced	char(1)	X	**Llave primaria:** clave de la escuela de procedencia del alumno	1
procedencia	char(25)		Nombre de la escuela	PUBLICA

TABLA: ***proceso***
Almacena los datos necesarios para el proceso en curso.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
periodo	char(5)	X	**Llave primaria**: semestre escolar en curso	20042
proceso	char(1)	X	**Llave primaria**: tipo de proceso en curso, puede ser: • 'E': extraordinario • 'R': ordinario • 'C': altas, bajas y cambios	E
cve_ext	char(1)	X	**Llave primaria:** clave del tipo de examen en curso, puede ser: • 'EA' : extraordinario 'EA' • 'EB' : extraordinario 'EB'	A
encurso	Boolean	X	Determina si el proceso está en curso, puede ser: • 'T' : verdadero • 'F' : falso	T
escuela	char(3)	X	Clave de la facultad	010

TABLA: ***profesor***
Esta tabla contiene información de los profesores que imparten cursos en la Facultad.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_prof	char(10)	X	**Llave primaria:** RFC del profesor	CADA010101
homoclave	char(3)		Anexo del RFC del profesor pero que para fines prácticos no se usa	null
grado	char(6)		Grado académico del profesor, puede ser: 'ARQ.','BIOL.','DR.','DRA.',' ESP. 'MED.C.','FIS.','ING.', 'MTRA.','LIC.','MTRO.', 'PASNT.','PROF.','PROFA	DR.
paterno	char(17)		Apellido paterno del profesor	CAZES
materno	char(17)		Apellido materno del profesor	MENACHE
nombre	char(17)	X	Nombre del profesor	DANIEL

TABLA: ***rec_sem***
Esta tabla contiene información respecto a determinadas asignaturas que deben de cumplir alguna regla en especial para poder inscribirlas (actualmente esta tabla ya no es vigente)

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_asign	char(4)	X	**Llave primaria:** clave de la asignatura	3021
a_ingr	char(4)		****	
regla	char(2)		****	

TABLA: ***requisito***
Catálogo que contiene los requisitos para cursar una asignatura específica.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_req	char(4)	X	**Llave primaria:** clave del requisito	0002
requisito	char(25)	X	Descripción del requisito	SEGUIMIENTO

TABLA: ***salón***
Catálogo que contiene la descripción de los salones

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_salon	char(3)	X	**Llave primaria:** clave del salón	123
salón	char(35)	X	Nombre del salón	SAL 123
cap_salon	Smallint		Capacidad del salón	null

TABLA: ***seriac_mod***
Esta tabla contiene información de aquellos alumnos de la carrera de Letras Modernas que pueden cursar determinadas asignaturas seriadas.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
num_cta	char(9)	X	**Llave primaria:** número de cuenta del alumno	074070662
cve_plan	char(4)	X	Llave primaria, clave del plan del alumno	0312
tpo_sub	char(1)	X	Nivel de avance que tiene el alumno	8

TABLA: ***sesión***
Esta tabla es una bitácora que contiene la información de las sesiones que cada alumno ha realizado en el sistema, en cada uno de los procesos.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
num_cta	Char(9)	X	**Llave primaria**: número de cuenta del alumno	402115340
consec	integer	X	**Llave primaria**: número de veces que el alumno ha ingresado al sistema (valor consecutivo)	14
Id_sesion	Char varying		Identificador único de cada sesión, es la combinación de varios datos	1e191c505bbc34e041 f8a4566f7f83f4
Cve_plan	Char(4)		Clave del plan del alumno	0315
Hr_inicio	abstime		Hora en que entró al sistema el alumno	2004-02-24 15:05:06
Hr_termino	abstime		Hora en que salió del sistema el alumno	2004-02-24 16:43
causa_termino	Char(3)		Motivo por el cual el alumno terminó su sesión	FIN
Ip	Inet		Número de la IP donde el alumno realizó sus movimientos	200.125.124.56

TABLA: ***sin_inscr***
Catálogo que contiene los números de cuenta de los alumnos a los cuales no les procedió ninguna asignatura de su inscripción, y que por lo tanto no tienen derecho a comprobante de inscripción definitivo.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
num_cta	Char(9)	X	**Llave primaria:** número de cuenta del alumno	402115340
cve_plan	char(4)	X	Llave primaria: clave del plan del alumno	0315

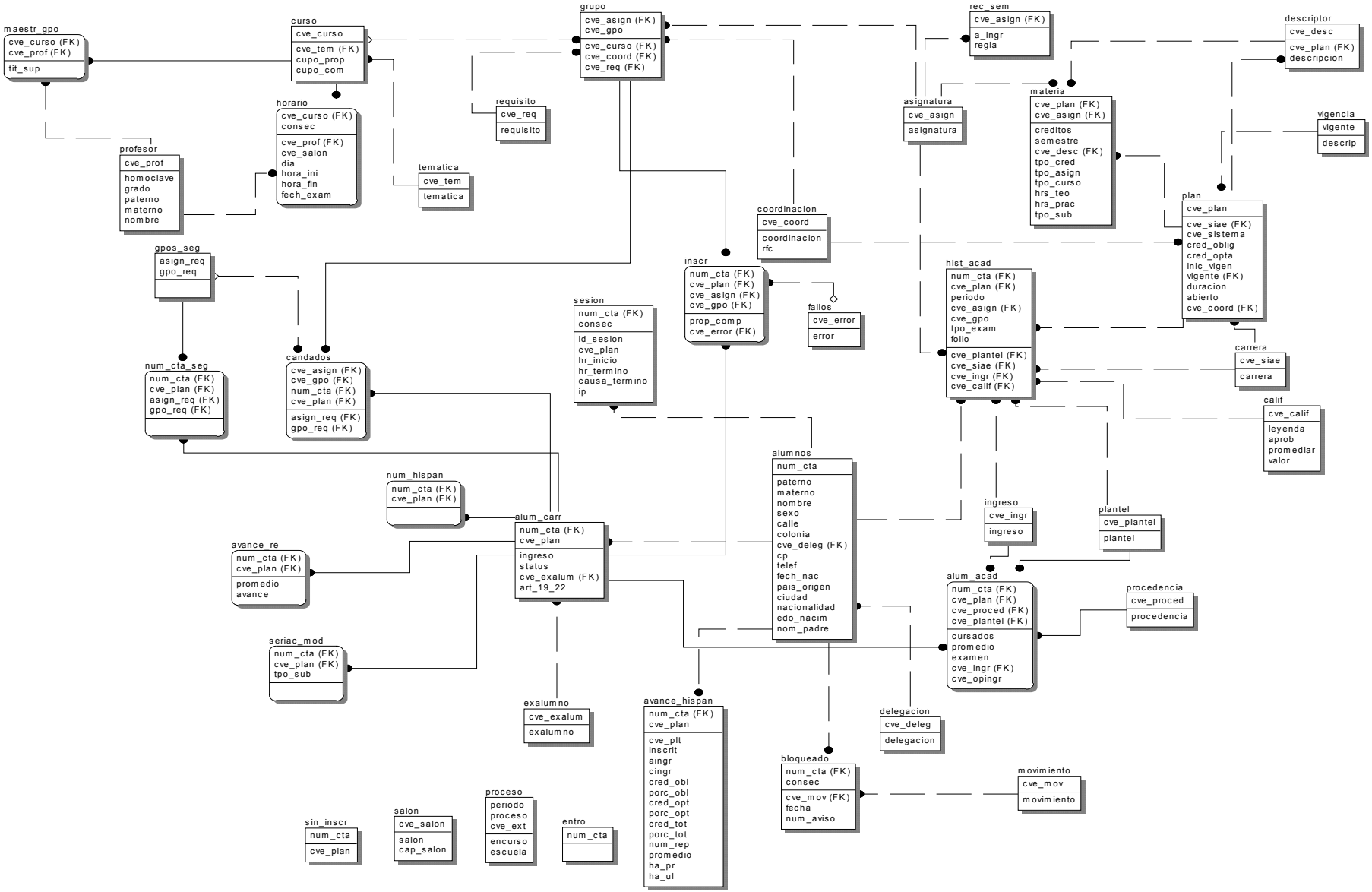
TABLA: ***temática***
Catálogo de temática de las materias de la carrera de Historia plan 1999.

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
cve_tem	char(3)	X	**Llave primaria**: clave de la temática	180
temática	char(40)	X	Descripción	REFORMA Y CONTRAREFORMA
Semestre	Char(1)		Contenido para semestre non (1) o par (2)	REFORMA Y CONTRAREFORMA 1 ò REFORMA Y CONTRAREFORMA 2

TABLA: ***vigencia***
Catálogo de claves de vigencia de los planes de estudio

Campo	Tipo	No nulo	Descripción	Ejemplos de instancias
Vigente	char(1)	X	**Llave primaria:** clave de la vigencia, puede ser: ‘0’ : NO VIGENTE ‘1’ : VIGENTE PARA PI ‘2’ : VIGENTE PARA REINS ‘3’ : EN CONSTRUCCIÓN ‘4’ : PI Y REINS	1
descrip	char(35)	X	Descripción de la clave	VIGENTE PARA PRIMER INGRESO

6.2 DIAGRAMA ENTIDAD RELACIÓN FÍSICO



6.3 SCRIPT PARA LA CREACIÓN DE LA BASE DE DATOS

```
CREATE TABLE alum_acad (
      num_cta            char(9) NOT NULL,
      cve_plan           char(4) NOT NULL,
      cve_proced          char(1),
      cursados           smallint,
      promedio            numeric(2,2),
      examen             numeric(2,2),
      cve_ingr           char(2),
      cve_opingr          char(1),
           PRIMARY KEY (num_cta, cve_plan),
           FOREIGN KEY (num_cta, cve_plan) REFERENCES alum_carr
);



CREATE TABLE alum_carr (
      num_cta            char(9) NOT NULL,
      cve_plan           char(4) NOT NULL,
      ingreso            char(4) NOT NULL,
      status             char(1) NOT NULL,
      cve_exalum          char(2)
      art_19_22          char(5) NOT NULL,
                  PRIMARY KEY (num_cta, cve_plan),
                  FOREIGN KEY (num_cta) REFERENCES alumnos,
                  FOREIGN KEY (cve_exalum) REFERENCES exalumno

);


CREATE TABLE alumnos (
      num_cta            char(9),
      paterno            char(25),
      materno            char(25),
      nombre             char(25),
      sexo               char(1),
      calle              char(38),
      colonia            char(38),
      cve_deleg          char(2),
      cp                 char(5),
      telef              char(10),
      fech_nac           char(8) NOT NULL,
```

```
    nom_padre          char(45),
    pais_origen        char(10),
    nacionalidad       char(1),
    ciudad             char(25),
    edo_nacim          char(2),

                PRIMARY KEY (num_cta),
                FOREIGN KEY (cve_deleg) REFERENCES delegacion
);



CREATE TABLE asignatura (
    cve_asign          char(4),
    asignatura         char(40) NOT NULL,
                PRIMARY KEY (cve_asign)
);



CREATE TABLE avance_hispan (
    num_cta            char(9) NOT NULL,
    cve_plan           char(4),
    cve_plt            char(3),
    inscrit            boolean,
    aingr              char(4),
    cingr              char(2),
    cred_obl           smallint,
    porc_obl           numeric(5,2),
    cred_opt           smallint,
    porc_opt           numeric(5,2),
    cred_tot           smallint,
    porc_tot           numeric(5,2),
    num_rep            smallint,
    promedio           numeric(4,2),
    ha_pr              char(5),
    ha_ul              char(5),
                PRIMARY KEY (num_cta, cve_plan),
                FOREIGN KEY (num_cta) REFERENCES alumnos
);


CREATE TABLE avance_re (
    num_cta            char(9),
    cve_plan           char(4),
```

```
    promedio        smallint,
    avance          smallint,
                PRIMARY KEY (num_cta, cve_plan)
                FOREIGN KEY (num_cta, cve_plan) REFERENCES alum_carr
);


CREATE TABLE bloqueado (
    num_cta         char(9) NOT NULL,
    consec          smallint,
    cve_mov         char(2) NOT NULL,
    fecha           char(8) NOT NULL,
    num_aviso       char(1),
                PRIMARY KEY (num_cta, consec),
                FOREIGN KEY (num_cta) REFERENCES alumnos,
                FOREIGN KEY (num_cta, cve_mov) REFERENCES movimientos

);


CREATE TABLE calif (
    cve_calif       char(2),
    leyenda         char(20),
    aprob           boolean NOT NULL,
    promediar       boolean NOT NULL,
    valor           smallint,
                PRIMARY KEY (cve_calif)
);


CREATE TABLE candados (
    cve_asign       char(4),
    cve_gpo         char(4),
    num_cta         char(9),
    cve_plan        char(9),
    asign_req       char(4),
    gpo_req         char(4),
                PRIMARY KEY (cve_asign, cve_gpo, num_cta,cve_plan),
                FOREIGN KEY (cve_asign, cve_gpo) REFERENCES grupo,
                FOREIGN KEY (num_cta, cve_plan) REFERENCES alum_carr,
                FOREIGN KEY (asign_req, gpo_req) REFERENCES gpos_seg
);


CREATE TABLE carrera (
```

```
    cve_siae        char(3),
    carrera         char(45) NOT NULL,
                    PRIMARY KEY (cve_siae)
);


CREATE TABLE coordinacion (
    cve_coord       char(2),
    coordinacion    char(45) NOT NULL,
    rfc             varchar(20),
                    PRIMARY KEY (cve_coord)
);


CREATE TABLE curso (
    cve_curso       smallint,
    cve_tem         char(3) NOT NULL,
    cupo_prop       smallint DEFAULT 0,
    cupo_com        smallint DEFAULT 0,
                    PRIMARY KEY (cve_curso),
                    FOREIGN KEY (cve_tem) REFERENCES tematica
);
```

```
CREATE TABLE delegacion (
    cve_deleg       char(2),
    delegacion      char(38) NOT NULL,
                    PRIMARY KEY (cve_deleg)
);


CREATE TABLE descriptor (
    cve_desc        char(2),
    cve_plan        char(4) NOT NULL,
    descripcion     char(40),
                    PRIMARY KEY (cve_desc),
                    FOREIGN KEY (cve_plan) REFERENCES plan
);


CREATE TABLE entro (
    num_cta         char(9),
                    PRIMARY KEY (num_cta)
);
```

```
CREATE TABLE exalumno (
      cve_exalum          char(2),
      exalumno            char(80) NOT NULL,
                  PRIMARY KEY (cve_exalum)
);


CREATE TABLE fallos (
      cve_error           char(5),
      error               varchar,
                  PRIMARY KEY (cve_error)
);


CREATE TABLE gpos_seg (
      asign_req           char(4),
      gpo_req             char(4),
                  PRIMARY KEY (asign_req, gpo_req)
);


CREATE TABLE grupo (
      cve_asign           char(4),
      cve_gpo             char(4),
      cve_curso           smallint,
      cve_coord           char(2) NOT NULL,
      cve_req             char(4),
                  PRIMARY KEY (cve_asign, cve_gpo),
                  FOREIGN KEY (cve_asign) REFERENCES asignatura,
                  FOREIGN KEY (cve_curso) REFERENCES curso,
                  FOREIGN KEY (cve_coord) REFERENCES coordinacion,
                  FOREIGN KEY (cve_req) REFERENCES requisito
);


CREATE TABLE hist_acad (
      num_cta             char(9) NOT NULL,
      cve_plantel         char(3) NOT NULL,
      cve_siae            char(3) NOT NULL,
      cve_plan            char(4) NOT NULL,
      cve_ingr            char(2) NOT NULL,
      cve_asign           char(4) NOT NULL,
      periodo             char(5),
      cve_calif           char(2) NOT NULL,
```

```
    cve_gpo             char(4),
    folio               char(7),
    tpo_exam            char(1) NOT NULL,
                PRIMARY KEY (num_cta, cve_plan, periodo,
                    cve_asign, cve_gpo, tpo_exam, folio)
);

CREATE INDEX folios ON hist_acad
(
    folio
);

CREATE INDEX mat_x_periodo ON hist_acad
(
    periodo,
    cve_asign,
    cve_gpo,
    tpo_exam
);

CREATE INDEX alum_plan ON hist_acad
(
    cve_plan,
    num_cta,
    cve_ingr
);


CREATE TABLE horario (
    cve_curso           smallint,
    consec              smallint,
    cve_prof            char(10) NOT NULL,
    cve_salon           char(4),
    dia                 char(1),
    hora_ini            char(2),
    hora_fin            char(2),
    fech_exam           char(8)
                PRIMARY KEY (cve_curso, consec),
                FOREIGN KEY (cve_curso) REFERENCES curso,
                FOREIGN KEY (cve_prof) REFERENCES profesor
);


CREATE TABLE ingreso (
```

```
        cve_ingr        char(2),
        ingreso         char(60),
                PRIMARY KEY (cve_ingr)
);


CREATE TABLE inscr (
        num_cta         char(9),
        cve_plan        char(4),
        cve_asign       char(4),
        cve_gpo         char(4),
        prop_comp       char(1) NOT NULL DEFAULT 'P',
        cve_error       char(5),
                PRIMARY KEY (num_cta, cve_plan, cve_asign,cve_gpo),
                FOREIGN KEY (num_cta, cve_plan) REFERENCES alum_carr,
                FOREIGN KEY (cve_asign, cve_gpo) REFERENCES grupo,
                FOREIGN KEY (cve_error) REFERENCES fallos
);


CREATE TABLE maestr_gpo (
        cve_curso       smallint,
        cve_prof        char(10) NOT NULL,
        tit_sup         char(1)
                PRIMARY KEY (cve_curso, cve_prof),
                FOREIGN KEY (cve_curso) REFERENCES curso,
                FOREIGN KEY (cve_prof) REFERENCES profesor
);


CREATE TABLE materia (
        cve_plan        char(4) NOT NULL,
        cve_asign       char(4) NOT NULL,
        creditos        smallint NOT NULL,
        semestre        smallint NOT NULL,
        cve_desc        char(2) NOT NULL,
        tpo_cred        char(1) NOT NULL,
        tpo_asign       char(2) NOT NULL,
        tpo_curso       char(1) NOT NULL,
        hrs_teo         smallint,
        hrs_prac        smallint,
        tpo_sub         char(1),
                PRIMARY KEY (cve_plan, cve_asign),
                FOREIGN KEY (cve_plan) REFERENCES plan,
                FOREIGN KEY (cve_asign) REFERENCES asignatura,
```

```
                FOREIGN KEY (cve_desc) REFERENCES descriptor
);


CREATE TABLE movimiento (
    cve_mov          char(2),
    movimiento       char(40) NOT NULL,
            PRIMARY KEY (cve_mov)
);

CREATE TABLE num_cta_seg (
    num_cta          char(9),
    cve_plan         char(4),
    asign_req        char(4),
    gpo_req          char(4),
            PRIMARY KEY (num_cta, cve_plan, asign_req,gpo_req),
            FOREIGN KEY (num_cta, cve_plan) REFERENCES alum_carr,
            FOREIGN KEY (asign_req, gpo_req)REFERENCES gpos_seg
);


CREATE TABLE num_hispan (
    num_cta          char(9),
    cve_plan         char(4),
            PRIMARY KEY (num_cta, cve_plan),
            FOREIGN KEY (num_cta, cve_plan) REFERENCES alum_carr
);



CREATE TABLE plan (
    cve_plan         char(4),
    cve_siae         char(3) NOT NULL,
    cve_sistema      char(1) NOT NULL,
    cred_oblig       smallint NOT NULL,
    cred_opta        smallint NOT NULL,
    inic_vigen       char(4),
    vigente          char(1) NOT NULL,
    duracion         smallint NOT NULL,
    abierto          char(1) NOT NULL,
    cve_coord        char(2) NOT NULL,
            PRIMARY KEY (cve_plan),
            FOREIGN KEY (cve_siae) REFERENCES carrera,
            FOREIGN KEY (vigente) REFERENCES vigencia,
            FOREIGN KEY (cve_coord) REFERENCES coordinacion
```

```
);


CREATE TABLE plantel (
      cve_plantel         char(3),
      plantel             char(42) NOT NULL,
                  PRIMARY KEY (cve_plantel)
);


CREATE TABLE procedencia (
      cve_proced          integer NOT NULL,
      procedencia         varchar(20),
                  PRIMARY KEY (cve_proced)
);


CREATE TABLE proceso (
      periodo             char(5),
      proceso             char(1),
      cve_ext             char(1) DEFAULT 'U',
      encurso             boolean NOT NULL,
      escuela             char(3) NOT NULL DEFAULT '010',
                  PRIMARY KEY (periodo, proceso, cve_ext)
);


CREATE TABLE profesor (
      cve_prof            char(10),
      homoclave           char(3),
      grado               char(6),
      paterno             char(17),
      materno             char(17),
      nombre              char(17) NOT NULL
                  PRIMARY KEY (cve_prof)
);


CREATE TABLE rec_sem (
      cve_asign           char(4) NOT NULL,
      a_ingr              char(4),
      regla               char(2),
                  PRIMARY KEY (cve_asign),
                  FOREIGN KEY (cve_asign) REFERENCES asignatura
);
```

```
CREATE TABLE requisito (
	cve_req		char(4),
	requisito	char(25) NOT NULL,
			PRIMARY KEY (cve_req)
);


CREATE TABLE salon (
	cve_salon	char(3) NOT NULL,
	salon		char(35),
	cap_salon	smallint,
			PRIMARY KEY (cve_salon)
);


CREATE TABLE seriac_mod (
	num_cta		char(9) NOT NULL,
	cve_plan	char(4) NOT NULL,
	tpo_sub		char(1) NOT NULL,
			PRIMARY KEY (num_cta, cve_plan, tpo_sub)
);
```

```
CREATE TABLE sesion (
	num_cta		char(9) NOT NULL,
	consec		int4,
	id_sesion	varchar unique,
	cve_plan	char(4),
	hr_inicio	abstime,
	hr_termino	abstime,
	causa_termino	char(3),
	ip		inet,
			PRIMARY KEY (num_cta, consec),
			FOREIGN KEY (num_cta) REFERENCES alumnos
);


CREATE TABLE sin_inscr (
	num_cta		char (9),
	cve_plan	char(4),
			PRIMARY KEY (num_cta)
);
```

```
CREATE TABLE tematica (
      cve_tem            char(3),
      tematica           char(40) NOT NULL,
                  PRIMARY KEY (cve_tem)
);


CREATE TABLE vigencia (
      vigente            char(1),
      descrip            char(35) NOT NULL,
                  PRIMARY KEY (vigente)
);

ALTER TABLE alum_acad
      ADD CONSTRAINT FOREIGN KEY (cve_proced)
                  REFERENCES procedencia;
```

7. FORMA DE PREPARAR EL SISTEMA PARA CADA PROCESO DE REGISTRO

Una vez que se tiene la información de horarios, en la tabla correspondiente "hor_or>semestre>" , así como los requerimientos de cada coordinación con respecto al orden de atención, seguimientos y candados, se procede a actualizar las tablas correspondientes.

