UNIVERSIDAD NACIONAL AUTÓNOMA
DE MÉXICO

FACULTAD DE CONTADURÍA Y ADMINISTRACIÓN

APLICACIÓN DEL CONCEPTO CLIENTE / SERVIDOR EN UN SISTEMA DE CONTROL ESCOLAR EN LA UNAM.

SEMINARIO DE INVESTIGACIÓN INFORMÁTICA

QUE PARA OBTENER EL TÍTULO DE:

LICENCIADO EN INFORMÁTICA

PRESENTAN:

JULIO CÉSAR ROLDÁN CAMPOS

RUTH ELIZABETH ENRÍQUEZ MONTAUT

ASESOR DEL SEMINARIO:
M.A. LUIS EDUARDO LÓPEZ CASTRO

MÉXICO, D.F. 1995

ANÁLISIS Y DISEÑO
DE SISTEMAS

A la Universidad Nacional Autónoma de México:

Por ser la base de nuestra formación profesional y personal.

Al Mtro. Luis Eduardo López Castro:

Por sus conocimientos compartidos, la asesoría y apoyo en el desarrollo de este trabajo y por su valiosa amistad.

A la Dirección de Cómputo para la Administración Académica:

Por su apoyo y colaboración principalmente

A la Act. Ma. Teresa Molina Charpenel
Al Sr. Octavio López Chávez
Al Sr. Ricardo Teapila Cualeco

A la Sra. Esperanza Díaz Vargas

Por su apoyo y amistad invaluable

A mi padre

Por su amor, enseñanzas, apoyo y por lo importante que es para mí.

A Malena

Por su apoyo y amor que me han ayudado a superarme.

A José Luis

Mi compañero y amigo, por su amor, comprensión y apoyo para lograr mis metas.

A mi hermana

Por el entusiasmo y motivación que siempre he recibido para seguir adelante.

A Liliana

Por su presencia en la familia, alegría, cariño y ternura.

A mi abuelo

Por su apoyo, confianza, cariño y por creer en mí.

A mi abuela

Por su cariño y apoyo que me han ayudado siempre.

A mis amigos

Por brindarme momentos agradables y compartir experiencias. Especialmente a Julio por su entusiasmo y dedicación en la elaboración de este trabajo.

Ruth

A mis Padres:

Por su amor y apoyo, por estar conmigo en todo momento y por darme un ejemplo constante de superación.

A mi Abuela Leonor Campos:

Por haber vivido entregada a sus hijos y ser la base de nuestra familia.

A mis Hermanos:

Porque son una parte muy importante de mi vida y me motivan a ser mejor día a día.

A Ma. Eugenia Arista:

Por su amor, por compartir conmigo este esfuerzo, y por todas las cosas que hemos aprendido juntos.

A mis familiares y amigos:

Por contar siempre con su amistad, y por que de cada uno de ellos he aprendido algo valioso para mi. En especial a Ruth ya que gracias a ella ha sido posible la realización de este trabajo.

Julio

ÍNDICE

INTRODUCCIÓN

En la actualidad, los sistemas de información con que cuentan las organizaciones, deben adaptarse a la evolución tecnológica en que vivimos, de no hacerlo, caen en el riesgo de no cubrir sus objetivos eficientemente o, inclusive, desaparecer; por este motivo es de vital importancia conocer y, en su caso, aplicar las nuevas formas de desarrollar sistemas de cómputo para aprovechar las bondades que esto trae consigo.

El objetivo de esta tesis no es explicar cuales son los beneficios ni desarrollar un sistema de cómputo para la administración escolar de manera tradicional, sino por el contrario, presentar una propuesta de un sistema de esta índole utilizando la arquitectura cliente servidor, que sirva de modelo para la modernización de los procedimientos de control escolar.

La población estudiantil de las instituciones educativas demandan una importante cantidad de servicios académicos a la administración escolar de cada plantel, lo que trae como consecuencia que se tenga que agilizar y simplificar todos los trámites para poder brindar un servicio eficiente al alumnado.

Debido al crecimiento de la población estudiantil de la UNAM, es necesario, por un lado, automatizar las funciones de la administración escolar en algunas escuelas y facultades que hasta la fecha las realizan manualmente y, por otro, modernizar los sistemas ya existentes para que se aprovechen los avances tecnológicos.

Dado que en la UNAM no se ha desarrollado un sistema de administración escolar bajo la arquitectura cliente/servidor,

decidimos incluir una breve explicación de los conceptos fundamentales de bases de datos y de esta tecnología.

El presente trabajo está fundamentado en el desarrollo de un sistema para la ENEP Iztacala (ADESI).

Para realizar el análisis y diseño de ADESI utilizamos la metodología moderna estructurada de Yourdon. En la programación se utilizó Visual Basic como front-end (cliente) y Sybase como back-end (servidor). La elaboración de los reportes que genera el sistema fue a través del reporteador Report Writer (R & R).

Finalmente, el utilizar una arquitectura cliente/servidor permite obtener un tiempo de respuesta menor y mayor seguridad en los procesos que contempla ADESI.

CAPÍTULO I
MANEJADORES DE BASES DE DATOS

INTRODUCCIÓN

Un sistema de bases de datos nos permite tener un conjunto de datos o listas de información que son relevantes en nuestro trabajo o en nuestras vidas. También nos proporciona la capacidad de almacenar y mantener esa información en un solo lugar. Consisteen dos partes: el sistema manejador de bases de datos (DBMS) que es el programa que organiza y mantiene los datos, y la aplicación de la base de datos, la cual nos permite consultar, insertar, actualizar la información almacenada, etc.

Una arquitectura cliente/servidor incrementa el poder de procesamiento al separar el manejador de la base de datos de la aplicación. La aplicación se ejecuta en una o más estaciones de trabajo, que comunmente son PCs, y se comunican a través de una red con uno o más manejadores de bases de datos que se encuentran en otras computadoras.

CONCEPTOS GENERALES

Un **dato** es el elemento susceptible de una observación sin significado, es la unidad mínima de información.

Información es un conjunto ordenado de datos interrelacionados con el objeto de proporcionar un conocimiento para tomar decisiones.

Una **base de datos** puede definirse como una colección de datos interrelacionados y almacenados en conjunto sin redundancias innecesarias, cuya finalidad es la de servir a una aplicación o más de la mejor manera posible. Los datos son almacenados de tal manera que resulten completamente independientes de los programas de las aplicaciones.

Una **entidad** puede ser un objeto tangible o intangible que tiene propiedades o características propias que son llamadas **atributos.**

Una **relación** es la asociación de dos o más entidades y puede ser:

- Uno a uno. En una relación de este tipo un elemento en A está asociado sólo con un elemento en B, y un elemento en B está asociado sólo con un elemento en A.

- Uno a muchos. En una relación uno a muchos un elemento en A está asociado con varios elementos en B y un elemento en B puede estar asociado sólo con un elemento en A.

- Muchos a uno. En una relación muchos a uno un elemento en A está asociado sólo a un elemento en B y un elemento en B puede estar asociado a varios elementos en A.

- Muchos a muchos. En una relación muchos a muchos un elemento en A está asociado a varios elementos en B y un elemento en B está asociado a varios elementos en A.

Independencia de los datos

Se puede definir como *la capacidad de modificar una definición de esquema en un nivel sin afectar la definición del esquema en el nivel inmediato superior.* [KOR86]

Implica que los programas de aplicación y los datos sean completamente independientes.

La independencia de datos la podemos dividir en independencia física e independencia lógica.

Independencia física *es la habilidad de modificar el esquema físico sin causar ningún cambio en los programas de aplicación, las modificaciones en el esquema físico son necesarias cuando deseamos mejorar el desempeño del sistema.* [KOR86]

Independencia lógica *es la habilidad de modificar el esquema conceptual sin necesidad de modificar los programas de aplicación, las modificaciones al nivel conceptual son necesarias cuando la estructura lógica de la base de datos ha sido alterada.* [KOR86]

Integridad implica el uso de procedimientos para salvaguardar la base de datos de alteraciones no válidas o de su destrucción con datos inconsistentes.

La **llave primaria** identifica en forma única a cada renglón de la tabla. Una llave primaria no permite valores nulos y no puede tener valores duplicados.

La **llave foránea** es una columna de una tabla que es llave primaria en alguna otra. Indican asociaciones entre tablas.

¿Qué es un Manejador de Bases de Datos?

Un sistema manejador de bases de datos es una colección de archivos interrelacionados y un conjunto de programas que son utilizados por diferentes usuarios para accesar o modificar dichos archivos. El principal propósito de un manejador de bases de datos es proporcionar a los usuarios el acceso a los datos de manera eficiente y transparente, es decir completamente independiente a la forma de almacenamiento. [KOR86]

El manejador de bases de datos es el responsable de proporcionar, al usuario , los servicios necesarios para manipular los datos.

Asimismo, el manejador de bases de datos debe de proveer algún tipo de integridad de los datos para evitar que éstos sean corrompidos. Esta característica es muy importante en bases de datos multiusuarios, en las cuales uno o más usuarios pueden actualizar un dato al mismo tiempo, por lo que el manejador de base de datos debe verificar que sólo uno de estos cambios se lleve a cabo.

Una base de datos puede almacenar cualquier tipo de información pero para que sea útil, los datos deben ser almacenados de acuerdo a su dominio. Un dominio es una categoría y tipo de dato que es guardado en un campo en particular.

Un manejador de base de datos debe proveer los siguientes servicios:

- La definicón de datos.
- El mantenimiento de datos, en donde cada campo contiene información particular que describe al registro.
- La manipulación de datos permite al usuario insertar, modificar, borrar y ordenar datos.
- Despliegue de información, permite presentar al usuario los datos de diferentes maneras.
- La integridad de los datos provee diferentes métodos para evitar que la información sea corrompida.

MODELOS DE MANEJADORES DE BASES DE DATOS

Los sistemas manejadores de bases de datos pueden ser agrupados dentro de cuatro diferentes modelos: el Modelo Manejador de Archivos, el Modelo de Bases de Datos Jerárquicas, el Modelo de Bases de Datos Red y el Modelo de Bases de Datos Relacional. Cada modelo es una descripción conceptual de cómo funciona la base de datos. Describen, específicamente, cómo los datos son presentados al usuario y al programador y cómo se llevan a cabo los accesos. Asimismo, describen las relaciones entre las diferentes entidades.

Estos modelos no describen la manera de cómo los datos son almacenados en el disco. Estos detalles son responsabilidad de los diseñadores de manejadores de bases de datos.

Modelo Manejador de Archivos

En este modelo cada campo es almacenado secuencialmente en el disco en uno o varios archivos. Para realizar la búsqueda de un dato en particular, será necesario que se realice desde el principio del archivo hasta encontrar el registro deseado, por lo que se vuelve lenta la localización de la información.

Este modelo fue el primero en utilizarse para almacenar información en bases de datos computarizadas. La ventaja que tiene este modelo es su sencillez.

Dentro de las desventajas de este modelo tenemos que no existe ninguna indicación de las relaciones entre varios elementos dentro del almacenamiento secuencial. Asimismo, crea ciertos problemas a la integridad de los datos; los valores de los datos tienen que ser checados por la aplicación antes de ser almacenados en el disco. La misma base de datos puede ser accesada por diferentes aplicaciones, y cada una puede tener diferentes valores para un mismo campo. Las diferentes aplicaciones tienen que ser controladas manualmente para asegurar que la definición de cada campo sea la adecuada respecto al dominio.

La única manera de ordenar los datos es leyendo todo el archivo y reescribiéndolo en el nuevo orden. Esto puede ser resuelto con el uso de un archivo de índices, el cual contiene apuntadores a cada registro de la base de datos.

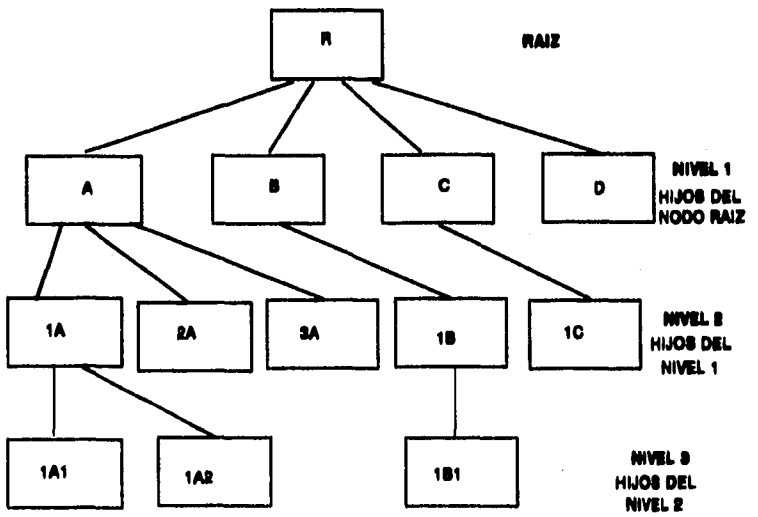
El sistema manejador de archivos no permite que se modifique fácilmente la estructura de la base de datos.

Modelo de Bases de Datos Jerárquicas

En este modelo los datos son organizados en una estructura de árbol, la cual comienza desde una raíz. Cada clase de dato puede ser localizado en diferentes niveles a partir de una rama cuyo origen es la raíz. En este tipo de estructura cada nivel es llamado nodo, en caso de que sea el último, es decir que dicha rama no tenga un nodo inferior es llamado hoja.

La estructura jerárquica permite que las búsquedas de datos sean más fáciles y rápidas. El manejador de base de datos no tiene que buscar en todo el archivo. Deberá examinar el requerimiento de

la búsqueda y realizarla a partir de la raíz hacia una rama y los niveles inferiores. Un índice puede ser utilizado en este modelo para incrementar la velocidad de las búsquedas y puede ser creado en una clase particular de datos (nivel).



Fuente: [SAL93]

En un modelo jerárquico existe sólo un nodo llamado raíz, los apuntadores descienden desde la raíz a los nodos del nivel 1, que es donde realmente empiezan los datos.

La estructura física de los datos en el disco es completamente independiente del modelo jerárquico.

Este modelo facilita el agregar nuevos campos en cualquier nivel de la estructura, el manejador de base de datos sólo tendrá que cambiar el apuntador final para que apunte al nodo que se desea insertar.

La relación entre padres e hijos sólo podrá ser modificada si se rediseña toda la estructura. Otro problema de este modelo tan rígido es que no es fácil cambiar la definición de los niveles (clases de datos). Asimismo, no facilita la definición de relaciones de muchos a muchos.

Modelo de Bases de Datos Red

El modelo de bases de datos red consiste en una colección de registros conectados entre sí a tavés de ligas. Cada registro es un conjunto de campos(atributos) y una liga es una asociación entre dos registros. [KOR86]

El nombre de red no tiene nada que ver con el medio físico en el cual actualmente corren las bases de datos. Este modelo describe conceptualmente las relaciones muchos a muchos. Las relaciones entre los diferentes tipos de datos son referidas como un solo grupo para distinguirlas de las relaciones tan estrictas de pardres-hijos definidas en el modelo jerárquico.

La flexibilidad del modelo red permite mostrar las relaciones de muchos a muchos. Estas interrelaciones entre los diferentes conjuntos pueden ser tan complejas que se dificulte su representación gráfica.

El diseño inicial de una base de datos bajo este modelo es fundamental, es decir que una vez que ha sido creado, cualquier cambio en algún grupo de datos implica realizar una nueva estructura.

Modelo de Bases de Datos Relacionales

En 1969, el Dr. E. F. Codd publicó el primer artículo en el que se define un modelo de bases de datos basado en el concepto matemático de relaciones entre conjuntos.

En este modelo los datos son organizados en conjuntos lógicos y matemáticos en una estructura tabular, donde cada campo representa una columna de la tabla y cada registro un renglón.

Podemos tener varias tablas dentro de las cuales, una columna puede tener un nombre que se repite en otra y son precisamente estas columnas repetidas las que nos permiten relacionar las diferentes tablas.

El modelo relacional es completamente flexible en la descripción de relaciones entre varios grupos de datos. Asimismo, el cambiar la estructura de una tabla es tan sencillo como agregar o borra una columna de la misma. Las tablas nuevas pueden crearse de tal manera que sean completamente independientes o bien como subconjuntos de las ya existentes.

La meta principal de este modelo es el preservar la integridad de los datos, por lo que se vuelve ideal para sistemas de procesamiento de transacciones, y para bases de datos con arquitectura cliente/servidor.

Un manejador de bases de datos relacional trata cada cambio (o conjunto de cambios) a los datos como una transacción, que es ejecutada en una copia temporal de la tabla que va a ser modificada.

Los cambios no son efectuados hasta que el usuario o la aplicación realiza el commit de la transacción. En este modelo, el manejador de bases de datos controla los posibles conflictos que surjan de modificar algún dato.

ARQUITECTURAS

El tipo de sistemas de cómputo, en el que las bases de datos corren, puede clasificarse en cuatro diferentes categorías o plataformas: centralizadas, ambientes multiusuarios, computadoras personales, cliente/servidor y sistemas distribuidos. La arquitectura del manejador de base de datos no determina necesariamente el tipo del sistema de cómputo en el que la base de datos tiene que correr, aun cuando algunas arquitecturas son más comunes para ciertas plataformas que otras.

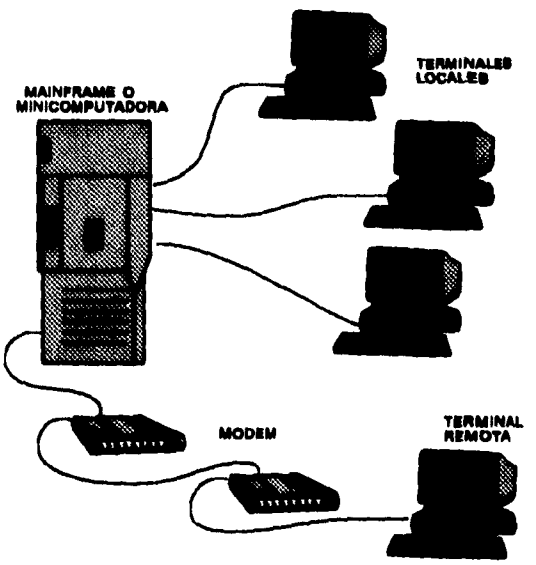
Plataformas Centralizadas

En un sistema centralizado todos los programas corren en un "host" o computadora principal, incluyendo el manejador de base de datos, las aplicaciones que accesan a la base de datos y el software de comunicaciones que envía y recibe datos de las terminales de los usuarios.

Las aplicaciones y el manejador de base de datos corren en el mismo host, comunicándose a través de áreas compartidas de memoria y de tareas que son administradas por el sistema operativo del host. El manejador es el responsable de mover los datos de o al disco o sistema de almacenamiento utilizando los recursos proveídos por el sistema operativo.

Las ventajas principales de un sistema centralizado son, en primer lugar que el manejo de la seguridad también se encuentra centralizado y, en segundo lugar la habilidad de tener grandes cantidades de datos en dispositivos de almacenamiento. Asimismo, esta plataforma permite que existan numerosos usuarios simultáneos.

Las desventajas generalmente están relacionadas con los costos elevados de actualización, de mantenimiento (pisos falsos, sistemas de enfriamiento por agua, sistemas de control de clima, etc.) y de un equipo de operadores y de programadores.



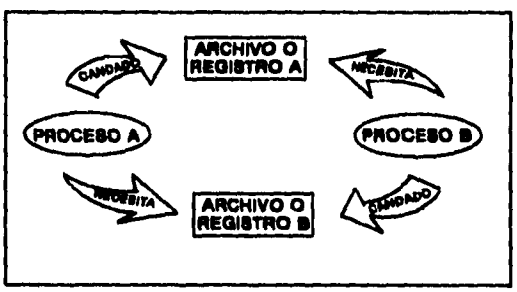
Fuente: [SAL93]

Ambientes Multiusuarios

Para los sistemas de bases de datos multiusuarios los manejadores proporcionan un balance entre dos objetivos: la concurrencia y la integridad de los datos. Asegurar la integridad implica el bloquear los registros y/o archivos, para prevenir que dos o más usuarios actualicen el mismo dato al mismo tiempo. Para maximizar la concurrencia es necesario mantener al mínimo el nivel de bloqueos.

Una aplicación multiusuario que lleva a cabo operaciones de bloqueo de registros y archivos podría causar un problema de estancamiento (deadlock).

Esto ocurre cuando un proceso A bloquea un registro o archivo y necesita un registro o archivo que ha sido bloqueado por el proceso B y el proceso B necesita el registro o archivo que fue bloqueado por el proceso A. Cuando esto pasa los procesos esperarán indefinidamente a menos que el estancamiento sea detectado y resuelto por el sistema.



Computadoras Personales

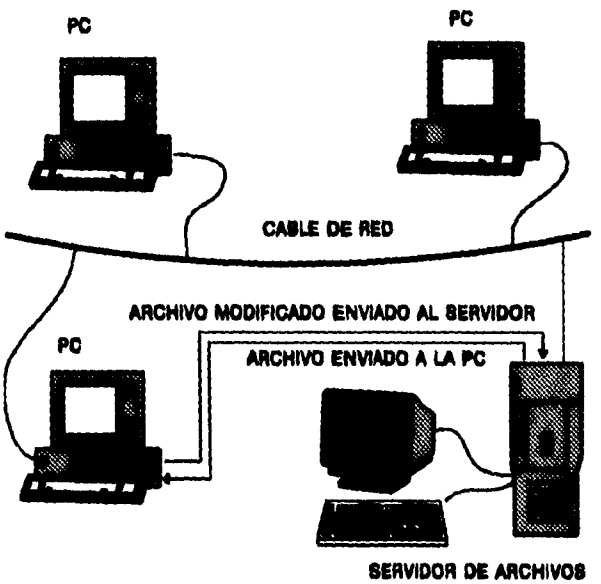
El manejador y las aplicaciones están combinadas formando un solo sistema. Los desarrollos con bases de datos en PCs continen las pantallas de entrada, salida y consulta de los datos. El combinar estas tres funciones en una sola proporciona mayor poder, flexibilidad y velocidad al manejador, además de que el costo disminuye.

En una red de área local los datos y las aplicaciones se encuentran generalmente en el servidor, el cual, contienen además el sistema operativo de red. El servidor de la red permite que se puedan compartir datos entre los usuarios y diversos recursos como son las impresoras.

Todos los procesos del manejador son realizados en la PC. El servidor sólo busca en disco los datos requeridos por el usuario, y los envía a través de la red a la PC del usuario. Para cualquier cambio que se realice a la información es necesario que la PC envíe de regreso al servidor todo el archivo de datos para que sea nuevamente almacenado en el disco. Cuando varios usuarios están accesando la base de datos, el servidor debe enviar a cada PC los datos que está solicitando lo que ocasiona un gran tráfico en la red disminuyendo así su velocidad.

Un problema de un sistema multiusuario es el de poder efectuar cambios simultáneos sobre un dato por diversos usuarios. Esto se soluciona usualmente agregando algún esquema de bloqueo de registros, en el que un registro o archivo es bloqueado cuando un usuario está modificándolo, impidiendo que otros usuarios puedan modificarlo al mismo tiempo.

Otro problema es el de realizar muchas transacciones simultáneas que incrementan el tráfico en la red, además de la limitante del poder de procesamiento de las PCs que afecta el desempeño del sistema. La solución que se ha desarrollado para estas limitantes es el uso de arquitecturas cliente/servidor.



Fuente: [SAL93]

Cliente/Servidor

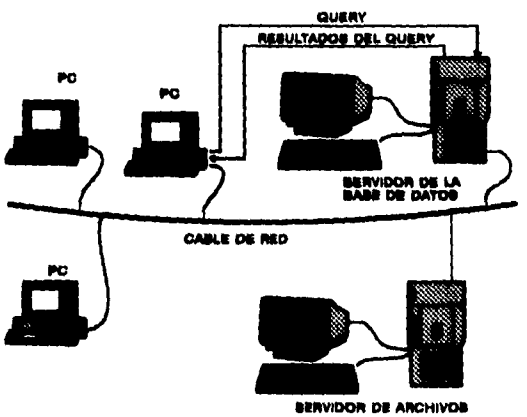
En esta arquitectura el procesamiento de la base de datos se encuentra repartido en el cliente que generalmente son PCs y en el servidor. Los clientes se encargan de ejecutar la aplicación, y el

servidor del manejador de base de datos y de compartir recursos como el espacio en disco, impresoras, etc.

Los clientes generalmente son PCs y el sevidor puede ser desde una PC hasta un mainframe.

La aplicación que corre en los clientes es conocida como front-end; contiene todas la pantallas y procesos de entrada y salida de datos para el usuario. El servidor contiene el back-end, el cual se encarga del procesamiento de los datos y de los accesos a disco.

Una de las ventajas al compartir el procesamiento entre clientes y servidores es que se reduce el tráfico de datos en la red.



Fuente: [SAL93]

Procesamiento Distribuído

Una base de datos distribuida, es aquella que no se encuentra almacenada en su totalidad en un solo lugar físico, sino

que se distribuye a lo largo de una red de computadoras geográficamente separados que se conectan por medio de enlaces de comunicación. [DAT86]

En esta arquitectura los datos son compartidos entre varios hosts a través del envío de actualizaciones, así como de conexiones directas o remotas por medio de una línea telefónica o líneas dedicadas. Una aplicación que corre en uno o más hosts extrae la porción de datos que va a ser modificada durante el proceso. Posteriormente se transmiten los datos a un host centralizado o a otros hosts que se encuntran distribuidos en el circuito. Las otras bases de datos son actualizadas también de tal manera que todo el sistema se encuentra en sincronía.

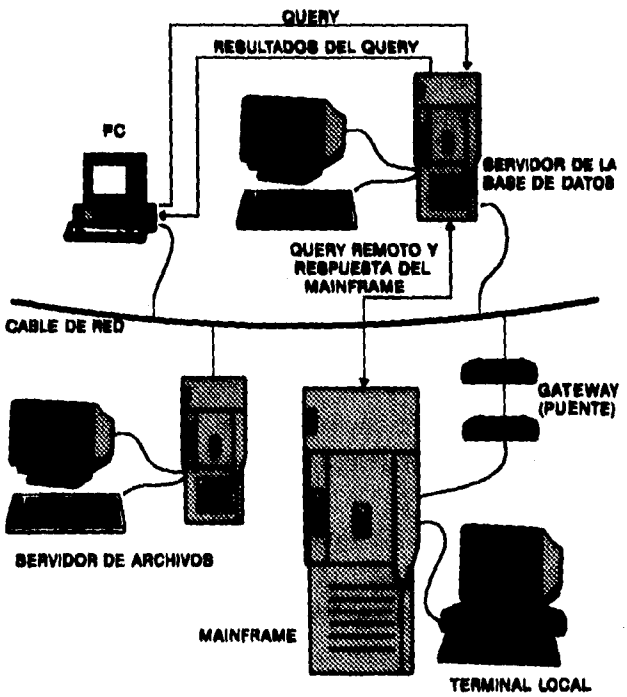
En un sistema distribuido, un usuario solicita información al host local, si los datos no se encuentran en el host, éste los busca a través de la red en todo el sistema hasta encontrarlos.

El usuario percibe a un sistema distribuido como si fuera un sistema centralizado ya que no necesita saber en dónde se encuentran físicamente almacenados los datos, es decir, que es completamente transparente para él. Así pues, el hecho de que la base de datos esté distribuida debe ser importante tan sólo a nivel interno, y no a niveles externos.

Las ventajas de tal distribución son que combina la eficiencia del procesamiento local (sin excesivos costos de comunicación) de la mayoría de las operaciones, confiabilidad y autonomía local.

Los problemas que se presentan en un sistema distribuido es el control global de concurrencia, actualización de datos duplicados,

transparencia de ubicación y optimización de los procesos de consulta.



Fuente: [SAL93]

LENGUAJES DE PROGRAMACIÓN

Los lenguajes utilizados para crear aplicaciones de bases de datos pueden ser agrupados en tres categorías: lenguajes estructurados, cláusulas SQL y otros lenguajes.

Lenguajes Estructurados

En estos lenguajes cada procedimiento realiza una pequeña parte de la aplicación, como por ejemplo un procedimiento para realizar búsquedas, otro para actualizar datos, etc. Todos los procedimientos son ligados o unidos a través de un procedimiento general como puede ser un menú.

Las funciones de interfaz de programación de aplicaciones generalmente están contenidas en librerías, las cuales son incluidas en la aplicación cuando es compilada.

Todos estos lenguajes son conocidos como lenguajes de tercera generación.

Los lenguajes estructurados que son específicos de algún manejador de bases de datos son conocidos como lenguajes de cuarta generación.

Dentro de las características de estos lenguajes tenemos el uso de indentación para hacer las cosas más claras y detectar los bloques de secuencia, debemos considerar la secuencia, selección y repetición o iteracción de acciones.

SQL

El SQL (Structured Query Language) inicialmente fue desarrollado como un lenguaje para realizar accesos a una base de datos basada en el modelo relacional. La versión inicial de este lenguaje se conoció con SEQUEL y fue desarrollado por IBM como

un lenguaje estandard que corría en mainframes de la misma compañía. Posteriormente su nombre fue abreviado a SQL.

El SQL es el lenguaje más popular en arquitecturas cliente/servidor. Diferentes manejadores de bases de datos se pueden comunicar entre sí a través de SQL utilizando comandos para extraer y manipular datos.

Dentro de las diferentes versiones de SQL existen unas que son más estandard que otras, es decir que algunas contienen comandos o sintaxis que otras no pueden interpretar.

Otros Lenguajes

Los más comunes de estos otros lenguajes son los de programación orientada a objetos. Estos representan una manera diferente de programación ya que los procesos o acciones son definidas en objetos en lugar de una serie de procedimientos.

Otro tipo de lenguaje es el de macro o scripts. Este lenguaje no es completamente de programación; son una lista de instrucciones que un usuario puede introducir en una aplicación para automatizar ciertas tareas.

Finalmente existe el QBE (Query by example) que no es estrictamente un lenguaje de programación, es una interfaz que le presenta al usuario una o más copias de las estructuras de las tablas que corresponden a las de la base de datos. El usuario selecciona las columnas que van a ser incluidas y establece las condiciones de búsqueda.

CAPÍTULO II
ARQUITECTURA CLIENTE / SERVIDOR

INTRODUCCIÓN

En el primer capítulo se describió a grandes rasgos lo que es la arquitectura Cliente/servidor (C/S), ahora profundizaremos un poco más en este tema.

La arquitectura C/S se refiere a la automatización de actividades con recursos de cómputo integrados en una red de trabajo. El beneficio de este enfoque es que las aplicaciones pueden ser desarrolladas más rápido y con recursos no tan costosos como los tradicionales mainframes.

Después de definir la arquitectura C/S y resaltar sus principales ventajas y desventajas, mostramos las particularidades tanto del cliente como del servidor, así como aspectos importantes de las diversas plataformas de hardware en que puede ser implantada esta arquitectura.

Finalmente dedicamos una parte de este capítulo a la seguridad, elemento de suma importancia en los sistemas de información, y describimos los diferentes puntos que deben tomarse en cuenta al planear la seguridad de un sistema.

Definición

La arquitectura C/S es una nueva tecnología de bases de datos relacionales que integra a las mejores características de las computadoras personales (software amigable y respuesta rápida) con las mejores características de los mainframes (gran capacidad de almacenamiento y fuerte seguridad). También se combinan hardware y software que le permiten dividir la carga de procesamiento entre la estación de trabajo del usuario (cliente) y el servidor. La interfaz del usuario corre en la estación de trabajo, mientras que el software de la base de datos corre en el servidor y la red une a ambos.

A diferencia de otras formas de procesamiento de información que involucran a un sólo tipo de computadora, la arquitectura C/S contempla múltiples computadoras conectadas a una red de cómputo. Algunos de estos equipos procesan los programas de la aplicación y están designados a trabajar como clientes. Otra computadora se desempeña como servidor y tiene la importante tarea de administrar la base de datos.

Las computadoras clientes pueden ser, en teoría, mainframes, minicomputadoras o microcomputadoras, pero debido al costo, la mayoría de las veces se emplean las microcomputadoras. Similarmente, el servidor puede ser cualquier tipo de computadora, pero de igual forma, casi siempre se trata de una microcomputadora. El servidor y los clientes generalmente se conectan usando una red de área local.

Esta arquitectura promete una nueva forma de accesar a los datos a un costo más bajo que con las soluciones clásicas de los mainframes por lo que, en muchos casos, las grandes computadoras se están viendo desplazadas por redes de PCs.

Características

Las características básicas de la arquitectura C/S que la diferencian de las demás son:

- Recursos Compartidos. Un servidor puede dar respuesta a peticiones de muchos clientes al mismo tiempo y regular el acceso de éstos a los recursos con que cuente.

- Servicios. Cliente/Servidor es una relación entre procesos que corren en máquinas separadas. El proceso del servidor es un proveedor de servicios. El cliente es un consumidor de servicios. En resumen, C/S proporciona una clara separación de funciones basadas en la idea de servicios.

- Protocolos Asimétricos. Existe una relación de muchos a uno entre clientes y servidor. Los clientes son los que siempre inician el diálogo solicitando un servicio. Los servidores se encuentran esperando pasivamente a que los clientes realicen peticiones.

- Transparencia en la ubicación. El servidor es un proceso que puede localizarse en la misma máquina que ocupa el cliente, o en una diferente que se encuentre en la red.

- Intercambios basados en mensajes. Los clientes y servidores son sistemas que interactúan a través de un mecanismo de paso de mensajes. El mensaje es el mecanismo de entrega de servicios requeridos y contestados.

- Escalabilidad. Los sistemas C/S pueden ser escalados horizontal o verticalmente. La escalabilidad horizontal significa añadir o quitar estaciones clientes con tan sólo un ligero impacto en el nivel de eficiencia de ejecución del sistema. La escalabilidad vertical se refiere a la migración hacia servidores o multiservidores más grandes y más rápidos.

- Integridad. El código y los datos del servidor son mantenidos de manera centralizada, característica que se ve reflejada en un costo de mantenimiento y de conservación de la integridad de los datos más barato.

- Encapsulamiento de servicios. El servidor es un especialista. Cada mensaje le indica al servidor qué es lo que se está requiriendo. Al servidor le corresponde determinar como resolver el requerimiento. Los servidores pueden ser mejorados, entiéndase cambiados o actualizados, sin afectar a los clientes, siempre y cuando la interfaz de mensajes no sea cambiada.

Cliente

El cliente representa la interfaz con el usuario y puede hacer uso de varios sistemas operativos, incluyendo DOS, OS/2, UNIX o Apple Macintosh. El software del cliente (front-end) puede ser un manejador de base de datos, una hoja de cálculo, un procesador de palabras o cualquier otro tipo de aplicación. Sus funciones básicas son la presentación y validación de datos y realización de peticiones al servidor.

En la parte del cliente se ha desarrollado ampliamente el concepto GUI (Interfaz gráfica del usuario) que hace más que proveer una capa para presentar la aplicación, también provee un ambiente de trabajo arriba del sistema operativo del cliente; por ejemplo, Windows 3.1 expande la administración de memoria del sistema operativo con su propio administrador de memoria y simula un ambiente de multitarea.

Para muchos usuarios el cliente es todo el sistema, por lo tanto la interfaz debe estar libre de errores y ser amigable. Para el desarrollo de la interfaz se debe seguir los siguientes puntos:

- Definir lineamientos. Se hace con la finalidad de que al usuario final le sea más fácil el aprendizaje y uso de la aplicación. Los estandard deben abarcar programación, sintaxis, iconos y colores.

- Conocer al usuario. Los desarrolladores deben tomar en cuenta la posición del usuario y analizar diversos aspectos, tales como la terminología que utilizará, proveer teclas rápidas para los procesos más usados, empleo del mouse, combinaciones de colores, etc.

- Simplicidad en los procesos. Hay que buscar la manera más fácil de programar cada opción de la aplicación, por ejemplo en los GUI generalmente se dispone de diversas herramientas, a veces se puede usar más de una de estas para programar un mismo proceso, aquí entonces habrá que elegir el mejor.

- Informar sobre la ejecución de procesos Internos. Este punto se refiere a los casos en que se hagan lecturas o escrituras al servidor. Para que el usuario sepa que la computadora está

trabajando, se recomienda que el apuntador del mouse cambie de forma (puede ser a reloj de arena) o bien, cuando se sabe que el proceso tardará más de 10 segundos, es conveniente enviar en mensaje de que se requiere un momento de espera.

Servidor

Por su parte, el servidor (o back-end) puede correr en cualquier sistema operativo, almacena los datos, acepta los requerimientos de los clientes, los procesa y regresa una respuesta al cliente que lo solicitó. Mientras esto ocurre, el servidor revisa la integridad de la base de datos y optimiza la ejecución de la petición. La ejecución de estas actividades se ve favorecida por dos elementos distintivos de esta arquitectura:

1.- Procedimientos Almacenados. Son instrucciones SQL que se almacenan en el servidor como parte de la base de datos. Son llamados desde los clientes pero se ejecutan completamente en el servidor. Los DBMS ya incluyen un conjunto de estos procedimientos enfocados a la administración de usuarios, monitoreo del servidor y de apoyo para crear bases de datos, y los programadores o administradores del sistema pueden agregar más de estos procedimientos. Todos ellos cuentan con la seguridad propia de los DBMS y pueden tener un fuerte impacto en el nivel de eficiencia de ejecución de las instrucciones que lleguen al servidor

2.- Triggers. Son un tipo especial de procedimientos que automáticamente se ejecutan cuando ocurre una inserción, borrado o actualización en el servidor. Tienen el objetivo de

conservar la integridad referencial y poder realizar validaciones antes de que la instrucción se lleve a cabo.

En un sistema C/S puede existir más de un servidor que procesen diferentes bases de datos o que brinden otros servicios a los clientes. Si se presenta este caso, entonces cada uno de ellos debe procesar diferentes bases de datos del mismo sistema para que se pueda considerar que está bajo la estructura cliente/servidor, ya que cuando dos servidores procesan la misma base de datos, el sistema deja de ser cliente / servidor y se convierte en un sistema de base de datos distribuido.

Red

La red de cómputo, compuesta por tarjetas, cables, sistema operativo, etc, proporciona un canal de comunicación entre los clientes y el servidor, por medio del cual se transmiten datos entre ambos. Cabe mencionar que a nivel de red el concepto middleware es de gran importancia.

Middleware es un término que cubre todo el software necesario para soportar interacciones entre clientes y servidores. Es el medio que permite al cliente obtener servicios del servidor, éste inicia sus funciones cuando el cliente invoca un servicio y cubre la transmisión de la petición a través de la red así como la respuesta que el servidor regresa. No incluye el software del DBMS ni el utilizado para desarrollar la aplicación del cliente.

VENTAJAS Y DESVENTAJAS

Las bondades o limitantes que la arquitectura cliente/servidor puede traer consigo dependen mucho de las plataformas en las que trabajen, el front-end y el back-end, así como del nivel de distribución del trabajo entre ambos. En seguida presentamos algunas de las principales ventajas que tiene esta arquitectura y posteriormente mostraremos la contraparte, es decir, las desventajas que hemos identificado en esta tecnología.

Ventajas

La división en el procesamiento de la información trae consigo diversos beneficios tanto en la parte del cliente como en el propio servidor.

Se reduce el tráfico de datos en la red de cómputo, esto debido a que a diferencia de otras arquitecturas, a través de la red sólo viajan peticiones llamadas queries y respuestas a los mismos, y no todo un archivo de datos. Cabe mencionar que en algunos servidores de bases de datos inclusive se pueden almacenar y ejecutar procedimientos o queries, disminuyendo aun más el tráfico en la red.

Los clientes deben preocuparse únicamente por ejecutar adecuadamente el front-end, por lo que se extiende de manera efectiva la vida de algunas computadoras que no tienen el poder suficiente para soportar un sistema manejador de bases de datos completo.

Otro beneficio de separar el cliente del servidor es que los usuarios no están limitados a utilizar un determinado tipo de sistema o plataforma. En un sistema cliente/servidor las estaciones de trabajo pueden ser PCs compatibles, Macintoshes, estaciones de trabajo UNIX, o una combinación de estas, y pueden correr múltiples sistemas operativos. De esta forma los usuarios pueden continuar usando software con el que ya están familiarizados para accesar la base de datos.

Una ventaja también importante es la facilidad para preservar la integridad de los datos. Actualmente los servidores de bases de datos corren sistemas basados en el modelo relacional y la seguridad de los datos se garantiza previniendo que el usuario accese a los archivos desde afuera del manejador de bases de datos ya que estos se encuentran encriptados. También se cuenta con discos espejos en los que la información automáticamente se graba en una base de datos idéntica que se localiza en un disco duro diferente.

La última ventaja en mencionar es el procesamiento de transacciones que consiste en un método con el cual el manejador de base de datos conserva un registro de las modificaciones hechas a la base de datos en un período determinado que se conoce como log. Principalmente se utiliza en bases de datos que constantemente se modifican y tiene por función poder restablecer la base de datos, tanto como sea posible, a un estado libre de errores.

Desventajas

La principal desventaja de los sistemas cliente/servidor es el incremento en el costo del personal de soporte y administración quienes mantienen la base de datos. En una red pequeña que contemple hasta 20 usuarios, el administrador de la red puede

generalmente mantener el servidor de base de datos y el acceso de los usuarios a éste. Sin embargo, conforme el número de usuarios se incrementa o crezca la base de datos, el número de personal también deberá incrementarse para poder cumplir adecuadamente con esta función. Obviamente en comparación con una arquitectura centralizada es menor el número de personas necesarias para administrar el equipo.

Otra desventaja es el hecho de que al haber una independencia de aplicaciones (front-ends) en los clientes, si se realiza una modificación en la estructura de la base de datos entonces se incrementará el trabajo de mantenimiento en las aplicaciones de los clientes, porque se tendrá que hacer modificaciones en una variedad de código en los diferentes clientes.

El costo del manejador de la base de datos es significativamente superior al tradicionalmente utilizado en las PCs, y habría que sumarle el costo de los front-end o herramientas de desarrollo así como el costo de capacitar a personal como programadores.

PLATAFORMAS

La plataforma es la combinación de software y hardware sobre la cual corre el sistema cliente/servidor. Existen cuatro categorías de plataformas:

- Computadoras Personales
- Estaciones de trabajo UNIX
- Minicomputadoras
- Mainframes

A pesar de que la plataforma más común para el procesamiento cliente/servidor sea el de PCs, las cuatro presentan ventajas y desventajas.

Aunque los sistemas de hardware varían ampliamente en características y capacidades, se requieren ciertas características para el software del sistema operativo(SO), mismo que es un software básico que actúa como interfaz entre el hardware y las aplicaciones.

El SO utilizado en un sistema C/S debe contemplar las siguientes características:

- Multitareas. Se refiere a poder ejecutar numerosas aplicaciones concurrentemente.
- Multiusuarios. Poder soportar el trabajo de numerosos y simultáneos usuarios realizando diferentes tareas.
- Multihilos. Es la capacidad de que una misma aplicación pueda tener la característica de multitarea.

A continuación presentamos los principales aspectos de las diferentes plataformas:

Computadoras Personales

Sólo en los más recientes años las computadoras personales compatibles con IBM se han convertido en una plataforma aceptable para las bases de datos C/S. Los sistemas 80486 con discos duros en gigabytes y sistemas operativos multitareas han impulsado a las PCs

a actuar como plataformas de los sistemas manejadores de bases de datos (DBMS).

La gran ventaja de las PCs es que son bien conocidas y en muchos casos ya se dispone de ellas. De hecho una red de área local requiere de adiciones mínimas de hardware para agregar una PC como servidor.

Una computadora PC 80386 puede trabajar adecuadamente como servidor pero cabe mencionar que definitivamente un procesador 80486 se podrá desempeñar significativamente mejor. Aquí hay que recordar que una computadora de tecnología escalable es muy recomendable debido a la facilidad de convertirla a 80486 o 80586 (pentium).

Una de las preguntas en cuanto a hardware se refiere podría ser cuánta memoria RAM utilizar en el caso de un servidor PC, el mínimo se de 16 Mb, pero se recomiendan 32 Mb. Esto dependerá del número de usuarios y del DBMS. También se debe tomar en cuenta que con el paso del tiempo el sistema puede requerir más memoria debido a conexiones de más usuarios y habrá que tomar las medidas adecuadas a tiempo.

En lo que se refiere al sistema operativo, existen varias opciones que bien pueden ser usados en el servidor. Las base de datos pueden correr bajo NetWare 3.11, OS/2, o cualquier versión de UNIX para PC 486.

Por otra parte podemos decir que el costo referente al personal se puede ver incrementado, especialmente si el personal ya existente no puede, debido a falta de conocimientos, hacerse cargo de la nueva plataforma. Los sistemas cliente/servidor pequeños

pueden ser administrados por personal que conozca de redes de área local, pero en sistemas de mayores dimensiones, se vuelve necesario contar con personas capacitadas que se hagan responsables únicamente del buen funcionamiento del DBMS.

Por lo tanto, consideramos que debe hacerse un balance que por un lado tome en cuenta las características del DBMS con sus respectivas ventajas y, por el otro lado, contemple el costo de obtenerlo y de proporcionarle mantenimiento, estando conscientes que es un nuevo sistema que puede traer consigo fuertes complicacones.

Estaciones de trabajo UNIX

En la actualidad los sistemas UNIX no tienen un mercado muy grande (aproximadamente el 9%, según PC Magazine). Esto es difícil de explicar siendo que gran parte de los servidores de sistemas manejadores de bases de datos relacionales para PCs fueron adaptados de bases de datos que utilizaban UNIX. Los expertos opinan que lo anterior se debe a la errónea idea que algunas personas tienen de que UNIX es difícil de aprender y de usar, particularmente quienes están muy familiarizados con las PCs.

Otra razón importante sin duda es el costo que puede observarse en dos rubros, el primero es adquirir el equipo y el segundo es contar con el soporte técnico que permita el funcionamiento de dicho equipo.

Los precios de los superservidores y minicomputadoras de menor capacidad generalmente van desde 20,000 dólares y soportan a un número reducido de usuarios, comparado con los cientos de

usuarios que permiten los grandes sistemas. En cuanto al software, los paquetes para UNIX son más caros que sus equivalentes para PC y debido a las pequeñas incompatibilidades entre las diferentes versiones de UNIX, los proveedores de DBMS se ven obligados a contar también con numerosas versiones de sus productos.

El personal capacitado en esta área es aun muy poco, lo que se traduce en tener que pagar salarios elevados a personas experimentadas o bien, invertir en capacitar al personal de la organización.

Finalmente, otra razón del reducido tamaño de los sistemas UNIX es el protocolo de red. Siendo TCP/IP el protocolo primario de los sistemas UNIX, la mayoría de las LANs (redes de área local) de PCs no lo soportaban como protocolo nativo. Las PCs podían hacer uso de TCP/IP para comunicarse con sistemas UNIX pero como terminales inteligentes al sistema host. En los últimos años diversas compañías como Novell e IBM empezaron a ofrecer TCP/IP como protocolo nativo en sus respectivas LANs, dándole a los clientes la capacidad de comunicarse con servidores UNIX de base de datos, resolviéndose así el problema del protocolo.

Después de mencionar todas estas limitantes de los sistema UNIX, también debemos decir que al hacer uso de esta plataforma se obtendrán significantes beneficios, tales como:

- El poder de un superservidor o una minicomputadora es sumamente superior al de una PC, lo que trae consigo soporte para más usuarios.

- Mayor velocidad que se traduce en mejores tiempos de respuesta, esto gracias a las formas de multiprocesamiento que

las minicomputadoras y superservidores soportan debido a que cuentan con más de un CPU.

- Pueden tener discos con gran capacidad de almacenamiento (desde 500 Mb hasta 100G generalmente), esto comparado con las PCs que se limitan a cantidades de espacio en disco no mayores a los 10G.

- La seguridad es otro factor que debe tomarse en cuenta. Los sistemas UNIX bien administrados pueden ser tan seguros contra accesos no autorizados como los sistemas operativos propietarios de minicomputadoras y mainframes.

Los procesadores de las estaciones de trabajo RISC (Reduced Instruction Set Computing) han sido generalmente utilizados para aplicaciones científicas. Los CPUs de este tipo de computadoras realizan su procesamiento reduciendo la cantidad de microcódigo en un chip; con menos código el CPU puede ejecutar sus operaciones internas más rápidamente. En la actualidad ya existen numerosos chips RISC disponibles, tales como Sun SPARC, DEC's Alpha, las series 88000 de motorola, etc. Desde que el sistema operativo que se emplea para las estaciones de trabajo RISC es UNIX, esta plataforma ha empezado a usar un servidor de base de datos.

Minicomputadoras

Las minicomputadoras, así como los mainframes, han sido las plataformas tradicionales para el desarrollo de sistemas de base de datos. Un sistema C/S puede aumentar los alcances de estas computadoras de la siguiente manera:

- Extiende el acceso a los datos desde computadoras personales o bien desde redes de área local.

- Permite ser utilizado por más usuarios sin tener que hacer modificaciones de hardware debido a que se reduce la caga de trabajo de la minicomputadora moviendo parte del procesamiento hacia el front-end.

El aumento en el uso de las microcomputadoras y sistemas RISC basados en UNIX redujo el poderío que tenían los vendedores de minicomputadoras tales como Hewlett-Packard y Digital Equipment Corporation, mismo que durante el período de 1970 a mediados de los 80's dominaron el mercado de estos procesadores. Esos sistemas eran propietarios debido a que principalmente corrían el sistema operativo proveído por dichos vendedores y no podía ser ejecutado en ningún otro hardware. Posteriormente los mismos proveedores se defendieron lanzando versiones de UNIX para sus sistemas, dándoles un toque de sistemas abiertos porque ya permiten conexiones con PCs y estaciones de trabajo a través de LANs. Ya en los últimos años de los 80's las grandes compañías de minicomputadoras no tuvieron más remedio que enfocarse a los sistemas abiertos y olvidar los sistemas propietarios es decir, ahora en lugar de minicomputadoras manejan superservidores.

Las minicomputadoras propietarias ofrecen las mismas ventajas que los superservidores que trabajan en UNIX en cuanto a número de usuarios que soportan y al tamaño potencial de la base de datos. Las grandes minicomputadoras pueden soportar a más de 500 usuarios y bases de datos de tamaño de cientos de gigabytes.

Sin embargo esta gran potencialidad va acompañada de una desventaja importante que es el costo y que se ve reflejada de la siguiente manera:

- El valor del hardware necesario para lograr dicha potencialidad es muy superior al valor de los superservidores, inclusive siendo ambos del mismo proveedor.

- El costo de las licencias de los sistemas operativos igualmente es caro y, a diferencia de UNIX.

- El personal técnico que pudiera dar soporte a estos sistemas es realmente reducido y, haciendo nuevamente una comparación con UNIX, es más difícil para una persona con experiencia en un sistema operativo propietario familiarizarse con un sistema operativo propietario diferente que para una persona con experiencia en administración de sistemas acoplarse a una versión diferente de UNIX, en el primer caso la inversión en capacitar al personal es indispensable.

Por todo lo antes mencionado, podemos decir que el hacer uso de esta plataforma sólo es recomendable en aquellos casos en que ya se cuente con la minicomputadora y para obtener más beneficios de ese equipo se puede modificar el ambiente, haciendo uso de la arquitectura C/S así el sistema podrá extender sus alcances sin tener que trasladar los datos a un sistema de base de datos diferente.

Mainframes

La IBM recientemente les empezó a llamar "warehouses for large businesses" es decir, almacenes para grandes compañías. Estas computadoras son de propósito general y son las más poderosas. Soportan múltiples procesos que requieren de alta velocidad, enormes cantidades de espacio en disco y de cientos a miles de usuarios simultáneamente accesando a múltiples aplicaciones a través de terminales o conexiones de redes. También ofrecen gran seguridad de los datos.

Estas computadoras todavía son empleadas en muchas de las grandes compañías como sistemas primarios en aplicaciones de bases de datos centralizadas, pero el rápido aumento del uso de PCs y estaciones de trabajo han provocado una lenta evolución en el uso de los mainframes como host para sistemas con arquitectura C/S.