7.1 PERIODO DE REINSCRIPCIÓN

1. Hacer un respaldo de todas las tablas de la base de datos que contengan número de cuenta. Esto se hace ejecutando desde Postgres el archivo crea_temp.txt
2. Borrar el contenido de las siguientes tablas, para poder actualizar el directorio de los alumnos y las historias académicas.
    - alum_acad
    - alum_carr
    - alumnos
    - avance_re
    - inscr
    - num_hispan
    - seriac_mod
    - sin_inscr
    - candados
    - hist_acad
    - boqueado
    - sesión
    - avance_hispan

3. Hacerle un dump a la tabla alumnos del servidor *Prometeo* y vaciarla en el servidor *Sócrates* (esto es con la finalidad de tener en ambos servidores la misma información ya que en Prometeo se están actualizando constantemente los datos).
4. Ejecutar el programa **carga_directorio.pl** para hacer el llenado de las tablas "alum_carr" y "alum_acad".
5. Ejecutar el programa **carga_hist_acad.pl** para llenar la tabla "hist_acad".
6. Se borra el contenido de las tablas:
    - horario
    - maestr_gpo
    - grupo
    - curso
    - profesor

7. Se hace un dump de las mismas tablas en el servidor *bdserv* y se vacían en el servidor *Sócrates*, en el siguiente orden:

- profesor
- curso
- grupo
- maestr_gpo
- horario

8. Se ejecuta el programa **horarios_ord_html.pl** para generar los archivos html que se van a subir a la página web.
9. Se vuelven a llenar las tablas que estaban relacionadas con "alumnos", "alum_carr" y "alum_acad", excepto la tabla "inscr" y "sesion" esto se hace con las tablas de respaldo (Ej. insert into avance_re select * from w20041_avance_re).
10. Se ejecuta el programa **carga_limpia.pl** para llenar la tabla "avance_re" (es decir los alumnos que ya tienen determinado avance de créditos y promedio)
11. Se ejecuta el programa **carga_seguimientos.pl** para llenar las tablas "gpos_seg", "candados", y "cve_req" en la tabla grupo. **(este punto sólo aplica cuando el periodo de registro es par, para los grupos de Pedagogía, Geografía y Letras Hispánicas, que son los que manejan seguimiento)**
12. Se hace un update a la tabla "grupo" para marcar la cve_req de los grupos de Griego y Latín del colegio de Letras Clásicas.

Update grupo set cve_req='0040' where cve_asign = '3600' ;
Update grupo set cve_req='0040' where cve_asign = '3601' ;
Update grupo set cve_req='0040' where cve_asign = '3800' ;
Update grupo set cve_req='0040' where cve_asign = '3801' ;

13. Se carga el listado de alumnos que ya aprobaron el Taller de Redacción de la carrera de Letras Hispánicas; actualizando la tabla "num_hispan"
14. Se ejecuta el programa **sube_tpo.pl** para actualizar (marcar) la tabla "materia" donde las asignaturas son seriadas.
15. Se ejecuta el programa **carga_limpia.pl** para cargar en la tabla "bloqueados" los deudores de libros, deudores de certificados de bachillerato, deudores de documentos personales etc.
16. Se borran de la tabla "hist_acad" todos los alumnos que cuyo art_19_22 es mayor o igual al periodo en curso (con la finalidad de disminuir el tamaño de la tabla)
17. Se ejecuta un vacuumdb a la base de datos.

7.2 PERIODO DE ALTAS BAJAS Y CAMBIOS

1. Generar el reporte de los cambios y bajas de grupo (si es que los hubo)
2. Se llena la tabla entro, con todos los números de cuenta de aquellos alumnos que inscribieron mínimo una asignatura.
3. Generar tablas temporales de :
    - inscr
    - candados
    - curso
    - grupo
    - horario
    - maestr_gpo
    - profesor
4. Se hace un dump de las últimas cuatro tablas de la lista anterior del servidor *bdserv* y se vacían en el servidor *Sócrates* en el siguiente orden:

    - profesor
    - curso
    - grupo
    - maestr_gpo
    - horario

5. Se vuelve a llenar la tabla "inscr." y "candados", con las tablas temporales y 'estas se borran de la base de datos.
6. Se quita la seriación de los grupos de Pedagogía, Geografía y Letras Hispánicas, para esto se modifica la tabla "grupo". **(esto solo se hace en el periodo de registro par)**

Update grupo set cve_req=null where cve_req = '0002';

Estos puntos son posteriores al proceso de registro.

7. Se ejecuta el programa **archivos_salida.pl** para generar los archivos de inscripción que se envían a DGAE.
8. No se envían a la DGAE las inscripciones de las asignaturas (5000).
9. Se ejecuta el programa inscr_proced_2.pl para cargar en la tabla "Instr." los ajustes de inscripción que no procedió (del diagnóstico que manda DGAE).
10. Se ejecuta el programa sin_inscripcion.pl y con el archivo que genera se llena la tabla "sin_inscr", la cual sirve para no imprimir los comprobantes de aquellos alumnos a los que no les procedió ninguna asignatura.
11. Se ejecuta el programa imprime.pl para generar los comprobantes definitivos de inscripción.

7.3 PERIODOS DE EXTRAORDINARIOS

Para cada semestre se programan dos periodos de exámenes extraordinarios, el periodo "EA" que es al inicio del semestre y el periodo "EB" que coincide con el fin de semestre.

1. Se repiten los pasos del 1 al 5 del proceso de reinscripción.
2. Al contenido de las tablas "curso", "grupo", "maestr_gpo" y "horario" se le agregan los horarios del periodo ya sea EA o EB
3. Se ejecuta el programa horarios_extra_html.pl para generar los horarios en formato html que se van a subir a la página web
4. Se vuelven a llenar las tablas de las que se hicieron respaldos (las que están relacionadas con "alumnos","alum_carr" y "alum_acad").

   Ej. Insert into bloqueados select * from w20041_bloqueados;
5. Se ejecuta el programa carga_limpia.pl para llenar la tabla "avance_re" (es decir los alumnos que ya tienen determinado avance de créditos y promedio)
6. En este periodo la tabla "hist_acad" se queda con todos los registros, ya que es importante tener a los alumnos que son art_19_22 .
7. Se ejecuta el programa archivos_salida.pl para generar los archivos de inscripción que se envían a DGAE.
8. Se manda un listado con los alumnos que son causa 20 u 11 y que se inscribieron para que la DGAE los reactive y pueda proceder su inscripción.

CONCLUSIONES Y RECOMENDACIONES

El sistema SIFYL se ha utilizado en 17 semestres, desde el semestre 2002-2 hasta 2010-2, en donde semestre a semestre se han realizado cambios, desde modificaciones a la base de datos, como la forma de programar y la interfaz con el usuario.

Por ejemplo, para realizar cambio de grupo; conservar la misma materia y sólo cambiar el grupo, en la version inicial se tenía que:

- Seleccionar la opción de BAJA y digitar la clave de asignatura grupo a dar de baja.
- Seleccionar la opción de ALTA y digitar la clave de asignatura y grupo, con el grupo a dar de alta.

Después de varios cambios la version del semestre 2010-1, funciona de la siguiente manera:

- Seleccionar la opción CAMBIOS.
- El sistema muestra un combo con las asignaturas que tiene registradas. Una vez seleccionada la materia en donde se desea el cambio, el sistema muestra un combo con los grupos en los cuales existe cupo.

Existen 3 versiones más: “Interno”, “Ventanillas” y “CoordinacionHist”, dichas versiones están diseñadas para uso exclusivo del departamento de Sistemas, las ventanillas de la Secretaría y la Coordinación de Historia respectivamente.

Interno: Esta versión permite inscribir a los alumnos de todos los colegios, sin validar el NIP del alumno, las fechas de inscripción por carrera y generación y sin tope en el número de intentos.

Ventanillas: Versión que permite que en Ventanillas de la Secretaría, solo ingresando el numero de cuenta, se pueda consultar y/o imprimir los “comprobantes de inscripción”.

CoordinacionHist: Los alumnos del Colegio de Historia, son atendidos en su Coordinación y posteriormente la misma Coordinación realiza el registro.

ANEXOS
1. IPCHAINS

```
# Firewall configuration written by lokkit
# Manual customization of this file is not recommended.
# Note: ifup-post will punch the current nameservers through the
#       firewall; such entries will *not* be listed here.

:input DENY
:forward DENY
:output DENY
-A input -s 132.248.123.39 -d 132.248.123.69 5432 -p tcp -j ACCEPT
-A input -s 132.248.123.58 -d 132.248.123.69 5432 -p tcp -j ACCEPT
-A input -s 132.248.123.49 -d 132.248.123.69 5432 -p tcp -j ACCEPT
-A input -s 132.248.123.39 -d 132.248.123.69 22 -p tcp -j ACCEPT
-A input -s 132.248.123.49 -d 132.248.123.69 22 -p tcp -j ACCEPT
-A input -s 132.248.123.39 22 -d 132.248.123.69 -p tcp -j ACCEPT
-A input -s 132.248.123.49 22 -d 132.248.123.69 -p tcp -j ACCEPT
-A input -s 132.248.123.38 -d 132.248.123.69 22 -p tcp -j ACCEPT
-A input -s 132.248.123.38 22 -d 132.248.123.69 -p tcp -j ACCEPT
-A input -s 0/0 -d 132.248.123.69 -l -j DENY
-A input -s 0/0 -d 0/0 -i lo -j ACCEPT
-A output -s 132.248.123.69 -d 132.248.123.39 -j ACCEPT
-A output -s 132.248.123.69 -d 132.248.123.58 -j ACCEPT
-A output -s 132.248.123.69 -d 132.248.123.38 -j ACCEPT
-A output -s 132.248.123.69 -d 132.248.123.49 -j ACCEPT
-A output -s 132.248.123.69 -d 0/0 -l -j DENY

#
#
#
#:input ACCEPT
#:forward ACCEPT
#:output ACCEPT
#-A input -s 0/0 -d 0/0 -i lo -j ACCEPT
#-A input -s 132.248.204.1 53 -d 0/0 -p udp -j ACCEPT
#-A input -s 132.248.10.2 53 -d 0/0 -p udp -j ACCEPT
#-A input -s 0/0 -d 0/0 -p tcp -y -j REJECT
#-A input -s 0/0 -d 0/0 -p udp -j REJECT
```

2. POSTGRESQL

```
#! /bin/sh
# postgresql    This is the init script for starting up the PostgreSQL
#               server
#
# chkconfig: - 85 15
# description: Starts and stops the PostgreSQL backend daemon that handles \
#              all database requests.
# processname: postmaster
# pidfile: /var/run/postmaster.pid

# Version 6.5.3-2 Lamar Owen
# Added code to determine if PGDATA exists, whether it is current version
#     or not, and initdb if no PGDATA (initdb will not overwrite a database).

# Version 7.0 Lamar Owen
# Added logging code
# Changed PGDATA.
#
# Version 7.0.2 Trond Eivind Glomsrød <teg@redhat.com>
# use functions, add conditional restart

# Version 7.0.3 Lamar Owen <lamar@postgresql.org>
# Check for the existence of functions before blindly using them
# in particular -- check for success () and failure () before using.
# More Cross-distribution support -- PGVERSION variable, and docdir checks.

# Version 7.1 Release Candidate Lamar Owen <lamar@postgresql.org>
# initdb parameters have changed.

# Version 7.1.2 Trond Eivind Glomsrød <teg@redhat.com>
# Specify shell for su
# Handle stop better - kill unwanted output, make it wait until the database is ready
# Handle locales slightly differently - always using "C" isn't a valid option
# Kill output from database initialization
# Mark messages for translation

# Version 7.1.2-2.PGDG Lamar Owen <lamar.owen@wgcr.org>
# sync up.
# Karl's fixes for some quoting issues.
```

```
# Version 7.2b2 Lamar Owen <lamar.owen@wgcr.org>
# version change.

# Version 7.2 final.  Lamar Owen <lamar.owen@wgcr.org>
# reload from Peter E.
# Eliminate the pidof postmaster test in stop -- we're using pg_ctl so we don't need pidof.
# Tested the $? return for the stop script -- it does in fact propagate.
# TODO: multiple postmasters.

# PGVERSION is:
PGVERSION=7.2

# Source function library.
INITD=/etc/rc.d/init.d
. $INITD/functions

# Get function listing for cross-distribution logic.
TYPESET=`typeset -f|grep "declare"`

# Get config.
. /etc/sysconfig/network

# Check that networking is up.
# Pretty much need it for postmaster.
[ "${NETWORKING}" = "no" ] && exit 0

[ -f /usr/bin/postmaster ] || exit 0


start(){
        PSQL_START=$"Starting postgresql service: "

        # Check for older PGDATA location.
        if [ -f /var/lib/pgsql/PG_VERSION ] && [ -d /var/lib/pgsql/base/template1 ]
        then
                export PGDATA=/var/lib/pgsql
        else
                export PGDATA=/var/lib/pgsql/data
        fi

        # Check for the PGDATA structure
        if [ -f $PGDATA/PG_VERSION ] && [ -d $PGDATA/base ]
        then
        # Check version of existing PGDATA
```

```
                if [ `cat $PGDATA/PG_VERSION` != '7.2' ]
                then
                        SYSDOCDIR="(Your System's documentation directory)"
                        if [ -d /usr/doc/postgresql-$PGVERSION ]
                        then
                                SYSDOCDIR=/usr/doc
                        fi
                        if [ -d /usr/share/doc/postgresql-$PGVERSION ]
                        then
                                SYSDOCDIR=/usr/share/doc
                        fi
                        if [ -d /usr/doc/packages/postgresql-$PGVERSION ]
                        then
                                SYSDOCDIR=/usr/doc/packages
                        fi
                        if [ -d /usr/share/doc/packages/postgresql-$PGVERSION ]
                        then
                                SYSDOCDIR=/usr/share/doc/packages
                        fi
                        echo
                        echo -e $"An old version of the database format was found.\nYou need to upgrade the data format before using PostgreSQL.\nSee $SYSDOCDIR/postgresql-$PGVERSION/README.rpm-dist for more information."
                        exit 1
#                This doesn't seem to do anything useful...
#               else
#                       if echo "$TYPESET"|grep "declare -f success ()" >/dev/null
#                       then
#                               success "$PSQL_CHECK"
#                       else
#                               echo "  [ OK ]"
#                       fi
#                       echo
                fi

        # No existing PGDATA! Initdb it.

        else
              echo -n $"Initializing database: "
           if [ ! -d $PGDATA ]
                then
                        mkdir -p $PGDATA
                        chown postgres.postgres $PGDATA
                fi
```

```
                # Make sure the locale from the initdb is preserved for later startups...
                [ -f /etc/sysconfig/i18n ] && cp /etc/sysconfig/i18n
$PGDATA/../initdb.i18n
                # Just in case no locale was set, use en_US
                [ ! -f /etc/sysconfig/i18n ] && echo "LANG=en_US" >
$PGDATA/../initdb.i18n
                # Is expanded this early to be used in the command su runs
                echo "export LANG LC_ALL LC_CTYPE LC_COLLATE LC_NUMERIC
LC_CTYPE LC_TIME" >> $PGDATA/../initdb.i18n
                # Initialize the database
                su -l postgres -s /bin/sh -c "/usr/bin/initdb --pgdata=/var/lib/pgsql/data >
/dev/null 2>&1" < /dev/null
                [ -f $PGDATA/PG_VERSION ] && echo_success
                [ ! -f $PGDATA/PG_VERSION ] && echo_failure
                echo
        fi

        # Check for postmaster already running...
        pid=`pidof -s postmaster`
        if [ $pid ]
        then
                echo $"Postmaster already running."
        else
                #all systems go -- remove any stale lock files
                rm -f /tmp/.s.PGSQL.* > /dev/null
                echo -n "$PSQL_START"
                su -l postgres -s /bin/sh -c "/usr/bin/pg_ctl  -D $PGDATA -o -i -p
/usr/bin/postmaster start  > /dev/null 2>&1" < /dev/null
                sleep 1
                pid=`pidof -s postmaster`
                if [ $pid ]
                then
                        if echo "$TYPESET"|grep "declare -f success ()" >/dev/null
                        then
                                success "$PSQL_START"
                        else
                                echo "  [ OK ]"
                        fi
                        touch /var/lock/subsys/postgresql
                        echo $pid > /var/run/postmaster.pid
                        echo
                else
                        if echo "$TYPESET"|grep "declare -f failure ()" >/dev/null
                        then
                                failure "$PSQL_START"
```

```
                        else
                                echo " [ FAILED ]"
                        fi
                        echo
                fi
        fi
}

stop(){
        echo -n $"Stopping postgresql service: "
        # Check for older PGDATA location.
        if [ -f /var/lib/pgsql/PG_VERSION ] && [ -d /var/lib/pgsql/base/template1 ]
        then
                export PGDATA=/var/lib/pgsql
        else
                export PGDATA=/var/lib/pgsql/data
        fi
        su -l postgres -s /bin/sh -c "/usr/bin/pg_ctl stop -D $PGDATA -s -m fast" >
/dev/null 2>&1
      ret=$?
      if [ $ret -eq 0 ]; then
           echo_success
      else
         echo  failure
      fi
         echo
         rm -f /var/run/postmaster.pid
         rm -f /var/lock/subsys/postgresql
}

restart(){
   stop
   start
}

condrestart(){
   [ -e /var/lock/subsys/postgresql ] && restart || :
}

reload(){
   su -l postgres -s /bin/sh -c "/usr/bin/pg_ctl reload -D $PGDATA -s" > /dev/null 2>&1
}

# This script is slightly unusual in that the name of the daemon (postmaster)
# is not the same as the name of the subsystem (postgresql)
```

```
# See how we were called.
case "$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;
  status)
        status postmaster
        ;;
  restart)
        restart
        ;;
  condrestart)
        condrestart
        ;;
  reload|force-reload)
        reload
        ;;
  *)
        echo $"Usage: $0 {start|stop|status|restart|condrestart|reload|force-reload}"
        exit 1
esac

exit 0
```

3. PG_HBA.CONF

```
#
#                   PostgreSQL HOST-BASED ACCESS (HBA) CONTROL FILE
#
#
# This file controls:
#         o which hosts are allowed to connect
#         o how users are authenticated on each host
#         o databases accessible by each host
#
# It is read on postmaster startup and when the postmaster receives a SIGHUP.
# If you edit the file on a running system, you have to SIGHUP the postmaster
# for the changes to take effect.
#
# Each line is a new record. Records cannot be continued across multiple
# lines. Comments begin with # and continue to the end of the line.
# Blank lines are ignored. A record consists of tokens separated by
# multiple spaces or tabs.
#
# Each record specifies the authentication method to be used for connections
# of a certain type that match a certain set of IP addresses (if relevant
# for the connection type) and a certain database or databases.  The
# postmaster finds the first record that matches the connection type,
# client address, and database name, and uses that record to perform client
# authentication.  If no record matches, the connection is rejected.
#
# The first token of a record indicates its type. The remainder of the
# record is interpreted based on its type.
#
# Record Types
# ============
#
# There are three types of records:
#         o host
#         o hostssl
#         o local
#
# host
# ----
#
# This record identifies networked hosts that are permitted to connect
# via IP connections.
#
```

```
# Format:
#
#         host      DBNAME      IP_ADDRESS      ADDRESS_MASK      AUTH_TYPE
[AUTH_ARGUMENT]
#
# DBNAME can be:
#         o the name of a PostgreSQL database
#         o "all" to indicate all databases
#         o "sameuser" to allow access only to databases with the same
#           name as the connecting user
#
# The superuser needs access to the 'template1' database because it is used
# by a variety of PostgreSQL utility commands.
#
# IP_ADDRESS and ADDRESS_MASK are standard dotted decimal IP address and
# mask values. IP addresses can only be specified numerically, not as
# domain or host names.
#
# AUTH_TYPE and AUTH_ARGUMENT are described below.
#
#
# hostssl
# -------
#
# The format of this record is identical to "host".
#
# This record identifies a set of network hosts that are permitted to
# connect to databases over secure SSL IP connections. Note that a "host"
# record will also allow SSL connections.  "hostssl" matches *only*
# SSL-secured connections.
#
# This keyword is only available if the server was compiled with SSL
# support enabled.
#
#
# local
# -----
#
# This record identifies the authentication to use when connecting to
# the server via a local UNIX domain socket.  UNIX-socket connections are
# allowed only if this record type appears.
#
# Format:
#   local  DBNAME  AUTH_TYPE  [AUTH_ARGUMENT]
#
```

```
# This format is identical to the "host" record type except the IP_ADDRESS
# and ADDRESS_MASK fields are omitted.
#
#
#
# Authentication Types (AUTH_TYPE)
# ================================
#
# AUTH_TYPE indicates the method used to authenticate users. The username
# is specified in the connection request.  A different AUTH_TYPE can be
# specified for each record in the file.
#
#   trust:        No authentication is done. Any valid username is accepted,
#                 including the PostgreSQL superuser. This option should
#                 be used only for hosts where all users are trusted.
#
#   password:     Authentication is done by matching a password supplied
#                 in clear by the host. If no AUTH_ARGUMENT is used, the
#                 password is compared with the user's entry in the
#                 pg_shadow table.
#
#                 If AUTH_ARGUMENT is specified, the username is looked up
#                 in that file in the $PGDATA directory. If the username
#                 is found but there is no password, the password is looked
#                 up in pg_shadow. If a password exists in the file, it is
#                 used instead. These secondary files allow fine-grained
#                 control over who can access which databases and whether
#                 a non-default password is required. The same file can be
#                 used in multiple records for easier administration.
#                 Password files can be maintained with the pg_passwd(1)
#                 utility. Remember, these passwords override pg_shadow
#                 passwords.
#
#   md5:          Same as "password", but the password is encrypted while
#                 being sent over the network. This method is preferable to
#                 "password" except for pre-7.2 clients that don't support it.
#                 NOTE: md5 can use usernames stored in secondary password
#                 files but ignores passwords stored there.  The pg_shadow
#                 password will always be used.
#
#   crypt:        Same as "md5", but uses crypt for pre-7.2 clients.  You can
#                 not store encrypted passwords in pg_shadow if you use this
#                 method.
#
#   ident:        For TCP/IP connections, authentication is done by contacting
```

```
#               the ident server on the client host.  Remember, this is
#               only as secure as the client machine.  On machines that
#               support unix-domain socket credentials (currently Linux,
#               FreeBSD, NetBSD, and BSD/OS), this method also works for
#               "local" connections.
#
#               AUTH_ARGUMENT is required: it determines how to map
#               remote user names to Postgres user names. The
#               AUTH_ARGUMENT is a map name found in the
#               $PGDATA/pg_ident.conf file. The connection is accepted
#               if that file contains an entry for this map name with
#               the ident-supplied username and the requested Postgres
#               username. The special map name "sameuser" indicates an
#               implied map (not in pg_ident.conf) that maps each ident
#               username to the identical PostgreSQL username.
#
#  krb4:        Kerberos V4 authentication is used.  Allowed only for
#               TCP/IP connections, not for local UNIX-domain sockets.
#
#  krb5:        Kerberos V5 authentication is used.  Allowed only for
#               TCP/IP connections, not for local UNIX-domain sockets.
#
#  pam:       Authentication is passed off to PAM (PostgreSQL must be
#           configured --with-pam), using the default service name
#           "postgresql" - you can specify your own service name, by
#           setting AUTH_ARGUMENT to the desired service name.
#
#   reject:     Reject the connection. This is used to reject certain hosts
#               that are part of a network specified later in the file.
#               To be effective, "reject" must appear before the later
#               entries.
#
#
#
# Examples
# ========
#
#
# Allow any user on the local system to connect to any database under any
# username using Unix-domain sockets (the default for local connections):
# TYPE          DATABASE      IP_ADDRESS      MASK                    AUTH_TYPE
AUTH_ARGUMENT
# local     all                               trust
#
# The same using local loopback IP connections:
```

```
# TYPE          DATABASE      IP_ADDRESS      MASK                    AUTH_TYPE
AUTH_ARGUMENT
# host      all         127.0.0.1     255.255.255.255    trust
#
# Allow any user from any host with IP address 192.168.93.x to
# connect to database "template1" as the same username that ident reports
# for the connection (typically his Unix username):
#
# TYPE          DATABASE      IP_ADDRESS      MASK                    AUTH_TYPE
AUTH_ARGUMENT
# host      template1   192.168.93.0  255.255.255.0      ident      sameuser
#
# Allow a user from host 192.168.12.10 to connect to database "template1"
# if the user's password in pg_shadow is correctly supplied:
#
# TYPE          DATABASE      IP_ADDRESS      MASK                    AUTH_TYPE
AUTH_ARGUMENT
# host      template1   192.168.12.10 255.255.255.255    md5
#
# In the absence of preceding "host" lines, these two lines will reject
# all connection from 192.168.54.1 (since that entry will be matched
# first), but allow Kerberos V5-validated connections from anywhere else
# on the Internet. The zero mask means that no bits of the host IP address
# are considered, so it matches any host:
#
#
# TYPE          DATABASE      IP_ADDRESS      MASK                    AUTH_TYPE
AUTH_ARGUMENT
# host      all       192.168.54.1   255.255.255.255    reject
# host      all       0.0.0.0       0.0.0.0          krb5
#
# Allow users from 192.168.x.x hosts to connect to any database if they
# pass the ident check. For example, if ident says the user is "james" and
# he requests to connect as PostgreSQL user "guest", the connection is
# allowed if there is an entry in $PGDATA/pg_ident.conf with map name
# "phoenix" that says "james" is allowed to connect as "guest":
#
# TYPE          DATABASE      IP_ADDRESS      MASK                    AUTH_TYPE
AUTH_ARGUMENT
# host      all       192.168.0.0    255.255.0.0       ident      phoenix
#
# If these are the only two lines for local connections, they will allow
# local users to connect only to their own databases (database named the
# same as the user name), except for administrators who may connect to
# all databases.  The file $PGDATA/admins lists the user names who are
```

```
# permitted to connect to all databases.  Passwords are required in all
# cases.  (If you prefer to use ident authorization, an ident map can
# serve a parallel purpose to the password list file used here.)
#
# TYPE          DATABASE      IP_ADDRESS      MASK                    AUTH_TYPE
AUTH_ARGUMENT
# local     sameuser                            md5
# local     all                                 md5  admins
#
# See $PGDATA/pg_ident.conf for more information on Ident maps.
#
#
#
# Put your actual configuration here
# ==================================
#
# This default configuration allows any local user to connect with any
# PostgreSQL username, over either UNIX domain sockets or IP.
#
# If you want to allow non-local connections, you will need to add more
# "host" records. Also, remember IP connections are only enabled if you
# start the postmaster with the -i option.
#
# CAUTION: if you are on a multiple-user machine, the default
# configuration is probably too liberal for you. Change it to use
# something other than "trust" authentication.
#
# TYPE        DATABASE      IP_ADDRESS      MASK                    AUTH_TYPE
AUTH_ARGUMENT

#local     all                                 trust
#host      all       127.0.0.1     255.255.255.255    trust

# Using sockets credentials for improved security. Not available everywhere,
# but works on Linux, *BSD (and probably some others)

#local  all       ident    sameuser

local       all                                 trust
host        all       127.0.0.1     255.255.255.255     trust
host        escbd      132.248.123.38     255.255.255.255     trust
host        all       132.248.123.39     255.255.255.255     trust
host        all       132.248.123.58     255.255.255.255     trust
host        escbd      132.248.123.41     255.255.255.255     trust
host        escolar     132.248.123.49     255.255.255.255     trust
```

4. PF.CONF

```
#definicion de la interfaz de red
IntRED="ep1"
#ip no validas
NoRouteIPs="{ 127.0.0.0/8, 192.168.0.0/16, 172.16.0.0/12, 10.0.0.0/8 }"
IpBloq="{ 200.66.191.1/32, 200.65.137.176/32, 148.243.87.64 }"
#servicios sobre los que se permite conexión a todos
Serv="{ www }"

scrub in all
block in quick on $IntRED from $IpBloq to any
block in quick on $IntRED from $NoRouteIPs to any
block out quick on $IntRED from any to $NoRouteIPs

# bloquear indeseables
block in quick on $IntRED from 66.119.34.0/24 to any

# deja pasar conexiones de la red interna
pass in quick on $IntRED from 132.248.123.39/32 to any
pass in quick on $IntRED from 132.248.123.40/32 to any
pass in quick on $IntRED from 132.248.123.49/32 to any
pass in quick on $IntRED from 132.248.123.69/32 to any

#dejar pasar al puerto web
pass in quick on $IntRED proto tcp from any to any port $Serv  flags S/SA keep state


block in quick on ep1 inet from any to any
pass out on ep1 inet from any to any
```

5. HTTPD.CONF

```
#
# Based upon the NCSA server configuration files originally by Rob McCool.
#
# This is the main Apache server configuration file.  It contains the
# configuration directives that give the server its instructions.
# See <URL:http://www.apache.org/docs/> for detailed information about
# the directives.
#
# Do NOT simply read the instructions in here without understanding
# what they do.  They're here only as hints or reminders.  If you are unsure
# consult the online docs. You have been warned.
#
# After this file is processed, the server will look for and process
# /var/www/conf/srm.conf and then /var/www/conf/access.conf
# unless you have overridden these with ResourceConfig and/or
# AccessConfig directives here.
#
# The configuration directives are grouped into three basic sections:
#  1. Directives that control the operation of the Apache server process as a
#     whole (the 'global environment').
#  2. Directives that define the parameters of the 'main' or 'default' server,
#     which responds to requests that aren't handled by a virtual host.
#     These directives also provide default values for the settings
#     of all virtual hosts.
#  3. Settings for virtual hosts, which allow Web requests to be sent to
#     different IP addresses or hostnames and have them handled by the
#     same Apache server process.
#
# Configuration and logfile names: If the filenames you specify for many
# of the server's control files begin with "/" (or "drive:/" for Win32), the
# server will use that explicit path.  If the filenames do *not* begin
# with "/", the value of ServerRoot is prepended -- so "logs/foo.log"
# with ServerRoot set to "/usr/local/apache" will be interpreted by the
# server as "/usr/local/apache/logs/foo.log".
#

### Section 1: Global Environment
#
# The directives in this section affect the overall operation of Apache,
# such as the number of concurrent requests it can handle or where it
# can find its configuration files.
#
```

```
#
# ServerType is either inetd, or standalone.  Inetd mode is only supported on
# Unix platforms.
#
ServerType standalone

#
# ServerRoot: The top of the directory tree under which the server's
# configuration, error, and log files are kept.
#
# NOTE!  If you intend to place this on an NFS (or otherwise network)
# mounted filesystem then please read the LockFile documentation
# (available at <URL:http://www.apache.org/docs/mod/core.html#lockfile>);
# you will save yourself a lot of trouble.
#
# Do NOT add a slash at the end of the directory path.
#
ServerRoot "/var/www"

#
# The LockFile directive sets the path to the lockfile used when Apache
# is compiled with either USE_FCNTL_SERIALIZED_ACCEPT or
# USE_FLOCK_SERIALIZED_ACCEPT. This directive should normally be left at
# its default value. The main reason for changing it is if the logs
# directory is NFS mounted, since the lockfile MUST BE STORED ON A LOCAL
# DISK. The PID of the main server process is automatically appended to
# the filename.
#
#LockFile logs/accept.lock

#
# PidFile: The file in which the server should record its process
# identification number when it starts.
#
PidFile logs/httpd.pid
#
# ScoreBoardFile: File used to store internal server process information.
# Not all architectures require this.  But if yours does (you'll know because
# this file will be  created when you run Apache) then you *must* ensure that
# no two invocations of Apache share the same scoreboard file.
#
ScoreBoardFile logs/apache_runtime_status

#
```

```
# In the standard configuration, the server will process httpd.conf,
# srm.conf, and access.conf in that order.  The latter two files are
# now distributed empty, as it is recommended that all directives
# be kept in a single file for simplicity.  The commented-out values
# below are the built-in defaults.  You can have the server ignore
# these files altogether by using "/dev/null" (for Unix) or
# "nul" (for Win32) for the arguments to the directives.
#
#ResourceConfig conf/srm.conf
#AccessConfig conf/access.conf

#
# Timeout: The number of seconds before receives and sends time out.
#
Timeout 300

#
# KeepAlive: Whether or not to allow persistent connections (more than
# one request per connection). Set to "Off" to deactivate.
#
KeepAlive On

#
# MaxKeepAliveRequests: The maximum number of requests to allow
# during a persistent connection. Set to 0 to allow an unlimited amount.
# We recommend you leave this number high, for maximum performance.
#
MaxKeepAliveRequests 100

#
# KeepAliveTimeout: Number of seconds to wait for the next request from the
# same client on the same connection.
#
KeepAliveTimeout 15

#
# Server-pool size regulation.  Rather than making you guess how many
# server processes you need, Apache dynamically adapts to the load it
# sees --- that is, it tries to maintain enough server processes to
# handle the current load, plus a few spare servers to handle transient
# load spikes (e.g., multiple simultaneous requests from a single
# Netscape browser).
#
# It does this by periodically checking how many servers are waiting
# for a request.  If there are fewer than MinSpareServers, it creates
```

```
# a new spare.  If there are more than MaxSpareServers, some of the
# spares die off.  The default values in httpd.conf-dist are probably OK
# for most sites.
#
MinSpareServers 5
MaxSpareServers 10

#
# Number of servers to start initially --- should be a reasonable ballpark
# figure.
#
StartServers 5

#
# Limit on total number of servers running, i.e., limit on the number
# of clients who can simultaneously connect --- if this limit is ever
# reached, clients will be LOCKED OUT, so it should NOT BE SET TOO LOW.
# It is intended mainly as a brake to keep a runaway server from taking
# the system with it as it spirals down...
#
maxClients 160

#
# MaxRequestsPerChild: the number of requests each child process is
# allowed to process before the child dies.  The child will exit so
# as to avoid problems after prolonged use when Apache (and maybe the
# libraries it uses) leak memory or other resources.  On most systems, this
# isn't really needed, but a few (such as Solaris) do have notable leaks
# in the libraries.
#
MaxRequestsPerChild 30

#
# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, in addition to the default. See also the <VirtualHost>
# directive.
#
#Listen 3000
#Listen 12.34.56.78:80

#
# BindAddress: You can support virtual hosts with this option. This directive
# is used to tell the server which IP address to listen to. It can either
# contain "*", an IP address, or a fully qualified Internet domain name.
# See also the <VirtualHost> and Listen directives.
```

```
#
#BindAddress *

#
# Dynamic Shared Object (DSO) Support
#
# To be able to use the functionality of a module which was built as a DSO you
# have to place corresponding `LoadModule' lines at this location so the
# directives contained in it are actually available _before_ they are used.
# Please read the file README.DSO in the Apache 1.3 distribution for more
# details about the DSO mechanism and run `httpd -l' for the list of already
# built-in (statically linked and thus always available) modules in your httpd
# binary.
#
# Note: The order is which modules are loaded is important.  Don't change
# the order below without expert advice.
#
# Example:
# LoadModule foo_module libexec/mod_foo.so
LoadModule php4_module        /usr/lib/apache/modules/libphp4.so

#AddModule mod_php4.c

#
# ExtendedStatus controls whether Apache will generate "full" status
# information (ExtendedStatus On) or just basic information (ExtendedStatus
# Off) when the "server-status" handler is called. The default is Off.
#
ExtendedStatus On

### Section 2: 'Main' server configuration
#
# The directives in this section set up the values used by the 'main'
# server, which responds to any requests that aren't handled by a
# <VirtualHost> definition.  These values also provide defaults for
# any <VirtualHost> containers you may define later in the file.
#
# All of these directives may appear inside <VirtualHost> containers,
# in which case these default settings will be overridden for the
# virtual host being defined.
#

#
# If your ServerType directive (set earlier in the 'Global Environment'
# section) is set to "inetd", the next few directives don't have any
```

```
# effect since their settings are defined by the inetd configuration.
# Skip ahead to the ServerAdmin directive.
#

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

##
##  SSL Support
##
##  When we also provide SSL we have to listen to the
##  standard HTTP port (see above) and to the HTTPS port
##
#<IfDefine SSL>
#Listen 80
#Listen 443
#</IfDefine>

#
# If you wish httpd to run as a different user or group, you must run
# httpd as root initially and it will switch.
#
# 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 #-1 on these systems!
#  On OpenBSD, use user www, group www.
#
User www
Group www

#
# ServerAdmin: Your address, where problems with the server should be
# e-mailed.  This address appears on some server-generated pages, such
# as error documents.
#
#ServerAdmin you@your.address
 ServerAdmin dalilatn@hotmail.com
#
```

```
# ServerName allows you to set a host name which is sent back to clients for
# your server if it's different than the one the program would get (i.e., use
# "www" instead of the host's real name).
#
# Note: You cannot just invent host names and hope they work. The name you
# define here must be a valid DNS name for your host. If you don't understand
# this, ask your network administrator.
# If your host doesn't have a registered DNS name, enter its IP address here.
# You will have to access it by its address (e.g., http://123.45.67.89/)
# anyway, and this will make redirections work in a sensible way.
#
ServerName galileo.filos.unam.mx

#
# 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 "/var/www/htdocs"

#
# Each directory to which Apache has access, can be configured with respect
# to which services and features are allowed and/or disabled in that
# directory (and its subdirectories).
#
# First, we configure the "default" to be a very restrictive set of
# permissions.
#
<Directory />
    Options FollowSymLinks
    AllowOverride None
</Directory>

#
# Note that from this point forward you must specifically allow
# particular features to be enabled - so if something's not working as
# you might expect, make sure that you have specifically enabled it
# below.
#

#
# This should be changed to whatever you set DocumentRoot to.
#
<Directory "/var/www/htdocs">
```

```
#
# This may also be "None", "All", or any combination of "Indexes",
# "Includes", "FollowSymLinks", "ExecCGI", or "MultiViews".
#
# Note that "MultiViews" must be named *explicitly* --- "Options All"
# doesn't give it to you.
#
    Options -Indexes FollowSymLinks

#
# This controls which options the .htaccess files in directories can
# override. Can also be "All", or any combination of "Options", "FileInfo",
# "AuthConfig", and "Limit"
#
    AllowOverride None

#
# Controls who can get stuff from this server.
#
    Order allow,deny
    Allow from all
</Directory>

#
# UserDir: The name of the directory which is appended onto a user's home
# directory if a ~user request is received.  "disabled" turns this feature
# off; other reasonable defaults are "public_dir" and ".html"
#
UserDir disabled

#
# Control access to UserDir directories.  The following is an example
# for a site where these directories are restricted to read-only and
# are located under /home/<username>public_html
# You will need to change this to match your site's home directories.
#
#<Directory /home/*/public_html>
#    AllowOverride FileInfo AuthConfig Limit
#    Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
#    <Limit GET POST OPTIONS PROPFIND>
#        Order allow,deny
#        Allow from all
#    </Limit>
#        <Limit PUT DELETE PATCH PROPPATCH MKCOL COPY MOVE LOCK
UNLOCK>
```

```
#       Order deny,allow
#       Deny from all
#    </Limit>
#</Directory>

<Directory "/var/www/htdocs/interno">
   Options -Indexes FollowSymLinks
   AllowOverride None
   Order allow,deny
   Allow from 132.248.123.39 132.248.123.49 132.248.123.40 132.248.123.46
132.248.123.41 132.248.123.70 132.248.123.73 132.248.123.52 132.248.123.53
132.248.123.59 132.248.123.44
#132.248.123.38 132.248.123.46 132.248.123.59 132.248.123.41
</Directory>

<Directory "/var/www/htdocs/interno/manual">
   Options Indexes FollowSymLinks
   AllowOverride None
   Order allow,deny
   Allow from 132.248.123.39
</Directory>

<Directory "/var/www/htdocs/ventanillas">
   Options -Indexes FollowSymLinks
   AllowOverride None
   Order allow,deny
   Allow from 132.248.123.52 132.248.123.53 132.248.123.44 132.248.123.42
132.248.123.39 132.248.123.49 132.248.123.46 132.248.123.73
</Directory>

<Directory "/var/www/htdocs/coordinacionHist">
   Options -Indexes FollowSymLinks
   AllowOverride None
   Order allow,deny
   Allow from 132.248.123.39 132.248.123.49 132.248.123.38 132.248.123.70
132.248.123.46  132.248.123.41
</Directory>


#
# DirectoryIndex: Name of the file or files to use as a pre-written HTML
# directory index.  Separate multiple entries with spaces.
#
DirectoryIndex index.html index.php
```

```
#
# AccessFileName: The name of the file to look for in each directory
# for access control information.
#
AccessFileName .htaccess

#
# The following lines prevent .htaccess files from being viewed by
# Web clients.  Since .htaccess files often contain authorization
# information, access is disallowed for security reasons.  Comment
# these lines out if you want Web visitors to see the contents of
# .htaccess files.  If you change the AccessFileName directive above,
# be sure to make the corresponding changes here.
#
<Files .htaccess>
    Order allow,deny
    Deny from all
</Files>

#
# CacheNegotiatedDocs: By default, Apache sends "Pragma: no-cache" with each
# document that was negotiated on the basis of content. This asks proxy
# servers not to cache the document. Uncommenting the following line disables
# this behavior, and proxies will be allowed to cache the documents.
#
#CacheNegotiatedDocs

#
# UseCanonicalName:  (new for 1.3)  With this setting turned on, whenever
# Apache needs to construct a self-referencing URL (a URL that refers back
# to the server the response is coming from) it will use ServerName and
# Port to form a "canonical" name.  With this setting off, Apache will
# use the hostname:port that the client supplied, when possible.  This
# also affects SERVER_NAME and SERVER_PORT in CGI scripts.
#
UseCanonicalName On

#
# TypesConfig describes where the mime.types file (or equivalent) is
# to be found.
#
TypesConfig conf/mime.types

#
# DefaultType is the default MIME type the server will use for a document
```

```
# if it cannot otherwise determine one, such as from filename extensions.
# If your server contains mostly text or HTML documents, "text/plain" is
# a good value.  If most of your content is binary, such as applications
# or images, you may want to use "application/octet-stream" instead to
# keep browsers from trying to display binary files as though they are
# text.
#
DefaultType text/plain

#
# The mod_mime_magic module allows the server to use various hints from the
# contents of the file itself to determine its type.  The MIMEMagicFile
# directive tells the module where the hint definitions are located.
# mod_mime_magic is not part of the default server (you have to add
# it yourself with a LoadModule [see the DSO paragraph in the 'Global
# Environment' section], or recompile the server and include mod_mime_magic
# as part of the configuration), so it's enclosed in an <IfModule> container.
# This means that the MIMEMagicFile directive will only be processed if the
# module is part of the server.
#
<IfModule mod_mime_magic.c>
    MIMEMagicFile conf/magic
</IfModule>

#
# HostnameLookups: Log the names of clients or just their IP addresses
# e.g., www.apache.org (on) or 204.62.129.132 (off).
# The default is off because it'd be overall better for the net if people
# had to knowingly turn this feature on, since enabling it means that
# each client request will result in AT LEAST one lookup request to the
# nameserver.
#
HostnameLookups Off

#
# ErrorLog: The location of the error log file.
# If you do not specify an ErrorLog directive within a <VirtualHost>
# container, error messages relating to that virtual host will be
# logged here.  If you *do* define an error logfile for a <VirtualHost>
# container, that host's errors will be logged there and not here.
#
ErrorLog logs/error_log

#
# LogLevel: Control the number of messages logged to the error_log.
```

```
# Possible values include: debug, info, notice, warn, error, crit,
# alert, emerg.
#
LogLevel warn

#
# The following directives define some format nicknames for use with
# a CustomLog directive (see below).
#
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\""
combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent

#
# The location and format of the access logfile (Common Logfile Format).
# If you do not define any access logfiles within a <VirtualHost>
# container, they will be logged here.  Contrariwise, if you *do*
# define per-<VirtualHost> access logfiles, transactions will be
# logged therein and *not* in this file.
#
CustomLog logs/access_log common

#
# If you would like to have agent and referer logfiles, uncomment the
# following directives.
#
#CustomLog logs/referer_log referer
#CustomLog logs/agent_log agent

#
# If you prefer a single logfile with access, agent, and referer information
# (Combined Logfile Format) you can use the following directive.
#
#CustomLog logs/access_log combined

#
# Optionally add a line containing the server version and virtual host
# name to server-generated pages (error documents, FTP directory listings,
# mod_status and mod_info output etc., but not CGI generated documents).
# Set to "EMail" to also include a mailto: link to the ServerAdmin.
# Set to one of:  On | Off | EMail
#
ServerSignature On
```

```
#
# Aliases: Add here as many aliases as you need (with no limit). The format is
# Alias fakename realname
#
# Note that if you include a trailing / on fakename then the server will
# require it to be present in the URL.  So "/icons" isn't aliased in this
# example, only "/icons/"..
#
Alias /icons/ "/var/www/icons/"

<Directory "/var/www/icons">
    Options -Indexes MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>

#
# ScriptAlias: This controls which directories contain server scripts.
# ScriptAliases are essentially the same as Aliases, except that
# documents in the realname directory are treated as applications and
# run by the server when requested rather than as documents sent to the client.
# The same rules about trailing "/" apply to ScriptAlias directives as to
# Alias.
#
ScriptAlias /cgi-bin/ "/var/www/cgi-bin/"

#
# "/var/www/cgi-bin" should be changed to whatever your ScriptAliased
# CGI directory exists, if you have that configured.
#
<Directory "/var/www/cgi-bin">
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
</Directory>

#
# Redirect allows you to tell clients about documents which used to exist in
# your server's namespace, but do not anymore. This allows you to tell the
# clients where to look for the relocated document.
# Format: Redirect old-URI new-URL
#
```

```
RedirectMatch (.*)\root.exe$ http://no_existe.com$1
RedirectMatch (.*)\cmd.exe$ http://no_existe.com$1
RedirectMatch (.*)\cmd1.exe$ http://no_existe.com$1
RedirectMatch (.*)\favicon.ico$ http://abcdefghij.klmoppo.mmsdjsdf.com$1
RedirectMatch (.*)\default.ida$ http://abcdefghij.klmoppo.mmsdjsdf.com$1
RedirectMatch (.*)\_vti_bin/shtml\.exe/_vti_rpc$
http://abcdefghij.klmoppo.mmsdjsdf.com$1
RedirectMatch (.*)\inscripcion.html$ http://galileo.filos.unam.mx/index.html
RedirectMatch (.*)\registro.html$ http://galileo.filos.unam.mx/index.html
RedirectMatch (.*)\cambios.html$ http://galileo.filos.unam.mx$1

#
# Directives controlling the display of server-generated directory listings.
#

#
# FancyIndexing is whether you want fancy directory indexing or standard
#
IndexOptions FancyIndexing

#
# AddIcon* directives tell the server which icon to show for different
# files or filename extensions.  These are only displayed for
# FancyIndexed directories.
#
AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip

AddIconByType (TXT,/icons/text.gif) text/*
AddIconByType (IMG,/icons/image2.gif) image/*
AddIconByType (SND,/icons/sound2.gif) audio/*
AddIconByType (VID,/icons/movie.gif) video/*

AddIcon /icons/binary.gif .bin .exe
AddIcon /icons/binhex.gif .hqx
AddIcon /icons/tar.gif .tar
AddIcon /icons/world2.gif .wrl .wrl.gz .vrml .vrm .iv
AddIcon /icons/compressed.gif .Z .z .tgz .gz .zip
AddIcon /icons/a.gif .ps .ai .eps
AddIcon /icons/layout.gif .html .shtml .htm .pdf
AddIcon /icons/text.gif .txt
AddIcon /icons/c.gif .c
AddIcon /icons/p.gif .pl .py
AddIcon /icons/f.gif .for
AddIcon /icons/dvi.gif .dvi
AddIcon /icons/uuencoded.gif .uu
```

```
AddIcon /icons/script.gif .conf .sh .shar .csh .ksh .tcl
AddIcon /icons/tex.gif .tex
AddIcon /icons/bomb.gif core

AddIcon /icons/back.gif ..
AddIcon /icons/hand.right.gif README
AddIcon /icons/folder.gif ^^DIRECTORY^^
AddIcon /icons/blank.gif ^^BLANKICON^^

#
# DefaultIcon is which icon to show for files which do not have an icon
# explicitly set.
#
DefaultIcon /icons/unknown.gif

#
# AddDescription allows you to place a short description after a file in
# server-generated indexes.  These are only displayed for FancyIndexed
# directories.
# Format: AddDescription "description" filename
#
#AddDescription "GZIP compressed document" .gz
#AddDescription "tar archive" .tar
#AddDescription "GZIP compressed tar archive" .tgz
```

```
#
# ReadmeName is the name of the README file the server will look for by
# default, and append to directory listings.
#
# HeaderName is the name of a file which should be prepended to
# directory indexes.
#
# The server will first look for name.html and include it if found.
# If name.html doesn't exist, the server will then look for name.txt
# and include it as plaintext if found.
#
ReadmeName README
HeaderName HEADER

#
# IndexIgnore is a set of filenames which directory indexing should ignore
# and not include in the listing.  Shell-style wildcarding is permitted.
#
IndexIgnore .??* *~ *# HEADER* README* RCS CVS *,v *,t
```

```
#
# AddEncoding allows you to have certain browsers (Mosaic/X 2.1+) uncompress
# information on the fly. Note: Not all browsers support this.
# Despite the name similarity, the following Add* directives have nothing
# to do with the FancyIndexing customization directives above.
#
AddEncoding x-compress Z
AddEncoding x-gzip gz

#
# AddLanguage allows you to specify the language of a document. You can
# then use content negotiation to give a browser a file in a language
# it can understand.  Note that the suffix does not have to be the same
# as the language keyword --- those with documents in Polish (whose
# net-standard language code is pl) may wish to use "AddLanguage pl .po"
# to avoid the ambiguity with the common suffix for perl scripts.
#
AddLanguage en .en
AddLanguage fr .fr
AddLanguage de .de
AddLanguage da .da
AddLanguage el .el
AddLanguage it .it
#
# LanguagePriority allows you to give precedence to some languages
# in case of a tie during content negotiation.
# Just list the languages in decreasing order of preference.
#
LanguagePriority en fr de

#
# AddType allows you to tweak mime.types without actually editing it, or to
# make certain files to be certain types.
#
# For example, the PHP3 module (not part of the Apache distribution)
# will typically use:
#
#AddType application/x-httpd-php3 .phtml
#AddType application/x-httpd-php3-source .phps
#
#AddType application/x-httpd-phP444444 .php
AddType application/x-httpd-php .php
#
# AddHandler allows you to map certain file extensions to "handlers",
```

```
# actions unrelated to filetype. These can be either built into the server
# or added with the Action command (see below)
#
# If you want to use server side includes, or CGI outside
# ScriptAliased directories, uncomment the following lines.
#
# To use CGI scripts:
#
#AddHandler cgi-script .cgi

#
# To use server-parsed HTML files
#
#AddType text/html .shtml
#AddHandler server-parsed .shtml

#
# Uncomment the following line to enable Apache's send-asis HTTP file
# feature
#
#AddHandler send-as-is asis

#
# If you wish to use server-parsed imagemap files, use
#
#AddHandler imap-file map

#
# To enable type maps, you might want to use
#
#AddHandler type-map var

#
# Action lets you define media types that will execute a script whenever
# a matching file is called. This eliminates the need for repeated URL
# pathnames for oft-used CGI file processors.
# Format: Action media/type /cgi-script/location
# Format: Action handler-name /cgi-script/location
#

#
# MetaDir: specifies the name of the directory in which Apache can find
# meta information files. These files contain additional HTTP headers
# to include when sending the document
#
```

```
#MetaDir .web

#
# MetaSuffix: specifies the file name suffix for the file containing the
# meta information.
#
#MetaSuffix .meta

#
# Customizable error response (Apache style)
#  these come in three flavors
#
#    1) plain text
#ErrorDocument 500 "The server made a boo boo.
#  n.b.  the (") marks it as text, it does not get output
#
#    2) local redirects
#ErrorDocument 404 /missing.html
#  to redirect to local URL /missing.html
#ErrorDocument 404 /cgi-bin/missing_handler.pl
#  N.B.: You can redirect to a script or a document using server-side-includes.
#
#    3) external redirects
#ErrorDocument 402 http://some.other_server.com/subscription_info.html
#  N.B.: Many of the environment variables associated with the original
#  request will *not* be available to such a script.

#
# The following directives modify normal HTTP response behavior.
# The first directive disables keepalive for Netscape 2.x and browsers that
# spoof it. There are known problems with these browser implementations.
# The second directive is for Microsoft Internet Explorer 4.0b2
# which has a broken HTTP/1.1 implementation and does not properly
# support keepalive when it is used on 301 or 302 (redirect) responses.
#
BrowserMatch "Mozilla/2" nokeepalive
BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0 force-response-1.0

#
# The following directive disables HTTP/1.1 responses to browsers which
# are in violation of the HTTP/1.0 spec by not being able to grok a
# basic 1.1 response.
#
BrowserMatch "RealPlayer 4\.0" force-response-1.0
BrowserMatch "Java/1\.0" force-response-1.0
```

```
BrowserMatch "JDK/1\.0" force-response-1.0

#
# Allow server status reports, with the URL of http://servername/server-status
# Change the ".your_domain.com" to match your domain to enable.
#
<Location /server-status>
    SetHandler server-status
    Order deny,allow
    Deny from all
    Allow from 132.248.123.39
</Location>

#
# Allow remote server configuration reports, with the URL of
#  http://servername/server-info (requires that mod_info.c be loaded).
# Change the ".your_domain.com" to match your domain to enable.
#
<Location /server-info>
    SetHandler server-info
    Order deny,allow
    Deny from all
    Allow from 132.248.123.39
</Location>
```

```
#
# There have been reports of people trying to abuse an old bug from pre-1.1
# days.  This bug involved a CGI script distributed as a part of Apache.
# By uncommenting these lines you can redirect these attacks to a logging
# script on phf.apache.org.  Or, you can record them yourself, using the script
# support/phf_abuse_log.cgi.
#
#<Location /cgi-bin/phf*>
#    Deny from all
#    ErrorDocument 403 http://phf.apache.org/phf_abuse_log.cgi
#</Location>

#
# Proxy Server directives. Uncomment the following lines to
# enable the proxy server:
#
#<IfModule mod_proxy.c>
#ProxyRequests On
#
#<Directory proxy:*>
```

```
#    Order deny,allow
#    Deny from all
#    Allow from .your_domain.com
#</Directory>

#
# Enable/disable the handling of HTTP/1.1 "Via:" headers.
# ("Full" adds the server version; "Block" removes all outgoing Via: headers)
# Set to one of: Off | On | Full | Block
#
#ProxyVia On

#
# To enable the cache as well, edit and uncomment the following lines:
# (no cacheing without CacheRoot)
#
#CacheRoot "/var/www/proxy"
#CacheSize 5
#CacheGcInterval 4
#CacheMaxExpire 24
#CacheLastModifiedFactor 0.1
#CacheDefaultExpire 1
#NoCache a_domain.com another_domain.edu joes.garage_sale.com
#</IfModule>
# End of proxy directives.

### Section 3: Virtual Hosts
#
# VirtualHost: If you want to maintain multiple domains/hostnames on your
# machine you can setup VirtualHost containers for them.
# Please see the documentation at <URL:http://www.apache.org/docs/vhosts/>
# for further details before you try to setup virtual hosts.
# You may use the command line option '-S' to verify your virtual host
# configuration.

#
# If you want to use name-based virtual hosts you need to define at
# least one IP address (and port number) for them.
#
#NameVirtualHost 12.34.56.78:80
#NameVirtualHost 12.34.56.78

#
# VirtualHost example:
```

```
# Almost any Apache directive may go into a VirtualHost container.
#
#<VirtualHost ip.address.of.host.some_domain.com>
#    ServerAdmin webmaster@host.some_domain.com
#    DocumentRoot /www/docs/host.some_domain.com
#    ServerName host.some_domain.com
#    ErrorLog logs/host.some_domain.com-error_log
#    CustomLog logs/host.some_domain.com-access_log common
#</VirtualHost>

#<VirtualHost _default_:*>
#</VirtualHost>


##
##  SSL Global Context
##
##  All SSL configuration in this context applies both to
##  the main server and all SSL-enabled virtual hosts.
##

#
#   Some MIME-types for downloading Certificates and CRLs
#
<IfDefine SSL>
AddType application/x-x509-ca-cert .crt
AddType application/x-pkcs7-crl    .crl
</IfDefine>

<IfModule mod_ssl.c>

#   Pass Phrase Dialog:
#   Configure the pass phrase gathering process.
#   The filtering dialog program (`builtin' is a internal
#   terminal dialog) has to provide the pass phrase on stdout.
SSLPassPhraseDialog  builtin

#   Inter-Process Session Cache:
#   Configure the SSL Session Cache: First either `none'
#   or `dbm:/path/to/file' for the mechanism to use and
#   second the expiring timeout (in seconds).
SSLSessionCache         dbm:logs/ssl_scache
SSLSessionCacheTimeout  300

#   Semaphore:
```

```
#   Configure the path to the mutual exclusion semaphore the
#   SSL engine uses internally for inter-process synchronization.
SSLMutex  file:logs/ssl_mutex

#   Pseudo Random Number Generator (PRNG):
#   Configure one or more sources to seed the PRNG of the
#   SSL library. The seed data should be of good random quality.
SSLRandomSeed startup builtin
SSLRandomSeed connect builtin
#SSLRandomSeed startup file:/dev/random  512
#SSLRandomSeed startup file:/dev/urandom 512
#SSLRandomSeed connect file:/dev/random  512
#SSLRandomSeed connect file:/dev/urandom 512
SSLRandomSeed startup file:/dev/arandom  512

#   Logging:
#   The home of the dedicated SSL protocol logfile. Errors are
#   additionally duplicated in the general error log file.  Put
#   this somewhere where it cannot be used for symlink attacks on
#   a real server (i.e. somewhere where only root can write).
#   Log levels are (ascending order: higher ones include lower ones):
#   none, error, warn, info, trace, debug.
SSLLog      logs/ssl_engine_log
SSLLogLevel info

</IfModule>

<IfDefine SSL>

##
## SSL Virtual Host Context
##

<VirtualHost _default_:443>

#  General setup for the virtual host
DocumentRoot /var/www/htdocs
ServerName new.host.name
ServerAdmin you@your.address
ErrorLog logs/error_log
TransferLog logs/access_log

#   SSL Engine Switch:
#   Enable/Disable SSL for this virtual host.
SSLEngine on
```

```
#   SSL Cipher Suite:
#   List the ciphers that the client is permitted to negotiate.
#   See the mod_ssl documentation for a complete list.
#SSLCipherSuite ALL:!ADH:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP

#   Server Certificate:
#   Point SSLCertificateFile at a PEM encoded certificate.  If
#   the certificate is encrypted, then you will be prompted for a
#   pass phrase.  Note that a kill -HUP will prompt again. A test
#   certificate can be generated with `make certificate' under
#   built time.
SSLCertificateFile    /etc/ssl/server.crt

#   Server Private Key:
#   If the key is not combined with the certificate, use this
#   directive to point at the key file.
SSLCertificateKeyFile /etc/ssl/private/server.key

#   Certificate Authority (CA):
#   Set the CA certificate verification path where to find CA
#   certificates for client authentication or alternatively one
#   huge file containing all of them (file must be PEM encoded)
#   Note: Inside SSLCACertificatePath you need hash symlinks
#         to point to the certificate files. Use the provided
#         Makefile to update the hash symlinks after changes.
#SSLCACertificatePath    @@ServerRoot@@/conf/ssl.crt
#SSLCACertificateFile    @@ServerRoot@@/conf/ssl.crt/ca-bundle.crt

#   Client Authentication (Type):
#   Client certificate verification type and depth.  Types are
#   none, optional, require and optional_no_ca.  Depth is a
#   number which specifies how deeply to verify the certificate
#   issuer chain before deciding the certificate is not valid.
#SSLVerifyClient require
#SSLVerifyDepth  10

#   Access Control:
#   With SSLRequire you can do per-directory access control based
#   on arbitrary complex boolean expressions containing server
#   variable checks and other lookup directives.  The syntax is a
#   mixture between C and Perl.  See the mod_ssl documentation
#   for more details.
#<Location />
#SSLRequire (    %{SSL_CIPHER} !~ m/^(EXP|NULL)-/ \
```

```
#           and %{SSL_CLIENT_S_DN_O} eq "Snake Oil, Ltd." \
#           and %{SSL_CLIENT_S_DN_OU} in {"Staff", "CA", "Dev"} \
#           and %{TIME_WDAY} >= 1 and %{TIME_WDAY} <= 5 \
#           and %{TIME_HOUR} >= 8 and %{TIME_HOUR} <= 20       ) \
#          or %{REMOTE_ADDR} =~ m/^192\.76\.162\.[0-9]+$/
#</Location>
#   SSL Engine Options:
#   Set various options for the SSL engine.
#   FakeBasicAuth:
#    Translate the client X.509 into a Basic Authorisation.  This means that
#    the standard Auth/DBMAuth methods can be used for access control.  The
#    user name is the `one line' version of the client's X.509 certificate.
#    Note that no password is obtained from the user. Every entry in the user
#    file needs this password: `xxj31ZMTZzkVA'.
#   ExportCertData:
#    This exports two additional environment variables: SSL_CLIENT_CERT and
#    SSL_SERVER_CERT. These contain the PEM-encoded certificates of the
#    server (always existing) and the client (only existing when client
#    authentication is used). This can be used to import the certificates
#    into CGI scripts.
#   CompatEnvVars:
#    This exports obsolete environment variables for backward compatibility
#    to Apache-SSL 1.x, mod_ssl 2.0.x, Sioux 1.0 and Stronghold 2.x. Use this
#    to provide compatibility to existing CGI scripts.
#SSLOptions +FakeBasicAuth +ExportCertData +CompatEnvVars
#   Per-Server Logging:
#   The home of a custom SSL log file. Use this when you want a
#   compact non-error SSL logfile on a virtual host basis.
CustomLog logs/ssl_request_log \
        "%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x \"%r\" %b"
SetEnvIf Request_URI "/*/cmd\.exe" nimda
SetEnvIf Request_URI "/*/root\.exe" nimda
SetEnvIf Request_URI "/MSADC/root\.exe" nimda
SetEnvIf Request_URI "/\.\." nimda
SetEnvIf Request_URI "\.\./" nimda
SetEnvIf Request_URI "/default\.ida" basura
SetEnvIf Request_URI "/favicon\.ico" basura
CustomLog /var/www/logs/access_log common env=!nimda
CustomLog /var/www/logs/access_log common env=!basura
</VirtualHost>

</IfDefine>
```