En cuanto a su costo, los mainframes son las computadoras más caras en términos de hardware, software y personal. A diferencia de los otros tipos de sistemas de cómputo, los mainframes requieren un ambiente controlado, incluyendo temperatura, nivel de piso y hasta dispositivos especiales de refrigeración.

Otra diferencia sobresaliente de los mainframes es que físicamente no se componen de una sola caja o gabinete, generalmente consisten de varios subsistemas que realizan diferentes tareas, todas unidas por cables de fibra óptica. Los subsistemas típicos son los de CPU, RAM, sistema de comunicaciones y los de disco y controladores de cintas.

La velocidad y poder de procesamiento de los mainframes se logra gracias a que cuentan con múltiples CPUs, controladores de

discos de alta velocidad y rutas de comunicación también de alta velocidad entre todos los diferentes elementos que los conforman.

Los mainframes generalmente cuentan con más de 256Mb de RAM y algunos soportan hasta gigabytes en RAM.

Los más rápidos y con más poder de procesamiento (y más costosos) son las supercomputadoras, tales como los de Cray y Control Data Corporation, computadoras que son usadas para aplicaciones especiales.

En cuanto se refiere al sistema operativo, éste se encuentra muy modularizado, con diferentes subsistemas que coordinan asignaciones al CPU, comunicaciones con sistemas de almacenamiento de cintas y discos, e interacciones del usuario con la computadora.

Los mainframes de IBM corren uno de los dos sistemas operativos propietarios multitarea y multiusuario: VM y varias versiones de VMS. IBM también cuenta con una versión de AIX que permite correr aplicaciones de UNIX en el mainframe.

Todas estas características de los mainframes que afectan el costo del personal y mantenimiento de los mismos, es lo que ha conducido a someter a los grandes sistemas a un proceso downsizing es decir, a pasar esos complicados sistemas de mainframes a minicomputadoras y PCs. Los sistemas C/S están alcanzando el poder de muchas aplicaciones de mainframes, sin embargo, estos sistemas no pueden todavía reemplazar completamente a los mainframes.

Actualmente la tendencia es integrar a los mainframes ya existentes en nuevos sistemas corporativos C/S es decir, aprovechar este valioso hardware y adaptarlo a mejores arquitecturas.

Front-Ends

El Front-end es la aplicación que se encuentra en los clientes y es quién se encarga de realizar peticiones al servidor. En la actualidad existen front-ends disponibles para cualquier plataforma (aunque la mayoría trabaja en PCs) y soportan ambientes de DOS, Windows y OS2. Los Front-ends proporcionan un conjunto de herramientas que el programador puede utilizar para diseñar y programar las aplicaciones.

Existe en el mercado de software una enorme cantidad de Front-ends (más de 400 según PC Magazine de mayo de 1992) pero todos estos pueden ser agrupados en las siguientes cuatro categorías:

- **Add-ons de productos ya existentes.** Son módulos que permiten a determinado software (dBase, Lotus 1-2-3, etc.) previamente lanzado al mercado, poder comunicarse con servidores de base de datos, es decir es un complemento que les permite poder realizar peticiones SQL al servidor. Estos módulos hacen más fácil el proceso de diseñar y desarrollar un sistema C/S, partiendo de la aplicación que ya exista así como de la base de datos que utilice. *Esta integración en red será menos costosa y más sencilla porque los usuarios y los programadores ya están familiarizados con el código y con la base de datos existentes, así con una capacitación mínima*

podrán adaptar sus técnicas utilizadas para accesar a los datos del servidor. [SAL93]

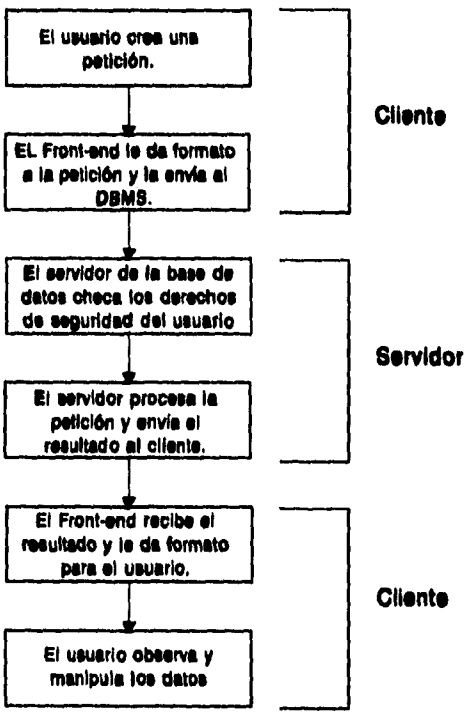
Algunos ejemplos de estos Add-ons son: dbase Server Edition, Paradox y Paradox SQL Link, DataEASE SQL, Superbase 4 y DataLens drives for SQL de Lotus 1-2-3 para Windows.

- **Herramientas de desarrollo de aplicaciones.** Tienen el objetivo de facilitar el trabajo a los programadores en el proceso de creación de una aplicación particular. Esta categoría es muy importante, tal vez la más utilizada. Los proveedores de DBMS manejan algunos toolkits de programación que pueden ser usados con un lenguaje de tercera generación(3GL) para crear aplicaciones de Front-end. Los productos de desarrollo de aplicaciones apoyan en la creación de formas que contienen peticiones al servidor, reportes y menús: Algunos ejemplos de estos son: ObjectVision, PowerBuilder, Q&A, SQL Windows y Visual Basic.

- **Reporteadores.** Como su nombre lo indica, son programas que permiten la elaboración de reportes, con un ambiente muy amigable. Estos programas interpretan lo que el usuario realiza en la pantalla y lo convierten en instrucciones SQL. Algunos reporteadores tienen excelentes interfaces para la conformación de las peticiones pero tienen capacidades mínimas para detallar el reporte, otros tienen características contrarias. La elección de este dependerá de la complejidad de los reportes que se quieran elaborar. Algunos ejemplos de estos son: Quest, Personal Access, Oracle Card y Report Writer (R&R).

- **Herramientas de análisis e integración de datos.** Están diseñadas para administradores o ejecutivos que necesiten examinar datos localizados en diversas fuentes. Al poder reunir información que se encuentra dispersa, se vuelve un arma fuerte para la toma de decisiones. Algunos ejemplos de este tipo de Front-end son: Forest & Trees, InfoAlliance y LightShip.

El diagrama siguiente describe la secuencia de los eventos que ocurren cuando un usuario accesa un servidor de base de datos desde un Front-end.



SEGURIDAD

En los sistemas de información, la seguridad es un punto que no puede pasarse por alto. Consiste en una serie de medidas que protejen a la base de datos contra accesos no autorizados, alteraciones erróneas intencionales y perdida de la integridad referencial de los datos. El administrador de la base de datos es el responsable de administrar y monitorear la seguridad de los datos, asignación de claves de acceso (passwords con sus respectivas restricciones), y checar violaciones de seguridad. El nivel de seguridad que se pueda alcanzar dependerá fuertemente del sistema operativo y del software de base de datos que se utilice, sin embargo, cabe mencionar que muchos de los DBMS para PCs ofrecen muy poca seguridad y no conservan logs de las actividades de los usuarios, a diferencia de los DBMS creados para servidores de bases de datos que, como se indicó anteriormente, contribuyen a la seguridad de los datos, previniendo que el usuario accese a los archivos desde afuera del DBMS, ya que estos se encuentran encriptados y sí guardan un registro de las transacciones que realiza el usuario.

Otras medidas de seguridad de los DBMS es que cuando se va a realizar un conjunto de transacciones (instrucciones) que modifique a más de una tabla, estas pueden tratarse como si fuera una sola instrucción a través de herramientas conocidas como commits, cuyo objetivo es garantizar que se ejecuten todas o ninguna de esas instrucciones, protegiendo así la integridad de la información. Los passwords son usados para controlar el acceso a la aplicación entera y, dependiendo del DBMS, también para el uso de menús, formas, reportes, tablas y hasta determinadas columnas de dichas tablas.

Al trabajar en red, diversos usuarios pueden tener acceso a las tablas que integran a la base de datos, esto comúnmente origina el problema de que simultáneamente se quiera modificar una tabla por dos usuarios. Este problema se resuelve bloqueando archivos, páginas o registros. El bloqueo de los datos trae consigo lentitud en el acceso de otros usuarios a los datos bloqueados pero es indispensable. El bloqueo presenta un problema más llamado dead lock (descritos en al capitulo 1) pero es de mencionar que hay DBMSs que resuelven este problema en forma automática porque detectan este conflicto y le dan prioridad al usuario que primero accesó. Un enfoque diferente de visualizar la seguridad es el que la divide en las siguientes características:

- *Física. El lugar que contienen los sistemas de cómputo debe ser físicamente seguro contra la entrada de intrusos.*
- *Humana. La autorización de usuarios debe ser hecha de manera cuidadosa para reducir el riesgo de que un usuario autorizado por medio de un soborno o como un favor, le de acceso a una persona no autorizada.*
- *Sistema Operativo. No importa que tan seguro sea el DBMS, si el sistema operativo es débil en cuanto a seguridad puede permitir accesos no autorizados a la base de datos.*
- *Sistema de base de datos. Desde la misma base de datos se puede ejercer un control muy importante que tiene que ver con la asignación adecuada de permisos. De este modo, a algunos usuarios se les podría autorizar tener acceso a toda la información, a otros se les podría permitir realizar consultas a una porción limitada de la base de datos y prohibirles realizarmodificaciones. Será responsabilidad del DBMS asegurar que esas restricciones no sean violadas.* [KOR86]

CAPÍTULO III
ADESI

INTRODUCCIÓN

La administración escolar es un aspecto fundamental para el desarrollo universitario, misma que desde hace 3 décadas es apoyada por herramientas de cómputo, cuando se instaló la primera computadora en la sección de máquinas de la Secretaría Auxiliar, con el objetivo de atender los requerimientos de automatización del sistema de control escolar, generando las actas de exámenes y listas de alumnos, así como requerimientos de índole administrativo.

De esta manera la Universidad Nacional Autónoma de México es una de las primeras instituciones educativas en apoyar sus actividades académico-administrativas y científicas con el uso de sistemas de cómputo.

Las actividades de administración escolar se han convertido en un factor crítico ya que la Universidad Nacional Autónoma de México cuenta, en sus diferentes planteles, colegios de ciencias y humanidades, escuelas y facultades, con una población estudiantil de más de 270,000 alumnos.

Desde hace algunos años, dado el crecimiento en el número de alumnos de la UNAM, surgió la necesidad de mejorar los servicios de administración escolar, esto con el deseo de poder dar respuesta inmediata a los requerimientos de los estudiantes.

Dada la magnitud de la información y de los diferentes procesos involucrados en la administración escolar, es necesario realizar un diseño detallado para que el sistema tenga una vida útil que satisfaga las necesidades presentes y futuras haciendo uso de nuevas tecnologías adecuadas y que tiendan a la descentralización de procesos e información.

En este capítulo se plantea la situación que guarda actualmente la administración escolar en la ENEP Iztacala, así como las ventajas y desventajas que representa el automatizar esta actividad.

Para lograr el objetivo anterior se desarrolló el sistema de **AD**ministración **ES**colar **I**ztacala (ADESI) cuyos módulos se explican en este apartado.

ANTECEDENTES

Dada la cantidad de alumnos que la Escuela Nacional de Estudios Profesionales Iztacala tiene que controlar, aproximadamente 9000, surge la necesidad de contar con sistemas capaces de registrar en forma oportuna y veraz los movimientos que se generen por cada alumno, desde el inicio de sus estudios hasta la terminación de los mismos.

Presentaremos entonces una reseña del nacimiento y presente de la ENEP Iztacala.

ENEP Iztacala

Durante el rectorado del Dr. Guillermo Soberón Acevedo, el Consejo Técnico aprobó un Programa de Descentralización que tuvo la intención de promover la creación de las Unidades Multidisciplinarias. Gracias a este programa surgió la Escuela Nacional de Estudios Profesionales Iztacala (ENEP Iztacala) en el año de 1975.

La ENEP Iztacala se encuentra en el Fraccionamiento Los Reyes, Iztacala, en Tlalnepantla, Estado de México. Las instalaciones que ocupa incluye más de 25 edificios diversos entre edificios de aulas, unidades de investigación y laboratorios, biblioteca, cafetería, gimnasio y estacionamientos, además de los campos deportivos, clínica odontológica y clínica universitaria de salud integral, así como una unidad móvil de atención odontológica. [GUI91]

La ENEP Iztacala cuenta con más de 1900 miembros de personal académico que atienden una población de licenciatura superior a los 7000 alumnos, cerca de un centenar de estudiantes de posgrado y más de 1000 técnicos en enfermería. Cada año ingresan a la ENEP Iztacala más de 2000 estudiantes, y egresan alrededor de 1200, aprobando examen profesional más de 600 alumnos anualmente. [AGE93]

En la ENEP Iztacala se imparten la carrera de técnico en enfermería y las licenciaturas de Biólogo, Cirujano Dentista, Médico Cirujano, Psicología y Optometría. A nivel de posgrado se imparte la especialización en Endoperiodontología y las maestrías en Biología de la Reproducción, Biología de Recursos Vegetales, Farmacología Conductual, Investigación de Servicios de Salud, Modificación de Conducta, Neurociencias y Psicología (Metodología de la Teoría de la Investigación Conductual).

A la vez es indispensable que este sistema que apoya a las actividades de administración escolar, se apeguen completamente a los lineamientos que determina la Legislación Universitaria. Por lo cual, a continuación se presenta el marco legal con los principales señalamientos jurídicos relativos a esta materia.

Marco legal (Tomado de la Legislación Universitaria)

Como ya se ha comentado, es de suma importancia que los sistemas de cómputo que se desarrollan para apoyar la administración escolar, respeten fielmente las disposiciones que en esa materia marca la Legislación Universitaria, por lo que a continuación se presenta un resumen de los principales artículos que al respecto se deben considerar:

Ley Orgánica de la UNAM

Artículo 1o.: En materia Legislativa, la Universidad Nacional Autónoma de México es un organismo descentralizado del Estado y dotado de plena capacidad jurídica para los fines de: impartir educación superior para formar profesionistas, investigadores, profesores universitarios y técnicos útiles a la sociedad; así como organizar y realizar investigaciones principalmente acerca de las condiciones y problemas nacionales, y extender con la mayor amplitud posible los beneficios de la cultura.

Estatuto General de la UNAM

Artículo 7: La Universidad está integrada por sus autoridades, investigadores, técnicos, profesores, alumnos, empleados y los graduados de ella.

Artículo 8: La función docente se realiza por las siguientes instituciones:

I. Facultad de Filosofía y Letras;
II. Facultad de Ciencias;
III. Facultad de Derecho;
IV. Facultad de Ciencias Políticas y Sociales;
V. Facultad de Economía;
VI. Facultad de Contaduría y Administración;
VII. Escuela Nacional de Trabajo Social;
VIII. Facultad de Medicina;
IX. Escuela Nacional de Enfermería y Obstetricia;
X. Facultad de Odontología;
XI. Facultad de Medicina Veterinaria y Zootecnia;
XII. Facultad de Ingeniería;
XIII. Facultad de Química;
XIV. Facultad de Psicología;
XV. Facultad de Arquitectura;
XVI. Escuela Nacional de Artes Plásticas;
XVII. Escuela Nacional de Música;
XVIII. Facultad de Estudios Superiores Cuautitlán;
XIX. Escuela Nacional de Estudios Profesionales Acatlán;
XX. Escuela Nacional de Estudios Profesionales Iztacala;
XXI. Escuela Nacional de Estudios Profesionales Aragón;
XXII. Facultad de Estudios Profesionales Zaragoza;
XXIII. Escuela Nacional Preparatoria.

Artículo 12: Las autoridades universitarias constituyen:

I. La Junta de Gobierno;
II. El Consejo Universitario;
III. El Rector;
IV. El Patronato;
V. Los Directores de facultades, escuelas e institutos y aquellos que se designen con motivo de la coordinación de los anteriores en las unidades académicas del Colegio de Ciencias y Humanidades, y
VI. Los consejos técnicos de las facultades y escuelas y los de investigación científica y humanidades.

Reglamento General de Inscripciones

Artículo 1: La Universidad Nacional Autónoma de México selecciona a sus estudiantes tomando en cuenta el grado de capacitación académica y las condiciones de salud de los mismos.

Artículo 2: Para ingresar, es indispensable:

a) Solicitar la inscripción de acuerdo con los instructivos que se establezcan.

b) Haber obtenido en el ciclo de estudios inmediato anterior un promedio mínimo de siete o su equivalente.

c) Ser aceptado mediante concurso de selección, que comprenderá una prueba escrita y que deberá realizarse dentro de los períodos que al efecto se señalen.

Artículo 4: Para ingresar a nivel de licenciatura el antecedente académico indispensable es el bachillerato. Para efectos de revalidación o reconocimiento, la Comisión de Incorporación y Revalidación de Estudios del Consejo Universitario, determinará los requisitos mínimos que deberán reunir los planes y programas de estudios de bachillerato.

Artículo 5: El Consejo Técnico de cada facultad o escuela establecerá el número de estudiantes de primer ingreso que cada año podrá ser inscrito en cada carrera o plantel.

Artículo 7: Una vez establecido el cupo para cada carrera o plantel, los aspirantes serán seleccionados según el siguiente orden:

a) Alumnos egresados de la Escuela Nacional Preparatoria o del bachillerato del Colegio de Ciencias y Humanidades.

b) Aspirantes con promedio mínimo de siete en el bachillerato, aprobados en el concurso de selección, con la siguiente prioridad:

1. Egresados de escuelas del Distrito Federal incorporadas a la UNAM.

2. Egresados de escuelas de provincia incorporadas a la UNAM.
3. Egresados de escuelas del Distrito Federal no incorporadas a la UNAM.
4. Egresados de escuelas de provincia no incorporadas a la UNAM.

En los casos 2 y 4 sólo se atenderán solicitudes de inscripción para carreras que no se imparten en la Universidad de la entidad federativa donde el aspirante realizó sus estudios.

Artículo 8: Los aspirantes que provengan de otras instituciones de enseñanza superior podrán ingresar al nivel de licenciatura, en años posteriores al primero, cuando cumplan los requisitos indispensables y el cupo de los planteles lo permita. El concurso de selección consistirá para el caso, en un examen global, escrito y oral, de las materias que pretendan revalidar o acreditar, por lo menos ante dos sinodales, y en ningún caso se revalidará o acreditará más del 40% del total de los créditos de la carrera respectiva.

Artículo 9: Los aspirantes a ingresar a la UNAM que sean admitidos, adquirirán la condición de alumnos con todos los derechos y obligaciones que establecen las leyes, reglamentos y disposiciones de la Universidad, y una vez inscritos, recibirán un registro de las asignaturas que cursarán con sus grupos correspondientes y, para efectos de identificación, deberán obtener su credencial, conforme al procedimiento que para el efecto se establezca.

Artículo 13: Los aspirantes a cursar estudios posteriores a la licenciatura solicitarán su inscripción en la división de estudios superiores correspondiente, de acuerdo con el reglamento respectivo y con sus normas internas de trabajo y, serán admitidos mediante un concurso de selección, que podrá implicar:

a) Examen y revalidación o reconocimiento de antecedentes.

b) Examen de clasificación.

c) Práctica previa.

d) Otros requisitos, según las normas internas de las divisiones de estudiantes superiores aprobados por los consejos técnicos.

Artículo 16: No podrán cursarse dos carreras simultáneamente, salvo que:

a) El cupo de los planteles lo permita.

b) El solicitante haya cubierto, por lo menos, el 50% de los créditos de la primera carrera.

c) Haya obtenido en las asignaturas acreditadas en la primera carrera un número de calificaciones MB mayor o igual al número de calificaciones S.

Artículo 18: Los cambios de carrera o de unidad académica se concederán siempre que el cupo de los planteles lo permita, de acuerdo con las siguientes bases:

a) En las carreras de la misma área del conocimiento (únicamente en las ENEPs se interpreta "área del conocimiento" como carreras que se inician con el mismo plan de estudios, es decir, con un tronco común), dentro de una misma facultad o escuela, bastará el acuerdo escrito del director del plantel correspondiente.

b) En las mismas carreras de diferente plantel, se requerirá la autorización escrita del director del plantel aceptante.

c) En las carreras de diferente área del conocimiento, así como las de la misma área de diferente plantel, se requerirá ser aceptado mediante el mismo concurso de selección, al que deberán someterse quienes pretendan ingresar por primera vez al nivel profesional.

Artículo 19: Los límites de tiempo para estar inscrito en la Universidad serán:

a) 4 años para cada uno de los ciclos de bachillerato,

b) 50% adicional a la duración señalada en el plan de estudios respectivo en el ciclo de licenciatura,

c) 50% como máximo de la duración establecida en el plan de estudios respectivo de las materias específicas en las carreras cortas.

Estos términos se contarán a partir del ingreso al ciclo correspondiente, aunque se interrumpan los estudios. Los alumnos que no terminen sus estudios en los plazos señalados no serán reinscritos y sólo podrán acreditar las materias faltantes por medio de exámenes extraordinarios.

Artículo 20: Los alumnos que hayan interrumpido sus estudios podrán reinscribirse, en caso de que los plazos señalados, no hubieran concluido; pero tendrán que sujetarse al plan de estudios vigente en la fecha de su reingreso y, en caso de una interrupción mayor de tres años, deberán aprobar un examen global según lo establezca la facultad o escuela.

Artículo 27: Ningún alumno podrá ser inscrito más de dos veces en una misma asignatura. En caso de no acreditarla, sólo podrá hacerlo en un examen extraordinario, de acuerdo con lo dispuesto en el Capítulo III del Reglamento General de Exámenes.

Artículo 28: Los alumnos tendrán derecho a escoger los grupos a los que deseen ingresar, sin más limitación que el cupo señalado por las autoridades competentes.

Artículo 29: Sólo se concederán cambios de grupo dentro de los quince días siguientes a la iniciación de cursos, si el cupo de los grupos lo permite.

Artículo 30: La Universidad señalará discrecionalmente el número de estudiantes extranjeros que podrán inscribirse en sus planteles. Los aspirantes, además de cumplir con los requisitos establecidos para los estudiantes nacionales, deberán satisfacer los que en particular se determine en los instructivos correspondientes.

Reglamento General de Exámenes

Artículo 4: Para fines de promedio se utilizará la siguiente conversión a la escala decimal:

MB	(Muy bien)	igual a 10
B	(Bien)	igual a 8
S	(Suficiente)	igual a 6
NA	(No acreditada)	carece de equivalente
NP	(No presentado)	carece de equivalente

Artículo 5: Los exámenes se realizarán de acuerdo con el calendario que establezca el Consejo Técnico y los horarios que fije el director de la facultad o escuela correspondiente, dentro de los períodos establecidos por el Consejo Universitario. El examen de cada materia deberá terminarse en un lapso máximo de siete días contados a partir de la fecha de su iniciación, y la documentación respectiva deberá remitirse a la Coordinación de la Administración Escolar en un período máximo de siete días a partir de la conclusión del examen.

Artículo 11: Habrá dos períodos de exámenes ordinarios: uno al término de los cursos correspondientes y otro antes del siguiente período lectivo. El estudiante podrá presentarse en cualquiera de esos períodos, o en ambos; pero si acredita la materia en alguno de ellos, la calificación será definitiva.

Artículo 14: Los exámenes extraordinarios tienen por objeto calificar la capacitación de los sustentantes que hayan acreditado las materias correspondientes cuando:

a) Habiéndose inscrito en la asignatura, no hayan llenado los requisitos para acreditarla.

b) Siendo alumnos de la Universidad, no hayan estado inscritos en la asignatura correspondiente, o no la hayan cursado.

c) Habiendo estado inscritos dos veces en una asignatura, no puedan inscribirse nuevamente.

d) Hayan llegado al límite de tiempo en que pueden estar inscritos en la Universidad.

Artículo 15: Los exámenes extraordinarios se efectuarán en los períodos señalados en el calendario escolar. Serán realizados por dos sinodales, que deberán ser profesores definitivos de la asignatura correspondiente o de una afín. En casos justificados los alumnos podrán solicitar por escrito, a la dirección

de la facultad o escuela correspondiente, que designe otro jurado.

Artículo 16: Los estudiantes tendrán derecho a presentar hasta dos materias por semestre mediante exámenes extraordinarios. Solamente el secretario general de la Universidad podrá conceder un número mayor de exámenes extraordinarios, previo informe favorable de la dirección de la facultad o escuela y de la Coordinación de la Administración Escolar.

Artículo 18: Los objetivos de los exámenes profesionales y de grado son valorar en conjunto los conocimientos generales del sustentante en su carrera o especialidad y que éste demuestre su capacidad para aplicar los conocimientos adquiridos y que posee criterio profesional.

Artículo 19: En nivel licenciatura el título se expedirá a petición del interesado, cuando haya cubierto el plan de estudios respectivo y haya sido aprobado en el examen profesional correspondiente.

Artículo 24: Los jurados para exámenes profesionales y para obtener el grado de maestría se integrarán con tres sinodales. Si el consejo técnico así lo decide, este número podrá aumentarse hasta cinco en la facultad o escuela correspondiente. Para el grado de doctor los sinodales serán cinco.

Artículo 29: Al terminar el examen cada sinodal emitirá su voto; el resultado se expresará mediante la calificación: aprobado o suspendido.

Artículo 30: En caso de suspensión no se podrá conceder otro examen antes de seis meses.

Artículo 31: En examen de excepcional calidad, y tomando en cuenta los antecedentes académicos, el jurado podrá otorgar mención honorífica; que justificará por escrito ante el director de la facultad o escuela.

DEFINICIÓN DEL PROBLEMA

La ENEP Iztacala no cuenta con un sistema automatizado para apoyar los servicios que proporciona la Unidad de Administración Escolar, los cuales implican una gran cantidad de recursos humanos y materiales debido al número de datos que se genera; al automatizar las actividades se logrará un mejor uso de dichos recursos para así requerir menos tiempo en los procesos y actualización de la información.

PROCESOS CONTEMPLADOS EN ADESI

Es importante aclarar que este diagnóstico se hizo desde el punto de vista de la metodología para análisis de sistemas. Se muestra a continuación una tabla con las situaciones críticas de algunos procesos, explicando cada una de ellas y proponiendo una alternativa de solución.

Inscripciones

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Elaboración de relación de alumnos con documentos faltantes.	Registro manual de la relación de alumnos que no tienen completos sus documentos de inscripción.	Al automatizar el sistema de inscripciones a primer ingreso, se podrá generar automáticamente la relación que es necesaria para completar los documentos de todos los alumnos.
Elaboración de relación de alumnos no inscritos.	Se emite manualmente la relación de los alumnos que no concluyeron el trámite de inscripción.	El sistema, registrará a los alumnos que concluyan su trámite de inscripción, por ende podrá determinar quienes no lo hicieron.
Envío de la relación de alumnos no inscritos a DGAE.	Necesidad de dar de baja a los alumnos que no concluyeron su trámite de inscripción a primer ingreso y avisar a DGAE.	Enviar la relación a través de un archivo usando Red UNAM.
Realización de estadísticas en forma manual.	Necesidad de generar diversas estadísticas de inscripciones a partir de la información disponible.	Al automatizar el sistema se podrá generar en forma automática los cálculos necesarios para la elaboración de las estadísticas.

Reinscripciones

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Dependencia del historial académico para el trámite de reinscripción.	Necesidad de usar un documento oficial para que la sección de carrera pueda verificar la seriación de las asignaturas de cada alumno en su respectiva carrera.	Al automatizar el sistema de reinscripciones se encargará de verificar la seriación particular de cada una de las carreras.
Verificación de cupos en los grupos solicitados.	Necesidad de verificar el cupo en los grupos y ajustarlos en el momento de las reinscripciones, teniendo que llevar un registro manual del número de alumnos inscritos en cada uno de los grupos disponibles.	Al automatizar el sistema se eliminará el control manual de los alumnos inscritos en cada grupo.
Saturación de grupos.	Crear nuevos grupos o tener que ampliar el cupo de los mismos en el momento de las reinscripciones.	Basándose en estadísticas cuya generación sea más rápida se podrá reducir el número de grupos que se crean por saturación.
Elaboración manual de estadísticas.	Se requiere elaborar diversas estadísticas de reinscripción.	Las estadísticas podrán ser emitidas en forma automática con datos actualizados, debido al registro que el sistema hará de todas las reinscripciones realizadas.

Bajas Definitivas Voluntarias

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Solicitud del alumno de realizar baja definitiva voluntaria.	Manualmente se registra la baja de cada alumno que lo solicita.	Utilizando un sistema en línea, el alumno estará dado de baja en el momento en que lo solicite.
Envío de la relación de alumnos dados de baja a DGAE.	Necesidad de enterar a DGAE de los alumno dados de baja.	Transferir el archivo con los alumnos dados de baja, vía Red UNAM.
Elaboración manual de estadísticas de alumnos dados de baja.	Necesidad de generar estadíticas de bajas definitivas voluntarias.	Al llevar un registro de las bajas se podrá generar automáticamente las estadísticas con datos actualizados.
Entrega de un comprobante de baja definitiva voluntaria.	Se elabora de forma manual un documento para el alumno que demuestre la realización de este trámite.	Basándose en la información previamente registrada el sistema imprimirá el comprobante de baja.

Otros Ingresos

Ingresos a Años Posteriores

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Registro de los alumnos aceptados a ingresar a años posteriores.	Necesidad de conocer cuales asignaturas le fueron revalidadas, acreditadas o covalidadas a cada alumno aceptado.	Con un sistema automatizado y en línea, después de registrar las asignaturas del alumno, éste podra realizar cualquier trámite, contando con datos vitales actualizados tales como la seriación.
Expedición de un comprobante de ingreso a años posteriores.	Se emite en forma manual un documento que avala la aceptación del alumno a años posteriores.	El sistema será capaz de imprimir tal documento una vez registrados los datos necesarios.
Realización de estadísticas.	Se elaboran, en forma manual, estadísticas de ingreso a años posteriores.	El registro de los alumnos aceptados a años posteriores podrá ser consultada, teniendo la seguridad de que la información es correrta.

Carrera Simultánea

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Solicitud del alumno de cursar una carrera simultánea.	Necesidad de verificar que el alumno esté en condiciones de cursar carrera simultánea.	Gracias a que el sistema conocerá todos los datos de los alumnos, automáticamente se sabrá si el alumno puede o no cursar carrera simultánea.
Envío de la relación de alumnos aceptados a cursar carrera simultánea a DGAE.	Se requiere eleborar una lista de los alumnos que estarán cursando carrera simultánea.	Transferir el archivo con los alumnos aceptados a carrera simultánea vía red UNAM.
Realización de estadísticas de carrera simultánea.	Se elaboran, en forma manual, estadísticas de carrera simultánea.	Al registrar a los alumnos que solicitan y que cubren los requisitos para presentar carrera simultánea, el sistema podrá emitir las estadísticas en forma automática.

Segunda Carrera

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Solicitud del alumno de cursar una segunda carrera.	Consultar si el solicitante se puede inscribir a una segunda carrera, es decir, que ya haya finalizado la primera.	El sistema, al conocer la historia académica de los alumnos, inmediatemente informará si se cubre o no con los requisitos.
Envío de una relación de los alumnos registrados a segunda carrera a DGAE.	Se elabora manualmente una relación de quienes fueron aceptados a cursar una segunda carrera.	Transferir el archivo con los alumnos aceptados a carrera simultánea vía red UNAM.
Realización de estadísticas de segunda carrera.	Se elaboran, en forma manual, estadísticas de segunda carrera.	Al registrar a los alumnos que solicitan y que cubren los requisitos para cursar una segunda carrera, el sistema podrá emitir las estadísticas en forma automática.

Exámenes Extraordinarios

Hasta Dos Exámenes Extraordinarios

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Solicitud de inscripción exámenes extraordinarios por parte del alumno.	Se revisa en forma manual las asignaturas a que tiene derecho el alumno utilizando el historial académico y las actas de calificación.	La revisión puede hacerse de manera automática por el sistema, ya que éste conocerá la seriación de cada una de las carreras, y la historia académica del alumno.
Elaboración de la relación de alumnos inscritos a exámenes extraordinarios.	Se requiere saber qué alumnos se han inscrito a cada asignatura en examen extraordinario.	El sistema estará en condiciones de emitir en forma automática esta relación porque contará con los registros necesarios.
Elaboración de estadísticas de inscripción a examen extraordinario.	Necesidad de emitir diversas estadísticas que reflejen la demanda de exámenes extraordinarios.	El sistema contará con un registro de todos los alumnos inscritos a exámenes extraordinarios que permitirá obtener dichas estadísticas.

Más de Dos Exámenes Extraordinarios

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Solicitud de inscripción a más de dos exámenes extraordinarios por parte del alumno.	Se revisa en forma manual las asignaturas a que tiene derecho el alumno utilizando el historial académico y las actas de calificación.	La revisión puede hacerse de manera automática por el sistema, ya que éste conocerá la seriación de cada una de las carreras y la historia académica del alumno.
Autorización de más de dos exámenes extraordinarios.	La autorización es realizada por la Unidad de Administración Escolar y se entrevista a cada alumno.	El sistema consultará los datos del alumno y podrá orientar a quien tiene la facultad de autorizar.
Elaboración de estadísticas de inscripción a más de dos exámenes extraordinarios.	Necesidad de emitir diversas estadísticas relacionadas con las inscripciones a más de dos examenes extraordinarios.	El sistema contará con un registro de todos los alumnos inscritos a más de dos examenes extraordinarios que permitirá obtener las estadísticas.

Cambios

Cambios de Plantel

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Solicitud del alumno de realizar cambio de plantel.	Se registra a cada uno de los alumnos que desean tramitar cambio de plantel.	El sistema agilizará el registro de cambios de plantel.
Autorización de los cambios de plantel,	La Unidad de administración escolar autoriza o rechaza las solicitudes de cambio de plantel.	El sistema proporcionará un modo más sencillo de autorizar los cambios de plantel, y eliminará registros manuales.
Elaboracióna de una relación de solicitudes de cambio de plantel.	En forma manual por cada solicitud autorizada.	Basándose en la información previamente registrada, el sistema imprimirá automáticamente el comprobante.
Realización de estadísticas de cambios de plantel.	Necesidad de elaborar manualmente estadísticas de cambios de plantel.	Al registrar a los alumnos que realizan cambio de grupo, el sistema podrá emitir las estadísticas en forma automática.

Cambios de Grupo

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Solicitud del alumno de cambiar de grupo.	Se registra a cada uno de los alumnos que desean tramiter cambio de grupo.	Utilizando un sistema en línea este registro podrá realizarse con mayor velocidad.
Autorización de los cambios de grupo.	La Unidad de administración escolar autoriza o rechaza cada una de las solicitudes de cambio de grupo.	El sistema proporcionará un modo más sencillo de autorizar, eliminando papeleo.
Entrega de un comprobante de cambio de grupo.	Este documento se elabora en forma manual por cada solicitud autorizada.	Basándose en la información previamente registrada, el sistema imprimirá automáticamente la relación de solicitudes
Realización de estadísticas de cambios de grupo.	Se elaboran, en forma manual, estadísticas de cambios de grupo.	Al registrar a los alumnos que realizan cambio de plantel, el sistema podrá emitir las estadísticas en forma automática.

Cambios de Plan de Estudios

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Solicitud del alumno de cambiar de plan de estudios.	Se revisa que el alumno aun esté a tiempo de realizar este trámite.	El sistema facilitará esta revisión y el cambio quedará registrado en el momento en que se solicite.
Realización de estadísticas de cambios de plan de estudios.	Necesidad de emitir, manualmente, estadísticas de cambios de plan de estudios.	El sistema podrá elaborar las estadísticas en forma automática, ya que contará con los datos necesarios para hacerlo.

Cambios de Carrera

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Solicitud del alumno de cambiar de carrera.	Se revisa, en forma manual, si el alumno está en condiciones de realizar cambio de carrera.	Utilizando un sistema en línea este registro podrá realizarse con mayor velocidad.
Se emite un listado con los alumnos que solicitaron cambio de carrera.	La Unidad de Administración Escolar emite un listado general y otro por carrera origen de los alumnos solicitantes.	El sistema generará automáticamente dichos listados.
Realización de estadísticas de cambios de carrera.	Se elaboran, en forma manual, estadísticas de cambios de carrera.	Al registrar a los alumnos que realizan la solicitud de cambio de carrera, el sistema podrá emitir las estadísticas en forma automática.

Cambios de Carrera

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Solicitud del alumno de cambiar de carrera.	Se revisa, en forma manual, si el alumno está en condiciones de realizar cambio de carrera.	Utilizando un sistema en línea este registro podrá realizarse con mayor velocidad.
Se emite un listado con los alumnos que solicitaron cambio de carrera.	La Unidad de Administración Escolar emite un listado general y otro por carrera origen de los alumnos solicitantes.	El sistema generará automáticamente dichos listados.
Realización de estadísticas de cambios de carrera.	Se elaboran, en forma manual, estadísticas de cambios de carrera.	Al registrar a los alumnos que realizan la solicitud de cambio de carrera, el sistema podrá emitir las estadísticas en forma automática.

Seguro Facultativo

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Verificación de que el alumno esté inscrito y de que no tenga seguro facultativo.	Necesidad de checar en el expediente del alumno que no tenga seguro facultativo y que esté inscrito.	El sistema se encargará de hacer estas verificaciones automáticamente cuando el alumno realice este trámite.
Registrar manualmente a los alumnos que solicitan el seguro y asimismo la entrega de comprobantes.	Se requiere llevar un registro manual de los alumnos que ya han tramitado el seguro facultativo y de los comprobantes que se han entregado.	El sistema se eliminará este control manual de los alumnos que han tramitado el seguro facultativo.
Elaboración manual de estadísticas.	Se requiere información estadística de los alumnos que han tramitado el seguro facultativo.	Al llevar un registro automático de este trámite, la generación de estadísticas se podrá realizar en el momento necesario con la información actualizada.

Control de Actas

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Registro manual de las actas que se reciben de DGAE.	Es necesario registrar y llevar el control manualmente de los folios de las actas que envía DGAE para los diferentes trámites.	El sistema permitirá registrar las actas que se reciban, así como el tipo, grupo, asignatura, profesor y alumnos que incluye el acta.
Registro manual de las actas, ya llenadas, que son entregadas por los profesores a la Unidad de Admon. Escolar.	Se requiere llevar un control de las actas que ya fueron entregadas por los profesores, para que de esta manera se pueda avisar a las jefaturas de carrera de las actas faltantes y a qué profesor pertenecen.	Se llevará un registro automático, el cual facilitará el control de las actas y podremos saber en cualquier momento cuáles son las actas que faltan por entregar.
Elaboración manual de las relaciones de actas que son devueltas a DGAE.	Es necesario elaborar una relación de las actas que son enviadas nuevamente a DGAE.	El sistema elaborará automáticamente estas relaciones y serán transmitidas a DGAE a través de la redunam.

Tramita Titulación

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Dependencia del historial académico para el trámite de revisión de estudios.	Es necesario verificar con el historial académico del alumno que ya tenga cubierto el 100% de créditos.	El sistema validará, en el momento de registrar el trámite, si el alumno ya cubrió el 100% de crédito, evitando así, pérdida de tiempo y que el alumno presente el historial académico.
Registrar la solicitud al exámen profesional.	Se lleva a cabo un registro manual de los alumnos que se inscriben al examen profesional oral. Asimismo, se lleva el registro de los alumnos que registran un trabajo de investigación.	Con la automatización de este proceso, se eliminará el registro manual, lo cual facilitará el trámite y se ahorrará tiempo.
Asignación de grupos, fechas y horarios, en el caso de exámenes objetivos. Para los exámenes orales, registrar fecha, hora, jurado y lugar.	Necesidad de registrar manualmente a los alumnos que realizan el trámite de examen profesional objetivo. Necesidad de registrar y asignar fecha, hora, jurado y lugar a los alumnos que solicitan el examen profesional oral.	El sistema facilitará el registro de los alumno y la asignación de fecha, hora, lugar y jurado de una manera automática evitando así que se asignen equivocadamente.
Elaboración manual de estadísticas.	Se requiere información estadística de los alumnos que tramitan el examen profesional objetivo u oral.	El sistema facilitará la consulta de estos trámites en el momento que sea necesario así como la emisión de estadísticas.

Registra Sanciones Académicas

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Se registra manualmente la sanción en el expediente del alumno.	En caso de que el alumno cometa alguna falta será sancionado, para lo cual es necesario registrar la sanción en su expediente. Dicha sanción está en función a la falta que cometió.	Se registrará dicha sanción en el sistema, el cual se encargará de verificar en el momento que se realice cualquier otro trámite el tipo de la sanción y dependiendo de éste será posible o no continuar con el trámite solicitado.

Emite Documentos

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
El alumno solicita la reposición del historial académico, del comprobante de inscripción, de la credencial o de alguna constancia.	Dependiendo del documento que está solicitando el alumno, se revisa su expediente y se elabora manualmente el documento.	El sistema verificará automáticamente los datos necesarios para la emisión del documento solicitado y lo imprimirá.
Emisión de etiquetas para abrir los expedientes de los alumnos.	Es necesario elaborar manualmente una etiqueta por cada alumno en la cual, se anotan los datos personales y el número de cuenta del alumno.	Al solicitarle la emisión de etiquetas al sistema, éste se encargará de imprimirlas automáticamente.

Emite Estadísticas

PROCESO	EXPLICACIÓN	SOLUCIÓN (ADESI)
Emitir estadísticas de los diferentes procesos.	Elaborar manualmente las estadísticas de los procesos para la toma de decisiones.	Al llevar un registro automatizado de la información involucrada en los procesos, se facilitará la emisión de estadísticas, ya que se llevará a cabo, en el momento solicitado y con la información actualizada.

Por otra parte, es innegable que las demoras derivadas del tiempo de transporte de documentos entre la ENEP y DGAE, dada la distancia entre ambas, sería ampliamente superado si se intercambiasen archivos utilizando para ello la infraestructura de RedUNAM.

VENTAJAS Y DESVENTAJAS

Ventajas

- Disminuir las cargas excesivas de trabajo del personal de la Administración Escolar.

- Emitir los documentos necesarios con información actualizada en el momento en que sean requeridos.

- Consultar en línea la información de los alumnos, desde cualquier terminal o estación de trabajo de la ENEP Iztacala, siempre y cuando se cuente con la clave de acceso apropiada.

- Contar con información actualizada, confiable y oportuna, y de esta manera poder realizar las actividades de la Administración Escolar optimizando tiempos.

- Contar con una herramienta que facilite la coordinación de las actividades de la Administración Escolar, y de esta forma descentralizar el control sobre la información de los alumnos.

- Obtener resultados en línea, dando soluciones a los alumnos en el momento en que realizan el trámite.

- Obtener en línea los diferentes reportes y estadísticas que se emiten, para apoyar la toma de decisiones.

- Contar con una infraestructura, que en un futuro facilite a la ENEP Iztacala continuar con el proyecto de automatización de las diversas funciones que desempeñan en la misma institución.

- Disminuir la utilización de grandes cantidades de papel, así como el mobiliario para archivar los documentos, lo cual nos lleva a una reducción en los costos.

Desventajas

Algunas de las principales desventajas de la implantación de ADESI son:

- La posible resistencia del personal a la incorporación del sistema en su esquema de trabajo. (Se puede combatir dicha resistencia con pláticas de motivación y capacitación sobre el sistema).

- La relativa vulnerabilidad de los sistemas de cómputo en red, que puede ser salvada con medidas de administración consistentes. En cuanto a dichas medidas de administración y seguridad, el manejador de bases de datos (DBMS) deberá cumplir ciertos criterios de aceptación que contribuyan a la solución de esta posible desventaja.

COSTO DE ADESI

Antes de empezar a desarrollar el proyecto debemos realizar un análisis que nos muestre los beneficios que obtendremos del mismo. Para lo cual tendremos que calcular los costos en los que vamos a incurrir y analizar si son justificables. Dicho análisis tiene como objetivo el apoyar la decisión de llevar a cabo o no el desarrollo del proyecto.

En este trabajo presentamos el costo total en el que incurrimos al desarrollar el sistema y los beneficios más notables que proporcionará.

Beneficios

Dentro de los principales beneficios que obtendremos del sistema están:

Los diversos trámites que brinda la Administración Escolar se realizarán de manera más rápida y con una respuesta inmediata a las necesidades de los alumnos.

Se disminuye la carga de trabajo de la Unidad de Administración Escolar.

La Unidad de Administración Escolar contará con información confiable y actual para la toma de decisiones.

La emisión de estadísticas se llevará a cabo de manera inmediata y confiable, evitando así el desperdicio de papel, tiempo y esfuerzo.

A través de Red UNAM se podrá realizar el intercambio de información entre la ENEP y la DGAE ahorrando así tiempo valioso y facilitando los trámites.

Ahora presentamos los diversos costos en los que incurrimos para poder desarrollar el sistema:

COSTOS DE RECURSOS HUMANOS

			Sueldo		Lapso	
	Cantidad	Contrata	mensual	Total	(meses)	Total
Líder de proyecto	1	Base	2100.00	2100.00	7.00	14700.00
Analista programador	3	Base	1600.00	4800.00	7.00	33600.00
Jefe de soporte	1	Eventual	2100.00	2100.00	0.50	1050.00
Técnico de soporte	2	Eventual	1011.00	2022.00	0.50	1011.00
Secretaria	1	Base	950.00	950.00	1.00	950.00
Total						51311.00

COSTO DE RECURSOS MATERIALES

Concepto	Unidades	Precio Unitario	Total
Hojas blancas	2000	0.10	200.00
Lápices	8	0.50	4.00
Plumas	4	1.00	4.00
Folders	6	0.50	3.00
Carpetas	10	25.00	250.00
Diskettes	10	3.30	33.00
Toner	1	350.00	175.00
Fotocopias	3000	0.10	300.00
Cintas de 8 m.m.	2	30.00	60.00
Total			1029.00

COSTOS INDIRECTOS

Concepto	Costos
Agua	42.00
Luz	420.00
Teléfono	80.00
Total	542.00

COSTOS DE BIENES INMUEBLES

Concepto	Precio	Cantidad(m2)
Terreno		1200.00
Construcción		600.00
Renta	7100.00	
Renta x m2	11.83	
Area utilizada		55.00
Costo a aplicar	650.65	

REQUERIMIENTOS DE HARDWARE
(precios en dólares)

Concepto	Precio Unitario	Cantidad	Total
Sparc server 5 1.05 GB en disco duro 32 MB en RAM	9990	1	9990.00
Unidad de respaldo 8 GB	4000	1	4000.00
CD ROM	700	1	700.00
Computadora HP procesador 80486 SX a 25 MHz 4 MB en RAM 170 MB en disco duro Monitor VGA color 14"	1646	10	16460.00
Impresoras Epson FX-1170 9 agujas, 15", puerto paralelo, 8 KB, 240 cps	583	3	1749.00
Impresora HP LASER 4L minuto, HP PCL 5, 26 fuentes internas, 2 MB de memoria, alimentador de 100 hojas, puerto paralelo	802	1	802.00
UPS Deltec 2.4 KVA	3948	1	3948.00
Concentrador cabletron MRXI-e de 12 puertos	1619	1	1619.00
Computadora HP procesador 80386 SX a 25 MHz, 4 MB RAM, 120 MB en disco duro (servidor de impresión)	1400	1	1400.00
Tarjetas de red	163	12	1956.00
Total			50854.00

REQUERIM. DE SOFTWARE (precios en dólares)

Concepto	Precio
SQL server (16 usuarios)	3769.60
Data Workbench (8 usrs.)	1188.00
Net Libraries (15 usrs.)	435.00
Visual Basic Professional	34.57
SQL Report Writer	85.00
Total	5512.17

RESUMEN DE COSTOS

Concepto	Costo a Aplicar
Costo Rec. Humanos	N$51311.00
Costo Rec. Materiales	N$1029.00
Costos Indirectos	N$542.00
Costos Bienes Inmuebles	N$650.65
Costo de Hardware	dólares 50854.00
Costo del Software	dólares 5512.17

Debemos considerar que si el costo del sistema es alto, se debe en parte a que la ENEP no cuenta con equipo de cómputo, por lo que estamos partiendo de una base de cero en cuanto al hardware; asimismo, este equipo podrá ser usado en la implantación de otros sistemas de cómputo que se desarrollen para la ENEP. Cabe mencionar que una de las ventajas de la adquisición de Sybase, además de las que se mencionan en el capítulo V, es que esta compañía ofrece a la UNAM un decuento del 80% en sus productos, mismo que ya se ve reflejado en los cuadros anteriores.

CAPÍTULO IV
ANÁLISIS

INTRODUCCIÓN

El análisis es una etapa fundamental en el desarrollo de sistemas, su importancia radica en que el desarrollador recopila toda la información con que trabaja el sistema, por un lado obtiene los diversos documentos de entrada y salida, y por otro, realiza entrevistas con el usuario para tener una descripción de cada uno de los procesos que componen al sistema. Toda esta información debe ser plasmada en diagramas y demás documentos de análisis que tienen por objetivo expresar de forma lógica el funcionamiento del sistema.

En la actualidad existen diversas metodologías para llevar a cabo un análisis de este tipo, en nuestro caso, decidimos hacer uso del análisis estructurado.

El análisis de ADESI está dividido, de acuerdo a la metodología, en tres partes principales:

- **Modelo Ambiental.** Se compone por una definición breve pero clara del objetivo del sistema, un diagrama de contexto en donde se representa la relación que tiene el sistema con entidades del exterior y por una lista de eventos, en donde cada evento es un estímulo que hace que el sistema responda.

- **Modelo de Comportamiento.** Lo integran un conjunto de Diagramas de Flujo de Datos que modelan cada proceso del sistema, describiendo como se realiza el flujo y almacenamiento de los datos, aquí cabe señalar que para los fines del presente trabajo, únicamente se descendió a un nivel de detalle en los módulos principales de ADESI, éstos son: Tramita Registro, Exámenes Extraordinarios y Cambios. El Modelo de Comportamiento también se compone por Miniespecificaciones en las que, a nivel de pseudocódigo, se describe el trabajo que cada proceso primitivo debe realizar, finalmente se presenta el Diccionario de Datos en el que se define la composición de cada elemento utilizado en el análisis.

- **Modelo de Información.** En este modelo se presenta el Diagrama de Entidad - Relación, mismo que ilustra cuáles son y cómo se relacionan las entidades que forman parte de ADESI. Por último, se presenta una descripción detallada de los atributos de cada entidad.

MODELO AMBIENTAL

Yourdon establece que el primer modelo importante que se debe desarrollar como analista es uno que no haga mas que definir las interfaces entre el sistema y el resto del universo, es decir, el ambiente.

Además de determinar qué está en el interior del sistema y qué en el exterior, también es importante definir qué información entra al sistema desde el ambiente exterior, y qué información produce como salida al ambiente externo. Ningún sistema de información toma todos los datos disponibles en el universo, ni produce salidas al azar.

"Los sistemas tiene como propósito producir salidas como respuesta a algún acontecimiento o estímulo en el ambiente. Así, otro aspecto crítico del modelo ambiental consiste en identificar los acontecimientos que ocurren en el ambiente al cual debe responder el sistema. Sólo nos preocupan aquellos que ocurren en el ambiente exterior y requieren una respuesta del sistema." [YOU93]

Este modelo define el alcance del sistema. Considerando que un sistema es un conjunto de respuestas planteadas a eventos que ocurren en el ambiente, necesitamos definirlo, es decir, la razón por la que el sistema existe (objetivo del sistema), lo que está afuera del sistema y con lo que hay una interacción (diagrama de contexto), y lo que sucede y provoca que el sistema responda (lista de eventos).