6. PHP.INI

```
[PHP]

;;;;;;;;;;;
; WARNING ;
;;;;;;;;;;;
; This is the default settings file for new PHP installations.
; By default, PHP installs itself with a configuration suitable for
; development purposes, and *NOT* for production purposes.
; For several security-oriented considerations that should be taken
; before going online with your site, please consult php.ini-recommended
; and http://php.net/manual/en/security.php.


;;;;;;;;;;;;;;;;;;;
; About this file ;
;;;;;;;;;;;;;;;;;;;
; This file controls many aspects of PHP's behavior.  In order for PHP to
; read it, it must be named 'php.ini'.  PHP looks for it in the current
; working directory, in the path designated by the environment variable
; PHPRC, and in the path that was defined in compile time (in that order).
; Under Windows, the compile-time path is the Windows directory.  The
; path in which the php.ini file is looked for can be overridden using
; the -c argument in command line mode.
;
; The syntax of the file is extremely simple.  Whitespace and Lines
; beginning with a semicolon are silently ignored (as you probably guessed).
; Section headers (e.g. [Foo]) are also silently ignored, even though
; they might mean something in the future.
;
; Directives are specified using the following syntax:
; directive = value
; Directive names are *case sensitive* - foo=bar is different from FOO=bar.
;
; The value can be a string, a number, a PHP constant (e.g. E_ALL or M_PI), one
; of the INI constants (On, Off, True, False, Yes, No and None) or an expression
; (e.g. E_ALL & ~E_NOTICE), or a quoted string ("foo").
;
; Expressions in the INI file are limited to bitwise operators and parentheses:
; |       bitwise OR
; &       bitwise AND
; ~       bitwise NOT
; !       boolean NOT
```

;
; Boolean flags can be turned on using the values 1, On, True or Yes.
; They can be turned off using the values 0, Off, False or No.
;
; An empty string can be denoted by simply not writing anything after the equal
; sign, or by using the None keyword:
;
; foo =   ; sets foo to an empty string
; foo = none   ; sets foo to an empty string
; foo = "none" ; sets foo to the string 'none'
;
; If you use constants in your value, and these constants belong to a
; dynamically loaded extension (either a PHP extension or a Zend extension),
; you may only use these constants *after* the line that loads the extension.
;
; All the values in the php.ini-dist file correspond to the builtin
; defaults (that is, if no php.ini is used, or if you delete these lines,
; the builtin defaults will be identical).

;;;;;;;;;;;;;;;;;;;
; Language Options ;
;;;;;;;;;;;;;;;;;;;

; Enable the PHP scripting language engine under Apache.
engine = On

; Allow the <? tag. Otherwise, only <?php and <script> tags are recognized.
short_open_tag = On

; Allow ASP-style <% %> tags.
asp_tags = Off

; The number of significant digits displayed in floating point numbers.
precision   = 14

; Enforce year 2000 compliance (will cause problems with non-compliant browsers)
y2k_compliance = Off

; Output buffering allows you to send header lines (including cookies) even
; after you send body content, at the price of slowing PHP's output layer a
; bit. You can enable output buffering during runtime by calling the output
; buffering functions. You can also enable output buffering for all files by
; setting this directive to On. If you wish to limit the size of the buffer
; to a certain size - you can use a maximum number of bytes instead of 'On', as

```
; a value for this directive (e.g., output_buffering=4096).
output_buffering = Off

; You can redirect all of the output of your scripts to a function.  For
; example, if you set output_handler to "ob_gzhandler", output will be
; transparently compressed for browsers that support gzip or deflate encoding.
; Setting an output handler automatically turns on output buffering.
output_handler =

; Transparent output compression using the zlib library
; Valid values for this option are 'off', 'on', or a specific buffer size
; to be used for compression (default is 4KB)
zlib.output_compression = Off

; Implicit flush tells PHP to tell the output layer to flush itself
; automatically after every output block.  This is equivalent to calling the
; PHP function flush() after each and every call to print() or echo() and each
; and every HTML block.  Turning this option on has serious performance
; implications and is generally recommended for debugging purposes only.
implicit_flush = Off

; Whether to enable the ability to force arguments to be passed by reference
; at function call time.  This method is deprecated and is likely to be
; unsupported in future versions of PHP/Zend.  The encouraged method of
; specifying which arguments should be passed by reference is in the function
; declaration.  You're encouraged to try and turn this option Off and make
; sure your scripts work properly with it in order to ensure they will work
; with future versions of the language (you will receive a warning each time
; you use this feature, and the argument will be passed by value instead of by
; reference).
allow_call_time_pass_reference = On


;
; Safe Mode
;
safe_mode = Off

; By default, Safe Mode does a UID compare check when
; opening files. If you want to relax this to a GID compare,
; then turn on safe_mode_gid.
safe_mode_gid = Off

; When safe_mode is on, UID/GID checks are bypassed when
; including files from this directory and its subdirectories.
```

```
; (directory must also be in include_path or full path must
; be used when including)
safe_mode_include_dir =

; When safe_mode is on, only executables located in the safe_mode_exec_dir
; will be allowed to be executed via the exec family of functions.
safe_mode_exec_dir =

; open_basedir, if set, limits all file operations to the defined directory
; and below.  This directive makes most sense if used in a per-directory
; or per-virtualhost web server configuration file.
;
;open_basedir =

; Setting certain environment variables may be a potential security breach.
; This directive contains a comma-delimited list of prefixes.  In Safe Mode,
; the user may only alter environment variables whose names begin with the
; prefixes supplied here.  By default, users will only be able to set
; environment variables that begin with PHP_ (e.g. PHP_FOO=BAR).
;
; Note:  If this directive is empty, PHP will let the user modify ANY
; environment variable!
safe_mode_allowed_env_vars = PHP_

; This directive contains a comma-delimited list of environment variables that
; the end user won't be able to change using putenv().  These variables will be
; protected even if safe_mode_allowed_env_vars is set to allow to change them.
safe_mode_protected_env_vars = LD_LIBRARY_PATH

; This directive allows you to disable certain functions for security reasons.
; It receives a comma-delimited list of function names.  This directive is
; *NOT* affected by whether Safe Mode is turned On or Off.
disable_functions =

; Colors for Syntax Highlighting mode.  Anything that's acceptable in
; <font color="??????"> would work.
highlight.string  = #CC0000
highlight.comment = #FF9900
highlight.keyword = #006600
highlight.bg      = #FFFFFF
highlight.default = #0000CC
highlight.html    = #000000


;
```

```
; Misc
;
; Decides whether PHP may expose the fact that it is installed on the server
; (e.g. by adding its signature to the Web server header).  It is no security
; threat in any way, but it makes it possible to determine whether you use PHP
; on your server or not.
expose_php = On


.....................
;;;;;;;;;;;;;;;;;;;
; Resource Limits ;
.....................
;;;;;;;;;;;;;;;;;;;

max_execution_time = 30     ; Maximum execution time of each script, in seconds
memory_limit = 8M      ; Maximum amount of memory a script may consume (8MB)


.................................
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Error handling and logging ;
.................................
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

; error_reporting is a bit-field.  Or each number up to get desired error
; reporting level
; E_ALL             - All errors and warnings
; E_ERROR           - fatal run-time errors
; E_WARNING         - run-time warnings (non-fatal errors)
; E_PARSE           - compile-time parse errors
; E_NOTICE          - run-time notices (these are warnings which often result
;                     from a bug in your code, but it's possible that it was
;                     intentional (e.g., using an uninitialized variable and
;                     relying on the fact it's automatically initialized to an
;                     empty string)
; E_CORE_ERROR      - fatal errors that occur during PHP's initial startup
; E_CORE_WARNING    - warnings (non-fatal errors) that occur during PHP's
;                     initial startup
; E_COMPILE_ERROR   - fatal compile-time errors
; E_COMPILE_WARNING - compile-time warnings (non-fatal errors)
; E_USER_ERROR      - user-generated error message
; E_USER_WARNING    - user-generated warning message
; E_USER_NOTICE     - user-generated notice message
;
; Examples:
;
;   - Show all errors, except for notices
;
```

```
;error_reporting = E_ALL & ~E_NOTICE
;
;   - Show only errors
;
;error_reporting = E_COMPILE_ERROR|E_ERROR|E_CORE_ERROR
;
;   - Show all errors except for notices
;
error_reporting  =  E_ALL & ~E_NOTICE

; Print out errors (as a part of the output).  For production web sites,
; you're strongly encouraged to turn this feature off, and use error logging
; instead (see below).  Keeping display_errors enabled on a production web site
; may reveal security information to end users, such as file paths on your Web
; server, your database schema or other information.
display_errors = On

; Even when display_errors is on, errors that occur during PHP's startup
; sequence are not displayed.  It's strongly recommended to keep
; display_startup_errors off, except for when debugging.
display_startup_errors = Off

; Log errors into a log file (server-specific log, stderr, or error_log (below))
; As stated above, you're strongly advised to use error logging in place of
; error displaying on production web sites.
log_errors = Off

; Store the last error/warning message in $php_errormsg (boolean).
track_errors = Off

; Disable the inclusion of HTML tags in error messages.
;html_errors = Off

; String to output before an error message.
;error_prepend_string = "<font color=ff0000>"

; String to output after an error message.
;error_append_string = "</font>"

; Log errors to specified file.
;error_log = filename

; Log errors to syslog (Event Log on NT, not valid in Windows 95).
;error_log = syslog
```

; Warn if the + operator is used with strings.
warn_plus_overloading = Off

;;;;;;;;;;;;;;;;;;;
; Data Handling ;
;;;;;;;;;;;;;;;;;;;
;
; Note - track_vars is ALWAYS enabled as of PHP 4.0.3

; The separator used in PHP generated URLs to separate arguments.
; Default is "&".
;arg_separator.output = "&"

; List of separator(s) used by PHP to parse input URLs into variables.
; Default is "&".
; NOTE: Every character in this directive is considered as separator!
;arg_separator.input = ";&"

; This directive describes the order in which PHP registers GET, POST, Cookie,
; Environment and Built-in variables (G, P, C, E & S respectively, often
; referred to as EGPCS or GPC). Registration is done from left to right, newer
; values override older values.
variables_order = "EGPCS"

; Whether or not to register the EGPCS variables as global variables. You may
; want to turn this off if you don't want to clutter your scripts' global scope
; with user data. This makes most sense when coupled with track_vars - in which
; case you can access all of the GPC variables through the $HTTP_*_VARS[],
; variables.
;
; You should do your best to write your scripts so that they do not require
; register_globals to be on; Using form variables as globals can easily lead
; to possible security problems, if the code is not very well thought of.
register_globals = On

; This directive tells PHP whether to declare the argv&argc variables (that
; would contain the GET information). If you don't use these variables, you
; should turn it off for increased performance.
register_argc_argv = On

; Maximum size of POST data that PHP will accept.
post_max_size = 8M

; This directive is deprecated. Use variables_order instead.

```
gpc_order = "GPC"

; Magic quotes
;

; Magic quotes for incoming GET/POST/Cookie data.
magic_quotes_gpc = On

; Magic quotes for runtime-generated data, e.g. data from SQL, from exec(), etc.
magic_quotes_runtime = Off

; Use Sybase-style magic quotes (escape ' with " instead of \').
magic_quotes_sybase = Off

; Automatically add files before or after any PHP document.
auto_prepend_file =
auto_append_file =

; As of 4.0b4, PHP always outputs a character encoding by default in
; the Content-type: header.  To disable sending of the charset, simply
; set it to be empty.
;
; PHP's built-in default is text/html
default_mimetype = "text/html"
;default_charset = "iso-8859-1"


;;;;;;;;;;;;;;;;;;;;;;;;;
; Paths and Directories ;
;;;;;;;;;;;;;;;;;;;;;;;;;

; UNIX: "/path1:/path2"
;include_path = ".:/php/includes"
;
; Windows: "\path1;\path2"
;include_path = ".;c:\php\includes"

; The root of the PHP pages, used only if nonempty.
doc_root =

; The directory under which PHP opens the script using /~usernamem used only
; if nonempty.
user_dir =

; Directory in which the loadable extensions (modules) reside.
```

```
extension_dir = ./

; Whether or not to enable the dl() function.  The dl() function does NOT work
; properly in multithreaded servers, such as IIS or Zeus, and is automatically
; disabled on them.
enable_dl = On


;;;;;;;;;;;;;;;;;
; File Uploads ;
;;;;;;;;;;;;;;;;;

; Whether to allow HTTP file uploads.
file_uploads = On

; Temporary directory for HTTP uploaded files (will use system default if not
; specified).
;upload_tmp_dir =

; Maximum allowed size for uploaded files.
upload_max_filesize = 2M


;;;;;;;;;;;;;;;;;;;;
; Fopen wrappers ;
;;;;;;;;;;;;;;;;;;;;

; Whether to allow the treatment of URLs (like http:// or ftp://) as files.
allow_url_fopen = Off

; Define the anonymous ftp password (your email address)
;from="john@doe.com"


;;;;;;;;;;;;;;;;;;;;;;;
; Dynamic Extensions ;
;;;;;;;;;;;;;;;;;;;;;;;
;
; If you wish to have an extension loaded automatically, use the following
; syntax:
;
;   extension=modulename.extension
;
; For example, on Windows:
;
```

```
;   extension=msql.dll
;
; ... or under UNIX:
;
;   extension=msql.so
;
; Note that it should be the name of the module only; no directory information
; needs to go here.  Specify the location of the extension with the
; extension_dir directive above.


;Windows Extensions
;Note that MySQL and ODBC support is now built in, so no dll is needed for it.
;
;extension=php_bz2.dll
;extension=php_ctype.dll
;extension=php_cpdf.dll
;extension=php_curl.dll
;extension=php_cybercash.dll
;extension=php_db.dll
;extension=php_dba.dll
;extension=php_dbase.dll
;extension=php_dbx.dll
;extension=php_domxml.dll
;extension=php_dotnet.dll
;extension=php_exif.dll
;extension=php_fbsql.dll
;extension=php_fdf.dll
;extension=php_filepro.dll
;extension=php_gd.dll
;extension=php_gettext.dll
;extension=php_hyperwave.dll
;extension=php_iconv.dll
;extension=php_ifx.dll
;extension=php_iisfunc.dll
;extension=php_imap.dll
;extension=php_ingres.dll
;extension=php_interbase.dll
;extension=php_java.dll
;extension=php_ldap.dll
;extension=php_mbstring.dll
;extension=php_mcrypt.dll
;extension=php_mhash.dll
;extension=php_ming.dll
;extension=php_mssql.dll
```

```
;extension=php_oci8.dll
;extension=php_openssl.dll
;extension=php_oracle.dll
;extension=php_pdf.dll
;extension=php_pgsql.dll
;extension=php_printer.dll
;extension=php_sablot.dll
;extension=php_shmop.dll
;extension=php_snmp.dll
;extension=php_sockets.dll
;extension=php_sybase_ct.dll
;extension=php_xslt.dll
;extension=php_yaz.dll
;extension=php_zlib.dll


;;;;;;;;;;;;;;;;;;;
; Module Settings ;
;;;;;;;;;;;;;;;;;;;

[Syslog]
; Whether or not to define the various syslog variables (e.g. $LOG_PID,
; $LOG_CRON, etc.).  Turning it off is a good idea performance-wise.  In
; runtime, you can define these variables by calling define_syslog_variables().
define_syslog_variables  = Off

[mail function]
; For Win32 only.
SMTP = localhost

; For Win32 only.
sendmail_from = me@localhost.com

; For Unix only.  You may supply arguments as well (default: 'sendmail -t -i').
;sendmail_path =

[Logging]
; These configuration directives are used by the example logging mechanism.
; See examples/README.logging for more explanation.
;logging.method = db
;logging.directory = /path/to/log/directory

[Java]
;java.class.path = .\php_java.jar
;java.home = c:\jdk
```

```
;java.library = c:\jdk\jre\bin\hotspot\jvm.dll
;java.library.path = .\

[SQL]
sql.safe_mode = Off

[ODBC]
;odbc.default_db    =  Not yet implemented
;odbc.default_user  =  Not yet implemented
;odbc.default_pw    =  Not yet implemented

; Allow or prevent persistent links.
odbc.allow_persistent = On

; Check that a connection is still valid before reuse.
odbc.check_persistent = On

; Maximum number of persistent links.  -1 means no limit.
odbc.max_persistent = -1

; Maximum number of links (persistent + non-persistent).  -1 means no limit.
odbc.max_links = -1

; Handling of LONG fields.  Returns number of bytes to variables.  0 means
; passthru.
odbc.defaultlrl = 4096

; Handling of binary data.  0 means passthru, 1 return as is, 2 convert to char.
; See the documentation on odbc_binmode and odbc_longreadlen for an explanation
; of uodbc.defaultlrl and uodbc.defaultbinmode
odbc.defaultbinmode = 1

[MySQL]
; Allow or prevent persistent links.
mysql.allow_persistent = On

; Maximum number of persistent links.  -1 means no limit.
mysql.max_persistent = -1

; Maximum number of links (persistent + non-persistent).  -1 means no limit.
mysql.max_links = -1

; Default port number for mysql_connect().  If unset, mysql_connect() will use
; the $MYSQL_TCP_PORT or the mysql-tcp entry in /etc/services or the
; compile-time value defined MYSQL_PORT (in that order).  Win32 will only look
```

```
' at MYSQL_PORT.
mysql.default_port =

; Default socket name for local MySQL connects.  If empty, uses the built-in
; MySQL defaults.
mysql.default_socket =

; Default host for mysql_connect() (doesn't apply in safe mode).
mysql.default_host =

; Default user for mysql_connect() (doesn't apply in safe mode).
mysql.default_user =

; Default password for mysql_connect() (doesn't apply in safe mode).
; Note that this is generally a *bad* idea to store passwords in this file.
; *Any* user with PHP access can run 'echo cfg_get_var("mysql.default_password")
; and reveal this password!  And of course, any users with read access to this
; file will be able to reveal the password as well.
mysql.default_password =

[mSQL]
; Allow or prevent persistent links.
msql.allow_persistent = On

; Maximum number of persistent links.  -1 means no limit.
msql.max_persistent = -1

; Maximum number of links (persistent+non persistent).  -1 means no limit.
msql.max_links = -1

[PostgresSQL]
; Allow or prevent persistent links.
pgsql.allow_persistent = On

; Maximum number of persistent links.  -1 means no limit.
pgsql.max_persistent = -1

; Maximum number of links (persistent+non persistent).  -1 means no limit.
pgsql.max_links = -1

[Sybase]
; Allow or prevent persistent links.
sybase.allow_persistent = On

; Maximum number of persistent links.  -1 means no limit.
```

```
sybase.max_persistent = -1

; Maximum number of links (persistent + non-persistent).  -1 means no limit.
sybase.max_links = -1

;sybase.interface_file = "/usr/sybase/interfaces"

; Minimum error severity to display.
sybase.min_error_severity = 10

; Minimum message severity to display.
sybase.min_message_severity = 10

; Compatability mode with old versions of PHP 3.0.
; If on, this will cause PHP to automatically assign types to results according
; to their Sybase type, instead of treating them all as strings.  This
; compatability mode will probably not stay around forever, so try applying
; whatever necessary changes to your code, and turn it off.
sybase.compatability_mode = Off

[Sybase-CT]
; Allow or prevent persistent links.
sybct.allow_persistent = On

; Maximum number of persistent links.  -1 means no limit.
sybct.max_persistent = -1

; Maximum number of links (persistent + non-persistent).  -1 means no limit.
sybct.max_links = -1

; Minimum server message severity to display.
sybct.min_server_severity = 10

; Minimum client message severity to display.
sybct.min_client_severity = 10

[bcmath]
; Number of decimal digits for all bcmath functions.
bcmath.scale = 0

[browscap]
;browscap = extra/browscap.ini

[Informix]
; Default host for ifx_connect() (doesn't apply in safe mode).
```

```
ifx.default_host =

; Default user for ifx_connect() (doesn't apply in safe mode).
ifx.default_user =

; Default password for ifx_connect() (doesn't apply in safe mode).
ifx.default_password =

; Allow or prevent persistent links.
ifx.allow_persistent = On

; Maximum number of persistent links.  -1 means no limit.
ifx.max_persistent = -1

; Maximum number of links (persistent + non-persistent).  -1 means no limit.
ifx.max_links = -1

; If on, select statements return the contents of a text blob instead of its id.
ifx.textasvarchar = 0

; If on, select statements return the contents of a byte blob instead of its id.
ifx.byteasvarchar = 0

; Trailing blanks are stripped from fixed-length char columns.  May help the
; life of Informix SE users.
ifx.charasvarchar = 0

; If on, the contents of text and byte blobs are dumped to a file instead of
; keeping them in memory.
ifx.blobinfile = 0

; NULL's are returned as empty strings, unless this is set to 1.  In that case,
; NULL's are returned as string 'NULL'.
ifx.nullformat = 0

[Session]
; Handler used to store/retrieve data.
session.save_handler = files

; Argument passed to save_handler.  In the case of files, this is the path
; where data files are stored. Note: Windows users have to change this
; variable in order to use PHP's session functions.
session.save_path = /tmp

; Whether to use cookies.
```

```
session.use_cookies = 1


; Name of the session (used as cookie name).
session.name = PHPSESSID

; Initialize session on request startup.
session.auto_start = 0

; Lifetime in seconds of cookie or, if 0, until browser is restarted.
session.cookie_lifetime = 0

; The path for which the cookie is valid.
session.cookie_path = /

; The domain for which the cookie is valid.
session.cookie_domain =

; Handler used to serialize data.  php is the standard serializer of PHP.
session.serialize_handler = php

; Percentual probability that the 'garbage collection' process is started
; on every session initialization.
session.gc_probability = 1

; After this number of seconds, stored data will be seen as 'garbage' and
; cleaned up by the garbage collection process.
session.gc_maxlifetime = 1440

; Check HTTP Referer to invalidate externally stored URLs containing ids.
session.referer_check =

; How many bytes to read from the file.
session.entropy_length = 0

; Specified here to create the session id.
session.entropy_file =

;session.entropy_length = 16

;session.entropy_file = /dev/urandom

; Set to {nocache,private,public} to determine HTTP caching aspects.
session.cache_limiter = nocache
```

```
; Document expires after n minutes.
session.cache_expire = 180

; use transient sid support if enabled by compiling with --enable-trans-sid.
session.use_trans_sid = 1

url_rewriter.tags = "a=href,area=href,frame=src,input=src,form=fakeentry"

[MSSQL]
; Allow or prevent persistent links.
mssql.allow_persistent = On

; Maximum number of persistent links.  -1 means no limit.
mssql.max_persistent = -1

; Maximum number of links (persistent+non persistent).  -1 means no limit.
mssql.max_links = -1

; Minimum error severity to display.
mssql.min_error_severity = 10

; Minimum message severity to display.
mssql.min_message_severity = 10

; Compatability mode with old versions of PHP 3.0.
mssql.compatability_mode = Off

; Valid range 0 - 2147483647.  Default = 4096.
;mssql.textlimit = 4096

; Valid range 0 - 2147483647.  Default = 4096.
;mssql.textsize = 4096

; Limits the number of records in each batch.  0 = all records in one batch.
;mssql.batchsize = 0

[Assertion]
; Assert(expr); active by default.
;assert.active = On

; Issue a PHP warning for each failed assertion.
;assert.warning = On

; Don't bail out by default.
;assert.bail = Off
```

```
; User-function to be called if an assertion fails.
;assert.callback = 0

; Eval the expression with current error_reporting().  Set to true if you want
; error_reporting(0) around the eval().
;assert.quiet_eval = 0

[Ingres II]
; Allow or prevent persistent links.
ingres.allow_persistent = On

; Maximum number of persistent links.  -1 means no limit.
ingres.max_persistent = -1

; Maximum number of links, including persistents.  -1 means no limit.
ingres.max_links = -1

; Default database (format: [node_id::]dbname[/srv_class]).
ingres.default_database =

; Default user.
ingres.default_user =

; Default password.
ingres.default_password =

[Verisign Payflow Pro]
; Default Payflow Pro server.
pfpro.defaulthost = "test-payflow.verisign.com"

; Default port to connect to.
pfpro.defaultport = 443

; Default timeout in seconds.
pfpro.defaulttimeout = 30

; Default proxy IP address (if required).
;pfpro.proxyaddress =

; Default proxy port.
;pfpro.proxyport =

; Default proxy logon.
;pfpro.proxylogon =
```

```
; Default proxy password.
;pfpro.proxypassword =

[Sockets]
; Use the system read() function instead of the php_read() wrapper.
sockets.use_system_read = On

[com]
; path to a file containing GUIDs, IIDs or filenames of files with TypeLibs
;com.typelib_file =
; allow Distributed-COM calls
;com.allow_dcom = true
; autoregister constants of a components typlib on com_load()
;com.autoregister_typelib = true
; register constants casesensitive
;com.autoregister_casesensitive = false
; show warnings on duplicate constat registrations
;com.autoregister_verbose = true

[Printer]
;printer.default_printer = ""

[mbstring]
;mbstring.internal_encoding = EUC-JP
;mbstring.http_input = auto
;mbstring.http_output = SJIS
;mbstring.detect_order = auto
;mbstring.substitute_character = none;

[FrontBase]
;fbsql.allow_persistant = On
;fbsql.autocommit = On
;fbsql.default_database =
;fbsql.default_database_password =
;fbsql.default_host =
;fbsql.default_password =
;fbsql.default_user = "_SYSTEM"
;fbsql.generate_warnings = Off
;fbsql.max_connections = 128
;fbsql.max_links = 128
;fbsql.max_persistent = -1
;fbsql.max_results = 128
;fbsql.mbatchSize = 1000
```

; Local Variables:
; tab-width: 4
; End:

7. REGISTRO.PHP

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"
          "http://www.w3.org/TR/REC-html40/strict.dtd">
<HTML>
<BODY bgcolor='#FFFFFF'>

<?php
$tmp_act = time();
include ("/var/www/htdocs/lenguaje/tpos.php");
 if ($tmp_act > TMP_INI and $tmp_act < TMP_FIN )
{
          $metodo = $GLOBALS["REQUEST_METHOD"];
}
else
{
          $metodo = 'PAUSA';
}
       $metodo = 'PAUSA';    // pausado
if ($metodo=='POST' and !$id)        {

include ("/var/www/htdocs/lenguaje/comun.php");
include ("/var/www/htdocs/lenguaje/entrada.php");

$error = formato_correcto($num_cta, $fech);

if ($error) {
   formulario_entrada($PHP_SELF, '', $error);
   echo "</BODY> </HTML>";
   exit;
}

$conex= pg_connect("host=$host_db port=$port_db dbname=$base user=$usuario");

if (!$conex) {
  include "pag_encabezado.html";
   echo "<center><p> <font color='#666666' size='3'>
     <b>En este momento existen problemas
     con la conexi&#243;n en la Base de Datos, por
     favor intenta m&#225;s tarde.</b> <br><br></center>
     <div align='center'><img src='../imagenesNvo/linea.png' width='469' height='5'>
     </div>
```

```
  </BODY> </HTML>";
  exit;
}

$nom_comp = obten_nombre_x ($conex, $tabla01, $num_cta);
if ($nom_comp==") {
  $error = "<P align='center'><font size='2' >ERROR:</P><B>
      <P align='center'>El N&#250;mero de Cuenta $num_cta es incorrecto.</P>
        <p align='center'>El Número de cuenta debe estar formado por 9 digitos,
        sin el guión<br>
        Ej: si tu número es 9215329-7 debes teclear 092153297</p>";

  formulario_entrada($PHP_SELF, ", $error);
  rompe_con($conex);
}

if ($nom_comp[fech_nac]<>$fech) {
  $error = "<BR>NIP Incorrecto";
  formulario_entrada($PHP_SELF, $num_cta, $error);
  rompe_con($conex);
}

//        proceso de cambios de grupo
if (proceso == 'C')
{
 $sql = "select * from entro where num_cta='$num_cta'";
 $sql_id = pg_Exec($conex, $sql);
 $existe = pg_NumRows ($sql_id);
 if ($existe==0)      {
      $y_s = periodo;
      $yan = $y_s[0].$y_s[2].$y_s[2].$y_s[3];
      $sem = $y_s[4];
      include "pag_encabezado.html";

      echo "<center><p> <font color='#666666' size='3'><br> <br> <b>ERROR: No
puedes realizar
      cambios porque no te reinscribiste al semestre $yan-$sem.</b></center>";
      rompe_con ($conex);
 }
/*
      elseif ($existe == 0 and $existeA >= 1) {
      $y_s = periodo;
      $yan = $y_s[0].$y_s[2].$y_s[2].$y_s[3];
      $sem = $y_s[4];
      echo "<br> <br> <b> No puedes realizar cambios porque
```

```
        no te reinscribiste al semestre $yan-$sem <br>
        Para realizar el registro de tus asignaturas
        deberas pasar a la oficina de la secretaria academica  de servicios
        escolares  con el Lic. Arturo Astorga.<br></b>";
        rompe_con ($conex);
  }

*/
}

$nomb_est = $nom_comp["nombre"]." ".$nom_comp["paterno"]." ".
                    $nom_comp["materno"];

$id = inicializa_sesion($conex, $nomb_est, $num_cta);

$plan = planes_alumno ($conex, $num_cta, '');
$n_plan = count($plan);
if ($n_plan==0) {
   include "pag_encabezado.html";
   echo " <center> <font color='#666666' size='2'>
   <BR> <BR><B> ERROR. NO tienes
   derecho a Inscripci&#243;n.<BR><center>";
   echo "<center> <font color='#666666' size='2'>
   <br> NOMBRE DEL ALUMNO: <b>$nomb_est</b><BR>";
   carrera_motivo_noinscr($conex, $num_cta);
}
elseif ($n_plan>=2) {
   formulario_plan ($PHP_SELF, $nomb_est, $plan, $id);
}
else {
        formulario_materias_inicial($conex, $id, $num_cta, $plan);
}