El modelo ambiental está formado por tres componentes:

EL OBJETIVO DEL SISTEMA

"Es una declaración textual, breve y concisa del propósito del sistema, dirigida al nivel administrativo superior, la administración de los usuarios, y otros que no están directamente involucrados con el desarrollo del sistema." [YOU93]

El objetivo del sistema describe porqué el sistema existe y por lo tanto, proporciona una guía para definir tanto el ambiente como las respuestas del sistema. Puede constar de una, dos o varias frases pero no debe llegar a más de un párrafo, ya que la intención no es proporcionar una descripción completa y detallada del sistema.

Objetivo del Sistema ADESI

Contribuir con la Escuela Nacional de Estudios Profesionales de Iztacala, automatizando los procedimientos que integran a la Administración Escolar, buscando así apoyar los servicios escolares para brindar una atención más rápida al alumnado, facilitando el control y manejo de los procesos para favorecer la toma de decisiones.

DIAGRAMA DE CONTEXTO

El diagrama de contexto es un caso especial del diagrama de flujo de datos, en donde un solo círculo representa todo el sistema y los rectángulos son las entidades externas con las que el sistema tiene alguna relación.

El diagrama de contexto enfatiza varias características importantes del sistema:

- Las personas, organizaciones y sistemas en el ambiente con los que se comunica el sistema, se conocen como terminadores y se representan con un rectángulo.
- Los datos que el sistema recibe del mundo exterior y que deben procesarse de alguna forma.
- Los datos que el sistema produce y que se envían al mundo exterior.
- Los archivos de datos que el sistema comparte con los terminadores.
- La frontera entre el sistema y el resto del mundo.

El diagrama de contexto no describe los detalles internos del sistema.

Notación del Diagrama de Contexto

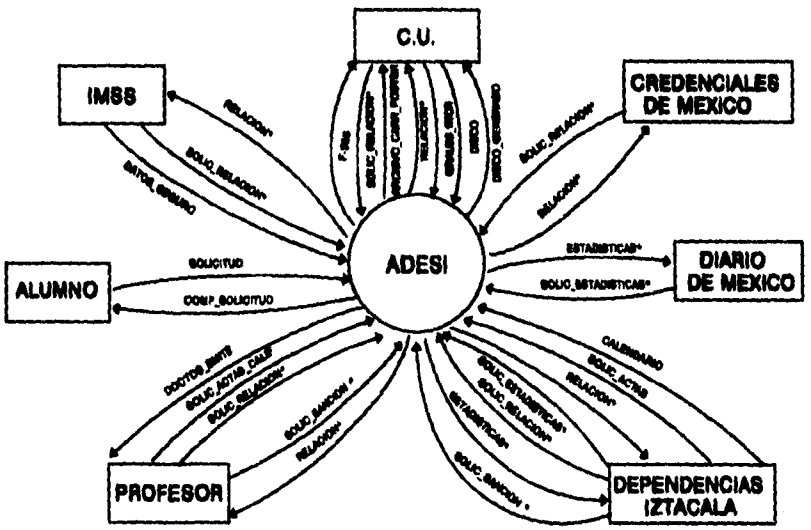
Entidad: es la gente y/u otros sistemas con los que hay comunicación.

Proceso sencillo: nombrado para describir lo que el sistema hace.

Flujos: muestran las conexiones de datos entre una entidad y el sistema, provocando una respuesta inmediata.

Archivos: muestran datos almacenados por una entidad y que son necesitados después por el sistema o viceversa.

Diagrama de Contexto



LISTA DE EVENTOS

"La lista de eventos es una lista narrativa de los estímulos que ocurren en el mundo exterior a los cuales el sistema debe responder." [YOU93]

La lista de eventos describe los acontencimientos o sucesos que ocurren en el ambiente y que ocasionan que el sistema responda.

Debe contener las cosas que ocurren en el ambiente y que provocan que el sistema responda y quién o qué inició el evento.

Lista de Eventos

1. C.U. envía relación de alumnos aceptados a inscripción.
2. Alumno solicita inscripción.
3. C.U. envía documentos de inscripción.
4. C.U. solicita relación de alumnos activos.
5. Profesor solicita relación de alumnos inscritos por grupo.
6. Alumno solicita examen extraordinario.
7. C.U. solicita relación de alumnos inscritos a examen extraordinario.

8. Dependencias Iztacala envía calendario de exámenes extraordinarios.

9. Dependencias Iztacala solicita relación de pago a sinodales.

10. Profesor solicita relación de alumnos inscritos a examen extraordinario.

11. Alumno solicita rectificación de calificación.

12. Alumno solicita cambio de plantel primer ingreso.

13. C.U. solicita relación de solicitudes de cambio de plantel primer ingreso.

14. Alumno solicita cambio de plantel reingreso.

15. C.U. solicita relación de solicitudes de cambio de plantel reingreso.

16. Alumno solicita cambio de grupo.

17. Alumno solicita cambio de ciclo.

18. Alumno solicita cambio de carrera.

19. C.U. solicita relación de solicitudes de cambio de carrera.

20. Alumno solicita cambio de plan de estudios.

21. C.U. envía actas.

22. Profesor solicita actas.

23. Profesor entrega actas calificadas.

24. C.U. solicita actas calificadas.

25. Alumno solicita baja definitiva.

26. Alumno solicita constancia de estudio.

27. Alumno solicita reposición de credencial.

28. Credenciales de México solicita relación de solicitudes de reposición de credencial.

29. Alumno solicita reposición de historia académica.

30. Alumno solicita reposición de comprobante de inscripción.

31. Alumno solicita actualización de sus datos personales.

32. Dependencia Iztacala solicita sanción académica al alumno.

33. Alumno solicita examen profesional.

34. Dependencias Iztacala solicita relación de alumnos inscritos a examen profesional objetivo.

35. C.U. solicita relación de alumnos inscritos a examen profesional.

36. Alumno solicita citatorio de examen profesional.

37. Alumno solicita revisión de estudios.

38. Alumno solicita ingreso a años posteriores.

39. C.U. envía relación de alumnos que solicitan ingreso a años posteriores.

40. Alumno solicita carrera simultánea.

41. C.U. solicita relación de alumnos que solicitaron carrera simultánea.

42. Alumno solicita segunda carrera.

43. C.U. solicita relación de alumnos que solicitaron segunda carrera.

44. Alumno solicita seguro facultativo.

45. IMSS solicita listado de alumnos que solicitaron seguro facultativo.

46. Diario de México solicita relación de alumnos sobresalientes.

47. C.U. solicita relación de alumnos sobresalientes.

48. La jefatura de administración escolar solicita al departamento correspondiente las estadísticas de inscirpciones y reinscripciones.

49. La jefatura de administración escolar solicita al departamento de estadísticas las estadísticas de exámenes extraordinarios.

50. La jefatura de administración escolar solicita al departamento de estadísticas las estadísticas de otros ingresos a la UNAM.

51. La jefatura de administración escolar solicita al departamento de estadísticas las estadísticas de titulación.

52. La jefatura de administración escolar solicita al departamento de estadísticas las estadísticas varias.

MODELO DE COMPORTAMIENTO

Posterior al modelo ambiental debemos construir el modelo de comportamiento. Esto involucrará el desarrollo de diagramas de flujo de datos, miniespecificaciones para los procesos primitivos y el diccionario de datos.

"El modelo de comportamiento describe el comportamiento que del sistema se requiere para que interactúe de manera exitosa con el ambiente." [YOU93]

Esto implica dibujar el borrador del diagrama de flujo de datos, con procesos para la respuesta del sistema, balanceados ante cada acontecimiento que se identificó en la lista de eventos. A continuación se dibujarán los archivos y finalmente se conectarán los flujos de entrada y salida apropiados a los procesos y se comparará el conjunto de diagramas de flujo de datos con el diagrama de contexto para asegurar la consistencia.

Al tener todos nuestros diagramas de flujo de datos nivelados procederemos a realizar las miniespecificaciones únicamente de los procesos primitivos, es decir, de aquellos en los que ya no hayamos descendido otro nivel. En paralelo debemos elaborar el diccionario de datos.

DIAGRAMAS DE FLUJO DE DATOS

El trabajo básico realizado por los sistemas es la transformación de entradas en salidas.

"El diagrama de flujo de datos es una técnica gráfica que representa el flujo de la información y las transformaciones que se aplican a los datos al moverse desde la entrada hasta la salida." [PRE93]

El diagrama de flujo de datos es una herramienta que exhibe un sistema como una red de procesos conectados uno al otro por flujos de datos. El propósito es poner atención en los datos y en las transformaciones. El control, la secuencia o los algoritmos procedurales no deben aparecer en estos diagramas.

Los diagramas de flujo de datos están formados por procesos, flujos de datos y archivos.

Un proceso cambia flujos de entrada en flujos de salida, está representado por un círculo y su nombre debe describir lo que el proceso hace.

El flujo representa un paquete de datos en movimiento, es representado por una flecha que muestra la dirección del movimiento del paquete. Un paquete puede ser de datos o materiales. Cada flujo puede contener sólo un paquete, el nombre del flujo debe describir el significado del paquete.

El archivo es una colección de paquetes (registros) que están en reposo. Se representa por un par de líneas paralelas que delimitan el nombre del arhcivo.

Existen ciertas convenciones para los diagramas de flujo de datos, las cuales deben facilitar las tareas de creación y revisión de los mismos.

- Cada proceso debe tener al menos un flujo de datos que entra a él. Puede ser un flujo que venga de otro proceso o un flujo proveniente de un archivo.

- Cada proceso debe producir al menos un flujo de datos para otro proceso, para un archivo o como salidas del sistema. Asimismo, deben producir nuevos datos o hacer modificaciones.

- Cuando los datos son creados por un proceso y utilizados por otro, pero no en forma inmediata, deben viajar a un archivo, de donde serán tomados por el proceso que los necesita.

- No hay un límite técnico para el número de flujos que van o que salen de un proceso o de un archivo. Si esto sucede, para evitar la dificultad de usar el diagrama, se pueden agrupar flujos en un nivel superior, convirtiendo diferentes flujos en uno sólo. No importa que tan complicado sea el sistema real, el modelo que nosotros manejamos puede ser rediseñado para facilitarnos las cosas.

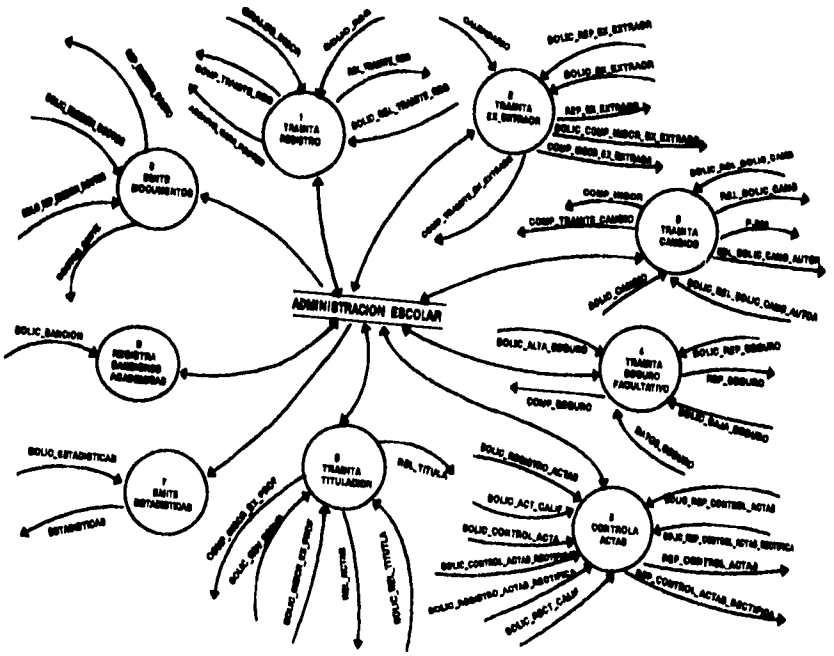
- Cuando un flujo llega a un archivo, esto significa que el archivo está siendo actualizado. Lo anterior implica que el archivo no es el mismo después que el proceso ha terminado.

- Cuando un flujo sale de un archivo podemos decir que se trata de un lectura.

- En el caso de una actualización a algún registro de un archivo, ésta se representa por una flecha hacia el archivo. Una flecha con doble punta significa que los datos que se han leído son modificados en el proceso y que se vuelven a escribir en el archivo.

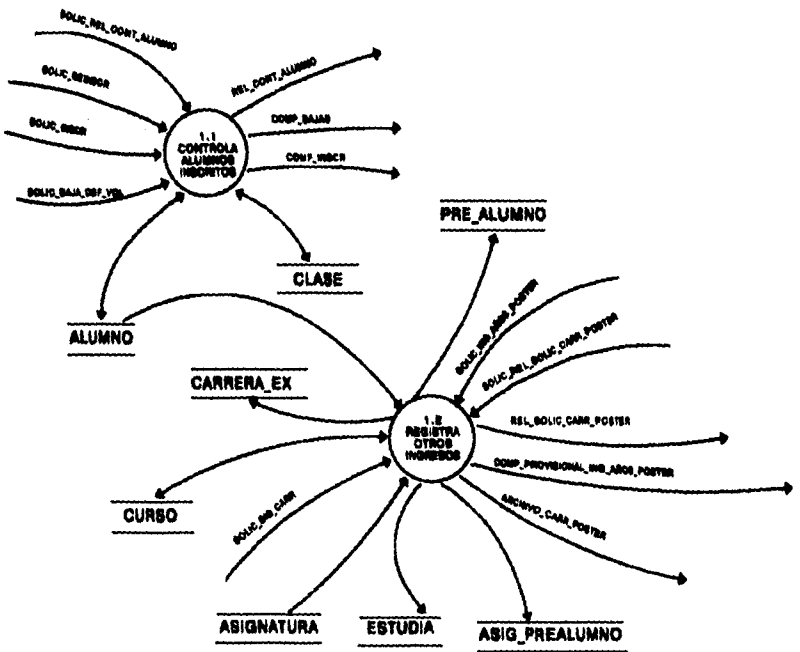
Aunque el diagrama de flujo de datos proporciona una visión global bastante conveniente de los componenetes funcionales del sistem, no da detalles de éstos. Para mostrarlos se ocupan dos herramientas textuales de modelado adicionales: las miniespecificaciones y el diccionario de datos.

D.F.D. 0


ADMINISTRACION ESCOLAR
1 TRAMITA INSCRITOS
2 EMITE DOCUMENTOS
3 TRAMITA EX_ENTRADOR
4 TRAMITA CAMBIOS
5 REGISTRA CAMBIOS ACADEMICOS
6 TRAMITA SEGURO FACULTATIVO
7 EMITE ESTADISTICAS
8 TRAMITA TITULACION
9 CONTROLA ACTAS

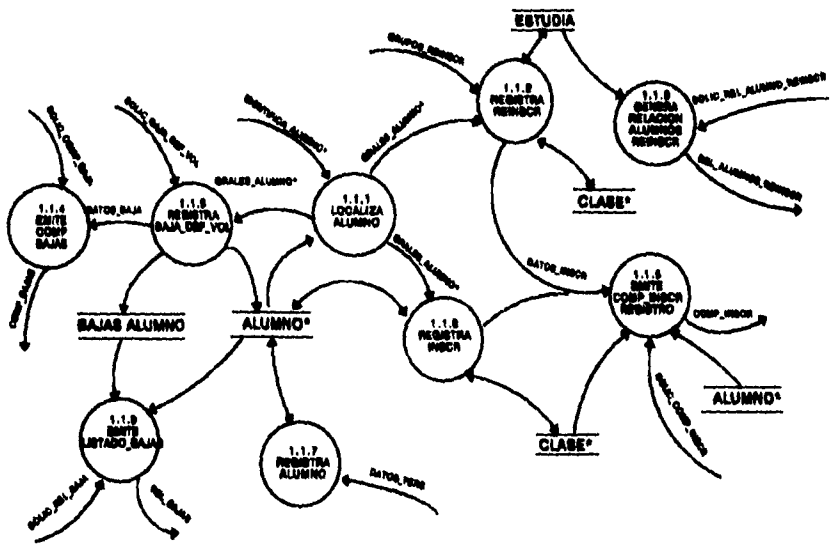
D.F.D. 1

TRAMITA REGISTRO



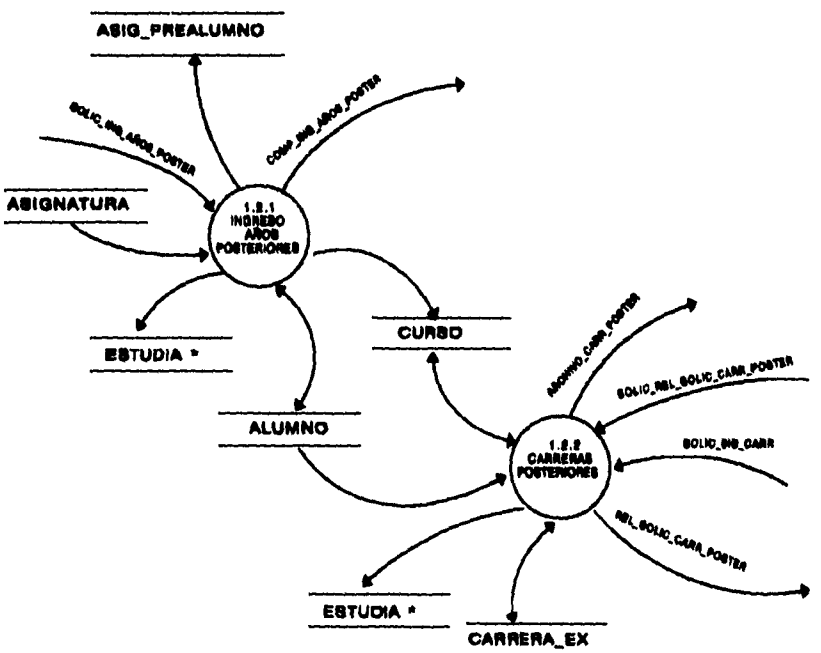
D.F.D. 1.1

CONTROLA ALUMNOS INSCRITOS



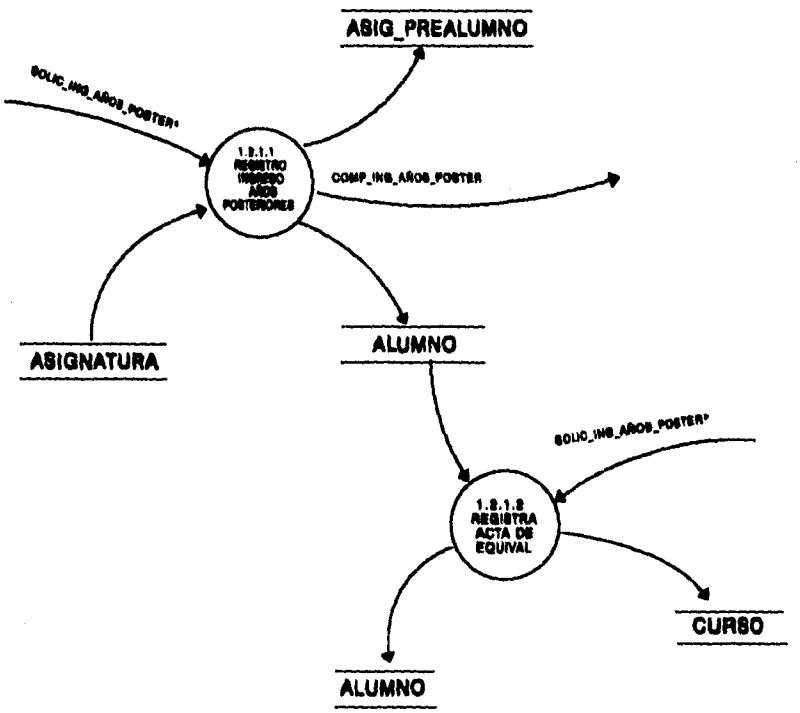
D.F.D. 1.2

REGISTRA OTROS INGRESOS



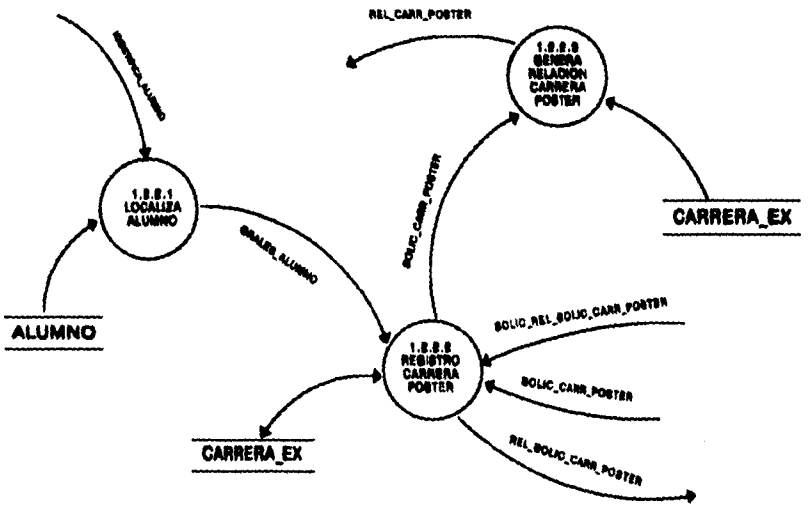
D.F.D. 1.2.1

INGRESO AÑOS POSTERIORES



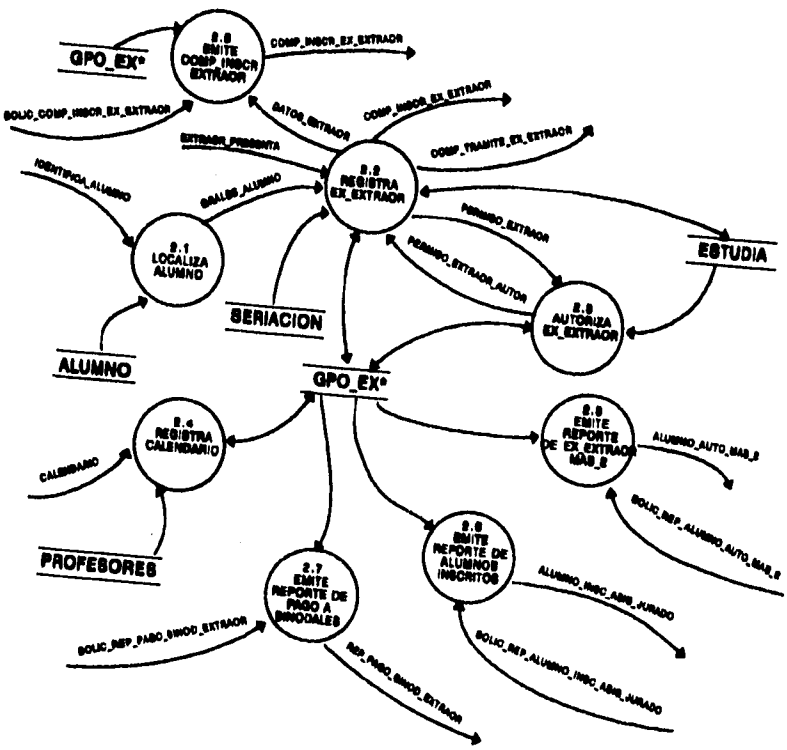
D.F.D. 1.2.2

CARRERAS POSTERIORES



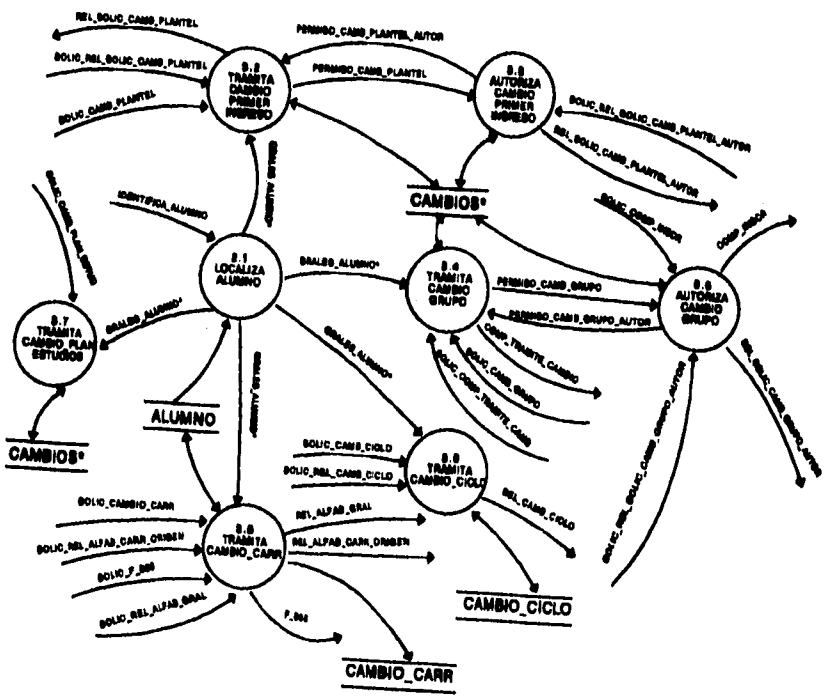
D.F.D. 2

TRAMITA EX_EXTRAOR



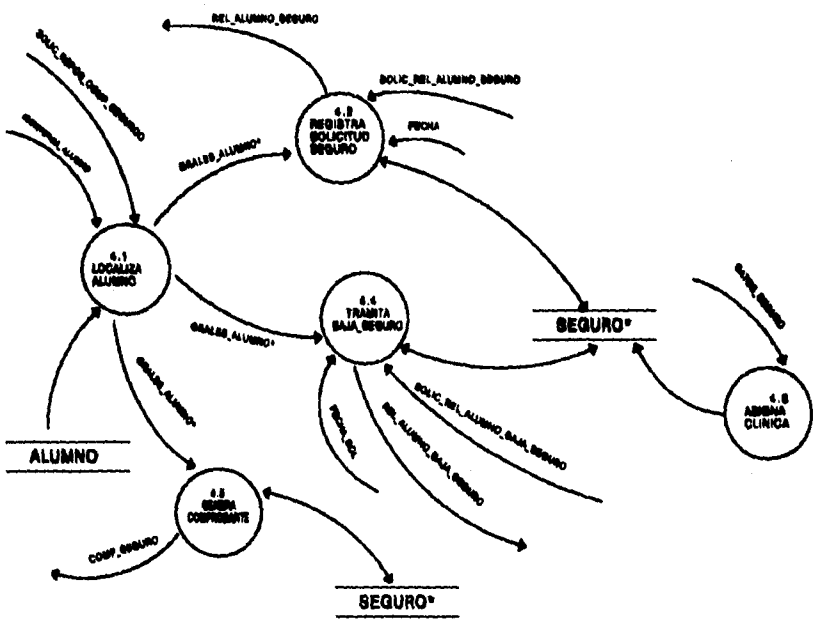
D.F.D. 3

TRAMITA CAMBIOS



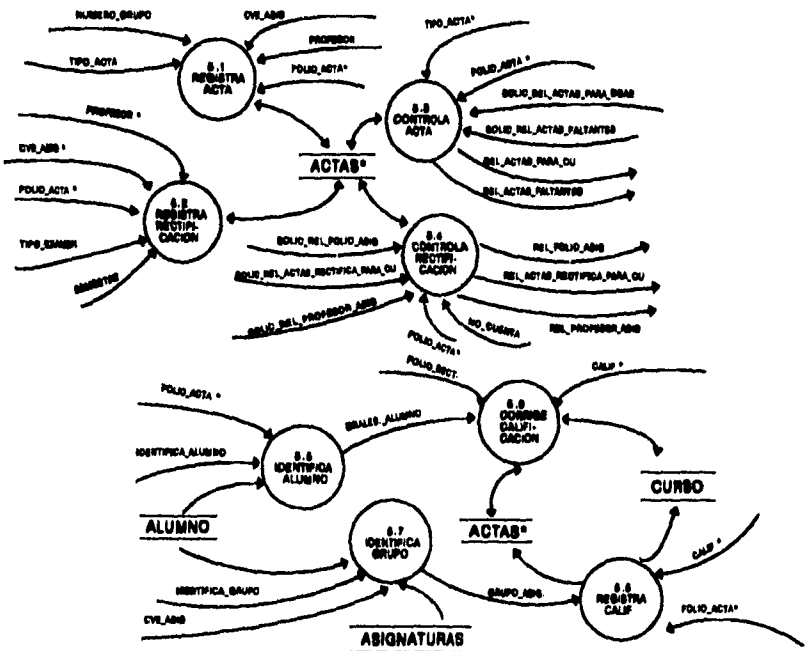
D.F.D. 4

TRAMITA SEGURO FACULTATIVO



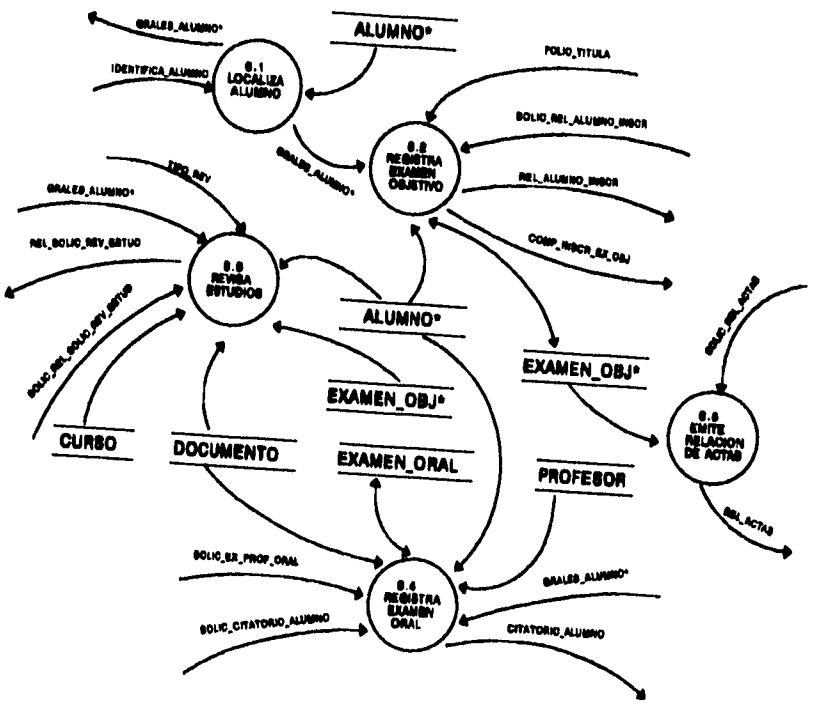
D.F.D. 5

CONTROLA ACTAS



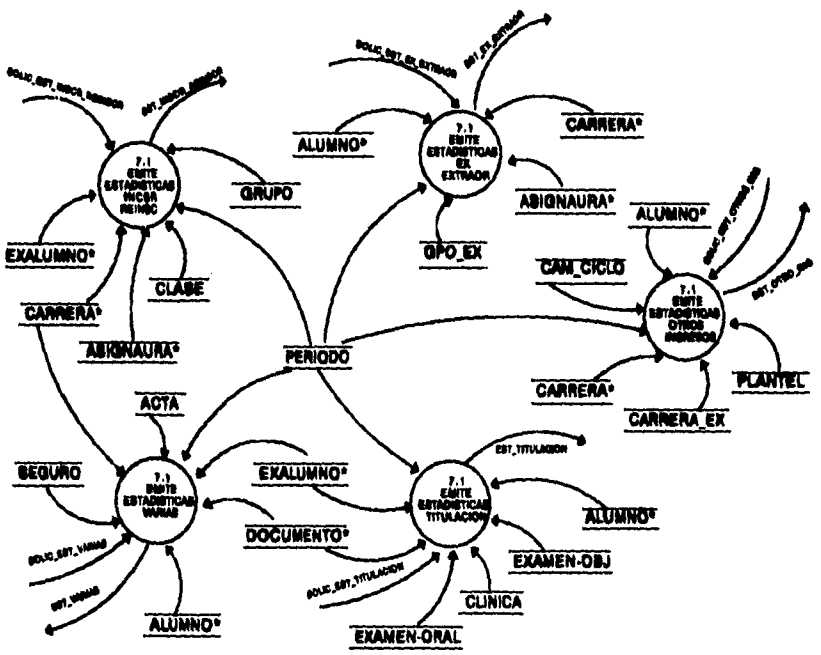
D.F.D. 6

TRAMITA TITULACIÓN



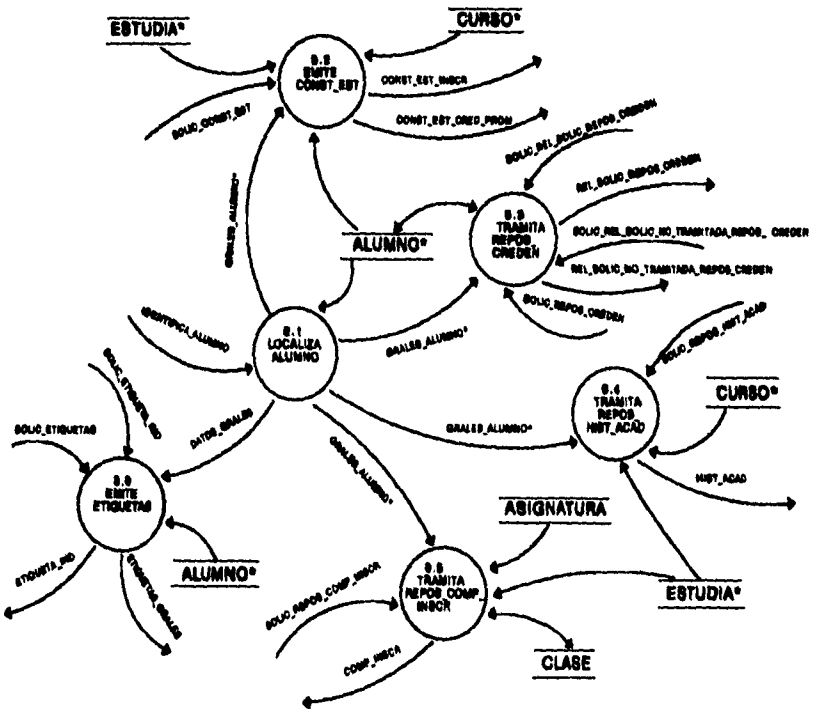
D.F.D. 7

EMITE ESTADÍSTICAS



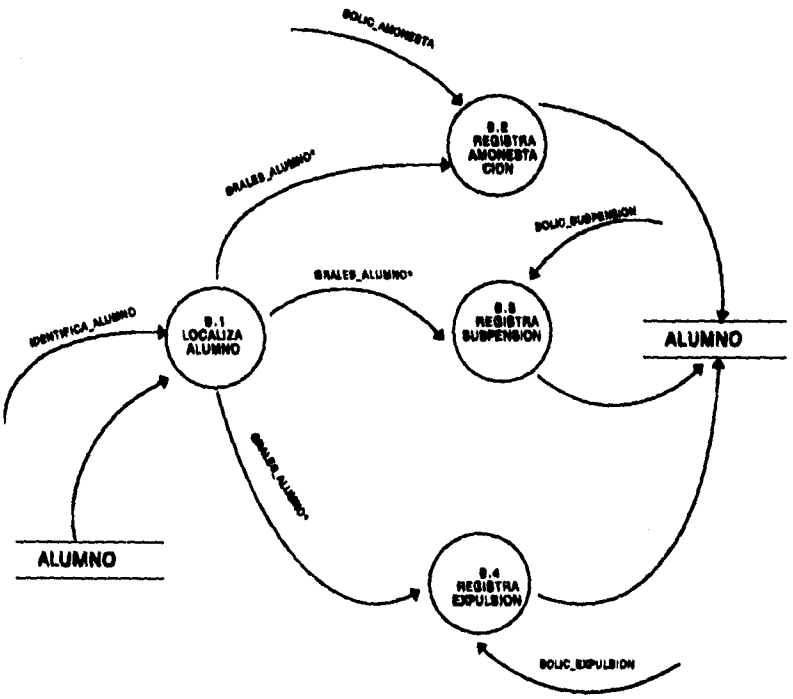
D.F.D. 8

EMITE DOCUMENTOS



D.F.D. 9

REGISTRA SANCIONES ACADÉMICAS



MINIESPECIFICACIONES

Una miniespecificación es la descripción de qué es lo que sucede en cada proceso primitivo de nivel más bajo en un diagrama de flujo de datos. Sin importar su nombre, el propósito de una miniespecificación es definir lo que debe hacerse para transformar entradas en salidas. [YOU93]

1.1.1, 1.2.2.1, 2.1, 3.1 LOCALIZA ALUMNO

```
Obtén NO_CTA y CVECARR
localiza en archivo alumno NO_CTA y CVECARR
si existe despliega GRALES_ALUMNO
sino despliega mensaje de error
fin si
```

1.1.2 REGISTRA REINSCR

```
Lee GRUPOS_REINSCR
obtén GRALES_ALUMNO
hacer mientras haya NUMERO_GRUPO y CVE_ASIG
 lee siguiente NUMERO_GRUPO y CVE_ASIG
 busca NUMERO_GRUPO y CVE_ASIG en CLASE
 si encontrado
   si vacantes > 0
      localiza en SERIACION las ASIGREQ
      verifica si el alumno tiene aprobadas las ASIGREQ
      si todas las ASIGREQ están aprobadas
```

```
          escribe vacantes = vacantes - 1 en CLASE
          escribe en CURSA NO_CUENTA, NUMERO_GRUPO,
CVE_CARR, CVEASIG
       si no
         manda mensaje de error "El alumno no ha aprobado
         asignaturas requisito"
       fin si
    si no
       manda mensaje de error "El grupo no tiene cupo"
    fin si
 si no
    manda mensaje de error "El grupo no existe"
 end si
fin hacer
escribe en ESTUDIA inscrito = S, colegiatura=colegiatura de
GRALES_ALUMNO
genera DATOS_INSCR
```

1.1.3 REGISTRA INSCR

```
Obtén GRALES_ALUMNO
lee GRUPO de PRIMER_INGRESO con
NO_CUENTA=NO_CUENTA de GRALES_ALUMNO
hacer mientras no sea fin de CLASE y GRUPO de CLASE=
GRUPO de PRIMER_INGRESO
  lee siguiente NUMERO_GRUPO y CVE_ASIG de CLASE
  si vacantes > 0
    valida seriación
    escribe vacantes = vacantes - 1 en CLASE
         escribe en CURSA NO_CUENTA, NUMERO_GRUPO,
CVE_CARR CVEASIG
```

si no
manda mensaje de error "El grupo no tiene cupo"
fin si
fin hacer
escribe en PRIMER_INGRESO INSCRITO = S, COLEGIATURA=COLEGIATURA de GRALES_ALUMNO
genera DATOS_INSCR

1.1.4 EMITE COMP_BAJAS

Obtén SOLIC_COMP_BAJA
Obtén DATOS_BAJA
imprime COMP_BAJAS

1.1.5 EMITE COMP_INSCR REGISTRO

Obtén DATOS_INSCR
Busca NO_CUENTA de DATOS_INSCR en ALUMNO
si encontrado
imprime COMP_INSCR
si no
manda mensaje de error "Alumno inválido"
fin Si

1.1.6 GENERA RELACION ALUMNOS REINSCR

Obtén SOLIC_REL_ALUMNO_REINSCR
hacer mientras no sea fin de ESTUDIA
leer siguiente ALUMNO
si inscrito = S

imprime ALUMNO
fin Si
fin hacer

1.1.7 REGISTRA ALUMNO

Obtén DATOS_PERS
escribe o Actualiza en ALUMNO DATOS PERS

1.1.8 REGISTRA BAJA_DEF_VOL

Obtén SOLIC_BAJA_DEF_VOL
obtén GRALES_ALUMNO
obtén SOLIC_COMP_BAJA
imprime ALUMNO donde NO_CUENTA = NO_CUENTA DE GRALES_ALUMNO
borra ALUMNO donde NO_CUENTA = NO_CUENTA DE GRALES_ALUMNO
escribe ALUMNO en BAJASALUMNO

1.1.9 EMITE LISTADO_BAJAS

Obtén SOLIC_REL_BAJA
hacer mientra no sea fin de BAJASALUMNO
leer siguiente registro
imprime registro
fin hacer

1.2.1 INGRESO AÑOS POSTERIORES

```
Obtén SOLIC_ING_AÑOS_POSTER
hacer mientras haya asignatura  cursada
 buscar asignatura cursada en ASIGNATURA
 si encontrada
  escribir asignatura cursada en CURSO
 si no
  manda mensaje de error "Asignatura inválida"
 fin si
fin hacer
escribir alumno en ALUMNO y en ESTUDIA
emitir COMP_ING_AÑOS_POSTER
```

1.2.2.2 REGISTRA CARRERA POSTER

```
Recibe GRALES_ALUMNO
Obtén SOLIC_CARR_POSTER
escribir SOLIC_CARR_POSTER + NO_CUENTA de
GRALES_ALUMNO en CARRERA_EX
obtén SOLIC_REL_SOLIC_CARR_POSTER
hacer mientras no sea fin de CARRERA_EX
 imprime NO_CUENTA,CVE_CARR
fin hacer
```

1.2.2.3 GENERA RELACIÓN CARRERA POSTER

```
Recibe SOLIC_REL_CARR_POSTER
hacer mientras haya alumno en CARRERA_EX
  imprime REL_CARR_POSTER
fin hacer
```

2.2 REGISTRAR EX_EXTRAOR

```
Si el alumno desea registrar hasta dos exámenes extraordinarios
    obtén CVEASIG
    localiza en el archivo SERIACION las ASIGREQ
    verifica si el alumno tiene aprobadas las ASIGREQ
    si todas las ASIGREQ están aprobadas
       registra ex_extraor
       genera DATOS_EXTRAOR
     si no
        manda mensaje de error "El alumno no ha aprobado
         asignaturas requisito"
      fin si
si no
    obtén CVEASIG
    localiza en el archivo SERIACION las ASIGREQ
    verifica si el alumno tiene aprobadas las ASIGREQ
    si todas las ASIGREQ están aprobadas
       registra el exámen extraordinario
        imprime COMP_TRAMITE_EX_EXTRAOR
        solicita PERMISO_EXTRAOR
        si PERMISO_EXTRAOR_AUTOR = true
          imprime COMP_INSCR_EX_EXTRAOR
        si no
          no registra ex_extraor
        fin si
     fin si
fin si
```

2.3 AUTORIZA EX_EXTRAOR

Despliega datos de PERMISO_EXTRAOR
lee historia académica del archivo ALUMNO
despliega historia académica
obtén autorizacion("S" o "N") y datos de PERMISO_EXTRAOR_AUTOR
guarda PERMISO_EXTRAOR_AUTOR en el archivo GPO_EX

2.4 REGISTRA CALENDARIO

Lee datos CALENDARIO
lee datos del archivo PROFESORES
si existen los datos en el archivo GPO_EX
  actualiza el archivo GPO_EX
sino
  escribe los datos leídos en el archivo GPO_EX
fin si

2.5 EMITE REPORTE DE EX_EXTRAOR MAS_2

Si recibes SOLIC_REP_ALUMNO_AUTO_MAS_2
  lee datos del archivo GPO_EX
  imprime ALUMNO_AUTO_MAS_2
fin si

2.6 EMITE REPORTE DE ALUMNOS INSCRITOS

Si recibes SOLIC_REP_ALUMNO_INSC_ASIG_JURADO
  lee datos del archivo GPO_EX
  imprime ALUMNO_INSC_ASIG_JURADO
fin si

2.7 EMITE REPORTE DE PAGO A SINODALES

Si recibes SOLIC_REP_PAGO_SINOD_EXTRAOR
  lee datos del archivo GPO_EX
  imprime REP_PAGO_SINOD_EXTRAOR
fin si

2.8 EMITE COMP_INSCR EXTRAOR

Recibe DATOS_EXTRAOR
mientras NO_CUENTA = NO_CUENTA en GPO_EX
  lee CVE_ASIG en GPO_EX
  imprime COMP_INSCR_EX_EXTRAOR
fin mientras

3.2 TRAMITA CAMBIO PRIMER INGRESO

Obtén GRALES_ALUMNO
obtén SOLIC_CAMB_PLANTEL
escribe en el archivo CAMBIOS GRALES_ALUMNO y SOLIC_CAMB_PLANTEL
solicita PERMISO_CAMB_PLANTEL

si PERMISO_CAMB_PLANTEL_AUTOR
  tramita cambio plantel primer ingreso
fin si
obtén SOLIC_REL_SOLIC_CAMB_PLANTEL
lee del archivo CAMBIOS
imprime REL_SOLIC_CAMB_PLANTEL

3.3 AUTORIZA CAMBIO PRIMER INGRESO

Lee PERMISO_CAMB_PLANTEL del archivo CAMBIOS
si los datos están completos
  autorización="S"
    escribe PERMISO_CAMB_PLANTEL_AUTOR en archivo CAMBIOS
fin si
obtén SOLIC_REL_SOLIC_CAMB_PLANTEL_AUTOR
lee del archivo CAMBIOS
imprime REL_SOLIC_CAMB_PLANTEL_AUTOR

3.4 TRAMITA CAMBIO GRUPO

Obtén GRALES_ALUMNO
obtén SOLIC_CAMB_GRUPO
escribe en el archivo CAMBIOS GRALES_ALUMNO y SOLIC_CAMB_GRUPO
recibe SOLIC_COMP_TRAMITE_CAMB
imprime COMP_TRAMITE_CAMBIO
Solicita PERMISO_CAMB_GRUPO
si PERMISO_CAMB_GRUPO_AUTOR
  tramita cambio plantel primer ingreso
fin si

3.5 AUTORIZA CAMBIO GRUPO

Lee PERMISO_CAMB_GRUPO del archivo CAMBIOS
si hay cupo en GRUPODESEADO
  autorización="S"
  escribe PERMISO_CAMB_PLANTEL_AUTOR en archivo
  CAMBIOS
  recibe SOLIC_COMP_INSCR
  imprime COMP_INSCR
fin si
obtén SOLIC_REL_SOLIC_CAMB_GRUPO_AUTOR
lee del archivo CAMBIOS
imprime REL_SOLIC_CAMB_GRUPO_AUTOR

3.6 TRAMITE CAMBIO_CICLO

Obtén GRALES_ALUMNO
obtén SOLIC_CAMB_CICLO
escribe SOLIC_CAMB_CICLO en archivo CAMBIO_CICLO
recibe SOLIC_REL_CAMB_CICLO
lee del archivo CAMBIO_CICLO
imprime REL_CAMB_CICLO

3.7 TRAMITA CAMBIO_PLAN_ESTUDIOS

Obtén GRALES ALUMNO
obtén SOLIC_CAMB_PLAN_ESTUD
escribe en el archivo CAMBIOS

3.8 TRAMITA CAMBIO_CARR

```
Obtén GRALES_ALUMNO
obtén SOLIC_CAMBIO_CARR
si los doctos están completos
  doctos="S"
si no
  doctos="N"
fin si
escribe en el archivo CAMBIO_CARR
obtén SOLIC_F_305
imprime F_305
obtén SOLIC_ REL_ALFAB_CARR_ORIGEN
imprime REL_ALFAB_CARR_ORIGEN
obtén SOLIC_REL_ALFAB_GRAL
imprime REL_ALFAB_GRAL
```

DICCIONARIO DE DATOS

"Es un listado organizado de todos los datos pertinentes al sistema, con definiciones precisas y rigurosas para que tanto el usuario como el analista tengan un entendimiento común de todas las entradas, salidas, componentes de datos y cálculos intermedios." [YOU93]

El diccionario de datos define todos los elementos de los datos que se han declarado en los diagramas de flujo de datos a través de:

- La definición del significado de los flujos y archivos

- La especificación de la composición de los flujos y archivos
- La especificación de las unidades y valores de los flujos y archivos

El uso del diccionario de datos reduce ambigüedades y especifica detalles.

El significado debe explicar la relevancia del flujo de datos o del archivo del sistema. Asimismo, debemos evitar cosas redundantes como la descripción de la composición de los datos o derivaciones, la repetición del significado del elemento ya que el elemento tiene su propia descripción, la repetición del nombre del dato.

Los símbolos siguientes se utilizan para definir la composición de un flujo de datos o de un archivo:

=	*está compuesto de*
+	*y*
()	*optativo (puede estar presente)*
{}	*iteración*
[]	*seleccionar una de varias alternativas*
**	*comentario*
¦	*separa opciones alternativas en la construcción*

[YOU93]

Idealmente cada término del diccionario de datos debe referirse a un elemento único en el modelo del sistema. Un alias es un nombre alternativo para algo ya definido en el diccionario.