}
elseif ($metodo=='POST' and $id)
{
include ("/var/www/htdocs/lenguaje/comun.php");
include ("/var/www/htdocs/lenguaje/entrada.php");
  if ($env_cve)
  {
        $conex=        pg_connect("host=$host_db        port=$port_db        dbname=$base
user=$usuario");
        $param_id = parametros_id ($conex, $id);
        $num_cta = $param_id["num_cta"];
```

```
$ip = $GLOBALS["REMOTE_ADDR"];
$error = valida_sesion ($param_id);

if ($error)
{  rompe_con($conex);          }


$nom_comp = obten_nombre_x ($conex, $tabla01, $num_cta);
$nomb_est = $nom_comp["nombre"]." ".$nom_comp["paterno"].
          " ".$nom_comp["materno"];
if (!$cve)
{
  $planes = planes_alumno ($conex, $num_cta, '');
  formulario_plan ($PHP_SELF, $nomb_est, $planes, $id);
}
else
{
  $planes = planes_alumno ($conex, $num_cta, $cve);
  $valid = plan_correcto($planes, $cve);
  if ($valid == 'OK')
  {
        formulario_materias_inicial($conex, $id, $num_cta, $planes);
  }
  else
        {
              rompe_con($conex);
        }
      }
      $descone = pg_Close($conex);
  }
}
else
{
  if ($metodo == 'PAUSA')
  {
        mensaje_de_pausa_en_sistema();
  }
  else
  {
        formulario_entrada($PHP_SELF, '', '');
  }
}

?>
</BODY>
```

```
</HTML>

<?php


function formulario_entrada ($action, $num, $mensaje_error)
{
 include "pag_encabezado.html";
 echo "
 <FORM NAME='entrada' METHOD='post' ACTION='$action'>
  <TABLE NAME='tabla' border='1' cellspacing='0' cellpadding='7' align='center'>
    <tr><td align='right'> <font color='#666666'>
      <b>N&#218;MERO DE CUENTA: </b>     <td>
      <input type='text' name='num_cta' value='$num' tabindex=10
        size=9 maxlength=9 > <br>
    <tr><td align='right'> <font color='#666666'>
      <b>NIP: </b>                  <td>
      <input type='password' name='fech' size=8 maxlength=8 tabindex=20><br>
    <tr><td align='right'> <font color='#666666' size='2'>
       Formato del NIP: <td><font color='#666666' size='2'> ddmmaaaa<br> (fecha de
nacimiento)";
 echo "
      <INPUT type='hidden' name='okDatos' value='f'>
    <tr><td align='right' valign='top'><br>
     <INPUT type='submit' value='Enviar datos' name='enviar' tabindex=30>
    <td align='left' valign='top'><br>
      <INPUT type='reset' value='Limpiar' name='B2' tabindex=40></p>
</table>
   <br>                    <p                    align='center'>                    <font
color='#3975B4'><B><EM>$mensaje_error</EM></B></p> <br>
   <div align='center'><img src='../imagenesNvo/linea.png' width='469' height='5'> </div>
 </FORM>
     ";
}

function formulario_materias_inicial($conex, $id, $num_cta, $plan)
{
global $procesador, $nomb_est;
        $cve_plan = $plan[0]["cve_plan"];
        guarda_sesion_plan ($conex, $id, $cve_plan);

/*
        planes_out ES UNA CONSTATE DEFINIDA EN entrada.php
        QUE CONTIENE LOS PLANES QUE DEBEN REGISTRAR MOVIMIENTOS
        EN SU COORDINACION
```

```
*/
        $de_mod = strpos(PLANES_OUT, $cve_plan);
        if ( ( $de_mod > 0)  and ( proceso <> 'E' ) )
        {  echo " <center><b>Para realizar tu tramite
            presentate en tu coordinacion </center></b><br>
                </body> </html>";
          $cerrar = cierra_sesion ($conex, $id, 'OUT');
          exit;
        }

        $por_fecha = strpos(PLANES_FECHA, $cve_plan);
        if (($por_fecha > 0) and (proceso <> 'E'))
        {       valida_fecha_plan($conex, $id, $cve_plan); }

        $por_generacion = strpos(PLANES_GENERACION, $cve_plan);
        if (($por_generacion > 0) and (proceso <> 'E'))
        {       valida_fecha_gen($conex, $id, $num_cta, $cve_plan);       }


        $carrera = $plan[0]["carrera"];
        if (proceso == "R")
        {
          $inscritas = mat_inscritas ($conex, $num_cta, $cve_plan);
          $num_inscr = count ($inscritas);
          if ($num_inscr == 0)
           {
        include "pag_encabezado.html";
           echo "<CENTER><p><font color='#666666' size='2'>NOMBRE DEL
ALUMNO: <B>$nomb_est</b></p>";
           echo "<p>CARRERA: $cve_plan <b> $carrera</b></p></font></CENTER>";
        formulario_materias ($procesador, $id);

        /*  if ( $cve_plan == '0319' )
          {
           echo "<center> A los alumnos que cursen las materias 1083,1270 y 1269 solo
tendran derecho a la practica de campo <br>
           del Dr. Alvaro Sánchez Crispin, si se inscriben en el grupo 0001 <br>";
          }
          elseif ($cve_plan == '0304')
          {
           echo "<center> Alumnos de tercer semestre, materia con division interna clave
1325: <br>
           1. El alumno se reinscribe una sola vez a la clave 1325, con profesor de 17 a 18
horas (prehispanicas)<br>
```

2. De forma interna (sin reinscripcion) se toma la otra parte con profesor de 19 a 20 (Medieval) o inversamente
3. Las materias de tercer semestre, las oficiales son 3, el alumno toma con esta materia (Mediaval) 4 en total";

```
            } */
             }
          else
             {
            $cerrada = cierra_sesion ($conex, $id, 'IMP');
               include ("/var/www/htdocs/lenguaje/plan_inc.php");
               imprime_final ($conex, $id, $num_cta, $cve_plan);
              }
          }
         elseif (proceso == "C")
         {
        $sql = "select * from entro where num_cta='$num_cta'";
        $sql_id = pg_Exec($conex, $sql);
        $existe = pg_NumRows ($sql_id);
        if ($existe >=1) {

         include "pag_encabezado.html";

         echo "<center><p> <font color='#666666' size='2'>NOMBRE DEL ALUMNO:
<B>$nomb_est</b></p>
            <p>CARRERA: $cve_plan <b> $carrera</b></p> </font></center>";
            verifica_ingresos($conex, $num_cta, $cve_plan, $id);
            imprime_inscritas ($conex, $num_cta, $cve_plan);
            formulario_cambios ($procesador, $id);
        rompe_con ($conex);
        }
        }

        elseif (proceso == "E")
        {
        include "pag_encabezado.html";
            echo "<center><p> <font color='#666666' size='2'>NOMBRE DEL ALUMNO:
<B>$nomb_est</b></p>
                 <p>CARRERA: $cve_plan <b> $carrera</b></p> </center>";
            verifica_ingresos($conex, $num_cta, $cve_plan, $id);
            $inscritas = mat_inscritas ($conex, $num_cta, $cve_plan);
            $num_inscr = count ($inscritas);
            if ($num_inscr > 0)
            { imprime_inscritas ($conex, $num_cta, $cve_plan); }
            if ($cve_plan == '0833')
            {     leyenda_hist_99 ();    }
```

```
                formulario_mat_extra($procesador, $id);
            }
}

function mensaje_de_pausa_en_sistema()
{
$hr_act = date ("H:i");
$minn = TMP_INI;
$fecha_hoy = date ("d/m/Y");
include "pag_encabezado.html";
echo "<br><br>
<center><p> <font color='#666666' size='3'>
<b>$minn <br>
EN ESTE MOMENTO NO PUEDES REALIZAR TRAMITES<BR>
VERIFICA EL CALENDARIO DE SERVICIOS ESCOLARES.
<!--<br><br>HORA ACTUAL : $hr_act<br></b> </p> </center> -->
<br><br>HOY ES : $fecha_hoy<br></b> </p><br> <br></center>
<div align='center'><img src='../imagenesNvo/linea.png' width='469' height='5'>
</div>
";
}

function verifica_ingresos ($conex, $num, $cve, $id)
{
$sql = "select count (*) from sesion
          where num_cta='$num' and cve_plan='$cve' and
          hr_inicio >= '2004-22-03 09:00:00'  and
          causa_termino = 'FIN' "; //<> 'PRG' and causa_termino <> 'VIM'";
//        echo "$sql <br>";
  $sql_id = pg_Exec($conex, $sql);
  $num_tuplas = pg_fetch_array($sql_id, 0);
//        echo "NUMERO : $num_tuplas[0] <br>";
  if ($num_tuplas[0] > 5  )
  {
    ?>
          <br> <br> <br>
       <center><p> <font color='#666666' size='3'>
          <b>Sólo tienes derecho a ingresar tres veces al sistema
          de Registro.
          </b>
          </body> </html>
    <?php
   $cerrar = cierra_sesion ($conex, $id, 'NIN');
    exit;
```

```
  }
}
function valida_fecha_gen($conex, $id, $num, $cve)
{

 $t_a = time();
 $min = TMP_INI;
 $max = TMP_FIN;

 if ($t_a < TMP_RZG_INI )
 {
   $sqln = "select ingreso from alum_carr
            where num_cta='$num' and
                    cve_plan = '$cve'";
   $sqln_id = pg_Exec($conex, $sqln);
   $el_ingres = pg_fetch_array($sqln_id, 0);
   $el_ingr = $el_ingres[0];

 $pg = $cve.$el_ingr;

 $d1 = strpos(PLAN_G_D1, $pg);
 $d2 = strpos(PLAN_G_D2, $pg);
 $d3 = strpos(PLAN_G_D3, $pg);
 $d4 = strpos(PLAN_G_D4, $pg);
 $d5 = strpos(PLAN_G_D5, $pg);

 if ($d1 > 0) {
         $min = TMP_DIA_1_INI;
         $max = TMP_DIA_1_FIN;   }
 elseif ($d2 > 0)        {
         $min = TMP_DIA_2_INI;
         $max = TMP_DIA_2_FIN;   }
 elseif ($d3 > 0)        {
         $min = TMP_DIA_3_INI;
         $max = TMP_DIA_3_FIN;   }
 elseif ($d4 > 0)        {
         $min = TMP_DIA_4_INI;
         $max = TMP_DIA_4_FIN;   }
 elseif ($d5 > 0)        {
         $min = TMP_DIA_5_INI;
         $max = TMP_DIA_5_FIN;   }
 else
 {     $min = TMP_RZG_INI;
       $max = TMP_RZG_FIN;   }
```

```
      if ($t_a < $min or $t_a > $max)
//     if (($min == TMP_INI and $max == TMP_FIN) or  ( $d1 == null))
   {
     $inicio = date('d ', $min);
     $fin = date('d ', $max);
     echo "<center><p> <font color='#666666' size='3'><br> <br><br><b>
          ERROR: REVISA EN EL CALENDARIO DE INSCRIPCION<BR>
          QUE FECHA LE CORRESPONDE A TU CARRERA Y GENERACION.
          </b>
          </font></center><br> <br>
       <div        align='center'><img        src='../imagenesNvo/linea.png'        width='469'
height='5'></div> ";

          echo "</body> </html>";
          $sql = "UPDATE sesion SET hr_termino = $t_a,
                    causa_termino = 'PRG' where id_sesion='$id'";
          $sql_id = pg_Exec($conex, $sql);
          exit;
   }
 }
}

function valida_fecha_plan($conex, $id, $cve)
{
 $t_a = time();
 $min = TMP_INI;
 $max = TMP_FIN;
 echo "plan $cve <br> I:$tmp_ini F:$tmp_fin";


 $d1 = strpos(PLAN_F_D1, $cve);
 $d2 = strpos(PLAN_F_D2, $cve);
 $d3 = strpos(PLAN_F_D3, $cve);
 $d4 = strpos(PLAN_F_D4, $cve);
 $d5 = strpos(PLAN_F_D5, $cve);

 if ($d1 > 0) {
          $min = TMP_DIA_1_INI;
          $max = TMP_DIA_1_FIN;    }
 elseif ($d2 > 0)         {
          $min = TMP_DIA_2_INI;
          $max = TMP_DIA_2_FIN;    }
 elseif ($d3 > 0)         {
```

```
            $min = TMP_DIA_3_INI;
            $max = TMP_DIA_3_FIN;   }
 elseif ($d4 > 0)          {
            $min = TMP_DIA_4_INI;
            $max = TMP_DIA_4_FIN;   }
 elseif ($d5 > 0)          {
            $min = TMP_DIA_5_INI;
            $max = TMP_DIA_5_FIN;   }

  if ($t_a < $min or $t_a > $max)
  {
    $inicio = date('d ', $min);
    $fin = date('d ', $max);
    echo " <center><p> <font color='#666666' size='3'><br> <br><br><b>
            ERROR: REVISA EN EL CALENDARIO DE INSCRIPCION<BR>
            QUE FECHA LE CORRESPONDE A TU CARRERA .
            </b>
            </font></center><br><br>
        <div          align='center'><img          src='../imagenesNvo/linea.png'          width='469'
height='5'></div>";

            echo "</body> </html>";
            $sql = "UPDATE sesion SET hr_termino = $t_a,
                    causa_termino = 'PRG' where id_sesion='$id'";
            $sql_id = pg_Exec($conex, $sql);
            exit;
    }
}
?>
```

8. MOVIMIENTOS.PHP

```
<html>
<body bgcolor='#FFFFFF'>

<?php
$metodo = $GLOBALS["REQUEST_METHOD"];
if ($metodo=='POST' and $id)
{
include ("/var/www/htdocs/lenguaje/comun.php");
include ("/var/www/htdocs/lenguaje/materias.php");
include ("/var/www/htdocs/lenguaje/plan_inc.php");
$conex = pg_connect ("host=$host_db port=$port_db dbname=$base user=$usuario");

 if (!$conex)
 {
  echo "<p align='center'><font color='#3975B4' size='5'>Error en la conexi&#243;n con
la base de datos<br>";
  echo "</body></html>";
  exit;
 }

 $parametros_sesion = parametros_id($conex, $id);
 $sesion_inval = valida_sesion($parametros_sesion);

 if ($sesion_inval)
 {
  echo "$sesion_inval<br>";
  echo "</body></html>";
  exit;
 }

 $num_cta = $parametros_sesion["num_cta"];
 $cve_plan = $parametros_sesion["cve_plan"];

 if ($cancel)
 {
  $cancelado =cancela_inscripcion ($conex, $num_cta, $cve_plan);
        $cerrada = cierra_sesion ($conex, $id, 'CAN');
        echo "<p align='center'><font color='#3975B4' size='5'><b>La inscripci&#243;n
fue cancelada.</b><br> </p>
          </body> </html>";
        exit;
 }
```

```
if ($terminar)
{
        $cerrada = cierra_sesion ($conex, $id, 'FIN');
        imprime_final ($conex, $id, $num_cta, $cve_plan);
        exit;
}

if($confirmado)
{
 $sql = "select count(*) from inscr where num_cta='$num_cta' and
                cve_plan='$cve_plan' and cve_asign='$cve_asign' and
                cve_gpo='$cve_gpo'";
 $sql_id = pg_Exec ($conex, $sql);
 $tupla = pg_Fetch_Array ($sql_id, 0);
 if ($tupla[0]==1)
 {
  $sql = "delete from inscr where num_cta='$num_cta' and
                cve_plan='$cve_plan' and cve_asign='$cve_asign' and
                cve_gpo='$cve_gpo'";
  $sql_id = pg_Exec ($conex, $sql);
  $movimiento = pg_cmdTuples($sql_id);
 }
 elseif ($tupla[0]==0)
  {
   echo "<center>
        <p><p align='center'><font color='#3975B4' size='5'>ERROR: No puede dar de
baja la
        asignatura <b>$cve_asign</b> grupo <b>$cve_gpo</b>.<br>
        No tiene inscripción en este grupo, verifica la clave. </p></font>
        </center>";
  }
 }

$tope = 7;
if($alta)
{
 if (proceso == 'E')
 {
 $sql = "select count(*) from inscr where num_cta='$num_cta'
                and cve_plan='$cve_plan'     and
                cve_gpo like 'EA%' ";
 }
 else
 {
```

```
$sql = "select count(*) from inscr where num_cta='$num_cta'
                    and cve_plan='$cve_plan'      and
                    cve_gpo <'EA' ";
}
$sql_id = pg_Exec($conex, $sql);
$num_insc = pg_Fetch_Array($sql_id, 0);

if (proceso == 'E')
{
 $verbo = 'registrar';
 $sql_perm = "select count (*) from avance_re where num_cta='$num_cta'
                    and cve_plan='$cve_plan'";
 $sql_id_perm = pg_Exec($conex, $sql_perm);
 $num_perm = pg_Fetch_Array($sql_id_perm, 0);
 $periodo = periodo;
 $sql_art19 = "select count (*) from alum_carr where num_cta='$num_cta'
                    and cve_plan='$cve_plan' and art_19_22 <= '$periodo'";
 $sql_id_art19 = pg_Exec($conex, $sql_art19);
 $num_art19 = pg_Fetch_Array($sql_id_art19, 0);
 if (($num_perm[0] == 1) or ($num_art19[0] == 1))
 {      $tope = 6;       }
 else
 {      $tope = 2;       }
}
else
{
 $verbo = 'inscribir';
 $pos = strpos(planes_8, $cve_plan);
 if ($pos > 0)
 {      $tope = 8;       }
 elseif ($cve_plan > '0305' and $cve_plan <> '0308')
 {      $tope = 7;       }
 elseif ($cve_plan =='0303' or $cve_plan =='0304')
 {      $tope = 4;       }
 elseif ($cve_plan == '0305')
 {      $tope = 4;       }
 elseif ($cve_plan == '0308')
 {      $tope = 5;       }
 elseif ($cve_plan == '1114')
 {   $tope = 6;      }
}

if ($num_insc[0] >= $tope)
{
 echo "<p align='center'><font color='#3975B4' size='5'>NOTA. S&#243;lo puedes
```

```
        $verbo $tope asignaturas.</p></font></center>";
   if (proceso == 'E' and cve_ext=='B')
   {
        echo "<p align='center'><font color='#3975B4' size='5'>Se consideran las
asignaturas del periodo 'EA'
                y 'EB'</center></p>";
   }
  }
else
{
  $materia = ya_seleccionada ($conex, $num_cta, $cve_plan, $cve_asign);
  if ($materia)
  {
    echo "<center>
        <p align='center'><font color='#3975B4' size='5'>ERROR: Ya est&#225;
registrada la
        asignatura <b>$cve_asign</b>.</p></font>
        </center>";
  }
  else
  {
/***   encaso de que se trate de un plan cerrado verificamos que la asignatura
        grupo seleccionado sea del plan del alumno          *****/
if (proceso == 'E')
{   $cve_gpo = proceso.cve_ext.$cve_gpo;    }

$es_cerrado = strpos(cerrados, $cve_plan);
$es_vjo_nvo = strpos(vjo_nvo, $cve_plan);
$es_opta_h  = es_mat_opta_hist($conex, $cve_asign, $cve_gpo, $cve_plan);
//$es_seriado = es_idioma($conex, $cve_asign, $num_cta, $cve_plan);
$asign_acreditada = asign_ya_acreditada($conex, $cve_asign, $num_cta);
if (proceso != 'E')
{
 $no_aprobada_en_ord = asign_no_aprobada($conex, $cve_asign, $num_cta);
}


$prop_comp = '';

if( $es_opta_h == 1 )
{
   $prop_comp = 'E';
}
elseif ( $es_seriado == 1 )
```

```
{
  $prop_comp = 'D';
}
elseif ( $asign_acreditada == 1 )
{
  $prop_comp = 'I';
}
elseif ( $no_aprobada_en_ord == 1 )
{
  $prop_comp = 'J';
}
elseif ($es_cerrado > 0)
{
      $plan_asign =
            planes_mat_escol_cerr($conex, $cve_plan, $cve_asign, $cve_gpo);
      if (!$plan_asign)
      {
        $prop_comp = 'H';
      }
      else
      {
        $prop_comp = 'P';
      }
}
elseif ($es_vjo_nvo > 0)
{
      $plan_asign = planes_mat_escol ($conex, $cve_asign, $cve_gpo);
      if ($plan_asign)
      {
       $propia = plan_correcto ($plan_asign, $cve_plan);
       if ($propia)
       {
        $prop_comp = 'P';
       }
       else
       {
        $carrs = carrera_mat_escol($conex, $cve_asign, $cve_gpo);
        $cve_carr = $carr_vn["$cve_plan"];
        $exist_carr = checa_carr($carrs, $cve_carr);
        if ($exist_carr)
        {
            $prop_comp = 'F';
        }
        else
        {
```

```
                $prop_comp = 'C';
            }
          }
        }
        else
        {
          $prop_comp = 'G';
        }
}
else
{
        $plan_asign = planes_mat_escol ($conex, $cve_asign, $cve_gpo);
        if (!$plan_asign)
        {
           $prop_comp = 'G';
        }
        else
        {
          $propia = plan_correcto ($plan_asign, $cve_plan);
          if ($propia)
          {
            $prop_comp = 'P';
          }
          else
          {
            $prop_comp = 'C';
          }
        }

    }

    if ($prop_comp =='F')
    {
          echo "<p align='center'><font color='#3975B4' size='5'>ERROR. La asignatura
<b>$cve_asign</b>
           pertenece a otro Plan de tu misma Carrera.</p></font><br>";
    }
    elseif($prop_comp =='G')
     {
       echo "<p align='center'><font color='#3975B4' size='5'>ERROR. La asignatura
<b>$cve_asign</b>
          grupo <b>$cve_gpo</b> no existe en horarios, verifica la clave.<br> </p>
          </font><br>";
     }
     elseif($prop_comp =='H')
```

```
    {
     echo "<p align='center'><font color='#3975B4' size='5'>ERROR. La asignatura
<b>$cve_asign</b>
        no pertenece a tu Plan.</p></font><br>";
    }
    elseif ($prop_comp =='D')
    {
     echo "<p align='center'><font color='#3975B4' size='5'>ERROR. Asignatura seriada,
no tienes calificación
        previa. Acude a la Coordinación de Modernas.
    </p></font><br>";
    }
    elseif ($prop_comp =='E')
    {
     echo "<p align='center'><font color='#3975B4' size='5'>ERROR. No puedes registrar
     asignaturas optativas de Historia Plan 1999, tampoco podrás registrar asignaturas
     de otro Colegio si eres alumno de Historia. Acude a tu Coordinación.
    </p></font><br>";
     }
    elseif ($prop_comp =='I')
     {
     echo "<p align='center'><font color='#3975B4' size='5'>ERROR. No puedes volver
registrar
     asignaturas que ya aprobaste en periodos pasados.
     </p></font><br>";
    }
    elseif ($prop_comp =='J')
    {
     echo "<p align='center'><font color='#3975B4' size='5'>ERROR. Art 33.-
Ning&uacute;n alumno podr&aacute;
        ser inscrito m&aacute;s de 2 veces
        en una misma asignatura. En caso de no acreditarla, solo podr&aacute; hacerlo
en examen extraordinario
        de acuerdo con lo dispuesto en el cap&iacute;tulo III del reglamento general de
examenes.
        </p></font><br>";
    }

    elseif ($prop_comp =='C' OR $prop_comp == 'P')
    {
        $sql = "select
        asign_gpo_valido('$cve_asign', '$cve_gpo', '$num_cta', '$cve_plan')";
        $sql_id = pg_Exec ($conex, $sql);
        $tupla = pg_Fetch_Array ($sql_id, 0);
        if ($tupla["asign_gpo_valido"]=='t')
```