Diccionario de Datos

acta_complementaria_extraordinario =
* acta complementaria de control de exámenes extraordinarios *

acta_complementaria_ordinario =
* acta complementaria de control de exámenes ordinarios *

acta_extraordinario =
* acta de control de exámenes extraordinarios *

acta_ordinario =
* acta de control de exámenes ordinarios *

alumno_auto_mas_2 =
1{no_cuenta + nomb_alumno+num_exámenes} N

alumno_insc_asig_jurado =
asignatura + cve_jurado + 1{no_cuenta + nomb_alumno + nombre_sinodal}N

alumnos_asig = 1{dígito}3

alumnos_inscr = 1{dígitos}3

año_ing = 2 {dígito} 2

año_ing_ciclo = 4{dígito}4

ape_materno_alumno =
* apellido materno del alumno *

ape_materno_profesor =
* apellido materno del profesor *

ape_paterno_alumno =
* apellido paterno del alumno *

ape_paterno_profesor =
        * apellido paterno del profesor *

aprobado =
        * no acreditar el examen profesional *

aprobados = 1{dígito}6

archivo_carr_poster =
        [archivo_carr_simult | archivo_seg_carr]

archivo_carr_simult =
        1{no_cuenta + nomb_alumno + cve_carr}N

archivo_seg_carr =
        1{no_cuenta + nomb_alumno}N

asig_a_cursar = cve_asig

asig_aprob =
        * Asignaturas aprobadas *

asig_cursada = cve_asig

asig_no_aprob =
        * Asignaturas no aprobadas *

asignatura = nombre_asig

asignatura = 1{caracter}25

aula =
        * espacio físico en que se dividen los edificios *

autor = [si/no]

bajas = 1{dígito}4

bajas_art_23 =1{dígito}4

C.U. = [DGAE | DGIRE | plantel]

calendario =
esc_carr + cve_asig + cve_jurado + 1{rfc_sinodal + nombre_sinodal} 5
+ lugar_examen + fecha

calif =
[MB | B | S | NA | NP | AC | RE | CO | 10 | 9 | 8 | 7 | 6]

calificación =
[aprobado | aprobado con mención honorífica | suspendido]

calle =
nombre_calle + número

camb_gpo =1{dígito}4

cambios =
* comentario sobre la inscripción a alguna asignatura *

carr = nombre_carr

carrera = nombre_carr

carrera_destino = cve_carr

carrera_origen = cve_carr

carta_acepta =
* Documento en el cual se notifica al alumno que ha sido admitido en la
UNAM *

citatorio_alumno =
fecha + hora + lugar + nombre_carr + nomb_alumno + jurado

clinica =
    * nombre de la institución donde el alumno recibirá servicios médicos *

código_postal = 5 {dígito} 5

colegiatura =
    * pago por parte del alumno, por concepto de reinscripción *

colonia =
    * nombre de la colonia donde vive el alumno *

comp_bajas =
    no_cuenta + nomb_alumno + cve_carr + nombre_carr + fecha

comp_ing_años_poster =
    número + nomb_alumno + cve_carr + 1{cve_asig + asignatura + número_grupo}7

comp_inscr =
    fecha_emisión + num_hoja + no_cuenta + cve_carr + ape_paterno_alumno + ape_materno_alumno + nombre_alumno + año_ing + período + página + 1 {cve_plantel + nombre_asig + créditos + semestre + número_grupo + cambios} 7 + firma_alumno

comp_inscr_ex_extraor =
    [comp_inscr_ext_normal | comp_inscr_ext_2]
comp_inscr_ex_obj =
    folio + no_cuenta + nomb_alumno + cve_carr + fecha + hora + aula

comp_inscr_ex_prof =
    [comp_inscr_ex_obj | citatorio_alumno]

comp_inscr_ext_2 =
    comp_inscr_ext_normal

comp_inscr_ext_normal =
    1{no_cuenta + nomb_alumno + número_grupo + cve_asig + asignatura + jurado}N

comp_prov = comp_insc + leyenda

comp_seguro =
no_cuenta + nomb_alumno + fecha_inicio + no_afiliación + nombre_escuela + clinica

comp_solicitud =
[comp_tramite_reg | comp_inscr_ex_extraor | comp_inscr | comp_seguro | comp_inscr_ex_prof | comp_tramite_cambio | comp_tramite_ex_extraor]

comp_tramite_cambio =
fecha + no_cuenta + nomb_alumno + cve_plante + cve_carr + año_ing + periodo + pagina + grupo_actual + grupo_deseado

comp_trámite_ex_extraord =
1{no_cuenta + nomb_alumno + cve_asig + asignatura + número_grupo + leyenda}N

comp_tramite_reg =
[comp_inscr | comp_provisional_ing_años_poster]

const_est_cred_prom =
1{nomb_alumno + no_cuenta + nombre_carr + semestre + prom + creditos_cubiertos + horario + periodo + fecha}N

const_est_inscr =
1{nomb_alumno + no_cuenta + nombre_carr + semestre + horario + periodo + fecha}N

cred_oblig = * Créditos obligatorios *

cred_optat = * Créditos optativos *

credenciales_mexico =
* entidad que expide la credencial del alumno *

créditos = 2 {dígito} 2

créditos_cubiertos =
* suma de los créditos de cada asignatura aprobada *

cuadro_comp_primer_ing =
1{año_ing_ciclo + nombre_carr + cupos + alumnos_asig + alumnos_inscr}N

cupos = 1{dígitos}3

cve_asig = 4 {dígito} 4

cve_carr = 7 {dígito} 7

cve_jurado = 4{dígito}4

cve_plantel = 3 {dígito} 3

datos_baja =
grales_alumno + fecha

datos_extraor =
grales_alumno + fecha

datos_inscr =
grales_alumno + fecha

datos_pers =
ape_paterno_alumno + ape_materno_alumno + nomb_alumno + nacionalidad + fecha_nac + edad + sexo + edo_civil_alumno + tel_alumno + medio_transp + tiempo_transp + escuela_proced + tipo_escuela_proced + promedio_escuela_proced + calle + colonia + delegación + entidad_fed_alumno + código_postal + empresa_trabaja + padre_alumno + seguro_fac + tel_empresa_trabaja + entidad_fed_empresa

datos_seguro =
clinica + no_afiliación + lugar + vigencia + fecha_admvo + fecha_movto

delegación = 1{caracter}30

dependencias_iztacala =
[biblioteca | depto_titulación | jefatura_sección | jefatura_carrera | dirección]

DGAE =
* Dirección General de Administración Escolar *
DGIRE =
* Dirección General de Incorporación y Revalidación de Estudios *

diario_mexico =
* entidad que publica las estadísticas de alumnos sobresalientes *

doctos_emite =
[etiquetas_ind | etiquetas_grales | const_est_inscr | const_est_cred_prom | hist_acad | comp_inscr]

domicilio =
calle + número + colonia + delegación + entidad_fed_alumno + código_postal

edad = 1{dígito}2

edo_civil =
[soltero | casado | viudo | divorciado | unión_libre]

edo_civil_alumno = edo_civil

egreso =1{dígito}3

empresa_trabaja =
* en caso de que trabaje el alumno, nombre de la empresa *

entidad_fed_alumno =
* Nombre del Estado donde vive el alumno *

entidad_fed_empresa =
* en caso de que el alumno trabaje, Estado en el que se localiza la empresa *

entregado =1{dígito}4

esc_carr = cve_carr + nombre_carr

escuela_proced =
*institución educativa,donde el alumno realizó sus estudios anteriores *

est_actas =
[est_actas_grales | est_actas_rectif]

est_actas_grales =
1{periodo + nombre_carr + tipo_acta + total}N

est_actas_rectif =
1{periodo + nombre_carr + no_actas + total}N

est_alumnos_diploma =
1{periodo + nombre_carr + generación + no_lugar + total}N

est_alumnos_medalla =
1{periodo + nombre_carr + generación + num_alumnos + promedio + total}N

est_alumnos_sobresalientes =
[est_alumnos_diploma | est_alumnos_medalla]

est_asig_primer_ing =
período + nombre_carr + {número_grupo + asignatura + nombre_asig + total}

```
est_bajas =
        1{periodo + nombre_carr + mes_tot + total}

est_camb_carr =
        1{periodo + nombre_carr + egreso + ingreso + total}N
est_camb_ciclo =
        1{año + nombre_carr + tram_conc + tram_no_conc + total}N

est_camb_gpo_bajas_art_23 =
        1{nombre_carr + bajas_art_23 + camb_gpo + total}N

est_camb_plantel_ing =
        1{periodo + nombre_carr + plantel_solic + total}N

est_camb_plantel_reing =
        est_camb_plantel_ing

est_carr_sim =
        1{periodo + nombre_carr + num_alumnos + total}N

est_certificados =
        1{periodo + nombre_carr + mes_tot + total}N

est_constancia_100% =
        1{periodo + nombre_carr + mes_tot + total}N

est_constancia_sit_esc =
        est_constancia_100%

est_constancias =
        [est_constancia_100% | est_constancia_sit_esc]

est_constancias_tit =
        1{periodo + nombre_carr + mes_tot + total}N

est_credencial =
        [est_tarjetón_esp | est_credenciales_méxico]
```

est_credenciales_méxico =
    est_tarjetón_esp

est_cuotas_volunt =
    1{semestre + nombre_carr + periodo + total}N

est_cupos_gpos =
    1{periodo + nombre_carr + semestre + número_grupo + total

est_dictámenes =
    1{periodo + nombre_carr + mes_tot + total}N

est_egresados =
    1{periodo + nombre_carr + total}N

st_ex_extraor =
    [est_ex_extraor_inscr_asig | est_ex_extraor_EB | est_ex_extraor_inscr |
    est_ex_extraor_EA | est_ex_extraor_mas_dos]

est_ex_extraor_EA =
    est_ex_extraor_EB

est_ex_extraor_EB =
    1{periodo + nombre_carr + inscritos + aprobados + no_aprobados +
    total}N

est_ex_extraor_inscr =
    1{periodo + nombre_carr + total}N

est_ex_extraor_inscr_asig =
    1{periodo + nombre_carr + cve_asig + asignatura + num_jurado +
    total}N

est_ex_extraor_mas_dos =
    1{periodo + nombre_carr + alumnos_inscr + exam_auto + total}N

est_ex_prof_aprob_rep =
    1{periodo + año + nombre_carr + total}N

est_ex_prof_inscr =
periodo + {nombre_carr + num_ext + total}

est_gpos_asig =
1{periodo + carr + número_grupo + asignatura + total}N

est_ing_años_post =
1{periodo + nombre_carr + num_alumnos}N

est_inscr_reinscr =
[est_asig_primer_ing | est_camb_gpo_bajas_art_23 |
cuadro_comp_primer_ing | est_cuotas_volunt | est_cupos_gpos |
est_gpos_asig | est_pago_lab | resumen_inscr_anual]

est_otros_ing =
[est_camb_ciclo | est_camb_carr + est_carr_sim | est_camb_plantel_ing
| est_seg_carr | est_ing_años_post | est_camb_plantel_reing]

est_pago_lab =
1{nombre_carr + periodo + semestre + num_alumnos + importe +
total}N

est_rev_estud =
1{periodo + nombre_carr + tot_tram_unic + tot_tram_simul + tot_certif
+ total}N

est_seg_carr =
1{periodo + nombre_carr + num_alumnos + total}N

est_solic_seg_fac =
1{periodo + nombre_carr + mes_tot + bajas + entregado +
no_entregado + total}N

est_tarjetón_esp =
1{periodo + nombre_carr + mes_tot + total}N

est_titulación =
[est_constancias | est_egresados | est_ex_prof_aprob_rep | est_ex_prof_inscr | est_rev_estud]

est_varias =
[est_actas | est_constancias | est_bajas | est_certificados | est_credencial | est_dictámenes | est_alumnos_sobresalientes | est_solic_seg_fac]

estadisticas =
[est_insc_reinsc | est_ex_extraor | est_otros_ing | est_titulación | est_varias]

etiqueta_ind =
nombre_carr + plan + num_comprobante + no_cuenta + nomb_alumno + tel + domicilio + generación + colonia + código_postal

etiquetas_grales =
1{nombre_carr + plan + num_comprobante + no_cuenta + nomb_alumno + tel_alumno + domicilio + generación + colonia + código postal}N

exam_auto = 1{dígito}5

extraor_autor =
nombre_autor + autor + fecha_movimiento

extraor_presenta =
1 {cve_asig + cve_jurado} 6 + (solic_mas_2)

extraordinario =
* examen que se presenta para acreditar una asignatura fuera del período ordinario *

f = femenino

F-305 =
* Forma preimpresa para trámite de cambios de carrera *

fecha =
día + mes + año

fecha_admvo =
fecha * día en el que el IMSS tramita el seguro facultativo *

fecha_emisión =
fecha * día en que se emitió el documento *

fecha_ex = fecha

fecha_fin_sanción = fecha

fecha_inici = fecha

fecha_movimiento =
fecha * día en que se autoriza la presentación del exámen extraordinario, por parte del funcionario *

fecha_movto =
fecha * día en el que la UNAM inicia trámite del seguro facultativo *

fecha_nac = día + mes + año

fecha_sol =
fecha * día en el que el alumno solicita el seguro facultativo *

firma_alumno = * firma del alumno *

firma_plantel_destino =
* comprobante de acptación del plantel destino *

firma_plantel_origen =
* comprobante de aceptación del plantel origen *

folio = 1{dígito}5

folio_acta = 7{dígito}7

folio_rect = 7{dígito}7

folio_titula = 7{dígito}7

generación = año_ing

gpo_autor = * Grupo autorizado *

grado =
[licenciado | maestro | doctor | externo]

grales_alumno =
nomb_alumno | num_comprobante + año_ing + plan + cve_carr

grupo_actual = número_grupo

grupo_asig = cve_asig

grupo_autor = gpo_autor

grupo_deseado = número_grupo

grupos_reinscr =
pago_lab + colegiatura + 1 {número_grupo + cve_asig} 7

hist_acad =
fecha + nomb_alumno + no_cuenta + pagina + hoja + plantel + cve_plantel + año_ing + nombre_carr + cve_carr + periodo + cred_oblig + cred_optat + asig_aprob + asig_no_aprob + prom + 1 {cve_asig + creditos + nombre_asig + calif + tipo_exam + folio_acta + número_grupo + num_ord + num_ext} N

hoja =
* número consecutivo asignado automáticamente *

hora =
* momento del día en que se realiza un evento determinado*

horario =
* período en el que se realiza una determinada actividad *

identifica_alumno =
no_cuenta + cve_carr

identifica_grupo =
cve_carr + número_grupo + cve_asig

importe =
1{dígito}6 * suma de las colegiaturas de todos los alumnos de una carrera, un semestre y un período determinado *

IMSS =
* Instituto Mexicano del Seguro Social *

ingreso = 1{dígito}3

inscritos = 1{dígito}6

jurado =
presidente + vocal + secretario + suplente + suplente + grado

leyenda =
* Comprobante provisional de trámite *

listado_bajas =
1 {no_cuenta + nomb_alumno + nombre_carr} N

lugar =
* espacio físico en donde se realiza una determinada actividad *

lugar_exámen =
fecha_ex_extraor + hora_ex_extraor + aula_ex_extraor

m = masculino

medio_transp =
[metro | camión | colectivo | particular | otro]

mes_tot = 1{dígito}4

nacionalidad = [mex | ext]

no_actas = 1{dígito}3

no_actas_rectifica = no_actas

no_afiliación = 1{dígito}17

no_aprobado = * dar por bueno el examen profesional *

no_aprobados = 1{dígito}6

no_cuenta = 8 {dígito} 8

no_entregado =1{dígito}4

no_lugar =
[1o_lugar | 2o_lugar | 3er_lugar]

no_rep =
* Número de reposición de credencial *

nom_trab_titulación =
* Nombre del trabajo de titulación *

nomb_alumno =
ape_paterno_alumno + ape_materno_alumno + nombre_alumno

nombre_alumno =
* nombre de pila del alumno *

nombre_asig = 1 {caracter} 28

nombre_autor =
* nombre del funcionario que autorizo que el alumno puede presentar el exámen extraordniario *

nombre_calle = 1{caracter}35

nombre_carr =
[biólogo | enfermería_nivel_técnico | médico_cirujano | cirujano_dentista | lic_psicología | lic_optometría]

nombre_escuela = 1{caracter}30

nombre_profesor = * nombre de pila del profesor *

nombre_sinodal =
1{caracter}30 *nombre del profesor sinodal*

num_alumnos = 1{dígito}4

num_comprobante = 1{dígito}5

num_consecutivo = 1{dígito}3

num_ex_1 =
1{dígito}3 * número de exámenes con menos de 3 sinodales *

num_ex_2 =
1{dígito}3 * número de exámenes con más de 2 sinodales *

num_exámenes = num_ext

num_ext = 1{dígito}2 * número de extraordinarios *

num_hoja = 1{dígito}3

num_jurado = 1{dígito}2

num_ord =
1{dígito}1 * número de ordinarios *

número = 1{dígito}4

número_grupo = 4 {dígito} 4

opción_por_acreditación =
* Selecciona la opción de ingreso años posteriores por acreditación*

opción_por_revalidación =
* Selecciona la opción de ingreso años posteriores por revalidación *

ordinario =
* examen que el alumno presenta para acreditar una asignatura dentro del periodo establecido *

padre_alumno =
* nombre del padre del alumno *

página =
* en caso de más de una página, número de página *

pago_lab = [si | no]

pago_sinod_extraor =
periodo + 1{rfc_sinodal + nombre_sinodal + num_exámen1 + num_exam2}N

período = 1{dígito}3

permiso_camb_grupo =
1 {no_cuenta + nomb_alumno} n

permiso_camb_grupo_autor =
1 {no_cuenta + nomb_alumno + autor} n

permiso_camb_plantel =
[permiso_cambio_pi | permiso_cambio_reins]

permiso_camb_plantel_autor =
[permiso_cambio_pi_autor]

permiso_cambio_pi =
1 {no_cuenta + num_consecutivo} N

permiso_cambio_pi_autor =
1 {no_cuenta + num_consecutivo + autor} N

permiso_cambio_reins =
permiso_cambio_pi

permiso_cambio_reins_autor =
permiso_cambio_pi_autor

permiso_extraor =
1 {no_cuenta + nomb_alumno + cve_asig + cve_jurado} 4

permiso_extraor_autor =
permiso_extraor + 1 {extraor_autor} 4

petición =
* orden del operador de emitir un cierto reporte, comprobante, etc. *

plan = 1{dígito}2

plantel = 1{dígito}3

plantel_destino = 1{dígito}3

plantel_origen = 1{dígito}3

plantel_solic = 1{dígito}3

presidente =
profesor * es la máxima jerarquía del jurado *

profesor =
ape_paterno_profesor + ape_materno_profesor + nombre_profesor

prom =
* Promedio de las calificaciones *

promedio = prom

promedio_escuela_proced =
* representa el promedio del alumno, obtenido en la escuela de procedencia *

rel_actas =
1{nomb_alumno + nombre_carr}N

rel_actas_faltantes =
cve_carr + periodo + fecha + 1{asignatura + profesor + folio_acta + número_grupo + tipo_examen}N

rel_actas_para_DGAE =
nombre_carr + periodo + fecha + 1{tipo_acta + folio_acta + asignatura + profesor + número_grupo + tipo_examen}N

rel_actas_rectifica_para_cu =
nombre_carr + período + [folio_acta + asignatura + profesor + número_grupo + tipo_examen]

rel_alfab_carr_origen =
nombre_carr + periodo + 1 {no_cuenta + nomb_alumno + año_ing + carrera_origen + carrera_destino} N * incluye alumnos de una carrera en específico *

rel_alfab_gral =
nombre_carr + periodo + 1 {no_cuenta + nomb_alumno + año_ing + carrera_origen + carrera_destino} N * son todos los alumnos, sin importar la carrera a la que pertenecen *

rel_alumno_baja_seguro =
periodo + fecha + 1{no_cuenta + ape_paterno_alumno + ape_materno_alumno + nombre_alumno

rel_alumno_inscr =
1{folio + no_cuenta + nomb_alumno + calificación}N

rel_alumno_reinscr =
1{nombre_carr + no_cuenta + nomb_alumno}N

rel_alumno_seguro =
periodo + fecha + 1{no_cuenta + ape_paterno_alumno + ape_materno_alumno + nombre_alumno

rel_autor_camb_grupo =
1{fecha + no_cuenta + nomb_alumno + gpo_autor}N

rel_bajas =
período + cve_carr + fecha + 1{no_cuenta + nombre_alumno}

rel_camb_ciclo =
periodo + nombre_carr + 1{ no_cuenta + nomb_alumno}N

rel_camb_plantel_autor_rei =
1{fecha + no_cuenta + nomb_alumno + plantel_destino}N

rel_camb_plantel_autor_p_i =
1{fecha + no_cuenta + nomb_alumno + plantel_destino}N

rel_camb_plantel_p_i =
1{fecha + no_cuenta + nomb_alumno + F-305 + comp_inscr + tarjetón_credencial + carta_aceptación + sistema}N

rel_camb_plantel_rei =
1{no_cuenta + nomb_alumno + F-305 + hist_acad + comp_inscr + tarjetón_credencial + carta_aceptación + sistema}N

rel_cont_alumno =
[rel_alumno_reinscr | comp_bajas | rel_bajas]

rel_folio_asig =
cve_carr + periodo + 1{folio_acta + asignatura + profesor + número_grupo + tipo_examen}N

rel_profesor_asig =
cve_carr + periodo + 1{profesor + asignatura + no_actas_rectifica}N

rel_solic_camb =
[rel_solic_camb_plantel | rel_camb_ciclo | rel_alfab_carr_origen | rel_alfab_gral]

rel_solic_camb_autor =
[rel_solic_camb_plantel_autor | rel_solic_camb_grupo_autor]

rel_solic_camb_grupo_autor =
nombre_carr + 1 {no_cuenta + nomb_alumno + grupo_autor} N

rel_solic_camb_pi =
nombre_carr + 1 {num_consecutivo + no_cuenta + nomb_alumno + F-305 + comp_inscr + tarjetón_cred + carta_acepta + sistema } N + firma_plantel_origen + firma_plantel_destino

rel_solic_camb_plantel =
[rel_solic_camb_pi | rel_solic_camb_reins]

rel_solic_camb_plantel_autor =
[rel_solic_camb_ri_autor | rel_solic_camb_reins_autor]

rel_solic_camb_reins =
nombre_carr + 1 {num_consecutivo + no_cuenta + nomb_alumno + F-305 + hist_acad + comp_inscr + tarjetón_cred + carta_acepta + sistema } N + firma_plantel_origen + firma_plantel_destino

rel_solic_camb_reins_autor =
1 {nombre_carr + no_cuenta + nomb_alumno + plantel_destino} N

rel_solic_camb_ri_autor =
nombre_carr + 1 {num_consecutivo + no_cuenta + nomb_alumno + plantel_destino} N

rel_solic_carr_poster =
[rel_solic_carr_simult | rel_solic_seg_carr]

rel_solic_carr_simult =
periodo + 1 {no_cuenta + nomb_alumno + carrera_origen + carrera_destino} N

rel_solic_no_tramitadas_repos_creden =
período + nombre_carr + 1{no_cuenta + nomb_alumno}
rel_solic_repos_creden =
1{no_cuenta + nomb_alumno + nombre_carr + no_rep}N

rel_solic_rev_estud =
rel_solic_tipo_rev

rel_solic_seg_carr =
rel_solic_carr_simult

rel_solic_tipo_rev =
periodo + cve_carr + fecha + 1{no_cuenta + nomb_alumno + tipo_rev}N

rel_solicitudes =
no_cuenta + nombre + nombre_carr

rel_titula =
[rel_alumno_inscr | rel_actas | rel_solic_rev_estud]

rel_tramite_reg =
[rel_cont_alumno | rel_solic_carr_poster]

relación =
[rel_tramite_reg | rep_ex_extraor | rel_solic_camb |
rel_solic_camb_auto | rep_seguro | rep_control_actas |
rep_control_actas_rectifica | rel_actas | rel_titula | rep_emisión_doctos]

rep_control_actas =
[rel_actas_para_DGAE + rel_actas_faltantes]

rep_control_actas_rectifica =
[rel_folio_asig | rel_actas_rectifica_para_DGAE | rel_profesor_asig]

rep_emisión_doctos =
[rel_solic_repos_creden | rel_solic_no_tramitadas_repos_creden]

rep_ex_extraor =
[alumno_auto_mas_2 | alumno_insc_asig_jurado |
rep_pago_sinod_extraor]

rep_pago_sinod_extraor =
período + nombre_carr + {rfc_sinodal + profesor + num_ex_1 +
num_ex_2 + total}

rep_seguro =
[rel_alumno_seguro | rel_alumno_baja_seguro]

resultado =
[aprobado | no_aprobado]

resumen_inscr_anual =
1{periodo + nombre_carr + total}N

rfc_sinodal =
    * registro federal de contribuyentes del profesor sinodal *

secretario =
    profesor * es la tercera jerarquía del jurado*

seguro_fac =
    * en caso de que el alumno tenga seguro facultativo, número de afiliación *

semestre =
    [01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10]

sexo = [m | f]

sistema = [escolarizado | SUA]

solic_acredita =
    numero + nomb_alumno + cve_carr+ cve_asig

solic_act_calif =
    identifica_grupo + cve_asig + folio_acta + calif

solic_acta_equival =
    numero + no_cuenta + calif + creditos

solic_actas =
    [solic_registro_actas | solic_control_actas | solic_control_actas_rectifica | solic_registro_actas_rectifica]

solic_alta_seguro =
    identifica_alumno +[fecha | repos_comp_seguro]

solic_amonesta = petición

solic_auto_mas_2 = petición

solic_baja_def_vol = petición

solic_baja_seguro =
identifica_alumno + fecha_sol

solic_camb =
identifica_alumno + [solic_camb_plantel | solic_camb_grupo | solic_camb_ciclo | solic_camb_plan_estud | solic_cambio_carr]

solic_camb_ciclo =
no_cuenta + nomb_alumno

solic_camb_grupo =
grupo_deseado

solic_camb_pi =
f-305 + comp_inscr + tarjeton_cred + carta_acepta + sistema

solic_camb_plan_estud =
no_cuenta + nomb_alumno + plan_destino

solic_camb_plantel =
[solic_camb_pi | solic_camb_reins]

solic_camb_reins =
f-305 + hist_acad + comp_inscr + tarjeton_cred + carta_acepta + sistema

solic_cambio =
identifica_alumno + [solic_camb_plantel | solic_camb_grupo | solic_camb_ciclo | solic_camb_plan_estud | solic_camb_carr]

solic_cambio_carr =
no_cuenta + nomb_alumno + carrera_destino

solic_carr_poster =
[solic_carr_simult | solic_seg_carr]

solic_carr_simult =
    plantel_origen + carrera_origen + plantel_destino + carrera_destino

solic_citatorio_alumno = petición

solic_comp_baja = petición

solic_comp_inscr = petición

solic_comp_inscr_ex_extraor = petición

solic_comp_tramite_camb = petición

solic_const_est = petición

solic_control_actas =
    tipo_acta + folio_acta

solic_control_actas_rectifica =
    folio_acta + no_cuenta

solic_cuadro_comp_primer_ing =
    petición

solic_def_vol =
    no_cuenta + nomb_alumno

solic_emisión_doctos =
    identifica_alumno + solic_emisión_doctos

solic_emisión_doctos =
    [solic_etiquetas | solic_etiquetas_ind | solic_const_est |
    solic_repos_comp_inscr | solic_respos_creden | solic_respos_hist_acad]

solic_est_actas = petición

solic_est_alumnos_sobresalientes = petición

solic_est_asig_primer_ing = petición

solic_est_bajas = petición

solic_est_camb_carr = petición

solic_est_camb_ciclo = petición

solic_est_camb_gpo_bajas_art_23 = petición

solic_est_camb_plantel_ing = petición

solic_est_camb_plantel_reing = petición

solic_est_carr_sim = petición

solic_est_certificados = petición

solic_est_constancias = petición

solic_est_constancias_tit = petición

solic_est_credencial = petición

solic_est_cuotas_volunt = petición

solic_est_cupos_gpos = etición

solic_est_dictámenes = petición

solic_est_egresados = petición

solic_est_ex_extraor =
[solic_est_ex_extraor_inscr_asig | solic_est_ex_extraor_EB |
solic_est_ex_extraor_inscr | solic_est_ex_extraor_EA |
solic_est_ex_extraor_mas_dos]

solic_est_ex_extraor_inscr = petición

solic_est_ex_extraor_mas_dos = petición

solic_est_ex_extraor_EA = petición

solic_est_ex_extraor_EB = petición

solic_est_ex_extraor_inscr_asig = petición

solic_est_ex_prof_aprob_rep = petición

solic_est_ex_prof_inscr = petición

solic_est_gpos_asig =
petición

solic_est_ing_años_post =
petición

solic_est_insc_reinsc =
[solic_est_asig_primer_ing | solic_est_camb_gpo_bajas_art_23 |
solic_cuadro_comp_primer_ing | solic_est_cuotas_volunt |
solic_est_cupos_gpos | solic_est_gpos_asig | solic_est_pago_lab |
solic_resumen_inscr_anual]

solic_est_otros_ing =
[solic_est_camb_ciclo | solic_est_camb_carr | solic_est_carr_sim |
solic_est_camb_plantel_ing | solic_est_seg_carr |
solic_est_ing_años_post | solic_est_camb_plantel_reing]

solic_est_pago_lab =
petición

solic_est_rev_estud =
petición

solic_est_seg_carr =
petición

```
solic_est_solic_seg_fac =
        petición

solic_est_titulación =
        [solic_est_constancias          |       solic_est_egresados          |
        solic_est_ex_prof_aprob_rep     |       solic_est_ex_prof_inscr      |
        solic_est_rev_estud]

solic_est_varias =
        [solic_est_actas    |  solic_est_bajas    |  solic_est_constancias    |
        solic_est_certificados | solic_est_credencial | solic_est_dictámenes |
        solic_est_alumnos_sobresalientes | solic_est_solic_seg_fac]

solic_estadisticas =
        [solic_est_insc_reinsc  |  solic_est_ex_extraor  |solic_est_otros_ing  |
        solic_est_titulación |solic_est_varias]
solic_etiqueta_ind =
        petición

solic_etiquetas =petición

solic_ex_EA =
        identifica_alumno + 1 {extraor_presenta} 6

solic_ex_extraor =
        identifica_alumno + extraor_presenta

solic_ex_prof =
        [solic_ex_prof_oral | solic_ex_prof_obj]

solic_ex_prof_obj =
        identifica_alumno + folio_titula

solic_ex_prof_oral =
        identifica_alumno + folio_titula + tesis + jurado +

solic_expulsión = petición
```

solic_f-305 = años_posteriores = petición

solic_ing_años_poster =
no_cuenta + nomb_alumno + cve_carr + plan + semestre + 1{cve_asig + calif}N

solic_inscr =
identifica_alumno + datos_pers

solic_listado_bajas = petición

solic_mas_2 =
* solicitud para presentar más de dos exámenes extraordinarios *

solic_no_tramitadas_repos_creden =
no_cuenta + nomb_alumno + nombre_carr + no_rep

solic_rect_calif =
identifica_alumno + folio_acta + folio_rect + calif

solic_reg =
[solic_reinscr | solic_inscr | solic_baja_def_vol | solic_sig_carr | solic_ing_años_poster]

solic_registro_acta =
tipo_acta + número_grupo + cve_asig + profesor + folio_acta

solic_registro_actas_rectifica =
profesor + folio_acta + cve_asig + tipo_examen + semestre

solic_reinscr =
identifica_alumno + grupos_reinscr + datos_pers

solic_rel_actas = petición

solic_rel_actas_faltantes = petición

solic_rel_actas_para_DGAE = petición

solic_rel_actas_rectifica_para_cu = petición

solic_rel_alfab_carr_origen = petición

solic_rel_alfab_gral = petición

solic_rel_alumno_baja_seguro =petición

solic_rel_alumno_inscr = petición

solic_rel_alumno_reinscr = petición

solic_rel_alumno_seguro = petición

solic_rel_autor_camb_grupo = petición

solic_rel_bajas = petición

solic_rel_camb_ciclo = petición

solic_rel_camb_plantel_autor_p_i = petición

solic_rel_camb_plantel_autor_rei = petición

solic_rel_camb_plantel_p_i = petición

solic_rel_camb_plantel_rei = petición

solic_rel_carr_poster =
    [solic_rel_carr_simult | solic_rel_seg_carr]

solic_rel_carr_simult = petición

solic_rel_cont_alumno =
    [solic_rel_alumno_reinscr | solic_rel_bajas | solic_comp_baja]

solic_rel_folio_asig = petición

solic_rel_profesor_asig = petición

solic_rel_seg_carr = petición

solic_rel_solic_camb =
[solic_rel_solic_camb_plantel | solic_comp_tramite_camb |
solic_comp_inscr | solic_rel_camb_ciclo | solic_rel_alfab_carr_origen |
solic_rel_alfab_gral | solic_f-305]

solic_rel_solic_camb_autor =
[solic_rel_solic_camb_plantel_autor|solic_rel_solic_camb_grupo_autor]

solic_rel_solic_camb_grupo_autor = petición

solic_rel_solic_camb_pi = petición

solic_rel_solic_camb_plantel =
[solic_rel_solic_camb_pi | solic_rel_solic_camb_reins]
solic_rel_solic_camb_plantel_autor =
[solic_rel_solic_camb_ri_autor | solic_rel_solic_camb_reins_autor]

solic_rel_solic_camb_reins = petición

solic_rel_solic_camb_reins_autor = petición

solic_rel_solic_camb_ri_autor = petición

solic_rel_solic_carr_poster =
[solic_rel_solic_carr_simult | solic_rel_solic_seg_carr]

solic_rel_solic_carr_simult = petición

solic_rel_solic_no_tramitadas_repos_creden = petición

solic_rel_solic_repos_creden = petición

solic_rel_solic_rev_estud =
    solic_rel_solic_tipo_rev

solic_rel_solic_tipo_rev = tipo_rev

solic_rel_solic_seg_carr = petición

solic_rel_solicitudes = petición

solic_rel_titula =
    [solic_rel_alumno_inscr | solic_rel_actas | solic_rel_solic_rev_estud |
    solic_citatorio_alumno]

solic_rel_tramite_reg =
    [solic_rel_cont_alumno | solic_rel_solic_carr_poster]

solic_relación =
    [solic_rel_tramite_reg | solic_rep_ex_extraor | solic_rel_solic_camb |
    solic_rep_seguro | solic_rep_control_actas |
    solic_rep_conrtrol_actas_rectifica | solic_rel_titula |
    solic_rep_emisión_doctos | solic_rel_solic_camb_autor]

solic_rep_alumno_auto_mas_2 = petición

solic_rep_alumno_insc_asig_jurado = petición

solic_rep_cambios =
    [solic_rel_camb_plantel_p_i | solic_rel_camb_plantel_rei |
    solic_rel_camb_plantel_autor_p_i | solic_rel_camb_plantel_autor_rei |
    solic_rel_autor_camb_grupo | solic_rel_camb_ciclo |
    solic_rel_alfab_carr_origen | solic_rel_alfab_gral]

solic_rep_control_actas =
    [solic_rel_actas_para_DGAE | solic_rel_actas_faltantes]

solic_rep_control_actas_rectifica =
    [solic_rel_folio_asig | solic_rel_actas_rectifica_para_DGAE |

solic_rep_emisión_doctos =
[solic_rel_solic_repos_creden | solic_rel_solic_no_tramitadas_ repos_ creden]

solic_rep_ex_extraor =
[solic_rep_alumno_auto_mas_2 | solic_rep_alumno_insc_asig_jurado | solic_rep_pago_sinod_extraor]

solic_rep_pago_sinod_extraor = petición

solic_rep_reinscr = petición

solic_rep_seguro =
[solic_rel_alumno_seguro | solic_rel_alumno_baja_seguro]

solic_repos_comp_inscr = petición

solic_repos_comp_seguro = petición

solic_repos_creden = petición

solic_repos_hist_acad = petición

solic_resumen_inscr_anual = petición

solic_rev_estud =
identifica_alumno + tipo_rev

solic_revalida =
nomb_alumno + cve_carr + cve_asig

solic_sancion =
identifica_alumno + [solic_amonesta | solic_suspensión | solic_expulsión]

solic_seg_carr =
plantel_origen + carrera_origen + plantel_destino + carrera_destino

solic_sig_carr = identifica_alumno + solic_carr_poster

solic_suspensión = fecha_fin_sanción + petición

solicitud =
[solic_reg | solic_ex_extraor | solic_cambios | solic_baja_seguro | solic_alta_seguro | solic_rect_calif | solic_inscr_ex_prof | solic_rev_estud | solic_emisión_doctos]

suplente =
profesor * en caso de que al examen profesional no asista alguno de los otros integrantes del jurado, éste tomará su lugar *

tarjetón_cred =
* Documento que se utiliza para llevar un control de las reposiciones de la credencial del alumno *

tel = teléfono

tel_alumno = teléfono

tel_empresa_trabaja = teléfono

teléfono = 7 {dígito} 12

tesis =
* trabajo de indagación que realiza el alumno para presentar su examen profesional *

tiempo_transp =
* duración en horas en que el alumno tarda en trasladarse de su domicilio a la escuela *

tipo_acta =
[acta_ordinario | acta_extraordinario | acta_complementaria_ordinario | acta_complementaria_extraordinario]

tipo_escuela_proced =
    [oficial | particular | extranjera]

tipo_exam =
    [ordinario | extraordinario]

tipo_examen =
    [ordinario|extraordinario]

tipo_rev =
    [trámite_único | trámite_certificado | trámite_simultáneo]

tot_certif = 1{dígito}3

tot_tram_simul = 1{dígito}3

tot_tram_unic = 1{dígito}3

total = 1{dígito}6

tram_conc = 1{dígito}2

tram_no_conc = 1{dígito}3

trámite_certificado = no_cuenta + fecha_sol

trámite_simultáneo =
    no_cuenta + fecha_sol

trámite_único =
    no_cuenta + fecha_sol

vigencia =
    * tiempo de validez del seguro facultativo *

vocal =
    profesor * es la segunda jerarquí del jurado *

MODELO DE INFORMACIÓN

Una vez realizado el análisis de la información recopilada y de los requerimientos del usuario, el siguiente paso es identificar las entidades, así como las relaciones existentes entre éstas, y estructurarlas de tal forma que se obtenga un diagrama que representa el diseño lógico de la base de datos. Este diagrama es llamado Diagrama de Entidad - Relación (DER).

El principal propósito del DER es representar los objetos de datos y sus relaciones. Este diagrama consta de dos componentes principales:

1. Tipos de objetos. Se representan por medio de un rectángulo en el diagrama. Son una colección o conjunto de objetos (cosas) del mundo real cuyos miembros juegan algún papel en el desarrollo del sistema; pueden además ser identificados de manera única y ser descritos por uno o más atributos.

2. Relaciones. Son la serie de conexiones o asociaciones entre los tipos de objetos que están conectados con la relación por medio de líneas.

Con la finalidad de facilitar la interpretación del Diagrama Entidad-Relación incluimos la explicación de la notación utilizada.

Las ***entidades*** se representan mediante un rectángulo dentro del cual se incluye el nombre de dicha entidad.

Las ***relaciones*** se simbolizan con una línea que une a las entidades involucradas y se indica el nombre de la misma.

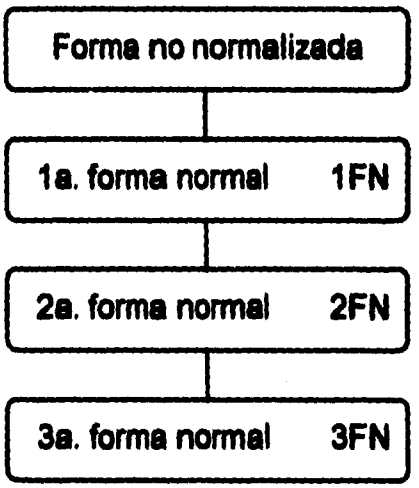
Las relaciones pueden ser entre dos ó más entidades, en el caso de sólo dos entidades la ***cardinalidad*** máxima se representa de la siguiente manera:

muchos a muchos	M N
muchos a uno	N 1
uno a uno	1 1

Asimismo, realizamos una descripción de las entidades, relaciones y los atributos de ambas.

Un segundo paso es aplicar la técnica de normalización, la cual, es una técnica desarrollada para asegurar que las estructuras de datos sean eficientes.

Los pasos para llevar a cabo la normalización son los siguientes:



Una relación está en la 1FN si todos los campos en cada registro contienen un solo valor tomado de sus dominios respectivos. En este paso se convierte la estructura de datos a la forma de tablas bidimensionales.

Una relación está en la 2FN si es 1FN y cada atributo no llave de la relación es total y funcionalmente dependiente de su llave principal. En este paso cada atributo depende totalmente de la llave principal.

Una relación está en 3FN si es 2FN y ningún atributo no llave de la relación es funcionalmente dependiente de algún otro atributo no llave. En este paso se elimina cualquier dependencia transitiva de atributos no primos.

El atributo b de una relación r es ***funcionalmente dependiente*** del atributo a de la relación r si, en cada instante, cada valor de a está asociado con no más de un valor de b dentro de la relación r.

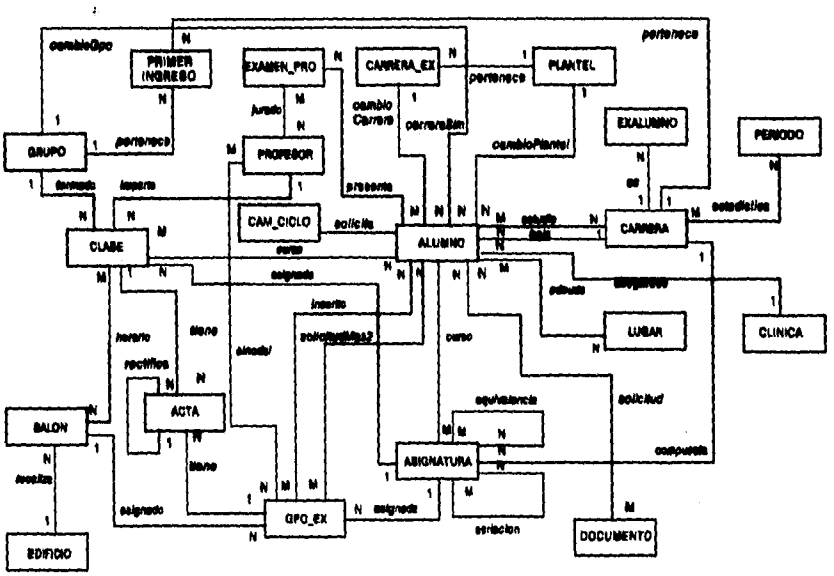
Suponga que a, b y c son tres atributos de una relación r, si c es funcionalmente dependiente de b y b lo es de a, entonces c es funcionalmente dependiente de a. A esta dependencia se le conoce como ***dependencia transitiva.***

Dentro de los beneficios que podemos encontrar con la aplicación de esta técnica tenemos:

- Nos libra de dependencias indeseables
- Se minimiza la reestructuración de datos
- Se mejora la dependencia de datos, permitiendo que las extensiones a la base de datos tengan pocos o ningún efecto sobre los programas o aplicaciones que la accesan.

En este trabajo sólo incluimos las entidades y relaciones ya normalizadas.

Diagrama de Entidad - Relación



ENTIDADES, RELACIONES Y ATRIBUTOS

A continuación mostramos la definición de las entidades y asociaciones presentadas en el DER. Unicamente presentamos aquellas relaciones para las cuales se obtiene una tabla en el Diseño Físico.

Para cada entidad y relación se listan sus atributos subrayando aquellos que formen parte de la llave primaria. Las llaves foráneas se indican en letra cursiva.

Para algunas entidades y relaciones, se definen algunos atributos duplicados o calculados, ya que se considera que beneficiarán el rendimiento de las consultas y operaciones que se lleven a cabo sobre la Base de Datos (redundancia necesaria).

Entidades y sus atributos

acta

cveAsig
cveCarr
entregada
folio
folioRec
gpo
tipo
rfc

alumno

- colonia
- cp
- delegacion
- domicilio
- edoCivil
- empresa
- entidad
- entidadProc
- fechaNac
- nacionalidad
- noCta
- nombre
- padre
- procedencia
- promedioProc
- sexo
- tel
- telOfna
- tipoProc
- tiempoTrans
- transporte

asignatura

- creditos
- cveAsig
- *cveCarr*
- nombre
- semestre
- tipo
- anioPlan

camCiclo
noCta

carrera
credObliga
credOpta
cveCarr
duracion
nivel
nombre
anioPlan
TotalAsig
cvePlantel

carreraEx
cveCarr
duracion
nombre

clase
cupo
cveAsig
cveCarr
grupo
vacantes

delegación
cveDel
nombre

documento
- cveDocto
- nombre

edificio
- cveEdificio
- descripcion

entidad
- cveEntidad
- nombre

exAlumno
- colonia
- cp
- *cveCarr*
- delegacion
- domicilio
- entidad
- ingreso
- nacionalidad
- noCta
- nombre
- regAlumno
- regExalumno
- sexo
- tel
- ultimoMov
- promedio

examenPro
- cveEx
- fecha

folio
tipo

grupo
cveCarr
gpo

grupoEx
cveAsig
cveCarr
fecha
gpo
noJurados
periodo
salon

lugar
cveLugar
lugar

período
fechaFin
fechaIni
periodo

plantel
cvePlantel
nombre

primerIngreso
colonia
comprobante

cp
cuotaLab
cuotaVol
cveCarr
domicilio
entidad
gpo
inscrito
noCta
nacionalidad
nombre
procedencia
ingreso
delegacion
edoCivil
empresa
entidadProc
fechaNac
padre
promedioProc
sexo
tel
telOfna
tipoProc
transporte
tiempoTrans
anioplan

profesor
nombre
dir
rfc

tel
telOfna
grado

salon
salon
ubicacion

Relaciones y sus atributos

adeuda
cveLugar
noCta

cambioCarrera
carreraDestino
cveCarr
noCta

cambioGpo
cveCarr
gpo
gpoSolicitado
noCta
autorizacion

cambioPlantel
cveCarr
cvePlantel
noCta

carreraSim

carreraSimultanea
cveCarr
noCta

cursa

cveAsig
cveCarr
gpo
noCta

curso

calificacion
cveAsig
cveCarr
folioActa
folioActaRec
gpo
noCta
periodo
tipoExamen

equivalencia

cveAsigAnterior
cveAsigEquiv
cveCarr

estadistica

La relación estadística en realidad no se compone de un conjunto de atributos, sino de un subconjunto de tablas con sus propios campos. Lo anterior es con el propósito de almacenar datos estadísticos del sistema. Cada tabla tiene su propia estructura, pero para este trabajo

no consideramos indispensable incluirlas ya que sus campos de alguno u otra manera forman parte de todas las tablas que se describen en este capítulo.

_estadistica11
_estadistica12
_estadistica13
_estadistica14
_estadistica15
_estadistica16
_estadistica17
_estadistica18
_estadistica21
_estadistica22
_estadistica23
_estadistica24
_estadistica25
_estadistica31
_estadistica32
_estadistica33
_estadistica34
_estadistica35
_estadistica36
_estadistica37
_estadistica416
_estadistica42
_estadistica43
_estadistica46
_estadistica51
_estadistica510
_estadistica52
_estadistica53

estudia

- aprobadas
- creditosOblig
- creditosOpta
- creditosRev
- cuotaLab
- cuotaVol
- *cveCarr*
- ingreso
- inscrito
- *noCta*
- nombre
- promedio
- regAlumno
- reprobadas
- reposición
- ultimoPeriodo
- semestre
- semestreAnterior
- sistema

horario

- *cveAsig*
- *cveCarr*
- *dia*
- *gpo*
- hora
- *salon*

imparte
- *cveAsig*
- *cveCarr*
- *gpo*
- rfc

inscrito
- *cveAsig*
- *cveCarr*
- *gpo*
- *periodo*
- *noCta*

jurado
- nombramiento
- *cveEx*
- *rfc*

presenta
- calificacion
- *cveCarr*
- fecha
- mencion
- *noCta*
- *cveEx*

solicitud
- *cveDocto*
- *noCta*
- fechaTram

solicitudMas2

autorizado
cveAsig
cveCarr
gpo
noCta
nomAutorizo
periodo

sInodal

cveAsig
cveCarr
anioPlan
gpo
rfc

seriacion

cveAsig
cveAsigReq
cveCarr
anioPlan

DESCRIPCIÓN DE LAS ENTIDADES Y SUS ATRIBUTOS

Para cada entidad se presenta una descripción de ella, si se trata de una entidad dependiente se indican las entidades de las que se depende y finalmente se describen cada uno de sus atributos.

Para cada atributo se presenta una breve descripción indicando y el tipo asociado. La cardinalidad Requerida indica que el atributo no puede tener valor nulo.

acta

Documento emitido por la Dirección General de Administración Escolar (DGAE) de la Universidad Nacional Autónoma de México en donde el profesor de una asignatura o los sinodales de un examen extraordinario asientan las calificaciones de los alumnos.

Descripción de atributos:

cveAsig
Clave de la asignatura.
Atributo alfanumérico de cuatro dígitos.
Cardinalidad: Requerida.

cveCarr
Clave de la carrera a la que pertenece la asignatura.
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

entregada
Atributo que indica si el acta ya fue entregada por el profesor o sinodal del grupo.
Atributo caracter que tiene los siguientes posibles valores:

N	no ha sido entregada.
S	ya fue entregada.

folio
Número de folio del acta en que se asienta la calificación. Cada hoja representa un acta con diferente folio.
Atributo numérico de 7 dígitos.
Cardinalidad: Requerida.

folioRec
Si el acta es de rectificación, este atributo hace referencia al folio del acta que se rectifica.
Atributo numérico de 7 dígitos.

gpo
Grupo en donde se imparte la asignatura o grupo de extraordinario.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

tipo
Este atributo indica si el acta es de rectificación o normal.
Atributo caracter, con los siguientes posibles valores:

N	acta normal.
R	acta de rectificación.

rfc
Registro Federal de Contribuyentes del profesor.
Atributo alfanumérico de 13 dígitos.

alumno
Estudiante de la Universidad Nacional Autónoma de México, que fue aceptado para estudiar alguna carrera de la Escuela Nacional de ENEP Iztacala.

Descripción de atributos:

colonia
Para alumnos que radican en el D.F., colonia de residencia. Para alumnos del interior de la República Mexicana, es el nombre de la población o ciudad.
Atributo alfanumérico de 30 caracteres.

cp
Código Postal del domicilio del alumno.
Atributo alfanumérico de 5 dígitos.

delegacion
Para alumnos que radican en el D.F., código de la delegación política en donde se encuentra su domicilio. Para alumnos del interior de la República Mexicana, es el código del municipio.
Atributo numérico de 2 dígitos.

domicilio
Lugar de residencia del alumno, está formado por el nombre de la calle, el número exterior y el número interior.
Atributo alfanumérico de 30 caracteres.

edoCivil
Estado civil del alumno.
Atributo caracter con el siguiente significado:

C	casado
D	divorciado
S	soltero
V	viudo
O	otro

empresa
Si el alumno trabaja, nombre legal de la Empresa en donde trabaja.
Atributo alfanumérico de 30 caracteres.

entidad
Código de la entidad federativa en donde se encuentra el domicilio del alumno.
Atributo numérico de 2 dígitos.

entidadProc
Código de la entidad federativa de la escuela de procedencia del alumno.
Atributo numérico de 2 dígitos.

fechaNac
Fecha de nacimiento del alumno.
Atributo tipo fecha

nacionalidad
Nacionalidad del alumno.
Atributo caracter con los siguientes posibles valores:

M	mexicano
E	extranjero

noCta
Número de Cuenta, Identificación de un alumno registrado en la UNAM. El Número de Cuenta está formado por 8 dígitos con el siguiente formato:
AAFFFFFV

donde:

AA	es el año lectivo en el que el alumno ingreso a la UNAM.
FFFFF	es el número de folio en el registro del período.
V	es un dígito verificador atributo alfanumérico.

Cardinalidad: Requerida

nombre
Nombre con el que el alumno queda registrado en la UNAM.
Atributo alfanumérico de 32 caracteres.
Cardinalidad: Requerida

padre
Nombre del padre o tutor del alumno
Atributo alfanumérico de 32 caracteres.

procedencia
Código de la escuela de procedencia del alumno.
Atributo alfanumérico de 2 dígitos.

promedioProc
Promedio obtenido por el alumno en su escuela de procedencia.
Atributo numérico real.

sexo
Sexo del alumno.
Atributo de tipo caracter:

M	masculino
F	femenino
blanco	no proporcionado

tel
Teléfono del domicilio del alumno.
Atributo alfanumérico de 10 caracteres.

telOfna
Si el alumno trabaja, teléfono de la empresa en donde se le puede localizar.
Atributo alfanumérico de 10 caracteres.

tipoProc
Caracter de la escuela de procedencia del alumno, puede ser:

O	oficial
P	particular
N	otro.

tiempoTrans
Tiempo que el alumno tarda en llegar de su casa a la escuela.
Atributo numérico de 2 dígitos

transporte
Medio principal de transporte que el alumno utiliza para llegar a la ENEP Iztacala.
Atributo de 4 caracteres los cuales pueden tener los siguientes valores.

M	metro
C	colectivo
P	particular
O	otro

asignatura

Asignatura que está o estuvo contemplada en el plan de estudios de alguna carrera de la ENEP Iztacala.

Descripción de atributos:

créditos
Valor académico de una asignatura, que representa el cursarla satisfactoriamente, de acuerdo a los planes de estudio vigentes.
Atributo numérico de 2 dígitos.
Cardinalidad: Requerida.

cveAsig
Clave única de una asignatura en una carrera.
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

cveCarr
Clave de la carrera a la que pertenece la asignatura.
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

nombre
Nombre de la asignatura.
Atributo alfanumérico de 28 caracteres.
Cardinalidad: Requerida.

semestre
Número de semestre al que pertenece la asignatura dentro de la carrera y plan correspondiente.
Atributo numérico de 2 dígitos.
Cardinalidad: Requerida.

tipo
Este atributo indica si la asignatura está catalogada como optativa u obligatoria, sus valores pueden ser:

O	Optativa
B	Obligatoria

Atributo de tipo caracter.

anioPlan
Año del plan de estudios .
Atributo numérico de 4 dígitos

camCiclo

Entidad dependiente en la que se registran los números de cuenta de los alumnos que lleven a cabo el trámite de cambio de ciclo. El cambio de ciclo solamente es válido para la carrera de Enfermería, y el cambio se hace a Bachillerato.

Descripción de atributos:

noCta
Número de cuenta del alumno de la carrera de Enfermería que solicita el cambio de ciclo.
Atributo alfanumérico de 8 dígitos.
Cardinalidad: Requerida.

carrera

Area profesional a nivel técnico o licenciatura que se ofrece en la ENEP Iztacala. Algunas carreras tienen diferentes planes de estudio, se considera carreras diferentes a cada uno de los planes que existan para una carrera.

credObliga
Número de créditos correspondientes a materias obligatorias que marca el plan de estudios correspondiente para acreditar la carrera.
Atributo numérico de 3 dígitos.
Cardinalidad: Requerida.

credOpta
Número de créditos de asignaturas optativas que marca el plan de estudios correspondiente para acreditar la carrera.
Atributo numérico de 3 dígitos.
Cardinalidad:Requerida

cveCarr
Identificador de una carrera en la ENEP Iztacala. Este atributo tiene la forma:
DDPP

donde:

DD	clave única de carrera
PP	número de plan

Para cada plan existirá una clave de carrera diferente, por lo que se considera como una carrera diferente.
Cardinalidad: Requerida.

duración
Número de semestres del que está compuesto el plan de la carrera.
Atributo numérico de 2 dígitos.
Cardinalidad: Requerida.

nivel
Nivel profesional de la carrera. Se consideran los siguientes valores:

L	licenciatura
T	técnico

Atributo tipo caracter.
Cardinalidad: Requerida.

nombre
Nombre de la carrera.
Atributo alfanumérico de 36 caracteres.
Cardinalidad: Requerida.

anioPlan
Año a partir del cual entro en vigor el plan de la carrera.
Atributo numérico de 4 dígitos.

totalAsig
Total de asignaturas de la carrera.
Atributo numérico de 4 dígitos.

cvePlantel
Clave del plantel.
Atributo alfanumérico de 4 dígitos.

carreraEx

Area profesional a nivel técnico o licenciatura que se ofrece en alguna escuela, facultad o instituto de la UNAM.

Descripción de atributos:

cveCarr
Identificador de una carrera en la UNAM. Es una cadena de 4 caracteres.
Cardinalidad: Requerida.

duración
Número de semestres del que esta compuesto el plan de la carrera.
Atributo numérico de 2 dígitos.
Cardinalidad: Requerida.

nombre
Nombre de la carrera.
Atributo alfanumérico de 36 caracteres.
Cardinalidad: Requerida.

clase

Clase a la cual los alumnos se inscriben para cursar una asignatura, dentro de un grupo.

Descripción de atributos:

cupo
Cupo estimado para la clase.
Atributo numérico de 3 dígitos.

cveAsig
Clave de la asignatura correspondiente.
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

cveCarr
Clave de la carrera a la que pertenece la asignatura.
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

grupo
Grupo al que pertenece la clase. Un grupo esta formado por las asignaturas correspondientes a un semestre de una carrera.
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

vacantes
Este atributo indica el número de vacantes en la clase. Inicialmente tiene un valor igual al atributo cupo.
Atributo numérico de 3 dígitos.

delegacion
Catálogo de las delegaciones políticas del Distrito Federal y municipios del interior de la República.

Descripción de atributos:

CveDel
Clave única de la delegación o municipio.
Atributo numérico entero de 2 dígitos.
Cardinalidad: Requerida.

nombre
Nombre de la delegación o municipio.
Atributo alfanumérico de 25 caracteres.
Cardinalidad: Requerida

documento

Esta entidad representa a los diferentes documentos que un alumno puede solicitar en Administración Escolar para algún propósito. En el sistema únicamente se necesita saber quien los solicita en el período actual y cuantos se solicitaron en períodos anteriores.

Descripción de atributos:

cveDocto
Identificador único asignado por el sistema.
Atributo numérico secuencial.
Cardinalidad: Requerida.

nombre

Nombre del documento. Este puede ser:

- Solicitud de Carrera Simultánea.
- Carta de buena conducta
- Certificado de estudios parciales
- Constancia de 100% de créditos
- Inscripción a especialidad médica
- Constancia de situación escolar
- Solicitud de credencial especial
- Solicitud de baja definitiva voluntaria
- Solicitud de inscripción al seguro facultativo del IMSS
- Solicitud de revisión de estudios
- Dictámenes
- Otro

Atributo alfanumérico de 60 caracteres.
Cardinalidad: Requerida.

edificio

Edificio o lugar físico donde puedan existir salones, laboratorios, aulas o un espacio destinado a la impartición de clases de alguna asignatura de las carreras de la ENEP Iztacala.

Descripción de atributos:

cveEdificio
Identificador único asignado por el sistema.
Atributo numérico entero secuencial.
Cardinalidad: Requerida.

descripción
Nombre o descripción del edificio.
Atributo alfanumérico de 40 caracteres.
Cardinalidad: Requerida.

entidad

Catálogo de las Entidades Federativas de la República Mexicana.

Descripción de atributos:

CveEntidad
Clave única de la Entidad Federativa.
Atributo numérico de 2 dígitos.
Cardinalidad: Requerida.

nombre
Nombre de la Entidad Federativa.
Atributo alfanumérico de 25 caracteres.
Cardinalidad: Requerida

exAlumno

Estudiante de la ENEP Iztacala, que ha terminado los estudios de una carrera, se ha dado de baja por algún motivo o bien no ha hecho ningún movimiento.

Descripción de atributos:

colonia
Para exalumnos que radican en el D.F., colonia de residencia. Para exalumnos del interior de la República Mexicana, es el nombre de la población o ciudad.
Atributo alfanumérico de 30 caracteres.

cp
Código Postal del domicilio del exalumno.
Atributo alfanumérico de 5 dígitos.

cveCarr
Clave de la carrera del exalumno, incluye el número de plan.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

delegación
Para exalumnos que radican en el D.F., código de la delegación política en donde se encuentra su domicilio. Para exalumnos del interior de la República Mexicana, es el código del municipio.
Atributo numérico de 2 dígitos.

domicilio
Lugar de residencia del exalumno, está formado por el nombre de la calle, el número exterior y el número interior.
Atributo alfanumérico de 30 caracteres.

entidad
Código de la entidad federativa en donde se encuentra el domicilio del exalumno.
Atributo numérico de 2 dígitos.

ingreso
Año de ingreso del exalumno a la ENEP Iztacala.
Atributo numérico de 2 dígitos.
Cardinalidad: Requerida.

nacionalidad
Nacionalidad del exalumno:

M	**mexicano**
E	**extranjero**

noCta
Número de Cuenta, Identificación del exalumno que adquirió cuando era alumno.

Atributo alfanumérico de 8 caracteres.
Cardinalidad: Requerida

nombre
Nombre con el que el alumno queda registrado en la UNAM.
Atributo alfanumérico de 32 caracteres.
Cardinalidad: Requerida

regAlumno
Clave utilizada por DGAE para el alta de alumnos al Sistema de Registro y Control Escolar vía primer ingreso.
Atributo alfanumérico de 2 dígitos, con valores en el rango cerrado 50-59.
Cardinalidad: Requerida

regExalumno
Clave utilizada por DGAE para la baja de alumnos al Sistema de Registro y Control Escolar.
Atributo alfanumérico de 2 dígitos.
Cardinalidad: Requerida

sexo
Sexo del exalumno.
Atributo de tipo caracter ('F', 'M', blanco).

tel
Teléfono del domicilio del exalumno.
Atributo alfanumérico de 10 caracteres.

ultimoMov
Ultimo período en el que se registró algún movimiento en la historia académica del exalumno.
Atributo alfanumérico de 3 dígitos.
Cardinalidad: Requerida.

promedio
Calificación promedio de las asignaturas acreditadas por el alumno.
Atributo numérico de 2 enteros y 2 decimales.

examenPro

Examen Profesional que un alumno que ha concluido los estudios de una carrera, presenta como requisito para la obtención de su título profesional.

Descripción de atributos:

cveEx
Clave interna del sistema para registrar los exámenes profesionales.
Atributo numérico secuencial.
Cardinalidad: Requerida.

fecha
Fecha en la que el alumno presentará su examen profesional.
Atributo de tipo fecha.

folio
Número de registro del trabajo de titulación que se presentará en el examen profesional.
Atributo numérico de 7 dígitos.
Cardinalidad: Requerida.

tipo
Tipo del examen profesional.
Atributo de tipo alfanumérico
O = examen profesional oral
B = examen profesional objetivo

grupo
La entidad grupo no hace referencia a la impartición de una asignatura en un horario, sino más bien a un conjunto de clases correspondientes a las asignaturas de un semestre de una carrera.

Descripción de atributos:

cveCarr
Identificador de una carrera en la ENEP Iztacala.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

gpo
Identificador de grupo.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

grupoEx

Grupo que se abre para inscripción a examen extraordinario en el período escolar actual. Los grupos de examen extraordinario pueden ser de dos tipos:

Examen extraordinario EA

Es el examen al que tienen derecho los alumnos cuando:

- Habiéndose inscrito en la asignatura, no hayan llenado los requisitos para acreditarla.
- Siendo alumnos, no hayan estado inscritos en la asignatura correspondiente, o no la hayan cursado.
- Habiendo estado inscritos dos veces en una asignatura, no pueden inscribirse nuevamente.
- Hayan llegado al límite de tiempo en que pueden estar inscritos en la Universidad.

Examen extraordinario EB
Se llama examen por Derecho de pasantes a aquél al que tienen derecho los alumnos que con dos exámenes terminan la carrera.

Descripción de atributos:

cveAsig
Clave única de una asignatura en una carrera.
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

cveCarr
Identificador de una carrera en la ENEP Iztacala.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

fecha
Fecha en la que se realizará el examen extraordinario. Incluye día y hora.
Atributo tipo fecha.

gpo
Identificador de grupo. Es un identificador alfanumérico de cuatro caracteres.
Cardinalidad:Requerida

noJurados
Atributo que indica el número de sinodales para el examen.
Calculo:
número de renglones en la tabla de relación *sinodal* pertenecientes a este examen extraordinario.
Atributo numérico de 1 dígito.

periodo
Período de exámenes extraordinarios en el semestre.
Atributo alfanumérico de 3 caracteres.
Cardinalidad: Requerida.

salón
Identificador de un salón en donde se pueden impartir asignaturas o aplicar exámenes extraordinarios.
Atributo alfanumérico de 6 caracteres.

lugar

Departamento, laboratorio, biblioteca, o lugar en el que se da un servicio de préstamo de material, y en el que el alumno se puede encontrar como deudor de material.

Descripción de atributos:

cveLugar
Identificador único asignado por el sistema.
Atributo numérico secuencial.
Cardinalidad: Requerida.

lugar
Nombre del lugar en donde se da el servicio de préstamo de material.
Atributo alfanumérico de 40 caracteres.
Cardinalidad: Requerida.

periodo

Período escolar en el cual se cursan asignaturas de las carreras que se imparten en la ENEP Iztacala. La duración de un período escolar es de un semestre.

Descripción de atributos:

fechaFin
Fecha en la que termina oficialmente el período escolar.
Atributo de tipo fecha.
Cardinalidad: Requerida.

fechaIni
Fecha en la que comienza oficialmente el período escolar.
Atributo de tipo fecha.
Cardinalidad: Requerida.

periodo
Identificador del período escolar. Con el siguiente formato:
AAN

donde:

AA	últimos dos dígitos del año.
N	número de semestre o trimestre en el año. En el caso de periodos anuales este dígito no aplica y vale 0.

Cardinalidad: Requerida.

plantel

Escuela, Facultad o Instituto en la Universidad Nacional Autónoma de México en donde se imparten carreras a nivel licenciatura o técnico.

Descripción de atributos:

cvePlantel
Código asignado por la UNAM a sus planteles.
Atributo alfanumérico de 3 caracteres.
Cardinalidad: Requerida.

nombre
Nombre con el que se designa al plantel en la UNAM.
Atributo alfanumérico de 60 caracteres.
Cardinalidad: Requerida.

primerIngreso

Estudiante de primer ingreso a la Universidad Nacional Autónoma de México, que fue aceptado para estudiar alguna carrera de la Escuela Nacional de Estudios Profesionales de Iztacala.

Descripción de atributos:

colonia
Para alumnos que radican en el D.F., colonia de residencia. Para alumnos del interior de la República Mexicana, es el nombre de la población o ciudad.
Atributo alfanumérico de 30 caracteres.

comprobante
Número del comprobante de aceptación que el alumno recibe en su domicilio.
Atributo numérico.
Cardinalidad: Requerida.

cp
Código Postal del domicilio del alumno.
Atributo alfanumérico de 5 dígitos.

cuotaLab
Cuota que el alumno aporta a la ENEP Iztacala por concepto de laboratorio.
Atributo numérico real.

cuotaVol
Cuota que el alumno aporta en forma voluntaria por concepto de su inscripción a la ENEP Iztacala.
Atributo numérico real.

cveCarr
Identificador de la carrera en la ENEP Iztacala en la que fue aceptado el alumno.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

delegación
Para alumnos que radican en el D.F., código de la delegación política en donde se encuentra su domicilio. Para alumnos del interior de la República Mexicana, es el código del municipio.
Atributo numérico de 2 dígitos.

domicilio
Lugar de residencia del alumno, está formado por el nombre de la calle, el número exterior y el número interior.

Atributo alfanumérico de 20 caracteres.

entidad
Código de la entidad federativa en donde se encuentra el domicilio del alumno.
Atributo numérico de 2 dígitos.

gpo
Identificador del grupo de primer semestre en el que está inscrito el alumno.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

inscrito
Atributo que indica si el alumno terminó los trámites de su inscripción en la ENEP Iztacala. Si el alumno no lleva a cabo estos trámites, posteriormente será dado de baja en DGAE.
Atributo caracter con los siguientes posibles valores:

S	si concluyó sus trámites
N	no concluyó sus trámites

nacionalidad
Nacionalidad del alumno.
Atributo caracter con los siguientes posibles valores:

M	mexicano
E	extranjero

noCta
Número de Cuenta, Identificación de un alumno registrado en la UNAM.
Atributo alfanumérico de 8 caracteres.
Cardinalidad: Requerida

nombre
Nombre con el que el alumno queda registrado en la UNAM.
Atributo alfanumérico de 32 caracteres.
Cardinalidad: Requerida

ingreso
Año de ingreso del alumno a la ENEP.

Atributo numérico de 4 dígitos.
Cardinalidad: Requerida

edoCivil
Estado civil del alumno.
Atributo caracter con el siguiente significado:

C	casado
D	divorciado
S	soltero
V	viudo
O	otro

empresa
Si el alumno trabaja, nombre legal de la Empresa en donde trabaja.
Atributo alfanumérico de 30 caracteres.

entidadProc
Código de la entidad federativa de la escuela de procedencia del alumno.
Atributo numérico de 2 dígitos.

fechaNac
Fecha de nacimiento del alumno.
Atributo tipo fecha

padre
Nombre del padre o tutor del alumno
Atributo alfanumérico de 32 caracteres.

promedioProc
Promedio obtenido por el alumno en su escuela de procedencia.
Atributo numérico real.

sexo
Sexo del alumno.
Atributo de tipo caracter:

M	masculino
F	femenino
blanco	no proporcionado

tel
Teléfono del domicilio del alumno.
Atributo alfanumérico de 10 caracteres.

telOfna
Si el alumno trabaja, teléfono de la empresa en donde se le puede localizar.
Atributo alfanumérico de 10 caracteres.

tipoProc
Caracter de la escuela de procedencia del alumno, puede ser:

O	oficial
P	particular
N	otro.

tiempoTrans
Tiempo que el alumno tarda en llegar de su casa a la escuela.
Atributo numérico de 2 dígitos

transporte
Medio principal de transporte que el alumno utiliza para llegar a la ENEP Iztacala.
Atributo de 4 caracteres los cuales pueden tener los siguientes posibles valores.

M	metro
C	colectivo
P	particular
O	otro

anioPlan
Año a partir del cual entró en vigor el plan de la carrera.
Atributo numérico de 4 dígitos.

procedencia
Nacionalidad de la escuela de procedencia.
Atributo caracter con los siguientes posibles valores:

M	mexicano
E	extranjero

tel
Teléfono del domicilio del alumno.
Atributo alfanumérico de 10 caracteres.

telOfna
Si el alumno trabaja, teléfono de la empresa en donde se le puede localizar.
Atributo alfanumérico de 10 caracteres.

tipoProc
Caracter de la escuela de procedencia del alumno, puede ser:

O	oficial
P	particular
N	otro.

tiempoTrans
Tiempo que el alumno tarda en llegar de su casa a la escuela.
Atributo numérico de 2 dígitos

transporte
Medio principal de transporte que el alumno utiliza para llegar a la ENEP Iztacala.
Atributo de 4 caracteres los cuales pueden tener los siguientes posibles valores.

M	metro
C	colectivo
P	particular
O	otro

anioPlan
Año a partir del cual entró en vigor el plan de la carrera.
Atributo numérico de 4 dígitos.

procedencia
Nacionalidad de la escuela de procedencia.
Atributo caracter con los siguientes posibles valores:

M	mexicano
E	extranjero

profesor

Personal de la ENEP Iztacala contratado en alguna plaza académica como profesor.

Descripción de atributos:

nombre
Nombre del profesor.
Atributo alfanumérico de 32 caracteres.
Cardinalidad: Requerida.

dir
Lugar de residencia del profesor, formado por calle, número y colonia.
Atributo alfanumérico de 40 caracteres.

rfc
Registro Federal de Causantes del profesor.
Atributo alfanumérico de 13 caracteres con el siguiente formato:
ABCDAAMMDDHHH

donde:

A	primer letra del primer apellido.
B	primer vocal del primer apellido.
C	primer letra del segundo apellido.
D	primer letra del primer nombre.
AA	dos últimos dígitos del año de nacimiento.
DD	día del mes de nacimiento.
HHH	homoclave.

Cardinalidad: Requerida.

tel
Teléfono del domicilio del profesor.
Atributo alfanumérico de 10 caracteres.

telOfna
Teléfono del lugar de trabajo del profesor.
Atributo alfanumérico de 10 caracteres.

grado
Nivel de estudios del profesor.
Atributo alfanumérico de 7 caracteres.

salón

Salón, aula, sala, laboratorio o lugar en donde se pueden impartir asignaturas de las carreras de la ENEP o aplicar exámenes extraordinarios.

Descripción de atributos:

salón
Identificador del salón.
Atributo alfanumérico de 6 caracteres con el siguiente formato:
Cardinalidad: Requerida.

ubicación
Identificador del edificio o lugar de la ubicación física del salón.
Atributo numérico de 2 dígitos.

DESCRIPCIÓN DE LAS RELACIONES Y SUS ATRIBUTOS

Esta sección complementa la anterior, detallando cada una de las características de las relaciones generadas del DER, así como sus atributos. Solamente se presentan las relaciones para las cuales se generará tabla en el Diseño Físico.

Para cada relación se presenta una descripción de ella, la cardinalidad requerida y finalmente se describen cada uno de sus atributos.

Para cada atributo se presenta una breve descripción indicando su longitud y el tipo asociado. La cardinalidad Requerida indica que el atributo no puede tener valor nulo.

adeuda

Indica los lugares en donde un alumno puede tener un adeudo.

Descripción de atributos:

cveLugar
Clave del lugar de adeudo.
Atributo numérico de 2 dígitos.
Cardinalidad: Requerida.

noCta
Número de cuenta del alumno que tiene el adeudo.
Atributo alfanumérico de 8 dígitos.
Cardinalidad: Requerida.

cambioCarrera

Esta relación mantiene la información de los alumnos que solicitan cambio de carrera.

Descripción de atributos:

carreraDestino
Clave de alguna carrera impartida en la UNAM a la que se desea cambiar el alumno.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

cveCarr
Clave de la carrera de la ENEP Iztacala a la cual está inscrito actualmente el alumno.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

noCta
Número de cuenta del alumno que solicita el cambio.

Atributo alfanumérico de 8 dígitos.
Cardinalidad: Requerida.

cambioGpo

Esta relación mantiene la información de los alumnos que solicitan cambio de grupo.

Descripción de atributos:

cveCarr
Clave de la carrera de la ENEP Iztacala a la cual está inscrito actualmente el alumno.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

gpo
Grupo de la carrera al que actualmente está inscrito el alumno.
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

gpoSolicitado
Grupo para el que se solicita el cambio.
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

noCta
Número de cuenta del alumno que solicita el cambio.
Atributo alfanumérico de 8 dígitos.
Cardinalidad: Requerida.

autorizacion
Indica si la solicitud de cambio de grupo ha sido autorizada.
Atributo que puede ser :

S autorizada
N no autorizada

cambioPlantel

Esta relación mantiene la información de los alumnos que solicitan cambio de plantel.

Descripción de atributos:

cveCarr
Clave de la carrera de la ENEP Iztacala a la cual está inscrito actualmente el alumno.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

cvePlantel
Plantel en la UNAM para el que se solicita el cambio. Debe de ser un plantel en donde se imparta la carrera a la que actualmente está inscrito el alumno.
Atributo alfanumérico de 3 caracteres.
Cardinalidad: Requerida.

noCta
Número de cuenta del alumno que solicita el cambio.
Atributo alfanumérico de 8 dígitos.
Cardinalidad: Requerida.

carreraSim

Esta relación mantiene la información de los alumnos que solicitan carrera simultánea.

Descripción de atributos:

carreraSimultanea
Clave de alguna carrera impartida en la UNAM que se desea cursar como simultánea.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

cveCarr
Clave de la carrera de la ENEP Iztacala a la cual está inscrito actualmente el alumno.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

noCta
Número de cuenta del alumno que solicita carrera simultánea
Atributo alfanumérico de 8 dígitos.
Cardinalidad: Requerida.

CURSA

Esta relación mantiene la información de las asignaturas que cursan actualmente los alumnos.

Descripción de atributos:

cveAsig
Clave de la asignatura que actualmente cursa el alumno.
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

cveCarr
Clave de la carrera a la que pertenece la asignatura.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

gpo
Grupo en el que se cursa la asignatura.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

noCta
Número de cuenta del alumno que cursa la asignatura.

Atributo alfanumérico de 8 dígitos.
Cardinalidad: Requerida.

cumo

Esta relación mantiene la información de las asignaturas que ha presentado el alumno, ya sea en examen ordinario o extraordinario. Esta relación refleja la Historia Académica del alumno en una carrera de la ENEP Iztacala.

Descripción de atributos:

calificación
Código que representa la última calificación obtenida por un alumno al presentar un examen ordinario o extraordinario de una asignatura.
Atributo alfanumérico de 2 caracteres, con los siguientes posibles valores:

0 - 10	calificación numérica
MB	muy bien
B	bien
S	suficiente
NA	no acreditada
NP	no presentada
AC	acreditada
RE	revalidada
CO	covalidada

Cardinalidad: Requerida.
cveAsig
Clave de la asignatura que el alumno ha presentado de alguna forma (ordinario, extraordinario o ambas).
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

cveCarr
Clave de la carrera a la que pertenece la asignatura.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

folioActa
Folio del acta en donde se acentó la última calificación de la asignatura.
Atributo numérico de 7 dígitos.
Cardinalidad: Requerida.

gpo
Grupo en el que se presentó por última vez la asignatura.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

noCta
Número de cuenta del alumno que cursó la asignatura.
Atributo alfanumérico de 8 dígitos.
Cardinalidad: Requerida.

periodo
Período escolar en el que se presentó la asignatura.
Atributo alfanumérico de 3 caracteres.
Cardinalidad: Requerida.

tipoExamen
Tipo de examen en donde se presentó la asignatura.
Atributo caracter con los siguientes posibles valores:

D	Extraordinario por derecho de pasante.
E	Extraordinario.
O	Ordinario.

Cardinalidad: Requerida

equivalencia

Esta relación mantiene la tabla de equivalencia de asignaturas en una carrera. Esta información es utilizada cuando se requiere de un cambio de plan de estudios.

Descripción de atributos:

cveAsigAnterior
Clave de la asignatura, para la cual se busca su equivalente en el plan al cual se va ha cambiar.
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

cveAsigEquiv
Clave de la asignatura que se tomará como equivalente en el nuevo plan.
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

cveCarr
Clave de la carrera a la que pertenece la asignatura.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

estadistica

Esta relación mantiene la información referente a las estadísticas semestrales por carrera que se obtienen referentes a varios tópicos.

estudia

Esta relación mantiene la información referente a la carrera que estudia un alumno. La mayor parte de los atributos son calculados o duplicados, esto con el propósito de obtener rápidamente parámetros que determinan el avance en la carrera.

Descripción de atributos:

aprobadas
Número de asignaturas de la carrera que han sido acreditadas de alguna forma (ordinario, extraordinario, revalidación, etc.).
Este atributo se obtiene sumando todas las asignaturas acreditadas registradas en la relación *curso*.

Atributo numérico de 2 dígitos.
Cardinalidad: Requerida.

creditosOblig
Número de créditos que han aportado las asignaturas obligatorias de la carrera que han sido acreditadas mediante examen ordinario o extraordinario.
Este atributo se obtiene sumando los créditos de todas las asignaturas obligatorias acreditadas registradas en la relación *curso*.
Atributo numérico de 3 dígitos.
Cardinalidad: Requerida.

creditosOpta
Número de créditos que han aportado las asignaturas optativas de la carrera que han sido acreditadas mediante examen ordinario extraordinario.
Este atributo se obtiene sumando los créditos de todas las asignaturas optativas acreditadas registradas en la relación *curso*.
Atributo numérico de 3 dígitos.
Cardinalidad: Requerida.

creditosRev
Número de créditos que han aportado las asignaturas de la carrera que han sido acreditadas mediante revalidación.
Este atributo se obtiene sumando los créditos de todas las asignaturas revalidadas acreditadas registradas en la relación *curso*.
Atributo numérico de 3 dígitos.
Cardinalidad: Requerida.

cuotaLab
Cuota que el alumno aporta a la ENEP Iztacala por concepto de laboratorio.
Atributo numérico real.

cuotaVol
Cuota que el alumno aporta en forma voluntaria por concepto de su inscripción a la ENEP Iztacala.
Atributo numérico real.

cveCarr
Clave de la carrera a la que pertenece la asignatura.

Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

ingreso
Año de ingreso del alumno a la ENEP Iztacala.
Atributo numérico de 4 dígitos.
Cardinalidad: Requerida.

inscrito
Atributo que indica si el alumno está inscrito, en el período actual, en ordinario en alguna de las asignaturas correspondientes a la carrera que cursa.
Atributo caracter que tiene los siguientes posibles valores:

0	no inscrito.
1	inscrito (registrado en la relación *inscrito*).

noCta
Número de cuenta del alumno que cursó la asignatura.
Atributo alfanumérico de 8 dígitos.
Cardinalidad: Requerida.

nombre
Nombre del alumno que cursa la carrera.
Atributo alfanumérico de 32 caracteres.
Cardinalidad: Requerida.

promedio
Atributo que indica el promedio que tiene actualmente el alumno en las asignaturas presentadas en examen ordinario o extraordinario.
Atributo numérico real en el rango 0.0 - 10.0 . Para la obtención del promedio se toma la siguiente tabla de conversión para las calificaciones obtenidas, registradas en la relación *curso*:

MB	10
B	8
S	6
NA	carece de equivalente
NP	carece de equivalente
RE	carece de equivalente

AC	carece de equivalente
CO	carece de equivalente

Cardinalidad: Requerida.

regAlumno
Clave utilizada por DGAE para el alta de alumnos al Sistema de Registro y Control Escolar vía primer ingreso.
Atributo alfanumérico de 2 dígitos.
Cardinalidad: Requerida

reposición
Atributo que indica cuantas reposiciones de credencial ha hecho el alumno. Solamente tiene derecho a tres.
Atributo numérico de 1 dígito.

reprobadas
Número de asignaturas de la carrera que han sido presentadas de alguna forma (ordinario, extraordinario, etc.); pero no han sido acreditadas.
Este atributo se obtiene sumando todas las asignaturas reprobadas registradas en la relación *curso*.
Atributo numérico de 2 dígitos.
Cardinalidad: Requerida.

ultimoPeriodo
Ultimo período escolar en que el alumno presentó asignaturas de sus carrera.
Atributo alfanumérico de 3 caracteres.

semestre
Es el semestre en el que el alumno se encuentra inscrito.
Atributo numérico de 2 dígitos.

semestreAnterior
Es el semestre anterior al que el alumno se encuentra inscrito.
Atributo numérico de 2 dígitos.

sistema
Indica si el alumno pertenece al sistema escolarizado o al de universidad abierta.
Atributo alfanumérico de 3 caracteres.

horario

Esta relación mantiene la información referente a los horarios y salones en los que se imparten las asignaturas correspondientes a las carreras de la ENEP Iztacala.

Descripción de atributos:

cveAsig
Clave de la asignatura.
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

cveCarr
Clave de la carrera a la que pertenece la asignatura.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

día
Número de día en el que se imparte la asignatura en ese salón en ese grupo.
Atributo numérico de 1 dígito.
Cardinalidad: Requerida.

gpo
Grupo en el que se imparte la asignatura.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

hora
Este atributo indica la hora a la que se imparte la asignatura.
Atributo de tipo fecha.

salón
Identificador del salón en el que se imparte la asignatura.
Atributo alfanumérico de 6 caracteres.

imparte

Esta relación mantiene la información referente a los profesores que imparten una materia. En un grupo, por cada asignatura puede haber hasta dos profesores.

Descripción de atributos:

cveAsig
Clave de la asignatura que se imparte en el grupo.
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

cveCarr
Clave de la carrera a la que pertenece la asignatura.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

gpo
Identificador del grupo.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

rfc
registro Federal de Contribuyentes del profesor.
Atributo alfanumérico de 13 caracteres.
Cardinalidad: Requerida.

inscrito

Esta relación mantiene la información de los alumnos inscritos a un grupo de examen extraordinario. El tipo de examen extraordinario puede ser EA o EB.

Descripción de atributos:

cveAsig
Clave de la asignatura a presentar en el examen extraordinario.

Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

cveCarr
Clave de la carrera a la que pertenece la asignatura.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

gpo
Identificador del grupo de examen extraordinario.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

noCta
Número de cuenta del alumno que presenta el examen.
Atributo alfanumérico de 8 dígitos.
Cardinalidad: Requerida.

periodo
Período escolar en el que se presentó la asignatura.
Atributo alfanumérico de 3 caracteres.
Cardinalidad: Requerida.

jurado

Esta relación mantiene la información de los jurados para un examen profesional.

Descripción de atributos:

nombramiento
Nombramiento que tiene el profesor en el examen profesional.
Atributo caracter con los siguientes posibles valores:

D	director
P	presidente
S	suplente
V	vocal

Cardinalidad: Requerida.

cveEx
Número de registro del Trabajo de Titulación que se presentará en el examen profesional.
Atributo numérico.
Cardinalidad: Requerida.

rfc
Registro Federal de Causantes del profesor que actuará como jurado en el examen profesional.
Atributo alfanumérico de 13 caracteres.
Cardinalidad: Requerida.

presenta

Esta relación mantiene la información de los alumnos que presentaron o presentarán su examen profesional.

Descripción de atributos:

calificación
Atributo que indica la calificación obtenida por el alumno en el examen profesional.
Atributo caracter con los siguientes posibles valores:

A	aprobado
E	emplazado

fecha
Fecha en la que el alumno presentó o presentará el examen profesional.
Atributo tipo fecha.
Cardinalidad: Requerida.

mención
Atributo que indica si el alumno obtuvo mención honorífica en el examen profesional.
Atributo caracter, con los siguientes posibles valores:

S	mención honorífica.
N	no obtuvo mención honorífica.

Cardinalidad: Requerida.

noCta
Número de cuenta del alumno que presenta el examen profesional.
Atributo alfanumérico de 8 caracteres.
Cardinalidad: Requerida.

cveEx
Número de registro de la Tesis que se presentará en el examen profesional.
Atributo numérico.
Cardinalidad: Requerida.

cveCarr
Clave de la carrera a la que pertenece la asignatura.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

solicitud

Esta relación mantiene la información de los alumnos que solicitan documentos en Administración Escolar para algún propósito.

Descripción de atributos:

cveDocto
Identificador del documento pedido.
Atributo numérico de 2 dígitos.
Cardinalidad: Requerida.

noCta
Número de cuenta del alumno que pide el documento.
Atributo alfanumérico de 8 caracteres.
Cardinalidad: Requerida.

fechaTram
Fecha en la que el alumno realiza el trámite.
Atributo de tipo fecha.

solicitudMas2

Esta relación mantiene la información de los alumnos que solicitan inscripción a más de 2 exámenes extraordinarios en un período de exámenes extraordinarios.

Descripción de atributos:

autorizado
Atributo que indica si el alumno tiene autorización para presentar el examen extraordinario.
Atributo caracter, con los siguientes posibles valores:

S	**si tiene autorización.**
N	**no tiene autorización.**

Cardinalidad: Requerida.

cveAsig
Clave de la asignatura del grupo de examen extraordinario.
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

cveCarr
Clave de la carrera a la que pertenece la asignatura.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

gpo
Identificador del grupo de examen extraordinario.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

noCta
Número de cuenta del alumno que solicita más de dos exámenes extraordinarios.

Atributo alfanumérico de 8 dígitos.
Cardinalidad: Requerida.

nomAutorizo
Nombre de la persona que autorizó la solicitud a más de 2 exámenes extraordinarios.
Atributo alfanumérico de 32 caracteres.

periodo
Período escolar en el que se presentó la asignatura.
Atributo alfanumérico de 3 caracteres.
Cardinalidad: Requerida.

seriación

Esta relación mantiene la tabla de seriación de asignaturas. Esta información es utilizada cuando se hacen las inscripciones a ordinario o extraordinario.

Descripción de atributos:

cveAsig
Clave de la asignatura para la cual se necesita haber aprobado una asignatura anterior.
Atributo alfanumérico de cuatro dígitos.
Cardinalidad: Requerida.

cveAsigReq
Clave de la asignatura que se debió haber aprobado para cursar la asignatura definida por el atributo *cveAsig*.
Atributo alfanumérico de cuatro dígitos.
Cardinalidad: Requerida.

cveCarr
Clave de la carrera a la que pertenece la asignatura.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

anioPlan
Año a partir del cual entró en vigor el plan de la carrera.
Atributo numérico de 4 dígitos.

sinodal

Esta relación mantiene la información de los profesores que son sinodales de algún examen extraordinario.

Descripción de atributos:

cveAsig
Clave de la asignatura del grupo de examen extraordinario.
Atributo alfanumérico de 4 dígitos.
Cardinalidad: Requerida.

cveCarr
Clave de la carrera a la que pertenece la asignatura.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

gpo
Identificador del grupo de examen extraordinario.
Atributo alfanumérico de 4 caracteres.
Cardinalidad: Requerida.

rfc
Registro Federal de Causantes del profesor asignado como sinodal.
Atributo alfanumérico de 13 caracteres.
Cardinalidad: Requerida.

CAPÍTULO V
DISEÑO

DIAGRAMA DE PROCESADOR

Este diagrama es un modelo que ilustra la elección del procesador y las interfaces entre procesadores.

Muestra las elecciones de tecnología del procesador que hayamos hecho para que puedan ser evaluadas.

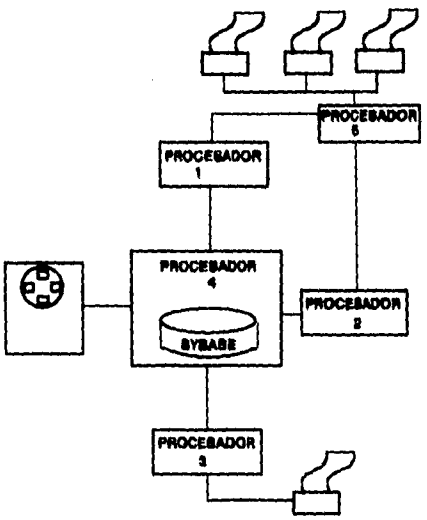
Un procesador es un mecanismo capaz de ejecutar instrucciones y almacenar datos. Esta definición incluye computadoras, personal u organizaciones y dispositivos de hardware.

Un procesador sencillo puede ejecutar porciones o procesos esenciales y almacenar datos utilizados por éstos.

Si varios procesadores tienen la misma tarea esencial, deberán representarse en el diagrama como uno solo.

Nos basamos en el diagrama de flujo de datos 0 para establecer qué procesos son ejecutados por el(los) procesador(es) y agruparlos.

Si un procesador es utilizado únicamente para transferir datos no deberá representarse en el diagrama.



PROCESADOR 1

Computadora 486 SX A 33 Mhz 8 MB en RAM, 210 MB en disco duro y monitor VGA color de 14".

Dentro de este procesador se consideran 5 computadoras que realizan los procesos de tramita registro, tramita ex_extraor. y tramita cambios.

PROCESADOR 2

Computadora 486 SX A 33 Mhz 8 MB en RAM, 210 MB en disco duro y monitor VGA color de 14".

En este procesador se consideran 3 computadoras en las que se realizan los procesos de registra sanciones académicas, emite estadísticas y controla actas.

PROCESADOR 3

Computadora 486 SX A 33 Mhz 8 MB en RAM, 210 MB en disco duro y monitor VGA color de 14".
En este procesador se consideran 2 computadoras en las que se realizan los procesos de tramita seguro facultativo, titulación y emite documentos.

PROCESADOR 4

Sparc station 5 procesador microSPAR-11 a 80 Mhz., con 1.05 GB en disco duro y 64 MB en memoria RAM, SOLARIS 2.x.

Este procesador es el servidor UNIX, el cual contiene un servidor de SYBASE en el que se encuentra la base de datos.

PROCESADOR 5

Computadora 386 sx a 25 Mhz, 4 Mb en RAM, 120 Mb en disco duro.

En este procesador se considera solo una compuatadora que será servidor de impresión

IMPRESORAS

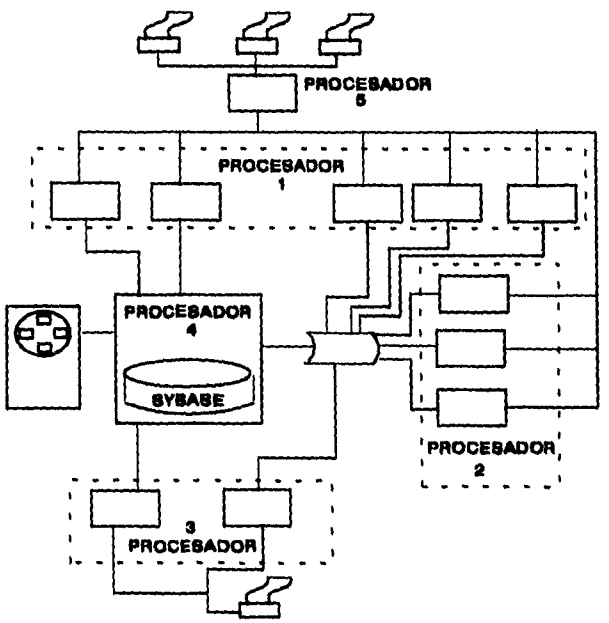
Se propone que se tengan 4 impresoras, las cuales estarán conectadas al servidor de impresión.

CONCENTRADOR

Será necesario un concentrador de 12 puertos debido a la distribución geográfica en la que se encuentra la unidad de Administración Escolar. En este concentrador se conectarán 7computadoras.

UNIDAD DE CINTA

Será necesaria para realizar los respaldos tanto de la base de datos como del log de transacciones.



PROTOTIPOS

Los prototipos no son una fase más del desarrollo, son una técnica que permite producir buenos y rápidos resultados. Un prototipo es una versión de la aplicación que cubre la mayoría de los requisitos de la bases de datos, es decir, es un modelo vivo de una parte o de todo un sistema.

Un prototipo no necesariamente debe ser perfecto, puede tener algunas fallas ya que no es el sistema terminado.

El objetivo de un prototipo es el de reducir el tiempo de desarrollo y lograr producir un producto que cumpla de la mejor manera con los requerimientos del usuario.

Asimismo, permite que exista una retroalimentación para el desarrollador y que el usuario evalúe cómo funciona el sistema, ya que le muestra todos los detalles que de otra manera, en los documentos de análisis es difícil describir.

Con el uso de prototipos los usuarios participan directamente con el analista, logrando con esto que exista un compromiso más estrecho. Puede ser verificado el desempeño que tendrá el sistema con datos reales y de esta manera realizar los cambios necesarios.

Es posible en poco tiempo darle al usuario un prototipo de la aplicación para que lo evalúe y lo pruebe. Dado que el tiempo de desarrollo de un prototipo es corto podemos, en caso de que éste no cumpla totalmente con las necesidades del usuario, realizar tantos prototipos como sea necesario hasta que tengamos la aprobación del mismo.

Para desarrollar un prototipo debemos utilizar un software apropiado. Los procesos pueden hacerse en lenguajes de cuarta generación, las tablas usando bases de datos relacionales y los diálogos con visualizador, que minimicen el uso de código y que permitan realizar pruebas con datos como por ejemplo Visual Basic, DataEase, PowerBuilder Alpha Four, etc. y no un lenguaje procedural como Cobol en el que es muy laborioso realizar pantallas, reportes, etc. Generalmente los prototipos se realizan en PC's ya que es más rápido, menos costoso y más portable.

Los prototipos son una herramienta de modelado muy efectiva, ya que permiten verificar en sistemas con muchas categorías de datos, relaciones entre estas categorías y elementos, es decir que el diseño de la base de datos sea correcto.

Asimismo, facilita el diseño de las pantallas cuando la secuencia entre éstas es muy compleja o tienen muchos datos.

Una vez que las pantallas, menú, reportes y relaciones están terminados debemos presentar el prototipo al usuario para que lo cheque y evalúe, ya que ellos podrán detectar con mayor facilidad las posibles fallas que pudiera tener.

DIAGRAMAS DE TRANSICIÓN DE ESTADO

Un diagrama de transición de estado es un modelo utilizado cuando ocurren eventos en un proceso o como consecuencia de la interacción con el usuario.

Un diagrama de transición de estado está compuesto de un estado, una transición, una condición de transición y una acción de transición.

Se dice que un estado es pasivo porque representa la espera de que ocurra algo. Un diagrama de transición de estado sólo puede representar un estado al mismo tiempo.

El nombre del estado representa qué es lo que el proceso está esperando, qué es lo que el proceso está realizando o qué será desplegado en las pantallas.

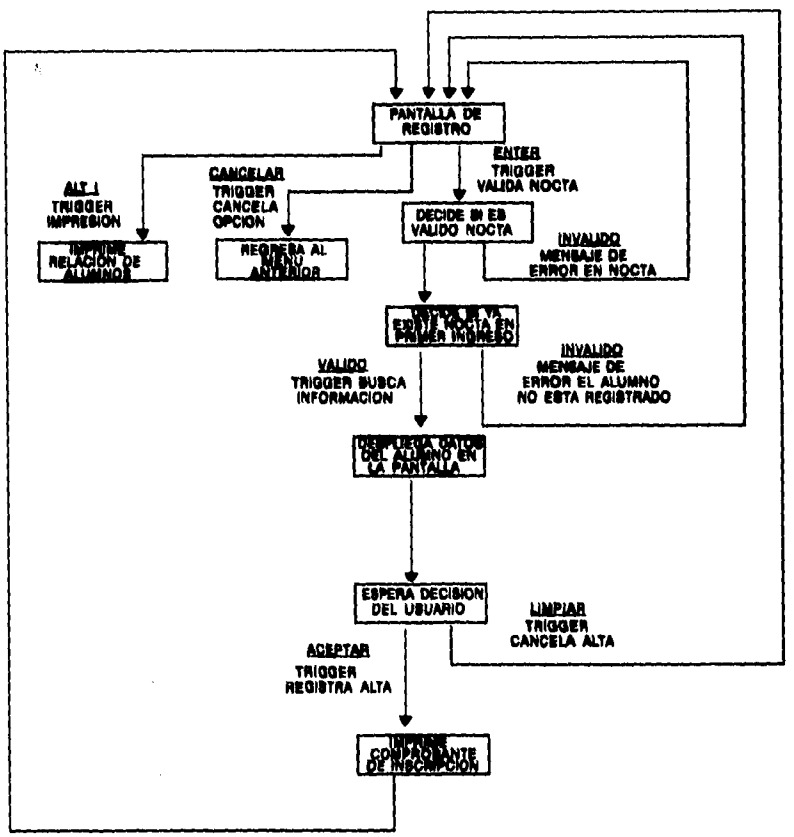
Un estado inicial es denotado por un transacción de entrada y describe lo que está ocurriendo antes de que se ejecute cualquier condición. Un estado final no tiene transiciones de salida y describe una situación, la cual no es posible cambiar.

Una transición representa un cambio de estado y está formada por la condición de transición y por la acción de transición.

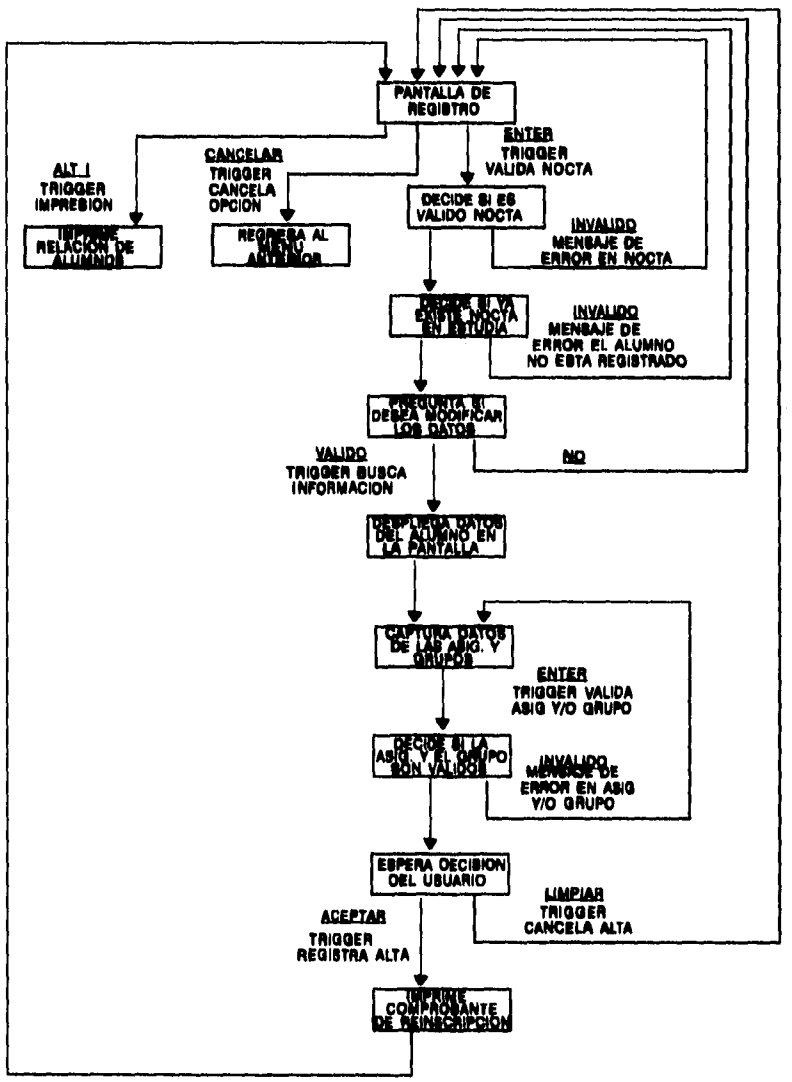
Múltiples transacción de o a un estado son permitidas y también es posible que una transición regrese al mismo estado del cuál se originó. Una condición de transición indica qué es lo que ocasionó que el sistema cambiara de un estado a otro. Una acción de transición ocurre cuando se cumple la condición. La acción puede causar procesos o señales que deberán ser ejecutados.

A continuación mostramos los diagramas de transición de estado del sistema ADESI.