```
        {
          $guardado = guarda_inscr ($conex, $num_cta, $cve_plan, $cve_asign,
                                     $cve_gpo, $prop_comp);
          if ($guardado == 0)
          {
                echo "<p align='center'><font color='#3975B4' size='5'>ERROR. No hay
cupo en esta
                asignatura.</p></font><br>";
          }
        }
        else
        {
          echo "<p align='center'><font color='#3975B4' size='5'>ERROR. La asignatura
<b>$cve_asign</b>
                grupo <b>$cve_gpo</b> no cumple con los requisitos
                establecidos por la Coordinaci&#243;n.
                <br></p></font>";
        }
    }
  }
}               //      cierre del else  de 7 inscritas
}               //      cierre del if alta

$nom_comp = obten_nombre_x ($conex, 'alumnos', $num_cta);
$nombre         =       $nom_comp["nombre"]."           ".$nom_comp["paterno"]."
".$nom_comp["materno"];
include "pag_encabezado.html";

echo "<center><p> <font color='#666666' size='2'>NOMBRE: <b>$nombre</b></p>";
echo "<p>CARRERA: $cve_plan <B>".$plan_carr["$cve_plan"]."</b> </p>";
echo "</font></center>";

$num_insc = imprime_inscritas ($conex, $num_cta, $cve_plan);
if ($num_insc >= $tope)       { $max_perm = 'YA'; }

if ($opcion)
{
echo "
     <center><p> <font color='#666666' size='2'>
         Captura en los recuadros las claves de la asignatura y
         grupo que vas a dar de <i>$opcion</i> y después presiona
         el botón.<br>
       En caso de extraordinario, si es alta solo teclea
      la parte númerica de grupo, si es baja teclea los 4 digitos </font></p></center>";
formulario_ok($id, $opcion);
```

```
}
else
{
        formulario_cambios ($PHP_SELF, $id, '');
//      formulario_mat_extra ($procesador, $id);
//      formulario_materias ($procesador, $id, $max_perm);
}

//      ************************************************      //
}

?>
</body>
</html>

<?php

function formulario_ok($id, $indicador)
{
if ($indicador[0] == 'A')
{       $nom_bot='alta';        }
elseif ($indicador[0] == 'B')
{       $nom_bot = 'confirmado';        }
echo"   <center>
 <FORM METHOD='post' ACTION='movimientos.php'>
<table border='0' cellspacing='0' cellpadding='7'>
<tr><td align='right'><font color='#666666' size='2'>
  <b>CLAVE DE LA ASIGNATURA: </b></font> </td> <td>
  <input type='text' name='cve_asign' tabindex=10 size=4
maxlength=4><br>
</td>
<tr><td align='right'><font color='#666666' size='2'>
  <b>CLAVE DEL GRUPO: </b> </font> </td> <td>
  <input type='text' name='cve_gpo' size=4 maxlength=4 tabindex=20><br>
</td>
<td><br></td> <td><br></td> <td><br></td>
<td><INPUT type='submit' value='$indicador' name='$nom_bot'
tabindex=30></td></b></td>
</table>
  <br> <br>
  <input type='hidden' name=id value='$id'>
</center>
  </FORM>
";
}
```

9. COMUN.PHP

```
<?php

define ("periodo", "20042");
define ("proceso", "R");
define ("cve_ext", "U");
define ("escuela", "010");

 $host_db='132.248.123.69';
#$host_db='127.0.0.1';
$port_db='5432';
$base="escolar";
$usuario="jcarlos";
$procesador = "movimientos.php";

function obten_nombre_x ($id_conex, $tabla, $cve)
{
  if ($tabla == 'alumnos') {
        $reg_cve = 'num_cta';
        $otra_col = 'fech_nac';
  } elseif ($tabla == 'profesor') {
        $reg_cve = 'cve_prof';
        $otra_col = 'grado';
  }
  $sql = "select
          trim(nombre) as nombre,
          trim(paterno) as paterno,
          trim(materno) as materno,
          $otra_col
        from $tabla where $reg_cve='$cve'";
  $sql_id = pg_Exec($id_conex, $sql);
  $num_ocur = pg_NumRows($sql_id);
  if ($num_ocur == 0) {
        return '';
  } else {
        $arr = pg_fetch_array ($sql_id, 0);
        return $arr;
  }
}

function parametros_id ($conex, $num_id)
{
  $sql = "select * from sesion where id_sesion='$num_id'";
```

```
  $sql_id = pg_Exec($conex, $sql);
  $arr = pg_Fetch_Array($sql_id, 0);
  return $arr;
}

function plan_correcto ($planes, $selec)
{
 $num_tuplas = count($planes);
 for ($i=0 ; $i < $num_tuplas ; $i++)
 {
        if ($planes[$i][0] == $selec)
        {
          $plan = "OK";
        }
 }
 return $plan;
}

function valida_sesion ($param)
{
  if ($param["hr_termino"])
  {     $error = "sesion finalizada";  }
  $tmp_act = time();
  if (tmp_act < $param["hr_inicio"])
  {     $error = "hora inicio posterior";       }
  return $error;
}

function extrae_tuplas ($conex, $sql)
{
 $sql_id = pg_Exec($conex, $sql);
 $num_tup = pg_NumRows($sql_id);
 for ($i=0; $i<$num_tup ; $i++)
 {
  $arr[$i] = pg_Fetch_Array($sql_id, $i);
 }
 return $arr;
}

function mat_inscritas ($conex, $numero, $plan)
{
 if (proceso == 'E')
 {
 $tpo_ext = proceso.cve_ext;
 $sql = "select asignatura.cve_asign, cve_gpo, trim(asignatura) as asignatura,
```

```
                prop_comp
        from inscr, asignatura
        where ((num_cta ='$numero') and (cve_plan = '$plan')) and
                asignatura.cve_asign = inscr.cve_asign        and
                cve_gpo > '$tpo_ext'
        order by asignatura.cve_asign";

 }
 else
 {
 $sql = "select asignatura.cve_asign, cve_gpo, trim(asignatura) as asignatura,
                prop_comp
        from inscr, asignatura
        where ((num_cta ='$numero') and (cve_plan = '$plan')) and
                asignatura.cve_asign = inscr.cve_asign        and
                cve_gpo < 'EA'
        order by asignatura.cve_asign";
 }
 $arr = extrae_tuplas($conex, $sql);
 return $arr;
}

function imprime_deudores ($conex, $num_cta)
{
 $sql = "SELECT count(*) from bloqueado where
                cve_mov='00' and num_cta='$num_cta'";
 $arr = extrae_tuplas($conex, $sql);
 $num_avi = $arr[0]["count"];
 if ($num_avi > 0)
 {      echo "<center><font color='#3975B4' size='3'> <br>
        <b>ADEUDA LIBRO EN BIBLIOTECA ACUDA A ACLARAR SU
SITUACIÓN.
        PRIMER AVISO.</b> <BR> </Font></center>
        ";      }
}

function imprime_inscritas ($conex, $num_cta, $cve_plan)
{
$inscritas = mat_inscritas($conex, $num_cta, $cve_plan);
$num_insc = count($inscritas);
$tipo["P"] = '';
$tipo["C"] = '*';

        echo "<center>
            <table border='1' cellspacing='0' cellpadding='5' width='90%'>
```

```
        <tr bgcolor='#3975B4'><td align='center'><font size='-1'
color='#FFFFFF'>CLAVE</td>
          <td align='center'><font size='-1' color='#FFFFFF'>NOMBRE DE LA
ASIGNATURA</td>
          <td align='center'><font size='-1' color='#FFFFFF'>GRUPO</td>
              ";
        $opta = 'NO';
        for ($i=0; $i<$num_insc; $i++)
        {
          echo "<tr><td align='center'><font size='-1'
color='#666666'>".$inscritas[$i]["cve_asign"].
              "<td><font size='-1' color='#666666'>
<b>".$inscritas[$i]["asignatura"]."</b>".
              "<td align='center'><font size='-1'
color='#666666'>".$inscritas[$i]["cve_gpo"].
              $tipo[$inscritas[$i]["prop_comp"]]."
              ";
          if ($tipo[$inscritas[$i]["prop_comp"]] == '*')
          {   $opta = 'SI';   }
        }
        echo "</table>";
        if ($opta == 'SI')
        {
         echo "
         <table border='0' cellspacing='0' cellpadding='5' width='90%'>\r
         <tr><td><font size='-1' color='#666666'>
         * Esta asignatura se considera Optativa porque no
               pertenece a tu Plan de Estudios.
         </font> </table>
         ";
        }
        echo "</center> ";
        imprime_deudores ($conex, $num_cta);
        return $num_insc;
        }

        function formulario_cambios ($action, $ident_sesion)
        {
         echo "
        <center>
        <FORM METHOD='post' ACTION='$action'>
        <table border='0' cellspacing='10'><tr><td>
        <b><INPUT type='submit' value='B A J A' name='opcion' value='B'
        tabindex=10></td></b>
        </td> <td> <br> </td> <td> <br> </td> <td> <br> </td> <td>
```

```
        <b><INPUT type='submit' value='A L T A' name='opcion' value='A'
        tabindex=20></td></b>
        <!--<td><input type='submit' value='CANCELAR TODAS LAS ASIGNATURAS'
                name='cancel' tabindex=40></td>
        </td> -->
        </table>  <br> <br>
          <INPUT type='submit' value='T E R M I N A R' name='terminar' value='1'
                tabindex=30></td></b>
          <input type='hidden' name=id value='$ident_sesion'>
        </center>
          </FORM>
        <div  align='center'><img  src='../imagenesNvo/linea.png'  width='469'  height='5'>
</div>

        <center>
        <table border='0' cellspacing='0'>
        <br> <br>
        <tr><td><font face='Arial,Helvetica,sans-serif' size='2' color='#666666'>
        '<i><b>B A J A</b></i>' Para eliminar una asignatura de tu inscripcion.<br>
                        En  caso  de  extraordinario  teclea  la  clave  completa  del
grupo.<br>
        <font face='Arial,Helvetica,sans-serif' size='2' color='#666666'>
        '<i><b>A L T A</b></i>' Para añadir una asignatura a tu inscripción. <br>
                        En  caso  de  extraordinario  solo  teclea  la  parte  numerica  del
grupo<br>
        <!--<font face='Arial,Helvetica,sans-serif' size='2' color='#666666'>
        <i><b>'CANCELAR  TODAS  LAS  ASIGNATURAS</b></i>'Cancela  todo  el
proceso de inscripcion en cuyo caso podras
                                reiniciar el proceso.<br>
        -->
        <font face='Arial,Helvetica,sans-serif' size='2' color='#666666'>
        '<i><b>TERMINAR  </i></b>'  Para  salir  del  sistema  y  concluir  tu  tramite.
<br><br>
        </table>

        ";
        }

        /* {
          echo "
        <center>
        <table border='0' cellspacing='0'>
        <tr><td>Presiona el boton
        '<i><b>B A J A</b></i>' para eliminar o
        '<i><b>A L T A</b></i>' para <tr><td>añadir una materia a tu
```

```
inscripción.
</table>

  <FORM METHOD='post' ACTION='$action'>
<table border='0' cellspacing='10'><tr><td>
<b><INPUT type='submit' value='B A J A' name='opcion' value='B'
tabindex=10></td></b>
</td> <td> <br> </td> <td> <br> </td> <td> <br> </td> <td>
<b><INPUT type='submit' value='A L T A' name='opcion' value='A'
tabindex=20></td></b>
</td> </table>  <br> <br>
Para salir del sistema presiona: <b>
  <INPUT type='submit' value='T E R M I N A R' name='terminar' value='1'
        tabindex=30></td></b>
  <input type='hidden' name=id value='$ident_sesion'>
</center>
  </FORM>
  ";
}
*/

function formulario_mat_extra ($action, $ident_sesion)
{
 $prdo_ext = cve_ext;
  echo "
  <FORM METHOD='post' ACTION='$action'>
<center>
<table border='0' cellspacing='0' cellpadding='7'>
<tr><td align='right'>
<font color='#666666' size='3'>CLAVE DE LA ASIGNATURA: </font>       <td>
  <input type='text' name='cve_asign' tabindex=10 size=4 maxlength=4><br>
<tr><td align='right'>
 <font color='#666666' size='3'>CLAVE DEL GRUPO: </font>                    <td><font
color='#666666' size='3'>E$prdo_ext</font>
  <input type='text' name='cve_gpo' size=2 maxlength=2 tabindex=20><br>
  <td align='center'>
  <INPUT type='submit' value='ALTA' name='alta' tabindex=30></td>
</table>
  <br>
  <td><input type='submit' value='TERMINAR REGISTRO'
        name='terminar' tabindex=40></td>
  <br> <br>
<!--  <td><input type='submit' value='CANCELAR TODAS LAS ASIGNATURAS'
        name='cancel' tabindex=40></td> -->
  <input type='hidden' name=id value='$ident_sesion'>
```

```
</center>
  </FORM>
 ";
}

function leyenda_hist_99 ()
{
    echo "<p align='center'<font face='Arial,Helvetica,sans-serif' size='2' color='#3975B4'>
              <b>Solo podras registrar asignaturas obligatorias <br>
              y tendras que acudir a tu coordinacion para  <br>
              registrar las asignaturas optativas</font> </p>";
}

/*function leyenda_clas_97 ()                    // esta leyenda solo se activa en semestres
nones
{
  echo "<p><b>Examen intermedio de Lengua <br>
  Latín : 19 de agosto de 2003  12:00 hrs. Salón 123 <br>
  Griego: 25 de agosto de 2003  12:00 hrs. Salón 302.</b></p> ";
}
*/
function imprime_final ($conex, $id, $cta, $plan)
{
global $plan_carr;
$prdo_ext = cve_ext;
$sm = periodo ;
$semestre = $sm[0].$sm[1].$sm[2].$sm[3].'-'.$sm[4];
$numero =
$cta[0].$cta[1].$cta[2].$cta[3].$cta[4].$cta[5].$cta[6].$cta[7]."-".$cta[8];

if ($plan < '0309')
{       $sist = 'ABIERTO';    }
else
{       $sist = 'ESCOLARIZADO';   }

if (proceso == 'E')
{       $leyenda = 'COMPROBANTE DE REGISTRO A EXTRAORDINARIOS';
        $prdo = 'E'.$prdo_ext;          }
elseif (proceso == 'R')
{       $leyenda = 'COMPROBANTE PROVISIONAL DE INSCRIPCIÓN';
        $prdo ='ORDINARIO'; } //$sm[4];              }
else
{
$leyenda = 'COMPROBANTE PROVISIONAL DE INSCRIPCIÓN --  CAMBIOS';
        $prdo = 'ORDINARIO'; } //$sm[4];             }
```

```
$hora = date ("d-H");
$verificador = $hora[0].$cta[6].$cta[7].$hora[1].$plan[3].
            $hora[4].$plan[2].$hora[5].$plan[1].$id[0];

echo "<center>";
echo "<table border='0' cellspacing='0' cellpadding='0' width='700'>\r";
echo "<tr>    <td height='175' width='20%'>";

?>
<div align="center"><img src="/imagenesNvo/CA2H9ZOM.jpg" width="85"></div>
</td>
<td height="175" width="60%">
<div align="center"> <p><b><font face="Verdana, Arial, Helvetica,
sans-serif"          size="-1"          color="#666666">UNIVERSIDAD          NACIONAL
AUT&#211;NOMA DE M&#201;XICO
</font></b></p> </div>
<p align="center"> <font face="Verdana, Arial, Helvetica,
sans-serif"  size="-1"  color="#666666"><b>FACULTAD  DE  FILOSOF&#205;A  Y
LETRAS</b></font></p>
<p align="center"> <font face="Verdana, Arial, Helvetica,
sans-serif"  size="-1"  color="#666666"><b>SECRETAR&#205;A  ACAD&#201;MICA
DE SERVICIOS ESCOLARES
</b></font></p>
<p align="center"> <font face="Verdana, Arial, Helvetica,
sans-serif"          size="-1"          color="#666666"><b>DEPARTAMENTO          DE
SISTEMAS</b></font></p>
</td>
<td height="175" width="20%">
  <div align="center"><img src="/imagenesNvo/atenea-2.png" height="102" ></div>
</td>
</tr> </table>
<?php
 $nom_comp = obten_nombre_x ($conex, 'alumnos', $cta);
 $nombre = $nom_comp["nombre"]." ".$nom_comp["paterno"]." "
          .$nom_comp["materno"];

 echo "<table border='0' cellspacing='0' cellpadding='2' width='600'>\r";
 echo "<caption><font size='-1'>
          $leyenda
          </font></caption>
 <tr>
   <td height='30' align='LEFT'><font size='-1' color='#666666'>SEMESTRE:</td>
   <td height='30' colspan='2'><font size='-1' color='#666666'><b>$semestre</b></td>
   <td> <br> </td>
```

```
   <td height='30' align='LEFT' ><font size='-1' color='#666666'>PERIODO:</td>
   <td height='30' ><font size='-1' color='#666666'><b>$prdo</b></td>
<tr><td height='30' align='LEFT'><font size='-1'color='#666666'>NOMBRE :</td>
   <td colspan='3'><font size='-1' color='#666666'><b>$nombre</b></td>
   <td height='30' align='left' ><font size='-1' color='#666666'>No. DE CUENTA:</td>
   <td height='30'><font size='-1' color='#666666'><b>$numero</b></td>
<tr><td height='30' align='LEFT'><font size='-1' color='#666666'>CARRERA:</td>
   <td height='30'><font size='-1' color='#666666'><B>".$plan_carr["$plan"]."</b></td>
   <td height='30' align='right' ><font size='-1' color='#666666'>PLAN:</td>
   <td height='30'><font size='-1' color='#666666'><b>$plan</b></td>
   <td height='30' align='left'><font size='-1' color='#666666'>SISTEMA:</td>
   <td height='30'><font size='-1' color='#666666'><b>$sist</b></td>
       </table> </center> \r";
      echo "<br>";

 imprime_inscritas ($conex, $cta, $plan);
 echo "
 <center>
 <table border='0' cellspacing='0' cellpadding='5' width='90%'>\r
 <tr><td><font size='-1' color='#666666'><b>NOTA:</b>El registro no procederá si
     la asignatura solicitada está acreditada.</td></tr>
 <tr><td align='right'><font size='-1' color='#666666'>".date("d/m/Y - H:i")."
 <br>$verificador </font></table>";

 echo "<center> <font size='2' color='#666666'>
         FIN DEL PROCESO
     <br><br><br> ";

 echo '<form>
         <INPUT TYPE="BUTTON" NAME="IMP"
         VALUE="Imprimir comprobante" onClick="javascript:window.print()">
         <INPUT TYPE="BUTTON" NAME="CER"
         VALUE="Cerrar ventana" onClick="javascript:window.close()">
     ';


     echo "  </body> </html>";
}

function rompe_con ($con)
{
 $desconecta = pg_Close($con);
 echo "</BODY></HTML>";
 exit;
}
```

10. ENTRADA.PHP

```
<?php
/*      SECCION DE VARIABLES
        planes_out PLANES QUE SE DEBEN REINSCRIBIR EN COORDINACION
        planes_fecha PLANES QUE SE FILTRAN POR FECHA
        planes_generacion PLANES QUE SE FILTRAN POR GENERACION
*/

define ("PLANES_OUT", " ");
define ("PLANES_FECHA", " ");
define ("PLANES_GENERACION", " ");

define ("PLAN_F_D1", " ");
define ("PLAN_F_D2", " ");
define ("PLAN_F_D3", " ");
define ("PLAN_F_D4", " ");
define ("PLAN_F_D5", " ");

define ("PLAN_G_D1", " ");
define ("PLAN_G_D2", " ");
define ("PLAN_G_D3", " ");
define ("PLAN_G_D4", " ");
Define ("PLAN_G_D5", " ");

$tabla01="alumnos";

/*      FUNCIONES DEFINIDAS
*/

function imprime_plan ($arr, $cve)
{
  $num_tuplas = count($arr);
  for ($i=0 ; $i < $num_tuplas ; $i++)
  {
        if ($arr[$i][0] == $cve)
        {
          echo "<font color='#0000CC'><p>CARRERA: </font>" . $arr[$i][0] .
          " <b>" . $arr[$i][1] . "</b></p>";
        }
  }
}

function planes_alumno ($conex, $numero, $cve)
```

```
{
echo "$cve <br>";
if ($cve > '0')
{       $linea = "alum_carr.cve_plan = '$cve'          and";   }
else
{       $linea = '';       }
   $sql = "select
                plan.cve_plan, carrera.carrera
           from
                plan, carrera, alum_carr, proceso
           where
                alum_carr.num_cta = '$numero'                  and
                status = 'A' and
                $linea
                plan.cve_plan = alum_carr.cve_plan and
                (plan.vigente = '4' or plan.vigente = '2') and
                (alum_carr.cve_exalum is NULL or alum_carr.cve_exalum = '20' or
alum_carr.cve_exalum = '11') and
                proceso.encurso = true and
                (alum_carr.art_19_22 > proceso.periodo      or
                proceso.proceso = 'E')and
                carrera.cve_siae = plan.cve_siae
        ;";
   $i_resultado = pg_Exec($conex, $sql);
   $tuplas = pg_NumRows($i_resultado);
   for ($i = 0 ;  $i < $tuplas ; $i++)
   {
        $arreglo[$i] = pg_Fetch_Array ($i_resultado, $i);
   }
   return $arreglo;
}


function formulario_plan ($action, $nombre, $planes, $num_id)
{
   $tuplas = count($planes);

   echo "NOMBRE DEL ALUMNO: <b>$nombre</b>";
   echo "<FORM METHOD='post' ACTION='$action'><SELECT size='1' name='cve'>";
          echo '<OPTION value="">SELECIONE UNA CARRERA</OPTION>';

          for ($i=0; $i < $tuplas; $i++)       {
               echo "<OPTION value='".$planes[$i][0]."'>"
                        .$planes[$i][0]." ".$planes[$i][1]."</OPTION>";
          }
```