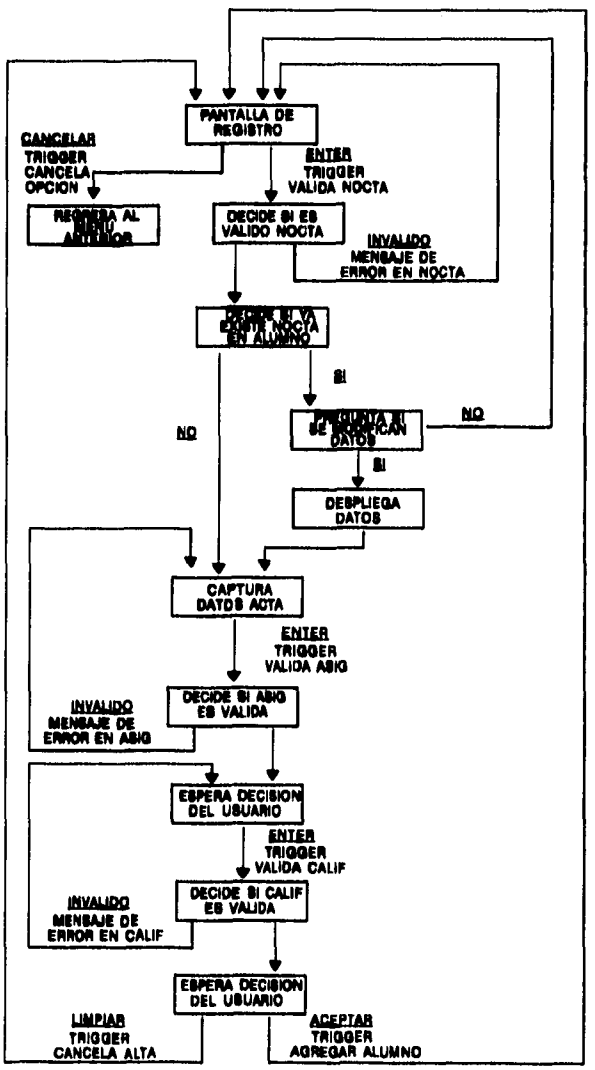
Controla Alumnos Inscritos



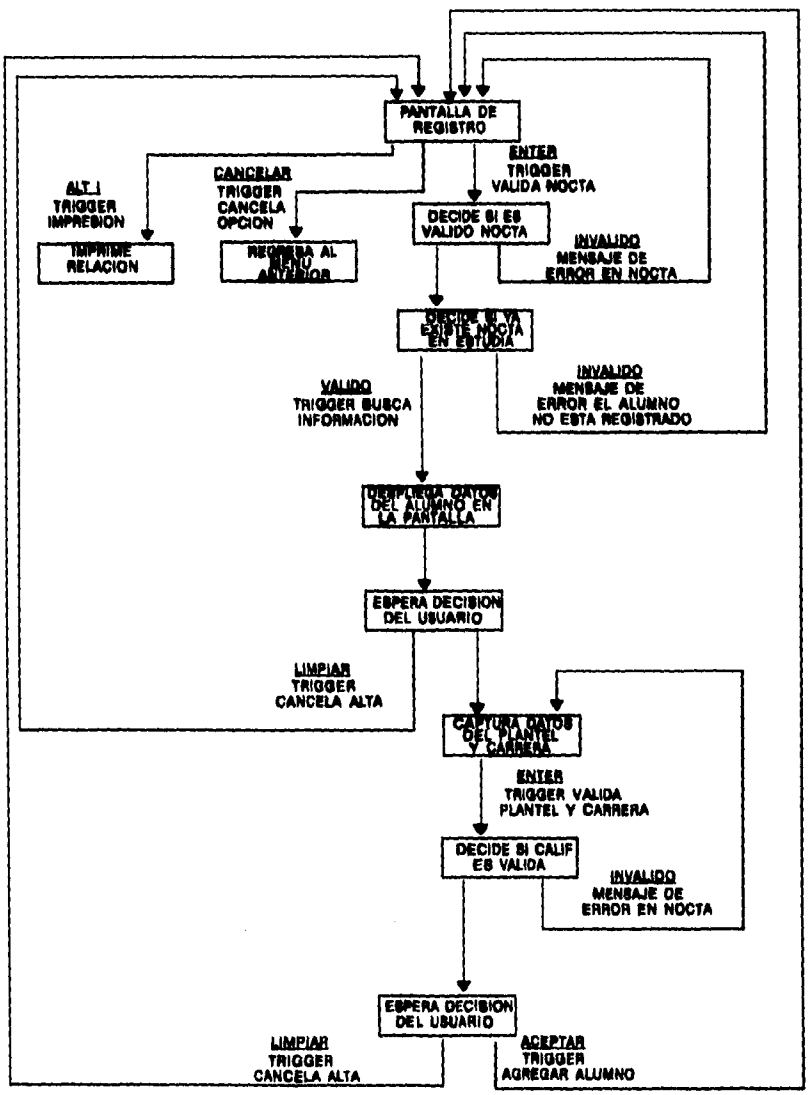
Controla Alumnos Reinscritos



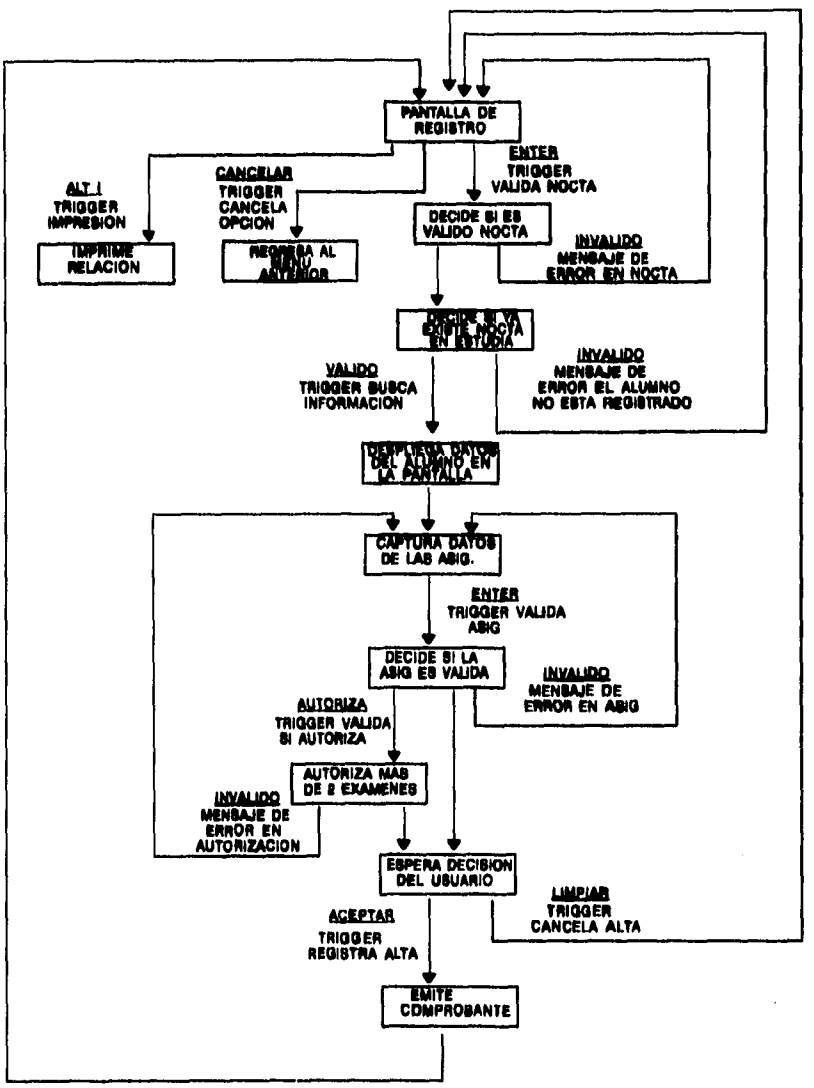
Ingreso a Años Posteriores



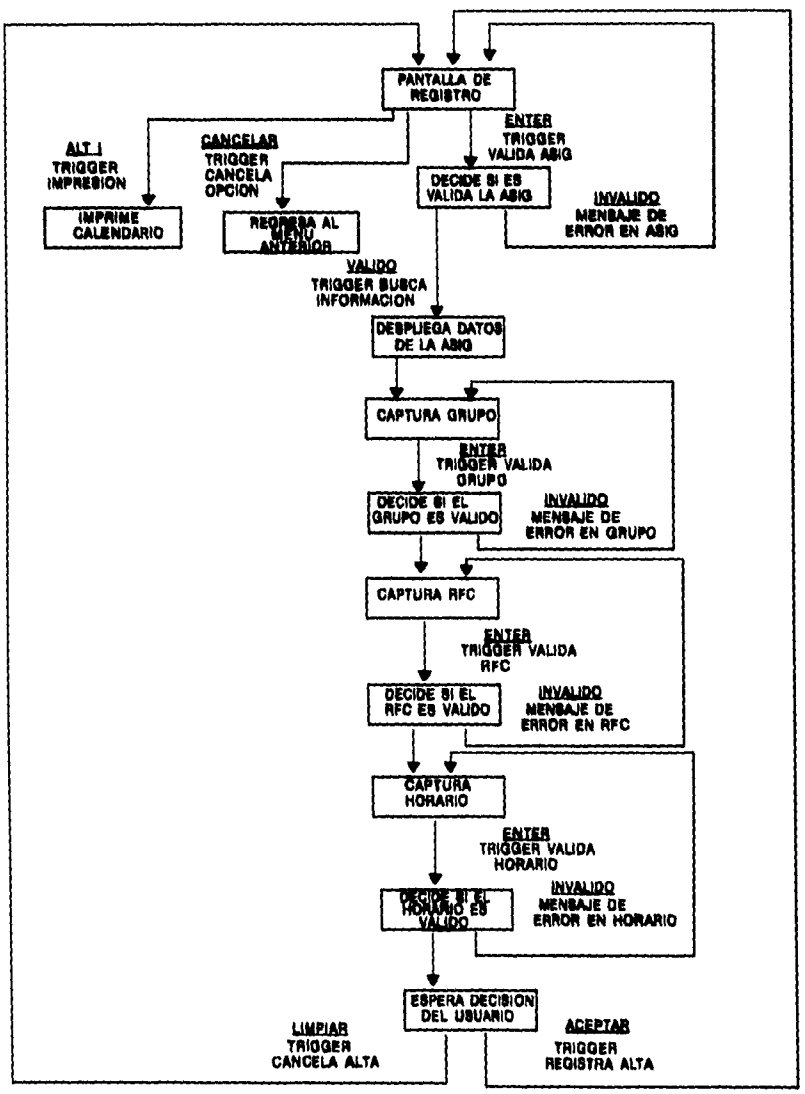
Carreras Posteriores



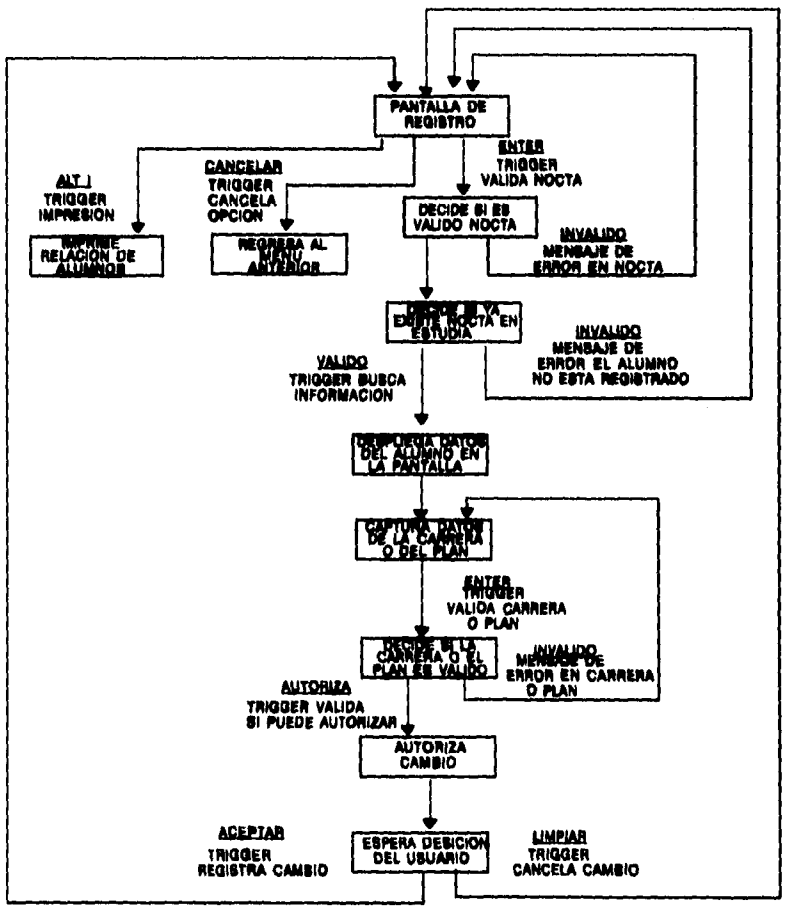
Exámenes Extraordinario



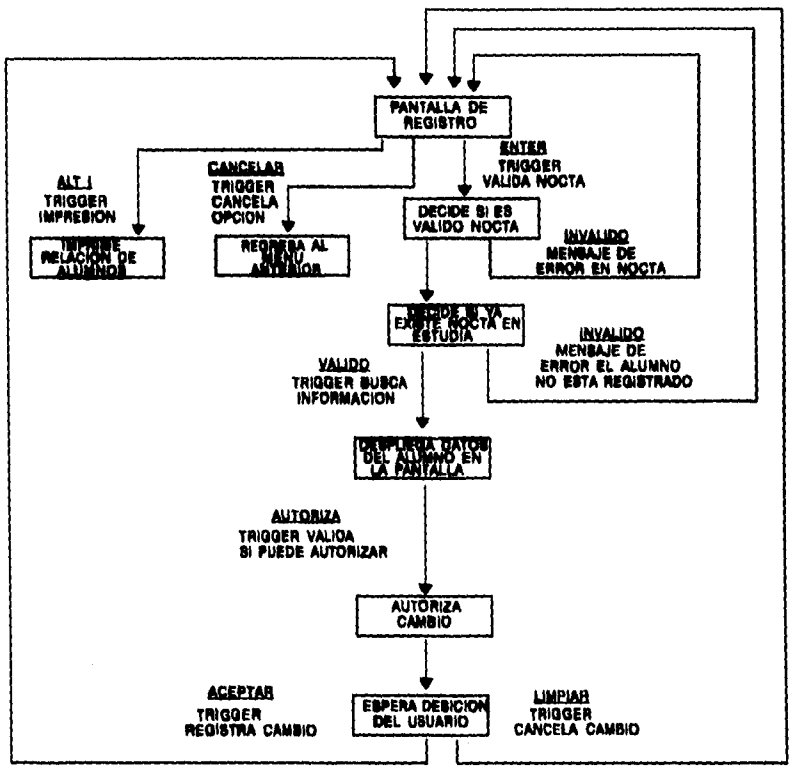
Calendario de Exámenes extraordinarios



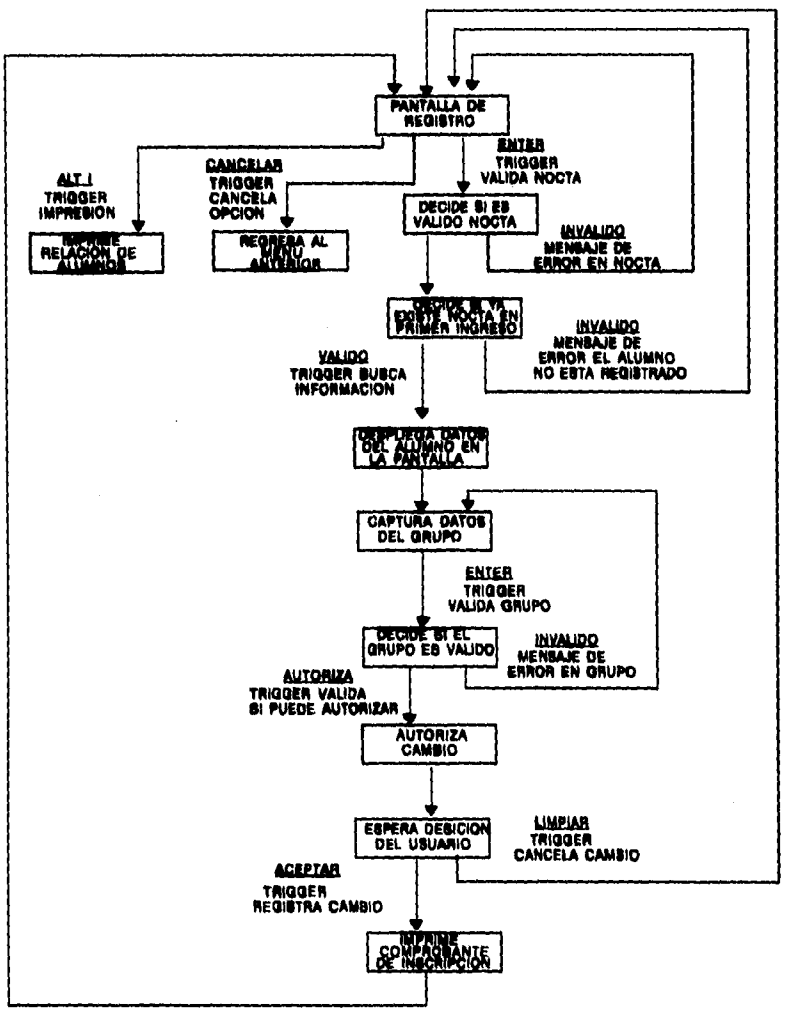
Cambio de Carrera o Plan de Estudios



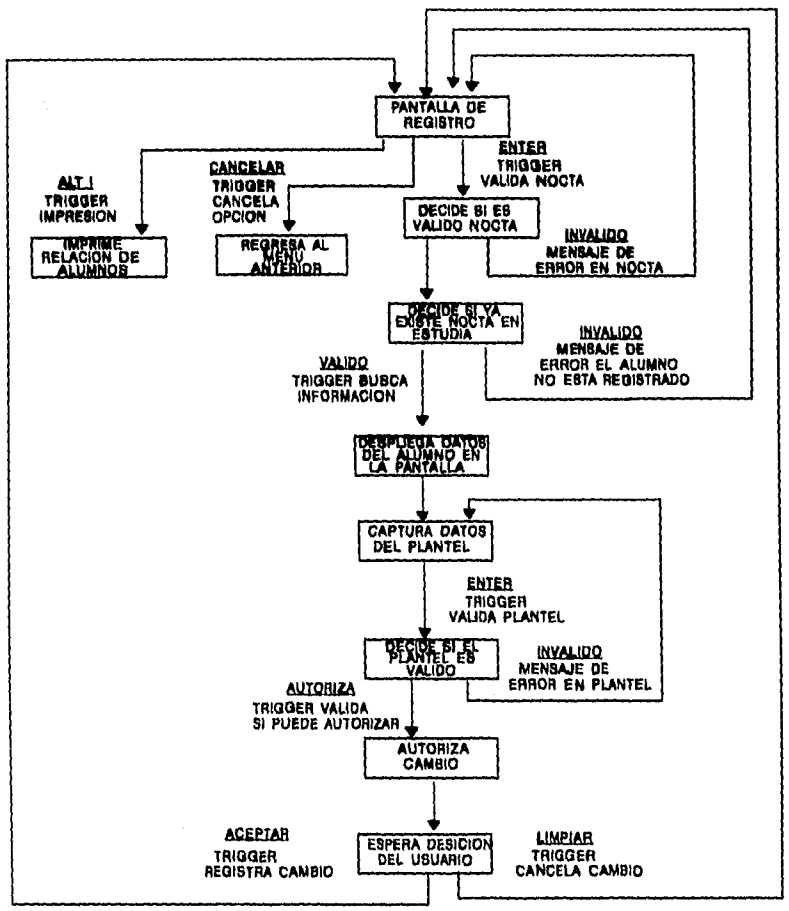
Cambio de Ciclo



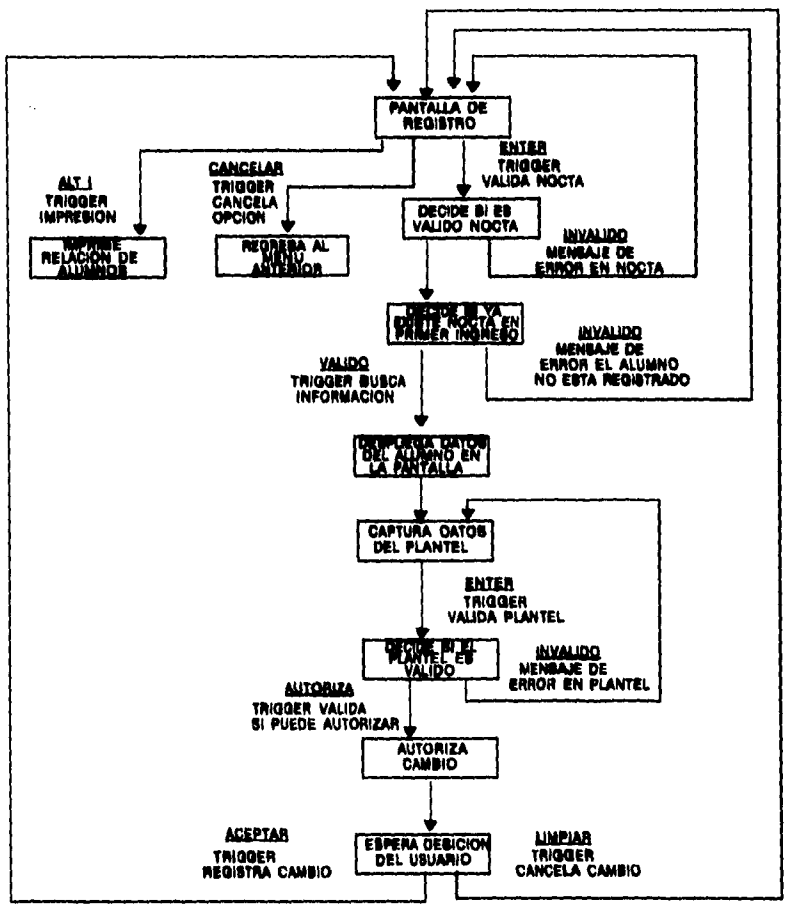
Cambio de Grupo



Cambio de Plantel Reingreso



Cambio de Plantel Primer Ingreso



DIAGRAMAS DE ESTRUCTURA

Son una herramienta que sirve para representar en forma gráfica las partes componentes de un proceso del sistema. Estas partes representadas son:

Módulos componentes del proceso
Conexiones entre los módulos
Jerarquía de los módulos

Un módulo es un grupo de instrucciones tratadas como una unidad y es invocado por su nombre desde otras unidades y que a su vez puede llamar a otros módulos.

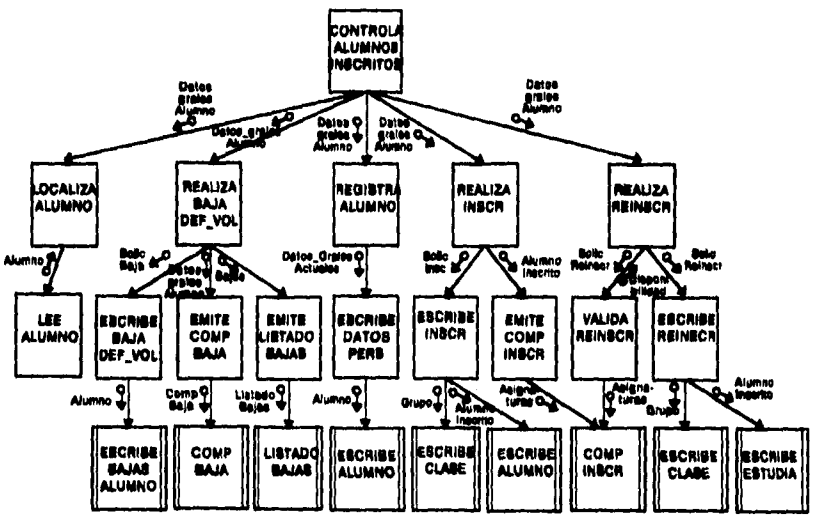
El proceso se organiza en módulos donde cada uno de éstos realice una función perfectamente bien definida y sea lo suficientemente pequeño para entenderse sin dificultades; estas características contribuyen a facilitar la implantación y el mantenimiento del sistema.

Para una buena organización de los módulos, el diseñador debe tener en mente una clara separación de lo que es el control y el trabajo en sí del proceso; asimismo, debe mantener una organización top-down de las funciones del proceso.

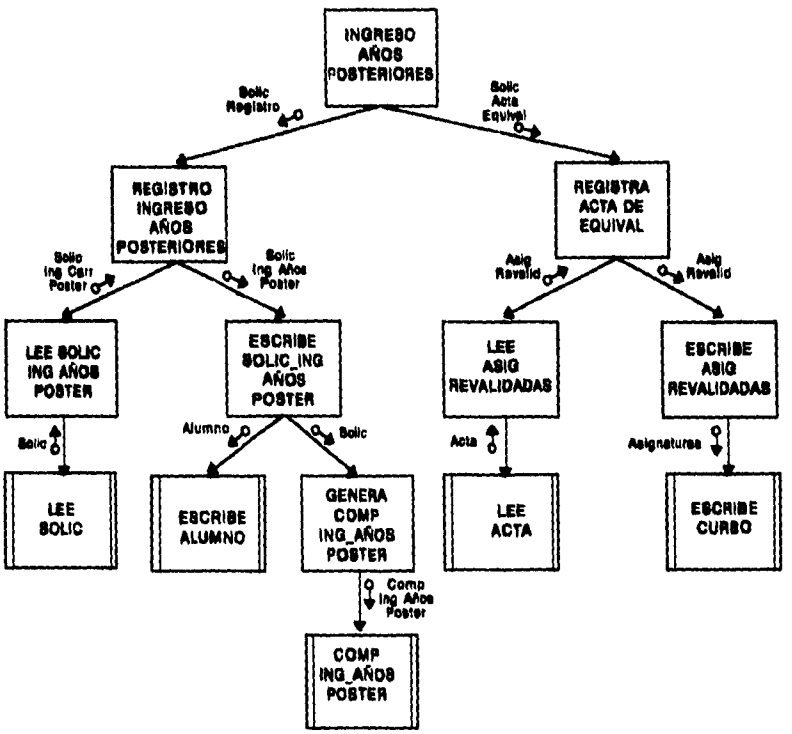
En todo diagrama de estructura los módulos se encuentran unidos por flechas, mismas que representan una llamada entre módulos y en cada llamada se puede presentar un paso de parámetros, ya sea de datos o de señales de control.

Los parámetros se representan con un círculo del cual se desprende una pequeña flecha. Si el círculo está vacío indica que el parámetro es uno o varios datos, de lo contrario se tratará de un parámetro de control. Finalmente, la pequeña flecha apunta del módulo que envía el parámetro hacia el que lo recibe.

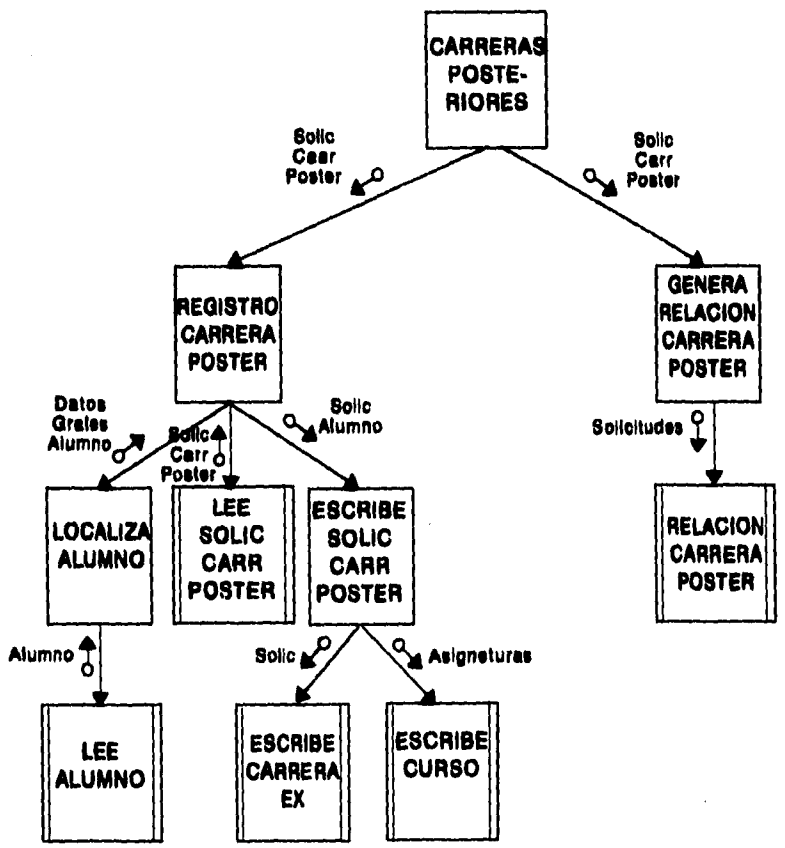
CONTROLA ALUMNOS INSCRITOS



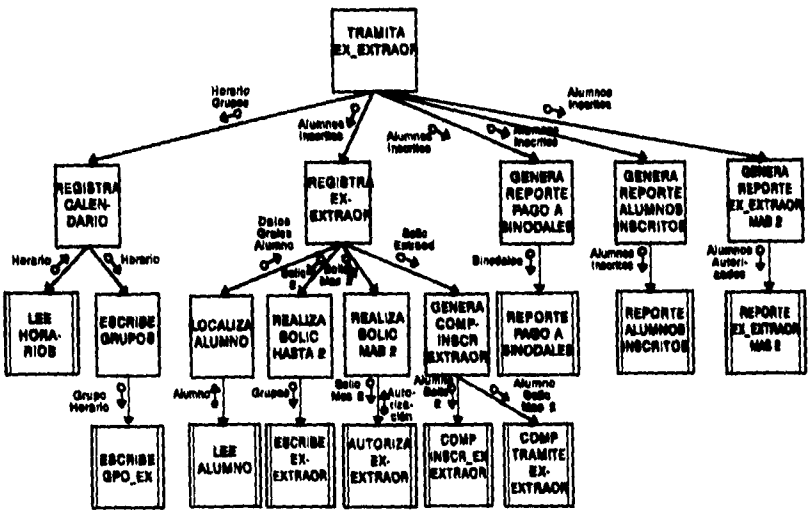
INGRESO A AÑOS POSTERIORES



CARRERAS POSTERIORES



TRAMITA EX_EXTRAOR



ESPECIFICACIÓN DE MÓDULOS

Se trata de una explicación que se elabora para cada módulo de un diagrama de estructura.

"Una especificación de módulo detallará las acciones requeridas en cada módulo en términos técnicos"

Estas explicaciones técnicas posteriormente se traducirán en código del programa final, por lo que el diseñador debe tomar en cuenta que el código del sistema debe ser flexible, de fácil mantenimiento, reusable e implantable.

1.1 CONTROLA ALUMNOS INSCRITOS

```
LOCALIZA ALUMNO(IDENTIFICA ALUMNO)
Si inscripción
    Si primer Ingreso
        REGISTRA INSCRIPCION(GRALES_ALUMNO)
    Otro
        VALIDA REINSCRIPCION(GRALES_ALUMNO)
        REGISTRA REINSCRIPCION(GRALES_ALUMNO)
    Fin si
    GENERA COMP_INSCRIP(SOLIC_COMP_INSCR)
    GENERA RELACION ALUMNOS INSCRITOS(SOLIC_REL_
        ALUMNO_REINSCR)
Otro
    Si baja Definitiva
        REGISTRA BAJA_DEF_VOL(GRALES_ALUMNO)
        GENERA COMP_BAJA(DATOS_BAJA)
        GENERA LISTADO_BAJAS(SOLIC_REL_BAJAS)
    Otro
        REGISTRA ALUMNO(DATOS_PERS)
    Fin Si
Fin Si
```

1.2.1 INGRESO AÑOS POSTERIORES

REALIZA INGRESO AÑOS POSTERIORES(SOLIC_ING_AÑOS
_POSTER)
REGISTRA ACTA DE EQUIVAL(SOLIC_ING_AÑOS_POSTER)

1.2.2 CARRERA POSTERIORES

LOCALIZA ALUMNO(IDENTIFICA ALUMNO)
LEE SOLIC CARR POSTER(SOLIC CARR POSTER)
ESCRIBE SOLIC CARR POSTER(GRALES_ALUMNO)
GENERA RELACION CARR POSTER(SOLIC_REL_CARR_POSTER)

2 TRAMITA EX_EXTRAOR
ESCRIBE CALENDARIO(CALENDARIO)
LOCALIZA ALUMNO(IDENTIFICA ALUMNO)
ESCRIBE EX_EXTRAOR(GRALES_ALUMNO)
Si más de 2 asignaturas
AUTORIZA EX_EXTRAOR(PERMISO_EXTRAOR_AUTOR)
Fin Si
GENERA COM_INSCR_EX_EXTRAOR(DATOS_EXTRAOR)
GENERA REPORTE PAGO A SINODALES(SOLIC_REP_PAGO_SINOD
_EXTRAOR)
GENERA REPORTE ALUMNOS_INSCRITOS(SOLIC_REP
_ALUMNO_INSC_ASIG_JURADO)
GENERA REPORTE EX_EXTRAOR MAS 2(SOLIC_REP_ALUMNO
_AUTO_MAS_2)

PROPUESTA DE SOFTWARE Y HARDWARE

Dentro de los pasos para desarrollar un sistema es importante considerar el software y plataforma en el que va a ser implantado.

Asimismo, debemos mencionar que dicha elección debe realizarse tomando en cuenta las expectativas que se tienen del sistema, en cuanto a tiempo de respuesta, seguridad de la información, una agradable interfaz para el usuario, facilidad para realizar el trabajo, etc. y por otro lado el costo que representa para la entidad.

Debido al avance tecnológico y al auge de arquitecturas abiertas, existen en el mercado diversos productos que pueden cubrir las necesidades de procesamiento, captura y presentación de la información e impresión, por lo que se vuelve necesario realizar un análisis de éstos para que al combinarlos se logre la configuración idónea.

En una institución educativa de la magnitud de la UNAM es importante considerar el costo y beneficio, además de la tendencia tecnológica de la misma, porque si bien es cierto que estamos observando una futura descentralización de las actividades de la administración escolar, no debemos olvidar que la ENEP es una escuela que pertenece a la Universidad por lo que debe regirse bajo cierta normatividad.

Para un esquema cliente/servidor, se pueden considerar diversos sistemas operativos, protocolos de red, herramientas de desarrollo y manejadores de bases de datos.

Tomando en cuenta que ADESI se pensó bajo una arquitectura cliente/servidor y después de realizar un análisis se decidió lo siguiente:

En cuanto al **sistema operativo** se pensó en UNIX ya que tiene las siguientes características:

- Es portable
- Existe en la mayoría de las plataformas de hardware
- La mayoría de los manejadores de bases de datos diseñados para arquitectura cliente/servidor lo presentan como el ideal
- Posee una gran cantidad de herramientas de desarrollo y administración
- Posee características que lo hacen un sistema operativo de alta seguridad
- Es ideal para un ambiente de redes heterogéneas
- Compatibilidad con sistemas de otros equipos de cómputo con que cuenta la UNAM

En lo que se refiere al **manejador de bases de datos** se optó por Sybase (versión System 10) ya que ha sido adoptado por la UNAM además de contar con las siguientes características:

- Proporciona una interfaz abierta que permite el desarrollo de aplicaciones por medio de una gran cantidad de herramientas de terceros.
- El SQL Server de Sybase está diseñado para trabajar en ambientes en línea.
- La tecnología del SQL Server (Procedimientos almacenados, triggers, manejo de transacciones, etc.) permite el desarrollo de aplicaciones de alto desempeño.

- Es un manejador de bases de datos adaptable a las necesidades, puede instalarse desde en una PC 386 hasta en equipos mucho más grandes.
- Existen versiones de los productos de Sybase para una gran variedad de plataformas.

Sistema de seguridad de SYBASE

Respaldo

El SQL Server de Sybase permite hacer respaldos de la Base de Datos y del log de transacciones.

El respaldo es una copia completa de la Base de Datos hacia un dispositivo de dump previamente definido, el cual, puede ser un archivo o un dispositivo de almacenamiento externo.

El respaldo del log de transacciones es un respaldo incremental, esto quiere decir, que cuando se hace el respaldo solamente se copia la información que existe desde el último respaldo hasta la fecha. Después del respaldo el log se borra.

Esto permite implantar esquemas de respaldo en donde cada semana, por ejemplo, se respalde la base de datos y en períodos más pequeños el log de transacciones.

Dispositivos de Espejo

Los dispositivos de bases de datos previamente definidos implican que toda la información que se graba en un dispositivo

espejeado también se graba en el dispositivo espejo, siempre y cuando los dispositivos físicos sean del mismo tipo. Si el dispositivo espejeado falla, el otro permite seguir trabajando con la información sin ningún problema.

Procedimiento de Recuperación

Este procedimiento se lleva a cabo automáticamente por el SQL Server al momento de levantarlo. Tiene como función principal asegurar la consistencia de las bases de datos.

Al levantar el SQL Server el procedimiento de recuperación (recovery) lee el log de transacciones de cada base de datos y aquellas que no hayan sido terminadas y que estén reflejadas en disco les da Roll Back; las que se terminaron y no se reflejaron en disco les da Roll Forward. Este mecanismo también se extiende para transacciones distribuidas que utilizan Two-Phase commit. Este procedimiento garantiza, una vez que se levanta el servidor, que en las bases de datos solamente se reflejen las transacciones que fueron concluidas satisfactoriamente antes de que el servidor se diera de baja.

En cuanto al **front-end** se decidió utilizar Visual Basic 3.0 para Windows Versión Profesional ya que cuenta con las siguientes características:

- Permite desarrollar aplicaciones gráficas en ambiente Windows de manera sencilla.
- Debido a que se basa en la programación orientada a eventos permite desarrollar aplicaciones de manera más rápida haciendo uso de las herramientas con las que cuenta.

- Es compatible con Sybase así como con los principales manejadores de bases de datos.
- Permite el uso de ODBC (Open DataBase Connectivity) facilitando la conexión con diversos tipos de servidores locales o remotos.
- A través de OLE (Object Linking and Embedding) podemos crear aplicaciones que hagan llamadas a otros productos bajo Windows.
- Incluye la herramienta Crystal Reports que nos permite crear reportes de manera rápida. Asimismo, podemos ejecutar reportes creados con otras herramientas.
- Permite desarrollar aplicaciones que incluyan ayuda en línea.
- Permite crear discos de instalación sobre un archivo ejecutable creando automáticamente el icono de grupo y elemento de trabajo dentro de windows.

En relación al equipo de cómputo se eligió la **plataforma Sun** para ser el servidor, misma que incluye el sistema operativo SOLARIS 2.3, y computadoras personales 80486 que serán los clientes.

A continuación presentamos los cuadros con el equipo de cómputo y el software propuesto para la implantación del sistema.

HARDWARE

CAN-TI-DAD	DESCRIPCIÓN
1	Sparc server 5 Procesador microSPARC-11 a 80 MHZ 1.05 GB en disco duro 32 MB en RAM
1	Unidad de respaldo 8 mm
1	CD ROM
11	Computadora HP con procesador 80486 SX 25 MHz 4 MB en RAM 170 MB en disco duro Monitor VGA color 14"
1	Computadora HP con procesador 80386 SX 25 MHZ 4 MB RAM 120 MB en disco duro (Servidor de impresión)
3	Impresoras Epson FX-1170 9 agujas, 15", puerto paralelo, 8 KB, 240 cps
1	Impresora HP LASER 4L resolución 300ppp, 4 pág. por minuto, HP PCL 5, 26 fuentes internas, 2 MB de memoria, alimentador de 100 hojas, puerto paralelo
1	UPS Deltec 2.4 KVA

SOFTWARE

DESCRIPCIÓN DE SOFTWARE
SQL Server
Data Workbench
Net Library
Visual Basic Professional
Report Writer

CAPÍTULO VI
PROGRAMACIÓN

TABLAS

Una base de datos está formada por varias tablas que contienen los datos, una tabla contiene columnas que pueden tener asociados tipos de datos, reglas, defaults e índices.

Una base de datos puede contener vistas, procedimientos y triggers, los cuales están basados en las tablas de la misma.

Para crear una tabla debemos considerar los siguientes puntos:

- Decidir las entidades que se van a utilizar y definir sus relaciones, lo anterior a través de un diagrama entidad relación.
- Especificar los atributos de cada entidad.
- Definir las tablas que formarán la base de datos y normalizarlas hasta una tercera forma normal.
- Definir el tipo y longitud de los datos que tiene cada tabla.
- Decidir cuáles columnas van a permitir valores nulos y cuáles no.

- Crear las tablas
- Definir en caso de ser necesario los defaults y/o reglas y asociarlos a las columnas que así lo requieran.
- Definir y crear los índices.

A continuación presentamos los scripts para crear la base de datos, en el cual se definen los tipos de datos y las tablas que utiliza el sistema.

```
/*    Script de creación de las tablas que conforman la  base de datos     */

print "Tipos de datos"
   exec sp_addtype SHORTINT, tinyint,    "not null"
   exec sp_addtype ENTERO,  smallint,   "not null"
   exec sp_addtype LONGINT,      int,   "not null"
   exec sp_addtype opcion, "char(1)",   "not null"
   exec sp_addtype fecha,   smalldatetime
go

print "Tabla _estadistica11"
create table _estadistica11 (
   generacion    char(4),
   biologo        smallint,
   enfermeria     smallint,
   medico         smallint,
   dentista       smallint,
   psicologia     smallint,
   optometria     smallint )
go

print "Tabla _estadistica12"
create table _estadistica12 (
   clave          char(2),
```

```
   periodo        char(3),
   bajas          smallint,
   cambios        smallint )
go

print "Tabla _estadistica13"
create table _estadistica13 (
   generacion        char(4),
   biolDeclara       smallint,
   enfeDeclara       smallint,
   mediDeclara       smallint,
   dentDeclara       smallint,
   psicDeclara       smallint,
   optoDeclara       smallint,
   biolAlumAsig      smallint,
   enfeAlumAsig      smallint,
   mediAlumAsig      smallint,
   dentAlumAsig      smallint,
   psicAlumAsig      smallint,
   optoAlumAsig      smallint,
   biolAlumInsc      smallint,
   enfeAlumInsc      smallint,
   mediAlumInsc      smallint,
   dentAlumInsc      smallint,
   psicAlumInsc      smallint,
   optoAlumInsc      smallint )
go

print "Tabla _estadistica14"
create table _estadistica14 (
   clave          char(2),
   periodo        char(3),
   semestre       tinyint,
   importe        real )
go

print "Tabla _estadistica15"
create table _estadistica15 (
```

```
  periodo       char(3),
  clave         char(2),
  grupo         char(4),
  cveAsig       char(4),
  inscritos     tinyint )
go

print "Tabla _estadistica17"
create table _estadistica17 (
  clave         char(2),
  periodo       char(3),
  semestre      tinyint,
  numAlumnos    int,
  importe       real )
go

print "Tabla _estadistica18"
create table _estadistica18 (
  periodo       char(3),
  biologo       smallint,
  enfermeria    smallint,
  medico        smallint,
  dentista      smallint,
  psicologia    smallint,
  optometria    smallint )
go

print "Tabla _estadistica21"
create table _estadistica21 (
  cveAsig       char(4),
  cveCarr       char(4),
  gpo           char(4),
  periodo       char(3),
  noCta         char(8) )
go

print "Tabla _estadistica22"
create table _estadistica22 (
```

```
   clave              char(2),
   carrera            varchar(30),
   periodo            char(3),
   inscritos          smallint,
   aprobados          smallint,
   reprobados         smallint )
go

print "Tabla _estadistica23"
create table _estadistica23 (
   clave              char(2),
   carrera            varchar(30),
   periodo            char(3),
   inscritosEA        smallint,
   inscritosEB        smallint )
go

print "Tabla _estadistica24"
create table _estadistica24 (
   clave              char(2),
   carrera            varchar(30),
   periodo            char(3),
   inscritos          smallint,
   aprobados          smallint,
   reprobados         smallint )
go


print "Tabla _estadistica25"
create table _estadistica25 (
   clave              char(2),
   carrera            varchar(30),
   periodo            char(3),
   inscritos          smallint,
   autorizados        smallint )
go

print "Tabla _estadistica31"
```

```
create table _estadistica31 (
   clave          char(2),
   periodo        char(3),
   egreso         tinyint,
   ingreso        tinyint )
go

print "Tabla _estadistica32"
create table _estadistica32 (
   clave                  char(2),
   periodo                char(3),
   concluyeron            tinyint,
   noConcluyeron          tinyint )
go

print "Tabla _estadistica33"
create table _estadistica33 (
   clave          char(2),
   periodo        char(3),
   Iztacala_CU    tinyint,
   Iztacala_FES   tinyint,
   CU_Iztacala    tinyint,
   FES_Iztacala   tinyint )
go


print "Tabla _estadistica34"
create table _estadistica34 (
   clave          char(2),
   periodo        char(3),
   FESEsc         tinyint,
   FESSua         tinyint,
   CUEsc          tinyint,
   CUSua          tinyint )
go

print "Tabla _estadistica35"
create table _estadistica35 (
```

```
  clave              char(2),
  periodo            char(3),
  numAlumnos         smallint )
go

print "Tabla _estadistica36"
create table _estadistica36 (
  clave              char(2),
  periodo            char(3),
  acreditados        smallint,
  revalidados        smallint )
go

print "Tabla _estadistica37"
create table _estadistica37 (
  clave              char(2),
  periodo            char(3),
  numAlumnos         smallint )
go

print "Tabla _estadistica416"
create table _estadistica416 (
  clave              char(2),
  carrera            char(30),
  mes01              smallint,
  mes02              smallint,
  mes03              smallint,
  mes04              smallint,
  mes05              smallint,
  mes06              smallint,
  mes07              smallint,
  mes08              smallint,
  mes09              smallint,
  mes10              smallint,
  mes11              smallint,
  mes12              smallint,
  anio               char(4),
  cveDoc             tinyint )
```

```
go

print "Tabla _estadistica42"
create table _estadistica42 (
  clave              char(2),
  periodo            char(3),
  generacion         char(4),
  numEgresados smallint,
  numHombres   smallint,
  numMujeres   smallint )
go

print "Tabla _estadistica43"
create table _estadistica43 (
   clave             char(2),
   anio              char(4),
   generacion        char(4),
   numAlumnos smallint )
go

print "Tabla _estadistica44"
create table _estadistica44 (
   clave             char(2),
   anio              char(4),
   generacion        char(4),
   numAlumnos  smallint )
go

print "Tabla _estadistica46"
create table _estadistica46(
   clave             char(2),
   carrera           varchar(30),
   unico             smallint,
   simultaneo        smallint,
   certificado       smallint,
   total             smallint,
   anio              char(6) )
go
```

```
print "Tabla _estadistica510"
create table _estadistica510 (
   clave          char(2),
   carrera        varchar(30),
   mes01          smallint,
   mes02          smallint,
   mes03          smallint,
   mes04          smallint,
   mes05          smallint,
   mes06          smallint,
   mes07          smallint,
   mes08          smallint,
   mes09          smallint,
   mes10          smallint,
   mes11          smallint,
   mes12          smallint,
   anio           char(4),
   otraEsc        smallint,
   entregado      smallint,
   noEntregado    smallint,
   bajas          smallint )
go

print "Tabla _estadistica51"
create table _estadistica51 (
   clave          char(2),
   periodo        char(3),
   tipoActa       opcion,
   numActas       smallint )
go

print "Tabla _estadistica52"
create table _estadistica52 (
   clave          char(2),
   carrera        varchar(30),
   noCta          char(8),
   generacion     char(4),
```

```
   promedio      float,
   anio          char(4))
go

print "Tabla _estadistica53"
create table _estadistica53 (
   clave         char(2),
   carrera       varchar(30),
   mes01         smallint,
   mes02         smallint,
   mes03         smallint,
   mes04         smallint,
   mes05         smallint,
   mes06         smallint,
   mes07         smallint,
   mes08         smallint,
   mes09         smallint,
   mes10         smallint,
   mes11         smallint,
   mes12         smallint,
   anio          char(4) )
go

print "Tabla _estadistica54"
create table _estadistica54 (
   clave         char(2),
   carrera       varchar(30),
   mes01         smallint,
   mes02         smallint,
   mes03         smallint,
   mes04         smallint,
   mes05         smallint,
   mes06         smallint,
   mes07         smallint,
   mes08         smallint,
   mes09         smallint,
   mes10         smallint,
   mes11         smallint,
```

```
   mes12          smallint,
   anio           char(4) )
go

print "Tabla _estadistica55"
create table _estadistica55 (
   clave          char(2),
   carrera        varchar(30),
   mes01          smallint,
   mes02          smallint,
   mes03          smallint,
   mes04          smallint,
   mes05          smallint,
   mes06          smallint,
   mes07          smallint,
   mes08          smallint,
   mes09          smallint,
   mes10          smallint,
   mes11          smallint,
   mes12          smallint,
   anio           char(4) )
go

print "Tabla _estadistica56"
create table _estadistica56 (
   clave          char(2),
   carrera        varchar(30),
   mes01          smallint,
   mes02          smallint,
   mes03          smallint,
   mes04          smallint,
   mes05          smallint,
   mes06          smallint,
   mes07          smallint,
   mes08          smallint,
   mes09          smallint,
   mes10          smallint,
   mes11          smallint,
```

```
   mes12          smallint,
   anio           char(4))
go

print "Tabla _estadistica57"
create table _estadistica57 (
   clave          char(2),
   carrera        varchar(30),
   mes01          smallint,
   mes02          smallint,
   mes03          smallint,
   mes04          smallint,
   mes05          smallint,
   mes06          smallint,
   mes07          smallint,
   mes08          smallint,
   mes09          smallint,
   mes10          smallint,
   mes11          smallint,
   mes12          smallint,
   anio           char(4) )
go

print "Tabla _estadistica58"
create table _estadistica58 (
   clave          char(2),
   carrera        varchar(30),
   mes01          smallint,
   mes02          smallint,
   mes03          smallint,
   mes04          smallint,
   mes05          smallint,
   mes06          smallint,
   mes07          smallint,
   mes08          smallint,
   mes09          smallint,
   mes10          smallint,
   mes11          smallint,
```

```
   mes12          smallint,
   anio           char(4) )
go

print "Tabla _estadistica59"
create table _estadistica59 (
   clave          char(4),
   carrera        varchar(30),
   generacion     char(4),
   primero        tinyint,
   segundo        tinyint,
   tercero        tinyint,
   anio           char(4),
   promedio       float )
go

print "Tabla _estadistica590"
create table _estadistica590 (
   clave          char(2),
   carrera        varchar(30),
   generacion     char(4),
   primero        tinyint,
   segundo        tinyint,
   tercero        tinyint,
   anio           char(4),
   promedio       float )
go

print "Tabla _permisos"
create table _permisos (
   acceso                  real,
   carrera                 opcion,
   control                 tinyint )
go

print "Tabla _sesiones"
create table _sesiones (
   control         tinyint,
```

```
  fechaEntrada  fecha,
  fechaSalida   fecha    null,
  numSesion     tinyint )
go

print "Tabla _usuarios"
create table _usuarios (
  clave          varchar(20)      null,
  control        tinyint,
  maxSesiones    tinyint,
  nombre         varchar(20))
go

print "Tabla acta"
create table acta(
  cveAsig       char(4),
  cveCarr       char(4),
  entregada     opcion,
  folio         LONGINT,
  folioRec      LONGINT,
  gpo           char(4),
  tipo          opcion,
  rfc           char(13) null )
go

print "Tabla adeuda"
create table adeuda (
  cveLugar      SHORTINT,
  noCta         char(8))
go

print "Tabla alumno"
create table alumno (
  colonia        varchar(30)      null,
  cp             char(5),
  delegacion     varchar(25),
  domicilio      varchar(30)      null,
  edoCivil       opcion,
```

```
  empresa        varchar(30)       null,
  entidad        SHORTINT,
  entidadProc    SHORTINT,
  fechaNac       fecha             null,
  nacionalidad   opcion,
  noCta          char(8),
  nombre         varchar(32),
  padre          varchar(32)   null,
  procedencia    varchar(30)   null,
  promedioProc   real,
  sexo           opcion,
  tel            varchar(10)   null,
  telOfna        varchar(10)   null,
  tipoProc       opcion,
  tiempoTrans    SHORTINT,
  transporte     opcion )
go

print "Tabla asegurado"
create table asegurado (
  cveClinica     SHORTINT,
  entregado      opcion,
  fechaAfiliacion fecha    null,
  fechaSol       fecha,
  movimiento     char(2),
  noAfiliacion   char(10)  null,
  noCta          char(8),
  vigencia       SHORTINT)
go

print "Tabla asignatura"
create table asignatura (
  creditos       tinyint,
  cveAsig        char(4),
  cveCarr        char(4),
  nombre         varchar(28),
  semestre       tinyint,
  tipo           opcion,
```

```
   anioPlan       smallint)
go

print "Tabla bajasAlumnos"
create table bajasAlumnos (
   cveCarr        char(4),
   noCta          char(8),
   fechaTram      datetime )
go

print "Tabla camCiclo"
create table camCiclo (
   noCta          char(8) )
go

print "Tabla cambioCarrera"
create table cambioCarrera (
   carreraDestino  char(4),
   cveCarr         char(4),
   noCta           char(8))
go

print "Tabla cambioGpo"
create table cambioGpo (
   cveCarr         char(4),
   gpo             char(4),
   gpoSolicitado   char(4),
   noCta           char(8),
   autorizacion    opcion null )
go

print "Tabla cambioPlantel"
create table cambioPlantel (
   cveCarr                 char(4),
   cvePlantel              char(3),
   noCta                   char(8),
   F305                    opcion     null,
   historialAcademico      opcion     null,
```

```
   comprobanteInscrip     opcion     null,
   tarjetonCredencial     opcion     null,
   cartaAceptacion        opcion     null,
   autorizado             pcion      null,
   sistemaSol             char(3)    null)
go

print "Tabla carrera"
create table carrera (
   credObliga     smallint,
   credOpta       smallint,
   cveCarr        char(4),
   duracion       SHORTINT,
   nivel          opcion,
   nombre         varchar(36),
   anioPlan       smallint,
   TotalAsig      tinyint,
   cvePlantel     char(4) )
go

print "Tabla carreraEx"
create table carreraEx (
   cveCarr        char(4),
   duracion       SHORTINT,
   nombre         varchar(36) )
go

print "Tabla carreraSim"
create table carreraSim (
   carreraSimultanea     char(4),
   cveCarr               char(4),
   noCta                 char(8) )
go

print "Tabla clase"
create table clase (
   cupo           SHORTINT,
   cveAsig        char(4),
```

```
  cveCarr       char(4),
  grupo         char(4),
  vacantes      SHORTINT )
go

print "Tabla clinica"
create table clinica (
  cveClinica    SHORTINT,
  direccion     varchar(60),
  nombre        varchar(40),
  tel           varchar(10) )
go

print "Tabla credencial"
create table credencial (
  noCta         char(8),
  cveCarr       char(4),
  tramitada     char(1) )
go

print "Tabla creditos"
create table creditos (
  cveCarr       char(4),
  semestre      tinyint,
  totCreditos   smallint,
  anioPlan      smallint )
go

print "Tabla cursa"
create table cursa (
  cveAsig     char(4),
  cveCarr     char(4),
  gpo         char(4),
  noCta       char(8) )
go
```

```
print "Tabla curso"
create table curso (
   calificacion     char(2),
   cveAsig          char(4),
   cveCarr          char(4),
   folioActa        LONGINT,
   folioActaRec     LONGINT,
   gpo              char(4),
   noCta            char(8),
   periodo          char(3),
   tipoExamen       opcion )
go

print "Tabla cursoEx"
create table cursoEx (
   calificacion     char(2),
   cveAsig          char(4),
   cveCarr          char(4),
   folioActa        int,
   gpo              char(4),
   noCta            char(8),
   periodo          char(3),
   tipoExamen       opcion,
   folioActaRec     int)
go

print "Tabla dia"
create table dia (
   dia              SHORTINT,
   nombre           char(10) )
go




print "Tabla documento"
create table documento (
```

```
  cveDocto      SHORTINT,
  nombre        varchar(60) )
go

print "Tabla edificio"
create table edificio (
  cveEdificio   SHORTINT,
  descripcion   varchar(40) )
go

print "Tabla entidad"
create table entidad (
  cveEntidad    SHORTINT,
  nombre        varchar(25) )
go

print "Tabla equivalencia"
create table equivalencia (
  cveAsigAnterior  char(4),
  cveAsigEquiv     char(4),
  cveCarr          char(4) )
go

print "Tabla escuela"
create table escuela (
  cveEscuela    SHORTINT,
  nombre        varchar(25) )
go

print "Tabla estudia"
create table estudia (
  anioPlan        smallint       null,
  aprobadas       tinyint,
  creditosOblig   smallint,
  creditosOpta    smallint,
  creditosRev     smallint,
  cuotaLab        float,
  cuotaVol        float,
```

```
    cveCarr          char(4),
    ingreso          smallint        null,
    inscrito         opcion,
    noCta            char(8),
    nombre           varchar(32),
    promedio         float,
    regAlumno        char(2),
    reprobadas       SHORTINT,
    reposicion       SHORTINT,
    ultimoPeriodo    char(3),
    semestre         tinyint         null,
    semestreAnterior tinyint         null,
    sistema          char(3)             null)
go

print "Tabla exAlumno"
create table exAlumno (
    colonia          varchar(30),
    cp               char(5),
    cveCarr          char(4),
    delegacion       varchar(25),
    domicilio        varchar(30),
    entidad          SHORTINT,
    ingreso          smallint,
    nacionalidad     opcion,
    noCta            char(8),
    nombre           varchar(32),
    regAlumno        char(2),
    regExalumno      char(2),
    sexo             opcion,
    tel              varchar(10),
    ultimoMov        char(3),
    promedio         float,
    anioPlan         smallint)
go

print "Tabla examenPro"
create table examenPro (
```

```
    cveEx           int,
    fecha           fecha  null,
    folio           int,
    tipo            opcion )
go

print "Tabla examenProObj"
create table examenProObj (
    cveEx       int,
    clinica     varchar(35) )
go

print "Tabla examenProOral"
create table examenProOral (
    asesor      varchar(32),
    cveEx       int,
    nombre      varchar(60) )
go

print "Tabla externo"
create table externo (
    cveEx               int,
    nombre              varchar(32),
    nombramiento        opcion )
go

print "Tabla grupoEx"
create table grupoEx (
    cveAsig     char(4),
    cveCarr     char(4),
    fecha       fecha,
    gpo         char(4),
    noJurados   smallint,
    periodo     char(3),
    salon       char(6) )
go

print "Tabla horario"
```

```
create table horario (
   cveAsig     char(4),
   cveCarr     char(4),
   dia          SHORTINT,
   gpo          char(4),
   hora         fecha,
   salon        char(6))
go

print "Tabla imparte"
create table imparte (
   cveAsig    char(4),
   cveCarr    char(4),
   gpo         char(4),
   rfc         char(13) )
go

print "Tabla inscrito"
create table inscrito (
   cveAsig     char(4),
   cveCarr     char(4),
   gpo          char(4),
   periodo     char(3),
   noCta       char(8) )
go


print "Tabla jurado"
create table jurado (
   nombramiento   opcion,
   cveEx           int,
   rfc             varchar(13) )
go

print "Tabla lugar"
create table lugar (
  cveLugar     tinyint,
```

```
    lugar           varchar(40) )
go

print "Tabla pagoLab"
create table pagoLab (
    clave           char(2),
    semestre        tinyint,
    importe         real )
go

print "Tabla pagoSinodal"
create table pagoSinodal (
    rfc             char(13),
    Exam1           smallint,
    Exam2           smallint )
go

print "Tabla periodo"
create table periodo (
    fechaFin        fecha,
    fechaIni        fecha,
    periodo         char(3) )
go


print "Tabla pertenece"
create table pertenece (
    cveCarr         char(4),
    cvePlantel      char(3) )
go

print "Tabla plantel"
create table plantel (
    cvePlantel      char(3),
    nombre          varchar(60) )
go
```

```
print "Tabla preAlumno"
create table preAlumno (
   cveCarr      char(4),
   nombre       varchar(32),
   numero       char(8) )
go

print "Tabla presenta"
create table presenta (
   calificacion  opcion,
   cveCarr       char(4),
   fecha         fecha,
   mencion       opcion,
   noCta         char(8),
   cveEx         int )
go

print "Tabla primerIngreso"
create table primerIngreso (
   colonia          varchar(30)    null,
   comprobante      int,
   cp               char(5),
   cuotaLab         real,
   cuotaVol         real,
   cveCarr          char(4),
   domicilio        varchar(20)    null,
   entidad          smallint,
   gpo              char(4),
   inscrito         opcion,
   noCta            char(8),
   nacionalidad     opcion,
   nombre           char(32),
   regAlumno        char(2),
   delegacion       varchar(25)    null,
   edoCivil         opcion         null,
   empresa          varchar(30)    null,
   entidadProc      tinyint        null,
   fechaNac         fecha          null,
```

```
    padre           varchar(32)      null,
    promedioProc    real             null,
    sexo            opcion           null,
    tel             char(10)null,
    telOfna         char(10)null,
    tipoProc        opcion           null,
    transporte      opcion           null,
    tiempoTrans     tinyint          null,
    anioplan        integer          null,
    ingreso         smallint null,
    sistema         char(3)          null)
go

print "Tabla profesor"
create table profesor (
    nombre      varchar(32),
    dir         varchar(40),
    rfc         char(13),
    tel         varchar(10),
    telOfna     varchar(10),
    grado       varchar(7)      null)
go

print "Tabla reportes"
create table reportes (
    control         tinyint,
    sesion          tinyint,
    parametro       char(20))
go

print "Tabla salon"
create table salon (
    salon           char(6),
    ubicacion       SHORTINT)
go

print "Tabla sancion"
create table sancion (
```

```
    cveSancion          ENTERO,
    noCta               char(8),
    tipo                char(1),
    fechaSancion        smalldatetime,
    descripcion         varchar(25)         null,
    fechaVencimiento    smalldatetime       null )
go

print "Tabla segundaCarr"
create table segundaCarr (
    segundaCarr     char(4),
    cveCarr         char(4),
    noCta           char(8) )
go

print "Tabla seriacion"
create table seriacion (
    cveAsig         char(4),
    cveAsigReq      char(4),
    cveCarr         char(4),
    anioPlan        smallint)
go

print "Tabla sinodal"
create table sinodal (
    cveAsig     char(4),
    cveCarr     char(4),
    gpo         char(4),
    rfc         char(13) )
go

print "Tabla soliMas2"
create table soliMas2 (
    cveAsig     char(4),
    cveCarr     char(4),
    gpo         char(4),
    periodo     char(3),
    noCta       char(8),
```

```
   hora          char(8) )
go

print "Tabla solicitud"
create table solicitud (
   cveDocto      tinyint,
   noCta         char(8),
   fechTram      datetime        null,
   cveCarr       char(4)         null )
go

print "Tabla solicitudMas2"
create table solicitudMas2 (
   autorizado    opcion,
   cveAsig       char(4),
   cveCarr       char(4),
   gpo           char(4),
   noCta         char(8),
   nomAutorizo   varchar(32),
   periodo       char(3) )
go
```

QUERIES

Son instrucciones de SQL utilizadas para realizar peticiones al servidor, sirven para consultar, modificar, borrar e insertar registros.

Gracias a las diversas cláusulas con las que cuenta SQL en los queries se pueden hacer validaciones, filtros, relaciones, etc. Podemos crear los objetos necesarios para que funcione una base de

datos tales como: la misma base de datos, tablas, índices, reglas, defaults, procedimientos almacenados, triggers, etc.

A continuación presentamos los queries utilizados en la forma de reinscripciones para accesar desde Visual Basic la base de datos del sistema.

Cuando el usuario teclea el número de cuenta del alumno:

Ejecuta el procedimiento almacenado DespPrimIngr, pasándole como parámetro el número de cuenta del alumno:

```
Set myset = mydb.CreateDynaset("DespPrimIngr " & pnoCta, 64)
```

Ejecuta el procedimiento almacenado RevisaInscrito, pasándole como parámetro el número de cuenta del alumno:

```
Set revisaInsc = mydb.CreateDynaset("RevisaInscrito " & pnoCta, 64)
```

Solicita todas las asignatura, y diversos datos de éstas, que pertenecen al grupo que el alumno eligió:

```
Set asigs = mydb.CreateSnapshot("select clase.cveAsig,
asignatura.nombre,
grupo,semestre from asignatura,clase where clase.grupo = """ &
myset("gpo")
 & """ and clase.cveCarr = """ & lblEscCarrera & """ And
clase.cveCarr = asignatura.cveCarr And clase.cveAsig =
asignatura.cveAsig", 64)
```

Al momento de dar click en Aceptar:

Inserta, en la tabla de cursa, una asignatura que el alumno desea cursar.

```
Set inscribir = mydb.CreateSnapshot("Insert into cursa values
```

("''' & lstCveAsig & '''", ''' & lblEscCarrera & '''", ''' & lstgrupo & '''',
''' & masknumCta & ''')", 64)

Solicita el número de vacantes de una asignatura en el grupo que el alumno desea registrarse:

Set inscribir = mydb.CreateSnapshot("select vacantes from clase
where cveAsig = ''' & lstCveAsig & ''' and cveCarr = ''' &
lblEscCarrera & '''
and grupo = ''' & lstgrupo & ''''', 64)

Actualiza el número de vacantes de la asignatura una vez que se registró la inscripción:

Set inscribir = mydb.CreateSnapshot("update clase set vacantes =
vacantes - 1
where cveAsig = ''' & lstCveAsig & ''' and cveCarr = ''' &
lblEscCarrera & '''
and grupo = ''' & lstgrupo & ''''', 64)

Ejecuta el procedimiento ActualizaInscrito, pasándole como parámetro el número de cuenta del alumno:

Set inscribir = mydb.CreateSnapshot("ActualizaInscrito " & pnoCta,
64)

Ejecuta el procedimiento ActCuotaVol, pasándole como parámetro el número de cuenta del alumno y la cantida que aportó:

Set myset = mydb.CreateDynaset("ActCuotaVol " & pnoCta & ", " &
coleg, 64)

Cabe mencionar que la parte resaltada en negrillas de las instrucciones anteriores, son la parte escencial del query, el resto pertenece al lenguaje de Visual Basic.

VISTAS

Una vista es una tabla virtual ya que sus características son las de una tabla normal con la diferencia de que ésta no almacena físicamente los datos, por lo que no ocupa espacio.

Una vista es una ventana de datos de una o más tablas reales. Generalmente son utilizadas para aumentar la seguridad, facilitar los queries, limitar a los usuarios en el acceso de cierta información, mantener la independencia lógica de los datos, etc.

Las vistas simpre van a estar asociadas a una o más tablas, si se borra alguna de las tablas a la cual está asociada ocurrirá un error de ejecución al tratar de utilizarla.

Es importante mencionar que cuando creamos una vista lo único que se almacena es la definición de la misma.

Presentamos el escript para crear la vista del historial académico que se utiliza en el sistema.

```
/* Definición de vistas */

print "Vista historial "
go

CREATE VIEW hist AS

SELECT a.semestre,c.cveAsig, a.tipo, a.creditos, nombreAsig = a.nombre,
    calificacion, tipoExamen, periodo, folioActa, gpo, nombreAlum =
    e.nombre, noCta=substring(parametro,1,8), nombrePlan=p.nombre,
```

```
    p.cvePlantel, e.ingreso, nombreCarr=cr.nombre, c.cveCarr,
    e.ultimoPeriodo, e.promedio, totoblig=e.creditosOblig,
    totopt=e.creditosOpta,cr.credObliga,cr.credOpta,totApr=aprobadas,
    totRep=reprobadas,control,sesion
FROM  reportes r,curso c, asignatura a, estudia e,plantel p, carrera cr
WHERE c.noCta=substring(r.parametro,1,8) and c.cveAsig=a.cveAsig and
    c.cveCarr=a.cveCarr and c.cveCarr=substring(parametro,9,4) and
    e.noCta=substring(parametro,1,8) and
    cr.cveCarr=substring(parametro,9,4) and cr.cvePlantel=p.cvePlantel
go
```

INTEGRIDAD

El término de integridad se refiere a los procesos de validar, revisar exactitud y mantener actualizadas las llaves primarias y formáneas de la base de datos.

El mantener la integridad es más bien responsabilidad del sistema y no del usuario. Con el fin de lograr esta tarea el sistema deberá ser proveido de elementos como por ejemplo reglas que sean aplicadas a los datos. Se deberán monitorear las operaciones que modifiquen la base de datos para confirmar que no violen ninguna de las reglas.

Las herramientas para mantener la integridad deberán ser especificadas como parte de la definición de la base de datos; de tal forma que estarán almacenadas en la misma y seerán utilizadas para controlar las modificaciones a las bases de datos.

ÍNDICES

Un ínidice es una estructura que está almacenada físicamente cuyo objetivo es el de cuidar la integridad de las llaves primarias y realizar las búsquedas más rápido.

Facilitan las búsquedas directas de un valor determinado dentro de una tabla ya que de otra manera tendría que recorrerse toda la tabla hasta encontrar el valor deseado.

Asimismo facilitan las búsquedas secuenciales ya que sin el índice se tendría que ordenar toda la tabla.

Podemos mencionar que algunas de las desventajas son que ocupan espacio en la base de datos y que así como facilitan las búsquedas también alentan los procesos de modificar, borrar o insertar datos ya que deberán realizarse tanto en la tabla como en el índice.

Al definir un índice podemos establecer que los valores de la columna, a la cual está asociado, sean únicos. Esto va a mantener la integridad de las llaves primarias impidiendo que existan valores duplicados.

A continuación presentamos el escript para crear los índices de la base de datos del sistema, podremos observar qué tablas tiene índices, de qué tipo son, si son únicos y a qué columna están asociados.

```
/*      Script para la creación de índices de la Base de Datos */
print "CREACION DE INDICES"
go
```

```
print "Indice _estadistica15Ind1"
create nonclustered index _estadistica15Ind1 on
estadistica15(periodo,clave,cveAsig)
go

print "Indice _estadistica21Ind1"
create nonclustered index _estadistica21Ind1 on
estadistica21(periodo,cveCarr,cveAsig,gpo)
go

print "Indice _permisosInd1"
create nonclustered index _permisosInd1 on permisos(control,carrera)
go

print "Indice _sesionesInd1"
create unique nonclustered index _sesionesInd1 on sesiones(control,numSesion)
go

print "Indice _usuariosInd1"
create unique clustered index _usuariosInd1 on usuarios(control)
go

print "Indice _usuariosInd2"
create unique nonclustered index _usuariosInd2 on usuarios(nombre)
go

print "Indice actaInd1"
create unique clustered index actaInd1 on acta(folio)
go

print "Indice alumnoInd1"
create unique clustered index alumnoInd1 on alumno(noCta)
go

print "Indice aseguradoInd1"
create unique clustered index aseguradoInd1 on asegurado(noCta)
go
```

```
print "Indice asigInd1"
create unique clustered index asigInd1 on asignatura(cveAsig, cveCarr)
go

print "Indice asigInd2"
create index asigInd2 on asignatura(cveCarr,semestre)
go

print "Indice claseInd1"
create unique clustered index claseInd1 on clase(cveCarr, cveAsig, grupo)
go

print "Indice cursaInd1"
create clustered index cursaInd1 on cursa(gpo,cveCarr,cveAsig)
go

print "Indice cursoInd1"
create clustered index cursoInd1 on curso(noCta,cveCarr,cveAsig,periodo)
go

print "Indice cursoInd2"
create nonclustered index cursoInd2 on curso(periodo,cveCarr,gpo)
go

print "Indice cursoExInd1"
create clustered index cursoExInd1 on cursoEx(noCta,cveCarr,cveAsig,periodo)
go

print "Indice estudiaInd1"
create unique clustered index estudiaInd1 on estudia(noCta, cveCarr)
go

print "Indice estudiaInd2"
create index estudiaInd2 on estudia(promedio)
go

print "Indice exAluInd1"
```

```
create unique clustered index exAluInd1 on exAlumno(noCta, cveCarr)
go

print "Indice examenProInd1"
create unique clustered index examenProInd1 on examenPro(cveEx)
go

print "Indice examenProInd2"
create index examenProInd2 on examenPro(fecha)
go

print "Indice examenProOralInd1"
create unique clustered index examenProOralInd1 on examenProOral(cveEx)
go

print "Indice grupoExInd1"
create unique clustered index grupoExInd1 on
grupoEx(periodo,cveCarr,cveAsig,gpo)
go

print "Indice imparteInd1"
create unique clustered index imparteInd1 on imparte(cveAsig,cveCarr,gpo)
go

print "Indice inscritoInd1"
create unique nonclustered index inscritoInd1 on
inscrito(periodo,cveCarr,cveAsig,gpo,noCta)
go

print "Indice primerIngInd1"
create unique clustered index primerIngInd1 on primerIngreso(comprobante)
go

print "Indice primerIngInd2"
create unique index primerIngInd2 on primerIngreso(noCta)
go

print "Indice primerIngInd3"
```

```
create nonclustered index primerIngInd3 on primerIngreso(cveCarr)
go


print "Indice profesorInd1"
create unique clustered index profesorInd1 on profesor(rfc)
go

print "Indice seriacionInd1"
create clustered index seriacionInd1 on seriacion(cveCarr,cveAsig)
go
```

REGLAS Y DEFAULTS

Una regla puede ser una lista de valores, un rango específico o un valor determinado y deberá estar asociada a una columna de una o varias tablas.

Una regla nos permite controlar los valores que serán insertados en una tabla validando que cumplan con la condición que ha sido establecida en ella.

Primeramente debemos crear una regla y ya que ha sido creada deberemos asociarla a la columna de la tabla. No podemos borrar una regla que no ha sido desasociada de la tabla.

Un default especificará un valor determinado cuando se desee insertar un registro y el usuario no proporcione todos los datos. Esto se hace con el fin de que si no se proporciona un dato cuyo campo no acepte valores nulos, al momento de insertar no nos

marque un error ya que el default proporcionará un valor definido anteriormente.

Así como con las reglas primero debemos crear el default y luego asociarlo a la columna de la tabla y en caso de querer borrarlo debemos desasociarlo.

A continuación presentamos el script para crear los defaults y reglas de la base de datos del sistema.

```
/*    Script de creación de las reglas y defults   */

print "Defaults"
go
create default autorizado          as "N"
go
create default cero                 as 0
go
create default ceroReal             as 0.0
go
create default codigoP              as "00000"
go
create default entidadD             as 0
go
create default entidadDef           as 00
go
create default entrega_def          as "N"
go
create default entregadoDef         as "N"
go
create default estadoCivil           as "S"
go
create default ingresoAnio          as 1995
go
create default inscritoDef           as "N"
go
```

```
create default mencion                as "N"
go
create default movimiento             as "TA"
go
create default nacionalidad           as "M"
go
create default nivelCarrera           as "L"
go
create default nombramiento           as "S"
go
create default opcionSancion          as "A"
go
create default procedencia            as "O"
go
create default sexo                   as "F"
go
create default sistemaEscolarizado    as "ESC"
go
create default tipoActa               as "N"
go
create default tipoAsignatura         as "O"
go
create default tipoExamen             as "O"
go
create default tipoExamenPro          as "O"
go
create default transporte             as "M"
go

print "Reglas"
go
create rule   mayorCero
      as @numero > 0
go
create rule reglaPeriodo
      as @per like "[0-9][0-9][12]"
go
create rule menorCuatro
```

```
        as @var < 4
go

print "Asociar defaults a tipo de dato"
   exec sp_bindefault cero, SHORTINT
   exec sp_bindefault cero, ENTERO
   exec sp_bindefault cero, LONGINT
go

print "Asociar reglas a los campos"
   exec sp_bindrule mayorCero,"_estadistica416.cveDoc"
   exec sp_bindrule menorCuatro,"_estadistica416.cveDoc"
   exec sp_bindrule mayorCero,"_usuarios.maxSesiones"
go

print "Asociar defaults a los campos"
exec sp_bindefault entrega_def, "acta.entregada"
exec sp_bindefault tipoActa, "acta.tipo"
exec sp_bindefault codigoP, "alumno.cp"
exec sp_bindefault estadoCivil, "alumno.edoCivil"
exec sp_bindefault nacionalidad,"alumno.nacionalidad"
exec sp_bindefault sexo, "alumno.sexo"
exec sp_bindefault procedencia, "alumno.tipoProc"
exec sp_bindefault transporte, "alumno.transporte"
exec sp_bindefault ceroReal, "alumno.promedioProc"
exec sp_bindefault entregadoDef, "asegurado.entregado"
exec sp_bindefault movimiento, "asegurado.movimiento"
exec sp_bindefault tipoAsignatura,"asignatura.tipo"
exec sp_bindefault cero, "asignatura.anioPlan"
exec sp_bindefault autorizado, "cambioGpo.autorizacion"
exec sp_bindefault nivelCarrera, "carrera.nivel"
exec sp_bindefault cero, "carrera.anioPlan"
exec sp_bindefault tipoExamen, "curso.tipoExamen"
exec sp_bindefault tipoExamen, "cursoEx.tipoExamen"
exec sp_bindefault inscritoDef, "estudia.inscrito"
exec sp_bindefault ceroReal, "estudia.cuotaLab"
exec sp_bindefault ceroReal, "estudia.cuotaVol"
exec sp_bindefault ceroReal, "estudia.promedio"
```

```
exec sp_bindefault sistemaEscolarizado,"estudia.sistema"
exec sp_bindefault codigoP, "exAlumno.cp"
exec sp_bindefault nacionalidad, "exAlumno.nacionalidad"
exec sp_bindefault sexo, "exAlumno.sexo"
exec sp_bindefault tipoExamenPro, "examenPro.tipo"
exec sp_bindefault cero, "grupoEx.noJurados"
exec sp_bindefault nombramiento, "jurado.nombramiento"
exec sp_bindefault mencion, "presenta.mencion"
exec sp_bindefault codigoP, "primerIngreso.cp"
exec sp_bindefault inscritoDef, "primerIngreso.inscrito"
exec sp_bindefault nacionalidad, "primerIngreso.nacionalidad"
exec sp_bindefault procedencia, "primerIngreso.procedencia"
exec sp_bindefault ceroReal, "primerIngreso.cuotaLab"
exec sp_bindefault ceroReal, "primerIngreso.cuotaVol"
exec sp_bindefault sistemaEscolarizado, "primerIngreso.sistema"
exec sp_bindefault ingresoAnio, "primerIngreso.ingreso"
exec sp_bindefault entidadDef, "primerIngreso.entidad"
exec sp_bindefault opcionSancion, "sancion.tipo"
exec sp_bindefault autorizado, "solicitudMas2.autorizado"
go
```

TRIGGERS

Los triggers son una especie de procedimientos almacenados que están asociados a una tabla. Son programas en automático, es decir, que se ejecutan automáticamente cuando ocurre un evento determinado, generalmente una actualización, borrado o inserción; no se pueden ejecutar manualmente.

Solamente se puede definir un trigger por cada uno de los procesos de insertar, modificar o borrar, para cada tabla, pero puede ser tan completo como deseemos y puede llamar a otros procedimientos almacenados.

Los triggers son utilizados para mantener la integridad referencial permitiendo que si se modifica una llave primaria de una tabla y ésta es llave foranea en otra, se actualice automáticamente.

En caso de que tengamos datos duplicados para aumentar la velocidad de las consulas, se presenta el riesgo de insconsistencia de la información, es aquí cuando los trigger actualizan automáticamente los datos duplicados.

Durante la ejecución de los triggers se crea una tabla temporal llamada inserted o deleted dependiendo de la operación realizada, que sólo puede ser consultada desde el mismo trigger y que contiene los datos que se van a insertar (tabla inserted) o los datos que se van a borrar (tabla deleted) o a actaulizar (tablas deleted e inserted).

A continuación presentamos el script para crear los tiggres de la base de datos del sistema.

```
/* Definición de triggers */

print "Trigger Est40y50"
go
/* Este trigger se ejecuta cuando se inserta algún dato en la tabla de solicitud */
create trigger Est40y50 on solicitud for insert as
  declare @lineas int,@mes smallint,@anio smallint,@Carr char(4),
     @Cve char(2),@doc tinyint
  select @lineas=@@rowcount
  if @lineas=0 return
  if (select count(*) from inserted ) > 1 begin
    rollback transaction
    print "No se permite insertar mas de un registro a la vez"
    return
  end
```

```
  select @Carr = e.cveCarr, @mes=datepart(mm,fechTram),
@anio=datepart(yy,fechTram),
        @doc=cveDocto from inserted i, estudia e where e.noCta=i.noCta
  if @Carr = NULL or @doc > 9
    begin
      rollback transaction
      if @Carr = NULL print "El numero de cuenta no existe"
      if @doc > 9 print "Clave de documento invalido"
      return
  end
  if @mes > 9 and ( @doc < 4 or @doc= 9)
      select @anio=@anio+1
  if @mes > 8 and @doc > 3
      select @anio=@anio+1
  select @Cve=substring(@Carr,1,2)
  select @Carr=convert(char(4),@anio)
  if @doc<4 or @doc=9
      exec prepEst416 @Cve,@Carr,@mes,@doc
  if @doc=4
      exec prepEst58 @Cve,@Carr,@mes
  if @doc=5
      exec prepEst56 @Cve,@Carr,@mes
  if @doc=6
      exec prepEst55 @Cve,@Carr,@mes
  if @doc=7
      exec prepEst57 @Cve,@Carr,@mes
  if @doc=8
      exec prepEst54 @Cve,@Carr,@mes
  return
go

print "Trigger Est510"
go
/* Este trigger se ejecuta cuando se hace una actualización en la tabla de
asegurado */
create trigger Est510 on asegurado for update as
  declare @lineas int, @mes smallint, @anio smallint, @Carr char(4),
    @nombre char(30),@Cve char(2),@aa char(4)
```

```
select @lineas=@@rowcount
if @lineas=0
   return
if @lineas > 1
   begin
      rollback transaction
      print "No se permite actualizar mas de un registro a la vez"
      return
   end
if update(fechaAfiliacion)
  begin
      select @Carr=cveCarr,@mes=datepart(mm,fechaAfiliacion),
          @anio=datepart(yy,fechaAfiliacion) from inserted i, estudia e
      where e.noCta=i.noCta
      select @aa=convert(char(4),@anio),@Cve=substring(@Carr,1,2)
      if @Carr !=NULL
        exec prepEst510 @Cve,@aa,@mes
  end
if update(entregado)
  begin
      select @Carr=cveCarr, @anio=datepart(yy,fechaAfiliacion)
          from inserted i, estudia e
      where e.noCta=i.noCta and entregado="S"
      select @aa=convert(char(4),@anio), @Cve=substring(@Carr,1,2)
       if @Carr != NULL
          exec prepEst510 @Cve,@aa,13
  end
if update(fechaSol)
   begin
       select @Carr=cveCarr, @anio=datepart(yy,fechaSol)
           from inserted i,estudia e
       where e.noCta=i.noCta and movimiento="B"
       select @aa=convert(char(4),@anio), @Cve=substring(@Carr,1,2)
        if @Carr != NULL exec prepEst510 @Cve,@aa,14
   end
return
go
```

```
print "Trigger Est53"
go
/* Este trigger se ejecuta cuando se hace una inserción en la tabla de
bajasAlumnos */
create trigger Est53 on bajasAlumnos for insert as
  declare @lineas int, @mes smallint, @anio smallint, @Carr char(4),
     @nombre char(30),@Cve char(2)
  select @lineas=@@rowcount
  if @lineas=0
    return
  if (select count(*) from inserted ) > 1
    begin
      rollback transaction
      print "No se permite insertar mas de un registro a la vez"
      return
    end
  select @Carr = e.cveCarr, @mes=datepart(mm,fechaTram),
@anio=datepart(yy,fechaTram)
      from inserted i, estudia e
  where  e.noCta=i.noCta and e.cveCarr=i.cveCarr
  if @Carr = NULL
    begin
      rollback transaction
      print "El numero de cuenta-clave_carrera no existe"
      return
    end
  select @Cve=substring(@Carr,1,2)
  select @Carr=convert(char(4),@anio)
  exec prepEst53 @Cve,@Carr,@mes
  return
go
```

PROCEDIMIENTOS ALMACENADOS

Un procedimiento almacenado es un conjunto de cláusulas SQL y se encuentra en la base de datos.

Un procedimiento almacenado es compilado sólo la primera vez, por lo que las siguientes veces su ejecución será más rápida.

Puede recibir parámetros y regresar valores o estatus, ser llamado desde otro procedimiento almacenado, ser ejecutado en un servidor remoto y es ejecutado por su nombre.

El uso de procedimientos almacenados disminuye el tráfico de la red ya que únicamente viaja el nombre del procedimiento y los parámetros que recibió.

Debido a que son almacenados en la base de datos, éstos pueden ser ejecutados por todos los usuarios que tengan permiso para ejecutarlos.

A continuación presentamos el escript para crear los procedimientos almacenados de la base de datos del sistema.

```
/* Definicion de procedimientos */

print "Procedimiento ActCoutaVol"
go
/* Este procedimiento actualiza la cuota voluntaria en la tabla de primerIngreso
*/
create proc ActCuotaVol (@vNoCta char(8), @vCuotaVol real) as
  update primerIngreso set cuotaVol= @vCuotaVol
    where primerIngreso.noCta= @vNoCta
return
go
```

```
print "Procedimiento ActualizaInscrito"
go
/* Este procedimiento actualiza el campo inscrito con una "S" en la tabla de
primerIngreso */
create proc ActualizaInscrito (@vNoCta char(8)) as
 update primerIngreso set inscrito = "S"
  where primerIngreso.noCta=@vNoCta
return
go

print "Procedimiento Calendario "
go
 /* Crea o actualiza los datos de un grupo de extraordinario EA o EB en
    las tablas grupoEx y sinodal */
CREATE PROCEDURE Calendario (@Carrera char(4), @Materia char(4),
 @Fecha fecha, @Grupo char(4), @Periodo char(3), @Salon char(6),
 @Cve1 char(13)=NULL, @Cve2 char(13)=NULL, @Cve3 char(13)=NULL,
 @Cve4 char(13)=NULL) AS
 IF EXISTS ( SELECT gpo FROM grupoEx WHERE cveCarr=@Carrera
  AND cveAsig=@Materia AND gpo=@Grupo )
  UPDATE grupoEx SET fecha=@Fecha,salon=@Salon,periodo=@Periodo
 WHERE
          cveCarr=@Carrera AND cveAsig=@Materia AND gpo=@Grupo
  ELSE
   INSERT INTO grupoEx (cveCarr,cveAsig,fecha,gpo,periodo,salon)
 VALUES
   (@Carrera,@Materia,@Fecha,@Grupo,@Periodo,@Salon)
 DELETE sinodal WHERE cveCarr=@Carrera AND cveAsig=@Materia AND
gpo=@Grupo
 IF (@Cve1!=NULL)
  INSERT INTO sinodal (cveCarr,cveAsig,gpo,rfc)
  values (@Carrera,@Materia,@Grupo,@Cve1)
 IF (@Cve2!=NULL)
  INSERT INTO sinodal (cveCarr,cveAsig,gpo,rfc)
  values (@Carrera,@Materia,@Grupo,@Cve2)
 IF (@Cve3!=NULL)
  INSERT INTO sinodal (cveCarr,cveAsig,gpo,rfc)
  VALUES (@Carrera,@Materia,@Grupo,@Cve3)
```

```
IF (@Cve4!=NULL)
      INSERT INTO sinodal (cveCarr,cveAsig,gpo,rfc)
      VALUES (@Carrera,@Materia,@Grupo,@Cve4)
RETURN
go

print "Procedimiento DespInscPrimIngr "
go
/* Este procedimiento regresa las asignaturas y el grupo al que el alumno de
primer ingreso
   deberá inscribirse */
create proc DespInscPrimIngr (@vNoCta char(8)) as
  select c.cveAsig, c.grupo, a.nombre, a.semestre, c.cveCarr
    from clase c, asignatura a, primerIngreso p
    where (p.noCta=@vNoCta) and p.gpo=c.grupo and p.cveCarr=c.cveCarr
    and c.cveAsig=a.cveAsig and c.cveCarr=a.cveCarr and c.cveCarr=p.cveCarr
return
go


print "Procedimiento DespPrimIngr "
go
/* Este procedimiento regresa los datos generales de un alumno de primer
ingreso */
create proc DespPrimIngr (@vNoCta char(8)) as
  select nombre,comprobante,cveCarr, anioplan,ingreso, gpo
  from primerIngreso where noCta=@vNoCta
return
go

print "Procedimiento InscribeExt "
go
/* Este procedimiento registra de 1 a 2 asignaturas para examen extraordinario
*/
CREATE PROC InscribeExt(@Carr char(4), @Per char(3), @Cuenta char(8) ,
  @Mat1 char(4)=NULL, @Gpo1 char(4)=NULL, @Mat2 char(4)=NULL,
  @Gpo2 char(4)=NULL) AS
  DECLARE @Cuantas tinyint
```

```
  SELECT @Cuantas = 0
  IF @Mat1 != NULL
    BEGIN
      INSERT INTO inscrito (cveAsig,cveCarr,gpo,periodo,noCta)
      VALUES (@Mat1,@Carr,@Gpo1,@Per,@Cuenta)
      SELECT @Cuantas = @Cuantas + 1
    END
  IF @Mat2 != NULL
    BEGIN
      INSERT INTO inscrito (cveAsig,cveCarr,gpo,periodo,noCta)
      VALUES (@Mat2,@Carr,@Gpo2,@Per,@Cuenta)
      SELECT @Cuantas = @Cuantas + 2
    END
  SELECT valor=@Cuantas
RETURN
go

print "Procedimiento RevisaInscrito "
go
/* Este procedimiento regresa el campo de inscrito de  un alumno de primer
ingreso */
create proc RevisaInscrito (@vNoCta char(8)) as
  select inscrito from primerIngreso
    where @vNoCta = primerIngreso.noCta
return
go

print "Procedimiento asigCursadas "
go
/* Este procedimiento regresa la lista de las asignaturas que un alumno ya ha
cursado */
CREATE PROC asigCursadas (@carrera char(4), @numCta char(8)) as
select c.cveAsig, c.gpo, a.semestre, a.nombre from asignatura a, cursa c
  where c.cveCarr = @carrera and c.noCta = @numCta
  and c.cveCarr = a.cveCarr and  c.cveAsig = a.cveAsig
return
go
```

```
print "Procedimiento asignaturasAlumno "
go
/* Este procedimiento regresa las asignaturas, tipo de examen y calificación de
un alumno
   que tramita su reinscripción */
create proc asignaturasAlumno  (@cveCarr char (4), @noCta char (8)) as
  select cveAsig, tipoExamen, calificacion from curso where cveCarr =
  @cveCarr and noCta = @noCta order by cveAsig
return
go

print "Procedimiento borraCalen"
go
  /*  Este procedimiento elimina los datos de un grupo de extraordinario EA o
EB en las tablas grupoEx y sinodal */
CREATE PROCEDURE borraCalen (@Carrera char(4), @Materia char(4),
  @Grupo char(4)) AS
  IF EXISTS ( SELECT * FROM grupoEx WHERE cveCarr=@Carrera
  AND cveAsig=@Materia AND gpo=@Grupo )
    BEGIN
      DELETE grupoEx WHERE cveCarr=@Carrera AND
        cveAsig=@Materia AND gpo=@Grupo
      DELETE sinodal WHERE cveCarr=@Carrera AND
        cveAsig=@Materia AND gpo=@Grupo
      SELECT valor = 0
      RETURN
    END
  SELECT valor = -1
RETURN
go

print "Procedimiento catCarrera"
go
/* Este procedimiento despliega el catálogo de carreras */
CREATE PROC catCarrera AS
  select nombre, cveCarr from carrera order by nombre
return
go
```

```
print "Procedimiento catPlantel"
go
/* Este procedimiento despliega el catálogo de planteles */
CREATE PROC catPlantel AS
  select * from plantel order by cvePlantel
return
go

print "Procedimiento credTodas"
go
/* Este procedimiento verifica los creditos de al menos 1 y hasta 7
   materias con los datos de entrada:
   Clave de Carrera, anio plan, semestre, Materia1 [Materia2 ...]
   y devuelve la suma de los creditos de las asignaturas */
CREATE PROC credTodas (@Carrera char(4), @anio smallint, @sem int,
  @Cve1 char(4), @Cve2 char(4)=NULL, @Cve3 char(4)=NULL, @Cve4
char(4)=NULL,
  @Cve5 char(4)=NULL, @Cve6 char(4)=NULL, @Cve7 char(4)=NULL) AS
  declare @totCred int, @credSemestre int
  SELECT @totCred=sum(creditos) FROM asignatura WHERE cveCarr =
@Carrera
    and cveAsig IN ( @Cve1, @Cve2, @Cve3, @Cve4, @Cve5, @Cve6,
@Cve7 )
  SELECT @credSemestre = totCreditos from creditos where cveCarr =
@Carrera
    and anioPlan=@anio and semestre = @sem
  if @totCred > @credSemestre
    select creditosValidos = 0
  else
    select creditosValidos = 1
return
go

print "Procedimiento cupoGrupo"
go
/*Obtiene el cupo de una asignatura en un grupo.
        Entradas:       grupo           clave de grupo
```

```
                        cveCarr         clave de carrera
                        cveAsig         clave de asignatura
        Valor de retorno: cupo del grupo    */
create procedure cupoGrupo ( @grupo char(4),
  @cveCarr char(4),
  @cveAsig char(4) ) as
  declare @cupo smallint
  if not exists ( SELECT * FROM clase WHERE grupo=@grupo AND
cveCarr=@cveCarr
  AND cveAsig=@cveAsig )
    begin
      raiserror 20001 "Grupo inexistente"
      return -1
    end
  SELECT @cupo=cupo FROM clase WHERE grupo=@grupo AND
cveCarr=@cveCarr AND
    cveAsig=@cveAsig
return @cupo
go

print "Procedimiento ingresoAñosPost"
go
/* Este porcedimiento inserta en la tabla de curso los datos (acta de equivalencia
académica
   de un alumno que solicita  ingreso a años posteriores */
CREATE PROC ingresoAñosPost (@calif char(2), @cveAsig char(4),
  @carrera char(4), @folio int, @numCta char (8), @periodo char(3)) as
  insert into curso values
(@calif,@cveAsig,@carrera,@folio,@numCta,@periodo,'O')
return
go

print "Procedimiento listAsig"
go
/* Este procedimiento depliega un catálogo de asignaturas por carrera y plan de
estudios /*
create procedure listAsig (@cveCarrera char(4),@pl smallint) as
  select cveAsig, nombre from asignatura
```

```
    where cveCarr = @cveCarrera and anioPlan = @pl
    order by cveAsig
return
go

print "Procedimiento matAprobada"
go
/* Este procedimiento verifica con que calificacion y cuando aprobo una materia
   un alumno con los datos de     entrada:
   Clave de Carrera, Numero de Cuenta, Materia1
   y devuelve la calificacion aprobatoria, el periodo y tipo de examen */
CREATE PROC matAprobada(@Carrera char(4), @Cuenta char(8), @Cve1
char(4) ) AS
  SELECT calificacion,periodo,tipoExamen FROM curso WHERE calificacion
    IN ("MB","B ","S  ","AC","RE","CO","10","9 ","8 ","7 ","6 ")
    AND cveAsig=@Cve1 AND cveCarr = @Carrera AND noCta = @Cuenta
RETURN
go

print "Procedimiento obtenPeriodoActual"
go
/* Este procedimiento obtiene el período actual */
CREATE PROC obtenPeriodoActual as
  select periodo from periodo where fechaIni = (select max(fechaIni) from
periodo
    where substring(periodo,3,1) like "[12]")
return
go

print "Procedimiento prepEst11"
go
/* Este procedimiento prepara la estadística de Asignación Primer Ingreso */
create proc prepEst11 as
  select * from _estadistica11
  declare @Per char(3),@Gen char(4),@Var int
  select @Per=periodo from periodo where fechaIni = (select max(fechaIni)
    from periodo where     substring(periodo,3,1)="1")
  if @Per = NULL
```

```
    begin
      print "El periodo actual no es semestre impar"
      return
    end
  if not exists(select * from primerIngreso) return
  if convert(int,substring(@Per,1,2)) < 50
    select @Gen="20"+substring(@Per,1,2)
  else
    select @Gen="19"+substring(@Per,1,2)
  if exists(select * from _estadistica11 where generacion=@Gen)
    update _estadistica11
      set biologo=(select count(noCta) from primerIngreso where
      substring(cveCarr,1,2)="03"),enfermeria=(select count(noCta) from
      primerIngreso where substring(cveCarr,1,2)="09"), medico=(select
      count(noCta) from primerIngreso where substring(cveCarr,1,2)="12"),
      dentista=(select count(noCta) from primerIngreso where
      substring(cveCarr,1,2)="14"),psicologia=(select count(noCta) from
      primerIngreso where substring(cveCarr,1,2)="19"),optometria=(select
      count(noCta) from primerIngreso where substring(cveCarr,1,2)="42")
      where generacion=@Gen
  else
    insert into _estadistica11
    select generacion=@Gen,biologo=(select count(noCta) from primerIngreso
      where substring(cveCarr,1,2)="03"),enfermeria=(select count(noCta) from
      primerIngreso where substring(cveCarr,1,2)="09"), medico=(select
      count(noCta) from primerIngreso where substring(cveCarr,1,2)="12"),
      dentista=(select count(noCta) from primerIngreso where
      substring(cveCarr,1,2)="14"),psicologia=(select count(noCta) from
      primerIngreso where substring(cveCarr,1,2)="19"),optometria=(select
     count(noCta) from primerIngreso where substring(cveCarr,1,2)="42")
return
go

print "Procedimiento prepEst12"
go
/* Este procedimiento prepara la estadística de Cambios de Grupo y Bajas
Artículo 23
   Primer Ingreso */
```

```
create proc prepEst12 as
  declare @Cve char(2), @Per char(3), @Cual tinyint
  select @Per=periodo from periodo where fechaIni = (select max(fechaIni)
    from periodo where substring(periodo,3,1) ="1" )
  if @Per = NULL
    begin
      print "El periodo actual no es semestre impar"
      return
    end
  if not exists (select * from primerIngreso) return
    select @Cual = 1
    while @Cual < 7
      begin
        if @Cual = 1 select @Cve="03"
        if @Cual = 2 select @Cve="09"
        if @Cual = 3 select @Cve="12"
        if @Cual = 4 select @Cve="14"
        if @Cual = 5 select @Cve="19"
        if @Cual = 6 select @Cve="42"
        if exists(select * from _estadistica12 where clave+periodo=@Cve+@Per)
          update _estadistica12 set bajas = ( select count(noCta) from
            primerIngreso where substring(cveCarr,1,2) = @Cve and
            inscrito="N"), cambios = ( select count(noCta) from
            cambioGpo where substring(cveCarr,1,2) = @Cve and
            autorizacion="S") where clave+periodo = @Cve+@Per
        else
          insert into _estadistica12
          select clave = @Cve , periodo = @Per, bajas = (select count(noCta)
            from primerIngreso where substring(cveCarr,1,2) = @Cve and
            inscrito="N"), cambios = (select count(noCta)
            from cambioGpo where substring(cveCarr,1,2) = @Cve and
            autorizacion="S")
          select @Cual=@Cual+1
      end
return
go
```

```
print "Procedimiento prepEst13"
go
/* Este procedimiento prepara la estadística Cuadro Comparativo de Primer
Ingreso */
create proc prepEst13 as
  declare @Per char(3),@Gen char(4),@Var int
  select @Per=periodo from periodo where fechaIni = (select max(fechaIni)
    from periodo where   substring(periodo,3,1)="1")
  if @Per = NULL
    begin
      print "El periodo actual no es semestre impar"
      return
    end
  if not exists(select * from primerIngreso) return
  if convert(int,substring(@Per,1,2)) < 50
    select @Gen="20"+substring(@Per,1,2)
  else
    select @Gen="19"+substring(@Per,1,2)
  if exists(select * from _estadistica13 where generacion=@Gen)
    update _estadistica13 set biolAlumAsig=(select count(noCta) from
      primerIngreso where substring(cveCarr,1,2)="03"),enfeAlumAsig=(select
      count(noCta) from primerIngreso where substring(cveCarr,1,2)="09"),
      mediAlumAsig=(select count(noCta) from primerIngreso where
      substring(cveCarr,1,2)="12"),dentAlumAsig=(select count(noCta) from
      primerIngreso where substring(cveCarr,1,2)="14"),psicAlumAsig=(select
      count(noCta) from primerIngreso where substring(cveCarr,1,2)="19"),
      optoAlumAsig=(select count(noCta) from primerIngreso where
      substring(cveCarr,1,2)="42"),biolAlumInsc=(select count(noCta) from
      primerIngreso where substring(cveCarr,1,2)="03" and inscrito="S"),
      enfeAlumInsc=(select count(noCta) from primerIngreso where
      substring(cveCarr,1,2)="09" and inscrito="S"),mediAlumInsc=(select
      count(noCta) from primerIngreso where substring(cveCarr,1,2)="12" and
      inscrito="S"),dentAlumInsc=(select count(noCta) from primerIngreso
      where substring(cveCarr,1,2) = "14" and inscrito="S"), psicAlumInsc=
      (select count(noCta) from primerIngreso where substring(cveCarr,1,2)=
      "19" and inscrito="S"),optoAlumInsc=(select count(noCta) from
      primerIngreso where substring(cveCarr,1,2)="42" and inscrito="S")
      where generacion=@Gen
```

```
 else
   insert into _estadistica13 select generacion=@Gen,biolDeclara=0,
     enfeDeclara=0,mediDeclara=0,
dentDeclara=0,psicDeclara=0,optoDeclara=0,
     biolAlumAsig=(select count(noCta) from primerIngreso where
     substring(cveCarr,1,2)="03"),enfeAlumAsig=(select count(noCta) from
     primerIngreso where substring(cveCarr,1,2)="09"),mediAlumAsig=(select
     count(noCta) from primerIngreso where substring(cveCarr,1,2)="12"),
     dentAlumAsig=(select count(noCta) from primerIngreso where
     substring(cveCarr,1,2)="14"),psicAlumAsig=(select count(noCta) from
     primerIngreso where substring(cveCarr,1,2)="19"),optoAlumAsig=(select
     count(noCta) from primerIngreso where substring(cveCarr,1,2)="42"),
     biolAlumInsc=(select count(noCta) from primerIngreso where
     substring(cveCarr,1,2)="03" and inscrito="S"),enfeAlumInsc=(select
     count(noCta) from primerIngreso where substring(cveCarr,1,2)="09" and
     inscrito="S"),mediAlumInsc=(select count(noCta) from primerIngreso
     where substring(cveCarr,1,2)="12" and inscrito="S"),
     dentAlumInsc=(select count(noCta) from primerIngreso where
     substring(cveCarr,1,2) = "14" and inscrito="S"),psicAlumInsc=(select
     count(noCta) from primerIngreso where substring(cveCarr,1,2) = "19"
     and inscrito="S"), optoAlumInsc=(select count(noCta) from primerIngreso
     where substring(cveCarr,1,2)="42" and inscrito="S")
return
go

print "Procedimiento prepEst18"
go
/* Este procedimiento prepara la estadística Resumen de Inscripción Anual */
create proc prepEst18 as
  declare @Per char(3),@Gen char(4),@Var int
  select @Per=periodo from periodo where fechaIni = (select max(fechaIni)
    from periodo where substring(periodo,3,1) like "[12]")
  if exists(select * from _estadistica18 where periodo=@Per)
    update _estadistica18 set
      biologo=(select count(noCta) from primerIngreso where
      substring(cveCarr,1,2) = "03" and inscrito="S")+
      (select count(noCta) from estudia where
      substring(cveCarr,1,2) = "03" and inscrito="S"),
```

```
    enfermeria=(select count(noCta) from primerIngreso where
    substring(cveCarr,1,2) = "09" and inscrito="S")+
    (select count(noCta) from estudia where
    substring(cveCarr,1,2) = "09" and inscrito="S"),
    medico=(select count(noCta) from primerIngreso where
    substring(cveCarr,1,2) = "12" and inscrito="S")+
    (select count(noCta) from estudia where
    substring(cveCarr,1,2) = "12" and inscrito="S"),
    dentista=(select count(noCta) from primerIngreso where
    substring(cveCarr,1,2) = "14" and inscrito="S")+
    (select count(noCta) from estudia where
    substring(cveCarr,1,2) = "14" and inscrito="S"),
    psicologia=(select count(noCta) from primerIngreso where
    substring(cveCarr,1,2) = "19" and inscrito="S")+
    (select count(noCta) from estudia where
    substring(cveCarr,1,2) = "19" and inscrito="S"),
    optometria=(select count(noCta) from primerIngreso where
    substring(cveCarr,1,2) = "42" and inscrito="S")+
    (select count(noCta) from estudia where
    substring(cveCarr,1,2) = "42" and inscrito="S")
    where periodo=@Per
else
  insert into _estadistica18
  select periodo=@Per,
    biologo=(select count(noCta) from primerIngreso where
    substring(cveCarr,1,2) = "03" and inscrito="S")+
    (select count(noCta) from estudia where
    substring(cveCarr,1,2) = "03" and inscrito="S"),
    enfermeria=(select count(noCta) from primerIngreso where
    substring(cveCarr,1,2) = "09" and inscrito="S")+
    (select count(noCta) from estudia where
    substring(cveCarr,1,2) = "09" and inscrito="S"),
    medico=(select count(noCta) from primerIngreso where
    substring(cveCarr,1,2) = "12" and inscrito="S")+
    (select count(noCta) from estudia where
    substring(cveCarr,1,2) = "12" and inscrito="S"),
    dentista=(select count(noCta) from primerIngreso where
    substring(cveCarr,1,2) = "14" and inscrito="S")+
```

```
    (select count(noCta) from estudia where
    substring(cveCarr,1,2) = "14" and inscrito="S"),
    psicologia=(select count(noCta) from primerIngreso where
    substring(cveCarr,1,2) = "19" and inscrito="S")+
    (select count(noCta) from estudia where
    substring(cveCarr,1,2) = "19" and inscrito="S"),
    optometria=(select count(noCta) from primerIngreso where
    substring(cveCarr,1,2) = "42" and inscrito="S")+
    (select count(noCta) from estudia where
    substring(cveCarr,1,2) = "42" and inscrito="S")
return
go

print "Procedimiento prepEst21"
go
/* Este procedimiento prepara la estadística Registro de Inscripción EA(índice
de aprobación
   y reprobación */
create proc prepEst21 as
 declare @Per char(3)
 select @Per=periodo from periodo where fechaIni = (select max(fechaIni)
  from periodo where substring(periodo,3,1) like "[12]" )
 if not exists (select * from inscrito where periodo=@Per and
  substring(gpo,1,2)="EA") return
 if exists(select * from _estadistica21 where periodo=@Per)
  delete _estadistica21 where periodo=@Per and
  substring(gpo,1,2)="EA"
 insert into _estadistica21
 select * from inscrito where periodo=@Per and
  substring(gpo,1,2)="EA" order by cveCarr,cveAsig,gpo
return
go

print "Procedimiento prepEst22"
go
/* Este procedimiento prepara la estadística Registro de Inscripción EB (índice
de aprobación
   y reprobación */
```

```
create proc prepEst22 as
 declare @Cve char(2), @Per char(3), @Cual tinyint,@nombre varchar(30)
 select @Per=periodo from periodo where fechaIni = (select max(fechaIni)
  from periodo where substring(periodo,3,1) like "[12]" )
 if not exists (select * from inscrito where periodo=@Per) return
 select @Cual = 2
  while @Cual < 7
   begin
    if @Cual = 1 select @Cve="03",@nombre = "BIOLOGO"
    if @Cual = 2 select @Cve="09",@nombre = "ENFERMERIA NIVEL
TECNICO"
    if @Cual = 3 select @Cve="12",@nombre = "MEDICO CIRUJANO"
    if @Cual = 4 select @Cve="14",@nombre = "CIRUJANO DENTISTA"
    if @Cual = 5 select @Cve="19",@nombre = "LIC. EN PSICOLOGIA"
    if @Cual = 6 select @Cve="42",@nombre = "LIC. EN OPTOMETRIA"
    if exists(select * from _estadistica22 where clave+periodo=@Cve+@Per)
     update _estadistica22 set inscritos =
      (select count(*) from inscrito where periodo=@Per and
      substring(cveCarr,1,2)=@Cve and gpo like "EA__" ), aprobados =
      (select count(*) from curso where periodo=@Per and
      substring(cveCarr,1,2)=@Cve and gpo like "EA__" and calificacion
      in ("MB","B","S","10","9","8","7","6","AC","RE","CO")),
      reprobados= (select count(*) from curso where periodo=@Per and
      substring(cveCarr,1,2)=@Cve and gpo like "EA__" and calificacion
      not in ("MB","B","S","10","9","8","7","6","AC","RE","CO"))
      where clave+periodo = @Cve+@Per
    else
     insert into _estadistica22
     select clave=@Cve,carrera=@nombre,periodo=@Per, inscritos =
      (select count(*) from inscrito where periodo=@Per and
      substring(cveCarr,1,2)=@Cve and gpo like "EA__" ), aprobados =
      (select count(*) from curso where periodo=@Per and
      substring(cveCarr,1,2)=@Cve and gpo like "EA__" and calificacion
      in ("MB","B","S","10","9","8","7","6","AC","RE","CO")),
      reprobados= (select count(*) from curso where periodo=@Per and
      substring(cveCarr,1,2)=@Cve and gpo like "EA__" and calificacion
      not in ("MB","B","S","10","9","8","7","6","AC","RE","CO"))
      select @Cual=@Cual+1
```

```
   end
return
go

print "Procedimiento prepEst23"
go
/* Este procedimiento prepara la estadística Exámenes Extraordinarios
(inscripción) */
create proc prepEst23 as
 declare @Cve char(2), @Per char(3), @Cual tinyint,@nombre varchar(30)
 select @Per=periodo from periodo where fechaIni = (select max(fechaIni)
  from periodo where substring(periodo,3,1) like "[12]" )
 if not exists (select * from inscrito where periodo=@Per) return
 select @Cual = 1
 while @Cual < 7
  begin
   if @Cual = 1 select @Cve="03",@nombre = "BIOLOGO"
   if @Cual = 2 select @Cve="09",@nombre = "ENFERMERIA NIVEL
TECNICO"
   if @Cual = 3 select @Cve="12",@nombre = "MEDICO CIRUJANO"
   if @Cual = 4 select @Cve="14",@nombre = "CIRUJANO DENTISTA"
   if @Cual = 5 select @Cve="19",@nombre = "LIC. EN PSICOLOGIA"
   if @Cual = 6 select @Cve="42",@nombre = "LIC. EN OPTOMETRIA"
   if exists(select * from _estadistica23 where clave+periodo=@Cve+@Per)
    update _estadistica23
     set inscritosEA = (select count(cveAsig) from inscrito where
     substring(cveCarr,1,2) = @Cve and substring(gpo,1,2)="EA" and
     periodo=@Per),
     inscritosEB = (select count(cveAsig) from inscrito where
     substring(cveCarr,1,2) = @Cve and substring(gpo,1,2)="EB" and
     periodo=@Per)
     where clave+periodo = @Cve+@Per
   else
    insert into _estadistica23
    select clave=@Cve,carrera=@nombre,periodo=@Per,inscritosEA =
     (select count(cveAsig) from inscrito where substring(cveCarr,1,2)
     = @Cve and substring(gpo,1,2)="EA" and periodo=@Per),
    inscritosEB=(select count(cveAsig) from inscrito where
```

```
      substring(cveCarr,1,2)=@Cve and substring(gpo,1,2)="EB" and
      periodo=@Per)
     select @Cual=@Cual+1
end
return
go

print "Procedimiento prepEst24"
go
/* Este procedimiento prepara la estadística Exámenes Extraordinarios (insc. por
asignatura) */
create proc prepEst24 as
 declare @Cve char(2), @Per char(3), @Cual tinyint,@nombre varchar(30)
 select @Per=periodo from periodo where fechaIni = (select max(fechaIni)
  from periodo where substring(periodo,3,1) like "[12]" )
 if not exists (select * from inscrito where periodo=@Per) return
 select @Cual = 2
 while @Cual < 7
  begin
   if @Cual = 1 select @Cve="03",@nombre = "BIOLOGO"
   if @Cual = 2 select @Cve="09",@nombre = "ENFERMERIA NIVEL
TECNICO"
   if @Cual = 3 select @Cve="12",@nombre = "MEDICO CIRUJANO"
   if @Cual = 4 select @Cve="14",@nombre = "CIRUJANO DENTISTA"
   if @Cual = 5 select @Cve="19",@nombre = "LIC. EN PSICOLOGIA"
   if @Cual = 6 select @Cve="42",@nombre = "LIC. EN OPTOMETRIA"
   if exists(select * from _estadistica24 where clave+periodo=@Cve+@Per)
    update _estadistica24 set inscritos =
      (select count(*) from inscrito where periodo=@Per and
      substring(cveCarr,1,2)=@Cve and gpo like "EB__" ), aprobados =
      (select count(*) from curso where periodo=@Per and
      substring(cveCarr,1,2)=@Cve and gpo like "EB__" and calificacion
      in ("MB","B","S","10","9","8","7","6","AC","RE","CO")),
      reprobados= (select count(*) from curso where periodo=@Per and
      substring(cveCarr,1,2)=@Cve and gpo like "EB__" and calificacion
      not in ("MB","B","S","10","9","8","7","6","AC","RE","CO"))
      where clave+periodo = @Cve+@Per
   else
```

```
      insert into _estadistica24
      select clave=@Cve,carrera=@nombre,periodo=@Per, inscritos =
       (select count(*) from inscrito where periodo=@Per and
       substring(cveCarr,1,2)=@Cve and gpo like "EB__" ), aprobados =
       (select count(*) from curso where periodo=@Per and
       substring(cveCarr,1,2)=@Cve and gpo like "EB__" and calificacion
       in ("MB","B","S","10","9","8","7","6","AC","RE","CO")),
       reprobados= (select count(*) from curso where periodo=@Per and
       substring(cveCarr,1,2)=@Cve and gpo like "EB__" and calificacion
       not in ("MB","B","S","10","9","8","7","6","AC","RE","CO"))
      select @Cual=@Cual+1
end
return
go

print "Procedimiento prepEst25"
go
/* Este procedimiento prepara la estadística Más de Dos Exámenes
Extraordinarios */
create proc prepEst25 as
 declare @Cve char(2), @Per char(3), @Cual tinyint,@nombre varchar(30)
 select @Per=periodo from periodo where fechaIni = (select max(fechaIni)
  from periodo where substring(periodo,3,1) like "[12]" )
 select @Cual = 1
 while @Cual < 7
  begin
   if @Cual = 1 select @Cve="03",@nombre = "BIOLOGO"
   if @Cual = 2 select @Cve="09",@nombre = "ENFERMERIA NIVEL
TECNICO"
   if @Cual = 3 select @Cve="12",@nombre = "MEDICO CIRUJANO"
   if @Cual = 4 select @Cve="14",@nombre = "CIRUJANO DENTISTA"
   if @Cual = 5 select @Cve="19",@nombre = "LIC. EN PSICOLOGIA"
   if @Cual = 6 select @Cve="42",@nombre = "LIC. EN OPTOMETRIA"
   if exists(select * from _estadistica25 where clave+periodo=@Cve+@Per)
    update _estadistica25
     set inscritos =
     (select count(*) from solicitudMas2 where periodo=@Per and
     substring(cveCarr,1,2)=@Cve ), autorizados =
```

```
      (select count(*) from solicitudMas2 where periodo=@Per and
      substring(cveCarr,1,2)=@Cve and autorizado="S")
      where clave+periodo = @Cve+@Per
    else
      insert into _estadistica25
      select clave=@Cve,carrera=@nombre,periodo=@Per, inscritos =
       (select count(*) from solicitudMas2 where periodo=@Per and
       substring(cveCarr,1,2)=@Cve ), autorizados =
       (select count(*) from solicitudMas2 where periodo=@Per and
       substring(cveCarr,1,2)=@Cve and autorizado="S")
      select @Cual=@Cual+1
end
return
go

print "Procedimiento prepEst31"
go
/* Este procedimiento prepara la estadística Cambio de Carrera */
create proc prepEst31 as
 declare @Per char(3),@Gen char(4),@Cual tinyint,@Cve char(2)
 select @Per=periodo from periodo where fechaIni = (select max(fechaIni)
  from periodo where substring(periodo,3,1)="1")
 if @Per = NULL
  begin
   print "El periodo actual no es semestre impar"
   return
  end
select @Cual = 1
while @Cual < 6
 begin
  if @Cual = 1 select @Cve="03"
  if @Cual = 2 select @Cve="12"
  if @Cual = 3 select @Cve="14"
  if @Cual = 4 select @Cve="19"
  if @Cual = 5 select @Cve="42"
   if exists(select * from _estadistica31 where clave = @Cve and
    periodo = @Per)
    update _estadistica31 set
```

```
      ingreso=(select count(*) from primerIngreso p, alumno a where
      substring(cveCarr,1,2)=@Cve and p.noCta=a.noCta),
      egreso=(select count(*) from cambioCarrera where
      substring(cveCarr,1,2)=@Cve)
      where clave = @Cve and periodo = @Per
    else
      insert into _estadistica31
      select clave=@Cve,periodo=@Per,
       ingreso=(select count(*) from primerIngreso p, alumno a where
       substring(cveCarr,1,2)=@Cve and p.noCta=a.noCta),
       egreso=(select count(*) from cambioCarrera where
       substring(cveCarr,1,2)=@Cve)
      select @Cual=@Cual+1
end
return
go

print "Procedimiento prepAluBajas"
go
/* Este procedimiento extrae de alumno,estudia y curso los alumnos con
baja definitiva voluntaria hacia las tablas exAlumno y cursoEx */
create proc prepAluBajas as
 declare @Per char(3)
 select @Per=periodo from periodo where fechaIni=(select max(fechaIni)
  from periodo where substring(periodo,3,1) like "[12]")
 insert into exAlumno
 select colonia,cp,e.cveCarr,delegacion,domicilio,entidad,ingreso,
  nacionalidad,e.noCta,e.nombre,regAlumno,regExalumno="08",sexo,
  tel,ultimoMov=@Per,promedio,e.anioPlan
  from alumno a, estudia e, bajasAlumnos b
  where a.noCta=*e.noCta and b.noCta=e.noCta and b.cveCarr=e.cveCarr
 insert into cursoEx
 select calificacion,cveAsig,r.cveCarr,folioActa,gpo,
  r.noCta,periodo,tipoExamen,folioActaRec
  from curso r, estudia e, bajasAlumnos b
  where r.noCta=e.noCta and r.cveCarr=e.cveCarr and e.cveCarr=b.cveCarr
  and b.noCta=e.noCta
 delete alumno where noCta in (select noCta from bajasAlumnos)
```

```
 delete estudia where noCta in (select noCta from bajasAlumnos)
 delete curso where noCta in (select noCta from bajasAlumnos)
return
go

print "Procedimiento prepCamAnual"
go
/* Este procedimiento limpia y actualiza las tablas para iniciar un nuevo año */
create proc prepCamAnual(@anio smallint) as
 print "Estadistica 43"
 exec prepEst43 @anio
 print "Estadistica 44"
 exec prepEst44 @anio
 print "Fin de proceso"
return
go

print "Procedimiento prepCamPeriodo"
go
/*Este procedimiento actualiza las tablas de estadísticas */
create proc prepCamPeriodo as
 declare @Per char(3),@Gen char(4)
 select @Per=periodo from periodo where fechaIni = (select max(fechaIni) from
periodo
 where substring(periodo,3,1) like "[12]")
 if @Per=NULL
  begin
   print "No existe ningun periodo !!!"
   return
  end
 if convert(int,substring(@Per,1,2)) < 50
  select @Gen="20"+substring(@Per,1,2)
 else
  select @Gen="19"+substring(@Per,1,2)
 print "Este procedimiento tardara en ejecutarse alrededor de 5 horas"
 if substring(@Per,3,1) = "1"
  begin
   print "Es semestre impar"
```

```
    print "Estadistica 11"
    exec prepEst11
    print "Estadistica 12"
    exec prepEst12
    print "Estadistica 13"
    exec prepEst13
    print "Estadistica 31"
    exec prepEst31
    print "Estadistica 33"
    exec prepEst33
    print "Estadistica 34"
    exec prepEst34
    print "Estadistica 35"
    exec prepEst35
    print "Estadistica 36"
    exec prepEst36
    print "Estadistica 37"
    exec prepEst37
  end
else
  begin
    print "Semestre par"
    print "Estatidistica 32"
    exec prepEst32
  end
print "Todos los periodos:"
print "Estadistica 14"
exec prepEst14
print "Estadistica 15"
exec prepEst15
print "Estadistica 17"
exec prepEst17
print "Estadistica 18"
exec prepEst18
print "Estadistica 21"
exec prepEst21
print "Estadistica 22"
exec prepEst22
```

```
 print "Estadistica 23"
 exec prepEst23
 print "Estadistica 24"
 exec prepEst24
 print "Estadistica 25"
 exec prepEst25
 print "Estadistica 51"
 exec prepEst51
 print "Estadistica 510"
 exec prepEst510 "*",@Gen,15
 if substring(@Per,3,1) = "1"
  begin
   print "Cambiando alumnos de 1er ingreso a estudia"
   exec prepAluPrimer
  end
 print "Actualiza tabla estudia con tabla curso, tiempo aprox. 5 hrs"
 exec actEst
 print "Actualizando exAlumno con alumnos egresados"
 exec prepAlu100
 print "Estadistica 590"
 exec prepEst590
 print "Estadistica42"
 exec prepEst42
 print "Estadistica 52"
 exec prepEst52
 print "Estadistica 59"
 exec prepEst59
 print "Bajas definitivas voluntarias"
 exec prepAluBajas
 print "Bajas por expulsion definitiva"
 exec prepAluSancion
 print "Fin de proceso."
return
go
```

```
print "Procedimiento prepRep"
go
/* Este procedimiento se encarga de registrar una serie de parámetros que son
necesarios
   para generar un reporte en la tabla de reporte */
create proc prepRep ( @control tinyint ,@sesion tinyint,@param char(20)) as
  insert into reportes (control,sesion,parametro)
values(@control,@sesion,@param)
return
go

print "Procedimiento registraBaja"
go
/* Este procedimiento registra en la tabla de bajasAlumnos a los alumnos que
solicitaron su
   baja voluntaria */
CREATE PROC registraBaja (@numCta char (8), @cveCarrera char(4),
  @fecha fecha) as
  insert into bajasAlumnos values (@cveCarrera,@numCta,@fecha)
return
go

print "Procedimiento registraCarrSimult"
go
/* Este procedimiento registra en la tabla de carreraSim al alumno que está
cursando una
    carrera simultánea */
CREATE PROC registraCarrSimult (@carrera char (4), @cveCarrera char(4),
  @numCta char(8)) as
  insert into carreraSim values (@carrera,@cveCarrera,@numCta)
return
go

print "Procedimiento registraReposCreden"
go
/*Este procedimiento registra una reposición de credencial */
CREATE PROC registraReposCreden (@numCta char(8), @carrera char(4),
  @fecha fecha) as
```

```
 update estudia set reposicion = reposicion + 1
   where noCta = @numCta and cveCarr = @carrera
 insert into solicitud values (4,@numCta,@fecha)
return
go

print "Procedimiento registraSegundaCarr"
go
/* Este procedimiento registra en la tabla de carreraSim al alumno que está
cursando una
   segunda carrera */
CREATE PROC registraSegundaCarr (@carrera char (4), @cveCarrera char(4),
 @numCta char(8)) as
 insert into segundaCarr values (@carrera,@cveCarrera,@numCta)
return
go

print "Procedimiento serAcredita"
go
 /* Este procedimiento verifica la seriacion de al menos 1 y hasta 7 materias
    con los datos de entrada: Clave de Carrera,
    Numero de Cuenta, Materia1 [Materia2 ...] y devuelve las asignaturas
    requisito APROBADAS, incluyendo las materias mismas */
CREATE PROC serAcredita (@Carrera char(4), @Cuenta char(8), @Cve1
char(4),
 @Cve2 char(4)=NULL, @Cve3 char(4)=NULL, @Cve4 char(4)=NULL,
 @Cve5 char(4)=NULL, @Cve6 char(4)=NULL, @Cve7 char(4)=NULL) AS
 SELECT DISTINCT cveAsig,calificacion,periodo,tipoExamen FROM curso
WHERE
 calificacion IN ("MB","B ","S ","AC","RE","CO","10","9 ","8 ","7 ","6 ")
 AND (cveAsig IN (SELECT DISTINCT cveAsigReq FROM seriacion WHERE
cveAsig
 IN ( @Cve1,@Cve2,@Cve3,@Cve4,@Cve5,@Cve6,@Cve7) AND
cveCarr=@Carrera) OR
 cveAsig IN (@Cve1,@Cve2,@Cve3,@Cve4,@Cve5,@Cve6,@Cve7)) AND
 cveCarr = @Carrera AND noCta = @Cuenta
return
go
```

```
print "Procedimiento serMateria "
go
/* Este procedimiento verifica la seriación de una asignatura */
create proc serMateria (@cveAsig char(4), @cveCarr char(4) ) as
 select cuantas=count(*) from seriacion where cveCarr = @cveCarr
 and cveAsig = @cveAsig
return
go

print "Procedimieto serPasadas "
go
/* Este procedimiento regresa la lista de las asignaturas que un alumno ya
acreditó */
create proc serPasadas (@cveAsig char(4), @cveCarr char(4),@noCta char(8))
as
 select cuantos=count(*) from seriacion s, curso c where
  s.cveAsig = @cveAsig and s.cveAsigReq = c.cveAsig and
  c.calificacion in ("MB","B ","S ") and c.noCta = @noCta and
  c.cveCarr = @cveCarr and s.cveCarr = @cveCarr
return
go

print "Procedimiento serRequisito"
go
 /* Este procedimiento verifica la seriacion de al menos 1 y hasta 7 materias
    con los datos de entrada: Clave de Carrera, Materia1 [Materia2 ...]
    y devuelve las asignaturas y las asignaturas requisito. */
CREATE PROC serRequisito (@Carrera char(4), @Cve1 char(4), @Cve2
char(4)=NULL,
 @Cve3 char(4)=NULL, @Cve4 char(4)=NULL, @Cve5 char(4)=NULL,
 @Cve6 char(4)=NULL, @Cve7 char(4)=NULL) AS
 SELECT DISTINCT cveAsig,cveAsigReq FROM seriacion WHERE cveAsig
IN
 ( @Cve1,@Cve2,@Cve3,@Cve4,@Cve5,@Cve6,@Cve7) AND
cveCarr=@Carrera
return
go
```

```
print "Procedimiento serTodas"
go
 /* Este procedimiento verifica la seriacion de al menos 1 y hasta 7 materias
     con los datos de entrada:
     Clave de Carrera, Numero de Cuenta, Materia1 [Materia2 ...]
     y devuelve las asignaturas y asignaturas requisito NO APROBADAS */
CREATE PROC serTodas (@Carrera char(4), @Cuenta char(8), @Cve1
char(4),
 @Cve2 char(4)=NULL, @Cve3 char(4)=NULL, @Cve4 char(4)=NULL,
 @Cve5 char(4)=NULL, @Cve6 char(4)=NULL, @Cve7 char(4)=NULL) AS
 SELECT cveAsig,cveAsigReq FROM seriacion WHERE cveAsig IN
 ( @Cve1, @Cve2, @Cve3, @Cve4, @Cve5, @Cve6, @Cve7 ) AND
cveAsigReq NOT IN
 (SELECT cveAsig FROM curso WHERE calificacion IN
 ("MB","B ","S ","AC","RE","CO","10","9 ","8 ","7 ","6 ") AND cveAsig IN
 (SELECT DISTINCT cveAsigReq FROM seriacion WHERE cveAsig IN
 ( @Cve1, @Cve2, @Cve3, @Cve4, @Cve5, @Cve6, @Cve7 ) AND cveCarr =
@Carrera)
  AND cveCarr = @Carrera AND noCta = @Cuenta ) AND cveCarr =
@Carrera
 ORDER BY cveAsig,cveAsigReq
return
go

print "Procedimiento soliCarrSimult"
go
/* Este procedimiento regresa los datos generales de un alumno que está
solicitando
   carrera simultánea */
CREATE PROC solicCarrSimult (@numCta char(8)) as
 select name=estudia.nombre, promedio, estudia.cveCarr, carrera.nombre,
   credCursados=creditosOblig + creditosOpta,
   credCarr=credObliga + credOpta, sistema
   from estudia, carrera
   where noCta = @numCta and carrera.cveCarr = estudia.cveCarr
return
go
```

```
print "Procedimiento solicSegCarr"
go
/* Este procedimiento regresa los datos generales de un alumno que está
solicitando
   segunda carrera */
CREATE PROC solicSegCarr (@numCta char(8)) as
 select name=exAlumno.nombre, exAlumno.promedio, exAlumno.cveCarr,
   carrera.nombre
   from exAlumno, carrera where exAlumno.noCta = @numCta
   and carrera.cveCarr = exAlumno.cveCarr
return
go

print "Procedimiento verificaBaja"
go
  /* Este procedimiento regresa el registro de un alumno dado de baja en
     caso de que exista en la tabla e bajasAlumno*/
CREATE PROC verificaBaja (@carrera char(4), @numCta char(8)) as
 select * from bajasAlumnos where cveCarr = @carrera and noCta=@numCta
return
go
```