```
                echo "</SELECT><br>";
                echo "<input type='hidden' name=id value='$num_id'>";
                echo '<p> <INPUT type="submit" value="Enviar datos" name="env_cve">
                      <input type="reset" value="Limpiar" name="B2"></p>';
                echo '</FORM>';
}

function formato_correcto ($numero_cta, $fecha)
{
  $numero_cta = trim($numero_cta);
  $fecha = trim($fecha);
  $numero_cta = ltrim($numero_cta);
  $fecha = ltrim($fecha);
  $mensaje_error ='';
  if (($numero_cta=='') or ($fecha==''))
  {
    $mensaje_error = 'ERROR. Faltan Datos.<br> Debe introducir tanto
     N&#250;mero de Cuenta como NIP.';
  }
  else
  {
    $tipo_correcto = ((9 ==  strlen($numero_cta)) and (8 == strlen($fecha)));
    $test_num = $numero_cta;
    $test_num2 = $fecha;
    $tipo_num = settype($test_num, integer);
    $tipo_num2 = settype($test_num2, integer);
//        echo "$tipo_num $test_num $tipo_num2 $test_num2 <br>";
    if ((!$tipo_correcto) or ($test_num==0) or ($test_num2==0)) {
        $mensaje_error = "ERROR. En el N&#250;mero de Cuenta y/o en el NIP.";
    }
  }
  return $mensaje_error;
}

function carrera_motivo_noinscr ($id_conex, $numero)
{
  $sql = "
        select
                trim(carrera.carrera) || ' PLAN  ' || trim(plan.inic_vigen),
                plan.cve_sistema,
                alum_carr.cve_plan,
                alum_carr.cve_exalum,
                alum_carr.art_19_22,
                alum_carr.status,
                plan.vigente
```

```
        from
                carrera, plan, alum_carr
        where
                (alum_carr.num_cta='$numero') and
                (plan.cve_plan=alum_carr.cve_plan) and
                (carrera.cve_siae = plan.cve_siae);
";

 $i_resultado = pg_Exec($id_conex, $sql);
 $tuplas = pg_NumRows($i_resultado);

 for ($i=0; $i < $tuplas; $i++)        {
        echo "<br> CARRERA: <b>".pg_result($i_resultado, $i, 2)." ".
                pg_result($i_resultado, $i, 0)."</b>  ";
        $sis = pg_result($i_resultado, $i, 1);
        if ($sis == 'A')
        {       echo "(Sistema Abierto) <br>";        }
        else
        {       echo "(Sistema Escolarizado) <br>";}
        $causa_ex = pg_result($i_resultado, $i, 3);
        $art_19 = pg_result($i_resultado, $i, 4);
        $sit = pg_result($i_resultado, $i, 5);
        $vigencia =pg_result($i_resultado, $i, 6);
        if (($causa_ex != '20') and ($causa_ex))
        {
                $sql2 = "select exalumno from exalumno
                        where cve_exalum = '$causa_ex'";
                $sql2_id = pg_Exec($id_conex,$sql2);
                $causa = pg_result($sql2_id, 0, 0);
                $mensaje = "Causa de exalumno: $causa_ex $causa";
        }
        elseif (($art_19 <= periodo) and (proceso != 'E'))
        {       $mensaje = "Artículo 19";     }
        elseif ($sit == 'B')
        {       $mensaje = "Tercer aviso de dictámen";       }
        elseif ($sit == 'S')
        {       $mensaje = "Suspension temporal";  }
        elseif ($vigencia == '0' or $vigencia=='1' or $vigencia=='3')
        {      $mensaje = 'Plan no vigente';          }
        echo "<br>Motivo: <b>$mensaje</b><br>";
   }
}

function inicializa_sesion ($conex, $nombre, $numero) {
```

```
  $ident_sesion = md5(uniqid($nombre.$numero));
  $hora = time();
  $ip = $GLOBALS["REMOTE_ADDR"];
  //      echo "ID:$session_id <br> TIME: $hora<br>ip: $ip<br>";

//// asigna el contador correspondiente a la sesion a la que entro

  $sql = "select count(*) from sesion where num_cta='$numero'";
  $sql_id = pg_Exec($conex, $sql);
  $numero_sesiones = pg_Fetch_Row($sql_id, 0);
  $numero_sesiones[0]++;

//        cerrar sesiones anteriores que no habian sido cerradas
  $sql_cierr = "update sesion SET hr_termino = $hora,
                        causa_termino = 'UPD'
                  where num_cta = '$numero'  and
                        hr_termino is null      and
                        consec < $numero_sesiones[0]";
  $sql_id = pg_exec($conex, $sql_cierr);

//        ************************************
  $sql = "insert into sesion(num_cta, consec, id_sesion, hr_inicio, ip)
        values('$numero', $numero_sesiones[0], '$ident_sesion', $hora,
        '$ip')";
  //      echo "$sql <br>";
  $sql_id = pg_Exec($conex, $sql);

  return $ident_sesion;
}

function guarda_sesion_plan ($conex, $id_sesion, $plan)
{
  $sql = "update sesion SET cve_plan = '$plan'
        where id_sesion = '$id_sesion'";
  $sql_id = pg_Exec($conex, $sql);
  if (!$sql_id)
  {
        echo "ERROR. El Sistema no puede responder por el momento.<BR>";
  }
}

function datos_proceso ($conex)
{
  $sql = "select * from proceso where encurso=true";
  $sql_id = pg_Exec($conex, $sql);
```

```
  if (!$sql_id)
  {
      echo "ERROR. El Sistema no puede responder por el momento.<BR>";
         echo "</body> </html>";
         exit;
  }
  else
  {
         $arr = pg_Fetch_Array($sql_id, 0);
  }
  return $arr;
}

function cierra_sesion ($conex, $id_sesion, $causa)
{
 $hora = time();
 $sql = "update sesion
        set    hr_termino = $hora,
               causa_termino = '$causa'
        where  id_sesion = '$id_sesion'
       ";
 $sql_id = pg_Exec ($conex, $sql);
 $se_cerro = pg_cmdTuples ($sql_id);
 if ($se_cerro == 1)
 {    return true;    }
 else
 {    return false;   }
}

fu

nction mensaje_inscrito($claves)
{
 $numero = count($claves);
 echo "ERROR. Ya te inscribiste.<br>
         Cualquier modificaci&#243;n tendr&#225;s
      que realizarla en Semana de Cambios.<br>";
}

function formulario_materias ($action, $ident_sesion)
{
 echo "
 <FORM METHOD='post' ACTION='$action'>
<center>
<table border='0' cellspacing='0' cellpadding='7'>
```

```
<tr><td align='right'>
  <b><font face='Arial,Helvetica,sans-serif' size='2' color='#666666'>CLAVE DE LA
ASIGNATURA: </b></font> <td>
  <input type='text' name='cve_asign' tabindex=10 size=4 maxlength=4><br>
<tr><td align='right'>
  <b><font face='Arial,Helvetica,sans-serif' size='2' color='#666666'>CLAVE DEL
GRUPO: </b></font> <td>
  <input type='text' name='cve_gpo' size=4 maxlength=4 tabindex=20><br>
<tr> <td align='center'>
<table border='0' cellspacing='0' cellpadding='7'> <tr> <td>
  <INPUT type='submit' value='ALTA' name='alta' tabindex=30></td>
  <td> <br> </td>
</table>
</table>
  <input type='hidden' name=id value='$ident_sesion'>
</center>
  </FORM>
<div align='center'><img src='../imagenesNvo/linea.png' width='469' height='5'> </div>
  ";
}

?>
```

11. MATERIAS.PHP

```
<?php

define ("planes_8", " 0311 0326 0327 0328 0329 0837");
define ("cerrados",
 " 0326 0327 0328 0329 0321 0322 0834 0303 0304 0305 0306 0307 0308");
define ("vjo_nvo",
 " 0310 0560 0309 0836 0311 0837 0316 0833 0317 1104 0325 1114");
#    clasi    filos    hispa    histo    bibli    latin

$carr_vn['0309'] = '411';
$carr_vn['0310'] = '419';
$carr_vn['0311'] = '414';
$carr_vn['0316'] = '412';
$carr_vn['0317'] = '402';
$carr_vn['0325'] = '409';
$carr_vn['0560'] = '419';
$carr_vn['0836'] = '411';
$carr_vn['0833'] = '412';
$carr_vn['0837'] = '414';
$carr_vn['1104'] = '402';
$carr_vn['1114'] = '409';

function es_idioma($conex, $asgn ,$num ,$plan)
{
 $sql="select count(*)
        from materia
        where materia.cve_asign   ='$asgn'    and
            materia.tpo_sub     is not null and
          ( materia.cve_plan    ='0312'  or
            materia.cve_plan    ='0313'  or
               materia.cve_plan    ='0314'  or
            materia.cve_plan    ='0315'  )";


 $sql_id=pg_Exec ($conex, $sql);
 $tipo = pg_fetch_Array($sql_id,0);

 if ( $tipo[0] == 0 )
   return 2 ;
 else
 {
 $sql="select count(*)
```

```
          from seriac_mod,materia
         where seriac_mod.num_cta = '$num'     and
              materia.cve_asign  = '$asgn'    and
              seriac_mod.cve_plan= '$plan'    and
               ( materia.tpo_sub    = seriac_mod.tpo_sub  or
              materia.tpo_sub    < seriac_mod.tpo_sub ) "  ;

  $sql_id=pg_Exec ($conex, $sql);
  $tipo = pg_fetch_Array($sql_id,0);
  if ( $tipo[0] == 0 )
    return 1;      // no procede
  else
    return 2;
  }
}

function es_mat_opta_hist($conex, $asgn ,$gpo ,$plan)
{
  $sql="select count(*)
          from materia,grupo
        where (materia.cve_asign ='$asgn' and
              materia.cve_plan  ='0833'  and
              materia.tpo_cred  ='1'     and
              grupo.cve_asign   ='$asgn' and
              grupo.cve_gpo     ='$gpo'  and
              grupo.cve_coord   ='09' ) or
             ( $plan            ='0833'  and
                 grupo.cve_asign   ='$asgn' and
              grupo.cve_gpo     ='$gpo'  and
              grupo.cve_coord  !='09' )";

  $sql_id=pg_Exec ($conex, $sql);
  $tipo = pg_fetch_Array($sql_id,0);

  if ($tipo[0] > 0)
    return 1;   // optativa - no procede
  else
    return 2;

}


function guarda_inscr ($conex, $numero, $plan, $asign, $grupo, $prop_comp)
{
  $cupos = cupo_grupo ($conex, $asign, $grupo);
```

```
 if ($prop_comp == 'P')
 {      $tipo_cup = "cupo_prop";     }
 elseif ($prop_comp == 'C')
 {      $tipo_cup = "cupo_com";      }

 if (proceso == 'E')
 {      $ocupado = 0; }
 else
 {      $ocupado = ocupacion ($conex, $asign, $grupo, $prop_comp);      }

 if ($cupos["$tipo_cup"]>$ocupado)
 {
        $sql = "insert into inscr values
          ('$numero', '$plan', '$asign', '$grupo', '$prop_comp')";
        $sql_id = pg_Exec($conex, $sql);
        $registrado = pg_cmdTuples ($sql_id);
 }
 return $registrado;
}

function cupo_grupo ($conex, $asgn, $gpo)
{
 $sql = "select cupo_prop, cupo_com
          from grupo, curso
          where cve_asign='$asgn' and cve_gpo='$gpo'
                and curso.cve_curso = grupo.cve_curso";
 $sql_id = pg_Exec ($conex, $sql);
 $arr = pg_Fetch_Array($sql_id, 0);
 return $arr;
}

function ocupacion ($conex, $asgn, $gpo, $p_c)
{
 $sql = "select count(*) from inscr
          where cve_asign='$asgn' and
                cve_gpo = '$gpo'  and
                prop_comp = '$p_c'";
 $sql_id = pg_Exec ($conex, $sql);
 $arr = pg_Fetch_Array($sql_id, 0);
 return $arr[0];
}

function ya_seleccionada ($conex, $numero, $plan, $asig)
{
 if (proceso == 'E')
```

```
{
$tpo_ext = proceso.cve_ext;
$sql = "select * from inscr
        where  num_cta = '$numero' and
                cve_plan = '$plan'  and
                cve_asign = '$asig'     and
                cve_gpo > '$tpo_ext'
        ";
}
else
{
$sql = "select * from inscr
        where  num_cta = '$numero' and
                cve_plan = '$plan'  and
                cve_asign = '$asig'
        ";
}
$sql_id = pg_Exec($conex, $sql);
$num = pg_NumRows($sql_id);
if ($num == 1)
{
  $arr = pg_Fetch_Array ($sql_id, 0);
}
return $arr;
}

/* Esta funcion va a validar que el alumno ya no pueda inscribir
   una asignatura que ya tenga aprobada  en periodos anteriores
  (valida s/historia academica)
*/

function asign_ya_acreditada($conex, $asgn, $num )
{
$sql = "select count (*)
        from hist_acad
        where hist_acad.cve_asign = '$asgn' and
        hist_acad.num_cta = '$num' and
        hist_acad.cve_calif = calif.cve_calif and
        calif.aprob = 't' ";

$sql_id= pg_Exec ($conex, $sql);
$resql = pg_fetch_Array($sql_id,0);

if ($resql[0] >= 1)
    return 1;   // asignatura ya acredita - no procede
```

```
  else
     return 2;
}

/* Esta funcion va a validar que el alumno ya haya registrado la misma
   asignatura 2 o mas veces en periodo ordinario sin aprobarla (valida s/historia academica)
*/

function asign_no_aprobada ($conex, $asgn, $num )
{
$sql = "select count (*)
        from hist_acad
        where hist_acad.cve_asign = '$asgn' and
        hist_acad.num_cta = '$num' and
        hist_acad.tpo_exam   = 'O' and
        hist_acad.cve_calif = calif.cve_calif and
        calif.aprob = 'f'";

$sql_id= pg_Exec ($conex, $sql);
$resql = pg_fetch_Array($sql_id,0);
if ($resql[0] >= 2)
     return 1;   // asignatura registrada 2 o más veces y no aprobada - no procede
  else
     return 2;
}

function planes_mat_escol ($conex, $asign, $grupo)
{
  $sql ="select materia.cve_plan
            from   plan, materia, grupo
            where grupo.cve_gpo = '$grupo' and
                     grupo.cve_asign = '$asign' and
                     materia.cve_asign = grupo.cve_asign and
                     plan.cve_plan = materia.cve_plan and
                     plan.cve_sistema = 'E' and
                     (plan.vigente = '2' or
                     plan.vigente ='4')";
  $arr = extrae_tuplas($conex, $sql);
  return $arr;
}

function carrera_mat_escol ($conex, $asign, $grupo)
{
  $sql ="select distinct carrera.cve_siae
            from   carrera, plan, materia, grupo
```

```
            where grupo.cve_gpo = '$grupo' and
                  grupo.cve_asign = '$asign' and
                  materia.cve_asign = grupo.cve_asign and
                  plan.cve_plan = materia.cve_plan and
                  plan.cve_sistema = 'E' and
                  carrera.cve_siae = plan.cve_siae
           ";
  $arr = extrae_tuplas($conex, $sql);
  return $arr;
}

function planes_mat_escol_cerr ($conex, $plan, $asign, $grupo)
{
  $sql ="select materia.cve_plan
           from   plan, materia, grupo
           where grupo.cve_gpo = '$grupo' and
                  grupo.cve_asign = '$asign' and
                  materia.cve_asign = grupo.cve_asign and
                  materia.cve_plan = '$plan' and
                  plan.cve_plan = materia.cve_plan and
                  (plan.vigente = '4' or
                  plan.vigente ='2')";
//         echo "$sql <br>";
//                  plan.cve_sistema = 'E' and
  $arr = extrae_tuplas($conex, $sql);
  return $arr;
}

function checa_carr($carreras, $seleccion)
{
  $num_tuplas = count($carreras);
  for ($i = 0; $i < $num_tuplas; $i++)
  if ($carreras[$i][0] == $seleccion)
  {       $checa = "OK";                    }
  return $checa;
}

function cancela_inscripcion ($conex, $num_cta, $plan)
{
  if (proceso == 'E')
  {
  $tpo_ext = proceso.cve_ext;
  $sql ="delete
           from   inscr
           where num_cta='$num_cta'  and
```

```
                    cve_plan= '$plan'       and
                    cve_gpo >'$tpo_ext'
                    ";
  }
  else
  {
  $sql ="delete
            from   inscr
            where num_cta='$num_cta'  and
                    cve_plan= '$plan'       and
                    cve_gpo <'EA'           ";
  }
  $sql_id = pg_Exec ($conex, $sql);
  $borrados = pg_cmdTuples ($sql_id);
  if ($borrados > 0)
  {       return true;      }
  else
  {       return false;     }
}

function cierra_sesion ($conex, $id_sesion, $causa)
{
  $hora = time();
  $sql = "update sesion
             set      hr_termino = $hora,
                      causa_termino = '$causa'
             where id_sesion = '$id_sesion'
            ";
  $sql_id = pg_Exec ($conex, $sql);
  $se_cerro = pg_cmdTuples ($sql_id);
  if ($se_cerro == 1)
  {       return true;      }
  else
  {       return false;     }
}

function formulario_materias ($action, $ident_sesion, $maxim)
{
  echo "
  <FORM METHOD='post' ACTION='$action'>
<center>";
if (!$maxim)
{
echo"
<table border='0' cellspacing='0' cellpadding='7'>
```

```
<tr><td align='right'>
  <b>CLAVE DE LA ASIGNATURA: </b>       <td>
  <input type='text' name='cve_asign' tabindex=10 size=4 maxlength=4><br>
<tr><td align='right'>
  <b>CLAVE DEL GRUPO: </b>              <td>
  <input type='text' name='cve_gpo' size=4 maxlength=4 tabindex=20><br>
<tr> <td align='center'>";
}
echo "
<table border='0' cellspacing='0' cellpadding='7'> <tr> <td align='center'>";
if (!$maxim)
{
echo"
  <INPUT type='submit' value='Alta' name='alta' tabindex=30></td>
  <td><br></td>";
}
echo "
  <tr><td align='center'>
  <INPUT type='submit' value='Cancelar Inscripci&#243;n' name='cancel'
tabindex=40></td>
  <td align='center'>
  <INPUT type='submit' value='Terminar' name='terminar' tabindex=50></td>
</table> ";
if (!$maxim)
{
echo "
</table>";
}
echo "
  <input type='hidden' name=id value='$ident_sesion'>
</center>
Con base en el acuerdo del Consejo Técnico de fecha 3 de diciembre
de 2002 los grupos correspondientes a asignaturas optativas o
seminarios que no registren una inscripción minima de 3 alumnos,
 serán cancelados.
  </FORM>
  ";
}

?>
```

12. TPOS.PHP

```
<?php
/*
        TODAS LAS FECHAS 'TMP' SON timestamp
        TODAS LAS FECHAS 'FCH' SON 'año-mes-dia hora-min-seg'
    tmp_ini FECHA DE INICIO DEL PROCESO DE REGISTRO
    tmp_fin FECHA DE FIN DEL REGISTRO
    dia1_ini FECHA DE INICIO DEL DIA 1
    dia1_fin FECHA DEL FIN DEL DIA 1
*/

define  ("TMP_INI", 1079622000 ); //1079967600  22 de marzo  2004
define  ("TMP_FIN", 1083171600 ); //  29 de marzo 2004
define  ("FCH_INI", "2004-22-03  09:00:00");

/*define ("TMP_DIA_1_INI", TMP_INI);      //  16 febrero 09:00
define ("TMP_DIA_1_FIN", 1077028200 );  //  17 febrero 08:30
define ("TMP_DIA_2_INI", 1077030000 );  //  17 febrero 09:00
define ("TMP_DIA_2_FIN", 1077114600 );  //  18 febrero 08:30
define ("TMP_DIA_3_INI", 1077116400 );  //  18 febrero 09:00
define ("TMP_DIA_3_FIN", 1077201000 );  //  19 febrero 08:30
define ("TMP_DIA_4_INI", 1077202800 );  //  19 febrero 09:00
define ("TMP_DIA_4_FIN", 1077287400 );  //  20 febrero 08:30
define ("TMP_DIA_5_INI", 1077289200 );  //  20 febrero 09:00
define ("TMP_DIA_5_FIN", 1077324600 );  //  20 febrero 18:50
define ("TMP_RZG_INI",   1077325200 );  //  20 febrero 19:00 */

define ("TMP_RZG_FIN", TMP_FIN);        //  23 febrero 09:00
?>
```

13. PLAN_INC.PHP

```
<?php
$plan_carr["0303"] = "FILOSOFIA (SUA)";
$plan_carr["0304"] = "LENGUA Y LITERATURAS HISPANICAS (SUA)";
$plan_carr["0305"] = "LENGUA Y  LIT MODERNAS INGLESAS (SUA)";
$plan_carr["0306"] = "HISTORIA (SUA)";
$plan_carr["0307"] = "PEDAGOGIA (SUA)";
$plan_carr["0308"] = "GEOGRAFIA (SUA)";
$plan_carr["0309"] = "FILOSOFIA";
$plan_carr["0310"] = "LETRAS CLASICAS";
$plan_carr["0311"] = "LENGUA Y LITERATURAS HISPANICAS";
$plan_carr["0312"] = "LENGUA Y LIT MODERNAS ALEMANAS";
$plan_carr["0313"] = "LENGUA Y LIT MODERNAS ITALIANAS";
$plan_carr["0314"] = "LENGUA Y LIT MODERNAS FRANCESAS";
$plan_carr["0315"] = "LENGUA Y  LIT MODERNAS INGLESAS";
$plan_carr["0316"] = "HISTORIA";
$plan_carr["0317"] = "BIBLIOTECOLOGIA";
$plan_carr["0318"] = "PEDAGOGIA";
$plan_carr["0319"] = "GEOGRAFIA";
$plan_carr["0321"] = "LENGUA Y  LIT MODERNAS INGLESAS (SUA)";
$plan_carr["0322"] = "LENGUA Y  LIT MODERNAS INGLESAS (SUA)";
$plan_carr["0325"] = "ESTUDIOS LATINOAMERICANOS";
$plan_carr["0326"] = "LITERATURA DRAMATICA Y TEATRO";
$plan_carr["0327"] = "LITERATURA DRAMATICA Y TEATRO";
$plan_carr["0328"] = "LITERATURA DRAMATICA Y TEATRO";
$plan_carr["0329"] = "LITERATURA DRAMATICA Y TEATRO";
$plan_carr["0560"] = "LETRAS CLASICAS";
$plan_carr["0833"] = "HISTORIA";
$plan_carr["0834"] = "LENGUA Y  LIT MODERNAS INGLESAS (SUA)";
$plan_carr["0836"] = "FILOSOFIA";
$plan_carr["0837"] = "LENGUA Y LITERATURAS HISPANICAS";
$plan_carr["1104"] = "BIBLIOT. Y EST. DE LA INFORM.";
$plan_carr["1114"] = "ESTUDIOS LATINOAMERICANOS";
?>
```

BIBLIOGRAFÍA

Rasmus Lerdorf, Kevin Tatroe, and Peter MacIntyre. Programming PHP, Paperback, April 2006

Adam Trachtenberg and David Sklar. PHP Cookbook: Solutions and Examples for PHP Programmers Paperback, 2006.

David Powers. PHP Solutions: Dynamic Web Design Made Easy. Paperback, 2006

Neil Matthew and Richard Stones. Beginning Databases with PostgreSQL: From Novice to Professional, Second Edition (Beginning from Novice to Professional). Apress, 2005.

W. J. Gilmore and Robert H. Treat. Beginning PHP and PostgreSQL 8: From Novice to Professional (Beginning: From Novice to Professional). Apress, 2006.

Daniel J. Barrett. Linux Pocket Guide. O'reilly 2004

Tony Steidler-Dennison. Run Your Own Web Server Using Linux & Apache. Sitepoint 2005