TRANSACCIONES

Una transacción es un conjunto de una o más clausulas SQL que son consideradas como un solo evento.

Todos las modificaciones de los datos que se realizan en una transacción pueden ser controladas, de tal manera que se realicen completamente o no.

Cuando ejecutamos una transacción ningún proceso puede ver los cambios que se realizaron hasta que se ejecuta el commit de la misma ya que existe la posibilidad de que se cancele la transacción a través de una instrucción de roll back.

A continuación presentamos el escript para crear las transacciones que forman parte de la base de datos del sistema.

```
print "Procedimiento borraAsig"
go
/*Este procedimiento cancela la reinscripción de un alumno */
CREATE PROCEDURE borraAsig (@NumCta char(8), @Carrera char(4),
  @Cve1 char(4)=NULL, @Gpo1 char(4)=NULL, @Cve2 char(4)=NULL,
  @Gpo2 char(4)=NULL, @Cve3 char(4)=NULL, @Gpo3 char(4)=NULL,
  @Cve4 char(4)=NULL, @Gpo4 char(4)=NULL, @Cve5 char(4)=NULL,
  @Gpo5 char(4)=NULL, @Cve6 char(4)=NULL, @Gpo6 char(4)=NULL,
  @Cve7 char(4)=NULL, @Gpo7 char(4)=NULL) AS
  BEGIN TRANSACTION
    delete cursa  /* Borra la asig. de la tabla de cursa */
     where cveAsig+gpo in
(@Cve1+@Gpo1,@Cve2+@Gpo2,@Cve3+@Gpo3,
     @Cve4+@Gpo4,@Cve5+@Gpo5,@Cve6+@Gpo6,@Cve7+@Gpo7)
     and noCta=@NumCta and cveCarr=@Carrera
```

```
    update clase set vacantes = vacantes+1 where cveAsig+grupo in
     /* Aumenta en 1 el número de vacantes de esa asig. */
     (@Cve1+@Gpo1,@Cve2+@Gpo2,@Cve3+@Gpo3,
     @Cve4+@Gpo4,@Cve5+@Gpo5,@Cve6+@Gpo6,@Cve7+@Gpo7)
     and cveCarr=@Carrera
    select NumRegistros=@@rowcount
  COMMIT TRANSACTION
RETURN
go


print "Procedimiento inscribe"
go
/* Registra asig. y grupos que el alumno desea cursar */
CREATE PROCEDURE inscribe (@NumCta char(8), @Carrera char(4), @sem
int,
  @Cuota float, @Cve1 char(4)=NULL, @Gpo1 char(4)=NULL, @Cve2
char(4)=NULL,
  @Gpo2 char(4)=NULL, @Cve3 char(4)=NULL, @Gpo3 char(4)=NULL,
@Cve4 char(4)=NULL,
  @Gpo4 char(4)=NULL, @Cve5 char(4)=NULL, @Gpo5 char(4)=NULL,
@Cve6 char(4)=NULL,
  @Gpo6 char(4)=NULL, @Cve7 char(4)=NULL, @Gpo7 char(4)=NULL) AS
  /* Se reciben como parámetros 7 asig. y 7 gpos, de las cuales puede haber
nulos */
  declare @UltimoSemestre tinyint
  BEGIN TRANSACTION
    /* Bloquea los registros, de la tabla clase, de las asig. que el alumno desea
cursar */
    SELECT cveAsig, grupo FROM clase HOLDLOCK WHERE
cveCarr=@Carrera AND
      cveAsig+grupo IN (SELECT cveAsig+grupo FROM clase WHERE
cveCarr=@Carrera
      AND cveAsig+grupo IN
(@Cve1+@Gpo1,@Cve2+@Gpo2,@Cve3+@Gpo3,@Cve4+@Gpo4,
      @Cve5+@Gpo5,@Cve6+@Gpo6,@Cve7+@Gpo7) AND vacantes = 0 )
    UPDATE clase SET vacantes=vacantes-1 WHERE cveCarr=@Carrera AND
```

```
    /* Disminuye el número de vacantes de las asig. */
    cveAsig+grupo IN (SELECT cveAsig+grupo FROM clase WHERE
    cveCarr=@Carrera AND cveAsig+grupo IN
    (@Cve1+@Gpo1,@Cve2+@Gpo2,@Cve3+@Gpo3,@Cve4+@Gpo4,
    @Cve5+@Gpo5,@Cve6+@Gpo6,@Cve7+@Gpo7) AND vacantes > 0 )
  IF @Cve1!='' and @Gpo1!=''
    /* Valida que el primer parámetro no sea nulo y registra la asig.
       en la tabla de cursa */
    insert into cursa values (@Cve1,@Carrera,@Gpo1,@NumCta)
  IF @Cve2!='' and @Gpo2!=''
    insert into cursa values (@Cve2,@Carrera,@Gpo2,@NumCta)
  IF @Cve3!='' and @Gpo3!=''
    insert into cursa values (@Cve3,@Carrera,@Gpo3,@NumCta)
  IF @Cve4!='' and @Gpo4!=''
    insert into cursa values (@Cve4,@Carrera,@Gpo4,@NumCta)
  IF @Cve5!='' and @Gpo5!=''
    insert into cursa values (@Cve5,@Carrera,@Gpo5,@NumCta)
  IF @Cve6!='' and @Gpo6!=''
    insert into cursa values (@Cve6,@Carrera,@Gpo6,@NumCta)
  IF @Cve7!='' and @Gpo7!=''
    insert into cursa values (@Cve7,@Carrera,@Gpo7,@NumCta)
  DELETE cursa
    /* Borra las asignaturas que el alumno desea dar de baja, después de
       que ya habían sido registradas */
    WHERE cveAsig IN (SELECT cveAsig FROM clase WHERE
    cveCarr=@Carrera AND cveAsig+grupo IN (SELECT cveAsig+grupo
FROM clase
    WHERE cveCarr=@Carrera AND cveAsig+grupo IN
(@Cve1+@Gpo1,@Cve2+@Gpo2,
    @Cve3+@Gpo3,@Cve4+@Gpo4,
@Cve5+@Gpo5,@Cve6+@Gpo6,@Cve7+@Gpo7)
    AND vacantes = 0 ) )
  if exists (select * from cursa where cveCarr=@Carrera AND
noCta=@NumCta)
    UPDATE estudia
      /* Actualiza la tabla de estudia marcando con una "S" el campo
"inscrito" */
```

```
        SET inscrito="S", cuotaVol=@Cuota,semestreAnterior=semestre,
semestre=@sem
        WHERE cveCarr=@Carrera AND noCta=@NumCta
    else
      begin
        UPDATE estudia
          SET semestre=semestreAnterior, inscrito="N", cuotaVol=0.0 where
          cveCarr=@Carrera AND noCta=@NumCta
      end
    COMMIT TRANSACTION
RETURN
go


print "Procedimiento actualizaTablas"
go
/* Prepara las tablas después de cada semestre, limpiando algunas y actualizando
otras */
create proc actualizaTablas as
begin tran a
  delete acta
commit tran a
begin tran a
  delete bajasAlumnos
commit tran a
begin tran a
  delete sancion where tipo="E"
commit tran a
begin tran a
  delete camCiclo
commit tran a
begin tran a
  delete cambioCarrera
commit tran a
begin tran a
  delete cambioGpo
commit tran a
```

```
begin tran a
  delete cambioPlantel
commit tran a
begin tran a
  delete credencial
commit tran a
begin tran a
  delete cursa
commit tran a
begin tran a
  delete examenPro
commit tran a
begin tran a
  delete examenProObj
commit tran a
begin tran a
  delete examenProOral
commit tran a
begin tran a
  delete pagoSinodal
commit tran a
begin tran a
  delete preAlumno
commit tran
begin tran a
  delete primerIngreso
commit tran a
begin tran a
  delete solicitud
commit tran a
declare @Per char(3),@Gen int
select @Per=periodo from periodo where fechaIni = (select max(fechaIni)
  from periodo where substring(periodo,3,1) like "[12]")
if @Per=NULL return
select @Gen=convert(int,substring(@Per,1,2))-2
select @Gen=@Gen*10
  begin tran a
    delete inscrito where convert(int,periodo) < @Gen
```

```
  commit tran a
  begin tran a
    delete soliMas2 where convert(int,periodo) < @Gen
  commit tran a
  begin tran a
    delete solicitudMas2 where convert(int,periodo) < @Gen
  commit tran a
  begin tran a
    update estudia set inscrito="N"
  commit tran a
return
go


print "Procedimiento actEstAlu "
go
/* Actualiza la tabla de estudia en los campos de promedio, credOblig,
   credOpta, reprobadas, y ultimoPeriodo que el alumno cursó*/
create proc actEstAlu(@Cuenta char(8),@Carr char(4)=NULL) as
  declare @ap tinyint,@tco smallint,@tcp smallint,@tcr smallint,
  @noap tinyint, @pro float,@Ult char(3)
  if @Carr=NULL select @ap=(select count(distinct cveCarr) from estudia
    where noCta=@Cuenta ),@Carr=cveCarr  from estudia where
noCta=@Cuenta
  else select @ap=count(distinct cveCarr) from estudia where noCta=@Cuenta
    and cveCarr=@Carr
  if @ap = 0
    begin
      print "%1! de %2! no existe",@Cuenta,@Carr
      return
    end
  if @ap = NULL
    begin
      print "%1! no existe",@Cuenta
      return
    end
```

```
if (@ap > 1)
  begin
    print "Especifica carrera de %1!",@Cuenta
    return
  end
create table #matAp(cveCarr char(4),cveAsig char(4),c tinyint,t char(1))
  begin tran a
    insert into #matAp
    select distinct c.cveCarr,c.cveAsig,c=creditos,t=tipo
      from curso c, asignatura a where noCta=@Cuenta and calificacion in
      ("MB","B","S","AC","RE","CO","10","9","8","7","6") and
      c.cveCarr=@Carr and c.cveCarr+c.cveAsig=a.cveCarr+a.cveAsig
    select @ap=@@rowcount
  commit tran a
select @noap=count(distinct c.cveCarr+c.cveAsig) from curso c,asignatura
  a where noCta=@Cuenta and c.cveCarr=@Carr and c.cveCarr+c.cveAsig=
  a.cveCarr+a.cveAsig and c.cveCarr+c.cveAsig not in
  (select cveCarr+cveAsig from #matAp)
if @noap = NULL select @noap = 0
  select @Ult=convert(char(3),max(convert(int,periodo))) from curso c where
  noCta=@Cuenta and c.cveCarr=@Carr
if @Ult = NULL select @Ult=periodo from periodo where fechaIni=(select
  max(fechaIni) from periodo where periodo like "__[12]")
if @ap=0
  begin
    update estudia set aprobadas=0, creditosOblig=0,
    creditosOpta=0, creditosRev=0, promedio=0.0, reprobadas=@noap ,
    ultimoPeriodo=@Ult where noCta=@Cuenta and cveCarr=@Carr
    drop table #matAp return
  end
create table #matCal(cveCarr char(4),cveAsig char(4),c tinyint,t char(1),
callet char(2),calnum float)
  begin tran b
    insert into #matCal
    select m.cveCarr, m.cveAsig, c, t,callet=(select calificacion from
      curso c where noCta=@Cuenta and c.cveCarr=@Carr and
      c.cveCarr+c.cveAsig=m.cveCarr+m.cveAsig and calificacion in
      ("MB","B","S","AC","RE","CO","10","9","8","7","6")),calnum=10.0
```

```
        from #matAp m
    commit tran b
  drop table #matAp
    begin tran c
      update #matCal set calnum=0.0 where callet not in ("MB","10")
      update #matCal set calnum=8.0 where callet="B"
      update #matCal set calnum=6.0 where callet="S"
      update #matCal set calnum=convert(float,callet) where callet in
        ("9","8","7","6")
    commit tran c
  select @tco=isnull((select sum(c) from #matCal where t="O"),0)
  select @tcp=isnull((select sum(c) from #matCal where t="P"),0)
  select @tcr=isnull((select sum(c) from #matCal where calnum=0.0),0)
  delete #matCal where calnum=0.0
  select @pro = avg(calnum) from #matCal
  if @pro=NULL select @pro=0.0
    drop table #matCal
  update estudia set aprobadas=@ap,creditosOblig=@tco, creditosOpta=@tcp,
    creditosRev=@tcr, promedio=@pro,reprobadas=@noap,
ultimoPeriodo=@Ult
    where noCta=@Cuenta and cveCarr=@Carr
return
go


print "Procedimiento prepEst14"
go
/* Prepara la estadística de Cuotas Voluntarias de Colegiatura */
create proc prepEst14 as
  declare @Cve char(2),@Per char(3),@Cual tinyint,@Sem tinyint,@Imp real
  select @Per=periodo from periodo where fechaIni = (select max(fechaIni)
    from periodo where substring(periodo,3,1) like "[12]")
  select clave=substring(cveCarr,1,2),cuotaVol,semestre into #alumnos
    from estudia where inscrito="S"
  if not exists( select * from #alumnos ) return
  select @Cual = 1
  while @Cual < 7
    begin
```

```
if @Cual = 1 select @Cve="03"
if @Cual = 2 select @Cve="09"
if @Cual = 3 select @Cve="12"
if @Cual = 4 select @Cve="14"
if @Cual = 5 select @Cve="19"
if @Cual = 6 select @Cve="42"
select @Sem = 11
select @Imp=sum(cuotaVol) from primerIngreso where inscrito="S" and
  substring(cveCarr,1,2) = @Cve
if @Imp = NULL select @Imp = 0.0
begin tran a
   if exists(select * from _estadistica14 where clave = @Cve and
      periodo = @Per and semestre = @Sem )
      update _estadistica14 set importe = @Imp where clave = @Cve and
        periodo = @Per and semestre = @Sem
   else
      insert into _estadistica14 values ( @Cve, @Per, @Sem, @Imp )
commit tran a
select @Sem = 12
select @Imp = sum(cuotaVol) from #alumnos where clave = @Cve
    and semestre = 1
if @Imp = NULL select @Imp = 0.0
  begin tran b
    if exists(select * from _estadistica14 where clave = @Cve and
    periodo = @Per and semestre = @Sem )
    update _estadistica14 set importe = @Imp where clave = @Cve and
      periodo = @Per and semestre = @Sem
else
  insert into _estadistica14 values ( @Cve, @Per, @Sem, @Imp )
 commit tran b
   select @Sem = 2
   while @Sem < 11
     begin
       select @Imp = sum(cuotaVol) from #alumnos where
       clave = @Cve and semestre = @Sem
       if @Imp = NULL select @Imp = 0.0
         begin tran c
```

```
                if exists(select * from _estadistica14 where clave = @Cve and
                  periodo = @Per and semestre = @Sem )
                  update _estadistica14 set importe = @Imp where clave =
                  @Cve and periodo = @Per and semestre = @Sem
                else
                  insert into _estadistica14 values ( @Cve, @Per, @Sem,
                  @Imp )
              commit tran c
            select @Sem=@Sem+1
          end
        select @Cual=@Cual+1
    end
return
go


print "Procedimiento prepEst15"
go
/* Prepara la estadística de Inscripción y Reinscripción (Cupos y Grupos) */
create proc prepEst15 as
  declare @Per char(3)
  select @Per=periodo from periodo where fechaIni = (select max(fechaIni)
    from periodo where substring(periodo,3,1) like "[12]")
  if exists (select * from _estadistica15 where periodo=@Per) and
    not exists(select * from clase)
    return
  begin tran a
    if exists (select * from _estadistica15 where periodo=@Per) and
      exists(select * from clase)
      delete _estadistica15 where periodo=@Per
    commit tran a
      insert into _estadistica15
      select periodo=@Per,clave=substring(cveCarr,1,2),grupo,cveAsig,
        inscritos=cupo-vacantes from clase where substring(cveCarr,1,2)
        in ("03","09","12","14","19","42")
return
go
```

```
print "Procedimiento prepEst17"
go
/* PRepara la estadística de Pago de Laboratorio */
create proc prepEst17 as
  declare @Cve char(2),@Per char(3),@Cual tinyint,@Sem tinyint,@Imp real,
    @Tot int
  select @Per=periodo from periodo where fechaIni = (select max(fechaIni)
    from periodo where substring(periodo,3,1) like "[12]")
  select clave=substring(cveCarr,1,2),semestre,noCta into #alumnos
    from estudia where inscrito="S"
  if not exists( select * from #alumnos ) return
  select @Cual = 1
  while @Cual < 7
    begin
      if @Cual = 1 select @Cve="03"
      if @Cual = 2 select @Cve="09"
      if @Cual = 3 select @Cve="12"
      if @Cual = 4 select @Cve="14"
      if @Cual = 5 select @Cve="19"
      if @Cual = 6 select @Cve="42"
      select @Sem = 1
      while @Sem < 11
        begin
          select @Tot=(select count(noCta) from #alumnos where
            clave = @Cve and semestre = @Sem )
          if @Tot = NULL select @Tot = 0
          select @Imp = (select sum(importe) from pagoLab where clave=@Cve
             and semestre=@Sem ) * @Tot
          if @Imp = NULL select @Imp = 0.0
          begin tran a
            if exists(select * from _estadistica17 where clave = @Cve and
              periodo = @Per and semestre = @Sem )
              update _estadistica17 set importe=@Imp, numAlumnos=@Tot
                where clave=@Cve and periodo=@Per and semestre=@Sem
            else
              insert into _estadistica17 values (@Cve,@Per,@Sem,@Tot,@Imp)
              commit tran a
```

```
                select @Sem=@Sem+1
        end
       select @Cual=@Cual+1
    end
return
go


print "Procedimiento prepPagSin"
go
/* Prepara relación de Pago a Sinodales */
create proc prepPagSin (@Per1 char(3)) as
  declare @RFC char(13),@Grupo char(12),@total int,@Alu1 int,@Alu2 int
  set rowcount 0
  delete pagoSinodal
  select * into #sin from sinodal
  select distinct s.rfc,nombre into #pro from sinodal s, profesor p
    where s.rfc=p.rfc
  set rowcount 1
  while (select count(*) from #pro) > 0
    begin
      begin tran a
        select @RFC = rfc from #pro
        select @Alu1 = 0
        select @Alu2 = 0
        while (select count(*) from #sin where rfc=@RFC ) > 0
          begin
            begin tran b
              select @Grupo=cveCarr+cveAsig+gpo from #sin where rfc=@RFC
              select @total=count(rfc) from #sin
                where cveCarr+cveAsig+gpo=@Grupo
              if @total = 1
                select @Alu1=@Alu1+count(noCta) from inscrito
                where cveCarr+cveAsig+gpo = @Grupo  and
                convert(int,periodo) = convert(int,@Per1)
              else
                select @Alu2=@Alu2+count(noCta) from inscrito
                where cveCarr+cveAsig+gpo = @Grupo  and
```

```
            convert(int,periodo) = convert(int,@Per1)
          delete #sin where rfc=@RFC and cveCarr+cveAsig+gpo=@Grupo
        commit tran b
      end
    insert pagoSinodal values(@RFC,@Alu1,@Alu2)
    print " Valores RFC %1! Total1 %2!  Total2 %3!",@RFC,@Alu1,@Alu2
    delete #pro where rfc=@RFC
    commit tran a
  end
 set rowcount 0
return
go
```

CONCLUSIONES

La implantación de este sistema permitirá a la Unidad de Administración Escolar realizar de una manera más fácil y rápida sus actividades lo que se traduce en un mejor servicio para la comunidad estudiantil.

El contar con un registro de todos los trámites que se llevan a cabo en la Unidad de Administración Escolar permitirá la emisión de documentos oficiales y estadísticas en forma automática apoyando a las autoridades en la toma de decisiones.

El nuevo sistema permitirá el ahorro de tiempo y recursos facilitando los trámites escolares.

Este sistema constituye un primer paso en la descentralización de la Administración Escolar de la UNAM, mismo que traerá consigo que cada entidad académica sea autónoma en el control de su información.

Asimismo, sentará las bases para lograr una automatización integral permitiendo más adelante que el mismo alumno realice sus trámites desde terminales que se encuentren en su escuela, o bien, vía telefónica.

En una primera instancia el costo del sistema puede ser elevado debido a la tecnología bajo la cual está desarrollado pero garantiza que no se volverá obsoleto tan rápidamente y facilita la realización de un sistema integral en un futuro gracias a que es una arquitectura abierta. Por otro lado, si bien es cierto que estos costos se pueden disminuir utilizando un servidor PC 486 que incluya la versión de UNIX correspondiente, no resulta ventajoso un ahorro de este tipo ya que no podemos comparar la seguridad y rapidez de un equipo SUN con la vulnerabilidad de una PC. Si consideramos la importancia de la información de administración escolar y lo delicado que puede ser la alteración y pérdida de la misma debemos preocuparnos por utilizar los avances tecnológicos y las ventajas que nos presentan, procurando así, utilizar plataformas más robustas que permitirán una conectividad con otros sistemas y equipos, logrando con ésto una automatización integral.

El utilizar un manejador de bases de datos relacional como Sybase nos permite tener una mayor seguridad en los accesos a la información y en la integridad de la misma.

Gracias al ambiente gráfico de Visual Basic es posible desarrollar pantallas muy amigables que facilitan el aprendizaje y uso del sistema.

Como ya se mencionó anteriormente el sistema está desarrollado bajo una arquitectura cliente/servidor que reduce el tráfico en la red debido a que se distribuye la carga de trabajo entre el cliente y el servidor, lo que trae consigo un mejor tiempo de respuesta.

Es importante mencionar que el equipo y software necesarios para el desarrollo de ADESI pueden ser utilizados, además para otros sistemas de información dentro de la ENEP Iztacala como por ejemplo Administración de Personal, Biblioteca, etc.

No podemos negar, que se han desarrollado diversos sistemas de administración escolar, la mayoría en plataformas de PCs, que incluyen los módulos principales como son inscripciones, reinscripciones y exámenes extraordinarios. Sin embargo, la propuesta presentada en este trabajo requirió de un análisis y diseño más detallado debido a la complejidad que tiene inmersa esta actividad de administración y a los módulos que incluye ADESI, que van desde los tradicionales ya mencionados, hasta titulación, seguro facultativo, sanciones, otros ingresos a la UNAM, etc.

Al incluir estos módulos en el sistema, pretendemos que además de facilitarle las funciones a la Unidad de Administración Escolar, se la facilitemos al alumno y que sea uno de los pequeños pasos para que en un tiempo no muy lejano el alumno, desde su casa, realice sus trámites de reinscripción, exámenes extraordinarios, revisión de estudios, titulación, servicio social, etc., siempre y cuando se cuente con el equipo adicional necesario conectado a la base de datos de ADESI.

BIBLIOGRAFÍA

[AGE93] Agenda Estadística
Dirección General de Estadística y Sistemas de Información Institucional
UNAM
México, D.F, 1993

[DAT86] Date, C.J.
"Introducción a los Sistemas de Bases de Datos"
Addison Wesley Iberoamericana
Tercera edición
Wilmington, 1986

[DEW93] Dewire, Dawna Travis
"Client/Server Computing"
Mc Graw-Hill
New York, 1993

[ELB94] Elbert, Bruce y Martyna, Bobby
"Client/Server Computing Architecture, Applications and Ditributed Systems Management"
Artech House
Boston, 1994

[GUI91] Guía de la Universidad
UNAM
México, D.F. 1991

[HAN92] Hansen, Gary W. y Hansen, James V.
"Database Management and Design"
Prentice-Hall
Englewood Cliffs, 1992

[KOR86] Korth, Henry F. y Silberschatz, Abraham
"Data Base System Concepts"
Mc Graw Hill Advanced Computer Science Series
New York, 1986

[MCG93] McGoveran, D. y Date, C.J.
"A Guide to Sybase and SQL Server"
Addison-Wesley
New York, 1993

[ORF94] Orfali, Robert y Harkey, Dan
"Essential Client/Server Survival Guide"
Van Nostrand Reinhold
New York, 1994

[PRE93] Pressman, Roger S.
"Ingeniería del Software un Enfoque Práctico"
Mc Graw Hill
Tercera Edición
Madrid, 1993

[SAL93] Salemi, Joe
"Guide to Client/Server Databases"
Ziff-Davis Press
Emeryville, 1993

[TOW92] Townsend, James J.
"Introduction to Databases"
QUE
Carmel, 1992

[YOR93] Yourdon, Edward
"Análisis Estructurado Moderno"
Prentice Hall
México, 1993

GLOSARIO

Administración Escolar. Organización que se encarga de manejar los asuntos relacionados con los alumnos inscritos a la institución educativa.

Atributo. Básicamente es una propiedad descriptiva de una entidad o una relación.

Back-End. Es el software que contiene el servidor de la base de datos, capaz de resolver las peticiones de los clientes. Se desarrolla con un manejador de base de datos y es quién administra la información.

Base de Datos. Colección de datos interrelacionados y almacenados en conjunto cuya finalidad es la de servir a una aplicación o más de la mejor manera. Los datos son almacenados de tal forma que resulten independientes de los programas de las aplicaciones.

Cliente. El cliente representa la interfaz con el usuario y es quien realiza las peticiones al servidor. Puede hacer uso de varios sistemas operativos incluyendo DOS, OS/2, UNIX o Apple Macintosh.

Dato. Unidad mínima de información que es susceptible de una observación sin significado.

DBMS(DataBase Managment System). Sistema manejador de base de datos que está formado por un conjunto de rutinas de software interelacionadas donde cada una de estas es responsable de una tarea determinada.

Dead Lock. Nombre que recibe el conflicto que se produce cuando dos o más procesos poseen reservados (bloqueados) datos que otro procesos necesitan.

Default. Especifica el dato que será almacenado en la columna de una tabla cuando el usuario no proporciona ningún valor.

Diagrama de Contexto. Es un caso especial de diagramas de flujo de datos; es un modelo en el que se muestra un proceso que representa al sistema, los terminators que representan a las entidades con que el sistema tiene relación y a los metaflujos que unen al sistema con las entidades.

Diagrama de Entidad Relación. Es una herramienta de modelado que tiene el propósito de representar los objetos de datos y sus relaciones. Los objetos de datos se representan con un rectángulo etiquetado y las relaciones mediante líneas que conectan a las entidades relacionadas.

Diagrama de Estructura (DE). Es una herramienta de diseño que modela a los procesos como una jerarquía de módulos y describe las conexiones de datos y control entre estos.

Diagrama de Flujo de Datos (DFD). Es una herramienta empleada para la elaboración de modelos que muestra un sistema como una red de procesos conectados entre sí por flujos.

Diagrama de Procesador. Es una herramienta que modela la elección de los procesadores y las interfaces entre ellos, en él se puede evaluar la elección de la tecnología de procesamiento elegida.

Diagrama de Transición de Estado. Se utilizan para representar los diferentes estados que se van presentando durante la ejecución de un proceso. Un estado es cualquier modo de comportamiento observable del sistema.

Entidad. Son todos aquellos organismos, instituciones, dependencias, etc., que están relacionados con el sistema.

Front-End. Es la aplicación que se encuentra en los clientes, puede ser un manejador de base de datos, una hoja de cálculo, un procesador de palabras o cualquier otro tipo de aplicación. Sus funciones básicas son la presentación y validación de datos y realización de peticiones al servidor.

Índice. Es una estructura almacenada físicamente cuyo objetivo es el de cuidar las llaves primarias y realizar las búsquedas más rápido.

Integridad Referencial. Es el mecanismo que consiste en mantener una relación completa entre las llaves primarias y las llaves foráneas de las tablas que forman una base de datos.

Iteración. Es una característica del lenguaje estructurado y se refiere a que las instrucciones de un proceso se llevan a cabo varias veces pero con algún límite.

Lisa de Eventos. Es una narración de los estímulos que ocurren en el ambiente exterior que hacen que el sistema responda.

Llave. Es el campo de una tabla utilizado para identificar a los registros, también, una llave es usada para ordenar los registros en un archivos secuencial.

Middleware. Término que cubre todo el software necesario para soportar interacciones entre clientes y servidores.

Miniespecificación. Es una explicación de lo que sucede en cada proceso primitivo de nivel más bajo en un diagrama de flujo de datos.

Modelo. Representación abstracta de la realidad.

Módulo. Es un grupo de instrucciones utilizado como unidad que al ser invocado realiza una función determinada.

Multihilos. Es la capacidad de que una misma aplicación pueda tener la característica de multitarea.

Multitareas. Se refiere a poder ejecutar numerosas aplicaciones concurrentemente.

Multiusuarios. Poder soportar el trabajo de numerosos y simultáneos usuarios realizando diferentes tareas.

Password. Medida de seguridad que sólo permite el acceso al software y a los datos a personal autorizado.

Procedimiento Almacenado. Es un conjunto de instrucciones SQL almacenadas en la base de datos que pueden ser ejecutadas por su nombre.

Prototipo. Es un modelo del sistema que permite conocer las necesidades del usuario, plasmándolas en interfaces del sistema.

Sistema. Es una colección de actividades y de información almacenada que está organizada para realizar un propósito específico y que responden a eventos en su ambiente.

SQL (Structured Query Language). Es, desde la década de los 80's, el lenguaje de programación estándar de los manejadores de bases de datos.

Tabla. Es una estructura de datos que posee un encabezado y un cuerpo, el encabezado es un conjunto determinado de columnas y tipos de datos, el cuerpo es un conjunto de renglones de valor cambiante y cuyo tipo es igual al definido para cada columna en el encabezado.

Topología. Es la distribución geométrica y física de una red de cómputo.

Transacción. Es un conjunto de instrucciones SQL que son tratadas como una sola, es decir, o toman efecto todas o no toma efecto ninguna.

Trigger. Es un tipo especial de procedimientos almacenados cuyo objetivo es conservar la integridad de los datos.

Usuario. Persona que va a hacer uso del sistema, conoce los procesos que se realizan. Es la fuente de información.

RDBMS(Relational DataBase Managment System). Sistema manejador de base de datos que está formado por un conjunto de rutinas de software interelacionadas donde cada una de estas es responsable de una tarea determinada y que almacena los datos bajo un esquema relacional.

Regla. Especifican una lista o un rango de valores que una columna de una tabla puede obtener.

Relación. Se utilizan en los Diagramas de Entidad Relación para establecer un vínculo entre una entidad y otra. Es la asociación entre 2 o más entidades.

Rollback. Es una instrucción que sirve para cancelar total o parcialmente las instrucciones que forman a una transacción.

Reporteador. Como su nombre lo indica, es un programa que permiten la elaboración de reportes con un ambiente muy amigable.

Secuencia. Es una característica del lenguaje estructurado y se refiere a que la programación de un proceso consisten de una o más piezas que se aplicarán una después de la otra sin interrupción.

Selección. Es una característica del lenguaje estructurado y se refiere a que la programación de un proceso se compone de construcciones sintácticas consistentes de dos o más piezas en las que solamente una de ellas aplica en un caso específico.

Servidor. Es el procesador que se encarga de la administración de la base de datos, recibe las peticiones de todos los clientes conectados a él y tiene por función básica resolver los requerimientos de estos. Controla el acceso a la información a través de claves de seguridad.